LAPPEENRANTA UNIVERSITY OF TECHNOLOGY
Faculty of Technology
Department of Mathematics and Physics
Laboratory of Applied Mathematics

Stochastic approximation methods in stochastic optimization

The topic of this Master's thesis was approved by the council of the Department of Mathematics and Physics on 25^{th} of October, 2013.

The examiners of the thesis were:
Prof. Heikki Haario and PHD Tuomo Kauranne.

The thesis was supervised by Professor Heikki Haario.

In Lappeenranta, 8.11.2013

Ilya Bobylev
Ruskonlahdenkatu 13-15 E6
53850 Lappeenranta
Phone: +3580417061366
Ilya.Bobylev[at]lut.fi

Abstract

Lappeenranta University of Technology
Faculty of Technology
Department of Mathematics and Physics

Ilya Bobylev

Stochastic approximation methods in stochastic optimization

Thesis for the Degree of Master of Science in Technology

2013 year

72 pages, 33 figures, 6 tables

Supervisors: Prof. Heikki Haario, PHD Tuomo Kauranne.
Examiners: Prof. Heikki Haario, PHD Tuomo Kauranne.

Key words: Stochastic Optimization, Deterministic equivalents, Stochastic Quasi-gradient algorithm (SQG), mean-value, value-at-risk (VaR), conditional value-at-risk (CVaR).

Stochastic approximation methods for stochastic optimization are considered. Reviewed the main methods of stochastic approximation: stochastic quasi-gradient algorithm, Kiefer-Wolfowitz algorithm and adaptive rules for them, simultaneous perturbation stochastic approximation (SPSA) algorithm. Suggested the model and the solution of the retailer's profit optimization problem and considered an application of the SQG-algorithm for the optimization problems with objective functions given in the form of ordinary differential equation.

Acknowledgements

First of all, I would like to express my respect to the Department of Mathematics for the scholarship during my studies, otherwise my stay in Finland would not be possible.

Secondly, I have to thank the Finnish atmosphere: peaceful and quiet, that gives wide possibilities for productive study and work.

Finally my sincere gratitude goes to professors Heikki Haario, Matti Heilio and Tuomo Kauranne for their supervision and advices.

Lappeenranta, 2013.

Ilya Bobylev

Symbols and abbreviations

- CDF - cumulative distribution function.
- PDF - probability density function.
- SQG - stochastic quasi-gradient algorithm.
- VaR - value-at-risk.
- CVaR - conditional value-at-risk.
- w.r.t - with respect to.
- a.s. - convergence almost sure.
- $\sigma\{X_1, X_2, ..., X_n\}$ - sigma-algebra, generated by the distribution of random variables $X_1, X_2, ..., X_n$.
- MC - Monte-Carlo simulation.
- MCMC - Markov Chain Monte-Carlo simulation.
- LP - linear programming.
- SPSA - Simultaneous perturbation stochastic approximation.
- SA - Stochastic approximation.
- KW - Kiefer-Wolfowitz algorithm.
- ODE - Ordinary differential equation.

Contents

List of Tables

List of Figures

1 Introduction

Optimization problems under uncertainty have been intensively studied by many researches during the last 50 years. In fact mathematical models based on stochastic theory are more approached to the real life then deterministic because the random parameters which considered in objective function or in constraints reflect the nature of uncertainty during the decision making.

Stochastic models are widely used in insurance, financial and portfolio optimization where the behavior of market cannot be precisely forecasted or in aerospace technologies where it is required to obtain robust results.

Generally the solution of the stochastic optimization problems looks as follows: firstly researcher attempts to derive a deterministic equivalent and use deterministic method of linear or nonlinear optimization depending on the case. If the cost function is quite complicated but if the gradient can be explicitly found for the given realization of the random parameters or successfully approximated with finite differences, then stochastic approximation (SA) will be a good choice as a solution technique. Finally if the method fails (example: an objective function is multimodal, contains both discrete and continuous arguments), then the Monte-Carlo simulation, genetic algorithms, simulated annealing or other methods of global optimization are used.

Robustness, simplicity and computation speed makes stochastic approximation a very popular algorithm for the wide class of optimization problems therefore development of SA is an important and demanded research area.

1.1 Objective of the thesis

The main purpose of the thesis is to study and develop stochastic approximation algorithms which are used in stochastic optimization.

1.2 Structure of the thesis

The material of the thesis is organized as follows: the first 3 chapters are devoted for the definitions, theoretical background of the stochastic approximation and are mainly needed to prepare the reader for the application problems that are studied in the chapters 4-5.

1.3 Scientific novelty

The thesis contains several new theoretical results: deterministic equivalents of the optimization problems with mean-value and CVaR - electricity retailer profit optimization problem (see theorems in the corresponding section) and interesting computational idea: equation in variations used in SQG-algorithm for the objective functions given in ODE-form.

The first result gives an opportunity to substitute the solution of the stochastic problem with deterministic one and instantly get the result using a well-known simplex-method for the linear-programming.

The second result significantly increases computational speed of the stochastic approximation procedure for the considered problem.

2 Optimization criteria, used in stochastic programming

Let $\Phi(u, X)$ further in this section denotes a cost function, where $u \in U \subset \mathbf{R}^n$ is an optimization strategy and X is a random vector with the realizations belonging to the set $E \subseteq \mathbf{R}^m$.

2.0.1 Mean-value criterion

Historically the first was a mean-value criterion which can be defined as follows:

$$\min_{u \in U} E[\Phi(u, X)]. \tag{1}$$

Intuitively the criterion means to chose those strategy, which minimizes average loses, represented by the objective function.

The main properties of the mean-value criterion that allows one to derive a solution and build effective algorithms are:

- Linearity:
$$E[aX + bY + c] = aE[X] + bE[Y] + c, \tag{2}$$
where X, Y are random variables and parameters $a, b, c \in \mathbf{R}$;
- Simple representation for the discrete distribution with finite number of realizations:
$$E[X] = \sum_{i=1}^{n} x_i p_i, \tag{3}$$

where x_i, $i = 1, .., n$ are realisations of the random vector X, and p_i, $i = 1, .., n$ are the corresponding probabilities, so that $\mathbf{P}(X = x_i) = p_i$.

Unfortunately, the mean-value criterion together with such attractive properties has a number of cases when it fails: to get a basic idea one can imagine the situation when the decision about the conditions of the patients in a clinic are made, based on the average temperature among them. The nonsense is obvious - it is possible to have the value of the criterion $36,6^\circ$ but all the patients will be about to die: the first half with $\sim 34C^\circ$ and the rest with $\sim 40C^\circ$.

2.0.2 Minimax criterion

The minimax criterion comes from the game theory and in the statistical analysis means to chose those strategy, which attains the minimum of loses in the worst possible case:

$$\min_{u \in U} \max_{X \in E} \Phi(u, X). \tag{4}$$

The problem statement can be treated as a game against an aggressive nature all the time forming the worst possible case for the decision maker. The positive side of the strategy chosen according to minimax is an assurance that the losses will be always below forecasted. Drawback effects are obvious: such strategy could be too cautious and wasteful and often unachievable. For example in the book [4] the chapter devoted to optimization of the aircraft runway area contains a joke that in order to solve the problem with the minimax criterion it is required to concrete the whole globe together with oceans and sees.

2.0.3 Value-at-Risk (VaR) criterion

To show an idea behind the VaR criterion let us define two concepts: the probability function $P_\phi(u)$ and the quantile function $\phi_\alpha(u)$ [4].

$$P_\phi(u) = \mathbf{P}\{\Phi(u, X) \leq \phi\}, \tag{5}$$

$$\phi_\alpha(u) = \min\{\varphi : P_\varphi(u) \geq \alpha\}. \tag{6}$$

The probability function represents probability that the cost function $\Phi(u, X)$ does not exceed the level ϕ for a chosen fixed strategy u, while the quartile function indicates the corresponding minimal level. Finally, to get the VaR-strategy, the following optimization task have to be solved:

$$\min_{u \in U} \phi_\alpha(u). \tag{7}$$

Unfortunately VaR lacks of some desirable theoretical properties. Firstly, it does not preserve convexity, i.e. convexity of the objective function w.r.t. to a strategy, does not always results in a convexity of the quantile function. Secondly, there are quite a small number of cases when it is possible to derive a gradient w.r.t to a strategy of the VaR explicitly. This makes the solution process of VaR problems extremely complicated and therefore the Monte-Carlo simulation is often used. One approach to deal with the quantile criterion is to build an upper-value estimate, based on the confidence set.

According to [4] the problem (7) may be solved if one gets a solution of the following minimax problem:

$$\phi_\alpha(u) = \min_{E \in \mathcal{E}_\alpha} \max_{x \in E} \Phi(u, x),$$

where $\mathcal{E}_\alpha$ is a family of the confident sets $P(E) \geq \alpha$, $E \in \mathcal{E}_\alpha$. Obviously if instead of the optimal confident set E^* will be used an arbitrary one $\hat{E}$, then an upper estimate to the optimal solution will be found. Based on that observation there were many application problem solved [23, 24, 25].

2.0.4 Conditional Value-at-Risk (CVaR) criterion

The criterion was created as an enhancement to VaR and represents an average rate of loses exceeding the level $\phi_\alpha(u)$:

$$\psi_\alpha(u) = E[\Phi(u, X) \mid \Phi(u, X) \geq \phi_\alpha(u)] = \frac{1}{1-\alpha} \int\limits_{\Phi(u, X(w)) \geq \phi_\alpha(u)} \Phi(u, X(w)) dP(w), \quad (8)$$

According to [7], a minimization of CVaR over the strategy $u \in U$ equals to the solution of the following optimization problem:

$$\psi^* = \min_{(u,\phi) \in U \times R} F_\alpha(u, \phi), \quad (9)$$

where

$$F_\alpha(u, \phi) = \phi + \frac{1}{1-\alpha} \cdot E[\max\{F(u, Z) - \phi, 0\}. \quad (10)$$

This results opens wide possibilities for the CVaR-optimization via well-known techniques of the mean-value minimization.

2.0.5 Demonstration example

Let us consider a simple example to show the difference between the foregoing criteria in practice. Assume that a cost function is defined as presented below:

$$\Phi(u, X) = u^2 + X \quad (11)$$

and a random parameter is a normally-distributed $X \sim N(0, 1)$.

- Mean-value solution:

$$\min_{u \in \mathbf{R}} \{E[u^2 + X]\}. \quad (12)$$

Using a linearity of the mathematical expectation, we get the following deterministic equivalent:

$$\min_{u \in \mathbf{R}} u^2 \tag{13}$$

and the corresponding solution is $u^* = 0$, $E[u^*, X] = 0$.

- Minimax solution:

$$\min_{u \in \mathbf{R}} \max_{X} \{u^2 + X\}. \tag{14}$$

The Normal distribution has a continuous CDF, therefore $\forall\, C \in \mathbf{R}$: $\mathbf{P}(X > C) > 0$. As a result, the left hand side of the previous equation is a non-restricted and the problem does not have a solution.

- VaR-solution:

$$\min_{u \in \mathbf{R}} \{\phi_\alpha(u)\}. \tag{15}$$

The probability function looks as follows:
$P_\phi(u) = \mathbf{P}(u^2 + X \leq \phi) = \mathbf{P}(X \leq \phi - u^2) = F_X(\phi - u^2)$, where F_X is a CDF of X
The quantile function:
$\phi_\alpha(u) = min\{\phi : P_\phi(u) \geq \alpha\} = min\{\phi : F_X(\phi - u^2) \geq \alpha\}$
By the definition CDF is a monotonically non-decreasing function, therefore:
$F_X(\phi - u^2) \geq \alpha \iff \phi - u^2 \geq x_\alpha \Longrightarrow \phi \geq u^2 + x_\alpha \Longrightarrow \phi_\alpha(u) = u^2 + x_\alpha$
Finally, the deterministic equivalent assumes the shown below form:

$$\min_{u \in \mathbf{R}} \{u^2 + x_\alpha\}. \tag{16}$$

As it can be clearly seen: $u^* = 0$, $\phi_\alpha(u^*) = x_\alpha$.

- CVaR-solution: Considering the previous VaR soultion:
$\psi_\alpha(u) = E[\Phi(u, X) \mid \Phi(u, X) \geq \phi_\alpha(u)] = E[u^2 + x \mid u^2 + x \geq u^2 + x_\alpha] = u^2 + E[x \mid x \geq x_\alpha] = u^2 + y_\alpha$, where $y_\alpha = \alpha - CVaR_{N(0,1)}$. Finally, the deterministic equivalent assumes the following form:

$$\min_{u \in \mathbf{R}} \{u^2 + y_\alpha\} \tag{17}$$

and the solution: $u^* = 0$, $\psi_\alpha(u^*) = y_\alpha$.

It is worth to note, that in this particulary example, the strategy $u = 0$ is an optimal simultaneously for the mean-value, VaR, and CVaR criteria.

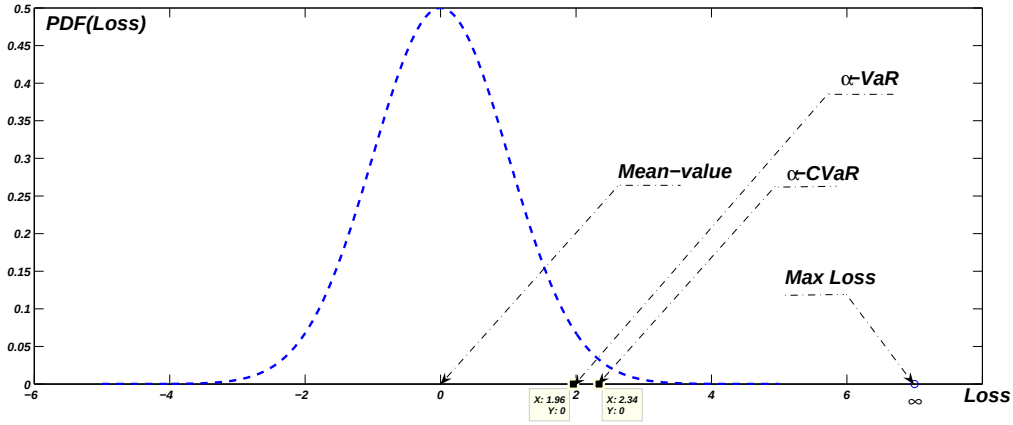

Figure 1: Illustration of the solution.

3 Stochastic approximation

A history of the stochastic approximation starts from the article of Robbins and Monro [20] where authors applied a recursive SA procedure to find a root of the equation observing noisy data. As it has been discovered, there is no need to know exact values of the function, because during the solution we are only interested in the direction of the root. As a result of this observation, the process can work with noisy values instead of the exact data.

Let us review several illustrative application problems from the real life where the stochastic approximation algorithms can be quite useful.

- During the testing of insecticides there are a problem of determining the maximum dose which guarantee the intensity of the reaction.
- The hardness of the iron-copper alloy depends on the time of temperature impact. Let u denotes a time, and $Y(u)$ - the hardness of alloy. The computation problem is to determine such u when the hardness of alloy $Y(u)$ reaches α. It is also a well-known fact that the hardness of alloy varies over the items.
- The sensitivity of the explosive substance to the physical impact are measured by hitting. The small item of the substance are punched by solid object thrown from the fixed height. Some samples will explode and some of them not. The problem is to determine the critical height for the substance.
- A grain field are fertilized by some chemicals. Let u denotes an amount of the fertilizer and Y(u) denotes a corresponding amount of grain gathered from the field. It can be clearly seen that lack of the fertilizer (as well as over-fertilization) yields to a reduction of the crops. Only some optimal amount of chemicals leads to a significant outcome. Moreover the prolificness of the field are changing over the years even if an amount of the fertilizer u remains the same.

The solution of the first three cases can be organized according to the procedure of Robins and Monro: a tester picks an arbitrary value x_1 from the admissible area, conducts an experiment and observes a realisation $y(x_1)$ of the random variable $Y(x_1)$ with expectation $M(x_1) = E\{Y(x_1)\}$, where M is some increasing function. The tester also chooses a decreasing sequence $a_n = \frac{c}{n}$, where c is a positive constant and n is a number of the step. The problem to be solved is to determine such θ, that $M(\theta) = \alpha$. For the next experiment he takes x according to the rule:

$$x_{n+1} = x_n - a_n(y(x_n) - \alpha). \tag{18}$$

To understand the main idea of the method assume $\alpha = 0$. In this case the previous statement yields

$$x_{n+1} = x_n - \frac{c}{n} y(x_n). \tag{19}$$

If $y(x_n) > 0$, then $x_{n+1} < x_n$, and $y(x_n) < 0$ leads correspondingly to $x_{n+1} > x_n$, meaning that the equation (19) looks reasonable since we are searching for $\theta : M(\theta) = 0$.

The solution of the last problem can be organized with the help of stochastic quasi-gradient method which is studied in details further in the thesis.

Kiefer and Wolfowitz (in 1d-case) [17] and Blum [21] (in multidimensional case) applied stochastic approximation to optimize mean-value functionals.

The books of Ermoliev [9], Powel [22] contain easy understandable proofs and motivation of using stochastic approximation.

Kan and Kibzun [4] have applied stochastic approximation for the optimization problems with VaR criterion.

Spall [19] have suggested to use random perturbations in the classical KW-procedure and decrease the number of evaluations of the objective function. Moreover his personal page [19] lists a large number of references regarding applications of SPSA.

Let us list several different areas of science where the SPSA-algorithms have been successfully applied [19]:

- Aircraft modeling and control
- Atmospheric and planetary modeling
- Cardiological data analysis
- Noise cancellation
- Queuing network design
- Robot control
- Sensor placement
- Traffic signal timing or other transportation problems
- Underground mine detection

Granichin and Polyak [29] made a generalization of the SPSA-method, estimated the convergence speed and suggested several different versions of the procedure.

The next section contains explanations of the stochastic approximation procedures which are used for the optimization of cost-functions.

The presentation is organized as follows: firstly we introduce conditional expectation, then on its basis some definitions from martingale theory will be shown. Having insight in the martingale convergence theory we gradually start the main proof regarding the convergence of the SQG-procedure. At the end of the section we describe a relatively new area of stochastic approximation - the SPSA algorithm and discuss adaptive procedures that practically enhance convergence.

3.1 Conditional mathematical expectation

Let $\{\Omega, \mathcal{F}, \mathbf{P}\}$ denotes a probability space, $\mathcal{G}$ is a σ-algebra of random events $\mathcal{G} \subseteq \mathcal{F}$ and ξ is a random variable $E[\xi] < \infty$. A random variable $E[\xi \,|\, \mathcal{G}]$ is called a conditional mathematical expectation of ξ w.r.t. $\mathcal{G}$ if the following conditions are valid:

1. $E[\xi \,|\, \mathcal{G}]$ is a $\mathcal{G}$-measurable function,
2. for every set $A \subseteq \mathcal{G}$,

$$\int\limits_A \xi(\omega)\mathbf{P}(d\omega) = \int\limits_A E\left[\xi \,|\, \mathcal{G}\right](\omega)\mathbf{P}(d\omega). \tag{20}$$

The main properties of the conditional mathematical expectation, that is used in the next paragraphs are [13]:

1. if $\xi = C = const$, then

$$E[\xi \,|\, \mathcal{G}] = C \;\; (a.s.), \tag{21}$$

2. if $\xi \leq \eta$ (a.s) then

$$E[\xi \,|\, \mathcal{G}] \leq E[\eta \,|\, \mathcal{G}] \;\; (a.s.), \tag{22}$$

3. if the random variable ξ is measurable with respect to σ-algebra $\mathcal{G}$, then

$$E[\eta \,|\, \mathcal{G}] = \xi \;\; (a.s.), \tag{23}$$

4. connection with unconditional expectation:

$$E\left[E[\xi \mid \mathcal{G}]\right] = E[\xi] \ \ (a.s.), \tag{24}$$

5. linearity. If $a, b \in \mathbf{R}$ and $E\left[\xi\right] < \infty$, $E\left[\eta\right] < \infty$ then

$$E\left[a\xi + b\eta \mid \mathcal{G}\right] = aE\left[\xi \mid \mathcal{G}\right] + bE\left[\eta \mid \mathcal{G}\right] \ \ (a.s.), \tag{25}$$

6. if the random variable ξ does not depend on σ-algebra $\mathcal{G}$, then

$$E[\xi \mid \mathcal{G}] = E[\xi] \ \ (a.s.), \tag{26}$$

7. conditional mathematical expectation with respect to a random variable is defined as an expectation w.r.t. the σ-algebra, generated by this variable:

$$E\left[\xi \mid \mathcal{F}^{n}\right] = E\left[\xi \mid \eta\right], \mathcal{F}^{n} = \sigma\{\eta\}. \tag{27}$$

To get intuitive sense regarding conditional mathematical expectation, one can imagine $E\left[X \mid \mathcal{F}\right]$ as a rough, averaged version of X, because on the arbitrary set $A \in \mathcal{F}$ random variable X can assume any values, but $E\left[X \mid \mathcal{F}\right]$ gets only one fixed value: $E\left[X \mid A\right]$ what is an average value of X on the set $A \in \mathcal{F}$.

