Statistical Computing 1, Stat 590 Homework 5b Name: Answer Key
Fall 2015 Prof. Erik B. Erhardt

Part I. (80 points) Do all calculations in LaTeX + R + knitr. Insert computer text output and graphics to support what you are saying. For this assignment, all R code should well commented and be visible (`echo=TRUE`) in the document where you have written it.

(10pts) **1. Inverse CDF method**
Suppose $X \sim$ Exponential(λ) with density

$$f(x|\lambda) \;=\; \lambda e^{-\lambda x}, \quad x \geq 0,$$

and 0 otherwise. Show by the inverse cdf method that if $u \sim$ Uniform$(0,1)$, that

$$x \;=\; -\frac{1}{\lambda}\log_e(u) \;\sim\; \text{Exponential}(\lambda).$$

Solution: If $X \sim \text{Exp}(\lambda)$, the pdf is $f(x|\lambda) = \lambda e^{-\lambda x}, x \geq 0, \lambda > 0$, with cdf $F(x|\lambda) = 1 - e^{-\lambda x} = v$, where we will have $v \sim \text{Unif}(0,1)$. Solving for x, $x = -\frac{1}{\lambda}\ln(1-v)$ and we know $x \sim \text{Exp}(\lambda)$. Let $u = 1 - v$, since $v \sim \text{Unif}(0,1)$, $1 - v = u \sim \text{Unif}(0,1)$, so $x = -\frac{1}{\lambda}\ln(u)$ is simplified.
This method works because by construction, the probability of producing a number less than x is the same as the definition of the cdf, that the probability of an exponential number being less than x is $F(x)$. Because these are equal, the probability of generating any range of numbers is correct.

(40pts) **2. Importance sampling, Beta**
Suppose $X \sim \text{Beta}(\alpha, \beta)$ and we wish to estimate the moment generating function

$$\text{M}_X(t) \;=\; \text{E}[e^{tX}] \;=\; \int e^{tX} f(x|\alpha,\beta)\,\text{d}x,$$

where $f(x|\alpha,\beta)$ is the Beta(α,β) density function. In the notes we discussed two methods. The first was a crude MC method based on sampling $X_1, X_2, \ldots, X_n \overset{\text{iid}}{\sim} \text{Beta}(\alpha,\beta)$. The second is a simple importance sample based on sampling $X_1, X_2, \ldots, X_n \overset{\text{iid}}{\sim} \text{Uniform}(0,1)$.
(a) (20 pts) Write a script that

- specifies n, α, and β,
- computes the two estimates for $t = -1$ to 1 in increments of size 0.025 or smaller,
- computes an estimated standard error for each MC estimate, as a function of t,
- computes a point-wise approximate 95% CI of the form EST $\pm$ 1.96SE for each method, and plots the estimate and CI bounds as a function of t, one plot for each estimate labeled appropriately, and
- computes the ratio of the standard error obtained by crude MC and that obtained by IS as a function of t and plot the results.

Choosing, say $n = 2000$ or so, run your script for the following scenarios:

1. $\alpha = \beta = 0.25$
2. $\alpha = \beta = 3$
3. $\alpha = 0.25$, $\beta = 3$
4. $\alpha = 3$, $\beta = 0.25$

