Информационные процессы, Том 15, № 3, 2015, стр. 351–376

Separate Testing Inputs vs. Linear Programming Relaxation

M. Malyutov*, P. Grosu* and H. Sadaka*

Mathematics Dept., Northeastern University, 360 Huntington Ave., Boston, MA 02115, USA E-mail: m.malioutov@neu.edu, pgrosu@gmail.com, h.sadaka@neu.edu

Received September, 15, 2015

Abstract—The theory of Compressed Sensing (highly popular in recent years) has a close relative that was developed around thirty years earlier and has been almost forgotten since – the *design of screening experiments*. For both problems the main assumption is *sparsity of active inputs*, and the fundamental feature in both theories is the *threshold phenomenon*: reliable recovery of sparse active inputs is possible when the rate of design is less than the so-called capacity threshold, and impossible with higher rates.

Another close relative of both theories is *multi-access information transmission*. A collection of tight and almost tight screening capacity bounds for *non-adaptive* experimental designs were obtained by 1980. We compare here the simulated capacity and operation time of two analysis methods: (i) linear programming relaxation methods used in compressed sensing, and (ii) separate testing of inputs for non-adaptive strategies. The parallel implementation of the latter allows *fast processing of high-dimensional models*.

KEYWORDS: Random design, active inputs, separate testing of inputs, sparsity, linear programming relaxation, capacity, statistical simulation, parallel computing, factorial models.

1. Outline of history and content

The idea of using 'sparsity' of inputs actively influencing various phenomena appears repeatedly throughout a diverse range of applications in fields from computational biology to machine learning and engineering. Notably, [11] used the assumption of sparsity of contaminated blood donors to dramatically reduce the number of experiments needed to **adaptively** screen them out. Troubleshooting complex electronic circuits using a non-adaptive identification scheme was considered in [35] under the assumption that only a few elements (a sparse subset) become defective. A recent application of these ideas enabled affordable genetic screening to successfully eliminate lethal genetic diseases prevalent in an orthodox Jewish community in New York city, as described in [12].

Successful optimization of industrial output for **dozens of real world applications** was reported in [5]. They used the sparsity assumption of non-negligible (active) coefficients in the multivariate regression model of second order, randomized design, and proposed the **Random Balance Method** (RBM) of analysis of outputs. Their method of analysis was essentially, a visual greedy inspection of scatterplots to identify the most active inputs and consequent optimization of the model with only active inputs influencing the output.

The RBM was inspired by the celebrated *Fisher's idea of randomization* which has been proposed to get rid of the undesirable bias of estimates due to hidden lurking variables at the expense of their somewhat larger standard deviation. Presently, the randomization is a necessary requirement in planned experiments ([27], chapter 3).

The RBM was officially buried by the leading Western statisticians (G.E.P. Box, D.R. Cox, O. Kempthorne, W. Tukey et al) in their unjustified disparaging discussion following the publication of [5] in the first vol-

ume of Technometrics. They especially ridiculed the analysis of results different from the conventional Least Squares (LS) method, and the random design of less support cardinality than the total number of parameters which later was proved not only appropriate but even asymptotically optimal for sparse models! As a result, F.E. Satterthwaite, the author of RBM, suffered a breakdown and was confined for the rest of his life to a psychiatric clinic. In the A. N. Kolmogorov's lab, the team of his deputy - gifted self-taught applied statistician V. V. Nalimov, tested the RBM on real and simulated problems [32] which showed RBM's remarkable efficiency and leaded to the claim in [28]: 'This method is the triumph of psycho-physiological experimental intuition. Mathematicians will never understand the reasons for its effectiveness!' It sounded like challenge! As a reply, L.D. Meshalkin soon published in [26] a combinatorial result which in an idealized situation implied that RBM can work. Two years later [25] explained the information-theoretic grounds of the [26]'s result.

Ironically enough, W. Tukey, one of the RBM's harshest critics, soon after the aforementioned fatal discussion paper, became a supervisor of an undergraduate student named David Donoho in Princeton. After more than forty years since that sad story, Donoho initiated and successfully marketed an enthusiastically accepted revival of related ideas under the name 'Compressed Sensing' for estimation in over-parameterized linear models with L_1-penalty analysis (basis pursuit). It was also proposed around the same time in [33] under the name of LASSO for related statistical problems. Neither of the authors were aware apparently of the connection to the RBM method. Thanks to D. Donoho's popularity, sparsity is now a well established assumption in statistical applications!

The threshold phenomenon was observed in [10] as a result of intensive simulation performed for randomized designs. Connection to Shannon's celebrated justification for a closely related phenomenon arising in information transmission using his notion of the **channel capacity** started to be noticed only later. Numerous publications on compressive sensing during the last decade are listed on www.dsp.ece.rice.edu/cs.

Asymptotically sharp capacity bounds inspired by the *Multi-Access Capacity Region construction* were obtained in [18,20] for brute-force (BF) analysis and in [17,19] for separate testing of inputs methods (STI) which replaced the visual inspection of scatterplots in [5] with the Empirical Shannon Information (ESI) maximization criterion first introduced in [7] for conventional communication problem. These bounds can greatly enhance current understanding of the threshold phenomenon in sparse recovery.

These bounds are obtained for *asymptotically optimal designs* (which turn out to include *random designs*) and thus imply *upper bounds* for the performance of recovery under *arbitrary designs*. Since R. Fisher, random designs are a simple natural choice for planned experimental design enabling bias reduction due to lurking variables [27], and they additionally enable the effectiveness of the STI analysis of asymptotically optimal ESI maximization between inputs and the output. The STI replaces the visual inspection in RBM [5]. The STI and greedy STI capacities *outperform* that of Linear Programming relaxation for randomized designs in a wide range of models (see our sections 3-4) and admit a straightforward generalization to *nonparametric noisy models* including *'colored noise'*, see [14] .

The parallel implementation of the STI outlined in our paper allows *fast processing of high-dimensional models* which are hard to even process with LP relaxation.

Another novelty of this paper is the theory of STI for factorial models under random designs which further validates the ideas of the RBM's authors and establishes the factorial models as linear models in even higher dimensions. This theoretical development is complemented by our novel intensive parallel implementation of the STI simulation for factorial models.

The outline of the rest of the paper is as follows:

2. Introduction and elementary models

3. Factorial models

4. Matrix design

5. The MPI system

6. The general program execution

2. Introduction and the main models

To illustrate ideas of this paper we utilize the following example of the ALOHAnet (see Wikipedia) in order to tie the ideas together. Suppose that one has a large collection of t enumerated satellites (say, t=1000) which are used to receive and transmit packets of information. Now suppose there is small subset $\Lambda = (\lambda_1, \ldots, \lambda_s)$ of satellites which have packets ready for transmission. These satellites begin to be *active* during the specially arranged announcement time-window, when they synchronously indicate their readiness, while the rest $t - s$ satellites are *silent*. The λ-th active satellite uses *binary string* $x^N(\lambda) := x_1^N(\lambda)$ to announce its readiness. If transmissions of these strings do not interfere, all $x^N(\lambda)$ must simply be distinct for the receiver to identify all active inputs. Since all announcements are sent synchronously, there exists an *interference* between them which is modeled as follows: the receiver gets a string $y^N := y_1^N$ which components y_i are a known function $g(x_i(\lambda))$ for every i-th time-point. These transmitted mixtures can be also corrupted by noise to z^N. The concatenation of mixing and noise is the Multi-Access Channel (MAC), see [2]. This scheme is illustrated in the following Fig. 1.

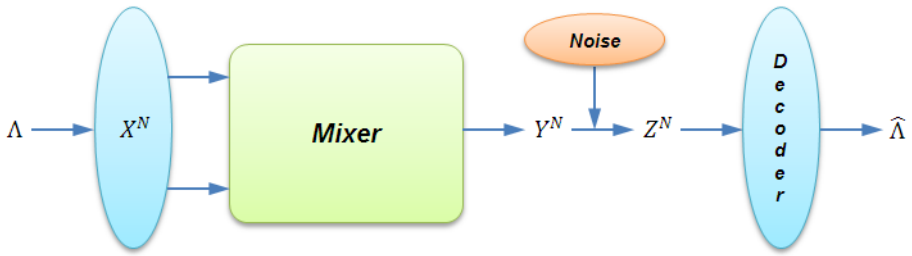


Figure 1. Multi-Access Channel.

The receiver of (noisy) output sequence $z^N := z_1^N$ knows the encoding matrix (design)

$$X^N = (x_i(\lambda),\ i = 1, \ldots, N;\ \lambda = 1, \ldots, t),$$

the mixing function $g(\cdot)$, and makes decision $\hat{\Lambda}$ about unknown subset Λ using certain analysis method. Decision $\hat{\Lambda}$ is wrong, if $\hat{\Lambda} \neq \Lambda$. The probability of this event over the Cartesian product of the random design X, the uniform Λ-distribution over all possible allocations of Λ and noise is called the Mean Error Probability (MEP).

