Constructing and maintaining reproducible bioinformatics pipelines in your research

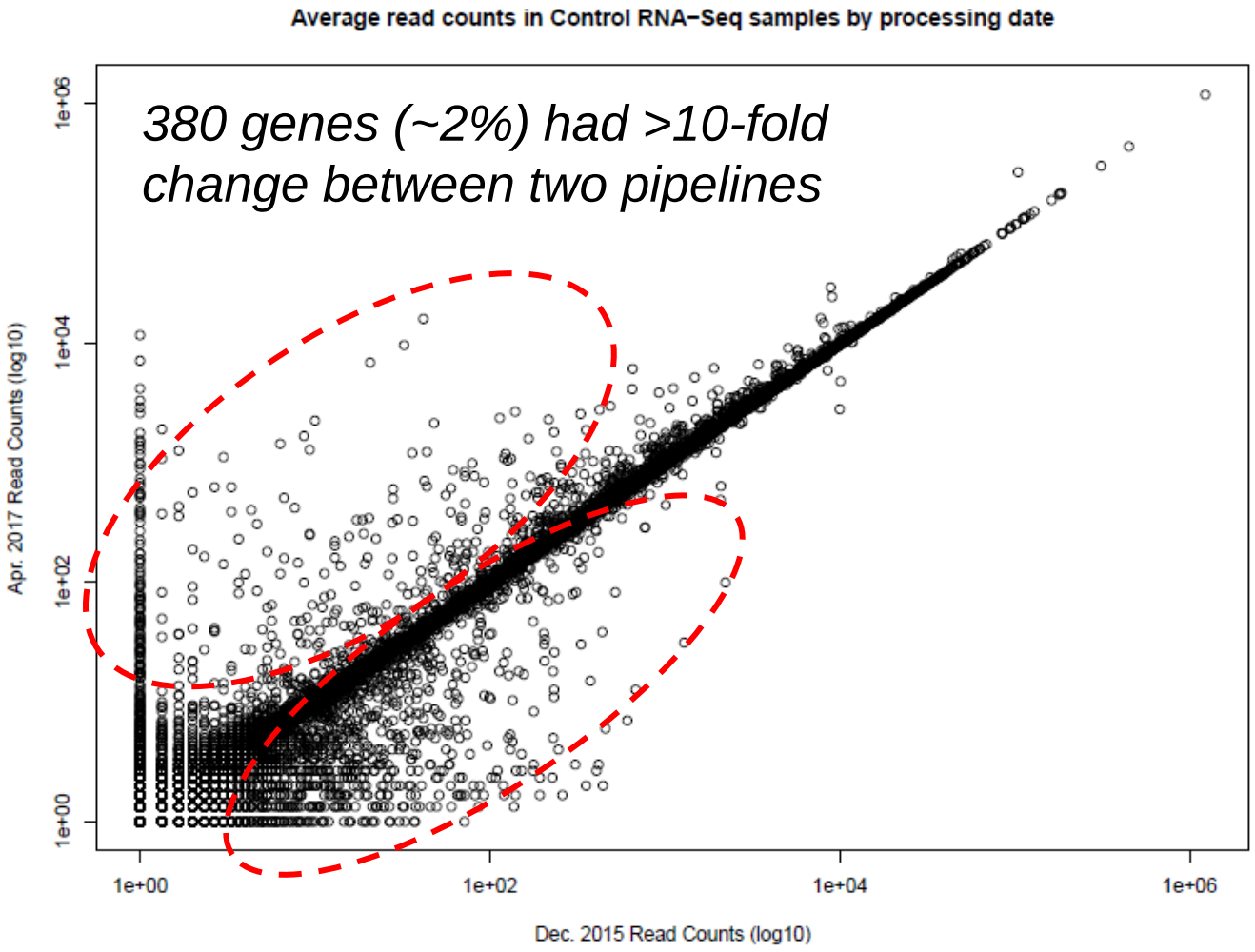
Matthew Jensen
Girirajan lab, Bioinformatics and Genomics program
2020 Data Reproducibility Bootcamp
August 13, 2020

