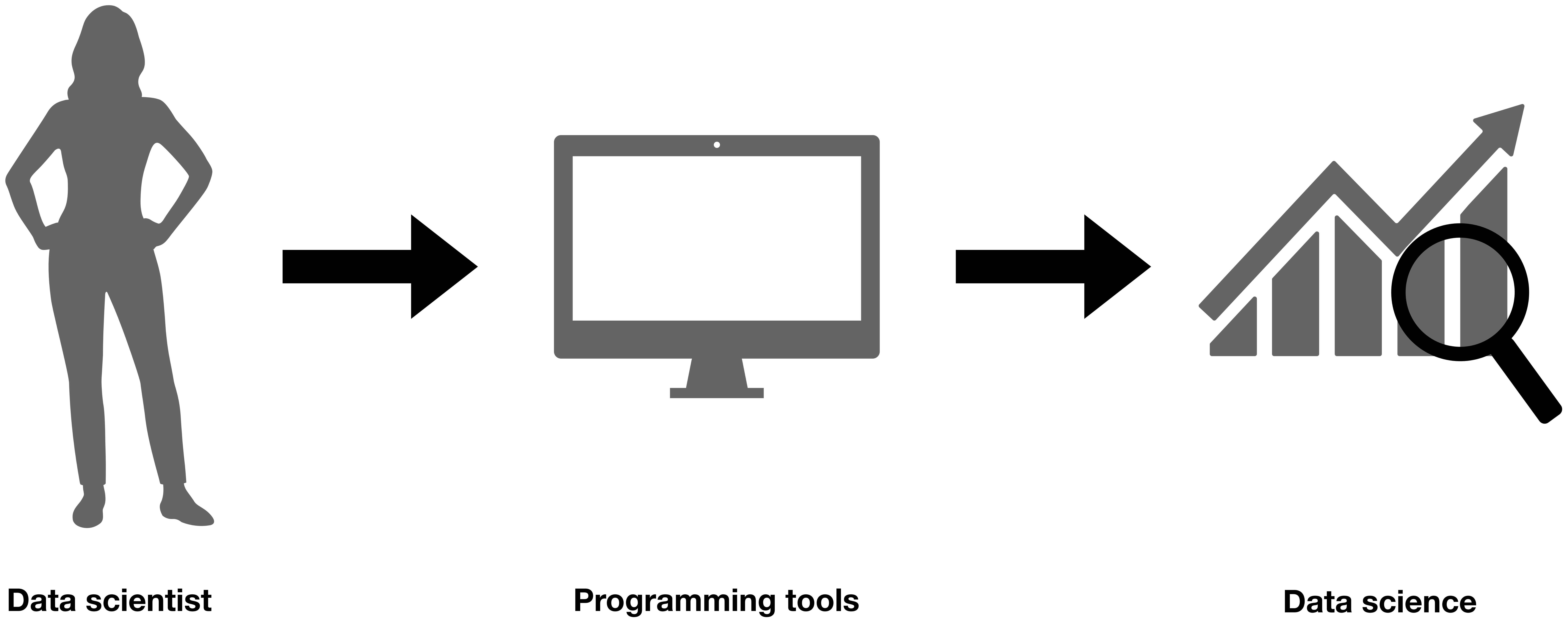
Current Topics in Media Computing and HCI

