Experimental Designs under Resource Constraints: Algorithmic Construction

Alena Bachratá
joint work with Lenka Filová and Radoslav Harman

Comenius University in Bratislava,
Slovakia

LinStat 2014, 27.8.2014

Outline

- Optimal design of experiments
- Resource constraints
- Algorithm
- Applications

What is experimental design?

- Optimal design of experiments is an approach to constructing efficient experimental designs using a statistically motivated optimality criterion
- The experiment arranged by the optimal design can save your time, money or material
- Applications are in agriculture, industry, medical and genetic research and many other areas

What is experimental design?

Exact experimental design

$$\xi \in \{0, 1, 2 \ldots\}^n$$

$$\xi = [\xi(x_1), \xi(x_2), \ldots, \xi(x_n)]^T \,.$$

$\xi(x_i)$ - the number of replicated trails in the design point $x_i \in \mathfrak{X}$

$\mathfrak{X}$ - a finite design space $\mathfrak{X} = \{x_1, \ldots, x_n\}$

Approximate experimental design

$$\xi \in [0, \infty)^n.$$

What is experimental design?

Exact experimental design

$$\xi \in \{0, 1, 2 \dots\}^n$$

$$\xi = [\xi(x_1), \xi(x_2), \dots, \xi(x_n)]^T .$$

$\xi(x_i)$ - the number of replicated trails in the design point $x_i \in \mathfrak{X}$

$\mathfrak{X}$ - a finite design space $\mathfrak{X} = \{x_1, \dots, x_n\}$

Approximate experimental design

$$\xi \in [0, \infty)^n.$$

A measure of the quality of designs

Optimality criterion

$$\phi : [0, \infty)^n \to [0, \infty).$$

We estimate unknown parameters of an underlying statistical model, and the value $\phi(\xi)$ is a measure of the information about the parameters of interest obtained from the experiment ξ.

For example, we can use D, E or A optimality criterion that maximizes the determinant, the minimum eigenvalue or minimizes a trace of an inverse of the Fisher information matrix, respectively.

Monotonicity

Augmentation of an experiment by additional trials cannot decrease its quality for statistical inference, i.e., if designs ξ and ζ satisfy $\xi \leq \zeta$ componentwise, then $\phi(\xi) \leq \phi(\zeta)$.

A measure of the quality of designs

Optimality criterion

$$\phi : [0, \infty)^n \to [0, \infty).$$

We estimate unknown parameters of an underlying statistical model, and the value $\phi(\xi)$ is a measure of the information about the parameters of interest obtained from the experiment ξ.

For example, we can use D, E or A optimality criterion that maximizes the determinant, the minimum eigenvalue or minimizes a trace of an inverse of the Fisher information matrix, respectively.

Monotonicity

Augmentation of an experiment by additional trials cannot decrease its quality for statistical inference, i.e., if designs ξ and ζ satisfy $\xi \leq \zeta$ componentwise, then $\phi(\xi) \leq \phi(\zeta)$.

Resource constraints

Standard constraint on design

$$\xi(x_1) + ... + \xi(x_n) \leq N.$$

General resource constraints

$$\sum_{i=1}^{n} a_r(x_i)\xi(x_i) \leq b_r, \quad r \in \{1, \dots, k\},$$

$a_r(x_i)$ - the consumption of the r-th resource by a single trail in x_i

$\xi(x_i)$ - the number of trails in x_i

b_r - a limit on the r-th resource

Resource constraints

General resource constraints

$$\sum_{i=1}^{n} a_r(x_i)\xi(x_i) \leq b_r, \quad r \in \{1, \ldots, k\}.$$

Assumptions

- $b_1, ..., b_k > 0$
- $a_r(x_i) \geq 0$ for all $r \in \{1, \ldots, k\}$ and $i \in \{1, \ldots, n\}$
- for all $i \in \{1, \ldots, n\}$ there exists $r \in \{1, \ldots, k\}$ such that $a_r(x_i) > 0$

Matrix form

$$\mathbf{A}\xi \leq \mathbf{b}.$$

Resource constraints

General resource constraints

$$\sum_{i=1}^{n} a_r(x_i)\xi(x_i) \leq b_r, \qquad r \in \{1, \ldots, k\}.$$

Examples

- The total number of all trials

$$\xi(x_1) + ... + \xi(x_n) \leq N, \qquad a_1(x_i) = 1, b_1 = N$$

- The number of trials in design point $x_i, i \in \{1, 2, \ldots, n\}$

$$\xi(x_i) \leq N_i, \qquad a_2(x_i) = 1, a_2(x_j) = 0 \text{ for } j \neq i, b_2 = N_i$$

- The total cost of the experiment

$$a_3(x_i) = \text{cost}(x_i) \text{ for all } i \in \{1, 2, \ldots, n\}, \quad b_3 = \text{budget}$$

Initial design and set of all feasible designs

Initial design $\xi^{(0)}$ can be $\xi^{(0)}(x_1) = \ldots = \xi^{(0)}(x_n) = 0$, or it can be formed by the observations from a previous experiment.

The set of all feasible exact designs

$$\Xi^{\mathrm{ex}} = \{\xi \in \mathbb{Z}_+^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}.$$

The set of all feasible approximate designs

$$\Xi^{\mathrm{ap}} = \{\xi \in [0,\infty)^n : \xi^{(0)} \leq \xi,\ \mathbf{A}\xi \leq \mathbf{b}\}.$$

Maximal design $\xi^M \in \Xi^{\mathrm{ex}}$ is a design that can not be augmented without violation of some of the resource constraints.

