Size proportional Venn and Euler diagrams in 2 and 3 dimensions: `vennplot(...)` in `R`

Abstract

Venn and Euler diagram is a popular way to visualize factor data. In this artical, we will introduce a statistic model for fitting size-proportional Venn and Euler diagram; this model is based on a loss function we defined and continuous optimization procedure for searching minimum. An `R` function `vennplot(...)` can provide both 2D and 3D layout.

1 Introduction

In recent years variations of Venn diagrams have seen increased use in scientific publications. For example, Figure 1 shows the results of an online search for the phrase "Venn diagram" over all articles from 1998

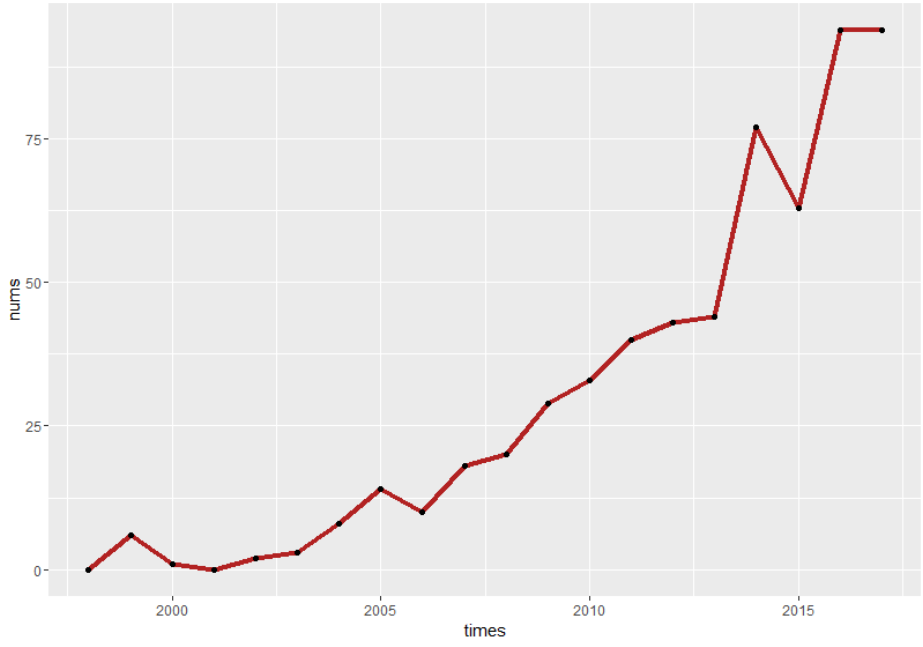


Figure 1: Number of articles containing "Venn diagram" over time from the journals *Genetics* and *Nature*

to 2017 appearing in the journals *Nature* and *Genetics* (including G3: Genes, Genomes, Genetics). As can be seen, there has been nearly a 10 fold increase since the turn of the century, particularly in genetic applications, the use of "Venn diagram" soars from 5/4972 in 1999 to 75/3208 in 2017 (due July 18th)

1.1 Definition

Venn diagrams for m component sets must contain all possible intersections (2^m). Multiple closed curves, like circles, ellipses, and other irregular polygons overlap with each other to depict Venn diagrams. The interior points of the closed curve represent the elements of the set, while the exterior points represent elements that are not in this set. Unlike Venn diagrams, Euler diagrams only contain the relevant relations [3]. In Venn diagrams, a shaded zone or figure zero may represent non-intersection, but in Euler diagrams, the corresponding zone is usually missing. Both Venn and Euler diagrams are used in illustrating samples in mathematics, nature science, genetics and other areas. However, in somes cases, the Venn diagrams and Euler diagrams are not up to the job independently and so are often stretched beyond their definitions [7]. Thus, we will not struggle the names and call it Venn and Euler diagram.

1.2 Brief description

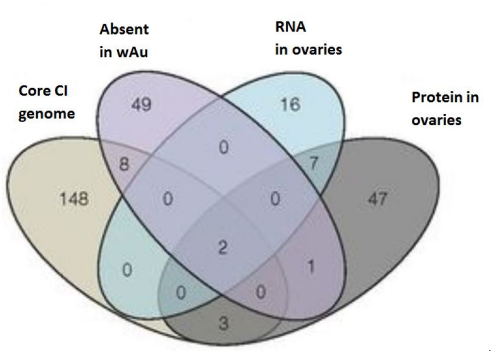
Interest often lies in the number of genes shared by different species, or perhaps by different groups of individuals. For example, in Figure 2, Venn and Euler diagram in (a) shows Wmel strain gene of Wolbachia pipientis shared in a combination of four specific criteria [15], and (b) describes genes shared by five asterid species [20]. The common features of these two graphs are that: (1) they use the same size ellipse. (2) the number of ellipses m cuts picture into 2^m disjoint areas and the size of each area does not match the counts. Venn and Euler diagrams in (c) and (d) both include six data sets: in (c), irregular polygons are drawn to illustrate six woody species. Although it contains all the disjoint intersections, the visualization of interacting characteristics is absent (13 and 4872 share the same area; the total size of "Poplar" is the largest, however, the total area of it is the third smallest); in (d), diagram is depicted by differently shaped triangles. Compared with (c), the ability of cutting area is worse and the sharp corner makes the visualization less aesthetic.

Good diagrams clarify. Very good diagrams force the ideas upon the viewer. The best diagrams compellingly embody the ideas themselves [7]. If we look at Venn and Euler diagrams (e) and (f), both of them convey the size of intersections by visualization. In (e), It is clear that majority "Sub-high level" gene ontologies share with the "High level" ones; only the half of "High level" genes partake with the "Sub" ones. In (f), based on the diagram, we can tell location and interest are the two main factors affacting friendships and age just impact a little part (13% in total)

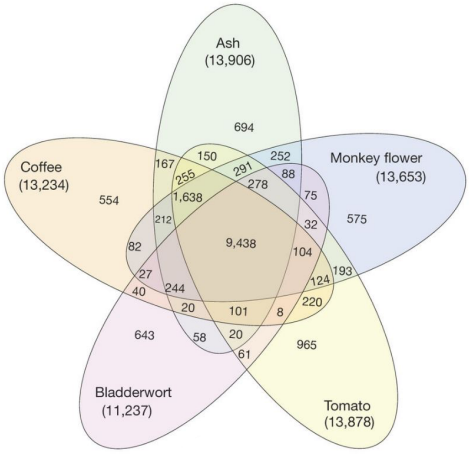
An informal survey of 112 Venn and Euler diagrams published in articles of journal *Nature* and *Genetics* in the past two years, here are some common features: (1) close to half of them (49/112) use size-proportional characteristics; (2) over two thirds of them (75/112) make Venn and Euler diagram circles and the number of circles is two or three; (3) In these 75 aritcles using circles, 39 of them contain the property of size-proportion; (4) the rest 37 articles without making Venn diagram circles, 24 of them has more than four closed curves. In other words, when the numeber of sets is smaller than four, almost all of them(75/88) make circular Venn and Euler diagrams. Figure 3 shows pie charts of these four cases. Hence, we can deduce that the majority of scientists prefer to make Venn and Euler diagram circles. But as the number of sets increases, in existing softwares, the automatical fit may not be good enough. Thus, scientists have to choose ellipse or irregular polygons to illustrate the interacting relationships. In this article, we will add a restriction to force our Venn and Euler diagrams to be size-proportional, circles when 2D and balls when 3D.

1.3 Perceptual laws

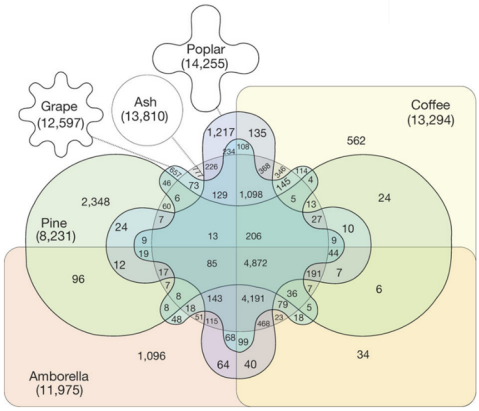
In a series of experiments, Cleveland and McGill [8–10] investigated the quality of a variety of different encodings for magnitude. On account of the visual perception theory, area and volume both give high accurancy in graphical perception experiments. Here, we need to be more cautious, in Stevens' power law [21], a person's perceived magnitude of a stimulus of magnitude varies by different perceived scales (like length, area and volume). For example, consider visually comparing two areas (volumes) of size p and q.



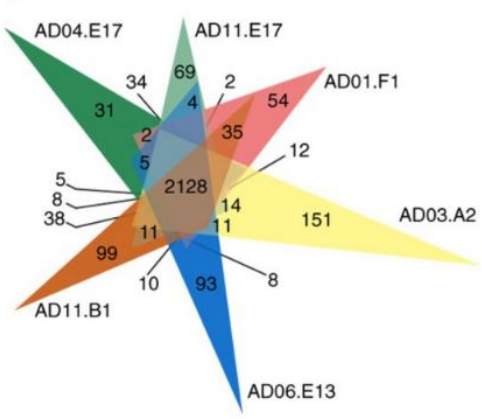
(a)



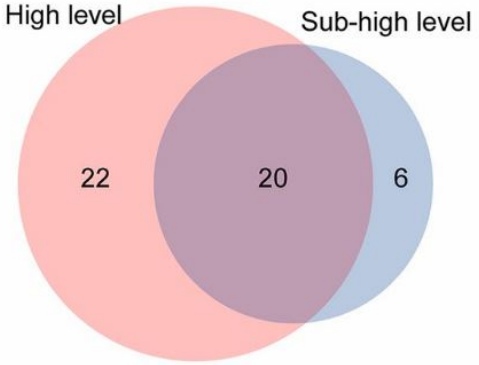
(b)



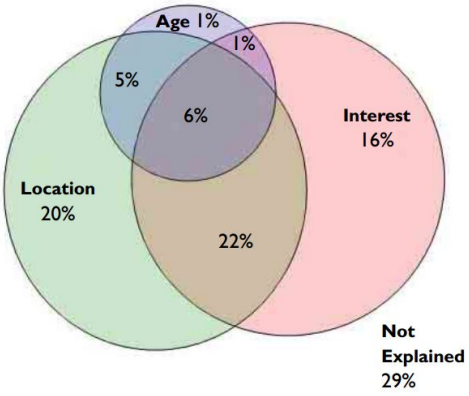
(c)



(d)



(e)



(f)

Figure 2: (a) Showing the number of wMel genes fitting these four components. [15]. (b) Genes sharing by five asterid species (coffee, ash, morkey flower, tomato, bladderwort) [20]. (c) Venn diagram of gene sharing by six woody species (pine, grape, ash, poplar, coffee, amborella) [20]. (d) Showing the number of genes shared between isolates from investigative patients [5]. (e) Illustrating the overlapping of gene ontology between the highland and subhighland lineages [25]. (f) Explaining the friendships through locations, ages and interests. (data source: www.livejournal.com) [14].

