Venn diagrams
Technical details and regression checks

Jonathan Swinton

19$^{\text{th}}$ June, 2007

- Try CR for weight=0
- Plot faces for Chow-Ruskey
- General set membership
- implement not showing dark matter eg Fig 1
- Different choices of first and second sets for AWFE
- Add in the equatorial sets for AWFE
- AWFE-book like figures
- naming of weights for triangles
- likesquares argument for triangles
- likesquares argument for 4-squares
- central dark matter
- Comment on triangles
- Comment on AWFE return geometry
- calculate three circle areas correctly
- text boxes
- use grob objects/printing properly
- "Exact" slot mess
- proper data handling:
- choose order;
- cope with missing data including missing zero intersection;
- Define weights via names
- graphical parameters
- discuss Chow-Ruskey zero=nonsimple

1 Venn objects

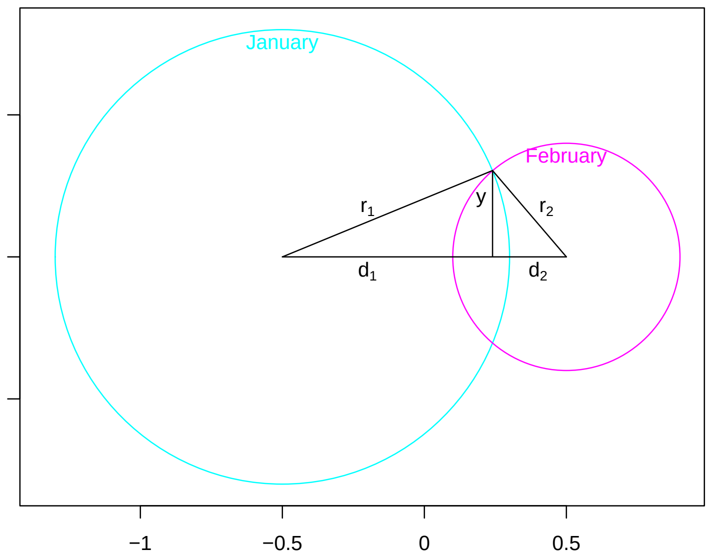
```
> library(Vennerable)
> Vcombo <- Venn(SetNames = c("Female",
+     "Visible Minority", "CS Major"),
+     Weight = c(0, 4148, 409, 604,
+         543, 67, 183, 146))
```

For a running example, we use sets named after months, whose elements are the letters of their names.

```
> setList <- strsplit(month.name, split = "")
> names(setList) <- month.name
> VN3 <- VennFromSets(setList[1:3])
> V2 <- VN3[, c("January", "February"),
+     ]

> V4 <- VennFromSets(setList[1:4])
> V4f <- V4
> V4f@IndicatorWeight[, ".Weight"] <- 1

> setList <- strsplit(month.name, split = "")
> names(setList) <- month.name
> VN3 <- VennFromSets(setList[1:3])
> V2 <- VN3[, c("January", "February"),
+     ]

> V3.big <- Venn(SetNames = month.name[1:3],
+     Weight = 2^(1:8))
> V2.big <- V3.big[, c(1:2)]

> Vempty <- VennFromSets(setList[c(4,
+     5, 7)])
> Vempty2 <- VennFromSets(setList[c(4,
+     5, 11)])
> Vempty3 <- VennFromSets(setList[c(4,
+     5, 6)])
```

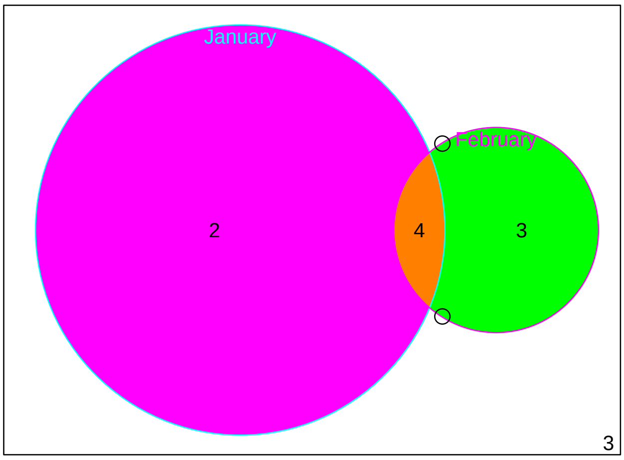
2 Two circles

2.1 Two circles



We rely on the relationships

$$\begin{aligned} d_1 &= (d^2 - r_2^2 + r_1^2)/(2d) \\ d_2 &= d - d_1 \\ y &= (1/(2d))\sqrt{4d^2r1^2 - (d^2 - r2^2 + r1^2)^2} \end{aligned}$$


January
February
2
4
3
3

2.2 Weighted 2-set Venn diagrams for 2 Sets

2.2.1 Circles

It is always possible to get an exactly area-weighted solution for two circles as shown in Figure 1.

```
       00        10        01        11
       NA  67.98454 135.95841 271.95839

[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

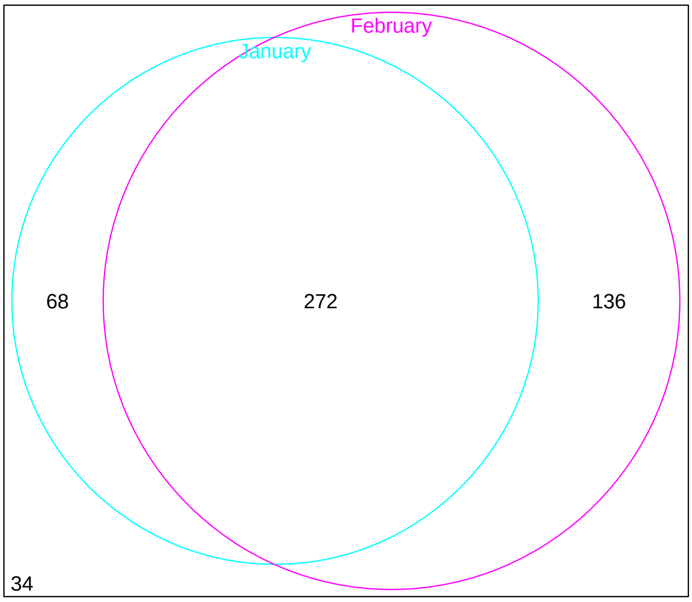


Figure 1: Weighted 2d Venn

2.3 2-set Euler diagrams

2.3.1 Circles

