11/26/21

Finished poster for SUMS conference

Changed layout away from three columns to more of a free-flowing format

Color-coded sections

11/25/21

Redid block cipher diagram to also include encryption diagram to save space

Found an article that shows that SHA 1-3 have different levels of randomness

[Randomness Analyses of the Secure Hash Algorithms, SHA-1, SHA-2 and Modified SHA](https://ieeexplore.ieee.org/abstract/document/8991643)

11/24/21

Trying to find an article that differentiates levels of randomness for cryptographic algorithms

Working on creating a poster for the SUMS conference

1. What is a block cipher?
2. A brief introduction of NIST and Dieharder
3. Mode dependence definition/explanation
4. Results for NIST and Dieharder that show mode dependence
5. Introduction to CrytoStat and how it works
6. Results of CrytoStat and interpretation of the results
7. Suggestions on improving CrypoStat if needed

10/1/21

Read Overview of Randomness Test on Cryptographic Algorithms which is a really useful summary on the field. The article mentioned a study by Chen Hua which includes information on testing hash functions which I plan on reading soon.

“The analysis shows that at present, the randomness detection mainly focuses on the randomness evaluation of the algorithm output sequence, and the algorithm itself has an impact on the random number of the output sequence. At present, there is no relevant technology, standards or toolkit to analyze it. Therefore, the next step of this paper is to analyze the impact of the encryption algorithm's own implementation on the random number of the cipher algorithm's output sequence, and design related analysis and evaluation standards and tools for experiments” (Mengdi 2021).

Chen Hua. Security Test on Cryptographic Algorithms and Design of Key Cryptographic Components. Graduate School of the Chinese Academy of Sciences, 2004.

9/13/21

Changed purple handwriting to text to appear more professional

articles that discuss randomness of hash functions

[Overview of Randomness Test on Cryptographic Algorithms](https://iopscience.iop.org/article/10.1088/1742-6596/1861/1/012009/pdf)

[Randomness Properties of Cryptographic Hash Functions](https://cpb-us-w2.wpmucdn.com/people.smu.edu/dist/c/1257/files/2019/02/ts-slides-11qf8e7.pdf)

[(PDF) Cryptographic Randomness Testing of Block Ciphers and Hash Functions.](https://www.researchgate.net/publication/220335414_Cryptographic_Randomness_Testing_of_Block_Ciphers_and_Hash_Functions)

9/10/21

Professor Chen says that I can arbitrarily increase the number of rounds in order to increase precision

She also agrees that partial randomness theoretically exists

Integrated more slides from professor Chen into mine

Converted slideshow to PowerPoint in order to use equations

9/8/21

Worked on research proposal

Talked to Mr. Estes (cybersecurity teacher) about my research project

9/3/21

Wrote introduction for research proposal

Not really sure about how to write the literature review, I should probably talk to Ms. Klus

Can I use the increasing randomness of SHA functions to demonstrate that CyrptoStat can detect different levels of randomness?

How is the randomness of hash functions typically tested?

[Randomness testing of non-cryptographic hash functions for real-time hash table based storage and look-up of URLs](https://www.sciencedirect.com/science/article/abs/pii/S1084804513002853)

<https://vcc-lof-primo.hosted.exlibrisgroup.com/permalink/f/1t1hca3/TN_cdi_ieee_primary_8991643>

9/1/21

increased resolution of LaTeX images

emailed Professor Chen about problems I’m encountering, increasing sample size and my discoveries, hash algorithms as additional data points, increasing rounds, and poster layout

8/29/21

created LaTeX images for equations in presentation

8/27/21

create equations in word and import them?

trying to find a way to calculate 1,000,000 factorial

convert from presentation to poster format

8/22/21

[CURAND - Device API and seed](https://stackoverflow.com/questions/19737812/curand-device-api-and-seed)

[Generate Random Number Within Cuda Kernel](https://stackoverflow.com/questions/36274112/generate-random-number-within-cuda-kernel)

So it seems like this doesn’t work because even though it is done on the device part of it steel needs to be done globally which I can’t do within the confines of pj2

