Optimal Design Foundations

John Sherrill

Spring 2015

1 Motivation

Algorithmic design is primarily concerned with the generation of experimental designs by computational means. Particular design characteristics are usually desired and these characteristics are often made with respect to a given model:

$$\mathbf{y} = X\beta + \epsilon$$
$$\epsilon_i \overset{iid}{\sim} (0, \sigma^2)$$

It's got p parameters. A specific design region, R, is usually specified such that the design properties of X are only considered over $\mathbf{x} \in R$. While a great many characteristics exist (blocking, completeness, balance, orthogonality, confoundedness, etc.) those of present interest concern the variance of predicted values of a model.

Much of what is available in evaluation and comparison of RSM designs as well as computer-generated design were the work of Kiefer (1959, 1961) and Keifer and Wolfowitz (1959) in **optimal design theory**. Their work is couched in a measure theoretic approach in which an experiment is viewed in terms of design measure. Design optimality moved into the practical arena in the 1970s and 1980s as designs were put forth as being efficient in terms of criteria inspired by Keifer and his coworkers. (Montgomery 8.2)

2 Scaled Prediction Variance Function Derivation

Say there are p parameters in the above model. If $\mathbf{b}$ is the OLS estimate of β then the variance of the predicted values[1] is given by

$$\begin{aligned}
\text{Var}[\hat{\mathbf{y}}(\mathbf{x})|X] &= \text{Var}[\mathbf{xb}] \\
&= \text{Var}[x_1 b_1 + x_2 b_2 + \cdots + x_p b_b] \\
&= \sum_{i,j} \text{Cov}(x_i b_i, x_k b_j) \\
&= \sum_{i,j} x_i \text{Cov}(b_i, b_j) x_j \\
&= \mathbf{x}' \text{Var}[\mathbf{b}|X]\mathbf{x} \\
&= \mathbf{x}' E[(b - E(b))(b - E(b))'|X]\mathbf{x} \\
&= \mathbf{x}' E[(b - \beta)(b - \beta)'|X]\mathbf{x}
\end{aligned}$$

Note that $b = (X'X)^{-1}X'\mathbf{y} = (X'X)^{-1}X'(X\beta + \epsilon) = \beta + (X'X)^{-1}X'\epsilon$

$$\begin{aligned}
&= \mathbf{x}' E[(\beta + (X'X)^{-1}X'\epsilon - \beta)(\beta + (X'X)^{-1}X'\epsilon - \beta)'|X]\mathbf{x} \\
&= \mathbf{x}' E[(X'X)^{-1}X'\epsilon)((X'X)^{-1}X'\epsilon)'|X]\mathbf{x} \\
&= \mathbf{x}' E[(X'X)^{-1}X'\epsilon\epsilon'X(X'X)^{-1}|X]\mathbf{x} \\
&= \mathbf{x}'(X'X)^{-1}X'E[\epsilon\epsilon'|X]X(X'X)^{-1}\mathbf{x}
\end{aligned}$$

[1] See `http://en.wikipedia.org/wiki/Mean_and_predicted_response`

Assuming that $\epsilon \overset{iid}{\sim} N(0, \sigma^2)$,

$$\begin{aligned} &= \mathbf{x}'(X'X)^{-1}X'\sigma^2 IX(X'X)^{-1}\mathbf{x} \\ &= \mathbf{x}'\sigma(X'X)^{-1}\mathbf{x} \end{aligned}$$

To be able to make comparisons between designs of different size, the "Scaled Prediction Variance" (SPV) function will be used. All things being equal, designs of larger sizes will have smaller prediction variances and multiplying by the design size provides a measure of prediction variance on a per-observation basis. Dividing by σ^2 makes the SPV scale-free. Thus we have

$$SPV_X(\mathbf{x}) = \frac{N\text{Var}[\hat{\mathbf{y}}(\mathbf{x})]}{\sigma^2} = N\mathbf{x}'(X'X)^{-1}\mathbf{x}$$

Because this function provides a penalty for an increased sample size, cost comparisons between designs can be made (Montgomery, 7.4.4). Also, division by σ^2 means that the SPV function is *not* a function of σ and therefore it isn't nescessary for data to have been collected to estimate $\hat{\sigma} = \frac{SSE}{n-p}$.

