labibi Documentation

Release 1.0

C. Titus Brown

September 25, 2015

This workshop was given on Sep 24th and 25th, 2015, by C. Titus Brown and Ben Johnson. See the workshop organization page for more information, or contact Titus directly.

CHAPTER 1

Welcome!

1.1 1. Learning goals

For you:

- get a first (or second) look at tools;
- gain some experience in the basic command line;
- get 80% of way to a complete analysis of some data;

For us:

- what are the on campus needs? who are the on-campus people?

1.2 2. Safe space and code of conduct

This is intended to be a safe and friendly place for learning!

Please see the Software Carpentry workshop Code of Conduct: http://software-carpentry.org/conduct.html

In particular, please ask questions, because I guarantee you that your question will help others!

1.3 3. Instructor introductions

Titus Brown - prof here at UC Davis in the School of Vet Med.

Ben Johnson - graduate student at Michigan State University.

1.4 4. Amazon and cloud computing - why?!

- simplifies software installation;
- can be used for bigger analyses quite easily;
- good for “burst” capacity (just got a data set!)
- accessible everywhere;
- they give us $100 gift certificates... cue handing them out!

1.5 5. Sticky notes and how they work... + Minute Cards

Basic rules:

- no sticky note - “working on it”
- green sticky note - “all is well”
- red sticky note - “need help!”

Place the sticky notes where we can see them from the back of the room – e.g. on the back of your laptop.

At the end of each session (coffee break, lunch, end of day) please write down on an index card **one thing you learned** and **one thing you’re still confused about.**

1.6 6. Warning:

The bit that isn’t going to work is at the beginning. This is unavoidable. To reward you, we have a coffee break at 10:30am...

—

Next: Getting started with Amazon EC2

CHAPTER 2

Getting started with Amazon EC2

Summary:

- Go to http://aws.amazon.com/, log in, then “EC2” (upper left);
- Select “Launch instance”;
- Select “Ubuntu 14.04” from the list;
- Select “m3.xlarge” from the list (towards bottom of “General purpose”);
- Click “Review and launch”
- Select “Launch”;
- If your first time through, create a key pair; otherwise select existing;
- Click “launch instance”

2.1 Start up an EC2 instance

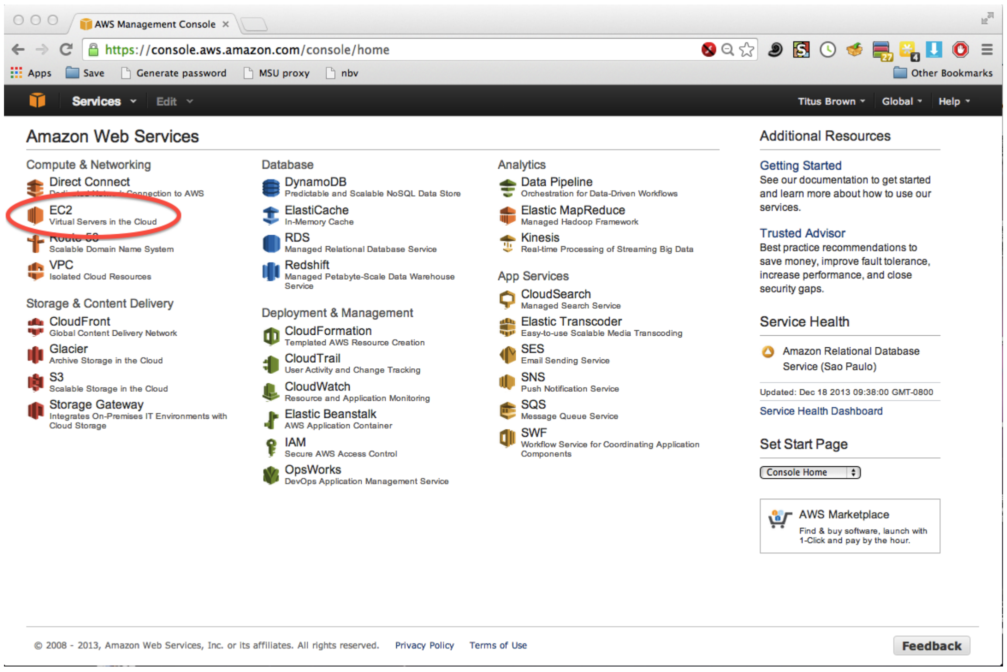
2.1.1 Log in

Go to ‘https://aws.amazon.com‘ in a Web browser.

Select ‘My Account/Console’ menu option ‘AWS Management Console.”

Log in with your username & password.

Click on EC2 (upper left).



Amazon Web Services

Compute & Networking
Direct Connect — Dedicated Network Connection to AWS
EC2 — Virtual Servers in the Cloud
Route 53 — Scalable Domain Name System
VPC — Isolated Cloud Resources

Storage & Content Delivery
CloudFront — Global Content Delivery Network
Glacier — Archive Storage in the Cloud
S3 — Scalable Storage in the Cloud
Storage Gateway — Integrates On-Premises IT Environments with Cloud Storage

Database
DynamoDB — Predictable and Scalable NoSQL Data Store
ElastiCache — In-Memory Cache
RDS — Managed Relational Database Service
Redshift — Managed Petabyte-Scale Data Warehouse Service

Deployment & Management
CloudFormation — Templated AWS Resource Creation
CloudTrail — User Activity and Change Tracking
CloudWatch — Resource and Application Monitoring
Elastic Beanstalk — AWS Application Container
IAM — Secure AWS Access Control
OpsWorks — DevOps Application Management Service

Analytics
Data Pipeline — Orchestration for Data-Driven Workflows
Elastic MapReduce — Managed Hadoop Framework
Kinesis — Real-time Processing of Streaming Big Data

App Services
CloudSearch — Managed Search Service
Elastic Transcoder — Easy-to-use Scalable Media Transcoding
SES — Email Sending Service
SNS — Push Notification Service
SQS — Message Queue Service
SWF — Workflow Service for Coordinating Application Components

Additional Resources

Getting Started — See our documentation to get started and learn more about how to use our services.

Trusted Advisor — Best practice recommendations to save money, improve fault tolerance, increase performance, and close security gaps.

Service Health
Amazon Relational Database Service (Sao Paulo)
Updated: Dec 18 2013 09:38:00 GMT-0800
Service Health Dashboard

Set Start Page
Console Home

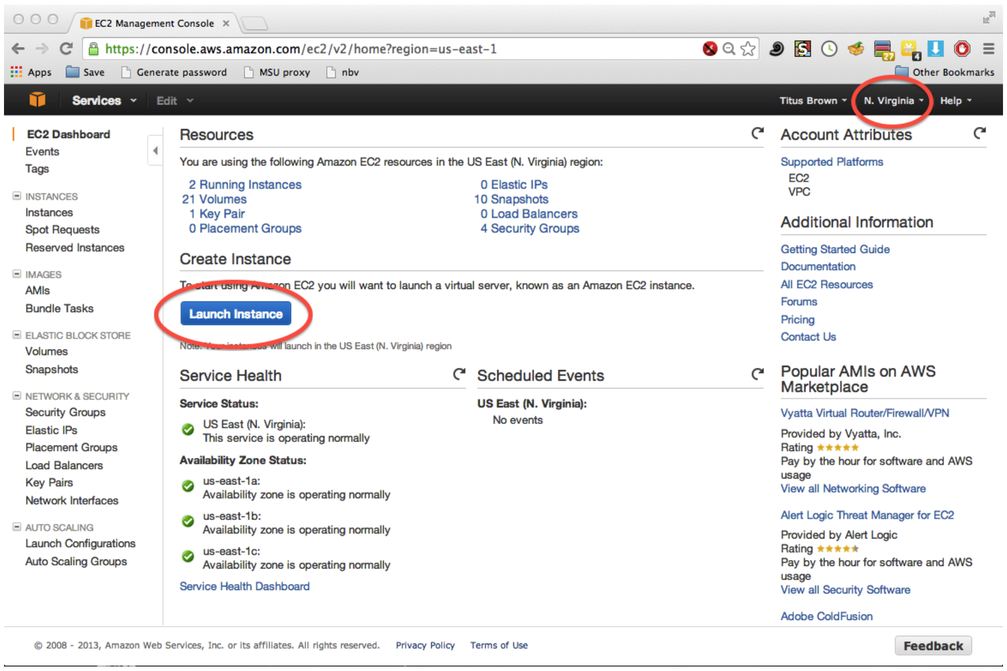
AWS Marketplace — Find & buy software, launch with 1-Click and pay by the hour.

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use
Feedback

2.1.2 Select your zone

Many of the resources that we use are hosted by Amazon on the East coast. Make sure that your dashboard has 'N. Virginia' on the upper right.

Then click on Launch Instance.



Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) region:

2 Running Instances
21 Volumes
1 Key Pair
0 Placement Groups
0 Elastic IPs
10 Snapshots
0 Load Balancers
4 Security Groups

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the US East (N. Virginia) region

Service Health

Service Status:
US East (N. Virginia): This service is operating normally

Availability Zone Status:
us-east-1a: Availability zone is operating normally
us-east-1b: Availability zone is operating normally
us-east-1c: Availability zone is operating normally

Service Health Dashboard

Scheduled Events

US East (N. Virginia):
No events

Account Attributes
Supported Platforms
EC2
VPC

Additional Information
Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

Popular AMIs on AWS Marketplace

Vyatta Virtual Router/Firewall/VPN
Provided by Vyatta, Inc.
Rating ★★★★★
Pay by the hour for software and AWS usage
View all Networking Software

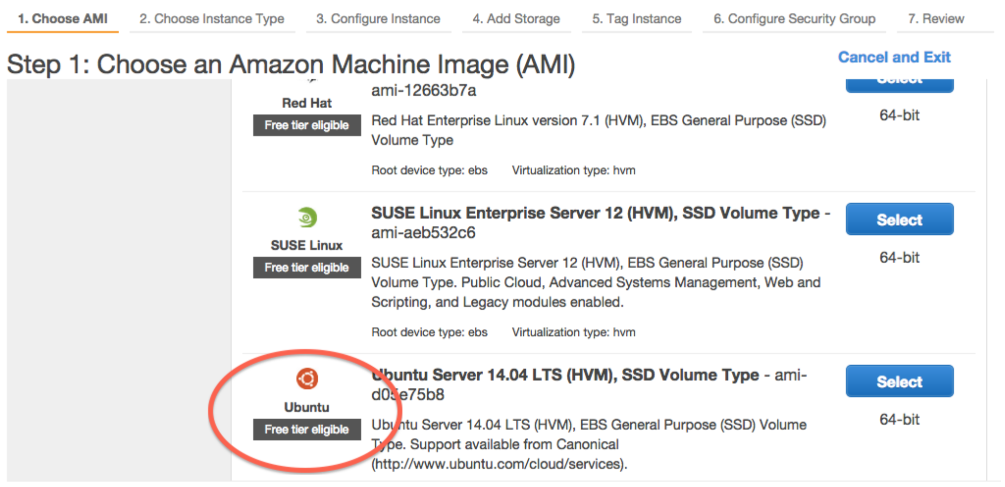
Alert Logic Threat Manager for EC2
Provided by Alert Logic
Rating ★★★★★
Pay by the hour for software and AWS usage
View all Security Software

Adobe ColdFusion

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use
Feedback

2.1.3 Select the machine operating system to boot

Find the "Ubuntu Server 14.04" image in the first list to show up.



1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Tag Instance | 6. Configure Security Group | 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

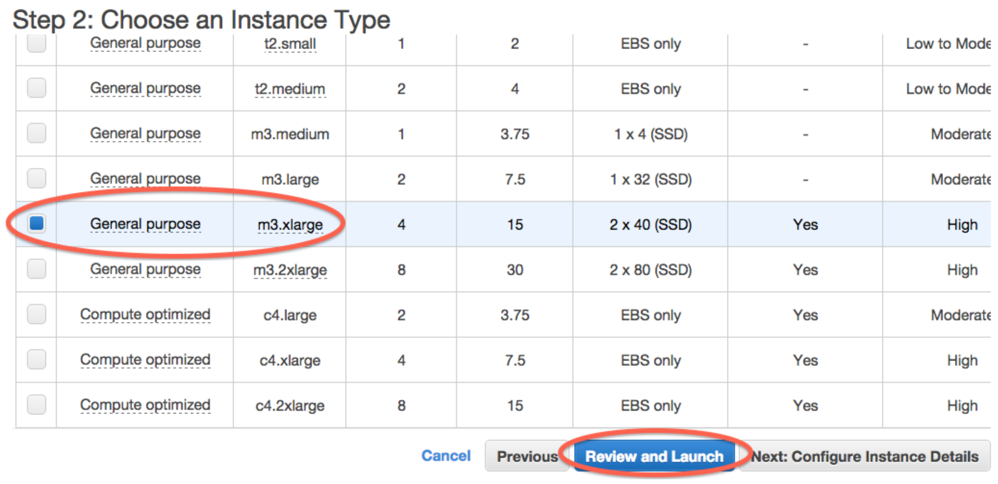
Red Hat
Free tier eligible
ami-12663b7a
Red Hat Enterprise Linux version 7.1 (HVM), EBS General Purpose (SSD) Volume Type
Root device type: ebs Virtualization type: hvm
64-bit

SUSE Linux
Free tier eligible
SUSE Linux Enterprise Server 12 (HVM), SSD Volume Type - ami-aeb532c6
SUSE Linux Enterprise Server 12 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.
Root device type: ebs Virtualization type: hvm
Select
64-bit

Ubuntu
Free tier eligible
Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d05e75b8
Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Select
64-bit

2.1.4 Choose the machine size

Select 'General purpose', 'm3.xlarge', and then 'Review and Launch'.

