1

a)

```
$ carmel --train-cascade -HJ TRAIN1 source.wfsa channel.wfst
```

Without setting the convergence ratio or using random restart, the algorithm quickly converges in 3 iterations and get a per-example perplexity of 2^293.554.

The resulting `source.wfsa.trained` is:

```
3
(0 (1 *e* c1 0.5))
(0 (2 *e* c2 0.5))
(1 (1 *e* c1 0.492734031767489))
(1 (2 *e* c2 0.492734031767489))
(1 (3 *e* *e* 0.0145319364650233))
(2 (1 *e* c1 0.492734031767489))
(2 (2 *e* c2 0.492734031767489))
(2 (3 *e* *e* 0.0145319364650233))
```

The resulting `channel.wfst.trained` is (only showing transitions with probability > 0.01):

```
0
(0 (0 c1 a 0.0777289624873271))
(0 (0 c1 b 0.0172355525515377))
(0 (0 c1 c 0.028387968908415))
(0 (0 c1 d 0.0341331530922611))
(0 (0 c1 e 0.127069956066239))
(0 (0 c1 f 0.0223048327137543))
(0 (0 c1 g 0.0233186887461981))
(0 (0 c1 h 0.0537343697195007))
(0 (0 c1 i 0.0642108820547476))
(0 (0 c1 l 0.0439337614058794))
(0 (0 c1 m 0.02500844880027))
(0 (0 c1 n 0.061845217979047))
(0 (0 c1 o 0.0848259547144304))
(0 (0 c1 p 0.0206150726596824))
(0 (0 c1 r 0.0635349780331196))
(0 (0 c1 s 0.0679283541737091))
(0 (0 c1 t 0.0996958431902673))
(0 (0 c1 u 0.0270361608651567))
(0 (0 c1 v 0.0104765123352483))
(0 (0 c1 w 0.0216289286921255))
(0 (0 c1 y 0.0162216965190939))
(0 (0 c2 a 0.0777289624873271))
(0 (0 c2 b 0.0172355525515377))
(0 (0 c2 c 0.028387968908415))
(0 (0 c2 d 0.0341331530922611))
(0 (0 c2 e 0.127069956066239))
(0 (0 c2 f 0.0223048327137543))
(0 (0 c2 g 0.0233186887461981))
(0 (0 c2 h 0.0537343697195007))
(0 (0 c2 i 0.0642108820547476))
(0 (0 c2 l 0.0439337614058794))
(0 (0 c2 m 0.02500844880027))
(0 (0 c2 n 0.061845217979047))
(0 (0 c2 o 0.0848259547144304))
(0 (0 c2 p 0.0206150726596824))
(0 (0 c2 r 0.0635349780331196))
(0 (0 c2 s 0.0679283541737091))
(0 (0 c2 t 0.0996958431902673))
(0 (0 c2 u 0.0270361608651567))
(0 (0 c2 v 0.0104765123352483))
(0 (0 c2 w 0.0216289286921255))
(0 (0 c2 y 0.0162216965190939))
```

We can see that `source.wfsa.trained`, the WFSA we get is completely symmetric. In `channel.wfst.trained`, for each letter, the probability of it been tagged as `c1` and that of `c2` is also the same. This shows that the algorithm cannot distinguish between `c1` and `c2`.

b)

```
carmel -X 0.99999 --train-cascade -HJ TRAIN1 source.wfsa channel.wfst
```

Now we change the converging criteria by `-X 0.99999`. Basically nothing changes, including the both `channel.wfst.trained` and `source.wfsa.trained`, as well as the per-example perplexity of the resulting model.

Clearly, if we start from uniform distribution, we are stuck at this local optimum.

c)

```
carmel -! 30 --train-cascade -HJ TRAIN1 source.wfsa channel.wfst
```

Now we add `-! 30` to require 30 random restarts. Most rounds ends in less than 5 iterations, getting similar performance as we had previously. However, some rounds ends in more than 10 iterations and get considerably better performance. The final best per-example perplexity is 2^282.595, which is from a 13-iteration round.

`source.wfsa.trained` now looks like:

```
3
(0 (1 *e* c1 0.59262594005618))
(0 (2 *e* c2 0.40737405994382))
(1 (1 *e* c1 0.266031284450264))
(1 (2 *e* c2 0.71302911532071))
(1 (3 *e* *e* 0.0209396002290256))
(2 (1 *e* c1 0.883160112749434))
(2 (2 *e* c2 0.110185696680701))
(2 (3 *e* *e* 0.00665419056986494))
```

Note that it is no longer symmetric. `channel.wfst.trained` now looks like:

```
0
(0 (0 c1 b 0.0305815812598713))
(0 (0 c1 c 0.0409740721231034))
(0 (0 c1 d 0.0439779290600126))
(0 (0 c1 f 0.0380669803550327))
(0 (0 c1 g 0.0232108304963006))
(0 (0 c1 h 0.0974204787373134))
(0 (0 c1 l 0.0756489178339035))
(0 (0 c1 m 0.0452412949239767))
(0 (0 c1 n 0.112106896022319))
(0 (0 c1 o 0.0181464532168415))
(0 (0 c1 p 0.0246595943564622))
(0 (0 c1 r 0.115001657089756))
(0 (0 c1 s 0.122587835121227))
(0 (0 c1 t 0.0954867265343356))
(0 (0 c1 u 0.0134187640480944))
(0 (0 c1 v 0.0189979808849185))
(0 (0 c1 w 0.0327574616679341))
(0 (0 c1 y 0.0183973333112048))
(0 (0 c2 a 0.161125175234571))
(0 (0 c2 c 0.0129142913580189))
(0 (0 c2 d 0.0220297334489913))
(0 (0 c2 e 0.274774254737849))
(0 (0 c2 g 0.0234512924402563))
(0 (0 c2 i 0.140364144391813))
(0 (0 c2 k 0.0011699008639412))
(0 (0 c2 l 0.00494233604915171))
(0 (0 c2 m 0.000133669728091023))
(0 (0 c2 o 0.166803441136973))
(0 (0 c2 p 0.0156426341285482))
(0 (0 c2 t 0.104870638978522))
```

```
(0 (0 c2 u 0.0437777371758541))
(0 (0 c2 y 0.0135469129412346))
```

`c1` and `c2` are treated differently now. For example, letters like `a`, `e`, `i`, `o`, `u` clearly prefer `c2`.