```
       00        10        01        11
       NA 0.1353743 3.1339495 3.8635058
```

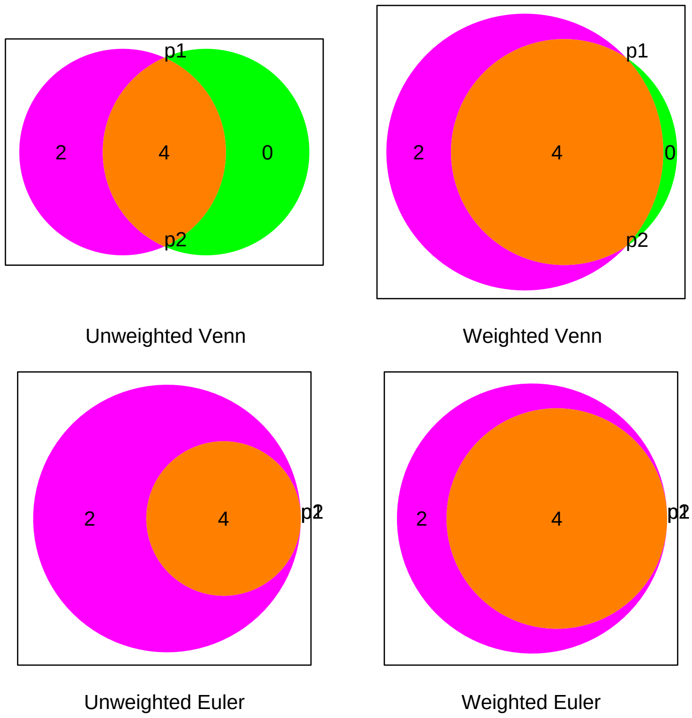


Figure 2: Effect of the Euler and `doWeights` flags.

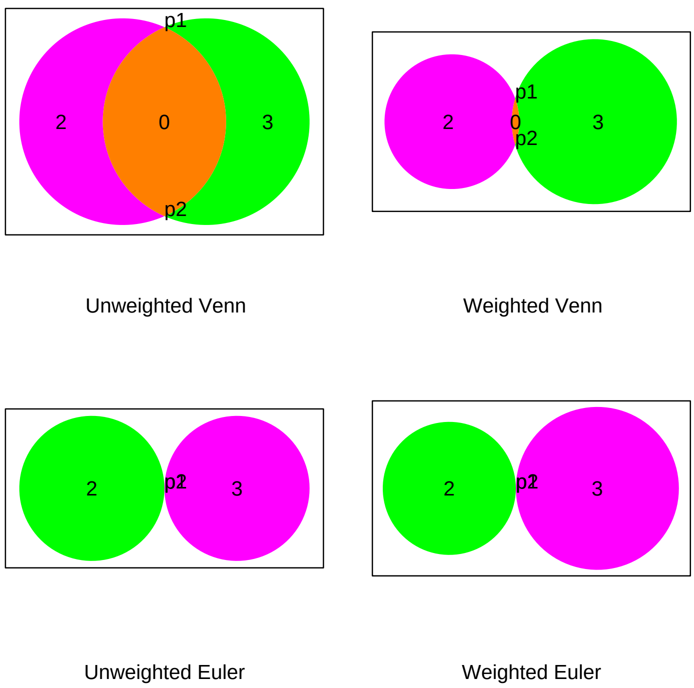


Figure 3: As before for a different set of weights

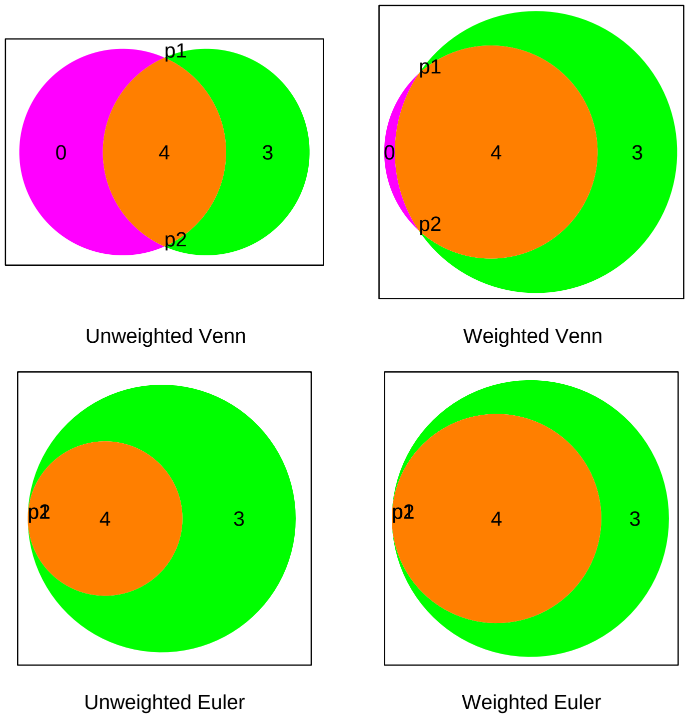
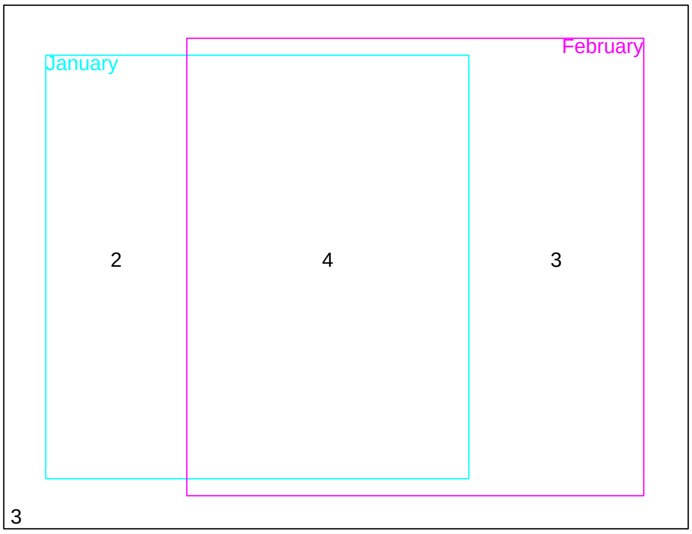


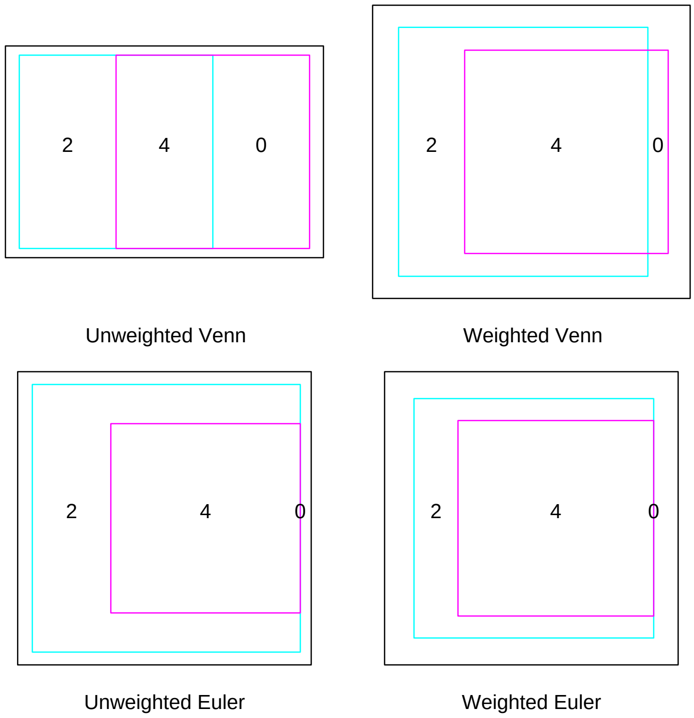
Figure 4: As before for a different set of weights

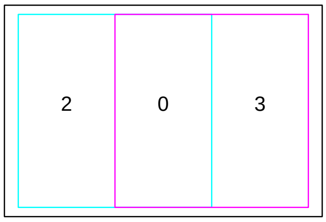
3 Two squares



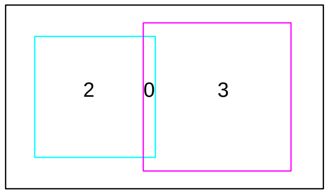
3.0.2 Weights

3.0.3 Squares

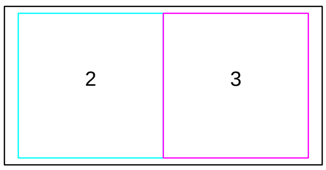




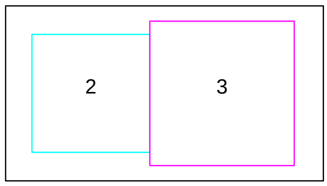
Unweighted Venn



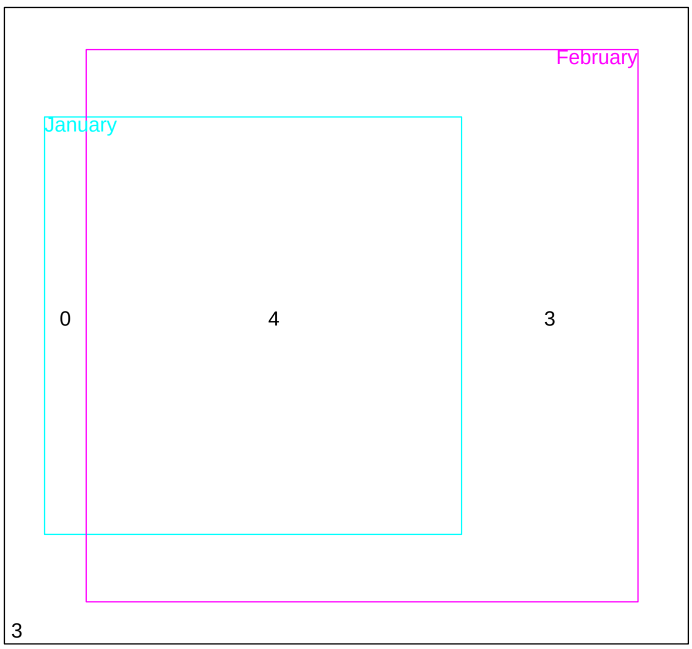
Weighted Venn



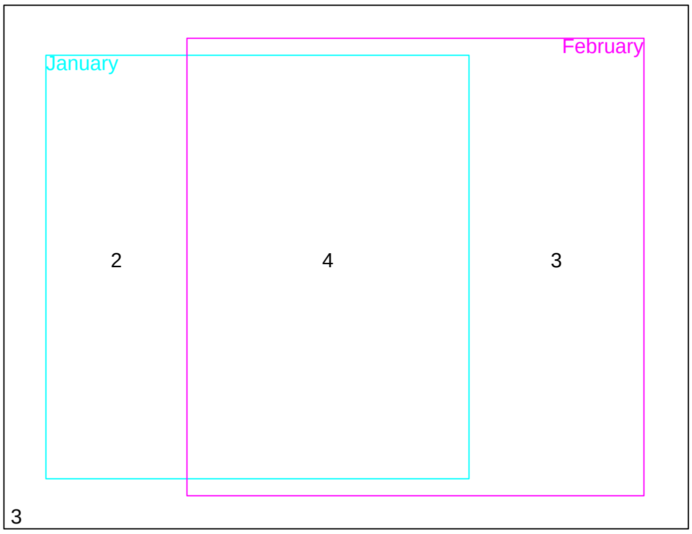
Unweighted Euler



Weighted Euler



3.1 Two squares



4 Three circles

