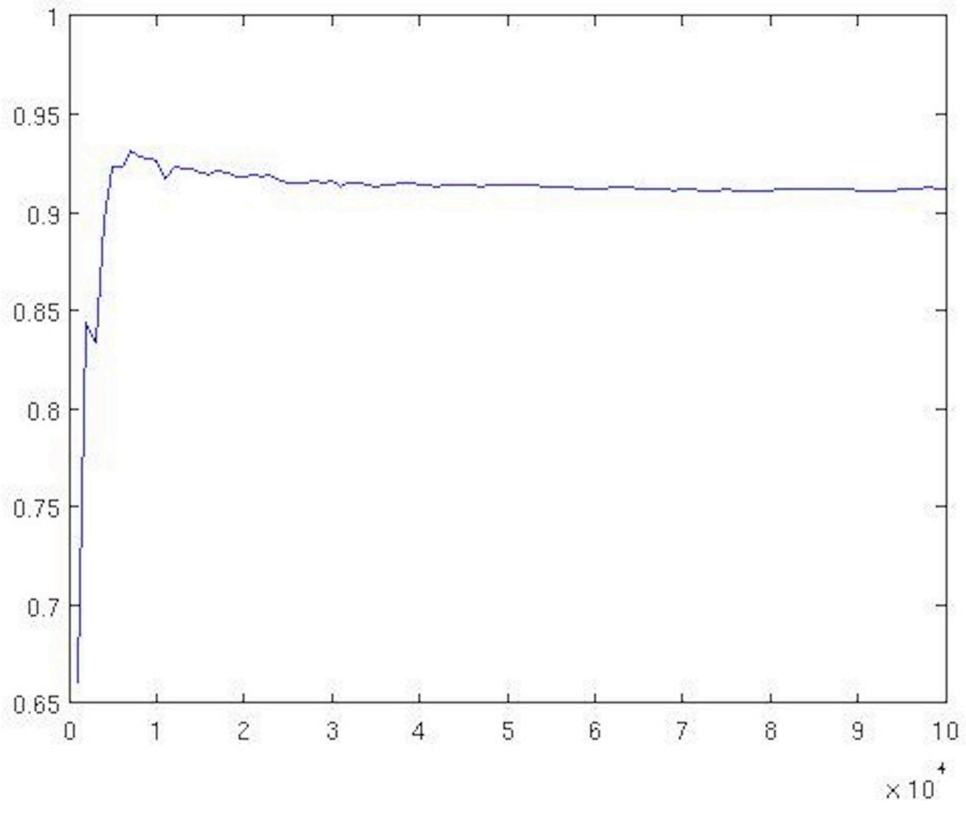
```
num_samples = 100;
noise_level = 0.1;
constant = (1-noise_level)/(1+noise_level);
%Q = B7 and E = Z
%P(Q=q/E=e) = sum[I(q,q_i)*P(E/pa(E))]/sum[P(E/pa(E))]
%P(B7=1/Z=64) = sum[I(b7_i, 1)*P(Z=64/pa(Z))]/sum[P(Z=64/pa(Z))] = num/den
num = 0;
den = 0;
estimate = zeros(1,100);
samples = zeros(1,100);
for num_samples = 1000:1000:100000
    for i=1:num_samples
        b1 = randint; b2 = randint; b3 = randint; b4 = randint; b5 = randint; b6 = randint;
        b7 = randint; b8 = randint; b9 = randint; b10 = randint;
        dec = b1+2*b2+4*b3+8*b4+16*b5+32*b6+64*b7+128*b8+256*b9+512*b10;
        pre = constant*power(noise_level, abs(64-dec)); %P(Z=64/pa(Z)_i)
        if(b7==1)
            num = num + pre;
        end
            den = den + pre;
    end
    estimate(num_samples/1000) = num/den;
    samples(num_samples/1000) = num_samples;
end
```


1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0
1
2
3
4
5
6
7
8
9
10
x 10^4

```
import math
f1 = open('vocab.txt','r');
f2 = open('unigram.txt', 'r');
f3 = open('bigram.txt', 'r');

vocab = f1.readlines();
unigram = f2.readlines();
bigram = f3.readlines();

total_count = 0;
for each in unigram:
     total_count = total_count + int(each)

###########parta##########
print '--PARTA--'
prob_word = []
for i in range(len(unigram)):
     prob_i = float(unigram[i])/total_count
     prob_word.append(prob_i)
     if(vocab[i].startswith('A')):
          print "%-15s  %0.6f" % (vocab[i].strip(),
prob_i)
print '\n'
###########partb##########
print '--PARTB--'
index_the = 0
for i in range(len(unigram)):
     if(vocab[i].strip() == 'THE'):
          index_the = i
          break
i = 0
while(1):
     b = bigram[i].split()
     if(b[0].strip()==str(index_the+1)):
          break
     i = i + 1
the_start = i
the_count = int(unigram[index_the]);
prob_bi = []
while(1):
     b = bigram[i].split()
     if not (b[0].strip()==str(index_the+1)):
          break
     i = i + 1
     prob_bi.append(float(b[2])/the_count)
temp = prob_bi[:]
temp.sort(reverse=True)
temp = temp[0:10]
for each in temp:
     ind = prob_bi.index(each)
