Z notes

January 26, 2022

Turnpike Reconstruction

The Turnpike reconstruction problem is to reconstruct a point set from the distances. This finds applications in physics and molecular biology (see the references for pointers to more specific information). The name derives from the analogy of points to turnpike exits on east-coast highways. Just as factoring seems harder than multiplication, the reconstruction problem seems harder than the construction problem. Nobody has been able to give an algorithm which is guaranteed to work in polynomial time.[1]

Implementation in Julia, C++, and Python.

[1] [4] Chapter 4. › jones-2004-introd-bioin-algor

Galton-Watson process

The Galton-Watson process is part of branching processes. Let X be a random variable (r.v.) with values in $\mathbb{N}$, and for $k \in \mathbb{N}$, $p_k = \Pr(X = k)$, and $m = \mathbb{E}(X) = \sum_{k=0}^{\infty} k p_k < \infty$. Given $(X_{i,j})_{i,j \in \mathbb{N}}$ a family of r.v. with law $\mathbb{P}_X$, we define the sequence $(Z_n)_{n \in \mathbb{N}}$ as follows:

$$\begin{cases} Z_0 = 1 \\ \forall n \in \mathbb{N},\ Z_{n+1} = \sum_{i=1}^{Z_n} X_{i,n} \end{cases}$$

We also define $\pi_n := \Pr(Z_n = 0)$, and $\mathbb{P}_{\text{ext}} := \Pr(\exists n \in \mathbb{N}, Z_n = 0)$.

The above model allows to consider a set of particles (from the same family) which are able to generate other particles. Each particle has a probability p_k of generating k independent particles, and this probability is fixed across generations. The children of the nth generation belong to the $n + 1$th generation. The quantity Z_n represents the number of particles at generation n. Each particle i at generation n has $X_{i,n}$ children ($1 \leq i \leq Z_n$), π_n is the probability of extinction at generation n, and $\mathbb{P}_{\text{ext}}$ is the probability of extinction of the population. If there exists n such that $Z_n = 0$, then the population dies. In other words, the hypotheses of this model are:

- If $p_0 = 0$, then $\forall n \in \mathbb{N}*$, $Z_n \geq 1$ a.s. and $\mathbb{P}_{\text{ext}} = 0$.
- If $p_0 = 1$, then $\forall n \in \mathbb{N}*$, $Z_n = 0$ a.s. and $\mathbb{P}_{\text{ext}} = 1$.

Streams in Scheme

A nice analogy for streams, or infinite sequences, is the time-varying behavior of a quantity x as a function of time x(t): If time is discrete,

then a time function is nothing but a (possibly infinite) sequence – the function itself doesn't change over time. Streams can be implemented as *delayed lists*.[2]

[2] [1] §3.5. › abelson-1996-struc-inter

> Streams are a clever idea that allows one to use sequence manipulations without incurring the costs of manipulating sequences as lists. With streams we can achieve the best of both worlds: We can formulate programs elegantly as sequence manipulations, while attaining the efficiency of incremental computation. The basic idea is to arrange to construct a stream only partially, and to pass the partial construction to the program that consumes the stream. If the consumer attempts to access a part of the stream that has not yet been constructed, the stream will automatically construct just enough more of itself to produce the required part, thus preserving the illusion that the entire stream exists.

Programming languages for educational purpose

> Marvin Minsky and Seymour Papert formed many of our attitudes about programming and its place in our intellectual lives. To them we owe the understanding that computation provides a means of expression for exploring ideas that would otherwise be too complex to deal with precisely. They emphasize that a student's ability to write and modify programs provides a powerful medium in which exploring becomes a natural activity.[3]

[3] SICP, Acknowledgments, Harold Abelson and Gerald Jay Sussman with Julie Sussman

Epidemiology and outbreak analysis

> We describe two forms of bias that may affect the estimation of the overall CFR—preferential ascertainment of severe cases and bias from reporting delays—and review solutions that have been proposed and implemented in past epidemics. Also of interest is the estimation of the causal impact of specific interventions (e.g., hospitalization, or hospitalization at a particular hospital) on survival, which can be estimated as a relative CFR for two or more groups. When observational data are used for this purpose, three more sources of bias may arise: confounding, survivorship bias, and selection due to preferential inclusion in surveillance datasets of those who are hospitalized and/or die.[4]

[4] Potential Biases in Estimating Absolute and Relative Case-Fatality Risks during Outbreaks, [5] › lipsitch-2015-poten-biases

On the use of splines for the Titanic dataset

A modern take on regression modeling using the beautiful `rms` package. Fifteen years of use, and counting. Besides, the following illustrates the use of `rcs` to account for possible non-linearities, in this case regarding age of passengers.[5]

