

Version 3: 4 clusters

Simulate data

```
I <- 50
K <- 4
S <- 10

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

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)
}
colnames(a) <- colnames(b) <- paste0("sample", seq_len(ncol(a)))
# reorder a,b matrices to match ordering of means (U) in S1
U <- a/(a+b)
V <- a+b
ix <- order(U[, 1])
U.ordered <- U[ix, ]
a.ordered <- a[ix, ]
b.ordered <- b[ix, ]
V.ordered <- V[ix, ]

#pi <- as.vector(rdirichlet(1, rep(1, K)))
pi <- c(0.2, 0.3, 0.2, 0.3)
z <- sample(1:K, size = I, replace = T, prob = pi)

##
## omega (w) is the cancer cell fraction
## - shouldn't omega have dimension K x S
## - I'd like to be able to recover the true values of a and b
## -> a single random ordinate drawn from this distribution
##   might not be representative
##   an average over 1000 random ordinates guarantees that
##   we have a representative ordinate of beta(a, b)
w <- matrix(NA, nrow=K, ncol=S)
for(k in seq_len(K)){
  tmp <- rowMeans(replicate(1000,
                           rbeta(S, a.ordered[k, ],
                                   b.ordered[k, ])))

  w[k, ] <- tmp
}

tcn <- matrix(2, nrow=I, ncol=S)
```

```

m <- matrix(rep(sample(1:2, size = I, replace = T), S),
            nrow=I, ncol=S)
W <- w[z, ]
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])
  }
}

test.data <- list("I" = I, "S" = S, "K" = K,
                 "y" = y, "n" = n,
                 "m" = m, "tcn" = tcn)

