Stopping rules in estimation and simulation

Contents

Relevant theory is summarized in the article

M. Kárný, J. Kracík, I. Nagy, P. Nedoma
When Has Estimation Reached A Steady State?

read the article and examples

1 Basic stopping techniques

Each learning process contains a transient period followed by a stacionary part. The stopping rules determine when the stationary part begins. It means that a *stopping statistic* Q is computed and compared with a *threshold value*. If Q is below it, the estimation already has reached the stationary part.

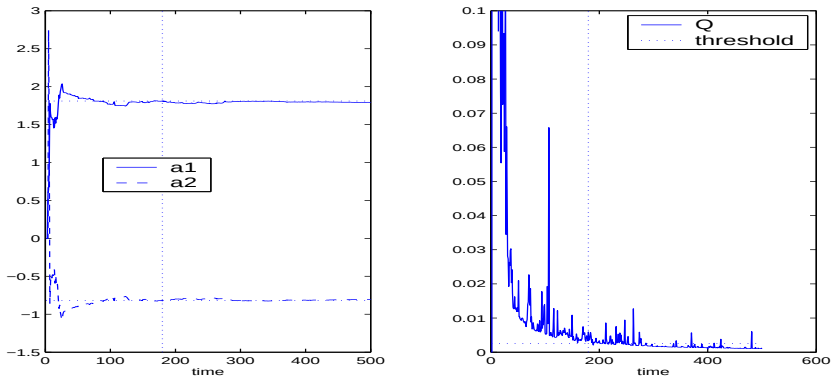
1.1 Learning with normal ARX factors

The estimation of an ARX factor is discussed.

The function *stopstac* supports application of stopping rules. It is called as:

```
[Fac,Q] = stopstac(Fac, dvect)          % stopping criterion
```

where
- *Fac* is the recursively updated factor;
- Q is the stopping statistics;
- *dvect* is data vector used for updating of *Fac*. When missing, the data vector is extracted from signal database using *Fac.str*.



run example view code

Figure 1: Factor estimation with stopping rules

Example: dynamic factor of the 2nd order is estimated and the stopping rule applied. Results are in the Fig. 1. The left part shows trajectory of estimation of the first two regression coefficients, the right part contains the trajectory of the stopping statistics. The theshold value is ploted by dotted line.

The treshold for stopping used is 0.0025.

1.2 Stopping based on a statistics

An "indirect" reasoning about stationarity of a process is based on a statistics computed. This statistics is feeded into a factor estimation. The stationarity of the factor estimation implies the process stationarity.

In learning with a mixture, the statistics used is posterior data likelihood.

Example: dynamic factor of the 2nd order is estimated and the stopping rule applied. Results are in the Fig. 2. The trajectories of data pdf and the stopping statistics are in the lower part of the figure.

1.3 Estimation of credibility intervals

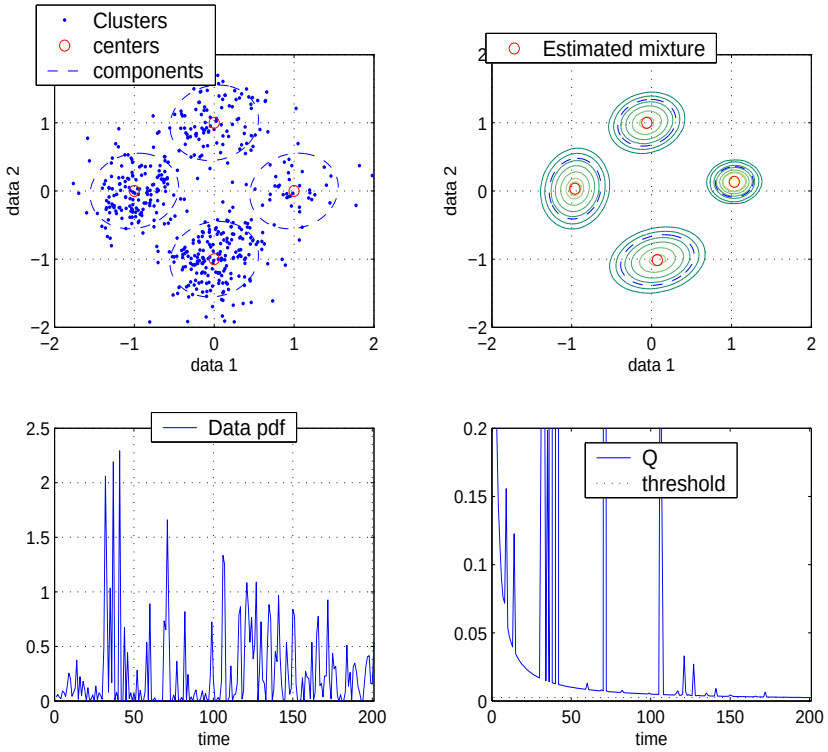
The use of stopping rules speeds substantionaly the mixture estimation. To be sure that this promising result is not random, the estimation should be repeated independently and credibility interval for the moments of stopping computed.

