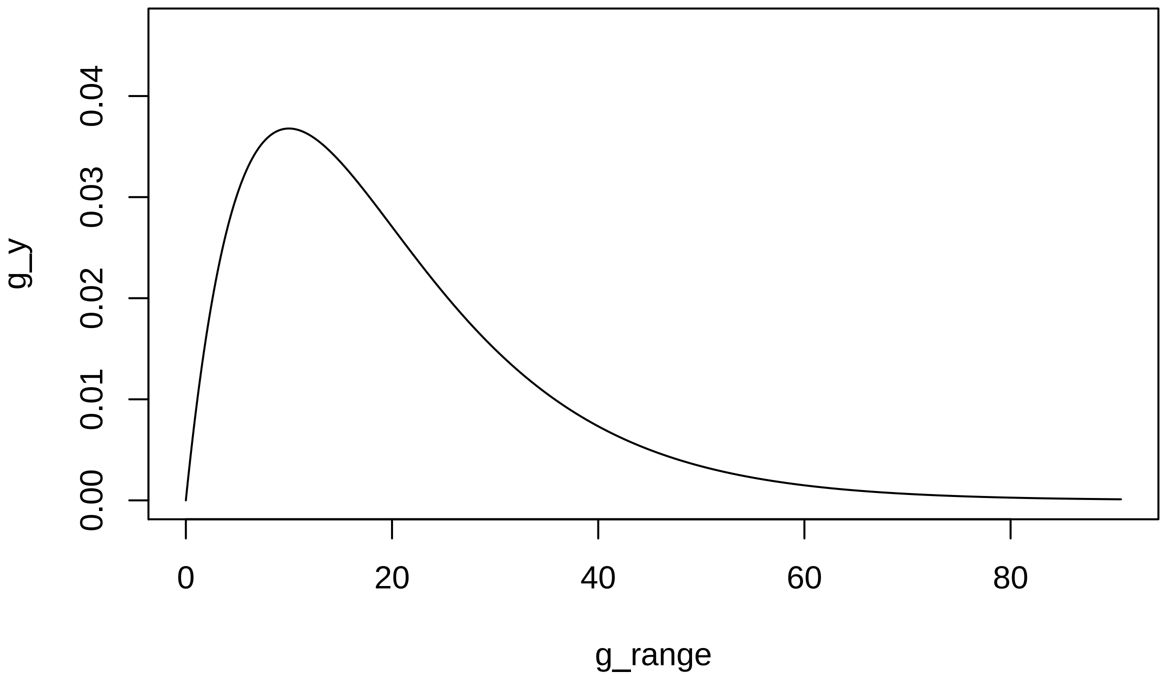
JAGS test

Simulate data

```
I <- 50
K <- 2
S <- 10

# choose diffuse priors for gamma
a_gamma <- 2
b_gamma <- 10

avrg <- a_gamma * b_gamma
std.dv <- sqrt(a_gamma*b_gamma^2)
g_range = seq(0, avrg + 5*std.dv, 0.01)
g_y = dgamma(g_range, a_gamma, rate = 1/b_gamma)
plot(g_range, g_y, type = "l", ylim=c(0, max(g_y) + 0.01))
```



```
set.seed(123)

a <- matrix(NA, nrow=K, ncol=S)
b <- matrix(NA, nrow=K, ncol=S)
for (s in 1:S) {
  a[, s] <- rgamma(K, a_gamma, rate = 1/b_gamma)
  b[, s] <- rgamma(K, a_gamma, rate = 1/b_gamma)
}

pi <- as.vector(rdirichlet(1, rep(1, K)))
z <- sample(1:K, size = I, replace = T, prob = pi)
```

```r
w <- matrix(NA, nrow=I, ncol=S)
for (s in 1:S) {
  w[, s] <- rbeta(I, a[,s][z], b[,s][z])
}

tcn <- matrix(2, nrow=I, ncol=S)
m <- matrix(rep(sample(1:2, size = I, replace = T), S), nrow=I, ncol=S)

calcTheta <- function(m, tcn, w) {
  (m * w) / (tcn * w + 2*(1-w))
}
theta <- calcTheta(m, tcn, w)

n <- replicate(S, rpois(I, 100))
y <- matrix(NA, nrow=I, ncol=S)
for (i in 1:I) {
  for (s in 1:S) {
    y[i, s] <- rbinom(1, n[i, s], theta[i,s])
  }
}
```

JAGS

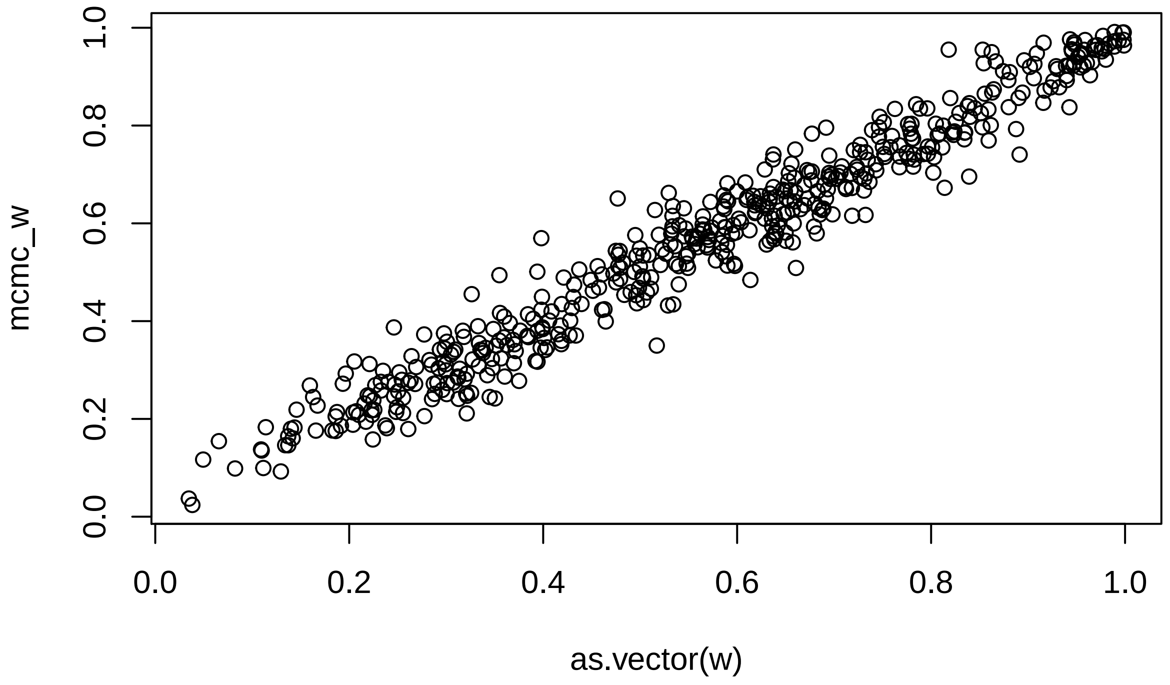
```r
jags.file <- file.path(working.dir, "model_ab.jags")

test.data <- list("I" = I, "S" = S, "K" = K,
                  "y" = y, "n" = n,
                  "m" = m, "tcn" = tcn)
jags.m <- jags.model(jags.file, test.data,
                     n.chains = 1,
                     n.adapt = 100)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 500
##    Unobserved stochastic nodes: 591
##    Total graph size: 6389
##
## Initializing model
```

