JavaScript is disabled on your browser.

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/org/opencv/ml/DTrees.html)
* [Next Class](http://docs.google.com/org/opencv/ml/KNearest.html)
* [Frames](http://docs.google.com/index.html?org/opencv/ml/EM.html)
* [No Frames](http://docs.google.com/EM.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* [Field](#3znysh7) |
* Constr |
* [Method](#tyjcwt)
* Detail:
* [Field](#2s8eyo1) |
* Constr |
* [Method](#3j2qqm3)

org.opencv.ml

## Class EM

* java.lang.Object
  + [org.opencv.core.Algorithm](http://docs.google.com/org/opencv/core/Algorithm.html)
    - [org.opencv.ml.StatModel](http://docs.google.com/org/opencv/ml/StatModel.html)
      * org.opencv.ml.EM
* public class EM  
  extends [StatModel](http://docs.google.com/org/opencv/ml/StatModel.html)  
  The class implements the Expectation Maximization algorithm. SEE: REF: ml\_intro\_em

### Field SummaryFields

| Modifier and Type | Field and Description |
| --- | --- |
| static int | [**COV\_MAT\_DEFAULT**](http://docs.google.com/org/opencv/ml/EM.html#COV_MAT_DEFAULT) |
| static int | [**COV\_MAT\_DIAGONAL**](http://docs.google.com/org/opencv/ml/EM.html#COV_MAT_DIAGONAL) |
| static int | [**COV\_MAT\_GENERIC**](http://docs.google.com/org/opencv/ml/EM.html#COV_MAT_GENERIC) |
| static int | [**COV\_MAT\_SPHERICAL**](http://docs.google.com/org/opencv/ml/EM.html#COV_MAT_SPHERICAL) |
| static int | [**DEFAULT\_MAX\_ITERS**](http://docs.google.com/org/opencv/ml/EM.html#DEFAULT_MAX_ITERS) |
| static int | [**DEFAULT\_NCLUSTERS**](http://docs.google.com/org/opencv/ml/EM.html#DEFAULT_NCLUSTERS) |
| static int | [**START\_AUTO\_STEP**](http://docs.google.com/org/opencv/ml/EM.html#START_AUTO_STEP) |
| static int | [**START\_E\_STEP**](http://docs.google.com/org/opencv/ml/EM.html#START_E_STEP) |
| static int | [**START\_M\_STEP**](http://docs.google.com/org/opencv/ml/EM.html#START_M_STEP) |

### Fields inherited from class org.opencv.ml.[**StatModel**](http://docs.google.com/org/opencv/ml/StatModel.html)[COMPRESSED\_INPUT](http://docs.google.com/org/opencv/ml/StatModel.html#COMPRESSED_INPUT), [PREPROCESSED\_INPUT](http://docs.google.com/org/opencv/ml/StatModel.html#PREPROCESSED_INPUT), [RAW\_OUTPUT](http://docs.google.com/org/opencv/ml/StatModel.html#RAW_OUTPUT), [UPDATE\_MODEL](http://docs.google.com/org/opencv/ml/StatModel.html#UPDATE_MODEL)

