BO Verification and Planning

Testing the BLAST Oligo Implementation in JavaMAGE

Samir Ahmed

Thursday November 17th2011

Select 500 Random Basepairs and Duplicate

```
> No fixed matches
CCATACCGTCCTAATTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCCTA
CCTGCCTTTGCGTCTTGATACCAATGAAAAACCTATGCACTTTGTACAGG
GTGCCATCGGGTTTCTGAACTCTCAGATAGTGGGGATCCCGGGTAAAGAC
CTATATCTGCGGTCCAACTTAGGCATAAACCTCCATGCTACCTAGTCAGA
CCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG
TTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTAT
TACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA
ATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGG
TAACGGAGAGACTCTGCGGCGTACTTAATTATGCATTTGAAACGCGCCCA
AGTGACGCTAGGCAAGTCAGAGCAGGTTCCCGTGTTAGCTTGAGGGTAAA
CCATACCGTCCTAATTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCCTA
CCTGCCTTTGCGTCTTGATACCAATGAAAAACCTATGCACTTTGTACAGG
GTGCCATCGGGTTTCTGAACTCTCAGATAGTGGGGATCCCGGGTAAAGAC
CTATATCTGCGGTCCAACTTAGGCATAAACCTCCATGCTACCTAGTCAGA
CCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG
TTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTAT
TACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA
ATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGG
TAACGGAGAGACTCTGCGGCGTACTTAATTATGCATTTGAAACGCGCCCA
AGTGACGCTAGGCAAGTCAGAGCAGGTTCCCGTGTTAGCTTGAGGGTAAA
```

Add 2 oligos At positions 250 and 750

```
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 250) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 750) );
```

Span 1

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG …
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Span 2

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG …
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Overlap

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Everything Matches

Add 2 oligos At positions 250 and 850

```
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 250) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 850) );
```

Span 1

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG …
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Span 2

TGGTAGCACAAAAGTATTACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA …
aattccggATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGGTAACGGAGAGACTCTGC

Overlap

TGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Match at	Span	Start	End
	Span 1 :	109	142
	Span 2 :	1	34

3 Oligo Test Case

3 Oligos at Target positions 250, 650, 850

No Match Between B & C

3 Oligo Test Case

3 Oligos, with 2 mistargets

```
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 250) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 650) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 850) );
```

Oligo A (250)

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG …
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTT***TGGTAGCACAAAAGTATTACCATGGTCCTAGAAG***

Oligo B (650)

TATGCACTTTGTACAGGGTGCCATCGGGTTTCTGAACTCTCAGATAGTGGGGATCCCGGGTAAAGAC
aattccggCTATATCTGCGGTCCAACTTAGGCATAAACCTC***CATGCTACCTAGTCAGACCCACCCCGCACGGGGT***

Oligo C (850)

TGGTAGCACAAAAGTATTACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA …
aattccggATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGGTAACGGAGAGACTCTGC

How the Heuristic Works

Given a pool of *n* spans with *L* possible oligo subsequences of ideal length,

For all *n,* calculate local Blast Oligo values for each oligo L.

Find the oligo tbe Blast Oligo value

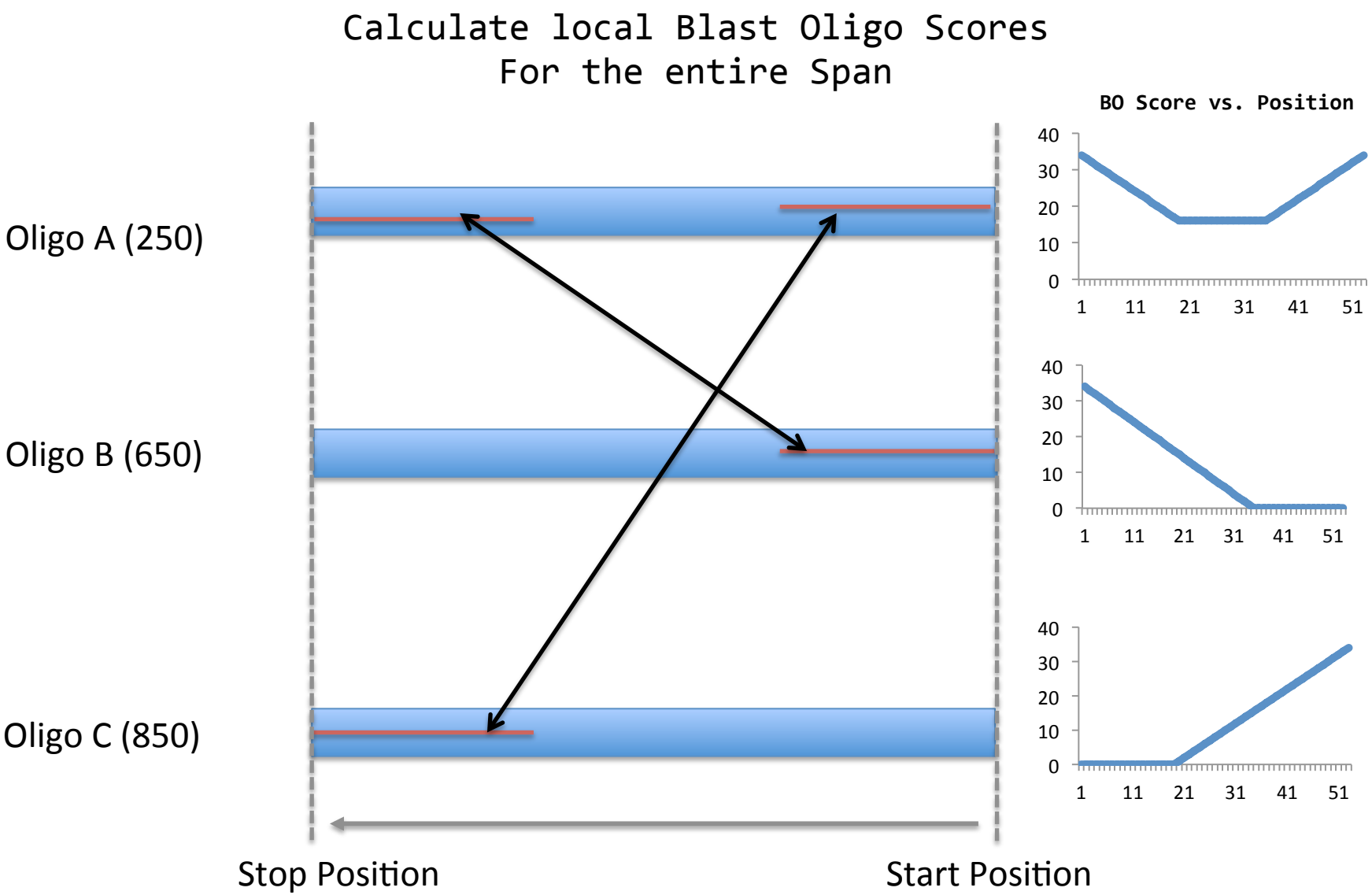
Select that **optimized** oligo

Repeat from step 1.

