Genome 540: Discussion Section Class - 14

Chengxiang Qiu

HW6

- Due 11:59pm on Sunday, Feb 20
- Assignment: use D-segment algorithm to identify sequence segments with high copy number.
    - Input:
        - File with read start counts at each position along a chromosome (Chromosome\tPosition\tScore)
        - Scoring scheme
    - Output:
        - Number of normal and elevated copy-number segments
        - List of elevated copy-number segments (start, end, score)
        - Annotations for the three segments with the highest scores (look up using UCSC genome browser)
        - Histograms of read-start counts (i.e. number of positions with 0, 1, 2, and >=3 read-starts) for non-elevated and elevated segments

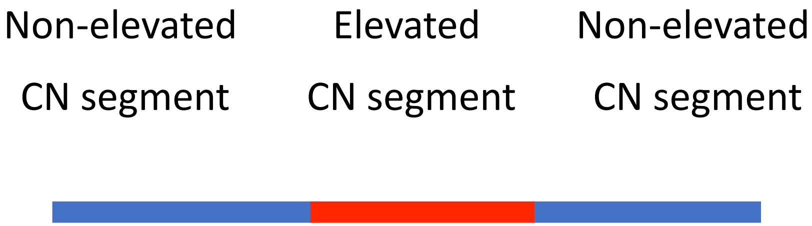
- Input (test data)

```
17   1    0
17   2    0
17   3    0
17   4    0
17   5    2
17   6    0
17   7    0
17   8    0
17   9    0
17   10   0
17   11   0
17   12   1
17   13   0
17   14   3
17   15   0
17   16   0
17   17   2
17   18   0
17   19   0
17   20   0
17   21   5
```

```
0 -0.3464
1 0.2488
2 0.8439
>=3 1.5337
D -33.219
S 33.219
```

- Output

```
Segment Histogram:
Non-Elevated CN Segments=8
Elevated CN Segments=7

Segment List:
48164 48273 66.76
67646 68115 97.51
105528 106003 63.04
106904 107345 41.67
122792 123034 66.56
164376 164665 62.09
165086 166103 225.95

Annotations:

Start: 165086
End: 166103
Description: Something interesting (e.g., "Overlaps with exon5 of the protein coding gene cMyc")

Start: 67646
End: 68115
Description: Something interesting (e.g., "Overlaps with exon5 of the protein coding gene cMyc")

Start: 48164
End: 48273
Description: Something interesting (e.g., "Overlaps with exon5 of the protein coding gene cMyc")

Read start histogram for non-elevated copy-number segments:
0=331908
1=19439
2=4272
>=3=1332

Read start histogram for elevated copy-number segments:
0=1656
1=542
2=352
>=3=499
```

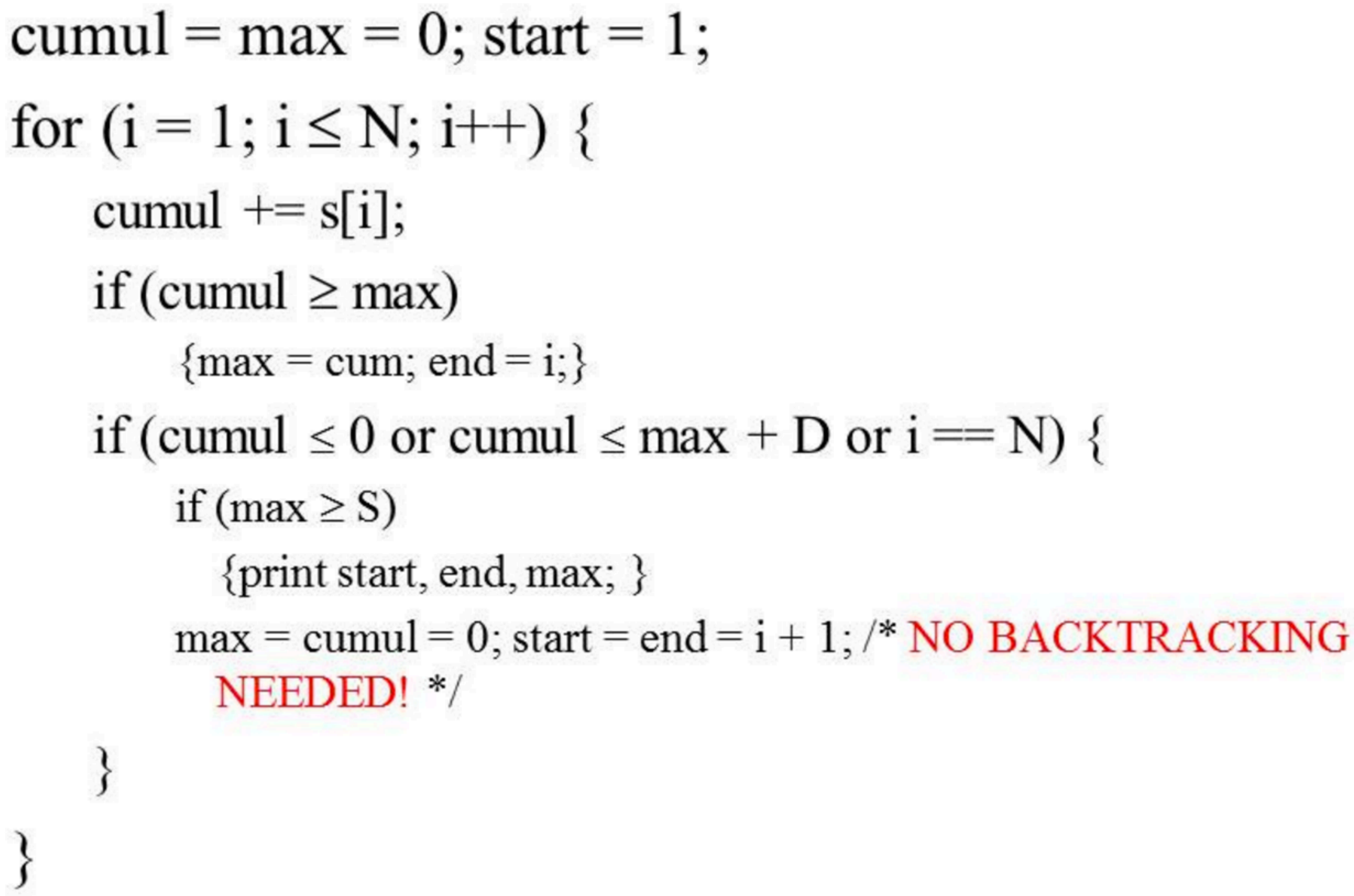


- Input (test data)