### Method SummaryMethods

| Modifier and Type | Method and Description |
| --- | --- |
| static [EM](http://docs.google.com/org/opencv/ml/EM.html) | [**\_\_fromPtr\_\_**](http://docs.google.com/org/opencv/ml/EM.html#__fromPtr__(long))(long addr) |
| static [EM](http://docs.google.com/org/opencv/ml/EM.html) | [**create**](http://docs.google.com/org/opencv/ml/EM.html#create())() Creates empty %EM model. |
| int | [**getClustersNumber**](http://docs.google.com/org/opencv/ml/EM.html#getClustersNumber())() SEE: setClustersNumber |
| int | [**getCovarianceMatrixType**](http://docs.google.com/org/opencv/ml/EM.html#getCovarianceMatrixType())() SEE: setCovarianceMatrixType |
| void | [**getCovs**](http://docs.google.com/org/opencv/ml/EM.html#getCovs(java.util.List))(java.util.List<[Mat](http://docs.google.com/org/opencv/core/Mat.html)> covs) Returns covariation matrices Returns vector of covariation matrices. |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getMeans**](http://docs.google.com/org/opencv/ml/EM.html#getMeans())() Returns the cluster centers (means of the Gaussian mixture) Returns matrix with the number of rows equal to the number of mixtures and number of columns equal to the space dimensionality. |
| [TermCriteria](http://docs.google.com/org/opencv/core/TermCriteria.html) | [**getTermCriteria**](http://docs.google.com/org/opencv/ml/EM.html#getTermCriteria())() SEE: setTermCriteria |
| [Mat](http://docs.google.com/org/opencv/core/Mat.html) | [**getWeights**](http://docs.google.com/org/opencv/ml/EM.html#getWeights())() Returns weights of the mixtures Returns vector with the number of elements equal to the number of mixtures. |
| static [EM](http://docs.google.com/org/opencv/ml/EM.html) | [**load**](http://docs.google.com/org/opencv/ml/EM.html#load(java.lang.String))(java.lang.String filepath) Loads and creates a serialized EM from a file Use EM::save to serialize and store an EM to disk. |
| static [EM](http://docs.google.com/org/opencv/ml/EM.html) | [**load**](http://docs.google.com/org/opencv/ml/EM.html#load(java.lang.String,%20java.lang.String))(java.lang.String filepath, java.lang.String nodeName) Loads and creates a serialized EM from a file Use EM::save to serialize and store an EM to disk. |
| float | [**predict**](http://docs.google.com/org/opencv/ml/EM.html#predict(org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples) Returns posterior probabilities for the provided samples |
| float | [**predict**](http://docs.google.com/org/opencv/ml/EM.html#predict(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) results) Returns posterior probabilities for the provided samples |
| float | [**predict**](http://docs.google.com/org/opencv/ml/EM.html#predict(org.opencv.core.Mat,%20org.opencv.core.Mat,%20int))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) results, int flags) Returns posterior probabilities for the provided samples |
| double[] | [**predict2**](http://docs.google.com/org/opencv/ml/EM.html#predict2(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) sample, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Returns a likelihood logarithm value and an index of the most probable mixture component for the given sample. |
| void | [**setClustersNumber**](http://docs.google.com/org/opencv/ml/EM.html#setClustersNumber(int))(int val) getClustersNumber SEE: getClustersNumber |
| void | [**setCovarianceMatrixType**](http://docs.google.com/org/opencv/ml/EM.html#setCovarianceMatrixType(int))(int val) getCovarianceMatrixType SEE: getCovarianceMatrixType |
| void | [**setTermCriteria**](http://docs.google.com/org/opencv/ml/EM.html#setTermCriteria(org.opencv.core.TermCriteria))([TermCriteria](http://docs.google.com/org/opencv/core/TermCriteria.html) val) getTermCriteria SEE: getTermCriteria |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainE**](http://docs.google.com/org/opencv/ml/EM.html#trainE(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainEM**](http://docs.google.com/org/opencv/ml/EM.html#trainEM(org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainEM**](http://docs.google.com/org/opencv/ml/EM.html#trainEM(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainEM**](http://docs.google.com/org/opencv/ml/EM.html#trainEM(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainEM**](http://docs.google.com/org/opencv/ml/EM.html#trainEM(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainM**](http://docs.google.com/org/opencv/ml/EM.html#trainM(org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainM**](http://docs.google.com/org/opencv/ml/EM.html#trainM(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainM**](http://docs.google.com/org/opencv/ml/EM.html#trainM(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. |
| boolean | [**trainM**](http://docs.google.com/org/opencv/ml/EM.html#trainM(org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat,%20org.opencv.core.Mat))([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. |

### Methods inherited from class org.opencv.ml.[**StatModel**](http://docs.google.com/org/opencv/ml/StatModel.html)[calcError](http://docs.google.com/org/opencv/ml/StatModel.html#calcError(org.opencv.ml.TrainData,%20boolean,%20org.opencv.core.Mat)), [empty](http://docs.google.com/org/opencv/ml/StatModel.html#empty()), [getVarCount](http://docs.google.com/org/opencv/ml/StatModel.html#getVarCount()), [isClassifier](http://docs.google.com/org/opencv/ml/StatModel.html#isClassifier()), [isTrained](http://docs.google.com/org/opencv/ml/StatModel.html#isTrained()), [train](http://docs.google.com/org/opencv/ml/StatModel.html#train(org.opencv.core.Mat,%20int,%20org.opencv.core.Mat)), [train](http://docs.google.com/org/opencv/ml/StatModel.html#train(org.opencv.ml.TrainData)), [train](http://docs.google.com/org/opencv/ml/StatModel.html#train(org.opencv.ml.TrainData,%20int))

