Perfect Shuffle

Oliver de Pug

Diaconis and others have noted that a “random” shuffle may not be random. In fact, they have suggested that if one shuffles a 52 card poker deck “perfectly” eight times, the deck is returned to its original order.

We start by defining a deck of cards as

(new.deck <- 1:52)

## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46  
## [47] 47 48 49 50 51 52

## Create a Perfect Shuffle

Let us start by creating a function that shuffles a deck perfectly.

shuffle <- function(deck, perfect = FALSE){  
 n <- length(deck)  
 if (perfect){  
 deck.shuffled <- rep(0, n)  
 deck.shuffled[2\*(1:(n/2)) - 1] <- deck[1:(n/2)] # righthand (top) cards  
 deck.shuffled[2\*(1:(n/2))] <- deck[(n/2 + 1):n] # lefthand (bottom) cards  
 return(deck.shuffled)   
 }else{  
 return(deck[sample(1:n)])  
 }  
 }

We now check to see that it shuffles correctly.

deck <- new.deck  
 print("Check an imperfect shuffle.")

## [1] "Check an imperfect shuffle."

shuffle(deck, FALSE)

## [1] 37 8 4 25 36 1 47 19 52 31 35 45 30 22 46 16 10 20 51 7 34 26 50  
## [24] 32 14 15 3 48 23 43 29 42 11 44 12 17 21 2 41 9 6 13 28 49 39 18  
## [47] 24 27 38 40 33 5

print("Check a perfect shuffle.")

## [1] "Check a perfect shuffle."

shuffle(deck, TRUE)

## [1] 1 27 2 28 3 29 4 30 5 31 6 32 7 33 8 34 9 35 10 36 11 37 12  
## [24] 38 13 39 14 40 15 41 16 42 17 43 18 44 19 45 20 46 21 47 22 48 23 49  
## [47] 24 50 25 51 26 52

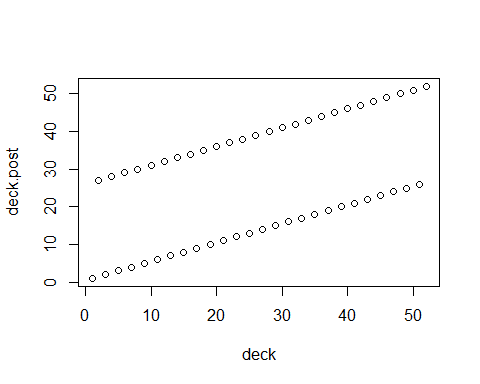
Next, we iterate through eight shuffles to check that we do get back to the original order.

deck.pre <- deck  
 print(deck)

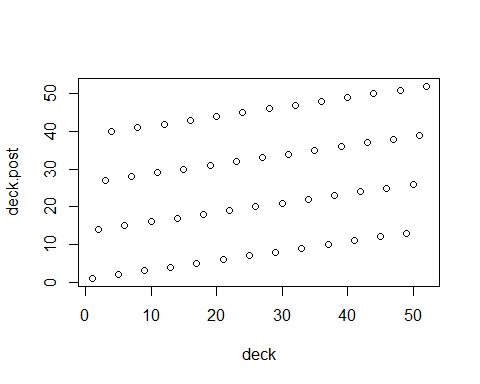
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46  
## [47] 47 48 49 50 51 52

for (i in 1:8){  
 deck.post <- shuffle(deck.pre, perfect = TRUE)  
 print(paste("Shuffles:", i))  
 print(deck.post)  
 plot(deck, deck.post)  
 deck.pre <- deck.post  
 }

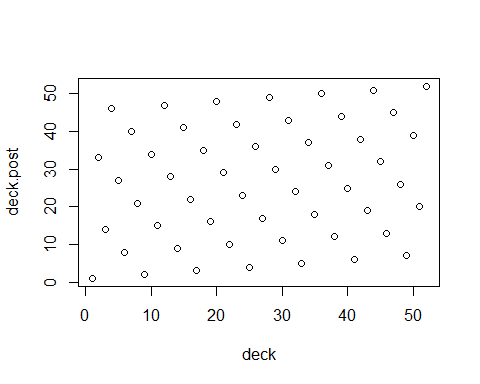
## [1] "Shuffles: 1"  
## [1] 1 27 2 28 3 29 4 30 5 31 6 32 7 33 8 34 9 35 10 36 11 37 12  
## [24] 38 13 39 14 40 15 41 16 42 17 43 18 44 19 45 20 46 21 47 22 48 23 49  
## [47] 24 50 25 51 26 52



