MATLAB probability demos

Contents

Toss a coin

```
x=rand<0.5
```

```
x =

     0
```

Roll a die

```
x=ceil(6*rand)
```

```
x =

     5
```

Roll a pair of dice

```
x=ceil(6*rand)+ceil(6*rand)
```

```
x =
```

```
    2
```

Toss a coin a bunch of times

```
n=10
x=rand(1,n)<.5
```

```
n =

    10


x =

     1     0     1     0     0     1     0     0     1     1
```

Toss a coin a bunch of times; count the number of heads

```
n=10
t=rand(1,n)<.5
x=sum(t)
```

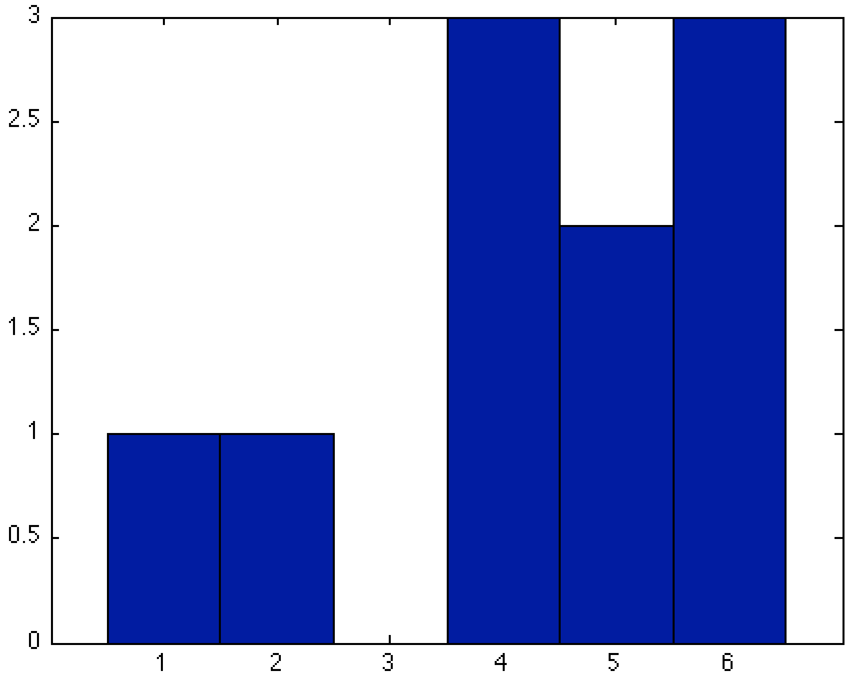
```
n =

    10


t =

     0     0     0     0     1     1     0     0     0     1


x =

     3
```

Roll a die a bunch of times; make a histogram

```
n=10
x=ceil(6*rand(1,n))
hist(x,1:6)
```

```
n =

    10
```

```
x =

     2     4     4     6     1     5     4     5     6     6
```