This is done by the function *credits*: For it, the estimation previously done is repeated on independent realizations. With each run, the time needed to reach a stationary state is recorded and Algorithm **??** is used for checking the need for a further realization. The repetition was stopped when the test statistics reached the threshold level 0.5%. The function *credits* makes the estimation of credibility intervals. It computes the stopping statistics and stops processing when the stationary behaviour is reached:

```
[Cl,Cu,Chat,flag,Q] = credits(C, beta, threshold)
```

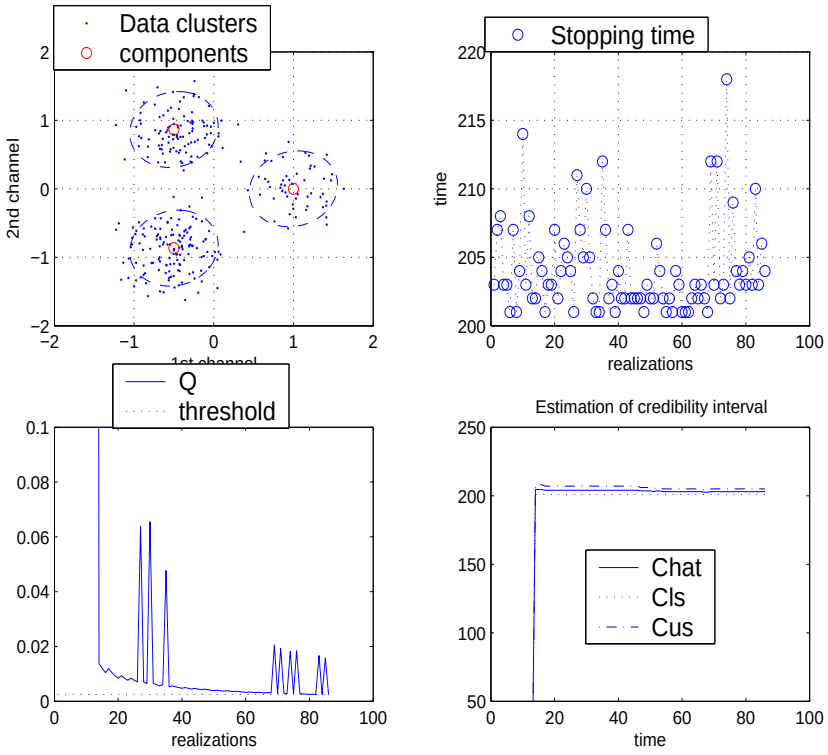
where

Cl	lower credibility bound
Cu	upper credibility bound
Chat	center of the credibility interval
flag	stopping flag 1 - stop, 0 - do not stop data acqusition
Q	value of the stopping statistics
C	vector of independent realizations
beta	credibility level in (0,1)
threshold	upper bound on relative error



run example view code

Figure 2: ARX mixture estimation with stopping rules



run example view code

Figure 3: ARX mixture estimation with stopping rules

Example follows. Repetitive simulation-estimation runs are done. The process is stoped using the posterior data log-likelihood.

Example: Repetitive simulation and estimation of a mixture is done. The stopping times are collected and credibility intervals displayed. Results are in the Fig. 3.

