PROBABILITY ESTIMATION AND ITS APPLICATIONS

ASHRAF ABD EL MOEZ GOUDA

B.Sc., M.Sc.

Supervisor

Prof. Dr.(DSc.) Tamás Szántai

Budapest University of Technology and Economics

Budapest
2005

Abstract

The computation of the cumulative distribution function values of Dirichlet distribution by using several algorithms are to be described and estimation of the probability in network models via cross entropy method and completion time estimation in PERT network will be investigated. The results presented in this thesis were published in ([46], [47], [48], [49], [50], [51] and [52]).

The Dirichlet distribution is one of the important multivariate distributions that appear in many applications, in order statistics, probabilistic constrained programming models and delivery problems. Another application is the modeling of consumer purchasing behavior for non durable items such as foods and toiletries. A model for multibrand purchasing behavior is given for the use of multivariate beta distribution for the independent case. Suppose this is a product field with k brands and let the random variables Z_i represent the average rate of purchase of brand i let $W = \sum_{i=1}^{k} Z_i$ represent a consumer's rate of buying of the product field as a whole. Then the joint distribution of $(X_1, X_2, ..., X_{k-1})$ follows a Dirichlet distribution where $X_i = \frac{Z_i}{W}$ represents the proportion of a consumer's total purchases devoted to brand i. Another application would be modeling the activity times in a PERT (Program Evaluation and Review Technique) network. A PERT network has a collection of activities and each activity is usually modeled as a random variable following a beta distribution. A Dirichlet distribution for the entire network follows directly since each marginal distribution of Dirichlet is a beta. Using the properties of the Dirichlet distribution we can see that any subnetwork will follow a Dirichlet distribution.

The evaluation of multivariate probability distribution is an important problem in applications. Methods for evaluation of multivariate probability distribution can be classified into broad categories, numerical integration, bounding techniques,and numerical approximations. While several methods have been proposed for the computation of multivariate normal probabilities, only a few paper is dealing with the computation of Dirichlet probabilities.

In the first part of the dissertation there will be developed a recursive algorithm for

the calculation of Dirichlet probability distribution function values up to 7 dimensions. This procedure is based on a generalization of Szántai's result published in his dissertation for candidate degree of HAS and the Lauricella series expansion. This gives the possibility of application all algorithms for bounding and estimating multivariate normal probability distribution function values developed before by J. Bukszár, A. Prékopa and T. Szántai. In stochastic programming applications one need the gradient vector and Hessian matrix of the multivariate probability distribution function, too. A new algorithm for the Hessian matrix calculation will be given. All these estimations are most effective when the estimated probability value is close to one. However many times one need to estimate small probability values, too. These are called rare event probabilities in the literature, the Sequential Conditioned Sampling (SCS) and Sequential Conditioned Importance Sampling (SCIS) algorithms will be developed to estimate the rare event probabilities of Dirichlet distribution. Using Szántai's result published in his dissertation for candidate degree of HAS and interesting property of the Dirichlet distribution new versions of the SCS and SCIS algorithms will be developed, called SCSA, SCSB, SCISA and SCISB, respectively. The efficiency of the new methods will be compared to Crude Monte Carlo (CMC) simulation method. Numerical test results will also be presented.

In the second part of the dissertation estimation of rare event probabilities in stochastic networks will be described. Stochastic simulation has proven itself in practice; it is commonly used for accurate estimations in many problems. However, there are some limitations to the standard stochastic simulation method in many cases. An important class of problems that cannot be efficiently solved using standard simulation is that involving rare events. Since these rare events occur so seldom in a standard simulation, one has to apply very large sample sizes what need too much CPU time. This is why different methods and techniques have been developed to estimate rare event probabilities starting at the last decade. Lieber, Rubinstein and Elmakis ([72]) developed the Cross Entropy (CE) method as an adaptive technique for the estimation of reference parameters applied in Importance Sampling (IS) method. The CE method can be viewed as a model-based optimization technique, which involves two phases. (1) Generation of

a sample of random vectors according to a specified random mechanism. (2) Updating the parameters of the random mechanism, on the basis of the data, in order to produce a better sample in the next iteration. The significance of the CE method is that it defines a precise mathematical framework for deriving fast, and in some sense "optimal" updating rules, based on advanced simulation theory. Estimation of the probability of rare events is essential for guaranteeing that the performance of engineering systems is adequate. For example, consider a telecommunication system that accepts calls from many customers. Under normal operating conditions each client may be rejected with a very small probability. In order to estimate this small probability the system should be simulated under normal operating conditions for a long time. A better way to estimate this probability is to use IS, in which the system is simulated under a different set of parameters, so as to make the occurrence of the rare event more likely. A major drawback of the IS technique is that the optimal reference parameters to be used in IS are usually very difficult to obtain. The advantage of the CE method is that it provides a simple adaptive procedure for estimating the near optimal reference parameters, and the CE method enables simulating the system under irregular conditions and estimating the rejection probability for normal conditions. Moreover, the CE method also enjoys asymptotic convergence properties. The basic methodology behind the CE algorithm will be discussed. Rubinstein ([92]) and his collaborators applied CE algorithm for estimation of rare event probabilities in stochastic networks with exponential distribution (see De Boer, Kroese, Mannor and Rubinstein ([10]). We test this simulation technique also for medium sized stochastic networks and compare its effectiveness to the simple CMC simulation. The extension of CE method for estimation of rare event probabilities in stochastic networks with normal and beta distributions will be developed. In second case the calculation of reference parameters of the importance sampling distribution requires numerical solution of a nonlinear equation system. This is done by applying a Newton–Raphson iteration scheme. In this case the CPU time spent for calculation of the reference parameter values can not be neglected. The basic CE algorithm will be specialized for the shortest path problem with exponential, beta and normal distributed activity duration times. Two modifications of the basic CE algorithm for rare event sim-

ulation will be developed. The first modification is new result, the second was published by Homem-de Mello and Rubinstein ([56]). The effectiveness of a variance reduction simulation algorithm is measured in the following way. We calculate the product of the necessary CPU time and the estimated variance of the estimation. This product is compared to the same for the simple Crude Monte Carlo simulation. This was originally used for comparison of different variance reduction techniques by Hammersley and Handscombe ([55]). Numerical results for the comparison of CMC and IS, based on CE reference parameter estimation algorithms will be presented.

In the third part of the dissertation there will be given a stochastic programming approach to PERT modeling. Main drawback of the traditional PERT modeling is that the probabilistic characteristics determined for the finishing time of the project are only valid when it is supposed that any activity can be started promptly after finishing all of its predecessor activities. This is possible in the case of scheduling computer tasks, however it is impossible in the case of architectural project planning what is the most important application area of PERT modeling. A new PERT modeling technique to solve this problem will be introduced. This modeling will produce deterministic earliest starting times for the activities of the project. These deterministic starting times will be attainable with prescribed probability. So we also get an estimated finishing time of the project what is realizable with the same prescribed probability. As the random activity duration times in PERT are supposed to be independent and beta distributed, the application of the multivariate Dirichlet distribution is plausible in this context. The code developed for Dirichlet probability calculations can be incorporated into the AMPL modeling language environment. Moderate sized numerical examples will be given for comparing the traditional and the newly introduced PERT modeling techniques.

Acknowledgements

First all thanks are due to ALLAH the beneficent and merciful. I would like to express special thanks, deepest gratitude and appreciation to my supervisor Prof. Dr. Tamás Szántai for his supervision, unlimited guidance, continuous valuable help, instructive criticism and generous support during my learning of many computer programs and algorithms, and the preparation of this dissertation. I would like to express my thanks to Prof. István Deák and my sincere thanks to Dr. Mihály Hujter for his kind help. I wish also to thank Prof. Miklós Farkas and Prof. Éva Gyurkovics. I would like to acknowledge all staff of the department of differential equations, especially Prof. Péter Moson, Prof. József Fritz, Prof. Barna Garay, Dr. Gergely Madi-Nagy and Irénke Szabó. I would like to thank Hungarian Ministry of Education for granting a scholarship, which enable me to accomplish this work as well as Egyptian Ministry of Higher Education represented in Egyptian Cultural Bureau, Budapest, for providing the financial support during the period of the study.

List of Acronyms

AOA	Activity-On-Arrow
AON	Activity-On-Node
CE	Cross-Entropy method
CMC	Crude Monte Carlo method
CPM	Critical Path Method
IS	Importance Sampling
PERT	Program Evaluation and Review Technique
SCS	Sequential Conditioned Sampling method
SCIS	Sequential Conditioned Importance Sampling method

List of Symbols

cdf	cummulative distribution function
pdf	probability density function

F_Z	cdf of the random variable Z
f_Z	pdf of the random variable Z

F_Q	cdf of a random variable having distribution Q
f_Q	pdf of a random variable having distribution Q

$(X_1, \ldots, X_k) \sim Q(p_1, \ldots, p_s)$	$(X_1, \ldots, X_k)$has the joint distribution Q with parameters $p_1, \ldots, p_s$
$F(x_1, \ldots, x_k)$	the value of the joint cdf of the random variables $(X_1, \ldots, X_k)$ at $(x_1, \ldots, x_k)$
$f(x_1, \ldots, x_k)$	the value of the joint pdf of the random variables $(X_1, \ldots, X_k)$ at $(x_1, \ldots, x_k)$
$\mu_{r_1,\ldots,r_k}(X_1, \ldots, X_k)$	the $r_1, \ldots, r_k$th mixed moment of the random variables $(X_1, \ldots, X_k)$

$E[f]$	The expectation value of the function f
$E[X]$	The expectation value of the random variable X
$E_g[f]$	The expectation value of f over the pdf g
$E_v[f]$	The expectation value of f over a distribution defined by the parameter v
μ_x	The expectation value of the random variable X
$Var(X)$	The variance of the random variable X
$Cov(X, Y)$	The covariance of the random variables X and Y
$S(X)$	The score function of the random variable X
$W(X, u, v)$	The ratio of the probability of random variable X in the distributions defined by the parameters u and v
$D(f, g)$	The Cross-Entropy distance between distributions g and h

J	Jacobian of a transformation
x^T	transpose of the vector x

$(\lambda, s) := \frac{\Gamma(\lambda+s)}{\Gamma(\lambda)} = \lambda(\lambda+1)\ldots(\lambda+s-1)$, where $\Gamma(n) = \int_0^\infty t^{n-1}\exp(-t)\,dt$, noting, that for this Γ function $\Gamma(n+1) = n\Gamma(n)$ and for every n natural number $\Gamma(n+1) = n!$ holds.

Contents

1 Introduction

This thesis is about the estimation of probabilities according to multivariate distributions and its applications. In this introductory chapter, some definitions, notations, background and motivation for this work is provided, and an outline of the thesis' content presented.

Approximation of multivariate probability integral is a hard problem in general. However, if the domain of probability integral is the multidimensional interval, then the problem reduces to the approximation of multivariate probability distribution function values. The evaluation of probabilities according to multivariate distribution is an important problem in many applications of statistics and related fields. Over the years, several methods have been proposed for the computation of probabilities according to multivariate distributions which can be classified into broad categories, numerical integration, numerical approximations, bounding techniques and simulation. While several methods have been proposed for the computation of multivariate normal probabilities, see for example Plackett ([83]), Szántai ([106], [107], [110]), Genz ([40], [41]), Prékopa ([87]), Ambartzumian et al. ([1]), Pandey ([80]), Gassmann ([38]), Gassmann, Deák and Szántai ([39]), Kotz, Balakrishnan and Johnson ([65]) and Pandey and Sakar ([81]), only a few paper is dealing with the computation of Dirichlet probabilities. Yassaee ([121], [122], [123]) computes the probability integral of Dirichlet distribution by computing the probability integral of inverted Dirichlet distribution, Szántai ([105], [109]), used variance reduction simulation techniques to estimate the probability of Dirichlet distribution. For more details see Wilks ([119]), Prékopa ([87]), Kotz, Balakrishnan and Johnson ([65]).

The Dirichlet distribution is one of the important multivariate distributions that appears in many applications, in order statistics, probabilistic constrained programming models, delivery problems. Such applications may be found in Prékopa and Kelle ([84]), Prékopa ([87]), Tiao and Guttman ([112]), Johnson ([61]), Sobel and Uppuluri ([104]), Phillips ([82]), Fabius ([31]), James ([60]), Chotikapanich and Griffiths ([19]) and Goodman and Nguyen ([44]). Another application is the modeling of consumer purchasing behavior for non durable items such as foods and toiletries (Narayanan ([79]) and Chat-

field and Goodhardt ([17])). A model for multibrand purchasing behavior is given for the use of multivariate beta distribution for the independent case. Suppose this is a product field with k brands and let the random variables Z_i represent the average rate of purchase of brand i let $W = \sum_{i=1}^{k} Z_i$ represent a consumer's rate of buying of the product field as a whole. Then the joint distribution of $(X_1, X_2, ..., X_{k-1})$ follows a Dirichlet distribution where $X_i = \frac{Z_i}{W}$ represents the proportion of a consumer's total purchases devoted to brand i. Another application would be modeling the activity times in a PERT (Program Evaluation and Review Technique) network. A PERT network has a collection of activities and each activity is usually modeled as a random variable following a beta distribution. A Dirichlet distribution for the entire network follows directly since each marginal distribution of Dirichlet is a beta. Using the properties of the Dirichlet distribution we can see that any subnetwork will follow a Dirichlet distribution. Monhor ([75], [76]) uses the Dirichlet distribution for modeling the activity times of a PERT network and derive's an upper bound for the completion time of the project. More details about the applications of Dirichlet distribution can be found in Kotz, Balakrishnan and Johnson ([65]).

Rare event simulation. Stochastic simulation has proven itself in practice; it is commonly used for accurate estimations in many problems. However, there are some limitations to the standard stochastic simulation method in many cases. An important class of problems that cannot be efficiently solved using standard simulation is that involving rare events. Since these rare events occur so seldom in a standard simulation the simplest method to estimate the rare event probability is the Crude Monte-Carlo (CMC) simulation method but it needs a very large sample size, what need too much CPU time. This is why different methods and techniques have been developed to estimate rare event probabilities starting at the last decade. Lieber, Rubinstein and Elmakis ([72]) developed the Cross Entropy (CE) method as an adaptive technique for the estimation of reference parameters applied in Importance Sampling (IS) method. CMC, CE and IS are described in some more detail below.

Crude Monte Carlo simulation. The rare event probability estimation problem is hard to do with CMC simulation when the desired probability, which we will call, is

extremely small. In this case the CMC simulation method is very inefficient and simulation takes a very long time. The CMC method consists of simulating the system without making any changes to its stochastic behavior; as a consequence, very few samples will actually hit the rare event. Let we have a sample of size n $(X_1, \ldots, X_n)$, say, random observations in which the rare event A may occur; let $P(A) = l$ and the value of a random variable X_i equals one when the rare event is seen in the i-th attempt, and equals zero otherwise. The CMC estimator is simply

$$\hat{l} = \frac{1}{n} \sum_{i=1}^{n} X_i,$$

and the variance of a CMC estimator using n samples is equal to

$$Var\left(\hat{l}\right) = \frac{l\,(1-l)}{n}.$$

Importance Sampling simulation. Importance sampling is typically presented as a method for reducing the variance of the estimate of an expectation by carefully choosing a sampling distribution (Rubinstein ([91])). For example, the most direct method for evaluating $E_g[f(\eta)] = \int f\,(x)\,g\,(x)\,dx$ is to sample independent identical distribution $x_i \sim g\,(x)$ and use $\frac{1}{n}\sum_i f\,(x_i)$ as the estimate. However, by choosing a different distribution $q(x)$ which has higher density in the places where $|f(x)|$ is larger, we can get a new estimate which is still unbiased and has lower variance. In particular, we can draw $x_i \sim q\,(x)$ and use $\frac{1}{n}\sum_i f\,(x_i)\,\frac{g\,(x_i)}{q\,(x_i)}$ which is like approximating $\int f\,(x)\,\frac{g\,(x)}{q\,(x)} q\,(x)\,dx$ with samples drawn from $q(x)$. If $q(x)$ is chosen properly, our new estimate has lower variance. It is always unbiased provided that the support of $g(x)$ and $q(x)$ are the same. In this thesis we always have the same support. The density function $q(x)$ will be choosen from the parametric family of $g(x)$ and $q(x)$ to reduce variance, the CE method will be used to estimate the reference parameters of $q(x)$ which will achieve minimal variance.

Cross Entropy method. Rubinstein ([92]) and Lieber, Rubinstein and Elmakis ([72]) developed the CE method as an adaptive technique for the estimation of reference parameters applied in IS variance reduction technique. The CE method can be viewed as a model-based optimization technique, which involves two phases. (1) Generation of a sample of random vectors according to a specified random mechanism. (2) Updating

the parameters of the random mechanism, on the basis of the data, in order to produce a better sample in the next iteration. The significance of the CE method is that it defines a precise mathematical framework for deriving fast, and in some sense "optimal" updating rules, based on advanced simulation theory. Estimation of the probability of rare events is essential for guaranteeing that the performance of engineering systems is adequate. For example, consider a telecommunication system that accepts calls from many customers. Under normal operating conditions each client may be rejected with a very small probability. In order to estimate this small probability the system should be simulated under normal operating conditions for a long time. A better way to estimate this probability is to use IS, in which the system is simulated under a different set of parameters, so as to make the occurrence of the rare event more likely. A major drawback of the IS technique is that the optimal reference parameters to be used in IS are usually very difficult to obtain. The advantage of the CE method is that it provides a simple adaptive procedure for estimating the near optimal reference parameters, and the CE method enables simulating the system under irregular conditions and estimating the rejection probability for normal conditions. Moreover, the CE method also enjoys asymptotic convergence properties.

PERT models. The original PERT technique, developed by Malcolm et al. ([73]), is a technique to approximate the expected duration of the project. PERT networks have been used extensively in the business world. Analysis of PERT networks, also known as stochastic activity networks, has received considerable attention in the literature (Elmaghraby, [27]). PERT is based on the concept that a project is divided into a number of activities which are arranged in some order according to the job requirements. A PERT network consists of a set of nodes and arcs, where a node represents the beginning or completion of one or more activities and an activity is represented by an arc (arrow) connecting two nodes in activity-on-arrow (AOA) representation. Activity-on-node (AON) representations have also been used. We use the AOA representation in this thesis. The project starts at the initial node and ends at the terminal node. A path is a set of nodes connected by arrows which begin at the initial node and end at the terminal node. This collection of arcs, nodes and paths is collectively called an activity

network. A project is deemed complete if work along all paths is complete. After the development of the network, the next major planning step is the estimation of activity and project times. Typical methods for estimating activity times have been to use point estimates or some sort of ranges or distributions. The type of method used depends on the situation facing the project manager. Hershauer and Nabielsky ([54]) categorize the situations into three major categories, viz., certainty, risk and uncertainty. They further subdivide these categories based on availability of knowledge regarding the mode, range and distribution on the time estimates. They then map the situation and estimations to the appropriate methods to be adopted. If activity times are deterministic, the duration of the project completion time is determined by the length of the longest path in the network. For a stochastic activity network, Kulkami and Adlakha ([70]) have identified three important measures of performance: (a)Distribution of the project completion time. (b) The probability that a given path is critical, also known as the "path criticality index". (c) The probability that a given activity belongs to a critical path, also known as the "activity criticality index". Performance measures derived from (a) are the most commonly used; measures and most studies have concentrated on the properties of the completion time of the project Dodin and Sirvanci ([23]), Kamburowski ([62]), Sculli ([99]) among others. Determination of the exact distribution of the completion time of the project is complicated by the fact that different paths are correlated and also because of the need to find the maximum of a set of random variables, as we shall see later. Hence one cannot easily determine the exact distribution of the completion time of the project. The research has branched off primarily in three directions: (1) Exact methods, Martin ([77]), Dodin ([22]), Fisher et al. ([32]), Hagstrom ([53]) and Kulkami and Adlakha ([70]) are some of the papers that deal with these methods. Most of their results are limited in that they make quite restrictive assumptions. For example Martin ([77]) assumes that the arc duration density functions are polynomial. Hagstrom ([53]) assumes task durations have discrete distributions. (2) Approximating and bounding approaches. These have been the most abundant in the literature. Malcolm et al. ([73]). Sculli ([99]) Golenko-Ginzburg [45]), Dodin ([24]), Sculli and Wong ([100]), and Dodin and Sirvanci ([23]) determine approximations for the distribution and moments of the

completion time of the project. Kamburowski ([63]), Shogan ([102]), Kleindorfer ([68]) and Robillard and Trahan ([90]), on the other hand, try to find upper and lower bounds for the distributions and moments of the completion time of the project and Prékopa, Long and Szántai ([86]) describe new bounds and approximations for the probability distribution of the length of the critical path. (3) Simulation methods. These methods have been discussed in the literature by Van Slyke ([116]), Burt and Garman ([16]) and Sigal et al.([103]). Simulation provides a powerful methodology to obtain desired statistics for any network with specified distribution of activities. To obtain reliable results, however, it may be necessary to repeat the experiments several times. Main drawback of the traditional PERT modeling is that the probabilistic characteristics determined for the finishing time of the project are only valid when it is supposed that any activity can be started promptly after finishing all of its predecessor activities. This is possible in the case of scheduling computer tasks, however it is impossible in the case of architectural project planning what is the most important application area of PERT modeling. We shall introduce new PERT modeling technique to solve this problem in Chapter 3.

Outline of the dissertation. The computation of the cumulative distribution function values of Dirichlet distribution by using several algorithms are to be described and probability estimation in network models via cross entropy method and completion time estimation in PERT network are to be investigated.

In Chapter 2 the estimation of probabilities according to Dirichlet distribution will be described. A recursive algorithm for the calculation of Dirichlet probability distribution function values up to 7 dimensions will be developed. This procedure is based on a generalization of Szántai's result published in his dissertation for candidate degree of HAS and the Lauricella series expansion. This gives the possibility of application all algorithms for bounding and estimating multivariate normal probability distribution function values developed before by J. Bukszár, A. Prékopa and T. Szántai. In stochastic programming applications one need the gradient vector and Hessian matrix of the multivariate probability distribution function, too. The gradient vector calculation was developed by T. Szántai, in this chapter a new algorithm for the Hessian matrix calculation will be given. All these estimations are most effective when the esti-

mated probability value is close to one. However many times one need to estimate small probability values, too. These are called rare event probabilities in the literature, the Sequential Conditioned Sampling (SCS) and Sequential Conditioned Importance Sampling (SCIS) algorithms will be developed to estimate the rare event probabilities of Dirichlet distribution. Using Szántai's result published in his dissertation for candidate degree of HAS there will be developed new versions of these algorithms, called SCSA, SCSB, SCISA and SCISB, respectively. The efficiency of the new methods will be compared to CMC simulation method. The efficiency of all algorithms for the calculation of Dirichlet probability distribution will be compared to CMC. Numerical test results will also be presented.

In Chapter 3 the estimation of rare event probabilities in stochastic networks with exponential, beta and normal distributions will be described. The basic methodology behind the CE algorithm will be discussed. Two modifications of the basic CE algorithm for rare event simulation will be discussed. The first modification is new result, the second was published by Homem-de Mello and Rubinstein ([56]). Numerical results for the comparison of CMC and IS, based on CE reference parameter estimation algorithms will be presented. A stochastic programming approach to PERT modeling will be given. This modeling will produce deterministic earliest starting times for the activities of the project. These deterministic starting times will be attainable with prescribed probability. So we also get an estimated finishing time of the project what is realizable with the same prescribed probability. As the random activity duration times in PERT are supposed to be independent and beta distributed, the application of the multivariate Dirichlet distribution is plausible in this context. The code developed for Dirichlet probability calculations can be incorporated into the AMPL modeling language environment. Moderate sized numerical examples will be given for comparing the traditional and the newly introduced PERT modeling techniques.

New scientific results

1. A recursive algorithm for the calculation of Dirichlet probability distribution function values up to 7 dimensions was developed.

2. The hypermultitree bounding algorithm and a variance reduction simulation procedure based on these bounds for the calculation of higher dimensions of Dirichlet probability distribution function values was developed.

3. A new algorithm to calculate the Hessian matrix of Dirichlet probability distribution function values and the formulae for the calculation of the first and second order partial derivatives were developed.

4. New sampling techniques as Sequential Conditioned Sampling (SCS), the Sequential Conditioned Importance Sampling (SCIS) algorithms and modified versions called SCSA, SCSB, SCISA and SCISB algorithms to calculate of Dirichlet probability distribution function values were introduced.

5. The application of the basic CE algorithm for the shortest path problem with normal and beta distributed activity duration times was developed.

6. Development of a modification of the basic CE algorithm for rare event simulation.

7. A new stochastic programming approach and the application of the multivariate Dirichlet distribution to PERT modeling was developed.

8. The traditional and the newly developed PERT modeling techniques were compared on larger sized numerical examples than it was published before.

2 Estimation of Probabilities According to Dirichlet Distribution

The main numerical difficulty in probabilistic constrained stochastic programming problems is the calculation of the probability values according to the underlying multivariate probability distributions. From point of view of the nonlinear programming algorithms to be applied it is preferable to be able to calculate the first and second order partial derivatives of these probability functions according to the decision variables.

In this chapter there will be given a solution to the above problems in the case of Dirichlet distribution. For the calculation of the cummulative distribution function values, the Lauricella function series expansions will be applied up to 7 dimensions. For higher dimensions we propose the hypermultitree bound calculations and a variance reduction simulation procedure based on these bounds. There will be given formulae for the calculation of the first and second order partial derivatives, too. The common property of these formulae is that they involve only lower dimensional cummulative distribution function calculations. For estimation of the small probability values new sampling techniques as Sequential Conditioned Sampling (**SCS**) and Sequential Conditioned Importance Sampling (**SCIS**) will be introduced. On the base of an interesting property of the Dirichlet distribution new versions of the **SCS** and **SCIS** algorithms will be developed, called **SCSA**, **SCSB**, **SCISA** and **SCISB**, respectively. **SCIS** and modified algorithms need more CPU time but they result significant variance reduction. Their resultant efficiency will be compared to the simple " hit-or-miss Monte Carlo" simulation method with conventional sampling of the Dirichlet distributed random vectors what we call **CMC** simulation method. Numerical test results will also be presented.

2.1 Introduction

2.1. Definition *We shall refer to the distribution defined by the following probability density function (pdf)*

$$f(x) = \begin{cases} \dfrac{\Gamma(\vartheta_1+\vartheta_2)}{\Gamma(\vartheta_1)\Gamma(\vartheta_2)} x^{\vartheta_1-1}(1-x)^{\vartheta_2-1}, & \text{for } 0<x<1, \\ 0, & \text{otherwise,} \end{cases} \tag{2.1}$$

where $\vartheta_1 > 0, \vartheta_2 > 0$, *as the beta distribution* $Be(\vartheta_1, \vartheta_2)$.

The Dirichlet distribution is the multivariate generalization of the beta distribution.

2.2. Definition *The random variables* $X_1, X_2, ..., X_k$ *have* $k-$*dimensional Dirichlet distribution with parameters* $\vartheta_1 > 0, \ldots, \vartheta_k > 0, \vartheta_{k+1} > 0$, *if their joint pdf is*

$$f(x_1,\ldots,x_k) = \frac{\Gamma\left(\sum_{j=1}^{k+1}\vartheta_j\right)}{\prod_{j=1}^{k+1}\Gamma(\vartheta_j)} \prod_{j=1}^{k} x_j^{\vartheta_j-1}\left(1-\sum_{j=1}^{k}x_j\right)^{\vartheta_{k+1}-1}, \tag{2.2}$$

over the simplex: $S_k : \left\{(x_1,\ldots,x_k) : x_i \geq 0, \quad i=1,\ldots,k, \quad \sum_{j=1}^{k} x_j \leq 1\right\}$.

2.3. Definition *Some authors (see [42]) define the* $k-$*dimensional Dirichlet distribution with the following pdf:*

$$f(x_1,\ldots,x_k) = \frac{\Gamma\left(\sum_{j=1}^{k}\vartheta_j\right)}{\prod_{j=1}^{k}\Gamma(\vartheta_j)} \prod_{j=1}^{k-1} x_j^{\vartheta_j-1}\left(1-\sum_{j=1}^{k-1}x_j\right)^{\vartheta_k-1}, \tag{2.3}$$

at any point in the simplex $S_k' : \left\{(x_1,\ldots,x_k) : x_i \geq 0, \quad i=1,\ldots,k, \quad \sum_{j=1}^{k} x_j = 1\right\}$.

In the particular case when $\vartheta_1 = \vartheta_2 = \ldots = \vartheta_k = a$, we obtain the k-dimensional symmetric Dirichlet distribution whose density function is given by

$$f(x_1,\ldots,x_k) = (\Gamma(a))^{-k}\Gamma(ka)\prod_{j=1}^{k-1} x_j^{a-1}\left(1-\sum_{j=1}^{k-1}x_j\right)^{a-1},$$

at any point in the *simplex* $S_k^{'}: \left\{(x_1,\ldots,x_k): x_i \geq 0, \quad i=1,\ldots,k, \quad \sum_{j=1}^{k} x_j = 1\right\}$.

Let $X_1, ..., X_k; X_{k+1}$ be independent random variables having standard *gamma* distributions with parameters $\vartheta_1, \ldots, \vartheta_k, \vartheta_{k+1}$ and

$$Y_j = \frac{X_j}{X_1 + \cdots + X_k + X_{k+1}}, \qquad j = 1,\ldots,k, \tag{2.4}$$

then $(Y_1, ..., Y_k)$ have a $k-$dimensional Dirichlet distribution, denoted by $D_k(\vartheta_1,\ldots,\vartheta_k;\vartheta_{k+1})$.

