

Scientific Programming Practical 1 (QCB)

--> Data science in A103

Introduction

Outline

- ❖ Personal introduction
- ❖ Introduction to the practical
- ❖ Hands-on practical

About me

Computer Science

Ph.D. at the University of Verona, Italy, with thesis on Simulation of Biological Systems

Research Fellow at Cranfield University - UK

Three years at Cranfield University working at proteomics projects (GAPP, MRMAid, X-Tracker...)

Module manager and lecturer in several courses of the MSc in Bioinformatics

Bioinformatician at IASMA – FEM

Currently bioinformatician in the Computational Biology Group at Istituto Agrario di San Michele all'Adige – Fondazione Edmund Mach, Trento, Italy

Collaborator uniTN - CiBio

I ran the Scientific Programming Lab for QCB for the last couple of years

Fondazione Edmund Mach

FEM – San Michele, Trento - Italy



Agricultural Institute

Research and Innovation Centre

Genomics, metabolomics wet labs on
fruits (apple, grape, small fruits,...)

Bioinformatics and computational biology

Bioinformatics @FEM (UBC)

❖ Genomics

- Assembly and annotation of complex genomes (plants, insects, etc.)
- Development of SNP Chips for genetic screening
- Resequencing of genomes / Variant discovery

❖ Metagenomics

- Targeted metagenomic data
- *Feature selection* algorithms
- Algorithms for strain-level identification from un-targeted metagenomics

❖ Transcriptomics

- RNA-seq data analysis, gene and pathway enrichment
- Data integration and compilation of expression atlases

❖ Metabolomics

- Data analysis pipelines for targeted and untargeted data
- Methods for MS imaging

❖ Statistical data analysis

- Integration of -omic data and analysis of correlation networks



Bix @FEM - Examples

Genome assembly

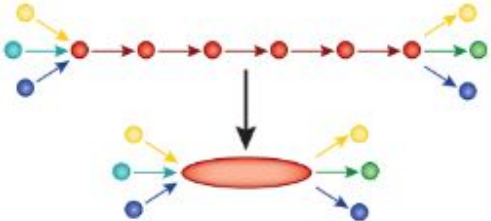
1. Fragment DNA and sequence



2. Find overlaps between reads

...AGCCTAGACCTACAGGATGCGCGACACGT
GGATGCGCGACACGT CGCATATCCGGT

3. Assemble overlaps into contigs



4. Assemble contigs into scaffolds



In a nutshell... (Tunis' version...)



Reads



Assembled genome

[Virgil and the Muses, Bardo Museum, Tunis]

[from M. Baker, Nature Methods, 2014]



Bix @FEM - Examples

Genome assembly of DH of Pear and Apple

Input data:

Illumina: ~60x – 100x PE information + (mate pairs for Apple)

Pacific Biosciences ~ 30x + 30x (35x only for Apple)

Bionano optical maps: ~ 600x (for both)

Hi-C: pear only

Genetic maps: integrated genetic map from 21 mapping populations (Apple only)

Output result (example for Apple):

Chromosome scale assembly

Contigs: 2150 for a total of **625Mb**

N50 Contigs (hybrid dbg2olc): ~ **620Kbps**

280 **Scaffolds**, for an N50 **5,6Mb**

17 chromosomes + lg0 unanchored sequences



[Daccord et al, Nature Genetics, 49, 2017; Linsmith et al., GigaScience, to appear 2020]



Bix @FEM - Examples

SNP-Chips development for GWAS

20K SNP Illumina Infinium II Array (reseq of 16 Apple cultivars, Illumina 30x)

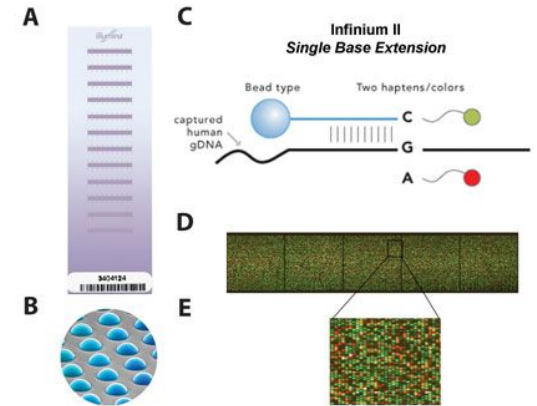
487K SNP Affymetrix Axiom Array (reseq of 63 Apple cultivars, Illumina 20-30x)

600K SNP Affymetrix Axiom Array Walnut (reseq. 18 cultivars, Illumina 80x)

1. Reads alignment and filtering
2. SNP calling
3. Identification of most reliable SNPs
4. Selection of (20K) 487K target SNPs

Several Terabytes of data produced!!!!

Peach, pear and walnut done too!