2 Model characteristics via extended simulations

Estimated model characteristics are often obtained by repeated simulations. This is usually time-consuming task. Function *simeval* is designed to collect basic confidence intervals and to record repeated trajectories effectively.

Two benefits of the using *simeval* are:

- individual simulation runs are stopped when stationary state is reached (the function *stopstac* is used);
- the repetitive simulation runs are finished when stacionarity is reached (the function *credits* is employed);
- MEX function solution makes experiments in acceptable computing time.

The function *simeval* makes the simulations:

```
[res, tstop] = simeval(Sim, chns, maxrep, maxdat, threshold)
```

The arguments are:

outputs	
res	cell vector containing results
tstop	stop time of individual trajectories
inputs	
Sim	simulator or a task
chns	list of relevant channels or Facs - cell array of factors
maxrep	maximum number of simulation runs
maxdat	maximum length of trajectories
threshold	threshold value for stopping

The processing results are held in a cell vector. Each cell contains results related to an individual channels. The result fields are:

stats	confidence interval for range and increments
tra	trajectories of individual simulation runs

Note: the confidence intervals are presented as (low border - mean - high border). The evaluation is done by the function *credit*.

2.1 Examples of SISO model evaluation

The example shows application of the function *simeval* and presentation of results. Overshoots of a SISO model in open loop are analyzed. The processing steps are:

- Data are generated
- Model is estimated.
- The characteristics of the model are obtained via simulation with the function *stopstac*.
- Various vharacteristics are evaluated.

Results are in Fig. 4.

2.2 Examples of MIMO model evaluation

The example shows application of the function *simeval* to an MIMO model in open loop. Two models of the previous section are coupled. Results are in Fig. 5.

3 Stopping rules in estimation

The estimation function with stopping rules are:

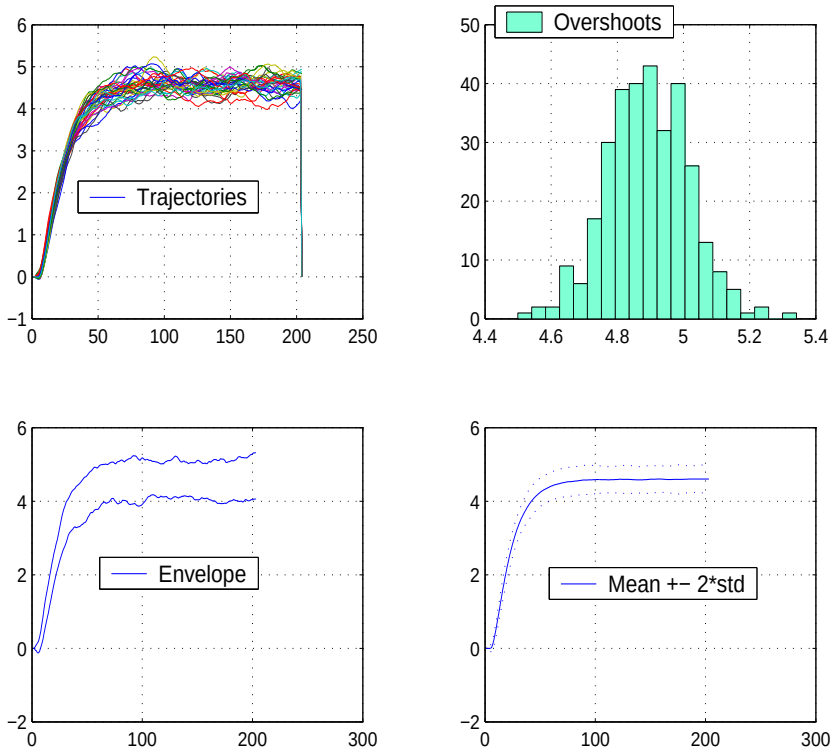
mixestimps	projection method
mixestims	quasi-Bayes estimation
mixestpbs	iterative projection
mixestqbs	iterative quasi-Bayes
mixest	iterative estimation

3.1 Stopping rule in projection and Quasi-Bayes estimation

The function prototypes are:

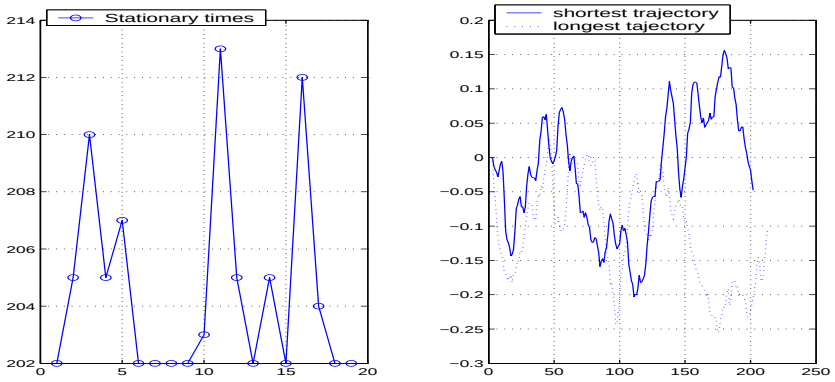
```
[Mix, tstop, Q] = mixestimps(Mix, frg, maxdat, Mixa)
[Mix, tstop, Q] = mixestims(Mix, frg, maxdat, Mixa)
```

The arguments that are different from the versions without stopping:



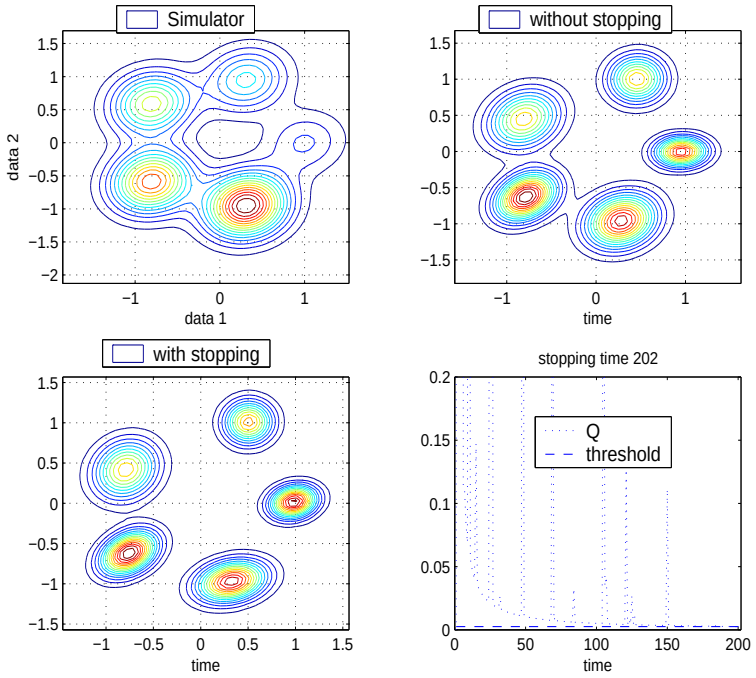
run example view code

Figure 4: Model evaluation via simulation (SISO)



run example view code

Figure 5: Model evaluation via simulation (MIMO)



run example view code z

Figure 6: Stopping rules in quasi-Bayes estimation

tstop	stopping time
Q	value of statistics used for stopping
maxdat	maximum length of data

An example of stopping rules in quasi-Bayes estimation compares esult of estimation without and with stopping. Results are in Fig. 8

3.2 Stoping rules in iterative estimation

The function prototypes are:

```
[Mix, tstop, Q] = mixestimps(Mix, frg, maxdat, Mixa)
[Mix, tstop, Q] = mixestims(Mix, frg, maxdat, Mixa)
```

Results of comparison of all estimation functions are shown in the Fig. 7.