Solution:

```
## Importance sampling

# alpha and beta parameters
beta.param <- data.frame(alpha = c(0.25, 3, 0.25, 3   )
                         , beta = c(0.25, 3, 3   , 0.25))
# sample size
n <- 2000
# value of t in MGF, E[e^{tX}]
t <- seq(-1, 1, by = 0.01)
method <- c("mc", "is")

init <- rep(NA, nrow(beta.param) * length(t) * length(method))
is.df <- data.frame(alpha    = init
                    , beta     = init
                    , t        = init
                    , method   = init
                    , mu       = init
                    , se       = init
                    , lci      = init
                    , uci      = init
                    , se.ratio = init)

ii <- 0
for (i.beta in 1:nrow(beta.param)) {
  for (i.t in t) {
    # draw samples from beta distribution for crude MC
    x.beta <- rbeta(n, beta.param$alpha[i.beta], beta.param$beta[i.beta])
    # g.x is e^{t*x}
    e.tx <- exp(i.t * x.beta)

    # MGF estimated mean, SE, and 95\% CI
    mc.mu  <- mean(e.tx)
    mc.se  <- sqrt(var(e.tx) / n)
    mc.lci <- mc.mu - 1.96 * mc.se
    mc.uci <- mc.mu + 1.96 * mc.se

    # increment row of df to store results in
    ii <- ii + 1