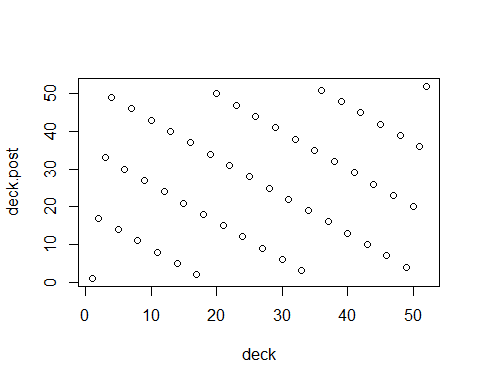
## [1] "Shuffles: 2"  
## [1] 1 14 27 40 2 15 28 41 3 16 29 42 4 17 30 43 5 18 31 44 6 19 32  
## [24] 45 7 20 33 46 8 21 34 47 9 22 35 48 10 23 36 49 11 24 37 50 12 25  
## [47] 38 51 13 26 39 52



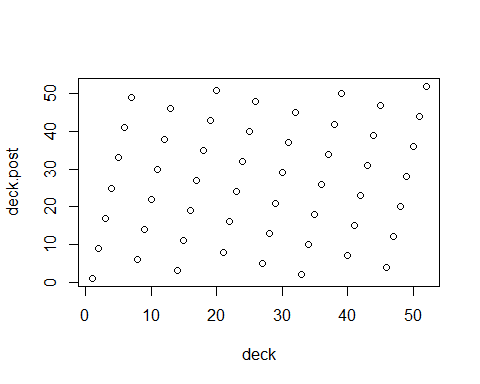
## [1] "Shuffles: 3"  
## [1] 1 33 14 46 27 8 40 21 2 34 15 47 28 9 41 22 3 35 16 48 29 10 42  
## [24] 23 4 36 17 49 30 11 43 24 5 37 18 50 31 12 44 25 6 38 19 51 32 13  
## [47] 45 26 7 39 20 52



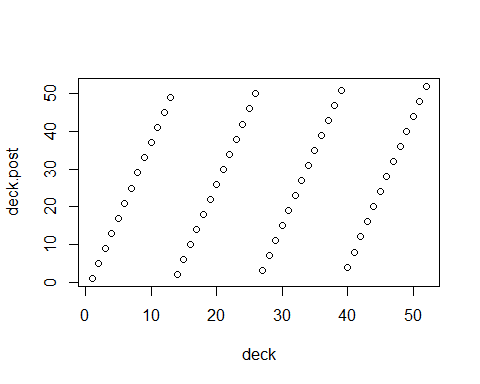
## [1] "Shuffles: 4"  
## [1] 1 17 33 49 14 30 46 11 27 43 8 24 40 5 21 37 2 18 34 50 15 31 47  
## [24] 12 28 44 9 25 41 6 22 38 3 19 35 51 16 32 48 13 29 45 10 26 42 7  
## [47] 23 39 4 20 36 52



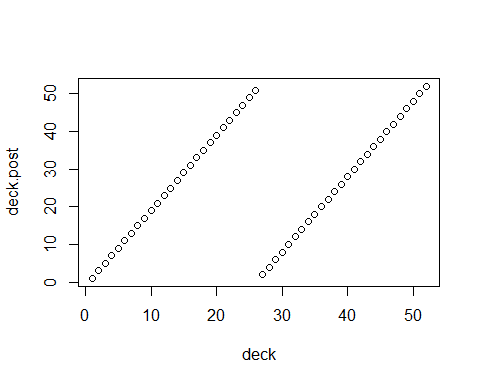
## [1] "Shuffles: 5"  
## [1] 1 9 17 25 33 41 49 6 14 22 30 38 46 3 11 19 27 35 43 51 8 16 24  
## [24] 32 40 48 5 13 21 29 37 45 2 10 18 26 34 42 50 7 15 23 31 39 47 4  
## [47] 12 20 28 36 44 52



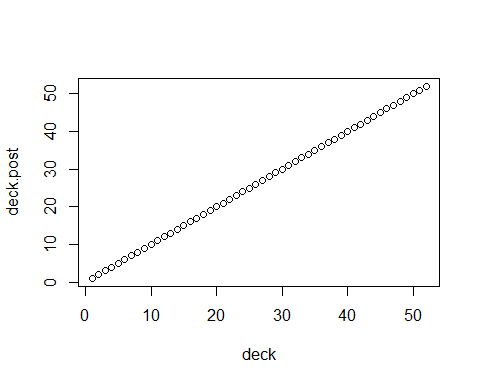
## [1] "Shuffles: 6"  
## [1] 1 5 9 13 17 21 25 29 33 37 41 45 49 2 6 10 14 18 22 26 30 34 38  
## [24] 42 46 50 3 7 11 15 19 23 27 31 35 39 43 47 51 4 8 12 16 20 24 28  
## [47] 32 36 40 44 48 52