Initial design and set of all feasible designs

Initial design $\xi^{(0)}$ can be $\xi^{(0)}(x_1) = \ldots = \xi^{(0)}(x_n) = 0$, or it can be formed by the observations from a previous experiment.

The set of all feasible exact designs

$$\Xi^{\mathrm{ex}} = \{\xi \in \mathbb{Z}_+^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}.$$

The set of all feasible approximate designs

$$\Xi^{\mathrm{ap}} = \{\xi \in [0,\infty)^n : \xi^{(0)} \leq \xi,\ \mathbf{A}\xi \leq \mathbf{b}\}.$$

Maximal design $\xi^M \in \Xi^{\mathrm{ex}}$ is a design that can not be augmented without violation of some of the resource constraints.

Initial design and set of all feasible designs

Initial design $\xi^{(0)}$ can be $\xi^{(0)}(x_1) = \ldots = \xi^{(0)}(x_n) = 0$, or it can be formed by the observations from a previous experiment.

The set of all feasible exact designs

$$\Xi^{\mathrm{ex}} = \{\xi \in \mathbb{Z}_+^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}.$$

The set of all feasible approximate designs

$$\Xi^{\mathrm{ap}} = \{\xi \in [0, \infty)^n : \xi^{(0)} \leq \xi,\ \mathbf{A}\xi \leq \mathbf{b}\}.$$

Maximal design $\xi^M \in \Xi^{\mathrm{ex}}$ is a design that can not be augmented without violation of some of the resource constraints.

The main purpose

The general resource-constrained exact optimal design

$$\xi^* \in \arg\max\{\phi(\xi) : \xi \in \Xi^{\mathrm{ex}}\},$$

$$\Xi^{\mathrm{ex}} = \{\xi \in \mathbb{Z}_+^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}.$$

We can write and solve also many problems from other mathematical areas, in the form of the general problem

- knapsack problem
- t-optimal graphs in the graph theory
- optimal redundancy allocation in the reliability theory

The main purpose

The general resource-constrained exact optimal design

$$\xi^* \in \arg\max\{\phi(\xi) : \xi \in \Xi^{\mathrm{ex}}\},$$

$$\Xi^{\mathrm{ex}} = \{\xi \in \mathbb{Z}_+^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}.$$

We can write and solve also many problems from other mathematical areas, in the form of the general problem

- knapsack problem
- t-optimal graphs in the graph theory
- optimal redundancy allocation in the reliability theory

Example - Marine engineer

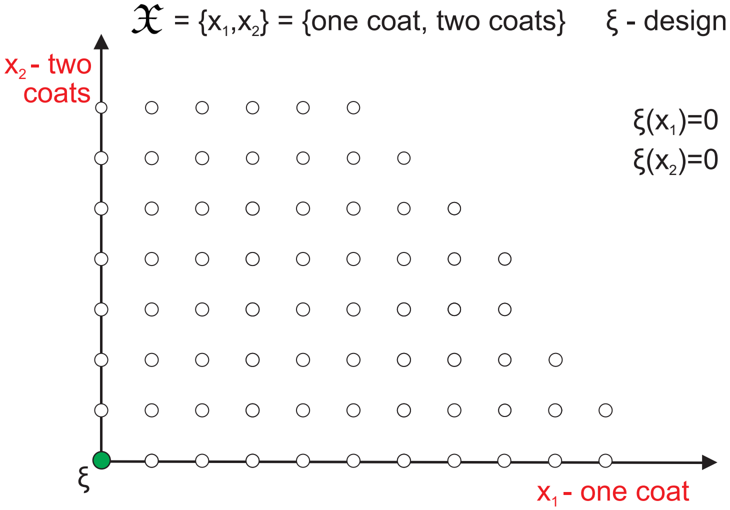
Inspired by *Design of Comparative Experiments*, R. A. Bailey, p.41, ex.2.2

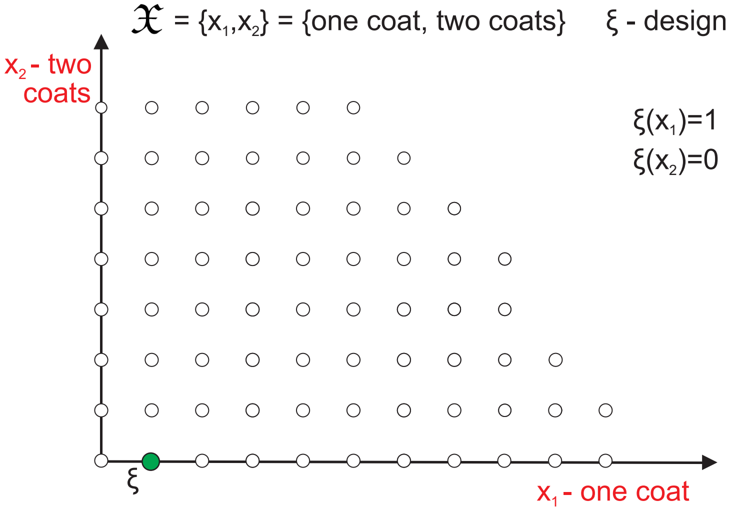
A marine engineer wants to protect underwater metal components against corrosion. His colleague has developed a new paint for that. The engineer would like to estimate the degree of protection with *one coat* and with *two coats* of paint.