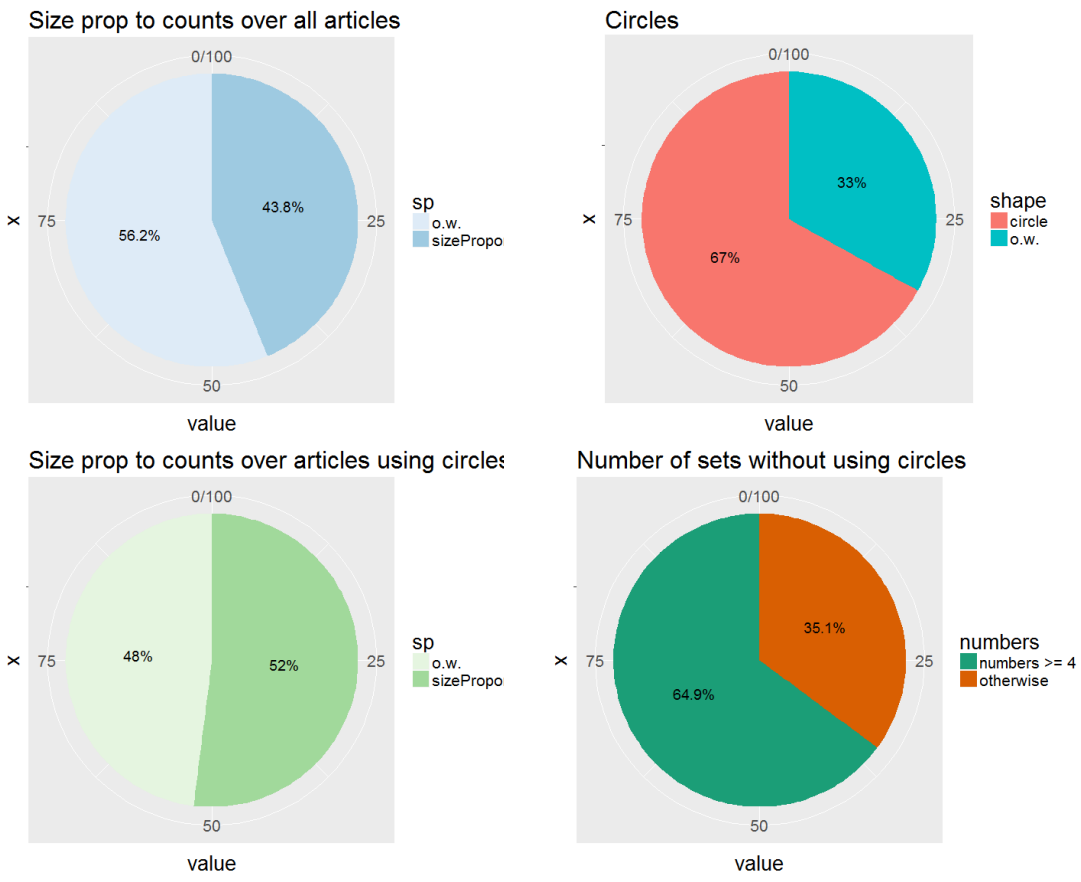


Figure 3:

According to Stevens' law, the ratio of the areas will be perceived to be

$$\left(\frac{p}{q}\right)^{\varpi}$$

instead of p/q, where $0.6 \leq \varpi \leq 0.9$ if area and $0.5 \leq \varpi \leq 0.8$ if volume. Hence, when we make circles size proportional to counts, we need to give some tolerance to our eyes.

2 Related Work

Currently, there are two main branches to the drawing of Venn and Euler diagram. One approach is more close to the defination of Venn diagram: m closed curves divide the whole picture into 2^m pieces and then yielding figure to each disjoint part. The counts do not need to be necessarily proportional to the area, like package `vennDiagram(...)` [6] and `venn(...)` in `R`. The other approach is more close to the defination of Euler diagram: a statistical model is built to produce area proportional circular Venn and Euler diagrams for one or more sets and some critical values are created to evaluate the goodness of fit, like package `venneuler(...)` [24] in `R` and `venn.js(...)` [11] in `JavaScript`.

2.1 R package `venneuler(...)`

In Wilkinson's paper [24], a statistical loss function and a minimization procedure are first invented to estimate the Venn and Euler area-proportional model. The goodness of his fit is evaluated by a critical value of null hypothesis test, *stress*. We will show more details of his work in the following sections.

Besides, he also compares his work `venneuler(...)` to `VennMaster(...)` and `Chow/Rodgers` algorithm. In his comparison, `venneuler(...)` has several adventages: (1) the quality of fit *stress*, is better than the other two. (2) `VennMaster(...)` program relies on the random seeds and the solutions give no indication, which makes it not trustworthy; `Chow/Rodgers` algorithm is limited to 3-ring generalized Venn and Euler diagrams, so superset problems cannot be handled.

2.2 Javascript `venn.js(...)`

Ben Frederickson introduces a new model to minimize a sum of squared errors function comparing the actual intersection sizes to the desired sizes. His algorithm positions the sets by optimizing the distances bewteen circles, instead of the intersection areas directly. We will discuss this work further in the following sections.

In his blog [11], he compares his work `venn.js(...)` with `venneuler(...)`. A few tests are built and `venneuler(...)` doesn't perform all that well. One of the reasons he gives is that "`venneuler(...)` frequently gets a solution that is close to being correct, it rarely gets a solution that is close enough for this test to say it succeeded" [1].

2.3 Modified model

In this article, we will start with review of the previous work, `venneuler(...)` and `venn.js(...)`, and then modify and create a new algorithm including groups detection, a loss function, continuous optimization process and groups combination; meanwhile, the layout is not restricted into two dimension, 3D version is also available. At last, we will make a comparison between `vennplot(...)` with `venneuler(...)` and `venn.js(...)`.

3 Notation

- $p \in \{2, 3\}$ is the dimensionality of the Venn and Euler representation
- sets
  - m sets $S_1, S_2, \ldots, S_m$
  - intersections $S_{ij\cdots k} = S_i \cap S_j \cap \cdots \cap S_k$
  - for any set S, $s = size(S)$ denotes its cardinality if countable, and otherwise its measure (in some sense)
- venn diagram balls
  - a ball B is defined by its centre $\mathbf{c}$ and its radius ρ (e.g. balls are circles when $p = 2$ and spheres when $p = 3$)
  - m balls $B_1, \ldots, B_m$
  - ball intersections are denoted $B_{ij\cdots k} = B_i \cap B_j \cap \cdots \cap B_k$
  - for any ball, or part of a ball, B, $b = size(B)$ denotes its area when $p = 2$ or its volume when $p = 3$
  - ball B_i is centred at $\mathbf{c}_i \in \mathbb{R}^p$ having radius ρ_i and origin $\sum_{i=1}^{m} \mathbf{c}_i = \mathbf{0}$
  - $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_m]^\mathsf{T}$ is the $m \times p$ matrix of ball centres where $p \in \{2, 3\}$ is the dimension of the display; the point configuration of the centres

- disjoint operator $disjoint(\cdots)$: for any collection $P = \{P_1, P_2, \ldots, P_m\}$ for some $N \geq m$

$$disjoint(P) = P^\star = \{P_1^\star, P_2^\star, \ldots, P_N^\star\}$$

where $\forall\ i$

1. $P_i^\star \subset P_i$
2. $P_i^\star \cap P_j^\star = \varnothing$ for $i \neq j$
3. $P_1 \cup P_2 \cup \cdots \cup P_k = P_1^\star \cup P_2^\star \cup \cdots \cup P_N^\star$

- point configurations and distance
  - $\mathbf{G} = [g_{ij}] = \mathbf{C}\mathbf{C}^\mathsf{T}$ is the Gram matrix
  - $\mathbf{g} = diag(\mathbf{G}) = (g_{11}, g_{22}, \ldots, g_{mm})^\mathsf{T}$
  - $\mathbf{D} = [d_{ij}^2]$ is the matrix of squared distances $d_{ij} = ||\mathbf{c}_i - \mathbf{c}_j||$ for some distance measure d_{ij} or norm $||\cdot||$

    Given a matrix of squared Euclidean distances $\mathbf{D}$, a point configuration $\mathbf{C}$ can be determined from the relationship [18]

$$\mathbf{G} = \frac{1}{2}\left(\mathbf{1}_m\mathbf{g}^\mathsf{T} - \mathbf{D} + \mathbf{g}\mathbf{1}_m^\mathsf{T}\right) = -\frac{1}{2}(\mathbf{I} - \mathbf{H})\mathbf{D}(\mathbf{I} - \mathbf{H})$$

    Where H $= \frac{1}{n}\mathbf{1}_m\mathbf{1}^\mathsf{T}{}_m$, $\mathbf{1}_m = [1, 1, \ldots, 1]^\mathsf{T}$ with length m and $\mathbf{I}$ is the $m \times m$ indentity matrix. Letting $\mathbf{G} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\mathsf{T}$ be the eigen-decomposition of the Gram matrix, we take $\mathbf{C} = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}$ as the initial point configuration

4 Choosing distance

4.1 Jaccard index

The Jaccard index, also known as intersection over union, is used for comparing the distance over sample sets and can be defined as follows [13]:

- let $s_{ij} = size(S_i \cap S_j)$ and $s_i = size(S_i)\ \forall i$

$$d_{ij} = 1 - \frac{size(S_i \cap S_j)}{size(S_i \cup S_j)} = 1 - \frac{s_{ij}}{s_i + s_j - s_{ij}}$$

4.2 Geometry distance

Consider the case of two balls B_i and B_j of radius ρ_i and ρ_j with $i \neq j$:

1. let $s = size(S_1 \cup S_2 \cup \cdots \cup S_m)$
2. let $b_i = size(B_i) = \frac{s_i}{s}$, $b_{ij} = size(B_i \cap B_j) = \frac{s_{ij}}{s}$, and ρ_i be the radius of a ball at this size
   - for $p = 2$, $\rho_i = \sqrt{\frac{b_i}{\pi}}$
   - for $p = 3$, $\rho_i = (\frac{3b_i}{4\pi})^{\frac{1}{3}}$
3. If $B_i \cap B_j = \varnothing$, then $d_{ij} \geq \rho_i + \rho_j$ and we choose to set $d_{ij} = \rho_i + \rho_j$
4. If $B_i \subset B_j$, then $d_{ij} \leq \rho_j - \rho_i$ and we choose to set $d_{ij} = \rho_j - \rho_i$

5. If $B_i \cap B_j \neq \varnothing$, $B_i \not\subset B_j$, $B_j \not\subset B_i$, use $B_i \cap B_j$ to determine the d_{ij}

   (a) for $p = 2$, distances are determined as in Figure 4,

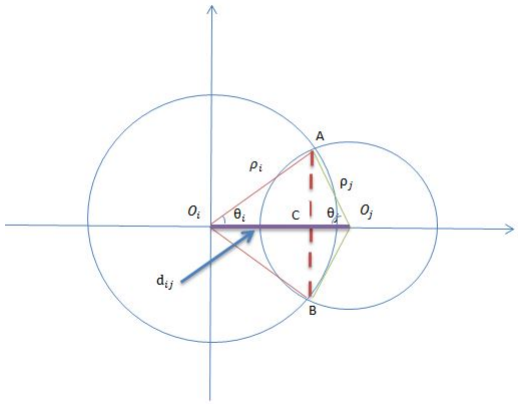


Figure 4: Dimension p = 2

O_i and O_j are the centres of the two circles and d_{ij} is the distance between the two centres. A and B are the points of intersection. AB $\perp$ O_iO_j at point C. θ_i and θ_j are two angles of the triangle AO_iO_j. Thus, d_{ij} can be found by:

$$d_{ij} = |O_iA|\cos(\theta_i) + |O_jA|\cos(\theta_j)$$

The remaining task is to find θ_i and θ_j. Firstly, $|AC| = |O_iA|\sin(\theta_i) = |O_jA| \times \sin(\theta_j)$. Secondly, area b_{ij} can be separated by line AB into two parts $Area(\widehat{AB}_{left})$ and $Area(\widehat{AB}_{right})$; $Area(\widehat{AB}_{left})$ equals to area of arc $O_j\widehat{AB}$ minus triangle O_jAB and $Area(\widehat{AB}_{right})$ equals to area of arc $O_i\widehat{AB}$ minus triangle O_iAB, where $|O_iA| = |O_iB| = \rho_i$, $|O_jA| = |O_jB| = \rho_j$. Hence, θ_i and θ_j can be found by the following equations:

$$\begin{aligned} 0 &= \theta_i\rho_i^2 - \rho_i^2\sin(\theta_i)\cos(\theta_i) + \theta_j\rho_j^2 - \rho_j^2\sin(\theta_j)\cos(\theta_j) - b_{ij} \\ 0 &= \rho_i\sin(\theta_i) - \rho_j\sin(\theta_j) \end{aligned}$$

   (b) $p = 3$

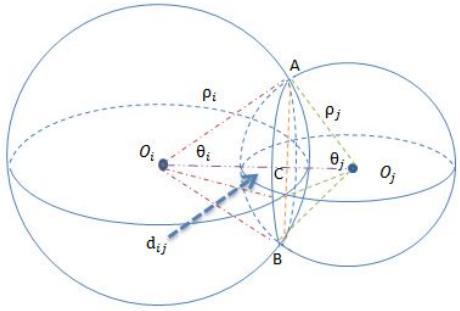


Figure 5: Dimension p = 3

It is very similar with $p = 2$. In Figure 5, O_i and O_j are the centres of these two spheres. A and B are the points of intersection and line AB is the diameter of the intersect plane, so AB $\perp O_iO_j$ at point C. θ_i and θ_j are two angles of the triangle AO_iO_j. Thus, d_{ij} can be found by:

$$d_{ij} = |O_iA|\cos(\theta_i) + |O_jA|\cos(\theta_j)$$

The remaining task is to find θ_i and θ_j. Firstly, $|AC| = |O_iA|\sin(\theta_i) = |O_jA|\sin(\theta_j)$. Secondly, volume b_{ij} can be separated by the plane, with centre C and radius $|AC|$ ($|BC|$), into two parts $SphereCap_{left}$ and $SphereCap_{right}$:

$$\begin{aligned} SphereCap_{left} &= \tfrac{\pi(|O_jA|-|O_jC|)}{6}(3\,|AC|^2 + (|O_jA| - |O_jC|)^2) \\ SphereCap_{right} &= \tfrac{\pi(|O_iA|-|O_iC|)}{6}(3\,|AC|^2 + (|O_iA| - |O_iC|)^2) \end{aligned}$$

where $|O_iA| = |O_iB| = \rho_i$, $|O_jA| = |O_jB| = \rho_j$, $|AC| = |BC| = \rho_i\sin(\theta_i)$, $|O_iC| = \rho_i\cos(\theta_i)$ and $|O_jC| = \rho_j\cos(\theta_j)$; $SphereCap_{left}$ and $SphereCap_{right}$ can be expressed as:

$$\begin{aligned} SphereCap_{left} &= \tfrac{\pi(\rho_j-\rho_j\cos(\theta_j))}{6}(3\rho_j\sin(\theta_j)^2 + (\rho_j - \rho_j\cos(\theta_j))^2) \\ SphereCap_{right} &= \tfrac{\pi(\rho_i-\rho_i\cos(\theta_i))}{6}(3\rho_i\sin(\theta_i)^2 + (\rho_i - \rho_i\cos(\theta_i))^2) \end{aligned}$$

