

Genome Informatics 2023

Lesson 2 - Portable and reproducible bioinformatic analysis

Lesson overview

1. Common Workflow Language (CWL). Building apps (tools and workflows)
2. Docker
3. Constructing and running portable and reproducible bioinformatics analysis
4. Jupyter Notebook bioinformatic analysis on the cloud
5. Python introduction

Common Workflow Language

- CWL is a way to describe command line tools execution
- Every tool has defined set of inputs and outputs
- Every tool is executed in its own environment (Docker)
- Execution on the cloud or local environment
- Enables portable and reproducible execution

1st-tool.cwl

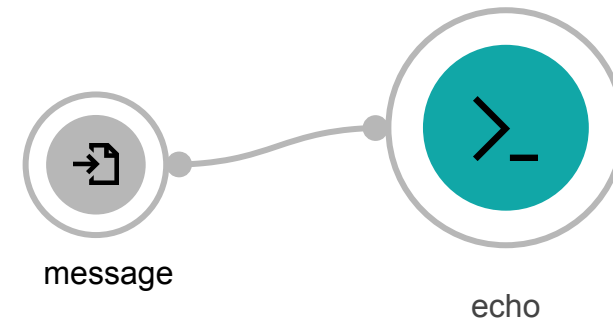
```
#!/usr/bin/env cwl-runner

cwlVersion: v1.0
class: CommandLineTool
baseCommand: echo
inputs:
  message:
    type: string
    inputBinding:
      position: 1
outputs: []
```

echo-job.yml

```
message: Hello world!
```

Used by CWL executor



CWL: Simple instructions for reproducible analyses

```
1 class: CommandLineTool
2 cwlVersion: v1.0
3
4 id: bam_tools_index
5 label: Bam Tools Index
6
7 requirements:
8   - class: DockerRequirement
9     dockerPull: 'images.sbgenomics.com/markop/bamtools:2.4.0'
10 # - class: InitialWorkDirRequirement
11 #   listing:
12 #     - $(inputs.input_bam)
13
14 baseCommand:
15   - /opt/bamtools/bin/bamtools
16   - index
17
18 inputs:
19   - id: input_bam
20     type: File
21     inputBinding:
22       position: 1
23       prefix: '-in'
24
25 outputs:
26   - id: indexed_bam
27     type: File
28     outputBinding:
29       glob: '*.bam'
30     secondaryFiles:
31       - .bai
```

Text in YAML or JSON format.

Describes the tools and workflows.

Easier and faster to deploy tools

Wide adoption by 40+ institutes/research groups

Avoids lock-in to a given system

produces the command line

```
/opt/bamtools/bin/bamtools index -in input_bam.
```

How do I learn CWL?

You can learn the syntax: [CWL User Guide](#)

BUT you don't have to!

With the Seven Bridges [Software Development Kit](#) (Tools/Workflow Editor & Rabix Composer), you can easily create tools and chain them into workflows interactively and without any programming experience.

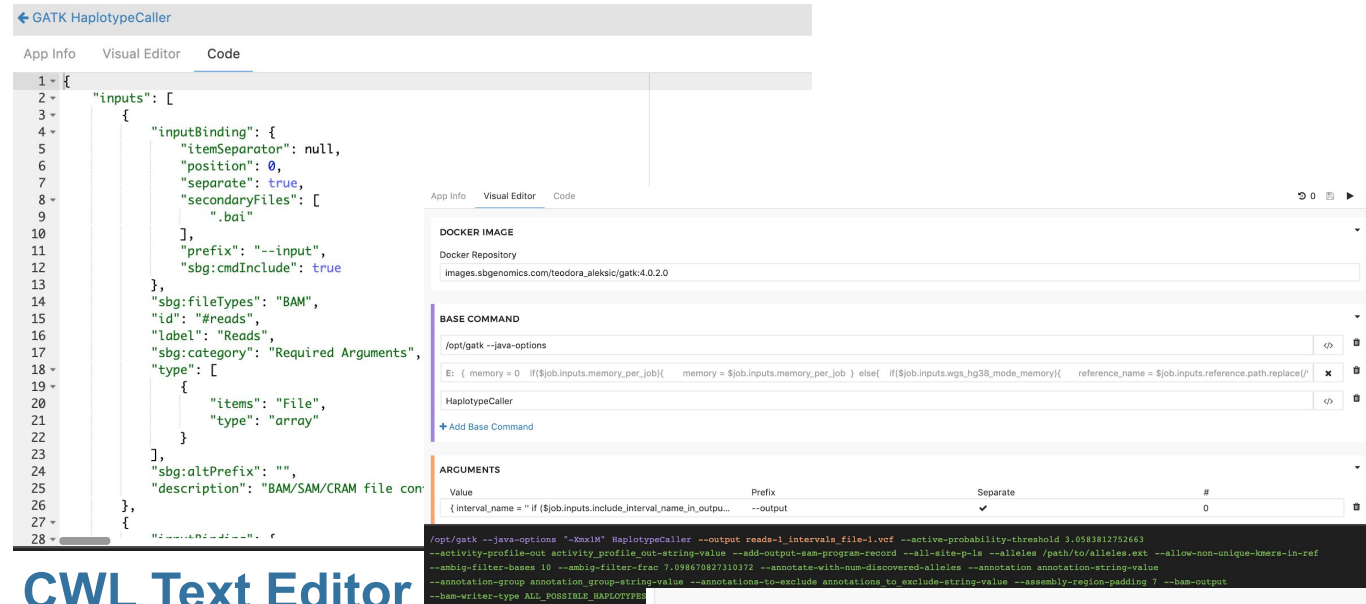
[**The Seven Bridges SDK will create the CWL code for you**](#)

so you can get your tool up and running on the platform more quickly and easily.

Rabix Composer

An Integrated Development Environment for CWL developers

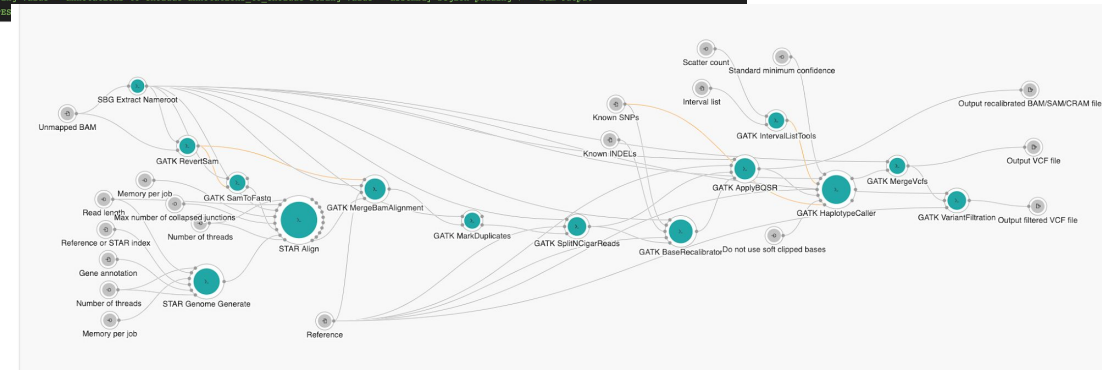
- **Compatible** with different versions of CWL
- **Version history**
- **Graphical editors**
- **In-line documentation**
- Support for popular **scripting** languages
- **Desktop Version** - local testing
- **Web Composer**



CWL Text Editor

Tool Editor

Workflow Editor



Rabix Composer

An Integrated Development Environment for CWL developers

Sambamba Sort (Edited)

vladimirk

App InfoVisual EditorCode

0

DOCKER IMAGE

Docker Repository

images.sbgenomics.com/mladenlsbg/sambamba:0.5.9

BASE COMMAND

/opt/sambamba_0.5.9/sambamba_v0.5.9 sort

</>

+ Add Base Command

ARGUMENTS

Value	Prefix	Separate	#
{ if (\$job.inputs.input) { file = [],concat(\$job.inputs.input) filename = file[0].path return fi...	--out=	x	3

