BayClone2 Demo

Juhee Lee and Subhajit Sengupta

November 19, 2014

We provide a demo with simulated data to run the MCMC algorithm as described in the paper titled “Bayesian Inference for Tumor Subclones Accounting for Sequencing and Structural Variants” [Lee et al., 2014]. We have tested `BayClone2` on mac OSX 10.8.4 and 10.9 and CentOs 6.5.
The R package, BayClone2 provides four functions for the end users.

- BayClone2: This function conducts the posterior Markov chain Monte Carlo (MCMC) simulation for BayClone2, a BAYesian feature allocation for tumor subCLONEs (Lee, et al (2014)) .
- fn_post_C: This function computes the posterior marginal distribution of the number of subclones.
- fn_posteior_point: This function returns point estimates of parameters such as a subclonal copy number matrix, a matrix of the number of copies with variant sequence in subclones, a composition weight matrix of samples in subclones and the expected read count with two copies in samples (i.e posterior point estimate for L, Z, W, PHI, PI, P0 for a chosen value of C – for notation, see Lee et al. (2014)).
- export_N_n: This function takes a VCF file as input and generates two output files (names are specified by users) which contains the total number of reads and the number of reads that bear a mutated sequence, respectively, at a particular locus in a specific tissue sample. The two output files can be used as data sets for the function, BayClone2.

We give an example to demonstrate the functionality of the four functions. We first illustrate the functionality and usage of the function, export_N_n with an example:

```
> ### Illustrate the functionality of the function export_N_n
> library("BayClone2")
> export_N_n("/Users/subho/Desktop/R_PACKAGE/test_Data.VCF",
+                 "/Users/subho/Desktop/R_PACKAGE/N_tot.txt",
+                 "/Users/subho/Desktop/R_PACKAGE/n_alt.txt")
```

```
files are generated!
[[1]]
[1] "/Users/subho/Desktop/R_PACKAGE/test_Data.VCF"

[[2]]
[1] "/Users/subho/Desktop/R_PACKAGE/N_tot.txt"

[[3]]
[1] "/Users/subho/Desktop/R_PACKAGE/n_alt.txt"

>
```

The following example reproduces the result of Simulation 1 and generates Figure 4 (a), (d)–(f) in Lee et al (2014). The function, BayClone2 takes 40 minutes on a desktop for 10000 posterior samples after burning 6000 samples (processor: 3.33 GHz 6-Core Intel Xeon, Memory: 32 GB 1333 MHz DDR3). In specific, the R function, proc.time() returns

```
    user   system  elapsed
2016.010  230.163 2247.748
```

he function, fn_posterior_point takes 15 minutes on a desktop for $C = 3$ with MCMC samples returned from the function, BayClone2 (processor: 3.33 GHz 6-Core Intel Xeon, Memory: 32 GB 1333 MHz DDR3). In specific, the R function, proc.time() returns

```
> proc.time()
   user  system elapsed
816.233  59.042 878.636
```

In addition, we tested the function for a larger dataset having $S = 3000$ and $T = 25$. We burn-in 500 samples for each value of C, $2 \leq C \leq 15$ and it takes approximately 10 minutes for each C. For 500 samples that we will use for posterior inference, it takes about 1.2 hours. That is, in total 2.5 hours was taken on the same machine for 500 posterior samples after burning in 500 samples. T

For the demonstration, we use data sets, BayClone2_Simulation1_tot and BayClone2_Simulation1_mut, included in the package. The data sets (**N** and **n**) are simulated and used for simulation study in Lee et al. (2014) where **N** and **n** are $S \times T$ integer matrices. We also specify the number of loci (S) and the number of samples (T). In Simulation 1, the true number of subclones, $C^{TRUE} = 2$, $S = 100$ and $T = 4$ are assumed. Find the details of how the data were simulated from the paper.

```
> ##ILLUSTRATE BayClone2 WITH A SMALL SIMULATION.
> ###REPRODUCE SIMULATION 1 OF LEE ET AL.
> library("BayClone2")
> ##READ IN DATA
> ##TOTAL NUMBER OF READS AT LOCUS s IN SAMPLE t
> N <- as.matrix(read.table("BayClone2_Simulation1_tot.txt",header = TRUE))
> ##NUMBER OF READS WITH VARIANT SEQUENCE AT LOCUS s IN SAMPLE t
> n <- as.matrix(read.table("BayClone2_Simulation1_mut.txt",header = TRUE))
> S <- nrow(N)  # THE NUMBER OF LOCI (I.E. NUMBER OF ROWS OF N (AND n))
> T <- ncol(N) #THE NUMBER OF TISSUE SAMPLES  (I.E. NUMBER OF COLUMNS OF N (AND n))
```

