Probabilistic processes

Tad Dallas

Contents

Logistic model revisited

```
logisticGrowth <- function(n, r, k){
  n*exp(r*(1-(n / k)))
}


logisticDynamics <- function(n,r,k, steps=100){
  ret <- c()
  ret[1] <- n
  if(length(r) == 1){
    r <- rep(r, steps)
  }
  for(i in 1:(steps-1)){
    ret[i+1] <- logisticGrowth(ret[i], r[i], k)
  }
  return(ret)
}
```

But this is not how populations change over time, right?

Why not?

- No individual variation in birth rates
- No temporal changes to carrying capacity or growth rate

but perhaps most importantly . . .

Birth and death are probabilistic processes

This process is often called 'demographic stochasticity', and is especially important when thinking about small population sizes. As a thought experiment, imagine flipping a fair coin 5 times. The probability of landing on 'heads' is 0.5, but with only 5 trials, the resulting number of heads is far more variable than if we flipped that same coin 100 times. Considering birth and death as probabilistic processes, we can start to understand how we might expect population dynamics to be more variable at small population sizes. That is, the effects of 'demographic stochasticity' are dependent on population density.

What if we treated birth as offspring drawn from a Poisson distribution, with some mean number of offspring λ?

```
logisticP1 <- function(Nt, b=0.1, d=0.1) {
  births <- sum(rbinom(Nt,1,b))
  deaths <- 0
  pop <- (Nt + births - deaths)
  return(pop)
}
```

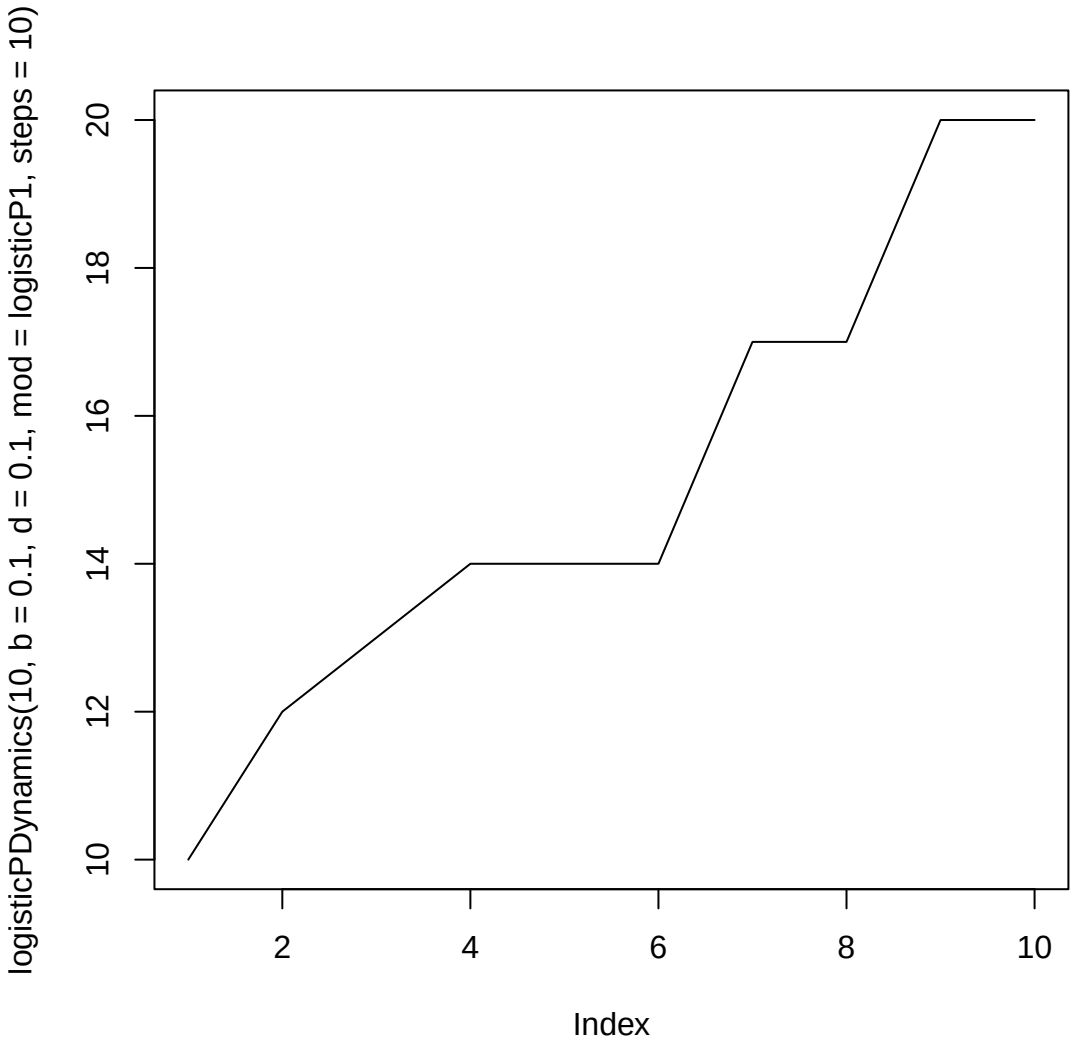
So the model above treats birth as drawn from a Poisson distribution, and death as dependent on the population density.

What if we treated death as binomial, with some probability p?

