Optimal design of experiments

Session 5: Design construction

Peter Goos

Universiteit Antwerpen

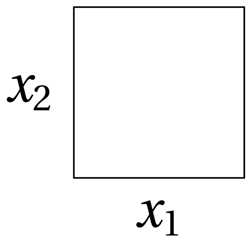
Constructing exact optimal designs

- direct approaches to maximize $|\mathbf{X}^T\mathbf{X}|$
  - Microsoft Excel
  - mathematical programming

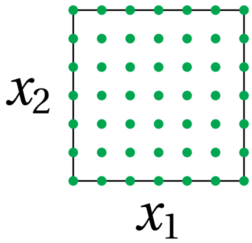
  → run into problems for “large” problems
- rounding off continuous optimal designs
  → performs well for large n
  → only for completely randomized designs (uncorrelated observations)
- design construction algorithms
  → point-exchange algorithms
  → coordinate-exchange algorithms

Point exchange algorithms

- require discretizing the problem
- instead of searching the whole design region χ



- the exchange algorithm searches over a discrete grid of points



→ candidate points (say c)

Fedorov's exchange algorithm

- Fedorov (1972)
- structure

  step 1: select starting design

  step 2: sequentially improve design using exchange procedure

  step 3: stop when no more beneficial exchanges are found

Step 1: Construct starting design

1. select n_1 points at random and compute $(\mathbf{X}^T\mathbf{X})$
2. for all candidate points, calculate prediction variance
$$\mathbf{f}^T(\mathbf{x})(\mathbf{X}^T\mathbf{X}+\mathbf{K})^{-1}\mathbf{f}(\mathbf{x})$$
3. select point with largest prediction variance and add it to the design
why? large prediction variance indicates lack of information
4. compute new $\mathbf{X}^T\mathbf{X}$ and repeat step 2 and 3 until starting design has n points

Step 2: Improve starting design

I. consider all possible exchanges

1. quantify impact of replacing 1st design point by 1st candidate point: $|\mathbf{X}^T\mathbf{X}|_1$
2. quantify impact of replacing 1st design point by 2nd candidate point: $|\mathbf{X}^T\mathbf{X}|_2$
3. ...
4. quantify impact of replacing 1st design point by last candidate point: $|\mathbf{X}^T\mathbf{X}|_c$
5. quantify impact of replacing 2nd design point by 1st candidate point: $|\mathbf{X}^T\mathbf{X}|_{c+1}$
6. ...
7. quantify impact of replacing last design point by last candidate point: $|\mathbf{X}^T\mathbf{X}|_{nc}$

Step 2: Improve starting design

I. consider all possible exchanges
II. save best exchange if it yields an improved $|\mathbf{X}^T\mathbf{X}|$
III. go back to step I if an improvement was found
IV. stop if no improvement was found

- does this guarantee you've found the best design?

no!

- generate other starting designs and go through the algorithm again (perform different tries of the algorithm)

Demonstration of the algorithm I

1. run `blkl.exe` (Atkinson & Donev, 1992), specify input file: `munster1.prn`
   - calculates D-optimal design for $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \epsilon$
   - 9 observations
   - guides you through algorithm step by step
     - → candidate set
     - → starting design
     - → evaluation of all possible exchanges

     (type `integer` and press enter each time algorithm stops)
   - output: `output.prn`

munster1.prn

```
2 number of variables
0 no mixture variables
1 no blocks
9 number of observations
2 order of model
6 number of parameters
0       0       intercept
1       0       linear term variable 1
2       0       idem variable 2
1       2       interaction between variables
1       1       quadratic term variable 1
2       2       idem variable 2
9  k should be at most number of observations
9  l should be at most number of candidates
3  number of times you try
0  technical stuff
1
1
```


Demonstration of the algorithm II

2. run `blklbis.exe`,
   specify input file: `munster2.prn`
   (same algorithm but doesn't stop)
   - compare outcome to optimal continuous design
3. run `blklbis.exe`,
   specify input file: `munster3.prn`
   - user-specified candidate set
   - change number of observations to 6
4. run `blklbis.exe`,
   specify input file: `munster4.prn`
   - first-order model
   - only two levels per factor
5. what would you do if
   - quadratic model in 2 variables,
   - levels between 1 and 3,
   - $x_1 + x_2 \leq 5$?

