Distribution of Letters on the Web as a Random Number Generator

Florida International University

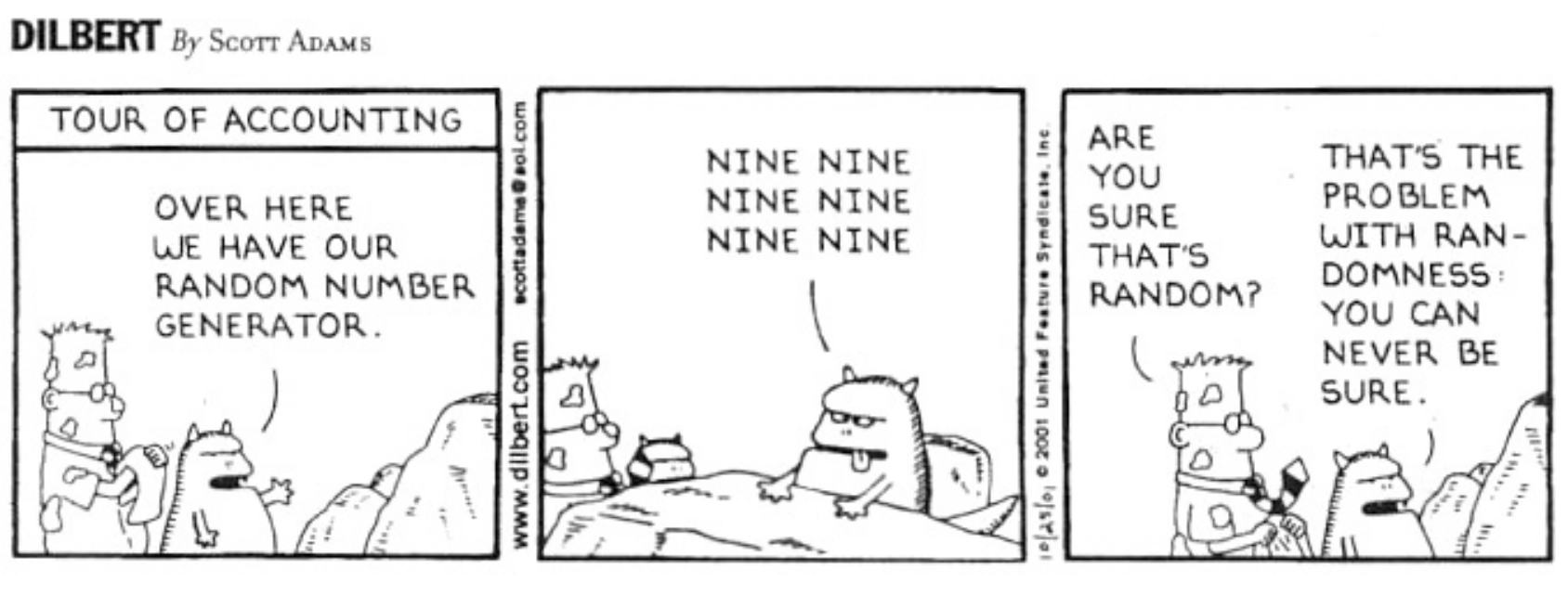
PHY-4821L Fall 2012

By: Thomaz L Santana

Can you use the distribution of letters on the internet as a random number generator?

- What is a random sequence of numbers?
- Methods of generating pseudorandom numbers.
- Testing numbers for randomness
- How to read many web pages
- How to convert letters to numbers
- Results

What is a random sequence of numbers?



What is a random sequence of numbers?

- A numeric sequence is said to be statistically random when it contains no recognizable patterns (e.g. the result of a dice roll or the digits of π)

- Global randomness and local randomness

What is a random sequence of numbers?

- Pseudorandom numbers are numbers that appear to be random but are generated by a deterministic process.

Testing for Randomness

- Frequency test
- Gap test
- Serial test
- Poker test

Frequency Test

- Uses the chi-square test to compare the distribution of the numbers generated to a theoretical distribution (the distribution should be uniform.)