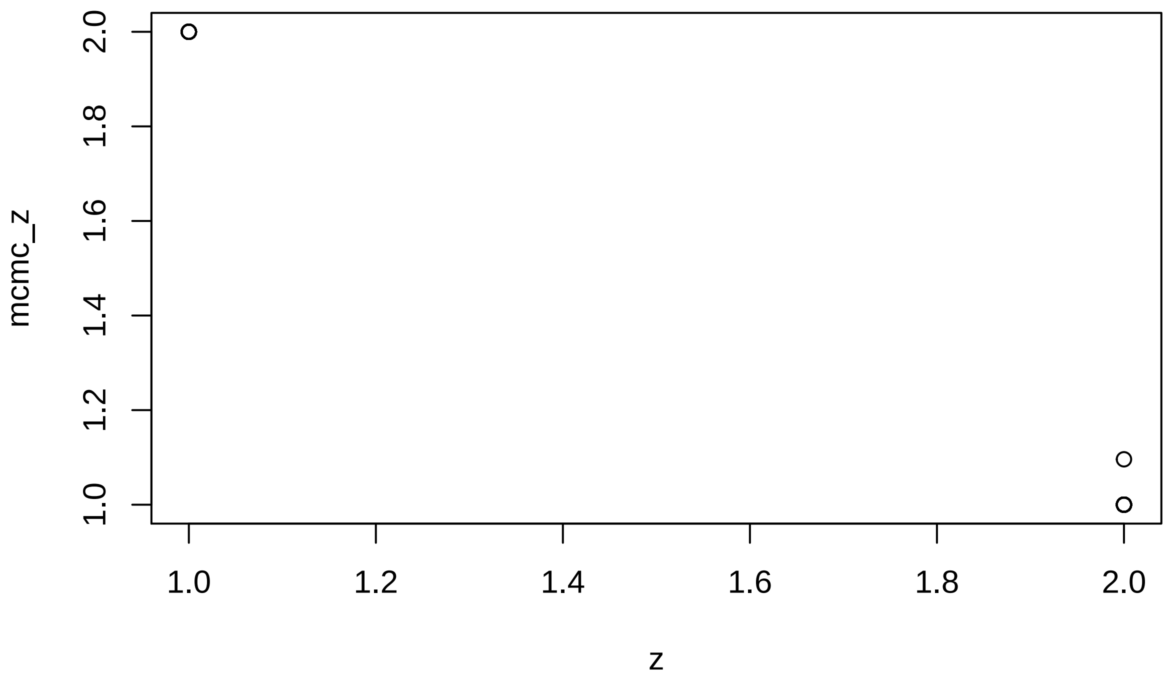
```r
params <- c("z", "w", "a", "b")
samps <- coda.samples(jags.m, params, n.iter=1000)
s <- summary(samps)
pdf(file.path(working.dir, "trace-plots.pdf"))
plot(samps)
dev.off()
```

```
## pdf
##   2
```

```
mcmc_vals <- s$statistics
mcmc_w <- mcmc_vals[substr(rownames(mcmc_vals), 1, 1) == "w", "Mean"]
plot(as.vector(w), mcmc_w, type = "p")
```

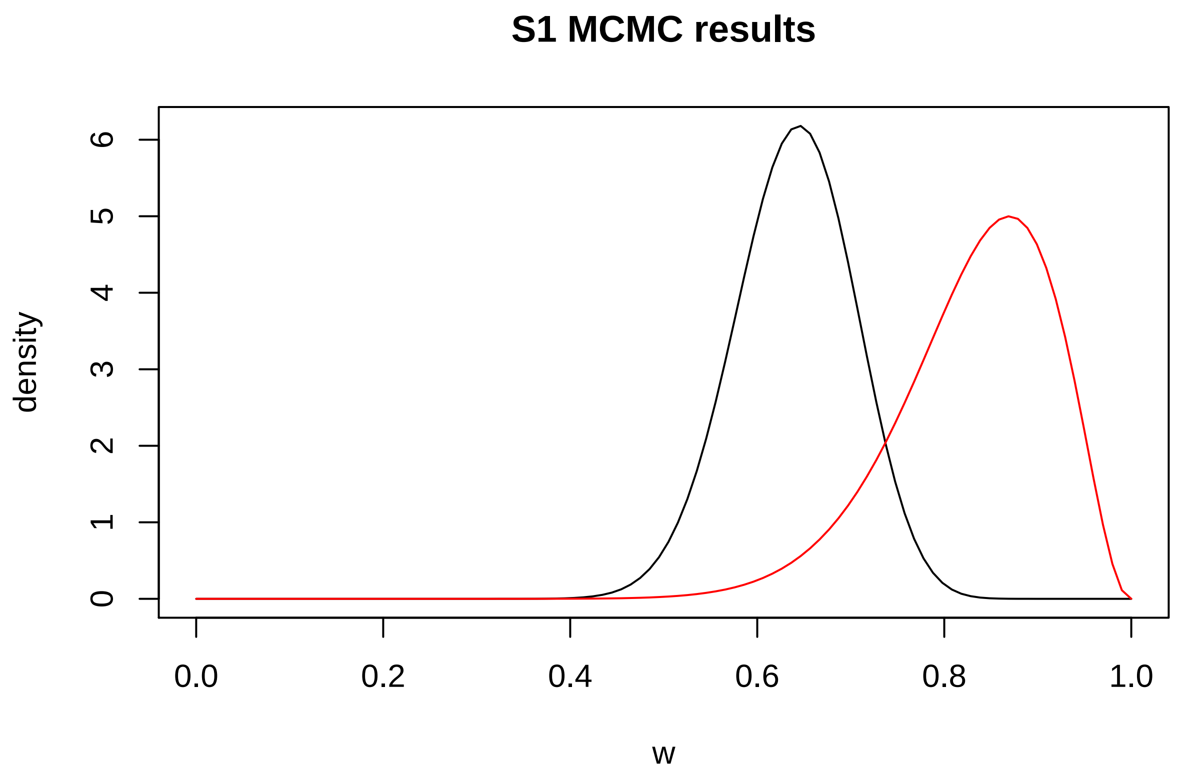


```
mcmc_z <- as.vector(mcmc_vals[substr(rownames(mcmc_vals), 1, 1) == "z", "Mean"])
#mcmc_z <- round(mcmc_z, 0)
plot(z, mcmc_z, type = "p")
```

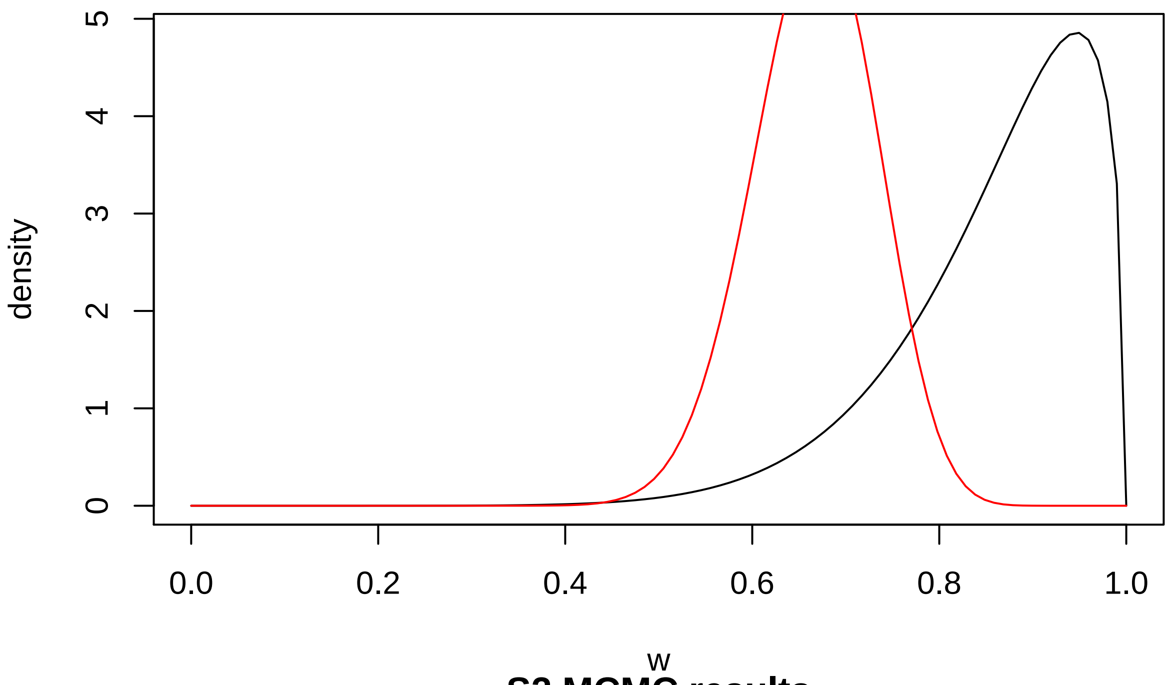


```
mcmc_a <- mcmc_vals[substr(rownames(mcmc_vals), 1, 1) == "a", "Mean"]
mcmc_a <- matrix(mcmc_a, nrow=K)
mcmc_b <- mcmc_vals[substr(rownames(mcmc_vals), 1, 1) == "b", "Mean"]
```