How to construct and maintain bioinformatics pipelines

- What could go wrong if your pipelines aren’t reproducible?
- Building a reproducible pipeline
- Maintaining a reproducible pipeline
- Publishing your reproducible pipelines

A word of caution regarding irreproducible pipelines

- Differentially-expressed genes from RNA-Seq data
  - *Pipeline A*: Aligned with TopHat v.2.1.0 (BowTie v.2.2.6); run separately on technical replicates
  - *Pipeline B*: Aligned with BowTie v.2.2.9; run on merged files for biological replicates



A word of caution regarding irreproducible pipelines

- Differentially-expressed genes from RNA-Seq data
  - *Pipeline A*: Aligned with TopHat v.2.1.0 (BowTie v.2.2.6); run separately on technical replicates
  - *Pipeline B*: Aligned with BowTie v.2.2.9; run on merged files for biological replicates

Differentially-expressed gene count by pipeline

	edgeR (Pipeline A)	edgeR (Pipeline B)	DeSeq2 (Pipeline A)	DeSeq2 (Pipeline B)
C16orf53	96	168	42	72
CDIPT	1087	1130	546	679
CORO1A	458	525	166	359
DOC2A	24	42	12	14
KCTD13	528	569	255	332
MAPK3	328	385	123	189
Total	2521	2819	1144	1645

-Pipeline A results could not be reproduced, even with correct software

-Pipeline B results showed increase in # of differentially-expressed genes

Building a reproducible bioinformatics pipeline

- 1: Select the right software for your analysis—*Read the literature*

  - Review several papers that may use softwares in different contexts
  - Benchmarking/comparative reviews are very helpful to read
  - Consider trying out multiple softwares with complementary approaches

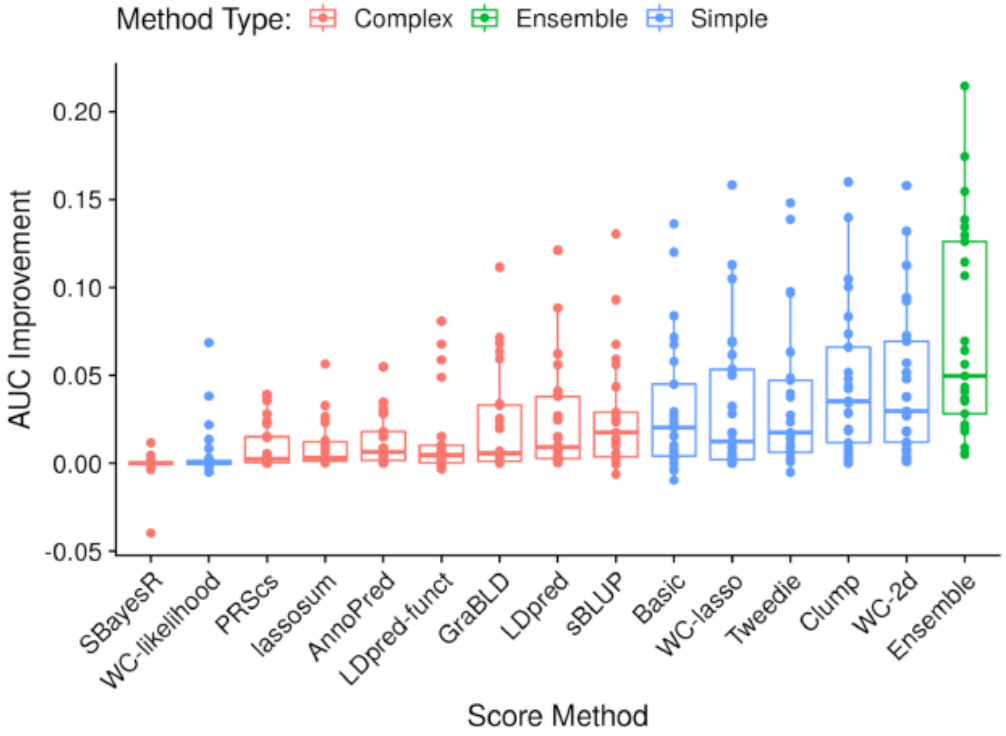
A benchmarking of workflows for detecting differential splicing and differential expression at isoform level in human RNA-seq studies

