A Study of the Parametric and Nonparametric Linear-Circular Correlation Coefficient

Robin Tu
California Polytechnic State University, San Luis Obispo
Statistics Department
1 Grand Avenue
San Luis Obispo, California 93410

Abstract

A simulation study of the parametric and nonparametric Linear-Circular Correlation Coefficient was carried out to evaluate the mathematical distribution the statistics followed. A further study was conducted to investigate the effect of ties on the nonparametric correlation coefficient. Lastly, a comparison of power between the parametric and nonparametric Linear-Circular Correlation coefficient was conducted with varying sample sizes, means, and distributions.

Introduction

Circular statistics are specialized statistical methods that deal specifically with directional data. Data that is angular require specialized techniques due to the modulo 2π (in radians) or modulo 360° (in degrees) nature of angles. Critical methods such as the mean ($\bar{x}$) in "linear" statistics do not correctly report the "average" angle. e.g., the average angle between 1° and 359° should not be 180° but 0°.

Correlation, typically in terms of Pearson's correlation coefficient, is a measure of association between two linear random variables x and y. In this paper, the specific circular technique of the parametric and nonparametric linear-circular correlation coefficient will be explored where correlation is no longer between two linear variables x and y, but between a linear random variable x and circular random variable θ. Some specific examples of where measuring linear-circular Correlation may be interesting include:

- Observing wind speed and wind direction.
- Radiation emitted and the sun's position in the sky.
- Watts produced and the crank arm angle when bicycling.
- Ocean current direction and water temperature.

© 2015. Robin Tu

Parametric Linear-Circular Correlation Coefficient

Similar to Pearson's Correlation Coefficient where the underlying conditions require that both x and y are normally distributed, the variability about y does not change with x, linearity between x and y, and independence between x and y, the parametric Linear-Circular Correlation Coefficient, as introduced by Mardia (1976), requires x and θ to be independent and x to be normally distributed. According to Johnson & Wehrly (1977), the multiple correlation coefficient of x and the random vector $(cos\theta, sin\theta)^T$ is $R^2_{x\theta}$ as defined below:

$$R^2_{x\theta} = \frac{r^2_{xc} + r^2_{xs} - 2r_{xc}r_{xs}r_{cs}}{1 - r^2_{cs}}$$

where $r_{xc} = corr(x, cos\theta), r_{xs} = corr(x, sin\theta), r_{cs} = corr(cos\theta, sin\theta)$. The correlations are Pearson sample correlation coefficients. When x and θ are independent and x is normally distributed, the following follows an F distribution with 2 numerator and $n-3$ denominator degrees of freedom.

$$\frac{\frac{1}{2}(n-3)R^2_{x\theta}}{1 - R^2_{x\theta}} \sim F_{2,n-3}$$

Nonparametric Linear-Circular Correlation Coefficient

Analogous to Spearman's Rank Correlation Coefficient between two linear random variables, the nonparametric Linear-Circular Correlation coefficient measures association through rank. Introduced by Mardia (1976) the nonparametric Linear-Circular Correlation Coefficient does not rely on an underlying distribution for the linear variable x and the circular variable θ.

To calculate the nonparametric linear-circular correlation between a linear variable x and circular variable θ, we must first assign ranks to each circular θ_i. Once ranks are assigned to the circular variable, the linear variable is ordered from smallest to largest. A ranked circular variable, β_i, is calculated to be:

$$\beta_i = \frac{2\pi(r_i)}{n}$$

where n is the total number of linear-circular pairs, (x_i,θ_i), and r_i the corresponding circular ranks of $\theta_1, ..., \theta_n$. We then calculate T_c and T_s as shown below:

$$T_c = \sum_{i=1}^{n} x_i cos(\beta_i)$$

$$T_s = \sum_{i=1}^{n} x_i sin(\beta_i)$$

where x_i is the rank of x (when x is tied, the average rank is used for all tied x). The final step is to calculate the correlation coefficient:

$$U = \frac{24(T_c^2 + T_s^2)}{n^2(n+1)} \sim \chi_2^2, \; as \; n \to \infty$$

U follows a χ^2 distribution with 2 degrees of freedom asymptotically. Notice that U does not scale within the traditional R^2 values of between [0,1]. This is taken care of with the following transformations:

$$a_n = \frac{1}{1 + 5cot^2(\frac{\pi}{n}) + 4cot^4(\frac{\pi}{n})}$$

when n is even and:

$$a_n = \frac{2sin^4(\frac{\pi}{n})}{(1 + cos(\frac{\pi}{n}))^3}$$

when n is odd. The scaled correlation D_n is calculated in the following:

$$D_n = a_n(T_c^2 + T_s^2)$$

Simulating the Parametric Linear-Circular Correlation Coefficient

Simulation of the parametric linear-circular correlation coefficient was done using *R* version 3.1.0 *Spring Dance*. 10,000 linear-circular pairs were simulated for sample sizes of 15, 30, 50, 100, and 500 from a Normal(6,2) for the linear variable and Uniform(0,2π) for the circular variable. For each Figure (1 through 5), the Kolmogorov-Smirnov (KS) Test Statistic was used to assess the goodness-of-fit to the theoretical $F_{2,n-3}$ distirbution as stated by Johnson and Wehrly (1977). Additionally, both a plot of the distribution with the $F_{2,n-3}$density overlaid (left) and the F probability plot (right) were generated.

From the adjusted (from multiple testing) p-values corresponding to the Kolmogrov-Smirnov Test, we found that regardless of sample size the distribution of the parametric linear-circular correlation coefficient fits the prescribed $F_{2,n-3}$ distribution well. This can also be seen from both the F probability plot where few points fall off the diagonal line, and the histogram follows, almost exactly, the overlaid F density.

Simulating the Nonparametric Linear-Circular Correlation Coefficient

To investigate the behavior of the nonparametric linear-circular correlation coefficient under ideal conditions, 100,000 linear-circular pairs of sample sizes of 15, 30, 50, 100, 500, and 1000 from a Normal(0,1) for the linear variable and a von Mises(0,1) were generated. Since the nonparametric linear-circular correlation coefficient asymptotically approaches a χ^2 distribution with two degrees of freedom, the particular sample size and the distribution being sampled from were both factors of interest. To investigate how quickly (size of sample needed) the distribution of the correlations followed a χ^2 with two degrees of freedom when sampling from a non-normal distribution, an additional simulation of 100,000 linear-circular pairs

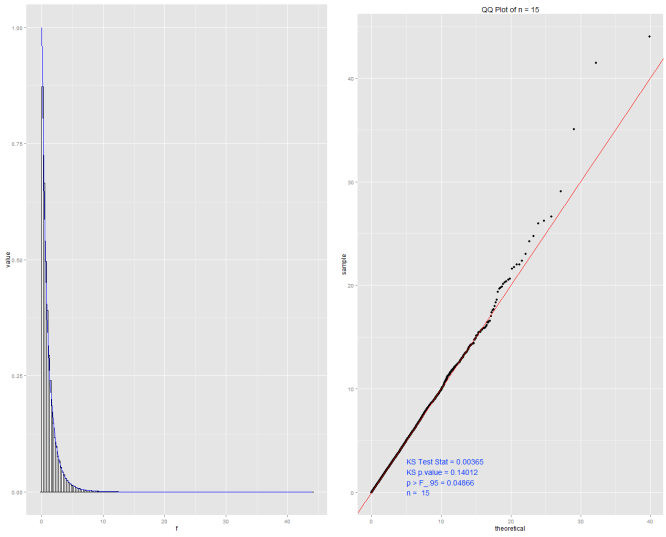

Figure 1: n=15;KS p-value: 0.14012

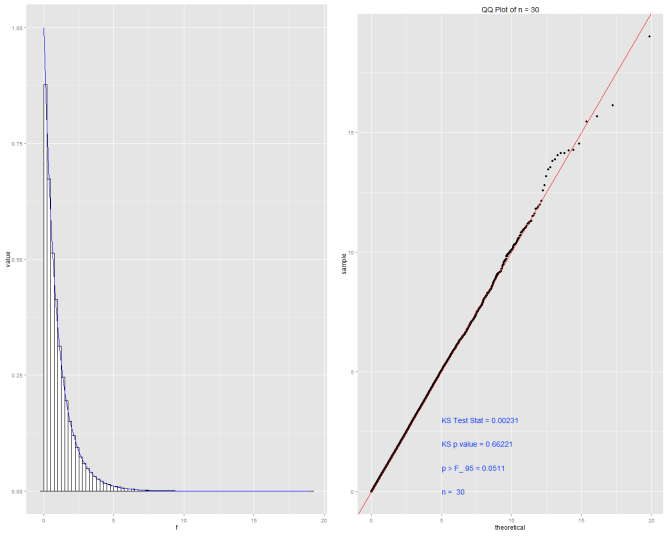

Figure 2: n=30; KS p-value: 0.66221

of sample sizes of 15, 30, 50, 100, 500, and 1000 were generated from a uniform(0,1) for the linear variable and uniform(0,2π) for the circular variable.

In Figure 6 through Figure 11 shown on the next page, the histogram of correlation coefficients (with density curve) and the Chi-Square Probability Plot were created for the linear-circular pairs drawn from a Normal(0,1) and von Mises(0,1). Figure 12 through Figure 17 are also histograms of the correlation coefficient (with density curve) and Chi-Square Probability Plots, but represents the linear-circular pairs drawn from a uniform(0,1) and uniform(0,2π). For both combinations of distribtuions that were sampled from, the Kolmogorov-Smirnov Test Statistic was generated to assess Goodness-of-Fit.

Judging only by the Chi-Square probability plot, at approximately a sample size of 100 is when the fit of the correlation coefficient starts to follow χ^2 with two degrees of freedom. We can see that the Kolmogorov-Smirnov test statistic shows the fit of the correlation coefficient to be improving as sample size increases. Specifically, between a sample size of 100 and 500, the p-value of the Kolmogorov-Smirnov changes from significant to nonsignificant at the $\alpha = 0.05$ significance level. However, from just the fit of the density curve, it appears at a sample size of 15 the linear-circular correlation coefficient begins to follow the theoretical distribution.

Similarly, for Figure 12 through Figure 17, the distribution of the linear-circular correlation coefficient when pairs are drawn from uniform distributions start to follow a χ^2 with two degrees of freedom at a sample size of approximately 100 as seen with the Chi-Square Probability Plot; The Chi-Square Probability Plots of sample sizes greater than 100 do not deviate much from the diagonal line. This is confirmed with the Kolmogorov-Smirnov Test statistic, which changes from significant to nonsignificant at the $\alpha = 0.05$ significance level at this sample size. The density fit of a χ^2 with two degrees of freedom at a sample size of 15 does not seem to be a terrible fit. However, going below this sample size the distribution begins to appear discrete.

Other combinations of distributions were explored such as:

- Linear variable sampled from a Normal Distribution and a Circular variable sampled from a Wrapped-Exponential distribution.
- Linear variable sampled from a Exponential Distribution and a Circular variable sampled from a Wrapped-Exponential distribution.

However, they all exhibited the same properties of the two linear-circular combination exhibited in this paper.

