Hakaru Language: Standard Library Implementation and Language Validation Testing

Justin Staples, Mahmoud Khattab, Nevin Mahilal and Aryan Sohrabi, supervised by Dr. Christopher Anand & Dr. Jacques Carette

{staplejw, khattm, mahilank, sohraa3, anandc, carette}@mcmaster.ca

Department of Computing and Software, McMaster University

Introduction

Hakaru is an experimental **probabilistic programming language**, which simplifies implementing models of statistical distributions.

- Niche application, small language.
- Models output a stream of numbers distributed according to the implemented distribution.
- Models can be compiled to C and Haskell for use in larger applications.

`hk-maple` is an inference algorithm suite that uses Maple to perform algebraic transformations on Hakaru programs to produce equivalent models with greater sampling efficiency.

Motivation

Increase language accessibility:

- Standard Library Development: focus on univariate distributions.
- Added in primitive math functions that were incomplete (log(x)) or missing (choose(n,k)).
- Syntax-highlighting-for-hakaru package for Sublime Text (Figures 2 and 3).

Test language validity:

- Use known relationships between distributions to test the validity of program transformations like `hk-maple`.

Key Concepts

- ⋆ *PDF: Probability Density.*
- ⋆ *UDR chart: Univariate Distribution Relationship chart.*
- ⋆ '$\sim$' vs '$<\sim$':
  - – $X \sim Normal(0,1)$: random variable, X, is distributed according to a normal distribution (statistics literature).
  - – `x <~ normal(0, 1)`: pull a random sample from a normal distribution and bind it x (Hakaru code).

Standard Library Development

The Standard Library implements commonly used univariate distributions. We have followed the UDR to implement distributions with the following guiding principles:

- When possible, implement distributions as transformations on pre-existing models.
- In case of many possible implementations, take the shortest path from a primitive distribution on the UDR.

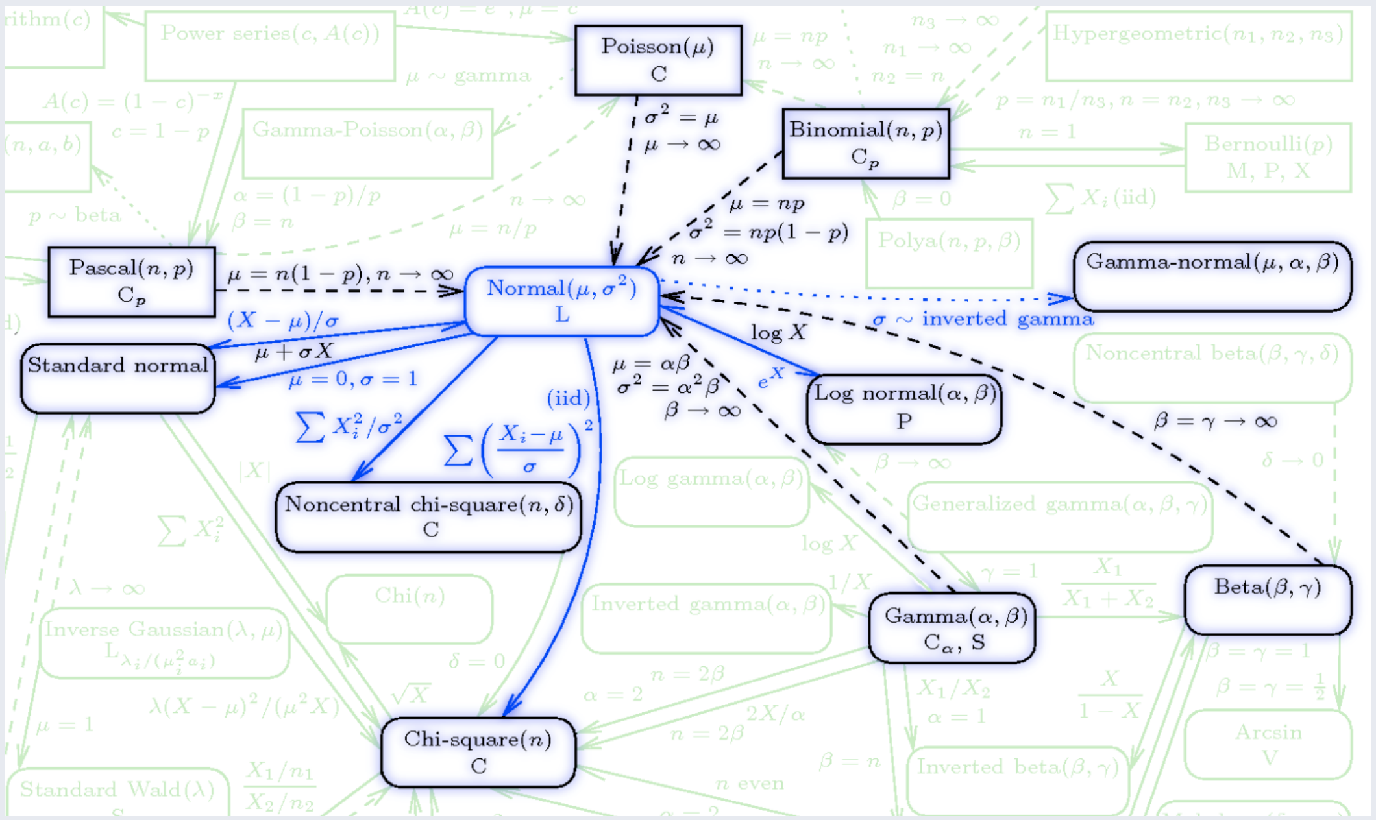
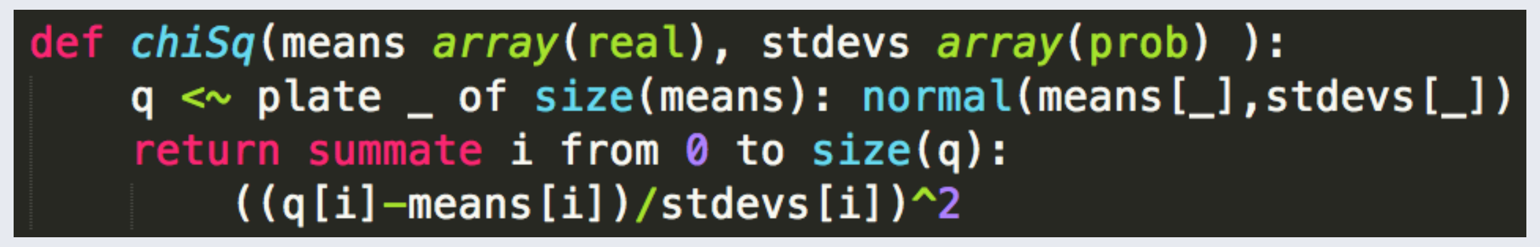


