Our ideas about the problem

A histogram is an ordered set of bins and frequencies. Each bin represents a data range and the associated frequency to it summarizes the number of values that lie in the range. This is intuitive for unidimensional data, were this kind of representation of the data is powerful and, if correctly done, can lead to good approximation of the real distribution of the data. In this situation, the biggest inconvenient is that histograms need to be stored in memory. More specifically, to store them we need:

* One array containing the bounds of the bins
* One array containing the frequencies belonging to in each bin

In this specific project, our interest is on saving information about ranges of the kind It means that each value to store is, in a way, bidimensional, being composed by both the lower and the upper bound. If our goal is to predict the number of rows that will match a certain property (for example, *“how many ranges are strictly left respect a certain point in the time”*), we have to find somehow a way to summarize the data. That should be done considering memory issues.

Furthermore, one of the targets of this section is to start providing solution also for the estimation of the cardinality of the joins on range types. In other words, results of our present analysis are supposed to make prevision on the number of rows generated by a join easier.

We propose three different approaches:

1. Optimizing the current way estimation is done
2. Using a linear-model-based approach
3. Assuming a parametric probability distribution for the data.

The following description is done for the selectivity of *“strictly left”* operator, considering that for the *“Overlap”* operator we can use the same results. In fact, A && B <=> NOT (A << B OR A >> B).

**Optimizing the current way estimation is done**

Currently, the estimation of the selectivity of the operator *“strictly left of”* is based on equidepth histograms of upper bounds of the range type data. The operation is of the type: *“how many ranges end before a certain range R started”.*

The basic idea is that, since each bin contains the same number of values the only thing to calculate is the number of bins that precedes the lower bound of R. In addition, if R’s lower bound does not correspond to a bin boundary (i.e. if it is *inside* one of the bins), we have to consider also the tuples that are in the interval that goes from the starting point of the bin in which lower bound of R is (A in Figure 2.1) to the lower bound of R. This number is currently approximating assuming uniform distribution of the values inside the bin. Then, the result should be adjusted for the number of empty and null values.

More precisely, given:

,

It is assumed that:

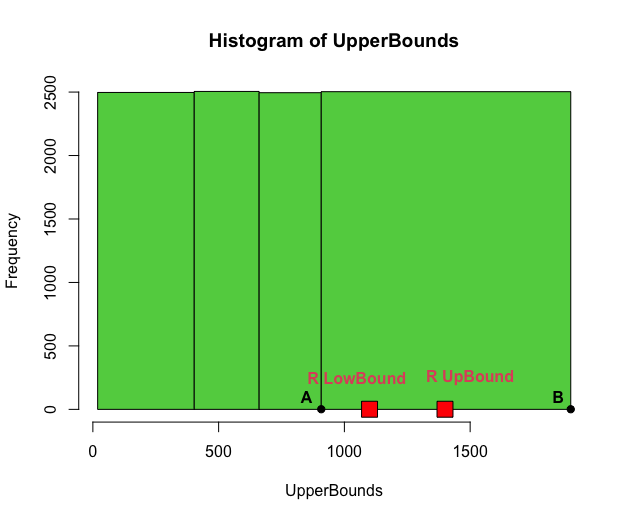


Figure 2.1. Upper and lower bound of R represented within the histogram of upper bounds

Maintaining the current implementation of histograms, the only thing that can be improved is the evaluation of *“a”.* The assumption of uniformity can be not respected in some situation, in particular when we are dealing with extremity bin.

For example, in the current example in Figure 2.1, the bin of interest is the last one. This bin goes from 900 to 1900. Let’s suppose that all the 2500 observations contained in it have value 90, just one has value 1800 nd R’s lower bound equal to 1200. In this extreme situation, it’s clear that the assumption of uniform distribution of the data inside this bin leads to biased results. In fact, we would have , while we know that the real fraction of values between 900 and 1200 is .

