Simulate regression with measurement error in x variables*

```
> source("https://www.utstat.toronto.edu/~brunner/openSEM/fun/rmvn.txt")
>
> n = 500 # Sample size
> # Regression coefficients
> beta0 = 1; beta1 = 1; beta2 = 0
> # Expected values of x variables
> mux = c(10,20)
> # Variance-covariance matrix of x variables: Correlation = 0.75
> Phi = rbind(c(20,  30),
+             c(30,  80))
> # Variance of epsilon
> psi = 50
> # Variances of measurement error terms: Both reliabilities = 0.80
> omega1 = 5; omega2 = 20
>
> # Simulate data
> set.seed(9999)
>
> X = rmvn(n,mux,Phi); head(X)
          [,1]     [,2]
[1,] 11.844842 30.51695
[2,] 13.784308 23.57709
[3,] 18.372468 46.74904
[4,]  6.935507 11.24761
[5,] 12.717602 33.65471
[6,]  5.619894 17.13223
> x1 = X[,1]; x2 = X[,2]
> epsilon = rnorm(n,0,sqrt(psi))
> e1 = rnorm(n,0,sqrt(omega1)); e2 = rnorm(n,0,sqrt(omega2))
>
> y = beta0 + beta1*x1 + beta2*x2 + epsilon
> w1 = x1 + e1
> w2 = x2 + e2
> mod = lm(y ~ w1 + w2)
> summary(mod)

Call:
lm(formula = y ~ w1 + w2)

Residuals:
     Min       1Q   Median       3Q      Max
-27.4119  -4.8809  -0.0529   4.7602  18.3825

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.02893    0.81016   3.739 0.000206 ***
w1           0.63673    0.08086   7.874 2.17e-14 ***
w2           0.08801    0.04198   2.096 0.036543 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.569 on 497 degrees of freedom
Multiple R-squared:  0.2054,	Adjusted R-squared:  0.2022
F-statistic: 64.22 on 2 and 497 DF,  p-value: < 2.2e-16

> summary(mod)$coefficients[3,4]
[1] 0.03654336
```

```
> onesim = function()
+     {
+     X = rmvn(n,mux,Phi); head(X)
+     x1 = X[,1]; x2 = X[,2]
+     epsilon = rnorm(n,0,sqrt(psi))
+     e1 = rnorm(n,0,sqrt(omega1)); e2 = rnorm(n,0,sqrt(omega2))
+     y = beta0 + beta1*x1 + beta2*x2 + epsilon
+     w1 = x1 + e1
+     w2 = x2 + e2
+     mod = lm(y ~ w1 + w2)
+     return(summary(mod)$coefficients[3,4])
•     } # End of onesim
```

```
> onesim()
[1] 0.003387256
> onesim()
[1] 0.157549
> onesim()
[1] 0.0002330494
> onesim()
[1] 0.03703943
> onesim()
[1] 0.01261625
> onesim()
[1] 0.01333088
> onesim()
[1] 0.00381823
> onesim()
[1] 0.09167006
> onesim()
[1] 0.006254434
> onesim()
[1] 0.1936868
> onesim()
[1] 0.1007802
> onesim()
[1] 0.06456996
> onesim()
[1] 0.001853765
> onesim()
[1] 0.1529416
> onesim()
[1] 0.01487092
> onesim()
[1] 0.007516158
> onesim()
[1] 0.007077781
> onesim()
[1] 0.002252957
> onesim()
[1] 0.3086806
> onesim()
[1] 0.2501769
> onesim()
[1] 0.04181193
> onesim()
[1] 0.0270884
> onesim()
[1] 0.001101151
```

```
mereg <- function(beta0=1, beta1=1, beta2=0, sigmasq = 0.5,
          mu1=0, mu2=0, phi11=1, phi22=1, phi12 = 0.80,
          rel1=0.80, rel2=0.80, n=200)
####################################################################
# Model is    Y  = beta0 + beta1 X1 + beta2 X2 + epsilon
#             W1 = X1 + e1
#             W2 = W2 + e2
# Fit naive model
#             Y  = beta0 + beta1 W1 + beta2 W2 + epsilon
# Inputs are
#
#   beta0, beta1 beta2      True regression coefficients
#   sigmasq                 Var(epsilon)
#   mu1                     E(X1)
#   mu2                     E(X2)
#   phi11                   Var(X1)
#   phi22                   Var(X2)
#   phi12                   Cov(X1,X2) = Corr(X1,X1), because
#                           Var(X1) = Var(X2) = 1
#   rel1                    Reliability of W1
#   rel2                    Reliability of W2
#   n                       Sample size
# Note: This function uses rmvn, a multivariate normal random number
#       generator I wrote. The rmultnorm of the package MSBVAR does
#       the same thing but I am having trouble installing it.
####################################################################

    {
    # Calculate SD(e1) and SD(e2)
    sd1 <- sqrt((phi11-rel1)/rel1)
    sd2 <- sqrt((phi22-rel2)/rel2)
    # Random number generation
    epsilon <- rnorm(n,mean=0,sd=sqrt(sigmasq))
    e1 <- rnorm(n,mean=0,sd=sd1)
    e2 <- rnorm(n,mean=0,sd=sd2)
    # X1 and X2 are bivariate normal. Need rmvn function.
    Phi <- rbind(c(phi11,phi12),
                 c(phi12,phi22))
    X <- rmvn(n, mu=c(mu1,mu2), sigma=Phi) # nx2 matrix
    X1 <- X[,1]; X2 <- X[,2]
    # Now generate Y, W1 and W2

    Y = beta0 + beta1*X1 + beta2*X2 + epsilon
    W1 = X1 + e1
    W2 = X2 + e2

    # Fit the naive model
    mereg <- summary(lm(Y~W1+W2))$coefficients
    mereg # Returns table of beta-hats, SEs, t-statistics and p-values
    } # End function mereg
```