There is a natural deduction of the Dirichlet distribution from χ^2 distributions. Let $Y_1,\ldots,Y_k,Y_{k+1}$ be independent random variables distributed as χ^2 with $\vartheta_1,\ldots,\vartheta_k,\vartheta_{k+1}$degrees of freedom, respectively. The joint probability density function of $Y_j, j=1,\ldots,k+1$ is then

$$f\left(y_1,\ldots,y_k,y_{k+1}\right) = \begin{cases} \left(\prod_{j=1}^{k+1}\Gamma\left(\frac{\vartheta_j}{2}\right)\right)^{-1} 2^{-\sum_{j=1}^{k+1}\frac{\vartheta_j}{2}} \prod_{j=1}^{k+1} y_j^{\frac{\vartheta_j}{2}-1} \exp\left(-\sum_{j=1}^{k+1} y_j\right), & \text{for } 0 \leq y_j, \\ 0, & \text{otherwise.} \end{cases}$$

Now apply the transformation

$$X_i = Y_i / \sum_{j=1}^{k+1} Y_j \qquad (i=1,\ldots,k); \qquad X_{k+1} = \sum_{j=1}^{k+1} Y_j,$$

so that

$$Y_i = X_i X_{k+1} \qquad (i=1,\ldots,k); \qquad Y_{k+1} = X_{k+1}\left(1 - \sum_{j=1}^{k} X_j\right).$$

The Jacobian of this transformation is

$$\frac{\partial\left(y_1,\ldots,y_k,y_{k+1}\right)}{\partial(x_1,\ldots,x_k,x_{k+1})} = \left(x_{k+1}\right)^k,$$

and so the joint probability density function of $X_1,\ldots,X_k,X_{k+1}$ is

$$\begin{aligned} &f(x_1,\ldots,x_k,x_{k+1}) = \\ &= \frac{2^{-\sum_{j=1}^{k+1}\frac{\vartheta_j}{2}}}{\prod_{j=1}^{k+1}\Gamma\left(\frac{\vartheta_j}{2}\right)} \left[x_{k+1}\left(1-\sum_{j=1}^{k} x_j\right)\right]^{\frac{\vartheta_{k+1}}{2}-1} \prod_{j=1}^{k}\left(x_j x_{k+1}\right)^{\frac{\vartheta_j}{2}-1} \exp\left(-\frac{x_{k+1}}{2}\right)\left(x_{k+1}\right)^k. \end{aligned}$$

Integrating out the variable x_{k+1}, we find the pdf of $X_1, \ldots, X_k$ is

$$f(x_1, \ldots, x_k) = \frac{\Gamma\left(\sum_{j=1}^{k+1} \frac{\vartheta_j}{2}\right)}{\prod_{j=1}^{k+1} \Gamma\left(\frac{\vartheta_j}{2}\right)} \left(1 - \sum_{j=1}^{k} x_j\right)^{\frac{\vartheta_{k+1}}{2}-1} \prod_{j=1}^{k} x_j^{\frac{\vartheta_j}{2}-1}.$$

It is usual to replace $\frac{\vartheta_j}{2}$ by ϑ_j for all $j, j = 1, \ldots, k+1$. The (standard) Dirichlet distribution with parameters $\vartheta_1, \ldots, \vartheta_k, \vartheta_{k+1}$ has pdf (2.2).

Properties. The Dirichlet is a convenient distribution on the simplex: it is an exponential family and has finite sufficient statistics. The expected values, variances and covariances of the random variables $X_1, \ldots, X_k$ are

$$E(X_i) = \frac{\vartheta_i}{\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1}}, \quad 1, \ldots, k,$$

$$Var(X_i) = \frac{\vartheta_i (\vartheta_1 + \ldots + \vartheta_{i-1} + \vartheta_{i+1} + \ldots + \vartheta_k + \vartheta_{k+1})}{(\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1})^2 (\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1} + 1)},$$

$$Cov(X_i, X_j) = \frac{-\vartheta_i \vartheta_i}{(\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1})^2 (\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1} + 1)}, i, j = 1, \ldots, k, i \neq j.$$

Marginal Distribution. If the random variables $X_1, \ldots, X_n$ have n–variate Dirichlet distribution $D(\vartheta_1, \ldots, \vartheta_n; \vartheta_{n+1})$ then the marginal distribution of the random variables $X_1, \ldots, X_k$, $k < n$ is the k–variate Dirichlet distribution $D(\vartheta_1, \ldots, \vartheta_k; \vartheta_{k+1} + \cdots + \vartheta_n + \vartheta_{n+1})$. In the special case $k = 1$, the one dimensional marginal probability distribution is the beta distribution with parameters ϑ_1 and $\vartheta_2 + \cdots + \vartheta_n + \vartheta_{n+1}$.

Conditional Distribution. Let us regard the k–variate marginal Dirichlet distribution $D(\vartheta_1, \ldots, \vartheta_k; \vartheta_{k+1} + \cdots + \vartheta_n + \vartheta_{n+1})$. The conditional random variable X_k given the values of $X_1 = x_1, \ldots, X_{k-1} = x_{k-1}$ has the property that the transformed conditional random variable $\frac{X_k}{1 - x_1 - \cdots - x_{k-1}}$ given the values of $X_1 = x_1, \ldots, X_{k-1} = x_{k-1}$ has the beta distribution $B(\vartheta_k; \vartheta_{k+1} + \cdots + \vartheta_n + \vartheta_{n+1})$, i.e. its probability density function is

$$f\left(t;\vartheta_k;\vartheta_{k+1}+\cdots+\vartheta_n+\vartheta_{n+1}\right)=$$

$$=\frac{\Gamma(\vartheta_k+...+\vartheta_n+\vartheta_{n+1})}{\Gamma(\vartheta_k)\Gamma(\vartheta_{k+1}+\cdots+\vartheta_n+\vartheta_{n+1})}t^{\vartheta_k-1}\left(1-t\right)^{\vartheta_{k+1}+\cdots+\vartheta_n+\vartheta_{n+1}-1},$$

if $0 \leq t \leq 1$ and zero otherwise. The corresponding cdf is $F\left(x;\vartheta_k;\vartheta_{k+1}+\cdots+\vartheta_n+\vartheta_{n+1}\right)$ which is the incomplete beta function. The general form of the conditional Dirichlet distribution can given by the next theorem.

2.1. Theorem: *If* $\mathbf{X}=\left(\mathbf{X}_1',\mathbf{X}_2'\right)' \sim D_k\left(\vartheta_1',\vartheta_2'\right)$ *where* $\mathbf{X}_1$ *and* ϑ_1 *consist of* r *elements and* $\mathbf{X}_2$ *and* ϑ_2 *consist of* s *elements and* $k=r+s$, *then*

$$\left(1-\mathbf{1}'\mathbf{x}_2\right)^{-1}\mathbf{X}_1 \mid \mathbf{X}_2=\mathbf{x}_2 \sim D_r(\vartheta_1').$$

More details and proof of this theorem can be found in Wilks ([119]).

Dirichlet distribution can be used for imputation of multivariate missing item values (see [111]). In general there are two types of linear edit constraints:

$$\begin{aligned} c_1X_1+\cdots+c_kX_k &= X_{k+1}, \\ c_1X_1+\cdots+c_kX_k &\leq X_{k+1}, \end{aligned} \tag{2.5}$$

which are called balance and inequality edits, respectively. We will first consider balance edit constraints. We believe that we can impute the missing items, satisfying the edit constraint directly and preserving the distribution of the data, by making use of the Dirichlet distribution. Consider edit rule (2.5) and transform it by dividing the different parts by the total X_{k+1}, this is done in order to restrict the domain of the variables $\tilde{X}_1,\ldots,\tilde{X}_k$ to the simplex; we take $\tilde{X}_i=c_iX_i/X_{k+1}$.

$$\begin{aligned} \frac{c_1X_1}{X_{k+1}}+\cdots+\frac{c_kX_k}{X_{k+1}}=1, \quad X_{k+1}>0 \\ \tilde{X}_1+\cdots+\tilde{X}_k=1. \end{aligned}$$

Note that we assume that the total, X_{k+1}, is known. This is done for two reasons. First of all since it is an aggregate the nonresponse rate will probably be low. And secondly, if it is indeed missing we expect to be able to estimate this value very well based on

the other variables in the survey, whereas the different subtotals are far more difficult to estimate this way. A special case arises when only one $\tilde{X}_i, \quad i = 1, \ldots, k$ is missing. In this instance deductive imputation can be used. Deductive imputation means that the value of the missing item can be established with certainty based on the other items in the survey. If all items of $\tilde{\mathbf{X}}$; are missing, we can obtain imputations by drawing from $D_n(\vartheta_1, \ldots, \vartheta_n;\ \vartheta_{n+1})$. However, a common circumstance is that a few item values are missing and the others are observed. In this case one needs to draw values from the conditional distribution of the missing items given the observed ones, which is also a Dirichlet distribution Partition $\tilde{\mathbf{X}}$ in $\tilde{\mathbf{X}}^{mis}$ and $\tilde{\mathbf{X}}^{obs}$, where $\tilde{\mathbf{X}}^{mis}$ represents the missing items and $\tilde{\mathbf{X}}^{obs}$ represents the observed items. The vector with missing, $\tilde{\mathbf{X}}^{mis}$, consists of m elements and $\tilde{\mathbf{X}}^{obs}$ consists of o elements, which are the number of observed items and $k = m + o$. Partition ϑ accordingly. Then it holds that

$$\left(1 - 1'\tilde{\mathbf{x}}^{obs}\right)^{-1} \tilde{\mathbf{X}}^{mis} \mid \tilde{\mathbf{X}}^{obs} = \tilde{\mathbf{x}}^{obs} \sim D_m(\vartheta_1, \ldots, \vartheta_m; \vartheta_{m+1}). \tag{2.6}$$

Thus imputations for missing items can be obtained by drawing from the conditional Dirichlet distribution (2.6).

Concavity. Prékopa ([88]) states the following theorem of Dirichlet distribution, which is important in probabilistic constrained programming problems.

2.2. Theorem: *The probability distribution of a k–variate Dirichlet distribution with parameters* $\vartheta_1, \ldots, \vartheta_k; \vartheta_{k+1}$ *is concave in the set* $\{t \mid t \geq 0\}$ *if* $\alpha \leq 0$ *and in* $\{t \mid t_i \geq \alpha, i = 1, \ldots, k\}$, *if* $\alpha > 0$, *where* $\alpha = \dfrac{\vartheta_1 + \ldots + \vartheta_k - 1}{\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1} - 2} \leq 1$, *(if* $\vartheta_1 + \ldots + \vartheta_k + \vartheta_{k+1} - 2 \neq 2$*).*

More details about the concavity of multivariate probability distribution functions and proof of this theorem can be found in Prékopa ([88]).

Logconcavity and Logconvexcty. From (2.2) if $\vartheta_1 \geq 1, \ldots, \vartheta_{k+1} \geq 1$,then

$$\log f\,(x_1, \ldots, x_k) = \log K + \sum_{j=1}^{k} (\vartheta_j - 1) \log x_j + (\vartheta_{k+1} - 1) \log(1 - \sum_{i=1}^{k} x_i),$$

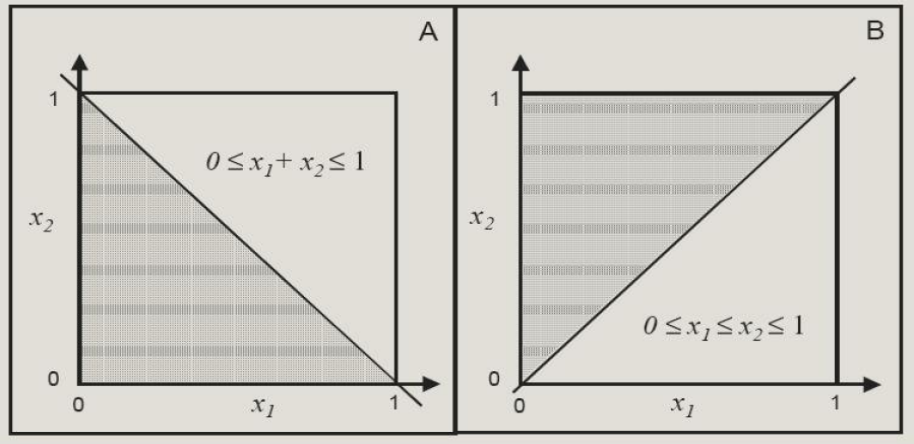

Figure 1: A: Support of a Bivariate Dirichlet Distribution B: Support of a Bivariate Ordered Dirichlet Distribution

where, $K = \dfrac{\Gamma(\vartheta_1 + \ldots + \vartheta_{k+1})}{\Gamma(\vartheta_1) \cdots \Gamma(\vartheta_{k+1})}$, is a concave function in the convex set where $f(x_1, \ldots, x_k) > 0$. In fact, the sum of concave functions is also concave and since the concavity of the first k terms is trivial, we have to remark that an increasing concave function of a concave function ($1 - x_1 - \cdots - x_k$ in this case) is also concave. Thus, $f(x_1, \ldots, x_k)$ is logconcave in R^k. If $\vartheta_1 \le 1, \ldots, \vartheta_{k+1} \le 1$, then $f(x_1, \ldots, x_k)$ is logconvex in the convex set where $f(x_1, \ldots, x_k) > 0$ (see Prékopa ([87])).

Distributions derived from Dirichlet:

Inverted Dirichlet distribution. Let $(\xi_1, \ldots, \xi_n)^T$ be Dirichlet distributed random vector, then the transformed random variables

$$\eta_i = \frac{\xi_i}{1 - \xi_1 - \cdots - \xi_n}, \quad i = 1, \ldots, n$$

define a random vector $(\eta_1, \ldots, \eta_n)^T$, which probability distribution is called inverted Dirichlet distribution. Its pdf is:

$$g(y_1, \ldots, y_n) = \frac{\Gamma(\vartheta_1 + \ldots + \vartheta_{n+1})}{\Gamma(\vartheta_k) \cdots \Gamma(\vartheta_{n+1})} y_1^{\vartheta_1 - 1} \cdots y_n^{\vartheta_n - 1} (1 + y_1 + \cdots y_n)^{-\sum_{i=1}^{n+1} \vartheta_i},$$
$$0 < y_i < \infty, \quad i = 1, \ldots, n.$$

Ordered Dirichlet distribution. Let $(\xi_1, \ldots, \xi_n)^T$ be Dirichlet distributed random vector, then the transformed random variables

$$\begin{aligned}
\eta_1 &= \xi_1, \\
\eta_1 &= \xi_1 + \xi_2, \\
&\vdots \\
\eta_1 &= \xi_1 + \xi_2 + \cdots + \xi_n,
\end{aligned}$$

define a random vector $(\eta_1, \ldots, \eta_n)^T$, which probability distribution is called ordered Dirichlet distribution. Its pdf is:

$$g(y_1, \ldots, y_n) = \frac{\Gamma(\vartheta_1 + ... + \vartheta_{n+1})}{\Gamma(\vartheta_k) \cdots \Gamma(\vartheta_{n+1})} y_1^{\vartheta_1 - 1} (y_2 - y_1)^{\vartheta_2 - 1} \cdots (y_n - y_{n-1})^{\vartheta_n - 1} (1 - y_n)^{\vartheta_{n+1} - 1},$$
$$0 < y_1 < \cdots < y_n < 1.$$

The problem considered in this chapter is that of finding the probability of a Dirichlet random vector over a cube. Let $(\mathbf{a}, \mathbf{b}) = (a_1, b_1) \times (a_2, b_2) \times \ldots \times (a_k, b_k)$ be a k–dimensional cube. The problem is then to find

$$\int_{a_1}^{b_1} \cdots \int_{a_k}^{b_k} \frac{\Gamma(\vartheta_1 + ... + \vartheta_k + \vartheta_{k+1})}{\Gamma(\vartheta_1) \cdots \Gamma(\vartheta_k) \Gamma(\vartheta_{k+1})} \prod_{j=1}^{k} t_j^{\vartheta_j - 1} \left(1 - \sum_{j=1}^{k} t_j\right)^{\vartheta_{k+1} - 1} dt_k \cdots dt_1, \qquad (2.7)$$

if $\mathbf{a} = (0, \ldots, 0), \mathbf{b} = (x_1, \ldots, x_k)$ and $0 \leq x_1 \leq 1, \ldots, 0 \leq x_k \leq 1$, then the problem is to find the cumulative distribution function (cdf) of Dirichlet distribution.

The problem of computation probabilities according to Dirichlet distribution has a little work in the literature and only a few solution methods have been proposed. Exton ([30]) used Lauricella series of the first type, Yassaee ([121], [122], [123]) proposed a procedure to evaluate the inverted Dirichlet distribution function and used the inverted Dirichlet distribution to evaluate Dirichlet distribution. Szántai ([105], [109]) proposed a variance reduction simulation algorithm and applied it for the numerical solution of probabilistic constrained stochastic programming problems.

The cumulative distribution function (cdf) of Dirichlet distribution is

$$F(x_1,\dots,x_k;\vartheta_1,\dots,\vartheta_k;\vartheta_{k+1}) =$$

$$= \frac{\Gamma(\vartheta_1+\ldots+\vartheta_k+\vartheta_{k+1})}{\Gamma(\vartheta_1)\cdots\Gamma(\vartheta_k)\Gamma(\vartheta_{k+1})}\times \tag{2.8}$$

$$\times \int_0^{x_1}\cdots\int_0^{x_k} t_1^{\vartheta_1-1}\cdots t_k^{\vartheta_k-1}(1-t_1-\cdots-t_k)^{\vartheta_{k+1}-1}dt_k\cdots dt_1,$$

if $0 \le x_1 \le 1,\dots,0 \le x_k \le 1$ and in all other cases or the cdf has one of the trivial values zero and one, or it can be expressed with lower dimensional Dirichlet cdf's. For simplicity, we will omit the parameters in the cdf of the Dirichlet distribution, i.e. instead of $F(x_1,\dots,x_n;\vartheta_1,\dots,\vartheta_n;\vartheta_{n+1})$ simply write $F(x_1,\dots,x_n)$. So $F(x_i)$ and $F(x_i,x_j)$ will simply designate the marginal cdf's of X_i and X_i, X_j, respectively. As the Dirichlet probability density function equals zero out of the unit simplex, by the application of the inclusion–exclusion formula easy to prove the following theorem (see Szántai ([105]) and Prékopa ([87])).

2.3. Theorem: *Let $x^{(1)} \le \cdots \le x^{(n)}$ be the ordered sequence of $x_1,\dots,x_n$, the arguments of the cdf of a Dirichlet distribution.*

Case (a) *If $x^{(1)}+x^{(2)} > 1$, then we have*

$$F(x_1,\ldots,x_n) = 1-n+\sum_{i=1}^{n} F(x_i). \tag{2.9}$$

Case (b) *If $x^{(1)}+x^{(2)}+x^{(3)} > 1$, then we have*

$$\begin{aligned} F(x_1,\ldots,x_n) &= \frac{1}{2}(n-1)(n-2)-(n-2)\sum_{i=1}^{n} F(x_i) \\ &+ \sum_{1\le i<j\le n} F(x_i,x_j). \end{aligned} \tag{2.10}$$

Proof. Let be $\bar{A}_i = \{X_i > x_i\}$, $i = 1,\dots,n$ then by the inclusion exclusion formula we have

$$F(x_1,\dots,x_n;\nu_1,\dots,\nu_n;\nu_{n+1}) = P(A_1\cdots A_n) = 1-\sum_{i=1}^{n} P\left(\bar{A}_i\right) +$$
$$+\sum_{i<j} P\left(\bar{A}_i\bar{A}_j\right) - \sum_{i<j<k} P\left(\bar{A}_i\bar{A}_j\bar{A}_k\right) + \cdots + (-1)^n P\left(\bar{A}_i\cdots\bar{A}_n\right),$$

As we know that all the marginal distributions of a Dirichlet distribution are also Dirichlet distributions in case (a) we have

$$F(x_1,\ldots,x_n;\nu_1,\ldots,\nu_n;\nu_{n+1}) = P\left(A_1\cdots A_n\right) = \\ = 1-\sum_{i=1}^{n} P\left(\bar{A}_i\right) = 1-n+\textstyle\sum_{i=1}^{n} F_i\left(x_i\right),$$

where all the remaining terms equal zero, and in case (b) we have

$$F(x_1,\ldots,x_n;\nu_1,\ldots,\nu_n;\nu_{n+1}) = P\left(A_1\cdots A_n\right) = \\ = 1-\sum_{i=1}^{n} P\left(\bar{A}_i\right)+\sum_{1\le i<j\le n} P\left(\bar{A}_i\bar{A}_j\right) = 1-n+ \\ +\textstyle\sum_{i=1}^{n} F_i(x_i)+\binom{n}{2}-(n-1)\sum_{i=1}^{n} F_i(x_i)+\sum_{1\le i<j\le n}^{n} F_{ij}(x_i,x_j) = \\ = \frac{1}{2}(n-1)(n-2)-(n-2)\sum_{i=1}^{n} F_i(x_i)+\sum_{1\le i<j\le n}^{n} F_{ij}(x_i,x_j),$$

where all the remaining terms equal zero.

In this chapter, the formulae (2.9) and (2.10) will be generalized and a recursive algorithm for the calculation of Dirichlet cumulative distribution function values up to 7 dimensions will be developed in Section 2. This procedure is based on the Lauricella function series expansion and the inclusion–exclusion formula. In Section 3 we propose the calculation of the hypermultitree lower and upper bounds developed by Bukszár ([14]) for higher dimensions. These bounds need the calculation of low (say up to seven) dimensional marginal cdf values and many times provide a sufficiently tight interval for the exact value of the higher dimensional Dirichlet cumulative distribution function value. If this interval is not tight enough we propose the application of a variance reduction simulation procedure based on these bounds. This technique was proposed for estimating multivariate normal probability distribution function values by Szántai ([110]). The application of the recursive algorithm and the hypermultitree bounding technique will be presented on numerical test problems. The efficiency of the proposed simulation procedure will be compared to CMC simulation method. In Section 4 the calculation of the first and second order partial derivatives will be given. The common property of these formulae is that they involve only lower dimensional cummulative distribution function calculations. In Section 5 **SCS** and **SCIS** will be introduced. On the base of an interesting property of the Dirichlet distribution new versions of the **SCS**

and **SCIS** algorithms will be developed, called **SCSA**, **SCSB**, **SCISA** and **SCISB**, respectively. Their resultant efficiency will be compared to CMC simulation method with conventional sampling of the Dirichlet distributed random vectors.

2.2 The recursive algorithm by Lauricella function

For the evaluation of Dirichlet cumulative distribution function $F(x_1,\ ...,x_n;\vartheta_1,\ \ldots,\vartheta_n;\vartheta_{n+1})$ on the unit simplex, we can use the following series expansion proposed by Exton ([30])

$$\begin{aligned} &F(x_1,...,x_n;\vartheta_1,\ldots,\vartheta_n;\vartheta_{n+1}) = \\ &= \frac{\Gamma(\vartheta_1+...+\vartheta_n+\vartheta_{n+1})}{\Gamma(\vartheta_1)\cdots\Gamma(\vartheta_n)\Gamma(\vartheta_{n+1})}\frac{x_1^{\vartheta_1}}{\vartheta_1}\cdots\frac{x_n^{\vartheta_n}}{\vartheta_n}\times \\ &\times F_A^{(n)}(1-\vartheta_{n+1},\vartheta_1,\ldots,\vartheta_n;\vartheta_1+1,\ldots,\vartheta_n+1;x_1,...,x_n), \end{aligned} \tag{2.11}$$

if $x_1+...+x_n\le 1, x_1\ge 0,\ldots,x_n\ge 0$, where

$$\begin{aligned} &F_A^{(n)}(a,b_1,\ldots,b_n;c_1,\ldots,c_n;x_1,...,x_n) = \\ &= \sum_{m_1,\ldots,m_n=0}^{\infty}\frac{(a,m_1+...+m_n)(b_1,m_1)\cdots(b_n,m_n)}{(c_1,m_1)\cdots(c_n,m_n)}\frac{x_1^{m_1}}{m_1!}\cdots\frac{x_n^{m_n}}{m_n!}, \end{aligned} \tag{2.12}$$

is the Lauricella function of the first kind and the symbol (λ,k) is defined by the relations

$$(\lambda,k) = \begin{cases} \dfrac{\Gamma(\lambda+k)}{\Gamma(\lambda)} = \lambda(\lambda+1)\cdots(\lambda+k-1), & \text{if } k>0 \\ \dfrac{(-1)^k}{(1-\lambda,k)}, & \text{if } k<0 \\ 1, & \text{if } k=0 \end{cases}$$

Our recursive algorithm will based on the following consideration. When we want to calculate the cdf of the Dirichlet distribution, first we should check wether the argument

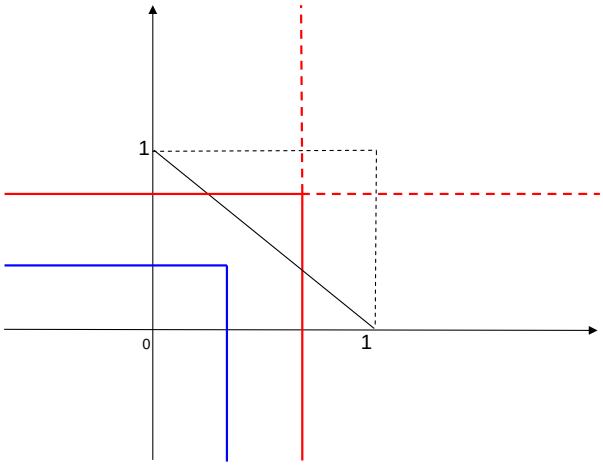

Figure 2: The Lauricella function

vector is in the unit cube or not. If it is outside of the unit cube then the cdf value is trivially zero (when at least one argument value is less then zero), one (when all argument values are greater than one) or it can be determined by calculating a lower dimensional cdf value. If the argument vector is inside the unit cube we should check wether it is in the unit simplex or not (see Figure 2). If it is in the unit simplex we can apply the Lauricella series expansion given by Equation (2.11). Otherwise as the cdf represents the probability $P(X_1 \leq x_1, \ldots, X_n \leq x_n)$ we can take one minus the opposite event probability and then can apply the inclusion–exclusion formula where the last term will trivially be zero.

Let designate $\overline{F}(x_1, \ldots, x_k) = P(X_1 > x_1, \ldots, X_k > x_k)$ for $k = 1, \ldots, n$.

First, we give formulae for the calculation of $F(x_1)$ and $\overline{F}(x_1)$:

$$F(x_1) = \begin{cases} 0, & \text{if } x_1 < 0, \\ \text{by Equation (2.11)}, & \text{if } 0 \leq x_1 < 1, \\ 1, & \text{if } x_1 \geq 1 \end{cases}$$

and trivially

$$\overline{F}(x_1) = 1 - F(x_1).$$

As the one dimensional Dirichlet distribution is beta distribution, the cdf is the so called

incomplete beta function, so we can also apply one of the existing library routines instead of using the Lauricella series expansion given by Equation (2.11).

Now, we are able to calculate $F(x_1, x_2)$ by Equation (2.11) if $x_1 + x_2 < 1$, otherwise if $x_1 + x_2 \geq 1$ we can apply the inclusion–exclusion formula as its last term vanishes:

$$F(x_1, x_2) = \begin{cases} \text{by Equation (2.11)}, & \text{if } x_1 + x_2 < 1, \\ \\ 1 - \overline{F}(x_1) - \overline{F}(x_2), & \text{if } x_1 + x_2 \geq 1. \end{cases}$$

Further, as now we can calculate $F(x_1, x_2)$, we can apply the inclusion–exclusion formula for the calculation of $\overline{F}(x_1, x_2)$, if $x_1 + x_2 < 1$, and it is trivially zero, if $x_1 + x_2 \geq 1$:

$$\overline{F}(x_1, x_2) = \begin{cases} 1 - F(x_1) - F(x_2) + F(x_1, x_2), & \text{if } x_1 + x_2 < 1 \\ \\ 0, & \text{if } x_1 + x_2 \geq 1 \end{cases}$$

Proceeding this way, in the general case, we can calculate $F(x_1, \ldots, x_k)$ and $\overline{F}(x_1, \ldots, x_k)$ for $k = 3, \ldots, n$ in the following way:

$$F(x_1, \ldots, x_k)$$

$$= \begin{cases} \text{by Equation (2.11)}, & \text{if } x_1 + \cdots + x_k < 1 \\ \\ 1 - \sum\limits_{1 \leq i_1 \leq k} \overline{F}(x_{i_1}) + \\ + \sum\limits_{1 \leq i_1 < i_2 \leq k} \overline{F}(x_{i_1}, x_{i_2}) - \\ \vdots \\ +(-1)^{k-1} \sum\limits_{1 \leq i_1 < \cdots < i_{k-1} \leq k} \overline{F}(x_{i_1}, \ldots, x_{i_{k-1}}), & \text{if } x_1 + \cdots + x_k \geq 1 \end{cases}$$

and

$$\overline{F}(x_1, \ldots, x_k) = \begin{cases} 1 - \sum\limits_{1 \leq i_1 \leq k} F(x_{i_1}) + \\ + \sum\limits_{1 \leq i_1 < i_2 \leq k} F(x_{i_1}, x_{i_2}) - \\ \vdots \\ +(-1)^k F(x_1, \ldots, x_k), & \text{if } x_1 + \cdots + x_k < 1 \\ 0, & \text{if } x_1 + \cdots + x_k \geq 1. \end{cases}$$

Theoretically this way we can determine the Dirichlet cdf for any high dimensions. However the application of Equation (2.11), i.e. the Lauricella series expansion will take too much CPU time in higher dimensions. This is why we propose the above described recursive calculation procedure only up to seven dimensions. In the same time we call the attention, that if in a higher dimensional case one can observe that the sum of the first eight smallest argument values is greater then one, the inclusion–exclusion formula can be applied again as in this case $\overline{F}(x_1, \ldots, x_k) = 0$, if $k \geq 8$ and our recursion can be applied when $k \leq 7$. In this sense the recursive algorithm of this Section can be regarded as the generalization of Theorem 2.3.

2.3 Bounding and simulations

The research on the probability of the union of a finite number of events has a long history. In the most cases an explicit calculation of the probability is impossible. Thus approximate methods are required. One of this is bounding the probability by upper and lower bounds. Let $A_1, A_2, \ldots, A_n$ be arbitrary events of a probability space (Ω, P). We begin by considering bounds on the probability of a union $P(A_1 + \ldots + A_n)$. If we have the complete information about these events $A_1, A_2, \ldots, A_n$, the inclusion exclusion method proved the following formula to compute the exact value of $P(A_1 + \ldots + A_n)$:

$$P(A_1 + \ldots + A_n) = \sum_{i=1}^{n} P(A_i) - \sum_{i=1}^{n} \sum_{j=1}^{i-1} P(A_i A_j) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} \sum_{k=1}^{j-1} P(A_i A_j A_k) - \\ - \ldots + (-1)^n P(A_1 \ldots A_n),$$

and we can write the exact value of $P(A_1 \ldots A_n)$:

$$P(A_1 \ldots A_n) = 1 - \sum_{i=1}^{n} P\left(\bar{A}_i\right) + \sum_{i=1}^{n} \sum_{j=1}^{i-1} P\left(\bar{A}_i \bar{A}_j\right) - \sum_{i=1}^{n} \sum_{j=1}^{i-1} \sum_{k=1}^{j-1} P\left(\bar{A}_i \bar{A}_j \bar{A}_k\right) + \\ + - \ldots + (-1)^n P\left(\bar{A}_1 \ldots \bar{A}_n\right).$$

Truncating this expansion after k terms gives a lower or upper bound depending on whether k is even or odd, respectively. A well known bound was presented by Boole ([8]) and Bonferroni ([7]).

The Bonferroni bounds

$$\sum_{k=1}^{2r}(-1)^{k-1}S_k \leq P(\sum_{i=1}^{n}A_i) \leq \sum_{k=1}^{2r-1}(-1)^{k-1}S_k,$$

where

$$S_k = \sum_{1\leq i_1<\ldots<i_k\leq n} P(A_{i_1}\cdots A_{i_k}), k = 1,\ldots,n,$$

are the so called binomial moments of the random variable μ defined as the number of those events $A_1, A_2, \ldots, A_n$ which occur in a random trial.

