STOCHASTIC PROJECT DURATION ANALYSIS USING PERT-beta DISTRIBUTIONS

Ron Davis
Department of Marketing and Decision Sciences
College of Business
San José State University
One Washington Square
San José, CA 95193-0069
Tel: 408 924 3547
Fax: 408 924 3445
E-mail: davis_r@cob.sjsu.edu

April 2006

Keywords:

Ref #: I-06-005

Not for circulation outside authors permission

STOCHASTIC PROJECT DURATION ANALYSIS USING PERT-beta DISTRIBUTIONS

By Dr. Ron Davis, San Jose State University
College of Business, One Washington Square, San Jose, California 95192
ron@mathproservices.com;

ABSTRACT

The built-in beta distributions provided by most software systems are parameterized by two shape parameters (alpha and beta) and two location parameters (A and B). In the PERT context, the desired beta distributions are specified by two statistics (mean and variance) and the same two location parameters (A and B). Hence in order to carry out project simulations with the software, one needs to first convert the PERT mean and variance and the extreme values A and B into the associated shape parameters alpha and beta. The formulas for doing this are available in certain statistical handbooks, but have not found their way into the OR/MS literature, so they are presented here for future reference in relation to carrying out PERT-beta simulations.

Because of long run times that may be required for Monte Carlo simulation of large projects, a stochastic forward pass algorithm for CPM is presented that uses beta distributions throughout. By assuming independence of preceding Early Finish times at each activity node, it enables determination of an approximate Cumulative Distribution Function (CDF) for total project duration without the use of Monte Carlo simulation. Hence it is faster than the simulation methodology. Also, because of it's analytic derivation, it can be used within the context of a nonlinear project crashing model that will reduce mean project duration at least cost. Example results using the Solver Add-In for Excel are presented.

INTRODUCTION

In a prior companion paper by the present author two CPM implementations for the spreadsheet are presented that incorporate efficiency enhancements to the earlier methodologies presented in this journal by Kala C Seal (Seal, 2001) and Cliff T Ragsdale (Ragsdale, 2003). These efficiency enhancements were motivated by the idea of separating project network structure computations from the MIN and MAX computations in the forward and backward pass of the procedure. In the context of simulation, the structure of the project network does not change during the simulation so it is preferable to compute and store this structure prior to the simulation trials. Hence during the simulation the MIN and MAX computations can be done using arrays that are no bigger than the number of predecessors and successors for each activity node in the project.

This second paper presents the formulas and algorithms for use of the beta distribution in the context of the PERT stochastic project duration analysis. The presentation is in three parts. In the first part the transformations between the [a,ß,A,B] and [µ,s^{2},A,B] parameterizations for the beta distribution are presented and utilized to determine the shape parameters associated with the PERT mean and variance formulas for activity durations. Utilization of these formulas to carry out Monte Carlo simulations of project duration is illustrated using a numerical example.

In the second part, we show how beta distributions may be utilized in an approximate way to represent the distributions of all of the early activity start and finish times in the entire project, including the last one that is the project duration itself. Although the family of beta distributions is not closed under summation, we can never-the-less define the pseudo-sum of two independent beta distributions in terms of their moments and endpoints as follows:

$$ß_1\#ß_2=[\mu_1+\mu_2,s_1^2+s_2^2,A_1+A_2,B_1+B_2]$$

We can also present a fast analytical algorithm for computing a beta approximation for the maximum of a set of independent beta distributions,

$$ß_{max} = MAX[ß_1, ß_2, \ldots, ß_n], \text{ where } F_{max}(x)=F_1(x)F_2(x)\ldots F_n(x)$$

The beta approximation for this last distribution utilizes the mean and variance of a fine-grid histogram representation of the indicated product of beta-CDFs. These approximations enable one to carry out a stochastic version of the standard forward pass computation in PERT

$$ES(j) = MAX[EF(i): \text{where } i \text{ precedes } j]$$
$$EF(j) = ES(j) + duration(j)$$

The EF times in the first of these formulas are not always independent one from the other. Our approximation, however, is to treat them as if they were independent. Recursive application of these formulas leads to a beta probability distribution for each early finish time, and therefore a beta distribution for the total project duration is obtained analytically without simulation. We call this the analytic PERT-beta approximation method (or SPBA for Stochastic PERT-beta Analytics) and illustrate it on a numerical example using several custom VBA functions implemented in EXCEL. Comparison of results obtained using the Monte Carlo and the analytic PERT-beta methods are presented.

In the third part of the paper we show how least cost project crashing can be carried out with probability of completion constraints based on the analytic beta CDF approximation developed in the second part. By systematically "tightening" the probability of completion constraint, one can develop a set of least cost crashing plans that shorten all of the relevant critical paths that impact on project duration distribution until some desired probability of completion is obtained, or until some budget constraint on crash cost is reached. This is accomplished using the nonlinear optimizer in the EXCEL SOLVER based on the analytic beta CDF for project duration implied by the crash plans developed by the Solver. The result is an optimal trade-off curve between time and money similar to that obtained by linear programming in the deterministic case. We call this methodology SPBC for Stochastic PERT-beta Crashing procedure.

PART I: **SPBS: Stochastic PERT-beta Simulation** using Monte Carlo methods.

Although the beta distribution has received lip service in relation to the PERT methodology since the very beginning, it has been conspicuously absent from the discussions of how to carry out project simulations based on the PERT paradigm. This is probably due to the fact that the PERT formulas give mean and variance statistics for all activity durations, but the beta function calls are generally expressed in terms of two shape parameters (which are referred to as alpha and beta in the Excel environment). In fact there are closed form transformations between these two sets of parameters, but heretofore it seems that these transformations have been known only to specialists working in statistical areas. Hence none of the management science texts currently or previously available (other than my own) have these formulas in them. In order to rectify this situation, the simple transformations are presented here.

For easy reference the applicable transformation formulas are summarized here as follows. For a beta distribution with the parameters [a, ß, a, b] one has

MEAN $$m = a + (b-a) * \frac{a}{a+b}$$

VARIANCE: $$s^2 = \left(\frac{a}{a+b}\right)\left(\frac{b}{a+b}\right)\left(\frac{(b-a)^2}{a+b+1}\right)$$

