ashishsa_hw3_p1

We need to analyze the tumor micro array data from the Elements Of Statistical Learning. It has 14 subtypes of tumor cells. We need to use the SOM Algorithm to cluster the data into multiple groups.

```r
library(kohonen)
```

```
## Warning: package 'kohonen' was built under R version 3.6.3
```

```r
library(Umatrix)
```

```
## Warning: package 'Umatrix' was built under R version 3.6.3
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(ape)
```

```
## Warning: package 'ape' was built under R version 3.6.3
```

```r
library(dendextend)
```

```
## Warning: package 'dendextend' was built under R version 3.6.3
```

```
##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------
```

```
##
## Attaching package: 'dendextend'
```

```
## The following objects are masked from 'package:ape':
##
##     ladderize, rotate
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(RColorBrewer)
library(scales)
df1 <- read.csv('nci.data.csv',sep=",",row.names=1,header = TRUE)
```

We first scale the data.

```
EigenValues <- eigen(cov(scale(df1)))
EigenValues$values
```

```
##  [1] 8.0957119 5.3797629 3.5583361 2.7208690 2.5096325 2.0444217 1.8296886
##  [8] 1.7528499 1.6115437 1.4860941 1.3443270 1.3216904 1.2671661 1.2168416
## [15] 1.1300098 1.0475872 1.0273544 0.9962967 0.9699703 0.9361556 0.8693644
## [22] 0.8621069 0.8323595 0.8037197 0.7926028 0.7645869 0.7440932 0.7356460
## [29] 0.7003424 0.6805286 0.6690675 0.6592918 0.6417513 0.6163924 0.5965587
## [36] 0.5798647 0.5741100 0.5487758 0.5305889 0.5299946 0.5128732 0.4968256
## [43] 0.4916484 0.4805703 0.4577736 0.4529224 0.4387011 0.4264547 0.4084076
## [50] 0.3811480 0.3749040 0.3643935 0.3466927 0.3349110 0.3211923 0.3018903
## [57] 0.2624658 0.2333543 0.2058479 0.1760215 0.1611637 0.1452791 0.1270222
## [64] 0.1194815
```

```
m <- apply(df1,2,mean)
s <- apply(df1,2,sd)
z <- scale(df1,m,s)
```

Now we turn the data into a matrix so that we can pass the data to the SOM Algorithm.

```
training_data <- as.matrix(z)
```

We need to apply SOM to the data and we know that the size of SOM is given by 5*sqrt(N) Here N is the number of observation. So to break it down, If Columns are your feature and each row is one observation then Number of Nodes = 5* sqrt (# of Rows * # of columns).

We observe that there are 6830 Rows and 64 Columns. So the Number Of Observations can be summed up as:

```
nr <- nrow(z)
nc <- ncol(z)
s_lower <- 0.25*sqrt((nr*nc))
s_lower
```

```
## [1] 165.2876
```

```
s_upper <- 4*sqrt((nr*nc))
s_upper
```

```
## [1] 2644.602
```

The counts lot gives us the count of the number of states map into the different units.

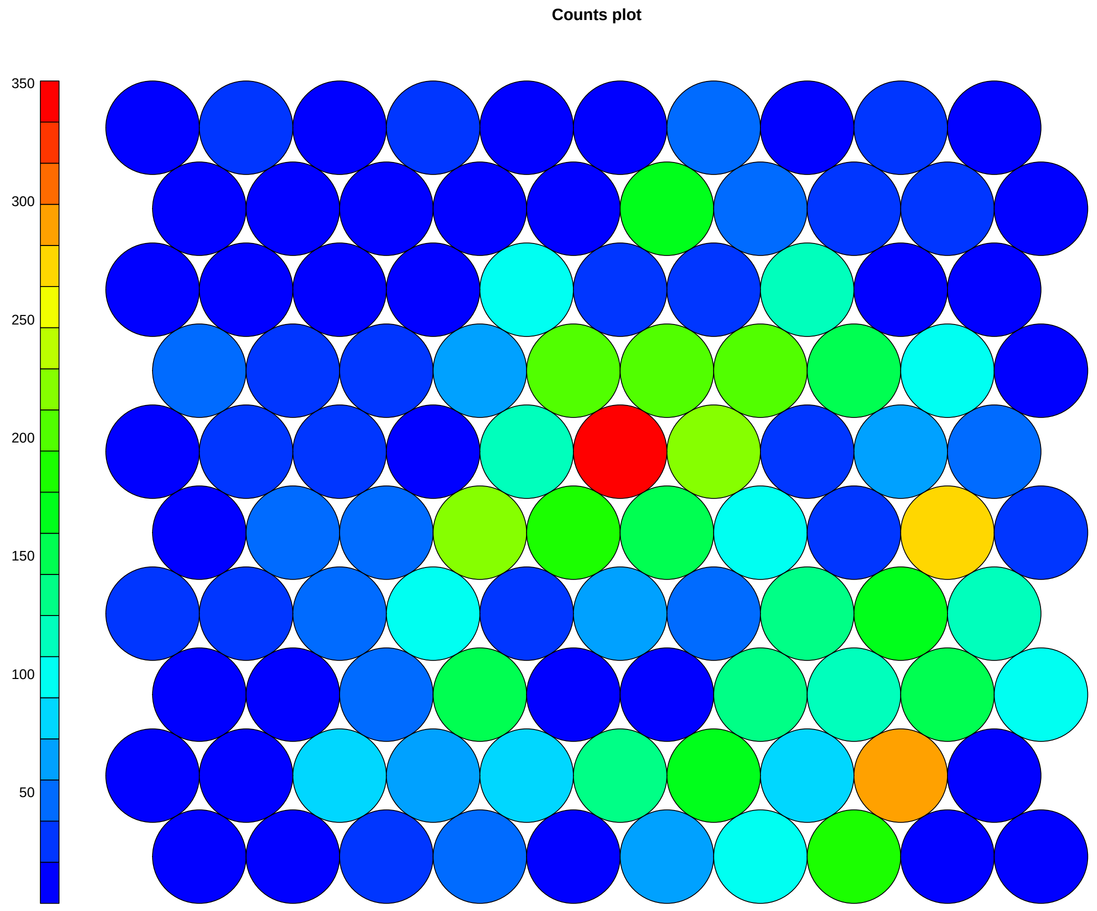
```
coolBlueHotRed <- function(n,alpha=1){rainbow(n,end=4/6,alpha=alpha)[n:1]}
```

```
data_train_matrix <- training_data
```

```
som_grid <- somgrid(xdim = 10, ydim = 10, topo="hexagonal")
som_model <- som(data_train_matrix,grid=som_grid,rlen=1000)
```

The counts lot gives us the count of the number of states map into the different units.

```
plot(som_model,type="counts",palette.name=coolBlueHotRed)
```

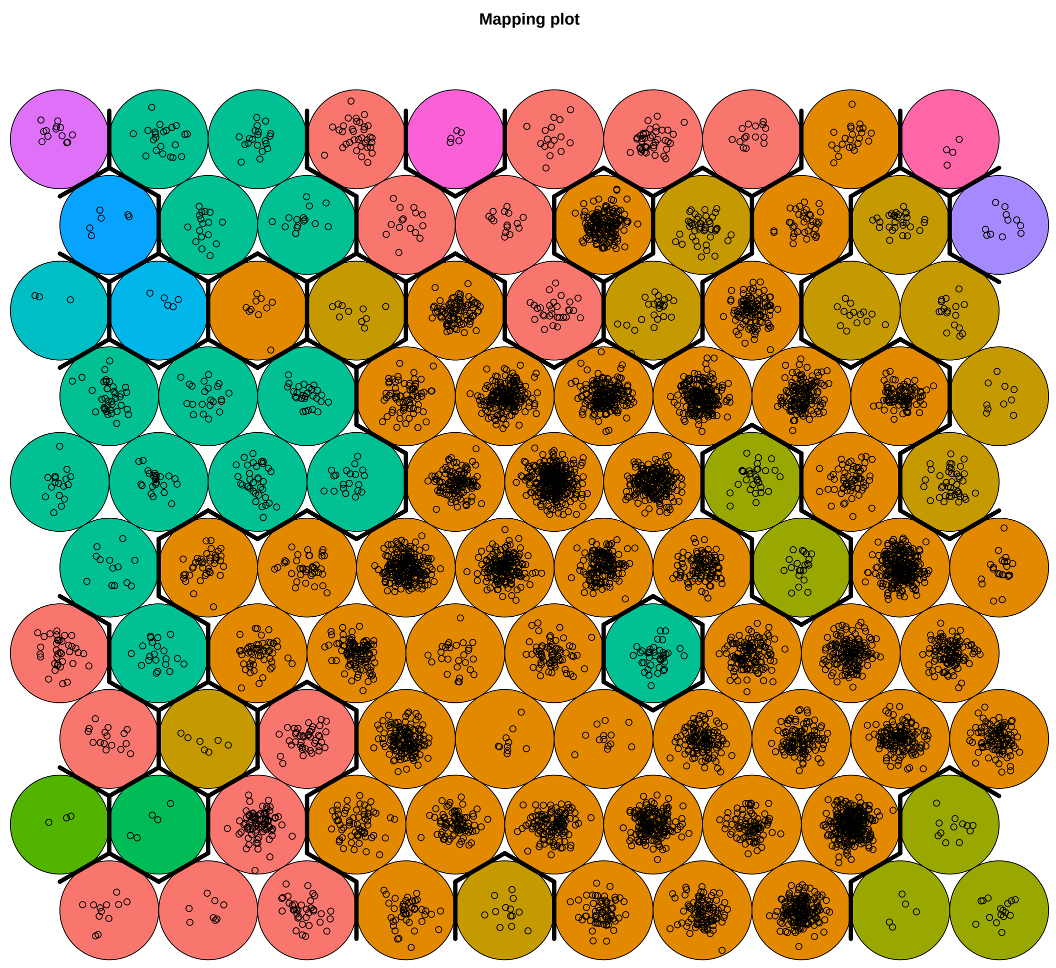


We now plot the SOM amd specify the cutoff point at a location where we can observe the approximate cutoff in the Dendrogram plotted above.

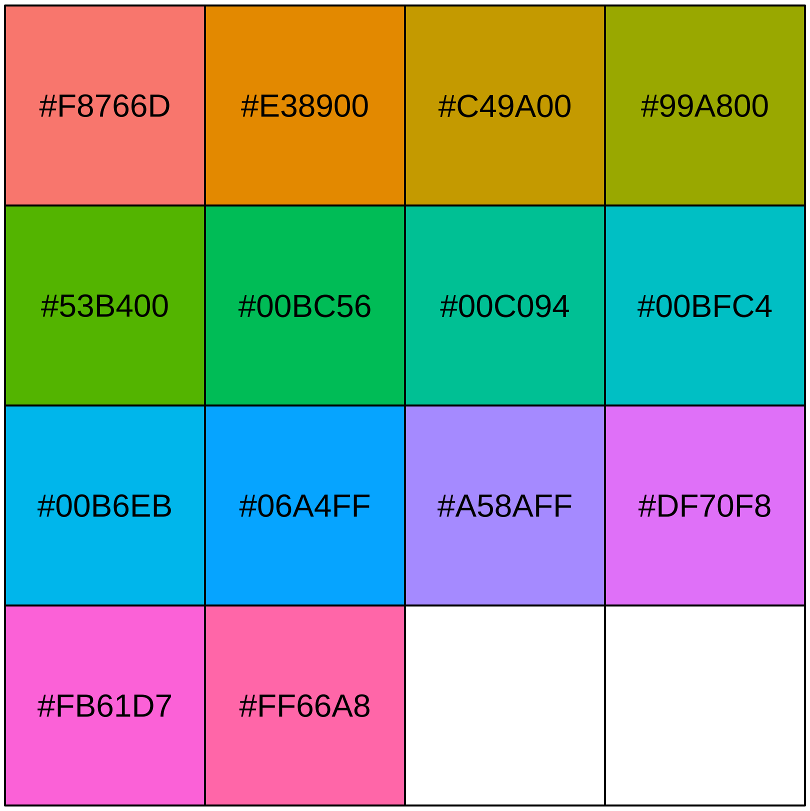
By specifying the cutoff point obtained from the plot above and clustering the data at that height gives us the approximate distribution of the data into 3 clusters.

```
codes <- som_model$codes[[1]]
d <- dist(codes)
hc <- hclust(d)
som_cluster <- cutree(hc,k=14)
# plot these results:
my_pal <- (hue_pal()(14))
my_bhcol <- my_pal[som_cluster]
```

```
{plot(som_model,type="mapping",col="black",bgcol = my_bhcol)
  add.cluster.boundaries(som_model,som_cluster)}
```



```
show_col(my_pal)
```

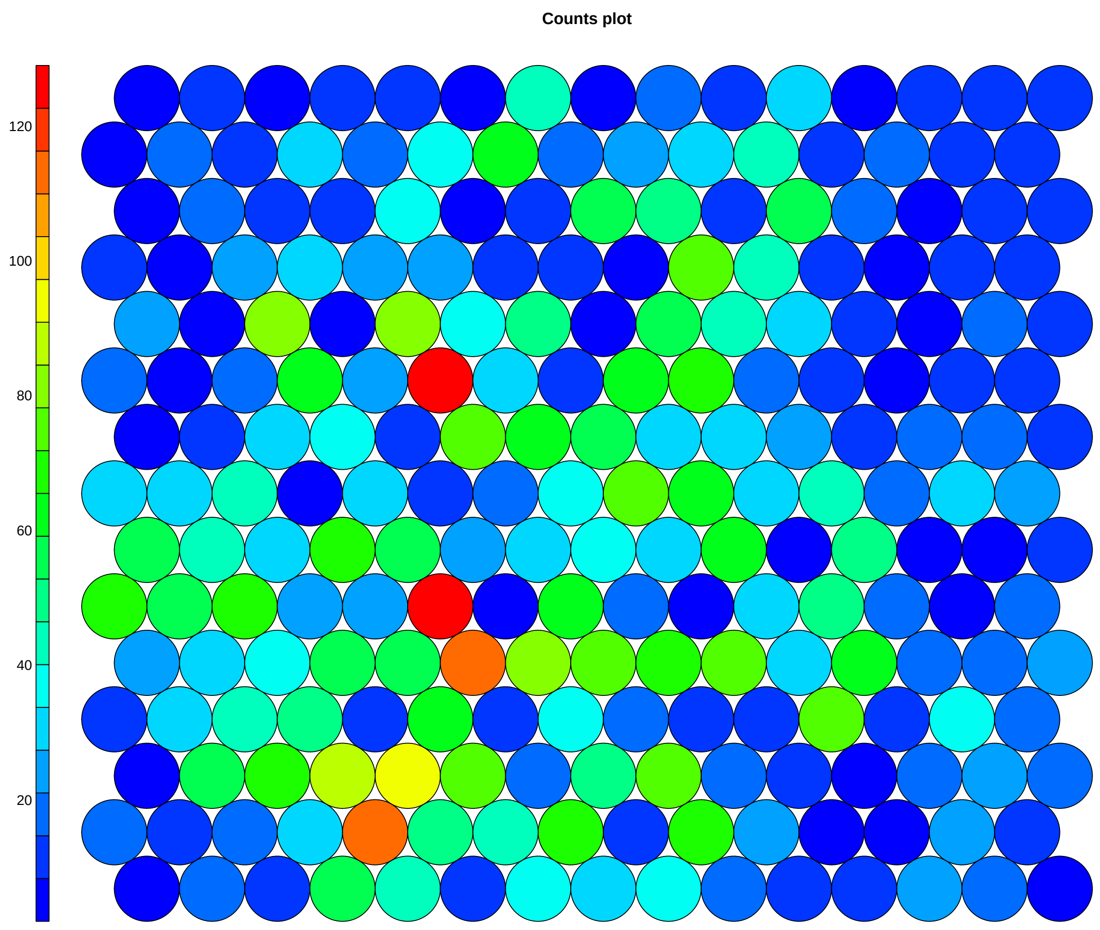


From the counts plot we had inferred that there are upwards of 300 Genes that are plotted to a given node in SOM in that it can cause overfitting of the data. Also mapping too many nodes potentially results in a loss of information as we are mapping too many objects of different types to the same dimension space. So we increase the grid size to see the change in the data and cluster. Most of the Nodes have upwards of 50 Genes mapped to it which creates a lot of generalization of the difference in the clusters.

```
som_grid <- somgrid(xdim = 15, ydim = 15, topo="hexagonal")
som_model <- som(data_train_matrix,grid=som_grid,rlen=1000)
```

The counts lot gives us the count of the number of states map into the different units.

```
plot(som_model,type="counts",palette.name=coolBlueHotRed)
```



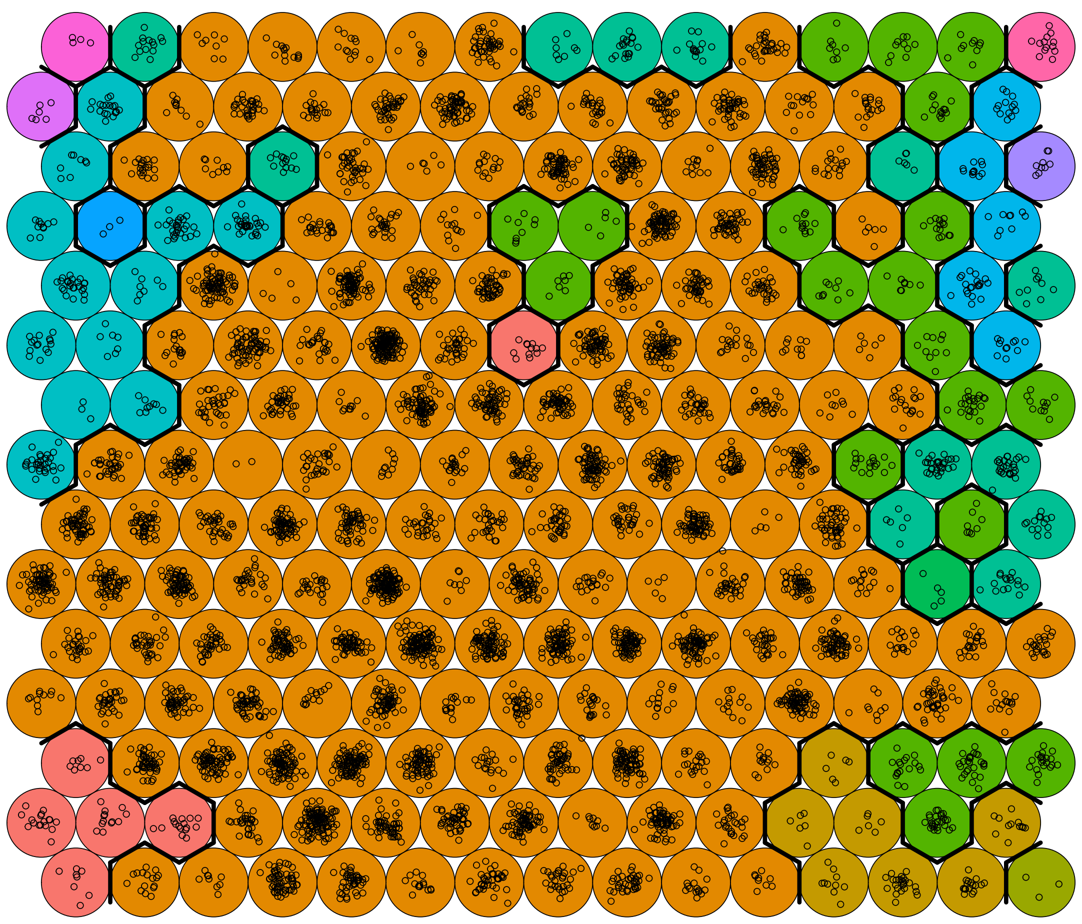
We now plot the SOM amd specify the cutoff point at a location where we can observe the approximate cutoff in the Dendrogram plotted above.

By specifying the cutoff point obtained from the plot above and clustering the data at that height gives us the approximate distribution of the data into 3 clusters.

