A New Approach to Specifying and Sampling Correlated Durations in Schedule Risk Analysis.

Tony Welsh, Barbecana Inc.

(Paper presented to PMI Scheduling Community of Practice, New York, May 8th 2012)

Abstract

To fully describe the relationship between two or more correlated random variables, one needs to specify the multivariate distribution function. This is beyond what is reasonable to expect a project manager to estimate subjectively. Schedule risk analysis tools therefore typically require the user to specify just the distributions of the individual durations and the correlation coefficients between them. Some systems require this to be specified by means of a covariance matrix, allowing correlations between any number of tasks. This presents a potential problem because not all matrices are valid covariance matrices. Other systems allow any one duration to be correlated with just one other duration. The challenge is to produce samples which exhibit the correct means and standard deviations, and the correct correlation coefficient, while also preserving the shape of the specified distributions, all based upon minimal user input. A technique is presented in which the user can specify the correlation between each duration and one of a set of user-named exogenous variables, but which does not require him to specify anything about the distribution of these exogenous variables and which cannot result in invalid covariance matrices. Results are presented showing a close correspondence between user requirements and sampled results.

Overview

The critical path method (CPM) using single-point estimates of task durations has been the standard method of creating project plans for over half a century. This in spite of the fact that for all that time it has been known that it fails to recognize the uncertainty attached to any projections of the future, and in particular its estimates of the project completion date and other key dates is always biased towards the optimistic. PERT (Program Evaluation and Review Technique) was an early attempt to model this uncertainty but is flawed in many ways, including its inability to take into account more than one path, which means that it does not address the bias mentioned above. Schedule risk analysis (SRA), using Monte Carlo simulation, is now the accepted way to model uncertainty in project networks, and indeed is mandated for many US Government contracts.

It is often required to model correlations between the duration estimates for different tasks, and most SRA software provides some capability to do this. Some systems permit pairs of tasks to be correlated directly with each other, but subject to restrictions. (Typically, if task A is correlated with tasks B and C it is not possible to also specify the correlation between B and C.) This presents a problem if tasks A, B, and C depend upon some outside factor like the weather and are therefore correlated in a symmetrical way, i.e. the correlation between each possible pairing of tasks is the same, then this cannot be represented. If the correlation coefficient between A and each of the others is 0.5 then the correlation between B and C is only 0.25 and the symmetry is lost.

Another problem is that it is difficult to correlate two durations if their distributions are different, for example if one is skewed left and the other is skewed right.

More recently the idea has been proposed, for example by David Hulett (1), that factors which affect multiple tasks should be explicitly recognized and modeled. He calls these "drivers," and the effect of each driver is interpreted as a multiplicative factor in arriving at the value of the duration. These factors typically vary around the value 1, so that for example a driver value of 1.2 would represent a 20% increase in duration over what it would otherwise be,

The solution proposed here seems superficially similar to this, in that it makes use of a set of external factors, which I call "correlation sources" to distinguish them from the "drivers" in the multiplicative model described above. However, the motivation behind it and the way it is implemented are both quite different. The primary motivation, as already discussed, is to preserve the symmetry when multiple tasks are partially dependent on a single correlation source or set of correlation sources.

My assumption is that duration estimates are inherently uncertain, and at least some of this uncertainty cannot be attributed to identifiable causes. My solution therefore allows the user to specify both the distribution of the task duration and the correlation coefficient between it and each of any number of sources. This preserves the symmetry of the situation described above, while preserving the ability to model the unsymmetrical case if necessary. As an added benefit, it turns out to be unnecessary to specify anything about the distribution of the sources themselves.

What is a Correlation Coefficient?

Before going on, we should define our terms a bit more closely. There are in fact several kinds of correlation coefficient. The most commonly known is the "**product moment**" or **Pearson** Correlation Coefficient:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

It varies from -1 to +1 and is a measure of the *linear* relationship between two variables. This is important, because a value of zero does not necessarily mean that two variables are unrelated. I copied the following scatter plots (figure 1) from Wikipedia. Each has the Pearson correlation coefficient next to it:

BARBECANA

- The first row shows typical scatter diagrams with correlation coefficients varying from 1 to -1.
- All the scatter diagrams on the second row have a correlation coefficient of 1, so note that it does *not* depend on the gradient. (Though there has to be some variation in both variables, so the middle one is actually indeterminate – zero divided by zero in the above equation.)
- The examples in the bottom row each clearly show some relationship between the two variables, but because this relationship is not linear all have a Pearson correlation of 0. Fortunately, for most practical purposes we can assume that the relationship is sufficiently linear that we won't get examples like these.