So, it’s clear that better solution could be implemented. One possibility can be to use Biased Histogram (as suggested in Viswanath et al., Improved Histograms for Selectivity Estimation of Range Predicates, 1996).

allocating some singleton buckets at the extremity of the histograms, having both advantages of saving allocation space in memory and exclude outliers from the evaluation.

Anyway, this approach **does not provide support for the evaluation of cardinality of joins made on range type columns**.

**Assuming a parametric probability distribution for the data**

Another approach, starting from the three-dimensional histogram, is to assume the data coming from a certain probability distribution. This distribution could be thought to be a double exponential , where .

This approach makes immediate the calculation of the probability of observing points (ranges) inside a certain range (period of time) using the cumulative probability function.

We just need to save in memory the parameter of the distribution () and not the whole histogram.

**Using a generalized linear-model-based approach**

As said in the previous section, the data can be seen as points with a pair of coordinates . In Figure 2.2 is provided a graph representing 10000 ranges randomly generated.

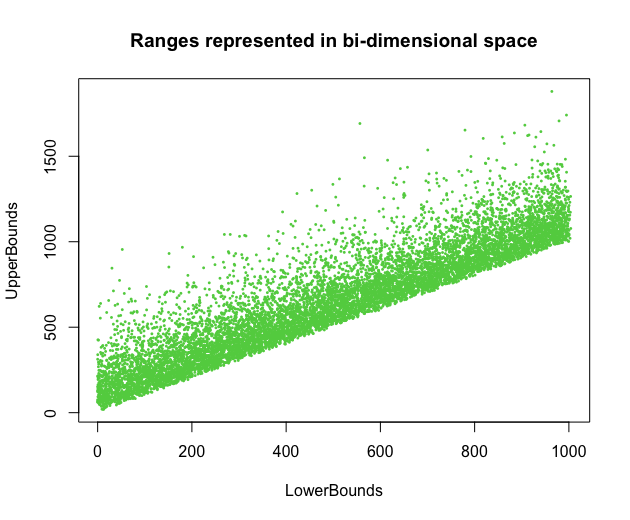


Figure 2.2. Ranges can be represented as points

On the top of this graph, we can build a histogram of frequencies, cutting the point’s space in an arbitrary way. We decided to do it with equiwidth squares. See Figure 2.3.

Immagine che contiene testo, strumento scrittorio, matita, stazionario

Descrizione generata automaticamente

Figure 2.3. Histograms of frequencies

Apparently, the three-dimensional histogram can be thought just as the extension of the bidimensional one, used with the same concept. Working by allocating this kind of histograms have, in general, two disadvantages:

* the huge problem of three-dimensional histograms is that they tend to be not continuous, while the data they summarize may be! Intuitively, using them we are losing the property of being able to differentiate the curve to obtain the cumulative distribution function from the density function, estimated by the frequencies.
* They need a lot of memory space to allocate the 3 dimensions.

Keeping in that we want to provide solutions useful for estimation about the selectivity of Joins operations too, these aspects are critical.

On the other hand, the display of data shows that the way they lie along the tridimensional space could be approximated by a plane. In other words, it means that we can use a linear model, built on the data we have, to provide estimations about the number of values equal to a certain range, given a certain pair of lower and upper bound.

We can assign the observed frequency in each bin of the three-dimensional histogram to its center, like shown in Figure 2.4. While doing it, we obtain a frequency observed for each pair of LowerBoud UpperBound present in the data.

Once built the model, **we do not need to keep the histogram in memory**. In fact, **we can just save the parameters of the linear model**.

Even if we could lose some percentual points of accuracy in these estimations ( and it has to be proved while benchmarking), this approach allowed to save as space as possible for more useful histograms.

