VisCanvas UML Description

Variable

|  |  |
| --- | --- |
| Name: | Type: |
| Description: | |

Function:

|  |  |
| --- | --- |
| Name: | Params: |
| Return type: | |
| Description: | |

Table 1. Structure of whole dataset which contains m nD points

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Dimension 1 | Dimension 2 | ... | Dimension n | Class |
| nD point/set/case 1 | DataNode (1,1) | DataNode (2,1) | ... | DataNode (n,1) | K |
| nD point/set/case 2 | DataNode (1,2) | DataNode (2,2) | ... | DataNode (n,2) | L |
| ... | ... | ... | ... | ... | ... |
| nD point/set/case m | DataNode (1,m) | DataNode (2,m) | ... | DataNode (n,m) | K |

DataNode

This is the data structure designed to contain a value of a single attribute of a single nD point(case)

Table 2. Highlights information stored in DataNode Class

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Dimension 1 | Dimension 2 | ... | Dimension n | Class |
| nD point/set/case 1 | DataNode (1,1) | DataNode (2,1) | ... | DataNode (n,1) | K |
| nD point/set/case 2 | DataNode (1,2) | DataNode (2,2) | ... | DataNode (n,2) | L |
| ... | ... | ... | ... | ... | ... |
| nD point/set/case m | DataNode (1,m) | DataNode (2,m) | ... | DataNode (n,m) | K |

Variables/Attributes

|  |  |
| --- | --- |
| Name: originalData | Type: double |
| Description: Contains the datum as read from the file. | |
| Name: currentData | Type: double |
| Description: Contains the current datum. | |

Functions/Methods

|  |  |
| --- | --- |
| Name: DataNode | Params: |
| Return type: | |
| Description: Initializes an empty DataNode | |
| Name: DataNode | Params: double data |
| Return type: | |
| Description: Initializes a DataNode and sets originalData and currentData to data. | |
| Name: getData | Params: |
| Return type: double | |
| Description: Returns the value of currentData. | |
| Name: getOriginalData | Params: |
| Return type: double | |
| Description: Returns the value of originalData. | |
| Name: setData | Params: double newData |
| Return type: | |
| Description: Sets originalData to passed newData. | |
| Name: setDataCurrent | Params: double newData |
| Return type: | |
| Description: Sets currentData to passed newData. | |
| Name: resetData | Params: |
| Return type: | |
| Description: Sets currentData to originalData. | |
| Name: multiplyData | Params: double multiplier |
| Return type: | |
| Description: Multiplies currentData by multiplier and saves in currentData. | |
| Name: divideData | Params: double divisor |
| Return type: | |
| Description: Divides currentData by divisor and saves in currentData. | |
| Name: addToData | Params: double addend |
| Return type: | |
| Description: Adds addend to currentData and saves in currentData. | |

Dimension  
A data structure designed to contain the data of a dimension. A list of Dimensions is the main data structure for the dataset.

Variables/Attributes

|  |  |
| --- | --- |
| Name: data | Type: vector<DataNode\*> |
| Description: Contains the list of DataNodes that are within the dimension. | |
| Name: shiftAmount | Type: double = 0.0 |
| Description: This value is added to the data upon retrieval. | |
| Name: inverted | Type: bool = true |
| Description: Determines whether or not the data are inverted upon retrieval. | |
| Name: isDrawn | Type: bool = false |
| Description: Determines whether the dimension is drawn or not | |
| Name: originalIndex | Type: int |
| Description: Contains the index that the dimension had in the original data. | |
| Name: useArtificialCalibration | Type: bool = false |
| Description: Determines whether or not artificial calibration should be used upon retrieval. | |
| Name: artificialMaximum | Type: double = 1.0 |
| Description: Contains the maximum value that is used to artificially calibrate the data upon retrieval | |
| Name: artificialMinimum | Type: double = 0.0 |
| Description: Contains the minimum value that is used to artificially calibrate the data upon retrieval | |
| Name: name | Type: string |
| Description: The name for the dimension | |