He will paint some components once, some twice. Then immerse them all in the tank of sea water. Later he will remove all of them, and measure the amount of corrosion of each.

The tank has room only for 10 components. The paint is new, and there is enough for 14 coats.

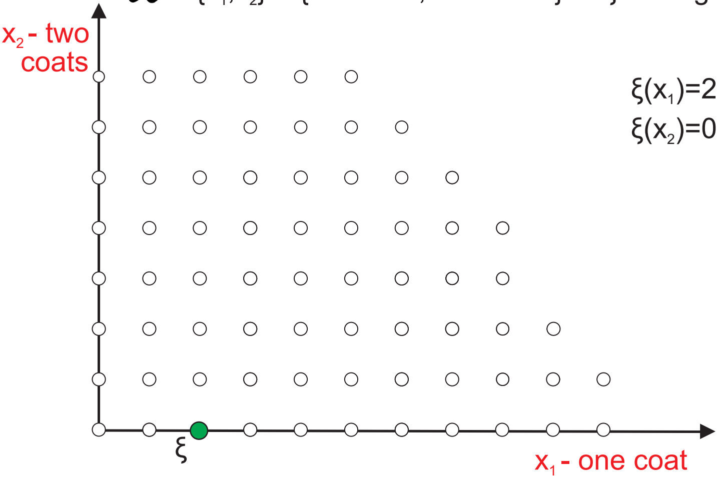
Advise the engineer how best to use his resources in his experiment.


$\mathfrak{X}$ = {x_1,x_2} = {one coat, two coats}
ξ - design
x_2 - two coats
$\xi(x_1)=0$
$\xi(x_2)=0$
ξ
x_1 - one coat

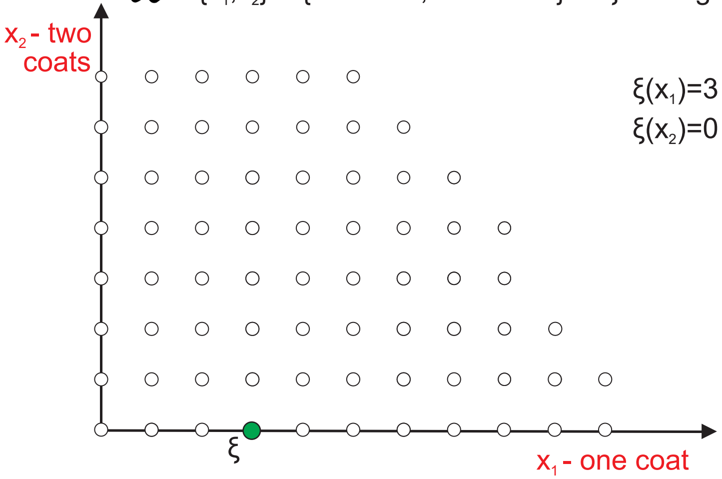


𝔛 = {x1,x2} = {one coat, two coats}
ξ - design
x2 - two coats
ξ(x1)=1
ξ(x2)=0
ξ
x1 - one coat

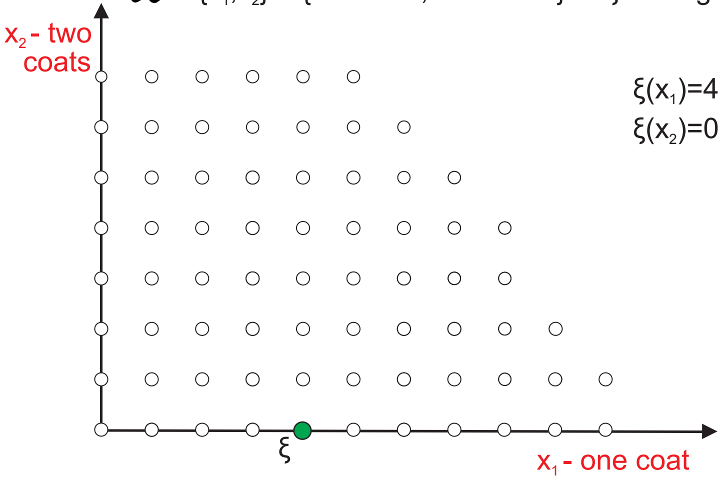
$\mathfrak{X} = \{x_1, x_2\} = \{\text{one coat, two coats}\}$ ξ - design



$\mathfrak{X} = \{x_1, x_2\} = \{\text{one coat, two coats}\}$ ξ - design



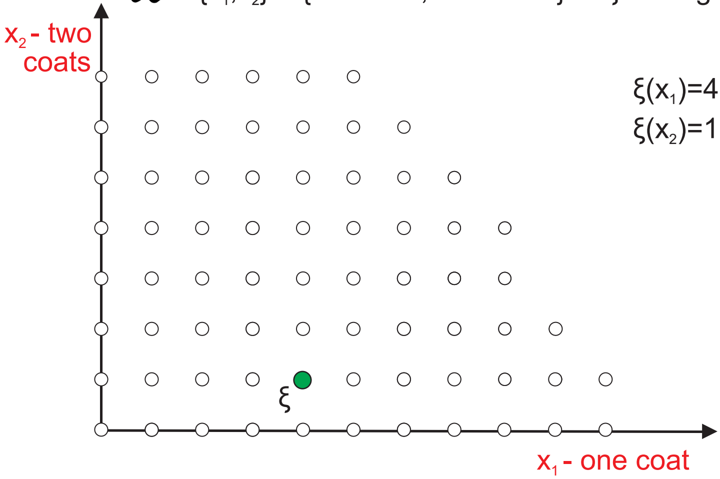
$\mathfrak{X} = \{x_1, x_2\} = \{\text{one coat, two coats}\}$ ξ - design