Here we specify the values of the hyperparameters in the model. For the details of the model, see Section 2 of Lee et al. (2014). Users may need to customize the hyperparameter values for their dataset.

```
> ###################################
> #HYPER-PARAMETER  ----SPECIFYING HYPERPARAMETER VALUES
> ######################################
> #HYPER-PARAMETER
> hyper <- NULL
> #NUMBER OF SUBCLONES (GEOMETRIC DIST)
> ### C ~ GEOMETRIC(r) WHERE E(C)=1/r
> hyper$r <- 0.2
> #PRIOR FOR L
> hyper$Q <- 3  #NUMBER OF COPIES -- q = 0, 1, 2, 3
> ##BETA-DIRICHLET
> ###PI_C | C ~ BETA-DIRICHLET (ALPHA/C, BETA, GAMMA)
> hyper$alpha <- 2
> hyper$beta <- 1
> hyper$gam <- c(0.5, 0.5, 0.5)
> #PRIOR FOR PHI--TOTAL NUMBER OF READS IN SAMPLE T
> ###PHI_T ~ GAMMA(A, B)
> hyper$b <- 3
> hyper$a <- median(N)*hyper$b
> #PRIOR FOR P_0
> ###P0 ~ BETA(a, b)
> hyper$a_z0 <- 0.3
> hyper$b_z0 <- 5
> #PRIOR FOR W
> ##W_T | L ~ DIRICHLET(D0, D, ..., D) WHERE W_T=(w_t0, w_t1, ..., w_tC)
> hyper$d0 <- 0.5
> hyper$d <- 1
```

We specify MCMC parameters. Here “burn.in” and “n.sam” are the number of burn-in samples and the number of samples for posterior inference, respectively. In particular, we burn in 6000 samples for each value of the number of subclones (C) using the test data and then we sample 6000 samples of the parameters which will be used to posterior inference.

```
> #WE USE THE MCMC SIMULATION STRATEGY PROPOSED IN LEE AT EL (2014)
> n.sam <- 10000;  ##NUMBER OF SAMPLES THAT WILL BE USED FOR INFERENCE
> ##NUMBER OF SAMPLES FOR BURN-IN
> #(USE THIS FOR A TRAINING DATA---FOR DETAILS, SEE THE REFERENCE)
> burn.in <- 6000
```

For computational efficiency, we set the maximum and minimum numbers of the subclones including the background subclones (max_C and min_C). Since we burn in 6000 samples for each value of C, the posterior computation may take more time with a large value of max_C. Particularly, we set $max_C = 16$ for this example. This value is large compared to the true value of this simulation, $C^{TRUE} = 2$ and the posterior inference with $max_C = 16$ may not be very different from that with no upper limit of C.

```
> ##############################################
> ###WE CONSIDER C BETWEEN 1 AND 15 IN ADDITION TO BACKGROUND SUBCLONE
> ####Max_C AND Min_C SPECIFIES VALUES OF C FOR POSTERIOR EXPLORATION
> Min_C <- 2  ##INCLUDING THE BACKGROUND SUBCLONE
> Max_C <- 16  ##INCLUDING THE BACKGROUND SUBCLONE
```

