 # Author: Amin Haghani, PhD
 # Date: 02.11.2021

```
# The alignment pipeline require the genome fasta file and annotation GFF or GTF files.
# We aligned the probe source sequences (available in mammalian array manifest file) to different mammalian genomes.
# Here is an example pipeline.

if(!require(easypackages)){install.packages(easypackages)}
library(easypackages)
libraries("tidyr", "dplyr","BiocManager","parallel", "QuasR", "Rsamtools", "ChIPseeker")
alignment <- qAlign("manifest.txt", genome = "Genome FASTA path",
                    bisulfite = "undir", alignmentParameter = "-k 2 --strata --best -v 3")

# the result will be saved in bam format
aln <- BamFile(filepath)
aln <- scanBam(aln)
aln <- as.data.frame(aln[[1]])

# Determination of CG location based on the probe design. The probe is designed by either top or bottom strand.

aln <- manifest %>% dplyr::select(IlmnID, SourceSeq, targetCG) %>% dplyr::rename(qname = IlmnID)
%>% right_join(aln, by="qname")%>%
  mutate(targetCG = as.character(targetCG))

CGcount <- rbindlist(lapply(1:nrow(aln), function(i){
  pattern <- DNAString(as.character(aln$SourceSeq[i]))
  subject <- DNAString(aln$seq[i])
  matches <- matchPattern(pattern, subject, max.mismatch = 0, algorithm = "naive-inexact")
  locations = paste(start(matches), end(matches), sep=":")
  pattern2 <-reverseComplement(DNAString(as.character(aln$SourceSeq[i])))
  matches2 <- matchPattern(pattern2, subject, max.mismatch = 0, algorithm = "naive-inexact")
  locations2 = paste(start(matches2), end(matches2), sep=":")
  hits <- data.frame(qname=aln$qname[i],
                     CGcount = length(start(matches))+length(start(matches2)),
                     forward = paste(locations, collapse = " ; "),
                     reverse = paste(locations2, collapse = " ; "))
}))


aln$alignedStand <- ifelse(CGcount$forward!="", "forward", "complementReverse")
aln$targetCG <- ifelse(aln$alignedStand=="forward", aln$targetCG,
                       ifelse(aln$alignedStand=="complementReverse"&aln$targetCG=="1:2", "49:50",
                              ifelse(aln$alignedStand=="complementReverse"&aln$targetCG=="49:50",
"1:2",NA)))
aln$targetCG <- as.numeric(as.character(factor(aln$targetCG, levels = c("1:2", "49:50"), labels =
c(0,48))))
aln <- aln %>% filter(!is.na(pos))
```

```
# convert to GRange for annotation
input <- aln %>% dplyr::select(qname, rname, strand, pos) %>% dplyr::filter(complete.cases(.)) %>%
  mutate(start = pos) %>% mutate(end = pos+49)
input <- input[,c(2,5,6,1, 3)]
names(input) <- c("chr","start", "end", "CGid", "strand")
target <- with(input,
               GRanges( seqnames = Rle(chr),
                        ranges   = IRanges(start, end=end, names=CGid),
                        strand   = Rle(strand(strand)) ))

# create TxDB
txdb <- makeTxDbFromGFF("gff file path", format = "gff3")


# annotating the probes and estimating the CG location
peakAnno <- annotatePeak(target, tssRegion=c(-10000, 1000),
                         TxDb=txDb,
                         sameStrand = FALSE, overlap = "all", addFlankGeneInfo=T)
genomeAnnotation <- data.frame(CGid = peakAnno@anno@ranges@NAMES, peakAnno@anno,
                           peakAnno@detailGenomicAnnotation)
genomeAnnotation <- genomeAnnotation %>% dplyr::rename(probeStart = start, probeEnd = end)
genomeAnnotation <- aln %>% dplyr::select(qname, targetCG, seq) %>%
  dplyr::rename(CGid = qname) %>%
  right_join(peakAnnotation, by="CGid") %>%
  mutate(CGstart = probeStart+targetCG, CGend =probeStart+targetCG+1) %>%
  relocate(... = c(CGstart, CGend, seq), .after = strand) %>% dplyr::select(-targetCG)


# Confirming if the CG is real. This step is done by extracting the sequence from the original FASTA file
BEDfile <- genomeAnnotation %>% dplyr::select(seqnames, CGstart, CGend,
                                  CGid, strand) %>%
  setnames(new = c("chrom", 'chromStart', 'chromEnd', 'name', "strand")) %>%
  filter(!is.na(chromStart)) %>% mutate(chromStart = chromStart-1)
write.table(BEDfile, "BEDfile.bed",
            sep = "\t", row.names=F, col.names=F, quote = F)

#bedtools getfasta -fi [FASTA file, usually .fa or .fna]
#    -bed [path to bed file that was just created]
#     -fo [output e.g. BEDfile.fasta]

CGs <- readDNAStringSet("BEDfile.fasta")
seq_name = names(CGs)
sequence = paste(CGs)
df <- data.frame(seq_name, sequence) %>% dplyr::rename(CG = sequence) %>%
  mutate(CG = ifelse(CG %in% c("CG", "GC"), TRUE, FALSE))
genomeAnnotation <- genomeAnnotation %>% mutate(seq_name = paste(seqnames,":", CGstart-1,"-",
CGend, sep = "")) %>%
  left_join(df) %>% dplyr::select(-seq_name) %>% filter(CG==TRUE)
```