A new algorithm for deriving optimal designs

Stefanie Biedermann, University of Southampton, UK

Joint work with
Min Yang, University of Illinois at Chicago

18 October 2012, DAE, University of Georgia, Athens, GA

Outline

Statistics and Optimal design of experiments

- The main goal of statistics is to gain information from collected data

- Design of experiments concerns the way of data collection

  $\hookrightarrow$ An optimal/efficient design can reduce the sample size and cost needed for achieving a specified precision of estimation, or can improve the precision for a given sample size

Exact and approximate designs

- Exact design ξ_n for sample size n:

$$\xi_n = \left\{ \begin{array}{cccc} x_1 & x_2 & \dots & x_m \\ n_1/n & n_2/n & \dots & n_m/n \end{array} \right\};\ n_i \text{ integers},\ \sum_{i=1}^{m} n_i = n$$

- Approximate design ξ:

$$\xi = \left\{ \begin{array}{cccc} x_1 & x_2 & \dots & x_m \\ \omega_1 & \omega_2 & \dots & \omega_m \end{array} \right\};\ \omega_i > 0,\ \sum_{i=1}^{m} \omega_i = 1$$

Note: To run an approximate design, some rounding procedure is required

Optimality criteria

- Suppose interest is in $F(\theta)$, a vector of functions of the model parameters
- The (asymptotic) covariance matrix of the MLE of $F(\theta)$ under design ξ, $\Sigma_\xi(F)$, can be written as

$$\Sigma_\xi(F) = \frac{\partial F(\theta)}{\partial \theta^T}(I_\xi)^- \frac{\partial F(\theta)}{\partial \theta}$$

  where I_ξ is the information matrix of the design ξ
- Aim: Obtain precise estimates
  $\hookrightarrow$ Minimise $\Phi(\Sigma_\xi(F))$, an optimality criterion

Optimality criteria

- Example 1: A *D*-optimal design minimises the determinant of the covariance matrix
  $\hookrightarrow$ minimises the volume of a confidence ellipsoid for $F(\theta)$
- Example 2: An *A*-optimal design minimises the trace of the covariance matrix
  $\hookrightarrow$ minimises the sum of the variances of the elements of $F(\hat{\theta})$

Locally optimal designs

- A large and popular class of optimality criteria are the Φ_p-criteria, of which *A*- and *D*-optimality are special cases:

$$\Phi_p(\Sigma_\xi(F)) = \left[\frac{1}{v}\text{trace}(\Sigma_\xi(F))^p\right]^{1/p}$$

where $0 \leq p < \infty$ and $\Sigma_\xi(F) \in R^{v \times v}$

- Problem: In many situations, Φ_p-optimal designs depend on the unknown parameter vector θ
  $\hookrightarrow$ *locally* optimal designs

Optimal multi-stage designs

Use a sequential strategy:

- Suppose n_0 observations are already available, from design ξ_{n_0}, and another n observations can be made
- Estimate $\theta \hookrightarrow \hat{\theta}_0$
- Design problem: Find ξ such that the combined design $\tilde{\xi} = n_0/(n_0+n)\xi_{n_0} + n/(n_0+n)\xi$ minimises $\Phi(\Sigma_{\tilde{\xi}}(F))$ for $\hat{\theta}_0$
- Note: $\tilde{\xi}$ and thus ξ are approximate designs

Aim of this project

Provide a fast, general, convenient, and easy to use approach to find locally and multi-stage optimal designs

Outline

Current algorithms

Three well-known algorithms are:

- Vertex direction method (VDM; Fedorov, 1972)
- Multiplicative algorithm (MA; Silvey, Titterington and Torsney, 1978)
- Vertex exchange method (VEM; Böhning, 1986)

Yu (2011) combines "the best features" of these algorithms with a nearest neighbour exchange step to improve speed
↪ Cocktail algorithm

