VERSITA

TATRA MOUNTAINS
Mathematical Publications

DOI: 10.2478/v10127-012-0002-3
Tatra Mt. Math. Publ. **51** (2012), 11–21

ON COMPUTATIONAL COMPLEXITY OF CONSTRUCTION OF c-OPTIMAL LINEAR REGRESSION MODELS OVER FINITE EXPERIMENTAL DOMAINS

Jaromír Antoch — Michal Černý — Milan Hladík

ABSTRACT. Recent complexity-theoretic results on finding $\boldsymbol{c}$-optimal designs over finite experimental domain $\mathcal{X}$ are discussed and their implications for the analysis of existing algorithms and for the construction of new algorithms are shown. Assuming some complexity-theoretic conjectures, we show that the approximate version of $\boldsymbol{c}$-optimality does not have an efficient parallel implementation. Further, we study the question whether for finding the $\boldsymbol{c}$-optimal designs over finite experimental domain $\mathcal{X}$ there exist a strongly polynomial algorithms and show relations between considered design problem and linear programming. Finally, we point out some complexity-theoretic properties of the SAC algorithm for $\boldsymbol{c}$-optimality.

1. Introduction

Consider the linear regression model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{1}$$

where $\boldsymbol{y}$ is the dependent variable, $\boldsymbol{X}$ is the design matrix and $\boldsymbol{\beta} \in \mathbb{R}^m$ is an unknown vector of regression parameters. Under traditional assumptions on the vector of errors $\boldsymbol{\varepsilon}$ the ordinary least squares (OLS) estimator $\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$ of $\boldsymbol{\beta}$ is the best linear unbiased estimator. Let $\boldsymbol{c} \in \mathbb{R}^m$ be a fixed non-zero vector. Then an estimator of $\boldsymbol{c}'\boldsymbol{\beta}$ is usually defined as $\boldsymbol{c}'\widehat{\boldsymbol{\beta}}$, see [14] for details. It is easy to show that this estimator is unbiased.

2010 Mathematics Subject Classification: Primary 62K05; Secondary 68Q25.
Keywords: optimal design, $\boldsymbol{c}$-optimality, P-completeness, parallel computation, linear programming, SAC algorithm.
The first author was supported by grant 201/09/0755 of the Czech Science Foundation. The second and the third author were supported by grant P403/12/1947 of the Czech Science Foundation. The second author was also supported by grant F4/18/2011 of the Internal Grant Agency of University of Economics, Prague.

If we can choose the values of regressors, i.e., if we can control the design matrix $\boldsymbol{X}$, then there appears a naturally associated optimization problem: *How should the design matrix be chosen so that the estimator $\widehat{\boldsymbol{\beta}}$ is "as good as possible"?* Usually there exist restrictions for the choice of the design matrix. For example:

(i) A set $\mathcal{X} \subseteq \mathbb{R}^m$, called the *experimental domain*, is given and for each row $\boldsymbol{x}'$ of $\boldsymbol{X}$ it holds $\boldsymbol{x} \in \mathcal{X}$.

(ii) The number of rows is bounded by a number N. This corresponds to the situation that we are allowed to make at most N observations. Etc.

The statement "to choose $\boldsymbol{X}$ so that the estimator $\widehat{\boldsymbol{\beta}}$ is as good as possible" may be formalized in different ways according to various optimality criteria. There exists an abundant theory on this issue, see [2], [12], [17], for examples.

In this paper we deal with the criterion known as $\boldsymbol{c}$-optimality. Said loosely, a design matrix $\boldsymbol{X}$ is *$\boldsymbol{c}$-optimal* if it respects restrictions of the type (i) and (ii) and minimizes the variance of $\boldsymbol{c}'\widehat{\boldsymbol{\beta}}$. A more precise statement of the $\boldsymbol{c}$-optimality problem will be given in Section 1.2.