Once you have selected an oligo from a Span, disregard it from the pool

- The term ‘optimized’ implies the best possible oligo, selected with the following precedence
  - Free Energy
  - Genome Homology
  - Oligo Pool Homology

3 Oligo Example Visualized



3 Oligo Visual Explanation

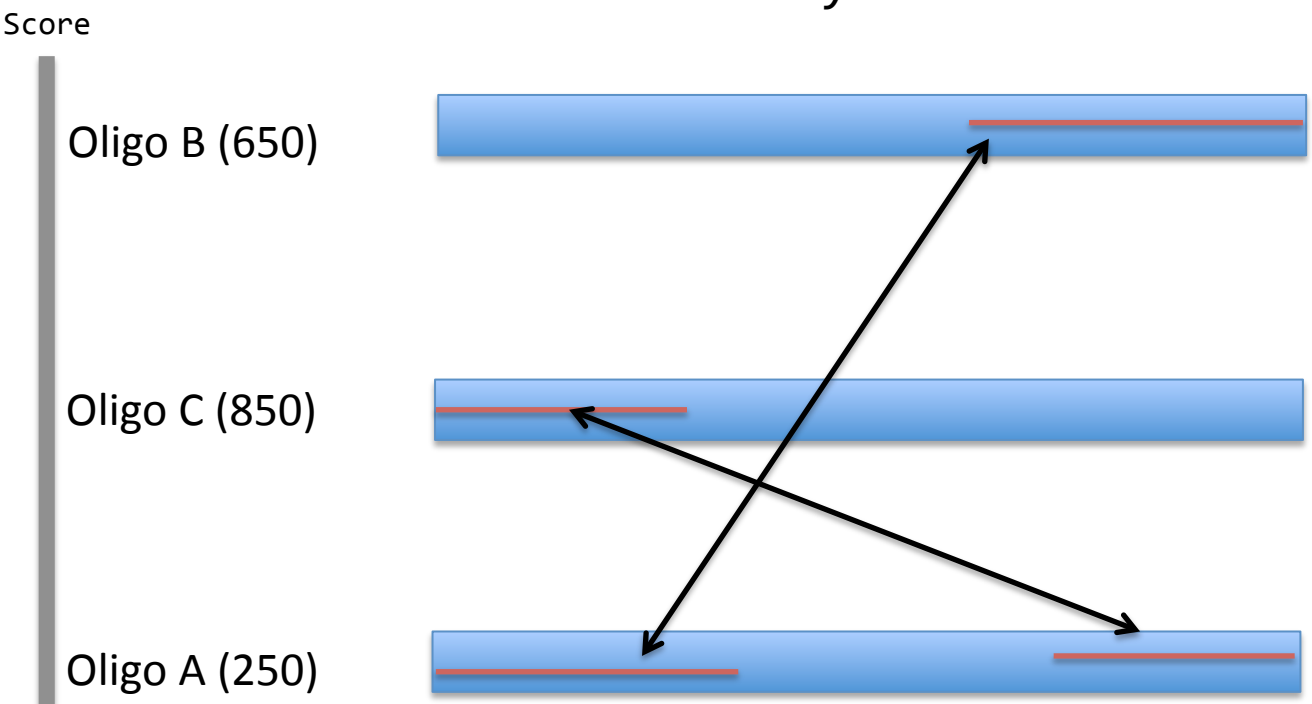
Lowest Individual BO Score

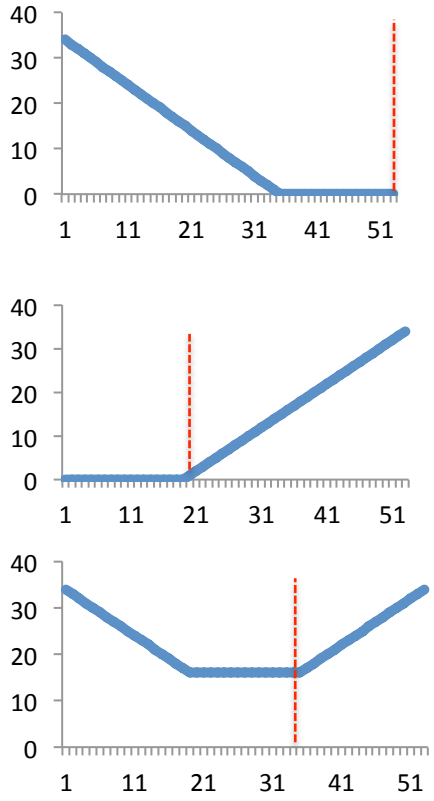
Sort by BO Values by Smallest Possible, Valid by DG Threshold

Oligo B (650)

Oligo C (850)

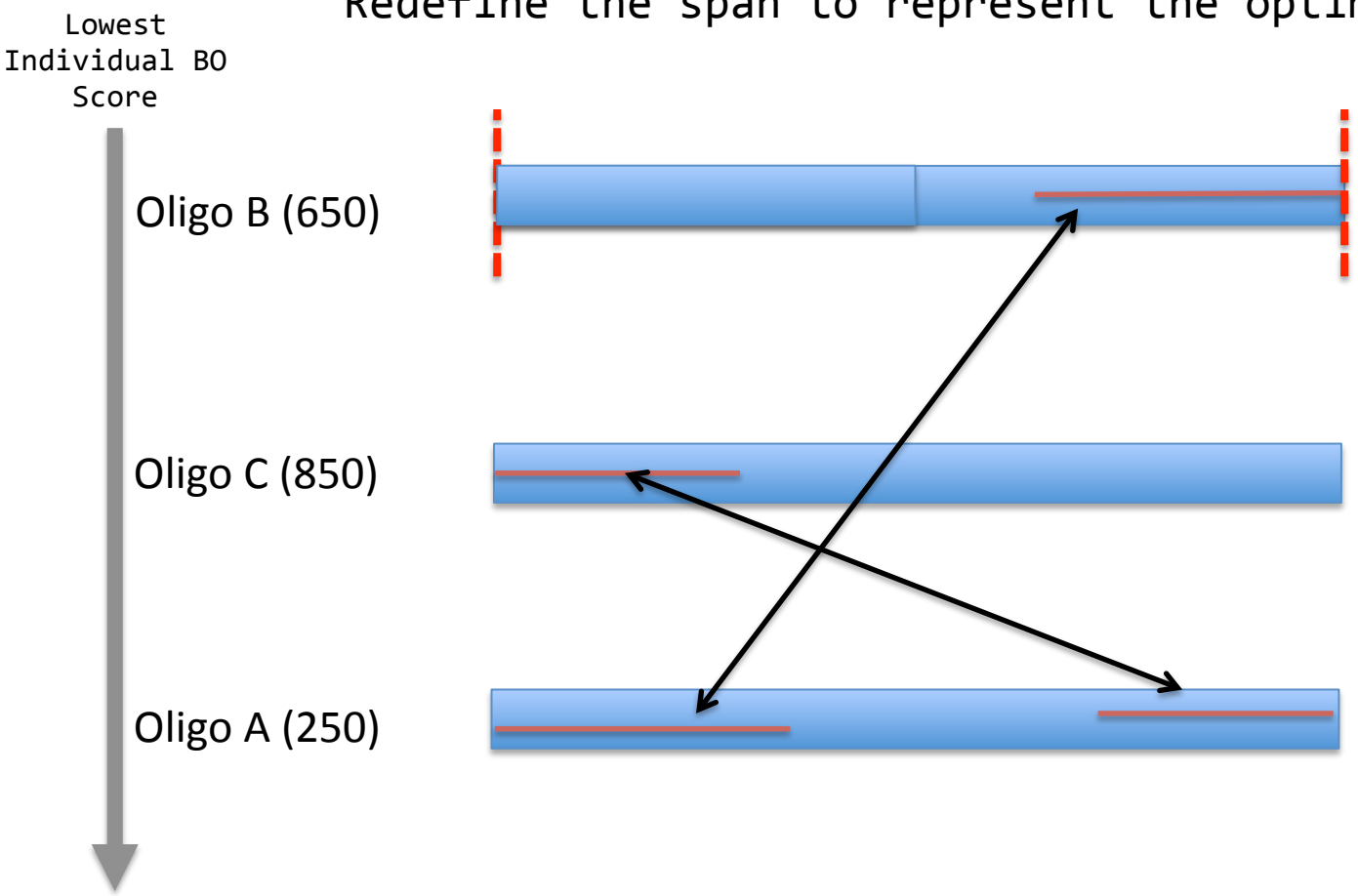
Oligo A (250)





