D-Optimal Design for Nested Sensor Placement

David Sudell, MSci. (Hons) MSc.

School of Mathematical Sciences

Lancaster University

This thesis is submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

March 2024

Abstract

This thesis introduces a new method for the construction of D-optimal designs with a nesting restriction on the choice of design points. It is motivated by an industrial problem in sensor placement for flood monitoring, although the applications to experiment design are not restricted to this field. The nesting structure has two levels. Design points occupy the lower level, and each design point is required to be associated with precisely one member of the higher level. An adaptation of an existing pairwise swap algorithm, sometimes called the Fedorov algorithm, is made to suit this nesting requirement for static linear models. The adaptation features swap operations at the higher level of the nesting structure as well as at the lower level. The performance of this method is demonstrated using simulated and application datasets. Further adaptations are then made to allow for Poisson models, employing a Gaussian quadrature method to effect a Bayesian design. Changes in design points over time are also made possible using applications of the algorithm for the linear model case. The primary motivation across all of the new methodology and its applications is the use of remote sensors in the natural environment as part of the Internet of Things.

Acknowledgements

The completion of this thesis would not have been possible without the constant support and encouragement of my two supervisors, Professor Rebecca Killick and Professor Andrew Titman, both of the school of Mathematical Sciences at the University of Lancaster. They have been extremely generous with their time, and their support is deeply appreciated.

Declaration

I declare that this thesis is my own work and has not been submitted in any form for the award of a higher degree elsewhere.

David Sudell

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

With an increasing number of Internet of Things devices being deployed in the natural environment to monitor conditions over time, problems concerning sensor placement are receiving ever-greater attention, see for example the Wireless Sensor Networks section detailed by Ray (2018). Accurate inference is important, but cost and other practical considerations must also be taken into account, by attempting to limit the resources used in sensor deployment (for example, limiting the number of sensors used). The problem considered in this thesis is that of determining the optimal sensor placement experiment under certain geographical constraints. Sensors are to be placed in a real-world environment in order to measure a single variable of interest. It is assumed that a number of static, location-dependent attributes are known for any potential sensor location. These attributes are to be used as predictors in a statistical model, with the final aim of estimating the model parameters in order to describe the relationship between the location attributes and the sensor-measured response variable. Thus the emphasis is primarily on parameter estimation, although prediction also has practical importance.

In the motivating example, the sensors detect water levels and are installed in gully pots at the sides of roads. A smaller number of localised data-collecting devices, referred to hereafter as *collectors*, are also installed, affixed to lampposts in the vicinity. Each deployed collector will in general have multiple sensors associated to it. This is illustrated hypotheti-

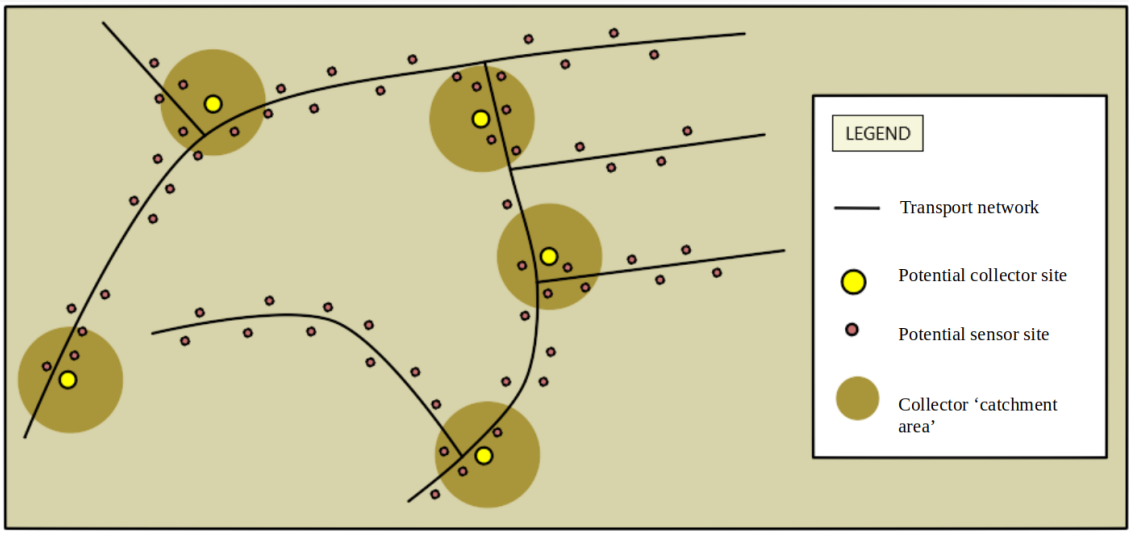

Figure 1.1: Sensor placement application example

cally in Figure 1.1. It is assumed that the sensors are identical in terms of reliability, and are therefore entirely exchangeable with one another; variation in sensor behaviour/reliability is not an area of investigation for this thesis. The sensors transmit their data readings at high frequency to the collector within range. The collectors retain and consolidate the data they have received from these in-range sensors, and then send these on in regular packets to an internet server. One benefit of such a system is the need for only a low number of devices that are capable of long-range transmission or internet access. In fact, transmission over long distances by the sensors themselves is often difficult or impossible due to their physical positioning, potentially below ground level. However, the arrangement does mean that each sensor is reliant upon the presence of a sufficiently close collector. It is usually desirable in such a situation to position clusters of sensors around collectors in such a way that each sensor has no more than one associated collector. This is not only because of limitations on sensor/collector resources that would make a double parentage wasteful; but also because of the nature of many natural or urban environments in which sensors are deployed. In such environments, the location attributes used to design the experiment often require sufficiently great separations in order to vary sufficiently. A greater spread of sensor locations, made possible by a greater spread of collectors will tend, in general, to result in higher variation in location attributes and correspondingly a better experimental design; it will also mean that there are few, if any, multiple collector parentages for sensors.

Thus the sensor placement problem is complicated by the imposition of a geographical two-level nested structure upon the sensor placement. The sensors are at the lower level, and a smaller number of collectors at the higher level. Analogous situations may be found in Wireless Sensor Network problems, as in the relay nodes of Bakhsh (2017), although with an energy-efficiency motivation as opposed to an optimal design one.

While optimal experiment design is a rich area with many results applicable to sensor placement problems, there is currently a lack of methods applicable to these hierarchical constraints. The closest example may be that of Goos and Jones (2019), in which performance of different machines is investigated. Allowance is made for a categorical *branching factor* and, dependent upon this, a categorical (or quantitative) *nested factor*. There is therefore a nesting structure here, where the branching factor could be considered analogous to the collectors in our application. However, in our application we assume that the potential collector sites are chosen based on practical considerations. Hence we do not consider the collector to which a sensor site is associated to have any direct bearing on the distribution of responses at that site and hence the nested models considered in Goos and Jones (2019) are not applicable. In practice, there may be some spatial correlation, as considered in Fedorov (1996). However, the practical desire to choose multiple sensors per collector would complicate this approach considerably. In addition to this, in the motivating scenario there is no guarantee that spatial separation is as important to the model as the other attributes used such as road type; often, very geographically close locations can have quite different attribute values and behaviours. Taking all of these issues into consideration; our objective is to achieve useful inference on the model parameters, and without necessarily having access to any response data before deploying the sensors. It is therefore appropriate to consider the sensor placement problem as an optimal experiment design problem, and build upon existing methods in this area to suit the needs of the nesting structure and practical sensor placement problem.

Chapter 2

Literature Review

2.1 Optimal experiment design

In the design of experiments, the aim is to investigate the effect of experimental *treatments* upon a response variable by applying the treatments to experimental *units*. The treatments are quantities of interest, and the units are the individual items to which the treatments are applied, allocated or assigned. A *factor* is a treatment that can be controlled as an independent variable in order to observe the effect upon the value of the response for each unit. In doing so, one chooses different values or *levels* for each factor involved. A combination of two or more factors at specified levels constitutes a *treatment combination* (Hicks and Turner (1999) p4), and the vector of factor levels for such a combination becomes a *design point*. The treatment combinations to be applied to each unit must be decided before an experiment can be conducted; that is, the set of design points must be chosen. These decisions can be made using the motivation of *optimality* (Morris, 2011). Optimal experiment design aims to engineer the choice of design points before collection of the data, according to some predetermined criterion. This criterion depends upon the requirements of the experiment, but often relates in some way to the information matrix (St John and Draper, 1975). There are many optimality criteria, including the D, A, G and E criteria (Nishii, 1993). The commonly-used D-optimality criterion maximises the value of the determinant of the information matrix; its interpretation as minimising the generalised variance of the parameter estimates (Nguyen and Miller, 1991) makes it a sensible choice as a criterion in investigating

potential methods for our inference problem. This optimisation can be performed before the acquisition of any data as it relies only upon the design. Further constraints may also be imposed in addition to maximizing or minimizing a criterion value; for example, Lee (1988) minimize a convex function of the information matrix, with additional constraints that other convex functions of the information matrix are less than or equal to zero.

2.2 The Fedorov exchange algorithm for exact designs

Exchange algorithms make sequential changes to a starting design by exchanging currently selected design points with currently unselected ones. The starting design usually contains as many points as are intended for the final design. A certain algorithm which we shall from hereon refer to as the *standard Fedorov algorithm*, or simply the *Fedorov algorithm*, is such an exchange algorithm and is one of several designed to achieve good exact designs with respect to D-optimality (Fedorov, 1972). It improves some initial randomly generated design, if possible, by the performance of successive one-for-one swaps of design points. Each swap is made so as to result in the maximal possible increase in the information matrix determinant given the current proposed design. The process is, briefly:

1. Start from an initial random design.

2. Compute, in turn, the change in determinant which would result from making a single swap of each design point in the design with each available non-design point.

3. Of these possible single swaps, choose the one corresponding to the greatest increase in determinant (if any does result in an increase)- perform the swap.

4. Repeat steps 2-3 until no increase is possible.

Note that another popular algorithm bearing Fedorov's name and proposed in the same work gives *continuous* designs, whereas the exchange algorithm described here gives an exact design. Reference to the Fedorov algorithm here will mean the Fedorov exchange algorithm.

2.2.1 Other exchange algorithms

A number of alternative or potentially superior exchange algorithms have been proposed since Fedorov (1972). Mitchell (1974) allows *excursions* in the design; that is, changes in the number of design points that can be exchanged at a time up to a varying limit; the limit is permitted to increase up to a threshold as it becomes increasingly difficult to improve the design close to the optimal design, until exploration is halted when this threshold is reached. Johnson and Nachtsheim (1983) provides an exchange algorithm based on selecting a set (of pre-specified cardinality) of points that results in the lowest variance of predictions. More recently, Labadi (2015) proposes refinements of the Fedorov exchange algorithm with exchange of two points at a time, and Harman and Filová (2020) proposes a 'randomised exchange algorithm' for D-optimality of approximate designs, based upon trying exchanges of weights (with the design expressed in terms of weights on each design point) within repeatedly selected batches of design points. Most of these algorithms are similar in nature to Fedorov but slightly alter the way in which the swap is decided upon, or the number of swaps performed at a time. These may include extra user-set parameters affecting run-time but not the final solution. Thus we acknowledge that such modifications to Fedorov could be employed, but we do not include them in our adaption of the algorithm.

2.3 Contributions

The broad scenario of a nested structure for sensor placement has been examined before using relay nodes (analogous to the collectors) in Wireless Sensor Networks; for example the placement of the relay nodes with fixed sensor locations, minimising the number of nodes and a cost function (Li et al., 2017), and clustering the relay nodes for energy efficiency (Chugh and Panda, 2018). However, there is an absence of sensor placement methodology relating to statistically-motivated parameter estimate optimization subject to a nested structure. Conversely, in the statistical literature, as mentioned above, optimal designs may currently be sought according to appropriate criteria on the information matrix and further constraints on the design space may be imposed, but thus far such constraints have not to the authors' knowledge included a nested structure.

The contribution of this thesis is to propose an algorithmic method to address this gap and combine the sensor placement and experiment design areas, while observing the nested structure requirement. In Chapter 3 notation is introduced, and an exchange algorithm is proposed for construction of exact designs with the aim of maximising D-optimality for linear models. Simulations and application data are used in assessing algorithm performance. Chapter 4 extends the methodology of Chapter 3 to allow for a Poisson model assumption, whilst Chapter 5 addresses the possibility of design point reallocation over time.

Chapter 3

Linear Models

3.1 Problem structure

This chapter addresses the problem of constructing D-optimal designs for linear models, subject to the nesting constraint (which was described in Section 1.1 and is formalised shortly using set notation to describe a partition). We begin by introducing the relevant notation. For $k \in \mathbb{N}$ we introduce a compact subset of $\mathbb{R}^k$, χ, as the design space; n elements x_i, $i = 1, \ldots, n$ of χ are to be chosen for observation in an experiment, where the x_i are k-tuples. The compactness of χ is to prevent exclusion of suprema of function values (Kiefer, 1959). Note that k will be equal to the number of predictor variables being recorded and used in the experiment. The choice of model(s) to be applied to the sensor data naturally affects the methods used in planning the experiment. In this chapter we make the assumption that standard linear models with least-squares estimation will be used. Suppose that the choices of the x_i are restricted to a pre-determined finite subset $\mathcal{A}$ of N distinct points of χ, corresponding in practical terms to the available sites for sensor installation. It may be assumed that the attributes corresponding to the entries of the x_i are permitted to be continuous or categorical. The composition of $\mathcal{A}$ is assumed to be known. Suppose further that we have a sequence $\mathbf{f}$ of p continuous real-valued known functions $f = (f_1, f_2, \ldots, f_p)$ on χ, to give the coefficient structure of the linear model; write $f(x)$ for $(f_1(x), f_2(x), \ldots, f_p(x))$ when $x \in \chi$. Observations y_i are to be collected once the x_i have been chosen, with the

observations assumed to be of the form:

$$y_i = \sum_{j=1}^{p} (f_j(x_i)\beta_j) + e_i \tag{3.1}$$

for p unknown coefficients β_j whose values are to be estimated by the experiment, and n error terms e_i which are uncorrelated, and each with Gaussian distribution; mean 0 and variance σ^2. Let ξ_A be a probability measure in a class of probability measures Ξ_A giving a design with the following restrictions. Suppose that, for every $x \in \chi$, $\xi_A(x) = q/n$, for some $q \in \{1, 2, \ldots, n\}$, then ξ_A assigns whole numbers of design points to the available experimental observation units (sensor sites), and the design is commonly referred to as *exact*. In practical terms this is of course a necessity. Suppose further that ξ_A uses each x either once or not at all (so that the experiment may be conducted with fixed sensors in certain available locations), and restrict ξ_A to using points in $\mathcal{A}$ (again, for the locations to be real possibilities for sensor installation). The design matrix for ξ_A is:

$$\mathbf{X}(\xi_A) = \begin{bmatrix} f_1(x_{A_1}) & f_2(x_{A_1}) & \ldots & f_k(x_{A_1}) \\ f_1(x_{A_2}) & f_2(x_{A_2}) & \ldots & f_k(x_{A_2}) \\ \vdots & & \ddots & \vdots \\ f_1(x_{A_n}) & f_2(x_{A_n}) & \ldots & f_k(x_{A_n}) \end{bmatrix}.$$

for $\mathcal{A} = \{x_{A_1}, \ldots, x_{A_n}\}$, and the information matrix $\mathbf{M}(\xi_A)$ is, up to proportionality,

$$\mathbf{M}(\xi_A) = (1/n)(\mathbf{X}(\xi_A)'\mathbf{X}(\xi_A)),$$

with individual elements, up to proportionality, given by:

$$m_{j_1,j_2}(\xi_A) = \frac{1}{n} \sum_{a \in \{A_1, A_2, \ldots, A_n\}} f_{j_1}(x_a) f_{j_2}(x_a).$$

Since we assume that each potential sensor site belongs to precisely one potential collector site, suppose that $\mathcal{A}$ is partitioned by a family of subsets $H_1, H_2, \ldots, H_{h_0}$ for $h_0 \in \mathbb{N}$, so that h_0 gives the number of potential collector sites available for installation of collectors.