3.2 Discrete time martingales

Assume that $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq ... \subseteq \mathcal{F}$ is a non-decreasing family of σ-algebras, defined on a probability space $\{\Omega, \mathcal{F}, \mathbf{P}\}$. For example if $\{\xi_n\}, n \geq 0$ is a set of random variables given on $\{\Omega, \mathcal{F}, \mathbf{P}\}$ and $\mathcal{F}_n^l = \sigma\{\xi_0, ..., \xi_n\}$ is a sigma-algebra, generated by $\{\xi_k, \ k = 0, ..., n\}$, then $\{\mathcal{F}_n^l\}$ is a non-decreasing family of σ-algebras.

Let $\{X_n, \ n \geq 0\}$ is a sequence of random variables defined on $\{\Omega, \mathcal{F}, \mathbf{P}\}$. If for all $n \geq 0$ X_n is $\mathcal{F}_n$-measurable, then $\{X_n, \mathcal{F}_n\}$ is called a stochastic sequence.

If for all $n \geq 0$ X_n is $\mathcal{F}_{n-1}$-measurable, then the sequence $\{X_n, \mathcal{F}_{n-1}\}$ is called a stochastic sequence.

Stochastic sequence $\{X_n, \mathcal{F}_n\}$, $E\left[|X_n|\right] < \infty$ is called:

- a martingale, if $E\left[X_{n+1} \mid \mathcal{F}_n\right] = X_n \ \ (a.s.)$,
- a submartingale, if $E\left[X_{n+1} \mid \mathcal{F}_n\right] \geq X_n \ \ (a.s.)$,
- a supermartingale, if $E\left[X_{n+1} \mid \mathcal{F}_n\right] \leq X_n \ \ (a.s.)$.

3.3 Convergence of the supermartingales

It can be clearly seen that supermartingales are stochastic analogues of non-increasing sequences of the real numbers. As a result, supermartingales (as well as submartingales) under some conditions can have a limit (what is in fact a random variable). The corresponding statement are shown in the following theorem [9].

Theorem 3.1 (Convergence theorem)**.** *If* $\{X_n, \mathcal{F}_n\}$ *is a supermartingale, such that* $\inf_n E[X_n^-] > -\infty$, $X_n^- = \min\{X_n, 0\}$ *then*

$$|\lim_{n\to\infty} X_n| < \infty. \tag{28}$$

From the theorem above it is possible to conclude: if $\{X_n, \mathcal{F}_n\}$ is a non-negative supermartingale, then a.s. (with probability 1) there exists its limit.

3.4 Convergence of the random variables

Let $\{X_1, ..., X_n, ...\}$ is a sequence of random variables, which are defined on the same probability space $\{\Omega, \mathcal{F}, \mathbf{P}\}$.

1. $X_n \xrightarrow{P} X$ (in probability) if $\forall \epsilon > 0 \lim_{n\to\infty} \mathbf{P}(|X_n - X| > \epsilon) = 0$,

2. $X_n \xrightarrow{m.s.} X$ (in mean square) if $\lim_{n\to\infty} E\{|X_n - X|^2\} = 0$,

3. $X_n \xrightarrow{a.s.} X$ (almost sure) if $\mathbf{P}\left(w \in \Omega : \lim_{n\to\infty} X_n(w) = X(w)\right) = 1$,

4. $X_n \xrightarrow{d} X$ (in distribution or weakly) if $\lim_{n\to\infty} F_n(x) = F(x)$ for every $x \in \mathbf{R}$ at which $F(x)$ is continuous. Here $F_n(x)$ and F are the cumulative distribution functions (CDF) of the random variables X_n and X respectively.

Let X_n is a sequence of random variables distributed as follows: $\mathbf{P}(X_n = 0) = 1 - \frac{1}{n}$, $\mathbf{P}(X_n = 1) = \frac{1}{n}$. This sequence converges in mean-square and therefore in probability but does not converges almost sure. [26]

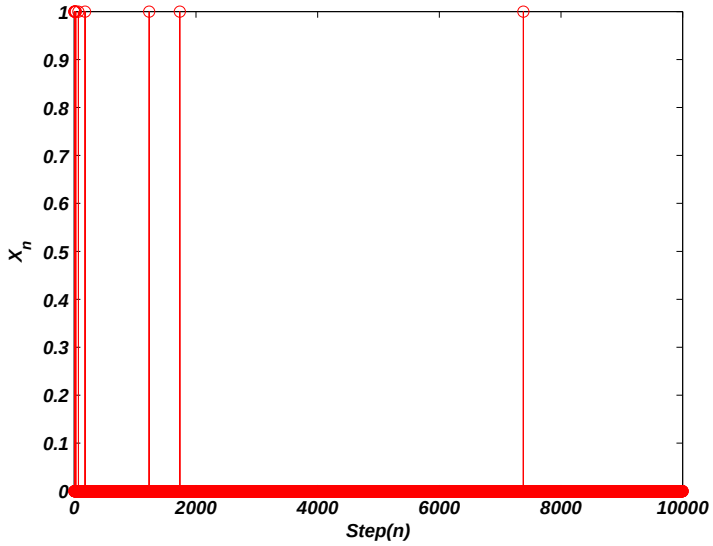

Figure 2: Convergence of the random variables in mean-square.

Let X_n is a sequence of random variables distributed as follows: $\mathbf{P}(X_n = 0) = 1 - \frac{1}{n^2}$, $\mathbf{P}(X_n = 1) = \frac{1}{n^2}$. This sequence converges almost sure and therefore in probability but does not converges in mean-square. [26]

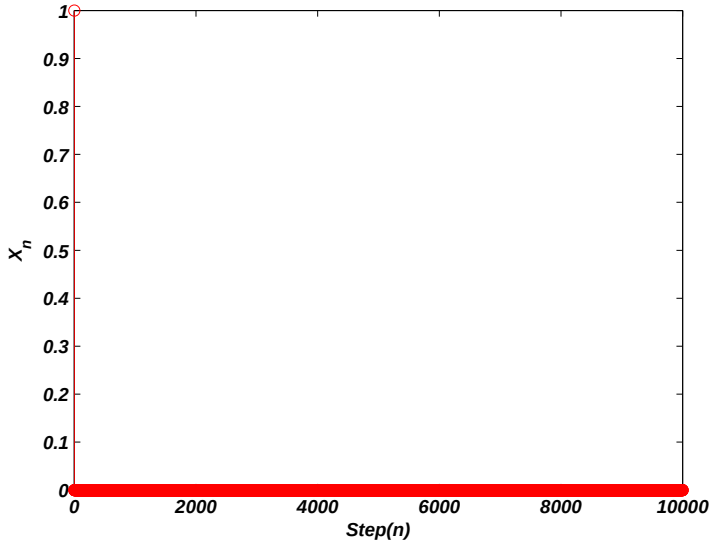

Figure 3: Convergence of the random variables almost sure.

As it can be seen from the figures, in the first case even after 5000 steps, there can be outliers from the accumulation point, but in the case of the convergence almost sure it

does not happen, as for any $w \in \Omega$ we cannot have values beyond the arbitrary given epsilon-neighborhood of the limited point.

3.5 Stochastic quasi-gradient method

Having understanding of the foregoing concepts we will proceed describing the stochastic quasi-gradient algorithm for the optimization problems with mean-value criterion, which is formulated below.

$$\begin{gathered} \min_{u \in U} F(u), \\ F(u) = E\{\Phi(u, X)\}. \end{gathered} \tag{29}$$

Here $\Phi(u, X)$ as before denotes an objective function dependent on a random vector X and the strategy u which should be chosen from the set $U \subseteq \mathbf{R}^n$.

If the gradient $\nabla F(u)$ was known we could use the classical gradient-descent method, starting from some initial point u^0. The main statement of the algorithm is written below:

$$u^{\nu+1} = \Pi_U \left(u^\nu - \rho_\nu \nabla F(u^\nu)\right), \tag{30}$$

where $\Pi_U(u)$ denotes a projection operator onto the set U:

$$\Pi_U(u) = \arg\min_{y \in U} ||u - y||. \tag{31}$$

The projection operator are used here to restrict the location of the points u_0,...,u_n generated by the algorithm to the admissible area U.

The problem to find a projection of the point onto the set is not always an easy task. Generally the set U has to be closed and convex in order to have a unique projection (it might happen that the projection does not exists at all if the set U is open and the projection might be non-unique in the case of non-convex U).

The projection can be found instantly for the elementary types of the set U. Such cases are listed below.

1. A non-negative orthant:

$$U = E_+^n = \{u : u \in \mathbf{R}^n, u_j \geq 0, j = 1, .., n\}. \tag{32}$$

Let us denote p_j the j-th coordinate of the projection $\Pi_U(u)$, then

$$p_j = \begin{cases} 0, & u_j < 0, \\ u_j, & u_j \geq 0. \end{cases} \tag{33}$$

2. n-dimensional parallelepiped:

$$U = \{u : u \in \mathbf{R}^n, a_j \leq u_j \leq b_j, j = 1, .., n\} \tag{34}$$

$$p_j = \begin{cases} a_j, & u_j < a_j, \\ u_j, & a_j \leq u_j \leq b_j, \\ b_j, & u_j > b_j. \end{cases} \tag{35}$$

where p_j as usuall the j-th coordinate of the projection $\Pi_U(u)$.

3. A sphere with the radius r:

$$U = \{u : u \in \mathbf{R}^n, ||u|| \leq r\}. \tag{36}$$

The projection can be found as follows:

$$\Pi_U(u) = \begin{cases} u, & u \in U, \\ \frac{r}{||u||}u, & u \notin U, \end{cases} \tag{37}$$

4. A hyperplane:

$$U = \{u : u \in \mathbf{R}^n, \langle c, y\rangle = b\}, \tag{38}$$

where $c \in E^n$, $b \in R$.

$$\Pi_U(u) = u + \frac{b - \langle c, u\rangle}{||c||^2}c. \tag{39}$$

5. A halfspace:

$$U = \{u : u \in \mathbf{R}^n, \langle c, y\rangle \leq b\}, \tag{40}$$

$$\Pi_U(u) = \begin{cases} u, & u \in U, \\ u + \frac{b - \langle c, u\rangle}{||c||^2}c, & u \notin U, \end{cases} \tag{41}$$

Considering the remark regarding projection operator a gradient descent algorithm can be writtern:

1. Initialize the step $\nu = 0$, choose the initial point u^0.

2. Compute the next point $u^{\nu+1}$ using the previous point u^{ν} and an equation:

$$u^{\nu+1} = \Pi_U \left(u^{\nu} - \rho_{\nu} \nabla F(u^{\nu}) \right), \tag{42}$$

   To find a projection onto the set U use above mentioned remarks regarding the projection operator.

3. Check the stopping criterion. If the stopping condition is satisfied, put $u^* = u^{\nu+1}$, otherwise assign $\nu = \nu + 1$ and go the step 2.

Let $f(u)$ is a function, defined on the set $U \subset \mathbf{R}^n$. The vector $\partial(u^0) \subset \mathbf{R}^n$ is called a subgradient of the function $f(u)$ at the point $u^0 \in U$ if $\forall u \in U$

$$f(u) \geq f(u^0) + \langle \partial(u^0), u - u^0 \rangle. \tag{43}$$

The formula above geometrically means that the graph of the function $f(u)$ located above the linear function $f(u^0) + \langle \partial(u^0), u - u^0 \rangle$ and at the point u^0 the graphs coincide.

Obviously if the function is differentiable there is only one such vector $\partial(u^0)$ equals to the gradient $\nabla f(u^0)$. For a non-differentiable function at the point (u^0) there exists a set of the subgradients, that is called subdifferential set.

From the point of view of optimization methods, if the objective function $F(u)$ is not differentiable, the gradient $\nabla F(u)$ is substituted by the subgradient $\partial F(u)$ and the rest of the gradient descent method is used without any changes. Such method is called subgradient algorithm.

In case of the stochastic objective function an exact gradient is unknown, and therefore its stochastic version is used:

$$u^{\nu+1} = \Pi_U(u^{\nu} - \rho_{\nu} \nabla_u \Phi(u^{\nu}, x^{\nu})), \tag{44}$$

where $x^{\nu}, (\nu = 1, ...)$ is a realization of X at the step ν of the algorithm. So that x^{ν} is a random variable, because we do not know the value which assumes X at the step ν. Moreover the point $u^{\nu}, (\nu = 1, ...)$ is also a random variable, because it depends on x^{ν} and the previous point $u^{\nu-1}$.

Finally note that an expression (44) describes a random process starting from the initial point u^0 and under some further formulated conditions should lead us to the optimal value of the optimization problem (29).

A detailed description of the stochastic gradient method looks as shown below.

1. Initialize the step $\nu = 0$, choose the initial point u^0.

2. Generate a realization x^ν of the random vector X. Compute a stochastic gradient $\nabla_u \Phi(u^\nu, x^\nu)$ using the previous point u^ν and the realization x^ν.

3. Calculate the next point $u^{\nu+1}$ using the previous point u^ν and an equation:

$$u^{\nu+1} = \Pi_U(u^\nu - \rho_\nu \nabla_u \Phi(u^\nu, x^\nu)), \tag{45}$$

   To find a projection onto the set U use the mentioned remarks regarding the projection operator.

4. Check the stopping criterion. If the stopping condition is satisfied, put $u^* = u^{\nu+1}$, otherwise assign $\nu = \nu + 1$ and go the step 2.

If the function $\Phi(u^\nu, x^\nu)$ is not differentiable with respect to variable u a stochastic gradient is substituted by the quasi-gradient:

$$E\left[\zeta^\nu \,|\, u^0, ..., u^{\nu-1}\right] \in \partial_u E\left[\Phi(u^\nu, X)\right]. \tag{46}$$

Intuitively the above mentioned statement means that the conditional expectation of the stochastic quasi-gradient at the point u^ν with respect to all previous points should coincide almost sure (a.s.) with one of the subgradients of the objective function at the point u^ν (i.e. belonging to the subdifferential set). And the main statement of the stochastic quasi-gradient method looks as follows:

$$u^{\nu+1} = \Pi_U(u^\nu - \rho_\nu \zeta^\nu). \tag{47}$$

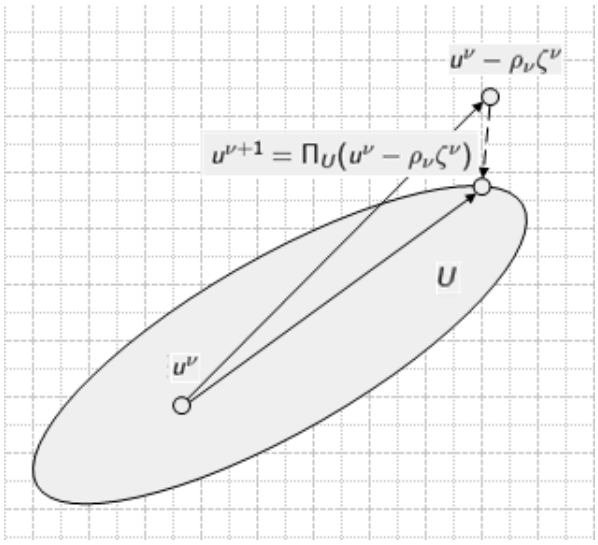

Figure 4: Illustration of the SQG-algorithm.

When we are dealing with a deterministic problem the step size $\rho_{\nu+1}$ can be obtained from the solution of a one-dimensional optimization problem:

$$\rho_{\nu+1} = \min_{\rho}\{F(u^{\nu} - \rho\nabla_u F(u^{\nu}))\}, \tag{48}$$

meaning that in every step we should move towards the gradient direction to minimize the objective function as much as possible.

In the stochastic version there are several problems which do not allow to use above shown technique. Firstly we do not know exactly an expected value of $\Phi(u, X)$ for the given u (we know only $\Phi(u, x)$). Secondly the stochastic gradient at some steps can even give a wrong direction, so any positive value ρ leads to the worse point than previous one.

The next figures show the classical gradient descent method and the stochastic version. In the first case we explicitly can find the gradient and computing an optimal step size reach an optimum with 1 step. In the second case we only observe noisy data and get an optimum of the mean-value criterion applying stochastic gradient method, what is of cause requires more steps.

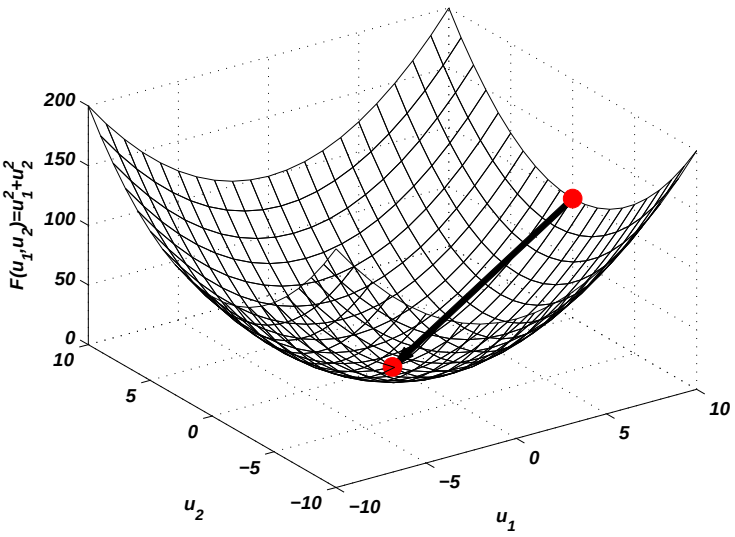

Figure 5: Illustration of the classical gradient descent method.

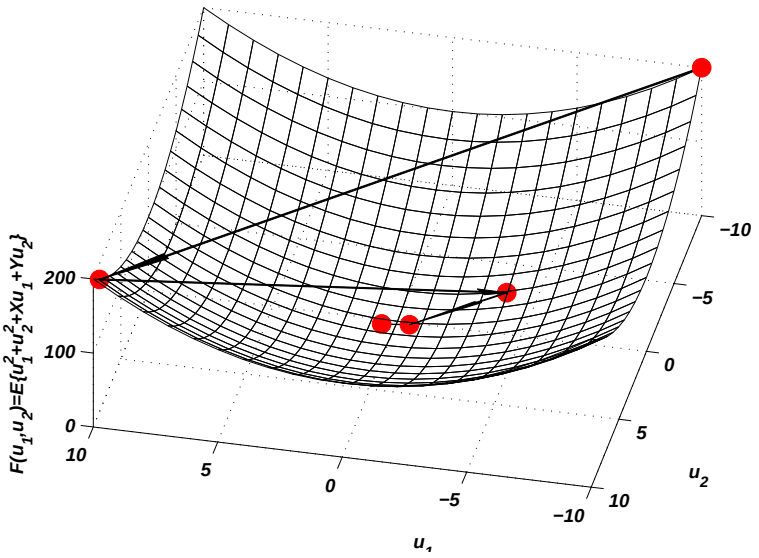

Figure 6: Illustration of the stochastic gradient method.