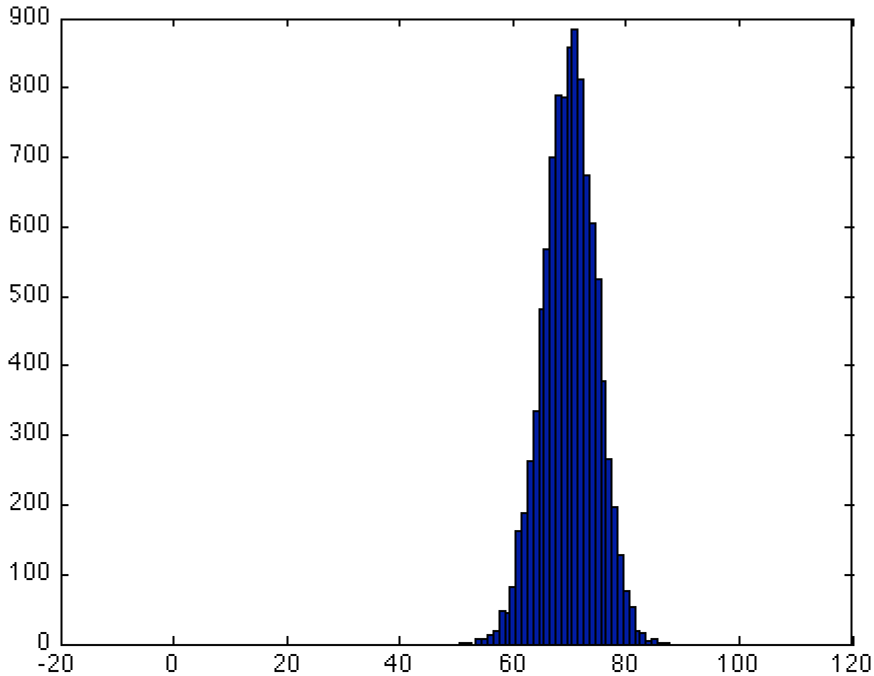


Toss a coin a bunch of times many times

```
clear
p=.7
n=100
trials=10000
x=zeros(1,trials);
for i=1:trials
    t=rand(1,n)<p;
    x(i)=sum(t);
end
hist(x,0:n)
```

```
p =

    0.7000


n =

   100


trials =
```

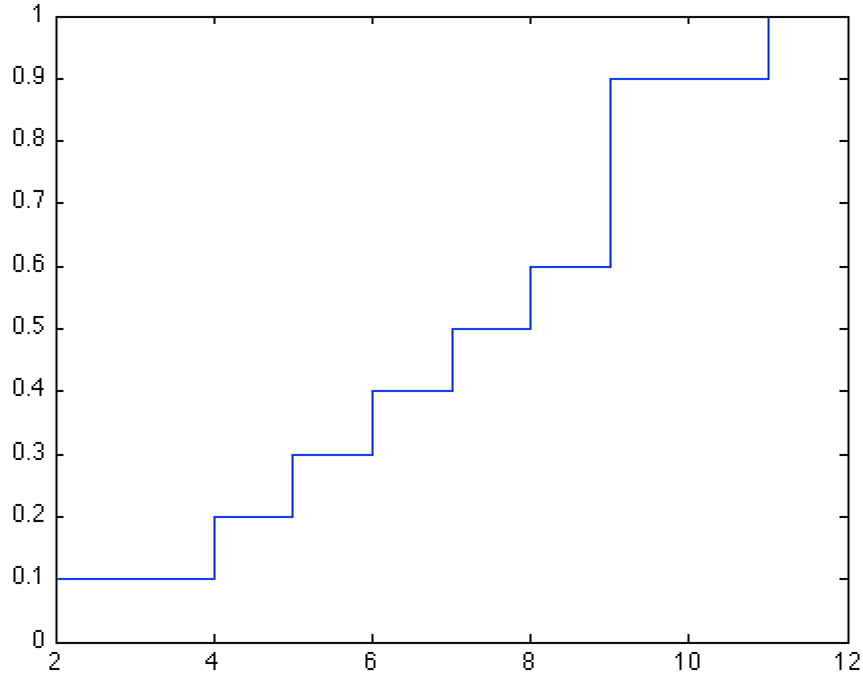
```
    10000
```



Plotting an empirical cdf

```
n=10
x=sum(ceil(6*rand(2,n))) % Roll a pair of dice

stairs([min(x) sort(x)],[0:1/length(x):1]) % Plot the c.d.f of x
```