Then, we can add them up to get b_{ij}; after simplifying, θ_i and θ_j can be found by the following equations:

$$\begin{aligned} 0 &= \tfrac{\pi}{3}\rho_i^3(1 - cos(\theta_i))^2(2 + cos(\theta_i)) \\ &\quad + \tfrac{\pi}{3}\rho_j^3(1 - cos(\theta_j))^2(2 + cos(\theta_j)) \;-\; b_{ij} \\ 0 &= \rho_i sin(\theta_i) - \rho_j sin(\theta_j) \end{aligned}$$

Use Newton-Raphson to solve for $\widehat{\theta}_i$, $\widehat{\theta}_j$, and hence d_{ij}. In conclusion, d_{ij} is a function of ρ_i, ρ_j, s_{ij} and can be showed as $d_{ij} = d(\rho_i, \rho_j, s_{ij})$:

4.3 Something

1. Let $\mathcal{P}(S)$ denote the power set excluding the null set

$$\mathcal{P}(S) = \{S_1, \ldots, S_m, S_{12}, \ldots, S_{12\cdots m}\}$$

and $\mathcal{P}(S)^\star = disjoint(\mathcal{P}(S))$. Denote by $\mathbf{s}^\star$ the vector containing the sizes of N sets of $\mathcal{P}(S)^\star$, where $\mathrm{N} \leq 2^m - 1$.
Similarly, define $\mathcal{P}(B)$, $\mathcal{P}(B)^\star$, and $\mathbf{b}^\star$ for the corresponding balls.

2. let $\mathcal{H}(S) = \mathcal{P}(S) \setminus S = \{\mathcal{S}_1, \ldots \mathcal{S}_\mathcal{N}\}$ be the higher order intersection set (the order is larger than 1) with size $\mathcal{N}$, where $\mathcal{N} = N - m$ and:
   - $\mathcal{H}(S) \cup S = \mathcal{P}(S)$
   - $\mathcal{H}(S) \cap S = \varnothing$

5 Computing areas

Following Wilkinson [24], imagine there are m 100×100 bit-squares, one for each ball. In any square, a bit is 1 if the ball for that square covers it, and is zero if it does not. Location of the Venn and Euler diagram is the pixel-wise logical disjunction of all m squares, pixels in each disjoint region of the diagram are identified by a unique pattern of the m bits for that location.

6 Wilkinson's model

6.1 Defining the model

For any configuration, the vector of sizes for the disjoint balls will be $\mathbf{b}^\star$. If fit perfectly, this should be proportional to the corresponding sizes of the disjoint sets. The extent that this is not the case is captured by fitting the linear model

$$\mathbf{b}^\star = \boldsymbol{\beta}\mathbf{s}^\star + \mathbf{r}$$

to the given $\mathbf{b}^\star$ and $\mathbf{s}^\star$ with $\mathbf{r}$ as a residual vector. The least squares fitted value for $\boldsymbol{\beta}$ is $\widehat{\boldsymbol{\beta}} = (\mathbf{s}^{\star\mathsf{T}}\mathbf{s}^\star)^{-1}\mathbf{s}^{\star\mathsf{T}}\mathbf{b}^\star$ and the estimated residual sum of squares

$$\begin{aligned} RSS &= \widehat{\mathbf{r}}^\mathsf{T}\widehat{\mathbf{r}} = (\mathbf{b}^\star - \widehat{\boldsymbol{\beta}}\mathbf{s}^\star)^\mathsf{T}(\mathbf{b}^\star - \widehat{\boldsymbol{\beta}}\mathbf{s}^\star) \\ TSS &= \mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star \end{aligned}$$

We can use $stress(\mathbf{b}^\star)$ as a measure of the quality of the fit, where:

$$stress(\mathbf{b}^\star) = \frac{RSS}{TSS} = \frac{(\mathbf{b}^\star - \widehat{\mathbf{b}}^\star)^\mathsf{T}(\mathbf{b}^\star - \widehat{\mathbf{b}}^\star)}{\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star}$$

6.2 Minimizing stress

The remaining task is to find a $\mathbf{b}^\star$ which corresponds to the minimum stress. Here, $\mathbf{b}^\star$ is a function of $\mathbf{C}$ and we can take derivative of $\mathbf{c}_i$ to get:

$$\frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{c}_i} = \frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{b}^\star}\frac{\partial \mathbf{b}^\star}{\partial \mathbf{c}_i}$$

First, let us start with $\frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{b}^\star}$:

$$\begin{aligned} \frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{b}^\star} &= \frac{2(\mathbf{b}^\star - \widehat{\mathbf{b}}^\star)\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star - 2\mathbf{b}^\star[(\mathbf{b}^\star - \widehat{\mathbf{b}}^\star)^\mathsf{T}(\mathbf{b}^\star - \widehat{\mathbf{b}}^\star)]}{(\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star)^2} \\ &= \frac{2\widehat{\mathbf{r}}\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star - 2\mathbf{b}^\star\widehat{\mathbf{r}}^\mathsf{T}\,\widehat{\mathbf{r}}}{(\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star)^2} \\ &= 2\frac{\widehat{\mathbf{r}}}{\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star} - 2\frac{\mathbf{b}^\star}{(\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star)^2}\widehat{\mathbf{r}}^\mathsf{T}\,\widehat{\mathbf{r}}. \end{aligned}$$

Now the second term above $\widehat{\mathbf{r}}^{\mathsf{T}}\widehat{\mathbf{r}}$ is of higher order than is $\widehat{\mathbf{r}}$ and can be written as $O(\widehat{\mathbf{r}})$. Hence,

$$\frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{b}^\star} \approx \frac{2}{\mathbf{b}^{\star\mathsf{T}}\mathbf{b}^\star}\widehat{\mathbf{r}} + O(\widehat{\mathbf{r}})$$

Second, find $\frac{\partial \mathbf{b}^\star}{\partial \mathbf{c}_i}$:
$\mathbf{c}_i$ and $\mathbf{c}_j$ are the centres of B_i and B_j. Thus, **Jaccard distance** can be expressed as:

$$||\mathbf{c}_i - \mathbf{c}_j|| = 1 - \frac{\mathbf{b}^{\star\mathsf{T}}\mathbf{I}_{ij}}{\mathbf{b}^{\star\mathsf{T}}\mathbf{1}_N}$$

where, $\mathbf{I}_{ij}$ is a length N vector, $i, j \in \{u_1, u_2, \ldots, u_\ell\}$ and the kth element $I_{ij}(k)$ is an indicator function

$$I_{ij}(k) = \begin{cases} 1 & \text{if } \; s_k^\star = s^\star_{u_1 u_2 \cdots u_\ell} \text{ and } \; i, j \in \{u_1, u_2, \ldots, u_\ell\} \\ \\ 0 & \text{otherwise} \end{cases}$$

that is one whenever the kth set $s_k^\star$ is the intersection set $s^\star_{u_1 u_2 \cdots u_\ell}$ and I and j identify any two sets that define the intersection. Also, $\mathbf{1}_N = [1, 1, \ldots, 1]^{\mathsf{T}}$ denotes the N-dimensional one vector.

Differentiating both sides with respect to $\mathbf{c}_i$ yields

$$\frac{\mathbf{c}_i - \mathbf{c}_j}{||\mathbf{c}_i - \mathbf{c}_j||} = -\frac{\partial}{\partial \mathbf{c}_i}\left(\frac{\mathbf{b}^{\star\mathsf{T}}\mathbf{I_{ij}}}{\mathbf{b}^{\star\mathsf{T}}\mathbf{1}_N}\right)$$

After simplifying:

$$\frac{\partial \mathbf{b}^{\star\mathsf{T}}}{\partial \mathbf{c}_i}\left((\mathbf{b}^{\star\mathsf{T}}\mathbf{1}_N)\mathbf{I}_{ij} - (\mathbf{b}^{\star\mathsf{T}}\mathbf{I}_{ij})\mathbf{1}_N\right) = -(\mathbf{b}^{\star\mathsf{T}}\mathbf{1})^2\frac{(\mathbf{c}_i - \mathbf{c}_j)}{||\mathbf{c}_i - \mathbf{c}_j||}$$

Somehow Wilkinson gets

$$\frac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{c}_i} \propto \widehat{\mathbf{r}}(\mathbf{c}_i - \mathbf{c}_j)$$

and so gets a descent step on each iteration for B_i to be (approximately) proportional to :

$$\begin{aligned} \tfrac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{c}_i} &\approx \textstyle\sum_{k=1}^N \sum_{j\neq i}^m (\mathbf{c}_i - \mathbf{c}_j)\widehat{r}_k I_{ij}(k) = \sum_{j\neq i}^m (\mathbf{c}_i - \mathbf{c}_j)\widehat{\mathbf{r}}^{\mathsf{T}}\mathbf{I}_{ij} \\ \mathbf{c}_{i+1} &= \mathbf{c}_i - \alpha\tfrac{\partial stress(\mathbf{b}^\star)}{\partial \mathbf{c}_i} \end{aligned}$$

where α is 0.01; then, with this local approximate gradient, he computes stress four times (up, down, left, right) for each ball centre by taking small steps 0.01. The gradient direction goes with the lowest stress values for $\mathbf{c_i}$.

7 An alternative loss function

For any configuration of centre locations $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_m]^{\mathsf{T}}$ we define a loss function

$$L(\mathbf{C}) = \sum_{i=1}^m \ell(\mathbf{c}_i)$$

where for each i

$$\ell(\mathbf{c}_i) = \sum_{j=1}^m l(\mathbf{c}_i, \mathbf{c}_j).$$

We could then update the configuration from its initial position by minimizing a suitably defined loss. Ben Frederickson [11] suggested that the following loss function:

$$l(\mathbf{c}_i, \mathbf{c}_j) = \begin{cases} 0 & \text{when } S_i \cap S_j = \varnothing \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \geq d_{ij}^2 \\ 0 & \text{when } S_i \subset S_j \text{ or } S_j \subset S_i \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \leq d_{ij}^2 \\ ((\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) - d_{ij}^2)^2 & \text{otherwise} \end{cases}$$

where the distances d_{ij} are fixed at their initial values, however determined from the sets S_i and S_j (e.g. Jacaard distance). This loss places a great deal of importance on the pairwise intersection between sets S_i and S_j and between balls B_i and B_j. For example, when $S_i \cap S_j = \varnothing$ then, arguably, the balls should not intersect so placing them farther apart than the distance d_{ij} incurs no loss on the pairwise intersections. Similarly, if one of S_i or S_j is a subset of the other, then ideally the corresponding ball, B_i or B_j, should be entirely inside the other. Once a distance d_{ij} has been determined by the intersection of the two sets (one is a subset of the other) the relative area of the intersection is preserved whatever the position of the centres provided they are no farther away from each other than d_{ij} – farther away and there is no guarantee that both the area and the subset relation are preserved.

Here we choose to minimize this loss using the geometric distances for $p = 2$ and $p = 3$ as defined earlier.