To proof the convergence a.s. (47) of the algorithm we have to make several assumptions [11]:

- the function $E\left[\Phi(u^\nu, X)\right]$ is a convex finite-valued function;
- the set U is a convex and compact;
- the parameters ρ and γ satisfy a.s.:

$$\rho_\nu \geq 0, \ \sum_{\nu=0}^{\infty} \rho_\nu = \infty, \ \sum_{\nu=0}^{\infty} E\left[\rho_\nu |\gamma_\nu| + \rho_\nu^2 \|\zeta^\nu\|^2\right] < \infty. \tag{49}$$

The first two conditions is classical requirements for an existence and a uniqueness of the solution in convex optimization. The last two requirements mostly affect the sequence of the step size. The sequence has to be very special: it should converge, but not so fast so that the point generated by the algorithm will be able to reach an optimum independently of the initial point where the algorithm starts.

In order to highlight the main steps of the proof [12], let us assume the following notations:
$u^* \in U$ is an optimal solution of the problem (29),
$\mathcal{F}^\nu = \sigma\{u^0, ..., u^\nu\}$ - a sigma-algebra, generated by the random variables $\{u^0, ..., u^\nu\}$.

The main idea of the proof is to show that under conditions of the theorem $E\left[\|u^{\nu+1}-u^*\| \ \middle|\mathcal{F}_\nu\right]$ is a non-negative supermartingale, therefore a.s. it has some limit and then to proof that the limit is 0, meaning $u^\nu \to u^*$ a.s.

- Projection implyes:
  $\|u^{\nu+1}-u^*\|^2 \le \|u^\nu-u^*\|^2 - 2\rho_\nu\langle\zeta^\nu, u^\nu-u^*\rangle + \rho_\nu^2\|\zeta^\nu\|^2$.
- Take conditional expectation w.r.t. $\mathcal{F}_\nu$:
  $E\left[\|u^{\nu+1}-u^*\|^2 \ \middle|\mathcal{F}_\nu\right] \le \|u^\nu-u^*\|^2 - 2\rho_\nu\langle E\left[\zeta^\nu \ \middle|\mathcal{F}_\nu\right], u^\nu-u^*\rangle + \rho_\nu^2 E\left[\|\zeta^\nu\|^2 \ \middle|\mathcal{F}_\nu\right]$.
- By the conditions of the theorem:
  $\gamma_\nu + \left\langle E\left[\zeta^\nu \ \middle|\mathcal{F}_\nu\right], u^*-u^\nu\right\rangle \ge E\left[\Phi(u^*,X)\right] - E\left[\Phi(u^\nu,X)\right] \ge 0 \Longrightarrow$
  $\left\langle E\left[\zeta^\nu \ \middle|\mathcal{F}_\nu\right], u^*-u^\nu\right\rangle \ge -\gamma_\nu$.
- Two previous steps together results in:
  $E\left[\|u^{\nu+1}-u^*\|^2 \ \middle|\mathcal{F}_\nu\right] \le \|u^\nu-u^*\|^2 + 2\rho_\nu\gamma_\nu + \rho_\nu^2 E\left[\|\zeta^\nu\|^2 \ \middle|\mathcal{F}_\nu\right] \le$
  $\|u^\nu-u^*\|^2 + \underbrace{2\rho_\nu|\gamma_\nu| + \rho_\nu^2 E\left[\|\zeta^\nu\|^2 \ \middle|\mathcal{F}_\nu\right]}_{R_\nu} = \|u^\nu-u^*\|^2 + R_\nu$.
- Assume $Y^\nu = \|u^*-u^\nu\|^2 + \sum\limits_{k=0}^{\infty} R_k$, hence:
  $E\left[Y^{\nu+1} \ \middle|\mathcal{F}_\nu\right] \le Y^\nu$, $Y^\nu \ge 0 \Longrightarrow Y^\nu \to Y$ (a.s.) - convergence theorem for supermartingales.
- Recursively, taking the full expectation:
  $E\left[\|u^{\nu+1}-u^*\|^2\right] \le \|u^0-u^*\|^2 + \sum\limits_{k=0}^{\nu}\rho_k^2 E\left[\|\zeta^k\|^2\right] - 2\sum\limits_{k=0}^{\nu}\rho_k\langle E\left[\zeta^k\right], u^k-u^*\rangle \Longrightarrow$
  $E\left[\|u^{\nu+1}-u^*\|^2\right] - E\left[\|u^0-u^*\|^2\right] - \sum\limits_{k=0}^{\nu} E\left[R_k\right] \le 2\sum\limits_{k=0}^{\nu}\rho_k\left(E\left[\Phi(u^*,X)\right] - E\left[\Phi(u^k,X)\right]\right)$.
- Considering conditions of the theorem:
  $\sum\limits_{k=0}^{\infty} E\left[R_k\right] < \infty$ and $E\left[\Phi(u^*,X)\right] < E\left[\Phi(u^k,X)\right]$ we have:
  $-\infty < \sum\limits_{k=0}^{\infty}\rho_k\left(E\left[\Phi(u^*,X)\right] - E\left[\Phi(u^k,X)\right]\right) \le 0$, therefore
  $E\left[\Phi(u^k,X)\right] \to E\left[\Phi(u^*,X)\right]$.

3.6 Kiefer-Wolfowitz algorithm

If the subgradient $\partial_u\Phi(u,x)$ or even the gradient $\nabla_u\Phi(u,x)$ of the function $\Phi(u,x)$ w.r.t. variable u can be explicitly found, then in every step of the SQG-algorithm $\partial_u\Phi(u,x)$

or $\nabla_u \Phi(u, x)$ is usually used as a quasi-gradient estimate, otherwise a finite-difference approximation is employed.

$$\zeta^k = \frac{1}{2\delta_k} \sum_{j=1}^{n} \left[\Phi(u^k + \delta_k e_j, x^k) - \Phi(u^k - \delta_k e_j, x^k) \right] e_j. \tag{50}$$

where x^k is as usual a realization of the random vector X at the step k of the algorithm, e_j, $j = 1, ..., n$ are the unit vectors directed along the coordinate axes.

Considering the formula above the method of Kiefer and Wolfowitz assumes the following form:

1. Initialize the step $k = 0$, choose the initial point u^0.
2. Generate a realization x^k of the random vector X. Compute a stochastic quasi-gradient ζ^k via formula (50) using the previous point u^k and the realization x^k.
3. Calculate the next point u^{k+1} using the previous point u^k according to the formula:
$$u^{k+1} = \Pi_U(u^k - \rho_k \zeta^k), \tag{51}$$
To find a projection onto the set U use the mentioned remarks regarding the projection operator.
4. Check the stopping criterion. If the stopping condition is satisfied, put $u^* = u^{k+1}$, otherwise assign $k = k + 1$ and go the step 2 .

As it can be seen the algorithm requires $2n$ evaluations of the objective function $\Phi(u, X)$ in every step of the method, where n is a dimension of the control varaible u.

The convergence theorem of Kiefer-Wolfowits [17] and its generalization can be found in [4, 9].

Theorem 3.2. *If the below conditions are valid:*

- *the function $f(u) = E[\Phi(u, X)]$ has only extremum $u^* \in int\{U\}$;*
- *the second derivatives of $f(u)$ are continuous and bounded;*
- *the variance $D[\Phi(u, X)] \leq C < \infty$ for all $u \in U$;*
- *the sequences ρ_k and δ_k satisfy*
$$\rho_k \geq 0, \ \sum_{k=1}^{\infty} \rho_k = \infty, \ \sum_{k=1}^{\infty} \left(\frac{\rho_k}{\delta_k} \right)^2 < \infty, \ \sum_{k=1}^{\infty} \rho_k |\delta_k| < \infty. \tag{52}$$

then the sequence generated by Kiefer-Wolfowits algorithm converges to the optimal solution of the mean-value problem almost sure ($u^k \to u^*$ a.s.).

3.7 Recipes for step sizes

As it was mentioned the step size is a very important feature directly affecting performance of the method. As it often happens the theoretical results regarding convergence of the method do not suggest the way to increase performance and therefore empirical schemes are used. An opposite side of such process is that adaptive algorithms should be theoretically justified otherwise there exist a risk to face the special case when an inaccurate adaptive method fails. As an example: a stochastic approximation adaptive rule for the step size have to satisfy step size conditions to guarantee convergence otherwise all the proof must be revised.

Additionally adaptation should not significantly increase complexity of the original method and preferably do not have a lot of tuning parameters to be adjusted. Despite this challenges several successful results regarding adaptive stochastic approximation are known.

3.7.1 Kesten Rule

The first idea to use adaptive step size in stochastic approximation belongs to Kesten. His method based on the simple idea that if we are far from the optimum, the errors tend to have the same sign but when we are getting closer, the errors start alternate. Under the error here it is assumed the difference between the previous u^{n-1} and the current u^n point. Kesten suggested the following rule:

$$\rho_{n-1} = \frac{a}{a + K^n - 1}, \tag{53}$$

where a is a parameter to be calibrated. K^n is a parameter, which counts the number of times that the sign of the errors have been changed.

$$K^n = \begin{cases} n & n = 1, 2, \\ K^{n-1} + 1_{\{\langle \epsilon^n, \epsilon^{n-1} \rangle < 0\}} & n > 2, \end{cases} \tag{54}$$

where

$$\epsilon_n = u^{n-1} - u^n. \tag{55}$$

To asses the performance of the adaptive method let us have a look at the following example:

$$\min_{u_1, u_2 \in [-10;10]} E\left\{u_1^2 + u_2^2 + max(u_1^2, u_2^2) + u_1 X\right\}, X \in N(0,1). \tag{56}$$

Obviously, the optimum of the function is $u_1^* = 0,\ u_2^* = 0$, because $\forall u_1, u_2\ u_1^2 + u_2^2 + max(u_1^2, u_2^2) \geq 0$ and all terms $u_1^2 \geq 0, u_2^2 \geq 0, max(u_1^2, u_2^2) \geq 0$.

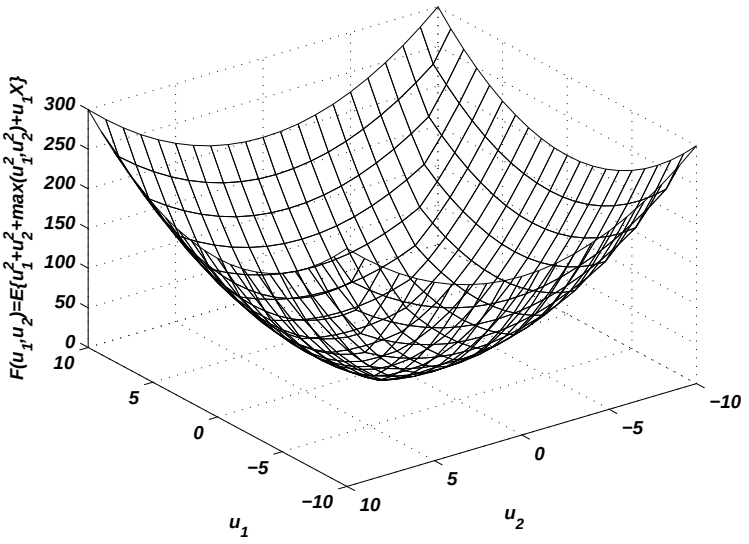

Figure 7: Optimization function.

The stochastic gradient can be written explicitly in the form:

$$\xi(u, X) = \begin{bmatrix} 2u_1 + 2u_1 I_{u_1^2 > u_2^2} + X \\ 2u_2 + 2u_2 I_{u_1^2 \leq u_2^2} \end{bmatrix}. \tag{57}$$

Let us choose the step size ρ_k according to the rule: $\rho_k = 0.1 \cdot \frac{1}{k}$. In the standard case the steps will be too small, and the adaptive rule will preserve the step size from the rapid drop enhancing convergence.

On the next figures we can see how the control variables where changing towards the optimum over the steps. Initial value of the step size were chosen the same for adaptive and non-adaptive methods. In fact non-adaptive algorithms even after 1000 iterations were relatively far from the optimum, while the adaptive schemes were able to rich the steady state after 100 steps.

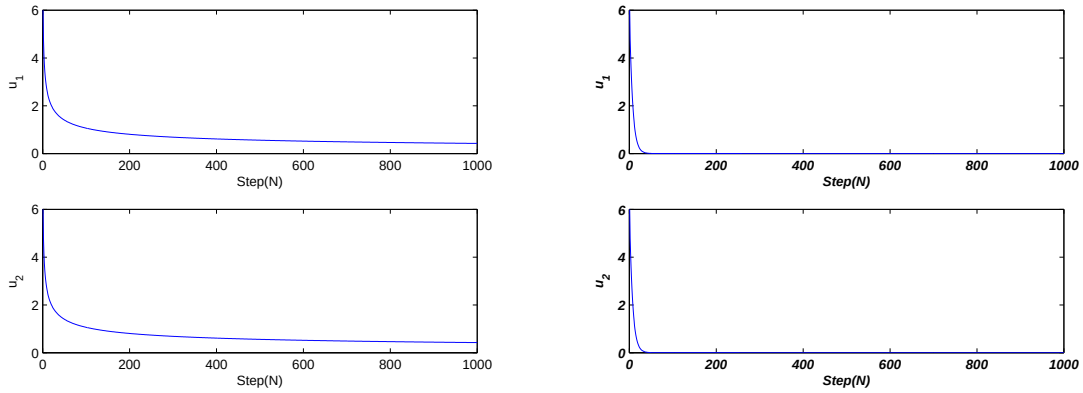

Figure 8: Small step size. Stochastic Gradient Descent (left). Stochastic Gradient Descent with adaptive Kesten rule (right).

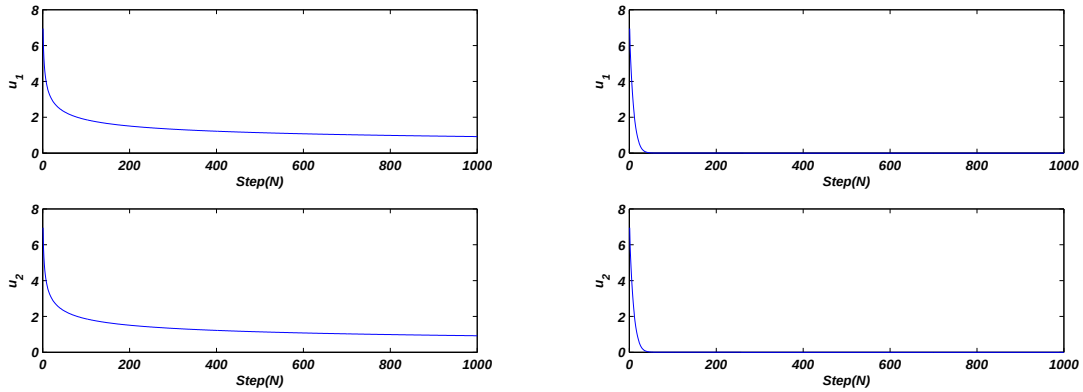

Figure 9: Kiefer-Wolfowitz algorithm (left). Kiefer-Wolfowitz algorithm with adaptive Kesten rule (right).

As it can be seen the test results agree with the theoretical reasoning.

Now let us test the adaptive rule for the large step lengths on the same test example, so $\rho_k = 10 \cdot \frac{1}{k}$. At the beginning when k is relatively small, the steps will be too large and the points, generated by the algorithms will try to leave the admissible area U but due to the projection operator they will be returned to the border of the set U. When the step size assumes the reasonable values, the procedures will start converge to the optimum. Here the adaptive rule does not enhance convergence as it does not decrease the steps faster than in standard versions.

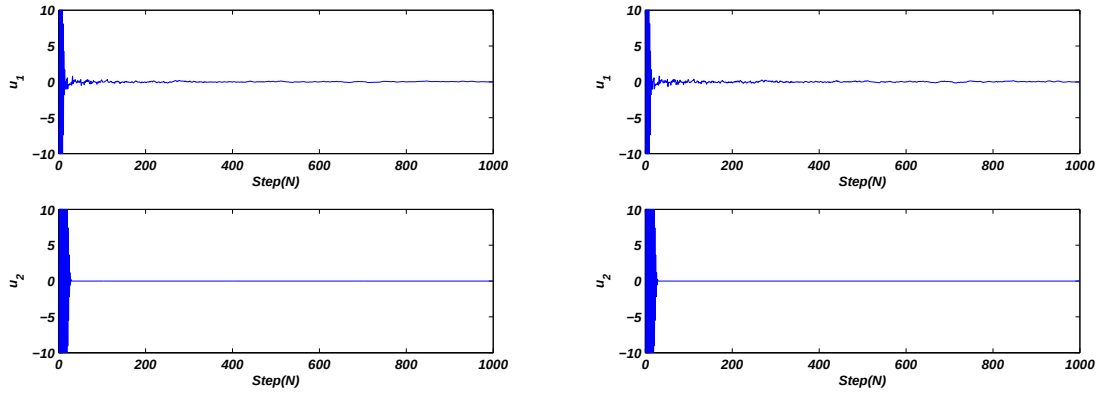

Figure 10: Stochastic Gradient Descent (left). Stochastic Gradient Descent with adaptive Kesten rule (right).

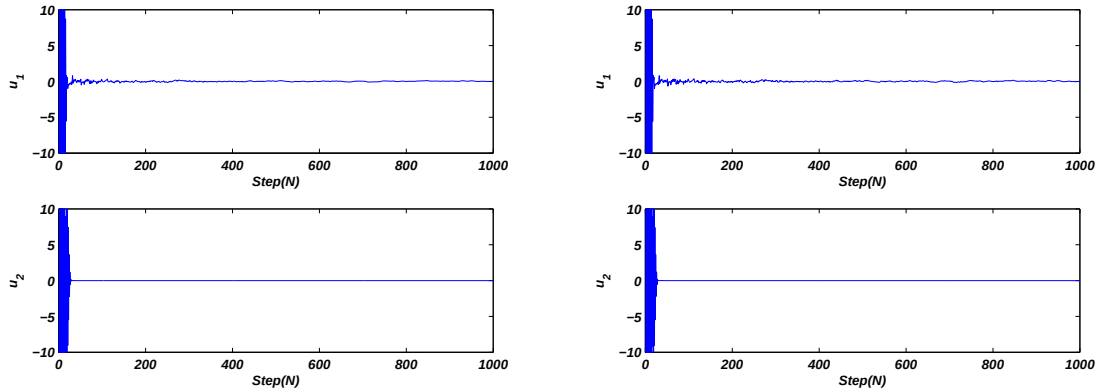

Figure 11: Kiefer-Wolfowitz algorithm (left). Kiefer-Wolfowitz algorithm with adaptive Kesten rule (right).

The test examples again agree with theoretical reasoning. This example also demonstrates the behaviour of the stochastic approximation methods: the convergence traces w.r.t. coordinate u_2 are smoother, then w.r.t to u_1 as a stochasticity mostly affects the first coordinate due to the term $u_1 X$ in the objective function.

Finally regarding Kesten rule we can conclude that it reasonable to use for the relatively small or middle step length as it combines constant step size rule (when we are far from the optimum) with the usual step size rule (when we are close to the end).

3.7.2 Adaptive rule of Uryasev

Uryasev [28] suggested more advanced adaptive rule which can decrease the steps as well as increase them depending on the generated sequence of points:

$$\begin{aligned} &\rho_{k+1} = \min(\bar{\rho}, \rho_k a^{-\langle \xi^{k+1}, \Delta^{k+1}\rangle - \delta\rho_k}), \\ &\Delta^{k+1} = u^{k+1} - u^k, \\ &a_k > 1,\ \delta > 0, \end{aligned} \tag{58}$$

To understand the meaning of the terms in the mentioned above rule let us have a look at the main formula of SQG-algorithm:

$$u_{k+1} = u_k - \rho_k \xi^k. \tag{59}$$

It would be logical to choose the step, that minimizes the function $F_k(\rho)$ w.r.t ρ:

$$F_k(\rho) = E[\Phi(u_k - \rho\xi^k) \,|\, \mathcal{F}^k]. \tag{60}$$

A computation of $F_k(\rho)$ is a very difficult task, therefore let us differentiate $\Phi(u_k - \rho\xi^k)$ w.r.t ρ at the point ρ_k.

$$\partial_\rho \Phi(u_k - \rho\xi^k)|_{\rho_k} = -\langle \xi^k, \partial\Phi(u_k - \rho_k\xi^k)\rangle = -\langle \xi^k, \xi^{k+1}\rangle. \tag{61}$$

Hence $-E[\langle \xi^k, \xi^{k+1}\rangle \,|\, \mathcal{F}^k] \in \partial F_k(\rho_k)$ and the following gradient procedure can be used to modify the step size:

$$\rho_{k+1} = \rho_k + \lambda_k \langle \xi^k, \xi^{k+1}\rangle. \tag{62}$$

The adaptive procedure (58) directly comes from the previous formula and works as follows: the term $\langle \xi^{k+1}, \Delta^{k+1}\rangle$ gives information wether or not the minimum of $F_k(\rho)$ w.r.t ρ has been reached. If $-\langle \xi^{k+1}, \Delta^{k+1}\rangle > 0$ then with high probability we can say that the minimum has not been achieved yet and the step ρ_k will be increased, otherwise it decreases.