Solving for alpha and beta one finds it convenient to first compute a quantity representing alpha plus beta:

$$(a+b) = \left(\frac{(b-m)(m-a)}{s^2}\right) - 1$$

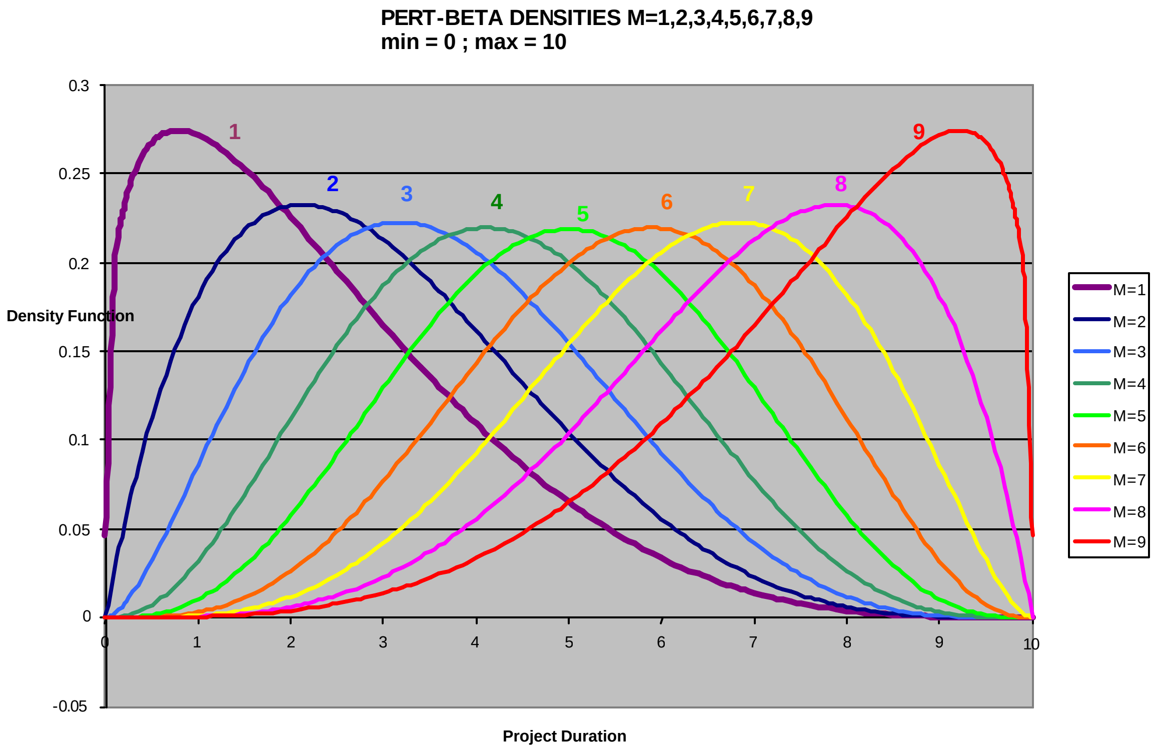
This is then split into the two parts alpha and beta according to

$$a = \left(\frac{m-a}{b-a}\right)(a+b) \text{ and } b = \left(\frac{b-m}{b-a}\right)(a+b) = a\left(\frac{b-m}{m-a}\right) = (a+b) - a$$

Letting [a, m, b] be the three PERT times assumed as given data, the usual PERT formulas give Mean = (a + 4m+b)/6 and Variance = (b-a)^2/36. When these Mean and Variance results are substituted into the formulas above for alpha and beta, one gets the unique beta distribution for each project activity. One can then generate random activity durations using the =BETAINV(RAND(), alpha, beta, min, max) percentile point function call in EXCEL and the simulation results will have statistics that converge to these theoretical statistics as the sample size becomes larger and larger. We call these the

PERT-beta distributions because they are beta distributions that arise as a result of the PERT mean and variance formulas.

The chart in Figure 1 shows the various density function shapes that occur as m varies from 1 to 9 when a is 0 and b is 10. One finds that alpha+beta is maximized with value 8 when the distribution is symmetric (m = 5) and the sum decreases towards four as mid moves further away from the middle of the interval. Note that a lateral shift, or a rescaling of the width of the interval, or a combination of both, has no effect on alpha, beta or alpha+beta. Any such affine transformation of a beta distribution leaves its shape, and its shape parameters, unchanged.



Recently, in 2001, Kala C. Seal published a “Generalized PERT/CPM Implementation in a Spreadsheet” in the INFORMS Transactions on Education. This was followed by the 2003 paper by Ragsdale and the 2004 paper by the present author that present various enhancements for the CPM method in the spreadsheet. About the same time, several management science texts appeared showing how to use the RAND() uniform variate function in combination with inverse CDF functions and the Data/Table command to generate simulation results for project duration that could be analyzed with statistical summary report generators such as the Analysis ToolPak add-in in EXCEL. Third party add-ins such as @Risk (from Palisade Corporation) and Crystal Ball (from Decisioneering) are also available to facilitate the simulation process in the spreadsheet. However, all essential results can be developed in EXCEL without such add-ins if one simply uses the built-in functions and commands as appropriate.

Hence in principle the Monte Carlo procedure for analyzing the distribution of project duration given beta distributed activity times can be specified in terms of a combination of any of the CPM implementations augmented with the =BETAINV(RAND(), alpha, beta, a, b) function calls for generating beta distributed random activity durations with the desired mean and variance properties and the Data/Table command for generating and tabulating multiple simulation trials that can be analyzed statistically after the fact. However, heretofore the appropriate beta shape parameter formulas have not been presented that make this possible. We have done so here in hopes that future textbook treatments will show these formulas and encourage students to use them.

The precedence structure matrix used in this context is an extension of the one shown by Seal. It is a symmetric matrix of zeros and off-diagonal ones where $M_{ij} = 1$ if and only if activity "i" is an immediate predecessor of activity "j" and therefore also activity "j" is an immediate successor of activity "i". In our spreadsheets, we let M_{ii} be the letter symbol for activity "i" since these diagonal elements are never used as mask indicators. The column vector above each M_{ii} is the indicator mask for the immediate predecessors of activity "i" (that we put into a predecessor pointer list denoted by PRED_i) and the column vector below each M_{ii} is the indicator mask for the immediate successors of activity "i"(that we put into a successor pointer list denoted by SUCC_i). One can then write

$$ES(j) = MAX(EF(PRED\_j))$$

and EF(j) = ES(j) + duration(j) for the forward pass computations where by EF(PRED_j) we mean the array of early finish times that correspond to the mask PRED_j. Likewise, for the backward pass one has

$$LF(i) = MIN(LS(SUCC\_i))$$

and LS(i) = LF(i) – duration(i). (Note: The notation used here is meant to be conceptually suggestive only. For the detailed Excel spreadsheet syntax for accomplishing these results, see the appendix for the companion paper Davis (2005-1)).

To see how this spreadsheet implementation works in practice we have simulated a modification of the example presented in the Seal paper using 16,000 trials in order to get a reasonably accurate result. The modifications were all increases to the time estimate data for a few project activities in order to make several different paths have very similar mean project durations so that there is meaningful "competition" between them for longest path. The modified project data and associated precedence structure matrix is shown in the following table.

Table 1: Example Project Data for Simulation Analysis

Activities	Description	Pred	ai	mi	bi	A	B	C	D	E	F
A	Plan Topic		8	10	12	A	0	0	1	1	0
B	Obtain Speakers		2	4	6	0	B	1	0	0	0
C	List Meeting Locations	B	3	6	9	0	1	C	1	0	0
D	Select Location	A, C	1	5	9	1	0	1	D	0	1
E	Finalize Speakers Schedule	A	4	5	6	1	0	0	0	E	1
F	Prepare/mail Seminar Brochures	D, E	2	8	20	0	0	0	1	1	F

Based on the usual PERT mean and variance formulas, and the neglected beta shape parameter formulas give above, one derives the following secondary parameter set for this example.

