```
#!/usr/bin/python
from __future__ import division
import sys, os, math, random, re, time
#from Bio import Seq as s

def get_params(d1,sample):
    data = open(d1,'r')
    snippets = []

    #snippets contains a small number of sequences from the file

    for i in range(0,sample):
        thisline = data.readline()
        if re.match('[GNATC]{25,45}',thisline):
            snippets.append(thisline)
    data.close()


    # now we fit a model to snippets: n bases of identity, with
probability 1-e, and N-n bases of .25 per base
    # has the barcode been removed already?
    err = 0
    for item in snippets[0:100]:
        for item2 in snippets[100:200]:
            for chr in range(0,3):
                if item[chr] != item2[chr]: err += 1
    if err < 750:
        N = 13
        n_saved = 1
        log_L_saved = 0
        log_L = 0

        maxlen = max([len(frag) for frag in snippets])

        basecount = []
        for char in range(0,maxlen):
            basecount.append([0,0,0,0,0])



        for frag in snippets:
            for char in range(0,len(frag)):
                #print char,len(frag)
                #print basecount[char]
                if frag[char].lower() == 'a':
                    basecount[char][0] += 1
                elif frag[char].lower() == 'c':
                    basecount[char][1] += 1
                elif frag[char].lower() == 'g':
                    basecount[char][2] += 1
```

```
                elif frag[char].lower() == 't':
                    basecount[char][3] += 1
                elif frag[char].lower() == 'n':
                    basecount[char][4] += 1
        maxima = []
        for entry in range(0,len(basecount)):
            maxloc = basecount[entry].index(max(basecount[entry]))
            maxima.append(maxloc)
        #print maxima

        # count times that base != most common base
        diffs = 0

        for n in range(1,N):
            for frag in snippets:
                # score each fragment
                for base in range(0,n):
                    '''if frag[base].lower() == 'a': ind = 0
                    elif frag[base].lower() == 'c': ind = 1
                    elif frag[base].lower() == 'g': ind = 2
                    elif frag[base].lower == 't': ind = 3'''
                    ind = 'acgtn'.index(frag[base].lower())

                    try:
                        if ind == maxima[base]:
                            pass
                           #log_p = math.log( basecount[base][ind]/
sum(basecount[base]) ) # sum should be 10K - 1, P(correct)
                        else:
                            diffs += 1
                            #log_p = math.log( 1 - basecount[base]
[ind]/sum(basecount[base]) ) #P(incorrect)

                    except ValueError:
                        print ind
                        print base
                        print basecount[base]
                        print basecount[base][ind]/
sum(basecount[base])
                        print ' '

            p_diff = (diffs)/n/len(snippets)

            if diffs == 0: logcont1 = 0
            else: logcont1 = math.log(p_diff/3)

            if diffs == n*len(snippets): logcont2 = 0
            else: logcont2 = math.log(1-p_diff)

            log_L = logcont1*diffs + logcont2*(n*len(snippets)-diffs)
```

```
+ (N-n)*math.log(0.25)*len(snippets)

            #        log_L += log_p
            # correct for smaller n

        #    log_L += len(snippets)*(N-n)*math.log(0.25)

        #    BIC = -2 * log_L - n*math.log(sum(basecount[0])*N)

        #    print ''.join(('For n = ',str(n),', ln L = ',str(log_L)))

            if n != 1: # always accept if n=1
                if log_L > log_L_saved:
                    n_saved = n
                    log_L_saved = log_L
            elif n == 1:
                log_L_saved = log_L

            log_L = 0 # prep for next iteration
            diffs = 0
        return [n_saved,log_L_saved,snippets ]
    else:
        return [0, 0, snippets]
#### tidy up
if __name__ == '__main__':
    [ n_saved,log_L_saved,snippets ] = get_params(sys.argv[1],40000)
    sys.stderr.write( ''.join(('Best fit is n = ',str(n_saved),', with
ln L = ',str(log_L_saved))))
    for ii in range(0,50):
        index = random.randint(0,len(snippets)-1)
        sys.stderr.write('\n')
        sys.stderr.write(snippets[ii][0:n_saved+3])
    sys.stderr.write('\n')
    data1 = open(sys.argv[1],'r')

    counter = 0
    c2 = 0

    if len(sys.argv) >= 2:
        prop = sys.argv[2]
    else:
        prop = 1.5

    while True:
        # inner loop is just to write out every million lines
        for i in range(0,4000000):
            counter += 1
            line = data1.readline()
            if counter % 4 == 1:
                use = random.uniform(0,1) < prop
```

```
    if line:
        if ((counter % 4 == 2) or (counter % 4 == 0)) and use:
            sys.stdout.write(line[n_saved:])
        elif use:
            sys.stdout.write(line)
    else:
        data1.close()
        sys.stderr.write('\n')
        sys.exit()
c2 += 1
sys.stderr.write('\n'+str(c2)+' million reads processed')
```