MLSS 2015 Probabilistic Programming Practical Getting Started Guide

Tom Jin, Frank Wood, Brooks Paige

July 2015

1 Errata

This is essentially the same document as was distributed previously. The key differences are:

- We have discovered that default `lein` distributed by many Linux package managers is quite old. Run `lein version` to check that your `lein` version is greater than 2. If it isn't, follow the installation directions below.
- MLSS-specific exercise materials should be downloaded *prior* to the practical session directly from `https://bitbucket.org/probprog/mlss2015/get/master.zip`. `git` is no longer specifically required.
- All exercises can now be run and edited in a browser-based notebook, so installing a Clojure IDE is no longer required.
- Please contact us (`fwood@robots.ox.ac.uk`, `brooks@robots.ox.ac.uk`) *before* your practical if you aren't able to get everything installed. We're both physically in Tübingen already.

2 Introduction

The probabilistic programming practical will give you hands-on experience with probabilistic programming, particularly the Anglican programming language and system `http://www.robots.ox.ac.uk/~fwood/anglican`.

By the end of the practical you should be comfortable programming in Clojure and Anglican, and, more importantly, familiar with how to program in probabilistic programming languages in general.

The exercises and short-lecture introductions to the exercises will be constructed so as to help you understand how probabilistic programming works in general. And while the amount of time we will have together will be insufficient for you to either implement a probabilistic programming system or implement a custom inference algorithm in an existing system, you should be left with a very

clear impression of a) how one would go about implementing such a probabilistic programming language and b) where you would go and what it would take to start extending, for instance, Anglican with both new probabilistic primitives and new inference algorithms.

Prior to arriving at the practical, please be sure to do the following pre-practical preparation (note that while this document is long, the required preparation work should take, realistically, no more than a few minutes to complete):

3 Required Pre-Practical Preparation

Anglican is a language that compiles to Clojure that compiles to the JVM. For this reason you need the Java and Clojure ecosystems installed on either your own personal laptop or on a machine into which you can ssh, and, in the latter case, to which you can open socket (http) connections.

3.1 Java Prerequisites

Clojure depends on having a JVM installed of version ≥ 1.5 (the most recent available version is fine). Windows and Mac OS X users can download Java installers from `https://www.java.com/en/download/manual.jsp`. Linux users who do not already have Java installed can install from their package managers:

```
# Debian/Ubuntu
sudo apt-get install default-jre

# Fedora
sudo yum install java-1.7.0-openjdk
```

3.2 Install Leiningen

Leiningen is a self-installing automated Clojure project management system. You must install Leiningen from `http://leiningen.org/`. "lein" (short for Leiningen) is a self installing script as well as the primary means of invoking both Anglican and Clojure read eval print loops (REPL). Fortunately "lein" is trivial to install in *nix environments (see below). Windows users have it just as easy but should refer to the website. **Note that Leiningen version 2.x is required**; the version in your linux package repositories may be quite a bit out of date.

```
# Download lien to ~/bin
mkdir ~/bin
cd ~/bin
wget http://git.io/XyijMQ

# Make executable
chmod a+x ~/bin/lein
```

```
# Add ~/bin to path
# Note: Mac OS X users should replace ".bashrc" with ".profile"
echo  'export PATH="$HOME/bin:$PATH"' >> ~/.bashrc

# Run lein
lein
```

3.3 Download the exercises

The exercises themselves can be downloaded from the following url:

`https://bitbucket.org/probprog/mlss2015/get/master.zip`

Download and unzip this file — the resulting directory contains everything you need for the practical.

When you have managed to do all this successfully then, in effect, you will have a Leiningen project sitting locally on your machine. Within it you should try to start a web-based Anglican REPL: [1]

```
# replace "mlss2015" with the unzipped directory containing
# the exercises
cd mlss2015
lein gorilla :port 8990
```

which will start a web service on port 8990 which allows you to view, edit, and run the different exercises. Open up a web browser, and visit `http://localhost:8990/worksheet.html`. Using the menu on the top right you should be able to open and interactively run the exercises, using the "Load a worksheet" command.

Start by loading `beta-flip/hello-world-worksheet.clj`. You can step through (and execute) the cells in the worksheet by pressing `shift+enter`.

The first exercise is `gaussian/gaussian-worksheet.clj`.

4 Optional Pre-Practical Preparation

4.1 Clojure Programming

Ideally you would be a Clojure programmer, or, at least, a functional programming expert in advance of the practical. Familiarizing yourself with Clojure can only help the overall experience. The Clojure main website `http://clojure.org/` has links to a large number of language learning resources, in particular `http://clojure-doc.org/articles/tutorials/introduction.html`.

[1] Users installing on a server will instead run `lein gorilla :ip 0.0.0.0` .

4.2 Probabilistic Programming Reading

There are a number of probabilistic programming papers and online resources which you may wish to read in advance of the practical. For the purposes of this practical we specifically recommend reading an introduction to probabilistic programming and the Anglican language [4] (this is an arXiv update to [3] that includes source code in an updated and, as of now, current syntax), a paper on how to implement inference in a probabilistic programming language, in this case probabilistic-C [2], and a paper about how about higher order probabilistic programming languages can be used to code expressive, particularly Bayesian nonparametric, models [1].

Also there are a number of online resources for Anglican in particular and probabilistic programming in general.

4.2.1 Online Resources

- `http://www.robots.ox.ac.uk/~fwood/anglican/`
- `http://dippl.org/`
- `http://probabilistic-programming.org/wiki/Home`

References

[1] Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: A language for generative models. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI2008)*, pages 220–229, 2008.

[2] Brooks Paige and Frank Wood. A compilation target for probabilistic programming languages. In *JMLR; ICML 2014*, pages 1935–1943, 2014.

[3] F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, pages 1024–1032, 2014.

[4] F. Wood, J. W. van de Meent, and V. Mansinghka. A new approach to probabilistic programming inference. *arXiv preprint arXiv:1507.00996*, 2015.