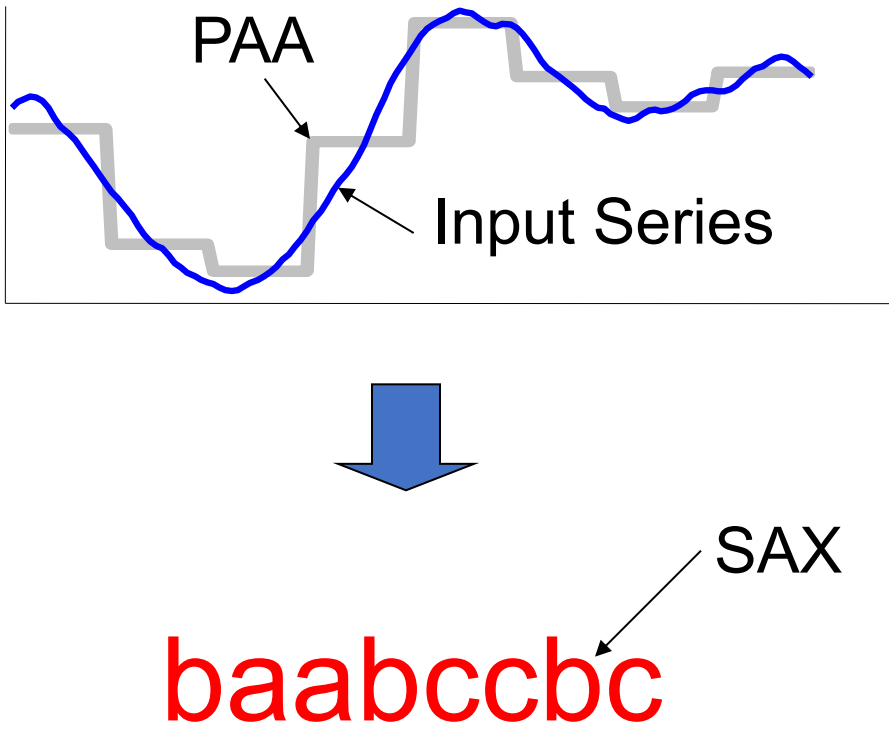
Additional Material – Example on how to calculate SAX

Creating Symbolic Aggregate Approximation (SAX)

- Input
  - Real valued time series (blue curve)
- Output
  - Symbolic representation of the input time series (red string)
- Process
  - First convert the input series into piecewise aggregate approximation (PAA) representation (grey steps)
  - Then convert the PAA into a string of symbols (red string)



Example Data

Time	Depth
20	4.2
40	9.2
60	14.8
80	15
100	17
120	18
140	19.7
160	20
180	20.8
200	21.3
220	21.6
240	20.6
260	16.9
280	12.8

Creating PAA

- Normalize the input time series
  - Subtract the mean from each value and divide the deviation with standard deviation
- Divide input time series of length n into w portions of equal length
  - w is the parameter that controls the length of PAA and therefore the length of SAX
  - If w is large you have a detailed (fine) PAA and a detailed SAX
  - If w is small you have an abstract (coarse) PAA and an abstract SAX
  - Choice of w should be based on the application requirements

Creating PAA (2)

- Two cases
  - n/w is a whole number
    - Simple case of each portion having n/w number of values from the input time series
  - n/w is a fraction
    - Complicated case because you cannot assign equal number of whole numbered values from the input series to w equal sized portions
    - Our example data has n = 14
    - If w = 3, then n/w is a fraction
    - The length of each portion is 14/3 = 4.66667
    - Each portion should have 4.66667 values from the original time series

Creating PAA (3)

- We use the following scheme to achieve 4.6667 values in each portion
- The following is the list of indexes of the 14 values in a input series

1 2 3 4 5 6 7 8 9 10 11 12 13 14

- The first portion will have values at 1, 2, 3, and 4
- We need 0.6667 more to complete this portion
- We achieve this by inserting 0.6667 times the 5^{th} value
- The remaining 0.3333 times the 5^{th} value is inserted into the second portion

Creating PAA (4)

- Using the above scheme our three lists are
  - 4.2, 9.2, 14.8, 15 and 0.6667*17
  - 0.3333*17, 18, 19.7, 20, 20.8, 0.3333*21.3
  - 0.6667*21.3, 21.6, 20.6, 16.9, 12.8
- (Note: here we have shown the values from the un-normalized input series)
- Each of the above sublists have equal portions from the input series
- Next for each of the sublists compute the average (mean)
- In our case, three sublists will each have an average value
- PAA is simply a vector of these average values
  - {avg1, avg2, avg3}
  - {-0.9338,0.53135,0.34767} for our example (using normalized values)

Properties of PAA

- PAA is simple to compute (as can be seen from the previous slides)
- Achieves dimensionality reduction
  - From 14 values our input series is reduced to 3 values
- Any similarities computed on the PAA will be true on input series as well
  - Lower bounding distance
  - Very useful property for a structural representation
  - Allows data analysis to be performed on the approximate representation rather than the original series

