OpSiMore/FIEM

OpSiMore is a web repositery for the contributions of the OpSiMorE team.

Author: Gersende Fort

Affiliation: CNRS and Institut de Mathématiques de Toulouse, France

Created in: July 2020

FIEM : Fast Incremental Expectation Maximization

Here are the codes associated to the paper "Fast Incremental Expectation Maximization for non-convex finite-sum optimization: non asymptotic convergence bounds" available from this web page; author : G. Fort, P. Gach and E. Moulines

In this repository, the subdirectory *ToyExample* contains the MATLAB codes for the toy example (section 4); the subdirectory *MNIST* contains the MATLAB codes for the more challenging example using the MNIST data set.

How to use the matlab files available in the subdirectory *ToyExample* ?

- Statistical model : n observations $(Y_1, \ldots, Y_n)$. For the value $\theta \in \mathbb{R}^q$ of the parameter: conditionally to $(Z_1, \ldots, Z_n)$, the random variables Y_i are independent with distribution

  $$\mathcal{N}_y(AZ_i, I);$$

  the random variables $(Z_1, \ldots, Z_n)$ are independent with the same distribution

  $$\mathcal{N}_p(X\theta, I).$$

  The matrices X and A are known.

- Statistical analysis: the goal is the estimation of the parameter θ through the minimum of the criterion

  $$-\frac{1}{n}\log g(Y_1, \ldots, Y_n; \theta) + \frac{v}{2}\|\theta\|^2$$

  where $g(y_1, \cdots, y_n; \theta)$ denotes the distribution of the vector $(Y_1, \ldots, Y_n)$. In this toy example, the solution exists, is unique and is even explicit (see section 4.1).

- Description of the file *SampleData.m*
  - The variables :
    - *n*: the size of the data set
    - *dim_theta*: the size of the vector θ
    - *dim_Y*: the size of each vector Y_i
    - *dim_Z* (or *p* in the paper): the size of each hidden variable Z_i
    - *theta_true*: the value of the parameter used to generate the observations as described by the above statistical model. The entries of *theta_true* are sampled uniformly in [-5,5] and then 40% of these entries are set to zero (they are chosen at random).
    - The columns of the *dim_Y x dim_Z* matrix A are sampled from a stationary AR(1) with variance 1 and coefficient *rho = 0.8*
    - The columns of the *dim_Z x dim_theta* matrix X are sampled from a stationary AR(1) with variance 1 and coefficient *rho = 0.9*
  - The ouput of the function: the function creates a file *DataLong.mat* which contains the observations $Y_1, \ldots, Y_n$, the matrices X and A, and the parameter *theta_true*.

- Description of the file *FIEM_Gamma.m*
  - The call

    ```
    >> FIEM_Gamma
    ```

    will launch the estimation of the parameter θ by the optimized FIEM algorithm described in Section 2.3.4 of the paper. The user is invited to choose different design parameters
  - Choices by the user
    - During the run of *FIEM_Gamma* graphical controls can be displayed (see the description below). The user is invited to choose if it accepts or not the display.
    - The data are loaded directly in the code. See the line

      ```
      >> load Data.mat
      ```

    Data.mat contains the matrices A and X which define the statistical model; and the n observations stored in a *dim_Y x n* matrix called *Ymatrix*.

    - *upsilon*: the numerical value of the regularization parameter in the penalty term $v\|\theta\|^2/2$. The default value is 0.1
    - *NbrMC*: independent runs of FIEM can be launched through a single call to *FIEM_Gamma*; they will be run with the same values of the design parameters and will only differ by the random mecanism for the selection of the examples. The user is invited to indicate this number. The default value is 1e3.
    - *Kmax*: the total length of a FIEM path. The default value is *20 n* where *n* is the number of observations, i.e. 20 epochs.
    - The learning rate is constant over the iterations of the path: the two strategies provided in the paper (see Proposition 5 and Proposition 6) are possible, the user can choose between resp. *rate n^(2/3)* and *rate n^(1/2)*. Both of these strategies depend on two parameters *mu* and *lambda* and the user is invited to specify these values; the default ones are resp. 0.25 and 0.5. For any other strategies, the user can modify the value of the variable *gamma_gfm* in the definition of *gamma_grid* directly in the code.