## [1] "Shuffles: 7"  
## [1] 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45  
## [24] 47 49 51 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40  
## [47] 42 44 46 48 50 52



## [1] "Shuffles: 8"  
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46  
## [47] 47 48 49 50 51 52



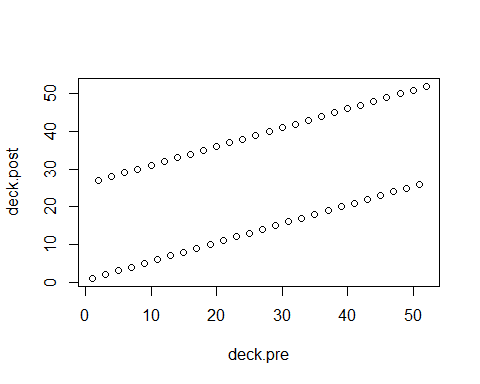
We repeat the above looking at the relationship of the deck to its previous state.

deck.pre <- deck  
 print(deck)

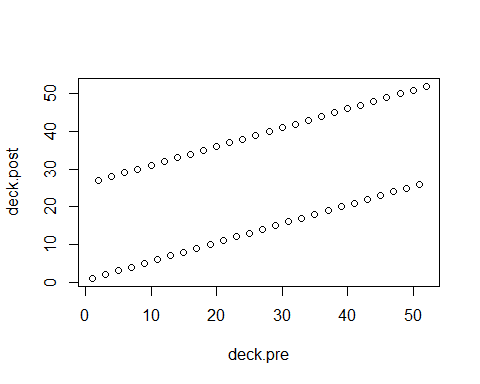
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46  
## [47] 47 48 49 50 51 52

for (i in 1:8){  
 deck.post <- shuffle(deck.pre, perfect = TRUE)  
 print(paste("Shuffles:", i))  
 print(deck.post)  
 plot(deck.pre, deck.post)  
 deck.pre <- deck.post  
 }

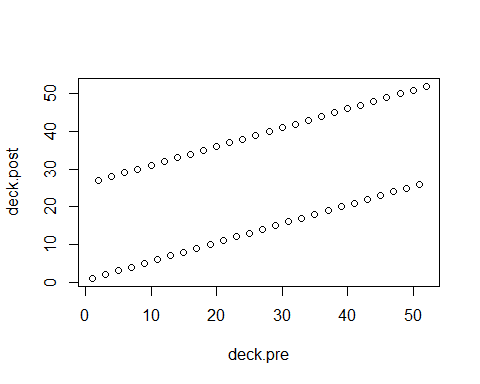
## [1] "Shuffles: 1"  
## [1] 1 27 2 28 3 29 4 30 5 31 6 32 7 33 8 34 9 35 10 36 11 37 12  
## [24] 38 13 39 14 40 15 41 16 42 17 43 18 44 19 45 20 46 21 47 22 48 23 49  
## [47] 24 50 25 51 26 52



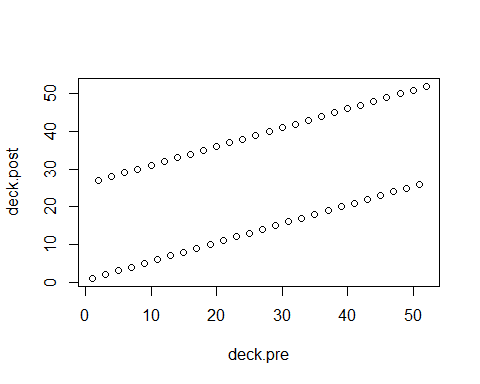
## [1] "Shuffles: 2"  
## [1] 1 14 27 40 2 15 28 41 3 16 29 42 4 17 30 43 5 18 31 44 6 19 32  
## [24] 45 7 20 33 46 8 21 34 47 9 22 35 48 10 23 36 49 11 24 37 50 12 25  
## [47] 38 51 13 26 39 52



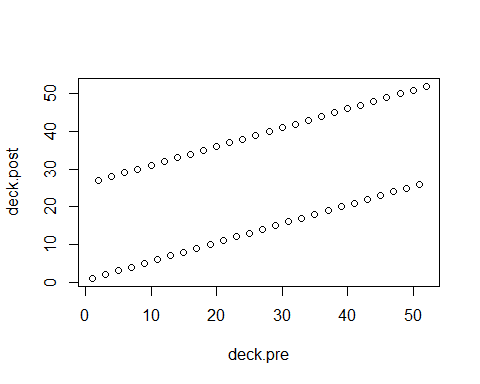
## [1] "Shuffles: 3"  
## [1] 1 33 14 46 27 8 40 21 2 34 15 47 28 9 41 22 3 35 16 48 29 10 42  
## [24] 23 4 36 17 49 30 11 43 24 5 37 18 50 31 12 44 25 6 38 19 51 32 13  
## [47] 45 26 7 39 20 52