Table 2: Secondary Activity Parameters

Activities	alpha	beta	min	max	Mean	Variance	alpha+beta
A	4	4	8	12	10	0.444444	8
B	4	4	2	6	4	0.444444	8
C	4	4	3	9	6	1	8
D	4	4	1	9	5	1.777778	8
E	4	4	4	6	5	0.111111	8
F	2.938272	4.617284	2	20	9	9	7.55555556

Notice that activities A through E are all symmetric whereas activity F is skewed right, with mean towards the left end of its interval. The first four columns of the table (after the activity names) are the beta distribution parameters for the activity durations given in the sequence that they are expected by the EXCEL built-in functions, BETADIST and BETAINV. Using the RAND() function for a uniformly distributed cumulative probability value (between zero and one) as the first argument in the BETAINV function call, one can then easily generate beta distributed activity durations for each of the activities and parameter sets listed above. Application of the standard forward and backward pass recursion formulas given before readily give rise to a set of additional columns for which simulation results can be tabulated using the Data/Table command. These are shown below for sample random activity duration times. The critical path in this case is seen to be BCDF.

Table 3: PERT Computations based on randomized activity durations

Activity	ActivityDur	ES	EF	LF	LS	Slack	Critical?
A	8.765789	0	8.765789032	9.435329	0.66954	0.66954	0
B	3.832306	0	3.832305908	3.8323059	0	0	1
C	5.603023	3.832306	9.43532896	9.435329	3.832306	0	1
D	7.061699	9.435329	16.49702787	16.497028	9.435329	0	1
E	5.30654	8.765789	14.07232952	16.497028	11.19049	2.424698	0
F	5.485666	16.49703	21.98269367	21.982694	16.49703	0	1

For this study, a Monte Carlo simulation in Excel was run having 16,000 simulation trials so that all mean, variance and skew estimates would be fairly accurate. The Data Analysis Toolpak (available as an add-in for Excel) was then invoked to get the

Descriptive Statistics Report and the Histogram Chart shown below. These give a point of reference for the analytical approximations described later in this paper. We shall say more about them then.

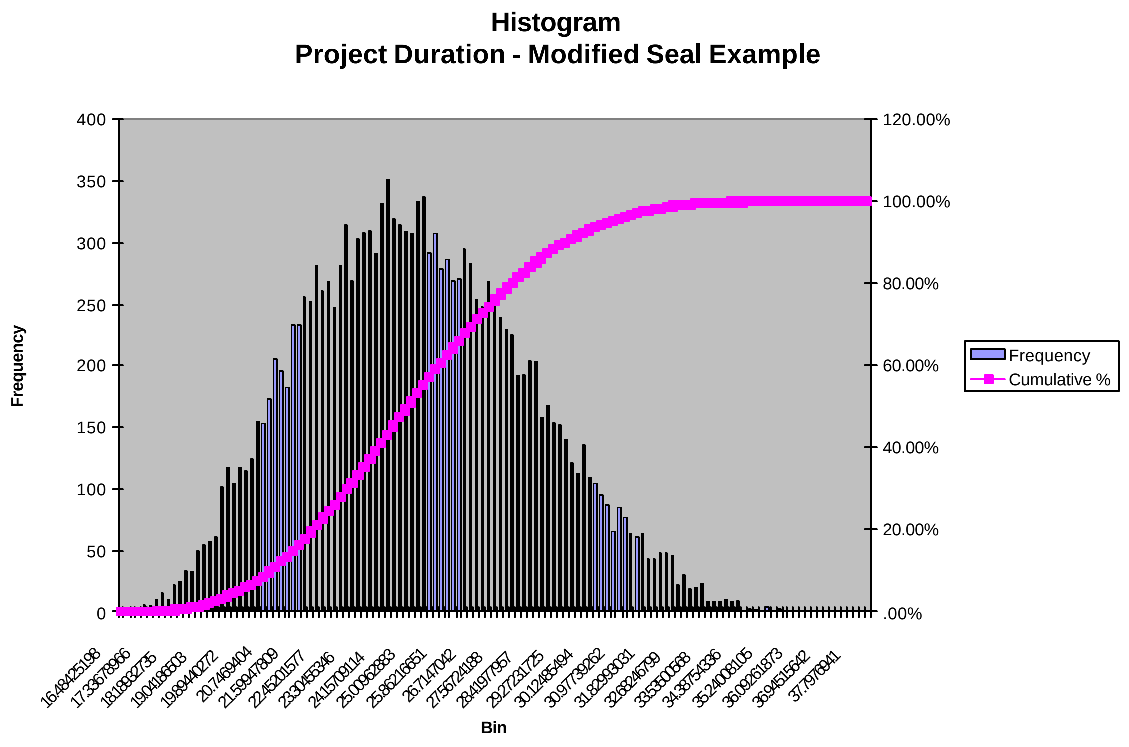
Table 4: Descriptive Statistics for PERT-beta Simulation

Project Duration Stats		*Cont'd*	
Mean	24.90411525		
Standard Error	0.02551292	Range	21.48395
Median	24.7417171	Minimum	16.48425
Mode	22.84211111	Maximum	37.9682
Standard Deviation	3.227157506	Sum	398465.8
Sample Variance	10.41454557	Count	16000
Kurtosis	-0.376652346	Largest(4000)	27.12112
Skewness	0.238764409	Smallest(4000)	22.50351

Table 5: Path Probs

Path	Probs
A-D-F	0.248438
A-E-F	0.375625
B-C-D-F	0.375938

Note that estimated mean duration is almost one day longer than the PERT result, and the estimated variance is somewhat less than the average of the three PERT variance results. The 4000th smallest and largest values give estimates of the first and third quartile that show the inter-quartile range about the median value. Note that third quartile minus median is greater than median minus first quartile, as one would expect for a distribution with right skew. Also, the mean is pulled to the right of the median by the longer tail on the right.



