Manual for Systematic Search and One Step

Department of Mathematics, Linköping University

Johan Persson

Credits: **16 hp**

Level: **G2**

Supervisor: **Frank Miller**,
Department of Statistics, Stockholm University

Examiner: **Zhenxia Liu**,
Department of Mathematics, Linköping University

Linköping: **July 2017**

Models

In one dimension.

Linear, $m_{1,L}$	$y = \beta_1 x_1 + \varepsilon$
Affine, $m_{1,A}$	$y = \beta_1 + \beta_2 x_1 + \varepsilon$
Quadratic, $m_{1,Q}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_1^2 + \varepsilon$
Cubic, $m_{1,C}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_1^2 + \beta_4 x_1^3 + \varepsilon$

In two dimensions.

Linear, $m_{2,L}$	$y = \beta_1 x_1 + \beta_2 x_2 + \varepsilon$
Affine, $m_{2,A}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \varepsilon$
Interaction, $m_{2,I}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_1 x_2 + \varepsilon$
Quadratic, $m_{2,Q}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_1 x_2 + \beta_5 x_1^2 + \beta_6 x_2^2 + \varepsilon$
Cubic, $m_{2,C}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_1 x_2 + \beta_5 x_1^2 + \beta_6 x_1^2 + \beta_7 x_1^3 + \beta_8 x_2^3 + \varepsilon$

In three dimensions.

Linear, $m_{3,L}$	$y = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$
Affine, $m_{3,A}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_3 + \varepsilon$
Interaction, $m_{3,I}$	$y = \beta_1 + \beta_2 x_1 + \beta_3 x_2 + \beta_4 x_3 + \beta_5 x_1 x_2 + \beta_6 x_1 x_3 + \beta_7 x_2 x_3 + \varepsilon$
Quadratic, $m_{3,Q}$	$y = Interaction + \beta_8 x_1^2 + \beta_9 x_2^2 + \beta_{10} x_3^2 + \varepsilon$
Cubic, $m_{3,C}$	$y = Quadratic + \beta_{11} x_1^3 + \beta_{12} x_2^3 + \beta_{13} x_3^3 + \varepsilon$

Design regions

One factor (d=1).

Full design space.

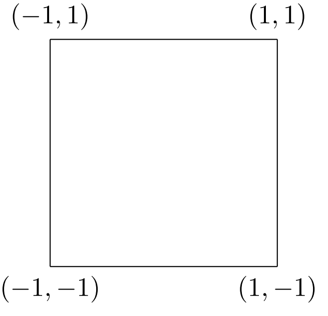
-1 ———————————— 1

With only one factor the cut-off region and the cut-out region look the same as the full design space.

Two factors (d=2).

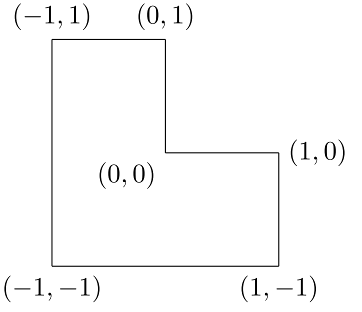
Full design space.

$\{\boldsymbol{x} = (x_1, x_2) : x_1, x_2 \in [\text{-}1, 1]\}$



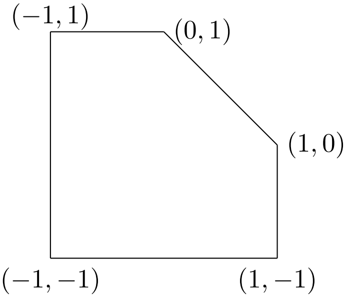
Cut-out region.

$\{\boldsymbol{x} = (x_1,\, x_2) : x_1,\, x_2 \in$ [-1, 1], $x_1 \vee x_2 \leq 0\}$
$\vee$ meaning at least one of x_1 and $x_2 \leq 0$



Cut-off region.