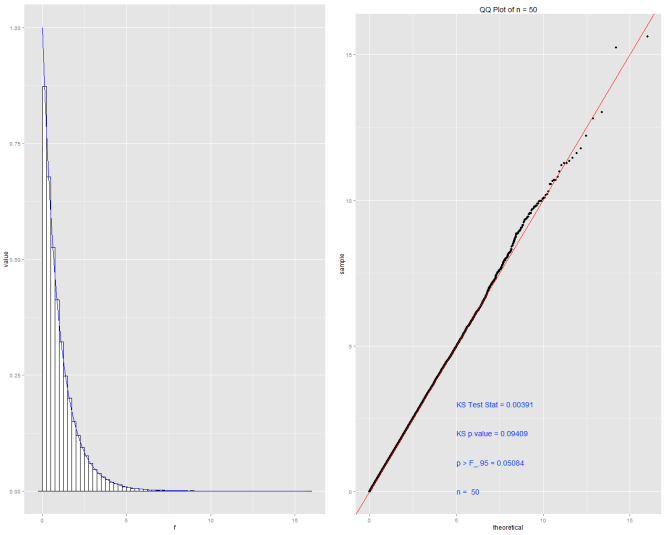

Figure 3: n=50; KS p-value: 0.09409

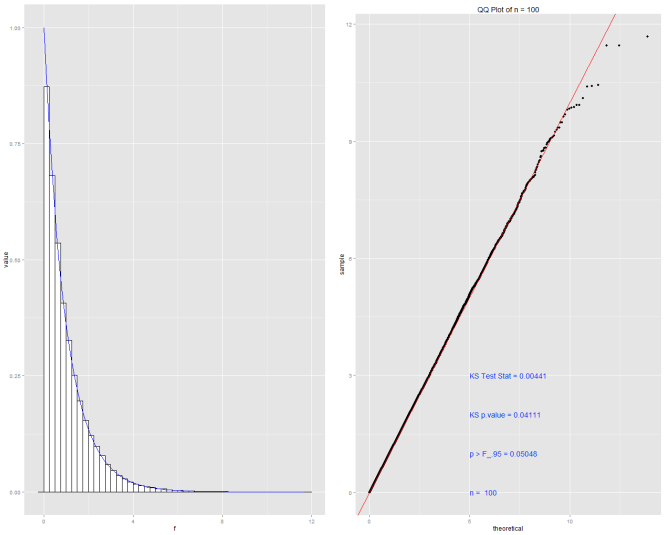

Figure 4: n=100; KS p-value: 0.04111

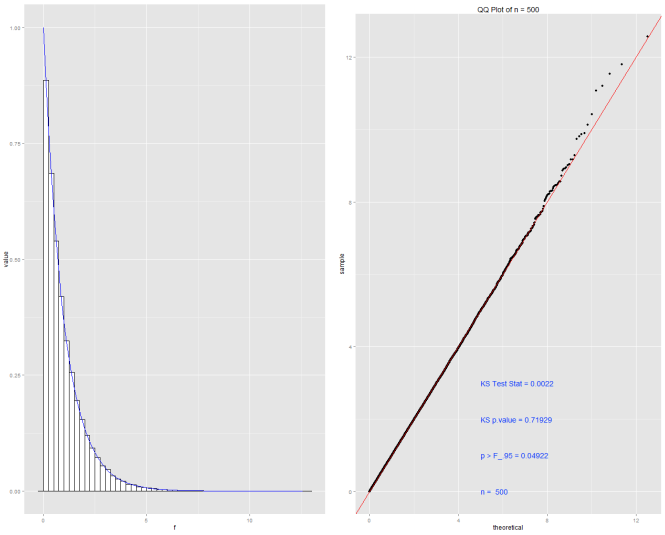

Figure 5: n=500; KS p-value: 0.71929

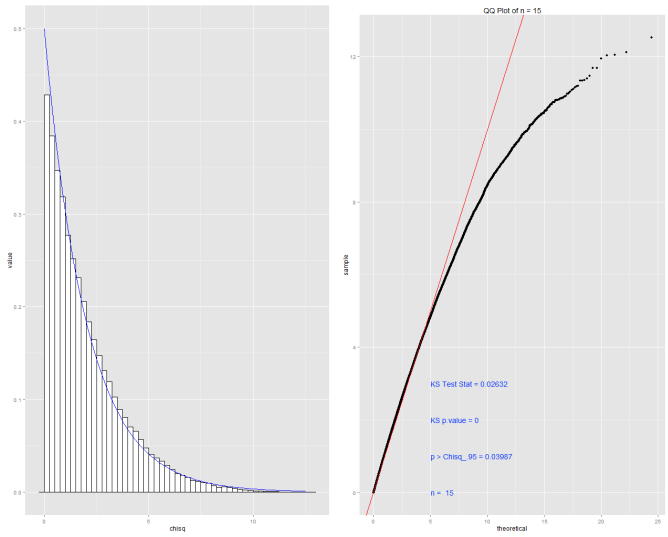

Figure 6: ; n=15; Normal-von Mises; P-val: 0.0263

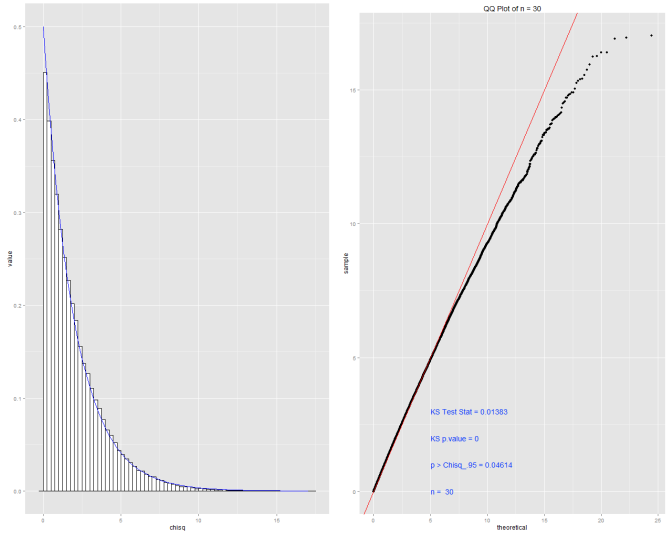

Figure 7: n=30; Normal-von Mises; P-val: 0.0138

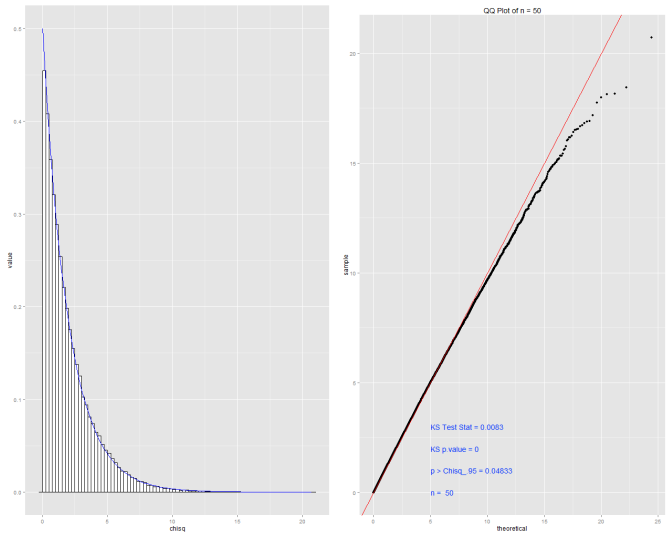

Figure 8: n=50; Normal-von Mises; P-val: 0.0083

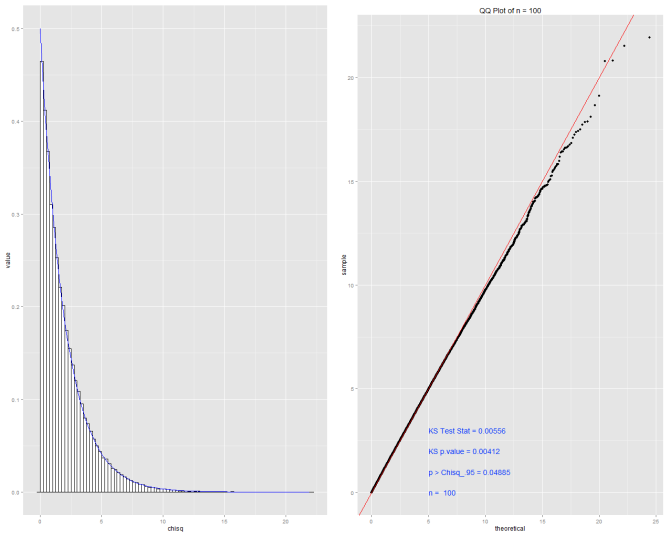

Figure 9: n=100; Normal-von Mises; P-val: 0.00412

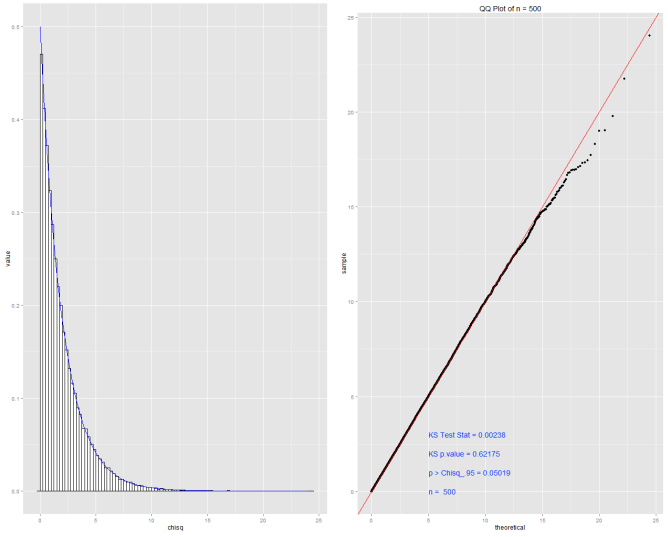

Figure 10: n=500; Normal-von Mises; P-val 0.62175

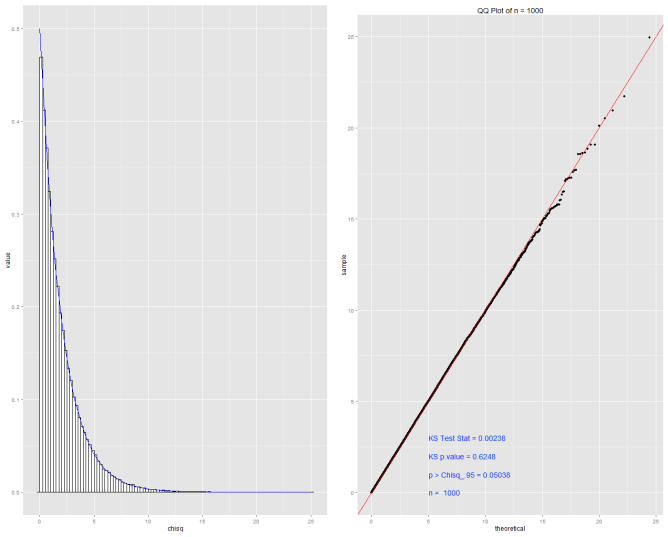

Figure 11: n=1000; Normal-von Mises; P-val 0.6248

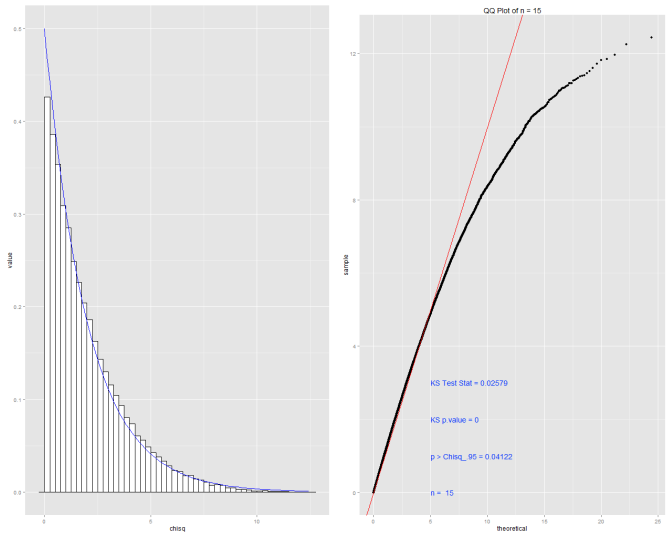

Figure 12: ; n=15; Uniform-Uniform; P-val: 0

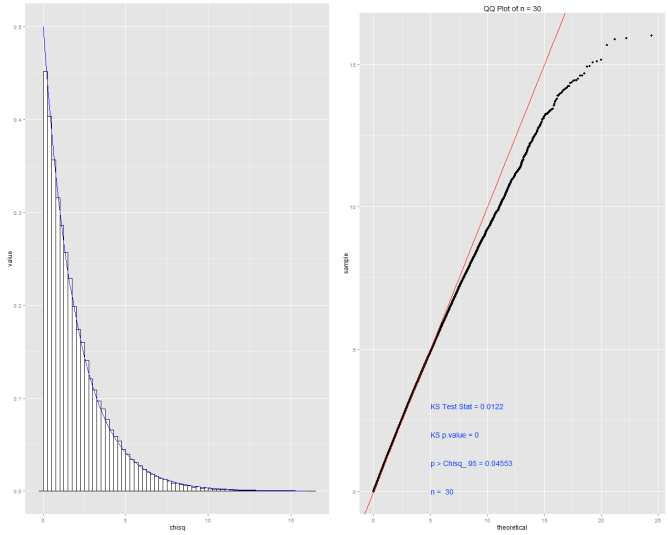

Figure 13: n=30; Uniform-Uniform; P-val: 0

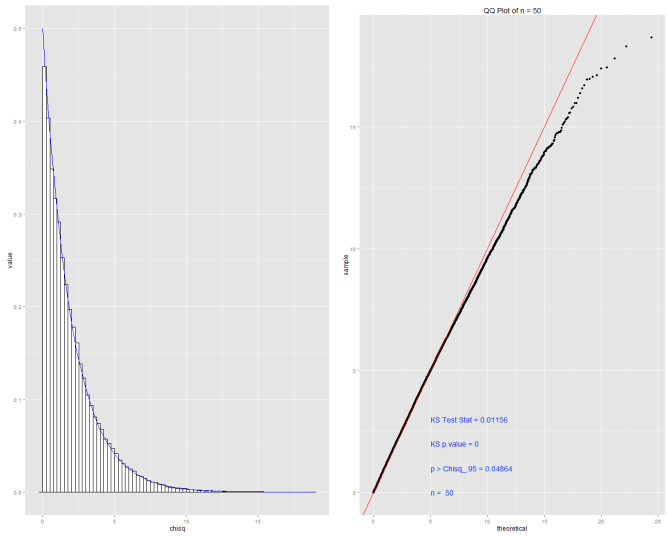

Figure 14: n=50; Uniform-Uniform; P-val: 0

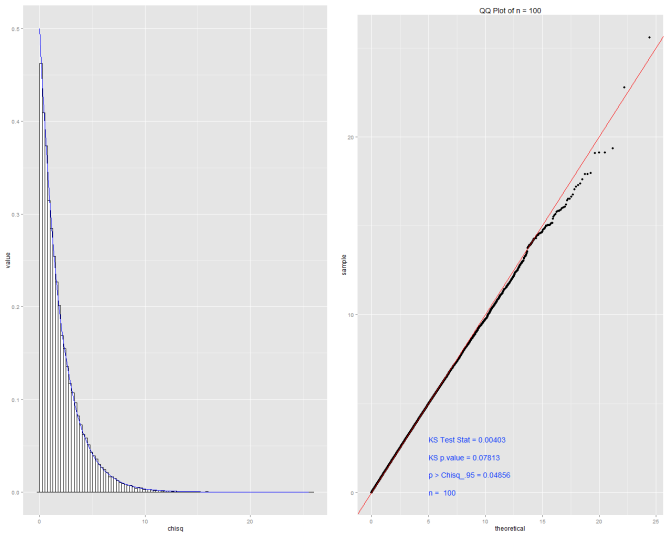

Figure 15: n=100; Uniform-Uniform; P-val: 0.07813

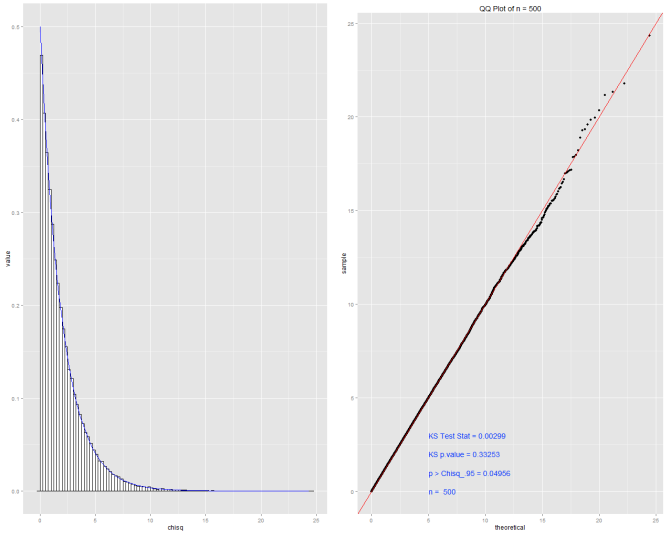

Figure 16: n=500; Uniform-Uniform; P-val 0.33253

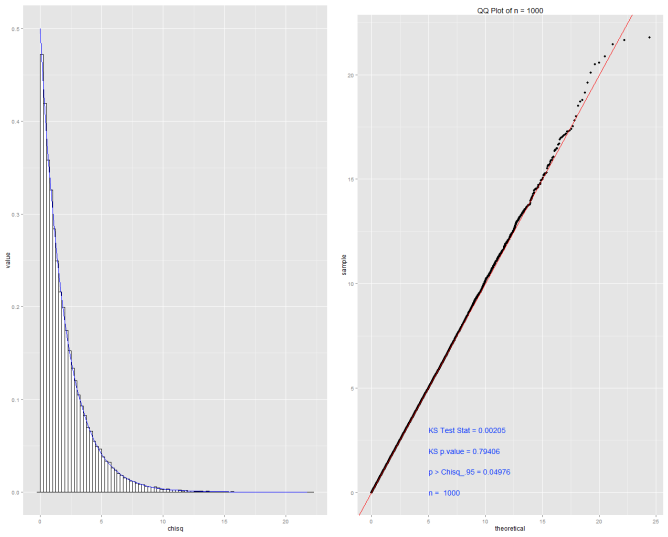

Figure 17: n=1000; Uniform-Uniform; P-val 0.79406

Ties and the Nonparametric Linear-Circular Correlation Coefficient

To further investigate the nonparametric linear-circular Correlation Coefficient, the effect of ties on the correlation's property of asymptotically following a χ^2 distribution with two degrees of freedom was explored. 1,000,000 simulations of sample size 15, 30, 50, and 100 were conducted and both the number of ties and the nonparametric correlation coefficient was recorded. The number of ties was computed to be:

$$n - (unique\ values)$$

The linear variable was sampled from Uniform(0,10) and the circular variable was sampled from Uniform(0,2π). Uniform distributions were chosen so that the location of the tie would be uniformly distributed, eliminating the effect of the location of the tie.

