Posterior Predictions with the `cellTypeCompositions` Package

Charles Berry

June 7, 2022

Consider a setup in which there are 2046 integration sites characterized by four celltypes having 20 different compositions and 10 different abundances. The create this setup, first sample 20 compositions according to Dirichlet distribution with parameter 0.5 as a 4 by 20 matrix and draw a sample of 2046 indexes on the columns:

```
> compositions <-
+   prop.table( matrix( rgamma(80,0.5), nrow=4),2)
> compIndex <- sample( 20, 2046, replace=TRUE)
```

Set the possible abundances as 0.5, 1.0, 2.0, . . . , 512 and let 1024 indexes point to 0.5, 512 to 1.0, and so on with 2 indexes pointing to 512.0:

```
> abundances <- 2^seq(-1,9)
> abIndex <- sample( rep(1:10, 2^(10:1)) )
```

For a specimen drawn under this setup, the integration sites have expected counts of cell types given by

```
> Ecells <- t( compositions[, compIndex] ) * abundances[ abIndex ]
```

The sample preparation, cell sorting and extraction of integration sites results in misidentification of cells by type and potential loss of material. The is reflected by a filtration matrix, `omega`, given here as

```
> omega <-
+   matrix(c(
+     0.80, 0.01, 0.00, 0.00,
+     0.01, 0.97, 0.01, 0.00,
+     0.01, 0.01, 0.60, 0.00,
```

```
+     0.00, 0.00, 0.01, 0.98),
+     ncol=4, byrow=TRUE) + 0.0001
>
```

To simulate a draw from the population, the `cellTypeCompositions` library is loaded and the `rtab` function is called with `impute.unseen=FALSE`, since all integrations sites are represented in the setup.

```
> library(cellTypeCompositions)
> pop <-
+   list(
+     eta=compositions,
+     lambda=abundances,
+     dataToEta=compIndex,
+     dataToLambda=abIndex)
> tab <- rtab(list(pop), omega, impute.unseen=FALSE)
```

The table only has 1075 integration sites in it as many with low abundance did not generate a positive count for any cell type, and there are only 191 unique rows. The data are rendered in a compact form using the `uniTab` function:

```
> wtab <- uniTab( tab )
```

Now the simulated data can be fitted using the `gibbsScan` function. One hundred MCMC samples are drawn from the posterior and the last draw is retained.

```
> fit <- gibbsScan(wtab, omega, nburn=100)
```

Using the last draw, the data are simulated with a posterior predictive draw. The default value of `impute.unseen=TRUE` is used, since many integrsation sites are not represented in the `fit` object. The imputation uses the probability of observation to guide sampling the unseen sites.

```
> newtab <- rtab(fit, omega)
> newWtab <- uniTab(newtab)
```

There are 1033 integration sites in this table and there are 198 rows. For comparison, here are the counts for most abundant rows (column `n` is the number of duplicate rows in the data)

```
> with(newWtab, cbind(tab,n)[tail(order( rowSums(tab)) ),]) # new
```

```
                  n
[1,]  5   2 68 25 1
[2,] 41   2  9 56 1
[3,]  8  99  5  0 1
[4,]  4  75 22 25 1
[5,]  7   3 79 51 1
[6,]  7 130 33 17 1
```

```
> with(wtab, cbind(tab,n)[tail(order( rowSums(tab)) ),]) # previous
```

```
                  n
[1,]  7   1 73 31 1
[2,]  3  72 20 18 1
[3,] 53   2 11 53 1
[4,] 20 101  3  0 1
[5,] 13   3 99 55 1
[6,]  8 140 33 25 1
```

and here are the counts for the rows having exactly one cell each:

```
> with(newWtab, cbind(tab,n)[ rowSums(tab) == 1,]) # new
```

```
               n
[1,] 0 0 1 0  90
[2,] 0 0 0 1 150
[3,] 1 0 0 0 132
[4,] 0 1 0 0 164
```

```
> with(wtab, cbind(tab,n)[ rowSums(tab) == 1,]) #  previous
```

```
               n
[1,] 0 1 0 0 160
[2,] 0 0 1 0 102
[3,] 1 0 0 0 145
[4,] 0 0 0 1 137
```

With the posterior predictive sample, further runs can be performed, viz.

```
> fit2 <- gibbsScan(newWtab, omega, nburn=100L)
```