**Online Appendices to Chapter 23**

# Table of contents

Online Appendix 23.A: Roe-Metz simulator for single modality CAD vs. radiologists

Online Appendix 23.B: Meanings of terms in simulator model

Online Appendix 23.C: Calibrating the simulator to a specific dataset

Online Appendix 23.D: Validation of the RM simulation and NH behavior

Online Appendix 23.E: References

# Online Appendix 23.A: Roe-Metz simulator

Let *t* denote the truth index, and  disease-free cases, corresponding to *t* = 1, be indexed by  (=1, 2,…, ) and  diseased cases, corresponding to *t* = 2, be indexed by  (=1, 2,…, ). Let *j* denoted a generic reader index, where *j* = 0 denotes the CAD algorithmic "reader" and *j* = 1, 2, …, *J* denote *J* radiologists interpreting the same cases as CAD. The *Z*-sample (i.e., the ROC rating) for a generic (CAD or radiologist) reader is denoted by . The decision variable for the CAD algorithmic "reader" denoted  is modeled as (the absence of an error term is explained following Eqn. ):

 .

The *z* superscripts are used to distinguish quantities at the ratings level from similar variables at the figure of merit level, e.g., in **Chapter 09**. The constant terms  are defined by:

 .

The random variable, corresponding to case , in Eqn. , is sampled according to:

 .

To summarize, CAD ratings are modeled by sampling two unit-variance normal distributions separated by . Since interest is in comparing a *specific* CAD system to a group of radiologists interpreting the same cases, variations between different CAD systems do not need to be accounted for. Furthermore, while a specific CAD system will give different results when applied repeatedly to the same case1-3, this source of algorithmic variability is usually much smaller than case-to-case ratings variability of a CAD system. For this reason an additional error term is not included in Eqn. .

The parameter  determines the true performance of the CAD system. The area under the equal-variance binormal model ROC curve is defined by4,5 and see **Chapter 03**:

 .

The decision variable  for radiologists (*j* = 1, 2,…, *J*) is modeled as:

 .

where,

 .

The random variables in Eqn. are sampled as follows:

 .

It is shown below that  is the expected Z-sample correlation between CAD and any of the human observers, i.e.,

 . .

**Proof:** For any two random variables A and B, the following theorem holds6:

 . .

If the random variables have the same variance, it follows that:

 . .

Since both random variables have the same (unit) variance, it follows, using Eqn. and Eqn. , that (the sampling is regarding cases as random, so the reader term does not contribute to the variance):

 .

****

Since the z-samples are latent variables, the correlation cannot be directly measured from the ratings. An indirect estimation procedure is described below.

# Online Appendix 23.B: Meanings of terms in simulator model

The term  in Eqn. is identical to that in the CAD sampling model, Eqn. . By itself, it would yield reader averaged performance identical to CAD. The term  models the increase in separation between the two distributions due to performance differences between CAD and the average of the radiologists. If this term is positive, then the radiologists as a group outperform CAD. The term  models between-radiologist variability: it is modeled as a sample from a zero-mean normal distribution with variance ; when this term increases the radiologists are expected to exhibit greater spread of  values. True  of the radiologist population is , i.e. that corresponding to two unit-variance normal distributions separated by  (the separation of the two unit-variance normal distributions for the *j*th radiologist is , whose expectation is ).

The term , in Eqn. , is a scaled replica of the random sample  previously obtained for CAD, Eqn. . It accounts for correlations between CAD and radiologist ratings. The final term accounts for the uncorrelated part of the radiologist ratings. The scaling parameter , where , controls the correlation; for example, if  then  and the case-dependent random components to the CAD and radiologist ratings become identical, leading to perfect correlation. Conversely, = 0 leads to zero correlation.

It can be seen from Eqn. that for any specific radiologist the net case variance is unity. This maintains the unit-variance binormal sampling model for each radiologist, which in turn preserves the interpretation of  as an index of performance directly related to .

# Online Appendix 23.C: Calibrating the RM simulator to a specific dataset

The next step is to implement the calibration procedure. Listed below are the first 25 lines of mainSimulatorCalibration.R.

## Online Appendix 23.C.1: Code Listing

rm(list = ls()) # mainSimulatorCalibration.R

library(RJafroc)

source("ReadNicoData.R")

source("RhoAvgRdrVsCAD.R")

source('CombineSamples.R')

source('Wilcoxon.R')

source("diffCorrelations.R")

seed <- NULL;set.seed(seed)

alpha <<- 0.05

RADIOLOGISTS <- TRUE # if FALSE, all readers are included

residents <- c(6,7,10) # as per Nico communication

retNico <- ReadNicoData(RADIOLOGISTS, residents)

zjk1 <- retNico$zjk1;zjk2Cl <- retNico$zjk2Cl;zjk2Il <- retNico$zjk2Il;zjk2 <- pmax(zjk2Cl,zjk2Il)

K1 <- length(zjk1[1,]);K2 <- length(zjk2Cl[1,]);K <- c(K1,K2);J <- length(zjk1[,1]) - 1

thetajc <- array (dim = (J+1));for (j in 1:(J+1)) thetajc[j] <- Wilcoxon(zjk1[j,],zjk2[j,])

mu <- sqrt(2)\*qnorm(thetajc[1]);cat("mu = ", mu, "\n") # mu applies to CAD performance

muHum <- array(dim = J);for (j in 2:(J+1)) muHum[j-1] <- sqrt(2)\*qnorm(thetajc[j]) # human observers

deltaMu <- mean(muHum) - mu; cat("deltaMu = ", deltaMu, "\n")

VarR <- var(muHum); cat("VarR = ", VarR, "\n") # this determines the reader variance term

retRho <- RhoAvgRdrVsCAD(zjk1, zjk2)

targetRho <- retRho$rho

cat("Avg thetajc correlation between human rdrs and CAD", targetRho, "\n")

...

...

At line 9, setting seed to NULL means "start with a random seed every time" (usually generated from the current time). Since results from several runs of the program will be averaged, involving different case samples, one does not wish to get the same results every time. Lines 10 – 15 are similar to the examples in the previous chapter: the Hupse-Karssemeijer LROC dataset is read, converted to ROC data and the empirical ROC-AUC thetajc is computed for each reader, *j* = 0, 1, 2,..., *J*, where *j* = 0 corresponds to CAD and the rest correspond to the *J* = 9 radiologists.