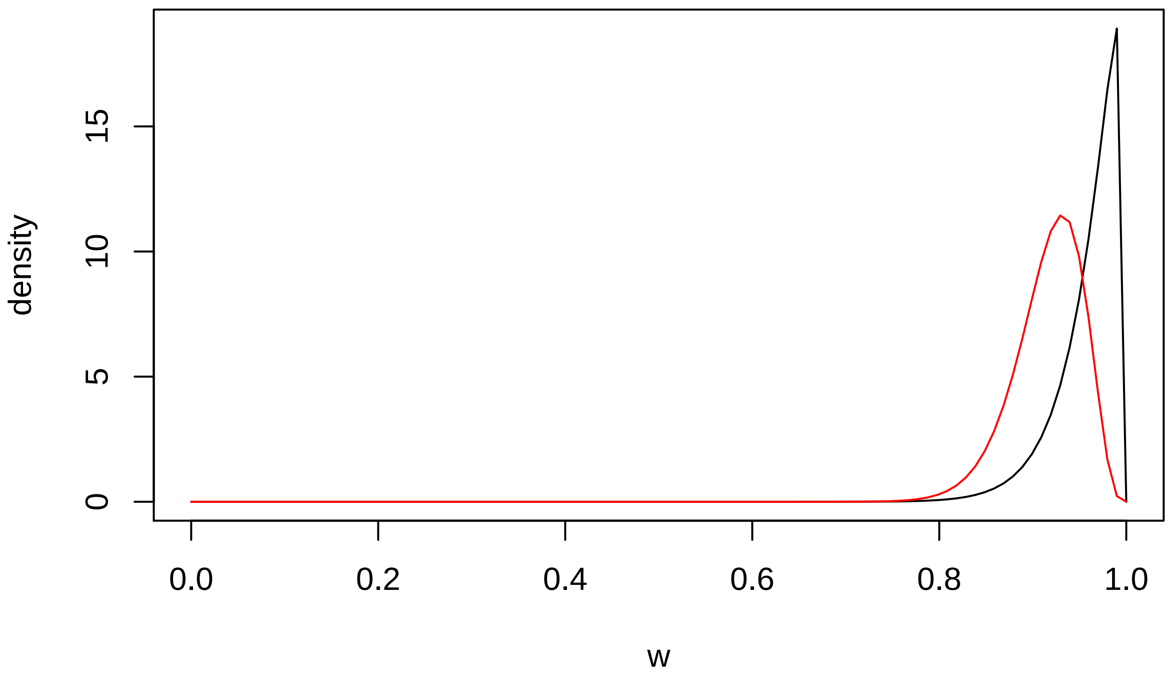
```r
mcmc_b <- matrix(mcmc_b, nrow=K)

p <- seq(0, 1, length = 100)
for (s in 1:S) {
  for (k in 1:K) {
    if (k == 1) {
      plot(p, dbeta(p, mcmc_a[k,s], mcmc_b[k,s]),
           main = paste0("S", s, " MCMC results"),
           ylab = "density", xlab = "w", type = "l", col = k)
    } else {
      lines(p, dbeta(p, mcmc_a[k,s], mcmc_b[k,s]), type = "l", col = k)
    }
  }

  for (k in 1:K) {
    if (k == 1) {
      plot(p, dbeta(p, a[k,s], b[k,s]),
           main = paste0("S", s, " truth"),
           ylab = "density", xlab = "w", type = "l", col = k)
    } else {
      lines(p, dbeta(p, a[k,s], b[k,s]), type = "l", col = k)
    }
  }
}
```