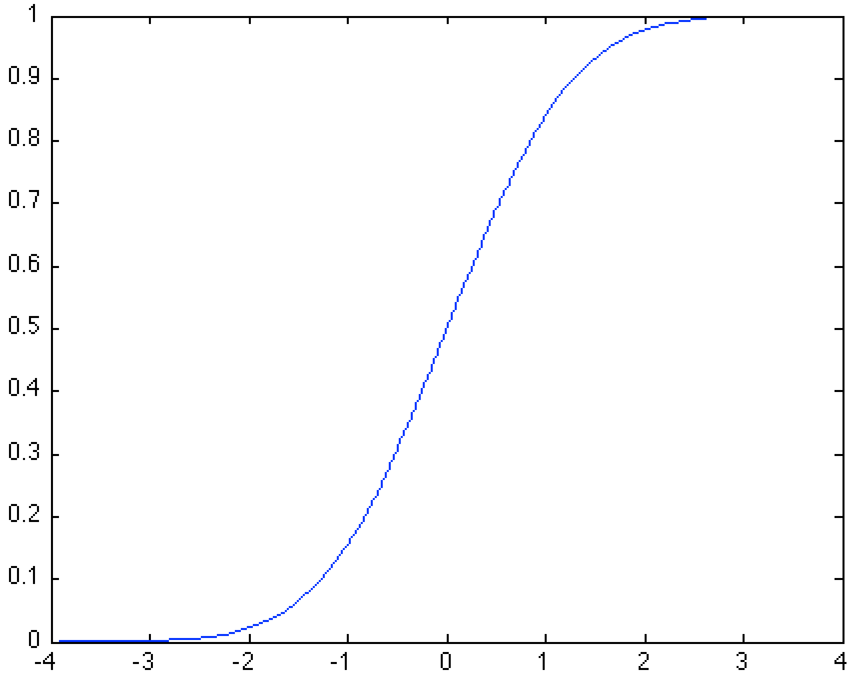
```
n =

    10


x =

     8     9    11     5     2     4     9     7     9     6
```



Normal cdf

```
n=10^4
x=randn(1,n); % Sample from the standard normal distribution.

stairs([min(x) sort(x)],[0:1/length(x):1]) % Plot the c.d.f of x
```

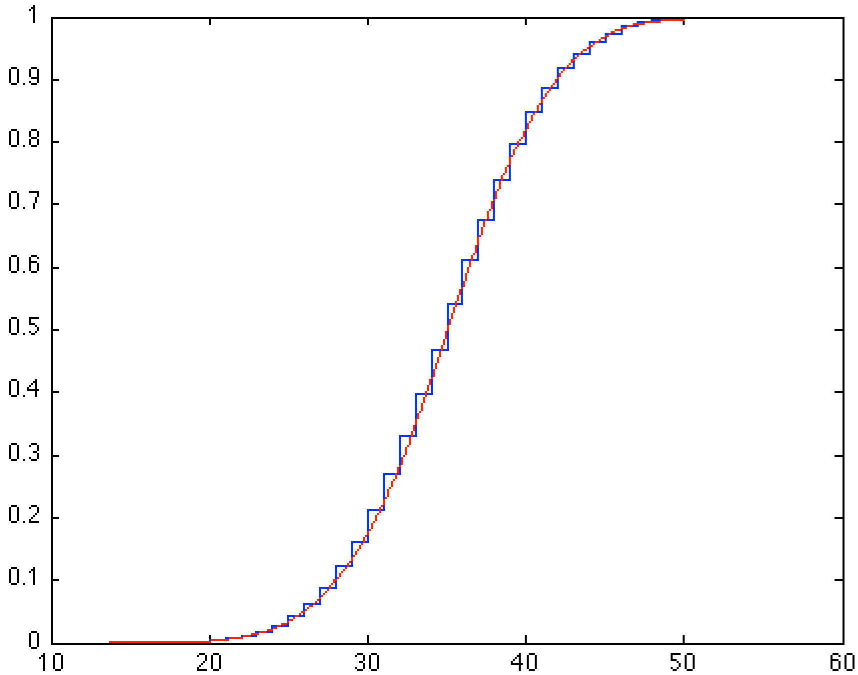
```
n =

       10000
```



The central limit theorem

```
n=10^4;
k=10 % Number of dice to roll
x=sum(ceil(6*rand(k,n))); % Roll k dice
stairs([min(x) sort(x)],[0:1/length(x):1]) % Plot the c.d.f of x

mu1=3.5 % Expected value of 1 die
v1=(sum([1:6].^2)/6-mu1^2) % Variance of 1 die

x1=k*mu1+sqrt(k*v1)*randn(1,n); % Sample the orresdonding normal r.v.

hold on % Superimpose the next plot, in red
stairs([min(x1) sort(x1)],[0:1/length(x1):1],'r')
hold off % End of superposition
```