The next step is to determine  appearing in Eqn. , deltaMu in the code. In **Chapter 22**, Table 22.3, the difference in empirical ROC-AUCs, i.e., reader averaged minus CAD, was determined to be 0.0317. *Do not make the mistake of thinking = 0.0317*. The value 0.0317 is the difference of *AUC* values. Needed instead is the separation of the unit-normal distributions yielding the observed AUC difference. Therefore, one needs to convert CAD and each radiologist’s AUC to an equal-variance binormal-model *separation parameter*, average the nine radiologist’s separation parameters and subtract the corresponding value for CAD.

Line 17 converts AUC for CAD to the separation parameter  for CAD (mu in the code). It implements the following equation, which is the inverse of Eqn. (qnorm()is the inverse of the pnorm() function):

 . .

Line 18 applies the same transformation to each radiologist's AUC and line 19 calculates the difference deltaMu, i.e., , between the CAD value and the average of the radiologists' values. Line 20 calculates the variance of the separation parameters for each radiologist. This is the between-radiologist variance term in Eqn. , i.e., (varR in the code).

Insert a break point at line 22 and click Source. The output is shown below (lines generated by the debugger are not shown):

## Online Appendix 23.C.2: Code Output

> debugSource('~/book2/05 E Advanced Topics/E3 CAD vs. radiologists/Software/mainSimulatorCalibration.R')

mu = 1.278048

deltaMu = 0.1904987

VarR = 0.03377221

Three of the calibration parameters have been determined, namely  = 1.28, ** and = 0.0338.

It remains to determine (whose *square* is named varC in the code). Recall that  is the scaling factor applied to the CAD ratings, Eqn. , representing the correlated part of the radiologist ratings. *If tempted to estimate this correlation by calculating the ratings-level correlation between CAD and each radiologist, and then averaging the nine resulting correlations, stop!* As emphasized repeatedly, working directly with ratings is dangerous, as they are subject to arbitrary monotonic increasing transformations without affecting the FOMs.

Instead, one calculates the average figure of merit correlation between CAD and the radiologists (called targetRho in the code) for the original dataset and adjusts varC until the desired value is reached. Calculation of the average figure of merit correlation is implemented in function RhoAvgRdrVsCAD(). It calculates the Pearson correlation coefficient between jackknife FOM values for each radiologist and CAD and returns the average. [To see this at work one clicks on the "*enter into function*" debug button which takes one to the listing in **Error! Reference source not found.**, and one keeps clicking Next to see the entire procedure at work; when satisfied, click the "get out of function" button to get back to the main level code.] Click Next a few times to advance the code pointer to line 25. The following additional output is produced.

## Online Appendix 23.C.3: Code Output

Browse[2]> n

Avg fom correlation between actual human rdrs and CAD 0.2784757

[The "n" in the output is the echoed Next click] Line 22 calculates the correlation between CAD and each radiologist using the jackknife technique and returns the average correlation, 0.278, named targetRho, as this is the value the simulator will attempt to reach by varying varC.

The procedure for finding varC is a one-dimensional search implemented by an R function called uniroot() that looks for the zero crossing of a supplied function. Using a starting (guessed) value of the unknown parameter varC and Eqn. and Eqn. , the simulation model, one has enough information to simulate a new dataset. To minimize sampling variability one simulates J = 40 radiologists, plus the CAD algorithmic "reader", interpreting 120 non-diseased and 80 diseased cases (one should experiment with more cases; the final value of varC is not expected to be sensitive to these choices). One calculates the average figure of merit correlation between CAD and the radiologists for this new dataset (currentRho in the code). One iteratively adjusts varC to make the *difference between the two correlations* (targetRho and currentRho) as close to zero as possible. The corresponding value of varC is the desired calibration value. The function returning the difference between the two correlations is named diffCorrelations(),**Error! Reference source not found.**, which is the function whose zero crossing is searched for by uniroot().

Here is the idea: one simulates new datasets for different values of varC, calculates the average correlation each time between FOMs of CAD and the radiologists, and "homes in" on the value of varC yielding correlation very close to the target value, targetRho. However, before doing this there is a practical problem. If one simulates *fresh* datasets each time varC is varied, sampling variability will introduce jitter, tending to mask the optimal value of varC. Stated differently, even for the same value of varC, because the random samples will have changed, the function diffCorrelations() will return different values. This will confuse uniroot(), which expects the function, whose zero-crossing is desired, to be continuous. To get around this problem, line 27 - 37 pre-samples- in one fell swoop - the requisite number of *unit variance* random variables – corresponding to the reader, case and epsilon terms. At line 26, nSimu, defined to be 10, is the number of independent datasets over which the correlation between radiologists and CAD will be averaged. So each simulated dataset consists of 40 radiologists, 120 non-diseased and 80 diseased cases, and there are 10 such datasets.

Lines 26– 30 create arrays with names prefixed with Pre, to denote the pre-sampled values. Lines 33 – 36 fill in the arrays with samples from N(0,1). At this point one *does not multiply* by the square roots of varC and (1-varC); for one thing, at this point varC is undefined in the code (try printing it); the root finding function uniroot() passes trial values of varC to a function whose zero-crossing one desires, named diffCorrelations(). This function returns the difference between the correlations for the current simulated dataset and the target value, i.e., targetRho – currentRho. To get the current simulated dataset it needs the combined z-samples, for which purpose it calls the function CombineSamples(), **Error! Reference source not found.**, with trial values of the variances which get multiplied *inside* the called function. It looks complicated, but with dedication and usage of the debug ability of RStudio, one should be able to figure out what is going on.

Mathematically, the algorithm for determining varC is as follows. Define the average correlation  between FOM of CAD and each reader as follows. The value of the correlation depends on the choice of . For example, if , the correlation will be unity and if , the correlation will be zero (the model can only simulate positive correlations).

 . .

The desired value is that which minimizes the difference between  and the target correlation, in other words, the value of  that solves the following equation:

 . .

Exit debug mode (red square Stop button in Console window), remove any enabled breakpoints, insert a breakpoint at line 26 and click on Source. A listing of the rest of the file mainSimulatorCalibration.R, starting with line 27, is shown below.

## Online Appendix 23.C.4: Code Listing

...

...

