Title: Rapid prototyping and parametric optimization of plastic acoustofluidic devices for blood–bacteria separation

Journal: Biomedical Microdevices

Authors: R. Silva, P. Dow, R. Dubay, C. Lissandrello, J. Holder, D. Densmore, and J. Fiering

Corresponding Author: Ryan Silva
Boston University
rjsilva@bu.edu

Algorithm 1: Find Peaks. Finds all local maxima in D and returns them as a list of peaks P.

Data: Ordered data set D
Result: A list of peaks $P \subset D$
begin
  $P \longleftarrow \emptyset$
  for $d_i \in D$ **do**
    Find all *neighbors* for $d_i \mid |i - i'| = 1$ `AppendToP`(*neighbors*)
  end
end

Algorithm 2: Calculate Prominence. Draw a horizontal line to the left (low) and right (high) of the peak until the end of the signal is reached or until the signal is intersected. Record the indices of each as i_{low} and i_{high}, respectively. Find the minima in each set and use the maximum of the minima to set the reference level. Calculate a peak's prominence by subtracting the reference level from the raw signal value of the peak.

Data: Ordered data set D and a list of local maxima $P \subset D$
Result: A list of prominence values, *Prom*, for each local maxima, p

```
begin
    Prom ⟵ ∅
    /* Define Scan Regions                                  */
    for p_j ∈ P do
        /* Scan Low                                         */
        for i = 0 to x(p_j) do
            i_low = x(p_j)
            Exit Loop
        end
        else if s[x(p_j) − i] ≥ s[x(p_j)] ∨ x(p_j) − i = 0 then
            i_low = x(p_j) − i
            Exit Loop
        end
    end
    /* Scan High                                            */
    for i = 0 to N − 1 do
        if x(p_j) = N − 1 then
            i_high = x(p_j)
            Exit Loop
        end
        else if s[x(p_j) + i] ≥ s[x(p_j)] ∨ x(p_j) + i = N − 1 then
            i_high = x(p_j) + i
            Exit Loop
        end
    end
    s_low = min(s[i_low] → s[x(p_j)])
    s_high = min(s[x(p_j)] → s[i_high])
    s_ref = max(s_low, s_high)
    Prom_j = s[x(p_j)] − s_ref
    AppendToWidth(Prom_j)
end
```

Algorithm 3: Calculate Peak Width at Half Prominence. Draw a horizontal line to the left (-) and right (+) at the median of the prominence line until either the end of the signal is reached or until the signal is intersected. Record the indices of each as i^- and i^+, respectively. The absolute value of the difference between these index values is the peak width at half prominence.

```
Data: A list of prominence values, Prom, for each local maxima, p
Result: A list of half-prominence peak widths HalfPromWidth, for each local
        maxima, p
begin
    Prom ⟵ ∅
    for p_k ∈ P do
        /* Scan Left                                        */
        for i = 0 to x(p_k) do
            if x(p_k) = 0 then
                i^- = x(p_k)
                Exit Loop
            end
            else if s[x(p_k) - i] ≤ s[x(p_k)] - Prom_k/2 ∨ x(p_k) - i = 0 then
                i^- = x(p_k) - i
                Exit Loop
            end
        end
        /* Scan Right                                       */
        for i = 0 to N - 1 do
            if x(p_k) = N - 1 then
                i^+ = x(p_k)
                Exit Loop
            end
            else if s[x(p_k) + i] ≤ s[x(p_k)] - Prom_k/2 ∨ x(p_k) + i = N - 1 then
                i^+ = x(p_k) + i
                Exit Loop
            end
        end
        HalfPromWidth_k = i^+ - i^-
        AppendToWidth(HalfPromWidth_k)
    end
end
```