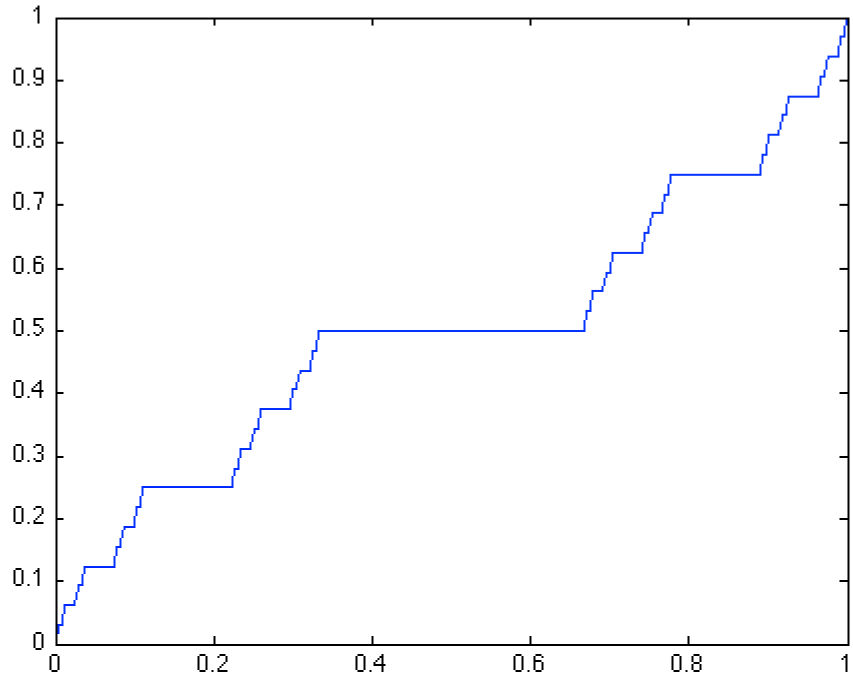
```
k =

    10


mu1 =

    3.5000


v1 =

    2.9167
```



Cantor's devil's staircase as a cumulative distribution function.

```
n=10^6
k=20

x=2*3.^-[1:k]*(rand(k,n)<1/2);

stairs([min(x) sort(x)],0:1/length(x):1) % Plot the c.d.f of x
```

```
n =

     1000000


k =

    20
```



Deal a poker hand.

```
deck=randperm(52);
hand=deck(1:5)
```

```
hand =

    42    15    14    33    49
```

Probabilities of poker hands.

```
straightflush=40
fourofakind=13*48
fullhouse=13*12*4*nchoosek(4,2)
flush=4*nchoosek(13,5)-40
straight=10*4^5-40
threeofakind=13*4*48*44/2
twopair=nchoosek(13,2)*nchoosek(4,2)*nchoosek(4,2)*44
pair=13*nchoosek(4,2)*48*44*40/factorial(3)
squat=nchoosek(13,5)*4^5-straight-flush-straightflush


hands=[straightflush,fourofakind,fullhouse,flush,straight,threeofakind,twopair,pair,
total=sum(hands)
totalshouldbe=nchoosek(52,5)
format long
probabilities=hands/total
format short
```

```
straightflush =

    40

fourofakind =

   624

fullhouse =

        3744

flush =

        5108

straight =

       10200

threeofakind =

       54912

twopair =

      123552

pair =

     1098240

squat =

     1302540

hands =

  Columns 1 through 5

          40          624         3744         5108        10200

  Columns 6 through 9

       54912       123552      1098240      1302540
```

```
total =

     2598960


totalshouldbe =

     2598960


probabilities =

  Columns 1 through 3

   0.000015390771693   0.000240096038415   0.001440576230492

  Columns 4 through 6

   0.001965401545233   0.003924646781790   0.021128451380552

  Columns 7 through 9

   0.047539015606242   0.422569027611044   0.501177394034537
```