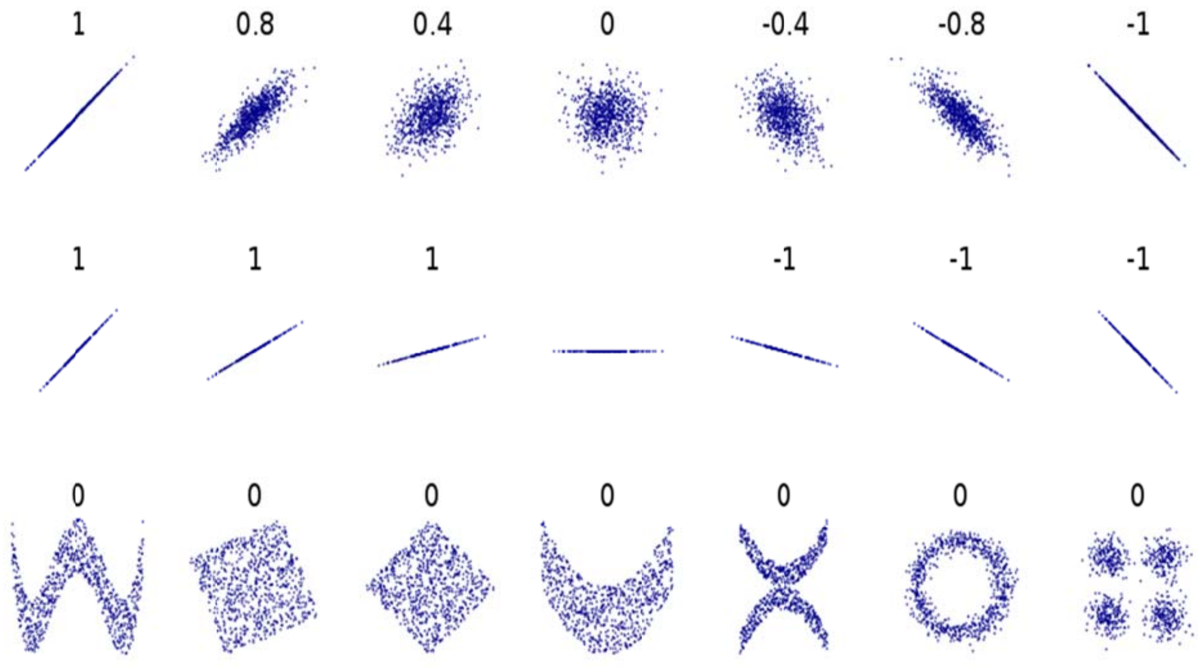


Figure 1.

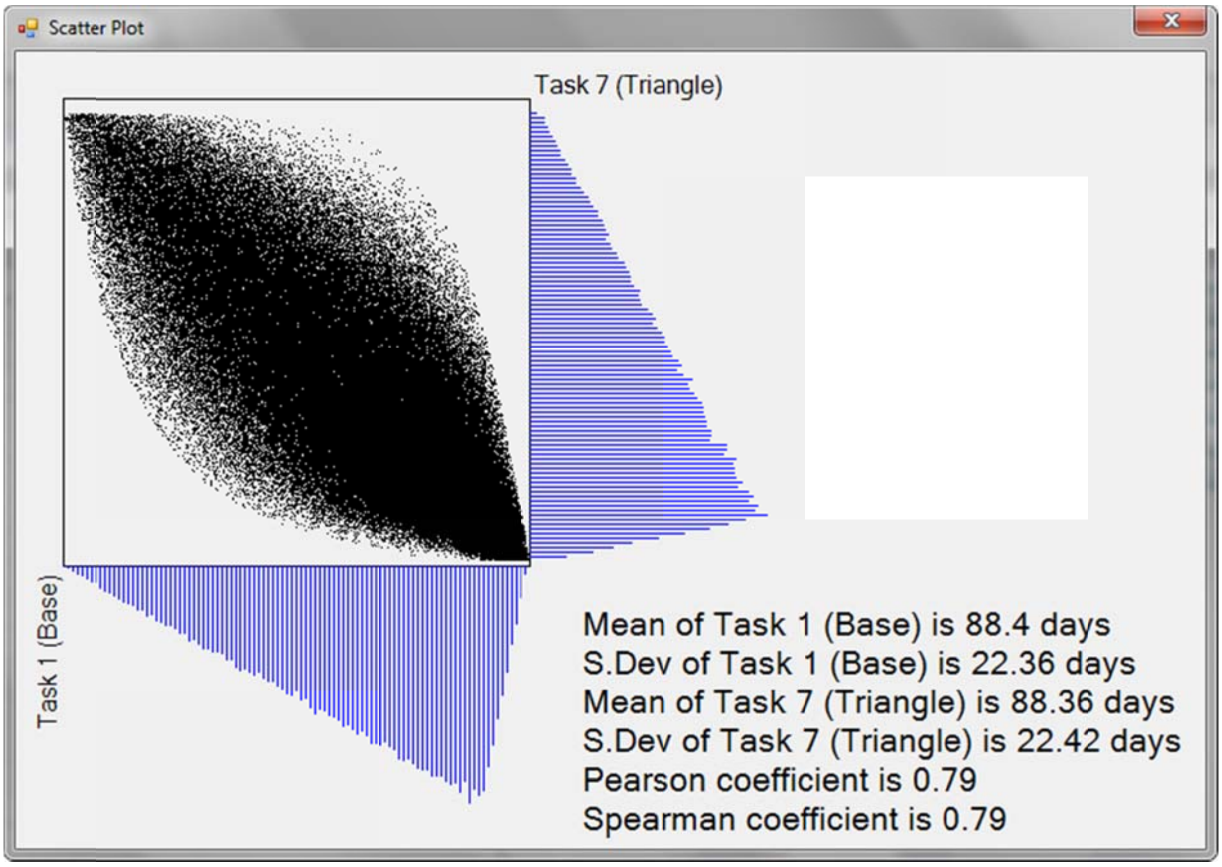
The other measure of relatedness between random variables is the "**rank order**" or **Spearman** correlation coefficient. This is merely the Pearson correlation coefficient of the rank orders of the two distributions. Clearly this is more difficult to compute because it requires storing all the observation pairs, and then sorting them when one is finished. (By contrast, the Pearson coefficient can be computed by accumulating various sums as one produces the samples, thus saving the memory necessary to store the observations and the computation necessary to sort them.)