17	1	0
17	2	0
17	3	0
17	4	0
17	5	2
17	6	0
17	7	0
17	8	0
17	9	0
17	10	0
17	11	0
17	12	1
17	13	0
17	14	3
17	15	0
17	16	0
17	17	2
17	18	0
17	19	0
17	20	0
17	21	5

0	-0.3464
1	0.2488
2	0.8439
>=3	1.5337
D	-33.219
S	33.219

Pseudo-code for the D-segment algorithm:



```
cumul = max = 0; start = 1;
for (i = 1; $i \le N$; i++) {
    cumul += s[i];
    if (cumul $\ge$ max)
        {max = cum; end = i;}
    if (cumul $\le 0$ or cumul $\le$ max + D or i == N) {
        if (max $\ge$ S)
          {print start, end, max; }
        max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
          NEEDED! */
    }
}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

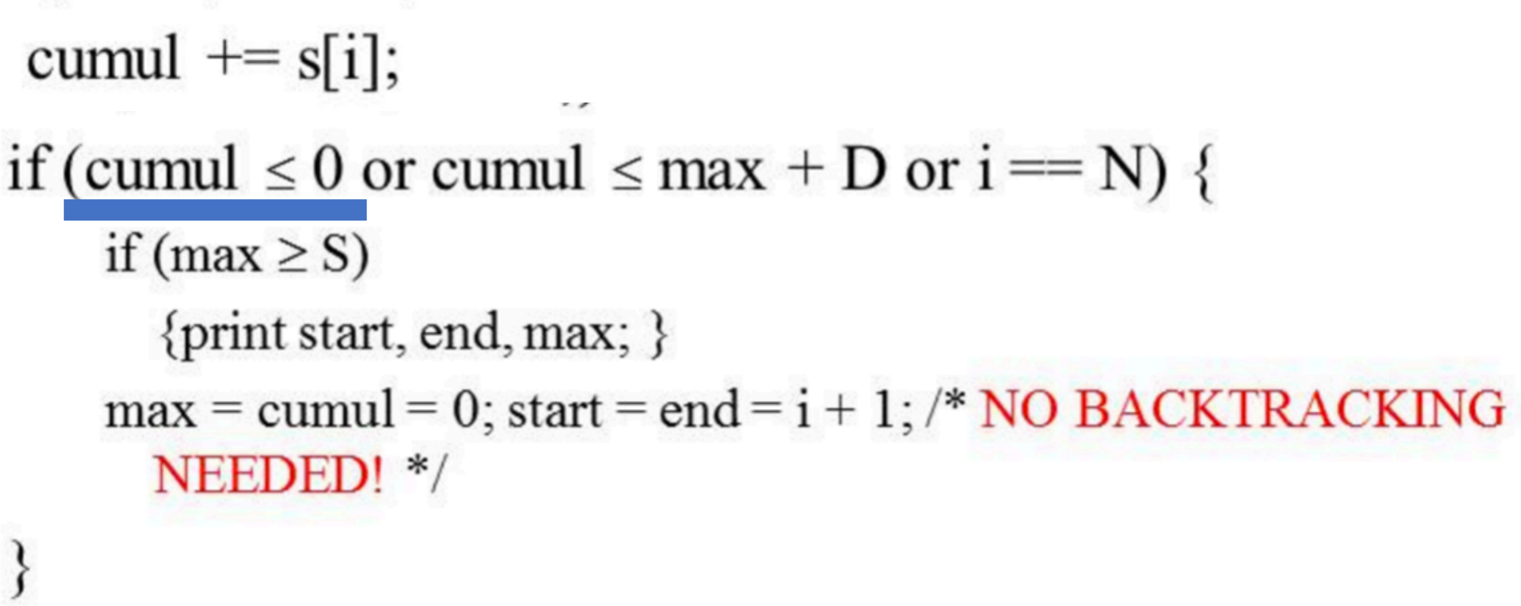
S = 3

max = 0

start = 1

end = 1

cumul = 0



```
cumul += s[i];
if $\underline{(\text{cumul} \le 0}$ or cumul $\le$ max + D or i == N) {
    if (max $\ge$ S)
        {print start, end, max; }
    max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
      NEEDED! */
}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