The function, BayClone2, simulates posterior samples via posterior MCMC simulation. We pass data sets ($\mathbf{N}$ and $\mathbf{n}$), hyperparemeter values (*hyper*), MCMC parameters (*burn.in* and *n.sam*), min_C and max_C as arguments to the function, BayClone2. In addition, we pass one more argument (0.25 in the code below). The argument is the mean proportion for a training dataset. It is used to split the data into two parts, a training dataset and a test dataset for the trans-dimensional MCMC. You may use a value different from 0.25 (we recommend a small value for this like 0.025). From our simulation studies, 0.025 worked well.

```
> ################################################################
> ##DO MCMC SAMPLING FROM BAYCLONE2!
> ################################################################
> ##THE LAST ARGUMENT (0.025) IS THE MEAN PROPORTION FOR THE TRAINING DATASET
> ## (IT IS SPECIFIED BY USERS)
> ##IT WILL BE USED TO SPLIT TRAINING AND TEST DATASETS
> ##FOR DETAILS, SEE THE REFERENCE LEE AT EL (2014)
> ##TO RUN, COMMENT IN THE LINE BELOW (WARNING! THIS MAY TAKE APPROXIMATELY 30 MINUTES)
> set.seed(11615)
> MCMC.sam <- BayClone2(Min_C, Max_C, S, T, burn.in, n.sam, N, n, hyper, 0.025)
```

```
[1] "Doing MCMC sampling for training dataset"
[1] "Wed Nov 19 22:23:29 2014"
[1] "working on i.c= 2"
[1] "Wed Nov 19 22:23:43 2014"
[1] "working on i.c= 3"
[1] "Wed Nov 19 22:24:07 2014"
[1] "working on i.c= 4"
[1] "Wed Nov 19 22:24:43 2014"
[1] "working on i.c= 5"
```

```
[1] "Wed Nov 19 22:25:26 2014"
[1] "working on i.c= 6"
[1] "Wed Nov 19 22:26:20 2014"
[1] "working on i.c= 7"
[1] "Wed Nov 19 22:27:28 2014"
[1] "working on i.c= 8"
[1] "Wed Nov 19 22:28:44 2014"
[1] "working on i.c= 9"
[1] "Wed Nov 19 22:30:01 2014"
[1] "working on i.c= 10"
[1] "Wed Nov 19 22:31:27 2014"
[1] "working on i.c= 11"
[1] "Wed Nov 19 22:33:16 2014"
[1] "working on i.c= 12"
[1] "Wed Nov 19 22:35:00 2014"
[1] "working on i.c= 13"
[1] "Wed Nov 19 22:36:53 2014"
[1] "working on i.c= 14"
[1] "Wed Nov 19 22:39:33 2014"
[1] "working on i.c= 15"
[1] "Wed Nov 19 22:43:17 2014"
[1] "working on i.c= 16"
[1] "Doing MCMC sampling using training and test datasets"
[1] "Wed Nov 19 22:47:13 2014"
[1] "50 % MCMC sampling has been done."
[1] "Wed Nov 19 22:53:58 2014"
[1] "100 % MCMC sampling has been done."
[1] "Wed Nov 19 23:01:41 2014"
```

The example codes below demonstrate how to summarize the joint posterior distribution. We first find the marginal posterior distribution of C and find posterior point estimates of the other parameters conditional on a chosen value of C. In the code below, $CC = 3$ is set. This includes the background subclone, implying that we choose $C = 2$, *not* including the background subclone that is the maximum a posteriori (MAP) estimate (see Figure 1). The value of CC should not exceed 10 due to the permutation of subclones in the evaluation (see Section 2.3 of Lee et al. (2014)).

```
> ####################################################################
> #COMPUTE THE POSTERIOR MARGINAL DIST OF C (THE NUMBER OF SUBCLONES)
> ####################################################################
> ##TO RUN, COMMENT IN THE LINE BELOW
> post_dist_C <- fn_post_C(MCMC.sam$C, Min_C, Max_C)
> ###########################################################################################
> ####WE FIND POSTERIOR POINT ESTIMATES OF L, Z, W, PHI, PI, P0 FOR A CHOSEN VALUE OF C
> ###########################################################################################
> ##THE FIRST ARGUMENT (3) IS A VALUE OF C CHOSEN BY USERS
> #C IS THE NUMBER OF SUBCLONES INCLUDING THE BACKGROUND SUBCLONE)
> ##THE CHOSE VALUE OF C SHOULD BE LESS THAN OR EQUAL TO 10
> ##(INCLUDING THE BACKGROUND SUBCLONE)
> #DUE TO THE PERMUTATION (FOR DETAILS, SEE THE REFERENCE LEE AT EL (2014))
> ##TO RUN, COMMENT IN THE LINE BELOW (WARNING! THIS MAY TAKE APPROXIMATELY 15 MINUTES)
> CC <- 3
> point.est <- fn_posterior_point(CC, S, T, MCMC.sam)

[1] "SAMPLE SIZE FOR THE GIVEN C = 6483"
```