Let us test this adaptive rule on the previous model example.

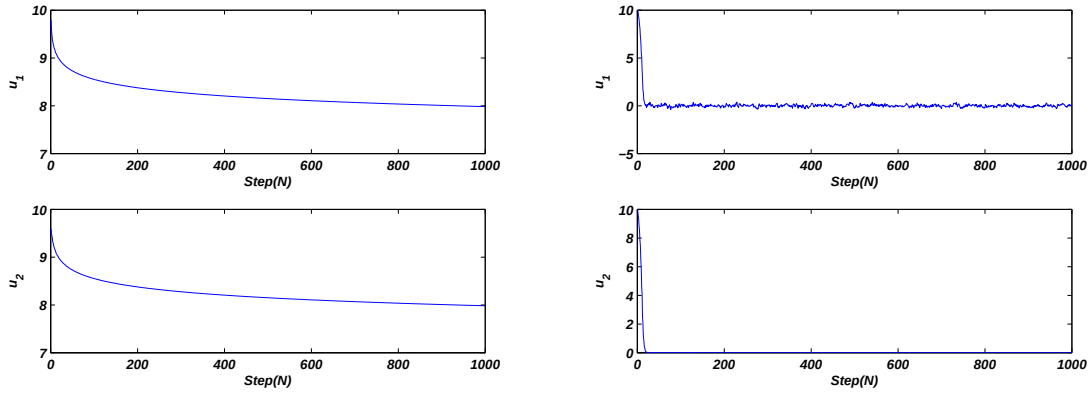

Figure 12: Stochastic Gradient Descent (left). Stochastic Gradient Descent with adaptive Uryasev rule (right).

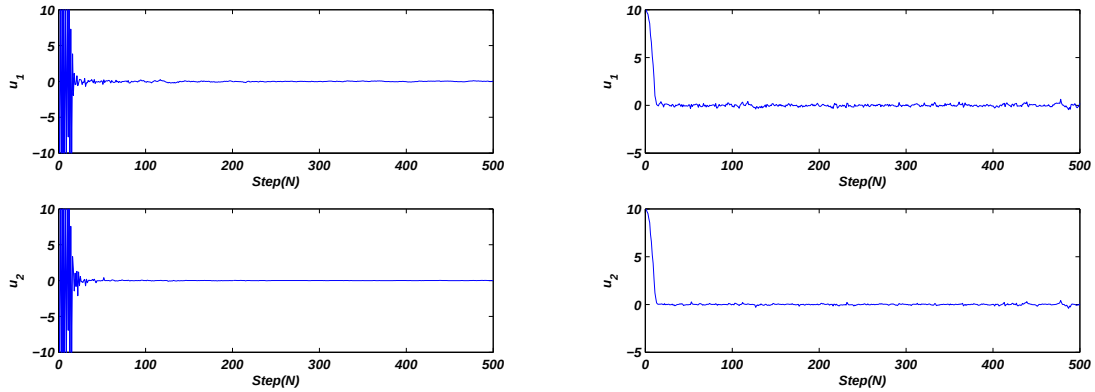

Figure 13: Kiefer-Wolfowitz algorithm (left). Kiefer-Wolfowitz algorithm with adaptive Uryasev rule (right).

It is worth to note, the adaptive rule of Uryasev is a very sensitive to the parameters Δ and a, and their tuning brings an additional complexity to the algorithm.

3.8 Simultaneous perturbation stochastic approximation

One special branch of the stochastic quasi-gradient method is simultaneous perturbation stochastic approximation (SPSA) algorithms. Generally speaking stochastic approximation procedures used for the optimization of the cost-functions differs mostly in a way how the gradient estimate is calculated.

The SPSA-algorithms use artificially generated random noise independent of the unknown random parameters therefore is called simultaneous perturbations. By virtue of such approach, methods require only 2 measurements of the objective function for the quasi-gradient estimation regardless of the dimension size, and under general conditions, the SPSA and the standard finite-difference stochastic approximation algorithm (i.e. Kiefer-Wolfowitz algorithm) achieve the same level of statistical accuracy [19]. The method look as follows:

1. Initialize the step $k = 0$, choose the initial point u^0.

2. Generate a realization x^k of the random vector X. Generate a vector Δ^k distributed according to the Bernoulli law ± 1 with the probability 0.5 for every outcome. The dimension of the vector Δ^k is n, so that

$$\Delta^k = \begin{pmatrix} \Delta_1^k \\ \dots \\ \Delta_n^k \end{pmatrix}. \tag{63}$$

3. Compute a stochastic quasi-gradient using the previous point u^k and the realizations x^k, Δ^k according to the formula:

$$g_k(u^k, x^k) = \begin{pmatrix} \frac{\Phi(u^k + c_k\Delta^k, x^k) - \Phi(u^k - c^k\Delta^k, x^k}{2c_k\Delta_1^k}) \\ \vdots \\ \frac{\Phi(u^k + c_k\Delta^k, x^k) - \Phi(u^k - c^k\Delta^k, x^k)}{2c_k\Delta_n^k} \end{pmatrix}, \tag{64}$$

where Δ_k is a random perturbation vector, independent of X and c_k is a decreasing non-random sequence:

$$c_k = \frac{1}{k^{-1}}, \tag{65}$$

4. Calculate the next point u^{k+1} using the previous point u^k according to the formula:

$$u^{k+1} = \Pi_U(u^k - a_k \cdot g_k(u^k, x^k)), \tag{66}$$

where

$$a_k = \frac{1}{(A + k)^{-1}}. \tag{67}$$

To find a projection onto the set U use the mentioned remarks regarding the projection operator.

5. Check the stopping criterion. If the stopping condition is satisfied, put $u^* = u^{k+1}$, otherwise assign $k = k + 1$ and go the step 2 .

The random perturbation vector Δ^k is used for the computation of $u^+ = u^k + c_k\Delta^k$ and $u^- = u^k - c_k\Delta^k$.

The values u^+ and u^- are then used to calculate the quasi-gradient estimate: g_k, that follows by the standard stochastic approximation step where the next point u^{k+1} is obtained.

As it can be seen from the remark above in the step k the method requires only 2 evaluations of the objective function to calculate u^+ and u^-.

On the next figure the comparison of the classical stochastic quasi-gradient method and SPSA is shown. The calculations are done on the basis of the previous test-example (a contour plot of the objective function are shown on the picture as well).

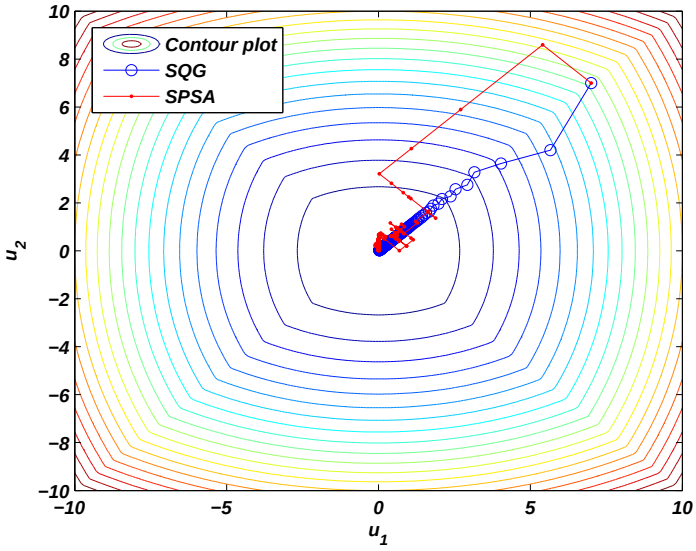

Figure 14: Comparison of the SQG and SPSA convergence traces.

Both methods started from the same initial point $u = (7, 7)^T$ and after 1000 steps successfully reached an optimum $u = (0, 0)^T$. The sequence of the points, produced by the classical method looks more smooth and predictable and one can hardly say that the method connected with randomness are used here, while the trace of SPSA-algorithm is definitely stochastic due to artificial perturbations incorporated in Δ^k.

The following figure shows how the values of the control variables u_1 and u_2 were changing over the steps. Stochastic nature of the method is clearly seen from here as well.

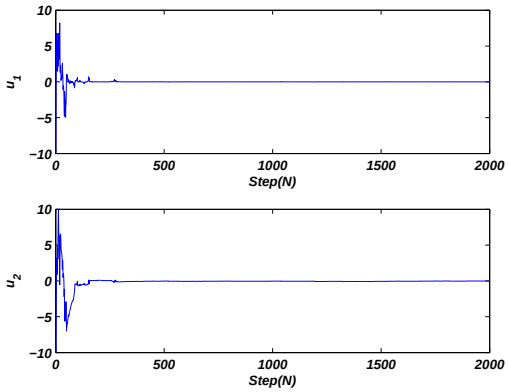

Figure 15: Illustration of the SPSA convergence.

The main advantage of the method in comparison with the algorithm of Kiefer and Wolfowitz is a constant number of function evaluations (only 2) needed in every step of the algorithm, independently of the optimization problem's dimension. This fact might be crucial when the objective function has a complicated structure (see the last section where the speed of computation is compared on the model example).

4 Application problem: Electricity Retailer Profit Optimization

Assume that an energy retailer is needed to supply electricity which he buys from the global supplier (electricity market) to its customers. Energy consumption even during a short term can not be exactly predicted, therefore possible income of the retailer contains uncertainty. Moreover electricity market does not imply that the prices for the energy will be fixed even during a short period of time.

Electricity nature does not allow to store its relatively large amount like for instance liquid, hence every global supplier (electricity market) introduces penalties for the positive as well as for the negative imbalance what are correspondingly positive and negative difference between the purchased and consumed energy [1]. Large penalties for the imbalance force the retailer to choose a cautious and less profitable strategy rather than risky one.

In this section we consider a simple stochastic model reflecting retailer's profit optimization with mean value and CVaR criteria and demonstrate how to get deterministic equivalents in the form of linear programming (LP) problems. Also we compare stochastic approximation solution with the mentioned above deterministic equivalents.

A possibility to obtain deterministic equivalent in a simple form is a distinguishing feature of this work. There are many stochastic models describing similar optimization processes, but the majority of them are solved by decision tree [2] or stochastic qvasigradient method [3], what is undoubtedly much slowly then the solution of a well-known deterministic task.

The model and solution technique can be used not only for the electricity retailer optimization, but with suitable modifications for any retailing (reselling) processes.

4.1 Model building

Let us introduce the following notations:
c_s - the price of sold electricity from the retailer to the consumer;
X - the random demand of energy by the consumer (electricity sales);
Y - the random price for the purchased electricity from the supplier;
c_d - the price of the additional purchased electricity from the supplier in case when the random demand is more than amount of the purchased energy (a price for the negative imbalance);
c_u - the price for the positive imbalance (when the consumer's demand is less than

amount of the purchased energy by the retailer);
u - an amount of energy which retailer buys in order to subsequently resell it to the consumer, $(u \geq 0)$;

In the notations above the retailer profit is affected by the terms:
$c_s \cdot \min\{X, u\}$ - the benefits from the sold energy to the consumer;
$Y \cdot u$ - the payment for the purchased energy from the global supplier;
$c_d \cdot \max\{X - u, 0\}$ - the losses for the negative imbalance;
$c_u \cdot \max\{u - X, 0\}$ - the losses for the positive imbalance.

Subsequently the random income of the retailer can be expressed in the following form:

$$\Phi(u, X, Y) = c_s \cdot \min\{X, u\} - Y \cdot u - c_d \cdot \max\{X - u, 0\} - c_u \cdot \max\{u - X, 0\}. \quad (68)$$

In order to restrict the negative and the positive imbalance let us introduce two probability constraints:
Δ_1 - a threshold for the positive imbalance. If the difference between the offer u and the demand X is more than the positive value Δ_1 then the retailer is heavily penalized.
β - the confident level for the positive imbalance. Retailer is interested to trade without violation of the defined by the supplier threshold with probability β.
Δ_2 and γ - a threshold for the negative imbalance and the corresponding confident level are defined similarly.

The probability constraint for the positive imbalance looks as follows:

$$P\{X - u \leq \Delta_1\} \geq \beta. \quad (69)$$

The probability constraint for the negative imbalance:

$$P\{u - X \leq \Delta_2\} \geq \gamma. \quad (70)$$

The confident levels β and γ define how wide will be an admissible area established by the corresponding probability constraint. There is no exact technique to determine β and γ, but one can use the following idea.

Assume that we have solved the problem for some parameters β and γ, and got an estimate for the profit $\phi^*(\beta, \gamma)$. Interpreting probability as a frequency we have that in $(1 - \beta - \gamma)$ cases our retailer got $(1 - \beta - \gamma)\phi^*(\beta, \gamma)$, but lost βC_β in β cases and γC_γ in γ cases. Introducing the constant $C \geq 0$ given by an expert to get an equality between the total profit and the total loss it is possible to tune confident levels β and γ:

$$C(1 - \beta - \gamma)\phi^*(\beta, \gamma) = \beta C_\beta + \gamma C_\gamma. \quad (71)$$

Note that in this case the initial problem have to be solved parametrically as an optimal solution $\phi^*(\beta, \gamma)$ depends on β and γ.

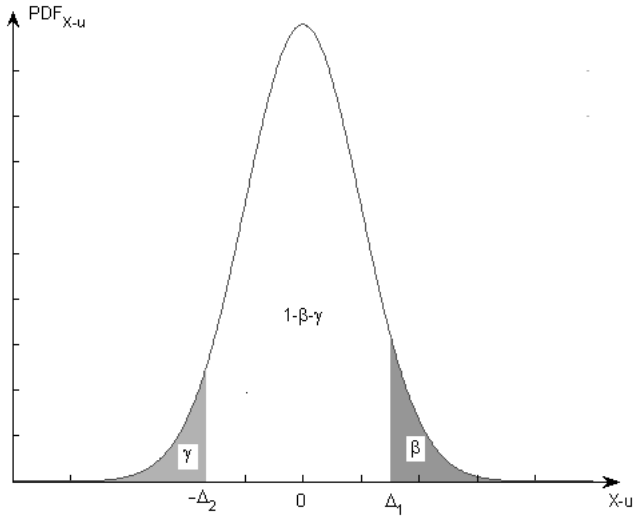

Figure 16: Distribution of the imbalance.

4.2 Solution with mean-value criterion

The following section contains a solution with mean-value criterion.

4.2.1 Deterministic equivalent

Considering all foregoing, an optimization problem with mean value criterion and probability constraints reads as follows:

$$\begin{aligned} &\max_u E\left\{c_s \cdot \min\{X, u\} - Y \cdot u - c_d \cdot \max\{X - u, 0\} - c_u \cdot \max\{u - X, 0\}\right\}, \\ &P\{X - u \leq \Delta_1\} \geq \beta, \\ &P\{u - X \leq \Delta_2\} \geq \gamma, \\ &u \geq 0. \end{aligned} \tag{72}$$

Theorem 4.1. *If the random variables X and Y have discrete distributions with finite number of realizations $x^i \in R$, $P(x^i) = p_x^i$, $i = 1, ..., n$, $y^i \in R$, $P(y^i) = p_y^i$, $i = 1, ..., m$, then:*

1. *Objective function $\Phi(u, X, Y)$, defined accordingly to (68) is a concave w.r.t. variable u on R.*

2. *Optimization problem (72) has a deterministic equivalent in the form of the linear programming (LP) problem:*

$$\begin{array}{l}\max\limits_{\substack{r_1^1,..,r_n^1\\ s_1^1,...,s_n^1\\ r_1^2,..,r_n^2\\ s_1^2,...,s_n^2\\ u}} , \left\{(c_s - \sum\limits_{i=1}^{m} y^i \cdot p_y^i)\cdot u - (c_s + c_u)\cdot \sum\limits_{i=1}^{n} p_x^i \cdot s_i^1 - c_d \cdot \sum\limits_{i=1}^{n} p_x^i \cdot s_i^2\right\},\\ r_i^1 - s_i^1 = x^i - u,\\ r_i^2 - s_i^2 = u - x^i,\\ r_i^1,\ s_i^1,\ r_i^1,\ s_i^1 \geq 0,\\ i = 1,..,n,\\ u \leq x_{1-\gamma} + \Delta_2,\\ u \geq x_\beta - \Delta_1,\\ u \geq 0,\end{array} \tag{73}$$

where x_β and $x_{1-\gamma}$ are critical values of the distribution of X.

Proof. 1. $\Phi(u,X,Y) = c_s\cdot\min\{X,u\} - Y\cdot u - c_d\cdot\max\{X-u,0\} - c_u\cdot\max\{u-X,0\} = c_s\cdot\min\{X,u\} - Y\cdot u + c_d\cdot\min\{u-X,0\} + c_u\cdot\min\{X-u,0\}$
The functions

$$\begin{array}{l} f_1(u,X,Y) = \min\{u-X,0\},\\ f_2(u,X,Y) = \min\{u-X,0\},\\ f_3(u,X,Y) = \min\{u-X,0\}\end{array} \tag{74}$$

are concave w.r.t. u-variable as a minimum of the concave w.r.t. u functions $f_1(u,X,Y)$, $f_2(u,X,Y)$, $f_3(u,X,Y)$.

The function

$$f_4(u,X,Y) = -Y\cdot u \tag{75}$$

is a concave w.r.t. u because it is a linear function. Therefore $\Phi(u,X,Y)$ is a concave w.r.t. u as a linear combination of the concave functions with nonnegative coefficients (c_s, c_d, c_u having the meaning of prices, hence all of them are nonnegative).

2. Let us consider the function

$$f_5(u) = \min\{f_6(u),0\}. \tag{76}$$

The value of $f_6(u)$ $\forall u \in R$ can be always split on the positive r and the negative s components [5],

$$\begin{array}{rl} f_6(u) = & r - s,\\ & r \geq 0,\\ & s \geq 0,\end{array} \tag{77}$$

therefore the value of $f_5(u)$ for the fixed u equals to the optimum value of the following optimization problem:

$$\begin{array}{rl} f_5(u) = & \max\limits_{r,s \in R}\{-s\}, \\ & f_6(u) = r - s, \\ & r \geq 0, \\ & s \geq 0. \end{array} \tag{78}$$

The following representation can be used to substitute the function

$$f_7(u) = \min\{f_8(u), f_9(u)\} = \min\{f_8(u) - f_9(u), 0\} + f_9(u) \tag{79}$$

with a smooth constrained optimization problem:

$$\begin{array}{rl} f_7(u) = & \max\limits_{r,s \in R}\{f_9(u) - s\}, \\ & f_8(u) - f_9(u) = r - s, \\ & r \geq 0, \\ & s \geq 0. \end{array} \tag{80}$$

Considering all mentioned above:

$$\begin{array}{l} E\{\Phi(u, X, Y)\} = E\{c_s \cdot \min\{X, u\} - Y \cdot u + c_d \cdot \min\{u - X, 0\} + c_u \cdot \min\{X - u, 0\}\} = \\ E\{c_s \cdot \min\{X - u, 0\} + c_s \cdot u - Y \cdot u + c_d \cdot \min\{u - X, 0\} + c_u \cdot \min\{X - u, 0\}\} = \\ E\{(c_s + c_u) \cdot \min\{X - u, 0\} + (c_s - Y) \cdot u + c_d \cdot \min\{u - X, 0\}\} = \\ (c_s + c_u) \cdot \sum\limits_{i=1}^{n} p_x^i \cdot \min\{x^i - u, 0\} + (c_s - \sum\limits_{i=1}^{m} y^i \cdot p_y^i) \cdot u + c_d \cdot \sum\limits_{i=1}^{n} p_x^i \cdot \min\{u - x^i, 0\} = \\ \max\limits_{\substack{r_1^1,..,r_n^1 \\ s_1^1,...,s_n^1 \\ r_1^2,..,r_n^2 \\ s_1^2,...,s_n^2}} \left\{ (c_s - \sum\limits_{i=1}^{m} y^i \cdot p_y^i) \cdot u - (c_s + c_u) \cdot \sum\limits_{i=1}^{n} p_x^i \cdot s_i^1 - c_d \cdot \sum\limits_{i=1}^{n} p_x^i \cdot s_i^2 \right\}, \\ r_i^1 - s_i^1 = x^i - u, \\ r_i^2 - s_i^2 = u - x^i, \\ r_i^1,\ s_i^1,\ r_i^1,\ s_i^1 \geq 0, \\ i = 1, .., n. \end{array} \tag{81}$$

Let's look at the probability constraints.

$$\begin{array}{l} P\{X - u \leq \Delta_1\} \geq \beta \Longleftrightarrow P\{X \leq u + \Delta_1\} \geq \beta \Longleftrightarrow \\ F(u + \Delta_1) \geq \beta \Longleftrightarrow u + \Delta_1 \geq x_\beta \Longleftrightarrow u \geq x_\beta - \Delta_1, \end{array} \tag{82}$$

where F is a CDF of the random variable X and x_β is β-critical value of the distribution of X.