$\xi(x_1)=4$

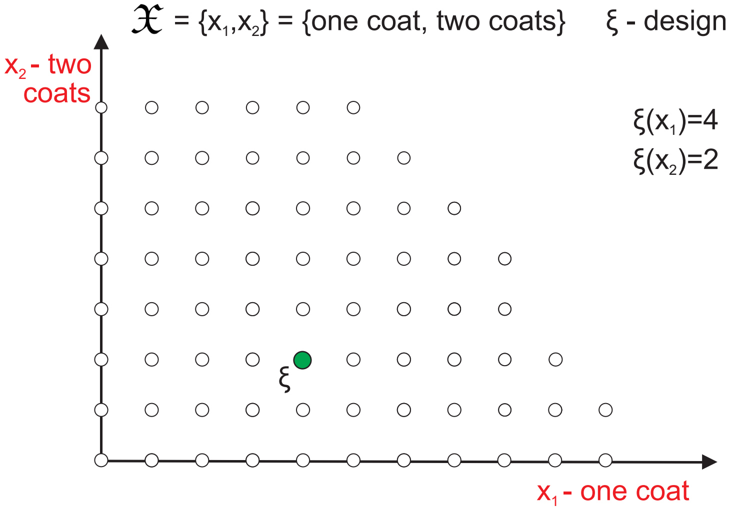
$\xi(x_2)=0$

$\mathfrak{X} = \{x_1, x_2\} = \{\text{one coat, two coats}\}$ $\quad \xi$ - design

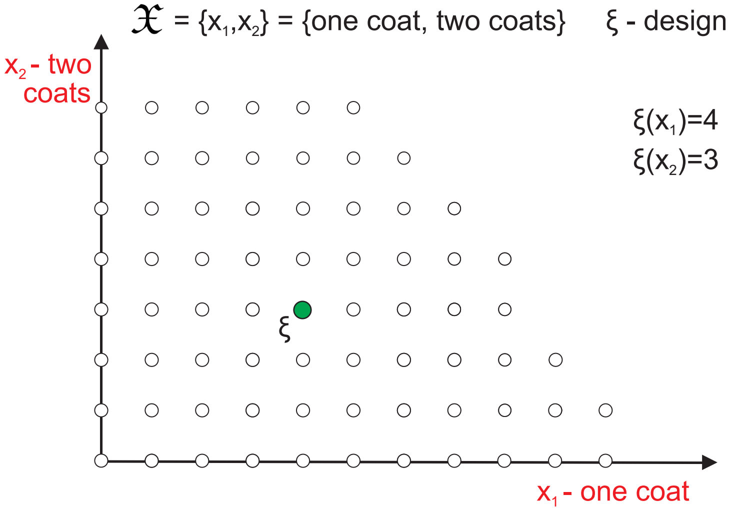


$\xi(x_1)=4$

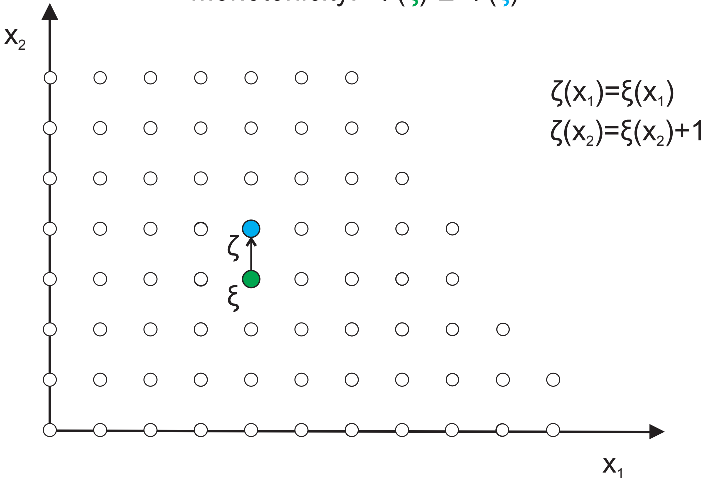
$\xi(x_2)=1$



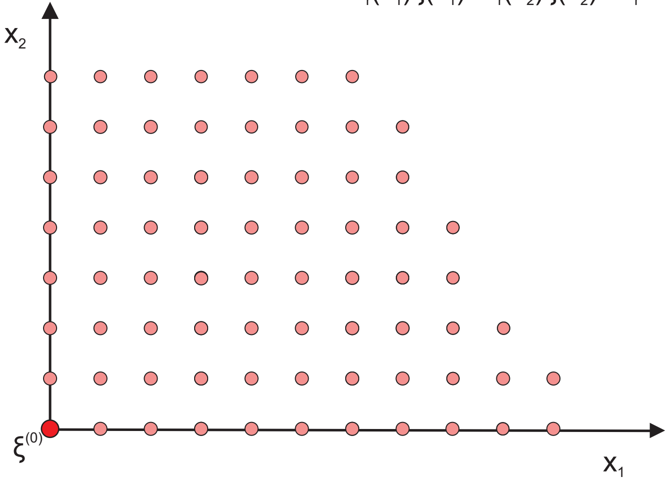
𝔛 = {x1,x2} = {one coat, two coats}
ξ - design
x2 - two coats
ξ(x1)=4
ξ(x2)=2
ξ
x1 - one coat