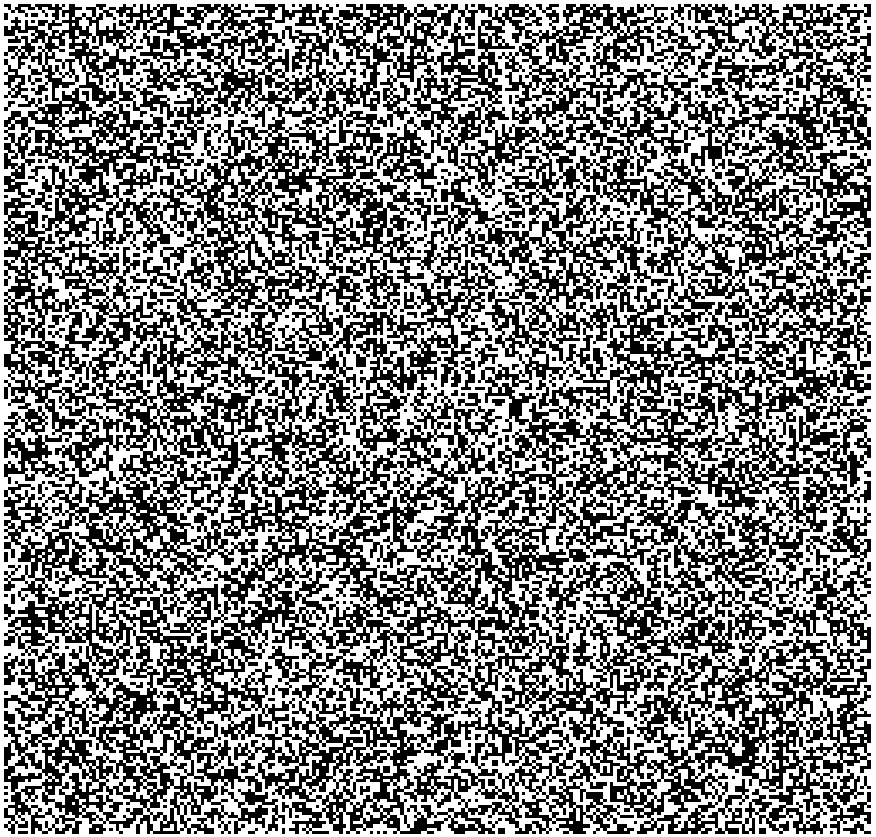
Gap Test

- Counts the number of digits that appear between repetitions of a particular digit and then uses the chi-squared test to compare it with the expected frequency of gaps.

