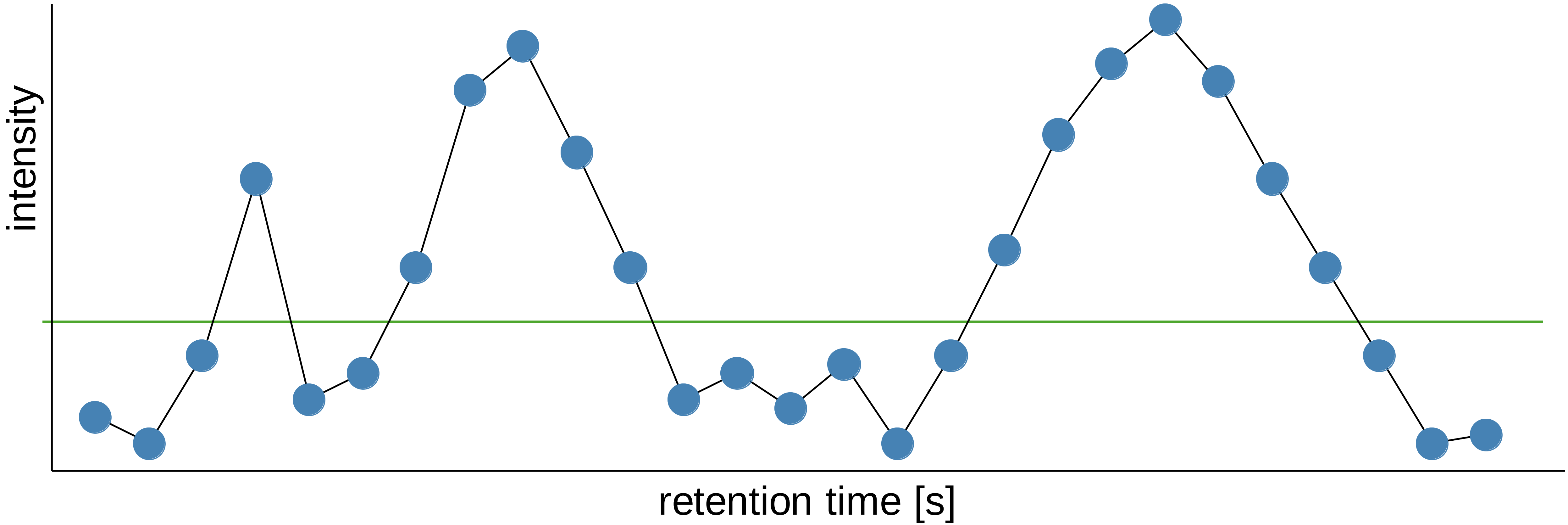
Pseudo code algorithm

min_intensity_threshold = green

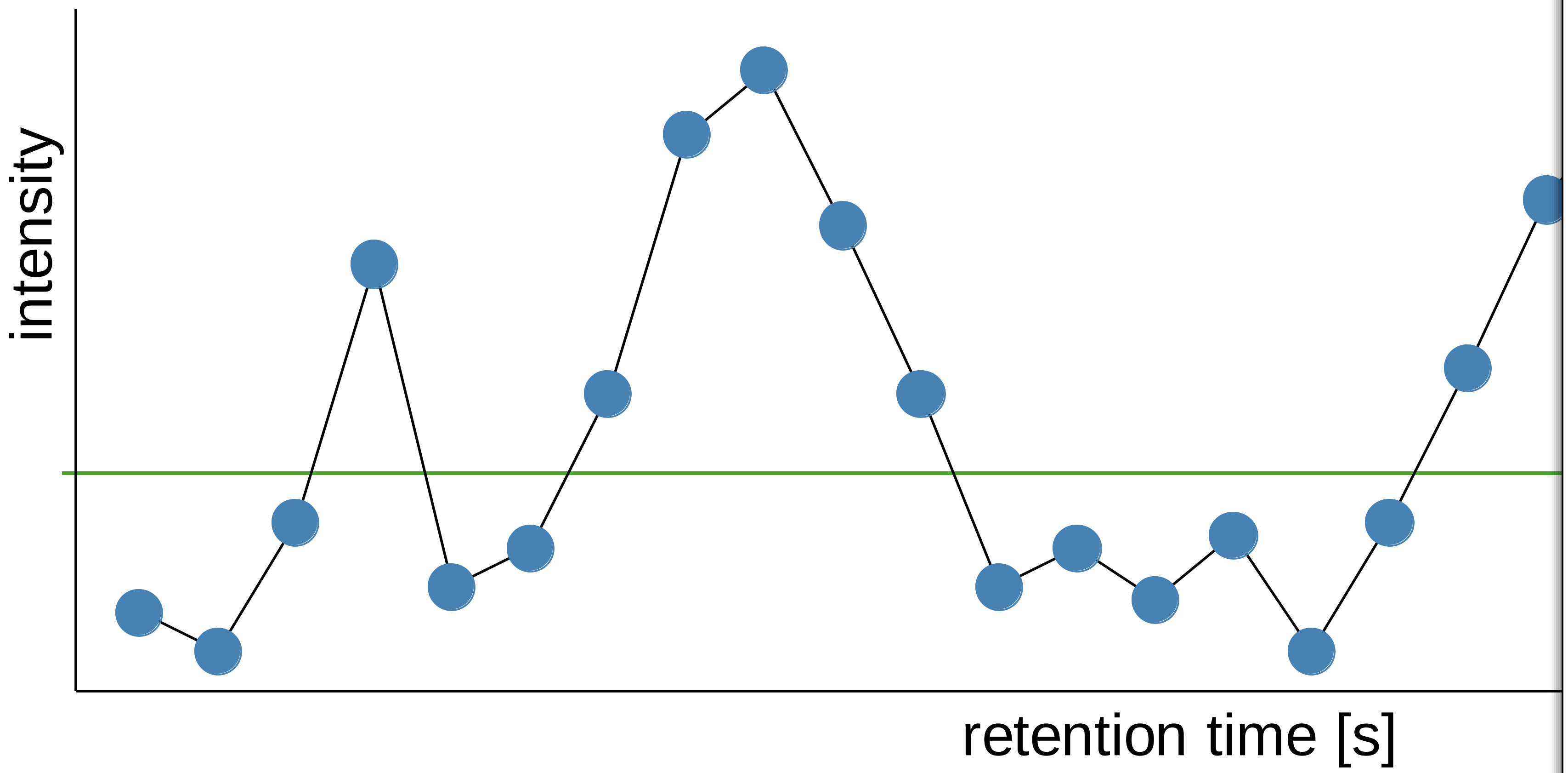
min_points = 4



Pseudo code algorithm

min_intensity_threshold = green

min_points = 4

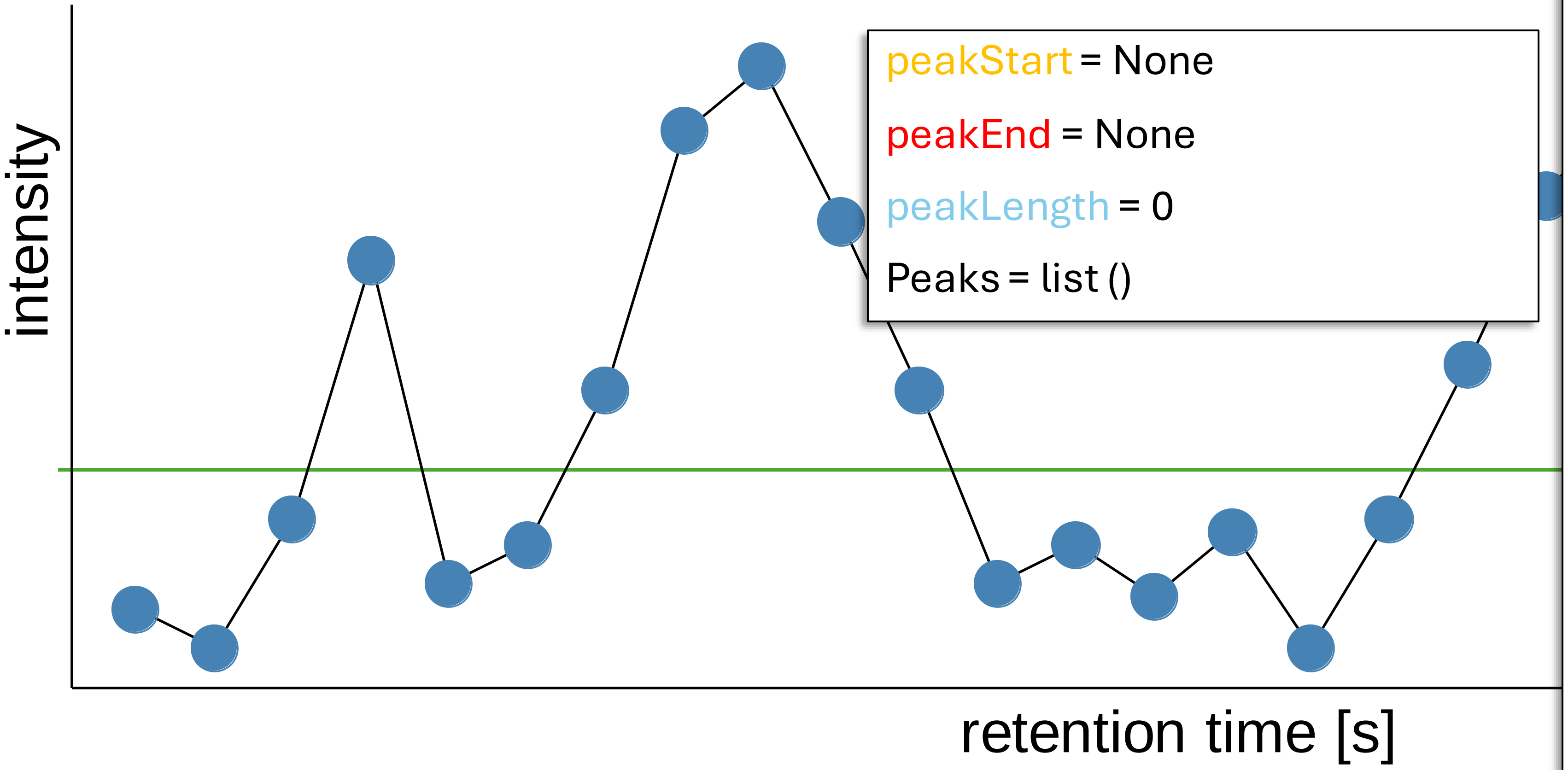


```
Algorithm  FindPeaksWithThreshold
(signal_vector,  min_int_threshold,  min_points)
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

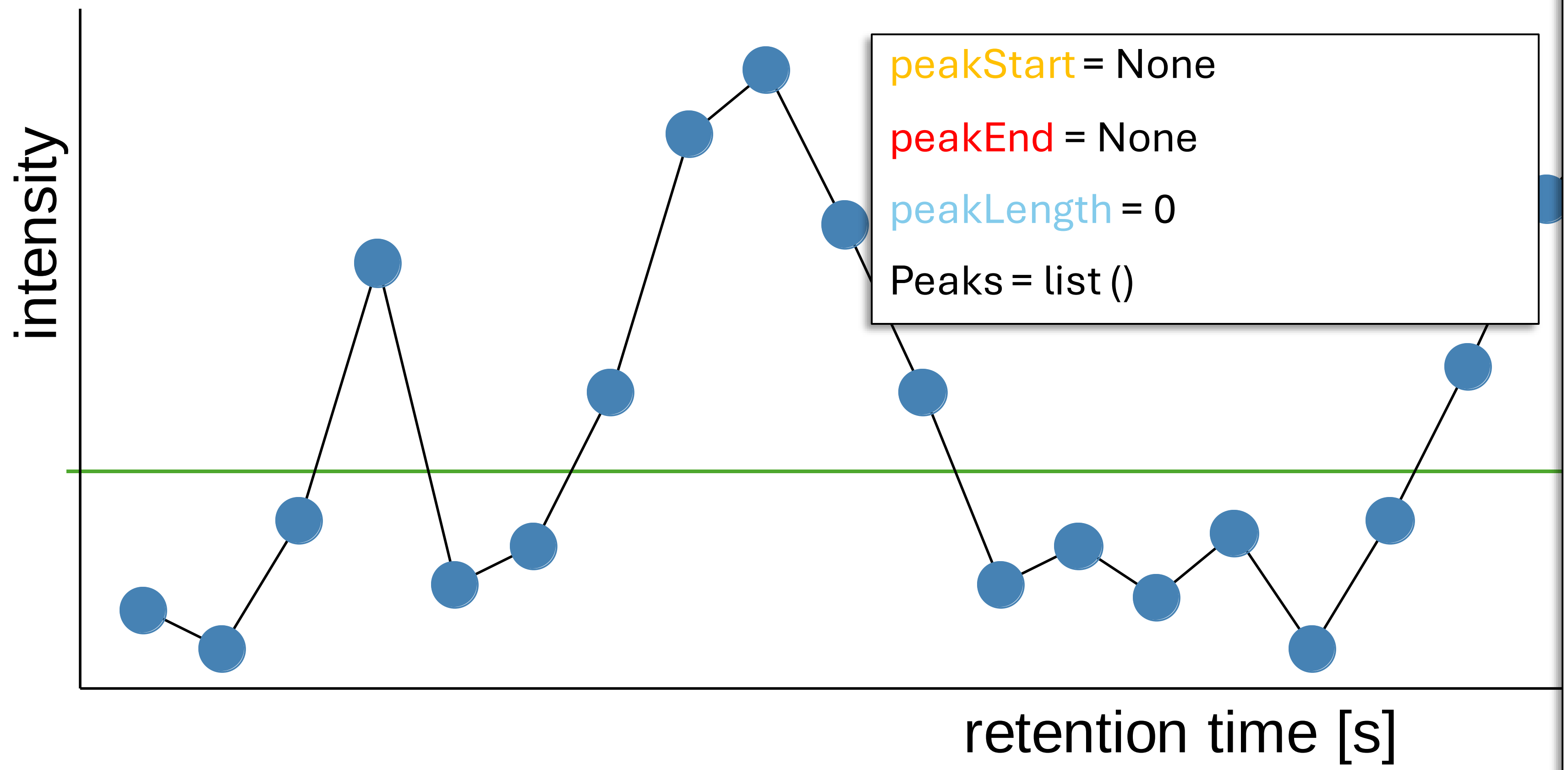
peakLength = 0

Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list

    peakStart = None

    peakEnd = None

    peakLength = 0
```

Pseudo code algorithm

min_intensity_threshold = green
min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

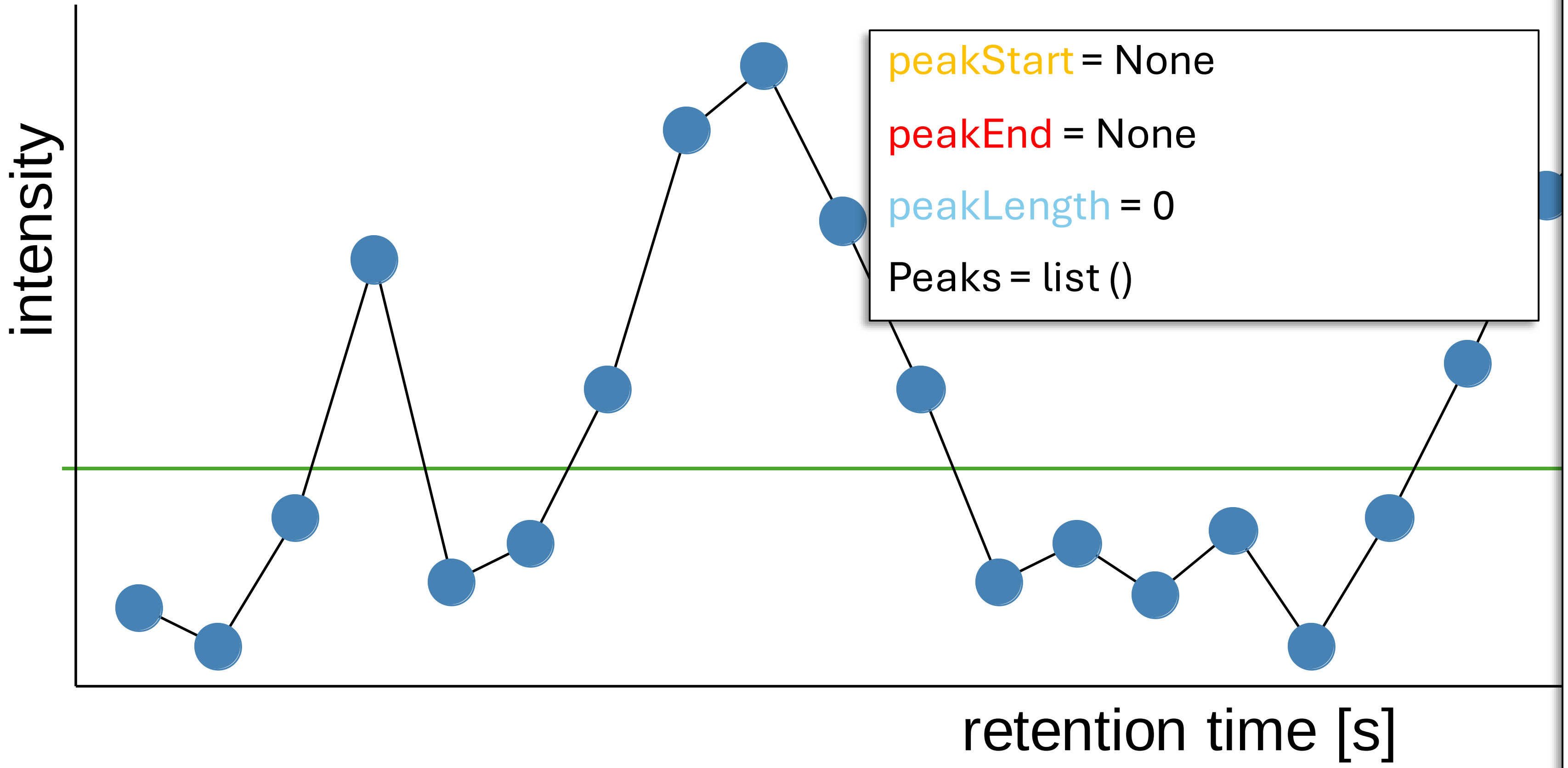
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

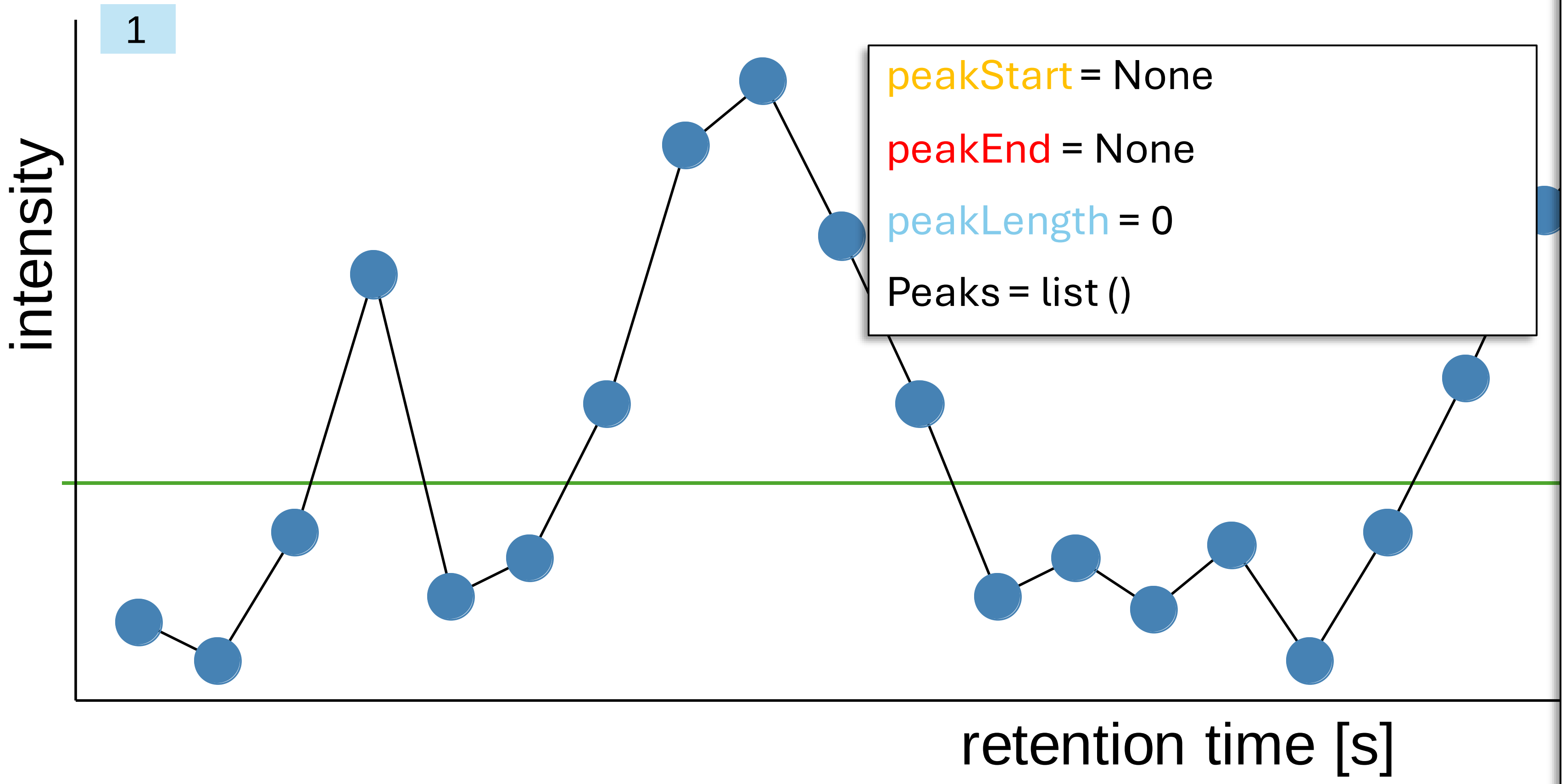
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

peaks = empty list
peakStart = None
peakEnd = None
peakLength = 0

For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
        If peakStart is None:
            peakStart = index - 1 if index > 1 else 1
        peakEnd = index
        peakLength = peakLength + 1
    Else if peakStart is not None:
        If peakLength >= min_points:
            Append [peakStart, peakEnd] to peaks
        peakStart = None
        peakEnd = None
        peakLength = 0

Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

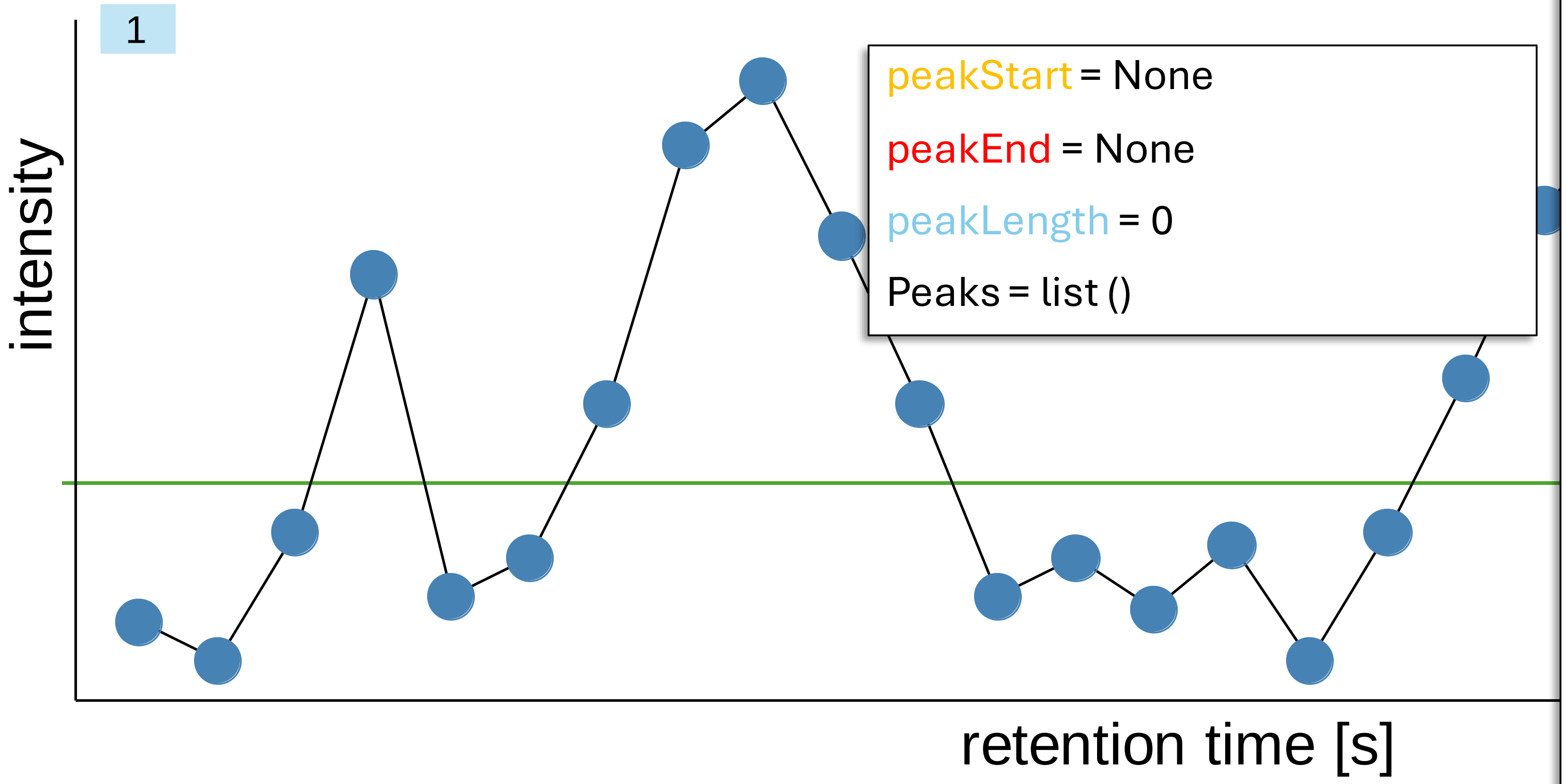
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

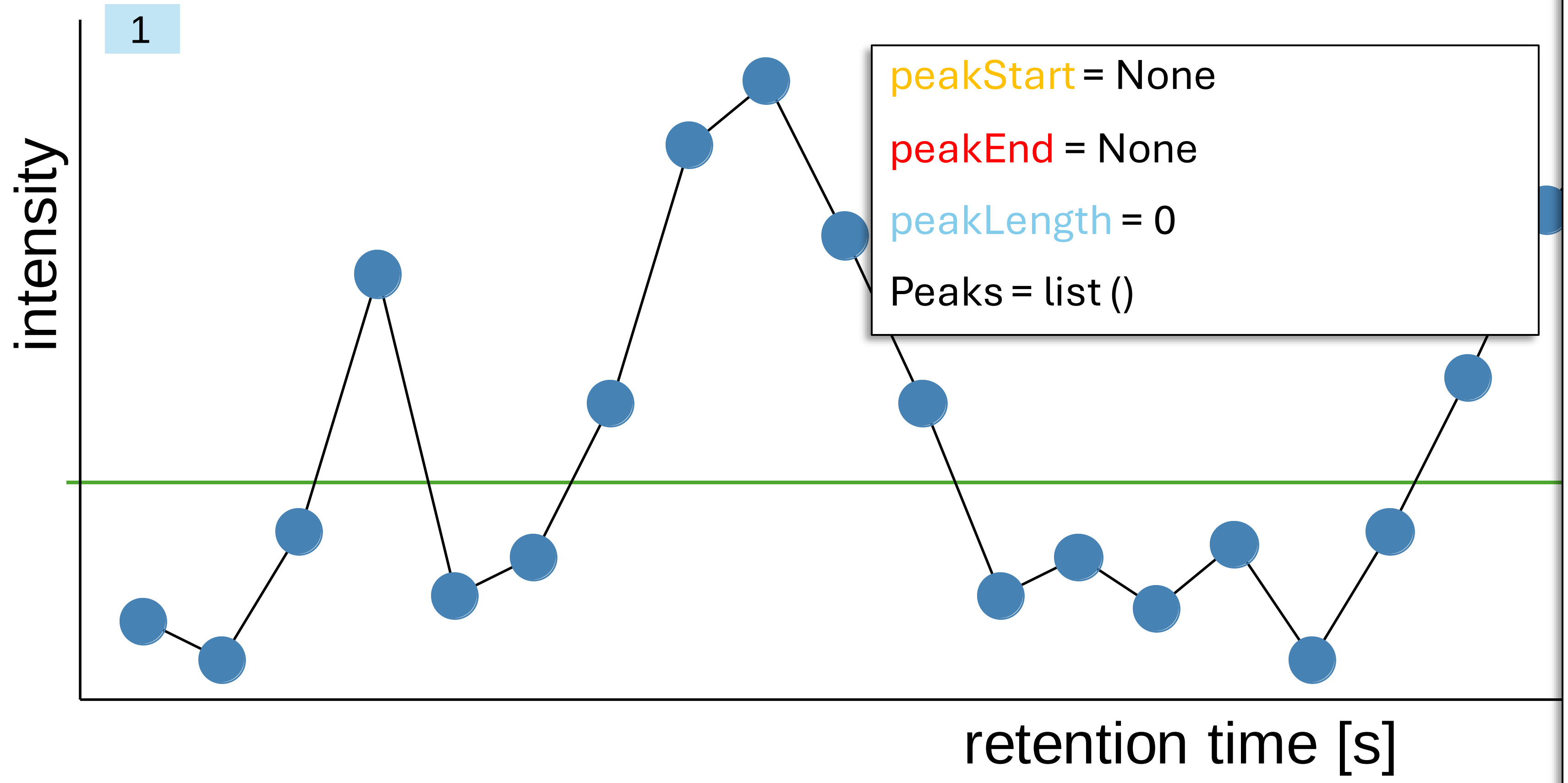
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

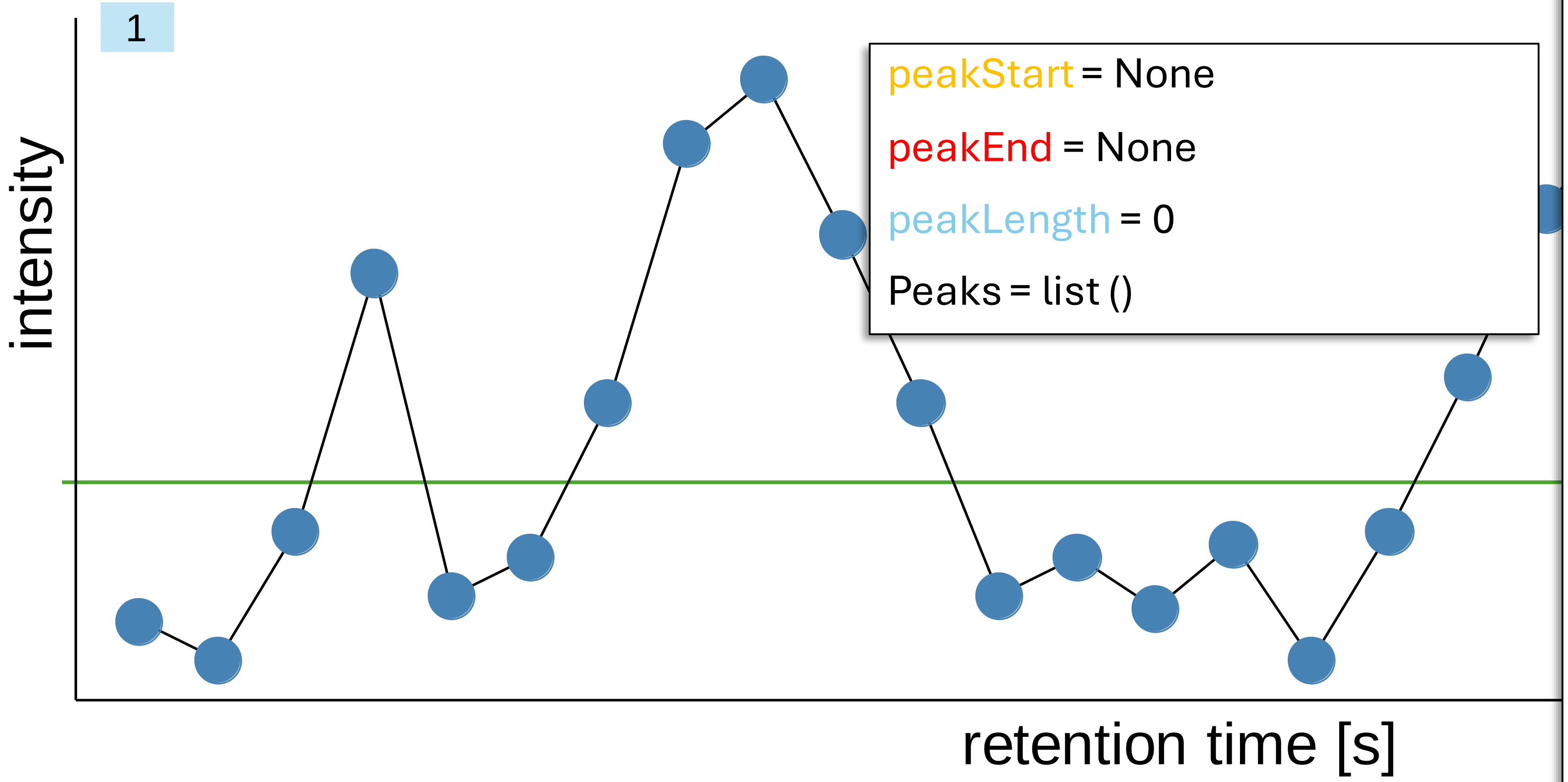
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

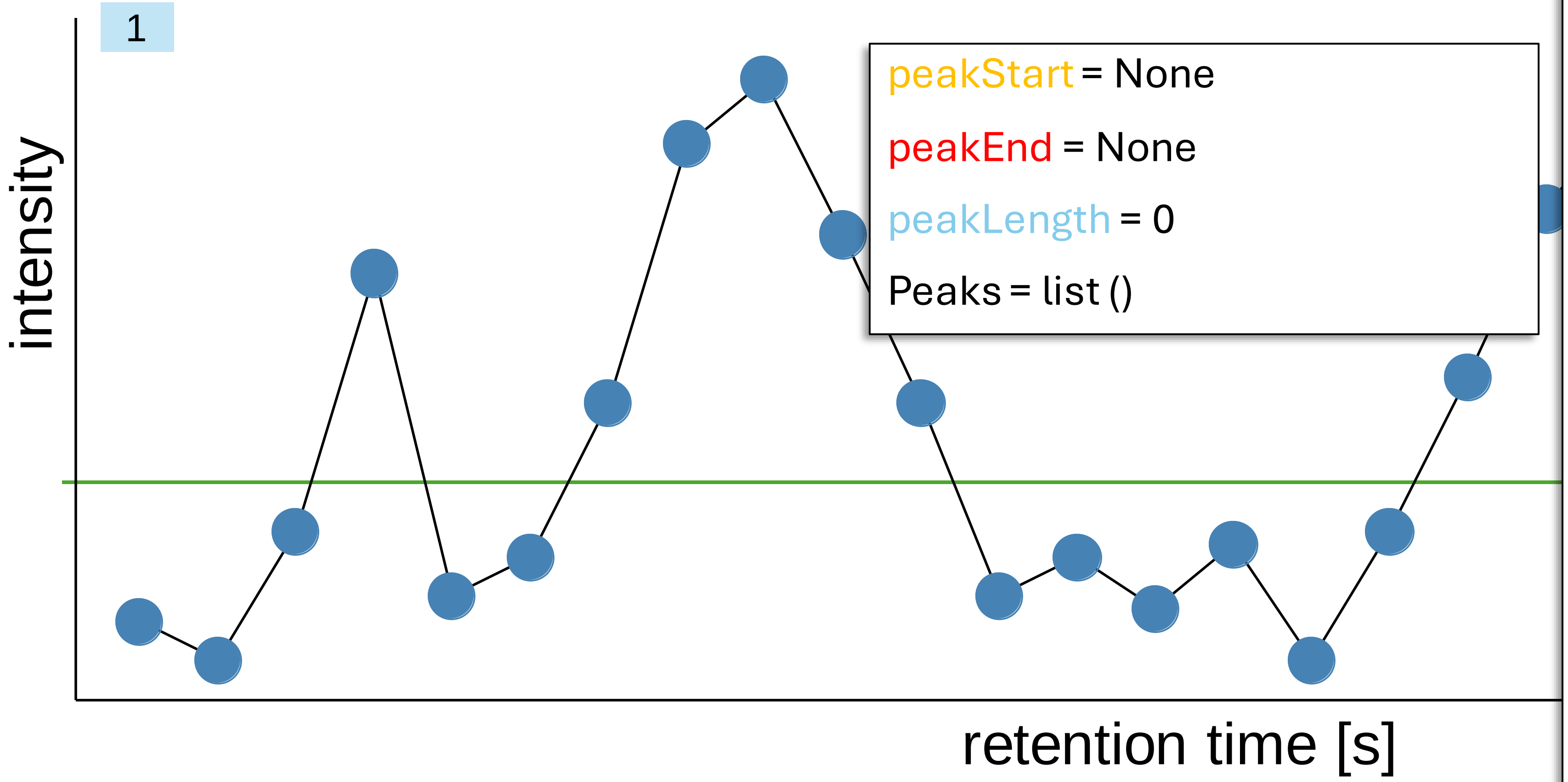
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :   X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

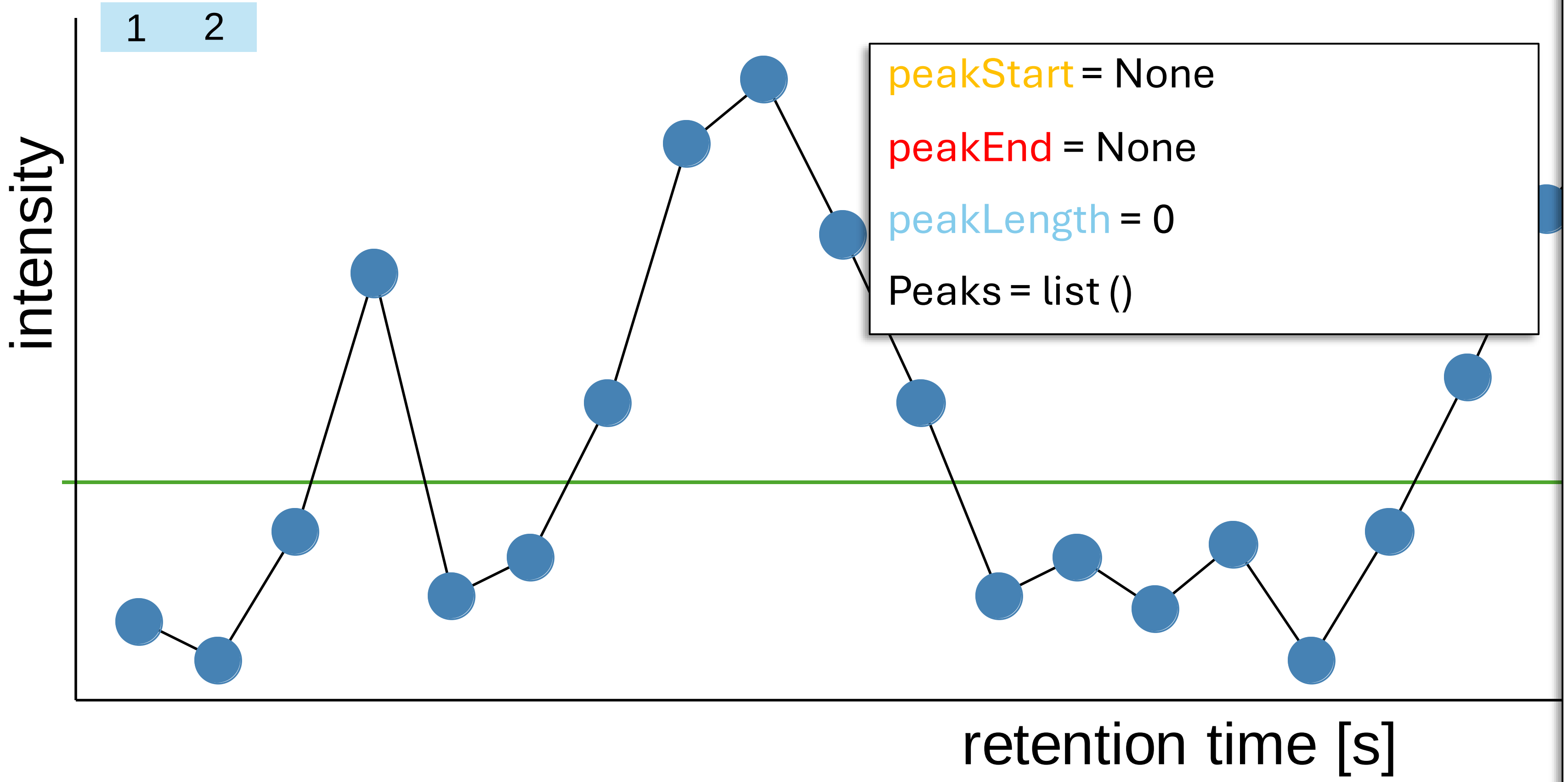
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :  X
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:  X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

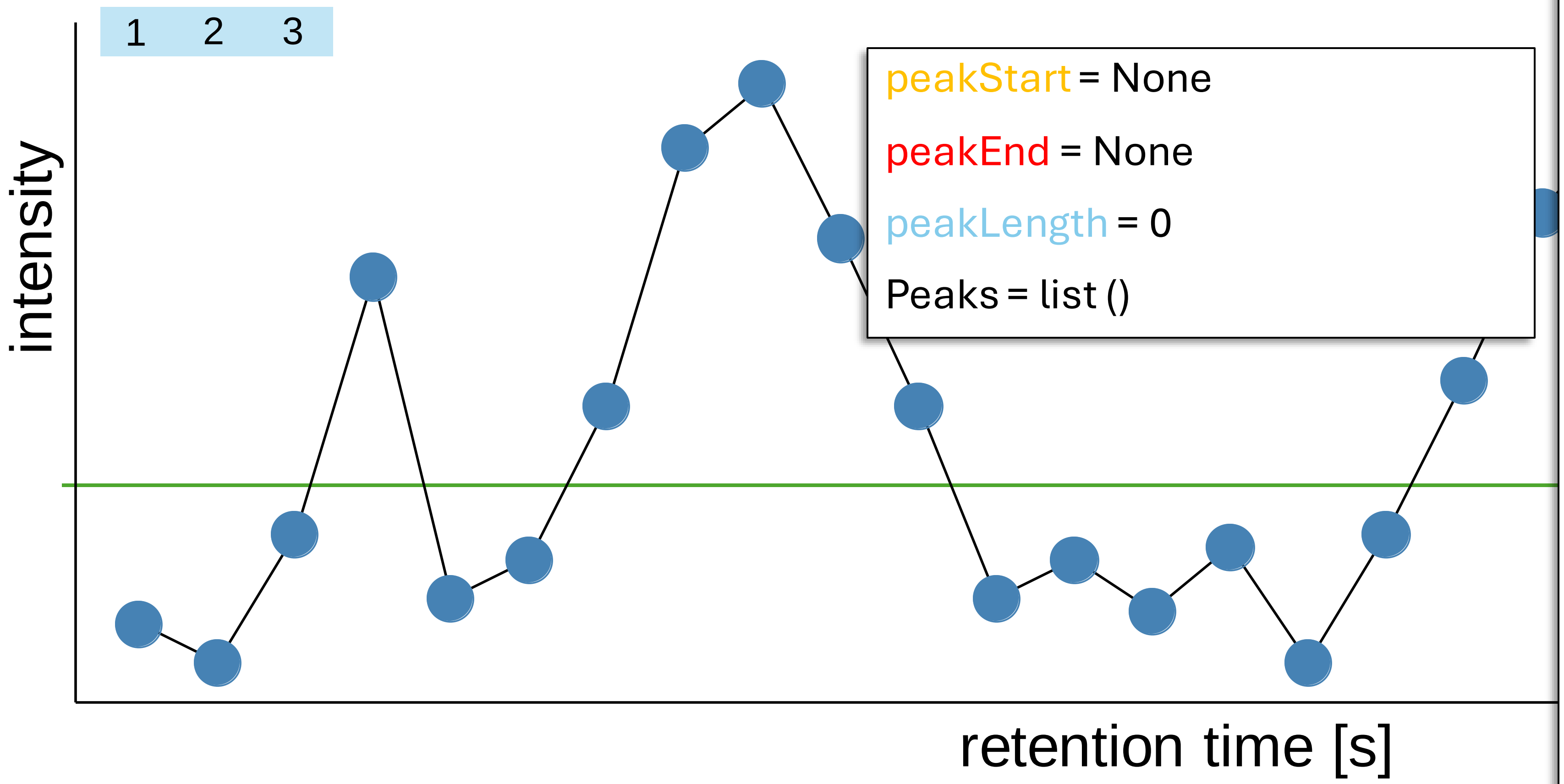
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :   X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:   X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