However, as we shall see, the Spearman coefficient is more readily applicable to the case when the two distributions are not the same.

How Can Variables with Different Distributions be Correlated?

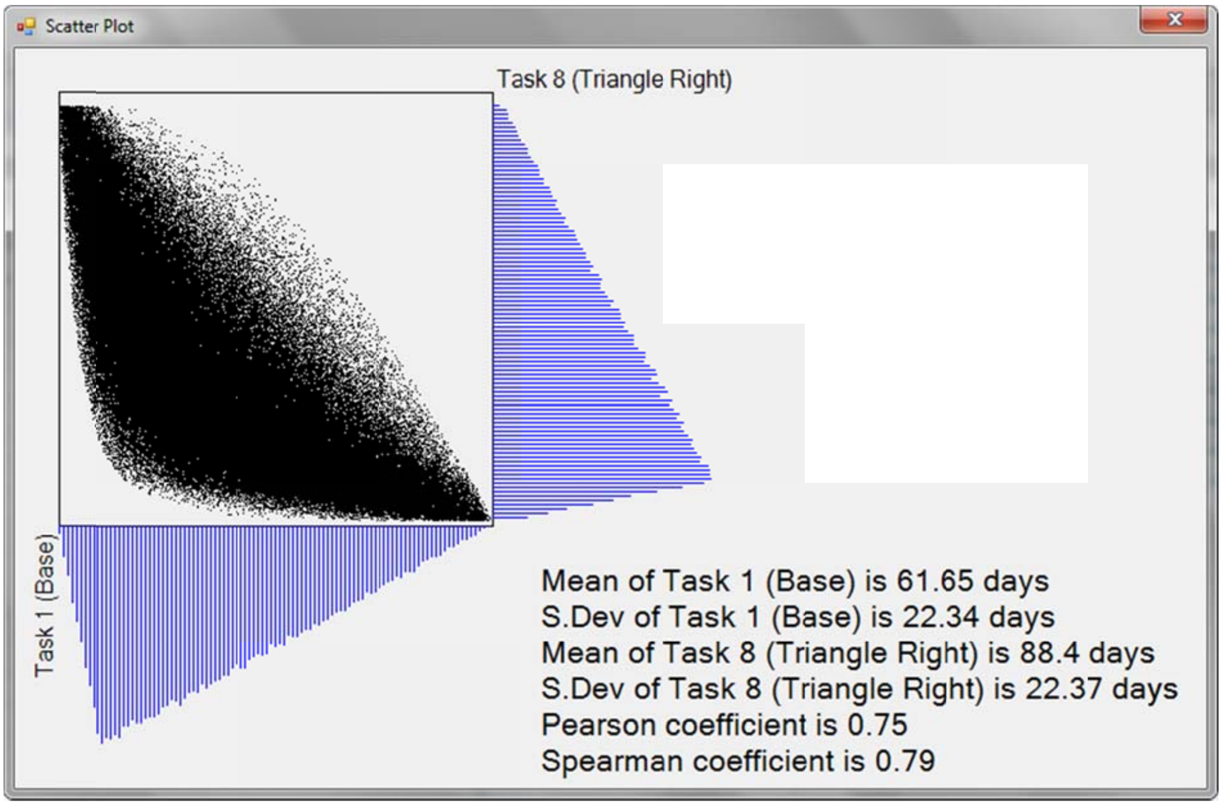
It might seem strange to try to correlate variables with different distributions, but it is perfectly possible as long as we use the Spearman correlation coefficient. (And in practice the Pearson coefficient will also be close.) Here are a few examples from Full Monte.