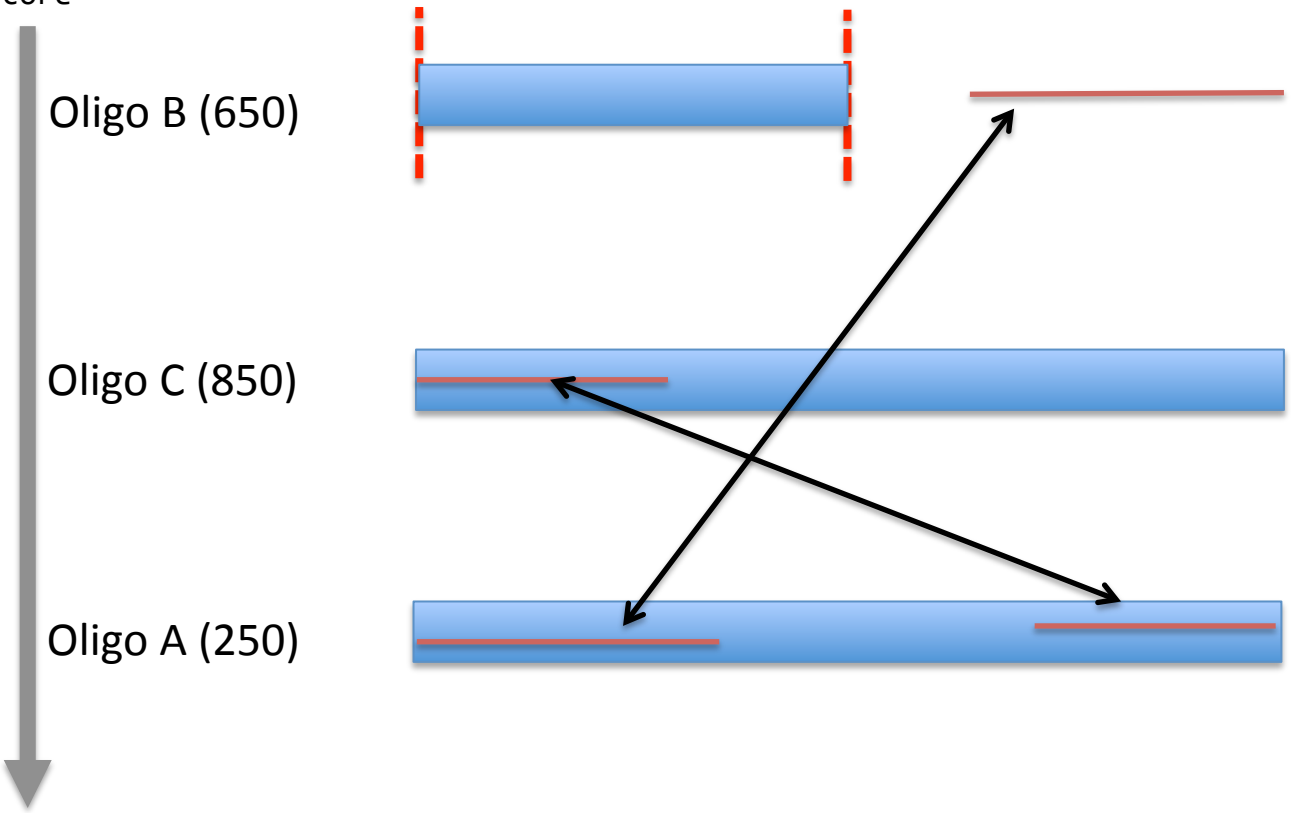
3 Oligo Visual Explanation

Redefine the span to represent the optimal Oligo

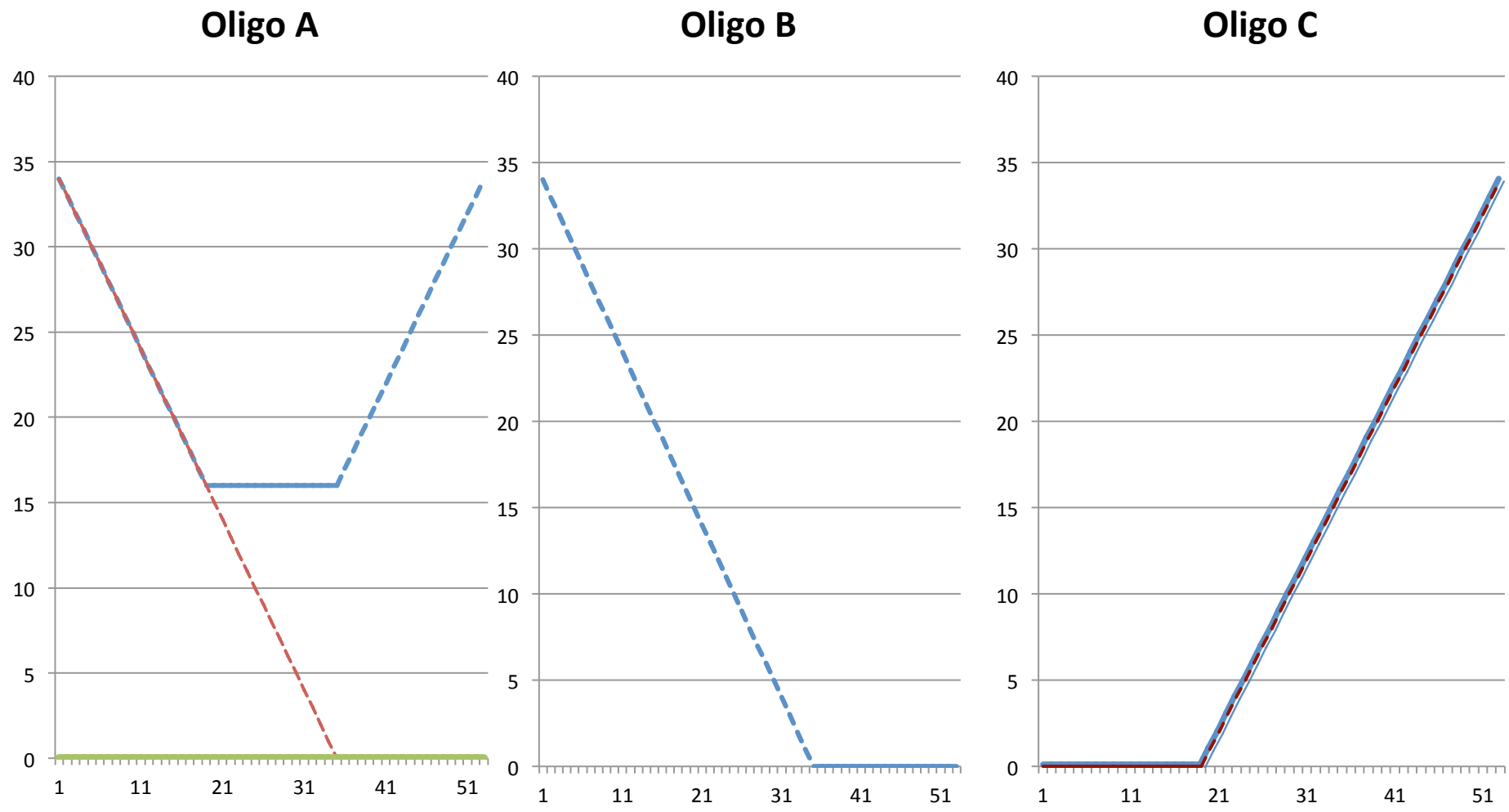


3 Oligo Visual Explanation

Recalculate the Blast Oligo Scores for the remainder
Of the Oligos, to update what has been affected