Computing time comparison I

Model: $Y = \theta_1 e^{-\theta_2 x} + \theta_3 e^{-\theta_4 x} + \varepsilon$

Design space $[0, 3]$ discretised to $\mathcal{X} = \{3i/N, i = 1, \ldots, N\}$

Computing time (in seconds) (Table 1 of Yu, 2011)

	$N = 20$	$N = 50$	$N = 100$	$N = 200$	$N = 500$
MA	14.3	63.7	147	307	762
VEM	0.17	1.43	23.1	206	555
Cocktail	0.07	0.11	0.25	0.36	0.96

(VDM is omitted since it is known to be slower than VEM)

Computing time comparison II

$$Y = \theta_1 + \theta_2 r + \theta_3 r^2 + \theta_4 s + \theta_5 rs + \varepsilon$$

Design space $[-1, 1] \times [0, 1]$ discretised to
$\mathcal{X} = \{(r_i, s_j), r_i = 2i/k - 1, i = 1, \ldots, k, s_j = j/k, j = 1, \ldots, k\}$, $N = k^2$

Computing time (in seconds) (Table 4 of Yu, 2011)

	$N = 20^2$	$N = 50^2$	$N = 100^2$	$N = 200^2$
MA	25.2	993.8	aborted	aborted
VEM	8.01	195.8	94.6	aborted
Cocktail	0.63	3.94	17.6	74.1

Summary of current algorithms

- All algorithms are relatively easy to implement
- The Cocktail algorithm leads to dramatically improved speed, sometimes by several orders of magnitude, relative to MA, VDM, and VEM.
- The Cocktail algorithm is restricted to *D*-optimal design for the full parameter vector
- Restricted to (one-stage) locally optimal designs

Outline

Goal of the new algorithm

- Any parameters (or functions thereof) can be of interest
- Any optimality criterion: Φ_p-optimality (includes D-, A-, and E-optimality)
- One-stage and multi-stage
- Convenient to use
- Even faster

Rationale

1. The existing algorithms typically seek “monotonic convergence”
↪ May not be the fastest way to converge to the optimal weights

2. New idea: instead of seeking monotonic convergence, we derive the optimal weights directly, which can be done by solving a system of nonlinear equations

Notations for the new algorithm

- $\mathcal{X} = \{x_1, \ldots, x_N\}$, the set of all possible design points
- $S^{(t)}$, the support after tth iteration
- $\xi^*_{S^{(t)}}$, design with support $S^{(t)}$ and *optimal* weights
- $d(x, \xi^*_{S^{(t)}})$, sensitivity function of design $\xi^*_{S^{(t)}}$

The new algorithm

Start with an initial design support, $S^{(0)}$. In the tth step:

- Derive $\xi^*_{S^{(t)}}$ using the Newton-Raphson method
- Compute the sensitivity function $d(x_i, \xi^*_{S^{(t)}})$ for $i = 1, \ldots, N$
- Select $1 \leq i_{max} \leq N$ such that

$$d(x_{i_{max}}, \xi^*_{S^{(t)}}) = \max_{i \leq n \leq N} d(x_i, \xi^*_{S^{(t)}})$$

- Stop when $d(x_{i_{max}}, \xi^*_{S^{(t)}}) \leq \epsilon$, $\epsilon > 0$ very small, otherwise
- $S^{(t+1)} = S^{(t)} \bigcup \{x_{i_{max}}\}$

Convergence (I)

Theorem 1: For a multi-stage design, let I_{ξ_0} be non-singular.

Then for any initial support $S^{(0)}$, the sequence of designs $\{\xi^*_{S^{(t)}}; t \geq 0\}$ converges to an optimal design minimising $\Phi_p(\Sigma_\xi(F))$ as $t \to \infty$, provided the optimal weights are found in each updating step

For a one-stage design, the above result holds if $I_{\xi_{S^{(0)}}}$ is non-singular, and if the Jacobian of F is square and of full rank

Convergence (II)

Theorem 2: For a multi-stage design, let p be an integer and I_{ξ_0} be non-singular. For given support, $\Phi_p(\Sigma_\xi(F))$ is minimised (w.r.t the weights) at any critical point in
$\Omega = \{w = (w_1, \ldots, w_{m-1})^T : \ w_i \geq 0, \ w_1 + \ldots + w_{m-1} \leq 1\}$ or its boundary. In addition, the Hessian of $\Phi_p(\Sigma_\xi(F))$ is non-negative definite.