     position = bigram[the_start+ind].split()
     word = vocab[int(position[1].strip())-1]
     print "%-15s  %0.6f" % (word.strip(), each)
print('\n')
###########partc##########
print '--PARTCandD--'
def uniprob(word):
     for i in range(len(vocab)):
          if(vocab[i].strip()==word.upper()):
               break
          i = i + 1
     return float(unigram[i])/total_count


def biprob(word2, word1):
     index_word1 = -1
     index_word2 = -1
     for i in range(len(unigram)):
          if(vocab[i].strip().upper() == word1.upper()):
               index_word1 = i
               break
     if(index_word1==-1):
          return 0
     for i in range(len(unigram)):
          if(vocab[i].strip().upper() == word2.upper()):
               index_word2 = i
               break
     if(index_word2==-1):
          return 0
     i = 0
     while(1):
          b = bigram[i].split()
          if(b[0].strip()==str(index_word1+1)):
               break
          i = i + 1
     word1_start = i
     word1_count = int(unigram[index_word1]);
```

```
     while(i<len(bigram)):
          b = bigram[i].split()
          if not (b[0].strip()==str(index_word1+1)):
               return 0
          if b[1].strip()==str(index_word2+1):
               return float(b[2])/
float(unigram[index_word1])
          i = i + 1
     if(i==len(bigram)):
          return 0
def unil(sentence):
     u = 1
     words = sentence.split()
     for each in words:
          u = u*uniprob(each)
     return u

def bil(sentence):
     b = biprob('the','<s>')
     words = sentence.split()
     for i in range(len(words)-1):
          b = b*biprob(words[i+1], words[i])
     return b

def loglikelihood(sent):
     print sent
     if(unil(sent)>0):
          print 'unigram probability : ' +
str(math.log(unil(sent)))
     else:
          print 'unigram probability is not defined'
     if(bil(sent)>0):
          print 'bigram probability : ' +
str(math.log(bil(sent)))
     else:
          print 'bigram probability is not defined'

loglikelihood('The stock market fell by one hundred points
last week')
print '\n'