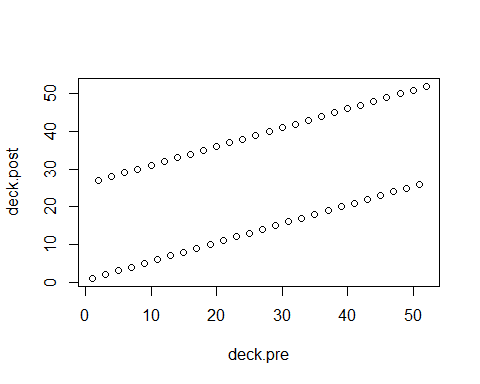
## [1] "Shuffles: 4"  
## [1] 1 17 33 49 14 30 46 11 27 43 8 24 40 5 21 37 2 18 34 50 15 31 47  
## [24] 12 28 44 9 25 41 6 22 38 3 19 35 51 16 32 48 13 29 45 10 26 42 7  
## [47] 23 39 4 20 36 52



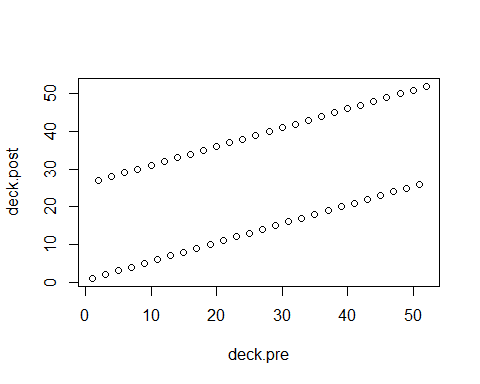
## [1] "Shuffles: 5"  
## [1] 1 9 17 25 33 41 49 6 14 22 30 38 46 3 11 19 27 35 43 51 8 16 24  
## [24] 32 40 48 5 13 21 29 37 45 2 10 18 26 34 42 50 7 15 23 31 39 47 4  
## [47] 12 20 28 36 44 52



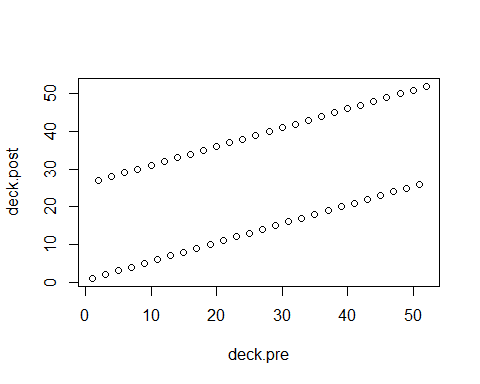
## [1] "Shuffles: 6"  
## [1] 1 5 9 13 17 21 25 29 33 37 41 45 49 2 6 10 14 18 22 26 30 34 38  
## [24] 42 46 50 3 7 11 15 19 23 27 31 35 39 43 47 51 4 8 12 16 20 24 28  
## [47] 32 36 40 44 48 52



## [1] "Shuffles: 7"  
## [1] 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45  
## [24] 47 49 51 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40  
## [47] 42 44 46 48 50 52



## [1] "Shuffles: 8"  
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23  
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46  
## [47] 47 48 49 50 51 52



## How Many Shuffles?

The number of perfect shuffles needed to return a deck to its original state is given by where (mod ). This is the deck’s multiplicative order of .

For the 52 card poker deck used above we have

n <- length(deck)  
  
 k <- 1  
 while (2^k %% (n-1) != 1){ ### Check for multiplicative order of 2 (mod n-1)  
 print(paste(k, 2^k %% (n-1)))  
 k <- k + 1  
 }

## [1] "1 2"  
## [1] "2 4"  
## [1] "3 8"  
## [1] "4 16"  
## [1] "5 32"  
## [1] "6 13"  
## [1] "7 26"

print(paste("To return the deck to its original state, use", k, "shuffles for", n, "cards."))

## [1] "To return the deck to its original state, use 8 shuffles for 52 cards."

For more information see: Weisstein, Eric W. “Out-Shuffle.” From MathWorld–A Wolfram Web Resource. <http://mathworld.wolfram.com/Out-Shuffle.html>