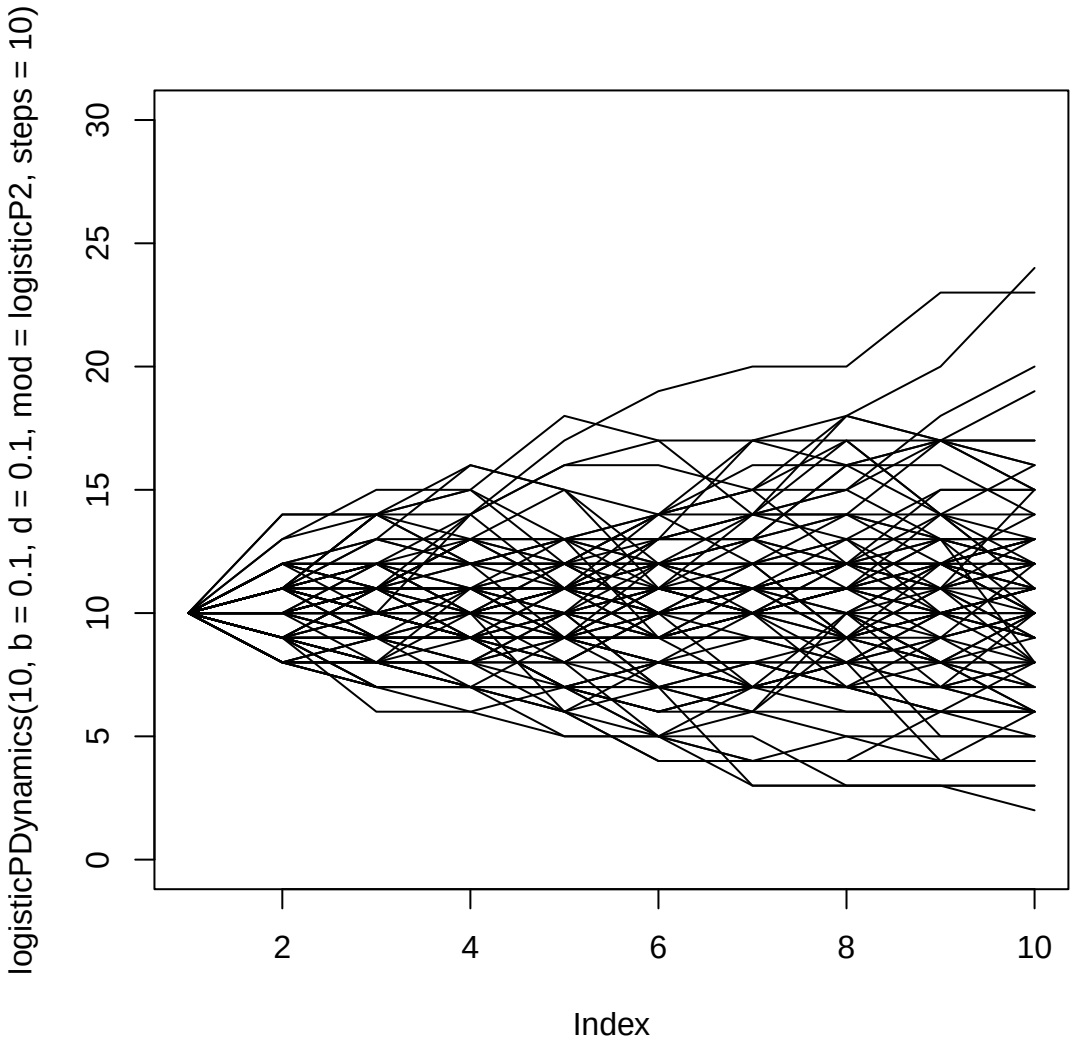
```
logisticP2 <- function(Nt, b=0.1, d=0.1) {
  births <- sum(rbinom(Nt, 1, b))
  deaths <- sum(rbinom(Nt, 1, d))
  pop <- (Nt + births - deaths)
  return(pop)
}
```

```
logisticPDynamics <- function(Nt, b, d, steps=1000, mod=logisticP1){
  ret <- c()
  ret[1] <- Nt
  if(length(b) == 1){
    b <- rep(b, steps)
  }
  if(length(d) == 1){
    d <- rep(d, steps)
  }
  for(i in 1:(steps-1)){
    ret[i+1] <- mod(ret[i], b=b[i], d=d[i])
  }
  return(ret)
}


plot(logisticPDynamics(10, b=0.1, d=0.1, mod=logisticP1, steps=10), type='l')
```




```
# birth death equal
plot(logisticPDynamics(10, b=0.1, d=0.1, mod=logisticP2, steps=10), type='l', ylim=c(0,30))
lapply(1:100, function(x){
  lines(logisticPDynamics(10, b=0.1, d=0.1, mod=logisticP2, steps=10))
})
```




```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
```

```
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## NULL
##
## [[18]]
## NULL
##
## [[19]]
## NULL
##
## [[20]]
## NULL
##
## [[21]]
## NULL
##
## [[22]]
## NULL
##
## [[23]]
## NULL
##
## [[24]]
## NULL
```

```
##
## [[25]]
## NULL
##
## [[26]]
## NULL
##
## [[27]]
## NULL
##
## [[28]]
## NULL
##
## [[29]]
## NULL
##
## [[30]]
## NULL
##
## [[31]]
## NULL
##
## [[32]]
## NULL
##
## [[33]]
## NULL
##
## [[34]]
## NULL
##
## [[35]]
## NULL
##
## [[36]]
## NULL
##
## [[37]]
## NULL
##
## [[38]]
## NULL
##
## [[39]]
## NULL
##
## [[40]]
## NULL
##
## [[41]]
## NULL
##
## [[42]]
## NULL
```