    is.df$alpha[ii] <- beta.param$alpha[i.beta]
    is.df$beta [ii] <- beta.param$beta[i.beta]
    is.df$t    [ii] <- i.t
    is.df$method[ii] <- method[1]

    is.df$mu [ii] <- mc.mu
    is.df$se [ii] <- mc.se
```

```
    is.df$lci[ii] <- mc.lci
    is.df$uci[ii] <- mc.uci


    # draw samples from uniform distribution for IS
    x.unif <- runif(n)
    # g.x is e^{t*x}
    e.tx <- exp(i.t * x.unif)
    # weights w(x)
    w.x  <- dbeta(x.unif, beta.param$alpha[i.beta], beta.param$beta[i.beta]) / dunif(x.unif)

    is.mu  <- mean(e.tx * w.x)
    is.se  <- sqrt(var(e.tx * w.x) / n)
    is.lci <- is.mu - 1.96 * is.se
    is.uci <- is.mu + 1.96 * is.se

    # increment row of df to store results in
    ii <- ii + 1

    is.df$alpha[ii] <- beta.param$alpha[i.beta]
    is.df$beta [ii] <- beta.param$beta[i.beta]
    is.df$t    [ii] <- i.t
    is.df$method[ii] <- method[2]

    is.df$mu [ii] <- is.mu
    is.df$se [ii] <- is.se
    is.df$lci[ii] <- is.lci
    is.df$uci[ii] <- is.uci

    # SE ratio
    is.df$se.ratio[ii] <- mc.se / is.se     #£
  }
}
is.df$alpha <- factor(is.df$alpha)
is.df$beta  <- factor(is.df$beta)
```

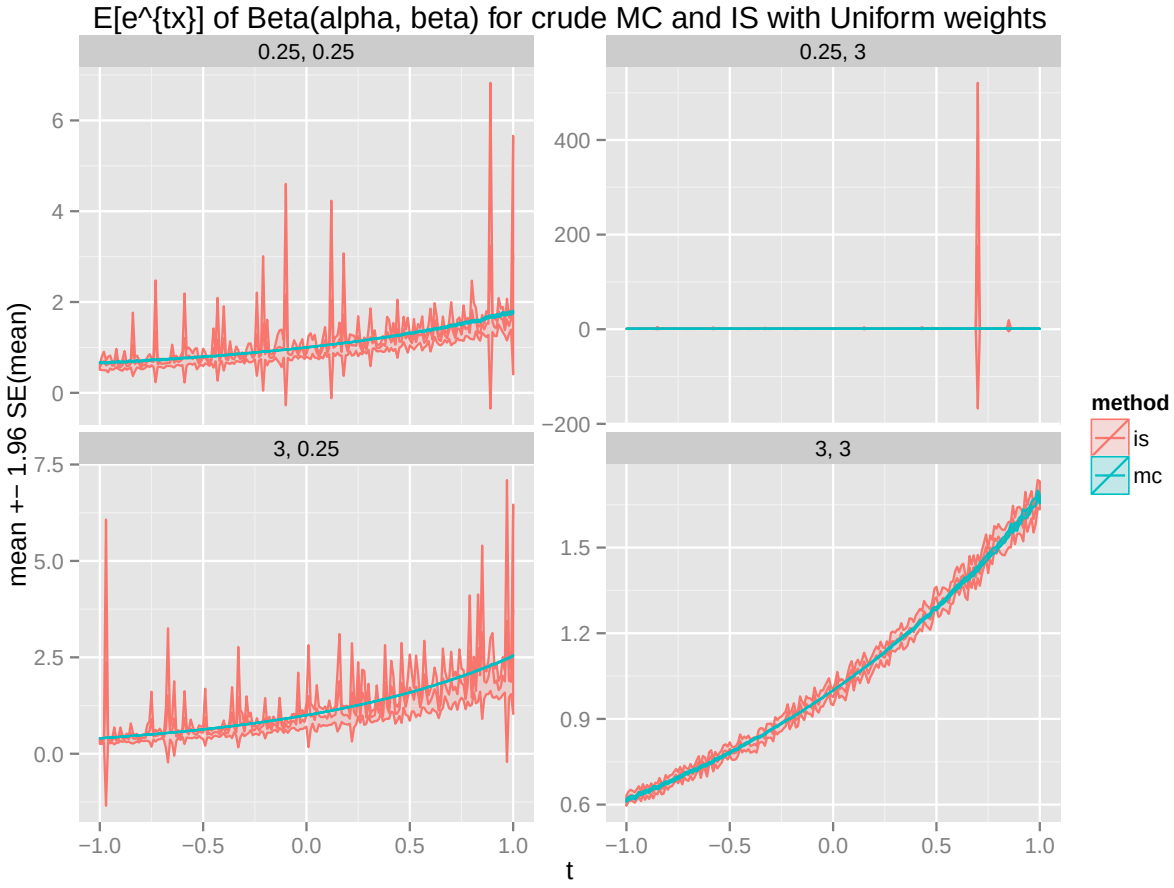
(b) (20 pts)