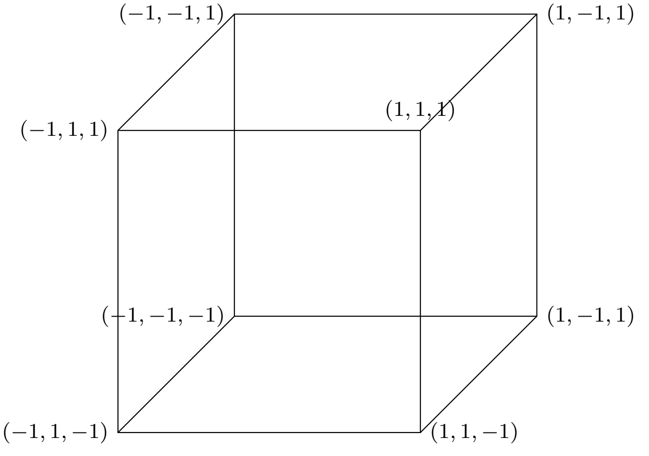
$\{\boldsymbol{x} = (x_1,\, x_2) : x_1,\, x_2 \in$ [-1, 1], $x_1 + x_2 \leq 1\}$



Three factors (d=3).

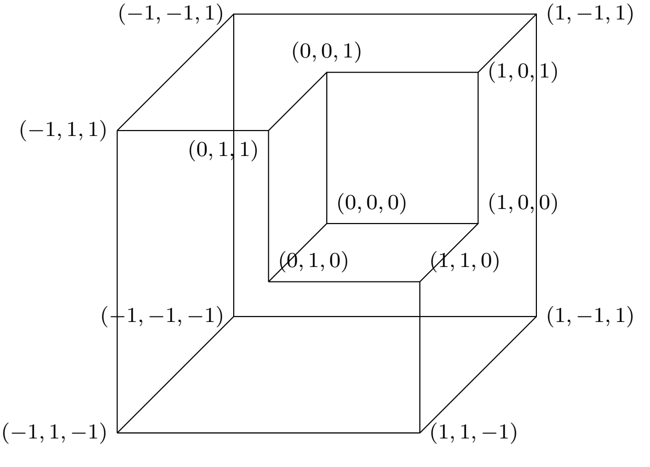
Full design space.

$\{\boldsymbol{x} = (x_1,\, x_2,\, x_3) : x_1,\, x_2,\, x_3 \in [\text{-1, 1}]\}$



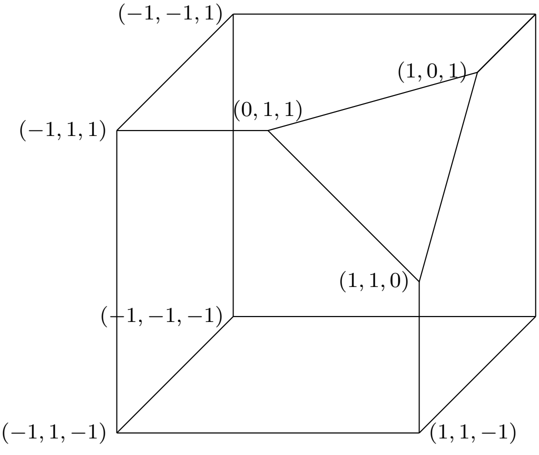
Cut out-design region.

$\{\boldsymbol{x} = (x_1,\, x_2,\, x_3) : x_1,\, x_2,\, x_3 \in [\text{-1, 1}],\, x_1 \vee x_2 \vee x_2 \leq 0\}$



Cut off-design region.

$$\{\boldsymbol{x} = (x_1,\, x_2,\, x_3) : x_1,\, x_2,\, x_3 \in [\text{-}1,\, 1],\, x_1 + x_2 + x_3 \leq 1\}\}$$