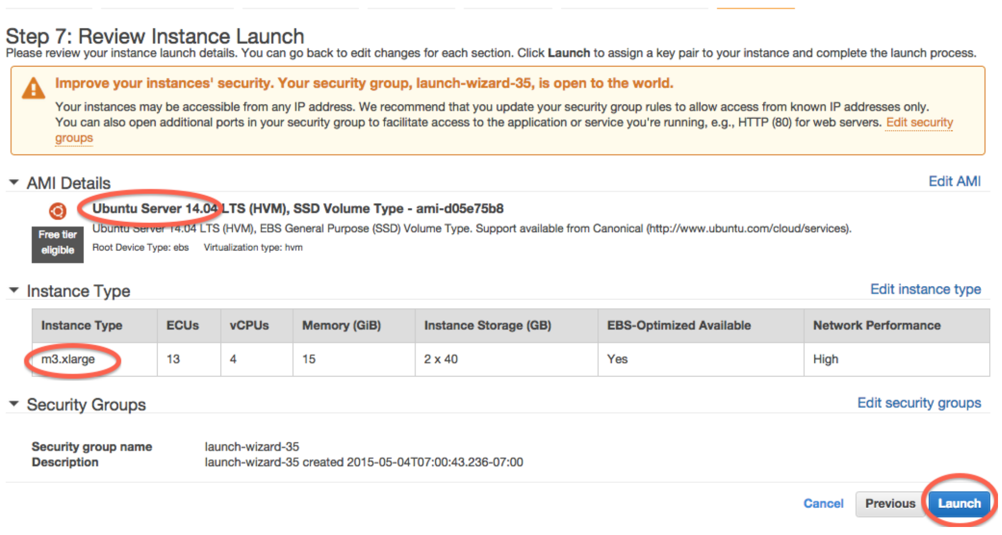


Step 2: Choose an Instance Type

☐	General purpose	t2.small	1	2	EBS only	-	Low to Mode
☐	General purpose	t2.medium	2	4	EBS only	-	Low to Mode
☐	General purpose	m3.medium	1	3.75	1 x 4 (SSD)	-	Moderate
☐	General purpose	m3.large	2	7.5	1 x 32 (SSD)	-	Moderate
☑	General purpose	m3.xlarge	4	15	2 x 40 (SSD)	Yes	High
☐	General purpose	m3.2xlarge	8	30	2 x 80 (SSD)	Yes	High
☐	Compute optimized	c4.large	2	3.75	EBS only	Yes	Moderate
☐	Compute optimized	c4.xlarge	4	7.5	EBS only	Yes	High
☐	Compute optimized	c4.2xlarge	8	15	EBS only	Yes	High

2.1.5 Confirm and launch

Review the details (ignore the warnings!) and click on Launch.



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

Improve your instances' security. Your security group, launch-wizard-35, is open to the world.

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. Edit security groups

AMI Details — Edit AMI

Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d05e75b8

Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Free tier eligible — Root Device Type: ebs Virtualization type: hvm

Instance Type — Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
m3.xlarge	13	4	15	2 x 40	Yes	High

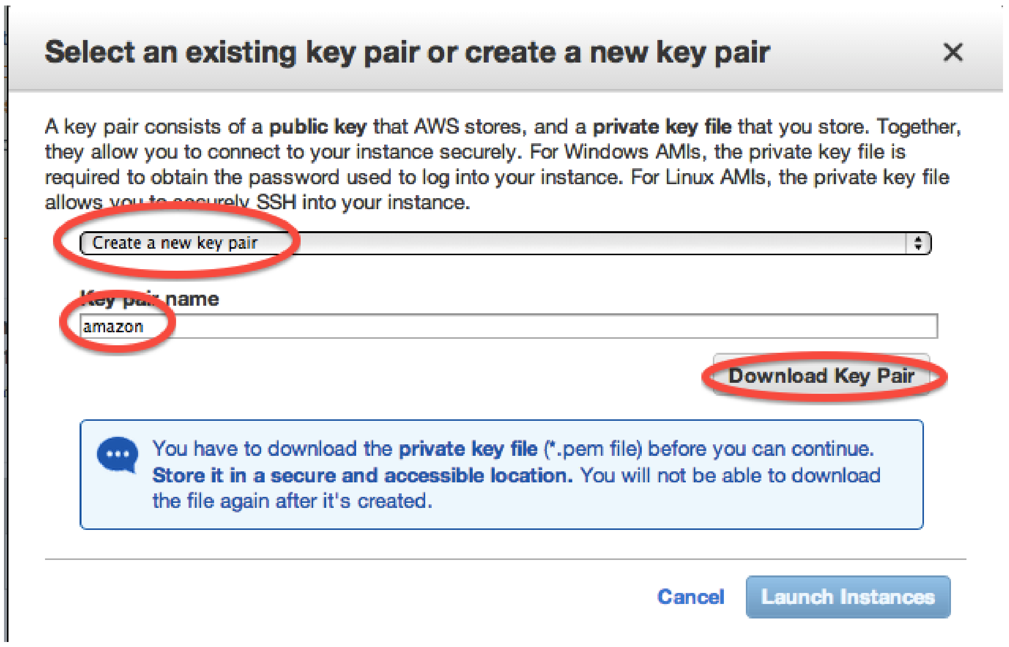
Security Groups — Edit security groups

Security group name	launch-wizard-35
Description	launch-wizard-35 created 2015-05-04T07:00:43.236-07:00

Cancel Previous Launch

2.1.6 (First time through) generate a new key pair

If you don't have any key pairs, enter a key pair name and then download a key pair. Then click Launch Instance.



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Create a new key pair

Key pair name

amazon

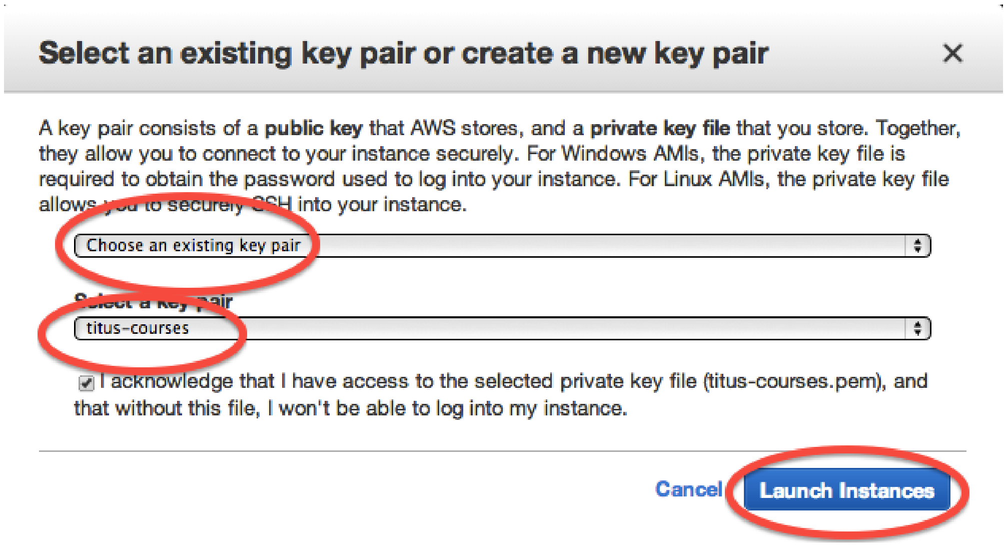
Download Key Pair

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Cancel Launch Instances

2.1.7 (Next times through) select an existing key pair

Select a key pair and click 'Launch'.



Select an existing key pair or create a new key pair

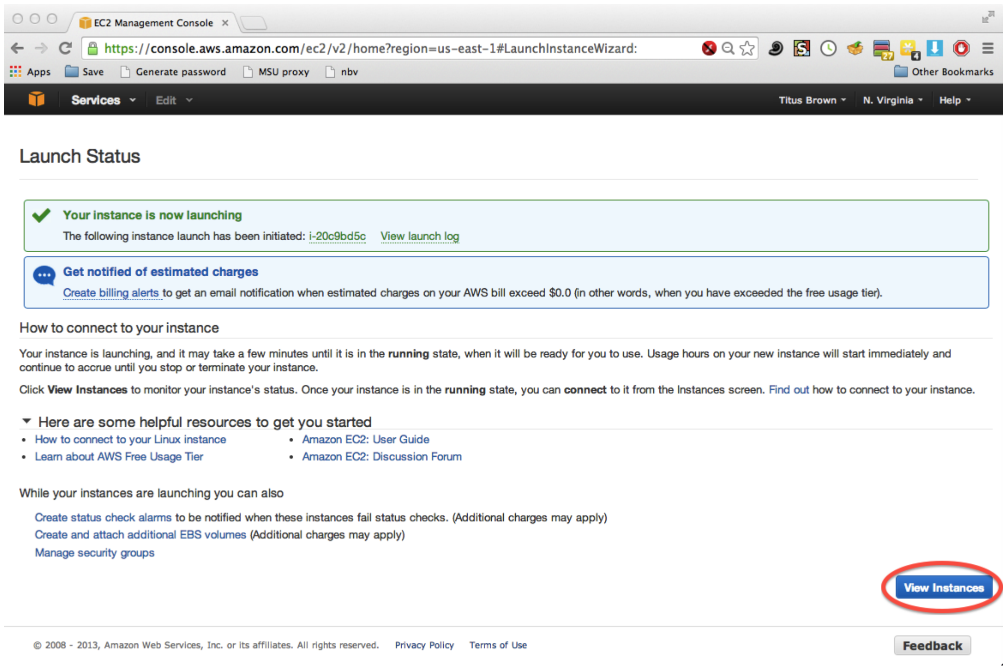
A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Choose an existing key pair

Select a key pair: titus-courses

✓ I acknowledge that I have access to the selected private key file (titus-courses.pem), and that without this file, I won't be able to log into my instance.

2.1.8 Click on View Instances



Launch Status

Your instance is now launching
The following instance launch has been initiated: i-20c9bd5c View launch log

Get notified of estimated charges
Create billing alerts to get an email notification when estimated charges on your AWS bill exceed $0.0 (in other words, when you have exceeded the free usage tier).

How to connect to your instance

Your instance is launching, and it may take a few minutes until it is in the **running** state, when it will be ready for you to use. Usage hours on your new instance will start immediately and continue to accrue until you stop or terminate your instance.

Click **View Instances** to monitor your instance's status. Once your instance is in the **running** state, you can **connect** to it from the Instances screen. Find out how to connect to your instance.

Here are some helpful resources to get you started

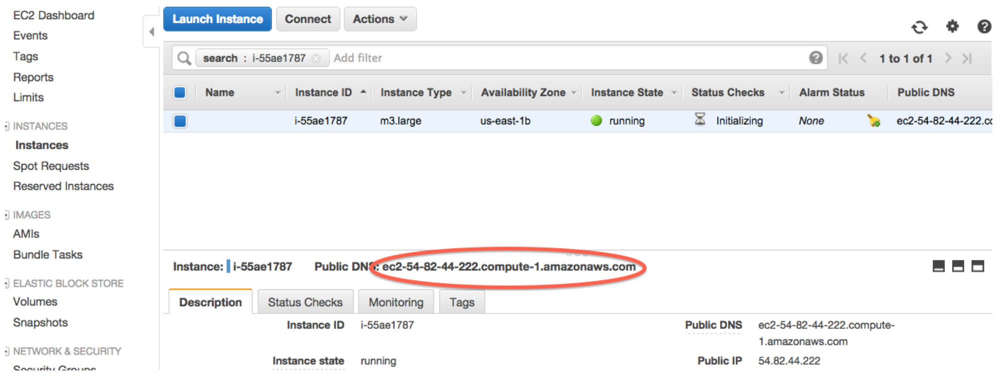
- How to connect to your Linux instance
- Learn about AWS Free Usage Tier
- Amazon EC2: User Guide
- Amazon EC2: Discussion Forum

While your instances are launching you can also

Create status check alarms to be notified when these instances fail status checks. (Additional charges may apply)
Create and attach additional EBS volumes (Additional charges may apply)
Manage security groups

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

2.1.9 Select the public DNS name for later use



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-55ae1787	m3.large	us-east-1b	running	Initializing	None	ec2-54-82-44-222.c

Instance: i-55ae1787 Public DNS: ec2-54-82-44-222.compute-1.amazonaws.com

Instance ID	i-55ae1787	Public DNS	ec2-54-82-44-222.compute-1.amazonaws.com
Instance state	running	Public IP	54.82.44.222

Next steps

Logging into your new instance "in the cloud" (Mac version) or `log-in-with-ssh-win`

2.2 Logging into your new instance "in the cloud" (Windows version)

OK, so you've created a running computer. How do you get to it?

The main thing you'll need is the network name of your new computer. To retrieve this, go to the instance view and click on the instance, and find the "Public DNS". This is the public name of your computer on the Internet.