𝔛 = {x1,x2} = {one coat, two coats}
ξ - design
x2 - two coats
ξ(x1)=4
ξ(x2)=3
ξ
x1 - one coat

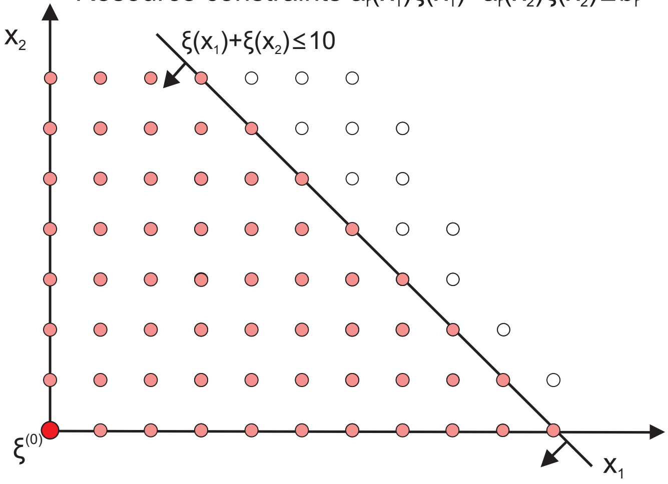
Monotonicity: $\Phi(\xi) \leq \Phi(\zeta)$

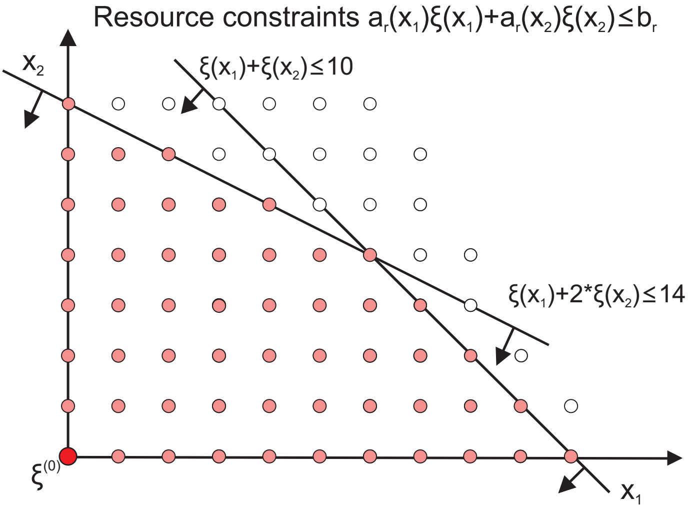


Resource constraints $a_r(x_1)\xi(x_1)+a_r(x_2)\xi(x_2)\leq b_r$

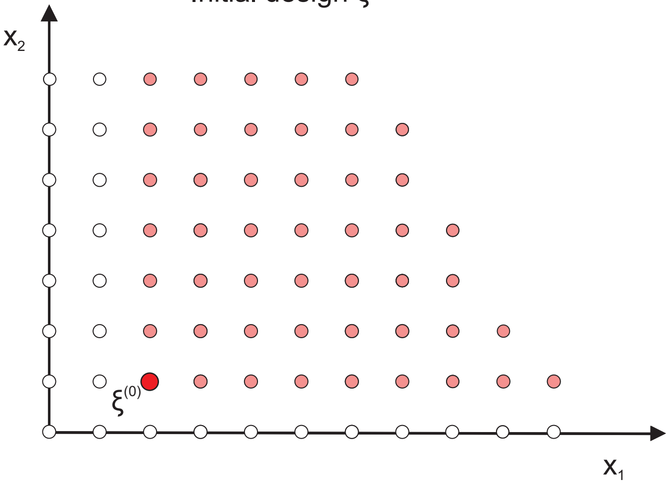


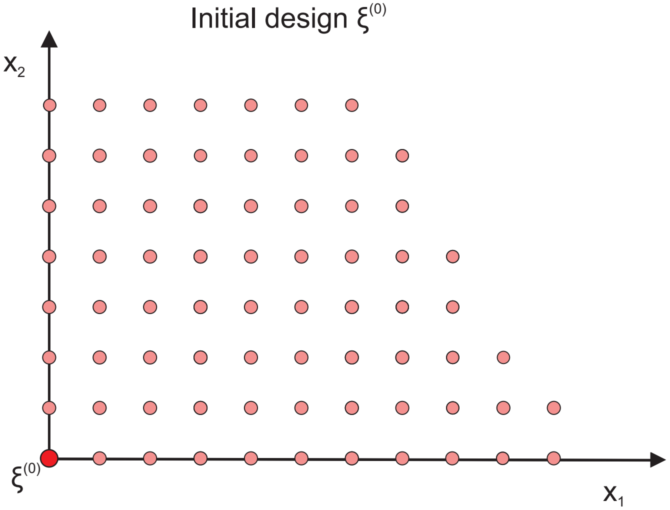
Resource constraints $a_r(x_1)\xi(x_1)+a_r(x_2)\xi(x_2)\leq b_r$