Similarly

$$P\{u - X \leq \Delta_2\} \geq \gamma \Longleftrightarrow u \leq x_{1-\gamma} + \Delta_2. \tag{83}$$

Combining (81) with (82) and (83) we get proposed deterministic equivalent in the form of a linear programming problem.

□

The first result regarding a convex property of the function $\Phi(u, X, Y)$ w.r.t. variable u means that $E\{\Phi(u, X, Y)\}$ will be a concave function as well [6], hence every local maximum of $E\{\Phi(u, X, Y)\}$ will be automatically a global, therefore various optimization algorithms for the non-smooth 1-d functions can be successfully applied in order to find a global optimum.

The second result opens a possibility to find an exact solution of the problem applying a simplex method for the linear programming problems.

Let us introduce the following notations:

$b_1 = \left[\begin{array}{cccccc} x^1 & \dots & x^n & x^1 & \dots & x^n \end{array} \right]^T$,

$z = \left[\begin{array}{cccccccccccc} u & s_1^1 & ... & s_1^n & r_1^1 & ... & r_1^n & s_2^1 & ... & s_2^n & r_2^1 & ... & r_2^n \end{array} \right]^T$,

$C = \left[\begin{array}{cccccccccccccc} c_s - \sum\limits_{i=1}^{m} y^i \cdot p_y^i & -p_x^1 \cdot (c_s + c_u) & ... & -p_x^n (c_s + c_u) & 0 & ... & 0 & -c_d p_x^1 & ... & -c_d p_x^n & 0 & ... & 0 \end{array} \right.$

I - an identity-matrix $(n \times n)$,

e - a vector $(n \times 1)$, $e = \left[\begin{array}{ccc} 1 & \dots & 1 \end{array} \right]^T$,

$0_{n \times n}$ - a zero-matrix $(n \times n)$,

$$A_1 = \left[\begin{array}{ccccc} e & -I & I & 0_{n \times n} & 0_{n \times n} \\ e & 0_{n \times n} & 0_{n \times n} & I & -I \end{array} \right],$$

$$A_2 = \left[\begin{array}{ccc} & \overbrace{\qquad}^{4n} & \\ 1 & 0 \dots 0 & \\ -1 & 0 \dots 0 & \end{array} \right],$$

$b_2 = \left[\begin{array}{cc} x_{1-\gamma} + \Delta_2 & \Delta_1 - x_\beta \end{array} \right]^T$.

Using the notations above the problem (73) assumes the following standard form of the LP-problem which is traditionally solved by the simplex-method.

$$\begin{array}{l} \max\limits_z \{C^T z\}, \\ A_1 z = b_1, \\ A_2 z \leq b_2, \\ z_i \geq 0, \\ i = 1, ..., 4n + 1. \end{array} \tag{84}$$

Finally it is worth to show the main technique used to proof the deterministic equivalent on the test example. Let

$$\psi = min(f, 0) \tag{85}$$

and the corresponding optimization problem:

$$\begin{aligned} \psi = \quad & \max_{s,r}\{-s\}, \\ & f = r - s, \\ & r \geq 0, \\ & s \geq 0. \end{aligned} \tag{86}$$

Assume that $f = 5$ and therefore $\psi = min(5, 0) = 0$. Let us check that the optimization problem gives the same result:

$$\begin{aligned} \psi = \quad & \max_{s,r}\{-s\}, \\ & 5 = r - s, \\ & r \geq 0, \\ & s \geq 0. \end{aligned} \tag{87}$$

Obviously $s = r - 5$ and

$$\begin{aligned} \psi = \quad & \max_{r}\{-(r-5)\}, \\ & r - 5 \geq 0, \\ & r \geq 0. \end{aligned} \tag{88}$$

$$\begin{aligned} \psi = \quad & \max_{r}\{5 - r\}, \\ & r \geq 5, \\ & r \geq 0. \end{aligned} \tag{89}$$

$$\begin{aligned} \psi = \quad & \max_{r}\{5 - r\}, \\ & r \geq 5. \end{aligned} \tag{90}$$

$$\begin{cases} \psi = 0, \\ r = 0. \end{cases} \tag{91}$$

Now assume $f = -5$ and therefore $\psi = min(-5, 0) = -5$.

$$\begin{aligned} \psi = \quad & \max_{s,r}\{-s\}, \\ & -5 = r - s, \\ & r \geq 0, \\ & s \geq 0. \end{aligned} \tag{92}$$

Obviously $s = r + 5$ and

$$\begin{array}{ll} \psi = & \max\limits_{r}\{-5 - r\}, \\ & r + 5 \geq 0, \\ & r \geq 0. \end{array} \tag{93}$$

$$\begin{array}{ll} \psi = & \max\limits_{r}\{-5 - r\}, \\ & r \geq 0. \end{array} \tag{94}$$

$$\left\{ \begin{array}{l} \psi = -5, \\ r = 0. \end{array} \right. \tag{95}$$

4.2.2 SQG-solution

Assume now, that the random variables are given with continuous distributions. In this case to solve the problem one can use the Monte-Carlo simulation and the computation might be time consuming. More elegant way is to use stochastic approximation.

Let us derive a subgradient of the objective function in order to employ the SQG-algorithm. Firstly we transform the cost function to the piece-wise form:

$$\Phi(u, X, Y) = \left\{ \begin{array}{ll} c_s u - Yu - c_d(X - u), & u \leq X \\ c_s X - Yu - c_u(u - X), & u \geq X. \end{array} \right. \tag{96}$$

A subgradient of the objection function reads as follows:

$$\partial_u \Phi(u, X, Y) = \left\{ \begin{array}{ll} c_s - Y - c_d, & u < X \\ \forall \in [-Y - c_u \,;\, c_s - Y - c_d], & u = X \\ -Y - c_u, & u > X. \end{array} \right. \tag{97}$$

Now, to solve the problem, we just need to sample random numbers accordingly to the given distribution and apply (47), using $\partial_u \Phi(u, X, Y)$ as a quasi-gradient and projecting to the admissible area $U = [\max\{x_\beta - \Delta_1, 0\} \,;\, x_{1-\gamma} + \Delta_2]$.

4.3 Solution with CVaR criterion

A previous result with mean-value criterion does not assess the probability of getting the solution which is less than mean-value, depending on the realization of random parameters, therefore it is quite useful to obtain a solution with VaR or CVaR criterion.

Let $G(u, Z(w))$ denote a criterion cost function dependent on a random vector $Z(w)$ and the strategy u which should be chosen from some set $U \subset \mathbf{R}^n$ in order to optimize the cost function according to a certain criterion. For the given fixed strategy u the random variable $G(u, Z(w))$ usually having the meaning of the random losses, associated with the chosen strategy u. If we denote $\mathbf{P}\{\cdot\}$ as a probability measure, generated by the distribution of the random vector $Z(w)$, we can define the probability function $P_\phi(u)$ and the VaR criterion denoted by the quantile function $\phi_\alpha(u)$:

$$P_\phi(u) = \mathbf{P}\{G(u, Z(w)) \leq \phi\}, \tag{98}$$

$$\phi_\alpha(u) = \min\{\varphi : P_\varphi(u) \geq \alpha\}. \tag{99}$$

The probability function represents a probability that our cost function $G(u, Z(w))$ does not exceed a level ϕ for a fixed strategy u while the qunatile function indicates the corresponding minimal level.

The CVaR criterion, defined by the function $\psi_\alpha(u)$, estimates an average rate of the losses exceeded $\phi_\alpha(u)$:

$$\psi_\alpha(u) = E[G(u, Z(w))|G(u, Z(w)) \geq \phi_\alpha(u)] = \frac{1}{1-\alpha} \int\limits_{G(u,Z(w))\geq\phi_\alpha(u)} G(u, Z(w))dP(w). \tag{100}$$

According to [7] the CVaR-minimization over the strategy $u \in U$ equals to the solution of the following optimization problem:

$$\psi^* = \min_{(u,\phi)\in U\times R} F_\alpha(u, \phi), \tag{101}$$

where

$$F_\alpha(u, \phi) = \phi + \frac{1}{1-\alpha} \cdot E[\max\{G(u, Z(w)) - \phi, 0\}. \tag{102}$$

The present case deals with the retailer's profit maximization, therefore in order to apply mentioned above theoretical properties, we put "-" sign before the objective function and further solve minimization problem as it is required by the original definitions. As a result we get an average profit of the retailer which is less or equal to the critical value, defined by VaR.

Applying (102) to the objective function (68), we get:

$$\begin{aligned} \psi^* = \quad & -\min_{u,\phi} \left\{\phi + \tfrac{1}{1-\alpha} \cdot E[\max\{-\Phi(u, X, Y) - \phi, 0\}]\right\}, \\ & u \leq x_{1-\gamma} + \Delta_2, \\ & u \geq x_\beta - \Delta_1, \\ & u \geq 0, \end{aligned} \tag{103}$$

where

$$\Phi(u, X, Y) = c_s \cdot \min\{X, u\} - Y \cdot u - c_d \cdot \max\{X - u, 0\} - c_u \cdot \max\{u - X, 0\}. \quad (104)$$

The random demand X and the global price Y can be treated as independent random variables (at least during the short term) without big limitations, because between the retailer and the consumer there is a contract with fixed price for the electricity c_s.

4.3.1 Deterministic equivalent

Considering the remark above, the next result is valid.

Theorem 4.2. *If the random variables X an Y are independent and have discrete distributions with finite number of realizations $x^i \in R$, $P(x^i) = p_x^i$, $i = 1, ..., n$, $y^i \in R$, $P(y^i) = p_y^i$, $i = 1, ..., m$, then (103) has a deterministic equivalent in the linear programming (LP) form:*

$$\begin{aligned}
\psi^* = \quad & \max_{\substack{u,\phi,\\ r_i^1, r_i^2,\\ s_i^1, s_i^2,\\ \Psi_{ij},\\ i=1,...,n,\\ j=1,...,m}} \left\{ -\phi - \tfrac{1}{1-\alpha} \cdot \textstyle\sum_{i,j=1}^{m \cdot n} p_x^i p_y^j \cdot \Psi_{ij} \right\}, \\
& - ((c_s - y^j) \cdot u - (c_s + c_u) \cdot s_i^1 - c_d \cdot s_i^2) - \phi \leq \Psi_{ij}, \\
& \Psi_{ij} \geq 0, \\
& r_i^1 - s_i^1 = x^i - u, \\
& r_i^2 - s_i^2 = u - x^i, \\
& r_i^1,\ s_i^1,\ r_i^2,\ s_i^2 \geq 0, \\
& i = 1, .., n, \\
& j = 1, .., m, \\
& u \leq x_{1-\gamma} + \Delta_2, \\
& u \geq x_\beta - \Delta_1, \\
& u \geq 0.
\end{aligned} \quad (105)$$

Proof. Transforming minimization into maximization:

$$\begin{aligned}
\psi^* = \quad & \max_{u,\phi} \left\{ -\phi - \tfrac{1}{1-\alpha} \cdot E[\max\{-\Phi(u, X, Y) - \phi, 0\}] \right\}, \\
& u \leq x_{1-\gamma} + \Delta_2, \\
& u \geq x_\beta - \Delta_1, \\
& u \geq 0.
\end{aligned} \quad (106)$$

Discrete random variables X and Y are independent, therefore mathematical expectation

can be computed according to the rule:

$$\begin{array}{rl} \psi^* = & \max\limits_{u,\phi} \left\{ -\phi - \frac{1}{1-\alpha} \cdot \sum\limits_{i,j=1}^{mn} p_x^i p_y^j \cdot \max\{-\Phi(u, x^i, y^j) - \phi, 0\} \right\}, \\ & u \le x_{1-\gamma} + \Delta_2, \\ & u \ge x_\beta - \Delta_1, \\ & u \ge 0. \end{array} \tag{107}$$

where the objective function substituted by the constrained optimization problem:

$$\begin{array}{rl} \Phi(u, x^i, y^j) = & \max\limits_{r_i^1, r_i^2, s_i^1, s_i^2} \{(c_s - y^j) \cdot u - (c_s + c_u) \cdot s_i^1 - c_d \cdot s_i^2\}, \\ & r_i^1 - s_i^1 = x^i - u, \\ & r_i^2 - s_i^2 = u - x^i, \\ & r_i^1,\ s_i^1,\ r_i^2,\ s_i^2 \ge 0. \end{array} \tag{108}$$

Combining (107) and (108) together, changing the order of maximization, we get:

$$\begin{array}{rl} \psi^* = & \max\limits_{\substack{u,\phi, \\ r_i^1, r_i^2, \\ s_i^1, s_i^2 \\ i=1,..,n}} \left\{ -\phi - \frac{1}{1-\alpha} \cdot \sum_{i,j=1}^{m \cdot n} p_x^i p_y^j \cdot \max\{-\left((c_s - y^j) \cdot u - (c_s + c_u) \cdot s_i^1 - c_d \cdot s_i^2\right) - \phi, 0\} \right\}, \\ & r_i^1 - s_i^1 = x^i - u, \\ & r_i^2 - s_i^2 = u - x^i, \\ & r_i^1,\ s_i^1,\ r_i^2,\ s_i^2 \ge 0, \\ & i = 1, .., n, \\ & u \le x_{1-\gamma} + \Delta_2, \\ & u \ge x_\beta - \Delta_1, \\ & u \ge 0. \end{array} \tag{109}$$

Finally, to get rid of an internal maximization in the last expression, we introduce extra variables Ψ_{ij} and obtain a large size linear programming problem (105). □

It is worth to note, that according to [7]

$$\phi_\alpha(u) = \min\{\arg\min_{\phi \in R} F_\alpha(u, \phi)\}, \tag{110}$$

therefore an optimal value of ϕ^* acquired from (105) can be treated as an estimate of the solution with VaR-criterion and for instance used as a good starting point for iterative algorithms.

For convenience we introduce the following notations :
$\phi^1 - \phi^2 = \phi$,

$\phi^1, \phi^2 \geq 0$,
$z = \left[\begin{array}{cccccccccccccccccc} u & \phi^1 & \phi^2 & s_1^1 & ... & s_1^n & r_1^1 & ... & r_1^n & s_2^1 & ... & s_2^n & r_2^1 & ... & r_2^n & \Psi_{11} & ... & \Psi_{mn} \end{array} \right]^T$,
$C = \left[\begin{array}{cccccccccccccccccc} 0 & -1 & 1 & 0 & ... & 0 & 0 & ... & 0 & 0 & ... & 0 & 0 & ... & 0 & -\frac{1}{1-\alpha} \cdot p_x^1 p_y^1 & ... & -\frac{1}{1-\alpha} \cdot p_x^n p_y^m \end{array} \right]^T$,
$b_1 = \left[\begin{array}{cccccc} x^1 & \dots & x^n & x^1 & \dots & x^n \end{array} \right]^T$.
I - an identity-matrix $(n \times n)$,
e - vector $(n \times 1)$, $e = \left[\begin{array}{ccc} 1 & \dots & 1 \end{array} \right]^T$,
$0_{n \times n}$ - zero-matrix $(n \times n)$.

$$A_1 = \left[\begin{array}{cccccccc} e & 0_{n\times 2} & -I & I & 0_{n\times n} & 0_{n\times n} & \overbrace{0 \dots 0}^{mn} \\ e & 0_{n\times 2} & 0_{n\times n} & 0_{n\times n} & I & -I & 0 \dots 0 \end{array} \right],$$

$$A_2 = \left[\begin{array}{cccccccccccc} -(c_s - y^1) & -1 & 1 & (c_s + c_u) & \overbrace{0 \dots 0}^{n-1} & \overbrace{0 \dots 0}^{n} & c_d & \overbrace{0 \dots 0}^{n-1} & \overbrace{0 \dots 0}^{n} & -1 & \overbrace{0 \dots 0}^{mn-1} \\ ... & ... & ... & ... & ... & ... & ... & ... & ... & ... & ... \\ -(c_s - y^m) & -1 & 1 & \overbrace{0 \dots 0}^{n-1} & (c_s + c_u) & \overbrace{0 \dots 0}^{n} & \overbrace{0 \dots 0}^{n-1} & c_d & \overbrace{0 \dots 0}^{n} & \overbrace{0 \dots 0}^{mn-1} & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right],$$

$$b_2 = \left[\begin{array}{ccc} \overbrace{0 \dots 0}^{mn} & x_{1-\gamma} + \Delta_2 & \Delta_1 - x_\beta \end{array} \right]^T.$$

In the notations above the problem (105) again assumes standard form of the LP-problem which is solved by simplex-method.

$$\begin{array}{l} \max C^T z \\ A_1 z = b_1, \\ A_2 z \leq b_2, \\ z_i \geq 0, \\ i = 1, ..., mn + 4n + 3. \end{array} \tag{111}$$

4.3.2 SQG-solution

To apply the SQG-algorithm for the present case let us slightly transform the function (106) using linearity of the mathematical expectation:

$$F_1(u, \phi) = \max_{u,\phi} \left\{ E[-\phi - \frac{1}{1-\alpha} \cdot \max\{-\Phi(u, Z) - \phi, 0\}] \right\}. \tag{112}$$

Having such representation, we are ready to employ a finite-difference approach to get the quasi-gradient estimate:

$$\begin{cases} \zeta_u^k = \frac{F_2(u_k+\delta_k,\phi_k,Z_k)-F_2(u_k-\delta_k,\phi_k,Z_k)}{2\delta_k}, \\ \zeta_\phi^k = \frac{F_2(u_k,\phi_k+\delta_k,Z_k)-F_2(u_k,\phi_k-\delta_k,Z_k)}{2\delta_k}, \end{cases} \tag{113}$$

where $F_2(u, \phi, Z)$ defined as follows:

$$F_2(u, \phi, Z) = -\phi - \frac{1}{1-\alpha} \cdot \max\{-\Phi(u, Z) - \phi, 0\}. \tag{114}$$

Now, to solve the problem, we just need to sample random numbers accordingly to the given distribution and apply (47).

4.4 Numerical examples

The following numerical values has been artificially chosen (in the case of discrete distribution we used deterministic equivalent, and for the continuous case a stochastic approximation algorithm has been employed).

c_s	c_d	c_u	Δ_1	Δ_2	α	β	γ
2.2	0.3	0.1	40	35	0.7	0.6	0.7

Table 1: Numerical values.

x^i	10	20	30	40	50	60	70	80	90	100
p_x^i	0.05	0.05	0.05	0.05	0.1	0.2	0.2	0.15	0.1	0.05

Table 2: Discrete distribution of the random demand.

y^i	0.1	0.2	0.3	0.4	0.5	0.6
p_y^i	0.1	0.2	0.3	0.2	0.15	0.05

Table 3: Discrete distribution of the random price.

In the continuous case the random demand and the price were chosen normally distributed $X \sim N(70, 10^2)$ and $Y \sim N(0.4, 0.1^2)$.

Optimal solutions with mean-value and CVaR criteria are shown on the Figures (17-22):

A solution with mean-value criterion (by the corresponding deterministic equivalent):
$u^* = 80$,
$E\{\Phi(u^*, X, Y)\} = 103.4$.