J <- 40;K <- c(120,80);nSimu <- 10

PreCasektCad <- array(dim=c(nSimu, max(K),2)) # last index is case truth

PreRjt <- array(dim = c(nSimu,J,2))

PreCasektCad <- array(dim = c(nSimu,max(K),2))

PreCasektHum <- array(dim = c(nSimu,max(K),2))

PreEpsjkt <- array(dim = c(nSimu,J,max(K),2))

for (sim in 1:nSimu) {

PreRjt[sim,,] <- rnorm( 2\*J, sd = sqrt(VarR) )

PreCasektCad[sim,,] <- rnorm( 2\*max(K))

PreCasektHum[sim,,] <- PreCasektCad[sim,,] # dont use VarC here; get N(0,1) samples; multiply by VarC factor inside zero-crossing function

PreEpsjkt[sim,,,] <- rnorm( J\*2\*max(K)) # dont use 1-VarC here; get N(0,1) samples; multiply by 1-VarC factor inside zero-crossing function

}

Line 39 uses the function uniroot(), which implements finding the *root* of a function of a single argument (a root is defined as the value of the argument at which the function evaluates to zero). The uniroot() function takes as its 1st argument the *name* of the function whose root one desires, i.e., diffCorrelations; the 2nd argument is an interval that (hopefully) brackets the desired root (if not, an error message will be generated and the program will come to a grinding halt). In the code interval = c(0.05,0.15), which was found by trial and error. This means that only values of varC that are inside the interval (0.05, 0.15) are passed to the zero-crossing function. Other variables, described below, that need to be passed to diffCorrelations(), are supplied after the interval variable.

Starting with the supplied end-point values in interval, the uniroot() function repeatedly calls diffCorrelations(), with different values of VarC, until the magnitude of the returned value is smaller than a pre-set tolerance value (consult the help page for the uniroot() function for details on this function).

Exit debug mode, remove any break points and Source the file mainSimulatorCalibration.R, and wait a while (blank lines have been removed in the following output):

## Online Appendix 23.C.5: Code Output

> source('~/book2/05 D Advanced Topics/D23 Validate CAD analysis/software/mainSimulatorCalibration.R')

mu = 1.278048

deltaMu = 0.1904987

VarR = 0.03377221

Avg thetajc correlation between human rdrs and CAD 0.2784757

current VarC = , ret = 0.05 -9.892419

current VarC = , ret = 0.15 3.646945

current VarC = , ret = 0.1230641 1.447377

current VarC = , ret = 0.08653207 -2.872897

current VarC = , ret = 0.1108252 0.1994605

current VarC = , ret = 0.108996 -0.0009733256

current VarC = , ret = 0.109057 0.006769786

current VarC = , ret = 0.108996 -0.0009733256

Final (optimized) value of VarC parameter = 0.108996

$root

[1] 0.108996

$f.root

[1] -0.0009733256

$iter

[1] 5

$init.it

[1] NA

$estim.prec

[1] 6.103516e-05

Your values will be different; the relevant value is preceded by $root, is (0.108996); on 11 runs I have obtained the following values, with mean and twice standard deviations (95% CI) shown below (see commented lines 44-45).

## Online Appendix 23.C.6: Code Snippet

> values <- c(0.1087649, 0.1071889, 0.1046044, 0.0995922, 0.1094468, 0.1036791, 0.1094728, 0.09862758, 0.1133044, 0.1089707, 0.108996)

> cat("mean = ", mean(values), ", 2 x std = ", 2\*sqrt(var(values)), "\n")

mean = 0.1066043 , 2 x std = 0.009008518

One is done with calibration of the simulator! Table 23.4.1 summarizes the parameters of the simulator.

Table 23.4.1: Parameter values for CAD-human observer simulator matched to the dataset in Hupse-Karssemeijer.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Name in code | Value | Meaning |
|  | mu | 1.278 | Separation parameter of unit variance distributions signifying performance of CAD |
|  | deltaMu | 0.190 | Incremental separation parameter of radiologists as a group over CAD |
|  | VarR | 0.0338 | Variance of radiologists |
|  | VarC | 0.107 ± 0.009 | Case variance of radiologists that is common to CAD |

# Online Appendix 23.D: Validation of the simulation and analysis method

To validate the analysis method one repeatedly simulates independent ROC ratings data for a CAD system and a group of radiologists interpreting the same cases and typically, one uses 2000 simulations. The code for this is in file mainNhTesting.R. The listing of this file and the explanation are in Appendix A3. If you source the file after about 10 minutes, depending on your computer, the program should complete and you should see the rejection fraction printed out. The program, with useCluster set to TRUE, uses the parallel computing ability of R. In repeated runs, the following rejection rates were observed: 0.0465, 0.06, 0.0545, 0.055 and 0.059, 0.0515 for the rejection fractions, which are all within the expected 95% confidence interval assuming the true value is 0.05. So the method passes the null hypothesis test. If one repeats this many times, one expects to see see an occasional NH failure (with probability 5%). For an example of a well-conducted validation study see Ref. 7. The code output is shown below:

### Online Appendix 23.D.1: Code output

> source('~/book2/05 D Advanced Topics/D23 Validate CAD analysis/software/mainNhTesting.R')

foreach: simple, scalable parallel programming from Revolution Analytics

Use Revolution R for scalability, fault tolerance and more.

http://www.revolutionanalytics.com

Loading required package: iterators

Loading required package: parallel

Loading required package: rngtools

Loading required package: pkgmaker

Loading required package: registry

Attaching package: ‘pkgmaker’

The following object is masked from ‘package:base’:

isNamespaceLoaded

AVG Var = 0.001328572

AVG Cov2 = 0.0004852261

reject fraction = 0.0515

Looking at the last three lines, listed first is the averaged Var followed by the average Cov2, where these are the two Obuchowski-Rockette FOM covariance parameters needed to describe the error term in a single modality multiple reader study. These should be compared to the values for the original dataset, §22.3.2, namely 0.0014 and 0.000924. The variance value for the simulator is close, but the covariance term is about half the expected value. The final listed value is the observed NH rejection rate.

