How many times should you shuffle a deck of cards?

Renato Feres

Abstract

These are notes for the course Math 450 - Random Processes, taught during the spring 07 semester at Washington University, and for a talk given at the Wash. U. undergraduate math club. We look at the subject of random walks on the symmetric group and card shuffling from an "experimental" viewpoint.

1 Permutations

Consider a deck of n cards, which we label by the integers $1, 2, \ldots, n$. Let us agree to represent the order of the virgin unshuffled deck as $12\ldots n$. We call it the *natural order*. A shuffling of the deck can be defined mathematically using the concept of an *action* of the group of permutations of n elements on the orderings of the deck. We begin by explaining what this means.

A permutation of $X = \{1, 2, \ldots, n\}$ is a one-to-one function $\pi : X \rightarrow X$. Any two such functions, π_1, π_2, can be composed and the resulting function, $\pi_2 \circ \pi_1$, is also a permutation. The inverse, π^{-1}, of a permutation is clearly also a permutation. The set of all permutations of X, with the operations of composition and inverse comprises a *group*. This is called the *symmetric group* on n elements and is usually denoted by S_n. We will often represent a permutation π by the list $(\pi(1)\ \pi(2)\ \ldots\ \pi(n))$.

Elements of S_n will act on the orderings of the deck. An ordering of the deck can itself be viewed as a function from X to the set of cards. If the individual cards are named $c_1, c_2, \ldots, c_n$, then an ordering of the deck may be viewed as a function $f : X \rightarrow \{c_1, c_2, \ldots, c_n\}$. This indicates that $f(i)$ is the card that lies in position i on the deck (say, counted from top to bottom).

If π is an element of S_n, then the action of π on an ordering f will be defined as

$$\pi \star f = f \circ \pi^{-1}.$$

This means that a card that is in position i before the rearrangement of the deck is in position $\pi(i)$ after it. Note that $(\pi \star f)(\pi(i)) = f(i)$ so the following two cards are the same:

$$\begin{aligned} \text{card in position } i \text{ before rearrangement} &= f(i) \\ \text{card in position } \pi(i) \text{ after rearrangement} &= (\pi \star f)(\pi(i)). \end{aligned}$$

I should point out that this definition is not quite standard in discussions about card shuffling. One often defines $\pi \star f = f \circ \pi$. The reason I prefer this definition is that it implies the algebraically pleasing property of associativity:

$$(\pi_1 \circ \pi_2) \star f = \pi_1 \star (\pi_2 \star f).$$

On the other hand, notice that it behaves in a somewhat counterintuitive way when we describe the deck rearrangement in terms of the permutation itself. For example, take $n = 3$ and let π be the cyclic permutation represented by the following table:

$$\begin{array}{ccccc} i & = & 1 & 2 & 3 \\ \pi(i) & = & 2 & 3 & 1 \end{array}$$

This permutation represents under the given definition a rearrangement that places the bottom card at the top, and not the other way around. In fact,

$$\text{new top card} = (\pi \star f)(1) = f(\pi^{-1}(1)) = f(3) = \text{old bottom card.}$$

There is no serious reason to use one or the other definition for the action of S_n on rearrangements, and you can choose the one you like better.

2 A note on computer experiments

It can be enlightening to implement our discussion on a computer and do some experimental work. I will give a few hints on how to play with permutations using Matlab.

Let us consider for convenience a deck of only 5 cards. We denote the individual cards (rather than their position on the deck) by a, b, c, d, e. The virgin unshuffled deck will be represented in Matlab by the row vector

```
f0=['a' 'b' 'c' 'd' 'e'];
```

A permutation of the set $\{1, 2, 3, 4, 5\}$ will be represented by a similar row vector. Say, for example,

```
pi=[4 5 1 2 3];
```

is the permutation π such that $\pi(1) = 4$, $\pi(2) = 5$, etc. This represents a cut between the second and third cards followed by placing the bottom group of three cards on top. To perform the operation $\pi \star f_0$ in Matlab, do the following. First invert π using the commando `sort`, which sorts a list of numbers in increasing order. We only need the ordering used by `sort`, which is the second output variable (I will call the first variable `ignore`):

```
[ignore pi_inv]=sort(p);
```