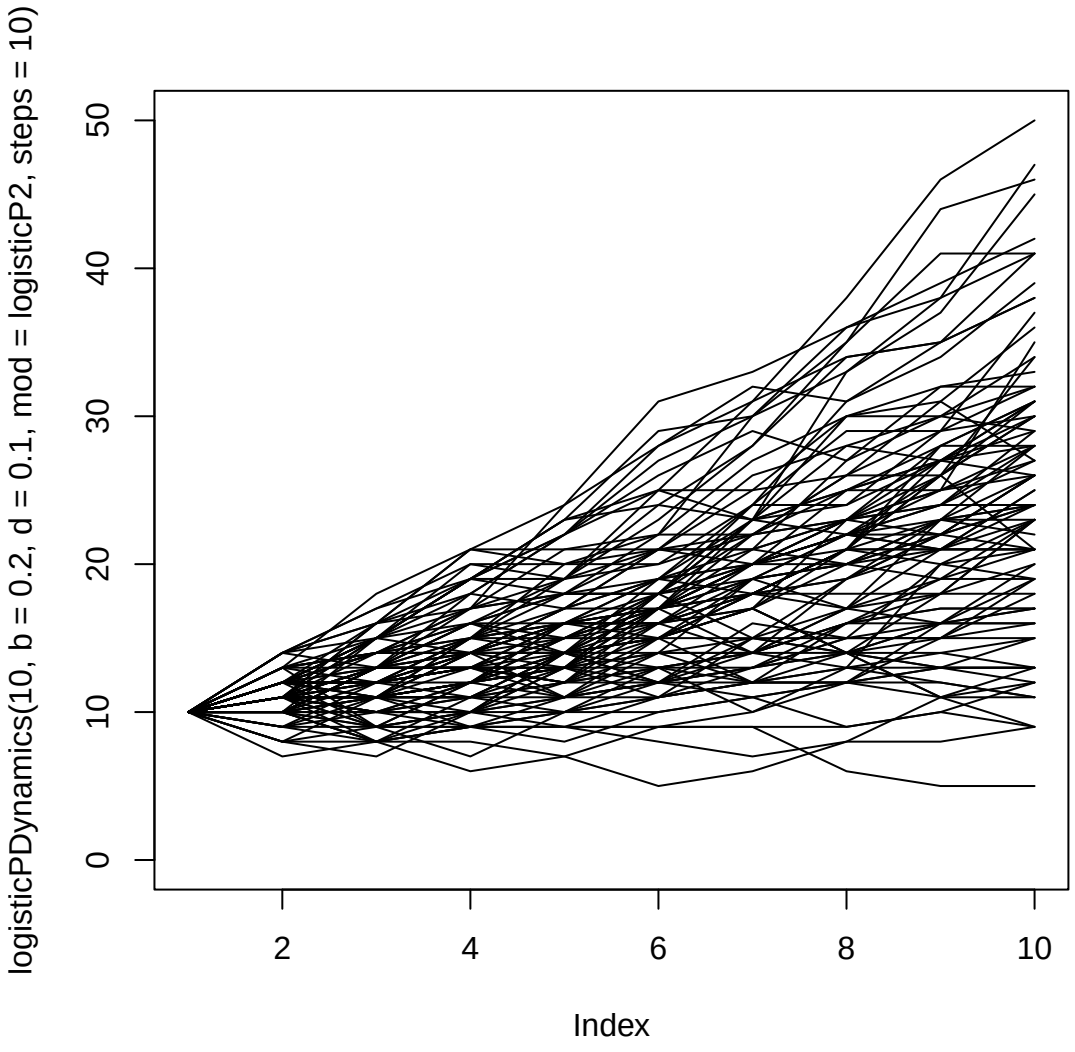
```
##
## [[43]]
## NULL
##
## [[44]]
## NULL
##
## [[45]]
## NULL
##
## [[46]]
## NULL
##
## [[47]]
## NULL
##
## [[48]]
## NULL
##
## [[49]]
## NULL
##
## [[50]]
## NULL
##
## [[51]]
## NULL
##
## [[52]]
## NULL
##
## [[53]]
## NULL
##
## [[54]]
## NULL
##
## [[55]]
## NULL
##
## [[56]]
## NULL
##
## [[57]]
## NULL
##
## [[58]]
## NULL
##
## [[59]]
## NULL
##
## [[60]]
## NULL
```

```
##
## [[61]]
## NULL
##
## [[62]]
## NULL
##
## [[63]]
## NULL
##
## [[64]]
## NULL
##
## [[65]]
## NULL
##
## [[66]]
## NULL
##
## [[67]]
## NULL
##
## [[68]]
## NULL
##
## [[69]]
## NULL
##
## [[70]]
## NULL
##
## [[71]]
## NULL
##
## [[72]]
## NULL
##
## [[73]]
## NULL
##
## [[74]]
## NULL
##
## [[75]]
## NULL
##
## [[76]]
## NULL
##
## [[77]]
## NULL
##
## [[78]]
## NULL
```

```
##
## [[79]]
## NULL
##
## [[80]]
## NULL
##
## [[81]]
## NULL
##
## [[82]]
## NULL
##
## [[83]]
## NULL
##
## [[84]]
## NULL
##
## [[85]]
## NULL
##
## [[86]]
## NULL
##
## [[87]]
## NULL
##
## [[88]]
## NULL
##
## [[89]]
## NULL
##
## [[90]]
## NULL
##
## [[91]]
## NULL
##
## [[92]]
## NULL
##
## [[93]]
## NULL
##
## [[94]]
## NULL
##
## [[95]]
## NULL
##
## [[96]]
## NULL
```

```
##
## [[97]]
## NULL
##
## [[98]]
## NULL
##
## [[99]]
## NULL
##
## [[100]]
## NULL
```

```
# birth 2x death
plot(logisticPDynamics(10, b=0.2, d=0.1, mod=logisticP2, steps=10), type='l', ylim=c(0,50))
lapply(1:100, function(x){
  lines(logisticPDynamics(10, b=0.2, d=0.1, mod=logisticP2, steps=10))
})
```



```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## NULL
##
## [[18]]
## NULL
##
```

```
## [[19]]
## NULL
##
## [[20]]
## NULL
##
## [[21]]
## NULL
##
## [[22]]
## NULL
##
## [[23]]
## NULL
##
## [[24]]
## NULL
##
## [[25]]
## NULL
##
## [[26]]
## NULL
##
## [[27]]
## NULL
##
## [[28]]
## NULL
##
## [[29]]
## NULL
##
## [[30]]
## NULL
##
## [[31]]
## NULL
##
## [[32]]
## NULL
##
## [[33]]
## NULL
##
## [[34]]
## NULL
##
## [[35]]
## NULL
##
## [[36]]
## NULL
##
```