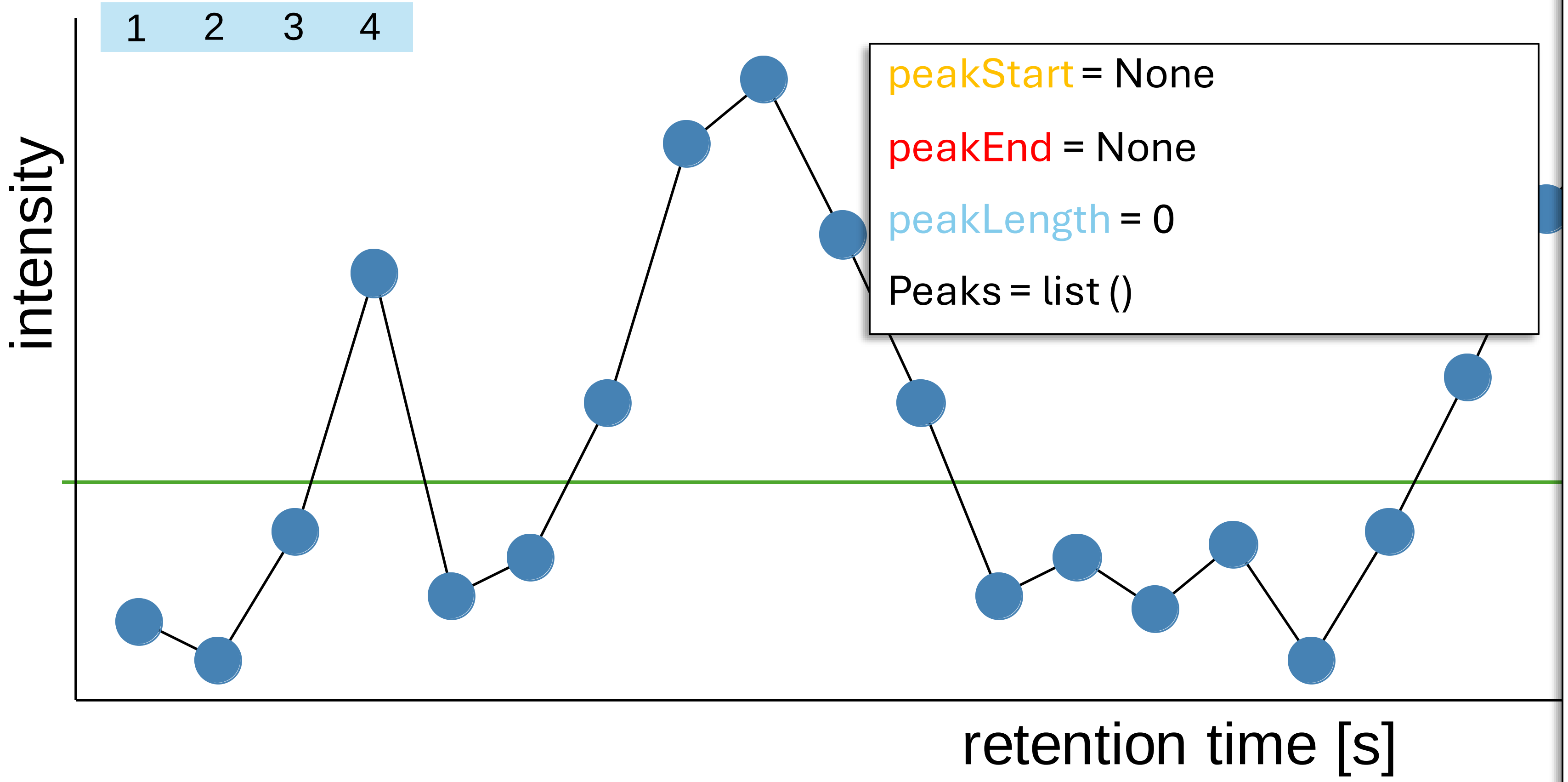
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :   X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:   X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

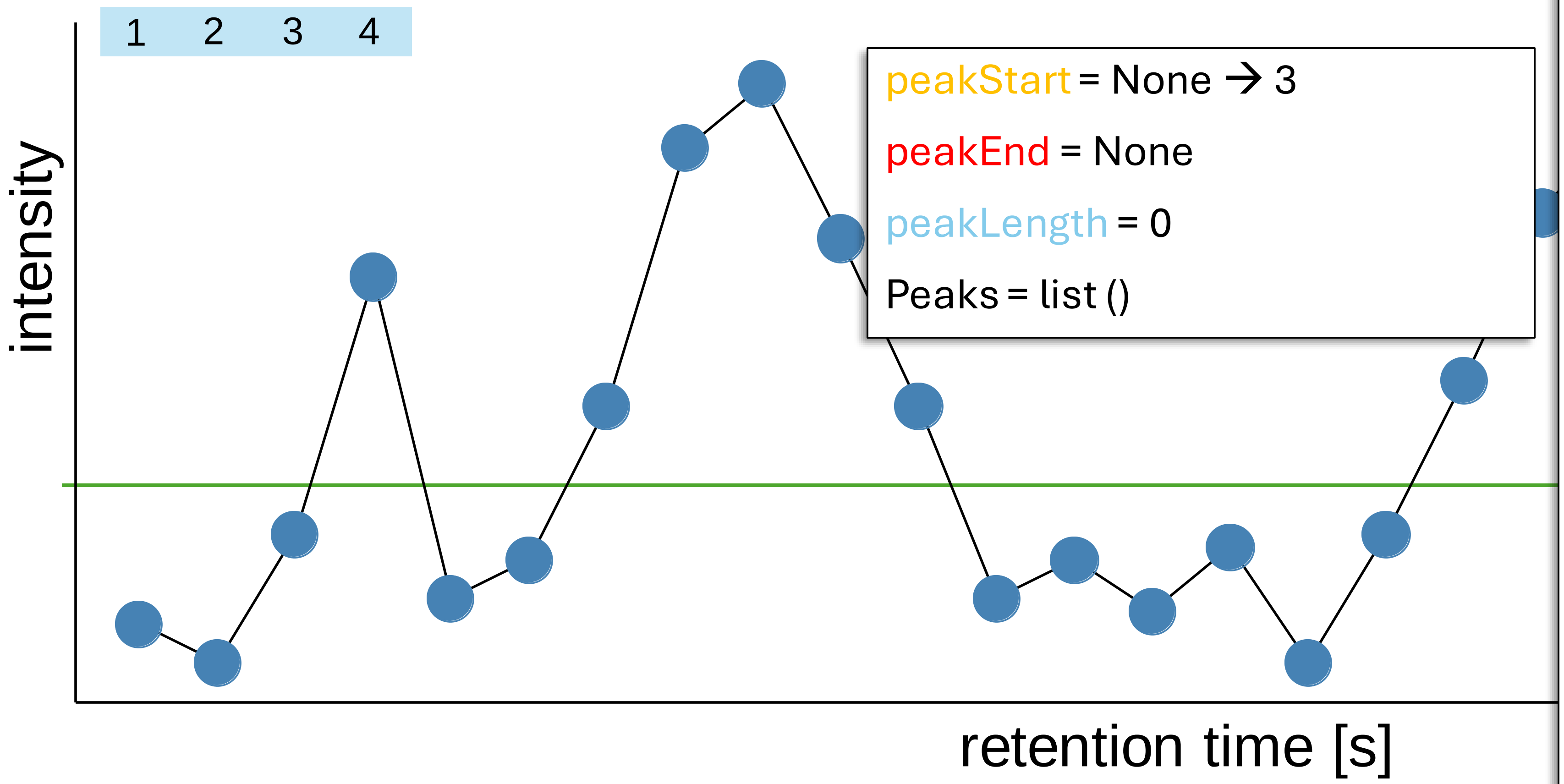
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None → 3

peakEnd = None

peakLength = 0

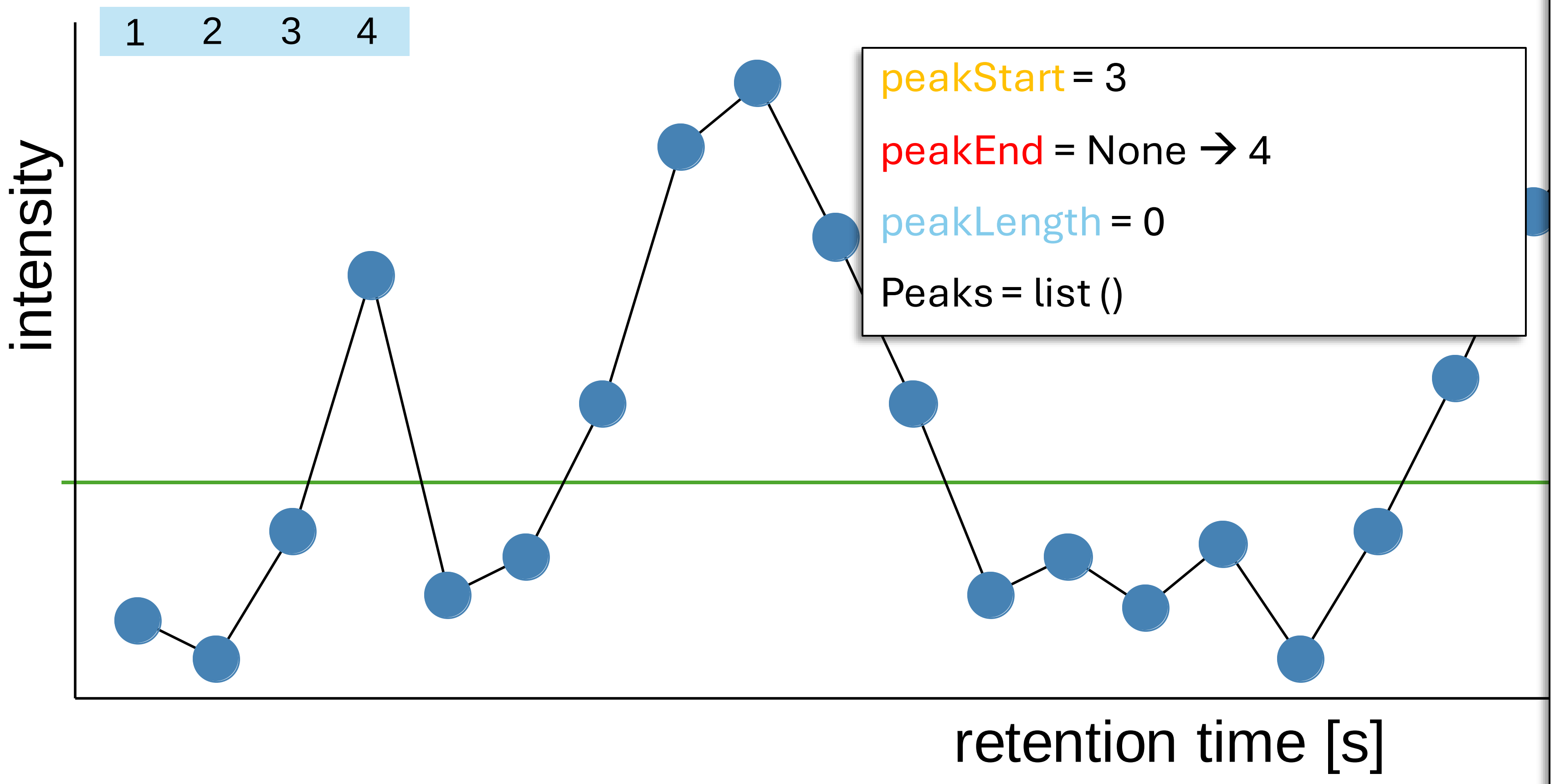
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 3

peakEnd = None → 4

peakLength = 0

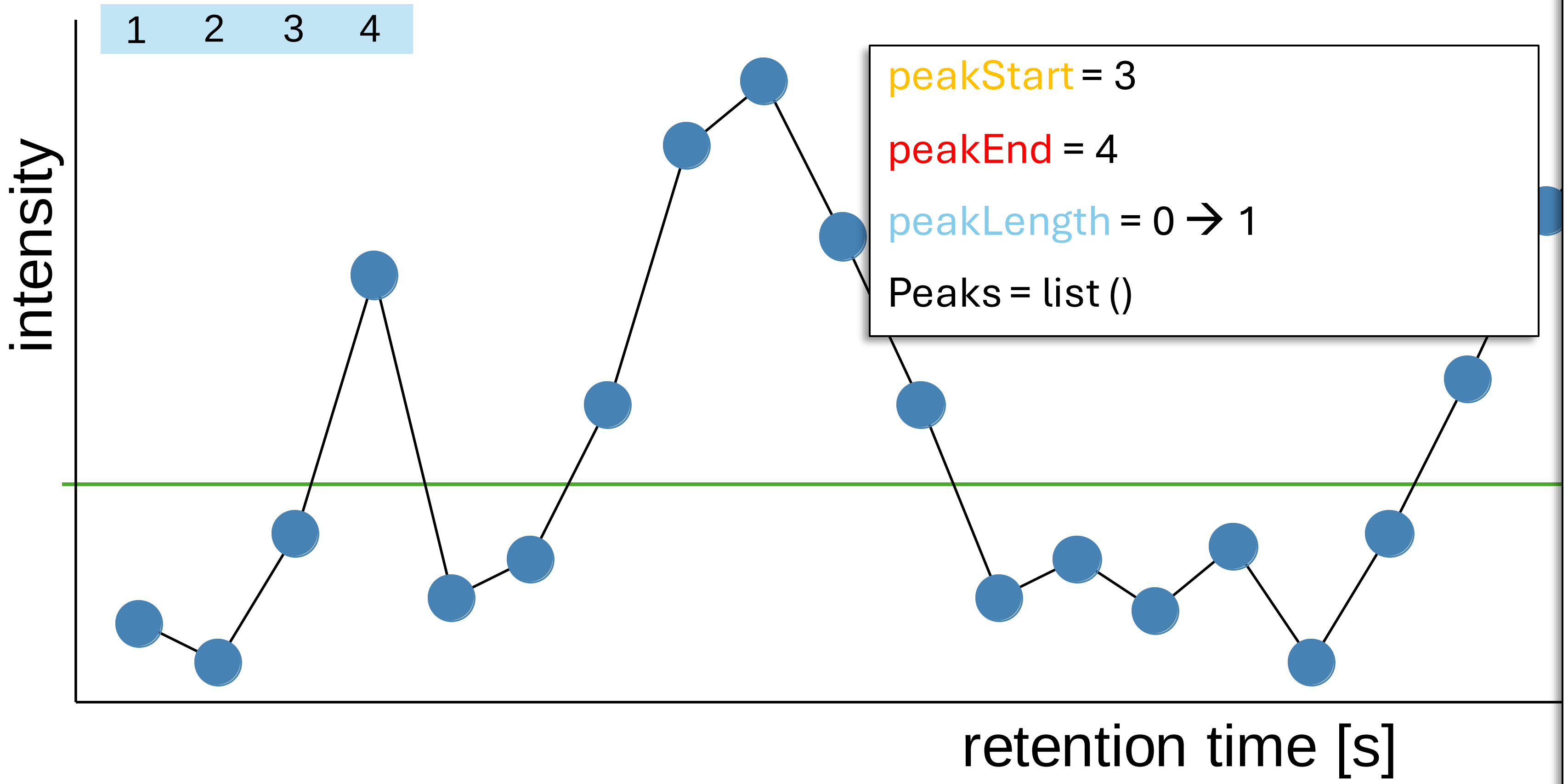
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 3

peakEnd = 4

peakLength = 0 → 1

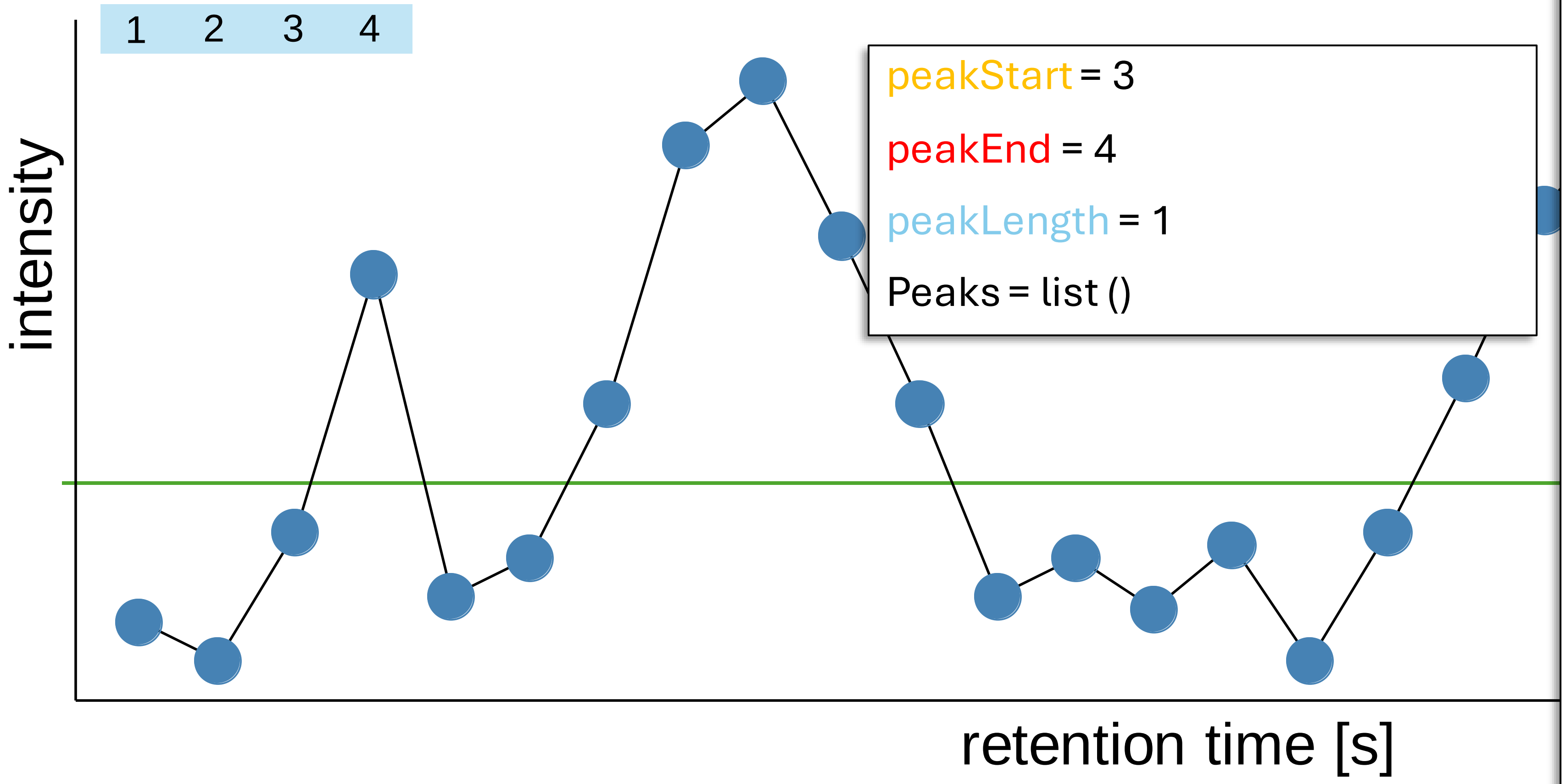
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 3

peakEnd = 4

peakLength = 1

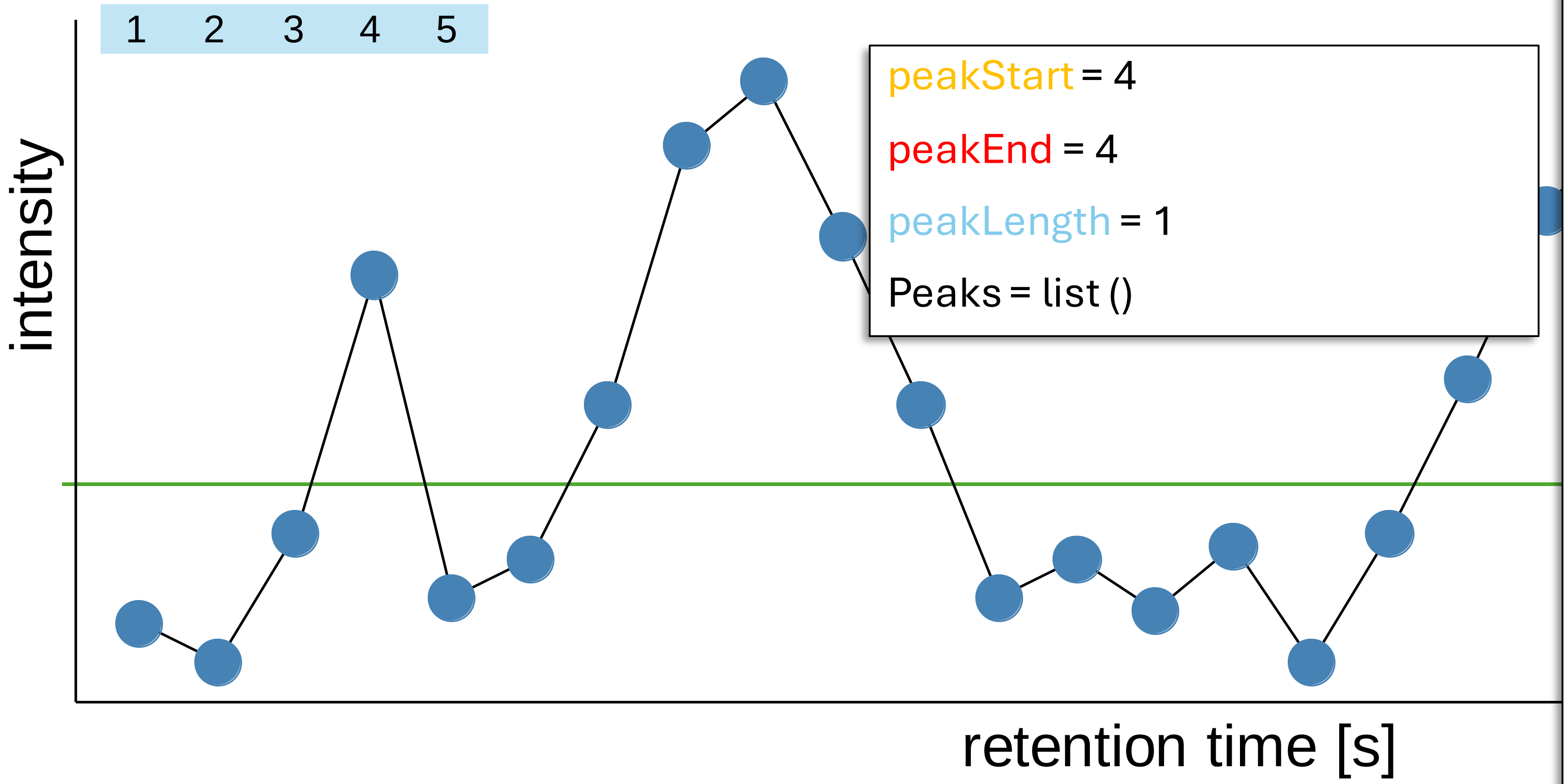
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:    X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 4