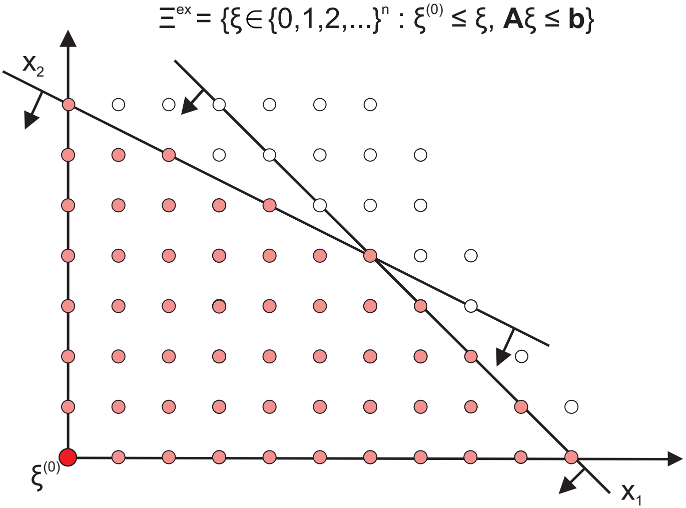



Resource constraints $a_r(x_1)\xi(x_1)+a_r(x_2)\xi(x_2)\leq b_r$
x_2
$\xi(x_1)+\xi(x_2)\leq 10$
$\xi(x_1)+2*\xi(x_2)\leq 14$
$\xi^{(0)}$
x_1

Initial design $\xi^{(0)}$

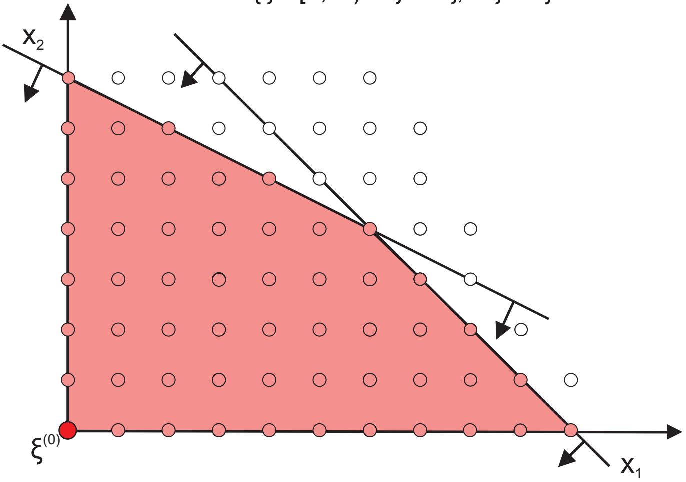



Initial design ξ(0)
x2
ξ(0)
x1

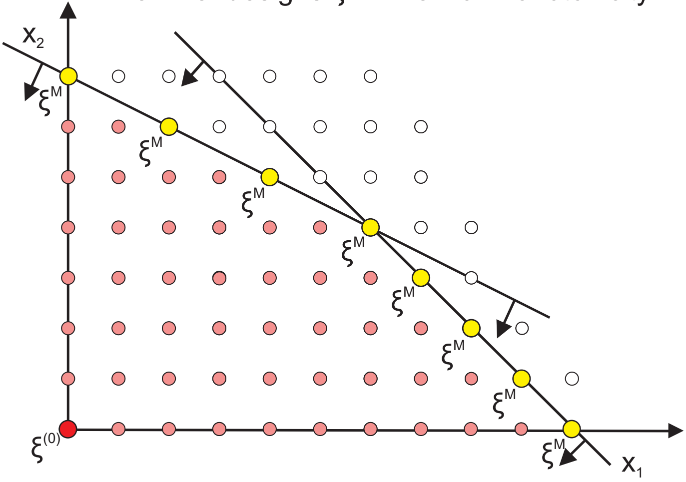


$\Xi^{ex} = \{\xi \in \{0,1,2,\dots\}^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}$
x_2
$\xi^{(0)}$
x_1

$$\Xi^{ap} = \{\xi \in [0,\infty)^n : \xi^{(0)} \leq \xi, \mathbf{A}\xi \leq \mathbf{b}\}$$



Maximal designs ξ^M in view of monotonicity



Known algorithms

$$\xi^* \in \arg\max\{\phi(\xi) : \xi \in \Xi^{\mathrm{ex}}\}$$

Small-size problems can be solved by using enumeration methods (e.g. branch and bound). That guarantees finding a globally optimal solution.

Large problems can't be solved by an algorithm that can find a provably optimal solution. We can use only a heuristic that leads to an efficient feasible experimental design.

For the standard constraint, there exist several methods – exchange algorithms, the Detmax algorithm (Mitchell 1974), rounding of approximate designs, etc.

For general resource constraints, there is no universal method. Only some methods for computing optimal designs under particular types of constraints.