```
> plot(Vcombo, doWeights = FALSE)
```

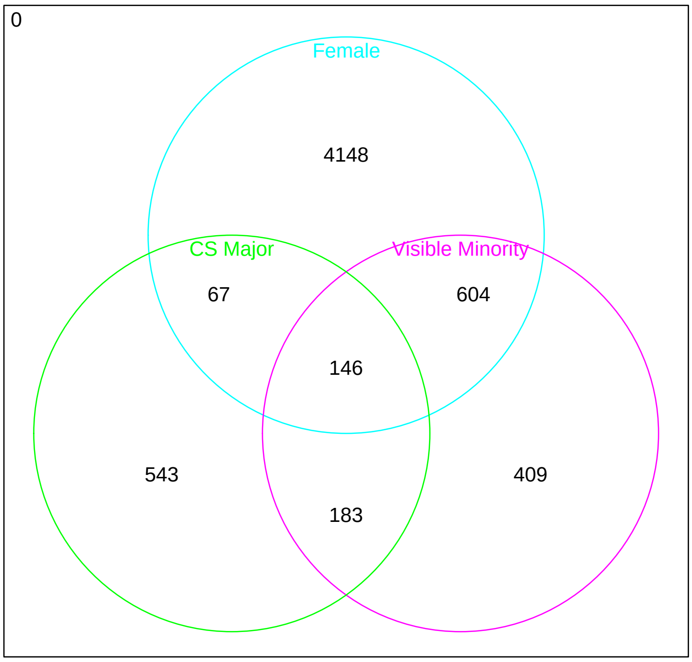


Figure 5: A three-circle Venn diagram

4.0.1 Weights

There is no general way of creating area-proportional 3-circle diagrams. The package makes an attempt to produce approximate ones.

```
       000        100        010        110
2161.42418 4050.03149  397.85795  595.97496
       001        101        011        111
 522.89901   69.82813  181.87792  139.65626

[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

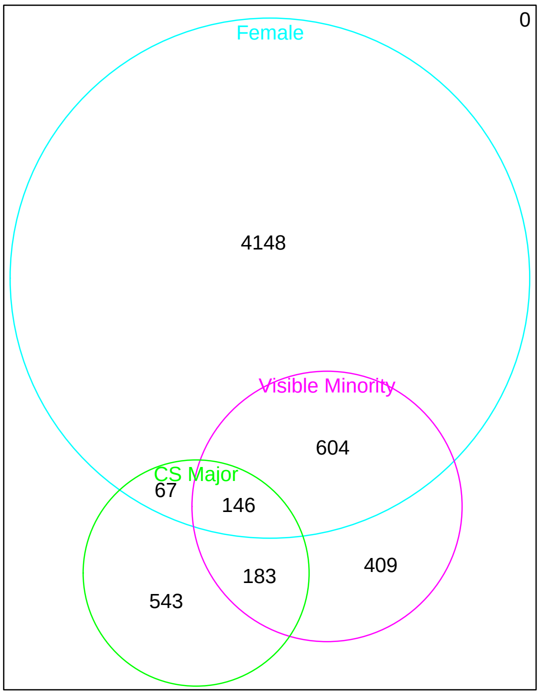


Figure 6: 3D Venn diagram. All of the areas are correct to within 10%

TODO check areas

4.1 Three circles

If not uniform, we have to compute the centroids by quadrature

5 Three Triangles

The triangular Venn diagram on 3-sets lends itself nicely to an area-proportional drawing under some contrainsts on the weights

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

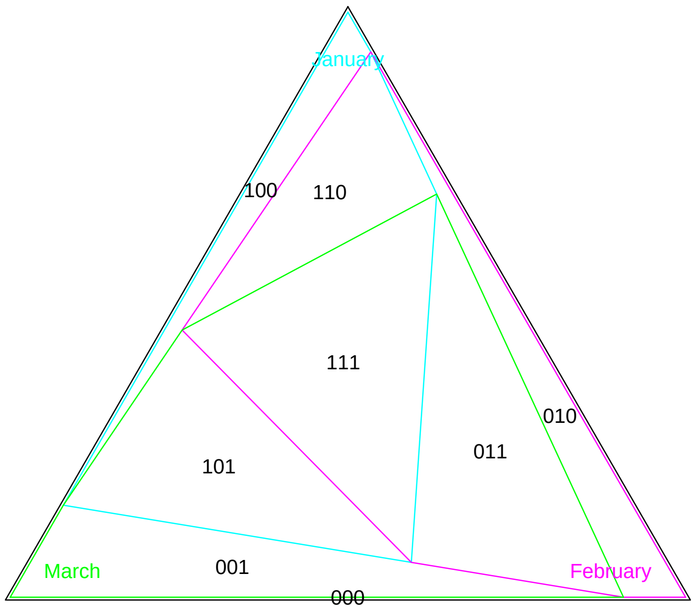


Figure 7: Triangular Venn with external universe

5.1 Triangular Venn diagrams

Has intersection shapes so would be easy to define faces but we don't. No nodes either.

5.1.1 Triangles

```
          000           100           010
1.776357e-15 2.000000e+00 3.000000e+00
          110           001           101
2.000000e+00 3.000000e+00 0.000000e+00
          011           111
0.000000e+00 2.000000e+00
```

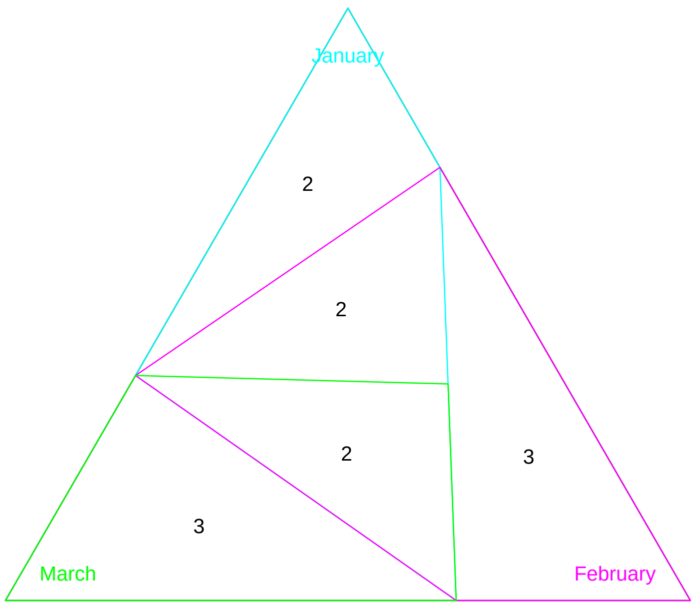


Figure 8: 3d Venn triangular with one empty intersection

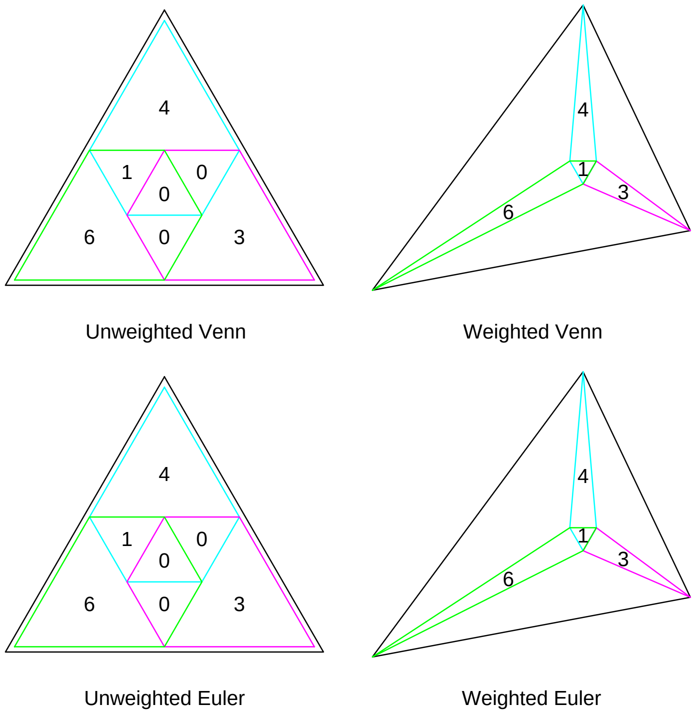
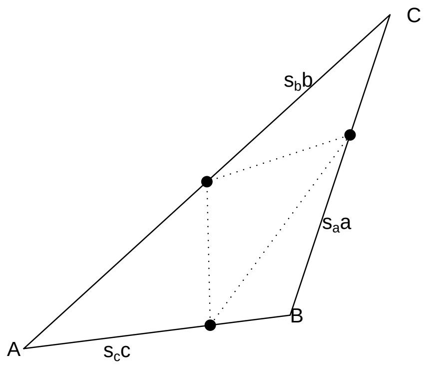


Figure 9: 3d Venn triangular with two empty intersection



Given a triangle ABC of area Δ and some nonnegative weights $w_a + w_b + w_c < 1$ we want to set s_c, s_a and s_b so that the areas of each of the apical triangles are Δ-proportional to w_a, w_b and w_c. This means

$$\begin{aligned} s_c(1-s_b)bc\sin A &= 2w_a\Delta & (1)\\ s_a(1-s_c)ca\sin B &= 2w_b\Delta & (2)\\ s_b(1-s_a)ab\sin C &= 2w_c\Delta & (3)\end{aligned}$$