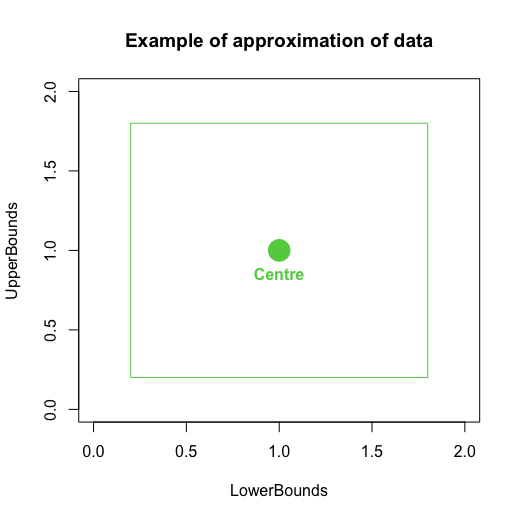


Figure 2.4. The frequency has been observed inside the whole green square. To be able to work with points, we assume the center to be given the frequency observed in its square neighborhood.

After some evaluations of different models (see documentations), we decided to use a Zero Inflated Negative Binomial Generalized Linear Model.

This model assumes the data coming from this distribution:

Where and

It implies and

Both the variables ( and ) are used to estimate and :

This model can be fitted maximizing the likelihood function using Netwon Rapson approximations.

Once fitted the model, the prevision of the frequencies is done throw the following equation:

**Our work**

We decided to focus our attention on the implementation of the third approach. To summarize, this approach needs 6 floats and 8 integers to be stored: , and

SELECTIVITY OF STRICTLY LEFT “<<”

Having the model, it is easy to estimate the selectivity of a query of the kind *SELECT \* FROM T WHERE T.range << [a,b]*. Since we want to estimate the number of ranges that have lower than a, the following integral define exactly what is needed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_}

SELECTIVITY OF OVERLAPS “&&”

The selectivity of a query of the *kind SELECT \* FROM T WHERE T.range && [a,b]* can be estimated by sub setting it in strictly left and strictly right operations. If fact, as mentioned before, . So, using basically the same idea as before, the final formula would be:

=\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CARDINALITY OF JOINS

The selectivity of a query of the kind *SELECT \* FROM T1, T2 WHERE T1.range && T2.*range can be estimated thinking about the 3 dimensional graphs.

Look the following graph. The red and the green axis are respectively the abscissas (lower bounds) and the ordinate’s (upper bounds) ones, while the blue one represents the frequency of ranges value. The two plans represent two easy fitted models that describe the distribution of the ranges of two different columns(T1 and T2). In real scenarios, these two plans would be surfaces, estimated with the zero inflated negative binomial model. A point red plane represents the expected frequency in T1.range of the range .