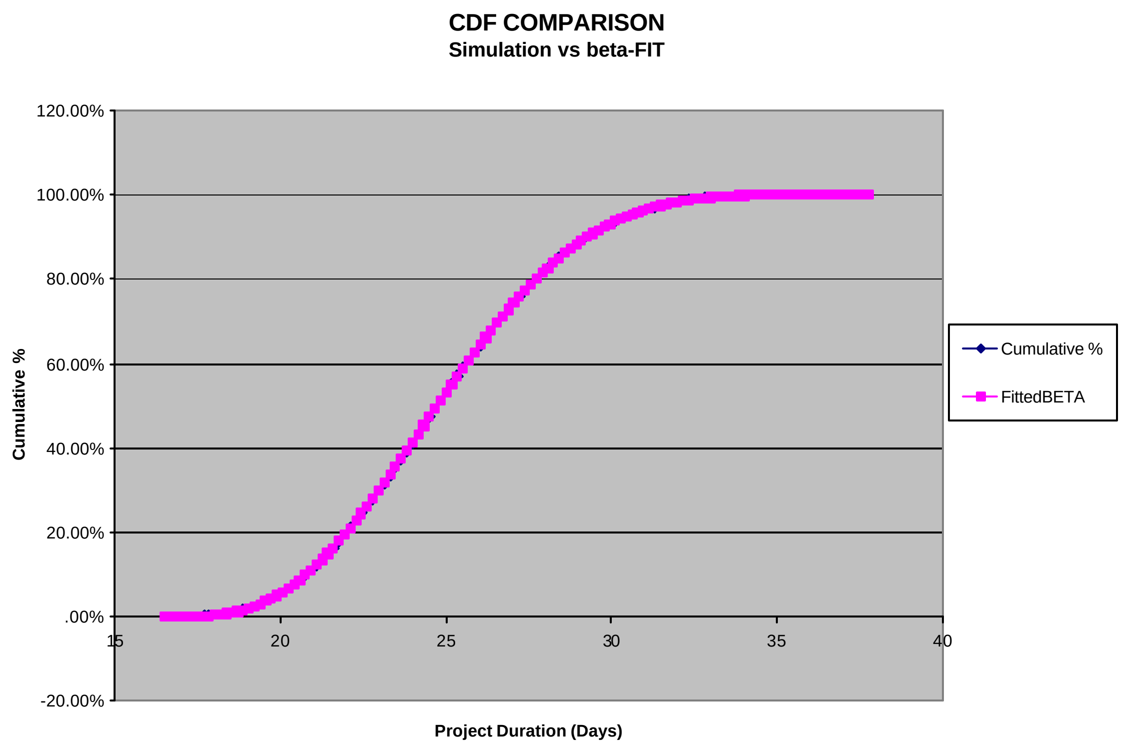
Observe that the distribution for total project duration, based on the interaction of all three project paths, assumes a shape that looks very much like a beta distribution with a substantial right skew. The right skew derives partly from the fact that the final activity F is itself right skewed, and partly from the fact that the early start time for F, being the maximum of two early finish times, is also skewed to the right. The path probability table shows that each of the paths is critical a significant portion of the time. Note that since the duration of A-E and B-C-D both have the same mean and are both symmetric, the criticality probabilities for A-E-F and B-C-D-F are virtually identical. Note that right skew implies that percentiles for percents close to 100 will lie further to the right than for the symmetric PERT approximation.

Using the method of moments, we can use the mean and variance results and the min and max times observed during the simulation as a basis for computing a beta-FIT for the simulation result as described earlier. Quite simply, it is the beta on the given range that matches the first two moment statistics just given in the Descriptive Statistics Report shown. The parameter estimates for the Beta-FIT obtained in this way are as follows.

Table 5.

Beta Fit	Parameters
alpha_fit	3.7474599
beta_fit	5.8144816
min_fit	16.484252
max_fit	37.968202
alpha+beta_fit	9.5619415

The CDF of this beta-FIT and the CDF of the simulation result are then plotted on the same axes to get a measure of how well the fitted beta matches the cumulative % results from the simulation across the entire range of project durations observed. As is seen in the following chart, the fit is exceedingly close, with the beta-fit points lying right on top of the cumulative % points across the entire range of the distribution. This leads to the conjecture that perhaps ALL of the early finish time distributions are closely approximated by beta distributions. And sure enough, for the present example, that turns out to be the case, although we do not take the space to show all the CDF plots that indicate this is so.



The significance of this result is two fold. First it means that probability and percentile questions posed about project duration can be readily answered using the beta-fit for the PERT-beta simulation results, rather than using the histogram of the simulation results directly. In particular, the probability of not completing the project by any given target time “t” is

1 – BETADIST(t, alpha_fit, beta_fit, min_fit, max_fit),

and the time required to achieve a probability “p” of timely completion is

BETAINV(p, alpha_fit, beta_fit, min_fit, max_fit).

Secondly, and more importantly, it suggests the idea of developing beta distributions for ALL of the Early Finish times during the forward pass in an analytical way, so that the Monte Carlo simulation process can be omitted altogether. Since the distribution for project duration is given by the distribution of the early finish time for the final project node in the project network, the beta approximation for project duration would be at hand at the end of the forward pass and no Monte Carlo simulation would be needed. The details of this “stochastic” analytical forward pass procedure are described next.

PART II **SPBA: Stochastic PERT-beta Analytics** (without simulation)

The general strategy of this procedure has already been noted. The forward pass is carried out by deriving a beta distribution for each ES and EF time based on the very same recursive formulas that are used in the deterministic case. The ES distributions are based on the product of the EF distributions for nodes immediately preceding it and the

EF distributions are obtained using the pseudo-sum formulas presented at the beginning of the article. There is an implicit assumption of independence between the EF times in the product that is not true in general, but that is the nature of the approximation we propose here. Rigorous treatment of the covariance between the EF times appears to be very complex, and may not materially affect the results in most practical situations. We relegate concern about this issue to some future research effort.

The main issue remaining in this context is how to use the histogram distribution to develop the beta-fit for a distribution defined as a MAX of several (independent) beta distributions. So if $F_1, F_2, \ldots, F_n$ are CDF functions for N beta distributions (all EF times), how do we assign a beta-fit for the distribution having a CDF given as

$$F(x) = F_1(x)F_2(x)\ldots F_n(x)$$

First note that the min x value for F(x) will be the maximum of the left endpoints of the beta distributions appearing in the product. That is

$$\text{MinF} = \text{MAX}(\text{MinF}_k\text{: } k=1,\ldots,n)$$

Likewise

$$\text{MaxF} = \text{MAX}(\text{MaxF}_k\text{: } k=1,\ldots,n)$$

To compute the shape parameters for the product distribution, we estimate the mean and variance of F(x) by means of the histogram distribution associated with a suitably fine grid of x values from MinF to MaxF. These lead to the associated shape parameters via the transformation formulas developed previously.

If we evaluate F(x) on a grid of equally spaced points, and connect these points with straight lines, we obtain the CDF of a histogram that has constant height bars on each interval. If we denote the endpoints of the intervals $x_0, x_1, \ldots, x_n$ and the associated interval probabilities as $p_1, p_2, \ldots, p_n$, (where $p_i = F(x_i)-F(x_{i-1})$) then the mean and variance of the histogram distribution are given by

$$m_H = \sum_{i=1}^{n} p_i \left(\frac{x_{i-1} + x_i}{2} \right)$$

and

$$s_H^2 = \sum_{i=1}^{n} p_i \left(\frac{x_{i-1}^2 + x_{i-1}x_i + x_i^2}{3} \right) - m_H^2$$

Hence the mean is a weighted average of the interval midpoints, and the variance is the second moment less the square of the mean (as usual) where the second moment is a weighted average of the interval second moments.