# Online Appendix 23.E: Simulator calibration helper functions

## Online Appendix 23.E.1: Function to calculate average correlation between readers and CAD

RhoAvgRdrVsCAD <- function (zjk1, zjk2)

{

J <- length(zjk1[,1]) - 1 # number of radiologists

K1 <- length(zjk1[1,]);K2 <- length(zjk2[1,]);K <- K1 + K2

CADjk = array(dim = c(K))

RDRjk = array(dim = c(J,K))

for (k in 1:K) {

if (k <= K1) {

zjk1jk <- zjk1[,-k]

zjk2jk <- zjk2

} else {

zjk1jk <- zjk1

zjk2jk <- zjk2[,-(k-K1)]

}

temp <- WilcoxonMR(zjk1jk, zjk2jk)

CADjk[k] <- temp[1]

RDRjk[,k] <- temp[2:(J+1)]

}

rho <- array(dim = J)

for (j in 1:J) {

rho[j] <- cor(CADjk,RDRjk[j,], method = "pearson")

}

rho <- mean(rho)

return (list (

rho = rho

))

}

This function takes two arguments zjk1, zjk2, the ROC ratings of the non-diseased and diseased cases, respectively. Line 3 extracts the number of radiologists, which is one less than the length of the first dimension of zijk1. Line 4 extracts the variables K1, K2 and initializes K, the total number of cases. Line 6 initializes the array CADjk[1:K] with NAs; this will hold the jackknife figure of merit values for CAD. Line 7 initializes the array RDRjk[1:J,1:K] with NAs; this will hold the jackknife figure of merit values for the radiologists. Line 8-19 implements the jackknife, which code should be familiar by now. Line 16 calculates and saves to temp the jackknife figure of merit array of length J+1, the first element of which is the jackknife figure of merit corresponding to deleted case k for CAD, and line 17 extracts this value and line 18 extract the jackknife figures of merit corresponding to deleted case k for the J radiologists. Line 21-25 calculates the correlation between the jackknife figures of merit for CAD vs. each of the J radiologists, and averages them. [One could have used jackknife pseudovalues, but the result would be the same; also note that there is no need for the jackknife inflation factor in these calculations.]

## Online Appendix 23.E.2: Function for difference of correlation from target value

diffCorrelations <- function (VarC, K, nSimu, targetRho, mu = mu, deltaMu = deltaMu, PreRjt = PreRjt,

PreCasektCad = PreCasektCad, PreCasektHum = PreCasektHum, PreEpsjkt = PreEpsjkt)

{

J <- dim(PreRjt)[1]

rhoFom <- array(dim = c(nSimu))

for (sim in 1:nSimu) {

Rjt <- PreRjt[sim,,]

CasektCad <- PreCasektCad[sim,,]

CasektHum <- PreCasektHum[sim,,]\*sqrt(VarC)

Epsjkt <- PreEpsjkt[sim,,,]\*sqrt(1-VarC)

z <- CombineSamples (J, K, mu, deltaMu, Rjt, CasektCad, CasektHum, Epsjkt)

zjk1 <- z$zjk1;zjk2 <- z$zjk2

retRho <- RhoAvgRdrVsCAD(zjk1, zjk2)

rhoFom[sim] <- retRho$rho

}

currentRho <- mean(rhoFom)

ret <- (currentRho-targetRho)/sqrt(var(rhoFom))

cat("current VarC = , ret = ", VarC, ret, "\n")

return(ret)

}

Line 4 extracts the number of radiologists J = 40. The for loop executes nSimu = 10 times, each time fetching (lines 7 - 10) fresh set of pre-simulated data Rjt =, CasektCad = , CasektHum =  and Epsjkt = , for 40 radiologists plus CAD, 120 non-diseased and 80 diseased cases. Line 11 combines the random samples after multiplying them with the necessary standard deviations using the function CombineSamples() described below. The latter takes as arguments J, ,,and the random samples , ,  and  and combines them to yield the ratings. Line 13 calculates the correlation using the function RhoAvgRdrVsCAD() described above. Line 16 averages the nSimu = 10 correlations and assigns it to the variable currentRho. Line 17 calculates the return value ret of the function, defined as (currentRho - targetRho) divided by the standard deviation of the correlations. Line 18 lists the current values of VarC and ret: with longer simulations, it is wiser to periodically check the progress of the algorithm, instead of staring at a blank screen.

## Online Appendix 23.E.3: Function to combine the random variable after including variances

CombineSamples <- function( J, K, mu, deltaMu, Rjt, CasektCad, CasektHum, Epsjkt)

{

muArr <- c(0, mu)

deltaMuArr <- c(0, deltaMu)

zjkt <- array(dim=c(J+1,max(K),2))

zjkt[1,1:K[1],1] <- CasektCad[1:K[1],1]

zjkt[1,1:K[2],2] <- CasektCad[1:K[2],2] + muArr[2]

for (j in 2:(J+1)) {

for (t in 1:2) {

for (k in 1:K[t]) {

zjkt[j,k,t] <- muArr[t] + deltaMuArr[t] + Rjt[j-1,t] + CasektHum[k,t] + Epsjkt[j-1,k,t]

}

}

}

zjk1 <- zjkt[,1:K[1],1]

zjk2 <- zjkt[,1:K[2],2]

return( list(

zjk1 = zjk1,

zjk2 = zjk2)

)

}

Line 3 constructs the variable muArr, which stands for , line 4 constructs the variable deltaMuArr which stands for  (this array actually has zeroes but is included for generality) and line 5 creates a NA initialized array zjkt with dimensions J+1, 2 and max(K) to hold the z-samples. The additional reader is to accommodate the CAD ratings (the first radiologist is CAD, the second radiologist is the first radiologist, etc.). Line 7 initializes the CAD ratings for non-diseased cases, while line 8 does the same for the diseased cases and adds  (the random samples were all obtained by sampling zero mean normal distributions; here is where we include the shift between the non-diseased and diseased cases; the non-diseased samples are from a zero mean unit normal distribution). Line 10-16 implements Eqn. 5-7 for the radiologists. There are three for-loops, one corresponding to the radiologist, starting with j = 2, the other for the two states of truth and the case loop, where the number of times it cycles through the case loop is determined by the appropriate element of the K vector. Line 18 extracts the correct number of elements corresponding to each index and assigns the result to the array zkj1, corresponding to the non-diseased cases. Line 19 does similar steps to get the array zkj2, corresponding to the diseased cases.