1. Discuss the efficiency of the IS method here.
2. Relative to the crude MC method, are there some situations where IS works better than others?
3. Try to formulate whether any choices of α, β, and t result in a fairly good performance by IS, and why.
4. Can you think of an alternative IS distribution then the Uniform that might be better suited (and why) to estimate $\mathrm{M}_X(t)$?

This is an open-ended question. The uniform distribution was used for convenience, so we want to know does it ever work well, and if not, what might work better.

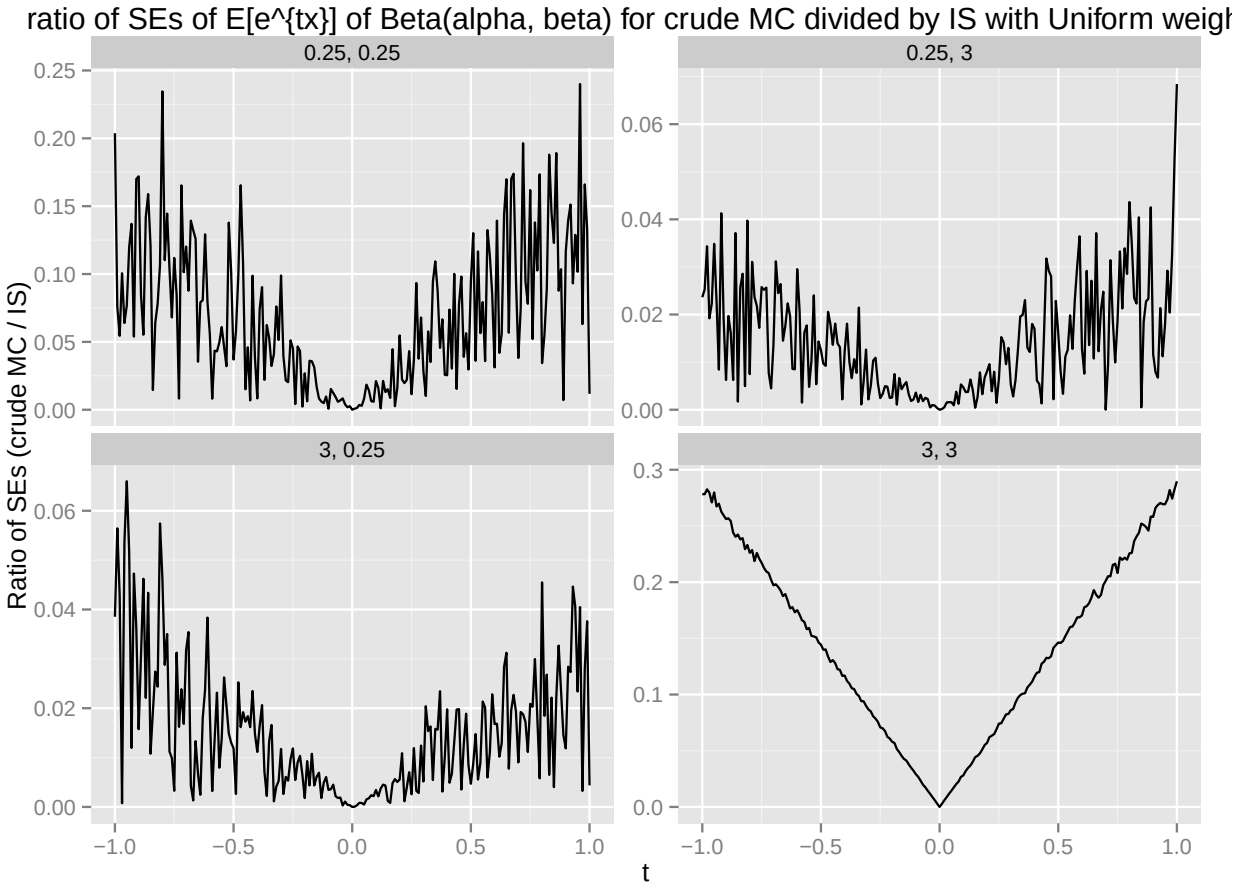
Solution: The plots below indicate that the method using the Beta distribution has extremely small standard errors, but the importance sampling method based on the uniform distribution has large and unpredicable standard errors.

```
# mean plots, ggplot
p <- ggplot(is.df, aes(x = t, y = mu, colour = method))
p <- p + geom_line()
p <- p + geom_ribbon(aes(ymin = lci, ymax = uci, fill = method), alpha = 0.2)
p <- p + facet_wrap(~ alpha + beta, scales = "free_y", ncol = 2)
p <- p + labs(title = "E[e^{tx}] of Beta(alpha, beta) for crude MC and IS with Uniform weights")
p <- p + ylab("mean +- 1.96 SE(mean)")
print(p)
```