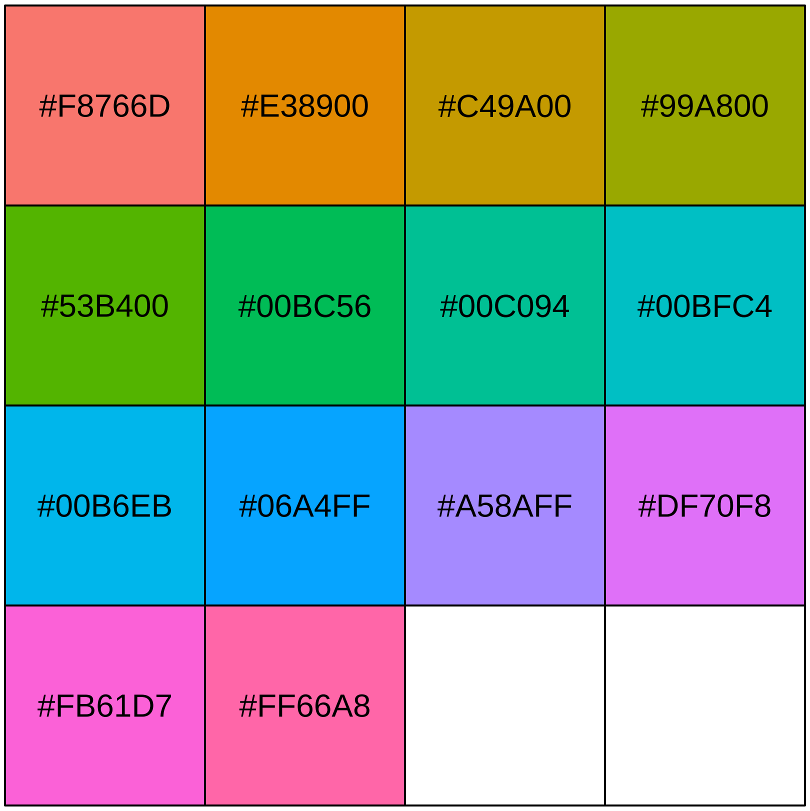
```
codes <- som_model$codes[[1]]
d <- dist(codes)
hc <- hclust(d)
som_cluster <- cutree(hc,k=14)
# plot these results:
my_pal <- (hue_pal()(14))
my_bhcol <- my_pal[som_cluster]

{plot(som_model,type="mapping",col="black",bgcol = my_bhcol)
add.cluster.boundaries(som_model,som_cluster)}
```

Mapping plot



```
show_col(my_pal)
```

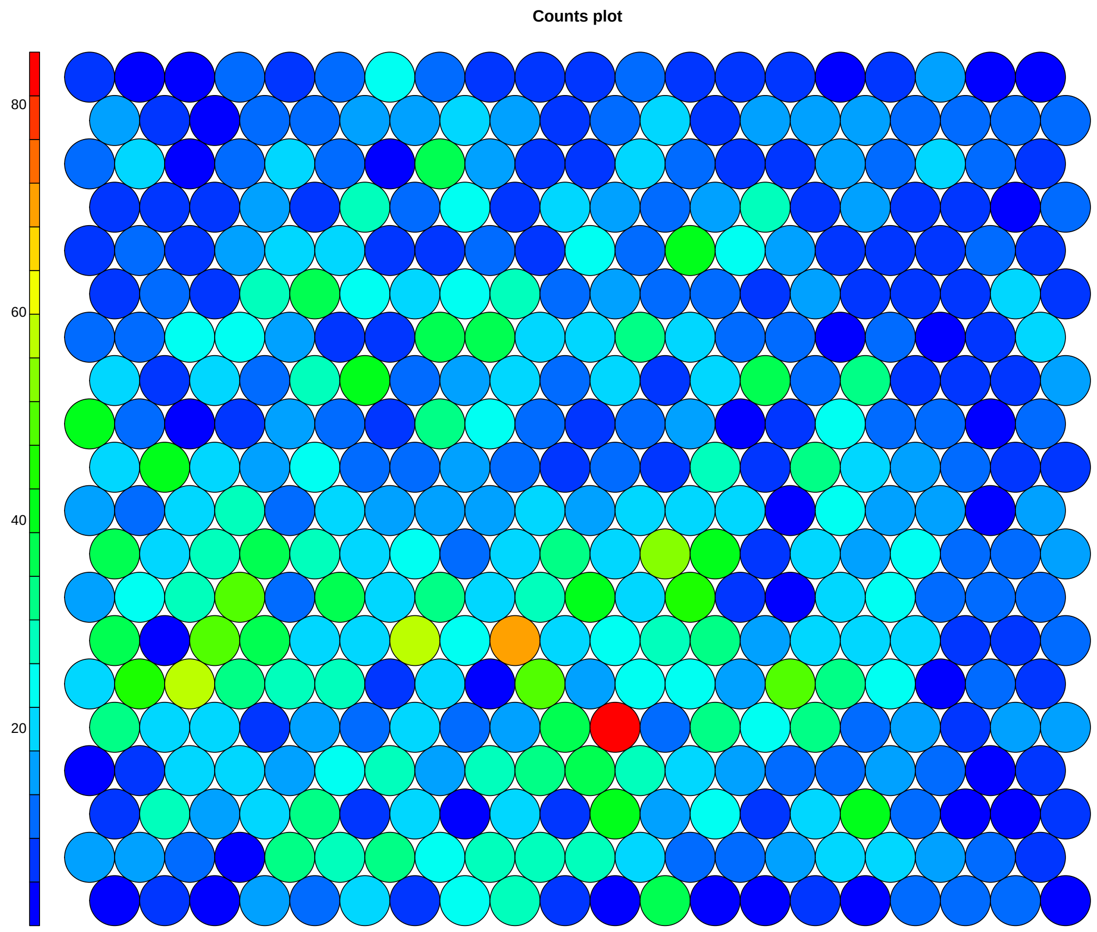


From the counts plot we had inferred that there are upwards of 100 Genes that are plotted to a given node in SOM in that it can cause overfitting of the data. Also mapping too many nodes potentially results in a loss of information as we are mapping too many objects of different types to the same dimension space. Although the results improve a lot than the last grid size and the number of genes mapped have reduced. So we increase the grid size to see the change in the data and cluster. Most of the Nodes have upwards of 80 Genes mapped to it which creates a higher degree of generalization of the difference in the clusters.

```r
som_grid <- somgrid(xdim = 20, ydim=20, topo="hexagonal")
som_model <- som(data_train_matrix,grid=som_grid,rlen=1000)
```

The counts lot gives us the count of the number of states map into the different units.

```r
plot(som_model,type="counts",palette.name=coolBlueHotRed)
```



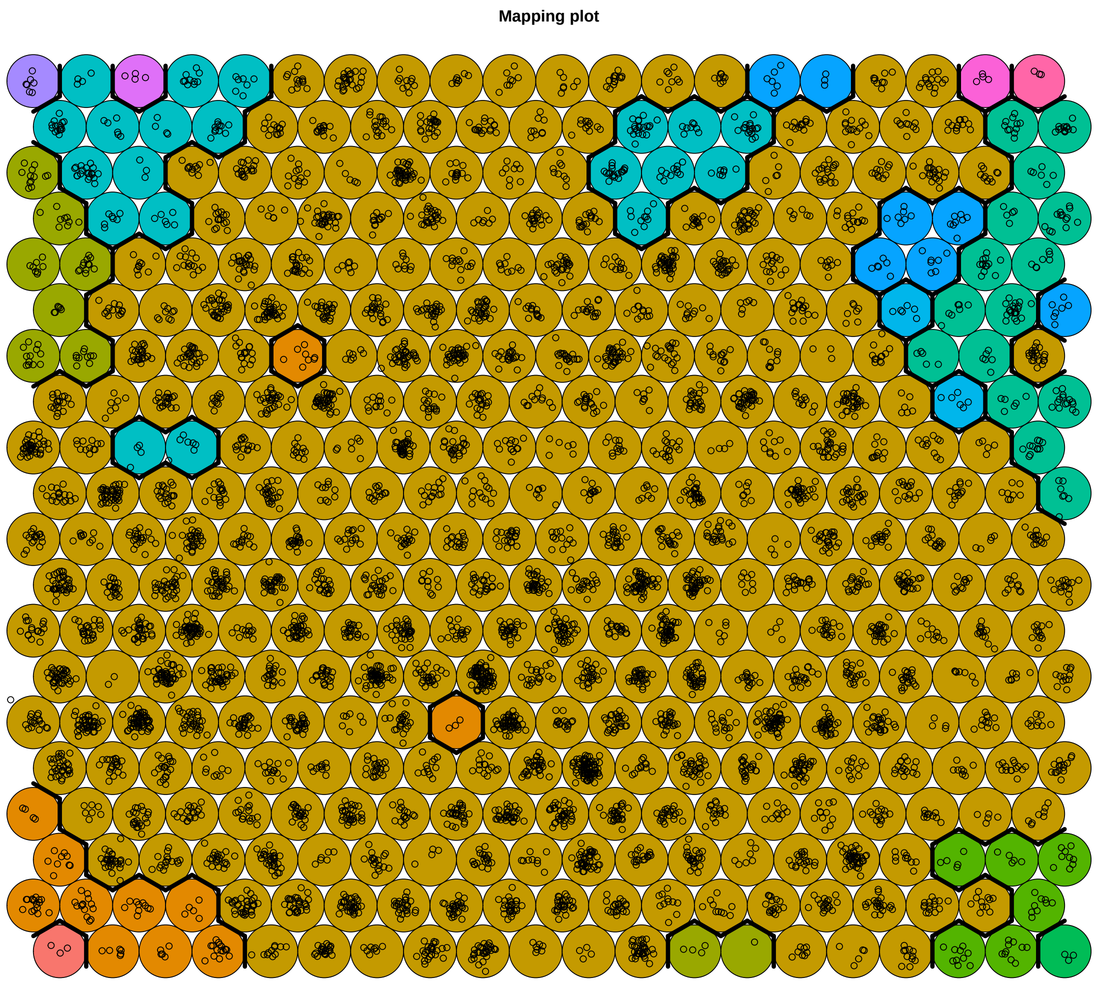
We now plot the SOM amd specify the cutoff point at a location where we can observe the approximate cutoff in the Dendrogram plotted above.

By specifying the cutoff point obtained from the plot above and clustering the data at that height gives us the approximate distribution of the data into 3 clusters.

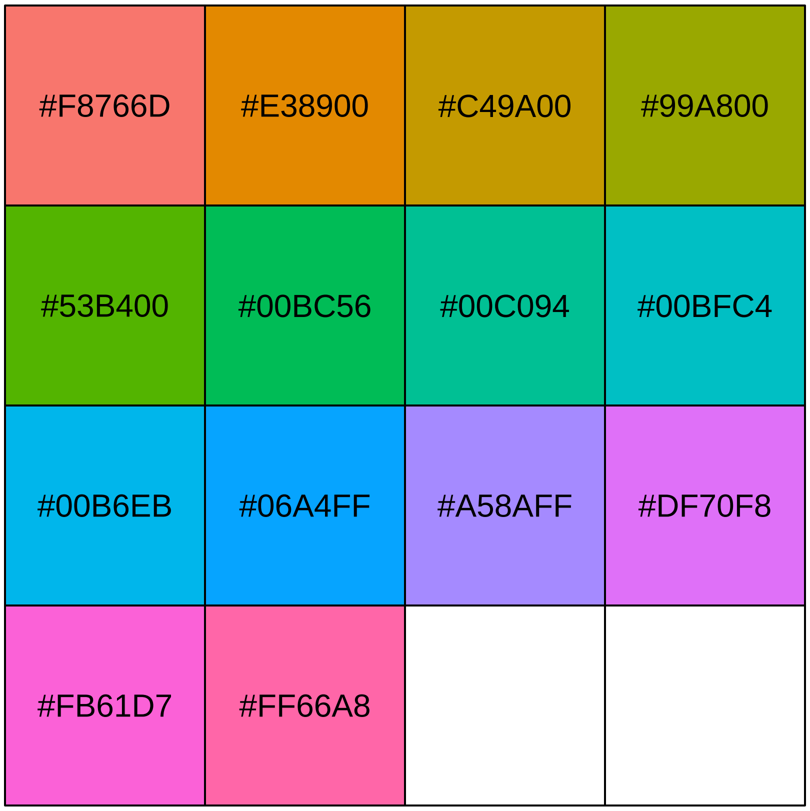
```r
codes <- som_model$codes[[1]]
d <- dist(codes)
hc <- hclust(d)
som_cluster <- cutree(hc,k=14)
# plot these results:
my_pal <- (hue_pal()(14))
my_bhcol <- my_pal[som_cluster]

{plot(som_model,type="mapping",col="black",bgcol = my_bhcol)
```

```
  add.cluster.boundaries(som_model,som_cluster)}
```



```
show_col(my_pal)
```

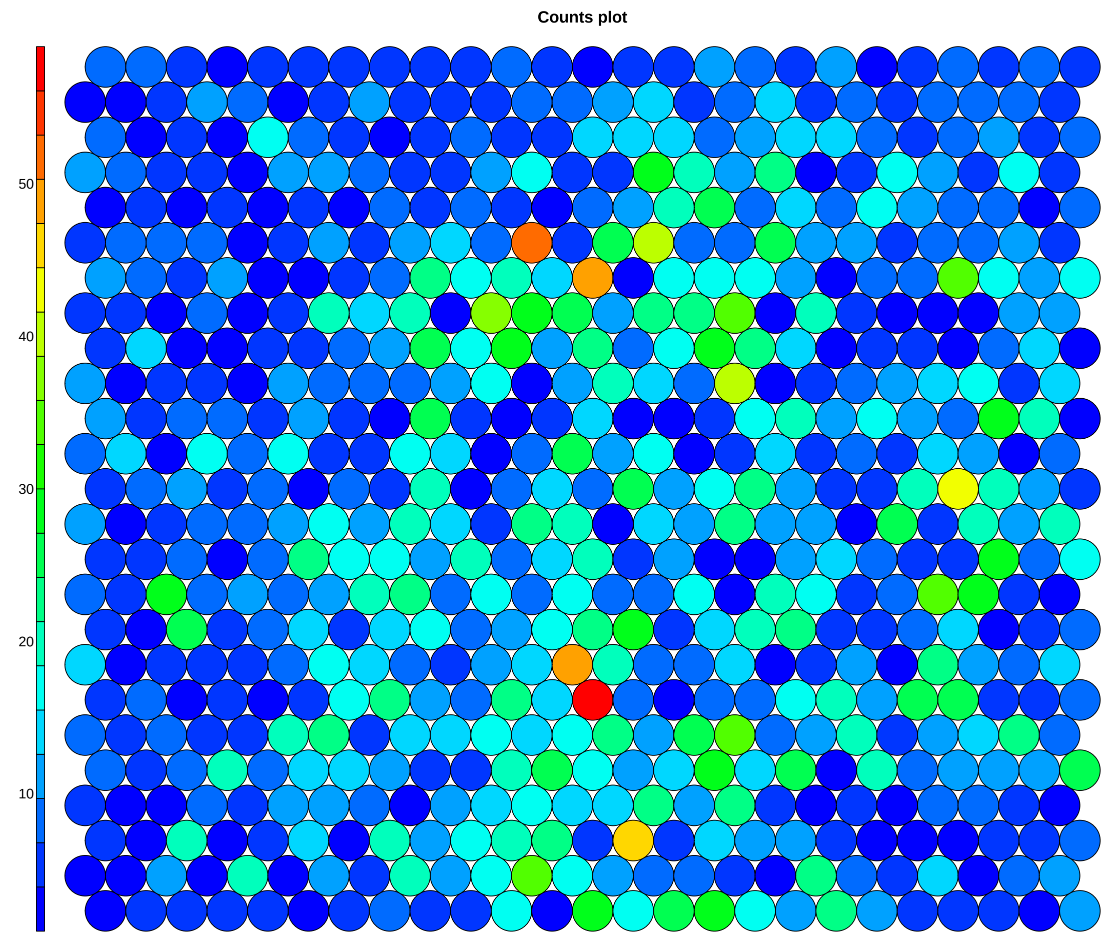


From the counts plot we had inferred that there are upwards of 70 Genes that are plotted to a given node in SOM in that it can cause overfitting of the data. Also mapping too many nodes potentially results in a loss of information as we are mapping too many objects of different types to the same dimension space. So we increase the grid size to see the change in the data and cluster. Most of the Nodes have upwards of 50 Genes mapped to it which creates a lot of generalization of the difference in the clusters.

```
som_grid <- somgrid(xdim = 25, ydim=25, topo="hexagonal")
som_model <- som(data_train_matrix,grid=som_grid,rlen=10000)
```

The counts lot gives us the count of the number of states map into the different units.

```
plot(som_model,type="counts",palette.name=coolBlueHotRed)
```

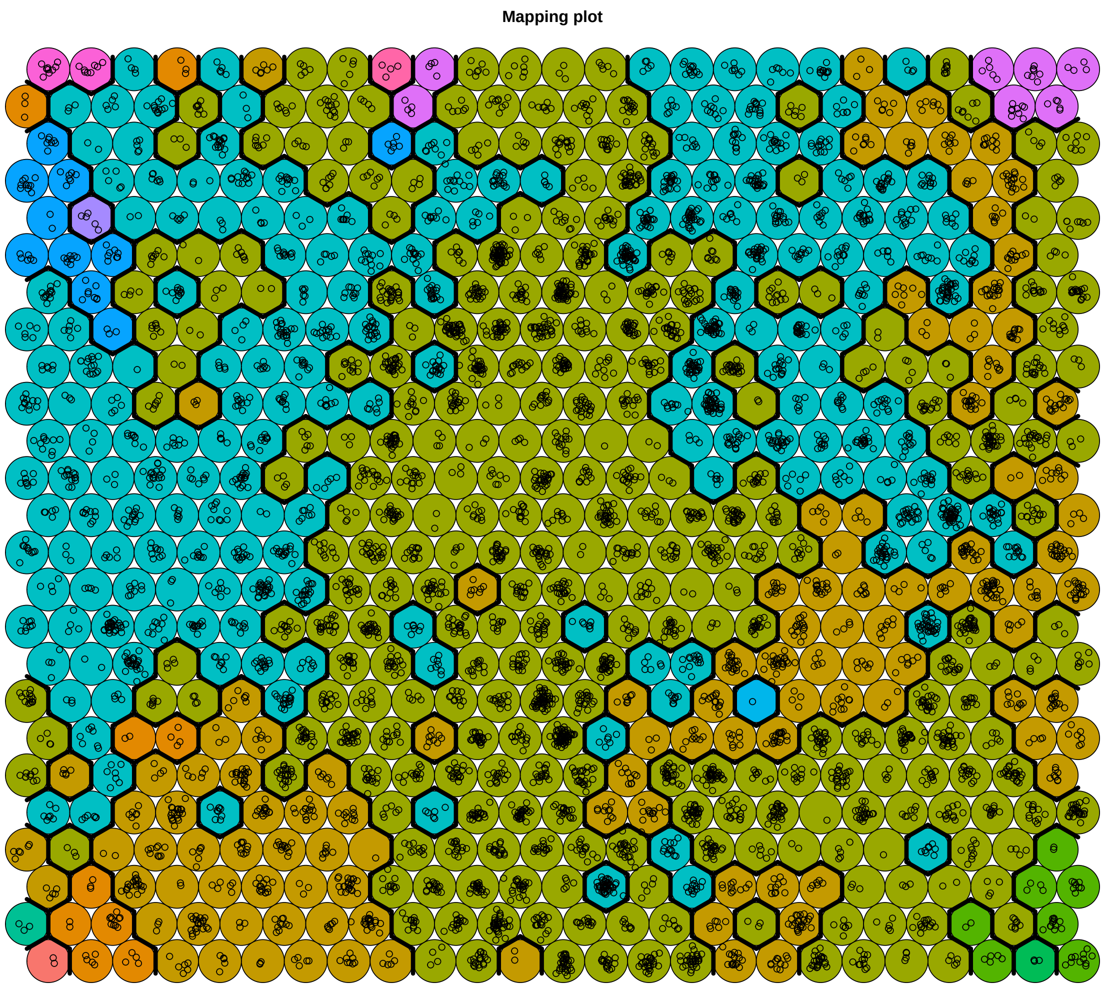


We now plot the SOM amd specify the cutoff point at a location where we can observe the approximate cutoff in the Dendrogram plotted above.