<https://www.cs.rit.edu/~ark/pj2/doc/edu/rit/gpu/package-summary.html>

<https://www.cs.rit.edu/~ark/runningpj2.shtml>

kernel: controls the interaction between hardware and software

unpackBigEndian/biUnpackBigEndian - BigInt → int[]/uint32\_t[]

biElem3D - uint32\_t\* → BigInt

Ran CryptoStat on other variations of AES and SHA1 SHA2 SHA3 and SHAKE

The number of rounds used in CryptoStat is rather low which reduces the precision of the tool. I don’t know what the rounds are so I don’t know if they can be arbitrarily changed, but increasing the number of rounds has the potential to easily increase its precision.

8/21/21

So I tried a bunch of different methods for generating random sequences in CUDA, but I kept running into the same error which said that I couldn’t do something host on a device or something like that

[cuRAND :: CUDA Toolkit Documentation](https://docs.nvidia.com/cuda/curand/device-api-overview.html#device-api-overview)

[CUDA random number generating](https://stackoverflow.com/questions/11832202/cuda-random-number-generating)

it seems like this should solve the issue because it is a PRNG made specifically for cuda and seems to work on the device whatever that means

it seems like the Mersenne Twister PRNG is the best option

Continued integrating slides from professor Chen into my slideshow

8/20/21

Start integrating information about Bayesian thinking and Bayes tests from professor Chen into the presentation. Unfortunately, latex does not seem to work with google slides so I will have to find a workaround ([LaTeX to Image converter](https://latex2image.joeraut.com/), <https://www.codecogs.com/latex/eqneditor.php>)

Started writing research proposal to continue this research

8/18/21

Created a git repository (<https://github.com/lukefly2/CryptoStat-Research>) and started tracking changes using it

Currently trying to create a program that is somewhat random, but I am struggling with generating randomness in the cuda code. Why is there java code and cuda code to do the same thing in CryptoStat? What is the point and why do I have to write code for both of them????

I can’t figure out how to write a prng for cuda while also remaining consistent with the java prng and also most cuda tutorials don’t work because CryptoStat uses pj2 which I don’t understand

I found out that the command I was using earlier

java pj2 Analyze -v "edu.rit.aes.AES256()" "edu.rit.crst.Single(00000000000000000000000000000000)" "edu.rit.crst.Rand(100,123456789)" "edu.rit.crst.Direct()" "edu.rit.crst.Adjacent(8)"

did not have enough data to accurately represent the randomness of AES so I created a different command

java pj2 Analyze -v "edu.rit.aes.AES256()" "edu.rit.crst.Seq(128)" "edu.rit.crst.Rand(100,1234)" "edu.rit.crst.Direct()" "edu.rit.crst.Adjacent(8)"

that feeds about 10x more data and no longer uses 00000000s as the plaintext

8/15/21

So I tried to implement the algorithms without GPU acceleration from pj2, it's not going so great

8/14/21

So I found out that the paper that implements blowfish in cryptostat is completely useless because cryptostat has received two complete rewrites since then so that's cool

My new plan: since I don't really feel like understanding how these algorithms work on a deep enough letter to implement them while also using pj2 (gpu acceleration), I’m going to write an “encryption algorithm” that just spits out a bunch of zeroes no matter what. This should be able to act as a control and lend some credibility to Cryptostat. Hopefully I can at least figure this out even though the documentation isn’t really that helpful.

I”VE DONE ITTITITIT

i managed to write an algorithm that just outputs zeroes and cryptostat correctly evaluates it as non random

8/11/21

Working on presentation

8/10/21

[Factorial of a large number](https://www.geeksforgeeks.org/factorial-large-number/)

[HyperCalc JavaScript, by Robert P. Munafo and Kenny TM~ Chan at MROB](https://mrob.com/pub/comp/hypercalc/hypercalc-javascript.html) (online calculator capable of calculating 1,000,000 factorial, but does not give precise numbers)

created diagrams for encryption and block ciphers in the style of the diagrams for different modes of operation in the article ([NIST SP 800-38A, Recommendation for Block Cipher Modes of Operation Methods and Techniques](https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf))