peakEnd = 4

peakLength = 1

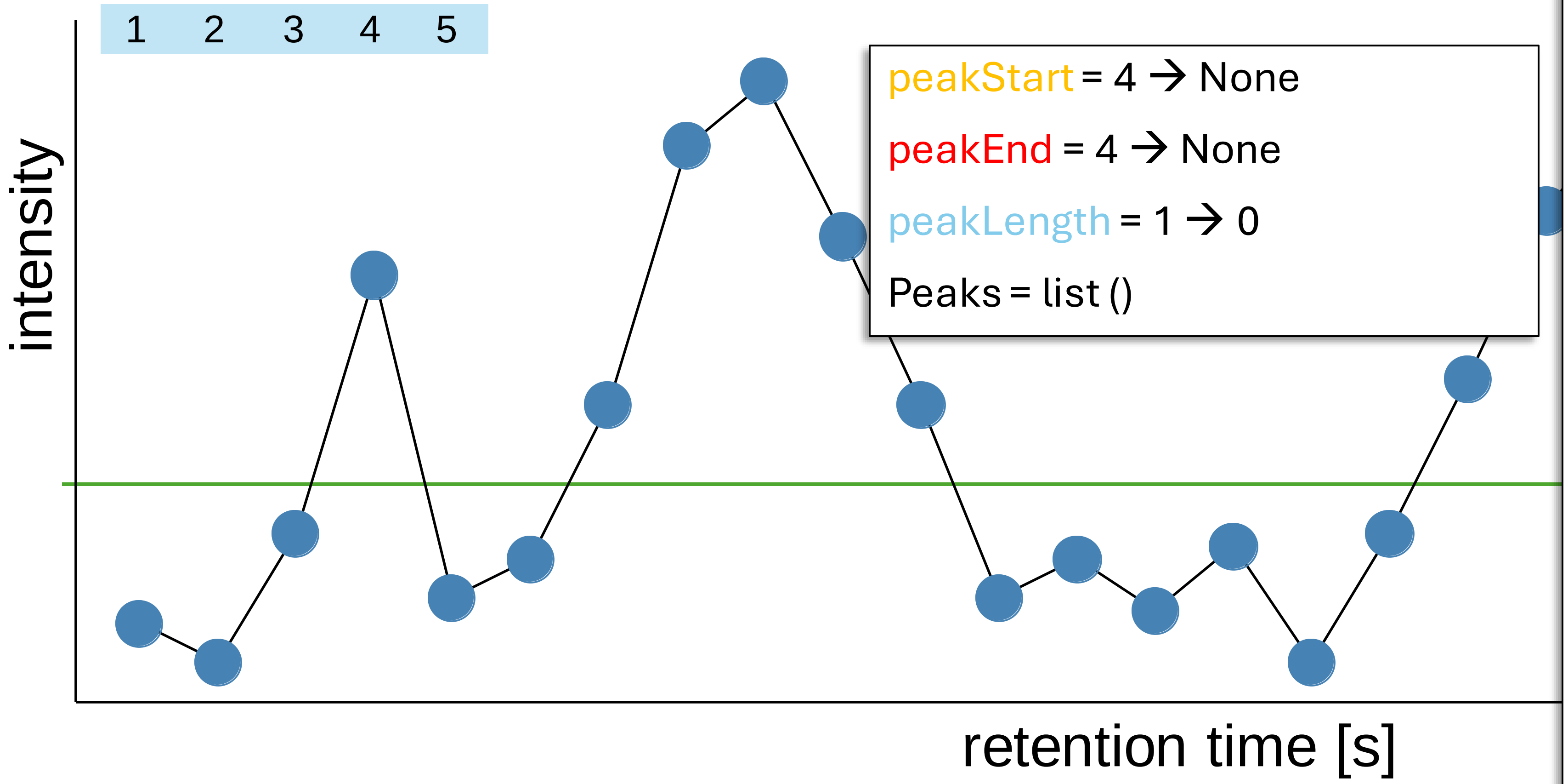
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :  X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:  X
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 4 → None

peakEnd = 4 → None

peakLength = 1 → 0

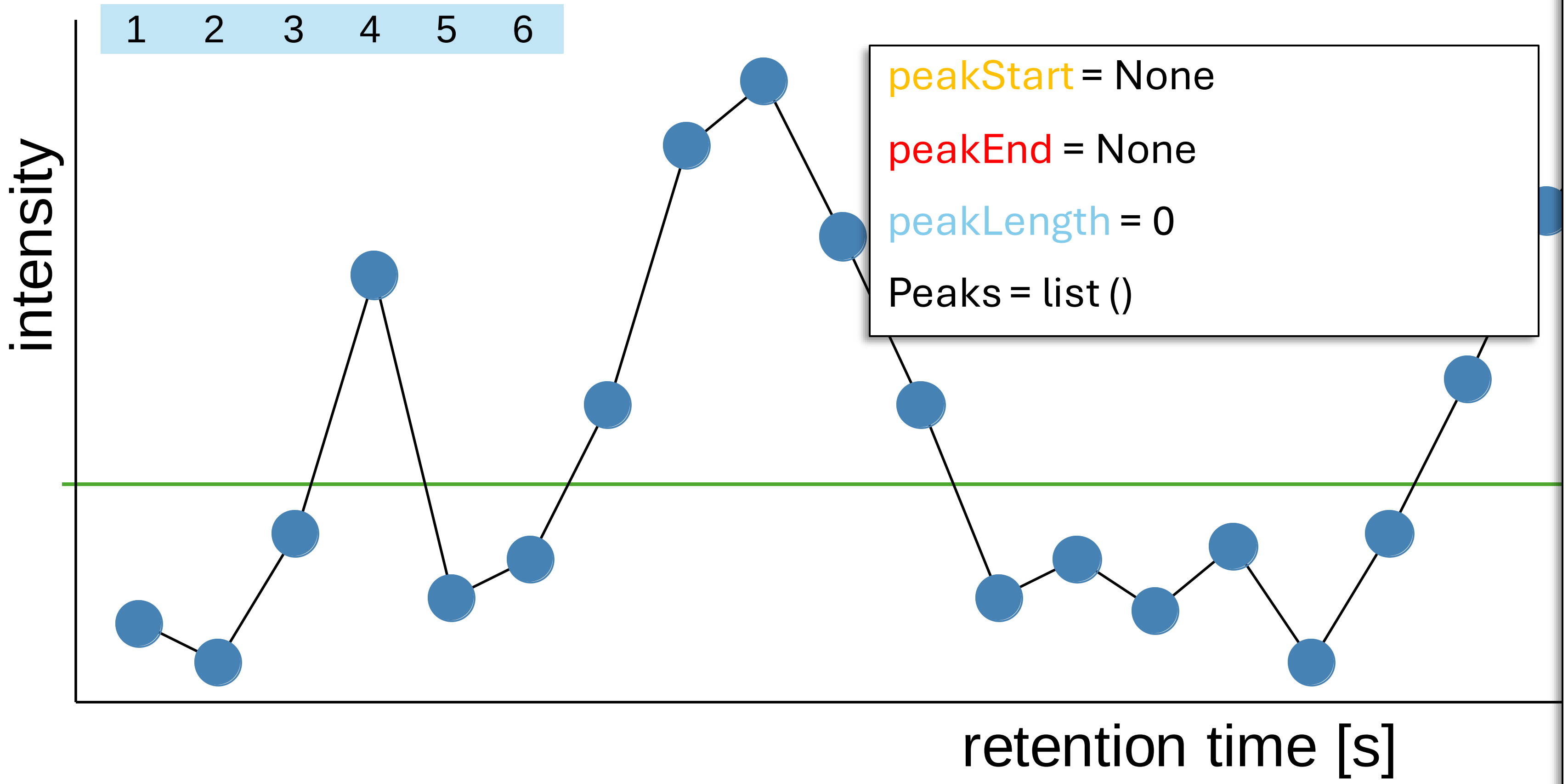
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :  X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:
            If peakLength >= min_points:  X
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

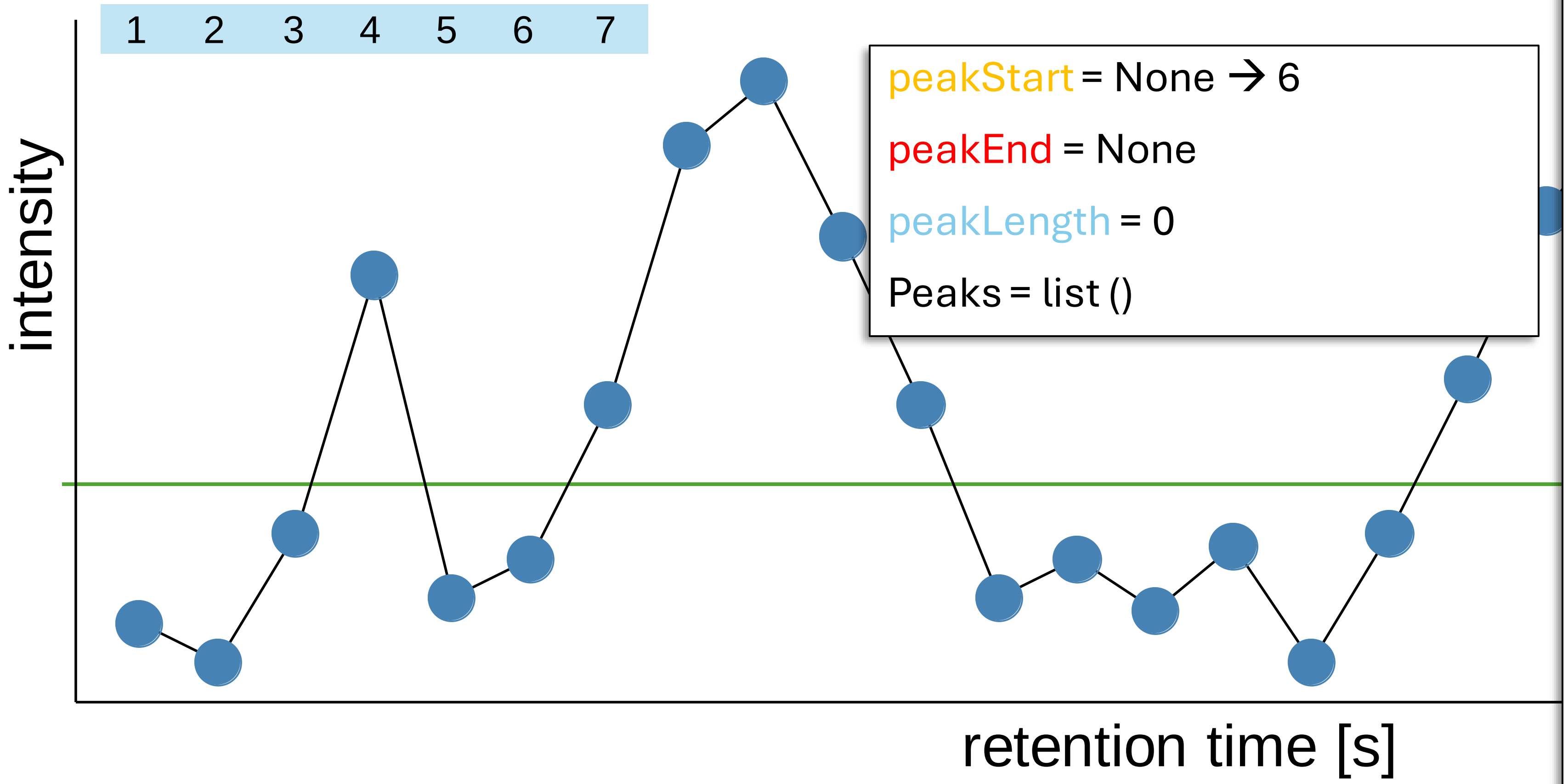
peakLength = 0

Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold : X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green
min_points = 4



peakStart = None → 6

peakEnd = None

peakLength = 0

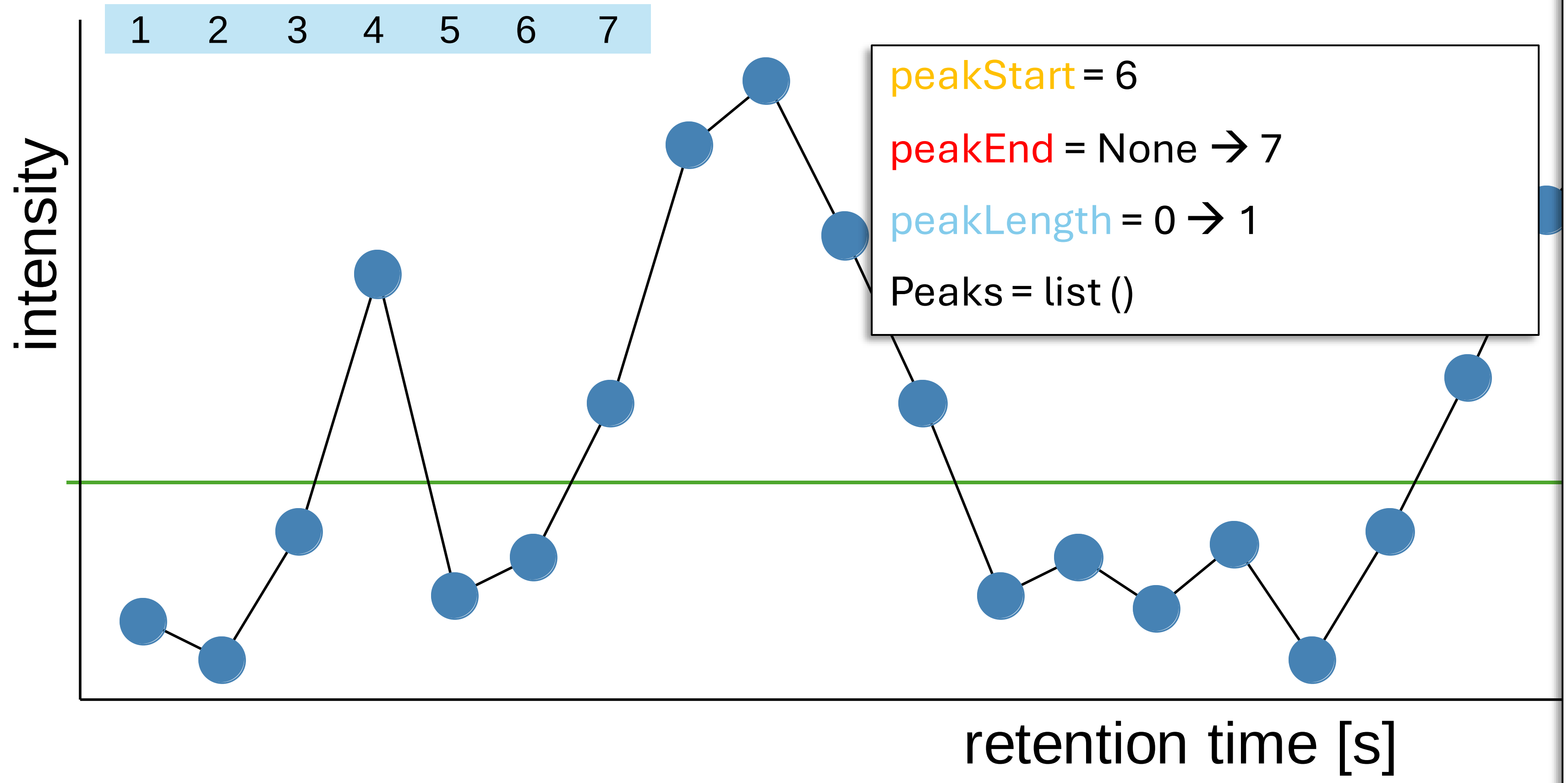
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6

peakEnd = None → 7

peakLength = 0 → 1

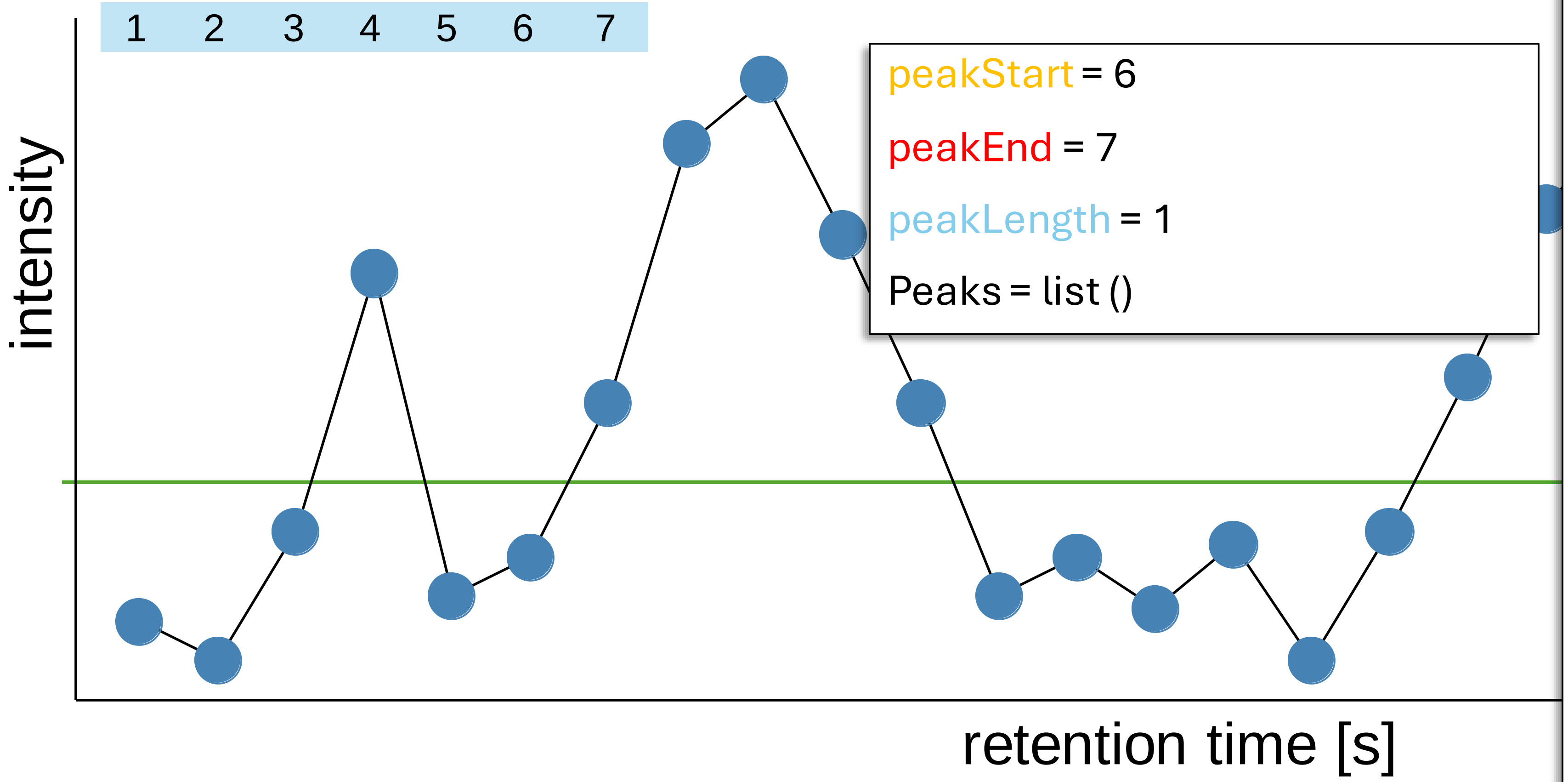
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:  X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6