Gabriela A. Merino, Ana Conesa and Elmer A. Fernández

Corresponding authors: Ana Conesa, Genomics of Gene Expression Lab, Centro de Investigaciones Príncipe Felipe, Eduardo Primo Yúfera 3, 42012, Valencia, España. Tel.: +34 96 328 96 80; and Microbiology and Cell Science Department, Institute for Food and Agricultural Research, University of Florida, 2033 Mowry Road, FL 32610, Gainesville. Tel.: +1 352 2738127; E-mail: aconesa@cipf.es; Elmer A. Fernández, Centro de Investigación y Desarrollo en Inmunología y Enfermedades Infecciosas (CIDIE), CONICET, Av. Armada Argentina 3555, X5016DHK, Córdoba, Argentina. Tel.: +54 351 4938094; E-mail: efernandez@bdmg.com.ar

Benchmarking the Accuracy of Polygenic Risk Scores and their Generative Methods

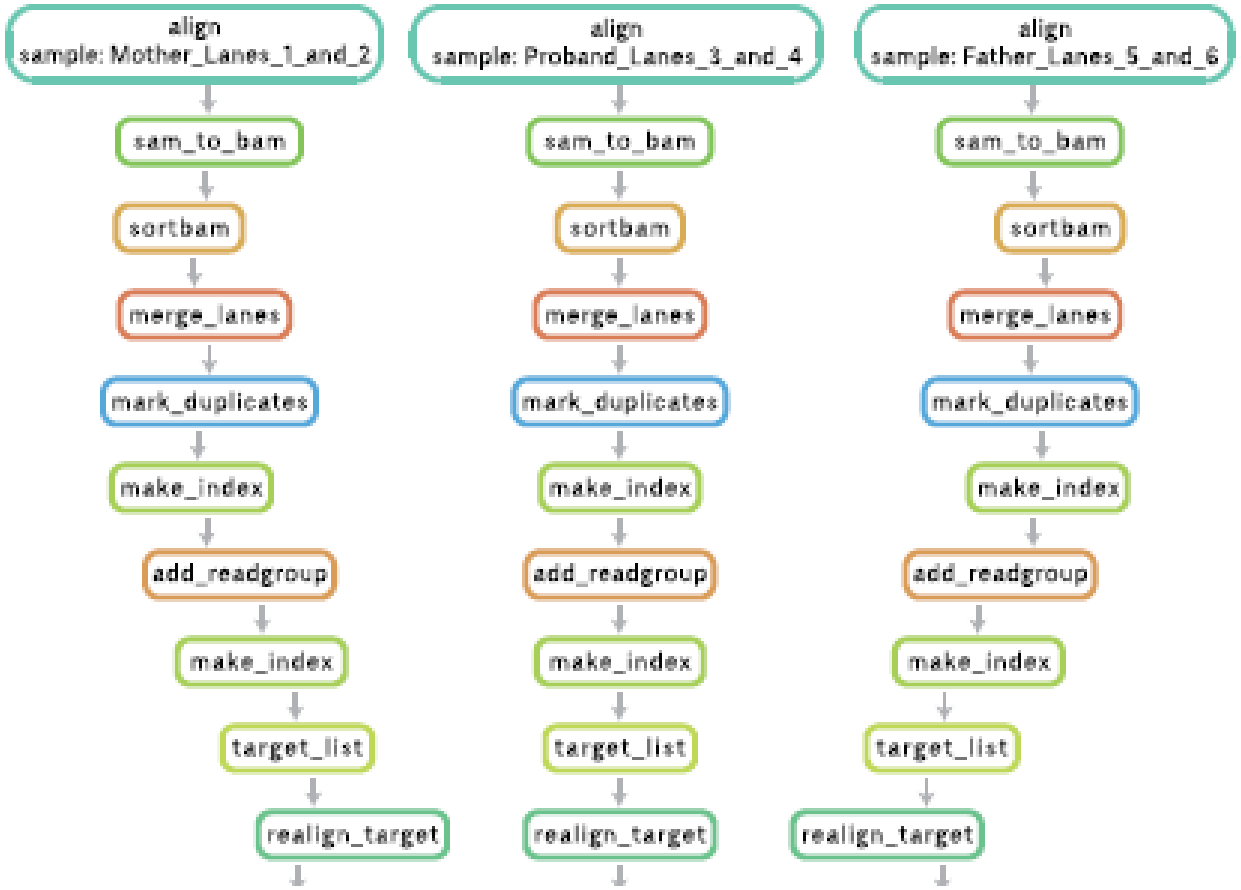
Scott Kulm[1,2,3], Jason Mezey[4,5,*], and Olivier Elemento[2,3,*]