The Boole–Bonferroni bounds

The sharpest possible bounds in possession of the information involved in the first few $S_k, k = 1,\ldots,m, m << n$ terms are called Boole–Bonferroni bounds. These are known in explicit form only for the following special cases:

Lower and upper bounds when S_1, S_2 are given ($m = 2$)

$$\frac{2}{k^*+1}S_1 - \frac{2}{k^*(k^*+1)}S_2$$

$$\leq P(A_1 + A_2 + \ldots + A_n)$$

$$\leq S_1 - \frac{2}{n}S_2,$$

where

$$k^* = 1 + \left\lfloor \frac{2S_2}{S_1} \right\rfloor.$$

Lower and upper bounds when S_1, S_2, S_3 are given ($m = 3$)

$$\frac{i^*+2n-1}{(i^*+1)n}S_1 - \frac{2(2i^*+n-2)}{i^*(i^*+1)n}S_2 + \frac{6}{i^*(i^*+1)n}S_3$$

$$\leq P(A_1 + A_2 + \ldots + A_n)$$

$$\leq S_1 - \frac{2(2j^*-1)}{j^*(j^*+1)}S_2 + \frac{6}{j^*(j^*+1)}S_3,$$

where

$$i^* = 1 + \left\lfloor \frac{-6S_3 + 2(n-2)S_2}{-2S_2 + (n-1)S_1} \right\rfloor,$$

and

$$j^* = 2 + \left\lfloor \frac{3S_3}{S_2} \right\rfloor.$$

Upper bound when S_1, S_2, S_3, S_4 are given ($m = 4$)

$$P(A_1 + A_2 + \ldots + A_n)$$

$$\leq S_1 - \frac{2((i^*-1)(i^*-2) + (2i^*-1)n)}{i^*(i^*+1)n} S_2 + \frac{6(2i^* + n - 4)}{i^*(i^*+1)n} S_3 - \frac{24}{i^*(i^*+1)n} S_4,$$

where

$$i^* = 1 + \left\lfloor \frac{-12S_4 + 3(n-4)S_3 + (n-2)S_2}{-3S_3 + (n-2)S_2} \right\rfloor.$$

The Kounias bounds

In 1968, Kounias established the following linear lower bound

$$P\left(\sum_{i=1}^{n} A_i\right) \geq \max_{\jmath} \left\{ \sum_{i \in \jmath} P(A_i) - \sum_{\substack{i,j \in \jmath \\ i<j}} P(A_i A_j) \right\},$$

where the maximum is taken over all subsets $\jmath$ of set $S = \{1, 2, \ldots, n\}$. Since there are in total $2^n - 1$ non-empty subsets of S Kounias ([69]) also provided a Bonferroni type upper bound as follows

$$P\left(\sum_{i=1}^{n} A_i\right) \leq \sum_{i=1}^{n} P(A_i) - \max_{j} \sum_{i=1, i \neq j} P(A_i A_j),$$

where the maximum is taken over $\jmath \in \{1, 2, \ldots, n\}$.

The Kounias upper bound is a special case of the upper bound due to Hunter ([58]) and Worsley ([120]).

Hunter- Worsley upper bound

Hunter ([58]) and Worsley ([120]) gave an upper bound for $P(A_1 + A_2 + \ldots + A_n)$ by the use of S_1 and the individual probabilities $P(A_i A_j), 1 \leq i < j \leq n$ involved in S_2.

Construct a non-oriented complete graph with n vertices and assign to vertex i the event A_i (or the probability $P(A_i)$) and to edge (i,j) the weight $P(A_iA_j)$. A connected, acyclic subgraph of a graph is called tree. A spanning tree of a graph is a tree in the graph such that it connects all vertices. Find T, a maximal weight spanning tree, then

$$P(A_1 + A_2 + \ldots + A_n) \leq \sum_{j=1}^{n} P(A_j) - \sum_{(i,j)\in T} P(A_iA_j).$$

The maximal spanning tree in a graph can be found quickly by greedy algorithms (by Kruskal, Prim etc.). Denote the weight of spanning tree T by $w(T)$ then the Hunter–Worsley bound can be written simply as

$$P(A_1 + A_2 + \ldots + A_n) \leq S_1 - w(T).$$

It is easy to see that the Hunter–Worsley bound is always sharper than the best Boole–Bonferroni upper bound using S_1 and S_2. Designating by $T^\star$ the maximal weight spanning tree, by $w(T^\star)$ its weight and by $T_1, T_2, \ldots, T_N$ the existing spanning trees in the graph, we have the relations

$$Nw(T^\star) \geq \sum_{j=1}^{N} w(T_j) = N(n-1)\frac{\sum\limits_{1\leq i<j\leq n} P(A_iA_j)}{\binom{n}{2}} = N\frac{2}{n}S_2.$$

Dividing both sides of the above inequality by N one can conclude

$$S_1 - w(T^\star) \leq S_1 - \frac{2}{n}S_2.$$

Tomescu bounds

Tomescu ([113]) generalized Hunter's and Worsley's bound by introducing the concept of hypertrees. Hypertrees are defined in hypergraphs like trees were defined in simple graphs.

An *h–uniform hypergraph* H is a pair $H = (V, \mathcal{E}_h)$, where V is a set of vertices and $\mathcal{E}_h$ is a family of h–sets of V, i.e. $\mathcal{E}_h = (E_1, \ldots, E_m)$ so that $|E_i| = h$ for all $i = 1, \ldots, m$ and $h \geq 2$. The sets $E_i, i = 1, \ldots, m$ are called the edges of H. The order of a hypergraph H is equal to $|V|$. The degree of a vertex $v \in V$, denoted by $d_H(v)$, is the number of edges of H containing it.

By definition an *h–hypertree* or an *h–tree* is an h–uniform hypergraph $T = (V, \mathcal{E}_h)$ such that for $h = 2$, T is a tree with vertex set V and for $h \geq 3$, T is defined recursively by the following two rules:

(i) If $V = \{v_1, \ldots, v_h\}$ then T has a unique edge $E_1 = \{v_1, \ldots, v_h\}$.

(ii) If $|V| \geq h + 1$ then there exists a vertex $v_i \in V$ such that if $E_1, \ldots, E_q$ denote all edges containing v_i then $E_1 \setminus \{v_i\}, \ldots, E_q \setminus \{v_i\}$ induce an $(h-1)$–hypertree with vertex set $V \setminus \{v_i\}$.

We remark that every h–hypertree of order n has $\binom{n-1}{h-1}$ edges and every h–hypertree T of order n has the property that $d_T(v) \geq \binom{n-2}{h-2}$ for any $v \in V$ and $d_T(v) = \binom{n-2}{h-2}$ for any $v \in V$ if and only if v is a terminal vertex of T.

Some examples for the h–hypertrees are the following.

Let us regard the set of vertices $\{1, 2, 3, 4, 5\}$ and the 3–sets:

$$\begin{aligned} E_1 &= \{1, 2, 3\}, \\ E_2 &= \{1, 2, 4\}, \\ E_3 &= \{2, 3, 4\}, \\ E_4 &= \{1, 2, 5\}, \\ E_5 &= \{2, 3, 5\}, \\ E_6 &= \{3, 4, 5\} \end{aligned}$$

are edges. The following are 3–hypertrees:

a) $H_1 = (\{1, 2, 3\}, \{E_1\})$

b) $H_2 = (\{1, 2, 3, 4\}, \{E_1, E_2, E_3\})$ since $E_2 \setminus \{4\} = \{1, 2\}$ and $E_3 \setminus \{4\} = \{2, 3\}$ are the edges of a tree on vertices 1, 2 and 3.

c) $H_3 = (\{1, 2, 3, 4, 5\}, \{E_1, E_2, E_3, E_4, E_5, E_6\})$ since $\{1, 2\}, \{2, 3\}, \{3, 4\}$ are the edges of a tree with vertex set $\{1, 2, 3, 4\}$.

The h–hypertrees like trees in Hunter–Worsley bound can improve the Bonferroni bounds. The following theorem proved by Tomescu shows this fact.

2.4. Theorem: *Let $T_h^n = \{V, \mathcal{E}_h\}$ denote any h–hypertree with vertex set $V = \{1, \ldots, n\}$. The following inequalities hold:*

$$P(\sum_{i=1}^{n} A_i) \leq \sum_{k=1}^{2p-1} (-1)^{k-1} S_k - \sum_{(i_1,\ldots,i_{2p})\epsilon\mathcal{E}_{2p}} P(A_{i_1} \cdots A_{i_{2p}}),$$

for every $1 \leq p \leq n/2$, where the second sum is over all edges of T_{2p}^n and hence contains $\binom{n-1}{2p-1}$ terms;

$$P(\sum_{i=1}^{n} A_i) \geq \sum_{k=1}^{2p} (-1)^{k-1} S_k + \sum_{(i_1,\ldots,i_{2p+1})\epsilon\mathcal{E}_{2p+1}} P(A_{i_1} \cdots A_{i_{2p+1}}),$$

for every $1 \leq p \leq (n-1)/2$, where the second sum is over all edges T_{2p+1}^n and hence contains $\binom{n-1}{2p}$ terms.

Bukszár bounds

Bukszár ([14]) generalized Tomescu bounds by introducing a new hypergraph structure called (h, m)–hypermultitree. This has the following definition.

(h, m)-hypermultitrees

We need the following two definitions, to introduce the concept of (h, m)-hypermultitrees.

2.4. Definition *Let m be a positive integer. An m-multicherry is a hypergraph of the form $(V, \mathcal{E}_2, \ldots, \mathcal{E}_{m+1})$, where $V = \{v_1, \ldots, v_{m+1}\}$ is the set of vertices and the family of hyperedges $\mathcal{E}_i$ is the set of all subsets of $\{v_1, \ldots, v_{m+1}\}$ containing i vertices with v_{m+1} included, i.e. $\mathcal{E}_i = \{H \mid v_{m+1} \in H \subset \{v_1, \ldots, v_{m+1}\}, \mid H \mid = i\}$. The vertex v_{m+1} is called the dominating vertex of the m-multicherry.*

The m-multicherry with dominating vertex v_{m+1} and with non-dominating vertices $v_1, \ldots, v_m$ is denoted by $(\{v_1, \ldots, v_m\}, v_{m+1})$.

The 3-multicherry $(\{1, 2, 3\}, 4)$ is illustrated in Figure 3 by its vertices and edges. Its hyperedges containing more than two vertices $\{1, 2, 4\}, \{1, 3, 4\}, \{2, 3, 4\}$ and $\{1, 2, 3, 4\}$ are not marked in the figure because of the perspicuity.

Note that a 1-multicherry is a single edge together with its incident vertices.

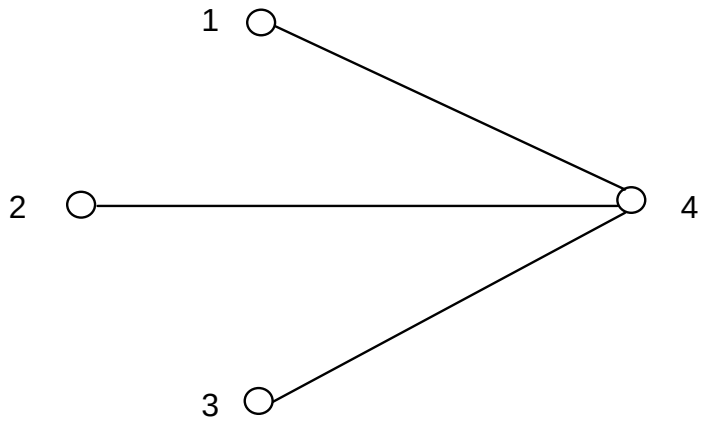

Figure 3: The 3-multicherry $(\{1,2,3\},4)$.

2.5. Definition *Let m be a positive integer. An m-multitree is a hypergraph of the form $(V,\mathcal{E}_2,\ldots,\mathcal{E}_{m+1})$, where V is the set of vertices and $\mathcal{E}_i$'s are sets of hyperedges containing i vertices. An m-multitree is recursively defined by the following two rules.*

i) The smallest m–multitree $\Delta=(V,\mathcal{E}_2,\ldots,\mathcal{E}_{m+1})$ has m vertices and $\mathcal{E}_i$ is the family of all subsets of V containing i vertices, where $i=2,\ldots,m$, and $\mathcal{E}_{m+1}=\emptyset$.

ii) From an m–multitree $\Delta=(V,\mathcal{E}_2,\ldots,\mathcal{E}_{m+1})$ we can obtain a new m–multitree $\Delta'=(V',\mathcal{E}'_2,\ldots,\mathcal{E}'_{m+1})$ by adjoining an m–multicherry $(\{v_1,\ldots,v_m\},v_{m+1})$, where $v_1,\ldots,v_m\in V$ and v_{m+1} is a new vertex (i.e. $v_{m+1}\notin V$). More precisely $V'=V\cup\{v_{m+1}\},\mathcal{E}'_i=\mathcal{E}_i\cup\{H\mid v_{m+1}\in H\subset\{v_1,\ldots,v_{m+1}\},\mid H\mid=i\}$.

A 3-multitree $\Delta=(V,\mathcal{E}_2,\mathcal{E}_3,\mathcal{E}_4)$ is illustrated in Figure 4. Given $1,2,3$ and the hyperedges $\{1,2\},\{1,3\},\{2,3\}$, $\{1,2,3\}$, we subsequently adjoin the 3-multicherries $(\{1,2,3\},4),(\{1,4,4\},5)$ and $(\{2,3,5\},6)$ as shown in the figure. The edges of a 3-multicherry are drawn with the same line-style. The vertices and hyperedges of Δ are $V=\{1,2,3,4,5,6\},\mathcal{E}_2=\{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,4\},\{2,6\},\{3,4\},\{3,5\},\{3,6\},\{4,5\},\{4,6\}\},\mathcal{E}_3=\{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{2,3,4\},\{1,3,5\},\{1,4,5\},\{3,4,5\},\{2,3,6\},\{2,5,6\},\{3,5,6\}\}$, $\mathcal{E}_4=\{\{1,2,3,4\},\{1,3,4,5\},\{2,3,5,6\}\}$.

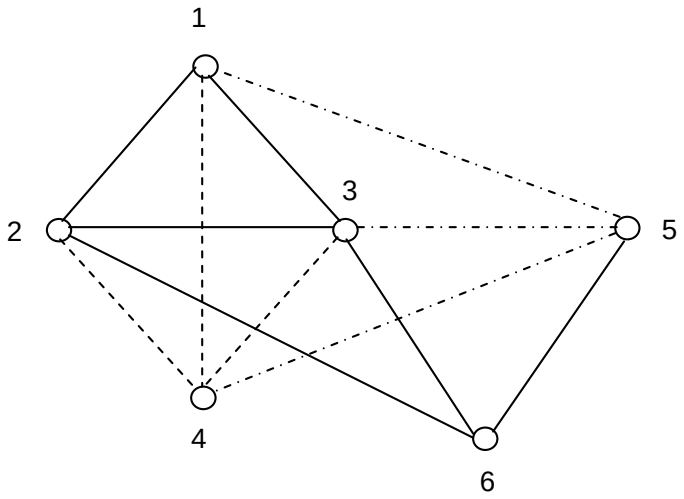

Figure 4: A 3-multitree.

Note that a 1-multitree is a usual tree. It can be proved that a 1-multitree is determined by its vertices and edges.

2.6. Definition *Let $h \geq 0$ and $m \geq 1$ be arbitrary integers. An (h, m)–hypermultitree is a hypergraph of the form $(V, {}_h\mathcal{E}_2, \ldots, {}_h\mathcal{E}_{m+1})$, where V is the set of vertices and ${}_h\mathcal{E}_i$'s are sets of hyperedges containing $h + i$ vertices. An (h, m)–hypermultitree is defined recursively by the following rules:*

(i) The $(0, m)$–hypermultitrees are the same as the m–multitrees.

(ii) The smallest (h, m)–hypermultitree $\Delta = (V, {}_h\mathcal{E}_2, \ldots, {}_h\mathcal{E}_{m+1})$ has $h + m$ vertices and ${}_h\mathcal{E}_i$ consists of all subsets of the vertices of V containing $h + i$ vertices, where $i = 2, \ldots, m$, and ${}_h\mathcal{E}_{m+1} = \emptyset$.

(iii) From an (h, m)–hypermultitree $\Delta = (V, {}_h\mathcal{E}_2, \ldots, {}_h\mathcal{E}_{m+1})$ we can obtain a new (h, m)–hypermultitree in the following manner. Let $\Gamma = (V, {}_{h-1}\mathcal{E}_2^*, \ldots, {}_{h-1}\mathcal{E}_{m+1}^*)$ be an arbitrary $(h-1, m)$–hypermultitree with the same set of vertices as in Δ. By adjoining a new vertex v to Δ and hyperedges of Γ extended by v, we obtain the new (h, m)–hypermultitree $\Delta' = (V', {}_h\mathcal{E}_2', \ldots, {}_h\mathcal{E}_{m+1}')$ with $V' = V \cup \{v\}$ and ${}_h\mathcal{E}_i' = {}_h\mathcal{E}_i \cup \bigcup_{E \in {}_{h-1}\mathcal{E}_i} \{E \cup \{v\}\}$.

The smallest (h, m)–hypermultitrees are generalizations of Tomescu's hypertrees, which are the $(h, 1)$–hypermultitrees in our definition.

2.1. Example $\Delta = (V,_1\mathcal{E}_2,_1\mathcal{E}_3)$, *with* $_1\mathcal{E}_2 = \{\{1,2,3\}\}$ *and* $_1\mathcal{E}_3 = \emptyset$ *is an* $(1,2)$*–hypermultitree by (ii). From* Δ *we can obtain the* $(1,2)$*–hypermultitree* $\Delta' = (V',_1\mathcal{E}'_2,_1\mathcal{E}'_3)$ *on the basis of* $(0,2)$*–hypermultitree* $\Gamma_1 = (\{1,2,3\},\{\{1,2\},\{1,3\},\{2,3\}\},\{\{1,2,3\}\})$ *shown in Figure 5 by adjoining vertex 4 by (iii), where* $V = \{1,2,3,4\}$, $_1\mathcal{E}'_2 = \{\{1,2,3\},\{1,2,4\},\{1,3,4\},\{2,3,4\}\}$, $_1\mathcal{E}'_3 = \{\{1,2,3,4\}\}$. *From* Δ' *we obtain* $\Delta'' = (V'',_1\mathcal{E}''_2,_1\mathcal{E}''_3)$*on the basis of* Γ_2, *where* $V\!I = \{1,2,3,4,5\}$, $_1\mathcal{E}''_2 =_1 \mathcal{E}'_2 \cup \{\{1,3,5\},\{1,4,5\},\{2,3,5\},\{2,4,5\},\{3,4,5\}\}$, $_1\mathcal{E}''_3 =_1 \mathcal{E}'_3 \cup \{\{1,3,4,5\},\{2,3,4,5\}\}$. *Finally, from* Δ'' *we obtain* $\Delta''' = (V''',_1\mathcal{E}'''_2,_1\mathcal{E}'''_3)$ *on the basis of* Γ_3, *where* $V\!I\!I = \{1,2,3,4,5,6\}$, $_1\mathcal{E}'''_2 =_1 \mathcal{E}''_2 \cup \{\{1,3,6\},\{1,4,6\},\{2,3,6\},\{2,4,6\},\{3,5,6\},\{4,5,6\}\}$, $_1\mathcal{E}'''_3 =_1 \mathcal{E}''_3 \cup \{\{1,3,5,6\},\{2,3,4,6\}\}$.

2.7. Definition *Let* $A_1,\ldots,A_n$ *be arbitrary events. Then the weight of* (h,m)*–hypermultitree* $\Delta = (V,_h\mathcal{E}_2,\ldots,_h\mathcal{E}_{m+1})$ *is:*

$$w(\Delta) = \sum_{(i_1,\ldots,i_{h+2})\in{}_h\mathcal{E}_2} P(A_{i_1}\cdots A_{i_{h+2}})$$

$$- \sum_{(i_1,\ldots,i_{h+3})\in{}_h\mathcal{E}_3} P(A_{i_1}\cdots A_{i_{h+3}})$$

$$+ \ldots + (-1)^{m+1} \sum_{(i_1,\ldots,i_{h+m+1})\in{}_h\mathcal{E}_{m+1}} P(A_{i_1}\cdots A_{i_{h+m+1}}).$$

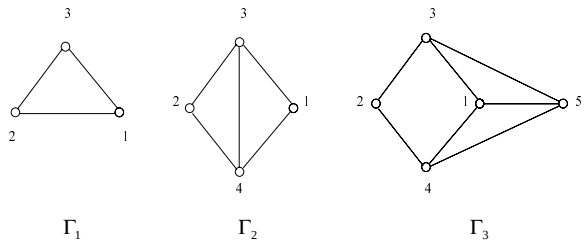

Figure 5: The Γ_1, Γ_2, Γ_3, hypermultitrees of Example 2.1.

2.5. Theorem: *Let $A_1, \dots, A_n$ be arbitrary events and let $\Delta = (V, {}_h\mathcal{E}_2, \dots, {}_h\mathcal{E}_{m+1})$ be an arbitrary (h, m)–hypermultitree, where $V = \{1, \dots, n\}$. The following inequalities hold:*

If h is even:

$$P\left(\sum_{i=1}^{h+1} A_i\right) \leq \sum_{k=1}^{h+1} (-1)^{k-1} S_k - w(\Delta), \tag{2.13}$$

If h odd:

$$P\left(\sum_{i=1}^{h+1} A_i\right) \geq \sum_{k=1}^{h+1} (-1)^{k-1} S_k + w(\Delta), \tag{2.14}$$

where $S_k = \sum_{1 \leq i_1 < \dots < i_k \leq n} P(A_{i_1} \cdots A_{i_k})$. For the proof see Bukszár ([15]). The theorem has been proved for the special case of $m = 1$ by Tomescu ([113]) and in the more special case of $m = 1$ and $h = 0$ by Hunter ([58]) and Worsley ([120]). In the case of $m = 2$ and $h = 0$ and some other results are detailed in Bukszár and Prékopa ([12]). For higher levels their calculation may become very time consuming, the best choice is to use the possible smallest value, i.e. $h = 0$ for upper and $h = 1$ for lower bounds. The (h, m)–hypermultitree bounds are as good as heavy hypermultitrees are applied. The construction of heavy weighted $(0, m)$- and $(1, m)$–hypermultitrees algorithms given by Bukszár's ([14]).

Bukszár algorithms:

Bukszár ([14]) found a heavy $(0, m)$–hypermultitree as follows. Create the input data by evaluating the probabilities $P(A_iA_j)(1 \leq i < j \leq n)$ first, then introduce a weight function on the edges of the complete graph Λ with vertex set $\{1, \dots, n\}$: the edge incident to the vertices i and j has the weight $P(A_iA_j)$. Find the maximum weight (1-multi)tree on Λ by Prim's algorithm and obtain the Hunter-Worsley bound with the maximum weight tree (see Theorem 2.5.). Then, extended the maximum weight tree to a 2-multitree by the following recursion.

i) Take the maximum weight edge together with the vertices incident to it. Then adjoin 2-multicherries subsequently, where an edge of the 2-multicherry is in the tree and its other one is the heaviest possible.

ii) Extend the r-multitree to an $r + 1$ vertices of which are those selected first in the recursion producing the r-multitree. (The $r + 1$-multitree with $r + 1$ vertices has

a unique structure.) Then, we subsequently adjoin an $(r+1)$-multicherry which is an extension of an r-multicherry of the r-multitree with the heaviest possible edge.

To execute the algorithm, we need to evaluate all $P(A_iA_j)$ probabilities and to compute the weight of the m-multitree obtained by the algorithm we need to evaluate only $n-m$ probabilities of the intersection of $m+1$ events.

Bukszár ([14]) found a heavy $(1,m)$–hypermultitree as follows.

Create the input data by evaluating the probabilities $P(A_iA_jA_k)(1 \leq i < j < k \leq n)$ and construct a $(1,1)$–hypermultitree by the following greedy algorithm. First take a hyperedge h,l,m for which $P(A_hA_lA_m) = \max\{P(A_iA_jA_k) \mid (1 \leq i < j < k \leq n)\}$ and construct the $(1,1)$–hypermultitree $\{\{h,l,m\},\{\{h,l,m\}\},0\}$.

i) Create a sequence of $P(A_iA_jA_k)$ a member of which is obtained by adjoining a vertex and some hyperedges to the previous member of the sequence. The last member of the sequence will be required heavy $(1,1)$–hypermultitree.

   More precisely, assume that we have already constructed a $(1,1)$–hypermultitree the vertex set of which is denoted by V.

ii) Assign a tree to all vertices in $\{1,\ldots,n\} \setminus V$ in the following way. Let k be a vertex in $\{1,\ldots,n\} \setminus V$. We introduce a weight function on the edges of the complete graph with vertex set V: the edge incident to the vertices i and j has the weight $P(A_iA_jA_k)$. Then we find the maximum weight tree on the complete graph by Prim's algorithm and assign this tree to vertex k. We adjoin the hyperedges obtained by extending vertex. To obtain $(1,m)$–hypermultitree from $(1,1)$–hypermultitree constructed above we extended the (1-multi)trees which were assigned to the adjoined vertices to m-multitrees as described previously.

   To execute the algorithm, we need to evaluate all $P(A_iA_jA_k)$ probabilities and to compute the weight of the $(1,m)$-hypermultitree obtained by the algorithm we need to evaluate only $\binom{n-m}{2}$ probabilities of the intersection of $m+2$ events.

The general scheme of (h,m)–hypermultitree bounds is the following:

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \leq \sum_{k=1}^{h+1}(-1)^{k-1}S_k \\ - \sum_{(i_1,\ldots,i_{h+2})\in {}_h\overline{\mathcal{E}}_2} P(A_{i_1}\cdots A_{i_{h+2}}) \\ + \sum_{(i_1,\ldots,i_{h+3})\in {}_h\overline{\mathcal{E}}_3} P(A_{i_1}\cdots A_{i_{h+2}}A_{i_{h+3}}) \\ \vdots \\ + (-1)^m \sum_{(i_1,\ldots,i_{h+m+1})\in {}_h\overline{\mathcal{E}}_{m+1}} P(A_{i_1}\cdots A_{i_{h+2}}\cdots A_{i_{h+m+1}}), \end{aligned} \tag{2.15}$$

where h is an arbitrary even number, $\Delta = (V, {}_h\overline{\mathcal{E}}_2, \ldots, {}_h\overline{\mathcal{E}}_{m+1})$ is an (h,m)–hypermultitree, in which V is the set of vertices and ${}_h\overline{\mathcal{E}}_i$'s are sets of hyperedges containing $h+i$ vertices.

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \geq \sum_{k=1}^{h+1}(-1)^{k-1}S_k \\ + \sum_{(i_1,\ldots,i_{h+2})\in {}_h\underline{\mathcal{E}}_2} P(A_{i_1}\cdots A_{i_{h+2}}) \\ - \sum_{(i_1,\ldots,i_{h+3})\in {}_h\underline{\mathcal{E}}_3} P(A_{i_1}\cdots A_{i_{h+2}}A_{i_{h+3}}) \\ \vdots \\ + (-1)^{m+1} \sum_{(i_1,\ldots,i_{h+m+1})\in {}_h\underline{\mathcal{E}}_{m+1}} P(A_{i_1}\cdots A_{i_{h+2}}\cdots A_{i_{h+m+1}}), \end{aligned} \tag{2.16}$$

where h is an arbitrary odd number, $\Delta = (V, {}_h\underline{\mathcal{E}}_2, \ldots, {}_h\underline{\mathcal{E}}_{m+1})$ is an (h,m)–hypermultitree, in which V is the set of vertices and ${}_h\underline{\mathcal{E}}_i$'s are sets of hyperedges containing $h+i$ vertices.

In the special case $h = 0$ the upper bound is the following:

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \leq S_1 & \\ & - \sum_{(i_1,i_2)\in {}_0\overline{\mathcal{E}}_2} P(A_{i_1}A_{i_2}) \\ & + \sum_{(i_1,i_2,i_3)\in {}_0\overline{\mathcal{E}}_3} P(A_{i_1}A_{i_2}A_{i_3}) \\ & \vdots \\ & + (-1)^m \sum_{(i_1,\ldots,i_{m+1})\in {}_0\overline{\mathcal{E}}_{m+1}} P(A_{i_1}A_{i_2}\cdots A_{i_{m+1}}) = {}_0U_m. \end{aligned} \tag{2.17}$$

In the special case $h = 1$ the lower bound is the following:

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \geq S_1 - S_2 & \\ & + \sum_{(i_1,i_2,i_3)\in {}_1\underline{\mathcal{E}}_2} P(A_{i_1}A_{i_2}A_{i_3}) \\ & - \sum_{(i_1,\ldots,i_4)\in {}_1\underline{\mathcal{E}}_3} P(A_{i_1}A_{i_2}A_{i_3}A_{i_4}) \\ & \vdots \\ & + (-1)^m \sum_{(i_1,\ldots,i_{m+2})\in {}_1\underline{\mathcal{E}}_{m+1}} P(A_{i_1}A_{i_2}A_{i_3}\cdots A_{i_{m+2}}) = {}_1L_m. \end{aligned} \tag{2.18}$$

In Bukszár's ([14]) paper there was given a practical reformulation of the above lower and upper bounds, by allowing some events to be opposite in the products of the events $A_i, i = 1, \ldots, n$.

The reformulated upper bound is:

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \quad \leq \quad & 1 - P\left(\overline{A}_{i_1}\overline{A}_{i_2}\cdots\overline{A}_{i_{m+1}}\right) \\ & + \sum_{\{\{j_1,\ldots,j_m\},j_{m+1}\}\in\mathcal{M}} P\left(\overline{A}_{j_1}\cdots\overline{A}_{j_m}A_{j_{m+1}}\right), \end{aligned} \tag{2.19}$$

where $\{\{i_1, \ldots, i_m\}, i_{m+1}\}$ is the first m–multicherry of the $(0, m)$–hypermultitree, while $\mathcal{M}$ is the set of all m–multicherries of the $(0, m)$–hypermultitree, except the first one. As the m–multitree consists of $n - m$ number of m–multicherries, the calculation of

the upper bound by the new formula (2.19) need only $n-m$ number of product event probability calculations where all products consist of $m+1$ events.

The reformulated lower bound is:

$$\begin{aligned} P(\sum_{i=1}^{n} A_i) \quad \geq \quad & 1 - P(\overline{A}_{i_1} \cdots \overline{A}_{i_{m+2}}) \\ & + \sum_{k=2}^{n-m-1} \Bigg[P(\overline{A}_{i_1^{(k)}} \cdots \overline{A}_{i_{m+1}^{(k)}} A_{j^{(k)}}) \\ & - \sum_{\left\{\left\{j_1^{(k)},\ldots,j_m^{(k)}\right\},j_{m+1}^{(k)}\right\} \in \mathcal{M}^{(k)}} P(\overline{A}_{j_1^{(k)}} \cdots \overline{A}_{j_m^{(k)}} A_{j_{m+1}^{(k)}} A_{j^{(k)}}) \Bigg], \end{aligned} \tag{2.20}$$

where $\{i_1, \ldots, i_{m+2}\}$ is the first m–multicherry in the recursive construction of the $(1,m)$–hypermultitree, $j^{(k)}$ is the index of the vertex to be entered in the k-th step; $\left\{\left\{i_1^{(k)}, \ldots, i_m^{(k)}\right\}, i_{m+1}^{(k)}\right\}$ is the first m–multicherry of the $(0,m)$–hypermultitree constructed from the verteces existing in the k-th step; while $\mathcal{M}^{(k)}$ is the set of all m–multicherries of the $(0,m)$–hypermultitree constructed from the verteces existing in the k-th step, except the first one. It is easy to check that this way the calculation of the lower bound by the new formula (2.20) need only $(n-m)(n-m-1)/2$ number of product event probability calculations where all products consist of $m+2$ events.