In figure 2, both distributions are the same (though not symmetrical) and the Pearson and Spearman coefficients are both close to the specified value of 0.8.



Mean of Task 1 (Base) is 88.4 days
S.Dev of Task 1 (Base) is 22.36 days
Mean of Task 7 (Triangle) is 88.36 days
S.Dev of Task 7 (Triangle) is 22.42 days
Pearson coefficient is 0.79
Spearman coefficient is 0.79

Figure 2: Attempt at 80% correlation between two identical triangular distributions.



Mean of Task 1 (Base) is 61.65 days
S.Dev of Task 1 (Base) is 22.34 days
Mean of Task 8 (Triangle Right) is 88.4 days
S.Dev of Task 8 (Triangle Right) is 22.37 days
Pearson coefficient is 0.75
Spearman coefficient is 0.79

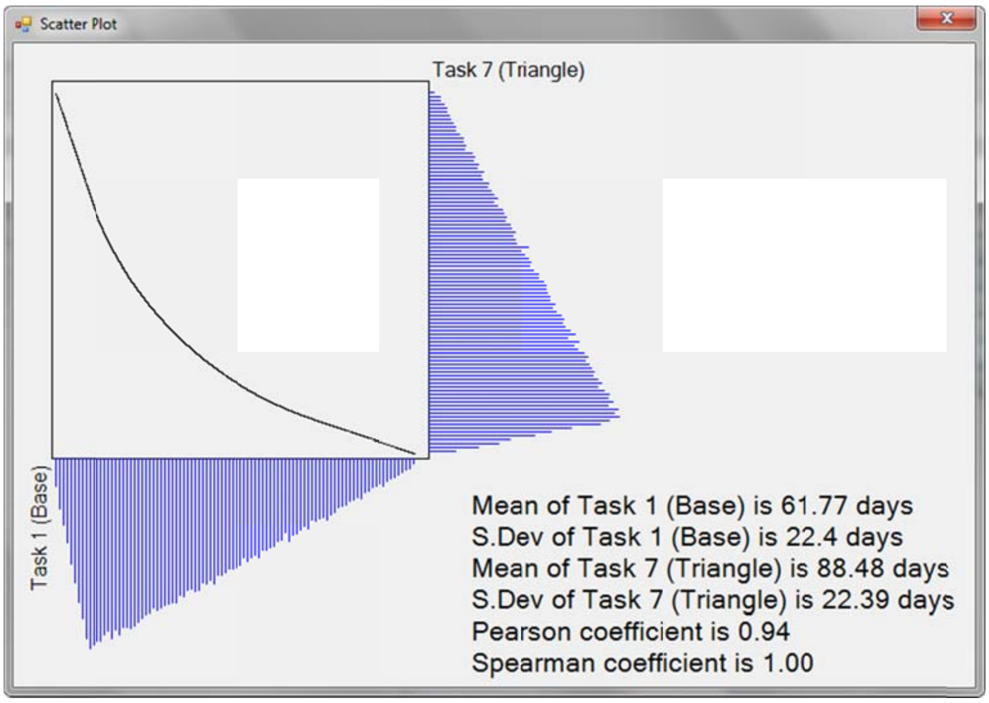
Figure 3: Attempt at 80% correlation between two triangular distributions with opposite skews.

Comparing figures 3 through 5 it can be seen that:

- the shape of the distributions and their means and standard deviations are not compromised by the process to be presented here.
- except when the distributions are identical, the Pearson coefficient is always a little less than the Spearman coefficient, the latter being very close to the theoretical value in all cases.
- in particular, when the distributions are not identical it is still possible to have them perfectly correlated according to Spearman but not according to Pearson.

