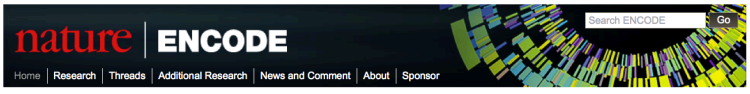
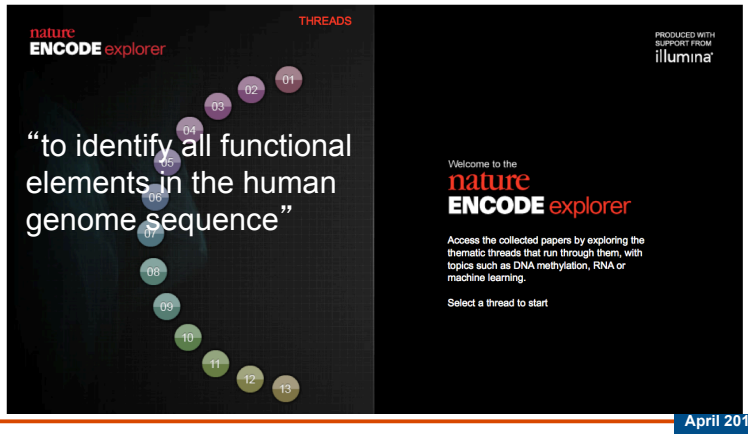
The Genome Access Course

Functional Genomic Elements From The ENCODE Project

The Genome Access Course

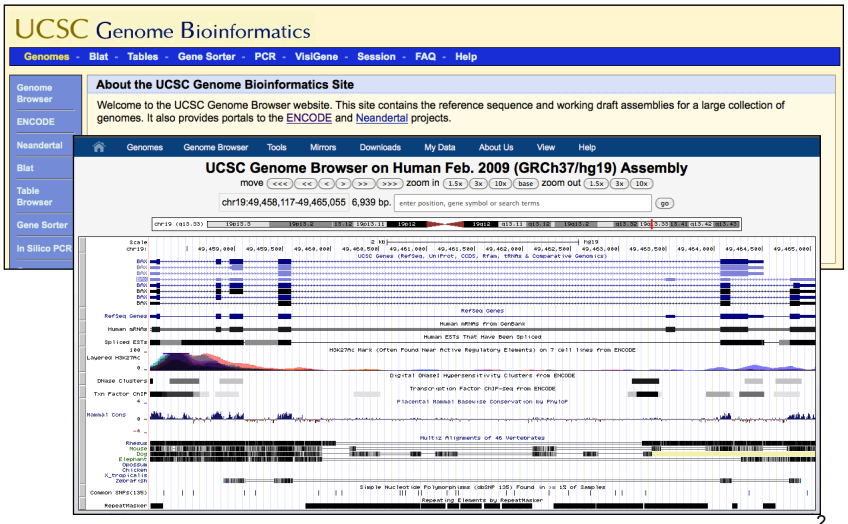
ENCODE – encyclopedia of DNA elements





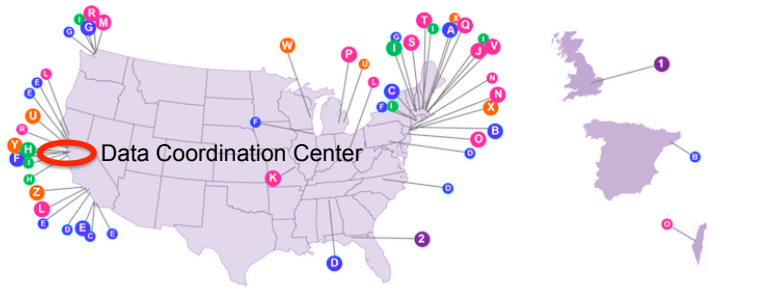
The Genome Access Course

UCSC Genome Browser



The Genome Access Course

Consortium Members



Production Groups
- A Broad Institute
- B Cold Spring Harbor; Centre for Genomic Regulation (CRG);
- C University of Connecticut Health Center; UCSD
- D HudsonAlpha; Pennsylvania State; UC Irvine; Duke; Caltech
- E UCSD; Salk Institute ; Joint Genome Institute;

Technology Development Groups
- J MIT
- K Washington University, St. Louis
- L USC; Ohio State University; UC, Davis
- M University of Washington
- N Sloan-Kettering; Weill Cornell Medical College
- O Princeton; Weizmann
- P University of Michigan
- Q Broad Institute

Affiliated Groups
- 1 Wellcome Trust Sanger Institute
- 2 Florida State University

www.encodeproject.org

The Genome Access Course

Some Historical Perspective

1990	2001	2003	2006	2007	2012
• Human Genome Project Launched	• Draft Human • Genome Published	• Complete Genome Sequence • **ENCODE project Launched by NHGRI**	• First Next-generation sequencers commercially sold	• ENCODE Production Phase	• ENCODE Project Data Release and multiple publications

Genome-wide *in situ* exon capture for selective resequencing

Emily Hodges[1,4], Zhenyu Xuan[1,2,4], Vivekanand Balija[2], Melissa Kramer[2], Michael N Molla[3], Steven W Smith[3], Christina M Middle[3], Matthew J Rodesch[3], Thomas J Albert[3], Gregory J Hannon[1] & W Richard McCombie[2]

the ENCODE pilot project

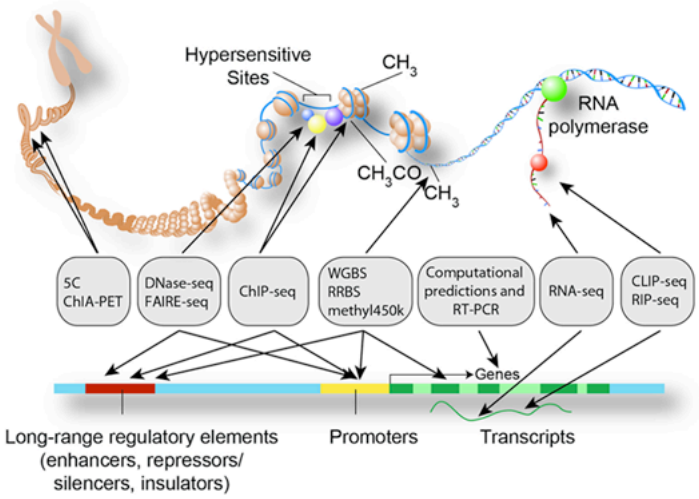
The ENCODE Project Consortium*

Vol 447(14): pp 799-814

Special issue: Vol 17, issue 6

The Genome Access Course

The ENCODE Project



5

April 2015

The Genome Access Course

ENCODE @ UCSC

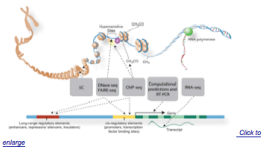
ENCODE

Encyclopedia of DNA Elements

General
Resources & FAQ
Publications
Software Tools
Data Standards
Human
Downloads
Experiment Matrix
Search
Genome Browser (hg19)
Integrative Analysis
Session Gallery
Experiment List

About ENCODE Data

The Encyclopedia of DNA Elements (ENCODE) Consortium is an international collaboration of research groups funded by the National Human Genome Research Institute (NHGRI). The goal of ENCODE is to build a comprehensive parts list of functional elements in the human genome, including elements that act at the protein and RNA levels, and regulatory elements that control cells and circumstances in which a gene is active.



Click to enlarge

ENCODE data are now available for the entire human genome. ***All ENCODE data are free and available for immediate use via*** :

- Search for displayable tracks and downloadable files
- Download of data files
- Visualization in the UCSC Genome Browser (ENCODE data marked with the NHGRI logo)
- Data mining with the UCSC Table Browser and other UCSC Genome Bioinformatics tools

To search for ENCODE data related to your area of interest and set up a browser view, use the UCSC Experiment Matrix or Track Search tool (*Advanced* features). The Experiment List (Human) and Experiment List (Mouse) links provide comprehensive listings of ENCODE data that is released or in preparation.

All ENCODE data is freely available for download and analysis. However, before publishing research that uses ENCODE data, please read the ENCODE Data Release Policy, which places some restrictions on publication use of data for nine months following data release. Read more about ENCODE data at UCSC.

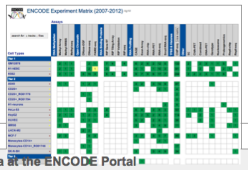
6

April 2015

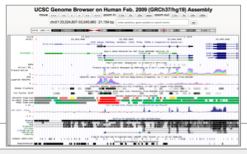
The Genome Access Course

ENCODE @ UCSC

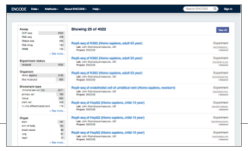
Explore ENCODE data at UCSC



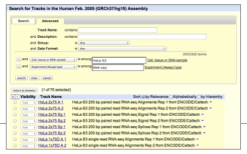
View ENCODE data in the UCSC Genome Browser



Search for data at the ENCODE Portal



Search for ENCODE tracks in the UCSC Browser



7

April 2015

The Genome Access Course

ENCODE Experiment Matrix (2007-2012) *hg19*

search for: tracks files

Assays

Cell Types	DNA Methylation	Methyl Array	Methyl RRBS	Methyl-seq	Open Chromatin	DNase-DGF	DNase-seq	FAIRE-seq	RNA Binding Proteins	RIP Gene ST	RIP Tiling Array	RIP Validation	RIP-seq	RNA Profiling	CAGE	Exon Array	RNA-chip	RNA-PET	RNA-seq	Small RNA-seq	TFBS & Histones	ChIP-seq	view matrix	Other	5C	ChIA-PET	Combined	DNA-PET	Genotype	Nucleosome	Proteogenomics	Repli-chip	Repli-seq
Tier 1																																	
GM12878	2	1	1				2	1		7	4		4		6	2	6	2	12	5		133			2		2	3	1	1	5		1
H1-hESC	2	1	1				2	1		3					4	1		1	10	3		91			1		2		1		2	1	
K562	2	1	1			3	16	3		6	4		4		9	7	9	6	17	7		244			2	2	2	3	1	1	6		1
Tier 2																																	
A549	1	1				1	2	1							3	2		3	10	9		87							1					
CD20+															1				2	1		4											
CD20+_RO01778						1	1															2											
CD20+_RO01794							1															5											
H1-neurons																				3		4								1			
HeLa-S3	1	1	1				3	3		4					6	4		3	8	3		93			1	1	2		1			1	1
HepG2	2	1	1			1	2	1		4					6	2	5	2	8	3		114			1		2		1				1
HUVEC	1					1	2	1							5	2		2	8	1		36					2		1				1
IMR90	2	1					1								3			3	4	9		11							1			1	1
LHCN-M2						2	2												2														
MCF-7	2	3					8	3							3	7		3	5	7		49			1	3			1				1
Monocytes-CD14+							1								1				2	1													
Monocytes-CD14+_RO01746						1	1															17											
SK-N-SH	1	1					1								3			3	4	9		34											1
Tier 3																																	
8988T							1									1																	

April 2015

The Genome Access Course

How do I use ENCODE data?

- View and Search for tracks in the UCSC browser
- Understand and customize tracks and displays
- Access links to additional resources
- Browse and download data (Bulk Retrieval)

The Genome Access Course

I have a sequencing dataset from an assay that selectively sequences DNA from regions of accessible chromatin*?

How can I use the ENCODE browser to better understand my data?

*DNase Hypersensitivity sequencing or
Transposase accessible chromatin sequencing

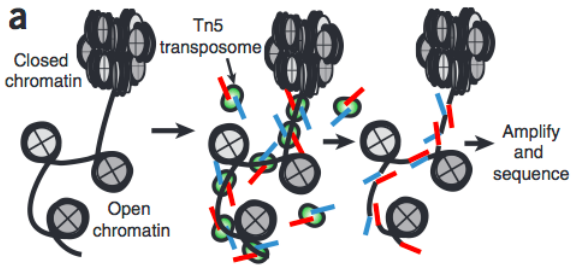
The Genome Access Course

ATAC-seq data

Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position

Jason D Buenrostro[1–3], Paul G Giresi[2,3], Lisa C Zaba[2,3], Howard Y Chang[2,3] & William J Greenleaf[1]

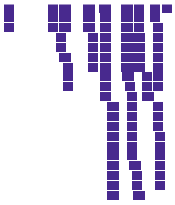
NATURE METHODS | VOL.10 NO.12 | DECEMBER 2013 | 1213



The Genome Access Course

Convert Interval track to Histogram track

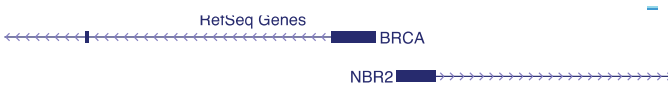
Bed or Bam



Bed or Bam track displays "intervals" of mapped sequences"

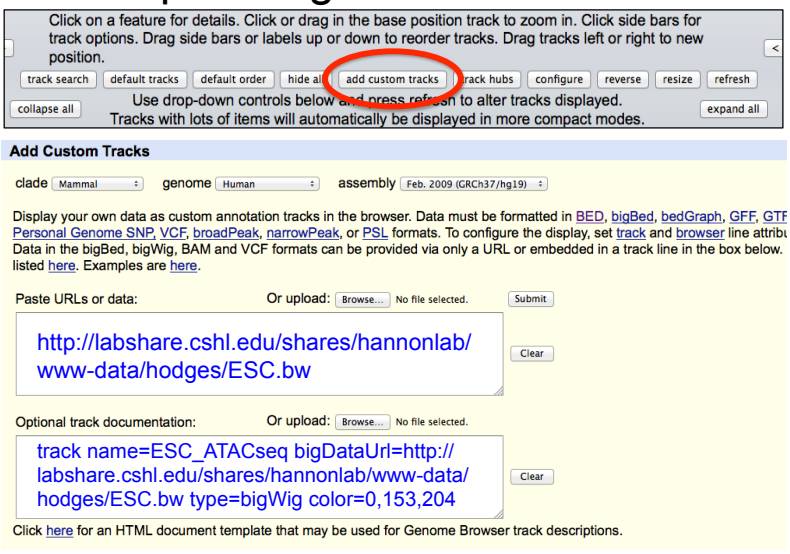
Wiggle or bigWig

Counting intervals at each base position can convert Interval counts into quantitative information.



The Genome Access Course

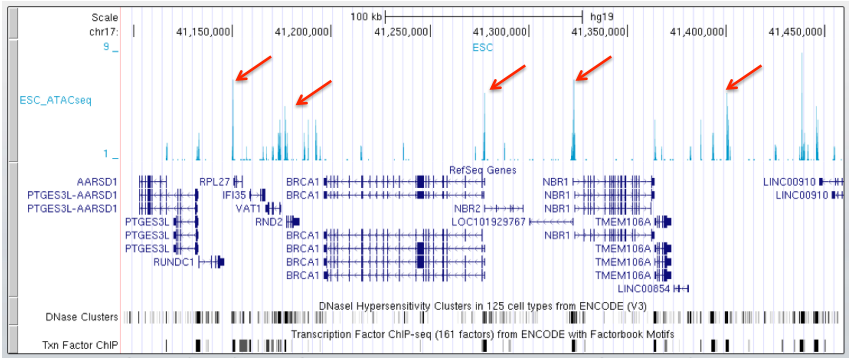
Uploading Custom Tracks



The Genome Access Course

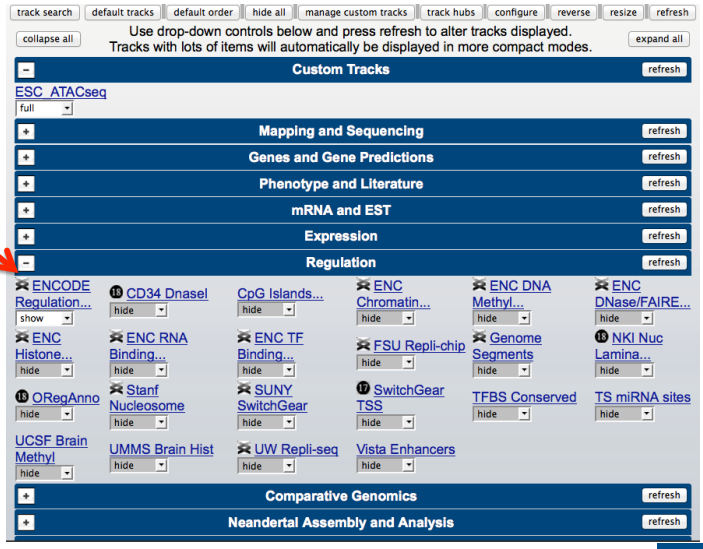
Custom Track

10x Zoomed Out



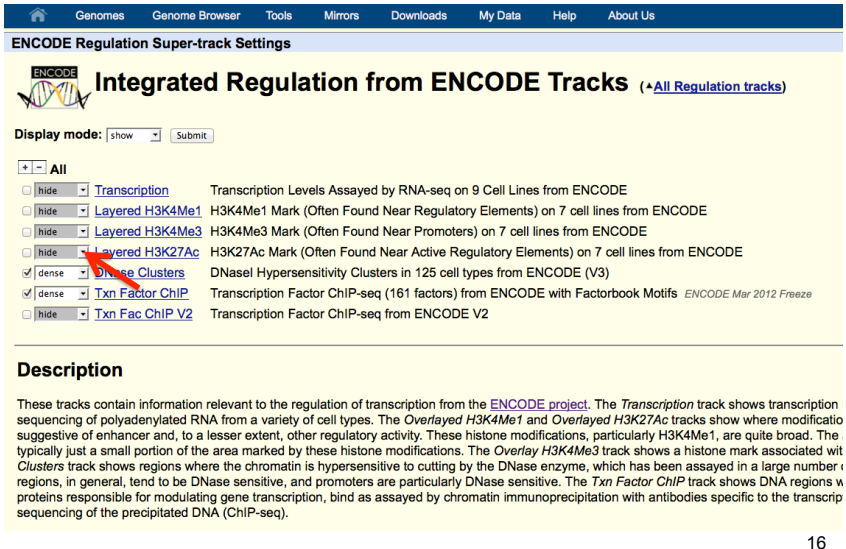
The Genome Access Course

Turning on ENCODE tracks



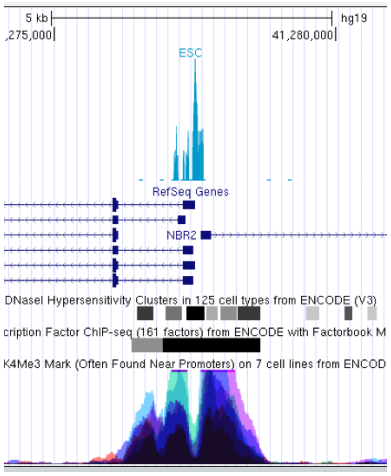
The Genome Access Course

ENCODE Super-tracks

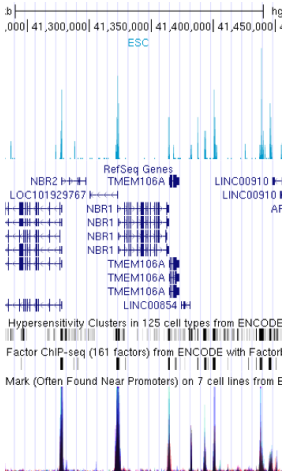


The Genome Access Course

H3K4me3 overlaps ATACseq peaks



Zoom Out

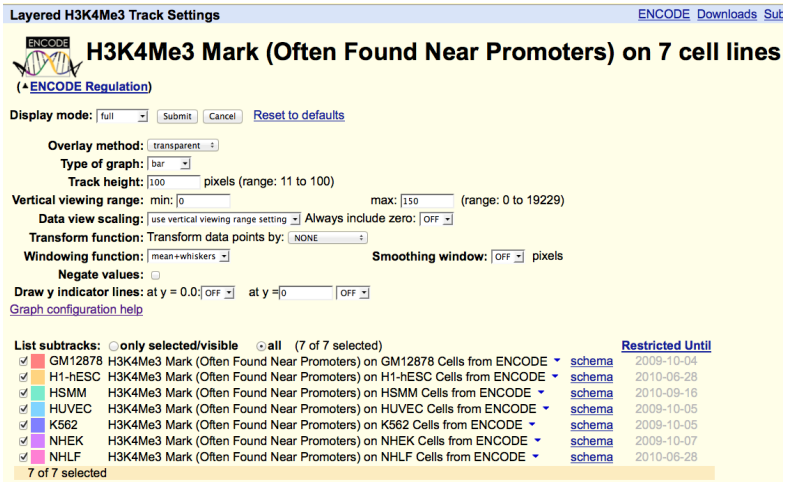


17

April 2015

The Genome Access Course

Layered H3K4Me3 Track Settings



Layered H3K4Me3 Track Settings ENCODE Downloads Sub

H3K4Me3 Mark (Often Found Near Promoters) on 7 cell lines

(▲ENCODE Regulation)

Display mode: full Submit Cancel Reset to defaults

Overlay method: transparent
Type of graph: bar
Track height: 100 pixels (range: 11 to 100)
Vertical viewing range: min: 0 max: 150 (range: 0 to 19229)
Data view scaling: use vertical viewing range setting Always include zero: OFF
Transform function: Transform data points by: NONE
Windowing function: mean+whiskers **Smoothing window:** OFF pixels
Negate values:
Draw y indicator lines: at y = 0.0: OFF at y = 0 OFF
Graph configuration help

List subtracks: only selected/visible all (7 of 7 selected) **Restricted Until**

GM12878	H3K4Me3 Mark (Often Found Near Promoters) on GM12878 Cells from ENCODE	schema	2009-10-04
H1-hESC	H3K4Me3 Mark (Often Found Near Promoters) on H1-hESC Cells from ENCODE	schema	2010-06-28
HSMM	H3K4Me3 Mark (Often Found Near Promoters) on HSMM Cells from ENCODE	schema	2010-09-16
HUVEC	H3K4Me3 Mark (Often Found Near Promoters) on HUVEC Cells from ENCODE	schema	2009-10-05
K562	H3K4Me3 Mark (Often Found Near Promoters) on K562 Cells from ENCODE	schema	2009-10-05
NHEK	H3K4Me3 Mark (Often Found Near Promoters) on NHEK Cells from ENCODE	schema	2009-10-07
NHLF	H3K4Me3 Mark (Often Found Near Promoters) on NHLF Cells from ENCODE	schema	2010-06-28

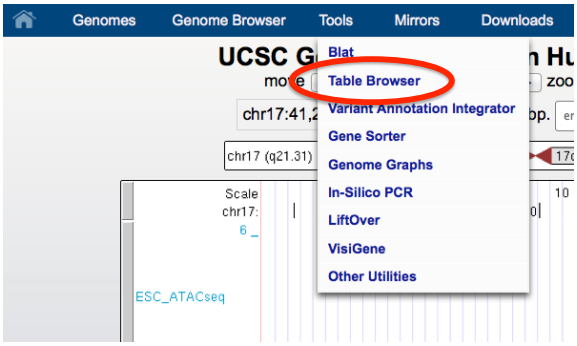
7 of 7 selected

18

April 2015

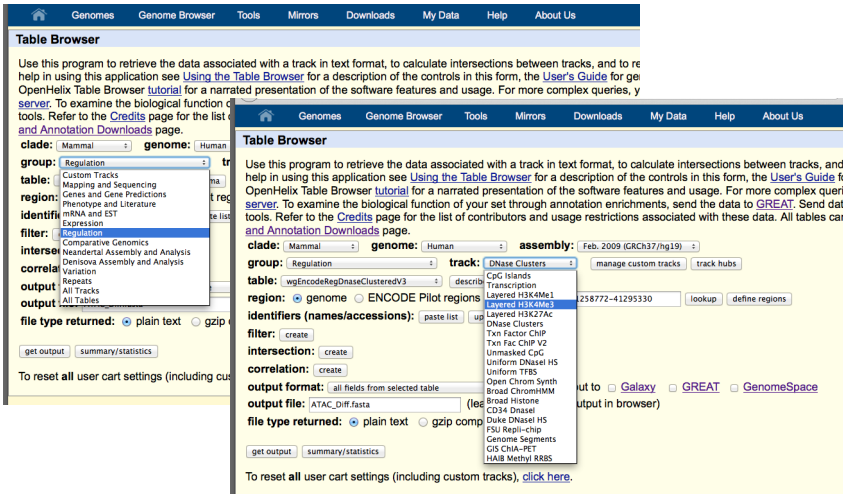
The Genome Access Course

Bulk Retrieval of ENCODE data



The Genome Access Course

Table Browser



The Genome Access Course

Downloading Data

The Genome Access Course

Sequence Polymorphisms

The Genome Access Course

Sequence polymorphisms

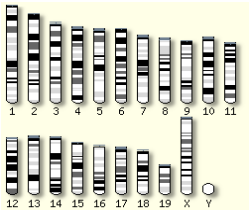
Association studies

NGS

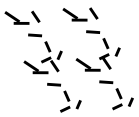
April 2015

The Genome Access Course

Sequencing the first mammalian genomes

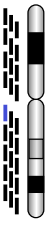


fragment

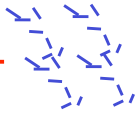


BACs
bacterial
artificial
chromosomes
Ave size 150 kb

map



fragment

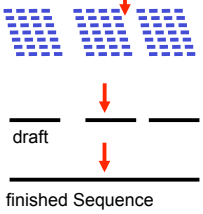


sequence
assemble

pUCs
ave size 2-4 kb

The hybrid approach

WGS



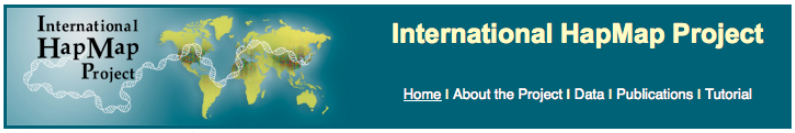
draft

finished Sequence

April 2015

The Genome Access Course

Genetic mapping and genome wide association studies using by SNPs



Phase 1: SNP identification by limited Sanger re-sequencing,

Phase 2: Identification of 'tag' SNPs

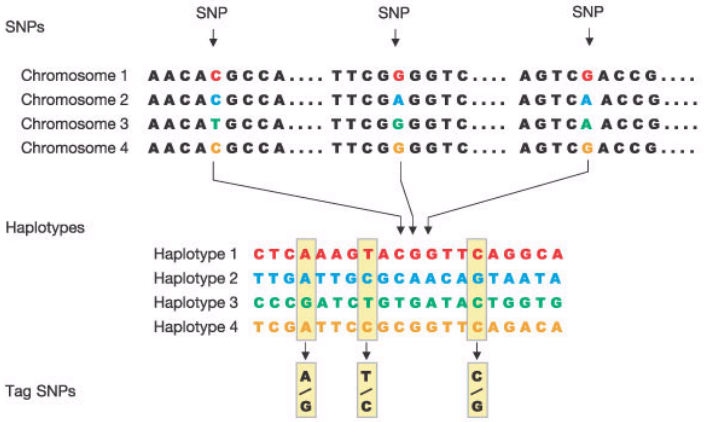
Phase 3: Screening tag SNPs in multiple populations

http://hapmap.ncbi.nlm.nih.gov

April 2015

The Genome Access Course

Haplotypes as data reduction

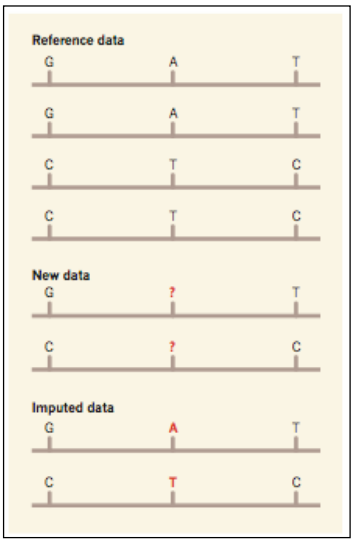


The International HapMap Project (*Nature* 2003)

April 2015

The Genome Access Course

Haplotypes facilitate genotype imputation

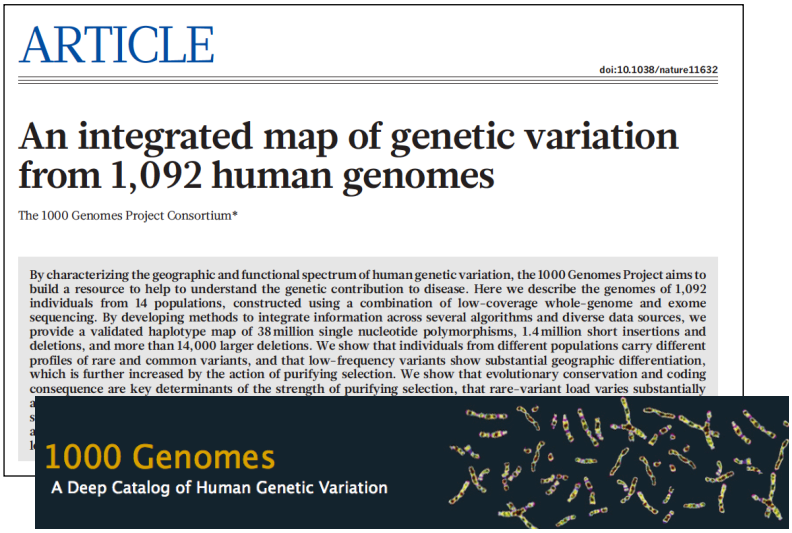


R Nielsen (*Nature* 2010)

April 2015

The Genome Access Course

From 1 genome to 1000 genomes



ARTICLE

doi:10.1038/nature11632

An integrated map of genetic variation from 1,092 human genomes

The 1000 Genomes Project Consortium*

By characterizing the geographic and functional spectrum of human genetic variation, the 1000 Genomes Project aims to build a resource to help to understand the genetic contribution to disease. Here we describe the genomes of 1,092 individuals from 14 populations, constructed using a combination of low-coverage whole-genome and exome sequencing. By developing methods to integrate information across several algorithms and diverse data sources, we provide a validated haplotype map of 38 million single nucleotide polymorphisms, 1.4 million short insertions and deletions, and more than 14,000 larger deletions. We show that individuals from different populations carry different profiles of rare and common variants, and that low-frequency variants show substantial geographic differentiation, which is further increased by the action of purifying selection. We show that evolutionary conservation and coding consequence are key determinants of the strength of purifying selection, that rare-variant load varies substantially

1000 Genomes
A Deep Catalog of Human Genetic Variation

April 2015

The Genome Access Course

The 1000 genomes project

- Characterizing the geographic and functional spectrum of human genetic variation
- Build a resource to help to understand the genetic contribution to disease.
- 1,092 individuals from 14 populations, constructed using a combination of low-coverage whole-genome and exome sequencing
- Validated haplotype map of 38 million single nucleotide polymorphisms, 1.4 million short insertions and deletions, and more than 14,000 larger deletions
- 1000genomes.org

April 2015

The Genome Access Course

Sequencing 17 mouse strains

ARTICLE

doi:10.1038/nature10413

Mouse genomic variation and its effect on phenotypes and gene regulation

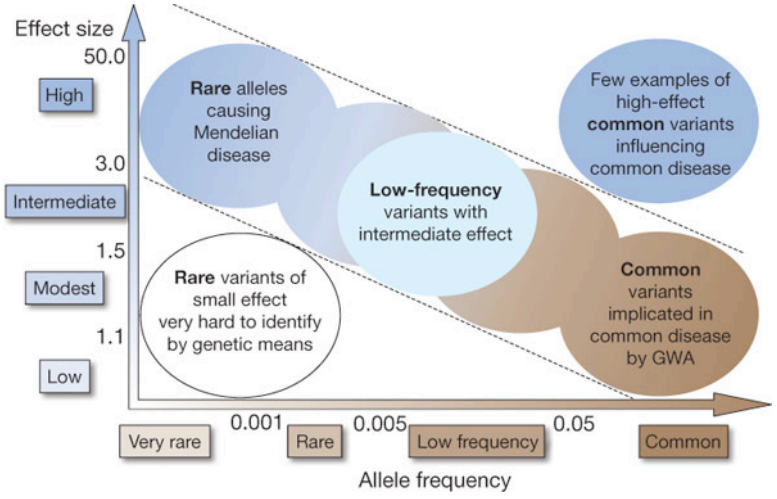
Thomas M. Keane[1]*, Leo Goodstadt[2]*, Petr Danecek[1]*, Michael A. White[3], Kim Wong[1], Binnaz Yalcin[2], Andreas Heger[4], Avigail Agam[2,4], Guy Slater[1], Martin Goodson[2], Nicholas A. Furlotte[5], Eleazar Eskin[5], Christoffer Nellåker[4], Helen Whitley[2], James Cleak[2], Deborah Janowitz[2,6], Polinka Hernandez-Pliego[2], Andrew Edwards[2], T. Grant Belgard[4], Peter L. Oliver[4], Rebecca E. McIntyre[1], Amarjit Bhomra[2], Jérôme Nicod[2], Xiangchao Gan[2], Wei Yuan[2], Louise van der Weyden[1], Charles A. Steward[1], Sendu Bala[1], Jim Stalker[1], Richard Mott[2], Richard Durbin[1], Ian J. Jackson[7], Anne Czechanski[8], José Afonso Guerra-Assunção[9], Leah Rae Donahue[8], Laura G. Reinholdt[8], Bret A. Payseur[3], Chris P. Ponting[4], Ewan Birney[9], Jonathan Flint[2] & David J. Adams[1]

We report genome sequences of 17 inbred strains of laboratory mice and identify almost ten times more variants than previously known. We use these genomes to explore the phylogenetic history of the laboratory mouse and to examine the functional consequences of allele-specific variation on transcript abundance, revealing that at least 12% of transcripts show a significant tissue-specific expression bias. By identifying candidate functional variants at 718 quantitative trait loci we show that the molecular nature of functional variants and their position relative to genes vary according to the effect size of the locus. These sequences provide a starting point for a new era in the functional analysis of a key model organism.

April 2015

The Genome Access Course

Missing heritability

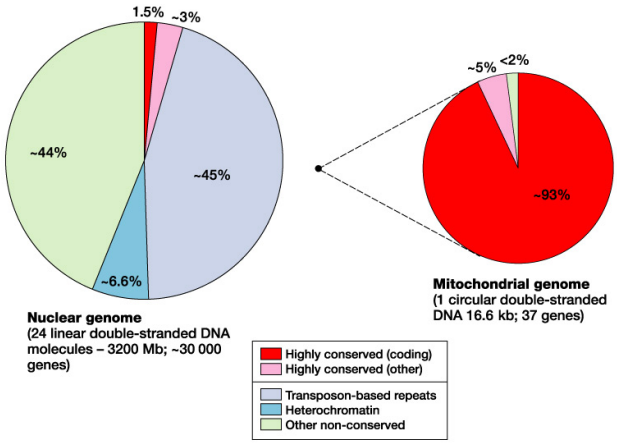


Manolio *et al,* Nature 461, 747-753 (8 October 2009)

April 2015

The Genome Access Course

Most of the human genome does not code for protein

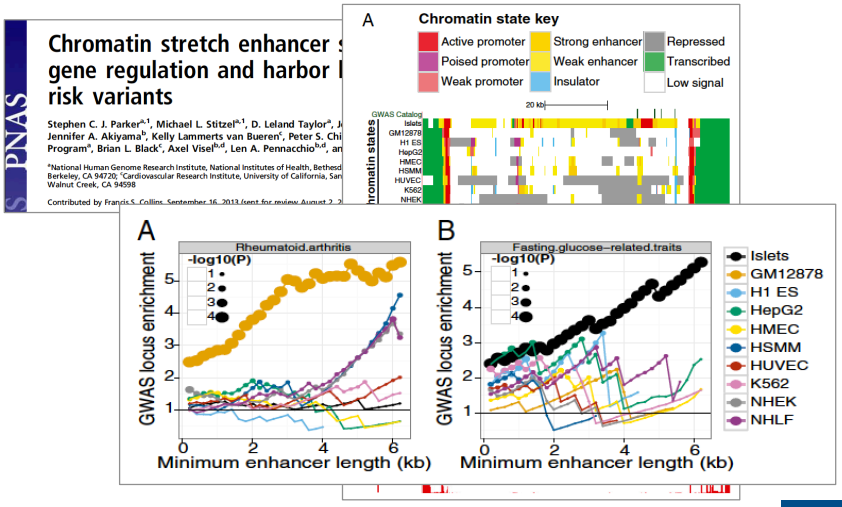


Tom Strachan & Andrew P. Read, *Human Molecular Genetics*

April 2015

The Genome Access Course

Human sequence variants lie in non-coding regions



April 2015

The Genome Access Course

Accessing the data

- Entrez – the holder of all things variation
  - **dbSNP:** database for short genetic variation
  - **dbVar:** database for structural variation
  - **dbGAP:** database of genotypes and phenotypes
- Genome Browsers
  - **Ensembl** and **UCSC**
- Anciliary databases
  - **DGV** (database of genomic variants)

April 2015

The Genome Access Course

The limits of GWAS

- GWAS use TAG SNPs
- The Tag SNP is only “associated” with the phenotype
- All SNPs in the haplotype are associated with the disease
- Its challenging to identify the causal SNP
- It may lie in coding, genic or non-genic regions

HaploReg v2

HaploReg is a tool for exploring annotations of the noncoding genome at variants on haplotype blocks, such as candidate regulatory SNPs at disease-associated loci. Using LD information from the 1000 Genomes Project, linked SNPs and small indels can be visualized along with their predicted chromatin state, their sequence conservation across mammals, and their effect on regulatory motifs. HaploReg is designed for researchers developing mechanistic hypotheses of the impact of non-coding variants on clinical phenotypes and normal variation.

Update 2013.02.14: Version 2 now includes an expanded library of SNPs (based on dbSNP 137), motif instances (based on PWMs discovered from ENCODE experiments), enhancer annotations (adding 90 cell types from the Roadmap Epigenome Mapping Consortium), and eQTLs (from the GTex eQTL browser). In addition, LD calculations are provided based on the 1000 Genomes Phase 1 individuals, and r^2 and D' measurements are available down to an r^2 threshold of 0.2. Display improvements include improved cell metadata, gene metadata, and PWM display on the detail pages and the option for text output. Version 1 is available here.

April 2015

The Genome Access Course

Large-scale annotation of sequence variation

April 2015

The Genome Access Course

DNA-seq analysis

Alignment

Variant prediction

Variant annotation

Filtering and Prioritizing variants

BWA

Samtools **GATK**

Mpileup Unified Genotyper

e.g. Seattleseq, SNP effect, Variant Annotator
Variant Effect Predictor (Ensembl)

April 2015

The Genome Access Course

Prioritizing filtered variants

Presence/Absence in affected/unaffected

Consequence to protein – strongest impact

Conservation – most highly conserved

Allele frequency in ESV – low minor allele freq.

