Web-appendix of 'On an Inflated Unit-Lindley Distribution'

A R function used for simulation study

```
rm(list = ls())
library(LindleyR)
library(expint)

f = function(p, alpha, theta){

  ruy = numeric()
  t_y = numeric()
  rby = numeric()

  b <- c(50, 100, 200, 500, 1000)

simstudy <- matrix(0, nrow=15, ncol=5, byrow = TRUE)

rownames(simstudy) <- c("Bias.alpha", "Bias.theta", "Bias.p", "MSE
    ↪ .alpha",
"MSE.theta", "MSE.p", "alpha.est", "theta.est", "p.est", "E_y", "V
    ↪ _y", "bias.E", "bias.V", "mse.E", "mse.V")

colnames(simstudy) <- c(b[1], b[2], b[3], b[4], b[5])

#True mean and variance
mean = alpha * p + ((1 - alpha)/(1 + theta))
Variance = alpha * p + ((1 - alpha)/(1 + theta)) * ((theta^2) *
    ↪ expint_Ei(theta) - theta + 1)

for(k in 1:5)
  {
  n <- 2000
  dummy = matrix(5*n, nrow = n, ncol = 5, byrow = TRUE)

  for(i in 1:n)
  {
  w <- b[k]
    for(j in 1:w)
    {
      rl <- rlindley(1, theta)
      ruy[j] <- rl/(1 + rl) #unit-Lindley random variables
```

```
    rby[j] <- rbinom(1, 1, p) #bernoulli random variables
  }
    #partition data using alpha
    uy <- sample(ruy, (1 - alpha) * w, replace = FALSE)
    by <- sample(rby, alpha * w, replace = FALSE)

    y <- sort(c(uy, by))
    x <- ifelse(y == 1, 1, 0)
    T2 <- sum(x)

    a <- ifelse(y==0 | y==1, 1, 0)
    T1 <- sum(a)

    alpha_hat <- T1/w # estimate of mixture parameter, alpha
    p_hat <- T2/T1 # estimate of Bernoulli parameter p

    len <- length(uy)
    for(pp in 1: len)
      {
    t_y[pp] <- uy[pp]/(1 - uy[pp])
      }
    ty <- sum(t_y)
    # estimate of unit lindley parameter, theta
    theta_hat <-(1/(2 * ty)) * (len - ty + sqrt(ty^2 + 6 * len *
       ↪ ty + len^2))

dummy[i,1] = alpha_hat
dummy[i,2] = theta_hat
dummy[i,3] = p_hat
dummy[i,4] = dummy[i,1] * dummy[i,3] + ((1 - dummy[i,1])/(1 +
    ↪ dummy[i,2]))
dummy[i,5] = dummy[i,1] * dummy[i,3] + ((1 - dummy[i,1])/(1 +
    ↪ dummy[i,2])) * (dummy[i,2]^2 * expint_Ei(dummy[i,2]) -
    ↪ dummy[i,2] + 1)
}
dummy <- dummy[complete.cases(dummy),]

rw <- nrow(dummy)

bias.alpha = mean(dummy[,1] - alpha)
bias.theta = mean(dummy[,2] - theta)
bias.p = mean(dummy[,3] - p)
bias.E = mean(dummy[,4] - mean)
bias.V = mean(dummy[,5] - Variance)
```

```
  mse.alpha = (1/rw) * sum((dummy[,1] - alpha)^2)
  mse.theta = (1/rw) * sum((dummy[,2] - theta)^2)
  mse.p = (1/rw) * sum((dummy[,3] - p)^2)
  mse.E = (1/rw) * sum((dummy[,3] - mean)^2)
  mse.V = (1/rw) * sum((dummy[,3] - Variance)^2)

  alpha.est = sum(dummy[,1])/rw
  theta.est = sum(dummy[,2])/rw
  p.est = sum(dummy[,3])/rw
  E_p_est = sum(dummy[,4])/rw
  V_p_est = sum(dummy[,5])/rw

  simstudy[1,k] <- bias.alpha
  simstudy[2,k] <- bias.theta
  simstudy[3,k] <- bias.p

  simstudy[4,k] <- mse.alpha
  simstudy[5,k] <- mse.theta
  simstudy[6,k] <- mse.p

  simstudy[7,k] <- alpha.est
  simstudy[8,k] <- theta.est
  simstudy[9,k] <- p.est

  simstudy[10,k] <- E_p_est
  simstudy[11,k] <- bias.E
  simstudy[12,k] <- mse.E

  simstudy[13,k] <- V_p_est
  simstudy[14,k] <- bias.V
  simstudy[15,k] <- mse.V
}
simstudy
write.csv(simstudy,"simulation_result.csv")
}
```

B R code for generating the hypothetical data

```
rm(list = ls())
library{LindleyR}
set.seed(99)
theta <- 1.444589
for(p in 1:190)
  {
    h[p] <- rlindley (1, theta)
    hul[p] <- h[p] / (1 + h[p])
  }
hdata <- c(rep(0,20),hul, rep(1,60)) #Hypothetical data
```