We now reproduce some of the figures in Lee at el. (2014). Figure 1 (p. 7) is the marginal posterior distribution of the number of subclones (C), *not* including the background subclone. It is the same as Figure 4-(a) in Lee et al.(2014). It can be produced by the following code:

```
> ####SUMMARY FOR FIGURES######################
> ##PLOT THE MARGINAL DISTRIBUTION OF C
> par(mar=c(4.5, 4.5, 2.1, 2.1))
> plot((0:(Max_C-1)), post_dist_C[,2], type="o", lwd=4, col=1,
+ xlab="C", ylab="P(C|n, N)", main="", cex.axis=1.5, cex.lab=1.5)
> abline(v=2, lty=2, lwd=3, col=2)  #ASSUMED TRUE C=2
```

Figure 2 (p. 8) is the posterior point estimate of $\mathbf{L}$, a $(S \times C)$ matrix of subclonal copy numbers for a chosen value of C (here, MAP, $C = 2$). The figure is the same as Figure Figure 4-(d) in Lee et al (2014). It can be produced by the following code:

```
> #####################################################
> library(gplots);
> lmat = rbind(c(0,3),c(2,1),c(0,4))
> lwid = c(1.5,4)
> lhei = c(1.5,4,1.2)
> post.L <- point.est$L
> colnames(post.L) <- (0:(CC-1))
> heatmap.2(post.L[,-1], trace="none", Colv=FALSE, Rowv=FALSE,
+ dendrogram="none", scale="none", col=redgreen, colsep=(1:(CC-1)),
+ sepcol=c("white", "white"), sepwidth=c(0.05, 0.1),
+ main="",lmat = lmat, lwid = lwid, lhei = lhei)
>
```

Figure 3 (p. 9) is the posterior point estimate of $\mathbf{Z}$, a $(S \times C)$ matrix of numbers of copies with variant sequence in subclones for chosen C and $\mathbf{L}$ (here, MAP, $C = 2$ and the chosen $\mathbf{L}$ is $\mathbf{L}$ in Figure 2). The figure is the same as Figure Figure 4-(e) in Lee et al (2014). It can be produced by the following code:

```
> #####################################################
> post.Z <- point.est$Z
```

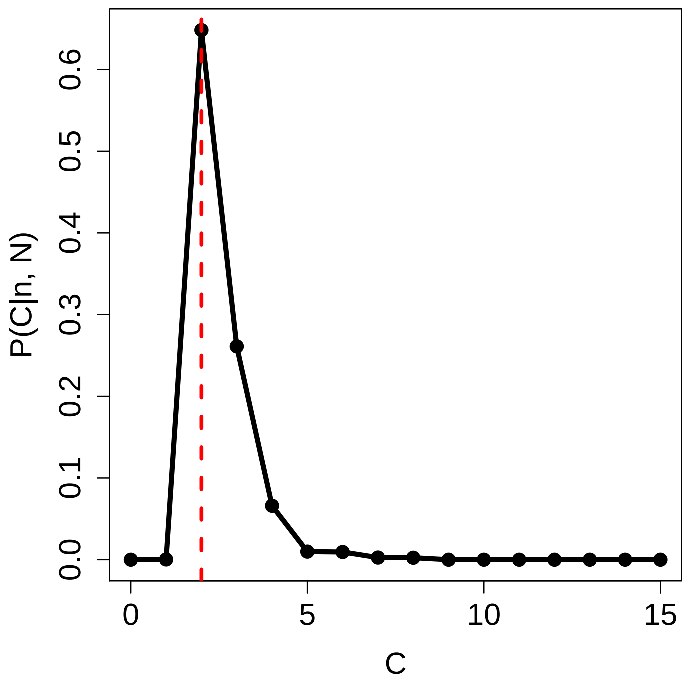

Figure 1: Posterior marginal distribution of C (the same as Figure 4-(a) in Lee et al (2014))

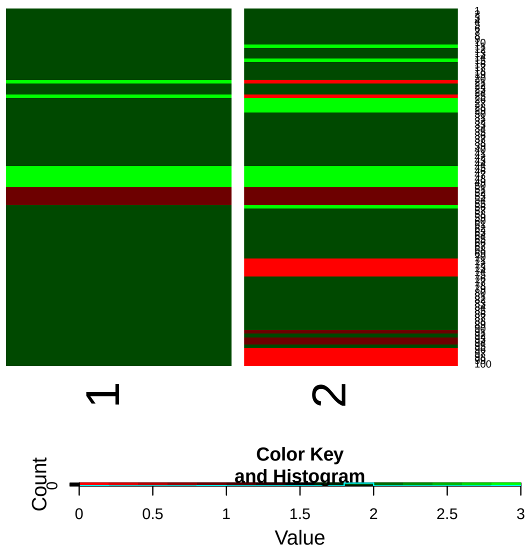

Figure 2: Heatmap of the posterior point estimate of $\mathbf{L}$ conditional on $C = 2$ (Figure 4-(d) in Lee et al (2014))