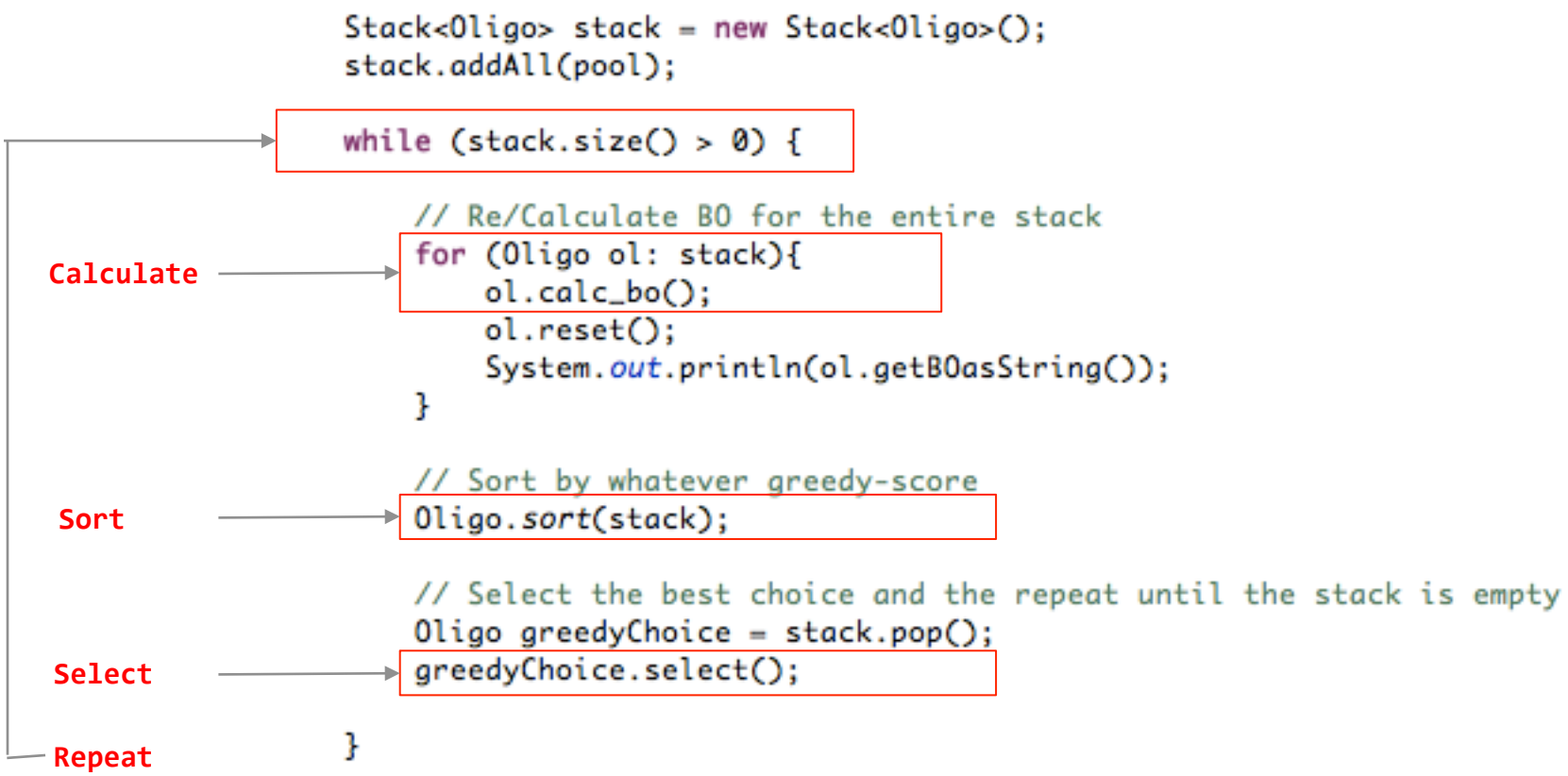


BO Scores at Each Iteration

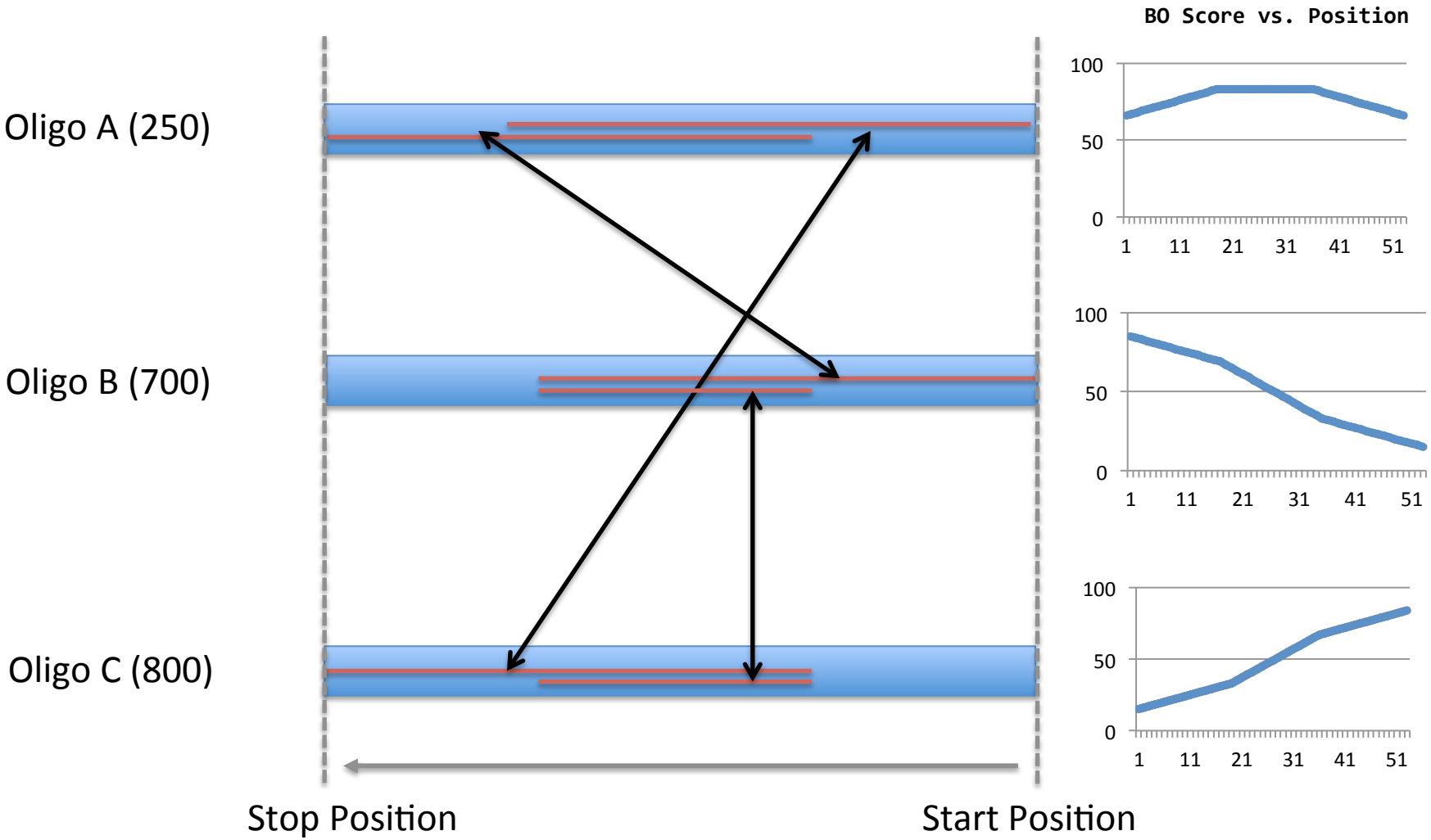


