Checking our intuition about estimates, via R

R Meetup, Mumbai

Radha

June 21, 2014

Outline

- Stainslaw Ulam
  - Manhattan project - Outcome of Neutrons.
  - Probabilistic simulation to solve problems.
  - Idea came to his mind, when he was sick, lying in his bed and playing Solitaire.
- Most of the educational curricula at the undergraduate or graduate level does not equip students with *basic understanding of probability.*
- Lack of *simulation skills* + Lack of *available software*
- Thanks to `R`, we can all do *Street fighting Statistics*

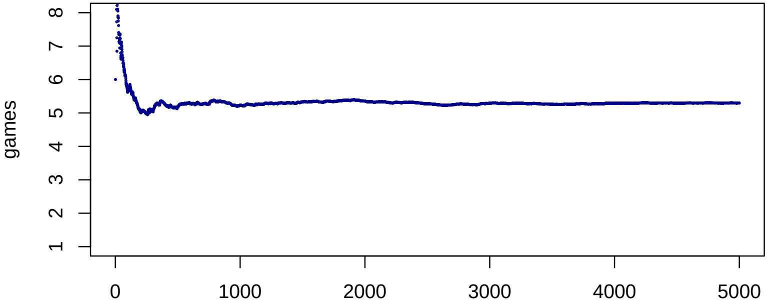
Odd man out - What's your intuition ?

Along with five of your friends, you decide to have drinks at a bar. Who should foot the bill ? You all agree to play the "odd man out" game which goes like this :

Each of you toss a coin. If there are 5H+1T or 5T+1H, the "odd man out" foots the bill. If there is any other combination, you play the game again. On an average, how many games would be played before the "odd man" is decided ?

```
trial    <- function(n){
  reps   <- replicate(n, {
    x    <- sum(rbinom(6,1,0.5))
    x == 5 | x == 1
  })
   which(reps==1)[1]
}
simulations <- replicate(5000, trial(100))
print( paste("mean = ",round(mean(simulations),1),
             " sd = ",round(sd(simulations),1)))

## [1] "mean =  5.3  sd =  4.8"
```



- Probability of success , $p = \binom{6}{1}(1/2)^6 + \binom{6}{5}(1/2)^6 = 12/64$
- Expected number of games is $1/p = 64/12 = 5.33$

Odd man out + Sholay coin

Along with five of your friends, you decide to have drinks at a bar. Who should foot the bill ? You all agree to play the "odd man out" game which goes like this :

Each of you toss a coin. If there are 5H+1T or 5T+1H, the "odd man out" foots the bill. If there is any other combination, you play the game again. On an average, how many games would be played before the "odd man" is decided ?

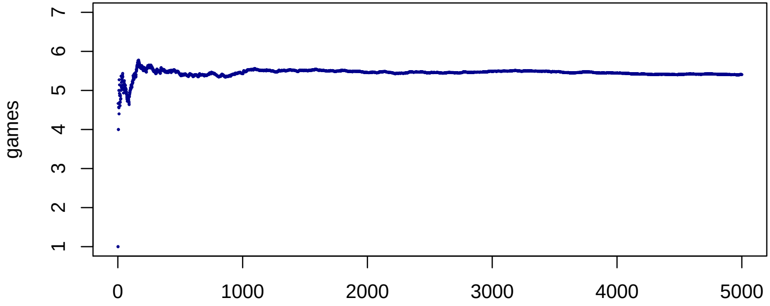
AND

You have a *Sholay* coin

```r
trial    <- function(n){
  reps   <- replicate(n, {
    x    <- sum(rbinom(6,1,prob=c(rep(0.5,5),0.999)))
    x == 5 | x == 1
  })
   which(reps==1)[1]
}
simulations <- replicate(5000, trial(100))
print( paste("mean = ",round(mean(simulations),1),
             " sd = ",round(sd(simulations),1)))

## [1] "mean =  5.4  sd =  5"
```

- Biased coin does not alter the result



- By having a *Sholay* coin, do you have any advantage ?

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate
  - Polling your neighbors is surprisingly a robust method.

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate
  - Polling your neighbors is surprisingly a robust method.
- But

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate
  - Polling your neighbors is surprisingly a robust method.
- But
  - Sparsity of neighbors

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate
  - Polling your neighbors is surprisingly a robust method.
- But
  - Sparsity of neighbors
  - If the method tries to befriend a critical number of neighbors, then the method is no longer local.

