# Single Linkage Clustering

*Parisimita Das*

*15 March 2017*

### Single Linkage Criterion

In Single Linlage Criterion of Agglomerative Hierarchical clustering, The distance between the elements of each cluster is minimised. As the Naive single linkage clustering has a worst case time complexity of O(N^3). It becomes useless for large data which are frequency used in the field of bioinformatics.

Here by restricting the data to h-band similarity matrix and using min-heap we reduce the time complexity to O(plogp + ph)

## The Input

The input is taken as Cosine Similarity matrix which is then converted into Distance Matrix in file adjClust-Band_heap.R

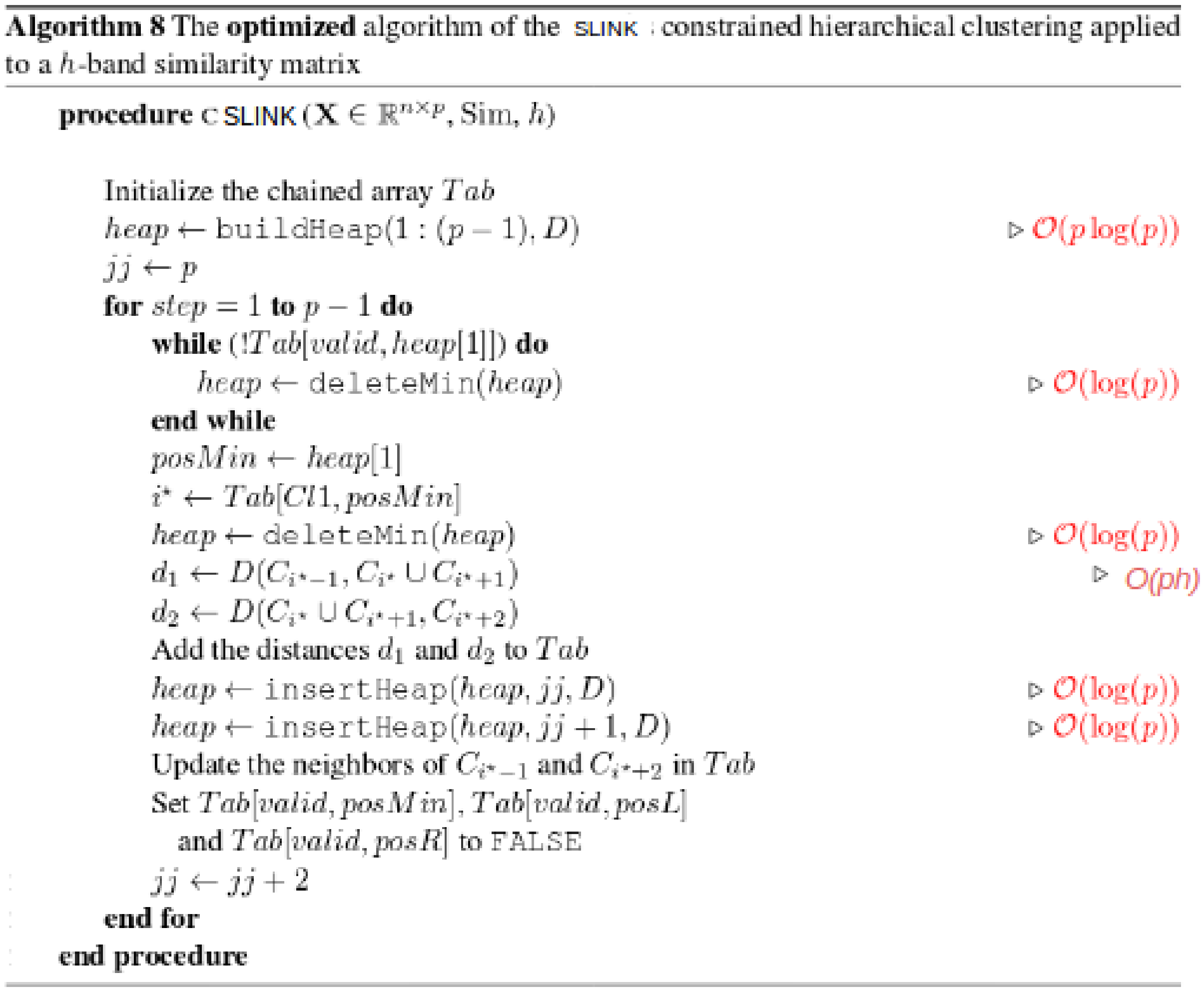
```
matL <- .toMatLeft(xt, p, h)     ## a matrix p x h (with zeros at the bottom) of the LD values
dissim <- as.numeric(1 - matL)    ## cosine distance metric
```