It is well known that computational complexity of finding $\boldsymbol{c}$-optimal design depends on the experimental domain $\mathcal{X}$. In this paper we assume the case of $\mathcal{X}$ finite. This situation occurs, e.g., if changes in the experimental conditions can be made in discrete steps only, or if the domain $\mathcal{X}$ is so intricate that only a description of $\mathcal{X}$ in terms of a (sufficiently dense) grid is available, or if the domain points have been generated by some kind of a nondeterministic process. Recall that the finite case is very important in practice as any compact set can be approximated using a grid consisting of finitely many points.

1.1. Statistical formalization of the problem

Let the experimental domain $\mathcal{X} := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k\}$ be fixed.

Definition 1.

(a) A vector $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_k)'$ satisfying $\mathbf{1}'\boldsymbol{\xi} = 1$ and $\boldsymbol{\xi} \geq \mathbf{0}$ (i.e., a discrete probabilistic measure on $\mathcal{X}$) is called an *approximate design*, or *design* for short.

(b) If a natural number N is given and the numbers $N\xi_1, \ldots, N\xi_k$ are integers, then the approximate design $\boldsymbol{\xi}$ is called an *N-exact design*.

The design $\boldsymbol{\xi}$ has the meaning that we shall perform $100\xi_i\,\%$ of measurements in the point $\boldsymbol{x}_i \in \mathcal{X},\ i = 1, \ldots, k$.

The number N in (b) stands for the number of observations to be made. We say that an N-exact design $\boldsymbol{\xi}$ *determines* an N-row design matrix $\boldsymbol{X}$ if $\boldsymbol{X}$ is,

up to a permutation of rows, of the form

$$\boldsymbol{X} = (\underbrace{\boldsymbol{x}_1\ \boldsymbol{x}_1\ \cdots\ \boldsymbol{x}_1}_{N\xi_1 \text{ times}}\ \underbrace{\boldsymbol{x}_2\ \boldsymbol{x}_2\ \cdots\ \boldsymbol{x}_2}_{N\xi_2 \text{ times}}\ \cdots\ \underbrace{\boldsymbol{x}_k\ \boldsymbol{x}_k\ \cdots\ \boldsymbol{x}_k}_{N\xi_k \text{ times}})'. \tag{2}$$

For a given N-exact design $\boldsymbol{\xi}$ the variance of $\boldsymbol{c}'\widehat{\boldsymbol{\beta}}$ can be decomposed as

$$\operatorname{var}(\boldsymbol{c}'\widehat{\boldsymbol{\beta}}) = \frac{\sigma^2}{N} \cdot \operatorname{var}_{\boldsymbol{c}}(\boldsymbol{\xi}), \tag{3}$$

where σ^2 is the variance of error terms, $\widehat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$ with the N-row design matrix $\boldsymbol{X}$ determined by $\boldsymbol{\xi}$ according to (2) and the number $\operatorname{var}_{\boldsymbol{c}}(\boldsymbol{\xi})$, called $\boldsymbol{c}$-*variance* of $\boldsymbol{\xi}$, depends on $\boldsymbol{\xi}$ but not on N and σ^2. The number $\operatorname{var}_{\boldsymbol{c}}(\boldsymbol{\xi})$, which is implicitly defined by (3), represents the (multiplicative) contribution of the structure of the design $\boldsymbol{\xi}$ on the variance of $\boldsymbol{c}'\widehat{\boldsymbol{\beta}}$. Thus, $\operatorname{var}_{\boldsymbol{c}}(\boldsymbol{\xi})$ is a natural measure of the quality of the design $\boldsymbol{\xi}$.

Definition 2. N-exact design $\boldsymbol{\xi}$ is said to be $\boldsymbol{c}$-*optimal* if for any other N-exact design $\widetilde{\boldsymbol{\xi}}$ it holds $\operatorname{var}_{\boldsymbol{c}}(\widetilde{\boldsymbol{\xi}}) \geq \operatorname{var}_{\boldsymbol{c}}(\boldsymbol{\xi})$.

Aside the exact version we shall also investigate its *approximate*, i.e., *asymptotic*, counterpart, which relaxes the N-exactness requirement and corresponds to the situation when we can make any finite number of observations.