April 2015

The Genome Access Course

Prioritizing filtered variants based on location in gene

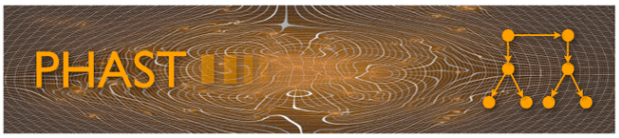
- Consequence to protein

1. High impact:
   Stop codon, splice site mutation
2. Moderate impact:
   Non-synonymous (polyphen – potentially damaging), intron-near exon,
3. Low impact
   Synonymous, intronic, UTR, intergenic

April 2015

The Genome Access Course

Prioritizing filtered variants based on sequence conservation



- http://compgen.bscb.cornell.edu/phast/
- Identifies evolutionary conserved elements
- Produces base by base conservation score between 0 and 1 (1=highly conserved)
- Predict likelihood a base is part of a conserved element

April 2015

The Genome Access Course

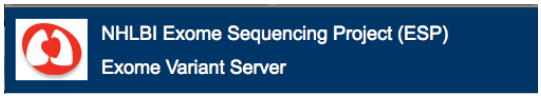
Prioritizing filtered variants based on sequence conservation

- http://mendel.stanford.edu/SidowLab/downloads/gerp/
- GERP: Genomic evolutionary rate profiling
- Identifies constrained elements in multiple alignments by quantifying substitution deficits
- Score between 5.82 and -11.6 (5.82 highest conservation)
- Deficits represent substitutions that would have occurred if the element were neutral DNA

April 2015

The Genome Access Course

Prioritizing filtered variants based on allele frequency in exome variant server

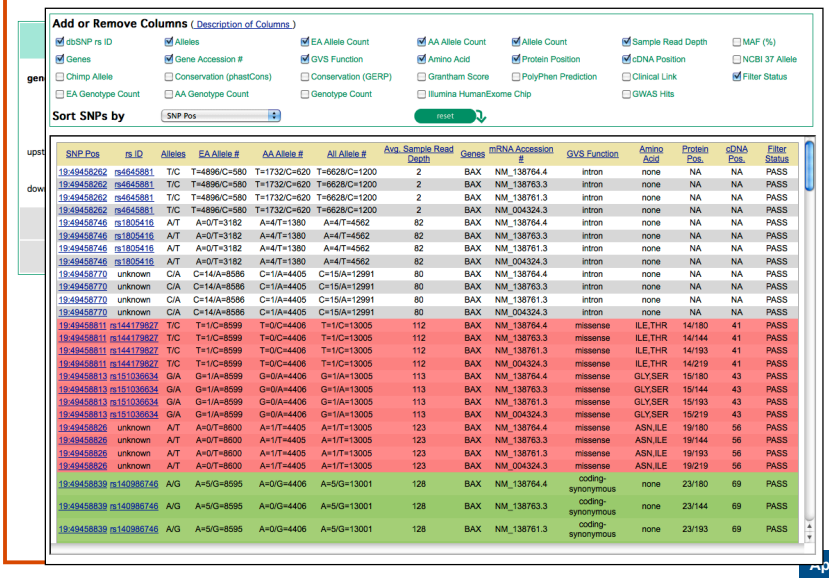


- Exome sequencing project: to discover novel genes and mechanisms contributing to heart, lung and blood disorders
- To date – sequenced 6503 samples from multiple ESP cohorts
- Generated Exome variant server
  - Predict allele frequency for every variant identified
  - http://evs.gs.washington.edu/EVS/

April 2015

The Genome Access Course

Exome variant server

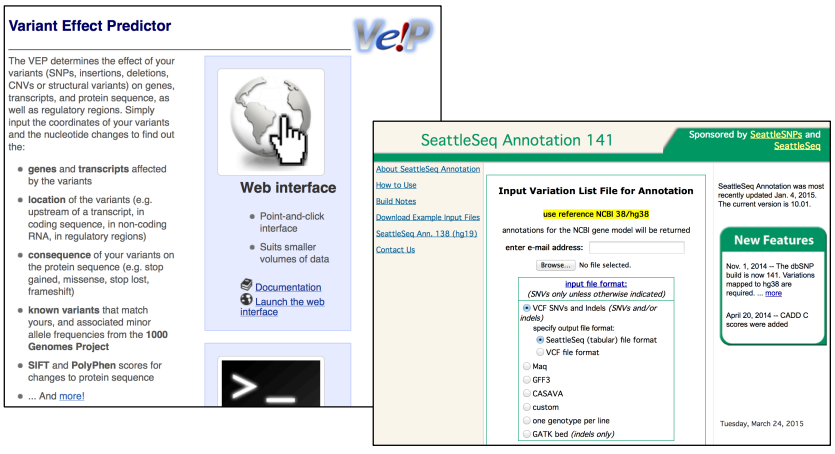


SNP Pos	rs ID	Alleles	EA Allele #	AA Allele #	All Allele #	Avg. Sample Read Depth	Genes	mRNA Accession #	GVS Function	Amino Acid	Protein Pos.	cDNA Pos.	Filter Status
19:49458262	rs4645881	T/C	T=4896/C=580	T=1732/C=620	T=6628/C=1200	2	BAX	NM_138764.4	intron	none	NA	NA	PASS
19:49458262	rs4645881	T/C	T=4896/C=580	T=1732/C=620	T=6628/C=1200	2	BAX	NM_138763.3	intron	none	NA	NA	PASS
19:49458262	rs4645881	T/C	T=4896/C=580	T=1732/C=620	T=6628/C=1200	2	BAX	NM_138761.3	intron	none	NA	NA	PASS
19:49458262	rs4645881	T/C	T=4896/C=580	T=1732/C=620	T=6628/C=1200	2	BAX	NM_004324.3	intron	none	NA	NA	PASS
19:49458746	rs1805416	A/T	A=0/T=3182	A=4/T=1380	A=4/T=4562	82	BAX	NM_138764.4	intron	none	NA	NA	PASS
19:49458746	rs1805416	A/T	A=0/T=3182	A=4/T=1380	A=4/T=4562	82	BAX	NM_138763.3	intron	none	NA	NA	PASS
19:49458746	rs1805416	A/T	A=0/T=3182	A=4/T=1380	A=4/T=4562	82	BAX	NM_138761.3	intron	none	NA	NA	PASS
19:49458746	rs1805416	A/T	A=0/T=3182	A=4/T=1380	A=4/T=4562	82	BAX	NM_004324.3	intron	none	NA	NA	PASS
19:49458770	unknown	C/A	C=14/A=8586	C=1/A=4405	C=15/A=12991	80	BAX	NM_138764.4	intron	none	NA	NA	PASS
19:49458770	unknown	C/A	C=14/A=8586	C=1/A=4405	C=15/A=12991	80	BAX	NM_138763.3	intron	none	NA	NA	PASS
19:49458770	unknown	C/A	C=14/A=8586	C=1/A=4405	C=15/A=12991	80	BAX	NM_138761.3	intron	none	NA	NA	PASS
19:49458770	unknown	C/A	C=14/A=8586	C=1/A=4405	C=15/A=12991	80	BAX	NM_004324.3	intron	none	NA	NA	PASS
19:49458811	rs144179827	T/C	T=1/C=8599	T=0/C=4406	T=1/C=13005	112	BAX	NM_138764.4	missense	ILE,THR	14/180	41	PASS
19:49458811	rs144179827	T/C	T=1/C=8599	T=0/C=4406	T=1/C=13005	112	BAX	NM_138763.3	missense	ILE,THR	14/144	41	PASS
19:49458811	rs144179827	T/C	T=1/C=8599	T=0/C=4406	T=1/C=13005	112	BAX	NM_138761.3	missense	ILE,THR	14/193	41	PASS
19:49458811	rs144179827	T/C	T=1/C=8599	T=0/C=4406	T=1/C=13005	112	BAX	NM_004324.3	missense	ILE,THR	14/219	41	PASS
19:49458813	rs151036634	G/A	G=1/A=8599	G=0/A=4406	G=1/A=13005	113	BAX	NM_138764.4	missense	GLY,SER	15/180	43	PASS
19:49458813	rs151036634	G/A	G=1/A=8599	G=0/A=4406	G=1/A=13005	113	BAX	NM_138763.3	missense	GLY,SER	15/144	43	PASS
19:49458813	rs151036634	G/A	G=1/A=8599	G=0/A=4406	G=1/A=13005	113	BAX	NM_138761.3	missense	GLY,SER	15/193	43	PASS
19:49458813	rs151036634	G/A	G=1/A=8599	G=0/A=4406	G=1/A=13005	113	BAX	NM_004324.3	missense	GLY,SER	15/219	43	PASS
19:49458826	unknown	A/T	A=0/T=8600	A=1/T=4405	A=1/T=13005	123	BAX	NM_138764.4	missense	ASN,ILE	19/180	56	PASS
19:49458826	unknown	A/T	A=0/T=8600	A=1/T=4405	A=1/T=13005	123	BAX	NM_138763.3	missense	ASN,ILE	19/144	56	PASS
19:49458826	unknown	A/T	A=0/T=8600	A=1/T=4405	A=1/T=13005	123	BAX	NM_138761.3	missense	ASN,ILE	19/193	56	PASS
19:49458826	unknown	A/T	A=0/T=8600	A=1/T=4405	A=1/T=13005	123	BAX	NM_004324.3	missense	ASN,ILE	19/219	56	PASS
19:49458839	rs140986746	A/G	A=5/G=8595	A=0/G=4406	A=5/G=13001	128	BAX	NM_138764.4	coding-synonymous	none	23/180	69	PASS
19:49458839	rs140986746	A/G	A=5/G=8595	A=0/G=4406	A=5/G=13001	128	BAX	NM_138763.3	coding-synonymous	none	23/144	69	PASS
19:49458839	rs140986746	A/G	A=5/G=8595	A=0/G=4406	A=5/G=13001	128	BAX	NM_138761.3	coding-synonymous	none	23/193	69	PASS

April 2015

The Genome Access Course

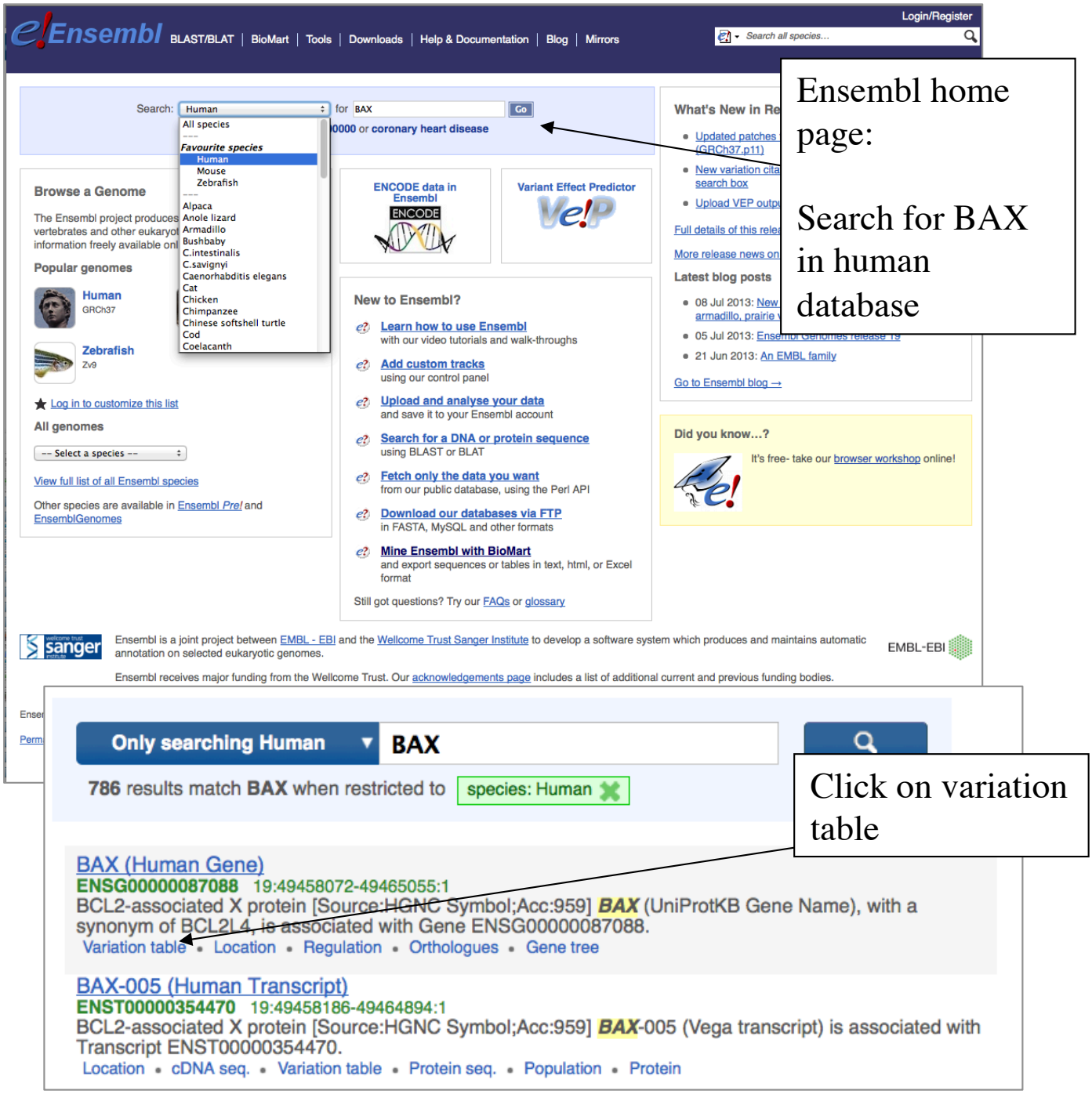
High throughput annotation of sequence variations

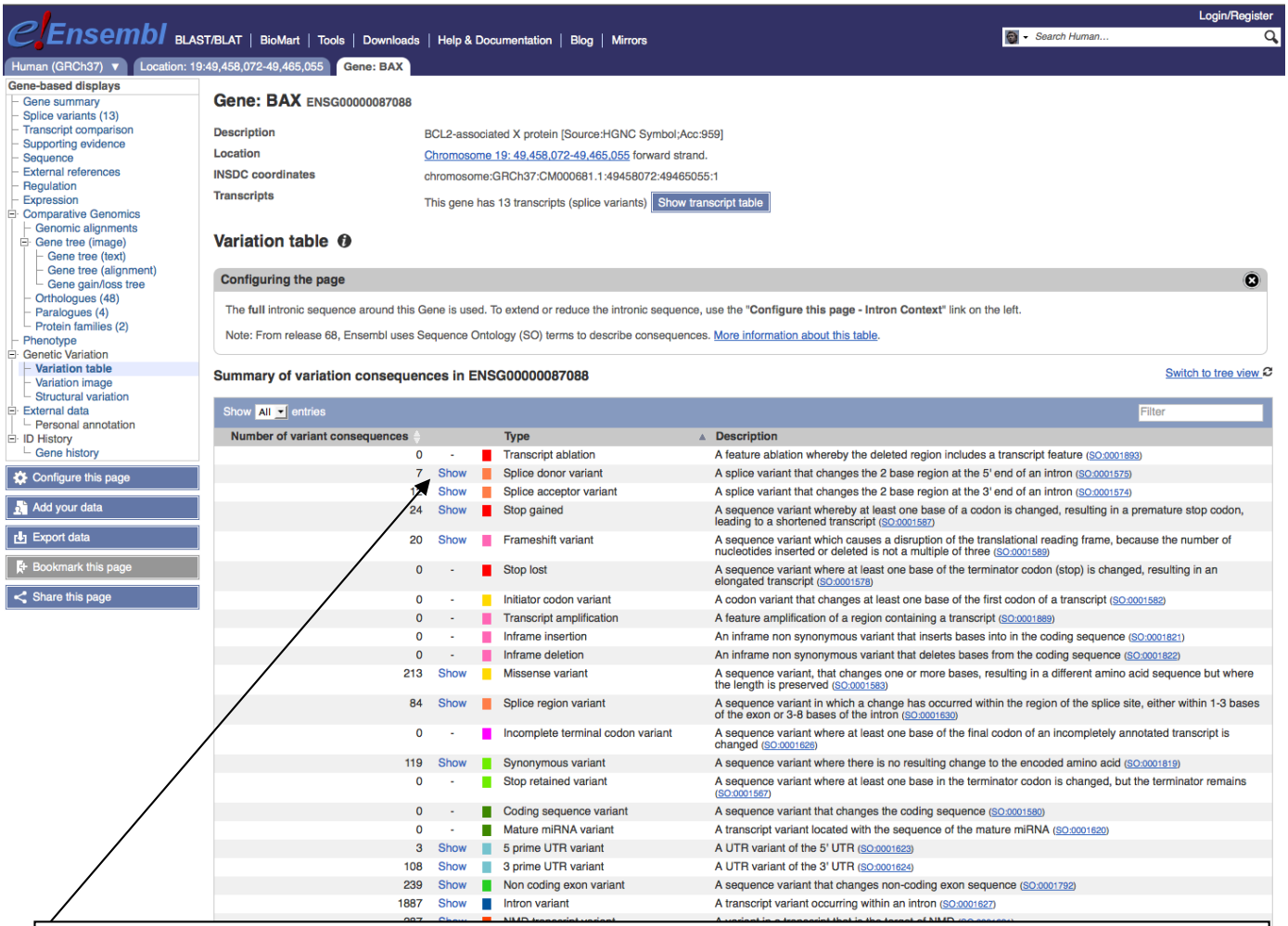


April 2015

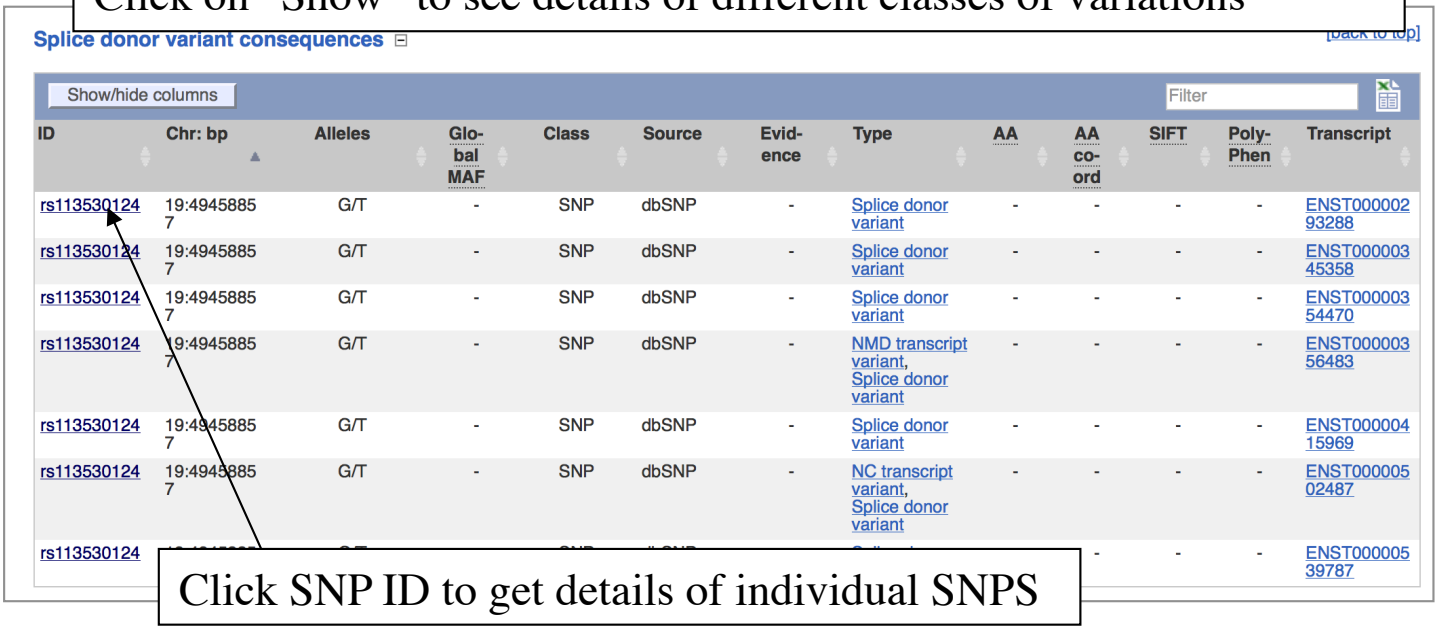
Sequence polymorphisms - worked examples

Find all variation data for the BAX gene *(or your gene of interest)*

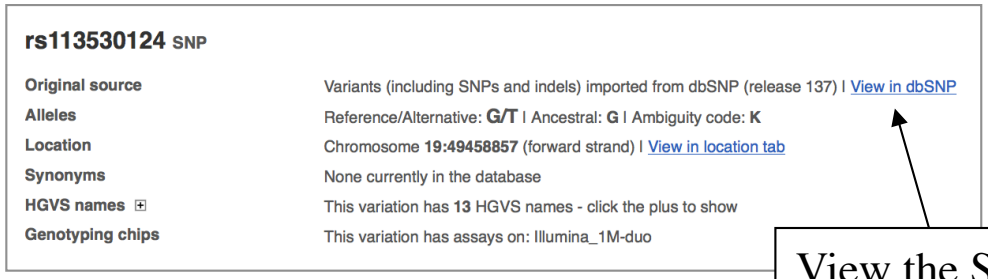




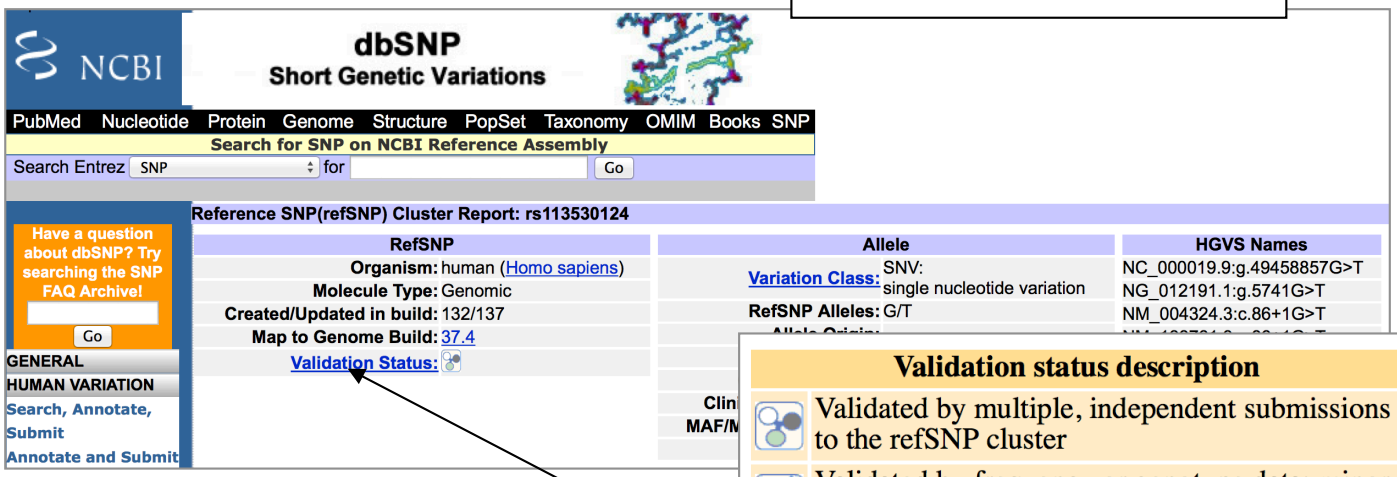
All currently annotated sequence variations in the human BAX gene
Click on "Show" to see details of different classes of variations



Click SNP ID to get details of individual SNPS



View the SNP in dbSNP



Check out validation status for the SNP

Validation status description

- Validated by multiple, independent submissions to the refSNP cluster
- Validated by frequency or genotype data: minor alleles observed in at least two chromosomes.
- Validated by submitter confirmation
- All alleles have been observed in at least two chromosomes apiece
- Genotyped by HapMap project
- SNP has been sequenced in 1000Genome project.
- Suspect SNPs: snp suspected from paralogous region (PMID: 21030649). Added to dbSNP on 01/21/2011.

Scroll down to identify individual experiments

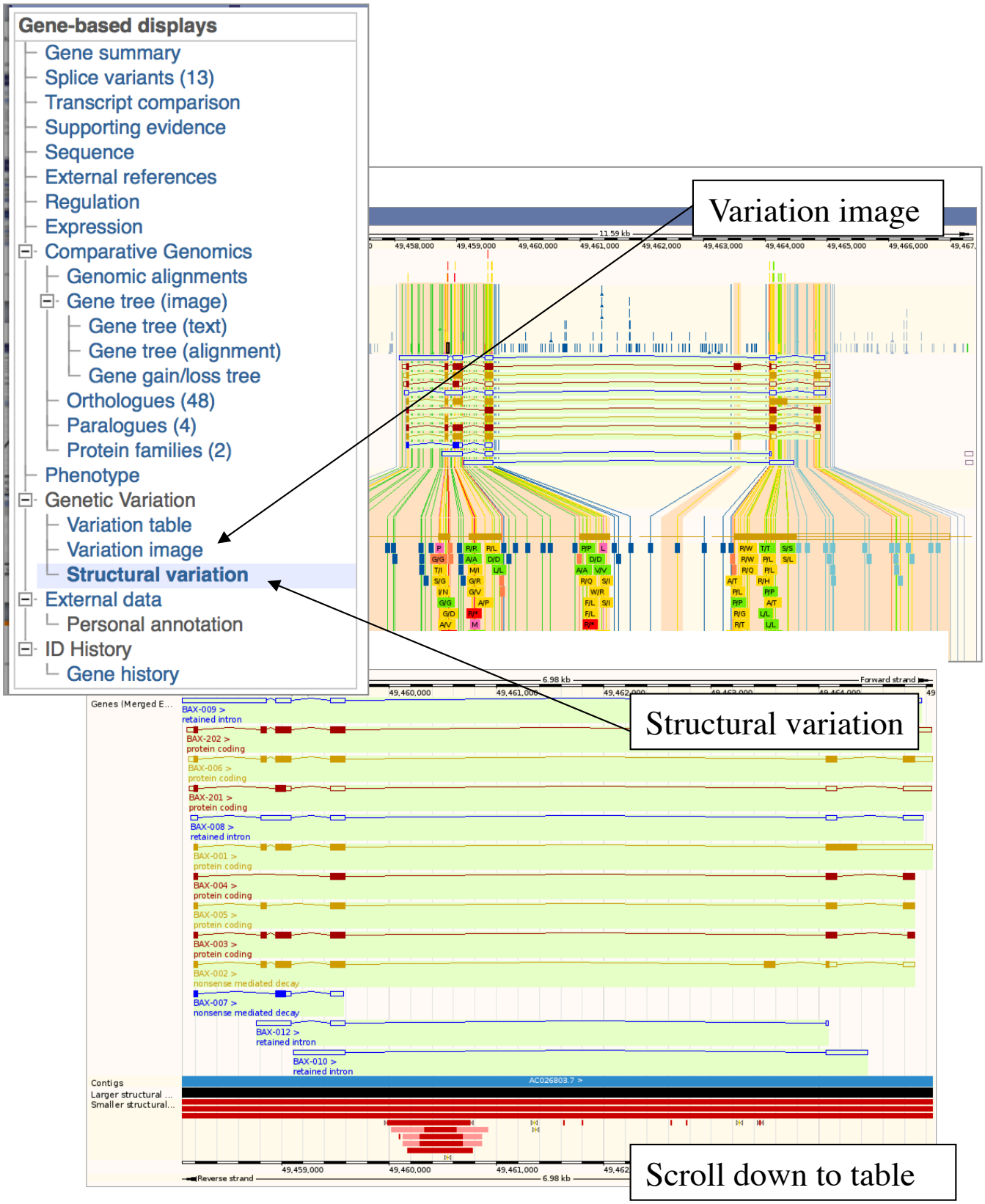
The submission **ss168991386** has the longest flanking sequence of all cluster members and was used to instantiate sequence for **rs113530124** during BLAST analysis for the current build.

NCBI Assay ID	Handle|Submitter ID	Validation Status	ss to rs Orientation /Strand	Alleles	5' Near Seq 30 bp	3' Near Seq 30 bp	Entry Date	Update Date
ss168991386	ILLUMINA|Human1M-Duov3_B_GA016638-0_B_F_1533446678		fwd/B	G/T	catgaagacaggggccctttgcttcaggg	tgagtttgaggtctgattattgtggcacag	10/01/09	10/01/09
ss532738535	ILLUMINA|HumanOmni5-4v1_B__GA016638-0_B_F_1852969807		fwd/B	G/T	catgaagacaggggccctttgcttcaggg	tgagtttgaggtctgattattgtggcacag	06/22/12	06/22/12

Fasta sequence (Legend)

```
>gnl|dbSNP|rs113530124|allelePos=61|totalLen=121|taxid=9606|snpclass=1|alleles='G/T'|mol=Genomic|build=138
TCCTCTAGGG CCCACCAGCT CTGAGCAGAT CATGAAGACA GGGGCCCTTT TGCTTCAGGG
K
TGAGTTTGAG GTCTGATTAT TGTGGCACAG ATTTGAGGAG TGACACCCCG TTCTGATTCT
```

Click to Gene page for BAX in Ensembl



Structural variants

Show 10 entries | Show/hide columns | Filter

Name	Chr:bp	Genomic size (bp)	Class	Source Study	Study description
esv2713835	19:46640739-51871541	-	inversion	DGVa:estd192	Database of Genomic Variants Archive: Catalogue of Somatic Mutations in Cancer (COSMIC) version 61
nsv9739	19:48405097-50646320	2,241,224	CNV	DGVa:nstd4	Database of Genomic Variants Archive: Perry 2 complex architecture of human copy-number v [remapped from build NCBI35]
nsv531628	19:48411797-59051332	10,639,536	CNV	DGVa:nstd37	Database of Genomic Variants Archive: International Standards for Cytogenetic Array Consortium PMID:21844811 PMID:20466091 [remapped from build NCBI36]
nsv531629	19:48432832-59083573	10,650,742	CNV	DGVa:nstd37	Database of Genomic Variants Archive: International Standards for Cytogenetic Array Consortium PMID:21844811 PMID:20466091 [remapped from build NCBI36]
nsv833856	19:49351123-49489740	138,618	CNV	DGVa:nstd68	Database of Genomic Variants Archive: Wong 2007 "A comprehensive analysis of common copy-number variations in the human genome" PMID:17160897 [remapped from build NCBI35]
nsv833857	19:49417733-49649577	231,845	CNV	DGVa:nstd68	Database of Genomic Variants Archive: Wong 2007 "A comprehensive analysis of common copy-number variations in the human genome" PMID:17160897 [remapped from build NCBI35]
nsv912248	19:49431839-49572547	140,709	CNV	DGVa:nstd71	Database of Genomic Variants Archive: Xu 2011 "SgD-CNV, a database for common and rare copy number variants in three Asian populations." PMID:21882294 [remapped from build NCBI36]
esv2718694	19:49459984-49460752	769	CNV	DGVa:estd201	Database of Genomic Variants Archive: Wong 2012b "Deep whole-genome sequencing of 100 southeast Asian Malays" PMID:23290073
esv1344170	19:49460095-49460101	7	deletion	DGVa:estd22	Database of Genomic Variants Archive: Levy 2007 "The diploid genome sequence of an individual human." PMID:17803354 [remapped from build NCBI36]
esv2665637	19:49460176-49460779	604	CNV	DGVa:estd199	Database of Genomic Variants Archive: 1000 Genomes Project Consortium - Phase 1. PMID:23128226
esv3318161	19:49460177-49460783	607	CNV	DGVa:estd59	Database of Genomic Variants Archive: 1000 Genomes Project Consortium - Pilot Project. PMID:20981092 [remapped from build NCBI36]
esv3318164	19:49460211-49460787	577	CNV	DGVa:estd59	Database of Genomic Variants Archive: 1000 Genomes Project Consortium - Pilot Project. PMID:20981092 [remapped from build NCBI36]
esv3318163	19:49460211-49460787	577	CNV	DGVa:estd59	Database of Genomic Variants Archive: 1000 Genomes Project Consortium - Pilot Project. PMID:20981092 [remapped from build NCBI36]
esv1113376	19:49460549-49460549	-	insertion	DGVa:estd22	Database of Genomic Variants Archive: Levy 2007 "The diploid genome sequence of an individual human." PMID:17803354 [remapped from build NCBI36]
nsv953598	19:49460901-49588400	127,500	CNV	DGVa:nstd73	Database of Genomic Variants Archive: Dogan 2013 "In this study we have sequenced the whole genome of an anonymous healthy male Turkish individual with high coverage (~35x). Resulting high quality data represented ~1.18 billion paired-end reads accounting for ~116,720 M bp. The structural variations (SV) submitted in this entry have been identified using paired-end and read-depth based SV calling algorithms."

Download to excel

Choose nsv953598

Structural variation: nsv953598

Variation class	CNV (SO:0001019)
Allele type(s)	deletion (SO:0000159)
Source	DGVa - Database of Genomic Variants Archive
Study	nstd73 - Dogan 2013 "In this study we have sequenced the whole genome of an anonymous healthy male Turkish individual with high coverage (~35x). Resulting high quality data represented ~1.18 billion paired-end reads accounting for ~116,720 M bp. The structural variations (SV) submitted in this entry have been identified using paired-end and read-depth based SV calling algorithms."
Location	Chromosome **19:49460901-49588400** (forward strand) I View in location tab
Genomic size	127,500 bp

Explore this SV

Genomic context | Genes and regulation | Supporting evidence | Phenotype data

Assess impact on genes and regulation

Using the website

- Video: Browsing SNPs and CNVs in Ensembl
- Video: Demo: Structural variation for a region

Analysing your data

Test your own structural variants with the Variant Effect Predictor

Programmatic access

- Tutorial: Accessing structural variatio

Reference materials

- Ensembl variation documentation portal
- Ensembl variation data description

Structural variation: nsv953598

Variation class	CNV (SO:0001019)
Allele type(s)	deletion (SO:0000159)
Source	DGVa - Database of Genomic Variants Archive
Study	nstd73 - Dogan 2013 "In this study we have sequenced the whole genome of an anonymous healthy male Turkish individual with high coverage (~35x). Resulting high quality data represented ~1.18 billion paired-end reads accounting for ~116,720 M bp. The structural variations (SV) submitted in this entry have been identified using paired-end and read-depth based SV calling algorithms."
Location	Chromosome **19:49460901-49588400** (forward strand) I View in location tab
Genomic size	127,500 bp

Genes and regulation

Impact of CNV on genes

Gene and Transcript consequences

Show 25 entries | Show/hide columns | Filter

Gene	Transcript (strand)	Allele type	Consequence types	Position in transcript	Position in CDS	Position in protein	Exons	Transcript coverage
ENSG00000087086 HGNC: FTL	ENST00000331825 (1) biotype: protein_coding	deletion	Transcript ablation	-	-	-	1-4 of 4	1578bp, 100.00%
ENSG00000087088 HGNC: BAX	ENST00000502487 (1) biotype: retained_intron	deletion	Non coding exon variant Intron variant NC transcript variant Feature truncation	-	-	-	4-5 of 5	4061bp, 58.94%
ENSG00000087088 HGNC: BAX	ENST00000539787 (1) biotype: protein_coding	deletion	Stop lost Coding sequence variant 3 prime UTR variant Intron variant Feature truncation	-	-	-	5-7 of 7	4152bp, 59.86%
ENSG00000087088 HGNC: BAX	ENST00000345358 (1) biotype: protein_coding	deletion	Stop lost Coding sequence variant 3 prime UTR variant Intron variant Feature truncation	-	-	-	5-6 of 6	4155bp, 60.03%

Regulatory consequences

Impact of CNV on regulatory regions

Show 25 entries | Show/hide columns

Feature	Feature type	Allele type	Consequence types	
ENSR00000045095	Regulatory feature	deletion	Regulatory region ablation | Regulatory	
ENSR00000045096	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	834bp, 100.00%
ENSR00000045099	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	928bp, 100.00%
ENSR00000045103	Regulatory feature	deletion	Regulatory region variant	582bp, 37.60%
ENSR00000142826	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	315bp, 100.00%
ENSR00000142837	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	902bp, 100.00%
ENSR00000217412	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	157bp, 100.00%
ENSR00000217417	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	642bp, 100.00%
ENSR00000217421	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	346bp, 100.00%
ENSR00000217422	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	468bp, 100.00%
ENSR00000217427	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	668bp, 100.00%
ENSR00000217429	Regulatory feature	deletion	Regulatory region ablation | Regulatory region variant	177bp, 100.00%

Haploreg – part of the Encode project
Finding all SNPs within a haplotype and determine their potential consequence (beyond coding, e.g. regulatory elements)

Search for rs12989701 in Haploreg

HaploReg v2

HaploReg is a tool for exploring annotations of the noncoding genome at variants on haplotype blocks, such as candidate regulatory SNPs at disease-associated loci. Using LD information from the 1000 Genomes Project, linked SNPs and small indels can be visualized along with their predicted chromatin state, their sequence conservation across mammals, and their effect on regulatory motifs. HaploReg is designed for researchers developing mechanistic hypotheses of the impact of non-coding variants on clinical phenotypes and normal variation.

Update 2013.02.14: Version 2 now includes an expanded library of SNPs (based on dbSNP 137), motif instances (based on PWMs discovered from ENCODE experiments), enhancer annotations (adding 90 cell types from the Roadmap Epigenome Mapping Consortium), and eQTLs (from the GTex eQTL browser). In addition, LD calculations are provided based on the 1000 Genomes Phase 1 individuals, and r² and D' measurements are available down to an r² threshold of 0.2. Display improvements include improved cell metadata, gene metadata, and PWM display on the detail pages and the option for text output. Version 1 is available here.

Build Query | Set Options | Documentation

Use one of the three methods below to enter a set of variants. If an r² threshold is specified (see the Set Options tab), results for each variant will be shown in a separate table along with other variants in LD. If r² is set to NA, only queried variants will be shown, together in one table.

Query (comma-delimited list of rsIDs OR a single region as chrN:start-end): rs12989701

chr	pos (hg19)	LD (r²)	LD (D')	variant	Ref	Alt	AFR freq	AMR freq	ASN freq	EUR freq	SiPhy cons	Promoter histone marks	Enhancer histone marks	DNAse	Proteins bound	eQTL tissues	Motifs changed	GENCODE genes	dbSNP func annot
2	127845956	0.94	1	rs143596712	T	TGGA	0.11	0.12	0.05	0.13			HSMM	MCF-7,CD20+,HL-60			7 altered motifs	BIN1	intronic
2	127846505	0.94	1	rs35832505	T	C	0.08	0.12	0.05	0.13			HSMM				AP-2,Ets	BIN1	intronic
2	127850292	0.93	0.99	rs28434131	A	G	0.12	0.12	0.01	0.13			HSMM				ATF3,Hic1	BIN1	intronic
2	127855392	0.93	0.99	rs34854727	C	T	0.03	0.11	0.01	0.13			HSMM	SK-N-SH_RA			4 altered motifs	BIN1	intronic
2	127859418	0.92	0.99	rs873270	T	C	0.07	0.12	0.01	0.13			HSMM	5 cell types				BIN1	intronic
2	127861766	0.89	0.99	rs4663096	T	A	0.37	0.26	0.26	0.14			HSMM	Th1,GM19240			ATF4	BIN1	intronic
2	127861906	0.89	0.99	rs4663097	G	C	0.42	0.26	0.26	0.14			HSMM	HMEC,pHTE			6 altered motifs	BIN1	intronic
2	127863029	0.92	0.99	rs7583814	C	T	0.04	0.11	0.06	0.13			4 cell types	HAEpiC,HRPEpiC,WERI-Rb-1			STAT,TCF12	BIN1	intronic
2	127864921	0.81	0.93	rs76516995	C	G	0.09	0.11	0.01	0.13		8 cell types	K562	92 cell types	6 bound proteins		25 altered motifs	BIN1	
2	127864922	0.87	0.95	rs78710909	G	C	0.02	0.10	0.01	0.12		8 cell types	K562	92 cell types	6 bound proteins		26 altered motifs	BIN1	
2	127866047	0.94	0.99	rs35860453	C	T	0.07	0.12	0.01	0.13		HepG2	H1, NHLF				GR,Smad3	1.1kb 5' of BIN1	
2	127869982	0.94	0.99	rs36085158	C	T	0.16	0.12	0.01	0.13							Mrg,TFIIA,Tgif1	5.1kb 5' of BIN1	
2	127872622	0.92	0.96	rs13029812	A	G	0.16	0.12	0.01	0.13				FibroP,HMVEC-dBl-Neo,WERI-Rb-1			5 altered motifs	7.7kb 5' of BIN1	
2	127872945	0.83	0.92	rs200210018	AG	A	0.06	0.12	0.01	0.13			HepG2				11 altered motifs	8kb 5' of BIN1	
2	127873161	0.94	0.99	rs7557280	T	A	0.01	0.11	0.01	0.13			HepG2	PanIslets,HRPEpiC			Nkx3	8.2kb 5' of BIN1	
2	127874020	0.95	1	rs12994284	T	C	0.14	0.11	0.01	0.13			HepG2				GR,Irx,Nkx3	9.1kb 5' of BIN1	
2	127874226	0.95	1	rs12994718	T	A	0.15	0.11	0.01	0.13							8 altered motifs	9.3kb 5' of BIN1	
2	127875384	0.95	1	rs34745987	C	T	0.03	0.11	0.01	0.13				GM19239,Hepatocytes			Myc	10kb 5' of BIN1	
2	127876242	0.95	1	rs71414738	C	T	0.01	0.10	0.01	0.13			HepG2	Fibrobl				11kb 5' of BIN1	
2	127881219	0.93	0.98	rs34212842	G	A	0.12	0.11	0.01	0.13							10 altered motifs	16kb 5' of BIN1	
2	127882015	0.95	1	rs6720234	T	A	0.05	0.11	0.01	0.13							4 altered motifs	17kb 5' of BIN1	
2	127887560	1	1	rs13004848	T	C	0.05	0.10	0.01	0.13				Chorion			Foxd3	23kb 5' of BIN1	
2	127887985	1	1	rs12989701	C	A	0.12	0.10	0.01	0.13				Fibrobl			5 altered motifs	23kb 5' of BIN1	
2	127889118	1	1	rs13009551	C	T	0.04	0.10	0.01	0.13							PU.1,STAT	24kb 5' of BIN1	
2	127889124	0.98	1	rs35579178	CG	C	0.05	0.10	0.01	0.13							11 altered motifs	24kb 5' of BIN1	
2	127889932	0.95	0.99	rs													Dobox4,Ncx	25kb 5' of BIN1	

Click on entry for rs12989701

Link to dbSNP entry

Sequence facts

chr	pos (hg19)	Reference	Alternate	1000 Genomes Phase 1 Frequencies				Sequence constraint		dbSNP functional annotation
				AFR	AMR	ASN	EUR	by GERP	by SiPhy	
chr2	127887985	C	A	0.12	0.1	0.01	0.13	No	No	none

Closest annotated gene					
Source	Distance	Direction	ID/Link	Common name	Description
GENCODE	5'	23053	ENSG00000136717.10	BIN1	bridging integrator 1 [Source:HGNC Symbol;Acc:1052]
RefSeq	5'	23120	NM_004305	BIN1	bridging integrator 1 [Source:HGNC Symbol;Acc:1052]

DNAse (ENCODE)

Cell ID	Cell description	Treatment	Production center
Fibrobl	child fibroblast	None	Duke

This is so cool!

Regulatory motifs altered

PWM	Strand	Ref	Alt	Match on:
				Ref: TAAAAAATAGTATAAATGCTTAATTCCTTCTCTTTATTTCCAGAATACGGAATTGATAG Alt: TAAAAAATAGTATAAATGCTTAATTCCTTATCTTTATTTCCAGAATACGGAATTGATAG
GATA_disc1	+	1.7	13.6	BCTTATCWSB
GATA_known1	-	10.4	18.6	HBDHVDYTCTTATCTHYHHWHV
GATA_known12	-	-0.7	11.3	NVYMBTATCDNBDM
GATA_known13	-	1.1	13	DDKNYTTATCH
GATA_known2	-	3.5	12.1	NWNBYCTTATCWRHHHD
GATA_known4	-	5.8	14.4	NNHHBSTTATCWHBNDW
GATA_known6	-	-2	10	NNYWATCDDN
HDAC2_disc1	-	8	12.4	BBYTTATCWS
HDAC2_disc6	-	14.1	8	WKYYYTTYYTYYYYH
Hltf	+	6.9	9.1	NHMCWTDKVN

D930–D934 *Nucleic Acids Research, 2012, Vol. 40, Database issue* *Published online 7 November 2011*
doi:10.1093/nar/gkr917

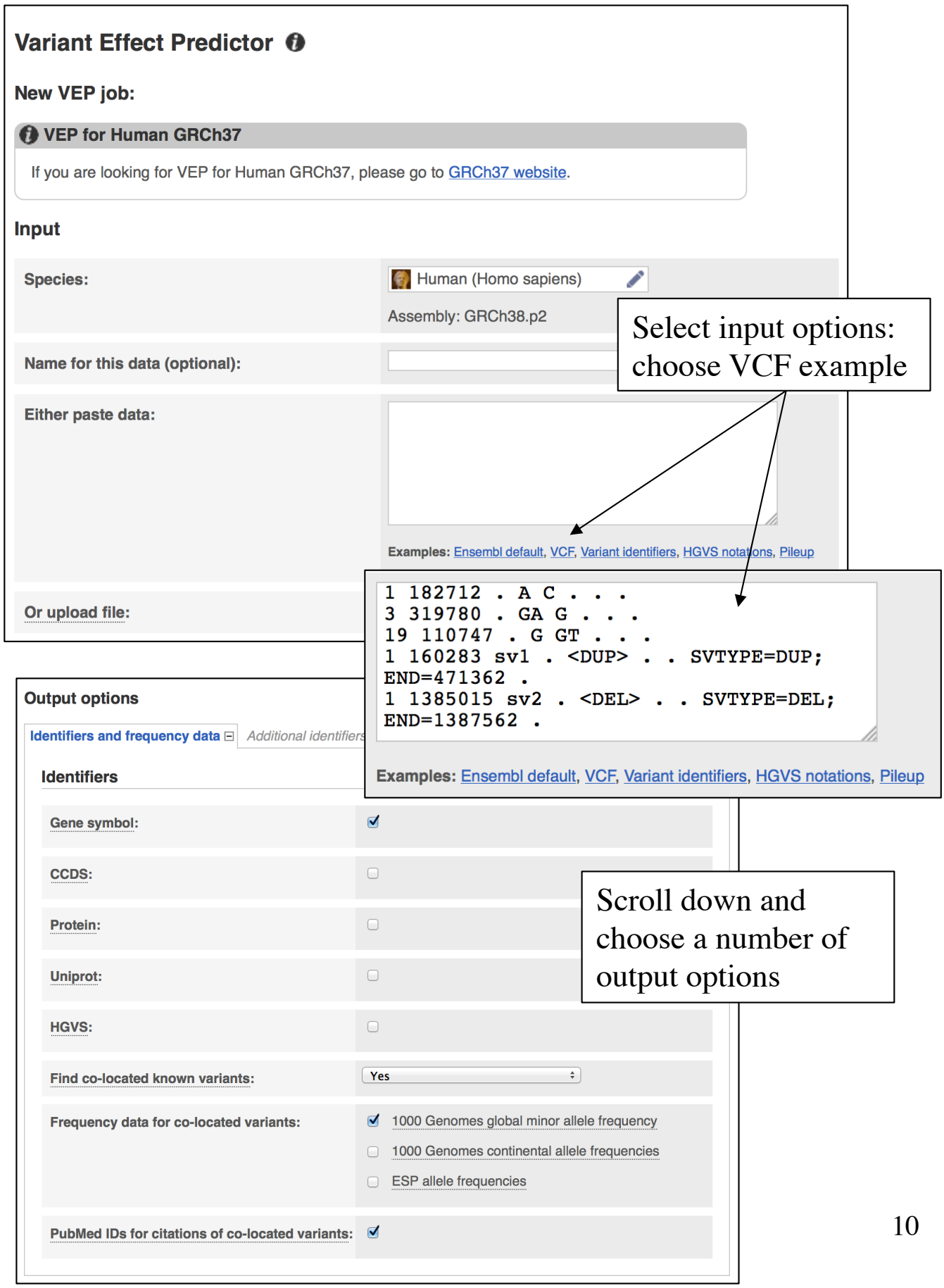
HaploReg: a resource for exploring chromatin states, conservation, and regulatory motif alterations within sets of genetically linked variants

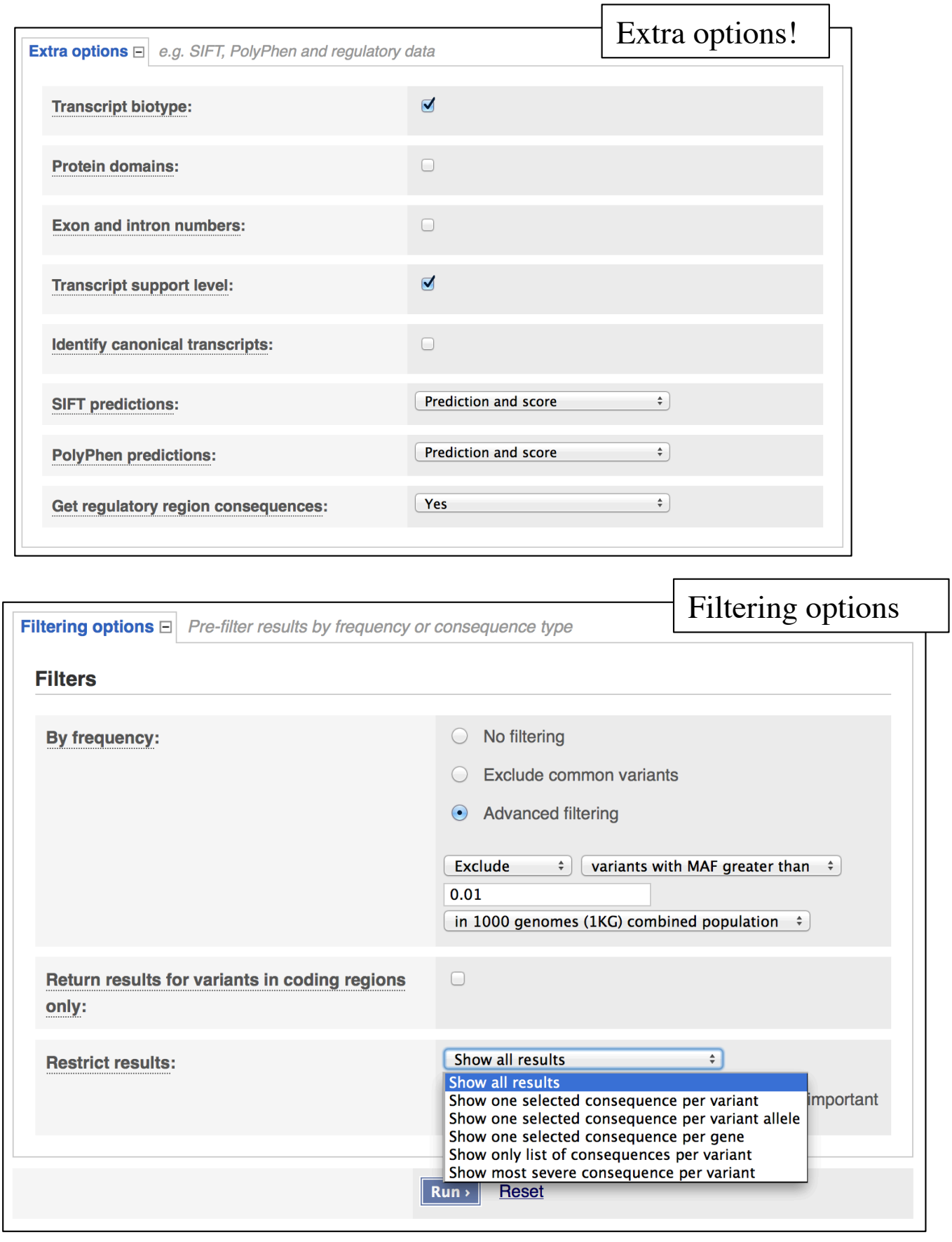
Lucas D. Ward[1,2,*] and Manolis Kellis[1,2,*]