This gives the permutation

```
pi_inv =

     3     4     5     1     2
```

which is the inverse π^{-1}. This is the rearrangement of 4 5 1 2 3 need to bring it back to the standard ordering 1 2 3 4 5, that is, to sort it in increasing order. To obtain $\pi \star f_0$, simply write

```
f=f0(pi_inv);
```

This produces the row vector `cdeab`. I.e., the bottom three cards go to the top. The composition of two permutations $\pi = \pi_1 \circ \pi_2$ is similarly obtained:

```
pi=pi_1(pi_2);
```

3 Random permutations and shuffles

Abstractly, a shuffle is simply a probability distribution on the group S_n. Let us consider some special cases.

3.1 A completely random shuffle

The group S_n has $n!$ elements as is easy to check. (Choose one of n positions in which to place 1, then one of $n-1$ remaining position to place 2, etc.) A completely random shuffle can be defined as assigning equal probability, $1/n!$ to each permutation. In other words, this corresponds to the uniform probability distribution on S_n.

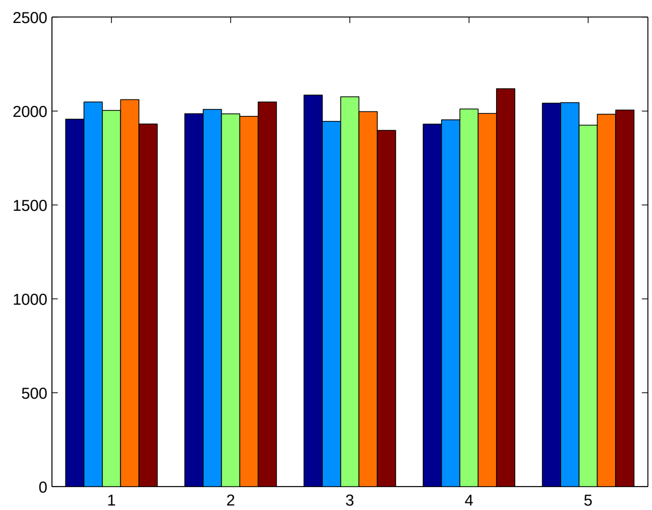

Figure 1: Histograms of the relative frequency for the appearance of each of the numbers $1, 2, 3, 4, 5$ in each of the five positions. This can be interpreted as follows: the first histogram gives the number of occurrences of each number in the first position, etc.

Here is a simple way to simulate a completely random shuffle in Matlab. Choose n random numbers independently in the interval $[0, 1]$. This is done with the command `rand(1,n)`. Then `sort` to bring them to increasing order. The permutation needed to do it is the second output argument of the `sort` command. The following describes a function `rperm(n)` that produces a random permutation of n numbers.

```
function pi=rperm(n)
%Obtains a completely random permutation
%of 1 2 3 ... n
[ignore pi]=sort(rand(1,n));
```

Here is a little experiment on using the function. We draw 10000 permutations of 5 elements according to the uniform probability distribution. Then we draw a histogram of the relative frequency for each of the numbers $1, 2, 3, 4, 5$ in each of the five positions. The following program does this.

```
permutationlist=[];
for i=1:10000
    permutationlist=[permutationlist; rperm(5)];
end
hist(permutationlist, 1:5)
```

3.2 Random cuts

When one *cuts* a deck of cards, one separates the cards in two piles of size k and $n-k$, then puts the bottom k cards on top of the first $n-k$. This rearrangement is produced by the permutation

$$\begin{array}{ccccccccc} i & = & 1 & \dots & k & k+1 & \dots & n \\ \pi_k(i) & = & k+1 & \dots & n & 1 & \dots & k \end{array}$$