Use a membership function g:

$$g : \mathcal{A} \to \{1, 2, \ldots, h_0\}$$

such that

$$x \in H_l \Leftrightarrow g(x) = l \quad \forall x \in \mathcal{A}, l \in \{1, 2, \ldots, h_0\},$$

and let Ξ_H be a class of designs ξ_H satisfying a final restriction in addition to those already described; that ξ_H uses design points from at most h_1 of the subsets forming the partition, with h_1 representing the number of collectors to be used. That is:

$$|\{g(x) : \xi_H(x) = 1, x \in \mathcal{A}\}| \leq h_1;$$

for some $h_1 \in \{1, 2, \ldots, h_0\}$. Note that in a slight abuse of the notation, h_1 here is an integer quantity of these potential collector sites to be used, and does not represent the identity of a particular element of the partition. Assuming that ordinary least squares estimates of the parameters are to be obtained following the data collection phase, we have

$$\hat{\beta} = (\mathbf{X}(\xi_H)'\mathbf{X}(\xi_H))^{-1}\mathbf{X}(\xi_H)'\mathbf{y}$$

for the observations $\mathbf{y} = (y_1, y_2, \ldots, y_n)'$, and the variance-covariance matrix of the parameter estimates $\hat{\beta}$ will be given by

$$\sigma^2(\mathbf{X}(\xi_H)'\mathbf{X}(\xi_H))^{-1}.$$

One further item of notation is helpful in describing the algorithm methodology below. Suppose a design ξ_H is in the process of being altered, for instance by the addition or removal of one of its design points. Suppose also that there are several candidate alterations. The candidate altered designs will be denoted in subscript with lower-case indexing letters, for example ξ_{H_i}. Once the change has been made, subsequent reference to ξ_H can then be assumed to imply the new design; that is, in our example the i is dropped following the change. Additional levels of conditionality are expressed with successive letters, so that a two-level-deep conditionality may for example be expressed by $\xi_{H_{ij}}$.

3.1.1 Assumptions

The following assumptions will be made regarding the experiment:

- Responses from the experiment are of the form given in expression 3.1; that is, it is sensible to attempt to describe them through use of a linear model, allowing for interactions.
- We assume Gaussianity with zero covariance; in practical terms, this means that observations from distinct locations are independent. This is a reasonable assumption as distinct sensor sites are not expected to exhibit dependent behaviour beyond the effects of the known location attributes.
- Observations are permitted only once at sites in $\mathcal{A}$, and will be taken only for points in $\mathcal{A}$.
- The nesting structure restriction, caused by the need for local data collectors and described by the $H_1, H_2, \ldots, H_{h_0}$ partition, is to be observed.

We may now proceed to a description of the algorithm methodology.

3.2 Algorithm Methodology

The material in Section 1.1 provides some background to optimal experiment design and in particular the use of the Fedorov algorithm to attempt to construct D-optimal exact designs. If we are to assume that construction of a design with a high determinant $|M(\xi_H)|$ is desirable for the sensor placement problem, the methodology of the Fedorov algorithm provides a starting point. We require a design in Ξ_A that also satisfies the nesting structure requirement (i.e. that is also in Ξ_H). The Fedorov algorithm does not respect nesting structure. Thus we propose a new algorithm which, inspired by the Fedorov algorithm, solves the nested design problem. In designing such an algorithm, several problematic questions particular to the nesting structure scenario must be considered. Naïvely, one might try to optimise individual designs for each chosen collector. However, the lack of inter-subset communication is likely to lead to replication of structure between collectors,

and hence sub-optimal designs. Instead, it is necessary to account for the effect of any design change on the determinant of the overall design. However, note that the presence of nesting induces a large number of possibilities. It is not practical to consider each possibility explicitly; indeed, the choice of collectors alone is equal to $\binom{h_0}{h_1}$.

3.2.1 Nested Fedorov algorithm

The algorithm developed adopts the general approach of attempting to make pairwise exchanges, but on two levels rather than the one used by the standard Fedorov algorithm. Were the collector installation locations fixed, the standard Fedorov algorithm could be applied to the set of potential swaps that satisfy the nesting constraints. However, since the optimal set of collector locations is unknown, the algorithm also accommodates the exchange of collectors. The algorithm proceeds from an initial random design in Ξ_H to make swaps of the potential collector sites in the nesting. Each such swap involves identifying the best pair of candidate collector sites for the removal and addition, based upon the effect on the determinant $|M(\xi_H)|$ given the design points associated with each candidate collector site. Once a collector swap has been made, the standard Fedorov algorithm is run with respect to the design points (sensor locations) and only allowing swaps consistent with the currently chosen collector sites. Allowances are made for particular cases involving fluctuations in the numbers of design points being used, and cases of an entire chosen collector site becoming unused. Algorithm 1 gives an overview of the algorithm. The various matrix operations involved are made significantly more efficient using the methods described in Section 3.3. The algorithm inputs for the Nested Fedorov algorithm are:

1. n (the prescribed number of design points (sensors)) and h_1 (the prescribed number of collectors to be used)- assume that the h_0 potential collector sites are simply labelled $1, \ldots, h_0$;

2. $\{(x, g(x))\}\ \forall x \in \mathcal{A}$ (the nesting structure);

3. Criterion for minimum determinant before a matrix is deemed singular- this is not referred to explicitly in the algorithm description, but checks are made at the relevant points before altering the design, in order to avoid a singular matrix M.

input : n, h_1, h_0,$\{(x, g(x))\}$, and minimum determinant criterion.

Create a random sample $C \in \{1, 2, \ldots h_0\}$, with $|C| = h_1$;

Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with a random starting set of $\min(n, |\{x : g(x) \in C\}|)$ design points- resultant design $\xi_H \in \Xi_H$;

while *an increase in* $|M(\xi_H)|$ *is achieved* **do**

  if $|\{x : g(x) = c \ AND \ \xi_H(x) = 1\}| > 0 \ \forall c \in C$ **then**

    for $c_i \in \{c : c \in C\}$ **do**

      Set the design for ξ_{H_i} to $\{x : g(x) \in (C \setminus c_i)$ AND $\xi_H(x) = 1\}$.;

      for $p_j \in \{g(x) : g(x) \notin C\}$ **do**

        if $|\{x : g(x) = p_j\}| \leq (n - |\{x : \xi_{H_i}(x) = 1\}|)$ **then**

          Set the design for $\xi_{H_{ij}}$ to $\{x : \xi_{H_i}(x) = 1$ OR $g(x) = p_j\}$

        else

          Set $\xi_{H_{ij}} = \xi_{H_i}$;

          for k *in* $1 : (n - |\{x : \xi_{H_i}(x) = 1\}|)$ **do**

            for $x_l \in \{x : g(x) = p_j \ AND \ \xi_{H_{ij}}(x) = 0\}$ **do**

              Set the design for $\xi_{H_{ijk}}$ to $\{x : \xi_{H_{ij}}(x) = 1\} \cup x_l$

            end

            $\xi_{H_{ij}} \leftarrow \arg\max_{\xi_{H_{ijk}}} |M(\xi_{H_{ijk}})|$.

          end

        end

      end

      $\xi_{H_i} \leftarrow \arg\max_{\xi_{H_{ij}}} |M(\xi_{H_{ij}})|$.

    end

    $\xi_H \leftarrow \arg\max_{\xi_{H_i}} |M(\xi_{H_i})|$ and then $C \leftarrow \{g(x) : \xi_H(x) = 1\}$;

    Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with starting design ξ_H.

  else

    for $p_i \in \{g(x) : g(x) \notin C\}$ **do**

      Set the design for ξ_{H_i} to $\{x : \xi_H = 1$ OR $g(x) = p_i\}$.

    end

    $C \leftarrow C \cup p_i$ where $i = \arg\max_i |M(\xi_{H_i})|$. Note that ξ_H is not altered here;

    Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with starting design ξ_H.

  end

  if *The current run of the while loop has not resulted in an increase in* $|M(\xi_H)|$, **then**

    undo its effects, and halt the while loop at this point.

  end

end

output: ξ_H (C may be inferred from this).

Algorithm 1: Nested Fedorov Algorithm

Given the nature of the nesting structure of possibilities in our problem, the quality of the end design is sensitive to the starting design (see the Type A simulation results in Section 3.5.1 for an illustration of this), and it is therefore advisable to perform multiple *random starts* of the algorithm when searching for a design. In addition to this Nested Fedorov algorithm, two further methods are employed in the testing process, purely for comparative purposes. The first method takes a simple random sample from the set of potential collectors sites, and then from the associated design points a random sample giving a non-singular design; this process is repeated as many times as required. The second method, referred to as the *naïve* one, chooses a set of collector sites, and thence the design points, based upon expectation of the points averaged over a variety of weightings for the predictors- points with higher mean expectation are favoured, with the reasoning that a naïve deployment of sensors might give priority to geographical locations at which events of interest have been observed to occur with highest frequency (that is, for example, at flooding hotspots).

3.3 Updating the information matrix

The material in this section describes methods for computing the effects of design point changes upon the relevant matrix determinant and inverse. Such methods have been employed before in optimal experiment design, for example in Arnouts and Goos (2009).

3.3.1 Determinant update

Each comparison of a candidate design over a current one, during one pass of the Fedorov algorithm, requires knowledge of the candidate design's information determinant as well as the existing determinant $|M(\xi)|$. Let x_i be the candidate row to be added to the design and x_k the row to be removed (using k rather than j since j was associated with columns in Section 3.1). Then, given that inserting the row x_i in the (design) matrix $\mathbf{X}$ causes the matrix $M = \mathbf{X}'\mathbf{X}$ to become $M + x_i x_i'$ whilst removing the row x_k from $\mathbf{X}$ gives the matrix $M - x_k x_k'$, (treating x_i and x_k here as column vectors so that $x_i x_i'$ and $x_k x_k'$ are equal to the outer products of each respective vector with itself), the expression of interest for the determinant resulting from the potential swap of x_i and x_k is $|M + x_i x_i' - x_k x_k'|$.

Proposition 3.3.1 *For a square invertible matrix M and column vectors x_i, x_k;*

$$|((M + x_i x_i') - x_k x_k')| = (1 + \Delta)|M|$$

where

$$\Delta = b - a - ab,$$

and

$$a = x_k' \left(M^{-1} - \frac{M^{-1} x_i x_i' M^{-1}}{1 + x_i M^{-1} x_i} \right) x_k \text{ and}$$
$$b = x_i' M^{-1} x_i.$$

Proof. By Theorem 18.1.1 in Harville (1997), for an invertible square matrix M and two column vectors v_1 and v_2,

$$|M + v_1 v_2'| = (1 + v_2' M^{-1} v_1)|M|.$$

Since the case for removal of a row follows easily as a slight variant on this,

$$\begin{aligned} |M - v_1 v_2'| &= |M + (-v_1) v_2'| \\ &= (1 + v_2' M^{-1} (-v_1))|M| \\ &= (1 - v_2' M^{-1} v_1)|M|, \end{aligned}$$

the simultaneous insertion of one row and deletion of another can be expressed as

$$\begin{aligned} |((M + x_i x_i') - x_k x_k')| &= (1 - x_k (M + x_i x_i')^{-1} x_k)|(M + x_i x_i')| \\ &= (1 - x_k' (M + x_i x_i')^{-1} x_k)(1 + x_i' M^{-1} x_i)|M|. \end{aligned}$$

As can be seen from the above, the approach requires the inverse of M. Sherman and Morrison (1950) consider the effect on an inverse of changing single matrix elements led to an expression for the update of a matrix in the required manner (Bartlett, 1951); for a

square invertible matrix M and two column vectors v_1 and v_2,

$$(M + v_1 v_2')^{-1} = M^{-1} - \frac{(M^{-1} v_1 v_2' M^{-1})}{(1 + v_2' M^{-1} v_1)}, \tag{3.2}$$

where similarly to the case of the determinant update, it can be seen that the case for removal of a row is:

$$(M - v_1 v_2')^{-1} = (M + (-v_1) v_2')^{-1} \tag{3.3}$$

$$= M^{-1} - \frac{(M^{-1}(-v_1) v_2' M^{-1})}{(1 + v_2' M^{-1}(-v_1))} \tag{3.4}$$

$$= M^{-1} + \frac{(M^{-1} v_1 v_2' M^{-1})}{(1 - v_2' M^{-1} v_1)}. \tag{3.5}$$

Thus we arrive at an improved expression for the matrix update:

$$|((M + x_i x_i') - x_k x_k')| = \left(1 - x_k' \left(M^{-1} - \frac{M^{-1} x_i x_i' M^{-1}}{1 + x_i M^{-1} x_i}\right) x_k\right) (1 + x_i' M^{-1} x_i) |M|$$

Multiplying out gives the form with Δ defined as in the proposition statement. ■

From the above, to calculate the determinant that would result from a single row swap in the design matrix $\mathbf{X}$ (not in M, note, which is $\mathbf{X}'\mathbf{X}$ assuming an exact design) during the process of running the Fedorov algorithm, all that is required is to know M^{-1}, as well as the two design points involved in the row swap. Note that whilst $|M|$ also appears in the expression in (3.3.1), multiplication out of everything preceding the $|M|$ results in an expression of the form $(1+\Delta)$ so that if the Δ is positive, the corresponding row swap should cause an increase in $|M|$. This quantity Δ is often referred to as *Fedorov's delta* (Cook and Nachtsheim, 1980). Thus swaps can actually be made on the basis of their effect on Δ rather than on $|M|$ directly. Whilst $|M|$ is not required for the update of M^{-1} either, as shall be seen below, it is still useful to be able to calculate and update $|M|$ for comparisons and observation, and the extra computation to convert Δ to $|((M + x_i x_i') - x_k x_k')|$ is small ($o(1)$).

3.3.2 Inverse update

The determinant update in Section 3 necessitates knowledge of M^{-1}. To avoid calculating a new inverse when performing a row swap, the following allows update of the inverse in a similar manner to that of the determinant above:

Corollary 3.3.2 *For a square invertible matrix M and column vectors x_i, x_k;*

$$(M + x_i x_i' - x_k x_k')^{-1} =$$
$$\left(M^{-1} - \frac{M^{-1}x_i x_i' M^{-1}}{1 + x_i' M^{-1} x_i}\right) + \frac{\left(M^{-1} - \frac{M^{-1}x_i x_i' M^{-1}}{1+x_i' M^{-1}x_i}\right) x_k x_k' \left(M^{-1} - \frac{M^{-1}x_i x_i' M^{-1}}{1+x_i' M^{-1}x_i}\right)}{1 - x_k' \left(M^{-1} - \frac{M^{-1}x_i x_i' M^{-1}}{1+x_i' M^{-1}x_i}\right) x_j}$$

Proof. By (3.2),

$$(M + x_i x_i' - x_k x_k')^{-1} = (M + x_i x_i')^{-1} + \frac{(M + x_i x_i')^{-1} x_k x_k' (M + x_i x_i')^{-1}}{1 - x_k'(M + x_i x_i)^{-1} x_k},$$

which by (3.3) is equal to the stated expression. ■

This expression relies only upon M^{-1} and the two rows involved in the swap. When large numbers of successive row swaps are being performed, these update techniques reduce computation time considerably by avoiding any more than one initial calculation of an inverse and determinant per run of the Fedorov algorithm.

3.4 Datasets

Three types of dataset are used in evaluating the Nested Fedorov Algorithm's efficacy. The first two are simulated, and the third is an application dataset; all three are now described in detail.