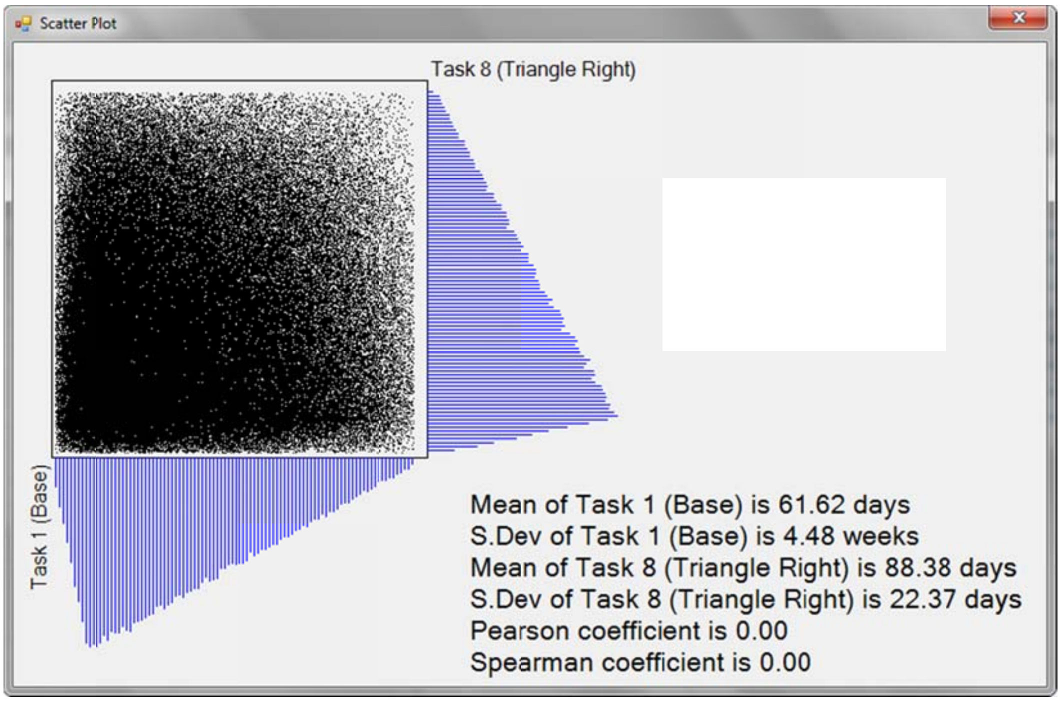
In figure 4, the two variables are totally dependent upon each other as indicated by the fact the scatter plot is a line, but because the distributions are different this line is not straight and consequently the Pearson correlation coefficient is less than 1. The two distributions are as closely related as possible given their respective parameters being different.

(In the live presentation I plan to use the illustrated program in real time and may illustrate more examples.)



Mean of Task 1 (Base) is 61.77 days
S.Dev of Task 1 (Base) is 22.4 days
Mean of Task 7 (Triangle) is 88.48 days
S.Dev of Task 7 (Triangle) is 22.39 days
Pearson coefficient is 0.94
Spearman coefficient is 1.00

Figure 4: Attempt at perfect correlation between two triangular distributions with opposite skews.



Mean of Task 1 (Base) is 61.62 days
S.Dev of Task 1 (Base) is 4.48 weeks
Mean of Task 8 (Triangle Right) is 88.38 days
S.Dev of Task 8 (Triangle Right) is 22.37 days
Pearson coefficient is 0.00
Spearman coefficient is 0.00

Figure 5: The same two triangular distributions, uncorrelated.

BARBECANA

To explain the method being proposed, we first need to cover some theory.

The Correlation Matrix

To fully describe the correlations between a set of N random variables one needs to specify a correlation matrix. This is a symmetrical matrix representing the correlation between each pair of variables something like this:

Task	A	B	C	D	E
A	1	0.5	0.4	0.3	0.2
B		1	0	0.2	0.3
C			1	0.3	0.1
D				1	0.2
E					1

Figure 6: A correlation matrix.

Notice that the cells on the diagonal are all 1 because each represents the correlation of each variable with itself. Also note that since the correlation coefficient between A and B is the same as that between B and A we need only fill in the values in the upper part of matrix, colored yellow. Such matrices are called upper triangular matrices.

This is still quite a lot of values to enter, and it goes up roughly with the square of the number of tasks. (Actually as (n-1)*(n-2)/2.)

There is another problem with allowing users to enter the whole upper triangular matrix, and that is that not all such matrices are valid. It should be intuitively obvious for example that if B and C are both perfectly correlated with A they must also be perfectly correlated with each other. So a matrix like this is not allowed:

Task	A	B	C	D	E
A	1	1	1	1	1
B		1	0.5	0.5	0.5
C			1	0.5	0.5
D				1	0.5
E					1

Figure 7: An invalid correlation matrix.

BARBECANA

(The above case is intuitively obvious, but in general it is not trivial to identify an invalid matrix, as we shall see later.)

In the symmetrical case discussed in the overview we would like to specify a matrix specifying the correlation between each pair something like this:

Task	A	B	C	D	E
A	1	0.5	0.5	0.5	0.5
B		1	0.5	0.5	0.5
C			1	0.5	0.5
D				1	0.5
E					1

Figure 8: Symmetrical correlations between durations of 5 tasks.