The simplest models of mixing that we selected for simulation are *Boolean Sum, Additive, Linear and Second Order Factorial* Regression Models. Thus to summarize, the general formula is the following:

$$y_i = g(x_i\Lambda),\ i = 1, \ldots, N$$

	x_1	x_2	x_3	x_4	...	x_{t-1}	x_t	y
1	±1	±1	±1	±1	...	±1	±1	$g(x_{1,s_1}, x_{1,s_2}, x_{1,s_2}, x_{1,s_4})$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
N	±1	±1	±1	±1	...	±1	±1	$g(x_{N,s_1}, x_{N,s_2}, x_{N,s_3}, x_{N,s_4})$

Below are three simplest widely used mixing functions:

1. ***Linear:*** Linear (in inputs and parameters) model:

$$y_i = \sum_{j=1}^{s} b_{s_j} x_i(s_j),$$

with binary carriers $x_i(s_j) = \pm1$ and $i = 1, \ldots, N$.

2. ***Additive (special case of linear model):*** Additive (in inputs, all parameters are equal to 1) model:

$$y_i = \sum_{j=1}^{s} 1 \cdot x_i(s_j),$$

with binary carriers $x_i(s_j) \in \{0, 1\}$ and $i = 1, \ldots, N$.

3. ***Boolean sum:*** Boolean sum model with binary carriers $x_i(s_j) \in \{0, 1\}$ with $P(0) = 2^{1/s}, P(1) = 1 - 2^{1/s}, i = 1, \ldots, N$ and $P(y = 0) = 1/2$:

$$y_i = \bigvee_{j=1}^{s} x_i(s_j),$$

where $\bigvee$ denotes the Boolean sum of binary variables.

In our simulation we assume that $|s| \leq 4$. Recovering the set λ of all active inputs (AIs) using the minimal number of N observations was studied theoretically in [17, 19], and by simulation in [14, 16, 17] determining s for the *Boolean sum*, *Additive* and *Linear* models (see the next section) using the ESI-minimization described below.

Earlier results [17, 19] suggest that the length of the sequence of active s inputs for t total inputs approaches $C(s) \cdot log_2(t)$, where $C(s)$ is a mixing model and noise. The idea is that any of the $silent$ inputs have smaller ESI-relation with the output y than the $active$ ones. The formula for the ESI is

$$ESI(x, y) = \sum_{y} \sum_{x} \tau(x, y) log \frac{\tau(x, y)}{\tau(x)\tau(y)}$$

where $\tau(x, y)$ is the joint frequency of each column's value with the values of column y which is calculated as provided above. For the initial phase of the implementation it will be performed for only one input to clarify the results.

3. Factorial models

We first give here a schematic outline of the celebrated *Fisher's combined ideas of randomization and Complete Factorial Experimentation (CPE)* and the so-called Response Surface Methodology (RSM) developed in [3] and continued by the authors of the RBM.

The RSM models problems in which a response of interest is influenced by several variables and the objective is to optimize this response, was created during R. Fisher's work in the Rothamsted agricultural center near London. For example, the growth of a plant is affected by a certain amount of water x_1 and fertilizer x_2. The plant can grow under certain range of treatments x_1 and x_2 combinations. The RSM is used for improving and optimizing the response variable. In this case, the plant growth y is the response variable, it is a function of water and fertilizer as expressed by $y = f(x_1, x_2) + e$, where e is random experimental error which is assumed IID (Independent and Identically Distributed) in the sequence of experiments. In order to develop a proper approximation for $f(\cdot)$, the experimenter usually starts with a linear function of independent variables in some operability region $\mathcal{O}$ (See figure 2)

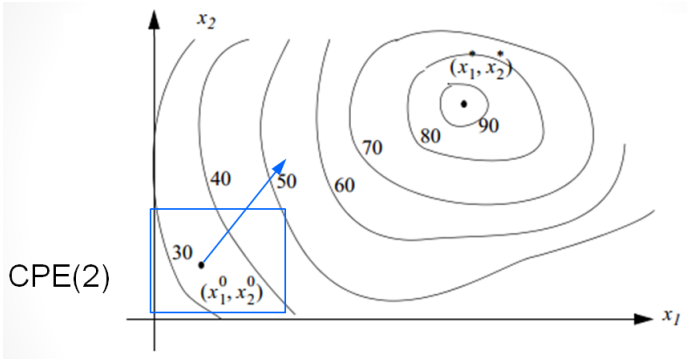


Figure 2. How do we get from starting position (x_1^0, x_2^0) in the direction of the gradient.

If the response can be *adequately* described by a linear function of independent variables, then we can find the direction of the fastest $f(\cdot)$ growth and move the operability region in this direction. If a significant curvature in the response surface is established, then a higher degree polynomial should be used. The next to apply is a mixed first and second-order model

$$y_i = m_0 + \sum_{1}^{d} \theta(\beta) x_i(\beta) + \sum_{\beta=1}^{d} \sum_{\alpha=1}^{\beta-1} \theta_{\alpha,\beta} x_i(\alpha) x(\beta) + e_i,\ i = 1, \ldots, N, \tag{1}$$

which can approximate a maximum of $f(\cdot)$, at least if it is in $\mathcal{O}$. The CPE theory specifies $\mathcal{O}$ in case of d independent variables as a cube of dimension d and proves some optimality of measurements in its vertices. This cube can be rescaled as a unit cube. If saddle type surfaces cannot happen, then G.E. P. Box argued that the most economical way to detect a curvature is to place some control measurements in the center of the cube. If saddle points cannot be excluded, then measurements in additional locations should be added, see [24]. In what follows, we omit intercept m_0 from consideration, since a single measurement at the center suffices to estimate it. The full second order model requires measurements in some additional points, see [3].

Among CPE(d) advantages found by R. Fisher, is the joint IID property of LS-estimates for all main effects and interactions, and their variance minimality as compared to all other designs of the same number of experiments. A major problem with the CPE(d) for large d is an exponential growth of experiments with $d \to \infty$. Fisher, Yates and their followers proposed an algebraic theory of fractional CPE see [4], which

reduced the support of the design several times at the expense of lack of estimability of substantial part of coefficients.

3.1. RSM under sparsity assumption

If dimension d is *too large and additional measurements are too expensive*, the RBM authors essentially proposed in a vague form a new paradigm: estimating the linear model (and in case of its inadequacy, the **factorial second order model** consisting of pairwise interactions) **under assumption of sparsity** of coefficients in both linear and factorial models, and a **random sample of measurements among the CPE vertices**. Their many applications showed fruitfulness of this approach! Among the origins of the sparsity hypothesis is the 'Occam razor' or 'bottleneck' principle stating that many phenomena have few significant factors ruling it. Of course, this principle is not applicable everywhere! For example, certain traits in genetics are ruled by few genes, but others, like height, depend on enormous number of genes.

Let us now study the consequences of these two assumptions theoretically. We show the asymptotic validity of the Fisher's CPE optimality properties under these two assumptions for the mixed second order factorial model with $d \to \infty$ variables and total number $t = d_{1,2} := d + 2\binom{d}{2}$ *ordered* coefficients. We process STI easily and rapidly for d of around 1000 in parallel, while its popular rival—Linear programming relaxation has major problems in processing such a big data. The generalization to factorial model of larger order is straightforward. Our focus will be on models with small noise under continuous prior distribution of s active (non-null) coefficients of the model. The latter assumption implies **incommensurability of active coefficients** (validity of *condition U* of [14]) with probability 1. Let us denote by μ the minimal distance between 2^s combinations $\{\sum \pm b(a)\} := \mathcal{V}$ of active coefficients.

Theorem. *1) Consider measurements (1) of the mixed factorial model of order 2 with constant variance σ^2 over the uniform random sample of vertices of a unit cube $\mathcal{O}$ of size N. 2) Assume that only fixed number s of incommensurable coefficients of the model (1) are non-null (sparsity of active coefficients) and $\sigma = k\mu$. 3) Let the following asymptotic relations hold: $NC_\sigma(s)/\log d_{1,2} \geq (1+\varepsilon), k \to 0$ as $d \to \infty$ for some $\varepsilon > 0$, here $C_\sigma(s) \to 1$ as $\sigma \to 0$, is the capacity of the additive channel (1) converting equally likely 2^s linear combinations of AIs into the noisy measurements z belonging to asymptotically disjoint output clouds around elements of $\mathcal{V}$.*

Then all active coefficients of the model admit jointly asymptotically unbiased estimate with probability approaching 1 as $d \to \infty$ and covariance matrix of these estimates converges to $(1/N)\mathcal{I}$, where $\mathcal{I}$ is the identity matrix.

The estimation goes in two steps. In step 1, we determine the set of all active variables with probability converging to 1. Step 2 is the LS estimate in the reduced model consisting only of s active inputs.

Remarks 1. *The capacity under arbitrary set of incommensurable active coefficients is the same.*

2. In applications, the assumption (2) is replaced with an empirical one - that only s coefficients and all their linear combinations with coefficients ± 1 significantly surpass the standard deviation of the additive noise.