Example: Data are generated, *mixestims* is called in the a cycle of $TIME$, intermediate results are stored and displayed in Fig. 8. The core of processing:

```
for TIME=maxtd+1:maxdat
    [Mix, tstop, Q] = mixestims(Mix);
    ...
    if tstop, break; end
end
```

3.3 Stopping rules in repeated quasi-Bayes estimation

The function *mixestims*:

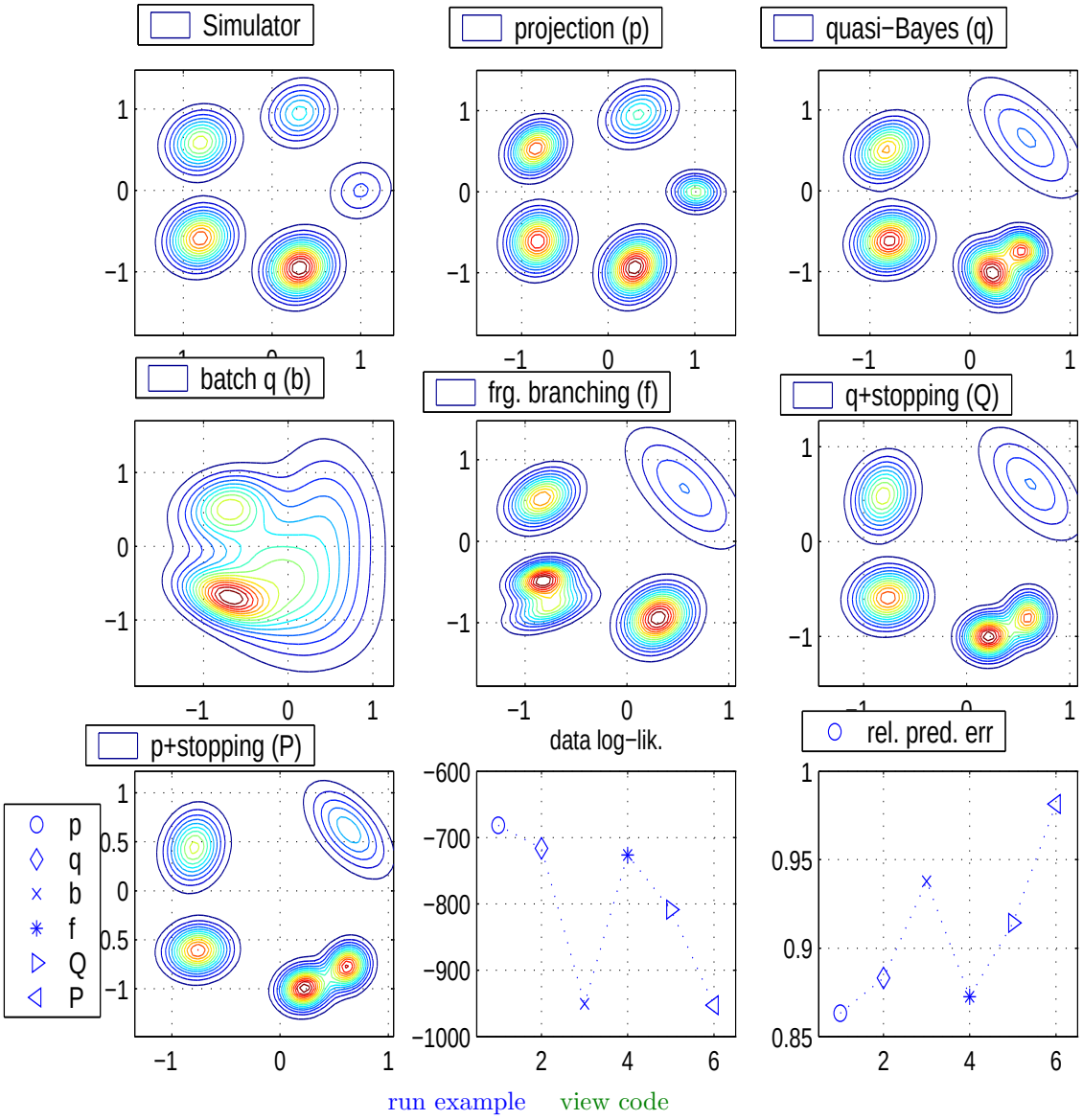
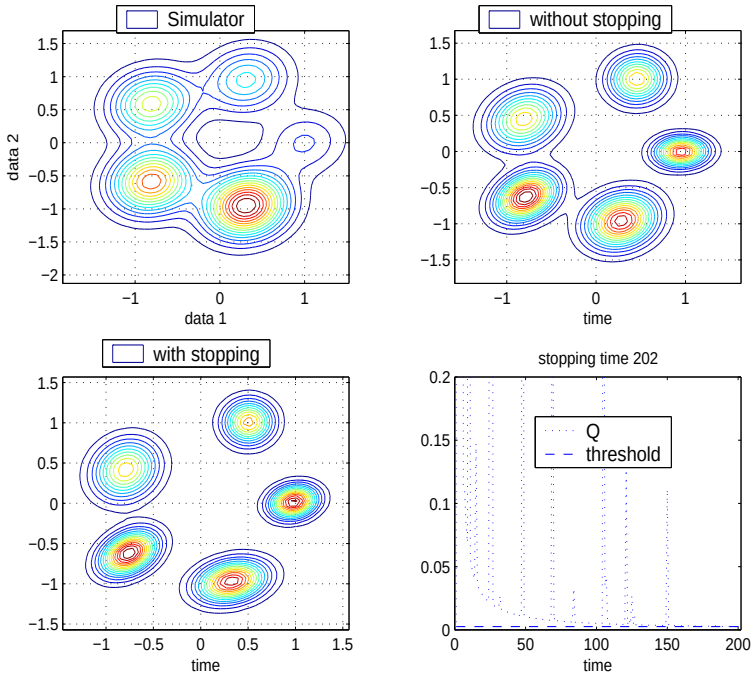