This is a lot of data to enter. For this reason, and also to avoid invalid correlation matrices, most systems permit correlations between each task and one other, which is equivalent to entering just the first row of the above matrix:

Task	A	B	C	D	E
A	1	0.5	0.5	0.5	0.5
B		1			
C			1		
D				1	
E					1

Figure 9: What we actually get to specify in some systems, leaving the rest to the software.

The empty cells are in effect completed by the system as follows:

Task	A	B	C	D	E
A	1	0.5	0.5	0.5	0.5
B		1	0.25	0.25	0.25
C			1	0.25	0.25
D				1	0.25
E					1

Figure 10: What we may end up with.

BARBECANA

So, the correlation coefficients between B, C, D, E are only .25 and the symmetry is lost. (By the way, we could have chosen to correlate B with A, C with B, D with C, etc. but the missing values would still default to 0.25.) This value represents the correlation between, say, B and C, which is due to their both being correlated with A, and is the minimum correlation that is possible given these correlations with A. It also guarantees that the matrix is valid, but at the cost of user choice.

Correlation Sources

My solution to the above problem is to correlate each task not directly with other tasks but with one or more outside factors which I am calling correlation sources, and to further assume that:

- The sources are uncorrelated with each other.
- The tasks are uncorrelated with each other except to the extent implied by their correlations with one or more shared sources.

(It may not be obvious, but this does not limit the user's ability to indirectly specify any valid set of correlations between tasks.)

The valid matrices consequently look like this:

Task	I1	I2	I3	T1	T2
I1	1	0	0	0.5	0
I2	0	1	0	0.5	0.5
I3	0	0	1	0	0.5
T1	0.5	0.5	0	1	0.25
T2	0	0.5	0.5	0.25	1

Figure 11: Correlations between 2 tasks and 3 correlation sources.

In figure 11, there are a number of sources followed by a number of tasks across the top and down the side. The upper left part of the matrix has 1's on the diagonal and 0's in the other places, indicating that the sources are not correlated with each other. The yellow cells represent the correlation between the tasks and the sources. Task T1 is correlated with sources 1 and 2, while task T2 is correlated with sources 2 and 3. Because they share a source, they are also correlated with each other, indicated by the pink cells. Values entered by the user are only the four non-zero yellow cells. (I have shown this in full rather than upper triangular form because it makes the next step clearer.)

As mentioned earlier, not all conceivable correlation matrices are valid. In general it is not so easy to spot an invalid matrix. (The technical requirement is that it be positive definite but this is beyond the scope of this paper.)

But with the above assumptions it becomes simple: **the matrix is valid if and only if the sum of the squares of the absolute values of the correlations between any task and its sources does not exceed 1.**

Given that this simple condition is satisfied, it turns out that in order to generate N random variables correlated as specified by the matrix, it is necessary just to generate a set of **N uncorrelated random variables**, after which one can generate the required correlated variables **as linear combinations of the uncorrelated ones.**

The Cholesky Decomposition

But how do we determine the coefficients in this linear combination? They are given by the Cholesky decomposition, which is an algorithm to find a lower-triangular matrix L such that:

A = L L*

Where A is the valid (positive definite) covariance matrix and L* is the transpose of L. The matrix L can be used to transform a set of uncorrelated random variables into a set of correlated ones. (By the way, if the matrix is not positive definite, there will be no solution to this equation.)

(André-Louis Cholesky was a French mathematician. Born in 1875, he was also an army officer and died in 1918 near the end of World War I.)

Finding the decomposition in the general case is complicated (2), but the particulars of the assumptions made above make it much easier. For example, the 5x5 matrix above decomposes into the following matrix (and its transpose):

Task	I1	I2	I3	T1	T2
I1	1	0	0	0	0
I2	0	1	0	0	0
I3	0	0	1	0	0
T1	0.5	0.5	0	0.707	0
T2	0	0.5	0.5	0	0.707

Figure 12: Cholesky decomposition.

BARBECANA

The columns are marked as before, but they actually represent a set of independent random variables not directly with the original sources and tasks. If we multiply L by a vector composed of these uncorrelated variables we get the correlated variables we want:

x' = L x

So, for example, to obtain a duration sample for task T1 (which is on row 4) we merely add .5 times x1, .5 times x2, and .71 times x4, where x1, x2, and x4 are independent random variables.

Note that the 3 sources are identical to the first 3 uncorrelated variables, and in fact x4 and x5 can also be loosely associated with T1 and T2, representing that variation in these task durations which does not depend upon the sources. In fact we can easily compute the coefficient knowing that the coefficients of the sources are the same as the desired correlation coefficients and that the sum of the squares in each row must add to 1.

This leads us to the simple rule for knowing when we have a valid (positive definite) correlation matrix, subject to the assumptions already noted; the squares of the correlation coefficients for each task must not exceed 1. (btw, you may notice that 0.707 is the square root of 0.5. The rule is that the sum of the squares in each row add to 1.)