8/9/21

Still don’t know how to implement DES, but I think I’m getting closer

Found a paper (<https://www.cs.rit.edu/~ark/students/vxs9986/report.pdf>) that implements Blowfish in cryptostat, unfortunately it does not appear to have much code a quick glance, hopefully it describes the process some more than the sparse Cryptostat documentation

List of block ciphers to test:

AES (done)

DES  
3DES  
Blowfish

8/8/21

Still trying to figure out how to use other encryption algorithms with CryptoStat

Still haven’t figured it out. Hopefully I don’t have to learn exactly how different encryption algorithms work and implement them myself. I really hope that I can use other people’s algorithms and just modify them a little bit, but I see no similarities between the implementations I've found online and the one in CryptoStat so I'm not sure how to extend CryptoStat using them.

8/4/21

Trying to extend CryptoStat to support other encryption algorithms

Attended MAA MathFest to see what these sorts of events are like

8/3/21

Solved GPU compatibility error by solving permission issues with executing files on mounted drives by remounting drive with exec option in /etc/fstab. This was causing the GPU compatibility error because CryptoStat was unable to execute the CUDA scripts.

8/2/21

Reinstalled pj2 and CryptoStat

Encountered GPU compatibility error, this is strange because I didn’t have this error before and I am using the same GPU

export CLASSPATH=/media/storage/Projects/CryptoStat\_Research/CryptoStat/lib:/media/storage/Projects/CryptoStat\_Research/pj2/lib

export LD\_LIBRARY\_PATH=/media/storage/Projects/CryptoStat\_Research/pj2:/media/storage/Projects/CryptoStat\_Research/pj2/lib

export GCCBINDIR=/usr/bin/

export CRST\_HOME=/media/storage/Projects/CryptoStat\_Research/CryptoStat

7/28/21

Modify presentation to fit the outline that Professor Chen gave and better fit the new audience

7/27/21

Second meeting with Professor Chen

Presentation outline:

* What is a block cipher
* Test the randomness of a block cipher
* Popular test suites: Nist, Dieharder
* Problems: multiple outputs, modes-of-operation dependent
* Alternative approach: Cryptostat
  + Modes-of-operation independent
  + Bayes test aggregates multiple outputs - find an explanation
  + Empirical results
    - When the null is true - test for something we know is nonrandom
    - When the alternative is true - test for something we know is random
    - Dependence on sample size - find min size that tests as random
    - Sensitivity to prior distribution - see if I can calculate 1 million factorial
* Conclusions and discussions

7/26/21

Bayesian statistics book - [Chapter 1 The Basics of Bayesian Statistics | An Introduction to Bayesian Thinking](https://statswithr.github.io/book/the-basics-of-bayesian-statistics.html)

Modify presentation for nonverbal presentations

7/16/21

First meeting with Professor Chen

learn bayesian thinking

run tests with other encryption algorithms

|  | Pros | Cons |
| --- | --- | --- |
| CryptoStat | Aggregates values  Flexible  Unaffected by modes of operation | H1 p = 0.7  H2 p = uniform (0, 1)  aggregating (outliers overriding average?), log creates very large differences between similar numbers  weak criteria (20 vs 100) |
| NIST | More established | affected by modes of operation |
| DieHarder | More established | affected by modes of operation |

6/16/2021

Start reinstalling linux

6/15/2021

Emails with Professor Chen to set up mentorship

5/5/2021-5/28/2021

Converted paper to presentation form

Added diagrams

4/7/21-4/30/2021

Wrote abstract and fixed literature review (found source for previously uncited mode of operations section, fixed incorrect information, etc.)

Wrote paper

4/4/21

Finished running all tests

Conclusions I could draw:

Cryptostat solves the issue of testing the randomness of the mode of operation instead of the encryption algorithm itself.

4/3/21

Ran all 3 tests on AES-256

4/2/21

Fiiiinnaaalllyyy managed to run CryptoStat without any errors:

java pj2 Analyze "edu.rit.aes.AES256()" "edu.rit.crst.Single(00000000000000000000000000000000)" "edu.rit.crst.Rand(10,123456789)" "edu.rit.crst.Direct()" "edu.rit.crst.Adjacent(8)"

I had to explicitly tell it where to find each class file in order for it to run properly

I can’t use just one key for CryptoStat so I am using a random number generator. I will probably switch to a random key for the other encryption as well.