Birthday problem.

```
reps=1000
rec=NaN(1,reps);

days=365
people=23
for k=1:reps
    dates=ceil(rand(1,people)*days);
    rec(k)=length(unique(dates));
end
frac=sum(rec<people)/reps
```

```
reps =

        1000


days =

   365


people =

    23
```

```
frac =

    0.5250
```

Coupon collector problem

```
reps=1000
rec=NaN(1,reps);

n=100
m=ceil(n*(log(n)+log(1/log(2))))
for k=1:reps
    coupons=ceil(rand(1,m)*n);
    collection=unique(coupons);
    rec(k)=length(collection);
end
gotall=sum(rec==n)/reps
```

```
reps =

        1000


n =

   100


m =

   498


gotall =

    0.5210
```

Secretary problem.

```
n=100
for k=1:n-1
  s=sum(1./[k:n-1]);
  if s<=1
      break
  end
end
k % pass over the first k-1
reps=10^4
rec=NaN(1,reps);
```

```
for r=1:reps
    a=randperm(n);
    comp=min(a(1:k-1));
    sec=a(n); % Last resort
    for i=k:n-1
        if a(i)<comp
            sec=a(i);
            break
        end
    end

    rec(r)=sec;
end

successrate=sum(rec==1)/reps
```

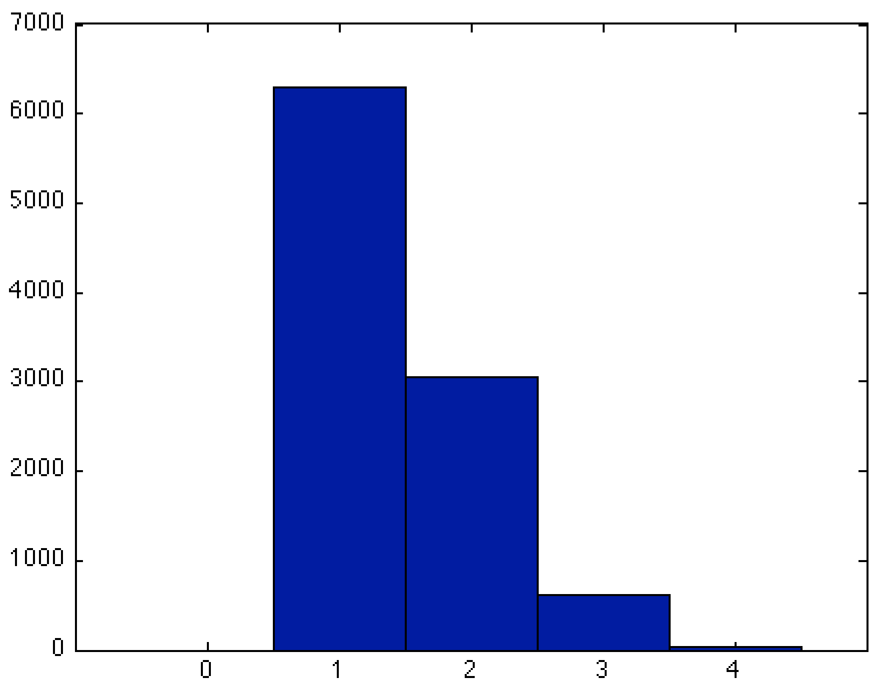
```
n =

   100


k =

    38


reps =

       10000


successrate =

    0.3777
```

Having any ace is good.

```
reps=10^4
rec=NaN(1,reps);

k=0;

while(k<reps)
    hand=sort(randsample(52,13)');
    aces=sum(hand<=4);

    if aces>=1
        k=k+1;
        rec(k)=aces;
    end

end
```

```
hist(rec,0:4)
meansaces=mean(rec)
```

```
reps =

       10000


meansaces =

    1.4374
```