Data Science Programming

Krishna Subramanian, M.Sc.
Media Computing Group
RWTH Aachen University

Summer Semester 2020

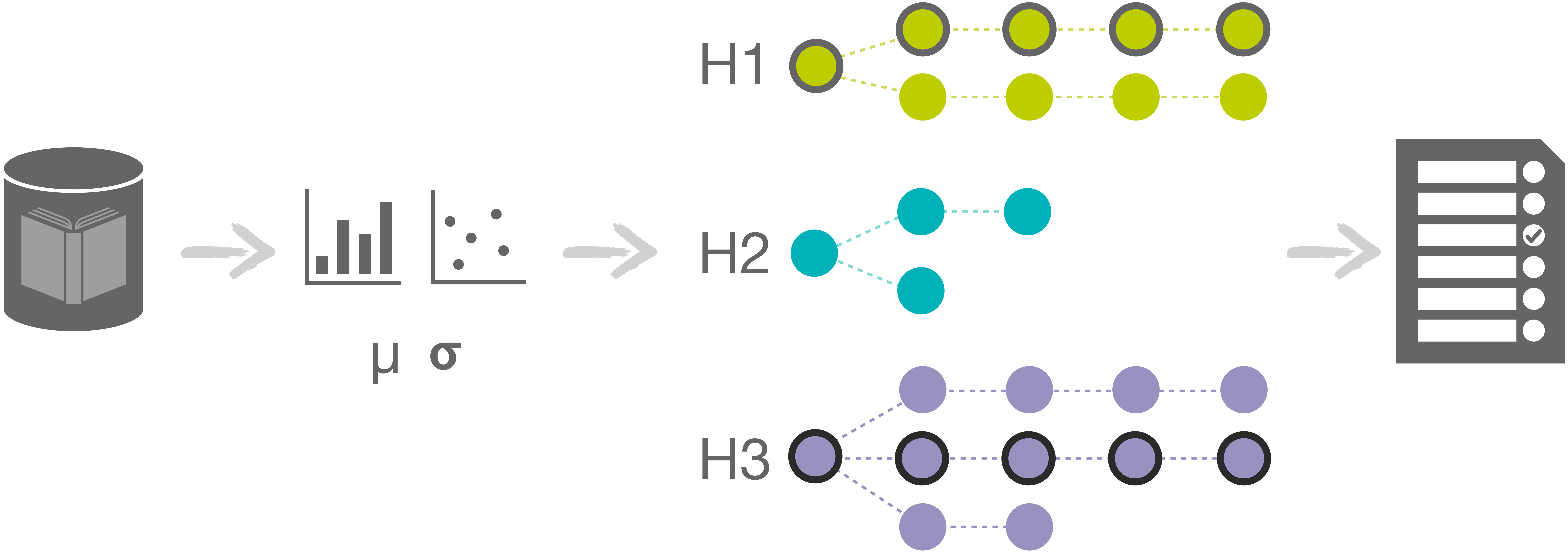
https://hci.rwth-aachen.de/cthci


Data scientist
Programming tools
Data science

Data Science

- How we extract knowledge from data
- Example applications: Targeted advertising/recommendations in Amazon/Netflix, validate research findings, and train a robot to detect humans
- What is so special about data science?
  - **Open-ended, iterative workflow** that involves a lot of backtracking
  - Usually involves *deliberate* **bad programming practices** like writing non-modular code, not using version control, and code hoarding

Data Science Workflow



[Subramanian et al., TRACTUS: Understanding and Supporting…, 2020]

Open-ended: Goals were not predefined, but were defined (and modified) during analysis

Iterative and involves backtracking: Previous analysis is revisited during analysis and afterwards when writing the final version of source code

Programming Practice in Data Science

- In data science, the focus is the end goal (results, findings) but not the process
  - Contrast this to software engineering, where the process (source code) also needs to be well documented, run fast, be secure, be memory efficient, etc.

[Kery et al., Exploring Exploratory Programming, 2017]

Messy Code

- *“I know how to write code. And I know that I could write functions to reuse functions and I could try to modularize things better, and sometimes I just don't care because why am I going to put effort in that if I'm not going to use it again?”*

- Leads to source code that is hard to re-use, navigate, and understand

```
# read the CSVs - each DV is a separate file
measure =
read.csv("Measurements_a_bit_clean.csv", sep =
";")

#apply the change to the hours.
measure <- within(measure, Time[(Schema == 1 &
IsHours == 1 & Time >= 6)|(Schema == 1 & IsHours
== 0 & Time >= 30)|(Schema == 2 & Time >= 10)|
(Schema == 3 & IsHours == 1 & (Time != 12&Time !=
9&Time != 6&Time != 3))|(Schema == 3 & IsHours ==
0 & (Time != 0&Time != 45&Time != 30&Time !=
15))] <- 'difficult')
measure <- within(measure, Time[Time !=
"difficult"] <- 'easy')

# change the column type of the experimental
design (expansive vs. constrictive)
measure$User <- as.factor(measure$User)
measure$Time <- as.factor(measure$Time)
measure$Schema <- as.factor(measure$Schema)
measure$IsHours <- as.factor(measure$IsHours)
```