3 Criterions

Naturally, a desirable property of a design would be low levels of prediction variance. Therefore, by taking note of the form of the SPV function, we seek to minimize $(X'X)^{-1}$ in some way (assuming the design region and size are fixed). There are several ways, refered to as design criterions, in which this can be achieved that are supported in the `OptimalDesign` package:

3.1 D Criterion

Minimize over all designs X

$$\text{D Criterion} = |(X'X)^{-1}|$$

This minimizes the volume of the confidence hyperellipsoid for parameter estimates. This is because $|(X'X)^{-1}|$ is proportional to the square of the volume of the confidence hyperellipsoid.

3.2 A Criterion

Minimize over all designs X

$$\text{A Criterion} = \text{tr}\big((X'X)^{-1}\big)$$

Minimizing the A Criterion results in minimizing the average variance of the estimates of the regression coefficients.

3.3 I Criterion

Minimize over all designs X

$$\text{I Criterion} = \int_{\mathbf{x}\in R} SPV_X(\mathbf{x})\, d\mathbf{x}$$

Minimizing the I criterion results in minimizing the average prediction variance over the entire design region R.

3.4 G Criterion

Minimize over all designs X

$$\text{G Criterion} = \max_{\mathbf{x}\in R} SPV_X(\mathbf{x})$$

That is, minimize the maximum prediction variance over the entire design region R.

4 Search Algorithms

There are several search algorithms for finding optimal designs. Examples include the Sequential Search, Exchange, DETMAX, and Fedorov algorithms which are the searches supported by the OPTEX procedure in SAS. Currently only a modifiable Fedorov algorithm is supported in `OptimalDesign`.

4.1 Fedorov Algorithm (Deluxe Monte Carlo Edition)

The Fedorov Algorithm takes an initial design and then randomly chooses a design point to exchange. The design point chosen to be added is the design point over the design region, R, that results in the largest decrease in the design criterion. `OptimalDesign` does not use the full Fedorov Algorithm as it is fairly slow. To speed things up, `OptimalDesign` selects a random point in the design region to potentially add to the design. If this exchange results in the lower criterion, then the exchange is implemented and another, Criterion-decerasing exchange is searched for. This algorithm is essentially the 'Monte Carlo' algorithm used in the R package `AlgDesign`.

To compute the D, A, I, and G criterions directly, a matrix inversion is required. There are, however, more efficient means of calculating the change to the criterions, given a particular exchange. The methods for the D, A, and I criterions are from identities provided in the `AlgDesign` documentation (Wheeler, 2009).

4.1.1 D Criterion

To more efficiently calculate the D criterion, ther Woodbury Matrix Identity can be used: let A be a p-by-p matrix, U is p-by-n, C is n-by-n, V is n-by-p. Then

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Let Y be the model matrix after the replacement of design point $\mathbf{x}$ with candidate point $\mathbf{y}$. $(Y'Y)^{-1}$ can be computed in a computationally simple manner by utilization of the above identity. If $(X'X)^{-1}$ is known,

$$\begin{aligned}(Y'Y)^{-1} &= \big((X'X) + [\mathbf{x},\mathbf{y}]I_2[-\mathbf{x},\mathbf{y}]'\big)^{-1} \\ &= (X'X)^{-1} - (X'X)^{-1}[\mathbf{x},\mathbf{y}](I_2 + [-\mathbf{x},\mathbf{y}]'(X'X)^{-1}[\mathbf{x},\mathbf{y}])^{-1}[-\mathbf{x},\mathbf{y}]'(X'X)^{-1}\end{aligned}$$

and the change in the D criterion is

$$\begin{aligned}\Delta_D &= (X'X)^{-1} - (Y'Y)^{-1} \\ &= (X'X)^{-1}[\mathbf{x},\mathbf{y}](I_2 + [-\mathbf{x},\mathbf{y}]'(X'X)^{-1}[\mathbf{x},\mathbf{y}])^{-1}[-\mathbf{x},\mathbf{y}]'(X'X)^{-1}\end{aligned}$$

Let $d_{uv} = u'(X'X)^{-1}$. It can be shown that

$$\Delta_D = d_{\mathbf{yy}} - d_{\mathbf{yy}} \cdot d_{\mathbf{xx}} + d_{\mathbf{yx}}^2 - d_{\mathbf{xx}}$$

which is a cheap computation.