+ Add an Argument

INPUT PORTS

ID	Type	Binding
input	File	pos: 100
filter	string?	--filter

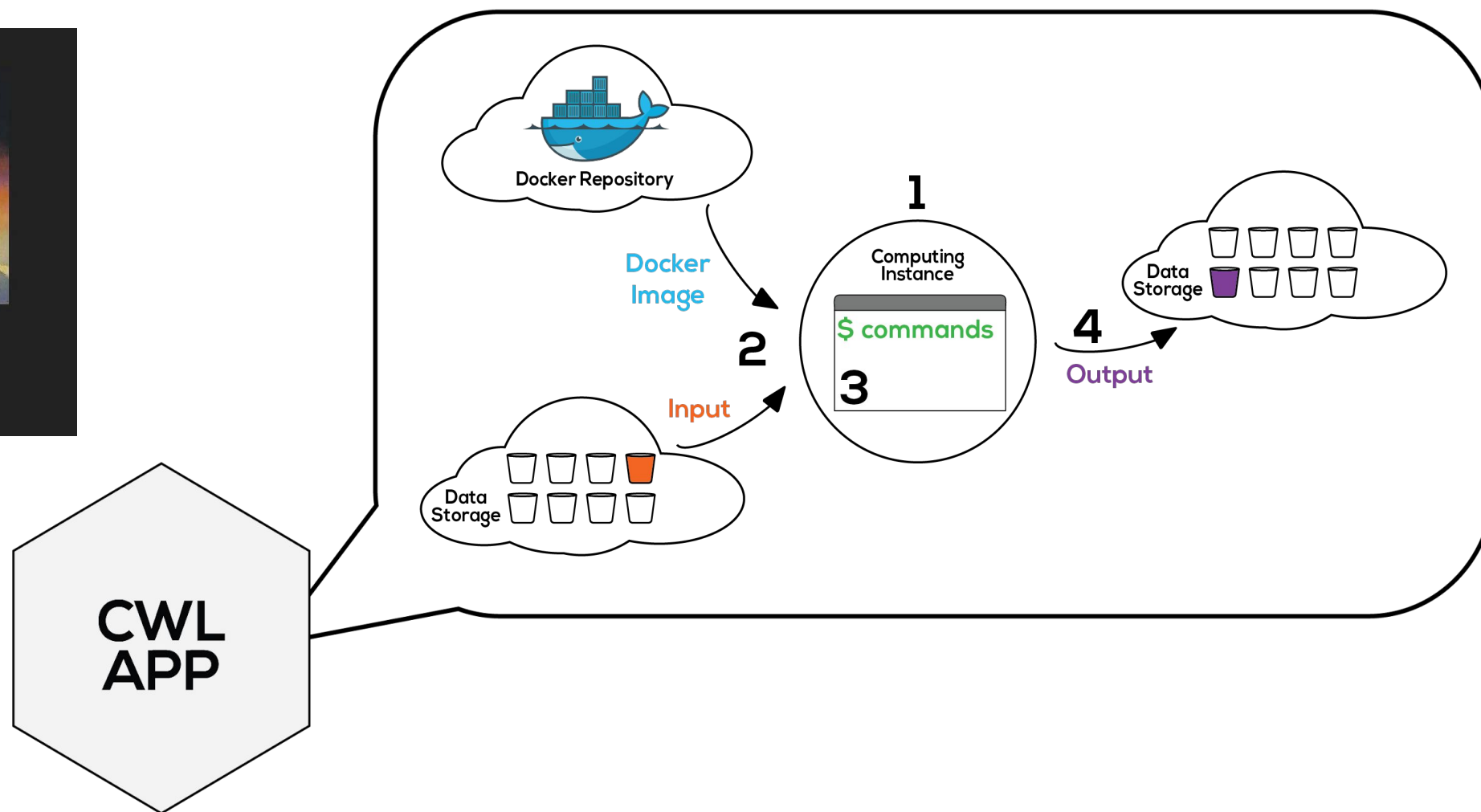
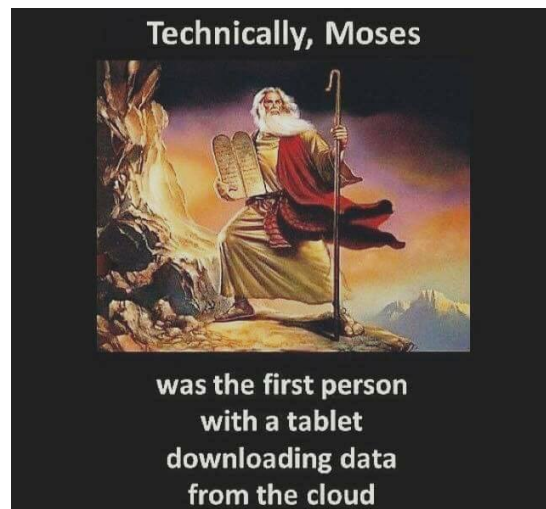
+ Add an Input

OUTPUT PORTS

ID	Type	Glob
sorted	File?	{ if (\$job.inputs.input) { filename = \$job.inputs.input.path return filename.split('.').slice(...

/opt/sambamba_0.5.9/sambamba_v0.5.9 sort --filter "filter-string-value" --out=input.sorted.bam /path/to/input.ext

