Marginal Change Restrictor Algorithm.

This algorithm applies to XYYY data, and in the text below each x-row corresponds to a different x value (in this case, the mass fragment number).

1. Cast data reference array to floats.
2. Remove sign change from data
   1. Loop through X-rows
      1. Loop through columns and determine if there is a sign change present and the difference is larger than the ignorable delta y threshold.
      2. Loop through columns and interpolate new abscissa values with new X-rows. Base the order of inserted X-rows off of a sorted list of abscissa values.
      3. Insert a new X-row containing a zero in the sign change location and interpolated values in the rest of the X-row.
3. Remove the effect of zero/nearZero values. The Marginal Change Restrictor cannot calculate with zero values due to the use of log(base=2). Additionally, the Marginal Change Restrictor needs a safeguard to keep it from interpolating values under the ignorableDeltaYThreshold(considered nearZero). For x-rows containing zeros and near-zero values with a threshold crossing, insert x-rows above/below containing half of the ignorableDeltaYThreshold above/below the value in question. The other values in the x-row are linearly interpolated from the x-rows above and below.
   1. Make a parallel array to the data as a key to the locations of the nearZeros and zeros (nearZeroIndicesArray). Locations of zeros and nearZeros will be represented by zeros while the other elements will be represented with 1s. This will not change size as the data changes size. It is only meant to make finding the locations in the original data array where rows need to be added
   2. Loop through X-rows of the nearZeroIndicesArray (These steps will be done separately for both rows above and below the row being evaluated.
      1. Loop through y-columns. If a nearZero value and a threshold crossing are present, calculate and store the abscissa value for rows that will be inserted with the inserted x-row containing the half-threshold value.
      2. Sort the abscissa values that need to be inserted according in an increasing order.
      3. Use the abscissa values to be inserted to interpolate the corresponding X-row
      4. Insert the interpolated abscissa values and the corresponding X-row into the dataList.
   3. Make a parallel array based on the data which will serve as a key to the locations of the zeros (zeroIndicesArray). Locations of zeros will be represented by zeros while non-zeros will be represented with 1s. This will be used in the main Marginal Change Restrictor function to return all originally zero values to zero.
   4. Set all zeros equal to one-tenth of the threshold value.
4. Go through the actual data interpolation.
   1. Initialize list variables to be added to in the main interpolation loop: interpolated\_data, interpolated\_abscissa, data\_gaps, insertion\_indices
   2. Make the data (YYY), abscissa(X) and zero indices arrays into lists
   3. For loop to loop through the X-rows of the original data and interpolate between the rows if necessary
      1. Make a list containing the differences of the Y value in the current row and the corresponding Y value in the next\_row.
         1. Make a parallel array (negligibleAbsoluteDifferencesArray) for the current\_row containing the location of differences less than the IgnorableDeltaYThreshold. Significant differences (larger than the threshold) are marked with a 1, while the insignificant differences are notated at a 0.
         2. If the whole significant difference row does not contain any significant differences, do not go through the rest of the Marginal Change Restrictor.
      2. Determine any gaps in any of the Y-columns that are larger than the Max Allowed Delta Y Ratio by taking the ratio of each element in the current X-row to the corresponding values in the next X-row
         1. Calculate the log of the ratios across the row to be used to form new abscissa points.
      3. Multiply the ratios across the X-row by the ZeroIndicesArray and the negligibleAbsoluteDifferencesArray to eliminate ratio evaluations based on points below the IgnorableDeltaYThreshold by making the log(ratio) at that point equal to 1.
         1. Multiply the current row in the data list by the zeroIndicesArray to return any zeros to zero that were changed during the adjust for zero step.
      4. Find the max log ratio and index and the min log ratio and index
      5. Form new abscissa points based on both the points with the maximum log ratio and the minimum log ratio.
         1. Combine the two abscissas and sort in an increasing or decreasing order depending on if the abscissa between the current row and next row is increasing or decreasing. .
         2. Interpolate X-rows containing data gaps to be inserted into the insertion points. Store the interpolated data in the data\_gaps list.
         3. Use the new abscissa to make a list of insertion points. Store the insertion points in the insertion\_indices list.
   4. Insert the new X-rows in the data gaps list into the data at the insertion points.
   5. Create a new zeroIndicesArray that contains the location of any added zeros from the main interpolation. This overwrites the previous zeroIndicesArray
5. Remove Superfluous Rows. The main marginal Change Restrictor function inserts abscissa values and the corresponding YYY data rows for both the minimum and the maximum log ratios in the row, which can create rows that are superfluous. If the rows are superfluous, all the values in the two surrounding rows are already within the MaxAllowedDeltaYThreshold (or the difference between points is less than the IgnorableDeltaYThreshold). Superfluous row cleanup deletes the superfluous abscissa value and corresponding row of YYY data. Additionally, rows that were not created through the main Marginal Change Restrictor will not be evaluated or deleted. This is a general description and further details on the algorithm and exceptions are included in the comments in the function.
6. Return the data and abscissa as arrays and the insertion\_indices.