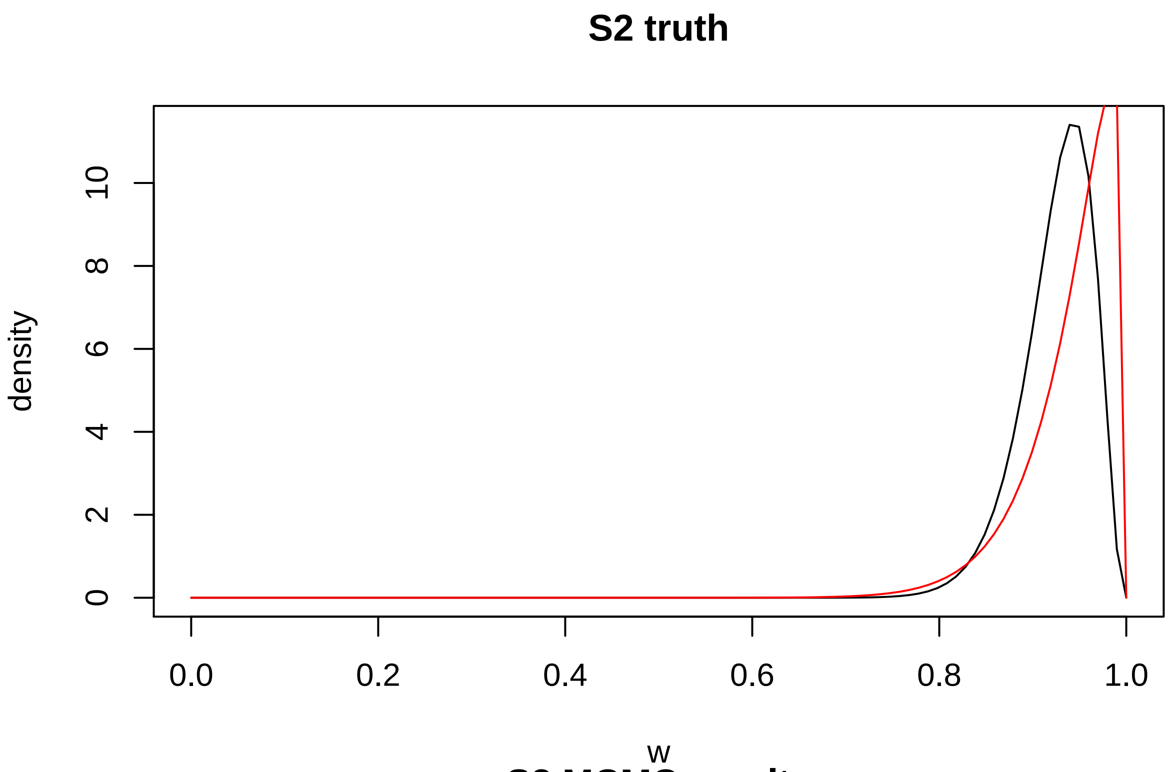


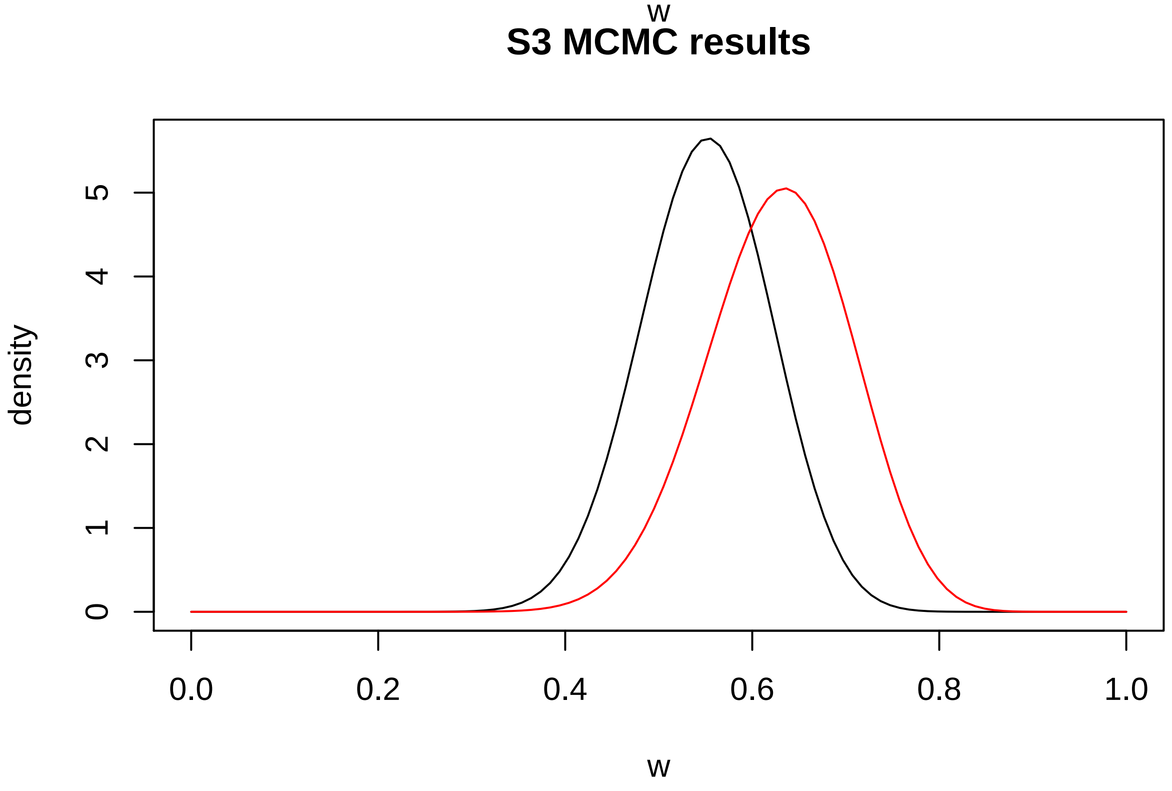
S1 truth

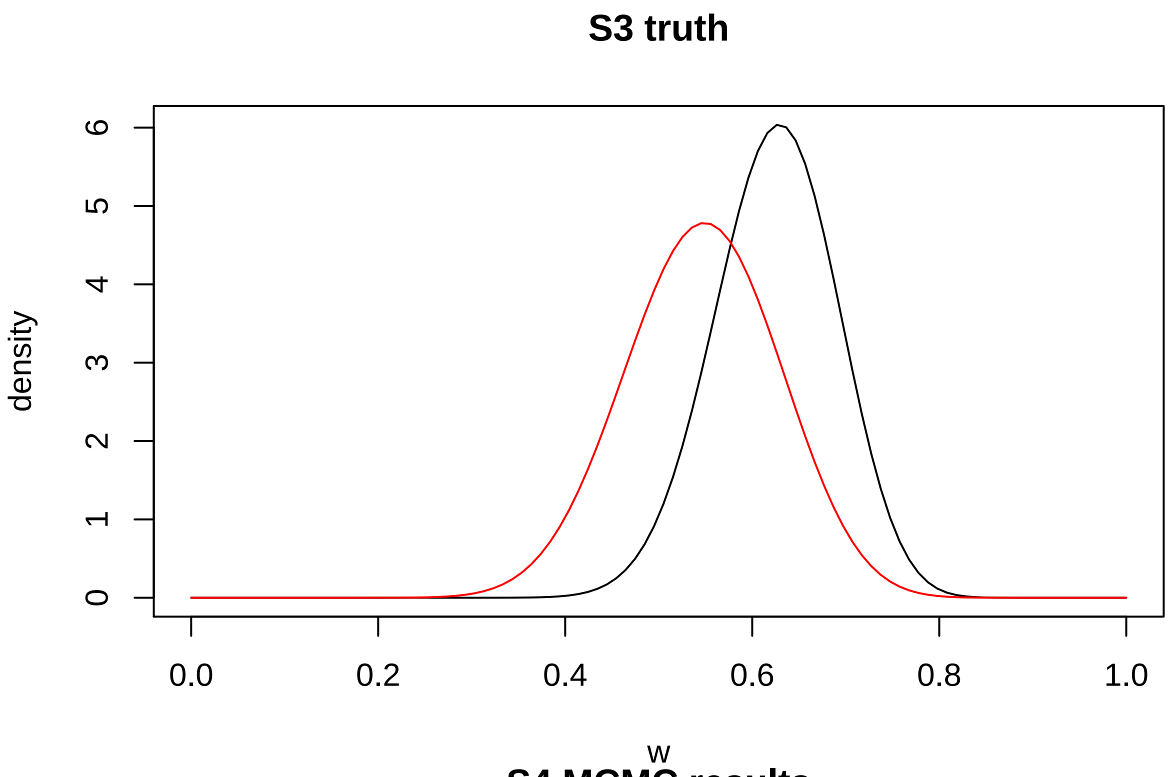


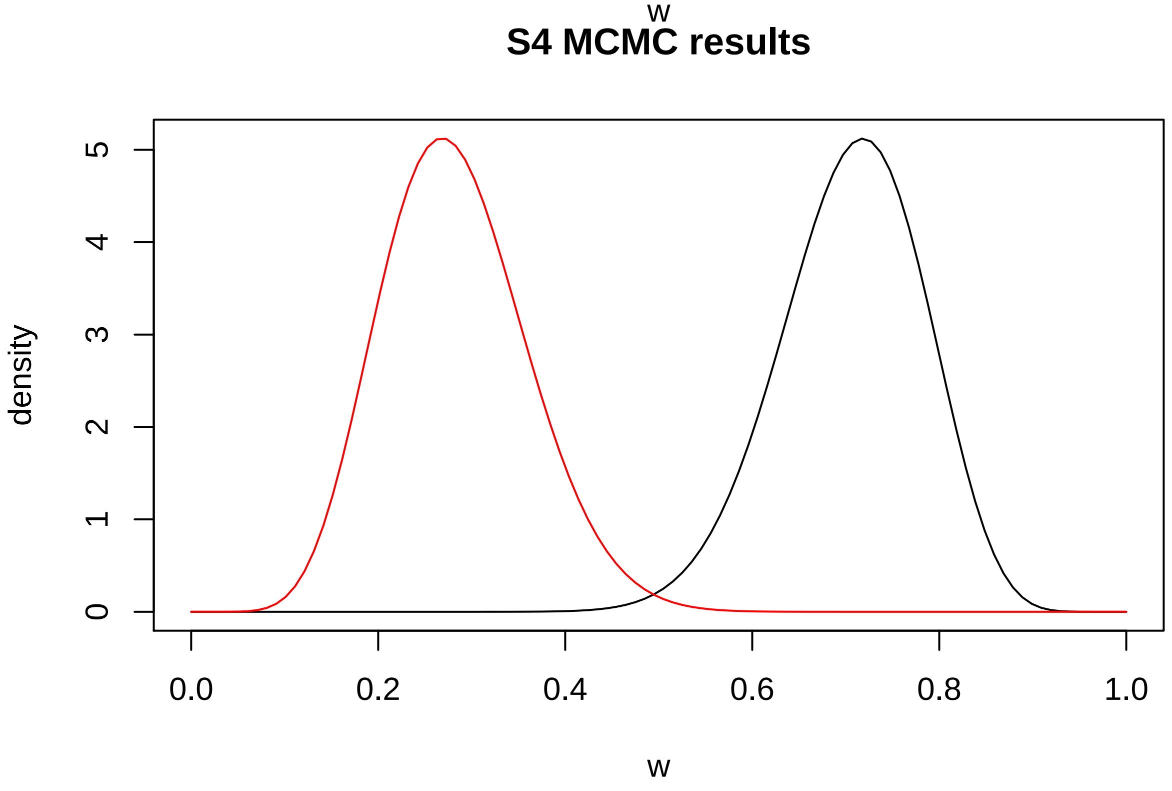
S2 MCMC results



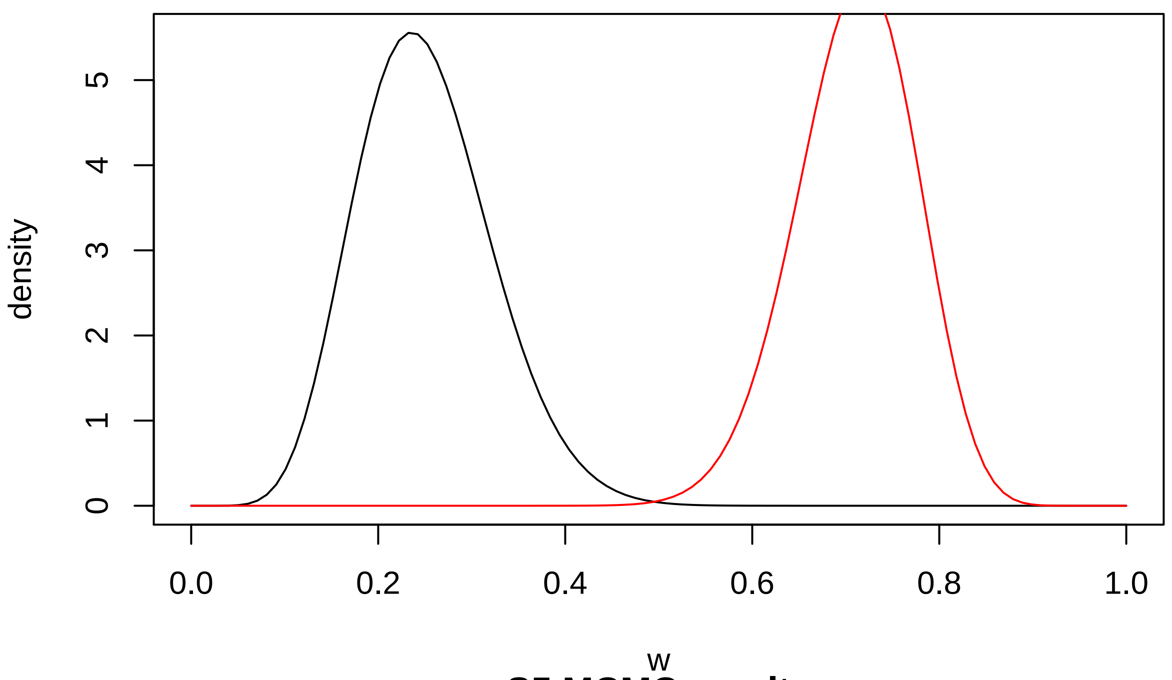

S2 truth
density
0
2
4
6
8
10
0.0
0.2