munster2.prn

```
2 number of variables
0 no mixture variables
1 no blocks
100 number of observations
2 order of model
6 number of parameters
0       0       intercept
1       0       linear term variable 1
2       0       idem variable 2
1       2       interaction between variables
1       1       quadratic term variable 1
2       2       idem variable 2
100  k should be at most number of observations
9  l should be at most number of candidates
200  number of times you try
0  technical stuff
1
1
```


munster3.prn

```
       2 number of variables
       0 no mixture variables
       1 no blocks
       9 number of observations
       2 order of model
       6 number of parameters
       0       0       intercept
       1       0       linear term variable 1
       2       0       idem variable 2
       1       2       interaction between variables
       1       1       quadratic term variable 1
       2       2       idem variable 2
       6  k should be at most number of observations
       6  l should be at most number of candidates
       3  number of times you try
       0  technical stuff
       441
-1.0 -1.0
-0.9 -1.0
...
```

munster4.prn

```
2 number of variables
0 no mixture variables
1 no blocks
8 number of observations
2 order of model
4 number of parameters
0       0       intercept
1       0       linear term variable 1
2       0       idem variable 2
1       2       interaction between variables
2  k should be at most number of observations
2  l should be at most number of candidates
3  number of times you try
0  technical stuff
1
1
```


Demonstration of the algorithm III

6. qualitative experimental variables
   - x_1 = max. speed car
   - x_2 = gas usage
   - $x_3 = \text{brand} \begin{cases} \text{BMW} & d_1 = 1, \quad d_2 = 0 \\ \text{Mercedes} & d_1 = 0, \quad d_2 = 1 \end{cases}$
   - y = willingness to pay
   - model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_3 d_1 + \underbrace{\beta_4 d_2}_{\text{drop from model}}$$

   - input file: munster5.prn
   - what do you observe when looking at optimal design?

munster5.prn

```
3 number of variables
0 no mixture variables
1 no blocks
10 number of observations
2 order of model
7 number of parameters
0       0       intercept
1       0       linear term variable 1
2       0       idem variable 2
1       2       interaction between variables
1       1       quadratic term variable 1
2       2       idem variable 2
3       0       dummy variable
9  k should be at most number of observations
9  l should be at most number of candidates
3  number of times you try
0  technical stuff
```


munster5.prn (continued)

```
18
-1. -1. 1
0. -1. 1
1. -1. 1
-1. 0. 1
0. 0. 1
1. 0. 1
-1. 1. 1
0. 1. 1
1. 1. 1
-1. -1. 0
0. -1. 0
1. -1. 0
-1. 0. 0
0. 0. 0
1. 0. 0
-1. 1. 0
0. 1. 0
1. 1. 0
1
```

Design construction in SAS

`proc factex`

- for constructing candidate set

`proc optex`

- for computing optimal designs

1. example: `simple1.sas`
2. example: `simple2.sas`
   constrained design region
3. example: `simple3.sas`
   qualitative variable

simple1.sas

```
proc factex;
factors x1 x2 / nlev = 3;
output out=can
   x1 nvals = (-1 0 1)
   x2 nvals = (-1 0 1);
proc print data=can;
run;
proc optex data=can seed=57922;
model x1 x2 x1*x2 x1*x1 x2*x2;
generate n=100 method=fedorov;
output out=des;
proc print data=des;
run;
```

simple2.sas

```
proc factex;
factors x1 x2 / nlev = 3;
output out=can
   x1 nvals = (1 2 3)
   x2 nvals = (1 2 3);
data can;
set can;
if x1 + x2 <= 5;
proc print data=can;
run;
proc optex data=can seed=57922;
model x1 x2 x1*x2 x1*x1 x2*x2;
generate n=9 method=fedorov;
output out=des;
proc print data=des;
run;
```


simple3.sas

```
proc factex;
factors x1 x2 / nlev = 3;
output out=intermediate
   x1 nvals = (-1 0 1)
   x2 nvals = (-1 0 1);
run;
factors x3 / nlev = 2;
output out=can designrep=intermediate;
proc print data=can;
run;
proc optex data=can seed=57922;
class x3;
model x1 x2 x1*x2 x1*x1 x2*x2 x3;
generate n=10 criterion=d method=fedorov;
examine information variance;
output out=des;
proc print data=des;
run;
```

Some details about `proc optex`

- examine information $\mathbf{X}^T\mathbf{X}$ and variance $(\mathbf{X}^T\mathbf{X})^{-1}$
- criterion = D; other options: A, U, S, …
- method = Fedorov; other options: M_FEDOROV (modified Fedorov)

remark about D-efficiency reported by SAS

- $\dfrac{|\mathbf{X}^T\mathbf{X}|^{1/p}}{n}$
- see `simple4.sas`
  according to D-efficiency, design with 9 observations is worse than one with 8

simple4.sas

```
proc factex;
factors x1 x2 / nlev = 3;
output out=can
   x1 nvals = (-1 0 1)
   x2 nvals = (-1 0 1);
proc print data=can;
run;
proc optex data=can seed=57922;
model x1 x2 x1*x2;
generate n=8 method=fedorov;
examine information variance;
output out=des;
proc print data=des;
run;
proc optex data=can seed=57922;
model x1 x2 x1*x2;
generate n=9 method=fedorov;
examine information variance;
output out=des;
proc print data=des;
run;
```