Figure 1: A snapshot of the UDR [1] shows how the normal distribution can be transformed into a multitude of other distributions.

Hakaru does not allow us to transform models directly. Must apply transformations to samples pulled from the model using the bind <~ operator. Therefore, we are interested in implementing transformations of the form:

$$\boxed{R(p,q) \Rightarrow X \sim A(p) \Rightarrow f(X) \sim B(q)}$$

We can extend this definition to include transformations defined in terms of an aggregation of many independent samples. For example, the standard chi-square distribution is defined as the sum of the squares of n normal random variables (see Figure 2).

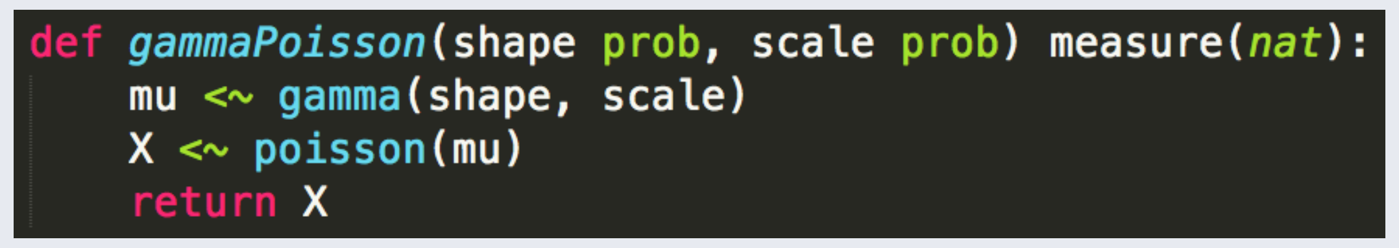


```
def chiSq(means array(real), stdevs array(prob) ):
    q <~ plate _ of size(means): normal(means[_],stdevs[_])
    return summate i from 0 to size(q):
        ((q[i]-means[i])/stdevs[i])^2
```

Figure 2: Our implementation of the chi-square distribution.