### Methods inherited from class org.opencv.core.[**Algorithm**](http://docs.google.com/org/opencv/core/Algorithm.html)[clear](http://docs.google.com/org/opencv/core/Algorithm.html#clear()), [getDefaultName](http://docs.google.com/org/opencv/core/Algorithm.html#getDefaultName()), [getNativeObjAddr](http://docs.google.com/org/opencv/core/Algorithm.html#getNativeObjAddr()), [save](http://docs.google.com/org/opencv/core/Algorithm.html#save(java.lang.String))

### Methods inherited from class java.lang.Objectequals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

### Field Detail

#### COV\_MAT\_DEFAULT public static final int COV\_MAT\_DEFAULTSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.COV_MAT_DEFAULT)

#### COV\_MAT\_DIAGONAL public static final int COV\_MAT\_DIAGONALSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.COV_MAT_DIAGONAL)

#### COV\_MAT\_GENERIC public static final int COV\_MAT\_GENERICSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.COV_MAT_GENERIC)

#### COV\_MAT\_SPHERICAL public static final int COV\_MAT\_SPHERICALSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.COV_MAT_SPHERICAL)

#### DEFAULT\_MAX\_ITERS public static final int DEFAULT\_MAX\_ITERSSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.DEFAULT_MAX_ITERS)

#### DEFAULT\_NCLUSTERS public static final int DEFAULT\_NCLUSTERSSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.DEFAULT_NCLUSTERS)

#### START\_AUTO\_STEP public static final int START\_AUTO\_STEPSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.START_AUTO_STEP)

#### START\_E\_STEP public static final int START\_E\_STEPSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.START_E_STEP)

#### START\_M\_STEP public static final int START\_M\_STEPSee Also:[Constant Field Values](http://docs.google.com/constant-values.html#org.opencv.ml.EM.START_M_STEP)

### Method Detail

#### \_\_fromPtr\_\_ public static [EM](http://docs.google.com/org/opencv/ml/EM.html) \_\_fromPtr\_\_(long addr)

#### create public static [EM](http://docs.google.com/org/opencv/ml/EM.html) create() Creates empty %EM model. The model should be trained then using StatModel::train(traindata, flags) method. Alternatively, you can use one of the EM::train\\* methods or load it from file using Algorithm::load<EM>(filename).Returns:automatically generated

#### getClustersNumber public int getClustersNumber() SEE: setClustersNumberReturns:automatically generated

#### getCovarianceMatrixType public int getCovarianceMatrixType() SEE: setCovarianceMatrixTypeReturns:automatically generated

#### getCovs public void getCovs(java.util.List<[Mat](http://docs.google.com/org/opencv/core/Mat.html)> covs) Returns covariation matrices Returns vector of covariation matrices. Number of matrices is the number of gaussian mixtures, each matrix is a square floating-point matrix NxN, where N is the space dimensionality.Parameters:covs - automatically generated

#### getMeans public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getMeans() Returns the cluster centers (means of the Gaussian mixture) Returns matrix with the number of rows equal to the number of mixtures and number of columns equal to the space dimensionality.Returns:automatically generated

#### getTermCriteria public [TermCriteria](http://docs.google.com/org/opencv/core/TermCriteria.html) getTermCriteria() SEE: setTermCriteriaReturns:automatically generated

#### getWeights public [Mat](http://docs.google.com/org/opencv/core/Mat.html) getWeights() Returns weights of the mixtures Returns vector with the number of elements equal to the number of mixtures.Returns:automatically generated

#### load public static [EM](http://docs.google.com/org/opencv/ml/EM.html) load(java.lang.String filepath) Loads and creates a serialized EM from a file Use EM::save to serialize and store an EM to disk. Load the EM from this file again, by calling this function with the path to the file. Optionally specify the node for the file containing the classifierParameters:filepath - path to serialized EM Returns:automatically generated

#### load public static [EM](http://docs.google.com/org/opencv/ml/EM.html) load(java.lang.String filepath, java.lang.String nodeName) Loads and creates a serialized EM from a file Use EM::save to serialize and store an EM to disk. Load the EM from this file again, by calling this function with the path to the file. Optionally specify the node for the file containing the classifierParameters:filepath - path to serialized EMnodeName - name of node containing the classifier Returns:automatically generated

#### predict public float predict([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples) Returns posterior probabilities for the provided samples**Overrides:** [predict](http://docs.google.com/org/opencv/ml/StatModel.html#predict(org.opencv.core.Mat)) in class [StatModel](http://docs.google.com/org/opencv/ml/StatModel.html) Parameters:samples - The input samples, floating-point matrix posterior probabilities for each sample from the input Returns:automatically generated