3.4.1 Simulated data Type A

The first set of small-scale simulated sets of potential design points is constructed in the following way. It is desirable that a design ξ_Ω that is known to have an optimal value of $|M(\xi_H)|$ be concealed in the simulated data, in order to test the algorithm's tendency to

identify it. For each simulation scenario, a factorial design array size 16×4 is constructed on the range $[-1, 1]$, and a column of $1'$s appended. This forms the optimal design, so that $n = 16$, $p = 5$. This is then augmented with points lying within the hypercuboid formed by this design, in one of two possible ways. The first is to add uniformly randomly sampled vectors on the hypercuboid to make the total number of points up to 160. The second consists firstly in augmenting the optimal design with a copy of itself, one of whose columns is scaled by 0.95, then augmenting further to a total of 160 points using uniformly randomly sampled vectors confined to the hypercuboid, as with the first method. These two types of simulated datasets are referred to later as *hidden-design* and *hidden-design-with-decoy* respectively. The set of candidate design points in each case is then allocated a nesting partition structure such that:

- For some predetermined value of h_1, the known optimal design would be accessible through a choice of h_1 collector sites, thus allowing the algorithm the possibility of attaining this design.
- For the nesting structure of the hidden-design-with-decoy datasets, as for the optimal design, the altered copy of the optimal design is accessible through a choice of h_1 collector sites; additionally, it is ensured that no element of the altered optimal design is assigned to the same potential collector site as any element of the actual optimal design, thus testing the algorithm by inviting it to choose a different set of collector sites than those that would lead to the unaltered optimal solution.
- Overall the expected number of points per potential collector site is the same for all potential collector sites, accounting for the need to satisfy the preceding two requirements by adjusting the sampling probabilities accordingly for potential collector sites belonging to/not belonging to the optimal and (if present) altered optimal designs.

Beyond these conditions, the creation of the partition is random.

3.4.2 Simulated data Type B- environmental

A 50×50 grid of discrete location points is generated, to represent locations evenly spaced in each of the two axes, and a Euclidean distance matrix D computed for the grid (2500×2500,

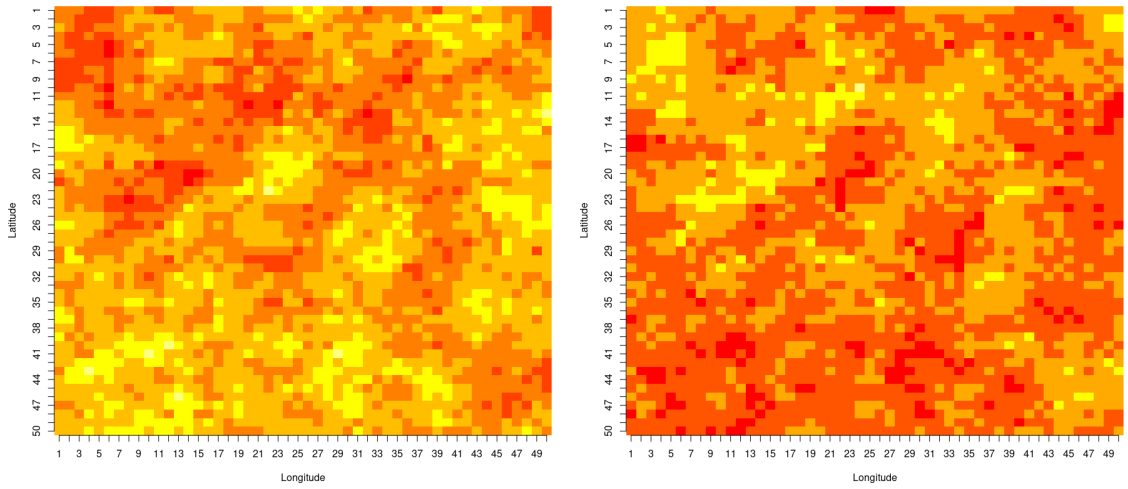

Figure 3.1: Simulated environmental datasets heatmaps for two negatively correlated characteristics

symmetric and with 0 diagonal). A 2500-dimension multivariate normal sample is then generated, in the manner of the generation of artificial distribution data by Dormann et al. (2007), with all means equal to 0 and a covariance matrix Σ with elements σ_{lm} given by $\sigma_{lm} = exp(\text{-}\rho \times d_{lm})$ where d_{lm} is the corresponding entry in D and ρ is a pre-selected constant to control the degree to which sites are mutually correlated. This sample, arranged in a 50×50 matrix $\Phi = [\phi_{ij}]$ to match the elements of the location grid, is then used to generate location characteristics p_{ijr}, with r the index for the location characteristic in question, in the following way:

$$p_{ijr} = \alpha_r \phi_{ij} + \epsilon_{ijr},$$

where:

1. $\epsilon_{ijr} \sim N(0, 0.25)$ is randomly sampled, the choice of variance value being made in order to achieve what appears to be a reasonable amount of variance in the location characteristics across the grid.

2. the quantities $\alpha_r \in \mathbb{R}$ are specified parameters, chosen collectively based upon the location characteristics pertaining to the r's such as to match common sense expectations of the behaviour of the location characteristics- for example, α_r might be chosen to be positive for altitude and rainfall, and correspondingly negative for traffic, al-

though it is in a sense safer to treat each of these variables in a more abstract manner, given their artificiality.

Figure 3.1 gives an example pair of heatmaps for two characteristics in one of these simulated environments. The two characteristics are negatively correlated with one another. Following the generation of these simulated environments, potential collector sites are randomly sampled from a lower-granularity 5×5 subset grid, and potential sensor locations sampled from those members of the original 50×50 grid belonging to one of the sampled potential collector sites, on the basis of Euclidean distance. Four different possible numbers of available collectors are used; $2, 3, 4$ and 5; and for each of these, the total number of potential collector sites made available is greater by a factor of four (i.e. 8,12,16,20); the number of sensors used is greater by a factor of six; and the number of potential sensor sites made available is greater by a factor of 60, all relative to the number of collectors used. For each of these four cases, 50 different environments are simulated, and on each of these 50 environments the Nested Fedorov Algorithm is run 16 times (parallel processing is utilised here).

3.4.3 Application data- monitoring flooding of roads

A dataset containing different variables giving attributes of real-world locations on roadsides was provided by InTouch Ltd.. The attributes available include, for each potential sensor site:

- Latitude
- Longitude
- Altitude
- Road speed limit (all locations were on roadsides)
- Tree count within a 100m radius
- Average daily number of stops of any bus at a bus stop within 100m.

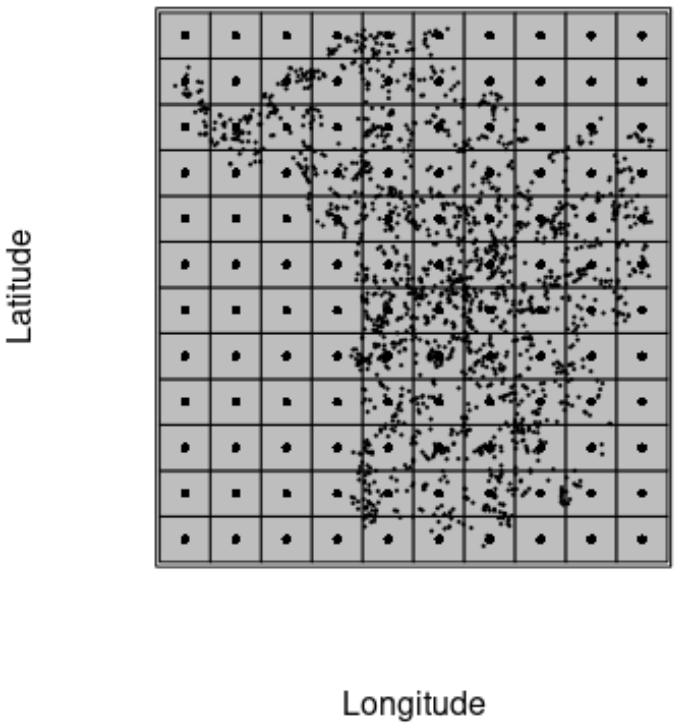

Figure 3.2: Application dataset locations

A regular grid of squares is used to partition the geographical region concerned, with the centre of each square as a potential collector site. The range of a collector is indicated by the size of the grid squares. A ten by twelve $\sqrt{2}$km-wide spacing is used, such that each collector's 'range' is at most 1km. Figure 3.2 depicts the set of all potential sensor sites, superimposed on the grid of square collector regions described above. Note that certain potential collector sites near the top left have no potential sensor sites.

3.5 Testing Results

Results for applying the Nested Fedorov algorithm to each of the simulated dataset types, as well as to the application dataset, are described with accompanying plots. Subsequently, further results for all dataset types are described for calculation of expected values, and prediction variances.

3.5.1 Testing simulated datasets Type A

The Nested Fedorov algorithm is first applied to the two type A simulated hidden-design and hidden-design-with-decoy datasets- recall that the hidden-design datasets contains a hidden optimal design with the remaining points being randomly uniformly sampled, while the hidden-design-with-decoy datasets also contains a slightly altered copy of the optimal design. As described above, the simulated datasets are constructed using $|\mathcal{A}| = 160$, $n = 16$; for the testing results, the nesting structure is then set up with $h_0 = 20$ and $h_1 = 5$. 500 initial sets of potential design points and nesting structures are run for each of the two types, with 16 algorithm random starts (parallel processing is utilised here) for each of these 500. It should be noted that the practicality of generating such algorithm testing scenarios with known D-optimal solutions does of course dwindle rapidly as the parameters are changed so as to increase the numbers of possibilities. Figure 3.3 shows the D-efficiency achieved (that is, $(|M(\xi_H)|/|M(\xi_\Omega)|)^{\frac{1}{p}}$) for the known optimal design for design space type A (hidden-design), at successive numbers of collector swaps performed during an algorithmic run; the corresponding plot for the hidden-design-with-decoy datasets is extremely similar and is not shown. Table 3.1 summarises these simulated results. With the known optimal design

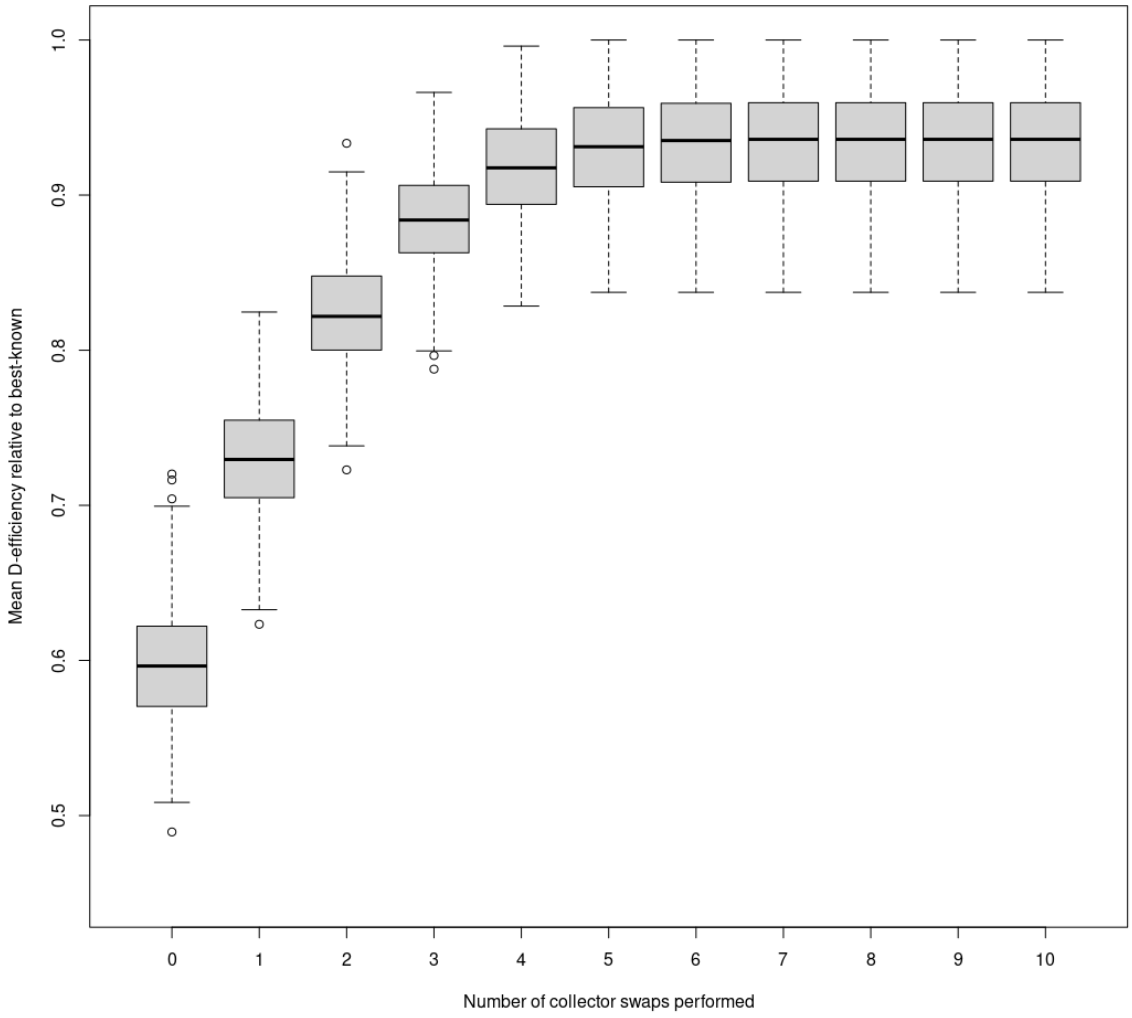

Figure 3.3: Mean efficiency relative to known optimal design

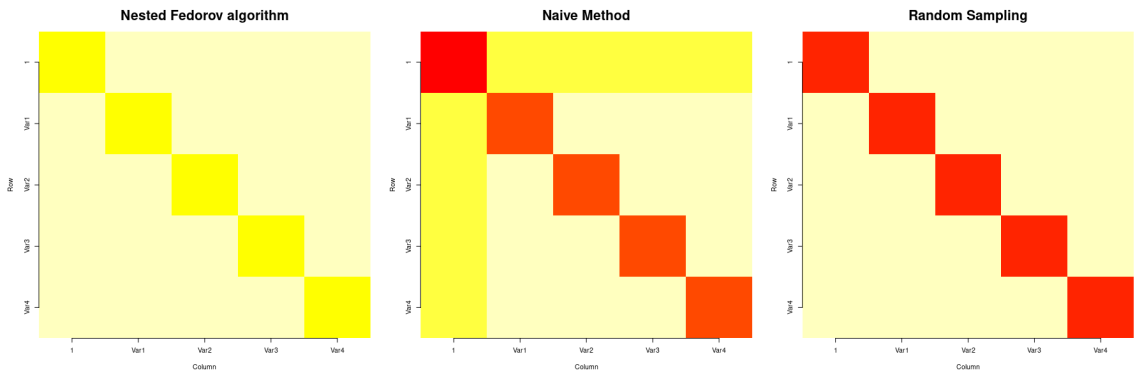

Figure 3.4: Simulated datasets (Type A hidden-design), absolute values of mean entries of $M(\xi_H)^{-1}$. Darker shading indicates higher values.

	Method	**Statistic type**			
		P_Ω		D	
		Dataset type			
		hidden design	**hidden-design with-decoy**	**hidden design**	**hidden-design with-decoy**
1	Nested Fedorov, after 2 swaps	0.25	0.19	0.73	0.82
2	Nested Fedorov, after 10 swaps	0.42	0.35	0.94	0.96
3	Naïve method	0.15	0.12	0.52	0.56
4	Random sampling	0.05	0.05	0.36	0.42

Table 3.1: Simulated results (for Type A, hidden-design and hidden-design-with-decoy). Numerical values are to 2 decimal places.

denoted by ξ_Ω, the statistic types are defined as:

- $D = \text{mean}(|M(\xi_H)|/|M(\xi_\Omega)|)^{1/p}$ is the mean D-efficiency;
- $P_\Omega = \text{mean}|\{x : \xi_H(x) = 1, \xi_\Omega(x) = 1\}|/n$ where $n = 16$- this is the mean proportion of the sensors belonging to the optimal design that were chosen by the algorithm (note that the || notation here denotes set cardinality, not matrix determinant as in the previous bullet point).