In the case of Dirichlet cdf calculations the probability terms in the formulae (2.19) and (2.20) can be calculated by our recursive calculation algorithm:

$$\begin{aligned} & P(X_1 > x_1, \ldots, X_k > x_k, X_{k+1} \leq x_{k+1}) \\ & = P(X_1 > x_1, \ldots, X_k > x_k) \\ & - P(X_1 > x_1, \ldots, X_k > x_k, X_{k+1} \leq x_{k+1}) \\ & = \overline{F}(x_1, \ldots, x_k) \\ & - \overline{F}(x_1, \ldots, x_k, x_{k+1}) \end{aligned}$$

and

$$
\begin{aligned}
&P(X_1 > x_1, \ldots, X_k > x_k, X_{k+1} \leq x_{k+1}, X_{k+2} \leq x_{k+2}) \\
&= P(X_1 > x_1, \ldots, X_k > x_k) \\
&-P(X_1 > x_1, \ldots, X_k > x_k, X_{k+1} > x_{k+1}) \\
&-P(X_1 > x_1, \ldots, X_k > x_k, X_{k+2} > x_{k+2}) \\
&+P(X_1 > x_1, \ldots, X_k > x_k, X_{k+1} > x_{k+1}, X_{k+2} > x_{k+2}) \\
&= \overline{F}(x_1, \ldots, x_k) - \overline{F}(x_1, \ldots, x_k, x_{k+1}) - \overline{F}(x_1, \ldots, x_k, x_{k+2}) \\
&+\overline{F}(x_1, \ldots, x_k, x_{k+1}, x_{k+2}).
\end{aligned}
$$

The variance reduction simulation procedure for the estimation of the probability $P(A_1 + \cdots + A_n)$ can be designed in the following way. Let us make large number of random trials in the probability space where $A_1, \ldots, A_n$ are arbitrary events. Let μ designate the number of those $A_1, \ldots, A_n$ events which occur; ${}_0\tau_2, \ldots, {}_0\tau_{m+1}$ designate the number of occurring product events according to the elements of the ${}_0\overline{\mathcal{E}}_2, \ldots, {}_0\overline{\mathcal{E}}_{m+1}$ hyperedge sets in the ${}_0U_m$–hypermultitree upper bound (see Equation (2.9)); and finally ${}_1\tau_2, \ldots, {}_1\tau_{m+1}$ designate the number of occurring product events according to the elements of the ${}_1\underline{\mathcal{E}}_2, \ldots, {}_1\underline{\mathcal{E}}_{m+1}$ hyperedge sets in the ${}_1L_m$–hypermultitree lower bound (see Equation (2.10)). Define the following random variables:

$$
\nu_{CR} = \begin{cases} 0, & \text{if } \mu = 0 \\ 1, & \text{if } \mu \geq 1 \end{cases}
$$

$$
{}_0\nu_m = \begin{cases} 0, & \text{if } \mu \leq 1 \\ 1 - \mu + {}_0\tau_2 - \ldots + (-1)^{m+1}{}_0\tau_{m+1}, & \text{if } \mu \geq 2 \end{cases}
$$

$$
{}_1\nu_m = \begin{cases} 0, & \text{if } \mu \leq 2 \\ 1/2(\mu-1)(\mu-2) - {}_1\tau_2 + \ldots + (-1)^m{}_1\tau_{m+1}, & \text{if } \mu \geq 3. \end{cases}
$$

Then it is easy to check that the expected value of the random variables ν_{CR}, ${}_0\nu_m + {}_0U_m$, ${}_1\nu_m + {}_1L_m$ equals the probability value $P(A_1 + \cdots + A_n)$. Now, if we estimate in a simulation procedure the expected values of these random variables and the elements of their variance–covariance matrix, then by the well known regression technique we can combine these estimators into a final estimator with minimal variance. More details of the similar simulation procedures can be found in the paper by Szántai ([110]).

2.4 First and second order partial derivatives

When we know the conditional probability distribution of a random vector according to a condition on any one of its components, then the partial derivative of its cdf can be determined by the following general formula (see Prékopa ([87])):

$$\frac{\partial F(x_1,\ldots,x_n)}{\partial x_k} = F(x_1,\ldots,x_{k-1},x_{k+1},\ldots,x_n|x_k)f(x_k). \tag{2.21}$$

In the case of the Dirichlet distribution the conditional random vector $(X_1,\ldots,X_{k-1},X_{k+1},\ldots,X_n|X_k=x_k)$ has the property that the transformed conditional random vector $(\frac{X_1}{1-x_k},\ldots,\frac{X_{k-1}}{1-x_k},\frac{X_{k+1}}{1-x_k},\ldots,\frac{X_n}{1-x_k}|X_k=x_k)$ has the $n-1$–variate Dirichlet distribution $D(\vartheta_1,\ldots,\vartheta_{k-1},\vartheta_{k+1},\ldots,\vartheta_n;\vartheta_{n+1})$.

So the general formula (2.21) leads to the first order partial derivatives:

$$\begin{aligned}&\frac{\partial F(x_1,\ldots,x_{k-1},x_k,x_{k+1},\ldots x_n;\Theta;\vartheta_{n+1})}{\partial x_k}\\ &= F(\frac{x_1}{1-x_k},\ldots,\frac{x_{k-1}}{1-x_k},\frac{x_{k+1}}{1-x_k},\ldots,\frac{x_n}{1-x_k};\Theta\setminus\{\vartheta_k\};\vartheta_{n+1})\\ &\times f(x_k;\vartheta_k;\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}), k=1,\ldots,n,\end{aligned} \tag{2.22}$$

where Θ designates the parameter set $\{\vartheta_1,\ldots,\vartheta_n\}$.

Equation (2.22) shows that the first order partial derivatives can be calculated as a product of an $n-1$–variate Dirichlet cdf and a 1–variate Dirichlet, i.e. beta pdf. So when we are taking the second order derivatives we can apply again our general formula (2.21).

For the mixed second order partial derivatives we get

$$\begin{aligned}&\frac{\partial^2 F\Big(x_1,\ldots,x_{k-1},x_k,x_{k+1},\ldots,x_{l-1},x_l,x_{l+1},\ldots x_n;\Theta;\vartheta_{n+1}\Big)}{\partial x_k\partial x_l}\\ &= \frac{1}{1-x_k}F\Big(\frac{x_1}{1-x_k-x_l},\ldots,\frac{x_{k-1}}{1-x_k-x_l},\frac{x_{k+1}}{1-x_k-x_l},\ldots,\\ &\frac{x_{l-1}}{1-x_k-x_l},\frac{x_{l+1}}{1-x_k-x_l},\ldots,\frac{x_n}{1-x_k-x_l};\Theta\setminus\{\vartheta_k,\vartheta_l\};\vartheta_{n+1}\Big)\\ &\cdot f\Big(\frac{x_l}{1-x_k};\vartheta_l;\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}\Big)f\Big(x_k;\vartheta_k;\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k,\vartheta_l\}}\vartheta_i+\vartheta_{n+1}\Big),\\ &k=1,\ldots,n-1, l=k+1,\ldots,n.\end{aligned} \tag{2.23}$$

For the pure second order partial derivatives we get

$$
\begin{aligned}
&\frac{\partial^2 F\Big(x_1,\dots,x_{k-1},x_k,x_{k+1},\dots x_n;\Theta;\vartheta_{n+1}\Big)}{\partial^2 x_k}\\
&=\Bigg[\sum_{i=1,i\neq k}^{n}\frac{x_i}{(1-x_k)^2}F\Big(\frac{x_1}{1-x_k-x_i},\dots,\frac{x_{k-1}}{1-x_k-x_i},\\
&\frac{x_{k+1}}{1-x_k-x_i},\dots,\frac{x_{i-1}}{1-x_k-x_i},\frac{x_{i+1}}{1-x_k-x_i},\dots,\frac{x_n}{1-x_k-x_i};\\
&\Theta\setminus\{\vartheta_k,\vartheta_i\};\vartheta_{n+1}\Big)f\Big(\frac{x_i}{1-x_k};\vartheta_i;\sum_{\vartheta_j\in\Theta\setminus\{\vartheta_k,\vartheta_i\}}\vartheta_j+\vartheta_{n+1}\Big)\Bigg]\\
&\cdot f\Big(x_k;\vartheta_k;\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}\Big)\\
&+F\Big(\frac{x_1}{1-x_k},\dots,\frac{x_{k-1}}{1-x_k},\frac{x_{k+1}}{1-x_k},\dots,\frac{x_n}{1-x_k};\Theta\setminus\{\vartheta_k\};\vartheta_{n+1}\Big)\\
&\cdot\frac{\Gamma\Big(\sum_{\vartheta_i\in\Theta}\vartheta_i+\vartheta_{n+1}\Big)}{\Gamma(\vartheta_k)\Gamma\Big(\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}\Big)}\Bigg[(\vartheta_k-1)x_k^{\vartheta_k-2}\Big(1-x_k\Big)^{\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}-1}\\
&-x_k^{\vartheta_k-1}\Big(\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}-1\Big)\Big(1-x_k\Big)^{\sum_{\vartheta_i\in\Theta\setminus\{\vartheta_k\}}\vartheta_i+\vartheta_{n+1}-2}\Bigg],
\end{aligned}
\tag{2.24}
$$

$k=1,\dots,n.$

Numerical results

2.2. Example *Let us regard a 20–dimensional Dirichlet distribution. The parameter values* $\vartheta_i, i=1,\dots,21$ *and the argument values of the cdf* $x_i, i=1,\dots,20$ *are given in Table 1. As the sum of the 8 smallest argument values is greater than one, the cdf value can be calculated by our Lauricella series expansion based recursive algorithm. If we restrict the summation upper limits to 20 in the Lauricella series expansion, the result is accurate only for two digits. If the summation upper limits are increased to 30, then the result becomes accurate, but the CPU time of the calculations increases significantly, too. The gap between the* ${}_0L_1$ *lower and the* ${}_1U_1$ *upper bound is small and the variance reduction simulation produces accurate estimation with* $N=100,000$ *sample size. In Table 2 the Crude Monte Carlo estimator with the same sample size and the efficiency*

Table 1: Parameter and cdf argument values of the Dirichlet distribution in Example 2.2.

index	ϑ values	x values	index	ϑ values	x values
1	1.5	0.3	11	1.5	0.3
2	1.4	0.2	12	1.4	0.2
3	1.3	0.2	13	1.3	0.2
4	1.2	0.2	14	1.2	0.2
5	1.1	0.2	15	1.1	0.2
6	1.2	0.3	16	1.2	0.3
7	1.3	0.2	17	1.3	0.2
8	1.4	0.3	18	1.4	0.3
9	1.2	0.2	19	1.2	0.2
10	1.3	0.3	20	1.3	0.3
			21	2.1	

of the variance reduction simulation compared to the crude Monte Carlo estimation are also given. The efficiency means the ratio of the estimated variance times CPU time for the two compared simulation procedures. Observe that the (0,1)–hypermultitree bound is lower bound and the (1,1)–hypermultitree bound is upper bound for the cdf as it is probability of intersection of events and not union of events.

2.3. Example *Let us regard the same 20–dimensional Dirichlet distribution as in Example 2.2, but the argument values of the cdf let be modified such a way that replace all 0.3 values by 0.1. These new* $x_i, i = 1, \ldots, 20$ *values are given in Table 3. Now the sum of the 8 smallest argument values is less than one, so the Lauricella series expansion based recursive algorithm cannot be applied. The gap between the* ${}_0L_1$ *lower and the* ${}_1U_1$ *upper bound is not as small as it was before, however the variance reduction simulation produces quite accurate estimation with* $N = 100,000$ *sample size in reasonable CPU time.*

Table 2: Numerical results for the Dirichlet cdf of Example 2.2.

Lauricella ($m = 20$)	=	0.941206
CPU in seconds	=	21.58
Lauricella ($m = 30$)	=	0.943113
CPU in seconds	=	189.11
Estimated value (crude)	=	0.943270
Std. deviation (crude)	=	0.000732
CPU in seconds	=	1.21
Lower bound ($_0L_1$)	=	0.942888
Upper bound ($_1U_1$)	=	0.943113
Estimated value (var. red.)	=	0.943113
Std. deviation (var. red.)	=	0.000010
CPU in seconds	=	3.57
Efficiency	=	1813.72

Table 3: Cdf argument values of the Dirichlet distribution in Example 2.3.

index	x values	index	x values	index	x values	index	x values
1	0.1	6	0.3	11	0.1	16	0.1
2	0.2	7	0.2	12	0.2	17	0.2
3	0.2	8	0.1	13	0.2	18	0.1
4	0.2	9	0.2	14	0.2	19	0.2
5	0.2	10	0.1	15	0.2	20	0.1

Table 4: Numerical results for the Dirichlet cdf of Example 2.3.

Estimated value (crude)	=	0.367700
Std. deviation (crude)	=	0.001525
CPU in seconds	=	1.21
Lower bound $({}_0L_1)$	=	0.238344
Upper bound $({}_1U_1)$	=	0.377891
Estimated value (var. red.)	=	0.367073
Std. deviation (var. red.)	=	0.000255
CPU in seconds	=	3.57
Efficiency	=	12.12
Lower bound $({}_0L_2)$	=	0.290188
Upper bound $({}_1U_1)$	=	0.377891
CPU in seconds	=	3.63
Lower bound $({}_0L_3)$	=	0.322980
Upper bound $({}_1U_2)$	=	0.371765
CPU in seconds	=	10.76
Lower bound $({}_0L_4)$	=	0.345615
Upper bound $({}_1U_3)$	=	0.368858
CPU in seconds	=	211.79

In Table 4 again the Crude Monte Carlo estimator with the same sample size and the efficiency of the variance reduction simulation compared to the crude Monte Carlo estimation are given. The efficiency of the variance reduction simulation is much less than it was before. The higher order ${}_0L_m, {}_1U_{m-1}, m = 2, 3, 4$ lower and upper bounds were also determined, however their CPU calculation time increased quickly.

2.4. Example *Let us generate a 5–dimensional Dirichlet distribution from the 20–dimensional Dirichlet distribution of Example 2.2 so, that keep the first five parameter and argument values and the sixth parameter value be equal the value of* ϑ_{21} *in Example 2.2. In Table 5 we give the estimated cdf value, and in Table 6 the estimated gradient*

Table 5: Numerical results for the Dirichlet cdf of Example 2.4.

Estimated value (crude)	=	0.134431
Std. deviation (crude)	=	0.000341
CPU in seconds	=	2.78
Lower bound ($_0L_1$)	=	0.000000
Upper bound ($_1U_1$)	=	0.157245
Estimated value (var. red.)	=	0.134502
Std. deviation (var. red.)	=	0.000102
CPU in seconds	=	2.66
Efficiency	=	11.69

Table 6: Estimators of the gradient vector and Hessian matrix elements in Example 2.4.

	estimator		estimator		estimator		estimator
g_1	0.717479	h_{11}	-1.191840	h_{33}	-1.645818	h_{51}	2.035615
g_2	0.657314	h_{21}	3.682934	h_{41}	2.803651	h_{52}	1.848502
g_3	0.597790	h_{22}	-1.449139	h_{42}	2.587722	h_{53}	1.664471
g_4	0.538996	h_{31}	3.575082	h_{43}	2.367718	h_{54}	1.483955
g_5	0.442658	h_{32}	3.291370	h_{44}	-1.782822	h_{55}	-1.110385

vector and Hessian matrix elements.

2.5. Example *Let us generate a 5–dimensional Dirichlet distribution from the 20– dimensional Dirichlet distribution of Example 2.3 in the same way as it was done in the case of Example 2.4 from the 20–dimensional Dirichlet distribution of Example 2.2. In this case the cdf value is small enough and the variance reduction simulation procedure is not very effective. Hence in Table 7 we give also the results of the* **SCIS** *simulation method (Gouda and Szántai ([48])), which is more effective for this example and becomes even more better for the estimation of smaller cdf values. We will describe* **SCIS** *in the next section.*

Table 7: Numerical results for the Dirichlet cdf of Example 2.5.

Estimated value (crude)	=	0.035870
Std. deviation (crude)	=	0.000588
CPU in seconds	=	0.55
Lower bound ($_0L_1$)	=	0.000000
Upper bound ($_1U_1$)	=	0.150463
Estimated value (var.red.)	=	0.034756
Std. deviation (var. red.)	=	0.000370
CPU in seconds	=	0.50
Efficiency (var. red.)	=	2.77
Estimated value (SCIS)	=	0.034971
Std. deviation (SCIS)	=	0. 000031
CPU in seconds	=	12.99
Efficiency (SCIS)	=	15.23

2.5 New Sampling Techniques

In this section, we will describe and compare several simulation algorithms for computing the cumulative distribution function values of Dirichlet distribution. In the previous section, we have described the variance reduction simulation procedure for the estimation of Dirichlet probabilities which are most effective when the estimated probability value is close to one, but in case of small probability values (rare event probability) is not very effective. Thus we will apply **SCS** and **SCIS** methods for this purpose. We will show that the simulation algorithms based on **SCS** and **SCIS** techniques can be successfully applied for finding the cumulative distribution function values of Dirichlet distribution.

The **SCS** and **SCIS** techniques were applied first for the estimation of the cumulative distribution function of a multivariate normal distribution by Ambartzumian et al. ([1]). These algorithms will be described first. The application of **SCS** and **SCIS** simulation methods in the estimation of Dirichlet cumulative distribution function values will be

given later in this section. Based on properties of the Dirichlet distribution and Theorem 2.3., new versions of the **SCS** and **SCIS** algorithms will be developed, called **SCSA, SCSB**, **SCISA** and **SCISB**, respectively. We present some numerical results, too.

Crude Monte Carlo Algorithm:

We will call **CMC** the simple "hit-or-miss Monte Carlo" simulation method with conventional sampling of the Dirichlet distributed random vectors (i.e. using the construction given in the formula (2.4)). This is the simplest method for computing the Dirichlet cumulative distribution function values.

2.1. Algorithm *The steps of **CMC** algorithm are as follows:*

1. *Generate N Dirichlet distributed random vectors by each time generating $n+1$ standard gamma distributed Y_i random numbers and using the construction given in the formula (2.4).*

2. *Count the number K of those random points that fall inside the domain of the cdf and estimate*

$$\hat{P} = \frac{K}{N},$$

with variance

$$Var(\hat{P}) = P(1-P)/N,$$

which can be estimated (consistently) by

$$\hat{P}(1-\hat{P})/N.$$

The Sequential Conditioned Sampling Algorithm(SCS)

It is well known that any multivariate probability density function $f(x_1, x_2, \ldots, x_n)$ can be expressed in a product form of a series of one dimensional conditional probability density functions:

$$f(x_1, x_2, \ldots, x_n) = f_1(x_1) f_2(x_2 \mid x_1) \cdots f_n(x_n \mid x_1, x_2, \ldots, x_{n-1}).$$

As in the case of Dirichlet distribution we know the conditional probability distributions involved in the above expression (all of them are one dimensional beta distributions),

the cumulative probability distribution function can be easily expressed in the following form:

$$
\begin{aligned}
&D(a_1,\ldots,a_n;\nu_1,\ldots,\nu_n;\nu_{n+1}) = \\
&= \int_0^{a_1}\cdots\int_0^{a_n} d(x_1,\ldots,x_n;\nu_1,\ldots,\nu_n;\nu_{n+1})dx_n\cdots dx_1 = \\
&= \int_0^{a_1} b\left(x_1;\nu_1;\nu_2+\cdots+\nu_n+\nu_{n+1}\right) \\
&\vdots \\
&\int_0^{\frac{a_k}{1-x_1-\cdots-x_{k-1}}} b\left(x_k;\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1}\right) \\
&\vdots \\
&\int_0^{\frac{a_n}{1-x_1-\ldots-x_{n-1}}} b\left(x_n;\nu_n;\nu_{n+1}\right) dx_n\cdots dx_k\cdots dx_1,
\end{aligned}
\tag{2.25}
$$

where $\nu_1,\ldots,\nu_n,\nu_{n+1}$ are positive parameters of the Dirichlet distribution.

Using the equation (2.25) one can construct the sequential conditioned sampling (**SCS**) algorithm for estimation of the Dirichlet cdf. The outcome of a trial by **SCS** algorithm is random and can be failure or success.

2.2. Algorithm *The steps of **SCS** algorithm are as follows:*

1. *We generate a random value $X_1 = x_1$ from beta distribution with parameters $\nu_1, \nu_2+ \ldots +\nu_n + \nu_{n+1}$.*

2. *If $x_1 \notin [0, a_1]$ the trial is terminated and the outcome is failure, if $x_1 \in [0,\ a_1]$ we go to the next step*

3. *If in the first $k-1$ steps we have success, then the outcome of $k-1$ steps satisfy $0 \le x_i \le \dfrac{a_i}{1-x_1-\cdots x_{i-1}}$, $i = 1, \ldots, k-1$. Thus we generate $X_k = x_k$ using beta distribution with parameters $\nu_k, \nu_{k+1}+ \ldots + \nu_n + \nu_{n+1}$.*

4. *If $x_k \notin [0,\ \dfrac{a_k}{1-x_1-\cdots x_{i-1}}]$ then the trial is terminated and the outcome is failure, otherwise $k = k+1$ and repeat the iteration until $k = n$.*

5. *If no failure occur in the iterations then the trial is terminated and the outcome is success.*

*We perform N independent **SCS** trials and estimate $P(Q)$ by*

$$\hat{P} = \frac{number\ of\ successes}{N},$$

with estimated variance

$$var\left(\hat{P}\right) = \frac{\hat{P}(1-\hat{P})}{N}.$$

The Modified Versions of the Sequential Conditioned Sampling Algorithm (SCSA and SCSB)

The **SCS** algorithm will be modified by the application of formulae (2.9) and (2.10) when it becomes possible in the iterations of the algorithm. When trying to apply formula (2.9) we obtain algorithm **SCSA** and when trying to apply formula (2.10) we obtain algorithm **SCSB.**

2.3. Algorithm *The steps of **SCSA** and **SCSB** algorithms are as follows:*

1. *Initialize as $p = 0.0$ and sort the argument values of the cumulative distribution function and the according parameters of the Dirichlet distribution in increasing order of the argument values.*

2. *For $i = 1$, to n do:*

Generate a random value $X_i = x_i$ from beta distribution with parameters $\nu_i, \nu_{i+1} + ... + \nu_n + \nu_{n+1}$.

Case SCSA: *If for any value i the inequality $\frac{a_{i+1}}{1 - x_1 - \cdots - x_i} + \frac{a_{i+2}}{1 - x_1 - \cdots - x_i} > 1$ is fulfilled, then let us calculate the value*

$$q = 1 - (n-i) + \\ + \sum_{j=i+1}^{n} B(\frac{a_j}{1 - x_1 - \cdots - x_i}, \nu_j; \nu_{j+1} + \cdots + \nu_n + \nu_{n+1}).$$

As result of the trial let us accept the value q:

$$p = p + q,$$

*otherwise proceed the **SCS** algorithm and if the result of the trial becomes 'success' add one to the value of p else if the result of the trial becomes 'failure' add zero to the value of p i.e. its value remains unchanged.*

Case SCSB: *If for any value i the inequality* $\frac{a_{i+1}}{1-x_1-\cdots-x_i} + \frac{a_{i+2}}{1-x_1-\cdots-x_i} + \frac{a_{i+3}}{1-x_1-\cdots-x_i} > 1$ *is fulfilled, then let us calculate the values*

$$d_1 = \sum_{j=i+1}^{n} B(\frac{a_j}{1-x_1-\cdots-x_i}, \nu_j; \nu_{j+1}+\cdots+\nu_n+\nu_{n+1}),$$
$$d_2 = \sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} D(\frac{a_j}{1-x_1-\cdots-x_i}, \frac{a_k}{1-x_1-\cdots-x_i}; \nu_j, \nu_k; \sum_{l=i+1}^{n+1}\nu_l - \nu_j - \nu_k).$$

Using these two auxiliary values calculate the value

$$q = (n-i-1)(n-i-2)/2 - (n-i-2)d_1 + d_2.$$

and as result of the trial let us accept the value q:

$$p = p + q,$$

*otherwise proceed the **SCS** algorithm and if the result of the trial becomes 'success' add one to the value of p else if the result of the trial becomes 'failure' add zero to the value of p i.e. its value remains unchanged.*

The Sequential Conditioned Importance Sampling Algorithm (SCIS)

First we introduce importance sampling densities in the equation (2.25) i.e. we take

truncated beta sampling densities in the following way.

$$
\begin{aligned}
&D(a_1,\ldots,a_n;\nu_1,\ldots,\nu_n;\nu_{n+1}) = \\
&= \int_0^{a_1} b\left(x_1;\nu_1;\nu_2+\cdots+\nu_n+\nu_{n+1}\right) \\
&\vdots \\
&\int_0^{\frac{a_k}{1-x_1-\cdots-x_{k-1}}} b\left(x_k;\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1}\right) \\
&\vdots \\
&\int_0^{\frac{a_n}{1-x_1-\ldots-x_{n-1}}} b\left(x_n;\nu_n;\nu_{n+1}\right) dx_n\cdots dx_k\cdots dx_1 = \\
&= \int_0^1 B(a_1;\nu_1;\nu_2+\ldots+\nu_n+\nu_{n+1})\cdot \\
&\cdot\frac{b\left(x_1;\nu_1;\nu_2+\cdots+\nu_n+\nu_{n+1}\right)}{B(a_1;\nu_1;\nu_2+\ldots+\nu_n+\nu_{n+1})} I_{[0,a_1]}(x_1) \\
&\vdots \\
&\int_0^1 B\left(\frac{a_k}{1-x_1-\ldots-x_{k-1}};\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1}\right)\cdot \\
&\cdot\frac{b\left(x_k;\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1}\right)}{B\left(\frac{a_k}{1-x_1-\ldots-x_{k-1}};\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1}\right)} I_{\left[0,\frac{a_k}{1-x_1-\cdots-x_{k-1}}\right]}(x_k) \\
&\vdots \\
&\int_0^1 B\left(\frac{a_n}{1-x_1-\ldots-x_{n-1}};\nu_n;\nu_{n+1}\right)\cdot \\
&\cdot\frac{b\left(x_n;\nu_n;\nu_{n+1}\right)}{B\left(\frac{a_n}{1-x_1-\ldots-x_{n-1}};\nu_n;\nu_{n+1}\right)} I_{\left[0,\frac{a_n}{1-x_1-\ldots-x_{n-1}}\right]}(x_n) dx_n\cdots dx_k\cdots dx_1
\end{aligned}
\tag{2.26}
$$

The outcome of a trial by **SCIS** algorithm is a real number and each trial consists of exactly n steps.

2.4. Algorithm *The steps of **SCIS** algorithm are as follows:*

1. *Generate a random value* x_1 *from the pdf*

$$\frac{b\,(x_1;\nu_1;\nu_2+\cdots+\nu_n+\nu_{n+1})}{B(a_1;\nu_1;\nu_2+...+\nu_n+\nu_{n+1})}I_{[0,a_1]}(x_1).$$

2. *After the values* $x_1,...,x_{k-1}$ *have been sampled, we sample a random value* x_k *from the pdf*

$$\frac{b\,(x_k;\nu_k;\nu_{k+1}+\cdots+\nu_n+\nu_{n+1})}{B(\frac{a_k}{1-x_1-...-x_{k-1}};\nu_k;\nu_{k+1}+\cdots+\nu_{n+1})}I_{[0,\frac{a_k}{1-x_1-...-x_{k-1}}]}(x_k).$$

3. *The trial is terminated after we generate the nth value* x_n.

4. *Compute the value of random variable*

$$T=\prod_{i=1}^{n}B(\frac{a_i}{1-x_1-\cdots-x_{i-1}};\nu_i;\nu_{i+1}+...+\nu_n+\nu_{n+1}).$$

We perform N *independent* ***SCIS*** *trials and obtain the* N *outcomes* $t_1,...,t_N$ *of the random variable* T *in Step 4. Then estimate*

$$\hat{P}=\frac{1}{N}\sum_{i=1}^{N}t_i,$$

with estimated variance

$$var\left(\hat{P}\right)=(\frac{1}{N}\sum_{i=1}^{N}t_i^2-\hat{P}^2)/N.$$

The Modified Versions of the Sequential Conditioned Importance Sampling Algorithm(SCISA and SCISB)

The **SCIS** algorithm will be modified by the application of formulae (2.9) and (2.10) when it becomes possible in the iterations of the algorithm. When trying to apply formula (2.9) we obtain algorithm **SCISA** and when trying to apply formula (2.10) we obtain algorithm **SCISB.**

2.5. Algorithm *The steps of* ***SCISA*** *and* ***SCISB*** *algorithms are as follows:*

1. *Initialization: Let* $p=0.0$, $T=1.0$ *and sort the components of the Dirichlet distributed random vector* $\mathbf{X}$ *according to increasing order of* $x_1,\ldots,x_n$.

2. *Iteration step: For $i = 1$ to n generate x_i from pdf*

$$\frac{b\,(x_i;\nu_i;\nu_{i+1}+\cdots+\nu_n+\nu_{n+1})}{B(\frac{a_i}{1-x_1-\cdots-x_{i-1}};\nu_i;\nu_{i+1}+\cdots+\nu_n+\nu_{n+1})}\ I_{[0,\frac{a_i}{1-x_1-\cdots-x_{i-1}}]}(x_i)$$

calculate

$$T = TB(\frac{a_i}{1-x_1-...-x_{i-1}};\nu_i;\nu_{i+1}+...+\nu_n+\nu_{n+1}).$$

Case SCISA : *If for any value i the inequality*

$$\frac{a_{i+1}}{1-x_1-\cdots-x_i}+\frac{a_{i+2}}{1-x_1-\cdots-x_i}>1,$$

is fulfilled, then let us calculate the value

$$q = 1-(n-i)+$$
$$+\sum_{j=i+1}^{n} B(\frac{a_j}{1-x_1-\cdots-x_i};\nu_j,\nu_{j+1}+\cdots+\nu_{j+1}),$$

let $T = Tq$ and accept $p = p + T$ as result of the trial else proceed with computing the result of the trial from the ***SCIS*** *algorithm.*

Case SCISB: *If for any value i the inequality*

$$\frac{a_{i+1}}{1-x_1-\cdots-x_i}+\frac{a_{i+2}}{1-x_1-\cdots-x_i}+\frac{a_{i+3}}{1-x_1-\cdots-x_i}>1,$$

is fulfilled, then let us calculate the values

$$d_1 = \sum_{j=i+1}^{n} B(\frac{a_j}{1-x_1-\cdots-x_i},\nu_j;\nu_{j+1}+\cdots+\nu_n+\nu_{n+1}),$$

$$d_2 = \sum_{j=i+1}^{n-1}\sum_{k=j+1}^{n} D(\frac{a_j}{1-x_1-\cdots-x_i},\frac{a_k}{1-x_1-\cdots-x_i};$$
$$\nu_j,\nu_k;\sum_{l=i+1}^{n+1}\nu_l-\nu_j-\nu_k),$$

Using these two auxiliary values calculate the value

$$q = (n-i-1)(n-i-2)/2-(n-i-2)d_1+d_2,$$

let $T = Tq$ and accept $p = p + T$ as result of the trial else proceed with computing the result of the trial from the ***SCIS*** *algorithm.*

Table 8: Comparison of the algorithms: high probability case

Dimension $= 10, N = 100000, x_1 = x_2 = \cdots = x_{10} = 0.3$ Parameters $= 1.1, 1.2, 1.3, 1.4, 1.5, 1.1, 1.2, 1.3, 1.4, 1.5, 4.1$				
	Est.value	**Std.deviation**	**Time(sec)**	**Efficiency**
CMC	0.93058	8.04×10^{-4}	3.13	1.00
SCS	**0.93144**	$\mathbf{7.99 \times 10^{-4}}$	**2.97**	**1.07**
SCSA	0.93117	7.07×10^{-4}	5.33	0.76
SCSB	0.93047	3.99×10^{-4}	208.27	0.06
SCIS	0.93080	1.87×10^{-4}	103.09	0.56
SCISA	0.93106	1.81×10^{-4}	69.75	0.88
SCISB	0.93067	1.18×10^{-4}	169.17	0.86

Numerical Results

We present different examples to estimate the Dirichlet cumulative distribution function value and compare the numerical efficiency of **SCS, SCSA, SCSB, SCIS, SCISA and SCISB** methods with **CMC** method.

