libhmm

Introduction

This is a simple Hidden Markov Model source code written in C++. At this time, Continuous-HMM with Baum-Welch algorithm is only supported. Use of the source code is governed by a BSD license that can be found in the LICENSE file.

Environment

This program was checked only with GCC 3.3.2 on Vine Linux 3.3.2.
This program uses Newmat library for matrix calculation, so it is required to install Newmat preliminarily.

Newmat: `http://www.robertnz.net/nm_intro.htm`

Example code

```
/*including a header file "HMM.h"*/
#include "HMM.h"

 ....

/*Make the matrices for structure definition
  "Number" is tha number of states       */
ColumnVector P(Number);
Matrix T(Number,Number);

/*left-to-right*/
for(int i=0;i<Number;i++){
    if(i) P.element(i) = 0.0;
    else P.element(i) = 1.0;
 }
for(int i=0;i<Number;i++){
    for(int j=0;j<Number;j++){
```

```
T.element(i,j) = 0.0;
if(i==j-1) T.element(i,j) = 1.0;
else if(i==j) T.element(i,j) = 1.0;
    }
}

/*circular
  for(int i=0;i<Number;i++){
  P.element(i) = 1.0;
  }
  for(int i=0;i<Number;i++){
  for(int j=0;j<Number;j++){
  T.element(i,j) = 0.0;
  if(i==j-1) T.element(i,j) = 1.0;
  else if(i==j) T.element(i,j) = 1.0;
  else if(i==Number-1&&j==0) T.element(i,j) = 1.0;
  }
  }
*/

/*elgodic
  for(int i=0;i<Number;i++){
  if(i) P.element(i) = 0.0;
  else P.element(i) = 1.0;
  }
  for(int i=0;i<Number;i++){
  for(int j=0;j<Number;j++){
  T.element(i,j) = 1.0;
  }
  }
*/

/*Make a HMM model (MIX: number of mixture components of a state)*/
HMM model(Number, Dimension, MIX, P, T);

/*set training data*/
/*TRAINING：number of the sequensial data*/
Matrix* train = new Matrix[TRAINING];

...input values: each time observation must be a column element...

/*Initialize HMM model*/

model.set_HMM();
model.init_segmental(train,TRAINING); //for left-to-right
//model.init_k_mean(train,TRAINING);    for other structures

/*estimation*/
model.estimates(train,TRAINING,true);

Matrix test;
... make test sample in the same manner as the training data ...

/*get the log-likelihood of a test data*/
double x = model.get_log_p(test);


/*clean up the memory. (caution: Don't forget the release of the
  region used for Newmat matrices)*/
model.CleanUp();
```

To Do

- implementation of discrete-type HMM
- implementation of diagonal covariant CHMM
- implementation of the Variational Bayes estimation (pending..)