Curse of Dimensionality

Nearest Neighbor methods dread this animal.

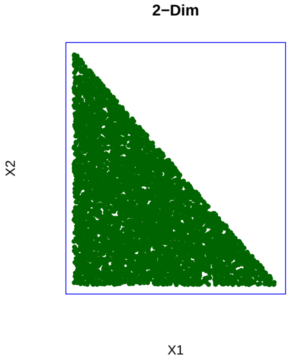
- When the number of input variables increase :
  - Nonlinear relationships are hard to estimate
  - Polling your neighbors is surprisingly a robust method.
- But
  - Sparsity of neighbors
  - If the method tries to befriend a critical number of neighbors, then the method is no longer local.
- What does all this mean ? Sounds intuitively right , but how do we check it ?

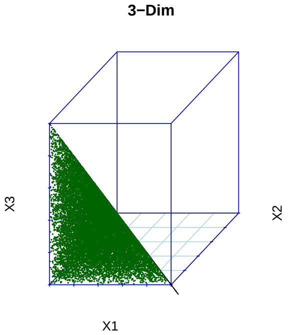
Estimating the probability - - What's your intuition ?

Let's consider N uniform random variables as predictors. Let's say we are interested in predicting the value of y at origin.
Rule for choosing neighbors : $X_1 + X_2 + \ldots + X_N \leq 1$

Estimating the probability - - What's your intuition ?

Let's consider N uniform random variables as predictors. Let's say we are interested in predicting the value of y at origin.
Rule for choosing neighbors : $X_1 + X_2 + \ldots + X_N \leq 1$





For N dimensions ?

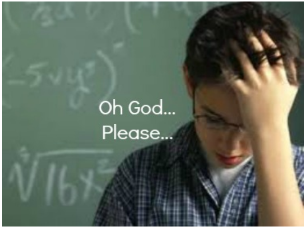
$$P(X_1 + X_2 + \ldots + X_N \leq 1) =$$

$$\int_0^1 \int_0^{1-x_1} \int_0^{1-x_1-x_2} \ldots \int_0^{1-x_1-x_2-\ldots-x_{n-1}} dx_1 dx_2 \ldots dx_n$$

For N dimensions ?

$$P(X_1 + X_2 + \ldots + X_N \leq 1) =$$

$$\int_0^1 \int_0^{1-x_1} \int_0^{1-x_1-x_2} \ldots \int_0^{1-x_1-x_2-\ldots-x_{n-1}} dx_1 dx_2 \ldots dx_n$$



```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$ 0.1646

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$ 0.1646

$P(X_1 + X_2 + X_3 + X_4 \leq 1) =$

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$ 0.1646

$P(X_1 + X_2 + X_3 + X_4 \leq 1) =$ 0.0414

```r
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$ 0.1646

$P(X_1 + X_2 + X_3 + X_4 \leq 1) =$ 0.0414

$P(X_1 + X_2 + X_3 + X_4 + X_5 \leq 1) =$

```
prob <- function(N) {
    mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$P(X_1 + X_2 \leq 1) =$ 0.4995

$P(X_1 + X_2 + X_3 \leq 1) =$ 0.1646

$P(X_1 + X_2 + X_3 + X_4 \leq 1) =$ 0.0414

$P(X_1 + X_2 + X_3 + X_4 + X_5 \leq 1) =$ 0.0086

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$ 6.0761

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

E(data points for 2 dim) = 2.0018

E(data points for 3 dim) = 6.0761

E(data points for 4 dim) =

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$ 6.0761

$E(\text{data points for 4 dim}) =$ 24.1429

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$ 6.0761

$E(\text{data points for 4 dim}) =$ 24.1429

$E(\text{data points for 5 dim}) =$

```
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$ 6.0761

$E(\text{data points for 4 dim}) =$ 24.1429

$E(\text{data points for 5 dim}) =$ 116.0093

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

E(data points for 2 dim) = 2.0018

E(data points for 3 dim) = 6.0761

E(data points for 4 dim) = 24.1429

E(data points for 5 dim) = 116.0093

E(data points for N dim) =

```r
points <- function(N) {
    1/mean(replicate(1e+05, sum(runif(N)) < 1))
}
```

$E(\text{data points for 2 dim }) =$ 2.0018

$E(\text{data points for 3 dim}) =$ 6.0761

$E(\text{data points for 4 dim}) =$ 24.1429

$E(\text{data points for 5 dim}) =$ 116.0093

$E(\text{data points for N dim}) =$????