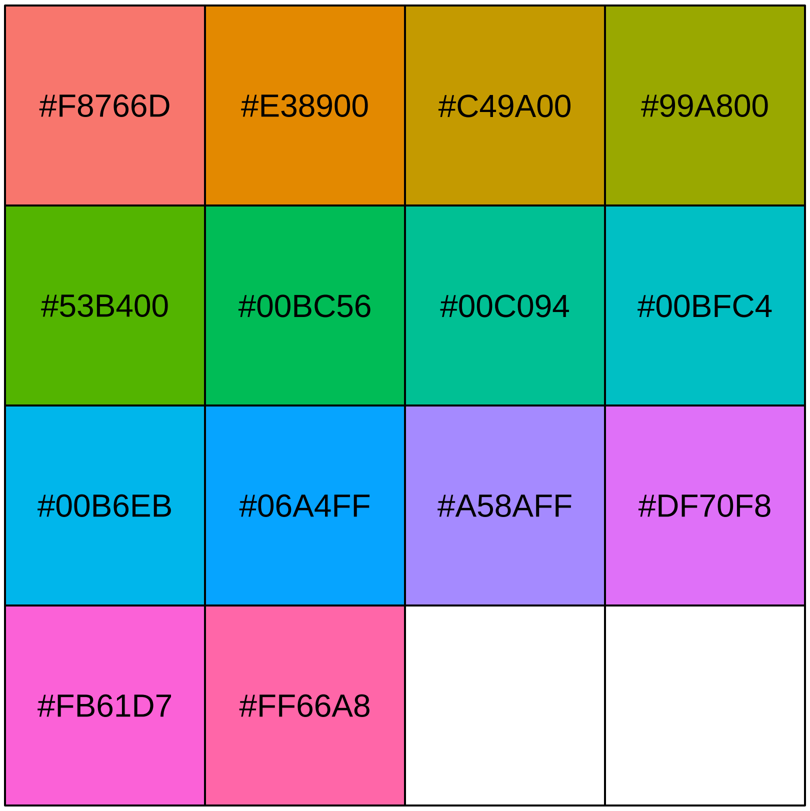
By specifying the cutoff point obtained from the plot above and clustering the data at that height gives us the approximate distribution of the data into 3 clusters.

```
codes <- som_model$codes[[1]]
d <- dist(codes)
hc <- hclust(d)
som_cluster <- cutree(hc,k=14)
# plot these results:
my_pal <- (hue_pal()(14))
my_bhcol <- my_pal[som_cluster]

{plot(som_model,type="mapping",col="black",bgcol = my_bhcol)
add.cluster.boundaries(som_model,som_cluster)}
```



```
show_col(my_pal)
```

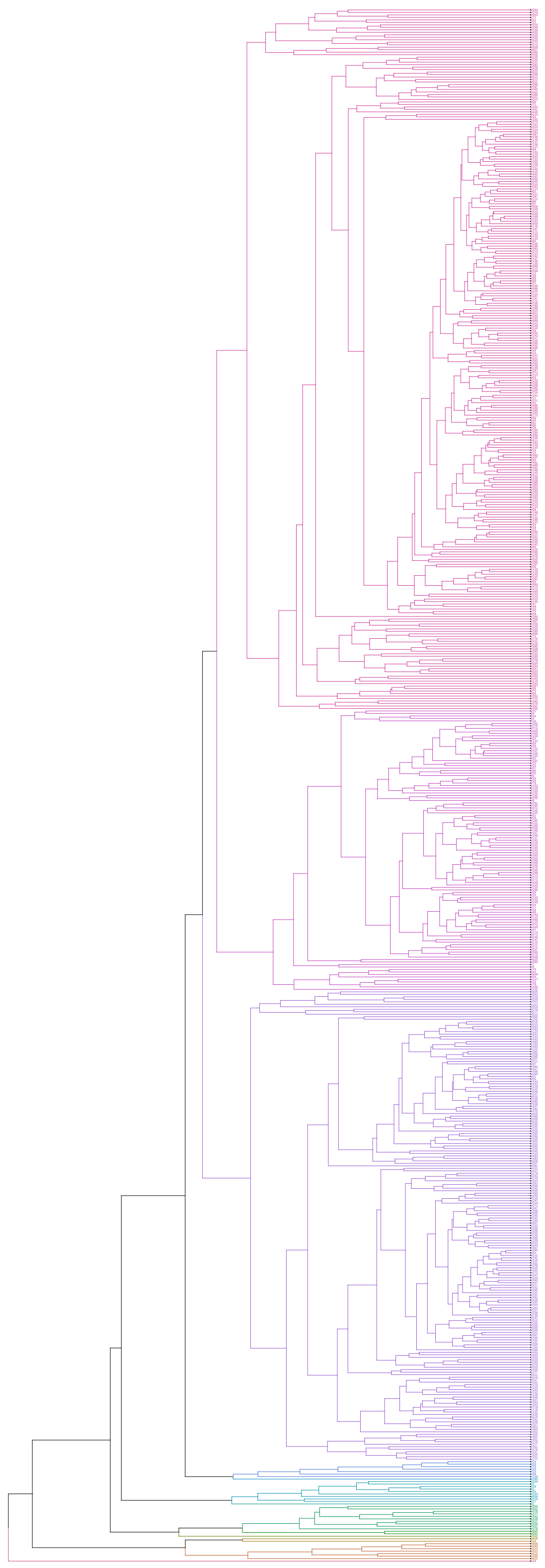


From the counts plot we had inferred that there are upwards of 50 Genes that are plotted to a given node in SOM.MOst of the nodes have 15-20 genes mapped to it which is a decent size for the cluster and also this is not a lot of nodes that could potentially over-generalize the results of the data.

We now visualize the data that is mapped using SOM into a Heirarchical Cluster Map. We find the distance using the SOM_Model codes and observe a clear distinction between the height of the dendrogram between the 2 clusters. This helps us find the point for the cutoff and also helps us easily cluster the dat into 3 distinct groups which was difficult in the case of Plain Heirarchical Clustering using Eucledian Distance.

```
dend <- codes %>% dist %>%
  hclust %>% as.dendrogram %>%
  set("branches_k_color", k = 14) %>% set("branches_lwd", 0.7) %>%
  set("labels_cex", 0.6) %>% set("labels_colors", k = 14) %>%
  set("leaves_pch", 19) %>% set("leaves_cex", 0.5)

ggd1 <- as.ggdend(dend)
ggplot(ggd1, horiz = TRUE)
```



This gives us the summary of the Self Organization Map.

```
summary(som_model)
```

```
## SOM of size 25x25 with a hexagonal topology and a bubble neighbourhood function.
## The number of data layers is 1.
## Distance measure(s) used: sumofsquares.
## Training data included: 6830 objects.
## Mean distance to the closest unit in the map: 27.877.
```

We now analyze the list of all the states and the prototypes that are closest to them.

```
head(som_model$unit.classif)
```

```
## [1] 472  61  64 267 541 247
```

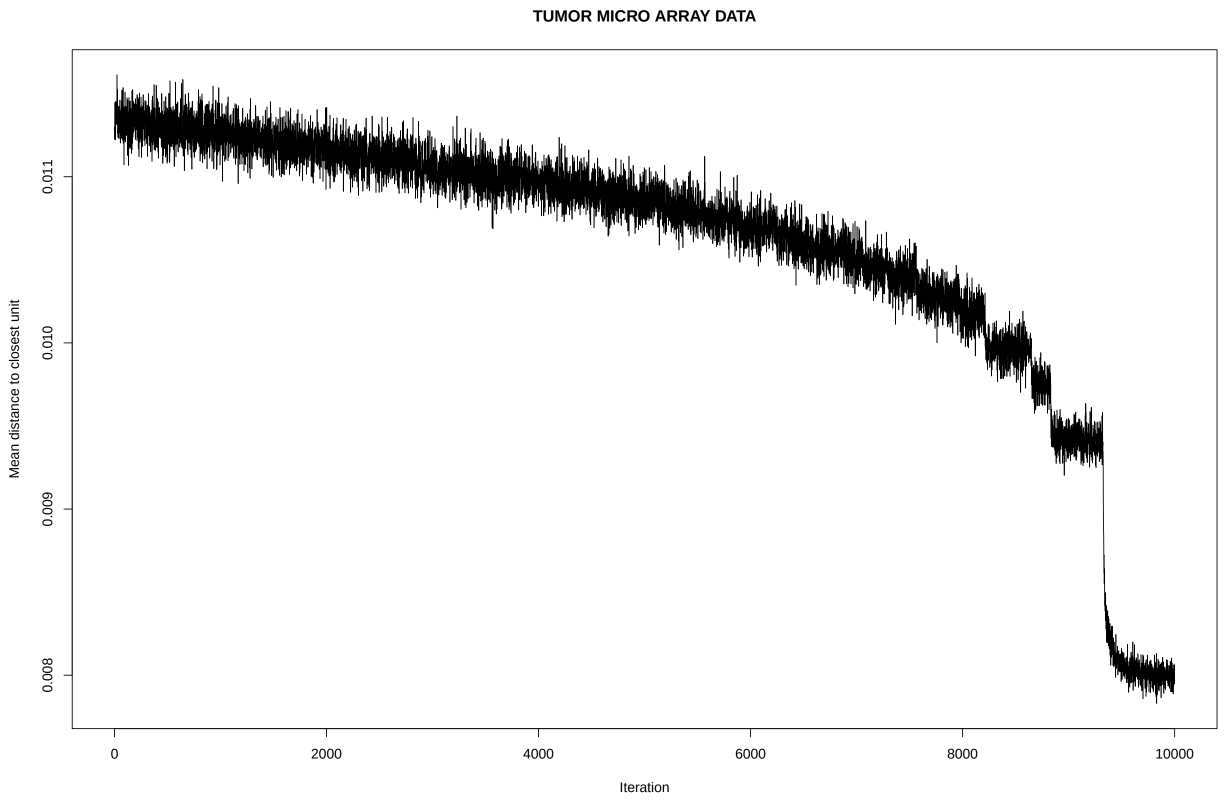
We now analyze the relative changes of the prototype code book vectors as we proceed through 200 different iteration.

```
head(som_model$changes)
```

```
##             [,1]
## [1,] 0.01130158
## [2,] 0.01122451
## [3,] 0.01144502
## [4,] 0.01129193
## [5,] 0.01140451
## [6,] 0.01123508
```

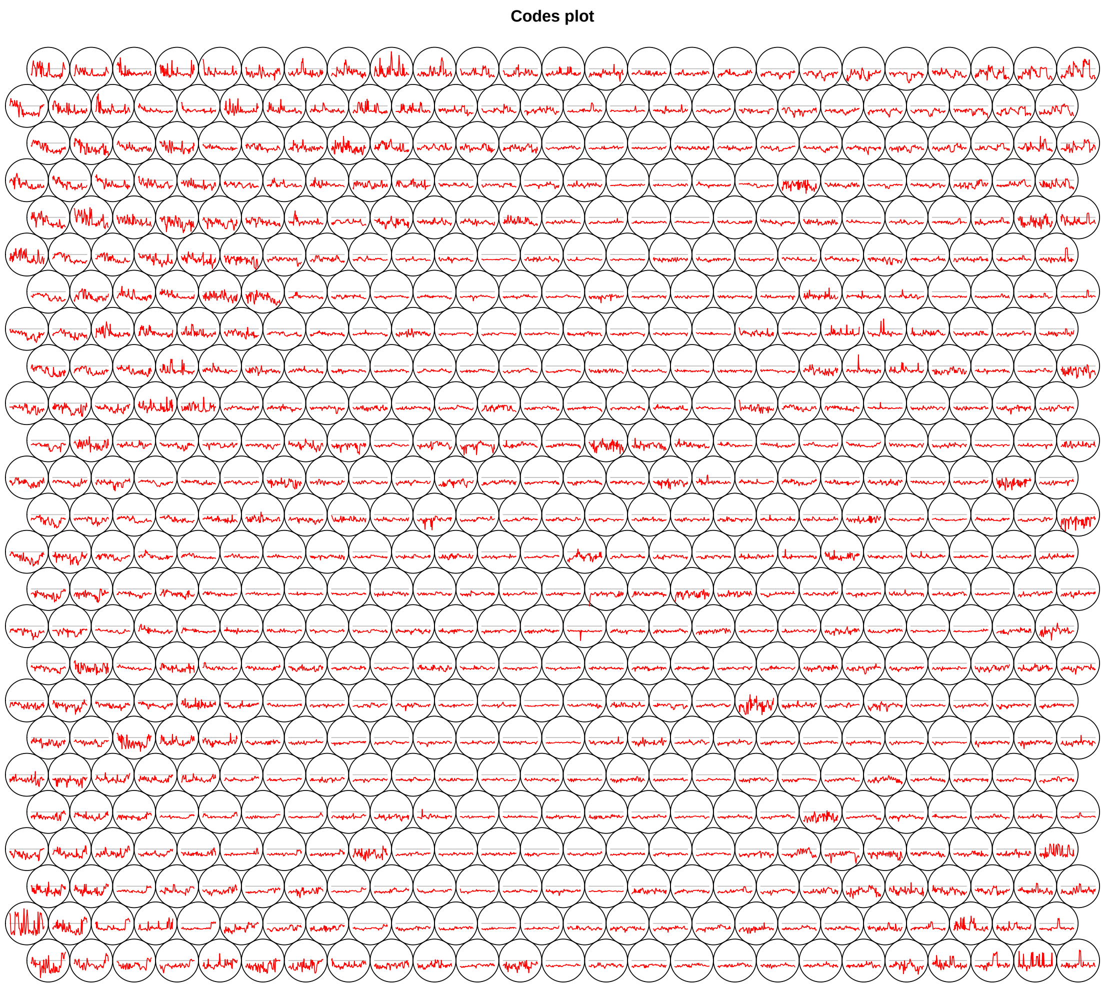
We observe that the changes start decreasing as we start increasing the number of iterations. A better understanding can be obtained from the Graph which represents the changes in the prototype. The things start converging as we move in the iteration.

```
plot(som_model,type="changes",main="TUMOR MICRO ARRAY DATA")
```



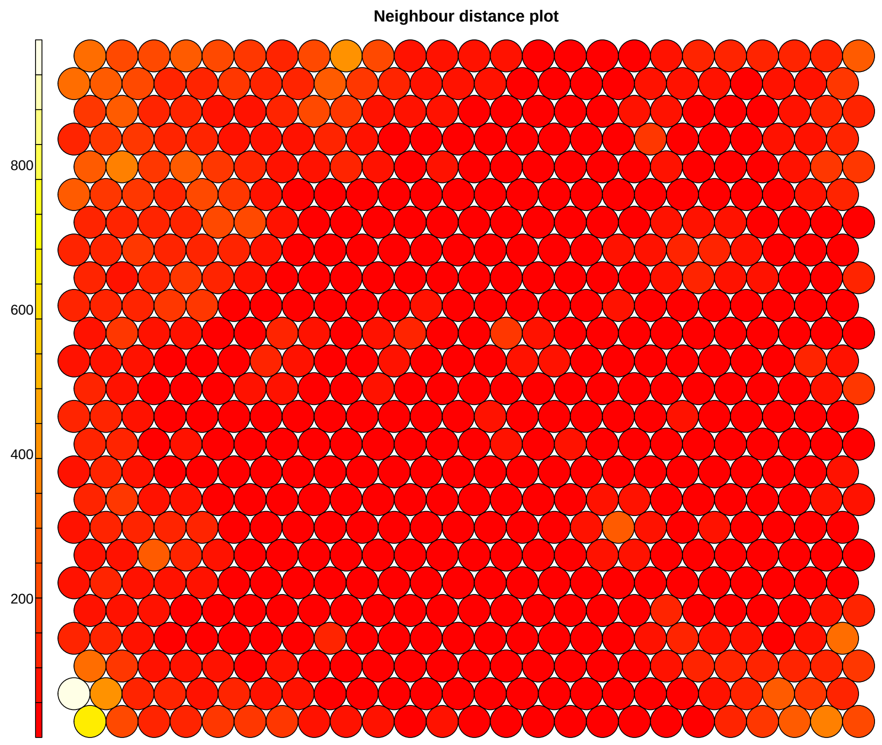
We now plot the Codes plot which gives us the distribution of various crimes across various nodes.

```
plot(som_model,type = "codes")
```



We now plot the distance to the Neighbours between each states that are plotted into a particular node.

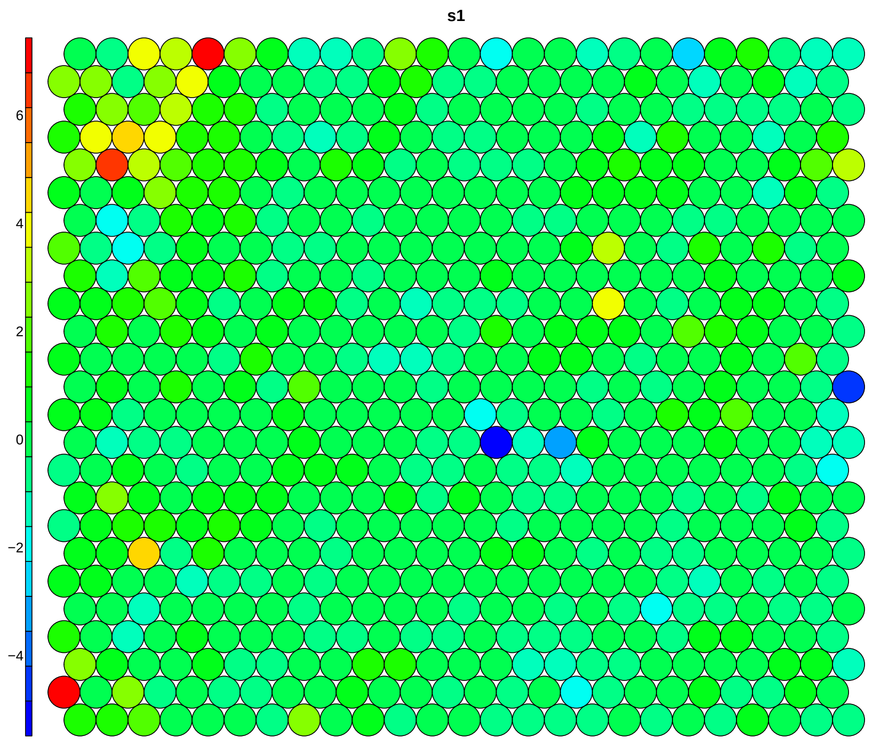
```
plot(som_model,type="dist.neighbours")
```

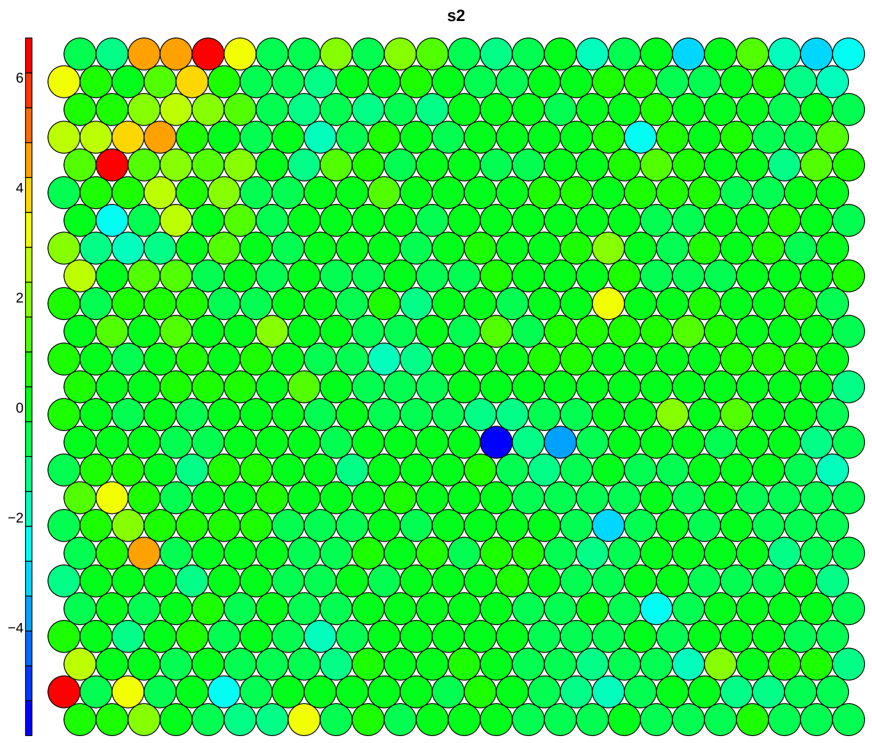


We now plot the componenet plane plots. Which Visualizes the original data over the SOM that we have created.