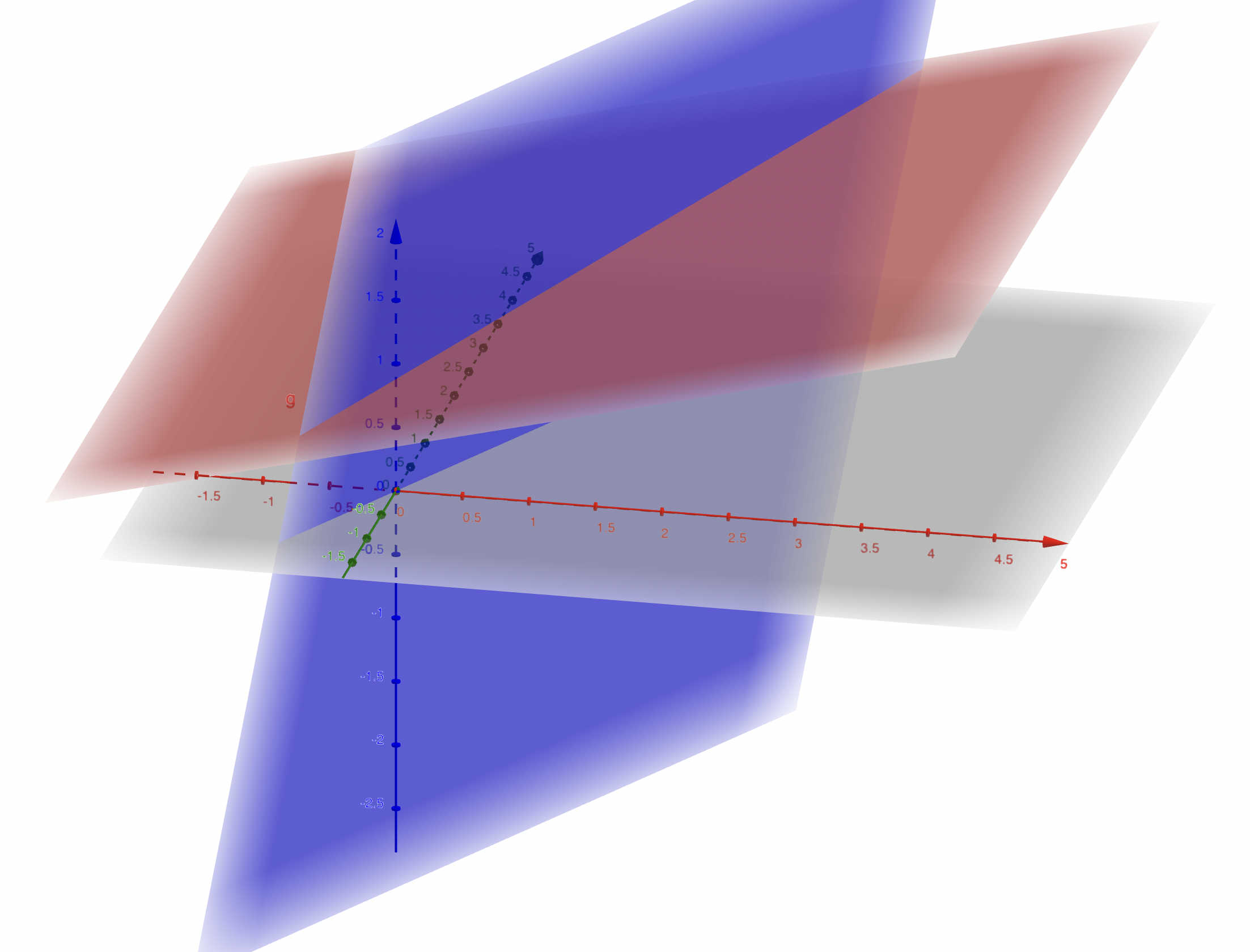


Figure 2.5. The blue plane is the estimation for the first column of range type, the red one for the second.

Our scope is to find how many elements will be returned by the join of the columns from where these two planes are generated. From now on, we will speak about the plans as they were the original columns.

First, let’s discuss the easiest overlap query: *SELECT \* FROM T1, T2 WHERE T1.range && T2.range AND T1.range=[a,b] and T1.range=[a,b].* In our representation, is a point in the space of LB and UB. The cardinality of the join of the two plans will be the moltiplication between the values of the two planes in the point A, respectively and . In fact, for each row coming from the red column there will be rows with the same range value in the blue column. So, .

