There are now the problem of what order of alignment needed so that we can guarantee alignment

Currently, if index is positive, the updated TDC is the one earlier in the comparison.

Lets say that we are aligning pairwise first of all

We can align

0 to 1, 1 to 2, 2 to 4

It would be a sanity check to see if the alignment persists by further comparison

Which are 0 to 2, 0 to 4, 1 to 4.

By doing this, we are saying that if last was bad, we look further into the range of alignment. This caused instability at the start of the sequence, where a adjustment of 22 was needed to fully align the data.

There are two problems, one is large adjustment needed, two is data sloshing, where by chance you get a set of lucky alignment where TDC4 didn’t read as much data. Hence you slosh the data around, making them very poor  
Found a new alignment, offsetting by 6 idx is 133000 updated TDC 1

Found a new alignment, offsetting by -6 idx is 133000 updated TDC 4

Found a new alignment, offsetting by -6 idx is 133000 updated TDC 4

Found a new alignment, offsetting by 6 idx is 133800 updated TDC 0

Found a new alignment, offsetting by -6 idx is 133800 updated TDC 2

Found a new alignment, offsetting by 6 idx is 133800 updated TDC 2

Found a new alignment, offsetting by 6 idx is 133800 updated TDC 0

Found a new alignment, offsetting by -6 idx is 134600 updated TDC 1

Found a new alignment, offsetting by 6 idx is 134600 updated TDC 1

Found a new alignment, offsetting by -6 idx is 134600 updated TDC 4

Found a new alignment, offsetting by -6 idx is 134600 updated TDC 4

Found a new alignment, offsetting by 6 idx is 135400 updated TDC 0

Found a new alignment, offsetting by -6 idx is 135400 updated TDC 2

Found a new alignment, offsetting by 6 idx is 135400 updated TDC 2

Found a new alignment, offsetting by 6 idx is 135400 updated TDC 0

One possible solution could be to say that when an alignment occurs, it updates everything that is to the back of the list. For example

TDC A, B, C, D

When A com B, if positive, A adjust, if negative, all B, C, D adjust for that value. Doing this method, you can no longer do pair wise comparisons since it is even more susceptible to oscillation. However, what one can do is to do [0,1], [1, 2], [2,4] using continuous update, and individual updates between [0, 2] [0, 4][1, 4]

A blue rectangular graph with white text

Description automatically generated

As you can see, without minor adjustment, the global alignment is VERY VERY VERY VERY GOOD!!!! This was done using the test of [0, 2], [0, 4] and [1, 4], the condition of True is that any pairwise TDC must match.

A graph of different colored lines

Description automatically generated with medium confidence

A graph of a diagram

Description automatically generated with medium confidence

Beam 2

A graph with a blue rectangle

Description automatically generated

A graph of a line

Description automatically generated

Beam 1

A graph with a blue rectangle

Description automatically generated

A graph with blue lines

Description automatically generated

This was shit

A graph of a number of different sizes

Description automatically generated with medium confidence

0.0267

A graph of a bar graph

Description automatically generated

However this doesn’t tell us much, because the metric sometimes fails due to unluckiness, hence individual chunk misalignment can be misleading. If we extend the chunk to larger values, this might help if we just use a window to scan over it. However what is more interesting might be to look at is how does changing intervals affect your beam 2 True percentage. Doing this might need to extend my search further

A graph with a line going up

Description automatically generated

This is very useless.but it did make sense, larger intervals means more average right?

A graph of a window number

Description automatically generated