Program description

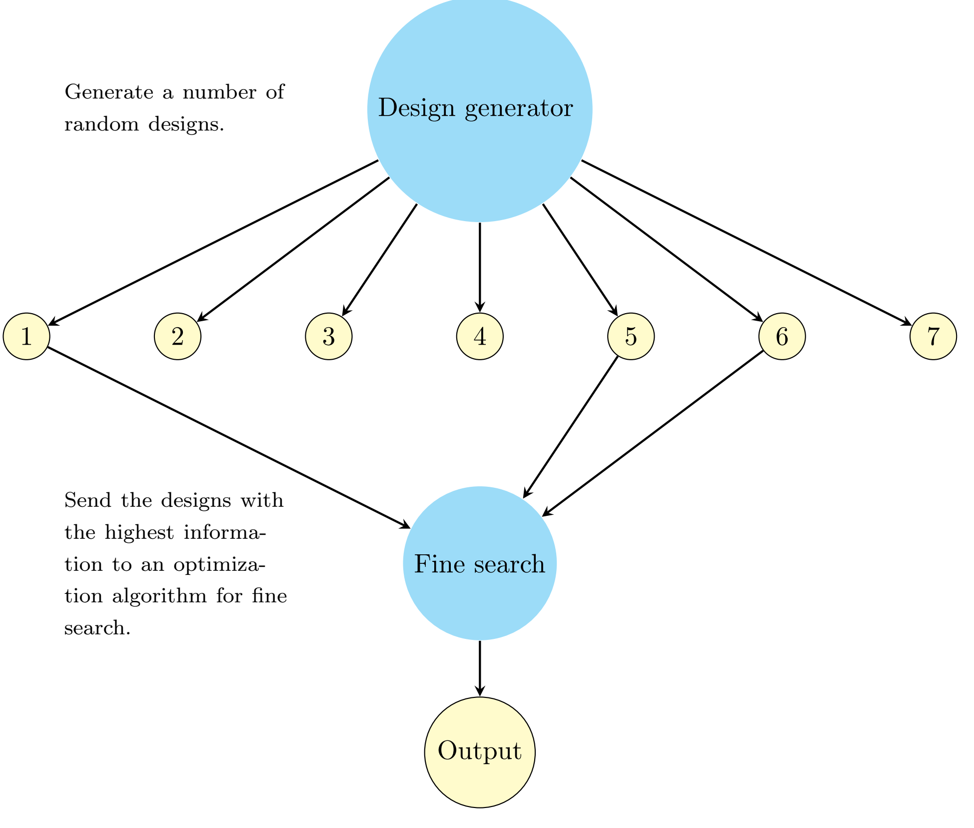


Figure 1: A schematic of what the program does.

0.0.1 Program description

Function description

Spans a 1, 2 or 3-dimensional space.

Restricts the design space due to technical or medical reasons.

Randomly generates designs, D. The entries in D are the number of repetitions N_i at each design point x_i.

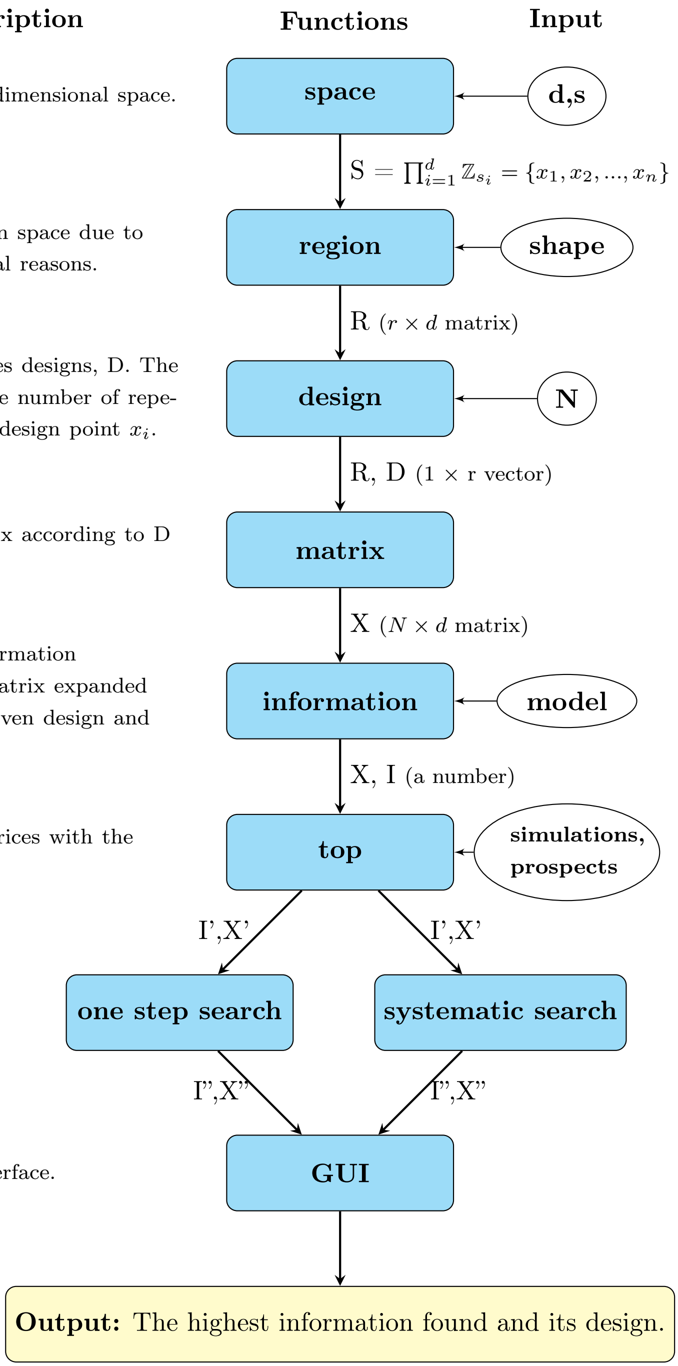
Constructs a matrix according to D and R.