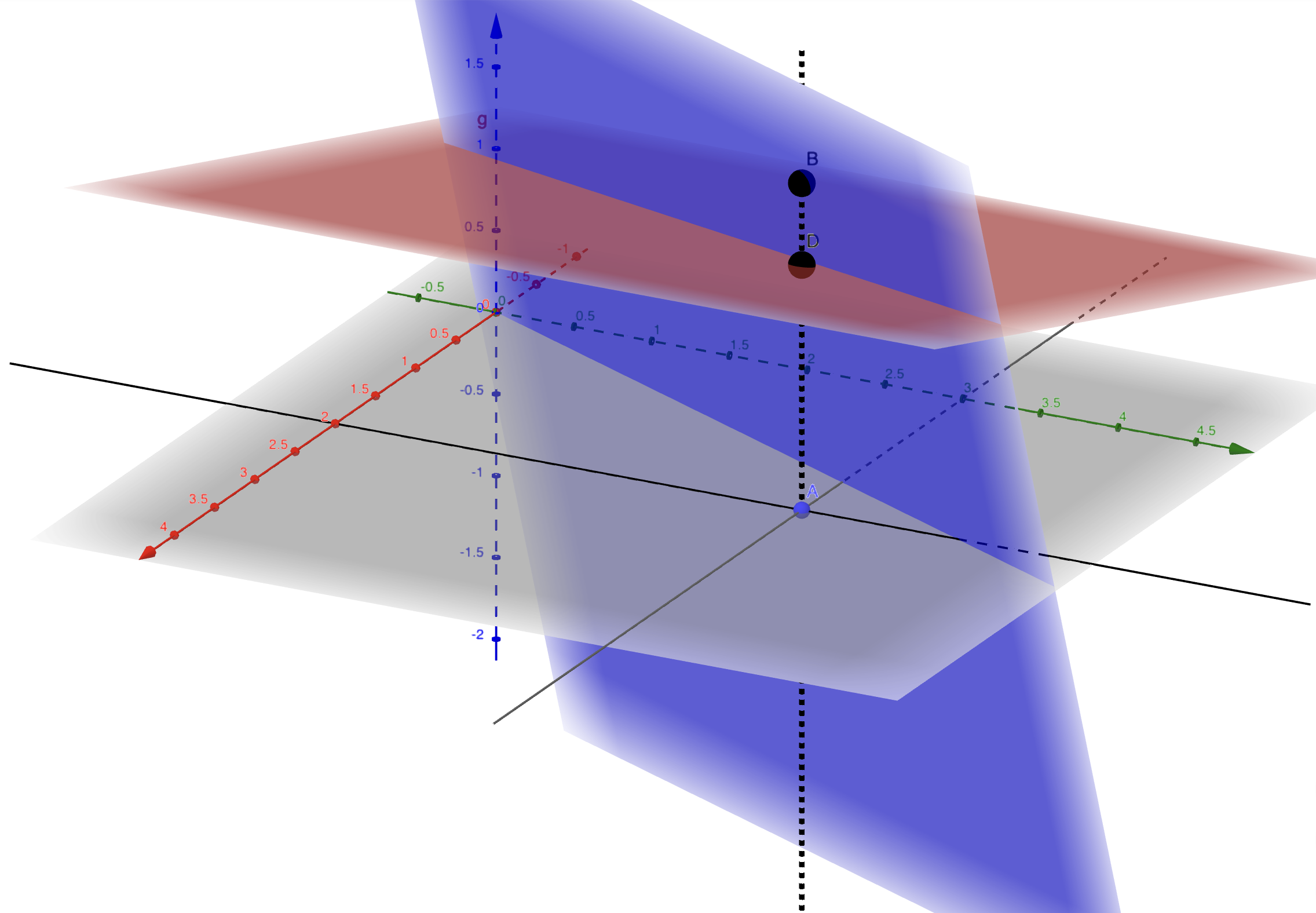


Figure 2.6. Representation of join on a single range value.

The same idea can be used for a query like *SELECT \* FROM T1, T2 WHERE T1.range && T2.range AND “T1.range and T ends in a”.* Instead of simply multiplying the punctual values, we must calculate the integrals of the two planes along the plane parallel to the dimension passing throw , from the lowest to the greatest . We are simply expanding the previous idea to a bidimensional problem. The idea is to sum all the cardinality of the punctual join between the blue and the red columns that fall into the black line in figure 2.7. For this reason, it can be called “one-dimensional” join, since it is made on one dimension, expanding the zero-dimensional punctual approach. So,