[Kery et al., Variolite: Supporting Exploratory …, 2017]

Informal Versioning and Code Hoarding

- Remember that data science involves iteration and backtracking
  - This requires data scientists to keep all source code from explorations

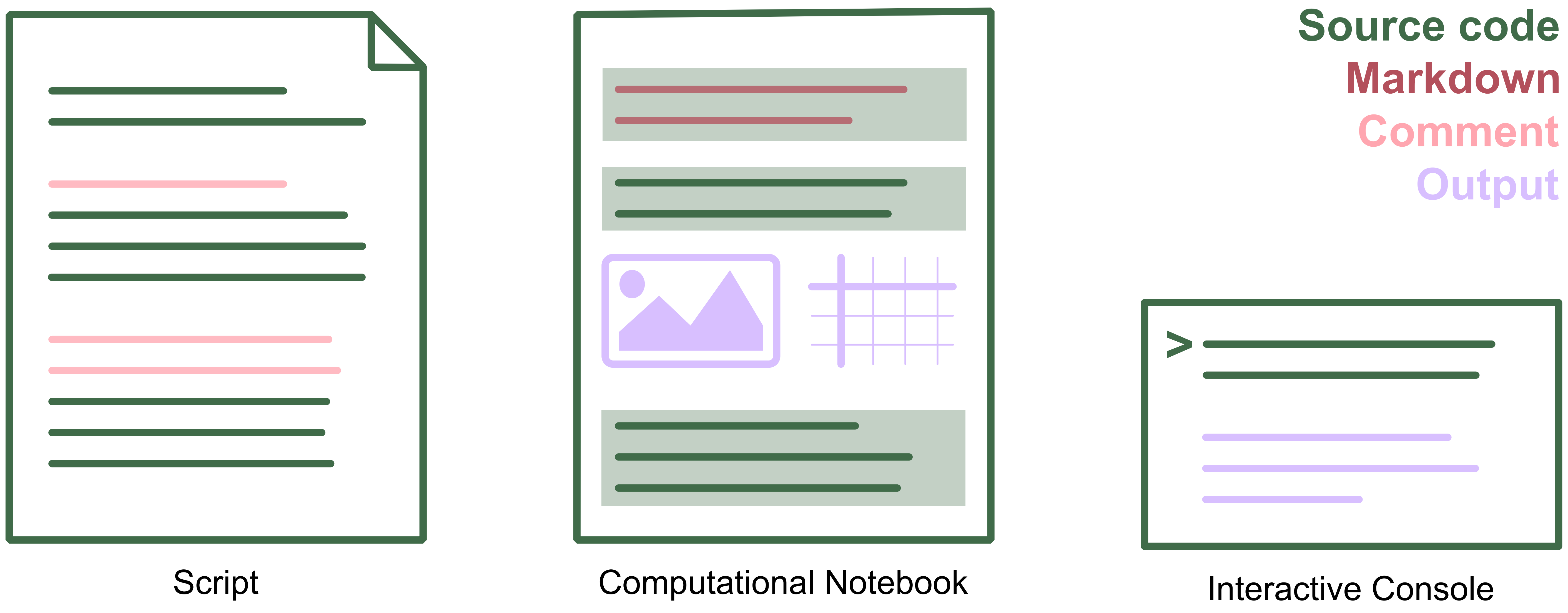
```
print "hc policy distribution entropy: \n" + str
#print "2-fold CV Var: \n" + str(CV_variances)

#print "Stdev of (V_M under h^f, avg over states
#print "Stdev of (V_M under h^c, avg over states
#print "Average V_M under h^f: \n" + str(V_M_hf)
#print "Average V_M under h^c: \n" + str(V_M_hc)

fig = plt.figure(figsize=(9,4))
ax = fig.add_subplot(1,1,1)
ax.set_xticks(D_sizes)
plt.plot(D_sizes, [V_star_avgState]*len(D_sizes)
#plt.plot(D_sizes, AvgState_V_hfs)
plt.errorbar(D_sizes,AvgState_V_hfs,yerr=Std_Vs_
#plt.plot(D_sizes, AvgState_V_hcs)
plt.errorbar(D_sizes,AvgState_V_hcs,yerr=Std_Vs_
plt.xlim(0,max(D_sizes))
plt.xlabel('|D|')
plt.ylabel('mean(V)')
plt.grid()
plt.show()

'''
fig = plt.figure(figsize=(10,4))
ax = fig.add_subplot(1,1,1)
ax.set_xticks(D_sizes)
```