Estimating the probability

Let's consider N uniform random variables as predictors. Let's say we are interested in predicting the value of y at origin.
Rule for choosing neighbors : $X_1 + X_2 + \ldots + X_N \leq 1$

Estimating the probability

Let's consider N uniform random variables as predictors. Let's say we are interested in predicting the value of y at origin.
Rule for choosing neighbors : $X_1 + X_2 + \ldots + X_N \leq 1$

- Number of points needed is $N!$

$$N! = N^N \exp^{-N} \sqrt{2\pi N}, \quad \text{Sterling's approximation}$$

Estimating the probability

Let's consider N uniform random variables as predictors. Let's say we are interested in predicting the value of y at origin.
Rule for choosing neighbors : $X_1 + X_2 + \ldots + X_N \leq 1$

- Number of points needed is $N!$

$$N! = N^N \exp^{-N} \sqrt{2\pi N}, \quad \text{Sterling's approximation}$$

- Required number of neighbors is a $\mathrm{GIGANTIC}$ number

Duration for a repeat - What's your intuition ?

Let's say that there is a bag with n balls, each ball numbered from 1 to n. You pick a ball and observe the outcome. You put it back. On an average after how many drawings(N) do you see a repeat ?

Let's be more specific. Let n = 1000, After how many drawings do you see a repeat ?

- Bootstrapping
- Boosting
- Random Forests

$N = 1$

$$N = 1 \qquad \frac{n}{n} \times \frac{1}{n}$$

$$N = 1 \qquad\qquad \frac{n}{n} \times \frac{1}{n}$$

$$N = 2$$

$$N = 1 \qquad \frac{n}{n} \times \frac{1}{n}$$

$$N = 2 \qquad \frac{n}{n} \times \frac{n-1}{n} \times \frac{2}{n}$$

$N = 1$ $\qquad \dfrac{n}{n} \times \dfrac{1}{n}$

$N = 2$ $\qquad \dfrac{n}{n} \times \dfrac{n-1}{n} \times \dfrac{2}{n}$

$N = 3$

$$N = 1 \qquad \frac{n}{n} \times \frac{1}{n}$$

$$N = 2 \qquad \frac{n}{n} \times \frac{n-1}{n} \times \frac{2}{n}$$

$$N = 3 \qquad \frac{n}{n} \times \frac{n-1}{n} \times \frac{n-2}{n} \times \frac{3}{n}$$

$$N = 1 \qquad \frac{n}{n} \times \frac{1}{n}$$

$$N = 2 \qquad \frac{n}{n} \times \frac{n-1}{n} \times \frac{2}{n}$$

$$N = 3 \qquad \frac{n}{n} \times \frac{n-1}{n} \times \frac{n-2}{n} \times \frac{3}{n}$$

$$\vdots \qquad \vdots$$

$$\vdots \qquad \vdots$$

```r
duration <- function(n){
  prob    <- c(1/n,sapply(2:n, function(z){
             z*prod(n-(1:(z-1)))/n^z}))
  sum((1:n)*prob)
}
duration1 <- function(n){
  prob <- c(1/n,sapply(2:n, function(z){
            exp(log(z) + sum(log(n-(1:(z-1)))) - z*log(n))
            }))
  sum((1:n)*prob)
}
```

```
duration <- function(n){
  prob    <- c(1/n,sapply(2:n, function(z){
             z*prod(n-(1:(z-1)))/n^z}))
  sum((1:n)*prob)
}
duration1 <- function(n){
  prob <- c(1/n,sapply(2:n, function(z){
            exp(log(z) + sum(log(n-(1:(z-1)))) - z*log(n))
            }))
  sum((1:n)*prob)
}
```

n	# draws for a repeat
10	4
100	12
1000	39
10000	125
100000	396

FIDE 2014 - Probability of a tie, What's your intuition ?

Assume we are at the FIDE 2014 arena. Anand and Carlsen are about to play 12 games. (Win 1 point and draw 0.5 point). Assume the probability of win, lose and draw to be 1/3 each, for both the players.

What's the probability that scores are tied after 12 matches ?

FIDE 2014 - Probability of a tie, What's your intuition ?

