On March 1, 2019, an article was published in Israel’s Ynetnews website, under the title “[The curious case of the concentration of cancer](https://www.ynetnews.com/articles/0,7340,L-5470384,00.html)”. The story reports on a concentration of cancer cases in the town of Rosh Ha’ayin in central Israel.

In the past few years dozens of cases of cancer have been discovered in the center of Rosh Ha’ayin. About 40 people have already died of the disease. Residents are sure that the cause of the disease is cellular antennas on the roof of a building belonging to the municipality. “Years we cry and no one listens”, They say, “People die one after the other.”

I do not underestimate the pain of the residents of Rosh Ha’ayin. I also do not intend to discuss the numbers mentioned in the article. I accept them as they are. I just want to relate only to the claim that the cause of the disease is cellular antennas. It is easy (at least for me) to explain why this claim is at least questionable: there are many more cellular antennas in many places, and around them there is no high rate of morbidity in cancer. If the antennas are carcinogenic, then they need cancer everywhere, not just in Rosh Ha’ayin. On the other hand, I can’t blame them for blaming the antennas. People try to rationalize what they see and find a cause.

I also must emphasize that the case of Rosh Ha’ayin must not be neglected. If it is not because the cellular antennas, it is possible that there is some other risk factor, and the state authorities must investigate.

So why does Rosh Ha’ayin have such a large cluster of cancer morbidity? One possible answer is that there is an environmental factor (other than the antennas) that does not exist elsewhere. Another possible answer is that there may be a non-environmental factor that does not exist elsewhere, maybe a genetic factor (most of the town’s residents are immigrants from Yemen and their descendants). A third and particularly sad possibility is that the local residents suffer from really bad luck. Statistics rules can be cruel.

Clusters happen: if there are no local (environmental or other) risk factors that cause cancer (or any other disease), and the disease spreads randomly across the whole country, then clusters are formed. If the country is divided into units of equal size, then the number of cases in a given area unit follows a [Poisson distribution](https://en.wikipedia.org/wiki/Poisson_distribution). Then there is a small, but not negligible possibility that one of these units will contain a large number of cases. The problem is that there is no way of knowing in advance where it will happen.

The opposite is also true: If the distribution of the number of cases in a given area unit is a Poisson distribution, then it can be concluded that the dispersion on the surface is random.

I will demonstrate the phenomenon using simulation.

Consider a hypothetical country that has a perfect square shape, and its size is 100 x 100 kilometers. I randomized 400 cases of morbidity across the country:

# generate n random points on 0:100 x 0:100

> set.seed(21534)

> n=400

> x=runif(n)\*100

> y=runif(n)\*100

> dat=data.frame(x,y)

> head(dat)

x y

1 15.73088 8.480265

2 12.77018 78.652808

3 45.50406 31.316797

4 86.46181 6.669138

5 27.25488 48.164316

6 17.42388 98.429575

I plotted the 400 cases.

> # plot the points

> plot(dat$x, dat$y, ylim=c(0,100), xlim=c(0,100),

+ asp=1, frame.plot=FALSE, axes=FALSE,

+ xlab=' ', ylab=' ', col='aquamarine', pch=16,

+ las=1, xaxs="i", yaxs="i",

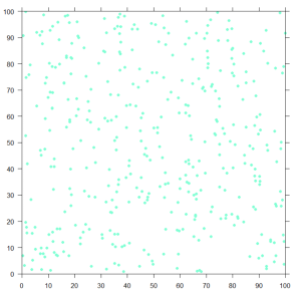
+ )

> axis(1, at=0:10\*10)

> axis(2, at=0:10\*10, las=1, pos=0)

> axis(3, at=0:10\*10, labels=FALSE)

> axis(4, at=0:10\*10, labels=FALSE, pos=100)

Here’s the map I got.  
[](https://sciprincess.files.wordpress.com/2019/03/map1.png)

Next, I divided the map into 100 squares, each 10 x 10 kilometers:

> #draw gridlines

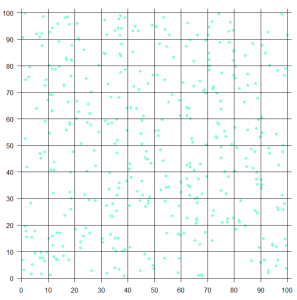
> for (j in 1:9){

+ lines(c(0,100), j\*c(10,10))

+ lines(j\*c(10,10), c(0,100))

+ }

>

[](https://sciprincess.files.wordpress.com/2019/03/map2.png)

In order to count the cases in each square, I assigned row and column numbers for each of the squares, and recorded the position (row and column) for every case/dot:

> # row and column numbers and cases positions

> dat$row=0

> dat$col=0

> dat$pos=0

> for (j in 1:nrow(dat)){

+ dat$row=ceiling(dat$y/10)

+ dat$col=ceiling(dat$x/10)

+ dat$pos=10\*(dat$row-1)+dat$col

+ }

>

Now I can count the number of points/cases in each square:

> # calculate number of points for each position

> # ppp=points per position

> dat$count=1

> ppp=aggregate(count~pos, dat, sum)

> dat=dat[,-6]