If we re-run the algorithm, we get similar results (although `c1` and `c2` might switch positions).

d)

```
carmel -X 0.99999 -! 30 --train-cascade -HJ TRAIN1 source.wfsa channel.wfst
```

By adding `-! 30` we get the per-example perplexity of `2^282.014`. The algorithm usually needs around 100 iterations to converge. Here's `source.wfsa.trained`:

```
3
(0 (1 *e* c1 0.334328500391879))
(0 (2 *e* c2 0.66567149960812))
(1 (1 *e* c1 0.136061268039988))
(1 (2 *e* c2 0.856917664301208))
(1 (3 *e* *e* 0.007021067658803 39))
(2 (1 *e* c1 0.685821955342742))
(2 (2 *e* c2 0.293607545915665))
(2 (3 *e* *e* 0.0205704987415933))
```

And `channel.wfst.trained`:

```
0
(0 (0 c1 a 0.174315907381127))
(0 (0 c1 b 2.26030719319958e-13))
(0 (0 c1 d 0.017285365726644))
(0 (0 c1 e 0.285083382526707))
(0 (0 c1 g 0.0219442261253233))
(0 (0 c1 i 0.144034009609265))
(0 (0 c1 o 0.18810409613281))
(0 (0 c1 p 0.0105904089500579))
(0 (0 c1 t 0.083784404063381))
(0 (0 c1 u 0.0444827440049435))
(0 (0 c1 y 0.0167059121529042))
(0 (0 c2 b 0.0310925323680459))
(0 (0 c2 c 0.0460659682779883))
(0 (0 c2 d 0.0476783787984359))
(0 (0 c2 f 0.0402350102292547))
(0 (0 c2 g 0.0244237244414161))
(0 (0 c2 h 0.0969355029316754))
(0 (0 c2 l 0.079253674758478))
(0 (0 c2 m 0.0451146548087938))
(0 (0 c2 n 0.111567322023095))
(0 (0 c2 p 0.0286746670520287))
(0 (0 c2 r 0.11461560951419))
(0 (0 c2 s 0.121811968358247))
(0 (0 c2 t 0.11248826688405))
(0 (0 c2 u 0.0130095174073789))
(0 (0 c2 v 0.0188993824199032))
(0 (0 c2 w 0.0355639714599405))
(0 (0 c2 y 0.0158323985117031))
```

We can see that the resulting probabilities are further polarized.

If we instead use `-X 0.999999` instead, the algorithm would often need more than 300 iterations to converge. The resulting per-example perplexity is 2^282.003, which is not such a huge improvement.