A completely random cut may be defined as a probability distribution on S_n that assigns probability $1/n$ to π_k, $k = 1, 2, \dots, n$, and probability 0 to the other permutations. Notice that if $k = n$, the corresponding permutation is trivial, i.e., it does nothing to the deck. (If you don't like this, simply make k range from 1 to $n-1$ with probabilities $1/(n-1)$. Of course, you need at least two cards in the deck for this to make sense.) A random cut can be implemented in Matlab as follows:

```
function pi=rcut(n)
%Obtains a random cut permutation of n cards
k=ceil(n*rand);%this produces a random integer
%between 1 and n with the uniform distribution
pi=[k+1:n 1:k];
```

Note that the set of all cuts of a deck of n cards constitutes a subgroup of S_n. This means that composition and inverse of cut permutations are also cut permutations.

3.3 Random transpositions

A transposition is a permutation that changes the positions of two cards and leaves the remaining cards fixed. It is not too hard to show that the group S_n is generated by transpositions, in the sense that any permutation of S_n can be factors as a product of transpositions.

There are $n(n-1)/2$ (n-choose-2) ways to pick two different numbers in $\{1, 2, \ldots, n\}$. One model of random transposition consists in assigning probability $1/n(n+1)$ to each transposition, and 0 to all the other permutations. Another simpler model may be to accept the same card twice, so that the trivial permutation is also allowed. In this case there $n^2/2$ possibilities, n of which give the trivial permutation. Thus the trivial permutation has probability $1/n$ and each nontrivial transposition has probability $2/n^2$. The following program implements this second model. I leave it as an exercise for you do simulate a random transposition using the first model.

```
function pi=rtransp(m)
%Obtain a random transposition of 1, 2, ..., m
a=sort(ceil(m*rand(1,2)));
pi=[1:a(1)-1 a(2) a(1)+1:a(2)-1 a(1) a(2)+1:m];
```

3.4 Riffle shuffle

Riffle shuffles are a model for what people do when they actually shuffle a deck of cards. A standard riffle shuffle consists in splitting the deck into two piles then interleaving the piles back into a single one.

The particular mathematical model we describe was studied by Gilbert and Shannon, and independently by Reeds. It is sometimes called the GSR shuffle. The definition is as follows. We first cut $(1, 2, \ldots, n)$ into two piles, $(1, 2, \ldots, k)$ and $(k+1, k+2, \ldots, n)$ of approximately equal size. What should this mean? A model that has particularly nice properties is to assume that k has the binomial distribution. Denoting the binomial coefficients by $C(n, k) = n!/k!(n-k)!$, we assume that k is chosen with probability

$$\text{Prob}(k) = C(n, k)/2^n.$$

We illustrate the effect of a binomial cut with a simulation.

To generate values according to the binomial distribution we use the Matlab script:

```
function y=binomial(n,m)
%Simulates drawing m independent realizations
%of a binomial random variable. The value of
%the random variable is the number of heads
%in n tosses of a fair coin.
y=sum(rand(n,m)<=1/2);
```

We now draw a histogram for 10000 random cuts of 52 cards with 52 bins. This can be done using the commands:

```
y=binomial(52,10000);
hist(y,1:52)
```

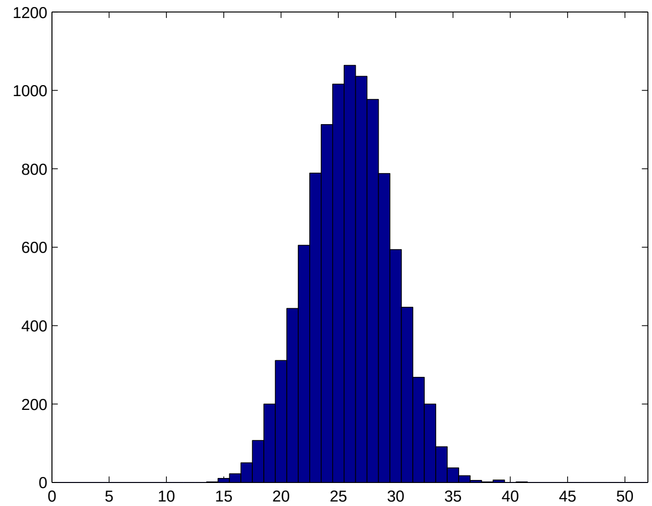

Figure 2: Distribution of position of cut in a deck of 52 cards for 10000 cuts.