Finally note that we have no interest in the source variables *per se*. In fact we don't care how the sources are distributed, and the user therefore never has to specify anything about the sources.

Preserving the Distributions.

So, problem solved? Not so fast. The linear combinations above generate variables with the right correlation coefficients, but almost everything else is wrong. For example, we are adding .5 times each of two variables and .7 times another. If both original variables are in the range {0, 1} then the result will be in the range {0, 1.7}, so both the mean and the standard deviation will be different. In fact, nothing in the above indicates anything about the shape or parameters of the desired duration distributions. Before we deal with this problem we must step back a bit and understand two other issues.

First, we must ask ourselves what distributions we should use to sample the uncorrelated variables referred to above. These are destined to be combined in various linear combinations. Linear combinations of random variables tend towards being normally distributed whatever the individual distributions are. (For example two uniform distributed variables added together gives us a triangle, another one added to that starts to round off the corners, and as we add more the distribution gradually becomes normal.) So, in order to know what the distributions of the linear combinations are, it pays to start with normal distributions and assume the resulting linear combination is also normal.

BARBECANA

Secondly, we must understand how random variables are sampled by inversion. (There are other ways to sample, but we need to use inversion in order for this to work.) For this purpose, a probability distribution is represented by its cumulative distribution function,

F(x)

Which represents the probability that the actual value will not exceed x. This function naturally goes from 0 for sufficiently low values of x to 1 for sufficiently high values of x. It is monotonic increasing and generally looks like an S-Curve:

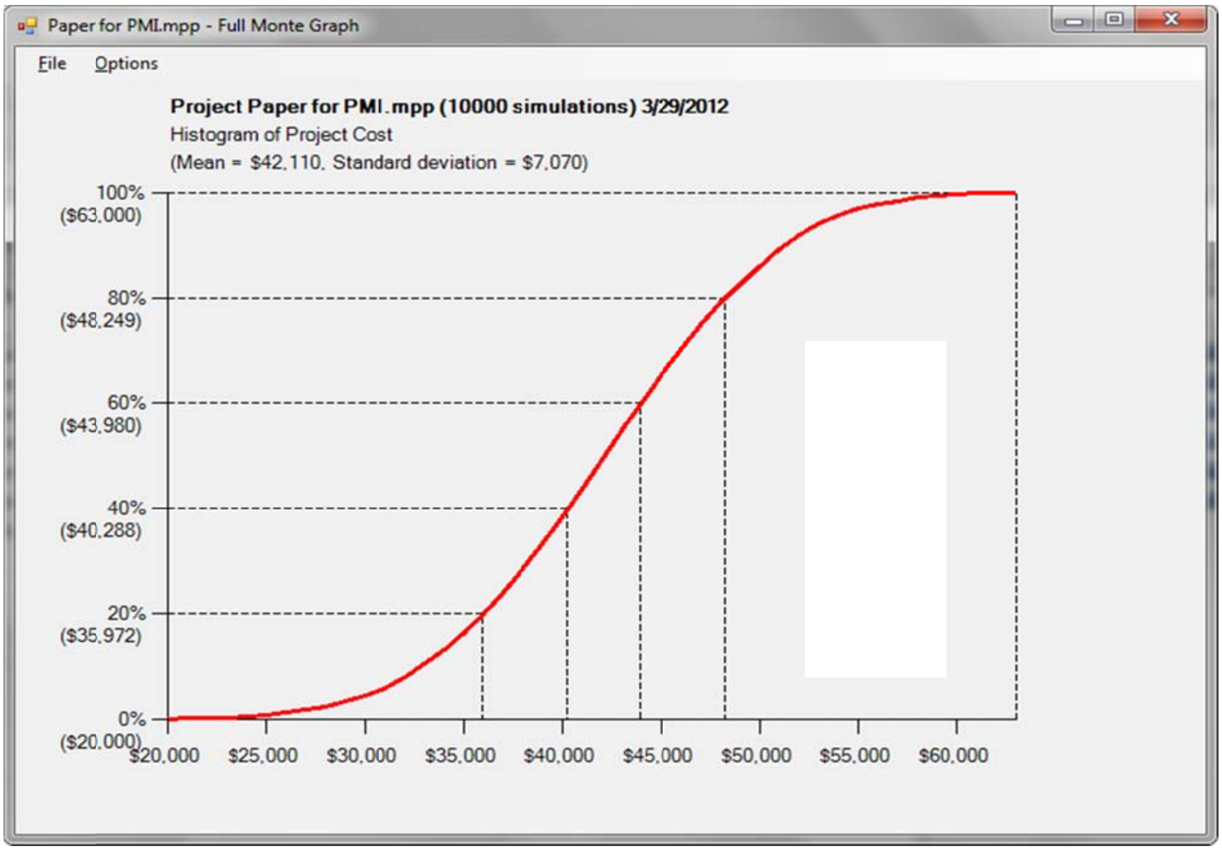


Figure 9: An S-Curve representing a cumulative distribution probability function (CDF).