Ties were generated artificially by rounding to the nearest hundredth and the nearest tenth of just the linear variable. We were not capable of controlling the total proportion of ties—only the way we tied the data. The rank of the ties were decided by their average ranks, as was done by Fisher and Lee (1981) with the data supplied from Johnson and Wehrly (1977).

Although we rounded to the hundredth and tenth digit, only the rounding to the nearest integer for the linear variable will be shown. Roughly 80-90% of the data resulted in ties. Although the p-value corresponding to the Kolmogorov-Smirnov test statistic is significant, as shown in the following Figures (18 through 21), the χ^2 density fit onto the histogram is excellent despite the large proportion of tied values, demonstrating the robustness of the test statistic.

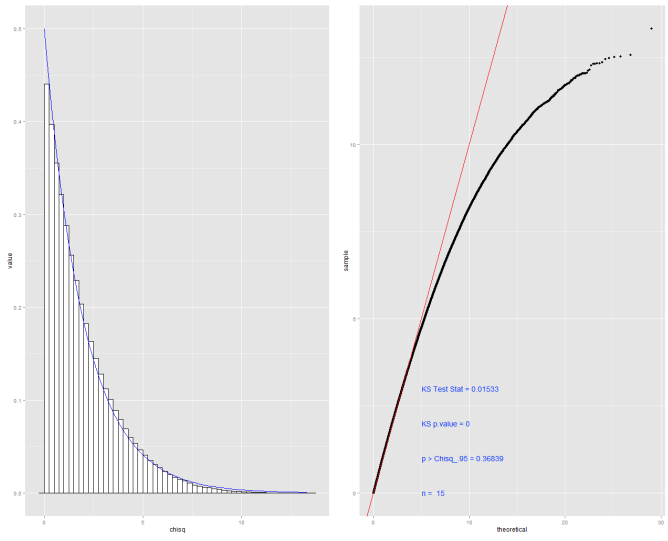

Figure 18: n=15; Ties Uniform (0,10); P-val: 0

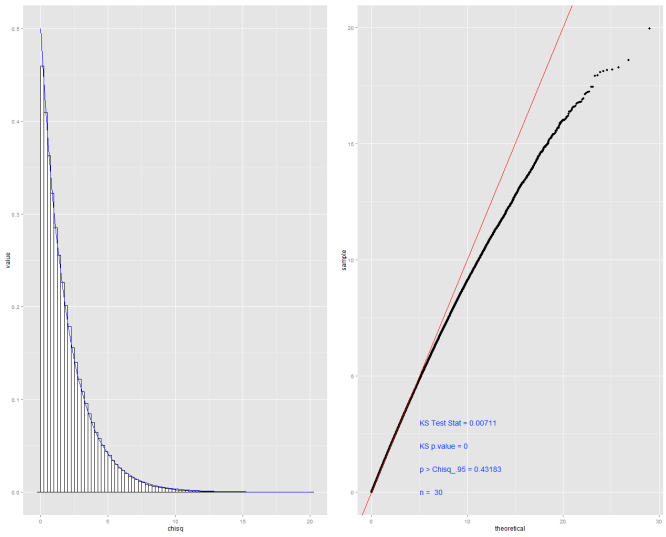

Figure 19: n=30; Ties Uniform (0,10); P-val: 0

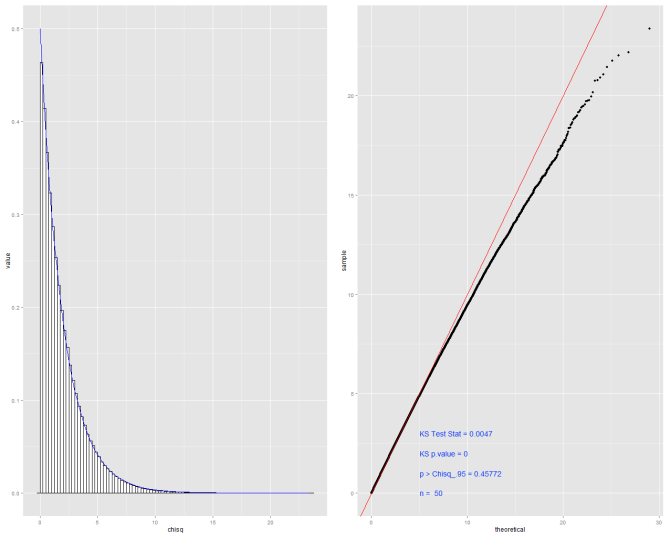

Figure 20: n=50;Ties Uniform (0,10); P-val 0

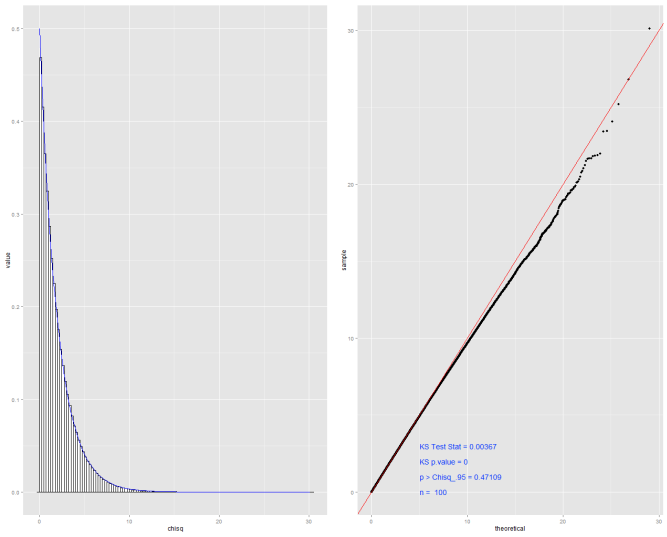

Figure 21: n=100;Ties Uniform (0,10); P-val 0

Comparing Power Between the Parametric and Nonparametric Linear-Circular Correlation Coefficient

Power, the probability of rejecting the null hypothesis, H_o, was simulated by sampling the linear variable from a normal distribution and the circular variable from a Uniform Distribution. Correlation was created in the data by forcing linear values with circular values between 0° (0 radians) and 90° ($\frac{\pi}{2}$ radians) to follow a Normal (20,1) distribution. The corresponding linear values for circular values that fell outside of the arc (90° to 360° or $\frac{\pi}{2}$ to 2π radians) are to follow a Normal Distribution with variance of 1, but mean increasing from 1 to 20 in increments of 0.2. As the simulation approaches N(20,1) on the interval $[\pi/2,\ 2\pi]$ power should decline until it becomes the set significance level. This was done for sample sizes of 15, 30, 50, 100, and 500.

The following Figures best demonstrate how power was simulated. Each of the Figures below are a single simulation of sample size of 1000 from the respective distributions in the caption.