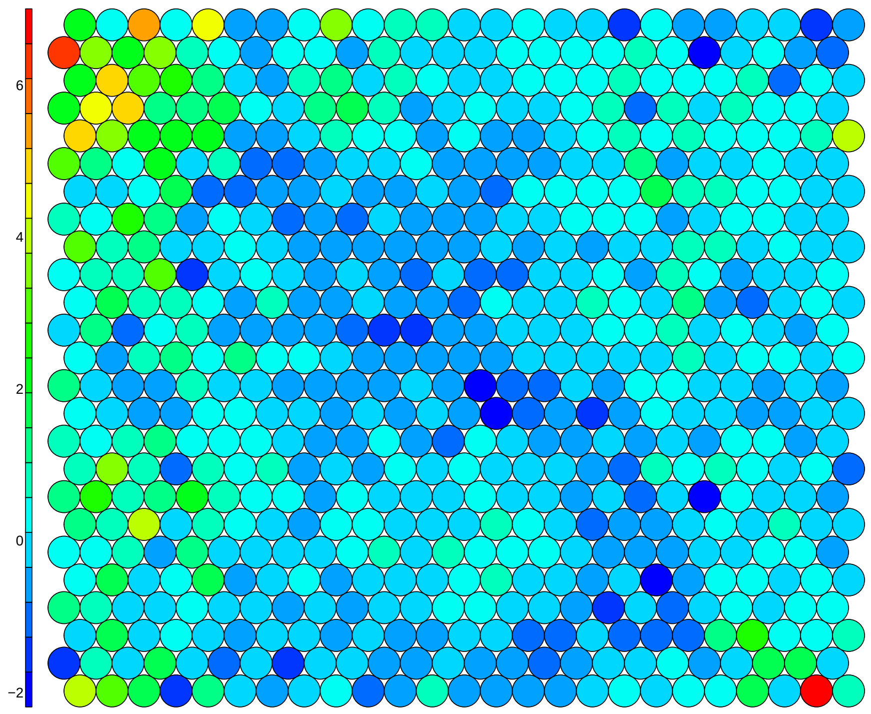
```
som_model$data <- data.frame(som_model$data)

{for (i in 1:50){
  plot(som_model, type = "property", property = codes[,i], main=names(som_model$data)[i],palette.name=c
  }}
```