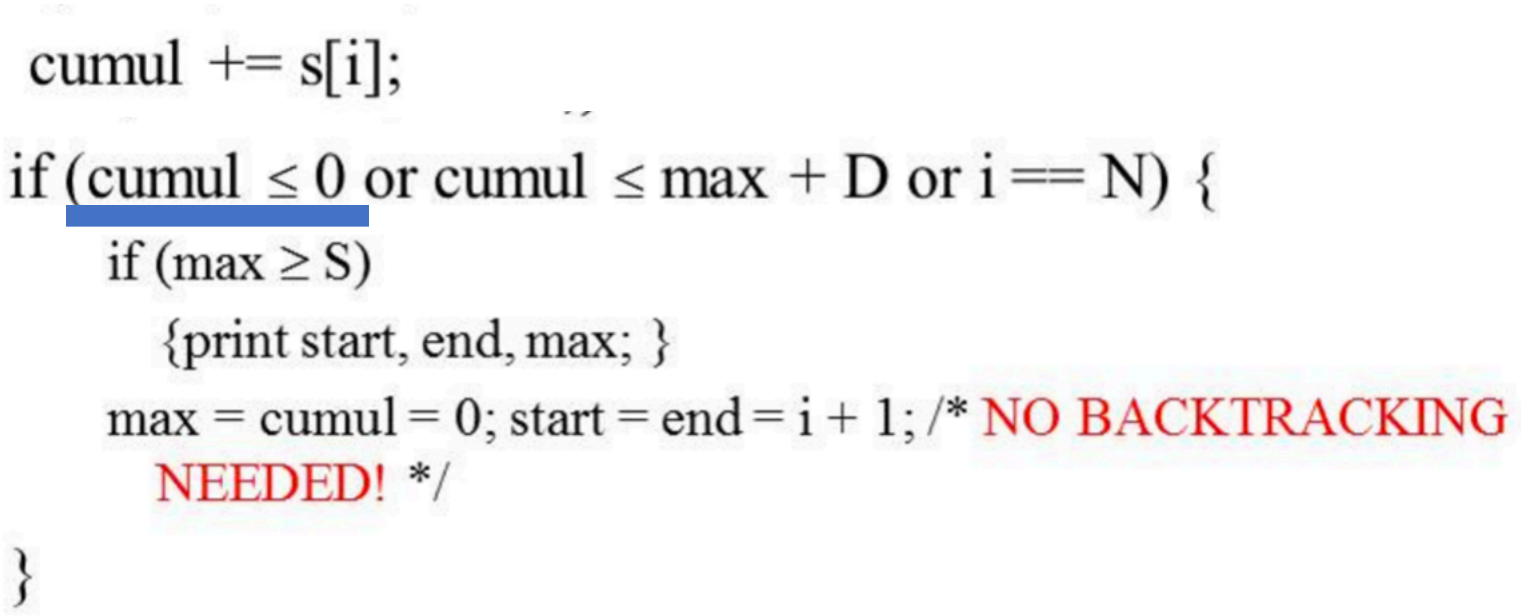
S = 3

max = 0

start = 2

end = 2

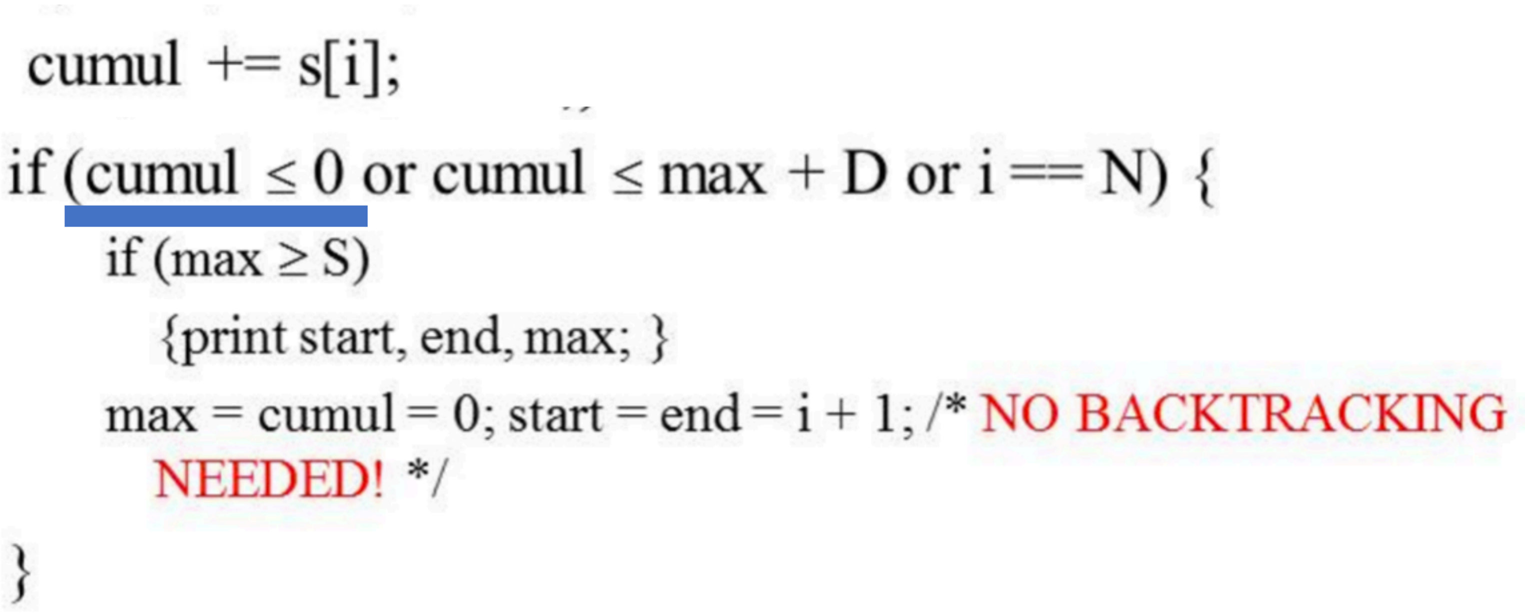
cumul = 0



```
cumul += s[i];
if ($\underline{\text{cumul} \le 0}$ or cumul $\le$ max + D or i == N) {
    if (max $\ge$ S)
        {print start, end, max; }
    max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
        NEEDED! */
}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

```
D = -3
S = 3
max = 0
start = 3
end = 3
cumul = 0
```



```
cumul += s[i];
if (cumul ≤ 0 or cumul ≤ max + D or i == N) {
    if (max ≥ S)
      {print start, end, max; }
    max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
      NEEDED! */
}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