$\hookrightarrow$ Newton-Raphson method will converge provided the starting point is "close enough" to the critical point. Since Ω is compact, we can always find a critical point (given it exists) by using sufficiently many starting values

Outline

Set up

Consider model $Y = \theta_1 e^{\theta_2 x} + \theta_3 e^{\theta_4 x} + \varepsilon$. Let $\theta = (\theta_1, \theta_2, \theta_3, \theta_4)$ and $\mathcal{X} = \{3i/N, i = 1, \ldots, N\}$. Assume that $\theta = (1, -1, 1, -2)$.

The same set up as that of Table 1 of Yu (2011)

Algorithms are coded using SAS IML and computed at a Dell laptop (2.2GHz and 8Gb Ram)

Computing time (in seconds) comparison I

Computing time (in seconds) for the D-optimal design for the full parameter vector θ

	$N = 500$	$N = 1000$	$N = 5000$	$N = 10000$
Cocktail	0.32	0.46	2.54	5.16
New algorithm	0.14	0.21	0.99	1.26

Locally A- and D-optimal designs

Computing time (in seconds) under the new algorithm

	D			A		
	θ	(θ_1, θ_3)	(θ_2, θ_4)	θ	(θ_1, θ_3)	(θ_2, θ_4)
$N = 500$	0.14	0.10	0.10	0.10	0.10	0.10
$N = 1000$	0.21	0.12	0.15	0.11	0.12	0.12
$N = 5000$	0.99	0.32	0.46	0.24	0.28	0.23
$N = 10000$	1.26	0.54	0.85	0.45	0.42	0.45

Multi-stage *A*- and *D*-optimal designs

Suppose we have an initial design
$\xi_0 = \{(0, 0.25), (1, 0.25), (2, 0.25), (3, 0.25)\}$ with $n_0 = 40$

Aim: Allocate the next 80 observations

Computing time (in seconds) under the new algorithm

	D			*A*		
	θ	(θ_1, θ_3)	(θ_2, θ_4)	θ	(θ_1, θ_3)	(θ_2, θ_4)
$N = 500$	0.36	0.34	0.32	0.09	0.09	0.10
$N = 1000$	0.42	0.37	0.37	0.10	0.09	0.11
$N = 5000$	0.78	0.57	0.67	0.34	0.29	0.23
$N = 10000$	1.27	0.78	0.98	0.54	0.57	0.40

Set up

Consider model $Y = \theta_1 + \theta_2 r + \theta_3 r^2 + \theta_4 s + \theta_5 rs + \varepsilon$.
Let $\theta = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ and
$\mathcal{X} = \{(r_i, s_j), r_i = 2i/k - 1, i = 1, \ldots, k, s_j = j/k, j = 1, \ldots, k\}$.

The same set up as that of table 4 of Yu (2011)

Algorithms are coded using SAS IML and computed at a Dell laptop (2.2GHz and 8Gb Ram)

Computing time (in seconds) comparison II

D-optimal design for the full parameter vector θ

	$N = 20^2$	$N = 50^2$	$N = 100^2$	$N = 200^2$	$N = 500^2$
Cocktail	0.20	0.82	2.30	8.68	53.69
new algorithm	0.15	0.24	0.51	1.66	11.03

Set up

Consider multinomial logit model with 3 categories, and 3 variables:

Response $\mathbf{Y} = (Y_1, Y_2, Y_3)^T$ at experimental condition $\mathbf{x}$ with $Y_1 + Y_2 + Y_3 = 1$ and

$$\pi_i(\mathbf{x}) = P(Y_i = 1|\mathbf{x}) = \frac{e^{g(\mathbf{x})^T\theta_i}}{1 + e^{g(\mathbf{x})^T\theta_1} + e^{g(\mathbf{x})^T\theta_2}}, \quad i = 1, 2.$$

where $g(\mathbf{x})^T = (1, x_1, x_2, x_3)$

Modification to algorithm (example III only)

