**QUICKSTART**

**(1) Get example data**

--------------------------------

$ mkdir tmp

$ cd tmp

$ cp -a /ccmb/BioinfCore/Common/pipelines/Watermelon/example\_data .

--------------------------------

**(2) Load the watermelon module**

--------------------------------

$ module load watermelon

loaded watermelon/0.1

--------------------------------

**(3) Setup your analysis using: watermelon-init**

--------------------------------

$ watermelon-init –help

usage: watermelon-init [-h] --genome\_build {hg19,mm10,rn5}

[--job\_suffix JOB\_SUFFIX]

source\_fastq\_dir

--------------------------------

This creates input directory with the (1) Fastqfiles (2) Analysis dir with a config.yaml which contains sample and reference paths for the genome-build you specified.

--------------------------------

$ mkdir Tuttle\_ctuttle\_RS1\_balex\_HI\_1592

$ cd Tuttle\_ctuttle\_RS1\_balex\_HI\_1592

$ watermelon-init --genome\_build mm10 ../example\_data/00-multiplexed\_reads

$ tree

.

├── analysis\_02\_10

│   └── config\_02\_10.yaml

├── inputs

│   └── 00-multiplexed\_reads

│   ├── Sample\_61483 -> /tmp/example\_data/00-multiplexed\_reads/Sample\_61483

│   ├── Sample\_61484 -> /tmp/example\_data/00-multiplexed\_reads/Sample\_61484

│   ├── Sample\_61485 -> /tmp/example\_data/00-multiplexed\_reads/Sample\_61485

...

│   └── Sample\_61506 -> /tmp/example\_data/00-multiplexed\_reads/Sample\_61506

└── watermelon.README

--------------------------------

**(4) Review analysis set-up**

--------------------------------

$ cd analysis\_02\_10

--------------------------------

**Edit config.yaml to accurately represent your analysis design**

The yaml file that is generated does contain the sample information from the source fastq dir you specified when you ran watermelon-init. However, it is not complete. You need to:

1. Review/adjust phenotype names
2. Review and adjust samples names (Note: if you change names in config, also change the sample dir names in the input dir)
3. Add a phenotype group(s) for each sample
4. Add comparisons
5. Review genome and references
6. Review alignment options
7. Review trimming options

**A working config.yaml for the test data is available in:**

--------------------------------

example\_data/example\_config.yaml

Note that if you simply copy this file, you will need to adjust the “input\_dir”.

--------------------------------

**(5) Run RNA-Seq analysis with Watermelon**

**From the analysis directory, start a screen session**

--------------------------------

$ screen -S ctuttle\_rs1\_02\_01

--------------------------------

**Run watermelon with the --dryrun option to validate the config and check the execution plan.**

--------------------------------

Dry run gives you an overview of the job you will be running:

It has the following sections:

1. Validation
2. snakemake command
3. Tasks
4. Job counts
5. Conclusion

$ watermelon --dryrun -c config\_02\_01.yaml

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

config validation: OK

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

watermelon (v0.1) begins

snakemake --configfile config\_02\_07.yaml --snakefile /nfs/med-bfx-common/pipelines/Watermelon/Watermelon/rnaseq.snakefile --cores 40 -T --dryrun

**[Wed Feb 1 14:00:34 2017] rule concat\_reads:**

**input: /ccmb/BioinfCore/SoftwareDev/projects/Watermelon\_spike/Tronson\_RS1\_tests/test\_data/inputs/00-multiplexed\_reads/Sample\_61483/**

**output: alignment\_results/01-raw\_reads/Sample\_61483\_R1.fastq.gz**

**wildcards: sample=Sample\_61483**

**[Wed Feb 1 14:00:34 2017] rule concat\_reads:**

**input: /ccmb/BioinfCore/SoftwareDev/projects/Watermelon\_spike/Tronson\_RS1\_tests/test\_data/inputs/00-multiplexed\_reads/Sample\_61486/**

**output: alignment\_results/01-raw\_reads/Sample\_61486\_R1.fastq.gz**

**wildcards: sample=Sample\_61486**

**[Wed Feb 1 14:00:34 2017] rule concat\_reads:**

**…..**

……

**[Wed Feb 1 14:00:35 2017] Job counts:**

**count jobs**

**1 align\_qc\_metrics**

**1 all**

**4 annotate\_diffex\_flag**

**4 build\_group\_replicates**

**1 build\_run\_info**

**8 concat\_reads**

**1 create\_transcriptome\_index**

**4 cuffdiff**

**4 cummerbund**

**8 cutadapt**

**1 deliverables\_align\_cuffdiff**

**4 deliverables\_cummerbund**

**7 diffex\_excel**

**4 diffex\_flag**

**4 diffex\_flip**

**7 diffex\_split**

**8 fastqc\_tophat\_align**

**8 fastqc\_trimmed\_reads**

**1 htseq\_merge**

**8 htseq\_per\_sample**

**1 last\_split**

**8 tophat**

**97**

elapsed seconds: 3

elapsed time: 0h:0m:3s

Watermelon complete: /tmp/Tuttle\_ctuttle\_RS1\_balex\_HI\_1592/analysis\_02\_07/config\_02\_07.yaml (No errors)

-------------------------------

Note: you can also run watermelon in phases: e.g. just until alignment. See ‘Watermelon in Detail’ doc.