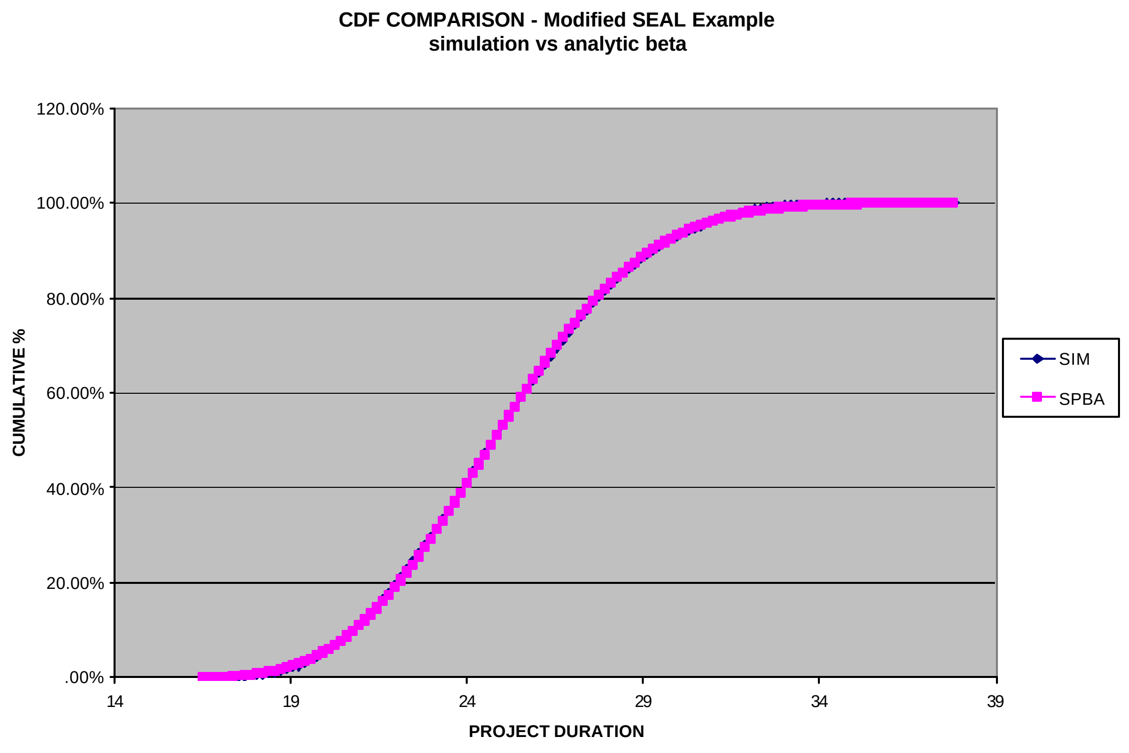
These mean and variance values lead to shape parameters for F(x) as described previously. The question of how fine the grid of points must be in order to get a sufficiently accurate beta-fit may be resolved in the following fashion. The first approximation is made using 100 intervals and a second approximation is made using 200 intervals. The absolute percentage change in the estimate of the mean and the standard deviation is noted. If both of these are less than some pre-selected epsilon, then the approximation is accepted. If not, the interval size is bisected again and the process repeated. In a finite number of refinements, the change in the mean and standard deviation will be small enough, and the approximation is accepted. A VBA custom function betaPROD() carries out this process so that the beta-fit for each EF distribution can be obtained as a function call rather than by cluttering up the spreadsheet itself. Dynamic array dimensioning is used to accommodate the increasing number of intervals that are required to achieve the desired degree of accuracy.

With the distribution for an ES time thus determined, the associated EF time is obtained via the pseudo-sum formula based on adding the means and variances as well as the mins and maxs of the ES time and the activity duration time. This makes sense since the means and variances of independent distributions do in fact add. A VBA custom function betaSUM() carries out this process so that the parameters of the beta pseudo-sum can be obtained as the output of an array function.

This analytic procedure has been carried out using the two beta distribution approximation functions on the data presented before for the example project treated in this paper. The output arrays from the VBA functions include the mean, variance and alpha+beta statistics as well as the four beta parameters. The results are as follows:

Distribution	alpha	beta	min	max	mean	variance	alpha+beta
EF_A	4	4	8	12	10	0.444444444	8
EF_B	4	4	2	6	4	0.444444444	8
EF_C	8.153846154	8.153846154	5	15	10	1.444444444	16.30769231
MAX(EF_A,EF_C)	6.126854343	10.66861087	8	15	10.55354525	0.638040269	16.79546522
EF_D	9.573961512	12.33928044	9	24	15.55354525	2.415818046	21.91324195
EF_E	7.6	7.6	12	18	15	0.555555556	15.2
MAX(EF_D,EF_E)	7.797075116	15.58762746	12	24	16.00111573	1.312481118	23.38470257
EF_F	7.065491954	12.20207724	14	44	25.00111573	10.31248112	19.26756919

Calls to betaSUM() are used in the rows for EF_C, EF_D, EF_E and EF_F. Calls to betaPROD() are used in the rows for ES_D and ES_F indicated with MAX functions. This analytic method does not use simulation at any point in the process. Hence the predicted mean duration is 25 days in comparison with 24 based on PERT and 24.9 based on PERT simulation. Variance is projected to be 10.31 instead of 10.41 by the PERT simulation. For the visual test we plot the simulation cumulative % and the analytic beta CDF for EF_F on the same axes, as follows.



In this case, which has a substantial right skew as previously seen, there is no apparent difference between the simulation results and the analytic PERT-beta result. The hope, of course, is that this will remain to be true for most practical project analyses, no matter how large. That will have to await the accumulation of more empirical experience with the method. Our purpose here is to document the approximation method in a public domain context with only an indication that the method might be sufficiently accurate for many practical applications.

Normally, the CDF for the PERT single critical path approximation would have been shown as well, but because of the ambiguity regarding variance in this case, it has been omitted in this comparison chart. Typically, it lies to the left and above the true distribution. Our main point here is that the PERT-beta simulation result and the analytic PERT-beta results are virtually identical in terms of the CDF associated with the two distributions.

PART III **SPBC: Stochastic PERT-beta Crashing** based on the SPBA approximation

One of the analytical extensions of the Critical Path Method is the problem of finding least cost plans for reducing project duration. This is typically formulated as a linear programming problem in which the cost of activity "crashing" is computed in the objective function and project duration is subject to an upper bound constraint. As the upper bound on project duration is reduced, the cost of crashing increases in a continuous monotonic way, exhibiting a piecewise linear shape with increasing marginal cost as project time decreases. The plot of time versus cost in this situation is often called the project crashing time-cost trade-off curve. It is piecewise linear if the cost functions for each activity are linear.