# Online Appendix 23.F: Code to perform NH testing

## Online Appendix 23.F.1: Code listing

rm(list = ls()) # MainNhTesting.R

library(foreach) # this code is to speed up the loop using parallel processing techniques

library(doParallel) # defer questions to xuetong zhai

library(doMC)

library(doRNG)

seed <- NULL;set.seed(seed)

source("GenerateSamples.R")

source("CombineSamples.R")

source('Wilcoxon.R')

source("DiffFomSingleModalityAnal2007Hillis53.R")

alpha <- 0.05

J <- 9;K <- c(80,120)

mu <- 1.278;deltaMu <- 0;VarR <- 0.0338;VarC <- 0.107

FOM <- "AUC";FPFValue <- 0.0

NSimu <- 2000

useCluster <- TRUE

arrayVar <- array(dim = NSimu)

arrayCov2 <- array(dim = NSimu)

if (useCluster){

registerDoMC(cores = detectCores())

clusterRet <- foreach(s = 1:NSimu, .combine = "rbind") %dorng% {

ret <- GenerateSamples ( J, K, VarR, VarC)

Rjt <- ret$Rjt

CasektCad <- ret$CasektCad

CasektHum <- ret$CasektHum

Epsjkt <- ret$Epsjkt

ret <- CombineSamples(J, K, mu, deltaMu, Rjt, CasektCad, CasektHum, Epsjkt)

zjk1 <- ret$zjk1;zjk2 <- ret$zjk2

ret\_nh <- DiffFomSingleModalityAnal2007Hillis53 (alpha, zjk1, zjk2, FOM, FPFValue)

if(ret\_nh$p\_val <= alpha) {

return(c(1, ret\_nh$Var, ret\_nh$Cov2))

}else{

return(c(0, ret\_nh$Var, ret\_nh$Cov2))

}

}

arrayRej <- clusterRet[, 1]

arrayVar <- clusterRet[, 2]

arrayCov2 <- clusterRet[, 3]

reject <- sum(arrayRej)

}else{

reject <- 0

for (s in 1:NSimu) {

ret <- GenerateSamples ( J, K, VarR, VarC)

Rjt <- ret$Rjt

CasektCad <- ret$CasektCad

CasektHum <- ret$CasektHum

Epsjkt <- ret$Epsjkt

ret <- CombineSamples(J, K, mu, deltaMu, Rjt, CasektCad, CasektHum, Epsjkt)

zjk1 <- ret$zjk1;zjk2 <- ret$zjk2

ret\_nh <- DiffFomSingleModalityAnal2007Hillis53 (alpha, zjk1, zjk2, FOM, FPFValue)

arrayVar[s] <- ret\_nh$Var

arrayCov2[s] <- ret\_nh$Cov2

if(ret\_nh$p\_val <= alpha) reject <- reject + 1

}

}

cat("AVG Var = ", mean(arrayVar), "\n")

cat("AVG Cov2 = ", mean(arrayCov2), "\n")

cat("reject fraction = ", reject/NSimu, "\n") # 0.0465 # 0.06 # 0.0545 # 0.055 # 0.059 # 0.0515

# # you can update the rhs of values as additional simulations are conducted

# AVG Var = 0.001328572

# AVG Cov2 = 0.0004852261 # this always comes out low; why???

The program uses parallel processing techniques to speed up the calculations. However, if one sets useCluster to FALSE the simpler but longer running code, between lines 43 and 55, is used, and these lines are explained below.

Line 14 defines J, the number of radiologists, to be 9, and each case-set has 120 non-diseased and 80 diseased cases, just as in the clinical study. Line 15 defines the parameters appearing in the simulation model Eqn. and Eqn. , which are listed in Table 1. *To enforce the null hypothesis conditionthe parameter (*deltaMu in the code*) has been deliberately set to zero*. Line 17 sets NSimu to 2000. Line 43 initializes the reject counter variable to zero. Line 44 sets up a for-loop to conduct the 2000 simulations. Line 45 uses the GenerateSamples() function, Appendix TBA, to generate the random samples according to the simulator equations, Eqn. and Eqn. . Lines 44-47 extract the different random components from the list variable ret. These are passed as arguments to CombineSamples(), Appendix TBA, which returns the ratings, zjk1 and zjk2, corresponding to non-diseased and diseased cases, respectively. Line 51 uses the Hillis single modality analysis function DiffFomAnal2007Hillis53() and line 52 extracts the p-value, and if it is smaller than 0.05, the counter reject is incremented by one. Finally, after all the simulations have been conducted, line 56 lists the rejection fraction.

## Online Appendix 23.F.2: Function to generate random samples

The function GenerateSamples() takes as arguments the number of radiologist J, the array  (K in the code), which contains the numbers of non-diseased and diseased cases, respectively, and the variances VarR and VarC and returns the samplesand  as a list variable.

GenerateSamples <- function(J, K, VarR, VarC)

{

# first do CAD

CasektCad <- rnorm( 2\*max(K))

dim(CasektCad) <- c(max(K),2)

#end CAD

# now do radiologists

Rjt <- rnorm( 2\*J, sd = sqrt(VarR) )

dim(Rjt) <- c(J,2)

CasektHum <- sqrt(VarC) \* CasektCad

Epsjkt <- rnorm( J\*2\*max(K), sd = sqrt(1-VarC) )

dim(Epsjkt) <- c(J,max(K),2)

# end radiologists

return ( list(

Rjt = Rjt,

CasektCad = CasektCad,

CasektHum = CasektHum,

Epsjkt = Epsjkt)

)

}

Line 3 – 6 implements the CAD ratings model, Eqn. 5-7. Line 8-14 implements the radiologist’s ratings model, Eqn. 5-7. Line 9 generates *2J* values from  and the line 10 dimensions the array to 2 rows (corresponding to the two states of truth) and J columns. Line 11 does not create any new samples; rather it simply re-uses the previously sampled values for CAD and scales them by the factor , as in Eqn. 5. This is the term yielding the correlation between the radiologist and CAD ratings. Line 12 obtains the necessary samples for the term and the next line shapes the matrix appropriately. Finally, line 16-21 returns the samples , ,  and  as a list variable.

# Bookmarks

Chapter "Preliminaries"

Chapter "Binary Paradigm"

Chapter "Modeling Binary Paradigm"

Chapter "Ratings Paradigm"

Chapter "Empirical AUC"

Chapter "Binormal Model"

Chapter "Sources of variability in AUC"

Chapter "Hypothesis Testing"

Chapter "DBMH Analysis"

Chapter "ORH Analysis"

Chapter "Sample Size"

Chapter "FROC Paradigm"

Chapter "Empirical plots from FROC data"

Chapter "Meanings of FROC FOMs"

Chapter "Visual Search Paradigms"

Chapter "Radiological Search Model (RSM)"

Chapter "Predictions of the RSM"

Chapter Analyzing FROC Data

Chapter "RSM Fitting"

Chapter "Other proper ROC models"

Chapter "Bivariate binormal model"

Chapter "CAD Evaluation"

Chapter "Validating CAD Analysis"