Lemma. *All $x(\alpha)$ and all interactions of any order are mutually independent and uniformly distributed in vertices of $\mathcal{O}$.*

For notational simplicity, let us confine the **proof** of the lemma to typical interactions $x(1)x(\beta), \beta = 2, 3$. Their joint Moment Generating Function $MGF(u_2, u_3) = E[\exp \sum_{\beta=2}^{3} u_\beta x(1)x(\beta)]$, due to the joint independence of all $x(\beta)$. Now for $\beta = 2, 3$, we have

$$E_{x(1)}[\exp\,[x(1)\sum u_\beta x(\beta)]] = E_{x(1)}[ch(v_2)ch(v_3)] = ch(u_2)ch(u_3),$$

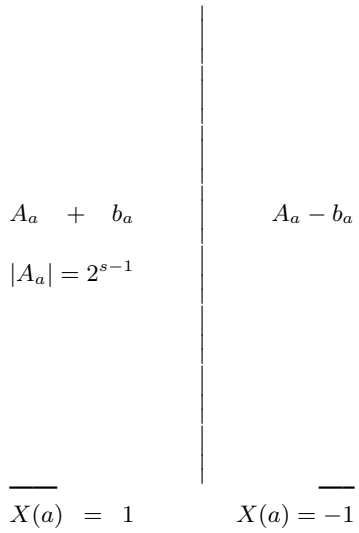


Figure 3. Outline of limiting scatter diagrams: a significant variable $X(a)$, no noise.

where $v_\beta = x(1)u(\beta)$.

Thus, random variables $x(1)x(\beta), \beta = 2, 3$ are IID—their joint MGF is the product of identical functions. Extending the proof to the general case is straightforward.

Proof of the theorem. The lemma states that the factorial model is linear model of dimension $d_{1,2}$ with jointly independent carriers ± 1.

We first consider the STI-detection of AIs **without noise** under condition U on significant coefficients. We examine all pairs $(x^N(a), z^N)$, where $x^N(a)$ is the binary input column and z^N is the output column with components taking 2^s values.
Let us assume for definiteness WLOG that variables $x(1), \ldots, x(s)$ are AI.
Fixing the value of one of them (say, the first) we still have 2^{s-1} equally likely combinations of the rest with coefficients ± 1.

In the left (right) hand side of the scatter diagrams corresponding to $x(a) = \pm 1$ we have non-overlapping sets of outputs $y - b_a$ $(y + b_a)$, respectively, with coefficients $b_a,\ a = 1, \ldots, s;\ y$, where $y \in A_a$, A_a is the set of linear combinations of significant variables different from $X(a)$. The cardinality $|A_a|$ is 2^{s-1}. Hence for each significant variable $X(a)$ we have a separate partition of the output set $\mathcal{Z}$ into two subsets $\{\pm b_a + A_a\}$ displayed on the scatter diagrams.

It is clear that $\mathbf{I}(x_1 \wedge Y) = \mathbf{H}(y) - \mathbf{H}(Y|x_1) = 1$. Thus $C^{STI} = C^{BF} > C^{LP}$. The last inequality follows from the results of simulation in Fig 12, which also shows that the LP-capacity shrinks with growing s in contrast to that of STI. The first equality follows also from elementary combinatorial argument: A particular inactive input can be regarded as active by STI, if 2^s output values of its **both** partial scatterplots shrink to 2^{s-1} for random design with $N > (1 + \epsilon) \log t$ rows, $\epsilon > 0$. These events are independent and have probability 1/2 for every row. Thus it happens for at least one of inactive inputs with probability $\leq (d_{1,2} - s)2^{-N} \leq d_{1,2}^{-\epsilon}$. Since $\epsilon > 0$ is arbitrary, the STI capacity for random design ≥ 1. If we assume a continuous prior distribution for the active (non-null) coefficients of this extended model, then all these active coefficients are incommensurable (i.e. condition U of [14] is valid) with probability 1, and thus all arguments of [14], section 4 for the noiseless linear model remain valid. Informal explanation of proof: Every measurement gives almost s bits of information on the allocation of active coefficients. Since the entropy of the uniformly distributed uncertainty about the subset of active inputs is $\log \Pi_{j=0}^{s-1} d_{1,2}^{s-j} \asymp s \log t$, we get an asymptotic expression $N \asymp \log d_{1,2}$. We get asymptotically one bit of information from each

measurement on whether a fixed input is active, but uncertainty is only $\log d_{1,2}$ asymptotically as $d \to \infty$. Hence the result.

Its generalization to noisy models is straightforward due to the asymptotic disjointness of clouds around elements of $\mathcal{V}$. As a result, the STI has asymptotically the same capacity as the brute force analysis meaning that

$$\liminf NC(s)/\log d_{1,2} \leq 1,$$

where $d_{1,2}$ is the dimension of the mixed model and N is the number of measurements providing the mean error probability less than arbitrary $\gamma > 0$ which slowly converges to 0 as $d \to \infty$. Informally, $C(s)$ is the entropy of the same distribution of each noisy cloud around elements of $\mathcal{V}$.

We expect the simulation results for the second order factorial model can show somewhat less capacity as compared to the linear one because the standard 'random number generators' we use are not perfect and give worse performance for larger simulation size.

3.2. Simulation for 2^{nd} Order Factorial model

2^{nd} ***Order Factorial:*** In the Second Order Factorial (in inputs and parameters) model:

$$y_i = \sum_{\beta,\alpha \in S} \theta_{\beta,\alpha} x_i(\beta) x_i(\alpha),$$

with binary carriers $x_i(s_j) \in \{0,1\}$ and $i = 1,\ldots,N$ and only those coefficients $S = \{1,...,s\}$. The coefficients of this model are $\theta_{1,2} = 1$ and $\theta_{3,4} = \sqrt{2}$ and 0 otherwise. Thus the implemented equation for $s = 2$ would be:

$$y_i = 1 \cdot x_i(s_1) \cdot x_i(s_2) + \sqrt{2} \cdot x_i(s_3) \cdot x_i(s_4)$$

with binary carriers $x_i(s_j) \in \{0,1\}$ and $i = 1,\ldots,N$.

The coefficients of this model are $\theta_{1,2} = 1$, $\theta_{2,3} = \sqrt{2}$ and 0 otherwise. Thus the implemented equation for $s = 2$ would be:

$$y_i = 1 \cdot x_i(s_1) \cdot x_i(s_2) + \sqrt{2} \cdot x_i(s_2) \cdot x_i(s_3)$$

4. The Matrix Design

The implementation we chose is via the use of matrices, which generally follow the same design for all the models. Below is the general matrix design for the *Linear Model*, the *Additive Model* and the 2^{nd} *order factorial model* as follows:

$$\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ N \end{array} \begin{array}{c} \begin{array}{cccc|ccc} 1 & 2 & \cdots & s & s+1 & \cdots & t \end{array} \\ \left(\begin{array}{cccc|ccc} \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} \\ \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \{-1,1\} \\ \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} & \{-1,1\} & \ldots & \{-1,1\} \end{array}\right) \end{array} \begin{array}{l} \\ g_{model}([1,1\ldots s]) \mapsto \\ g_{model}([2,1\ldots s]) \mapsto \\ \vdots \\ g_{model}([N,1\ldots s]) \mapsto \end{array} \begin{array}{c} Y \\ \left(\begin{array}{c} y_1 \\ y_2 \\ \vdots \\ y_N \end{array}\right) \end{array}$$

For the *Boolean Sum Model* the matrix is the following:

$$\begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ N \end{array}\begin{array}{c} \begin{array}{cccc|ccc} 1 & 2 & \cdots & s & s+1 & \cdots & t \end{array} \\ \begin{pmatrix} \{0,1\} & \{0,1\} & \dots & \{0,1\} & \{0,1\} & \dots & \{0,1\} \\ \{0,1\} & \{0,1\} & \dots & \{0,1\} & \{0,1\} & \dots & \{0,1\} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \{0,1\} \\ \{0,1\} & \{0,1\} & \dots & \{0,1\} & \{0,1\} & \dots & \{0,1\} \end{pmatrix} \end{array} \begin{array}{l} [1,1] \vee [1,2] \vee \cdots \vee [1,s] \mapsto \\ [1,1] \vee [1,2] \vee \cdots \vee [1,s] \mapsto \\ \vdots \\ [1,1] \vee [1,2] \vee \cdots \vee [1,s] \mapsto \end{array} \begin{array}{c} Y \\ \begin{pmatrix} \{0,1\} \\ \{0,1\} \\ \vdots \\ \{0,1\} \end{pmatrix} \end{array}$$

4.1. Linear Programming (LP) Relaxation, Hybrid LP-BF

Let us outline the use of Linear Programming (LP) relaxations for both linear and nonlinear (boolean sum) models. For the linear version of the problem we use the popular ℓ_1-norm relaxation of sparsity. Here the problem in Fig.1 can be rewritten as

$$\tilde{\Lambda} = \min |\Lambda|, \text{ such that } y_i = \sum_{a \in \Lambda} x_i(a), \ \forall i. \tag{2}$$

Here $|\Lambda|$ is the number of elements in Λ. Let us define the indicator function I_Λ such that $I_\Lambda(a) = 1$ iff $a \in \Lambda$, and notice that the ℓ_1-norm of I_Λ, i.e. on $\sum_a I_\Lambda(a)$. Note that the range of I_Λ is $D = \{0, 1\}$, so it is always nonnegative, and instead of $\sum_a |I_\Lambda(a)|$ we can use $\sum_a I_\Lambda(a)$. We introduce the relaxation by replacing the minimization over D with that over the continuous range [0,1]

$$\tilde{\Lambda} = \min \sum_a I_\Lambda(a), \text{ such that } y_i = \sum_{a \in \Lambda} x_i(a), \ \forall i. \tag{3}$$

This type of relaxation has received considerable amount of attention in many fields, including statistics (Lasso regression, [33]), and in signal processing (basis pursuit, [8,9], [13]). While the much simpler linear programming relaxation is not guaranteed to solve the original combinatorial problem, both theoretical and simulated study over the the random design showed that if the unknown sparse signal is 'sparse enough', then the linear programming relaxation exactly recovers the unknown signal ([10] et al) with high probability summarized in the following sentence from [10]: 'for many (design) matrices there is a threshold phenomenon: if the sparsest solution is sufficiently sparse, it can be found by linear programming'.