Random samples of 100 for each dataset/nesting structure are also run, as well as a Naïve sampling for each of these, and the results averaged.

Observations and further information:

- For the hidden-design scenario (no decoy), the algorithm was able to identify an optimal design in 488 out of 500 of the datasets that were simulated, for at least one of the sixteen random starts performed per dataset (and in many cases for more than one of these). For the hidden-design-with-decoy scenario, this figure was 493 out of 500. Figure 3.3 indicates that the performance of six swaps is almost always sufficient to achieve a high efficiency under these circumstances (but this should not be taken as a general rule).

- From Table 3.1, while the naïve method tends to find better designs than does the random sampling method, both are consistently outperformed by the algorithm, as hoped.

- It can be seen that the presence of a scaled copy of the known optimal design is not a detriment to the algorithm's ability to find good designs; indeed, the mean efficiency tends to be greater when a decoy is present, as the algorithm has been provided with alternative routes to a good if not optimal design, lessening slightly the importance of the choices that a given random start makes. The presence of scaled copies or structures resembling an optimal solution(s) is of course in practice dependent upon the environment and the nesting structure, and with the optimal design unknown is impossible to detect.

- The variance of the D-efficiences is 0.00143 for the initial designs, and has only reduced to 0.00136 upon completion of the tenth collector swap. This illustrates the high sensitivity of the end design's quality to the starting design.

Figure 3.4 gives a visualisation of the absolute values of $M(\xi_H)^{-1}$, with the matrix entries first averaged (mean) over the relevant results for each of the algorithm results, the naïve method and the random sampling. This matrix is proportional to the covariance matrix for the parameter estimates in the linear model. As hoped, the algorithm gives lower values (lighter shading on the plot) for these estimator variances than do either of the other two methods. The off-diagonal covariances are almost all lower, and those that are not are still of comparable value.

3.5.2 Testing simulated type B (environmental) datasets

Recall that the simulated Type B environmental datasets differ from the Type A hidden-design and hidden-design-with-decoy datasets not only in the method of construction, but in the fact that no hidden design is built into the structure. It is therefore not possible to make any comparison with a known optimal design's determinant. However, the algorithm does appear to settle on average on designs which would probably not be bettered by further algorithm random starts. Figure 3.5 gives achieved D-efficiency values relative to

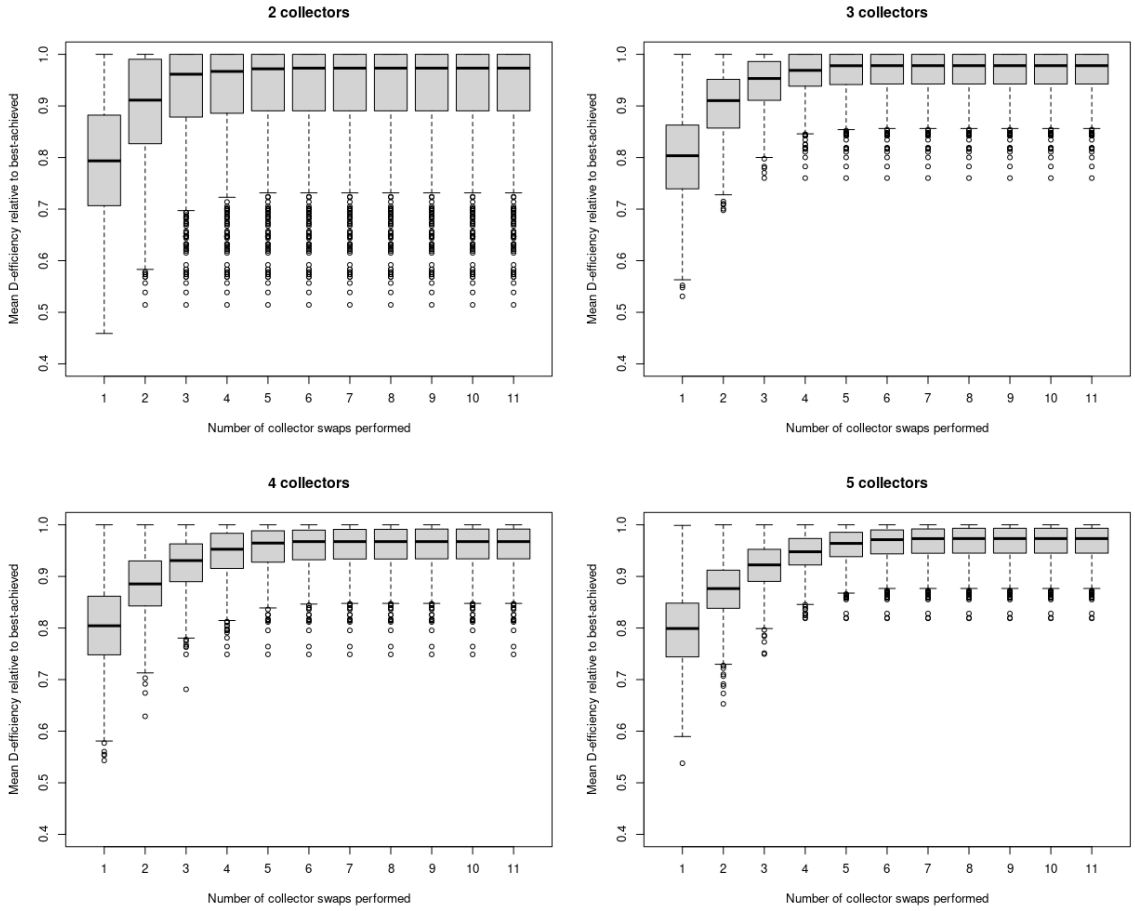

Figure 3.5: Simulated environmental, D-efficiency achieved

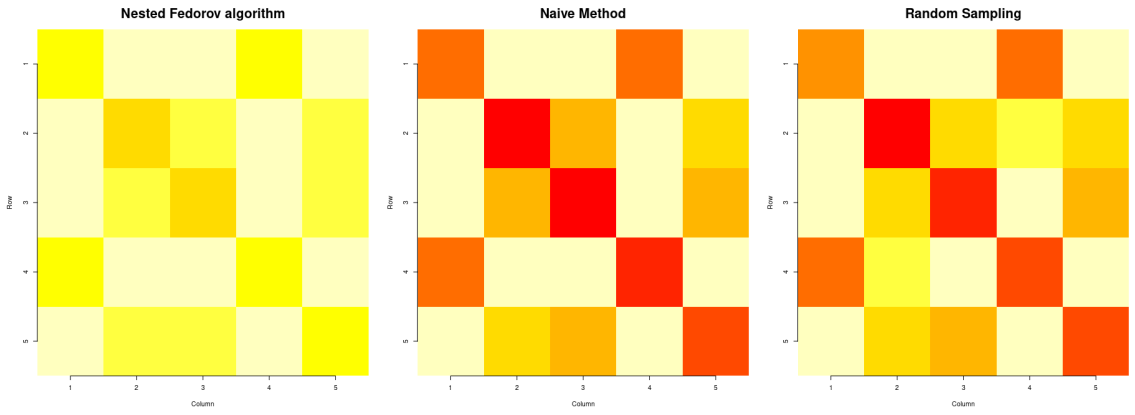

Figure 3.6: Simulated environmental datasets, absolute values of mean entries of $M(\xi_H)^{-1}$. Darker shading indicates higher values.

the best found design, with increasing numbers of collector swaps made, for each of the four setup scenarios described in Section 3.4.2. While an increasing number of collectors (and associated proportional increases in the numbers of potential collector sites, potential sensor sites and available sensors, again as described in Section 3.4.2) causes the algorithm to require a greater number of collector swaps before no further increases appear possible, the algorithm still appears able to find no better solutions after approximately six collector swaps in the worst case. It should be noted that with a higher number of available collectors comes in general a higher potential for improvement after multiple collector swaps, because of the potential need to change each of the originally chosen collectors in turn. In terms of comparison of the algorithm performance with that of the other two methods, Figure 3.6 depicts absolute values of means of entries of $M(\xi_H)^{-1}$ as in Figure 3.4- once again, the algorithm gives lower variances (lighter-shaded diagonal) for the parameter estimates than do either of the other methods, and the off-diagonal covariances are almost all lower, with the remaining being of comparable value.

3.5.3 Testing on the application data

The Nested Fedorov algorithm is run on the application dataset using all six attributes, with 128 random starts of the algorithm performed (parallel processing is utilised here in batches of sixteen). 10 collectors are used, over 60 potential collector sites, and 100 sensors over 2037 potential sites (subject of course to the choice of collector sites). The results are depicted in Figure 3.7 with D-efficiency relative to best design found, plotted against the number of collector swaps performed. As with the simulated environmental datasets, the application one affords no known means of calculating the value of the optimal determinant for our problem, and so a value of 1.0 on the y axis this plot should not be taken necessarily to mean an optimum- it corresponds to the highest-determinant designs found by the algorithm. By the point of 10 swaps, the mean determinant appears to be levelling off. As was the case for the simulated data, there is a marked increase in determinant during the earlier collector swaps, suggesting that performing a large number of algorithm random starts with a suitably calibrated relatively low number of collector swaps is probably the most efficient means of achieving the best design possible with whatever resources are available. Figure

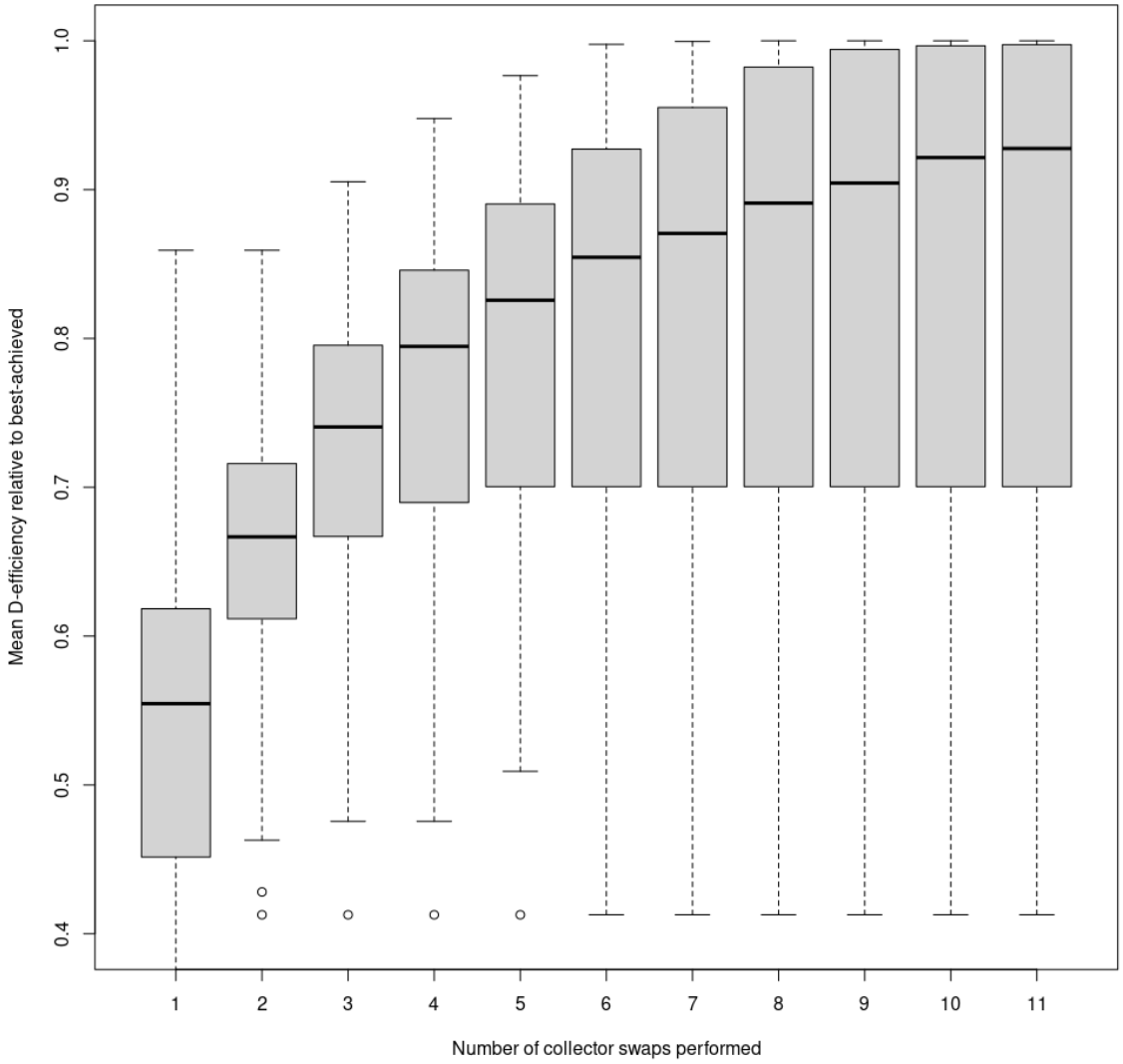

Figure 3.7: Application dataset, mean D-efficiency achieved using nested Fedorov algorithm, 128 algorithm random starts

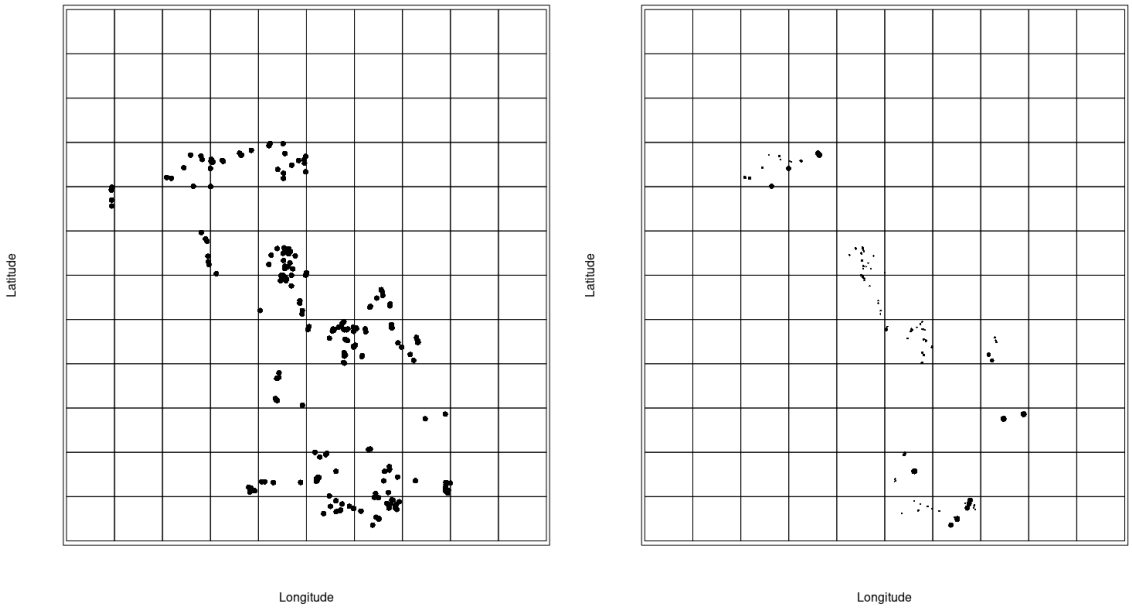

Figure 3.8: Application dataset: chosen sensor locations in any design with at least half the maximal value achieved overall; and chosen sensors in the maximal design.

3.8 shows the geographical distribution of chosen sensors for the application dataset. The left plot includes any sensor that features in any design that achieved at least half of the maximal value achieved overall; a total of 23 different collectors appear- recall that the use of at most 10 collectors was permitted during the course of any one of the 132 algorithm random starts. This suggests that of the available 60 collectors, the same ones tended to be chosen in order to achieve the designs that were, in the results, maximal or approaching maximal. The maximal designs themselves were identical and appeared three times in the 132 algorithm random start results. The plot on the right of the figure shows the sensor sites chosen for this maximal design, with point size representing the value of one of the covariates (in this case, altitude) at those sensor sites. The algorithm is tending to have to obtain values of similar magnitude around a given collector, due to the nature of the dataset.