peakEnd = 7

peakLength = 1

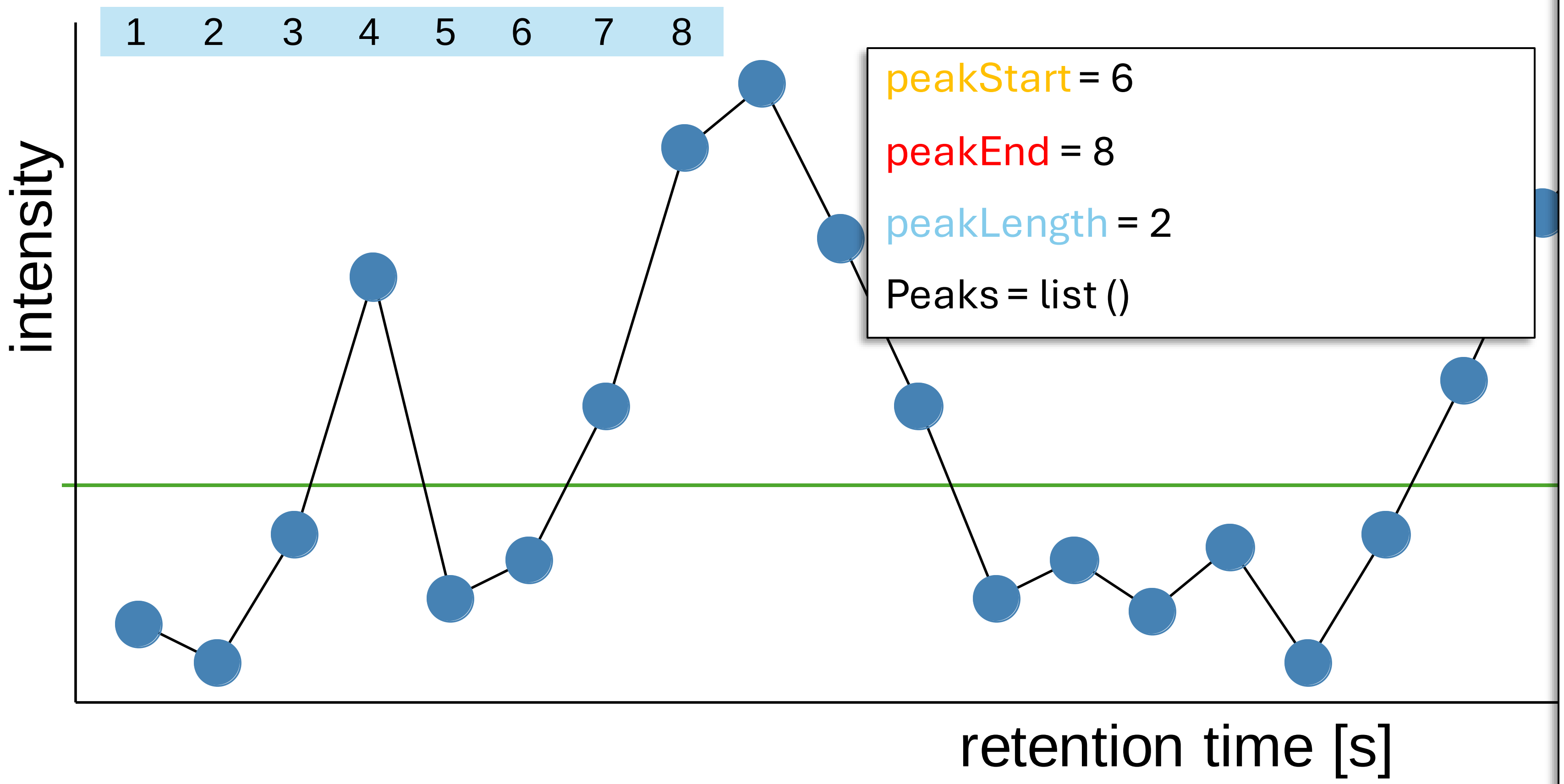
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:  X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6

peakEnd = 8

peakLength = 2

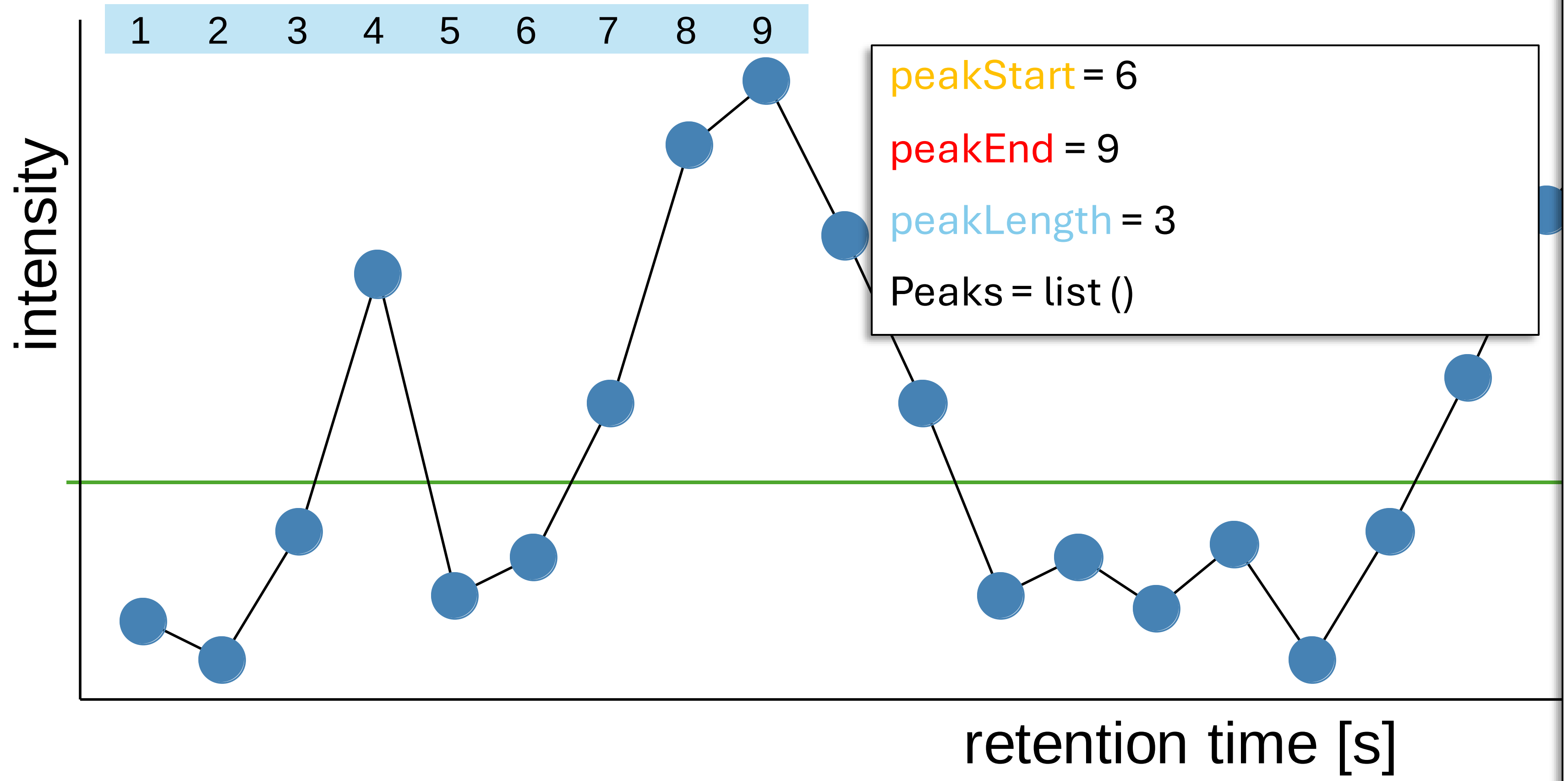
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6

peakEnd = 9

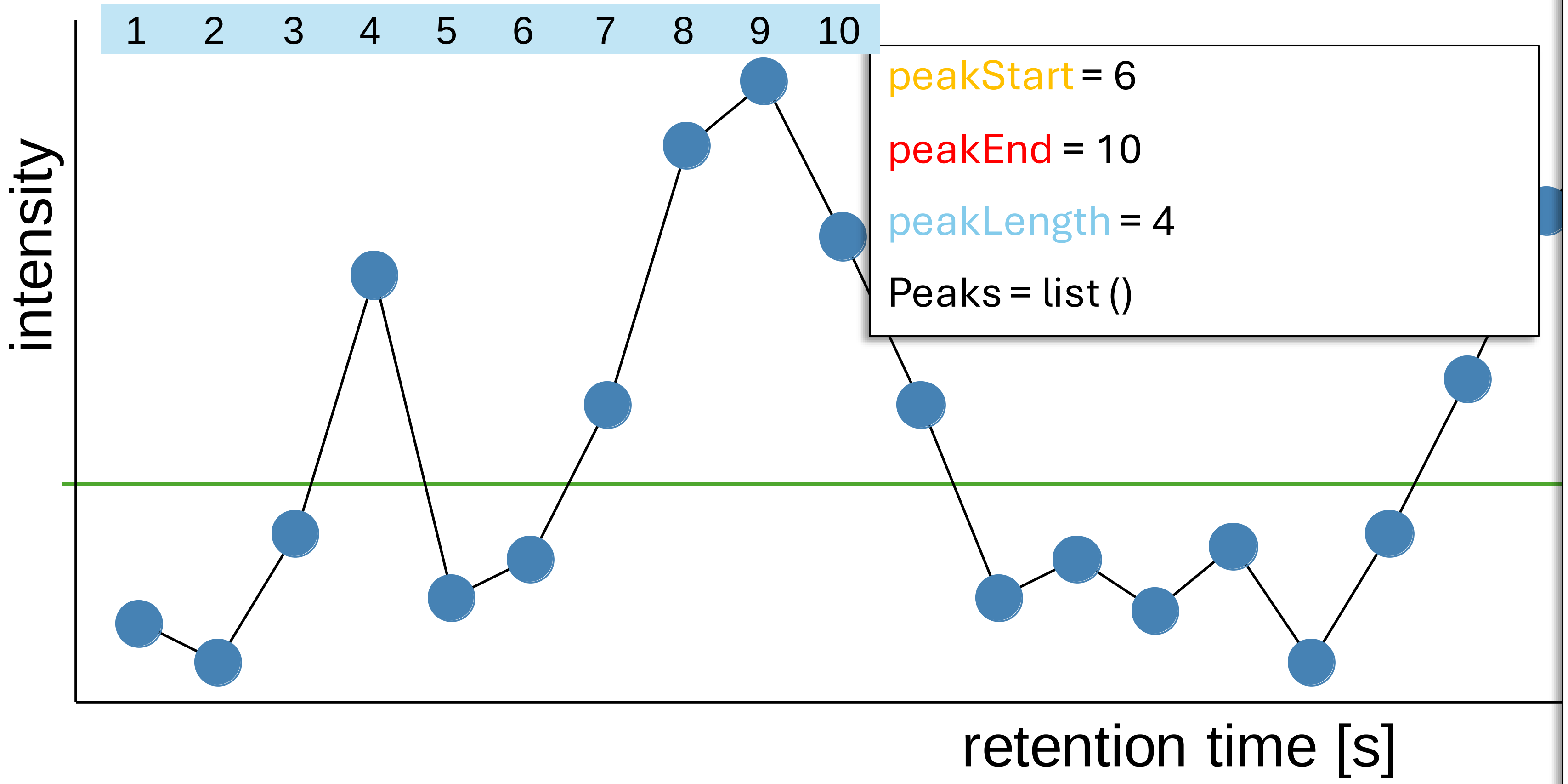
peakLength = 3

Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green
min_points = 4



peakStart = 6

peakEnd = 10

peakLength = 4

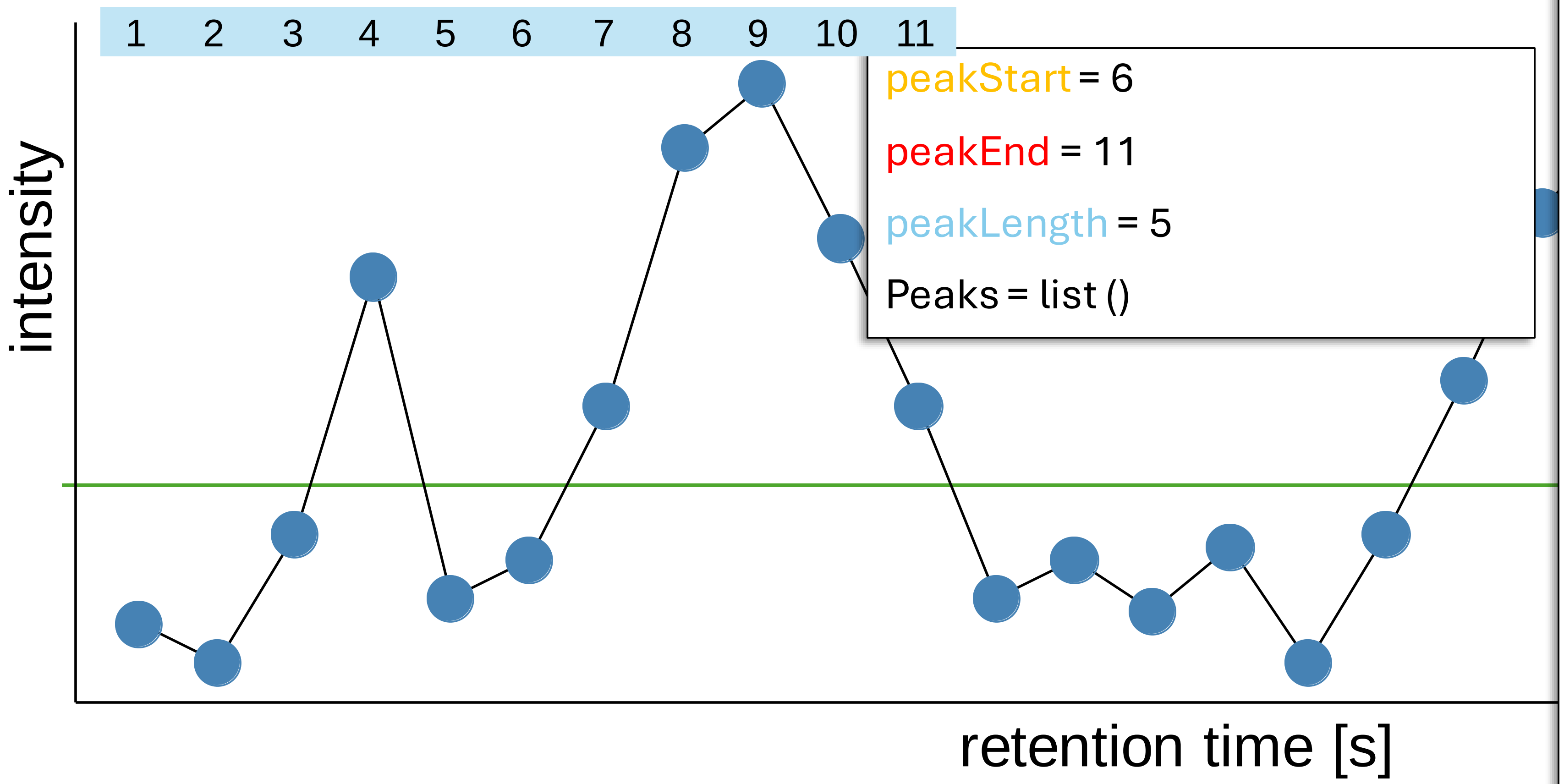
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)
    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6

peakEnd = 11

peakLength = 5

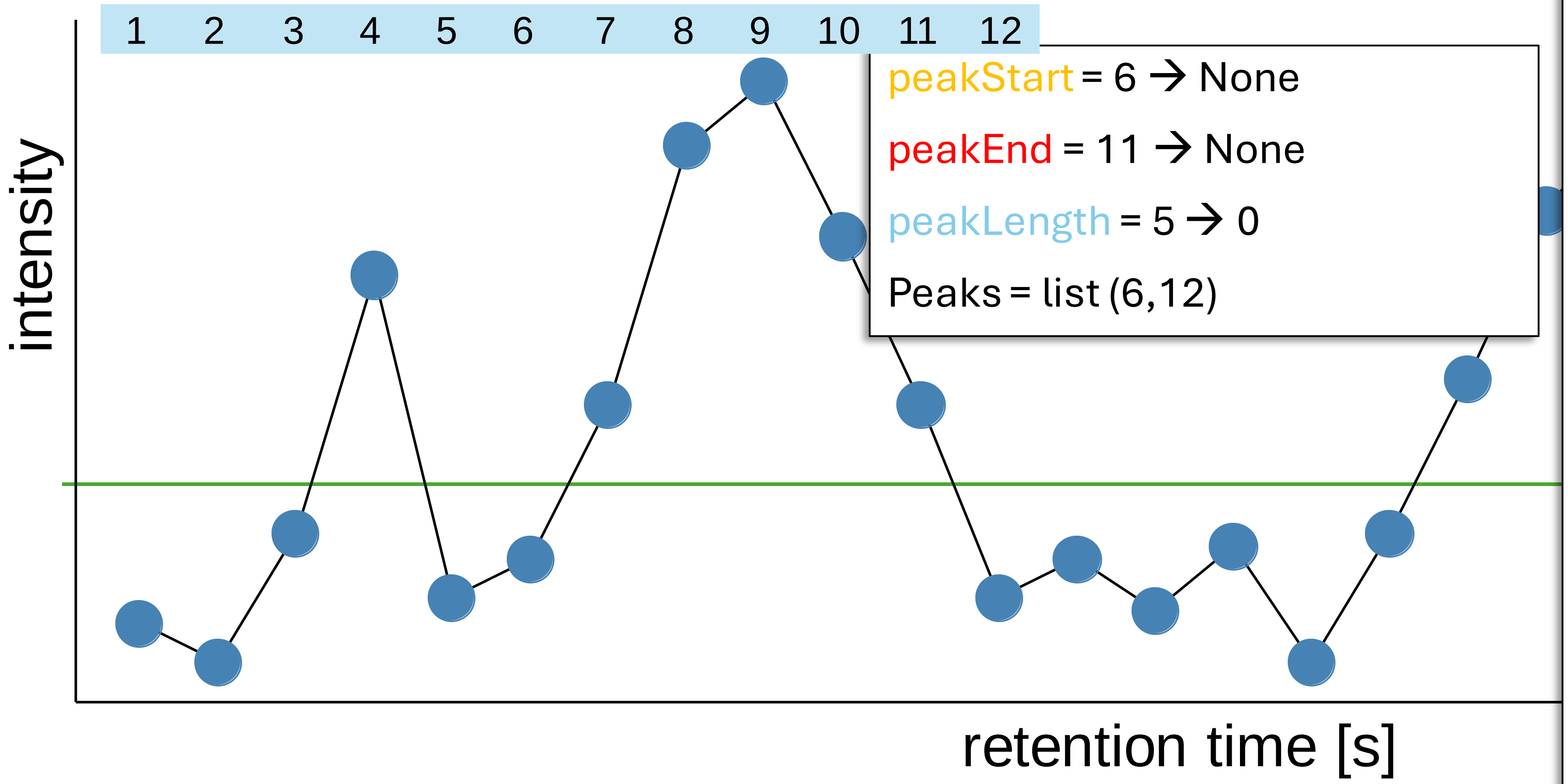
Peaks = list ()

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None: X
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None: X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 6 → None

peakEnd = 11 → None

peakLength = 5 → 0

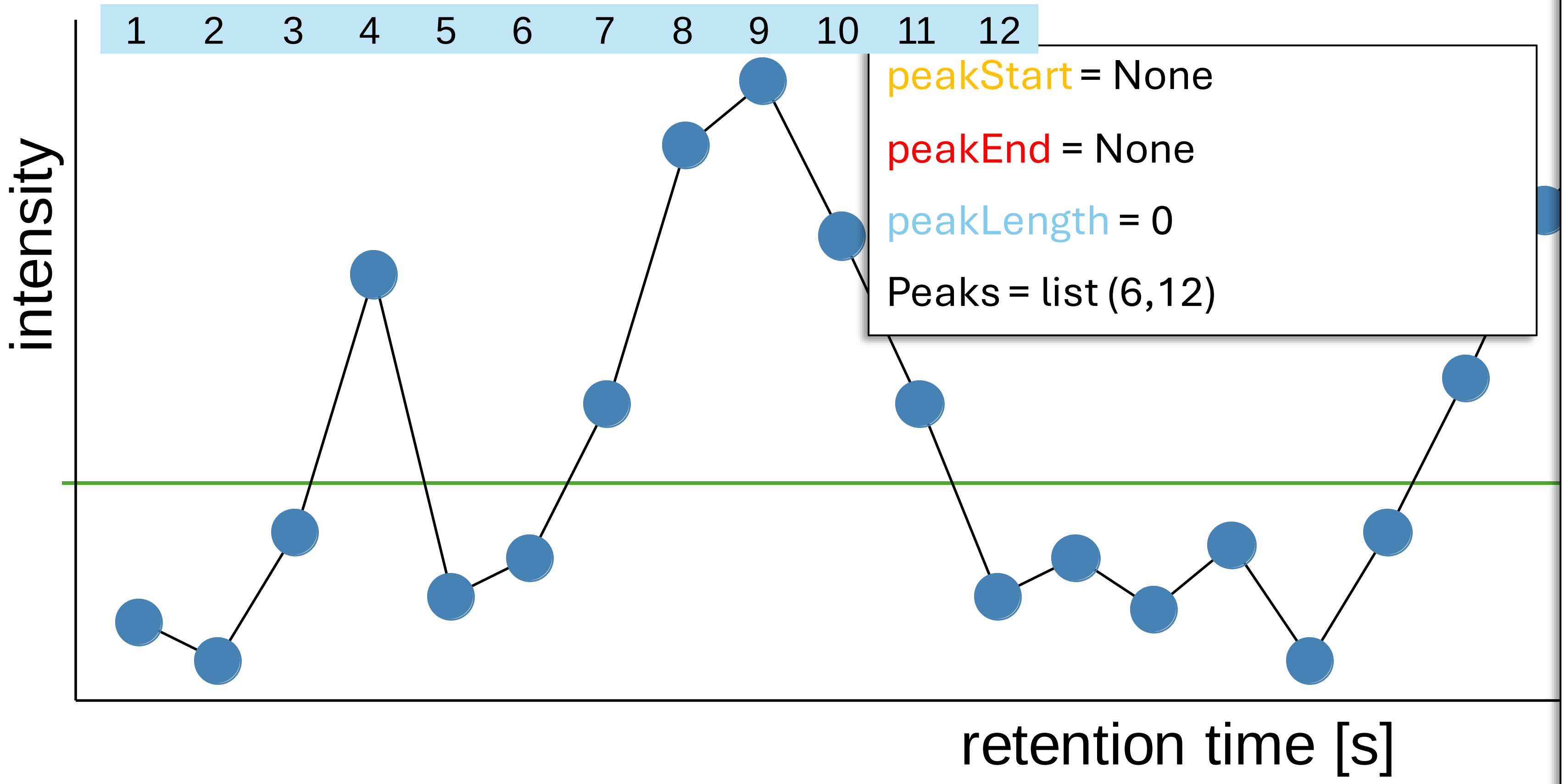
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold : X
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