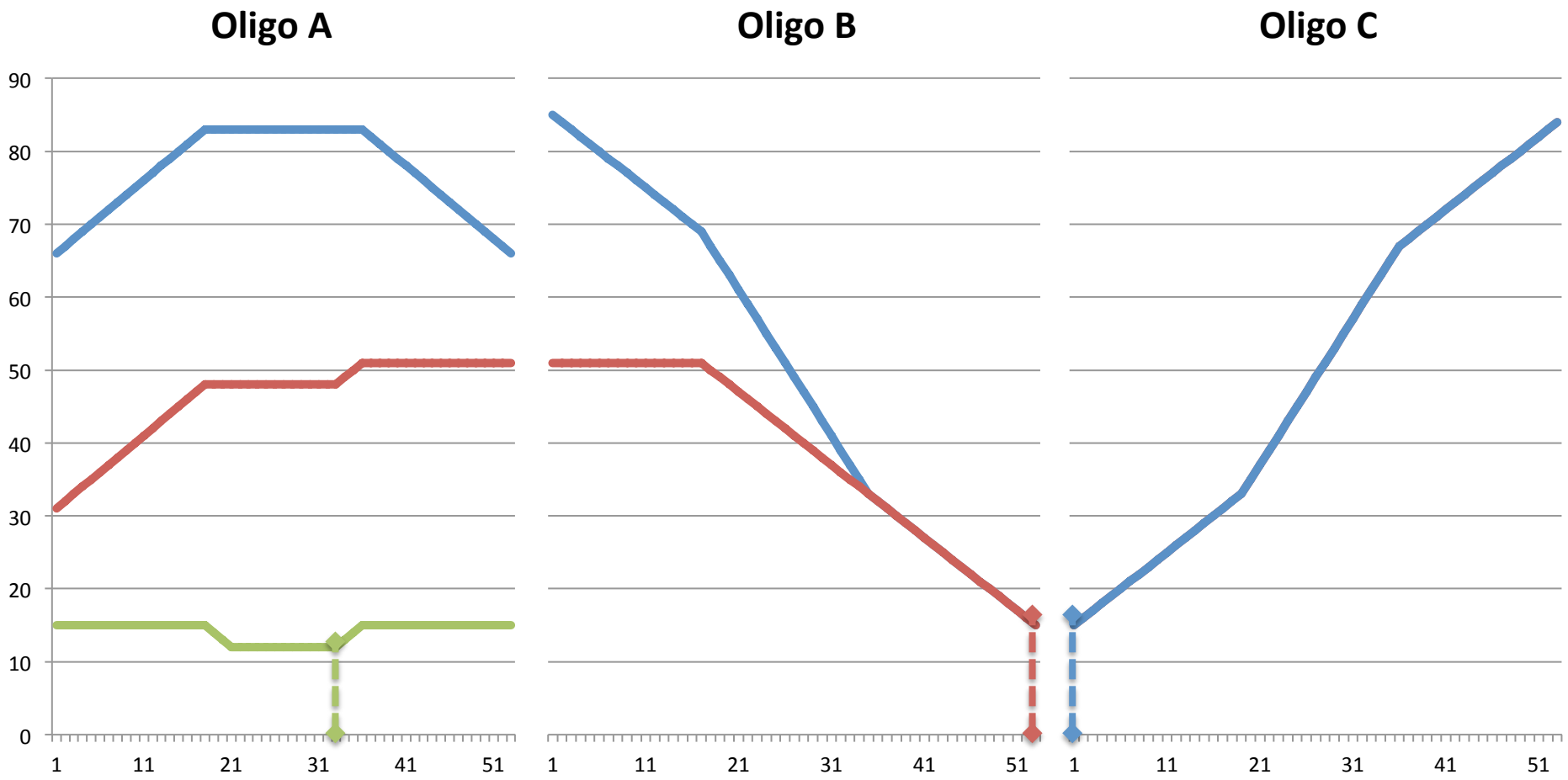
Blue = 1st Iteration
Red = 2nd Iteration
Green = 3rd Iteration