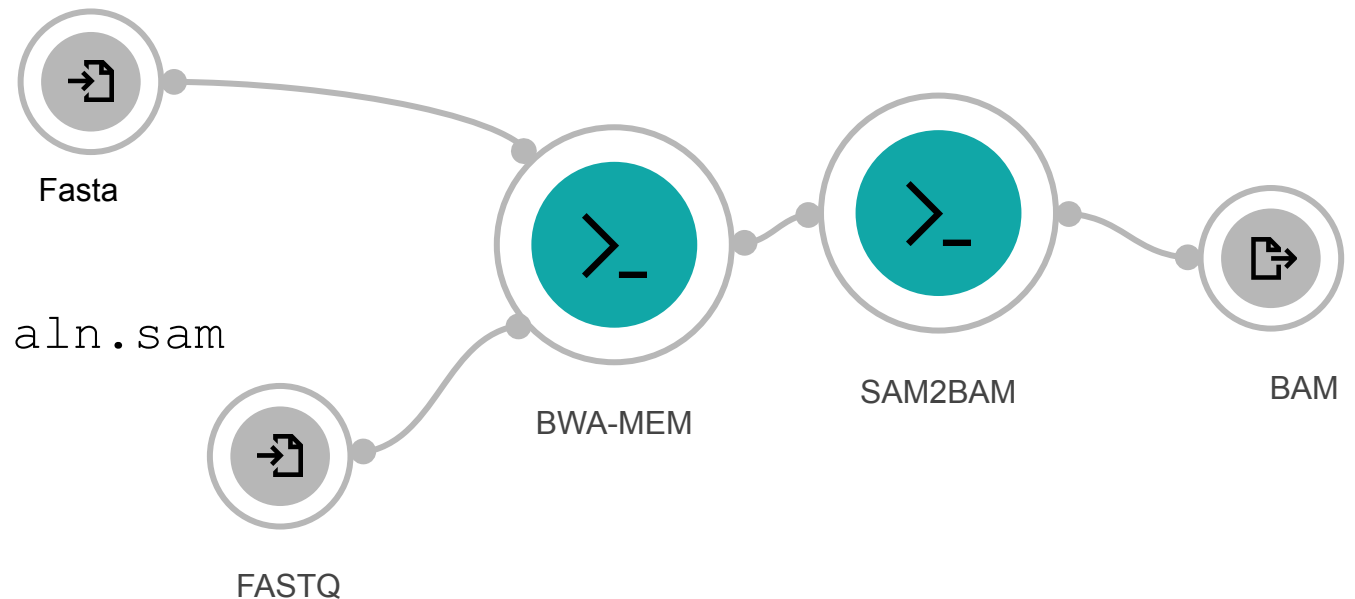
CWL @ Cloud



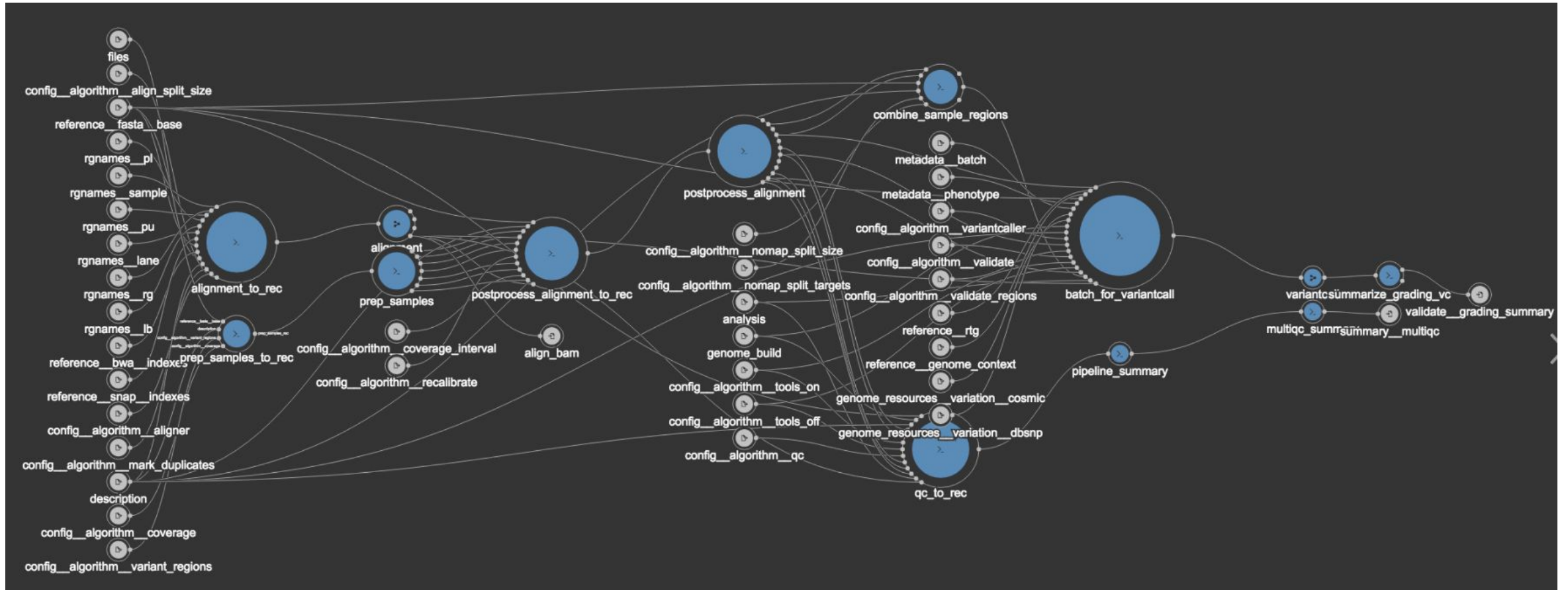
What is a CWL workflow?

- Acyclic graph of tools connected to perform some analysis
- Workflow's nodes are:
 - Inputs (file or parameter)
 - Tools
 - Outputs
 - Workflow

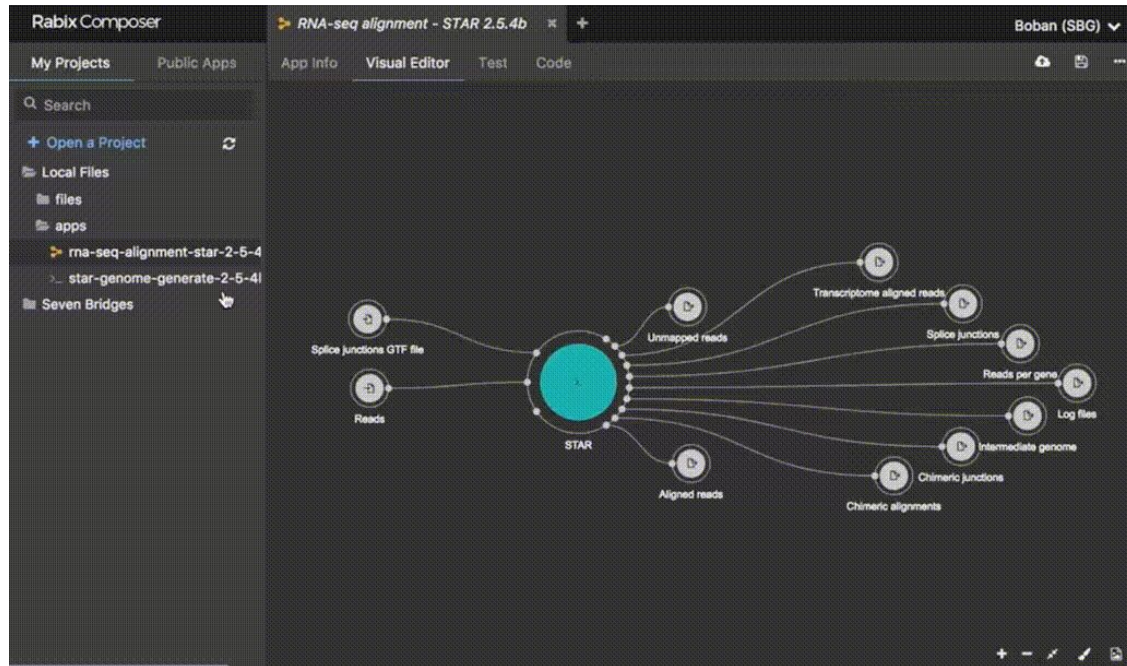
```
bwa mem ref.fa read1.fq read2.fq > aln.sam  
sam2bam aln.sam > aln.bam
```



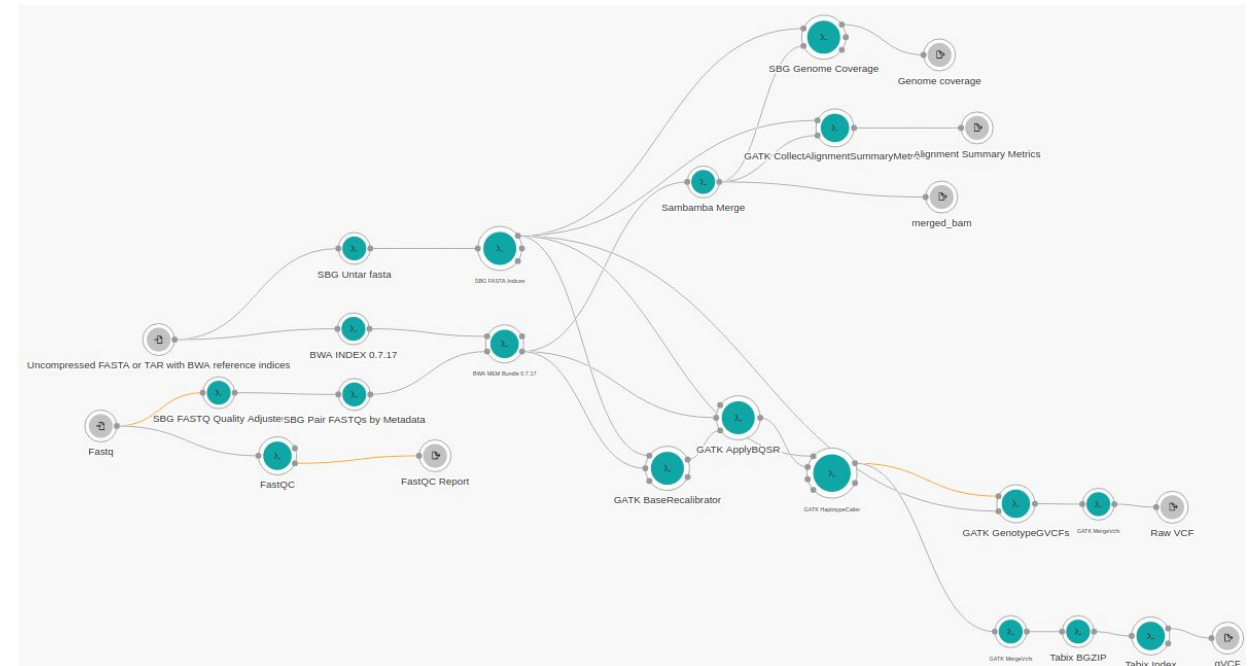
Why we need a workflow?