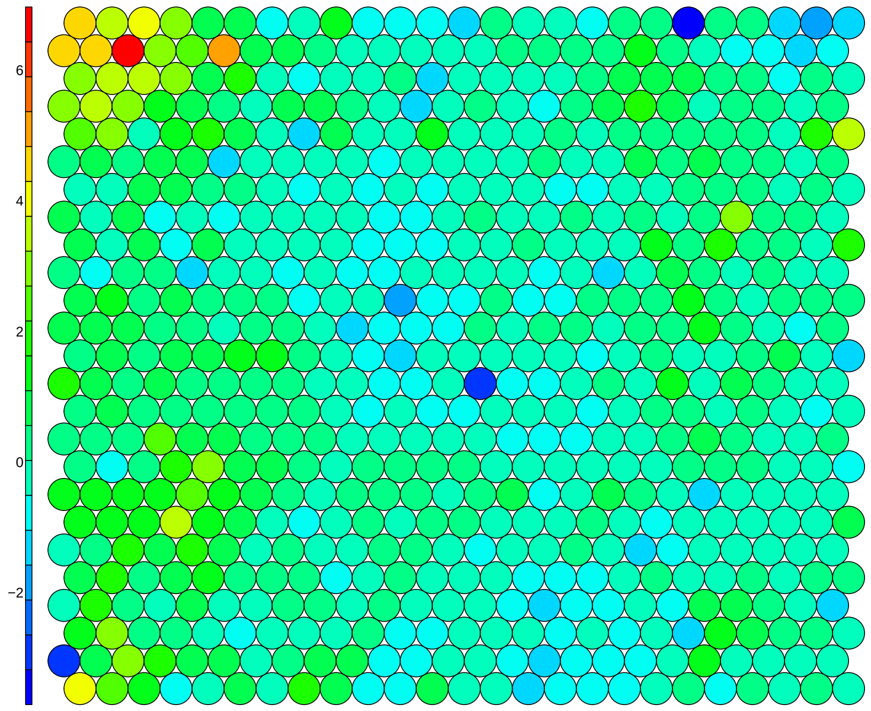

s1
6
4
2
0
−2
−4

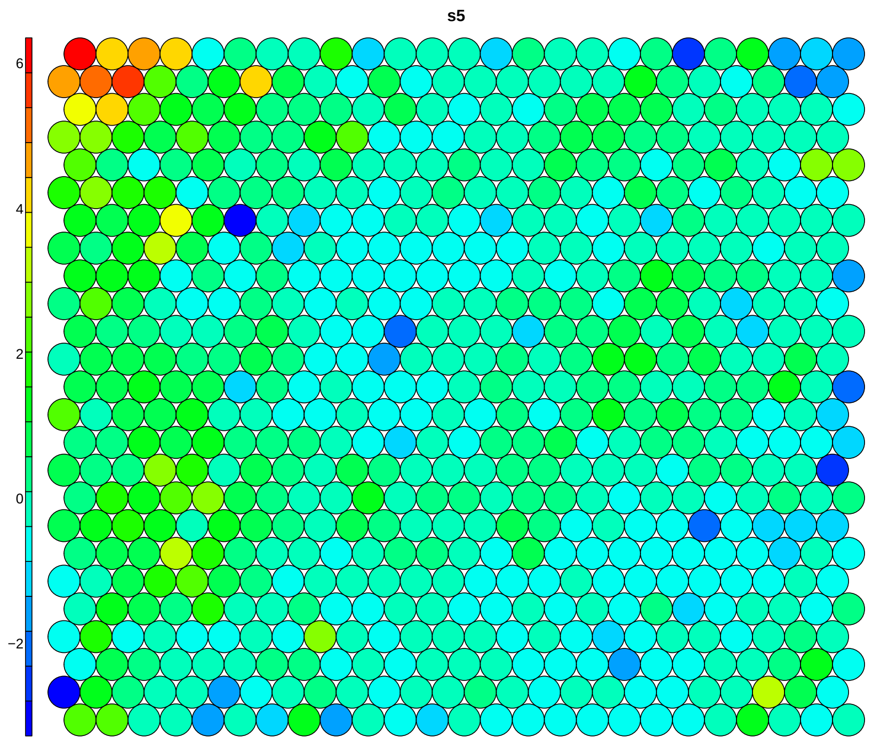

s2
6
4
2
0
−2
−4

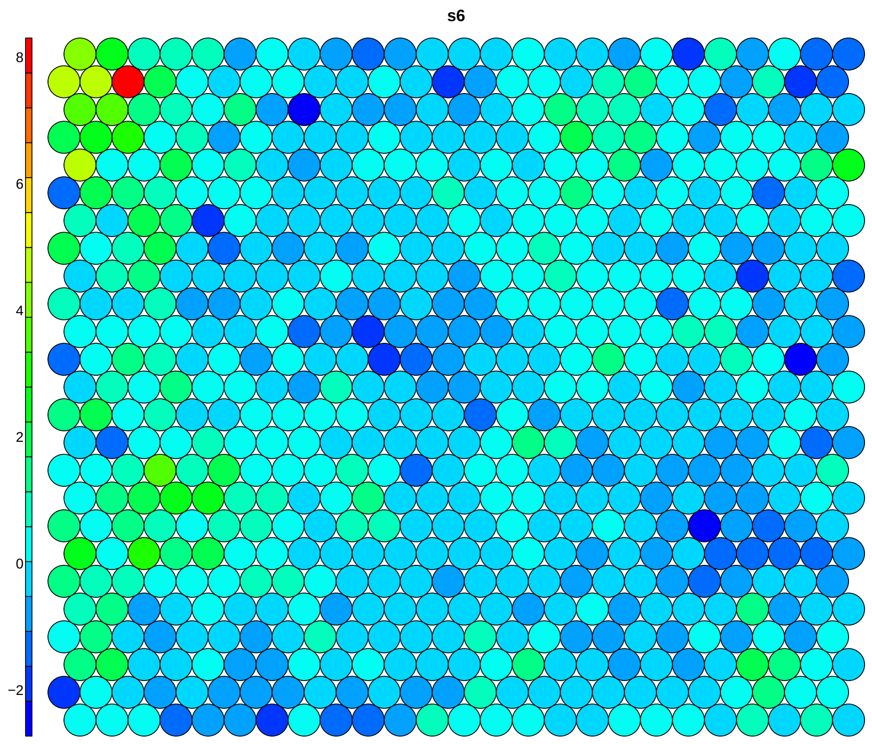
s3



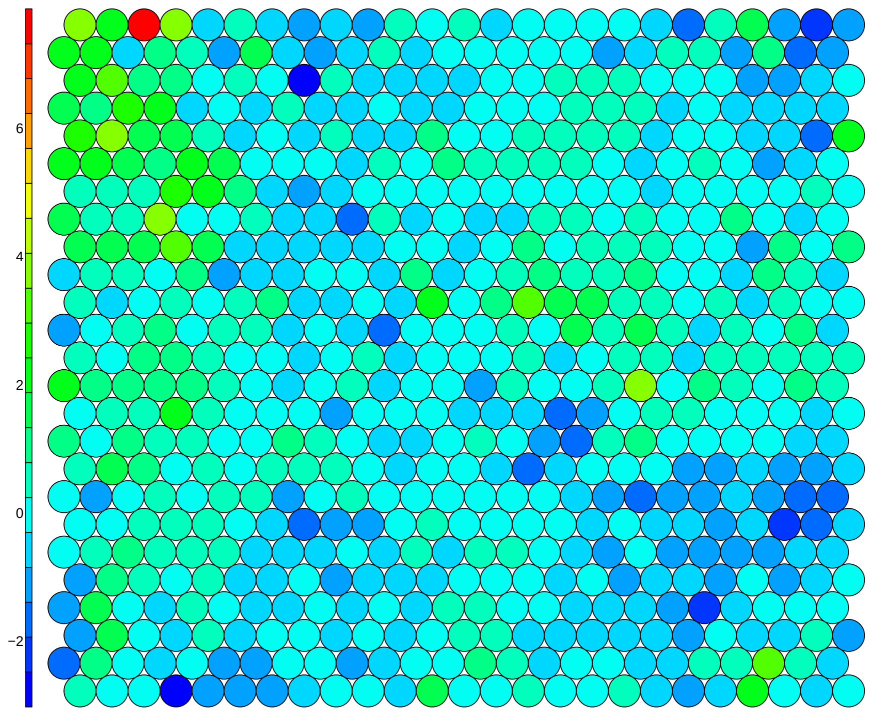
s4



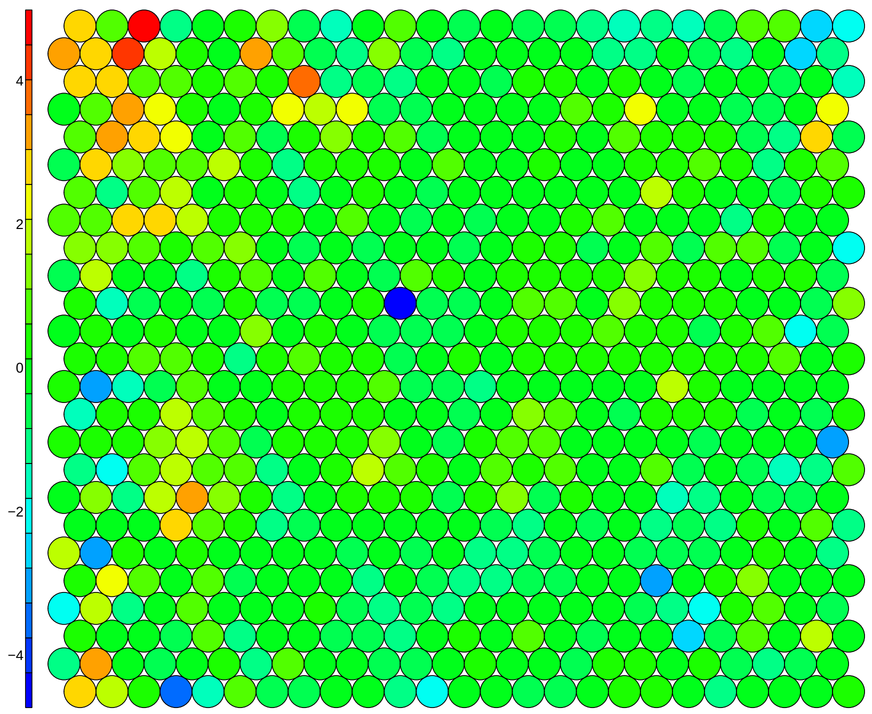

s5
6
4
2
0
−2


s6
8
6
4
2
0
−2

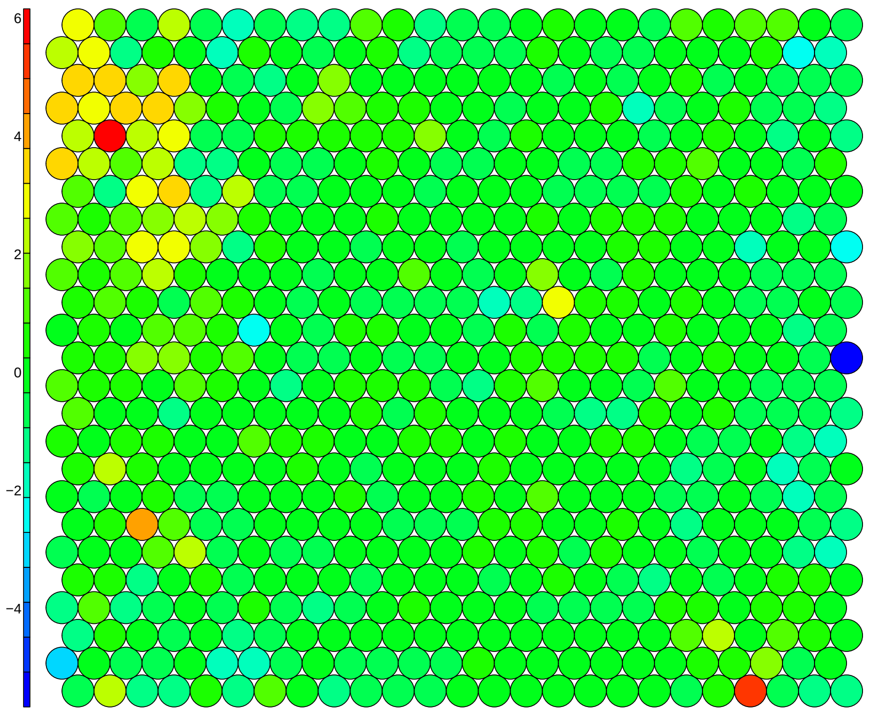
s7



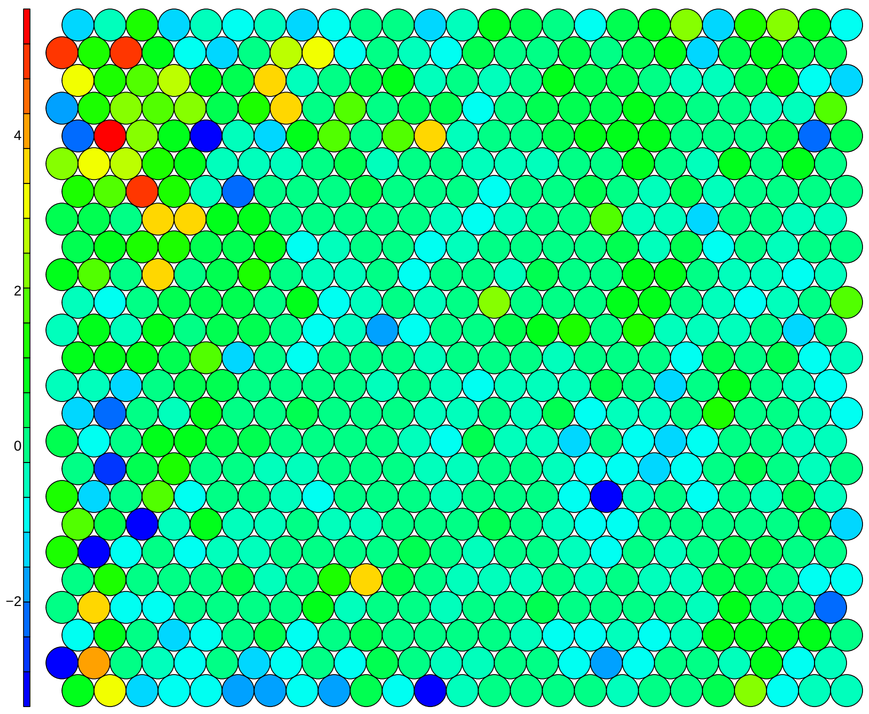
s8

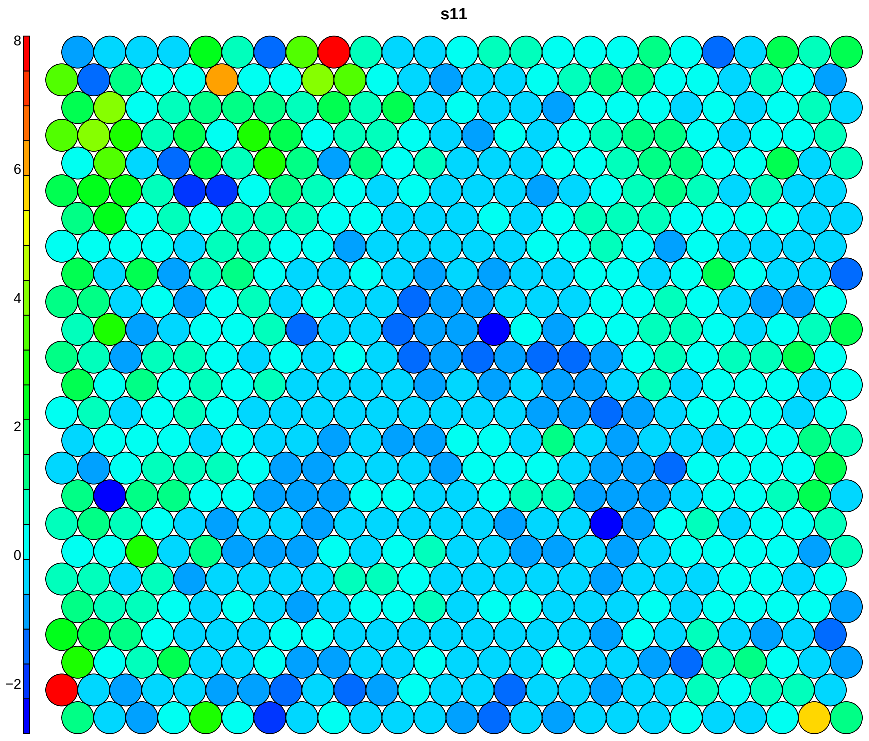


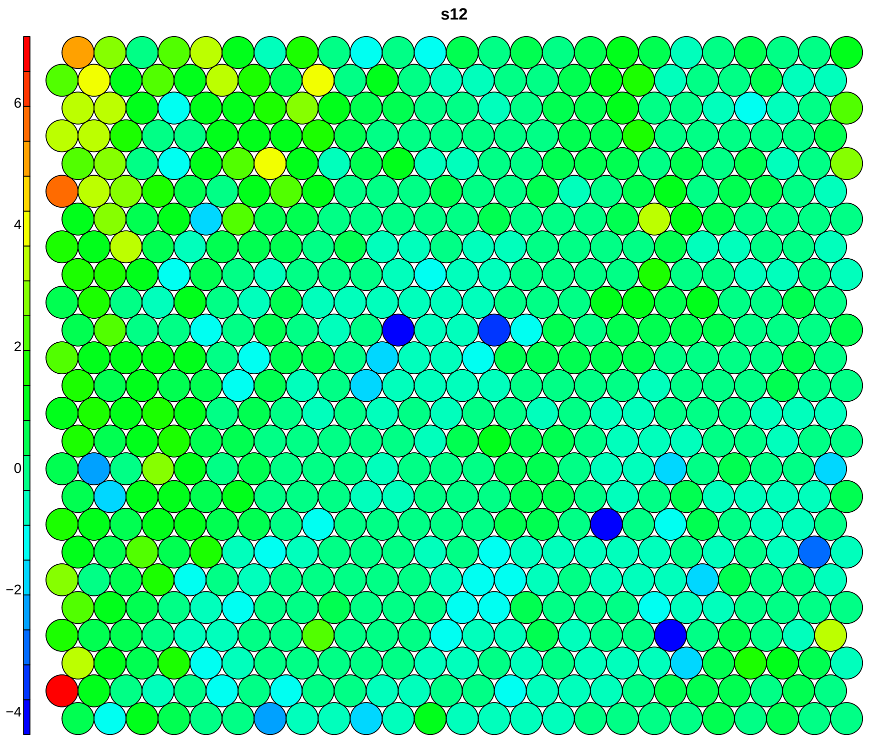
s9



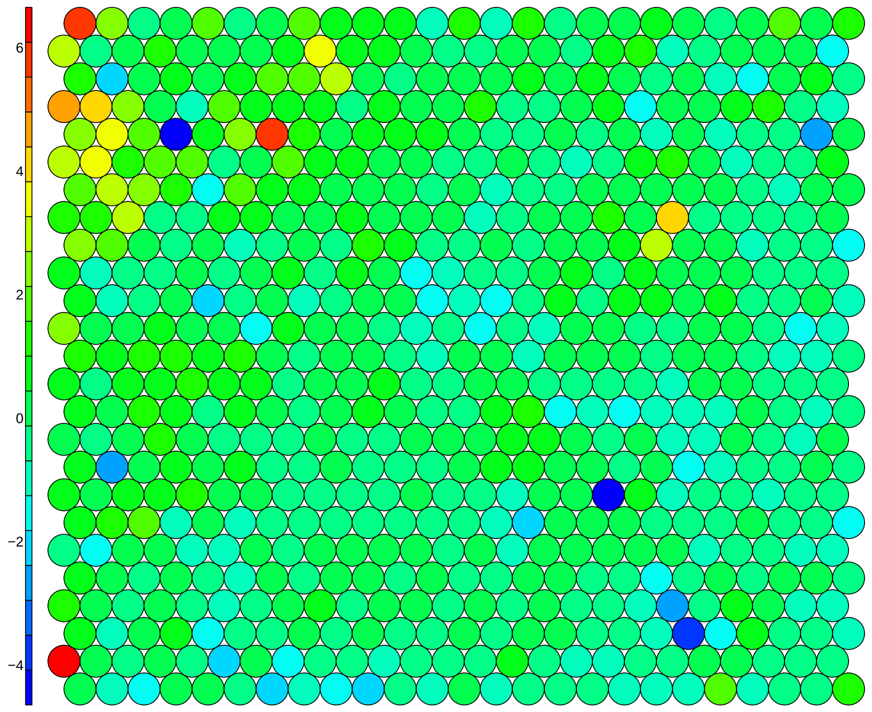
s10



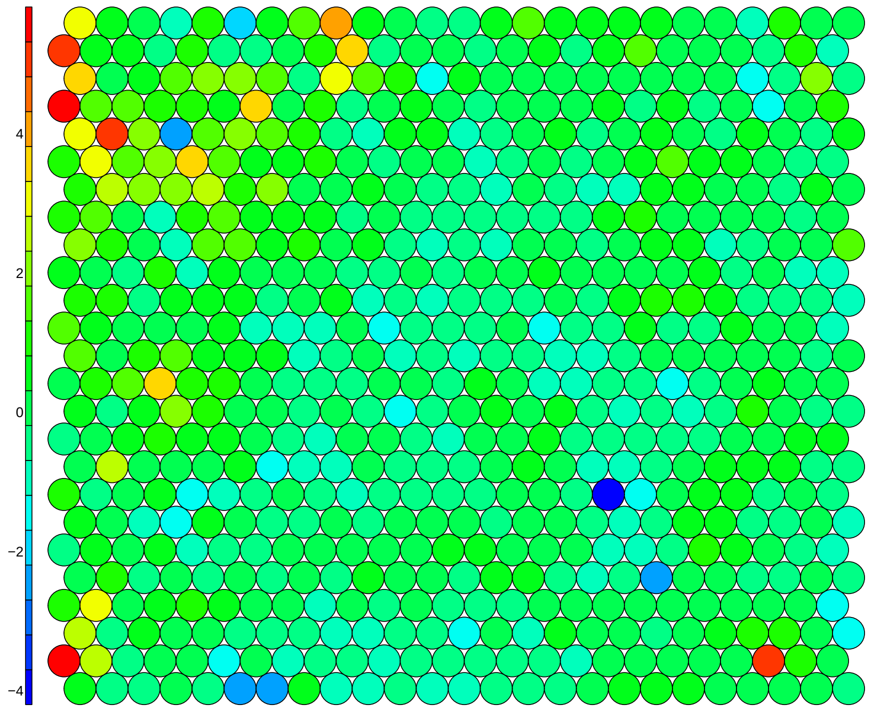

s11
8
6
4
2
0
−2


s12
6
4
2
0
−2
−4