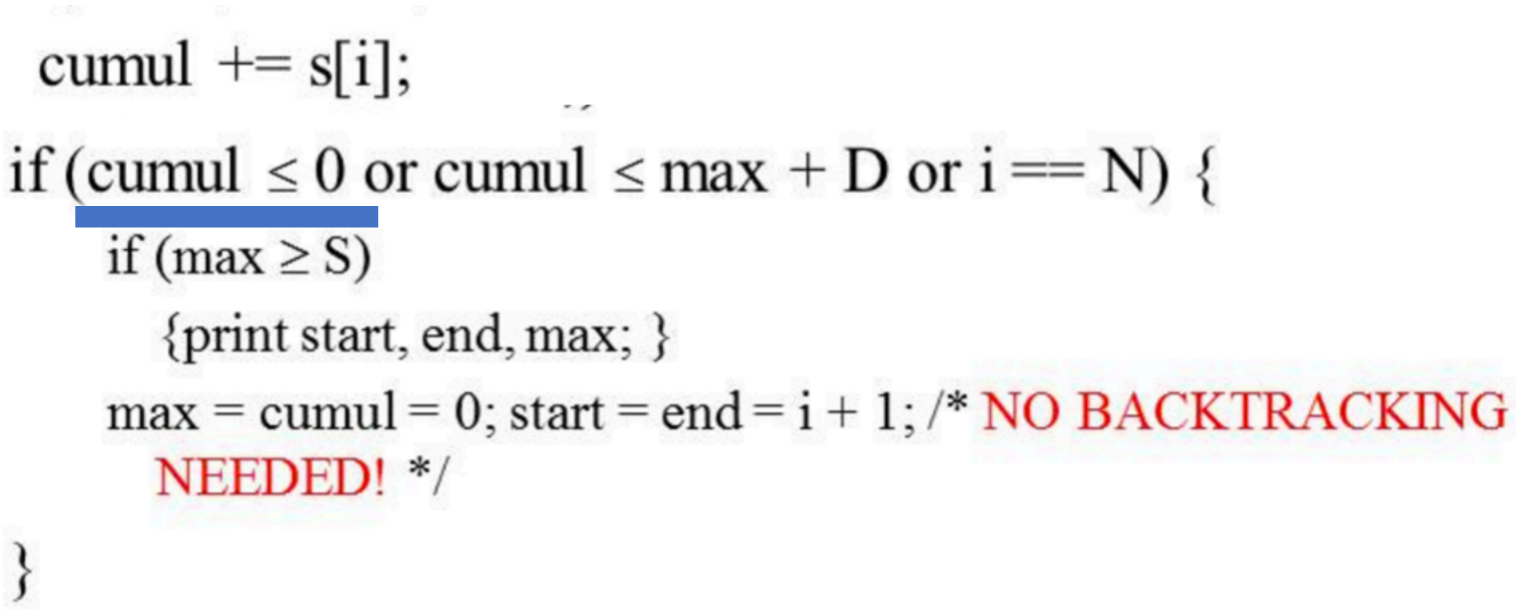
S = 3

max = 0

start = 4

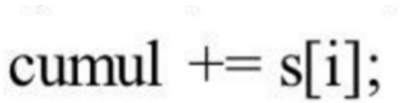
end = 4

cumul = 0



```
cumul += s[i];
if (cumul ≤ 0 or cumul ≤ max + D or i == N) {
    if (max ≥ S)
        {print start, end, max; }
    max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
        NEEDED! */
}
```

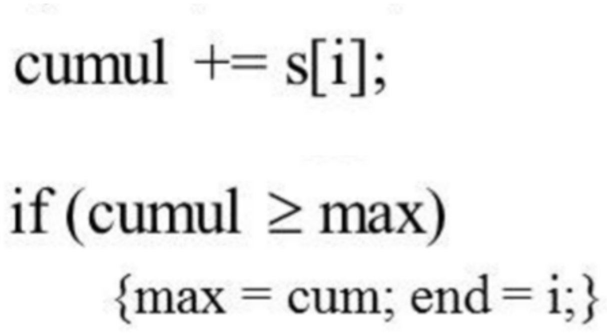
position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5



```
cumul += s[i];
```

```
D = -3
S = 3
max = 0.52
start = 5
end = 5
cumul = 0.52
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

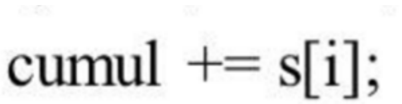


```
cumul += s[i];

if (cumul ≥ max)
    {max = cum; end = i;}
```

```
D = -3
S = 3
max = 1.62
start = 5
end = 6
cumul = 1.62
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

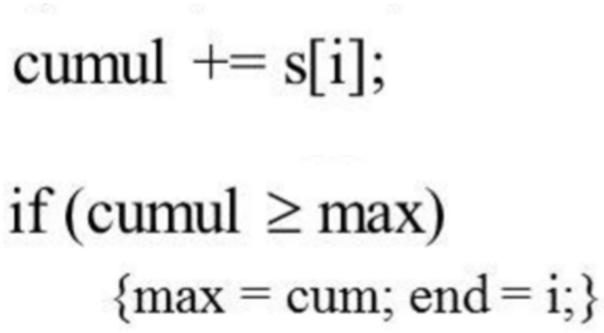
```
D = -3
S = 3
max = 1.62
start = 5
end = 6
cumul = 1.12
```



```
cumul += s[i];
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

```
D = -3
S = 3
max = 2.82
start = 5
end = 8
cumul = 2.82
```



```
cumul += s[i];

if (cumul ≥ max)
    {max = cum; end = i;}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

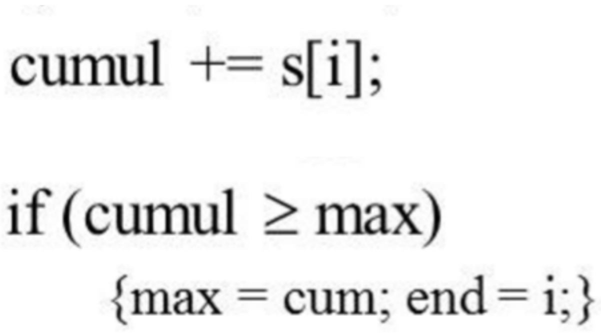
S = 3

max = 3.34

start = 5

end = 9

cumul = 3.34



```
cumul += s[i];

if (cumul ≥ max)
    {max = cum; end = i;}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

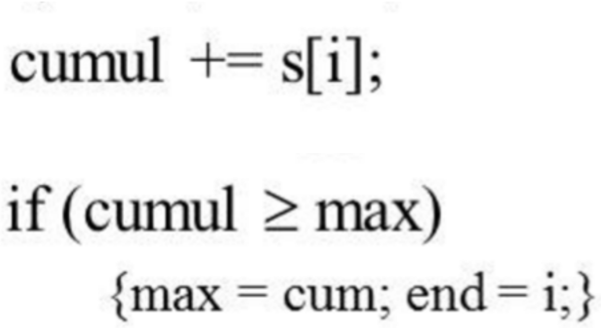
S = 3

max = 4.44

start = 5

end = 10

cumul = 4.44



```
cumul += s[i];

if (cumul ≥ max)
    {max = cum; end = i;}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

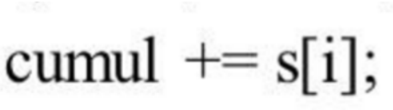
S = 3

max = 4.44

start = 5

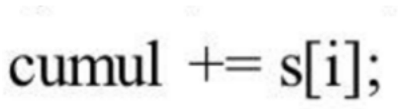
end = 10

cumul = 3.94



```
cumul += s[i];
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

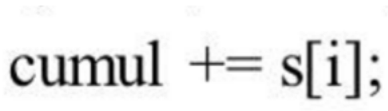