Functions/Methods

|  |  |
| --- | --- |
| Name: Dimension | Params: |
| Return type: | |
| Description: Initializes a dimension of size 0 at index 0. | |
| Name: Dimension | Params: int index, int size |
| Return type: | |
| Description: Creates a dimension of the passed size(size) at the passed index(index). | |
| Name: getOriginalIndex | Params: |
| Return type: int | |
| Description: Returns the value of originalIndex. | |
| Name: setOriginalIndex | Params: int newIndex |
| Return type: int | |
| Description: Sets originalIndex to newIndex and returns the old value of originalIndex. | |
| Name: calibrateData | Params: |
| Return type: | |
| Description: Calibrates the data to [0, 1]. | |
| Name: getData | Params: int dataIndex |
| Return type: double | |
| Description: Returns the datum at the passed index(dataIndex). | |
| Name: getCalibratedData | Params: int dataIndex |
| Return type: double | |
| Description: Returns the calibrated datum at the passed index(dataIndex). This does not return the inverted or shifted datum. | |
| Name: getOriginalData | Params: int dataIndex |
| Return type: double | |
| Description: Returns the original data at the passed index. | |
| Name: getName | Params: |
| Return type: string\* | |
| Description: Returns the value of name. | |
| Name: setName | Params: string\* newName |
| Return type: | |
| Description: Sets name to the value of newName. | |
| Name: setData | Params: int dataIndex, double newData |
| Return type: | |
| Description: Sets the value of data for the DataNode to the passed new data(newData) at the passed index(dataIndex). | |
| Name: deleteData | Params: int dataIndex |
| Return type: | |
| Description: Deletes the DataNode at the passed index(dataIndex). | |
| Name: multiplyData | Params: double multiplier |
| Return type: | |
| Description: Mulitplies all data in the dimension by the multiplier. | |
| Name: divideData | Params: double divisor |
| Return type: | |
| Description: Divides all data in the dimension by the divisor. | |
| Name: addToData | Params: double addend |
| Return type: | |
| Description: Adds addend to all data in the dimension. | |
| Name: shiftDataBy | Params: double modToShiftAmount |
| Return type: | |
| Description: Adjusts the shiftAmount variable in the dimension via addition. | |
| Name: getShift | Params: |
| Return type: double | |
| Description: Returns the amount by which the data in the dimension is shifted. | |
| Name: isInverted | Params: |
| Return type: bool | |
| Description: Returns whether the data is inverted or not. | |
| Name: invert | Params: |
| Return type: | |
| Description: Inverts data by updating the inverted variable. | |
| Name: size | Params: |
| Return type: int | |
| Description: Returns the number of sets(DataNodes) in the dimension. (A.K.A length of the data vector) | |
| Name: isArtificiallyCalibrated | Params: |
| Return type: bool | |
| Description: Returns the value of the useArtificialCalibration variable. | |
| Name: clearArtificiallyCalibration | Params: |
| Return type: | |
| Description: Sets useArtificialCalibration to false. | |
| Name: setCalibrationBounds | Params: double newMaximum, double newMinimum |
| Return type: | |
| Description: Updates the values of artificialMaximum and artificialMinimum and sets useArtificalCalibration to true. | |
| Name: getArtificialMaximum | Params: |
| Return type: double | |
| Description: Returns the value of artificialMaximum. | |
| Name: getArtificialMinimum | Params: |
| Return type: double | |
| Description: Returns the value of artificialMinimum. | |
| Name: getMaximum | Params: |
| Return type: double | |
| Description: Calculates and returns the maximum value in the dimension. | |
| Name: getMinimum | Params: |
| Return type: double | |
| Description: Calculates and returns the minimum *non-empty* value in the dimension. | |
| Name: getAllData | Params: |
| Return type: vector<DataNode\*> | |
| Description: Returns the value of the data variable. | |
| Name: isVisible | Params: |
| Return type: bool | |
| Description: Returns the value of isDrawn. | |
| Name: setVisibility | Params: bool isDrawn |
| Return type: | |
| Description: Updates the dimension’s isDrawn variable to the new isDrawn. | |

SetCluster

This is the data structure designed to contain a set of nD points defined by limits in each dimension. These are known as hyper-blocks (HB).