3.5.4 Testing results- Expected Values

Further inspection of the algorithm results can be made using expected values of algorithm performance. Figures 3.9 and 3.10 plot the expected maximum determinant value against

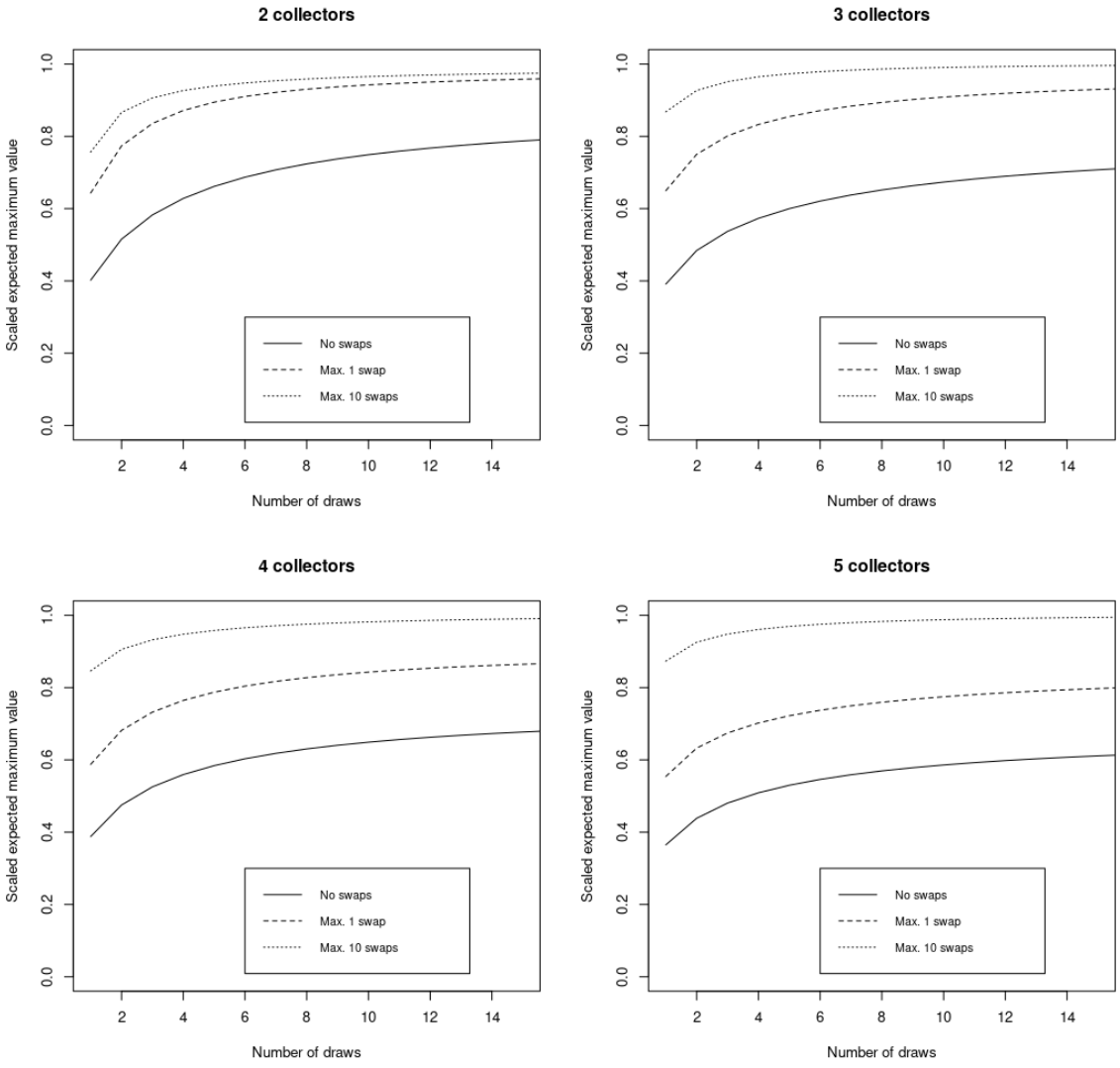

Figure 3.9: Simulated environmental datasets, expected values

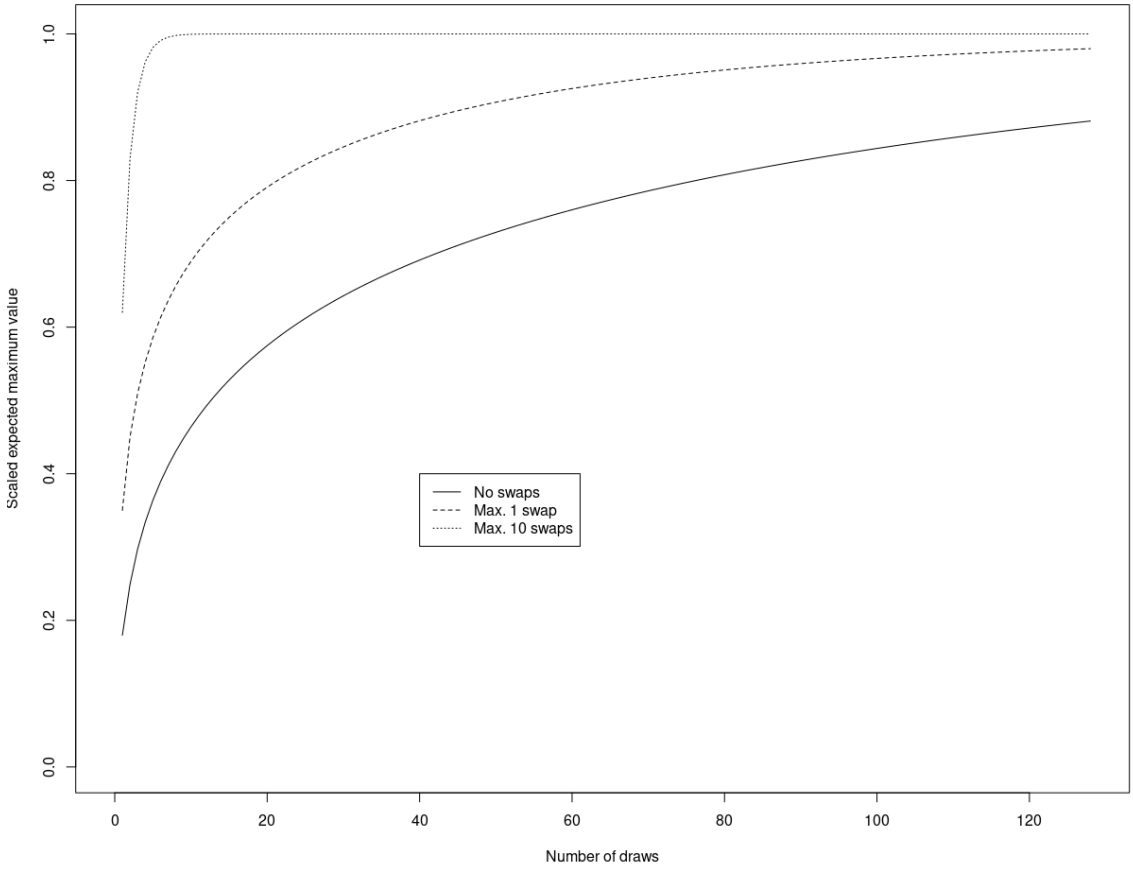

Figure 3.10: Application dataset, expected maximum determinant achieved with increasing number of algorithm random starts (draws)

the number of algorithm random starts, for the cases of performing zero, one or ten collector swaps, with the simulated Type B (environmental) and application datasets. The expected maximum value here is calculated in the following manner: let R be the number of algorithm runs (random starts). The random variable g of interest is the maximum of R independent algorithm run results. If $F(X)$ is the cumulative distribution function for the (scaled) determinant value for a single algorithm run, $G(X) = F(X)^R$, and $E[X] = \Sigma X g(X)$. Using empirical values from the algorithm runs to produce a histogram leads to the values shown in the plots. The importance of performing a sufficient number of collector swaps can be seen; performing only two swaps can lead to very slow increase in maximal value achieved with respect to the number of algorithm random starts. If the number of collector swaps is substantially below the number of collectors being used, and the latter is in some sense large, then supposing the probability of randomly choosing good collectors in the starting design is small, the probability of attaining a good design after the collector swaps will also be small. It would seem logical to perform at least as many collector swaps as there are collectors, to allow in theory for each collector to be swapped once, in an attempt to keep this probability as high as possible.

3.5.5 Testing results- Prediction Variance

The prediction variance of a point $x \in \mathcal{A}$ in the design space is proportional to $x'M^{-1}x$ (Cook and Nachtsheim, 1980). While our method takes as its criterion the minimisation of variance of the parameter estimates, the motivating problem still makes it desirable that prediction variances be low. A brief check confirms at least that these variances are improved upon by the nested Fedorov algorithm, in comparison with the naïve and random sampling methods described in section 3.2.1. A method for depicting prediction variance over a design space is described in Zahran et al. (2003). Prediction variance is considered cumulatively on its range over the design space, plotted against the proportion of the design space having this prediction variance or lower. Thus, for the maximum possible prediction variance, 100% of the space will have this value or lower, while the plot shape illustrates the degree to which prediction variance varies across points of the space. A flatter line suggests more homogeneity of prediction variance. This method is used for the construction of plots

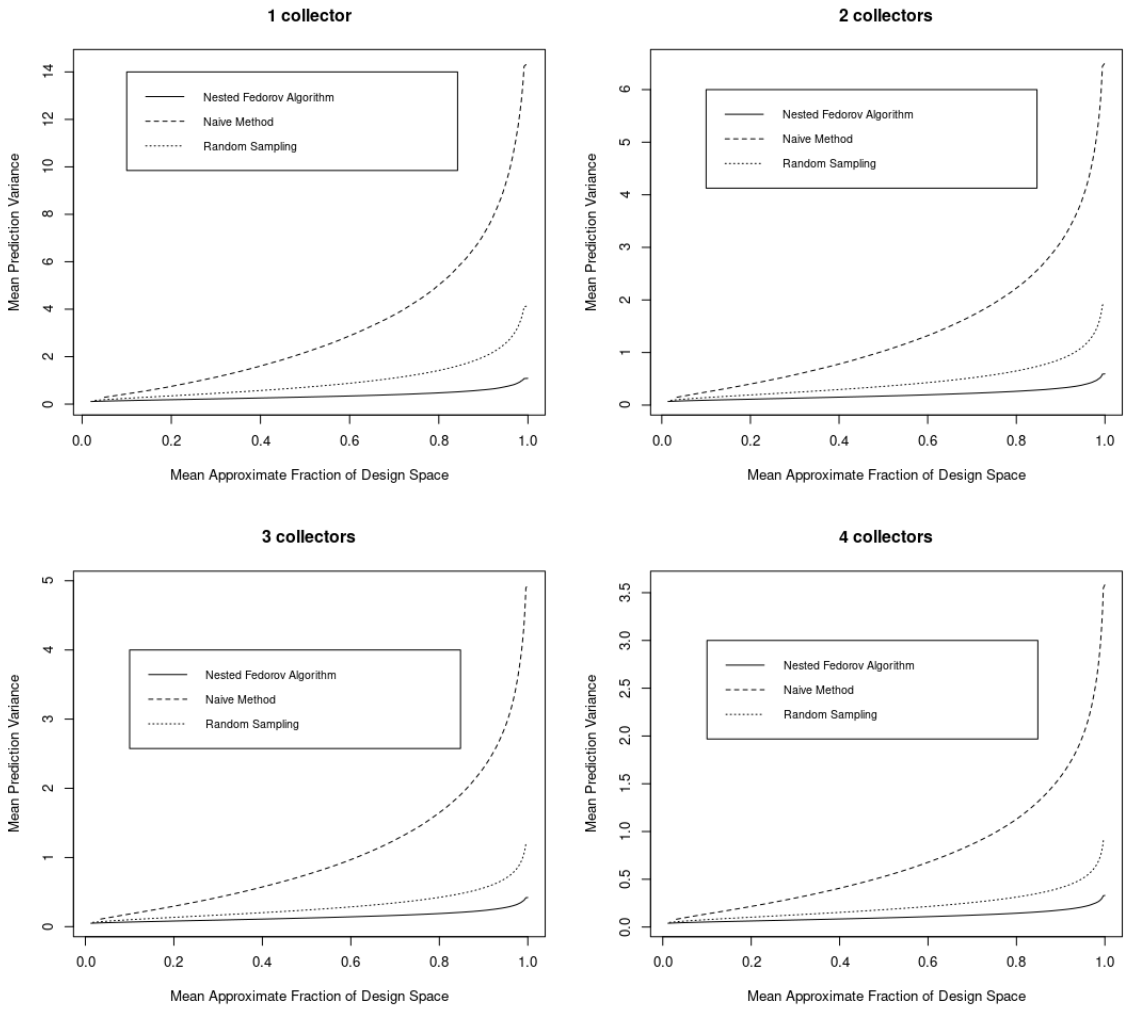

Figure 3.11: Fraction of Design Space plots for simulated datasets (simulated environment)

in Figure 3.11 (for Simulated Type B) All the plots demonstrate lower prediction variances for the Nested Fedorov algorithm's designs than for the naïve or random sample designs, for all points in the design space, as well as a much more homogeneous variance across the space. Similar behaviours are observed for the Simulated Type A and application data (not plotted here).

3.6 Discussion

The work presented in this chapter represents the first investigation of a specific, thus far under-explored problem in the area of optimal experiment design. An adapted version of the Fedorov algorithm has been shown to perform well in correctly identifying a set of design points corresponding to a known D-optimal Ξ_H design in small-scale testing scenarios, and to converge to at least local maxima when applied to further simulated datasets and to the application dataset supplied by the industrial partner. It consistently performs better than randomly sampling designs to satisfy the nesting structure requirement, or than using a naïve point-expectation method. The need to run a sufficient number of random starts, and collector swaps, has been identified, as this will help to avoid local maxima in the design space- it should be noted here that while the standard Fedorov algorithm can become trapped in local maxima dependent upon starting design, the Nested Fedorov algorithm can in addition to this become trapped due to the combinatorics induced by the partition; that is, not only due to the shape of the space that exists with any given, fixed choice of collectors. It is suggested that the numbers of random starts and collector swaps may realistically be dictated by computational constraints, but that it is advisable to be prepared to perform at least as many collector swaps as there are collectors. In addition to constraining the total number of collectors, it may be practically desirable to limit the total number of sensors per collector to be within some range (neither too high nor too low). The Nested Fedorov algorithm implementation in R has this functionality included, which is described in the Appendix A. Some obvious next steps are: to adapt the algorithm to account for non-normal errors; and to explore adaptations of the method to allow for changing the sensor locations during the course of the experiment in order to increase the information they

provide. Regarding the first of these points, in many applications the sensor observations may be in the form of counts meaning a Poisson model would be more appropriate. Russell et al. (2009) note the difficulty of finding optimal designs in the standard non-nested Poisson scenario, if there is no prior knowledge of the parameter values.

Chapter 4

Poisson Models

4.1 Introduction

In Chapter 2, we solved the problem of identifying designs with high determinants with respect to the D-optimality criterion, satisfying the nesting constraint, and under the assumptions of a linear model. In practice, the assumption of responses generated from a Normal distribution may not be appropriate. In order to address this potential shortcoming, the manner of calculation of the optimality criterion must be adjusted.

We assume that a Poisson count model is required, with canonical log-link. Then the D-optimality criterion can be expressed in terms of X^TWX where W is a diagonal matrix with non-zero entries equal to the Poisson means for the corresponding points (Yanping et al, 2006). The motivation here is that one potential application of our methods is in flood prediction, and one appropriate response variable to model is the count of flood events- whatever these may be defined to be- over time. Thus a Poisson count process would be likely to be an appropriate choice to model the response. This chapter will describe adaptations to our methodology for dealing with this scenario.