To reformulate the project crashing problem in the stochastic case, which becomes a nonlinear programming problem, we define a compression factor k_i for each activity, where $0 \le k_i \le 1$. This represents the fractional amount by which the maximum time b_i has been "compressed" towards the minimum "crash" time a_i, and similarly for the modal time m_i. Hence the original time frame $[a_i, m_i, b_i]$ becomes the compressed time frame $[a_i, m_i - (m_i – a_i)* k_i, b_i - (b_i – a_i)* k_i]$. Since the shape parameters of a beta distribution are not affected by a compression of this sort, this leaves the shape parameters unchanged. Also, the crash time a_i is unchanged as well. Hence the original parameter set $[a_i, ß_i, a_i, b_i]$ is replaced with $[a_i, ß_i, a_i, b_i – (b_i-a_i)* k_i]$ after compression by a factor k_i.

Then if c_i is the cost of fully crashing activity i down to the crash time a_i (i.e. $k_i = 1$) the linear model for total crash cost becomes

$$TCC = \sum_{i=1}^{n} c_i k_i$$

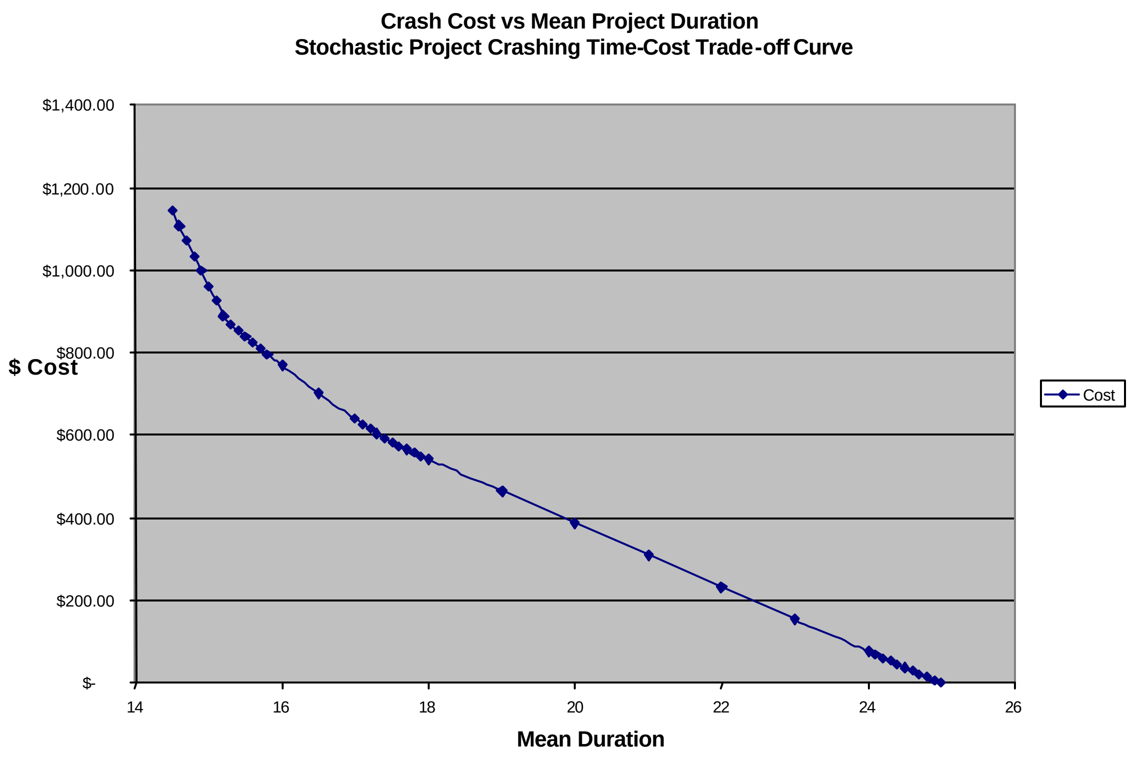
In other words, we assume that the crash cost for each activity is proportional to the compression factor for the activity, as is assumed also for the deterministic case.

The time constraint in the formulation will be placed on the mean of project duration, which follows directly from the beta parameter set for the early finish of the project. This is obtained analytically from the same sequence of betaSUM() and betaPROD() function calls used before, but now evaluated using the compressed max times in the last argument. If we denote the beta parameters for early project finish as $[a_p, ß_p, A_p, B_p]$ then the constraint on mean project duration is expressed as follows:

$$A_p + \frac{a_p}{a_p + b_p}(B_p - A_p) \quad \le \quad Time$$

Hence the stochastic project crashing problem is to minimize TCC subject to the unitary constraints on the compression factors and the time constraint on the mean duration of the early finish time distribution.

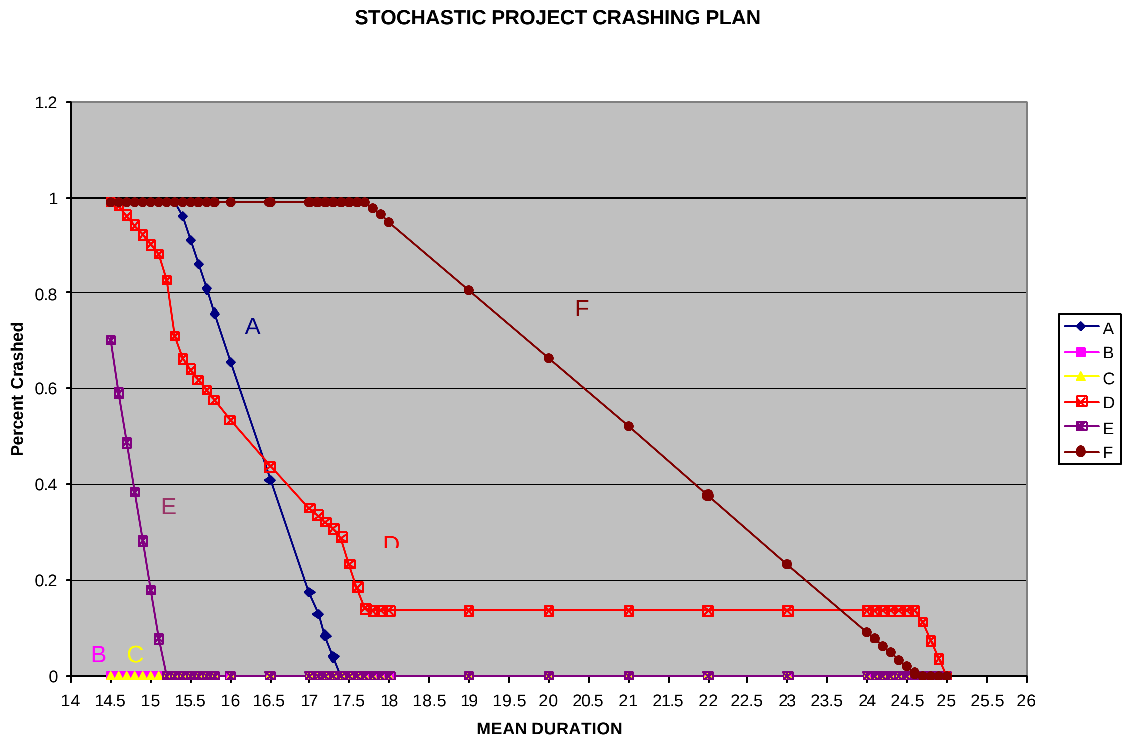
In the example at hand, we postulated crash cost coefficients as follows [200, 350, 275, 185, 321, 544] for activities A through F, respectively. The mean duration without crashing is about 25 days. By tightening the mean duration constraint gradually, resolving the least cost crashing problem at each point on a grid of decreasing mean times, one maps out the time-cost trade-off curve parametrically and obtains the curve shown below. We used the Excel Solver with the GRG optimization option to develop these results.