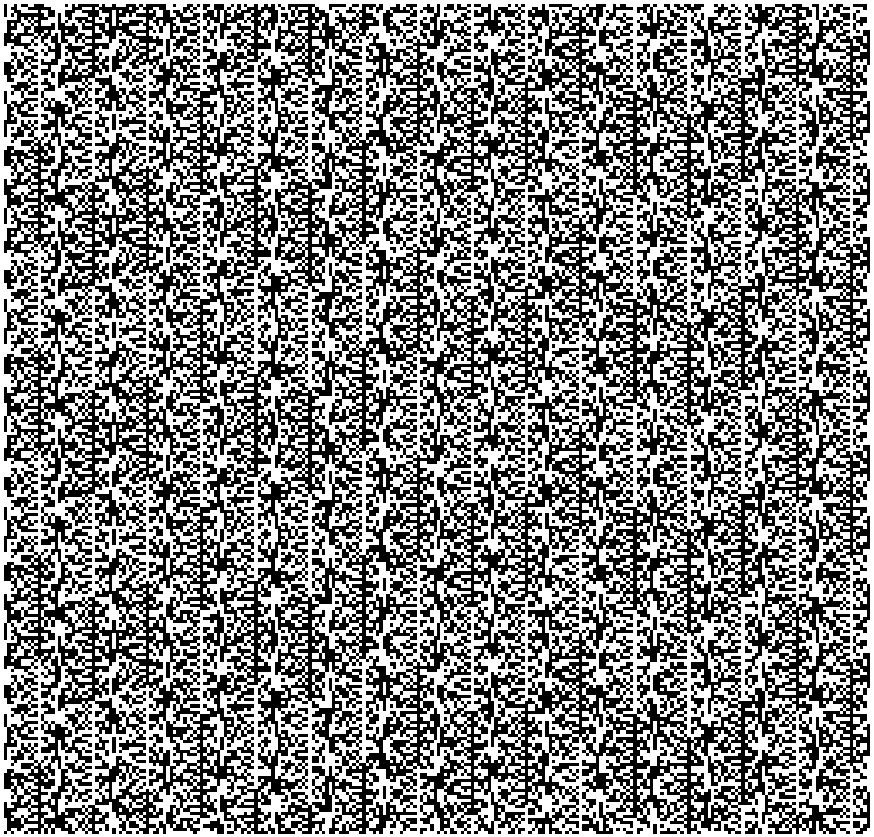
Serial Test

- Same as the frequency test but compares two digits at a time. (e.g. 00, 01, 02)

Visual Example



RANDOM.ORG



PHP rand() on Microsoft Windows

Source: random.org

Generating Pseudorandom Numbers

MidSquare Method:

- $X_0 = 4567$ (seed)
- $X_0^2 = 20857489$
- $X_1 = 8574$
- $X_1^2 = 73513476$
- $X_2 = 5134$
- $X_2^2 = 26357956$
- $X_3 = 3579$

How to Read Many Web Pages

- I used Wikipedia pages.
- Python modules:
  - urllib2
    - Use to create user-agent and handle http
  - BeautifulSoup
    - Use parser to get the text your interested in
- Only use the text in the paragraphs of the page. (i.e. ignore the common words on every page e.g. “main page”, “wikipedia”)
- Loop over a few hundred pages and save the letters in a file.

From Letters to Numbers

- Create a dictionary relating letters to numbers from 0 to 9.
  - a=0, b=1, c=2, d=3, e=4, f=5, g=6, h=7, i=8, j=9, k=0, etc
- The dictionary will be the “seed”
- In Python a dictionary relates a key to a value

From Letters to Numbers

- The problems:
  - The 26 letters are not evenly divided by 10 digits
  - The letter ‘e’ is way more popular than ‘z’
- The solution:
  - Create a counter that counts by one after each letter and resets to zero after it reaches 10
- How will this help?

From Letters to Numbers

- The following sequence of letters will produce the following numbers:

a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a	a
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7

L	e	t	t	e	r	s
1	5	1	2	8	2	4

- The algorithm:

$$Number = (Dictionary[letter] + \text{counter})\%10$$

From Letters to Numbers

- Dictionary[letter] is the numerical value of the letter (e.g. a=0, b=1, c=2, etc)
- Counter is the position of the letter in the sequence and gets reset to zero after it reaches ten
- %10 is the remainder when the two values are divided by 10

From Letters to Numbers

- The letter ‘a’ has the value zero but does its position in a sequence of letters have a pattern?
- Recall:

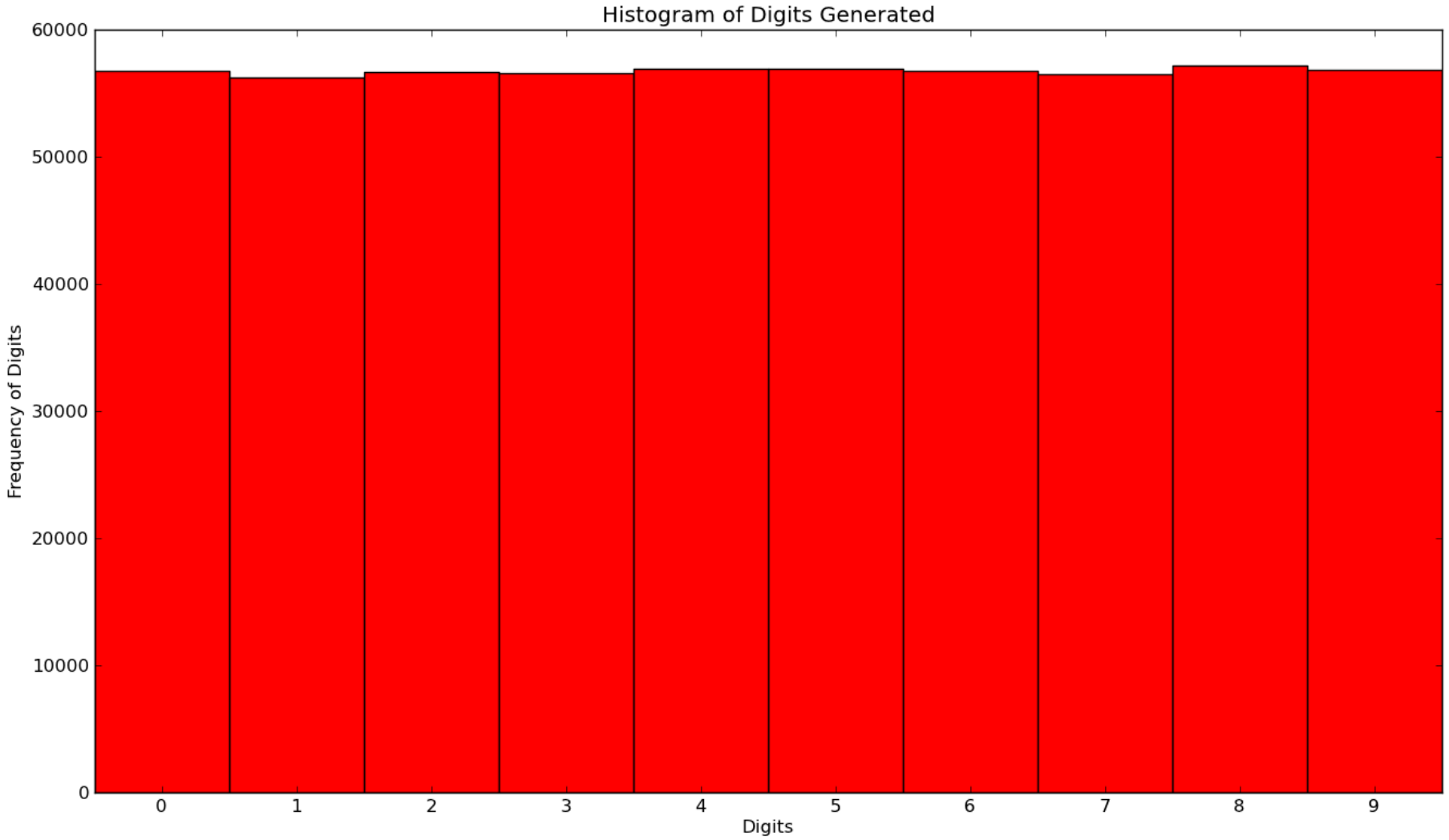
$$Number = (Dictionary[letter] + \text{counter})\%10$$

If there is no pattern to the distribution of ‘a’ (and every other letter) then we can expect a uniform distribution of numbers from 0 – 9.

Results

- Expect each digit from 0-9 to have an equal probability (P=0.1)

Results


Histogram of Digits Generated
Frequency of Digits
Digits
60000
50000
40000
30000
20000
10000
0
0
1
2
3
4
5
6
7
8
9