Implementation (High Level)



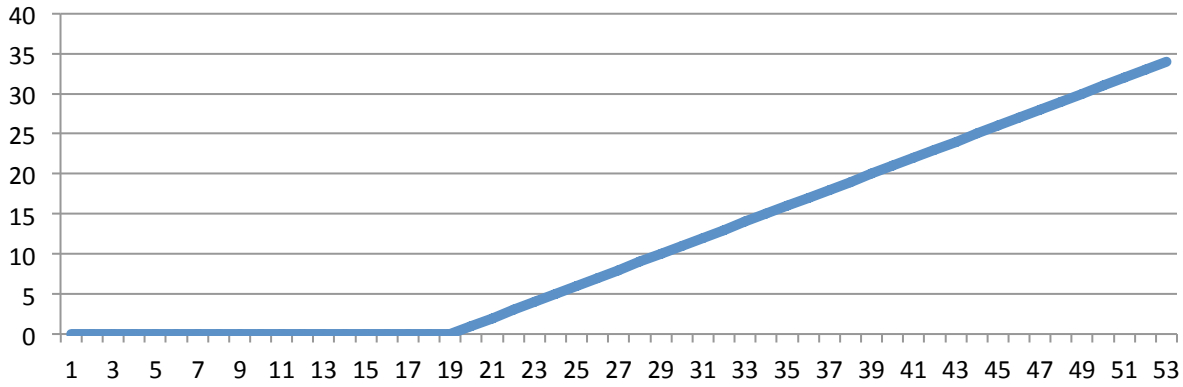
Another Example with More Mistargets

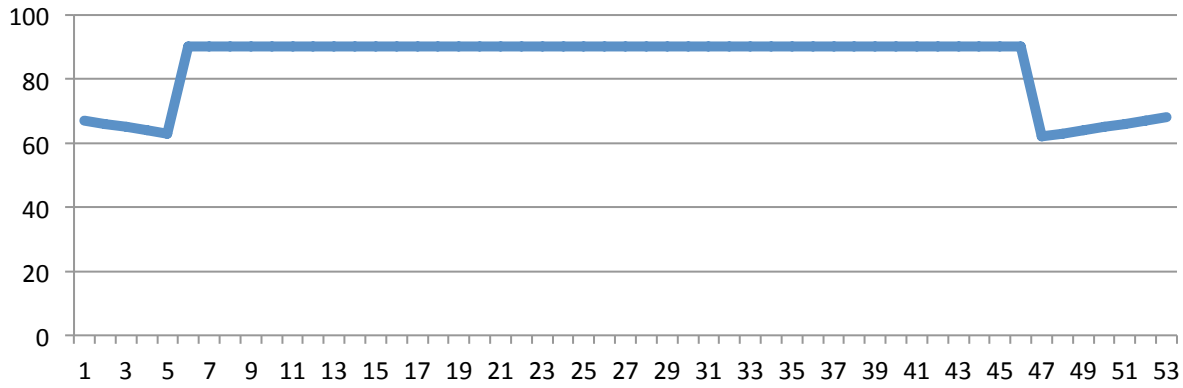


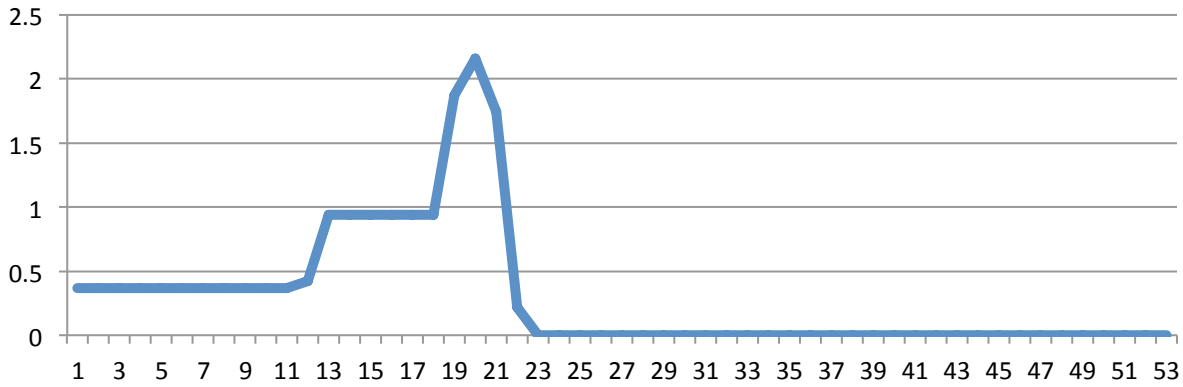


Blue = 1st Iteration
Red = 2nd Iteration
Green = 3rd Iteration

How Do you Compare Them?



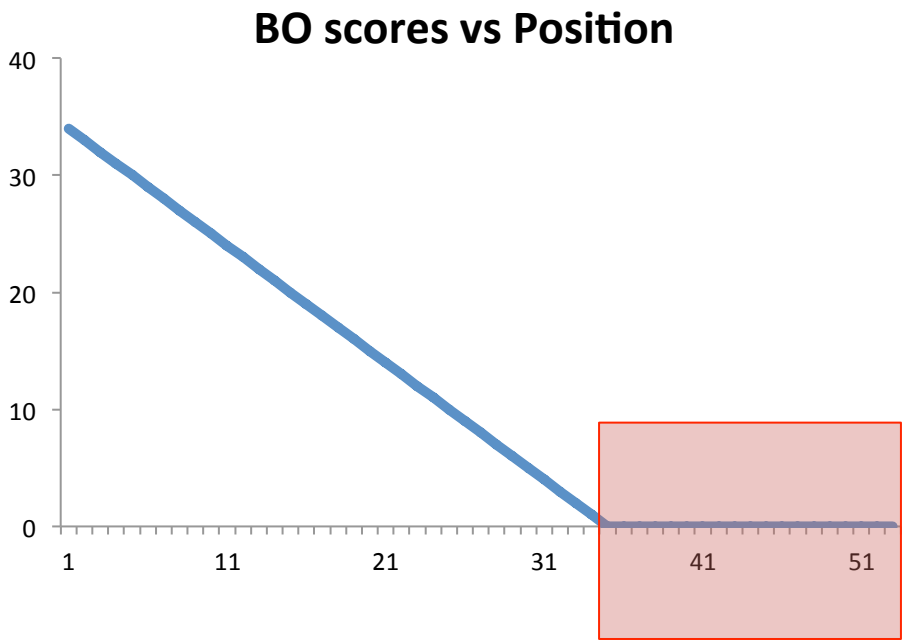




However BG, is not factored in

In a case where we have multiple, Ideal BO Choices (seen right)