So

$$\begin{aligned} s_c(1-s_b) &= w_a & (4)\\ s_a(1-s_c) &= w_b & (5)\\ s_b(1-s_a) &= w_c & (6)\end{aligned}$$

$$\begin{aligned} s_b &= 1-w_a/s_c & (7)\\ s_a &= w_b/(1-s_c) & (8)\\ (s_c-w_a)(1-s_c-w_b) &= s_c(1-s_c)w_c & (9)\end{aligned}$$

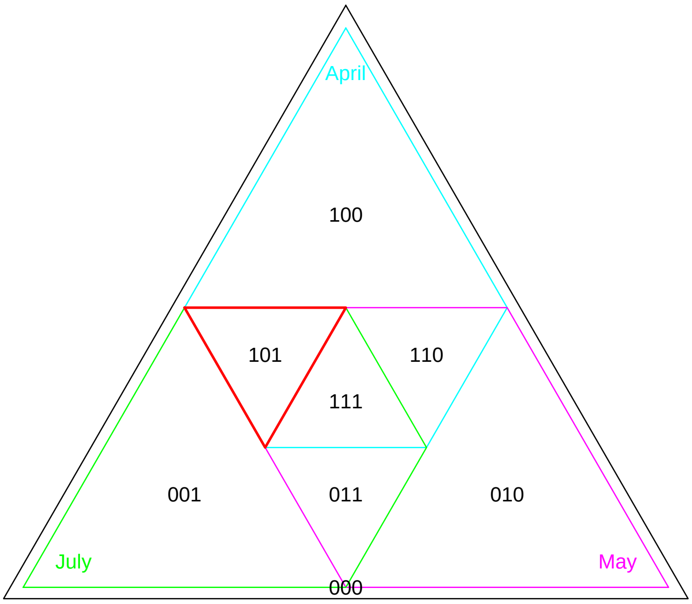
$$s_c^2(1-w_c)+s_c(w_b+w_c-w_a-1)+w_a(1-w_b) = 0 \qquad (10)$$

Iff

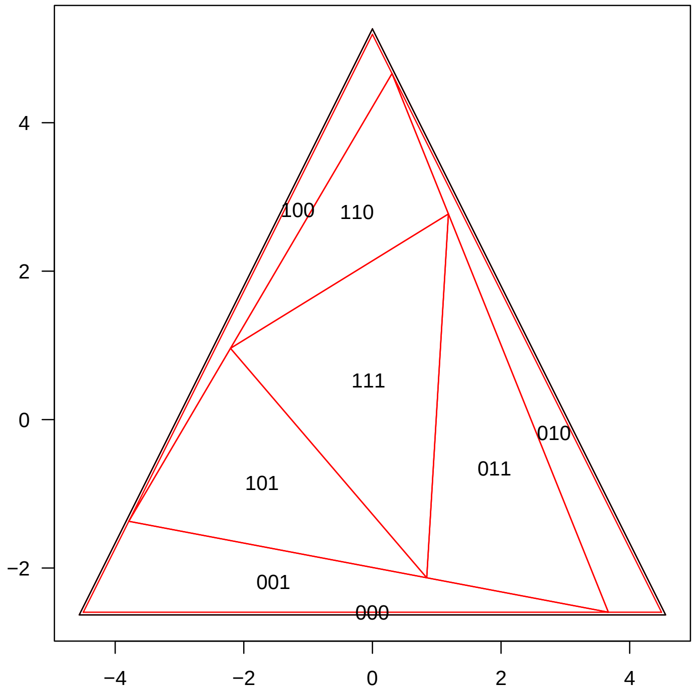
$$4w_aw_bw_c < (1-(w_a+w_b+w_c))^2 \qquad (11)$$

this has two real solutions between w_a and $1-w_b$.

```
[1] TRUE
```



5.2 Three triangles



6 Three Squares

This is a version of the algorithm suggested by [1]. TODO likesquares

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

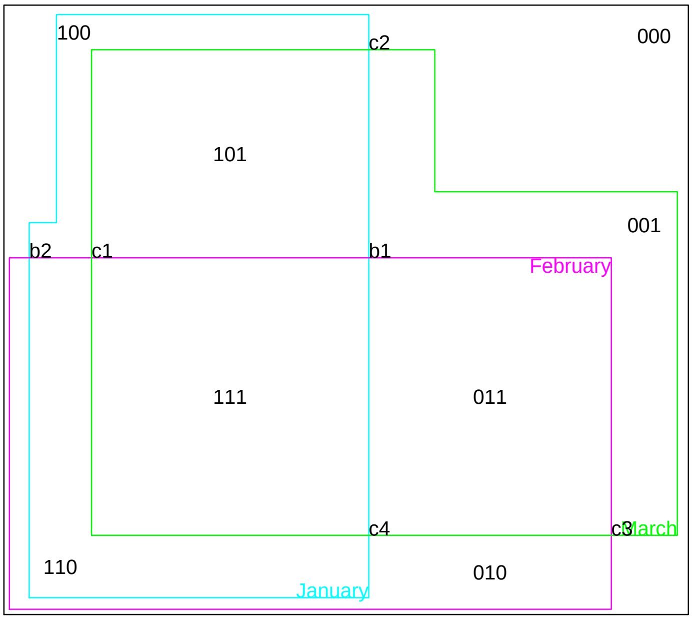
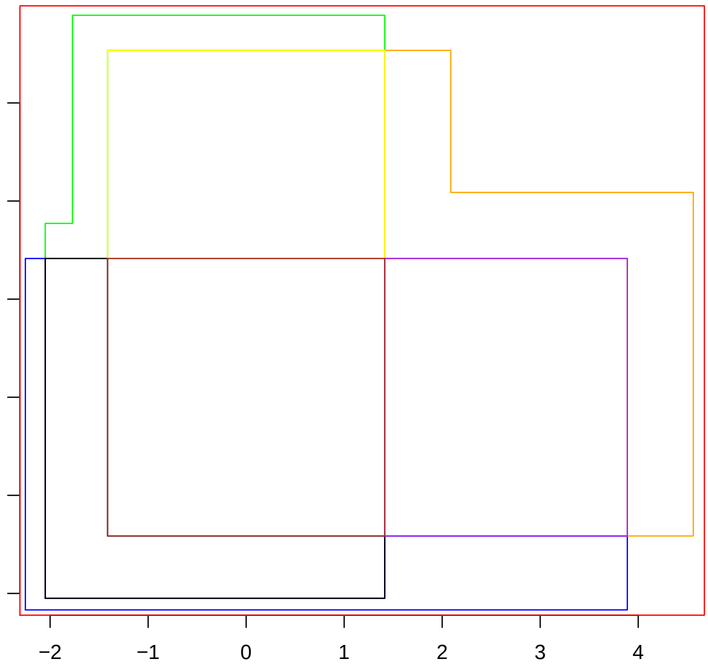


Figure 10: Weighted 3-set Venn diagram based on the algorithm of [1]

6.1 Three squares



7 Four squares

7.1 Unweighted 4-set Venn diagrams

```
> doans <- function(V4, s, likeSquares) {
+     S4 <- compute.S4(V4, s = s, likeSquares = likeSquares)
+     CreateViewport(S4)
+     PlotSetBoundaries(S4, gp = gpar(lwd = 4:1,
+         col = trellis.par.get("superpose.symbol")$col))
+     UpViewports()
+ }
```