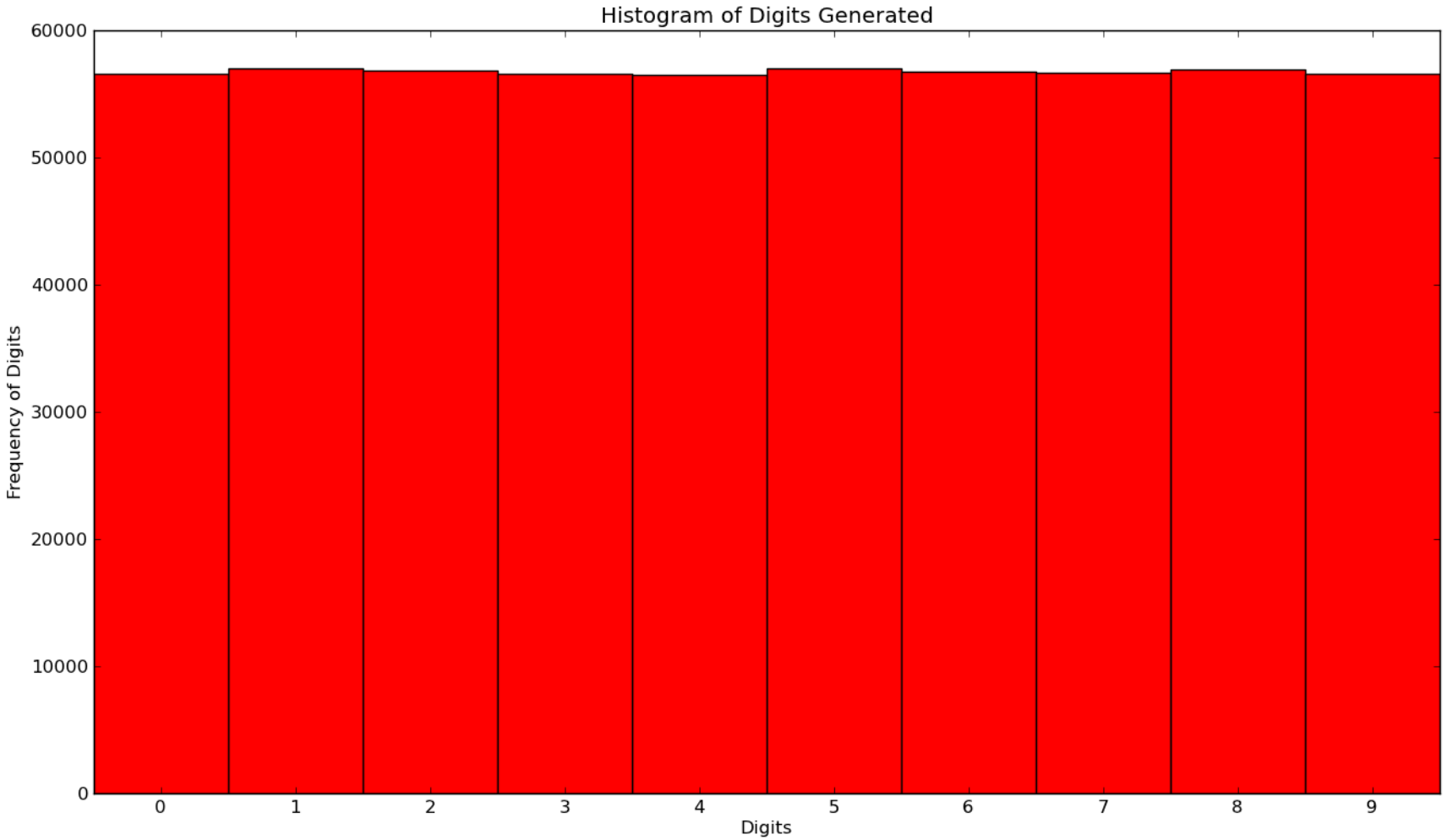
Results

Digit	F(observed)	Probability	F(expected)	Uncertainty	Chi-Squared
0	56798	0.1	56767	238	0.0175
1	56265	0.1	56767	237	4.4700
2	56692	0.1	56767	238	0.0979
3	56609	0.1	56767	238	0.4382
4	56962	0.1	56767	239	0.6710
5	56906	0.1	56767	239	0.3420
6	56789	0.1	56767	238	0.0089
7	56557	0.1	56767	238	0.7760
8	57198	0.1	56767	239	3.2552
9	56889	0.1	56767	239	0.2638

$$X^2/\nu = 1.15$$

Results

The same frequency test using Python's random number generator



Results

Digit	F(observed)	Probability	F(expected)	Uncertainty	Chi-Squared
0	56622	0.1	56766.5	237.95378	0.36876567
1	57006	0.1	56766.5	238.75929	1.00621426
2	56875	0.1	56766.5	238.4848	0.20698462
3	56592	0.1	56766.5	237.89073	0.53806633
4	56492	0.1	56766.5	237.68046	1.3338216
5	57042	0.1	56766.5	238.83467	1.33060289
6	56800	0.1	56766.5	238.32751	0.01975792
7	56666	0.1	56766.5	238.04621	0.1782418
8	56964	0.1	56766.5	238.67132	0.68475265
9	56606	0.1	56766.5	237.92015	0.45507985

$$X^2/\nu = 0.681$$

Results

- Using the numbers generated from letters on the internet what would a Monte Carlo computation of PI give?