Having split the deck into piles of size k and $n-k$, we now interleave them. Notice that when that is done, the order of the first k cards and the order of the second $n-k$ cards are not changed. The operation of interleaving is completely described if we specify k numbers from $\{1, 2, \ldots, n\}$, which are then to be the positions where the first k cards will be inserted to form the new arrangement of the deck. But there are exactly n-choose-k, or $C(n,k)$ ways to do it. We assume that all these possibilities of interleaving are equally likely, so each has probability $1/C(n,k)$.

We can now describe the probability distribution of a riffle shuffle. We do this by looking at the action of the shuffle on a deck with the standard order and counting how many arrangements can arise from a given riffle shuffle. The resulting arrangement of cards will always be a deck that contains at most two increasing sequences. One possibility is that the cards are not mixed at all, so the overall effect of the cut followed by interleaving is the trivial permutation. Let ϵ denote the identity permutation and let $\#_k$ represent the event of having k cards in the first pile. Then

$$\begin{aligned}\text{Prob}(\epsilon) &= \sum_{k=0}^{n} \text{Prob}(\epsilon|\#_k)\text{Prob}(\#_k) \\ &= \sum_{k=0}^{n} \frac{1}{C(n,k)} \times \frac{C(n,k)}{2^n} \\ &= \frac{(n+1)}{2^n}.\end{aligned}$$

Suppose now that π is a permutation other than ϵ that produces exactly two increasing sequences of cards. To find these sequences we pick a card, with label k say, and look above it for the card labeled $k+1$. If we can find it, we add the new card to the list and repeat the procedure with $k+1$ instead of k. We continue in this way until there is no more cards with their successor above them. We now go back to the original card k and reverse the process, looking for the $k-1$ card below the k card, and so on. When this is finished we have one (and therefore also the other) of the two increasing sequences. For example, in the list 6 1 2 7 3 8 4 9 5, starting with $k = 3$ gives the list 1 2 3 4 5, and a second list 6 7 8 9. This means that the cut number k and the interleaving are both determined by the permutation. Therefore, we can now split the probability of a π different from ϵ that may arise from a riffle shuffle, in the following way:

$$\text{Prob}(\pi) = \text{Prob}(\pi|\#_k)\text{Prob}(\#_k) = \frac{1}{C(n,k)} \times \frac{C(n,k)}{2^n} = \frac{1}{2^n},$$

where k is the cut number of π. We conclude that every permutation, other than the trivial, that can arise from a riffle shuffle has the same probability $1/2^n$.

Here are a couple of ways to simulate a riffle shuffle. First consider the program:

```
function pi=ruffle(n)
%Obtains a riffle suffle for a deck of n cards
pi=zeros(1,n);
a=(rand(1,n)<=1/2);
k=sum(a);
d=find(a==1);
e=find(a==0);
pi(d)=1:k;
pi(e)=k+1:n;
```

Although this is very simple, notice that this program does not produce a shuffle with the same probability distribution as in the above model since the cut number k and the interleaving are not independent. The following program should produce a riffle shuffle with the desired probability distribution.

```
function pi=riffleshuffle(n)
%Obtains a riffle shuffle with the
%probability distribution as in Shannon model
k=sum(rand(1,n)<=1/2);
S=1:n;
m=n;
b=[];
for i=1:k
    s=ceil(m*rand);
    b=[b S(s)];
    S=[S(1:s-1) S(s+1:n-i+1)];
    m=m-1;
end
```

```
b=sort(b);
pi=zeros(1,n);
pi(b)=1:k;
a=find(pi==0);
pi(a)=k+1:n;
```