loglikelihood('The sixteen officials sold fire insurance')
print '\n'

##########parte##########
print '--PARTE--'
def probm(word2, word1, lam):
     return lam*uniprob(word2) + (1-lam)*biprob(word2,
word1)

l = 0.01
maxl = 0.01
maxp = -100
while(l<1.01):
     s = 'The sixteen officials sold fire
insurance'.split()
     mix = probm('the','<s>', l)
     for i in range(len(s)-1):
          mix = mix*probm(s[i+1], s[i], l)
     if(math.log(mix)>maxp):
          maxp = math.log(mix)
          maxl = l
     #print str(math.log(mix))+','
     l = l+0.01
print maxl
```

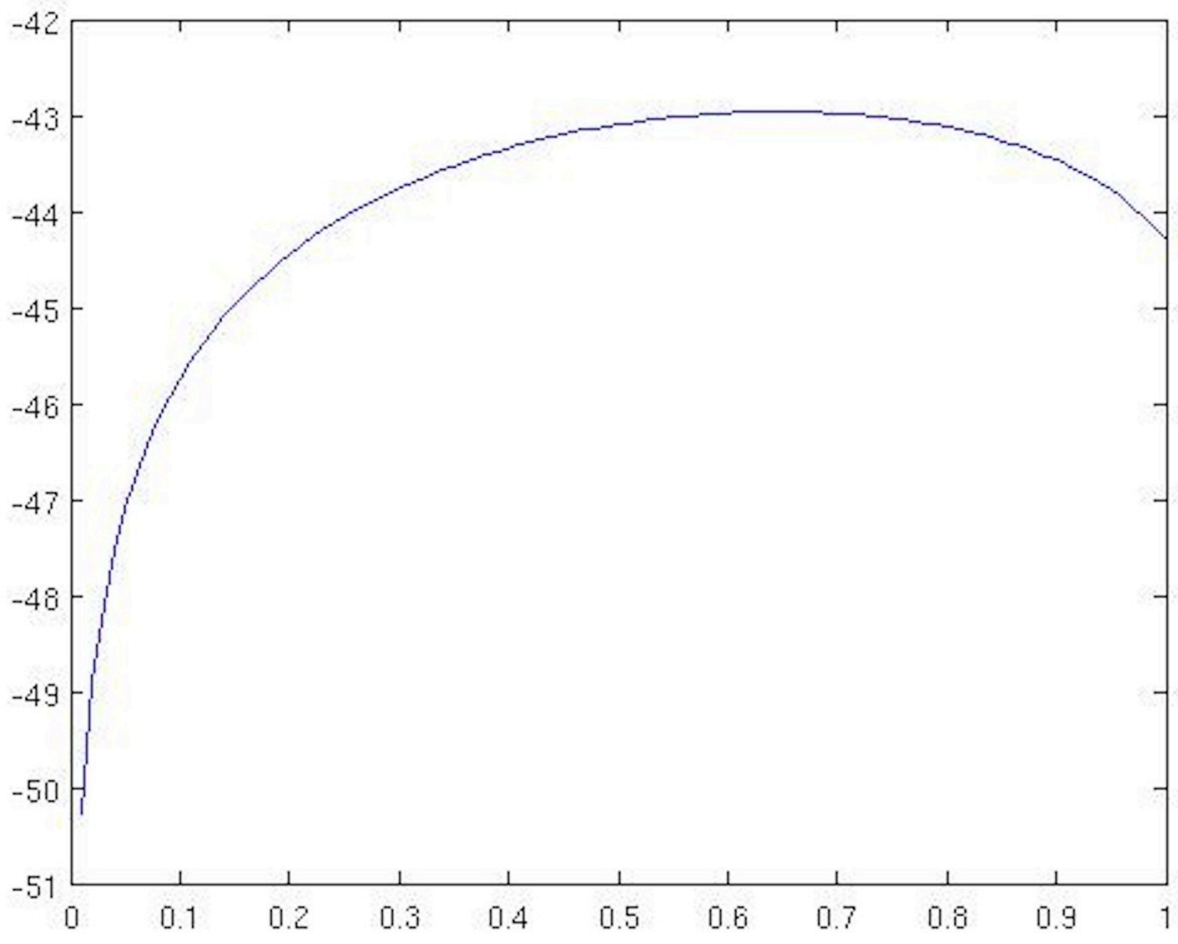
```
--PARTA--
A                0.018407
AND              0.017863
AT               0.004313
AS               0.003992
AN               0.002999
ARE              0.002990
ABOUT            0.001926
AFTER            0.001347
ALSO             0.001310
ALL              0.001182
A.               0.001026
ANY              0.000632
AMERICAN         0.000612
AGAINST          0.000596
ANOTHER          0.000428
AMONG            0.000374
AGO              0.000357
ACCORDING        0.000348
AIR              0.000311
ADMINISTRATION   0.000292
AGENCY           0.000280
AROUND           0.000277
AGREEMENT        0.000263
AVERAGE          0.000259
ASKED            0.000258
ALREADY          0.000249
AREA             0.000231
ANALYSTS         0.000226
ANNOUNCED        0.000227
ADDED            0.000221
ALTHOUGH         0.000214
AGREED           0.000212
APRIL            0.000207
AWAY             0.000202


--PARTB--
<UNK>            0.615020
U.               0.013372
FIRST            0.011720
COMPANY          0.011659
NEW              0.009451
UNITED           0.008672
GOVERNMENT       0.006803
NINETEEN         0.006651
SAME             0.006287
TWO              0.006161


--PARTCandD--
The stock market fell by one hundred points last week
unigram probability : -64.5094403436
bigram probability : -40.9181321338


The sixteen officials sold fire insurance
unigram probability : -44.2919344731
bigram probability is not defined


--PARTE--
0.65
```


-42
-43
-44
-45
-46
-47
-48
-49
-50
-51
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1