[Kery et al., Variolite: Supporting Exploratory …, 2017]

Programming Tools

- Scripting languages are very common: R, Python, and MATLAB

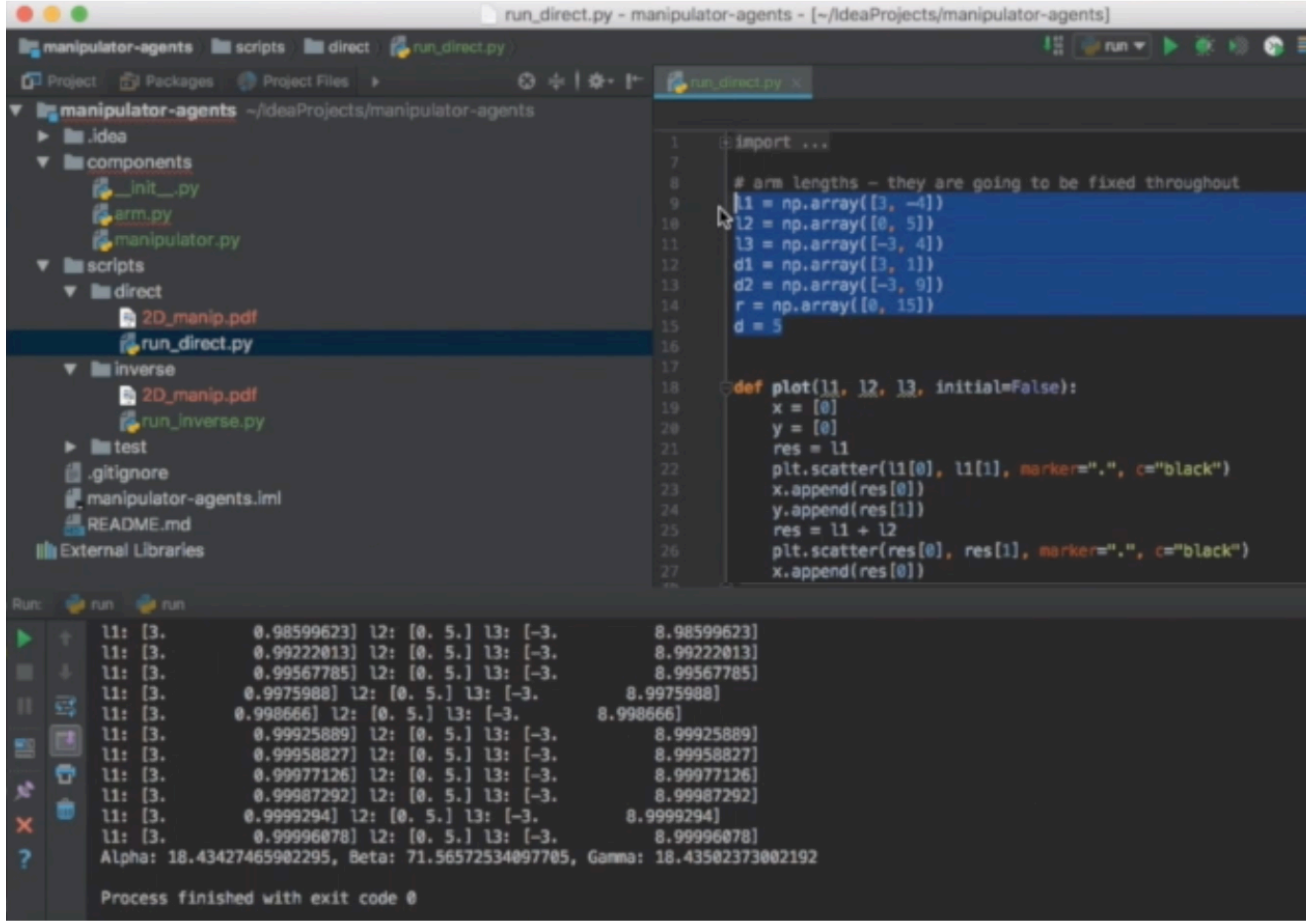
- IDEs for these languages offer three interfaces to program in
  - Scripts
  - Computational notebooks
  - Consoles