4.2 The model

Assume that a Poisson count model is being used in the experiment; that is, we have a Generalized Linear Model with a log-link function, and each of the n observations y_i has a

Poisson distribution with rate parameter λ_i where

$$\log(\lambda_i) = \sum_{j=1}^{p} (f_j(x_i)\beta_j).$$

As with the Gaussian model, we have p unknown coefficients β_j whose values are to be estimated by the experiment. The information matrix is $X^T W X$ where W is the diagonal matrix:

$$W = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & & \\ \vdots & & \ddots & \vdots \\ 0 & & \dots & \lambda_n \end{bmatrix}$$

where $\lambda_i = \exp(\sum_{j=1}^{p} f_j(x_j)\beta_j)$ for $i = 1, \dots, n$. The situation has clearly changed from that of the standard linear model.

If an experiment is to be designed for the use of a Poisson model, values are required for the parameters λ_j in order to compute $X^T W X$, and the design of a D-optimal experiment can no longer be conducted solely on the basis of the attribute values chosen at certain potential design point locations. If the parameters were known, the method would closely resemble that used for the linear model case in Chapter 3, but with the D-optimality criterion being based upon $|X^T W X|$ rather than $|X^T X|$. However, in practice the parameters are unknown. If we assume that the design is to be constructed entirely before any data are collected, they cannot be estimated from the data. It then becomes necessary to design the experiment accounting for possible plausible values of these unknown model parameters. One possible method in this regard is to account for ranges of possible values of these parameters; either via a grid search, or through a more sophisticated method that uses probability distributions of the parameters. The latter option is clearly preferable from a theoretical viewpoint, given it allows for incorporation of prior knowledge of the parameters' behaviours- indeed, even if such knowledge is unavailable, an assumption of multivariate normality for the parameters' distribution might give more realistic results than a uniform grid.

The issue of computation must also be considered in comparing the two options; however,

since either would involve evaluation of a determinant at each of the chosen points, the amount of computation would be likely to be comparable for a given number of points. Thus it would seem sensible to use a method based on prior distributions for the parameters. In particular, Gaussian quadrature (Golub and Welsch, 1969) is a useful tool for achieving this.

4.3 Gaussian quadrature

Gaussian quadrature is a method for estimating definite integrals through a weighted sum of the function evaluated at certain points; the weights and locations of these points are chosen based upon roots of suitable orthogonal polynomials (Golub and Welsch, 1969). The general form of the approximation is:

$$\int_D f(x)dx \approx \sum_{i=1}^{d} w_i f(x_i)$$

over a region of integration D, with weights w_i and quadrature points x_i for a chosen number of quadrature points d. Assuming that the design is for a model with more than one parameter (which is true for all applications in our target area), our problem requires that we optimise the information determinant over a multidimensional integral estimate (using a multivariate distribution for the combined model parameters). This is a Bayesian optimal design procedure, and the quantity to be optimised is of the form: $E(\log(\det(I(\theta)))$, where I is the information matrix of the design. (Khuri et al, 2006). Gaussian quadrature has been used before for Bayesian optimal experiment design, for example in Bliemer et al. (2009), Gotwalt et al (2009) and Goos et al (2018).

4.3.1 Procedure

With the assumption of a Poisson model, the procedure for calculating a determinant of a given design becomes the following:

1. Use the quadrature points as inputs to calculate weight matrices- each weight matrix will correspond to one quadrature point, and its (diagonal) entries will correspond to the design points being used (with the β's in W above provided by the quadrature

point- W depends upon the unknown parameters in the model).

2. From the weight matrices, calculate the logged determinant of X^TWX for each quadrature point, using the design X;

3. Use the quadrature weights to sum the obtained values.

The resultant sum can be used in optimisation to find a design; for example, in an adapted version of the Standard Fedorov algorithm, a value of this sum for each potential design (resulting from each potential swap) will be considered, and the design corresponding to the highest value of the estimated determinant chosen. A toy example of obtaining this sum is as follows: suppose $n = 4$ and $p = 2$; that is, four design points are to be chosen, and there are two parameters (so that each design point will be of length two). Suppose also that three quadrature points are to be used per parameter. This means that there will be $3^p = 3^2$ quadrature points overall. Based upon whatever multivariate distribution is assumed for the two parameters, a point and weight is obtained for each quadrature point. We take Q as a 9×2 matrix of the quadrature points, and we take X as the 4×2 matrix of the design points. We calculate $\exp(QX^T)$. Each row of the resultant 9×4 matrix provides the diagonal for one 4×4 diagonal weight matrix. There will then be nine values of $|X^TWX|$, one for each weight matrix. Using the nine quadrature weight values, a weighted sum of the logs of these values is computed. This value is then used in optimisation for the choice of design points.

4.3.2 Distribution of parameters

In the absence of any specific knowledge, a sensible prior assumption is that the parameters follow a multivariate Gaussian distribution, given its ubiquity and utility (Rencher and Christensen (2012) p91). A particular version of the Gaussian quadrature method, Gauss-Hermite quadrature, estimates an integral of a function multiplied by e^{-x^2}, comparison of which with a Gaussian density function shows its suitability for the estimation of the expectation of a Gaussian-distributed variable. A set of quadrature points (and weights) can be obtained for a univariate Gaussian variable, and then used to correspond to a multivariate distribution by duplicating/combining, rotating, scaling and translating the points. In this

way, an estimate may be obtained of the integral of interest over a multivariate Gaussian distribution of the unknown parameters. There are several algorithms for computing Gauss-Hermite quadrature points, among them the Golub-Welsch algorithm (Golub and Welsch, 1969), which is employed in the current work.

4.3.3 Number of points

The number of points in each dimension of the multivariate distribution (that is, the number of points with respect to each parameter of the model) is a variable that must be chosen carefully. Note that if the routine for conversion to a multivariate distribution described above is to be followed, it will be constant for all parameters. An increase in the number of quadrature points per dimension clearly brings with it an exponential increase in the number of determinants that must be evaluated- for p predictors and q quadrature points per dimension, we would have q^p points at which to evaluate a determinant. Since the eventual goal is, as in previous chapters, to obtain a D-optimal design using an exchange algorithm, it is necessary to perform q^p determinant calculations repeatedly, for every potential swap to be evaluated- this can quickly cause the amount of computation required to become unmanageable. However, choose too few points and the approximation afforded by the quadrature method becomes unacceptably poor. As a bare minimum one would expect to have to use three points per dimension, in order to account somewhat for the shape of the Gaussian distribution, with the use of further points up to a practical limit determined by computing power available.

If a comparable amount of computing power to that used in the search for an optimal linear design were available in the search for an optimal Poisson-based design, a lower number of algorithmic runs and/or collector swaps would likely be possible.

4.3.4 Implications for algorithm

The use of a weight matrix necessitates in itself a substantial amount of alteration to the Nested Fedorov algorithm (Algorithm 1). The computation time increases dramatically when using a Poisson model in the implementation of the Nested Fedorov algorithm, as one would expect due to the necessity of performing the Gauss-Hermite quadrature for each

potential change to the candidate design. The methods described in Section 3.3 may be adapted to the update of the determinant and inverse of M when M is defined not as $M = X^T X$, but as $M = X^T W X$ with W being a diagonal matrix with diagonal entries λ_i. This allows for the use of the computational shortcuts. Since $M = X^T W X$, which we can consider as $M = X^T(WX)$, where WX consists in multiplying each row of the design matrix by its corresponding weight, the matrix of interest with design point x_i inserted and design point x_j removed becomes $M + x_i x_i' \lambda_i - x_j x_j' \lambda_j$, and the only adjustment required to the statements of Proposition 3.3.1 and Corollary 3.3.2 is the substitution of $x_i' \lambda_i$ for x_i', and $x_j' \lambda_j$ for x_j'. This is only possible because the weight matrix is diagonal. Suppose that the design has points of length p, and d points are used per quadrature dimension. Then in order to implement the quadrature for the Poisson case and use the computational shortcuts, d^p separate M matrices and associated determinants are required. These must not only be computed, but stored in parallel in order to be individually updated. This can potentially lead to storage issues for large values of d and p. There is an additional reason for caution in the use of the shortcut. It can lead to problems in numerical accuracy depending upon the sizes of values present in the weight matrices involved. In particular, small discrepancies can arise between computed matrix determinants/inverses, and the determinants/inverses obtained as a result of matrix linear transformations. These discrepancies can compound over time, with the error propagation potentially leading to differences in designs chosen with and without use of the shortcut.

4.4 Results

The three main types of dataset used in the linear case (Chapter 2) are used again here for evaluating the algorithm's performance in identifying Poisson-based designs. That is,

- Type A Simulated: Uniform sampling to augment (hide) specified designs; in this case, separate optimal designs for both the linear and Poisson-model case can be generated, to enable comparisons.
- Type B Simulated: Sampling based on a simulated grid with spatial correlation.

- Application Data: the same application dataset that was used for the linear model testing.

As in the linear model case, repetitions of each type of simulated dataset are generated, and the proposed Nested Fedorov algorithm run repeatedly upon each. The following sections describe the characteristics of the simulations performed for each dataset type. In all cases, we assume a prior multivariate Normal distribution for the p parameters, $\beta = (\beta_1, \ldots, \beta_p)$:

$$\beta \sim \mathcal{N}_p(\mu, \mathbf{\Sigma}),$$

where various values are assumed for the vector μ and the matrix $\mathbf{\Sigma}$ for the specific dataset scenarios.

4.4.1 Simulated Type A Data

Two main sets of Type A simulations are presented; the first, A_1, incorporating known optimal Poisson designs, and the second, A_2, without the presence of such designs. For A_1, comparisons are also made involving linear and Poisson known optimal designs. There are advantages to each of the two approaches. The method for construction of known D-optimal Poisson designs given in Ford et al (1992) makes it possible to construct a design that would be optimal given the true model parameters. Suppose there are p predictors $x_1, x_2, \ldots, x_p$, with parameters $\beta_1, \beta_2, \ldots, \beta_p$, and the predictors are each on the range $[l, u]$. The known D-optimal design is constructed beginning with one point $x*$ with highest possible expectation given the parameter values. This will be a p-vector with ith entry the lower bound l or upper bound u, according to whether the corresponding parameter value is negative or positive respectively. In addition to this point, p further points are included in the design. For i in $1, \ldots, p$, the ith of these additional points is identical to $x*$ with the exception that the ith entry becomes $l - 2/\beta_i$ if $\beta_i < 0$, or $u - 2/\beta_i$ if $\beta_i > 0$. This is subject to the condition that $|\beta_i(u - l)| \geq 2$ for all β_i. However, it should be noted this does not mean the design would necessarily be optimal, given the prior distribution of the parameters, as we are optimising a Bayesian criterion. Subsequent reference to a known optimal design in the Poisson model case should be read with this in mind. An additional issue is that

this construction of the known Poisson design imposes the constraint $n = p + 1$; that is, that the number of points in the design is one greater than the number of predictors. Since computation times increase dramatically with the number of predictors, particularly when using the quadrature method, it can become impractical to perform simulations involving as many predictors as desired under such a constraint. It is therefore helpful to perform a second set of simulations without the known Poisson designs present, permitting a higher number of design points relative to the number of predictors.

Simulated Type A_1- with known Poisson design

A known Poisson-optimal design is constructed with $n = p+1$ points, following the method in Ford et al (1992). The points that would form a known linear-optimal design are constructed factorially, constituting 2^p points. To the union of these two sets of points, uniformly sampled random points are appended, in the same manner as for the generation of Type A simulated data for the linear designs in Chapter 3. The resultant dataset has a hierarchy structure imposed in such a way that the known Poisson-optimal design is accessible through a choice of (at most) the appropriate number of collectors, and all the points in the linearly optimal design are accessible through a single choice of collector. This latter condition is imposed because there are a high number of possible choices of subsets of the 2^p points that make up the linearly optimal design that prove optimal for a linear design of only $p + 1$ points. In order to test the algorithm's ability to find an optimal Poisson design as opposed to a linear one, all of these possibilities are made accessible to it as 'linear decoys'. This procedure is followed for simulations with the characteristics:

- Use 4 predictors in the model, including the intercept;
- Stipulate that the algorithm is to choose 5 sensor locations from a pool of 80 potential design points, and 5 collectors to be chosen from a pool of 20 (note that 5 sensor locations corresponds to $p + 1$ for the size of the known optimal design);
- Use 5 quadrature points per dimension;
- Use covariances of 0.5 and variances of 1 for the parameters, so that the means are the only potential discrepancies between the parameters' prior distributions;

- Use three different sets of prior means for the parameters in turn: $\mu = c(-0.6, -0.6, 0.6, 0.6)$, $\mu = (0.6, 0.6, 0.6, 0.6)$, and $\mu = (-0.6, -0.6, -0.6, -0.6)$. Note that non-intercept predictors for the Type A simulations have no inherent meaningful ordering, given their identical variances and covariances;

- Generate 32 simulated datasets for each set of prior means, to assess convergence for different sets of available points subject to those conditions; and

- Apply the algorithm 16 times per simulated dataset, to assess convergence from different starting points (parallel processing is utilised here).

The Nested Fedorov algorithm is run on the datasets, using the Gaussian quadrature to find a Poisson design. The algorithm is also run independently each time with the objective of finding a linear model, and the output design fed into a third, Poisson run of the algorithm to serve as a 'warm start'. The results for the first and third of these runs- that is, for the runs for which a Poisson design was the goal- are shown side-by-side in Figures 4.1, 4.2 and 4.3, with the random-start results depicted in white and the warm-start ones in grey. The values are D-efficiencies assuming the multivariate Normal distribution used by the algorithm for the quadrature, and are relative to the known optimal Poisson design. In particular, the exponentiated weighted sum of logged determinants of the X^TWX for each quadrature point is calculated for a given design, and the ratio of this value to the corresponding value for the optimal Poisson design is taken to the power $1/p$. The most striking features of the results are that:

- The algorithm (with either warm or random starts) tends to converge after 3 collector swaps in most cases.

- The variance of the D-efficiencies is similar for all the scenarios;

- The warm starts result in better, or comparable, designs than random starts, and do so with a lower number of collector swaps.

The algorithm appears to choose clearly between points belonging to the known optimal Poisson design, and to the known optimal linear design set; a mean of 4.75 of the 5 design

points come from the optimal linear design when searching for a linear design, whilst when searching for a Poisson design, the algorithm chooses a mean of 3.6 of the 5 points from the optimal Poisson design with a warm start, and a mean of 3.975 with a random start. It is not surprising that these means are not closer to 5, given that there is duplication of certain vertices between the linearly optimal and Poisson-optimal design. A majority of points on average are chosen from the appropriate optimal design given the type of design the algorithm is instructed to find.