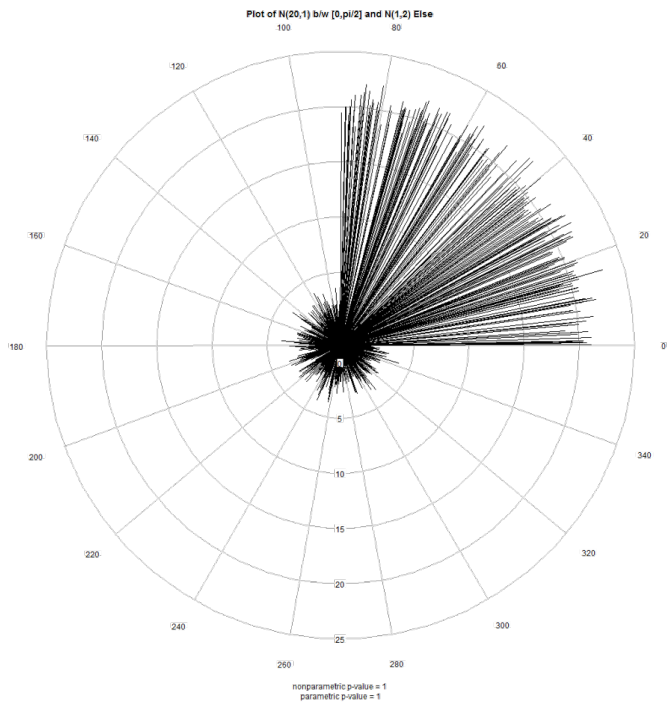

Figure 22: N(20,1) b/w $[0, \frac{\pi}{2}]$, N(11,2) Else;
nonparametric p-value = 1
parametric p-value = 1

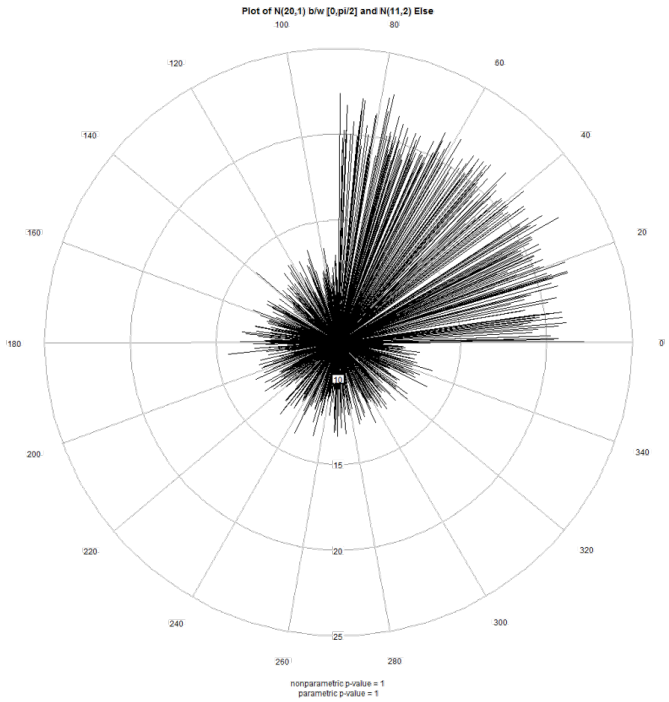

Figure 23: N(20,1) b/w $[0, \frac{\pi}{2}]$, N(1,2) Else;
nonparametric p-value = 1
parametric p-value = 1

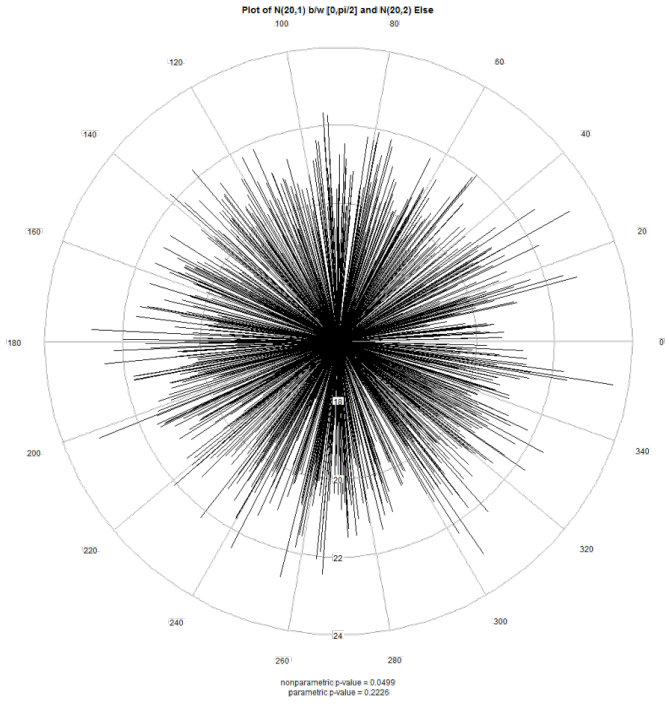

Figure 24: N(20,1) b/w $[0, \frac{\pi}{2}]$, N(20,2) Else;
nonparametric p-value = 0.0499
parametric p-value = 0.0501

Simulating X from a Normal Distribution. As seen in the simulation below, as sample size increases, the nonparametric linear-circular correlation power curve (red) becomes increasingly indistinguishable from the parametric linear-circular power curve (blue). At a sample size of 500, the two curves become one and the same.

Simulating X from an Exponential Distribution. A similar simulation study of power to compare the parametric vs. nonparametric linear-circular correlation coefficient using an exponential distribution instead of a normal distribution was carried out to see how the parametric linear-circular correlation fares when the condition of the linear variable following a Normal distribution is not satisfied. Values of $\bar{x}$ where values of θ fell between $[0,\frac{\pi}{2}]$ followed an Exponential distribution with $\lambda = 1/20$. Values from $[\frac{\pi}{2},2\pi]$ followed an Exponential distribution with $\lambda = 1/j$ where j took values from 1 to 20 in increments of 0.2. Similar to the simulation immediately previous to this one, the blue curve demonstrates the parametric Linear-Circular power, the red curve, nonparametric linear-circular power. It is noteworthy that again, the parametric linear-circular correlation coefficient seems to have more power at every sample size for every λ value simulated despite conditions not being met.