The sequential conditioned sampling and importance sampling can be applied in developing of effective algorithms for estimating the cumulative probability distribution function values of Dirichlet distribution. Although in the case of the multivariate normal distribution these sampling techniques proved to be more effective than the classical sampling techniques in almost the whole possible domain of the estimated probability value this is not true for the case of Dirichlet distribution. We have seen that these new sampling techniques, mainly importance sampling became effective enough only for the estimation of very small probability values. This difference may be caused by the fact that the classical sampling technique for Dirichlet distribution, i.e. transformation of Dirichlet distributed random vector from independent standard gamma distributed random numbers is fast enough while the same for the multivariate normal distribution, i.e. transformation of multivariate normally distributed random vector from independent standard normally distributed random numbers is quite slow. The modified versions of

Table 9: Comparison of the algorithms: medium probability case

Dimension $= 10, N = 100000, x_1 = x_2 = \cdots = x_{10} = 0.2$ Parameters $= 1.1, 1.2, 1.2, 1.4, 1.5, 1.1, 1.2, 1.2, 1.4, 1.5, 4.1$				
	Est.value	**Std.deviation**	**Time(sec)**	**Efficiency**
CMC	0.53278	1.58×10^{-3}	2.97	1.00
SCS	**0.53606**	$\mathbf{1.58 \times 10^{-3}}$	**2.14**	**1.39**
SCSA	0.53530	1.55×10^{-3}	2.74	1.12
SCSB	0.53545	1.31×10^{-3}	32.03	0.13
SCIS	0.53589	4.86×10^{-4}	126.54	0.25
SCISA	0.53567	4.87×10^{-4}	145.67	0.21
SCISB	0.53610	4.66×10^{-4}	105.51	0.32

Table 10: Comparison of the algorithms: low probability case

Dimension $= 10, N = 100000, x_1 = x_2 = \cdots = x_{10} = 0.1$ Parameters $= 1.1, 1.2, 1.3, 1.4, 1.5, 1.1, 1.2, 1.3, 1.4, 1.5, 4.1$				
	Est.value	**Std.deviation**	**Time(sec)**	**Efficiency**
CMC	6.80×10^{-3}	2.60×10^{-4}	3.02	1.00
SCS	6.82×10^{-3}	2.60×10^{-3}	1.09	2.74
SCSA	6.80×10^{-3}	2.60×10^{-4}	0.93	3.25
SCSB	6.92×10^{-3}	2.62×10^{-4}	1.26	2.36
SCIS	7.12×10^{-3}	1.35×10^{-5}	112.38	9.93
SCISA	7.13×10^{-3}	1.34×10^{-5}	144.89	7.84
SCISB	$\mathbf{7.10 \times 10^{-3}}$	$\mathbf{1.33 \times 10^{-5}}$	**112.71**	**10.22**

Table 11: Comparison of the algorithms: very low probability case

Dimension $= 10, N = 100000, x_1 = x_2 = \cdots = x_{10} = 0.075$ Parameters $= 1.1, 1.2, 1.3, 1.4, 1.5, 1.1, 1.2, 1.3, 1.4, 1.5, 4.1$				
	Est.value	**Std.deviation**	**Time(sec)**	**Efficiency**
CMC	2.99×10^{-4}	5.38×10^{-5}	2.96	1.00
SCS	4.60×10^{-4}	6.78×10^{-5}	0.71	2.63
SCSA	3.10×10^{-4}	5.57×10^{-5}	0.71	3.90
SCSB	3.30×10^{-4}	5.74×10^{-7}	0.72	3.61
SCIS	$\mathbf{3.71 \times 10^{-4}}$	$\mathbf{5.57 \times 10^{-7}}$	**114.74**	**241.27**
SCISA	3.72×10^{-4}	5.58×10^{-7}	145.93	188.75
SCISB	3.72×10^{-4}	5.53×10^{-7}	121.01	231.56

the new algorithms (**SCSA, SCSB, SCISA and SCISB**) reduced the variance of the estimation significantly however they needed too much CPU time, so their efficiency has been destroyed.

3 Probability Estimation in Network Models

In the first part of this chapter, we describe the estimation of rare event probabilities in stochastic networks. The well known variance reduction technique, called Importance Sampling (IS) is an effective tool for doing this. The main idea of IS is to simulate the random system under a modified set of parameters, so as to make the occurrence of the rare event more likely. The major problem of the IS technique is that the optimal modified parameters, called reference parameters to be used in IS are usually very difficult to obtain. Rubinstein ([92]) developed the CE method for the solution of this problem of IS technique and then he and his collaborators applied this for estimation of rare event probabilities in stochastic networks with exponential distribution (see De Boer, Kroese, Mannor and Rubinstein ([10]) . In this chapter we test this simulation technique also for medium sized stochastic networks and compare its effectiveness to the simple CMC simulation. The effectiveness of a variance reduction simulation algorithm is measured in the following way. We calculate the product of the necessary CPU time and the estimated variance of the estimation. This product is compared to the same for the simple Crude Monte Carlo simulation. This was originally used for comparison of different variance reduction techniques by Hammersley and Handscombe ([55]).

The main result of the first part of this chapter is the extension of CE method for estimation of rare event probabilities in stochastic networks with normal and beta distributions. In this case the calculation of reference parameters of the importance sampling distribution requires numerical solution of a nonlinear equation system. This is done by applying a Newton–Raphson iteration scheme. In this case the CPU time spent for calculation of the reference parameter values can not be neglected. Numerical results will also be presented.

In the second part of this chapter, a stochastic programming based PERT modeling will be introduced. This modeling will produce deterministic earliest starting times for the activities of the project. These deterministic starting times will be attainable with prescribed probability. So we also get an estimated finishing time of the project what is realizable with the same prescribed probability. Moderate sized numerical examples

will be given for comparing the traditional and the newly introduced PERT modeling techniques.

3.1 Cross Entropy Algorithm for the Shortest Path Problem

3.1.1 Introduction

Stochastic simulation has proven itself in practice; it is commonly used for accurate estimations in many problems. However, there are some limitations to the standard stochastic simulation method in many cases. An important class of problems that cannot be efficiently solved using standard simulation is that involving rare events. Since these rare events occur so seldom in a standard simulation, one has to apply very large sample sizes what need too much CPU time. This is why different methods and techniques have been developed to estimate rare event probabilities starting at the last decade.

The rest of this section is organized in the following way. we will discuss the basic methodology behind the CE algorithm. The basic CE algorithm for rare event simulation will also be given. The application of the basic CE algorithm for the shortest path problem in stochastic networks will be described. The basic CE algorithm will be specialized for the shortest path problem with exponential, beta and normal distributed activity duration times. We will discuss two modifications of the basic CE algorithm for rare event simulation. The first modification is new result, the second was published by Homem-de Mello and Rubinstein ([56]). The specialization of the modified algorithms for the shortest path problem will not be given, it can be done in the same way as it happened in the case of the basic CE algorithm. Numerical results for the comparison of CMC and IS algorithms based on CE reference parameter estimation will be presented.

3.1.2 Importance Sampling, Cross Entropy and Rare Event Probabilities

We briefly review the ideas behind the IS and CE methods for rare event simulation. Let us regard a random system described with a random vector $\mathbf{X}$ taking on its values in some space $\mathcal{X}$. Let $f(\mathbf{x}, \mathbf{v})$ be a parametric family of probability densities on $\mathcal{X}$. Let us

suppose that the investigated random vector $\mathbf{X}$ has probability density function $f(\mathbf{x}, \mathbf{v}^\star)$, where $\mathbf{v}^\star$ is a fixed parameter vector. Let $S(\mathbf{X})$ be a score function of the investigated random system. We are looking for the probability that $S(\mathbf{X})$ is greater than some real number γ. So we wish to estimate

$$L = P(S(\mathbf{X}) > \gamma) = E[H(\mathbf{X})], \tag{3.1}$$

by using discrete event simulation, where E is the expected value and $H(\mathbf{x})$ is the indicator function given by

$$H(\mathbf{x}) = I_{\{S(\mathbf{X})>\gamma\}} = \begin{cases} 1, & \text{if } S(\mathbf{x}) > \gamma, \\ 0, & \text{otherwise.} \end{cases}$$

Equation (3.1) can be estimated straightforward way by CMC simulation, we draw a random sample $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}$ from $f(\mathbf{x}; \mathbf{v}^\star)$ and compute the following unbiased estimator for L

$$\hat{L} = \frac{1}{n} \sum_{i=1}^{n} H(\mathbf{X}^{(i)}). \tag{3.2}$$

When γ is large the probability L will be very small and we say that $\{S(\mathbf{x}) > \gamma\}$ is a rare event. In this case CMC requires a large sample size in order to estimate L accurately (this means that n is very large). An alternative is to use IS (Rubinstein and Melamed ([97]), Rubinstein ([92]) and Rubinstein ([93])), i.e. change measure in the following way

$$\begin{aligned} L = P(S(\mathbf{X}) > \gamma) &= E_f\left[H(\mathbf{X})\right] \\ &= \int I_{\{S(\mathbf{X})>\gamma\}} f(\mathbf{x}; \mathbf{v}^\star) d\mathbf{x} \\ &= \int I_{\{S(\mathbf{X})>\gamma\}} \frac{f(\mathbf{x}, \mathbf{v}^\star)}{g(\mathbf{x})} g(\mathbf{x}) dx \\ &= \int I_{\{S(\mathbf{X})>\gamma\}} W(\mathbf{x}, \mathbf{v}^\star) g(\mathbf{x}) dx \\ &= E_g\left[H(\mathbf{X}) W(\mathbf{X}; \mathbf{v}^\star)\right], \end{aligned} \tag{3.3}$$

where $W(\mathbf{x};\mathbf{v}^\star) = f(\mathbf{x};\mathbf{v}^\star)/g(\mathbf{x})$ is the so called likelihood ratio. Now drawing a random sample $\mathbf{X}^{(1)},\cdots,\mathbf{X}^{(n)}$ from the probability density function $g(\mathbf{x})$, we can compute the following estimator

$$\hat{L} = \frac{1}{n}\sum_{i=1}^{n} H(\mathbf{X}^{(i)})W(\mathbf{X}^{(i)};\mathbf{v}^\star). \tag{3.4}$$

This is called IS estimator of L and it is also an unbiased estimator.

The aim is now to change for the best possible measure, i.e. to fix the probability density function $g(\mathbf{x})$ so that the resulting IS estimator have minimal variance. It is easy to see that the solution of this problem is the following probability density function

$$g(\mathbf{x}) = CH(\mathbf{x})f(\mathbf{x};\mathbf{v}^\star) \tag{3.5}$$

with normalization factor

$$C^{-1} = \int H(\mathbf{x})f(\mathbf{x};\mathbf{v}^\star)d\mathbf{x},$$

i.e. $g(\mathbf{x})$ is the original distribution conditioned on the rare event. The IS estimator based on the probability density function (3.5) would have variance zero, however we can not use this as the normalization factor is not at hand, it is exactly that value what we are just looking for, i.e. $E_f\left[H(\mathbf{X})\right] = P(S(\mathbf{X}) > \gamma) = L$. So we must look after some good approximation to the optimal probability density function given by equation (3.5).

For measuring the distance between two probability measures one can use the Kullback–Leibler cross entropy, which is defined as follows. Let $g(\mathbf{x})$ and $h(\mathbf{x})$ be two probability density functions. Their Kullback–Leibler cross entropy is defined as

$$CE = \int g(\mathbf{x}) \log \frac{g(\mathbf{x})}{h(\mathbf{x})} d\mathbf{x}. \tag{3.6}$$

If $g(\mathbf{x})$ and $h(\mathbf{x})$ have identical distributions, then $CE = 0$; otherwise, $CE > 0$.

When looking for approximation of the optimal probability density function given by equation (3.5) we restrict our attention to those probability densities only which belong to the same parametric family as our original probability density does. This way our problem is now to find a parameter vector $\hat{\mathbf{v}}$, called tilting parameter, such that the probability density function $f(\mathbf{x};\hat{\mathbf{v}})$ approximates the best possible probability density function (3.5) as well as it is possible. To do this, substitute in equation (3.6)

the probability density function $g(\mathbf{x})$ by the optimal probability density function given in equation (3.5) and the probability density function $h(\mathbf{x})$ by $f(\mathbf{x};\mathbf{v})$ and choose $\hat{\mathbf{v}}$ to minimize this expression.

$$\begin{aligned}
\hat{\mathbf{v}} &= \arg\min_{\mathbf{v}} \int CH(\mathbf{x}) f(\mathbf{x};\mathbf{v}^\star) \log \frac{CH(\mathbf{x}) f(\mathbf{x},\mathbf{v}^\star)}{f(\mathbf{x};\mathbf{v})} dx \\
&= \arg\min_{\mathbf{v}} [\int CH(\mathbf{x}) f(\mathbf{x};\mathbf{v}^\star) \log CH(\mathbf{x}) f(\mathbf{x},\mathbf{v}^\star) dx \\
&\quad - \int CH(\mathbf{x}) f(\mathbf{x};\mathbf{v}^\star) \log f(\mathbf{x};\mathbf{v}) dx] \\
&= \arg\max_{\mathbf{v}} \int H(\mathbf{x}) \log f(\mathbf{x};\mathbf{v}) f(\mathbf{x};\mathbf{v}^\star) dx \\
&= \arg\max_{\mathbf{v}} \int H(\mathbf{x}) \log f(\mathbf{x};\mathbf{v}) \frac{f(\mathbf{x};\mathbf{v}^\star)}{f(\mathbf{x};\mathbf{v}_k)} f(\mathbf{x};\mathbf{v}_k) dx \\
&= \arg\max_{\mathbf{v}} E_{\mathbf{v}_k}[H(\mathbf{x}) W(\mathbf{x};\mathbf{v}^\star,\mathbf{v}_k) \log f(\mathbf{x};\mathbf{v})],
\end{aligned} \tag{3.7}$$

where

$$W(\mathbf{x};\mathbf{v}^\star,\mathbf{v}_k) = \frac{f(\mathbf{x};\mathbf{v}^\star)}{f(\mathbf{x};\mathbf{v}_k)},$$

and $\mathbf{v}_k$ is any other tilting vector.

Therefore, the main CE algorithm for rare event simulation by using a smoothing parameter $0 \leq \lambda \leq 1$ (De Boer at el. ([9])) is

3.1. Algorithm *Basic CE algorithm for rare event simulation.*

1. *Set a starting value* $\mathbf{v}_0$. *An appropriate choice may be* $\mathbf{v}_0 = \mathbf{v}^\star$. *Let* $k = 0$ *and start the iteration at step 2.*

2. *Generate a random sample* $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}$ *from* $f(\mathbf{x};\mathbf{v}_k)$.

3. *Compute the score function* $S^{(i)} = S(\mathbf{X}^{(i)})$ *for all* i *and order them from smallest to biggest,* $S^{(1)} \leq \cdots \leq S^{(n)}$. *Let* $\hat{\gamma}_{k+1}$ *be the* $(1-\rho)$ *sample quantile of performances* $\hat{\gamma}_{k+1} = S^{([(1-\rho)n])}$, *provided this is less than* γ. *Otherwise set* $\hat{\gamma}_{k+1} = \gamma$.

4. *Use the same sample to estimate the vector* $\mathbf{v}_{k+1}$ *by solving the following equation with respect to* $\mathbf{v}$

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \mathbf{v}^{\star}, \mathbf{v}_k) \nabla \log f(\mathbf{X}^{(i)}; \mathbf{v}) = \mathbf{0}^T. \tag{3.8}$$

5. *Smooth the new parameter vector by* $\mathbf{v}_{k+1} := \lambda \mathbf{v}_{k+1} + (1-\lambda)\mathbf{v}_k$, *where* $0 \leq \lambda \leq 1$ *is a fixed smoothing parameter.*

6. *If* $\hat{\gamma}_{k+1} = \gamma$ *then denote the final tilting parameter vector* $\mathbf{v}_{k+1}$ *by* $\hat{\mathbf{v}}$ *and go to step 7; otherwise set* $k = k+1$ *and repeat steps 2–6 until* $\hat{\gamma}_{k+1} = \gamma$ *is reached.*

7. *Generate a random sample* $X^{(1)}, \ldots, X^{(N)}$ *with the final tilting parameter vector* $\hat{\mathbf{v}}$. *Estimate the value of rare event probability by the IS estimator:*

$$\hat{L} = \frac{1}{N} \sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W(\mathbf{X}^{(i)}; \mathbf{v}^{\star}, \hat{\mathbf{v}}),$$

with estimated variance

$$\left(\frac{1}{N} \sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W^2(\mathbf{X}^{(i)}; \mathbf{v}^{\star}, \hat{\mathbf{v}}) - \hat{L}^2 \right) / N.$$

3.1.3 Application of the Basic CE Algorithm for the Shortest Path Problem

The shortest path problem is to find the shortest path from one specific node to another in a network. The arcs connecting successive nodes on a path must point in the direction of travel. Such paths are sometimes referred to as directed paths. To determine a shortest path, we assume that we have a random network $G(N, A)$, where N is the set of nodes and A is the set of arcs. The arcs represent the activity durations with random variables $X^{(1)}, \cdots, X^{(m)}$. Naturally, we assume that these arcs are nonnegative. To find the shortest path from one node (say, O) to another (say, R), we will see that it is necessary to compute the shortest path from many, perhaps all, other nodes to R. Hence, we define the shortest path problem as the problem of finding the shortest path from every node in N to a specific node $R \in N$. Let $S(X)$ be the shortest path from the source node O to the destination node R.

Let us assume first that the activity durations $X^{(1)}, \ldots, X^{(m)}$ are independent and exponentially distributed with means $\alpha_1^\star, \ldots, \alpha_m^\star$ and so their joint probability density function is given by

$$f(\mathbf{x}; \alpha^\star) = \exp\left(-\sum_{j=1}^{m} \frac{x_j}{\alpha_j^\star}\right) \prod_{j=1}^{m} \frac{1}{\alpha_j^\star}. \tag{3.9}$$

The likelihood ratio involved in (3.8) is now given by

$$\begin{aligned} W(\mathbf{X}^{(i)}; \alpha^\star, \alpha_k) &= \frac{f(\mathbf{X}^{(i)}; \alpha^\star)}{f(\mathbf{X}^{(i)}; \alpha_k)} \\ &= \exp\left(-\sum_{j=1}^{m} X_j^{(i)} \left(\frac{1}{\alpha_j^\star} - \frac{1}{\alpha_{kj}}\right)\right) \prod_{j=1}^{m} \frac{\alpha_{kj}}{\alpha_j^\star}, \end{aligned}$$

where $\alpha_k = (\alpha_{k1}, \ldots, \alpha_{km})^T$ is the parameter vector of the k-th iteration and $\mathbf{X}^{(1)}, \cdots, \mathbf{X}^{(n)}$ is a random sample from $f(\mathbf{x}; \alpha_k)$. The logarithm of the probability density function $f(\mathbf{x}; \alpha)$ is

$$\log f(\mathbf{x}; \alpha) = -\sum_{j=1}^{m} \log \alpha_j - \sum_{j=1}^{m} \frac{x_j}{\alpha_j}.$$

After partial derivation according to α_j:

$$\frac{\partial \log f(\mathbf{x}; \alpha)}{\partial \alpha_j} = -\frac{1}{\alpha_j} + \frac{x_j}{\alpha_j^2}.$$

So equation (3.8) will now be specialized to:

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)}) > \hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \alpha_k) \left(-\frac{1}{\alpha_j} + \frac{x_j}{\alpha_j^2}\right) = 0, \; j = 1, \ldots, m. \tag{3.10}$$

From equation (3.10) $\alpha_j, j = 1, \ldots, m$ can be expressed explicitly so Algorithm 3.1. will become the following.

3.2. Algorithm *Basic CE algorithm for the shortest path problem with exponentially distributed activity duration times.*

1. *Set the starting value $\alpha_0 = \alpha^\star$. Let $k = 0$ and start the iteration at step 2.*
2. *Generate a random sample $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}$ from the exponential distribution $f(\mathbf{x}; \alpha_k)$.*

3. *Compute the shortest paths* $S^{(i)} = S(\mathbf{X}^{(i)})$ *for all* i *and order them from smallest to biggest,* $S^{(1)} \leq \cdots \leq S^{(n)}$. *Let* $\hat{\gamma}_{k+1}$*be the* $(1-\rho)$ *sample quantile of shortest paths* $\hat{\gamma}_{k+1} = S^{[(1-\rho)n]}$, *provided this is less than* γ. *Otherwise set* $\hat{\gamma}_{k+1} = \gamma$.

4. *Use the same sample to estimate the components of the parameter vector* α_{k+1} *by the following explicit formula*

$$\alpha_{k+1,j} = \frac{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)}) > \hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \alpha_k) X_j^i}{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)}) > \hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \alpha_k)}, \; j = 1, \ldots, m. \tag{3.11}$$

5. *Smooth the new parameter vector by* $\alpha_{k+1} := \lambda \alpha_{k+1} + (1-\lambda)\alpha_k$, *where* $0 \leq \lambda \leq 1$ *is a fixed smoothing parameter.*

6. *If* $\hat{\gamma}_{k+1} = \gamma$ *then denote the final tilting parameter vector* α_{k+1} *by* $\hat{\alpha}$ *and go to step 7; otherwise set* $k = k+1$ *and repeat steps 2–6 until* $\hat{\gamma}_{k+1} = \gamma$ *is reached.*

7. *Generate a random sample* $X^{(1)}, \cdots, X^{(N)}$ *from the exponential distribution with the final tilting parameter vector* $\hat{\alpha}$. *Estimate the value of rare event probability by the IS estimator:*

$$\hat{L} = \frac{1}{N} \sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W(\mathbf{X}^{(i)}; \alpha^\star, \hat{\alpha})$$

with estimated variance

$$\left(\frac{1}{N} \sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W^2(\mathbf{X}^{(i)}; \alpha^\star, \hat{\alpha}) - \hat{L}^2 \right) / N.$$

Let us now assume that the activity durations $X^{(i)}, i = 1, \ldots, m$ are independent random variables distributed according to beta distribution with parameters $\alpha_i^\star, \beta_i^\star, i = 1, \ldots, m$ and so their joint probability density function is given by

$$\begin{aligned} f(\mathbf{x}; \alpha^\star, \beta^\star) &= \prod_{j=1}^{m} \frac{\Gamma(\alpha_j^\star + \beta_j^\star)}{\Gamma(\alpha_j^\star)\Gamma(\beta_j^\star)} \left(\frac{1}{b-a} \right)^{\alpha_j^\star + \beta_j^\star - 1} \\ &\times (x_j - a)^{\alpha_j^\star - 1} (b - x_j)^{\beta_j^\star - 1}, \end{aligned} \tag{3.12}$$

where

$$a \leq x_j \leq b, \; \alpha_j^\star > 0, \; \beta_j^\star > 0, \; j = 1, \ldots, m.$$

The likelihood ratio involved in (3.8) is now given by

$$\begin{aligned} W(\mathbf{X}^{(i)};\alpha^\star,\beta^\star,\alpha_k,\beta_k) &= \frac{f\left(\mathbf{X}^{(i)};\alpha^\star,\beta^\star\right)}{f\left(\mathbf{X}^{(i)};\alpha_{\mathbf{k}},\beta_{\mathbf{k}}\right)} \\ &= \prod_{j=1}^{m}\frac{\Gamma\left(\alpha_j^\star+\beta_j^\star\right)}{\Gamma\left(\alpha_j^\star\right)\Gamma\left(\beta_j^\star\right)}\frac{\Gamma\left(\alpha_{kj}\right)\Gamma\left(\beta_{kj}\right)}{\Gamma\left(\alpha_{kj}+\beta_{kj}\right)}(b-a)^{\alpha_{kj}+\beta_{kj}-\alpha_j^\star-\beta_j^\star} \\ &\quad \times(b-X_j^{(i)})^{\beta_j^\star-\beta_{kj}}(X_j^{(i)}-a)^{\alpha_j^\star-\alpha_{kj}}. \end{aligned}$$

The logarithm of the probability density function $f\left(\mathbf{x};\alpha,\beta\right)$ is

$$\begin{aligned} \log f\left(\mathbf{x};\alpha,\beta\right) &= \textstyle\sum_{j=1}^{m}\left[\log\Gamma(\alpha_j+\beta j)-\log\Gamma(\alpha_j)-\log\Gamma(\beta_j)\right. \\ &+ (1-\alpha_j-\beta_j)\log(b-a)+(\alpha_j-1)\log(x_j-a) \\ &+ \left.(\beta_j-1)\log(b-x_j)\right]. \end{aligned}$$

After partial derivation according to α_j:

$$\frac{\partial\log f\left(\mathbf{x};\alpha,\beta\right)}{\partial\alpha_j}=\psi(\alpha_j+\beta j)-\psi(\alpha_j)-\log(b-a)+\log(x_j-a).$$

After partial derivation according to β_j:

$$\frac{\partial\log f\left(\mathbf{x};\alpha,\beta\right)}{\partial\beta_j}=\psi(\alpha_j+\beta j)-\psi(\beta_j)-\log(b-a)+\log(b-x_j).$$

Here

$$\psi\left(x\right)=\frac{d\log(\Gamma(x))}{dx}=\frac{\frac{d}{dx}\Gamma\left(x\right)}{\Gamma\left(x\right)},$$

is the so called digamma function.

So equation (3.8) will now be specialized to:

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}}W(\mathbf{X}^{(i)};\alpha^\star,\beta^\star,\alpha_k,\beta_k)\times$$

$$\times\left[\psi\left(\alpha_j+\beta_j\right)-\psi\left(\alpha_j\right)-\log(b-a)+\log(X_j^{(i)}-a)\right]=0, \tag{3.13}$$

$$j=1,\ldots,m,$$

and

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \alpha_k, \beta_k) \times$$

$$\times \left[\psi(\alpha_j + \beta_j) - \psi(\beta_j) - \log(b-a) + \log(b - X_j^{(i)})\right] = 0, \tag{3.14}$$

$$j = 1, \ldots, m.$$

From equations (3.13) and (3.14) we obtain the system of nonlinear equations

$$\psi(\alpha_j + \beta_j) - \psi(\alpha_j) - \log(b-a) = A_j,\ j = 1, \ldots, m \tag{3.15}$$

$$\psi(\alpha_j + \beta_j) - \psi(\beta_j) - \log(b-a) = B_j,\ j = 1, \ldots, m, \tag{3.16}$$

where

$$A_j = \frac{-\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \alpha_k, \beta_k) \log(X_j^{(i)} - a)}{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \alpha_k, \beta_k)},$$

and

$$B_j = \frac{-\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \alpha_k, \beta_k) \log(b - X_j^{(i)})}{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \alpha_k, \beta_k)}.$$

Equations (3.15) and (3.16) will be solved in $\alpha_j, \beta_j,\ j = 1, \ldots, m$ by Newton–Raphson method to obtain $\hat{\alpha}_j, \hat{\beta}_j,\ j = 1, \ldots, m$.