2

Code (written in scala)

```
import collection.mutable
```

```
/**
 * Compute log of base 2
 */
def log2(d: Double) = math.log(d) / math.log(2.0)

/**
 * Given l1 = log(x), l2 = log(y), compute the approximate value of log(x+y)
 */
def logPlus(l1: Double, l2: Double) = {
  assert(!l1.isNaN && !l2.isNaN)
  val l_larger = math.max(l1, l2)
  val l_smaller = math.min(l1, l2)
  if (l_smaller.isNegInfinity) l_larger
  else if (l_larger - l_smaller > 32) l_larger
  else l_larger + log2(1 + math.pow(2, l_smaller - l_larger))
}

/**
 * Normalize the given log vector, so that 2 to the power of each element sum up to 1.
 */
def normalizeLog(log_vec: IndexedSeq[Double]) = {
  val log_sum = log_vec.reduce(logPlus)
  log_vec.map(_ - log_sum)
}

/**
 * Create a random double Vector whose elements are positive and sum up to 1
 */
def randomLogProbVector(size: Int) =
  normalizeLog(IndexedSeq.fill(size)(util.Random.nextDouble()))

/**
 * Create a random Vector[Vector[Double]], each row of which sum up to 1
 */
def randomProbLogMatrix(nRows: Int, nColumns: Int) =
  IndexedSeq.fill(nRows)(randomLogProbVector(nColumns))

/**
 * Parameters for part-of-speech tagging
 * @param words The vocabulary.
 *              The index of each word corresponds to the word index in b
 * @param nTag Number of distinct tags
 * @param t t[tag1, tag2] = log P(tag2|tag1), i.e. bi-gram model
 * @param b b[tag, word] = log P(word|tag)
 */
case class Model(words: IndexedSeq[String], nTag: Int,
                 t: IndexedSeq[IndexedSeq[Double]],
                 b: IndexedSeq[IndexedSeq[Double]]) {
  assert(words.distinct.size == words.size)

  lazy val wordToIndex = words.zipWithIndex.toMap

  /**
   * Tags are indexed from 1 to nTag.
   * 0 represents the start and the end of a tag sequence.
   */
  val tags = 1 to nTag

  def params = (words, nTag, t, b)
}

object Model {
  /**
   * Create a model with random parameters for random restarts.
   */
  def newRandomModel(words: IndexedSeq[String], nTag: Int) =
    Model(words, nTag,
      (Double.NaN +: randomLogProbVector(nTag)) +: randomProbLogMatrix(nTag, nTag + 1),
      mutable.IndexedSeq.fill(words.size)(Double.NaN) +: randomProbLogMatrix(nTag, words.size))

  /**
   * Construct a lattice and compute the best part-of-speech tagging using Viterbi decoding
   * @return The lattice and the best path
   */
  def computePosTags(example: IndexedSeq[Int], model: Model) = {
    val (_, nTag, t, b) = model.params
    def newLattice[T](v: T) = IndexedSeq.fill(example.size)(
      mutable.IndexedSeq.fill(nTag + 1)(v)
    )
    val p = newLattice(Double.NaN)
    val track = newLattice(0)

    for (i <- 0 until example.size; tag <- model.tags) {
      if (i == 0) {
        p(i)(tag) = t(0)(tag) + b(tag)(example(i))
      }
      else {
        val (prevP, prevTag) = model.tags.map(prev_tag =>
          p(i - 1)(prev_tag) + t(prev_tag)(tag)
        ).zip(model.tags).maxBy(_._1)
```

```
        p(i)(tag) = prevP + b(tag)(example(i))
        track(i)(tag) = prevTag
      }
    }

    val path = {
      val (_, lastTag) = p(example.size - 1).zipWithIndex.drop(1).maxBy(_._1)

      def buildPath(i: Int, tag: Int, accu: List[Int] = Nil): IndexedSeq[Int] =
        if (i == 0) (tag :: accu).toIndexedSeq
        else buildPath(i - 1, track(i)(tag), tag :: accu)

      buildPath(example.size - 1, lastTag)
    }

    (p.map(_.toIndexedSeq), path)
  }

  /**
   * Construct a lattice and compute the best part-of-speech tagging using Viterbi decoding
   * @return The lattice and the best path
   */
  def computePosTags[X: ClassManifest](example: IndexedSeq[String],
                                       model: Model): (IndexedSeq[IndexedSeq[Double]], IndexedSeq[Int]) =
    computePosTags(example.map(model.wordToIndex), model)

  /**
   * Construct a lattice and compute the α values (forward)
   * @return α and the log probability of the example
   */
  def computeA(example: IndexedSeq[Int], model: Model) = {
    val (_, nTag, t, b) = model.params
    val α = IndexedSeq.fill(example.size)(
      mutable.IndexedSeq.fill(nTag + 1)(Double.NaN)
    )

    for (i <- 0 until example.size; tag <- model.tags) {
      α(i)(tag) =
        if (i == 0)
          t(0)(tag) + b(tag)(example(i))
        else
          model.tags.map(prev_tag =>
            α(i - 1)(prev_tag) + t(prev_tag)(tag)
          ).reduce(logPlus) + b(tag)(example(i))
    }

    val α_end = (model.tags).map(tag => α(example.size - 1)(tag) + t(tag)(0)).reduce(logPlus)

    (α.map(_.toIndexedSeq), α_end)
  }

  /**
   * Construct a lattice and compute the β values (backward)
   * @return β and the log probability of the example
   */
  def computeB(example: IndexedSeq[Int], model: Model) = {
    val (_, nTag, t, b) = model.params
    val β = IndexedSeq.fill(example.size)(
      mutable.IndexedSeq.fill(nTag + 1)(Double.NaN)
    )

    for (i <- (example.size - 1) to 0 by -1; tag <- model.tags) {
      β(i)(tag) =
        if (i == example.size - 1)
          t(tag)(0)
        else model.tags.map(next_tag =>
          t(tag)(next_tag) + b(next_tag)(example(i + 1)) + β(i + 1)(next_tag)
        ).reduce(logPlus)
    }

    val β_start = (model.tags).map(tag => t(0)(tag) + b(tag)(example(0)) + β(0)(tag)).reduce(logPlus)

    (β.map(_.toIndexedSeq), β_start)
  }
}

/**
 * The recursive helper function that computes the model for each random restart
 */
@annotation.tailrec
def computeForwardBackwardImpl(indexedCorpus: Seq[IndexedSeq[Int]],
                               model: Model, nIter: Int,
                               prevLogJointProb: Double,
                               isConverged: (Int, Double, Double) => Boolean): (Model, Double) = {
  import Model._
  val (words, nTag, t, b) = model.params
  val ex_α_β_logProb = indexedCorpus.map(ex => {
    val (α, α_end) = computeA(ex, model)
    val (β, β_start) = computeB(ex, model)
```

```
    assert(α_end - β_start < 1e10)

    (ex, α, β, α_end)
  })
  val logJointProb = ex_α_β_logProb.map(_._4).reduce(_ + _)

  if (nIter != 0 && isConverged(nIter, prevLogJointProb, logJointProb))
    (model, logJointProb)
  else {
    val new_model = {
      // Container for collecting partial counts for t
      val c_t = IndexedSeq.fill(nTag + 1)(mutable.IndexedSeq.fill(nTag + 1)(0.0))
      // Container for collecting partial counts for b
      val c_b =
        mutable.IndexedSeq.fill(words.size)(Double.NaN) +:
          IndexedSeq.fill(nTag)(mutable.IndexedSeq.fill(words.size)(0.0))

      ex_α_β_logProb.foreach({
        case (ex, α, β, logProb) =>
          // For each example, collect partial counts for t
          for (tag1 <- model.tags) {
            c_t(0)(tag1) =
              logPlus(c_t(0)(tag1),
                t(0)(tag1) + b(tag1)(ex(0)) + β(0)(tag1) - logProb)
            c_t(tag1)(0) =
              logPlus(c_t(tag1)(0),
                α(ex.size - 1)(tag1) + t(tag1)(0) - logProb)
            for (i <- 1 to ex.size - 2; tag2 <- model.tags) {
              c_t(tag1)(tag2) =
                logPlus(c_t(tag1)(tag2),
                  α(i)(tag1) + t(tag1)(tag2) + b(tag2)(ex(i + 1)) + β(i + 1)(tag2) - logProb)
            }
          }
          // For each example, collect partial counts for b
          for (tag <- model.tags; i <- 0 until ex.size) {
            c_b(tag)(ex(i)) =
              logPlus(c_b(tag)(ex(i)),
                α(i)(tag) + β(i)(tag) - logProb)
          }
      })

      // Compute the revised t and b by normalizing the partial counts
      val new_t = (0.0 +: normalizeLog(c_t.head.drop(1))) +: c_t.tail.map(normalizeLog)
      val new_b = mutable.IndexedSeq.fill(words.size)(Double.NaN) +: c_b.tail.map(normalizeLog)

      Model(words, nTag, new_t, new_b)
    }

    computeForwardBackwardImpl(indexedCorpus, new_model, nIter + 1, logJointProb, isConverged)
  }
}

/**
 * Compute parameters for unsupervised POS tagging
 * @param corpus A collection of training examples
 * @param nTag Number of tags
 * @param isConverged A function for testing convergence.
 *                    The three parameters are: the number of current iteration,
 *                    log training probability from last iteration and
 *                    log training probability of current iteration
 * @param nRandomRestart Number of random restarts
 * @param onEachRestartCompleted The function is called asynchronously after each random restart.
 *                               The three parameters are: current round of random restart,
 *                               the resulting model and the log training probability
 * @return The trained model and the log probability it assigns to corpus
 */
def computeForwardBackward(corpus: Seq[IndexedSeq[String]],
                           nTag: Int,
                           isConverged: (Int, Double, Double) => Boolean = (i, l0, l1) => l1 < l0 && (l1 / l0) > 0.99999,
                           nRandomRestart: Int = 10,
                           onEachRestartCompleted: (Int, Model, Double) => Unit = null) = {
  import scala.actors.Futures._

  val words = corpus.flatten.distinct.toIndexedSeq
  val wordToIndex = words.zipWithIndex.toMap
  val indexedCorpus = corpus.map(_.map(wordToIndex).toIndexedSeq)

  val (bestModel, bestLogJointProb) = (1 to nRandomRestart).map(currRound => {
    val initModel = Model.newRandomModel(words, nTag)
    val (model, logJointProb) =
      computeForwardBackwardImpl(
        indexedCorpus, initModel, 0, Double.NegativeInfinity, isConverged)

    if (onEachRestartCompleted != null) {
      future {
        onEachRestartCompleted(currRound, model, logJointProb)
      }
    }
    (model, logJointProb)
  }).maxBy(_._2)
```

```
  (bestModel, bestLogJointProb)
}

//
// Using the functions defined above:
//

def onEachRestartCompleted(currRound: Int, model: Model, logJointProb: Double) {
  Console.err.println("Round " + currRound + " completed. Corpus probability = 2^" + logJointProb)
  Console.err.println()
}

def isConverged(currIter: Int, prevLogJointProb: Double, currLogJointProb: Double) = {
  Console.err.println("Iteration " + currIter + " completed. Corpus probability = 2^" + currLogJointProb)
  val r = currLogJointProb / prevLogJointProb
  if (r > 0.99999 && r <= 1) {
    Console.err.println("Converged.")
    true
  }
  else false
}

val nTag = 2
val corpus =
  io.Source.fromFile( """./TRAIN1""").getLines().
    map(_.trim).filterNot(_.isEmpty).
    map(_.split(' ').toIndexedSeq).toIndexedSeq

val (model, prob) = computeForwardBackward(corpus, nTag, isConverged, 10, onEachRestartCompleted)
println("Best corpus probability = 2^" + prob)
model.b.zipWithIndex.drop(1).foreach({
  case (words, i) => {
    println("Tag " + i + ":")
    words.zipWithIndex.
      map(t => (t._1, model.words(t._2))).
      sortBy(- _._1).
      foreach({
      case (logProb, word) =>
        println(word + ":\t" + math.pow(2, logProb))
    })
    println()
  }
})

println(corpus.head.take(50).mkString)
println(Model.computePosTags(corpus.head.take(50), model)._2.mkString)
```

