Reference for cnt, reg

Generated by mfscanner

On June 25, 2010

Contents

cnt

average

```
average
         Use: average(est, param[, mode]);

 $Id: average.m,v 1.1 2006/04/04 17:05:39 oKp Exp $
```

ex_nealhint

```
 Script to illustrate the behavior of the REM algorithm on simulated data

 $Id: ex_nealhint.m,v 1.1 2006/07/12 16:51:31 cappe Exp $
```

ex_sim

```
Script to illustrate the behavior of the REM algorithm on simulated data

$Id: ex_sim.m,v 1.3 2006/04/06 08:36:52 oKp Exp $
```

plt_sim

```
Script to plot the results of ex_sim

$Id: plt_sim.m,v 1.1 2006/04/06 08:36:52 oKp Exp $
```

pm_nealhint

```
pm_nealhint Estimates the parameters of a Poisson mixture using the incremenatl EM algor
         Use: [wght,rate,logl] = pm_nealhint(count,wght_0,rate_0,Nit,alpha).
```

pm_rem

```
pm_rem   Estimates the parameters of a Poisson mixture using the REM algorithm.
         Use: [wght,rate,logl] = pm_rem(count,wght_0,rate_0,gamma,update)
         where wght and rate are the estimated model parameters, logl
         contains the log-likehood values for the successive
         iterations.

$Id: pm_rem.m,v 1.3 2006/04/06 08:40:10 oKp Exp $
```

pm_rem_step

```
pm_rem_step
         Elementary step of the REM algorithm.
         Use: [wght, rate, wght_stats, rate_stats, logl] = pm_rem_step ...
           (count, wght, rate,  wght_stats, rate_stats, gamma, dnorm, update);
         note: does not perform any checks.

$Id: pm_rem_step.m,v 1.2 2006/04/04 17:05:39 oKp Exp $
```

pm_titt

```
pm_titt  Estimates the parameters of a Poisson mixture using the recursive
         EM algorithm a la Titterington.
         Use: [wght,rate,logl] = pm_titt(count,wght_0,rate_0,gamma,update)
         where wght and rate are the estimated model parameters, logl
         contains the log-likehood values for the successive
         iterations.

$Id: pm_titt.m,v 1.2 2006/04/06 08:40:10 oKp Exp $
```

pm_titt_step

```
pm_titt_step
         Elementary step of the recursive EM algorithm a la Titterington.
         Use: [wght, rate, logl] = pm_rem_step ...
           (count, wght, rate, gamma, dnorm, update);
         note: does not perform any checks.

 $Id: pm_titt_step.m,v 1.1 2006/04/06 08:36:51 oKp Exp $
```

reg

ex_toy

```
 Script to run EM on the toy example taken from flexmix (1 and 6 used to produce figures
 the paper)

 $Id: ex_toy.m,v 1.6 2007/10/01 15:40:27 cappe Exp $
```

reg_batch

```
 reg_batch     The usual (batch) EM algorithm.
 Use: [w, beta, sigma2, logl] = reg_batch(Y, Z, w_0, beta_0, sigma2_0, nit) where
       Y: responses (1 x n)
       Z: covariates (d x n)
 and
       w: (1 x m x nit+1)
       beta: (d x m x nit+1)
       sigma2 (1 x m x nit+1)
 Note that the last iteration is here just to compute the likelihood.

 $Id: reg_batch.m,v 1.1 2007/09/04 16:37:10 cappe Exp $
```

reg_e

```
 reg_e         Computes the EM statistics for one data point.
 Use: [bS1, bS2, bS3, bS4, logl] = reg_e(Y, Z, w, beta, sigma2) where
       Y:      response (scalar)
       Z:      covariates (d x 1)
       w:      weights (1 x m)
       beta:   regression coeffs. (d x m)
       sigma2: variances (1 x m)
 and
       bS1 (1 x m)
       bS2 (d x m)
       bS3 (d x d x m)
       bS4 (1 x m)
```

```
$Id: reg_e.m,v 1.1 2007/09/04 16:37:10 cappe Exp $
```

reg_info

```
reg_info       Computes the observed information matrix for one data point (on
               regression parameter only)
Use: [inf1, inf2] = reg_info(Y, Z, w, beta, sigma2) where
      Y:      response (scalar)
      Z:      covariates (d x 1)
      w:      weights (1 x m)
      beta:   regression coeffs. (d x m)
      sigma2: variances (1 x m)

$Id: reg_info.m,v 1.1 2007/09/21 16:39:01 cappe Exp $
```

reg_m

```
reg_m         The M-step
Use: [w, beta, sigma2] = reg_m(bS1, bS2, bS3, bS4).

$Id: reg_m.m,v 1.1 2007/09/04 16:37:10 cappe Exp $
```

reg_online

```
reg_online     The on-line EM algorithm.
Use: [w, beta, sigma2, logl] = reg_batch(Y, Z, w_0, beta_0, sigma2_0, gam) where
      Y: responses (1 x n)
      Z: covariates (d x n)
      gam: step size (1 x n)
and
      w: (1 x m x n + 1)
      beta: (d x m x n + 1)
      sigma2 (1 x m x n + 1)

$Id: reg_online.m,v 1.3 2008/03/17 11:13:18 cappe Exp $
```