Our objective is to choose a configuration which minimizes this loss. To that end, we differentiate the loss with respect to $\mathbf{c}_i$ and solve. The derivative of $l(\mathbf{c}_i, \mathbf{c}_j)$ function with respect to $\mathbf{c}_i$ is

$$\frac{\partial l(\mathbf{c}_i, \mathbf{c}_j)}{\partial \mathbf{c}_i} = \begin{cases} \mathbf{0} & \text{when } S_i \cap S_j = \varnothing \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \geq d_{ij}^2 \\ \mathbf{0} & \text{when } S_i \subset S_j \text{ or } S_j \subset S_i \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \leq d_{ij}^2 \\ 4((\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) - d_{ij}^2)(\mathbf{c}_i - \mathbf{c}_j) & \text{otherwise} \end{cases}$$

Thus,

$$\frac{\partial \ell(\mathbf{c}_i)}{\partial \mathbf{c}_i} = \sum_j \frac{\partial l(\mathbf{c}_i, \mathbf{c}_j)}{\partial \mathbf{c}_i}$$

This can be used in a nonlinear conjugate gradient method [19] to find a minimum $L(\mathbf{C})$ as follows.

1. *Initialization*:

   the initial configuration

   $$\mathbf{C}^{(0)} \leftarrow [\mathbf{c}_1^{(0)}, \ldots, \mathbf{c}_m^{(0)}]^\mathsf{T}$$

   from the eigen decomposition of the gram matrix, the initial loss

   $$L(\mathbf{C}^{(0)}) \leftarrow \sum_{i=1}^{m} \ell(\mathbf{c}_i^0)$$

   and the iteration count

   $$n \leftarrow 0$$

2. *Outer loop over n*:

   (a) *Inner loop*: for $i = 1, \ldots, m$

i. Determine the conjugate direction $\mathbf{c}_i^{(n)}$:

$$\mathbf{c}_i^{(n)} \leftarrow \begin{cases} -\frac{\partial \ell(\mathbf{c}_i^{(n)})}{\partial \mathbf{c}_i^{(n)}} + \omega^{(n)} \mathbf{c}_i^{(n-1)} & n \geq 1 \\ -\frac{\partial \ell(\mathbf{c}_i^{(0)})}{\partial \mathbf{c}_i^{(0)}} & n = 0 \end{cases}$$

where

$$\omega_{PR}^{(n)} \leftarrow \frac{\frac{\partial \ell(\mathbf{c}_i^{(n)})}{\partial \mathbf{c}_i^{(n)}}^{\mathsf{T}} \left(\frac{\partial \ell(\mathbf{c}_i^{(n)})}{\partial \mathbf{c}_i^{(n)}} - \frac{\partial \ell(\mathbf{c}_i^{(n-1)})}{\partial \mathbf{c}_i^{(n-1)}} \right)}{\frac{\partial \ell(\mathbf{c}_i^{(n-1)})}{\partial \mathbf{c}_i^{(n-1)}}^{\mathsf{T}} \frac{\partial \ell(\mathbf{c}_i^{(n-1)})}{\partial \mathbf{c}_i^{(n-1)}}}$$

is the Polak-Ribiere choice [17] and

$$\omega^{(n)} \leftarrow \max\left(0, \omega_{PR}^{(n)}\right)$$

where $\omega^{(n)}$ is a popular choice [19].

ii. Perform a line search for

$$\alpha^{(n)} \leftarrow \arg\min_{\alpha} \quad \ell(\mathbf{c}_i^{(n)} + \alpha \mathbf{c}^{(n)})$$

via Newton's method.

iii. Update the position $\mathbf{c}_i$:

$$\mathbf{c}_i^{(n+1)} \leftarrow \mathbf{c}_i^{(n)} + \alpha^{(n)} \mathbf{c}^{(n)}$$

iv. end *inner loop*

(b) Update outer loop:

$$n \leftarrow n + 1$$

$$L(\mathbf{C}^{(n)}) \leftarrow \sum_{i=1}^{m} \ell(\mathbf{c}_i^{(n)})$$

(c) *Outer loop* ends when $\left| L(\mathbf{C}^{(n)}) - L(\mathbf{C}^{(n-1)}) \right| < \epsilon$.

3. Return point configuration $\mathbf{C} \leftarrow \mathbf{C}^{(n)}$

8 Our model

8.1 Case one

Let us start with data set $\mathbf{S} = \{S_1, S_2, S_3\}$ and the power set $\mathcal{P}(S) = \{S_1, S_2, S_3, S_1 \cap S_2, S_1 \cap S_3, S_2 \cap S_3, S_1 \cap S_2 \cap S_3\}$, exluding $\varnothing$.

8.1.1 Distance matrix and Initial location

Based on the point configuration (central gram matrix) and geometry distance $[d_{ij}]$, we can find the initial location $\mathbf{C}^{(0)}$.

8.1.2 Model redefinition

When $S_i \cap S_j = \varnothing$, it does not matter how long the distance is between these two centres as long as it is larger than $\rho_i + \rho_j$; when $S_i \subset S_j$ or $S_j \subset S_i$, it does not matter how short the distance is as long as it is shorter than $|\rho_i - \rho_j|$; beside these two cases, we can shrink or expand our layout a little bit to weight the order larger than two intersections, as shown in Figure 6.

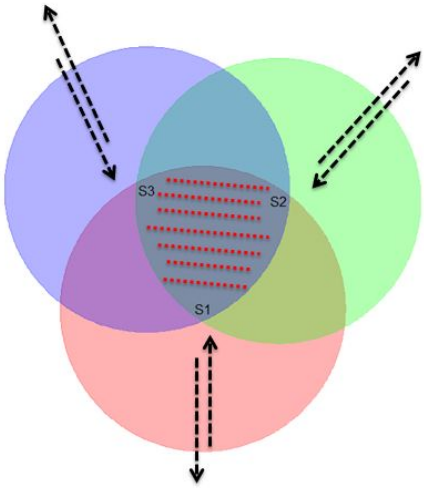


Figure 6: shrink or expand

Hence, our model can be defined as:

$$L(\mathbf{C}, \lambda) = \sum_{i=1}^{m} \ell(\mathbf{c_i}, \lambda)$$

For each i

$$\ell(\mathbf{c}_i) = \sum_{j=1}^{m} l(\mathbf{c}_i, \mathbf{c}_j, \lambda).$$

$$l(\mathbf{c}_i, \mathbf{c}_j, \lambda) = \begin{cases} 0 \quad \text{when} \quad S_i \cap S_j = \varnothing \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \geq d_{ij}^2 \\ 0 \quad \text{when} \quad S_i \subset S_j \quad \text{or} \quad S_j \subset S_i \quad \text{and} \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \leq d_{ij}^2 \\ (\lambda(\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) - d_{ij}^2)^2 \quad \text{otherwise} \end{cases}$$

The derivative of $l(\mathbf{c}_i, \mathbf{c}_j, \lambda)$ function on $\mathbf{c}_i$:

$$\frac{\partial l(\mathbf{c}_i, \mathbf{c}_j, \lambda)}{\partial \mathbf{c}_i} = \begin{cases} \mathbf{0} \quad \text{when} \quad S_i \cap S_j = \varnothing \text{ and } \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \geq d_{ij}^2 \\ \mathbf{0} \quad \text{when} \quad S_i \subset S_j \quad \text{or} \quad S_j \subset S_i \quad \text{and} \quad (\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) \leq d_{ij}^2 \\ 4\lambda(\lambda(\mathbf{c}_i - \mathbf{c}_j)^\mathsf{T}(\mathbf{c}_i - \mathbf{c}_j) - d_{ij}^2)(\mathbf{c}_i - \mathbf{c}_j) \text{ otherwise} \end{cases}$$

Thus,

$$\frac{\partial \ell(\mathbf{c}_i, \lambda)}{\partial \mathbf{c}_i} = \sum_j \frac{\partial l(\mathbf{c}_i, \mathbf{c}_j, \lambda)}{\partial \mathbf{c}_i}$$

where $\lambda \in R$. Use nonlinear conjugate gradient method to find the minimum $L(\mathbf{C}, \lambda)$. In this way, we can shrink or extend our layout and get the centres $\mathbf{C}$ ($\mathbf{C}$ is determined by λ).

To determine an appropriate value of λ, following Wilkinson et al, we introduce a $stress(\lambda)$ function based on the quality of the fit of the areas. For any configuration, given λ, the vector of sizes for the disjoint balls will be $\mathbf{b}^\star_\lambda$. If fit perfectly, this should be proportional to the corresponding sizes of the disjoint sets. The extent that this is not the case is captured by fitting the linear model

$$\mathbf{b}^\star_\lambda = \boldsymbol{\beta}\mathbf{s}^\star + \mathbf{r}$$

what's more, we can give different weights to each disjoint balls so that the quality of fit on higher intersections can be captured. Hence, the linear model can be defined as:

$$\mathbf{b}^\star_\lambda \mathbf{W}^{\frac{1}{2}} = \boldsymbol{\beta}\mathbf{s}^\star \mathbf{W}^{\frac{1}{2}} + \mathbf{r}\mathbf{W}^{\frac{1}{2}}$$

where $\mathbf{W} = diag(w_1, w_2, ..., w_N)$ be a $N \times N$ (in this example $N = 7$) diagonal matrix. To the given $\mathbf{b}^\star_\lambda$, $\mathbf{W}$ and $\mathbf{s}^\star$ with $\mathbf{r}$ as a residual vector. The weighted least squares fitted value for $\boldsymbol{\beta}$ is $\widehat{\boldsymbol{\beta}} = (\mathbf{s}^{\star\mathsf{T}}\mathbf{W}\mathbf{s}^\star)^{-1}\mathbf{s}^{\star\mathsf{T}}\mathbf{W}\mathbf{b}^\star_\lambda$ and the estimated residual sum of squares

$$\begin{aligned} RSS(\beta, \lambda) &= \widehat{\mathbf{r}}^\mathsf{T}\mathbf{W}\widehat{\mathbf{r}} = (\mathbf{b}^\star_\lambda - \widehat{\boldsymbol{\beta}}\mathbf{s}^\star)^\mathsf{T}\mathbf{W}(\mathbf{b}^\star_\lambda - \widehat{\boldsymbol{\beta}}\mathbf{s}^\star) \\ TSS &= \mathbf{b}^{\star\mathsf{T}}_\lambda \mathbf{b}^\star_\lambda \end{aligned}$$

We can use $stress(\lambda)$ as a measure of the quality of the fit, where:

$$stress(\lambda) = \frac{RSS(\beta, \lambda)}{TSS}$$

and

$$\text{stress} = \arg\min_\lambda stress(\lambda)$$

Here suggests two algorithms to find cnetre $\mathbf{C}$ and corresponding stress

1. *Initialization*:
   the initial point configuration
   $$\mathbf{C}^{(0)} \leftarrow [\mathbf{c}^{(0)}_1, \ldots, \mathbf{c}^{(0)}_m]^\mathsf{T}$$
   the initial λ
   $$\lambda^{(1)} \leftarrow 1$$
   and the initial count
   $$n \leftarrow 1$$
2. Fixing $\lambda^{(n)}$ and computing $stress(\lambda^{(n)})$
   - Minimizing $L(\mathbf{C}; \lambda^{(n)})$ and get $\mathbf{C}^{(n)}$
     $$\mathbf{C}^{(n)} \leftarrow \arg\min_{\mathbf{C}} \; L(\mathbf{C}; \lambda^{(n)})$$
   - Compute each area $\mathbf{b}^\star_{\lambda^{(n)}}$ and find $\widehat{\beta}^{(n)}$
     $$\widehat{\beta}^{(n)} \leftarrow \arg\min_\beta RSS(\beta; \lambda^{(n)})$$
     $\widehat{\beta}^{(n)}$ can be solved as $(\mathbf{s}^{\star\mathsf{T}}\mathbf{W}\mathbf{s}^\star)^{-1}\mathbf{s}^{\star\mathsf{T}}\mathbf{W}\mathbf{b}^\star_{\lambda^{(n)}}$

- Finding $stress(\lambda^{(n)})$

$$stress(\lambda^{(n)}) \leftarrow \frac{RSS(\widehat{\beta}^{(n)}, \lambda^{(n)})}{\mathbf{b}^{\star\mathsf{T}}_{\lambda^{(n)}} \mathbf{W} \mathbf{b}^{\star}_{\lambda^{(n)}}}$$

3. Two algorithms for finding $\mathbf{C}$ and $stress$:
   Set

$$\lambda_{+}^{(n)} \leftarrow \lambda^{(n)} + \Delta$$

$$\lambda_{-}^{(n)} \leftarrow \lambda^{(n)} - \Delta$$

   where $\Delta \in \Re^{+}$, a small step size. Fixing $\lambda_{+}^{(n)}$ and $\lambda_{-}^{(n)}$ to compute $stress(\lambda_{+}^{(n)})$ and $stress(\lambda_{-}^{(n)})$.