[Subramanian et al., Casual Notebooks and …, 2020]

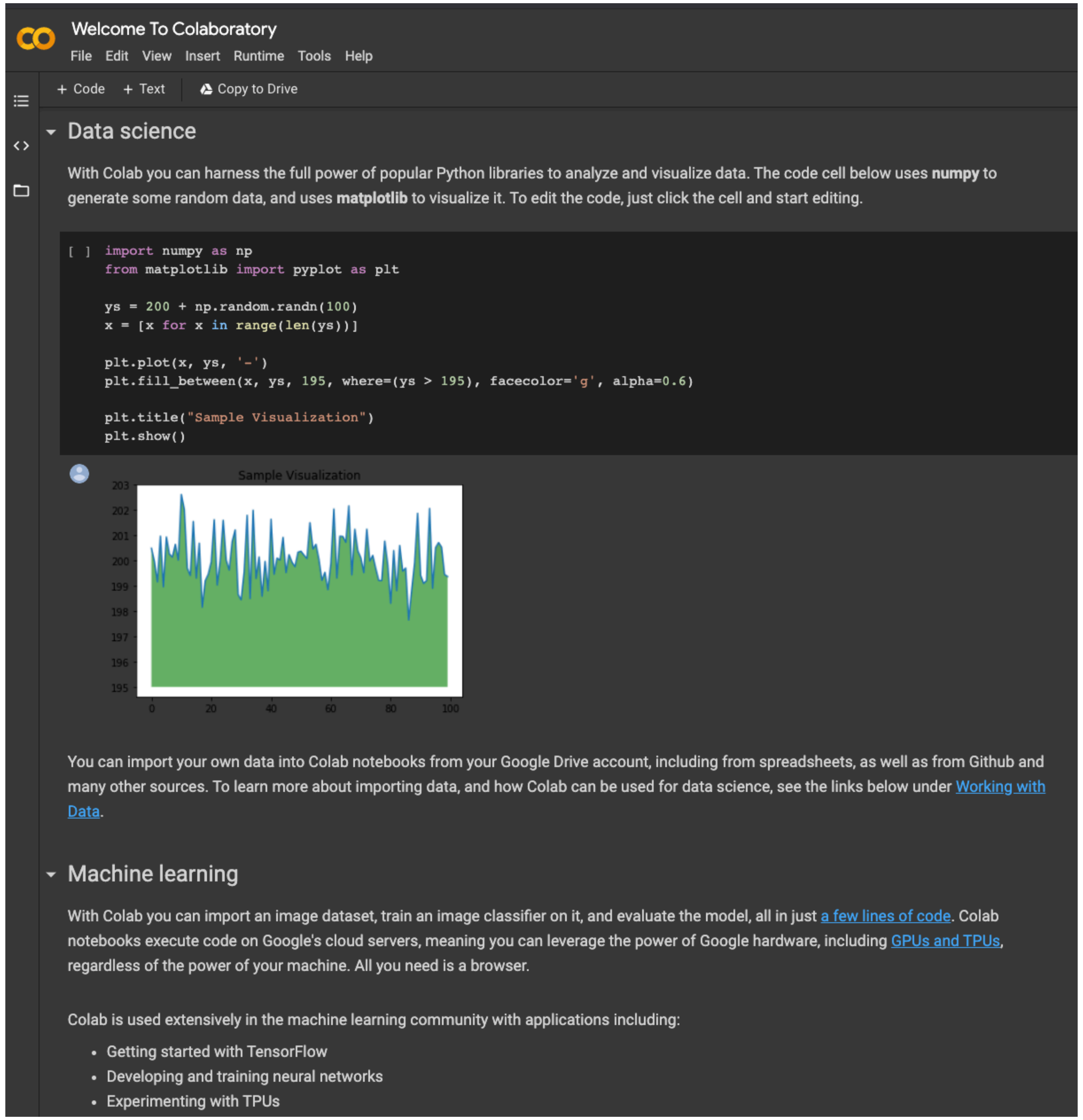
Scripts

- Traditional way to store source code
- Supports complete/partial execution of source code (via selection)
- Output is shown in a separate window or the console window