```
cumul += s[i];
```

```
D = -3
S = 3
max = 4.44
start = 5
end = 10
cumul = 3.44
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

```
D = -3
S = 3
max = 4.44
start = 5
end = 10
cumul = 2.94
```



```
cumul += s[i];
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D = -3

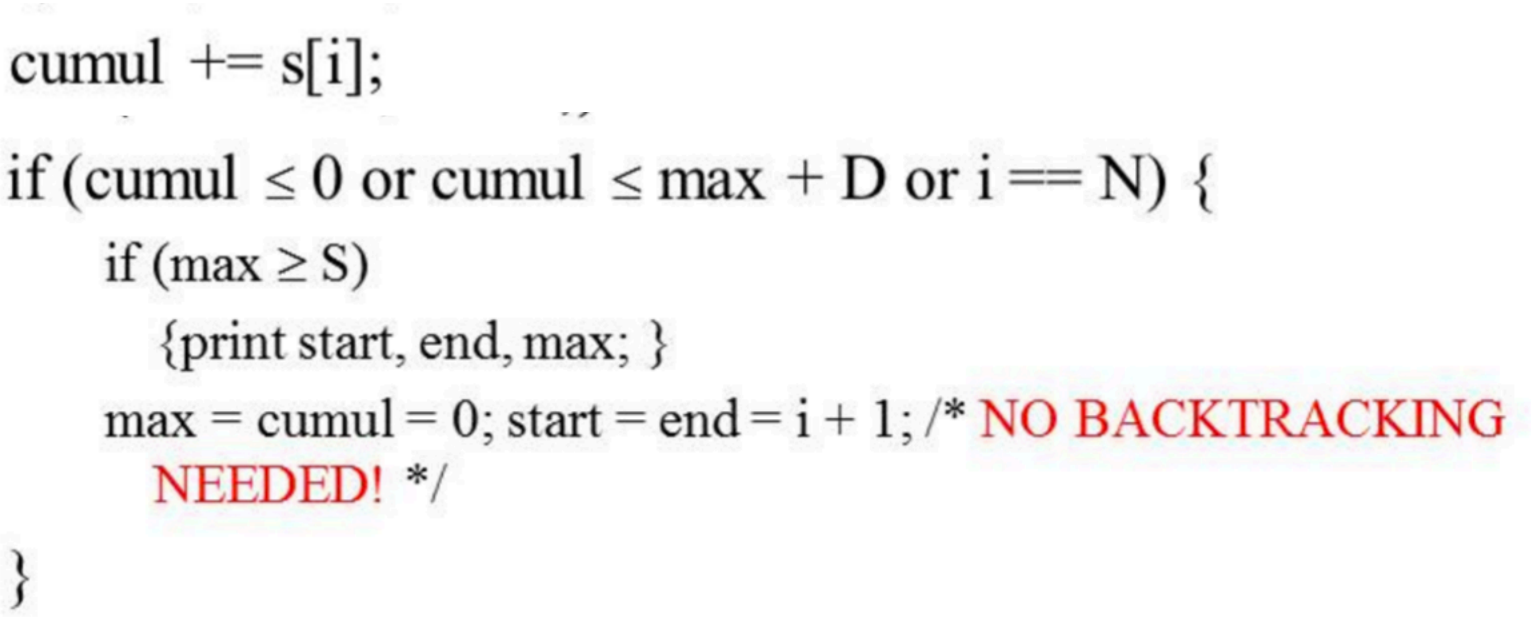
S = 3

max = 4.44

start = 5

end = 10

cumul = 2.44



```
cumul += s[i];
if (cumul $\leq$ 0 or cumul $\leq$ max + D or i == N) {
    if (max $\geq$ S)
      {print start, end, max; }
    max = cumul = 0; start = end = i + 1; /* NO BACKTRACKING
      NEEDED! */
}
```

position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# read starts	0	0	0	0	1	2	0	4	1	2	0	0	0	0
score	-0.5	-0.5	-0.5	-0.5	0.52	1.1	-0.5	1.7	0.52	1.1	-0.5	-0.5	-0.5	-0.5

D-segment: 5, 10, 4.44

(start, end, max)

D = -3

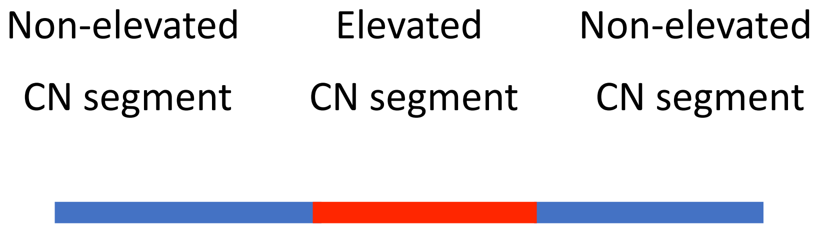
S = 3

max = 4.44

start = 5

end = 10

cumul = 2.44


Non-elevated CN segment

Elevated CN segment

Non-elevated CN segment

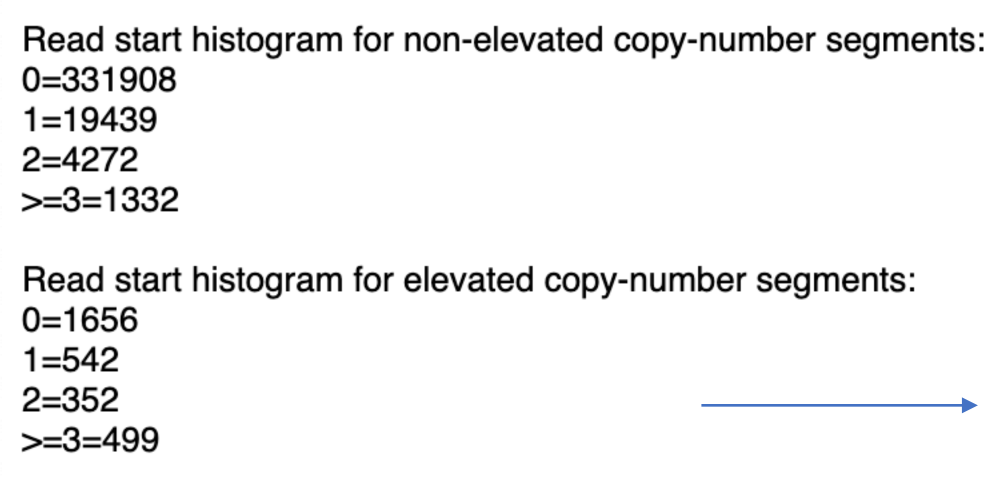
HW7: D-segments Revisited

- **Same input data** as for HW6 (file of read-start counts for chromosome 16)
- **Computing a new scoring scheme** for the read-start bins (0, 1, 2, and >=3)
- **S = -D = 5**

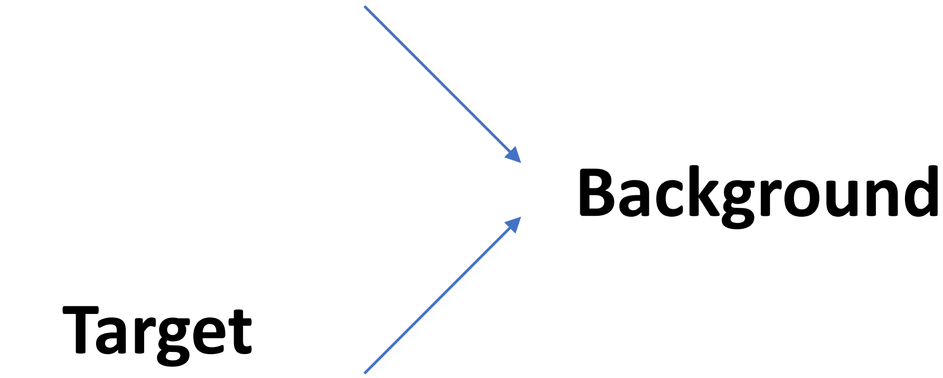
HW7: D-segments Revisited

In the real data, there are 8,422,401 sites corresponding to sites with 'N' in the reference genome and read alignments cannot start at an 'N'.

Output of HW6 (testing data)