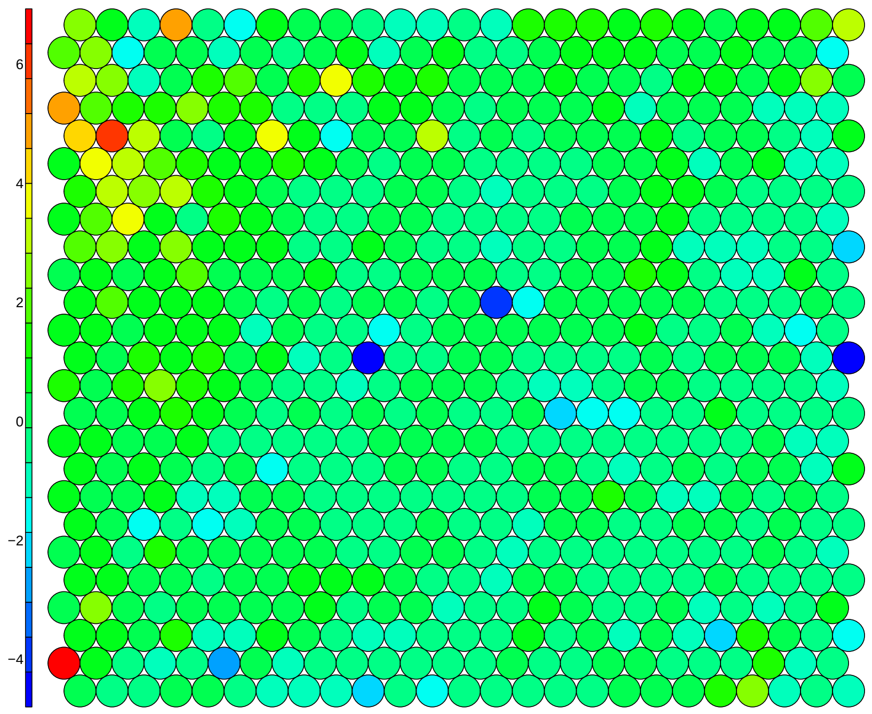
s13



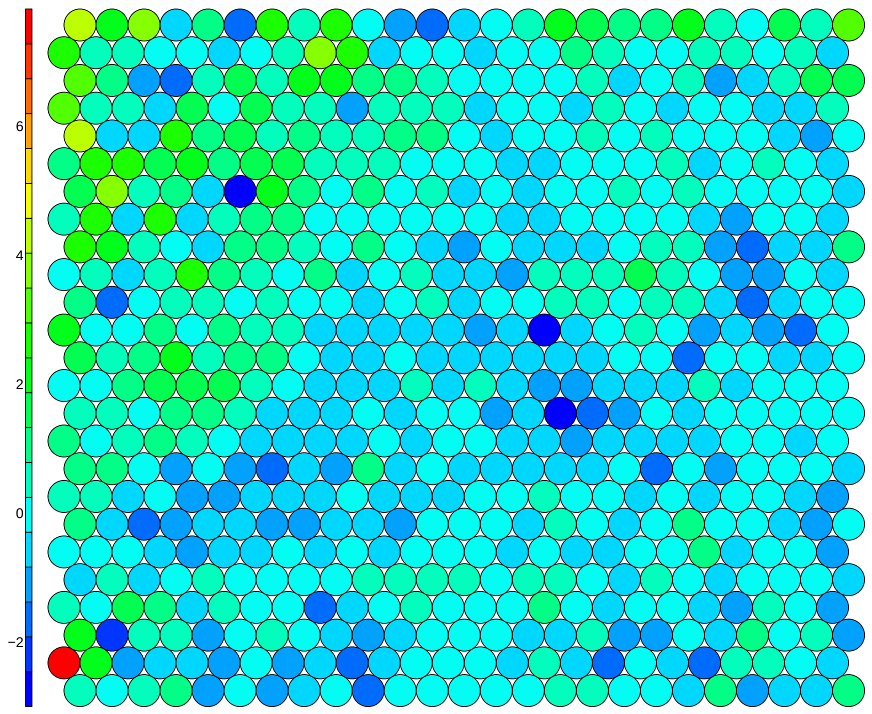
s14



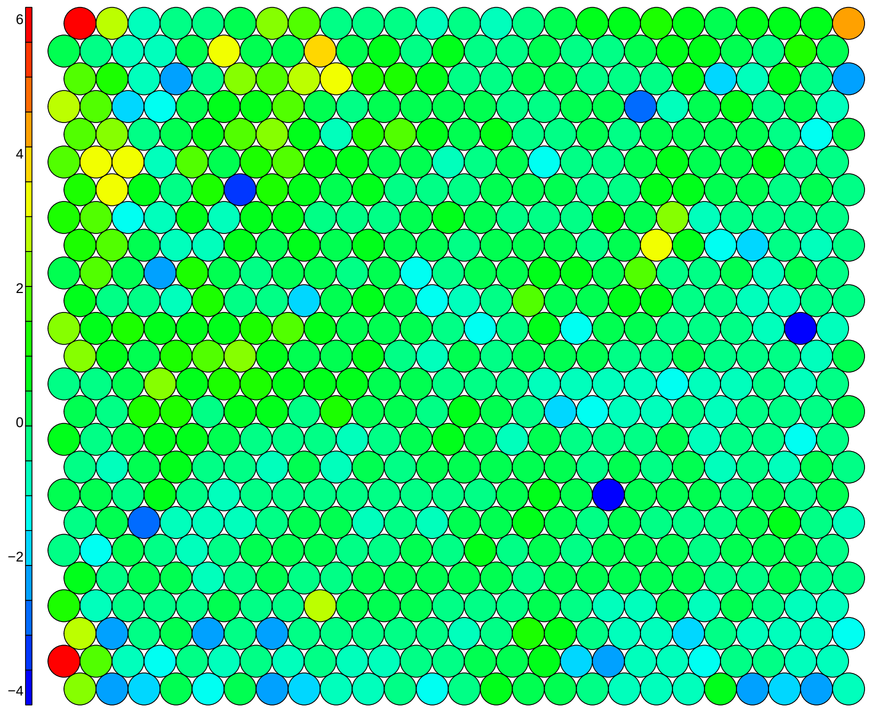
s15



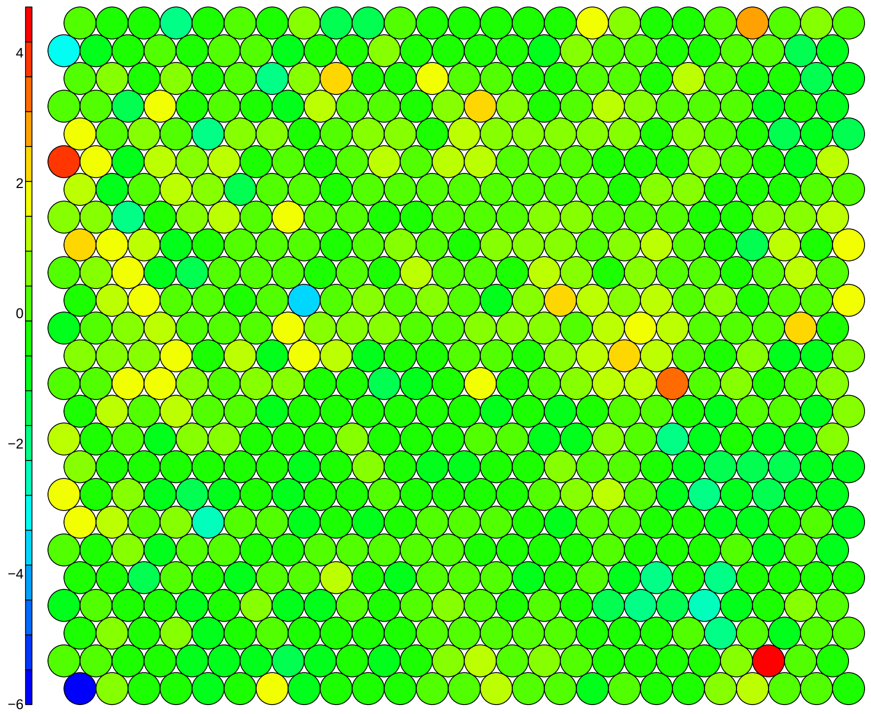
s16



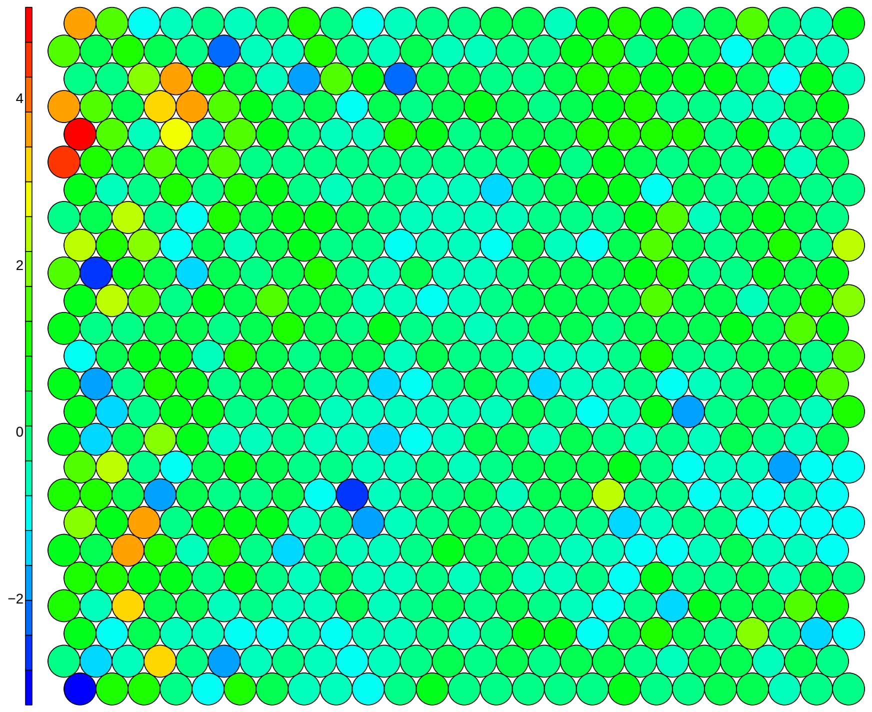
s17



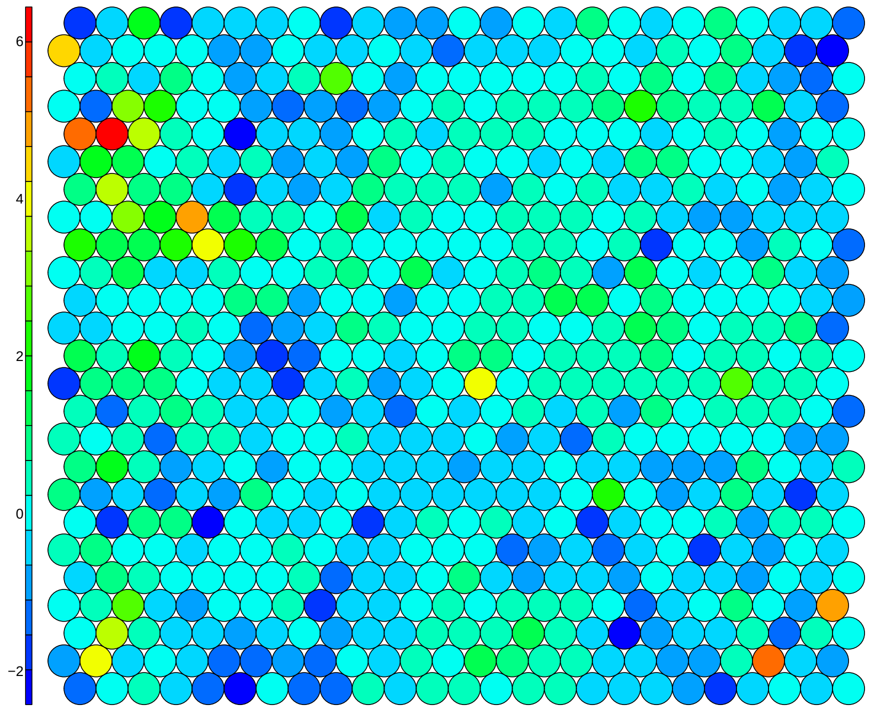
s18



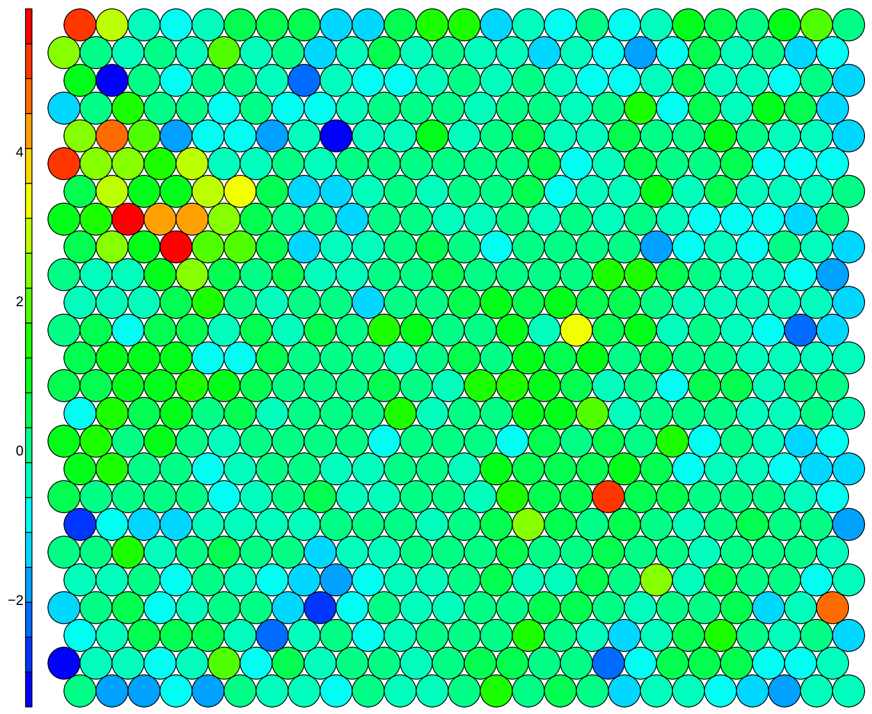
s19



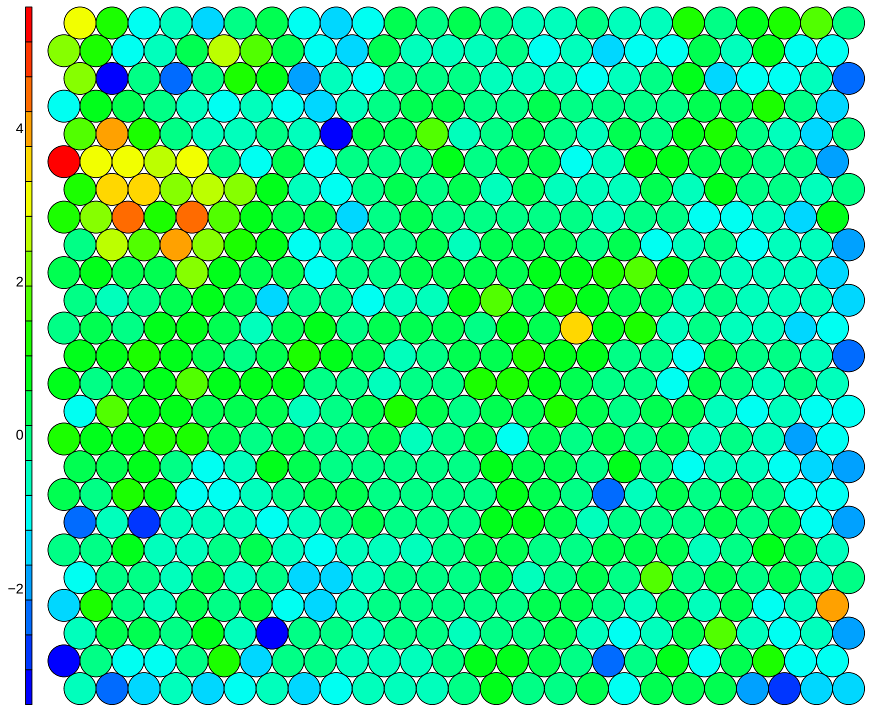
s20

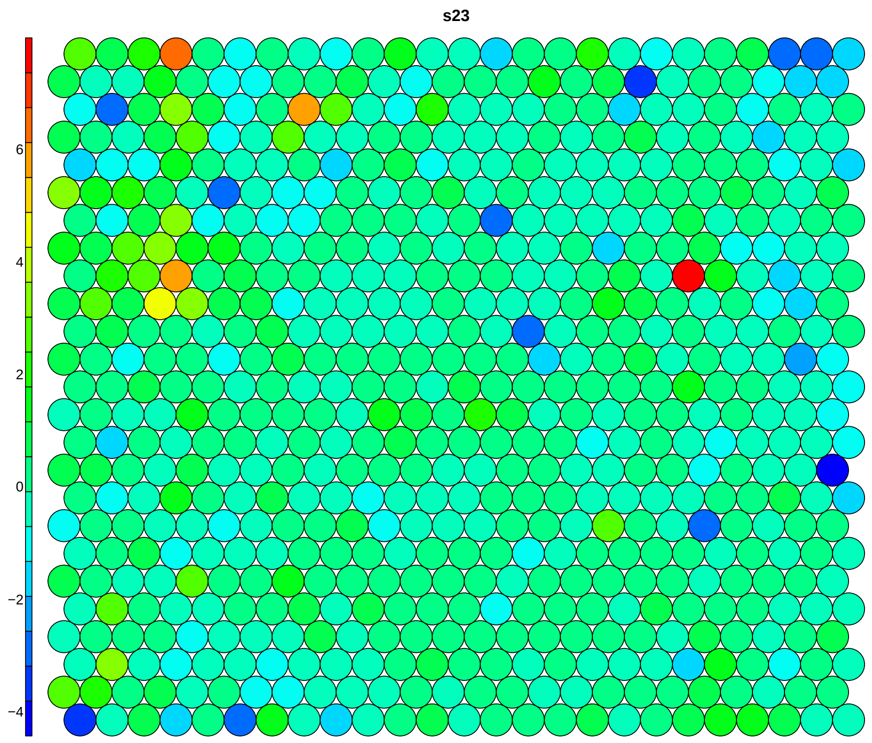


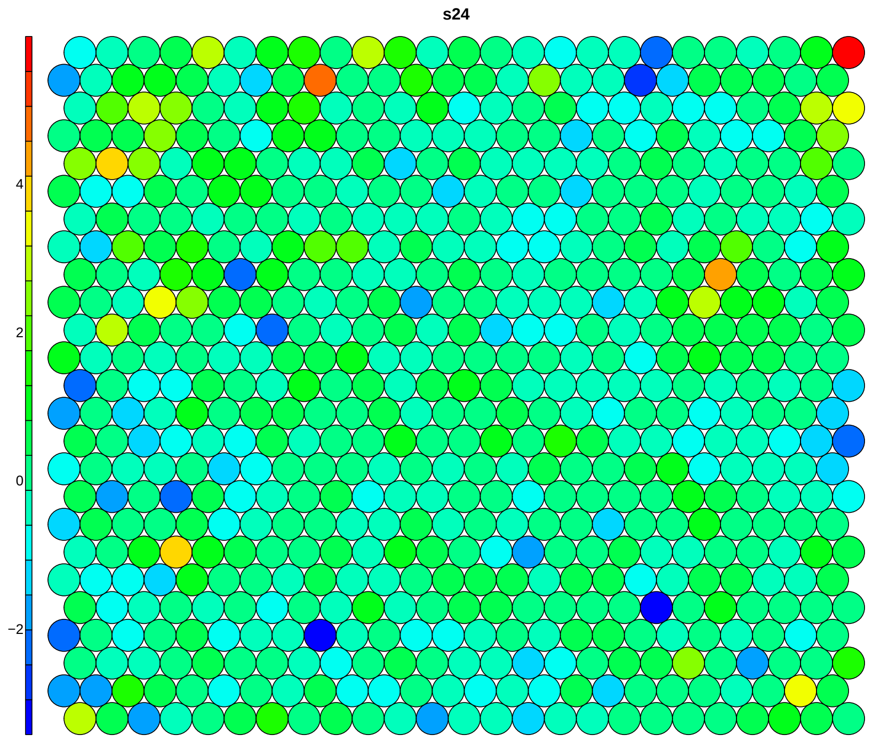
s21



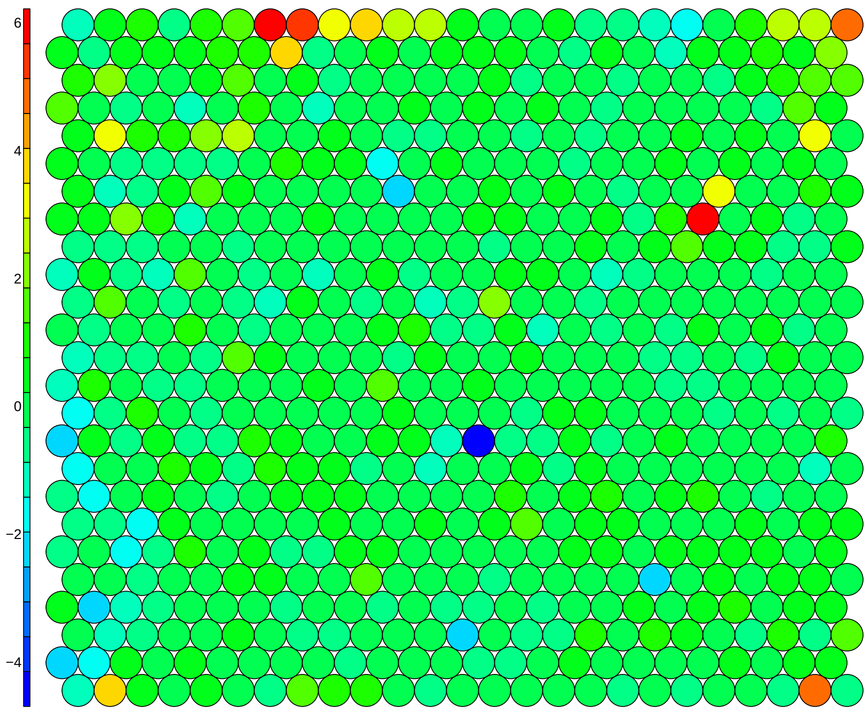
s22



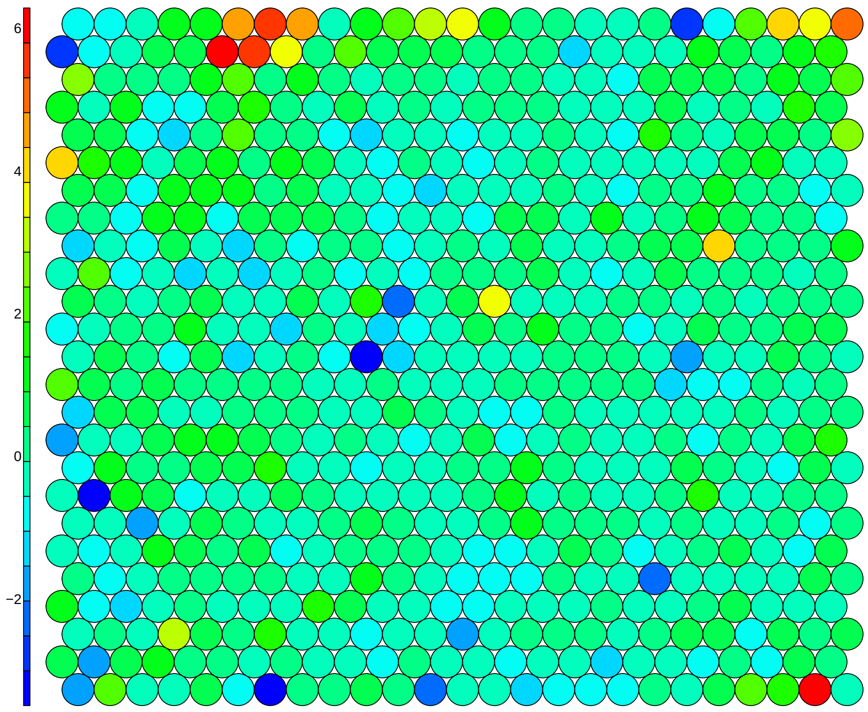

s23
6
4
2
0
−2
−4


s24
4
2
0
−2