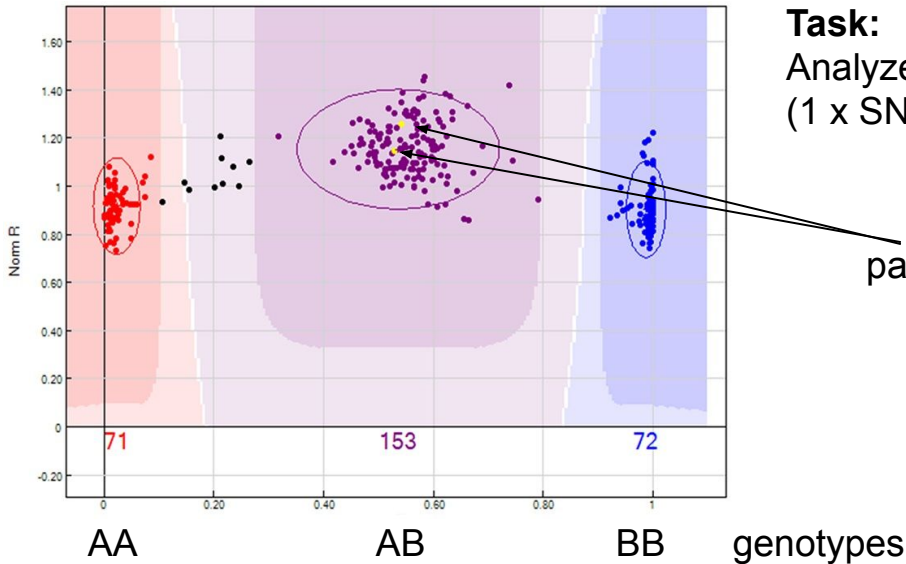
Bix @FEM - Examples

SNP-Chips development for GWAS

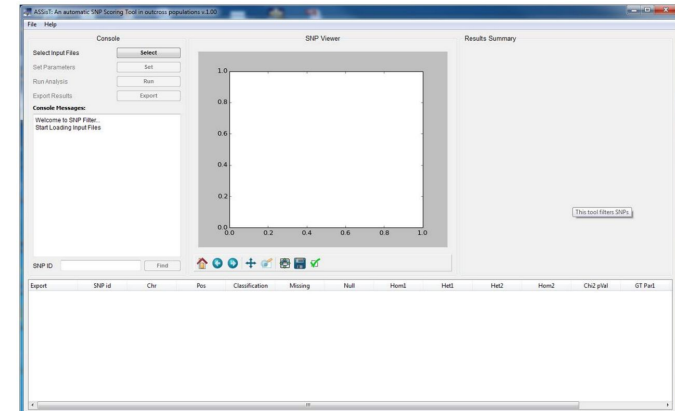
20K SNP Illumina Infinium II Array (reseq of 16 Apple cultivars, Illumina 30x)

487K SNP Affymetrix Axiom Array (reseq of 63 Apple cultivars, Illumina 20-30x)

600K SNP Affymetrix Axiom Array Walnut (reseq. 18 cultivars, Illumina 80x)



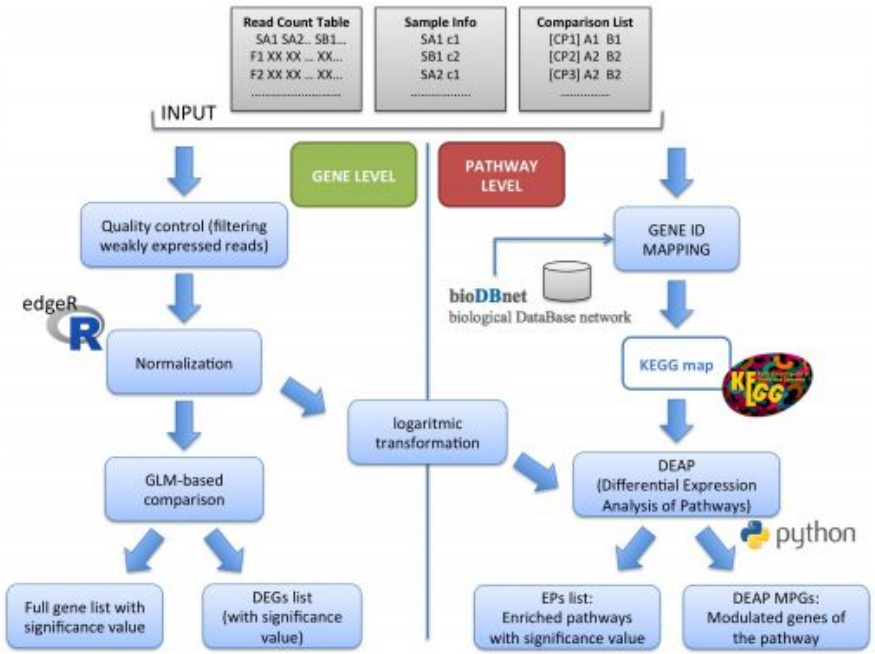
ASSiST



[Di Guardo et al., Bioinformatics, 2015]