Building a reproducible bioinformatics pipeline

- 2: Write your pipeline: *Read the documentation*

- Map out your pipeline workflow in a flowchart or pseudo-code
- Evaluate all parameters to determine whether they are applicable to your experiment
- Determine which steps are needed to process input or intermediary files
- Quick-start/tutorials can be helpful, but might not be fully applicable



Leipzig, Brief. Bioinf. 2016

Building a reproducible bioinformatics pipeline

Basic Tutorial for Polygenic Risk Score Analyses

Search docs

- Home
  - Overview
  - Datasets
  - Requirements
  - Citation
- 1. QC of Base Data
- 2. QC of Target Data
- 3. Calculating and analysing PRS
- PLINK
- PRSice-2
- LDpred-2
- lassosum
- 4. Visualizing PRS Results

Overview

This tutorial provides a step-by-step guide to performing basic polygenic risk score (PRS) analyses and accompanies our PRS Guide paper. The aim of this tutorial is to provide a simple introduction of PRS analyses to those new to PRS, while equipping existing users with a better understanding of the processes and implementation "underneath the hood" of popular PRS software.

The tutorial is separated into four main sections and reflects the structure of our guide paper: the first two sections on QC correspond to Section 2 of the paper and constitute a 'QC checklist' for PRS analyses, the third section on calculating PRS (here with examples using PLINK, PRSice-2, LDpred-2 and lassosum) corresponds to Section 3 of the paper, while the fourth section, which provides some examples of visualising PRS results, accompanies Section 4 of the paper.

1. Quality Control (QC) of Base Data
2. Quality Control (QC) of Target Data
3. Calculating and analysing PRS
4. Visualising PRS Results

We will be referring to our guide paper in each section and so you may find it helpful to have the paper open as you go through the tutorial.

Choi et al, Nat. Protocols 2020

Building a reproducible bioinformatics pipeline

- 3: Document your pipeline: *Add comments to your code*

PRS_file_processing.sh

```
#Adapted from: https://choishingwan.github.io/PRS-Tutorial/base/

#Stage 1: Quality control of GWAS summary statistics files
#Filter for imputation INFO score >0.8 (when available); skip MAF>0.01 filter, as most GWAS statistics don't have minor allele frequency
gunzip -c sumstats_neuro_sum_ctg_format.txt.gz |awk 'NR==1 || ($14 > 0.8) {print}' |gzip  > filtering/autism_info_filter.txt.gz

#Filter out ambiguous SNPs (SNPs where the alternate allele is complementary to the reference allele)
gunzip -c autism_info_filter.txt.gz | awk '!( ($3=="A" && $4=="T") || ($3=="T" && $4=="A") || ($3=="G" && $4=="C") || ($3=="C" && $4=="G")) {print}' |

#Remove duplicate SNPs (if any) from datasets
gunzip -c autism_ambig_snp_filter.txt.gz |awk '{ print $2}' |sort | uniq -d > dup_snp.txt
gunzip -c autism_ambig_snp_filter.txt.gz |grep -vf dup_snp.txt |gzip - > ../final_files/autism_final.txt.gz


#Stage 2: Generation and QC of PLINK genotype files from VCFs
#Merge VCF files from both batches using bcftools (assume genotypes at missing sites in files are 0/0)
bcftools merge final_calls/16p12_WGS_genotype_final.vcf.gz final_calls_batch4/16p12_WGS_genotype_final.vcf.gz -0 -Oz -o /data5/16p12_WGS/16p12_WGS_genc

#Adjust sample names to include families ("PSU_SG")
bcftools reheader -s sample_family_names.txt 16p12_WGS_genotype_final_all.vcf.gz

#Add dbSNP rsIDs to VCF file
bcftools annotate -a dbsnp_151_all_chr.vcf.gz -c ID -o 16p12_WGS_genotype_final_dbsnp.vcf.gz 16p12_WGS_genotype_final_all_reheader.vcf.gz

#Generate PLINK files (.bed, .bim, .fam) from VCF: Filter for QUAL<50, separate calls into family and sample IDs
plink --vcf 16p12_WGS_genotype_final_dbsnp.vcf.gz --id-delim '_' --vcf-min-qual 50 --out 16p12_WGS_genotype

#Add sex information for samples, and create custom variant IDs for SNPs w/o rsIDs for downstream analysis
plink2 --bfile 16p12_WGS_genotype --update-sex sample_sex_info.txt --set-missing-var-ids @:#[hg19]\$r,\$a --new-id-max-allele-len 750 --make-bed --out
```