Computational Notebooks

- Cell-based programming
- Cells can be executed in a non-sequential order
- Output is shown immediately next to the cell that was executed
- Cells can also include Markdown commands (to describe the analysis process)



Scripts vs. Notebooks

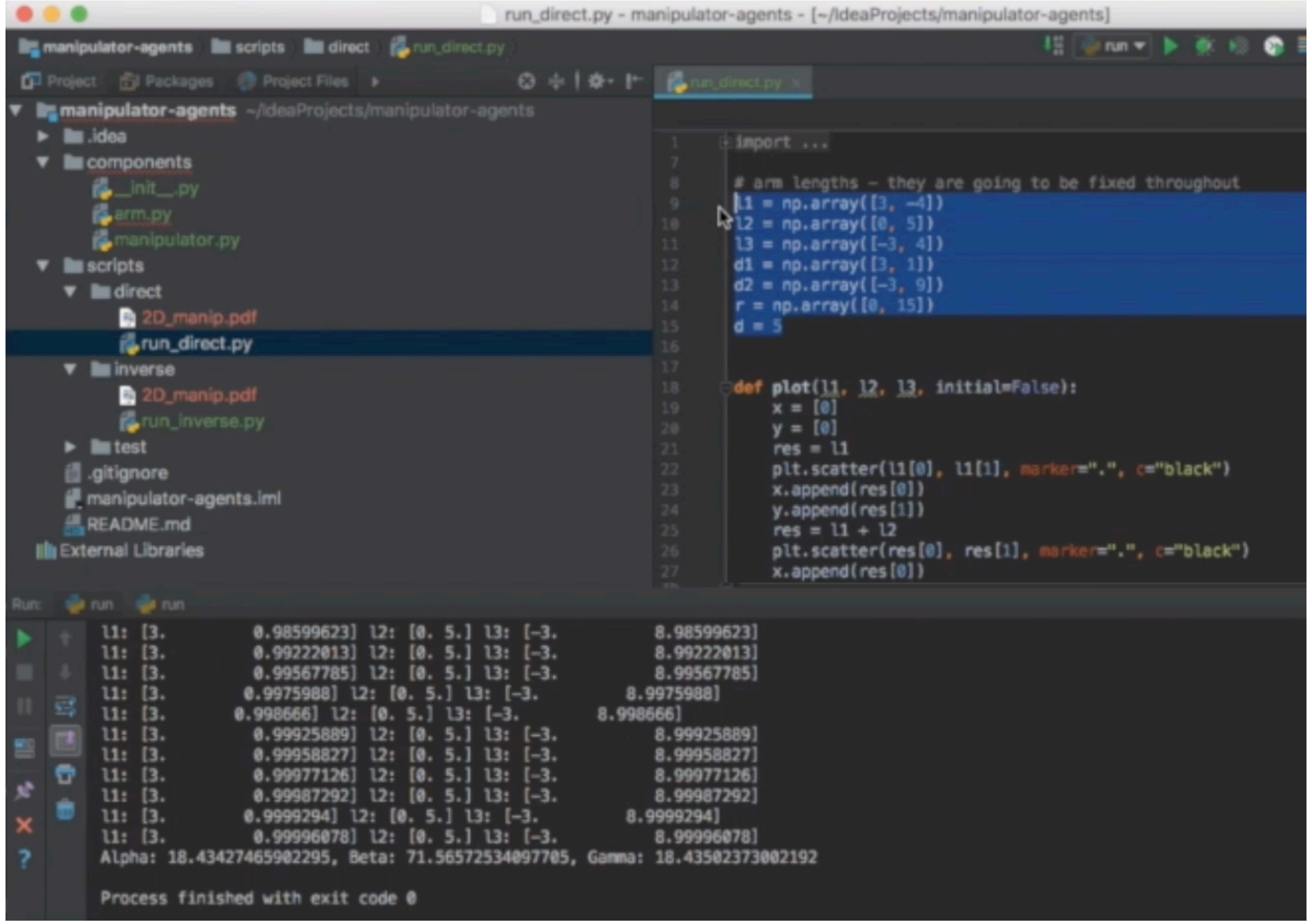
Data Science Task	Notebooks	Scripts
Experimentation	✅	
Refactor code		✅
Present code	✅	
Share code	✅	
Execute from command line/on GPU		✅

