Constrained Random Exploration

and Archimedes Maps

Bernard Beauzamy

October 2013

Quite commonly, we want to explore some situation, but we do not have enough resources for a complete exploration: time, money, people, may be insufficient.

Here are some simple examples:

– Testing the production of a factory;

– Testing the pollution in a field;

– Exploring some land for possible discoveries (water, gas, and so on).

Typically, in such cases, one will use some random procedure to determine the places to be searched. Depending on the situation, this procedure will not be completely random. Indeed, usually, two requirements appear:

– The search should cover as many aspects as possible;

– The same place should not be searched twice.

This is what we will call "constrained random exploration".

A simple explicit example is that of the Competitive Game 2013-2014; in this game, a company wants to test some products (cylinders), so they pick up some cylinders at random, and decompose them into layers. Each layer is a disk, made approximately of 80 square cells. We want to investigate 10 cells among the 80, at random. Of course, it would be a waste of time if the same cell was chosen twice.

We will take this example in order to illustrate the general theory. We have N objects (here $N = 80$ cells), among which we want to choose K (here $K = 10$), avoiding repetitions, and in a way which is as "explorative" as possible, whatever this means.

Let us insist first upon the fact that, quite possibly, if the cells are drawn randomly (with no repetition), one half of the disk is likely to receive more investigation than the other half. Pure randomness does not lead to equilibrium, but rather to desequilibrium; see the book [MPPR].

The best approach, in such a situation, is as follows:

Step 0: Write a list of all possible cells, that is a numbering from 1 to N

To do so might not be easy, especially if the set of all cells has a strange shape, is 2 or 3 dimensional. Here is an example; we have a disk, of radius 5, and the cells are numbered from the coordinates of the left bottom corner.

```
Sub macro1()
Dim x As Integer
Dim y As Integer

Dim k As Integer
Dim n As Integer
n = 1
Dim c1(1 To 100) As Integer
Dim c2(1 To 100) As Integer
Dim numero(-5 To 5, -5 To 5) As Integer
For x = -5 To 5
For y = -5 To 5
If (x + 1 / 2) ^ 2 + (y + 1 / 2) ^ 2 <= 25 Then
numero(x, y) = n
c1(n) = x
c2(n) = y
Sheets(1).Cells(x + 10, y + 10) = n
n = n + 1
End If
Next y
Next x
End Sub
```

Here is the resulting list:

			1	2	3	4			
	5	6	7	8	9	10	11	12	
	13	14	15	16	17	18	19	20	
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
	61	62	63	64	65	66	67	68	
	69	70	71	72	73	74	75	76	
			77	78	79	80			

Figure 1: numbering of the cells in a disk

When step 1 is performed, then there are two possibilities:

A. Using distinct sublists

Then the next step is:

1. Step 1: Decompose the list in K sublists, of almost equal length

In our case, we have a list of 80 cells, and we divide it into 10 lists of 8 items each, so the division is exact.

In the general case, we write the Euclidean division:

$$N = qK + r$$

with $r < K$, and we would have r lists of length $q+1$ and $K-r$ lists of length q. So, the lists are almost of same length.

When this is done, the next step is:

2. Step 2: Draw at random a number in each sublist

This is easy to: if the length of the sublist is q, the instruction:

$$n = 1 + int\left(\left(q-1\right) * rnd()\right)$$

will return a random interger between 1 and q.

3. **Step 3: Pick up the corresponding number in each sublist**

In our case, here are the numbers which were chosen:

8 12 22 29 35 48 56 60 71 73

and here is the resulting disk:

			1	2	3	4			
	5	6	7	**8**	9	10	11	**12**	
	13	14	15	16	17	18	19	20	
21	**22**	23	24	25	26	27	28	**29**	30
31	32	33	34	**35**	36	37	38	39	40
41	42	43	44	45	46	47	**48**	49	50
51	52	53	54	55	**56**	57	58	59	**60**
	61	62	63	64	65	66	67	68	
	69	70	**71**	72	**73**	74	75	76	
			77	78	79	80			

Figure 2: The cells chosen after constrained random choices

With this approach, we do not have to care about the chosen cells being distinct, since they belong to different lists.