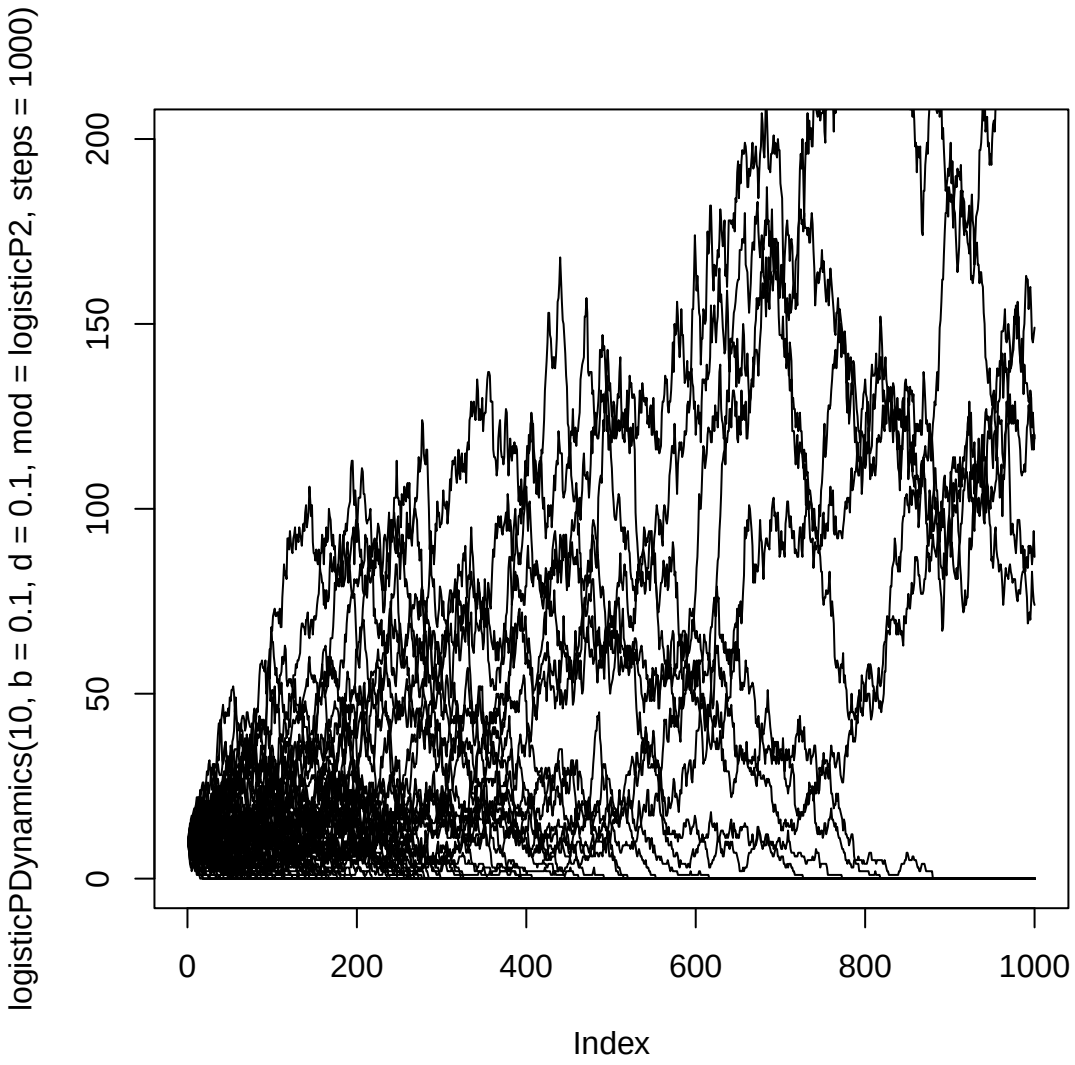
```
## [[37]]
## NULL
##
## [[38]]
## NULL
##
## [[39]]
## NULL
##
## [[40]]
## NULL
##
## [[41]]
## NULL
##
## [[42]]
## NULL
##
## [[43]]
## NULL
##
## [[44]]
## NULL
##
## [[45]]
## NULL
##
## [[46]]
## NULL
##
## [[47]]
## NULL
##
## [[48]]
## NULL
##
## [[49]]
## NULL
##
## [[50]]
## NULL
##
## [[51]]
## NULL
##
## [[52]]
## NULL
##
## [[53]]
## NULL
##
## [[54]]
## NULL
##
```

```
## [[55]]
## NULL
##
## [[56]]
## NULL
##
## [[57]]
## NULL
##
## [[58]]
## NULL
##
## [[59]]
## NULL
##
## [[60]]
## NULL
##
## [[61]]
## NULL
##
## [[62]]
## NULL
##
## [[63]]
## NULL
##
## [[64]]
## NULL
##
## [[65]]
## NULL
##
## [[66]]
## NULL
##
## [[67]]
## NULL
##
## [[68]]
## NULL
##
## [[69]]
## NULL
##
## [[70]]
## NULL
##
## [[71]]
## NULL
##
## [[72]]
## NULL
##
```

```
## [[73]]
## NULL
##
## [[74]]
## NULL
##
## [[75]]
## NULL
##
## [[76]]
## NULL
##
## [[77]]
## NULL
##
## [[78]]
## NULL
##
## [[79]]
## NULL
##
## [[80]]
## NULL
##
## [[81]]
## NULL
##
## [[82]]
## NULL
##
## [[83]]
## NULL
##
## [[84]]
## NULL
##
## [[85]]
## NULL
##
## [[86]]
## NULL
##
## [[87]]
## NULL
##
## [[88]]
## NULL
##
## [[89]]
## NULL
##
## [[90]]
## NULL
##
```

```
## [[91]]
## NULL
##
## [[92]]
## NULL
##
## [[93]]
## NULL
##
## [[94]]
## NULL
##
## [[95]]
## NULL
##
## [[96]]
## NULL
##
## [[97]]
## NULL
##
## [[98]]
## NULL
##
## [[99]]
## NULL
##
## [[100]]
## NULL
```

```
# 1000 time steps
plot(logisticPDynamics(10, b=0.1, d=0.1, mod=logisticP2, steps=1000), type='l', ylim=c(0,200))
lapply(1:100, function(x){
  lines(logisticPDynamics(10, b=0.1, d=0.1, mod=logisticP2, steps=1000))
})
```




```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
```

```
##
## [[7]]
## NULL
##
## [[8]]
## NULL
##
## [[9]]
## NULL
##
## [[10]]
## NULL
##
## [[11]]
## NULL
##
## [[12]]
## NULL
##
## [[13]]
## NULL
##
## [[14]]
## NULL
##
## [[15]]
## NULL
##
## [[16]]
## NULL
##
## [[17]]
## NULL
##
## [[18]]
## NULL
##
## [[19]]
## NULL
##
## [[20]]
## NULL
##
## [[21]]
## NULL
##
## [[22]]
## NULL
##
## [[23]]
## NULL
##
## [[24]]
## NULL
```

```
##
## [[25]]
## NULL
##
## [[26]]
## NULL
##
## [[27]]
## NULL
##
## [[28]]
## NULL
##
## [[29]]
## NULL
##
## [[30]]
## NULL
##
## [[31]]
## NULL
##
## [[32]]
## NULL
##
## [[33]]
## NULL
##
## [[34]]
## NULL
##
## [[35]]
## NULL
##
## [[36]]
## NULL
##
## [[37]]
## NULL
##
## [[38]]
## NULL
##
## [[39]]
## NULL
##
## [[40]]
## NULL
##
## [[41]]
## NULL
##
## [[42]]
## NULL
```

```
##
## [[43]]
## NULL
##
## [[44]]
## NULL
##
## [[45]]
## NULL
##
## [[46]]
## NULL
##
## [[47]]
## NULL
##
## [[48]]
## NULL
##
## [[49]]
## NULL
##
## [[50]]
## NULL
##
## [[51]]
## NULL
##
## [[52]]
## NULL
##
## [[53]]
## NULL
##
## [[54]]
## NULL
##
## [[55]]
## NULL
##
## [[56]]
## NULL
##
## [[57]]
## NULL
##
## [[58]]
## NULL
##
## [[59]]
## NULL
##
## [[60]]
## NULL
```

```
##
## [[61]]
## NULL
##
## [[62]]
## NULL
##
## [[63]]
## NULL
##
## [[64]]
## NULL
##
## [[65]]
## NULL
##
## [[66]]
## NULL
##
## [[67]]
## NULL
##
## [[68]]
## NULL
##
## [[69]]
## NULL
##
## [[70]]
## NULL
##
## [[71]]
## NULL
##
## [[72]]
## NULL
##
## [[73]]
## NULL
##
## [[74]]
## NULL
##
## [[75]]
## NULL
##
## [[76]]
## NULL
##
## [[77]]
## NULL
##
## [[78]]
## NULL
```

```
##
## [[79]]
## NULL
##
## [[80]]
## NULL
##
## [[81]]
## NULL
##
## [[82]]
## NULL
##
## [[83]]
## NULL
##
## [[84]]
## NULL
##
## [[85]]
## NULL
##
## [[86]]
## NULL
##
## [[87]]
## NULL
##
## [[88]]
## NULL
##
## [[89]]
## NULL
##
## [[90]]
## NULL
##
## [[91]]
## NULL
##
## [[92]]
## NULL
##
## [[93]]
## NULL
##
## [[94]]
## NULL
##
## [[95]]
## NULL
##
## [[96]]
## NULL
```

```
##
## [[97]]
## NULL
##
## [[98]]
## NULL
##
## [[99]]
## NULL
##
## [[100]]
## NULL
```