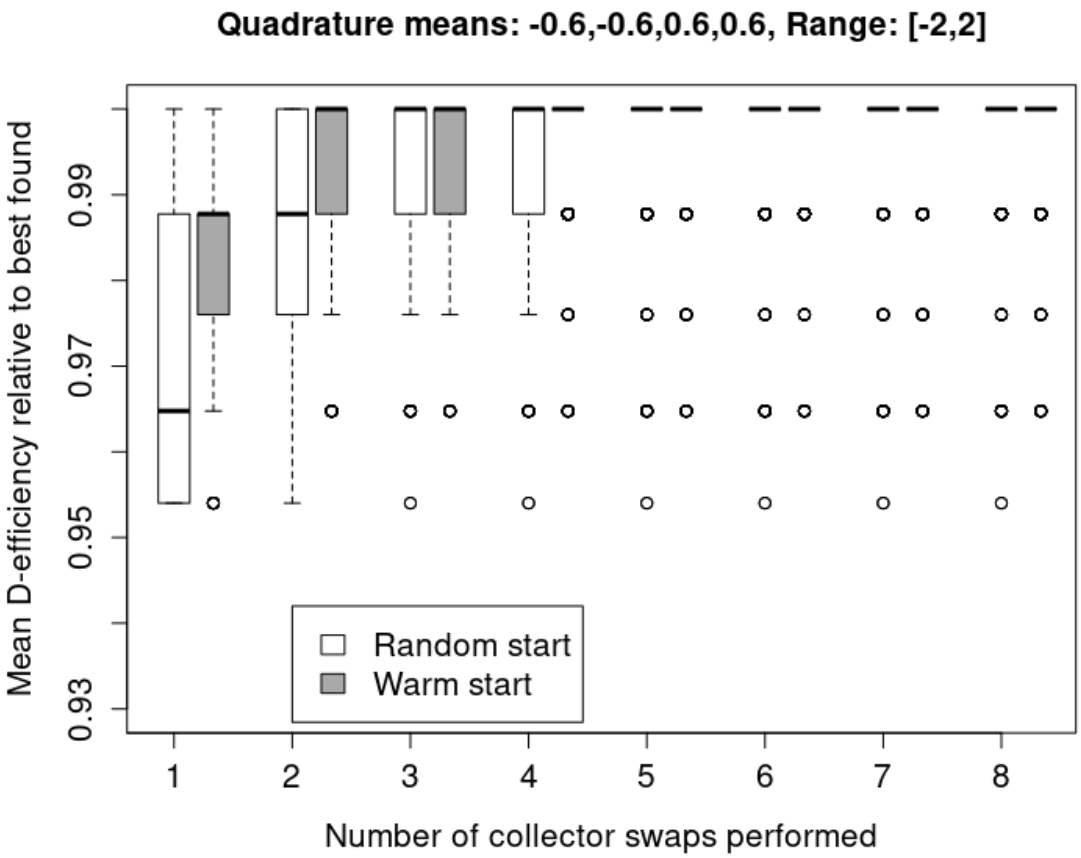

Figure 4.1: Simulation Type A_1, conditions 1. D-efficiencies achieved relative to known optimal Poisson design.

Simulated Type A_2- without known Poisson design

150 Uniformly sampled points are generated on a specified range. In contrast with the Type A_1 simulations, no Poisson or linear optimal design is constructed. Again, as for Type A_1,

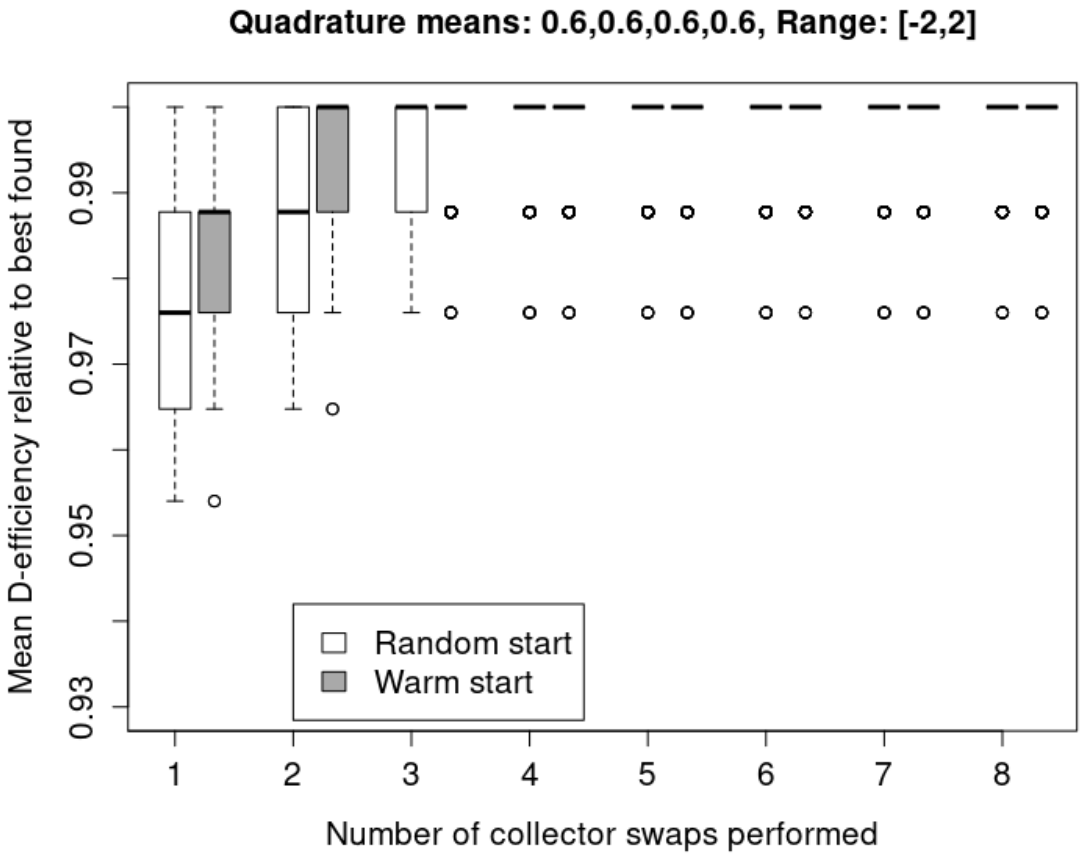

Figure 4.2: Simulation Type A_1, conditions 2. D-efficiencies achieved relative to known optimal Poisson design.

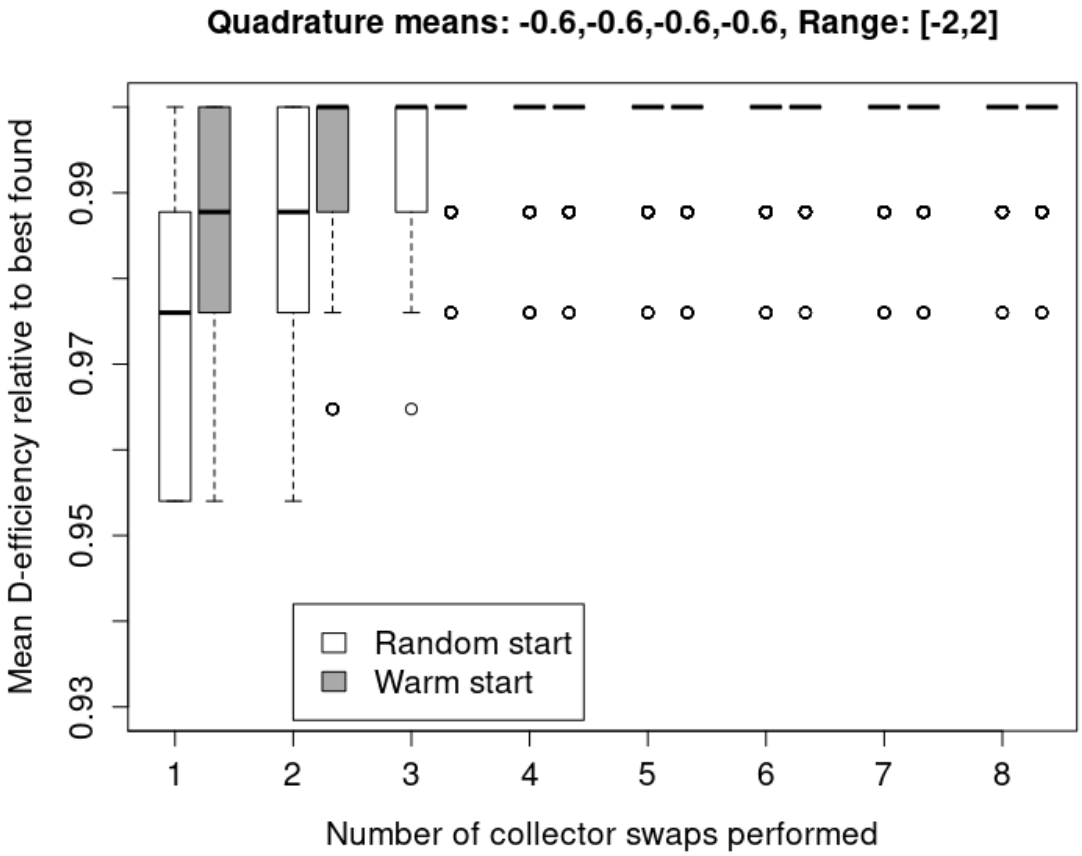

Figure 4.3: Simulation Type A_1, conditions 3. D-efficiencies achieved relative to known optimal Poisson design.

two starts are compared; firstly, the Nested Fedorov algorithm with random start is run to find a Poisson design, and secondly, warm start from the design identified from a linear model. The following characteristics are used in the simulations:

- Use 4 predictors in the model, including the intercept;
- Stipulate that the algorithm is to choose 16 sensor locations from a pool of 150 potential design points, and 5 collectors to be chosen from a pool of 20;
- Use 4 quadrature points per dimension;
- Use covariances of 0.2 and variances of 1 for the parameters, so that the means are the only potential discrepancies between the parameters' prior distributions.
- Use four different sets of prior means for the parameters in turn: $\mu = (0.1, 0.1, 0.1, 0.1)$, $\mu = (0.5, 0.5, 0.5, 0.5)$, $\mu = (0.1, 0.23, 0.37, 0.5)$, or $\mu = (0.5, 0.37, 0.23, 0.1)$. Note again that non-intercept predictors for the Type A simulations have no inherent meaningful ordering, given their identical variances and covariances.
- Generate 16 simulated datasets for each set of prior means, and for the ranges $[0, 0.5]$ and $[0, 1]$; and
- Apply the algorithm 16 times per simulated dataset (parallel processing is utilised here).

The results for the random and warm-start Poisson models are shown in Figures 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10 and 4.11, with the random-start results depicted in white and the warm-start ones in grey. As before, the values are D-efficiencies assuming the multivariate Normal distribution used by the algorithm for the quadrature, and are relative to the best design found on a by-dataset basis. The quadrature means are listed with the value corresponding to the intercept last. Observations on the results include that:

- The algorithm (with either random or warm starts) tends to converge after 2 collector swaps in most cases;
- The variance of the designs' D-efficiencies is greater when the datasets have range $[0, 0.5]$, rather than range $[0, 1]$;

- The warm starts tend to result in better, or comparable, designs in comparison with random starts, and do so with a lower number of collector swaps than required by the random-start runs.

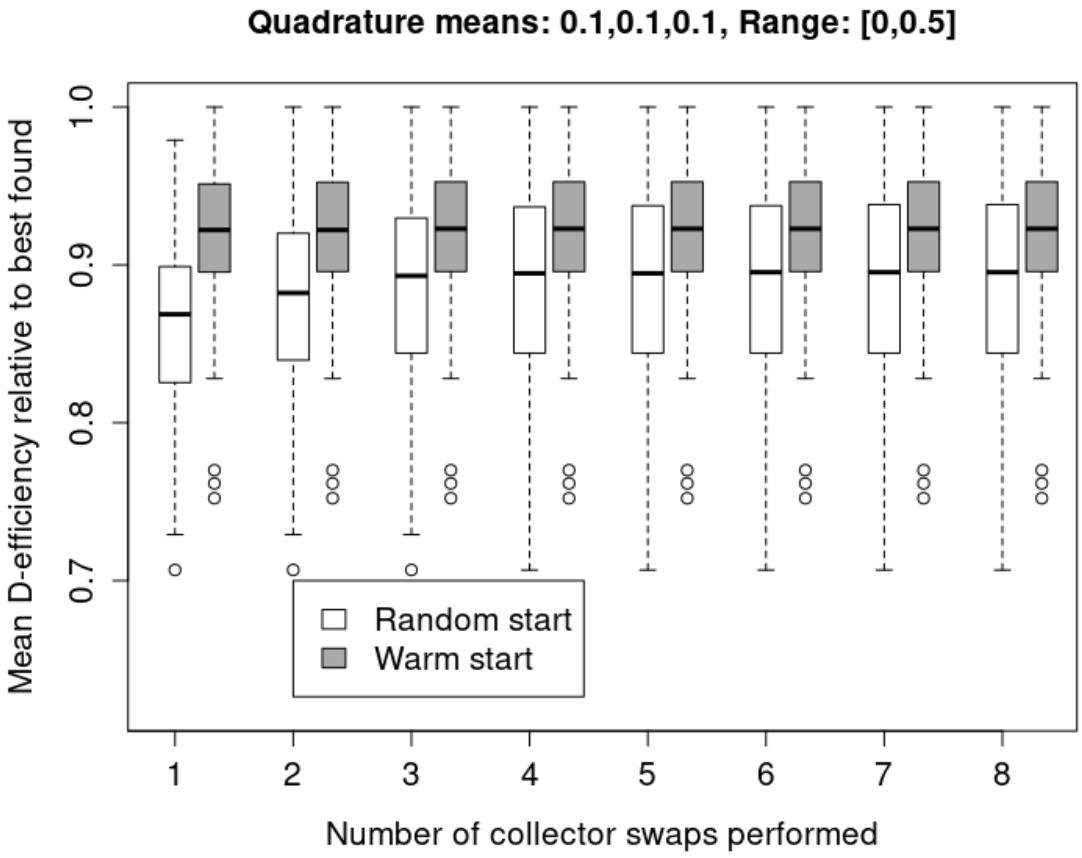

Figure 4.4: Simulation Type A_2 plot 1: D-efficiencies achieved relative to best-found design.

4.4.2 Simulated Type B Data

As in Chapter 3, the Type B datasets differ from the Type A datasets in that they simulate a spatial correlation. The following characteristics are used:

- Use 3 predictors in the model, including the intercept;
- Stipulate that the algorithm is to choose 30 sensor locations from a pool of 300 potential design points, and 5 collectors to be chosen from a pool of 20;
- Use 5 quadrature points per dimension;

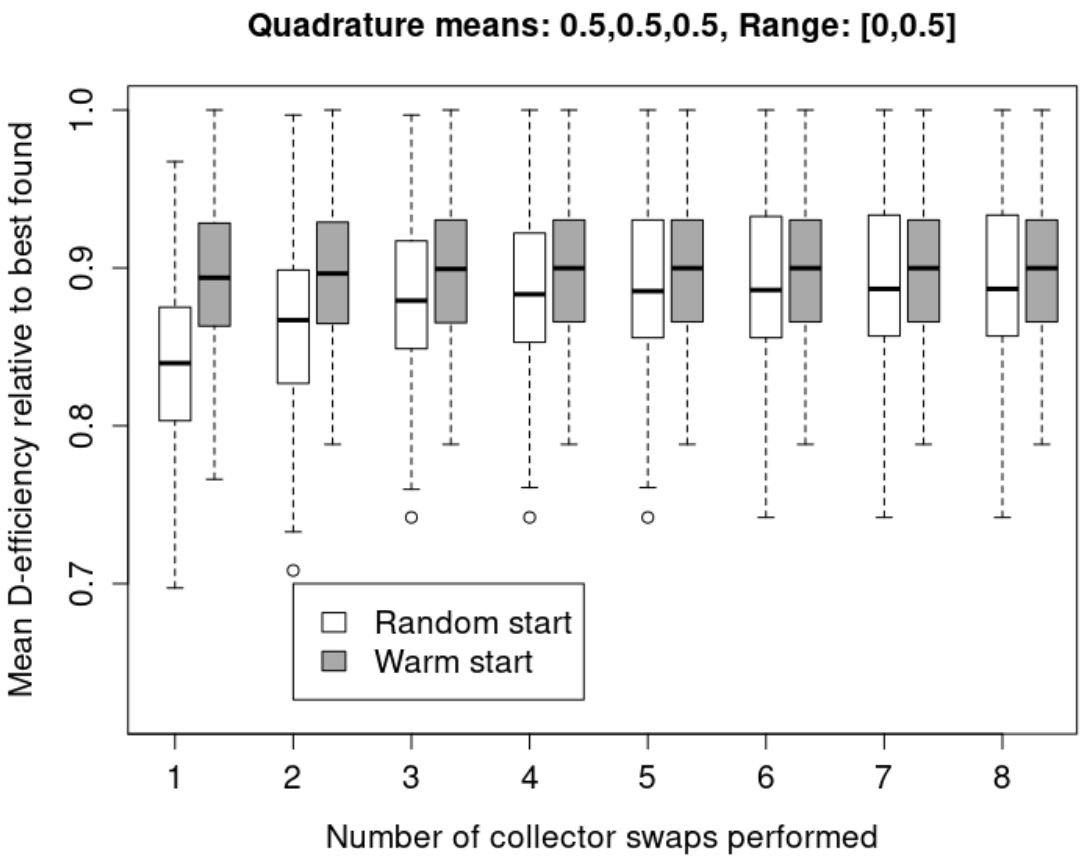

Figure 4.5: Simulation Type A_2 plot 2: D-efficiencies achieved relative to best-found design.