Channel Probabilities

```
Tag 1:
e:  0.26723263120850965
o:  0.1742144724081009
a:  0.16241528053777948
i:  0.13459175976233723
t:  0.09250527457063015
u:  0.045706563488934836
g:  0.023835400068638168
d:  0.020243241499339377
y:  0.017763048880717898
p:  0.013826321071787386
c:  0.010897655172179808
...

Tag 2:
s:  0.11954613586523556
r:  0.11710365872868869
n:  0.11394651203358283
t:  0.10399242013673807
h:  0.09858296726538651
l:  0.0791149412622195
d:  0.0463989183604296
m:  0.046084581628295936
c:  0.04398268695675024
f:  0.0394336166422655
w:  0.03436263122112723
b:  0.031766665192268395
p:  0.02711885813124016
g:  0.023357423909966424
v:  0.019790245742282803
y:  0.015595493365343464
u:  0.011106653742964923
...
```

Viterbi decoding

hebroughthisexpertisetothestudyofrecidivismratesam

21221212121212212212121221221221221212121222121212

Trace

```
Iteration 1 completed. Corpus probability = 2^-12625.654393875171
Iteration 2 completed. Corpus probability = 2^-12625.275330392851
Iteration 3 completed. Corpus probability = 2^-12624.900952332659
Iteration 4 completed. Corpus probability = 2^-12624.500719124026
Iteration 5 completed. Corpus probability = 2^-12624.052196203986
Iteration 6 completed. Corpus probability = 2^-12623.53790181757
Iteration 7 completed. Corpus probability = 2^-12622.944149407813
Iteration 8 completed. Corpus probability = 2^-12622.261064515167
Iteration 9 completed. Corpus probability = 2^-12621.483270702218
Iteration 10 completed. Corpus probability = 2^-12620.610815843955
Iteration 11 completed. Corpus probability = 2^-12619.649905057311
Iteration 12 completed. Corpus probability = 2^-12618.613031022733
Iteration 13 completed. Corpus probability = 2^-12617.518223960746
Iteration 14 completed. Corpus probability = 2^-12616.387394070285
Iteration 15 completed. Corpus probability = 2^-12615.244034552
Iteration 16 completed. Corpus probability = 2^-12614.110766373067
Iteration 17 completed. Corpus probability = 2^-12613.007238112352
Iteration 18 completed. Corpus probability = 2^-12611.94874272761
Iteration 19 completed. Corpus probability = 2^-12610.945667178255
Iteration 20 completed. Corpus probability = 2^-12610.003666156757
Iteration 21 completed. Corpus probability = 2^-12609.124322201676
Iteration 22 completed. Corpus probability = 2^-12608.306031548695
Iteration 23 completed. Corpus probability = 2^-12607.54490448449
Iteration 24 completed. Corpus probability = 2^-12606.835546060816
Iteration 25 completed. Corpus probability = 2^-12606.1716547028
Iteration 26 completed. Corpus probability = 2^-12605.546427026207
Iteration 27 completed. Corpus probability = 2^-12604.952785096828
Iteration 28 completed. Corpus probability = 2^-12604.383452153408
Iteration 29 completed. Corpus probability = 2^-12603.830900697907
Iteration 30 completed. Corpus probability = 2^-12603.28718782523
Iteration 31 completed. Corpus probability = 2^-12602.743679618708
Iteration 32 completed. Corpus probability = 2^-12602.190650218865
Iteration 33 completed. Corpus probability = 2^-12601.61672076703
Iteration 34 completed. Corpus probability = 2^-12601.008075966945
Iteration 35 completed. Corpus probability = 2^-12600.347356278477
Iteration 36 completed. Corpus probability = 2^-12599.612062988715
Iteration 37 completed. Corpus probability = 2^-12598.772216906871
Iteration 38 completed. Corpus probability = 2^-12597.786854390002
Iteration 39 completed. Corpus probability = 2^-12596.598684687755
Iteration 40 completed. Corpus probability = 2^-12595.125799043797
Iteration 41 completed. Corpus probability = 2^-12593.248597720656
Iteration 42 completed. Corpus probability = 2^-12590.788907724092
Iteration 43 completed. Corpus probability = 2^-12587.476380443557
Iteration 44 completed. Corpus probability = 2^-12582.894606295273
Iteration 45 completed. Corpus probability = 2^-12576.396818408288
Iteration 46 completed. Corpus probability = 2^-12566.98302188611
Iteration 47 completed. Corpus probability = 2^-12553.152075075706
Iteration 48 completed. Corpus probability = 2^-12532.822228650986
Iteration 49 completed. Corpus probability = 2^-12503.606515574356
Iteration 50 completed. Corpus probability = 2^-12463.957126429672
Iteration 51 completed. Corpus probability = 2^-12415.317238210964
Iteration 52 completed. Corpus probability = 2^-12363.500363039277
Iteration 53 completed. Corpus probability = 2^-12316.234284454807
Iteration 54 completed. Corpus probability = 2^-12278.228654338556
Iteration 55 completed. Corpus probability = 2^-12249.746225179897
Iteration 56 completed. Corpus probability = 2^-12228.932654901702
Iteration 57 completed. Corpus probability = 2^-12213.641338485211
Iteration 58 completed. Corpus probability = 2^-12202.149394517939
Iteration 59 completed. Corpus probability = 2^-12193.287415410477
Iteration 60 completed. Corpus probability = 2^-12186.294414634358
Iteration 61 completed. Corpus probability = 2^-12180.662900368075
Iteration 62 completed. Corpus probability = 2^-12176.044352570647
Iteration 63 completed. Corpus probability = 2^-12172.196385750545
Iteration 64 completed. Corpus probability = 2^-12168.94992172509
Iteration 65 completed. Corpus probability = 2^-12166.186248691432
Iteration 66 completed. Corpus probability = 2^-12163.820404671651
Iteration 67 completed. Corpus probability = 2^-12161.789465243082
Iteration 68 completed. Corpus probability = 2^-12160.04470982523
Iteration 69 completed. Corpus probability = 2^-12158.546672398576
Iteration 70 completed. Corpus probability = 2^-12157.262172871113
Iteration 71 completed. Corpus probability = 2^-12156.162604986364
Iteration 72 completed. Corpus probability = 2^-12155.222963066757
Iteration 73 completed. Corpus probability = 2^-12154.421271797108
Iteration 74 completed. Corpus probability = 2^-12153.738219524428
Iteration 75 completed. Corpus probability = 2^-12153.156886348612
Iteration 76 completed. Corpus probability = 2^-12152.662513272104
Iteration 77 completed. Corpus probability = 2^-12152.242289183383
Iteration 78 completed. Corpus probability = 2^-12151.885147858218
Iteration 79 completed. Corpus probability = 2^-12151.581574045156
Iteration 80 completed. Corpus probability = 2^-12151.323420168586
Iteration 81 completed. Corpus probability = 2^-12151.10373557553
Iteration 82 completed. Corpus probability = 2^-12150.916609812224
Iteration 83 completed. Corpus probability = 2^-12150.757030742541
Iteration 84 completed. Corpus probability = 2^-12150.620757670822
Iteration 85 completed. Corpus probability = 2^-12150.504209107692
Converged.
Round 1 completed. Corpus probability = 2^-12150.504209107692

Iteration 1 completed. Corpus probability = 2^-12625.471031122217
Iteration 2 completed. Corpus probability = 2^-12625.164979877096
Iteration 3 completed. Corpus probability = 2^-12624.829135041862
Iteration 4 completed. Corpus probability = 2^-12624.445283584719
Iteration 5 completed. Corpus probability = 2^-12623.997476728711
Iteration 6 completed. Corpus probability = 2^-12623.471641644355
```

```
Iteration 7 completed. Corpus probability = 2^-12622.85592099568
Iteration 8 completed. Corpus probability = 2^-12622.141562569228
Iteration 9 completed. Corpus probability = 2^-12621.324144370004
Iteration 10 completed. Corpus probability = 2^-12620.404830741132
Iteration 11 completed. Corpus probability = 2^-12619.39126917542
Iteration 12 completed. Corpus probability = 2^-12618.297731202809
Iteration 13 completed. Corpus probability = 2^-12617.14423609811
Iteration 14 completed. Corpus probability = 2^-12615.954671332081
Iteration 15 completed. Corpus probability = 2^-12614.754245793449
Iteration 16 completed. Corpus probability = 2^-12613.566834114423
Iteration 17 completed. Corpus probability = 2^-12612.412783813252