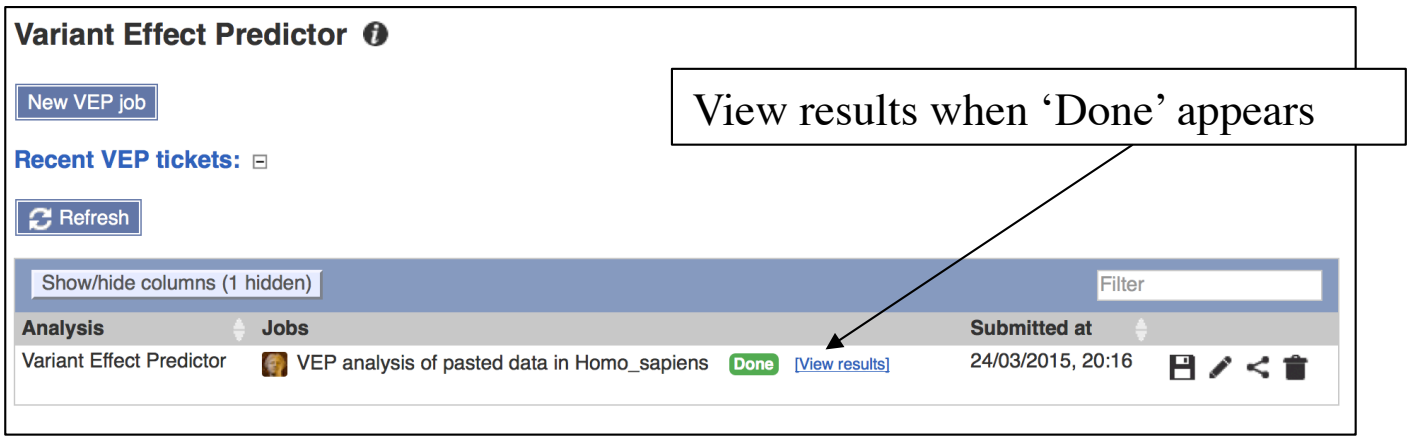
[1]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology and [2]The Broad Institute of MIT and Harvard, Cambridge, MA 02139, USA

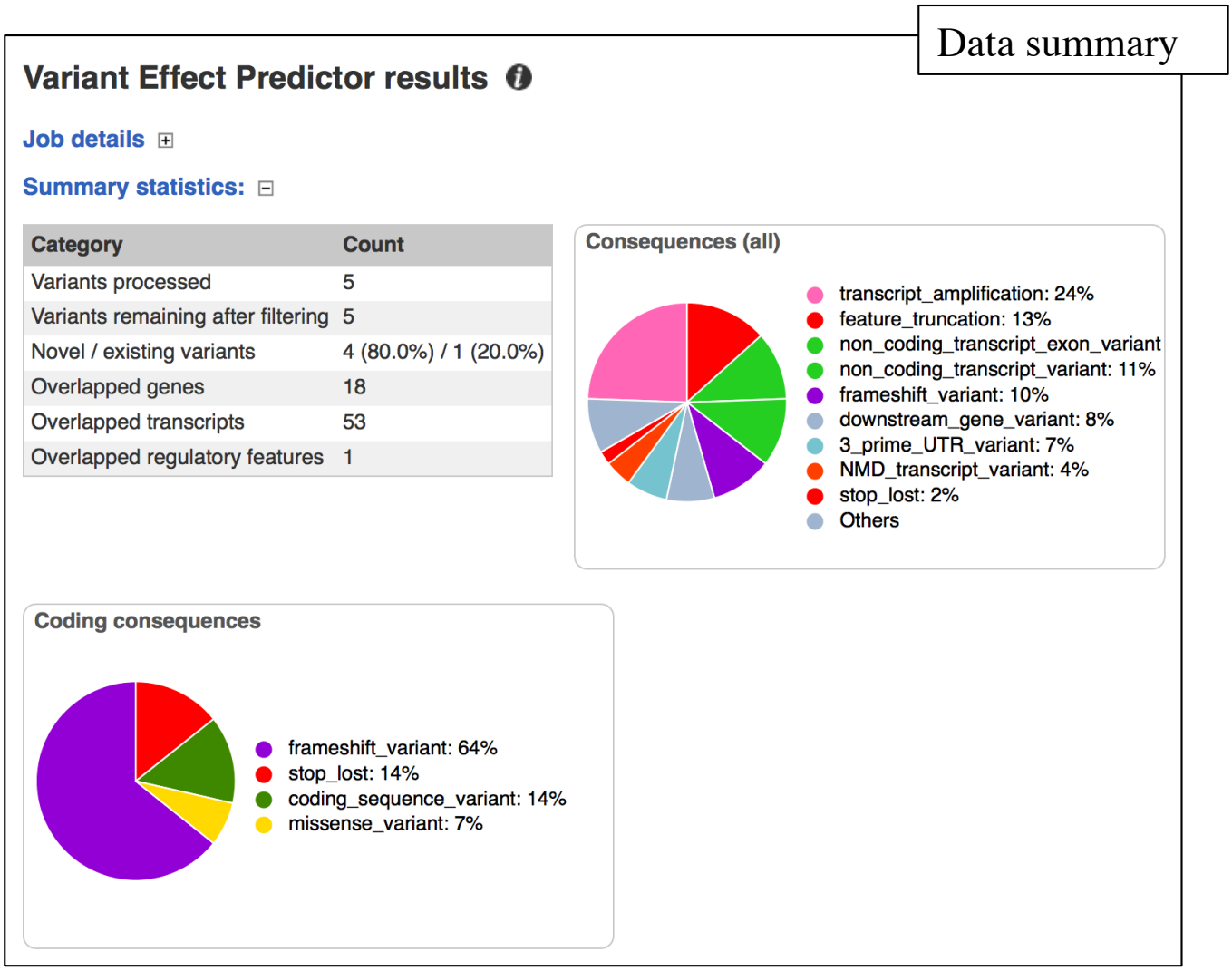
Received August 15, 2011; Revised October 6, 2011; Accepted October 8, 2011

Annotating sequence variations using Variant effect predictor (VEP) in Ensembl

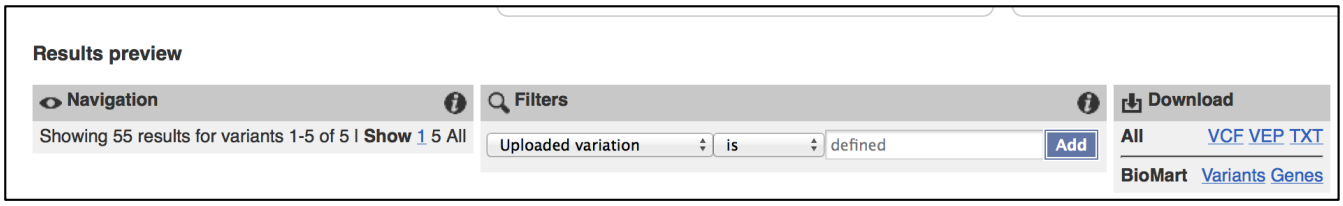

Variant Effect Predictor
New VEP job:
VEP for Human GRCh37
If you are looking for VEP for Human GRCh37, please go to GRCh37 website.
Input
Species:
Human (Homo sapiens)
Assembly: GRCh38.p2
Name for this data (optional):
Either paste data:
Examples: Ensembl default, VCF, Variant identifiers, HGVS notations, Pileup
Or upload file:
Select input options: choose VCF example
1 182712 . A C . . .
3 319780 . GA G . . .
19 110747 . G GT . . .
1 160283 sv1 . <DUP> . . SVTYPE=DUP;
END=471362 .
1 1385015 sv2 . <DEL> . . SVTYPE=DEL;
END=1387562 .
Examples: Ensembl default, VCF, Variant identifiers, HGVS notations, Pileup
Output options
Identifiers and frequency data
Additional identifiers
Identifiers
Gene symbol:
CCDS:
Protein:
Uniprot:
HGVS:
Find co-located known variants:
Yes
Frequency data for co-located variants:
1000 Genomes global minor allele frequency
1000 Genomes continental allele frequencies
ESP allele frequencies
PubMed IDs for citations of co-located variants:
Scroll down and choose a number of output options

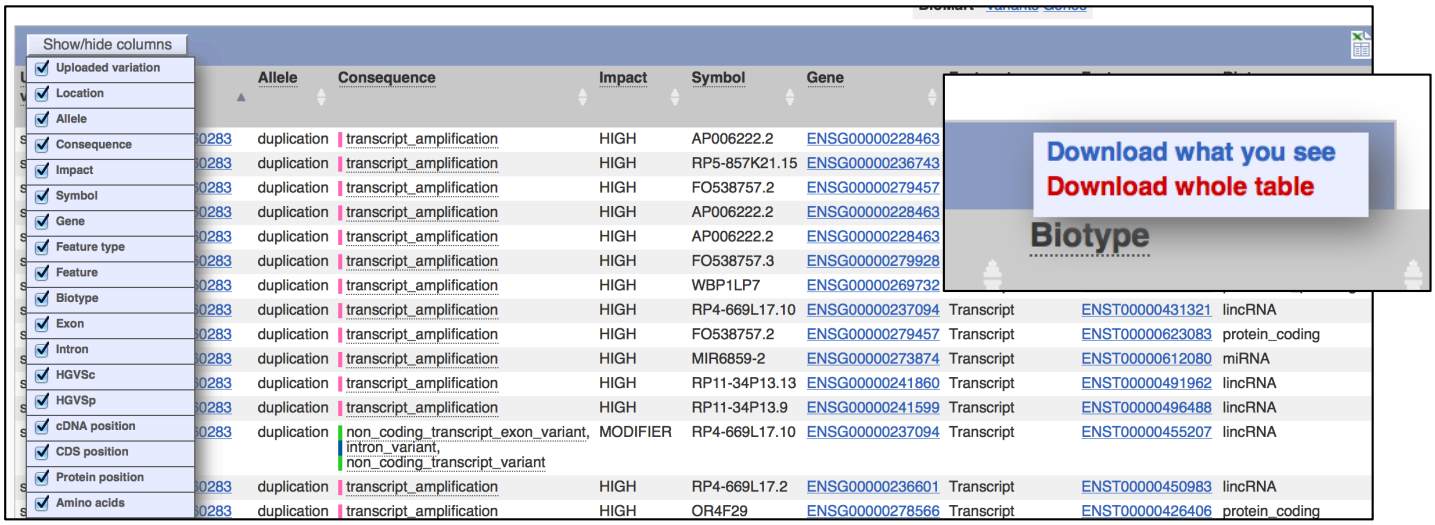

Extra options!
Extra options
e.g. SIFT, PolyPhen and regulatory data
Transcript biotype:
Protein domains:
Exon and intron numbers:
Transcript support level:
Identify canonical transcripts:
SIFT predictions:
Prediction and score
PolyPhen predictions:
Prediction and score
Get regulatory region consequences:
Yes
Filtering options
Filtering options
Pre-filter results by frequency or consequence type
Filters
By frequency:
No filtering
Exclude common variants
Advanced filtering
Exclude
variants with MAF greater than
0.01
in 1000 genomes (1KG) combined population
Return results for variants in coding regions only:
Restrict results:
Show all results
Show all results
Show one selected consequence per variant
Show one selected consequence per variant allele
Show one selected consequence per gene
Show only list of consequences per variant
Show most severe consequence per variant
important
Run ›
Reset


Variant Effect Predictor
New VEP job
Recent VEP tickets:
Refresh
View results when 'Done' appears
Show/hide columns (1 hidden)
Filter
Analysis
Jobs
Submitted at
Variant Effect Predictor
VEP analysis of pasted data in Homo_sapiens
Done
[View results]
24/03/2015, 20:16


Data summary
Variant Effect Predictor results
Job details
Summary statistics:
Category
Count
Variants processed
5
Variants remaining after filtering
5
Novel / existing variants
4 (80.0%) / 1 (20.0%)
Overlapped genes
18
Overlapped transcripts
53
Overlapped regulatory features
1
Consequences (all)
transcript_amplification: 24%
feature_truncation: 13%
non_coding_transcript_exon_variant
non_coding_transcript_variant: 11%
frameshift_variant: 10%
downstream_gene_variant: 8%
3_prime_UTR_variant: 7%
NMD_transcript_variant: 4%
stop_lost: 2%
Others
Coding consequences
frameshift_variant: 64%
stop_lost: 14%
coding_sequence_variant: 14%
missense_variant: 7%

Scroll down to see results – download to excel to save results and manipulate data.





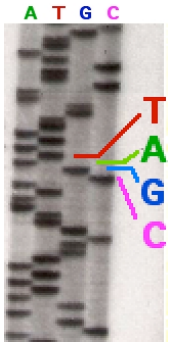
Now try Seatleseq – use the same VCF example that Ensembl provided.

The Genome Access Course

Analysis of Next Generation Sequencing Data

The Genome Access Course

Next Generation DNA Sequencing



Illumina HiSeq X
1.8 Tbp
(3 billion reads) in ~3 days

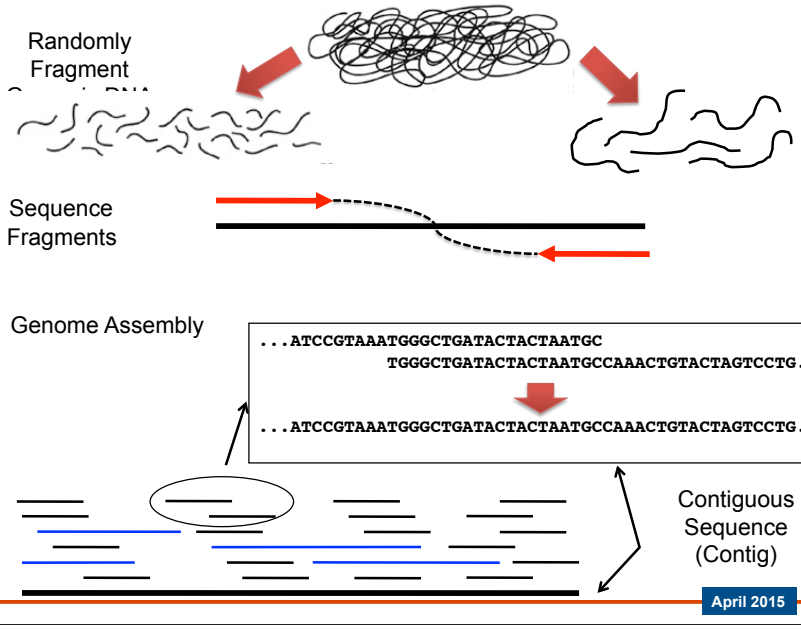
6685 SEQUENCING PUBLICATIONS

(as of 11/6/2014)

April 2015

The Genome Access Course

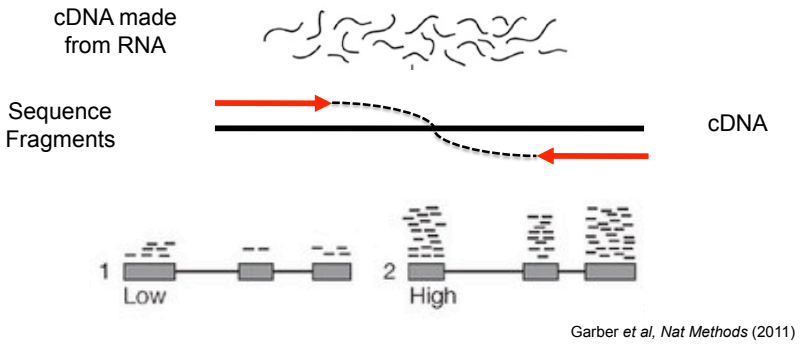
Whole Genome Shotgun Sequencing



April 2015

The Genome Access Course

RNA Sequencing (RNA-Seq)



Garber *et al, Nat Methods* (2011)

1. Characterize all RNA in sample
2. Gene expression level proportional to number of reads
3. Detect alternatively spliced transcripts

April 2015

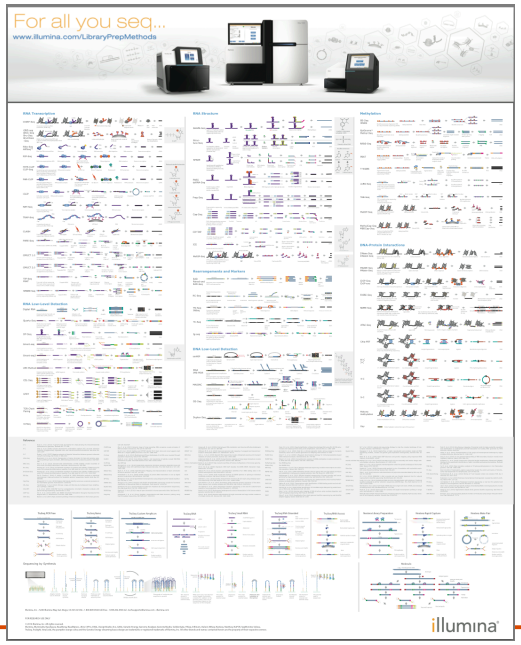
The Genome Access Course

Typical Next Gen Experiments

- Genome sequencing
  - Novel genomes
  - Resequencing
- Transcriptome sequencing (RNA-seq)
  - Characterize transcripts with or without reference genome
    - Typical length
    - Short (microRNAs, …)
  - Find differentially expressed transcripts
- Other
  - Methyl-seq
  - ChIP-seq

April 2015

The Genome Access Course



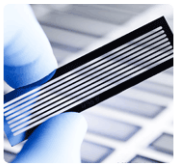
April 2015

The Genome Access Course

Illumina Sequencing

DNA Sample

Construct Library



Cluster Generation in Flow Cell

Sequencing by Synthesis

200+ million reads per lane (>100 bp reads)

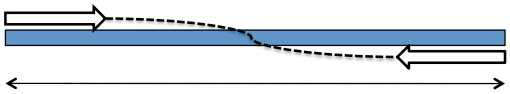
April 2015

The Genome Access Course

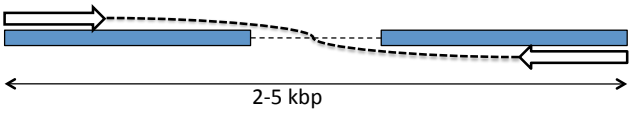
Types of Sequencing Libraries

Single-End Reads - 5’ or 3’ (random)

Paired-End Reads - 5’ and 3’

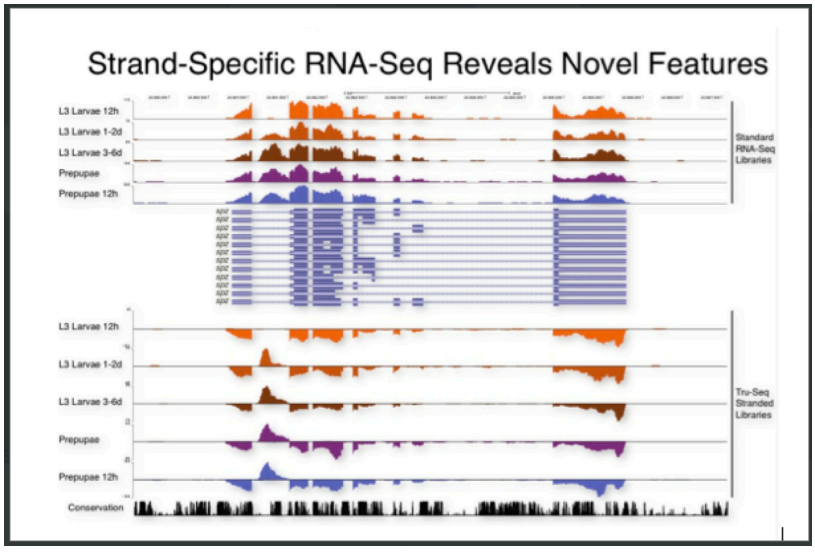


Mate-Pair Reads - 5’ and 3’



April 2015

The Genome Access Course



Taken from GIGA Newsletter 13 – Universite de Liège

April 2015

The Genome Access Course

What Does the Data Look Like?

FASTQ File Format

```
@HWI-EAS339:2:1:0:1945#0/1
TCTCAGGCCTTTTACACTTTATGCTTCCAGATCGGAAGAGCTAG
+HWI-EAS339:2:1:0:1945#0/1
FW]]]\][Y]]]]\]]\]]][]]\]\\\Y\]][][[[\XWRVFV
@HWI-EAS339:2:1:0:449#0/1
TATCGGAAGACTCGTATGCCGTCTTCTGCTTGGAAAGAAAGAGC
+HWI-EAS339:2:1:0:449#0/1
FW\[\X[[\[[\[U[\\[XZZYX\\[[VV[ZGLSGVKKUSFQFF
@HWI-EAS339:2:1:0:1211#0/1
TATCGGAAGAGCTCGTATGCCGTCTTCTGCTTGGAAAGGAAAAG
+HWI-EAS339:2:1:0:1211#0/1
UY\\ZGZ[\YX[[[[[[[XZ[[ZXZWFWXVZWFKSFVFFUSFOF
@HWI-EAS339:2:1:0:225#0/1
ACGAGCCGGAAGCATAAAGTGTAAAAGACCTGATAGATCGGAAT
+HWI-EAS339:2:1:0:225#0/1
^][][^\W\]V]^^^^\[V^]^^^^]^^]ZRWEWSZ]XYTYWE
@HWI-EAS339:2:1:0:255#0/1
ACCAGGTTATAAAAATATATCTTTTGGTGTGTGCAATATAGTTT
+HWI-EAS339:2:1:0:255#0/1
aaabaaaaaaaabbabaaaaaaaaZLX_Raa[UZHZaaaa^\[]W
@HWI-EAS339:2:1:0:361#0/1
ACTCTAGAACTCTGGGACTCTTACAGCCAAATAGCTGGTAAACT
+HWI-EAS339:2:1:0:361#0/1
--More--
```

Sequence

Quality (ASCII character for each base)

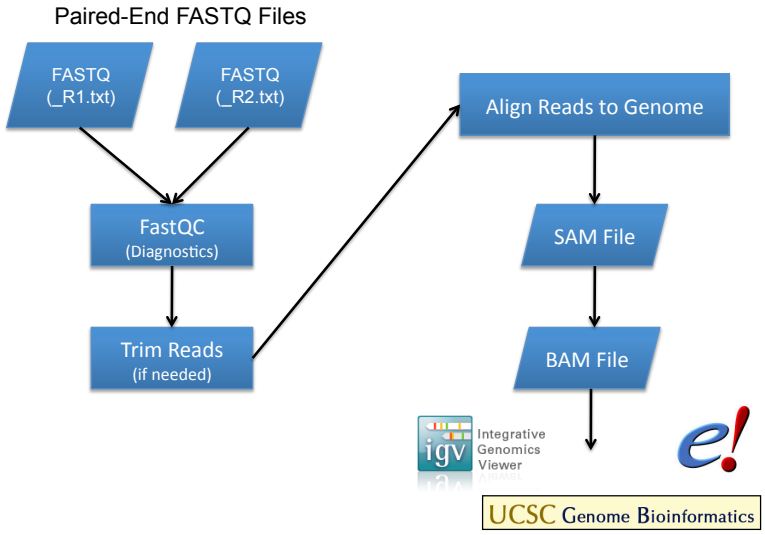
> 200 million reads in one lane

Files so big that they break them up in 40 million reads per file

April 2015

The Genome Access Course

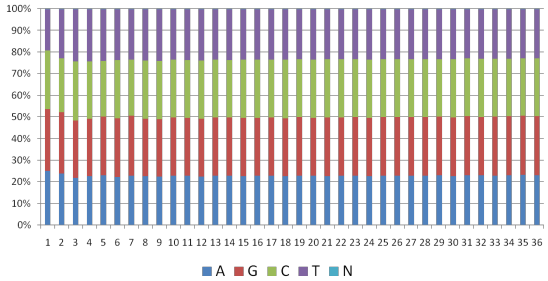
Example Analysis Workflow



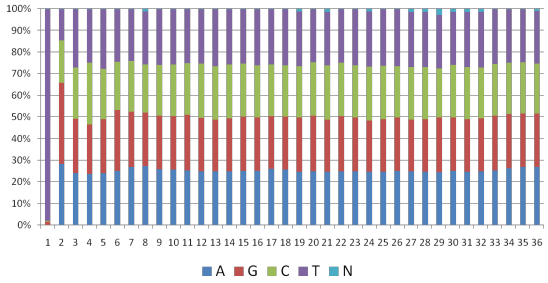
April 2015

The Genome Access Course

Sequence Composition Diagnostics



Unbiased Reads



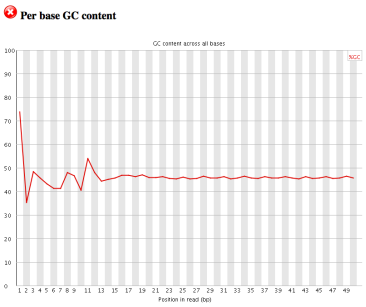
Biased Reads

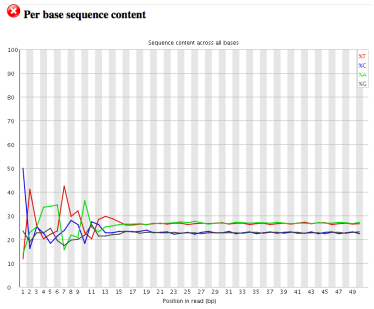
First Position Nearly Always "T"

April 2015

The Genome Access Course

GC Bias in First ~15 bp Due to Random Hexamer Priming





Short answer, the random hexamer priming is "not so random". Illumina has acknowledged this in one of their FAQs:

Quote:

Q482. Why is GC high in the first few bases?
It is perfectly normal to observe both a slight GC bias and a distinctly non-random base composition over the first 12 bases of the data. This is observed when looking, for instance, at the IVC (intensity versus cycle number) plots which are part of the output of the Pipeline. In genomic DNA sequencing, the base composition is usually quite uniform across all bases; but in mRNA-Seq, the base composition is noticeably uneven across the first 10 to 12 bases. Illumina believes this effect is caused by the "not so random" nature of the random priming process used in the protocol. This may explain why there is a slight overall G/C bias in the starting positions of each read. The first 12 bases probably represent the sites that were being primed by the hexamers used in the random priming process. The first twelve bases in the random priming full-length cDNA sequencing protocol (mRNA-seq) always have IVC plots that look like what has been described. This is because the random priming is not truly random and the first twelve bases (the length of two hexamers) are biased towards sequences that prime more efficiently.This is entirely normal and expected.

There was also a publication which investigated this:

Hansen KD, Brenner SE, Dudoit S. Biases in Illumina transcriptome sequencing caused by random hexamer priming. Nucleic Acids Res 2010 Apr.;

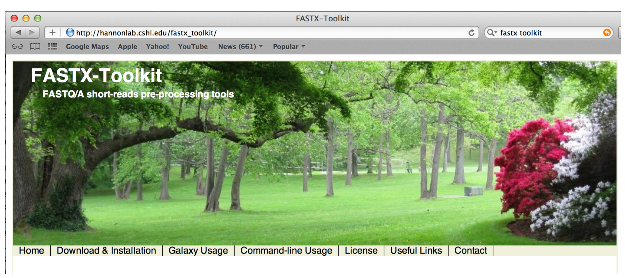
April 2015

Trim Sequences Prior To Analysis

- Make sure sequencing adapters are removed
- Trim ends of sequence based on quality scores

```
@100121_00RRPSSL:6:1:3:396:0:1
CGCTCGTCTGTACAGGGCAGGGGNANNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
+
@B@BBBBABB>A>B@@############################################################
@100121_00RRPSSL:6:1:3:1568:0:1
CGCGGCGCGAACAGCGCCAGCAGNTNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
+
ABBA<BA?A<AAA@@B@###########################################################
@100121_00RRPSSL:6:1:3:1526:0:1
GGATGACTGAGATTCCAGTGAATNGNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
+
BBBBBB:CBBBABA?<?###########################################################
@100121_00RRPSSL:6:1:3:403:0:1
AAATGATTGAATCTGGTGTCATCNANNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
+
BBBCCABBCBABBBBB?###########################################################
@100121_00RRPSSL:6:1:3:1359:0:1
GAGGAAGGGAGAGAGAGAGTGTGNCNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
+
A>?B=A?BB9A=B=>?############################################################
```


FastX Toolkit – Hannon Lab at CSHL



Trimmomatic

BIOINFORMATICS ORIGINAL PAPER
Vol. 30 no. 15 2014, pages 2114–2120
doi:10.1093/bioinformatics/btu170

Genome analysis

Advance Access publication April 1, 2014

Trimmomatic: a flexible trimmer for Illumina sequence data

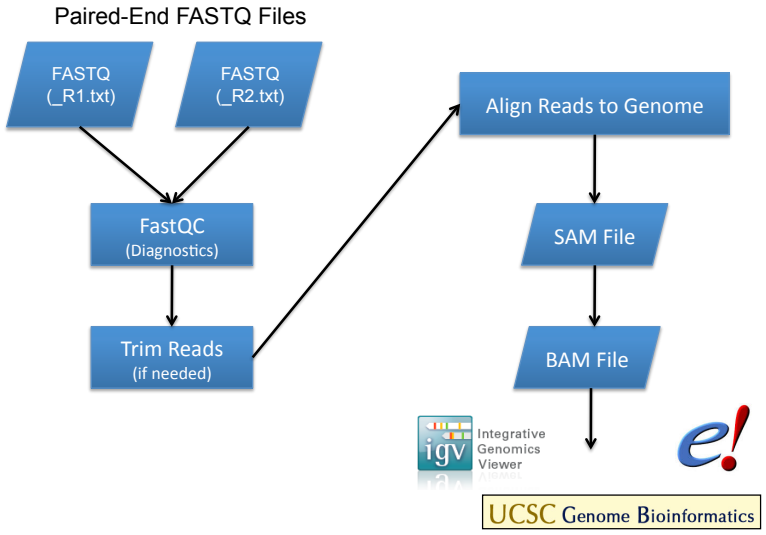
Anthony M. Bolger[1,2], Marc Lohse[1] and Bjoern Usadel[2,3,*]

[1]Department Metabolic Networks, Max Planck Institute of Molecular Plant Physiology, Am Mühlenberg 1, 14476 Golm,[2]Institut für Biologie I, RWTH Aachen, Worringer Weg 3, 52074 Aachen and [3]Institute of Bio- and Geosciences: Plant Sciences, Forschungszentrum Jülich, Leo-Brandt-Straße, 52425 Jülich, Germany

Associate Editor: Inanc Birol

The Genome Access Course

Example Analysis Workflow



April 2015

The Genome Access Course

Sequence Alignment/Map (SAM) Format

```
@HD     VN:1.0  SO:sorted
@PG     ID:TopHat       VN:1.0.14       CL:/home/bking/tophat-1.0.14.Linux_x86_64/tophat galGal3 /data/nextgen/Toronto_Aug_2010/lane8read1.phred33.fastq,/data/nextgen/Toronto_June_2010/lane2read1.phred33.fastq,/data/nextgen/Toronto_June_2010/lane1read1.phred33.fastq
100608_ccAPRAPP:2:58:15931:20463:0:1    0       chr1    2       255     76M     *       0       0       AGGGTTTCGGCCGT  CAGATCCCTCACCACGACGGACTGAGAGGGGTTTTCTGCCATCAGATCCTTTACCACGACAG  EDFDFGGGGBDFFFFAFFFFGGGDGGGFGGGGGGGEEE5ECC:>?=A+A.4=@=6=;=2@>CC=A-C########  NM:i:2  NH:i:1
100608_ccAPRAPP:2:67:12888:9899:0:1     0       chr1    17      255     76M     *       0       0       AGATCCCTCACCAC  GACGGACTGAGAGGGGTTTTCTGCCATCAGATCCTTTACCACGACAGACCAGTGAAGGGTTT  GGGGGGGGGGGGGGGFGGGDGGGGGGGGG?CEEEGGGGGFGGFDGGGGGGEGB  GGGGGG:DCDCFABEEEEFAEC  NM:i:2  NH:i:1
100608_ccAPRAPP:2:83:18913:19025:0:1    16      chr1    26      255     76M     *       0       0       CCCACGACGGACTG  AGAGGGGTTTTCTGCCATCAGATCCTTTACCACGACAGACCAGTGAAGGGTTTCGGCCGTCA  ##????<@AA@?CC?CBECAB?C?@@5AAC?-BBB?>AC>@ECFFFGFFAGFFG  EDGBGGEGGGGFFFFF?EEEEE  NM:i:2  NH:i:1
100608_ccAPRAPP:2:70:4148:3863:0:1      0       chr1    30      255     76M     *       0       0       CGACGGACTGAGAG  GGGTTTTCTGCCATCAGATCCTTTACCACGACAGACCAGTGAAGGGTTTCGGCCGTCAGATC  FDDFEE?DEEBECEECC9CCCC:DDEBEEDEEEEEEEEEEEDE?EEEDEDEABD  ?EEEE=@D@@-B=BB<@???5A  NM:i:1  NH:i:1
100727_RRRPRRRR:8:58:3878:16459:0:1     0       chr1    50      255     76M     *       0       0       TCTGCCATCAGATC  CTTTACCACGACAGACCAGTGAAGGGTTTCGGCCGTCAGATCCCTCACCACGACGGACTGGA  GGGGGGDGGGGGGGGGGGFGGEGGFGGGEGGGGGGFGGGGEGGGGGFGGGGGGF  GEEGGG?EDBEFECDF=EEE@A  NM:i:0  NH:i:1
100608_ccAPRAPP:1:100:10334:14009:0:1   16      chr1    56      255     76M     *       0       0       ATCAGATCCTTTAC  CACGACAGACCAGTGAAGGGTTTCGGCCGTCAGATCCCTCACCACGACGGACTGGAGGCTCT  #BDBFD=AB?BECCEEAFFAFFEEFEEEECDGGGGGGFGFDGGGGFFGDGGGGG  EFFGFGGGGGGGGGGGGGGGG  NM:i:0  NH:i:1
```

Common file format to store:

- Reads
- Quality of each base
- How reads align to a reference sequence

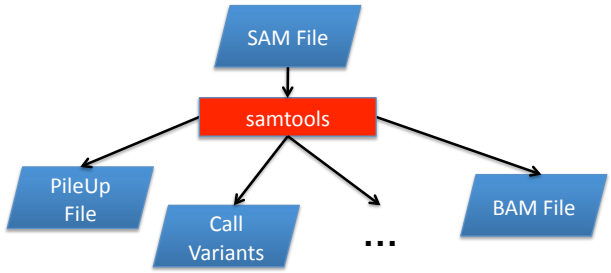
Generated by most next gen analysis software

samtools software package

April 2015

The Genome Access Course

samtools Used to Manipulate SAM Files



Pileup output file

```
chr1 272 T 24  ,.$.....,.,.,.....,.,..^+. <<<+;<<<<<<<<<<=<;<;7<&
chr1 273 T 23  ,.....,.,.,.....,.,..A <<<;<<<<<<<<3<=<<<;<<+
chr1 274 T 23  ,.$.....,.,.....,.,...    7<7;<;<<<<<<<<<=<;<;<<6
chr1 275 A 23  ,$....,.,.,.....,.,...^l.  <+;9*<<<<<<<<<=<<:;<<<<
chr1 276 G 22  TTTTTTTTTTTTTTTTTTTTTTT  33;+<<7=7<<7<&<<1;<<6<
chr1 277 T 22  ....,.,.,.C.,.,.,..G.  +7<;<<<<<<<&<=<<:;<<&<
chr1 278 G 23  ....,.,.,.....,.,....^k.   %38*<<;<7<<7<=<<<;<<<<<
chr1 279 C 23  A..T,,.,.,.....,.,..... ;75&<<<<<<<<<=<<<9<<:<<
```

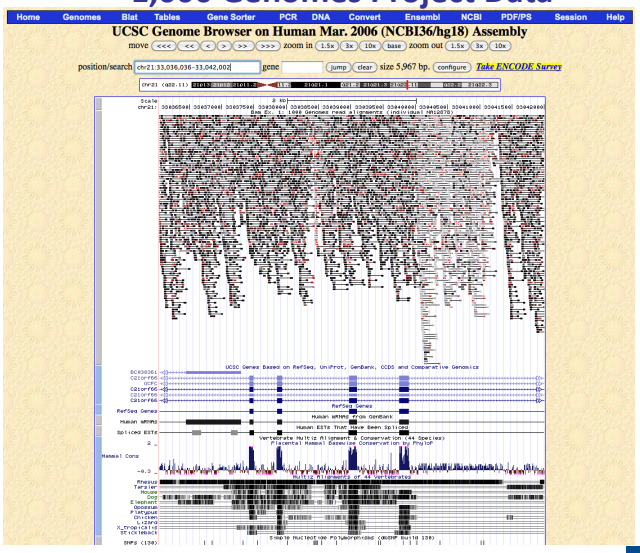

The Genome Access Course

Binary Alignment (BAM) Files

- Common file format to store reads and their alignment to a reference sequence
  - Generated by most next gen analysis software
    - samtools software package
- UCSC Genome Browser and Ensembl can display them as a custom track
  - IGV from Broad very useful

The Genome Access Course

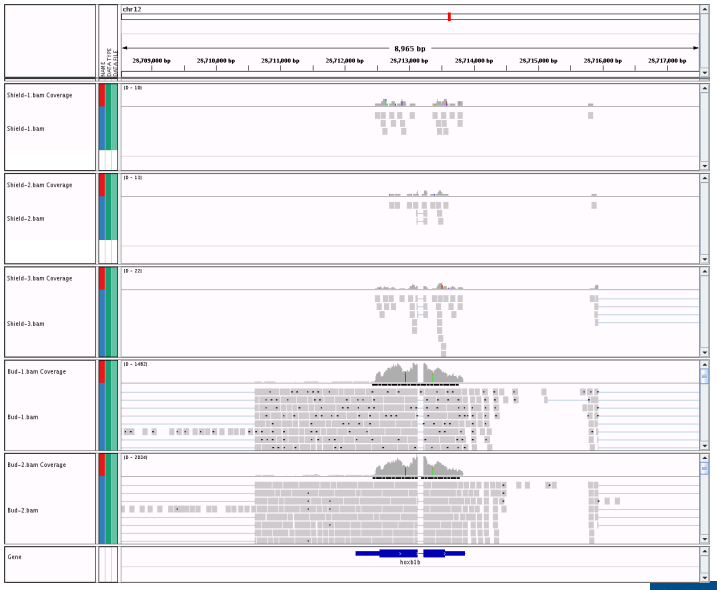
UCSC Genome Browser with 1,000 Genomes Project Data



April 2015

The Genome Access Course

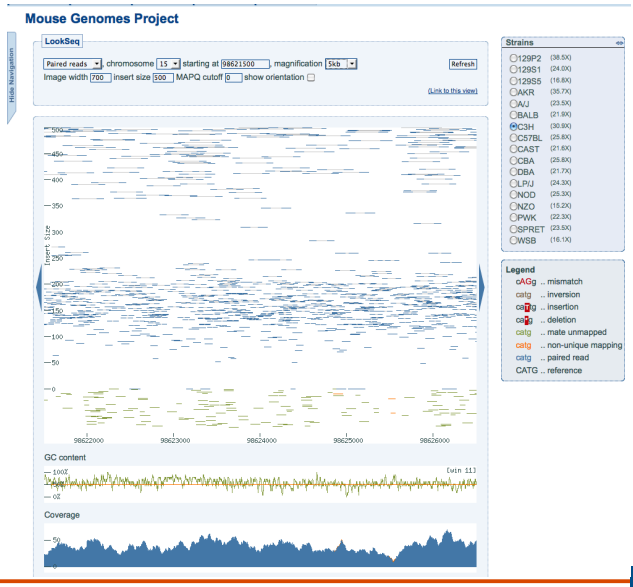
Integrated Genomics Viewer (IGV)



April 2015

The Genome Access Course

LookSeq at Sanger Mouse Genomes Project



April 2015

The Genome Access Course

Glo1 CNV Present in Mouse Genomes Data for A/J

Proximal Flank	Glo1 Locus	Distal Flank
Chr17: 30.5Mb	Chr17: 30.7Mb	Chr17: 31.2Mb
Max ~50x coverage	Max >100x coverage	Max ~50x coverage

50kb 50kb 50kb

April 2015

The Genome Access Course

Glo1 CNV Not Present in Mouse Genomes Data for NZO

Proximal Flank	Glo1 Locus	Distal Flank
Chr17: 30.5Mb	Chr17: 30.7Mb	Chr17: 31.2Mb
Max ~25x coverage	Max ~25x coverage	Max ~25x coverage

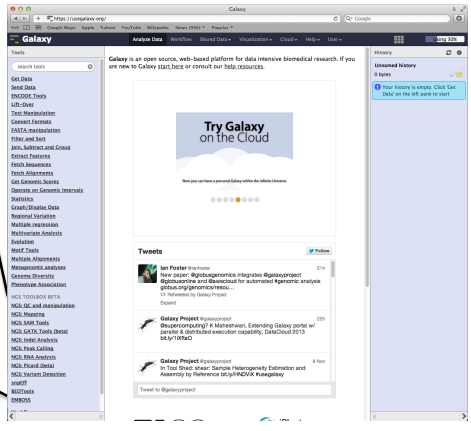
50kb 50kb 50kb

April 2015

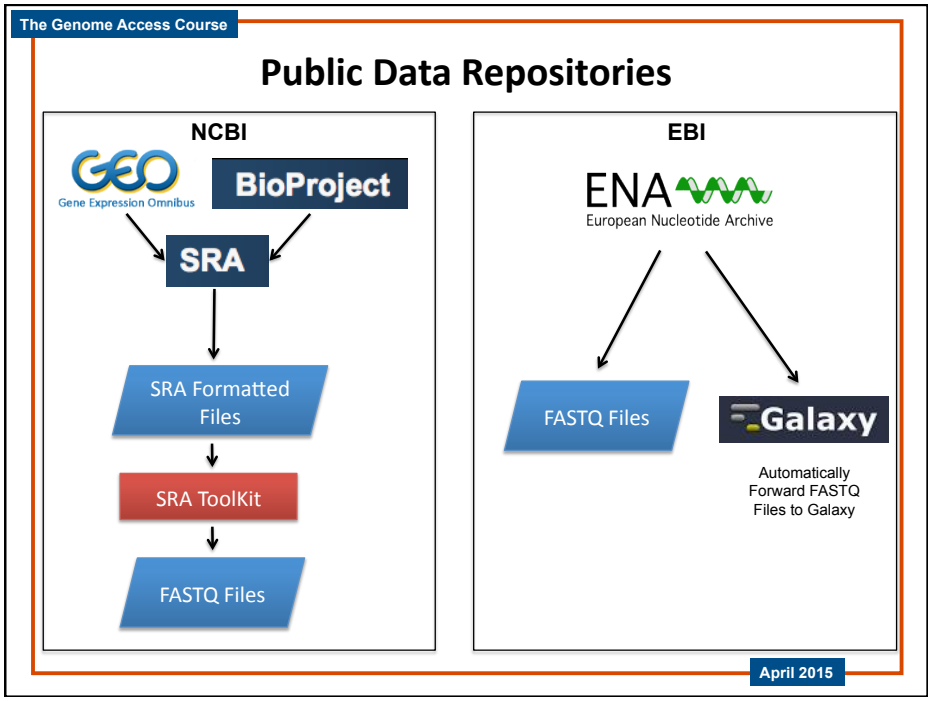
The Genome Access Course

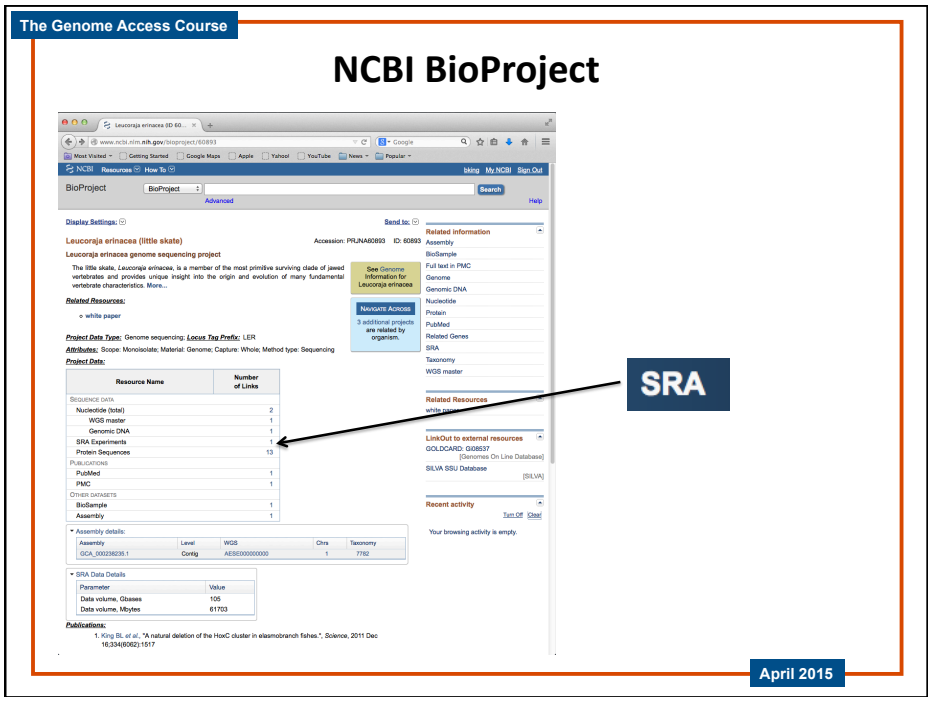
Galaxy (http://main.g2.bx.psu.edu)