Fig. 1.3.2

Appendix 1.A

Figure 1.A.4

Figure 1.A.5.1

Figure 1.A.5.2

Figure 1.A.5.1.2

2.9.2

Table 2.2.1 Truth Table

Table 2.2.2

Table 2.5

Appendix 2.A

Fig. 3.6

Fig. 3.6.1

Fig. 3.6.2

Fig. 3.C.1

Fig. 3.9

Fig. 3.9.5

Fig. 3.11.1 Beam

Fig. 3.A.2

Fig. 3.C.1

Fig. 3.G.2

Fig. 3.G.3

Fig. 3.C.4

Fig. 3.E.1

Fig. 3.G.2

Fig. 3.G.3

Fig. 4.G.4

Fig. 3.H.2.1

Fig. 3.H.2.2

Fig. 3.I.2

3.10

Table 3.7.6

Table 3.11.1 Beam

Appendix 3.A: R code demonstration of concepts

Appendix 3.B: Calculating a confidence interval

Appendix 3.C: Introduction to R/RStudio, part II

Appendix 3.D: Plotting in R

Appendix 3.E: Getting help in R – Part I

Appendix 3.F: Getting help in R – Part II

Appendix 3.G: What to do if a package is missing

Appendix 3.H: Showing shaded distributions in R

Appendix 3.I: Numerical integration in R

Chapter 4

Fig. 4.3.1

Fig 4.6.1

Fig. 4.7.1

Table 4.2.1

Table 4.3.1

Table 4.3.2

Table 4.3.3

Table 4.10.1

Appendix 4.A

Chapter 5

Table 5.2.1

Table 5.3.1

Fig. 5.3.1

Fig. 5.4.1

Fig. 5.6.1

5.6

Appendix 5.A

Fig. 5.A.2.1

Fig. 5.A.2.2

Fig. 5.A.2.3

Fig. 5.A.2.4

Fig. 5.A.2.5

Appendix 6.A

Appendix 6.B

Appendix 6.C

Appendix 6.D

Appendix 6.E

Appendix 6.F

6.2.18 Eqn. FPF

6.2.19 Eqn. TPF

6.4.2

Fig. 6.2.6

Fig. 6.2.1

Fig. 6.2.3

Fig. 6.2.8

Fig. 6.3.2

Table 6.2

Table 6.4.4

Fig. 6.A.2.1

Fig. 6.A.2.2

Fig. 6.A.3

Fig. 6.A.4

7.3.2

Table 7.4.1

Table 7.4.2

Table 7.6

Fig. 7.9

Appendix 7.A

Appendix 7.B

Appendix 7.C

Appendix 7.D

Appendix 7.E

Table 8.5.1

Fig. 8.5.1

9

Table 9.7

9.10.2.2

Fig. 9.7.3

Table 9.8

Table 9.9

Table 9.10.2

Table 9.10.2.1

Table 9.10.2.4

Table 9.10.2.5

Table 9.11.1

Fig. 9.10.1.1

Fig. 9.10.1.2

Fig. 9.11.1

9.13

Table 9.13.1

Table 9.13.3

Fig. 9.A.2

Fig. 9.A.3.1

Table 9.E.1

Table 9.H.1

Appendix 9.A

Appendix 9.B

Appendix 9.C datasets

Table 9.C.1 summary datasets

Appendix 9.D

Appendix 9.E

Appendix 9.F

Appendix 9.G

Appendix 9.H

10.2.5.1

10.2 1RMultipleT

10.5.1

Fig. 10.5.1

10.2.4 Meaning of Cov matrix

Appendix 10.A

Appendix 10.B

Appendix 10.C

Appendix 10.D

Appendix 10.E

Appendix 10.F

Table 11.2.1 2 types of errors

Fig. 11.3.1

11.7.1

11.7.2

Fig. 11.11.1

Fig. 11.11.2

Fig. 11.12.1

Fig. 11.12.2

11.12

Appendix 11.A

Appendix 11.B

Appendix 11.C

Appendix 11.D

Chapter 12

Fig. 12.2.1

Fig. 12.5.1

Fig. 12.5.2

Appendix 12.B

Fig. 12.A.1

Fig. 12.B.1

Fig. 12.B.2

Table 12.4.1

12.A.1

12.6

Chapter 13

Fig. 13.4.2

Fig. 13.10.1

Fig. 13.10.4

Fig. 13.13.1

Fig. 13.14.1

Fig. 13.15.1

Fig. 13.16.1

Fig. 13.16.2

Fig. 13.16.3

13.5

13.6

13.7

13.4.1

13.8.1

13.10.1

13.10.3.1

Fig. 13.10.4.1

13.10.5.1

13.10.5.2

13.11

13.13.1

13.12.1

13.12.2

13.12.3

13.14.1

13.15.1

Appendix 13.A.1

Table 13.2.1

Table 13.12.3

Table 13.14

Table 13.15

Table 13.16

Table 14.C.1

Fig. 14.4.1

Fig. 14.5.3.1

Fig. 14.5.3.2

Fig. 14.5.3

Fig. 14.5.4.1

Fig. 14.5.4.2

14.A

Table 14.5.2.1

Table 14.5.2.2

14.5.1

14.5.2

14.2

Appendix 14.A

Appendix 14.B

Appendix 14.C

Appendix 14.D

Fig. 15.2.1

Fig. 15.2.2

Fig. 15.4.2

Fig. 15.6.1

Fig. 15.7.1

Fig. 15.6.2

Fig. 15.A.1

Appendix 15A

Appendix 15B

Appendix 15C

Appendix 15D

Table 16.2.3.1

16.4

Appendix 16.A.1

Appendix 16.B.1

Appendix 16.C.1

17.3

Fig. 17.5.2.1

Fig. 17.5.2.2

Fig. 17.6.1

Fig. 17.8.1

Fig. 17.7.1

Fig. 17.7.4.1

Fig. 17.10.1

Fig. 17.10.2

Fig. 17.11.2

Fig. 17.11.3.1.1

Fig. 17.11.3.1.2

Table. 17.11.3.1.2

Table 17.11.3.2

17.5.1

17.5.2

17.6.2

17.7.2

17.8 Search performance

17.9 Class. Performance

17.10 FROC bad

17.11 Evidence for RSM

17.11.2.1 Eng Code demo

17.11.4

Table 17.4.6.2

Table 17.11.2

Online Appendix 17.A

Online Appendix 17.B

Online Appendix 17.C

Online Appendix 17.D

Online Appendix 17.E

Online Appendix 17.F

Online Appendix 17.G

18

Fig. 18.3.1

Fig. 18.3.2

Table 18.2.1

Table 18.2.2

Table 18.2.3

19.5

19.8

Fig. 19.3.1

Table 19.5.1

Table 19.5.2

Table 19.5.3

Table 19.5.4

Table 19.5.5

Fig. 19.7.1

Fig. 19.7.2

Table 19.7.1

Table 19.8.2.1

Fig. 19.8.2.1

Fig. 19.5.1

Fig. 19.5.2

Online Appendix 19A

Online Appendix 19.A.1

Online Appendix 19.A.2

Online Appendix 19.A.3

Online Appendix 19.B

Online Appendix 19.B.1

Online Appendix 19.C

Online Appendix 19.C.1

Online Appendix 19.C.2

Online Appendix 19.C.3

Online Appendix 19.C.4

Online Appendix 19.C.5

Online Appendix 19.C.6

Online Appendix 19.C.7

Online Appendix 19.C.8

20.7.2.1

Fig. 20.4.1

Fig. 20.4.2

Fig. 20.5.3

Fig. 20.6.1

Fig. 20.7.1

Fig. 20.8.1

Fig. 20.8.2

Fig. 20.8.3

Fig. 20.9.1

Fig. 20.9.2

Table 20.5.1

Table 20.7.1.1

Online Appendix 20.A

Online Appendix 20.B

Online Appendix 20.B.2

Online Appendix 20.B.4

Online Appendix 20.D

Fig. 21.4.1

Fig. 21.6.1

Table 21.6.1

Fig. 21.7.1

Fig. 21.7.2

Fig. 21.B.1

Fig. 21.D.1

Fig. 21.D.2

Fig. 21.D.3

Fig. 21.D.4

Fig. 21.D.5

Fig. 21.D.6

Appendix 21.A

Appendix 21.B

Appendix 21.C

Appendix 21.D

Appendix 21.E

22

Table 22.2.1

Appendix 22.A

Appendix 22.B

22.2.1.1

22.2.1.2

22.2.1.3

22.2.1.4

Fig. 22.2.1

Fig. 22.A.1

Table 22.3.1

22.3.2

22.4

22.5

Table 22.5.1

23.4

Online Appendix 23.A.1

Online Appendix 23.A.2

Online Appendix 23.A.3

Online Appendix 23.A

Online Appendix 23.B

Online Appendix.23.B.1

Online Appendix.23.B.2

Online Appendix 23.C

Online Appendix 23.D

# Cross-references

**Chapter 01**

**Chapter 02**

**Chapter 03**

**Chapter 04**

**Chapter 05**

**Chapter 06**

**Chapter 07**

**Chapter 08**

**Chapter 09**

**Chapter 10**

**Chapter 11**

**Chapter 12**

**Chapter 13**

**Chapter 14**

**Chapter 15**

**Chapter 16**

**Chapter 17**

**Chapter 18**

**Chapter 19**

**Chapter 20**

**Chapter 21**

**Chapter 22**

Chapter "Validating CAD Analysis"