Variables/attributes:

|  |  |
| --- | --- |
| Name: color | Type: ColorCustom |
| Description: Contains the color that is used to draw the HB | |
| Name: setsInCluster | Type: vector<int> |
| Description: Contains the indices for the sets that fall within the cluster. (List of nD points) | |
| Name: minimumValues | Type: vector<double> |
| Description: List of minimum values for all dimensions | |
| Name: maximumValues | Type: vector<double> |
| Description: List of maximum values for all dimensions | |
| Name: meanValues | Type: vector<double> |
| Description: List of mean values for all dimensions | |
| Name: medianValues | Type: vector<double> |
| Description: List of median values for all dimensions | |
| Name: size | Type: int |
| Description: Number of nD points within the HB | |
| Name: radius | Type: double |
| Description: Radius used for creation of HB | |
| Name: originalSet | Type: int |
| Description: Contains the index of the nD point used as a seed to create the HB | |
| Name: useMean | Type: bool |
| Description: Determines whether to use dimension means or medians for center line in HB | |
| Name: displayed | Type: bool |
| Description: Determines whether to draw HB or not | |
| Name: name | Type: string |
| Description: Contains the name of HB | |

Functions/methods

|  |  |
| --- | --- |
| Name: SetCluster | Params: |
| Return type: | |
| Description: Constructor for creation of HB | |
| Name: SetCluster | Params: ColorCustom &clusterColor |
| Return type: | |
| Description: Constructor for creation of HB with a specified color | |
| Name: SetCluster | Params: ColorCustom &clusterColor, vector<int\*> newSetsInCluster |
| Return type: | |
| Description: Constructor for creation of HB with a specified color and preexisting HB data | |
| Name: SetCluster | Params: ColorCustom &clusterColor, vector<int\*> newSetsInCluster, vector<Dimension\*>\* dimensionsToCalculateWith |
| Return type: | |
| Description: Constructor for creation of HB with a specified color, preexisting HB data and the dimensions used to create HB data | |
| Name: getMinimumValue | Params: Dimension\* dimension, vector<int\*> setIndexes |
| Return type: double | |
| Description: Static function: Calculates the minimum value in the sets at the passed indices for the indicated dimension | |
| Name: getMaximumValue | Params: Dimension\* dimension, vector<int\*> setIndexes |
| Return type: double | |
| Description: Static function: Calculates the maximum value in the sets at the passed indices for the indicated dimension | |
| Name: getMeanValue | Params: Dimension\* dimension, vector<int\*> setIndexes |
| Return type: double | |
| Description: Static function: Calculates the mean value of the sets at the passed indices for the indicated dimension | |
| Name: getMedianValue | Params: Dimension\* dimension, vector<int\*> setIndexes |
| Return type: double | |
| Description: Static function: Calculates the median value in the sets at the passed indices for the indicated dimension | |
| Name: getCenter | Params: |
| Return type: vector<double> | |
| Description: Static function: Returns the center nD point of the HB. | |
| Name: addSet | Params: int setIndex |
| Return type: bool | |
| Description: Adds set at passed index to the list of sets in the HB, returns true if set is added, returns false if set is already in cluster | |
| Name: removeSet | Params: int setIndex |
| Return type: bool | |
| Description: Removes set at passed index from the list of sets in the HB, returns true if set is removed, returns false if set is not in cluster | |
| Name: getSetNumber | Params: |
| Return type: int | |
| Description: Returns number of sets in HB | |
| Name: getIndexOfSet | Params: int clusterRelativeIndex |
| Return type: int | |
| Description: Returns the original index for the set in the cluster specified by passed index | |