NGS TOOLBOX BETA
NGS: QC and manipulation
NGS: Mapping
NGS: SAM Tools
NGS: GATK Tools (beta)
NGS: Indel Analysis
NGS: Peak Calling
NGS: RNA Analysis
NGS: Picard (beta)
NGS: Variant Detection
snpEff
BEDTools
EMBOSS



April 2015


The Genome Access Course
Public Data Repositories
NCBI
Gene Expression Omnibus
BioProject
SRA
SRA Formatted Files
SRA ToolKit
FASTQ Files
EBI
ENA
European Nucleotide Archive
FASTQ Files
Galaxy
Automatically Forward FASTQ Files to Galaxy
April 2015


The Genome Access Course
NCBI BioProject
SRA
April 2015

The Genome Access Course

NCBI Gene Expression Omnibus

April 2015

The Genome Access Course

Overall Analysis Workflow

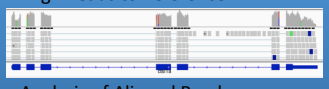
FASTQ Files

Gene Expression Omnibus

ENA European Nucleotide Archive

Secondary Analysis

1. Read Preprocessing & Diagnostics
2. Align Reads to Reference



3. Analysis of Aligned Reads
   e.g., Read counts per gene from RNA-Seq

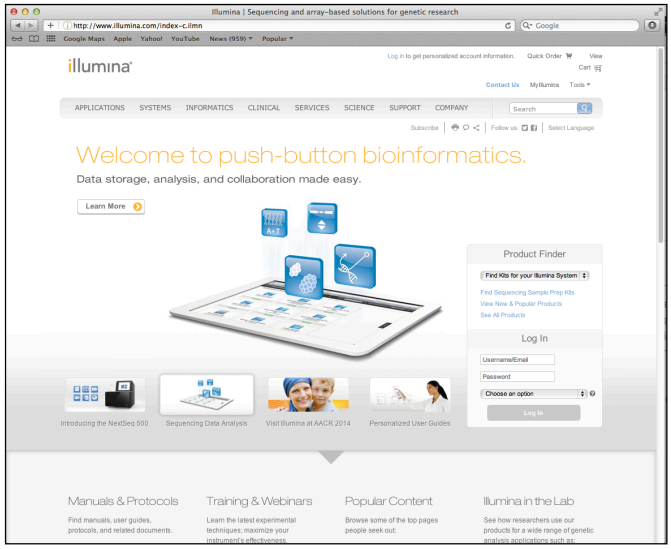
Tertiary Analysis

1. Analysis of Read Counts
   e.g., Differentially expressed genes
2. Analysis of Gene Lists
   1. Enrichment
   2. Pathway and networks
3. Analysis of Expression Patterns

April 2015

The Genome Access Course

Push-Button Bioinformatics … Be Careful



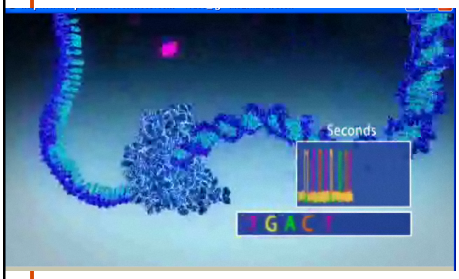
April 2015

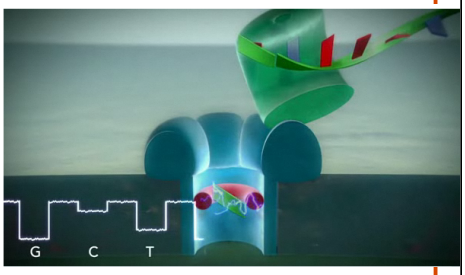
The Genome Access Course

Third Generation Sequencing

PACIFIC BIOSCIENCES

Oxford NANOPORE Technologies



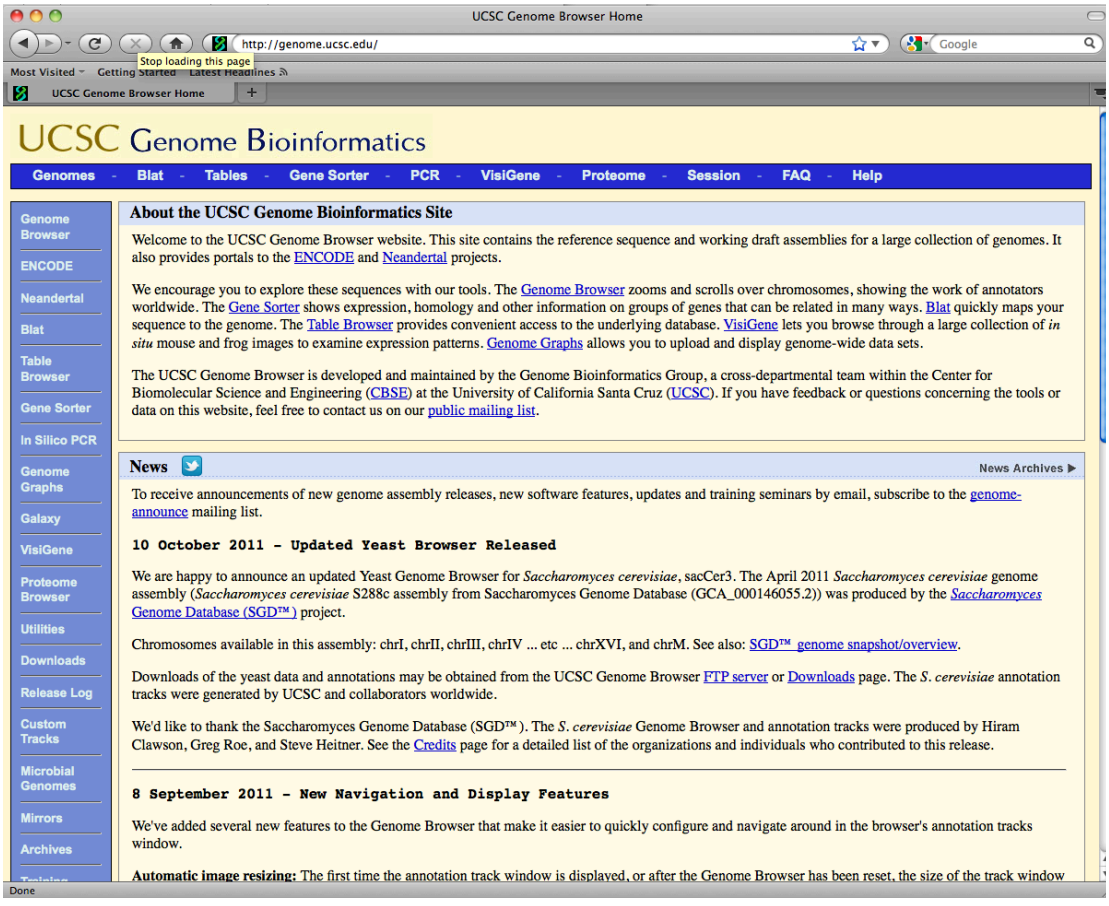


April 2015

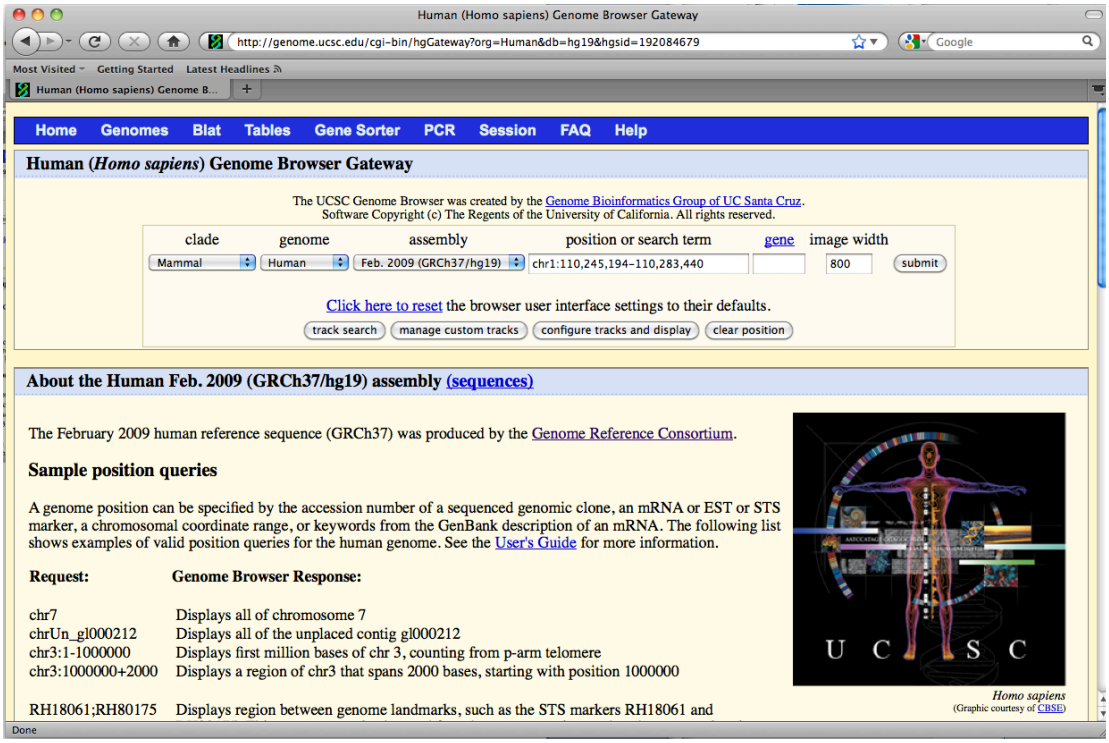
Worked Example #1. In this example, you will view STAT1 **ChIP-Seq data** from interferon-gamma stimulated and unstimulated human HeLa S3 cells in the **UCSC Genome Browser** described in the following paper. Specifically, you will examine the *CSF1* locus and look for consistency between this study and curated data in the ORegAnno database (http://www.oreganno.org).

> Robertson G, Hirst M, Bainbridge M, Bilenky M, Zhao Y, Zeng T, Euskirchen G, Bernier B, Varhol R, Delaney A, Thiessen N, Griffith OL, He A, Marra M, Snyder M, Jones S. Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nat Methods*. 2007 Aug;4(8):651-7.

STEP 1: Go to the UCSC Genome Browser home page (http://genome.ucsc.edu) and click on the "Genome Browser" link.



STEP 2: Click on the “add custom tracks” button near the top of the screen to add a custom track.

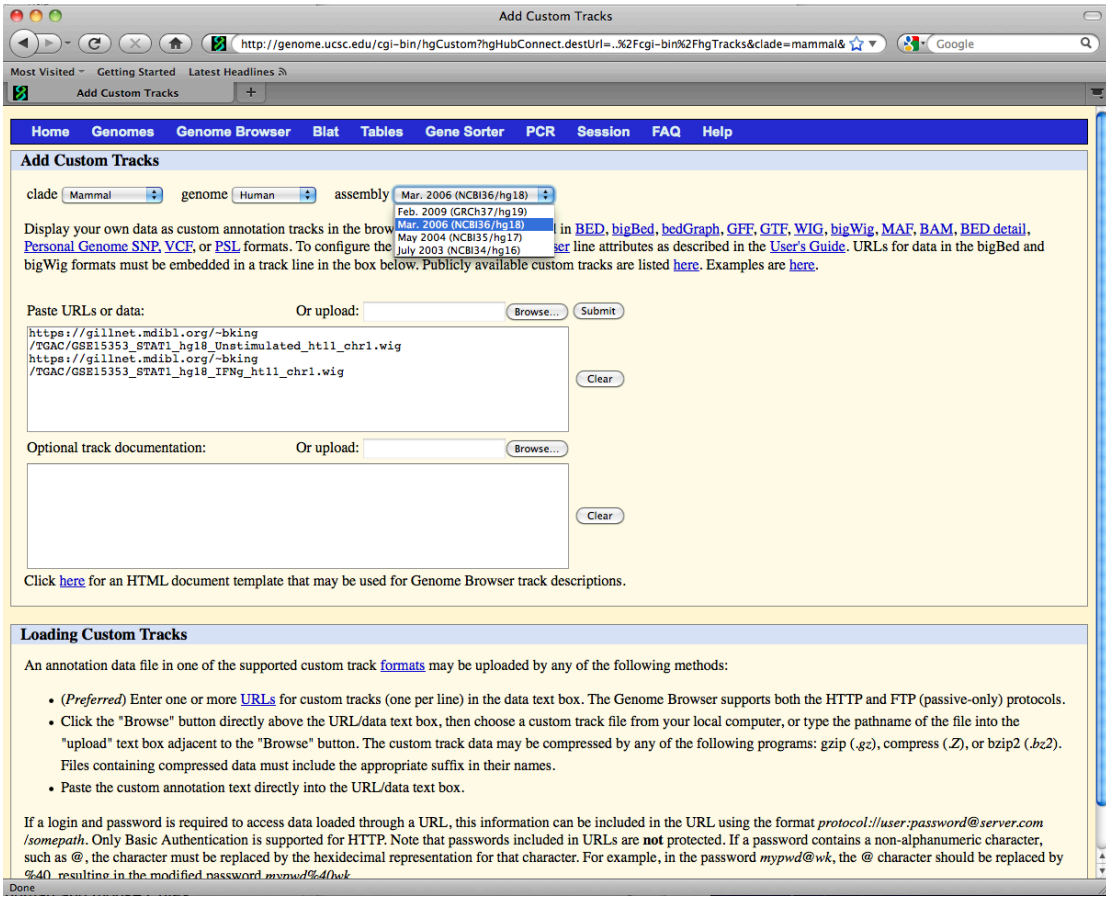


STEP 3: On this page, you must select the NCBI Build 36 human genome assembly and specify the URL of where the two .WIG files are located. The GEO (Gene Expression Omnibus) record GSE15353 also has .WIG files. However, the .WIG files that I've posted contain only the data for Chr. 1 to make them smaller in size.

```
https://gillnet.mdibl.org/~bking/TGAC/GSE15353_STAT1_hg18_Unstimulated_ht11_chr1.wig
https://gillnet.mdibl.org/~bking/TGAC/GSE15353_STAT1_hg18_IFNg_ht11_chr1.wig
```

We will specifically look at Chr. 1 as it is where the CSF1 locus is located.

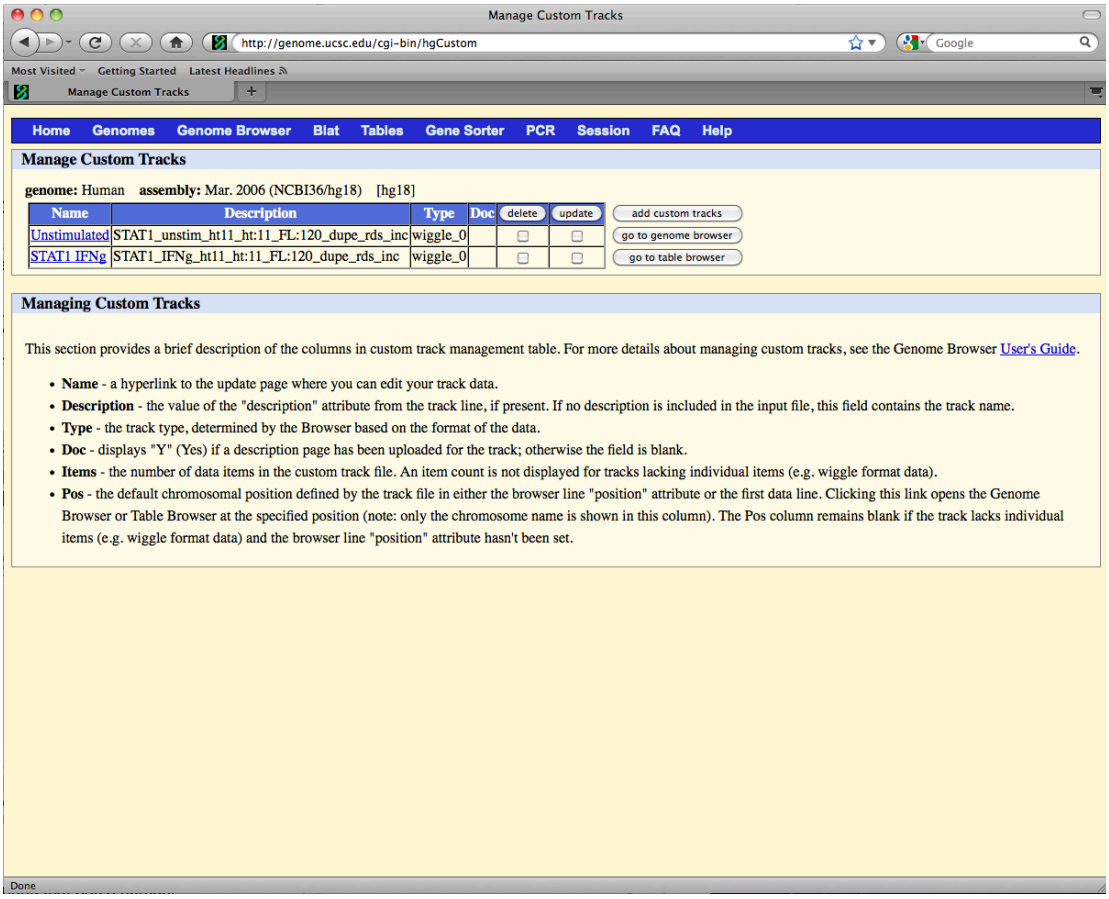
Then, click the "Submit" button.



In case you are interested, there is a web site in Switzerland that has a number of other .WIG files (technically bigWIG format) for other human and mouse ChIP-Seq experiments. You can view any of them by repeating this step and entering in a different URL.

http://ccg.vital-it.ch/chipseq/chip_seq_wig.html

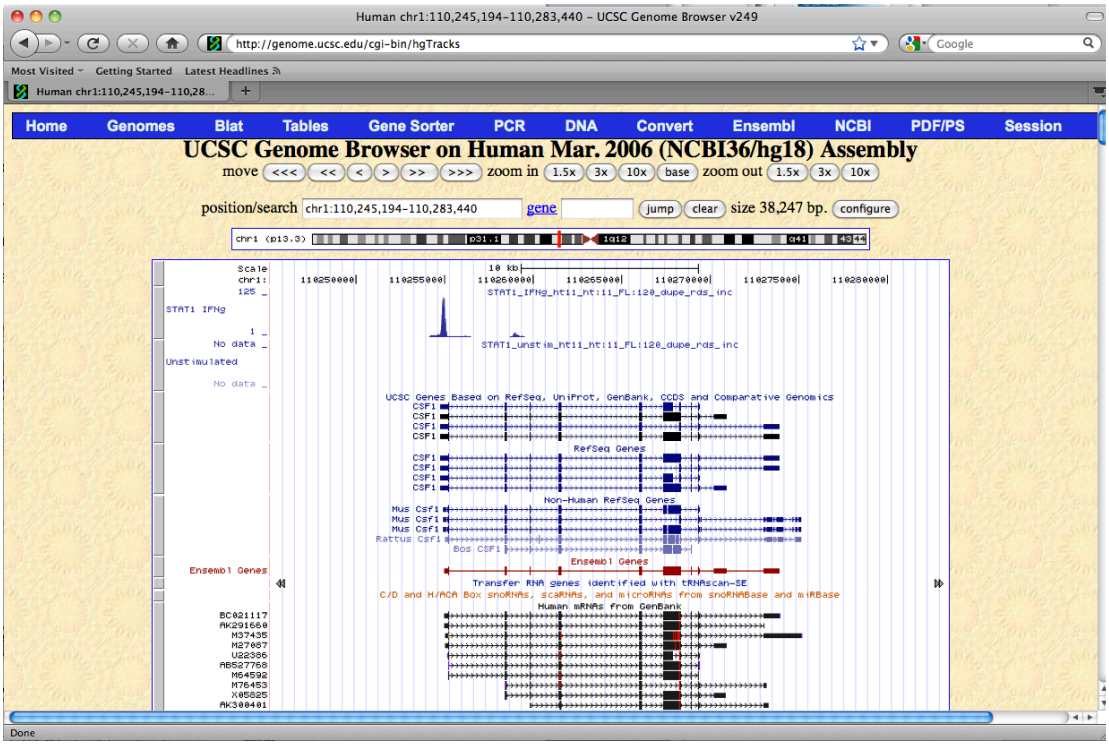
STEP 4: On the resulting page, you will see that two new tracks have been added. Click on the “go to genome browser” button to view the tracks.



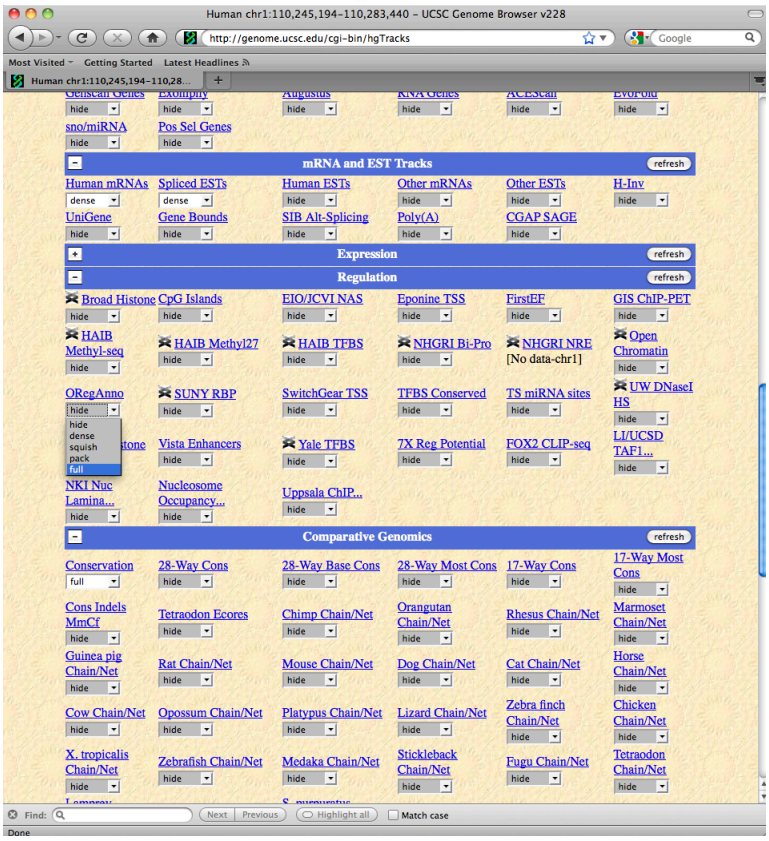
STEP 5: On the next page, you will want to advance to *CSF1* locus:

`chr1:110,245,194-110,283,440`

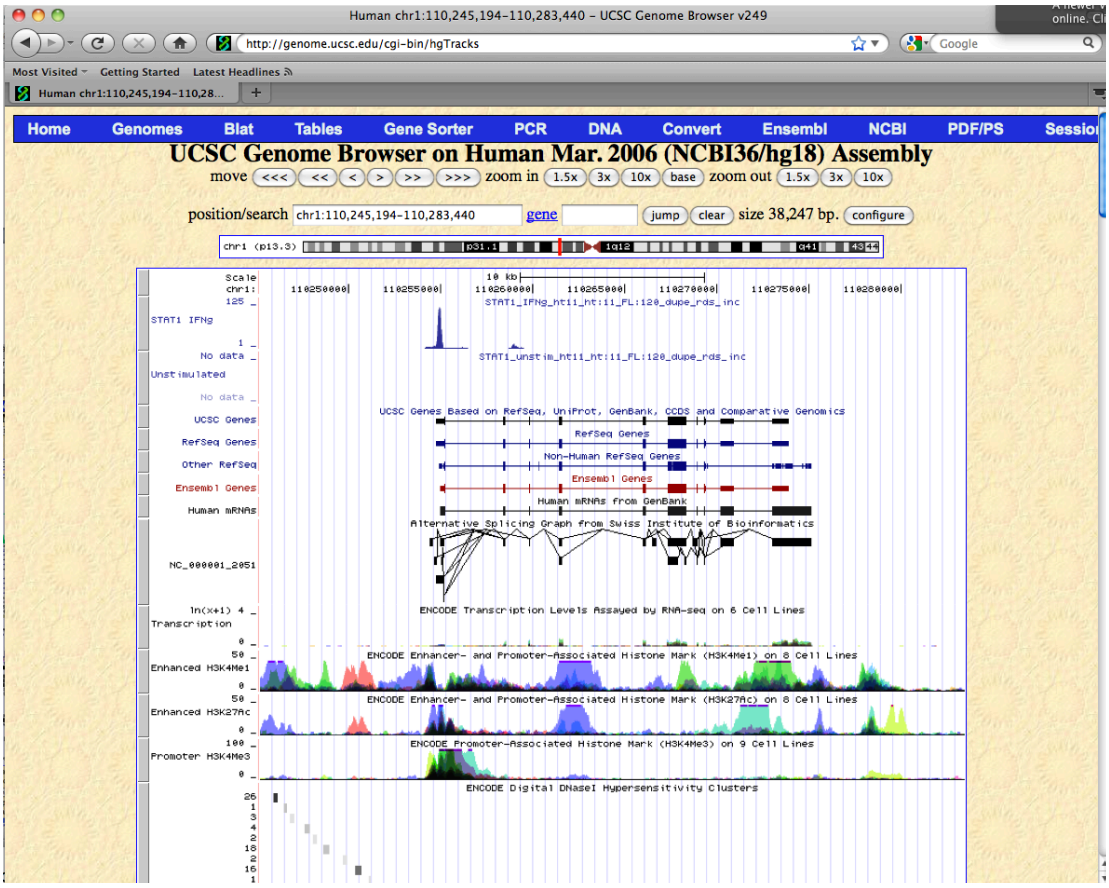
Paste the above chromosome coordinate range in the postion/search text field. Here, you will observe a peak in the promoter of *CSF1* from the stimulated sample with a value near 125 whereas the unstimulated sample has no signal.



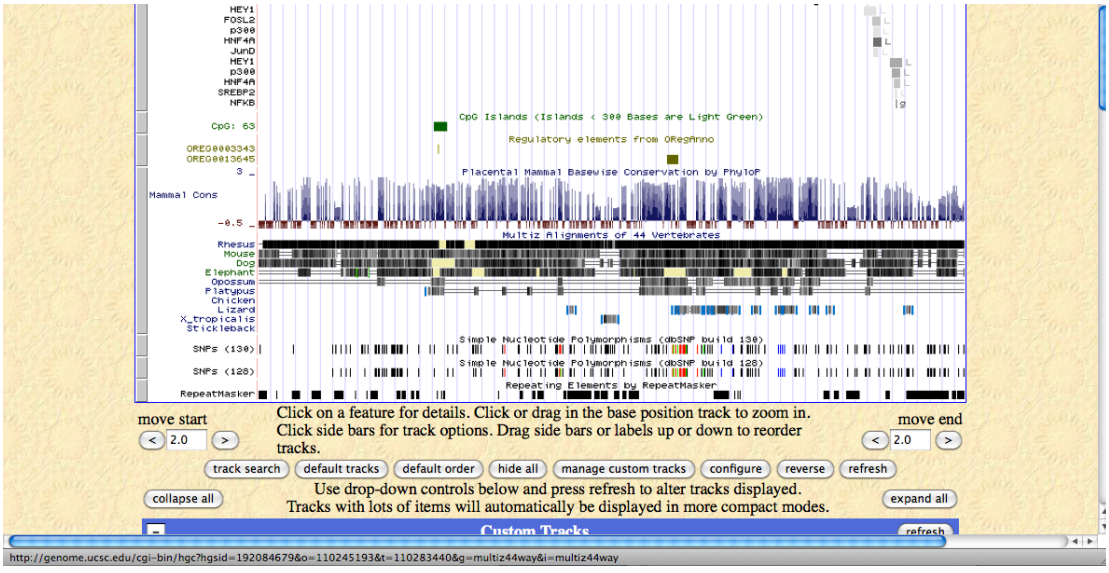
STEP 6: Next, scroll down towards the bottom of the page to turn on the ORegAnno track under the “Regulation” section of tracks. Select “full” from the pull-down menu and then click the “refresh” button.



STEP 7: Next, you will see that there is a ORegAnno feature underneath the peak from the stimulated ChIP-Seq study.

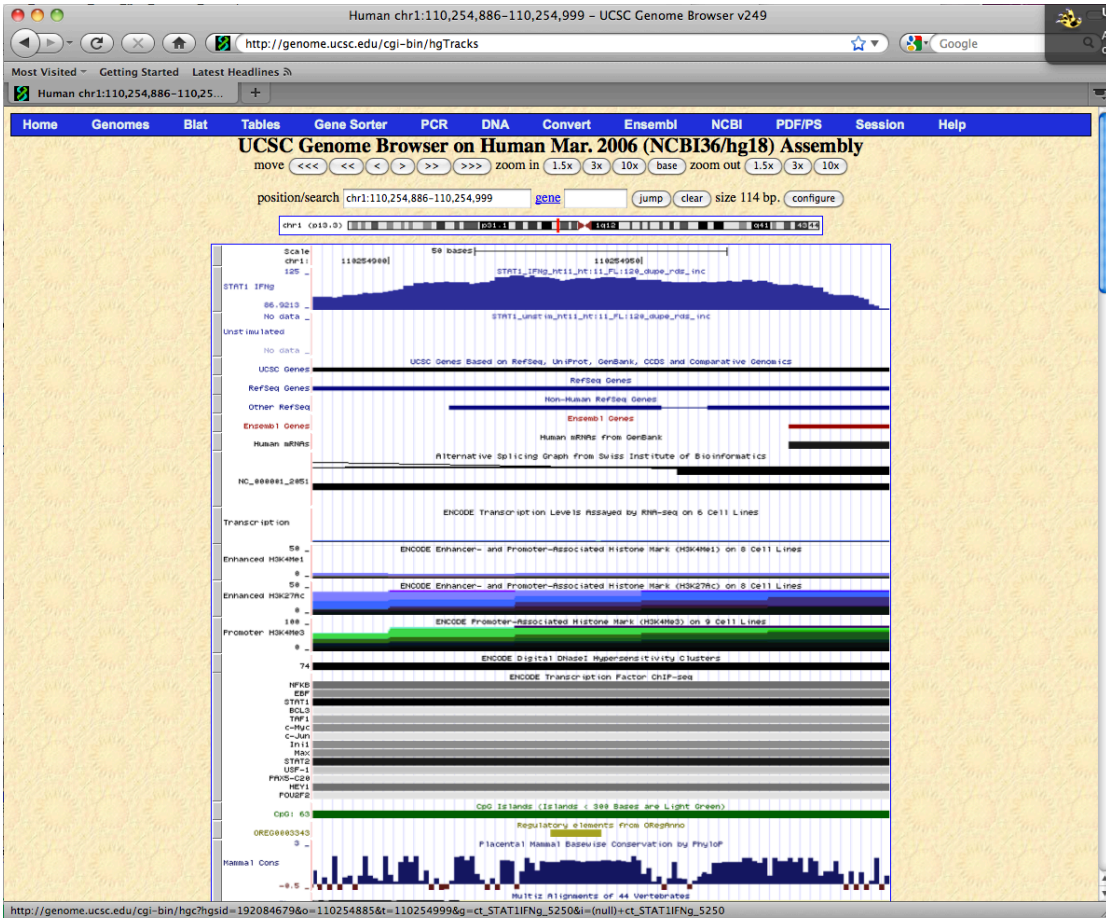


… keep scrolling down …



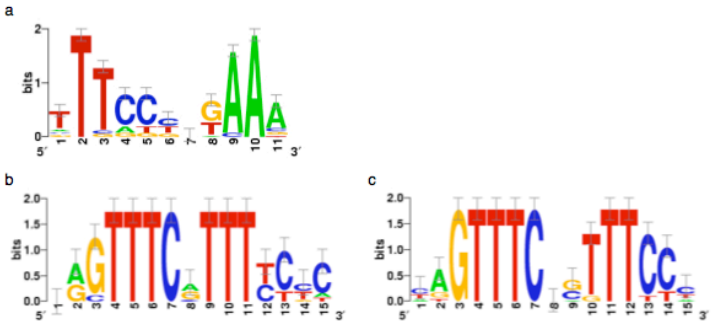
You will need to scroll into this coordinate range to see the feature closely.

chr1:110,254,886-110,254,999



In the paper, they detail the STAT1 binding motif. If you zoom into the base level in the browser, you may be able to locate it.

Sequence logos for functional STAT1 binding sites

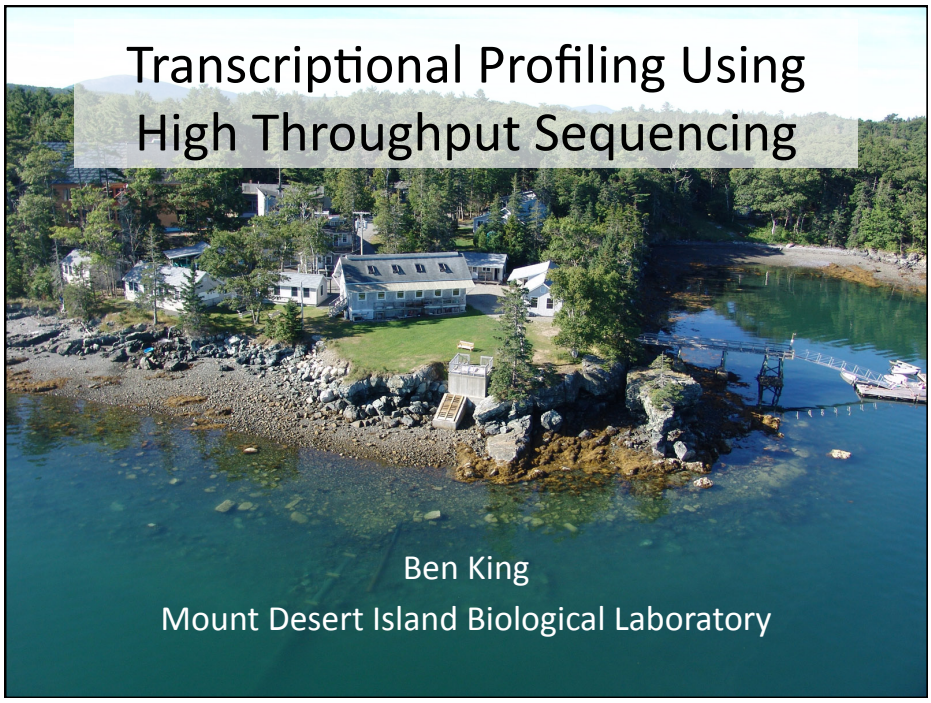


Supplementary Fig. 6 Sequence logos for sites known from the literature to be functional. a) 23 GAS sites, b) 9 sequences for known ISRE sites with a short spacer between conserved dimer halves ('ISRE_2'), and c) 10 ISRE sequences with a longer spacer ('ISRE_3'). Sequence logos were generated by WebLogo.[1] See Supplementary Data.

The Genome Access Course

RNA-Seq Data Analysis

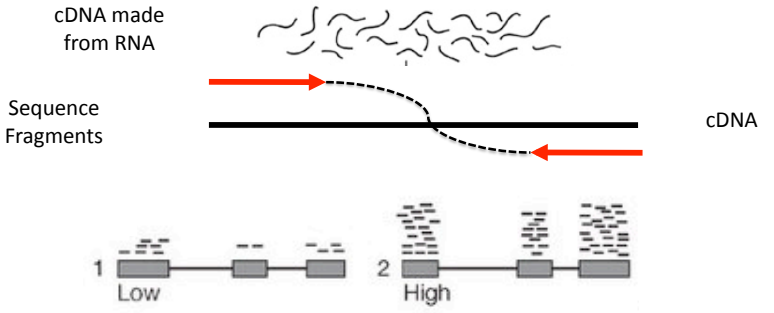
Transcriptional Profiling Using High Throughput Sequencing



Ben King
Mount Desert Island Biological Laboratory

The Genome Access Course

RNA Sequencing (RNA-Seq)



Garber *et al, Nat Methods* (2011)

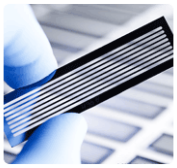
1. Characterize all RNA in sample
2. Gene expression level proportional to number of reads
3. Detect alternatively spliced transcripts

April 2015

The Genome Access Course

Experimental Design

Flow Cell Lanes



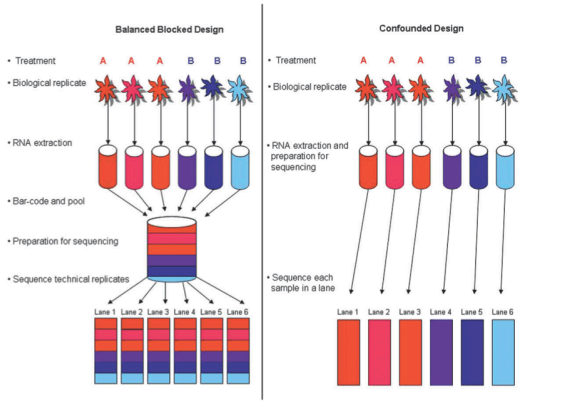
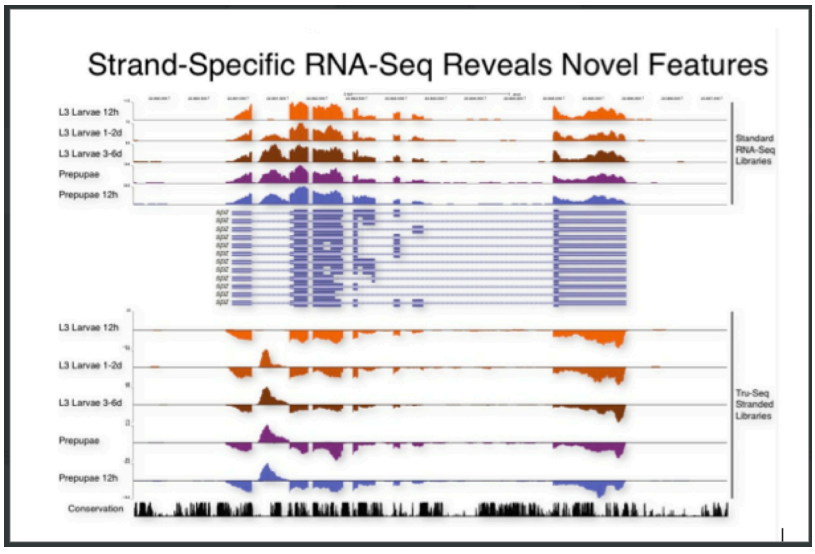


FIGURE 4.—Comparison of two designs for testing differential expression between treatments A and B. Treatment A is denoted by red tones and treatment B by blue tones. In the ideal balanced block design (left), six samples ($m = 6$) are bar coded, pooled, and processed together. The pool is then divided into six equal portions that are input to six lanes ($L = 6$) of the flow cell. Bar coding in the balanced block design results in six technical replicates ($T = 6$) of each sample, while balancing batch and lane effects and blocking on lane. The balanced block design also allows partitioning of batch and lane effects from the within-group biological variability. The confounded design (right) represents a typical RNA-Seq experiment and consists of the same six samples, with no bar coding, and does not permit partitioning of batch and lane effects from the estimate of within-group biological variability.

Auer & Doerge *Genetics* (2010) 185: 405-416

April 2015

The Genome Access Course



Taken from GIGA Newsletter 13 – Universite de Liège

April 2015

The Genome Access Course

Analysis Overview

FASTQ Files

GEO Gene Expression Omnibus

ENA European Nucleotide Archive

Secondary Analysis

1. Read Preprocessing & Diagnostics
2. Align Reads to Reference

3. Analysis of Aligned Reads
   e.g., Read counts per gene from RNA-Seq

Tertiary Analysis

1. Analysis of Read Counts
   e.g., Differentially expressed genes
2. Analysis of Gene Lists
   1. Enrichment
   2. Pathway and networks
3. Analysis of Expression Patterns

Linux

INGENUITY PATHWAY ANALYSIS

amazon web services

Galaxy

igv Integrative Genomics Viewer

UCSC Genome Bioinformatics

April 2015

The Genome Access Course

What is the Best Expression Metric?