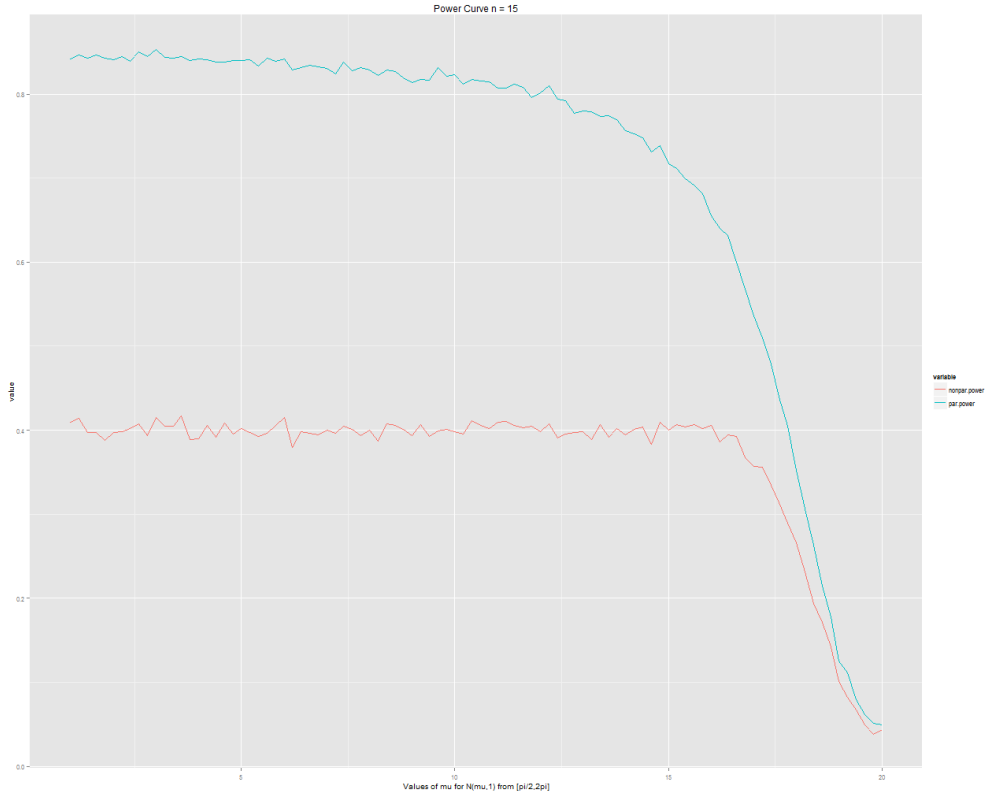

Figure 25: n=15; Normal-Wrapped Uniform

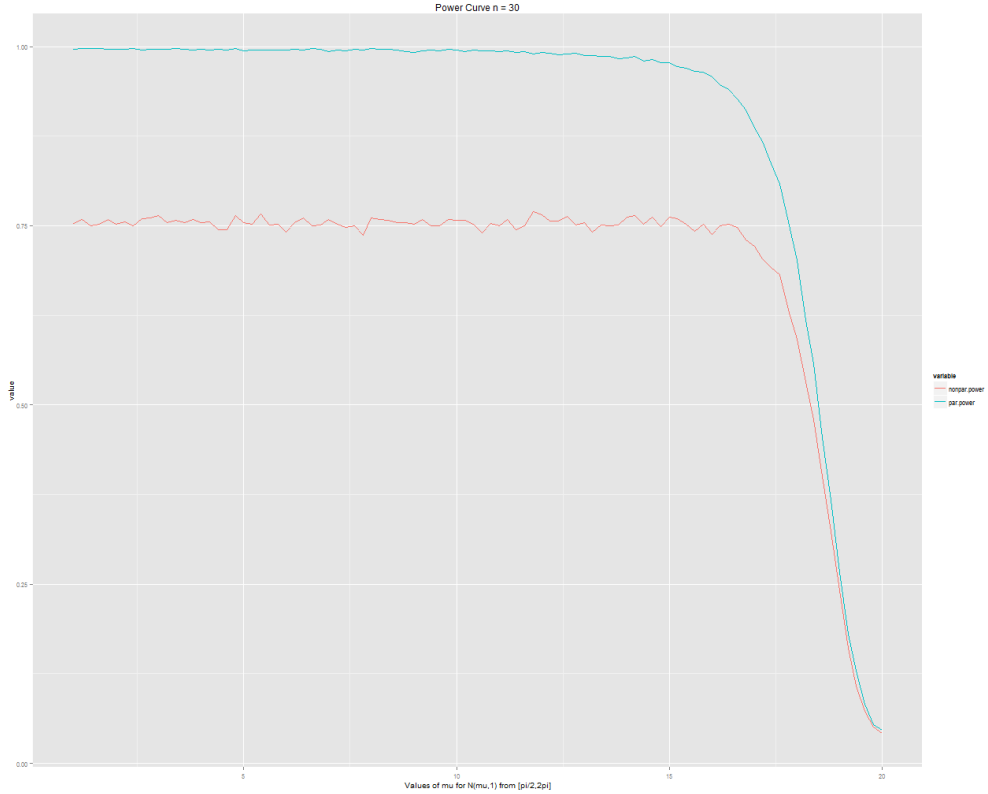

Figure 26: n=30;Normal-Wrapped Uniform

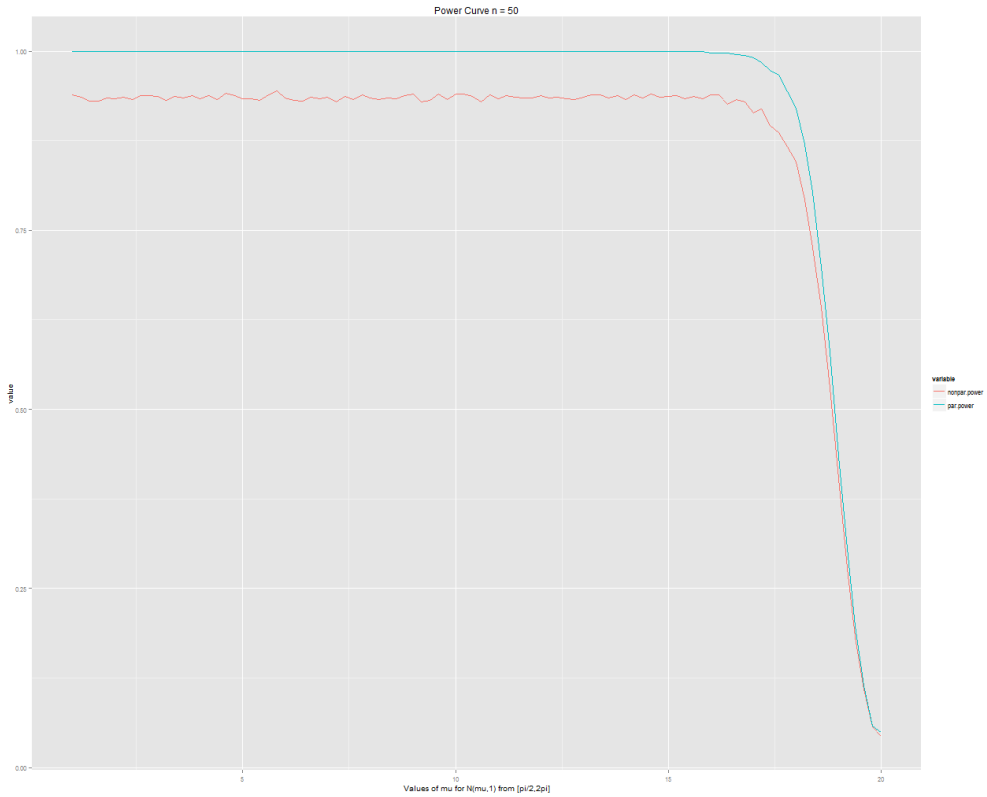

Figure 27: n=50;Normal-Wrapped Uniform

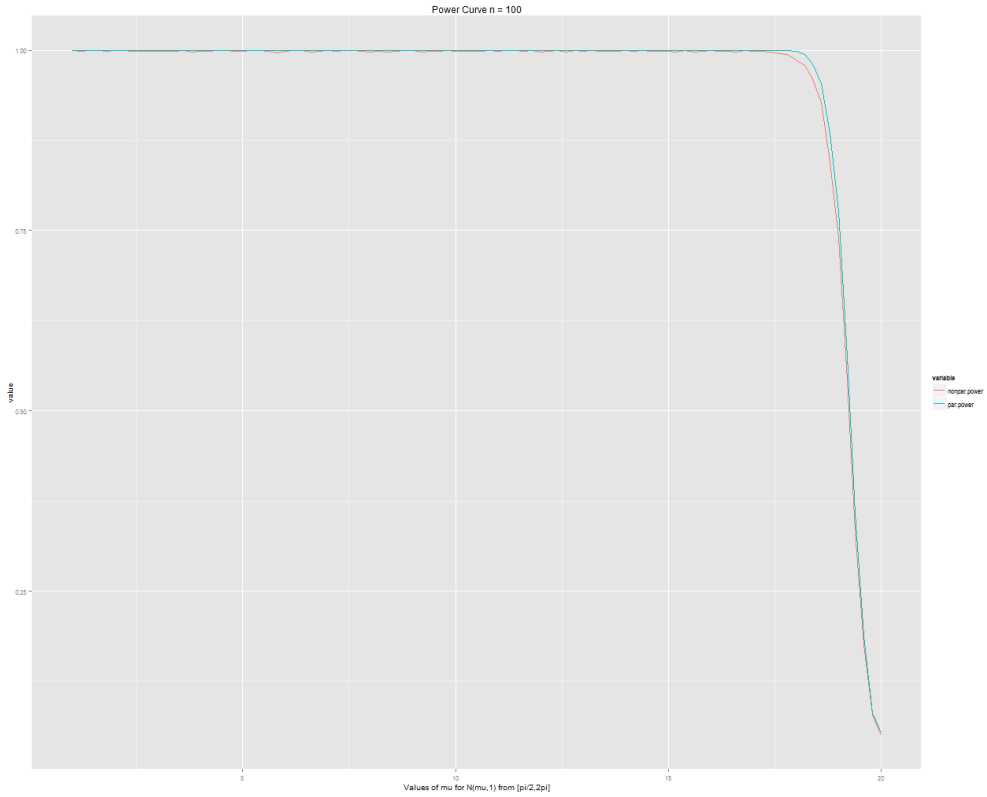

Figure 28: n=100;Normal-Wrapped Uniform

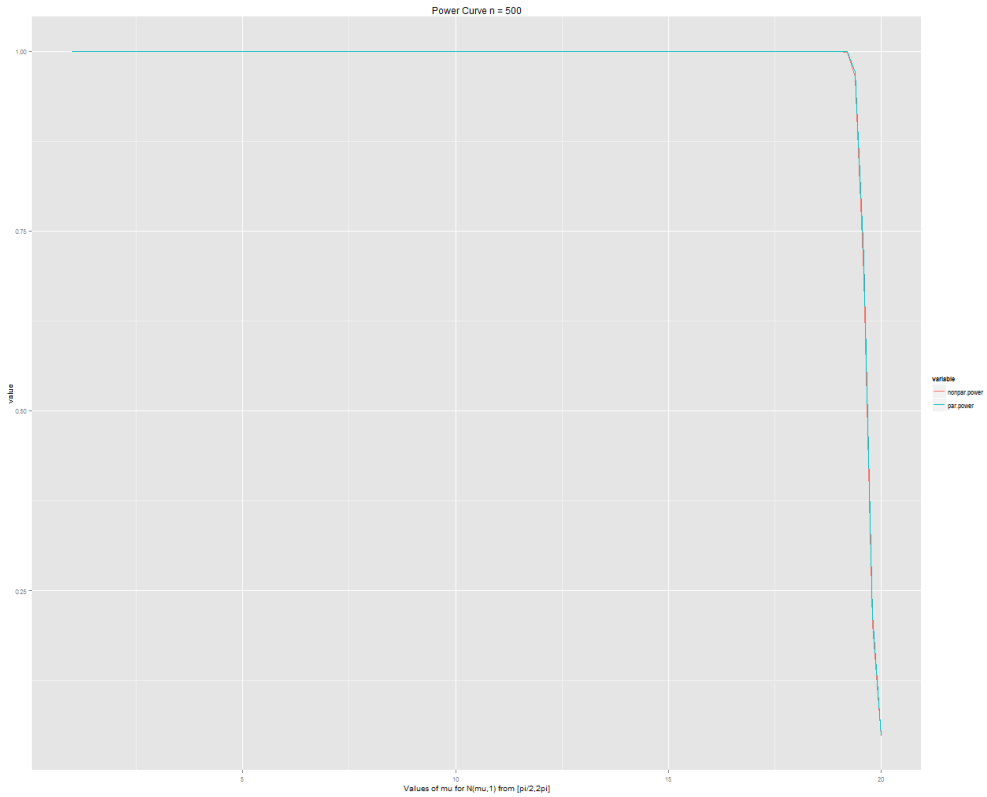

Figure 29: n=500;Normal-Wrapped Uniform

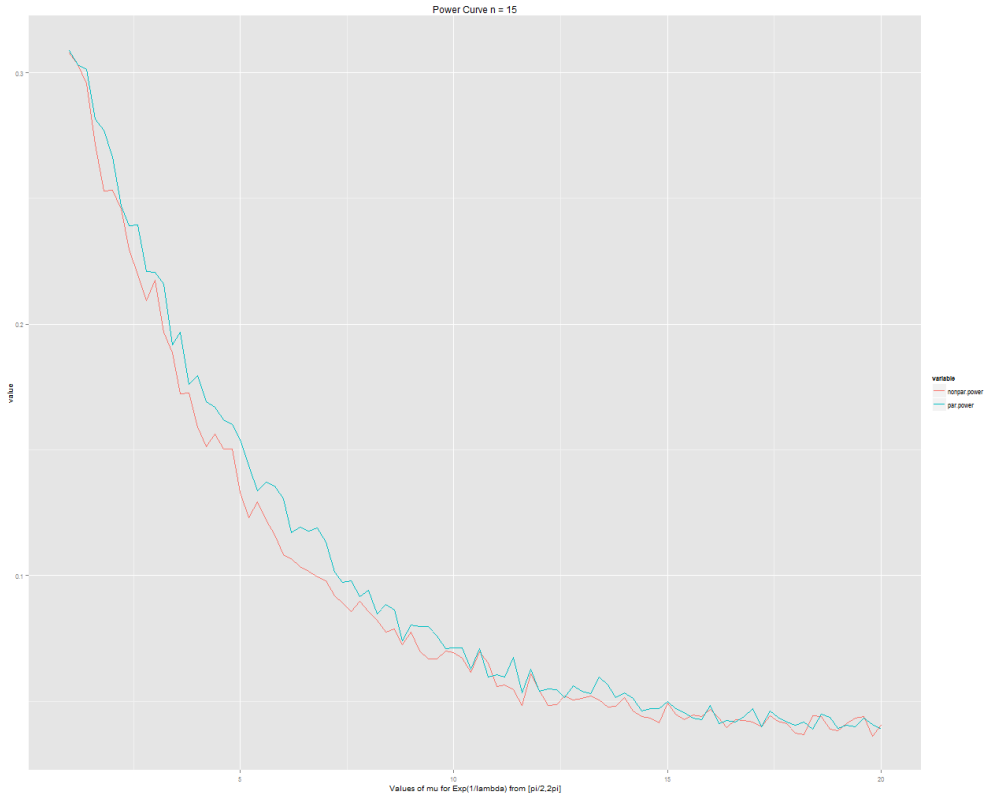

Figure 30: n=15;Exponential-Wrapped Uniform

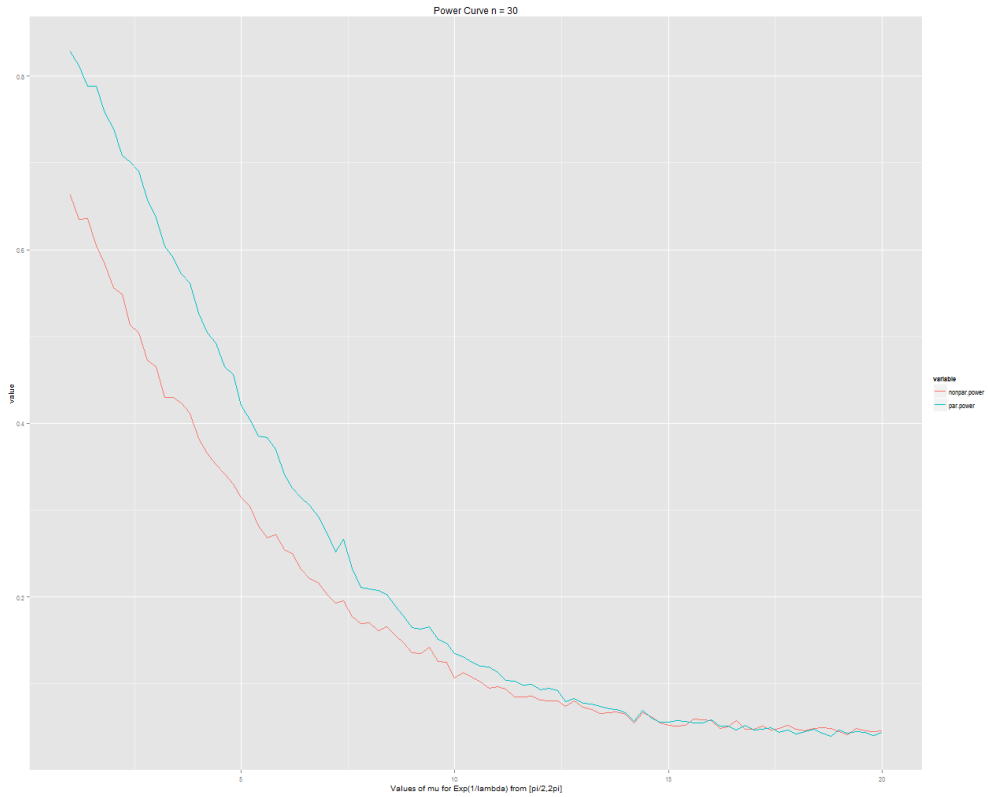

Figure 31: n=30;Exponential-Wrapped Uniform

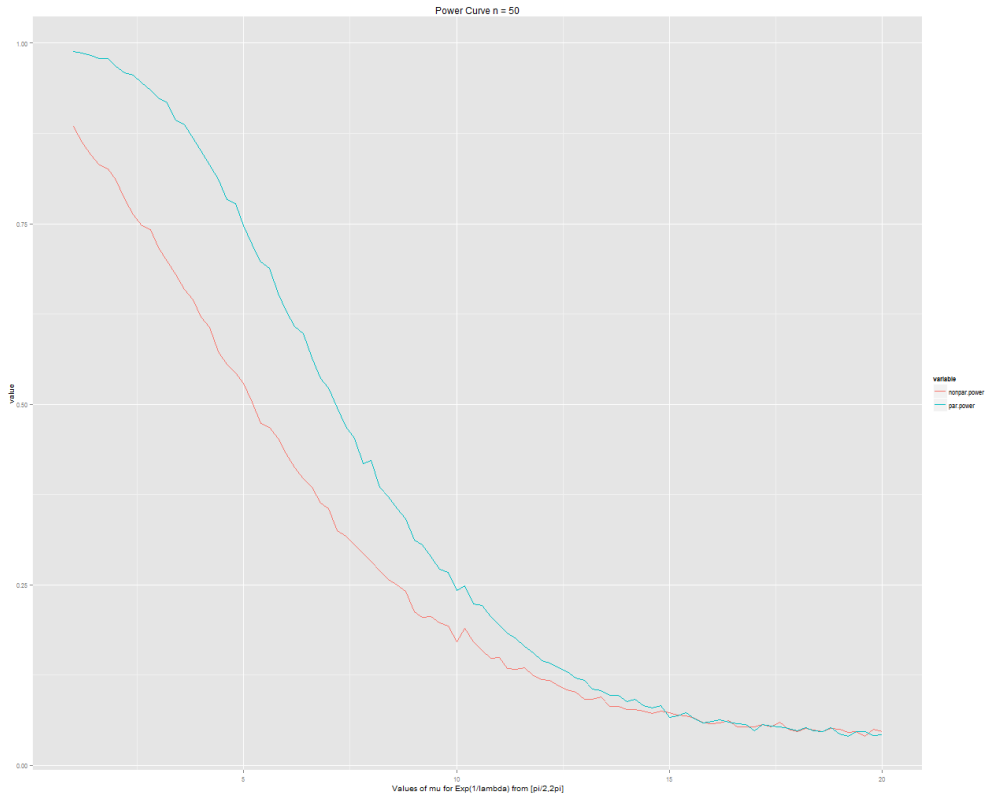

Figure 32: n=50;Exponential-Wrapped Uniform

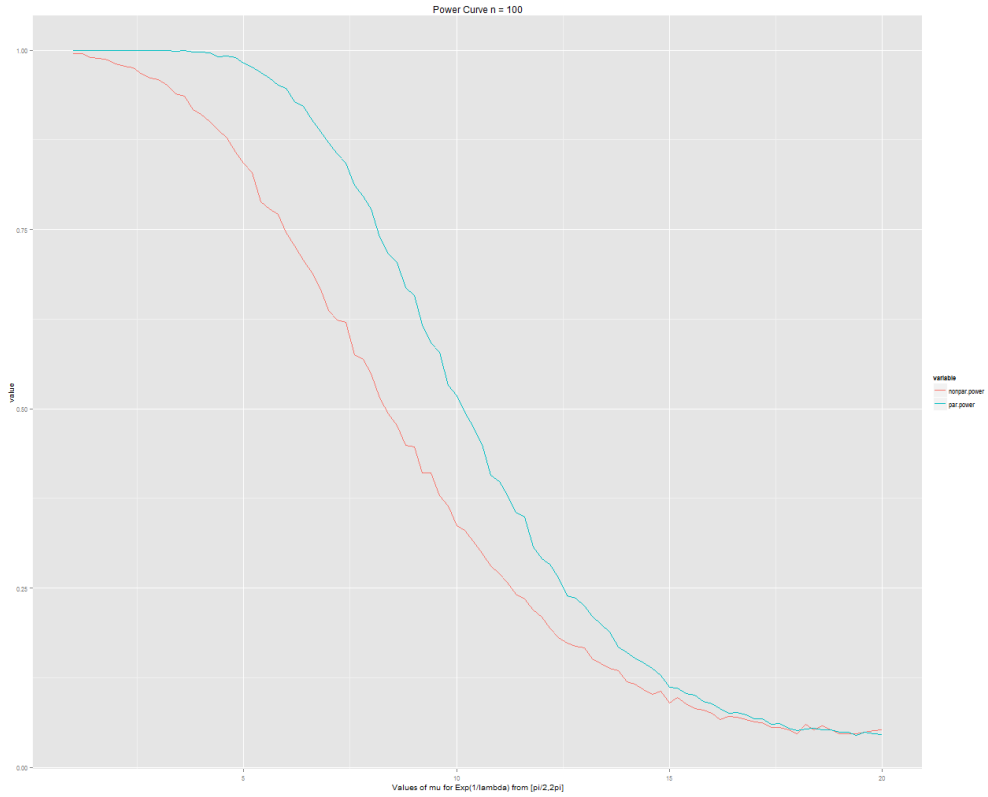

Figure 33: n=100;Exponential-Wrapped Uniform

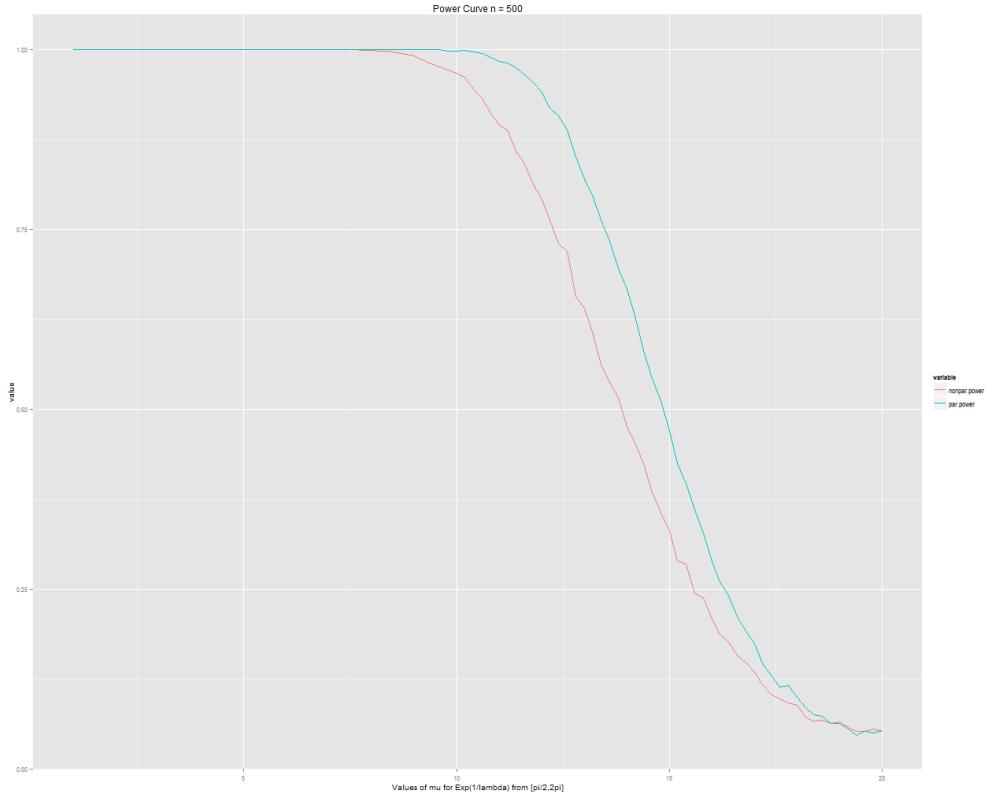

Figure 34: n=500;Exponential-Wrapped Uniform

Conclusion

In short, the nonparametric and parametric test statistics both follow their theoretical distributions (asymptotically for the nonparametric statistic). It was found that the nonparametric statistic was robust against ties. Additionally, it was also found that the power of the parametric statistic outperformed the nonparametric statistic for almost all values of λ for our exponential linear random variable.

There are many avenues of the parametric and nonparametric linear-circular correlation coefficients still left unexplored. In this paper the effect/impact of ties on the parametric linear-circular correlation coefficient was not explored. Additionally why the parametric linear circular correlation coefficient was more powerful for the nonnormal linear variable that was simulated was not explored. Furthermore, the simulation study only varied the mean and λ value for the normal and exponential distribution respectively.

Future studies can include different distributions as well as varying the various parameters required for those distributions. One could also improve on the method of tying and attempt to understand why the nonparametric test statistic is robust against ties.

Acknowledgments

Dr. Ulric Lund for being pleasant to talk to [almost] every Friday and really giving me direction in directional statistics.

A shameless plug for his other, nonacademic abilities is below.

P-values

Probability

null hypothesis is true.

No partial credit.

-Dr. Lund

References

- Batschelet E., Hillman D., Smolensky M., Halberg F. (1973), Angular-Linear Correlation Coefficient for Rhythometry and Circannually Changing Human Birth Rates at Different Geographic Latitudes, International Journal of Chronobiology, Vol.1, pp. 183-202
- Feridun Tasdan & Meral Cetin (2014), A simulation study on the influence of ties on uniform scores test for circular data, Journal of Applied Statistics, 41:5,pp. 1137-1146
- Fisher N.I., Lee A.J. (1981), Nonparametric Measures of Angular-Linear Association, Biometrika, Vol. 68 No. 3 (Dec., 1981), pp. 629-636
- Johnson R.A., Wehrly T. (1977), Measures and Models for Angular Correlation and Angular-Linear Correlation, Journal of the Royal Statistical Society, Series B (Methodological) Vol. 39 No. 2, pp. 222-229
- Liddell I.G., Ord J.K. (1978), Linear-Circular Correlation Coefficients: Some Further Results, Biometrika, Vol 65, No. 2 (Aug., 1978), pp. 448-450
- Mardia K.V. (1976), Linear-Circular Correlation Coefficients and Rhythmometry, Biometrika, Vol. 63 No. 2 (August., 1976), pp. 403-405