| Name: getMinimum | Params: int dimensionIndex |
| Return type: double | |
| Description: Returns minimum value in the HB at passed dimension | |
| Name: getMaximum | Params: int dimensionIndex |
| Return type: double | |
| Description: Returns maximum value in the HB at passed dimension | |
| Name: getMiddle | Params: int dimensionIndex |
| Return type: double | |
| Description: Returns middle value in the HB at passed dimension (Mean if useMean else Median ) | |
| Name: calculateValues | Params: vecor<Dimension\*>\* dimensionToCalculateWith |
| Return type: | |
| Description: Calculates values for minimum, maximum, mean, and median of the HB using the passed dimensions | |
| Name: invertValues | Params: int dimensionToInvertValuesAt |
| Return type: | |
| Description: Inverts the values of the sets at the passed index | |
| Name: moveValues | Params: int originalIndex, int indexOfInsertion |
| Return type: bool | |
| Description: Changes the position of the values in the set at originalIndex to the new index(indexOfInsertion) | |
| Name: getSize | Params: |
| Return type: int | |
| Description: Returns number of nD points in HB | |
| Name: getColor | Params: |
| Return type: vector<double>\* | |
| Description: Returns the color components of the cluster | |
| Name: setColor | Params: vector<double>& newColor |
| Return type: | |
| Description: Sets color components of the cluster | |
| Name: setColor | Params: vector<double>\* newColor |
| Return type: | |
| Description: Sets color components of the cluster | |
| Name: getRadius | Params: |
| Return type: double | |
| Description: Returns radius used to create HB | |
| Name: setRadius | Params: double newRadius |
| Return type: double | |
| Description: Sets value of radius to passed newRadius, returns old value of radius | |
| Name: getOriginalSet | Params: |
| Return type: int | |
| Description: Returns the index of the nD point that was used to create the HB | |
| Name: setOriginalSet | Params: int newSet |
| Return type: int | |
| Description: Sets value of originalSet to passed newSet, returns old value of originalSet | |
| Name: getName | Params: |
| Return type: string\* | |
| Description: Returns the name of the HB | |
| Name: setName | Params: string\* newname |
| Return type: | |
| Description: Sets value of name to newname | |
| Name: getSets | Params: |
| Return type: vector<int>\* | |
| Description: Returns the list of nD points in the HB as indices in original data | |
| Name: isUseMean | Params: |
| Return type: bool | |
| Description: Returns the value of useMean | |
| Name: setUseMean | Params: bool newUseMean |
| Return type: | |
| Description: Sets value of useMean to passed newUseMean | |
| Name: isDisplayed | Params: |
| Return type: bool | |
| Description: Returns the value of displayed | |
| Name: setDisplayed | Params: bool displayed |
| Return type: | |
| Description: Updates value of displayed to passed new value | |
| Name: getVirtualCenter | Params: int numofDimensions |
| Return type: vector<double> | |
| Description: Calculates and returns list of local centers for all dimensions | |
| Name: getRatio | Params: int numOfDimensions |
| Return type: double | |
| Description: Returns the ratio between number of dimensions and number of data in HB | |

DataSet

A class that stores data for an nD point(set) individually.

Variables/Attributes

|  |  |
| --- | --- |
| Name: setClass | Type: int |
| Description: The index of the class that the set belongs to | |
| Name: originalIndex | Type: int |
| Description: Contains the original index where the nD point was found | |
| Name: currentIndex | Type: int |
| Description: Contains the current index for the nD point | |
| Name: name | Type: string |
| Description: Contains the name of the nD point | |
| Name: isVisible | Type: bool |
| Description: Determines whether or not to draw the nD point or not | |