Iteration 18 completed. Corpus probability = 2^-12611.307564851664
Iteration 19 completed. Corpus probability = 2^-12610.261352500298
Iteration 20 completed. Corpus probability = 2^-12609.279384373165
Iteration 21 completed. Corpus probability = 2^-12608.36280285553
Iteration 22 completed. Corpus probability = 2^-12607.509688575346
Iteration 23 completed. Corpus probability = 2^-12606.716062543466
Iteration 24 completed. Corpus probability = 2^-12605.976729445529
Iteration 25 completed. Corpus probability = 2^-12605.285915557526
Iteration 26 completed. Corpus probability = 2^-12604.637707787082
Iteration 27 completed. Corpus probability = 2^-12604.026326791203
Iteration 28 completed. Corpus probability = 2^-12603.44627480407
Iteration 29 completed. Corpus probability = 2^-12602.89239596228
Iteration 30 completed. Corpus probability = 2^-12602.359879643815
Iteration 31 completed. Corpus probability = 2^-12601.844229289185
Iteration 32 completed. Corpus probability = 2^-12601.341212098261
Iteration 33 completed. Corpus probability = 2^-12600.846799482213
Iteration 34 completed. Corpus probability = 2^-12600.357104180204
Iteration 35 completed. Corpus probability = 2^-12599.868317266006
Iteration 36 completed. Corpus probability = 2^-12599.376646555831
Iteration 37 completed. Corpus probability = 2^-12598.8782569081
Iteration 38 completed. Corpus probability = 2^-12598.369212355288
Iteration 39 completed. Corpus probability = 2^-12597.845419775003
Iteration 40 completed. Corpus probability = 2^-12597.302573782817
Iteration 41 completed. Corpus probability = 2^-12596.736102651426
Iteration 42 completed. Corpus probability = 2^-12596.141115293301
Iteration 43 completed. Corpus probability = 2^-12595.512349675162
Iteration 44 completed. Corpus probability = 2^-12594.84412346654
Iteration 45 completed. Corpus probability = 2^-12594.130288279955
Iteration 46 completed. Corpus probability = 2^-12593.364189564767
Iteration 47 completed. Corpus probability = 2^-12592.53863510855
Iteration 48 completed. Corpus probability = 2^-12591.645876220053
Iteration 49 completed. Corpus probability = 2^-12590.677607049038
Iteration 50 completed. Corpus probability = 2^-12589.624989149326
Iteration 51 completed. Corpus probability = 2^-12588.478710258327
Iteration 52 completed. Corpus probability = 2^-12587.229088184753
Iteration 53 completed. Corpus probability = 2^-12585.866232317756
Iteration 54 completed. Corpus probability = 2^-12584.38027599109
Iteration 55 completed. Corpus probability = 2^-12582.761691859034
Iteration 56 completed. Corpus probability = 2^-12581.001698476668
Iteration 57 completed. Corpus probability = 2^-12579.092758491031
Iteration 58 completed. Corpus probability = 2^-12577.029157215411
Iteration 59 completed. Corpus probability = 2^-12574.807636781698
Iteration 60 completed. Corpus probability = 2^-12572.428050196939
Iteration 61 completed. Corpus probability = 2^-12569.893998270349
Iteration 62 completed. Corpus probability = 2^-12567.213426173925
Iteration 63 completed. Corpus probability = 2^-12564.399183697808
Iteration 64 completed. Corpus probability = 2^-12561.469578787955
Iteration 65 completed. Corpus probability = 2^-12558.448948176469
Iteration 66 completed. Corpus probability = 2^-12555.36819921219
Iteration 67 completed. Corpus probability = 2^-12552.265131836451
Iteration 68 completed. Corpus probability = 2^-12549.18416728041
Iteration 69 completed. Corpus probability = 2^-12546.174992786351
Iteration 70 completed. Corpus probability = 2^-12543.2897206827
Iteration 71 completed. Corpus probability = 2^-12540.578553675827
Iteration 72 completed. Corpus probability = 2^-12538.084583502923
Iteration 73 completed. Corpus probability = 2^-12535.83894107299
Iteration 74 completed. Corpus probability = 2^-12533.857671339763
Iteration 75 completed. Corpus probability = 2^-12532.141222123802
Iteration 76 completed. Corpus probability = 2^-12530.676522615036
Iteration 77 completed. Corpus probability = 2^-12529.440789677281
Iteration 78 completed. Corpus probability = 2^-12528.405852515005
Iteration 79 completed. Corpus probability = 2^-12527.541976573599
Iteration 80 completed. Corpus probability = 2^-12526.820642100416
Iteration 81 completed. Corpus probability = 2^-12526.216192730912
Iteration 82 completed. Corpus probability = 2^-12525.706552744332
Iteration 83 completed. Corpus probability = 2^-12525.273308309203
Iteration 84 completed. Corpus probability = 2^-12524.90142460877
Iteration 85 completed. Corpus probability = 2^-12524.578798869696
Iteration 86 completed. Corpus probability = 2^-12524.295774826654
Iteration 87 completed. Corpus probability = 2^-12524.044686369016
Iteration 88 completed. Corpus probability = 2^-12523.819460087518
Iteration 89 completed. Corpus probability = 2^-12523.615284396401
Iteration 90 completed. Corpus probability = 2^-12523.428341691557
Iteration 91 completed. Corpus probability = 2^-12523.25559530525
Iteration 92 completed. Corpus probability = 2^-12523.094621851722
Iteration 93 completed. Corpus probability = 2^-12522.943480101976
Iteration 94 completed. Corpus probability = 2^-12522.800608739926
Iteration 95 completed. Corpus probability = 2^-12522.664746717688
Iteration 96 completed. Corpus probability = 2^-12522.534871189973
Iteration 97 completed. Corpus probability = 2^-12522.410149087622
Converged.
Round 2 completed. Corpus probability = 2^-12522.410149087622

Iteration 1 completed. Corpus probability = 2^-12626.611412053837
Iteration 2 completed. Corpus probability = 2^-12626.248651066076
Iteration 3 completed. Corpus probability = 2^-12625.948042117674
Iteration 4 completed. Corpus probability = 2^-12625.671137940397
Iteration 5 completed. Corpus probability = 2^-12625.38976501065
Iteration 6 completed. Corpus probability = 2^-12625.081800865839
Iteration 7 completed. Corpus probability = 2^-12624.728545091253
Iteration 8 completed. Corpus probability = 2^-12624.313236046906
Iteration 9 completed. Corpus probability = 2^-12623.820440927586
Iteration 10 completed. Corpus probability = 2^-12623.23614743493
Iteration 11 completed. Corpus probability = 2^-12622.548403247329
```

```
Iteration 12 completed. Corpus probability = 2^-12621.748286134853
Iteration 13 completed. Corpus probability = 2^-12620.83086944761
Iteration 14 completed. Corpus probability = 2^-12619.795743419134
Iteration 15 completed. Corpus probability = 2^-12618.64666276524
Iteration 16 completed. Corpus probability = 2^-12617.390087201766
Iteration 17 completed. Corpus probability = 2^-12616.032721935559
Iteration 18 completed. Corpus probability = 2^-12614.578472523002
Iteration 19 completed. Corpus probability = 2^-12613.025294257672
Iteration 20 completed. Corpus probability = 2^-12611.362160921904
Iteration 21 completed. Corpus probability = 2^-12609.565905282772
Iteration 22 completed. Corpus probability = 2^-12607.597168948829
Iteration 23 completed. Corpus probability = 2^-12605.394218986554
Iteration 24 completed. Corpus probability = 2^-12602.862834280584
Iteration 25 completed. Corpus probability = 2^-12599.85955337848
Iteration 26 completed. Corpus probability = 2^-12596.163890392663
Iteration 27 completed. Corpus probability = 2^-12591.432156382756
Iteration 28 completed. Corpus probability = 2^-12585.120917587501
Iteration 29 completed. Corpus probability = 2^-12576.363058817076
Iteration 30 completed. Corpus probability = 2^-12563.782178403377
Iteration 31 completed. Corpus probability = 2^-12545.27434415572
Iteration 32 completed. Corpus probability = 2^-12517.95898992658
Iteration 33 completed. Corpus probability = 2^-12478.894540936131
Iteration 34 completed. Corpus probability = 2^-12427.298638047609