But of course, it is possible that there are squares with zero cases; (actually the data frame ppp has only 97 rows). Let’s identify them:

> # add positions with zero counts, if any

> npp=nrow(ppp)

> if(npp<100){

+ w=which(!(1:100 %in% ppp$pos))

+ addrows=(npp+1):(npp+length(w))

+ ppp[addrows,1]=w

+ ppp[addrows,2]=0

+ ppp=ppp[order(ppp$pos),]

+ }

>

And now we can get the distribution of number of cases in each of the 100 squares:

> # distribution of number of points/cases in each position

> tb=table(ppp$count)

> print(tb)

0 1 2 3 4 5 6 7 8 9 11

3 9 12 21 15 17 13 5 1 3 1

>

We see that there is one very unlucky cluster with 11 cases, and there also 3 squares with 9 cases each. Let’s paint them on the map:

> # identify largest cluster

> mx=max(ppp$count)

> loc=which(ppp$count==11)

> clusters=dat[dat$pos %in% loc,]

> points(clusters$x, clusters$y, col='red', pch=16)

>

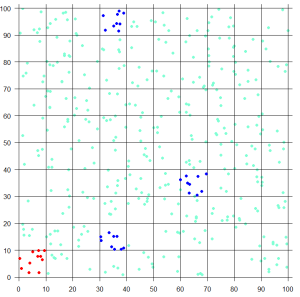
> # identify second lasrgest cluster/s

> loc=which(ppp$count==9)

> clusters=dat[dat$pos %in% loc,]

> points(clusters$x, clusters$y, col='blue', pch=16)

>

[](https://sciprincess.files.wordpress.com/2019/03/map3.png)

Let’s also mark the squares with zero points/cases. In order to do this, we first need to identify the row and column locations of these squares:

> # identify sqaures without cases

> # find row and column locations

> loc=which(ppp$count==0)

> zeroes=data.frame(loc)

> zeroes$row=ceiling(zeroes$loc/10)

> zeroes$col=zeroes$loc %% 10

> w=which(zeroes$col==0)

> if(length(w)>0){

+ zeroes$col[w]=10

+ }

> print(zeroes)

loc row col

1 8 1 8

2 31 4 1

3 99 10 9

>

So there is one empty square in the 8th column of the first row, one in the first column of the 4th row, and one in the 9th column of the 10th row. Let’s mark them. To do that, we need to know the coordinates of each of the four vertices of these squares:

# mark squares with zero cases

> for (j in 1:nrow(zeroes)){

+ h1=(zeroes$col[j]-1)\*10

+ h2=h1+10

+ v1=(zeroes$row[j]-1)\*10

+ v2=v1+10

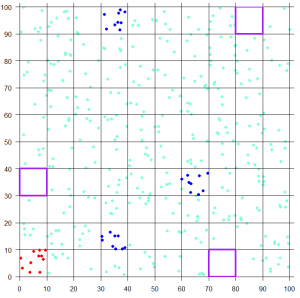
+ lines(c(h1,h2), c(v1,v1), lwd=3, col='purple')

+ lines(c(h1,h2), c(v2,v2), lwd=3, col='purple')

+ lines(c(h1,h1), c(v1,v2), lwd=3, col='purple')

+ lines(c(h2,h2), c(v1,v2), lwd=3, col='purple')

+ }

[](https://sciprincess.files.wordpress.com/2019/03/map4.png)

Do you see any pattern?

How well does the data fit the Poisson distribution? We can perform a goodness of fit test.  
Let’s do the log-likelihood chi-square test (also known as the [G-test](https://en.wikipedia.org/wiki/G-test)):

> # log likelihood chi square to test the goodness of fit

> # of the poisson distribution to the data

>

> # the obserevd data

> observed=as.numeric(tb)

> values=as.numeric(names(tb))

>

> # estimate the poisson distribution parameter lambda

> # it is the mean number of cases per square

> lambda=nrow(dat)/100

> print(lambda)

[1] 4

>

> # calculate the expected values according to

> # a poisson distribution with mean lambda

> expected=100\*dpois(values, lambda)

>

> # view the data for the chi-square test

> poisson\_data=data.frame(values, observed, expected)

> print(poisson\_data)

values observed expected

1 0 3 1.8315639

2 1 9 7.3262556

3 2 12 14.6525111

4 3 21 19.5366815

5 4 15 19.5366815

6 5 17 15.6293452

7 6 13 10.4195635

8 7 5 5.9540363

9 8 1 2.9770181

10 9 3 1.3231192

11 11 1 0.1924537

>

> # calculate the degrees of freedom

> df=max(values)

> print(df)

[1] 11

>

> # calculate the test statistic and p-value

> g2=sum(observed\*log(observed/expected))

> pvalue=1-pchisq(g2,df)

> log\_likelihood\_chi\_squrae\_test=data.frame(g2, df, pvalue)

> print(log\_likelihood\_chi\_squrae\_test)

g2 df pvalue

1 4.934042 11 0.9343187

>

We cannot reject the hypothesis that the data follows the Poison distribution. This does not imply, of course, that the data follows the Poisson distribution, but we can say that the Poisson model fits the data well.