Here is a little experiment we can run. Apply n shuffles to a deck of 52 cards initially in the natural order and look for the frequency of occurrences of a simple arithmetic progression of length 3 at the top. We do the experiment m times for different number n of shuffles and see how the frequency changes as the number of shuffles increases. The next program gives this frequency.

```
function a=progression(n,m)
%Obtain the relative frequency of occurrence of
%a simple arithmetic progression of length three
%at the top of a deck of 52 cards after n riffle
%shuffles of a deck initially in natural order.
a=0;
for i=1:m
    Pi=1:52;
    for i=1:n
        pi=riffleshuffle(52);
        Pi=Pi(pi);
    end
    s=(Pi(2)==Pi(1)+1 & Pi(3)==Pi(1)+2);
    a=a+s;
end
a=a/m; %Proportion showing an arithmetic progression of
      %of length 3 at the top
```

We now run it for different values of n. Note that the probability of a progression of length 3 is approximately $1/52^2 = 3.710^{-4}$, if the deck is fully mixed. For each n, we simulate 10000 runs of the experiment. The next graph shows the result for n up to 7.

3.5 Product of random permutations and convolution

What is the product of two independent random permutations? If P_1 is the probability distribution on S_n of a random permutation Π_1 and P_2 the probability distribution of a random permutation Π_2, we would like to find the probability distribution Q of the product $\Pi_1 \circ \Pi_2$. This is obtained as follows. The probability $Q(\pi)$ of an arbitrary element π in S_n

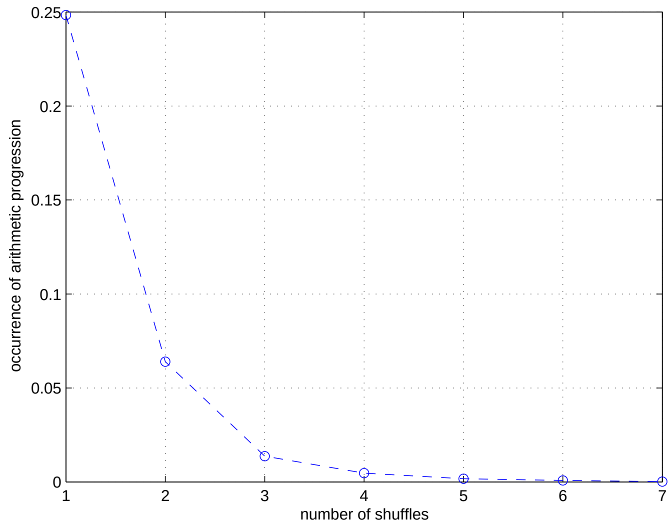

Figure 3: Proportion of occurrence of an arithmetic progression of length 3 at the top of the deck, as a function of the number of shuffles of a deck of 52 cards initially in the natural order.

is

$$\begin{aligned} Q(\pi) &= \text{Prob}(\Pi_1 \circ \Pi_2 = \pi) \\ &= \sum_{\pi_1 \in S_n} \text{Prob}(\Pi_1 = \pi_1 \text{ and } \Pi_2 = \pi_1^{-1} \circ \pi) \\ &= \sum_{\pi_1 \in S_n} \text{Prob}(\Pi_1 = \pi_1)\text{Prob}(\Pi_2 = \pi_1^{-1} \circ \pi) \\ &= \sum_{\pi_1 \in S_n} P_1(\pi_1) P_2(\pi_1^{-1} \circ \pi). \end{aligned}$$

The operation $P_1 * P_2$ defined by

$$(P_1 * P_2)(\pi) = \sum_{\pi_1 \in S_n} P_1(\pi_1) P_2(\pi_1^{-1} \circ \pi)$$

is called the *convolution* of the two probability distributions. If we denote by $P(\Pi)$ the probability distribution of a random permutation Π, then we have just shown that

$$P(\Pi_1 \circ \Pi_2) = P(\Pi_1) * P(\Pi_2).$$

This operation can be applied any number of times. Thus, the product of a random permutation Π with itself N times has probability distribution

$$P(\Pi^N) = P(\Pi) * \cdots * P(\Pi)$$

where the convolution is taken N times.