Figure 7: Comparison of estimation functions



run example view code z

Figure 8: Stopping rules in quasi-Bayes estimation

```
[Mix, mixlls, tstops, nrep] = mixestqbs(Mix0, frg, maxdat, niter, threshold)
```

Note: the input argument *threshold* should not be used.

The arguments that differ from *mixestqb*:

tstops	stopping times
nrep	resulting number of repetitions
maxdat	maximum length of data
threshold	threshold for stopping

Example: Data are generated, *mixesqb* and *mixestqbs* is called and results displayed in Fig. 11.

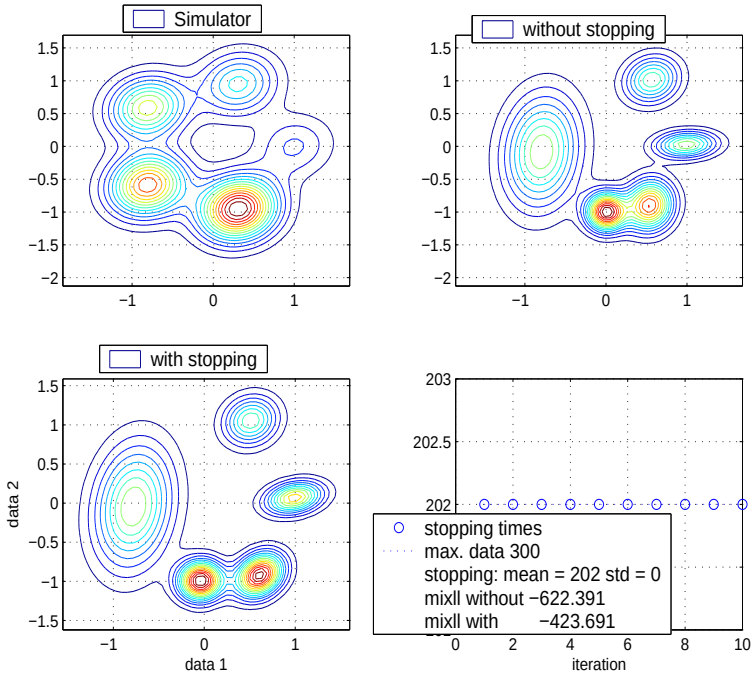
4 Stopping rules in initialization

The initialization can be done with stopping rules in estimation. It means, that the function *mixinit* accepts the code of estimation function $'P'$ and $'Q'$.

A case study follows. The iterative projection method is used for estimation. The results of each iteration is displayed in the Fig. **??**.