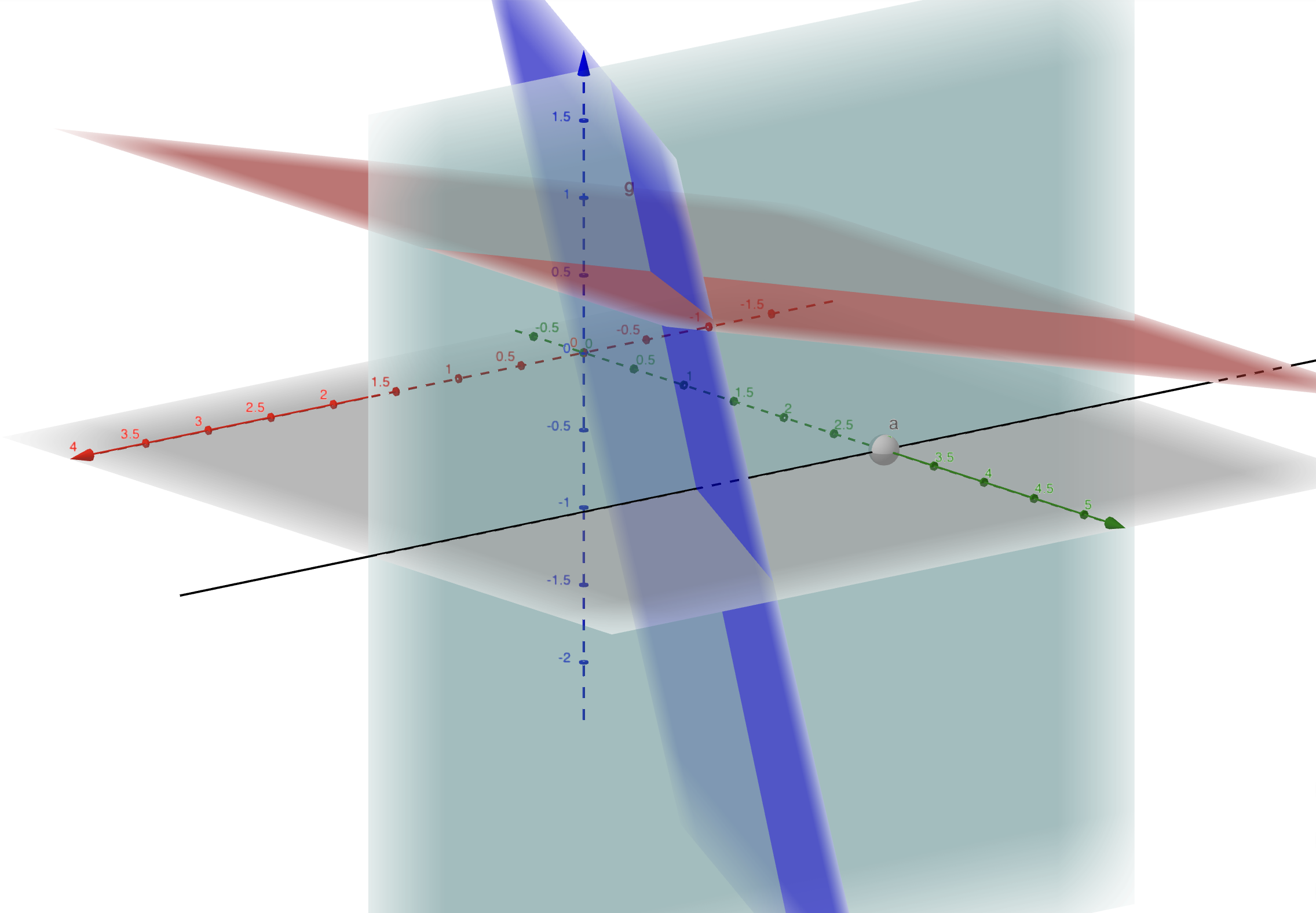


Figure 2.7. The light blue plane is the one along which to calculate the area under the blue and the red planes. These areas will be the estimations of the number of range values that ends in a certain range bound call .

