APL Healthcare Problem Set Solutions

Armando B. Cruz Hinojosa

July 24, 2017

Abstract

In this document it will be shown all the procedures and logic implemented in the Healthcare Problem Set's solutions for the 2017's Dyalog APL Contest.

1 Task 1

1.1 Problem Description

Write a function named Projection to model this patient population

In a group of disease sufferers, 120,000 people are being treated for a particular strain of the disease.

- On average, 2.5 of the total treated patient population will go into remission (that is, they will no longer require treatment) each month after treatment.
- On average, a further 3.25 will die each year from the disease or due to complications resulting from it.

The treatment for this disease consists of three drugs A, B and C. Patients start on drug A and move through B and on to C depending on their response (or lack of it).
At time zero (T0) the group has 70,000 patients taking drug A, 30,000 taking drug B and 20,000 taking drug C; consider remission and death rates to be identical for all 3 drugs.

Projection has the following syntax:(sa sb sc rem dec) ← years Projection a b c

The right argument consists of:
a, b, and c − the number of patients currently taking drugs A, B, and C respectively
The left argument consists of:
years − the number of years to run the projection
The result represents the population at the end of the projection and consists of:
sa, sb and sc − the number of surviving disease sufferers taking drugs A, B and C respectively
rem − the number of patients in remission
dec − the number of deceased patients.

1.2 Solution

As first approach, is obvious that the modeling of the Drugs can be solved in an iterative or even recursive way. In principle, this idea seems pretty good (and tempting), all we would have to do is create a loop and transform this problem into a simple implementation.

Nevertheless, APL was not designed for looping but for large calculations, and as we will see in Task 3, when we are working with large amounts of data, Iterative algorithms seems to never end.
Why? Simple, almost all looping algorithms had upper bounds in computational complexity of $O(n)$ or $O(n^2)$ meaning the bigger the input, the longer it takes to finish the computations.
What we want, then, is an algorithm that no matter the size of the inputs, it will always run in more less the same time. In other words, we want an algorithm with complexity upper bound $O(1)$.

We will achieve this throw mathematics, and dynamic optimization (DP), since is my bachelor's area and is more oriented to my thought process. Is good to notice that a mathematical solution, *A formula* in this case, represents more benefits to the problem than just a simple "Run in time" propertie.
Being generated the function, we will be able to extend this problem and the solution to the Calculus field, where we can work with derivatives, integrals, limits, and other tool that will let us know things such as:

- The instantaneous rate of change of the population
- The limit as we go closer to LONG periods of time
- When does the population reach the maximus and minimous

and a bunch of useful information that the industrie and the guys in the lab just love. I know it will be hard, but the benefits afford it.

First we must establish a temporal order in which the events happen, in order to *avoid ambiguities* and achieve a correct implementation.
We have two main events for every population:

- Every month a percentage of patients go into remission.
- Every year a percentage of patients dies *due the disease or complications resulting from it.*

Now let's suppose without generality lost, that every *half month*, the remission diagnostics are executed and the percentage of patients is removed from the population. Similarly we will suppose the deaths occure at the end of the year since our simulation ends by that time.
The problem explicitly tells us that the patients dies *due the disease or complications resulting from it*, this way, it's *impossible* that a patient in remission dies *due the disease or complications resulting from it*, with this in mind, we can say that only the sick population will die every year.

Notice that the events, **Going into remission** and **Not going into remission** are mutually exclusive, and so their probabilities add up to 1.

Now is obvious that if a percentage of patients go into remission, the ones who are still taking drugs are the *complement* of the **Going into remission** percentage. Let's call this complement x, logically the first percentage (the **Going into remission** one) will be $(1-x)$.
Just as the events **Going into remission** and **Not going into remission**, the events **Die** and **survive** are mutually exclusive, and their probabilities add up to 1. Let's call the **survive** percentage y, and it's complement **die** $(1-y)$.

REMAINING POPULATION
Since every half month, a percentage of the population goes into remission, by the end of each month there will be an x percentage left of the population, we can express this in a recursive way as follows.

$$P_{i_m} = P_{i_{m-1}} x$$

Where P_{i_m} is the current population of the i th group in the m th month. And $i \in \{a, b, c\}$.
This recursive definition is the well known *Geometric progression* definition, in wich each term of the serie is obtained by multiplying it by a constant number called *progression ratio*.
Knowing that for geometric serie $\{a_n\}$, the n th term is $a_n = a_0 r^n$, where a_n is the n th term and r the progression ratio. We can prove that by the end of the 12th month the population will be $P_{i_{12}} = P_{i_0} x^{12}$.