[5] [3] › harrell-2015-regres-model-strat

```
library(rms)
dd <- datadist(titanic3)
options(datadist = "dd")
```

```
f <- lrm(survived ~ rcs(sqrt(age), 5) * sex, data = titanic3)
print(f)  # or latex(f)
## Wald statistics
a <- anova(f)
print(a)
plot(a)
## Odds-ratio & Co.
s <- summary(f, age = c(2, 21))
plot(s, log = TRUE)
## Effect table
print(s, dec = 2)
## Nomogram
plot(nomogram(f, fun = plogis, funlabel = "Prob(survived)"))
## Plotly
plotp(Predict(f, age, sex))
```

IEEE-754 and NA values

First off, floating-point numbers are not real numbers, and floating-point arithmetic does not satisfy the axioms of real arithmetic. Trichotomy is not the only property of real arithmetic that does not hold for floats, nor even the most important. For example:[6]

[6] Stack Overflow

- Addition is not associative.
- The distributive law does not hold.
- There are floating-point numbers without inverses.

I could go on. It is not possible to specify a fixed-size arithmetic type that satisfies all of the properties of real arithmetic that we know and love. The 754 committee has to decide to bend or break some of them. This is guided by some pretty simple principles:

1. When we can, we match the behavior of real arithmetic.
2. When we can't, we try to make the violations as predictable and as easy to diagnose as possible.

Design and font size

Use different font sizes to create hierarchy. To create a scale of font sizes, take the base font size (say 16) and mutliply it by a fixed ratio to get more values:[7]

[7] [6] › wasserman-2004-all-statis

```
16 base    = 16    4
16 * 8/9   = 14    3
14 * 8/9   = 12.5  2
12.5 * 8/9 = 11    1
```

This works in both direction:

```
16 / 8/9   = 18    5
18 / 8/9   = 20    6
```

Popular ratios: 15/16 (minor second), 8/9 (major second), ($[1/\sqrt{5}]/2$) golden ratio, 5/6 (minor third), 4/5 (major third), 3/4 (perfect fourth).

Confidence intervals and probability statements

Confidence intervals are not probability statements about θ.

> Consider this example from Berger and Wolpert (1984). Let θ be a fixed, known real number and let X_1, X_2 be independent random variables such that $\Pr(X_i = 1) = \Pr(X_i = -1) = 1/2$. Now define $Y_i = e + X_i$, and suppose that you only observe Y_1 and Y_2.

Mean vs. median relative efficiency

Speaking of OLS and squared error, it is better to consider the mean than the median since it is more than half again more efficient. Indeed, it can be shown that:[8, 9]

$$\mathbb{E}\{(\check{x} - \mu)^2\}/\mathbb{E}\{(\bar{x} - \mu)^2\} = 1.57.$$

The relative efficiency (RE) is defined as the variance of the first estimator divided by that of the second estimator. Since the variance of the sample mean is σ^2/n, and that of the sample median $\pi\sigma^2/2n$, we have $\text{RE} = (\pi\sigma^2/2n)/(\sigma^2/n) = \pi/2 = 1.57$.

If, however, we are interested in prediction, the mean still wins but only by 2%:

$$\mathbb{E}\{(X - \check{x})^2\}/\mathbb{E}\{(X - \bar{x}\} = 1.02,$$

since most of the prediction error comes from the variability of X.

Division is an iterative algorithm (involving to shift the result from quotient to remainder using a Euclidean measure) while addition can be performed as a succession of bit manipulation tricks. It would be interesting to check whether Newton-Raphson is still used internally for reciprocating a number.

[8] Why is division so much more complex than other arithmetic operations?

[9] [2] › efron-2020-predic-estim-attrib

Slurping and barfing

Slurp and barf refers to part of structural editing whereby we pull something into an s-expression, or conversely push out something from the s-expression. These operations can be carried out in a

rightward or leftward manner. They are of course available in Emacs under the paredit package. Vim-sexp provides similar functionalities for Vi(m).[10]

[10] Riastradh on IRC #scheme (via Matrix)

```
(a b (c d) e)
(a (b c d) e) ;; slurp left
(a (b c) d e) ;; barf right
```

Scheme and equality testing

`eq?` compares object identity, which is inappropriate to compare floats. Indeed, many Schemes will store floating-point values on the heap, and represent them by a pointer into the storage. Two different floating-point operations may result in different heap allocations, but `eq?` may compare the pointers rather than the values they point to. On the other hand, `eqv?` is used to distinguish numbers by their exactness flag, while `=` is appropriate when we want to distinguish numbers by their value only, irrespective of exactness flag. For instance, `(eq? +inf.0 (abs (exp 1000.0)))` will return `#f`.

```
(define infinite?
  (and (number? obj) (= +inf.0 (abs obj))))
```