Having seen the two precedent results, it can be reasonably said that the result of the query *SELECT \* FROM T1, T2 WHERE T1.range && T2.range* can be thought as a “bi dimensional join”, where we can drop all the constrains (being over point in the first example, being along the plane that passes throw in the second) and simply calculating the integral of the product of the two planes from the lowest to the greatest and from the lowest to the greatest :

This integral is similar to the one already solved for the selectivity, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

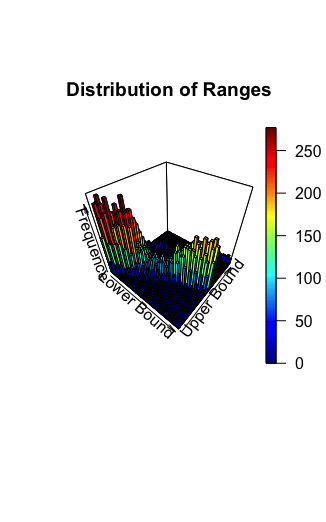
BENCHMARKING

a generalized linear model generating four different scenarios. For more information about the random generation of data in each scenario see the documentation.

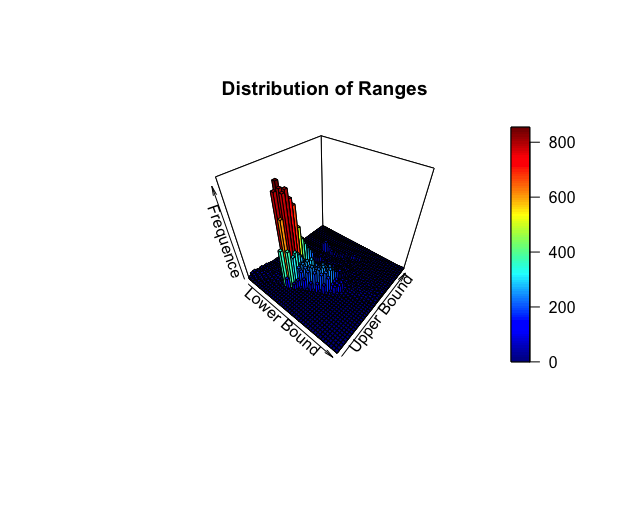
After every modification of the data, instead of rebuilding the model, it would be possible have a naïve estimation of the selectivity simply by allocating a new rate for the number of tuples in the column: . Once applied the model to the data, the resulting estimation of the selectivity would be .

The first scenario (called **Scenario 0**) has already been shown. The other three are following:

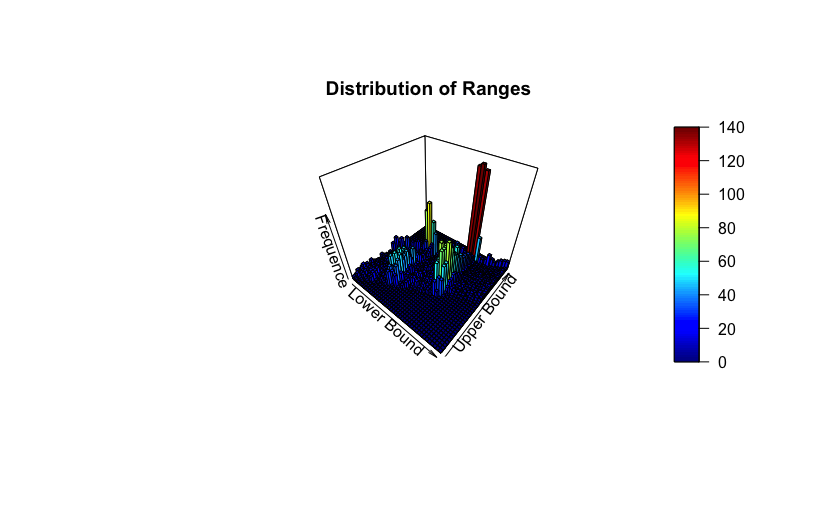
**Scenario 1:**



**Scenario 2:**



**Scenario 3:**



Since the main scope of this work is to find a way to optimize previsions of selectivity and cardinality, we decided to give more importance to find a good model than to being able to implement it in PostgreSQL in a short period of time. For this reason, we have not ………

**Conclusions**

In order to save more allocation space for Join cardinality estimations, we have provided two different ideas to be implemented. The time of fitting of the model or to search for the parameter of the distribution would be longer than the time needed to ……..

TO BE WRITTEN AT THE END