Symbol Mapping

- In this step, each average value from the PAA vector is replaced by a symbol from an alphabet
- An alphabet size, *M* of 5 to 8 is recommended
  - a,b,c,d,e
  - a,b,c,d,e,f
  - a,b,c,d,e,f,g
  - a,b,c,d,e,f,g,h
- Given an average value we need a symbol
- This is achieved by using the normal distribution from statistics
  - Because our input series is normalized we can use normal distribution as the data model
  - We divide the area under the normal distribution into '*M*' equal sized areas where *M* is the alphabet size
  - Each such area is bounded by breakpoints

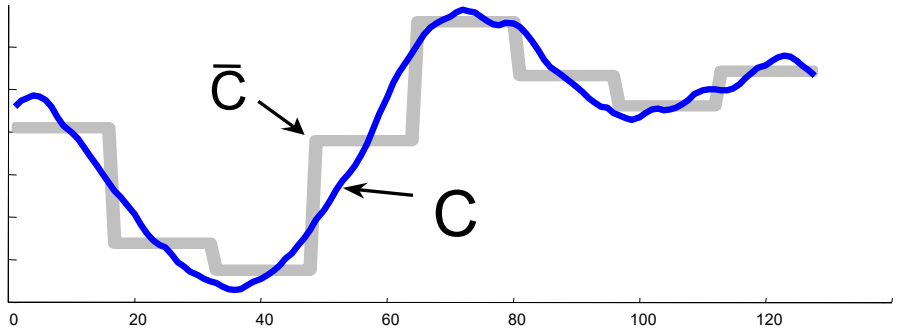
Symbol mapping - breakpoints

- Breakpoints for different alphabet sizes can be structured as a lookup table

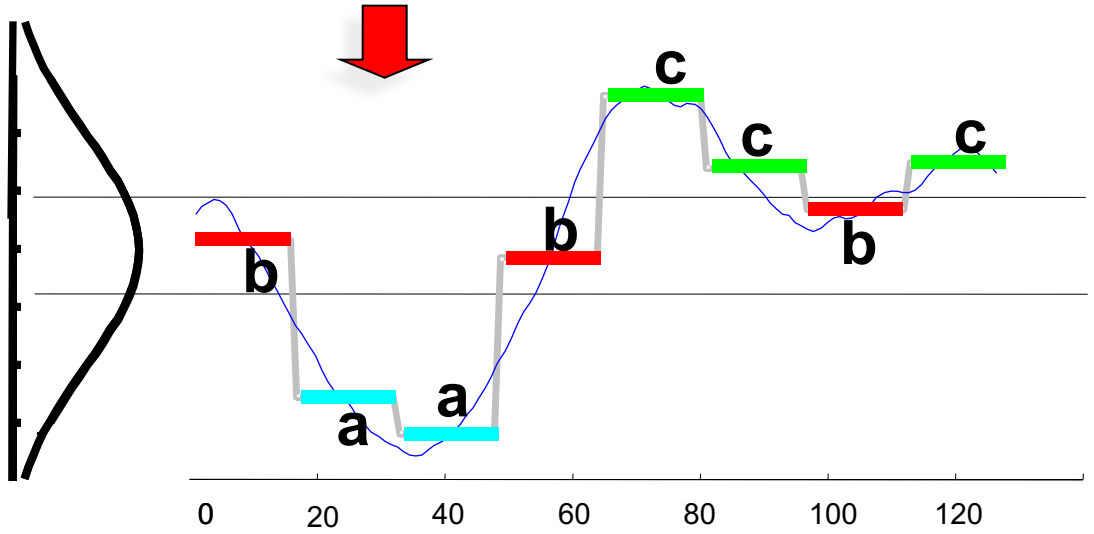
- When *M*=3
  - Average values below -0.43 are replaced by ‘A’
  - Average values between -0.43 and 0.43 are replaced by ‘B’
  - Average values above 0.43 are replaced by ‘C’

	α=3	α=4	α=5
b1	-0.43	-0.67	-0.84
b2	0.43	0	-0.25
b3		0.67	0.25
b4			0.84

SAX Computation – in pictures



This slide taken from Eamonn's Tutorial on SAX



baabccbc

Data Analysis using SAX

- A general approach is to convert time series into SAX
- Use SAX representations to train Markov models (details not here) on normal data
  - The model captures the probabilities of normal patterns
- The trained models are then used to test incoming data for known and unknown patterns

Visualisation using SAX

- Given a SAX representation
  - count the frequencies of patterns (substrings) of required length and
  - use them to colour code a mosaic for visualizing time series
- For example, given ‘baabccbc’ as the SAX representation
  - We calculate the frequencies of substrings of length 1 and represent them in a mosaic
- Visualisations for substrings of length>1 are possible (please refer to the SAX site)