But the above incorporation of probabilistic processes is based on phenomenological modeling, where many of us might also want to do some statistical modeling. Now we will go over probability distributions more generally in R, and how they relate to common statistical tests.

notation in R

"d" returns the height of the probability density function "p" returns the cumulative density function "q" returns the inverse cumulative density function (quantiles) "r" returns randomly generated numbers

Generate random draws from probability distributions

```
rnorm(100, 1, 1)
```

```
##   [1]  1.11621562  0.30402896  0.63965779  1.65466690  0.31072951  3.28798389
##   [7]  0.51965724  1.60634116  2.45291355  0.72205703  1.10917677  0.84690608
##  [13] -0.46201883  1.69012579  1.02126142 -1.40337412  1.10310250  3.05456244
##  [19]  1.39182874  2.49703739 -0.82751807  0.90018018  1.23292708  0.81809788
##  [25]  1.39089631 -1.02434614  1.45570625 -0.23019151 -0.09341636  0.95089135
##  [31]  0.51819842  1.77459557  1.97215153  1.64891275  1.54971694  1.34298239
##  [37]  1.42866213  1.23499736 -0.51665809  0.60023476 -0.32240519  2.16536038
##  [43]  0.71384750  0.84383543 -0.61401874  1.19809736  1.49981257  1.35895214
##  [49]  2.02093677  1.60575691  1.38755042  1.40847234  1.53914771  0.29767100
##  [55]  1.67384105  1.54085772  1.67394885  0.22258274  0.19727954  0.36416792
##  [61]  2.04881965  0.57632605  2.70290404  0.82065807 -0.16952841  0.57939483
##  [67]  2.21379404  0.73547450  0.73215330 -1.03226344  0.26912174  1.17368190
##  [73]  1.36720106  1.44013874  0.53530419 -0.78587755  0.82178138 -0.39815359
##  [79]  2.05548665  1.21066288  2.38722558  2.66616372  2.00184124  1.06361762
##  [85] -0.10405279  1.62832086  1.08269666 -0.20544919  0.55145231  0.61396206
##  [91]  2.05639289  2.03168724  0.53593892  0.72910104  0.17120632  2.53554219
##  [97]  0.86289089 -0.41057608 -0.69059842  0.73200085
```

```
rbinom(100, 1, 0.25)
```

```
##   [1] 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1 0 1 0 1 1 0 0 0 0 1 0 1
##  [38] 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0
##  [75] 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0
```

```
rpois(100, 1)
```

```
##   [1] 0 2 0 1 0 0 0 2 2 1 2 1 1 0 0 0 2 1 1 0 2 0 1 1 1 0 2 0 3 0 0 0 2 1 1 0 0
##  [38] 1 0 2 2 0 1 0 1 1 0 0 2 0 1 2 0 0 1 0 1 1 2 1 4 2 0 2 2 0 1 3 2 0 2 1 0 1
##  [75] 2 3 1 1 1 1 0 3 0 0 0 5 2 2 1 1 2 1 0 5 1 2 2 2 0 1
```

```
runif(100, 1, 20)
```

```
##   [1]  1.775050 14.492223  1.061392  4.962427 19.062054  6.198939 16.184497
##   [8]  6.344608  8.005543  9.080897 10.440173 14.071426  6.631727  5.282448
##  [15]  5.881584  8.715211 14.705688 15.886179 15.141123 15.208999  1.222564
##  [22] 10.083023  8.914297 15.225118 10.608769  8.600083 11.796200  7.445694
##  [29] 15.691906 18.909672  3.030220 11.053485  1.367637 12.243053  8.486541
##  [36] 13.955497 19.434514  2.633013 14.081091 15.036118  1.719115 10.973948
##  [43] 10.034798  5.259506  8.346761  2.624403  2.465146 17.765655 15.006546
##  [50]  5.178202  2.634908 17.370961 18.392128  9.242734  6.508900  4.753415
##  [57] 12.117583  7.270492 18.534859  7.779511 10.815374 19.455208 15.340573
##  [64]  9.263166 14.205161 17.914903 10.150644  9.378430  1.283520  4.833031
##  [71]  9.672971 19.398777 14.995723  5.643688 16.301392 11.180671  2.120431
##  [78] 18.787735 15.586146  4.024193  5.561851 18.694919  9.742778 15.628454
##  [85]  3.463737 16.278471  5.634311  9.340521 14.593710 18.884190 12.278654
##  [92] 13.554109 11.720149  4.971306  3.077000 16.576851 12.069849  9.996735
##  [99] 16.868386 10.860941
```

```
rgamma(100, 1, 1)
```

```
##   [1] 0.36084981 0.53965042 0.89415611 0.53105820 4.53605351 1.51615740
##   [7] 2.20094879 1.89143867 0.66608981 0.21304902 1.54232334 1.21971182
##  [13] 1.61859868 1.07093281 1.00227055 0.74631915 0.43607913 0.36352078
##  [19] 2.09015504 0.25465394 0.03357890 0.61654356 4.40942782 0.19073295
##  [25] 0.11538336 1.02833346 0.26559085 0.04597252 0.39309675 0.59856082
##  [31] 0.07098228 0.63692688 0.26972330 1.53414781 0.03436501 4.05796916
##  [37] 0.13082354 0.17158630 0.63549946 1.10161064 0.81947133 0.36108292
##  [43] 0.57289404 0.19978348 2.03615827 0.34936566 0.71768325 0.41154217
##  [49] 1.38641264 0.06309873 0.41855546 1.39460848 3.45845797 6.23525229
##  [55] 1.67457899 0.27505908 0.67917190 0.20954750 0.87248454 0.27534772
##  [61] 2.42148906 0.92062820 2.47743907 0.53512685 1.78915088 0.77953316
##  [67] 1.04790203 0.26117615 1.21105529 0.41604793 1.00459410 1.54216169
##  [73] 0.30765384 0.72841890 3.07190527 0.38222224 0.95345173 0.99102428
##  [79] 1.42208962 0.41013393 0.82144104 0.40456062 0.44614607 0.42132399
##  [85] 0.47116572 1.57001538 0.50553014 1.59945140 0.10056974 2.58049007
##  [91] 1.73106423 0.23406672 2.51033816 0.16636705 0.59219771 0.05342070
##  [97] 0.55611442 0.82604850 0.25283986 1.20069640
```