4/1/21

So apparently rebooting after opening windows first allows me to access ubuntu and avoid the black screen

I have no idea why

Anyways, when I run the command I created before it gives me an error that says that class can’t be found but the file is clearly within the CLASSPATH that I declared

3/31/21

Sooo uhhh for some reason ubuntu just shows a black screen

tried to troubleshoot this issue but I am not very proficient in linux

3/30/21

seems like running CryptoStat is impossible for me due to hardware limitations on my virtual machine because CryptoStat requires a graphics card with CUDA support and passing through my graphics card to my virtual machine does not seem possible

I am going to try to dual boot Ubuntu on my computer, although I have tried this before and was unsuccessful

MANAGED TO DUAL BOOT UBUNTU YAYASYAYYAAY!!!!!

i tried to do this for a whole week last year and didn’t succeed. it kept crashing after every little thing i did

I don’t know what was different this time but it was surprisingly easy

3/29/21

finally came up with a command that should successfully run CryptoStat. I might need to create my own input generation class because I need to be able to use file input.

java pj2 Analyze -v 768 "AES256()" "Single(0)" "Direct()" "Adjacent(100000000)"

currently trying to figure out how Parallel Java 2 and its GPU support works because it is a dependency for cryptostat

Trying to find correct location for native library file

Managed to find the correct location

however i get an error about initialization or something

found the issue, seems like my graphics card isn’t being found

looks like passthrough of graphics card with virtualbox is not supported

welp

3/22/21

why did this guy make running CryptoStat this difficult

3/21/21

reinstalled virtual machine because my initial virtual drive size of 20GB was too small to hold the files I was using

3/20/21

figured out what type of file CryptoStat is

3/19/21

installed NIST on linux

ran NIST on War and Peace, still don’t know what C1 - C10 are but i understand the p-values now

ran dieharder on War and Peace, apparently the famously too long book is not long enough for dieharder

switching plaintext to 10000000000 0s so that all randomness comes from encryption

3/18/21

So uhh, I found out that everything is just way easier on linux

dieharder is only available on linux (unless I want to install cygwin on windows)

openssl provides all of the encryption methods I want along with CBC

still haven’t figured out how to run Cryptostat

3/17/21

Managed to get the NIST test suite to run, however the test results are very confusing and difficult to interpret.

3/14/21

[Random Bit Generation | CSRC](https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software) (download for NIST test suite)

3/5/21

[Java Security Overview](https://docs.oracle.com/javase/9/security/java-security-overview1.htm#JSSEC-GUID-2EF91196-D468-4D0F-8FDC-DA2BEA165D10)

[javax.crypto (Java Platform SE 7 )](https://docs.oracle.com/javase/7/docs/api/javax/crypto/package-summary.html)

installed eclipse

2/28/21

Trying to figure out java crypto and java security so that I can run the encryption algorithms

So uhh I should probably brush up on my java

2/11/21

Trying to figure out how to run NIST, found a java implementation on github

2/10/21

Worked on literature review (wrote **modes of operation** and **multiple scores** sections)

2/8/21

Worked on literature review (fixed intro)

2/7/21

Block Ciphers: take input of a set size and then output the same size

Modes of Operation - how block ciphers are used on inputs that are not exactly the right size

* Electronic Codebook (ECB) - in order, reveals information when it is repeated (penguin pic)
* Cipher Block Chaining (CBC) - each block is dependent on the previous block to be encrypted or decrypted
* Counter - encrypted counter and then xor with input
* Galois Counter - counter with message authentication

[STATISTICAL TESTING of RANDOMNESS: NEW and OLD PROCEDURES appeared as Chapter 3 in Randomness through Computation, H. Zenil ed.](https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=903151)

[Cryptographic Randomness Testing of Block Ciphers and Hash Functions](https://eprint.iacr.org/2010/564.pdf)