[Subramanian et al., Casual Notebooks and …, 2020]

Interactive Consoles

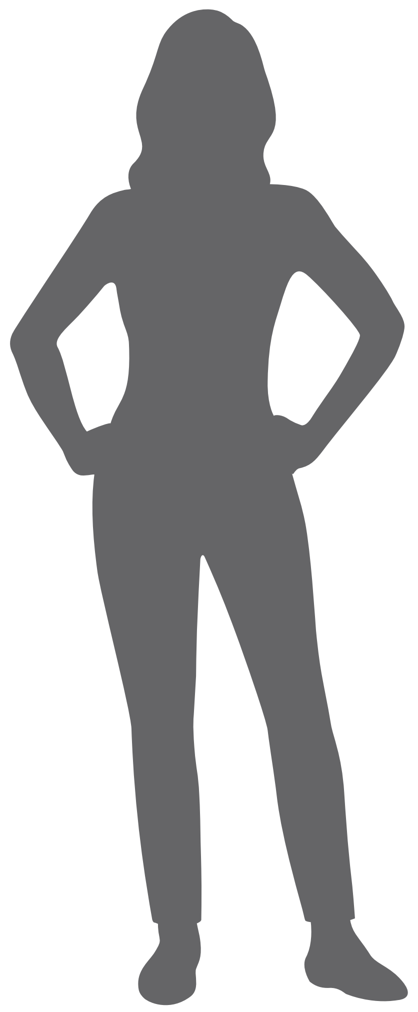
- Used mostly for secondary tasks like testing API, loading libraries, etc.

- However: Novice data workers who do not use notebooks tend to use consoles even for their primary data science work



[Subramanian et al., Casual Notebooks and …, 2020]

Data Scientists



- "… people who understand how to fish out answers to important business questions from today's tsunami of unstructured information." [Davenport 2012]

- Data scientists are
  - impactful,
  - help make key decisions

- Several data scientists are not professionals, we call them "data workers"
  - do not have formal training in data science
  - may not have good programming practices

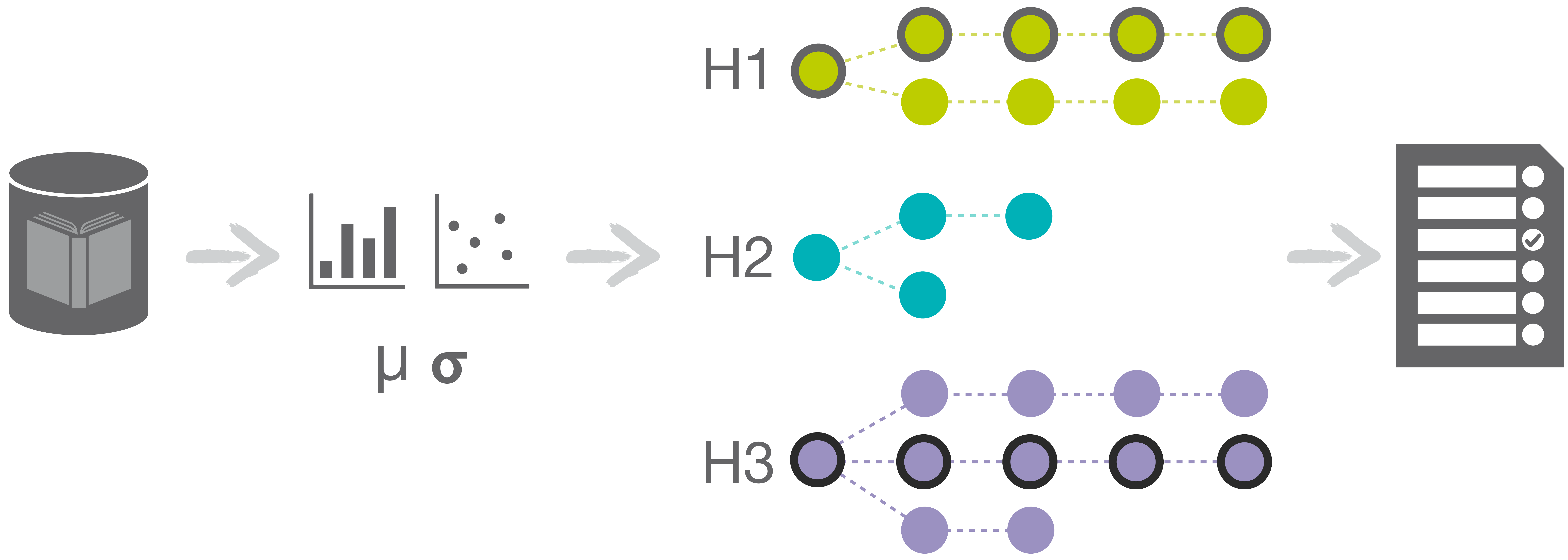
[Boukhelifa et al., How Data Workers…, 2017]