Definition 3. An approximate design $\boldsymbol{\xi}$ is said to be *asymptotically* $\boldsymbol{c}$-*optimal*, if for any N such that $\boldsymbol{\xi}$ is N-exact it is $\boldsymbol{c}$-*optimal*.

The term "asymptotic" reflects the fact that the least N, for which the approximate design $\boldsymbol{\xi}$ is N-exact, may be large. Moreover, note that the definition of asymptotic $\boldsymbol{c}$-optimality is interesting for the rational vectors $\boldsymbol{\xi}$ only, so that for the complexity-theoretic considerations only rational numbers will be taken into account.

Remark 1. Consider a finite experimental domain $\mathcal{X} \subseteq \mathbb{R}^\ell$ and the regression model

$$y_i = f(\boldsymbol{x}_i)'\boldsymbol{\beta} + \varepsilon_i, \qquad i = 1, \ldots, N, \tag{4}$$

where $f : \mathbb{R}^\ell \longrightarrow \mathbb{R}^m$ is a function of the form $f = (f_1, \ldots, f_m)'$ and the functions $f_1, \ldots, f_m : \mathbb{R}^\ell \longrightarrow \mathbb{R}$ are linearly independent. This model is more general than the basic model (1). Observe that in our setting the model (4) with the domain $\mathcal{X}$ is equivalent to the basic model (1) with another domain $\mathcal{X}'$. For example, consider $\ell = 1$ and $f(x) = (1, x, x^2)'$. Then, finding a good design over the experimental domain $\mathcal{X} = \{x_1, \ldots, x_k\}$ in the model (4) is equivalent to finding a good design over the domain $\mathcal{X}' = \{(1, x_1, x_1^2)', \ldots, (1, x_k, x_k^2)'\}$ in the model (1). Hence, if we take the more general model (4) instead of (1) as the basis for presentation, we get the same results.

1.2. Complexity-theoretic formalization of the problem

In statistics, the $\boldsymbol{c}$-optimality problem can be described as follows. *Given a rational experimental domain $\mathcal{X}$, a rational vector $\boldsymbol{c}$ and a nonzero natural number N,* **find** *the $\boldsymbol{c}$-optimal N-exact design $\boldsymbol{\xi}$ over the domain $\mathcal{X}$.*

As pointed out above, the main aim of this paper is to study corresponding complexity-theoretic properties. To achieve this goal, we need another formulation of the design problem because complexity theory predominantly deals with *decision problems* having a single bit answer (YES/NO). The natural decision version of the exact design problem is therefore as follows:

EOD Given a rational experimental domain $\mathcal{X}$, a rational vector $\boldsymbol{c}$, a natural number N and a positive rational number S^2, **decide** whether there exists an N-exact design $\boldsymbol{\xi}$ over the domain $\mathcal{X}$ satisfying $\mathrm{var}_{\boldsymbol{c}}(\boldsymbol{\xi}) \leq S^2$. Or, equivalently, decide whether the $\boldsymbol{c}$-variance of the $\boldsymbol{c}$-optimal N-exact experimental design over $\mathcal{X}$ does not exceed S^2. (An instance of the problem ***EOD*** is denoted by $[\mathcal{X}, \boldsymbol{c}, N, S^2]$ in what follows.)

Observe that $\mathrm{var}_{\boldsymbol{c}}(\boldsymbol{\xi})$ is proportional to the precision of measurement. Hence, the instance $[\mathcal{X}, \boldsymbol{c}, N, S^2]$ of the problem ***EOD*** can be also understood in the following way. "Is it possible to design an N-exact experiment over $\mathcal{X}$ such that $\boldsymbol{c}'\boldsymbol{\beta}$ can be measured with precision not worse than S^2"?

It is also natural to consider the following decision formulation of the approximate (asymptotic) version of the design problem:

AOD Given a rational experimental domain $\mathcal{X}$, a rational vector $\boldsymbol{c}$ and a positive rational number S^2, decide whether there exists an asymptotically $\boldsymbol{c}$-optimal design $\boldsymbol{\xi}$ over $\mathcal{X}$ satisfying $\mathrm{var}_{\boldsymbol{c}}(\boldsymbol{\xi}) \leq S^2$. In other words, if we are not restricted by the number of observations, decide whether it is possible to design an experiment over $\mathcal{X}$ with measurement precision not worse than S^2. (An instance of the problem ***AOD*** is denoted by $[\mathcal{X}, \boldsymbol{c}, S^2]$ in what follows.)