Calculates the information $\det(M^T M)$ of a matrix expanded according to the given design and model.

Takes out the matrices with the most information.

Two different ways of finding designs with maximum information.

Graphical user interface.



d is the number of dimensions, 1,2 or 3. $\mathbf{s} = (l_1, \ldots, l_d)$ is the vector of levels each dimension may take.

shape $\in$ {full design space, cut-out design region, cut-off design region}

The number of trials, $\mathbf{N} \in \mathbb{N}$.

model $\in$ {Linear, Affine, Interaction, Quadratic, Cubic}

simulations $\in \mathbb{N}$, the number of random designs created. **prospects** $\leq$ **simulations**, the number of designs with the highest information sent to fine search.

We'll now describe in detail what our program does and how each algorithm works. The program starts of by generating a number of random designs, this is done by the code modules *shape*, *region* and *design*. In the program implementation the random designs are called *simulations*. The code modules *matrix* and *information* creates model matrices and calculates the designs's information. The number of highest information designs to be sent to an algorithm is given by the number of *prospects*. The module *top* picks out this (prospect) number of best designs and sends it to one of the two algorithms. Which algorithm is used depends on if you've opened the program *Systematic Search* or *One Step*.

This is how *One Step* works:

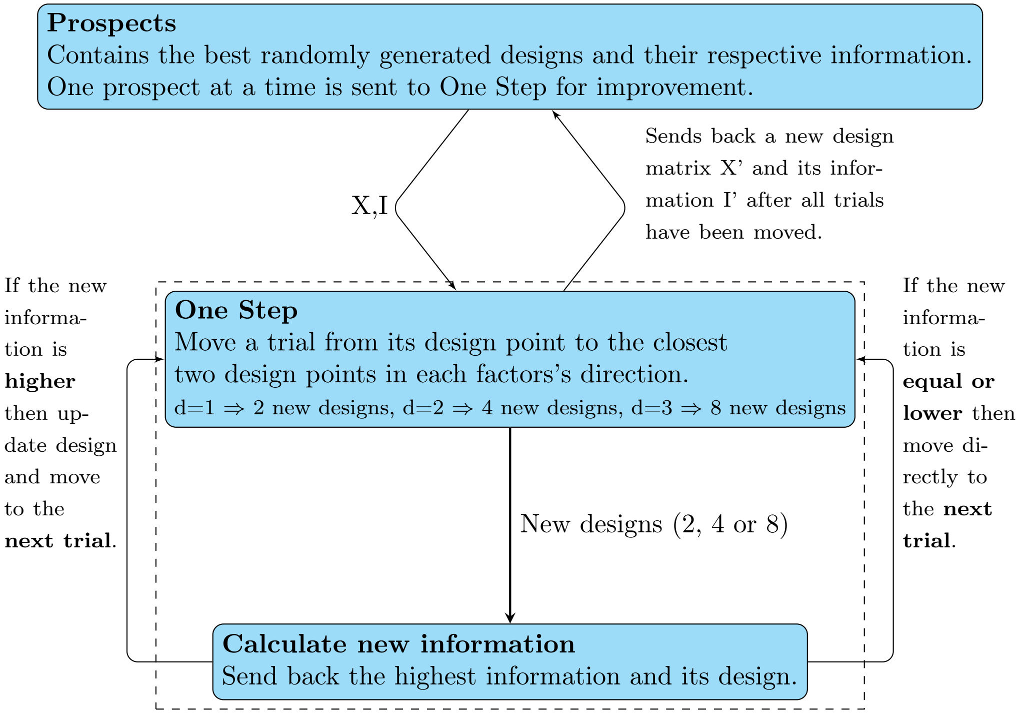


Figure 2: The functions inside the dashed box are iterated as long as the designs improve. This is done until the design have found an information/design maxima, that is the algorithm can't improve the design any more.