Design space $\mathcal{X} = \{(6i/k, 6j/k, 6l/k), i, j, l = 0, 1, \ldots, k\}$
$\hookrightarrow$ The number of design points in $\mathcal{X}$ rapidly increases as k increases ($N \propto k^3$)

- Start with a coarse grid
- Identify the best design based on the grid at that stage
- For the next stage, a finer grid is restricted to neighbourhoods of the best support points found at the current stage
- Continue until a specified accuracy for the design points is reached
- Verify optimality through the equivalence theorem

Computing time (in seconds) example III

	Locally optimal design				Multi-stage optimal design			
	θ		θ'		θ		θ'	
N	D	A	D	A	D	A	D	A
10^3	0.32	0.32	0.26	0.39	0.29	0.31	0.18	0.23
20^3	0.62	1.07	1.15	1.71	0.74	1.73	0.65	1.34
50^3	8.14	17.81	17.94	24.52	12.85	16.98	11.29	19.57
100^3	54.38	86.92	101.13	169.57	71.73	68.14	71.09	114.64
200^3	524.1	653.0	664.4	814.3	531.8	738.7	718.2	853.8

$\theta^T = (\theta_1^T, \theta_2^T)$ where $\theta_1 = (1, 1, -1, 2)$ and $\theta_2 = (-1, 2, 1, -1)$.
$\theta' = (\theta_{11}, \theta_{12}, \theta_{13}, \theta_{21}, \theta_{22}, \theta_{23})$

Outline

Summary

- The new algorithm is even faster than the Cocktail algorithm
- More importantly, the new algorithm is more widely applicable:
  - Φ_p-optimality
  - Single- and Multi-stage designs
  - interest in functions of the parameters
- Future work:
  - Extend algorithm to find Bayesian and (hopefully) Minimax designs
  - Compare its performance with further algorithms, e.g PSO

Thank You!

Selected references

Böhning, D. (1986), "A vertex-exchange-method in *D*-optimal design theory", *Metrika*, 33, 337-347.

Fedorov, V.V. (1972). *Theory of optimal experiments (transl. and ed. by Studden WJ, Klimko EM)*, New York: Academic Press.

Silvey, S.D., Titterington, D.M., and Torsney, B. (1978), "An algorithm for optimal designs on a finite design space", *Communications in Statistics - Theory and Methods*, 14, 1379-1389.

Yang, M. and Biedermann, S. (2012). "On optimal designs for nonlinear models: a general and efficient algorithm," *Under revision*.

Yu, Y. (2011), "*D*-optimal designs via a cocktail algorithm", *Statistics and Computing*, 21, 475-481.

VDM

Starting with an initial design $\xi^{(0)}$ (uniform design over a set of support points). Let $\xi^{(t)}$ be the design with the corresponding weight vector $\omega^{(t)}$. $\xi^{(t+1)}$ is updated through the following steps.

- Select $1 \leq i_{max} \leq N$ such that $d(i_{max}, \xi^{(t)}) = \max_{i \leq n \leq N} d(i, \xi^{(t)})$
- Stop when $d(i_{max}, \xi^{(t)}) \leq m + \epsilon$, otherwise
- $\omega_i^{(t+1)} = \begin{cases} (1-\delta)\omega_i^{(t)}, & i \neq i_{max} \\ (1-\delta)\omega_i^{(t)} + \delta, & i = i_{max} \end{cases}$ where $\delta = \frac{d(i_{max}, \xi^{(t)})/m - 1}{d(i_{max}, \xi^{(t)}) - 1}$.

MA

Starting with an initial design $\xi^{(0)}$ (uniform design over a set of support points). Let $\xi^{(t)}$ be the design with the corresponding weight vector $\omega^{(t)}$. $\xi^{(t+1)}$ is updated through the following steps.