0.4
0.6
0.8
1.0
w


S3 MCMC results
density
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0
w

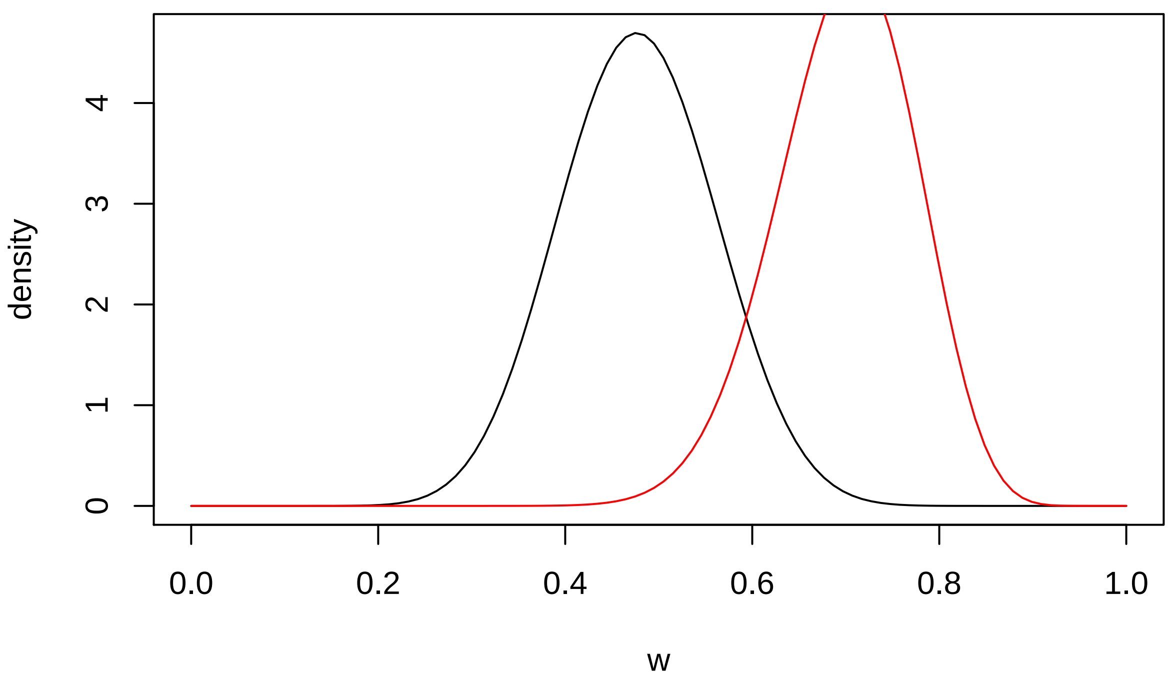

S3 truth
density
6
5
4
3
2
1
0
0.0
0.2
0.4
0.6
0.8
1.0
w

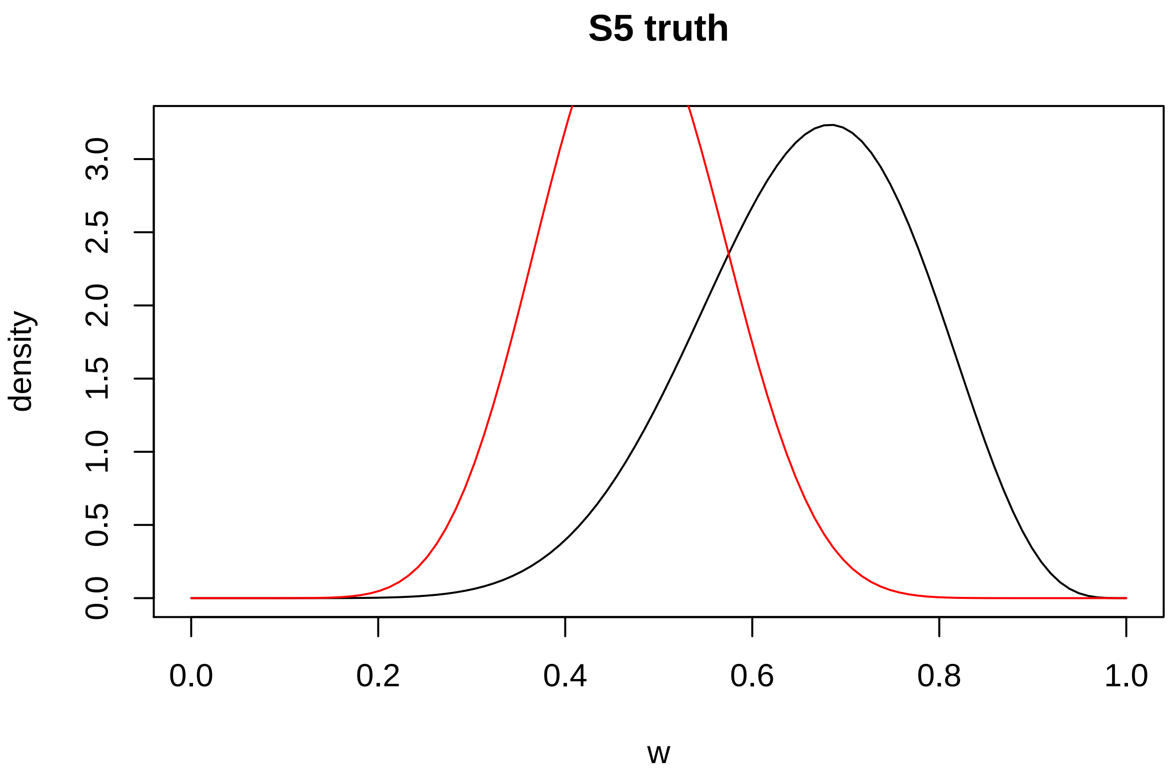

S4 MCMC results
density
5
4
3
2
1
