CS 482

Aiden Sirotkine

Spring 2025

Contents

Chapter 1

CS 482

1.1 EXAM 2

Using common and antithetic variance to calcualte covariance.

I went to office hours and was told

- If you know everything in slides 2 you should be good
- If you're told to make a confidence interval, you're probably going to be given the values. You won't be expected to do a whole bunch of stupid math

Chapter 2

PRNG

Pseudo Random Number Generators

They make pretty random U(0, 1) numbers without any sort of shenanigans

2.0.1 Desirable Properties

- Independent
- Uniform (E = 1/2, Var = 1/12)
- Reproducable
- Long Period
- Computationally convenient

You can use tables or actual simulation data or a couple algorithms

2.1 Linear Congruential Generators

These use modulus math to make pseudo random numbers

$$V_i = (a * V_{i-1} + c) \mod (m)$$

divide by m to get a U(0, 1)

2.1.1 Full Period

a full period LCG goes over all values 0 to m-1 before cycling
I'll remember it better as
CRITERIA

- If q (prime or 4) divides m, then q divides a-1
- m and c are relatively prime

2.1.2 Multiplicative Generator

Literally just set $c = 0$ and that's what it is.
Also called a power residue
Power residues CANNOT have a full period

2.1.3 E and Var

1/2 and 1/12 for full period LCG's

2.2 Uniformity Tests

2.2.1 Chi-Squared

2.2.2 Kolmogorov-Smirnov

order all of your datapoints and map them to the cumulative distribution (CDF) of whatever you think it is.

Check the max upwards and downwards deviations to see if it actually matches the CDF.

There's a table for the critical values.

the Andersen-Darling test is similar but you take the mean squared difference instead of looking at the max differences in both directions.

2.3 Independence Tests

2.3.1 Sign Test

Looking at runs of numbers above or below the median
should be normally distributed

$$\mu = 1 + \frac{N}{2} \qquad \sigma^2 = \frac{N}{2}$$

2.3.2 Runs Test

Number of runs of increasing or decreasing numbers.

Normally distributed but with some wack mean and variance

$$\mu = \frac{2N-1}{3} \qquad \sigma^2 = \frac{16N-29}{90}$$

for $N > 30$ you can use a Z test instead of a t-test

Chapter 3

Driving a Simulation

3.1 Trace

Use actual data from whatever system you're simulating

Easy to model and trustworthy

however

not able to be generalized and very expensive

3.2 Empirical Distributions (Histograms)

Pretty similar to the trace argument, plus easy to replicate

3.3 Parametric Distributions

Fit a probability distribution to the data

Results are generalized and replicable, but may not be representative of the real system

3.4 Input distributions

make a list of reasonable ones (exponential, normal, poisson, uniform, whatever)

Then test what's best with a couple methods

- look at mean and variance
- chi-squared or Kolmogorov-Smirnov test
- maximum likelihood estimator
- expert opinion

Chapter 4

Non-Uniform RNG

More often than not your probability distribution will not be a straight line

however

you can make all non-uniform distributions with uniform random numbers

4.0.1 Comparing Algorithms

Does the seed work well

is it efficient and stable

4.0.2 U(a, b)

Just shift the min and max

$$x(b - a) + a$$

That's it its not hard

4.1 Inversion

This is the best way to do it because it gives you just like a straight formula. If you can do it at least

if the CDF is F(X) = Y, then find $F^{-1}(Y)$ and Y is a $U(0, 1)$

This works because of the nature of CDF's and the fact that they need to go from 0 to 1 because that is the min and max of a probability

4.1.1 Exponential

The PDF is given by

$$f(x) = \frac{1}{\mu} e^{-x/\mu}$$

So the CDF is given by

$$F(X) = \int_{-\infty}^{X} f(x)dx = 1 - e^{-x/\mu}$$

So our inversion is given by

$$1 - e^{-x/\mu} = U \Rightarrow\Rightarrow\Rightarrow x = -\mu \ln(1 - U)$$

Where U is a $U(0, 1)$ variable and x is your exponential random variate

4.1.2 Triangular Distribution

It's obnoxious but I feel confident in myself to BS something good

4.2 Bernoulli Variables

It's just related to Bernoulli trials which have success rate p and fail rate $(1-p)$

4.2.1 Geometric Random Variable

Number of Bernoulli trials until first success

$$PDF = p(1-p)^{j-1} \qquad CDF = \sum_{j \leq k} p(1-p)^{j-1} = 1 - (1-p)^k = U$$

Then you can invert that and then k given U

4.3 Pros and Cons of Inversion

You can't always use inversion depending on the CDF
But

- Handle truncated distributions easily
- You can use variance reduction
- it only uses a single U(0, 1)

- order statistics are easy

  speaking of

4.4 Truncation

You just make a smaller part of a CDF go from 0 to 1 and it works

4.5 Order Statistics

Consider n ordered data points $x_1, x_2, \ldots, x_n$

x_1 is the failure time in a serial system

x_n is the failure time in a parallel system

Basically is a bunch of machines are running simultaneously and each have some failure time

If the machines are sequential, the first failure time is the system failure

If they're parallel, the last failure time is the system failure.