Considerations

Number of reads for a gene depends on:

- Expression Level (hopefully)
- Number of reads you generated for your sample
- How well you can map your read to your reference
  - → Alternative splicing
- Length of the transcript/gene

Approaches

RPKM (single-end)
FPKM (paired-end)

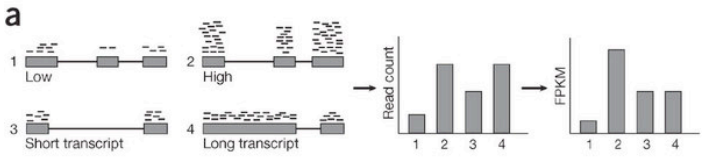
Counts

Proportion of counts per gene to all reads per sample

April 2015

The Genome Access Course

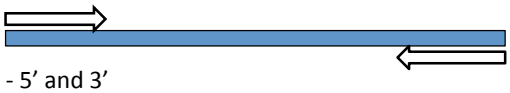
RPKM and FPKM



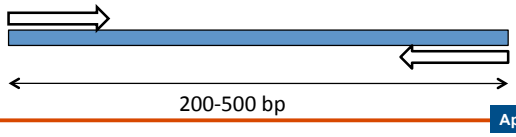
Reads/Fragments Per Kilobase of exon Model (RPKM, FPKM)

$$RPKM_i = \frac{n_i \bullet 1{,}000{,}000}{N} \bullet \frac{1{,}000}{l_i}$$

Single-End Reads - 5' or 3' (random)



Paired-End Reads - 5' and 3'

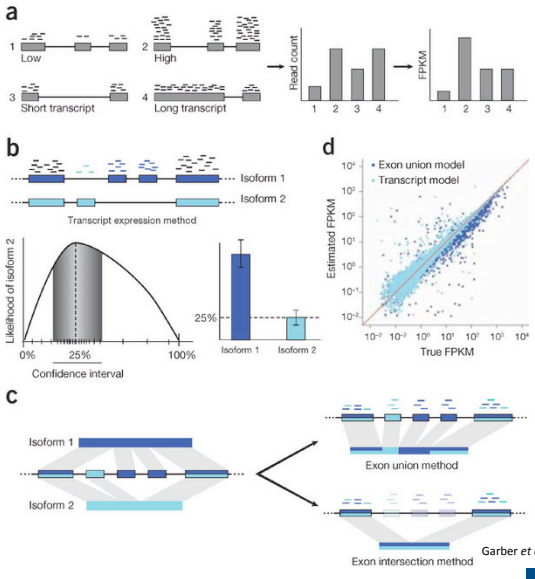


April 2015

The Genome Access Course

Tuxedo Suite RNA-seq Analysis Workflow

1. TopHat
2. Cufflinks
3. Cuffmerge
4. Cuffdiff
5. CummeRbund

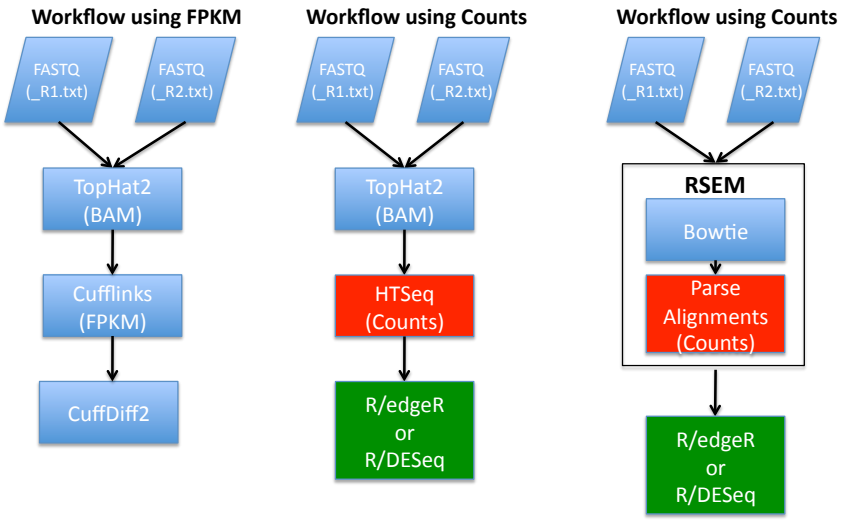


Garber *et al*, *Nat Methods* (2011)

April 2015

The Genome Access Course

RNA-Seq Analysis Workflows



April 2015

The Genome Access Course

Obtaining Counts per Gene from BAM File Using HTSeq

HTSeq – A Python framework to work with high-throughput sequencing data

Simon Anders, Paul Theodor Pyl and Wolfgang Huber
Genome Biology Unit, European Molecular Biology Laboratory, 69111 Heidelberg, Germany
2014-Aug-13

ABSTRACT
Motivation: A large choice of tools exists for many standard tasks in the analysis of high-throughput sequencing (HTS) data. However, once a project deviates from standard work flows, custom scripts are needed.
Results: We present HTSeq, a Python library to facilitate the rapid development of such scripts. HTSeq offers parsers for many common data formats in HTS projects, as well as classes to represent data such as genomic coordinates, sequences, sequencing reads, alignments, gene model information, variant calls, and provides data structures that allow for querying via genomic coordinates. We also present htseq-count, a tool developed with HTSeq that preprocesses RNA-Seq data for differential expression analysis by counting the overlap of reads with genes.
Availability: HTSeq is released as open-source software under the GNU General Public Licence and available from http://www-huber.embl.de/HTSeq or from the Python Package Index https://pypi.python.org/pypi/HTSeq.
Contact: sanders@fs.tum.de

found considerable use in the research community. The present article provides a description of the package and also reports on recent improvements.

HTSeq comes with extensive documentation, including a tutorial that demonstrates the use of the core classes of HTSeq and discusses several important use cases in detail. The documentation, as well as HTSeq's design, is geared towards allowing users with only moderate Python knowledge to create their own scripts, while shielding more advanced internals from the user.

2 COMPONENTS AND DESIGN OF HTSEQ

2.1 Parser and record objects

HTSeq provides parsers for reference sequences (FASTA), short reads (FASTQ), short-read alignments (the SAM/BAM format and some legacy formats), and for genomic feature, annotation and score data (GFF/GTF, VCF, BED, Wiggle).

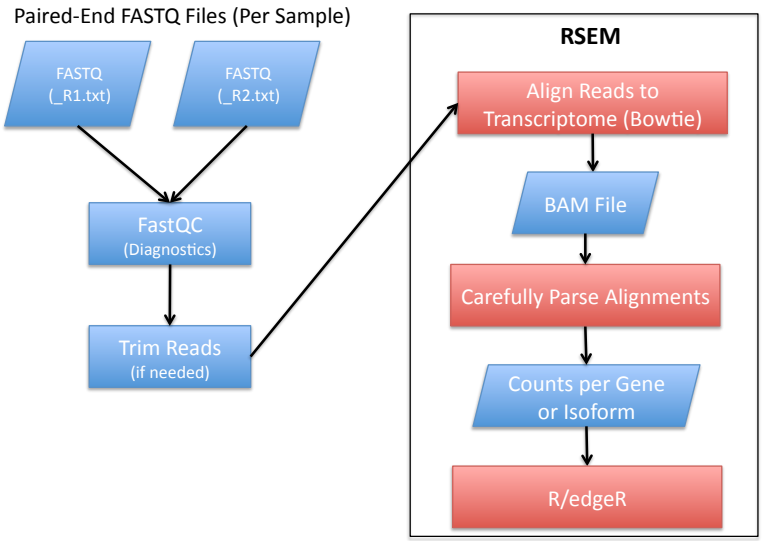
Each parser is provided as a class which whose objects are tied to a file name or open file or stream and work as iterator

bioRχiv beta
THE PREPRINT SERVER FOR BIOLOGY

April 2015

The Genome Access Course

An Alternative to Tuxedo Suite: RSEM



April 2015

The Genome Access Course

Using Counts: R/edgeR Package

BIOINFORMATICS APPLICATIONS NOTE Vol. 26 no. 1 2010, pages 139–140
doi:10.1093/bioinformatics/btp616

Gene expression

edgeR: a Bioconductor package for differential expression analysis of digital gene expression data

Mark D. Robinson[1,2,*,†], Davis J. McCarthy[2,†] and Gordon K. Smyth[2]

[†]Cancer Program, Garvan Institute of Medical Research, 384 Victoria Street, Darlinghurst, NSW 2010 and [2]Bioinformatics Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3052, Australia

Received on March 29, 2009; revised on October 19, 2009; accepted on October 23, 2009
Advance Access publication November 11, 2009
Associate Editor: Joaquin Dopazo

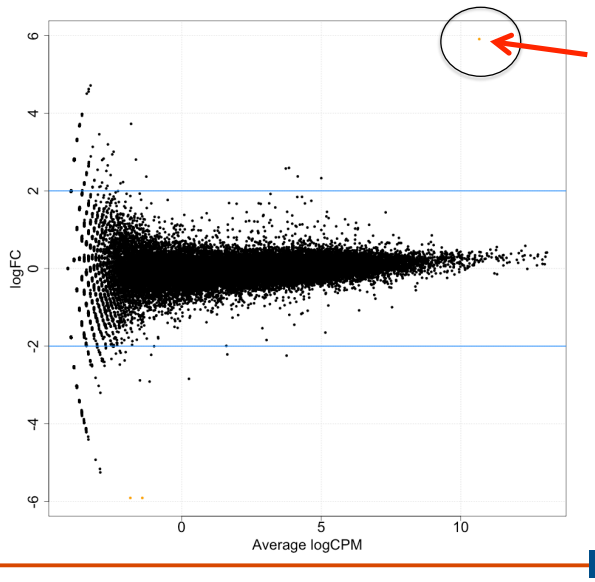
- RNA-seq
- small RNA-seq (e.g., microRNAs)

Many other packages as well: R/DESeq, …

April 2015

The Genome Access Course

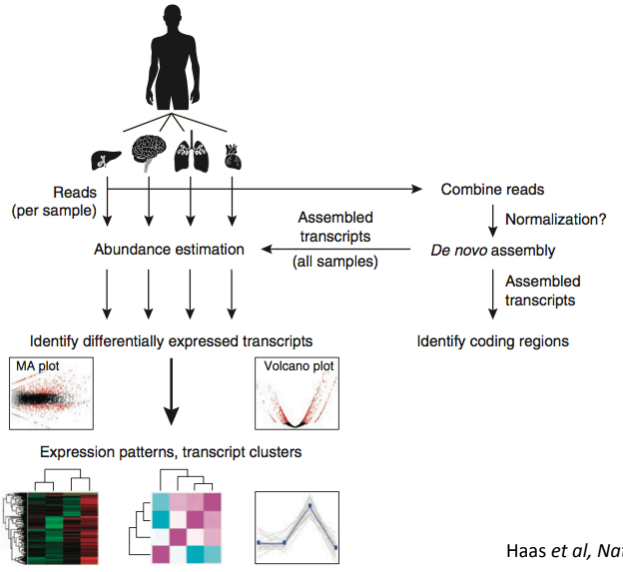
"Defaults" Don't Always Work



April 2015

The Genome Access Course

Trinity *de novo* RNA-Seq Workflow



Haas *et al, Nature Protocols* (2013)

April 2015

The Genome Access Course

View RNA-Seq Data

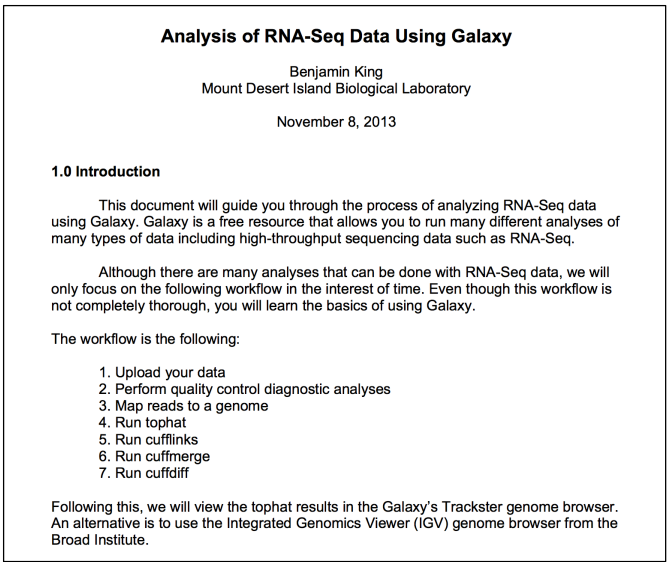
- Ensembl
  - Aligned zebrafish developmental profiling
    - hoxb1b (2 cell, 1dpf, 2dpf)
- UCSC Genome Browser
  - Human RNA-Seq BAM files (my own files)
    - CRISPLD2

http://applbio.mdibl.org/BAM/untreated_rep1_508_sorted_pos.bam
http://applbio.mdibl.org/BAM/alb_dex_rep1_511_sorted_pos.bam

April 2015

The Genome Access Course

RNA-Seq in Galaxy



Analysis of RNA-Seq Data Using Galaxy

Benjamin King
Mount Desert Island Biological Laboratory

November 8, 2013

1.0 Introduction

This document will guide you through the process of analyzing RNA-Seq data using Galaxy. Galaxy is a free resource that allows you to run many different analyses of many types of data including high-throughput sequencing data such as RNA-Seq.

Although there are many analyses that can be done with RNA-Seq data, we will only focus on the following workflow in the interest of time. Even though this workflow is not completely thorough, you will learn the basics of using Galaxy.

The workflow is the following:

1. Upload your data
2. Perform quality control diagnostic analyses
3. Map reads to a genome
4. Run tophat
5. Run cufflinks
6. Run cuffmerge
7. Run cuffdiff

Following this, we will view the tophat results in the Galaxy's Trackster genome browser. An alternative is to use the Integrated Genomics Viewer (IGV) genome browser from the Broad Institute.

April 2015

Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks

Cole Trapnell[1,2], Adam Roberts[3], Loyal Goff[1,2,4], Geo Pertea[5,6], Daehwan Kim[5,7], David R Kelley[1,2], Harold Pimentel[3], Steven L Salzberg[5,6], John L Rinn[1,2] & Lior Pachter[3,8,9]

[1]Broad Institute of MIT and Harvard, Cambridge, Massachusetts, USA. [2]Department of Stem Cell and Regenerative Biology, Harvard University, Cambridge, Massachusetts, USA. [3]Department of Computer Science, University of California, Berkeley, California, USA. [4]Computer Science and Artificial Intelligence Lab, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA. [5]Department of Medicine, McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University School of Medicine, Baltimore, Maryland, USA. [6]Department of Biostatistics, Johns Hopkins University, Baltimore, Maryland, USA. [7]Center for Bioinformatics and Computational Biology, University of Maryland, College Park, Maryland, USA. [8]Department of Mathematics, University of California, Berkeley, California, USA. [9]Department of Molecular and Cell Biology, University of California, Berkeley, California, USA. Correspondence should be addressed to C.T. (cole@broadinstitute.org).

Published online 1 March 2012; doi:10.1038/nprot.2012.016

Recent advances in high-throughput cDNA sequencing (RNA-seq) can reveal new genes and splice variants and quantify expression genome-wide in a single assay. The volume and complexity of data from RNA-seq experiments necessitate scalable, fast and mathematically principled analysis software. TopHat and Cufflinks are free, open-source software tools for gene discovery and comprehensive expression analysis of high-throughput mRNA sequencing (RNA-seq) data. Together, they allow biologists to identify new genes and new splice variants of known ones, as well as compare gene and transcript expression under two or more conditions. This protocol describes in detail how to use TopHat and Cufflinks to perform such analyses. It also covers several accessory tools and utilities that aid in managing data, including CummeRbund, a tool for visualizing RNA-seq analysis results. Although the procedure assumes basic informatics skills, these tools assume little to no background with RNA-seq analysis and are meant for novices and experts alike. The protocol begins with raw sequencing reads and produces a transcriptome assembly, lists of differentially expressed and regulated genes and transcripts, and publication-quality visualizations of analysis results. The protocol's execution time depends on the volume of transcriptome sequencing data and available computing resources but takes less than 1 d of computer time for typical experiments and ~1 h of hands-on time.

INTRODUCTION

High-throughput mRNA sequencing (RNA-seq) offers the ability to discover new genes and transcripts and measure transcript expression in a single assay[1–3]. However, even small RNA-seq experiments involving only a single sample produce enormous volumes of raw sequencing reads—current instruments generate more than 500 gigabases in a single run. Moreover, sequencing costs are reducing exponentially, opening the door to affordable personalized sequencing and inviting comparisons with commodity computing and its impact on society[4]. Although the volume of data from RNA-seq experiments is often burdensome, it can provide enormous insight. Just as cDNA sequencing with Sanger sequencers drastically expanded our catalog of known human genes[5], RNA-seq reveals the full repertoire of alternative splice isoforms in our transcriptome and sheds light on the rarest and most cell- and context-specific transcripts[6]. Furthermore, because the number of reads produced from an RNA transcript is a function of that transcript's abundance, read density can be used to measure transcript[7,8] and gene[2,3,9,10] expression with comparable or superior accuracy to expression microarrays[1,11].

RNA-seq experiments must be analyzed with robust, efficient and statistically principled algorithms. Fortunately, the bioinformatics community has been hard at work developing mathematics, statistics and computer science for RNA-seq and building these ideas into software tools (for a recent review of analysis concepts and software packages see Garber *et al.*[12]). RNA-seq analysis tools generally fall into three categories: (i) those for read alignment; (ii) those for transcript assembly or genome annotation; and (iii) those for transcript and gene quantification. We have developed two popular tools that together serve all three roles, as well as a newer tool for visualizing analysis results. TopHat[13] (http://tophat.cbcb.umd.edu/) aligns reads to the genome and discovers transcript splice sites. These alignments are used during downstream analysis in several ways. Cufflinks[8] (http://cufflinks.cbcb.umd.edu/) uses this map against the genome to assemble the reads into transcripts. Cuffdiff, a part of the Cufflinks package, takes the aligned reads from two or more conditions and reports genes and transcripts that are differentially expressed using a rigorous statistical analysis. These tools are gaining wide acceptance and have been used in a number of recent high-resolution transcriptome studies[14–17]. CummeRbund renders Cuffdiff output in publication-ready figures and plots. **Figure 1** shows the software used in this protocol and highlights the main functions of each tool. All tools used in the protocol are fully documented on the web, actively maintained by a team of developers and adopt well-accepted data storage and transfer standards.

Limitations of the protocol and software

TopHat and Cufflinks do not address all applications of RNA-seq, nor are they the only tools for RNA-seq analysis. In particular, TopHat and Cufflinks require a sequenced genome (see below for references to tools that can be used without a reference genome). This protocol also assumes that RNA-seq was performed with either Illumina or SOLiD sequencing machines. Other sequencing technologies such as 454 or the classic capillary electrophoresis approach can be used for large-scale cDNA sequencing, but analysis of such data is substantially different from the approach used here.

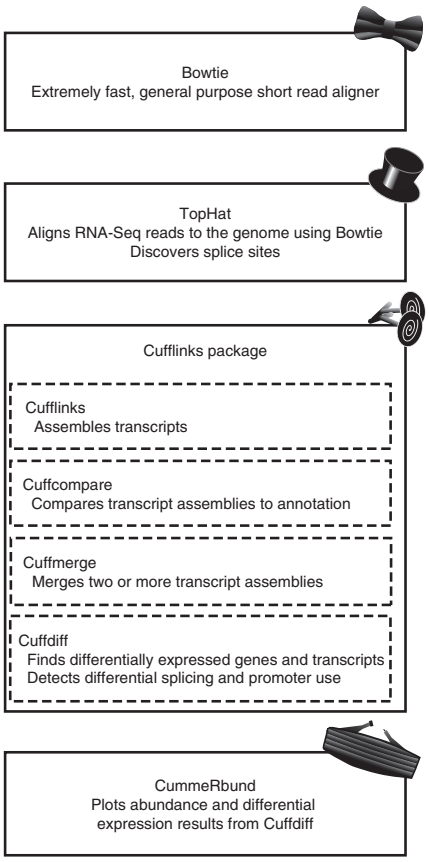


Figure 1 | Software components used in this protocol. Bowtie[33] forms the algorithmic core of TopHat, which aligns millions of RNA-seq reads to the genome per CPU hour. TopHat's read alignments are assembled by Cufflinks and its associated utility program to produce a transcriptome annotation of the genome. Cuffdiff quantifies this transcriptome across multiple conditions using the TopHat read alignments. CummeRbund helps users rapidly explore and visualize the gene expression data produced by Cuffdiff, including differentially expressed genes and transcripts.

TopHat and Cufflinks are both operated through the UNIX shell. No graphical user interface is included. However, there are now commercial products and open-source interfaces to these and other RNA-seq analysis tools. For example, the Galaxy Project[18] uses a web interface to cloud computing resources to bring command-line–driven tools such as TopHat and Cufflinks to users without UNIX skills through the web and the computing cloud.

Alternative analysis packages

TopHat and Cufflinks provide a complete RNA-seq workflow, but there are other RNA-seq analysis packages that may be used instead of or in combination with the tools in this protocol. Many alternative read-alignment programs[19–21] now exist, and there are several alternative tools for transcriptome reconstruction[22,23], quantification[10,24,25] and differential expression[26–28] analysis. Because many of these tools operate on similarly formatted data files, they could be used instead of or in addition to the tools used here. For example, with straightforward postprocessing scripts, one could provide GSNAP[19] read alignments to Cufflinks, or use a Scripture[22] transcriptome reconstruction instead of a Cufflinks one before differential expression analysis. However, such customization is beyond the scope of this protocol, and we discourage novice RNA-seq users from making changes to the protocol outlined here.

This protocol is appropriate for RNA-seq experiments on organisms with sequenced reference genomes. Users working without a sequenced genome but who are interested in gene discovery should consider performing *de novo* transcriptome assembly using one of several tools such as Trinity[29], Trans-Abyss[30] or Oases (http://www.ebi.ac.uk/~zerbino/oases/). Users performing expression analysis with a *de novo* transcriptome assembly may wish to consider RSEM[10] or IsoEM[25]. For a survey of these tools (including TopHat and Cufflinks) readers may wish to see the study by Garber *et al.*[12], which describes their comparative advantages and disadvantages and the theoretical considerations that inform their design.

Overview of the protocol

Although RNA-seq experiments can serve many purposes, we describe a workflow that aims to compare the transcriptome profiles of two or more biological conditions, such as a wild-type versus mutant or control versus knockdown experiments. For simplicity, we assume that the experiment compares only two biological conditions, although the software is designed to support many more, including time-course experiments.

This protocol begins with raw RNA-seq reads and concludes with publication-ready visualization of the analysis. **Figure 2** highlights the main steps of the protocol. First, reads for each condition are mapped to the reference genome with TopHat. Many RNA-seq users are also interested in gene or splice variant discovery, and the failure to look for new transcripts can bias expression estimates and reduce accuracy[8]. Thus, we include transcript assembly with Cufflinks as a step in the workflow (see **Box 1** for a workflow that skips gene and transcript discovery). After running TopHat, the resulting alignment files are provided to Cufflinks to generate a transcriptome assembly for each condition. These assemblies are then merged together using the Cuffmerge utility, which is included with the Cufflinks package. This merged assembly provides a uniform basis for calculating gene and transcript expression in each condition. The reads and the merged assembly are fed to Cuffdiff, which calculates expression levels and tests the statistical significance of observed changes. Cuffdiff also performs an additional layer of differential analysis. By grouping transcripts into biologically meaningful groups (such as transcripts that share the same transcription start site (TSS)), Cuffdiff identifies genes that are differentially regulated at the transcriptional or post-transcriptional level. These results are reported as a set of text files and can be displayed in the plotting environment of your choice.

We have recently developed a powerful plotting tool called CummeRbund (http://compbio.mit.edu/cummeRbund/), which provides functions for creating commonly used expression plots such as volcano, scatter and box plots. CummeRbund also handles the details of parsing Cufflinks output file formats to connect Cufflinks and the R statistical computing environment. CummeRbund transforms Cufflinks output files into R objects suitable for analysis with a wide variety of other packages available within the R environment and can also now be accessed through the Bioconductor website (http://www.bioconductor.org/).

This protocol does not require extensive bioinformatics expertise (e.g., the ability to write complex scripts), but it does assume familiarity with the UNIX command-line interface. Users should

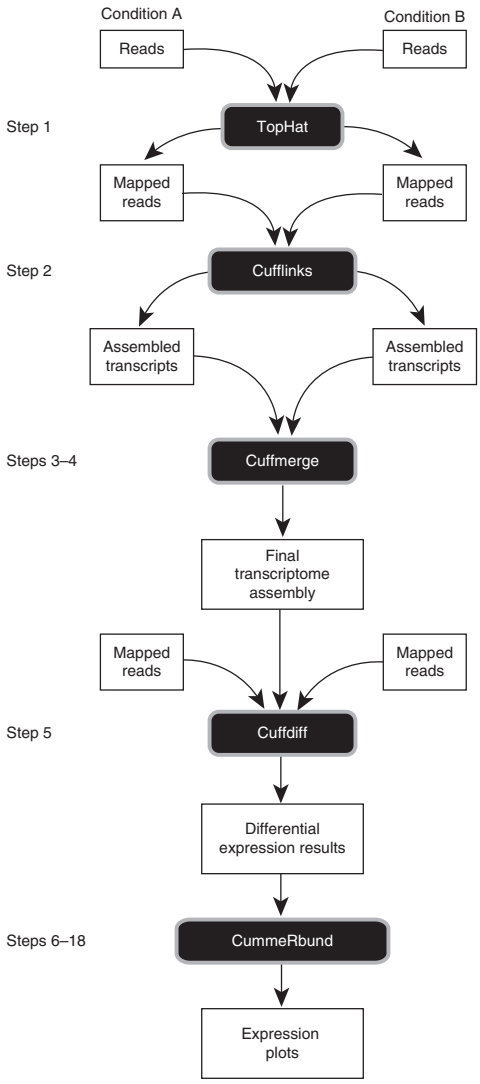


Figure 2 | An overview of the Tuxedo protocol. In an experiment involving two conditions, reads are first mapped to the genome with TopHat. The reads for each biological replicate are mapped independently. These mapped reads are provided as input to Cufflinks, which produces one file of assembled transfrags for each replicate. The assembly files are merged with the reference transcriptome annotation into a unified annotation for further analysis. This merged annotation is quantified in each condition by Cuffdiff, which produces expression data in a set of tabular files. These files are indexed and visualized with CummeRbund to facilitate exploration of genes identified by Cuffdiff as differentially expressed, spliced, or transcriptionally regulated genes. FPKM, fragments per kilobase of transcript per million fragments mapped.

feel comfortable creating directories, moving files between them and editing text files in a UNIX environment. Installation of the tools may require additional expertise and permission from one's computing system administrators.

Read alignment with TopHat

Alignment of sequencing reads to a reference genome is a core step in the analysis workflows for many high-throughput sequencing assays, including ChIP-Seq[31], RNA-seq, ribosome profiling[32] and others. Sequence alignment itself is a classic problem in computer science and appears frequently in bioinformatics. Hence, it is perhaps not surprising that many read alignment programs have been developed within the last few years. One of the most popular and to date most efficient is Bowtie[33] (http://bowtie-bio.sourceforge.net/index.shtml), which uses an extremely economical data structure called the FM index[34] to store the reference genome sequence and allows it to be searched rapidly. Bowtie uses the FM index to align reads at a rate of tens of millions per CPU hour. However, Bowtie is not suitable for all sequence alignment tasks. It does not allow alignments between a read and the genome to contain large gaps; hence, it cannot align reads that span introns. TopHat was created to address this limitation.

TopHat uses Bowtie as an alignment 'engine' and breaks up reads that Bowtie cannot align on its own into smaller pieces called segments. Often, these pieces, when processed independently, will align to the genome. When several of a read's segments align to the genome far apart (e.g., between 100 bp and several hundred kilobases) from one another, TopHat infers that the read spans a splice junction and estimates where that junction's splice sites are. By processing each 'initially unmappable' read, TopHat can build up an index of splice sites in the transcriptome on the fly without a priori gene or splice site annotations. This capability is crucial, because, as numerous RNA-seq studies have now shown, our catalogs of alternative splicing events remain woefully incomplete. Even in the transcriptomes of often-studied model organisms, new splicing events are discovered with each additional RNA-seq study.

Aligned reads say much about the sample being sequenced. Mismatches, insertions and deletions in the alignments can identify polymorphisms between the sequenced sample and the reference genome, or even pinpoint gene fusion events in tumor samples. Reads that align outside annotated genes are often strong evidence of new protein-coding genes and noncoding RNAs. As mentioned above, RNA-seq read alignments can reveal new alternative splicing events and isoforms. Alignments can also be used to accurately quantify gene and transcript expression, because the number of reads produced by a transcript is proportional to its abundance (**Box 2**). Discussion of polymorphism and fusion detection is out of the scope of this protocol, and we address transcript assembly and gene discovery only as they relate to differential expression analysis. For a further review of these topics, see Garber *et al.*[12].

Transcript assembly with Cufflinks

Accurately quantifying the expression level of a gene from RNA-seq reads requires accurately identifying which isoform of a given gene produced each read. This, of course, depends on knowing all of the splice variants (isoforms) of that gene. Attempting to quantify gene and transcript expression by using an incomplete or incorrect transcriptome annotation leads to inaccurate expression values[8]. Cufflinks assembles individual transcripts from RNA-seq reads that have been aligned to the genome. Because a sample may contain reads from multiple splice variants for a given gene, Cufflinks must be able to infer the splicing structure of each gene. However, genes sometimes have multiple alternative splicing events, and there may be many possible reconstructions of the gene model that explain the sequencing data. In fact, it is often not obvious how many splice variants of the gene may be present. Thus, Cufflinks reports a parsimonious transcriptome assembly of the data. The algorithm reports as few full-length transcript fragments or 'transfrags' as are needed to 'explain' all the splicing event outcomes in the input data.

Box 1 | Alternate protocols

A. Strand-specific RNA-seq

1. At Step 1, supply the option '`--library-type`' to TopHat to enable strand-specific processing of the reads. TopHat will map the reads for each sample to the reference genome and will attach meta-data to each alignment that Cufflinks and Cuffdiff can use for more accurate assembly and quantification. The `--library-type` option requires an argument that specifies which strand-specific protocol was used to generate the reads. See **Table 1** for help in choosing a library type.

```
$ tophat -p 8 -G genes.gtf -o C1_R1_thout --library-type=fr-firststrand \
    genome C1_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R2_thout --library-type=fr-firststrand \
    genome C1_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R3_thout --library-type=fr-firststrand \
    genome C1_R3_1.fq C1_R3_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R1_thout --library-type=fr-firststrand \
    genome C2_R1_1.fq C2_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R2_thout --library-type=fr-firststrand \
    genome C2_R2_1.fq C2_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R3_thout --library-type=fr-firststrand \
    genome C2_R3_1.fq C1_R3_2.fq
```

B. Quantification of reference annotation only (no gene/transcript discovery)

1. At Step 1, supply the option '`--no-novel-juncs`' to TopHat to map the reads for each sample to the reference genome, with novel splice discovery disabled:

```
$ tophat -p 8 -G genes.gtf -o C1_R1_thout --no-novel-juncs genome C1_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R2_thout --no-novel-juncs genome C1_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R3_thout --no-novel-juncs genome C1_R3_1.fq C1_R3_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R1_thout --no-novel-juncs genome C2_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R2_thout --no-novel-juncs genome C2_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R3_thout --no-novel-juncs genome C2_R3_1.fq C1_R3_2.fq
```

2. Skip PROCEDURE Steps 2–4.
3. Run Cuffdiff using the reference transcriptome along with the BAM files from TopHat for each replicate:

```
$ cuffdiff -o diff_out -b genome.fa -p 8 -u genes.gtf \
./C1_R1_thout/accepted_hits.bam,./C1_R2_thout/accepted_hits.bam,./C1_R3_thout/accepted_hits.
bam \
./C2_R1_thout/accepted_hits.bam,./C2_R3_thout/accepted_hits.bam,./C2_R2_thout/accepted_hits.
bam
```

C. Quantification without a reference annotation

1. Map the reads for each sample to the reference genome:

```
$ tophat -p 8 -o C1_R1_thout genome C1_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -o C1_R2_thout genome C1_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -o C1_R3_thout genome C1_R3_1.fq C1_R3_2.fq
$ tophat -p 8 -o C2_R1_thout genome C2_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -o C2_R2_thout genome C2_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -o C2_R3_thout genome C2_R3_1.fq C1_R3_2.fq
```

2. Perform PROCEDURE Steps 2 and 3.
3. Run Cuffmerge on all your assemblies to create a single merged transcriptome annotation:

```
cuffmerge -s genome.fa -p 8 assemblies.txt
```

D. Analysis of single-ended sequencing experiments

1. At Step 1, simply supply the single FASTQ file for each replicate to TopHat to map the reads for each sample to the reference genome:

```
$ tophat -p 8 -G genes.gtf -o C1_R1_thout genome C1_R1.fq
$ tophat -p 8 -G genes.gtf -o C1_R2_thout genome C1_R2.fq
$ tophat -p 8 -G genes.gtf -o C1_R3_thout genome C1_R3.fq
$ tophat -p 8 -G genes.gtf -o C2_R1_thout genome C2_R1.fq
$ tophat -p 8 -G genes.gtf -o C2_R2_thout genome C2_R2.fq
$ tophat -p 8 -G genes.gtf -o C2_R3_thout genome C2_R3.fq
```

2. Perform PROCEDURE Steps 2–18.

Box 2 | Calculating expression levels from read counts

The number of RNA-seq reads generated from a transcript is directly proportional to that transcript's relative abundance in the sample. However, because cDNA fragments are generally size-selected as part of library construction (to optimize output from the sequencer), longer transcripts produce more sequencing fragments than shorter transcripts. For example, suppose a sample has two transcripts, A and B, both of which are present at the same abundance. If B is twice as long as A, an RNA-seq library will contain (on average) twice as many reads from B as from A. To calculate the correct expression level of each transcript, Cufflinks must count the reads that map to each transcript and then normalize this count by each transcript's length. Similarly, two sequencing runs of the same library may produce different volumes of sequencing reads. To compare the expression level of a transcript across runs, the counts must be normalized for the total yield of the machine. The commonly used fragments per kilobase of transcript per million mapped fragments (or FPKM[8], also known as RPKM[1] in single-ended sequencing experiments) incorporates these two normalization steps to ensure that expression levels for different genes and transcripts can be compared across runs.

When a gene is alternatively spliced and produces multiple isoforms in the same sample, many of the reads that map to it will map to constitutive or shared exons, complicating the process of counting reads for each transcript. A read from a shared exon could have come from one of several isoforms. To accurately compute each transcript's expression level, a simple counting procedure will not suffice; more sophisticated statistical inference is required. Cufflinks and Cuffdiff implement a linear statistical model to estimate an assignment of abundance to each transcript that explains the observed reads with maximum likelihood.

Because Cufflinks and Cuffdiff calculate the expression level of each alternative splice transcript of a gene, calculating the expression level of a gene is simple—the software simply adds up the expression level of each splice variant. This is possible because FPKM is directly proportional to abundance. In fact, the expression level of any group of transcripts (e.g., a group of transcripts that share the same promoter) can be safely computed by adding the expression levels of the members of that group.

After the assembly phase, Cufflinks quantifies the expression level of each transfrag in the sample. This calculation is made using a rigorous statistical model of RNA-seq and is used to filter out background or artifactual transfrags[8]. For example, with current library preparation protocols, most genes generate a small fraction of reads from immature primary transcripts that are generally not interesting to most users. As these transfrags are typically far less abundant in the library than the mature, spliced transcripts, Cufflinks can use its abundance estimates to automatically exclude them. Given a sample, Cufflinks can also quantify transcript abundances by using a reference annotation rather than assembling the reads. However, for multiple samples, we recommend that the user quantify genes and transcripts using Cuffdiff, as described below.

When you are working with several RNA-seq samples, it becomes necessary to pool the data and assemble it into a comprehensive set of transcripts before proceeding to differential analysis. A natural approach to this problem would be to simply pool aligned reads from all samples and run Cufflinks once on this combined set of alignments. However, we do not usually recommend this tactic for two reasons. First, because assembly becomes more computationally expensive as read depth increases, assembling the pooled alignments may not be feasible with the machines available in your laboratory. Second, with a pooled set of reads, Cufflinks will be faced with a more complex mixture of splice isoforms for many genes than would be seen when assembling the samples individually, and this increases the probability that it will assemble the transcripts incorrectly (associating the wrong outcomes of different splicing events in some transcripts). A better strategy is to assemble the samples individually and then merge the resulting assemblies together. We have recently developed a utility program, Cuffmerge, which handles this task using many of the same concepts and algorithms as Cufflinks does when assembling transcripts from individual reads.

Box 3 | File formats and data storage

Storing RNA-seq data and analysis results in standardized, well-documented file formats is crucial for data sharing between laboratories and for reuse or reproduction of past experimental data. The next-generation sequencing informatics community has worked hard to adopt open file standards. Although some of these formats are still evolving, data storage conventions have matured substantially. Raw, unmapped sequencing reads may be one of several formats specific to the vendor or instrument, but the most commonly encountered format is FASTQ, a version of FASTA that has been extended with Phred base quality scores. TopHat accepts FASTQ and FASTA files of sequencing reads as input. Alignments are reported in BAM files. BAM is the compressed, binary version of SAM[43], a flexible and general purpose read alignment format. SAM and BAM files are produced by most next-generation sequence alignment tools as output, and many downstream analysis tools accept SAM and BAM as input. There are also numerous utilities for viewing and manipulating SAM and BAM files. Perhaps most popular among these are the SAM tools (http://samtools.sourceforge.net/) and the Picard tools (http://picard.sourceforge.net/). Both Cufflinks and Cuffdiff accept SAM and BAM files as input. Although FASTQ, SAM and BAM files are all compact, efficient formats, typical experiments can still generate very large files. It is not uncommon for a single lane of Illumina HiSeq sequencing to produce FASTQ and BAM files with a combined size of 20 GB or larger. Laboratories planning to perform more than a small number of RNA-seq experiments should consider investing in robust storage infrastructure, either by purchasing their own hardware or through cloud storage services[44].

TABLE 1 | Library type options for TopHat and Cufflinks.

Library type	RNA-seq protocol	Description
fr-unstranded (default)	Illumina TruSeq	Reads from the leftmost end of the fragment (in transcript coordinates) map to the transcript strand, and the rightmost end maps to the opposite strand
fr-firststrand	dUTP, NSR, NNSR[39]	Same as above except we enforce the rule that the rightmost end of the fragment (in transcript coordinates) is the first sequenced (or only sequenced for single-end reads). Equivalently, it is assumed that only the strand generated during first strand synthesis is sequenced
fr-secondstrand	Directional Illumina (Ligation), Standard SOLiD	Same as above except TopHat/Cufflinks enforce the rule that the leftmost end of the fragment (in transcript coordinates) is the first sequenced (or only sequenced for single-end reads). Equivalently, it is assumed that only the strand generated during second strand synthesis is sequenced

Cuffmerge is essentially a 'meta-assembler'—it treats the assembled transfrags the way Cufflinks treats reads, merging them together parsimoniously. Furthermore, when a reference genome annotation is available, Cuffmerge can integrate reference transcripts into the merged assembly. It performs a reference annotation-based transcript (RABT) assembly[35] to merge reference transcripts with sample transfrags and produces a single annotation file for use in downstream differential analysis. **Figure 3** shows an example of the benefits of merging sample assemblies with Cuffmerge.

Once each sample has been assembled and all samples have been merged, the final assembly can be screened for genes and transcripts that are differentially expressed or regulated between samples. This protocol recommends that you assemble your samples with Cufflinks before performing differential expression to improve accuracy, but this step is optional. Assembly can be computationally demanding, and interpreting assemblies is often difficult, especially when sequencing depth is low, because distinguishing full-length isoforms from partially reconstructed fragments is not always possible without further experimental evidence. Furthermore, although Cufflinks assemblies are quite accurate when they are provided with sufficiently high-quality data, assembly errors do occur and can accumulate when merging many assemblies. When you are working with RNA-seq data from well-annotated organisms such as human, mouse or fruit fly, you may wish to run the alternate protocol 'Quantification of reference annotation only' (**Box 1**; see also **Table 1**).

Even for well-studied organisms, most RNA-seq experiments should reveal new genes and transcripts. A recent analysis of deep RNA-seq samples from 24 human tissues and cell lines revealed over 8,000 new long, noncoding RNAs along with numerous potential protein-coding genes[6]. Many users of RNA-seq are interested in discovering new genes and transcripts in addition to performing differential analysis. However, it can be difficult to distinguish full-length novel transcripts from partial fragments using RNA-seq data alone. Gaps in sequencing coverage will cause breaks in transcript reconstructions, just as they do during genome assembly. High-quality reconstructions of eukaryotic transcriptomes will contain thousands of full-length transcripts. Low-quality reconstructions, especially those produced from shallow sequencing runs (e.g., fewer than 10 million reads), may contain tens or even hundreds of thousands of partial transcript fragments. Cufflinks includes a utility program called 'Cuffcompare' that can compare Cufflinks assemblies to reference annotation files and help sort out new genes from known ones. Because of the difficulty in constructing transcriptome assemblies, we encourage users to validate novel genes and transcripts by traditional cloning and PCR-based techniques. We also encourage validation of transcript ends by rapid amplification of cDNA ends (RACE) to rule out incomplete reconstruction due to gaps in sequencing coverage. Although a complete discussion of transcript and gene discovery is beyond the scope of this protocol, readers interested in such analysis should consult the Cufflinks manual to help identify new transcripts[6].

Differential analysis with Cuffdiff

Cufflinks includes a separate program, Cuffdiff, which calculates expression in two or more samples and tests the statistical significance of each observed change in expression between them. The statistical model used to evaluate changes assumes that the number of reads produced by each transcript is proportional to its abundance but fluctuates because of technical variability during library preparation and sequencing and because of biological variability between replicates of the same experiment. Despite its exceptional overall accuracy, RNA-seq, like all other assays for gene expression, has sources of bias. These biases have been shown to depend greatly on library preparation protocol[36–39]. Cufflinks and Cuffdiff

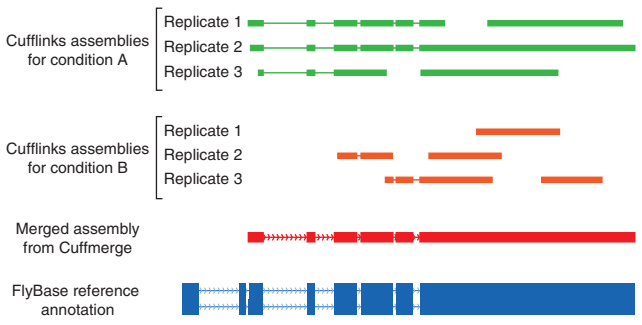


Figure 3 | Merging sample assemblies with a reference transcriptome annotation. Genes with low expression may receive insufficient sequencing depth to permit full reconstruction in each replicate. However, merging the replicate assemblies with Cuffmerge often recovers the complete gene. Newly discovered isoforms are also integrated with known ones at this stage into more complete gene models.

can automatically model and subtract a large fraction of the bias in RNA-seq read distribution across each transcript, thereby improving abundance estimates[38].

Although RNA-seq is often noted to have substantially less technical variability than other gene expression assays (e.g., microarrays), biological variability will persist[40]. Cuffdiff allows you to supply multiple technical or biological replicate sequencing libraries per condition. With multiple replicates, Cuffdiff learns how read counts vary for each gene across the replicates and uses these variance estimates to calculate the significance of observed changes in expression. We strongly recommend that RNA-seq experiments be designed in replicate to control for batch effects such as variation in culture conditions. Advances in multiplexing techniques during sequencing now make it possible to divide sequencing output among replicates without increasing total sequencing depth (and thus cost of sequencing).

Cuffdiff reports numerous output files containing the results of its differential analysis of the samples. Gene and transcript expression level changes are reported in simple tabular output files that can be viewed with any spreadsheet application (such as Microsoft Excel). These files contain familiar statistics such as fold change (in $\log_2$ scale), *P* values (both raw and corrected for multiple testing) and gene- and transcript-related attributes such as common name and location in the genome.

Cuffdiff also reports additional differential analysis results beyond simple changes in gene expression. The program can identify genes that are differentially spliced or differentially regulated via promoter switching. The software groups together isoforms of a gene that have the same TSS. These TSS groups represent isoforms that are all derived from the same pre-mRNA; accordingly, changes in abundance relative to one another reflect differential splicing of their common pre-mRNA. Cuffdiff also calculates the total expression level of a TSS group by adding up the expression levels of the isoforms within it. When a gene has multiple TSSs, Cuffdiff looks for changes in relative abundance between them, which reflect changes in TSS (and thus promoter) preference between conditions. The statistics used to evaluate significance of changes within and between TSS groupings are somewhat different from those used to assess simple expression level changes of a given transcript or gene. Readers interested in further statistical detail should see the supplemental material of Trapnell *et al.*[8]. **Figure 4** illustrates how Cuffdiff constructs TSS groupings and uses them to infer differential gene regulation.

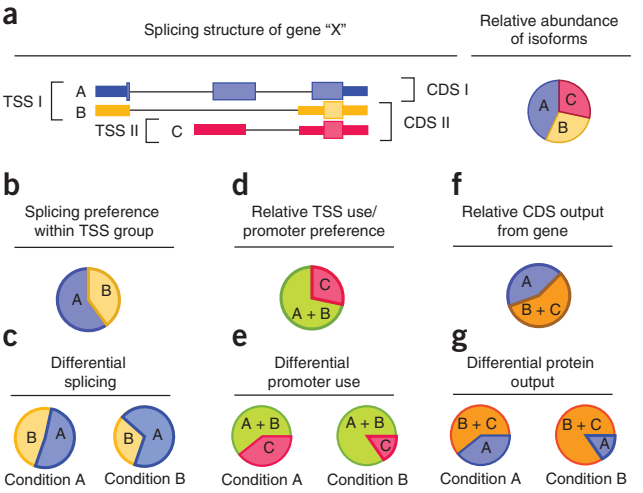


Figure 4 | Analyzing groups of transcripts identifies differentially regulated genes. (**a**) Genes may produce multiple splice variants (labeled A–C) at different abundances through alternative transcription start sites (TSS), alternative cleavage and polyadenylation of 3′ ends, or by alternative splicing of primary transcripts. (**b**) Grouping isoforms by TSS and looking for changes in relative abundance between and within these groups yield mechanistic clues into how genes are differentially regulated. (**c**) For example, in the above hypothetical gene, changes in the relative abundance between isoforms A and B within TSS I group across conditions may be attributable to differential splicing of the primary transcript from which they are both produced. (**d**) Adding their expression levels yields a proxy expression value for this primary transcript. (**e**) Changes in this level relative to the gene's other primary transcript (i.e., isoform C) indicate possible differential promoter preference across conditions. (**f**,**g**) Similarly, genes with multiple annotated coding sequences (CDS) (**f**) can be analyzed for differential output of protein-coding sequences (**g**).