[Evaluation of Randomness Test Results for Short Sequences](https://link.springer.com/chapter/10.1007/978-3-642-15874-2_27)

[A New Statistical Testing for Symmetric Ciphers and Hash Functions](https://link.springer.com/chapter/10.1007/3-540-36159-6_29)

1/29/21

[HotBits: Genuine Random Numbers](https://www.fourmilab.ch/hotbits/) - genuinely random numbers

[Randomness Analysis on Lightweight Block Cipher, PRESENT](https://thescipub.com/pdf/jcssp.2020.1639.1647.pdf) - NIST

[TestU01 and Practrand: Tools for a randomness evaluation for famous multimedia ciphers](https://hal.archives-ouvertes.fr/hal-02993846/document)

<http://ezlfcc.vccs.edu:2048/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=15585217&site=ehost-live&scope=site> - Finally a block cipher source on LFCC

1/28/21

CRYPTOSTAT LINK: <https://www.cs.rit.edu/~ark/parallelcrypto/cryptostat/>

Alani, M.M. (2010). Testing Randomness in Ciphertext of Block-Ciphers Using DieHard Tests. [(PDF) Testing Randomness in Ciphertext of Block-Ciphers Using DieHard Tests](https://www.researchgate.net/publication/268414157_Testing_Randomness_in_Ciphertext_of_Block-Ciphers_Using_DieHard_Tests) - similar to my study

[Testing the Randomness of Cryptographic Function Mappings](https://eprint.iacr.org/2019/078.pdf) - uses a special randomness test designed for cryptographic functions that produce fixed-length output

[Cryptographic Randomness Testing of Block Ciphers and Hash Functions](https://eprint.iacr.org/2010/564.pdf) - Another study similar to mine

[Randomness testing of the advanced encryption standard finalist candidates](https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir6483.pdf)

[NISTIR 6390, Randomness Testing of the Advanced Encryption Standard Candidate Algorithms](https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=151193)

What are partial encryption schemes? And can tests for them be applied to normal encryption?

MEASURING THE STRENGTH OF PARTIAL ENCRYPTION SCHEMES <https://ieeexplore-ieee-org.ezlfcc.vccs.edu:2443/stamp/stamp.jsp?tp=&arnumber=1530258&tag=1>

[On the design of partial encryption scheme for multimedia content](https://www.sciencedirect.com/science/article/pii/S0895717711003487#:~:text=In%20the%20partial%20encryption%20scheme,defined%20by%20Shannon%20%5B30%5D).

1/27/21

[A Literature Survey on Efficiency and Security of Symmetric Cryptography](http://ijcsn.org/IJCSN-2017/6-3/A-Literature-Survey-on-Efficiency-and-Security-of-Symmetric-Cryptography.pdf)

Reading articles for literature review

1/15/21

[A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications](https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762) (131 pages)

Forward unpredictability: you can’t guess the next number

Backward unpredictability: you can’t guess the seed

This is for PRNGs but I think it could be applied to encryption algorithms

“RNGs” are typically predictable, only appear random, and are time consuming

PRNGs take RNG numbers as seeds and convert them into something actually random

Need to read more sources in order to understand the field more

One problem I have is that I am trying to combine two fields (encryption, randomness testing) for my research, this makes it so that I have to summarize much more. I wonder if there's anything i can do to reduce what i have to research

12/21/20

Trying to understand how java crypto and java security work

“Encryption Key is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the cipher text.

Decryption Key is a value that is known to the receiver. The decryption key is related to the encryption key, but is not always identical to it. The receiver inputs the decryption key into the decryption algorithm along with the cipher text in order to compute the plaintext.” - [Java Cryptography - Keys](https://www.tutorialspoint.com/java_cryptography/java_cryptography_keys.htm)

12/20/20

Installed virtual machine Ubuntu 20.4 LTS

Ran into lots of permission issues

12/18/20

Finished up proposal

Honestly not all that satisfied with the literature review though

12/17/20

Finished everything except the literature review

Not sure how to do a literature review

12/12/20

Working on research proposal

Pseudorandom numbers often will not exhibit all of the properties that would be expected of a truly random sequence

Found explanation for why many tests are used

12/11/20

<https://www.sciencedirect.com/science/article/pii/S1877050916001101> - comparison of common crypto algorithms

“In information security, we require security algorithms to yield high randomness in encrypted messages, so that there is less or no dependency between key and ciphertext.”

