

```

Input: /*dynamically fixed arguments if absent*/
matrix, SPMiner, orientation, representation, symmetries, stopCriteria,
normalizer, discretizer, missingsHandler, noiseHandler, extender, merger, filter

main begin
    orderedIndexes  $\leftarrow$  runMappingStep();
    biclusters  $\leftarrow$  runMiningStep();
    return runClosingStep(biclusters);

runMappingStep begin
    if isColumn(orientation) then matrix  $\leftarrow$  transpose(matrix);
    discData  $\leftarrow$  discretize(matrix, normalizer, discretizer);
    treatedData  $\leftarrow$  applyHandlers(discData, missingsHandler, noiseHandler);
    return createTransactions(treatedData);

runMiningStep begin
    seqPatterns  $\leftarrow$   $\emptyset$ ;
    if symmetries then
        factors  $\leftarrow$   $\emptyset$ ;
        foreach column  $j$  in orderedIndexes do
            colAdjusts  $\leftarrow$  alignSignal(orderedIndexes[.][j]);
            if colAdjusts  $\in$  factors then continue;
            else factors  $\leftarrow$  factors  $\cup$  colAdjusts;
            alignedData  $\leftarrow$  alignDataByRows(colAdjusts, orderedIndexes);
            seqPatterns  $\leftarrow$  seqPatterns  $\cup$  runSPM(alignedData);
            if allCombinations(factors) then break; //pruning;
    else seqPatterns  $\leftarrow$  runSPM(orderedIndexes);
    return getBiclustersFromPatterns(seqPatterns, orientation);

runClosingStep(biclusters) begin
    biclusters  $\leftarrow$  merge(biclusters, merger);
    biclusters  $\leftarrow$  extend(biclusters, extender);
    return increaseConsistency(biclusters, filter);

runSPM(data) begin
    if estimateMinimumLimits(stopCriteria) then
        (minX, minY)  $\leftarrow$  findExpectedLimits(data);
        seqPatterns  $\leftarrow$  SPM(SPMiner, minX, minY, data, representation);
    else
        minSup  $\leftarrow$  0.5;
        while minAreaCovered(seqPatterns, 10%) do
            seqPatterns  $\leftarrow$  SPM(SPMiner, minSup, data, representation);
            minSup  $\leftarrow$  minSup  $\times$  0.8;
    return seqPatterns;

```