2. Computational complexity of *EOD* and *AOD*

Definitions of all of the complexity-theoretic notions used in the following text can be found in [1], [6], [10], [11].

From the definition of the problems ***EOD*** and ***AOD*** we immediately get the following observation:

Proposition 1. (a) *The problem **EOD** is decidable in exponential time.*

(b) *The problem **AOD** is Π_1 in arithmetic (co-recursively enumerable).*

The statement (b) does not guarantee decidability of ***AOD***. It would be tempting to try to prove its undecidability. However, a surprising fact holds. The set ***AOD*** is decidable and moreover:

Theorem 1. ***AOD*** $\in$ ***P***.

On the other hand, the statement (a) probably cannot be improved.

Theorem 2. *The set* ***EOD*** *is* ***NP****-complete.*

Theorem 1 is an important consequence of optimization-theoretic characterization of asymptotic ***c***-optimality shown in [7], which will be discussed in Section 3.2. Theorem 2 was proved in [3]; some of its consequences can be found in [4].

Theorems 1 and 2 show that N-exactness—seemingly just a technical requirement—makes the design problem extremely difficult from the computational perspective compared to the approximate version. Said otherwise: while we have fast algorithms for the approximate version, we cannot expect that we could find a faster-than-exponential algorithm for the exact version. In the asymptotic version of the problem we are able to process huge models with huge experimental domains, but nothing similar can be expected to hold in general for the exact version.

This is a theoretical justification of a phenomenon, observed by many practitioners, that finding good exact designs is a much harder problem (from computational perspective) than finding good approximate designs.

Theorems 1 and 2 also show that there is no efficient (i.e., computationally fast) rounding strategy which would be able to convert an asymptotically optimal design to the optimal N-exact design. More discussion on interesting rounding strategies can be found in [13].

3. Algorithmic properties of *AOD*

In this section we shall inspect more complexity-theoretic properties of the approximate (asymptotic) ***c***-optimal design problem, formalized as ***AOD***, and show their implications for design of algorithms.

3.1. Parallel computability of *AOD*

Once the set ***AOD*** is proved to be in ***P***, more questions arise. The most interesting question is whether ***AOD*** can be put into the ***NC***-hierarchy. Said roughly, problems in ***NC*** are problems decidable in polylogarithmic parallel time (for a rigorous definition see [1] or [11]); that is, they can be decided significantly faster than in polynomial time using parallel architectures. Recall also that it is known that ***NC*** $\subseteq$ ***P***, but it is an open complexity-theoretic question whether the inclusion is proper. The problem ***NC*** $=^?$ ***P*** is as significant question as the well-known problem ***P*** $=^?$ ***NP***. Generally it is believed that ***NC*** $\neq$ ***P***.

Assuming $\boldsymbol{NC} \neq \boldsymbol{P}$, there are problems in $\boldsymbol{P} \setminus \boldsymbol{NC}$. We will use the following terms.

Definition 4.

(a) A problem in $\boldsymbol{NC}$ is said to have *an efficient parallel implementation*.

(b) A problem in $\boldsymbol{P} \setminus \boldsymbol{NC}$ is said to be *hard to parallelize* or *inherently sequential*.

Said informally, if we want to parallelize an inherently sequential problem, we can expect that we do not achieve better than polynomial speedup. In the other words, such a problem cannot be solved by parallel computers significantly faster than using sequential computers.

3.1.1. The question $\boldsymbol{AOD} \in^{?} \boldsymbol{NC}$

The question $\boldsymbol{AOD} \in^{?} \boldsymbol{NC}$ means: *Does the approximate* $\boldsymbol{c}$*-optimality problem have an efficient parallel implementation? Or is it hard to parallelize?* Theorem 3, coming from [3], gives a strong evidence that the latter statement is true.