Hakaru also lends itself well to Bayesian transformations, which take the following form. The gamma-poisson distribution can be described by such a transformation (see Figure 3).

$$\boxed{X \sim A(p) \Rightarrow Y \sim B(q,X) = C(p,q)}$$

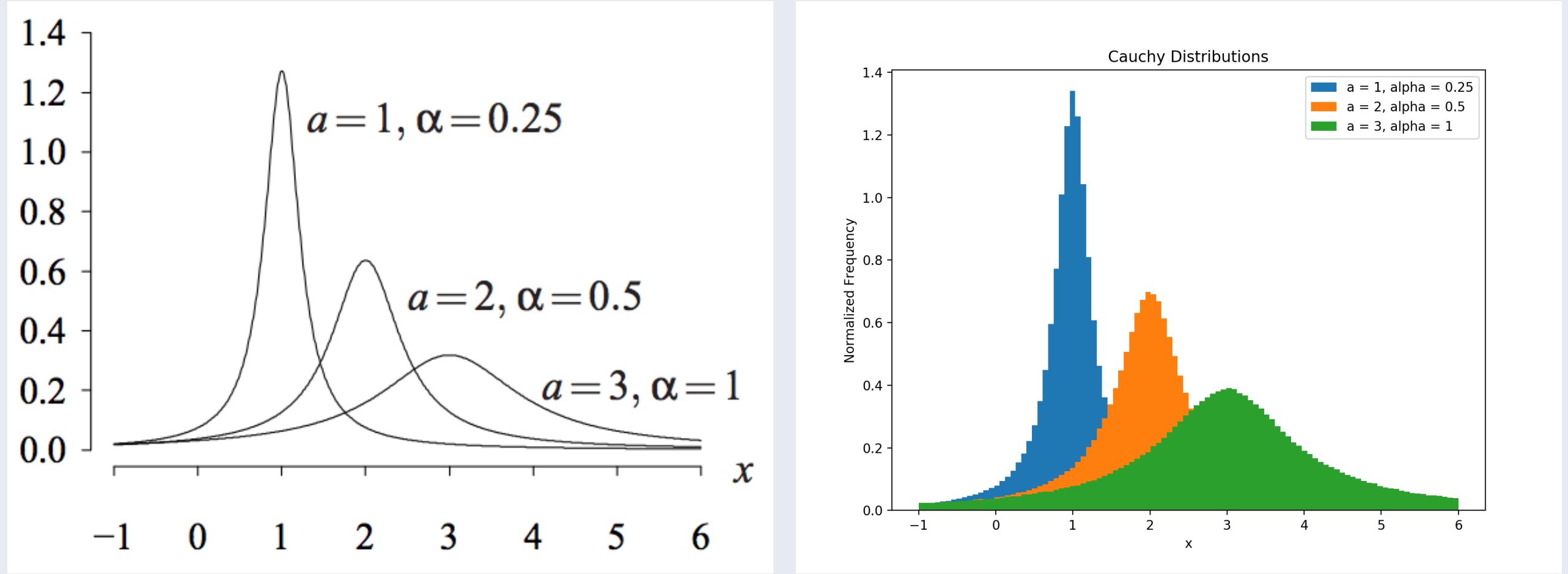


```
def gammaPoisson(shape prob, scale prob) measure(nat):
    mu <~ gamma(shape, scale)
    X <~ poisson(mu)
    return X
```

Figure 3: Our implementation of the gamma-poisson distribution.

In the case of unreachable distributions, we have implemented models in terms of their PDF/PMF.

Plotting a histogram of a sample population drawn from a model and comparing its shape to that of its PDF, we can have some confidence that an implementation is correct.



(a) (b)

Figure 4: A few plots of the Cauchy distribution are shown [1], along with a few histograms of data that have been sampled from a Hakaru program.

Testing Relationships Between Distributions

Hakaru's validity can be tested by checking if it recognizes known relationships between distributions. More specifically:

> *Hypothesis: By applying appropriate transformations to implementations of distributions, A and B, we can create two distinct Hakaru programs which, when passed to hk-maple, will reduce to equivalent Hakaru code.*

Test cases are written in Hakaru, therefore we test relationships of the forms discussed. There are hundreds of possible tests. We have focused on a small set of distributions.

Tests that pass help prove that Hakaru is valid. Tests that fail either indicate a language bug, or are the consequence of an intended design choice. These results give the language developers useful information for future work on the language.

Conclusions & Future Work

The project was successful in increasing language accessibility, both in terms of Standard Library Development and filled in language features. Testing has proven there is still a lot of work to be done. Here are some suggestions for future work:

- Languages Features: Hakaru needs import and error/exception handling.
- Stdlib Dev: multivariate distributions
- Testing: hundreds of known relationships untested

References

[1] L. Leemis, "Univariate Distribution Relationship Chart", Math.wm.edu, 2018. [Online]. Available: http://www.math.wm.edu/ leemis/chart/UDR/UDR.html.
[2] P. Narayanan, J. Carette, W. Romano, C. Shan and R. Zinkov, Probabilistic Inference by Program Transformation in Hakaru (System Defootnoteion), Functional and Logic Programming, pp. 62-79, 2016.