Add sources for code

Separate code into major steps

Explain what each line of code does, and why you're doing it

Include explanations for each parameter

Maintaining a reproducible bioinformatics pipeline

- Use GitHub or other version-controlled repository
- Add Readme files to explain datasets and pipelines in each folder
- Maintain a directory document indicating where your files are located

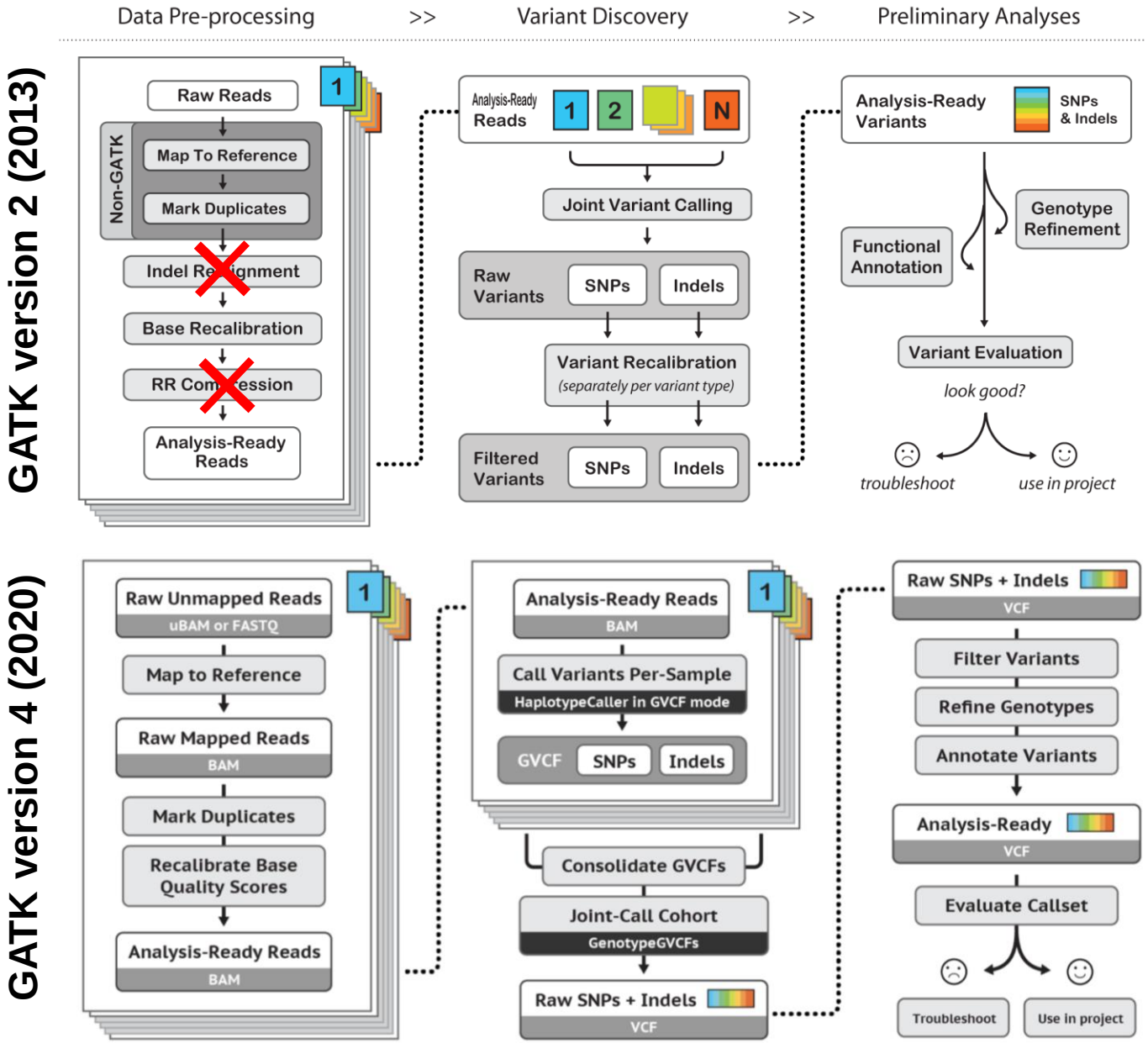
atyryshkina final de calcs `af0b222` 9 days ago