Visualization with CummeRbund

Cuffdiff provides analyses of differential expression and regulation at the gene and transcript level. These results are reported in a set of tab-delimited text files that can be opened with spreadsheet and charting programs such as Microsoft Excel. The Cuffdiff file formats are designed to simplify use by other downstream programs. However, browsing these files by eye is not especially easy, and working with data across multiple files can be quite difficult. For example, extracting the list of differentially expressed genes is fairly straightforward, but plotting the expression levels for each isoform of those genes requires a nontrivial script.

We have recently created a user-friendly tool, called CummeRbund, to help manage, visualize and integrate all of the data produced by a Cuffdiff analysis. CummeRbund drastically simplifies common data exploration tasks, such as plotting and cluster analysis of expression data (**Fig. 5**). Furthermore, you can create publication-ready plots with a single command. Scripted plotting also lets you automate plot generation, allowing you to reuse analyses from previous experiments. Finally, CummeRbund handles the transformation of Cuffdiff data into the R statistical computing environment, making RNA-seq expression analysis with Cuffdiff more compatible with many other advanced statistical analysis and plotting packages.

This protocol concludes with a brief exploration of the example data set using CummeRbund, but the plots illustrated here are only the beginning of what is possible with this tool. Furthermore, CummeRbund is new and under active development—future versions will contain powerful new views of RNA-seq data. Users familiar with ggplot[41], the popular plotting packaging around which CummeRbund is designed, may wish to design their own plots and analysis functions. We strongly encourage such users to contribute their plotting scripts to the open-source CummeRbund project.

Processing time and memory requirements

RNA-seq analysis is generally more computationally demanding than many other bioinformatics tasks. Analyzing large data sets requires a powerful workstation or server with ample disk space (see **Box 3**) and with at least 16 GB of RAM. Bowtie, TopHat and the Cufflinks tools are all designed to take advantage of multicore processors, and running the programs with multiple threads is

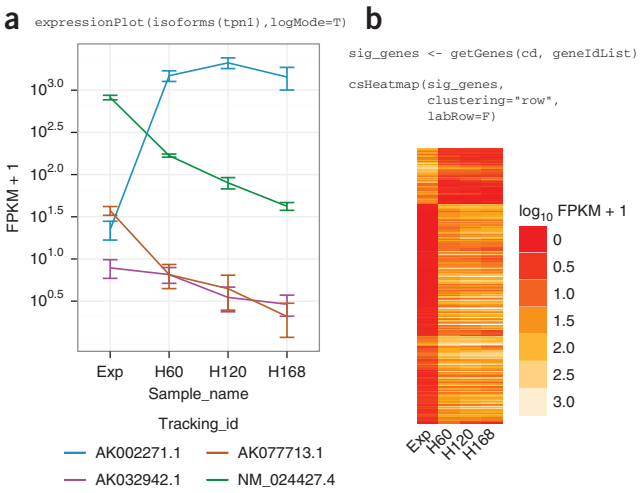


Figure 5 | CummeRbund helps users rapidly explore their expression data and create publication-ready plots of differentially expressed and regulated genes. With just a few lines of plotting code, CummeRbund can visualize differential expression at the isoform level, as well as broad patterns among large sets of genes. (**a**) A myoblast differentiation time-course experiment reveals the emergence of a skeletal muscle-specific isoform of tropomyosin I. (**b**) This same time-course data capture the dynamics of hundreds of other genes in the mouse transcriptome during muscle development[8]. FPKM, fragments per kilobase of transcript per million fragments mapped.

highly recommended. Of the tasks in this protocol, read mapping with TopHat is usually the least demanding task in terms of memory, but mapping a full lane of HiSeq 100 bp paired-end reads can still take a day or two on a typical workstation or compute cluster node. If possible, you should align the reads from each sample on a separate machine to parallelize the total alignment workload. Assembling transcripts can also be very demanding in terms of both processing time and memory. You may want to consider using the `--mask/-M` option during your Cufflinks runs to exclude genes that are extremely abundant in your samples (e.g., actin), because Cufflinks may spend a long time assembling these genes. When a reference transcriptome annotation is available, Cuffmerge will add these genes back into the final transcriptome file used during differential analysis. Thus, Cuffdiff will still quantify expression for these genes—excluding them during sample assembly simply amounts to forgoing discovery of novel splice variants.

RNA-seq experimental design

RNA-seq has been hailed as a whole-transcriptome expression assay of unprecedented sensitivity, but no amount of technical consistency or sensitivity can eliminate biological variability[40]. We strongly recommend that experimenters designing an RNA-seq study heed lessons learned from microarray analysis. In particular, biological replication of each condition is crucial. How deeply each condition must be replicated is an open research question, and more replicates are almost always preferable to fewer. Multiplexed RNA-seq is making replication possible without increasing total sequencing costs by reducing the total sequencing depth in each replicate and making experimental designs more robust. With currently available kits, sequencing each condition in triplicate is quite feasible. Thus, the protocol here is illustrated through an example experiment with three replicates of each condition.

When considering an RNA-seq experiment, two other design choices have a major effect on accuracy. Library fragments may be sequenced from one or both ends, and although paired-end reads are up to two times the cost of single-end reads, we and others[24] strongly recommend paired-end sequencing whenever possible. The marginal information provided by paired-end sequencing runs over single-end runs at the same depth is considerable. Cufflinks' algorithms for transcript assembly and expression quantitation are much more accurate with paired-end reads. Sequencing read length is also a major consideration, and longer reads are generally preferable to short ones. TopHat is more accurate when discovering splice junctions with longer reads, and reads of 75 bp and longer are substantially more powerful than shorter reads. However, as generating longer reads can add substantially to the cost of an RNA-seq experiment, many experimenters may wish to sequence more samples (or more replicates of the same samples) with shorter reads.

MATERIALS

EQUIPMENT

- Data (requirements vary according to your experimental goals; see EQUIPMENT SETUP)
- Bowtie software (http://bowtie-bio.sourceforge.net/index.shtml/)
- SAM tools (http://samtools.sourceforge.net/)
- TopHat software (http://tophat.cbcb.umd.edu/)
- Cufflinks software (http://cufflinks.cbcb.umd.edu/)
- CummeRbund software (http://compbio.mit.edu/cummeRbund/)
- Fruit fly iGenome packages (Ensembl build; download via the TopHat and Cufflinks websites, along with packages for many other organisms; see EQUIPMENT SETUP)
- Hardware (64-bit computer running either Linux or Mac OS X (10.4 Tiger or later); 4 GB of RAM (16 GB preferred); see EQUIPMENT SETUP)

EQUIPMENT SETUP

▲ **CRITICAL** Most of the commands given in the protocol are runnable at the UNIX shell prompt, and all such commands are meant to be run from the example working directory. The protocol also includes small sections of code runnable in the R statistical computing environment. Commands meant to be executed from the UNIX shell (e.g., bash or csh) are prefixed with a '$' character. Commands meant to be run from either an R script or at the R interactive shell are prefixed with a '>' character.

Required data This protocol is illustrated through an example experiment in *Drosophila melanogaster* that you can analyze to familiarize yourself with the Tuxedo tools. We recommend that you create a single directory (e.g., '`my_rnaseq_exp`') in which to store all example data and generated analysis files. All protocol steps are given assuming you are working from within this directory at the UNIX shell prompt.

To use TopHat and Cuffdiff for differential gene expression, you must be working with an organism with a sequenced genome. Both programs can also make use of an annotation file of genes and transcripts, although this is optional. TopHat maps reads to the genome using Bowtie (see EQUIPMENT), which requires a set of genomic index files. Indexes for many organisms can be downloaded from the Bowtie website.

If this is your first time running the protocol, download the fruit fly iGenome (see EQUIPMENT) to your working directory. Later, you may wish

to move the package's files along with the iGenomes for other organisms to a common location on your file system. The packages are 'read-only' and do not need to be redownloaded with each run of the protocol. They are resources that are reused each time the protocol is run.

Hardware setup The software used in this protocol is intended for operation on a 64-bit machine, running a 64-bit version of the operating system. This may exclude some Linux users running 32-bit kernels, but the tools used in the protocol can be compiled for 32-bit machines. See the Bowtie, TopHat, sequence alignment/map (SAM) tools and Cufflinks websites for more details. To process RNA-seq experiments, the machine used for the analysis will need at least 4 GB of RAM. We recommend a machine with at least 16 GB for analysis of deep sequencing data sets such as those produced by Illumina's HiSeq 2000 sequencer.

Downloading and organizing required data Unpack the fruit fly iGenome and inspect the contents. Assuming we stored the package at *my_rnaseq_exp/*, the package expands to contain a folder *Drosophila_melanogaster/Ensembl/BDGP5.25/*, which has the following structure: *Annotation/GenomeStudio/Sequence/* (i.e., three separate directories).

The Annotation directory contains another directory called 'Genes', which contains a file called 'genes.gtf'. For the time being, create a link to this file in your example working directory (to simplify the commands needed during the protocol). From your working directory, type:

```
$ ln -s ./Drosophila_melanogaster/Ensembl/BDGP5.25/Annotation/Genes/genes.gtf .
```

Similarly, create links to the Bowtie index included with the iGenome package:

```
$ ln -s ./Drosophila_melanogaster/Ensembl/BDGP5.25/Sequence/BowtieIndex/genome.*.
```

Downloading sequencing data In addition to the fruit fly iGenome package, to run the protocol through the examples given here you will need to download the sequencing data. Raw sequencing reads, aligned reads, assembled transfrags and differential analysis are all available through the Gene Expression Omnibus at accession GSE32038. Download these files and store them in a directory separate from your working directory so that you can compare them later with the files generated while running the protocol. Store the sequencing read files (those with extension '.fq') in your example working directory.

Downloading and installing software Create a directory to store all of the executable programs used in this protocol (if none already exists):

```
$ mkdir $HOME/bin
```

Add the above directory to your PATH environment variable:

```
$ export PATH=$HOME/bin:$PATH
```

To install the SAM tools, download the SAM tools (http://samtools.sourceforge.net/) and unpack the SAM tools tarball and cd to the SAM tools source directory:

```
$ tar jxvf samtools-0.1.17.tar.bz2
$ cd samtools-0.1.17
```

Copy the `samtools` binary to some directory in your PATH:

```
$ cp samtools $HOME/bin
```

To install Bowtie, download the latest binary package for Bowtie (http://bowtie-bio.sourceforge.net/index.shtml) and unpack the Bowtie zip archive and cd to the unpacked directory:

```
$ unzip bowtie-0.12.7-macos-10.5-x86_64.zip
$ cd bowtie-0.12.7
```

Copy the Bowtie executables to a directory in your PATH:

```
$ cp bowtie $HOME/bin
$ cp bowtie-build $HOME/bin
$ cp bowtie-inspect $HOME/bin
```

To install TopHat, download the binary package for version 1.3.2 of TopHat (http://tophat.cbcb.umd.edu/) and unpack the TopHat tarball and cd to the unpacked directory:

```
$ tar zxvf tophat-1.3.2.OSX_x86_64.tar.gz
$ cd tophat-1.3.2.OSX_x86_64
```

Copy the TopHat package executable files to some directory in your PATH:

```
cp * $HOME/bin
```

To install Cufflinks, download the binary package of version 1.2.1 for Cufflinks (http://cufflinks.cbcb.umd.edu/) and unpack the Cufflinks tarball and cd to the unpacked directory:

```
$ tar zxvf cufflinks-1.2.1.OSX_x86_64.tar.gz
$ cd cufflinks-1.2.1.OSX_x86_64
```

Copy the Cufflinks package executuble files to some directory in your PATH:

```
$ cp * $HOME/bin
```

To Install CummeRbund, start an R session:

```
$ R
R version 2.13.0 (2011-04-13)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-apple-darwin10.6.0/x86_64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
```

Install the CummeRbund package:

```
> source('http://www.bioconductor.org/biocLite.R')
> biocLite('cummeRbund')
```

PROCEDURE

Align the RNA-seq reads to the genome ● TIMING ~6 h

1| Map the reads for each sample to the reference genome:

```
$ tophat -p 8 -G genes.gtf -o C1_R1_thout genome C1_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R2_thout genome C1_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C1_R3_thout genome C1_R3_1.fq C1_R3_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R1_thout genome C2_R1_1.fq C1_R1_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R2_thout genome C2_R2_1.fq C1_R2_2.fq
$ tophat -p 8 -G genes.gtf -o C2_R3_thout genome C2_R3_1.fq C1_R3_2.fq
```

? TROUBLESHOOTING

Assemble expressed genes and transcripts ● TIMING ~6 h

2| Assemble transcripts for each sample:

```
$ cufflinks -p 8 -o C1_R1_clout C1_R1_thout/accepted_hits.bam
$ cufflinks -p 8 -o C1_R2_clout C1_R2_thout/accepted_hits.bam
$ cufflinks -p 8 -o C1_R3_clout C1_R3_thout/accepted_hits.bam
$ cufflinks -p 8 -o C2_R1_clout C2_R1_thout/accepted_hits.bam
$ cufflinks -p 8 -o C2_R2_clout C2_R2_thout/accepted_hits.bam
$ cufflinks -p 8 -o C2_R3_clout C2_R3_thout/accepted_hits.bam
```

? TROUBLESHOOTING

3| Create a file called assemblies.txt that lists the assembly file for each sample. The file should contain the following lines:

```
./C1_R1_clout/transcripts.gtf
./C2_R2_clout/transcripts.gtf
./C1_R2_clout/transcripts.gtf
./C2_R1_clout/transcripts.gtf
./C1_R3_clout/transcripts.gtf
./C2_R3_clout/transcripts.gtf
```

4| Run Cuffmerge on all your assemblies to create a single merged transcriptome annotation:

```
cuffmerge -g genes.gtf -s genome.fa -p 8 assemblies.txt
```

Identify differentially expressed genes and transcripts ● TIMING ~6 h

5| Run Cuffdiff by using the merged transcriptome assembly along with the BAM files from TopHat for each replicate:

```
$ cuffdiff -o diff_out -b genome.fa -p 8 -L C1,C2 -u merged_asm/merged.gtf \
./C1_R1_thout/accepted_hits.bam,./C1_R2_thout/accepted_hits.bam,./C1_R3_thout/
accepted_hits.bam \
./C2_R1_thout/accepted_hits.bam,./C2_R3_thout/accepted_hits.bam,./C2_R2_thout/
accepted_hits.bam
```

? TROUBLESHOOTING

Explore differential analysis results with CummeRbund ● TIMING variable

6| Open a new plotting script file in the editor of your choice, or use the R interactive shell:

```
$ R
R version 2.13.0 (2011-04-13)
Copyright (C) 2011 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-apple-darwin10.6.0/x86_64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
```

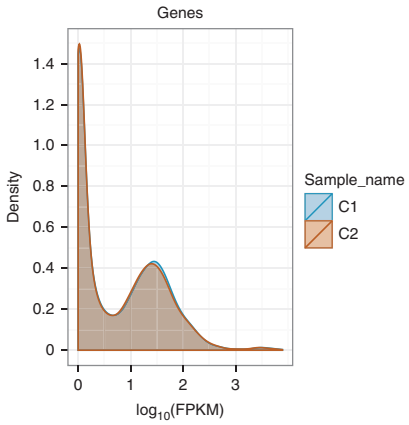


Figure 6 | CummeRbund plots of the expression level distribution for all genes in simulated experimental conditions C1 and C2. FPKM, fragments per kilobase of transcript per million fragments mapped.

```
Type 'demo()' for some demos, 'help()'
for on-line help, or
'help.start()' for an HTML browser
interface to help.
Type 'q()' to quit R.
```

7| Load the CummeRbund package into the R environment:

```
> library(cummeRbund)
```

8| Create a CummeRbund database from the Cuffdiff output:

```
> cuff_data <- readCufflinks('diff_out')
```

9| Plot the distribution of expression levels for each sample (**Fig. 6**):

```
> csDensity(genes(cuff_data))
```

10| Compare the expression of each gene in two conditions with a scatter plot (**Fig. 7**):

```
> csScatter(genes(cuff_data), 'C1', 'C2')
```

11| Create a volcano plot to inspect differentially expressed genes (**Fig. 8**):

```
> csVolcano(genes(cuff_data), 'C1', 'C2')
```

12| Plot expression levels for genes of interest with bar plots (**Fig. 9a**):

```
> mygene<- getGene(cuff_data,'regucalcin')
> expressionBarplot(mygene)
```

13| Plot individual isoform expression levels of selected genes of interest with bar plots (**Fig. 9b**):

```
> expressionBarplot(isoforms(mygene))
```

14| Inspect the map files to count the number of reads that map to each chromosome (optional). From your working directory, enter the following at the command line:

```
$ for i in *thout/accepted_hits.bam; do
echo $i; samtools index $i ; done;
$ for i in *thout/accepted_hits.bam; do
echo $i; samtools idxstats $i ; done;
```

The first command creates a searchable index for each map file so that you can quickly extract the alignments for a particular region of the genome or collect statistics on the entire alignment file. The second command reports the number of fragments that map to each chromosome.

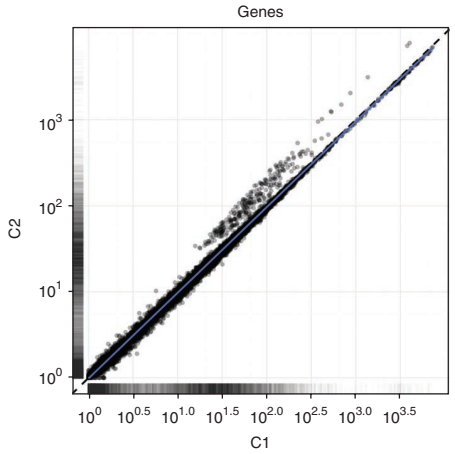


Figure 7 | CummeRbund scatter plots highlight general similarities and specific outliers between conditions C1 and C2. Scatter plots can be created from expression data for genes, splice isoforms, TSS groups or CDS groups.

Compare transcriptome assembly to the reference transcriptome (optional) ● TIMING **<5 min**

15| You can use a utility program included in the Cufflinks suite called Cuffcompare to compare assemblies against a reference transcriptome. Cuffcompare makes it possible to separate new genes from known ones, and new isoforms of known genes from known splice variants. Run Cuffcompare on each of the replicate assemblies as well as the merged transcriptome file:

```
$ find . -name transcripts.gtf > gtf_out_list.txt
$ cuffcompare -i gtf_out_list.txt -r genes.gtf
$ for i in 'find . -name *.tmap'; do echo $i; awk 'NR > 1 { s[$3]++ } END { \
    for (j in s) { print j, s[j] }} ' $i; done;
```

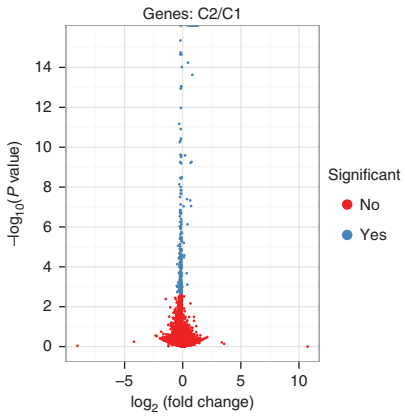


Figure 8 | CummeRbund volcano plots reveal genes, transcripts, TSS groups or CDS groups that differ significantly between the pairs of conditions C1 and C2.

The first command creates a file called gtf_out_list.txt that lists all of the GTF files in the working directory (or its sub-directories). The second command runs Cuffcompare, which compares each assembly GTF in the list to the reference annotation file genes.gtf. Cuffcompare produces a number of output files and statistics, and a full description of its behavior and functionality is out of the scope of this protocol. Please see the Cufflinks manual (http://cufflinks.cbcb.umd.edu/manual.html) for more details on Cuffcompare's output files and their formats. The third command prints a simple table for each assembly that lists how many transcripts in each assembly are complete matches to known transcripts, how many are partial matches and so on.

Record differentially expressed genes and transcripts to files for use in downstream analysis (optional) ● TIMING **<5 min**

16| You can use CummeRbund to quickly inspect the number of genes and transcripts that are differentially expressed between two samples. The R code below loads the results of Cuffdiff's analysis and reports the number of differentially expressed genes:

```
> library(cummeRbund)
> cuff_data <- readCufflinks('diff_out')
>
> cuff_data
CuffSet instance with:
    2 samples
    14353 genes
    26464 isoforms
    17442 TSS
    13727 CDS
    14353 promoters
    17442 splicing
    11372 relCDS
> gene_diff_data <- diffData(genes(cuff_data))
> sig_gene_data <- subset(gene_diff_data, (significant == 'yes'))
> nrow(sig_gene_data)
[1] 308
```

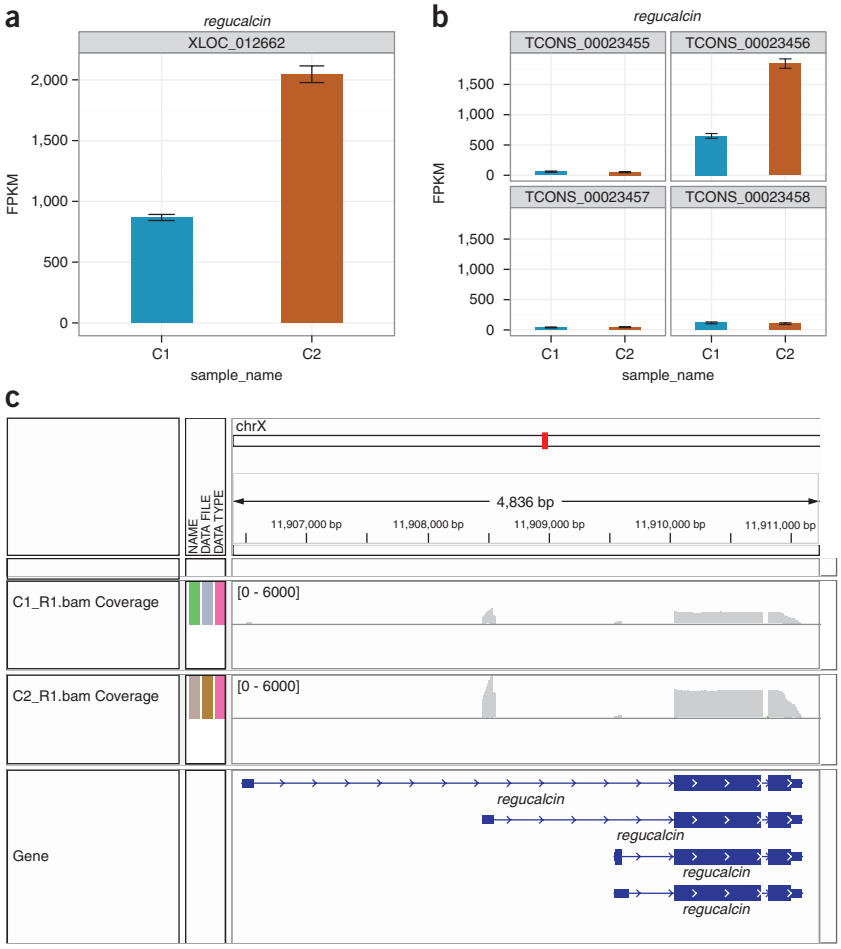


Figure 9 | Differential analysis results for *regucalcin*. (**a**) Expression plot shows clear differences in the expression of *regucalcin* across conditions C1 and C2, measured in FPKM (**Box 2**). Expression of a transcript is proportional to the number of reads sequenced from that transcript after normalizing for that transcript's length. Each gene and transcript expression value is annotated with error bars that capture both cross-replicate variability and measurement uncertainty as estimated by Cuffdiff's statistical model of RNA-seq. (**b**) Changes in *regucalcin* expression are attributable to a large increase in the expression of one of four alternative isoforms. (**c**) The read coverage, viewed through the genome browsing application IGV[42], shows an increase in sequencing reads originating from the gene in condition C2.

17| Similar snippets can be used to extract differentially expressed transcripts or differentially spliced and regulated genes:

```
> isoform_diff_data <- diffData(isoforms(cuff_data), 'C1', 'C2')
> sig_isoform_data <- subset(isoform_diff_data, (significant == 'yes'))
> nrow(sig_isoform_data)
> tss_diff_data <- diffData(TSS(cuff_data), 'C1', 'C2')
> sig_tss_data <- subset(tss_diff_data, (significant == 'yes'))
> nrow(sig_tss_data)
> cds_diff_data <- diffData(CDS(cuff_data), 'C1', 'C2')
> sig_cds_data <- subset(cds_diff_data, (significant == 'yes'))
> nrow(sig_cds_data)
> promoter_diff_data <- distValues(promoters(cuff_data))
> sig_promoter_data <- subset(promoter_diff_data, (significant == 'yes'))
> nrow(sig_promoter_data)
> splicing_diff_data <- distValues(splicing(cuff_data))
> sig_splicing_data <- subset(splicing_diff_data, (significant == 'yes'))
> nrow(sig_splicing_data)
> relCDS_diff_data <- distValues(relCDS(cuff_data))
> sig_relCDS_data <- subset(relCDS_diff_data, (significant == 'yes'))
> nrow(sig_relCDS_data)
```

18| The code above can also be modified to write out small files containing only the differentially expressed genes. These files may be more manageable for some spreadsheet software than the full output files produced by Cuffdiff. The R snippet below writes a table of differentially expressed genes into a file named diff_genes.txt.

```
> gene_diff_data <- diffData(genes(cuff_data))
> sig_gene_data <- subset(gene_diff_data, (significant == 'yes'))
> write.table(sig_gene_data, 'diff_genes.txt', sep='\t',
row.names = F, col.names = T, quote = F)
```

? TROUBLESHOOTING

Troubleshooting advice can be found in **Table 2**.

TABLE 2 | Troubleshooting table.

Step	Problem	Possible reason	Solution
1	TopHat cannot find Bowtie or the SAM tools	Bowtie and/or SAM tools binary executables are not in a directory listed in the PATH shell environment variable	Add the directories containing these executables to the PATH environment variable. See the man page of your UNIX shell for more details
2	Cufflinks crashes with a 'bad_alloc' error Cufflinks takes excessively long to finish	Machine is running out of memory trying to assemble highly expressed genes	Pass the –max-bundle-frags option to Cufflinks with a value of <1,000,000 (the default). Try 500,000 at first, and lower values if the error is still thrown
5	Cuffdiff crashes with a 'bad_alloc' error Cuffdiff takes excessively long to finish	Machine is running out of memory trying to quantify highly expressed genes	Pass the –max-bundle-frags option to Cuffdiff with a value of <1,000,000 (the default). Try 500,000 at first, and lower values if the error is still thrown
	Cuffdiff reports FPKM = 0 for all genes and transcripts	Chromosome names in GTF file do not match the names in the BAM alignment files	Use a GTF file and alignments that has matching chromosome names (e.g., the GTF included with an iGenome index)

● TIMING

Running this protocol on the example data provided will take ~18 h on a machine with eight processing cores and at least 8 GB of RAM. The time spent is nearly evenly divided between read alignment, assembly and differential analysis. However, larger data sets with more samples or deeper sequencing runs may take longer, and timing will obviously vary across different computers.

Step 1, align the RNA-seq reads to the genome: ~6 h
Steps 2–4, assemble expressed genes and transcripts: ~6 h
Step 5, identify differentially expressed genes and transcripts: ~6 h
Steps 6–14, explore differential analysis results with CummeRbund: variable
Step 15, compare transcriptome assembly to the reference transcriptome (optional): <5 min
Steps 16–18, record differentially expressed genes and transcripts to files for use in downstream analysis (optional): <5 min

ANTICIPATED RESULTS

RNA-seq read alignments

Accurate differential analysis depends on accurate spliced read alignments. Typically, at least 70% of RNA-seq reads should align to the genome, and lower mapping rates may indicate poor quality reads or the presence of contaminant. Users working with draft genome assemblies may also experience lower rates if the draft is missing a substantial fraction of the genes, or if the contigs and scaffolds have poor base call quality. The fraction of alignments that span splice junctions depends on read length and splicing complexity and the completeness of existing gene annotation, if available (see INTRODUCTION). **Table 3** lists the number of read alignments produced for each replicate during the execution of this protocol on the example data.

Transcriptome reconstruction

Because transcriptome annotations are still incomplete, most RNA-seq studies will reveal new genes and transcripts. However, some transcripts may be expressed at such low abundance that they may not be fully covered by sequencing reads

TABLE 3 | Expected read mapping statistics.

Chromosome	C1 rep 1	C1 rep 2	C1 rep 3	C2 rep 1	C2 rep 2	C2 rep 3
2L	4,643,234	4,641,231	4,667,543	4,594,554	4,586,366	4,579,505
2R	4,969,590	4,959,051	4,956,781	5,017,315	5,016,948	5,024,226
3L	4,046,843	4,057,512	4,055,992	4,111,517	4,129,373	4,104,438
3R	5,341,512	5,340,867	5,312,468	5,292,368	5,301,698	5,306,576
4	201,496	202,539	200,568	196,314	194,233	194,028
M	0	0	0	0	0	0
X	4,145,051	4,144,260	4,152,693	4,131,799	4,114,340	4,134,175
Total	23,347,726	23,345,460	23,346,045	23,343,867	23,342,958	23,342,948

and are thus only partially reconstructed by Cufflinks. The Cuffcompare utility used in Step 15 tabulates known and novel transcripts and can help triage newly discovered genes for further investigation.

Table 4 summarizes the transcriptome reconstructions for each replicate and the merged transcriptome assembly produced by Cufflinks from the example data. The merged assemblies (created in Step 4) contain more full-length reference transcripts and fewer partial transcripts than any of the individual replicate assemblies. In this simulation, we have sequenced only the reference transcriptome; hence, all of the 'novel' transfrags are in fact assembly artifacts. The merge contains more artifacts than any of the replicate assemblies as well. Note also that the merge with reference results in far more reference transcripts than the merge without reference assembly. This is because Cuffmerge includes all reference transcripts, even those that are not expressed in the assemblies being merged. Whenever possible, a reference annotation should be included during the merge.

Differential expression and regulation analysis

This protocol, if run correctly, should reveal markedly differentially expressed genes and transcripts between two or more conditions. In an ideal experiment, the protocol should not result in more spurious genes and transcripts than expected according to the false discovery rate (the default false discovery rate for Cuffdiff is 5%). However, poorly replicated conditions, inadequate depth or quality of sequencing and errors in the underlying annotation used to quantify genes and transcripts can all lead to artifacts during differential analysis. Transcriptome assembly errors during Steps 2–5 can contribute to missing or spuriously reported differential genes, and the prevalence of such errors is highly variable, depending on overall depth of sequencing, read and fragment length, gene density in the genome, transcriptome splicing complexity and transcript abundance.

Transcript expression levels vary over a dynamic range of 5–10 orders of magnitude and are often roughly log-normally distributed with an additional 'background' mode near 0. **Figure 6** shows the distribution of expression levels used in the example data set, which were generated from a real *Drosophila* sequencing experiment and represent typical expression profiles. The expression of each gene is compared in **Figure 7**, with the synthetically perturbed genes clearly visible. The 'volcano plot' in **Figure 8** relates the observed differences in gene expression to the significance associated with those changes under Cuffdiff's statistical model. Note that large fold changes in expression do not always imply statistical significance, as those fold changes may have been observed in genes that received little sequencing (because of low overall expression) or with many isoforms. The measured expression level for such genes tends to be highly variable across repeated sequencing experiments; thus, Cuffdiff places greater uncertainty on its significance of any observed fold changes. Cuffdiff also factors this uncertainty into the confidence intervals placed around the reported expression levels for genes and transcripts.

TABLE 4 | Transfrag reconstruction statistics for the example data set.

Assembly	Full length	Partial	Novel
C1 rep 1	8,549	940	1,068
C1 rep 2	8,727	958	1,151
C1 rep 3	8,713	996	1,130
C2 rep 1	8,502	937	1,118
C2 rep 2	8,749	945	1,158
C2 rep 3	8,504	917	1,091
Merged with reference	21,919	35	2,191
Merged without reference	10,056	590	1,952

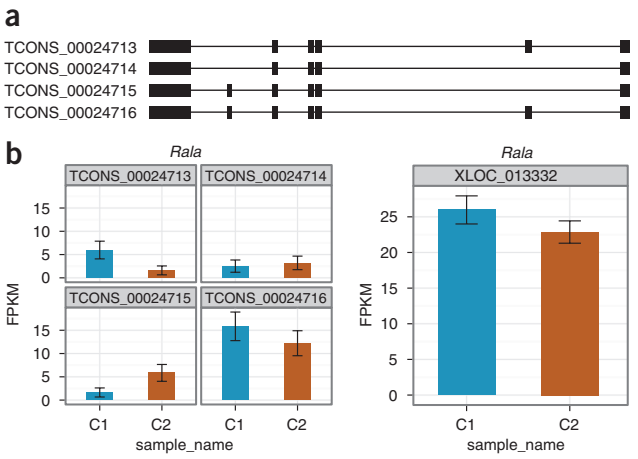


Figure 10 | Differential analysis results for *Rala*. (**a**) This gene has four isoforms in the merged assembly. (**b**) Cuffdiff identifies TCONS_00024713 and TCONS_00024715 as being significantly differentially expressed. The relatively modest overall change in gene-level expression, combined with high isoform-level measurement variability, leaves Cuffdiff unable to reject the null hypothesis that the observed gene level is attributable to measurement or cross-replicate variability.

Figure 9a shows the expression level of *regucalcin* (*D. melanogaster*; encoding CG1803 gene product from transcript CG1803-RA) in the two example conditions. Expression in condition 2 is approximately threefold higher than in condition 1, and the confidence interval is tight around each measurement. Tight confidence intervals are common around moderate and high gene expression values, especially when the genes have fewer than three or four isoforms. A plot of isoform-level expression values shows this change to be attributable to upregulation of a single regucalcin isoform (**Fig. 9b**). Again, confidence intervals are tight because overall depth of sequencing for this gene is high, and each isoform has a 'distinguishing' feature, such as a unique exon, covered by many reads in both samples. This allows Cuffdiff to calculate accurate measurements in which it has confidence. Increased sequenced depth on that isoform's unique initial exon is clearly visible (**Fig. 9c**), but we caution users from attempting to visually validate expression levels or fold change by viewing read depth in a browser. Expression depends on both depth and transcript length, and coverage histograms are susceptible to visual scaling artifacts introduced by graphical summaries of sequencing data.

In contrast to *regucalcin*, *Rala* (encoding Ras-related protein), which has lower expression and depth of sequencing than regucalcin, has larger isoform-level measurement uncertainty in expression; this, in turn, contributes to higher gene-level expression variance and prevents Cuffdiff from calling this gene's observed fold change significant (**Fig. 10**). Note that this gene also has significantly differentially expressed isoforms. However, as a gene's expression level is the sum of the expression levels of its isoforms, and some *Rala* isoforms are increased while others are decreased, the fold change in overall gene expression is modest.

The number of genes and transcripts reported as differentially expressed or regulated depends entirely on the conditions being compared. A comparison between true replicates should return few if any such genes and transcripts, whereas a comparison of different tissues or cell lines will generally return hundreds or even thousands of differentially expressed genes. It is not uncommon to find genes with relatively small fold changes (e.g., less than twofold) in expression marked as significant. This reflects the high overall sensitivity of RNA-seq compared with other whole-transcriptome expression quantification platforms. **Table 5** lists the values you should expect to see when running Steps 16 and 17 of the protocol on the example data.

TABLE 5 | Differentially expressed and regulated gene calls made for the example data set.

Differentially expressed genes	308
Differentially expressed transcripts	165
Differentially expressed TSS groups	226
Differentially expressed coding sequences	118
Differentially spliced TSS groups	75
Genes with differential promoter use	175
Genes with differential CDS output	42

ACKNOWLEDGMENTS We are grateful to D. Hendrickson, M. Cabili and B. Langmead for helpful technical discussions. The TopHat and Cufflinks projects are supported by US National Institutes of Health grants R01-HG006102 (to S.L.S.) and R01-HG006129-01 (to L.P.). C.T. is a Damon Runyon Cancer Foundation Fellow. L.G. is a National Science Foundation Postdoctoral Fellow. A.R. is a National Science Foundation Graduate Research Fellow. J.L.R. is a Damon Runyon-Rachleff, Searle, and Smith Family Scholar, and is supported by Director's New Innovator Awards (1DP2OD00667-01). This work was funded in part by the Center of Excellence in Genome Science from the US National Human Genome Research Institute (J.L.R.). J.L.R. is an investigator of the Merkin Foundation for Stem Cell Research at the Broad Institute.

AUTHOR CONTRIBUTIONS C.T. is the lead developer for the TopHat and Cufflinks projects. L.G. designed and wrote CummeRbund. D.K., H.P. and G.P. are developers of TopHat. A.R. and G.P. are developers of Cufflinks and its accompanying utilities. C.T. developed the protocol, generated the example experiment and performed the analysis. L.P., S.L.S. and C.T. conceived the TopHat and Cufflinks software projects. C.T., D.R.K. and J.L.R. wrote the manuscript.

COMPETING FINANCIAL INTERESTS The authors declare no competing financial interests.

Published online at http://www.natureprotocols.com/.
Reprints and permissions information is available online at http://www.nature.com/reprints/index.html.

1. Mortazavi, A., Williams, B.A., McCue, K., Schaeffer, L. & Wold, B. Mapping and quantifying mammalian transcriptomes by RNA-seq. *Nat. Methods* **5**, 621–628 (2008).
2. Cloonan, N. *et al.* Stem cell transcriptome profiling via massive-scale mRNA sequencing. *Nat. Methods* **5**, 613–619 (2008).

3. Nagalakshmi, U. *et al.* The transcriptional landscape of the yeast genome defined by RNA sequencing. *Science* **320**, 1344–1349 (2008).
4. Mardis, E.R. The impact of next-generation sequencing technology on genetics. *Trends Genet.* **24**, 133–141 (2008).
5. Adams, M.D. *et al.* Sequence identification of 2,375 human brain genes. *Nature* **355**, 632–634 (1992).
6. Cabili, M.N. *et al.* Integrative annotation of human large intergenic noncoding RNAs reveals global properties and specific subclasses. *Genes Dev.* **25**, 1915–1927 (2011).
7. Jiang, H. & Wong, W.H. Statistical inferences for isoform expression in RNA-seq. *Bioinformatics* **25**, 1026–1032 (2009).
8. Trapnell, C. *et al.* Transcript assembly and quantification by RNA-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.* **28**, 511–515 (2010).
9. Mortimer, S.A. & Weeks, K.M. A fast-acting reagent for accurate analysis of RNA secondary and tertiary structure by SHAPE chemistry. *J. Am. Chem. Soc.* **129**, 4144–4145 (2007).
10. Li, B., Ruotti, V., Stewart, R.M., Thomson, J.A. & Dewey, C.N. RNA-seq gene expression estimation with read mapping uncertainty. *Bioinformatics* **26**, 493–500 (2010).
11. Marioni, J.C., Mason, C.E., Mane, S.M., Stephens, M. & Gilad, Y. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome Res.* **18**, 1509–1517 (2008).
12. Garber, M., Grabherr, M.G., Guttman, M. & Trapnell, C. Computational methods for transcriptome annotation and quantification using RNA-seq. *Nat. Methods* **8**, 469–477 (2011).
13. Trapnell, C., Pachter, L. & Salzberg, S.L. TopHat: discovering splice junctions with RNA-seq. *Bioinformatics* **25**, 1105–1111 (2009).
14. Lister, R. *et al.* Hotspots of aberrant epigenomic reprogramming in human induced pluripotent stem cells. *Nature* **470**, 68–73 (2011).
15. Graveley, B.R. *et al.* The developmental transcriptome of *Drosophila melanogaster*. *Nature* **471**, 473–479 (2011).
16. Twine, N.A., Janitz, K., Wilkins, M.R. & Janitz, M. Whole transcriptome sequencing reveals gene expression and splicing differences in brain regions affected by Alzheimer's disease. *PLoS ONE* **6**, e16266 (2011).
17. Mizuno, H. *et al.* Massive parallel sequencing of mRNA in identification of unannotated salinity stress-inducible transcripts in rice (*Oryza sativa* L.). *BMC Genomics* **11**, 683 (2010).
18. Goecks, J., Nekrutenko, A. & Taylor, J. Galaxy Team Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.* **11**, R86 (2010).
19. Wu, T.D. & Nacu, S. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics* **26**, 873–881 (2010).
20. Wang, K. *et al.* MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Res.* **38**, e178 (2010).
21. Au, K.F., Jiang, H., Lin, L., Xing, Y. & Wong, W.H. Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Res.* **38**, 4570–4578 (2010).
22. Guttman, M. *et al.* Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. *Nat. Biotechnol.* **28**, 503–510 (2010).
23. Griffith, M. *et al.* Alternative expression analysis by RNA sequencing. *Nat. Methods* **7**, 843–847 (2010).
24. Katz, Y., Wang, E.T., Airoldi, E.M. & Burge, C.B. Analysis and design of RNA sequencing experiments for identifying isoform regulation. *Nat. Methods* **7**, 1009–1015 (2010).
25. Nicolae, M., Mangul, S., Măndoiu, I.I. & Zelikovsky, A. Estimation of alternative splicing isoform frequencies from RNA-seq data. *Algorithms Mol. Biol.* **6**, 9 (2011).
26. Anders, S. & Huber, W. Differential expression analysis for sequence count data. *Genome Biol.* **11**, R106 (2010).
27. Robinson, M.D., McCarthy, D.J. & Smyth, G.K. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* **26**, 139–140 (2009).
28. Wang, L., Feng, Z., Wang, X., Wang, X. & Zhang, X. DEGseq: an R package for identifying differentially expressed genes from RNA-seq data. *Bioinformatics* **26**, 136–138 (2010).
29. Grabherr, M.G. *et al.* Full-length transcriptome assembly from RNA-seq data without a reference genome. *Nat. Biotechnol.* **29**, 644–652 (2011).
30. Robertson, G. *et al.* *De novo* assembly and analysis of RNA-seq data. *Nat. Methods* **7**, 909–912 (2010).
31. Johnson, D.S., Mortazavi, A., Myers, R.M. & Wold, B. Genome-wide mapping of *in vivo* protein-DNA interactions. *Science* **316**, 1497–1502 (2007).
32. Ingolia, N.T., Ghaemmaghami, S., Newman, J.R.S. & Weissman, J.S. Genome-wide analysis *in vivo* of translation with nucleotide resolution using ribosome profiling. *Science* **324**, 218–223 (2009).
33. Langmead, B., Trapnell, C., Pop, M. & Salzberg, S.L. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.* **10**, R25 (2009).
34. Ferragina, P. & Manzini, G. An experimental study of a compressed index. *Information Sci.* **135**, 13–28 (2001).
35. Roberts, A., Pimentel, H., Trapnell, C. & Pachter, L. Identification of novel transcripts in annotated genomes using RNA-seq. *Bioinformatics* **27**, 2325–2329 (2011).
36. Li, J., Jiang, H. & Wong, W.H. Modeling non-uniformity in short-read rates in RNA-seq data. *Genome Biol.* **11**, R50 (2010).
37. Hansen, K.D., Brenner, S.E. & Dudoit, S. Biases in Illumina transcriptome sequencing caused by random hexamer priming. *Nucleic Acids Res.* **38**, e131 (2010).
38. Roberts, A., Trapnell, C., Donaghey, J., Rinn, J.L. & Pachter, L. Improving RNA-seq expression estimates by correcting for fragment bias. *Genome Biol.* **12**, R22 (2011).
39. Levin, J.Z. *et al.* Comprehensive comparative analysis of strand-specific RNA sequencing methods. *Nat. Methods* **7**, 709–715 (2010).
40. Hansen, K.D., Wu, Z., Irizarry, R.A. & Leek, J.T. Sequencing technology does not eliminate biological variability. *Nat. Biotechnol.* **29**, 572–573 (2011).
41. Wickham, H. *ggplot2: Elegant Graphics for Data Analysis (Use R)* p 224 (Springer, 2009).
42. Robinson, J.T. *et al.* Integrative genomics viewer. *Nat. Biotechnol.* **29**, 24–26 (2011).
43. Li, H. *et al.* The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**, 2078–2079 (2009).
44. Schatz, M.C., Langmead, B. & Salzberg, S.L. Cloud computing and the DNA data race. *Nat. Biotechnol.* **28**, 691–693 (2010).