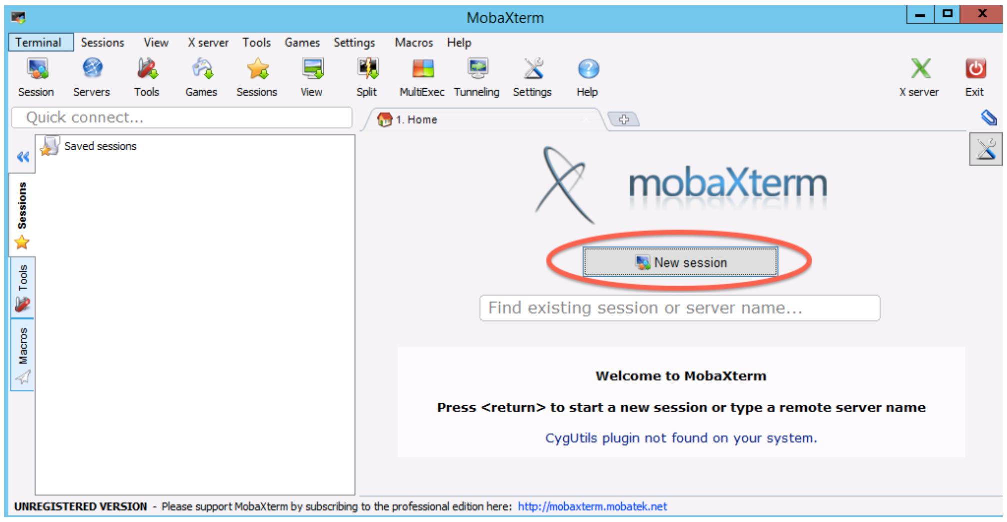
Copy this name, and connect to that computer with ssh under the username 'ubuntu', as follows.

—

2.2.1 Install mobaxterm

First, download mobaxterm and run it.

2.2.2 Start a new session


MobaXterm
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Settings Help X server Exit
Quick connect...
1. Home
Saved sessions
Sessions Tools Macros
mobaXterm
New session
Find existing session or server name...

Welcome to MobaXterm

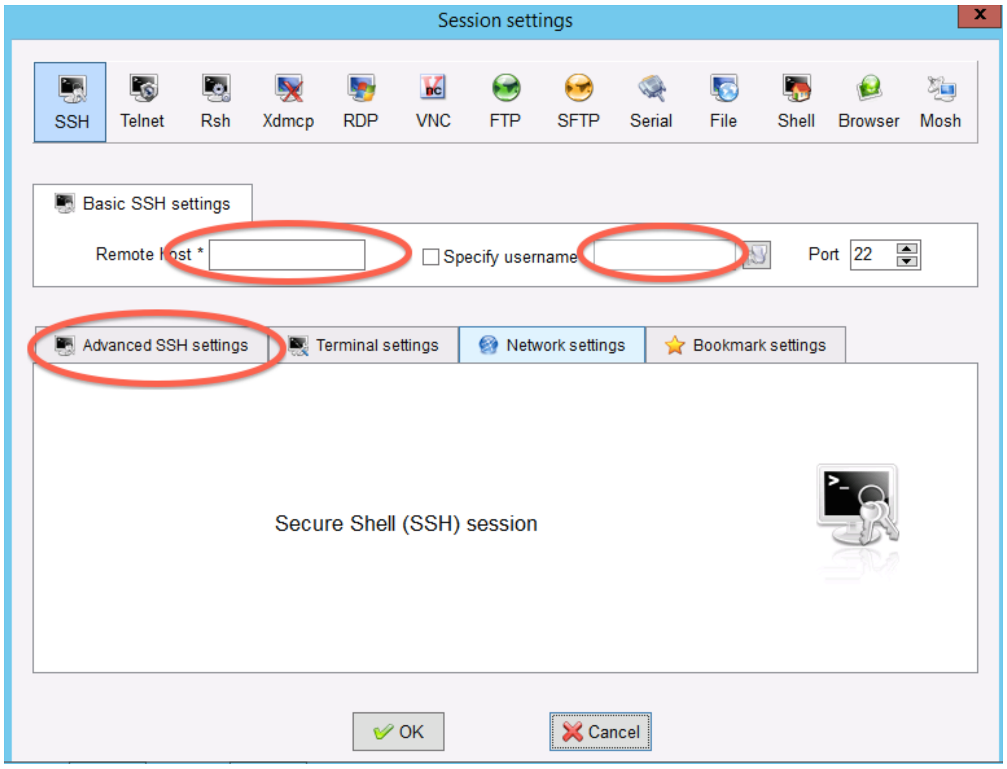
Press <return> to start a new session or type a remote server name

CygUtils plugin not found on your system.

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: http://mobaxterm.mobatek.net

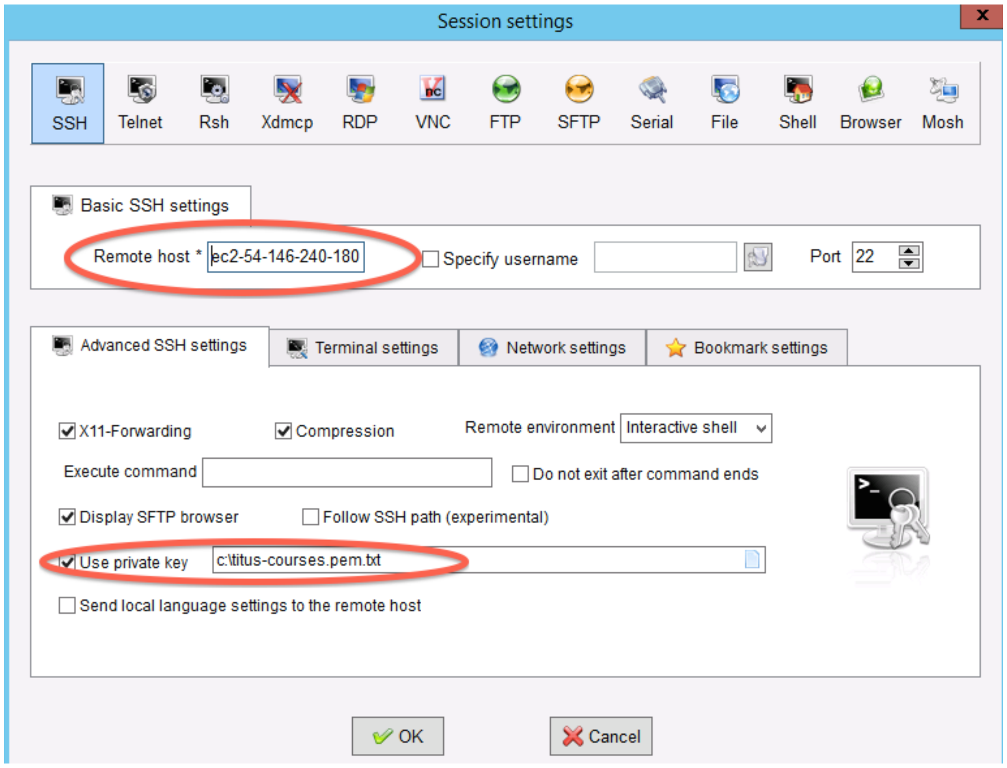
2.2.3 Fill in session settings

Put in your hostname (should be `ec2-XXX-YYY-ZZZ-AAA.compute-1.amazon.aws.com`), select 'specify username', and enter 'ubuntu'.



2.2.4 Specify the session key

Copy the downloaded .pem file onto your primary hard disk (generally C:) and the put in the full path to it.

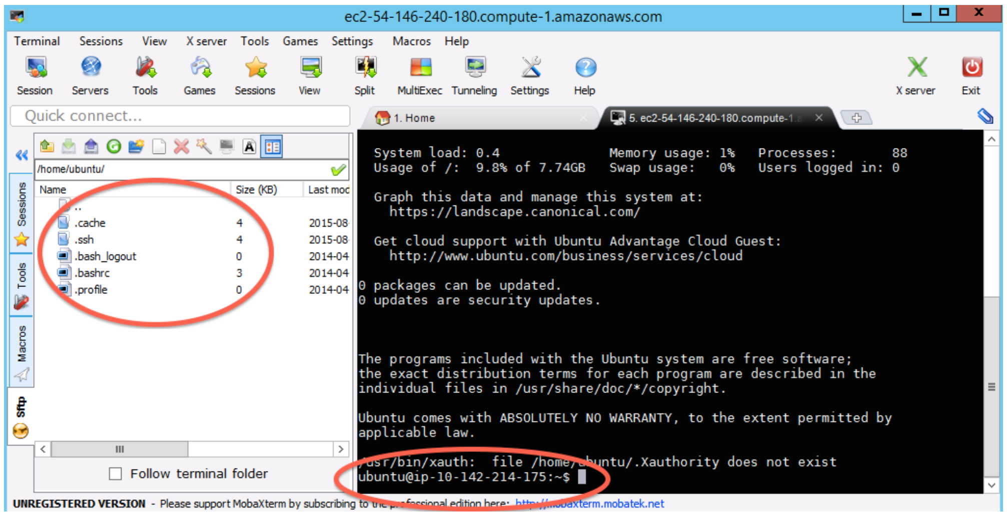


Remote host * ec2-54-146-240-180
☐ Specify username
Port 22

☑ X11-Forwarding
☑ Compression
Remote environment: Interactive shell
Execute command
☐ Do not exit after command ends
☑ Display SFTP browser
☐ Follow SSH path (experimental)
☑ Use private key c:\titus-courses.pem.txt
☐ Send local language settings to the remote host

2.2.5 Click OK

Victory! (?)



/home/ubuntu/

Name	Size (KB)	Last mod
..		
.cache	4	2015-08
.ssh	4	2015-08
.bash_logout	0	2014-04
.bashrc	3	2014-04
.profile	0	2014-04

```
  System load: 0.4              Memory usage: 1%   Processes:       88
  Usage of /:  9.8% of 7.74GB   Swap usage:   0%   Users logged in: 0

  Graph this data and manage this system at:
    https://landscape.canonical.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

/usr/bin/xauth:  file /home/ubuntu/.Xauthority does not exist
ubuntu@ip-10-142-214-175:~$
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: http://mobaxterm.mobatek.net

Return to index: Getting started with Amazon EC2

2.3 Logging into your new instance "in the cloud" (Mac version)

OK, so you've created a running computer. How do you get to it?

The main thing you'll need is the network name of your new computer. To retrieve this, go to the instance view and click on the instance, and find the "Public DNS". This is the public name of your computer on the Internet.

Copy this name, and connect to that computer with ssh under the username 'ubuntu', as follows.

First, find your private key file; it's the .pem file you downloaded when starting up your EC2 instance. It should be in your Downloads folder. Move it onto your desktop and rename it to 'amazon.pem'.

Next, start Terminal (in Applications... Utilities...) and type:

```
chmod og-rwx ~/Downloads/amazon.pem
```

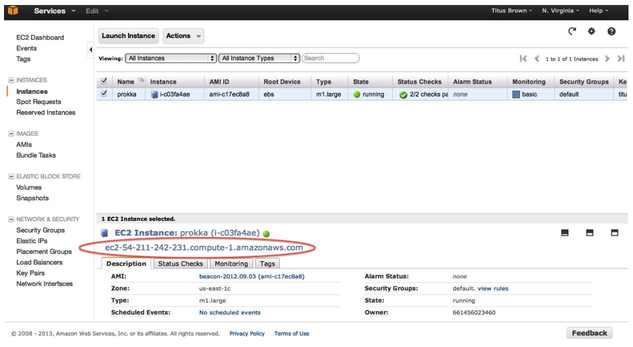
to set the permissions on the private key file to "closed to all evildoers".

Then type:

```
ssh -i ~/Downloads/amazon.pem ubuntu@ec2-???-???-???-???.compute-1.amazonaws.com
```

Here, you're logging in as user 'ubuntu' to the machine 'ec2-174-129-122-189.compute-1.amazonaws.com' using the authentication key located in 'amazon.pem' on your Desktop.

Note, you have to replace the stuff after the '@' sign with the name of the host; see the red circle in:



Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Groups	Ke
prokka	i-c03fa4ae	ami-c17ec8a8	ebs	m1.large	running	2/2 checks pa	none	basic	default	titu

1 EC2 Instance selected.

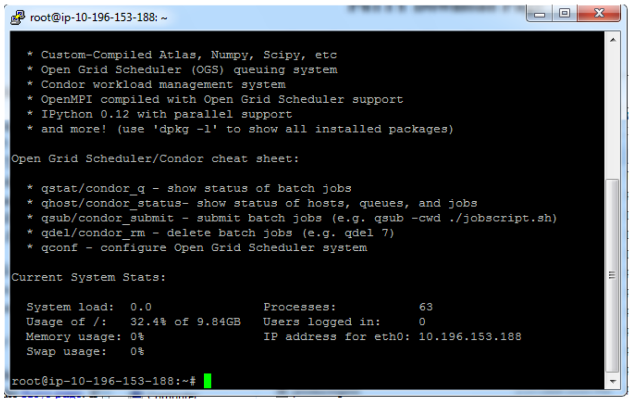
EC2 Instance: prokka (i-c03fa4ae)

ec2-54-211-242-231.compute-1.amazonaws.com

AMI:	beacon-2012.09.03 (ami-c17ec8a8)	Alarm Status:	none
Zone:	us-east-1c	Security Groups:	default. view rules
Type:	m1.large	State:	running
Scheduled Events:	No scheduled events	Owner:	661456023460

© 2008 - 2013, Amazon Web Services, Inc. or its affiliates. All rights reserved.

At the end you should see text and a prompt that look like this:



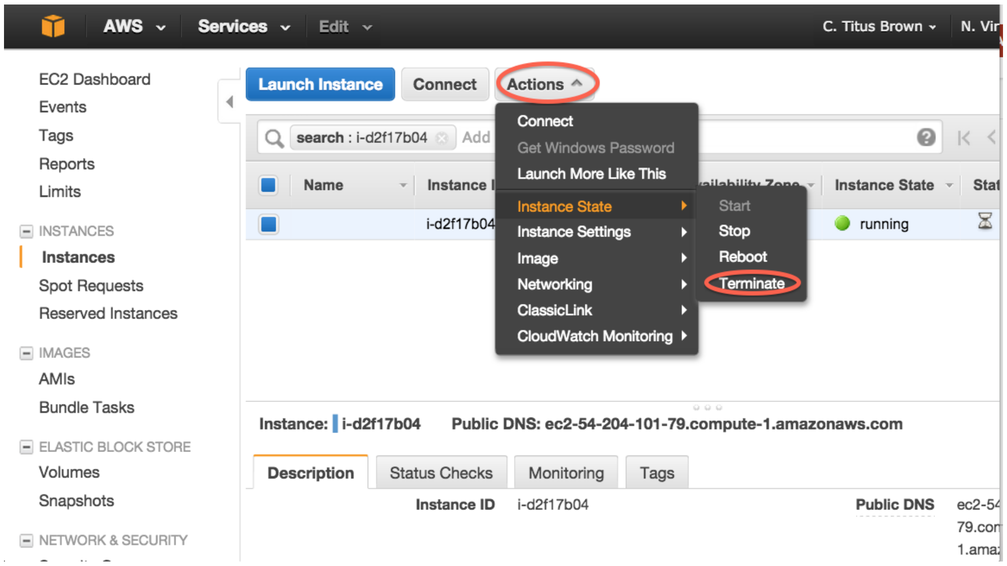
```
  * Custom-Compiled Atlas, Numpy, Scipy, etc
  * Open Grid Scheduler (OGS) queuing system
  * Condor workload management system
  * OpenMPI compiled with Open Grid Scheduler support
  * IPython 0.12 with parallel support
  * and more! (use 'dpkg -l' to show all installed packages)

Open Grid Scheduler/Condor cheat sheet:

  * qstat/condor_q - show status of batch jobs
  * qhost/condor_status- show status of hosts, queues, and jobs
  * qsub/condor_submit - submit batch jobs (e.g. qsub -cwd ./jobscript.sh)
  * qdel/condor_rm - delete batch jobs (e.g. qdel 7)
  * qconf - configure Open Grid Scheduler system

Current System Stats:

  System load:  0.0               Processes:           63
  Usage of /:   32.4% of 9.84GB   Users logged in:     0
  Memory usage: 0%                IP address for eth0: 10.196.153.188
  Swap usage:   0%

root@ip-10-196-153-188:~#
```

Return to index: Getting started with Amazon EC2

2.4 Terminating your instance

Be sure to terminate your instance(s) after transferring off any data that you want to keep!

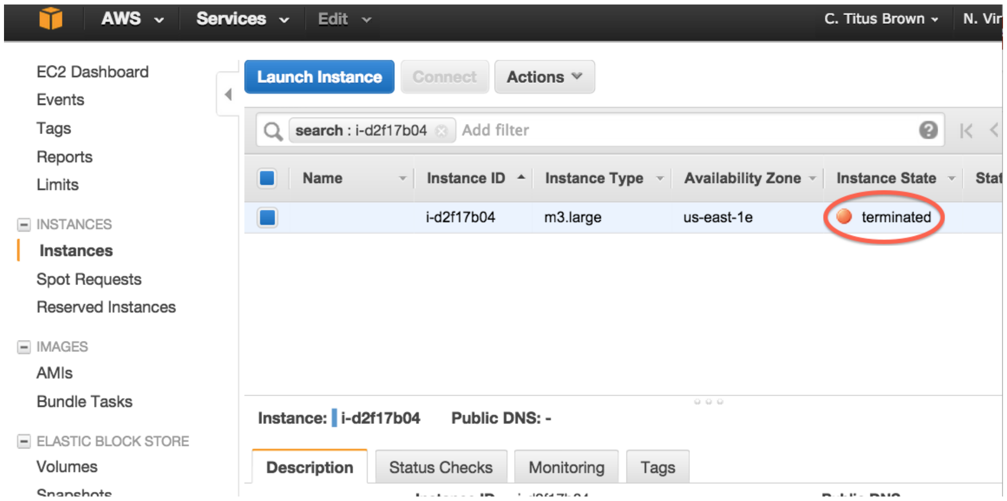
To terminate your instance, select the instance you want to terminate, and then go to the 'Actions' menu and select 'Instance actions', 'terminate':



Instance: i-d2f17b04 Public DNS: ec2-54-204-101-79.compute-1.amazonaws.com

Instance ID i-d2f17b04

Wait a minute or two to be sure that the instance state changes to "terminated":



Name	Instance ID	Instance Type	Availability Zone	Instance State
	i-d2f17b04	m3.large	us-east-1e	terminated

Instance: i-d2f17b04 Public DNS: -

A final checklist:

- You have a green EC2 instance!
- You used ubuntu 14.04;
- You didn't start a micro instance (m3.xlarge, or bigger);

2.5 Amazon Web Services reference material

Instance types

Instance costs

Next: Short read quality and trimming

CHAPTER 3

Genomics

3.1 Short read quality and trimming

Note: Reminder: if you're on Windows, you should install mobaxterm.

OK, you should now be logged into your Amazon computer! How exciting!

3.1.1 Prepping the computer

Before we do anything else, we need to set up a place to work and install a few things.

First, let's set up a place to work:

```
sudo chmod a+rwxt /mnt
```

This makes '/mnt' a place where we can put data and working files.

Next, let's install a few things:

```
sudo apt-get update
sudo apt-get install -y trimmomatic fastqc python-pip python-dev
```

These are the Trimmomatic and FastQC programs, which we'll use below, along with some software prerequisites that we'll need for other things below.

3.1.2 Data source

We're going to be using a subset of data from the E. coli reference published in Chitsaz et al., 2011. The data was originally downloaded from http://bix.ucsd.edu/projects/singlecell/nbt_data.html, E. coli reference lane 6.

3.1.3 1. Copying in some data to work with.

We've loaded subsets of the data onto an Amazon location for you, to make everything faster for today's work. We're going to put the files on your computer locally under the directory /mnt/data:

```
mkdir /mnt/data
```

Next, let's grab the data set:

```
cd /mnt/data
curl -O -L https://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/ECOLI_R1
curl -O -L https://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/ECOLI_R2
```

Now if you type:

```
ls -l
```

you should see something like:

```
-rw-rw-r-- 1 ubuntu ubuntu 418068323 Sep 24 15:04 ECOLI_R1.fastq.gz
-rw-rw-r-- 1 ubuntu ubuntu 429978135 Sep 24 15:05 ECOLI_R2.fastq.gz
```

These are each 5m read subsets of the original data. This is analogous to what you would see if you did a HiSeq or MiSeq run on a bacterial sample.

One problem with these files is that they are writeable - by default, UNIX makes things writeable by the file owner. Let's fix that before we go on any further:

```
chmod u-w *
```

We'll talk about what these files are below.

3.1.4 2. Copying data into a working location

First, make a working directory; this will be a place where you can futz around with a copy of the data without messing up your primary data:

```
mkdir /mnt/work
cd /mnt/work
```

Now, make a "virtual copy" of the data in your working directory by linking it in –

```
ln -fs /mnt/data/* .
```

These are FASTQ files – let's take a look at them:

```
less ECOLI_R1.fastq.gz
```

(use the spacebar to scroll down, and type 'q' to exit 'less')

Question:

- why are there R1 and R2 in the file names?

Links:

- FASTQ Format

3.1.5 3. FastQC

We're going to use FastQC to summarize the data. We already installed 'fastqc' on our computer - that's what the 'apt-get install' did, above.

Now, run FastQC on two files:

```
fastqc ECOLI_R1.fastq.gz
fastqc ECOLI_R2.fastq.gz
```

Now type 'ls':

```
ls -d *fastqc*
```

to list the files, and you should see:

```
ECOLI_R1
```

We are *not* going to show you how to look at these files right now - you need to copy them to your local computer to do that. We'll show you that tomorrow. But! we can show you what they look like, because I've made copiesd of them for you:

- ECOLI_R1_fastqc/fastqc_report.html
- ECOLI_R2_fastqc/fastqc_report.html

Questions:

- What should you pay attention to in the FastQC report?
- Which is "better", R1 or R2? And why?

Links:

- FastQC
- FastQC tutorial video

See slide 39 and onwards for what BAD FastQC reports look like!

3.1.6 4. Trimmomatic

Now we're going to do some trimming! We'll be using Trimmomatic, which (as with fastqc) we've already installed via apt-get.

The first thing we'll need are the adapters to trim off:

```
curl -O -L http://dib-training.ucdavis.edu.s3.amazonaws.com/mRNAseq-semi-2015-03-04/TruSeq2-PE.fa
```

Now, to run Trimmomatic:

```
TrimmomaticPE ECOLI_R1.fastq.gz ECOLI_R2.fastq.gz \
     ECOLI_R1.qc.fq.gz s1_se ECOLI_R2.qc.fq.gz s2_se \
     ILLUMINACLIP:TruSeq2-PE.fa:2:40:15 \
     LEADING:2 TRAILING:2 \
     SLIDINGWINDOW:4:2 \
     MINLEN:25
```

You should see output that looks like this:

```
...
Input Read Pairs: 5000000 Both Surviving: 4991513 (99.83%) Forward Only Surviving: 7422 (0.15%) Rever
TrimmomaticPE: Completed successfully
```

Capture the newly orphaned sequences like so:

```
cat s1_se s2_se | gzip > ECOLI_orphans.qc.fq.gz
```

Questions:

- How do you figure out what the parameters mean?
- How do you figure out what parameters to use?

- What adapters do you use?
- What version of Trimmomatic are we using here? (And FastQC?)
- Do you think parameters are different for RNAseq and genomic data sets?
- What's with these annoyingly long and complicated filenames?
- why are we running R1 and R2 together?

Links:

- Trimmomatic

3.1.7 5. FastQC again

Run FastQC again on the trimmed files:

```
fastqc ECOLI_R1.qc.fq.gz
fastqc ECOLI_R2.qc.fq.gz
fastqc ECOLI_orphans.qc.fq.gz
```

And now view my copies of these files:

- ECOLI_R1.qc.fq_fastqc/fastqc_report.html
- ECOLI_R2.qc.fq_fastqc/fastqc_report.html
- ECOLI_orphans.qc.fq_fastqc/fastqc_report.html

Let's take a look at the output files:

```
less ECOLI_R1.qc.fq.gz
```

(again, use spacebar to scroll, 'q' to exit less).

Questions:

- is the quality trimmed data "better" than before?
- Does it matter that you still have adapters!?

3.1.8 6. Interleave the sequences

Next, we need to take these R1 and R2 sequences and convert them into interleaved form, for the next step. To do this, we'll use scripts from the khmer package, which we need to install:

```
sudo pip install -U setuptools
sudo pip install khmer==2.0
```

Now, interleave the reads:

```
interleave-reads.py ECOLI_R1.qc.fq.gz ECOLI_R2.qc.fq.gz --gzip \
   -o ecoli_ref-5m-trim.pe.fq.gz
```

and rename the orphans:

```
cp ECOLI_orphans.qc.fq.gz ecoli_ref-5m-trim.se.fq.gz
```

Done!

Next: Assembling E. coli sequences with SPAdes

3.2 Assembling E. coli sequences with SPAdes

The goal of this tutorial is to show you the basics of assembly using the SPAdes assembler.

We'll be using data from Efficient de novo assembly of single-cell bacterial genomes from short-read data sets, Chitsaz et al., 2011.

3.2.1 Packages to install

Download and insatll the SPAdes assembler:

```
cd ~/
wget http://spades.bioinf.spbau.ru/release3.6.0/SPAdes-3.6.0-Linux.tar.gz
tar xvfz SPAdes-3.6.0-Linux.tar.gz
export PATH=$PATH:$HOME/SPAdes-3.6.0-Linux/bin
echo 'export PATH=$PATH:$HOME/SPAdes-3.6.0-Linux/bin' >> ~/.bashrc
```

as well as Quast, software for evaluating the assembly against the known reference:

```
cd
curl -L http://sourceforge.net/projects/quast/files/quast-3.0.tar.gz/download > quast-3.0.tar.gz
tar xvf quast-3.0.tar.gz
```

3.2.2 Getting the data

Now, let's create a working directory:

```
cd /mnt
mkdir assembly
cd assembly
```

Copy in the E. coli data that you trimmed (Short read quality and trimming):

```
ln -fs /mnt/work/ecoli_ref-5m-trim.*.fq.gz .
```

3.2.3 Running an assembly

Now, let's run an assembly:

```
spades.py --12 ecoli_ref-5m-trim.pe.fq.gz -s ecoli_ref-5m-trim.se.fq.gz -o spades.d
```

This will take about 15 minutes; it should end with the following output:

```
* Corrected reads are in /mnt/assembly/spades.d/corrected/
* Assembled contigs are in /mnt/assembly/spades.d/contigs.fasta (contigs.fastg)
* Assembled scaffolds are in /mnt/assembly/spades.d/scaffolds.fasta (scaffolds.fastg)
```

3.2.4 Looking at the assembly

Run QUAST:

```
~/quast-3.0/quast.py spades.d/scaffolds.fasta -o report
```

and then look at the report:

```
less report/report.txt
```

You should see:

```
All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0

Assembly                    scaffolds
# contigs (>= 0 bp)         152
# contigs (>= 1000 bp)      80
Total length (>= 0 bp)      4571384
Total length (>= 1000 bp)   4551778
# contigs                   89
Largest contig              285527
Total length                4558170
GC (%)                      50.74
N50                         133088
N75                         67332
L50                         12
L75                         23
# N's per 100 kbp           0.00
```

What does this all mean?

3.2.5 Comparing and evaluating assemblies - QUAST

Download the true reference genome:

```
cd /mnt/assembly
curl -O https://s3.amazonaws.com/public.ged.msu.edu/ecoliMG1655.fa.gz
gunzip ecoliMG1655.fa.gz
```

and run QUAST again:

```
~/quast-3.0/quast.py -R ecoliMG1655.fa spades.d/scaffolds.fasta -o report
```

Note that here we're looking at *all* the assemblies we've generated.

Now look at the results:

```
less report/report.txt
```

and now we have a lot more information! What all is there?

3.2.6 Adding in Nanopore data and doing a hybrid assembly

One challenge with short read data like Illumina is that if there are repeats in the genome, they can't be unambiguously resolved with short reads. Enter long reads, produced by PacBio and Nanopore sequencing. How much do long reads improve the assembly?

Let's download some trial Nanopore data provided by Nick Loman:

```
cd /mnt/assembly
curl -O https://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/FC20.wf1.9.
```

Let's take a quick look at these sequences and try BLASTing them at NCBI – note, you'll want to use blastn, and choose "somewhat similar sequences" at the bottom. You can also restrict the BLAST search to E. coli MG1655.

Grab part of a sequence with `gunzip -c FC20* | head` and paste it into BLAST. What do you see?

Now, let's try adding them into the assembly by running SPAdes with the Nanopore command line flag:

```
spades.py --sc --12 ecoli_ref-5m-trim.pe.fq.gz -s ecoli_ref-5m-trim.se.fq.gz --nanopore FC20.wf1.9.2I
```

How'd we do?

```
~/quast-3.0/quast.py -R ecoliMG1655.fa nanopore-ecoli-sc/scaffolds.fasta -o n_report
```

3.2.7 Reference-free comparison

Above, we've been using the genome reference to do assembly comparisons – but often you don't have one. What do you do to evaluate and compare assemblies without a reference?

One interesting trick is to just run QUAST with one assembly as a reference, and the other N assemblies against it. My only suggestion is to first eliminate short, fragmented contigs from the assembly you're going to use as a reference.

Let's try that, using `extract-long-sequences.py` from khmer:

```
extract-long-sequences.py -l 1000 nanopore-ecoli-sc/scaffolds.fasta > spades-long.fa
```

and then re-run QUAST and put the output in `report-noref/report.txt`:

```
~/quast-3.0/quast.py -R spades-long.fa spades.d/scaffolds.fasta \
         nanopore-ecoli-sc/scaffolds.fasta -o report-noref
```

When you look at the report,

```
less report-noref/report.txt
```

take particular note of the following –

```
Assembly                     spades.d_scaffolds  nanopore-ecoli-sc_scaffolds
# contigs (>= 0 bp)          152                 15
# contigs (>= 1000 bp)       80                  7
Total length (>= 0 bp)       4571384             4643870
Total length (>= 1000 bp)    4551778             4642289
# contigs                    89                  7
Largest contig               285527              3076878
Total length                 4558170             4642289
Reference length             4642289             4642289
   ...
Misassembled contigs length  134677              0
# local misassemblies        6                   0
...
Genome fraction (%)          98.161              99.923
Duplication ratio            1.000               1.001
# mismatches per 100 kbp     3.36                0.00
```

3.3 Bacterial genome annotation using Prokka

After you have de novo assembled your genome sequencing reads into contigs, it is useful to know what genomic features are on those contigs. The process of identifying and labelling those features is called genome annotation.

In this tutorial you will:

1. Download and install Prokka
2. Annotate a FASTA file of contigs

3. Visualize the annotation using Artemis

The instructions below will work on a Linux server (eg. EC2 instance), a Linux desktop, and even directly on your Mac OS X Laptop.

3.3.1 Download and install Prokka

Prokka is simple to install because it comes bundled with all its dependencies:

```
cd
sudo apt-get -y install git bioperl libxml-simple-perl ncbi-blast+
git clone https://github.com/tseemann/prokka.git
export PATH=$PWD/prokka/bin:$PATH
prokka --setupdb
prokka --version
```

3.3.2 Apply Prokka to your just-assembled E. coli genome

Prokka is a pipeline script which coordinates a series of genome feature predictor tools and sequence similarity tools to annotate the genome sequence (contigs).

```
cd /mnt/assembly
prokka --outdir anno --prefix prokka --centre X --compliant spades-long.fa
```