MJ_analysis	WGCNA goseq
ase	cadd and popf
compare_analysis_methods	heatmaps
data	protein genes annotations added
differential_expression_analysis	final de calcs
eqtl/output	added preliminary eqtl data
family_based_analysis	protein tables
figures	plots
outlier_expression_analysis	tried outliers vs gtex
panther_reactome	panther reactome
pathway_analysis/figures/goa	goa analysis
scratch	hgnc names
tissue_expression	tissue expression
README.md	Update README.md

Especially important when multiple people are working on the same project

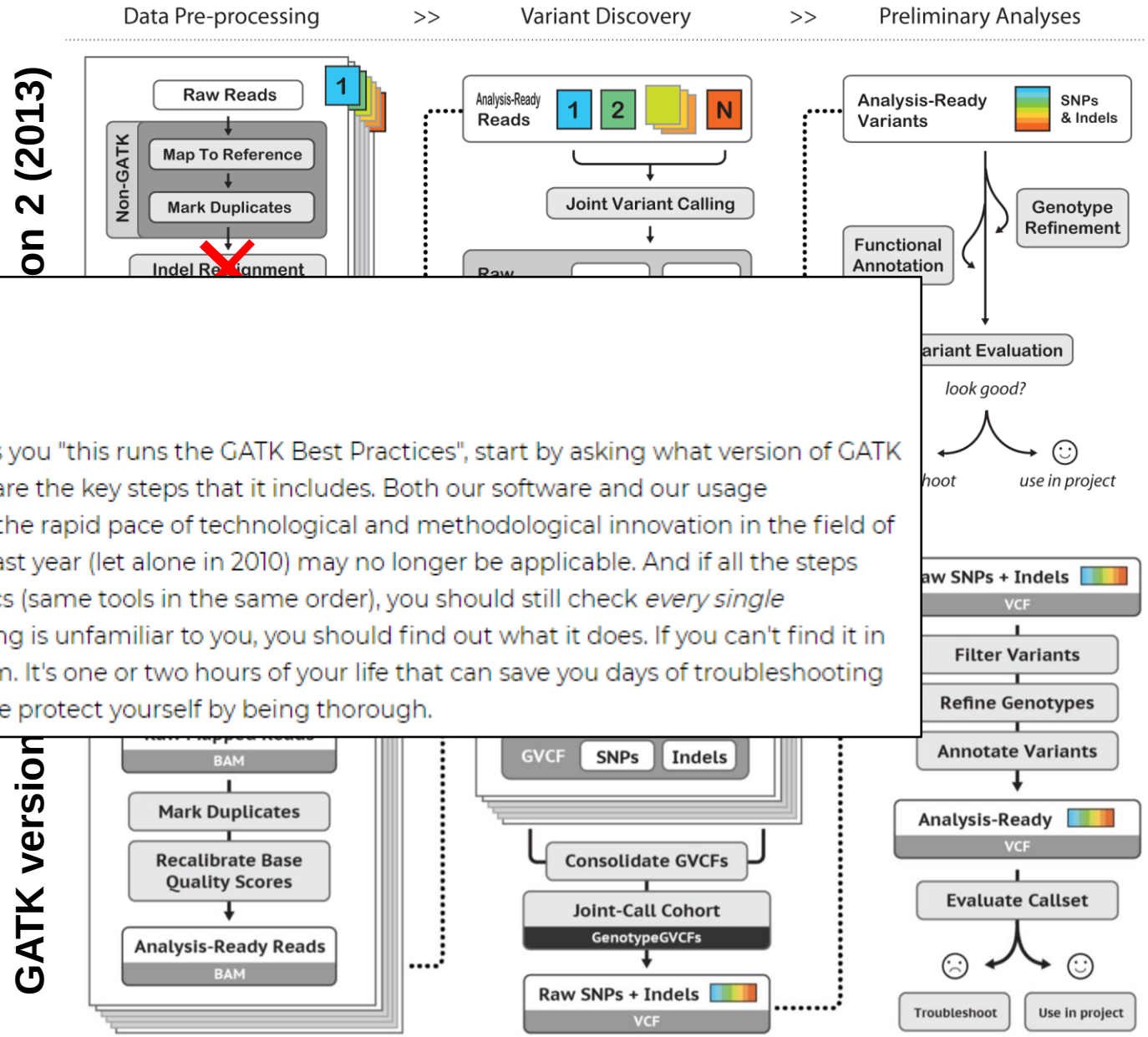
A word of caution for re-using old pipelines