Algorithm

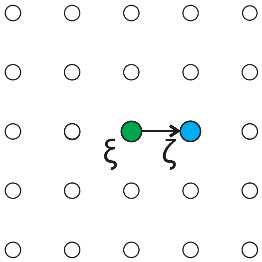
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Forward step

$$\zeta = \xi + \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) + 1, \ldots, \xi(x_n))$$

Upper neighbours

$$\mathcal{U}(\xi) = \{\xi + \mathbf{e}_1, ..., \xi + \mathbf{e}_n\} \cap \Xi^{\mathrm{ex}}$$



Algorithm

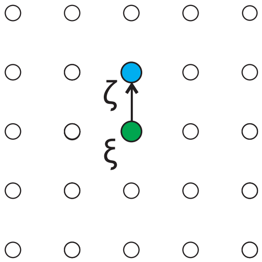
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Forward step

$$\zeta = \xi + \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) + 1, \ldots, \xi(x_n))$$

Upper neighbours

$$\mathcal{U}(\xi) = \{\xi + \mathbf{e}_1, ..., \xi + \mathbf{e}_n\} \cap \Xi^{\mathrm{ex}}$$



Algorithm

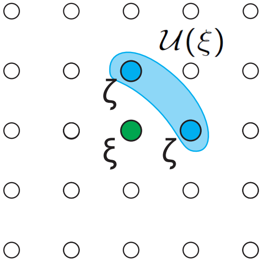
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Forward step

$$\zeta = \xi + \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) + 1, \ldots, \xi(x_n))$$

Upper neighbours

$$\mathcal{U}(\xi) = \{\xi + \mathbf{e}_1, ..., \xi + \mathbf{e}_n\} \cap \Xi^{\text{ex}}$$



Algorithm

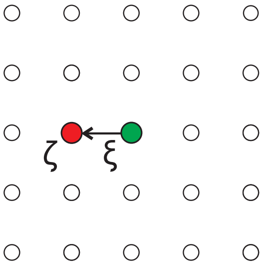
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Backward step

$$\zeta = \xi - \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) - 1, \ldots, \xi(x_n))$$

Lower neighbours

$$\mathcal{L}(\xi) = \{\xi - \mathbf{e}_1, ..., \xi - \mathbf{e}_n\} \cap \Xi^{\mathrm{ex}}$$



Algorithm

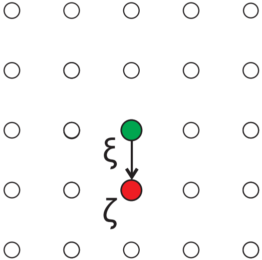
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Backward step

$$\zeta = \xi - \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) - 1, \ldots, \xi(x_n))$$

Lower neighbours

$$\mathcal{L}(\xi) = \{\xi - \mathbf{e}_1, ..., \xi - \mathbf{e}_n\} \cap \Xi^{\mathrm{ex}}$$



Algorithm

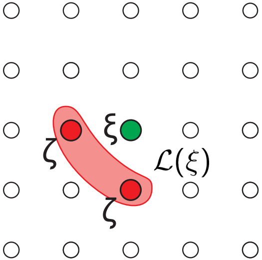
The algorithm is based on excursions in the set of all feasible designs Ξ^{ex}.

Backward step

$$\zeta = \xi - \mathbf{e}_i = (\xi(x_1), \ldots, \xi(x_i) - 1, \ldots, \xi(x_n))$$

Lower neighbours

$$\mathcal{L}(\xi) = \{\xi - \mathbf{e}_1, ..., \xi - \mathbf{e}_n\} \cap \Xi^{\mathrm{ex}}$$



Attribute and the local heuristic evaluation of ξ

Attribute of the design – $attr(\xi)$

It assigns different values to substantially different designs and the same value to essentially same designs. We choose $attr(\xi) = \Phi(\xi)$ rounded to a given number of significant digits.

Tabu list T

The list of attributes of already visited designs. Instead of storing complete designs we store only real-valued attributes and save time and computer memory.

Local heuristic evaluation of the design – $val(\xi)$

A real number that roughly estimates how promising is ξ as a part of an excursion leading to an efficient design.

Local heuristic evaluation of ξ – $val(\xi)$

Let

$$val^*(\xi) = \max\{\phi(\zeta) : \zeta \in \Xi^{\mathrm{ex}}, \xi \leq \zeta\}$$

be an ideal evaluation leading to optimal design. We estimate $val^*(\xi)$ by a "max. approximate design" accessible from the design ξ.

For every $i \in \{1, \ldots, n\}$, let $d_i(\xi) = \max\{d \geq 0 : \xi + d\mathbf{e}_i \in \Xi^{\mathrm{ex}}\}$. The estimate of the direction towards "large" feasible designs is

$$\mathbf{d}(\xi) = (d_1(\xi), ..., d_n(\xi))^T.$$

Let $\gamma(\xi) = \max\{\gamma \geq 0 : \xi + \gamma\mathbf{d}(\xi) \in \Xi^{\mathrm{ap}}\}$. Then the largest feasible approximate design in the direction $\mathbf{d}(\xi)$ is $\xi + \gamma(\xi)\mathbf{d}(\xi)$.

The value

$$val(\xi) = \phi(\xi + \gamma(\xi)\mathbf{d}(\xi))$$

gives us a rough estimate of $val^*(\xi)$.

Algorithm

1-a. Let ξ be the current design. If $attr(\xi) \notin T \Rightarrow$ make a forward step to a design ζ which is chosen such that it maximizes *val* among all designs satisfying $attr(\zeta) \notin T$.

1-b. If $attr(\xi) \in T \Rightarrow$ make a backward step to a design ζ which is chosen such that it maximizes *val* among all designs satisfying $attr(\zeta) \notin T$.

2. If we attempt a step, but there is no allowed design, then we reverse the direction of the search. If all these attempts fail, then we move to a random neighboring design.
3. Each time a maximal design is visited, the algorithm checks whether it is better than the best design ξ^* found so far.
4. After too many backward steps the excursion failed. We start a new excursion from the currently best design.
5. The algorithm stops after given time.