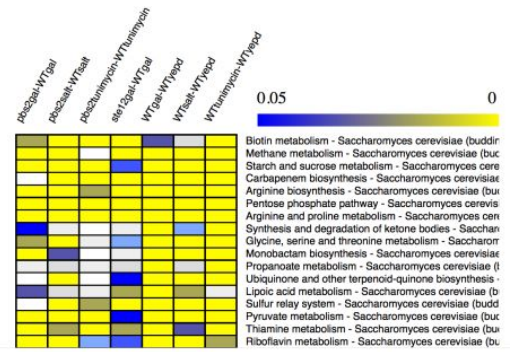
Bix @FEM - Examples

RNAseq data analysis with Pathway Inspector



Bix @FEM - Examples

RNAseq data analysis with Pathway Inspector

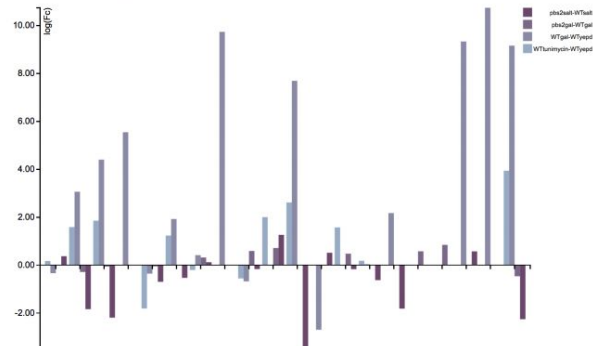


- Biotin metabolism - Saccharomyces cerevisiae (buddr)
- Methane metabolism - Saccharomyces cerevisiae (buc)
- Starch and sucrose metabolism - Saccharomyces cere
- Carbapenem biosynthesis - Saccharomyces cerevisiat
- Arginine biosynthesis - Saccharomyces cerevisiae (buc)
- Pentose phosphate pathway - Saccharomyces cerevis
- Arginine and proline metabolism - Saccharomyces cere
- Synthesis and degradation of ketone bodies - Sacchar
- Glycine, serine and threonine metabolism - Saccharor
- Monobactam biosynthesis - Saccharomyces cerevisiae
- Propanoate metabolism - Saccharomyces cerevisiae (l)
- Ubiquinone and other terpenoid-quinone biosynthesis -
- Lipoic acid metabolism - Saccharomyces cerevisiae (b
- Sulfur relay system - Saccharomyces cerevisiae (budd
- Pyruvate metabolism - Saccharomyces cerevisiae (buc)
- Thiamine metabolism - Saccharomyces cerevisiae (buc)
- Riboflavin metabolism - Saccharomyces cerevisiae (bu

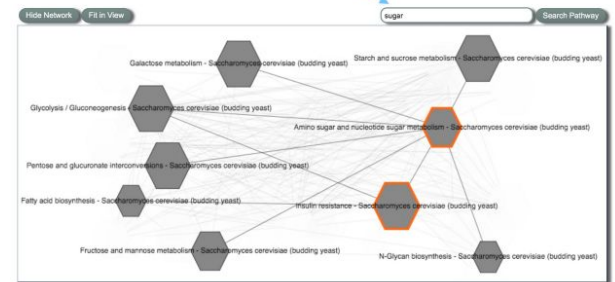
Intersection pbs2salt-WTsalt WTtunmycin-WTyeast pbs2gal-WTgal WTgal-WTyeast

Gene	Comparison	P-value	Fold Change	FDR
YOR04W	pbs2salt-WTsalt	0.02648082029846	0.2420313772254	0.0481391748189205
	WTtunmycin-WTyeast	2.38460349315057e-07	0.524172138340184	4.4748733128663e-07
	pbs2gal-WTgal	0.000678984853540209	-0.301105831814261	0.02297830818505987
YJL140W	WTgal-WTyeast	2.86768386434163e-15	0.83218983220768	7.1912488226014e-15
	pbs2salt-WTsalt	3.89247942741733e-24	0.833568919065478	1.88549837902352e-07
	WTtunmycin-WTyeast	1.98618787171733e-24	1.0016409182983	5.92391638177541e-24
YLR130C	pbs2gal-WTgal	1.87828251172735e-20	0.8076405182865	1.4986205124024e-08
	WTgal-WTyeast	0.30349129423675144	0.378549277440217	0.05482106278608931
	pbs2salt-WTsalt	0.30148182019198813	0.318821501388888	0.0308287774872549
YOR011W	WTtunmycin-WTyeast	3.8143420277802e-66	-1.98795788914384	3.8332116878679e-65
	WTgal-WTyeast	2.78703316888143e-09	0.488214138134561	2.3895374291843e-08
	pbs2salt-WTsalt	1.00574728111917e-16	0.72183053688741	8.28174875473501e-16

Comparative analysis for Amino sugar and nucleotide sugar metabolism - Saccharomyces cerevisiae (budding yeast)