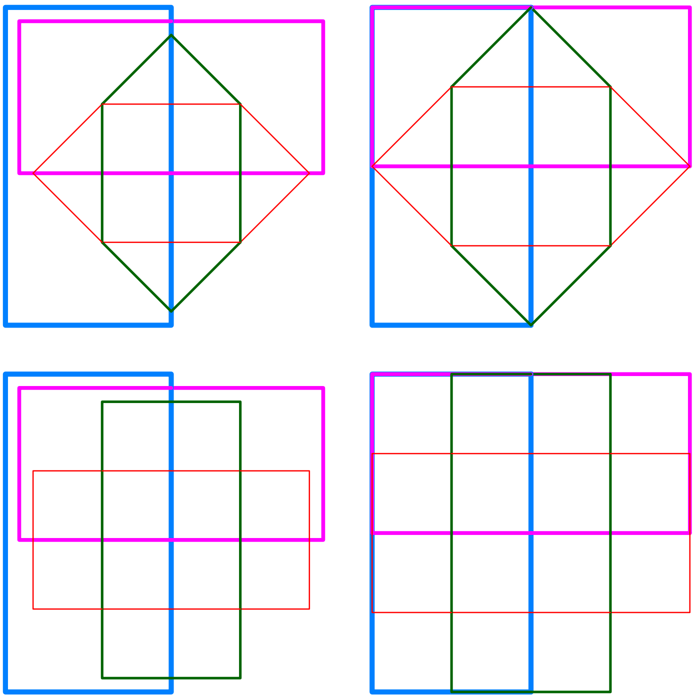


Figure 11: Four variants on the four-squares

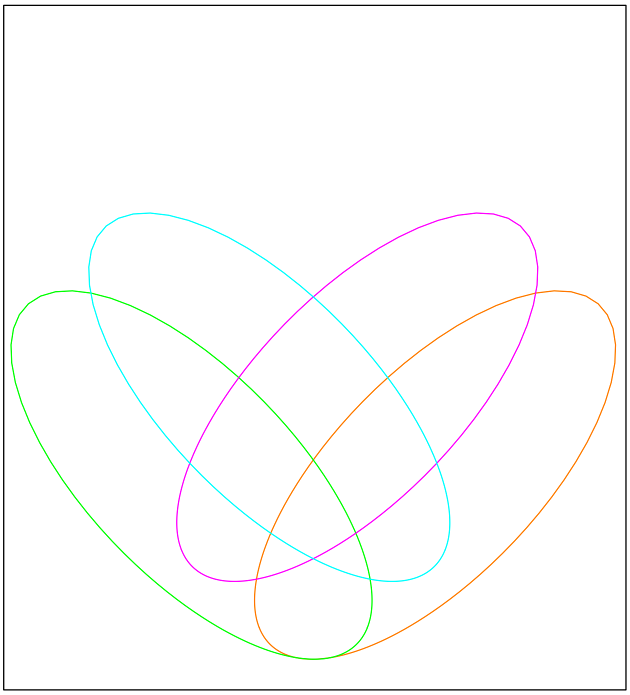
7.2 Four squares

```
$`p1|p2`
lines[GRID.lines.2938]

$`p1|p7`
lines[GRID.lines.2939]

$`p2|p3`
lines[GRID.lines.2940]

$`p2|p11`
lines[GRID.lines.2941]

$`p3|p4`
lines[GRID.lines.2942]

$`p3|p11`
lines[GRID.lines.2943]

$`p4|p5`
lines[GRID.lines.2944]

$`p4|p9`
lines[GRID.lines.2945]

$`p5|p6`
lines[GRID.lines.2946]

$`p5|p13`
lines[GRID.lines.2947]

$`p6|p1`
lines[GRID.lines.2948]

$`p6|p13`
lines[GRID.lines.2949]

$`p7|p8`
lines[GRID.lines.2950]

$`p7|p12`
lines[GRID.lines.2951]

$`p8|p4`
lines[GRID.lines.2952]

$`p8|p12`
lines[GRID.lines.2953]

$`p9|p10`
lines[GRID.lines.2954]

$`p9|p14`
lines[GRID.lines.2955]

$`p10|p1`
lines[GRID.lines.2956]

$`p10|p14`
```

```
$`p1|p2`
lines[GRID.lines.3026]

$`p1|p7`
lines[GRID.lines.3027]

$`p2|p3`
lines[GRID.lines.3028]

$`p2|p11`
lines[GRID.lines.3029]

$`p3|p4`
lines[GRID.lines.3030]

$`p3|p11`
lines[GRID.lines.3031]

$`p4|p5`
lines[GRID.lines.3032]

$`p4|p9`
lines[GRID.lines.3033]

$`p5|p6`
lines[GRID.lines.3034]

$`p5|p13`
lines[GRID.lines.3035]

$`p6|p1`
lines[GRID.lines.3036]

$`p6|p13`
lines[GRID.lines.3037]

$`p7|p8`
lines[GRID.lines.3038]

$`p7|p12`
lines[GRID.lines.3039]

$`p8|p4`
lines[GRID.lines.3040]

$`p8|p12`
lines[GRID.lines.3041]

$`p9|p10`
lines[GRID.lines.3042]

$`p9|p14`
lines[GRID.lines.3043]

$`p10|p1`
lines[GRID.lines.3044]

$`p10|p14`
lines[GRID.lines.3045]
```

8 Four Ellipses

Ellipses don’t have faces or nodes, and can’t have weights sent.



9 AWFE for more than four sets

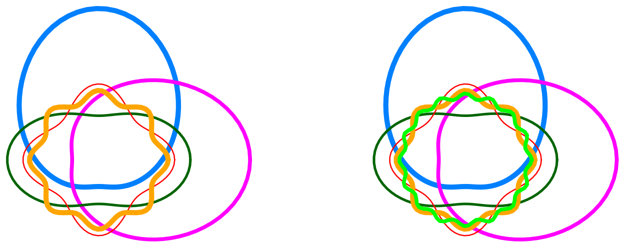
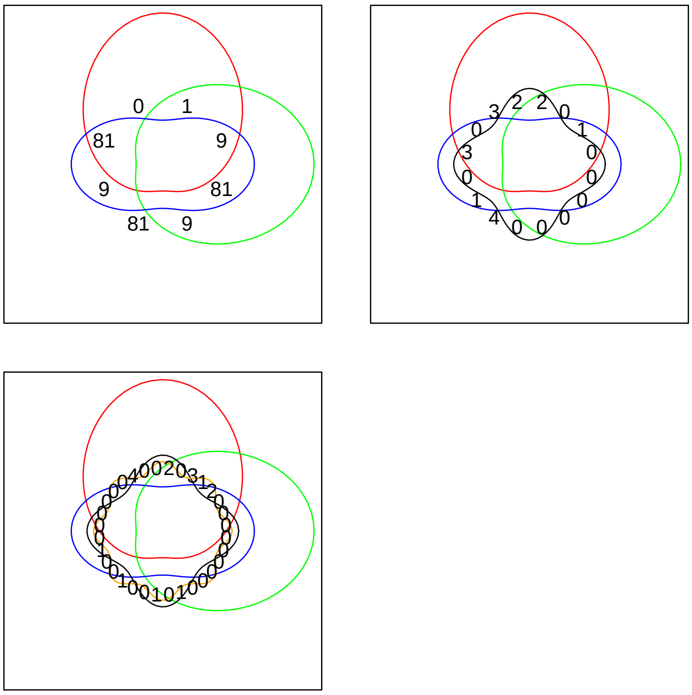


Figure 12: Edwards constructions for five and six sets

10 3, 4 and 5 set Edwards-Venn diagrams



```
> V4 <- VennFromSets(setList[1:4])
> V4f <- V4
> V4f@IndicatorWeight[, ".Weight"] <- 1
```

```
> setList <- strsplit(month.name, split = "")
> names(setList) <- month.name
> VN3 <- VennFromSets(setList[1:3])
> V2 <- VN3[, c("January", "February"),
+     ]
```

11 Chow-Ruskey

See [2, 1].

11.1 Chow-Ruskey diagrams for 3 sets

The general Chow-Ruskey algorithm can be implemented in principle for an arbitrary number of sets provided the weight of the common intersection is nonzero.

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

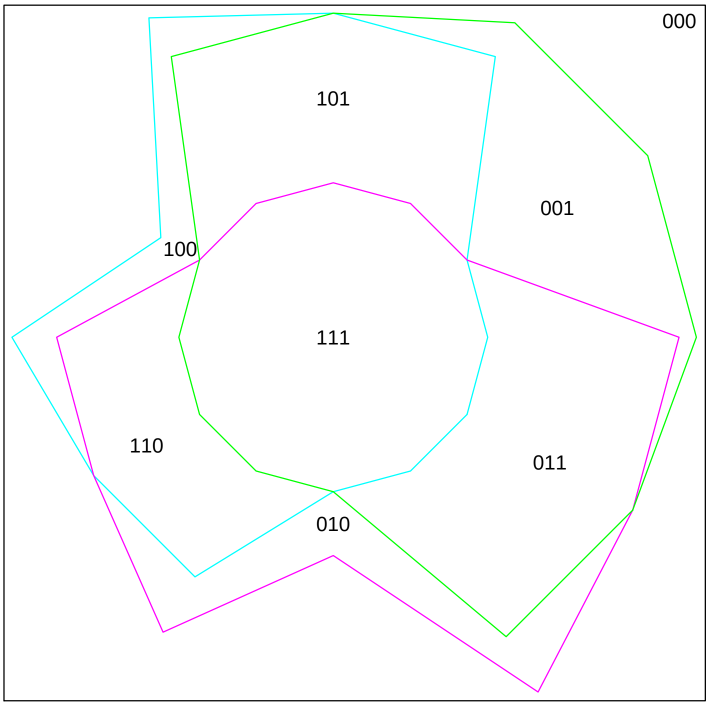
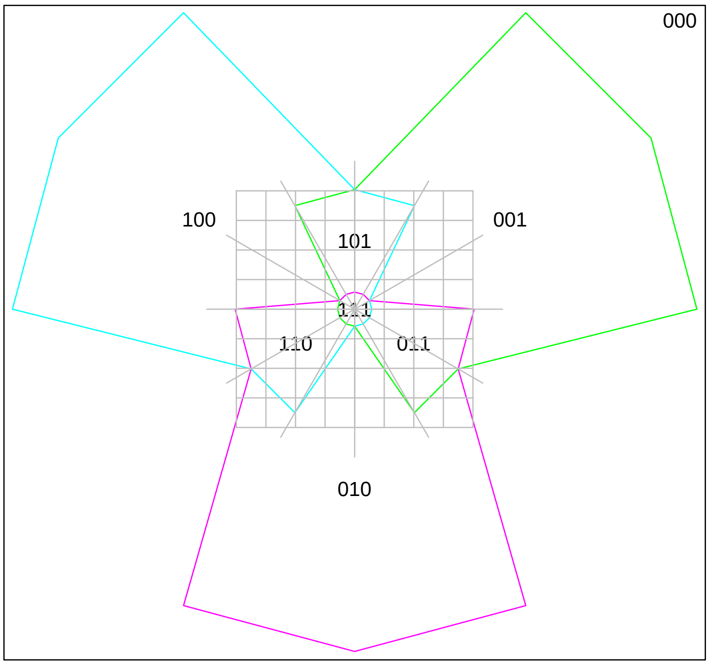