For the non-linear problem we are forced to relax not only the sparsity of the indicator vector I_A, but also the measurement model.

Since $y_i = \cup_{a \in A} x_i(a)$, then it must hold that $y_i \leq \sum_{a \in A} x_i(a)$. Hence, our first relaxation is

$$\min \sum_a I_A(a) \text{ such that } y_i \leq \sum_{a \in A} x_i(a), \quad 0 \leq I_A(a) \leq 1. \tag{4}$$

We also note that if $y_i = 0$, then it must hold that all $x_i(a) = 0$ and the inequality $y_i \leq \sum_{a \in A} x_i(a)$ holds with equality. Hence, a stronger relaxation is obtained by enforcing this equality constraint.

$$\min \sum_a I_A(a) \text{ such that } 0 \leq I_A(a) \leq 1, \text{ and } y_i \leq \sum_{a \in A} x_i(a), \text{ if } y_i \neq 0 \text{ and} \tag{5}$$

$$y_i = \sum_{a \in A} x_i(a) \text{ if } y_i = 0. \tag{6}$$

Thus taking into account particular features of this nonlinear model **before applying linear programming** is essential. To our knowledge, bounds for the performance of this linear programming relaxation of the nonlinear screening problem have not been studied in the literature.

Interesting relations between combinatorial properties of design matrices and correctness of the LP solution are in [15].

5. The Parallel Implementation

The parallel implementation was performed via the MPI (Message Passing Interface). The main idea was to keep only one random vector at a time in memory across multiple compute-nodes, while collecting the necessary information for the ESI-calculation before randomizing it again. This design allowed for testing convergence for larger values of t of the different models. Below is a representation of this vector as an array across the compute-nodes denoted as CPUs:

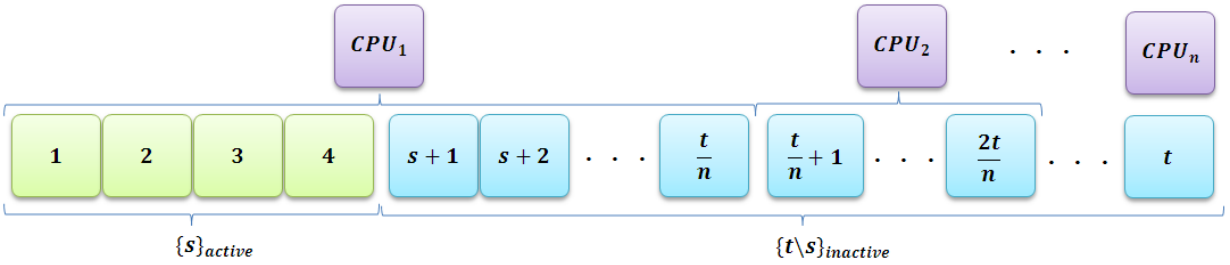


Figure 4. Illustrated is the array structure that is split across multiple compute nodes (CPUs) as a parallel MPI-enabled program.

As each array element will generate eventually an ESI-value, these then can be compared with each other. In an MPI-system the ability to broadcast the values among compute-nodes enables one to perform the convergence of N in parallel. For example, if the maximum value of all the silent elements from each compute-node is transmitted to the first node, then one can compare each node-specific maximum value with the minimum value of ESI from the active ones. If 95% of the time the active ones have a higher value than all the inactive ones then we say that convergence has been reached for that specific combination of s and t of the particular model, otherwise N is increased by one and the test for convergence is performed again. The value for $\hat{\Lambda}$ is calculated on the first node and then broadcast to all the others as it is needed for the element-wise ESI calculated on each compute-node.

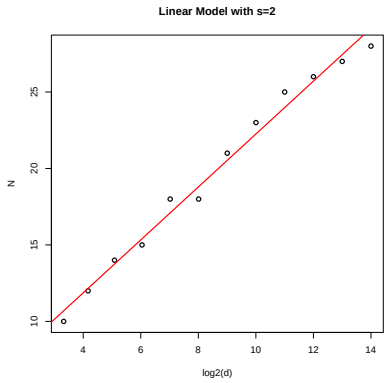
5.1. The Linear Model

Below are the converged values N for $s = \{2, 3, 4\}$ and d, for powers of two, which should be increased by s since the program was run with s as a separate matrix:

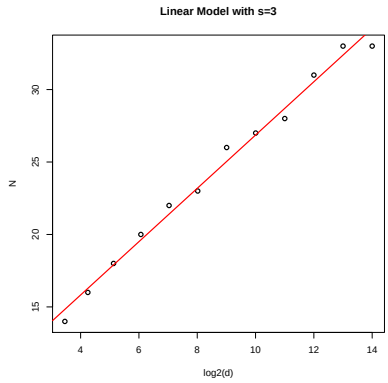
s \ d	$8+s$	$16+s$	$32+s$	$64+s$	$128+s$	$256+s$	$512+s$	$1024+s$	$2048+s$	$4096+s$	$8192+s$	$16384+s$
2	10	12	14	15	18	18	21	23	25	26	27	28
3	14	16	18	20	22	23	26	27	28	31	33	33
4	20	23	26	27	29	30	32	33	36	38	39	39

Table 1. The converged values of N for the linear model.

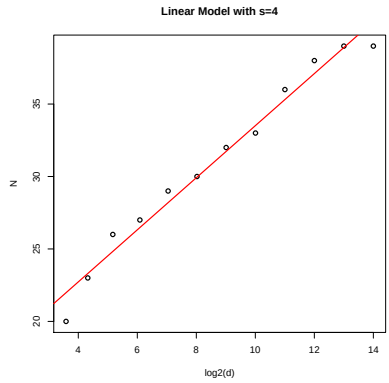
Below are plots of N versus $log_2(t)$ for the linear regression model :



(a) For $s = 2$, the linear regression is $N = 1.73 \cdot log_2(d) + 4.95$.



(b) For $s = 3$, the linear regression is $N = 1.84 \cdot log_2(t) + 8.47$



(c) For $s = 4$, the linear regression is $N = 1.80 \cdot log_2(t) + 15.54$

Figure 5. Linear regressions of of N versus $log_2(d)$ for $s = 2, 3, 4$ for the linear model using STI.

Below is the table to summarize the coefficients of $log_2(d)$ in relation to s:

s	Coefficient of $log_2(d)$
2	1.73
3	1.84
4	1.80

Table 2. The coefficients of $log_2 d$ with respect to s for the linear model using STI.

The linear regression of the above relation is the following with a plot below illustrating it:

$$\text{Coefficient of } log_2(d) = 0.033 \cdot s + 1.69.$$

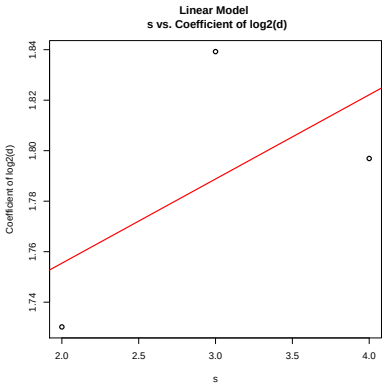


Figure 6. Linear regression of the coefficient of $log_2(d)$ versus s for the linear model using STI.

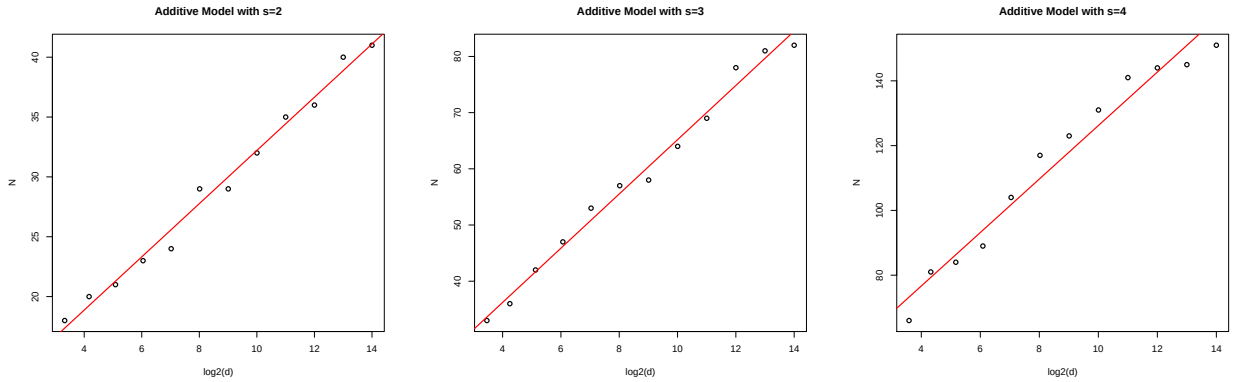
5.2. The Additive Model

Below are the converged values N for $s = \{2, 3, 4\}$ and d, for powers of two:

s \ d	$8+s$	$16+s$	$32+s$	$64+s$	$128+s$	$256+s$	$512+s$	$1024+s$	$2048+s$	$4096+s$	$8192+s$	$16384+s$
2	18	20	21	23	24	29	29	32	35	36	40	41
3	33	36	42	47	53	57	58	64	69	78	81	82
4	66	81	84	89	104	117	123	131	141	144	145	151

Table 3. The converged values of N for the additive model using STI.