Theorem 3. *The problem* $\boldsymbol{AOD}$ *is* $\boldsymbol{P}$*-complete.*

Recall that all $\boldsymbol{P}$-complete problems are in $\boldsymbol{P} \setminus \boldsymbol{NC}$ (assuming $\boldsymbol{NC} \neq \boldsymbol{P}$) as it is shown in Figure 1, where the topology of $\boldsymbol{P}$ is depicted.

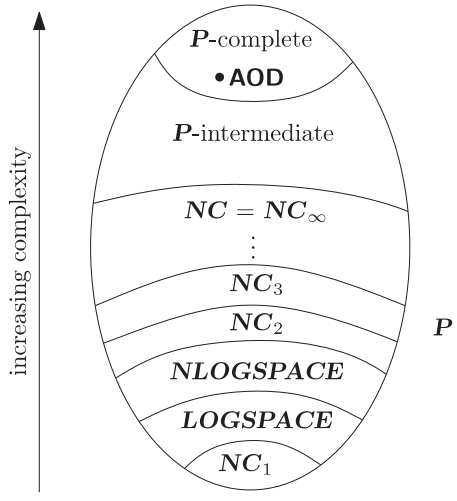

Figure 1. The $\boldsymbol{NC}$-hierarchy in $\boldsymbol{P}$ and $\boldsymbol{P}$-complete problems (assuming $\boldsymbol{P} \neq \boldsymbol{NC}$).

There are many known $\boldsymbol{P}$-complete problems, e.g.:

- deciding whether a formula in the Horn Normal Form is satisfiable;
- deciding whether a given point is a convex combination of a given set of points;
- linear programming in the following version:

  decide whether a rational linear system $\boldsymbol{Ax} \leq \boldsymbol{b}$ has a rational solution; (5)

and a lot of others. For a thorough study of $\boldsymbol{P}$-completeness see [6].

$\boldsymbol{P}$-complete problems can be also seen as *universal* problems for the class $\boldsymbol{P}$. Here, universality is understood in a similar way as in the case of $\boldsymbol{NP}$-completeness: $\boldsymbol{NP}$-complete problems can be understood as universal problems for $\boldsymbol{NP}$. It follows that the entire class $\boldsymbol{P}$ can be seen as a class of particular sets of instances of the design problem ***AOD***.

With Theorem 3 we can conclude that if $\boldsymbol{P} \neq \boldsymbol{NC}$, then ***AOD*** does not have an efficient parallel implementation. In other words, any attempt to find an efficient parallel algorithm for the approximately $\boldsymbol{c}$-optimal design problem is an attempt to prove $\boldsymbol{P} = \boldsymbol{NC}$, i.e., it is an attempt to resolve a longstanding open complexity-theoretic question which is considered to be extremely hard.

3.2. *AOD* and linear programming

In this section we discuss the interesting fact that the problem ***AOD*** is equivalent to the general linear programming. The equivalence should be understood in the way that the problem ***AOD*** and the linear programming problem formalized as (5) are reducible[1] to each other:

(R1) any instance of ***AOD*** can be transformed to a particular linear program;

(R2) any instance of linear programming can be transformed to a particular instance of ***AOD***.

The reduction (R1) was constructed in [7] and the reduction (R2) in [3]. The existence of these reductions is the essence of proofs of Theorems 1 and 3.

Let us emphasize that the reduction (R1) is of practical importance because we can use linear programming solvers for finding solutions of the approximate design problem.

[1]To be precise, the two problems are reducible to each other under ***LOGSPACE***-computable reducibilities.

On the other hand, the existence of the reduction (R2) is of theoretical importance. It is a complexity-theoretic hardness result showing that the problem ***AOD*** is at least as hard as general linear programming, or that general linear programming can be seen as a sub-problem of ***AOD***. It follows that any algorithm for ***AOD*** is necessarily also an algorithm for general linear programming (modulo the reduction (R2)). This is an important information for a designer of algorithms for the approximate design problem: she/he must keep in mind that designing an algorithm for ***AOD*** amounts to designing a competitor to the well-known algorithms for general linear programming such as Mehrotra's Predictor-Corrector Method. It follows that any designer of an algorithm for ***AOD*** has really strong competitors!