#### predict public float predict([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) results) Returns posterior probabilities for the provided samples**Overrides:** [predict](http://docs.google.com/org/opencv/ml/StatModel.html#predict(org.opencv.core.Mat,%20org.opencv.core.Mat)) in class [StatModel](http://docs.google.com/org/opencv/ml/StatModel.html) Parameters:samples - The input samples, floating-point matrixresults - The optional output \( nSamples \times nClusters\) matrix of results. It contains posterior probabilities for each sample from the input Returns:automatically generated

#### predict public float predict([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) results, int flags) Returns posterior probabilities for the provided samples**Overrides:** [predict](http://docs.google.com/org/opencv/ml/StatModel.html#predict(org.opencv.core.Mat,%20org.opencv.core.Mat,%20int)) in class [StatModel](http://docs.google.com/org/opencv/ml/StatModel.html) Parameters:samples - The input samples, floating-point matrixresults - The optional output \( nSamples \times nClusters\) matrix of results. It contains posterior probabilities for each sample from the inputflags - This parameter will be ignored Returns:automatically generated

#### predict2 public double[] predict2([Mat](http://docs.google.com/org/opencv/core/Mat.html) sample, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Returns a likelihood logarithm value and an index of the most probable mixture component for the given sample.Parameters:sample - A sample for classification. It should be a one-channel matrix of \(1 \times dims\) or \(dims \times 1\) size.probs - Optional output matrix that contains posterior probabilities of each component given the sample. It has \(1 \times nclusters\) size and CV\_64FC1 type. The method returns a two-element double vector. Zero element is a likelihood logarithm value for the sample. First element is an index of the most probable mixture component for the given sample. Returns:automatically generated

#### setClustersNumber public void setClustersNumber(int val) getClustersNumber SEE: getClustersNumberParameters:val - automatically generated

#### setCovarianceMatrixType public void setCovarianceMatrixType(int val) getCovarianceMatrixType SEE: getCovarianceMatrixTypeParameters:val - automatically generated

#### setTermCriteria public void setTermCriteria([TermCriteria](http://docs.google.com/org/opencv/core/TermCriteria.html) val) getTermCriteria SEE: getTermCriteriaParameters:val - automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing. covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing. floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size. each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.covs0 - The vector of initial covariance matrices \(S\_k\) of mixture components. Each of covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing. floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size. each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.covs0 - The vector of initial covariance matrices \(S\_k\) of mixture components. Each of covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing.weights0 - Initial weights \(\pi\_k\) of mixture components. It should be a one-channel floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size. each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.covs0 - The vector of initial covariance matrices \(S\_k\) of mixture components. Each of covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing.weights0 - Initial weights \(\pi\_k\) of mixture components. It should be a one-channel floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.covs0 - The vector of initial covariance matrices \(S\_k\) of mixture components. Each of covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing.weights0 - Initial weights \(\pi\_k\) of mixture components. It should be a one-channel floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainE public boolean trainE([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) means0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) covs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) weights0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. You need to provide initial means \(a\_k\) of mixture components. Optionally you can pass initial weights \(\pi\_k\) and covariance matrices \(S\_k\) of mixture components.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.means0 - Initial means \(a\_k\) of mixture components. It is a one-channel matrix of \(nclusters \times dims\) size. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.covs0 - The vector of initial covariance matrices \(S\_k\) of mixture components. Each of covariance matrices is a one-channel matrix of \(dims \times dims\) size. If the matrices do not have CV\_64F type they will be converted to the inner matrices of such type for the further computing.weights0 - Initial weights \(\pi\_k\) of mixture components. It should be a one-channel floating-point matrix with \(1 \times nclusters\) or \(nclusters \times 1\) size.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type.probs - The optional output matrix that contains posterior probabilities of each Gaussian mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainEM public boolean trainEM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. Initial values of the model parameters will be estimated by the k-means algorithm. Unlike many of the ML models, %EM is an unsupervised learning algorithm and it does not take responses (class labels or function values) as input. Instead, it computes the \*Maximum Likelihood Estimate\* of the Gaussian mixture parameters from an input sample set, stores all the parameters inside the structure: \(p\_{i,k}\) in probs, \(a\_k\) in means , \(S\_k\) in covs[k], \(\pi\_k\) in weights , and optionally computes the output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). The trained model can be used further for prediction, just like any other classifier. The trained model is similar to the NormalBayesClassifier.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing. each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainEM public boolean trainEM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. Initial values of the model parameters will be estimated by the k-means algorithm. Unlike many of the ML models, %EM is an unsupervised learning algorithm and it does not take responses (class labels or function values) as input. Instead, it computes the \*Maximum Likelihood Estimate\* of the Gaussian mixture parameters from an input sample set, stores all the parameters inside the structure: \(p\_{i,k}\) in probs, \(a\_k\) in means , \(S\_k\) in covs[k], \(\pi\_k\) in weights , and optionally computes the output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). The trained model can be used further for prediction, just like any other classifier. The trained model is similar to the NormalBayesClassifier.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainEM public boolean trainEM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. Initial values of the model parameters will be estimated by the k-means algorithm. Unlike many of the ML models, %EM is an unsupervised learning algorithm and it does not take responses (class labels or function values) as input. Instead, it computes the \*Maximum Likelihood Estimate\* of the Gaussian mixture parameters from an input sample set, stores all the parameters inside the structure: \(p\_{i,k}\) in probs, \(a\_k\) in means , \(S\_k\) in covs[k], \(\pi\_k\) in weights , and optionally computes the output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). The trained model can be used further for prediction, just like any other classifier. The trained model is similar to the NormalBayesClassifier.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainEM public boolean trainEM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Expectation step. Initial values of the model parameters will be estimated by the k-means algorithm. Unlike many of the ML models, %EM is an unsupervised learning algorithm and it does not take responses (class labels or function values) as input. Instead, it computes the \*Maximum Likelihood Estimate\* of the Gaussian mixture parameters from an input sample set, stores all the parameters inside the structure: \(p\_{i,k}\) in probs, \(a\_k\) in means , \(S\_k\) in covs[k], \(\pi\_k\) in weights , and optionally computes the output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). The trained model can be used further for prediction, just like any other classifier. The trained model is similar to the NormalBayesClassifier.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.logLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type.probs - The optional output matrix that contains posterior probabilities of each Gaussian mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainM public boolean trainM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Maximization step. You need to provide initial probabilities \(p\_{i,k}\) to use this option.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.probs0 - the probabilities each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainM public boolean trainM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Maximization step. You need to provide initial probabilities \(p\_{i,k}\) to use this option.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.probs0 - the probabilitieslogLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type. \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainM public boolean trainM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Maximization step. You need to provide initial probabilities \(p\_{i,k}\) to use this option.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.probs0 - the probabilitieslogLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type. mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