Below are plots of N versus $log_2(t)$ for the additive model:



(a) For $s = 2$, the linear regression is $N = 2.22 \cdot log_2(t) + 9.98$

(b) For $s = 3$, the linear regression is $N = 4.83 \cdot log_2(t) + 16.91$

(c) For $s = 4$, the linear regression is $N = 8.25 \cdot log_2(t) + 43.64$

Figure 7. Linear regressions of N versus $log_2(d)$ for $s = 2, 3, 4$.

Below is the table to summarize the coefficients of $log_2(d)$ in relation to s:

s	Coefficient of $log_2(d)$
2	2.22
3	4.83
4	8.25

Table 4. The coefficients of $log_2 d$ with respect to s for the additive model using STI.

The linear regression of the above relation has the coefficients $3.02 \cdot s - 3.95$ of $log_2(d)$

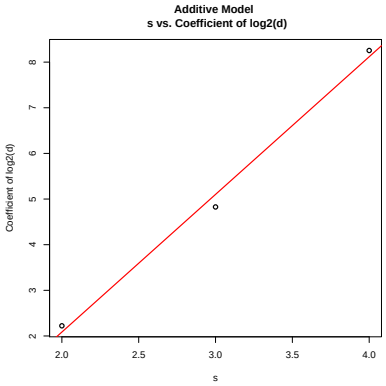


Figure 8. The plot of the coefficient of $log_2(d)$ versus s the additive model using STI.

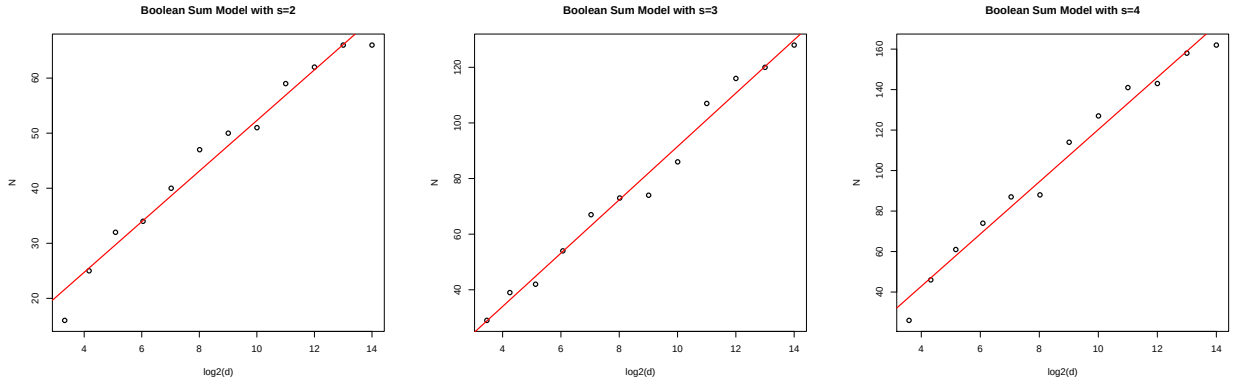
5.3. The Boolean Sum Model

Below are the converged values N for $s = \{2, 3, 4\}$ and d, for powers of two:

s \ d	$8+s$	$16+s$	$32+s$	$64+s$	$128+s$	$256+s$	$512+s$	$1024+s$	$2048+s$	$4096+s$	$8192+s$	$16384+s$
2	16	25	32	34	40	47	50	51	59	62	66	66
3	29	39	42	54	67	73	74	86	107	116	120	128
4	26	46	61	74	87	88	114	127	141	143	158	162

Table 5. The converged values of N for the boolean sum model using STI.

Below are plots and linear regression models of N versus $log_2(d)$:



(a) For $s = 2$, the linear regression is $N = 4.59 \cdot log_2(t) + 6.35$

(b) For $s = 3$, the linear regression is $N = 9.59 \cdot log_2(t) - 4.36$

(c) For $s = 4$, the linear regression is $N = 12.90 \cdot log_2(t) - 8.75$

Figure 9. Linear regressions of N versus $log_2(t)$ for $s = 2, 3, 4$ for the boolean sum model using STI.

Below is a table to summarize the coefficients of $log_2(t)$ in relation to s:

s	Coefficient of $log_2(d)$
2	4.59
3	9.59
4	12.90

Table 6. The coefficients of $log_2 d$ with respect to s for the boolean sum model using STI.

The linear regression of the above relation is the following with a plot below illustrating it:

$$\text{Coefficient of } log_2(d) = 4.15 \cdot s - 3.43.$$

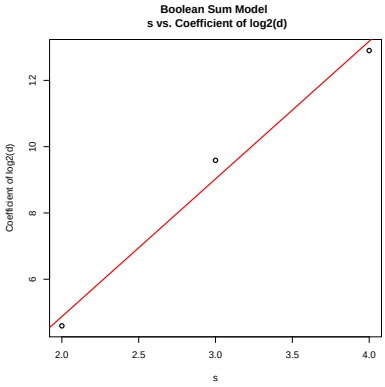


Figure 10. Illustrated is the plot of the coefficient of $log_2(d)$ versus s for the boolean sum model using STI.

5.4. The 2^{nd} Order Factorial for $y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_1x_3$

Below are the converged values of N:

s \ d	8	16	32	64	128	256	512	1024
$nCr(t,2) \rightarrow$	28	120	496	2016	8128	32640	130816	523776
N for $s = 2$	17	21	30	34	35	43	46	52

Table 7. The converged values of N for the 2^{nd} order factorial model ($y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_1x_3$) using STI.

Below is the plot and linear regression model of N versus $log_2(d)$:

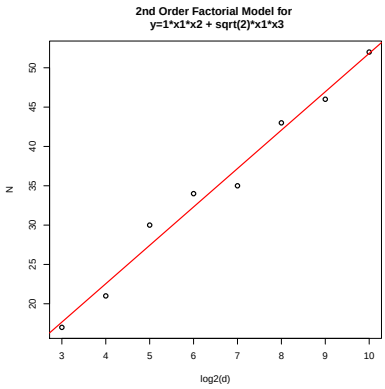


Figure 11. The plot of N versus $log_2(d)$ for $s = 2$ for the 2^{nd} order factorial model ($y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_1x_3$) using STI.

The linear regression function is $N = 2.44 \cdot log_2(d_2) + 3.02$.

5.5. The 2^{nd} Order Factorial for $y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_3x_4$

Below are the converged values of N:

s \ d	8	16	32	64	128	256	512	1024
$nCr(t,2) \rightarrow$	28	120	496	2016	8128	32640	130816	523776
N for $s = 2$	13	18	21	23	27	28	31	36

Table 8. The converged values of N for the 2^{nd} order factorial model ($y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_3x_4$).

Below is the plot and linear regression model of N versus $log_2(d)$:

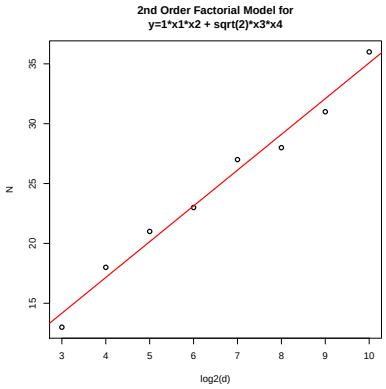


Figure 12. The plot of N versus $log_2(d)$ for $s = 2$ for the 2^{nd} Order Factorial for $y = 1 \cdot x_1x_2 + \sqrt{2} \cdot x_3x_4$.

The linear regression function is $N = 1.49 \cdot log_2(d_2) + 5.20$.