Cross-Comparison Results



Bix @FEM - Examples

Pedigree-based haplotype visualization

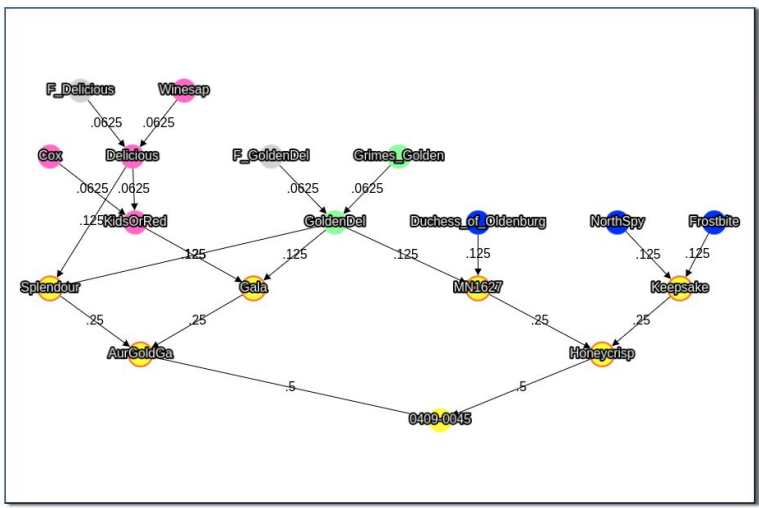
Family of sample: 0409-0045

0409-0045

Fit in View

View global stats

Pedigree browse



Choose one ancestor or other accessions

- Generation 1
 - Honeycrisp
 - AurGoldGa
 - Generation 2
 - Keepsake
 - MN1627
 - Gala
 - Splendour
 - Generation 3
 - Generation 4
 - Generation 5
- Relatedness 0-10% [2]
 - Relatedness 10-20% [2]
 - Relatedness 20-30% [17]
 - Relatedness 30-40% [182]
 - Relatedness 40-50% [183]
 - Relatedness 50-60% [9]
 - Relatedness 60-70% [0]
 - Relatedness 70-80% [0]
 - Relatedness 80-90% [0]
 - Relatedness 90-100% [0]

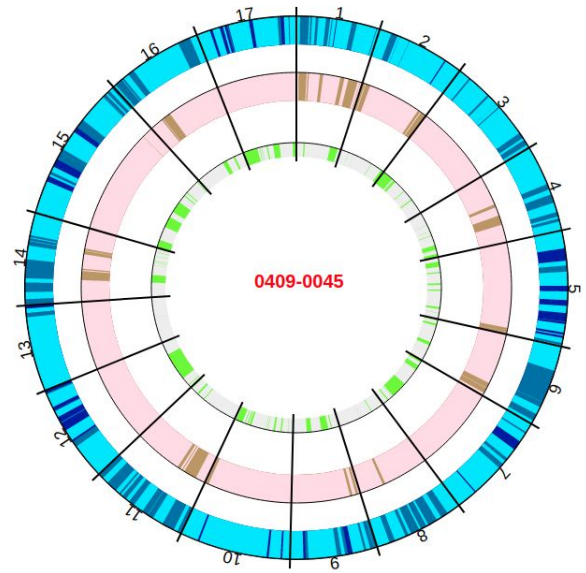
Legend
 Sample Mother Father Both No data

Comparison without pedigree check Comparison with pedigree check [Analyze](#) [Relatedness](#)

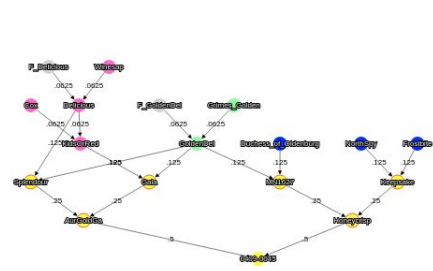
Temporary access: <http://10.234.110.141:8081>

Bix @FEM - Examples

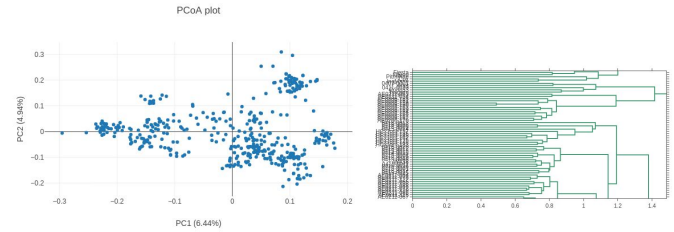
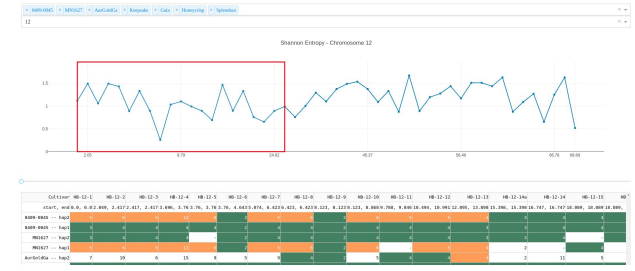
Pedigree-based haplotype visualization



Pedigree of 0409-0045



Cultivars	Relatedness	Color	Legend
MN1627	0.39	1CE6FF	<input checked="" type="checkbox"/>
AurGoldGa	0.55	FF4A46	<input checked="" type="checkbox"/>
Keepsake	0.42	008FA6	<input checked="" type="checkbox"/>
Gala	0.48	FFDBE5	<input checked="" type="checkbox"/>
Honeycrisp	0.52	0000A6	<input checked="" type="checkbox"/>
Splendour	0.48	B79762	<input checked="" type="checkbox"/>



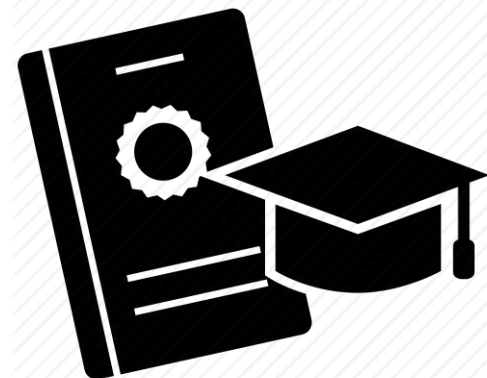
Opportunities @FEM

MSc External thesis

Are you interested in a bioinformatics project in NGS data analysis, RNA Seq, data integration?