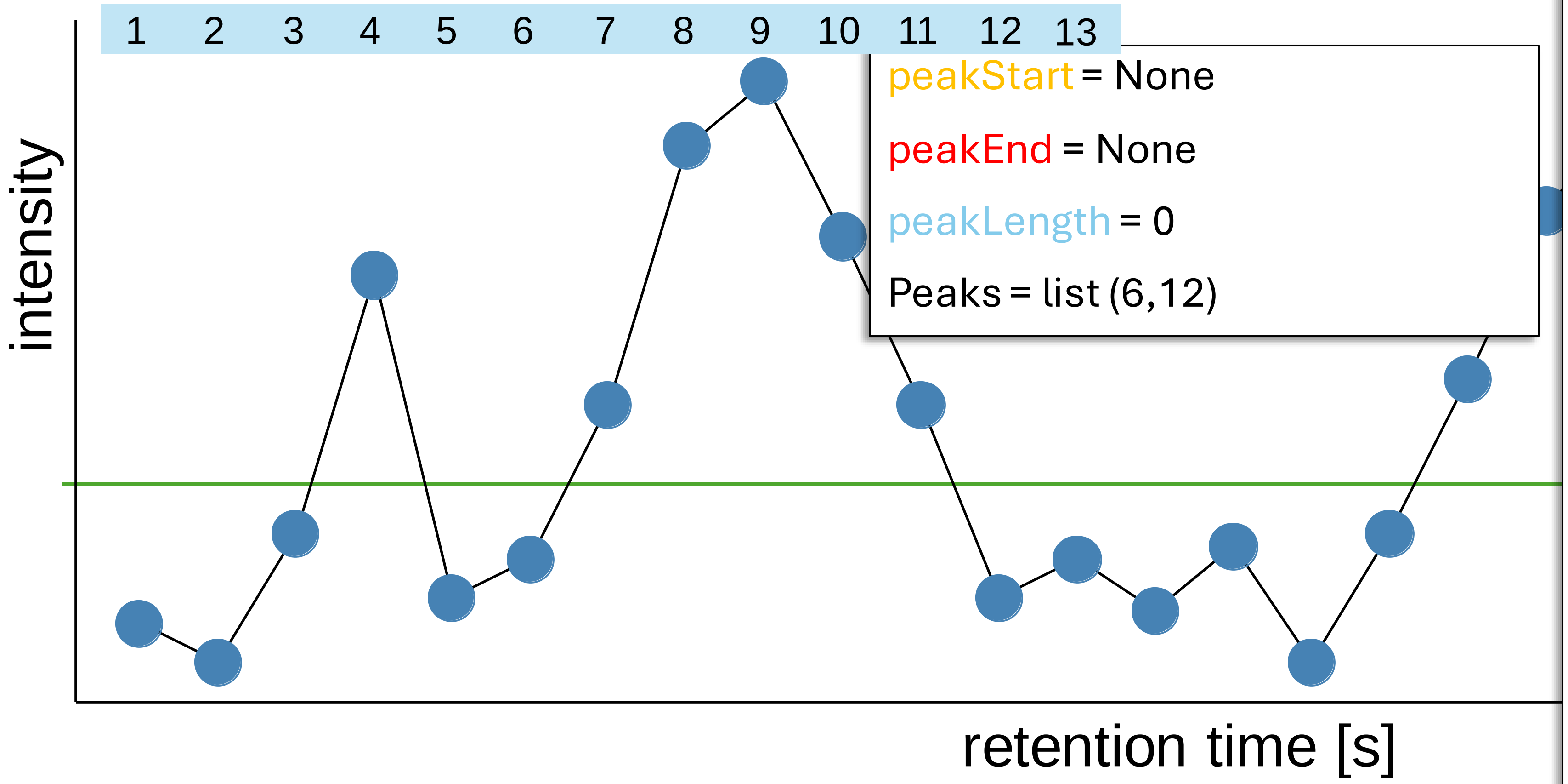
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold : X
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

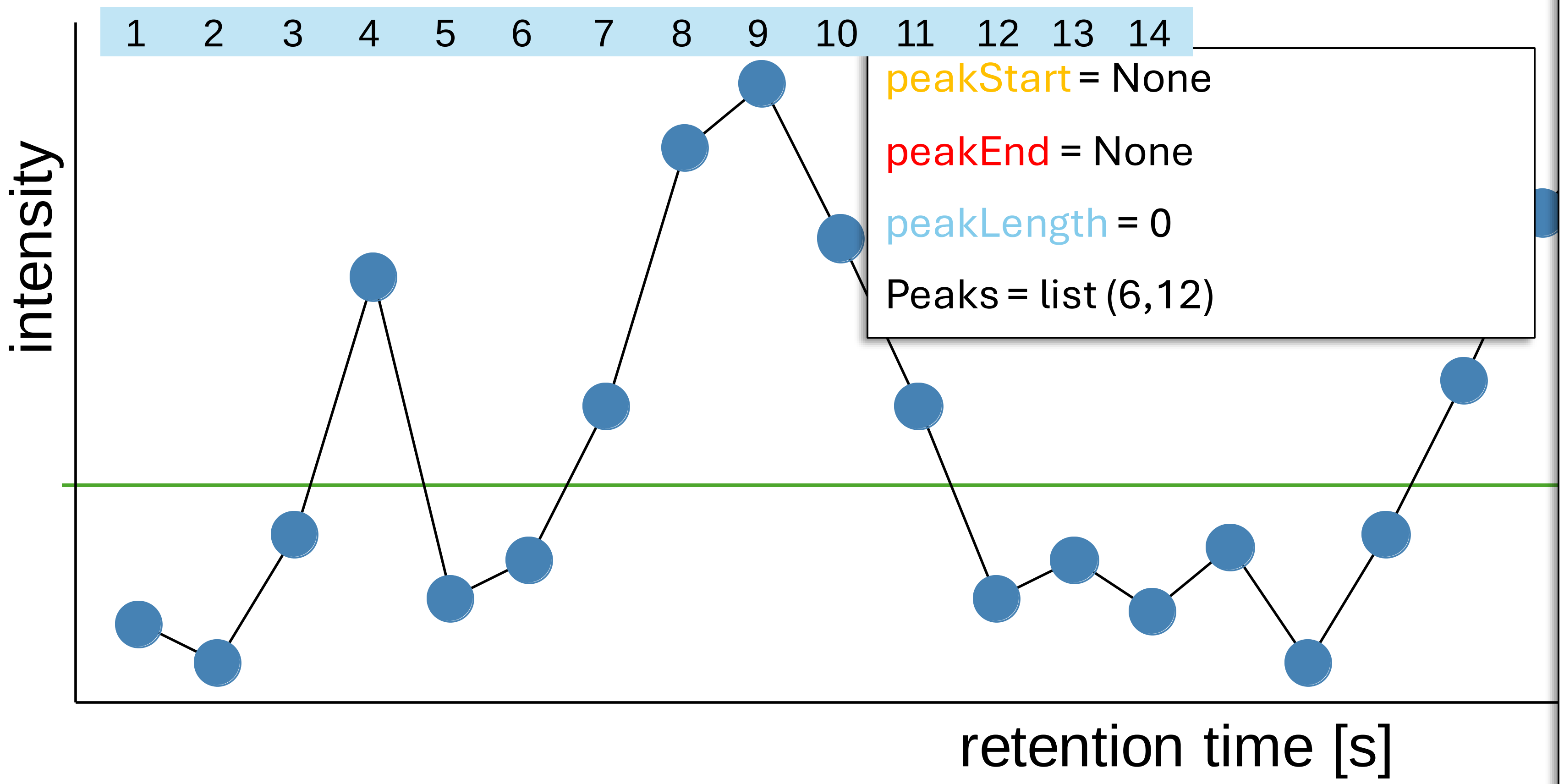
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold : X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

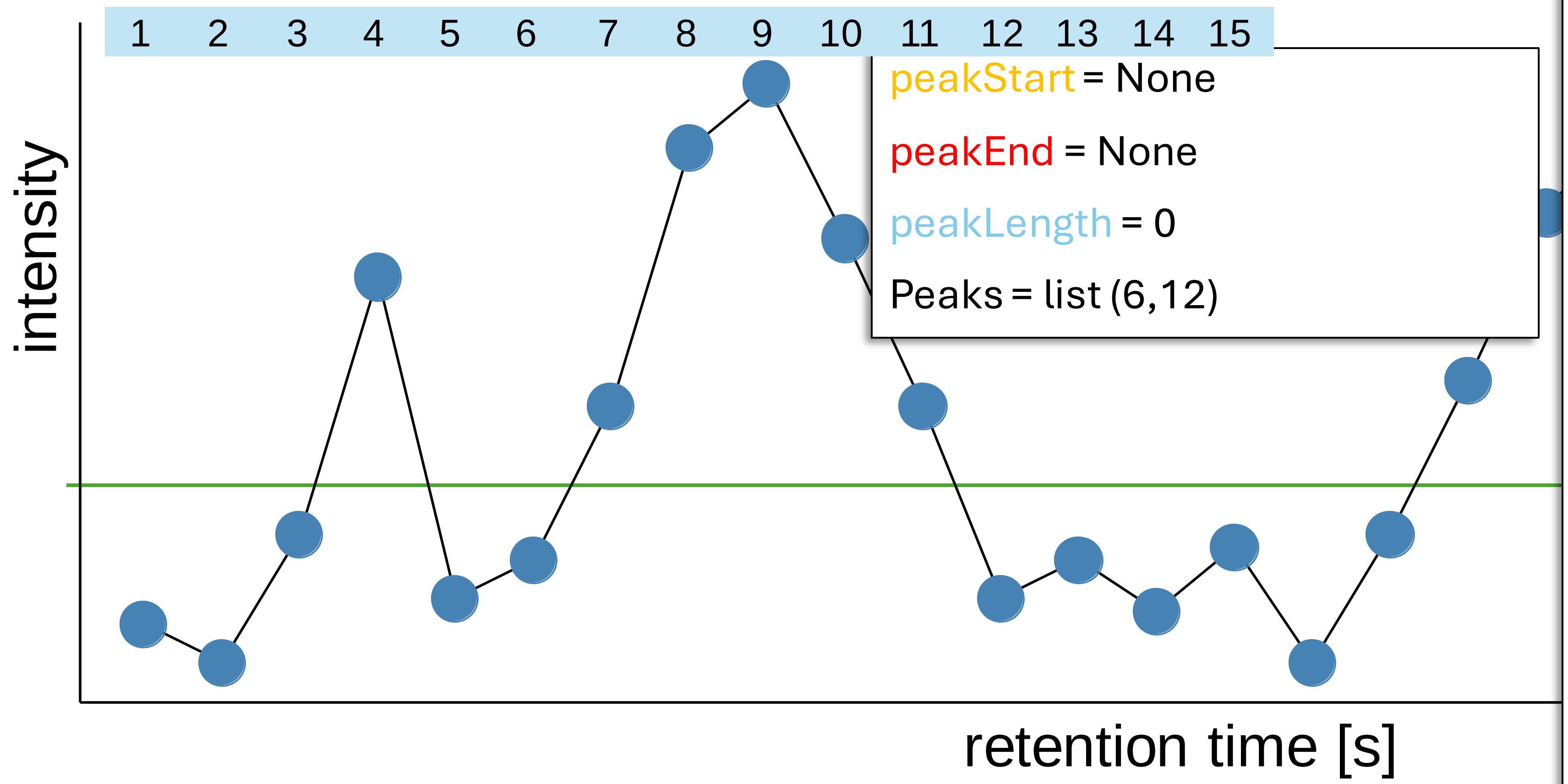
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold : X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

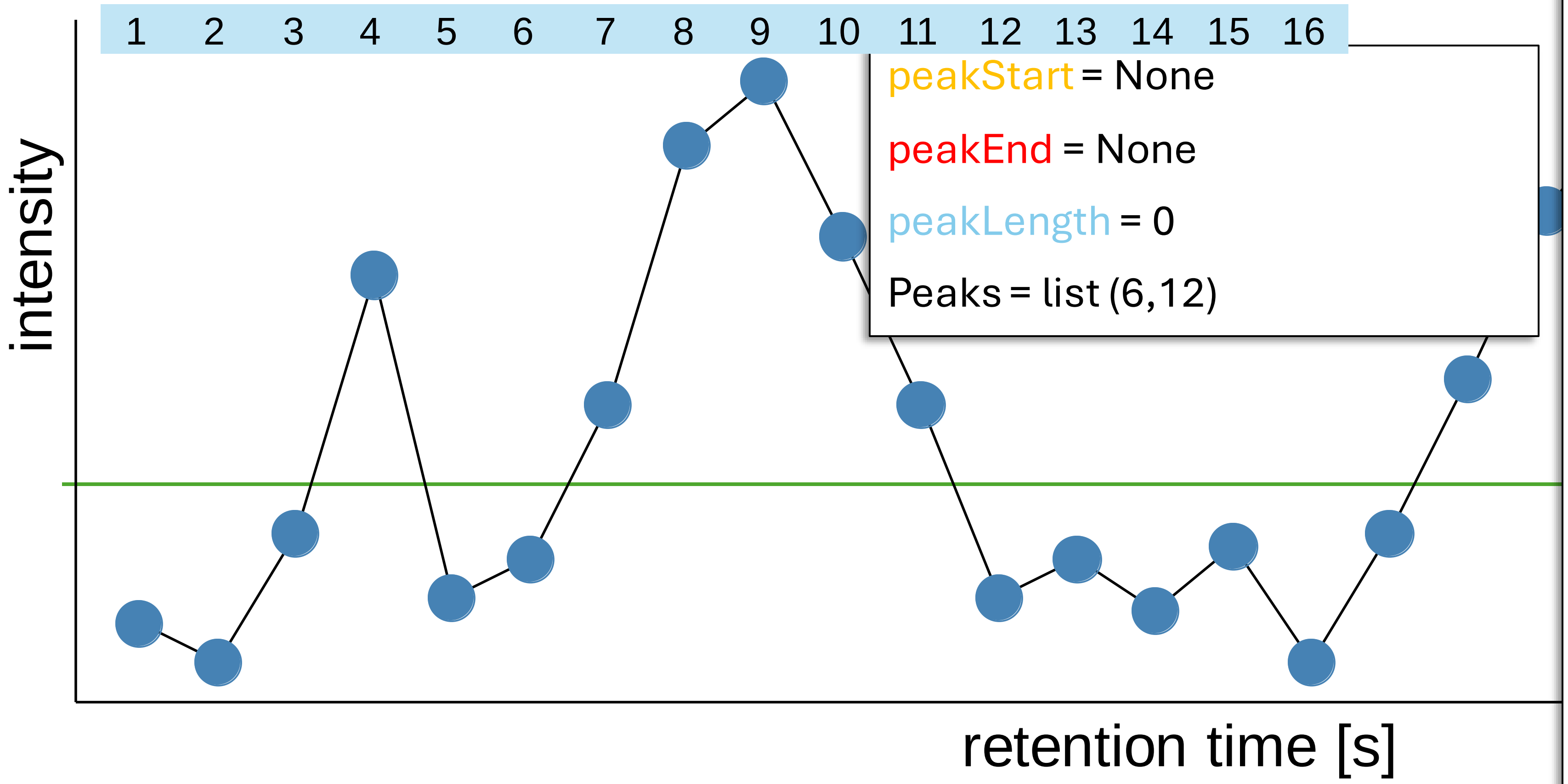
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

 peaks = empty list
 peakStart = None
 peakEnd = None
 peakLength = 0

 For each index in signal_vector:
   value = signal_vector[index]
   If value > min_int_threshold : X
     If peakStart is None:
       peakStart = index - 1 if index > 1 else 1
     peakEnd = index
     peakLength = peakLength + 1
   Else if peakStart is not None: X
     If peakLength >= min_points:
       Append [peakStart, peakEnd] to peaks
     peakStart = None
     peakEnd = None
     peakLength = 0

 Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

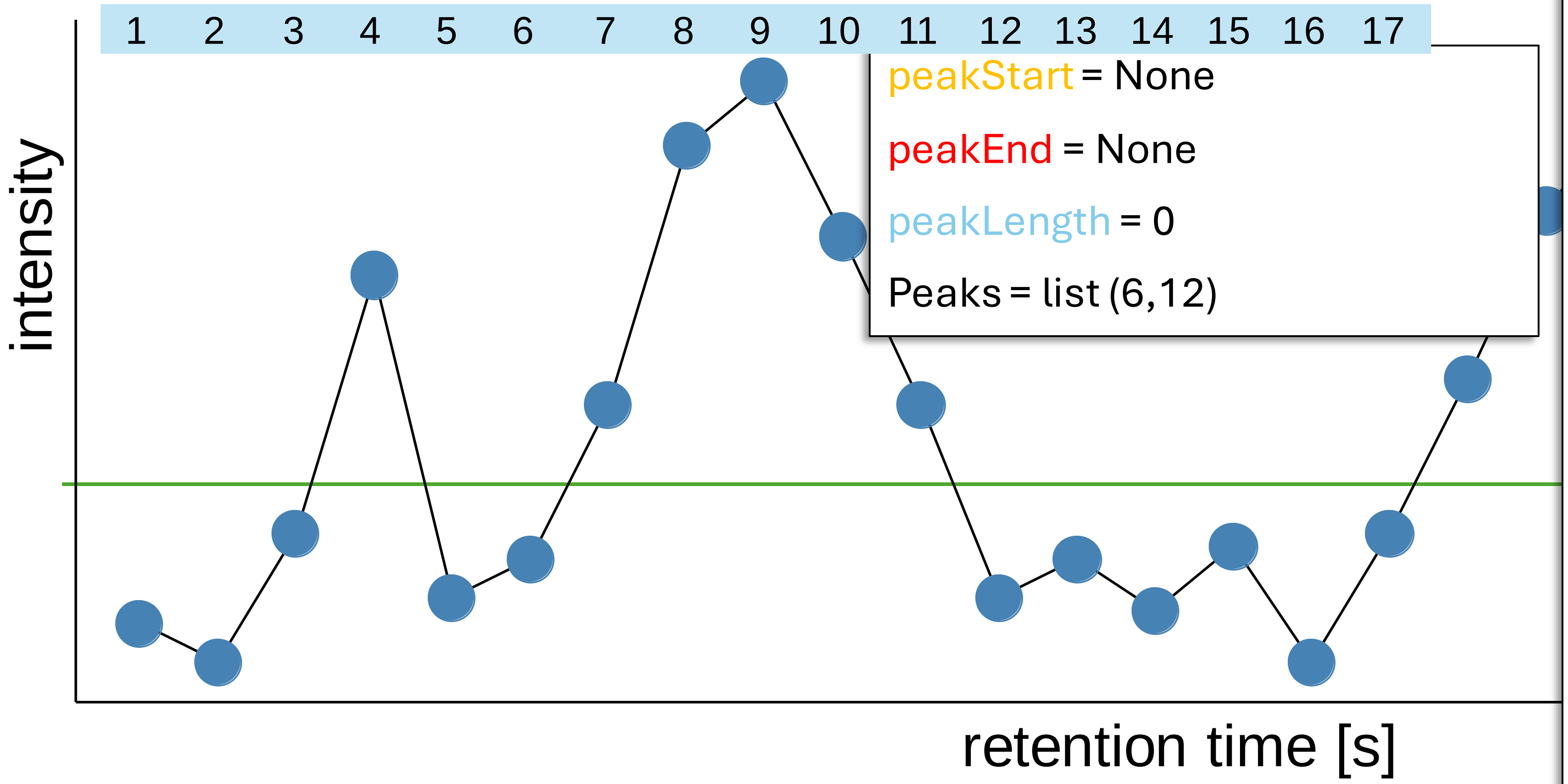
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold : X
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

peakLength = 0

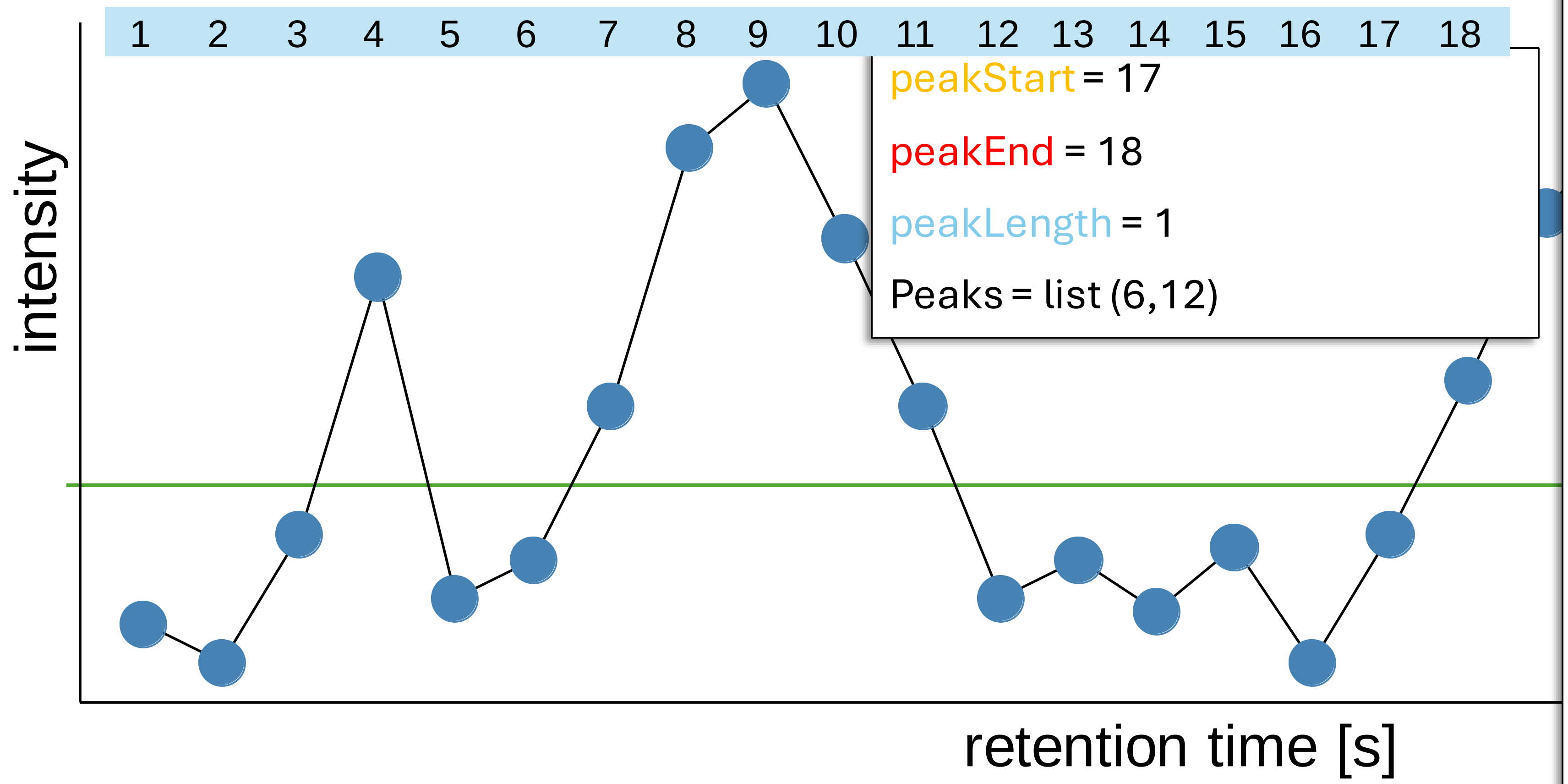
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold : X
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None: X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 17