Iteration 35 completed. Corpus probability = 2^-12367.188061718238
Iteration 36 completed. Corpus probability = 2^-12306.750492770041
Iteration 37 completed. Corpus probability = 2^-12254.52287418287
Iteration 38 completed. Corpus probability = 2^-12215.72539513298
Iteration 39 completed. Corpus probability = 2^-12190.354241547115
Iteration 40 completed. Corpus probability = 2^-12175.10760063516
Iteration 41 completed. Corpus probability = 2^-12166.24873437696
Iteration 42 completed. Corpus probability = 2^-12161.04159616957
Iteration 43 completed. Corpus probability = 2^-12157.86163001096
Iteration 44 completed. Corpus probability = 2^-12155.822558047454
Iteration 45 completed. Corpus probability = 2^-12154.447991028994
Iteration 46 completed. Corpus probability = 2^-12153.477626455591
Iteration 47 completed. Corpus probability = 2^-12152.764874461935
Iteration 48 completed. Corpus probability = 2^-12152.224039374076
Iteration 49 completed. Corpus probability = 2^-12151.802913583688
Iteration 50 completed. Corpus probability = 2^-12151.468307706842
Iteration 51 completed. Corpus probability = 2^-12151.198226876035
Iteration 52 completed. Corpus probability = 2^-12150.977519834405
Iteration 53 completed. Corpus probability = 2^-12150.795382572998
Iteration 54 completed. Corpus probability = 2^-12150.643877597182
Iteration 55 completed. Corpus probability = 2^-12150.517025007073
Iteration 56 completed. Corpus probability = 2^-12150.41022511362
Converged.
Round 3 completed. Corpus probability = 2^-12150.41022511362

Iteration 1 completed. Corpus probability = 2^-12625.526189328817
Iteration 2 completed. Corpus probability = 2^-12625.192876918516
Iteration 3 completed. Corpus probability = 2^-12624.839549564305
Iteration 4 completed. Corpus probability = 2^-12624.449498250819
Iteration 5 completed. Corpus probability = 2^-12624.010545807168
Iteration 6 completed. Corpus probability = 2^-12623.514071169999
Iteration 7 completed. Corpus probability = 2^-12622.95462843845
Iteration 8 completed. Corpus probability = 2^-12622.329804978324
Iteration 9 completed. Corpus probability = 2^-12621.640067705861
Iteration 10 completed. Corpus probability = 2^-12620.888438157968
Iteration 11 completed. Corpus probability = 2^-12620.079937887866
Iteration 12 completed. Corpus probability = 2^-12619.220845936661
Iteration 13 completed. Corpus probability = 2^-12618.317881540437
Iteration 14 completed. Corpus probability = 2^-12617.377442611027
Iteration 15 completed. Corpus probability = 2^-12616.404989430835
Iteration 16 completed. Corpus probability = 2^-12615.404583640142
Iteration 17 completed. Corpus probability = 2^-12614.378508105397
Iteration 18 completed. Corpus probability = 2^-12613.326831634691
Iteration 19 completed. Corpus probability = 2^-12612.24675337976
Iteration 20 completed. Corpus probability = 2^-12611.13155295955
Iteration 21 completed. Corpus probability = 2^-12609.968954104337
Iteration 22 completed. Corpus probability = 2^-12608.738640905278
Iteration 23 completed. Corpus probability = 2^-12607.40849618297
Iteration 24 completed. Corpus probability = 2^-12605.92879231481
Iteration 25 completed. Corpus probability = 2^-12604.222941510232
Iteration 26 completed. Corpus probability = 2^-12602.172288976506
Iteration 27 completed. Corpus probability = 2^-12599.590393877894
Iteration 28 completed. Corpus probability = 2^-12596.178529038509
Iteration 29 completed. Corpus probability = 2^-12591.44751584225
Iteration 30 completed. Corpus probability = 2^-12584.580263811755
Iteration 31 completed. Corpus probability = 2^-12574.19690918251
Iteration 32 completed. Corpus probability = 2^-12557.993106314903
Iteration 33 completed. Corpus probability = 2^-12532.352420807983
Iteration 34 completed. Corpus probability = 2^-12492.566477365732
Iteration 35 completed. Corpus probability = 2^-12435.39451591437
Iteration 36 completed. Corpus probability = 2^-12365.094658377215
Iteration 37 completed. Corpus probability = 2^-12296.344402091108
Iteration 38 completed. Corpus probability = 2^-12243.699765331889
Iteration 39 completed. Corpus probability = 2^-12209.983544340483
Iteration 40 completed. Corpus probability = 2^-12189.84429203309
Iteration 41 completed. Corpus probability = 2^-12177.643946197159
Iteration 42 completed. Corpus probability = 2^-12169.935127928242
Iteration 43 completed. Corpus probability = 2^-12164.849302422404
Iteration 44 completed. Corpus probability = 2^-12161.357463761331
Iteration 45 completed. Corpus probability = 2^-12158.871360561372
Iteration 46 completed. Corpus probability = 2^-12157.043142950546
Iteration 47 completed. Corpus probability = 2^-12155.660802224702
Iteration 48 completed. Corpus probability = 2^-12154.591223372408
Iteration 49 completed. Corpus probability = 2^-12153.748173958844
Iteration 50 completed. Corpus probability = 2^-12153.073918859935
Iteration 51 completed. Corpus probability = 2^-12152.528487193113
Iteration 52 completed. Corpus probability = 2^-12152.083305870034
Iteration 53 completed. Corpus probability = 2^-12151.717351312489
Iteration 54 completed. Corpus probability = 2^-12151.414768277695
Iteration 55 completed. Corpus probability = 2^-12151.163354818098
Iteration 56 completed. Corpus probability = 2^-12150.953567809023
Iteration 57 completed. Corpus probability = 2^-12150.777848603455
```

```
Iteration 58 completed. Corpus probability = 2^-12150.630151004105
Iteration 59 completed. Corpus probability = 2^-12150.505601043726
Iteration 60 completed. Corpus probability = 2^-12150.400245404619
Converged.
Round 4 completed. Corpus probability = 2^-12150.400245404619

Iteration 1 completed. Corpus probability = 2^-12624.955573631467
Iteration 2 completed. Corpus probability = 2^-12624.5326522422
Iteration 3 completed. Corpus probability = 2^-12624.03479542572
Iteration 4 completed. Corpus probability = 2^-12623.4457081554
Iteration 5 completed. Corpus probability = 2^-12622.75100322323
Iteration 6 completed. Corpus probability = 2^-12621.939211471743
Iteration 7 completed. Corpus probability = 2^-12621.002964281535
Iteration 8 completed. Corpus probability = 2^-12619.940004899234
Iteration 9 completed. Corpus probability = 2^-12618.75368487897
Iteration 10 completed. Corpus probability = 2^-12617.452740995936
Iteration 11 completed. Corpus probability = 2^-12616.050397922409
Iteration 12 completed. Corpus probability = 2^-12614.563067908726
Iteration 13 completed. Corpus probability = 2^-12613.008974513674
Iteration 14 completed. Corpus probability = 2^-12611.406888784526
Iteration 15 completed. Corpus probability = 2^-12609.774963244501
Iteration 16 completed. Corpus probability = 2^-12608.129549383086
Iteration 17 completed. Corpus probability = 2^-12606.483944808337
Iteration 18 completed. Corpus probability = 2^-12604.847150107104
Iteration 19 completed. Corpus probability = 2^-12603.222791202903
Iteration 20 completed. Corpus probability = 2^-12601.608319004337
Iteration 21 completed. Corpus probability = 2^-12599.994470248457
Iteration 22 completed. Corpus probability = 2^-12598.364838830348
Iteration 23 completed. Corpus probability = 2^-12596.695318193055
Iteration 24 completed. Corpus probability = 2^-12594.953137491391
Iteration 25 completed. Corpus probability = 2^-12593.095200545713
Iteration 26 completed. Corpus probability = 2^-12591.065414253027
Iteration 27 completed. Corpus probability = 2^-12588.79063923232
Iteration 28 completed. Corpus probability = 2^-12586.17480365752
Iteration 29 completed. Corpus probability = 2^-12583.09060955945
Iteration 30 completed. Corpus probability = 2^-12579.36819667762
Iteration 31 completed. Corpus probability = 2^-12574.780293026692
Iteration 32 completed. Corpus probability = 2^-12569.02420295227