A solution with CVaR criterion:
$u^* = 58$,
$\phi^* = 91.8$,
$\psi^* = 50.5$.

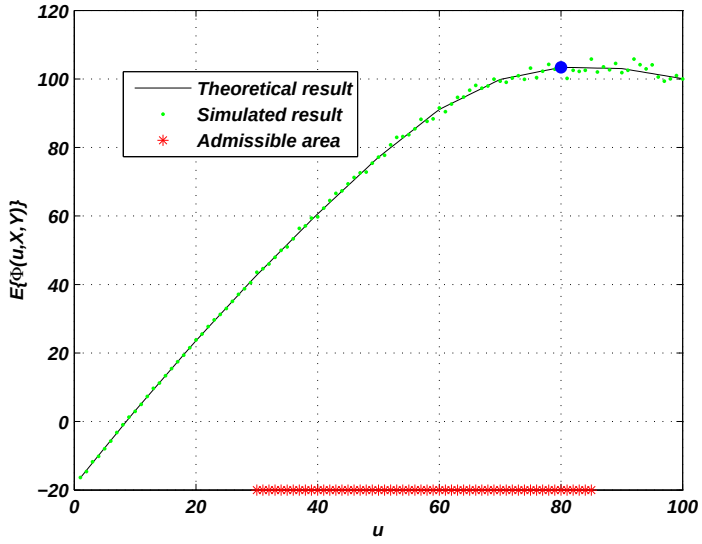

Figure 17: Solution with mean-value criterion for the discrete distribution.

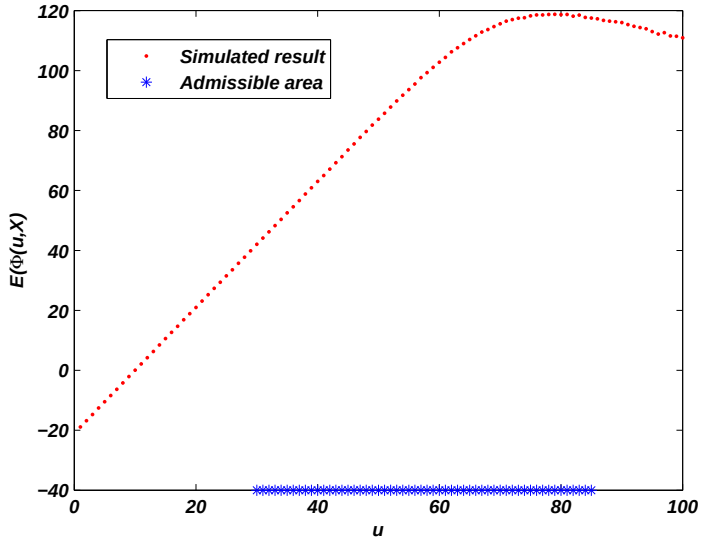

Figure 18: Solution with mean-value criterion for the continuous distribution.

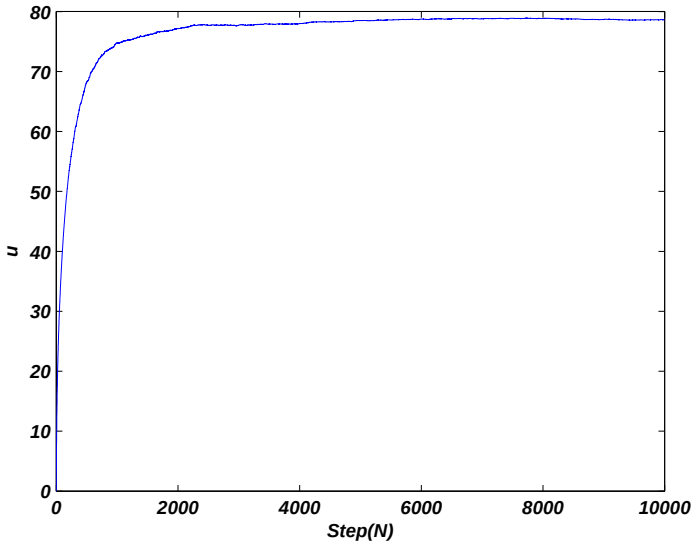

Figure 19: SQG-algorithm for the continuous distribution.

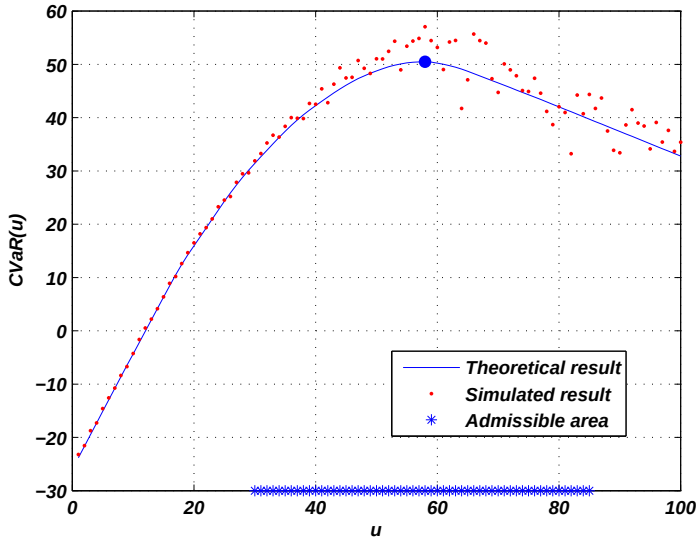

Figure 20: Solution with CVaR criterion for the discrete distribution.

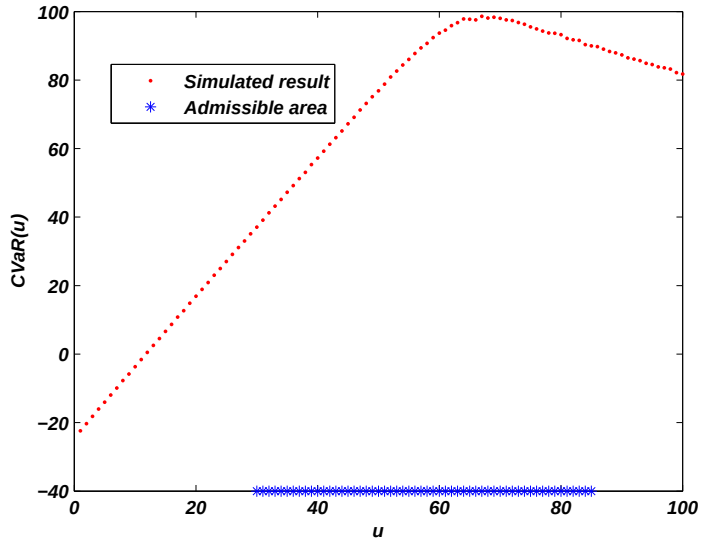

Figure 21: Solution with CVaR criterion for the continuous distribution.

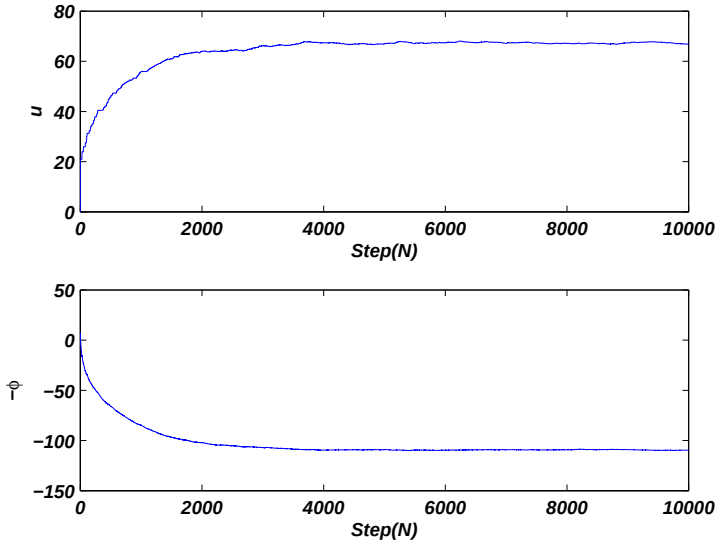

Figure 22: SQG-solution.

4.5 Analysis of the results

As it can be seen from the model examples, a mean-value solution are more optimistic than a α-CVaR solution for the high confident levels α. The strength of the CVaR

comparing with the mean-value is a reliability and as a result it gives more adequate measure of the risk.

5 Application problem: SQG-algorithm for the objective functions in ODE-form

Modern engineering optimization techniques employ the Markov-Chain-Monte-Carlo (MCMC) simulation in order to obtain a distribution of the random parameters using the limited set of measurements during the experiment. There have been quite many algorithms developed so far [16], but all of them as an output produce large samples of the parameters' realizations.

Considering foregoing it is logically to use stochastic approximation techniques together with the MCMC-simulation because the results of the parameter estimation procedure are perfectly suitable for the subsequent optimization.

One special brunch of the engineering optimization can be represented via processes given as a solution of ordinary differential eqations (ODE). Such functions often come from the chemical or biological research problems [8] and as it is shown in the next sections, the SQG-algorithm can be quite successfully applied to get an optimal solution in these cases.

5.1 Problem formulation

Let $\Phi(u, X)$ denotes an objective function dependent on the random vector X and the strategy u which should be chosen from the set $U \subset R^n$ in order to optimize the cost function according to a certain criterion.

Assume that the objective function is given as a solution of the differential equation or dependent on this solution:

$$\Phi(u,x) = \Phi\left(S_1(t, v_1, ..., v_l, x_1, ..., x_r), ..., S_m(t, v_1, ..., v_l, x_1, ..., x_r)\right), \tag{115}$$

$$u = \begin{pmatrix} t \\ v \end{pmatrix},$$

where

$$\begin{cases} \frac{dS_i}{dt} = f_i(t,\ S_1, ..., S_m,\ v_1, ..., v_l,\ x_1, ..., x_r), \\ S_i(t_0, v_1, ..., v_l,\ x_1, ..., x_r) = S_i^0(v_1, ..., v_l,\ x_1, ..., x_r), \\ i = 1, ..., m. \end{cases} \tag{116}$$

Further for a convenience the previous system will be written shortly:

$$\Phi(u, x) = \Phi(S(t, v, x)), \tag{117}$$

$$\begin{cases} \frac{dS_i}{dt} = f_i(t,\ S,\ v,\ x), \\ S_i(t_0, v, x) = S_i^0(v,\ x), \\ i = 1, ..., m. \end{cases} \tag{118}$$

Here $x \in R^r$ is a realization of the random vector X which contains information about parameter's uncertainty of the model, $u \in R^n$, $(n = l + 1)$ denotes a control variable. Objective functions in such form are often used to represent transformations of the components in chemical reactions where the distribution of the random parameter X is a posterior distribution $\pi(X|Y)$ estimated by the MCMC-simulation [8] for the given set of experimental measurements Y.

From the practical point of view after the MCMC-parameter estimation procedure we have a large sample representing $\pi(X|Y)$. The next step is usually an optimization of the cost function which depends on the random parameters and the control variables. Therefore effective procedures which use large samples for optimization are highly demanded research area in decision making and process optimization theory.

Different criteria can be used in optimization of the function (117): minimax, mathematical expectation, value-at-risk, conditional value-at-risk. The choice of the optimization criterion is usually determined by the features of a problem to be solved. Further in this section we consider a mean-value criterion (1):

$$\min_{u \in U} E\{\Phi(u, X)\}. \tag{119}$$

5.2 Solution technique

To find a solution we applied stochastic approximation procedures described in the section 3.5 and compare their outcomes.

The first algorithm is a classical Kiefer-Wolfowitz procedure [17]. In every step of the algorithm a stochastic quasi-gradient is estimated with a finite-difference approximation, so the procedure requires $2n$ evaluations of the objective function (also in every evaluation we have to solve an ODE-system).

The second solution is based on the stochastic gradient algorithm with the gradient,

estimated using an equation in variations. Instead of the finite difference approach to estimate a stochastic gradient we compose additional equations to the initial ODE-system and decrease the number of evaluations to the 1.

Finally we consider simultaneous perturbation stochastic approximation method (SPSA), which requires 2 function evaluations in every step of the procedure.

5.2.1 Equation in variations

Assume that we have a system of ordinary differential equations (ODE-system) with parameters $\mu^1, ..., \mu^n$.

$$\begin{gathered} \frac{dx^l}{dt} = f(t, x^1, ..., x^n, \mu^1, ..., \mu^l), \\ l = 1, ..., n. \end{gathered} \tag{120}$$

The previous system can be written in the vector-notation as follows:

$$\frac{d\mathbf{x}}{dt} = f(t, \mathbf{x}, \mu), \tag{121}$$

where $\mathbf{x}$ and μ are vectors. Sometimes it is needed to get an information about derivatives of the solution $\phi(t, \mu)$ of the system with respect to the parameters μ^k for a fixed value of $\mu = \mu^*$. One obvious way to get it is to find a solution $\phi(t, \mu)$ for a varying μ and then differentiate it with respect to μ^k. But it turns out that there is no need to do so, and it is possible to get unknown differentials considering some extended ODE-system. Let $\phi(t, \mu) = (\phi^1(t, \mu), ..., \phi^n(t, \mu))$ is a solution of the initial ODE-system for initial conditions t_0, $\mathbf{x}_0$ and $m_1 < t < m_2$ is an interval where the solution is defined for a fixed value of the parameter $\mu = \mu^*$.

The functions, which we need to find at the point μ^*:

$$\psi_k^i(t) = \frac{\partial \phi^i(t, \mu^*)}{\partial \mu^k}. \tag{122}$$

Let us introduce several notations:

$$f_j^i(t, \mathbf{x}, \mu) = \frac{\partial f^i(t, \mathbf{x}, \mu)}{\partial x^j}, \tag{123}$$

$$f_j^i(t) = f_j^i(t, \phi(t, \mu^*), \mu^*), \tag{124}$$

$$g_k^i(t, \mathbf{x}, \mu) = \frac{\partial f^i(t, \mathbf{x}, \mu)}{\partial \mu^k}, \tag{125}$$

$$g_k^i(t) = g_k^i(t, \phi(t, \mu^*), \mu^*). \tag{126}$$

Assuming that the partial derivatives of the right-hand sides $\frac{\partial f^i(t,x,\mu)}{\partial \mu^k}$ are continuous at some area G, the following linear ODE-system

$$\frac{dy^i}{dt} = \sum_{j=1}^{n} f_j^i(t) y^i + g_k^i(t) \tag{127}$$

is called an equation in variations [10] and the system of the functions

$$y^1 = \psi_k^1(t), ..., y^n = \psi_k^n(t) \tag{128}$$

is a solution of the equations (127) with the initial conditions $\psi_k^i(t_0) = 0$.

Note, that in the formulas (123-126) the function $\phi(t, \mu^*)$ is explicitly found solution of the initial system for the given value of parameter μ^*, practically meaning that to solve the equation in variations we still need to keep the initial system in order to successfully get $\phi(t, \mu^*)$.

To show how the mentioned above theory is working, let us consider the following example.

$$\begin{array}{c} \frac{dx}{dt} = \mu x, \\ x(0) = 1. \end{array} \tag{129}$$

Let the solution of this equation is $\phi(t, \mu)$. We need to find $\frac{\partial \phi(t,\mu)}{\partial \mu}$ for $\mu^* = 3$.

Obviously the result is clearly seen without any additional calculations:

$$\phi(t, \mu) = e^{\mu t} \tag{130}$$

and

$$\frac{\partial \phi(t, \mu)}{\partial \mu} = te^{\mu t}, \tag{131}$$

so that the answer:

$$\frac{\partial \phi(t, \mu)}{\partial \mu}|_{\mu=3} = te^{3t}. \tag{132}$$

Having the answer to compare with, we can write an equation in variations for the present case.

Keeping the notations:

$$f(t, x, \mu) = \mu x, \tag{133}$$

$$f_1^1(t, x, \mu) = \frac{\partial f(t, x, \mu)}{\partial x} = \mu, \tag{134}$$

$$f_1^1(t) = f_1^1(t, \phi(t, \mu^*), \mu^*) = 3, \tag{135}$$

$$g_1^1(t, x, \mu) = \frac{\partial f(t, \mathbf{x}, \mu)}{\partial \mu} = x, \tag{136}$$

$$g_1^1(t) = g_1^1(t, \phi(t, \mu^*), \mu^*) = \phi(t, \mu^*). \tag{137}$$

To find $\phi(t, \mu^*)$ we write the initial system, substituting μ with $\mu^* = 3$:

$$\begin{array}{l} \frac{d\phi}{dt} = \phi, \\ \phi(0) = 1. \end{array} \tag{138}$$

Assembling all together we get the equation in variations for the considered case:

$$\left\{ \begin{array}{l} \frac{dy}{dt} = 3y + \phi, \\ \frac{d\phi}{dt} = 3\phi, \\ \phi(0) = 1, \\ y(0) = 0. \end{array} \right. \tag{139}$$

Solving this system we get an answer coinciding with the previously given theoretical reasoning:

$$y(t) = te^{3t}. \tag{140}$$

5.2.2 Application of the equation in variations in the considered case

Let us recall that we are dealing with the objective function in ODE-form and therefore it is possible to get rid of the finite difference's approach by applying an equation in

variations [10] for the unknown differential's estimation and obtain a stochastic gradient with less computation error.

The result of such technique might enhance convergence and might not. The outcome depends on the concrete optimization task. For example in the case when the data contains large rate of noise and if in every step of the SQG we are using only one realization of random vector for the gradient estimation, the finite difference approach will give a better descent direction.

But in practice it is important to have several computation possibilities in order to be able to compare the results, especially in the complicated cases.

Another benefit of using the suggested technique could be a computational speed: sometimes it is faster or even more convenient to solve a slightly extended ODE-system than to conduct several computations with original one what is required in the case when we are employing finite differences.

Differentials with respect to t-variable can be found explicitly from the initial ODE according to the rule of the complex function's differentiation:

$$\frac{\partial \Phi(S(t,v,x))}{\partial t}|_{\substack{t=t^1\\v=v^1\\x=x^1}} = \left(\sum_{i=1}^{m} \frac{\partial \Phi(S(t,v,x))}{\partial S_i} \frac{\partial S_i(t,v,x)}{\partial t}\right)|_{\substack{t=t^1\\v=v^1\\x=x^1}}. \tag{141}$$

Considering the theorem from [10], an extended ODE-system based on the equation in variations for unknown differentials' estimation

$$\begin{aligned} &\frac{\partial \Phi(S(t,v,x))}{\partial v_j}|_{\substack{t=t^1\\v=v^1\\x=x^1}} = \left(\sum_{i=1}^{m} \frac{\partial \Phi(S(t,v,x))}{\partial S_i} \frac{\partial S_i(t,v,x)}{\partial v_j}\right)|_{\substack{t=t^1\\v=v^1\\x=x^1}}, \\ &j = 1, ..., l \end{aligned} \tag{142}$$

can be written as follows:

$$\begin{cases} \frac{dS_i}{dt} = f_i(t,\, S,\, v,\, x)|_{\substack{v=v^1\\x=x^1}}, \\ y_{i,k} = \frac{\partial S_i(t,v,x)}{\partial v_k}|_{\substack{v=v^1\\x=x^1}}, \\ \frac{dy_{i,k}}{dt} = \sum\limits_{j=1}^{l} \left(\frac{\partial f_i(t,S,v,x)}{\partial S_j} y_j + \frac{\partial f_i(t,S,v,x)}{\partial v_k}\right)|_{\substack{v=v^1\\x=x^1}}, \\ \frac{dS_i}{dt} = f_i(t,\, S,\, v,\, x)|_{\substack{v=v^1\\x=x^1}}, \\ S_i^0(t_0,v,x)|_{\substack{v=v^1\\x=x^1}} = S_i^0(v,x)|_{\substack{v=v^1\\x=x^1}}, \\ y_{i,k}(0) = \frac{\partial S_i^0(t,v,x)}{\partial v_k}|_{\substack{v=v^1\\x=x^1}}, \\ i = 1, ..., m, \\ k = 1, ..., l. \end{cases} \tag{143}$$

An extended ODE-system has the following structure: it contains the initial ODE-system and $m \times l$ additional equations, where again m is a number of equations in the initial ODE-system and l is a dimension of the control variable v.