3.3. Algorithm *Basic CE algorithm for the shortest path problem with beta distributed activity duration times.*

1. *Set the starting value $\alpha_0 = \alpha^\star, \beta_0 = \beta^\star$. Let $k = 0$ and start the iteration at Step 2.*

2. *Generate a random sample $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}$ from the beta distribution $f(\mathbf{x}; \alpha_k, \beta_k)$.*

3. *Compute the shortest paths $S^{(i)} = S(\mathbf{X}^{(i)})$ for all i and order them from smallest to biggest, $S^{(1)} \leq \cdots \leq S^{(n)}$. Let $\hat{\gamma}_{k+1}$ be the $(1-\rho)$ sample quantile of shortest paths $\hat{\gamma}_{k+1} = S^{[(1-\rho)n]}$, provided this is less than γ. Otherwise set $\hat{\gamma}_{k+1} = \gamma$.*

4. *Use the same sample to estimate the components of the parameter vectors* α_{k+1}, β_{k+1} *from equations (3.15), (3.16) by Newton–Raphson method.*

5. *Smooth the new parameter vectors by* $\alpha_{k+1} := \lambda\alpha_{k+1} + (1-\lambda)\alpha_k$ *and* $\beta_{k+1} := \lambda\beta_{k+1} + (1-\lambda)\beta_k$ *where* $0 \leq \lambda \leq 1$ *is a fixed smoothing parameter.*

6. *If* $\hat{\gamma}_{k+1} = \gamma$ *then denote the final tilting parameter vectors* α_{k+1}, β_{k+1} *by* $\hat{\alpha}$, $\hat{\beta}$ *and go to step 7; otherwise set* $k = k+1$ *and repeat steps 2–6 until* $\hat{\gamma}_{k+1} = \gamma$ *is reached.*

7. *Generate a random sample* $X^{(1)}, \cdots, X^{(N)}$ *from the beta distribution with the final tilting parameter vectors* $\hat{\alpha}$, $\hat{\beta}$. *Estimate the value of rare event probability by the IS estimator:*
$$\hat{L} = \frac{1}{N}\sum_{i=1}^{N} H(\mathbf{X}^{(i)})W(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \hat{\alpha}, \hat{\beta}),$$
with estimated variance
$$\left(\frac{1}{N}\sum_{i=1}^{N} H(\mathbf{X}^{(i)})W^2(\mathbf{X}^{(i)}; \alpha^\star, \beta^\star, \hat{\alpha}, \hat{\beta}) - \hat{L}^2\right)/N.$$

Let us now assume that the activity durations $X^{(i)}, i = 1, \ldots, m$ are independent random variables distributed according to normal distribution with parameters $\mu_i^\star, \sigma_i^{2\star}, i = 1, \ldots, m$ and so their joint probability density function is given by

$$f\left(\mathbf{x}; \mu^\star, \sigma^{\mathbf{2}\star}\right) = \prod_{j=1}^{m} \frac{1}{\sqrt{2\pi\sigma_j^{2\star}}} e^{-\frac{1}{2\sigma_j^{2\star}}(x_j - \mu_j^\star)^2}, \tag{3.17}$$

where

$$-\infty < x_j < \infty, -\infty < \mu_j^\star < \infty, \sigma_j^{2\star} > 0,\ j = 1, \ldots, m.$$

The likelihood ratio involved in (3.8) is now given by

$$\begin{aligned} W(\mathbf{X}^{(i)}; \mu^\star, \sigma^{2\star}, \mu_k, \sigma_k^2) &= \frac{f\left(\mathbf{X}^{(i)}; \mu^\star, \sigma^{\mathbf{2}\star}\right)}{f\left(\mathbf{X}^{(i)}; \mu_k, \sigma_{\mathbf{k}}^{\mathbf{2}}\right)} \\ &= \prod_{j=1}^{m} \left(\frac{\sigma_{kj}}{\sigma_j^\star}\right) e^{-\frac{1}{2}\sum_{j=1}^{m}\left[\frac{1}{\sigma_{kj}^2}(x_j^{(i)} - \mu_{kj})^2 - \frac{1}{\sigma_j^{2\star}}(x_j^{(i)} - \mu_j^\star)^2\right]}. \end{aligned}$$

The logarithm of the probability density function $f(\mathbf{x};\mu,\sigma^{\mathbf{2}})$ is

$$\log f(\mathbf{x};\mu,\sigma^2) = -\frac{m}{2}\log(2\pi)) - \frac{1}{2}\sum_{j=1}^{m}\log\sigma_j^2 - \frac{1}{2\sigma_j^2}\sum_{j=1}^{m}(x_j-\mu_j)^2.$$

After partial derivation according to μ_j:

$$\frac{\partial \log f(\mathbf{x};\mu,\sigma^2)}{\partial \mu_j} = \frac{1}{\sigma_j^2}(x_j-\mu_j).$$

After partial derivation according to σ_j^2:

$$\frac{\partial \log f(\mathbf{x};\mu,\sigma^2)}{\partial \sigma_j^2} = -\frac{1}{2\sigma_j^2} + \frac{1}{2\sigma_j^4}(x_j-\mu_j)^2.$$

So equation (3.8) will now be specialized to:

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2)\left[\frac{1}{\sigma_j^2}(x_j-\mu_j)\right]=0, \qquad j=1,\ldots,m \tag{3.18}$$

and

$$\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2)\left[-\frac{1}{2\sigma_j^2}+\frac{1}{2\sigma_j^4}(x_j-\mu_j)^2\right]=0, \qquad j=1,\ldots,m. \tag{3.19}$$

From equations (3.18) and (3.19) we obtain the system of equations where

$$\mu_{k+1,j} = \frac{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2) X_j^{(i)}}{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2)}, \tag{3.20}$$

and

$$\sigma_{k+1,j}^2 = \frac{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2)(X_j^{(i)}-\hat{\mu}_j)}{\sum_{i=1}^{n} I_{\{S(\mathbf{X}^{(i)})>\hat{\gamma}_{k+1}\}} W(\mathbf{X}^{(i)};\mu^\star,\sigma^{2\star},\mu_k,\sigma_k^2)}. \tag{3.21}$$

From (3.20) and (3.21) we obtain $\hat{\mu}_j, \hat{\sigma}_j^2,\ j=1,\ldots,m$.

3.4. Algorithm *Basic CE Algorithm for Normal Distribution*

1. *Set the starting value* $\mu_0 = \mu^\star, \sigma_0^2 = \sigma^{\star 2}$. *Let* $k = 0$ *and start the iteration at Step 2.*

2. *Generate a random sample* $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(n)}$ *from normal distribution* $N(\mu_k, \sigma_k^2)$.

3. *Compute the shortest paths* $S^{(i)} = S(\mathbf{X}^{(i)})$ *for all* i *and order them from smallest to biggest,* $S^{(1)} \leq \cdots \leq S^{(n)}$. *Let* $\hat{\gamma}_{k+1}$ *be the* $(1-\rho)$ *sample quantile of shortest paths* $\hat{\gamma}_{k+1} = S^{[(1-\rho)n]}$, *provided this is less than* γ. *Otherwise set* $\hat{\gamma}_{k+1} = \gamma$.

4. *Use the same sample to estimate the components of the parameter vectors* μ_{k+1}, σ_{k+1}^2 *from equations (3.20), (3.21).*

5. *Smooth the new parameter vectors by* $\mu_{k+1} := \lambda\mu_{k+1} + (1-\lambda)\mu_k$ *and* $\sigma_{k+1}^2 := \lambda\sigma_{k+1}^2 + (1-\lambda)\sigma_k^2$, *where* $0 \leq \lambda \leq 1$ *is a fixed smoothing parameter.*

6. *If* $\hat{\gamma}_{k+1} = \gamma$ *then denote the final tilting parameter vectors* $\mu_{k+1}, \sigma_{k+1}^2$ *by* $\hat{\mu}$, $\hat{\sigma}^2$ *and go to step7; otherwise set* $k = k+1$ *and repeat steps 2–6 until* $\hat{\gamma}_{k+1} = \gamma$ *is reached.*

7. *Generate a random sample* $X^{(1)}, \cdots, X^{(N)}$ *from the normal distribution with the final tilting parameter vectors* $\hat{\mu}$, $\hat{\sigma}^2$. *Estimate the value of rare event probability by the IS estimator:*

$$\hat{L} = \frac{1}{N}\sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W(\mathbf{X}^{(i)}; \mu^\star, \sigma^{\star 2}, \hat{\mu}, \hat{\sigma}^2),$$

with estimated variance

$$\left(\frac{1}{N}\sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W^2(\mathbf{X}^{(i)}; \mu^\star, \sigma^{\star 2}, \hat{\mu}, \hat{\sigma}^2) - L^2\right)/N.$$

3.1.4 Two Modifications of the Basic CE Algorithm for Rare Event Simulation

The idea of the basic algorithm is to introduce two sequences $\{\hat{v}_k\}$and$\{\hat{\gamma}_k\}$ depending on the parameter ρ such that $\rho >> L$. We start by a not very small ρ. The basic algorithm aims to reach the optimal reference parameter $\mathbf{v}^*$ by a sequence of calculations controlled by the parameter ρ which is used to estimate the sequence of sample quantile $\{\hat{\gamma}(\mathbf{v}_k, \rho)\}$.

In fact may occur the condition $\hat{\gamma}(\mathbf{v}_k, \rho) > \hat{\gamma}(\mathbf{v}_{k-1}, \rho)$ will never be fulfilled. This means that $\hat{\gamma}(\mathbf{v}_k, \rho) < \hat{\gamma}(\mathbf{v}_{k-1}, \rho)$ and the convergence can not be reached in this case. To overcome this problem we propose a modification of the basic algorithm. If the estimated sample quantile $\hat{\gamma}(\mathbf{v}_k, \rho)$ fails to satisfy $\hat{\gamma}(\mathbf{v}_k, \rho) > \hat{\gamma}(\mathbf{v}_{k-1}, \rho)$, then we replace $\hat{\gamma}(\mathbf{v}_k, \rho)$ by $\hat{\gamma}(\mathbf{v}_{k-1}, \rho)$ and estimate the vector of reference parameter and if $\hat{\gamma}(\mathbf{v}_k, \rho) \geq \gamma$, then we set $\hat{\gamma}(\mathbf{v}_k, \rho) = \gamma$ and compute the estimator by IS, otherwise reiterate until convergence is reached. The modified algorithm for estimating the rare event probability can be written by the following:

3.5. Algorithm *Modified CE algorithm for rare event simulation*

1. *Initialize: Let* $k = 0, n =$ *sample size and the initial parameter* $\mathbf{v}_0$. *Generate a sample* $X^{(1)}, \cdots, X^{(n)}$*from the* $f(\mathbf{x}; \mathbf{v}_0)$.

2. *Compute the scores function* $S(X^{(i)})$ *for all* $i = 1, \cdots, n$. *Order all values of* S *from the smallest to largest* $S^{(1)} \leq \cdots \leq S^{(n)}$. *Compute the quantile* $\hat{\gamma}_0 = S^{[(1-\rho)n]}$. *Set* $k = 1$, *choose a starting value* $\mathbf{v}_0$ *and* $\mathbf{v}_0 = \mathbf{v}$.

3. *Generate a random sample* $X^{(1)}, \cdots, X^{(n)}$ *from* $f(.; \mathbf{v}_k)$. *Compute the* $(1-\rho)$*-quantile* $\hat{\gamma}_k = S^{[(1-\rho)n]}$.

4. *If* $\hat{\gamma}_k \geq \gamma$, *then set* $\hat{\gamma}_k = \gamma$*;otherwise if* $\hat{\gamma}_k < \hat{\gamma}_{k-1}$, *then set* $\hat{\gamma}_{k-1} = \hat{\gamma}_k$, *and estimate the vector* $\mathbf{v}_k$ *by solving the stochastic Program (3.8).*

5. *Smooth the new parameter vector by* $\mathbf{v}_k := \lambda \mathbf{v}_k + (1-\lambda)\mathbf{v}_{k-1}$, *where* $0 \leq \lambda \leq 1$ *is a fixed smoothing parameter.*

6. *If* $\hat{\gamma}_k = \gamma$ *then denote the final tilting parameter vector* $\mathbf{v}_k$ *by* $\hat{\mathbf{v}}$ *and go to step 7; otherwise set* $k = k + 1$ *and repeat steps 3–6 until* $\hat{\gamma}_k = \gamma$ *is reached.*

7. *Generate a random sample* $X^{(1)}, \ldots, X^{(N)}$ *with the final tilting parameter vector* $\hat{\mathbf{v}}$. *Estimate the value of rare event probability by the IS estimator:*

$$\hat{L} = \frac{1}{N} \sum_{i=1}^{N} H(\mathbf{X}^{(i)}) W(\mathbf{X}^{(i)}; \mathbf{v}^{\star}, \hat{\mathbf{v}})$$

with estimated variance

$$\left(\frac{1}{N}\sum_{i=1}^{N} H(\mathbf{X}^{(i)})W^2(\mathbf{X}^{(i)};\mathbf{v}^\star,\hat{\mathbf{v}}) - \hat{L}^2\right)/N.$$

The value of ρ used in the CE algorithm plays a crucial role in the convergence of the CE algorithm, we can only expect the CE algorithm converge to the correct values if ρ is sufficiently small. To determine a priority which ρ is acceptable can be a difficult task. To overcome this problem, we can change the value of ρ, adaptively (see Rubinstein ([93]), Homem-de Mello and Rubinstein ([56])). The modified version of Algorithm 3.1 is stated below. It requires the definition of constants ρ; $(0.01 \leq \rho \leq 0.1), \theta > 1.0, \delta > 0.0$ and $0.0 < \lambda \leq 1$.

3.6. Algorithm *Homem-de Mello and Rubinstein's CE algorithm for rare event simulation*

1. *Initialize: Let* $k = 0, \rho_0 = \rho$, n *=initial sample size and the initial parameter* $\alpha = \alpha_0, \beta = \beta_0$. *Generate a sample* $X^{(1)}, \cdots, X^{(n)}$ *from the* $f(\mathbf{x}; \alpha_0, \beta_0)$.

2. *Compute the scores function* $S(X^{(i)})$ *for all* $i = 1, \cdots, n$. *Order all values of* S *from the smallest to largest* $S^{(1)} \leq \cdots \leq S^{(n)}$. *Compute the sample quantile* $\hat{\gamma}_0 = S^{[(1-\rho_k)n]}$. *Set* $k = 1$.

3. *Use the same current sample to get* $\hat{\mathbf{v}}_k = \hat{\mathbf{v}}_k(\hat{\gamma}_{k-1})$ *from stochastic Program (3.8).*

4. *Generate a new sample* $X^{(1)}, \cdots, X^{(n)}$ *from the* $f\left(\mathbf{x}; \hat{\alpha}_k, \hat{\beta}_k\right)$ *and* $\rho_k = \rho$.

5. *Compute the* $(1-\rho)$*-quantile* $\hat{\gamma}_k = S^{[(1-\rho_k)n]}$.

6. *If* $\hat{\gamma}_k \geq \gamma$. *then set* $\hat{\gamma}_k = \gamma$, *and solve stochastic Program 3.8 to get* $\hat{\mathbf{v}}_K$. *Go to Step 8.*

7. *Otherwise, check whether there exists* $\bar{\rho}$ *such that*
   $S^{[(1-\bar{\rho})n]} \geq \min\{\gamma, \hat{\gamma}_{k-1} + \delta\}$.

   7.1 If $\bar{\rho} = \rho_k$, *then set* $k = k + 1$ *and reiterate from Step 3.*

7.2 If $\bar{\rho} < \rho_k$, then set $\rho_k = \bar{\rho}$ and go to Step 5.

7.3 If there exists no such $\bar{\rho}$, let $n = \theta n$ and go back to Step 4.

8. Estimate the rare event probability by IS:

$$\hat{L} = \frac{1}{n} \sum_{i=1}^{n} H(\mathbf{x}^{(i)}) W(\mathbf{x}^{(i)}; \alpha_0, \beta_0, \hat{\alpha}_K, \hat{\beta}_K).$$

3.1.5 Numerical Results

In this section we present numerical examples, illustrating different versions of CE algorithms for the estimation of rare event probabilities in small and medium sized stochastic networks. The activity duration times are supposed to be exponentially or beta distributed independent random variables. The efficiency of the CE estimators will be compared to the crude Monte Carlo estimator in the following way (see Hammersley and Handscombe ([55])).

$$\text{Efficiency} = \frac{\text{Variance} \times \text{Time (CMC)}}{\text{Variance} \times \text{Time (CE)}}$$

The computations were implemented in Fortran language on a personal computer with an Intel © Pentium 4© processor at 3.06 GHz.

3.1. Example *Small sized stochastic network with exponentially distributed activity durations times. Let us regard the stochastic network with number of nodes 4 and number of arcs 5, as it is given in Figure 6, respectively. Let the arcs be distributed according*

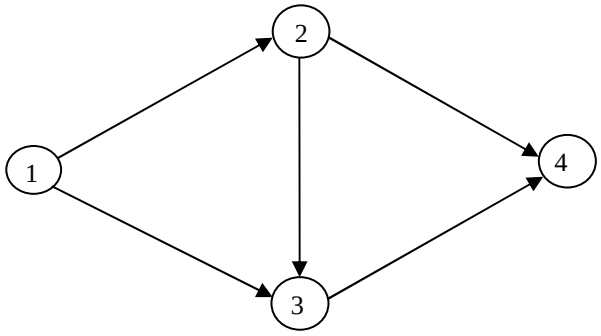

Figure 6: Small network.

to exponential distribution and the parameters of the exponential distributions be given in the following vector $\mathbf{v} = (0.350, 0.400, 0.310, 0.430, 0.420)$. *Suppose we want to estimate the probability that the length of the shortest path is greater than* $\gamma = 2$. *First we applied a Crude Monte Carlo simulation with sample size* $N = 1000000$. *As a result we got estimation* 1.9×10^{-5} *for the probability value with* 4.4×10^{-6} *estimated standard deviation. The necessary CPU time was* 3.66 *seconds. Then we used the three different CE algorithms with 12 different parameter settings to find the reference parameters to be applied in an IS variance reduction technique. In the CE part of the algorithms* $N = 1000$ *sample size while in the IS part of the algorithms* $N = 1000000$ *sample size was applied. In Table 12 the estimators, their standard deviations and calculated efficiency coefficients relative to the CMC simulation are given for the different CE algorithms and parameter settings.*

Table 12: Comparison of different versions and parameter settings of CE algorithms (small network, exponential distribution)

		Basic CE			*Modified CE*			*HM and R CE*		
ρ	λ	*Estimator*	*Std. dev.*	*Eff.*	*Estimator*	*Std. dev.*	*Eff.*	*Estimator*	*Std. dev.*	*Eff.*
0.10	0.1	1.6×10^{-5}	1.3×10^{-7}	913.7	1.6×10^{-5}	1.3×10^{-7}	887.9	1.6×10^{-5}	1.4×10^{-7}	**811.3**
0.10	0.5	1.6×10^{-5}	1.4×10^{-7}	896.7	1.6×10^{-5}	1.3×10^{-7}	**942.8**	1.6×10^{-5}	1.4×10^{-7}	797.1
0.10	0.7	1.6×10^{-5}	1.4×10^{-7}	856.1	1.6×10^{-5}	1.4×10^{-7}	848.1	1.6×10^{-5}	1.5×10^{-7}	732.8
0.10	1.0	1.6×10^{-5}	1.4×10^{-7}	862.3	1.6×10^{-5}	1.4×10^{-7}	900.6	1.6×10^{-5}	1.5×10^{-7}	765.4
0.05	0.1	1.6×10^{-5}	1.5×10^{-7}	787.1	1.6×10^{-5}	1.5×10^{-7}	767.0	1.6×10^{-5}	1.7×10^{-7}	573.1
0.05	0.5	1.6×10^{-5}	1.4×10^{-7}	901.8	1.6×10^{-5}	1.3×10^{-7}	909.6	1.6×10^{-5}	2.2×10^{-7}	329.8
0.05	0.7	1.6×10^{-5}	1.4×10^{-7}	905.5	1.6×10^{-5}	1.3×10^{-7}	905.4	1.6×10^{-5}	1.7×10^{-7}	554.3
0.05	1.0	1.6×10^{-5}	1.4×10^{-7}	834.1	1.6×10^{-5}	3.1×10^{-7}	764.9	1.6×10^{-5}	5.7×10^{-7}	577.7
0.01	0.1	1.6×10^{-5}	1.9×10^{-7}	473.3	1.6×10^{-5}	1.4×10^{-7}	806.9	1.5×10^{-5}	2.8×10^{-7}	214.5
0.01	0.5	1.6×10^{-5}	1.3×10^{-7}	**961.1**	1.6×10^{-5}	1.6×10^{-7}	644.5	1.5×10^{-5}	6.6×10^{-7}	39.3
0.01	0.7	1.6×10^{-5}	1.9×10^{-7}	465.6	1.6×10^{-5}	1.7×10^{-7}	554.9	1.6×10^{-5}	3.9×10^{-6}	1.1
0.01	1.0	1.8×10^{-5}	2.1×10^{-7}	4.16	1.6×10^{-5}	1.7×10^{-7}	593.0	1.1×10^{-5}	3.3×10^{-6}	1.6

3.2. Example *Medium sized stochastic network with exponentially distributed activity durations times.*

In this example the medium sized network in (Prékopa, Long, and Szántai, ([86])) was used. It has 28 nodes and 66 arcs. Nodes 1 and 28 are the original and terminal nodes,and it is given in Figure 7, respectively. The activities represented by the arcs

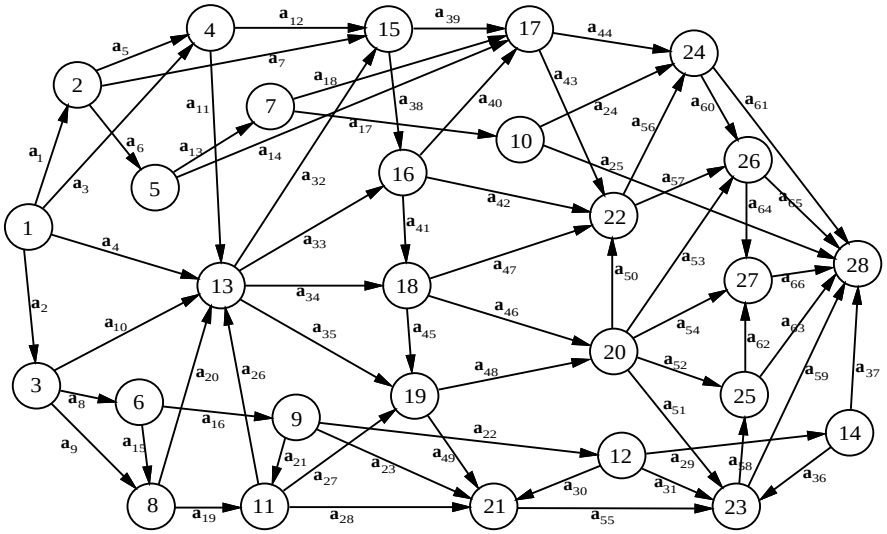

Figure 7: PERT network by A. Prékopa and J. Long

of the network are supposed to be distributed according to exponential distribution. The parameters of the exponential distributions were generated randomly according to uniform distribution in the interval $(1, 2)$. The Crude Monte Carlo simulation with sample size $N = 1000000$ resulted estimation 8.0×10^{-6} with 2.8×10^{-6} estimated standard deviation. The necessary CPU time was 38.56 seconds. Three different CE algorithms with 12 different parameter settings were used again to find the reference parameters to be applied in an IS variance reduction technique. In the CE part of the algorithms $N = 10000$ sample size while in the IS part of the algorithms $N = 1000000$ sample size was used. In Table 13 the estimators, their standard deviations and calculated efficiency coefficients relative to the CMC simulation are given for the different CE algorithms and parameter settings.

Table 13: Comparison of different versions and parameter settings of CE algorithms (medium network, exponential distribution)

		Basic CE			Modified CE			HM and R CE		
ρ	λ	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.
0.001	0.1	3.0×10^{-6}	2.5×10^{-7}	110.1	3.0×10^{-6}	2.6×10^{-7}	74.9	3.0×10^{-6}	3.8×10^{-7}	32.0
0.001	0.5	3.2×10^{-6}	5.7×10^{-7}	22.6	1.0×10^{-6}	2.1×10^{-7}	138.5	3.7×10^{-6}	6.3×10^{-7}	12.7
0.001	0.7	2.9×10^{-6}	1.4×10^{-6}	3.6	***[1]			4.3×10^{-6}	8.1×10^{-7}	7.8
0.001	1.0	5.4×10^{-6}	1.9×10^{-6}	2.06	5.6×10^{-7}	1.9×10^{-7}	**162.1**	4.0×10^{-6}	1.9×10^{-6}	1.4
0.005	0.1	3.7×10^{-6}	2.7×10^{-7}	85.3	3.1×10^{-6}	2.0×10^{-7}	125.3	2.9×10^{-6}	1.9×10^{-7}	**100.2**
0.005	0.5	2.9×10^{-6}	8.9×10^{-7}	9.0	3.2×10^{-6}	6.7×10^{-7}	12.1	3.4×10^{-6}	7.7×10^{-7}	8.2
0.005	0.7	2.1×10^{-6}	9.6×10^{-7}	8.5	4.9×10^{-7}	2.9×10^{-7}	83.4	3.0×10^{-6}	2.8×10^{-7}	68.0
0.005	1.0	4.5×10^{-7}	1.3×10^{-7}	**331.9**	3.1×10^{-6}	1.2×10^{-6}	3.4	2.4×10^{-6}	5.8×10^{-7}	14.2
0.010	0.1	3.1×10^{-6}	1.6×10^{-7}	205.2	2.7×10^{-6}	1.8×10^{-7}	115.5	3.2×10^{-6}	2.1×10^{-7}	67.8
0.010	0.5	2.4×10^{-6}	3.0×10^{-7}	79.5	3.1×10^{-6}	2.9×10^{-7}	58.4	3.1×10^{-6}	4.4×10^{-7}	23.6
0.010	0.7	1.5×10^{-6}	2.3×10^{-7}	121.1	***[1]			2.5×10^{-6}	7.4×10^{-7}	8.4
0.010	1.0	3.1×10^{-6}	1.6×10^{-7}	231.1	2.7×10^{-6}	1.8×10^{-7}	137.3	2.7×10^{-6}	2.1×10^{-7}	70.9

[1]The algorithm failed to give correct estimator.

3.3. Example *Small sized stochastic network with beta distributed activity durations times.*

In this example we use the same stochastic network as in Example 1. Now we assume that the activity durations are stochastically independent and have beta distribution. We suppose that the α parameters of the beta distributions are given in the vector $\alpha = (0.350, 0.400, 0.310,\ 0.430, 0.420)$ and the β parameters in the vector $\beta = (0.350, 0.440, 0.310,\ 0.430, 0.420)$. Suppose we want to estimate the probability that the length of the shortest path is greater than $\gamma = 1.99$. The Crude Monte Carlo simulation with sample size $N = 1000000$ gave the estimate 5.0×10^{-5} with 7.1×10^{-6} estimated standard deviation. The necessary CPU time was 10.06 seconds. Three different CE algorithms with 12 different parameter settings were used to find the reference parameters to be applied in an IS variance reduction technique. In the CE part of the algorithms $N = 1000$ sample size while in the IS part of the algorithms $N = 1000000$ sample size was used. In Table 14 the estimators, their standard deviations and calculated efficiency coefficients relative to the CMC simulation are given for the different CE algorithms and parameter settings.

Table 14: Comparison of different versions and parameter settings of CE algorithms (small network, beta distribution)

		Basic CE			Modified CE			HM and R CE		
ρ	λ	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.
0.10	0.1	4.3×10^{-5}	9.8×10^{-8}	4015.9	4.3×10^{-5}	1.1×10^{-7}	3058.4	4.3×10^{-5}	1.6×10^{-7}	1384.0
0.10	0.5	4.3×10^{-5}	1.1×10^{-7}	3442.4	4.2×10^{-5}	1.0×10^{-7}	3956.0	4.3×10^{-5}	3.3×10^{-7}	360.4
0.10	0.7	4.2×10^{-5}	4.8×10^{-8}	**16970.7**	4.2×10^{-5}	5.3×10^{-8}	**14029.3**	4.2×10^{-5}	1.5×10^{-7}	**1597.2**
0.10	1.0	4.3×10^{-5}	1.1×10^{-7}	3445.1	4.2×10^{-5}	7.9×10^{-8}	6142.9	4.3×10^{-5}	9.4×10^{-7}	242.4
0.05	0.1	4.3×10^{-5}	1.1×10^{-7}	2919.1	4.3×10^{-5}	1.1×10^{-7}	3207.8	4.3×10^{-5}	2.4×10^{-7}	671.6
0.05	0.5	4.2×10^{-5}	1.1×10^{-7}	2986.8	4.3×10^{-5}	1.8×10^{-7}	1186.5	4.2×10^{-5}	5.1×10^{-7}	156.5
0.05	0.7	4.2×10^{-5}	8.6×10^{-8}	5352.1	4.3×10^{-5}	6.3×10^{-8}	9618.9	4.2×10^{-5}	3.1×10^{-7}	428.4
0.05	1.0	4.2×10^{-5}	5.9×10^{-8}	11345.9	4.2×10^{-5}	2.0×10^{-7}	967.8	4.2×10^{-5}	1.0×10^{-7}	37.5
0.01	0.1	4.3×10^{-5}	2.2×10^{-7}	753.8	4.3×10^{-5}	1.7×10^{-7}	1318.5	4.1×10^{-5}	1.8×10^{-6}	11.8
0.01	0.5	4.2×10^{-5}	1.2×10^{-7}	2678.7	3.8×10^{-5}	5.8×10^{-7}	126.7	4.1×10^{-5}	1.5×10^{-6}	16.1
0.01	0.7	4.2×10^{-5}	1.2×10^{-7}	2607.8	4.2×10^{-5}	7.9×10^{-7}	63.5	1.9×10^{-5}	3.1×10^{-6}	4.3
0.01	1.0	4.2×10^{-5}	1.9×10^{-7}	7127.4	4.2×10^{-5}	1.9×10^{-7}	7193.9	4.2×10^{-5}	2.0×10^{-6}	65.3

For this example we show the iteration steps for the first phase of the basic CE algorithm, when the parameters are $\rho = 0.01$ and $\lambda = 0.1$. The changing values of the α and β parameters and the converging $(1-\rho)$-quantile values are given in Table 15.

Table 15: Iteration steps for the first phase of the basic CE algorithm ($\rho = 0.01$, $\lambda = 0.1$)

k	$\hat{\alpha}$					$\hat{\beta}$					$\hat{\gamma}_k$
0	0.35	0.40	0.31	0.43	0.42	0.35	0.44	0.31	0.43	0.42	
1	0.64	2.01	0.36	1.16	0.82	0.34	0.49	0.35	0.45	0.41	1.69
2	6.18	7.98	0.34	3.96	8.37	0.36	0.59	0.35	0.48	0.47	1.92
3	22.42	25.12	0.33	16.98	20.68	0.41	0.59	0.39	0.47	0.47	1.98
4	46.28	52.99	0.33	45.68	42.52	0.41	0.60	0.41	0.49	0.48	1.99

We also show the converging $(1-\rho)$-quantile values for the basic, modified and Homem-de Mello and Rubinstein's CE algorithms. The next three tables belong to different parameter settings.

Table 16: Convergence of the $\hat{\gamma}_t$, $t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.1, \lambda = 0.1$.

t	Basic CE	Modified CE	HM and R CE
1	1.174	1.197	1.396
2	1.356	1.432	1.599
3	1.554	1.575	1.716
4	1.739	1.713	1.834
5	1.803	1.833	1.889
6	1.881	1.881	1.919
7	1.933	1.927	1.952
8	1.953	1.954	1.969
9	1.971	1.969	1.980

Table 16: (continuation)

10	1.985	1.979	1.990
11	1.988	1.987	
12	1.990	1.990	

Table 17: Convergence of the $\hat{\gamma}_t,\ t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.05, \lambda = 0.7$.

t	Basic CE	Modified CE	HM and R CE
1	1.366	1.459	1.447
2	1.885	1.902	1.908
3	1.983	1.983	1.990
4	1.990	1.990	

Table 18: Convergence of the $\hat{\gamma}_t,\ t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.01, \lambda = 0.5$.

t	Basic CE	Modified CE	HM and R CE
1	1.689	1.754	1.690
2	1.975	1.974	1.990
3	1.990	1.990	

3.4. Example *Medium sized stochastic network with beta distributed activity durations times.*

In this example we use the same stochastic network as in Example 2. Now we assume that the activity durations are stochastically independent and have beta distribution. The activities represented by the arcs of the network are supposed to be distributed according to beta distribution. The parameters of the beta distributions were generated randomly according to uniform distribution in the interval $(1, 2)$. Suppose we want to

estimate the probability that the length of the shortest path is greater than $\gamma = 3.00$. The Crude Monte Carlo simulation with sample size $N = 1000000$ resulted estimation 1.0×10^{-5} with 3.2×10^{-6} estimated standard deviation. The necessary CPU time was 129.24 seconds. Three different CE algorithms with 12 different parameter settings were used again to find the reference parameters to be applied in an IS variance reduction technique. In the CE part of the algorithms $N = 10000$ sample size while in the IS part of the algorithms $N = 1000000$ sample size was used. In Table 19 the estimators, their standard deviations and calculated efficiency coefficients relative to the CMC simulation are given for the different CE algorithms and parameter settings.