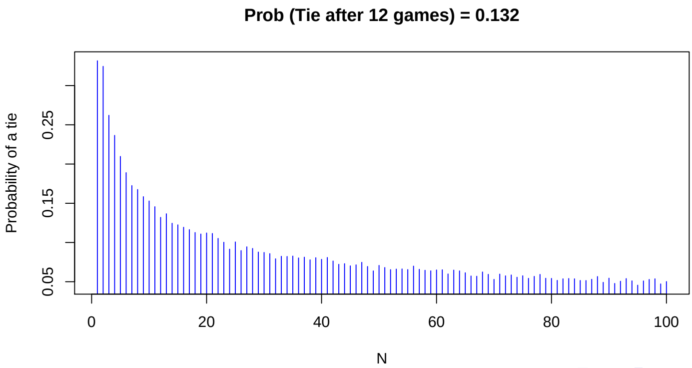
Assume we are at the FIDE 2014 arena. Anand and Carlsen are about to play 12 games. (Win 1 point and draw 0.5 point). Assume the probability of win, lose and draw to be 1/3 each, for both the players.

What's the probability that scores are tied after 12 matches ?

Markov chains ?

```
game <- function(N){
  mean(replicate(10000,sum(sample(c(1,0,-1),N,
                       prob = c(1/3,1/3,1/3),replace= TRUE))==0))
}
N          <- 1:100
tie.prob   <- sapply(N,game)
```

```r
game <- function(N){
  mean(replicate(10000,sum(sample(c(1,0,-1),N,
                  prob = c(1/3,1/3,1/3),replace= TRUE))==0))
}
N          <- 1:100
tie.prob  <- sapply(N,game)
```



- What is relationship between probability of tie and N as $N \to \infty$?

- What is relationship between probability of tie and N as $N \to \infty$?
- One line of `R` code

```
fit  <- lm(log(tie.prob)~log(N))
```

- What is relationship between probability of tie and N as $N \to \infty$?
- One line of `R` code

```
fit  <- lm(log(tie.prob)~log(N))
```

- Estimates

$$\log \hat{p} = -0.8406 - 0.4687 \log N$$
$$\hat{p} = \frac{0.44}{N^{0.47}}$$

- What is relationship between probability of tie and N as $N \to \infty$?
- One line of R code

```
fit  <- lm(log(tie.prob)~log(N))
```

- Estimates

$$\log \hat{p} = -0.8406 - 0.4687 \log N$$
$$\hat{p} = \frac{0.44}{N^{0.47}}$$

- Closed form solution :

$$p = \sqrt{\frac{3}{4\pi N}} = \frac{0.48}{N^{0.5}}$$

- Is it surprising that, as N increases, the probability of a tie goes to 0?

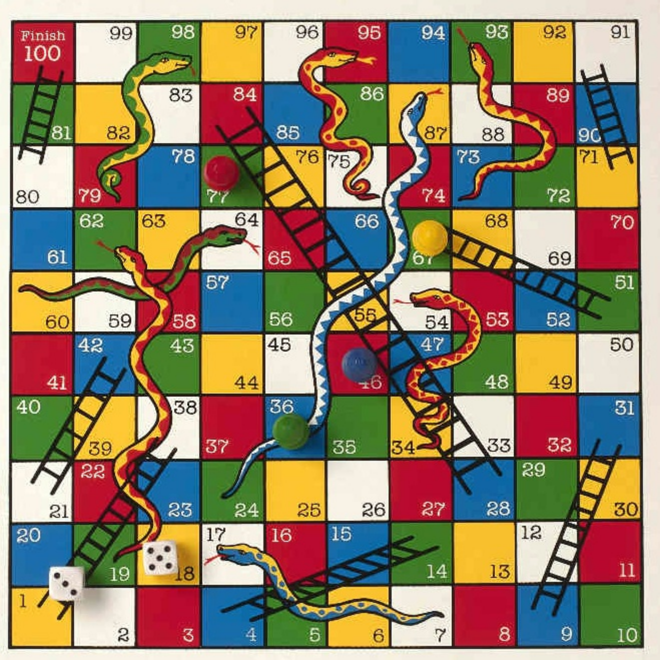
Snakes and Ladders Wager - What's your estimate ?

Imagine that you are a game operator where people come to your shop and play Snakes and Ladders.

This is a one player game.