The ratio of standard errors below are all less than 1, supporting the claim above that the crude MC estimates are much less variable than the IS estimates.

```
# ratio plots, ggplot
p <- ggplot(is.df, aes(x = t, y = se.ratio, group = method))
p <- p + geom_line()
p <- p + facet_wrap(~ alpha + beta, scales = "free_y", ncol = 2)
p <- p + labs(title = "ratio of SEs of E[e^{tx}] of Beta(alpha, beta) for crude MC divided by IS
p <- p + ylab("Ratio of SEs (crude MC / IS)")
print(p)
```



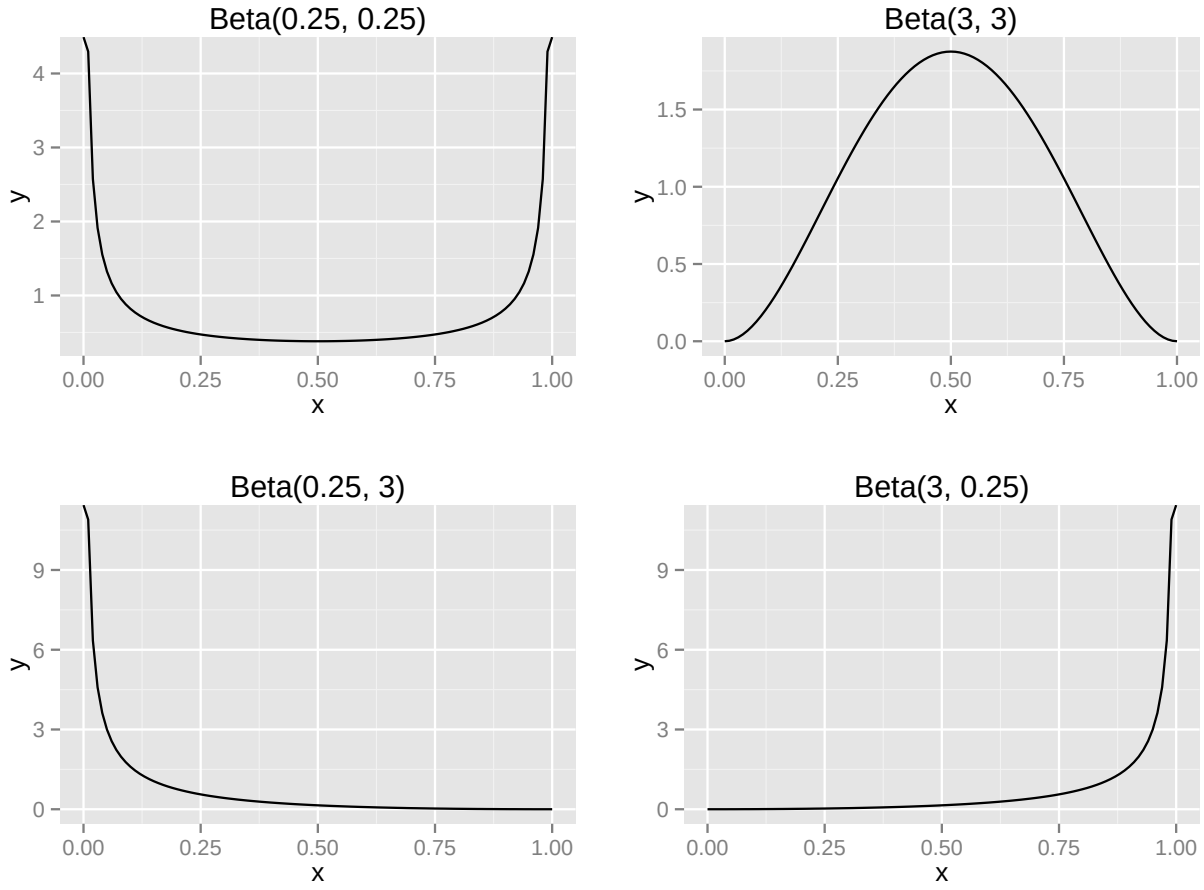
Below are the beta distributions we would like to draw from. If this is not possible, we can perform importance sampling. The distribution we draw from should mimic these beta distributions as closely as possible.
For the Beta$(0.25, 0.25)$ distribution, I would use two semi-triangular distributions and a uniform distribution in the middle, to mimic the U-shape. For the Beta$(3, 3)$ distribution, a triangular distribution might work well. For the Beta$(0.25, 3)$ distribution, a truncated exponential distribution. For the Beta$(3, 0.25)$ distribution, a reversed truncated exponential distribution.
In all cases,

```
# plot function
library(ggplot2)

pp <- list() # create a list of plots of Beta(a,b)
for (i.p in 1:4) {
  p <- ggplot(data.frame(x = c(0, 1)), aes(x))
  p <- p + stat_function(fun = dbeta, arg = list(shape1 = beta.param$alpha[i.p], shape2 = beta.p
  p <- p + labs(title = paste("Beta(", beta.param$alpha[i.p], ", ", beta.param$beta[i.p], ")", s
  pp[[i.p]] <- p
}

library(gridExtra)
```

```
grid.arrange( pp[[1]]
            , pp[[2]]
            , pp[[3]]
            , pp[[4]], ncol=2)
```



(30pts) **3. Control variates**

For the multinomial goodness-of-fit problem $\underset{\sim}{X} = \{X_1, X_2, \ldots, X_k\}$ has a Multinomial$(m, \underset{\sim}{\theta})$ distribution, where $\underset{\sim}{\theta} = (\theta_1, \ldots, \theta_k)$ such that $\theta_i > 0$ and $\sum_i^k \theta_i = 1$. The likelihood ratio statistic for testing $H_0 : \theta_1 = \theta_{01}, \ldots, \theta_k = \theta_{0k}$, versus $H_1 :$ not H_0, is

$$G^2 \;=\; 2\sum_{i=1}^{k} x_i \log_e\left(\frac{x_i}{m\theta_{0i}}\right),$$

where $0\log_e(0) \equiv 0$.

For a given value of m, k, and $\underset{\sim}{\theta}_0 = (\theta_{01}, \ldots, \theta_{0k})$ below you will be asked to generate n Multinomial samples, and from each compute G^2 and the Pearson statistic P. That is, get $G_1^2, \ldots, G_n^2$ and $P_1, \ldots, P_n$.