0
0.0
0.2
0.4
0.6
0.8
1.0
w

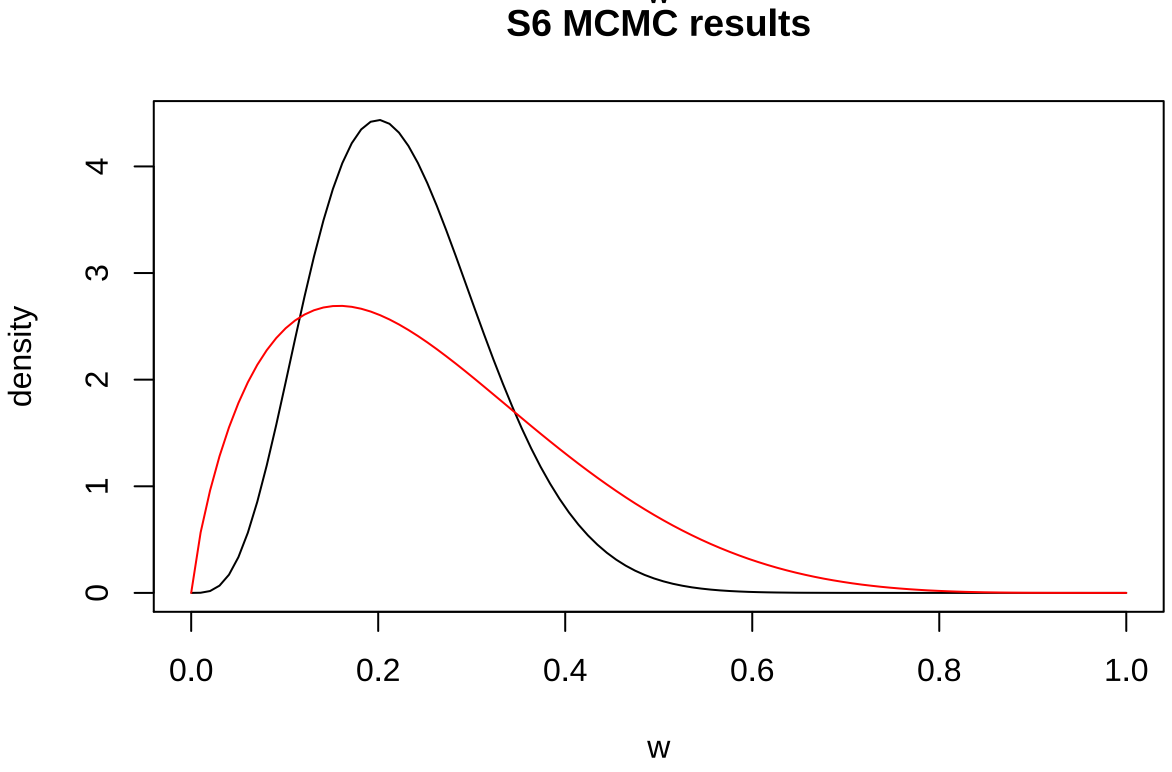
S4 truth



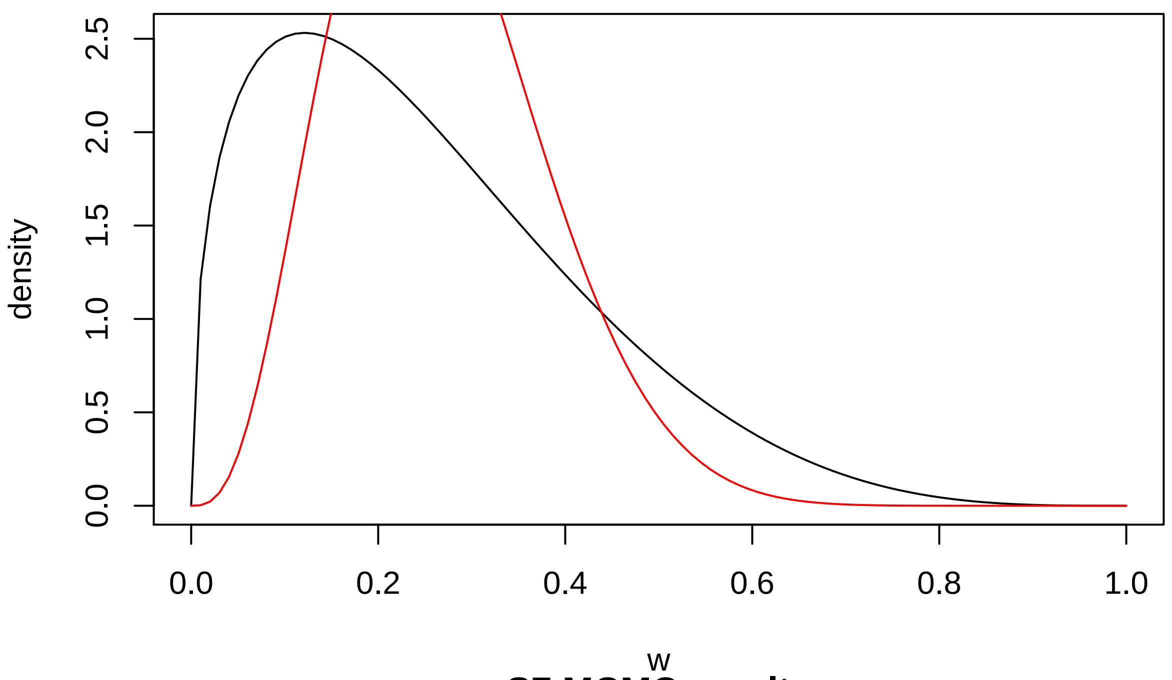
S5 MCMC results



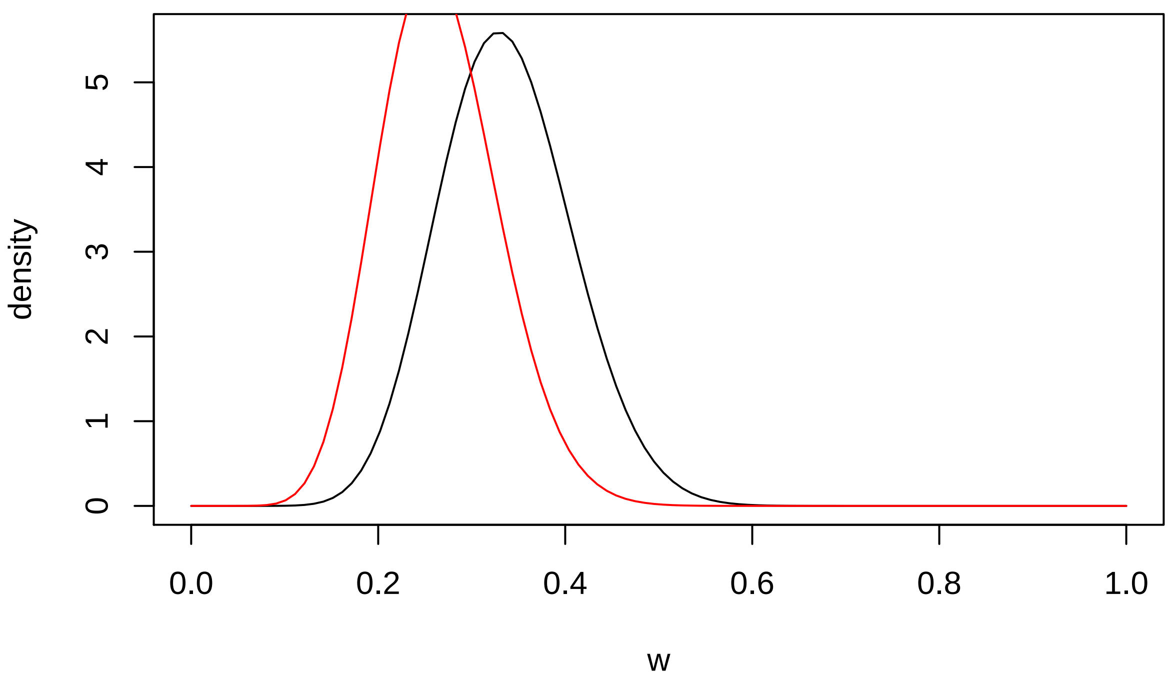

S5 truth
density
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
0.2
0.4
0.6
0.8
1.0
w


