

Plot a table of binomial distributions in R

Asked 11 years, 4 months ago Modified 3 years ago

Viewed 996 times  Part of [R Language](#) Collective



0



For a game design issue, I need to better inspect binomial distributions. Using **R**, I need to build a two dimensional table that - given a fixed parameters 'pool' (the number of dice rolled), 'sides' (the number of sides of the die) has:

- In rows --> minimum for a success (ranging from 0 to sides, it's a discrete distribution)
- In columns --> number of successes (ranging from 0 to pool)

I know how to calculate it as a single task, but I'm not sure on how to iterate to fill the entire table

EDIT: I forgot to say that I want to calculate the probability p of gaining at least the number of successes.

R **r** **dice**

Share

Improve this question

Follow

edited May 2, 2017 at 14:34



Brian Tompsett - 汤莱恩

5,875 ● 72 ● 61 ● 133

asked Jul 31, 2013 at 15:22



Gabriele B

2,685 ● 1 ● 28 ● 44

It looks like this is a programming issue. If so, this would be better served on Stack Overflow, and a moderator can migrate this question for you (flag this post, but don't repost on SO). – [chl](#) Jul 31, 2013 at 15:37

1 Show us what you've tried. – [Thomas](#) Jul 31, 2013 at 16:02

one way is to simply keep going until you have rolled all values at least once – [Ricardo Saporta](#) Jul 31, 2013 at 16:41

2 Answers

Sorted by:

Highest score (default)





0

Ok, i think this could be a simple solution. It has ratio of successes on rows and success thresholds on dice roll (p) on columns.



```
poolDistribution <- function(n, sides=10, digits=2, ro
  m <- 1:sides
  names(m) <- paste(m,ifelse(roll.Under,"-", "+"),sep=
  s <- 1:n
  names(s) <- paste(s,n,sep="/")
  sapply(m, function(m.value) round((if(roll.Under) (1
  (m.value)/sides))*100 else (1 - pbinom(s - 1, n, (side
  1)/sides))*100), digits=digits))
```

Share Improve this answer

Follow

answered Aug 1, 2013 at 14:35



Gabriele B

2,685 ● 1 ● 28 ● 44



0

This gets you half of the way.

If you are new to **R**, you might miss out on the fact that a very powerful feature is that you can use a vector of values as an index to another vector. This makes part of the problem trivially easy:



```
pool <- 3
sides <- 20 # <cough>D&D<cough>

# you need to store the values somewhere, use a vector
NumberOfRollsPerSide <- rep(0, sides)
names(NumberOfRollsPerSide) <- 1:sides # this will be

## Repeat so long as there are still zeros
##      ie, so long as there is a side that has not com
```

```
while (any(NumberOfRollsPerSide == 0)) {  
  # roll once  
  oneRoll <- sample(1:sides, pool, TRUE)  
  
  # add (+1) to each sides' total rolls  
  # note that you can use the roll outcome to index t  
  NumberOfRollsPerSide[oneRoll] <- NumberOfRollsPerSide + 1  
}  
  
# These are your results:  
NumberOfRollsPerSide
```

All you have left to do now is count, for each side, in which roll number it first came up.

Share Improve this answer

Follow

edited Dec 9, 2021 at 5:59



Nimantha

6,438 ● 6 ● 30 ● 75

answered Jul 31, 2013 at 16:49



Ricardo Saporta

55.3k ● 17 ● 148 ● 179

ehehe tnx, it's not my homework. Unfortunately, I'm too old to be at college ;) We just decided to spice up our RPG *cough*non D20-based*cough* design activities with more statistical rigor. I'm not totally new to R, but I usually use it for analysis rather than real programming. But lemme try your code now! :) tnx for your answer! – [Gabriele B](#) Aug 1, 2013 at 7:06

while this is not what i need to calculate (I edited the question accordingly), your snippet of code is still purely useful for something else i need to analyze! Tnx allot for your contribute! – [Gabriele B](#) Aug 1, 2013 at 7:19 ✎

Surely if it was D&D, it would be 3d6 *AHEM* – [Hong Ooi](#) Aug
1, 2013 at 7:37