The crude MC estimate of $\mathrm{E}(G^2)$ is

$$\hat{\mu}_{\mathrm{C}} \;=\; \frac{1}{n}\sum_{i=1}^{n} G_i^2,$$

while the estimate based on using P as a control variate is

$$\hat{\mu}_{\mathrm{CV}} \;=\; \frac{1}{n}\sum_{i=1}^{n}\{G_i^2 - P_i\} + (k-1).$$

Based on the set of n samples, also compute $\widehat{\mathrm{Var}}[\hat{\mu}_{\mathrm{C}}]$, $\widehat{\mathrm{Var}}[\hat{\mu}_{\mathrm{CV}}]$, and $\mathrm{Corr}[G^2, P]$. Tabulate the two estimates, the standard error of the two estimates, and the correlation for the following scenarios:

1. $\theta_{0i} = k^{-1}$ for $i = 1, \ldots, k$ (equal cell probabilities) with $k = 10, 25, 50$ and $m = 200$
2. $\theta_{01} = 0.9 + 0.1/k$ and $\theta_{0i} = 0.1/k$ for $i = 2, \ldots, k$ with $k = 10, 25, 50$ and $m = 200$

Consider two choices for n

1. $n = 2000$
2. n selected so that the "error" in $\hat{\mu}_{\mathrm{C}}$ is small, for example

$$2\sqrt{\widehat{\mathrm{Var}}[\hat{\mu}_{\mathrm{C}}]} \;\le\; c\mathrm{E}[G^2] \;=\; c\hat{\mu}_{\mathrm{C}}$$

where c is small, say $c = 0.01$. In this case, $\hat{\mu}_{\mathrm{C}}$ should be within $2\sqrt{\widehat{\mathrm{Var}}[\hat{\mu}_{\mathrm{C}}]}$ of $\mathrm{E}(G^2)$ with probability 0.95, that is, with high probability relative to the very small error in $\hat{\mu}_{\mathrm{C}}$. Noting that

$$\widehat{\mathrm{Var}}[\hat{\mu}_{\mathrm{C}}] \;=\; \frac{\mathrm{Var}[G^2]}{n},$$

you might choose n assuming, for simplicity, that

$$\mathrm{E}[G^2] \;=\; \mathrm{E}[\chi^2_{k-1}] \;=\; k-1,$$

but

$$\mathrm{Var}[G^2] \;\doteq\; 2\mathrm{Var}[\chi^2_{k-1}] \;=\; 2\times 2(k-1) \;=\; 4(k-1).$$

Remark: You created code in earlier problems to generate multiple multinomial samples and functions to compute P and G^2 from multiple samples. Use this code here. If possible, you might think of automating things by writing a script that loops over all combinations of k, m, n, and $\underset{\sim}{\theta}$.