Set of candidate points

- should cover entire design region
- best designs are found when vertices and edge centroids are included in candidate set
- how many interior points?
  - the more the better?
  - 2 levels are enough for linear models with/without interactions
  - 3 levels are enough when there are quadratic terms
  - cubic terms: 4 levels
- constructing set of candidates may be difficult when the design region is constrained

Set of candidate points

- constrained design region
  - vertices
  - edge centroids
  - pairwise averages
- spherical design regions
  candidate set should include points on the sphere's surface + center point

Update formulas

- add point **a** to the design
  - information matrix
    $(\mathbf{X}^T\mathbf{X})_{\text{NEW}} = (\mathbf{X}^T\mathbf{X})_{\text{OLD}} + \mathbf{f}(\mathbf{a})\mathbf{f}^T(\mathbf{a})$
  - determinant
    $|\mathbf{X}^T\mathbf{X}|_{\text{NEW}} = |\mathbf{X}^T\mathbf{X}|_{\text{OLD}} \left(1 + \underbrace{\mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})}_{\text{prediction variance}}\right)$
    (explains why point with largest prediction variance has to be added)
  - variance-covariance matrix
    $$(\mathbf{X}^T\mathbf{X})^{-1}_{\text{NEW}} = (\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}} - \frac{(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})\mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}}{1 + \mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})}$$
- saves computing time

Update formulas

- delete point **a** from the design
  - information matrix
    $(\mathbf{X}^T\mathbf{X})_{\text{NEW}} = (\mathbf{X}^T\mathbf{X})_{\text{OLD}} - \mathbf{f}(\mathbf{a})\mathbf{f}^T(\mathbf{a})$
  - determinant
    $|\mathbf{X}^T\mathbf{X}|_{\text{NEW}} = |\mathbf{X}^T\mathbf{X}|_{\text{OLD}} \left(1 - \underbrace{\mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})}_{\text{prediction variance}}\right)$
    (explains why point with smallest prediction variance has to be deleted)
  - variance-covariance matrix
    $$(\mathbf{X}^T\mathbf{X})^{-1}_{\text{NEW}} = (\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}} + \frac{(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})\mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}}{1 - \mathbf{f}^T(\mathbf{a})(\mathbf{X}^T\mathbf{X})^{-1}_{\text{OLD}}\mathbf{f}(\mathbf{a})}$$
- there are update formulas for the replacement of a point **a** by a point **b** too

KL exchange algorithm

- Atkinson & Donev (1989)
- intended to speed up Fedorov's algorithms
- does not evaluate all possible exchanges of design points and candidate points
- only K design points are considered for removal from the design (namely the ones with the smallest prediction variances)
- only L candidate points are considered for entry in the design (namely the ones with the largest prediction variances)
- inspired by update formula for determinant after exchanging a design point with a candidate point

Modified Fedorov algorithm

I. consider all exchanges for 1st design point
    - quantify impact of replacing 1st design point by 1st candidate point
    - quantify impact of replacing 1st design point by 2nd candidate point
    - ...
    - quantify impact of replacing 1st design point by last candidate point
    - save best exchange

Modified Fedorov algorithm

II. consider all exchanges for 2nd design point
   - quantify impact of replacing 2nd design point by 1st candidate point
   - quantify impact of replacing 2nd design point by 2nd candidate point
   - ...
   - quantify impact of replacing 2nd design point by last candidate point
   - save best exchange

III. ...

Modified Fedorov algorithm

IV. consider all exchanges for *n*th design point
   - ...

V. go back to I if one of the steps I to IV yielded improvement and led to change

VI. stop if no improvement found

Coordinate-exchange algorithm

- Meyer & Nachtsheim (1995)
- starting design is generated randomly
- improve starting design
  1. replace first coordinate of first design point → save if improvement
  2. replace second coordinate of first design point → save if improvement
  3. ...
  4. replace last coordinate of last design point → save if improvement
  5. if at least one improvement was found, go back to step 1
- this algorithm is faster, but not necessarily better

JMP

- uses coordinate exchange algorithm
- does not always find equally good designs but you can do more tries for a given amount of computing time
- offers the advantage that it is not required to construct a candidate set
- this may be important when
  - the design region is highly constrained
  - there are an awful lot of experimental variables
  - the candidate set would become too large (because then the classical algorithms have problems finding the globally optimal design)
- custom design instead of optimal design

Other algorithms

- genetic algorithms
- simulated annealing
- variable neighbourhood search
- tabu search
- ant colony optimisation
- each of these algorithms need substantial tuning
- what algorithm you use is a matter of taste