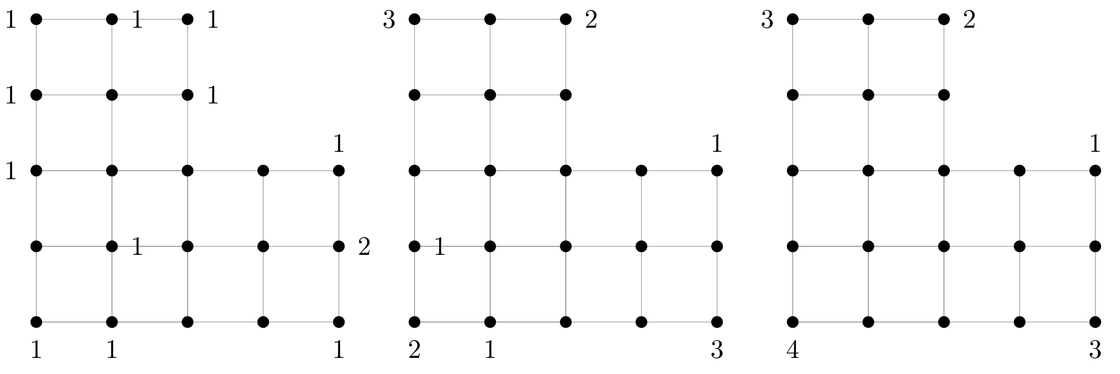


Figure 3: A design that goes through *One Step* and ends up in a sub-optimal design.

In the above figure the first design is a randomly generated design which is among the $\frac{prospects}{simulations}$ highest information designs out of the randomly generated designs. The second design has gone through the *One Step* algorithm once. The third design has gone through *One Step* once more and has ended up in a local maxima that it can't get out from.

Here's how *Systematic Search* works:

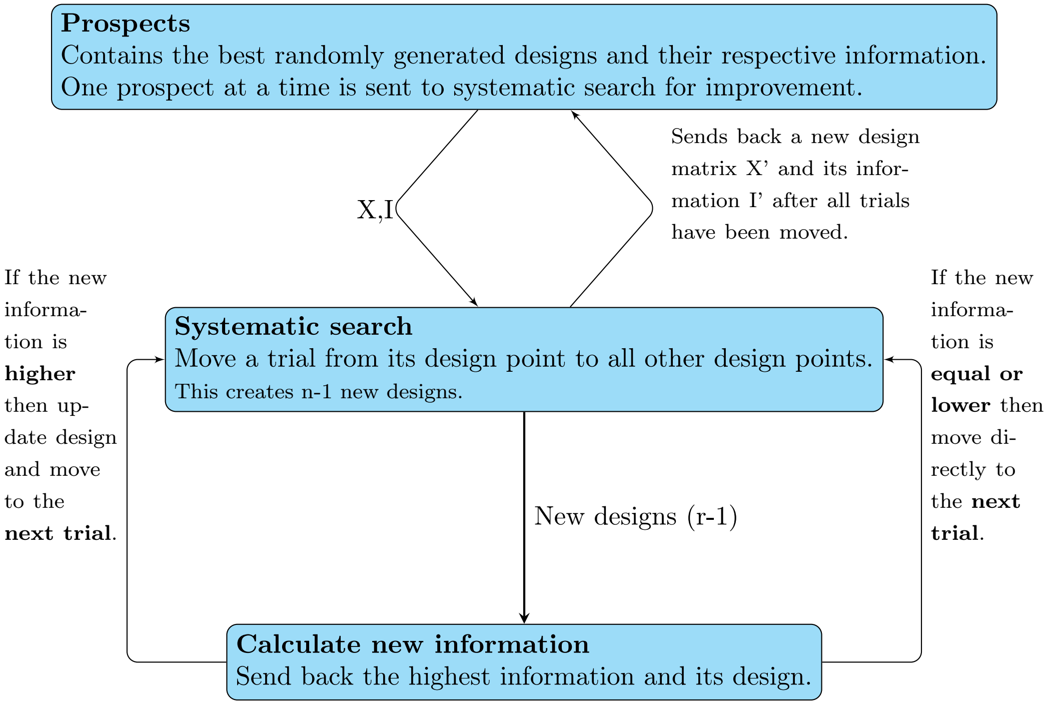


Figure 4: Notice that the *Systematic Search* is not iterated. As long as simulations and prospects are large it only needs one iteration to find a maxima. An advantage over One Step is that it is far less likely to end up in a local maxima.

When *Systematic search* has moved all trials once and when *One Step* has moved all trials until the algorithm can't improve the designs any more, then the program produces output.

Linköping University Electronic Press