```
Read start histogram for non-elevated copy-number segments:
0=331908
1=19439
2=4272
>=3=1332

Read start histogram for elevated copy-number segments:
0=1656
1=542
2=352
>=3=499
```

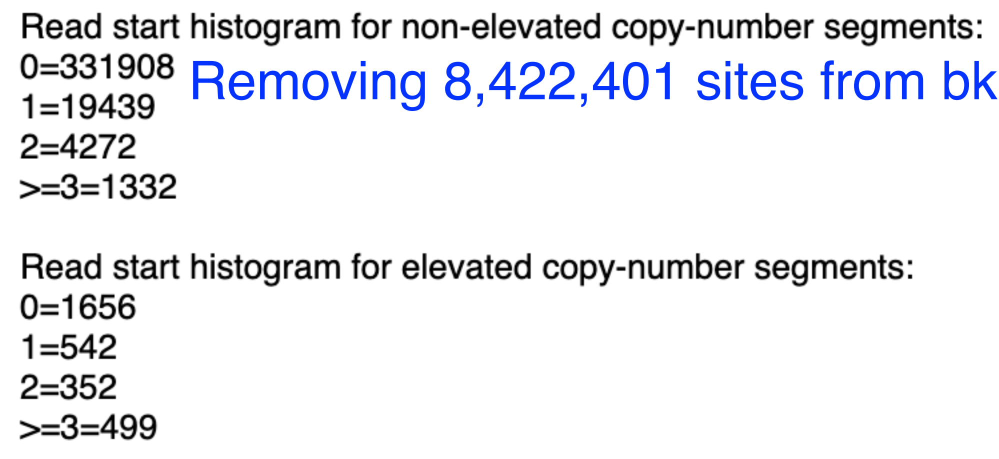


Target

Background

HW7: D-segments Revisited

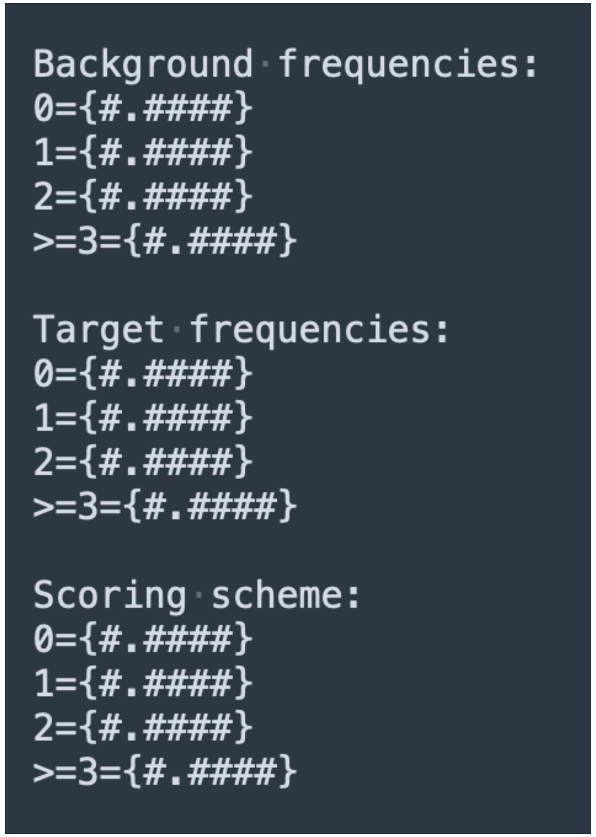
1. Create a scoring scheme (for each count value 0, 1, 2, 3) based on the background and target frequencies, using LLRs with base 2 logarithms



```
Read start histogram for non-elevated copy-number segments:
0=331908
1=19439
2=4272
>=3=1332

Read start histogram for elevated copy-number segments:
0=1656
1=542
2=352
>=3=499
```

Removing 8,422,401 sites from bkgd[0]

log2(freq_target/freq_background)

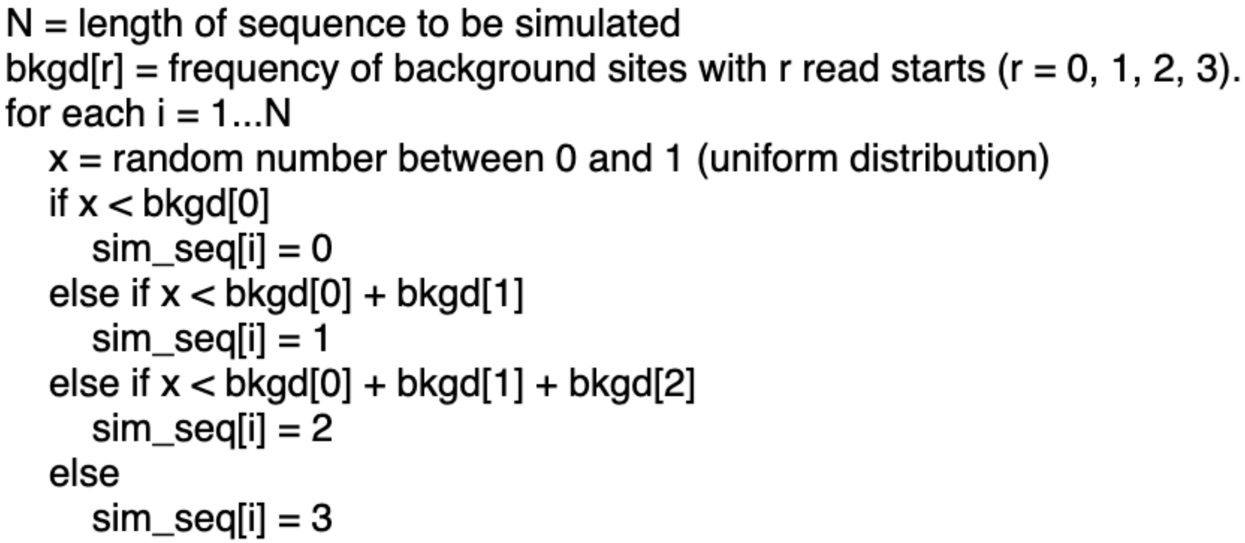