- *Double-check…*
  - Software version and documentation/instructions for use
  - Other softwares that may now be available
  - Parameters/options for application to your specific experiment
  - Reproduce previous results, for software/system configuration



A word of caution for re-using old pipelines

- *Double-check…*
  - Software version and documentation/instructions for use
  - Other software available
  - Parameters/opt application to y experiment
  - Reproduce previous results, for software/system configuration



7. Beware legacy scripts

Trust, but verify.

If someone hands you a script and tells you "this runs the GATK Best Practices", start by asking what version of GATK it uses, when it was written, and what are the key steps that it includes. Both our software and our usage recommendations evolve in step with the rapid pace of technological and methodological innovation in the field of genomics, so what was Best Practice last year (let alone in 2010) may no longer be applicable. And if all the steps seem to be in accordance with our docs (same tools in the same order), you should still check *every single parameter* in the commands. If anything is unfamiliar to you, you should find out what it does. If you can't find it in the documentation, ask us in the forum. It's one or two hours of your life that can save you days of troubleshooting on the tail end of the pipeline, so please protect yourself by being thorough.

Publishing your bioinformatics pipelines

- Most publishers require all code important for the paper's results to be publicly accessible by readers and reviewers

Code and Software Submission Checklist

Prior to submitting your work to Nature Research, we strongly recommend that you ask at least one colleague who is unfamiliar with your software to install the tool(s), follow the instructions, and provide feedback. This process will help ensure that reviewers will also be able to run your software.

You must submit all required content as a single zip file prior to peer review or provide a link where editors and reviewers can access all required content.

Required content

- [x] Compiled standalone software and/or source code
- [x] A small (simulated or real) dataset to demo the software/code

A README file that includes:

1. System requirements
   - [x] All software dependencies and operating systems (including version numbers)
   - [x] Versions the software has been tested on
   - [x] Any required non-standard hardware
2. Installation guide
   - [x] Instructions
   - [x] Typical install time on a "normal" desktop computer
3. Demo
   - [x] Instructions to run on data
   - [x] Expected output
   - [x] Expected run time for demo on a "normal" desktop computer
4. Instructions for use
   - [x] How to run the software on your data
   - [x] (OPTIONAL) Reproduction instructions

     We encourage you to include instructions for reproducing all the quantitative results in the manuscript.

Organizing a public GitHub README file

Description

This GitHub repository contains the scripts and pipelines used to generate bioinformatic data related to the analysis of *Drosophila* homologs of 3q29 genes.

There are three directories in this repository:

Organize repository by experiment/pipeline

1. Pipelines for identifying differentially-expressed fly genes from RNA-Sequencing data for homologs of 3q29 genes.

- This directory contains a batch script for aligning and quantifying read counts using TopHat2 v.2.1.1 and HTSeq v.0.6.1, and an R pipeline for identifying differentially-expressed genes using edgeR v.3.20.1.
- The raw RNA-Seq reads and quantified read counts for biological replicates are available at NCBI GEO accession number GSE128094.