S6 MCMC results
density
0
1
2
3
4
0.0
0.2
0.4
0.6
0.8
1.0
w

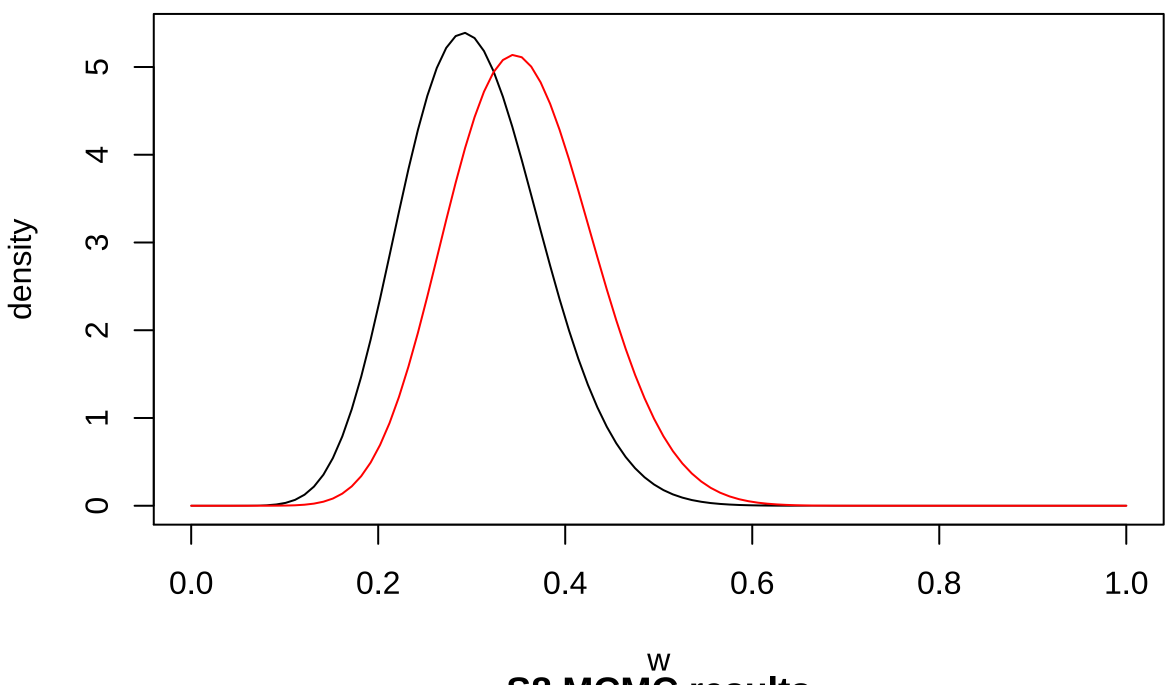
S6 truth



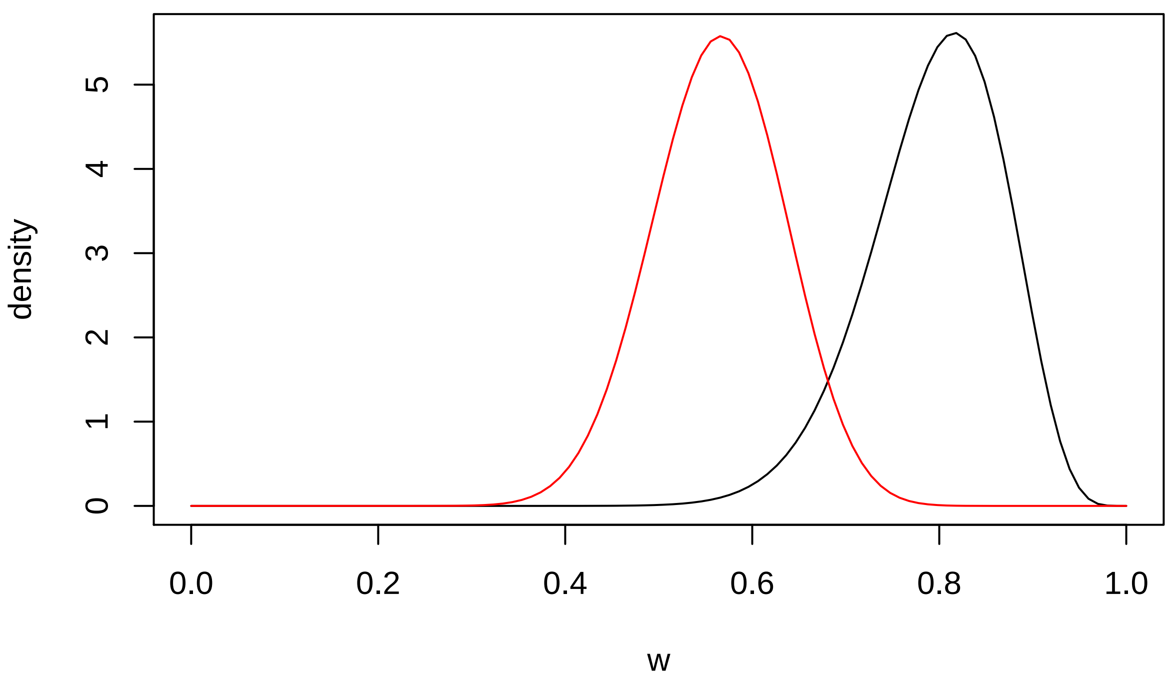
S7 MCMC results



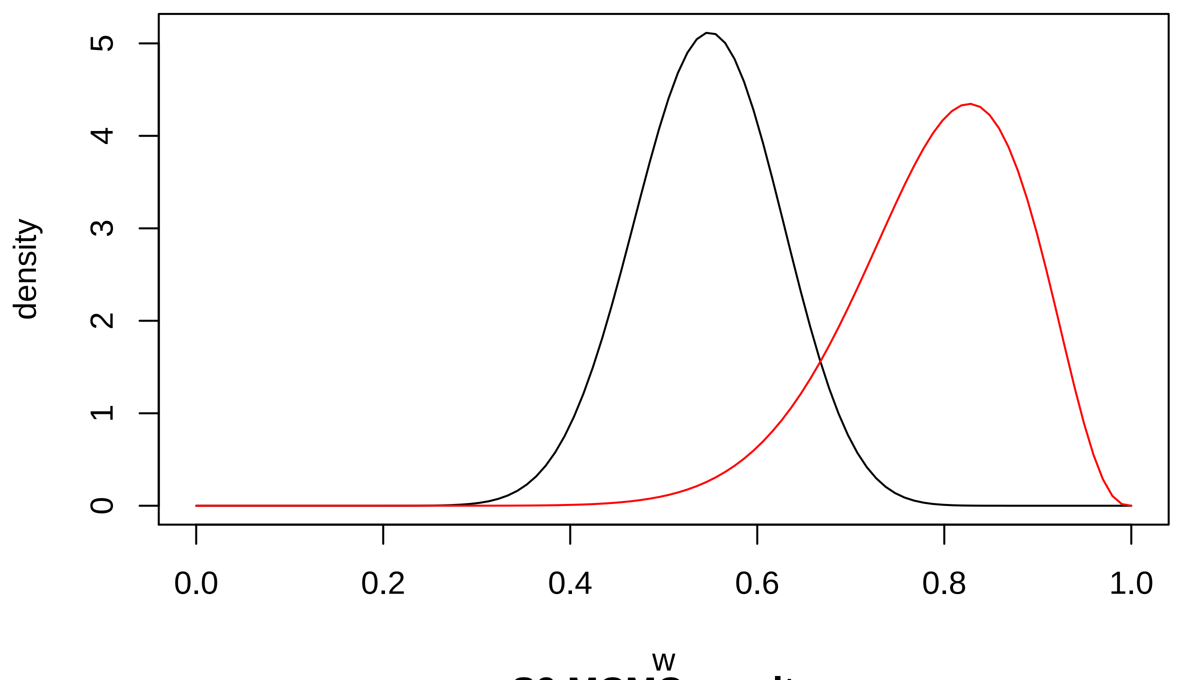
S7 truth



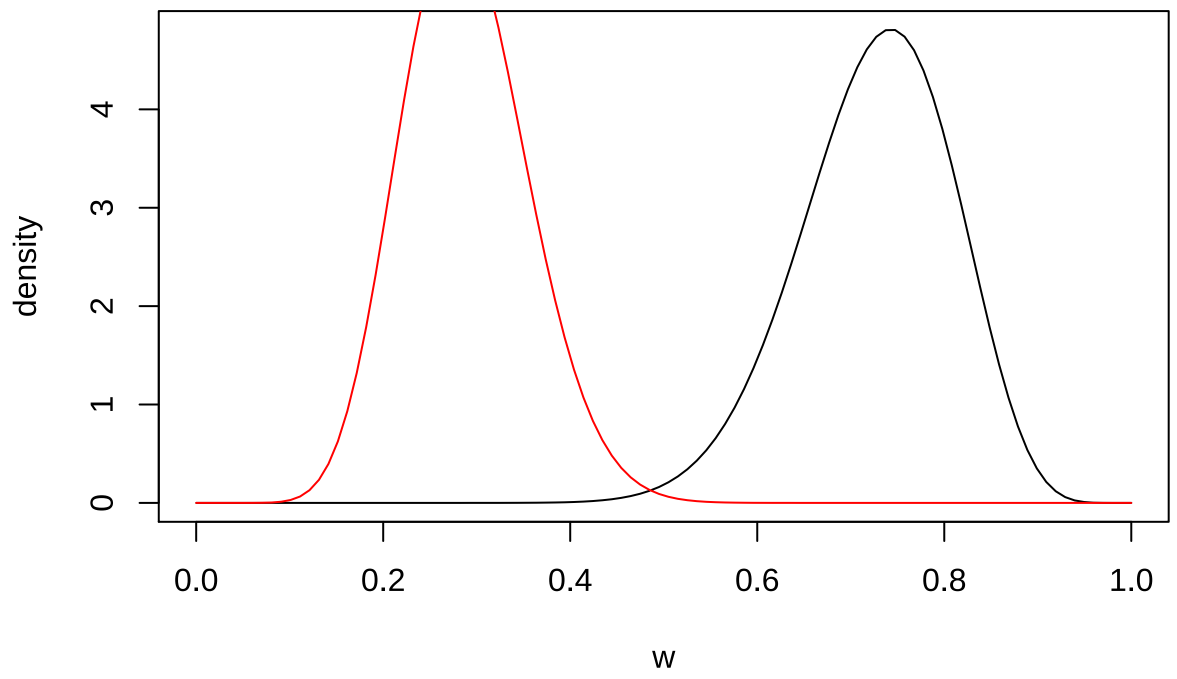