Given a number or a list it computes the probability that a random number will be less than that number.

```
pnorm(-1, mean=0, sd=1)
```

```
## [1] 0.1586553
```

```
pnorm(1, mean=0, sd=1)
```

```
## [1] 0.8413447
```

```
pnorm(1, mean=0, sd=1, lower.tail=FALSE)
```

```
## [1] 0.1586553
```

```
pbinom(0, 1, 0.15)
```

```
## [1] 0.85
```

```
ppois(1, lambda=2)
```

```
## [1] 0.4060058
```

```
punif(0.25, 0, 1)
```

```
## [1] 0.25
```

The next function we look at is `q----` which is the inverse of `p----`. The idea behind `q----` is that you give it a probability, and it returns the number whose cumulative distribution matches the probability.

```
pnorm(0.25, mean=0, sd=1)
```

```
## [1] 0.5987063
```

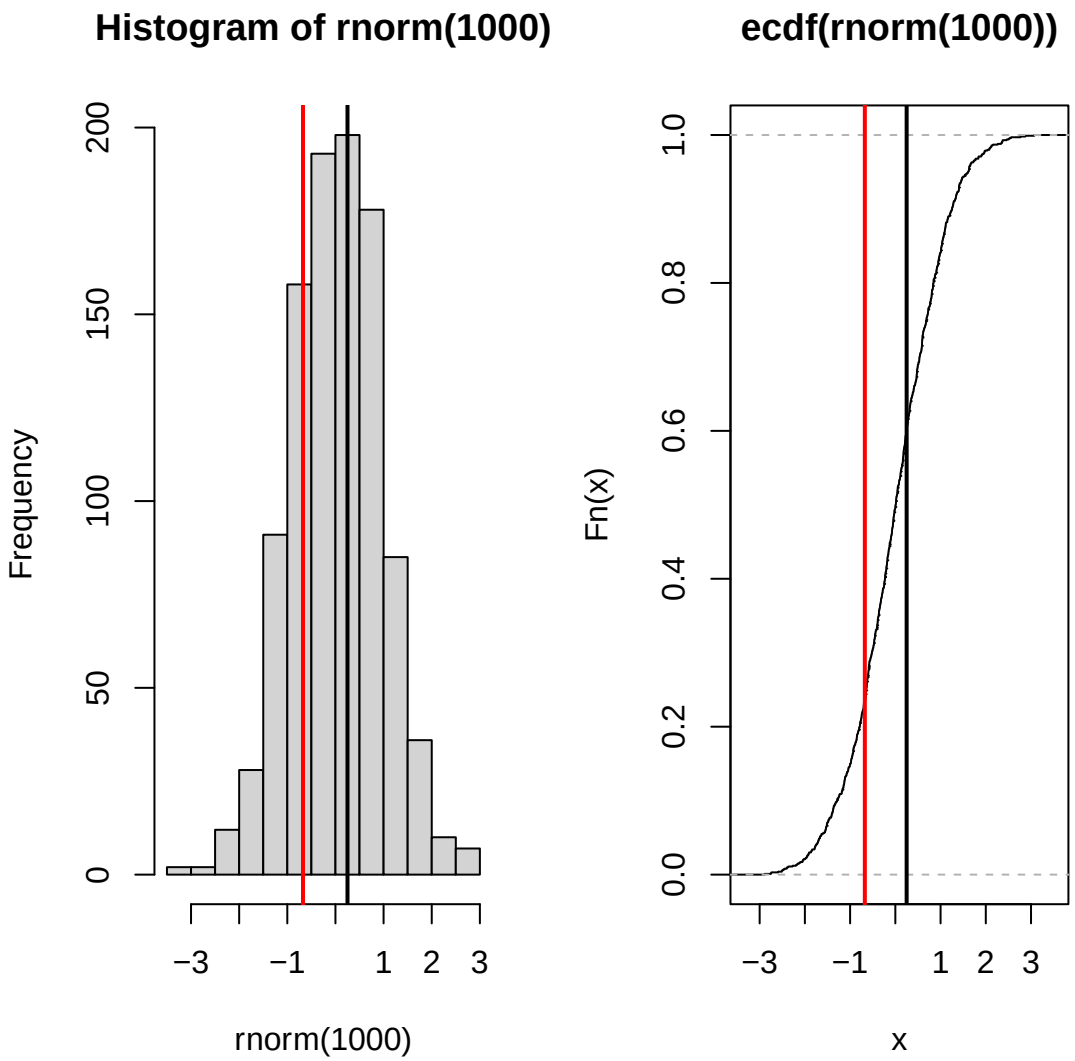
```
qnorm(0.25, mean=0, sd=1)
```

```
## [1] -0.6744898
```

```
pnorm(-0.6744898, mean=0, sd=1)
```

```
## [1] 0.25
```

```
par(mfrow=c(1,2))
hist(rnorm(1000))
abline(v=0.25, lwd=2)
abline(v=-0.6744898, col='red', lwd=2)
plot(ecdf(rnorm(1000)))
abline(v=0.25, lwd=2)
abline(v=-0.6744898, col='red', lwd=2)
```



The last is `d----`, which we will go ahead and ignore for now, as the `r---`, `p----`, and `q----` variants tend to be a bit more useful for thinking about statistical testing.