- Stop when $d(i_{max}, \xi^{(t)}) \leq m + \epsilon$, otherwise
- $\omega_i^{(t+1)} = \omega_i^{(t)} d(i, \xi^{(t)})/m$, where m is the dimension of I_ξ.

VEM

VEM (vertex exchange method) is close to VDM but performs better.

- Select i_{min} and i_{max} such that
  $d(i_{min}, \xi^{(t)}) = \min\{d(i, \xi^{(t)}) : w_i^{(t)} > 0\}$, and
  $d(i_{max}, \xi^{(t)}) = \max_{i \le n \le N} d(i, \xi^{(t)})$
- Stop when $d(i_{max}, \xi^{(t)}) \le m + \epsilon$, otherwise
- Set $\omega^{(t+1)} = VE(i_{max}, i_{min}, \omega^{(t)})$.

Vertex exchanges (VE)

$VE(j,k,\xi^{(t)})$ is to exchange the weight between x_j and x_k, where

$$\omega_i^{(t+1)} = \begin{cases} \omega_i^{(t)}, & i \neq j,k \\ \omega_i^{(t)} - \delta, & i = j \\ \omega_i^{(t)} + \delta, & i = k \end{cases} \quad \text{Here } \delta = \min\{\omega_j, \max\{-\omega_k, \delta^*(j,k)\}\}$$

and

$$\delta^*(j,k) = \frac{d(k,\xi^{(t)}) - d(j,\xi^{(t)})}{2\left(d(j,\xi^{(t)}) * d(k,\xi^{(t)}) - \left((f(x_j)^T I_{\xi^{(t)}}^{-1} f(x_k)\right)^2\right)}$$

Nearest neighbour exchanges (NNE)

Let $i_1, \ldots, i_{q+1}$ be the elements of $\{i : \omega_i^{(t)} > 0\}$ where $q+1$ is the number of support points of $\omega^{(t)}$. For each $i_j, j = 1, \ldots, q$, let i_j^* be any index $i \in \{i_{j+1}, \ldots, i_{q+1}\}$ such that $||x_i - x_{i_j}||$ is minimized. Perform VE between x_{i_j} and $x_{i_j^*}$ for $j = 1, \ldots, q$ in turn, i.e.,

$$\omega^{(t+j/q)} = VE(i_j, i_{j^*}, \xi^{(t+(j-1)/q)})$$

Computing optimal weights

Initial weight $\omega_0^{(t)}$ for $S^{(t)}$ can be updated through the following iteration:

(a) $\omega_j^{(t)} = \omega_{j-1}^{(t)} - \alpha \left(\frac{\partial^2 \Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega \omega^T} |_{\omega=\omega_{j-1}^{(t)}} \right)^{-1} \frac{\partial \Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega} |_{\omega=\omega_{j-1}^{(t)}}$

(b) Check if there are non-positive components of $\omega_j^{(t)}$

(c) If answer of (b) is no, check whether $|| \frac{\partial \Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega} |_{\omega=\omega_j^{(t)}} ||$ is less than some pre-specified ϵ_1. If yes, $\omega_j^{(t)}$ holds the optimal weights. If no, start the next iteration.

(d) If answer of (b) is yes, reduce α to $\alpha/2$. Repeat (a) and (b) until α reaches a pre-specified value. Remove the support point with smallest weight, and repeat (a) and (b)

Sensitivity function:

$$d(x_i, \xi^*_{S^{(t)}}) = Tr[\left(I(x_i) - I_{\xi^*_{S^{(t)}}}\right)\left(I_{\xi_0} + rI_{\xi^*_{S^{(t)}}}\right)^{-}\left(\frac{\partial F(\theta)}{\partial \theta^T}\right)^T$$
$$\left(\Sigma_{\xi^*_{S^{(t)}}}\right)^{p-1}\left(\frac{\partial F(\theta)}{\partial \theta^T}\right)\left(I_{\xi_0} + rI_{\xi^*_{S^{(t)}}}\right)^{-}]$$

For D-optimality, $p = 0$.