Generating Uniform random floats

The naive approach of generating an integer in $\{0, 1, \ldots, 2^k - 1\}$ for some k and dividing by 2^k, as used by, e.g., `Boost.Random` and GCC 4.8's implementation of C++11 $\text{uniform}_{\text{real distribution}}$, gives a nonuniform distribution:[11]

[11] Taylor R Campbell, `http://mumble.net/~campbell/tmp/random_real.c`

- If k is 64, a natural choice for the source of bits, then because the set $\{0, 1, \ldots, 2^{53} - 1\}$ is represented exactly, whereas many subsets of $\{2^{53}, 2^{53} + 1, \ldots, 2^{64} - 1\}$ are rounded to common floating-point numbers, outputs in $[0, 2^{-11})$ are underrepresented.
- If k is 53, in an attempt to avoid nonuniformity due to rounding, then numbers in $(0, 2^{-53})$ and 1 will never be output. These outputs have very small, but nonnegligible, probability: the Bitcoin network today randomly guesses solutions every ten minutes to problems for which random guessing has much, much lower probability of success, closer to 2^{-64}.

  See also: LLVM #18767 and LWG #2524.

On BSD and MIT licenses

Actually there are four BSD licenses:

1. zero-clause BSD: Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted.

2. two-clause BSD:

   > Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
   >
   > Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. three-clause BSD: two-clause BSD + Neither the name of the [organization] nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

4. four-clause BSD: three-clause BSD + All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the [organization].

On the contrary, the MIT license does not include claure reagrding advertisment like in the BSD 3 license, but it features an attribution clause:

> Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
>
> The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

The closest licenses are probably BSD 2 and MIT, although the MIT license further allows to merge, publish, sublicense or even sell existing work.

SRFI 179: Nonempty Intervals and Generalized Arrays

This SRFI extends Bawden-style arrays by adding storage-class object, array currying and permuting, as well as an interval and a mapping that has that interval as its domain, which allows for generalized and specialized arrays.[12]

[12] SRFI 179

This approach differs from SRFI-164 where arrays are seen as extension of generalized vectors (with additional support for Kawa ranges).

> The most peculiar thing to me about this discussion of arrays for Scheme is the seemingly widespread notion that arrays should be somehow identified with vectors that contain vectors. (One common corollary of this belief seems to be that if you don't have native arrays, you -have- to use vectors of vectors – which is inefficient.)
>
> Well, let me point out that the difference between a vector and an array is really just some arithmetic performed on the indices. If I'm writing a program that does a lot of manipulating of 4x4 matrices, I can do as well as any native array implementation by using 16 element vectors, and writing
>
> ```
> (vector-ref M (+ (* 4 i) j))
> ```
>
> instead of
>
> ```
> (array-ref M i j)
> ```
>
> If I write my loops so that, in fact, I process the elements of the matrix in the order that they are layed out in the vector, then a good compiler should be able to do just as well is it could if I used native arrays.
>
> — Alan Bawden

QIIME-2 for microbiome analysis

Although it is oriented toward the analysis of microbiomes, it might still be relevant for fungi metabarcoding. The workflow and associated tools are almost identical: demultiplexing, read pairs merging, primer removal (`cutadapt`), sequence clustering (`vsearch`), OTU clustering (apparently, it does rely on swarm), taxonoxmy assignment. The last step is performed using either a reference database or an ML classifier (scikit-learn, which apparently is a Naive Bayes classifier; see Optimizing taxonomic classification of marker-gene amplicon sequences with QIIME 2's q2-feature-classifier plugin. Microbiome 6: 90 (2018)).[13]

[13] `https://qiime2.org`

See also the SILVA project (no fungi of interest, though).

References

[1] Harold Abelson and Gerald Jay Sussman. *Structure and Interpretation of Computer Programs*. 2nd ed. MIT Press, 1996.

[2] Bradley Efron. "Prediction, Estimation, and Attribution". In: *Journal of the American Statistical Association* 115.530 (2020), pp. 636–655.

[3] Frank E Harrell. *Regression Modeling Strategies*. 2nd ed. Springer Series in Statistics. Cham: Springer International Publishing, 2015.

[4] Neil C. Jones and Pavel A. Pevzner. *An Introduction to Bioinformatics Algorithms*. The MIT Press, 2004.

[5] Marc Lipsitch et al. "Potential Biases in Estimating Absolute and Relative Case-Fatality Risks during Outbreaks". In: *PLOS Neglected Tropical Diseases* 9.7, e0003846 (2015), pp. 1–16.

[6] Larry Wasserman. *All of Statistics*. Springer, 2004.