- PI: 3.137
  - Iterations: 40547
  - Inside r: 31802

Results

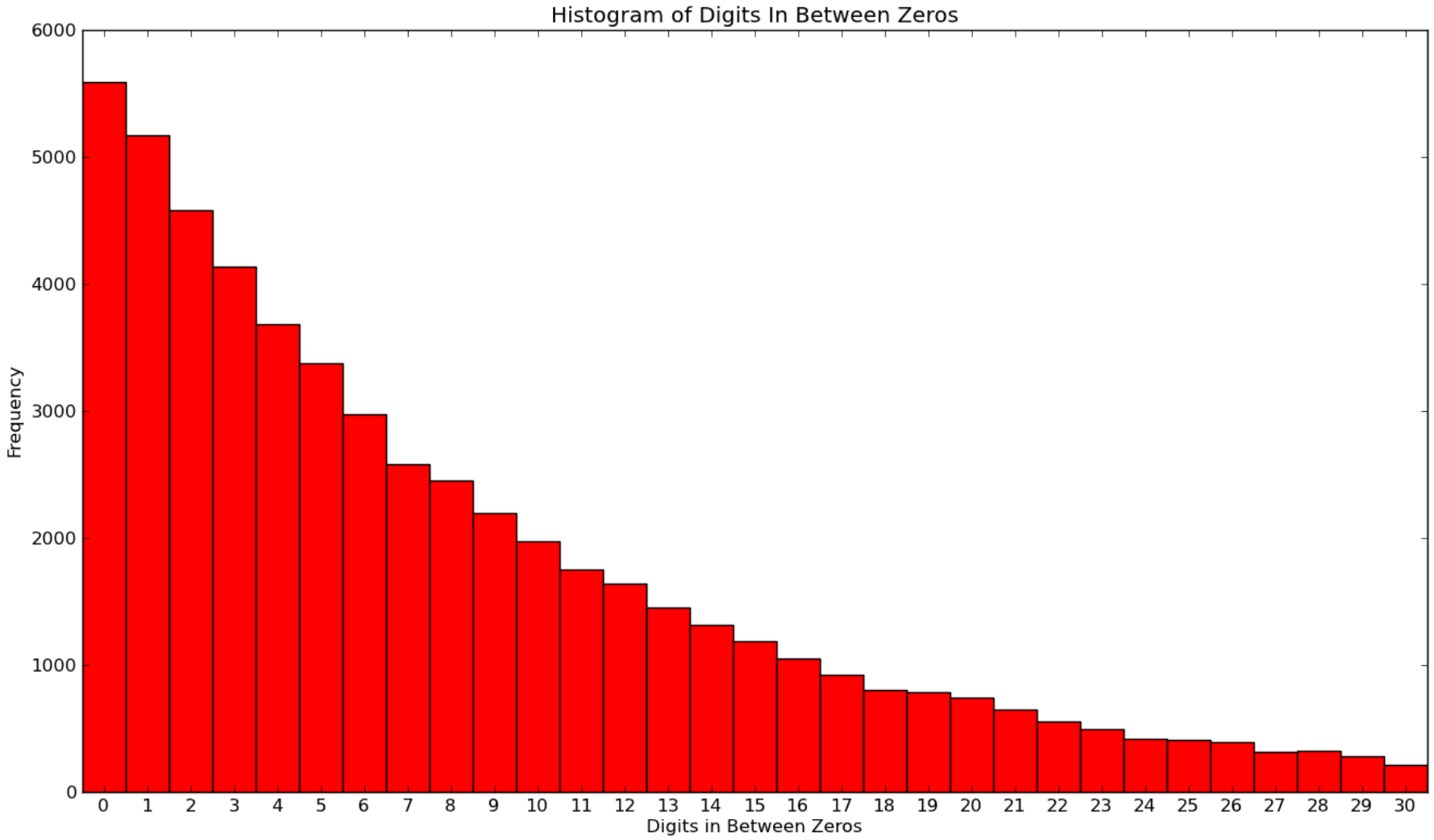
- Using Python's random number generator and the same amount of iterations
  - PI: 3.147