A case study of the t-test

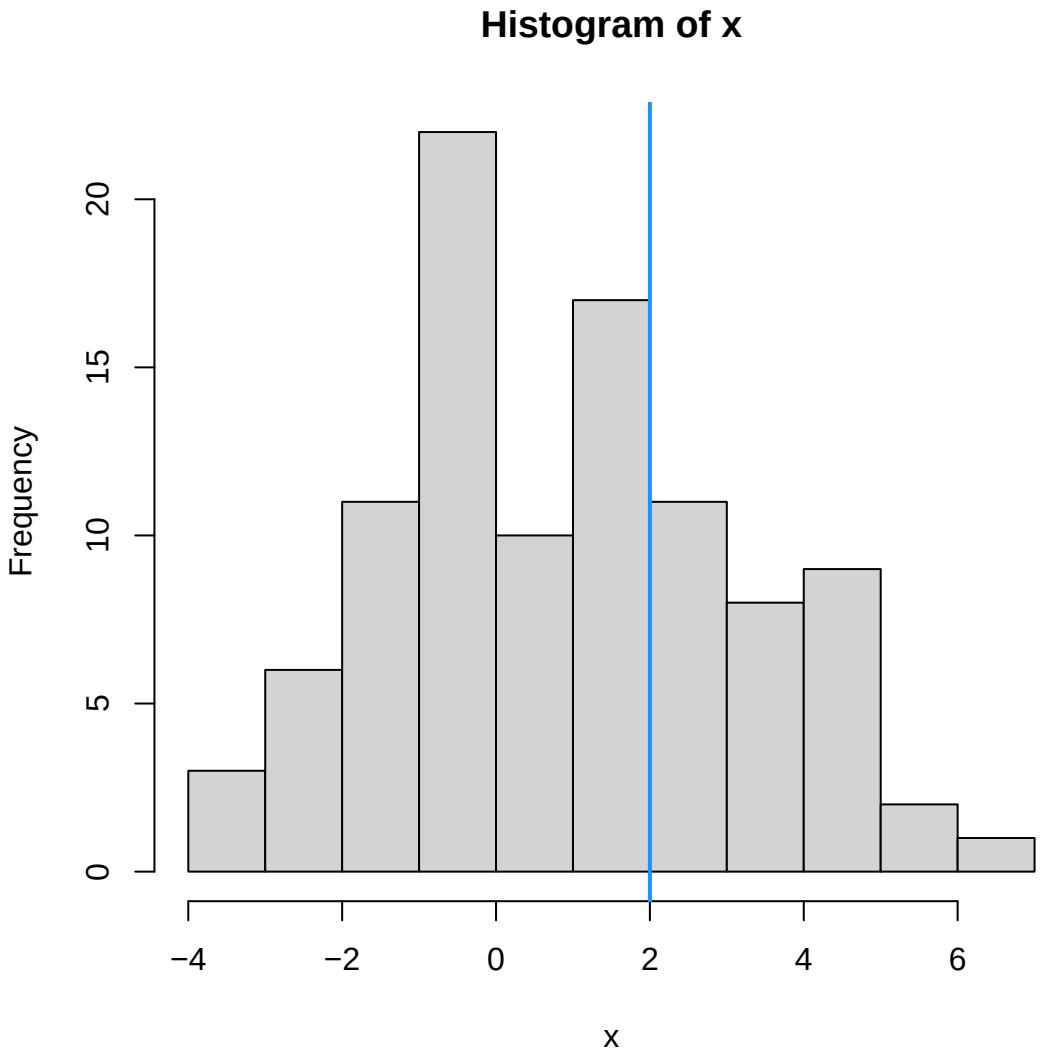
The t-test can be used to test for differences between two groups of data, or of one group of data and some mean μ. It is a comparison of means, so we implicitly assume no differences in variance or distributional shape between the two groups.

One sample

(X - mu) / (sd / sqrt(n))

```
x <- rnorm(100, 1, 2)
mu <- 2

hist(x)
abline(v=mu, col='dodgerblue', lwd=2)
```



