



The Jackson
Laboratory



Science and Technology
Consulting LLC

Dry Bench Skills for the Researcher

Day 1

Introduction to Command line in the JupyterLab Interface

2020 Dec 2

What are we doing here today?

We are taking you through the steps to create reproducible science



DATA

INSIGHT & PUBLICATION

DATA PROCESSING - REUSE & DEVELOPMENT

DATA PROCESSING - EXECUTION

RESULTS

Code
versioning

Container-
ization

Workflow
Deployment

High
performance
computing

Cloud
infrastructure

Reasoning

Reuse code
Modify code
Share &
Collaborate

Containerize
separate
processes

Stitch them
together into
a unified
workflow to
accomplish
the analysis

Provision &
manage a HPC
cluster to analyse
the large volumes
of genomics data

Provision & scale
analysis on cloud to
speed up analysis &
access compute
resources on
demand

Manually combine results
from data analysis &
make decisions based on
experience. Collaborate.



Dry Bench Skills Roadmap

Intro to Command
line in the
JupyterLab
Interface

Day 1

Bash skills intro
R
JupyterLab
Volcano plot
Zenodo

Git and
Collaboration

Day 2

Git
GitHub for Teamwork
JupyterLab
Volcano plot

Introduction Conda
and Docker

Day 3

Conda
Docker
Prepping for Nextflow

Nextflow

Day 4

Nextflow

Putting it all
together
RNASeq
Nextflow &
Jupyter Notebook

Day 5




Best Practices
RNASeq workflow

Dry Bench Skills Agenda

Intro to command line in the JupyterLab interface



Agenda for the day:

Time	Programme
12.00 - 12.10	<i>Welcome Address and Presentation of workshop agenda</i>
12.10 - 12.50	 Useful commands for the command line and introduction to the JupyterLab interface terminal
12.50 - 13.00	 Short break
13.00 - 14.00:	 Introduction to R. Useful commands to run R code and install required dependencies.

Science and Technology
Consulting LLC



Christina Chatzipantsiou
Bioinformatician
christina@lifebit.ai

Dr. Anne Deslattes Mays
adeslat@scitechcon.org





HM Government
G-Cloud
Supplier

• Thank you

...from everyone at



Science and Technology
Consulting LLC



Visit us at lifebit.ai

Appendix

Nextflow for reproducible hardware agnostic results

Nextflow enables reproducible computational workflows

To the Editor:

The increasing complexity of readouts for omics analyses goes hand-in-hand with concerns about the reproducibility of experiments that analyze 'big data'¹⁻³. When analyzing very large data sets, the main source of computational irreproducibility arises from a lack of good practice pertaining to software and database usage⁴⁻⁶. Small variations across computational platforms also contribute to computational irreproducibility by producing numerical instability⁷, which is especially relevant to high-performance computational (HPC) environments that are routinely used for omics analyses⁸. We present a solution to this instability named Nextflow, a workflow management system that uses Docker technology for the multi-scale handling of containerized computation.

In silico workflow management systems are an integral part of large-scale biological analyses. These systems enable the rapid prototyping and deployment of pipelines that combine complementary software packages. In genomics the simplest pipelines, such as Kallisto and Sleuth⁹, combine an RNA-seq quantification method with a differential expression module (Supplementary Fig. 1). Complexity rapidly increases when all aspects of a given analysis are included. For example,

the Sanger Companion pipeline¹⁰ bundles 39 independent software tools and libraries into a genome annotation suite. Handling such a large number of software packages, some of which may be incompatible, is a challenge. The conflicting requirements of frequent software updates and maintaining the reproducibility of original results provide another unwelcome wrinkle. Together with these problems, high-throughput usage of complex pipelines can also be burdened by the hundreds of intermediate files often produced by individual tools. Hardware fluctuations in these types of pipelines, combined with poor error handling, could result in considerable readout instability.

Nextflow (<http://nextflow.io>; Supplementary Methods, Supplementary Note and Supplementary Code 1) is designed to address numerical instability, efficient parallel execution, error tolerance, execution provenance and traceability. It is a domain-specific language that enables rapid pipeline development through the adaptation of existing pipelines written in any scripting language.

We present a qualitative comparison between Nextflow and other similar tools in Table 1 (ref. 11). We found that multi-scale containerization, which makes it possible to

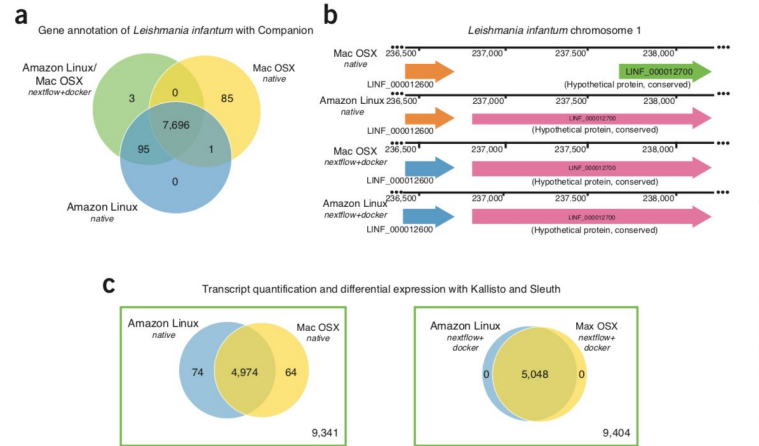


Figure 1 Nextflow enables stable analyses on different platforms. **(a)** *Leishmania infantum* clone JPCM5 genome annotation was carried out using either a native or a dockerized (Debian Linux) version of the Companion eukaryotic annotation pipeline. The native and dockerized versions were run on both Mac OSX and Amazon Linux platforms. The Venn diagram, plotted using the annotated genes, reveals small discrepancies when comparing the genomic coordinates of predicted coding genes, non-coding RNAs, and pseudogenes. **(b)** Some disparities included the annotation of the same genes at different genomic coordinates. Completely identical annotations were obtained when using the dockerized version on either the Mac OSX or Amazon Linux platforms. **(c)** Comparison of the Kallisto and Sleuth pipelines, applied to find differentially expressed genes (q -value < 0.01) in an RNA-seq experiment, using data from human lung fibroblasts, revealed differences when carried out on either the Mac OSX or the Amazon Linux platform. Both platforms produced identical readouts when deploying the dockerized version of the pipeline. All analyses were carried out at least twice and checked for numerical stability.