R Code for the paper *Computer Experiments: Prediction Accuracy, Sample Size and Model Complexity Revisited*

Ofir Harari*, Derek Bingham*, Angela Dean †and Dave Higdon ‡

```
#**************************************************************************
#**************************************************************************
#** This is the R code for the Simulation study of Section 5 of our      **
#** paper, entitled "Computer Experiments: Prediction Accuracy,          **
#** Sample Size and Model Complexity Revisited", tp appear in Statistica **
#** Sinica. For questions, comments or code for the accompanying R Shiny **
#** application, please write to oharari@sfu.ca                          **
#**************************************************************************
#**************************************************************************


#before running this script, make sure you have the following packages
#installed locally -

library(DiceKriging)
library(lhs)
library(fOptions)



#**************************************************************************
#**************************************************************************
#**** Rescaling everything from the unit cube to the original units    ****
#**************************************************************************
#**************************************************************************
rescale <- function(D.0)
{
        D.1 <- D.0
        D.1[,1] <- D.1[,1]*30+30
        D.1[,2] <- D.1[,2]*.015 + .005
```

*Department of Statistics and Actuarial Science, Simon Fraser University
†Department of Statistics, The Ohio State University
‡Social and Decision Analytics Laboratory, Biocomplexity Institute of Virginia Tech

```
        D.1[,3] <- D.1[,3]*.008 + .002
        D.1[,4] <- D.1[,4]*4000 + 1000
        D.1[,5] <- D.1[,5]*20000 + 90000
        D.1[,6] <- D.1[,6]*6 + 290
        D.1[,7] <- D.1[,7]*20 + 340

        return(D.1)
}
#*****************************************************************************
#*****************************************************************************



#*****************************************************************************
#*****************************************************************************
piston <- function(xx)
{
  ###########################################################################
  #
  # PISTON FUNCTION
  #
  # Authors: Sonja Surjanovic, Simon Fraser University
  #          Derek Bingham, Simon Fraser University
  # Questions/Comments: Please email Derek Bingham at dbingham@stat.sfu.ca.
  #
  # Copyright 2013. Derek Bingham, Simon Fraser University.
  #
  #
  # For function details and reference information, see:
  # http://www.sfu.ca/~ssurjano/
  #
  ###########################################################################
  #
  # OUTPUT AND INPUT:
  #
  # C = cycle time
  # xx = c(M, S, V0, k, P0, Ta, T0)
  #
  ###########################################################################

  M  <- xx[1]
  S  <- xx[2]
  V0 <- xx[3]
  k  <- xx[4]
  P0 <- xx[5]
  Ta <- xx[6]
  T0 <- xx[7]
```

```
  Aterm1 <- P0 * S
  Aterm2 <- 19.62 * M
  Aterm3 <- -k*V0 / S
  A <- Aterm1 + Aterm2 + Aterm3

  Vfact1 <- S / (2*k)
  Vfact2 <- sqrt(A^2 + 4*k*(P0*V0/T0)*Ta)
  V <- Vfact1 * (Vfact2 - A)

  fact1 <- M
  fact2 <- k + (S^2)*(P0*V0/T0)*(Ta/(V^2))

  C <- 2 * pi * sqrt(fact1/fact2)
  return(C)
}
#*******************************************************************************
#*******************************************************************************



#*******************************************************************************
#*******************************************************************************
#****                        Fitting a GP model                             ****
#*******************************************************************************
#*******************************************************************************
model.fit <- function(size, X.test, y.test)
{
        X.0 <- randomLHS(size, 5)
        X.0.1 <- cbind(X.0, .5, .5)
        X1 <- rescale(X.0.1)
        y <- apply(X1, 1, piston)
        X.0 <- data.frame(X.0)
        names(X.0) <- names(X.test)

        model <- km(design=X.0, response=y, covtype="gauss")
        y.hat <- predict(model, newdata=X.test, "UK")

        return(y.hat)
}
#*******************************************************************************
#*******************************************************************************