While the curve is no longer exactly piecewise linear because of the complex nonlinear relationships between compression fractions and mean project duration, it nevertheless looks very much like a piecewise linear curve, and the breakpoints in the curve where the slope changes are clearly caused, as in the deterministic case, by activities becoming fully crashed so that some other activities must be used to crash further.

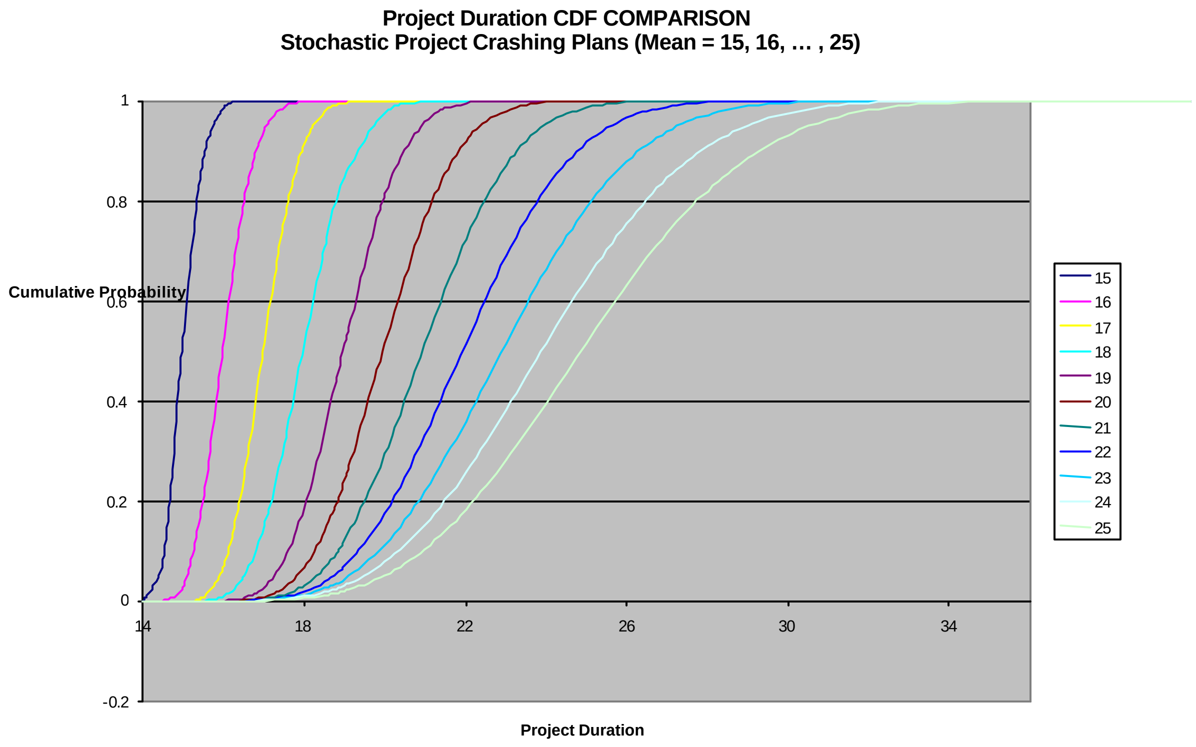
The crash plan can be seen in greater detail by plotting the compression factors as a function of the mean duration constraint limit. This is shown below. The relationship to the cost function is rather easy to see. There is a smaller marginal cost while F is being crashed, which brings mean project time down to about 17.5 days. The marginal cost increases slightly when F is fully crashed and A takes over as the activity being crashed which brings mean duration down to about 15.5 days, and then a more marked marginal cost increase occurs when A is fully crashed and E comes in to crash down to the minimum feasible mean duration of 14.5 days.

Through it all, however, D is also being used as a compressed activity. Its sequence is more complex, because it is sometimes the only activity being compressed and on other intervals it is being compressed in parallel with some other activity. By inspecting the result table from which the curve was plotted, one finds there are six intervals in the compression plan for activity D. It is the only activity being crashed from T = 25 down to about T=24.6. Then D holds the same compression factor while F is compressed and fully crashed at about T=17.7 days. Then D is compressed some more down to about T=17.3 days, at which time A comes into the crash plan. A and D are crashed in parallel until A is fully crashed at about T=15.3. Then D is crashed by itself until about T=15.1 when E comes into the crash plan. Finally, D and E are compressed until D is fully compressed at T=14.5 days, at which point E is about 70% crashed. B and C do not enter the crash plan at all.



(Note: In order to avoid beta function calls with intervals of zero width, we obtained the reported results with $k_i \leq 0.99$ rather than 1.0. Also, to avoid some out of range function evaluations done by the Solver, we found it necessary to impose $k_i = 0.99$ rather than the inequality form after it became clear that any given activity was to be fully crashed).

It is interesting to see the results of the crashing plan in terms of CDF for project duration as well. A comparison of the 11 project duration distributions with means from 15 to 25 is shown in the following chart. The min time is 14 throughout, but the max time, the mean and the variance all decrease as the constraint on mean duration is tightened. This chart can also be used to choose a crash plan when a constraint is applied to some percentile, such as the 95th percentile. For example if the planner wanted the 95th percentile to occur at 20 days, from the graph one could see that the mean would have to be somewhat less than 19 days, and the cost for the crash plan would somewhat greater than $460. By computing the location of the 95th percentile in the spreadsheet as a side calculation, one finds by rerunning the Solver a few more times that actually the mean duration must be brought down to 18.32 days to get the 95th percentile at 20 days, and at this point the cost of crashing is $516.50.



CONCLUSIONS

The principal points made in this paper may be summarized as follows.

1. Simulation of project duration may be done in the spreadsheet by making use of the closed form transformation of beta parameters that maps [mean, variance, min, max] into [a, ß, min, max]. In this connection, the RAND() function may be used to obtain uniformly distributed probability values and the BETAINV() function may be used to obtain beta distributed activity duration times. The forward and backward pass CPM computations may be done using the MAX() and MIN() functions generated by the VBA macros given in the companion paper (Davis 2005-1) that take the precedence relations into account.
2. Although the beta distribution family is not closed under addition, one can nevertheless use the method of moments to pick a beta to closely represent the sum of two beta distributions. This is easily done by adding the four statistics in the [mean, variance, min, max] parameterizations together. Moreover, the product of several beta distributions can be modeled by a beta having the appropriate mean, variance, min and max statistics. In this connection, the histogram distribution is used as an approximation for the product of the betas on a grid that can be refined until any desired degree of accuracy is obtained. For simplicity, a grid with 1000 intervals is shown in the appendix.
3. Using the betaSUM() and betaPROD() VBA functions developed for these approximations, one can perform the forward pass of the CPM in a stochastic mode, getting ES distributions using the betaPROD() function when two or more

predecessors are present, and getting EF distributions using the betaSUM() function that adds an activity duration to its ES time. The EF distribution at the final node is the project duration distribution, and this result is obtained analytically, without any Monte Carlo simulation. This provides a fast alternative to the Monte Carlo simulation approach on large networks for which simulation is computationally burdensome.