How to build a workflow?



[Desktop CWL composer](#)

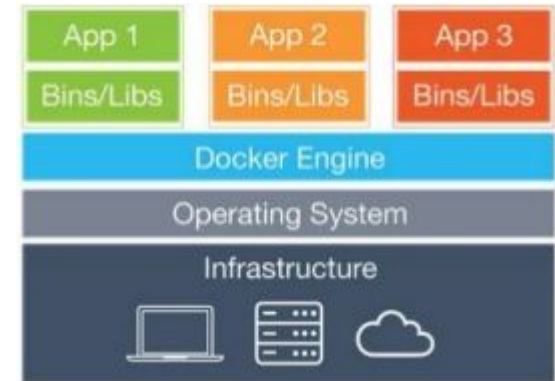
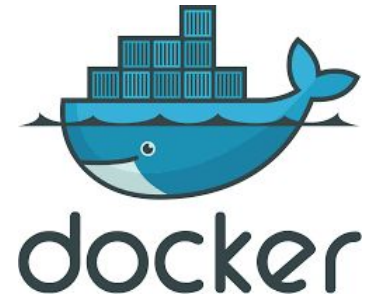


[Web CWL composer](#)

2. Docker

What is Docker?

- Docker is a light-weight virtual environment
- Allows you to package the tool (e.g. Python script or some C program) with all of its dependencies into the standardized unit for software development
- Docker containers run on any computer, on any infrastructure
- Layered container structure
- Can directly access resources of host operating system



How to create Docker image?

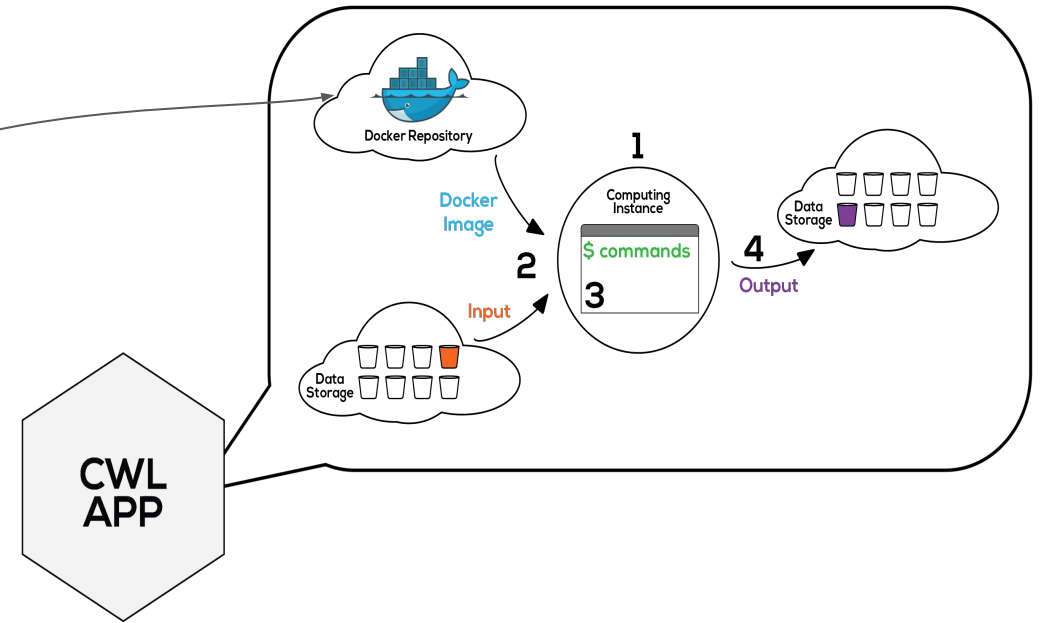
```
FROM ubuntu:16.04
MAINTAINER vladimir.kovacevic@sbgenomics.com
RUN apt-get update && apt-get install -y wget \
make \
gcc \
zlib1g-dev
WORKDIR /opt
RUN wget
https://github.com/bwa/releases/bwa-0.7.15.tar.bz
2
RUN tar xjf bwa-0.7.15.tar.bz2
WORKDIR /opt/bwa-0.7.15
RUN make
COPY Dockerfile /opt/Dockerfile
```

Docker
file

```
# Build image from Dockerfile and push to docker repo

docker build -t images.sbgenomics.com/vladimirk/bwa:0.7.15 .

docker push images.sbgenomics.com/vladimirk/bwa:0.7.15
```



How to create Docker image?

```
FROM ubuntu:16.04
MAINTAINER vladimir.kovacevic@sbgenomics.com
RUN apt-get update && apt-get install -y wget make
WORKDIR /opt
RUN wget https://github.com/bwa/releases/bwa-0.7.15.tar.bz2
RUN tar xfj bwa-0.7.15.tar.bz2
WORKDIR /opt/bwa-0.7.15
RUN make
COPY Dockerfile /opt/Dockerfile
```

Docker
file

```
# Build image from Dockerfile and push to docker repo
docker build -t vladimirk/bwa:0.7.15 .

docker push vladimirk/bwa:0.7.15
```

Best practice: Using a Dockerfile

A Dockerfile is a text file that stores commands to create a Docker image

- Uses a domain-specific language to describe how to build an image
- The Docker tool automates the building of an image from a Dockerfile
- Docker reads commands and executes in succession

Benefits:

- Stores whole procedure of image creation
- Helps facilitate and automate the process of maintaining tools that are wrapped for the platform
 - Automate builds
 - Can be used as the source of documentation at failure points and can restart failed builds
 - Transparency
 - Easy to share on GitHub/DockerHub

A Dockerfile consists of **Instructions** followed by **arguments** and comments:

#Comment

INSTRUCTION arguments

Dockerfile Instructions

FROM	<ul style="list-style-type: none">• Initializes new build stage and sets Base Image (“pulling”)
RUN	<ul style="list-style-type: none">• Executes the command of argument during build process• Execution results are committed to current image and resulting image is used for next instruction• Chain multiple commands with && and \ for a line break
CMD	<ul style="list-style-type: none">• Provides default command, which is executed inside container when it’s created based on image• Need to use argument [“/bin/bash”] , as that is how the container is invoked during task execution for SB Platform
ADD	<ul style="list-style-type: none">• Used to copy files, directories, or remote file URLs from original location <source> to container destination path <destination>• You can only specify those source paths that are within context directory
COPY	<ul style="list-style-type: none">• Used to copy files or directories to container at specified path• Unlike ADD, doesn’t take URL as <source> and will not unpack archived file as <source>
WORKDIR	<ul style="list-style-type: none">• Used to set default working directory for container. Instructions will be executed in the defined working directory