As a player the goal is to reach 100. For every roll of dice, the player will pay you 1 Rupee. What should be the prize money that you should offer to the players ?

- Set it too high - You will go bankrupt.
- Set it too low - Your clientele will not be motivated to play the game.

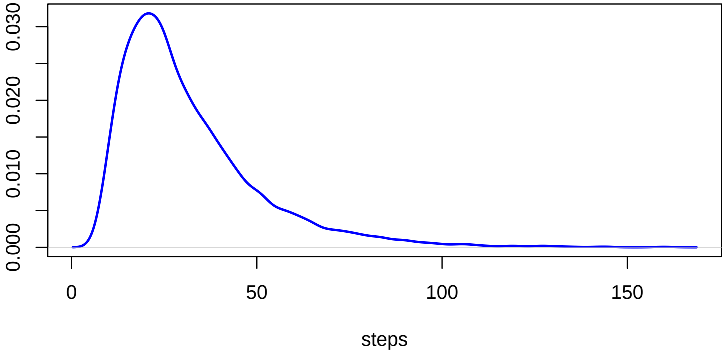

Finish
100
99 98 97 96 95 94 93 92 91
81 82 83 84 85 86 87 88 89 90
80 79 78 77 76 75 74 73 72 71
61 62 63 64 65 66 67 68 69 70
60 59 58 57 56 55 54 53 52 51
41 42 43 44 45 46 47 48 49 50
40 39 38 37 36 35 34 33 32 31
21 22 23 24 25 26 27 28 29 30
20 19 18 17 16 15 14 13 12 11
1 2 3 4 5 6 7 8 9 10

```
starting <- c(1,4,9,17,21,28,51,54,62,64,71,87,93,95,98,80)
ending <- c(38,14,31,7,42,84,67,34,19,60,91,24,73,75,79,100)

play          <- function(){
  count       <- 0
  score       <- 0
  while(score < 100){
    dice      <- sample(1:6,1,replace=TRUE)
    count     <- count + 1
    score     <- score + dice
    if(score %in% starting){
      score <- ending[match(score, starting)]
    }
  }
  return(count)
}
steps <- replicate(10000,play())
```

- Expected number of steps = 32
- Standard deviation for the number of steps = 19
- $\mu \pm \sigma = (13,\ 51)$

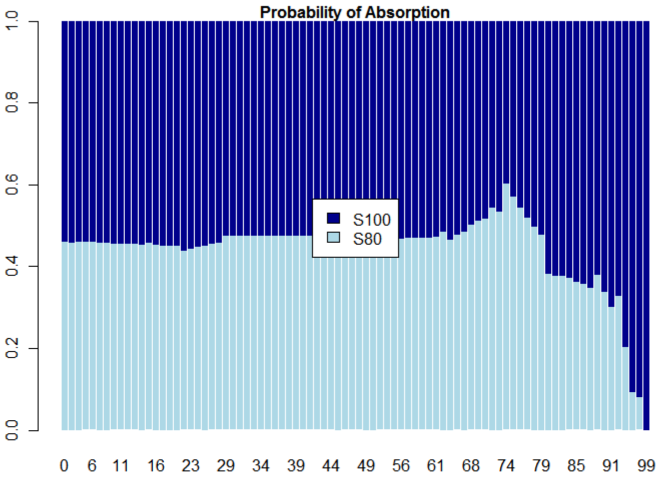
- Expected number of steps = 32
- Standard deviation for the number of steps = 19
- $\mu \pm \sigma = (13, 51)$



Markov chain analysis output - Expected number of steps

100	1		3.9	4.5		7.8		11.6	10.9
20.1	19.1	18.2	17.4	16.4	15.8		13	13.3	12.1
80	16.2	16.2	16	15.6	15	14.1	16.5	16.6	
20.3		17.9		16.1	16.2	16.2	15.9	15.8	15.8
21	21.8	22.7	23	23.8	23.1		24.7	25.2	
26	25.8	25.6	25.5	24.2	24.3	24.4	24.9	24.6	24.5
26.2	26.6	27	27.2	27.5	27.7	28	28.3	28.6	28.9
	27	27.3	27.6	27.8	27.9	28		29.5	29.2
28.2	28.3	28.4		29.3	29.6	30.3	30.6	31	31.4
	32.7	32.1		32.2	32	31.8	31.6		31.4

Markov chain analysis output



God doesn't play dice.

R does.