```

Visualize densities of simulated data

Clustering is by ω

```

test.data <- list("I" = I, "S" = S, "K" = K,
                 "y" = y, "n" = n,
                 "m" = m, "tcn" = tcn)

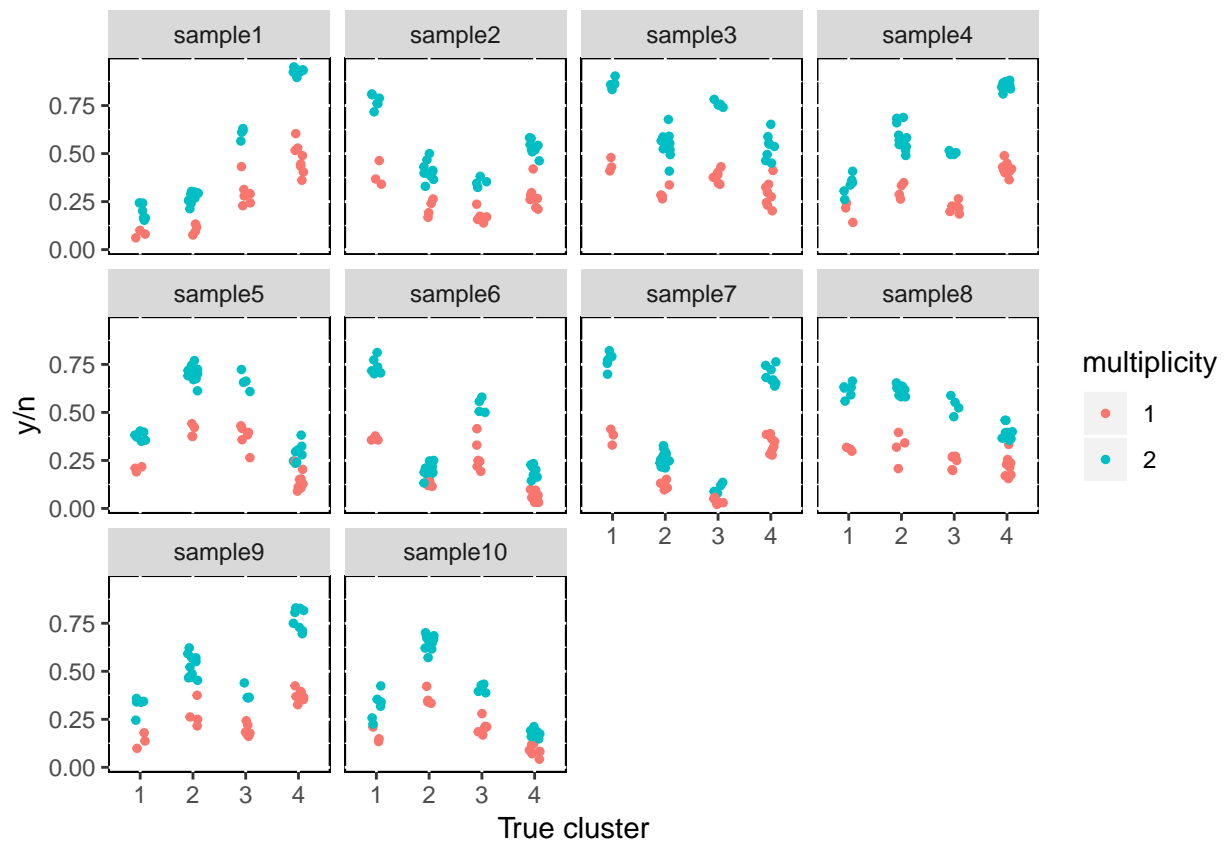
p <- test.data$y/test.data$n
colnames(w) <- colnames(p) <- paste0("sample", seq_len(ncol(p)))
colnames(m) <- colnames(p)

m2 <- m %>%
  as_tibble() %>%
  mutate(z=factor(z),
         variant_index=seq_len(nrow(.))) %>%
  gather("sample", "multiplicity", -c(z, variant_index)) %>%
  mutate(sample=factor(sample, levels=colnames(p)))

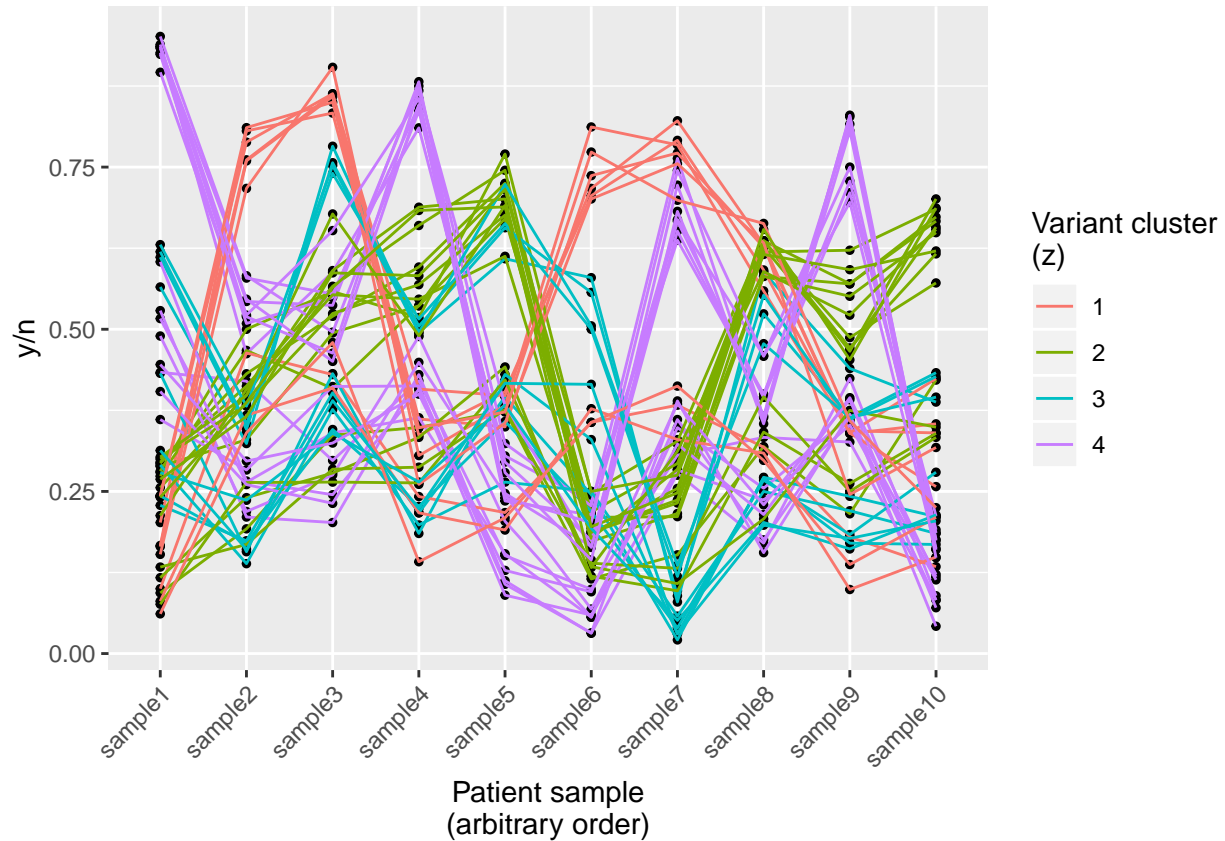
p2 <- p %>%
  as_tibble() %>%
  mutate(z=factor(z),
         variant_index=seq_len(nrow(.))) %>%
  gather("sample", "fraction", -c(z, variant_index)) %>%
  mutate(sample=factor(sample, levels=colnames(p))) %>%