R Packages Used:

Couldn't have done this project without the help of the following authors and their R packages.

- C. Agostinelli and U. Lund (2013). R package 'circular': Circular Statistics (version 0.4-7). URL https://r-forge.r-project.org/projects/circular/
- Hadley Wickham (2007). Reshaping Data with the reshape Package. Journal of Statistical Software, 21(12), 1-20. URL http://www.jstatsoft.org/v21/i12/.
- H. Wickham. ggplot2: elegant graphics for data analysis. Springer New York, 2009.
- Baptiste Auguie (2012). gridExtra: functions in Grid graphics. R package version 0.9.1. http://CRAN.R-project.org/package=gridExtra
- Lemon, J. (2006) Plotrix: a package in the red light district of R. R-News, 6(4): 8-12.
- Revolution Analytics and Steve Weston (2014). foreach: Foreach looping construct for R. R package version 1.4.2. http://CRAN.R-project.org/package=foreach
- Revolution Analytics and Steve Weston (2014). doSNOW: Foreach parallel adaptor for the snow package. R package version 1.0.12. http://CRAN.R-project.org/package=doSNOW
- R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

Appendix

Parametric Linear-Circular Correlation Coefficient

```
#### Parametric Linear Circular Correlation
    Coefficient ####
y = as.circular(theta,units = "degrees",
    type = "angles")

data = data.frame(x,y);


cor.circular.lc = function(x,y=NULL,test =
    FALSE){
  ### x   vector or matrix of linear data
  ### y   vector or matrix of circular data
  ### test if test == TRUE then a
      significance test for the correlation
      is computed

  if (!is.null(y) & NROW(x) != NROW(y))
    stop("x and y must have the same number
        of observations")
  if (is.null(y) & NCOL(x) < 2)
    stop("supply both x and y or a
        matrix-like x")
  ncx <- NCOL(x)
  ncy <- NCOL(y)
  if (is.null(y)) {
    ok <- complete.cases(x)
    x <- x[ok, ]
  }
  else {
    ok <- complete.cases(x, y)
    if (ncx == 1) {
      x <- x[ok]
    }
    else {
      x <- x[ok, ]
    }
    if (ncy == 1) {
      y <- y[ok]
    }
    else {
      y <- y[ok, ]
    }
  }
  n <- NROW(x)
  if (n == 0) {
    warning("No observations (at least after
        removing missing values)")
    return(NULL)
  }
  ### Converting y to radians ###
  if (!is.null(y)) {
    y <- conversion.circular(y, units =
        "radians", zero = 0,
                        rotation = "counter",
                            modulo = "2pi")
    attr(y, "class") <- attr(y, "circularp")
        <- NULL
  }
  if(is.null(y)){
    z = conversion.circular(x[,2], units =
        "radians", zero = 0,
                          rotation =
                              "counter",
                              modulo = "2pi");
    attr(z, "class") <- attr(z, "circularp")
        <- NULL;
    r_xs = cor(x[,1],sin(z));
    r_xc = cor(x[,1],cos(z));
    r_cs = cor(cos(z),sin(z));
  }else{

    ### calculating individual components ###
    r_xs = cor(x,sin(y));
    r_xc = cor(x,cos(y));
    r_cs = cor(cos(y),sin(y));
  }
  ### calculating correlation coeff
      linear-circular ###
  cor.lc = (r_xc^2 + r_xs^2 -
      2*(r_xc*r_xs*r_cs))/(1-r_cs^2);

  if(test){
    f.stat = (.5*(n-3)*cor.lc)/(1-cor.lc);
    p.val = pf(f.stat,df1 = 2, df2=
        n-3,lower.tail = FALSE);
    result = list(cor = cor.lc, statistic =
        f.stat, p.value = p.val);

  }else{
    result = list(cor = cor.lc);
  }
  return(result);
}
```