   - *Nelder Mead Algorithm [16]*:
     (a) *Loop over n*:
       i. *Previous preparation*:
       the initial $\boldsymbol{\lambda}^{(n)}$ vector

$$\boldsymbol{\lambda}^{(n)} \leftarrow [\lambda^{(n)}, \lambda_{+}^{(n)}, \lambda_{-}^{(n)}]$$

       the corresponding $\mathbf{stress}^{(n)}$

$$\mathbf{stress}^{(n)} \leftarrow [stress(\lambda^{(n)}), stress(\lambda_{+}^{(n)}), stress(\lambda_{-}^{(n)})]$$

       and assuming:

$$stress(\lambda_1) \leq stress(\lambda_2) \leq stress(\lambda_3)$$

       where, $\lambda_i \in \boldsymbol{\lambda}^{(n)}$, $stress(\lambda_i) \in \mathbf{stress}^{(n)}$ and $i = \{1, 2, 3\}$. λ_0 can be computed as:

$$\lambda_0 \leftarrow \frac{\lambda_1 + \lambda_2}{2}$$

       λ_r is:

$$\lambda_r \leftarrow 2\lambda_0 - \lambda_3$$

       ii. Update *Loop*
       i). the $\boldsymbol{\lambda}$:
       – **if**$(stress(\lambda_1) \leq stress(\lambda_r) < stress(\lambda_2))$ **then** do *reflection*

$$\boldsymbol{\lambda}^{(n+1)} \leftarrow [\lambda_1, \lambda_2, \lambda_r]$$

       – **else if**$(stress(\lambda_r) < stress(\lambda_1))$ **then** do *expansion*

$$\lambda_e \leftarrow 2\lambda_r - \lambda_0$$

       and get corresponding $stress(\lambda_e)$
       ∗ **if** $(stress(\lambda_e) < stress(\lambda_r))$ **then**

$$\boldsymbol{\lambda}^{(n+1)} \leftarrow [\lambda_1, \lambda_2, \lambda_e]$$

       ∗ **else**

$$\boldsymbol{\lambda}^{(n+1)} \leftarrow [\lambda_1, \lambda_2, \lambda_r]$$

       – **else if**$(stress(\lambda_r) \geq stress(\lambda_2))$ **then** do *contraction*:

$$\lambda_c \leftarrow \frac{\lambda_0 + \lambda_3}{2}$$

* **if**$(stress(\lambda_c) \leq stress(\lambda_3))$ **then**

$$\boldsymbol{\lambda}^{(n+1)} \leftarrow [\lambda_1, \lambda_2, \lambda_c]$$

ii). the count:

$$n \leftarrow n + 1$$

iii. *Shrink* $\boldsymbol{\lambda}$:
With $\boldsymbol{\lambda}^{(n)}$, we can get corresponding $\mathbf{stress}^{(n)}$, and assuming:

$$stress(\lambda_1) \leq stress(\lambda_2) \leq stress(\lambda_3)$$

where, $\lambda_i \in \boldsymbol{\lambda}^{(n)}$, $stress(\lambda_i) \in \mathbf{stress}^{(n)}$ and $i = \{1, 2, 3\}$.
Shrink λ_j, where $j = \{2, 3\}$

$$\lambda_j \leftarrow \frac{\lambda_j + \lambda_1}{2}$$

iv. *Loop* ends when

$$\left|\max(\boldsymbol{\lambda}^{(n)}) - \min(\boldsymbol{\lambda}^{(n)})\right| \leq \epsilon$$

or

$$\left|\max(\mathbf{stress}^{(n)}) - \min(\mathbf{stress}^{(n)})\right| \leq \epsilon$$

v. stress = $\min(\mathbf{stress}^{(n)})$

- *Line Search [4]*:
  - Direction search
    * **if**$(\min(stress(\lambda_+), stress(\lambda_-), stress(\lambda^{(n)})) = stress(\lambda^{(n)}))$
      Return $\mathbf{C} \leftarrow \mathbf{C}^{(n)}$
    * **else**
    (a) **if** $(\min(stress(\lambda_+), stress(\lambda_-), stress(\lambda^{(n)})) = stress(\lambda_+))$ which means shrinkage can decrease *stress*. Thus,

$$\lambda^{(n)} \leftarrow \lambda_+ \text{ and } stress(\lambda^{(n)}) \leftarrow stress(\lambda_+)$$

i. update λ

$$\lambda^{(n+1)} \leftarrow \lambda^{(n)} + \Delta$$

ii. update count n

$$n \leftarrow n + 1$$

iii. Fixing $\lambda^{(n)}$ and compute $stress(\lambda^{(n)})$
iv. Repeat i to iii until $stress(\lambda^{(n-1)}) \leq stress(\lambda^{(n)})$
v. Return $\mathbf{C} \leftarrow \mathbf{C}^{(n-1)}$ and stress = $stress(\lambda^{(n-1)})$

(b) **else** which means expansion can decrease stress. Thus,

$$\lambda^{(n)} \leftarrow \lambda_-$$

And λ can be updated as follows:

$$\lambda^{(n+1)} \leftarrow \lambda^{(n)} - \Delta$$

The rest procedure is similar with (a). ii to v.

4. Return $\mathbf{C}$ and stress

8.2 Case two

In this case, some two way intersections are missing. Due to this situation, we cannot exactly determine the geometric distances $[d_{ij}]$. Thus, we need to estimate the $[d_{ij}]$ (if missing), then to fit the model and get the centre $\mathbf{C}$ corresponding to the minimum *stress*.

8.2.1 Distance matrix and Initial location

For exmaple, the data set $\mathbf{S} = \{S_1, S_2, S_3, S_4\}$ and the power set $\mathcal{P}(S) = \{S_1, S_2, S_3, S_4, S_1 \cap S_2, S_1 \cap S_2 \cap S_3 \cap S_4\}$. We have:

$$\begin{cases} s_{1234} \leq \widehat{s}_{123}, \widehat{s}_{134} \leq \widehat{s}_{13} \leq min(s_1, s_3) & (1) \\ s_{1234} \leq \widehat{s}_{124}, \widehat{s}_{134} \leq \widehat{s}_{14} \leq min(s_1, s_4) & (2) \\ s_{1234} \leq \widehat{s}_{123}, \widehat{s}_{234} \leq \widehat{s}_{23} \leq min(s_2, s_3) & (3) \\ s_{1234} \leq \widehat{s}_{124}, \widehat{s}_{234} \leq \widehat{s}_{24} \leq min(s_2, s_4) & (4) \\ s_{1234} \leq \widehat{s}_{134}, \widehat{s}_{234} \leq \widehat{s}_{34} \leq min(s_3, s_4) & (5) \end{cases}$$

For the inequation (1), we have $\widehat{s}_{123} = \mu_1 s_{1234}$, $\widehat{s}_{134} = \mu_2 s_{1234}$ and $\widehat{s}_{13} = \mu_3 \widehat{s}_{123} = \mu_4 \widehat{s}_{134}$. Thus, $\widehat{s}_{13} = \mu_3 \mu_1 s_{1234} = \mu_4 \mu_2 s_{1234} \leq min(s_1, s_3)$ and the constraints for (1):

$$\begin{cases} \mu_1 \mu_3 = \mu_2 \mu_4 \\ \mu_1 \mu_3 \leq \frac{min(s_1, s_3)}{s_{1234}} \\ \mu_2 \mu_4 \leq \frac{min(s_1, s_3)}{s_{1234}} \\ \mu_i \geq 1 \quad i = \{1, 2, 3, 4\} \end{cases}$$

Since we have five inequations, four missing order three intersections $(\widehat{s}_{123}, \widehat{s}_{124}, \widehat{s}_{134}, \widehat{s}_{234})$, five missing order two intersections $(\widehat{s}_{12}, \widehat{s}_{13}, \widehat{s}_{14}, \widehat{s}_{23}, \widehat{s}_{24})$. Thus, we need nine parameters $\{\mu_1, \mu_2, \ldots, \mu_9\}$. For simplicity, we choose to set $\mu = \mu_1 = \mu_2 = \ldots = \mu_9$. In this way, we can shorten nine parameters to just one and we have:

$$\widehat{s}_{13} = \widehat{s}_{14} = \widehat{s}_{23} = \widehat{s}_{24} = \widehat{s}_{34} = \mu^2 s_{1234}$$

where $1 \leq \mu \leq \sqrt{\frac{min(s_1, s_2, s_3, s_4)}{s_{1234}}}$. Since d_{ij} is a function of ρ_i, ρ_j, s_{ij} (if order two intersection exists) and $\rho_i, \rho_j, \widehat{s}_{ij}$ (if order two intersection missing), and we can use $[\widehat{d}_{ij}]$ to represent geometic distance matrix, where $[\widehat{d}_{ij}]$ is determined by μ.

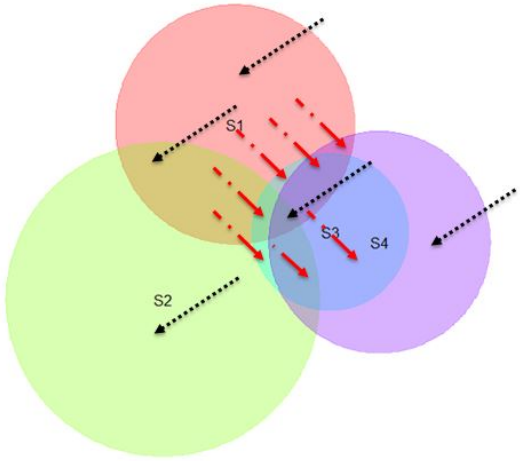


Figure 7: Missing order two intersections

In Figure 7, the red dots are the intersections we have and the black ones are the missing intersections. We want to estimate the black dots part so that we can fit the red dots part as well as possible.

8.2.2 Model redefinition

The model is the same with before but replace $[d_{ij}]$ to $[\widehat{d}_{ij}]$. Hence, we add parameter μ in our model.

$$L(\mathbf{C}, \lambda; \mu) = \sum_{i=1}^{m} \ell(\mathbf{c_i}, \lambda; \mu) = \sum_{i=1}^{m} \sum_{j=1}^{m} l(\mathbf{c}_i, \mathbf{c}_j, \lambda; \mu)$$

Minimize $L(\mathbf{C}, \lambda; \mu)$ to get centre $\mathbf{C}$ and compute each area $\mathbf{b}_{\mathbf{C}}^{\star}$. Then, weighted RSS and $stress$ can be expressed as:

$$RSS(\beta, \lambda; \mu) = (\mathbf{b}_{\lambda}^{\star} - \beta \mathbf{s}^{\star})^{\mathsf{T}} \mathbf{W} (\mathbf{b}_{\lambda}^{\star} - \beta \mathbf{s}^{\star})$$

$$stress(\lambda; \mu) = \frac{RSS(\beta, \lambda; \mu)}{\mathbf{b}_{\lambda}^{\star\mathsf{T}} \mathbf{W} \mathbf{b}_{\lambda}^{\star}}$$

Here, we have two parameters μ and λ. We will start with parameter μ, then λ, because it is meaningless to shrink or expand with a poor geomatric distance.

