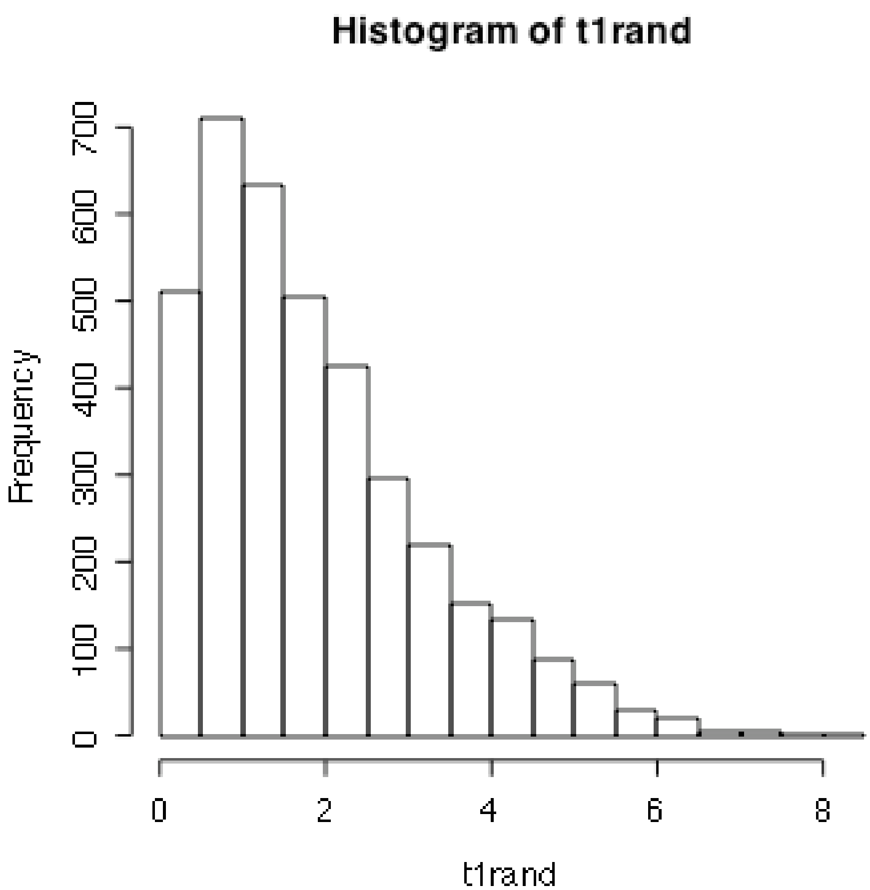
Repeated Measures Randomization Tests

```
> ########################################################
> # Wine1.R.txt  Repeated measures randomization test:    #
> # Compare F = 57.5, df=3,15, p<.0001                    #
> ########################################################
>
> ######################################################################
> # Define margin of error functions
> merror <- function(phat,M,alpha) # (1-alpha)*100% merror for a proportion
+     {
+     z <- qnorm(1-alpha/2)
+     merror <- z * sqrt(phat*(1-phat)/M)  # M is (Monte Carlo) sample size
+     merror
+     }
> mmargin <- function(p,cc,alpha)
+            # Choose m to get (1-alpha)*100% margin of error equal to cc
+            # If true p-value is p
+            {
+            mmargin <- p*(1-p)*qnorm(1-alpha/2)^2/cc^2
+            mmargin <- trunc(mmargin+1) # Round up to next integer
+            mmargin
+            } # End definition of function mmargin
> ######################################################################
>
> wine <- read.table("Wine.data",header=T)
> means <- apply(wine[,2:5],2,mean) ; means
   Wine1    Wine2    Wine3    Wine4
20.00000 22.00000 26.66667 26.00000
> T1 <- var(means) ; T1 # This will be our test statistic
[1] 10.22222
> # How many values are n the permutation distribution?
> # (4!)^6 = 24^6 = 191,102,976 Could do them all, but ...
>
> # How many random permutations should we use?
> # If true p-value is 0.06, want 99% margin of error for estimated p to be
> # less than 0.01
> mmargin(p=0.06,cc=0.01,alpha=0.01)
[1] 3743
> M <- 3800
> winemat <- as.matrix(wine)[,2:5] # Data frames are funny - can't
>                                  # randomize rows separately
>     wine; winemat
  Judge Wine1 Wine2 Wine3 Wine4
1     1    20    24    28    28
2     2    15    18    23    24
3     3    18    19    24    23
4     4    26    26    30    30
5     5    22    24    28    26
6     6    19    21    27    25
  Wine1 Wine2 Wine3 Wine4
1    20    24    28    28
2    15    18    23    24
3    18    19    24    23
4    26    26    30    30
5    22    24    28    26
6    19    21    27    25
```

```
> # Before doing it, show how it works
>     simdat <- NULL
>     for(j in 1:6) simdat <- rbind(simdat,sample(winemat[j,]))
>     simdat
     Wine4 Wine1 Wine3 Wine2
[1,]    28    20    28    24
[2,]    18    23    24    15
[3,]    23    19    18    24
[4,]    30    30    26    26
[5,]    22    24    28    26
[6,]    25    19    27    21
>     T1 ; var(apply(simdat,2,mean))
[1] 10.22222
[1] 1.685185
> # Okay, here we go.
> t1rand <- numeric(M) # Save random T1 values here.
> for(i in 1:M)
+     {
+     simdat <- NULL
+     for(j in 1:6) simdat <- rbind(simdat,sample(winemat[j,]))
+     t1rand[i] <- var(apply(simdat,2,mean))
+     }
> hist(t1rand)
```



```
> length(t1rand[t1rand>T1])
[1] 0
> # Oh well
> means
   Wine1    Wine2    Wine3    Wine4
20.00000 22.00000 26.66667 26.00000
> # Compare wines 1 and 2
> T2 <- abs(means[1]-means[2]) ; T2
Wine1
    2
> winemat <- as.matrix(wine)[,2:3] ; # Just cols 2 and 3
> winemat
  Wine1 Wine2
1    20    24
2    15    18
3    18    19
4    26    26
5    22    24
6    19    21
> 2^6 # Could do them all
[1] 64
> 1:5/64
[1] 0.015625 0.031250 0.046875 0.062500 0.078125
> # Only the top 3 could be significant. Look at winemat. It is in a
> # 4-way tie for first place. Ouch. Proportion Greater than or equal
> # to 2 (results this good or better) is 4/64 = 0.0625, not significant.
>
>
> # Do by simulation as an example.
> t2rand <- numeric(M) # Save random T2 values here.
> for(i in 1:M)
+     {
+     simdat <- NULL
+     for(j in 1:6) simdat <- rbind(simdat,sample(winemat[j,]))
+     meanz <- apply(simdat,2,mean)
+     t2rand[i] <- abs(meanz[1]-meanz[2])
+     }
> hist(t2rand)
>
> randp <- length(t2rand[t2rand >= T2])/M
> margin <- merror(randp,M,.01)
>
> cat ("\n")

> cat ("Randomization p-value = ",randp,"\n")
Randomization p-value =  0.0681579
> cat("99% CI from ",(randp-margin)," to ",(randp+margin),"\n")
99% CI from  0.05762726  to  0.07868853
> cat ("\n")

M <- 10000
Randomization p-value =  0.0573
99% CI from  0.05131339  to  0.06328661

M <- 100000
Randomization p-value =  0.0622
99% CI from  0.06023271  to  0.06416729
```