We note that what we did - enumerating the cells as a single list, and then dividing this list into sublists of equal (or almost equal) number of cells - is in fact the construction of an Archimedes map of our set (here the circle). We recall that an Archimedes map of a set is a division into subsets of equal area (or equal resources).

There are of course many ways to draw an Archimedes map of a disk; see the book [AMW] for the presentation of some of them. Establishing a list and then dividing it may not be the most appropriate one. In some cases, one may prefer concentric circles, or parallel bands, and so on.

B. Drawing distinct numbers in a single list

Still, one may want to draw distinct numbers in a single list. Let us see how to proceed. We still consider the case of a list of 80 items, in which we want to draw 10 distinct items.

Step 1

Draw K numbers at random, successively between 1 and N, 1 and $N-1$, …, 1 and $N-K+1$

This is done the following way (with Ktot=K)

```
Dim k As Integer
Dim n(1 To Ktot) As Integer

For k = 1 To Ktot
n(k) = 1 + Int((Ntot - k) * Rnd())
Sheets(1).Cells(k + 1, 1) = n(k)
Next k
```

This is the numbers we get:

56 42 45 23 23 58 2 55 58 50

These numbers do not need to be increasing, nor to be distinct.

Step 2

Choose the first number in the list.

Here, we chose 56, so we point out the number 56 in our list from 1 to 80.

Step 3

Make a new list, where this number has been crossed out.

In our case, the new list will be: 1,…,55,57,…,80. It contains 79 items.

Step 4

Choose the second random number obtained above, in our case 42, and extract the 42th number from the list obtained in step 3.

Continue inductively until all random numbers have been used. Since one works at each step with a new list, from which the previous numbers have been removed, one can be sure to obtain distinct numbers at the end.

Here is the VBA code performing these tasks:

```
Sub macro2()
Dim k As Integer
Dim n(1 To Ntot) As Integer
For k = 1 To Ktot
n(k) = Sheets(1).Cells(k + 1, 1)
Next k

Dim list() As Integer
```

```
        ReDim list(1 To Ntot) As Integer
        Dim newlist(1 To Ntot) As Integer

        For k = 1 To Ntot
        list(k) = k
        Sheets(1).Cells(k + 1, 5) = list(k)
        Next k

        Dim p As Integer
        For p = 1 To Ktot
        For k = 1 To Ntot - p
        Sheets(2).Cells(p, 1) = list(n(p))
        If k < n(p) Then newlist(k) = list(k)
        If k >= n(p) Then newlist(k) = list(k + 1)
        Sheets(1).Cells(k + 1, p + 5) = newlist(k)
        Next k
        ReDim list(1 To Ntot - p) As Integer
        For k = 1 To Ntot - p
        list(k) = newlist(k)
        Next k
        Next p
End Sub
```

Here is the final list of numbers we obtain:

56 42 46 23 24 63 2 61 66 55

and here are the corresponding cells in the disk:

			1	**2**	3	4			
	5	6	7	8	9	10	11	12	
	13	14	15	16	17	18	19	20	
21	22	**23**	**24**	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	**42**	43	44	45	**46**	47	48	49	50
51	52	53	54	**55**	**56**	57	58	59	60
	61	62	**63**	64	65	**66**	67	68	
	69	70	71	72	73	74	75	76	
			77	78	79	80			

Figure 3: The cells chosen after non-constrained random choices

We see in this example, as we said earlier, that the exploration is not as regular as in the previous case. Here, the cells below the first diagonal receive more attention.

Reference

[AMW] Bernard Beauzamy : Archimedes' Modern Works. SCM SA, août 2012.