Confusion: complex relationship between key and ciphertext

They calculated entropy using Shannon’s formula

Cryptographic hash and non-cryptographic hashes have no fundamental differences. Cryptographic hashes are just more secure (less collisions) in exchange for performance and speed.

Hashes convert something of any size to a string of fixed size

Can be used to compare files

Checksums are even less secure than non-cryptographic hashes but are even faster and typically used to check for file corruption

*A Practical Approach to Testing Random Number Generators in Computer Algebra Systems.* - very recent source that describes many randomness tests and how to use them

12/10/20

Encryption - can be reversed, often uses a key

Hash - can’t be reversed, each input will always produce the same output

*Randomness testing of non-cryptographic hash functions for real-time hash table based storage and look-up of URLs*. - Article on testing randomness of hashes

<https://vcc-lof-primo.hosted.exlibrisgroup.com/permalink/f/1t1hca3/TN_cdi_crossref_primary_10_32614_RJ_2016_016> - test for cryptographic randomness

12/8/20

Just found out today that dieharder can take files as input. This is a big relief and means that I finally have a reliable randomness test. However, I don’t know how I would justify its use in my research paper. Remember to ask Ms. Klus about this.

Dieharder appears to require a large amount of data though

[rngtest(1) - Linux man page](https://linux.die.net/man/1/rngtest) RNG test

12/6/20

My main goal right now is to find a reliable unit test for randomness

Komogorov Smirnov is used to compare two different curves or distributions, this does not seem to be a very good measure of randomness

12/5/20

Kolmogorov Smirnov tests for uniform distribution (available in Excel)

I figured out how to use liltest but i don’t know what the output means

Dieharder is used to test random number generators themselves not their output

Diehard tests output, but I don’t know if it is still considered effective

Still can’t find any people using compression to test for randomness

12/4/20

Kolmogorov complexity: random if the string is shorter than the computer program that needs to be written in order to output it. This is different than what I previously thought and does not appear to be easily done on many different strings.

I tried my own test of randomness and got some positive results, though I don’t know how to prove its effectiveness. I used a compression (DEFLATE) program and found that a text file of 01010101...010101 was much smaller when compressed than a text file that was random 01110101...100001. If I can find a source that approves this method of randomness testing, it would be much easier than the other randomness test. See if testing randomness with compression is viable and proven.

[Testing Randomness with Compression](https://www.mapleprimes.com/maplesoftblog/95731-Testing-Randomness-With-Compression)

[Is compressibility a good test for randomness of a pseudorandom sequence?](https://math.stackexchange.com/questions/3772131/is-compressibility-a-good-test-for-randomness-of-a-pseudorandom-sequence)

“It is straightforward to compute upper bounds for K(s) – simply compress the string s with some method, implement the corresponding decompressor in the chosen language, concatenate the decompressor to the compressed string, and measure the length of the resulting string – concretely, the size of a self-extracting archive in the given language.” [Kolmogorov complexity](https://en.wikipedia.org/wiki/Kolmogorov_complexity)

[Yongge Wang's Home Page](https://webpages.uncc.edu/yonwang/) Liltest

I need to find an automated randomness test.

[Kolmogorov–Smirnov test](https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test)

11/29/20

I believe that when asked to rate randomness, people actually rate the lack of order. To test this idea, I could ask people to rate randomness, and then ask them to rate order and see if there is an inverse relationship.

Compare randomness of different encryption algorithms

Compare randomness of different random number generation methods

11/20/20

Trying to find a better research question

Read about gambler’s fallacy and hot hand fallacy

I think it is interesting that there are two almost opposing fallacies in the same field.

Sympathetic magic and perceptions of randomness: The hot hand versus the gambler's fallacy.

11/13/20

Finished research proposal

11/6/20

Working on my research proposal.