Figure 13: Chow-Ruskey weighted 3-set diagram

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```



```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

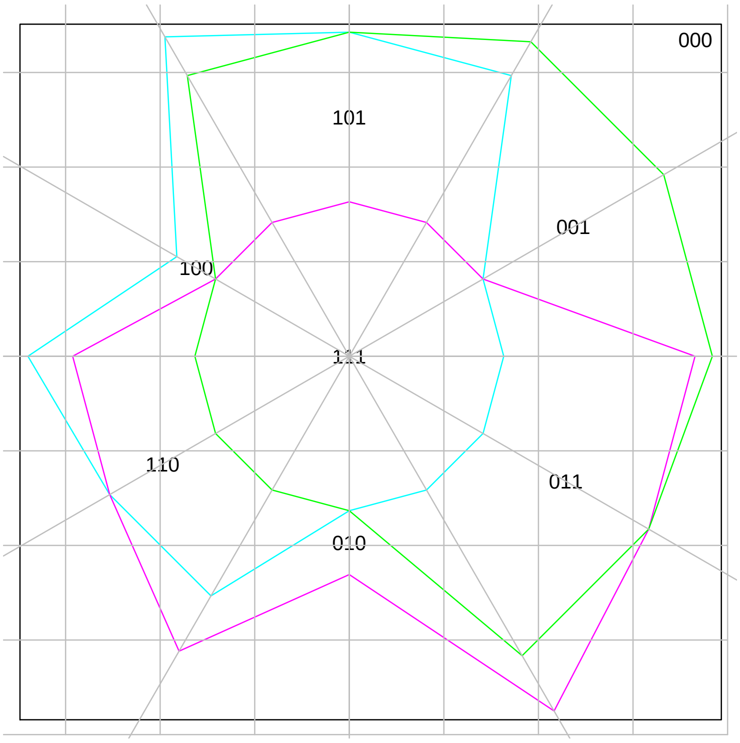


Figure 14: Chow-Ruskey CR3f

11.2 Chow-Ruskey diagrams for 4 sets

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

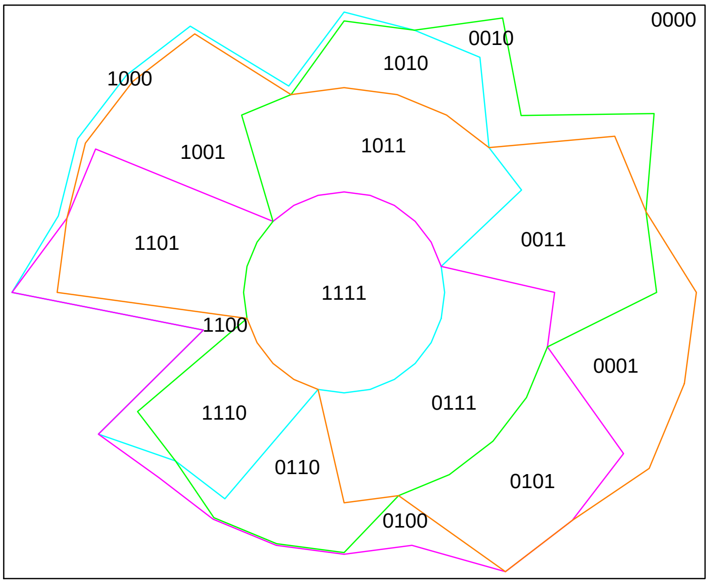


Figure 15: Chow-Ruskey weighted 4-set diagram

```
[1] Area              Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```

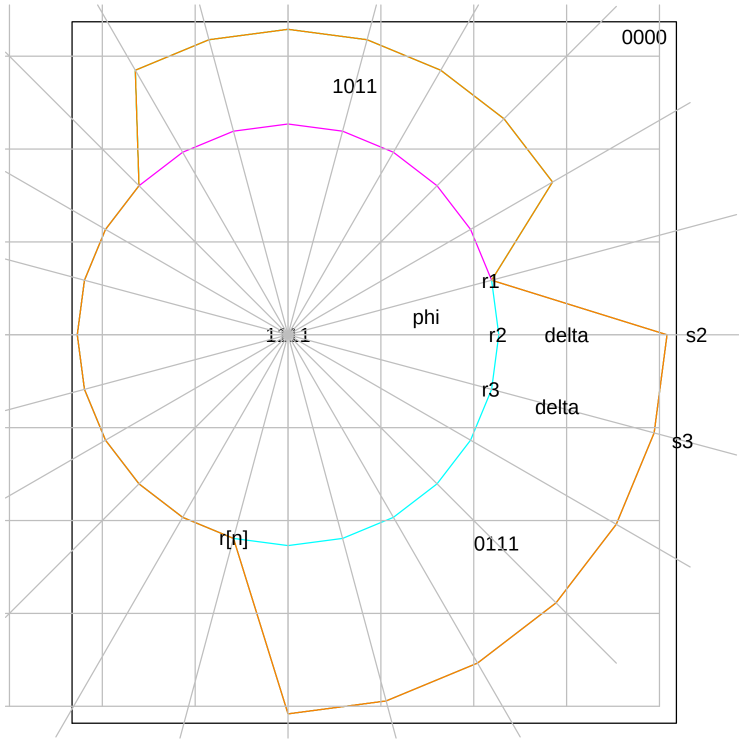


Figure 16: Chow-Ruskey weighted 4-set diagram

The area of the sector $0r_1r_2$ is $\frac{1}{2}r_1r_2\sin\phi$. The area of $0r_1s_2$ is $\frac{1}{2}(r_1(r_2+\delta)\sin\phi)$ and so the area of $r_1r_2s_2$ is $\frac{1}{2}(r_1\delta\sin\phi)$.

The area of $r_2r_2s_2s_3$ is $\frac{1}{2}[(r_3+\delta)(r_2+\delta)-r_3r_2)\sin\phi = \frac{1}{2}[(r_3+r_2)\delta+\delta^2]\sin\phi$.

The total area of the outer shape is

$$A = \frac{1}{2}(\sin\phi)\left[(r_1+r_n)\delta+\sum_{k=2}^{n-2}[(r_{k+1}+r_k)\delta+\delta^2]\right] \quad (12)$$

$$= \frac{1}{2}(\sin\phi)\left[(r_1+r_n)\delta+(n-2)\delta^2+\delta\sum_{k=2}^{n-2}[(r_{k+1}+r_k)]\right] \quad (13)$$

$$= \frac{1}{2}(\sin\phi)\left[(r_1+r_2+2r_3+\ldots+2r_{n-2}+r_{n-1}+r_n)\delta+(n-3)\delta^2\right] \quad (14)$$

so

$$0 = c_a\delta^2+c_b\delta+c_c \quad (15)$$

$$c_a = n-3 \quad (16)$$

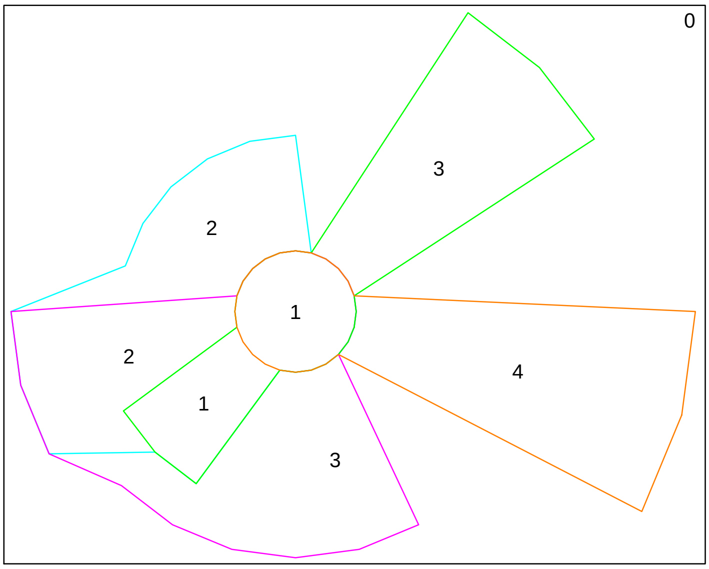
$$c_b = r_1+r_2+2r_3+\ldots+2r_{n-2}+r_{n-1}+r_n \quad (17)$$

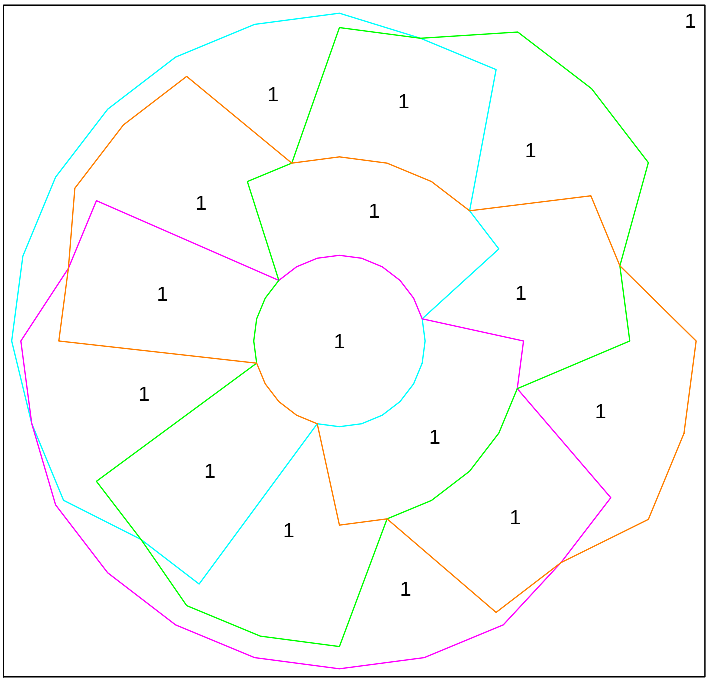
$$c_c = -A/\frac{1}{2}\sin\phi \quad (18)$$

This is implemented in the compute.delta function.