peakEnd = 18

peakLength = 1

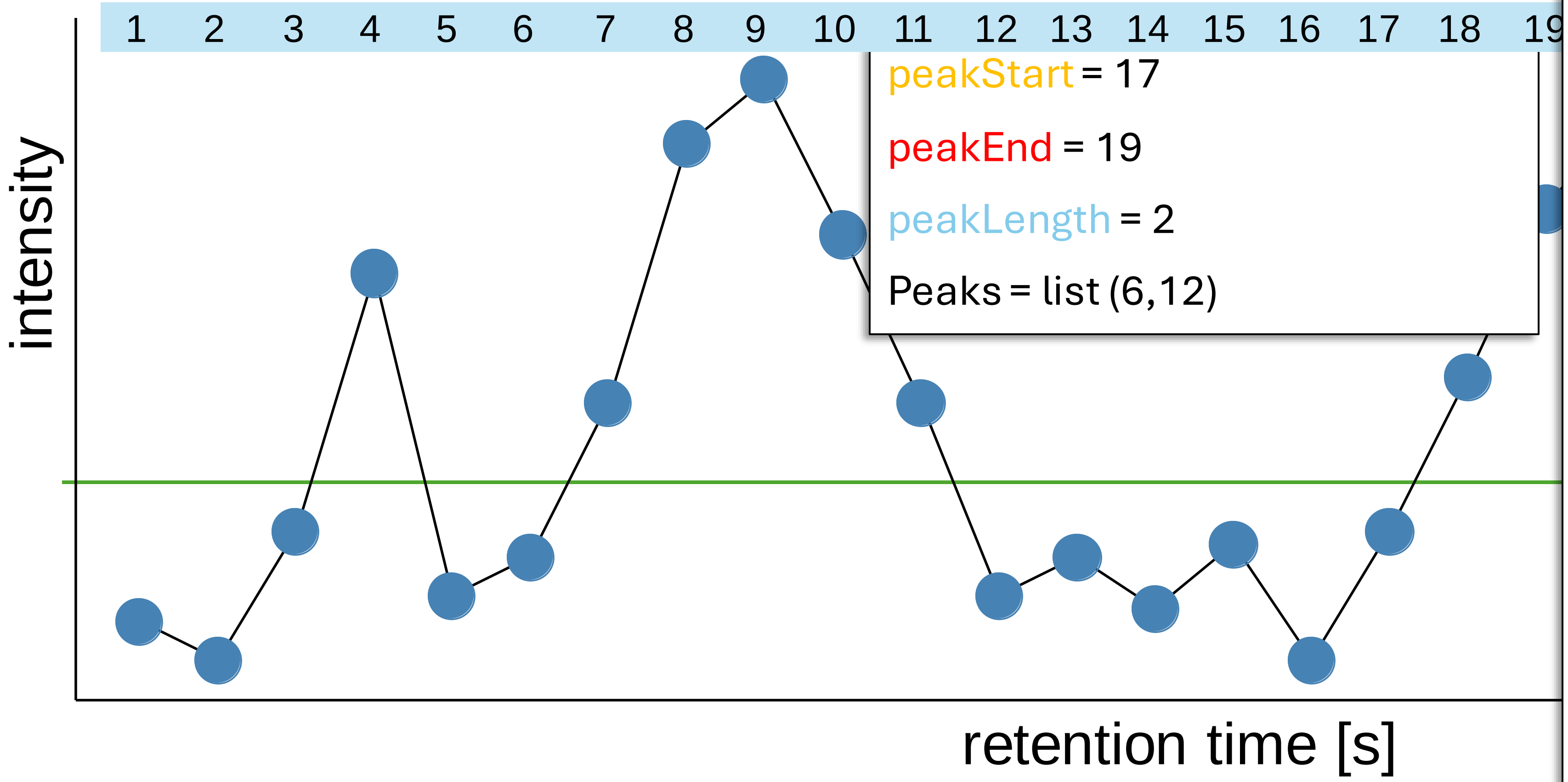
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None:
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None:  X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

Pseudo code algorithm

min_intensity_threshold = green

min_points = 4



peakStart = 17

peakEnd = 19

peakLength = 2

Peaks = list (6,12)

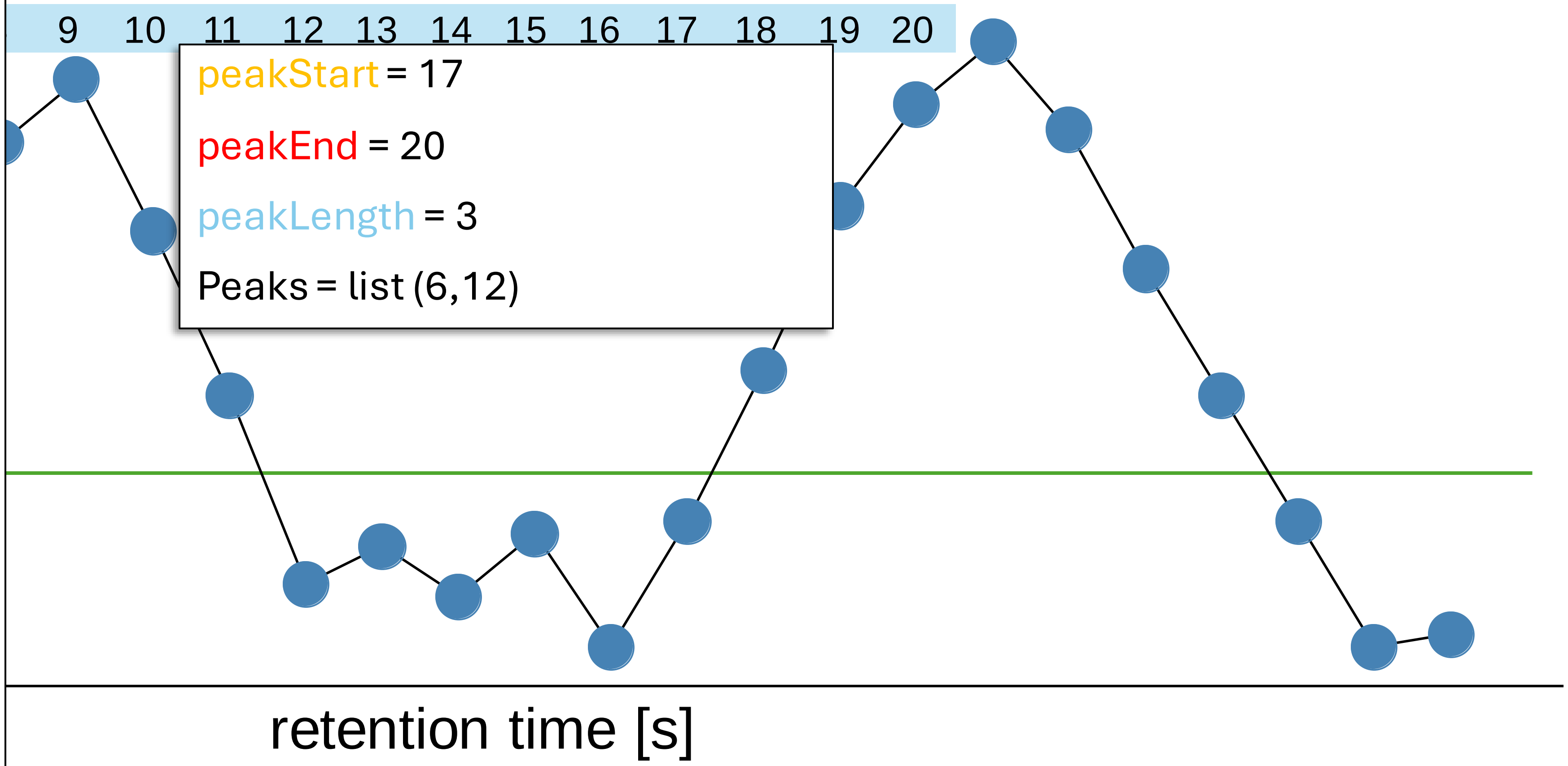
```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None: X
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None: X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

min_intensity_threshold = green

min_points = 4

5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1



peakStart = 17

peakEnd = 20

peakLength = 3

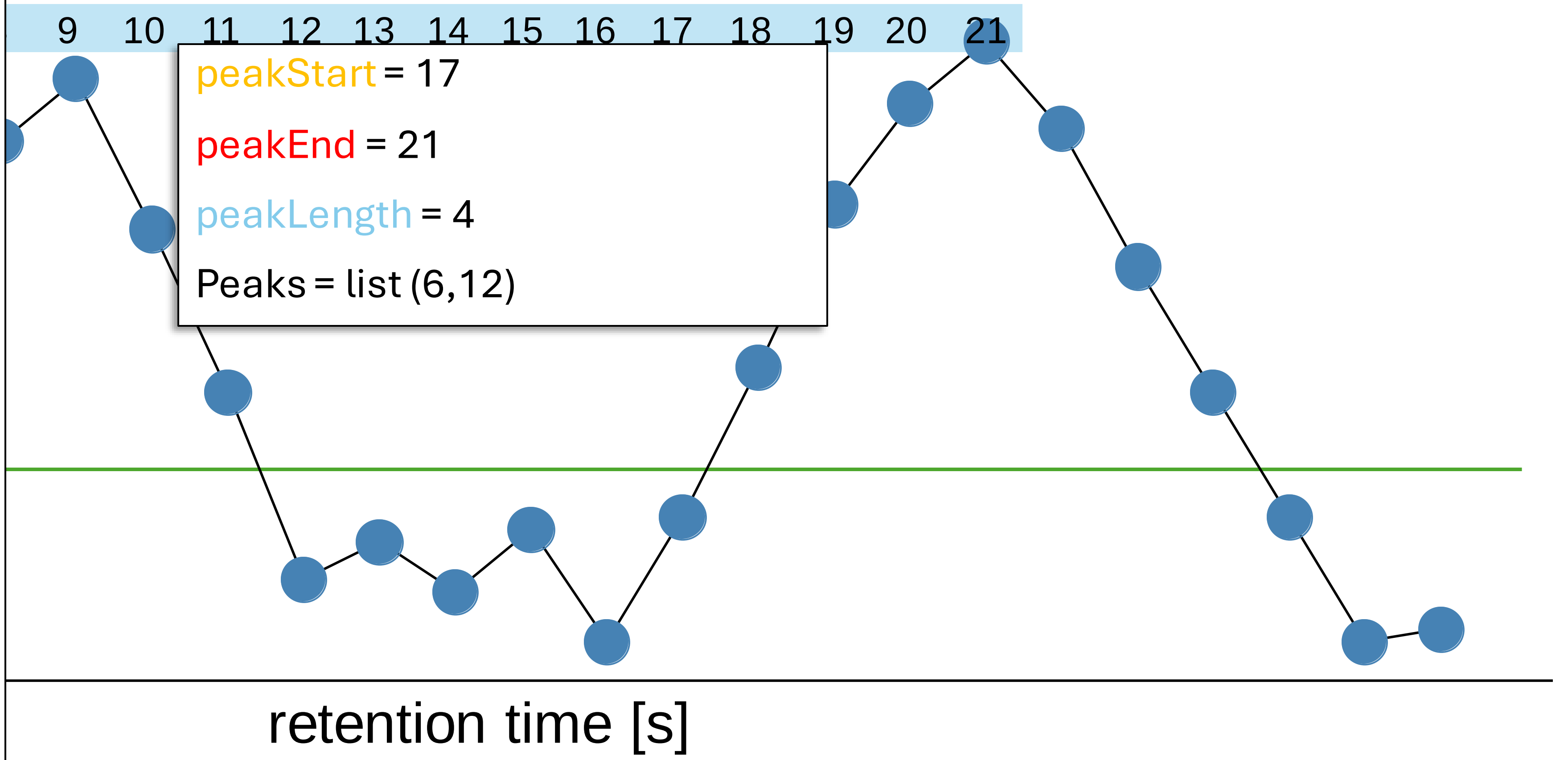
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list

    peakStart = None

    peakEnd = None

    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

min_intensity_threshold = green

min_points = 4

5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1



peakStart = 17

peakEnd = 21

peakLength = 4

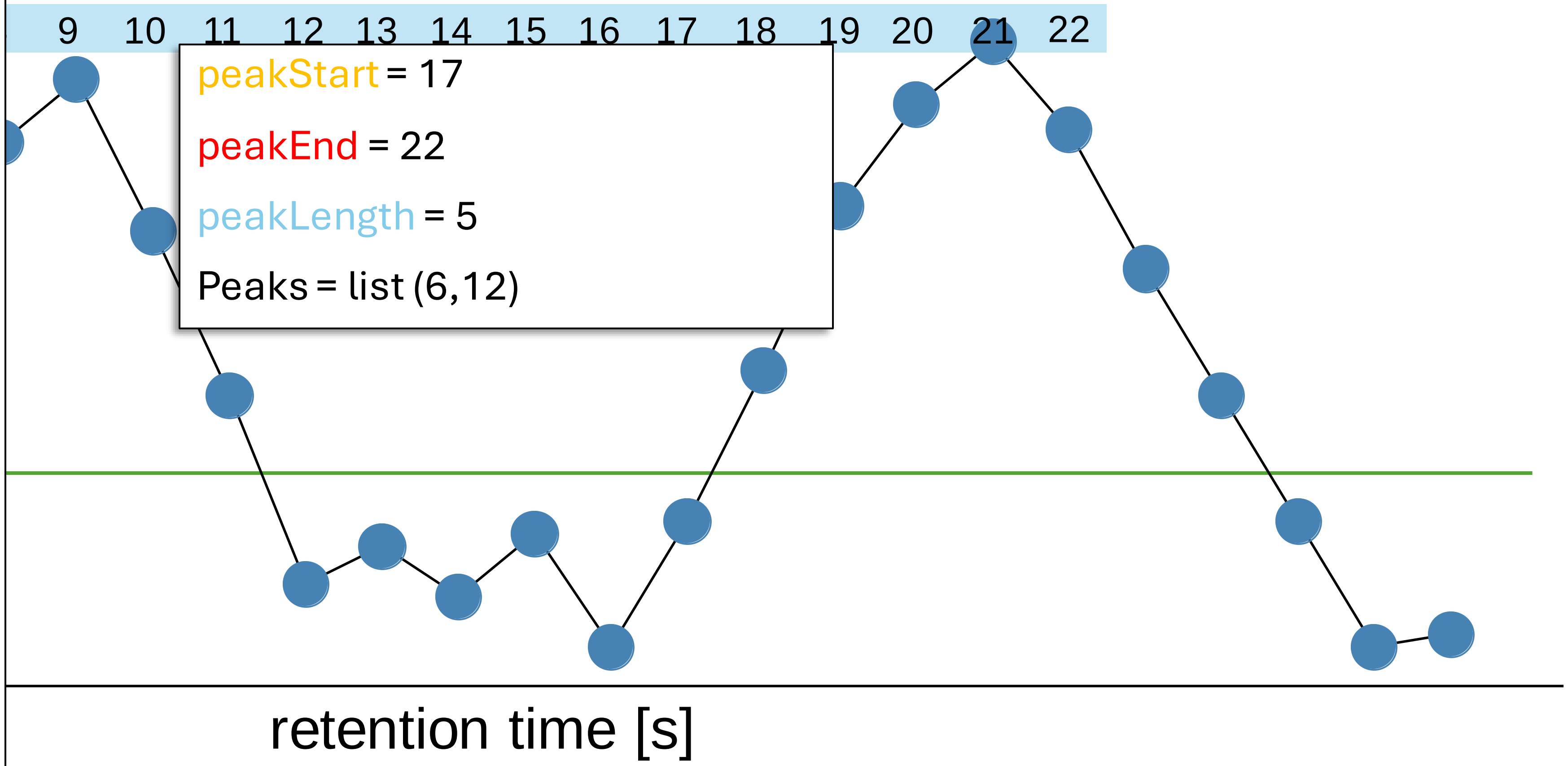
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None: X
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None: X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

min_intensity_threshold = green

min_points = 4

5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1



peakStart = 17

peakEnd = 22

peakLength = 5

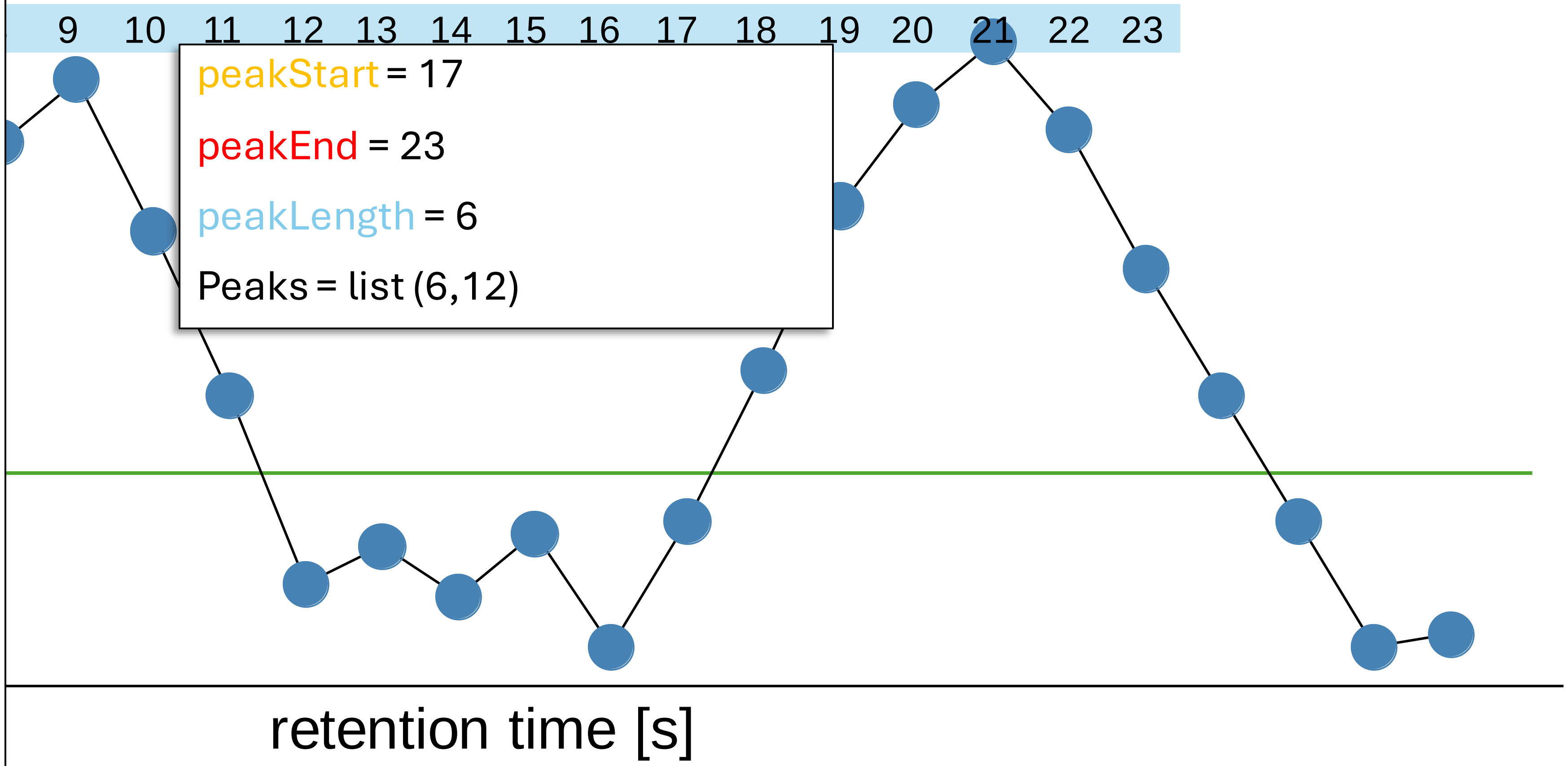
Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list

    peakStart = None

    peakEnd = None

    peakLength = 0

    For each index in signal_vector:

        value = signal_vector[index]

        If value > min_int_threshold :

            If peakStart is None: X

                peakStart = index - 1 if index > 1 else 1

            peakEnd = index

            peakLength = peakLength + 1

        Else if peakStart is not None: X

            If peakLength >= min_points:

                Append [peakStart, peakEnd] to peaks

            peakStart = None

            peakEnd = None

            peakLength = 0

    Return peaks
```

min_intensity_threshold = green

min_points = 4

5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1



peakStart = 17

peakEnd = 23

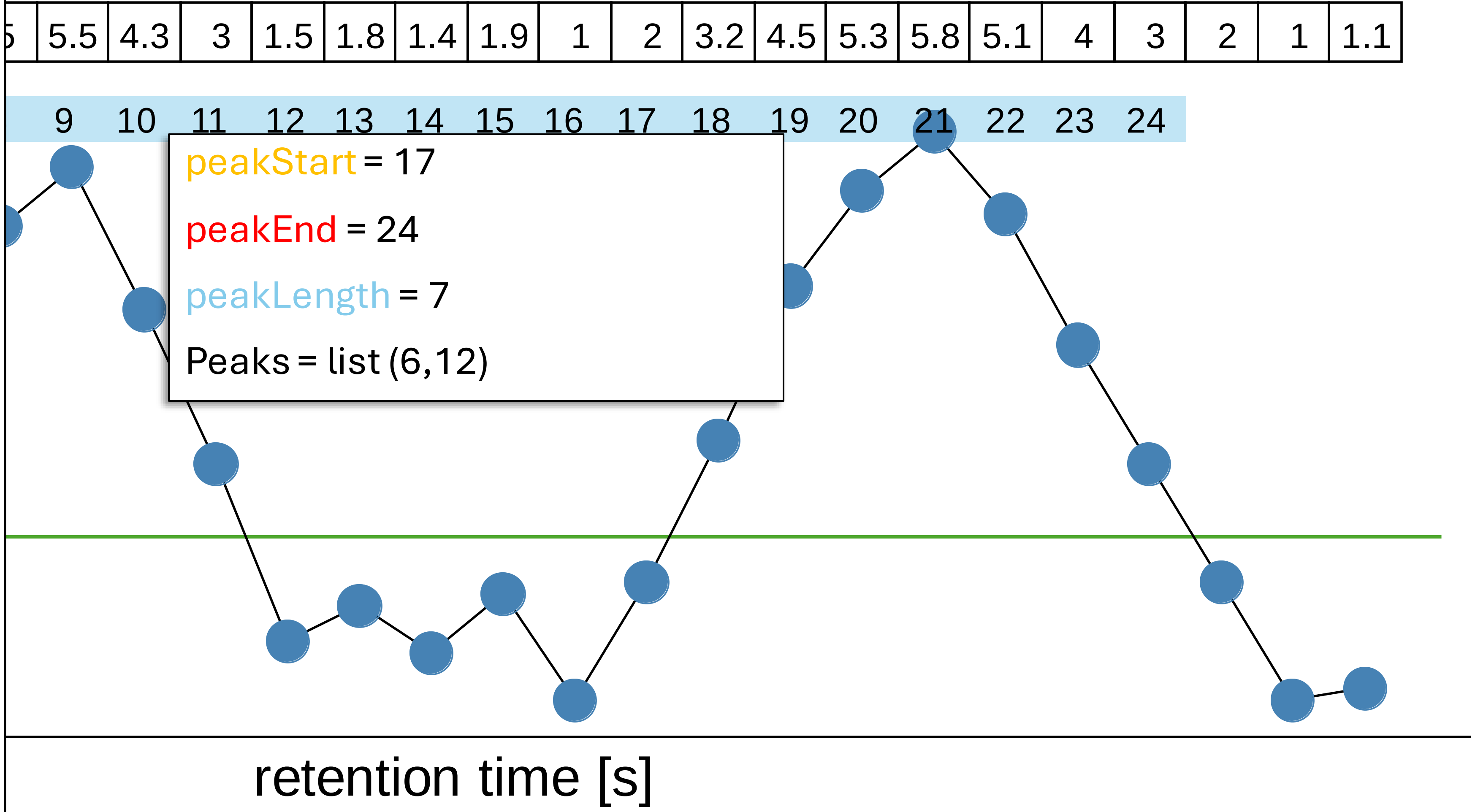
peakLength = 6

Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list
    peakStart = None
    peakEnd = None
    peakLength = 0