    - The initial value $\hat{S}^0$ of the FIEM path can be specified: either it is chosen at random by sampling the entries as standardized independent Gaussian variables; or it is read in a file *InitStat.mat* - this file has to contain the variable *Sinit* of size *dim_theta x 1*.
    - Two sequences of length *Kmax* containing indices in the range $\{1, \ldots, n\}$ have to be selected: they indicate the examples used in the updating mecanism of the auxiliary variable $\tilde{S}^{k+1}$ and the ones used in the updating mecanism of the statistics $\hat{S}^{k+1}$. Here again, the user is invited to choose between (i) a random selection; (ii) a choice stored in the file *RandomIndex.mat* (which contains the *NbrMC x Kmax* matrix *RandomIndexImatrix*) and *RandomIndexFIEM.mat* (which contains the *NbrMC x Kmax* matrix *RandomIndexJmatrix*).
    - Finally, by default, the *optimized FIEM* is implemented (see section 2.3.4); the computation of the leverage coefficient λ_{k+1} is done through the call to the function *findlambda.m*. The user can choose other strategies by modifying directly in the code the value of the variable *Coeff*. For example, *Coeff = 1* corresponds to the original FIEM algorithm (see section 2.3.3), and *Coeff = 0* corresponds to Online EM (see section 2.3.1)

  - The outputs: a file *StoreCoeffopt.mat* containing

    - *StoreCoeff*: the *NbrMC x Kmax* values of the leverage coefficient λ_k
    - *FielH*: a *NbrMC x Kmax* matrix containing the squared norm of $H_{k+1} = (\hat{S}^{k+1} - \hat{S}^k)/\gamma_{k+1}$.
    - *ExpFieldH*: a *NbrMC x Kmax* matrix containing the squared norm of the mean field $h(\hat{S}^k)$.
    - *ErrorTheta*: a *NbrMC x GG* matrix containing the squared norm of the difference $\theta^{k+1} - \theta_{\text{opt}}$ where θ_{opt} is the unique optimum of the objective function, and is explicit in this toy example. This error is evaluated at *GG* time steps in the range $\{1, \ldots, Kmax\}$.

  - The displayed graphs:

    - Figure1 and Figure2: the squared norm of $\theta^{k+1} - \theta_{\text{opt}}$ along the path, with a zoom on the first iterations.
    - Figure3: the value of the leverage coefficient λ_k along a FIEM path.
    - Figure4: the squared norm of H_{k+1} along a FIEM path.
    - Figure5: the evolution of each components of the FIEM statistic $\hat{S}^k$ along the FIEM path.

  - The function *findlambda.m* is called by *FIEM_Gamma* and computes the optimal leverage coefficient λ_k - see section 2.3.4

  - The functions *findcstar_1.m* and *findcstar_2.m* are called by *FIEM_Gamma* and compute the constant learning rate γ_{k+1} resp. provided in Proposition 5 and Proposition 6.

How to use the matlab files available in the subdirectory *MNIST* ?

- The data set can be found on https://www.kaggle.com/avnishnish/mnist-original#mnist-original.mat, by downloading the file *mnist-original.mat*. It contains a *label* vector of length 7e4 and a *data* matrix of size 784 x 7e4. The first 6e4 columns are the "learning set" and the last 1e4 columns are the "test set". A parametric statistical model is defined and the parameter of the model is learnt by using examples from the "learning set"; before this estimation phase, the data are pre-processed as described in *readMNIST.m*

First, download the file "mnist-original.mat" in the same directory as the .m following files. Then the call

```
>> readMNIST
```

will create the file Data.mat. Then, the call

```
>> FIEM_MNIST
```

will run FIEM (which will call "Initialisation.m"). The file RunEstimationFIEM.mat is created (see below for a description od its content).

- Description of the file "readMNIST.m"

  - The first step is to extract the training examples : *data_train* contains the first n=6e4 columns of *data*, and *label_train* contains the first n=6e4 components of *labels*.
  - Step 2: 67 features among the 784 ones are constant over the n examples: they are removed from the set of features. Therefore, the matrix *X* of features is of size *n x d_init* with *d_init=717*.
  - The third step reduces the dimension by ACP: standardize each column of X (expectation zero, standard deviation 1) and compute the singular value decomposition of *X' X*. Finally, consider the projection of *X* on the principal components. This yields the *n x d_init* matrix *Xred* which is saved in the file *Data.mat*.
  - Displayed
    - fig 1. an image from the training set with label equal to 3
    - fig 2. the eignevalues of *X'X* (the last 5 ones are removed)
    - fig 3. Projection of the *n* examples on the space spanned by the first three principal component.
    - fig 4. Projection of the *n* examples on the space spanned by the first three principal components; the colors indicate the class.
    - fig 5. Projection of the *n* examples on the space spanned by the first two principal components; the colors indicate the class.
    - fig 6. Projection of the *n* examples on the space spanned by the first principal component; the colors indicate the class.
  - Output file: *Data.mat* contains the *n x d_init* matrix *Xred*, with *n=6e4* and *d_init=717*.