Use a Dockerfile to build an image

```
1  # Define base image
2
3  FROM ubuntu:latest
4
5  # Install required packages
6
7  RUN apt-get update && apt-get install -y \
8      wget \
9      python3-pip \
10     libhdf5-dev
11
12 # Install python modules
13
14 RUN pip3 install numpy
15 RUN pip3 install h5py
16
17 #Install Kallisto
18
19 WORKDIR /opt
20 RUN wget https://github.com/pachterlab/kallisto/releases/download/
21     v0.43.1/kallisto_linux-v0.43.1.tar.gz
22 RUN tar -zxvf kallisto_linux-v0.43.1.tar.gz
23 RUN rm -rf kallisto_linux-v0.43.1.tar.gz
24
25 # Add to path
26 ENV PATH /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/opt/
27     kallisto_linux-v0.43.1
28
29 COPY Dockerfile /opt/
30 MAINTAINER Kristina Clemens, Seven Bridges, <kristina.clemens@sbgenomics.com>
```

3. Constructing and running portable and reproducible bioinformatics analysis

Cancer Genomics Cloud platform

[HOME](#)[ABOUT](#)[POLICIES](#)[KNOWLEDGE CENTER](#)[LOGIN](#)

- Two petabytes of multi-dimensional genomics data available to ~3800 authorized researchers to analyse on the cloud
- The Cancer Genome Atlas (TCGA), a landmark cancer genomics program, molecularly characterized over 20,000 primary cancer and matched normal samples

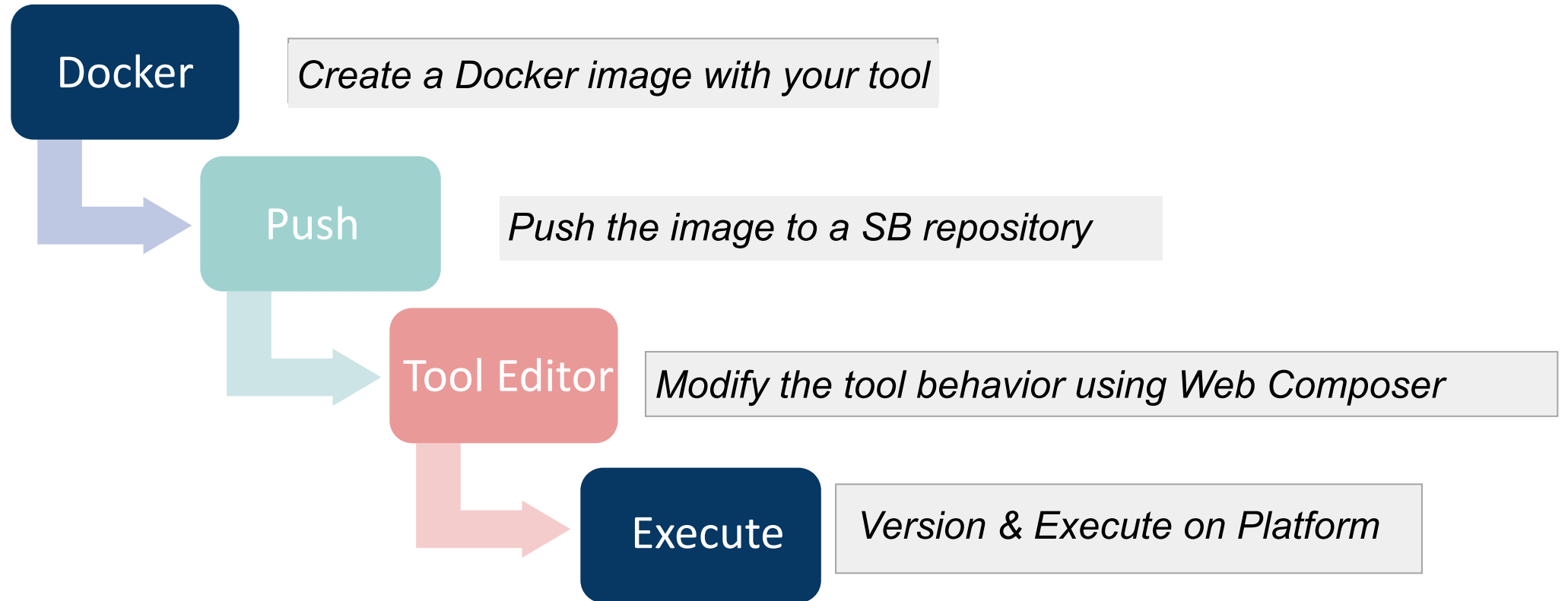
Learn from cancer genomics data.

FASTER

- Free registration for academia with \$300 credit!



Bringing your own tools to the Platform



Let's build some tool!

← grep

vladimir

App Info

Visual Editor

Code

3

DOCKER IMAGE

Docker Repository

images.sbgenomics.com/sevenbridges/ubuntu:14.04

BASE COMMAND

grep

+ Add Base Command

ARGUMENTS

Value	Prefix	Separate	#
> grep_out.txt	×	✓	2

+ Add an Argument

INPUT PORTS

ID	Type	Binding
input	File	pos: 1
pattern	string	pos: 0

+ Add an Input

OUTPUT PORTS


ID	Type	Glob
output	File?	grep_out.txt

+ Add an Output

COMPUTATIONAL RESOURCES

grep pattern-string-value /path/to/input.ext > grep_out.txt


...and run it!

 Projects ▾ Data ▾ Public Apps Public projects ▾ Automations Developer Staff ▾

42 ⓘ

Interactive Analysis Settings Notes


Dashboard Files Apps **Tasks**


COMPLETED **grep run - 10-23-19 22:09:34** 

Executed on Oct. 24, 2019 00:09 by [vladimirk](#)

Spot Instances: [On ⓘ](#) | Memoization: [On ⓘ](#) | Price: [\\$0.01 ⓘ](#) [Refund](#) [View refunds](#) | Duration: [1 minute ⓘ](#)

▾ App: [grep](#) - Revision: 3

Inputs 


▾ **Input file** ⓘ 


[PhiX_genome.fasta](#)

App Settings

Pattern to search for

[Show non-default ▾](#)

Outputs 

▾ **output** 


[grep_out.txt](#)

PhiX is an icosahedral, nontailed bacteriophage with a single-stranded DNA. It has a tiny **genome** with 5386 nucleotides and was the first DNA **genome** to be sequenced by Fred Sanger. Due to its small, well-defined **genome** sequence, **PhiX** has been commonly used as a control for Illumina sequencing runs.

So, what just happened?

- Request for default (c4.2xlarge) instance sent to aws
- Initialize instance
- cwl.job.json created from task inputs and parameters
- Together with cwl.app.json sent to initialized aws instance
- Download input files to the aws instance
- Download of docker image(s) of the tool(s)
- Run the tool inside docker container
- Collect marked outputs and upload them to the cloud storage attached to our platform's project

What about some real data?



Projects ▾

Data ▾

Public Apps

Public projects ▾

Automations

Developer

Staff ▾

42 ⓘ


Interactive Analysis


Settings


Notes


DashboardFilesAppsTasks

DRAFT

Whole Exome Sequencing - BWA + GATK 4.0 (with Metrics) run - 10-24-19 08:56:13 

 Get support

 Discard

 Run


Last update by vladimirk on Oct. 24, 2019 10:56


▾ App: Whole Exome Sequencing - BWA + GATK 4.0 (with Metrics) - Revision: 0

1 Task Inputs

Execution Settings

Inputs



Batching ⓘ Off 



▾ FASTQ * ⓘ  Select file(s)


This input is required.


No files selected


This field is required and cannot be empty.