Table 19: Comparison of different versions and parameter settings of CE algorithms (medium network, beta distribution)

		Basic CE			Modified CE			HM and R CE		
ρ	λ	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.	Estimator	Std. dev.	Eff.
0.10	0.1	7.1×10^{-6}	1.1×10^{-7}	348.6	7.5×10^{-6}	1.9×10^{-7}	112.3	7.1×10^{-6}	9.7×10^{-8}	368.9
0.10	0.5	1.1×10^{-5}	4.5×10^{-6}	0.38	7.1×10^{-6}	1.6×10^{-7}	219.0	7.3×10^{-6}	1.5×10^{-7}	249.5
0.10	0.7	5.8×10^{-6}	4.8×10^{-7}	39.9	7.6×10^{-6}	5.9×10^{-7}	21.5	7.7×10^{-6}	4.8×10^{-7}	32.1
0.10	1.0	2.1×10^{-6}	6.6×10^{-7}	16.9	6.3×10^{-6}	2.5×10^{-7}	97.4	6.7×10^{-6}	1.2×10^{-7}	342.4
0.05	0.1	7.4×10^{-6}	1.7×10^{-7}	255.6	7.3×10^{-6}	1.1×10^{-7}	**509.2**	7.4×10^{-6}	1.3×10^{-7}	335.8
0.05	0.5	7.3×10^{-6}	2.5×10^{-7}	125.1	7.4×10^{-6}	2.1×10^{-7}	155.1	7.4×10^{-6}	1.2×10^{-7}	**387.6**
0.05	0.7	7.5×10^{-6}	1.6×10^{-7}	307.0	7.1×10^{-6}	1.7×10^{-7}	218.7	7.4×10^{-6}	1.7×10^{-7}	205.1
0.05	1.0	7.2×10^{-6}	3.2×10^{-7}	76.8	7.4×10^{-6}	1.9×10^{-7}	188.2	7.4×10^{-6}	1.5×10^{-7}	272.8
0.01	0.1	7.3×10^{-6}	1.2×10^{-7}	**484.3**	7.4×10^{-6}	1.4×10^{-7}	312.4	7.2×10^{-6}	1.5×10^{-7}	241.6
0.01	0.5	7.1×10^{-6}	1.9×10^{-7}	219.3	6.7×10^{-6}	2.1×10^{-7}	153.2	7.4×10^{-5}	3.5×10^{-7}	49.7
0.01	0.7	7.8×10^{-6}	1.1×10^{-6}	6.0	7.4×10^{-6}	4.6×10^{-7}	33.3	6.8×10^{-6}	3.5×10^{-7}	55.5
0.01	1.0	6.7×10^{-6}	6.3×10^{-7}	20.8	7.7×10^{-6}	9.3×10^{-7}	7.7	6.6×10^{-6}	4.1×10^{-7}	38.6

For this example we can show the converging $(1-\rho)$-quantile values only for the basic, modified and Homem-de Mello and Rubinstein's CE algorithms. The next three tables belong to different parameter settings.

Table 20: Convergence of the $\hat{\gamma}_t,\ t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.05, \lambda = 0.7$.

t	Basic CE	Modified CE	HM and R CE
1	2.179	2.193	2.186
2	2.191	2.506	2.484
3	2.508	2.718	2.696
4	2.713	2.853	2.865
5	2.863	2.993	3.000
6	3.000	3.000	

Table 21: Convergence of the $\hat{\gamma}_t,\ t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.1, \lambda = 0.1$.

t	Basic CE	Modified CE	HM and R CE
1	2.403	2.413	2.469
2	2.489	2.479	2.539
3	2.537	2.575	2.586
4	2.613	2.619	2.635
5	2.664	2.653	2.692
6	2.720	2.709	2.745
7	2.774	2.767	2.798
8	2.805	2.834	2.863
9	2.856	2.867	2.903
10	2.900	2.925	2.944
11	2.953	2.990	3.000
12	3.000	3.000	

Table 22: Convergence of the $\hat{\gamma}_t,\ t = 1, 2, \ldots$ values for different versions of CE, case $\rho = 0.01, \lambda = 0.5$.

t	Basic CE	Modified CE	HM and R CE
1	2.402	2.411	2.391
2	2.689	2.715	2.701
3	2.937	2.917	3.000
4	3.000	3.000	

The numerical results show that an appropriate parameter setting of the CE algorithm may result significant improvement in the efficiency of the algorithm. Unfortunately it is not easy to give any general advise according to the best or near best possible parameter setting in any of the described three different CE algorithms. Our experiments show that producing a $\hat{\gamma}_t$ sequence reaching the γ value too quickly is generally not a good idea as may result a reference parameter set with relatively poor variance reduction in the IS phase of the algorithms.

3.2 Completion Time Estimation in the PERT Networks

Main drawback of the traditional PERT modeling is that the probabilistic characteristics determined for the finishing time of the project are only valid when it is supposed that any activity can be started promptly after finishing all of its predecessor activities. This is possible in the case of scheduling computer tasks, however it is impossible in the case of architectural project planning what is the most important application area of PERT modeling.

3.2.1 Introduction

A project is defined as the collection of *activities* (or *events*) $\{a, b, \ldots\}$ among which a precedence relation $a \prec b$ is defined. It is supposed to be transitive, *i.e.*, if $a \prec b$ and $b \prec c$ then $a \prec c$. Any project can be depicted as a directed network, where the directed

arcs represent the activities. Without restricting generality, we may assume that there is exactly one node such that no arc leads into it and there is exactly one node such that no arc goes out of it. These two nodes will be called original and terminal nodes, respectively.

Each activity has a *duration* (or *length*). The duration (or length) of a path is the sum of the durations of the arcs contained in the path.

Of particular importance are the paths connecting the original and terminal nodes. The maximum length of these paths is the shortest time needed to complete the project and we call it the project completion time. The corresponding path is the *critical path.*

Suppose that there are n arcs numbered by $1, 2, \ldots, n$. Suppose furthermore that there are p paths, numbered by $1, 2, \ldots, p$, which connect the original and terminal nodes. The elements of the path-arc incidence matrix $A = (a_{ij})$ are defined as

$$a_{ij} = \begin{cases} 1, \text{ if activity } j \text{ is contained in path } i \\ 0, \text{ otherwise.} \end{cases}$$

We will designate by A_i the ith row of the matrix A $(1 \leq i \leq p)$. Let $\xi = (\xi_1, \ldots, \xi_n)^T$ be the vector of the activity durations. Then the critical path length $R(\xi)$ equals

$$R(\xi) = \max_{1 \leq i \leq p} A_i \xi.$$

Designating by $P_1, \ldots, P_p$ the paths from the origin to the terminal nodes, we may also write

$$R(\xi) = \max_{1 \leq i \leq p} \sum_{j \in P_i} \xi_j.$$

If the durations $\xi_1, \ldots, \xi_n$ are random variables then $R(\xi)$ is a random variable, too. Its probability distribution function (cdf) will be designated by $F(x)$, i.e.,

$$F(x) = P(R(\xi) \leq x). \tag{3.22}$$

The original PERT technique, developed by Malcolm et al. ([73]), is a technique to approximate the expected duration of the project. Further approximations and bounds to this value are due to Fulkerson ([35]), Clingen ([18]), Elmaghraby ([26]), Robillard and Trahan ([89], [90]), Devroye ([20]) and others.

Even more important is, from the point of view of applications, to bound or approximate the probability distribution function of the critical path. In connection with this we mention the works by Kleindorfer ([68]), Shogan ([102]), Nádas ([78]), Meilijson and Nádas ([74]), Dodin ([22]), Weiss([118]), Monhor ([76]), Wallace ([117]) and (Prékopa, Long, and Szántai,([86])).

3.2.2 Singular normal distribution for PERT modeling

This section is based on the work Prékopa, Long, and Szántai, ([86]). The probability distribution of an activity duration ξ_i is frequently assumed to be the distribution of a random variable of the form $\xi_i = L_i + (U_i - L_i)\eta_i$, where L_i and U_i $(L_i < U_i)$ are called optimistic and pessimistic estimates, respectively, made by experts and η_i has beta distribution in the interval $(0, 1)$, $\quad i = 1, \ldots, n$. In addition to L_i and U_i, a third estimate M_i is also made which is the most likely duration of the activity i (the value maximizing the probability density function of $L_i + (U_i - L_i)\eta_i$). Given these three estimates, we obtain (see Battersby([5]), Littlefield and Randolph ([71]):

$$E(\xi_i) = \frac{L_i + 4M_i + U_i}{6}, \quad i = 1, \ldots, n \tag{3.23}$$

$$\mathrm{Var}(\xi_i) = \frac{(U_i - L_i)^2}{36}, \quad i = 1, \ldots, n. \tag{3.24}$$

If the non-eliminated paths contain a large number of arcs, then, whether or not the individual activities have the above "transformed beta distributions" or some other ones, is not decisive because the path lengths are approximately normally distributed by the central limit theorem. Formulas (3.23) and (3.24) can be used to obtain the expectation and the variance of each path.

Let q be the number of non-eliminated paths and suppose, for the sake of simplicity, that those are $P_1, \ldots, P_q$. In what follows we approximate not only the univariate marginal distributions of the random vector

$$(A_1\xi, \ldots, A_q\xi)^T,$$

by normal distributions, but also the joint distribution of the components by a multivariate normal distribution. (For theoretical foundation of such an approximation see Fomin ([33]).) If $\xi_1, \ldots, \xi_n$ are stochastically independent, then all characteristics (the expectation vector and the covariance matrix) are determined by the quantities $\mu_i = E(\xi_i)$, $\sigma_i^2 = \text{Var}(\xi_i)$, $i = 1, \ldots, n$. If, however, $\xi_1, \ldots, \xi_n$ are stochastically dependent, then, in addition to the above mentioned expectations and variances, we need to know the covariance matrix of the random vector $(\xi_1, \ldots, \xi_n)$. Designating by $\widetilde{A}$ the matrix consisting of the first q rows of A, the random vector (3.23) equals $\widetilde{A}\xi$. We have that $E\left(\widetilde{A}\xi\right) = \widetilde{A}\mu$, $\mu = (\mu_1, \ldots, \mu_n)^T$and the covariance matrix of $\widetilde{A}\xi$ equals $\widetilde{A}C\widetilde{A}^T$, where C is the covariance matrix of the random vector $\xi = (\xi_1, \ldots, \xi_n)^T$.

Since the (random) length of P_i equals $A_i\xi$, we have the relations

$$\begin{aligned} F(x) &= P\left(\max_{1\le i\le q} A_i\xi \le x\right) = P\left(A_1\xi \le x, \ldots, A_q\xi \le x\right) \\ &= P\left(\frac{A_1\xi - A_1\mu}{\sqrt{A_1CA_1^T}} \le \frac{x - A_1\mu}{\sqrt{A_1CA_1^T}}, \ldots, \frac{A_q\xi - A_q\mu}{\sqrt{A_qCA_q^T}} \le \frac{x - A_q\mu}{\sqrt{A_qCA_q^T}}\right), \end{aligned} \tag{3.25}$$

where the random variables

$$\frac{A_i\xi - A_i\mu}{\sqrt{A_iCA_i^T}}, \qquad i = 1, \ldots, q.$$

have standard normal probability distributions and their correlation matrix can easily be obtained from the covariance matrix $\widetilde{A}C\widetilde{A}^T$. This way the probability distribution function $F(x)$ of the critical path length $R(\xi)$ can be bounded and approximated by the techniques developed for bounding and approximating the value of the multivariate standard normal probability distribution function value (see ([86]).

Remark 1. The $0-1$ matrix $\widetilde{A}$ may have many columns with all zero elements. These columns and the corresponding components of the random vector $\xi = (\xi_1, \ldots, \xi_n)^T$ can be deleted, they do not play any role in the probability distribution of the critical path length. The number of remaining components of ξ can be smaller than the number of non-eliminated paths, so the multivariate normal probability distribution of the random variables (3.5) can be singular. Of course singularity can be caused also by the linear dependence of the rows of the matrix $\widetilde{A}$.

3.2.3 New Stochastic Programming Approach to PERT

Let be described the project by the $(\mathcal{N}, \mathcal{A})$ directed graph, which doesn't contain any loop. Here $\mathcal{N}$ is the set of nodes (events) and $\mathcal{A}$ is the set of arcs (activities). Let be designated by $c_j, j = 1, \ldots, n$ the nodes in the set $\mathcal{N}$, among which c_1 let be the original and c_n let be the terminal node. Let be assigned the variable x_j to the node c_j representing the earliest starting time for all activities starting from the node c_j, $j = 1, \ldots, n$. Let be designated by $e_i, i = 1, \ldots, m$ the arcs in the set $\mathcal{A}$ and let be assigned the number d_i to the arc e_i as the duration time of the represented activity. If these are deterministic numbers, then the shortest execution time of the whole project represented by the loopless directed graph $(\mathcal{N}, \mathcal{A})$ can be determined by solving the following linear programming problem:

$$\begin{gathered} x_{f_i} - x_{s_i} \geq d_i, i = 1, \ldots, m \\ x_j \geq 0, j = 1, \ldots, n \\ \min(x_n - x_1), \end{gathered} \tag{3.26}$$

where s_i, f_i are the indeces of the starting and finishing nodes of arc e_i. Obviously one can suppose, that $x_1 \equiv 0$ and the problem (3.26) can be simplified. If the activity duration times $d_i, i = 1, \ldots, m$ are random variables, then let be designated these by $\xi_i, i = 1, \ldots, m$ and let us solve the following jointly probabilistic constrained stochastic programming problem for finding the $x_j, j = 1, \ldots n$ earliest starting times:

$$\begin{gathered} P(x_{f_i} - x_{s_i} \geq \xi_i, i = 1, \ldots, m) \geq p \\ x_j \geq 0, j = 1, \ldots, n \\ \min(x_n - x_1), \end{gathered} \tag{3.27}$$

where p is a prescribed, large enough probability. If the activity starting times determined by the $x_1, \ldots, x_n$ variables according to the optimal solution of the optimization problem (3.27) are strictly applied then we can guarantee at reliability level p that the whole project can be executed without any conflict in the activity starting and finishing times.

In the literature of PERT the activity duration times are usually supposed to be independent. In these cases in the stochastic programming problem (3.27) the joint probability can easily be calculated by taking the product of the probabilities calculated

from the one dimensional marginal probability distributions. These problems are easy to solve from numerical point of view.

In the first problem of the next section we will show that the stochastic programming problem (3.27) can also be solved if we suppose the random activity duration times to be Dirichlet distributed. In the second problem the duration times are supposed to be independent normally distributed and the model will be solved both by the original PERT optimization technique as it is described in the paper [86] and by the solution of the stochastic programming problem (3.27). The numerical results will be compared.

3.2.4 Numerical results

Let us regard first the PERT problem given by the loopless, directed graph of Figure 8. In the Figure 8. $d_i, i = 1, \ldots, 15$ denotes the duration times of 15 activities and x_j denotes the earliest starting times for all activities starting at event j, $j = 1, \ldots, 8$. The event No. 1 is the original and the event No. 8 is the terminal event and we suppose that $x_1 = 0$. Now, if the activity duration times $d_i, i = 1, \ldots, 15$ are given deterministically, then the PERT model can be regarded as a CPM (*Critical Path Method*) problem and we have to solve the problem (3.26) according to the linear programming problem (3.28). The solution component x_8 gives the total execution time of the project and the solution components x_j, $j = 2, \ldots, 8$ give the earliest starting times for the appropriate activities. All questions of the CPM model can be answered from these results. For determining the critical path see ([66]).

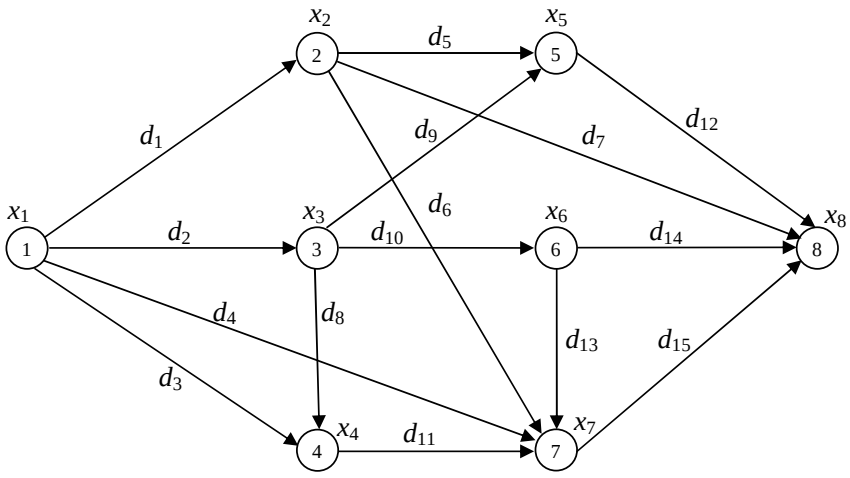

Figure 8: PERT network

$$
\begin{array}{cccccccc}
x_2 & & & & & & \geq d_1 \\
& x_3 & & & & & \geq d_2 \\
& & x_4 & & & & \geq d_3 \\
& & & & x_7 & & \geq d_4 \\
-x_2 & & & +x_5 & & & \geq d_5 \\
-x_2 & & & & +x_7 & & \geq d_6 \\
-x_2 & & & & & +x_8 & \geq d_7 \\
& -x_3 & +x_4 & & & & \geq d_8 \\
& -x_3 & & +x_5 & & & \geq d_9 \\
& -x_3 & & & +x_6 & & \geq d_{10} \\
& & -x_4 & & +x_7 & & \geq d_{11} \\
& & & -x_5 & & +x_8 & \geq d_{12} \\
& & & & -x_6 & +x_7 & \geq d_{13} \\
& & & & -x_6 & +x_8 & \geq d_{14} \\
& & & & & -x_7 \quad +x_8 & \geq d_{15} \\
\end{array} \tag{3.28}
$$

$$x_i \geq 0, i = 2, \ldots, 8$$

$$\min(\qquad\qquad\qquad\qquad\qquad\qquad x_8)$$

If however the activity duration times $d_i, i = 1, \ldots, 15$ are stochastic, then we have to

solve the problem (3.29) according to the stochastic programming problem (3.27). For the definition of this problem let be

$$\xi_i = a_i + (b_i - a_i)\eta_i, i = 1, \ldots, 15,$$

where a_i, b_i are the optimistic and the pessimistic estimators of the duration times of the ith activity, $i = 1, \ldots, 15$ and the random variables $\eta_1, \ldots, \eta_{15}$ are Dirichlet distributed with parameters $\vartheta_1 > 0, \ldots, \vartheta_{15} > 0, \vartheta_{16} > 0$. Their joint probability density function is

$$f(x_1, \ldots, x_{15}) = \frac{\Gamma(\vartheta_1 + \ldots + \vartheta_{15} + \vartheta_{16})}{\Gamma(\vartheta_1) \cdots \Gamma(\vartheta_{15})\Gamma(\vartheta_{16})} x_1^{\vartheta_1 - 1} \cdots x_{15}^{\vartheta_{15} - 1} (1 - x_1 - \ldots - x_{15})^{\vartheta_{16} - 1},$$

if $x_1 \geq 0, \ldots, x_{15} \geq 0$ and $x_1 + \ldots + x_{15} \leq 1$. So if we take the optimistic and pessimistic estimators for the duration times of activities and the set of parameters $\vartheta_i, i = 1, \ldots, 16$, as they are given in Table 23, then the most likely values, expected values and standard deviations of the activity duration times can be calculated in the following way. As η_i has beta marginal distribution with parameters $\vartheta_i, \overline{\vartheta}_i = \sum_{i=1, j \neq i}^{16} \vartheta_j$ the most likely value of ξ_i is given by

$$m_i = a_i + (b_i - a_i)\frac{\vartheta_i - 1}{\vartheta_i + \overline{\vartheta}_i - 2},$$

its expected value is

$$E(\xi_i) = \quad a_i + (b_i - a_i)\frac{\vartheta_i}{\vartheta_i + \overline{\vartheta}_i},$$

and its standard deviation is

$$\mathrm{D}(\xi_i) = b_i - a_i)\sqrt{\frac{\vartheta_i \overline{\vartheta}_i}{(\vartheta_i + \bar{\vartheta}_i)^2(\vartheta_i + \bar{\vartheta}_i + 1)}}.$$

We remark that in the above formula $\vartheta_i + \overline{\vartheta}_i = \sum_{j=1}^{16} \vartheta_j$, always. In Table 23 the calculated most likely values, expected values and standard observations are also given.

Now the joint probabilistic constrained stochastic programming problem is the fol-

lowing:

$$P\left(\begin{array}{llllllll}
\frac{1}{(b_1-a_1)}(& -a_1 & +x_2 & & & & &) \geq \eta_1 \\
\frac{1}{(b_2-a_2)}(& -a_2 & & +x_3 & & & &) \geq \eta_2 \\
\frac{1}{(b_3-a_3)}(& -a_3 & & & +x_4 & & &) \geq \eta_3 \\
\frac{1}{(b_4-a_4)}(& -a_4 & & & & & +x_7 &) \geq \eta_4 \\
\frac{1}{(b_5-a_5)}(& -a_5 & -x_2 & & & +x_5 & &) \geq \eta_5 \\
\frac{1}{(b_6-a_6)}(& -a_6 & -x_2 & & & & +x_7 &) \geq \eta_6 \\
\frac{1}{(b_7-a_7)}(& -a_7 & -x_2 & & & & & +x_8) \geq \eta_7 \\
\frac{1}{(b_8-a_8)}(& -a_8 & & -x_3 & +x_4 & & &) \geq \eta_8 \\
\frac{1}{(b_9-a_9)}(& -a_9 & & -x_3 & & +x_5 & &) \geq \eta_9 \\
\frac{1}{(b_{10}-a_{10})}(& -a_{10} & & -x_3 & & & +x_6 &) \geq \eta_{10} \\
\frac{1}{(b_{11}-a_{11})}(& -a_{11} & & & -x_4 & & +x_7 &) \geq \eta_{11} \\
\frac{1}{(b_{12}-a_{12})}(& -a_{12} & & & & -x_5 & & +x_8) \geq \eta_{12} \\
\frac{1}{(b_{13}-a_{13})}(& -a_{13} & & & & & -x_6 \quad +x_7 &) \geq \eta_{13} \\
\frac{1}{(b_{14}-a_{14})}(& -a_{14} & & & & & -x_6 & +x_8) \geq \eta_{14} \\
\frac{1}{(b_{15}-a_{15})}(& -a_{15} & & & & & -x_7 & +x_8) \geq \eta_{15}
\end{array}\right) \geq p \qquad (3.29)$$

$$x_i \geq 0, i = 2, \ldots, 8$$

$$\min x_8,$$

where p is the prescribed probability of finishing the project for the prescribed due date. In the table 23. the parameters of the Dirichlet distribution are given. The correlation coefficients between different pairs of activity duration times can be determined from these parameter values. We don't do it here just remark that the values of the correlation coefficients are between 0 and $-0,124409$.

Table 24. contains the solutions of the linear programming problem (3.28) for those cases, when the optimistic, the pessimistic and the most likely activity duration times are applied as deterministic values. There are given in the same table the solutions of the stochastic programming problem (3.29) for three different probability levels: 0.9, 0.95 and 0.99. The parameters of the Dirichlet distribution were taken from the Table 23. In Table 24 the value of the variable x_8 means also the completion time of the project. It can be seen that the deterministic cases do not provide appropriate results. If we

Table 23: Parameters of the Dirichlet distribution

No.	optimistic estimation	pessimistic estimation	most likely value	θ parameter	expected value	standard deviation
1	70	130	70.171	1.06	72.765	2.568
2	10	40	11.500	2.05	12.674	1.745
3	50	75	50.048	1.04	51.130	1.060
4	10	40	12.957	3.07	14.004	2.083
5	100	125	100.238	1.20	101.304	1.135
6	70	95	70.071	1.06	71.152	1.070
7	40	75	40.133	1.08	41.643	1.511
8	85	95	85.524	2.10	85.913	0.588
9	10	35	10.060	1.05	11.141	1.065
10	105	135	105.007	1.005	106.311	1.252
11	45	90	47.175	2.015	48.942	2.597
12	25	45	25.067	1.07	25.930	0.860
13	25	50	25.060	1.05	26.141	1.065
14	15	35	16.048	2.10	16.826	1.176
15	55	75	55.029	1.03	55.896	0.844
16				1.02		

Table 24: Solutions of the linear and stochastic programming problems

	deterministic cases			stochastic cases		
variables	optimistic	pessimistic	most likely	$p = 0.90$	$p = 0.95$	$p = 0.99$
x_1	0	0	0	0	0	0
x_2	70	130	70.171	79.695	82.784	96.084
x_3	10	40	11.500	15.291	24.443	27.131
x_4	95	135	97.024	104.492	114.908	117.901
x_5	170	255	170.410	177.479	181.694	193.509
x_6	115	175	116.507	126.897	138.993	148.067
x_7	140	225	144.199	162.424	181.097	184.299
x_8	195	300	199.227	226.194	247.067	250.336

work with the optimistic or with the most likely activity duration times then the project will be finished in the given time with less then 0.9 probability only. When working with the pessimistic activity duration times then the completion time of the project will be too large (300) although the reliability level is also high (0.99). The decision maker can choose from the stochastic versions according to his acceptable reliability level of completing the whole project for the calculated finishing time. We belive this choice will be easier for him than the choice between the deterministic versions. The solution of the stochastic programming problems may happen by the application of the code PCSP (*Probabilistic Constrained Stochastic Programming*) by T. Szántai (see [108], and by the versions of the code which can handle Dirichlet and multivariate gamma distributed right hand sides,

As a second numerical example let us regard the network of the Figure 7 what was applied in the paper by J. Long, A. Prékopa and T. Szántai (see [86]). This network consists of 66 activities and 28 events, the original event is the first one and the terminal event is the event No. 28. The optimistic and pessimistic estimates of the activity duration times are given in Table 25.

Table 25: Lower and upper bounds for the duration times of 66 activities

No.	Activity	Lower bound	Upper bound	No.	Activity	Lower bound	Upper bound
1	(1, 2)	24	32	34	(13,18)	47	55
2	(1, 3)	48	56	35	(13,19)	44	52
3	(1, 4)	49	57	36	(14,23)	11	19
4	(1,13)	24	32	37	(14,28)	36	44
5	(2, 4)	21	29	38	(15,16)	39	47
6	(2, 5)	43	51	39	(15,17)	18	26
7	(2,15)	30	38	40	(16,17)	13	21
8	(3, 6)	14	22	41	(16,18)	41	49
9	(3, 8)	28	36	42	(16,22)	42	50
10	(3,13)	29	37	43	(17,22)	38	46
11	(4,13)	36	44	44	(17,24)	27	35
12	(4,15)	19	27	45	(18,19)	26	34
13	(5, 7)	49	57	46	(18,20)	39	47
14	(5,17)	12	20	47	(18,22)	25	33
15	(6, 8)	35	43	48	(19,20)	13	21
16	(6, 9)	28	36	49	(19,21)	16	24
17	(7,10)	15	23	50	(20,22)	29	37
18	(7,17)	26	34	51	(20,23)	42	50
19	(8,11)	33	41	52	(20,25)	33	41
20	(8,13)	46	54	53	(20,26)	43	51
21	(9,11)	41	49	54	(20,27)	44	52
22	(9,12)	47	55	55	(21,23)	22	30
23	(9,21)	42	50	56	(22,24)	46	54
24	(10,24)	40	48	57	(22,26)	19	27
25	(10,28)	37	45	58	(23,25)	33	41
26	(11,13)	27	35	59	(23,28)	39	47
27	(11,19)	26	34	60	(24,26)	15	23
28	(11,21)	31	39	61	(24,28)	48	56
29	(12,14)	38	46	62	(25,27)	27	35
30	(12,21)	48	56	63	(25,28)	26	34
31	(12,23)	29	37	64	(26,27)	29	37

Table 25: (Continuation)

No.	Activity	Lower bound	Upper bound	No.	Activity	Lower bound	Upper bound
32	(13,15)	32	40	65	(26,28)	22	30
33	(13,16)	20	28	66	(27,28)	20	28

In the PERT network of Figure 7. there exist 1623 paths from the original to the terminal node. With the lower and upper bounds on duration times of the activities given in Table 25. the number of the remaining paths after the application of the first path elimination algorithm described in the paper [86] is 201, while after subsequent application of the second path elimination algorithm only 8 paths will remain as paths ever may become critical. As in these 8 paths only 21 activities are involved, the path–arc incidence matrix reduced to these paths only, has a size of 8×21 and it is given in Table 26.

Table 26: The path–arc incidence matrix of the remained 8 paths

	2	8	15	16	19	21	26	32	38	41	45	46	48	50	51	56	58	60	62	64	66
1	1	1	0	1	0	1	1	1	1	1	1	0	1	1	0	1	0	1	0	1	1
2	1	1	1	0	1	0	1	1	1	1	1	0	1	1	0	1	0	1	0	1	1
3	1	1	0	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1
4	1	1	1	0	1	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1
5	1	1	0	1	0	1	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1
6	1	1	1	0	1	0	1	1	1	1	1	0	1	0	1	0	1	0	1	0	1
7	1	1	0	1	0	1	1	1	1	1	0	1	0	0	1	0	1	0	1	0	1
8	1	1	1	0	1	0	1	1	1	1	0	1	0	0	1	0	1	0	1	0	1

This matrix has only 4 linearly independent column vectors, so the 8–variate normal probability distribution is restricted to a 4–dimensional subspace, i.e. the distribution is singular. Let us compare the results of the multivariate normal approach published in paper [86] with the results of the new stochastic programming based approach. Suppose

Table 27: Parameters of the multivariate normal probability distribution of remaining 8 paths

Expected value	508	507	504	503	487	486	483	482
Variance	8.9443	8.9443	8.6410	8.6410	8.6410	8.6410	8.267	8.3267
Correlation matrix	1.0000	0.8667	0.8971	0.7591	0.7591	0.6211	0.6445	0.5013
	0.8667	1.0000	0.7591	0.8971	0.6211	0.7591	0.5013	0.6445
	0.8971	0.7591	1.000	0.8571	0.6429	0.5000	0.7412	0.5930
	0.7591	0.8971	0.8571	1.0000	0.5000	0.6429	0.5930	0.7412
	0.7591	0.6211	0.6429	0.5000	1.0000	0.8571	0.8895	0.7412
	0.6211	0.7591	0.5000	0.6429	0.8571	1.0000	0.7412	0.8895
	0.6445	0.5013	0.7412	0.5930	0.8895	0.7412	1.0000	0.8462
	0.5013	0.6445	0.5930	0.7412	0.7412	0.8895	0.8462	1.0000

the random duration times of activities to be independent with given expected values and variations but otherwise their probability distributions be arbitrary. The expected values are defined as the arithmetical mean values of their lower and upper bounds given in Table 25. The variances will be defined as the squares of the differences between the lower and upper bounds divided by twelve, i.e. as if the probability distributions of the activity duration times would be uniform between their lower and upper bounds. Now the lengths of the remaining 8 paths have a singular multivariate normal probability distribution, concentrated on a 4–dimensional subspace. The parameters of this multivariate normal probability distribution are given in Table 27.