```
cat ./anno/prokka.txt
```

How many genes did Prokka find in the contigs?

3.3.3 Install Artemis

Artemis is a graphical Java program to browse annotated genomes. It is a a bit like IGV but specifically designed for bacteria. You will need to install this on your desktop computer. You could run it remotely over SSH using X11 forwarding from Amazon but it is probably too slow to be useful.

Download: https://www.sanger.ac.uk/resources/software/artemis/#download

3.3.4 Viewing the annotated genome

- Start Artemis
- Go to “File” -> “SSH File Manager”
- Type in the IP number of your Amazon EC2 instance
- Browse to the “anno/prokka.gff” file

CHAPTER 4

Transcriptomics

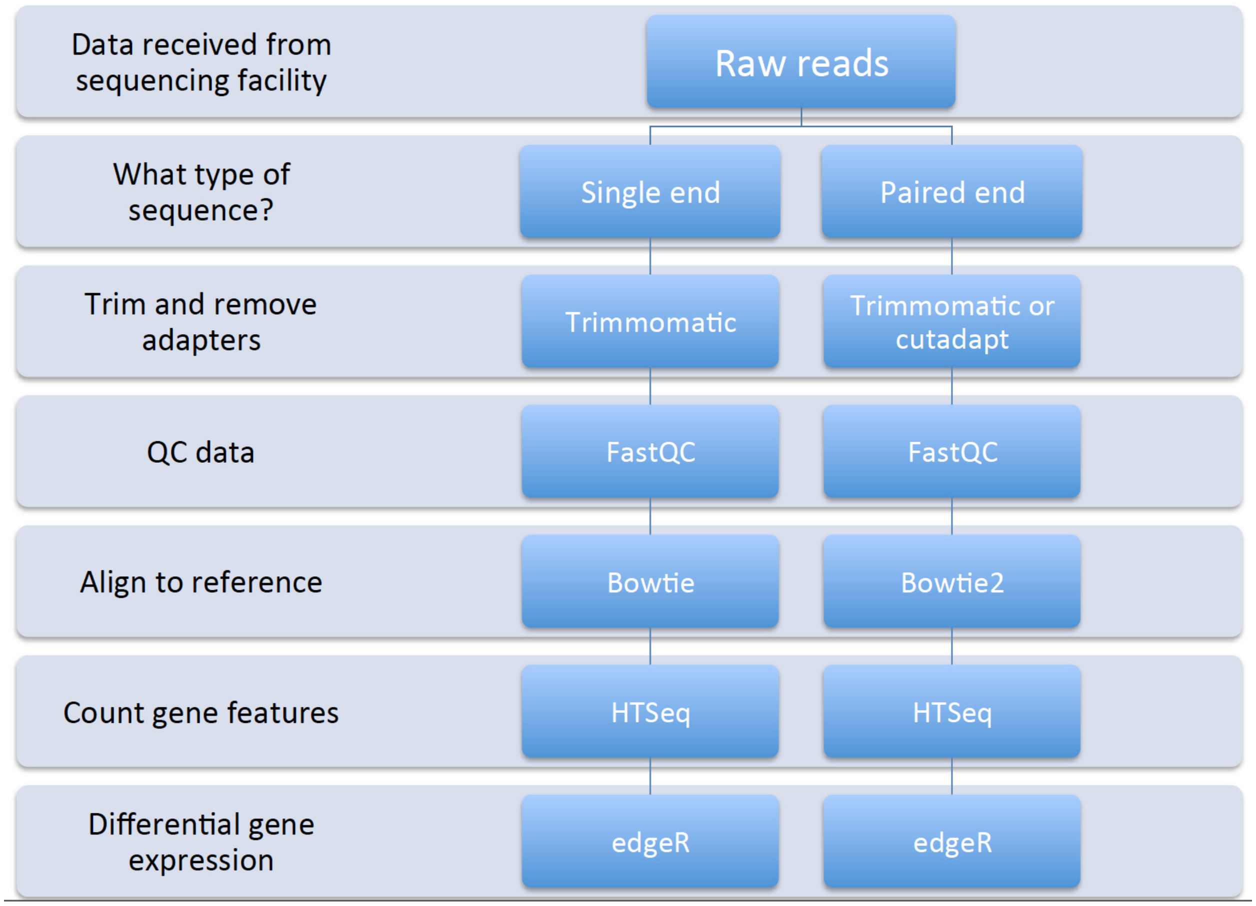
4.1 Considerations for reference vs reference-free transcriptome analysis

Whether you work with microbes that have a sequence reference genome or you have one without a reference, please know that there are *several* reasonably complete workflows that you can use for both types of analyses. No need to re-invent the wheel.

We will attempt to cover both approaches using available workflows and then point you to additional resources so that you can choose the one that best fits your needs.

4.1.1 Reference-based transcriptome analysis

The general approach for reference-based (if you already have an available sequenced genome) is as follows (can use different tools though):

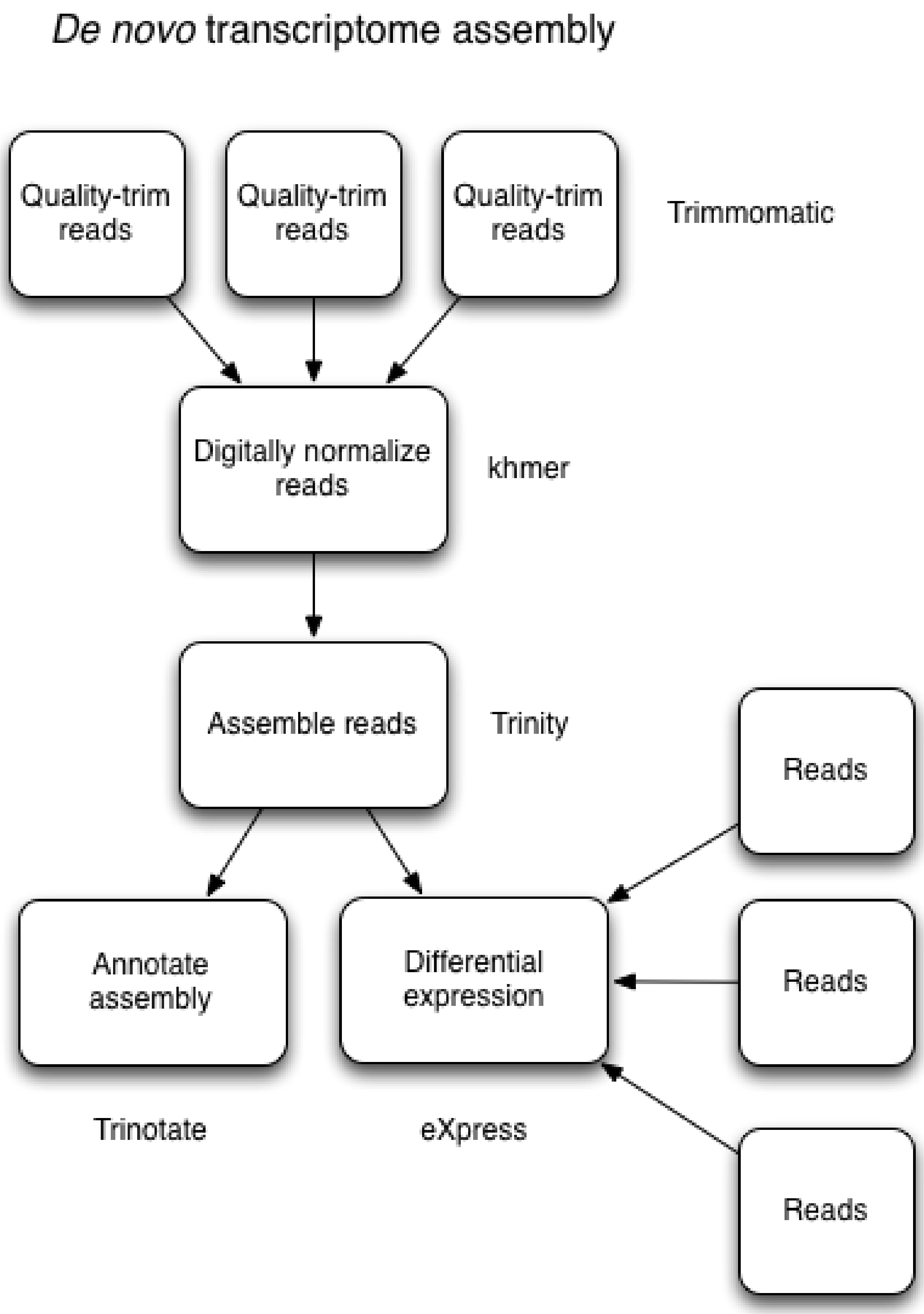


Several workflows can be utilized for this type of analysis, including:

- SPARTA - sparta.readthedocs.org
- Rockhopper - http://cs.wellesley.edu/~btjaden/Rockhopper/
- Galaxy (drag and drop cloud computing GUI interface for NGS data analysis) - https://galaxyproject.org/
- Several others not necessarily mentioned here but a Google search can help

4.1.2 Reference-free transcriptome analysis

The approach for reference-free (no good reference genome and not enough pre-existing mRNA-seq data for assembled transcriptomes) is an entirely different animal than reference-based approaches. The process is related and generally can follow this type of workflow (this is an example from eukaryotes):



This is usually an iterative process that may require additional tools and computational time to work through the data. But it can be done! A useful presentation from Meg Staton on de novo assembly.

There are fewer workflows readily available for *de novo* transcriptome assembly:

- Rockhopper2 - http://cs.wellesley.edu/~btjaden/Rockhopper/
- khmer and Trinity - http://khmer.readthedocs.org/en/v2.0/ and http://trinityrnaseq.github.io/
- Galaxy (drag and drop cloud computing GUI interface for NGS data analysis) - https://galaxyproject.org/

- What we will show today as a brief example

Next: RNA-seq and Differential Gene Expression in Bacteria

4.2 RNA-seq and Differential Gene Expression in Bacteria

Today we have a few objectives we would like to cover:

1. Set up a project for your data analysis for reproducible and robust analysis - Setting up your RNA-seq project
2. Quality control and trimming of short RNA-seq reads - Quality control and trimming of reads
3. Workflows for reference-based and reference-free transcriptome analysis - Align and count gene features and Considerations for reference vs reference-free transcriptome analysis
4. Differential gene expression and some potential pitfalls - Differential gene expression

4.2.1 Learning goals

- Familiarize yourself with creating documentation and data organization strategies
- Perform and assess read trimming and quality
- Analyze reference and reference-free transcriptome analysis through available workflows
- Understand underlying assumptions and potential issues with differential gene expression

At this point, go ahead and start up new m3.xlarge EC2 instances, as you did yesterday (see: Getting started with Amazon EC2).

4.2.2 Additional Resource - Basic Linux/Unix commands

To refresh your memory on some basic Linux/Unix commands, we will cover the basic commands necessary to:

1. Move through folders

2. List the contents of a folder

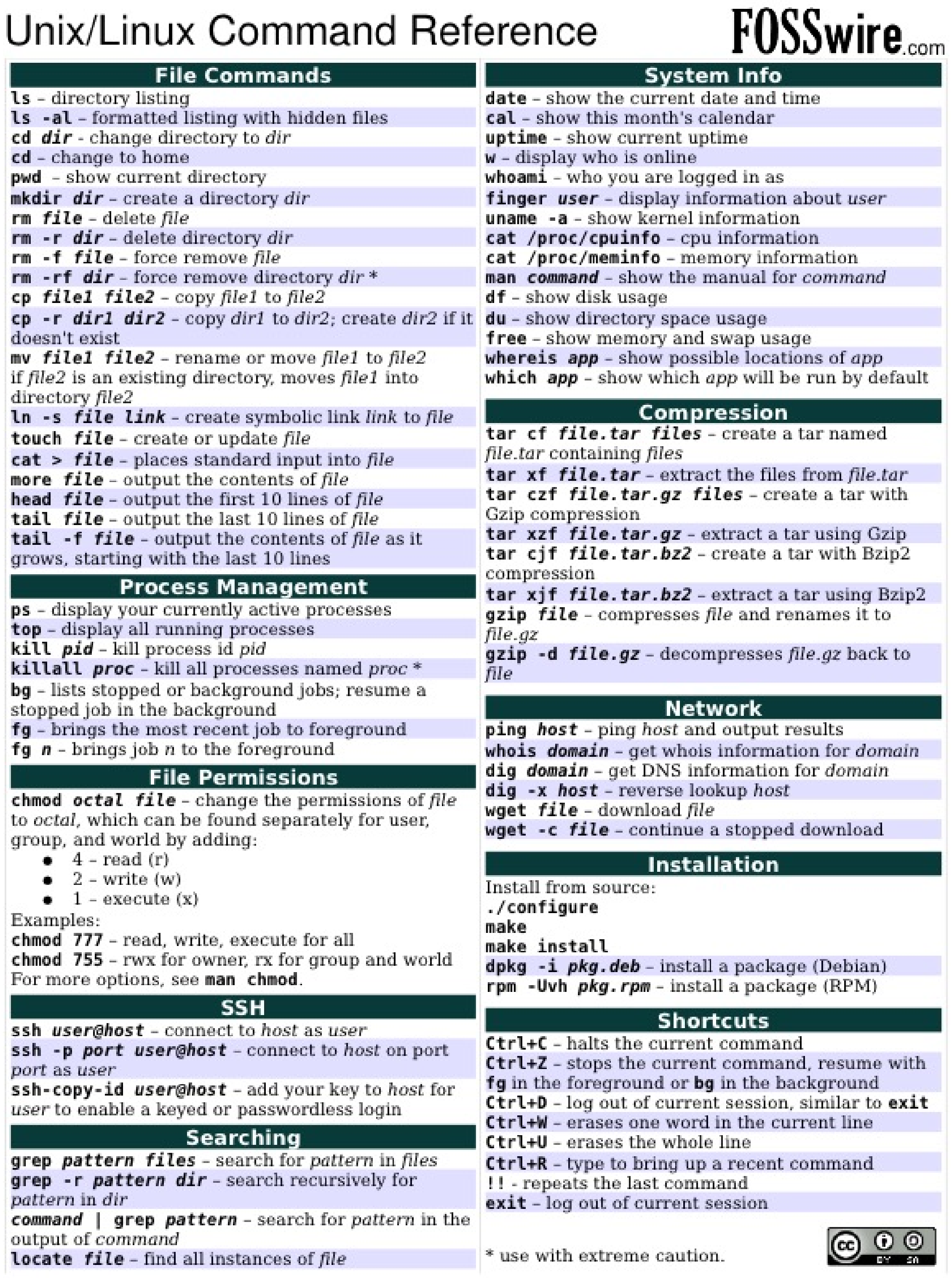
3. Make new folders

4. Rename files/folders

5. Delete files/folders

	Command	What it does...	Examples
1.	cd	Change directory/folder	**>** cd ~ (this changes to your home directory); **>** cd .. (this goes back one folder)
2.	ls	List the contents of a folder	**>** ls
3.	mkdir	Make a new directory/folder	**>** mkdir NewFolder (this will make a new folder called 'NewFolder' in your current directory)
4.	mv	Rename or move a file from one name to another	**>** mv file1 file2 (this will rename/move file1 to file2)
5.	rm	Remove a file (add the -r flag to remove a folder)	**>** rm file1 (remove file1); **>** rm -r folder1 (remove folder1)

Command reference sheet



Unix/Linux Command Reference FOSSwire.com

File Commands

`ls` – directory listing
`ls -al` – formatted listing with hidden files
`cd dir` - change directory to *dir*
`cd` – change to home
`pwd` – show current directory
`mkdir dir` – create a directory *dir*
`rm file` – delete *file*
`rm -r dir` – delete directory *dir*
`rm -f file` – force remove *file*
`rm -rf dir` – force remove directory *dir* *
`cp file1 file2` – copy *file1* to *file2*
`cp -r dir1 dir2` – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
`mv file1 file2` – rename or move *file1* to *file2* if *file2* is an existing directory, moves *file1* into directory *file2*
`ln -s file link` – create symbolic link *link* to *file*
`touch file` – create or update *file*
`cat > file` – places standard input into *file*
`more file` – output the contents of *file*
`head file` – output the first 10 lines of *file*
`tail file` – output the last 10 lines of *file*
`tail -f file` – output the contents of *file* as it grows, starting with the last 10 lines

Process Management

`ps` – display your currently active processes
`top` – display all running processes
`kill pid` – kill process id *pid*
`killall proc` – kill all processes named *proc* *
`bg` – lists stopped or background jobs; resume a stopped job in the background
`fg` – brings the most recent job to foreground
`fg n` – brings job *n* to the foreground

File Permissions

`chmod octal file` – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
`chmod 777` – read, write, execute for all
`chmod 755` – rwx for owner, rx for group and world
For more options, see `man chmod`.

SSH

`ssh user@host` – connect to *host* as *user*
`ssh -p port user@host` – connect to *host* on port *port* as *user*
`ssh-copy-id user@host` – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

`grep pattern files` – search for *pattern* in *files*
`grep -r pattern dir` – search recursively for *pattern* in *dir*
`command | grep pattern` – search for *pattern* in the output of *command*
`locate file` – find all instances of *file*

System Info

`date` – show the current date and time
`cal` – show this month's calendar
`uptime` – show current uptime
`w` – display who is online
`whoami` – who you are logged in as
`finger user` – display information about *user*
`uname -a` – show kernel information
`cat /proc/cpuinfo` – cpu information
`cat /proc/meminfo` – memory information
`man command` – show the manual for *command*
`df` – show disk usage
`du` – show directory space usage
`free` – show memory and swap usage
`whereis app` – show possible locations of *app*
`which app` – show which *app* will be run by default

Compression

`tar cf file.tar files` – create a tar named *file.tar* containing *files*
`tar xf file.tar` – extract the files from *file.tar*
`tar czf file.tar.gz files` – create a tar with Gzip compression
`tar xzf file.tar.gz` – extract a tar using Gzip
`tar cjf file.tar.bz2` – create a tar with Bzip2 compression
`tar xjf file.tar.bz2` – extract a tar using Bzip2
`gzip file` – compresses *file* and renames it to *file.gz*
`gzip -d file.gz` – decompresses *file.gz* back to *file*

Network

`ping host` – ping *host* and output results
`whois domain` – get whois information for *domain*
`dig domain` – get DNS information for *domain*
`dig -x host` – reverse lookup *host*
`wget file` – download *file*
`wget -c file` – continue a stopped download

Installation

Install from source:

```
./configure
make
make install
```

`dpkg -i pkg.deb` – install a package (Debian)
`rpm -Uvh pkg.rpm` – install a package (RPM)

Shortcuts

`Ctrl+C` – halts the current command
`Ctrl+Z` – stops the current command, resume with `fg` in the foreground or `bg` in the background
`Ctrl+D` – log out of current session, similar to `exit`
`Ctrl+W` – erases one word in the current line
`Ctrl+U` – erases the whole line
`Ctrl+R` – type to bring up a recent command
`!!` - repeats the last command
`exit` – log out of current session

* use with extreme caution.

(Ref. sheet from: http://files.fosswire.com/2007/08/fwunixref.pdf)

Next: Setting up your RNA-seq project

4.3 Setting up your RNA-seq project

So you've decided to do an RNA-seq experiment. Some things you will need to think about are:

- What is the end-goal? (differential gene expression, variant calling, *de novo* assembly)
- How many replicates do I need per condition? (related to first question; 3 is good, 6 is better)
- How am I going to analyze this data?

You will save yourself a *lot* of headache if you take the time to set up your project in a coherent manner.

Some tips:

1. Put your project in a single directory/folder (named something informative without spaces)
2. *Start a virtual notebook* for your analysis, commands, parameters, etc. - plain text or markdown
    - I like markdown because you can readily export to HTML to share with collaborators
3. Keep your raw data *raw* (e.g. make a working copy)
4. Name files something concise (NO SPACES!) but informative (tab auto-complete is your friend)
5. Make several *separate* folders within the main project folder to house different steps of the analysis
6. In each subfolder create a README to describe the contents of that folder, how they got there, etc. (metadata)
7. If possible, automate (script) analysis to allow you to re-run analyses on (many) files quickly and reproducibly

To put it another way, treat it as you would setting up a bench experiment and documenting progress in a lab notebook. You will write down dates, times, reagents (lot numbers, purity, etc.), buffer/media recipes, purpose of the experiment, experimental procedure, results (and if you believe them), and what you plan to do next to continue to investigate the question or if there are new questions to be answered.

You want to approach bioinformatics analyses in a similar manner. Document everything as best as you can. It will be useful for when you share data with others (publish, collaborate, etc.) and when you have to go back in 6 months to reassess what it was you did to generate graphs x, y, and z.

Also, be aware that tools and technology evolve quickly in this field. It's worth getting a Twitter account to interact with people who develop/perform these analyses and probing the literature for new/improved tools (they come out *frequently*). Further, Google is your friend and there are all kinds of resources (forums, etc.) to ask questions and get answers (usually pretty quickly). In particular, you should check out seqanswers.com and biostars.

Note, on Twitter, for microbiology, we suggest following Nick Loman, Torsten Seemann, and Jonathan Eisen.

4.3.1 Start a virtual notebook

Let's have a look at the Markdown syntax and how you can use it to document your work.

Markdown is a plain text format that can be rendered to HTML and is quite nice if working collaboratively, like on GitHub.

I will use an example from Vince Buffalo's book "Bioinformatics Data Skills" (*highly recommended*):

```
# *Zea Mays* SNP Calling

We sequenced three lines of *zea mays*, using paired-end
sequencing. This sequencing was done by our sequencing core and we
```

```
received the data on 2013-05-10. Each variety should have **two**
sequences files, with suffixes `_R1.fastq` and `_R2.fastq`, indicating
which member of the pair it is.