Online Appendix 1.A

Fig. 1.1

Figure 1.A.4

Figure 1.A.5.1

Figure 1.A.5.2

Figure 1.A.5.2

Figure 1.A.5.1.2

2.9.2 PPV NPV

Table 2.1

Table 2.2

Table 2.3

Appendix 2.A

Fig. 3.1

Fig. 3.2

Fig. 3.3

Fig. 3.A.2

Fig. 3.H.2.1

Fig. 3.H.2.2

Fig. 3.E.1

Fig. 3.4

Fig. 3.5

Fig. 3.6 beam et al study

Fig. 3.G.3

Fig. 3.C.4

Fig. 3.A.2

Fig. 3.C.1

Fig. 3.E.1

Fig. 3.G.2

Fig. 3.G.3

Fig. 4.G.4

Fig. 3.H.2.1

Fig. 3.I.2

3.10

Table 3.2

Table 3.3 beam et al study

Online Appendix 3.A

Online Appendix 3.B

Online Appendix 3.C

Online Appendix 3.D

Online Appendix 3.E Getting help in R-I

Online Appendix 3.F Getting help in R-II

Online Appendix 3.G

Online Appendix 3.H

Online Appendix 3.I

Fig. 4.3.1 equal var vs uneq var

Fig 4.6.1

Fig. 4.7.1  
Table 4.1 Roc counts table

Table 4.2

Table 4.3

Table 4.4

Table 4.5

Online Appendix 4.A

Table 5.1

Table 5.2

5.6

Online Appendix 5.A

Fig. 5.A.2.1 Debug buttons

Fig. 5.A.2.2 Debug buttons

Fig. 5.A.2.3 Debug buttons

Fig. 5.A.2.4 Debug buttons

Fig. 5.A.2.5 Debug buttons

Appendix 6.A

Appendix 6.B Eng Java

Appendix 6.C MLE

Appendix 6.D Goodness of fit

Appendix 6.E

Appendix 6.F Partial/full Az

6.2.18 Eqn. FPF binormal

6.2.19 Eqn. TPF

6.4.2

Fig. 6.2.6

Fig. 6.2.8

Fig. 6.2.1

Fig. 6.2.3

Fig. 6.3.2

Table 6.2

Table 6.4.4

Fig. 6.A.2.1

Fig. 6.A.2.2

Fig. 6.A.3

Fig. 6.A.4

7.3.2

7.3.2

Table 7.4.1

Table 7.4.2

Table 7.6

Fig. 7.9

Online Appendix 7.A

Online Appendix 7.B

Online Appendix 7.C

Online Appendix 7.D

Online Appendix 7.E

Table 8.5.1

Table 8.5.1

Fig. 8.5.1

Table 9.7

9.10.2.2

9.10.2.2

Fig. 9.7.3

Table 9.8

Table 9.9

Table 9.10.2.1

Table 9.10.2.4

Table 9.10.2.5

Table 9.11.1

Fig. 9.10.1.1

Fig. 9.10.1.2

Fig. 9.11.1

Table 9.10.2

9.13: Meaning of pseudovalues

Table 9.13.1

Table 9.13.3

Fig. 9.A.2

Fig. 9.A.3.1

Table 9.E.1

Table 9.H.1

Online Appendix 9.A

Online Appendix 9.B

Appendix 9.C datasets

Table 23.1 summary datasets

Online Appendix 9.C

Online Appendix 9.D

Online Appendix 9.E

Online Appendix 9.F

Online Appendix 9.G

10.2.5.1

10.2 1RIT

10.5.1

Fig. 10.5.1

10.5.1

10.2.4 Meaning of Cov matrix

Online Appendix 10.A

Online Appendix 10.B

Online Appendix 10.C

Online Appendix 10.D

Online Appendix 10.E

Online Appendix 10.F

Table 11.1 2 types of errors

Fig. 11.1

11.8.1

11.8.4

Fig. 11.2

Fig. 11.3

Fig. 11.4

Fig. 11.5

11.12

Online Appendix 11.A

Online Appendix 11.B

Online Appendix 11.C

Online Appendix 11.D

Online Appendix 12.B

Fig. 12.B.2

12.6

Table 9.H.1

Table 13.1 FROC notation

Table 13.2

Table 13.3

Table 13.4

Table 13.5

Fig. 13.4.2

Fig. 13.10.1

Fig. 13.10.4

Fig. 13.13.1

Fig. 13.14.1

Fig. 13.15.1

Fig. 13.16.1

Fig. 13.16.2

Fig. 13.16.3

13.6

13.7

13.4.1

13.16.2

13.10.1

13.10.3.1

13.10

13.10.5.1

13.15.1

13.15

Appendix 13.A.1

13.11.1

13.11.2

13.11.3

13.12.1

13.13.1

13.15.1

Table 14.C.1

Fig. 14.4.1

Fig. 14.5.3.1

Fig. 14.5.3.2

Fig. 14.5.4.1

Fig. 14.5.4.2

14.5.1

14.5.2

14.2

Online Appendix 14.A

Online Appendix 14.B

Online Appendix 14.C

Online Appendix 14.D

Table 14.5.2.1

Table 14.5.2.2

Fig. 15.1

Fig. 15.2

Fig. 15.3

Fig. 15.4

Fig. 15.6

Fig. 15.7

Fig. 15.A.1

Appendix 15A

Appendix 15B

Appendix 15C

Appendix 15D

Table 16.1

16.4

Online Appendix 16.A.1

Online Appendix 16.B.1

Online Appendix 16.C.1

17.3

Fig. 17.1