As we know, every end of the year a percentage of the ill population dies and a percentage y of patients survives. So by the end of the year there will survive a y percentage of the remaining population by the of the last month.
So, by the end of the 12th month the population will be

$$P_{i_{12_1}} = P_{i_0} x^{12} y$$

It is good to notice that this population $P_{i_{12_1}}$ *will be the initial population* for the second year, and that by the end of the second year we end up with

$$P_{i_{24_1}} = P_{i_{12_1}} x^{12} y$$

Indeen, for any year, the initial population will be the last year's final population, in other words we can say that

$$P_{i_n} = P_{i_{n-1}} x^{12} y$$

Where $n = 12m$ is the number of years.
Again, we notice that this definition give us a *Geometric progression*, so by we the end of the n th year the surviving population will be

$$P_{i_n} = P_{i_0} (x^{12} y)^n \tag{1}$$

DECEASED POPULATION
Notice that the remaining population P_{i_n} by the end of a year n is the y percentage of the final

month population, that *survived* death that year.

$$P_{i_n} = P_{i_{n_f}} y \tag{2}$$

Where $P_{i_{n_f}}$ is the final month population.

Now, the deceased population, is the $1 - y$ percentage of the final month population, that *died* that year (As we show above, both **Die** and **survive** events are mutually exclusive).

$$D_{i_n} = P_{i_{n_f}}(1 - y) \tag{3}$$

Where D_{i_n} is the deceased population by the end of a year n. Then

$$D_{i_n} = \frac{P_{i_n}}{y}(1 - y)$$

By substituting (2) in (3). And by (1)

$$D_{i_n} = \frac{1 - y}{y} P_{i_0}(x^{12}y)^n \tag{4}$$

This formula gives us the deceased population by the end of a year n, nevertheless the *deceased population* is the sum of all the individual *deceased patients by the end of all years*, in other words

$$D_i = \sum_{x=1}^{n} D_{i_x} \tag{5}$$

$$D_i = \frac{1 - y}{y} P_{i_0}(x^{12}y)^1 + \frac{1 - y}{y} P_{i_0}(x^{12}y)^2 + ... + \frac{1 - y}{y} P_{i_0}(x^{12}y)^{n-1} + \frac{1 - y}{y} P_{i_0}(x^{12}y)^n$$

$$D_i = \frac{1 - y}{y} P_{i_0} \left[(x^{12}y)^1 + (x^{12}y)^2 + ... + (x^{12}y)^{n-1} + (x^{12}y)^n\right]$$

$$D_i = \frac{1 - y}{y} P_{i_0}(x^{12}y) \left[1 + (x^{12}y)^1 + (x^{12}y)^2 + ... + (x^{12}y)^{n-2} + (x^{12}y)^{n-1}\right]$$

Now let's introduce the next identitie

$$(1 - z)(1 + z + z^2 + z^3 + ... + z^{r-1} + z^r) = 1 - z^{r+1} \tag{6}$$

Let $z = x^{12}y$ and $r = n - 1$. Then by (6)

$$D_i = \frac{1 - y}{y} P_{i_0}(x^{12}y) \left[\frac{1 - (x^{12}y)^n}{1 - x^{12}y}\right]$$

Finally

$$D_i = P_{i_0} \frac{1 - y}{y} \left[\frac{x^{12}y - (x^{12}y)^{n+1}}{1 - x^{12}y}\right] \tag{7}$$

POPULATION IN REMISSION
We managed to obtain both the Remaining (1) and Deceased (7) population formulas. Since there are no more patients that those who are in remission, and since the death rate doesn't affect them and our population acts as an isolated system (no incomes or outcomes), we can calculate this by substracting from the original population P_{i_0} both P_i and D_i.

$$R_i = P_{i_0} - (P_i + D_i) \tag{8}$$

1.3 Domain Restrictions

Now let's define the domain of our function. Do we want the domain to be continous, or discrete?, Do we want to extend the function to the negative years or not?
It's good to notice that Geometric series are defined in positive integer indexes, and our identitie (6) is written as the sum of consecutive positive integer powers of a base.
Is no surprise, then, that in order to our function to work propertly, the values of our domain n should be positive integers as well, being said and since modeling the patient projection to, for example -2 years is iloggical, let's define the domain as $n \in \mathbb{N}$.

What should the function do, then, if the user gives an $n < 0$?
In that case the function should return the default value n = 0.
And if the user gives a non integer n?
In that case the function should return the projection to the closest greater integer.

Note: The reader is free to change these events as pleased.

2 Comments

As we can see, this solution just need to execute *once* each formula, and will not need of *any looping* or nonoptimal method to return the answer. If you still had questions about the formulas (Bracketing, Order or Logic) or if you want to check the empirical modeling in a spreadsheet, remember to visit:

https://github.com/Aleph-GORY/Dyalog-APL-Challenge-2017