If all the rs are the same then $c_b = [2(n-3)+4]r = (2n-2)r$.

```
[1] Area            Weight
[3] IndicatorString Density
<0 rows> (or 0-length row.names)
```




1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1

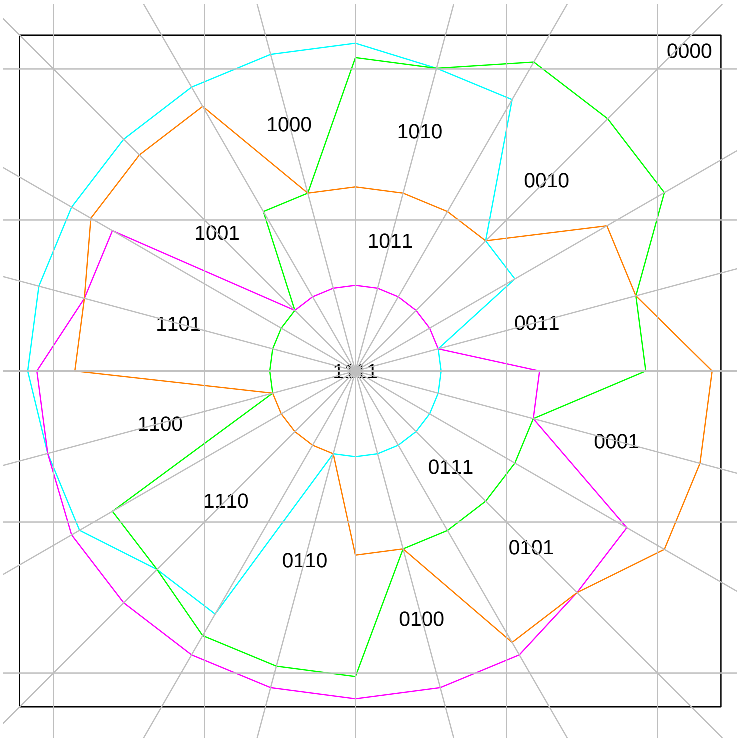


Figure 17: Chow-Ruskey 4

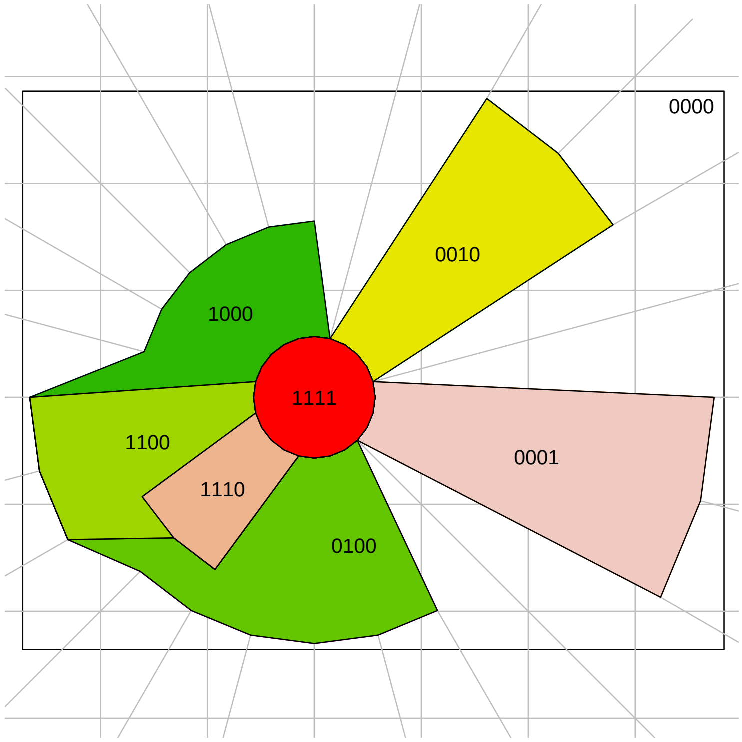


Figure 18: Garish fill

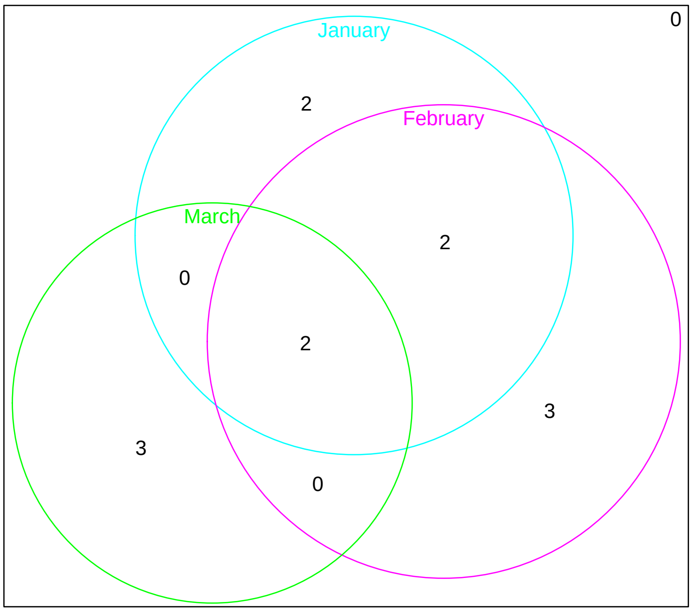
12 Euler diagrams

12.1 3-set Euler diagrams

12.1.1 Circles

There is currently no effect of setting doEuler=TRUE for three circles.

```
NULL

      000       100       010       110
4.3292157 1.4768614 2.4609159 2.4359636
      001       101       011       111
2.4546778 0.4772118 0.4772118 1.4830995
```



There are about 40 distinct ways in which patterns of zero intersections can occur.