- Statistical model :

  - *n* observations (Y_1, ..., Y_n) taking values in $\mathbb{R}^{d_{init}}$ colected in a *d_init x n* matrix. Only *d < d_init* are selected.
  - The statistical problem at hand consists in fitting a Gaussian Mixture Model in dimension *d*, with *L* components, on the features.
  - Therefore, the parameter collects *L* weights (i.e. non negative real numbers with sum equal to one), *L* vectors of length *d* collecting the *L* expectations of the Gaussian distributions, and the covariance matrices. Here, is it assumed that all the components have the same *d x d* covariance matrix. In this example, the objective function to be minimized is the negated normalized log-likelihood

$$-\frac{1}{n}\sum_{i=1}^{n} \log p(Y_i, \theta)$$

where for any observation $Y \in \mathbb{R}^d$, we have

$$p(Y, \theta) = \sum_{k=1}^{L} \alpha_k \, \mathcal{N}_d(\mu_k, \Sigma)(Y) \qquad \theta = (\alpha_1, \ldots, \alpha_L, \mu_1, \ldots, \mu_L, \Sigma).$$

- Description of the file "FIEM_MNIST.m"
    - General variables to be fixed:
        - *RunInit*: a binary variable to indicate if the initial parameter is read in a file called * .mat* or has to be constructed. The default value is 1.
        - *DisplayPlot*: a binary variable to indicate of some graphical controls are displayed during the run or not. The default value is 1.
    - Variables:
        - *L* is the number of classes in the Gaussian Mixture Model. The default value is *L=12*
        - *kswitch* defines the number of epochs of Online-EM before switching to epochs of FIEM. The default value is *kswitch=6*.
        - *NbrEpoch* is related to the number of conditional expectation evaluations : this number of evaluations will be set to *NbrEpoch x n* for an OnlineEM epoch and to *NbrEpoch x 2n* for a FIEM epoch. The default value is *NbrEpoch = 53*.
        - *minibatch* defines the size of the mini-batches. The default value is *minibatch = 100*.
        - *NbrRun* is the number of independent runs of the algorithm. The default value is *NbrRun = 10*.
        - *vectpas* collects the learning rate; it may be modified at each epoch. Its default value is *5e-3*.
    - Displayed
        - fig 1. evolution of the estimated parameter sequence of weights (*L* curves); along the current run, vs the number of iterations
        - fig 2. evolution of the estimated parameter sequence of a vector of expectation - the one of the first component of the Gaussian mixture (*d* curves); along the current run, vs the number of iterations.
        - fig 3. evolution of the eigenvalues of the estimated sequence of the covariance matrix (*d* curves); along the current run vs the number of iterations.
        - fig 4. evolution of the log-likelihood; along the current run vs the number of epochs.
    - Output file "RunEstimationFIEM.mat"
        - *RunStoreWeight* is a *L x NbrRun* matrix which collects the estimated weights $\alpha_1, \dots, \alpha_L$ at the end of each *NbrRun* independent runs of FIEM.
        - *RunStoreMean* is a *d x L x NbrRun* tensor which collects the estimated *L* vectors $\mu_1, \dots, \mu_L$ at the end of each *NbrRun* independent runs of FIEM.
        - *RunStoreInvCov* is a *d x d x NbrRun* tensor which collects the estimated *d x d* covariance matrix at the end of each *NbrRun* independent runs of FIEM.
        - *RunLogLikelihood* is a *NbrRun x (1+NbrEpoch)* matrix which collects, for each *NbrRun* independent runs of FIEM, the value of the normalized log-likelihood at the end of each epoch (the first column is the value for the initial parameter).
        - *StoreWeight* is a *L x (1+NbrEpoch)* matrix which collects the estimated weights $\alpha_1, \dots, \alpha_L$ at the end of each epoch (the first column is the value for the initial parameter).
        - *StoreMean1* is a *d x (1+NbrEpoch)* matrix which collects the estimated first expectation μ_1 at the end of each epoch (the first column is the value for the initial parameter).
        - *StoreEigen* is a *d x (1+NbrEpoch)* matrix which collects the eigenvalues of the estimated inverse-covariance matrix at the end of each epoch (the first column is the value for the initial parameter).
        - *Sweight* is a *L x NbrIter* matrix which collects the estimated weights $\alpha_1, \dots, \alpha_L$ at the end of each iteration.
        - *Smu* is a *d x NbrIter* matrix which collects the estimated expectation μ_1 at the end of each iteration.
        - *Seigen* is a *d x NbrIter* matrix which collects the eigenvalues of the estimated inverse-covariance matrix at the end of each iteration.

For example, the specifications *kswitch = 6*, *minibatch = 100*, *NbrEpoch = 53* will implement *6 n / 100* iterations of Online EM and then *47 n / 100* iterations of FIEM. The total number of conditional expectations evaluations will be *6 n + 2 x 47 n* since each FIEM iteration calls *2 minibatch* conditional expectations evaluations.

- Description of the file "Initialisation.m" A method for defining an initial value for the set of parameters (the weights, the expectations and the covariance matrix of a Gaussian mixture model) is proposed. This function calls "SampleInvCov.m".
    - Input variables:
        - *X*: the *d x n* matrix of features, modeled as *n* points from a Gaussian Mixture Model in dimension *d*.
        - *L*: the number of components in the Gaussian Mixture Model
    - Output variables:
        - a *d x L* matrix collecting the *L* expectations of the Gaussian components in dimension *d*
        - a *d x (dL)* matrix concatening *L* inverse covariance matrices of size *d x d*