Come and talk to me or email me at:

luca.bianco@fmach.it



Scientific Programming Practical

Back to business now!



Scientific Programming Practical

In this practical you will

1. Install Python 3.x (and pip)
2. Install Visual Studio Code
3. Get familiar with the Python console
4. Start using Visual Studio Code and advanced features (like debugging)
5. End the session with some exercises

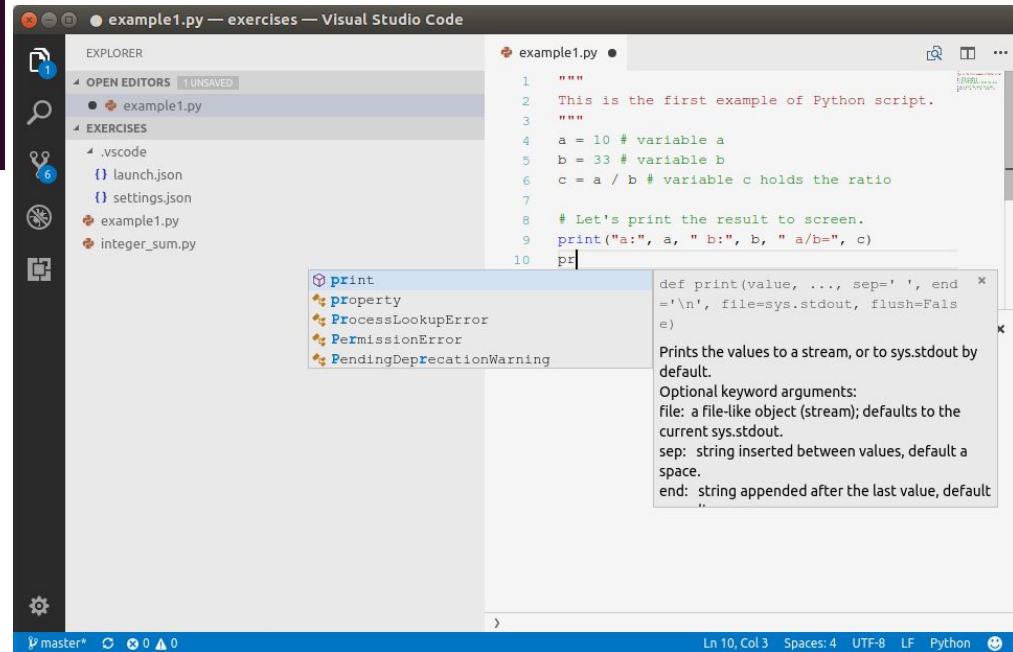


Scientific Programming Practical

Console VS. Integrated Development Environment (IDE)

```
biancol@bluhp:~$ python3
Python 3.5.2 (default, Aug 18 2017, 17:48:00)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

```
>>> print("Hi there")
Hi there
>>> print("{} + {} = {}".format(10,20, 10+20))
10 + 20 = 30
>>> █
```



The screenshot shows the Visual Studio Code IDE interface. The Explorer view on the left shows a project named 'exercises' with files like 'launch.json', 'settings.json', 'example1.py', and 'integer_sum.py'. The Editor view shows 'example1.py' with the following code:

```
1 """
2 This is the first example of Python script.
3 """
4 a = 10 # variable a
5 b = 33 # variable b
6 c = a / b # variable c holds the ratio
7
8 # Let's print the result to screen.
9 print("a:", a, " b:", b, " a/b=", c)
10 pr
```

A tooltip for the `print` function is visible, showing its signature and documentation:

```
print(value, ..., sep=' ', end='
', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.

Optional keyword arguments:
file: a file-like object (stream); defaults to the current sys.stdout.
sep: string inserted between values, default a space.
end: string appended after the last value, default "
```

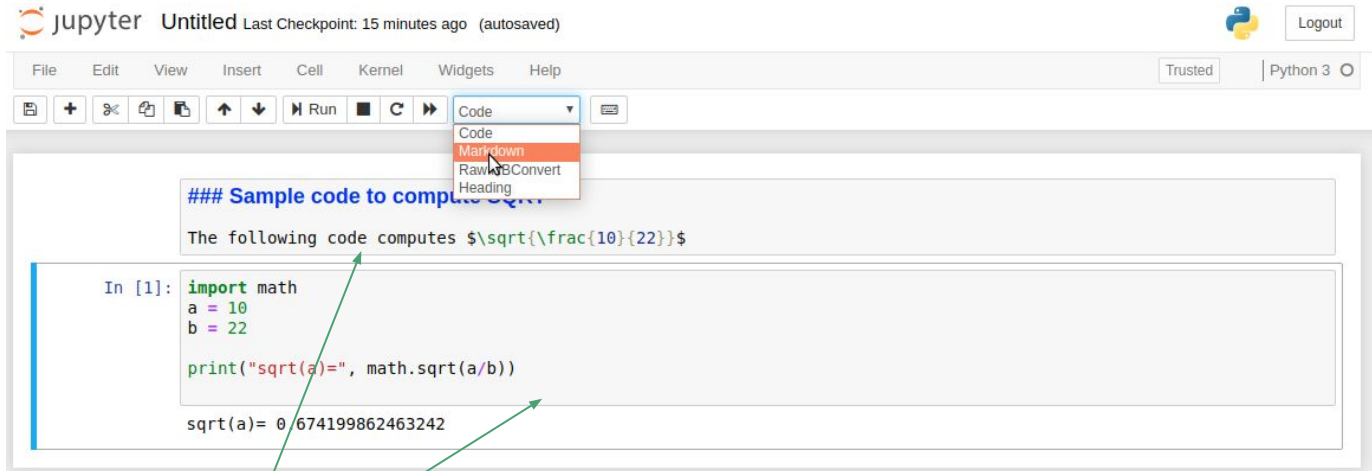
The status bar at the bottom indicates the current file is 'master*', the cursor is at 'Ln 10, Col 3', and the encoding is 'UTF-8 LF Python'.

Notebooks and Jupyter

“Jupyter is a web-based interactive development environment for python/R.. notebooks, code, and data.”

Notebooks contain both the **code**, some **text describing the code** and the **output of the code execution**,

Jupyter is becoming the de-facto standard for writing technical documentation.



The screenshot shows the Jupyter Notebook interface. At the top, it says "Jupyter Untitled Last Checkpoint: 15 minutes ago (autosaved)". There is a "Logout" button and a "Python 3" indicator. The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar is a toolbar with icons for adding, deleting, and running cells. A dropdown menu is open over the toolbar, showing options: Code, Code, Markdown, Raw, Convert, and Heading. The main area contains a code cell with the following content:

```
### Sample code to compute  $\sqrt{\frac{10}{22}}$ 

The following code computes  $\sqrt{\frac{10}{22}}$ 

In [1]: import math
a = 10
b = 22

print("sqrt(a)=", math.sqrt(a/b))

sqrt(a)= 0.674199862463242
```

Two green arrows point from the word "Cells" at the bottom to the code cell and the output area.

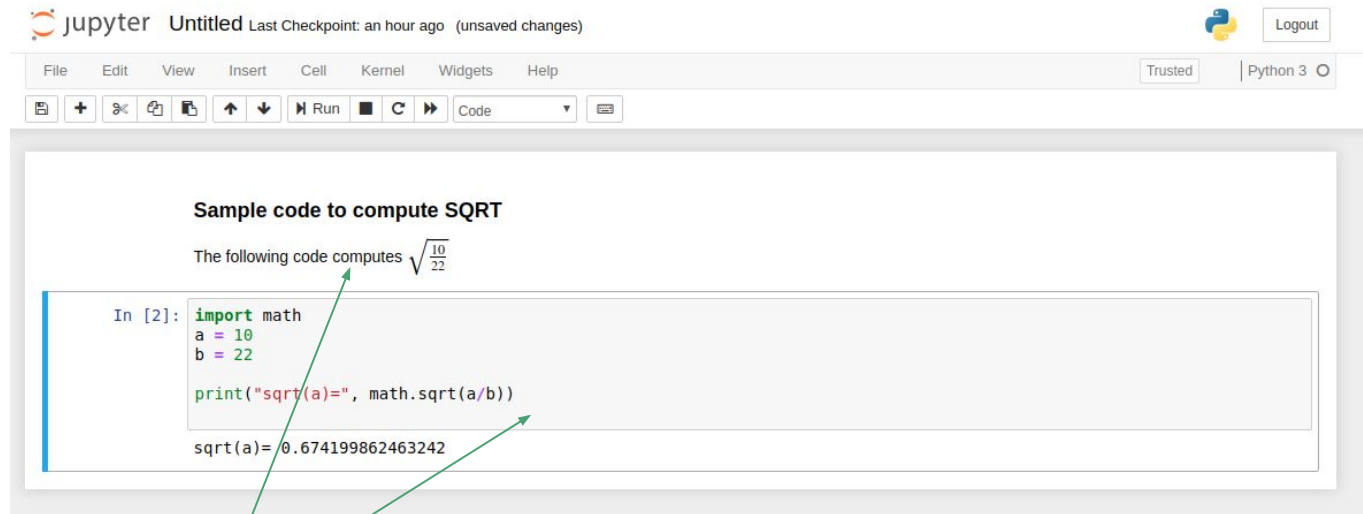
Cells

Notebooks and Jupyter

Notebooks contain both the **code**, some **text describing the code** and the **output of the code execution**,

Jupyter is becoming the de-facto standard for writing technical documentation.

A cell can be executed by clicking on **Run**



The screenshot shows a Jupyter Notebook interface. At the top, the title bar reads "jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)". The top right corner features a "Logout" button and a "Python 3" indicator. Below the title bar is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, Help. A toolbar below the menu bar contains icons for file operations, a "Run" button, and a "Code" dropdown menu. The main content area displays a code cell with the following text:

Sample code to compute SQRT

The following code computes $\sqrt{\frac{10}{22}}$