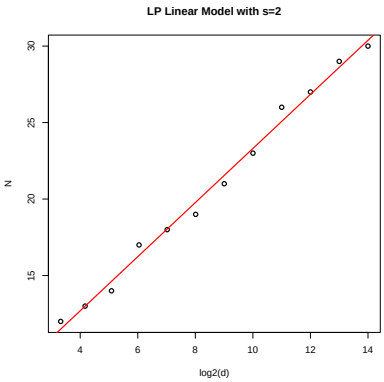
5.6. The LP Linear Model

Below are the converged values N for $s = \{2, 3, 4\}$ and d, for powers of two, which should be increased by s since the program was run with s as a separate matrix:

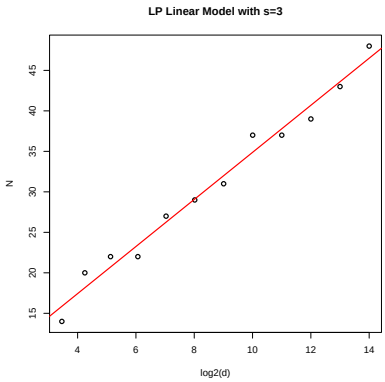
s \ d	$8+s$	$16+s$	$32+s$	$64+s$	$128+s$	$256+s$	$512+s$	$1024+s$	$2048+s$	$4096+s$	$8192+s$	$16384+s$
2	12	13	14	17	18	19	21	23	26	27	29	30
3	14	20	22	22	27	29	31	37	37	39	43	48
4	16	23	25	30	37	44	49	55	56	56	68	69

Table 9. The converged values of N for the LP linear model.

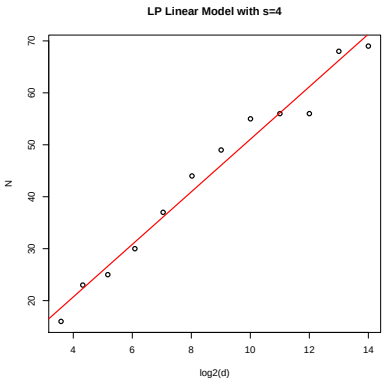
Below are plots of N versus $log_2(t)$ for the LP linear regression model :



(a) For $s = 2$, the linear regression is $N = 1.77 \cdot log_2(d) + 5.62$



(b) For $s = 3$, the linear regression is $N = 2.91 \cdot log_2(t) + 5.81$



(c) For $s = 4$, the linear regression is $N = 5.06 \cdot log_2(t) + 0.47$

Figure 13. Linear regressions of of N versus $log_2(d)$ for $s = 2, 3, 4$ for the linear model using LP.

Below is the table to summarize the coefficients of $log_2(d)$ in relation to s:

s	Coefficient of $log_2(d)$
2	1.77
3	2.91
4	5.06

Table 10. The coefficients of $log_2 d$ with respect to s for the LP linear model using STI.

The linear regression of the above relation is the following with a plot below illustrating it:

$$\text{Coefficient of } log_2(d) = 1.65 \cdot s - 1.69.$$

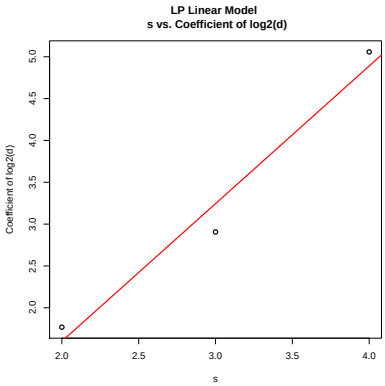
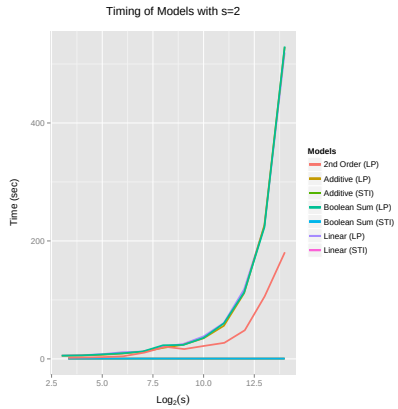
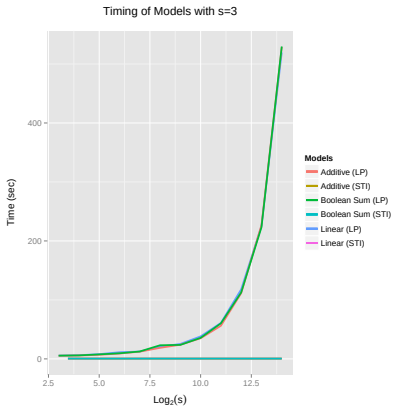


Figure 14. Linear regression of the coefficient of $log_2(d)$ versus s for the linear model using LP.

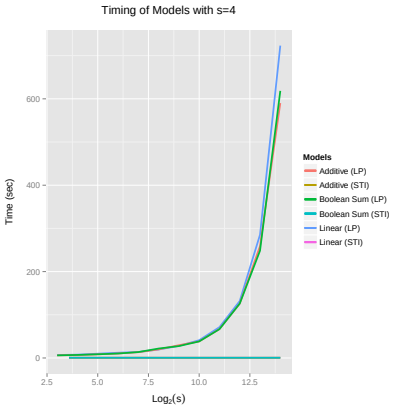
Below are the timing results of the STI versus LP approach.



(a) Timing of the STI and LP models for $s = 2$.



(b) Timing of the STI and LP models for $s = 3$.



(c) Timing of the STI and LP models for $s = 4$.

Figure 15. Plots of N versus $log_2(d)$ for $s = 2, 3, 4$ for STI and LP.

6. Discussion

By analyzing the minimal number N of measurements to obtain Mean Error Probability ≤ 0.05 for each of the models, we see that our simulation results are in a rather good agreement with our previous theoretical results on capacity of STI. As we expected, the simulated evaluation of the capacity for the second order factorial model shows more variability than for the linear model which was shown theoretically equivalent to it. Apparently, the reason is in a huge dimension of the factorial model which reveals the imperfection of the 'random number generator' used. For the linear, boolean sum and additive models, the number of inputs is $\binom{d}{1}$, while the 2^{nd} order factorial model is equi valent to the linear one of dimension $2\binom{d}{2}$ for the random design. In all models studied, results for the STI approximate the maximal capacity of the channel. If the channel capacity for some models empirically either exceeded or felt a little below the theoretical one, that is due apparently to the imperfections of the random number generators in constructing the matrices.

The MPI [36] approach was to utilize a broadcast/gather method in order to empirically determine minimal N with MEP ≤ 0.05.

We tested our results via linear programming (LP) and found that the results coincide with our expectations. Since STI can be implemented in parallel, we see that this implementation provides an overwhelming speedup as compared to the LP relaxation which is shown in Fig. 14.

6.1. Parallel Implementation of STI

The MPI System

In order to speed up the implementation we chose to parallelize it using MPI. MPI stands for Message Passing Interface and has the following structure [36]:

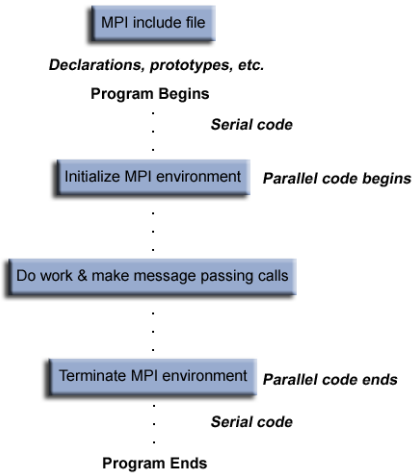


Figure 16. The structure of an MPI-enabled program as constructed so as to become parallel. Each program gets an ID (rank) by which they can be identified and communicate with each other.

For all the programs the **`mpi.h`** header file was required to be added to enable MPI functionality.

Under MPI, one performs the communication among programs through a communicator object, which usually is just one and set by the program. Below is a visualization of how each program parallel-instance - illustrated by their ID (rank) as a numbered circle - utilizes **`MPI_COMM_WORLD`** as a communication medium [36].

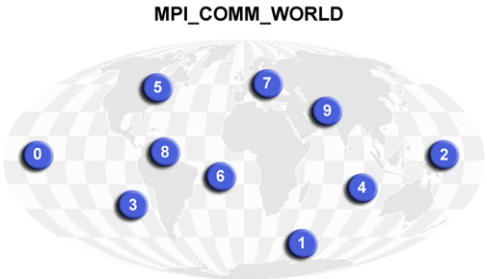


Figure 17. The communication object (**`MPI_COMM_WORLD`**) by which parallel-instances of a program communicate with each other.

MPI Broadcast

Using the **`MPI_Bcast()`**, below is a diagram that explains how node 0 - in this instance - broadcasts its value to all other nodes [37].

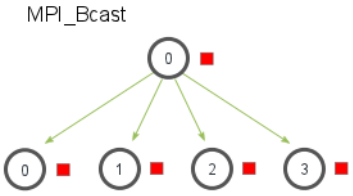


Figure 18. Illustrated is how node 0 broadcasts its value to all other nodes, which also have the same variable.

MPI Gather

Using the **`MPI_Gather()`**, below is a diagram that explains how all nodes send their values to node 0 - in this instance - where node 0 stores the values in an array [37].

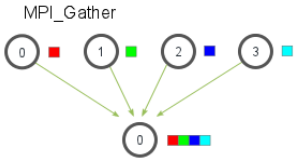


Figure 19. Illustrated is how all nodes send their value to node, which has an array variable.