3.3. The SAC algorithm

In [7], a special version of the Simplex Algorithm adapted for the $\boldsymbol{c}$-optimality problem, called SAC, has been proposed. The algorithm leans on the reduction (R1). The authors raised a question whether SAC is a polynomial-time method. The theory of Section 3.2 shows that the answer is probably negative. By (R2), SAC is necessarily an algorithm for general linear programming. If SAC runs in polynomial time, then there is a version of the Simplex Algorithm running in polynomial time. This would be a fascinating result — a question on the existence of a polynomial-time version of the Simplex Algorithm is one of the most famous questions in optimization, which has been open since the publication of the celebrated paper [9] forty years ago.

Open problem. Prove that SAC is not a polynomial time method. Find an infinite sequence of instances of the problem ***AOD*** forcing SAC to perform exponential number of steps.

3.4. Strongly polynomial algorithms for *AOD*

Recall that an algorithm processing a sequence of rational numbers $x_1, \ldots, x_n$ is *strongly polynomial* if it is polynomial and the number of arithmetical operations $+, -, \times, \div, \leq$ can be bounded by $p(n)$, where p is a polynomial.

Recall also that the bit-size of a rational number is the number of bits necessary to write down the numerator and the denominator. Observe that a polynomial algorithm processing rational numbers has the property that *any rational number occurring within the computation has bit-size bounded by a polynomial in bit-sizes of* $x_1, \ldots, x_n$ (as arithmetical operations with exponentially long numbers take exponential time).

The Gaussian Elimination of an $(n \times n)$-matrix is an interesting example. Though it requires $O(n^3)$ arithmetical operations, which is a polynomial number, naive implementations are not polynomial-time algorithms. The reason is

that bit-sizes of rational numbers occurring within the computation grow exponentially. If we use the Euclidean Algorithm transforming fractions into coprime forms after each arithmetical operation, then we get an algorithm which is polynomial (with the Euclidean Algorithm, the bit-sizes of numbers can be proved to be bounded polynomially, see [16]) but not strongly polynomial. The problem is that the Euclidean Algorithm itself is not a strongly polynomial procedure. It processes two integers (which is a constant number) and if it were a strongly polynomial procedure, then it would be allowed to perform only a constant number of divisions.

The example of Gaussian Elimination shows what kinds of problems are faced when strongly polynomial algorithms are designed. For the sake of completeness let us conclude that a strongly polynomial implementation of Gaussian Elimination is known; see [5].

Open problem. Does ***AOD*** have a strongly polynomial algorithm? In other words, is there a polynomial-time algorithm for ***AOD*** having the number of elementary arithmetical operations bounded only by the number of regression parameters m and by the cardinality of the experimental domain k, independent of the bit-sizes of rational numbers in the instance $[\mathcal{X}, \boldsymbol{c}, S^2]$?

The problem ***AOD*** is equivalent to general linear programming in the sense described in Section 3.2. By inspection of the reductions (R1) and (R2) from that Section we can show:

Proposition 2. ***AOD*** *has a strongly polynomial algorithm if and only if general linear programming does.*

Remind that no one of known version of the Simplex Algorithm is polynomial. It follows that the Simplex Algorithm is not strongly polynomial. (Recall that in Section 3.3 we conjectured that neither SAC is.) All of the known polynomial-time methods for linear programming, such as Khachiyan's Ellipsoid Algorithm [8], Karmarkar's Algorithm [15] as well as all known Interior Point Methods [15] suffer from the drawback that the number of iterations grows when the bit-sizes of rational numbers in their data grow. It follows that the algorithms are not strongly polynomial.

Existence of strongly polynomial algorithms for general linear programming is a major, long-standing open question in optimization. Thus we may read the equivalence of Proposition 2 also negatively:
If we want to design a strongly polynomial algorithm for ***AOD****, we are in fact attacking a hard open problem.*

As far as we know, there is no general consensus whether a strongly polynomial algorithm for linear programming exists or not. Hence we cannot conjecture whether a strongly polynomial algorithm for ***AOD*** exists.