[The definition of random sequences - ScienceDirect](https://doi.org/10.1016/S0019-9958(66)80018-9)

This link is to Martin Lof, the creator of a popular algorithm for determining randomness

<https://doi.org/10.1037/a0014821>

Review of previous research

There are opposing biases - gambler’s fallacy and hot-hand fallacy

How is it possible for this to occur?

I decided to focus on the differences between perceptions of randomness for different generation methods

10/30/20

Trying to differentiate my research from previous research

Ideas:

1. Compare how humans perceive randomness to a different way to measure randomness, ex: Kolmogorov complexity, Martin-Lof randomness
2. Ask people to rate the randomness of sequences generated with a set of elements that aren’t equally likely to appear, ex: rolling two dice

It seems that randomness is typically measured by measuring how “compressible” it is. Which is sort of like finding how many patterns it has. What if I compare how people rate randomness to how an algorithm rates randomness?

[Independence, Relative Randomness, and PA Degrees](https://projecteuclid.org/euclid.ndjfl/1390246434)

[Algorithmically random sequence](https://en.wikipedia.org/wiki/Algorithmically_random_sequence)

10/23/20

I read up on my previous two ideas for differentiating my research and decided they weren’t viable.

10/16/20

Reading other sources

Started doing tensorflow tutorial

10/14/20

Reading sources found previously and writing annotated bibliography

Kahneman & Tversky, 1972

1. People think a sequence is more likely, and hence random, if there is some irregularity in order of appearance (e.g., HHTHTH vs. HTHTHT).
2. People think a sequence is more likely, and hence random, if the equiprobable outcomes occur equally often.
3. The outcome alternation rate (i.e., how often H switches to T and vice versa) that people consider to be random is higher than that associated with chance.

Belief that local and global sequences should share the same properties → Gambler’s fallacy (ex: if there is a long streak of heads, the next flip is more likely to be tails)

10/12/20

Trying to differentiate my research from previous research

Ideas:

1. Considering including encryption / passwords as part of the project (no real reason to do this)
2. Maybe generate noise? Scatterplot analysis

10/9/20

Pruning sources / articles list

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5933241/>

<https://psycnet.apa.org/record/2009-04438-008>

10/2/20

Computers can’t easily generate random numbers. They typically produce pseudo-random numbers using a function that takes in a seed. In order to produce more random numbers, they measure the environment, such as the atmosphere, exact time or mouse position. (I currently have no source for this)

<http://ezproxy.vccs.edu:2048/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=144770611&site=ehost-live&scope=site>

True RNG vs Pseudo RNG

9/25/20

I read some of the articles below and expanded on their descriptions.

Changed main idea that is being considered to: teach a computer to generate or recognize sequences that would be perceived as perfectly random by humans.

I’m also trying to find alternative ideas so that I have backups.

9/24/20

<https://www.jstor.org/stable/1251762>

Music and marketing

<https://www.jstor.org/stable/10.1525/mp.2002.20.2.151>

Effect of tempo and mode on arousal and mood

9/23/20

<https://www.jstor.org/stable/1422628>

Recognizing randomness with sequences of heads and tails.

Perceived randomness isn’t based on how likely something was produced by random but rather how unlikely something was produced unnaturally.

Others argue that perception of randomness is based on a representativeness heuristic. Runs are less locally representative of equal distribution of randomness. Patterns are less representative of the disorder of randomness.

<https://www.jstor.org/stable/1434983>

Understanding imperfect randomness in test scores

<https://www.jstor.org/stable/20012563>

Randomness and its effect on free will vs determinism

<https://www.sciencedirect.com/science/article/pii/019688589190029I?via%3Dihub>

Summary of previous research (1991)

You can’t know if something is random after the fact, you can only truly know by knowing the generation method. There are no logical or physical proofs of randomness, so people have to rely on intuition.

Judgement tasks are better than production tasks.

<https://plato.stanford.edu/archives/spr2018/entries/chance-randomness/>

Difference between chance and randomness

9/18/20

5 research ideas

1. Perceived randomness
2. Something to do with computers / ai (maybe reading letters?)
3. How and why does music affect mood?
4. How does thinking in different languages change your thoughts / actions?
5. Echolocation device