Visualizing Hierarchical Clustering in Iterated Logarithmic Scales

Sung-Hyuk Cha

Seidenberg School of CSIS, Pace University, White Plains NY, 10606, USA
scha@pace.edu

Abstract

Clustering data has been of great interest to many researchers. Hierarchical clustering methods have been preferred because clusters can be visualized as a dendrogram. One of the problems of hierarchical clustering methods, however, is that the resulting dendrogram is not visually pleasing due to the scaling problem. Hence, a series of iterated logarithmic function is proposed so as to mitigate the scaling problem. Theoretical properties of the iterated logarithmic function are presented.

I. Introduction

Clustering is one of the most important data analysis concepts and numerous methods have been studied [1]. Amongst, hierarchical clustering methods [1,2] have been widely used especially in biological taxonomy [3] and bioinformatics [4] due to its outstanding visual output. They produce a visual output, so called ‘dendrogram’ which is a tree representation of the hierarchical clusters. However, the dendograms are often not scaled well for users to see, e.g., Fig 1 which appears in [5]. In Fig 1, 150 Fisher’s Iris flower data [6] represented by four features, {Sepal length, Sepal width, Petal length, Petal width} are clustered using the cosine measure to compute the distance between data.

Logarithms, invented by John Napier in 1615, have been used in many scientific areas [7]. Among many usages of the logarithm function, it has been widely used for the visualization purpose. Instead of ordinal scale, logarithmic scale allows to display a large range of data into visually pleasing range as given in the Fig 2. Most commercial data plotting software products provide both linear and logarithmic scales and occasionally logarithmic scales are preferred to linear scales.

Fig. 1. a sample dendrogram on the Fisher's Iris data set using the cosine measure [5].

(a) 2D plot in linear scales
Despite popularity of the logarithmic scale, it has a problem when data is ranged between 0 and 1, especially when it is used for drawing dendrograms. Hence, iterated logarithmic functions are considered and a new function denoted as \( vlog \) is formally defined. Albeit it is one way of scaling the linear values in a different scale, \( vlog \) allows various levels of displaying dendrograms iteratively for users to choose.

The rest of the paper is organized as follows. In section 2, the dendrogram of a hierarchical clustering is reviewed to raise an exemplary problem of linear and logarithmic scales. Section 3 introduces a new function, \( vlog \) which scales data iteratively. Section 4 concludes this work.

### II. Hierarchical Clustering

The dendrogram is a tree representation of the hierarchical clusters of data. One of the techniques to build a dendrogram is using the agglomerative single linkage clustering method [1] and a naïve algorithm is given in the Algorithm 1 (see [2] for efficiently algorithms and their computational complexities).

#### Algorithm 1 Naïve algorithm for agglomerative single linkage

| Input: | 1. \( n \) number of data \( D = \{d_1, \ldots, d_n\} \)  
|        | 2. a distance measure \( d \)  
|        | 3. User option = min, max, or average. |
| Output: | hierarchical cluster |
| Procedure: | Compute the upper triangle distance matrix \( M \) using the \( d \) for all pairs of clusters. |
|            | Begin with \( n \) clusters, each cluster corresponds to one sample, e.g., \( C_1 = \{d_1\} \), etc. |
|            | Repeat  
|            | Find the most similar clusters \( C_i \) and \( C_j \) and merge into one cluster \( C_{ij} \). |

To illustrate the algorithm 1, consider the following five input data. If the distance measure \( d \) is the Euclidean distance between two clusters, then the distance matrix \( M \) is generated.

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>4</td>
<td>8</td>
<td>15</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>( y )</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>12</td>
</tr>
</tbody>
</table>

\[
M = \begin{bmatrix}
0 & 4 & 11.7 & 20 & 21.5 \\
4 & 0 & 8.1 & 16 & 17.9 \\
11.7 & 8.1 & 0 & 9.8 & 9.8 \\
20 & 16 & 9.8 & 0 & 8 \\
21.5 & 17.9 & 9.8 & 8 & 0 \\
\end{bmatrix}
\]

As long as the distance measure \( d \) is metric, only the upper triangle distance matrix is necessary for the algorithm. As described in the algorithm 1, the first step is assigning each instance to each cluster. Next, merge the clusters which have the minimum value in \( M \) between them. To fill the new \( M \), the distances between the merged cluster and others have three choices, i.e., minimum, maximum, or average. In general, the following equations are defined for these choices.