MPI Barrier

Using the **`MPI_Barrier()`**, below is a diagram that explains how all nodes synchronize at points in the program where such a procedure is placed [38]:

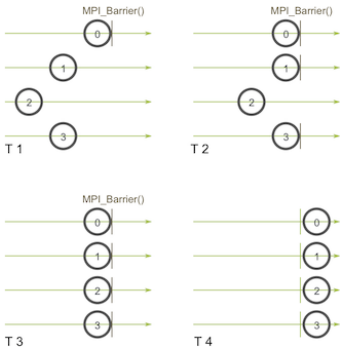


Figure 20. Illustrated is how all nodes synchronize at points of the parallel programs where the procedure **`MPI_Barrier()`** is placed.

Enabling the Run Environment

The first step is to add the library (file) dependencies on which our program is built upon. This is performed using the following commands:

```
module add gcc/4.8.2
module add slurm/2.6.6
module add mvapich2/gcc/64/2.0b
```

Node Allocation

Next the allocation of compute nodes, upon which the MPI program is run was performed using the **salloc** command. This command is a part of the open-source SLURM package [39], developed at the Lawrence Livermore National Laboratory. Below is the command that was issued for utilizing four nodes:

```
salloc -N 4 -p sched_neu_cooperman -time=4:00:00 -exclusive
```

Compiling the Code

Next the compilation of the code was performed using the custom **mpicc** wrapper script, which is called **c**. Below is the listed code:

```
#!/bin/bash
var=`echo $1 | sed s/..$//g`
mpicc -std=gnu99 $1 -o $var
```

Running the Compiled Code

The execution of the compiled programs was performed using the **srun** (SLURM command) with MPI features enabled [40], for which a custom script - called **m-with-nodes** - was created per model and is listed below[1]:

```
#!/bin/bash
mpirun -n ${1} ./$2 $3 $4 $5 $6 $7 $8 $9
```

Below is an example of running each of the programs:

```
./run_linear_model.pl 27 128 4 4
./run_additive_model.pl 117 512 4 4
./run_boolean_sum_model.pl 50 1024 2 4
./run_2nd_order_factorial_model.pl 43 512 1 2 1 3 4
./run_2nd_order_factorial_model.pl 28 512 1 2 3 4 4
```

Below is an example of a run:

[1] This program was corrected to run using `mpirun`, where previously it was using `srun`. Not all the results have been re-run because of time constraints, but through sampling the results are similar and thus the previous results have been provided.

```
./run_linear_model.pl 8 8 2 4

   t = 8
   s = 2

   For N = 8, 9 out of 100 rounds gave an error. N will be incremented by 1.
   For N = 9, 9 out of 100 rounds gave an error. N will be incremented by 1.
   For N = 10, 3 out of 100 rounds gave an error. N will be incremented by 1.

   CONVERGED! For s = 2 and t = 8, N = 10
```

The General Program Execution

Below is a diagram that describes the flow of the program. The **`model.c`** program is the one that executes one instance of the N, t, and s variables and reports back a 1 or -1. If the value is 1 then all the values of ESI for the s matrix are greater than then ones for the t matrix, otherwise that is considered an error. The Perl program keeps running the **`model.c`** for 100 times while counting the number of errors. If the error is greater than 5 then N is incremented, otherwise N is reported as convergence for an error of 5%. The **`model_declarations.c`** program provides the functions and variable declarations.

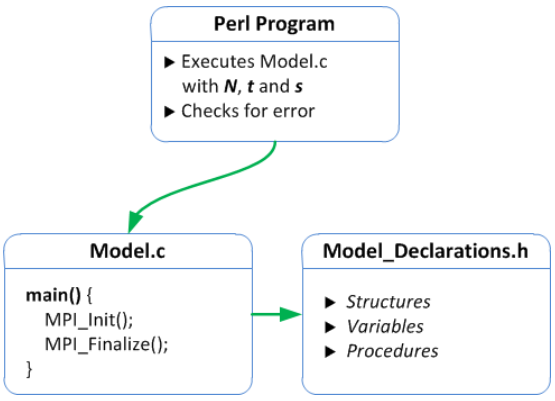


Figure 21. Illustrated is program flow. The Perl program executes the **`model.c`** program, while the **`model_declarations.h`** header file provides the functions and variable declarations.

For each program the execution is performed on a row-wise approach. Thus until N is reached a new s and t row is generated and the $\tau()$ (i.e. count) with the ESI calculations are performed. Afterwards the s and t row recreated in the next round. Therefore, instead of keeping a whole matrix in memory - this would allow for larger t and N values to be interrogated. Below is the general flow of the programs:

To launch the MPI programs, one just provides the program with an initial N, t and s parameters to the **`model.c`** program. The general idea of how it works is as follows:

Algorithm 1. *Implementation of MPI of STI*

INPUT: N, d, s

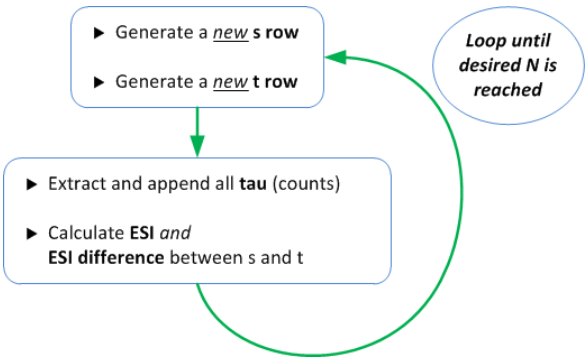


Figure 22. Illustrated is row-wise approach of the programs. The Perl program executes the **`model.c`** program, which in turn calls the **`performAnotherRound()`** procedure in the **`model_declarations.h`** header file to create and process a new *s* and *t* row.

OUTPUT: 1 if N converged, otherwise -1

procedure ($\forall$ nodes, where each node is assigned a unique id $\in \{0, \ldots, |nodes| - 1\}$)

    $row \leftarrow 0$

    while $row < N$ **do**

        $row \leftarrow row + 1$

        if nodeId == 0 **then**
            select randomly a node to generate the s vector
            generate s vector on random node
            broadcast s to all other nodes
        end if

        generate subvector of d based on the node number as an offset, where $|subvector| \approx \dfrac{d}{|nodes|}$
        generate y (output) based on s
        collect statistics of each vector element as compared to y

    end while

    calculate ESI for each column.
    calculate ESI difference $= [min(ESI) \in s] - [max(ESI) \notin s]$ *and* **broadcast** to node 0

    if nodeId == 0 **then**
        if $\exists$ ESI difference < 0 on any node **then**
            print -1
        else
            print 1
        end if
    end if

end procedure

If the Perl program notices that the output of -1 was more than 5 out of 100, then N is increased the **`model.c`** program is launched again with $N = N + 1$ until convergence.

7. Conclusions and future directions

The STI implementation in parallel can enable much faster computation, especially in comparison to much slower LP relaxation. We plan to demonstrate even faster STI parallel implementation on larger clusters of computers. A parallel simulated evaluation of the capacity and processing time for the full second order sparse multivariate model is presently under way for the maximin design studied in [24]. This development will be described in our next paper.

Acknowledgement Prof. I. Tsitovich generously aided us in the final technical preparation of this paper (and of our previous papers published in IP). National Science Foundation, NSF Award OCI 12-29059, "MRI Consortium: Acquisition of a heterogeneous, shared, computing instrument to enable science and computing research by the Mass. Green High performance Computing Consortium" enabled our parallel computations.

References

1. Adler Yu. P. V. V. Nalimov in my life. Paths we choose and destinies waiting for us (From Experimental Design to the quality management via Cybernetics), In *V. V. Nalimov—mathematician and philosopher, centennial, International conference, Moscow State University, 2010, ed. Drogalina J. A. and Panchenko L. A.*, Max press, 373–383, 2011.
2. Ahlswede R.: Multi-Way Communication Channels. In: *Proceedings of 2nd International Symposium on Information Theory, Tsahkadzor, 1971*. Budapest: Akademiai Kiado, 1973, 23–52.
3. Box, G. E. P. and Wilson, K.B. On the Experimental Attainment of Optimum Conditions (with discussion). *Journal of the Royal Statistical Society Series B*, **13**, No.1, 1951, 1–45.
4. Brodsky, S. *Introduction to Factorial Design of Experiment (mathematical foundations)*. N.Y.:Manhattan Academia, 2013.
5. Budne T.A. Application of Random Balance Designs. *Technometrics*, **1**, No 2, 1959, 139–155.
6. Candes E. J. Compressive Sampling. In: *International Congress of Mathematicians, Madrid, Spain, August 2006*. Zürich: Eur. Math. Soc., **3**, 2006, 1433–1452.
7. Csiszar I., Körner J. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Budapest: Academic Press and Akadémiai Kiadó, 1981.
8. Chen S. S. Donoho D. L., Saunders M. A. Atomic De–composition by Basis Pursuit. *SIAM J. on Scientific Computing*, **20**, 1998, 33–61.
9. Donoho D.L., Elad, M. Maximal Sparsity Representation via L_1 Minimization. *Proc. Nat. Acad. Sci.* **100**, 2003, 2197–2202.
10. Donoho D. L., Tanner J. Sparse Nonnegative Solution of Underdetermined Linear Equations by Linear Programming. *Proc. Nat. Acad. Sci.*, **102**, no. 27, 2005.
11. Dorfman R. The Detection of Defective Members of Large Populations. *Ann. Math. Statist.* **14**, No. 4, 1943, 436–440.
12. Erlich Y., Gordon A., Brand M., Hannon G. J. and Mitra P. P. Compressed Genotyping. *IEEE Trans. on Inform. Th.*, **56:2**, 2010, 706–723.