Results

- Globally random
  - But are the numbers locally random?
- Let's see the Gap Test
  - 00 == zero digits between zeros
  - 040 == one digit between zeros
  - 07830 == three digits between zeros
- Probability of Gap space
  - $p_k = p(1-p)^k$

Results

Python's random number generator



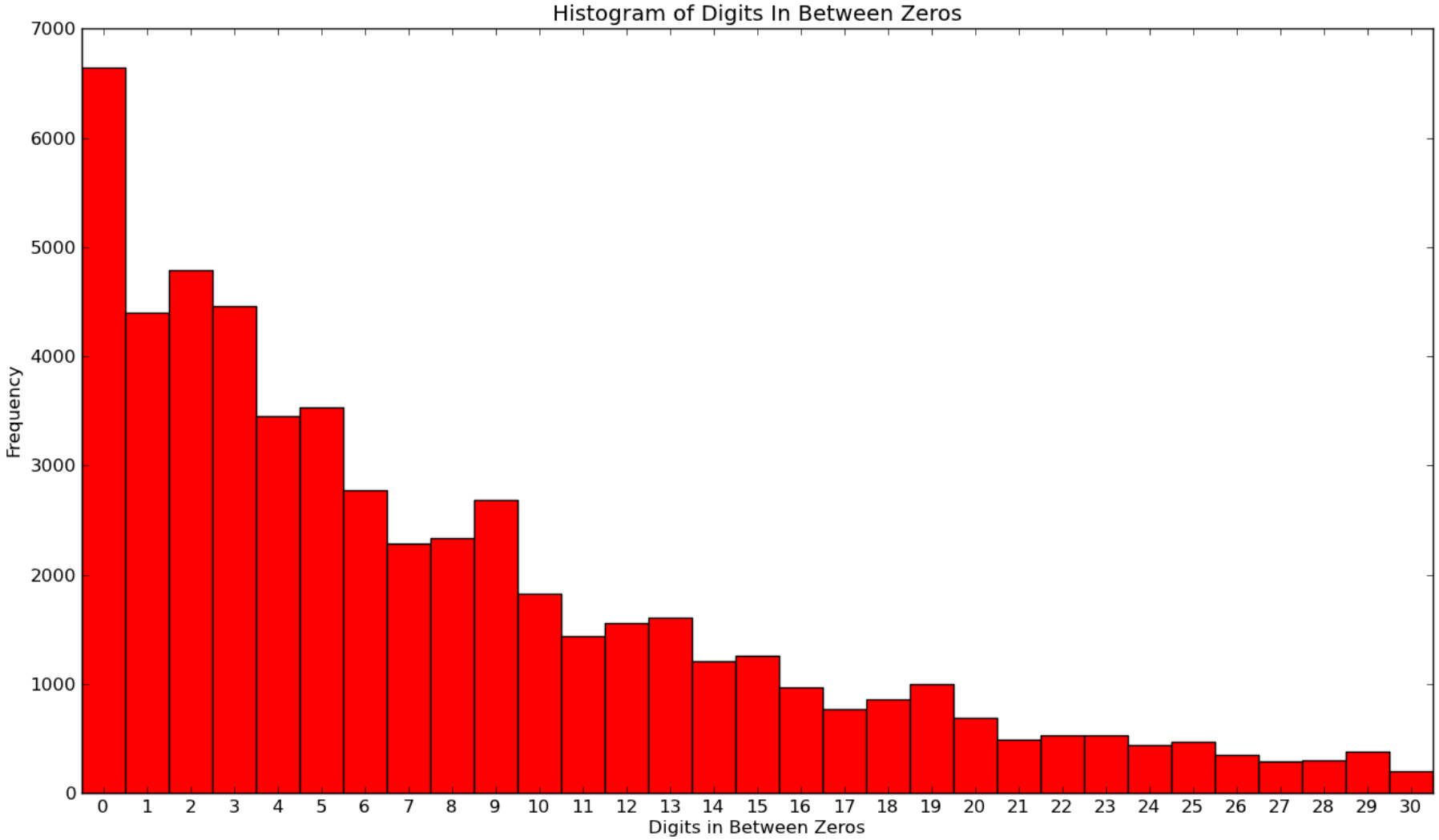
Results

Between Zero	F(observed)	Probability	F(expected)	Uncertainty	Chi-Squared
0	5595	1.00E-01	5446	75	3.95
1	5170	9.00E-02	4902	72	13.92
2	4582	8.10E-02	4412	68	6.34
3	4143	7.29E-02	3970	64	7.19
4	3684	6.56E-02	3573	61	3.32
5	3375	5.90E-02	3216	58	7.49
6	2978	5.31E-02	2894	55	2.34
7	2580	4.78E-02	2605	51	0.24
8	2451	4.30E-02	2344	50	4.63
9	2196	3.87E-02	2110	47	3.36
10	1974	3.49E-02	1899	44	2.85
11	1755	3.14E-02	1709	42	1.20
12	1640	2.82E-02	1538	40	6.32
13	1455	2.54E-02	1384	38	3.43
14	1316	2.29E-02	1246	36	3.73
15	1186	2.06E-02	1121	34	3.52
16	1053	1.85E-02	1009	32	1.82
17	929	1.67E-02	908	30	0.46
18	807	1.50E-02	817	28	0.14
19	786	1.35E-02	736	28	3.22
20	742	1.22E-02	662	27	8.59
21	648	1.09E-02	596	25	4.18
22	560	9.85E-03	536	24	1.00
23	495	8.86E-03	483	22	0.31
24	418	7.98E-03	434	20	0.65
25	412	7.18E-03	391	20	1.07
26	392	6.46E-03	352	20	4.10
27	317	5.81E-03	317	18	0.00
28	325	5.23E-03	285	18	4.91
29	280	4.71E-03	257	17	1.97
30	220	4.24E-03	231	15	0.54