4.5.1 Parallel System Failure

Given the individual component failure CDF, you get

$$F_n(a) = F(a)^n = u \to a = F^{-1}(u^{1/n})$$

The equation is just the chance of every single component failing

This kind of makes sense because u is going to be very small for the whole parallel system to fail because every machine needs to fail first.

4.5.2 Sequential System Failure

Given the failure chance of an individual component

$$F_1(a) = 1 - (1 - F(a))^n = u \to a = F^{-1}(1 - (1 - u)^n)$$

The equation is essentially the inverse of every component succeeding

The inside thing is going to be a function close to 1 for large n which makes sense because only a single component needs to fail for the whole system to go under

4.6 Special Properties

4.6.1 Erlang(r, μ)

It's the sum of r IID exponentials

4.6.2 Binomial(n, p)

It's connected to Bernoulli trials

The probability distribution is given by

$$f(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

You can do this with a legit Bernoulli algorithm
for n trials, make U(0, 1) and if $U < p$ increment X.
X will then be distributed Binomial(n, p)

4.6.3 Geometric

We talked about this one, but you can also use a legit Bernoulli approach

while forever, generate a random variable, and increment X for every failure. When there's a success, return X.

4.7 Acceptance-Rejection

Generate an x, generate a y, see if its in your PDF. That's it. If you repeat enough forever then like you're balling.

You can optimize it in a couple ways but like that's the gist.

4.8 Poisson(λ) Random Variable

The number of IID Exponentials under some value

$$P(X = k) = e^{-\lambda}\frac{\lambda^k}{k!}$$

And the expected value/mean of the variable is given by

$$E(X) = \lambda$$

You can do some math to get a pretty good algorithm for a Poisson using just U(0, 1) variables

Basically just counted U(0, 1) variables until you get one that is less than $e^{-\lambda}$

4.8.1 Poisson Processes

It's kind of the same as Bernoulli trials except instead of a U(0, 1) is exponential

Times at which events occur, where the interarrival times of the events are distributed exponential.

$$t_i = t_{i-1} - \frac{1}{\lambda} \ln(U_{i-1})$$

4.8.2 Inhomogeneous Poisson Process

The only difference in λ changes over time

Chapter 5

Random Discrete Variables

Consider a random permutation?

There's an efficient way to do it or you can just randomly select each location of each item 1 by 1

5.1 Alias Method

This one is actually pretty smart

put mini binary probabilities in each of your already existing uniform probabilities so that you only need 1 IID variables to get a non-uniform discrete variable :

5.1.1 Algorithm

for all of your probabilities, multiply each of them by the total number of numbers you're working with.

let H be the probabilities that are smaller than the uniform distribution let G be the probabilities that are larger than the uniform distribution

for any non-empty bucket H, just dump a probability from G into there.

$$Q_G = Q_G + Q_H - 1$$

if $Q_G < 1$, remove it from G and add it to H

You should only have 1 alias per bucket.

5.2 Marsaglia's Method

Only works for discrete random variables with probabilities that have denominators as powers of 2

Only use when you want to generate a bunch of values and you have only 1 U(0, 1) variable

5.2.1 Urns

YOU USE BINARY TO FIGURE OUT HOW MANY TIMES TO MULTIPLY URN

that's basically it

But what you do with you 1 U(0, 1) is it puts you in the urn and then it tells you what part of the urn you land in by dividing it by the number of things in the urn

You find the urn and then generate a uniform discrete random variable

5.3 Normal(0, 1) Random Variable

The easiest way to do it is just

$N = \sigma X + \mu$

and that works like well enough

5.4 Verification

LAW OF TOTAL PROBABILITY
Basically check if the sum of all the probabilities is 1

5.4.1 Crude CLT Method

Its a sneaky math trick

$$Y = \sum_n U_i \qquad \frac{Y - \frac{n}{2}}{(n/12)^{1/2}} \rightarrow N(0,1) : n \rightarrow \infty$$

like sure man go for it

5.4.2 Bivariate

You do like a cosine thing with 2 U(0, 1) variables
Just make sure they arent back to back from the same LCG

5.4.3 Polar Method

It's some log shenanigans also with 2 U(0, 1) variables

Chapter 6

Output Stuff

i DO know

6.1 Variance and Covariance

$$Cov(X, Y) = E(XY) - E(X)E(Y) = \frac{(X - \bar{X})(Y - \bar{Y})}{n}$$

$$Var(X + Y) = Var(X) + Var(Y) + 2Cov(X, Y)$$
$$Var(X - Y) = Var(X) + Var(Y) - 2Cov(X, Y)$$