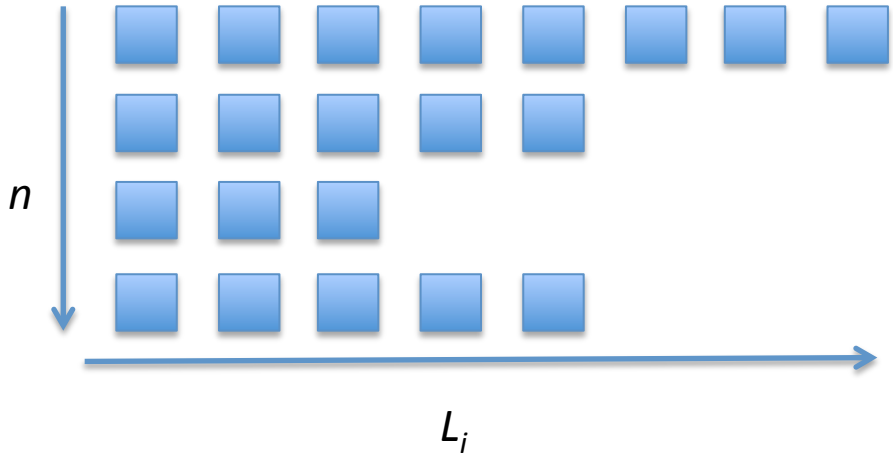
Select the best BG given a BO constraint



Exhaustive Search

Exhaustive Search to search every possibility.

Pool of *n* Spans, with L_i Possible Oligos each



Total Possibilities = $\prod_{i=0}^{n} L_i$

Recursive Approach to Exhaustive Search

```
ExhaustiveSearch (span)

if (Span == null)
    Calc_Score()
else
    for (i=1:Span.marginSize )
        if (span(i).hasValidDG)
            Span.select(i);
            ExhaustiveSearch(Span.next)
```

If we are not reach the end of the tree,

Calculate the score thus far.

Else take every possible Oligo

if the Oligo has a valid DG value

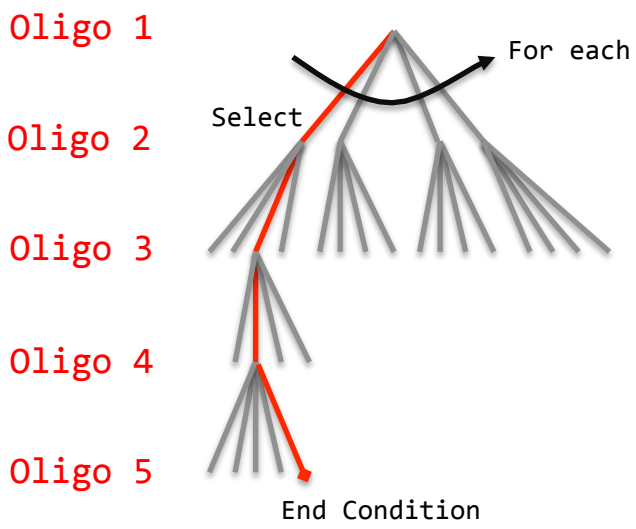
Redefine the bounds on the span

Repeat for next Span in Pool

Recursive Approach to Exhaustive Search

```
ExhaustiveSearch (span)


if (Span == null)
    Calc_Score()
else
    for (i=1:Span.marginSize )
        if (span(i).hasValidDG)
            Span.select(i);
            ExhaustiveSearch(Span.next)
```



Current Work

General

- Add Mistarget/ Deletion Functionality
- Add Sense/ Replicore Funcationaliy

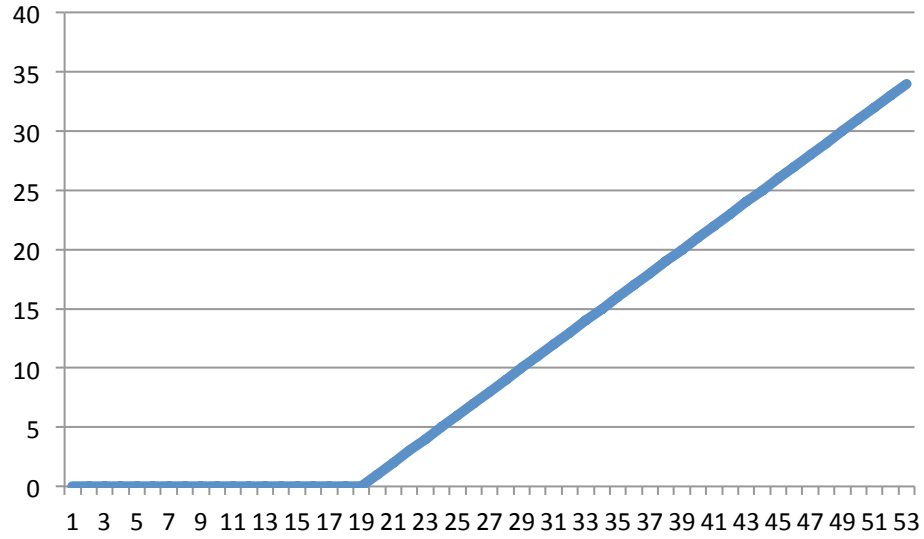
Heuristic

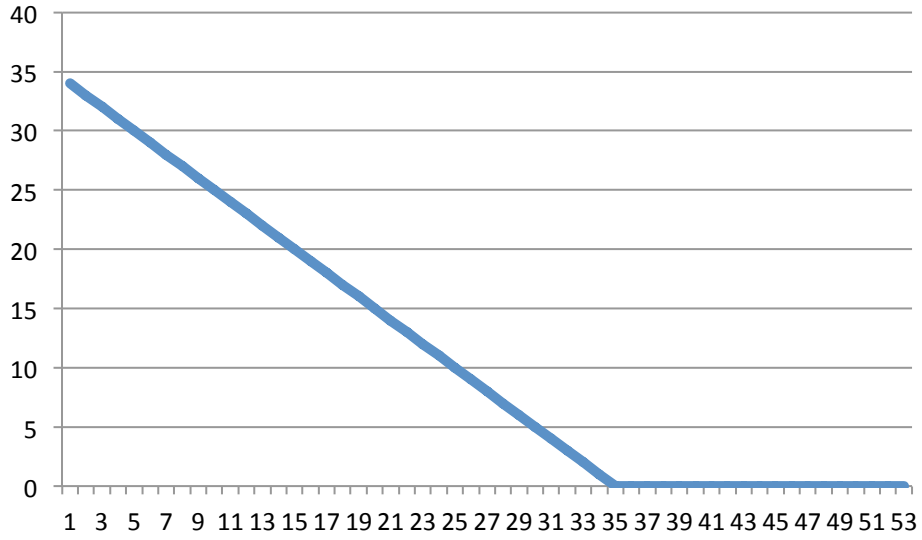
- Complete Switches
  - 100x1 vs. 1x100 Switching
  - Work in the DG
- Integrate Plotting
  - Watch the Visualization