\[
d(C_{x,y}, C_2) = \min(d(C_{x}, C_2), d(C_{y}, C_2)) \quad (1) \\
d(C_{x,y}, C_2) = \max(d(C_{x}, C_2), d(C_{y}, C_2)) \quad (2) \\
d(C_{x,y}, C_2) = \text{average}(d(C_{x}, C_2), d(C_{y}, C_2)) \quad (3)
\]

In Table 1, the minimum option in the eqn (1) is used for the simplicity sake. The vertical scale on the left of the dendrogram represents distance values between clusters and the horizontal scale represents input data. In all, the dendrogram provides excellent visual groupings of data.
Table 1. Step by step illustration of agglomerative single linkage algorithm 1

<table>
<thead>
<tr>
<th>Step</th>
<th>Clusters</th>
<th>Distance matrix $M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$C_1 = {d_1}$, $C_2 = {d_2}$, $C_3 = {d_3}$, $C_4 = {d_4}$, $C_5 = {d_5}$</td>
<td>$C_1$, $C_1$, $C_1$, $C_1$, $C_1$, $M_1 = 8.1$, $16$, $17.9$, $8$</td>
</tr>
<tr>
<td>2</td>
<td>$C_{1,2} = {d_1, d_2}$, $C_3 = {d_3}$, $C_4 = {d_4}$, $C_5 = {d_5}$</td>
<td>$C_1$, $C_1$, $C_1$, $C_1$, $C_1$, $M_2 = 8.1$, $16$, $17.9$, $8$</td>
</tr>
<tr>
<td>3</td>
<td>$C_{1,2} = {d_1, d_2}$, $C_3 = {d_3}$, $C_{4,5} = {d_4, d_5}$</td>
<td>$C_1$, $C_1$, $C_1$, $C_1$, $C_1$, $M_3 = 8.1$, $16$, $17.9$, $8$</td>
</tr>
<tr>
<td>4</td>
<td>$C_{1,2,3,4,5} = {d_1, d_2, d_3, d_4, d_5}$</td>
<td>$C_1$, $C_1$, $C_1$, $C_1$, $C_1$, $M_4 = 8.1$, $16$, $17.9$, $8$</td>
</tr>
</tbody>
</table>

As a result, the dendrogram in Figure 3 is generated.

Fig. 3. Sample Dendrogram

Note that the distance measure $d$ must be given by the user and numerous normalized distance or similarity measures such as angular metrics and correlation based measures give values between 0 and 1 [9]. When the logarithm is applied to these distance values, outputs are negative values and the serious ‘log of zero’ problem is encountered, i.e., $\log(0) = \infty$ and $\log(1) = 0$.

Table 1. Step by step illustration of agglomerative single linkage algorithm 1

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Distance matrix $M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bov</td>
<td>0.0179 0.9623 0.9764 0.9858 0.9906</td>
</tr>
<tr>
<td>Moo</td>
<td>0.9623 0.9575 0.9717 0.9811 0.9858</td>
</tr>
<tr>
<td>Gib</td>
<td>0.9764 0.9717 0.0236 0.0189 0.0330</td>
</tr>
<tr>
<td>Ora</td>
<td>0.9958 0.9811 0.0030 0.0142 0.0089</td>
</tr>
<tr>
<td>Gor</td>
<td>0.9906 0.9858 0.0377 0.0326 0.0142</td>
</tr>
<tr>
<td>Chim</td>
<td>0.0179 0.9623 0.9764 0.9858 0.9906</td>
</tr>
</tbody>
</table>

Fig. 4. A sample distance matrix and its problematic dendrograms

To illustrate how commercial plotting packages handle the normalized distance cases, consider the distance matrix in Fig 4 (a). All values in the distance matrix are between zero and one. The agglomerative single linkage with the average option is used and Fig 4 (b) shows its dendrogram in the linear scale. Typically, several data are clamped in the lower distance value range and thus dendrogram in the linear scale does not allow for users to distinguish them visually. Fig 4 (c) shows the dendrogram in the logarithmic scale using a commercial plotting tool by Matlab. While the logarithmic scale in the Fig 4 (c) seems to be slightly better than the linear scale, ‘Chimp’ and ‘Gorilla’ are no longer visually distinguishable. Better scaling system is necessary.

III. Vlog function

Let $x$ be a real number between 0 and 1. While $\log_2(x)$ has values between $\infty$ and 0, $\log_2(1 + x)$ has values between 0 and 1. At first glance, $\log_2(1 + x)$ seems to be a good choice for logarithmic scaling. Unfortunately, $x \approx \log_2(1 + x)$ when $x$ is very small. After all, $\log_2(1 + x)$ does not scale the dendrograms.
well, either. Hence, iterated logarithmic scaling is proposed by borrowing the idea of functional iteration in [10].