## The Output

The output is a hclust object with details of gain, height, label, merge data, order etc.

```
tree <- list(traceW=traceW,
                 gains=gains,
                 merge = res,
                 height = height,
                 seqdist = height,
                 order = 1:p,
                 labels = paste("",1:p),
                 method = "adjclust-heaps",
                 call = match.call(),
                 dist.method = attr(D, "method"))
    class(tree) <- "hclust"
```

## The Algorithm



**Algorithm 8** The **optimized** algorithm of the SLINK constrained hierarchical clustering applied to a $h$-band similarity matrix

```
procedure cSLINK($\mathbf{X} \in \mathbb{R}^{n \times p}$, Sim, $h$)

    Initialize the chained array $Tab$
    $heap \leftarrow$ buildHeap($1 : (p-1), D$)          ▷ $\mathcal{O}(p\log(p))$
    $jj \leftarrow p$
    for $step = 1$ to $p - 1$ do
        while (!$Tab[valid, heap[1]]$) do
            $heap \leftarrow$ deleteMin($heap$)          ▷ $\mathcal{O}(\log(p))$
        end while
        $posMin \leftarrow heap[1]$
        $i^\star \leftarrow Tab[Cl1, posMin]$
        $heap \leftarrow$ deleteMin($heap$)              ▷ $\mathcal{O}(\log(p))$
        $d_1 \leftarrow D(C_{i^\star-1}, C_{i^\star} \cup C_{i^\star+1})$          ▷ $O(ph)$
        $d_2 \leftarrow D(C_{i^\star} \cup C_{i^\star+1}, C_{i^\star+2})$
        Add the distances $d_1$ and $d_2$ to $Tab$
        $heap \leftarrow$ insertHeap($heap, jj, D$)      ▷ $\mathcal{O}(\log(p))$
        $heap \leftarrow$ insertHeap($heap, jj+1, D$)    ▷ $\mathcal{O}(\log(p))$
        Update the neighbors of $C_{i^\star-1}$ and $C_{i^\star+2}$ in $Tab$
        Set $Tab[valid, posMin]$, $Tab[valid, posL]$
          and $Tab[valid, posR]$ to FALSE
        $jj \leftarrow jj + 2$
    end for
end procedure
```

## Computation of Minimum Distance via SLINK

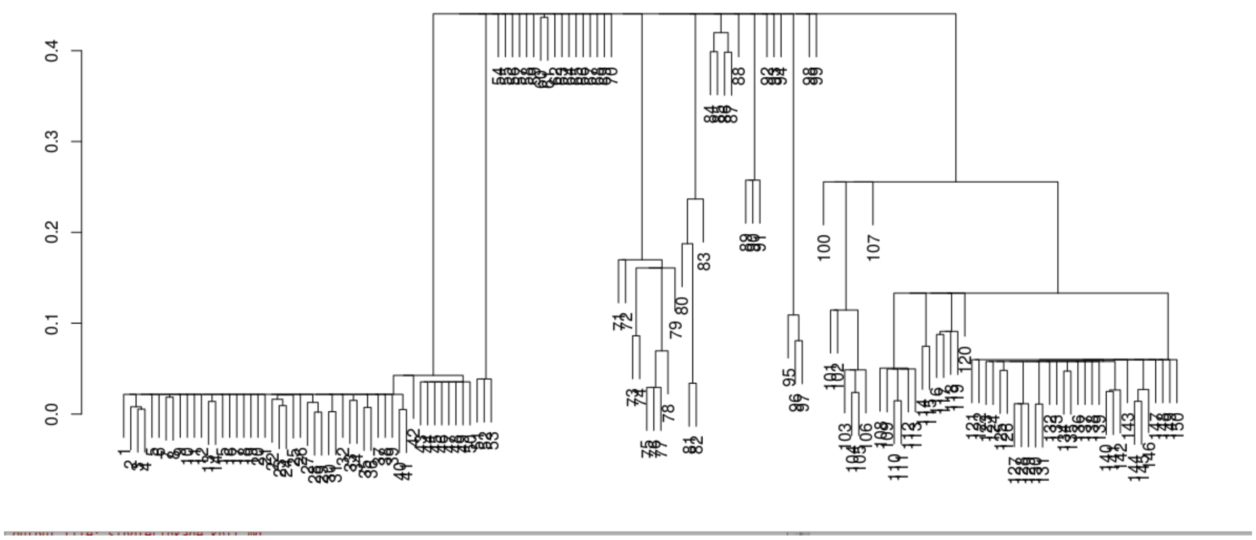
```
  for(int i=mini;i<=maxi;i++)
  {
    for(int j=minj-i;(j<=maxj-i)&&(j<=h);j++)
    {
    res = MIN(DIS(i,j,p),res);
    }
}
```