Link to software pages, and include version number

Include links to datasets in public repositories (i.e. GEO)

2. Simulation of apoptosis gene enrichment among candidate neurodevelopmental genes.

- This directory contains an R script which simulates enrichment of apoptosis genes in schizophrenia gene sets, and raw text files containing apoptosis, RefSeq, and candidate NDD gene sets used in the analysis.
- The candidate gene sets are derived from Purcell et al, *Nature* 2014 (schizophrenia), SFARI Gene (autism), and Developmental Delay G2P database (ID/DD).

Alternatively, provide datasets (or toy examples) along with code

3. Network analysis of CNV genes and simulated random gene sets.

- This directory contains two Python scripts for analyzing the connectivity of genes within a human brain-specific interaction network.
- One script (`nearest_neighbor_weighted_allgenes.py`) takes an individual gene as standard input (gene name and

Describe contents of each directory

Organizing a public GitHub README file

- The network file used in this analysis (`brain.degnorm-ge2.prob-gept02.dat`) is described in Greene *et al, Nat. Genet.* 2015 and Krishnan *et al, Nat. Neurosci.* 2016; we generated a sub-network that only contained edges with weights >2.0 (the top 0.5% of interactions in the network).

Bash pipeline scripts can be run in any Unix environment. R scripts can be run using any R version (scripts were generated using R v.3.4.2.). Python scripts for network analysis can be run in Python2 (scripts were generated using Python v.2.7.16) and require the NetworkX package v.2.4.

Provide operating system and package requirements

Citation

Singh MD, Jensen M, Lasser M, Huber E, Yusuff T, Pizzo L, Lifschutz B, Desai I, Kubina A, Yennawar S, Kim S, Iyer J, Rincon-Limas DE, Lowery LA, Girirajan S. *NCBP2* modulates neurodevelopmental defects of the 3q29 deletion in *Drosophila* and *X. laevis* models. **BioRxiv** 614750; 16 Oct. 2019.

Cite paper, and link back to personal or lab website

Copyright/License

The code in this repository is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Include GNU license as appropriate

Documenting pipeline code for public presentation

master 3q29_project / Apoptosis_NDD_simulation / apoptosis_overlap.r Go to file ...

Unknown Added network analysis scripts and expanded apoptosis/NDD gene overla... ... Latest commit 2c27d8f on Oct 17, 2019 History

0 contributors

75 lines (59 sloc) 3.68 KB Raw Blame

```
#This R script first finds the overlap of genes that are annotated for apoptosis function (apoptotic process, GO:0006915 or children terms)
#among three sets of candidate neurodevelopmental (NDD) genes: schizophrenia (derived from Purcell et al, Nature 2014); autism (derived from SFARI Gene database);
#and intellectual disablity (derived from DD G2P database). It then performs 100,000 simulations of randomly selecting the same number of genes in each
#NDD geneset from the set of all RefSeq genes, and determining the number of genes with apoptosis in each random set .

#A list of all apoptosis genes is provided in the file human_apoptotic_process_raw.txt, a list of all RefSeq genes is provided
#in the file refseq_genes.txt, and lists of the candidate genes are provided in the files schiz_genes.txt, ASD_genes.txt, and refseq_genes.txt.
#See README.MD for more information on these gene sets.

#Open gene lists
apoptosis<-read.table("human_apoptotic_process_raw.txt",sep="\t")
schiz<-read.table("schiz_genes.txt")
asd<-read.table("ASD_genes.txt",sep="\t")
id.dd<-read.table("ID-DD_genes.txt",sep=",")
refseq<-read.table("refseq_genes.txt")
```

Brief description/pseudo-code

Describe input files,
and how to obtain them

Key points for bioinformatics pipeline reproducibility

- Read through the literature and documentation of different softwares to determine if they are best for your pipeline
- Always comment your code, and keep track of software versions and configurations (especially with old pipelines)
- Provide as much information as possible for a lab member, collaborator, or peer reviewer to run your pipeline and get the same results

Questions?

Contact: mpj5142@psu.edu