#### trainM public boolean trainM([Mat](http://docs.google.com/org/opencv/core/Mat.html) samples, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs0, [Mat](http://docs.google.com/org/opencv/core/Mat.html) logLikelihoods, [Mat](http://docs.google.com/org/opencv/core/Mat.html) labels, [Mat](http://docs.google.com/org/opencv/core/Mat.html) probs) Estimate the Gaussian mixture parameters from a samples set. This variation starts with Maximization step. You need to provide initial probabilities \(p\_{i,k}\) to use this option.Parameters:samples - Samples from which the Gaussian mixture model will be estimated. It should be a one-channel matrix, each row of which is a sample. If the matrix does not have CV\_64F type it will be converted to the inner matrix of such type for the further computing.probs0 - the probabilitieslogLikelihoods - The optional output matrix that contains a likelihood logarithm value for each sample. It has \(nsamples \times 1\) size and CV\_64FC1 type.labels - The optional output "class label" for each sample: \(\texttt{labels}\_i=\texttt{arg max}\_k(p\_{i,k}), i=1..N\) (indices of the most probable mixture component for each sample). It has \(nsamples \times 1\) size and CV\_32SC1 type.probs - The optional output matrix that contains posterior probabilities of each Gaussian mixture component given the each sample. It has \(nsamples \times nclusters\) size and CV\_64FC1 type. Returns:automatically generated

* [Overview](http://docs.google.com/overview-summary.html)
* [Package](http://docs.google.com/package-summary.html)
* Class
* [Tree](http://docs.google.com/package-tree.html)
* [Index](http://docs.google.com/index-all.html)
* [Help](http://docs.google.com/help-doc.html)
* [Prev Class](http://docs.google.com/org/opencv/ml/DTrees.html)
* [Next Class](http://docs.google.com/org/opencv/ml/KNearest.html)
* [Frames](http://docs.google.com/index.html?org/opencv/ml/EM.html)
* [No Frames](http://docs.google.com/EM.html)
* [All Classes](http://docs.google.com/allclasses-noframe.html)
* Summary:
* Nested |
* [Field](#3znysh7) |
* Constr |
* [Method](#tyjcwt)
* Detail:
* [Field](#2s8eyo1) |
* Constr |
* [Method](#3j2qqm3)

Generated on 2021-04-02 03:15:03 / OpenCV 3.4.14