4 The speed of mixing

Let Π denote the riffle shuffle or, for now, any other random shuffling in S_n. Repeatedly applying Π defines a random walk on S_n. This is a special case of a random walk on a group. You may already be familiar with the abelian version of this process: let U denote a random element in the additive group of integers $\mathbb{Z}$, such that $P(U = 1) = P(U = -1) = 1/2$. Let $U_1, U_2, \ldots$ be independent random variables with the same probability distribution as U. Then $Z_n = U_1 + U_2 + \cdots + U_n$ is a random variable with values in $\mathbb{Z}$ corresponding to the position at time n of the standard random walk on the set of integers.

In the case of a random element of S_n, we wish to consider now the following problem: How fast does the distribution of Π^n (the n-fold product of independent riffle shuffles) converge to the completely random permutation? Recall that the latter is defined as the permutation with the uniform probability $1/n!$ for all elements of S_n.

We need now a measure of how far a given probability distribution is from the uniform one. A standard choice is the *variation distance*. The variation distance between two probability distributions P_1 and P_2 over a set S is defined as

$$\|P_2 - P_1\| = \frac{1}{2} \sum_{\pi \in S} |P_2(\pi) - P_1(\pi)|.$$

The (inessential) factor $1/2$ is included to insure that the resulting distance is between 0 and 1. In fact,

$$\sum_{\pi \in S} |P_2(\pi) - P_1(\pi)| \leq \sum_{\pi \in S} |P_2(\pi)| + \sum_{\pi \in S} |P_1(\pi)| = 2.$$

The variation distance is zero exactly when $P_1 = P_2$, and a distance close to 1 means that the probability distributions on S are significantly different.

We denote by R_n the probability distribution of the nth iteration of a riffle shuffle. Thus

$$R_n = P(\Pi^n) = P(\Pi) * \cdots * P(\Pi).$$

The distance between R_n and the uniform distribution is then

$$d_n = \|R_n - 1/n!\|.$$

In principle, the problem "How many time should we shuffle a deck of cards?" can be solved by calculated d_n for $n = 1, 2, \ldots$ and waiting until the value drops close enough to zero. However, for a standard sized deck of 52 cards, there are 52! elements to be added, and the explicit evaluation of the variation distance in impractical.

4.1 Seven is enough (according to Diaconis)

We briefly describe a clever result due to Persi Diaconis giving the value of d_n. The elementary, but somewhat tricky, proof can be found in [Mann]. For an ordinary deck of cards, we must pick $n = 52$.

Theorem 4.1 *The variation distance d_k between the kth iterate of the riffle shuffle and the uniform probability distribution on S_n is given by*

$$d_k = \frac{1}{2} \sum_{r=1}^{n} A_{n,r} \left| \frac{C(2^k + n - r, n)}{2^{nk}} - \frac{1}{n!} \right|,$$

where the coefficients $A_{n,r}$ are the so-called Eulerian numbers. They can be obtained by the recursive formula: $A_{n,1} = 1$ and

$$A_{n,r} = r^n - \sum_{j=1}^{r-1} C(n + r - j, n) A_{n,j}.$$

Although quite formidable, the formula is well within the reach of computer evaluation. It can be shown that d_k is above 0.9 for $k \leq 5$, then decreases abruptly and is below 0.1 for $k = 10$, quickly approaching zero afterward. A good middle point seems to be $k = 7$, which justifies the claim that 7 shuffles are enough.

References

[Mann] Brad Mann. *How many times should you shuffle a deck of cards?* in Topics in Contemporary Probability and its Applications, pp. 261-289. Ed. by J. Laurie Snell, CRC press, 1995.

[LaCo] Gregory F. Lawler and Lester N. Coyler. *Lectures on Contemporary Probability*, AMS and IAS, 1999.

[CGI] Ke Chen, Peter Giblin, and Alan Irving. Mathematical explorations with Matlab, Cambridge University Press, 1999.