# Bioinformatics training on VM and cloud technologies

An educational option in the GTPB programme

**Pedro L. Fernandes**
Instituto Gulbenkian de Ciência
Oeiras, PT



# The GTPB Programme

GTPB delivers hands-on intensive training in Portugal since 1999

>300 training courses
>4700 participants

We try to tackle Bioinformatics user needs

We seek improvement in training methods

# GTPB in 2015

**IB15F - Introductory Bioinformatics** (w/ 2d NGS intro) *April 13-17*

**PHSMCP15 - Promoter hunting and systems modelling of cellular pathways** *April 22-24*

**SMLMC15 - Structural Modelling for Large Macromolecular Complexes** *April 27-30*

**ARANGS15 - Automated Reproducible Analysis of NGS Data** *May 11-15*

**MEVR15 - Molecular Epidemiology of Viruses using R** *May 25-28*

**IBSTATB15 - Introductory Biostatistics for Biologists** *June 15-19*

**PGDH15 - Population genetics and demographic history: model-based approaches** *July 6-9*

**BPBR15 - Bioinformatics using Python for Biomedical Researchers** *July 20-24*

**PDA15 - Proteomics Data Analysis** *September 7-10*

**GACT15 - Genomic Architecture of Complex Traits** *September 28 – October 1*

**TBHD15 - Translational Bioinformatics in Health and Disease** *November 30 - December 3*

**IB15S - Introductory Bioinformatics** (w/ 2d NGS intro) *December 14-18*

# ARANGS12 - The motivation

In 2012, a new course design was proposed.
The motivation came from the observation that NGS workflows were often irreproducible.

We went after the most plausible reasons:

- Authors do not use revision control in their scripts
- Not concerned about workflow portability
- Did not foresee consequences of software updates
- Did not consider recording parameter changes
- Ignored that data itself is not (necessarily) stable...

# ARANGS12 Design

**Best practices**

- Standardized project organization
- Projects fully 'runnable' without user intervention
- No loss of data, metadata or source code through versioning

**Technologies**

- Next generation sequencing platforms
- File formats
- Command-line executables, command line scripting and batching
- High-level programming with domain-specific toolkits
- Revision control systems
- Workflow environments (both visual and command line)

**Use cases**

- Phylogenetic placement of metagenomic data
- Typing of pathogens
- Comparative analysis of multicellular genomic data

# Improving towards ARANGS16

**Best practices**

Command line scripting of analysis steps

Provisioning systems to standardise software environment requirements

Packaging of compute environment into static, portable units

Sharing of compute environment packages

**Technologies**

Next generation sequencing platforms

Command-line executables, command line scripting and batching

Provisioning Systems: **Puppet, Dockerfile**

Virtualisation with **Virtualbox** and **Vagrant**

Containerization with **Docker**

# ARANGS16 team


# Vagrant

Managed, scriptable virtualisation with a single file that contains a RUBY script

Command line launching at the host
Zhost> vagrant init
Zhost> vagrant up
Zhost> vagrant ssh
Zhost> vagrant halt
Zhost> vagrant destroy

# Simple Vagrantfile

```
Vagrant.configure do |config|
  config.vm.box = "olbat/tiny-core-micro"
  config.vm.provider "virtualbox" do |vb|
      # Display the VirtualBox GUI when booting the machine
      vb.gui = true
      # Customize the amount of memory on the VM:
      vb.memory = "2048"
  end
  config.ssh.shell="sh"
end
```

# Vagrant BOX

A vagrant box contains minimal functional elements
- Package manager
- SSH
- SSH user
- Provisioning (Chef or Puppet, etc)

Vagrant boxes easily created for Cloud providers (AWS, etc.) using Packer.
( see https://www.vagrantup.com )

# Puppet

Software to specify a manifest, equivalent to saying in a json file:
I want my system to have
wget, bzip2, tar, git, build-essential, gzip, zlibg-dev, ncurses-dev

Provisioning of the guest is performed via a host command
Zhost> vagrant provision

# Comparable Vagrant & Docker features

As seen by the course participants at the end of day 4

## Docker vs. Vagrant

| Docker | Vagrant |
|---|---|
| potential security nightmare | sandboxed |
| lightweight | entire OS |
| linux dependent | standalone |
| requires `docker` | requires `virtualbox` and `vagrant` |
| docker language | uses ruby |
| invoke tools from outside | log in on environment |
| ??? | deploy on grid |
| resource efficient | less so due to virtualization |
| works on MPI cluster | not so much |
| docker hub | vagrant cloud |

# Virtualisation example

**SuperSmart**
Self-Updating Platform for Estimating Rates of Speciation and Migration, Ages, and Relationships of Taxa
https://peerj.com/preprints/501v1/
http://www.supersmart-project.org/

On Github: https://github.com/naturalis/supersmart

Box: https://atlas.hashicorp.com/Naturalis/boxes/supersmart/

Tutorial: Building a phylogenetic tree of the large cats
https://atlas.hashicorp.com/Naturalis/boxes/supersmart/versions/0.1.22

Slides:
http://www.slideshare.net/rvosa/assembling-the-tree-of-life-from-public-dna-sequence-data

# The environment of a course

… can be replicated using virtualisation.

See this example GTPB

http://bioinformatics-core-shared-training.github.io/ndarc16/

Look for the docker file and inspect it.

Thank you for your attention

pfern@igc.gulbenkian.pt

http://gtpb.igc.gulbenkian.pt