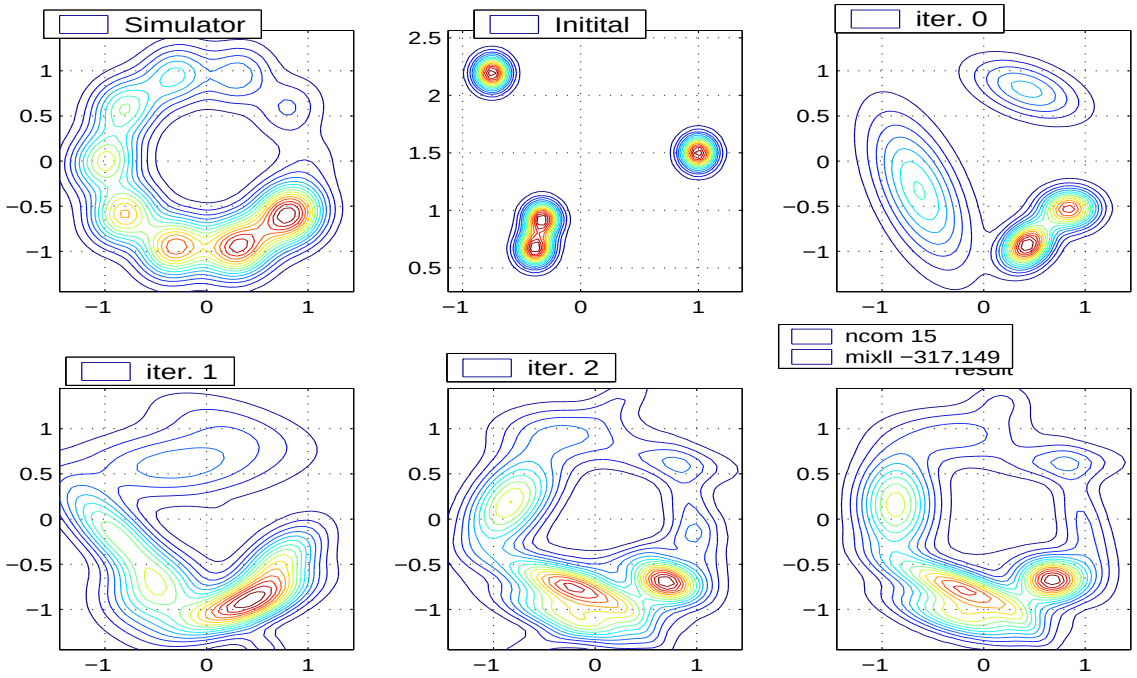
For comparison, the results obtained without stopping rules are in Fig. **??**.

Another benchmark example is initialization of a static mixture of many components. The results are in Fig. 12.