Finally the stochastic quasi-gradient algorithm enhanced by the equation in variations can be written as follows:

1. Obtain a posterior distribution of the unknown parameters using MCMC or other parameter estimation method (i.e. bootstrap by residuals) for the given measurements and the model of the process. The distribution in that case will be in the form of a large array or a chain, and every component of such chain can be treated as a realisation x of the unknown parameters.

2. Compose an extended ODE-system for the estimation of unknown differentials and solve it taking the realisation x^k from the chain and the current point u^k. The result of the solution will be an estimate of the stochastic gradient $\xi^k(u^k, x^k)$.

3. Find the next point u^{k+1} using the main equation of the stochastic quasi-gradient algorithm:
$$u^{k+1} = u^k - \rho_k \xi^k(u^k, x^k). \tag{144}$$
Go to the step 2 assuming u^{k+1} as a current point of the algorithm.

4. The algorithm stops when the number of iterations exceeds the predefined limit, other stopping criteria are also admissible.

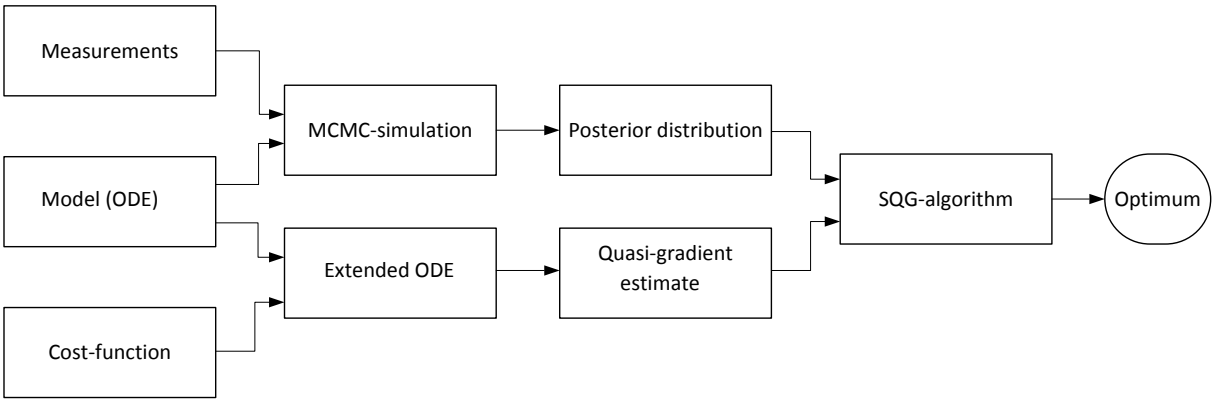

Figure 23: The scheme of the solution technique.

5.3 Numerical examples

Let us test the foregoing methods on the model examples. Consider the following chemical reaction that can be found in [8]:

$$A \overset{k_1}{\to} B \overset{k_2}{\to} C. \tag{145}$$

Given the measurements of components concentrations $\{A_1, ..., A_m\}$, $\{B_1, ..., B_m\}$ at the moments $\{t_1, ..., t_m\}$ and initial concentrations $A(0)$ and $B(0)$ at the beginning of the experiment. Required to determine the collection time that maximizes an average concentration of the component B.

time	0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
A	1	0.504	0.185	0.217	0.023	0.101	0.058	0.064	0	0.082
B	0	0.415	0.488	0.594	0.505	0.493	0.457	0.394	0.334	0.309

Table 4: Measurements of the reaction.

To find the solution firstly we derive the differential equation, reflecting the chemical reaction with the main meaning that the speed of the reaction is proportional to the mass of the interacting components with the coefficients k_1 and k_2.

$$\begin{cases} \frac{dA}{dt} = -k_1 A, \\ \frac{dB}{dt} = k_1 A - k_2 B, \\ A(0) = A_0, \\ B(0) = 0. \end{cases} \tag{146}$$

To get the posterior distribution $\Pi(k_1,\, k_2 \,|\, A_1, ..., A_m, B_1, ..., B_m)$ of the parameters for the given set of measurements we employ MCMC-simulation [8] and further X denotes the vector having such distribution $X \sim \Pi(k_1,\, k_2 \,|\, A_1, ..., A_m, B_1, ..., B_m)$.

The next figure shows the results of the MCMC-procedure: the distribution is almost spherical with the mean value attaining at the point $k_1 = 0.6$, $k_2 = 0.17$.

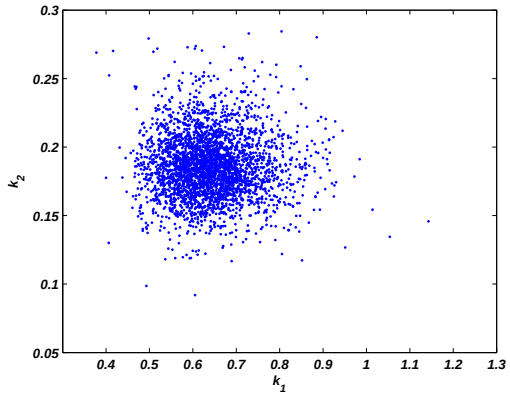

Figure 24: Distribution of the parameters estimated by MCMC for the first example.

The stochastic optimization problem in that case will be written as shown below.

$$\max_t E\{B(t, X)\}, \tag{147}$$

where vector X here as before represents posterior distribution of the parameters $X \sim \Pi(k_1,\, k_2 \,|\, A_1, ..., A_m, B_1, ..., B_m)$.

Obviously this problem is a one-dimensional because we have only time (t) as a control variable and therefore it can be simply solved by plotting the picture, but to demonstrate stochastic approximation methods we are employing here the stochastic quasi-gradient algorithm.

Note, that in this case the differential $\frac{dB(t,k_1,k_2)}{dt}|_{t=t^n}$ is already given explicitly in the ODE-system for any point $t = t^n$ and known realizations of the parameters $k_1 = k_1^n$, $k_2 = k_2^n$ and there is no need to estimate it via the finite difference approximation, meaning that everything is ready to apply the stochastic quasi-gradient method:

$$t^{n+1} = t^n + \rho_n \frac{dB}{dt}|_{t=t^n} \tag{148}$$

and substituting $\frac{dB}{dt}$ with the right-hand side of the initial equation we are getting the following iterative process:

$$t^{n+1} = t^n + \rho_n(k_1^n A(t^n) - k_2^n B(t^n)). \tag{149}$$

The values of $A(t^n)$ and $B(t^n)$ are found via the solution of the ODE-system (146) taking the corresponding to the step's number n random parameters k_1^n and k_2^n from the MCMC-chain. The values $A(t^n)$ and $B(t^n)$ depend on the realisations k_1^n and k_2^n

of the random parameters, therefore in every step of the algorithm we are required to solve the initial ODE.

The results of the calculations and the convergence trace are shown on the next figure, where we can see that the process starting from the initial point $t^0 = 10$ quickly achieves the optimum value $t^* = 2.3$. For comparison purposes the bunch of objective functions were drawn for all realizations of the random parameters from the MCMC chain.

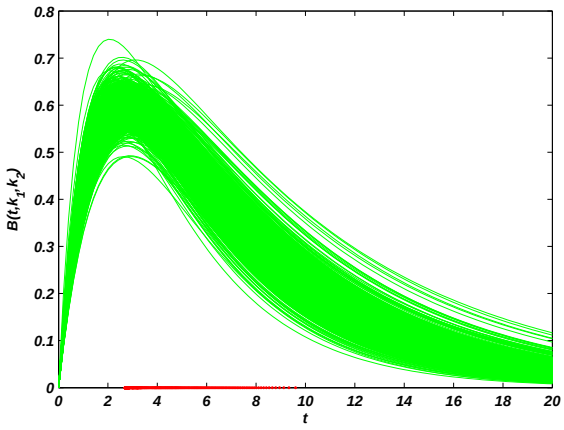

Figure 25: Objective function and the trace by the algorithm for the first test-case.

The next picture shows the convergence of the method, i.e. how the control variable t depends on the step. As it can be clearly seen 250 steps were enough to get the stable solution.

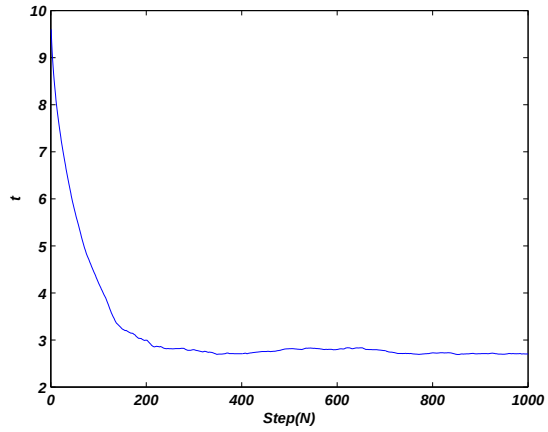

Figure 26: Convergence of the method for the first test-case.

To illustrate an application of the equation in variations lets have a look at the extended example [8].

$$\begin{cases} \frac{dA}{dt} = -k_1(T)A, \\ \frac{dB}{dt} = k_1(T)A - k_2(T)B, \\ k_i(T) = a_i exp\left[-\frac{E_i}{R}(\frac{1}{T} - \frac{1}{T_0})\right], \ i = 1, 2, \\ A(0) = A_0, \\ B(0) = 0. \end{cases} \tag{150}$$

Given the measurements of concentrations $\{A_1, ..., A_m\}$, $\{B_1, ..., B_m\}$ at the moments $\{t_1, ..., t_m\}$ for the fixed temperature T_1. For the sake of simplicity the values of variables R, E_i, T_0 are known. Required to determine a collection time and the temperature attaining maximum of the B-component concentration.

time	0	10.0	20.0	30.0	40.0	50.0	60.0	70.0	80.0	90.0
A	100	50.4	18.5	21.7	2.3	10.1	5.8	6.4	0	8.2
B	0	41.5	48.8	59.4	50.5	49.3	45.7	39.4	33.4	30.9

Table 5: Measurements of the reaction

The main difference between the previous example and the present is that the second contains two control variables time (t) and temperature (T). Let us denote for this example $u = [t \ T]^T$ - the vector of control variables, and $X = [a_1 \ a_2]^T$ - random parameters with the posterior distribution obtained by MCMC simulation.

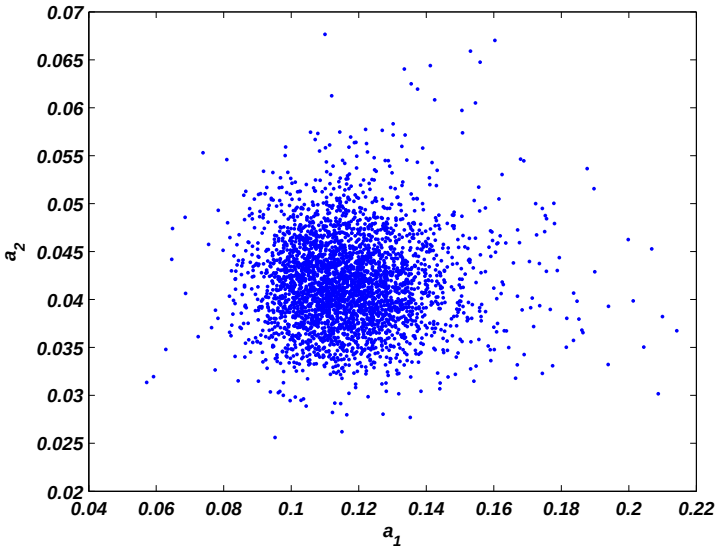

Figure 27: Distribution of the parameters estimated by MCMC for the second test-case.

In this example the differential $\frac{dB(t,T,a_1,a_2)}{dt}|_{\substack{t=t^n\\T=T^n}}$ is known explicitly from the initial ODE-system for any points $t = t^n$, $T = T^n$ and realizations $a_1 = a_1^n$, $a_2 = a_2^n$, while $\frac{dB(t,T,a_1,a_2)}{dT}|_{\substack{t=t^n\\T=T^n}}$ is unknown. Therefore to get it we follow suggested above technique and derive an extended ODE-system:

$$\begin{cases} k_i(T) = a_i^n exp\left[-\frac{E_i}{R}(\frac{1}{T} - \frac{1}{T_0})\right],\ i = 1,..,2, \\ \frac{dA}{dt} = -k_1(T)A|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ \frac{dB}{dt} = k_1(T)A - k_2(T)B|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ y_1 = \frac{dA}{dT}|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ y_2 = \frac{dB}{dT}|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ \frac{dy_1}{dt} = -k_1(T)y_1 - k_1'(T)A|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ \frac{dy_2}{dt} = k_1(T)y_1 - k_2(T)y_2 - k_1'(T)A - k_2'(T)B|_{\substack{a_1=a_1^n\\a_2=a_2^n\\T=T^n}}, \\ A(0) = A_0,\ B(0) = 0, \\ y_1(0) = 0,\ y_2(0) = 0. \end{cases} \tag{151}$$

From the system above we get unknown differential $y_2(t) = \frac{dB(t,T)}{dT}$ required for the gradient estimation and apply the main formula of the stochastic quasi-gradient method:

$$u^{n+1} = u^n + \rho_n \xi^n, \tag{152}$$

where

$$u^n = \begin{pmatrix} t^n \\ T^n \end{pmatrix}, \tag{153}$$

$$\xi^n = \begin{pmatrix} \frac{dB(t,X^n)}{dt} & \frac{dB(t,X^n)}{dT} \end{pmatrix}^T |_{t=t^n, T=T^n}, \tag{154}$$

$$X^n = \begin{pmatrix} a_1^n \\ a_2^n \end{pmatrix}. \tag{155}$$

For the comparison purposes all tested algorithms started from the same initial point $u^0 = [100\,100]^T$ and the same decreasing step sequence has been used $\rho_n = 20\frac{1}{n^{-0.6}}$.

The next table contains comparison results of the calculations which were conducted on the usual laptop (Processor Intel Celeron B810 1.6GHz core duo with 2GB of RAM). In fact, the fastest were enhanced by an equation in variations the SQG-algorithm. The second place took the SPSA, meaning that in the considered case the function's

evaluation consumed the majority of the time. This result looks reasonable, as in the case of the proposed method in every iteration we were solving only one ODE-system (but extended), for the SPSA-algorithm it was required to solve the ODE-system twice, and for the Kiefer-Wolfowitz method four times.

$Algorithm$	N_{steps}	$Time(sec)$	t	T	t_{opt}	T_{opt}
SQG	100	2.2	20.6	79.1	100.0	18.5
KW		8.4	22.4	60.5		
$SPSA$		4.0	39.1	32.6		
SQG	1000	10.0	21.2	73.3		
KW		47.5	46.6	28.3		
$SPSA$		23.5	44.6	29.3		
SQG	5000	58.1	20.0	65.5		
KW		228.7	94.9	18.7		
$SPSA$		125.3	99.8	18.4		

Table 6: Calculation results

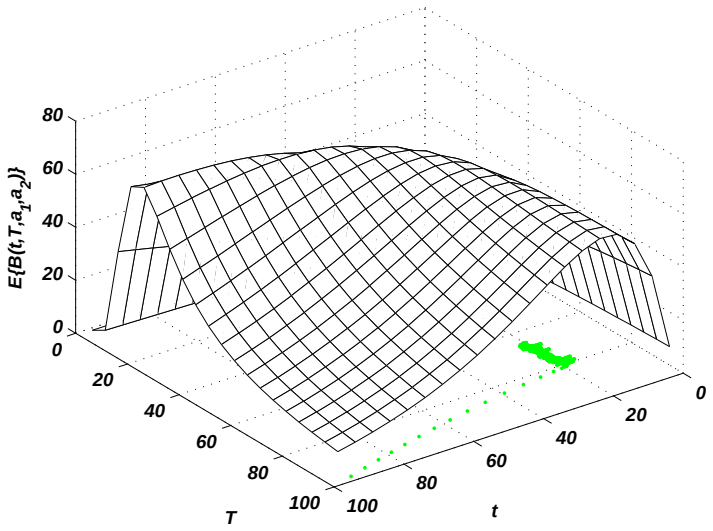

Figure 28: SQG with equation in variations enhancement for the second example.

The next figure depicts how the control variables were changing over the steps.

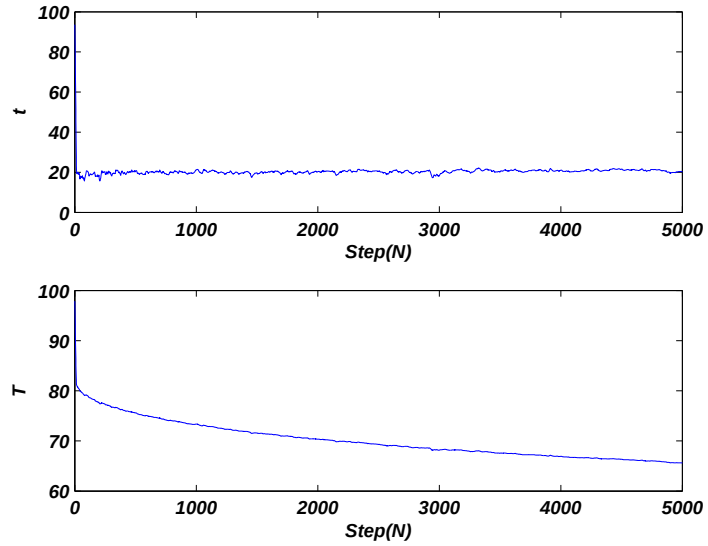

Figure 29: Convergence of the SQG-algorithm for the second example.

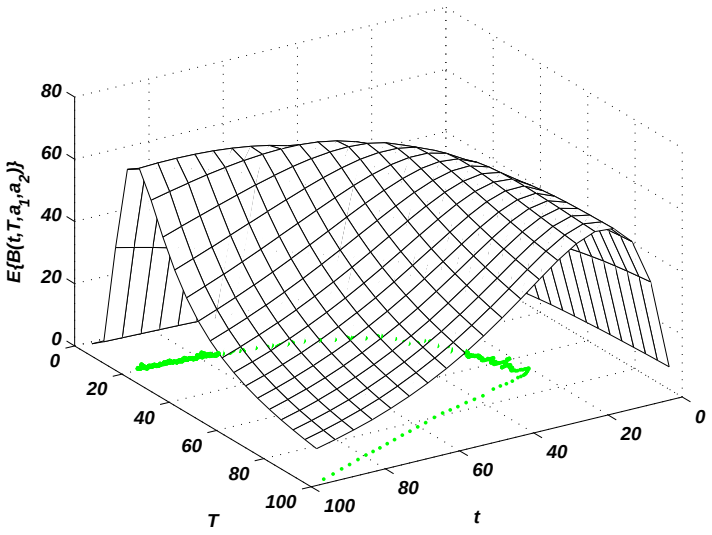

Figure 30: Kefer-Wolfowitz algroithm for the second test-case.

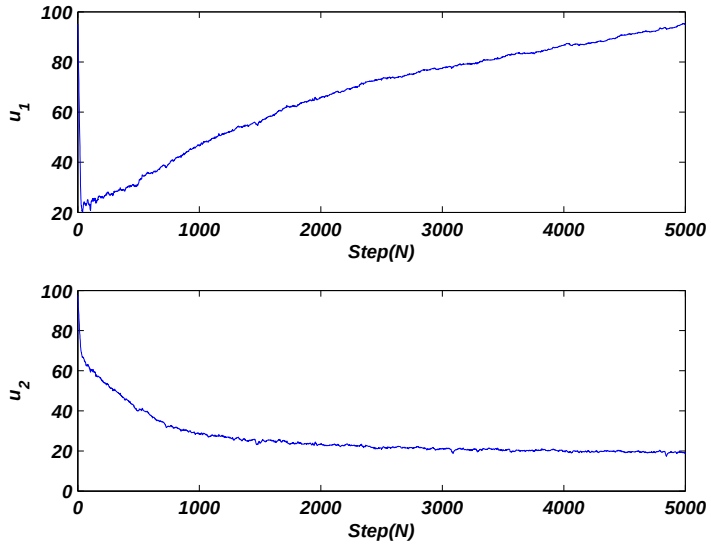

Figure 31: Convergence of the KW-algorithm for the second example.