**Definition 1:** $p$-iterated $vlog$

$$vlog^p(x) = \begin{cases} 
\log_2(1 + x) & \text{if } p = 1 \\
\log_2(1 + vlog^{p-1}(x)) & \text{if } p > 1 
\end{cases}$$

where $p$ is a positive integer

$vlog$ can be defined as in the following pseudocodes.

**Pseudocode 1:** $vlog$ using recursive function.

```plaintext
function y = vlog(x, p)
    if p == 1
        y = log2(1 + x);
    else
        y = log2(1 + vlog(x, p - 1));
    end
end
```

**Pseudocode 2:** $vlog$ using the iterative loop.

```plaintext
function y = vlog(x, p)
y = log2(1 + x);
for i = 2 to p
    y = log2(1 + y);
end
end
```

e.g.,

$vlog^1(x) = \log_2(1 + x)$
$vlog^2(x) = \log_2(1 + \log_2(1 + x))$
$vlog^3(x) = \log_2(1 + \log_2(1 + \log_2(1 + x)))$
$vlog^4(x) = \log_2(1 + vlog^3(x))$

\ldots

$vlog^{19}(x) = \log_2(1 + vlog^{18}(x))$
$vlog^{20}(x) = \log_2(1 + vlog^{19}(x))$

Fig 5 shows the plots for $vlog^1(x)$ to $vlog^{20}(x)$. Notice that when $p$ gets larger, it can visualize the smaller range much better. Fig 6 displays the several different $p$-iterated $vlog$ scales of Fig 3 (a).
The *iterated logarithm function* is well known as $\lg^* n$ or ‘log star of $n$’ in the analysis of algorithms [10]. Albeit conceptually similar, it should be noted that $\lg^* n$ is not defined for $n < 1$ and we are primarily interested in data between 0 and 1. Hence, $\text{vlog}^p(n)$ clearly differs from $\lg^* n$. Nevertheless, here are some facts about asymptotic growth [10] of vlog functions;

**Fact 1.** $\text{vlog}^p(n) = o(n)$ where $p$ is any positive integer.

**Fact 2.** $\text{vlog}^p(n) = \omega(\text{vlog}^q(n))$ where $p$ and $q$ are positive integers such that $p < q$.

**Fact 3.** $\text{vlog}^1(n) = \Theta(\log(n))$.

**Fact 4.** $\text{vlog}^p(n) = \omega(c)$ where $p$ is a positive integer and $c$ is a constant.

In terms of growth of functions, vlog functions grow between constant and log functions from Fact 3 and 4.

**Fact 5.** $\text{vlog}^p(n)$ converges to 1 as $p$ approaches to $\infty$ (as depicted in Fig 9).

$$\lim_{p \to \infty} \text{vlog}^p(n) \approx 1$$

<table>
<thead>
<tr>
<th>$\text{vlog}^0(10^{10})$</th>
<th>$\text{vlog}^1$</th>
<th>$\text{vlog}^2$</th>
<th>$\text{vlog}^{10}$</th>
<th>$\text{vlog}^{20}$</th>
<th>$\text{vlog}^{100}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>332.1928</td>
<td>8.3802</td>
<td>3.2296</td>
<td>1.0030</td>
<td>1.0001</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Fig. 8. Growth of vlog functions

Fig. 9. vlog values at $10^{100}$.
**IV. Discussion**

In this paper, excellent functions denoted as vlog\(^p\) that can scale the micro level data is introduced to visualize dendrograms better. Facts 1~4 in the macro level are completely opposite in the micro level as depicted in Fig 10. In micro level, i.e., \(x\) is very small, vlog\(^1\)(\(x\)) can approximate \(y = x\) function whereas in the macro level, i.e., \(x\) is very large, vlog\(^p\)(\(x\)) where \(p\) is very large can approximate \(y = 1\) function as in the Fact 5.

![Fig. 9. Growth of vlog values at 10^100.](image)

In this article, vlog had its base 2 by default. If different log bases such as \(b = \text{the natural number e}\) or \(b = 10\) are preferred, the definition 1 should be generalized to as following.

**Definition 2:** different base vlog series

\[
v\log^p_b(x) = \begin{cases} 
\log_b(b-1+x) & \text{if } p = 1 \\
 v\log^p_{b-1}(\log_b(b-1+x)) & \text{if } p > 1 
\end{cases}
\]

**References**