Solution: First I set up the simluation and create data.frame `df.cv` with the results.

```
## START Function from previous homework
# A helpful function for checking permissibility of theta
f.dmulti.error.check <- function(theta) {
  if (any(theta < 0) | any(theta > 1)) {
    warning("Proportions in theta must between 0 and 1")
    return(1)
  }
  if (abs(sum(theta) - 1) > 1e-10 ) {
    warning("Sum of theta vector is not 1")
    return(1)
  }
  if (length(theta) < 2 ) {
    warning("theta must be a vector of at least 2")
    return(1)
  }
  return(0)
}

## Multinomial random deviates via conditional binomial samples
# n = number of samples to draw
# m = total count
# theta = row vector of proportions.
# returns a matrix with each sample in a row.
f.dmulti <- function(theta, m, n = 1) {

  # basic error checking
  if(f.dmulti.error.check(theta) == 1) { return(NULL) }
  if (m <= 0) {
    warning("m must be greater than 0")
    return(NULL)
  }

  k <- length(theta) # number of categories
  x <- matrix(NA, ncol = k, nrow = n) # init return matrix

  # conditional probatilities for all steps
  theta.cond <- c(theta[1], theta[2:k]/(1 - cumsum(theta)[1:(k - 1)]))

  # for each of the first k - 1 bins
  for (i.k in 1:(k - 1)) {
    # determine how many observations we have left to allocate
    m.temp <- m - apply(x, 1, sum, na.rm = TRUE) # remaining sample size to draw
    # draw a binonmial rv with m.temp trials with the binomial conditional probabilities
    x[, i.k] <- rbinom(n, m.temp, theta.cond[i.k])
  }
  # put the rest of the sample in the last bin
  m.temp <- m - apply(x, 1, sum, na.rm = TRUE) # remaining sample size to draw
  x[, k] <- m.temp

  return(x)
}
## END Function from previous homework
```

```
df.cv <- NULL # init location to store results

for (i.m in 200) {
  for (i.k in c(10,25, 50)) {
    for (i.n in c(0, 2000)) {
      if (i.n == 0) { # n depending on c and k
        c <- 0.01
        i.n <- ceiling((4 / c)^2 / (i.k - 1))
      }
      for (i.t in c(1,2)) {
        # two versions of theta
        if (i.t == 1) {
          theta <- rep(1, i.k) / i.k
        }
        if (i.t == 2) {
          theta <- 0.1 * rep(1, i.k) / i.k
          theta[i.k] <- 0.9 + theta[i.k]
        }

        # draw multinomial samples
        x <- f.dmulti(theta, m = i.m, n = i.n)
```

```
        # calculate statistics
        out.P  <- f.P (theta, x) # Pearson's P
        out.G2 <- f.G2(theta, x) # likelihood G^2

        cv <- out.G2$G2 - out.P$P # control variate

        # crude estimate
        mu.c  <- mean(out.G2$G2)
        v.c   <- var(out.G2$G2)
        moe.c <- 2 * sqrt(v.c / i.n)
        lci.c <- mu.c - moe.c
        uci.c <- mu.c + moe.c

        # control variate
        mu.cv  <- mean(cv) + (i.k - 1)
        v.cv   <- var(cv)
        moe.cv <- 2 * sqrt(v.cv / i.n)
        lci.cv <- mu.cv - moe.cv
        uci.cv <- mu.cv + moe.cv

        cor.G2P <- cor(out.G2$G2, out.P$P)

        # bind results to previous results
        df.cv <- rbind(df.cv
                     , data.frame(
                            i.m     = i.m
                          , i.k     = i.k
                          , i.n     = i.n
                          , i.t     = i.t
                          , mu.c    = mu.c
                          , moe.c   = moe.c
                          #, lci.c   = lci.c
                          #, uci.c   = uci.c
                          , mu.cv   = mu.cv
                          , moe.cv  = moe.cv
                          #, lci.cv  = lci.cv
                          #, uci.cv  = uci.cv
                          , cor.G2P = cor.G2P)
                       )
      }
    }
  }
}
```

The table below gives the results of the MC estimates based on the crude $\hat{\mu}_c$ and control variate $\hat{\mu}_{cv}$, with 95% CI margin-of-errors (MOEs), and correlation between the Pearson P and Likelihood G^2 statistics. The table is sorted by scenario, k, then n.
n was selected for the MC error to be small, using $c = 0.01$ giving $n = ((4/c)^2/(k-1))$.
In those cases, the sample size is larger, but the CI is wider than the specified $4c = 0.04$.
The control variate MOE is much narrower than the crude MOE. The control variate does not reduce the MOE as much in the second $\underset{\sim}{\theta}$ condition.

	i.m	i.k	i.n	i.t	mu.c	moe.c	mu.cv	moe.cv	cor.G2P
1	200	10	17778	1	9.023	0.064	9.088	0.008	0.993
2	200	10	17778	2	10.315	0.068	10.237	0.041	0.828
3	200	10	2000	1	9.271	0.194	9.096	0.023	0.993
4	200	10	2000	2	10.323	0.203	10.278	0.123	0.832
5	200	25	6667	1	24.644	0.174	24.614	0.037	0.977
6	200	25	6667	2	26.864	0.127	26.784	0.121	0.866
7	200	25	2000	1	24.550	0.319	24.614	0.065	0.979
8	200	25	2000	2	27.054	0.231	26.720	0.225	0.868
9	200	50	3266	1	52.278	0.373	52.259	0.122	0.945
10	200	50	3266	2	46.032	0.200	45.986	0.321	0.976
11	200	50	2000	1	52.034	0.490	52.280	0.153	0.951
12	200	50	2000	2	45.965	0.255	46.062	0.407	0.976