## Sequencing Files

All raw FASTQ sequences are in `data/seqs/`:

$ find data/seqs -name "*.fastq"
data/seqs/zmaysA_R1.fastq
data/seqs/zmaysA_R2.fastq
data/seqs/zmaysB_R1.fastq
data/seqs/zmaysB_R2.fastq
data/seqs/zmaysC_R1.fastq
data/seqs/zmaysC_R2.fastq

## Quality Control Steps

After the sequencing data was received, our first stage of analysis
was to ensure the sequences were high quality. We ran each of the
three lines' two paired-end FASTQ files through a quality diagnostic
and control pipeline. Our planned pipeline is:

1. Create base quality diagnostic graphs.
2. Check reads for adapter sequences.
3. Trim adapter sequences.
4. Trim poor quality bases.

Recommended trimming programs:

- Trimmomatic
- Scythe
```

When this is rendered, it looks like this.

Here are some Markdown editors for

- Mac - http://25.io/mou/
- Windows - http://markdownpad.com/

If you use GitHub to work collaboratively on things, you can copy and paste things right into a Gist and it will render for you (if you add the .md file extension when you name your file) and you can share privately or publicly!

Markdown syntax guide

Next: Quality control and trimming of reads

4.4 Quality control and trimming of reads

For our purposes, let's create a project directory for all of our files for the transcriptomics component of the workshop

Update the machine and install some software:

```
sudo chmod a+rwxt /mnt
sudo apt-get update
sudo apt-get install -y git trimmomatic fastqc bowtie bwa \
     build-essential python2.7-dev python-numpy python-htseq default-jre \
     r-base r-base-dev r-bioc-edger
```

Make sure you're in your home directory and create the folders:

```
cd ~
mkdir TranscriptomicsWorkshop Desktop
```

Great! Now let's change directory into the folder, create a few more folders and the first README:

```
cd TranscriptomicsWorkshop
mkdir RawData QC Alignment TranscriptAbund DiffExp
mkdir QC/Trimmomatic QC/FastQC
nano README.txt
```

You are now in the terminal text editor called "nano". Enter a couple sentences describing what the project will entail, save (Ctrl + O and then hit Enter/Return), and exit (Ctrl + X).

Download the data:

```
cd RawData
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/gly7a.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/gly7b.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/gly5a.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/gly5b.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/pyr7a.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/pyr7b.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/pyr5a.fq.gz
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/pyr5b.fq.gz
```

This is a subset of data from Baker et al.. It contains the first 100,000 reads from a study looking at how *M. tuberculosis* changes its metabolism in response to different carbon sources at neutral and acidic pH.

The data are in FASTQ format and look similar to the reads we used yesterday.

Before we move on, let's create another README in the data folder and copy and paste the above lines describing the data, how we got the data, and include the date. Something like this:

```
nano README.txt
```

Inside the README:

```
Date data was downloaded: 9-25-15
Data description:

This is a subset of data from Baker et al. 2013 Mol. Micro. It contains the first
100,000 sequences from a study looking at how M. tuberculosis changes its
metabolism in response to different carbon sources at neutral and acidic pH.

Downloading the data:

git clone http://www.github.com/biobenkj/sample_data RawData

Data files:

find *.fq.gz

    gly5a.fq.gz
    gly5b.fq.gz
    gly7a.fq.gz
    gly7b.fq.gz
    pyr5a.fq.gz
    pyr5b.fq.gz
    pyr7a.fq.gz
    pyr7b.fq.gz
```

Now save (Ctrl + O and then hit Enter/Return), and exit (Ctrl + X).

Great! Let's move on to quality control and trimming our reads.

Since we covered Trimmomatic and FastQC yesterday, we will move through this section quickly, but pause for considerations in using FastQC for RNA-seq data.

4.4.1 Trimmomatic

You should be in the RawData folder currently. If you aren't:

```
cd ~/TranscriptomicsWorkshop/RawData
```

Download the adapters file:

```
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/TruSeq3-SE.fa
```

Trim the reads by scripting! (What is happening in the first line? What are the parameters to trim? What in the world are we doing in the first place?!):

```
for untrimmedreads in *.fq.gz
do
        #get the file name
        FILENAME=$(basename ${untrimmedreads%.*.*})

        #set a prefix to make understanding what has been done to the file easier
        PREFIX=trimmed

        #create the new file name (e.g. trimmedgly7a.fq.gz)
        NEWTRIMFILE=${PREFIX}${FILENAME}

        #do the trimming
        TrimmomaticSE $untrimmedreads ../QC/Trimmomatic/$NEWTRIMFILE.fq.gz \
        ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 \
        LEADING:3 \
        TRAILING:3 \
        SLIDINGWINDOW:4:15 \
        MINLEN:36
done
```

4.4.2 FastQC

Now let's generate the FastQC reports (What does that -o . thing do?):

```
cd ../QC/FastQC
for trimmedreads in ../Trimmomatic/*.fq.gz
do
        fastqc -o . $trimmedreads
done
```

It would be a good idea to make README files here to describe what happened, what script we used, etc. But for now, in light of time, we will forge ahead. Just wanted to make the point ;) Let's talk about what this script is doing so we know how we can script the analysis for all the files.

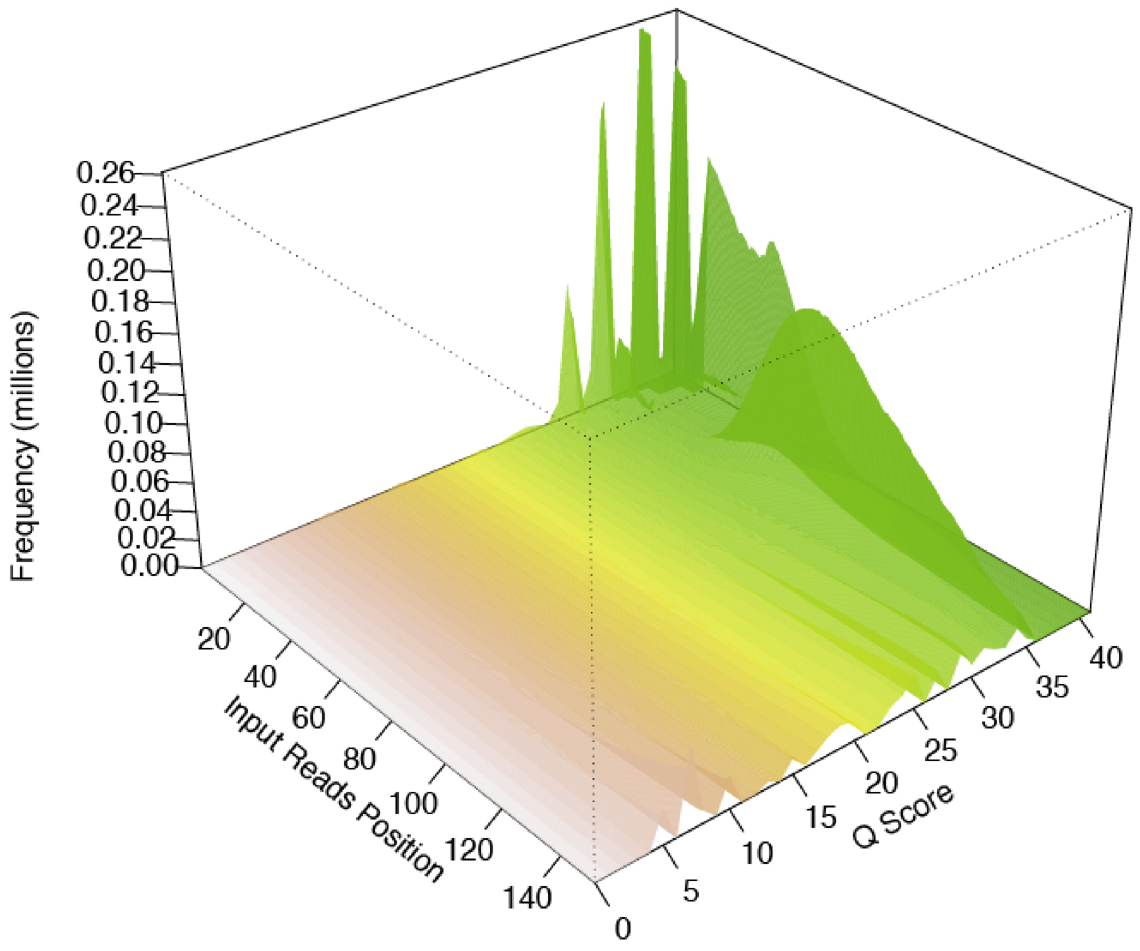
Look at the FastQC reports:

1. `trimmedgly7a_fastqc.html`
2. `trimmedgly7b_fastqc.html`
3. `trimmedgly5a_fastqc.html`
4. `trimmedgly5b_fastqc.html`
5. `trimmedpyr7a_fastqc.html`
6. `trimmedpyr7b_fastqc.html`
7. `trimmedpyr5a_fastqc.html`
8. `trimmedpyr5b_fastqc.html`

How does the data look? Any issues? Have a look at some of the FastQC documentation for the things that may not look "good" (e.g. orange and red flags for certain metrics) in relation to RNA-seq data. It's important to remember that FastQC was originally made for genome sequences.

Another tool that produces a pretty neat 3-D plot is called FaQCs.

This is what it looks like:



FaQCs takes a rather long time to run, but produces a nice pdf report when it's done.

Next: Align and count gene features

4.5 Align and count gene features

Several aligners exist for mapping reads back to a reference genome (Bowtie, Bowtie2, BWA, SOAP, TopHat, etc.). Each aligner was developed for a specific use case in mind and one should be aware of these differences. For example, TopHat (Bowtie) was optimized for identification of splice junctions. We don't have to worry about that for microbes, but the aligner itself (Bowtie) works just fine even in gene dense genomes like bacteria. Another caveat is there are *two* flavors of Bowtie: Bowtie1 and Bowtie2. Bowtie1 seems to perform better for 50 bp reads while 100+ bp reads tend to be faster using Bowtie2.

But what about the other aligners? Well, to start, look and see what other people are using in the literature, talk to your colleagues, or your friendly neighborhood bioinformatician.

The take home messages are:

- Many of these tools have great documentation, so it's worth having a look to see if they explicitly state the use case in mind and whether your experiment/data fits.
- Check the literature
- Try them out and compare the results

The next component in all of this is whether you are doing reference or reference free assembly. More on reference free assembly later. For our purposes, we have a reference.

My favorite place to download a reference genome and annotation files are from Ensembl.

4.5.1 Align the trimmed reads with Bowtie

Download the reference genome:

```
cd ~/TranscriptomicsWorkshop/Alignment
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/MtbCDC1551.fa
```

Build the index of the genome for bowtie to use to perform the mapping/alignment process:

```
bowtie-build MtbCDC1551.fa MtbCDC1551
```

The counting transcript abundance step won't read in compressed files (files that end with .gz), so we will unzip them now and hand them to bowtie (the aligner) unzipped.

We could do this in individual lines by typing something like:

```
# (don't run this!)
gunzip -c gly7a.fq.gz > gly7a.fq
```

But this would be rather tedious to do this for every file. Then we have to move the files to our Alignment folder by typing something like:

```
# (don't run this!)
mv gly7a.fq ~/TranscriptomicsWorkshop/Alignment
```

But again, this would be rather tedious to do... So what we will do is script it and combine the decompressing with moving the file into a single command with &&. This is *really* useful in a Linux type system to string two commands together. What it does is say "if the command to the left of the && works, then do the command to the right of the &&." Finally, the . in the move (mv) command means "the folder you are in right now".

So, let's do it!

Unzip the files and move the files to the Alignment folder (which we should already be in). Copy and paste the lines below into the terminal:

```
#decompress the files
for compfiles in ../QC/Trimmomatic/*.fq.gz
do
        FILENAME=${compfiles%.*.*}
        gunzip -c $FILENAME.fq.gz > $FILENAME.fq && mv $FILENAME.fq .
done
```

Now, we have been adding prefixes to files so we have an idea based on the name, how the file has been modified during the analysis. We want to strip off this prefix before moving on so that we can add a new one. To do this, we can use the move (mv) command again to rename a file.

The syntax looks something like this as an example:

```
# (don't run this!)
mv trimmedgly7a.fq gly7a.fq
```

So essentially, you have your command, mv and then the file you want to rename and then the new name of the file.

But to do this for every file is a pain and error prone. Script it!:

```
#rename the files by stripping off the trimmed prefix
for renametrim in *.fq
do
        LESSPREFIX=$(basename ${renametrim//trimmed/})
        mv $renametrim $LESSPREFIX
done
```

Whew, that was a lot of work to get to the alignment step. We are going to use Bowtie to do the alignment of the reads. Bowtie has **a lot** of options to tailor the alignment procedure. Here we only specifiy the -S option to let Bowtie know that we want our output in .sam format. The general syntax for running bowtie is as follows:

```
# (don't run this!)
bowtie -S [index file prefix from the bowtie-build command] [your input file] > [your output file tha
```

We have another new thing that looks like the greater than (>) sign. This says "take the output from the command on the left and put it in a new file on the right".

So let's align the reads:

```
#align the reads
for alignment in *.fq
do
        FILENAME=$(basename ${alignment%.*})
        PREFIX=align
        NEWALIGNFILE=${PREFIX}${FILENAME}
        bowtie -S MtbCDC1551 $FILENAME.fq > $NEWALIGNFILE.sam
done

#clean up the FASTQ files
rm *.fq
```

4.5.2 Count gene features/quantify transcript abundance with HTSeq

So now that we have aligned the reads to the genome, we want to count transcript abundance per gene using an annotation file (MtbCDC1551.gtf) that was generated when the genome is annotated. This file tells the software the gene coordinates and strandedness.

Download the .gtf file:

```
cd ../TranscriptAbund
wget http://s3-us-west-1.amazonaws.com/dib-training.ucdavis.edu/microbial-2015-09-24/MtbCDC1551.gtf
```

Let's have a look at what the first few lines of this file look like:

```
head MtbCDC1551.gtf
```

It should look something like this:

```
Chromosome  protein_coding  exon    1       1524    .       +       .        gene_id "MT0001"; transc
Chromosome  protein_coding  CDS     1       1521    .       +       0        gene_id "MT0001"; transc
    .
    .
    .
```

We can see the entire exon, its coordinates, the strand, gene_id, gene_name, etc. The software we will use (HTSeq) requires this file to "know" how to interpret the alignment file to count whether a transcript was observed.

We need to strip off prefixes again and we encounter another new command called copy (cp). The general syntax is similar to move (mv):

```
# (don't run this!)
cp [the file you want to copy] [where you want to copy the file to]
```

So let's take off the prefixes and copy the stripped files to the TranscriptAbund folder:

```
#strip off the align prefix and move the files into the TranscriptAbund folder
for renamealign in ../Alignment/*.sam
do
    LESSPREFIX=$(basename ${renamealign//align/})
    cp $renamealign ../TranscriptAbund/$LESSPREFIX
done
```

HTSeq has a spectacular function called htseq-count. This function will quantify transcript abundances for us. It also has several options we can specify, with two particularly important ones that tell it how to call whether a read is within a coding region (-m) and whether our data is stranded (–stranded). Have a look at the documentation on the three choices for the -m option. Further, if you are doing a single-end RNA-seq experiment with the Illumina TruSeq library preparation kit, your data will be reverse stranded. You can experiment with this to see by specifying –stranded=forward or –stranded=no.

The syntax is as follows:

```
htseq-count -m [how you want HTSeq to call a read in a gene region] --stranded=[reverse, forward, no]
```

Let's count!

To run the software:

```
for counts in *.sam
do
        FILENAME=$(basename ${counts%.*})
        PREFIX=abundcount_
        NEWMAPFILE=${PREFIX}${FILENAME}
        htseq-count -m intersection-nonempty --stranded=reverse $FILENAME.sam MtbCDC1551.gtf > $NEWMA
done

#clean up the .sam files
rm *.sam
```

Congratulations! We are now ready to do differential gene expression.

Next: Differential gene expression

4.6 Differential gene expression

So to this point we have QC'd our reads -> aligned them to a reference -> and counted transcript abundances. Now we are at the point where we can do differential gene expression. At this point using a statistical language like R where there are several great packages to do differential gene expression analysis and plotting is the best way to do things. So, we are going to use one commonly implemented package called edgeR. To do the analysis, we will download and run a script to produce some diagnostic plots and output we could open in Excel if we wanted.

Download the script:

```
cd ~/TranscriptomicsWorkshop/DiffExp
wget https://raw.githubusercontent.com/ngs-docs/2015-sep-microbial/master/files/diffexpref.R
```

(You can view the script contents here)

To execute the script:

```
R --vanilla --slave < diffexpref.R
```

Several different plots are produced and you can download them from the EC2 server, but I've linked them below so we can just quickly have a look at the results.

Multi-dimensional scaling plot (MDS plot)

`MDSplot.png`

Biological coefficient of variation plot (BCV plot)

`gly7vsgly5BCVplot.png`

`gly7vspyr5BCVplot.png`

`pyr7vspyr5BCVplot.png`

`gly7vspyr7BCVplot.png`

Scatter plots of significantly differentially expressed genes

`gly7vsgly5scatter.png`

`gly7vspyr5scatter.png`

`pyr7vspyr5scatter.png`

`gly7vspyr7scatter.png`

There is another neat way to visualize data through an online tool called Degust. This is an excellent tool that tends to resemble Genespring in terms of producing similar plots.

CSV files:

`download` or view

`download` or view

`download` or view

`download` or view

4.6.1 Using an available workflow that links all of these steps together

Download and install SPARTA:

```
cd ~/Desktop
git clone https://github.com/biobenkj/SPARTA_Linux
cd SPARTA_Linux
mv ExampleData ..
```

Edit the configuration file `ConfigFile.txt` by typing

```
nano ConfigFile.txt
```

and using the arrows to delete the two lines at the bottom and copy and paste this in:

```
Reference_Condition_Files: mapgly7a.sam, mapgly7b.sam
Experimental_Condition_2_Files:mapgly5a.sam, mapgly5b.sam
Experimental_Condition_3_Files:mappyr7a.sam, mappyr7b.sam
Experimental_Condition_4_Files:mappyr5a.sam, mappyr5b.sam
```

Start SPARTA in non-interactive mode:

```
python SPARTA.py --noninteractive
```

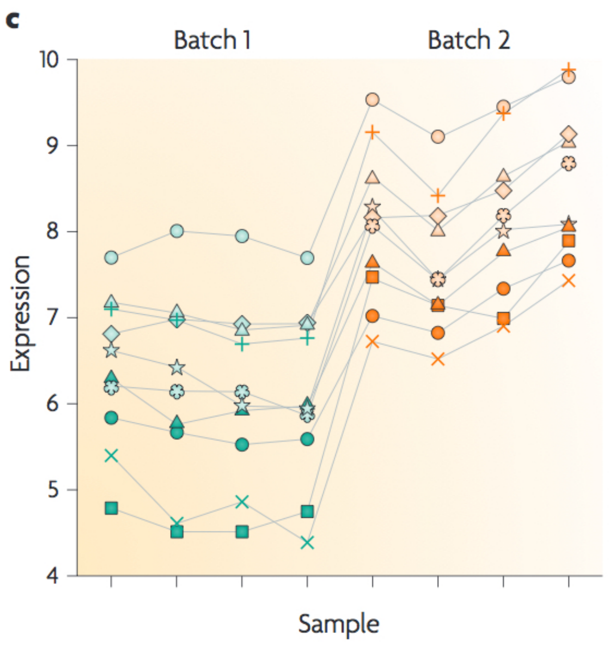
4.6.2 Identifying batch effects

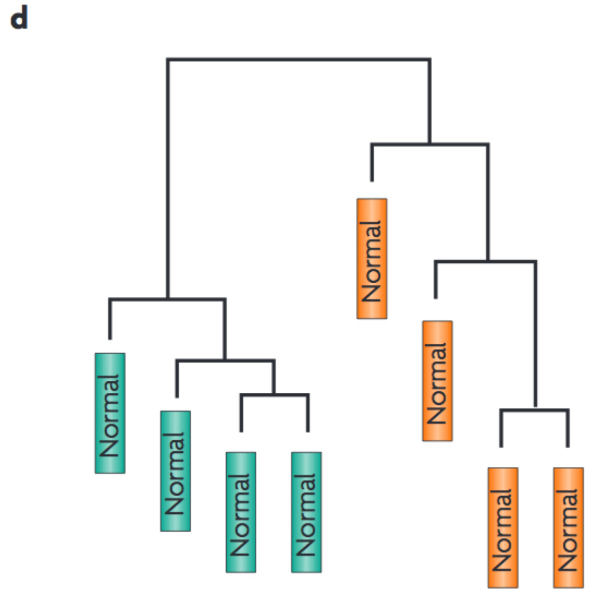
Batch effects can be a source of variation in RNA-seq data that can confound biological conclusions. In fact, there have been documented cases of batch effects present in published studies that led readers to be concerned for the validity of the results.

To quote a previously published paper in Nature Reviews Genetics, "Batch effects are sub-groups of measurements that have qualitatively different behaviour across conditions and are unrelated to the biological or scientific variables in a study. For example, batch effects may occur if a subset of experiments was run on Monday and another set on Tuesday, if two technicians were responsible for different subsets of the experiments or if two different lots of reagents, chips or instruments were used."

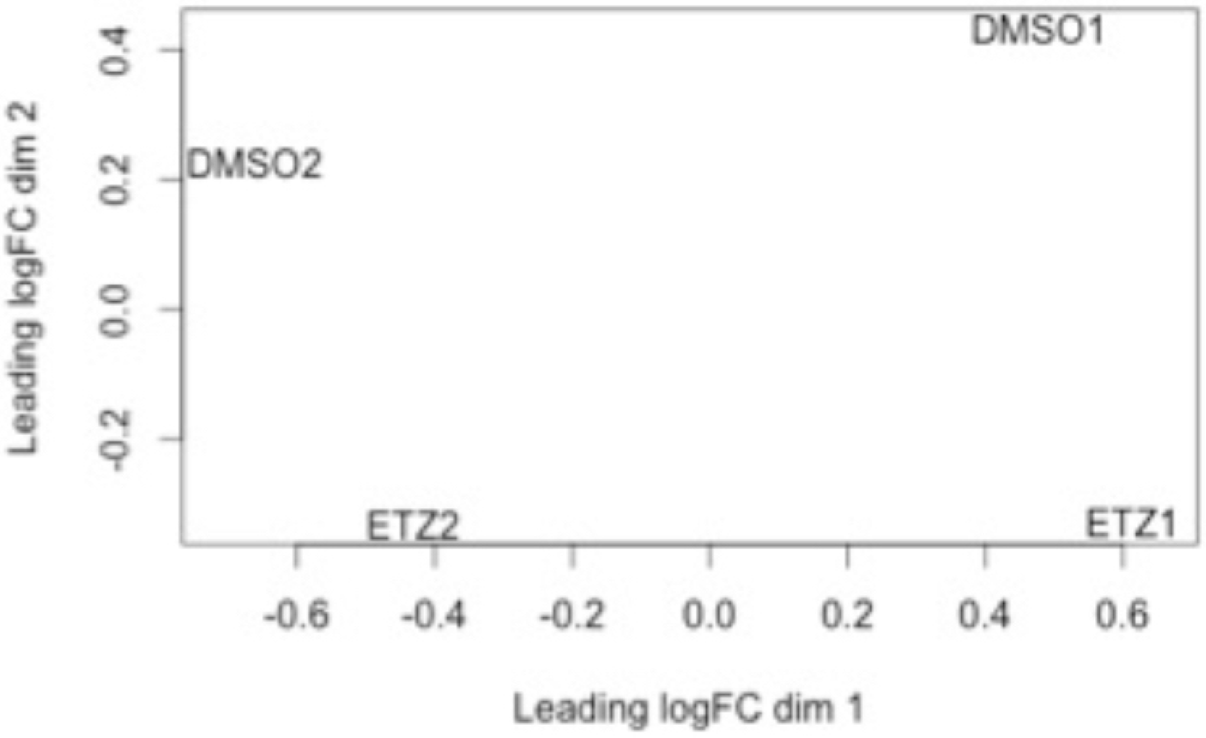
Thus, it is paramount that one address batch effects within their data before drawing biological conclusions from a specific RNA-seq experiment. To illustrate what a batch effect may look like within the data, we will utilize several different plots.

This first plot comes from the Nature Reviews Genetics paper where they examine Affymetrix data from a published bladder cancer study. You can quickly see that panels C and D from Figure 1 show that samples from batch 1 (blue) cluster together based on gene expression and samples from batch 2 (orange) cluster together.





Within RNA-seq data, using SPARTA and the MDS plot generated by edgeR, another example of batch effects within a study comparing *Mycobacterium tuberculosis* treated with a compound, we can clearly see that the mock-treated samples (DMSO) and compound-treated samples (ETZ) separate based on batch (A vs B) instead of by treatment. Ideally, we would have the samples group together based on treatment as opposed to batch.



From here, you will want to adjust your model to account for the batch effect. Within edgeR, this can be accomplished through an additive linear model. The documentation for edgeR contains a tutorial on how to deal with batch effects that can be found here.

4.6.3 Underlying assumptions (worth knowing) of differential gene expression packages (edgeR and DESeq)

When it comes to RNA-seq experiments, replication numbers tend to be small. Further, our gene counts are not normally distributed so we cannot use methods that were used for microarray data. Thus, statistical models that work well with low replicates have been developed.

Here is a spectacular discussion by Meeta (https://github.com/ngs-docs/msu_ngs2015/blob/master/hands-on.Rmd) on how replication number leads to fewer differences (greater power of detection) between using either edgeR or DESeq to do differential gene expression.

The take home message is that both DESeq and edgeR use a similar model (negative binomial), but the way the dispersions are estimated are different. In a kind of crazy, yet awesome, study that utilized 48 replicates! they found that the negative binomial model is a good approximation, 6 replicates is best (though not always feasible), and that the method implemented by edgeR (one gene is squeezed towards a common dispersion calculated across all genes) performed best.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search