Due to $\mu \geq 1$, here suggests a possible algorithm for finding a good centre $\mathbf{C}$:

1. *Initialization*:
   the initial point configuration
   $$\mathbf{C}^{(0)} \leftarrow [\mathbf{c}_1^{(0)}, \ldots, \mathbf{c}_m^{(0)}]^{\mathsf{T}}$$
   the initial μ
   $$\mu^{(1)} \leftarrow 1 \text{ and } \mu^{(1)} \Rightarrow [\widehat{d}_{ij}^{(1)}]$$
   the initial λ
   $$\lambda^{(1)} \leftarrow 1$$
   and the initial count
   $$n \leftarrow 1$$

2. *Two algorithms for finding* $\mathbf{C}$:

- *Nelder Mead Algorithm [16]*:
  (a) Set
  $$\mu_{+}^{(n)} \leftarrow \mu^{(n)} + \Delta$$
  $$\mu_{++}^{(n)} \leftarrow \lambda^{(n)} + 2\Delta$$
  and $\boldsymbol{\mu}$ can be defined as:
  $$\boldsymbol{\mu}^{(n)} \leftarrow [\mu^{(n)}, \mu_{+}^{(n)}, \mu_{++}^{(n)}]$$
  Thus we can get corresponding estimated distance matrix. Fixing $\lambda^{(1)}$, **stress** is:
  $$\mathbf{stress}^{(n)} = [stress(\lambda^{(1)}; \mu^{(n)}), stress(\lambda^{(1)}; \mu_{+}^{(n)}), stress(\lambda^{(1)}; \mu_{++}^{(n)})]$$
  (b) The rest procedure is similar with before: do *Loop*, but replace $\boldsymbol{\lambda}^{(n)}$ to $\boldsymbol{\mu}^{(n)}$ and fix $\lambda^{(1)}$
  (c) Return
  $$\mu = \frac{\sum(\boldsymbol{\mu}^{(n)})}{3}$$
  and corresponding distance $[\widehat{d}_{ij}]$
- *Line Search [4]*:
  – *Loop over* n:
  (a) Fixing $\lambda^{(1)}$ and $\mu^{(n)}$
  $$\mathbf{C}^{(n)} \leftarrow \arg\min_{\mathbf{C}} L(\mathbf{C}; \lambda^{(1)}, \mu^{(n)})$$
  (b) Compute area $\mathbf{b}^{\star}_{\lambda^{(1)},\mu^{(n)}}$ and
  $$\widehat{\beta}^{(n)} \leftarrow \arg\min_{\beta} RSS(\beta; \lambda^{(1)}, \mu^{(n)})$$
  get corresponding
  $$stress(\lambda^{(1)}; \mu^{(n)}) \leftarrow \frac{RSS(\widehat{\beta}^{(n)}, \lambda^{(1)}, \mu^{(n)})}{\mathbf{b}^{\star\mathsf{T}}_{\lambda^{(1)},\mu^{(n)}} \mathbf{W} \mathbf{b}^{\star}_{\lambda^{(1)},\mu^{(n)}}}$$
  (c) Update *Loop*
  $$\mu^{(n+1)} \leftarrow \mu^{(n)} + \Delta$$
  $$n \leftarrow n + 1$$
  (d) *Loop* ends when $stress(\lambda^{(1)}; \mu^{(n-1)}) \leq stress(\lambda^{(1)}; \mu^{(n)})$
  – Return $\mu \leftarrow \mu^{(n-1)}$ and get corresponding $[\widehat{d}_{ij}] \leftarrow [\widehat{d}_{ij}^{(n-1)}]$

3. Fixing $[\widehat{d}_{ij}]$ and shrinking or expanding λ, the same as **Case 1**, to find the minimum stress, then return $\mathbf{C}$.

8.3 In general

If any two way intersections are missing, such as $S_i \cap S_j$, but there are some higher than two way intersection sets $\{S_1 \cap S_2 \cap \cdots \cap S_M, S_1 \cap S_2 \cap \cdots \cap S_L, \ldots\} \subseteq S_i \cap S_j$ and $S_1 \cap S_2 \cap \cdots \cap S_M = \max(\{S_1 \cap S_2 \cap \cdots \cap S_M, S_1 \cap S_2 \cap \cdots \cap S_L, \ldots\})$. Based on the generation we mentioned before, we can use $S_1 \cap S_2 \cap \cdots \cap S_M$ to get $S_i \cap S_j$
$$\widehat{s}_{ij} = s_{12\cdots M} \times \mu^{M-2}$$

9 Undirected connected components

In graph theory, undirected connected components [12] is subgraph that any two nodes can be connected by undirected paths.

9.1 Notation

- Undirected connected components $\mathcal{G} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V}$ is a set whose elements are called nodes and $\mathbf{E}$ is the undirected edges connecting nodes. In power set $\mathcal{P}(S)$, any order k intersections, where $k \geq 2$, can be treated as $\binom{k}{2}$ edges ; such as $S_1 \cap S_2 \cap S_3$ implies S_1, S_2 and S_3 must be connected thus $\mathbf{E} = \{S_1 \cap S_2 \cap S_3\}$ and $\mathbf{V} = \{S_1, S_2, S_3\}$
- Assuming we have η $\mathcal{G}$s, let $\mathcal{G}_i = \{\mathbf{V}_1, \mathbf{E}_1\}, \ldots, \{\mathbf{V}_\eta, \mathbf{E}_\eta\}$, where $1 \leq \eta \leq m$. In each $\mathbf{V}_i$, any two sets can be connected by edges $\mathbf{E}_i$.
  1. – $\mathbf{V}_1 \cup \mathbf{V}_2 \cup \ldots \cup \mathbf{V}_\eta = \{S_1, S_2, \ldots, S_m\}$
     – $\mathbf{E}_1 \cup \mathbf{E}_2 \cup \ldots \cup \mathbf{E}_\eta = \mathcal{H}(S)$
  2. – $\mathbf{V}_i \cap \mathbf{V}_j = \varnothing$
     – $\mathbf{E}_i \cap \mathbf{E}_j = \varnothing$
     where $i \neq j$

The fewer disks we fit, the lower rank distance matrix we have. What's more, we can avoid some unnecessary overlay if we find $\mathcal{G}$s at the beginning. Hence, we can fit our model as below:

1. Detect $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_\eta\}$
2. Use $\mathbf{E}_i$ to detect which case $\mathcal{G}_i$ belongs to (whether distance matrix can be exactly determined), then fit with corresponding model.
3. Layout $\mathcal{G}$ with reasonable distance (not too far or too close)

9.2 $\mathcal{G}$ detection

Given data set $\mathbf{S} = \{S_1, S_2, ..., S_m\}$, a size $\mathcal{N}$ higher order set $\mathcal{H}(S)$ and a size N power set $\mathcal{P}(S)$ (excluding $\varnothing$). We can use the following algorithm to detect $\mathcal{G}$. Before we start, let us introduce some functions which can help us better understand this algorithm:

- *separate* function: input is a high order intersection; output is each unit. e.g. *separate*$(S_1 \cap S_2 \cap S_3) = \{S_1, S_2, S_3\}$
- *unique* function: returns a vector but with duplicate elements removed. e.g. *unique*$(S_1, S_2, S_3, S_1) = \{S_1, S_2, S_3\}$
- *any* function: give a set of logical vectors, is at least one of the values true. e.g. $\mathbf{z} = \{A, B, C\}$, $\mathbf{Z} = \{A, D, E\}$, then *any*$(\mathbf{z} \in \mathbf{Z}) = TRUE$; $\mathbf{z} = \{A, B, C\}$, $\mathbf{Z} = \{D, E\}$, then *any*$(\mathbf{z} \in \mathbf{Z}) = FALSE$; $\mathbf{z} = \{TRUE, FALSE\}$, then *any*$(\mathbf{z} = TRUE) = TRUE$.
- *which* function: give the true indices of a logical object. e.g. $\mathbf{z} = \{TRUE, FALSE, TRUE\}$;*which*$(\mathbf{z} = TRUE) = \{1, 3\}$
- *length* function: get the length of vectors. e.g. $\mathbf{z} = \{TRUE, FALSE, TRUE\}$; *length*$(\mathbf{z}) = 3$.

The following procedure can help us find the $\mathcal{G}$s

1. **if** $\mathcal{N} = 2^m - m - 1$ **then**
   $\eta = 1$; $\mathbf{V}_1 = \mathbf{S}$ and $\mathbf{E}_1 = \mathcal{H}(S)$

2. **else if** $\mathcal{N} = 1$ **then**
   Where $\mathcal{H}(S) = \{\mathcal{S}_1\}$, assume $\mathcal{S}_1$ is the order k intersection, thus seperate($\mathcal{S}_1$) is a k size set and $\eta = m - k + 1$. $S \setminus separate(\mathcal{S}_1) = \{\gamma_1, \gamma_2, \ldots, \gamma_{m-k}\}$. Hence $\mathbf{V}_i = \gamma_i$, $\mathbf{E}_i = \varnothing$, where $1 \leq i \leq m - k$ and $\mathbf{V}_\eta = seperate(\mathcal{S}_1)$, $\mathbf{E}_\eta = \mathcal{S}_1$.

3. **else**

   (a) $\mathcal{H}(S) = \{\mathcal{S}_1, \mathcal{S}_2, \ldots \mathcal{S}_\mathcal{N}\}$

   (b) *Outer loop:* $i = 1, \ldots, \mathcal{N}$:

      i. $\mathbf{Boolean}_1 = \{bool_1, \ldots, bool_\mathcal{N}\}$ and $bool_u = FALSE$, where $1 \leq u \leq \mathcal{N}$;

      ii. *Inner loop*: for $j = i, \ldots, \mathcal{N}$:

         - $bool_j = any(separate(\mathcal{S}_i) \in separate(\mathcal{S}_j))$

      iii. $\mathbf{A} = which(\mathbf{Boolean}_1 = TRUE)$; $length(\mathbf{A}) = a$ and $\mathbf{A} = \{A_1, \ldots, A_a\}$

      iv. $\mathbf{V}_i = unique(spearate(\mathcal{S}_{A_1}), \ldots, spearate(\mathcal{S}_{A_a}))$ and $\mathbf{E}_i = \{\mathcal{S}_{A_1}, \ldots, \mathcal{S}_{A_a}\}$

      v. **if** $i > 1$ **then**
      $\mathbf{Boolean}^{(2)} = \{bool_1, \ldots, bool_{i-1}\}$ and $bool_u = FALSE$, where $1 \leq u \leq i - 1$

         A. *Inner loop*: for $j = 1, \ldots, i - 1$:

            - $bool_j = any(\mathbf{V}_j \in \mathbf{V}_i)$

         B. $\tau = which(\mathbf{Boolean}^{(2)} = TRUE)$; $length(\tau) = 0$ or 1

         C. **if**$(length(\tau) = 1)$ **then**

            - $\mathbf{V}_\tau = unique(\mathbf{V}_\tau, \mathbf{V}_i)$ and $\mathbf{E}_\tau = unique(\mathbf{E}_\tau, \mathbf{E}_i)$
            - $\mathbf{V}_i = \mathbf{E}_i = \varnothing$

   (c) Get rid of all the emptysets and reduce $\mathcal{N}$ to ν, where $1 \leq \nu \leq \mathcal{N}$. $\forall\ i$, $\mathbf{V}_i \neq \varnothing$ and $\mathbf{E}_i \neq \varnothing$, where $1 \leq i \leq \nu$

   (d) **if**$(\mathbf{V}_1 \cup \ldots \cup \mathbf{V}_\nu = \mathbf{S})$ **then**

      - $\nu = \eta$

      else

      - $\mathbf{S} \setminus (\mathbf{V}_1 \cup \ldots \cup \mathbf{V}_\nu) = \{\gamma_1, \ldots \gamma_{\eta-\nu}\}$
      - $\mathbf{V}_{\nu+j} = \gamma_j$ and $\mathbf{E}_{\nu+j} = \varnothing$, where $1 \leq j \leq \eta - \nu$

4. $\mathcal{G}_i = \{\mathbf{V}_i, \mathbf{E}_i\}$, where $1 \leq i \leq \eta$ and return $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_\eta\}$

9.3 Case detection

We need to define a new function *order*:

- *order*: give the order of an intersection. e.g. $order(S_1 \cap S_2 \cap S_3) = 3$

For each $\mathcal{G}_i = \{\mathbf{V}_i, \mathbf{E}_i\}$, where $1 \leq i \leq \eta$ and $\mathbf{E}_i = \{e_1, \ldots, e_\kappa\}$, e_j means edges (intersections) belonging to undirected connected component $\mathcal{G}_i$, where $1 \leq j \leq \kappa$:

1. Notation

   - $\mathbf{L} = which(order(\mathbf{E}_i) = 2)$ and $length(\mathbf{L}) = l$
   - $length(\mathbf{E}_i) = \kappa$

2. **if**$(l = \kappa)$ **then** $\mathcal{G}_i$ belongs to **Case 1**

3. **else**

   - $\mathbf{L} = \{L_1, \ldots, L_l\}$, where $1 \leq L_1 \leq L_l \leq \kappa$

- $\mathcal{O}_2 = \{e_{L_1}, \ldots, e_{L_l}\}$
- $\mathbf{T}$ = *which*(*order*(E_i) > 2) and *length*($\mathbf{T}$) $= t$
  - $t + l = \kappa$
- $\mathbf{T} = \{T_1, \ldots, T_t\}$
- $\mathcal{O}_r = \{e_{T_1}, \ldots, e_{T_t}\}$
  - $\mathcal{O}_2 \cup \mathcal{O}_r = \mathbf{E}_i$

(a) *Outer Loop*: $j = 1, \ldots, t$

i. Assuming e_{T_j} is the order k intersection, where *order*(e_{T_j}) $= O > 2$ and it implies $\binom{O}{2}$ edges. Set $q = \binom{O}{2}$ and $\mathbf{Q}$ is an order two intersections list which e_{T_j} connotes.

$$\mathbf{Q} = \{Q_1, \ldots, Q_q\}$$

ii. **Boolean** $= \{bool_1, \ldots, bool_q\}$ and $bool_u = FALSE$, where $1 \leq u \leq \mathcal{Q}$

iii. *Inner Loop*: $u = 1, \ldots q$

- $bool_u$ = *any*($Q_u \in \mathcal{O}_2$)

iv. **If** (*any*(**Boolean** $= FALSE$)) **then**

- $\mathcal{G}_i$ belongs to **Case 2**
- **break** *Outer Loop*

else $\mathcal{G}_i$ to be determined

(b) **If** $\mathcal{G}_i$ still to be determined **then**
$\mathcal{G}_i$ belongs to **Case 1**

9.4 $\mathcal{G}$ layout

$\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_\eta\}$ and $\mathcal{G}_i = \{\mathbf{V}_i, \mathbf{E}_i\}$. $\mathbf{V}_i$ is the size m_i data set and $\mathbf{B}_i$ is the corresponding balls with radiuses $\mathbf{R}_i$. Hence, $\mathbf{R}_i$ is a size $m_i \times 1$ vector. After **Case Detection**, fit the model one by one and get $\mathcal{Y} = [\mathbf{C}_1, \ldots, \mathbf{C}_\eta]^\mathsf{T}$. Thus $\mathbf{C}_i$ is a $m_i \times p$ matrix and $p = 2$ or 3.

$$\sum_{i=1}^{\eta} m_i = m$$

The following procedures can help us to lay out $\mathbf{C}_i$ together but with reasonable distances.

