Data handling

Alphabet: A set of symbols
dna new DiscreteAlphabet(caseIn敏感的, alphabet): Create an arbitrary discrete alphabet
dna new ContinuousAlphabet(min, max): Create a continuous alphabet between min and max
DNAAlphabetContainer.SINGLETON: Singleton instance of a DNA-alphabet

Sequence seq: Representing a biological sequence
Sequence.create(alphabets, string): Create a sequence from a string
seq.getLength(): Obtain the length of a sequence
seq.discreteVal(pos): Obtain the discrete value at a position (counting from 0) of a sequence
seq.continuousVal(pos): Obtain the continuous value at a position (counting from 0) of a sequence

DataSet data: A set of sequences using the same AlphabetContainer
new DataSet(annotation, sequences): Create a data set from sequences
new DNADataSet(filename): Create a data set of DNA sequences from a FastA file
data.getNumOfElements(): Obtain the number of sequences in a data set
data.getElementAt(index): Obtain the sequence at index from a data set
data.getInfixDataSet(start, length): Get a data set containing all infixes of a given length starting at a given position of all sequences in the current data set

Statistical models

StatisticalModel statMod: Interface for all statistical models
TrainableStatisticalModel trainSM: Interface for statistical models that can be trained from a single data set
DifferentiableStatisticalModel diffSM: Interface for statistical models that can be trained using gradient-based methods

TrainableStatisticalModelFactory: Factory for standard implementations of TrainableStatisticalModels
TrainableStatisticalModelFactory.createPWM(alphabets, length, ess, order): Create a PWM model of a given length
TrainableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
TrainableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order): Create a homogeneous Markov model of a given order
TrainableStatisticalModelFactory.createMixtureModel(alphabets, ess, order): Create a mixture model from TrainableStatisticalModels

DifferentiableStatisticalModelFactory: Factory for standard implementations of DifferentiableStatisticalModels
DifferentiableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length
DifferentiableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
DifferentiableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order, priorLength): Create a homogeneous Markov model of a given order
DifferentiableStatisticalModelFactory.createMixtureModel(models): Create a mixture model from DifferentiableStatisticalModels

HMMFactory: Factory for standard implementations of hidden Markov models

Classifiers

AbstractClassifier classify: Abstract class of a classifier
new TrainSMBasedClassifier(models): Create a classifier from TrainableStatisticalModels that is learned by ML or MAP
new MSPClassifier(params, prior, models): Create a classifier from DifferentiableStatisticalModels that is learned by MCL or MSP
new GenDisMixClassifier(params, prior, learnPrinc, models): Classifier that learns DifferentiableStatisticalModels using a unified learning principle

ClassifierFactory: Factory for standard classifiers
classif.train(dataSets): Train a classifier from training data sets
classif.classify(sequence): Classify a sequence
classif.evaluate(performanceMeasures, exc, dataSet): Evaluate performance measures for a given classifier on test data sets

AbstractPerformanceMeasure: Abstract class of all performance measures
new NumericalPerformanceMeasureParameterSet(): Create a set of scalar standard performance measures that are applicable to two-class problems (binary classification)
new PerformanceMeasureParameterSet(measures): Create a set of scalar standard performance measures for binary classification problems that can immediately be used

Utilities

Storable: Interface of objects that can be stored to XML
XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuffer with given tags
XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal pairwise alignments of sequences
align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array
ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox: Compute several statistics from double arrays
Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs
Normalisation.sumNormalisation(double): Normalize a given array to probabilities

Jstacs reference card

StatisticalModelFactory: A set of symbols
TrainableStatisticalModelFactory: Create a homogeneous Markov model of a given order
DifferentiableStatisticalModelFactory: Create an inhomogeneous Markov model of a given length
DifferentiableStatisticalModelFactory: Create a PWM
Data handling

Alphabet: A set of symbols
new DiscreteAlphabet(caseInsensitive, alphabet): Create an arbitrary discrete alphabet
new ContinuousAlphabet(min, max): Create a continuous alphabet between min and max
DNAAlphabetContainer.SINGLETON: Singleton instance of a DNA-alphabet

Sequence seq: Representing a biological sequence
Sequence.create(alphabets, string): Create a sequence from a string
seq.getLength(): Obtain the length of a sequence
seq.discreteVal(pos): Obtain the discrete value at a position (counting from 0) of a sequence
seq.continuousVal(pos): Obtain the continuous value at a position (counting from 0) of a sequence

DataSet data: A set of sequences using the same AlphabetContainer
new DataSet(annotation, sequences): Create a data set from sequences
new DNADataSet(filename): Create a data set of DNA sequences from a FastA file
data.getNumOfElements(): Obtain the number of sequences in a data set
data.getElementAt(index): Obtain the sequence at index from a data set
data.getInfixDataSet(start, length): Get a data set containing all infixes of a given length starting at a given position of all sequences in the current data set

Statistical models

StatisticalModel statMod: Interface for all statistical models
TrainableStatisticalModel trainSM: Interface for statistical models that can be trained from a single data set
DifferentiableStatisticalModel diffSM: Interface for statistical models that can be trained using gradient-based methods