▾ 1000g phase1 indels * ⓘ  Change selection
 1000G_phase1.indels.b37.vcf

▾ Mills * ⓘ  Change selection
 Mills_and_1000G_gold_standard.indels.b37.sites.vcf


▾ Reference or TAR with BWA reference indices * ⓘ  Change selection
human_g1k_v37_decoy.fasta.tar

▾ SnpEff Database * ⓘ  Change selection
snpEff_v4_3_GRCh37.75.zip


▾ Target BED * ⓘ  Change selection
exome_targets.b37.sorted.bed

▾ dbsnp * ⓘ  Change selection
 dbsnp_137.b37.vcf

App Settings

 Edit parameters

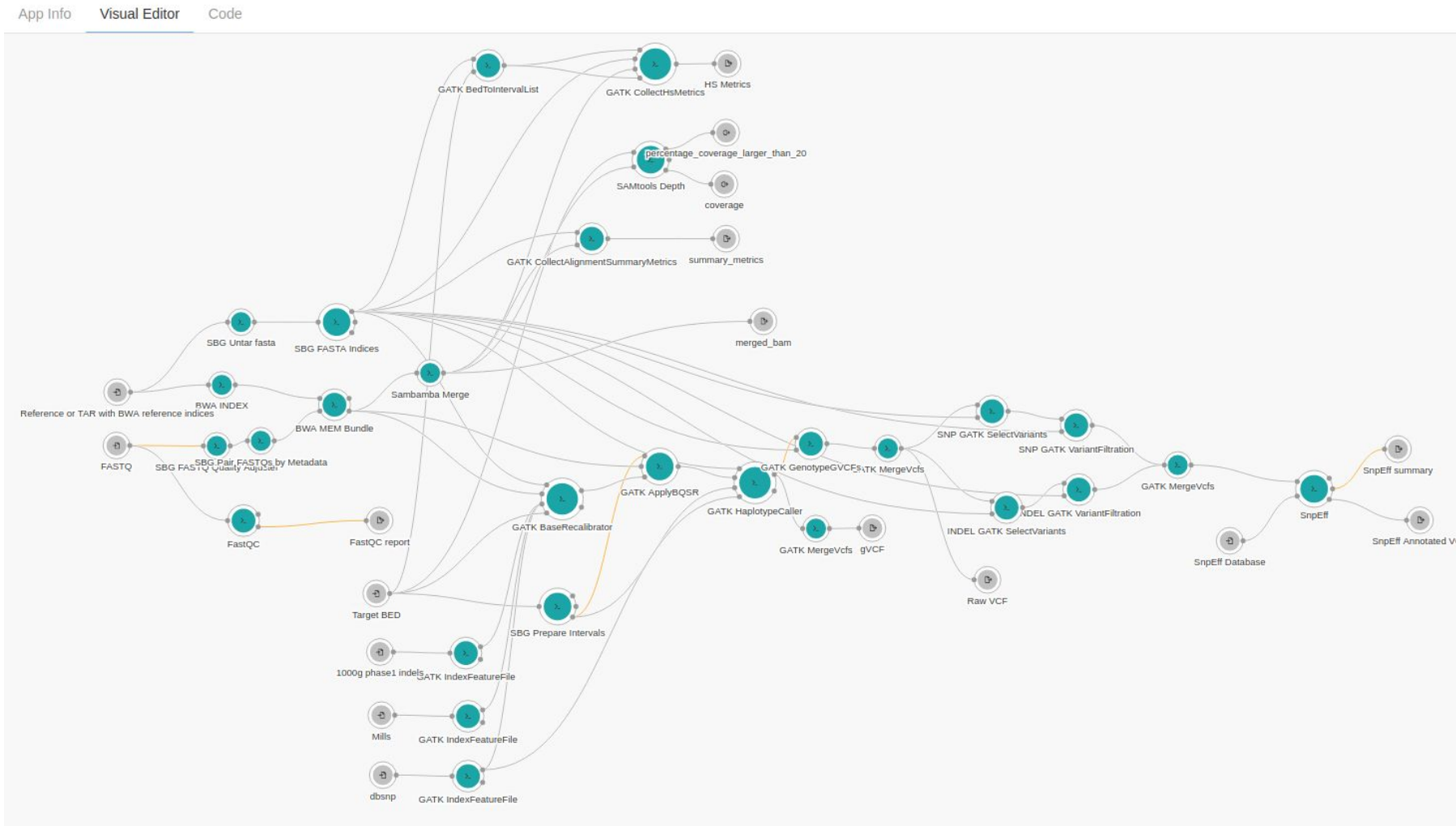
Show editable ▾

▾ SnpEff (#SnpEff)
Assembly (genome version) * ⓘ
GRCh37.75 

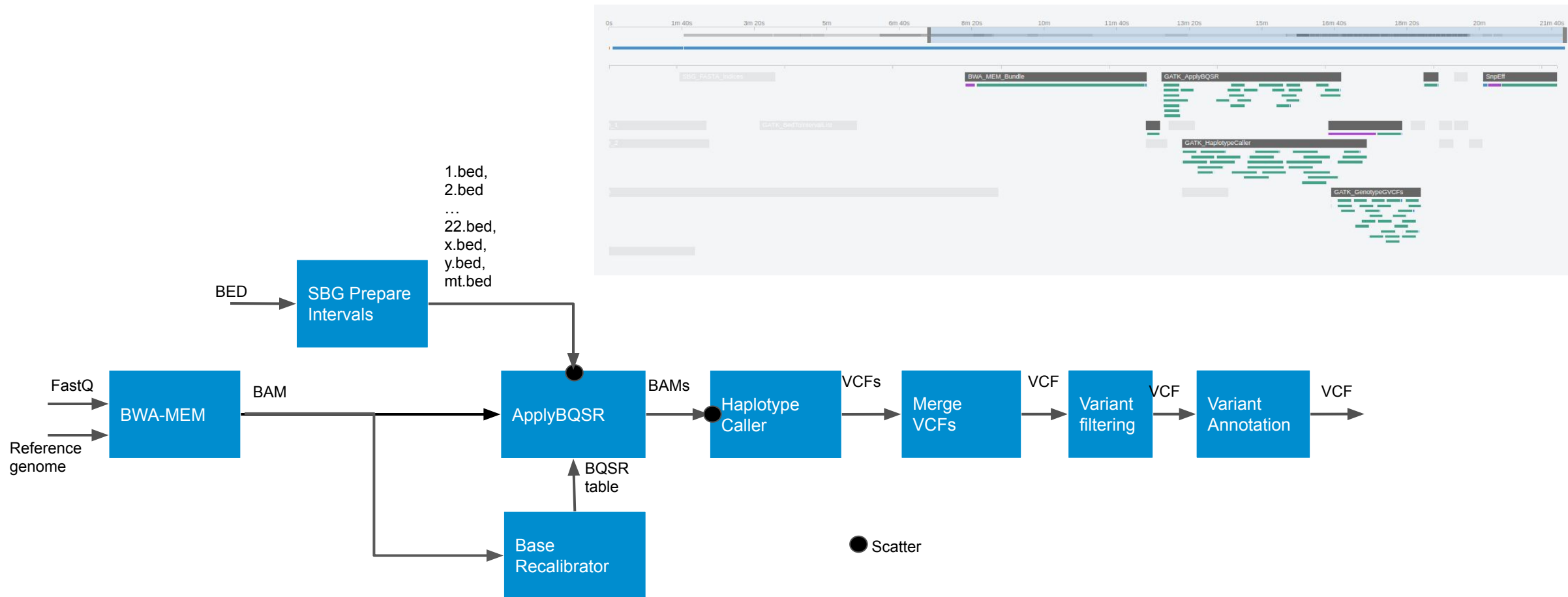
Outputs

FastQC report ⓘ	No value
HS Metrics ⓘ	No value
Raw VCF ⓘ	No value
SnpEff Annotated VCF ⓘ	No value
SnpEff summary ⓘ	No value
coverage	No value
gVCF ⓘ	No value
merged_bam ⓘ	No value
percentage_coverage_larger_than_20	No value
summary_metrics ⓘ	No value

...with real analysis!



...with real analysis!