```
Background frequencies:
0={#.####}
1={#.####}
2={#.####}
>=3={#.####}

Target frequencies:
0={#.####}
1={#.####}
2={#.####}
>=3={#.####}

Scoring scheme:
0={#.####}
1={#.####}
2={#.####}
>=3={#.####}
```

HW7: D-segments Revisited

2. Write a program that uses the background frequencies above to simulate a sequence of read start counts. The length of this sequence should be the total length of the chromosome used in HW6 minus the number of N's (as given above).

Background



```
N = length of sequence to be simulated
bkgd[r] = frequency of background sites with r read starts (r = 0, 1, 2, 3).
for each i = 1...N
    x = random number between 0 and 1 (uniform distribution)
    if x < bkgd[0]
        sim_seq[i] = 0
    else if x < bkgd[0] + bkgd[1]
        sim_seq[i] = 1
    else if x < bkgd[0] + bkgd[1] + bkgd[2]
        sim_seq[i] = 2
    else
        sim_seq[i] = 3
```

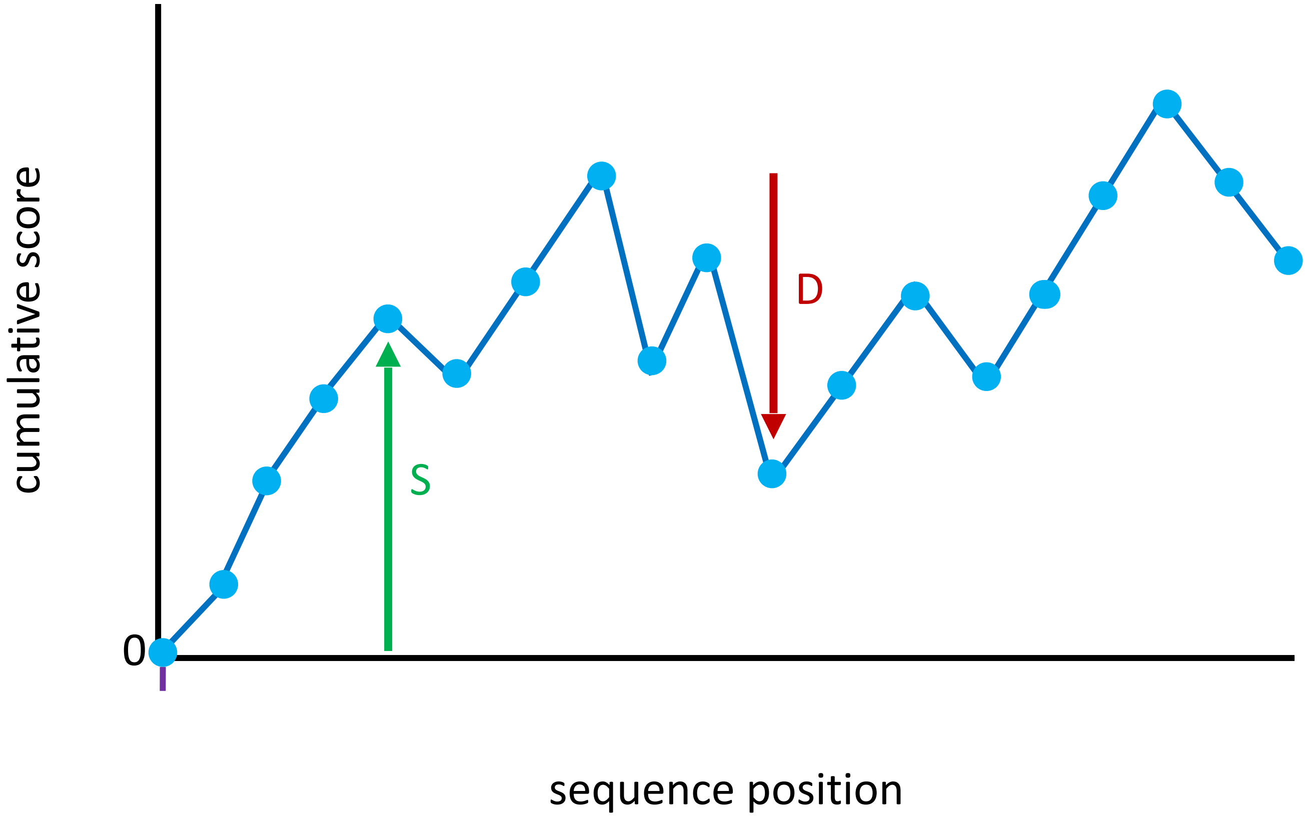
HW7: D-segments Revisited

3. Run your maximal D-segment algorithm on the simulated count sequence with S = -D = 5 and the above scoring scheme. Report a list of pairs, giving for each integer score s = 5, ... 30 the number N_seg(s) of D-segments with score >= s.

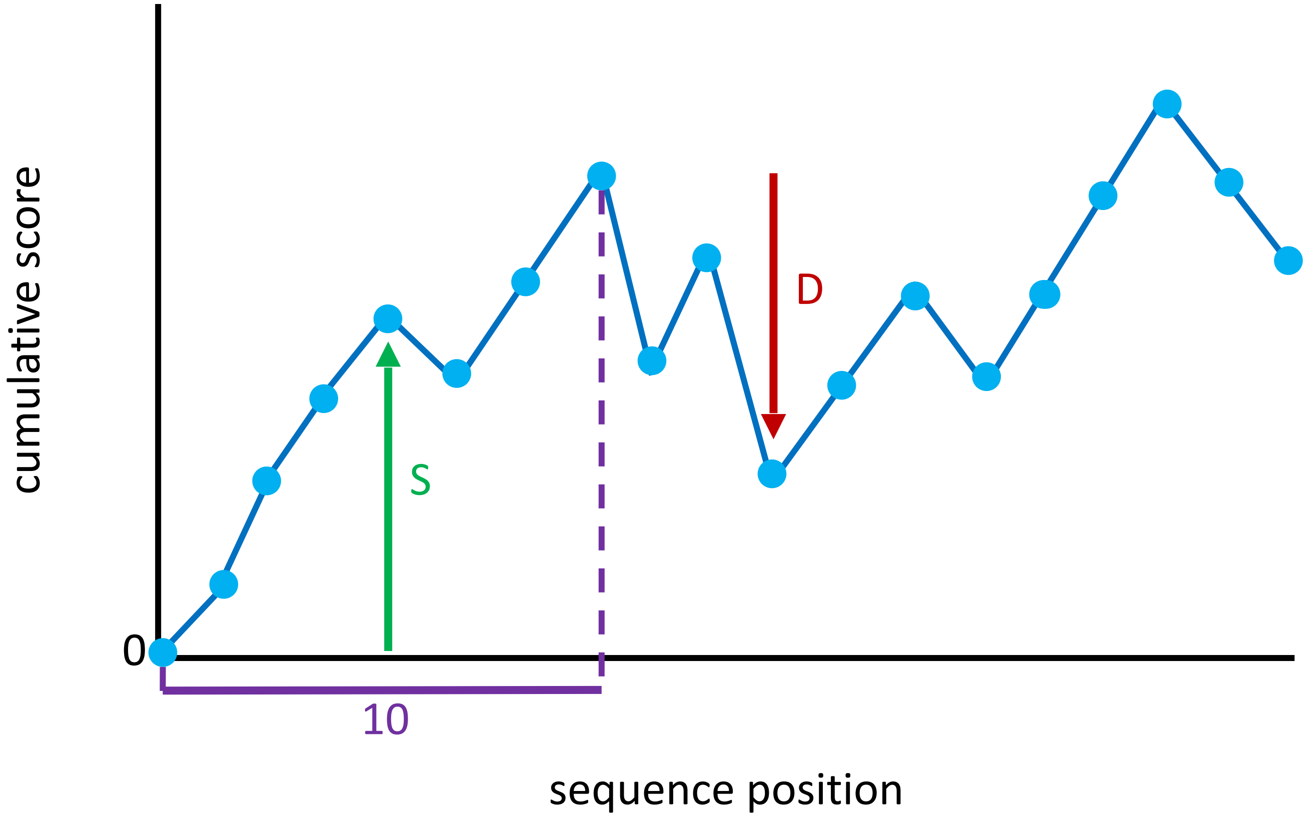
4. Run your maximal D-segment algorithm on the 'real data' sequence of read starts used in assignment 6 with the above S and D values, scoring scheme, and list output.

We care about