```
In [2]: import math
a = 10
b = 22

print("sqrt(a)=", math.sqrt(a/b))
```

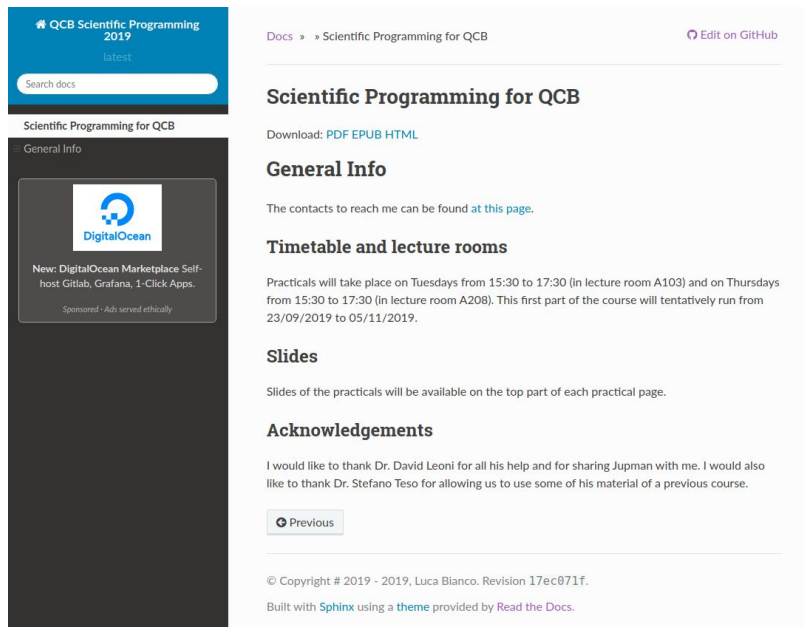
sqrt(a)= 0.674199862463242

Cells
(after Run)

Resources

All material regarding practicals will be found here:

<http://qcbsciprolab2019.readthedocs.io>




The screenshot shows the Read the Docs interface for the project 'Scientific Programming for QCB'. The top navigation bar includes the project name, a search bar, and a link to 'Edit on GitHub'. The main content area features a 'Download' section with links for PDF, EPUB, and HTML. Below this is a 'General Info' section with contact information, a 'Timetable and lecture rooms' section with a detailed schedule, and an 'Acknowledgements' section. A 'Previous' button is visible at the bottom of the main content area. The footer contains copyright information and mentions the Sphinx framework and a theme provided by Read the Docs.

QCB Scientific Programming 2019
latest

Search docs

Scientific Programming for QCB

General Info


New: DigitalOcean Marketplace Self-host Gitlab, Grafana, 1-Click Apps.
Sponsored - Ads served ethically

Docs » » Scientific Programming for QCB [Edit on GitHub](#)

Scientific Programming for QCB

Download: [PDF](#) [EPUB](#) [HTML](#)

General Info

The contacts to reach me can be found [at this page](#).

Timetable and lecture rooms

Practicals will take place on Tuesdays from 15:30 to 17:30 (in lecture room A103) and on Thursdays from 15:30 to 17:30 (in lecture room A208). This first part of the course will tentatively run from 23/09/2019 to 05/11/2019.

Slides

Slides of the practicals will be available on the top part of each practical page.

Acknowledgements

I would like to thank Dr. David Leoni for all his help and for sharing Jupman with me. I would also like to thank Dr. Stefano Teso for allowing us to use some of his material of a previous course.

[Previous](#)

© Copyright # 2019 - 2019, Luca Bianco, Revision 17ec071f.
Built with [Sphinx](#) using a [theme](#) provided by Read the Docs.



luca.bianco@fmach.it

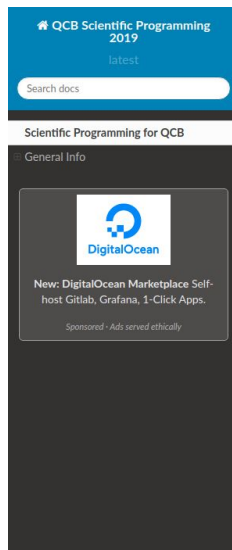
Timetable

Tuesdays:

A107: 15,30 - 17,30

Thursdays:

A107: 15,30 - 17,30



Docs » » Scientific Programming for QCB

[Edit on GitHub](#)

Scientific Programming for QCB

Download: [PDF](#) [EPUB](#) [HTML](#)

General Info

The contacts to reach me can be found [at this page](#).

Timetable and lecture rooms

Practicals will take place on Tuesdays from 15:30 to 17:30 (in lecture room A103) and on Thursdays from 15:30 to 17:30 (in lecture room A208). This first part of the course will tentatively run from 23/09/2019 to 05/11/2019.

Slides