Function/Methods

|  |  |  |  |
| --- | --- | --- | --- |
| Name: DataSet | Params: | | |
| Return type: | | | |
| Description: Creates a set at index -1 with name "-1" | | | |
| Name: DataSet | Params: int index, int classIndex | | |
| Return type: | | | |
| Description: Creates a set at the passed index that belongs in the passed class | | | |
| Name: getClass | Params: | | |
| Return type: int | | | |
| Description: Returns the index of the class that the set belongs to | | | |
| Name: setDataClass | Params: int newClass | | |
| Return type: int | | | |
| Description: Sets the value of setClass to the passed value, returns the previousl value | | | |
| Name: getIndex | Params: | | |
| Return type: int | | | |
| Description: Returns the index where the set can be found | | | |
| Name: setIndex | Params: int newIndex | | |
| Return type: int | | | |
| Description: Sets the value of currentIndex to the passed index, returns the previous value | | | |
| Name: getOriginalIndex | | | Params: |
| Return type: int | | | |
| Description: Returns the index where the set was found in the original data | | | |
| Name: getName | | Params: | |
| Return type: string\* | | | |
| Description: Returns the name of the set | | | |
| Name: setName | Params: string &newName | | |
| Return type: | | | |
| Description: Sets the value of name to the passed name | | | |
| Name: isVisible | Params: | | |
| Return type: bool | | | |
| Description: Returns the value of isVisible | | | |
| Name: setVisible | Params: bool newVisible | | |
| Return type: bool | | | |
| Description: Sets the value of isVisible to the passed visibility, returns the previous value | | | |

DataClass

The data structure that tracks and manages an individual class. Exists so that operations can be performed to a class as a whole rather than by going through points individually.

Variables/Attributes

|  |  |
| --- | --- |
| Name: index | Type: int |
| Description: Contains the index of the class | |
| Name: color | Type: ColorCustom |
| Description: Contains the color data of this class should be colored | |
| Name: name | Type: string |
| Description: Contains the name of the class | |
| Name: sets | Type: vector<int> |
| Description: Contains indexes for all of the sets in the class | |

Functions/Methods

|  |  |
| --- | --- |
| Name: DataClass | Params: |
| Return type: | |
| Description: Create a class at index -1 with name "-1" | |
| Name: DataClass | Params: int index, string name |
| Return type: | |
| Description: Creates a class at the passed index with the passed name | |
| Name: getIndex | Params: |
| Return type: int | |
| Description: Returns the index of the class | |
| Name: setIndex | Params: int newIndex |
| Return type: int | |
| Description: Changes index to passed index, returns previous value | |
| Name: getColor | Params: |
| Return type: vector<double>\* | |
| Description: Returns the color components for the class | |
| Name: setColor | Params: vector<double> &newColor |
| Return type: | |
| Description: Sets the color components for the class to the passed color components | |
| Name: setColor | Params: vector<double>\* newColor |
| Return type: | |
| Description: Sets the color components for the class to the passed color components | |
| Name: getName | Params: |
| Return type: string | |
| Description: Returns the name of the class | |
| Name: setName | Params: string\* newName |
| Return type: string | |
| Description: Sets the name of the class to the passed name, returns previous value | |
| Name: getSetNumber | Params: |
| Return type: int | |
| Description: Returns the number of nD points in the class. | |
| Name: getSetsInClass | Params: |
| Return type: vector<int>\* | |
| Description: Returns the index vector for the nD points in the class | |
| Name: addSet | Params: int setIndex |
| Return type: | |
| Description: Adds the nD point at the passed index to the class | |
| Name: removeSet | Params: ind setIndex |
| Return type: | |
| Description: Removes the nD point at the passed index from the class | |

DataInterface

The class used to interface with the input file(s) and save the data in memory as well as perform several different operations on the data, including but not limited to, creating hyperblocks, sorting and manipulating dimensions, and drawing hyperblocks. The main data structure for the data set is list of Dimensions contained in the variable dataDimensions.

Variables/attributes

|  |  |
| --- | --- |
| Name: dataDimensions | Type: vector<Dimension\*> |
| Description: Contains the dataset. Original data and current data. | |
| Name: dataClasses | Type: vector<DataClass> |
| Description: Contains the dataset divided into and indexed by class. Can be used to select an entire class | |
| Name: dataSets | Type: vector<DataSet> |
| Description: Contains the list of nD points and associates them with their respective classes | |
| Name: clusters | Type: vector<SetCluster> |
| Description: Contains all of the hyperblock information. Each in a separate SetCluster | |