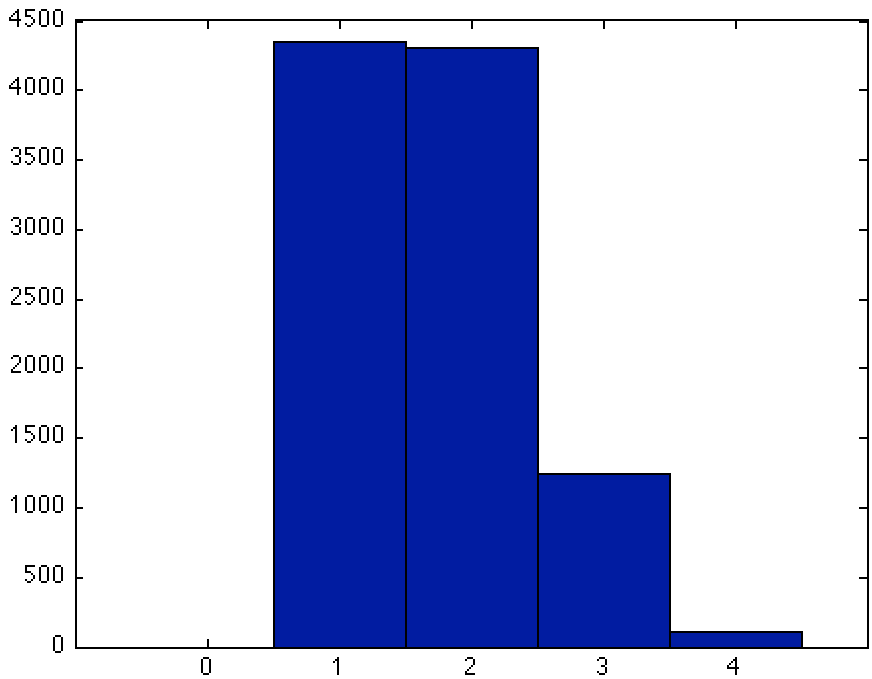


Having the ace of spades is better.

```
reps=10^4
rec=NaN(1,reps);

k=0;

while(k<reps)
    hand=sort(randsample(52,13)');
    aces=sum(hand<=4);

    if hand(1)==1
        k=k+1;
        rec(k)=aces;
    end
```

```
end

hist(rec,0:4)
meanaces=mean(rec)
```

```
reps =

       10000


meanaces =

    1.7102
```



Steady state for a Markov chain

```
p=[1/2,1/4,1/4;1/2,0,1/2;1/4,1/4,1/2]

pinf=p^1000
alpha=pinf(1,:)
```

```
p =

    0.5000    0.2500    0.2500
    0.5000         0    0.5000
    0.2500    0.2500    0.5000
```

```
pinf =

    0.4000    0.2000    0.4000
    0.4000    0.2000    0.4000
    0.4000    0.2000    0.4000


alpha =

    0.4000    0.2000    0.4000
```

Steady state for a periodic Markov chain

```
p=eye(3) % Start with the identity matrix
p=circshift(p,[0 1]) % Shift columns right one click
pinf=p^1000 % Powers don't approach a limit
q=1/2*(p+eye(3)) % Stay put half the time
qinf=q^1000
alpha=qinf(1,:)
```

```
p =

     1     0     0
     0     1     0
     0     0     1


p =

     0     1     0
     0     0     1
     1     0     0


pinf =

     0     1     0
     0     0     1
     1     0     0


q =

    0.5000    0.5000         0
         0    0.5000    0.5000
    0.5000         0    0.5000


qinf =

    0.3333    0.3333    0.3333
    0.3333    0.3333    0.3333
```

```
    0.3333    0.3333    0.3333


alpha =

    0.3333    0.3333    0.3333
```

Published with MATLAB® 7.11