Nonparametric-Linear Circular Correlation Coefficient

```
cor.circular.lc.rank = function(x, y =
    NULL, test = FALSE){

   if (!is.null(y) & NROW(x) != NROW(y))
     stop("x and y must have the same
         number of observations")
   if (is.null(y) & NCOL(x) < 2)
     stop("supply both x and y or a
         matrix-like x")
   ncx <- NCOL(x)
   ncy <- NCOL(y)
   if (is.null(y)) {
     ok <- complete.cases(x)
     x <- x[ok, ]
   }
   else {
     ok <- complete.cases(x, y)
     if (ncx == 1) {
       x <- x[ok]
     }
     else {
       x <- x[ok, ]
     }
     if (ncy == 1) {
       y <- y[ok]
     }
     else {
       y <- y[ok, ]
     }
   }
   n <- NROW(x)
   if (n == 0) {
     warning("No observations (at least
         after removing missing values)")
     return(NULL)
   }
   ### Converting y to radians ###
   if (!is.null(y)) {
     y <- conversion.circular(y, units =
         "radians", zero = 0,
                        rotation =
                            "counter",
                            modulo = "2pi")
     attr(y, "class") <- attr(y,
         "circularp") <- NULL

     ### assigning ranks to theta's
     r_i = rank(y,ties.method = "average");
     data = data.frame(x,y,r_i);

   }

   if(is.null(y)){
     y = conversion.circular(x[,2], units =
         "radians", zero = 0,
                        rotation =
                            "counter", modulo
                            = "2pi");
     attr(y, "class") <- attr(y,
         "circularp") <- NULL;

   # Creating the rank circular correlation
       coeff #

       r_i = rank(y,ties.method = "average");
       data = data.frame(x=x[,1],y,r_i);

   }

   # sorted data set by X, ascending #
   newdata = data[order(data$x),];


   # calculating beta stats
   n = nrow(newdata);
   newdata$iteration = rank(newdata$x,
       ties.method = "average");
   newdata$beta = 2*pi*newdata$r_i/n;

   T_C =
       with(newdata,sum(iteration*cos(beta)));
   T_S =
       with(newdata,sum(iteration*sin(beta)));
   U = (24*(T_C^2 + T_S^2))/((n^2)*(n+1));

   # scaled correlation coefficient D_n
       falls between [0,1]


   if(n %% 2 == 0){
     a_n = 1/(1+5*(1/(tan(pi/n)^2)) +
         4*(1/(tan(pi/n)^4)))
   }else{
     a_n = 2*(sin(pi/n))^4 /
         ((1+(cos(pi/n)))^3)
   }

   D_n = a_n * ((T_C^2) + (T_S^2))

   if(test){

     p.val = pchisq(q = U,df = 2,lower.tail =
         FALSE);
     #rank.correlation is our U statistic14.
     #scaled.correlation = D statistic
     #p-value. U stat follows a Chi-Square
         with 2 degree of freedom. as n->
         infinity.
     result = list(rank.correlation = U,
                   scaled.correlation = D_n ,
                   p.value = p.val);
   }else{
     result = list(rank.correlation = U,
                   scaled.correlation = D_n);
   }
   return(result);
   }
```

Simulating the Parametric Linear-Circular Correlation Coefficient

```
rm(list=ls());
dir = "C:/Users/Robin/Dropbox/Circular
    Data/";
setwd(dir);

library("circular");
library("reshape2");
library("ggplot2");
library("gridExtra");
library(foreach); # parallel processing
library(doSNOW); # more parallel processing
library("parallel"); # for the # of cores

finaldata = NULL;
## Sample Sizes we simulated ##
samplesize = c(15,30,50,100,500);
trials=100000;

## Multicore stuff ##
numcores = detectCores();
cluster = makeCluster(numcores, type =
    "SOCK");
registerDoSNOW(cluster);

## actual simulation ##
finaldata = foreach(n =
    1:length(samplesize),.combine = cbind)
    %dopar% {
  library(circular);
  #getting the correlation coeff
  replicate(trials,
      cor.circular.lc(rnorm(samplesize[n],6,2),
      circular(runif(samplesize[n],0,2*pi),units
      = "radians"),TRUE)$statistic);

};
stopCluster(cluster);

#### Renaming Columns ####
colnames(finaldata)=paste0("sample.size.",
as.character(samplesize));

finaldata = as.data.frame(finaldata);

write.csv(finaldata,file =
    paste0(dir,"(parametric) Sim
    ",format(Sys.time(), "%a %b %d
    %Y"),".csv"));

#### Plotting ####

for(j in 1:length(samplesize)){
  ## Calculating F Distribution
  finaldata$f = df(finaldata[,j],
                df1 = 2,
                df2= samplesize[j]-3);


  ### Melting the data and use the density
      aesthetic for the chi-sq density
  melt.data =
      melt(finaldata[,c(j,length(finaldata))],id.vars
      = "f");

  ## Calculating the Kolmogorov Smirnov ##
  ks.result =
      ks.test(finaldata[,j],"pf",df1 = 2,
      df2 = samplesize[j]-3,alternative =
      "two.sided");

  ## Calculating the simulated p-value ##
  sim.p.value = sum(finaldata[,j]>qf(p =
      .95,df1 = 2, df2 =
      samplesize[j]-3,lower.tail =
      TRUE))/trials;


  #plot1 is the plot of the distribution of
      the Chi-square and overlaying it with
      the curve of the Chi-Sq(df = 2)
  plot1 = ggplot(data = melt.data,
      aes(f,value))+
    geom_histogram(aes(x=value,y=..density..),
        # Histogram with density instead of
        count on y-axis
                binwidth=.25,
                colour="black", fill="white")+
    stat_function(fun=function(x)
        df(x,2,samplesize[j]-3),col="blue");
        #overlaying with the curve of the
        chi-sq

  #plot2 is the normal probability plot of
      the data
  plot2 = qplot(sample =
      finaldata[,j],distribution =qf,
      dparams = list(df1 = 2, df2 =
      samplesize[j]-3))+
    geom_abline(aes(intercept=0,
        slope=1),colour = "red")+
    labs(title = paste("QQ Plot of n
        =",samplesize[j]))+
    annotate("text",
          x=5,
          y=round((min(finaldata[,j]))+3):(round(min(fi
          label = c(paste("KS Test Stat
              =",round(ks.result$statistic,digits
              = 5)),
                  paste("KS p.value
                      =",round(ks.result$p.value,digits=
                  paste("p > F_.95
                      =",sim.p.value),
                  paste("n =
                      ",samplesize[j])),
          hjust=0,
          colour = "#0033FF");
  #picture saving
  png(filename = paste("n
      =",samplesize[j],".png"),height=1024,width=1280,b
      = "transparent", antialias =
      "cleartype");
  grid.arrange(plot1,plot2,ncol = 2);
  dev.off();
}
```

Simulating the Nonparametric Linear-Circular Correlation Coefficient

```
rm(list=ls());
dir = "C:/Users/Robin/Dropbox/Circular
    Data/";
setwd(dir);

library("circular");
library("reshape2");
library("ggplot2");
library("gridExtra");
library(foreach); # parallel processing
library(doSNOW); # more parallel processing
library("parallel"); # for the # of cores
## Sample Sizes we simulated ##
samplesize = c(15,30,50,100,500,1000);
trials = 100000;
## Multicore stuff ##
numcores = detectCores();
cluster = makeCluster(numcores, type =
    "SOCK");
registerDoSNOW(cluster);

## actual simulation ##
finaldata = foreach(n =
    1:length(samplesize),.combine = cbind)
    %dopar% {
  library(circular);
  replicate(trials,
      cor.circular.lc.rank(runif(samplesize[n],0,1),
      circular(runif(samplesize[n],0,2*pi),units
      = "radians"),TRUE)$rank.correlation);
  #replicate(trials,
      cor.circular.lc.rank(rexp(samplesize[n],1),
      circular(rexp(samplesize[n],1),units =
      "radians"),TRUE)$rank.correlation);
  #replicate(trials,
      cor.circular.lc.rank(rexp(samplesize[n],1),
      circular(rnorm(samplesize[n],0,1),units
      = "radians"),TRUE)$rank.correlation);
  #replicate(trials,
      cor.circular.lc.rank(rnorm(samplesize[n],0,1),
      circular(rvonmises(samplesize[n],0,1),units
      = "radians"),TRUE)$rank.correlation);

  };

stopCluster(cluster);

#### Renaming Columns ####
colnames(finaldata)=paste0("sample.size.",
as.character(samplesize));

finaldata = as.data.frame(finaldata);

write.csv(finaldata,file =
    paste0(dir,"(nonparametric) Sim
    ",format(Sys.time(), "%a %b %d
    %Y"),".csv"));

#### Plotting ####

for(j in 1:length(samplesize)){

  ## Calculating Chi-Sq Distribution
  finaldata$chisq = dchisq(finaldata[,j],df
      = 2)

  ### Melting the data and use the density
      asthetic to take care of the chi-sq
      density instead
  melt.data =
      melt(finaldata[,c(j,length(finaldata))],id.vars
      = "chisq");

  ## Calculating the Kolmogorov Smirnov ##
  ks.result =
      ks.test(finaldata[,j],"pchisq",2,alternative
      = "two.sided");

  ## Calculating the simulated p-value ##
  sim.p.value = sum(finaldata[,j]>qchisq(p
      = .95,df = 2,lower.tail =
      TRUE))/trials;

  #plot1 is the dist. of the Chi-square
      overlayed with the Chi-sq(df = 2) curve
  plot1 = ggplot(data = melt.data,
      aes(chisq,value))+
    geom_histogram(aes(x=value,y=..density..),
        # Histogram with density instead of
        count on y-axis
                                 binwidth=.25,
                                 colour="black",
                                    fill="white")+
    stat_function(fun=function(x)
        dchisq(x,2),col="blue"); #overlaying
        with the curve of the chi-sq
  #plot2 is the normal probability plot
  plot2 = qplot(sample =
      finaldata[,j],distribution =qchisq,
      dparams = list(df = 2))+
    geom_abline(aes(intercept=0,
        slope=1),colour = "red")+
    labs(title = paste("QQ Plot of n
        =",samplesize[j]))+
    annotate("text",
          x=5,
          y=round((min(finaldata[,j]))+3):(round(min(fi
          label = c(paste("KS Test Stat
              =",round(ks.result$statistic,digits
              = 5)),
                  paste("KS p.value
                      =",round(ks.result$p.value,digits=
                  paste("p > Chisq_.95
                      =",sim.p.value),
                  paste("n =
                      ",samplesize[j])),
          hjust=0,
          colour = "#0033FF");
  #picture saving
   png(filename = paste("n
       =",samplesize[j],"(nonparametric).png"),height=10
       = "transparent", antialias =
       "cleartype");
   grid.arrange(plot1,plot2,ncol = 2);
   dev.off();
}
```