Whole exome sequencing execution

DashboardFilesAppsTasks

Whole Exome Analysis - Connecting to your own cloud storage ⓘ

Interactive AnalysisNotes

COMPLETED

Whole Exome Sequencing - BWA + GATK 4.0 (with Metrics) run - 04-01-19 14:53:23: sample_id: TCRBOA7-T

Get supportView stats & logs

Executed on Apr. 1, 2019 16:53 by external-demos/himanshu.sharma

Spot Instances: On ⓘMemoization: On ⓘPrice: \$0.05 ⓘRefundView refundsDuration: 21 minutes ⓘ

App: Whole Exome Sequencing - BWA + GATK 4.0 (with Metrics) - Revision: 1

Inputs ⓘ

1000g phase1 indels ⓘ ⓘ
1000G_phase1.indels.b37.vcf

FASTQ ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read2.fastq
TCRBOA7-T-WEX-TEST.read1.fastq

Mills ⓘ ⓘ
Mills_and_1000G_gold_standard.indels.b37.sites.vcf

Reference or TAR with BWA reference indices ⓘ ⓘ
human_g1k_v37_decoy.fasta.tar

SnpEff Database ⓘ ⓘ
snpEff_v4_3_GRCh37.75.zip

Target BED ⓘ ⓘ
exome_targets.b37.sorted.bed

dbsnp ⓘ ⓘ
dbsnp_137.b37.vcf

App Settings

Show non-default ▾

SnpEff (#SnpEff)
Assembly (genome version) ⓘ
GRCh37.75

Outputs ⓘ

Aligned Reads Bwa mem ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.bam

Alignment Summary Metrics ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.summary_metrics.txt

FastQC report ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read2_fastqc.html
TCRBOA7-T-WEX-TEST.read1_fastqc.html

HS Metrics ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.txt

Raw VCF ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.vcf

SnpEff Annotated VCF ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.snpEff_annotated.vcf

SnpEff summary ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.html

coverage44

gVCF ⓘ ⓘ
TCRBOA7-T-WEX-TEST.read.g.vcf

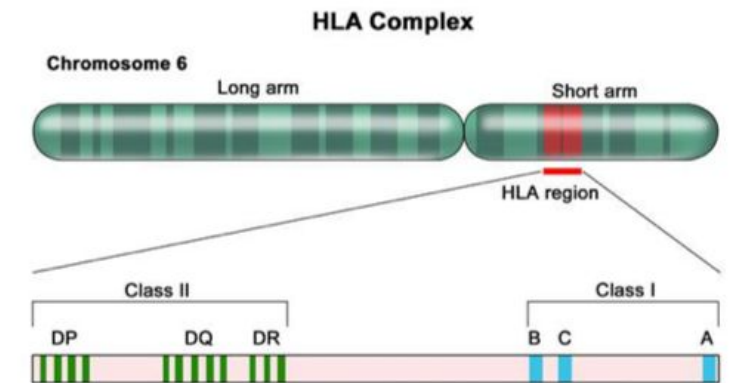
percentage_coverage_larger_than_2072.53%

Exercise 1: Wrap FastQC tool

- Complete the tutorial
- Add cgc user pedjaoetf to the project
- Send the link to the executed task at your CGC project to pedjao@etf.bg.ac.rs
- Include name, surname and number of index
- E-mail subject should be “GI2023_DZ1”
- Do it before the next lesson
- 10 (easy) points :)

HLA Typing

- The HLA gene family provides instructions for making a group of related proteins known as the human leukocyte antigen (HLA) complex.
- The HLA complex helps the immune system distinguish the body's proteins from proteins made by foreign invaders such as viruses and bacteria.
- HLA typing has been widely used for reducing the risk of organ rejection
- Specific HLA variants are associated with both autoimmune (e.g. type 1 diabetes, rheumatoid arthritis) and infectious (e.g. HIV, Hepatitis C) diseases



HLA Typing

External demos

Projects

Data

Public Apps

Public projects

Automations

Developer

Staff

Dashboard

Files

Apps

Tasks

Precision HLA typing - Testing Optitype genotyping tool

Interactive Analysis

Notes

COMPLETED

OptiType adjusted run - 03-25-19 16:00:36

Get support

View stats & logs

Executed on Mar. 25, 2019 17:00 by external-demos/himanshu.sharma_demo

Spot Instances: OnMemoization: OffPrice: \$0.10RefundView refundsDuration: 22 minutes

App: OptiType adjusted - Revision: 0

Inputs

Input files

SRR081250_1.fastq

SRR081250_2.fastq

App Settings

Sequencing data

--dna

Show non-default

Outputs

BAM files

SRR081250_2.bam

SRR081250_1.bam

Config output

_1_config.ini

Coverage plot

SRR081250.coverage_plot.pdf

Full HLA types

SRR081250.result_type.tsv

HLA Types

HLA results

SRR081250.result.tsv

Log file

SRR081250.command_log.bt

HLA-A*26:01

HLA-A*68:03

HLA-B*51:01



HLA-B*15:10

HLA-C*16:01

HLA-C*04:01

HLA

Public App Gallery

 Projects ▾ Data ▾ Public Apps Public projects ▾ Automations Developer Staff ▾  vladimirk

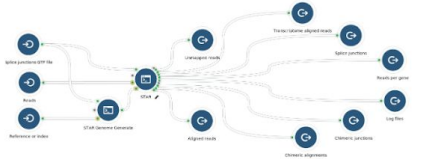
Public apps for your data analysis

Browse 407 publicly available Common Workflow Language workflows and tools to enable reproducible bioinformatics.

or [Explore all apps](#)

Featured Apps

RNA-Seq Alignment - STAR 2.5.4b





Toolkit: STAR 2.5.4b


This workflow performs the first step of RNA-seq analysis - alignment to a reference genome and transcriptome.

[ALIGNMENT](#) [RNA](#)

[Copy](#) [Run](#)

 **Whole Exome Sequencing - BWA + GATK 4.0 (with Metrics)**

 **Whole Genome Sequencing - BWA + GATK 4.0 (with Metrics)**

 **Multi-instance Whole Genome Sequencing GATK4.0 workflow**

?

Runnable from the command line
Suitable for local testing and development

rabix.io

<https://github.com/rabix/bunny>

RABIX: AN OPEN-SOURCE WORKFLOW EXECUTOR SUPPORTING RECOMPUTABILITY AND INTEROPERABILITY OF WORKFLOW DESCRIPTIONS.

Kaushik G¹, Ivkovic S, Simonovic J, Tijanic N, Davis-Dusenbery B, Kural D.

Author information

1 Seven Bridges Genomics, 1 Main Street, Cambridge, MA 02140, USA*Corresponding author., gaurav@sevenbridges.com.

Abstract

As biomedical data has become increasingly easy to generate in large quantities, the methods used to analyze it have proliferated rapidly. Reproducible and reusable methods are required to learn from large volumes of data reliably. To address this issue, numerous groups have developed workflow specifications or execution engines, which provide a framework with which to perform a sequence of analyses. One such specification is the Common Workflow Language, an emerging standard which provides a robust and flexible framework for describing data analysis workflows. RABIX is a workflow executor that allows for easy integration of existing workflow specifications and the purposes of this paper is to describe its architecture and implementation.

bioRxiv preprint doi: <https://doi.org/10.1101/250113>; this version posted May 10, 2018. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

rabix / bunny

Unwatch 27 Star 71 Fork 26

Code Issues 72 Pull requests 0 Actions Projects 1 Wiki Security Insights

Releases Tags

on Jun 29, 2018 Show 2 newer tags

v1.0.5

milos-ljubinkovic released this on Mar 22, 2018 · 10 commits to master since this release

- Some performance improvements
- Fixes for some edge case bugs in docker command line building
- Some TES S3 storage improvements
- Dedicated composer integration via special logging mode

Assets 4

rabix-1.0.5-TES.tar.gz	41.1 MB
rabix-1.0.5.tar.gz	31.2 MB
Source code (zip)	
Source code (tar.gz)	

Local executor

Install [docker](#) download and unpack [rabix](#)