Fig. 17.2

17.5.1

17.5.2

Fig. 17.3

Fig. 17.4

Fig. 17.8.1

Fig. 17.5

Fig. 17.6

Fig. 17.7

Fig. 17.8

Fig. 17.9

Fig. 17.10

Table. 17.11.3.1.2

Table 17.3

17.5.1

17.5.2

17.6.2

17.7.2

17.8 Search performance

17.9 Class. Performance

17.10 Froc is bad

17.11 Evidence for RSM

17.11.2.1

17.11.4

Table 17.1

Table 17.2

Online Appendix 17.A

Online Appendix 17.B

Online Appendix 17.C

Online Appendix 17.G

Online Appendix 17.E

Online Appendix 17.F

Online Appendix 17.G

18

Fig. 18.1

Fig. 18.2

Fig. 18.3

Table 18.1

Table 18.2

Table 18.3

19.5

19.8 Sample size

Table 19.5.1

Table 19.5.2

Table 19.5.3

Table 19.5.4

Table 19.5.5

Fig. 19.7.1

Fig. 19.7.2

Table 19.7.1

Fig. 19.3.1

Fig. 19.5.1

Table 19.8.2.1

Fig. 19.8.2.1

Fig. 19.5.2

Online Appendix 19A

Online Appendix 19.A.1

Online Appendix 19.A.2

Online Appendix 19.A.3

Online Appendix 19.A.4

Online Appendix 19.B

Online Appendix 19.C.1

Online Appendix 19.C

Online Appendix 19.C.1

Online Appendix 19.C.2

Online Appendix 19.C.3

Online Appendix 19.C.4

Online Appendix 19.C.5

Online Appendix 19.C.6

Online Appendix 19.C.7

Online Appendix 19.C.8

20.7.2.1

Fig. 20.1

Fig. 20.2

Fig. 20.3

Fig. 20.4

Fig. 20.5

Fig. 20.7

Fig. 20.8

Fig. 20.8.3

Fig. 20.9

Fig. 20.10

Table 20.1

Table 20.2

Online Appendix 20.A

Online Appendix 20.B

Online Appendix 20.B.2

Online Appendix 20.B.4

Online Appendix 20.D

Fig. 21.1

Fig. 21.2

Table 21.1

Fig. 21.4

Fig. 21.5

Fig. 21.B.1

Fig. 21.D.1

Fig. 21.D.2

Fig. 21.D.3

Fig. 21.D.4

Fig. 21.D.5

Fig. 21.D.6

Online Appendix 21.A

Online Appendix 21.B

Online Appendix 21.C

Online Appendix 21.D

Online Appendix 21.E

Table 22.1

Online Appendix 22.A

Online Appendix 22.B

22.2.1.1

22.2.1.2

22.2.1.3

22.2.1.4

Fig. 22.1

Fig. 22.2

Table 22.2

22.3.2

22.4

22.5

Table 22.3

23.4

Online Appendix 23.A.1

Online Appendix 23.A.2

Online Appendix 23.A.3

Online Appendix 23.A

Online Appendix 23.B

Online Appendix.23.B.1

Online Appendix.23.B.2

# Book last updated:

2/24/17 8:14:15 PM

# References

1. Baker JA, Rosen EL, Lo JY, Gimenez EI, Walsh R, Soo MS. Computer-aided detection (CAD) in screening mammography: sensitivity of commercial CAD systems for detecting architectural distortion. *American Journal of Roentgenology.* 2003;181(4):1083-1088.

2. Baker JA, Lo JY, Delong DM, Floyd CE. Computer-aided Detection in Screening Mammography: Variability in Cues 1. *Radiology.* 2004;233(2):411-417.

3. Kim SJ, Moon WK, Cho N, Cha JH, Kim SM, Im J-G. Computer-aided Detection in Full-Field Digital Mammography: Sensitivity and Reproducibility in Serial Examinations 1. *Radiology.* 2008;246(1):71-80.

4. Dorfman DD, Alf E. Maximum-Likelihood Estimation of Parameters of Signal-Detection Theory and Determination of Confidence Intervals - Rating-Method Data. *Journal of Mathematical Psychology.* 1969;6:487-496.

5. Thompson ML, Zucchini W. On the statistical analysis of ROC curves. *Stat Med.* 1989;8(10):1277-1290.

6. Marx ML, Larsen RJ. *Introduction to mathematical statistics and its applications.* Vol 31: Pearson/Prentice Hall Upper Saddle River, NJ, USA; 2006.

7. Roe CA, Metz CE. Dorfman-Berbaum-Metz Method for Statistical Analysis of Multireader, Multimodality Receiver Operating Characteristic Data: Validation with Computer Simulation. *Acad Radiol.* 1997;4:298-303.