4.1.2 A Criterion

It should be noted that the trace of a matrix is linear. Thus, for a design X, the A criterion is a linear functional of the dispersion matrix $(X'X)^{-1}$:

$$\operatorname{tr}\big((Y'Y)^{-1}\big) = \operatorname{tr}\big((X'X)^{-1}\big) - \operatorname{tr}\big((X'X)^{-1}[\mathbf{x},\mathbf{y}](I_2 + [-\mathbf{x},\mathbf{y}]'(X'X)^{-1}[\mathbf{x},\mathbf{y}])^{-1}[-\mathbf{x},\mathbf{y}]'(X'X)^{-1}\big)$$

and, similarly to above for the D criterion, the change in the A criterion is

$$\begin{aligned}\Delta_A &= \mathrm{tr}\big((X'X)^{-1}\big) - \mathrm{tr}\big((Y'Y)^{-1}\big) \\ &= \mathrm{tr}\big((X'X)^{-1}[\mathbf{x},\mathbf{y}](I_2 + [-\mathbf{x},\mathbf{y}]'(X'X)^{-1}[\mathbf{x},\mathbf{y}])^{-1}[-\mathbf{x},\mathbf{y}]'(X'X)^{-1}\big)\end{aligned}$$

It can be shown (laboriously) that if $\phi^A_{uv} = \mathrm{tr}\big((X'X)^{-1}uv'(X'X)^{-1}\big)$ then

$$\Delta_A = \frac{(1-d_{xx})\phi^A_{xx} + d_{yx}(\phi^A_{xy} + \phi^A_{yx}) - (1+d_{yy})\phi^A_{xx}}{1+\Delta_D}$$

4.1.3 I Criterion

Because integration is a linear functional there is a similar identity to the A criterion identity show just above. Let X_c denote the matrix containing the N set of candidate points approximating the design space R. If

$$B = \frac{1}{N}X_c'X_c, \qquad \text{and} \qquad \phi^I_{uv} = v'(X'X)^{-1}B(X'X)^{-1}u,$$

then it can be show (laboriously) that,

$$\Delta_I = \frac{(1-d_{xx})\phi^I_{xx} + d_{yx}(\phi^I_{xy} + \phi^I_{yx}) - (1+d_{yy})\phi^I_{xx}}{1+\Delta_D}$$

4.1.4 G Criterion

The G criterion is the most difficult to calculate, currently. This is because it requires a calculation of the SPV over the entire set of candidate points. To illustrate the difficulty, consider a search for a 5 factor, linear design (with no interaction). If for each factor we consider 21 different levels (-1 to 1 at .1 increments), then the candidate space contains $21^5 = 4084101$ points. A each interaction in the Fedorov search a matrix multiplication involving a matrix of this size will be required which is not cool.

The G criterion is relatively unstable as a function of different designs. That is, a small change in the design can lead to large changes in the G criterion. This implies that large candidate sets are required to find G-optimal designs, candidate sets so large that it is not practical to search for G-optimal designs in this manner (777, SAS OnlineDoc: Version 8).

The best that one can do is:

Let X denote a potentially new design matrix. Let X_c denote the candidate set matrix. Let X have singular value decomposition (SVD) $X = UDV'$ and X_c have SVD $X_c = U_cD_cV_c'$. We have

$$SVD_{thing} = U_cVD^{-1}$$

The 'leverage' of each point in the candidate space is the corresponding row-norm (Frobenius norm) in of the SVD_{thing} matrix. These leverages are the SPV value at each respective point. The maximum of these leverages are found and this is the G criterion of the potentially new design. If the new design has a lower G criterion, then the old design is exchanged for the new.

While this is more efficient than computing the SVD for each and every candidate point, it's not that much more efficient because the real bottle neck is the size of the candidate set which is not reduced by this technique.

4.2 Genetic Algorithm