./rabix -b ./ grep.cwl.json inputs.json

ll grep-2020-02-11-155503.852/root/

```
-rw-r--r-- 1 vladimirk staff 100 Feb 11 15:55 cmd.log
-rw-r--r-- 1 vladimirk staff 550 Feb 11 15:55 cwl.output.json
-rw-r--r-- 1 vladimirk staff 27 Feb 11 15:55 out.txt
```

cat grep-2020-02-11-155503.852/root/cwl.output.json

```
{
  "output" : {
    "basename" : "out.txt",
    "checksum" : "sha1$0a3e8ce4ad3bcd5db0804f28752499adfe2ca5d1",
    "class" : "File",
    "dirname" : "grep-2020-02-11-155503.852/root",
    "location" : "grep-2020-02-11-155503.852/root/out.txt",
    "nameext" : ".txt",
    "nameroot" : "out",
    "path" : "grep-2020-02-11-155503.852/root/out.txt",
    "size" : 27
  }
}
```

cat grep-2020-02-11-155503.852/root/out.txt

```
ACTGA
GAGAGAGA
GA
GGGAAAGA
```

cat grep-2020-02-11-155503.852/root/cmd.log

```
grep GA dummy.fasta > out.txt
```

grep.json

```
{
  "class": "CommandLineTool",
  "cwlVersion": "v1.0",
  "$namespaces": {"sbg": "https://sevenbridges.com"},
  "baseCommand": ["grep"],
  "inputs": [
    { "id": "pattern",
      "type": "string",
      "inputBinding": {"position": 1},
      "label": "Pattern"},
    { "id": "input",
      "type": "File",
      "inputBinding": {"position": 2}}
  ],
  "outputs": [
    { "id": "output",
      "type": "File?",
      "outputBinding": {
        "glob": "*.txt"}}
  ],
  "arguments": [
    {"position": 3, "prefix": "",
      "valueFrom": "> out.txt"}
  ],
  "requirements": [
    {"class": "ShellCommandRequirement"},
    {"class": "DockerRequirement", "dockerPull": ubuntu:14.04"}
  ]
}
```

inputs.json

```
{
  "input" : {
    "path" : "dummy.fasta",
    "class" : "File"
  },
  "pattern" : "GA"
}
```

4. Jupyter Notebook bioinformatic analysis on the cloud

Interactive analysis

Projects ▾ Data ▾ Public Apps Public projects ▾ Automations Developer Staff ▾

vladmirk ▾

Dashboard Files Apps Tasks

Copy of Data Cruncher Interactive Analyses ⓘ

Interactive Analysis Settings Notes

Search 🔍

Create new analysis

Analysis name	Created by	Environment	Created on	Status	Actions
Ballgown Interactive Analysis	vladmirk	JupyterLab	Oct. 24, 2019 17:48	DRAFT	
ChIP-seq Interactive Analysis	vladmirk	JupyterLab	Oct. 24, 2019 17:48	DRAFT	
VCF Visualization Interactive Analysis	vladmirk	JupyterLab	Oct. 24, 2019 17:48	DRAFT	
Microbiome Differential Abundance Analysis	vladmirk	JupyterLab	Oct. 24, 2019 17:48	SAVED	
Structural Variant Interactive Analysis	vladmirk	JupyterLab			

Find your way around the Data Cruncher workspace:

- Project files at your fingertips. These files can be accessed and used from the `/sbgenomics/project-files/` directory.
- The work takes place in the workspace. Files added to the analysis or produced by the analysis are located in the `/sbgenomics/workspace/` directory.
- Make use of the outputs. The `/sbgenomics/output-files/` directory is where you want to store the files you would like to save in Project Files.

Run python/R Jupyter Notebook on the cloud
Further process outputs from bioinformatics tasks

```
pattern = 'ACCT'
with
open('/sbgenomics/project-files/PhiX_genome.fast
a', 'r') as myfile:
    data=myfile.readlines()
data = ''.join(data).replace('\n', '')
cnt = 0
for i in range(0, len(data) - len(pattern)):
    if data[i:i+len(pattern)] == pattern:
        cnt += 1
    print(cnt, i)
```

Projects
 Data
 Public Apps
 Public projects
 Tutorials
 Feedback
 Staff
 Logout

Dashboard
 Files
 Apps
 Tasks

Copy of Data Cruncher Interactive Analyses

Interactive Analyses
 Settings
 Notes

SAVED

Microbiome Differential Abundance Analysis

Session started on Oct 26, 2023 17:40 by shadkhs
 Environment: JupyterLab (Price: \$0.00) | Duration: 14 hours, 18 minutes | [View Session](#)

Files
 Settings

Analysis files:
 Unsaved.py2
 Microbiome_Differential_Abundance_An...

Produced by this analysis
 No files

Microbiome Differential Abundance Analysis

The goal of this analysis is to detect microbes that are differentially abundant between two pre-determined classes of samples. This experimental design is applicable to [low-central research studies and other omics](#) which there is a prior knowledge about the existing microbiological conditions.

The required input files are:

- counts: table - containing mapped read counts per OTU for all samples
- taxonomy: table - containing taxonomy information for each OTU
- metadata: table - containing clinical data for each sample

Counts and taxonomy tables are default outputs of [MetaPhlAn](#), [Centrifuge](#) and [QIIME2](#) metagenomic workflows available on the platform, while the [BIOM](#) file format containing these tables is also supported. After loading required files, the available source code enables visualization of relative abundances of most abundant taxa and differential abundance analysis using the `Tiltest` or `elphid` functions of the [microbiomeR](#) R package.

We used the [Jupyter](#) Interactive visualization library which allows building of a variety of complex plots through simple commands.

Load `rrpy2`, the high-level interface designed to facilitate the use of R code by the Python kernel. This enables usage of both Python and R code within the same notebook file.

```
In [1]: %load_ext rrpy2.python
        %cpout warnings
        warnings.filterwarnings("ignore")
```

Load the required R packages.



```
In [2]: %R
        packages <- c("metagenomeSeq", "ggplot2", "plyr", "scales", "reshape2", "biomformat")
        invisible(suppressMessages(library(packages, require, character.only = TRUE)))
```

Set input file variables


Here you can set variables for counts, taxonomy, table and metadata table that will be loaded as metagenomeSeq objects. Navigate to the left side of the screen and enter the **Project Files** section, where you can select files from the current project. By clicking on a selected file, its path will be copied to the clipboard. Paste the path as a string in the cell below.

```
In [3]: %R
        counts_table = "/jsgenomics/project-files/counts_table.mdsl6s.tsv"
        taxonomy_table = "/jsgenomics/project-files/taxonomy_table.mdsl6s.tsv"
        metadata_table = "/jsgenomics/project-files/metadata_table.mdsl6s.tsv"
```

Data studio - Interactive analysis

 **SAVED** **VCF Filtering** 

Session started on Dec. 10, 2020 13:30 by **sbguser**
Environment: JupyterLab (SB Data Science - Python 3.8, R 3.6) | Price: \$0.07 | Duration: 8 minutes [View Sessions](#)

 Copy  Start

Files Settings

Workspace

-  REALIGNED_TUMOR.bam.filtered.vcf
-  notes.txt
-  variants.dat
-  VCF_Filtering.ipynb

Outputs

-  REALIGNED_TUMOR.bam.filtered.vcf

```
In [5]: vcf_column_names = ["CHROM", "POS", "ID", "REF", "ALT", "QUAL", "FILTER",  
"INFO", "FORMAT", "NORMAL", "TUMOR"]  
  
def read_vcf_df(vcf_path, vcf_column_names):  
    return pd.read_csv(vcf_path, comment="#", header=None, names = vcf_col  
umn_names , sep="\t")  
  
def ad_dp_calc(x):  
    ref = x[0]  
    alts = x[1].split(',')  
    gt = x[2].split(':')  
    if len(gt) == 1:  
        return 0  
    dp = float(gt[0].replace(",","."))  
    a = float(gt[4].replace(",","."))  
    c = float(gt[5].replace(",","."))  
    g = float(gt[6].replace(",","."))  
    t = float(gt[7].replace(",","."))  
    ad = 0  
    for alt in alts:
```

Thank you!

Questions?



@vladimir_bio