  left_join(m2, by=c("z", "variant_index", "sample")) %>%
  mutate(multiplicity=factor(multiplicity))

## what the data looks like by sample
ggplot(p2, aes(z, fraction)) +
  geom_jitter(width=0.1, size=1, aes(color=multiplicity)) +
  facet_wrap(~sample) +
  ylab("y/n") +
  xlab("True cluster") +
  theme(panel.background=element_rect(fill="white", color="black"))

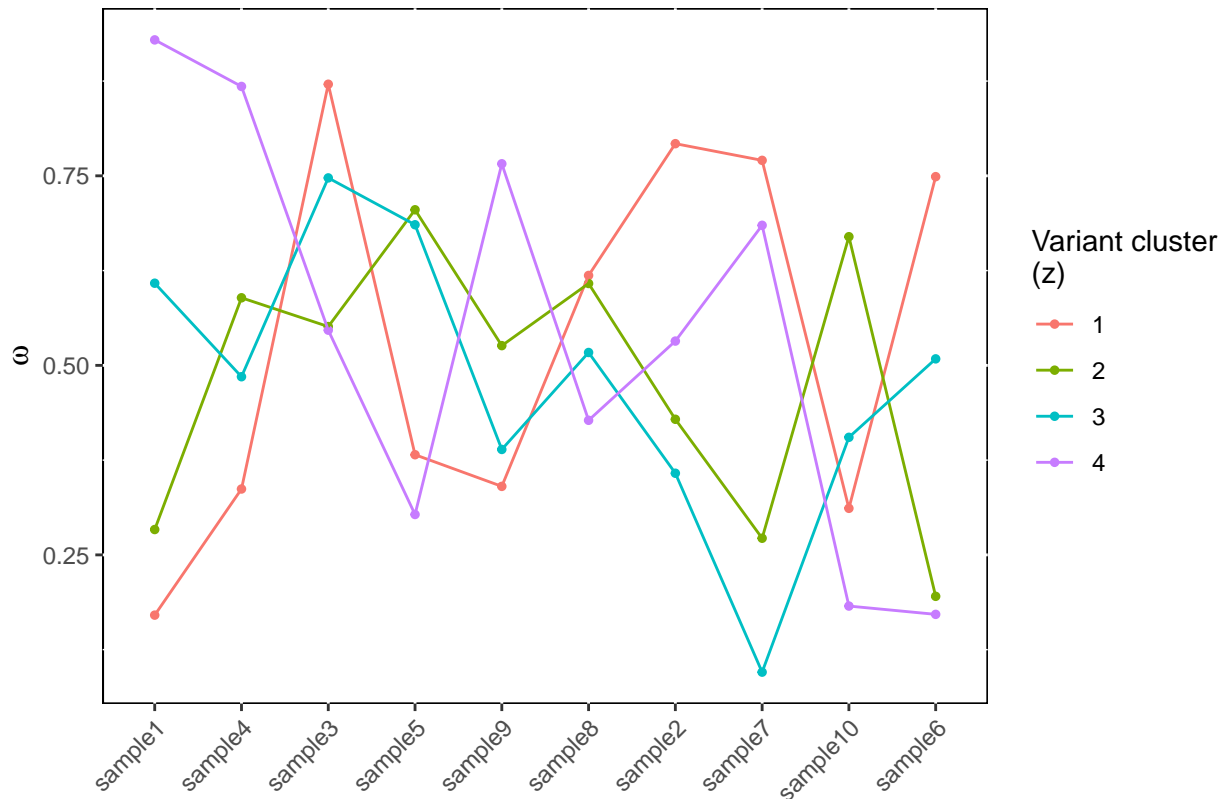
```



```
## by variant
ggplot(p2, aes(sample, fraction, group=variant_index)) +
  geom_point(size=1) +
  geom_line(aes(color=z)) +
  ylab("y/n") +
  xlab("Patient sample\n(arbitrary order)") +
  theme(axis.text.x=element_text(angle=45, hjust=1)) +
  guides(color=guide_legend(title="Variant cluster\n(z)"))
```



```
##
## Cluster means and variances
##
meds  <- apply(W, 2, median)
slevels <- colnames(w)[order(meds, decreasing=TRUE)]
w2 <- w %>%
  as_tibble() %>%
  mutate(z=factor(1:K)) %>%
  ## variant_index=seq_len(nrow(.))) %>%
  gather("sample", "omega", -z) %>%
  mutate(sample=factor(sample, levels=slevels))
##fig1 <- ggplot(w2, aes(z, omega)) +
##   geom_jitter(width=0.1, size=1) +
##   facet_wrap(~sample) +
##   ylab(expression(omega)) +
##   xlab("True cluster") +
##   theme(panel.background=element_rect(fill="white", color="black"))
fig2 <- ggplot(w2, aes(sample, omega, group=z)) +
  geom_point(size=1, aes(color=z)) +
  geom_line(aes(color=z)) +
  ylab(expression(omega)) +
  xlab("") +
  theme(axis.text.x=element_text(angle=45, hjust=1),
        panel.background=element_rect(fill="white", color="black"),
        legend.key=element_rect(fill="white", color="white")) +
  guides(color=guide_legend(title="Variant cluster\n(z)"))
fig2
```