Slides of the practicals will be available on the top part of each practical page.

Acknowledgements

I would like to thank Dr. David Leoni for all his help and for sharing Jupman with me. I would also like to thank Dr. Stefano Teso for allowing us to use some of his material of a previous course.

[Previous](#)

<http://qcbsciprolab2019.readthedocs.io>

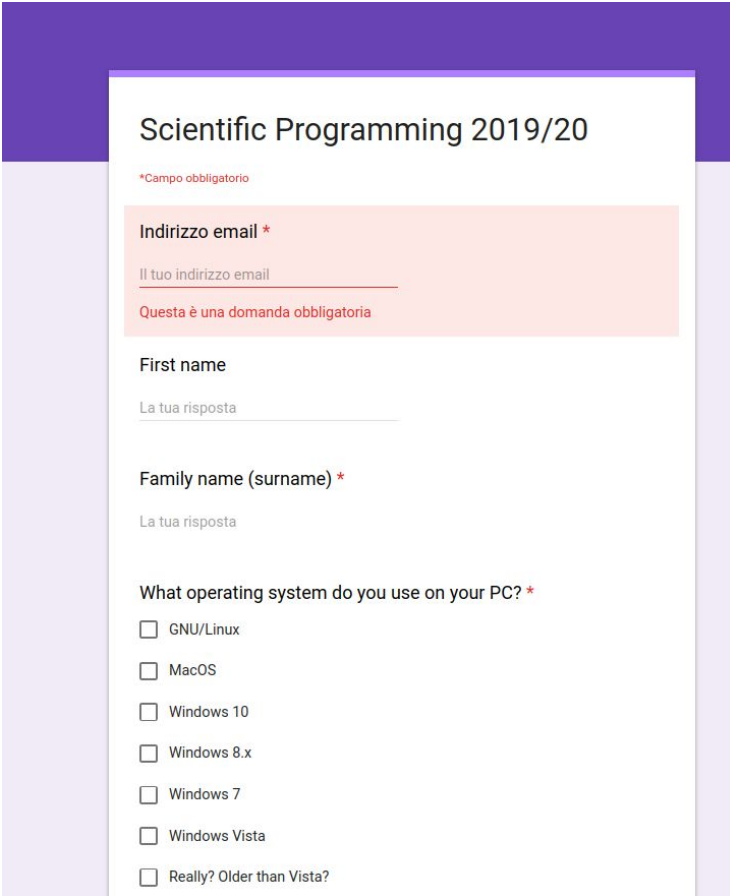


luca.bianco@fmach.it

Please, fill the form at

<https://tinyurl.com/y6nInx7l>

Deadline Sunday, September 29th



Scientific Programming 2019/20

*Campo obbligatorio

Indirizzo email *

Il tuo indirizzo email

Questa è una domanda obbligatoria

First name

La tua risposta

Family name (surname) *

La tua risposta

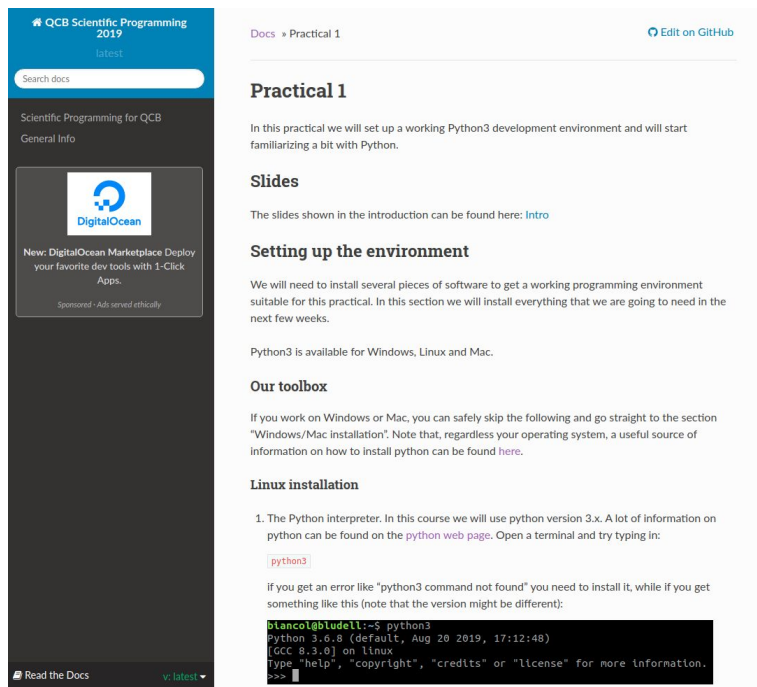
What operating system do you use on your PC? *

- GNU/Linux
- MacOS
- Windows 10
- Windows 8.x
- Windows 7
- Windows Vista
- Really? Older than Vista?

Any questions?

If not, please go to:

<https://qcbsciprolab2019.readthedocs.io/en/latest/introduction.html>




QCB Scientific Programming 2019

latest

Search docs

Scientific Programming for QCB

General Info

 DigitalOcean

New: DigitalOcean Marketplace Deploy your favorite dev tools with 1-Click Apps.

Sponsored - Ads served ethically

Docs • Practical 1 [Edit on GitHub](#)

Practical 1

In this practical we will set up a working Python3 development environment and will start familiarizing a bit with Python.

Slides

The slides shown in the introduction can be found here: [Intro](#)

Setting up the environment

We will need to install several pieces of software to get a working programming environment suitable for this practical. In this section we will install everything that we are going to need in the next few weeks.

Python3 is available for Windows, Linux and Mac.

Our toolbox

If you work on Windows or Mac, you can safely skip the following and go straight to the section "Windows/Mac installation". Note that, regardless your operating system, a useful source of information on how to install python can be found [here](#).

Linux installation

1. The Python interpreter. In this course we will use python version 3.x. A lot of information on python can be found on the [python web page](#). Open a terminal and try typing in:

```
python3
```

If you get an error like "python3 command not found" you need to install it, while if you get something like this (note that the version might be different):

```
blanco@bludell:~$ python3
Python 3.6.8 (default, Aug 20 2019, 17:12:48)
[GCC 8.3.0] on Linux
Type "help", "copyright", "credits" or "license" for more information.
->>
```



luca.bianco@fmach.it