For comparing the two different approaches first the stochastic programming problem (3.29) has been solved with different probability levels ranging from 0 to 1. This can be done easily by using the AMPL modeling language (see Fourer, R., Gay,D. M. and Kernighan,B. W. [34]) and the LOQO solver (see Vanderbei,R. J. [114] and [115]) as the random duration times of activities were supposed to be independent and normally distributed. The results are given in Table 28.

Table 28: The solutions of the stochastic programming problem for different probability levels

prob. level	execution time	prob. level	execution time	prob. level	execution time
0.01	531.358	0.35	560.794	0.70	577.641
0.05	540.927	0.40	563.087	0.75	580.677
0.10	546.329	0.45	565.351	0.80	584.153
0.15	550.112	0.50	567.624	0.85	588.333
0.20	553.206	0.55	569.945	0.90	593.796
0.25	555.927	0.60	572.353	0.95	602.312
0.30	558.427	0.65	574.897	0.99	619.592

The cummulative probability distribution function of the project completion time was calculated by linear interpolation in the interval 495–625 with unit steplength. The results of the multivariate normal approach are given in Table 29. In this table there are given not only the calculated values of the cummulative probability distribution function but its binomial moment and hypermultitree lower and upper bounds, too. The definition and calculation algorithms for these bounds were published in the paper [86] and they are designated by $BML3$, $BMU3$, resp. $HML(0,2)$, $HML(0,3)$, $HMU(1,1)$ and $HMU(1,2)$.

Table 29: Exact values and lower and upper bounds of the cdf of critical path length calculated by multivariate normal distribution

X	$BML3$	$HML(0,2)$	$HML(0,3)$	exact	$HMU(1,2)$	$HMU(1,1)$	$BMU3$
480	0.0000	0.0002	0.0003	0.0003	0.0003	0.0003	0.0070
481	0.0000	0.0003	0.0004	0.0004	0.0004	0.0005	0.0085
482	0.0000	0.0005	0.0006	0.0006	0.0006	0.0008	0.0102
483	0.0000	0.0007	0.0009	0.0009	0.0009	0.0012	0.0114

Table 29: (continuation)

X	$BML3$	$HML(0,2)$	$HML(0,3)$	exact	$HMU(1,2)$	$HMU(1,1)$	$BMU3$
484	0.0000	0.0011	0.0013	0.0013	0.0013	0.0017	0.0124
485	0.0000	0.0016	0.0019	0.0020	0.0020	0.0025	0.0138
486	0.0000	0.0024	0.0028	0.0028	0.0028	0.0037	0.0155
487	0.0000	0.0035	0.0040	0.0040	0.0040	0.0052	0.0178
488	0.0000	0.0050	0.0056	0.0057	0.0057	0.0073	0.0208
489	0.0000	0.0070	0.0078	0.0079	0.0079	0.0100	0.0248
490	0.0000	0.0097	0.0108	0.0108	0.0108	0.0137	0.0300
491	0.0000	0.0133	0.0146	0.0146	0.0146	0.0184	0.0366
492	0.0000	0.0180	0.0195	0.0195	0.0195	0.0244	0.0448
493	0.0000	0.0240	0.0258	0.0258	0.0258	0.0320	0.0551
494	0.0000	0.0316	0.0337	0.0337	0.0337	0.0415	0.0675
495	0.0000	0.0410	0.0434	0.0435	0.0435	0.0531	0.0824
496	0.0000	0.0526	0.0554	0.0554	0.0554	0.0670	0.0962
497	0.0000	0.0666	0.0697	0.0697	0.0697	0.0837	0.1125
498	0.0000	0.0833	0.0867	0.0867	0.0868	0.1032	0.1320
499	0.0000	0.1030	0.1067	0.1067	0.1067	0.1259	0.1549
500	0.0103	0.1258	0.1297	0.1297	0.1297	0.1518	0.1812
501	0.0487	0.1518	0.1560	0.1560	0.1560	0.1810	0.2109
502	0.0873	0.1812	0.1855	0.1855	0.1856	0.2134	0.2439
503	0.1270	0.2139	0.2183	0.2183	0.2183	0.2489	0.2800
504	0.1683	0.2497	0.2541	0.2541	0.2541	0.2873	0.3188
505	0.2179	0.2884	0.2928	0.2928	0.2928	0.3283	0.3600
506	0.2731	0.3296	0.3340	0.3340	0.3340	0.3714	0.4031
507	0.3270	0.3730	0.3772	0.3772	0.3772	0.4161	0.4476
508	0.3798	0.4180	0.4220	0.4220	0.4220	0.4619	0.4928
509	0.4318	0.4641	0.4678	0.4678	0.4679	0.5082	0.5334
510	0.4830	0.5106	0.5141	0.5141	0.5141	0.5543	0.5741
511	0.5330	0.5569	0.5601	0.5601	0.5601	0.5996	0.6148
512	0.5816	0.6024	0.6052	0.6053	0.6053	0.6436	0.6549
513	0.6284	0.6465	0.6490	0.6490	0.6490	0.6857	0.6938
514	0.6729	0.6887	0.6909	0.6909	0.6909	0.7255	0.7310
515	0.7148	0.7285	0.7304	0.7304	0.7304	0.7626	0.7661

Table 29: (continuation)

X	$BML3$	$HML(0,2)$	$HML(0,3)$	exact	$HMU(1,2)$	$HMU(1,1)$	$BMU3$
516	0.7538	0.7656	0.7671	0.7671	0.7671	0.7967	0.7987
517	0.7895	0.7996	0.8009	0.8009	0.8009	0.8277	0.8285
518	0.8220	0.8305	0.8315	0.8315	0.8315	0.8555	0.8555
519	0.8510	0.8581	0.8589	0.8589	0.8589	0.8801	0.8795
520	0.8766	0.8824	0.8831	0.8831	0.8831	0.9015	0.9006
521	0.8989	0.9037	0.9041	0.9042	0.9042	0.9199	0.9189
522	0.9181	0.9219	0.9223	0.9223	0.9223	0.9357	0.9346
523	0.9344	0.9374	0.9377	0.9377	0.9377	0.9488	0.9478
524	0.9480	0.9504	0.9506	0.9506	0.9506	0.9598	0.9583
525	0.9593	0.9611	0.9613	0.9613	0.9614	0.9687	0.9667
526	0.9685	0.9699	0.9700	0.9701	0.9700	0.9760	0.9737
527	0.9759	0.9769	0.9771	0.9770	0.9770	0.9817	0.9796
528	0.9818	0.9825	0.9827	0.9826	0.9825	0.9863	0.9843
529	0.9865	0.9869	0.9870	0.9869	0.9869	0.9898	0.9881
530	0.9900	0.9903	0.9904	0.9904	0.9904	0.9925	0.9911
531	0.9927	0.9929	0.9932	0.9931	0.9929	0.9946	0.9934
532	0.9948	0.9949	0.9949	0.9949	0.9951	0.9961	0.9952
533	0.9963	0.9963	0.9963	0.9964	0.9964	0.9973	0.9965
534	0.9974	0.9974	0.9977	0.9975	0.9976	0.9981	0.9975
535	0.9982	0.9982	0.9984	0.9983	0.9983	0.9987	0.9983
536	0.9987	0.9987	0.9989	0.9989	0.9990	0.9991	0.9988
537	0.9991	0.9992	0.9994	0.9993	0.9994	0.9994	0.9992
538	0.9994	0.9994	0.9996	0.9996	0.9995	0.9996	0.9994
539	0.9996	0.9996	0.9997	0.9997	0.9997	0.9997	0.9996
CPU	16.02	7.46	18.23	15.69	0.00[3]	0.00[3]	0.00[3]
mean abs. err.	0.0069	0.0064	0.0000	-	-	-	-
max. abs. err.	0.0184	0.0001	-	-	-	-	-
mean rel. err.	0.0374	0.0007	-	-	-	-	-
max. rel. err.	0.9059	0.0820	0.0250	-	-	-	-

[3]The lower and upper bounds were calculated simultaneously

Figure 9 shows the cummulative distribution functions of the project completion times when the two different, multivariate normal and stochastic programming approaches are applied. It can be seen, that the cdf curve produced by the stochastic programming approach runs along significantly higher values than the cdf curve produced by the multivariate normal approach. This means, if we were able to start any activity promptly when all of its predecessor activities are finished in a random instant, then the whole project could be finished in a much shorter time. On the contrary, if we prescribe a deterministic starting time for all of the activities in the project before the starting time of the first activities and guarantee a reliability level to beeing the whole project executable without any conflict, then the project can be finished in a much longer time only. Even so may happen the decision maker sometimes will accept this longer completion time as he cannot guarantee to start the activities of the project in random time instants.

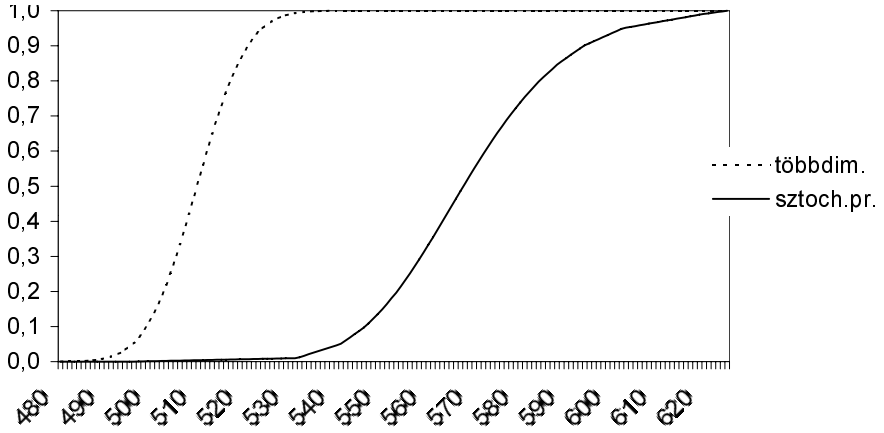

Figure 9: The cdf's of the project completion times determined by the multivariate normal and stochastic programming approaches.

References

[1] Ambartzumian, R., Der Kiureghian, A., Ohanian, V. and Sukiasian, H., Mutinormal probability by sequential conditioned importance sampling: theory and application, *Probab.Engng Mech,* **13(4)** (1998) 299–308.

[2] Anderson, T. W., *An introduction to Multivariate Statistical Analysis* (Wiley New York, 1984).

[3] Anklesaria, K. O. and Drezner, Z., A multivariate approach to estimating the completion time for PERT networks, *Journal of Operational Research Society* **37** (1986) 811–815.

[4] Asmussen, S. and Rubinstein, R. Y., Steady state rare events simulation in queueing models and its complexity properties. *In Advances in Queueing: Theory, Methods and Open Problems,* (1995) 429-461. CRC Press.

[5] Battersby, A., *Network Analysis, third edition*, (St. Martins Press, New York, 1970).

[6] De Boer, P. T., Analysis and efficient simulation of queueing models of telecommunication systems, *Ph.D. thesis*, University of Twente, 2000.

[7] Bonferroni, C. E., *Teoria Statistica Delle Classi e Calcolo Delle Probabilitá* (Volume in onor di Riccardo Dalla Volta, Universitá di Firenze, 1937) 1–62.

[8] Boole, G., *Laws of Thought* (American reprint of 1854 edition, Dover, New York, 1854)

[9] De Boer, P. T., Kroese, D.P. and Rubinstein, R. Y., A fast cross entropy method for estimating buffer overflows in queueing networks. In Fourth Workshop on Rare Event Simulation and Related Combinatorial Optimization Problems, *RESIM/COP* (2002).

[10] De Boer, P.T., Kroese, D.P., Mannor, S. and Rubinstein, R. Y., A tutorial on the cross entropy method, *Annals of Operations Research,* (2004) to appear.

[11] De Boer, P. T., Nicola, V. F. and Rubinstein, R. Y., Adaptive importance sampling simulation of queueing networks. In Proceedings of the 2000 Winter Simulation Conference, Orlando, Florida,(2000) 646–655.

[12] Bukszár, J. and Prékopa, A., Probability Bounds with Cherry trees, *Mathematics of Operations Research* **26** (2001) 174–192.

[13] Bukszár, J. and Szántai, T., Probability bounds given by hypercherry trees, *Optimization Methods and Software*, **17**(2002) 409–422.

[14] Bukszár, J., Quickly computable bounds on the probability of a union, in: *Lecture Notes in Economics and Mathematical Systems, 513,* Stochastic Optimization Techniques, Numerical methods and Technical Applications, Proceedings of the 4th GAMM/IFIP-Workshop on "Stochastic Optimization Techniques, Numerical methods and Technical Applications", held at the Federal Armed Forces University Munich, Neubiberg/München, Germany, June 27-29, 2000, eds. K. Marti, Springer-Verlag, Berlin, Heidelberg, (2002) 79–89.

[15] Bukszár, J., Hypermultitrees and Bonferroni inequalities, *Mathematical Inequalities and Applications* to appear.

[16] Burt, J. M. Jr. and Garman, M. B., Conditional Monte Carlo: A simulation technique for stochastic network analysis, *Management Science*, **18** (1971) 207–217

[17] Chatfield, C. and Goodhardt, G. J., Resultes concerning brand choice, *Journal of Marketing Research*, **12** (1975) 110–113.

[18] Clingen, C. T., A modification of Fulkerson's PERT algorithm, *Operations Research* **12** (1964) 629–632.

[19] Chotikapanich, D. and Griffiths, W., Estimating lorenz curves using a Dirichlet distribution, (2001)

[20] Devroye, L. P., Inequalities for the completion time of stochastic PERT networks, *Mathematics of Operations Research,* **4** (1979) 441–447.

[21] DEVROY, L., Non-Uniform Random Variate Generation, (Springer, New York 1986).

[22] DODIN, B. M., Bounding the project completion time distribution in PERT networks, *Operations Research,* **33** (1985) 862–881.

[23] DODIN, B. and SIRVANCI, M., Stochastic networks and the extreme value distribution, *Computers and Operations Research,* **17 No.4** (1990) 397–409.

[24] DODIN, B. M., Approximating the distribution functions in stochastic networks, *Computers and Operations Research,* **12 No.3** (1985) 251–264.

[25] DODIN, B. M., Rcducihility of stochastic nelworks, *Omega,* **13 No.3** (1985) 223–232.

[26] ELMAGHRABY, S. E., On the expected duration PERT type networks, *Management Science* **13** (1967) 299–306.

[27] ELMAGHRABY, S. E., Activity Networks: Project Planning and Control by Network Models, (Jobn Wiley and Sons, New York 1977.)

[28] ELMAGHRABY, S. E., Object bidding under deterministic and probabilistic activity durations, *European Journal of Operational Research,* **49** (1990) 14–34.

[29] EPPSTEIN, D., Finding the k shortest paths, *SIAM Journal on Computation* **28** (1999) 652–673.

[30] EXTON, H., Multiple hypergeometric functions and applications, (Ellis Horwood, 1976).

[31] FABIUS, J., Two characterizations of the Dirichlet distributions, *Annals of Statistics*, **3** (1973) 583–587.

[32] FISHER, D. L., SAISI, D. and GOLDSTEIN, W. M., PERT networks: OP diagrams, critical paths and the project completion time, *Computers and Operations Research,* **12 No.5** (1985) 471–482.

[33] FOMIN, S. V. , An estimate of the rate of convergence in the multi–dimensional central limit theorem, *Theory of Probability and Its Applications* **27** (1982) 365–368.

[34] FOURER, R., GAY, D. M. and KERNIGHAN, B. W., *AMPL A Modeling Language for Mathematical Programming* (Boyd & Fraser Publishing Company, Danvers, Massachusetts, 1993).

[35] FULKERSON, D. R., Expected critical path lengths in PERT networks, *Operations Research,* **10** (1962) 808–817.

[36] FUKUDA, K., cdd.c : C-implementation of the double description method for computing all vertices and extremal rays of a convex polyhedron given by a system of linear inequalities, Department of Mathematics, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1993. (program available from http://www.cs.mcgill.ca/ fukuda/soft/cdd_home/cdd.html)

[37] GARVELS, M. J. J., The splitting method in rare event simulation. *Ph.D. thesis*, (University of Twente 2000).

[38] GASSMANN, H., Rectangle probabilities of the trivariate normal distribution, working paper, school of Business Administration,Dalhousie University, Halifax, Canada; available from *http://www.mgmt.dal.ca/sba/profs/hgassmann,* (2000).

[39] GASSMAN, H., DEÁK, I. AND SZÁNTAI, T., Computing multivariate normal probabilities A new look, *Journal of Computational and Graphical Statistics,* **11 No.4** (2002) 920–949.

[40] GENZ, A., Numerical computation of the multivariate normal probabilities, *Journal of Computational and Graphical Statistics,* **1** (1992) 141–150.

[41] GENZ, A., Comparison of methods for the computation of multivariate normal probabilities, *Computing Science and Statistics*, **25** (1993) 400–405.

[42] GOLDBERG, G. and NEUMANN, M., Distribution of subdominant eigenvalues of matrices with random rows *SIAM J. MATRIX ANAL. APPL.,* **24 No.3** (2003) 747–761

[43] GOODHARDT, G. J., EHRENBERG, A. S. C. and CHATFIELD, C., The Dirichlet : A comprehensive model of bying behaviour, *Journal of the Royal Statistical Society, Series A*, **147** (1984) 621–655.

[44] GOODMAN, I. R., and NGUYEN, H. T., Probability updating using second order probabilities and conditional event algebra, *Information science,* **121** (1999) 295–347.

[45] GOLMKO-GINZBURG, D., A new approach to the activity-time distribution in PERT, *Journal of Operational Reseearch Society,* **40 No.4** (1989) 389–393.

[46] GOUDA, A., MONHOR, M. and SZÁNTAI, T., A New stochastic programing model of PERT, *Alkalmazott Matematikai Lapok,* (in press).

[47] GOUDA, A., MONHOR, M. and SZÁNTAI, T., A New stochastic programing model of PERT, in: The IFIP/IIASA/GAMM/–Workshop on Coping with Uncertainty, held at the International Institute for Applied Systems Analysis – IIASA, Laxenburg/Vienna, Austria, December 13-16, 2004, Special Volume to Springer-Verlag, Berlin, (2005) (submitted).

[48] GOUDA, A. and SZÁNTAI, T., New sampling techniques for calculation of Dirichlet probabilities, *The Centeral European Journal of Operations Research.* **12 No.4** (2004) 389–403.

[49] GOUDA, A. and SZÁNTAI, T., On Numerical calculation of probabilities according to Dirichlet distribution, *Annals of Operations Research,* (submitted).

[50] GOUDA, A. and SZÁNTAI, T., Estimation of Rare event probabilities in stochastic networks, *Mathematical and Computer Modeling,* (submitted)

[51] GOUDA, A. and SZÁNTAI, T., New sampling techniques in variance reduction Monte Carlo simulation algorithms for Dirichlet distribution, at Annual International Conference on Operations Research, Heidelberg, Germany September 3-5, 2003, (abstract).

[52] Gouda, A. and Szántai, T., Estimation of Rare event probabilities in stochastic networks with exponential and beta distribution. *RESIM/COP'04, Proceeding of 5th International Workshop on Rare Event Simulation and Related Combinatorial Optimization Problems,* Budapest, Hungary, Septemper 2004, 1–24.

[53] Hagstrom, J. N., Computing the probability distribution of project duration in a FERT network *Networks,* **20 No.23** (1990) l–244.

[54] Hershauer, J. C. and Nabielsky, G., Estimating activity times, *Journal of Systems Management,* (1972) 17–21.

[55] Hammersley, J. M. and Handscomb, D. C., Monte Carlo Methods, Methuen & Co ltd., London 1967.

[56] Homem-de Mello, T. and Rubinstein, R. Y., Rare event probability estimation for static models via cross-entropy and importance sampling, *Submitted for publication,* (2002).

[57] Homem-de Mello, T. and Rubinstein, R. Y., Estimation of Rare event probabilities using cross-entropy, In Proceedings of the 2002 Winter Simulation Conference, (Edited by E. Yucesan C.-H. Chen, J. L. Snowdon, and J. M. Charnes) (2002).

[58] Hunter, D., Bounds for the probability of a union, *Journal of Applied Probability* **13** (1976) 597–603.

[59] James, Ian R., Products of independent beta variables with application to Connor and Mosimann's generalized Dirichlet distribution, *Journal of the American Statistical Association*, **67** (1972) 910–912.

[60] James, Ian R., Multivariate distributions which have beta conditional distributions, *Journal of the American Statistical Association*, **70** (1975) 681–684.

[61] Johnson, N. L., An approximation to the multinomial distribution; some properties and applications *Biometrika*, **47** (1960) 93–103.

[62] KAMBUROWSKI, J., An upper bound on the expected completion time of PERT networks, *European Journal of Operational,* **21 No.23** (1985) 206–212.

[63] KAMBUROWSKI, J., Normally distributed activity durations in PERT networks, *Journal of Operational Research Society,* **36 No. 11** (1986) 10. 51–1057.

[64] KEITH, J. and KROESE, D. P., Sequence alignment by rare event simulation, In Proceedings of the 2002 Winter Simulation Conference, (2002) 320–327. (San Diego. 2002).

[65] KOTZ, S., BALAKRISHNAN, N. and JOHNSON, N. L., Continuous Multivariate Distributions, (Wiley, New York, 2000).

[66] KLAFSZKY EMIL, *Hálózati folyamok, Az operációkutatás matematikai módszerei előadássorozat jegyzete*, szerk. Prékopa András, Bolyai János Matematikai Társulat, Budapest, 1969.

[67] KLEIJNEN, J. P. C. and RUBINSTEIN, R. Y., Optimization and sensitivity analysis of computer simulation models by the score function method. *European Journal of Operations Research,* **88** (1996) 413–427.

[68] KLEINDORFER, G. B., Bounding distributions for a stochastic acyclic network, *Operations Research* **19** (1971) 1586–1601.

[69] KOUNIAS, E. G., Bounds on the probability of a union, with applications, *Annals of Mathematical Statistics*, **39 No.6** (1968) 2154–2158.

[70] KULKARNI, V. G. and ADLAKHA, V. G., Markov and Markovregenerative PERT networks, *Operations Research,* **34 No.5** (1986) 769–781.

[71] LITTLEFIELD, T. K. and RANDOLPH, P. H., An answer to Sasieni's question on PERT times, *Management Science* **33** (1987) 1357–1359.

[72] LIEBER, D., RUBINSTEIN, R. Y. and ELMAKIS, Quick estimation of rare events in stochastic networks, *IEEE Transactions on Reliability,* **46(2)** (1997) 254–265.

[73] Malcolm, D. G., Roseboom, J. H., Clark, C. E. and Fazar, W., Application of a technique for research and development program evaluation, *Operations Research* **7** (1959) 646–669.

[74] Meilijson, I. and Nádas, A., Convex majorization with an application to the length of critical paths, *Journal of Applied Probability* **16** (1979) 671–677.

bibitemmk97 McLachlan, G. and Krishnan, T., The EM algorithm and extensions, (John Wiley & Sons 1997).

[75] Monhor, D., An Inequality for the Dirichlet distribution, *Acta Mathematica Hungarica,* **47(1-2)** (1986) 161–163.

[76] Monhor, D., An approach to PERT: Application of Dirichlet distribution, *Optimization,* **18** (1987) 113–118.

[77] Martin, J. J., Distribution of the time through a directed acyclic network *Operations Research,* **13** (1965) 46–66.

[78] Nádas, A., Probabilistic PERT, *IBM J. Res. Dev.* **23** (1979) 339–347.

[79] Narayanan, A., A Small sample properties of parameter estimation in the Dirichlet Distribution, *Communications in Statistics-Simulation and computation,* **20** (1991) 647–666.

[80] Pandy, MD., An effective approximation to evaluate multinormal integras, *Structural Safety,* **20** (1998) 51–67.

[81] Pandy, MD. and Sakar, A., A Comparison of a simple approximation for multinormal integration with an importance sampling-based simulation method, *Probabilistic Engineering Mechanics,* **17(4)** (2002) 215–218.

[82] Phillips, P. C. B., The characteristic function of the Dirichlet and multivariate F distributions, *Cowles Foudation for Research in Econonmics,* Discussion paper **865** (1988) 1–17.

[83] PLACKETT, R. L., A reduction formula for normal multivariate probabilities, . *Biometrika,* **41** (1954) 351–360.

[84] PRÉKOPA, A. and KELLE, P., Reliability inventory models based on stochastic programming, *Studies in applied stochastic programming,* I, ed. by A. Prékopa, **80**(1978).

[85] PRÉKOPA, A. and LONG, J., New bounds and approximations for the probability distribution of the length of the critical path, *RUTCOR Research Report* **RRR 16-92** (1992).

[86] PRÉKOPA, A., LONG, J. and SZÁNTAI, T., New bounds and approximations for the probability distribution of the length of the critical path, in: Lecture Notes in Economics and Mathematical Systems, 532, Dynamic Stochastic Optimization, Proceedings of the IFIP/IIASA/GAMM-Workshop on 'Dynamic Stochastic Optimization', held at the International Institute for Systems Analysis (IIASA), Laxenburg, Austria, March 11-14, 2002, eds. K. Marti, Y. Ermoliev and G. Pflug, Springer-Verlag, Berlin, Heidelberg, 2004, 293–320.

[87] PRÉKOPA, A., Stochastic Programming, (Kluwer Academic Publishers, Dordrecht, 1995.)

[88] PRÉKOPA, A., On the concavity of multivariate probability distribution functions, *Operations research etters,* **29** (2001) 1–4.

[89] ROBILLARD, P. and TRAHAN, M., Expected completion time in PERT network, *Operations Research* **24** (1976) 177–182.

[90] ROBILLARD, P. and TRAHAN, M., The completion time of PERT network, *Operations Research* **25** (1977) 15–29.

[91] RUBINSTEIN, R. Y., Simulation and the Monte Carlo Method, (John Wiley & Sons, 1981).

[92] RUBINSTEIN, R. Y., Optimization of computer simulation models with rare events, *European Journal of Operations Research,* **99** (1997) 89–112.

[93] RUBINSTEIN, R. Y., The Cross-Entropy method for combinatorial and continuous Optimization, *Methodology and Computing in Applied Probability*, **1** (1999) 127–190.

[94] RUBINSTEIN, R. Y., Combinatorial optimization, cross-entropy, ants and rare events. In S. Uryasev and P. M. Pardalos, editors, *Stochastic Optimization: Algorithms and Applications, Kluwer,* (2001) 304–358.

[95] RUBINSTEIN, R. Y., Cross-entropy and rare events for maximal cut and partition problems, *Acm Transactions on Modeling and Computer Simulation,* **12 No.1** (2002) 27–53.

[96] RUBINSTEIN, R. Y., and KROESE, D., Lecture notes on the cross-entropy method. Manuscript, (2002).

[97] RUBINSTEIN, R. Y., and MELAMED, B., Modern Simulation and Modeling (John Wiley & Sons, New York 1998).

[98] RUBINSTEIN, R. Y., and SHAPIRO, A., Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization via the Score Function Method, (John Wiley & Sons, New York, 1993).

[99] SCULLI, D., The completion time of PERT networks, *Joumal of Opemtional Research Society,* **34** (1983) 155–158.

[100] SCULLI, D., and WONG, K. L., The maximum and sum of two beta variables and the analysis of PERT networks, *Omega,* **13** bf No.3 (1985) 233–240.

[101] SHAHABUDDIN, P., Rare Event Simulation of Stochastic Systems, Proceedings of the 1995 Winter Simulation Conference, Washington, D.C., IEEE Press, (1995) 178–185.

[102] SHOGAN, A. W. , Bounding distributions for a stochastic PERT network, *Networks,* **7** (1977) 359–381.

[103] SIGAL C. E., PRITSKER, A. A. B., and SOLBERG, J. J., The stochastic shortest route problem, *Opemtiorrs Research*, **28 No.5** (1980) 1122–1129.

[104] Sobel, M. and Uppuluri, V. R. R., Sparse and crowded cells and Dirichlet distributions, *Annals of Statistics*, **2** (1974) 977–987.

[105] Szántai, T., Numerical Evaluation of Probabilities Concerning Multidimensional Probability Distributions. Thesis, Hungarian Academy of Sciences, Budapest (1985).

[106] Szántai, T., Evaluation of a special multivariate gamma distribution function, *Mathematical Programming Study,* **27** (1986) 1–16.

[107] Szántai, T., Calculation of the multivariate probability distribution function values and their gradient vectors. Working paper wp-87-82, IIASA, (1987).

[108] Szántai, T., A computer code for solution of probabilistic-constrained stochastic programming problems. In Ermoliev, Y., Wets, R.J-B., editors, *Numerical Techniques for Stochastic Optimization,* (1988) 229–235. Springer Verlag.

[109] Szántai, T., Probabilistic constrained programming and distributions with given marginals, in : Distributions with given marginals and moment problems, Proceedings of the 3rd Conference on "Distributions with Given Marginals and Moment Problems", held at Czech Agricultural University, Prague, Czech Republic, September 2-6, 1996, eds. V. Benes and J. Stepan, Kluwer Academic Publishers, Dordrecht/Boston/London, (1997) 205–210.

[110] Szántai, T., Improved Bounds and Simulation Procedures on the Value of the Multivariate Normal Probability Distribution Function, *Annals of Operations Research,* **100** (2000) 85–101.

[111] Tempelman, C., Imputation of missing data items under linear restrictions, United Nations Statistical Commission and Economic Commission for Europe Conference of European Statisticians. (Madrid, Spain, 20-22 October 2003).

[112] Tiao, G. G. and Guttman, I., The inverted Dirichlet distribution with applications, *Journal of the American Statistical Association*, **60** (1965) 793–805, 1251–1252.

[113] Tomescu, I., Hypertrees and Bonferroni inequalities, *Journal of Combinatorial Theory, Series B* **41** (1986) 209–217.

[114] Vanderbei, R. J., LOQO: An interior point code for quadratic programming, *Optimization Methods and Software* **12** (1999) 451–484.

[115] Vanderbei, R. J., LOQO user's manual – version 3.10, *Optimization Methods and Software* **12** (1999) 485–514.

[116] Van Siyke, R. M., Monte Carlo methods and PERT problem, Operations Rese-Carch, *Operations Research,* **11** (1963) 839–960.

[117] Wallace, S. W., Bounding the expected time-cost curve for a stochastic PERT network from below, *Operations Research Letters* **8** (1989) 89–94.

[118] Weiss, G., Stochastic bounds on distributions of optimal value functions with applications to PERT, Network flows and reliability, *Operations Research* **34** (1986) 595–605.

[119] Wilks, S. S., Mathematical Statististics (John Wiley and Sons, New York, London. 1962.)

[120] Worsley, K. J., An Improved Bonferroni Inequality and Applications, *Biometrika*, **69** (1982) 297–302.

[121] Yassaee, H., Inverted Dirichlet distibution and multivariate logistic distributions, *Canadian Journal of Statistics*, **2** (1974) 99–105.

[122] Yassaee, H., Probability integral of inverted Dirichlet distribution and it's application, *Compstat*, **76** (1976) 64–71.

[123] Yassaee, H., On integrals of Dirichlet distribution and their applications, Preprint, Arya-Mehr University of Technology, Tehran, Iran, (1981).

List of Figures

List of Tables