run example view code

Figure 9: Stopping rules in repetitive quasi-Bayes estimation



run example view code

Figure 10: Iterations of initialization with stopping

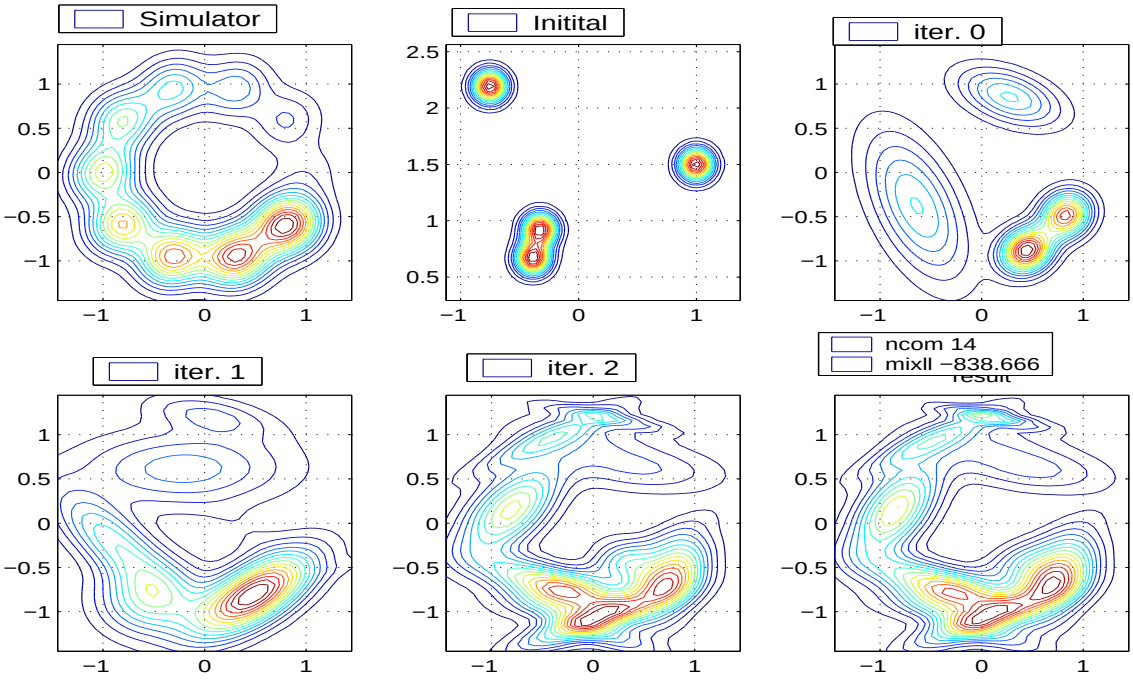


Figure 11: Iterations of initialization without stopping

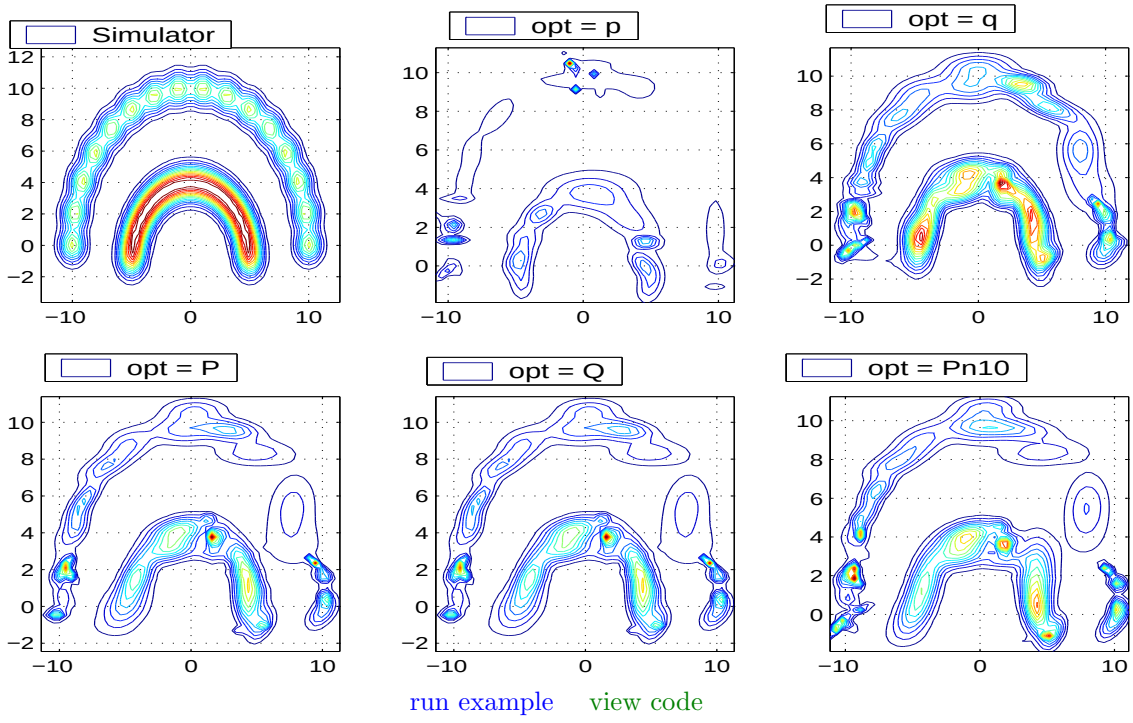


Figure 12: Iterations of initialization without stopping