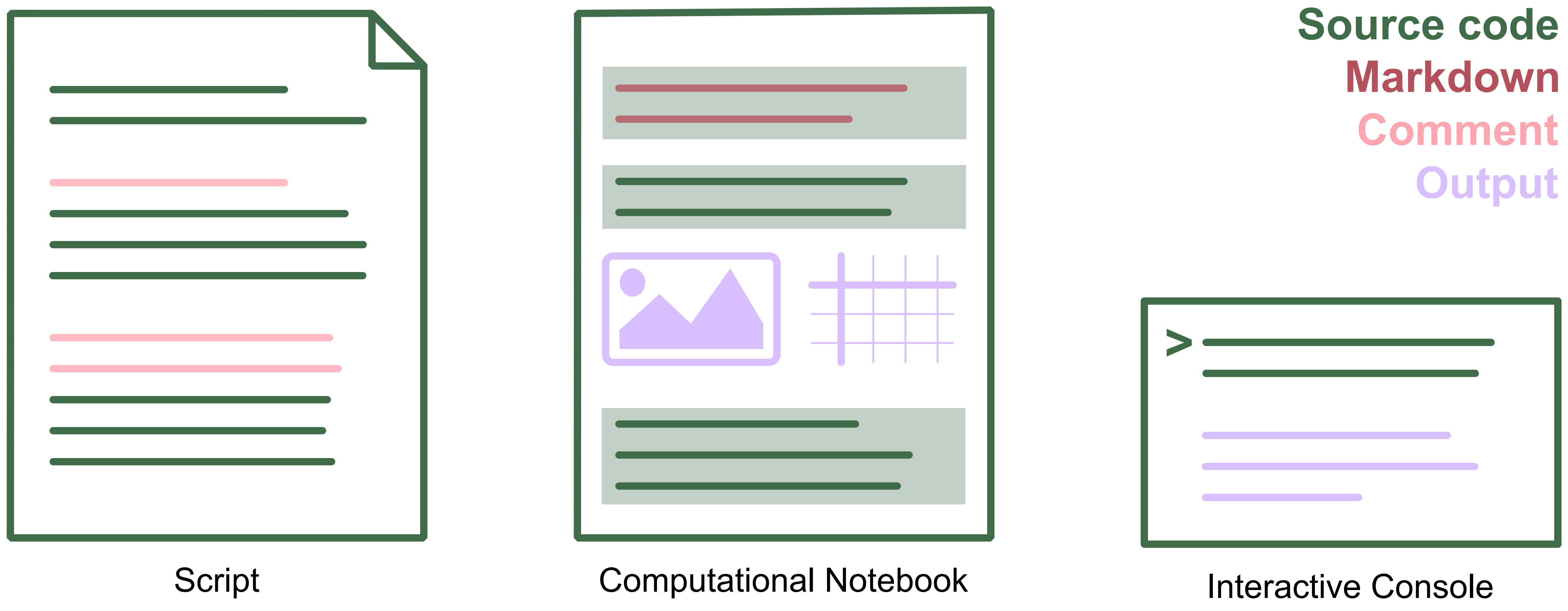
Future of Data Science Research

- Notebooks: Collaboration, better support for use in production, history navigation, etc.

- Understanding data science workflows across several fields like machine learning, significance testing, etc.

- Data science in AR, VR, and tabletops





- Data is everywhere, and data science is a valuable skill to have in the current day and age

- Improving data scientists' workflows and tools can be vastly beneficial!