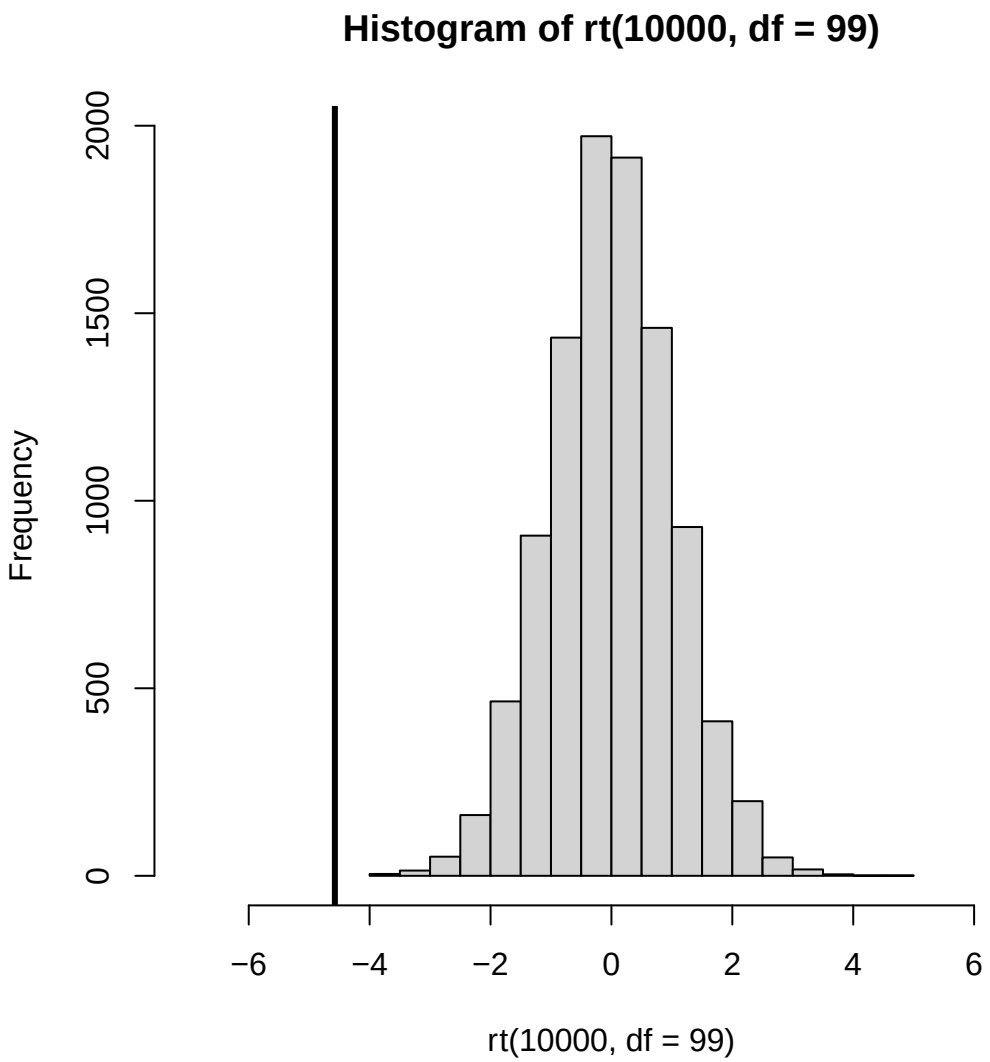

```
t.test(x, mu=mu)
```

```
##
##  One Sample t-test
##
## data:  x
## t = -4.575, df = 99, p-value = 1.382e-05
## alternative hypothesis: true mean is not equal to 2
## 95 percent confidence interval:
##  0.4882994 1.4028995
## sample estimates:
## mean of x
## 0.9455994
```

```
tt <- (mean(x) - mu) / (sd(x) / sqrt(length(x)))
pt(tt, df=length(x)-1)
```

```
## [1] 6.910634e-06
```

```
hist(rt(10000, df=99), xlim=c(-7,7))
abline(v=tt, lwd=3)
```



Two-sample

X1-X2 / (sp)

sp = sqrt((varx1 + varx2) / 2)

sp is pooled variance

```
x1 <- rnorm(100, 1, 2)
x2 <- rnorm(100, 2, 1)

t.test(x1, x2, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
```

```
## data:  x1 and x2
## t = -5.2762, df = 198, p-value = 3.445e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.6013226 -0.7299826
## sample estimates:
## mean of x mean of y
## 0.7496187 1.9152713
```

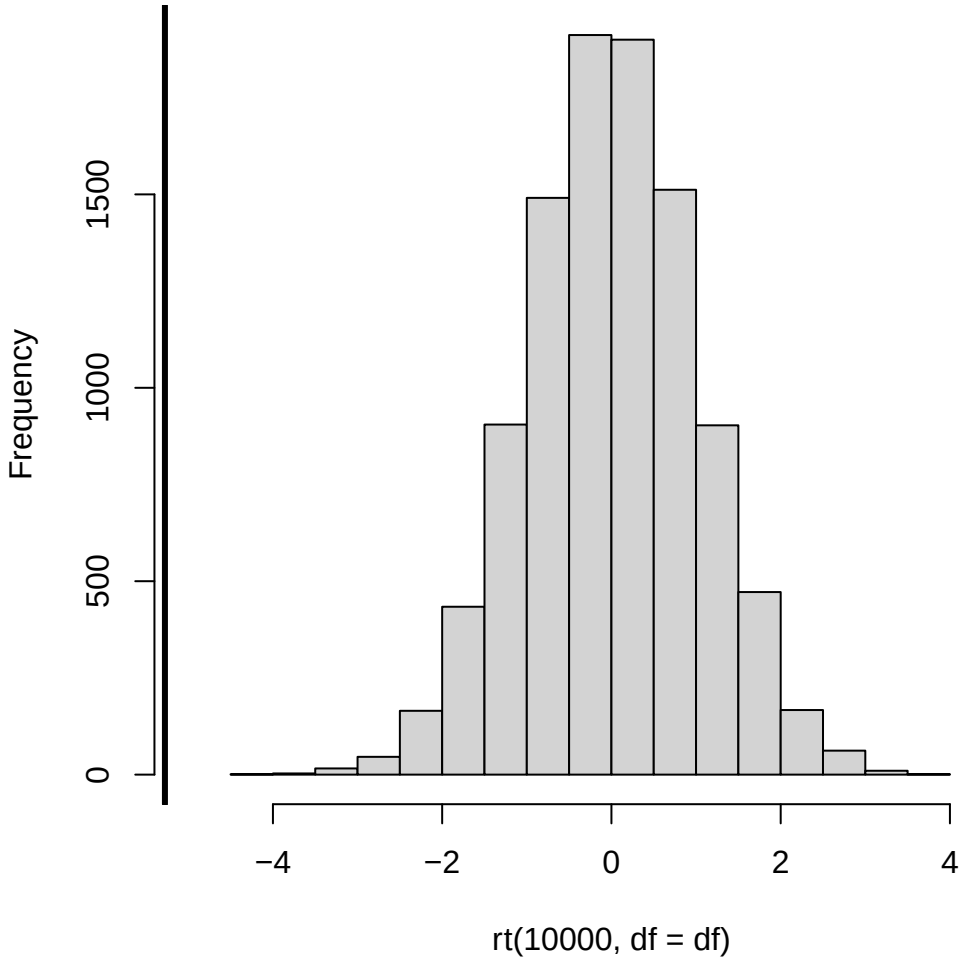
```
sp <- sqrt((var(x1)+var(x2)) / 2)
tt2 <- (mean(x1)-mean(x2)) / (sp*sqrt(2/length(x1)))

df <- (2*length(x1)) - 2

pt(tt2, df=df) * 2
```

```
## [1] 3.445315e-07
```

```
hist(rt(10000, df=df), xlim=c(-5,5))
abline(v=tt2, lwd=3)
```




```
## at larger sample sizes, the normal distribution is approximately equal to the t-distribution (a z-te
pt(tt2, df=df)
```

```
## [1] 1.722658e-07
```

```
pnorm(tt2)
```

```
## [1] 6.594064e-08
```