Preface

Over the course of the past twenty years I have learned many things relevant to this book while working in high-energy physics. As everyone in this field in the yearly to mid-90s, I was analyzing experimental data collected by particle colliders using the FORTRAN programming language. Then, gradually, I moved to C++ coding following the general trend at that time. I was not too satisfied with this transition: C++ looked overly complicated and C++ source codes were difficult to understand. With C++, we were significantly constrained by particular aspects of computer hardware and operating system (Linux and Unix) on which the source codes were compiled and linked against existing libraries. Thus, to bring the analysis environment outside the high-energy community to the Windows platform, used by most people, was almost impossible.

I began serious development of ideas that eventually led to the jHepWork Java analysis environment in 2004, when I was struck by the simplicity and by the power of the Java Analysis Studio (JAS) program developed at the SLAC National Accelerator Laboratory (USA). One could run it even on the Windows platform, which was incredible for high-energy physics applications; We never really used Windows at that time, since high-energy physics community had wholly embraced Unix and Linux as the platform of choice, together with its build-in GNU C++ and FORTRAN compilers. More importantly, JAS running on Windows had exactly the same interface and functionality as for Unix and Linux! It was a few months after that I made the decision to focus on a simplified version of this Java framework which, I thought, should befit from Java scripting, will be simpler and more intuitive. Thus, it should be better suited for general public use. I have called it "jHepWork" ("j" means Java, "HEP" is the abbreviation for high-energy physics, and "work" means a sedentary lifestyle in front of a computer monitor).

Indeed, I was able to simplify the language and semantic of the JAS analysis environment by utilizing more appropriate short names for classes and methods, which are more suited for scripting languages. The entire project had grown tremendously after inclusion of many new GNU-licensed packages and extending the functionality of JAS in many areas, such as 3D graphics, serialized I/O and numerous numerical packages. At present, jHepWork covers an impressive list of Java-written packages ranged from basic mathematical functions to neural networks and cellular automation. And, eventually, a little of JAS has left inside jHepWork! One important thing, however, has remained: As JAS, jHepWork was still an open-source software that can be downloaded freely from the Web.

For this project, Python was chosen as the main programming language because it is elegant and easy to learn. It is a great language for teaching scientific computation. For developers, this is an ideal language for fast prototyping and debugging. However, since the whole project was written in Java, it is Jython (Python implemented in Java) that was eventually chosen for the jHepWork project.

This book is intended for general audience, for those who use computing power to make sense of surrounding us data. This book is a good source of knowledge on data analysis for students and professionals of all disciplines. Especially, this book is for scientists and engineers, and everyone who devoted themselves to the quest of where we find ourselves in the Universe and what we find ourselves made of.

This book is also for those who study financial market; I hope it will be useful for them because the methods discussed in this book are undoubtedly common to any scientific research. However, I have to admit that this book may have little interest for a commercial use since financial-market analysts, unlike researches in basic scientific fields, could afford costly commercial products.

This book is about how to understand experimental data, how to reduce complexity of data, derive some meaningful conclusions and, finally, how to present results using Java graphical packages. It concentrates on computational aspects of these topics: as you will see, due to the simplicity of Python, one could catch ideas of many examples of this book just by looking at the code snippets without even explaining them in words. This book is also about how to simulate more or less realistic data samples which can mimic real situations. Such simulated data are used in this book in order to give simple and intuitive examples of data analysis techniques using Java scripting.

In this book I did not go deep inside of particular statistical or physics topics, since the aim was to give concrete numeral receipts and examples using Jython scripting language interfaced with Java numerical packages. My aim was also to give an introduction to many data-analysis subjects with sample code snippets based on Jython and jHepWork Java libraries. In cases when I could not cover the subject in detail, a sufficient number of relevant references was given, so the reader can easily find necessary information for each chapter using external sources.

Thus, this book presents practical approaches for data analysis, focusing on programming techniques. Each chapter describes the conceptual and methodological underpinning for statistical tools and their implementation in Java, covering essentially all aspects of data analysis, from simple multidimensional arrays and histograms to clustering analysis, curve fitting, metadata and neural networks. This book includes a comprehensive coverage of various numerical and graphical packages currently implemented in Java that are part of the jHepWork project.

The book was written by the primary developer of the software, and aimed to present a reliable and complete source of reference which lays the foundation for future data-analysis applications using Java scripting. The book includes more than

200 code snippets which are directly runnable and used to produce all graphical plots given in the text. A detailed description and several real-life data-analysis examples which develop a genuine feeling for data analysis techniques and their programming implementation are given in the last chapter of this book.

Finally, I am almost convinced myself that this book is self-contained and does not depend on knowledge of any computing package, Java, Python or Jython (although knowledge of Python and Java is desirable for professionals).

Chicago-Hamburg-Minsk Sergei V. Chekanov

Springer

http://www.springer.com/978-1-84996-286-5

Scientific Data Analysis using Jython Scripting and Java
Chekanov, S.V.
2010, XXIV, 440 p., Hardcover
ISBN: 978-1-84996-286-5