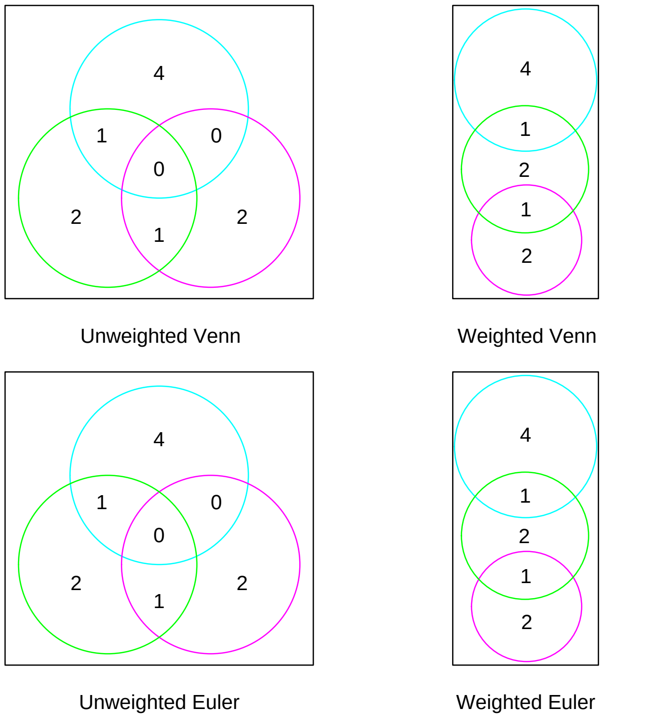


Figure 19: Weighted 3d Venn with an empty intersection

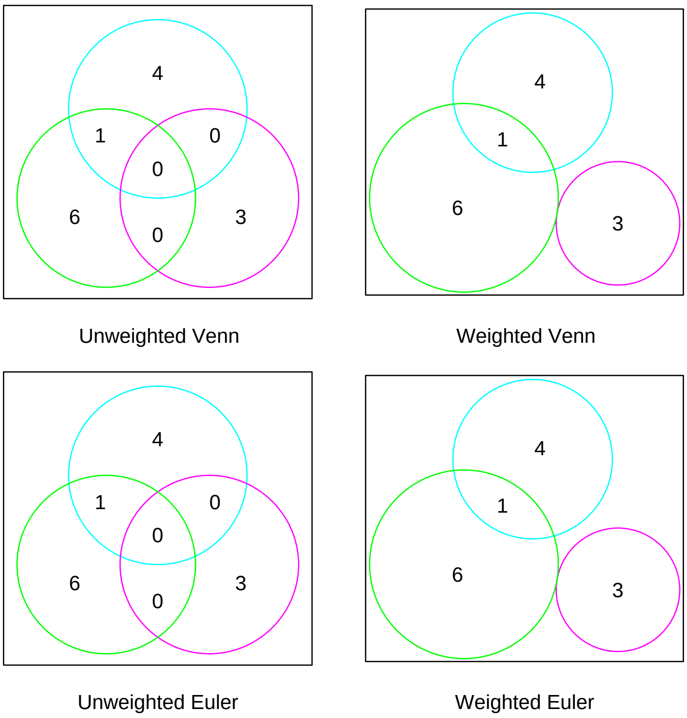


Figure 20: Weighted 3d Venn with two empty intersections

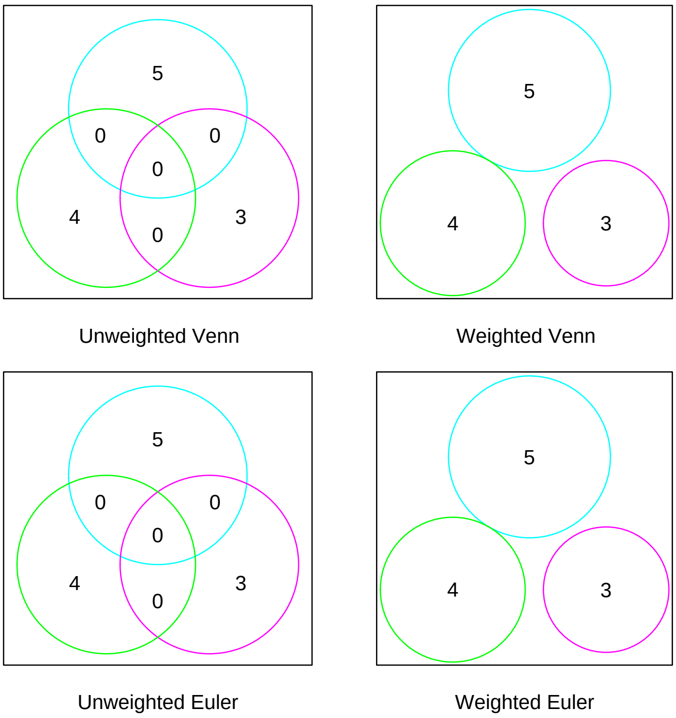


Figure 21: Weighted 3d Venn with three empty intersections

12.1.2 Other examples of circles

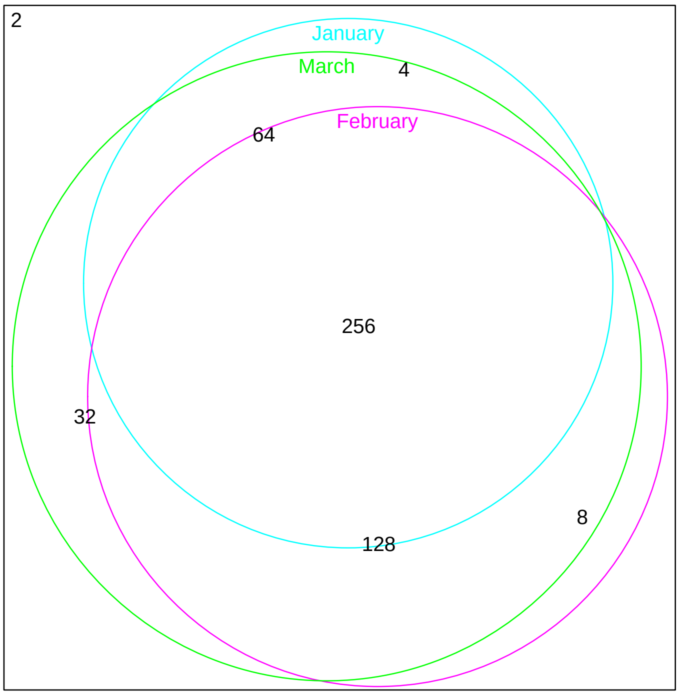
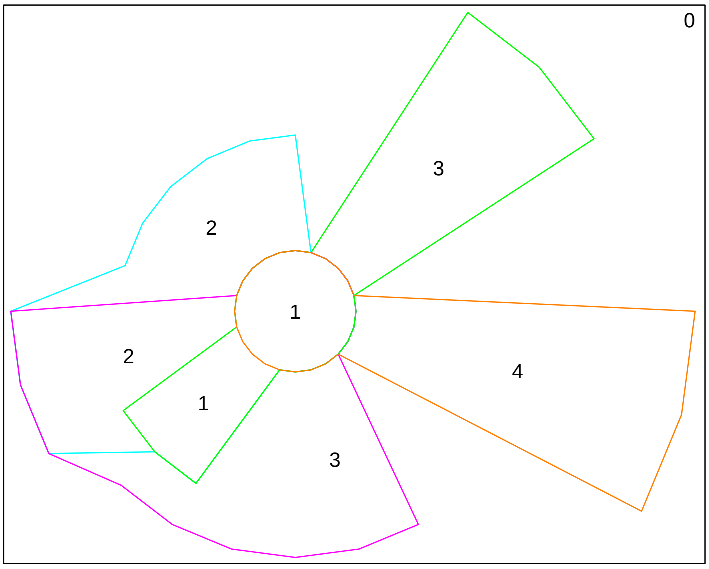


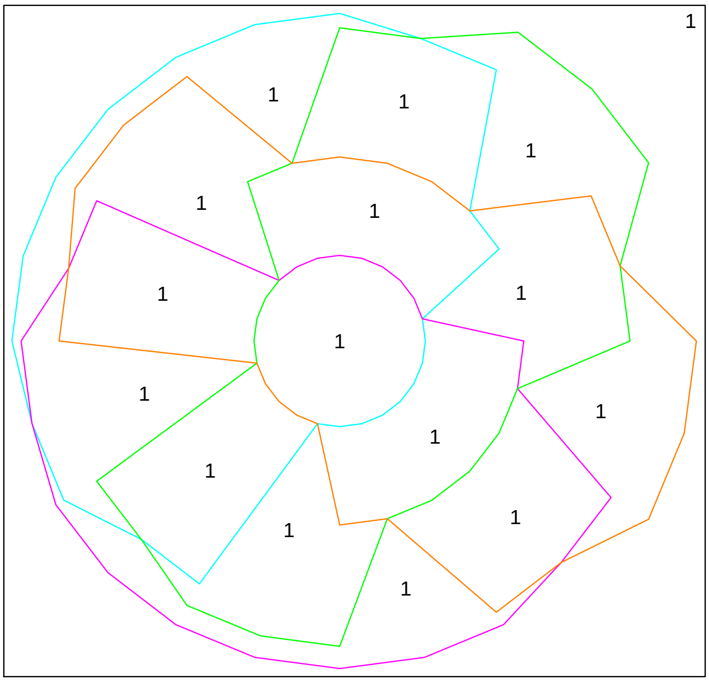
Figure 22: TODO Big weighted 3d Venn fails

12.2 4-set Euler diagrams

12.2.1 Chow-Ruskey diagrams

0000	1000	0100	1100
18.238486	2.000073	3.000240	2.000000
0010	1010	0110	1110
3.000000	0.000000	0.000000	1.000000
0001	1001	0101	1101
4.000000	0.000000	0.000000	0.000000
0011	1011	0111	1111
0.000000	0.000000	0.000000	1.000000




1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1

13 Error checking

These should fail

```
> print(try(Venn(NumberOfSets = 3, Weight = 1:7)))

[1] "Error in Venn(NumberOfSets = 3, Weight = 1:7) : \n  Weight length does not match numb
attr(,"class")
[1] "try-error"

> print(try(V3[1, ]))

[1] "Error in V3[1, ] : Can't subset on rows\n"
attr(,"class")
[1] "try-error"
```

Requesting a 2D plot for a 3D set produces a warning.
Empty objects work

```
NULL

NULL

character(0)
```

14 This document

Author	Jonathan Swinton
CVS id of this document	Id: Vennville.Rnw,v 1.6 2007/06/19 21:53:47 js229 Exp .
Generated on	19^{th} June, 2007
R version	R version 2.6.0 Under development (unstable) (2007-06-11 r41912)

References

[1] Stirling Chow and Frank Ruskey. Drawing area-proportional Venn and Euler diagrams. In Giuseppe Liotta, editor, *Graph Drawing*, volume 2912 of *Lecture Notes in Computer Science*, pages 466–477. Springer, 2003.

[2] Stirling Chow and Frank Ruskey. Towards a general solution to drawing area-proportional Euler diagrams. *Electr. Notes Theor. Comput. Sci.*, 134:3–18, 2005.