Application

We tested the algorithm on three different designs problems:

Designs for a block model with a constraint on the number of blocks and on the number of uses of individual treatments

Designs for a quadratic model with simultaneous marginal and cost constraints

Harman, R., Filová, L. (2014): *Computing efficient exact designs of experiments using integer quadratic programming.*

Designs for a non-linear regression model with simultaneous direct and cost constraints

Wright SE, Sigal BM, Bailer AJ (2010): *Workweek Optimization of Experimental Designs: Exact Designs for Variable Sampling Costs*

Design for a block model a)

Consider a block model with N blocks of size two and v treatments. Assume that the independent observations $Y_1, ..., Y_N$ satisfy

$$E(Y_j) = \tau(t_1(j)) - \tau(t_2(j)), \; j \in \{1, \dots, N\}.$$

Where $t_1(j)$ and $t_2(j)$ are the treatments selected for the j-th block, with effects $\tau(t_1(j))$ and $\tau(t_2(j))$. $Var(Y_j) = \sigma^2 < \infty$.

An experimental design is given by a selection of treatments $t_1(j), t_2(j) \in \{1, \dots, v\}$ to be used in the j-th block.

The design space can be viewed as the set of all possible pairs of treatments

$$\mathfrak{X} = \{(1,2), (1,3), \dots, (v-1, v)\}, \; |\mathfrak{X}| = n.$$

Design for a block model

Only restriction is not to exceed the given number N of blocks

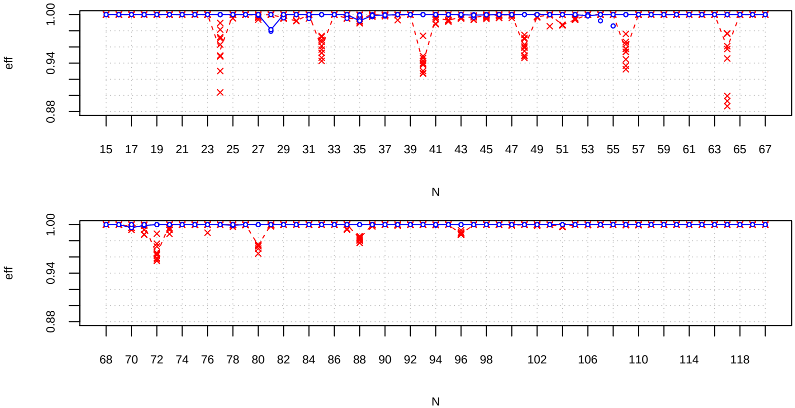
$$\xi(x_1) + ... + \xi(x_n) \leq N.$$

We have implemented our algorithm in the environment R and used it to compute D-efficient designs for $v = 16$ treatments and $N = 15, \ldots, 120$ blocks.

We compared the designs with the results of a simulated annealing method implemented in R package "smida".

We run both algorithms for 2 minutes. In most cases our heuristic found either the same or better designs than simulated annealing.

Design for a block model



N - number of blocks
eff - *D*-efficiencies relative to the best found exact design
Our algorithm – blue circles, Simulated annealing – red crosses

Design for a block model b)

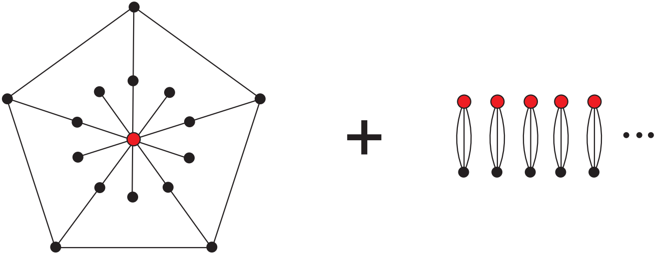
To illustrate the possibilities of our algorithm that go beyond the scope of the "smida" package, assume that we have no explicit limit on the number N of blocks, but upper limits on the numbers of uses of treatment samples.

Assume that 5 treatments can be used at most 4 times, another 5 treatments at most 5 times, another 5 treatments at most 6 times and one standard treatment at most 56 times. That creates $k = 16$ resource constraints.

We run algorithm ten times, each for 120 seconds.

Design for a block model

The algorithm produces the design with $N = 65$ blocks that consists of 20 blocks and of another 45 blocks that compare each of the first 15 treatments three times against the standard treatment 16.



To our best knowledge, there is no optimal design heuristic capable of solving this type of constraints.

Conclusions

Resource constraints

- cover many types of practical experimental constraints
- lead to a set of exact designs with relatively nice properties

Algorithm

- works for any monotonic optimality criterion
- computes efficient solutions for problems under diverse resource constraints

arXiv

`1402.7263 [stat.CO]`

References

[1] Bailey, R.A.: *Design of Comparative Experiments.* Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press (2008)

[2] Harman, R., Filová, L.: *Computing efficient exact designs of experiments using integer quadratic programming.* Comput. Stat. Data Anal. 71, 1159-1167 (2014)

[3] Mitchell, T.J.: *An Algorithm for the Construction of "D-Optimal" Experimental Designs.* Technometrics 16(2), 203-210 (1974)

[4] Wright, S.E., Sigal, B.M., Bailer, A.J.: *Workweek Optimization of Experimental Designs: Exact Designs for Variable Sampling Costs.* J. Agr. Biol. Environ. Stat. 15(4), 491-509 (2010)