1. *Initialization*:

   - Put $\mathbf{C}_1$ in a rectangle box, which can load all these balls.
   - **if** ($\eta = 1$) return $\mathbf{C}_i$
     else go to next *Outer Loop*

2. *Outer Loop*: $i = 2, \ldots, \eta$

   (a) Randomly select one point $\mathbf{x}$ in this box. Then pick one coordinate (row) $\mathbf{c}_j$ in $\mathbf{C}_i$ (with its radius ρ_j), where $1 \leq j \leq m_i$. This coordinate $\mathbf{c}_j$ must have either the largest (x or y or z) or smallest (x or y or z).

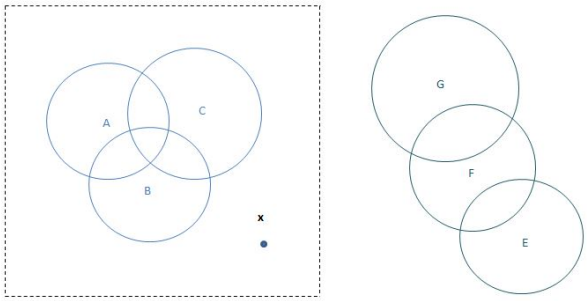


Figure 8: The choice of $\mathbf{c}_j$ and $\mathbf{x}$

In Figure 8, the imaginary line is a rectangular box and $\mathbf{x}$ is the point we randomly generate. $\mathbf{c}_j$ we choose is the centre ball E, which has the largest x (smallest y).

(b) Translate $\mathbf{C}_i$ to $\mathbf{C}_i'$:

$$\mathbf{C}_i' = \mathbf{C}_i + \mathbf{1}_{m_i}\mathbf{x} - \mathbf{1}_{m_i}\mathbf{c}_j$$

where $\mathbf{1}_{m_i} = [1, 1, \ldots, 1]^\mathsf{T}$ with size m_i

- All the distances $\{d_{1x}, d_{2x}, ..., d_{m_i x}\}$ between $\mathbf{x}$ and $\mathbf{C}_{i-1}$ are larger than $\mathbf{R}_{i-1} + \mathbf{1}^\mathsf{T}_{m_{i-1}}\rho_j$, where $\mathbf{1}_{m_{i-1}} = [1, 1, \ldots, 1]^\mathsf{T}$ with size m_{i-1}.
- At least one of distances $\{d_{1x}, d_{2x}, ..., d_{m_i x}\}$ is smaller than $\mathbf{R}_{i-1} + \mathbf{1}^\mathsf{T}_{m_{i-1}}\rho_j + \mathbf{1}^\mathsf{T}_{m_{i-1}}\delta$, where $\delta \in \Re^+$

If any of these conditions does not match, then go back to first step of *Outer Loop* and get another random point $\mathbf{x}$.

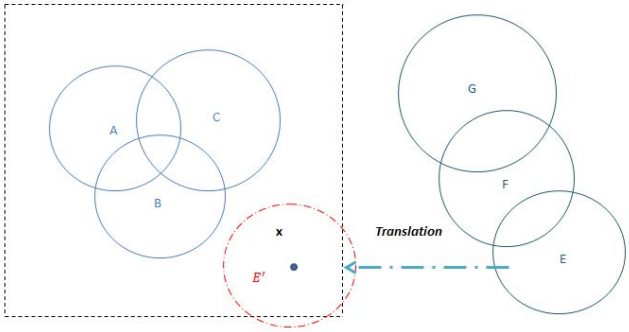


Figure 9: Translation $\mathbf{c}_j$

(c) Rotate $\mathbf{C}_i'$ until $\mathbf{C}_{i-1}$ and $\mathbf{C}_i'$ are totally separated.

i. Define a $m_{i-1} \times m_i$ matrix $\mathcal{D}$

$$\begin{aligned}\mathcal{D} &= \textstyle\sum_{k=1}^{p} \left(\mathbf{C}_{i-1}\mathbf{e}_k\mathbf{1}^\mathsf{T}_{m_i} - \mathbf{1}_{m_{i-1}}\mathbf{e}_k^\mathsf{T}\mathbf{C}_i^\mathsf{T}\right) \circ \left(\mathbf{C}_{i-1}\mathbf{e}_k\mathbf{1}^\mathsf{T}_{m_i} - \mathbf{1}_{m_{i-1}}\mathbf{e}_k^\mathsf{T}\mathbf{C}_i^\mathsf{T}\right) \\ &\quad - \left(\mathbf{R}_{i-1}\mathbf{1}^\mathsf{T}_{m_i} + \mathbf{1}_{m_{i-1}}\mathbf{R}_i^\mathsf{T}\right) \circ \left(\mathbf{R}_{i-1}\mathbf{1}^\mathsf{T}_{m_i} + \mathbf{1}_{m_{i-1}}\mathbf{R}_i^\mathsf{T}\right)\end{aligned}$$

where e_k is a p-dimension standard basis, $e_k = [0, \ldots, 1, \ldots, 0]^\mathsf{T}$ only the k th element is 1.

ii. **if** (all elements in $\mathcal{D}$ are equale or larger than 0)
Return $\mathbf{C}'_i$

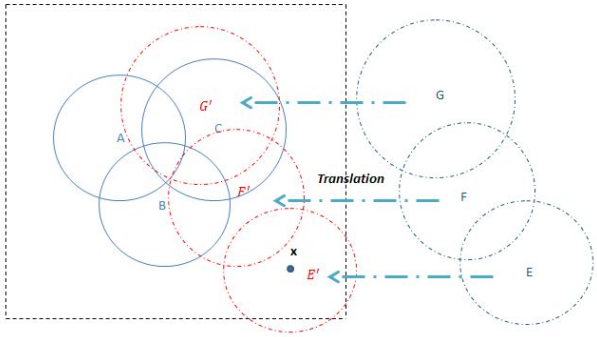


Figure 10: Translation

In Figure 10, not all elements in $\mathcal{D}$ are equale or larger than 0. Hence, we need to do rotation [23].

iii. **else**

- *Inner Loop*: $\theta = \frac{\pi}{18}, \frac{2\pi}{18}, \ldots, 2\pi$
- – $\mathbf{C}'_i \leftarrow \mathbf{C}'_i \times \mathcal{R}$
  - ∗ for p = 2

$$\mathcal{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

  - ∗ for p =3

$$\mathcal{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_1) & -\sin(\theta_1) \\ 0 & \sin(\theta_1) & \cos(\theta_1) \end{bmatrix} \begin{bmatrix} \cos(\theta_2) & 0 & \sin(\theta_2) \\ 0 & 1 & 0 \\ -\sin(\theta_1) & 0 & \cos(\theta_2) \end{bmatrix}$$

$$\begin{bmatrix} \cos(\theta_3) & -\sin(\theta_3) & 0 \\ \sin(\theta_3) & \cos(\theta_3) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

θ_1, θ_2 and θ_3 are not necessarily equal. For simplify, we can set $\theta_1 = \theta_2 = \theta_3 = \theta$

  - – Caculate $\mathcal{D}$
  - – **if** (all elements in $\mathcal{D}$ are equale or larger than 0) **then break** the *Inner Loop* and return $\mathbf{C}'_i$
    else $\theta \leftarrow \theta + \frac{\pi}{18}$ and repeat the *Inner Loop*
- If $\theta = 2\pi$ and $\mathcal{D}$ still doesn't meet the conditions, then go back to random point selection and pick a new $\mathbf{x}$

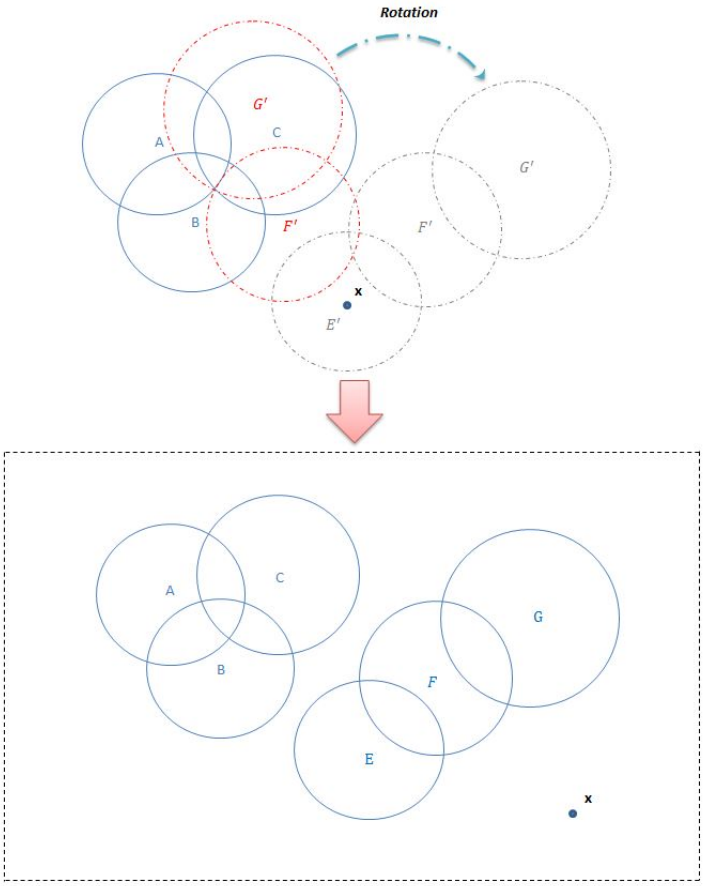


Figure 11: Rotation

(d) $\mathbf{C}_i = [\mathbf{C}_{i-1}^{\mathsf{T}}, \mathbf{C}_i'^{\mathsf{T}}]^{\mathsf{T}}$

(e) $i \leftarrow i + 1$

3. Return $\mathbf{C}_i$

10 Examples

Figure 12 is an example of factor data on human encountering with great white sharks and data is collected by Doctor Pierre-Jerome Bergeron [2]. In this example, it shows the relationship among nationality, time and fatality. Here, we need to notice that the supplementary sets of "AM", "Australia and USA", "Fatality" are "PM", "others", "Survive", respectively. So, any disjoint parts are either fall into sets or supplementary sets. The *stress* of this example is 0.06922327.

Figure 13 gives some two way intersections missing data set

$$venn plot(c(\quad A = 803, B = 304, C = 1015, D = 1100, E = 1005, f = 967, H = 3020,$$

$$C\&D = 1000, B\&C = 248, A\&B\&C = 185, A\&D\&E = 327, C\&D\&f\&H = 846))$$

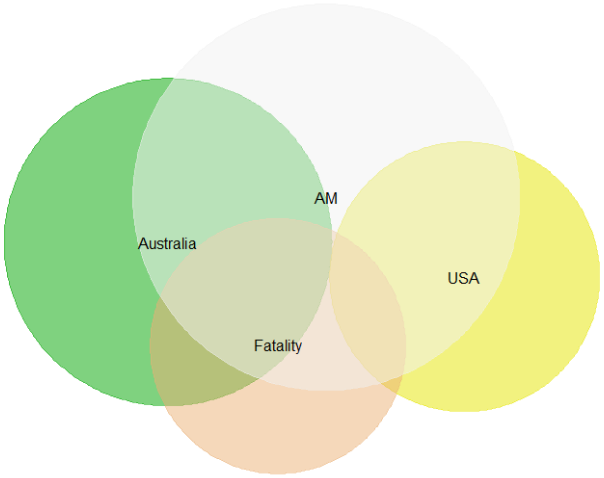


Figure 12: sharks data frame

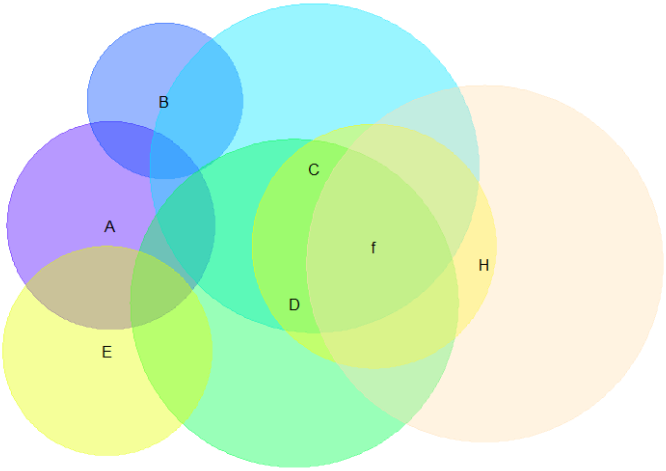


Figure 13: artifical data sets with some two way intersections missing

In this example, some two way intersections missing, like $\{A\&B, A\&C, A\&D, \ldots\}$, however, we can use parameters μ to generate and optimize λ with *stress* 0.08181548.
Figure 14 gives a comparison between 2D version and 3D version.