Iteration 33 completed. Corpus probability = 2^-12561.70334501891
Iteration 34 completed. Corpus probability = 2^-12552.316298722792
Iteration 35 completed. Corpus probability = 2^-12540.269844878962
Iteration 36 completed. Corpus probability = 2^-12524.938250806626
Iteration 37 completed. Corpus probability = 2^-12505.774193062689
Iteration 38 completed. Corpus probability = 2^-12482.422182471086
Iteration 39 completed. Corpus probability = 2^-12454.75781102026
Iteration 40 completed. Corpus probability = 2^-12422.898209128161
Iteration 41 completed. Corpus probability = 2^-12387.345911708288
Iteration 42 completed. Corpus probability = 2^-12349.23401050674
Iteration 43 completed. Corpus probability = 2^-12310.543533322969
Iteration 44 completed. Corpus probability = 2^-12274.01997407357
Iteration 45 completed. Corpus probability = 2^-12242.488377657168
Iteration 46 completed. Corpus probability = 2^-12217.9090865529
Iteration 47 completed. Corpus probability = 2^-12200.40777687811
Iteration 48 completed. Corpus probability = 2^-12188.469662811583
Iteration 49 completed. Corpus probability = 2^-12180.274538401054
Iteration 50 completed. Corpus probability = 2^-12174.460672811383
Iteration 51 completed. Corpus probability = 2^-12170.159742147796
Iteration 52 completed. Corpus probability = 2^-12166.844295205674
Iteration 53 completed. Corpus probability = 2^-12164.197431255823
Iteration 54 completed. Corpus probability = 2^-12162.02742233596
Iteration 55 completed. Corpus probability = 2^-12160.215557485006
Iteration 56 completed. Corpus probability = 2^-12158.685291701386
Iteration 57 completed. Corpus probability = 2^-12157.384422006604
Iteration 58 completed. Corpus probability = 2^-12156.274999899653
Iteration 59 completed. Corpus probability = 2^-12155.32772178537
Iteration 60 completed. Corpus probability = 2^-12154.518845636905
Iteration 61 completed. Corpus probability = 2^-12153.828495832142
Iteration 62 completed. Corpus probability = 2^-12153.239710893917
Iteration 63 completed. Corpus probability = 2^-12152.737878966922
Iteration 64 completed. Corpus probability = 2^-12152.310371653995
Iteration 65 completed. Corpus probability = 2^-12151.946278699863
Iteration 66 completed. Corpus probability = 2^-12151.636195286477
Iteration 67 completed. Corpus probability = 2^-12151.372039211501
Iteration 68 completed. Corpus probability = 2^-12151.14688787915
Iteration 69 completed. Corpus probability = 2^-12150.954830969711
Iteration 70 completed. Corpus probability = 2^-12150.790837202783
Iteration 71 completed. Corpus probability = 2^-12150.650634530875
Iteration 72 completed. Corpus probability = 2^-12150.530603293833
Converged.
Round 5 completed. Corpus probability = 2^-12150.530603293833

Iteration 1 completed. Corpus probability = 2^-12625.750648740952
Iteration 2 completed. Corpus probability = 2^-12625.579776164494
Iteration 3 completed. Corpus probability = 2^-12625.38581590434
Iteration 4 completed. Corpus probability = 2^-12625.156263951445
Iteration 5 completed. Corpus probability = 2^-12624.878931208448
Iteration 6 completed. Corpus probability = 2^-12624.541326468116
Iteration 7 completed. Corpus probability = 2^-12624.130548690433
Iteration 8 completed. Corpus probability = 2^-12623.63364979515
Iteration 9 completed. Corpus probability = 2^-12623.03847831958
Iteration 10 completed. Corpus probability = 2^-12622.334984421404
Iteration 11 completed. Corpus probability = 2^-12621.516862896531
Iteration 12 completed. Corpus probability = 2^-12620.583251297374
Iteration 13 completed. Corpus probability = 2^-12619.540042486213
Iteration 14 completed. Corpus probability = 2^-12618.400312176376
Iteration 15 completed. Corpus probability = 2^-12617.18349450371
Iteration 16 completed. Corpus probability = 2^-12615.913272941089
Iteration 17 completed. Corpus probability = 2^-12614.614570039586
Iteration 18 completed. Corpus probability = 2^-12613.310314390286
Iteration 19 completed. Corpus probability = 2^-12612.018681994086
Iteration 20 completed. Corpus probability = 2^-12610.751254519719
Iteration 21 completed. Corpus probability = 2^-12609.51215676217
Iteration 22 completed. Corpus probability = 2^-12608.297908363114
Iteration 23 completed. Corpus probability = 2^-12607.097548765589
Iteration 24 completed. Corpus probability = 2^-12605.892559441583
```

```
Iteration 25 completed. Corpus probability = 2^-12604.656137760612
Iteration 26 completed. Corpus probability = 2^-12603.351385627238
Iteration 27 completed. Corpus probability = 2^-12601.927893018354
Iteration 28 completed. Corpus probability = 2^-12600.315964794681
Iteration 29 completed. Corpus probability = 2^-12598.417295596066
Iteration 30 completed. Corpus probability = 2^-12596.090155958635
Iteration 31 completed. Corpus probability = 2^-12593.126013172545
Iteration 32 completed. Corpus probability = 2^-12589.212962267398
Iteration 33 completed. Corpus probability = 2^-12583.87985709439
Iteration 34 completed. Corpus probability = 2^-12576.41575324398
Iteration 35 completed. Corpus probability = 2^-12565.768982456491
Iteration 36 completed. Corpus probability = 2^-12550.465245527595
Iteration 37 completed. Corpus probability = 2^-12528.668740412051
Iteration 38 completed. Corpus probability = 2^-12498.626831761658
Iteration 39 completed. Corpus probability = 2^-12459.671847532587
Iteration 40 completed. Corpus probability = 2^-12413.295695754172
Iteration 41 completed. Corpus probability = 2^-12363.067280976165
Iteration 42 completed. Corpus probability = 2^-12313.399088587965
Iteration 43 completed. Corpus probability = 2^-12268.87578668178
Iteration 44 completed. Corpus probability = 2^-12233.409577876215
Iteration 45 completed. Corpus probability = 2^-12208.313724375188
Iteration 46 completed. Corpus probability = 2^-12191.892653589934
Iteration 47 completed. Corpus probability = 2^-12181.283572219352
Iteration 48 completed. Corpus probability = 2^-12174.1989470323
Iteration 49 completed. Corpus probability = 2^-12169.238613955742
Iteration 50 completed. Corpus probability = 2^-12165.60219708057
Iteration 51 completed. Corpus probability = 2^-12162.829432168259
Iteration 52 completed. Corpus probability = 2^-12160.648301208248
Iteration 53 completed. Corpus probability = 2^-12158.892194843716
Iteration 54 completed. Corpus probability = 2^-12157.454681968718
Iteration 55 completed. Corpus probability = 2^-12156.264501211059
Iteration 56 completed. Corpus probability = 2^-12155.271555585568
Iteration 57 completed. Corpus probability = 2^-12154.438966296586
Iteration 58 completed. Corpus probability = 2^-12153.738489746253
Iteration 59 completed. Corpus probability = 2^-12153.147812029558
Iteration 60 completed. Corpus probability = 2^-12152.648898698888
Iteration 61 completed. Corpus probability = 2^-12152.22694438987
Iteration 62 completed. Corpus probability = 2^-12151.869669872407
Iteration 63 completed. Corpus probability = 2^-12151.566826113316
Iteration 64 completed. Corpus probability = 2^-12151.30982659314
Iteration 65 completed. Corpus probability = 2^-12151.091463027818
Iteration 66 completed. Corpus probability = 2^-12150.905678346428
Iteration 67 completed. Corpus probability = 2^-12150.747381164569
Iteration 68 completed. Corpus probability = 2^-12150.612291831514
Iteration 69 completed. Corpus probability = 2^-12150.496813471978
Converged.
Round 6 completed. Corpus probability = 2^-12150.496813471978
…
Round 7 completed. Corpus probability = 2^-12150.485628362794
…
Round 8 completed. Corpus probability = 2^-12150.441134044615
…
Round 9 completed. Corpus probability = 2^-12596.639436894662
…
Round 10 completed. Corpus probability = 2^-12522.239259265558

Best corpus probability = 2^-12150.400245404619
```