Simulating Ties

```
dir = "C:/Users/Robin/Dropbox/Circular
    Data/";
setwd(dir);
library("circular");
library("reshape2");
library("ggplot2");
library("gridExtra");
library(foreach); # parallel processing
library(doSNOW); # more parallel processing
library("parallel"); # for the # of cores

samplesize = c(15,30,50,100);

multiplefunction =
    function(linear,circular,test = FALSE){
  ## multiplefunction ##
  #  inputs:
  #        linear - a linear variable.
  #        circular - a circular variable
  #  Outputs:
  #        a vector containing the
      nonparametric correlation coefficient
      and the number of unique values
  #
  correlation =
      cor.circular.lc.rank(linear,
      circular,test)$rank.correlation;
  linear.ties =
      length(linear)-length(unique(linear));
return(c(correlation,linear.ties))
}
#### IN PARALLELL!!! WOOHOO ####
numcores = detectCores();
cluster = makeCluster(numcores, type =
    "SOCK");
registerDoSNOW(cluster);

finaldata = foreach(n =
    1:length(samplesize),.combine = rbind)
    %dopar% {
  library(circular);
  replicate(1000000,
      multiplefunction(round(runif(samplesize[n],0,10)),
      circular(runif(samplesize[n],0,2*pi),units
      = "radians"),TRUE));
  };
stopCluster(cluster);
finaldata = as.data.frame(t(finaldata))

#### Renaming Columns ####
finaldata = as.data.frame(finaldata);
for(cnames in 1:length(finaldata)){
  if(cnames %%2 == 1){
  colnames(finaldata)[cnames]=c(paste0("sample.size.",
  as.character(samplesize[ceiling(cnames/2)])));
  }else{
  colnames(finaldata)[cnames]=c(paste0("ties.",
  as.character(samplesize[ceiling(cnames/2)])));
  }
}
write.csv(finaldata,file = paste0(dir,"Ties
    Sim ",format(Sys.time(), "%a %b %d
    %Y"),".csv"));
#### Plotting ####
for(j in 1:(2*length(samplesize))){
  if(j%%2==1){
    data.location = ceiling(j/2);
  finaldata$chisq = dchisq(finaldata[,j],df
      = 2)

  ### Melting the data and use the density
      asthetic to take care of the chi-sq
      density instead
  melt.data =
      melt(finaldata[,c(j,length(finaldata))],id.vars
      = "chisq");

  ks.result =
      ks.test(finaldata[,j],"pchisq",2,alternative
      = "two.sided");
  sim.p.value = sum(finaldata[,j]>qchisq(p
      = .95,df = 2,lower.tail =
      TRUE))/100000;

  plot1 = ggplot(data = melt.data,
      aes(chisq,value))+
    geom_histogram(aes(x=value,y=..density..),
        # Histogram with density instead of
        count on y-axis
                binwidth=.25,
                colour="black", fill="white")+
    stat_function(fun=function(x)
        dchisq(x,2),col="blue")

  plot2 = qplot(sample =
      finaldata[,j],distribution =qchisq,
      dparams = list(df = 2))+
    geom_abline(aes(intercept=0,
        slope=1),colour = "red")+
    annotate("text",
          x=5,
          y=round((min(finaldata[,j]))+3):(round(min(fi
          label = c(paste("KS Test Stat
              =",round(ks.result$statistic,digits
              = 5)),
                  paste("KS p.value
                      =",round(ks.result$p.value,digits=
                  paste("p > Chisq_.95
                      =",sim.p.value),
                  paste("n =
                      ",samplesize[data.location])),
          hjust=0,
          colour = "#0033FF");
  message(paste("Printing", "n
      =",samplesize[data.location],".png"));
  png(filename = paste("n
      =",samplesize[data.location],".png"),height=1024,
      = "transparent", antialias =
      "cleartype");
  grid.arrange(plot1,plot2,ncol = 2);
  dev.off();
  }
  }
```

Power Simulation where X is Exponential (...and Normal)

X is Normal is commented out in this code. This function is really long because both the Nonparametric and Parametric Linear-Circular Correlation Coefficients were written out completely in the foreach{} loop.

```
#### parameters to change ####
samplesize = c(15,30,50,100,500);
sim.results = NULL;
sim.results.temp = NULL;
trials = 5000;
Powercurve = NULL;
mu.val = seq(from = 1,to = 20, by = .2);

######## NORMAL ######################
# crt.cor = function (dta,mu){
#  if((dta>0 && dta<(pi/2))){
#    rnorm(n = 1,mean = 20,sd = 1);
#  }else{
#    rnorm(n = 1,mean = mu,sd = 1);
#  }
# }
######## Exponential ##################

crt.cor = function (dta,mu){
  if((dta>0 && dta<(pi/2))){
    rexp(n = 1,rate=1/20);
  }else{
    rexp(n = 1,rate=1/mu);
  }
}

#####################################

## Multicore stuff ##
numcores = detectCores();
cluster = makeCluster(numcores, type =
    "SOCK");
registerDoSNOW(cluster);


for(j in 1:length(samplesize)){
   Powercurve = NULL;
   for(k in mu.val){
          sim.results = foreach(i =
              1:trials,.combine =
              rbind)%dopar%{
             #functions required.
             library("circular");
#              crt.cor = function (dta,mu){
#                if((dta>0 && dta<(pi/2))){
#                  rnorm(n = 1,mean = 20,sd =
    1);
#                }else{
#                  rnorm(n = 1,mean = mu,sd =
    1);
#                }
#              }
               crt.cor = function (dta,mu){
                 if((dta>0 && dta<(pi/2))){
                   rexp(n = 1,rate=1/20);
                 }else{
                   rexp(n = 1,rate=1/mu);
                 }
               }

cor.circular.lc.rank =
    function(x, y = NULL, test
    = FALSE){


  if (!is.null(y) & NROW(x) !=
      NROW(y))
    stop("x and y must have the
        same number of
        observations")
  if (is.null(y) & NCOL(x) < 2)
    stop("supply both x and y
        or a matrix-like x")
  ncx <- NCOL(x)
  ncy <- NCOL(y)
  if (is.null(y)) {
    ok <- complete.cases(x)
    x <- x[ok, ]
  }
  else {
    ok <- complete.cases(x, y)
    if (ncx == 1) {
      x <- x[ok]
    }
    else {
      x <- x[ok, ]
    }
    if (ncy == 1) {
      y <- y[ok]
    }
    else {
      y <- y[ok, ]
    }
  }
  n <- NROW(x)
  if (n == 0) {
    warning("No observations
        (at least after
        removing missing
        values)")
    return(NULL)
  }
  ### Converting y to radians
      ###
  if (!is.null(y)) {
    y <- conversion.circular(y,
        units = "radians", zero
        = 0,
                             rotation
                                 =
                                 "counter",
                                 modulo
                                 =
                                 "2pi")
    attr(y, "class") <- attr(y,
        "circularp") <- NULL

    ### assigning ranks to
        theta's
    r_i = rank(y,ties.method =
        "average");
```

```
  data = data.frame(x,y,r_i);

}

if(is.null(y)){
  y =
      conversion.circular(x[,2],
      units = "radians", zero
      = 0,
                      rotation
                          =
                          "counter",
                          modulo
                          =
                          "2pi");
  attr(y, "class") <- attr(y,
      "circularp") <- NULL;


  ### Creating the rank
      circular correlation
      coeff
  r_i = rank(y,ties.method =
      "average");
  data =
      data.frame(x=x[,1],y,r_i);
   }

#### sorted data set by X,
    ascending ####
newdata =
    data[order(data$x),];

#### calculating beta stats
n = nrow(newdata);
newdata$iteration =
    rank(newdata$x,
    ties.method = "average");
newdata$beta =
    2*pi*newdata$r_i/n;

T_C =
    with(newdata,sum(iteration*cos(beta)));
T_S =
    with(newdata,sum(iteration*sin(beta)));
U = (24*(T_C^2 +
    T_S^2))/((n^2)*(n+1));

#### scalled correlation
    coefficient D_n falls
    between [0,1]

if(n %% 2 == 0){
  a_n =
      1/(1+5*(1/(tan(pi/n)^2))
      + 4*(1/(tan(pi/n)^4)))
}else{
  a_n = 2*(sin(pi/n))^4 /
      ((1+(cos(pi/n)))^3)
}

D_n = a_n * ((T_C^2) +
    (T_S^2))

if(test){
    p.val = pchisq(q = U,df =
        2,lower.tail = FALSE);
    #rank.correlation is our U
        statistic14.
    #scaled.correlation = D
        statistic
    #p-value. U stat follows a
        Chi-Square with 2
        degree of freedom. as
        n-> infinity.
    result =
        list(rank.correlation =
        U,
               scaled.correlation
                   = D_n ,
               p.value = p.val);
  }else{
    result =
        list(rank.correlation =
        U,
               scaled.correlation
                   = D_n);
  }
  return(result);
}
cor.circular.lc =
    function(x,y=NULL,test =
    FALSE){
  ### x   vector or matrix of
      linear data
  ### y   vector or matrix of
      circular data
  ### test if test == TRUE then
      a significance test for
      the correlation is
      computed

  if (!is.null(y) & NROW(x) !=
      NROW(y))
    stop("x and y must have the
        same number of
        observations")
  if (is.null(y) & NCOL(x) < 2)
    stop("supply both x and y
        or a matrix-like x")
  ncx <- NCOL(x)
  ncy <- NCOL(y)
  if (is.null(y)) {
    ok <- complete.cases(x)
    x <- x[ok, ]
  }
  else {
    ok <- complete.cases(x, y)
    if (ncx == 1) {
      x <- x[ok]
    }
    else {
      x <- x[ok, ]
    }
    if (ncy == 1) {
      y <- y[ok]
    }
    else {
      y <- y[ok, ]
```

```
  }
}
n <- NROW(x)
if (n == 0) {
  warning("No observations
      (at least after
      removing missing
      values)")
  return(NULL)
}
### Converting y to radians
    ###
if (!is.null(y)) {
  y <- conversion.circular(y,
      units = "radians", zero
      = 0,
                       rotation
                           =
                           "counter",
                           modulo
                           =
                           "2pi")
  attr(y, "class") <- attr(y,
      "circularp") <- NULL
}
if(is.null(y)){
  z =
      conversion.circular(x[,2],
      units = "radians", zero
      = 0,
                      rotation
                          =
                          "counter",
                          modulo
                          =
                          "2pi");
  attr(z, "class") <- attr(z,
      "circularp") <- NULL;
  r_xs = cor(x[,1],sin(z));
  r_xc = cor(x[,1],cos(z));
  r_cs = cor(cos(z),sin(z));
}else{

  ### calculating individual
      components ###
  r_xs = cor(x,sin(y));
  r_xc = cor(x,cos(y));
  r_cs = cor(cos(y),sin(y));
}
### calculating correlation
    coeff linear-circular ###
cor.lc = (r_xc^2 + r_xs^2 -
    2*(r_xc*r_xs*r_cs))/(1-r_cs^2);

if(test){
  f.stat =
      (.5*(n-3)*cor.lc)/(1-cor.lc);
  p.val = pf(f.stat,df1 = 2,
      df2= n-3,lower.tail =
      FALSE);
  result = list(cor = cor.lc,
      statistic = f.stat,
      p.value = p.val);

}else{
      result = list(cor = cor.lc);
    }
    return(result);
  }

  #simulating uniform around the
      circular than applying
      crt.cor
  dta =
      circular(runif(samplesize[j],0,2*pi),un
      = "radians")
  lin.dta =
      sapply(dta,function(x)
      crt.cor(x,k));

  sim.results.temp =
      c(cor.circular.lc(lin.dta,dta,test=TRUE
                  cor.circular.lc.rank(lin.dt
}

  nonpar.power.val =
      sum(sim.results[,2]<=.05)/trials;#nonparametri
      correlation
  par.power.val =
      sum(sim.results[,1]<=.05)/trials;#parametric
      correlation

 #rbinding the values power values from
     each sample size #
 if(k != 1){
  Powercurve =
      rbind(Powercurve,c(k,nonpar.power.val,
      par.power.val))
 }else{
  Powercurve = c(k,nonpar.power.val,
      par.power.val);
 }
message(paste0("sample size
    ",samplesize[j], ", mu value ",k))
 }



Powercurve = data.frame(Powercurve);
colnames(Powercurve) = c("k",
    "nonpar.power", "par.power");

#### Melting data for a stacked form or
    "long" form. the way ggplot likes it
    ####
stacked.Powercurve = melt(Powercurve, id
    = "k");

#### Outputting a png file ####
message(paste0("printing power curve for
    sample size ", samplesize[j]));


someplot = ggplot(data =
   stacked.Powercurve,aes(x=k,y=value,
   colour = variable))+
    geom_line()+
    labs(title = paste("Power Curve n
        =",samplesize[j]))+
```

```
    xlab("Values of mu for Exp(1/lambda)
        from [pi/2,2pi]");
  png(filename = paste("Power Study -
      sample size
      =",samplesize[j],".png"),height=1024,width=1280,bg
      = "transparent", antialias =
      "cleartype");
   grid.newpage();
   print(someplot+theme_gray(base_size=12*(1024/1280)))
 dev.off();

}


stopCluster(cluster);
```

Circular Data Sets Used for Code Verification

```
#### Testing the Parametric Linear-Circular
    Correlation

# Measurments of Ozone Concentration (x)
    and Wind Direction (theta)
x=c(28, 85.2, 80.5,4.7,45.9,12.7,72.5,
    56.6, 31.5, 112, 20, 72.5, 16,45.9,
    32.6, 56.6, 52.6, 91.8,55.2);
theta =
    c(327,91,88,305,344,270,67,21,281,8,204,86,333,18,57,6,11,27,84);


#### Testing the Nonparametric
    Linear-Circular Correlation
# Even n = 8
x = c(1.5,1.6,1.7,2.0,2.1,1.8,1.4,1.2);
theta = c(30,100,120,170,240,260,300,330);

# Possible odd
x=c(14,13.41421,12);
theta=c(0,45,90);

# Circannual distribution of simian births
    at different latitudes (Gauqelin 1968).
    n =5;
#U_n=.390, D_n = .098, R = .974

x = c(49,36,34,33.5,18);
theta = c(286,183,164,188,95);
```