Count-based differential expression analysis of RNA sequencing data using R and Bioconductor

Simon Anders[1], Davis J McCarthy[2,3], Yunshun Chen[4,5], Michal Okoniewski[6], Gordon K Smyth[4,7], Wolfgang Huber[1] & Mark D Robinson[8,9]

[1]Genome Biology Unit, European Molecular Biology Laboratory, Heidelberg, Germany. [2]Department of Statistics, University of Oxford, Oxford, UK. [3]Wellcome Trust Centre for Human Genetics, University of Oxford, Oxford, UK. [4]Bioinformatics Division, Walter and Eliza Hall Institute, Parkville, Victoria, Australia. [5]Department of Medical Biology, University of Melbourne, Melbourne, Victoria, Australia. [6]Functional Genomics Center UNI ETH, Zurich, Switzerland. [7]Department of Mathematics and Statistics, University of Melbourne, Melbourne, Victoria, Australia. [8]Institute of Molecular Life Sciences, University of Zurich, Zurich, Switzerland. [9]SIB Swiss Institute of Bioinformatics, University of Zurich, Zurich, Switzerland. Correspondence should be addressed to M.D.R. (mark.robinson@imls.uzh.ch) or W.H. (whuber@embl.de).

Published online 22 August 2013; doi:10.1038/nprot.2013.099

RNA sequencing (RNA-seq) has been rapidly adopted for the profiling of transcriptomes in many areas of biology, including studies into gene regulation, development and disease. Of particular interest is the discovery of differentially expressed genes across different conditions (e.g., tissues, perturbations) while optionally adjusting for other systematic factors that affect the data-collection process. There are a number of subtle yet crucial aspects of these analyses, such as read counting, appropriate treatment of biological variability, quality control checks and appropriate setup of statistical modeling. Several variations have been presented in the literature, and there is a need for guidance on current best practices. This protocol presents a state-of-the-art computational and statistical RNA-seq differential expression analysis workflow largely based on the free open-source R language and Bioconductor software and, in particular, on two widely used tools, DESeq and edgeR. Hands-on time for typical small experiments (e.g., 4–10 samples) can be <1 h, with computation time <1 d using a standard desktop PC.

INTRODUCTION

Applications of the protocol

The RNA-seq platform[1,2] addresses a multitude of applications, including relative expression analyses, alternative splicing, discovery of novel transcripts and isoforms, RNA editing, allele-specific expression and the exploration of non-model-organism transcriptomes.

Typically, tens of millions of sequences ('reads') are generated, and these, across several samples, form the starting point of this protocol. An initial and fundamental analysis goal is to identify genes whose expression level changes between conditions. In the simplest case, the aim is to compare expression levels between two conditions, e.g., stimulated versus unstimulated or wild type versus mutant. More complicated experimental designs can include additional experimental factors, potentially with multiple levels (e.g., multiple mutants, doses of a drug or time points) or may need to account for additional covariates (e.g. experimental batch or sex) or the pairing of samples (e.g., paired tumor and normal tissues from individuals). A crucial component of such an analysis is the statistical procedure used to call differentially expressed genes. This protocol covers two widely used tools for this task: DESeq[3] and edgeR[4–7], both of which are available as packages of the Bioconductor software development project[8].

Applications of these methods to biology and biomedicine are many. The methods described here are general and can be applied to situations in which observations are counts (typically, hundreds to tens of thousands of features of interest) and the goal is to discover changes in abundance. RNA-seq data are the standard use case (e.g., refs. 9,10), but many other differential analyses of counts are supported[11,12]. For RNA-seq data, the strategy taken is to count the number of reads that fall into annotated genes and to perform statistical analysis on the table of counts to discover quantitative changes in expression levels between experimental groups. This counting approach is direct, flexible and can be used for many types of count data beyond RNA-seq, such as comparative analysis of immunoprecipitated DNA[11–14] (e.g., ChIP-seq, MBD-seq[11,12]), proteomic spectral counts[15] and metagenomics data.

Development of the protocol

Figure 1 gives the overall sequence of steps, from read sequences to feature counting to the discovery of differentially expressed genes, with a concerted emphasis on quality checks throughout. After initial checks on sequence quality, reads are mapped to a reference genome with a splice-aware aligner[16]; up to this point, this protocol[3,6] is identical to many other pipelines (e.g., TopHat and Cufflinks[17]). From the set of mapped reads and either an annotation catalog or an assembled transcriptome, features, typically genes or transcripts, are counted and assembled into a table (rows for features and columns for samples). The statistical methods, which are integral to the differential expression discovery task, operate on a feature count table. Before the statistical modeling, further quality checks are encouraged to ensure that the biological question can be addressed. For example, a plot of sample relations can reveal possible batch effects and can be used to understand the similarity of replicates and the overall relationships between samples. After the statistical analysis of differential expression is carried out, a set of genes deemed to be differentially expressed or the corresponding statistics can be used in downstream interpretive analyses to confirm or generate further hypotheses.

Replication levels in designed experiments tend to be modest, often not much more than two or three. As a result, there is a need for statistical methods that perform well in small-sample situations. The low levels of replication rule out, for all practical purposes, distribution-free rank or permutation-based methods. Thus, for small-to-moderate sample sizes, the strategy used is to

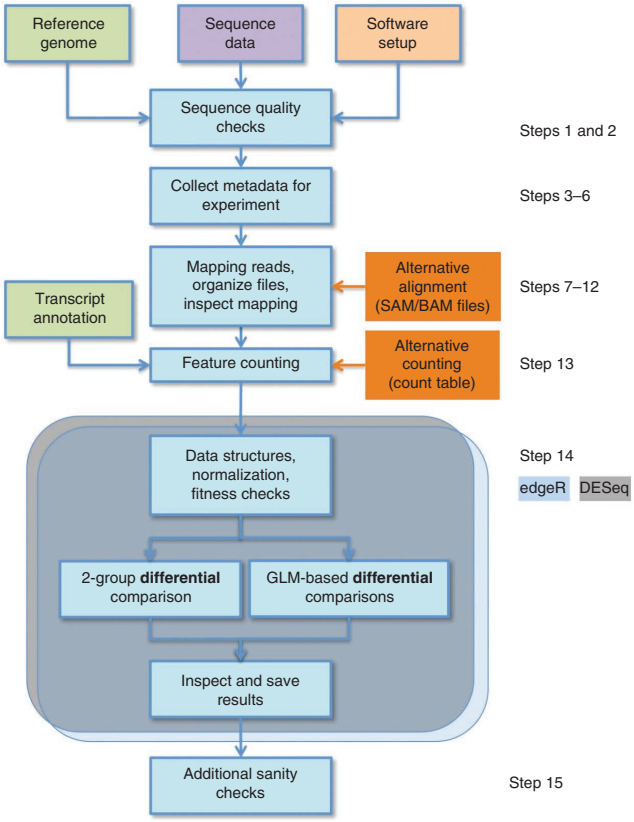


Figure 1 | Count-based differential expression pipeline for RNA-seq data using edgeR and/or DESeq. Many steps are common to both tools, whereas the specific commands are different (Step 14). Steps within the edgeR or DESeq differential analysis can follow two paths, depending on whether the experimental design is simple or complex. Alternative entry points to the protocol are shown in orange boxes.

make formal distributional assumptions about the data observed. The advantage of parametric assumptions is the ability, through the wealth of existing statistical methodology, to make inferences about parameters of interest (i.e., changes in expression). For genome-scale count data, including RNA-seq, a convenient and well-established approximation is the negative binomial (NB) model (**Box 1**), which represents a natural extension of the Poisson model (i.e., a mixture of gamma-distributed rates) that was used in early studies[18]; notably, Poisson variation can only describe technical (i.e., sampling) variation.

To analyze differential expression, this protocol focuses on DESeq and edgeR, which implement general differential analyses on the basis of the NB model. These tools differ in their look and feel, and they estimate the dispersions differently but offer overlapping functionality (**Box 2**).

Variations and extensions of the protocol

This protocol presents a workflow built from a particular set of tools, but it is modular and extensible; thus, alternatives that offer special features (e.g., counting by allele) or additional flexibility (e.g., specialized mapping strategy) can be inserted as necessary. **Figure 1** highlights straightforward alternative entry points to the protocol (orange boxes). The count-based pipeline discussed here can be used in concert with other tools. For example, for species without an available well-annotated genome reference, Trinity[19] or other assembly tools can be used to build a reference transcriptome; reads can then be aligned and counted, followed by the standard pipeline for differential analysis[20]. Similarly, to perform differential analysis on novel genes in otherwise annotated genomes, the protocol could be expanded to include merged per-sample assemblies (e.g., Cuffmerge within the Cufflinks package[17,21,22]) and used as input to counting tools.

The focus of this protocol is gene-level differential expression analysis. However, biologists are often interested in analyses beyond that scope, and many possibilities now exist as extensions of the count-based framework discussed here. The full details of such analyses are not covered here, and we make only a sketch of some promising approaches. First, an obvious extension to gene-level counting is exon-level counting, given a catalog of transcripts. Reads can be assigned to the exons that they align to and be counted. Reads spanning exon-exon junctions can be counted at the junction level. The DEXSeq package uses a generalized linear model (GLM) that tests whether particular exons in a gene are preferentially used in a condition, over and above changes in gene-level expression. In edgeR, a similar strategy is taken, except that testing is done at the gene level by effectively asking whether

Box 1 | The NB model

The NB model has been shown to be a good fit to RNA-seq data[7], yet it is flexible enough to account for biological variability. It provides a powerful framework (e.g., via GLMs) for analyzing arbitrarily complex experimental designs. NB models, as applied to genomic count data, make the assumption that an observation, say Y_{gj} (observed number of reads for gene g and sample j), has a mean μ_{gj} and a variance $\mu_{gj} + \varphi_g\mu^2$, where the dispersion $\varphi_g > 0$ represents overdispersion relative to the Poisson distribution[4]. The mean parameters μ_{gj} depend on the sequencing depth for sample j as well as on the amount of RNA from gene g in the sample. Statistical procedures can be formulated to test for changes in expression level between experimental conditions, possibly adjusting for batch effects or other covariates, and to estimate the log-fold changes in expression.

The dispersion φ_g represents the squared coefficient of variation of the true expression levels between biologically independent RNA samples under the same experimental conditions, and hence the square root of φ_g is called the biological coefficient of variation[7].

Obtaining good estimates of each gene's dispersion is critical for reliable statistical testing. Methods of estimating the genewise dispersion have received considerable attention[3,4,31,59]. Unless the number of samples is large, stable estimation of the dispersion requires some sort of sharing of information between genes. One can average the variability across all genes[5], or fit a global trend to the dispersion[3], or can seek a more general compromise between individual gene and global dispersion estimators[4].

Box 2 | Differences between DESeq and edgeR

The two packages described in this protocol, DESeq and edgeR, have similar strategies to perform differential analysis for count data. However, they differ in a few important areas. First, their look and feel differs. For users of the widely used limma package[60] (for analysis of microarray data), the data structures and steps in edgeR follow analogously. The packages differ in their default normalization: edgeR uses the trimmed mean of M values[56], whereas DESeq uses a relative log expression approach by creating a virtual library that every sample is compared against; in practice, the normalization factors are often similar. Perhaps most crucially, the tools differ in the choices made to estimate the dispersion. edgeR moderates feature-level dispersion estimates toward a trended mean according to the dispersion-mean relationship. In contrast, DESeq takes the maximum of the individual dispersion estimates and the dispersion-mean trend. In practice, this means DESeq is less powerful, whereas edgeR is more sensitive to outliers. Recent comparison studies have highlighted that no single method dominates another across all settings[27,61,62].

the exons are used proportionally across experiment conditions in the context of biological variation.

Comparison with other methods

Many tools exist for differential expression of counts, with slight variations of the method demonstrated in this protocol; these include, among others, baySeq[23], BBSeq[24], NOISeq[25] and QuasiSeq[26]. The advantages and disadvantages of each tool are difficult to elicit for a given data set, but simulation studies show that edgeR and DESeq, despite the influx of many new tools, remain among the top performers[27].

The count-based RNA-seq analyses presented here consider the total output of a locus, without regard to the isoform diversity that may be present. This is of course a simplification. In certain situations, gene-level count-based methods may not recover true differential expression when some isoforms of a gene are upregulated and others are downregulated[17,28]. Extensions of the gene-level count-based framework to differential exon usage are now available (e.g., DEXSeq[29]). Recently, approaches have been proposed to estimate transcript-level expression and to build the uncertainty of these estimates into a differential analysis at the transcript level (e.g., BitSeq[30]). Isoform deconvolution coupled with differential expression (e.g., Cuffdiff[17,21,22]) is a plausible and popular alternative, but in general, isoform-specific expression estimation remains a difficult problem, especially if sequence reads are short, if genes whose isoforms overlap substantially are analyzed or if very deeply sequenced data are unavailable. At present, isoform deconvolution methods and transcript-level differential expression methods only support two-group comparisons. In contrast, counting is straightforward regardless of the configuration, and depth of data and arbitrarily complex experiments are naturally supported through GLMs (see **Box 3** for further details on feature counting). Recently, a flexible Bayesian framework for the analysis of 'random' effects in the context of GLM models and RNA-seq count data was made available in the ShrinkSeq package[31]. In addition, count-based methods that operate at the exon level, which share the NB framework, and flexible coverage–based methods have become available to address the limitations of gene-level analyses[29,32,33]. These methods give a direct readout of differential exons, genes whose exons are used unequally or nonparallel coverage profiles, all of which reflect a change in isoform use.

Scope of this protocol

The aim of this protocol is to provide a concise workflow for a standard analysis, in a complete and easily accessible format, for users new to the field or to R. We describe a specific but very common analysis task: the analysis of an RNA-seq experiment, comparing two groups of samples that differ in terms of their experimental treatment. We also cover one common complication: the need to account for a blocking factor.

In practice, users will need to adapt this pipeline to account for the circumstances of their experiment. More complicated experimental designs will require further considerations not covered here. Therefore, we emphasize that this protocol is not meant to replace the existing user guides, vignettes and online documentation for the packages and functions described. These provide a large body of information that is helpful for tackling tasks that go beyond the single-standard workflow presented here.

In particular, edgeR and DESeq have extensive user guides, downloadable from http://www.bioconductor.org, which cover a wide range of relevant topics. Please consult these comprehensive resources for further details. Another rich resource for answers to commonly asked questions is the Bioconductor mailing list (http://bioconductor.org/help/mailing-list/) as well as online resources such as seqanswers.com (http://seqanswers.com/), stackoverflow.com (http://stackoverflow.com/) and biostars.org (http://www.biostars.org/).

Multiple entry points to the protocol

As mentioned, this protocol is modular, in that users can use an alternative aligner or a different strategy (or software package) to count features. Two notable entry points (see orange boxes in **Fig. 1**) for the protocol include starting with either (i) a set of sequence alignment map (SAM)/binary alignment map (BAM) files from an alternative alignment algorithm or (ii) a table of counts. With SAM/BAM files in hand, users can start at Step 13, although it is often invaluable to carry along metadata information (Steps 3–6), postprocessing the alignment files may still be necessary (Step 9) and spot checks on the mapping are often useful (Steps 10–12). With a count table in hand, users can start at Step 14, where again the metadata information (Steps 3–6) will be needed for the statistical analysis. For users who wish to learn the protocol using the data analyzed here, the **Supplementary Data** gives an archive containing: the intermediate COUNT files used, a collated count table (counts) in CSV (comma-separated values) format, the metadata table (samples) in CSV format and the CSV file that was downloaded from the National Center for Biotechnology Information (NCBI)'s short read archive (SRA).

Experimental design

Replication. Some of the early RNA-seq studies were performed without biological replication. If the purpose of the experiment

Box 3 | Feature counting

In principle, counting reads that map to a catalog of features is straightforward. However, a few subtle decisions need to be made. For example, how should reads that fall within intronic regions (i.e., between two known exons) or beyond the annotated regions be counted? Ultimately, the answer to this question is guided by the chosen catalog that is presented to the counting software; depending on the protocol used, users should be conscious to include all features that are of interest, such as polyadenylated RNAs, small RNAs, long intergenic noncoding RNAs and so on. For simplicity and to avoid problems with mismatching chromosome identifiers and inconsistent coordinate systems, we recommend using the curated FASTA files and GTF files from Ensembl or the prebuilt indices packaged with GTF files from http://tophat.cbcb.umd.edu/igenomes.shtml whenever possible.

Statistical inference based on the NB distribution requires raw read counts as input. This is required to correctly model the Poisson component of the sample-to-sample variation. Therefore, it is crucial that units of evidence for expression are counted. No prior normalization or other transformation should be applied, including quantities such as RPKM (reads per kilobase model), FPKM (fragments per kilobase model) or otherwise depth-adjusted read counts. Both DESeq and edgeR internally keep the raw counts and normalization factors separate, as this full information is needed to correctly model the data. Notably, recent methods to normalize RNA-seq data for sample-specific G+C content effects use offsets that are presented to the GLM, while maintaining counts on their original scale[63,64].

Each paired-end read represents a single fragment of sequenced DNA, yet (at least) two entries for the fragment will appear in the corresponding BAM files. Some simplistic early methods that operated on BAM files considered these as separate entries, which led to overcounting and would ultimately overstate the significance of differential expression.

Typically, there will be reads that cannot be uniquely assigned to a gene, either because the read was aligned to multiple locations (multi-reads) or the read's position is annotated as part of several overlapping features. For the purpose of calling differential expression, such reads should be discarded. Otherwise, genuine differential expression of one gene might cause another gene to appear differentially expressed, erroneously, if reads from the first gene are counted for the second due to assignment ambiguity. In this protocol, we use the tool htseq-count of the Python package HTSeq, using the default union-counting mode; more details can be found at http://www-huber.embl.de/users/anders/HTSeq/doc/count.html. In addition, Bioconductor now offers various facilities for feature counting, including *easyRNASeq* in the easyRNASeq package[65], the *summarizeOverlaps* function in the GenomicRanges package and *qCount* in the QuasR (http://www.bioconductor.org/packages/release/bioc/html/QuasR.html) package.

is to make a general statement about a biological condition of interest (in statistical parlance, a population), for example, the effect of treating a certain cell line with a particular drug, then an experiment without replication is insufficient. Rapid developments in sequencing reduce technical variation but cannot possibly eliminate biological variability[34]. Technical replicates are suited to studying properties of the RNA-seq platform[16], but they do not provide information about the inherent biological variability in the system or the reproducibility of the biological result (for instance, its robustness to slight variations in cell density, passage number, drug concentration or medium composition). In other words, experiments without biological replication are suited to making a statement regarding one particular sample that existed on one particular day in one particular laboratory, but not whether anybody could reproduce this result. When no replicates are available, experienced analysts may still proceed, using one of the following options: (i) by performing a descriptive analysis with no formal hypothesis testing; (ii) by selecting a dispersion value on the basis of past experience; or (iii) by using housekeeping genes to estimate variability across all samples in the experiment.

In this context, it is helpful to remember the distinction between designed experiments, in which a well-characterized system (e.g., a cell line or a laboratory mouse strain) undergoes a fully controlled experimental procedure with minimal unintended variation, and observational studies, in which samples are often those of convenience (e.g., patients arriving at a clinic) that have been subjected to many uncontrolled environmental and genetic factors. Replication levels of two or three are often a practical compromise between cost and benefit for designed experiments, but for observational studies, typically much larger group sizes (dozens or hundreds) are needed to reliably detect biologically meaningful results.

Confounding factors. In many cases, data are collected over time. In this situation, researchers should be mindful of factors that may unintentionally confound their results (e.g., batch effects), such as changes in reagent chemistry or software versions used to process their data[35]. Users should make a concerted effort to reduce confounding effects through experimental design (e.g., randomization, blocking[36]) and to keep track of versions, conditions (e.g., operators) of every sample, in the hope that these factors (or surrogates of them) can be differentiated from the biological factor(s) of interest in the downstream statistical modeling. In addition, there are emerging tools available that can discover and help eliminate unwanted variation in larger data sets[37,38], although these are relatively untested for RNA-seq data at present.

Software implementation. There are advantages to using a small number of software platforms for such a workflow, and these include simplified maintenance, training and portability. In principle, it is possible to do all computational steps in R and Bioconductor; however, for a few of the steps, the most mature and widely used tools are outside Bioconductor. Here R and Bioconductor are adopted to tie together the workflow and provide data structures, and their unique strengths in workflow components are leveraged, including statistical algorithms, visualization

Box 4 | Software versions

The original of this document was produced with Sweave[66] using the following versions of R and its packages:

```
> sessionInfo()
```

R output:

```
R version 3.0.0 (2013-04-03)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
[1] LC_CTYPE=en_CA.UTF-8       LC_NUMERIC=C                 LC_TIME=en_CA.UTF-8
[4] LC_COLLATE=en_CA.UTF-8     LC_MONETARY=en_CA.UTF-8      LC_MESSAGES=en_CA.UTF-8
[7] LC_PAPER=C                 LC_NAME=C                    LC_ADDRESS=C
[10] LC_TELEPHONE=C            LC_MEASUREMENT=en_CA.UTF-8   LC_IDENTIFICATION=C

attached base packages:
[1] parallel stats graphics grDevices utils datasets methods base

other attached packages:
[1]  DESeq_1.12.0      locfit_1.5-9.1       Biobase_2.20.0       edgeR_3.2.3
[5]  limma_3.16.2      ShortRead_1.18.0     latticeExtra_0.6-24  RColorBrewer_1.0-5
[9]  Rsamtools_1.12.3  lattice_0.20-15      Biostrings_2.28.0    GenomicRanges_1.12.4
[13] IRanges_1.18.1    BiocGenerics_0.6.0   CacheSweave_0.6-1    stashR_0.3-5
[17] filehash_2.2-1

loaded via a namespace (and not attached):
[1]   annotate_1.38.0   AnnotationDbi_1.22.5   bitops_1.0-5        DBI_0.2-7
[5]   digest_0.6.3      genefilter_1.42.0      geneplotter_1.38.0  grid_3.0.0
[9]   hwriter_1.3       RSQLite_0.11.3         splines_3.0.0       stats4_3.0.0
[13]  survival_2.37-4   tools_3.0.0            XML_3.96-1.1        xtable_1.7-1
[17]  zlibbioc_1.6.0
```

The versions of software packages used can be captured with the following commands (output is shown below each command):

```
> system("bowtie2 --version | grep align", intern=TRUE)
[1] "/usr/local/software/bowtie2-2.1.0/bowtie2-align version 2.1.0"
> system("tophat --version", intern=TRUE)
[1] "TopHat v2.0.8"
> system("htseq-count | grep version", intern=TRUE)
[1] "General Public License v3. Part of the 'HTSeq' framework, version 0.5.3p9."
> system("samtools 2 >&1 | grep Version", intern=TRUE)
[1] "Version: 0.1.18 (r982:295)"
```

and computation with annotation databases. Another major advantage of an R-based system, in terms of achieving best practices in genomic data analysis, is the opportunity for an interactive analysis whereby spot checks are made throughout the pipeline to guide the analyst. In addition, a wealth of tools is available for exploring, visualizing and cross-referencing genomic data. Although they are not used here directly, additional features of Bioconductor are readily available that will often be important for scientific projects that involve an RNA-seq analysis, including access to many different file formats, range-based computations, annotation resources, manipulation of sequence data and visualization.

In what follows, all Unix commands run at the command line appear in `Courier` font, prefaced by a dollar sign ($):

```
$ my_unix_command
```

whereas R functions in the text appear as *myFunction*, and (typed) R input commands and output commands appear in **`bold`** and `plain` Courier font, respectively:

```
> x = 1:10
> median(x)
[1] 5.5
```

Note that in R, the operators = and <- can both be used for variable assignment (i.e., $z = 5$ and $z <- 5$ produce the same result, a new variable z with a numeric value). In this protocol, we use the = notation; in other places, users may also see the <- notation.

Constructing a metadata table (Steps 3–6). In general, we recommend starting from a sample metadata table that contains

sample identifiers, experimental conditions, blocking factors and file names. In our example, we construct this table from a file downloaded from the SRA. Users will often obtain a similar table from a local laboratory information management system (LIMS) or sequencing facility and can adapt this strategy to their own data sets.

Mapping reads to reference genome (Steps 7 and 8). In the protocol, R is used to tie the pipeline together (i.e., loop through the set of samples and construct the full tophat2 command), with the hope of reducing typing and copy-and-paste errors. Many alternatives and variations are possible: users can use R to create and call the tophat2 commands, to create the commands (and call tophat2 independently of a Unix shell), or to assemble the commands manually independently of R. tophat2 creates a directory for each sample with the mapped reads in a BAM file, called accepted_hits.bam. Note that BAM files, and equivalently SAM files (an uncompressed text version of BAM), are the de facto standard file for alignments. Therefore, alternative mapping tools that produce BAM/SAM files could be inserted into the protocol at this point.

Organizing BAM and SAM files (Step 9). The set of files containing mapped reads (from tophat2, accepted_hits.bam) (typically) needs to be transformed before it can be used with other downstream tools. In particular, the samtools command is used to prepare variations of the mapped reads. Specifically, a sorted and indexed version of the BAM file is created, which can be used in genome browsers such as IGV; a sorted-by-name SAM file is created, which is compatible with the feature-counting software of htseq-count. Alternative feature-counting tools (e.g., in Bioconductor) may require different inputs.

Design matrix. For more complex designs (i.e., beyond two-group comparisons), users need to provide a design matrix that specifies the factors that are expected to affect expression levels. As mentioned above, GLMs can be used to analyze arbitrarily complex experiments, and the design matrix is the means by which the experimental design is described mathematically, including both biological factors of interest and other factors not of direct interest, such as batch effects. For example, Section 4.5 of the edgeR User's Guide ('RNA-seq of pathogen inoculated *Arabidopsis* with batch effects') or Section 4 of the DESeq vignette ('Multi-factor designs') presents worked case studies with batch effects. The design matrix is central for such complex differential expression analyses, and users may wish to consult a linear modeling textbook[39] or a local statistician to make sure their design matrix is appropriately specified.

Reproducible research. We recommend that users keep a record of all commands (R and Unix) and the software versions used in their analysis so that other researchers (e.g., collaborators, reviewers) can reproduce the results (**Box 4**). In practice, this is best achieved by keeping the complete transcript of the computer commands interweaved with the textual narrative in a single, executable document[40]. R provides many tools to facilitate the authoring of executable documents, including the *Sweave* function and the knitR package. The *sessionInfo* function helps with documenting package versions and related information. A recent integration with Rstudio is rpubs.com (http://rpubs.com/), which provides seamless integration of 'mark-down' text with R commands for easy web-based display. For language-independent authoring, a powerful tool is provided by Emacs org-mode.

MATERIALS

EQUIPMENT

▲ **CRITICAL** For many of the software packages listed below, new features and optimizations are constantly developed and released, so we highly recommend using the most recent stable version as well as reading the (corresponding) documentation for the version used. The package versions used in the production of this article are given in **Box 4**

Operating system

- This protocol assumes users have a Unix-like operating system (i.e., Linux or MacOS X), with a bash shell or similar. All commands given here are meant to be run in a terminal window. Although it is possible to follow this protocol with a Microsoft Windows machine (e.g., using the Unix-like Cygwin; http://www.cygwin.com/), the additional steps required are not discussed here

Software

- An aligner to map short reads to a genome that is able to deal with reads that straddle introns[16]. The aligner tophat2 (refs. 21,41) is illustrated here, but others, such as GSNAP[42], SpliceMap[43], Subread[44] or STAR[45], can be used
- (Optional) A tool to visualize alignment files, such as the Integrated Genome Viewer (IGV[46], or Savant[47,48]). IGV is a Java tool with 'web start' (downloadable from http://www.broadinstitute.org/software/igv/log-in), i.e., it can be started from a web browser and needs no explicit installation at the operating system level, provided a Java Runtime Environment is available
- The R statistical computing environment, downloadable from http://www.r-project.org/
- A number of Bioconductor[8] packages, specifically ShortRead[49], DESeq[3] and edgeR[6,7], and possibly GenomicRanges, GenomicFeatures and org.Dm.eg.db, as well as their dependencies
- The samtools program[50] (http://samtools.sourceforge.net/) for manipulation of SAM- and BAM-formatted files
- The HTSeq package (http://www-huber.embl.de/users/anders/HTSeq/doc/overview.html) for counting of mapped reads
- (Optional) If users wish to work with data from the SRA, they will need the SRA Toolkit (http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?cmd=show&f=software&m=software&s=software)

Input file formats

- In general, the starting point is a collection of FASTQ files, the commonly used format for reads from Illumina sequencing machines. The modifications necessary for mapping reads from other platforms are not discussed here

Example data

- The data set published by Brooks *et al.*[51] is used here to demonstrate the workflow. This data set consists of seven RNA-seq samples, each a cell

culture of *Drosophila melanogaster* S2 cells. Three samples were treated with siRNA targeting the splicing factor *pasilla* (CG1844) ('knockdown') and four samples are untreated ('control'). Our aim is to identify genes that change in expression between knockdown and control. Brooks *et al.*[51] have sequenced some of their libraries in single-end mode and others in paired-end mode. This allows us to demonstrate two variants of the workflow: if we ignore the differences in library type, the samples only differ by their experimental condition (knockdown or control), and the analysis is a simple comparison between two sample groups. We refer to this setting as an experiment with a simple design. If we want to account for library type as a blocking factor, our samples differ in more than one aspect (i.e., we have a complex design). To deal with the latter scenario, we use edgeR and DESeq's functions to fit GLMs.

EQUIPMENT SETUP

Install bowtie2, tophat2 and samtools Download and install samtools from http://samtools.sourceforge.net. bowtie2 and tophat2 have binary versions available for Linux and Mac OS X platforms. These can be downloaded from http://bowtie-bio.sourceforge.net/index.shtml and http://tophat.cbcb.umd.edu/. Consult the documentation on those sites for further information if necessary.

Install R and required Bioconductor packages Download the latest release version of R from http://cran.r-project.org/ and install it. Consult the R Installation and Administration manual if necessary. A useful quick reference for R commands can be found at http://cran.r-project.org/doc/contrib/Short-refcard.pdf. To install Bioconductor packages, start R by issuing the command R in a terminal window and type:

```
> source("http://www.Bioconductor.org/biocLite.R")
> biocLite("BiocUpgrade")
> biocLite( c("ShortRead","DESeq", "edgeR") )
```

This retrieves an automatic installation tool (*biocLite*) and installs the version-matched packages. In addition, the installation tool will automatically download and install all other prerequisite packages. Versions of Bioconductor packages are matched to versions of R. Hence, to use current versions of Bioconductor packages, it is necessary to use a current version of R. Note that R and Bioconductor maintain a stable release version and a development version at all times. Unless a special need exists for a particular new functionality, users should use the release version.

Download the example data To download SRA repository data, an automated process may be desirable. For example, from http://www.ncbi.nlm.nih.gov/sra?term=SRP001537 (the entire experiment corresponding to GEO accession GSE18508), users can download a table of the metadata into a comma-separated tabular file 'SraRunInfo.csv' (see the **Supplementary Data**, which contains an archive of various files used in this protocol). To do this, click on 'Send to:' (top right corner), select 'File', select format 'RunInfo' and click on 'Create File'. Read this CSV file 'SraRunInfo.csv' into R, and select the subset of samples that we are interested in (using R's string matching function grep), corresponding to the 22 SRA files shown in **Figure 2** by:

```
> sri = read.csv("SraRunInfo.csv",
                 stringsAsFactors=FALSE)
> keep = grep("CG8144|Untreated-",
              sri$ LibraryName)
> sri = sri[keep,]
```

The following R commands automate the download of the 22 SRA files to the current working directory (the functions *getwd* and *setwd* can be used to retrieve and set the working directory, respectively):

```
> fs = basename(sri$download_path)
> for(i in 1:nrow(sri))
    download.file(sri$download_path[i], fs[i])
```

▲ **CRITICAL** This download is only required if data originate from the SRA. Brooks *et al.*[51] deposited their data in the SRA of the NCBI's Gene Expression Omnibus (GEO)[52] under accession number GSE18508 and a subset of this data set is used here to illustrate the pipeline. Specifically, SRA files corresponding to the 4 'Untreated' (control) and 3 'CG8144 RNAi' (knockdown) samples need to be downloaded.

Alternative download tools The R-based download of files described above is just one way to capture several files in a semiautomatic fashion. Users can alternatively use the batch tools wget (Unix/Linux) or curl (Mac OS X), or

SraRunInfo.csv

	A	B	C	D	E	F
1	Run	ftp_path	LibraryName	LibraryLayout	Study	
2	SRR031718	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031718/SRR031718.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
3	SRR031719	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031719/SRR031719.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
4	SRR031720	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031720/SRR031720.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
5	SRR031721	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031721/SRR031721.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
6	SRR031722	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031722/SRR031722.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
7	SRR031723	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031723/SRR031723.sra	S2_DRSC_CG8144_RNAi-1	SINGLE	SRP001537	
8	SRR031724	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031724/SRR031724.sra	S2_DRSC_CG8144_RNAi-3	PAIRED	SRP001537	
9	SRR031725	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031725/SRR031725.sra	S2_DRSC_CG8144_RNAi-3	PAIRED	SRP001537	
10	SRR031726	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031726/SRR031726.sra	S2_DRSC_CG8144_RNAi-4	PAIRED	SRP001537	
11	SRR031727	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031727/SRR031727.sra	S2_DRSC_CG8144_RNAi-4	PAIRED	SRP001537	
12	SRR031708	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031708/SRR031708.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
13	SRR031709	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031709/SRR031709.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
14	SRR031710	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031710/SRR031710.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
15	SRR031711	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031711/SRR031711.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
16	SRR031712	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031712/SRR031712.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
17	SRR031713	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031713/SRR031713.sra	S2_DRSC_Untreated-1	SINGLE	SRP001537	
18	SRR031714	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031714/SRR031714.sra	S2_DRSC_Untreated-3	PAIRED	SRP001537	
19	SRR031715	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031715/SRR031715.sra	S2_DRSC_Untreated-3	PAIRED	SRP001537	
20	SRR031716	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031716/SRR031716.sra	S2_DRSC_Untreated-4	PAIRED	SRP001537	
21	SRR031717	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031717/SRR031717.sra	S2_DRSC_Untreated-4	PAIRED	SRP001537	
22	SRR031728	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031728/SRR031728.sra	S2_DRSC_Untreated-6	SINGLE	SRP001537	
23	SRR031729	ftp://ftp-private.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/SRR031729/SRR031729.sra	S2_DRSC_Untreated-6	SINGLE	SRP001537	
24						

Figure 2 | Screenshot of Metadata available from SRA.

download using a web browser. The (truncated) verbose output of the above R download commands looks as follows:

```
trying URL 'ftp://ftp-private.ncbi.nlm.nih.
gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/
SRR031714/SRR031714.sra'

ftp data connection made, file length 415554366
bytes

opened URL

==================================

downloaded 396.3 Mb

trying URL 'ftp://ftp-private.ncbi.nlm.nih.
gov/sra/sra-instant/reads/ByRun/sra/SRR/SRR031/
SRR031715/SRR031715.sra'

ftp data connection made, file length 409390212
bytes

opened URL

==================================

downloaded 390.4 Mb

[... truncated ...]
```

Convert SRA to FASTQ format Typically, sequencing data from a sequencing facility will come in (compressed) FASTQ format. The SRA, however, uses its own, compressed, SRA format. In order to convert the example data to FASTQ, use the fastq-dump command from the SRA Toolkit on each SRA file. Note that the use of R's *system* command is just one possibility. Users may choose to type the 22 fastq-dump commands manually into the Unix shell rather than using R to construct them. R can be used to construct the required shell commands, starting from the 'SraRunInfo.csv' metadata table, as follows:

```
> stopifnot( all(file.exists(fs)) ) # assure FTP
  download was successful
> for(f in fs) {
  cmd = paste("fastq-dump --split-3", f)
  cat(cmd,"\n")
  system(cmd) # invoke command
}
```

Using the *cat* command It is not absolutely necessary to use *cat* to print out the current command, but it serves the purpose of knowing what is currently running in the shell:

```
fastq-dump --split-3 SRR031714.sra

Written 5327425 spots for SRR031714.sra

Written 5327425 spots total

fastq-dump --split-3 SRR031715.sra

Written 5248396 spots for SRR031715.sra

Written 5248396 spots total

[... truncated ...]
```

▲ **CRITICAL** Be sure to use the --split-3 option, which splits mate-pair reads into separate files. After this command, single and paired-end data will produce one and two FASTQ files, respectively. For paired-end data, the file names will be suffixed _1.FASTQ and _2.FASTQ; otherwise, a single file with the extension .FASTQ will be produced.

Download the reference genome Download the reference genome sequence for the organism under study in (compressed) FASTA format. Some useful resources, among others, include: the general Ensembl FTP server (http://www.ensembl.org/info/data/ftp/index.html), the Ensembl plants FTP server (http://plants.ensembl.org/info/data/ftp/index.html), the Ensembl metazoa FTP server (http://metazoa.ensembl.org/info/data/ftp/index.html) and the University of California Santa Cruz (UCSC) current genomes FTP server (ftp://hgdownload.cse.ucsc.edu/goldenPath/currentGenomes/).

Using Ensembl For Ensembl, choose the 'FASTA (DNA)' link instead of 'FASTA (cDNA)', as alignments to the genome, not the transcriptome, are desired. For *D. melanogaster*, the file labeled 'toplevel' combines all chromosomes. Do not use the 'repeat-masked' files (indicated by 'rm' in the file name), as handling repeat regions should be left to the alignment algorithm. The *Drosophila* reference genome can be downloaded from Ensembl and uncompressed using the following Unix commands:

```
$ wget ftp://ftp.ensembl.org/pub/release-70/
fasta/drosophila_melanogaster/dna/Drosophila_
melanogaster.BDGP5.70.dna.toplevel.fa.gz

$ gunzip Drosophila_melanogaster.BDGP5.70.dna.
toplevel.fa.gz
```

For genomes provided by UCSC, users can select their genome of interest, proceed to the 'bigZips' directory and download the 'chromFa.tar.gz'; as above, this could be done using the wget command. Note that bowtie2 and tophat2 indices for many commonly used reference genomes can be downloaded directly from http://tophat.cbcb.umd.edu/igenomes.shtml.

Get gene model annotations Download a gene transfer format (GTF) file with gene models for the organism of interest. For species covered by Ensembl, the Ensembl FTP site mentioned above contains links to such files. The gene model annotation for *D. melanogaster* can be downloaded and uncompressed using:

```
$ wget ftp://ftp.ensembl.org/pub/release-70/gtf/
drosophila_melanogaster/Drosophila_melanogaster.
BDGP5.70.gtf.gz

$ gunzip Drosophila_melanogaster.BDGP5.70.gtf.gz
```

▲ **CRITICAL** Make sure that the gene annotation uses the same coordinate system as the reference FASTA file. Here, both files use BDGP5 (i.e., release 5 of the assembly provided by the Berkeley *Drosophila* Genome Project), as is apparent from the file names. To be on the safe side, here, we recommend always downloading the FASTA reference sequence and the GTF annotation data from the same resource provider.

▲ **CRITICAL** As an alternative, the UCSC Table Browser (http://genome.ucsc.edu/cgi-bin/hgTables) can be used to generate GTF files on the basis of a selected annotation (e.g., RefSeq genes). However, at the time of writing, GTF files obtained from the UCSC Table Browser do not contain correct gene IDs, which can cause problems with downstream tools such as htseq-count, unless they are corrected manually.

Build the reference index Before reads can be aligned, the reference FASTA files need to be preprocessed into an index that allows the aligner easy access. To build a bowtie2-specific index from the FASTA file mentioned above, use the command:

```
$ bowtie2-build -f Drosophila_melanogaster.
BDGP5.70.dna.toplevel.fa Dme1_BDGP5_70
```

A set of BT2 files will be produced, with names starting with Dme1_BDGP_70 specified above. This procedure needs to be run only once for each reference genome used. As mentioned, pre-built indices for many commonly used genomes are available from http://tophat.cbcb.umd.edu/igenomes.shtml.

PROCEDURE

Assess sequence quality control with ShortRead ● TIMING ~2 h

1| At the R prompt, type the commands (you may first need to use *setwd* to set the working directory to where the FASTQ files are situated):

```
> library("ShortRead")
> fqQC = qa(dirPath=".", pattern=".fastq$", type="fastq")
> report(fqQC, type="html", dest="fastqQAreport")
```

? TROUBLESHOOTING

2| Use a web browser to inspect the generated HTML file (here, stored in the 'fastqQAreport' directory) with the quality-assessment report (see ANTICIPATED RESULTS for further details)

Collect metadata of experimental design ● TIMING <1 h

3| Create a table of metadata called 'samples' (see 'Constructing metadata table' in Experimental Design). This step needs to be adapted for each data set, and many users may find a spreadsheet program useful for this step, from which data can be imported into the table samples by the *read.csv* function. For our example data, we chose to construct the samples table programmatically from the table of SRA files.

4| Collapse the initial table (sri) to one row per sample:

```
> sri$LibraryName = gsub("S2_DRSC_","",sri$LibraryName) # trim label
> samples = unique(sri[,c("LibraryName","LibraryLayout")])
> for(i in seq_len(nrow(samples))) {
  rw = (sri$LibraryName==samples$LibraryName[i])
  if(samples$LibraryLayout[i]=="PAIRED") {
    samples$fastq1[i] = paste0(sri$Run[rw],"_1.fastq",collapse=",")
    samples$fastq2[i] = paste0(sri$Run[rw],"_2.fastq",collapse=",")
   } else {
    samples$fastq1[i] = paste0(sri$Run[rw],".fastq",collapse=",")
    samples$fastq2[i] = ""
   }
}
```

5| Add important or descriptive columns to the metadata table (experimental groupings are set on the basis of the 'LibraryName' column, and a label is created for plotting):

```
> samples$condition = "CTL"
> samples$condition[grep("RNAi",samples$LibraryName)] = "KD"
> samples$shortname = paste(substr(samples$condition,1,2),
                            substr(samples$LibraryLayout,1,2),
                            seq_len(nrow(samples)), sep=".")
```

6| As the downstream statistical analysis of differential expression relies on this table, carefully inspect (and correct, if necessary) the metadata table. In particular, verify that there exists one row per sample, that all columns of information are populated and that the file names, labels and experimental conditions are correct.

```
> samples
```

R output:

	LibraryName	Library Layout	fastq1	fastq2	condition	shortname
1	Untreated-3	PAIRED	SRR031714_1.fastq,...	SRR031714_2.fastq,...	CTL	CT.PA.1
2	Untreated-4	PAIRED	SRR031716_1.fastq,...	SRR031716_2.fastq,...	CTL	CT.PA.2
3	CG8144_RNAi-3	PAIRED	SRR031724_1.fastq,...	SRR031724_2.fastq,...	KD	KD.PA.3
4	CG8144_RNAi-4	PAIRED	SRR031726_1.fastq,...	SRR031726_2.fastq,...	KD	KD.PA.4
5	Untreated-1	SINGLE	SRR031708.fastq,...		CTL	CT.SI.5
6	CG8144_RNAi-1	SINGLE	SRR031718.fastq,...		KD	KD.SI.6
7	Untreated-6	SINGLE	SRR031728.fastq,...		CTL	CT.SI.7

Align the reads (using tophat2) to the reference genome ● TIMING ~45 min per sample

7| By using R string manipulation, construct the Unix commands to call tophat2. Given the metadata table samples, it is convenient to use R to create the list of shell commands, as follows:

```
> gf = "Drosophila_melanogaster.BDGP5.70.gtf"
> bowind = "Dme1_BDGP5_70"
> cmd = with(samples, paste("tophat2 -G", gf, "-p 5 -o",
                            LibraryName, bowind, fastq1, fastq2))
> cmd
```

R output:

```
tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o Untreated-3 Dme1_BDGP5_70 \
SRR031714_1.fastq,SRR031715_1.fastq SRR031714_2.fastq,SRR031715_2.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o Untreated-4 Dme1_BDGP5_70 \
SRR031716_1.fastq,SRR031717_1.fastq SRR031716_2.fastq,SRR031717_2.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o CG8144_RNAi-3 Dme1_BDGP5_70 \
SRR031724_1.fastq,SRR031725_1.fastq SRR031724_2.fastq,SRR031725_2.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o CG8144_RNAi-4 Dme1_BDGP5_70 \
SRR031726_1.fastq,SRR031727_1.fastq SRR031726_2.fastq,SRR031727_2.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o Untreated-1 Dme1_BDGP5_70 \
SRR031708.fastq,SRR031709.fastq,SRR031710.fastq,SRR031711.fastq,SRR031712.fastq,
SRR031713.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o CG8144_RNAi-1 Dme1_BDGP5_70 \
SRR031718.fastq,SRR031719.fastq,SRR031720.fastq,SRR031721.fastq,SRR031722.fastq,
SRR031723.fastq

tophat2 -G Drosophila_melanogaster.BDGP5.70.gtf -p 5 -o Untreated-6 Dme1_BDGP5_70 \
SRR031728.fastq,SRR031729.fastq
```

▲ **CRITICAL STEP** In the call to tophat2, the option -G points tophat2 to a GTF file of annotation to facilitate mapping reads across exon-exon junctions (some of which can be found de novo), -o specifies the output directory, -p specifies the number of threads to use (this may affect run times and can vary depending on the resources available). Other parameters can be specified here, as needed; see the appropriate documentation for the tool and version you are using. The first argument, Dmel_BDGP5_70 is the name of the index (built in advance), and the second argument is a list of all FASTQ files with reads for the sample. Note that the FASTQ files are concatenated with commas, without spaces. For experiments with paired-end reads, pairs of FASTQ files are given as separate arguments and the order in both arguments must match.
? TROUBLESHOOTING

8| Run these commands (i.e., copy and paste) in a Unix terminal.