```
{# of segments with score >= S}
```



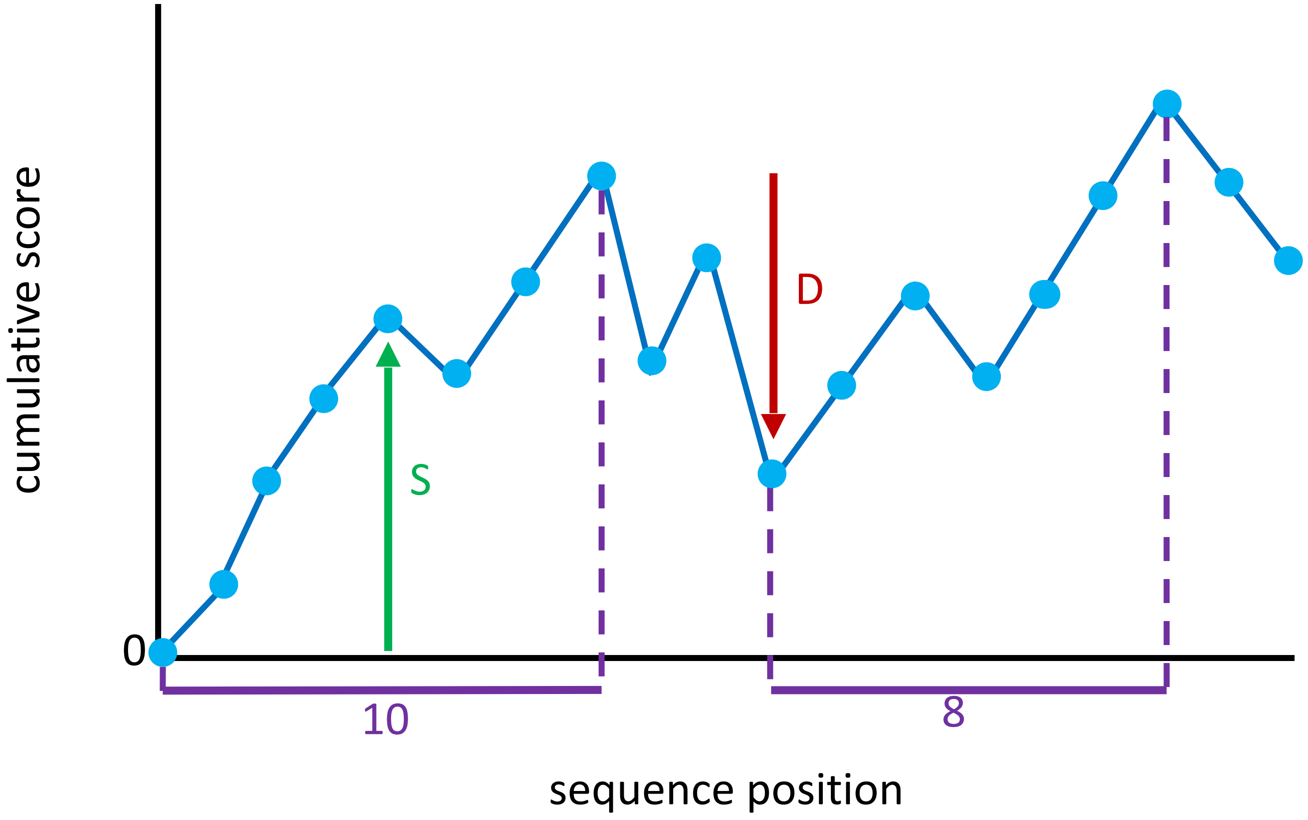
```
Simulated data:
5  0
6  0
7  0
8  0
9  0
10  0
```



We care about

{# of segments with score >= S}

```
Simulated data:
5 1
6 1
7 1
8 1
9 1
10 1
```



We care about

`{# of segments with score >= S}`

Simulated data:

5	2
6	2
7	2
8	2
9	1
10	1

HW7: D-segments Revisited

- Output:
  - Two lists of pairs, one for the original ‘real’ data and another for the simulated data. Each row should contain:
    - S-value
    - Number of D-segments found
  - A list of ratios based on the simulated data:
    - Label each row N_seg(S_i)/N_seg(S_{i+1})
    - Ratio of #D-seg(S_i)/#D-seg(S_{i+1}) rounded to 2 dec.
    - If there is a 0 in the denominator of your ratio, print -1
  - Brief written answers to the questions posed in the assignment text

```
Real data:
5 {# of segments with score >= 5}
6 {# of segments with score >= 6}
7 {# of segments with score >= 7}
.
.

Simulated data:
5 {# of segments with score >= 5}
6 {# of segments with score >= 6}
7 {# of segments with score >= 7}
.
.

Ratios of simulated data:
N_seg(5)/N_seg(6){ratio}
N_seg(6)/N_seg(7) {ratio}
N_seg(7)/N_seg(8) {ratio}
.
.
```

HW7: Questions?