13. Malioutov D. M., Cetin M. and Willsky A. S. Optimal Sparse Representations in General Overcomplete Bases. *IEEE International Conference on Acoustics, Speech and Signal Processing, May 2004, Montreal, Canada*, **2**, II, 2004, 793–796.

14. Malyutov M.B. Search for active inputs of a sparse system: a review, *Springer Lecture Notes in Computer Science*, **7777**, 2012, 478–507.

15. Malioutov D.M. and Malyutov M. B. Boolean Compressed Sensing: LP Relaxation for Group Testing. *IEEE International Conference on Acoustics, Speech and Signal Processing, March 2012, Kyoto, Japan*, 2012.

16. Malyutov, M. B., Sadaka, H.: Capacity of screening under Linear Programming analysis. *Proceedings of the 6th International Workshop on Simulation, St Petersburg, Russia*, 2009, 1041–1045.

17. Malyutov, M. B. and Sadaka H. Jaynes Principle in Testing Active Variables of Linear Model. *Random Operators and Stochastic Equations*, **6**, 1998, 311–330.

18. Malyutov M. B. and Dyachkov A. G. On Weakly Separating Designs. In: *Methods of Transmission and Processing Information*, M.: Nauka, , 1980, 87–104 (In Russian).

19. Malyutov M. B. and Mateev P. S. Screening Designs for Non-Symmetric Response Function. *Mat. Zametki*, **27**, 1980, l09–127.

20. Malyutov M. B. On the maximal rate of screening designs. *Theory Probab. and Appl.*, **XXIV**, 1979, 655–657.

21. Malyutov M. B. Separating Property of Random Matrices. *Mat. Zametki*, **23**, 1978, 155–167.

22. Malyutov M. B. Mathematical Models and Results in Theory of Screening Experiments. In: *Theoretical Problems of Experimental Design, ed. M.B. Malyutov* , M.: Soviet Radio, 1977, 5–69, (In Russian).

23. Malyutov M. B. On Planning of Screening Experiments. In: *Proceedings of 1975 IEEE-USSR Workshop on Inform. Theory*, N.Y. , IEEE Inc., 1976, 144–147.

24. Malyutov M. B. Maximin Designs for Testing Degree of a Polynomial, *Design of Optimal Experiments, ed. by Malyutov M. B.*, Moscow University Press, 1975 (In Russian).

25. Malyutov M. B. and Pinsker M. S. Note on the Simplest Model of the Random Balance Method. In: *Probabilistic Methods of Research, ed. A. N. Kolmogorov*, Moscow University Press, 1972 (In Russian).

26. Meshalkin L. D. Justification of the Random Balance Method. *Industrial Lab.*, 1970, 316–318.

27. Moore D., McCabe G. and Craig B. *Introduction to the Practice of Statistics*, 7th edition, W.H. Freeman, N.Y., 2012.

28. Nalimov V. V. and Chernova N. A. *Statisticheskie metody planirovaniya ekstremal'nyx eksperimentov*, M.: Nauka, 1965 (In Russian); English translation: Vassili Nalimov, *Statistical Methods for Design of Extremal Experiments*, Washington, D.C.: Foreign Technology Div., U.S. Air Force Systems Command, Wright-Patterson AFB, Ohio, USA, 1968.

29. Nalimov V. V. *Primenenie matematicheskoy statistiki pri analize veshchestva*, M. Fizmatgiz, 1960. (In Russian) English translation: The application of Mathematical Statisyics in Chemical Analysis, Oxford: Pergamon Press, 1963.

30. Nalimov V. V. *Kanatohodets*, M. Izdatel'skaya gruppa Progress, 1994, (In Russian) http://www.koob.ru/nalimov_v_v/kanatohodetc

31. Pacheco P. S. *Parallel Programming with MPI*. Morgan Kaufmann, San Francisco, CA, 1996. http://www.cs.usfca.edu/ peter/ppmpi/

32. Slobodchikova R.I., Freidlina V. L., Lapina Z. S., and Nalimov V. V. Enhancing effectiveness of Random Balance Method, *Industrial Laboratory*, 32, 1, 53-58, 1966.

33. Tibshirani R. Regression Shrinkage and Selection via the LASSO. *J. Royal. Statist. Soc. B*, **58**, 1996, 267–288.

34. Viemeister J. : Die Theorie Selectirenden Versuche und Multiple–Access–Kanäle, *Diplomarbeit, Fakulatät für Mathematik, Universität Bielefeld*, (1982)

35. Zubashich V. F., Lysyansky A. V., and Malyutov M. B., Block–Randomized Distributed Trouble–Shooting Construction in Large Circuits with Redundancy. *Izvestia of the USSR Acad. of Sci., Technical Cybernetics*, No. 6, 1976 .

36. https://computing.llnl.gov/tutorials/mpi/

37. http://mpitutorial.com/mpi-scatter-gather-and-allgather/

38. http://mpitutorial.com/mpi-broadcast-and-collective-communication/

39. https://computing.llnl.gov/linux/slurm/

40. http://slurm.schedmd.com/mpi_guide.html

Appendix. Sketch of comparative portraits — G. E. P. Box (1919-2013) and V. V. Nalimov (1910-1997)

Let us reveal additional information on two heroes of our main part and striking difference of conditions they worked under.

Statistics in England has been highly respected, especially since F. Galton's 'Statistics heir' Karl Pearson, the head of Biometry lab and co-founder of journal 'Biometrika'. English Statistics has had excellent tradition of combining theoretical and applied studies. This is fully true for R. A. Fisher and his son in law, G. E. P. Box. The WW2 prevented the latter from completing his Chemistry B. S. He spent WW2 in a military anti-poison-gas detachment participating in experiments over animals where he learned importance of statistical methods. He was sent once to R. A. Fisher for consultations. He was paid by the Army to get the Statistics B. S. in one of London Universities under supervision of E. Pearson, K. Pearson's son. Incidentally, he had his internships in Imperial Chemicals, who offered him a job and paid for his fulfilling Masters in Statistics. He worked there 8 years (with some breaks) over yield optimization of their production. Then he moved to NC University, Raleigh and to the Princeton University in the US. Eventually, he organized a Statistics Dept. at Wisconsin which became a center of excellence in the field. His most famous quotation is 'All models are wrong but some are useful'.

Soviet leaders hated Statistics because of its objectivity and completely eliminated it in Economics, Sociology, Biology, etc. The reason for it to survive at all was preparation of the main present to the 70th anniversary of tyran I. Stalin—nuclear bomb. Physisists and mathematicians involved in that project were almost free to do whatever they regarded helpful for the project. In a top level official meeting on what to do with Probability and Statistics, mathematicians pretended to agree with the slogan 'Science is against randomness', but insisted that randomness must be studied to fight against it. They were supported by a mighty artillery fire dispersion research group that had been assisted by Kolmogorov during the WW2. Thus, several centers were allowed to continue work including the Statistics Dept., Steklov Mathematics Institute in Moscow headed by the coauthor of the Kolmogorov-Smirnov celebrated criterion. L. N. Bolshev, Smirnov's deputy, a brilliant researcher and lecturer, a WW2-pilot-fighter, will play a role in our future story.

V. V. Nalimov left Mathematics Dept. of Moscow University in 1929 after the first year of education in protest to discrimination practices cultivated there. He worked both in civil and military research labs before imprisonment in 1936 for his involvement in social activities that were regarded as subversive by the authorities. He managed to do almost impossible thing—survive in labor camps, and was transferred to the 'industrial research facility under the bars—sharashka' inside the Metallurgy plant in Magadan until 1947 He worked with geologists a couple of years followed by an automatic repeated arrest and transfer to the same kind of sharashka' in Kazakhstan, 1949-1953 until his release by amnesty in 1953. There he could request access to the Western research literature and became excited about applying statistical methods. V.

V. Nalimov was rehabilitated in 1957. In several years he organized a lab in ‘GIREDMET’, where he continued research he carried out in ‘sharashka’ and prepared two dissertations and his first large book [29]. This period of his work is described in [1]. V. V. Nalimov was fortunate with a reviewer and editor of his first book—L. N. Bolshev, who improved its readability and recommended V. V. Nalimov to A. N. Kolmogorov for a position in a newly organized Lab. of Statistical Methods in Moscow University. V. V. Nalimov himself describes this period of his life in [30]. Due to his enthusiasm and successful advertisement, statistics and design became very popular in the Soviet industrial and applied science communities. G. E. P. Box visited V. V. Nalimov in the Kolmogorov Lab and brought there later a JMP team for two visits headed by W. J. Dixon. It is amazing that working under almost intolerable conditions, V. V. Nalimov managed to contribute significantly to the development of several fields including statistics and experimental design. This example enables the hope that these sciences will resurrect one day in Russia.