REFERENCES

[1] ARORA, S.—BARAK, B.: *Computational Complexity. A Modern Approach*. Cambridge University Press, Cambridge, 2009.

[2] ATKINSON, A.—DONEV, A.—TOBIAS, R.: *Optimum Experimental Designs with SAS*. Oxford University Press, Oxford, 2007.

[3] ČERNÝ, M.—HLADÍK, M.: *Two complexity results on c-optimality in experimental design*, Computational Optimization and Applications **51** 2012, 1397–1408, `http://www.springerlink.com/content/115ttx6lu150434k/fulltext.pdf`

[4] ČERNÝ, M.—HLADÍK, M.—SKOČDOPOLOVÁ, V.: *On computationally complex instances of the c-optimal experimental design problem: Breaking RSA-based cryptography via c-optimal designs,* in: Proc. of 19th Internat. Conference on Comput. Statist.––CompStat '10 (Y. Lechevallier, G. Saporta, eds.), Paris, France, Physica Verlag, Heidelberg, 2010, pp. 879–886.

[5] EDMONDS, J.: *Systems of distinct representatives and linear algebra*, J. Res. Natl. Bur. Stand., Sec. B **71B** (1967), 241–245.

[6] GREENLAW, R.—HOOVER, H.—RUZZO, W.: *Limits to Parallel Computation. P-completeness Theory*. Oxford University Press, Oxford, 1995.

[7] HARMAN, R.—JURÍK, T.: *Computing c-optimal experimental designs using the simplex method of linear programming*, Comput. Statist. Data Anal. **53** (2008), 247–254.

[8] KHACHIYAN, L.: *A polynomial algorithm for linear programming*, Dokl. Akad. Nauk SSSR **244** (1979), No. 5, 1093–1096.

[9] KLEE, V.—MINTY, G. J.: *How good is the simplex algorithm? Inequalities III,* in: Proc. of the 3rd Symposium on Inequalities held at the University of California, Los Angeles, Calif., 1969 (O. Shisha, ed.), Academic Press, New York, 1972, pp. 159–175.

[10] ODIFREDDI, P.: *Classical Recursion Theory. Volume I*. Stud. Logic Found. Math., Vol. 125, Elsevier, Amsterdam, 1999.

[11] PAPADIMITRIOU, C.: *Computational Complexity*. Addison-Wesley, Longman, 1995.

[12] PÁZMAN, A.: *Foundations of Optimum Experimental Design*. Reidel Publ. Comp., Dordrecht, 1986.

[13] PUKELSHEIM, F.—RIEDER, S.: *Efficient rounding in approximate designs*, Biometrika **79** (1992), 763–770.

[14] RAO, C. R.: *Linear Statistical Inference and its Applications*. John Wiley & Sons, New York, 1973.

[15] ROOS, C.—TERLAKY, T.—VIAL, J.-P.: *Interior Point Methods for Linear Optimization*. Springer, Heidelberg, 2006.

[16] SCHRIJVER, A.: *Theory of Linear and Integer Programming*. John Wiley & Sons, New York, 2000.

[17] WHITTLE, P.: *Some general points in the theory of optimal experimental design*, J. Roy. Statist. Soc. Ser. B. **35** (1973), 123–130.

Received November 7, 2011

Jaromír Antoch
Charles University of Prague
Department of Probability and
Mathematical Statistics
Sokolovská 83
CZ–186-75 Prague 8
CZECH REPUBLIC
E-mail: antoch@karlin.mff.cuni.cz

Michal Černý
University of Economics Prague
Department of Econometrics
Winston Churchill Square 4
CZ–130-67 Prague
CZECH REPUBLIC
E-mail: cernym@vse.cz

Milan Hladík
Charles University of Prague
Department of Applied Mathematics
Malostranské náměstí 25
CZ–118-00 Prague 1
CZECH REPUBLIC
E-mail: milan.hladik@mff.cuni.cz