To generate the variable, we merely need to generate a random (or pseudo-random) number between 0 and 1, and look it up "backwards" on this S-Curve. The probability of the corresponding value of x being selected is proportional to the slope of the curve, i.e. the probability density of the variable x. (Note that even intervals on the y-scale correspond to uneven ones on the x-scale.)

The trick therefore is to intervene early in the sampling process by generating pseudo-random numbers which are already correlated, and then to use these in the inverse sampling of the specified distribution.

Since the correlations have actually been generated between the uniformly distributed random numbers, and since these are mapped onto the cumulative distribution curve which is monotonic increasing, this is essentially the same as correlating the rank orders of the final variables. Hence it is the Spearman coefficient that matches the value specified. (As we have seen, the Pearson coefficient is the same only when the distributions are the same, but in any case the difference is relatively minor.)

Putting it all Together

OK, we are nearly done. We have seen that we can create correlated random variables as a linear combination of uncorrelated ones, and also that we can use uniformly distributed but correlated random variables as the input to the inversion process. The problem is that the correlated variables we generate by the process described above are normally distributed and we want them to be uniformly distributed.

This leads us to the final link in the puzzle. Normal variates can be turned into uniform ones by inverting the inversion process, so to speak. If we have the normal cumulative distribution function we can look up values generated by the Cholesky decomposition on the x-axis and read off uniformly distributed variables from the y-axis, and we use these to sample whatever distribution the user specified.

I have presented this in roughly the order in which it occurred to me, which is possibly not the best way to understand it, so I will recap here the process in the order in which it is done:

1. First (at the start of each trial) we sample a value for each source from a normal distribution.
2. Next we sample the task distributions by inversion using a uniformly distributed random number generated by one of two processes:

    a. If the task has no correlations we just use a standard random number.
    b. If the task has correlations, we sample a normal and combine it with the relevant source sample in the appropriate linear combination, and transform the result back to a uniform distribution in {0,1}.

Previous Published Work

The method described above and incorporated into Full Monte was developed independently, but I make no claim to its originality. In preparation for this presentation I went into the literature to see if someone else had developed it before me. I found references to a couple of papers (3 and 4) which attacked the more difficult problem of generating random variables from arbitrary distributions with specified Pearson correlation coefficients, but none that did this for the Spearman coefficient as described above. That does not however mean that it has not been done before.

BARBECANA

Conclusion

Essentially, there are two ideas presented here:

1. Using the Spearman rather than the Pearson correlation coefficient in order to properly model correlations between variables with different distributions.
2. Requiring that correlations be specified not directly between tasks but between each task and one or more independent sources.

The method described above enables samples to be produced which accurately reflect both the specified correlations and the shapes of the specified distributions. It addition, it turns out that it is not necessary to specify anything about the distributions of these sources.

But what does this mean for users? It means that:

- They can accurately represent the situation where three or more tasks each depend equally upon a single external factor (or "correlation source").
- They can also represent the situation where tasks may depend upon the same external factor but not to the same degree.
- They can do all this without having to specify a whole matrix of information, and without having to specify anything about the distribution of this external factor.
- They can specify the duration distributions independently of the correlations and these will be accurately represented, even when the duration distributions are different.

There is however one caveat. Users must be aware that the correlations entered are between a task and its sources. The resulting correlations between tasks with a shared source are the product of the two correlations specified. Thus, to create a correlation of 81% between two tasks, each should have a correlation of 90% with the shared source.

Note also that this means that in the rare cases when one wants a negative correlation between two tasks one must enter one positive and one negative correlation with the source; if **both** tasks are negatively correlated with a shared source they will be **positively** correlated with each other.

This however seems a small price to pay for the ability for the benefits listed above. In Full Monte it is also a simple matter to view the implied correlations between tasks.

References

1. Hullit, D. *Practical Schedule Risk Analysis.* Gower (2009)
2. Gene H. Golub and Charles F. Van Loan, *Matrix computations* (3rd ed.), Section 4.2, Johns Hopkins University Press. ISBN 0-8018-5414-8.
3. Iman, R. L and W.J Conover. “A Distribution-Free Approach to Inducing Rank Correlation Among Input Variables.” *Communications in Statistics* (1982).
4. Phoon, K, S.T. Quek, and H. Huang. “Simulation of Non-Gaussian Processes Using Fractile Correlation.” *Probabilistic Engineering Mechanics* (2004).

BARBECANA