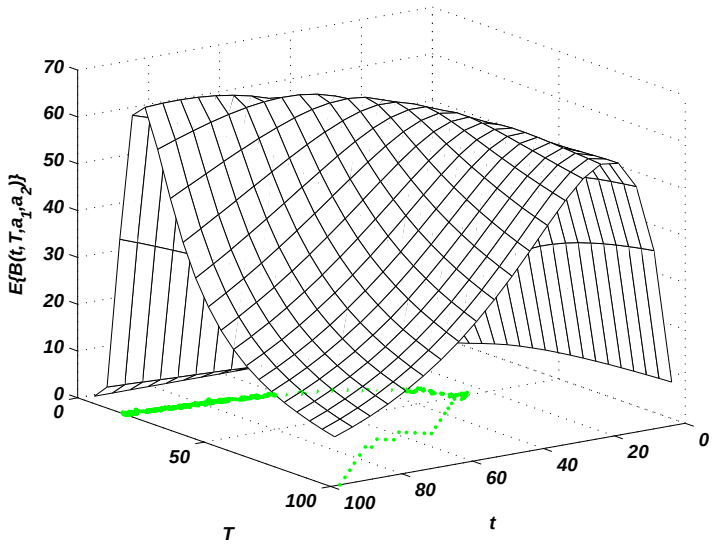

Figure 32: SPSA-algorithm for the second example.

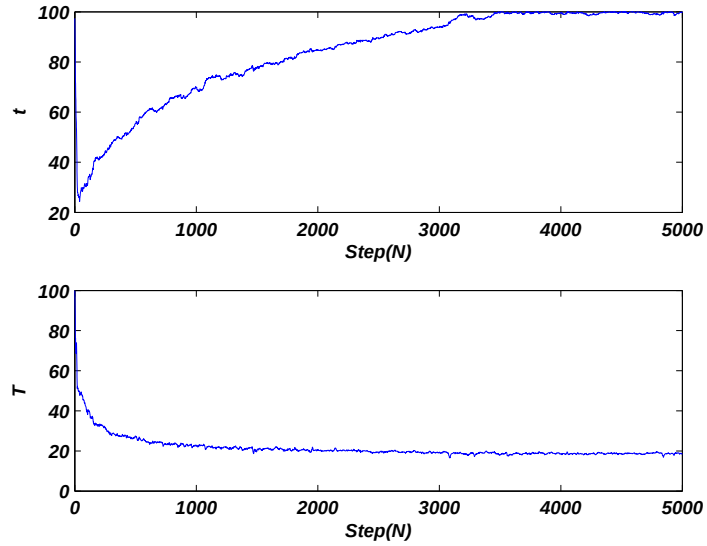

Figure 33: Convergence of the SPSA-algorithm for the second test-case.

5.4 Analysis of the results

As it can be seen from the figures the function $E\{B(t,T,a_1,a_2)\}$ has a complicated structure: it is a ravine and not a concave, therefore the convergence traces have such shape.

Initially all three methods rapidly found the bottom of the ravine and started the path to the optimum but with the different speed.

The reason of such behavior is that the SQG-algorithm with equation in variations enhancement estimates the gradient exactly for the given realization of the random parameters and as at the bottom of the ravine the gradient is very close to zero, the method cannot find the ascent direction and therefore slowly converges to the optimum.

the SPSA and KW-algorithm estimates the gradient based on the values of the objective function (finite differences in the case of the KW and the expression (64) for the SPSA). The distance between the points that will determine the gradient estimate are toughly defined in the algorithms (the sequence $\{\delta_k\}$ in the KW case and the parameters $c_k\Delta_{ki}$, $i = 1, ..., n$ in the SPSA case). Therefore these algorithms can easily find ascent direction and as a result converge faster in that case to the optimal point.

From the table we can also conclude that an application of the equation in variations indeed increases computational speed. Such observation means that in the present case

it is faster to solve one slightly extended ODE-system (to get the unknown differential), than to calculate the values of the objective function twice (to get the finite-difference approximation).

6 Conclusion and main results

The following main results have been achieved in the thesis:

- Review, description and implementation (see an application at the end of the thesis) of the stochastic approximation algorithms for stochastic optimization.
- Acquired new deterministic equivalents (73, 105) for the electricity retailer profit optimization problem with mean-value and CVaR criteria.
- Suggested to use an equation in variations enhancement (143) for the objective functions in the ODE-form, that increases the computational speed of the stochastic approximation procedure.
- Using implemented Matlab-library in the present work it has been found and compared solutions of the stochastic optimization problem with an objective function given in ODE-form with parameter uncertainty given by the sample of posterior distribution.

References

[1] Glachant J-M. and Saguan M. An Institutional Frame to Compare Alternative Market Designs in e U Electricity Balancing, Working Papers 0701, Massachusetts Institute of Technology, Center for Energy and Environmental Policy Research. 2007.

[2] Kettunen J., Salo A. and Bunn D.W. Optimization of Electricity Retailer's Contract Portfolio Subject to Risk Preferences, J. Power Systems, IEEE Transactions on, 2010, No 1.

[3] Mertens A.V. Stochastic Quasi-Gradient Techniques in VaR-based ALM Models. Theory of Stochastic Processes, Vol. 7, no. 1-2, 2001

[4] Kibzun A. and Kan Yu. Stochastic Programming Problems with Probability and Quantile Functions, Chichester: Wiley, 1996.

[5] Gill P. E., Murray W. and Wright M. H. Practical Optimization, Academic Press.

[6] Bergie J. and Louveaux F. Introduction to Stochastic Programming. Springer-Verlag New York, Inc, 1997.

[7] Rockafellar R.T and Uryasev S. Optimization of conditional value-at-risk, J. Risk. No 2, 2000.

[8] Solonen A. and Haario H. Model-based process optimization in the presence of parameter uncertainty, Engineering Optimization, 2011.

[9] Ermoliev Yu. Stochastic Quasi-Gradient Methods and Their Applications to System Optimization. Stochastics, 1983, 4, pp. 1-36.

[10] Pontryagin L.S. Ordinary Differential Equations. Moscow: Nauka, 1974.

[11] Wets R. Stochastic Programming. Handbook for Operations Research and Management Sciences, Vol 1, Elsevier Science Publishers, pp. 573-629. 1989

[12] Wets R. Stochastic Quasi-Gradient Methods. University of California. Feb. 15, 2005.

[13] Miller B.M., Pankov A.R. Teoriya sluchainih processov v primerah i zadachah, Moscow, Fizmatlit, 2002.

[14] Granichin O.N., Polyak B.T. Randomized Algorithms of an Estimation and Optimization Under Almost Arbitrary Noises. – Nauka, Moscow, 2003.

[15] Krasulina T.B. On Robbins—Monro procedure in the case of several roots. Teoriya Verojatnostei i ee Primeneneniya 12, No. 2, 1967.

[16] Haario H., Saksman E. and Tamminen J. Adaptive proposal distribution for random walk Metropolis algorithm. Computational Statistics. 14, 3, p. 375-395, 1999.

[17] Kiefer J., Wolfowitz J. Statistical estimation of the maximum of a regression function. Ann. Math. Statist., 1952, vol 23, p. 462-466

[18] Borodinova I.A., Saraev L.A Stochastic transportation problem, Vestnik SAMGTU, 2010, UDK 519.6.

[19] http://www.jhuapl.edu/spsa/index.html

[20] Robbins H., Monro S. A Stochastic Approximation Method. Ann. Math. Statist., 1951, 22, pp.400–407.

[21] Blum J.R. Multidimensional stochastic approximation. Ann. Math. Statist., 1954, Vol. 9. P. 737-744.

[22] Powell, W. B. Stochastic Approximation Methods, in Approximate Dynamic Programming: Solving the Curses of Dimensionality, John Wiley & Sons, Inc., Hoboken, NJ, USA. 2007.

[23] Naumov A.V. Linear Two-Stage Quantile Optimization Problem. 15th International Simposium on Mathematical Programming.Program and Abstracts. The University of Michigan, USA. August 15-19.

[24] A. V. Naumov and I. M. Bobylev On the two-stage problem of linear stochastic programming with quantile criterion and discrete distribution of the random parameters AUTOMATION AND REMOTE CONTROL Volume 73, Number 2, 265-275

[25] Kibzun A.I., Naumov A.V. Optimal Investment to the Regional Water-Supply Sys- tem // Proceedings of International Conference Mathematics, Computer, Control and Investments.— Russia, Moscow, 1993. — Pp. 72–78.

[26] Pankov A.R., Platonov E.N. Workshop on mathematical statistics (in russian), Moscow, MAI, 2006.

[27] Uryas'ev S., Pulkkinen U. Optimal Operational Strategies for an Inspected Component - Solution Techniques. International Institute for Applied Systems Analysis, Laxenburg, Austria, Report CP-91-13, 1991, 16 p.

[28] Adaptive Algorithms of Stochastic Optimization and Game Theory. Nauka, Moskow, 1990, 182 p. (in Russian).

[29] Granichin O.N., Polyak B.T. Randomized Algorithms of an Estimation and Optimization Under Almost Arbitary Noises. Nauka, Moskow, 2003.

7 Appendices

7.1 Implementation of the stochastic approximation algorithms

```
function [result] = sqggrun(gradientfun,costfun,rhofun,deltafun,
projfun,Nsteps,u0,chain,dim_name,flag_method)

% function result = sqggrun(gradientfun,costfun,rhofun,deltafun,
% projfun,Nsteps,u0,chain,dim_name,flag_method)
% The function calculates an optimal solution of the mean-value minimization
% problem by the stochastic approximation method
%
% INPUT:
%       gradientfun(u,X) - If the stochastic gradient may be
%                          computed explicityly, then costfun and deltafun
%                          can be left empty [] as they are needed only for
%                          the approximation by finite differnces
%
%
%       costfun(u,X)     - Cost function, where u - strategies, X - random
%                          params
%
%       rhofun(k)        - Decreasing step-size function, 1/k is suitable
%
%
%       deltafun(k)      - Decreaing function to compute finite differences
%                          1/k is suitable
%
%       projfun          - Projection onto the admissible set function.
%                          For the given point u it should return an
%                          ortoprojection onto the set U
%
%
%       Nsteps           - Number of steps to calculate
%       u0               - Starting point
%       chain            - array of realizations of the random vector X
%       dim_name         - axis labels for the sqgplot
%
%       flag_method      - SGD: Stochastic Gradient Descent
%                          - KW: Kiefer-Wolfowitz
%                          - SPSA: Simulateneous Perturbation Stochastic
%                            Approximation
%                          - KestenSGD: Stochastic Gradient Descent with
%                            Adaptive Keste Rule
%                          - KestenKW: Kiefer-Wolfowitz with Adaptive
%                            Keste Rule
%                          - UryasevSGD: Stochastic Gradient Descent with
%                            Adaptive Uryasev Rule
%                          - UryasevKW: Kiefer-Wolfowitz with Adaptive
%                            Uryasev Rule
%
% OUTPUT: The sequence of points generated by the algorithm
```

```
u=u0;
dim=size(u0,1);
result(1,:)=u;
h = waitbar(0,'1','Name','Calculation');
```

7.1.1 Implementation of the SQG-algorithm with explicitly defined gradient

```
if (strcmp(flag_method,'SGD'))
    for k=1:Nsteps
        X=chain(k,:);
        u=projfun(u-rhofun(k)*gradientfun(u,X));
        result(k,:)=u;
        if (mod(k,100)==0)
            waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
        end;
    end;
end;
```

7.1.2 Implementation of the SQG-algorithm with explicitly defined gradient and adaptive Kesten rule

```
if (strcmp(flag_method,'KestenSGD'))
    n=1;
    for k=1:Nsteps
        X=chain(k,:);
        u=projfun(u-rhofun(n)*gradientfun(u,X));
        result(k,:)=u;
        if (k>2)
            eps_curr=result(end,:)'-result(end-1,:)';
            eps_prev=result(end-1,:)'-result(end-2,:)';
            %inner product of the current and previous
            if (sum(eps_curr.*eps_prev)<0)
                n=n+1;
            end;
        else
            n=n+1;
        end;
        if (mod(k,100)==0)
            waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
        end;
    end;
end;
```

7.1.3 Implementation of the SQG-algorithm with explicitly defined gradient and adaptive Uryasev rule

```
if (strcmp(flag_method,'UryasevSGD'))
        rho=1;
        X=chain(1,:);
        xi=gradientfun(u,X);
```

```
    result(1,:)=u;
    step(1)=rho;
    for k=2:Nsteps
        X=chain(k,:);
        xi_next=gradientfun(u,X);
        lambda=k^(-1);
        rho=rho+lambda*sum(xi.*xi_next);
        step(k)=rho;
        xi=xi_next;
        u=projfun(u-rho*xi_next);
        result(k,:)=u;
        if (mod(k,100)==0)
            waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
        end;
    end;
    figure(100);
    plot(1:Nsteps,step);
end;
```

7.1.4 Implementation of the Kiefer-Wolfowitz algorithm

```
if (strcmp(flag_method,'KW'))
    n=1;
    for k=1:2:Nsteps
        X1=chain(k,:);
        X2=chain(k+1,:);
        u=projfun(u-rhofun(n)*QGradient(costfun,u,X1,X2,deltafun(n)));

        result(n,:)=u;
        if (mod(k-1,100)==0)
            waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
        end;
        n=n+1;
    end;
end;
```

7.1.5 Implementation of the Kiefer-Wolfowitz algorithm with adaptive Kesten rule

```
if (strcmp(flag_method,'KestenKW'))
    n=1;
    i=1;
    for k=1:2:Nsteps
        X1=chain(k,:);
        X2=chain(k+1,:);
        u=projfun(u-rhofun(n)*QGradient(costfun,u,X1,X2,deltafun(n)));
        result(i,:)=u;
        if (i>2)
            eps_curr=result(i,:)'-result(i-1,:)';
            eps_prev=result(i-1,:)'-result(i-2,:)';
            %inner product of the current and previous
            if (sum(eps_curr.*eps_prev)<0)
```

```
                n=n+1;
            end;
        else
            n=n+1;
        end;
        i=i+1;
        if (mod(k-1,100)==0)
            waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
        end;
    end;
end;
```

7.1.6 Implementation of the Kiefer-Wolfowitz algorithm with adaptive Uryasev rule

```
if (strcmp(flag_method,'UryasevKW'))
    rho=1;
    X1=chain(1,:);
    X2=chain(2,:);
    xi=QGradient(costfun,u,X1,X2,deltafun(1));
    result(1,:)=u;
    step(1)=rho;
    n=2;
    for k=3:2:2*Nsteps
        X1=chain(k,:);
        X1=chain(k+1,:);
        xi_next=QGradient(costfun,u,X1,X2,deltafun(1));
        lambda=n^(-1);
        rho=rho+lambda*sum(xi.*xi_next);
        step(n)=rho;
        xi=xi_next;
        u=projfun(u+rho*xi_next);
        result(n,:)=u;

        if (mod(n,100)==0)
            waitbar(n/Nsteps,h,sprintf('Step: %12.0f',n));
        end;
        n=n+1;
    end;
    figure(100);
    plot(1:Nsteps,step);
end;
```

7.1.7 Implementation of the SPSA algorithm

```
if (strcmp(flag_method,'SPSA'))
    n=1;
    for k=1:2:Nsteps
        X1=chain(k,:);
        X2=chain(k+1,:);
        delta=2*round(rand(dim,1))-1;
        u_plus=u+deltafun(n)*delta;
```

```
            u_minus=u-deltafun(n)*delta;
            QG=(costfun(u_plus,X1)-costfun(u_minus,X2))./(2*deltafun(n)*delta);
            u=projfun(u-rhofun(n)*QG);

            result(n,:)=u;
            if (mod(k-1,100)==0)
                waitbar(k/Nsteps,h,sprintf('Step: %12.0f',k));
            end;
            n=n+1;
        end;
    end;
    delete(h);
    sqgplot(result,dim_name);
```

7.1.8 The function which calculates finite difference approximation of the stochastic quasi-gradient

```
function QG=QGradient(fun,u,sample1,sample2,delta)
    n=size(u,1);
    for i=1:n
        u1=u;
        u2=u;
        u1(i,:)=u1(i,:)+delta;
        u2(i,:)=u2(i,:)-delta;
        QG(i,:)=(fun(u1,sample1)-fun(u2,sample2))/(2*delta);
    end;
```

7.1.9 The function displaying the convergence rate

```
function sqgplot(trace,dim_name);
%plot result
    dim=size(trace(1,:),2);
    steps=size(trace(:,1));
    Ny=ceil(sqrt(dim));
    if (Ny==dim)
        Nx=1;
    else
        Nx=Ny;
    end;
    r=1:steps;
    for k=1:dim
        subplot(Ny, Nx, k);
        plot(r,trace(r,k));
        xlabel('Step(N)');
        ylabel(dim_name(k));
    end;
```

7.1.10 Electricity retailer profit optimization - solution with mathematical expectation.

```
clc;
```

```
clear all;
close all;
c_buy=.4;
c_sell=2.2;
c_left=0.1;
c_def=.3;
alpha=0.6;
beta=0.7;
delta1=40;
delta2=35;
PDF_X=[0.05 0.05 0.05 0.05 0.1 0.2 0.2 0.15 0.1 0.05];
X=[10 20 30 40 50 60 70 80 90 100];
PDF_Y=[0.1 0.2 0.3 0.2 0.15 0.05];
Y=[0.1 0.2 0.3 0.4 0.5 0.6];
CDF_X=cumsum(PDF_X);
CDF_Y=cumsum(PDF_Y);
Num_sim=1000;                                   %number of days to simulate
for u=1:100                                     %try different size of storage
    for ia=1:Num_sim
        req=invcdf(X,CDF_X,1,2);
        %Sample(ia)=req;
        c_buy=invcdf(Y,CDF_Y,1,2);
        Profit1(ia)=-u*c_buy;                   %profit in the beginning
        N_sell=min(req,u);                      %the number he sells
        N_left=max(u-req,0);                    %the number of newsp. left
        N_def=max(req-u,0);                     %deficit papers

        Profit1(ia)=Profit1(ia)+N_sell*c_sell-N_left*c_left-N_def*c_def;
%profit without penalty
    end;
    X1(u)=mean(Profit1);                        %mean value of the profit in case a
end;
spare=X(CDF_X>=alpha);
x_alpha=spare(1);
spare=X(CDF_X>=1-beta);
x_1_beta=spare(1);
i=x_alpha-delta1:1:x_1_beta+delta2;                    %admissible area
%%f=@(u,x)min(x,u)*c_sell-u*y-max(u-x,0)*c_left-max(x-u,0)*c_def; we have
%%split the function on 2 parts
f1=@(u,x)min(x,u)*c_sell-max(u-x,0)*c_left-max(x-u,0)*c_def;
for u=1:100
    S=0;
    for j=1:10
        S=S+f1(u,X(j))*PDF_X(j);
    end;
    R=0;
    for j=1:6
        R=R+u*PDF_Y(j)*Y(j);
    end;
    S=S-R;

    Expectation(u)=S;
end;
```

```
u=1:100;
plot(u,Expectation,'-black',u, X1,'g.',i,-20,'r*'); hold on;
legend('Theoretical result','Simulated result','Admissible area');
xlabel('u');
ylabel('E\{\Phi(u,X,Y)\}');
grid on;
% Theoretical result by the linear programming
n=10;
e=ones(n,1);
I=eye(n,n);
E=zeros(n,n);
A1=[e -I  I  E E
    e  E E  I -I];
b1=[X X]';
c_buy=sum(PDF_Y.*Y);
C=[(c_sell-c_buy)];
for i=1:n
    C=[C -PDF_X(i)*(c_sell+c_left)];
end;
C=[C zeros(1,n)];
for i=1:n
    C=[C -PDF_X(i)*c_def];
end;
C=[C zeros(1,n)];
lb = zeros(4*n+1,1);
A2=[-1   zeros(1,4*n)
    1 zeros(1,4*n)];
b2=[x_alpha-delta1
    x_1_beta+delta2];
[x,fval,exitflag,output,lambda]  = linprog(-C,A2,b2,A1,b1,lb);
u=x(1)
f=-fval
plot(u,f,'b.','markersize',20);
```