S8 MCMC results



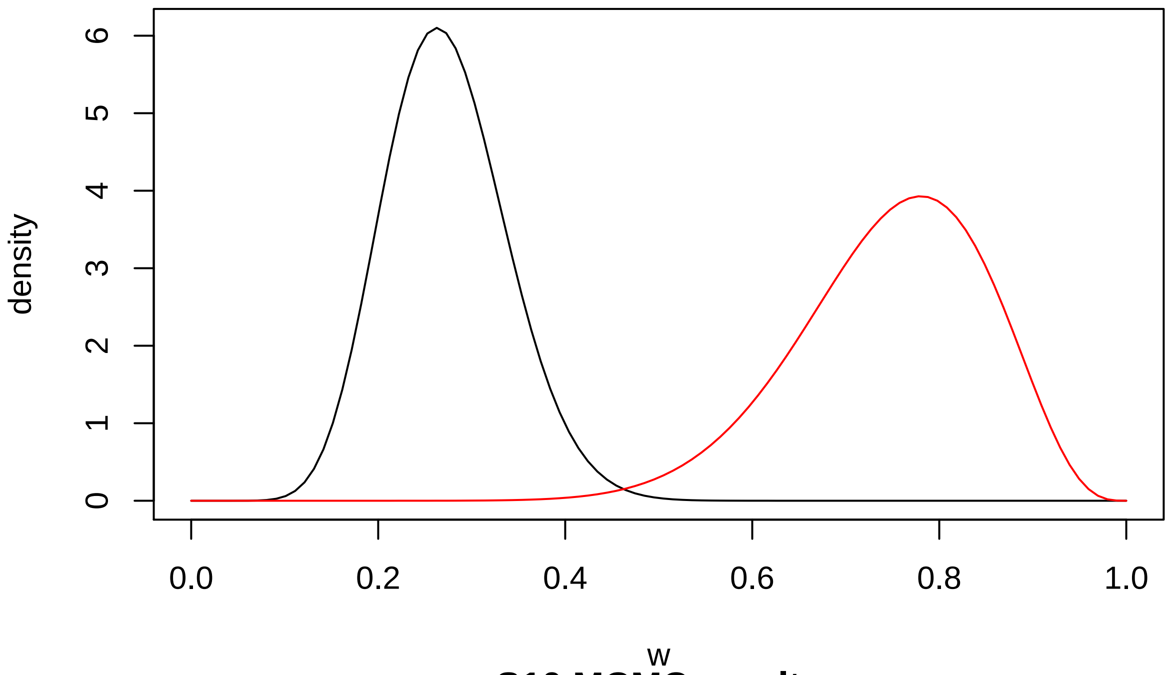
S8 truth



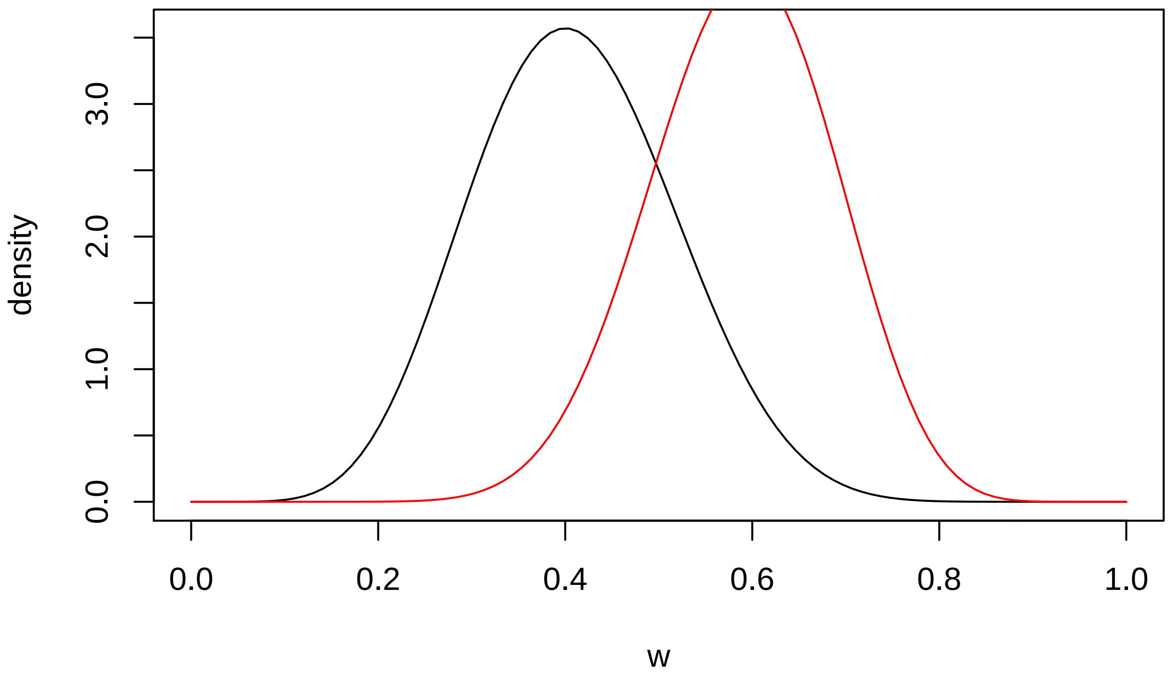
S9 MCMC results

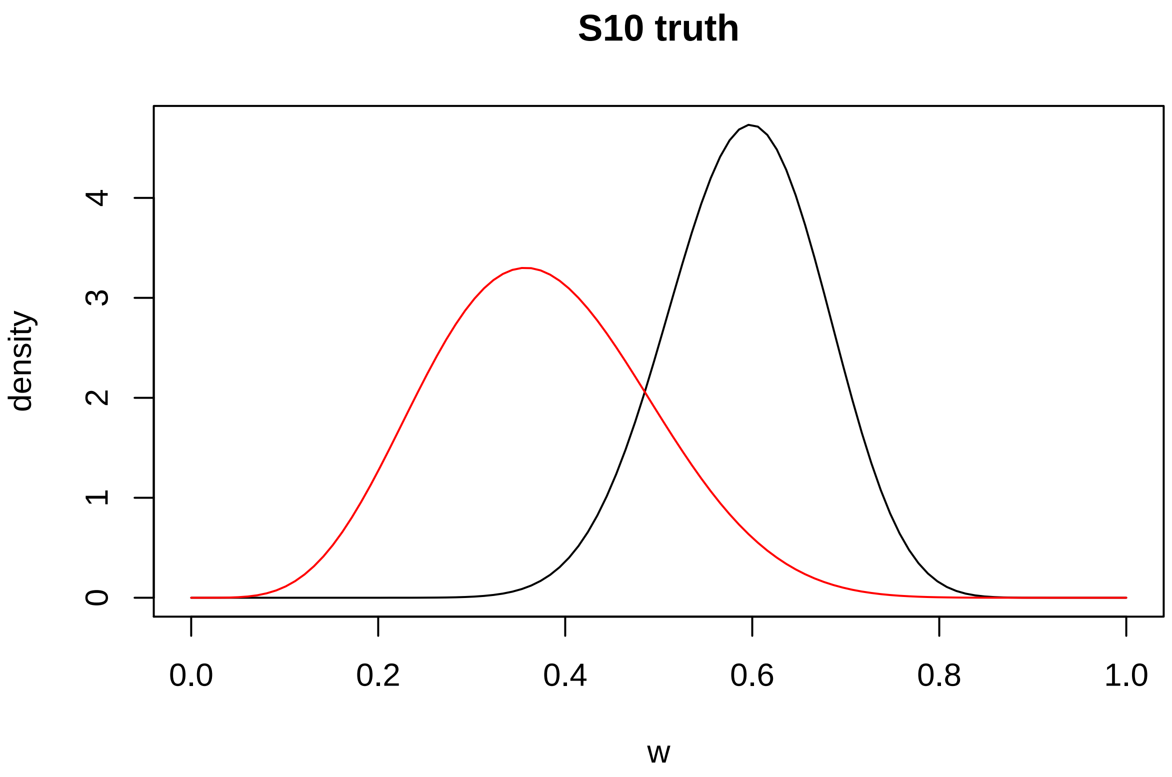


S9 truth



S10 MCMC results




S10 truth
density
w
0
1
2
3
4
0.0
0.2
0.4
0.6
0.8
1.0