    For each index in signal_vector:
        value = signal_vector[index]
        If value > min_int_threshold :
            If peakStart is None: X
                peakStart = index - 1 if index > 1 else 1
            peakEnd = index
            peakLength = peakLength + 1
        Else if peakStart is not None: X
            If peakLength >= min_points:
                Append [peakStart, peakEnd] to peaks
            peakStart = None
            peakEnd = None
            peakLength = 0

    Return peaks
```

min_intensity_threshold = green

min_points = 4



peakStart = 17

peakEnd = 24

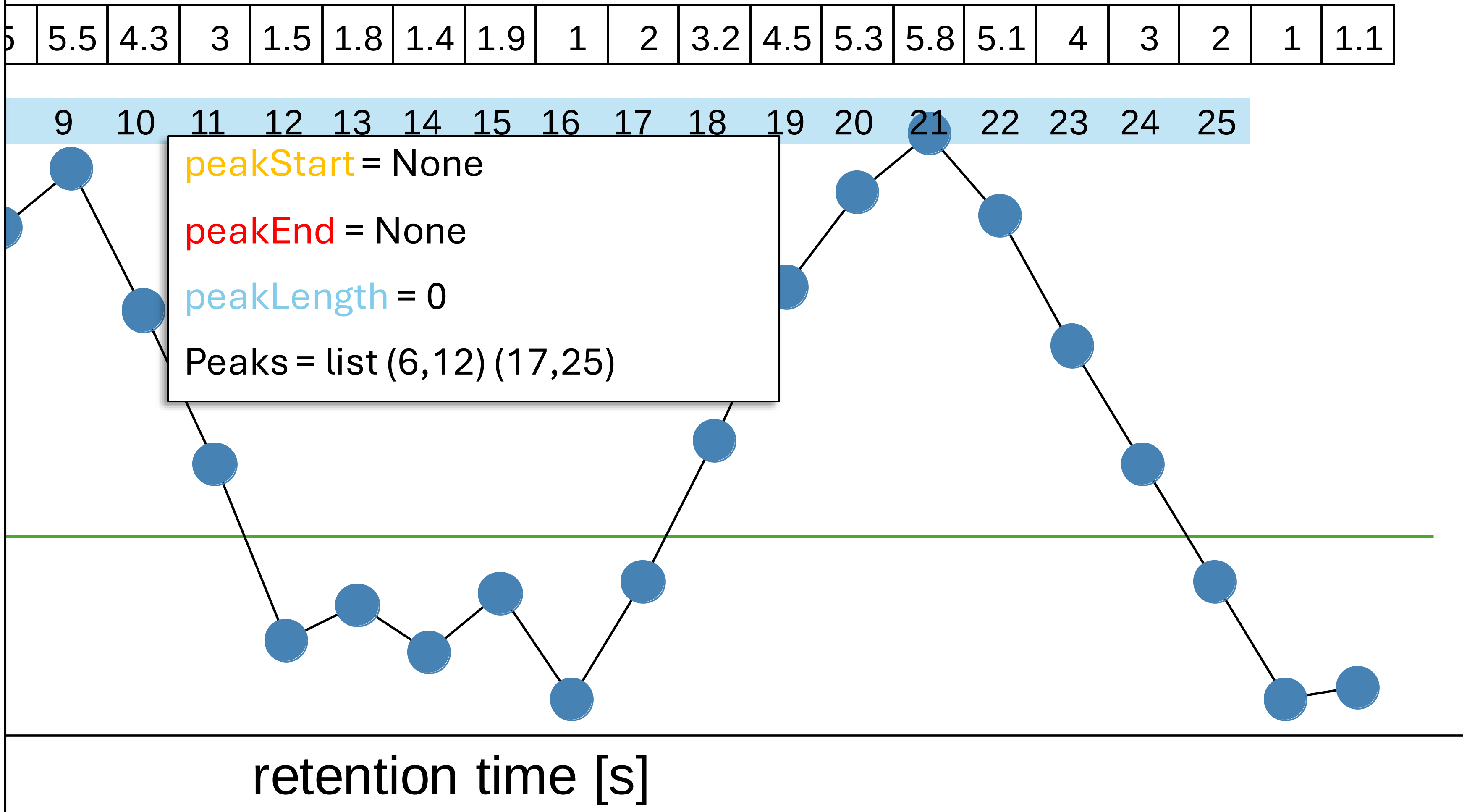
peakLength = 7

Peaks = list (6,12)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

    peaks = empty list

    peakStart = None

    peakEnd = None

    peakLength = 0

    For each index in signal_vector:

        value = signal_vector[index]

        If value > min_int_threshold : X

            If peakStart is None:

                peakStart = index - 1 if index > 1 else 1

            peakEnd = index

            peakLength = peakLength + 1

        Else if peakStart is not None:

            If peakLength >= min_points:

                Append [peakStart, peakEnd] to peaks

            peakStart = None

            peakEnd = None

            peakLength = 0

    Return peaks
```

min_intensity_threshold = green

min_points = 4



peakStart = None

peakEnd = None

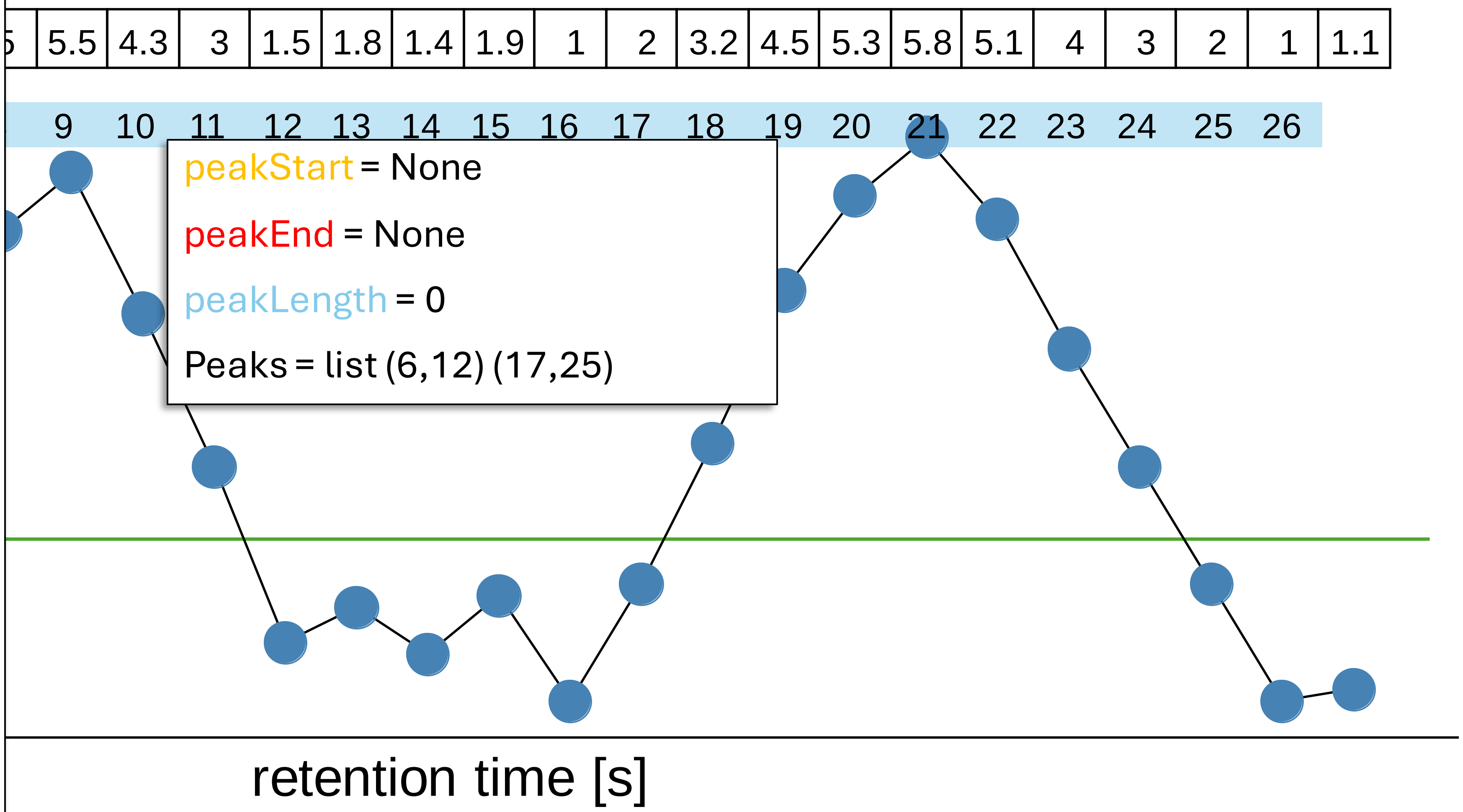
peakLength = 0

Peaks = list (6,12) (17,25)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold : X
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None: X
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

  Return peaks
```

min_intensity_threshold = green

min_points = 4



5 | 5.5 | 4.3 | 3 | 1.5 | 1.8 | 1.4 | 1.9 | 1 | 2 | 3.2 | 4.5 | 5.3 | 5.8 | 5.1 | 4 | 3 | 2 | 1 | 1.1

9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26

peakStart = None

peakEnd = None

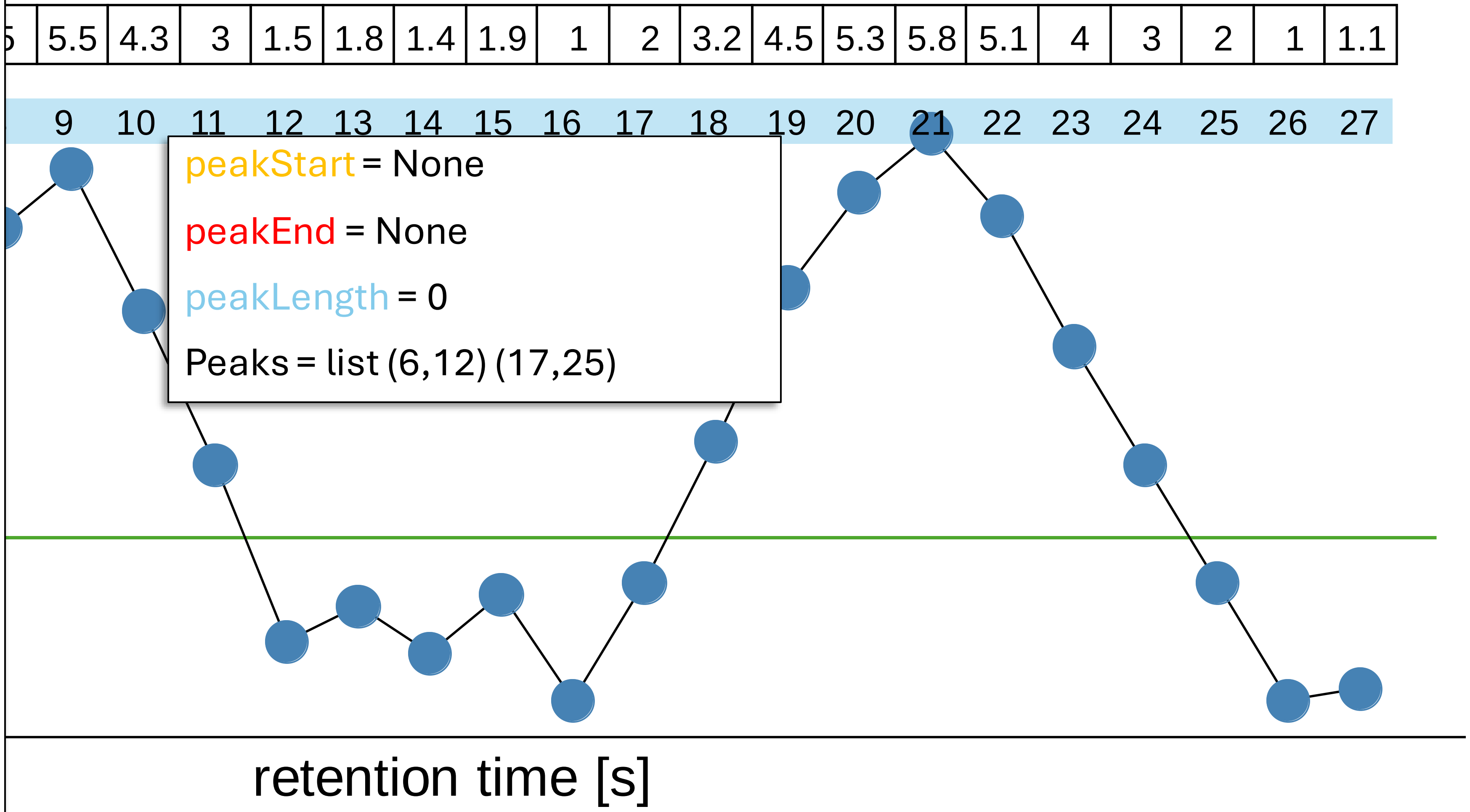
peakLength = 0

Peaks = list (6,12) (17,25)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list

  peakStart = None

  peakEnd = None

  peakLength = 0

  For each index in signal_vector:

    value = signal_vector[index]

    If value > min_int_threshold : X

      If peakStart is None:

        peakStart = index - 1 if index > 1 else 1

      peakEnd = index

      peakLength = peakLength + 1

    Else if peakStart is not None: X

      If peakLength >= min_points:

        Append [peakStart, peakEnd] to peaks

      peakStart = None

      peakEnd = None

      peakLength = 0

  Return peaks
```

min_intensity_threshold = green

min_points = 4



5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1
	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27

peakStart = None

peakEnd = None

peakLength = 0

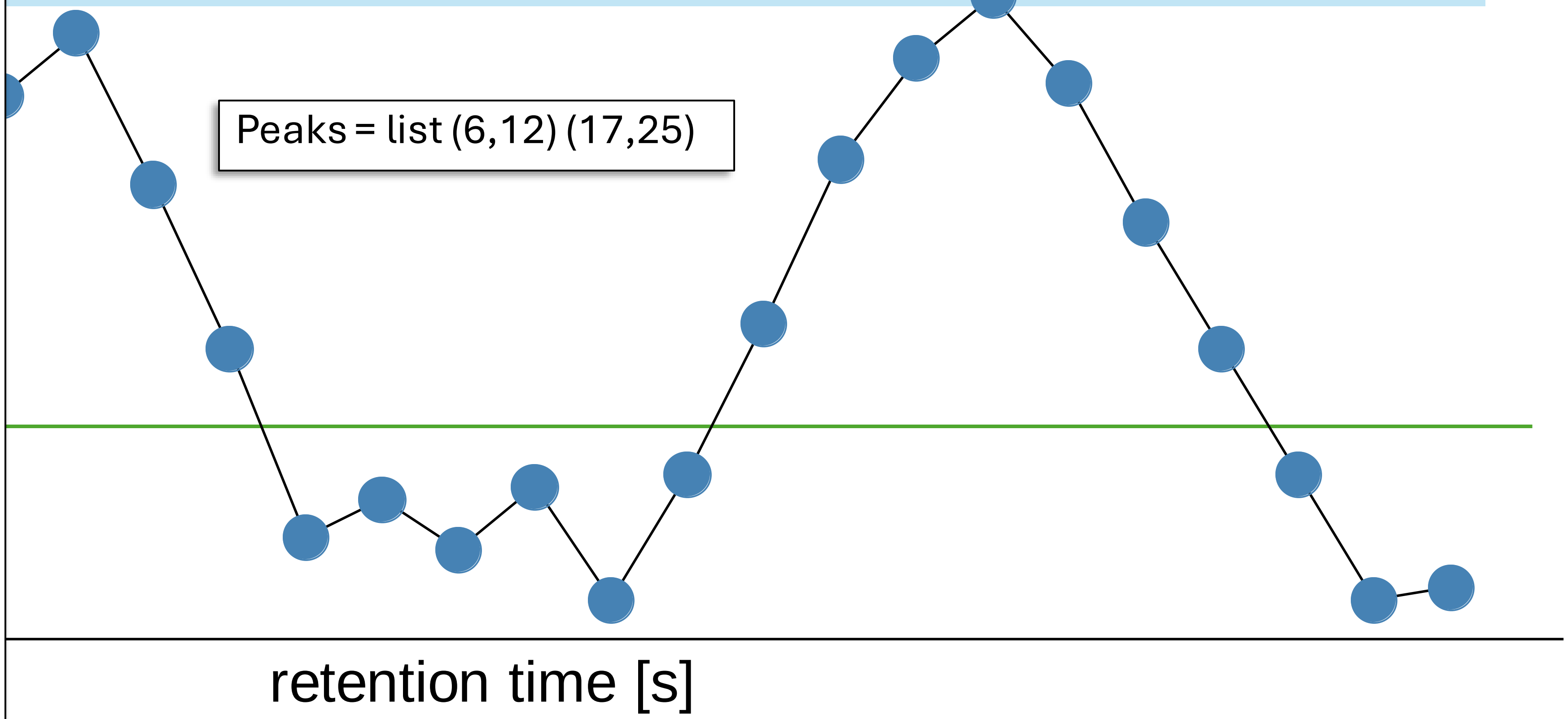
Peaks = list (6,12) (17,25)

```
Algorithm FindPeaksWithThreshold
(signal_vector, min_int_threshold, min_points)

  peaks = empty list
  peakStart = None
  peakEnd = None
  peakLength = 0

  For each index in signal_vector:
    value = signal_vector[index]
    If value > min_int_threshold :
      If peakStart is None:
        peakStart = index - 1 if index > 1 else 1
      peakEnd = index
      peakLength = peakLength + 1
    Else if peakStart is not None:
      If peakLength >= min_points:
        Append [peakStart, peakEnd] to peaks
      peakStart = None
      peakEnd = None
      peakLength = 0

Return peaks
```

min_intensity_threshold = green

min_points = 4

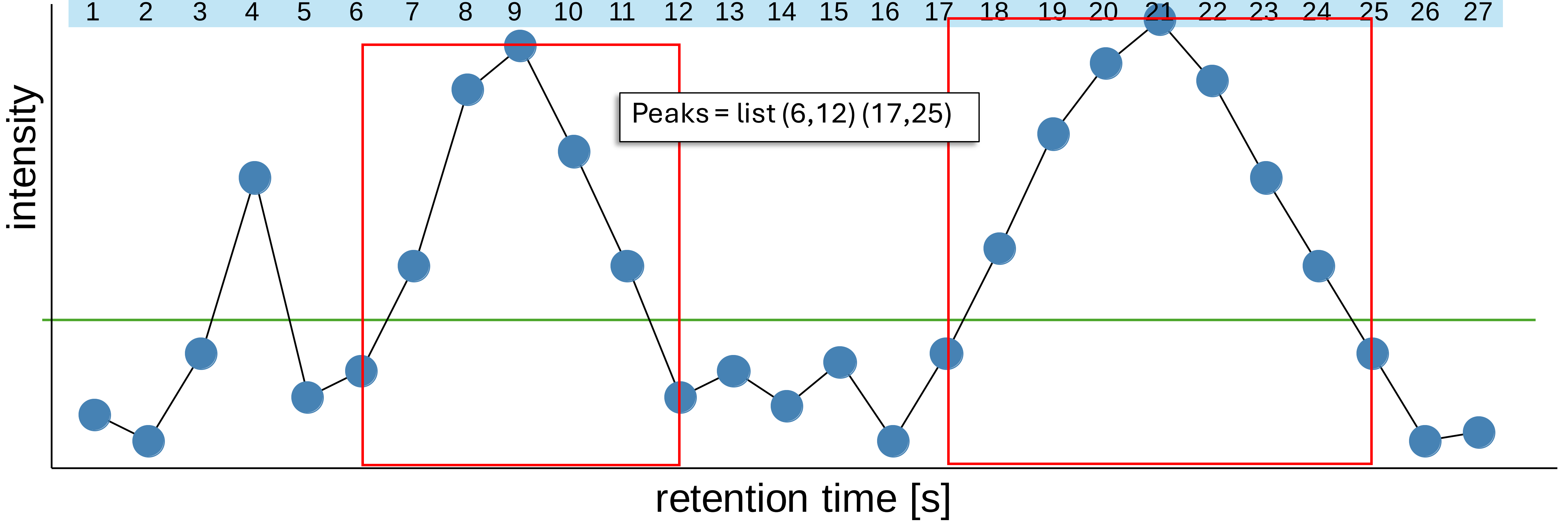
5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1
	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27



Peaks = list (6,12) (17,25)

Pseudo code algorithm

1.3	1	2	4	1.5	1.8	3	5	5.5	4.3	3	1.5	1.8	1.4	1.9	1	2	3.2	4.5	5.3	5.8	5.1	4	3	2	1	1.1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27



Peaks = list (6,12) (17,25)