TrainableStatisticalModelFactory: Factory for standard implementations of TrainableStatisticalModels
TrainableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length
TrainableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
TrainableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order): Create a homogeneous Markov model of a given order
TrainableStatisticalModelFactory.createMixtureModel(models): Create a mixture model from TrainableStatisticalModels

DifferentiableStatisticalModelFactory: Factory for standard implementations of DifferentiableStatisticalModels
DifferentiableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length
DifferentiableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
DifferentiableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order, priorLength): Create a homogeneous Markov model of a given order
DifferentiableStatisticalModelFactory.createMixtureModel(models): Create a mixture model from DifferentiableStatisticalModels

HMMFactory: Factory for standard implementations of hidden Markov models

Classifiers

AbstractClassifier classify: Abstract class of a classifier
new TrainSMBasedClassifier(models): Create a classifier from TrainableStatisticalModels that is learned by ML or MAP
new MSPClassifier(params, prior, models): Create a classifier from DifferentiableStatisticalModels that is learned by MCL or MSP
new GenDisMixClassifier(params, prior, learnPrinc, models): Classifier that learns DifferentiableStatisticalModels using a unified learning principle

ClassifierFactory: Factory for standard classifiers
classif.train(dataSets): Train a classifier from training data sets
classif.classify(sequence): Classify a sequence
classif.evaluate(performanceMeasures, exc, dataSet): Evaluate performance measures for a given classifier on test data sets

AbstractPerformanceMeasure: Abstract class of all performance measures
new NumericalPerformanceMeasureParameterSet(): Create a set of scalar standard performance measures that are applicable to two-class problems (binary classification)
new PerformanceMeasureParameterSet(measures): Create a set of scalar standard performance measures for binary classification problems that can immediately be used

Utilities

Storable: Interface of objects that can be stored to XML
XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuffer with given tags
XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal pairwise alignments of sequences
align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array
ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox: Compute several statistics from double arrays
Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs
Normalisation.sumNormalisation(double): Normalize a given array to probabilities

Jstacs reference card

StatisticalModel statMod: Interface for all statistical models
TrainableStatisticalModel trainSM: Interface for statistical models that can be trained from a single data set
DifferentiableStatisticalModel diffSM: Interface for statistical models that can be trained using gradient-based methods

TrainableStatisticalModelFactory: Factory for standard implementations of TrainableStatisticalModels
TrainableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length
TrainableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
TrainableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order): Create a homogeneous Markov model of a given order
TrainableStatisticalModelFactory.createMixtureModel(models): Create a mixture model from TrainableStatisticalModels

DifferentiableStatisticalModelFactory: Factory for standard implementations of DifferentiableStatisticalModels
DifferentiableStatisticalModelFactory.createPWM(alphabets, length, ess): Create a PWM model of a given length
DifferentiableStatisticalModelFactory.createInhomogeneousMarkovModel(alphabets, length, ess, order): Create an inhomogeneous Markov model of a given length and order
DifferentiableStatisticalModelFactory.createHomogeneousMarkovModel(alphabets, ess, order, priorLength): Create a homogeneous Markov model of a given order
DifferentiableStatisticalModelFactory.createMixtureModel(models): Create a mixture model from DifferentiableStatisticalModels

HMMFactory: Factory for standard implementations of hidden Markov models

Classifiers

AbstractClassifier classify: Abstract class of a classifier
new TrainSMBasedClassifier(models): Create a classifier from TrainableStatisticalModels that is learned by ML or MAP
new MSPClassifier(params, prior, models): Create a classifier from DifferentiableStatisticalModels that is learned by MCL or MSP
new GenDisMixClassifier(params, prior, learnPrinc, models): Classifier that learns DifferentiableStatisticalModels using a unified learning principle

ClassifierFactory: Factory for standard classifiers
classif.train(dataSets): Train a classifier from training data sets
classif.classify(sequence): Classify a sequence
classif.evaluate(performanceMeasures, exc, dataSet): Evaluate performance measures for a given classifier on test data sets

AbstractPerformanceMeasure: Abstract class of all performance measures
new NumericalPerformanceMeasureParameterSet(): Create a set of scalar standard performance measures that are applicable to two-class problems (binary classification)
new PerformanceMeasureParameterSet(measures): Create a set of scalar standard performance measures for binary classification problems that can immediately be used

Utilities

Storable: Interface of objects that can be stored to XML
XMLParser.appendObjectWithTags(buffer, storable, tag): Append storable object to StringBuffer with given tags
XMLParser.extractObjectForTags(buffer, tag): Extract storable object within tags from StringBuffer

Alignment align: Class for optimal pairwise alignments of sequences
align.getAlignment(seq1, seq2): Align two sequences

ArrayHandler.clone(array): Deep clone a multi-dimensional array
ArrayHandler.createArrayOf(template, num): Create an array containing num clones of a template

ToolBox: Compute several statistics from double arrays
Normalisation.getLogSum(doubles): Compute the logarithm of a sum of values given as their logs
Normalisation.sumNormalisation(double): Normalize a given array to probabilities