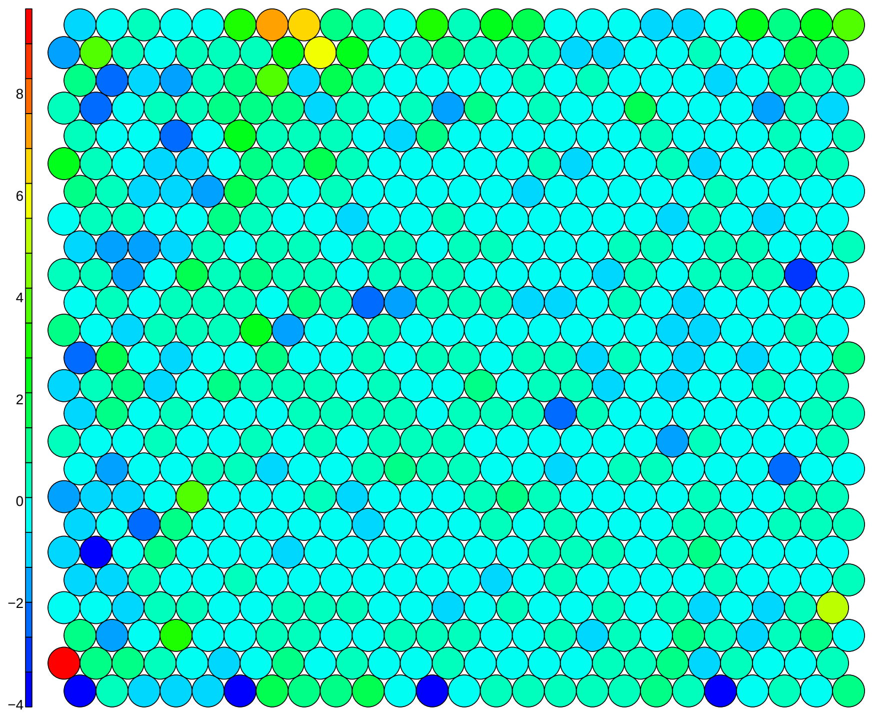
s25



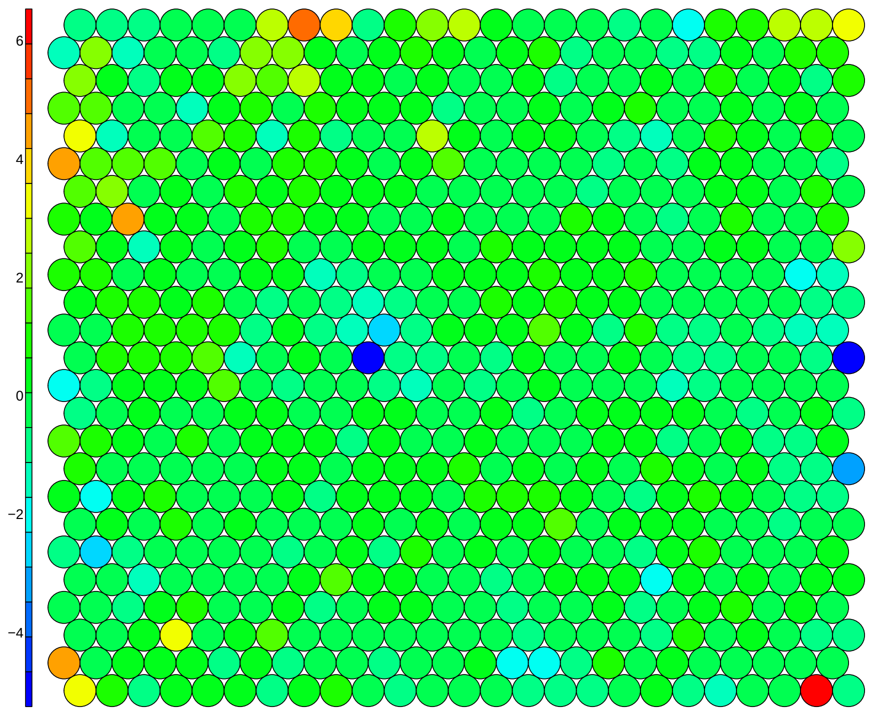
s26



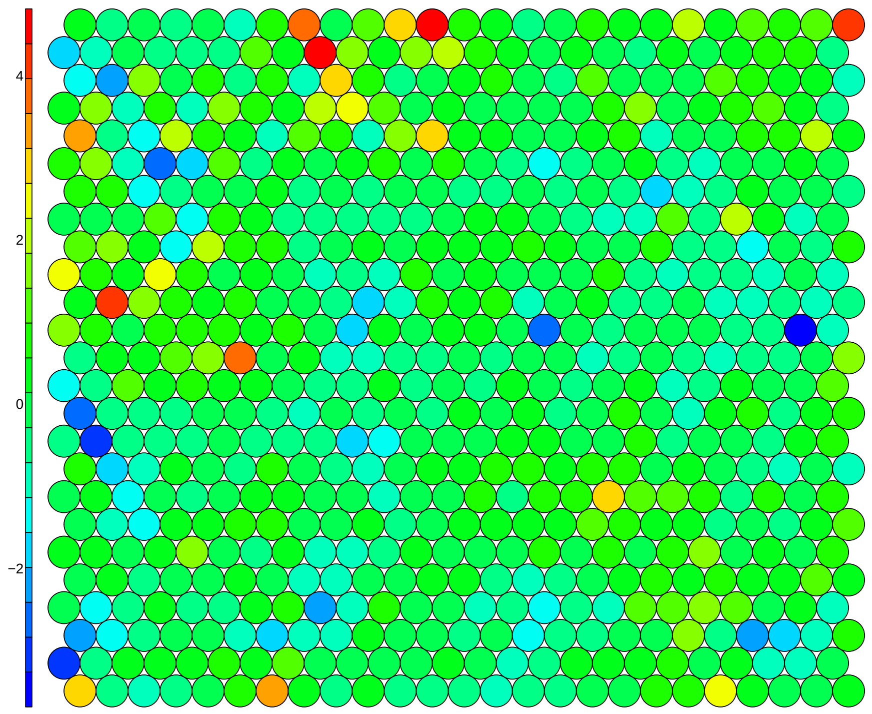
s27



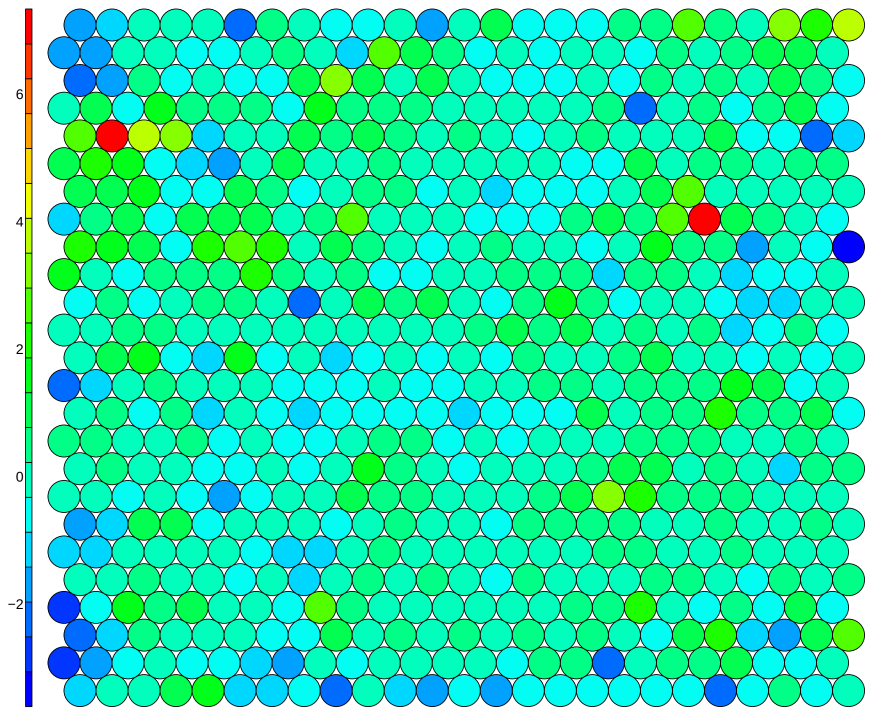
s28



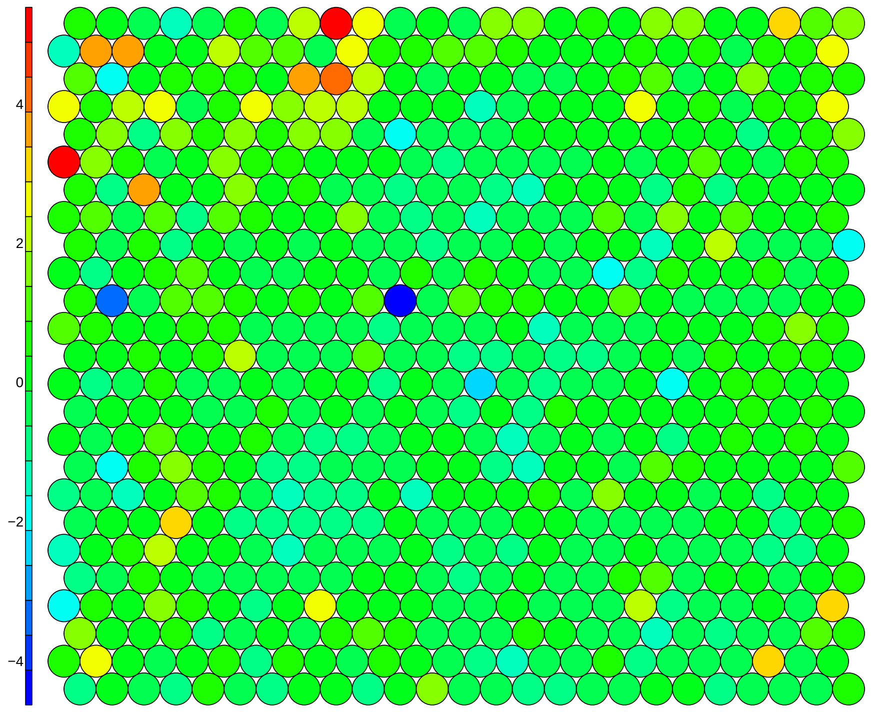
s29



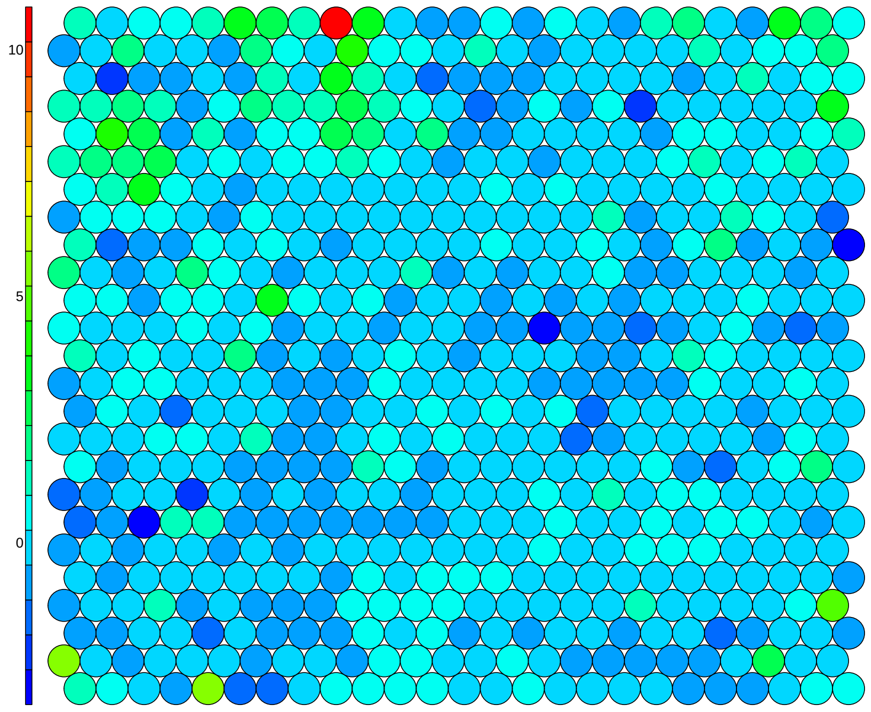
s30



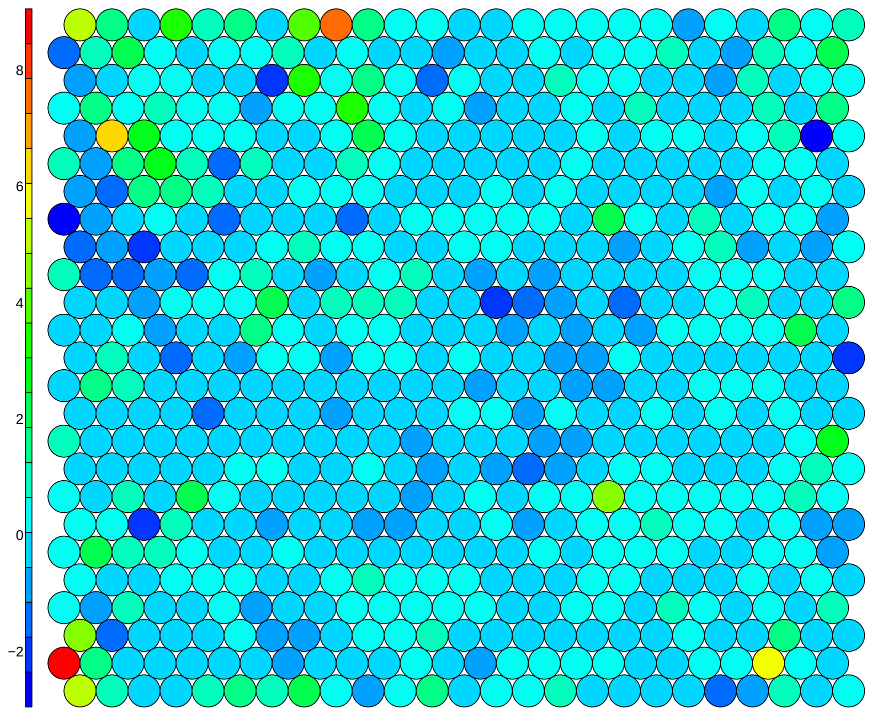
s31



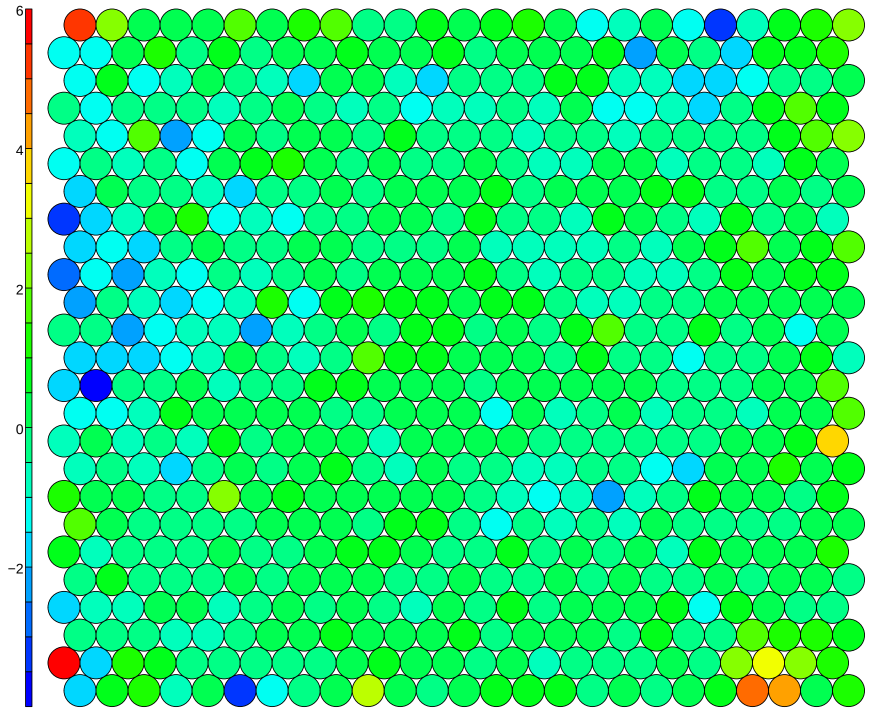
s32



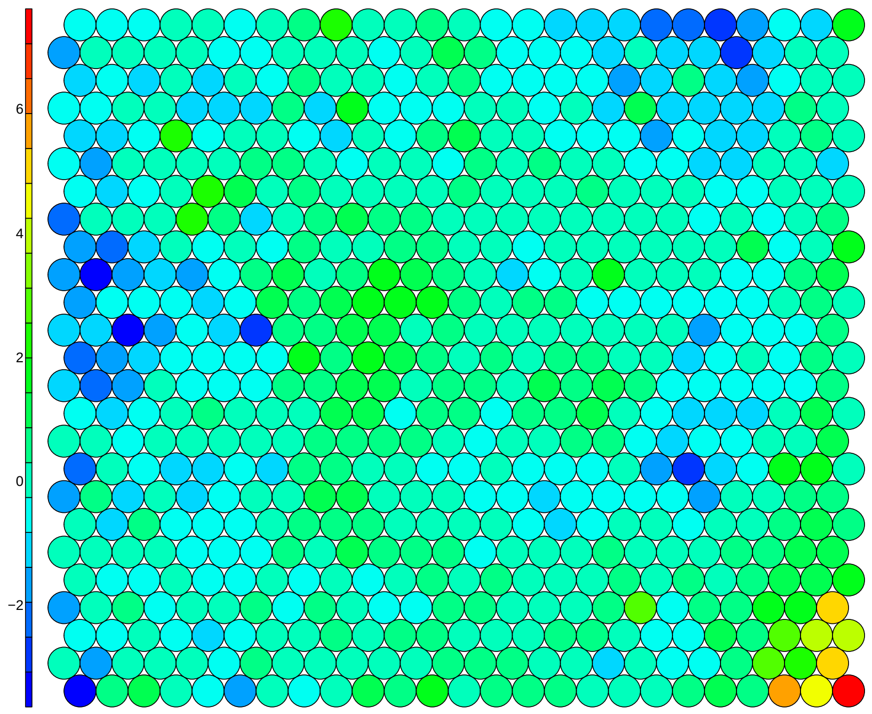
s33



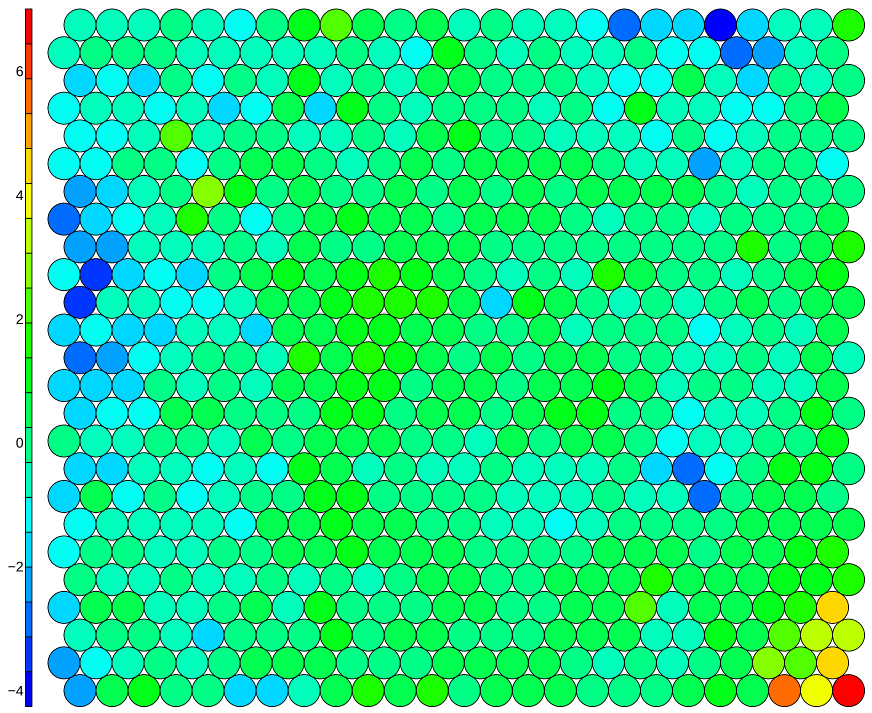
s34



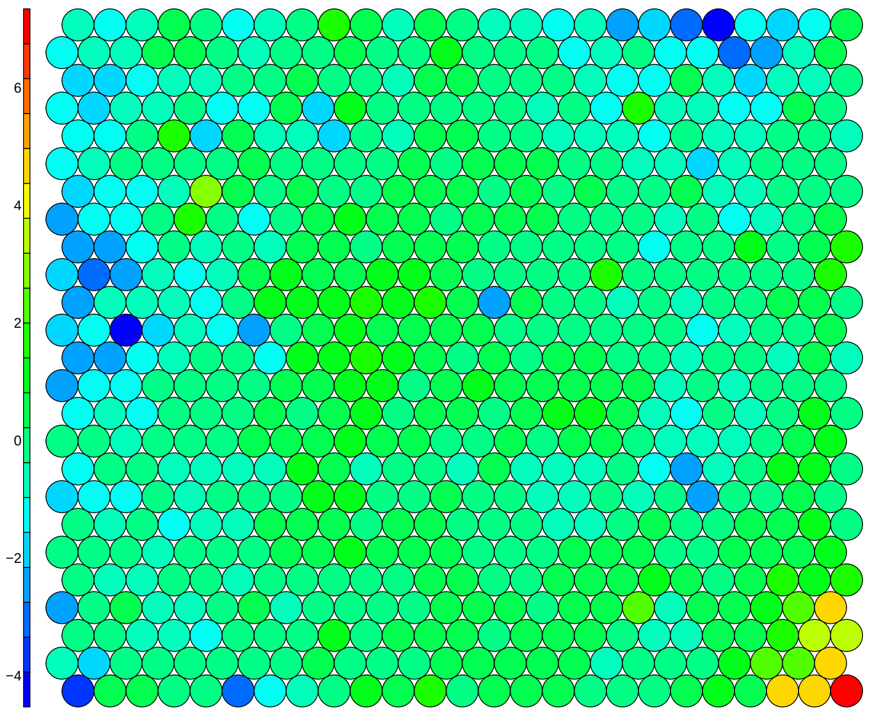
s35



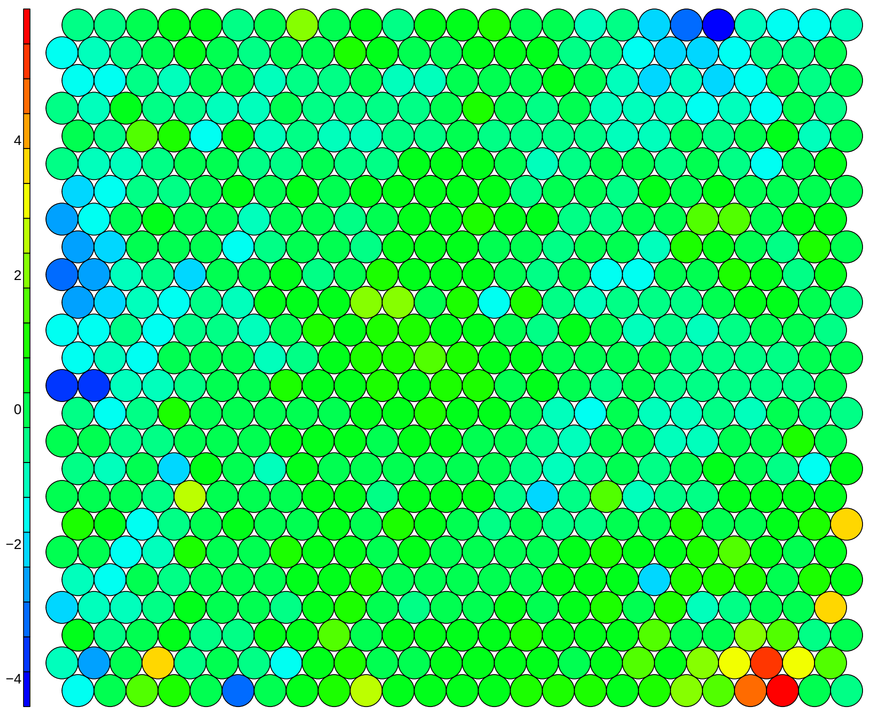
s36



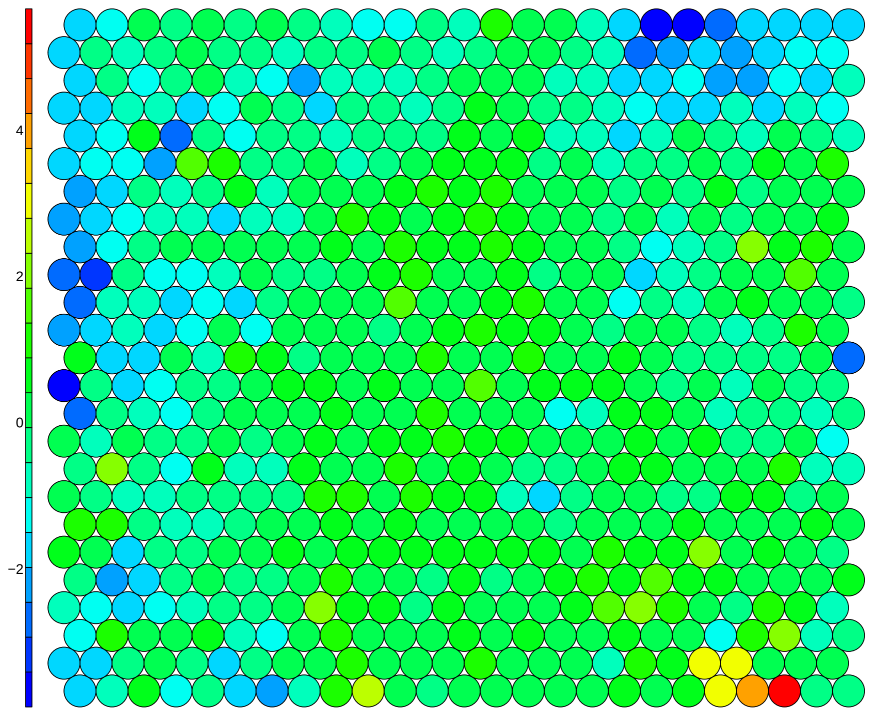