$$X^2/\nu = 3.68$$

Results

Letters from the web to numbers



Results

Between Zero	F(observed)	Probability	F(expected)	Uncertainty	Chi-Squared
0	6639	1.00E-01	5454	81	211.51
1	4395	9.00E-02	4909	66	60.02
2	4793	8.10E-02	4418	69	29.38
3	4461	7.29E-02	3976	67	52.74
4	3453	6.56E-02	3578	59	4.55
5	3534	5.90E-02	3221	59	27.80
6	2778	5.31E-02	2898	53	5.23
7	2290	4.78E-02	2609	48	44.33
8	2339	4.30E-02	2348	48	0.03
9	2680	3.87E-02	2113	52	119.96
10	1825	3.49E-02	1902	43	3.22
11	1440	3.14E-02	1712	38	51.20
12	1559	2.82E-02	1540	39	0.22
13	1609	2.54E-02	1386	40	30.81
14	1207	2.29E-02	1248	35	1.37
15	1260	2.06E-02	1123	35	14.91
16	973	1.85E-02	1011	31	1.46
17	774	1.67E-02	910	28	23.75
18	857	1.50E-02	819	29	1.72
19	1001	1.35E-02	737	32	69.76
20	688	1.22E-02	663	26	0.90
21	491	1.09E-02	597	22	22.79
22	527	9.85E-03	537	23	0.19
23	534	8.86E-03	483	23	4.80
24	438	7.98E-03	435	21	0.02
25	470	7.18E-03	392	22	13.10
26	351	6.46E-03	352	19	0.01
27	290	5.81E-03	317	17	2.54
28	306	5.23E-03	285	17	1.38
29	377	4.71E-03	257	19	38.27
30	201	4.24E-03	231	14	4.54

$$X^2/\nu = 29.1$$

Results

- Not locally random

- Why?
  - The English language has low information entropy
  - If you know the position of the letter ‘T’ you know that the chances of the next letter being ‘z’ is unlikely but the letter ‘h’ is.