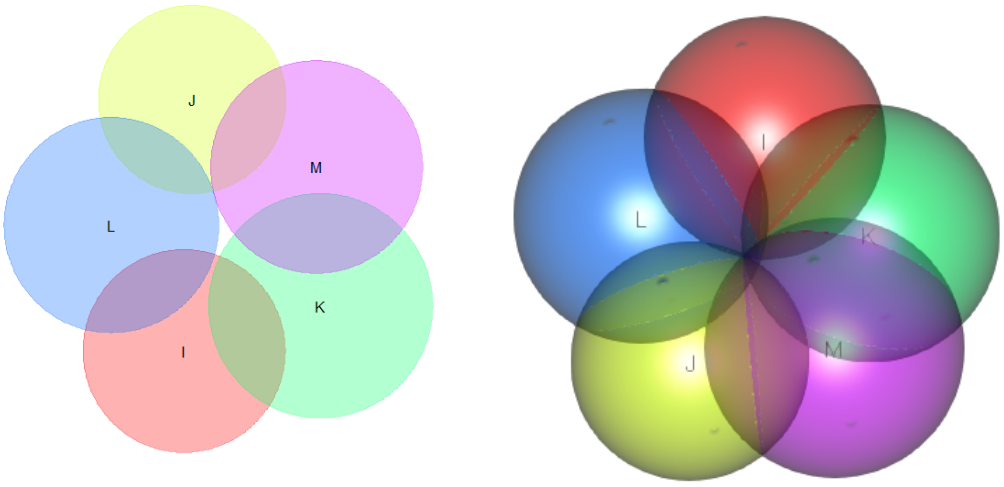


Figure 14:

In this example, stress is 8.75×10^{-5} in 2D and 0.0036 in 3D

11 Comparison with other Venn and Euler Algorithms

In this section, we want to compare `vennplot(...)` with other popular approaches to the circular area-proportional Venn and Euler algorithms.

11.1 `venneuler`

Here, we generate three to eight disjoint data sets 100 times. In each generation, values in each disjoint set are independently and randomly generated from uniform distribution, with a lower bound of 0 and a upper bound of 100. Then, we calculate and compare *stess* of `venneuler(...)` and `vennplot( scalemethod = ''NelderMead'')`. Figure 15 shows the comparison.

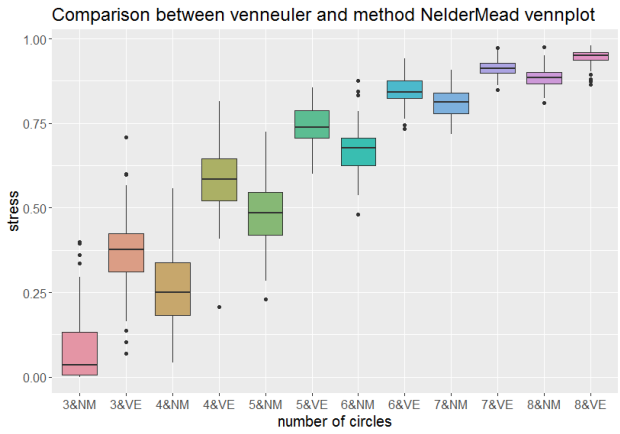


Figure 15: Comparison

We can find the stress of `vennplot( scalemethod = ‘‘NelderMead’’)` is much lower in each number data sets. And we need to notice, this generation is based on one group (undirected connected component). If the data sets have one more groups, the performance of `venneuler(...)` may be worse. Figure 16 shows a fifteen circles venn and euler diagram with input data sets

$$\begin{aligned} vennplot(c(\quad & A = 80, B = 50, C = 100, D = 100, E = 100, A\&C = 30, A\&D = 30, \\ & B\&E = 30, A\&E = 40, f = 50, g = 60, h = 40, g\&f = 20, B\&h = 10, \\ & i = 100, j = 40, k = 50, l = 100, k\&l = 20, m = 30, l\&m = 20, o = 50, p = 60, o\&p = 30)) \end{aligned}$$

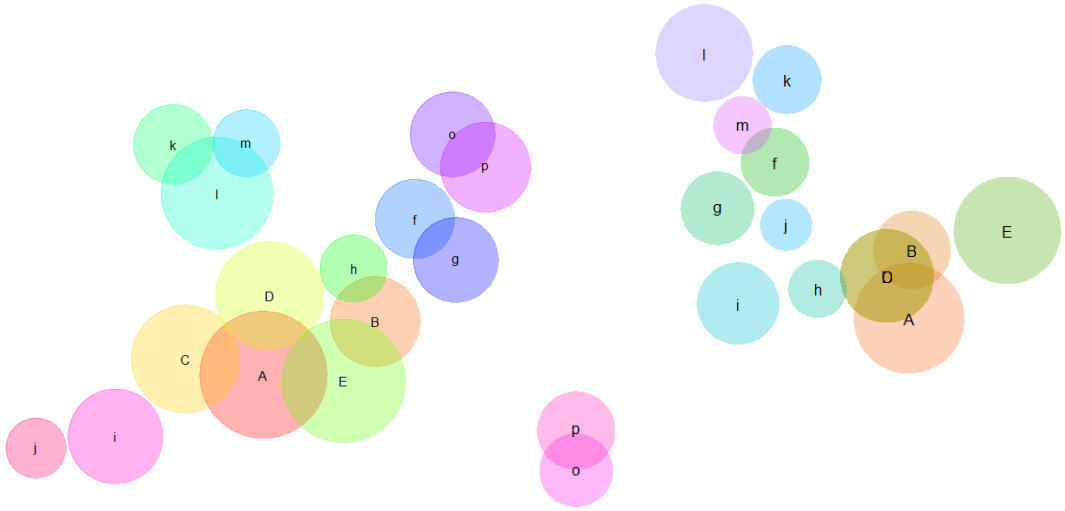


Figure 16: left one is `vennplot( scalemethod = ‘‘NelderMead’’)` and right one is `venneuler(...)`

The *stress* is 0.001463784 in `vennplot( ...)` and 0.3641411 in `venneuler(...)`.

11.2 `venn.js`

`venn.js(...)` is created by `JavaScript` [1, 11]. Since `JavaScript` is a dynamic and interpreted client-side programming language, often used to make webpages interactive and provide online programs [22]. It is hard to extract coordinates and radii to compute *stress*. Thus, we can start with his algorighm and recode in `R`. Figure 17 shows the comparison between `venn.JS(...)` and `vennplot( scalemethod = ‘‘NelderMead’’)`.

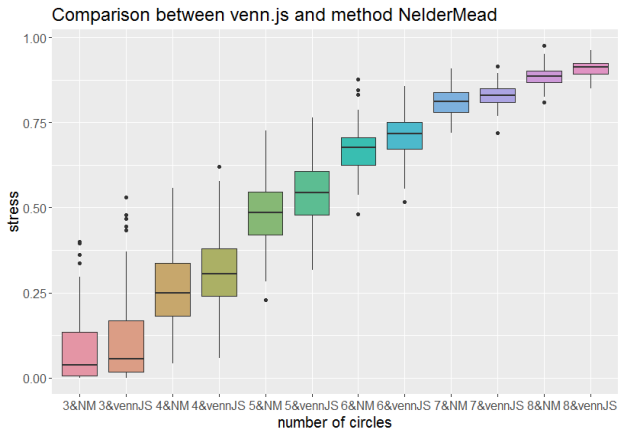


Figure 17: Comparison

We can find the stress of `vennplot( scalemethod = ‘‘NelderMead’’)` is slightly lower in each number data sets. However, if multiple groups are given, the adventage of `vennplot( scalemethod = ‘‘NelderMead’’)` will be obvious.

References

[1] Comparison with venneuler, 2015.

[2] sharkattackinfo.com, 2017.

[3] M. E. Baron. A Note on the Historical Development of Logic Diagrams: Leibniz, Euler and Venn. *The Mathematical Gazette*, 1969.

[4] M. J. Box, D. Davies, and W. H. Swann. *Non-linear optimization techniques*. 1969.

[5] A. L. Byrd, C. Deming, S. K. B. Cassidy, O. J. Harrison, W.-I. Ng, S. Conlan, N. C. S. Program, Y. Belkaid, J. A. Segre, and H. H. Kong. Staphylococcus aureus and Staphylococcus epidermidis strain diversity underlying pediatric atopic dermatitis. *Science*, 2017.

[6] H. Chen and P. C. Boutros. Venndiagram: a package for the generation of highly-customizable Venn and Euler diagrams in R. *BMC Bioinformatics*, 2011.

[7] W. Cherry and R. Oldford. Picturing Probability: the poverty of venn diagrams, the richness of Eikosograms. 2003.

[8] W. S. Cleveland and R. McGill. Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods . *American Statistical Association*, 1984.

[9] W. S. Cleveland and R. McGill. Graphical Perception and Graphical Methods for Analyzing Scientific Data. *Science*, 1985.

[10] W. S. Cleveland and R. McGill. Graphical Perception : The Visual Decoding of Quantitative Information on Graphical Displays of Data . *Royal Statistical Society*, 1987.

[11] B. Frederickson. A better algorithm for area proportional Venn and Euler diagrams. 2015.

[12] J. Hopcroft and R. Tarjan. Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 1973.

[13] P. Jaccard. Distribution de la Flore Alpine dans le Bassin des Dranses et dans quelques regions voisines. *Bulletin de la Socieete vaudoise des sciences naturelles*, 1901.

[14] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. Structure and evolution of blogspace. *Structure and evolution of blogspace*, 2004.

[15] D. P. LePage, J. A. Metcalf, S. R. Bordenstein, J. On, J. I. Perlmutter, J. D. Shropshire, E. M. Layton, L. J. Funkhouser-Jones, J. F. Beckmann, and S. R. Bordenstein. Prophage WO genes recapitulate and enhance Wolbachia-induced cytoplasmic incompatibility. *Nature*, 2017.

[16] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 1969.

[17] E. Polak and G. Ribiere. Note sur la convergence de méthodes de directions conjuguées. *Mathematical Modelling and Numerical Analysis*, 1969.

[18] A. Rahman and W. Oldford. Euclidean distance matrix completion and point configurations from the minimal spanning tree. 2016.

[19] J. R. Shewchuk. Technical report. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, 1994.

[20] E. S. A. Sollars, A. L. Harper, L. J. Kelly, C. M. Sambles, R. H. Ramirez-Gonzalez, D. Swarbreck, G. Kaithakottil, E. D. Cooper, C. Uauy, L. Havlickova, G. Worswick, D. J. Studholme, J. Zohren, D. L. Salmon, B. J. Clavijo, Y. Li, Z. He, A. Fellgett, L. V. McKinney, L. R. Nielsen, G. C. Douglas, E. D. Kjaer, J. A. Downie, and D. Boshier. Genome sequence and genetic diversity of European ash trees. *Nature*, 2016.

[21] S. Stevens. On the psychophysical law. *Psychological Review*, 1957.

[22] Wikipedia. Javascript.

[23] Wikipedia. Rotation.

[24] L. Wilkinson. Exact and approximate area-proportional circular Venn and Euler diagrams. *IEEE Trans Vis Comput Graph*, 2012.

[25] D. Zhang, M. Yu, P. Hu, S. Peng, Y. Liu, W. Li, C. Wang, S. He, W. Zhai, Q. Xu, and L. Chen. Genetic Adaptation of Schizothoracine Fish to the Phased Uplifting of the Qinghai - Tibetan Plateau. *Genetics*, 2017.