s37



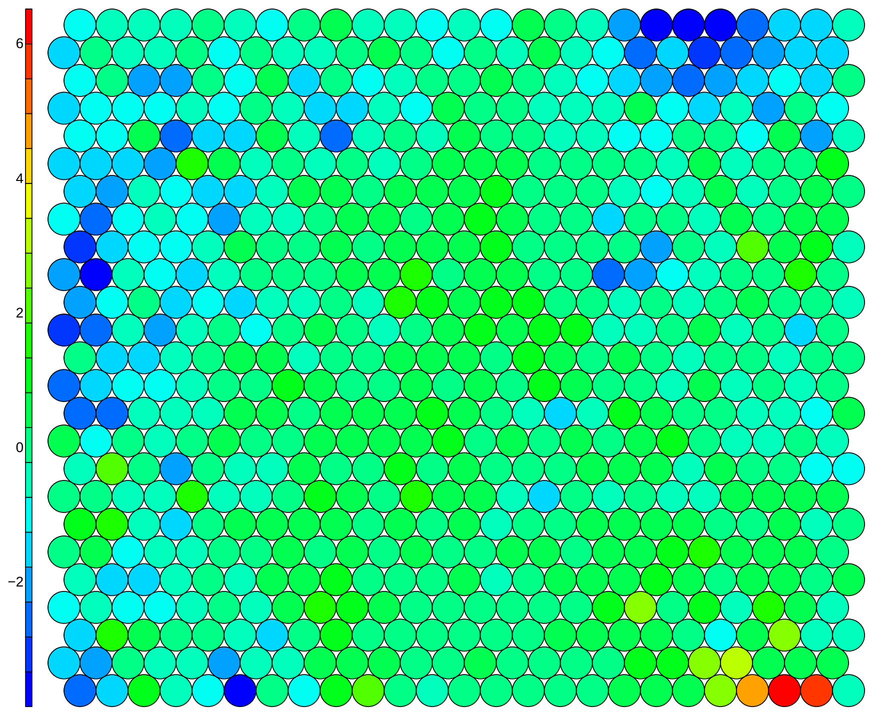
s38

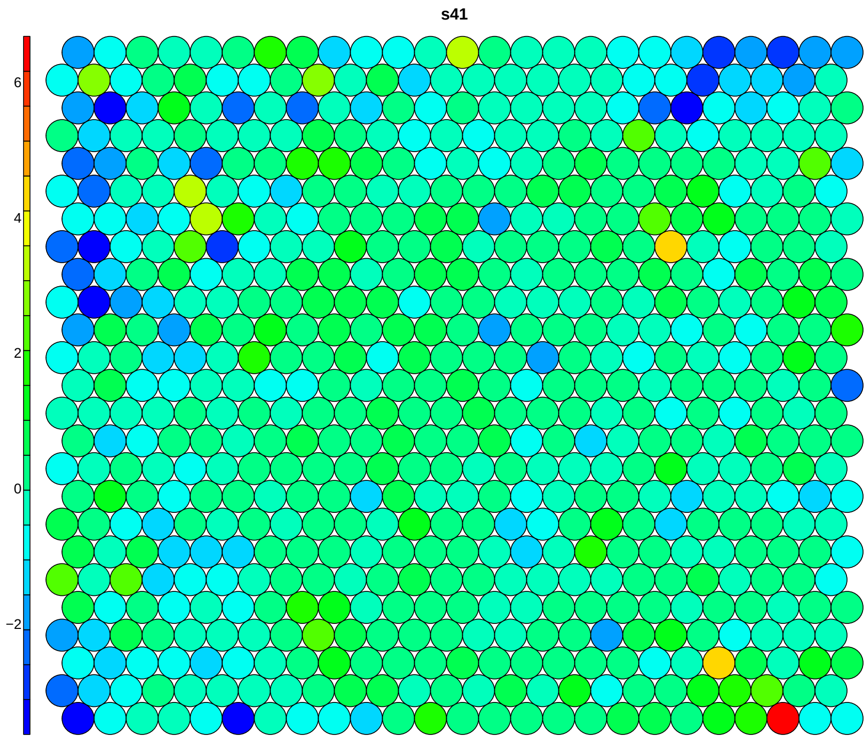


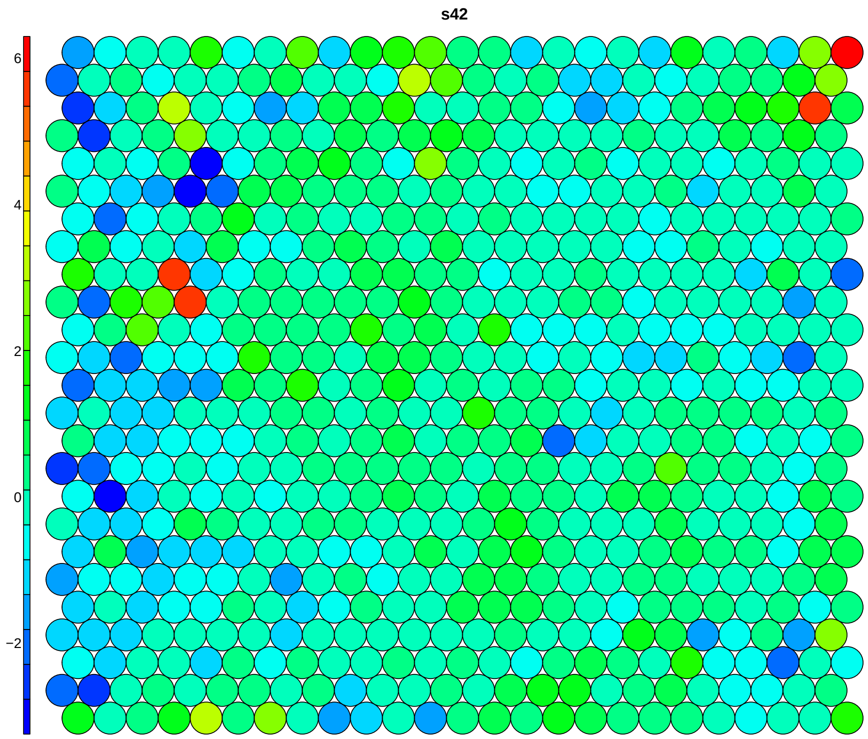
s39



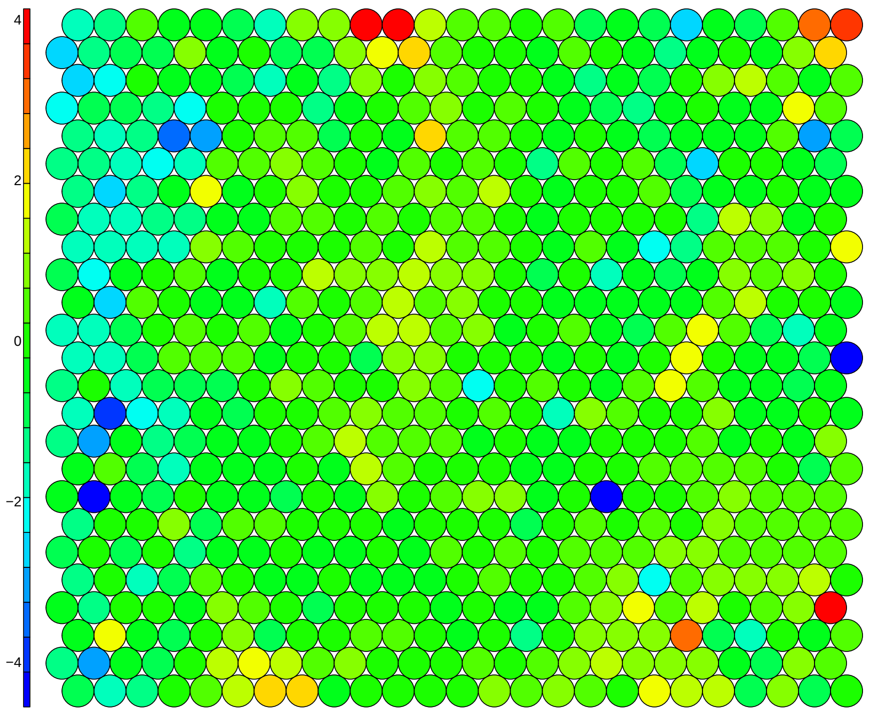
s40



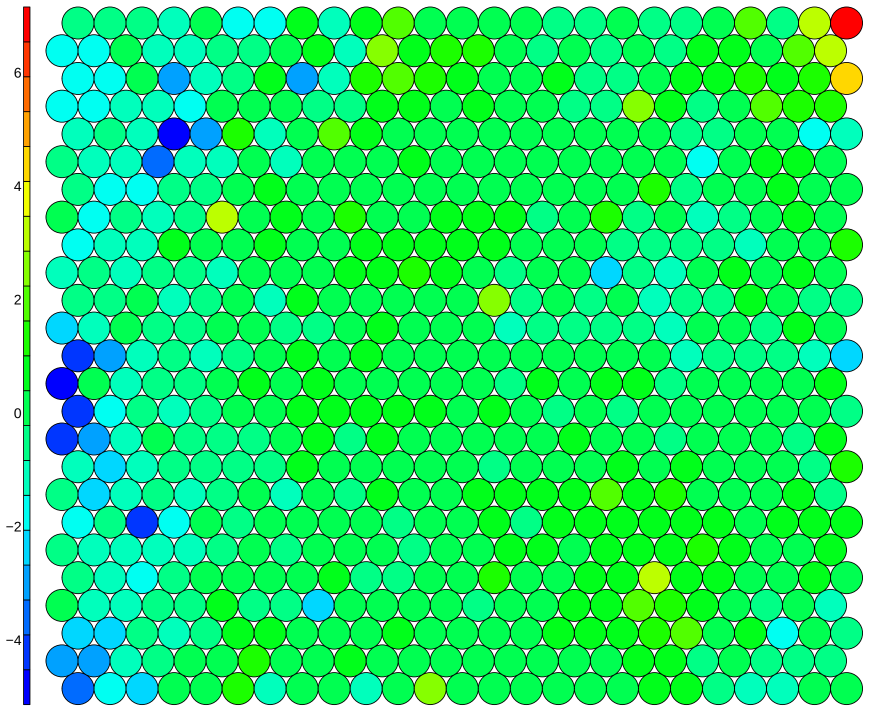

s41
6
4
2
0
−2


s42
6
4
2
0
−2

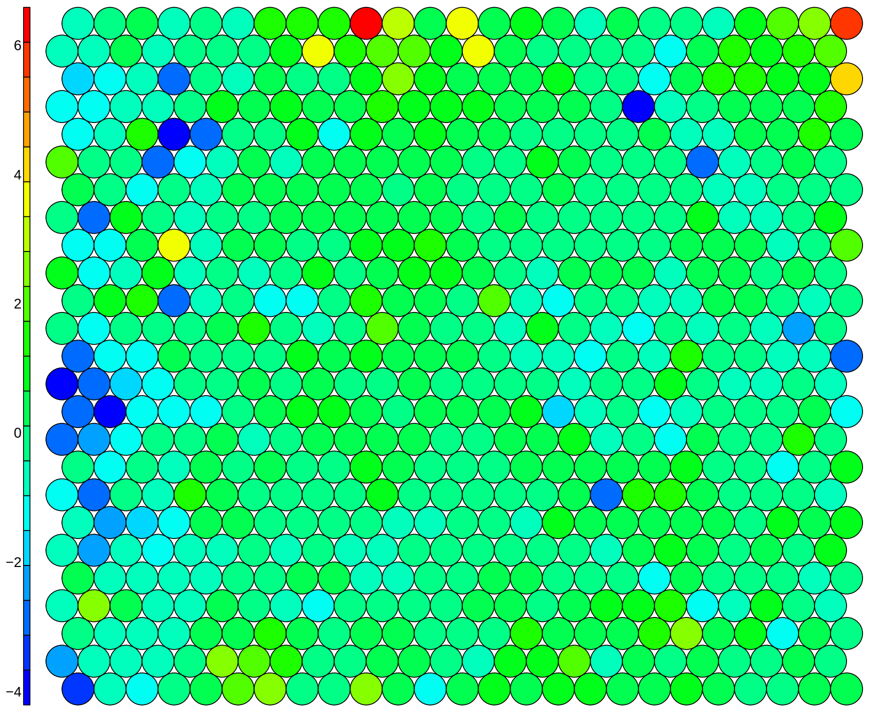
s43



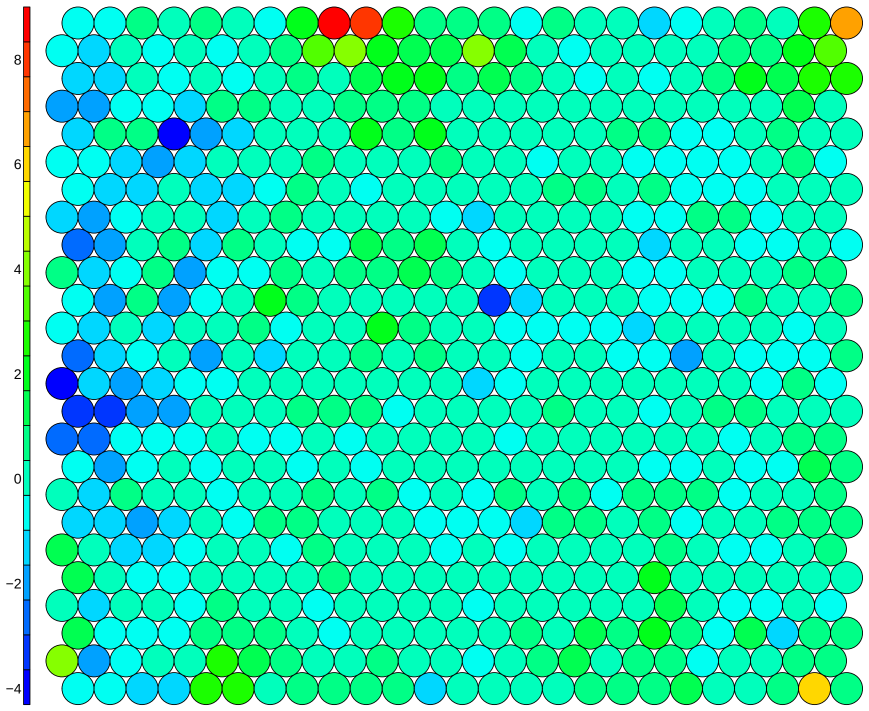
s44



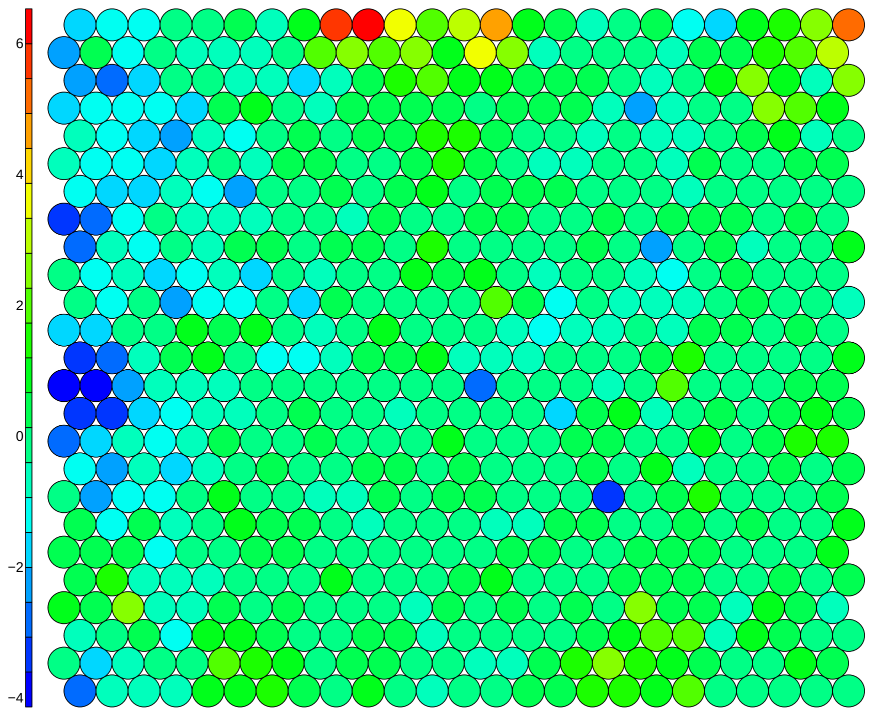
s45



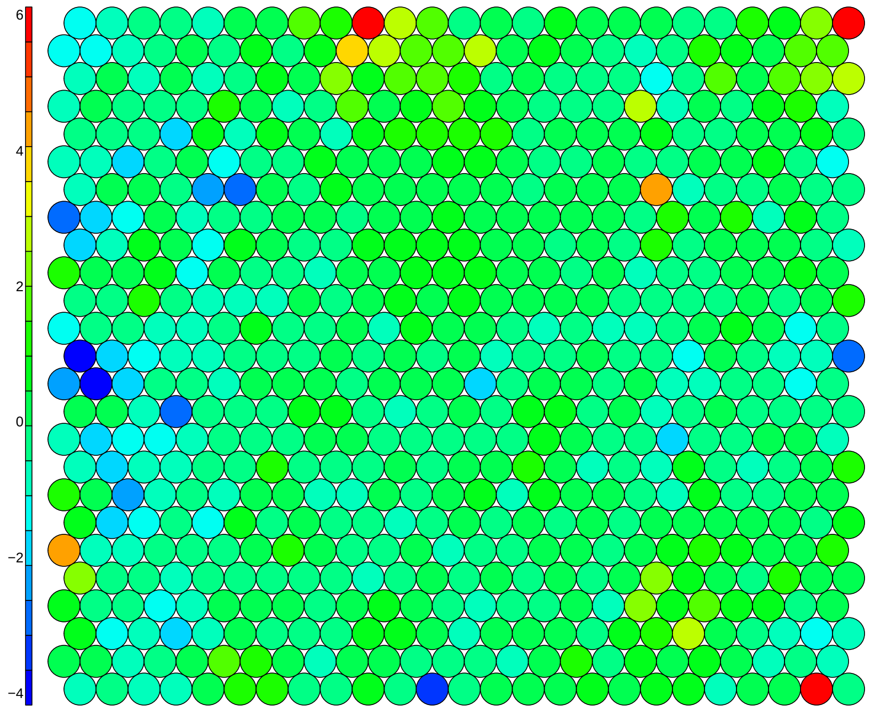
s46

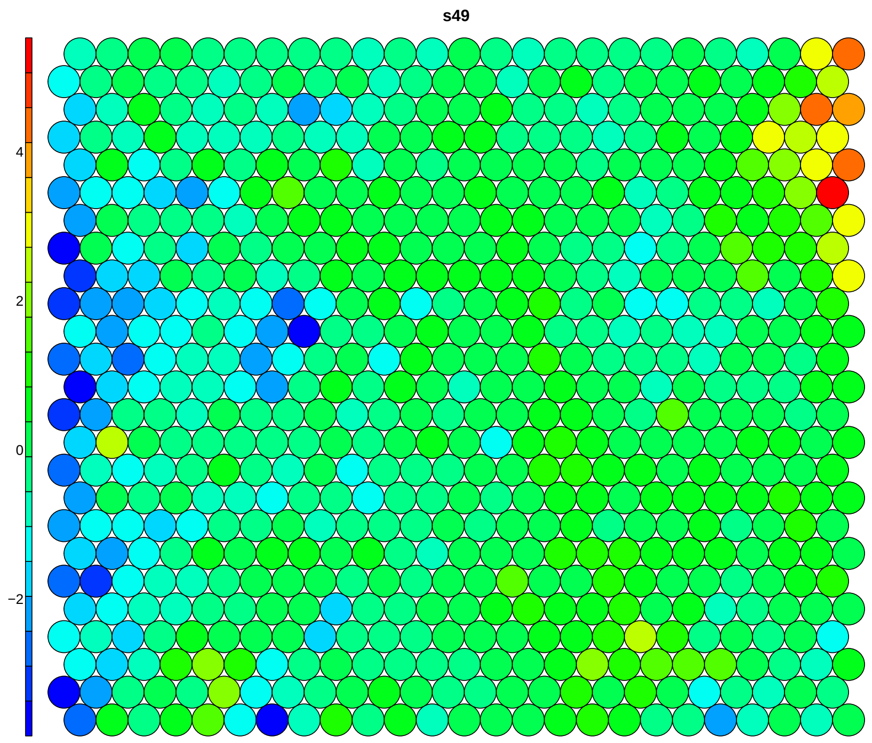


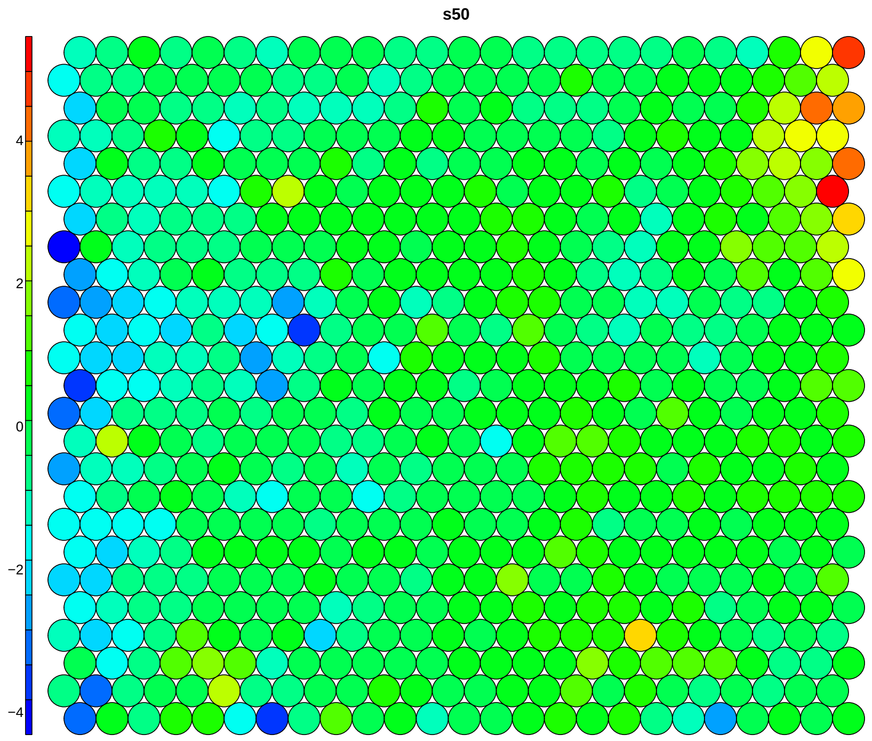
s47



s48




s49
4
2
0
−2


s50
4
2
0
−2
−4