Exhaustive Search

- Implement and Test

BO Scores for the Entire Span (Before Placing any Bounds)





Primary BO

The Blast Oligo Score of the 90bp Oligo taken from the primary position on the span

Primary Position = f(BlastGenome, Free Energy)

Span 1

CATGCTACCTAGTCAGACCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG …
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCCTAGAAG

Span 1
[Oligo @ Primary]

CGTCCGACCTGGTTCCTGGCG
aattccggTTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTATTACCATGGTCC

28bp Match out of 34bp

Span 2

TGGTAGCACAAAAGTATTACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA
aattccggATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGGTAACGGAGAGACTCTGC

Span 2
[Oligo @ Primary]

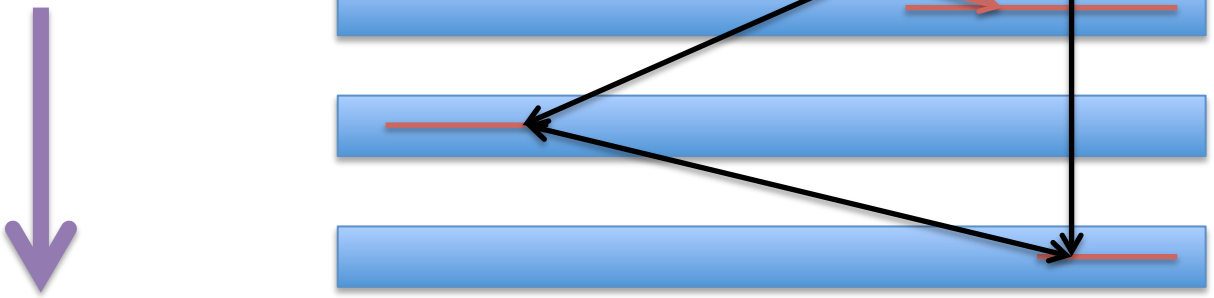
GTTCGAGCCTAATGTCACAA
aattccggATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGGTAACGGAGAGAC

No Match

Sort Pool by Individual BO Score

Given a Pool of Spans, Sort the Spans by Total *Individual* BO Scores. Each mistarget has a score associated with it, the BO score is the sum of these scores.

Highest Individual BO Score

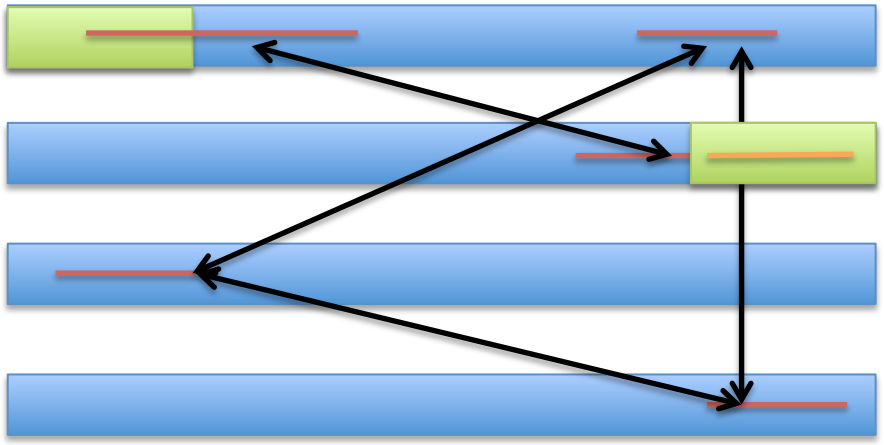


BO Scores

Using Overlap, we can calculate BO for Oligo on the Span for the defined bounds

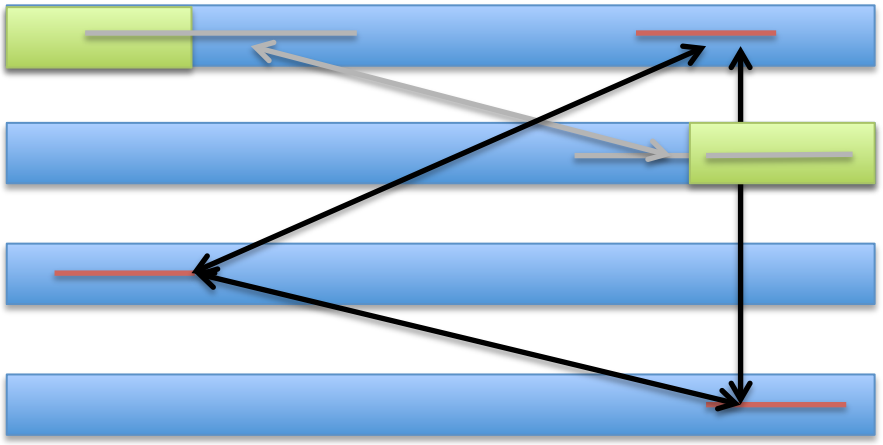
Calc BO

ReCalc BO



BO Scores

Using Overlap, we can calculate BO for Oligo on the Span for the defined bounds



Redo w/4 Oligos

```
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 125) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 250) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 500) );
pool.add(Oligo.InsertionFactory(genome, "aattccgg", 850) );
```

```
> No fixed matches
CCATACCGTCCTAATTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCCTA
CCTGCCTTTGCGTCTTGATACCAATGAAAAACCTATGCACTTTGTACAGG
GTGCCATCGGGTTTCTGAACTCTCAGATAGTGGGGATCCCGGGTAAAGAC
CTATATCTGCGGTCCAACTTAGGCATAAACCTCCATGCTACCTAGTCAGA
CCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG
TTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTAT
TACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA
ATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGG
TAACGGAGAGACTCTGCGGCGTACTTAATTATGCATTTGAAACGCGCCCA
AGTGACGCTAGGCAAGTCAGAGCAGGTTCCCGTGTTAGCTTGAGGGTAAA
CCATACCGTCCTAATTCTTCGGTTATGTTTCCGATGTAGGAGTGAGCCTA
CCTGCCTTTGCGTCTTGATACCAATGAAAAACCTATGCACTTTGTACAGG
GTGCCATCGGGTTTCTGAACTCTCAGATAGTGGGGATCCCGGGTAAAGAC
CTATATCTGCGGTCCAACTTAGGCATAAACCTCCATGCTACCTAGTCAGA
CCCACCCCGCACGGGGTAAATATGGCACGCGTCCGACCTGGTTCCTGGCG
TTCTACGCTGCCACGTGTTCATTAACTGTTGTTTGGTAGCACAAAAGTAT
TACCATGGTCCTAGAAGTTCGGCACAGTTAGTTCGAGCCTAATGTCACAA
ATGACGCAGAACGCCAATGAGTGCCAGACATTAGGTGGAGTTCAGTTCGG
TAACGGAGAGACTCTGCGGCGTACTTAATTATGCATTTGAAACGCGCCCA
AGTGACGCTAGGCAAGTCAGAGCAGGTTCCCGTGTTAGCTTGAGGGTAAA
```