Where the number of items in cluster I is mini - maxi and cluster J is minj - maxj. In loop-2 the constrain that the difference between index of item must not exceed h is applied.

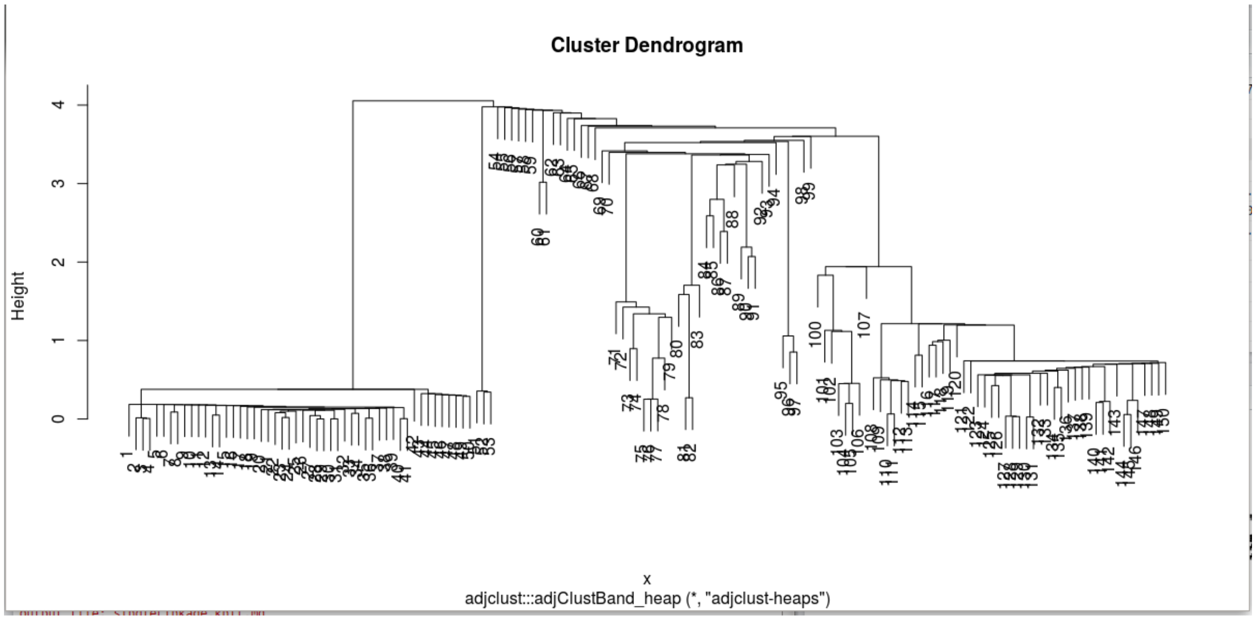
## Comparision with Rioja

Taking parameter h as the width of the matrix we get similar results as chclust function of rioja with conslink method.

- The output by rioja chclust (conslink)