▲ **CRITICAL STEP** Many similar possibilities exist for this step (see 'Experimental design' for further details). Users can use the R function *system* to execute these commands direct from R, cut-and-paste the commands into a separate Unix shell or

store the list of commands in a text file and use the Unix 'source' command. In addition, users could construct the Unix commands independently of R.
? TROUBLESHOOTING

Organize, sort and index the BAM files and create SAM files ● TIMING ~1 h

9| Organize the BAM files into a single directory, sort and index them and create SAM files by running the following R-generated commands:

```
> for(i in seq_len(nrow(samples))) {
  lib = samples$LibraryName[i]
  ob = file.path(lib, "accepted_hits.bam")

  # sort by name, convert to SAM for htseq-count
  cat(paste0("samtools sort -n ",ob," ",lib,"_sn"),"\n")
  cat(paste0("samtools view -o ",lib,"_sn.sam ",lib,"_sn.bam"),"\n")

  # sort by position and index for IGV
  cat(paste0("samtools sort ",ob," ",lib,"_s"),"\n")
  cat(paste0("samtools index ",lib,"_s.bam"),"\n\n")
}
```

R output:

```
samtools sort -n Untreated-3/accepted_hits.bam Untreated-3_sn
samtools view -o Untreated-3_sn.sam Untreated-3_sn.bam
samtools sort Untreated-3/accepted_hits.bam Untreated-3_s
samtools index Untreated-3_s.bam

samtools sort -n Untreated-4/accepted_hits.bam Untreated-4_sn
samtools view -o Untreated-4_sn.sam Untreated-4_sn.bam
samtools sort Untreated-4/accepted_hits.bam Untreated-4_s
samtools index Untreated-4_s.bam

samtools sort -n CG8144_RNAi-3/accepted_hits.bam CG8144_RNAi-3_sn
samtools view -o CG8144_RNAi-3_sn.sam CG8144_RNAi-3_sn.bam
samtools sort CG8144_RNAi-3/accepted_hits.bam CG8144_RNAi-3_s
samtools index CG8144_RNAi-3_s.bam

samtools sort -n CG8144_RNAi-4/accepted_hits.bam CG8144_RNAi-4_sn
samtools view -o CG8144_RNAi-4_sn.sam CG8144_RNAi-4_sn.bam
samtools sort CG8144_RNAi-4/accepted_hits.bam CG8144_RNAi-4_s
samtools index CG8144_RNAi-4_s.bam

samtools sort -n Untreated-1/accepted_hits.bam Untreated-1_sn
samtools view -o Untreated-1_sn.sam Untreated-1_sn.bam
samtools sort Untreated-1/accepted_hits.bam Untreated-1_s
samtools index Untreated-1_s.bam

samtools sort -n CG8144_RNAi-1/accepted_hits.bam CG8144_RNAi-1_sn
samtools view -o CG8144_RNAi-1_sn.sam CG8144_RNAi-1_sn.bam
samtools sort CG8144_RNAi-1/accepted_hits.bam CG8144_RNAi-1_s
samtools index CG8144_RNAi-1_s.bam
```

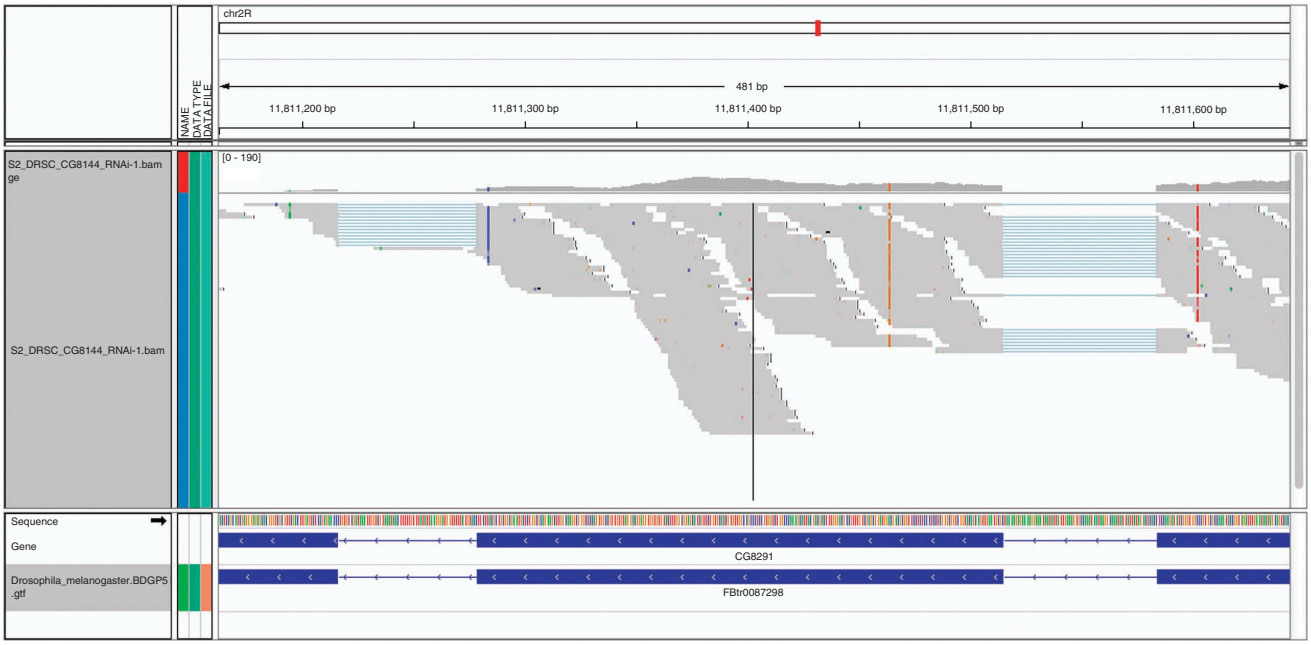


Figure 3 | Screenshot of reads aligning across exon junctions.

```
samtools sort -n Untreated-6/accepted_hits.bam Untreated-6_sn
samtools view -o Untreated-6_sn.sam Untreated-6_sn.bam
samtools sort Untreated-6/accepted_hits.bam Untreated-6_s
samtools index Untreated-6_s.bam
```

▲ **CRITICAL STEP** Users should be conscious of the disk space that may get used in these operations. In the command above, sorted-by-name SAM and BAM files (for htseq-count), as well as a sorted-by-chromosome-position BAM file (for IGV), are created for each original accepted hits.bam file. User may wish to delete (some of) these intermediate files after the steps below.

Inspect alignments with IGV ● **TIMING** **<20 min**

10| Start IGV, select the *D. melanogaster* (dm3) genome, and then load the BAM files (with s in the filename) as well as the GTF file.

11| Zoom in on an expressed transcript until individual reads are shown and check whether the reads align at and across exon-exon junctions, as expected, given the annotation (**Fig. 3**).

12| If any positive and negative controls are known for the system under study (e.g., known differential expression), direct the IGV browser to these regions to confirm that the relative read density is different according to expectation.

Count reads using htseq-count ● **TIMING** **~3 h**

13| Add the names of the COUNT files to the metadata table and call HTSeq from the following R-generated Unix commands:

```
> samples$countf = paste(samples$LibraryName, "count", sep=".")

> gf = "Drosophila_melanogaster.BDGP5.70.gtf"
> cmd = paste0("htseq-count -s no -a 10 ", samples$LibraryName,
              "_sn.sam ", gf," > ", samples$countf)

> cmd
```

R output:

```
htseq-count -s no -a 10 Untreated-3_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > Untreated-3.count
```

```
htseq-count -s no -a 10 Untreated-4_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > Untreated-4.count

htseq-count -s no -a 10 CG8144_RNAi-3_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > CG8144_RNAi-3.count

htseq-count -s no -a 10 CG8144_RNAi-4_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > CG8144_RNAi-4.count

htseq-count -s no -a 10 Untreated-1_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > Untreated-1.count

htseq-count -s no -a 10 CG8144_RNAi-1_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > CG8144_RNAi-1.count

htseq-count -s no -a 10 Untreated-6_sn.sam \
Drosophila_melanogaster.BDGP5.70.gtf > Untreated-6.count
```

▲ **CRITICAL STEP** The option -s signifies that the data are not from a stranded protocol (this may vary by experiment) and the -a option specifies a minimum score for the alignment quality.
? TROUBLESHOOTING

14| For differential expression analysis with edgeR, follow option A for simple designs and option B for complex designs; for differential expression analysis with DESeq, follow option C for simple designs and option D for complex designs.
(A) edgeR—simple design

(i) Load the edgeR package and use the utility function, *readDGE*, to read in the COUNT files created from htseq-count:

```
> library("edgeR")
> counts = readDGE(samples$countf)$counts
```

? TROUBLESHOOTING

(ii) Filter weakly expressed and noninformative (e.g., non-aligned) features using a command like:

```
> noint = rownames(counts) %in%
              c("no_feature","ambiguous","too_low_aQual",
                "not_aligned","alignment_not_unique")
> cpms = cpm(counts)
> keep = rowSums(cpms >1) >=3 & !noint
> counts = counts[keep,]
```

▲ **CRITICAL STEP** In edgeR, it is recommended to remove features without at least 1 read per million in *n* of the samples, where *n* is the size of the smallest group of replicates (here, *n* = 3 for the knockdown group).

(iii) Visualize and inspect the count table as follows:

```
> colnames(counts) = samples$shortname
> head( counts[,order(samples$condition)], 5 )
```

R output:

	CT.PA.1	CT.PA.2	CT.SI.5	CT.SI.7	KD.PA.3	KD.PA.4	KD.SI.6
FBgn0000008	76	71	137	82	87	68	115
FBgn0000017	3498	3087	7014	3926	3029	3264	4322
FBgn0000018	240	306	613	485	288	307	528
FBgn0000032	611	672	1479	1351	694	757	1361
FBgn0000042	40048	49144	97565	99372	70574	72850	95760

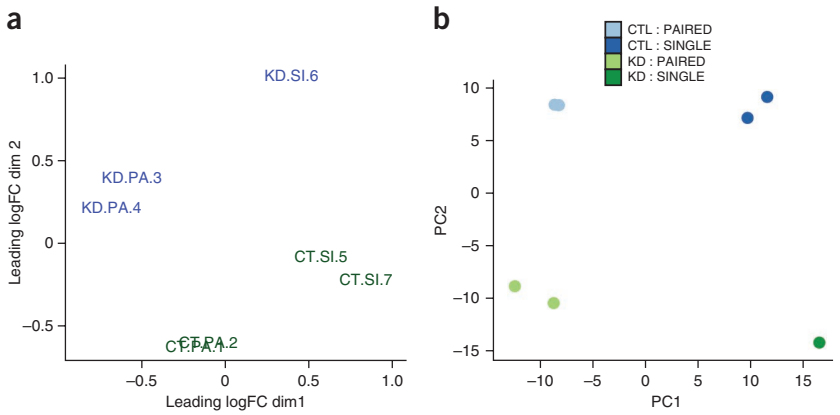


Figure 4 | Plots of sample relations. (**a**) By using a count-specific distance measure, edgeR's *plotMDS* produces a multidimensional scaling plot showing the relationship between all pairs of samples. (**b**) DESeq's *plotPCA* makes a principal component (PC) plot of VST (variance-stabilizing transformation)-transformed count data. CT or CTL, control; KD, knockdown.

(iv) Create a *DGEList* object (edgeR's container for RNA-seq count data), as follows:

```
> d = DGEList(counts=counts, group=samples$condition)
```

(v) Estimate normalization factors using:

```
> d = calcNormFactors(d)
```

(vi) Inspect the relationships between samples using a multidimensional scaling (MDS) plot, as shown in **Figure 4**:

```
> plotMDS(d, labels=samples$shortname,
          col=c("darkgreen","blue")[factor(samples$condition)])
```

(vii) Estimate tagwise dispersion (simple design) using:

```
> d = estimateCommonDisp(d)
> d = estimateTagwiseDisp(d)
```

(viii) Create a visual representation of the mean-variance relationship using the *plotMeanVar* (**Fig. 5a**) and *plotBCV* (**Fig. 5b**) functions, as follows:

```
> plotMeanVar(d, show.tagwise.vars=TRUE, NBline=TRUE)
> plotBCV(d)
```

(ix) Test for differential expression ('classic' edgeR), as follows:

```
> de = exactTest(d, pair=c("CTL","KD"))
```

(x) Follow Step 14B(vi–ix).

(B) edgeR—complex design

(i) Follow Step 14A(i–vi).

(ii) Create a design matrix (see 'Experimental design' for further details) to specify the factors that are expected to affect expression levels:

```
> design = model.matrix( ~ LibraryLayout + condition, samples)
> design
```

R output:

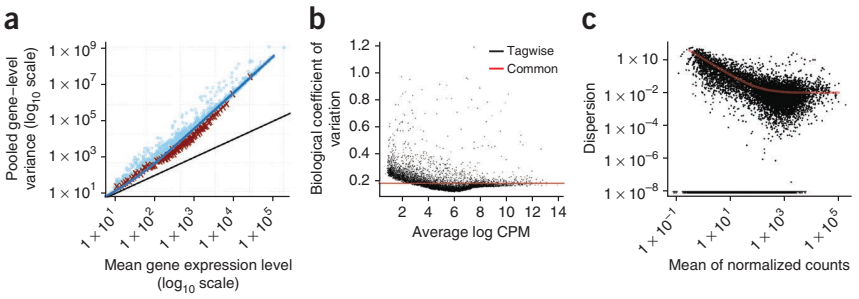


Figure 5 | Plots of mean-variance relationship and dispersion. (**a**) edgeR's *plotMeanVar* can be used to explore the mean-variance relationship; each dot represents the estimated mean and variance for each gene, with binned variances as well as the trended common dispersion overlaid. (**b**) edgeR's *plotBCV* illustrates the relationship of biological coefficient of variation versus mean log CPM. (**c**) DESeq's *plotDispEsts* shows the fit of dispersion versus mean. CPM, counts per million.

```
   (Intercept)  LibraryLayoutSINGLE  conditionKD
1  1            0                    0
2  1            0                    0
3  1            0                    1
4  1            0                    1
5  1            1                    0
6  1            1                    1
7  1            1                    0
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$LibraryLayout
[1] "contr.treatment"
attr(,"contrasts")$condition
[1] "contr.treatment"
```

(iii) Estimate dispersion values, relative to the design matrix, using the Cox-Reid (CR)-adjusted likelihood[7,53], as follows:

```
> d2 = estimateGLMTrendedDisp(d, design)
> d2 = estimateGLMTagwiseDisp(d2, design)
```

(iv) Given the design matrix and dispersion estimates, fit a GLM to each feature:

```
> f = glmFit(d2, design)
```

(v) Perform a likelihood ratio test, specifying the difference of interest (here, knockdown versus control, which corresponds to the third column of the above design matrix):

```
> de = glmLRT(f, coef=3)
```

(vi) Use the *topTags* function to present a tabular summary of the differential expression statistics (note that *topTags* operates on the output of *exactTest* or *glmLRT*, but only the latter is shown here):

```
> tt = topTags(de, n=nrow(d))
> head(tt$table)
```

R output:

```
             logFC   logCPM  LR    PValue      FDR
FBgn0039155  -4.61   5.87    902   3.96e-198   2.85e-194
FBgn0025111  2.87    6.86    641   2.17e-141   7.81e-138
FBgn0039827  -4.05   4.40    457   2.11e-101   5.07e-98
FBgn0035085  -2.58   5.59    408   9.31e-91    1.68e-87
FBgn0000071  2.65    4.73    365   2.46e-81    3.54e-78
FBgn0003360  -3.12   8.42    359   3.62e-80    4.34e-77
```

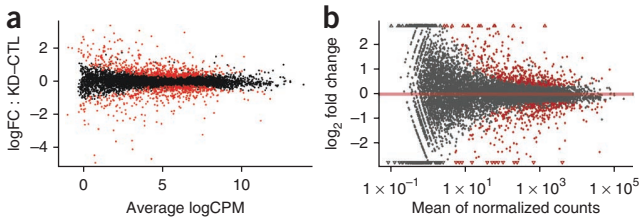


Figure 6 | M ('minus') versus A ('add') plots for RNA-seq data. (**a**) edgeR's *plotSmear* function plots the log-fold change (i.e., the log ratio of normalized expression levels between two experimental conditions) against the log counts per million (CPM). (**b**) Similarly, DESeq's *plotMA* displays differential expression (log-fold changes) versus expression strength (log average read count).

(vii) Inspect the depth-adjusted reads per million for some of the top differentially expressed genes:

```
> nc = cpm(d, normalized.lib.sizes=TRUE)
> rn = rownames(tt$table)
> head(nc[rn,order(samples$condition)],5)
```

R output:

	CT.PA.1	CT.PA.2	CT.SI.5	CT.SI.7	KD.PA.3	KD.PA.4	KD.SI.6
FBgn0039155	91.07	98.0	100.75	106.78	3.73	4.96	3.52
FBgn0025111	34.24	31.6	26.64	28.46	247.43	254.28	188.39
FBgn0039827	39.40	36.7	30.09	34.47	1.66	2.77	2.01
FBgn0035085	78.06	81.4	63.59	74.08	13.49	14.13	10.99
FBgn0000071	9.08	9.2	7.48	5.85	52.08	55.93	45.65

(viii) Create a graphical summary, such as an M (log-fold change) versus A (log-average expression) plot[54], here showing the genes selected as differentially expressed (with a 5% false discovery rate; **Fig. 6**):

```
> deg = rn[tt$table$FDR < .05]
> plotSmear(d, de.tags=deg)
```

(ix) Save the result table as a CSV file (alternative formats are possible) as follows:

```
> write.csv(tt$table, file="toptags_edgeR.csv")
```

(C) DESeq—simple design

(i) Create a *data.frame* with the required metadata, i.e., the names of the count files and experimental conditions. Here we derive it from the samples table created in Step 3.

```
> samplesDESeq = with(samples,
                      data.frame(shortname = I(shortname), countf = I(countf),
                                 condition    = condition,
                                 LibraryLayout = LibraryLayout))
```

(ii) Load the DESeq package and create a *CountDataSet* object (DESeq's container for RNA-seq data) from the count tables and corresponding metadata:

```
> library("DESeq")
> cds = newCountDataSetFromHTSeqCount(samplesDESeq)
```

? TROUBLESHOOTING

(iii) Estimate normalization factors using:

```
> cds = estimateSizeFactors(cds)
```

(iv) Inspect the size factors using:

```
> sizeFactors(cds)
```

R output:

```
CT.PA.1 CT.PA.2 KD.PA.3 KD.PA.4 CT.SI.5 KD.SI.6 CT.SI.7
  0.699   0.811   0.822   0.894   1.643   1.372   1.104
```

(v) To inspect sample relationships, invoke a variance-stabilizing transformation and inspect a principal component analysis (PCA) plot (**Fig. 4b**):

```
> cdsB = estimateDispersions(cds, method="blind")
> vsd = varianceStabilizingTransformation(cdsB)
> p = plotPCA(vsd, intgroup=c("condition","LibraryLayout"))
```

(vi) Use *estimateDispersions* to calculate dispersion values:

```
> cds = estimateDispersions(cds)
```

(vii) Inspect the estimated dispersions using the *plotDispEsts* function (**Fig. 5c**), as follows:

```
> plotDispEsts(cds)
```

(viii) Perform the test for differential expression by using *nbinomTest*, as follows:

```
> res = nbinomTest(cds,"CTL","KD")
```

(ix) Given the table of differential expression results, use *plotMA* to display differential expression (log-fold changes) versus expression strength (log-average read count), as follows (**Fig. 6b**):

```
> plotMA(res)
```

(x) Inspect the result tables of significantly upregulated and downregulated genes, at a 10% false discovery rate (FDR) as follows:

```
> resSig = res[which(res$padj < 0.1),]
> head( resSig[ order(resSig$log2FoldChange, decreasing=TRUE), ] )
```

R output:

	id	baseMean	baseMeanA	baseMeanB	foldChange	log2FoldChange	pval	padj
1515	FBgn0013696	1.46	0.000	3.40	Inf	Inf	4.32e-03	6.86e-02
13260	FBgn0085822	1.93	0.152	4.29	28.2	4.82	4.54e-03	7.11e-02
13265	FBgn0085827	8.70	0.913	19.08	20.9	4.39	1.02e-09	9.57e-08
15470	FBgn0264344	3.59	0.531	7.68	14.5	3.86	4.55e-04	1.10e-02
8153	FBgn0037191	4.43	0.715	9.39	13.1	3.71	5.35e-05	1.78e-03
1507	FBgn0013688	23.82	4.230	49.95	11.8	3.56	3.91e-21	1.38e-18

```
> head( resSig[ order(resSig$log2FoldChange, decreasing=FALSE), ] )
```

R output:

	id	baseMean	baseMeanA	baseMeanB	foldChange	log2FoldChange	pval	padj
13045	FBgn0085359	60.0	102.2	3.78	0.0370	-4.76	7.65e-30	4.88e-27
9499	FBgn0039155	684.1	1161.5	47.59	0.0410	-4.61	3.05e-152	3.88e-148
2226	FBgn0024288	52.6	88.9	4.25	0.0478	-4.39	2.95e-32	2.09e-29
9967	FBgn0039827	246.2	412.0	25.08	0.0609	-4.04	1.95e-82	8.28e-79
6279	FBgn0034434	104.5	171.8	14.72	0.0856	-3.55	8.85e-42	9.40e-39
6494	FBgn0034736	203.9	334.9	29.38	0.0877	-3.51	6.00e-41	5.88e-38

(xi) Count the number of genes with significant differential expression at a FDR of 10%:

```
> table( res$padj < 0.1 )
```

R output:

```
FALSE   TRUE
11861   885
```

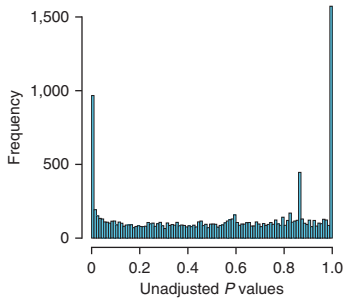


Figure 7 | Histogram of *P* values from gene-by-gene statistical tests.

(xii) Create a persistent storage of results using, for example, a CSV file:

```
> write.csv(res, file="res_DESeq.csv")
```

(xiii) Perform a sanity check by inspecting a histogram of unadjusted *P* values (**Fig. 7**) for the differential expression results, as follows:

```
> hist(res$pval, breaks=100)
```

(D) DESeq—complex design

(i) Follow Step 14C(i–v).

(ii) Calculate the CR-adjusted profile likelihood[53] dispersion estimates relative to the factors specified, developed by McCarthy *et al.*[7], according to:

```
> cds = estimateDispersions(cds, method = "pooled-CR",
                            modelFormula = count ~ LibraryLayout + condition)
```

(iii) Test for differential expression in the GLM setting by fitting both a full model and reduced model (i.e., with the factor of interest taken out):

```
> fit1 = fitNbinomGLMs(cds, count ~ LibraryLayout + condition)
> fit0 = fitNbinomGLMs(cds, count ~ LibraryLayout)
```

(iv) By using the two fitted models, compute likelihood ratio statistics and associated *P* values, as follows:

```
> pval = nbinomGLMTest(fit1, fit0)
```

(v) Adjust the reported *P* values for multiple testing:

```
> padj = p.adjust(pval, method="BH")
```

(vi) Assemble a result table from full model fit and the raw and adjusted *P* values and print the first few upregulated and downregulated genes (FDR <10%):

```
> res = cbind(fit1, pval=pval, padj=padj)
> resSig = res[which(res$padj < 0.1),]
> head( resSig[ order(resSig$conditionKD, decreasing=TRUE), ] )
```

R output:

	(Intercept)	LibraryLayoutSINGLE	conditionKD	deviance	converged	pval	padj
FBgn0013696	-70.96	36.829	37.48	5.79e-10	TRUE	3.52e-03	5.51e-02
FBgn0085822	-5.95	4.382	5.14	2.07e+00	TRUE	4.72e-03	7.07e-02
FBgn0085827	-3.96	4.779	5.08	2.89e+00	TRUE	5.60e-03	8.05e-02
FBgn0264344	-2.59	2.506	4.26	6.13e-01	TRUE	3.86e-04	9.17e-03
FBgn0261673	3.53	0.133	3.37	1.39e+00	TRUE	0.00e+00	0.00e+00
FBgn0033065	2.85	-0.421	3.03	4.07e+00	TRUE	8.66e-15	1.53e-12

```
> head( resSig[ order(resSig$conditionKD, decreasing=FALSE), ] )
```

R output:

	(Intercept)	LibraryLayoutSINGLE	conditionKD	deviance	converged	pval	padj
FBgn0031923	1.01	1.2985	-32.26	1.30	TRUE	0.00528	0.077
FBgn0085359	6.37	0.5782	-4.62	3.32	TRUE	0.00000	0.000
FBgn0039155	10.16	0.0348	-4.62	3.39	TRUE	0.00000	0.000
FBgn0024288	6.71	-0.4840	-4.55	1.98	TRUE	0.00000	0.000
FBgn0039827	8.79	-0.2272	-4.06	2.87	TRUE	0.00000	0.000
FBgn0034736	8.54	-0.3123	-3.57	2.09	TRUE	0.00000	0.000

(vii) Follow Step 14C(xi–xiii).

15| As another spot check, point the IGV genome browser (with GTF and BAM files loaded) to a handful of the top differentially expressed genes and confirm that the counting and differential expression statistics are appropriately represented.

? TROUBLESHOOTING

Troubleshooting advice can be found in **Table 1**.

TABLE 1 | Troubleshooting table.

Step	Problem	Possible reason	Solution
1, 14A(i), 14C(ii)	An error occurs when loading a Bioconductor package	Version mismatch	Make sure the most recent version of R is installed; reinstall packages using *biocLite*
7, 8	An error occurs while mapping reads to reference genome	Wrong files made available or version mismatch	Carefully check the command submitted, the documentation for the aligner and the setup steps (e.g., building an index); check that there is no clash between bowtie and bowtie2
13	An error occurs counting features	GTF format violation	Use an Ensembl GTF format or coerce your file into a compatible format. In particular, verify that each line of type exon contains attributes named gene_id and transcript_id, and ensure that their values are correct
14	Errors in fitting statistical models or running statistical tests	Wrong inputs, outdated version of software	Ensure versions of R and Bioconductor packages are up to date and check the command issued; if a command is correct but an error persists, post a message to the Bioconductor mailing list (http://bioconductor.org/help/mailing-list/) according to the posting guide (http://bioconductor.org/help/mailing-list/posting-guide/)

● TIMING

Running this protocol on the SRA-downloaded data will take <10 h on a machine with eight cores and 8 GB of RAM; with a machine with more cores, mapping of different samples can be run simultaneously. The time is largely spent on quality checks of reads, read alignment and feature counting; computation time for the differential expression analysis is comparatively smaller.

Step 1, sequence quality checks: ~2 h
Steps 3–6, organizing metadata: <1 h
Steps 7 and 8, read alignment: ~45 min per sample
Step 9, organize, sort and index the BAM files and create SAM files: ~1 h
Steps 10–12, inspect alignments with IGV: <20 min
Step 13, feature counting: ~3 h
Step 14, differential analysis: variable; computational time is often <20 min
Step 15, additional spot checks: <20 min

ANTICIPATED RESULTS

Sequencing quality checks

Step 1 results in an HTML report for all included FASTQ files. Users should inspect these (Step 2) and look for persistence of low-quality scores, over-representation of adapter sequence and other potential problems. From these inspections, users may choose to remove low-quality samples, trim ends of reads (e.g., using FASTX; http://hannonlab.cshl.edu/fastx_toolkit/) or modify alignment parameters. Note that a popular non-Bioconductor alternative for sequencing quality checks is FastQC (http://www.bioinformatics.babraham.ac.uk/projects/fastqc/).

Feature counting

In Step 13, we used htseq-count for feature counting. The output is a COUNT file (two columns: identifier, count) for each sample. Many alternatives exist inside and outside of Bioconductor to arrive at a table of counts given BAM (or SAM) files and a set of features (e.g., from a GTF file); see **Box 3** for further considerations. Each cell in the count table will be an integer that indicates how many reads in the sample overlap with the respective feature. Non-informative rows, such as features that are not of interest or those that have low overall counts, can be filtered. We recommend removing rows with a low overall sum of counts (or low CPM), as this generally increases the statistical power of the differential expression analysis[55].

Normalization

As different libraries will be sequenced to different depths, offsets are built in the statistical model to ensure that parameters are comparable. The term normalization is often used for that, but it should be noted that the raw read counts are not actually altered[56]. By default, edgeR uses the number of mapped reads (i.e., count table column sums) and estimates an additional normalization factor to account for sample-specific effects (e.g., diversity)[56]; these two factors are combined and used as an offset in the NB model. Analogously, DESeq defines a virtual reference sample by taking the median of each gene's values across samples and then computes size factors as the median of ratios of each sample to the reference sample. Generally, the ratios of the size factors should roughly match the ratios of the library sizes. Dividing each column of the count table by the corresponding size factor yields normalized count values, which can be scaled to give a counts per million interpretation (see also edgeR's *cpm* function). From an M (log ratio) versus A (log expression strength) plot, count data sets typically show a (left-facing) trombone shape, reflecting the higher variability of log ratios at lower counts (**Fig. 6**). In addition, points will typically be centered around a log ratio of 0 if the normalization factors are calculated appropriately, although this is just a general guide.

Sample relations

The quality of the sequencing reactions (Step 1) themselves is only a part of the quality assessment procedure. In Steps 14A(vi) or 14C(v), a 'fitness for use'[57] check is performed (relative to the biological question of interest) on the count data before statistical modeling. edgeR adopts a straightforward approach that compares the relationship between all pairs of samples, using a count-specific pairwise distance measure (i.e., biological coefficient of variation) and an MDS plot for visualization (**Fig. 4a**). Analogously, DESeq performs a variance-stabilizing transformation and explores sample relationships using a PCA plot (**Fig. 4b**). In either case, the analysis for the current data set highlights that library type (single-end or paired-end) has a systematic effect on the read counts and provides an example of a data-driven modeling decision: here, a GLM-based analysis that accounts for the (assumed linear) effect of library type jointly with the biological factor of interest (i.e., knockdown versus control) is recommended. In general, users should be conscious that the degree of variability between the biological replicates (e.g., in an MDS or PCA plot) will ultimately effect the calling of differential expression. For example, a single outlying sample may drive increased dispersion estimates and compromise the discovery of differentially expressed features. No general prescription is available for when and whether to delete outlying samples.

Dispersion estimation

As mentioned above, getting good estimates of the dispersion parameter is critical to the inference of differential expression. For simple designs, edgeR uses the quantile-adjusted conditional maximum (weighted) likelihood estimator[4,5], whereas DESeq uses a method-of-moments estimator[3]. For complex designs, the dispersion estimates are made relative to the design matrix, using the CR-adjusted likelihood[7,53]; both DESeq and edgeR use this estimator. edgeR's estimates are always moderated toward a common trend, whereas DESeq chooses the maximum of the individual estimate and a smooth fit (dispersion versus mean) over all genes. A wide range of dispersion-mean relationships exist in RNA-seq data, as viewed by edgeR's *plotBCV* or DESeq's *plotDispEsts*; case studies with further details are presented in both edgeR's and DESeq's user guides.

Differential expression analysis

DESeq and edgeR differ slightly in the format of results outputted, but each contains columns for log-fold change (log), counts per million (or mean by condition), likelihood ratio statistic (for GLM-based analyses), as well as raw and adjusted

P values. By default, *P* values are adjusted for multiple testing using the Benjamini-Hochberg[58] procedure. If users enter tabular information to accompany the set of features (e.g., annotation information), edgeR has a facility to carry feature-level information into the results table.

Post-differential analysis sanity checks

Figure 7 (Step 14C(xiii)) shows the typical features of a *P* value histogram resulting from a good data set: a sharp peak at the left side, containing genes with strong differential expression, a 'floor' of values that are approximately uniform in the interval [0, 1], corresponding to genes that are not differentially expressed (for which the null hypothesis is true), and a peak at the upper end at 1, resulting from discreteness of the NB test for genes with overall low counts. The latter component is often less pronounced, or even absent, when the likelihood ratio test is used. In addition, users should spot check genes called as differentially expressed by loading the sorted BAM files into a genome browser.

Note: Supplementary information is available in the online version of the paper.

ACKNOWLEDGMENTS We thank X. Zhou for comparing counting methods, O. Nikolayeva for feedback on an earlier version of the manuscript and participants in the European Conference on Computational Biology Workshop (Basel, September 2012) for their feedback. G.K.S. acknowledges funding from a National Health and Medical Research Council Project Grant (no. 1023454). D.J.M. acknowledges funding from the General Sir John Monash Foundation, Australia. M.D.R. wishes to acknowledge funding from the University of Zurich's Research Priority Program in Systems Biology and Functional Genomics and Swiss National Science Foundation Project grant (no. 143883). S.A., W.H. and M.D.R. acknowledge funding from the European Commission through the 7th Framework Collaborative Project RADIANT (grant agreement no. 305626).

AUTHOR CONTRIBUTIONS S.A. and W.H. are authors of the DESeq package. D.J.M., Y.C., G.K.S. and M.D.R. are authors of the edgeR package. S.A., M.O., W.H. and M.D.R. initiated the protocol format on the basis of the ECCB 2012 Workshop. S.A. and M.D.R. wrote the first draft and additions were made from all authors.

COMPETING FINANCIAL INTERESTS The authors declare no competing financial interests.

1. Mortazavi, A., Williams, B.A., McCue, K., Schaeffer, L. & Wold, B. Mapping and quantifying mammalian transcriptomes by RNA-seq. *Nat. Methods* **5**, 621–628 (2008).
2. Wang, Z., Gerstein, M. & Snyder, M. RNA-seq: a revolutionary tool for transcriptomics. *Nat. Rev. Genet.* **10**, 57–63 (2009).
3. Anders, S. & Huber, W. Differential expression analysis for sequence count data. *Genome Biol.* **11**, R106 (2010).
4. Robinson, M.D. & Smyth, G.K. Moderated statistical tests for assessing differences in tag abundance. *Bioinformatics* **23**, 2881–2887 (2007).
5. Robinson, M.D. & Smyth, G.K. Small-sample estimation of negative binomial dispersion, with applications to SAGE data. *Biostatistics* **9**, 321–332 (2008).
6. Robinson, M.D., McCarthy, D.J. & Smyth, G.K. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* **26**, 139–140 (2010).
7. McCarthy, D.J., Chen, Y. & Smyth, G.K. Differential expression analysis of multifactor RNA-seq experiments with respect to biological variation. *Nucleic Acids Res.* **40**, 4288–4297 (2012).
8. Gentleman, R.C. *et al.* Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.* **5**, R80 (2004).
9. Zemach, A. *et al.* The *Arabidopsis* nucleosome remodeler DDM1 allows DNA methyltransferases to access H1-containing heterochromatin. *Cell* **153**, 193–205 (2013).
10. Lam, M.T. *et al.* Rev-Erbs repress macrophage gene expression by inhibiting enhancer-directed transcription. *Nature* **498**, 511–515 (2013).
11. Ross-Innes, C.S. *et al.* Differential oestrogen receptor binding is associated with clinical outcome in breast cancer. *Nature* **481**, 389–393 (2012).
12. Robinson, M.D. *et al.* Copy-number-aware differential analysis of quantitative DNA sequencing data. *Genome Res.* **22**, 2489–2496 (2012).
13. Vanharanta, S. *et al.* Epigenetic expansion of VHL-HIF signal output drives multiorgan metastasis in renal cancer. *Nat. Med.* **19**, 50–56 (2013).
14. Samstein, R.M. *et al.* Foxp3 exploits a pre-existent enhancer landscape for regulatory T cell lineage specification. *Cell* **151**, 153–166 (2012).
15. Johnson, E.K. *et al.* Proteomic analysis reveals new cardiac-specific dystrophin-associated proteins. *PloS ONE* **7**, e43515 (2012).
16. Fonseca, N.A., Rung, J., Brazma, A. & Marioni, J.C. Tools for mapping high-throughput sequencing data. *Bioinformatics* **28**, 3169–3177 (2012).
17. Trapnell, C. *et al.* Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nat. Protoc.* **7**, 562–578 (2012).
18. Bullard, J.H., Purdom, E., Hansen, K.D. & Dudoit, S. Evaluation of statistical methods for normalization and differential expression in mRNA-seq experiments. *BMC Bioinform.* **11**, 94 (2010).
19. Grabherr, M.G. *et al.* Full-length transcriptome assembly from RNA-seq data without a reference genome. *Nat. Biotechnol.* **29**, 644–652 (2011).
20. Siebert, S. *et al.* Differential gene expression in the siphonophore *Nanomia bijuga* (Cnidaria) assessed with multiple next-generation sequencing workflows. *PLoS ONE* **6**, 12 (2011).
21. Trapnell, C., Pachter, L. & Salzberg, S.L. TopHat: discovering splice junctions with RNA-seq. *Bioinformatics* **25**, 1105–1111 (2009).
22. Trapnell, C. *et al.* Transcript assembly and quantification by RNA-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat. Biotechnol.* **28**, 511–515 (2010).
23. Hardcastle, T.J. & Kelly, K.A. baySeq: empirical Bayesian methods for identifying differential expression in sequence count data. *BMC Bioinform.* **11**, 422 (2010).
24. Zhou, Y.-H., Xia, K. & Wright, F.A. A powerful and flexible approach to the analysis of RNA sequence count data. *Bioinformatics* **27**, 2672–2678 (2011).
25. Tarazona, S., Garcia-Alcalde, F., Dopazo, J., Ferrer, A. & Conesa, A. Differential expression in RNA-seq: a matter of depth. *Genome Res.* **21**, 2213–2223 (2011).
26. Lund, S.P., Nettleton, D., McCarthy, D.J. & Smyth, G.K. Detecting differential expression in RNA-sequence data using quasi-likelihood with shrunken dispersion estimates. *Stat. Appl. Genet. Mol. Biol.* **11**, pii (2012).
27. Soneson, C. & Delorenzi, M. A comparison of methods for differential expression analysis of RNA-seq data. *BMC Bioinform.* **14**, 91 (2013).
28. Lareau, L.F., Inada, M., Green, R.E., Wengrod, J.C. & Brenner, S.E. Unproductive splicing of SR genes associated with highly conserved and ultraconserved DNA elements. *Nature* **446**, 926–929 (2007).
29. Anders, S., Reyes, A. & Huber, W. Detecting differential usage of exons from RNA-seq data. *Genome Res.* **22**, 2008–2017 (2012).
30. Glaus, P., Honkela, A. & Rattray, M. Identifying differentially expressed transcripts from RNA-seq data with biological variation. *Bioinformatics* **28**, 1721–1728 (2012).
31. Van De Wiel, M.A. *et al.* Bayesian analysis of RNA sequencing data by estimating multiple shrinkage priors. *Biostatistics* **14**, 113–128 (2013).

32. Blekhman, R., Marioni, J.C., Zumbo, P., Stephens, M. & Gilad, Y. Sex-specific and lineage-specific alternative splicing in primates. *Genome Res.* **20**, 180–189 (2010).
33. Okoniewski, M.J. *et al.* Preferred analysis methods for single genomic regions in RNA sequencing revealed by processing the shape of coverage. *Nucleic Acids Res.* **40**, e63 (2012).
34. Hansen, K.D., Wu, Z., Irizarry, R.A. & Leek, J.T. Sequencing technology does not eliminate biological variability. *Nat. Biotechnol.* **29**, 572–573 (2011).
35. Leek, J.T. *et al.* Tackling the widespread and critical impact of batch effects in high-throughput data. *Nat. Rev. Genet.* **11**, 733–739 (2010).
36. Auer, P.L. & Doerge, R.W. Statistical design and analysis of RNA sequencing data. *Genetics* **185**, 405–416 (2010).
37. Gagnon-Bartsch, J.A. & Speed, T.P. Using control genes to correct for unwanted variation in microarray data. *Biostatistics* **13**, 539–552 (2011).
38. Leek, J.T. & Storey, J.D. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genet.* **3**, 1724–1735 (2007).
39. Myers, R.M. *Classical and Modern Regression with Applications* 2nd edn. (Duxbury Classic Series, 2000).
40. Gentleman, R. Reproducible research: a bioinformatics case study. *Stat. Appl. Genet. Mol. Biol.* **4**, Article2 (2005).
41. Trapnell, C. & Salzberg, S.L. How to map billions of short reads onto genomes. *Nat. Biotechnol.* **27**, 455–457 (2009).
42. Wu, T.D. & Nacu, S. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics* **26**, 873–881 (2010).
43. Wang, K. *et al.* MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Res.* **38**, e178 (2010).
44. Liao, Y., Smyth, G.K. & Shi, W. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Res.* **41**, e108 (2013).
45. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29**, 15–21 (2013).
46. Thorvaldsdóttir, H., Robinson, J.T. & Mesirov, J.P. Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief. Bioinform.* **14**, 178–192 (2013).
47. Fiume, M., Williams, V., Brook, A. & Brudno, M. Savant: genome browser for high-throughput sequencing data. *Bioinformatics* **26**, 1938–1944 (2010).
48. Fiume, M. *et al.* Savant genome browser 2: visualization and analysis for population-scale genomics. *Nucleic Acids Res.* **40**, 1–7 (2012).
49. Morgan, M. *et al.* ShortRead: a Bioconductor package for input, quality assessment and exploration of high-throughput sequence data. *Bioinformatics* **25**, 2607–2608 (2009).
50. Li, H. *et al.* The sequence alignment/map format and SAMtools. *Bioinformatics* **25**, 2078–2079 (2009).
51. Brooks, A.N. *et al.* Conservation of an RNA regulatory map between *Drosophila* and mammals. *Genome Res.* **21**, 193–202 (2011).
52. Edgar, R., Domrachev, M. & Lash, A.E. Gene expression omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.* **30**, 207–210 (2002).
53. Cox, D.R. & Reid, N. Parameter orthogonality and approximate conditional inference. *J. Roy. Stat. Soc. Ser. B Method.* **49**, 1–39 (1987).
54. Dudoit, S., Yang, Y.H., Callow, M.J. & Speed, T.P. Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Stat. Sinica* **12**, 111–139 (2002).
55. Bourgon, R., Gentleman, R. & Huber, W. Independent filtering increases detection power for high-throughput experiments. *Proc. Natl. Acad. Sci. USA* **107**, 9546–9551 (2010).
56. Robinson, M.D. & Oshlack, A. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biol.* **11**, R25 (2010).
57. Cappiello, C., Francalanci, C. & Pernici, B. Data quality assessment from the user's perspective. *Architecture* **22**, 68–73 (2004).
58. Benjamini, Y. & Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. Roy. Stat. Soc. Ser. B Methodol.* **57**, 289–300 (1995).
59. Wu, H., Wang, C. & Wu, Z. A new shrinkage estimator for dispersion improves differential expression detection in RNA-seq data. *Biostatistics* **14**, 232–243 (2012).
60. Smyth, G.K. Limma: linear models for microarray data. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor* (eds. Gentleman, R. *et al.*) 397–420 (Springer, 2005).
61. Nookaew, I. *et al.* A comprehensive comparison of RNA-seq–based transcriptome analysis from reads to differential gene expression and cross-comparison with microarrays: a case study in *Saccharomyces cerevisiae*. *Nucleic Acids Res.* **40**, 10084–10097 (2012).
62. Rapaport, F. *et al.* Comprehensive evaluation of differential expression analysis methods for RNA-seq data http://arXiv.org/abs/1301.5277v2 (23 January 2013).
63. Hansen, K.D., Irizarry, R.A. & Wu, Z. Removing technical variability in RNA-seq data using conditional quantile normalization. *Biostatistics* **13**, 204–216 (2012).
64. Risso, D., Schwartz, K., Sherlock, G. & Dudoit, S. GC-content normalization for RNA-seq data. *BMC Bioinform.* **12**, 480 (2011).
65. Delhomme, N., Padioleau, I., Furlong, E.E. & Steinmetz, L. easyRNASeq: a Bioconductor package for processing RNA-seq data. *Bioinformatics* **28**, 2532–2533 (2012).
66. Leisch, F. Sweave: dynamic generation of statistical reports using literate data analysis. In *Compstat 2002 Proceedings in Computational Statistics* Vol. 69 (eds. Härdle, W. & Rönz, B.) 575–580. Institut für Statistik und Wahrscheinlichkeitstheorie, Technische Universität Wien (Physica Verlag, 2002).