#*******************************************************************************
#*******************************************************************************
```

```
#****  Running the Simulation for random LHDs and various sample sizes ****
#*************************************************************************
#*************************************************************************
RAUV.sim <- function(X.test, y.test, size, n.rep, sd2)
{
        RAUV <- rep(0, n.rep)
        for(i in 1:n.rep)
        {
                y.hat <- model.fit(size, X.test, y.test)
                e <- y.test-y.hat$mean
                RAUV[i] <- sqrt(mean(e^2)/sd2)
                cat("\n End of repetition ")
                cat(i)
                cat(" for sample size ")
                cat(paste(size, "\n"))
                flush.console()
        }

        RAUV
}
#*************************************************************************
#*************************************************************************



#*************************************************************************
#*************************************************************************
#****           Plotting y vs. y.hat for a given model fit             ****
#*************************************************************************
#*************************************************************************
plot.fit <- function(size, sd2)
{
        y.hat <- model.fit(size, X.test, y.test)

        e <- y.test - y.hat$mean
        Emp <- sqrt(mean(e^2)/sd2)

        lims.min <- min(min(y.test), min(y.hat$mean))-.05
        lims.max <- max(max(y.test), max(y.hat$mean))+.15
        lims <- c(lims.min, lims.max)
        plot(y.hat$mean, y.test, xlab=expression(hat(y)), ylab="y", cex.lab=1.5,
                axes=0, xlim=lims, ylim=lims)
        axis(1, pos=lims.min, at=round(seq(lims.min, lims.max, by=.2),1))
        axis(2, pos=lims.min, at=round(seq(lims.min, lims.max, by=.2),1))
        text(x=.8, y=.2, substitute(paste("ERAUV = ", tt), list(tt=round(Emp,3))))
        lines(c(0,1.1), c(0,1.1), col=2, lwd=2, lty=2)
```

```
        title(main=paste("GP fit for a sample size of", size), cex.main=1.2)


        Emp
}
#*************************************************************************
#*************************************************************************
```

```
#*************************************************************************
#*************************************************************************
#*************************************************************************
#*******        End of functions, Simulation starts here           ********
#*************************************************************************
#*************************************************************************
#*************************************************************************

N.integration <- 100000 #size of holdout set
X.test <- runif.halton(N.integration, 7) #holdout set
X.test1 <- rescale(X.test)
y.test <- apply(X.test1, 1, piston) #response evaluation for hodout set (y.ho)
X.test <- data.frame(X.test[,1:5]) #reducing dimensionality to 5



#****** Fitting a model on 1000 observations to estimate sigma^2
X.sd2.est <- runif.halton(1000, 5) #training set
X.sd2.est1 <- cbind(X.sd2.est,.5,.5)
y.sd2.est <- apply(rescale(X.sd2.est1), 1, piston) #observed data (y)
model.sd2.est <- km(design=X.sd2.est, response=y.sd2.est, covtype="gauss") #fitting a GP
sd2 <- coef(model.sd2.est)$sd2 #estimated sigma^2



#*********** Simulation *********
n.rep <- 50 # No. of repetitions per sample size
size <- c(30, 50, 70, 120) # sample sizes

#running simulation
RAUV <- sapply(size, RAUV.sim, X.test=X.test, y.test=y.test, n.rep=n.rep, sd2=sd2)
RAUV.hat <- apply(RAUV, 2, mean)
sds <- apply(RAUV, 2, sd)
```

```
data.frame(cbind(size, RAUV.hat, sds)) #summary of results



#***** plotting boxplots *******
dev.new(width=8)
par(mar=c(5,6,3,1))
boxplot(RAUV, axes=0, xlab="Sample Size", ylab="ERAUV", cex.lab=2, ylim=c(0.04, 0.24))
axis(1, at=c(-1,1:4,6), labels=c("","30","50","70","120",""), cex.axis=1.6)
axis(2, pos=.5, at=seq(0,1,by=.01))
lines(c(.5,3.6), rep(median(RAUV[,4]),2), lty=2, col=4, lwd=2)



#****** Example of y vs. y.hat plots for the different sample sizes
dev.new(height=9, width=9)
par(mfrow=c(2,2), mar=c(5,5,3,1))
set.seed(24)
plot.fit(30, sd2)
set.seed(18)
plot.fit(50, sd2)
set.seed(17)
plot.fit(70, sd2)
set.seed(3)
plot.fit(120, sd2)
```