- The output by adjClustBand_heap

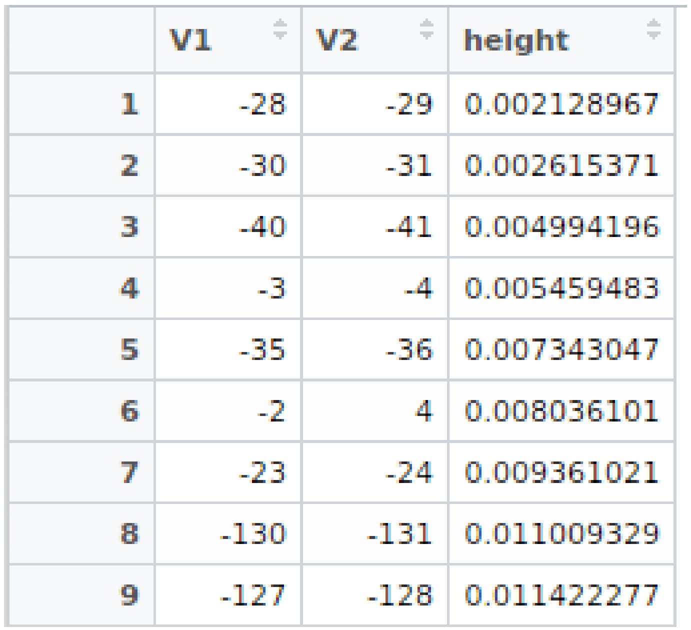


Comparing the Merge Data of both the functions we get same values initially and it deviates a little at higher heights.

- The Merge Data by chclust (conslink)

```
#Recalculating the similarity matrix as s_re = 2-2*s' and rechecking the rioja cluster value
s_re <- 2 - 2*s_scaled
```

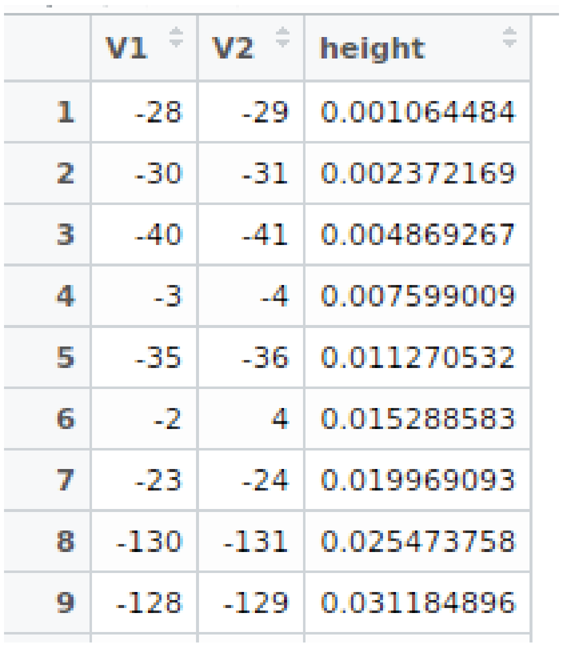
```
#rioja hclust object
diss <- as.dist(s_re)
clust <- chclust(diss, method = "conslink")
View(cbind(clust$merge ,height = clust$height))
```



| | V1 | V2 | height |
|---|---|---|---|
| 1 | -28 | -29 | 0.002128967 |
| 2 | -30 | -31 | 0.002615371 |
| 3 | -40 | -41 | 0.004994196 |
| 4 | -3 | -4 | 0.005459483 |
| 5 | -35 | -36 | 0.007343047 |
| 6 | -2 | 4 | 0.008036101 |
| 7 | -23 | -24 | 0.009361021 |
| 8 | -130 | -131 | 0.011009329 |
| 9 | -127 | -128 | 0.011422277 |

- The Merge Data by adjClustBand_heap

```
# hclust object using adjClustBand_heap function
# where s_scaled is the similatrity matrix
h<- ncol(s_scaled)
fit <- chaclust(s_scaled,h,f = 1)
plot(fit)
View(cbind(fit$merge ,height = fit$height))
```



| | V1 | V2 | height |
|---|---|---|---|
| 1 | -28 | -29 | 0.001064484 |
| 2 | -30 | -31 | 0.002372169 |
| 3 | -40 | -41 | 0.004869267 |
| 4 | -3 | -4 | 0.007599009 |
| 5 | -35 | -36 | 0.011270532 |
| 6 | -2 | 4 | 0.015288583 |
| 7 | -23 | -24 | 0.019969093 |
| 8 | -130 | -131 | 0.025473758 |
| 9 | -128 | -129 | 0.031184896 |

Though the Height is different, the overall height of dendrogran is almost same around ~0.4 and the merge data of clusters are same as shown in the data table.

---

References:

- https://tel.archives-ouvertes.fr/tel-01288568/en
- http://www.geeksforgeeks.org/greedy-algorithms-set-5-prims-minimum-spanning-tree-mst-2/