3

With 50 random restarts and converge ratio of 0.999999, we get a best P(COPIALE) = 2^-18503.77301940303. Channel probabilities:

```
Tag 1:
ns: 0.039259202328316464
iot:    0.037950275146292935
y..:    0.037260448726297794
ni: 0.03601095348935461
b:  0.03304288121634198
grr:    0.032059607888253544
uh: 0.03136549064404726
zzz:    0.03135206482097907
::  0.03129286609313752
hd: 0.031187821890488066
uu: 0.030578770595778983
eh: 0.030483379496897736
ah: 0.03045959479903865
n:  0.02841541408723681
ih: 0.028145146320489815
oh: 0.027106994829874165
tri:    0.026138940790453105
d:  0.025962180312533188
c:  0.02379890498244502
h.: 0.023110689385701433
gs: 0.02255480572210255
ki: 0.0224263240951017
n.: 0.02191801726352996
k:  0.02134542084275107
f:  0.018363021078925148
```

```
l:  0.017295666677315125
p:  0.017055204824768914
o.: 0.014804288723701117
h:  0.011906297870146174
p.: 0.011864803790394512
r:  0.011356525881027115
zs: 0.011057114701349662
v:  0.010926046325511835
...

Tag 2:
lam:    0.06961881488043858
bar:    0.06183952951525587
z:  0.05396278496717692
j:  0.04895357537685908
three:  0.042083763017707276
plus:   0.037076733391129726
pi: 0.03698074452635962
c.: 0.036562437283646734
g:  0.032474064994024526
r.: 0.03025874634290522
uu: 0.029503023491349676
arr:    0.02905118201357297
ru: 0.02817920459060358
sqp:    0.025142799767552453
hd: 0.024208442358897415
sqi:    0.023782388309731502
m.: 0.02348194638529563
bas:    0.023399054131723673
nu: 0.021683010117935813
mu: 0.02017176846974099
del:    0.01867023323756321
x.: 0.01789910953914455
s.: 0.01481908806543642
mal:    0.013982813490931898
oh: 0.012232562153164296
h.: 0.011949639480126166
d:  0.011565914155014349
inf:    0.010692850903820456
...
```

EM clearly suggests that there are two "kinds" of characters in the corpus, possibly vowels and consonants.