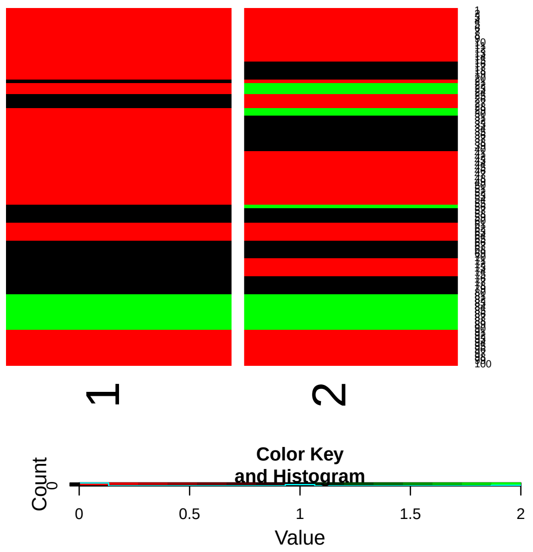

Figure 3: Heatmap of the posterior point estimate of $\mathbf{Z}$ conditional on $C = 2$ (Figure 4-(e) in Lee et al (2014))

```
> colnames(post.Z) <- (0:(CC-1))
> heatmap.2(post.Z[,-1], trace="none", Colv=FALSE, Rowv=FALSE,
+ dendrogram="none", scale="none", col=redgreen, colsep=(1:(CC-1)),
+ sepcol=c("white", "white"), sepwidth=c(0.05, 0.1),
+ main="",lmat = lmat, lwid = lwid, lhei = lhei)
>
```

Figure 4 (p. 10) is the posterior point estimate of $\mathbf{w}$, a $(T \times C)$ matrix of composition weights of samples over subclones for chosen values of C, $\mathbf{L}$ and $\mathbf{Z}$ (the chosen $\mathbf{L}$ and $\mathbf{Z}$ are $\mathbf{L}$ and $\mathbf{Z}$ in Figures 2 and 3). The figure is similar to Figure 4-(f) in Lee et al (2014). The slight difference is from difference in color scaling (not from the difference in the actual estimates). It can be produced by the following code:

```
> ####################################################
> post.w <- point.est$w
> colnames(post.w) <- (0:(CC-1))
> heatmap.2(post.w, trace="none", Colv=FALSE, Rowv=FALSE,
```

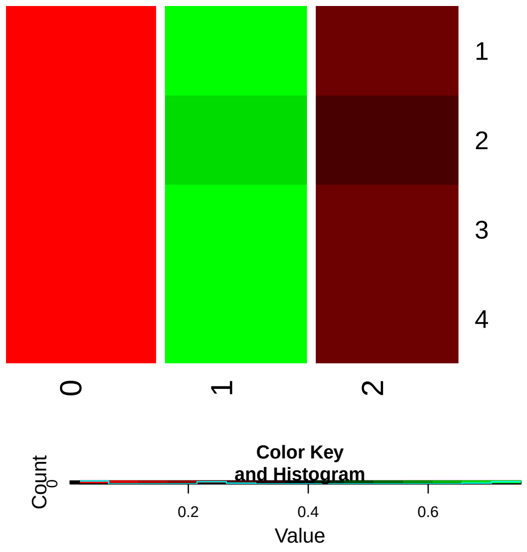

Figure 4: Heatmap of the posterior point estimate of **w** conditional on $C = 2$ (Similar to Figure 4-(f) in Lee et al (2014))

```
+ dendrogram="none", scale="none", col=redgreen, colsep=(1:(CC-1)),
+ sepcol=c("white", "white"), sepwidth=c(0.05, 0.1),
+ main="",lmat = lmat, lwid = lwid, lhei = lhei)
>
```

If you have any further questions, please contact juheelee@soe.ucsc.edu or subhajit06@gmail.com.