4. The availability of an analytic project duration approximation makes it possible to formulate a project crashing problem in the stochastic mode. The crash cost in the model is still linear, but the time constraint on mean project duration is nonlinear. Hence the GRG nonlinear programming option in the Excel Solver is used to carry out crashing plan optimizations. The details of the crash plans obtained are distinctly nontrivial, even for a small example such as considered here. However, the resulting project duration distributions exhibit the decreasing mean and variance property that one would expect for them, while minimizing the cost at each step of the way.

BIBLIOGRAPHY

- Albright, S. Christian (2001), ***VBA for Modelers**: Developing Decision Support Systems with Microsoft Excel*, Duxbury, Pacific Grove, CA
- Davis, Ronald E. (2005), "A fast spreadsheet implementation of the Critical Path Method", INFORMS Transactions on Education 5:2 (TBD)
- Ragsdale, Cliff T. (2003), "A New Approach to Implementing Project Networks in Spreadsheets", INFORMS Transactions on Education 3:3 (76-85).
- Roman, Steven (2002), *Writing Excel Macros with VBA (2nd ed.),* O'Reilly & Assoc., Inc., Sebastopol, CA
- Seal, Kala C. (2001), "A Generalized PERT/CPM Implementation in a Spreadsheet", INFORMS Transactions on Education 2:1 (16-26).

APPENDIX. VBA FUNCTIONS

Since both of these functions return seven values, they must be invoked as array functions. For example, seven cells in a row are selected as the location for the outputs, the formula with desired four-cell argument ranges are placed in the argument list, and then Cntl-Shift_Enter is pressed to get the array function results.

```
Function betaSUM(beta1 As Range, beta2 As Range)
Dim beta1_min, beta1_max, beta1_alpha, beta1_beta As Double
Dim beta2_min, beta2_max, beta2_alpha, beta2_beta As Double
Dim betas_min, betas_max, betas_alpha, betas_beta As Double
Dim f1, f2, mean1, mean2, means, var1, var2, vars As Double
Dim alphaplusbeta As Double

beta1_alpha = beta1.Cells(1): beta1_beta = beta1.Cells(2)
beta1_min = beta1.Cells(3): beta1_max = beta1.Cells(4)
```

```
beta2_alpha = beta2.Cells(1): beta2_beta = beta2.Cells(2)
beta2_min = beta2.Cells(3): beta2_max = beta2.Cells(4)

f1 = beta1_alpha / (beta1_alpha + beta1_beta)
f2 = beta2_alpha / (beta2_alpha + beta2_beta)

mean1 = beta1_min + f1 * (beta1_max - beta1_min)
mean2 = beta2_min + f2 * (beta2_max - beta2_min)
means = mean1 + mean2

var1 = f1 * (1 - f1) * (beta1_max - beta1_min) ^ 2 / (beta1_alpha + beta1_beta + 1)
var2 = f2 * (1 - f2) * (beta2_max - beta2_min) ^ 2 / (beta2_alpha + beta2_beta + 1)
vars = var1 + var2

betas_min = beta1_min + beta2_min
betas_max = beta1_max + beta2_max

alphaplusbeta = (means - betas_min) * (betas_max - means) / vars - 1
betas_alpha = alphaplusbeta * (means - betas_min) / (betas_max - betas_min)
betas_beta = alphaplusbeta - betas_alpha

betaSUM = Array(betas_alpha, betas_beta, betas_min, betas_max, means, vars,
alphaplusbeta)

End Function
----------------------------------------------------------------------------------------------------
Function betaPROD(beta1 As Range, beta2 As Range)
Dim beta1_min, beta1_max, beta1_alpha, beta1_beta As Double
Dim beta2_min, beta2_max, beta2_alpha, beta2_beta As Double
Dim betap_min, betap_max, betap_alpha, betap_beta As Double

beta1_alpha = beta1.Cells(1): beta1_beta = beta1.Cells(2)
beta1_min = beta1.Cells(3): beta1_max = beta1.Cells(4)

beta2_alpha = beta2.Cells(1): beta2_beta = beta2.Cells(2)
beta2_min = beta2.Cells(3): beta2_max = beta2.Cells(4)

Dim bval, bval1, bval2, Incr As Double
Dim Index As Integer

betap_min = Application.Max(beta1_min, beta2_min)
betap_max = Application.Max(beta1_max, beta2_max)
Incr = (betap_max - betap_min) / 1000
 Dim bins(0 To 1000) As Double
 Dim probs(0 To 1000) As Double
 bins(0) = betap_min: probs(0) = 0: bval = bins(0)
 For Index = 1 To 1000
    bins(Index) = bval + Incr
    bval = bins(Index)
    bval1 = Application.Min(bval, beta1_max)
    bval2 = Application.Min(bval, beta2_max)
    probs(Index) = Application.BetaDist(bval1, beta1_alpha, beta1_beta, beta1_min,
beta1_max) * Application.BetaDist(bval2, beta2_alpha, beta2_beta, beta2_min,
beta2_max)
 Next Index
 bins(1000) = betap_max
 probs(1000) = 1

 Dim Firstime As Boolean
 Dim bin_0, bin_i, bin_i1, cum_i, p_i, mid_i, scd_i, emv, ev2 As Double

 Firstime = True
For Index = 0 To 1000
    If Firstime Then
        cum_i = 0
        bin_i = bins(0)
        bin_0 = bin_i: emv = 0: ev2 = 0
        Firstime = False
    Else
        p_i = probs(Index) - cum_i
        cum_i = probs(Index)
        bin_i1 = bins(Index)
        mid_i = (bin_i + bin_i1) / 2
        scd_i = (bin_i ^ 2 + bin_i * bin_i1 + bin_i1 ^ 2) / 3
        bin_i = bin_i1
        emv = emv + p_i * mid_i
        ev2 = ev2 + p_i * scd_i
    End If
Next Index

Dim Var, alphaplusbeta As Double

        Var = ev2 - emv ^ 2
        alphaplusbeta = (emv - bin_0) * (bin_i - emv) / Var - 1
        betap_alpha = alphaplusbeta * (emv - bin_0) / (bin_i - bin_0)
        betap_beta = alphaplusbeta - betap_alpha
        betaPROD = Array(betap_alpha, betap_beta, bin_0, bin_i, emv, Var, alphaplusbeta)

End Function
```