JAGS

```
jags.file <- file.path(models.dir, "w.jags")
inits <- list(".RNG.name" = "base::Wichmann-Hill",
              ".RNG.seed" = 123)
jags.m <- jags.model(jags.file, test.data,
                    n.chains = 1,
                    inits = inits,
                    n.adapt = 1000)

## Compiling model graph
##   Resolving undeclared variables
##   Allocating nodes
## Graph information:
##   Observed stochastic nodes: 500
##   Unobserved stochastic nodes: 591
##   Total graph size: 7808
##
## Initializing model

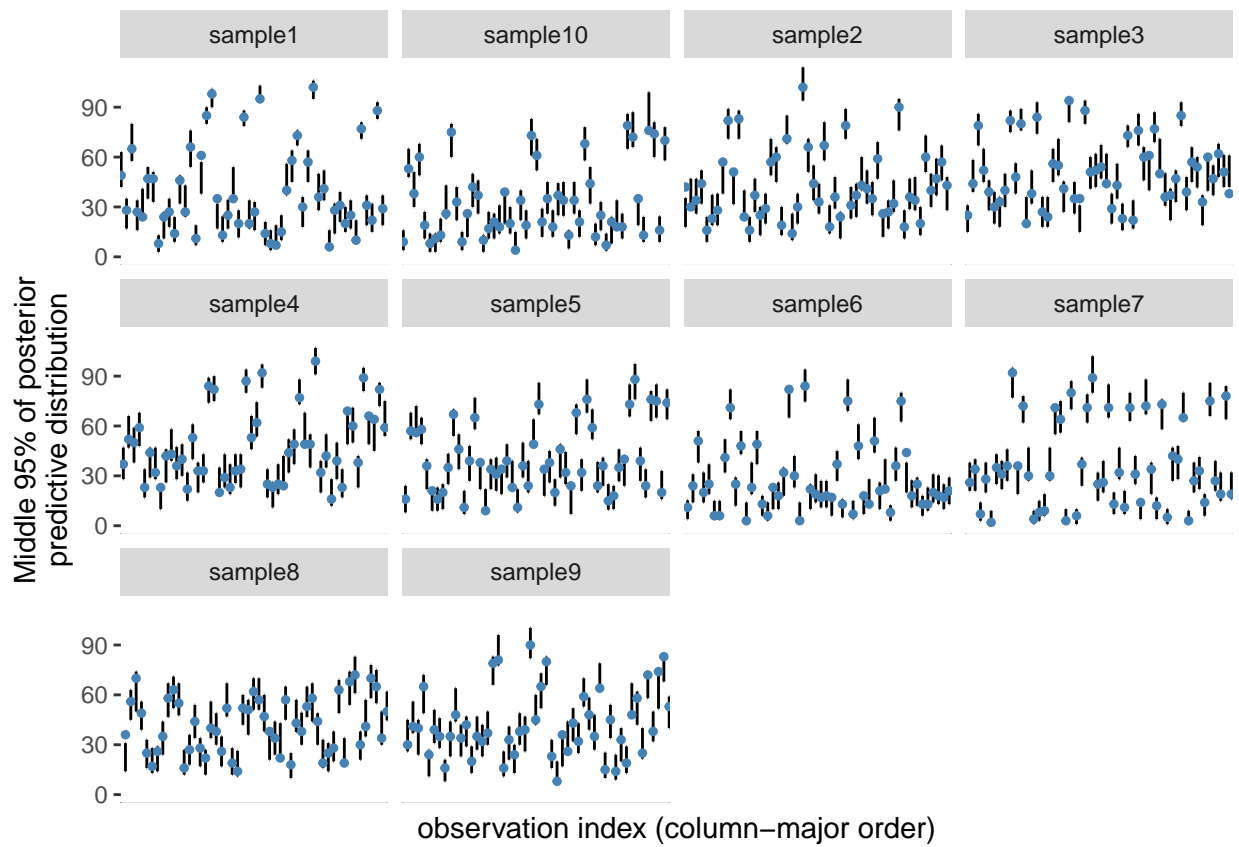
params <- c("z", "w", "ystar")
samps <- coda.samples(jags.m, params, n.iter=10000, thin=7)
jags_df <- ggs(samps)
##ggs_traceplot(jags_df, family="U")
s <- summary(samps)
#effectiveSize(samps)
#pdf(file.path(trace.dir, paste0(runName, "_trace.pdf")))
```

```

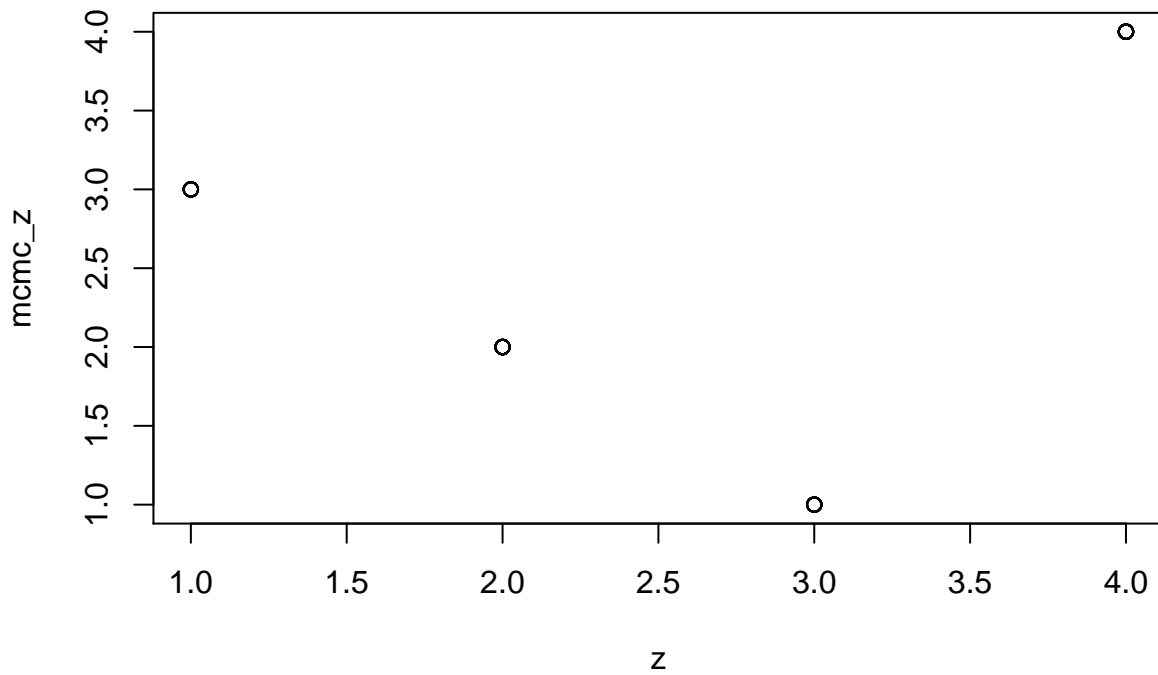
#plot(samps)
#dev.off()

##
## 50 mutations x 10 samples
chains <- do.call(rbind, samps)
ystar <- chains[, grep("ystar", colnames(chains))]
## each row of MCMC is in column-major order
orig.order <- tibble(statistic=colnames(ystar))
ppd.summaries <- ystar %>%
  as_tibble() %>%
  gather("statistic", "value") %>%
  group_by(statistic) %>%
  summarize(mean=mean(value),
            q1=quantile(value, 0.025),
            q3=quantile(value, 0.975))
ppd.summaries2 <- left_join(orig.order,
                          ppd.summaries, by="statistic") %>%
  mutate(observed=as.numeric(test.data$y)) %>%
  mutate(sample=paste0("sample", rep(1:S, each=I)),
         variant=rep(1:I, S))
ggplot(ppd.summaries2, aes(x=statistic, y=mean,
                          ymin=q1,
                          ymax=q3)) +
  geom_errorbar() +
  geom_point(aes(x=statistic, y=observed),
            size=1, color="steelblue") +
  theme(axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        panel.background=element_rect(fill="white",
                                       color="black")) +
  ylab("Middle 95% of posterior\npredictive distribution") +
  xlab("observation index (column-major order)") +
  facet_wrap(~sample)

```



Plots



z	mcmc_z
1	3
2	2
3	1
4	4