**Run watermelon**

This should take a while. Detach from the screen (ctrl-A D) to ensure connectivity issues do not affect your analysis job. You should receive an email when watermelon completes.

--------------------------------

$ watermelon -c config\_02\_01.yaml --cores 30

--------------------------------

**(6) Review analysis results**

Once the analysis is complete, you will get an email to your umich.edu account with the analysis status.

For a successful analysis:

--------------------------------

Watermelon complete: test\_run\_watermelon/analysis\_02\_01/config\_02\_01.yaml (No errors)

--------------------------------

For a failed analysis:

--------------------------------

ERROR: Watermelon failed: test\_run\_watermelon/analysis\_02\_01/config\_02\_01.yaml

See logs for details: test\_run\_watermelon/analysis\_02\_01/logs/watermelon.log

--------------------------------

**Results of the analysis are in : diffex\_results/Deliverables**

diffex\_results

└── Deliverables

├── diffex

│   ├── cuffdiff\_results

│   │   ├── diet

│   │   ├── female.diet

│   │   ├── gender

│   │   ├── log

│   │   └── male.diet

│   └── cummeRbund\_results

│   ├── diet\_cummeRbund\_plots

│   ├── female.diet\_cummeRbund\_plots

│   ├── gender\_cummeRbund\_plots

│   └── male.diet\_cummeRbund\_plots

└── qc

├── aligned\_reads\_fastqc

│   └── log

└── raw\_reads\_fastqc

└── log

--------------------------------

**Results of each analysis step are in specific sub-directories.**

**‘analysis\_results’ contains results for the following analysis steps:**

1. Concat raw reads
2. Trimming
3. FastQC (before and after alignment)
4. Tophat

--------------------------------

$ tree –d alignment\_results

alignment\_results/

├── 01-raw\_reads

├── 02-cutadapt

├── 03-fastqc\_reads

├── 04-tophat

│   ├── Sample\_61483

│   ├── Sample\_61484

│   ├── Sample\_61485

…

│   ├── Sample\_61506

│   └── transcriptome\_index

│   └── tophat\_out

│   ├── logs

│   └── tmp

├── 05-fastqc\_align

└── 06-qc\_metrics

--------------------------------

**‘diffex\_results’ contains results for the following analysis steps:**

1. HTSeq
2. Cuffdiff
3. Excel files of differentially expressed genes and isoforms
4. CummeRbund
5. Deliverables

--------------------------------

$ tree -d diffex\_results/

diffex\_results/

├── 07-htseq

├── 08-cuffdiff

│   ├── diet

│   ├── female.diet

│   ├── gender

│   └── male.diet

├── 09-diffex\_flip

│   └──[phenoptypes as above]

├── 10-diffex\_flag

│   └──[phenoptypes as above]

├── 11-annotate\_diffex\_flag

│   └──[phenoptypes as above]

├── 12-group\_replicates

│   └──[phenoptypes as above]

├── 13-cummerbund

│   ├── diet

│   │   └── Plots

│   ├── female.diet

│   │   └── Plots

│   ├── gender

│   │   └── Plots

│   └── male.diet

│   └── Plots

├── 14-diffex\_split

│   └──[phenoptypes as above]

├── 15-run\_info

├── 16-diffex\_excel

│   └──[phenoptypes as above]

└── Deliverables

├── diffex

│   ├── cuffdiff\_results

│   │   ├── diet

│   │   ├── female.diet

│   │   ├── gender

│   │   └── male.diet

│   └── cummeRbund\_results

│   ├── diet\_cummeRbund\_plots

│   ├── female.diet\_cummeRbund\_plots

│   ├── gender\_cummeRbund\_plots

│   └── male.diet\_cummeRbund\_plots

└── qc

├── aligned\_reads\_fastqc

└── raw\_reads\_fastqc

--------------------------------

**Logs:**

**Timestamped logs are in the logs dir.**

Watermelon.log contains the stdout and stderr of each step of the pipeline.

You can tail watermelon.log to view progress of the job. N.B. Snakemake parallelizes samples tasks where it can, so results from samples may be interleaved in this log.

Additional logs are available in each analysis sub-directory

--------------------------------

$ tree logs

logs

├── 20170201\_134430

  ├── config\_02\_01.yaml.20170201\_134430

   ├── rnaseq.snakefile.20170201\_134430

   ├── watermelon.detailed\_summary

   └── watermelon.log

--------------------------------