**Dan's**

Cocomo model can be simplified as $E_n = A_n * S_n^{B_n}$

At any time *t*

$$E_t = A_t * S_t^{B_t}$$
$$= A_t * (S_{t-1} + \Delta S_t)^{B_t}$$
$$= A_t * [\sum_{i=0}^{B_t} (C_{B_t}^i * S_{t-1}^i * \Delta S_t^{B_t - i})]$$

Some questions left by Dan are:

1. At time *t*, $S_t$ is already known, because it's the lines of code that is produced until time *t*. What's the purpose of decomposing it to $S_{t-1}+delta(S_t)$ ?

2. If I understand correctly, the essence continuous cocomo is constantly changing A and B. Should we try to estimate $A_t$ and $B_t$ for each period? Could Dan provide more input?

**Here is another model.**

I make strongest assumptions to get simplest model. The assumptions can be relaxed later.

The general steps in cost estimation (including those with Cocomo, but not restrict to Cocomo) are:

1. User estimate size (i.e. size is always exogenous to any cost estimation model).

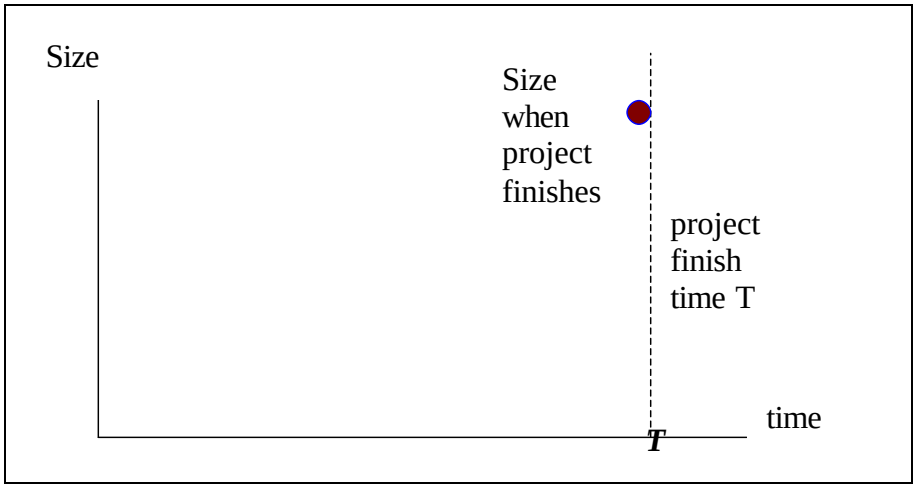
2. The model tells the user nominal effort (e.g. in person-month) based on formular like ***Effort = f (Size)***.

3. Convert nominal effort to calendar time based on the number of developers in the project. Or given project deadline, compute number of developers required.

Assume the number of developer working on the project is constant, and assume every developer works same hours per day. At the very beginning of the project, the information available is:

(a) The size of the project when it's finished. This is always exogenous to the model. It's user's responsibility to determine the correct size based on requirements or what so ever.

(b) The day the project will finish. This is estimated from cost estimation model chosen by user (e.g. Cocomo).

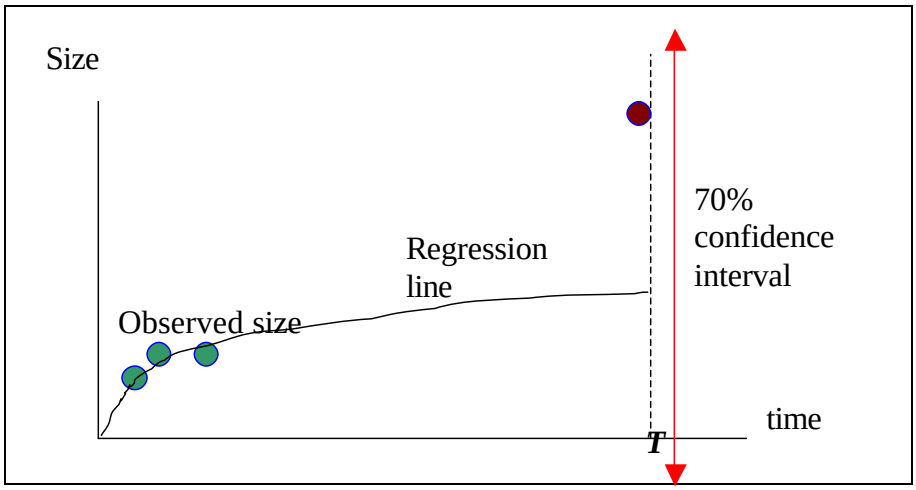



What cost estimation model doesn't tell us is how the project will proceed from the origin to the red dot, and when the project will not meet its deadline.

We can collect size at the end of each period. So at time t, we have following observations:

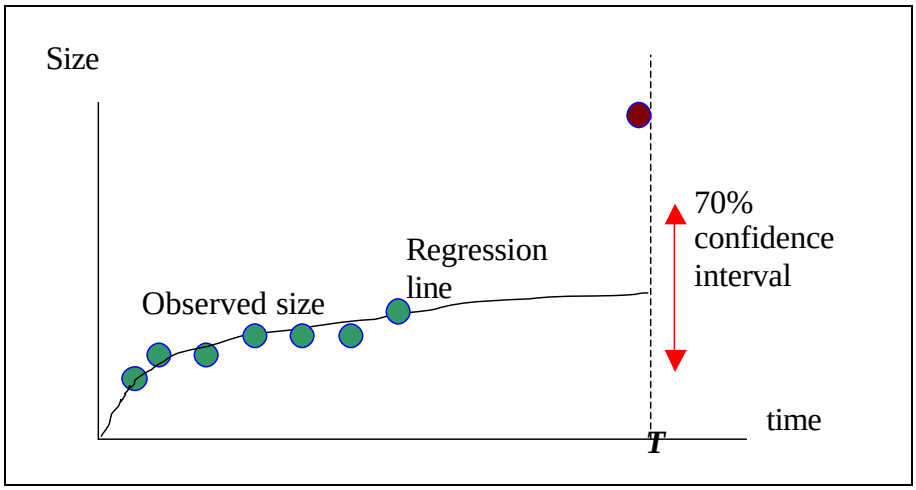
$(Period_0, Size_0)$ $(Period_1, Size_1)$ … $(Peroid_t, Size_t)$

We can fit them to some functional form, say ***Size = a * (Period – b) ^ c + d***, and then use the regression to forecast what size will be at time ***T*** (project finish time). We also compute 70% (can be any number) confidence interval of the forecast. If the red dot lies within the interval, we do nothing; otherwise, the original cost estimation model may be incorrect (over-estimation or under-estimation).




At the beginning of the project, we don't have too much observation, but we are extrapolating size at the end of project. 70% confidence interval must be very large. So the red dot will almost always lie within the interval.

As time goes by, more and more observations are available. Size prediction will become more and more accurate. Following example suggests that original cost estimation model might be wrong, and it's quite possible that the project will miss the deadline.




Some consequences:

1. The model is agnostic to cost estimation methods. For large projects, Cocomo can be plugged in. For student projects, we can use PSP to estimate effort.

2. The horizontal axis is time. In large projects when developer effort distribution is pretty uniform, it can be calendar days. In student projects, it can be active time measured by hackystat.

3. Regression formula is flexible. It can be of any functional form.

4. If everyone thinks this model is in the right direction, maybe Dan can make it fancier.