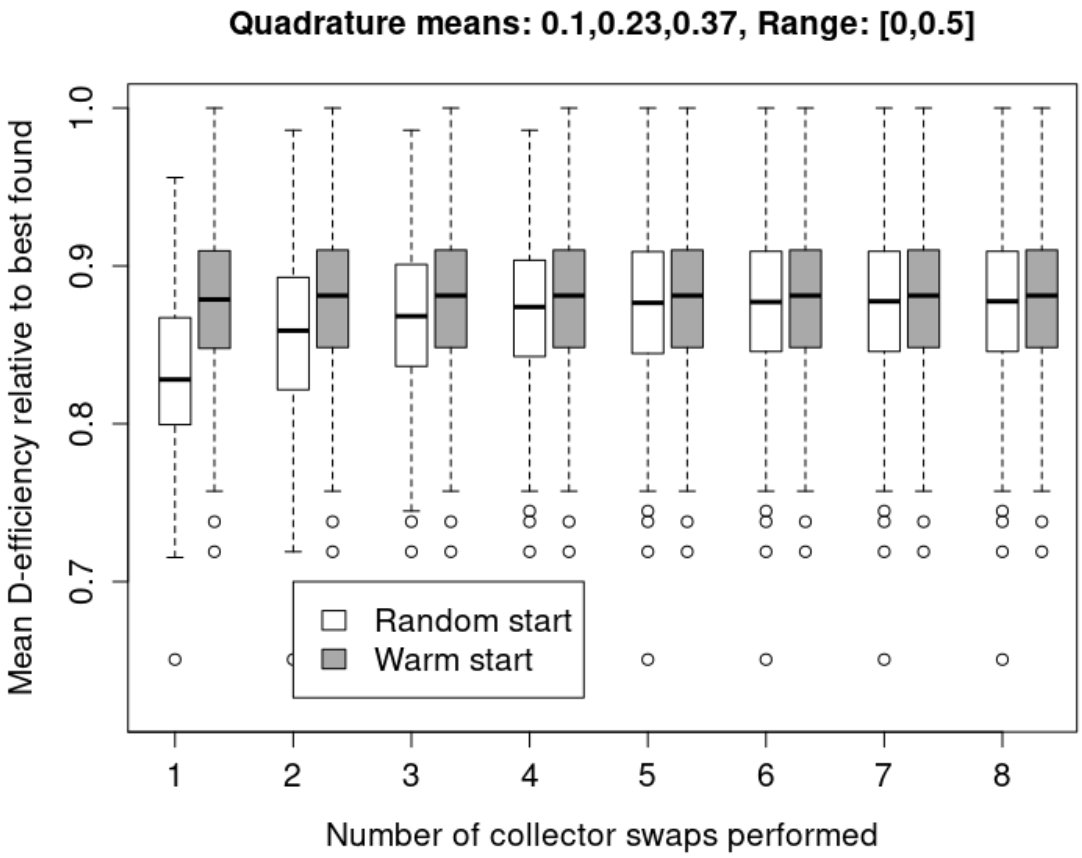

Figure 4.6: Simulation Type A_2 plot 3: D-efficiencies achieved relative to best-found design.

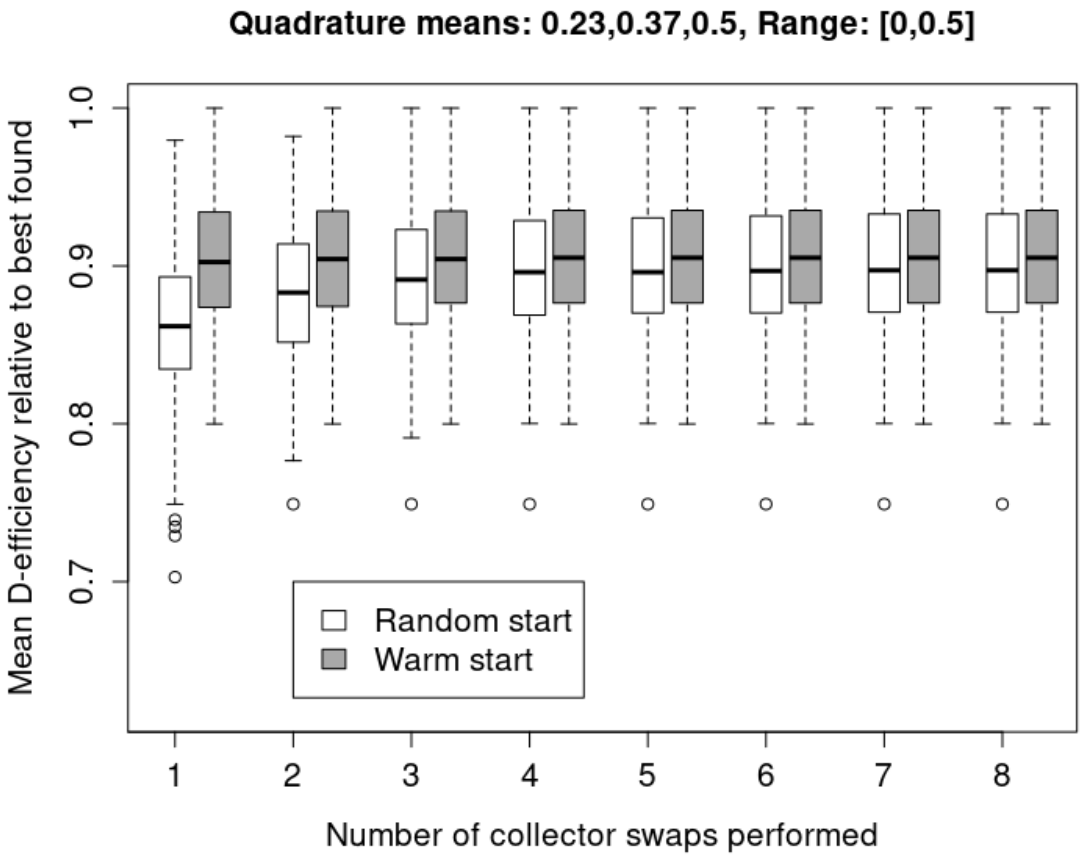

Figure 4.7: Simulation Type A_2 plot 4: D-efficiencies achieved relative to best-found design.

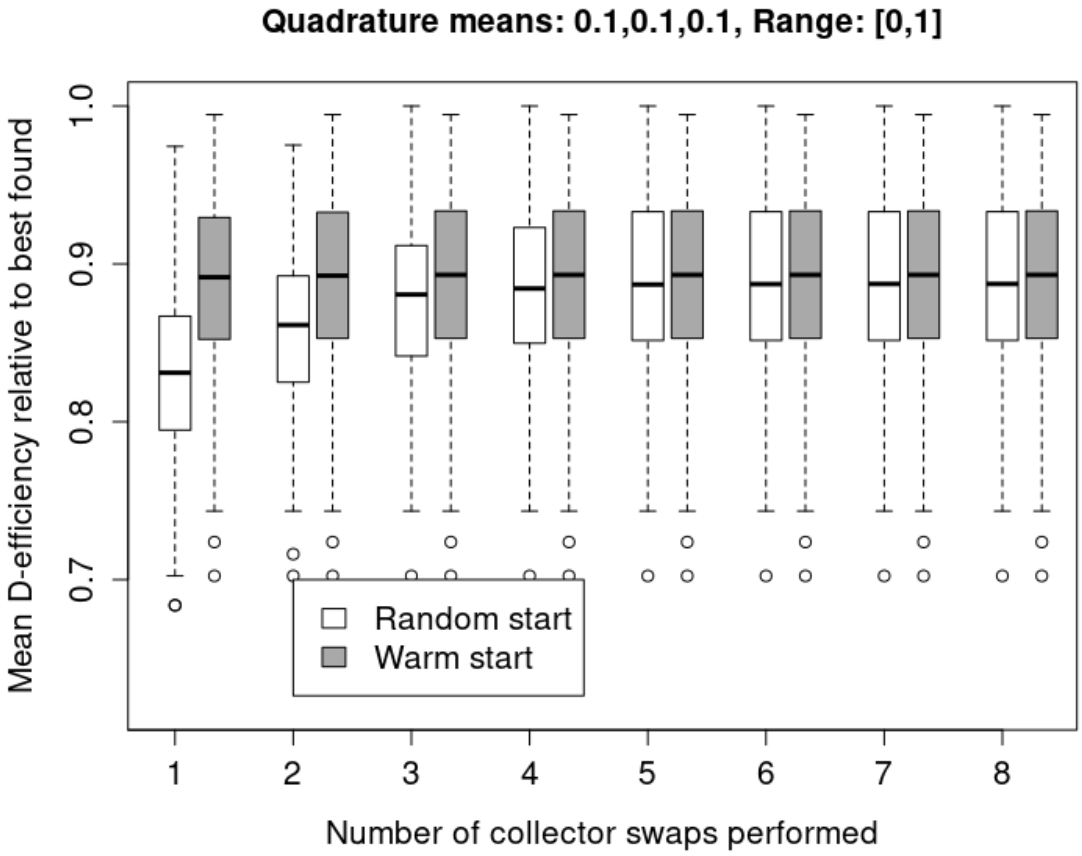

Figure 4.8: Simulation Type A_2 plot 5: D-efficiencies achieved relative to best-found design.

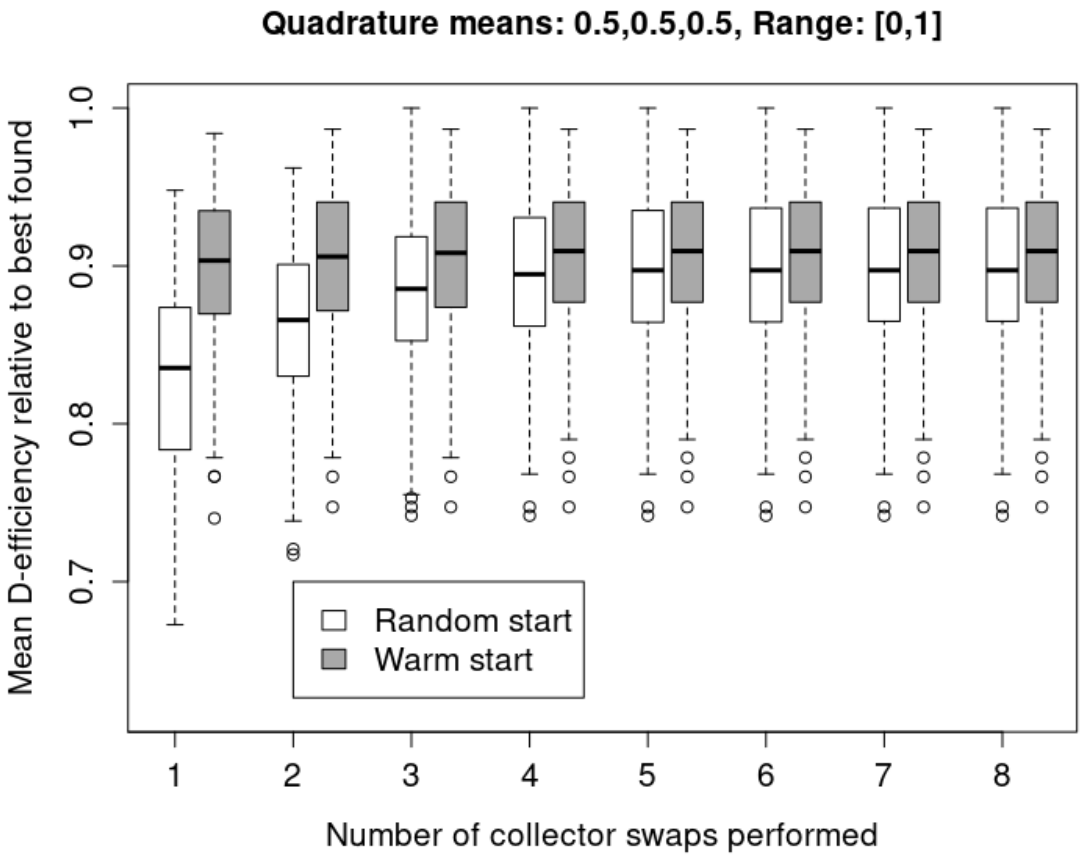

Figure 4.9: Simulation Type A_2 plot 6: D-efficiencies achieved relative to best-found design.

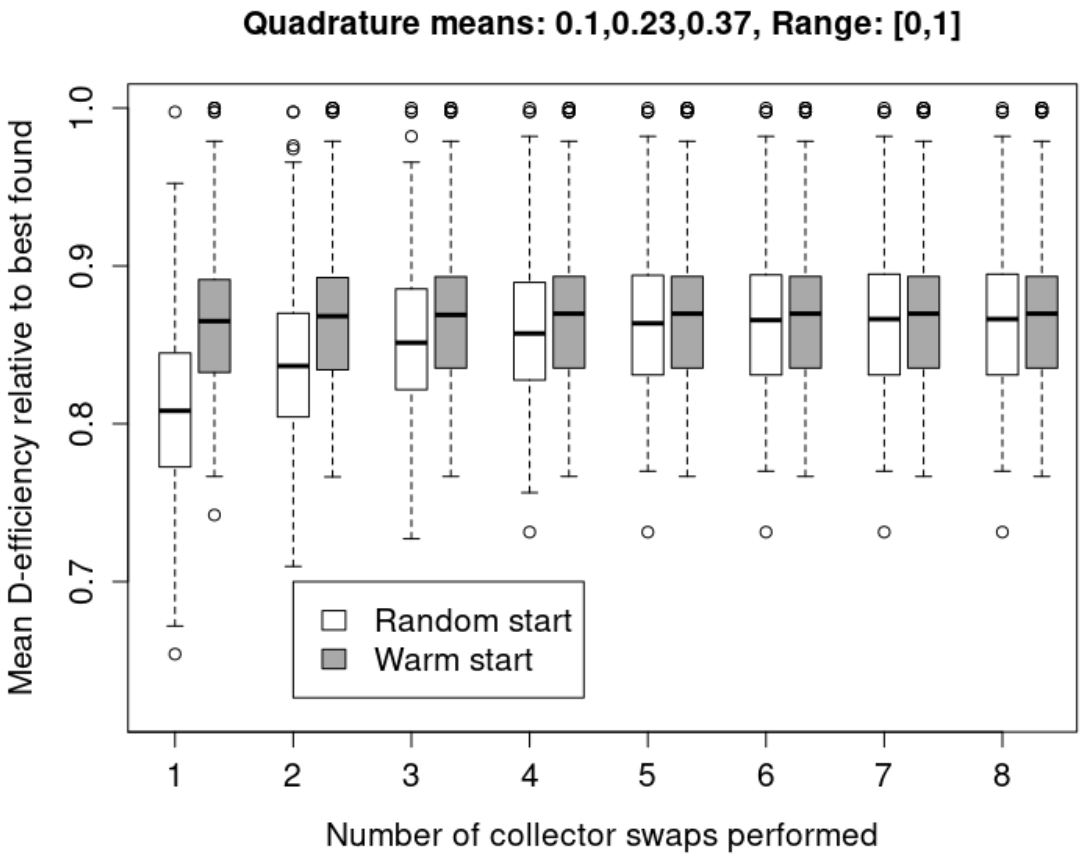

Figure 4.10: Simulation Type A_2 plot 7: D-efficiencies achieved relative to best-found design.

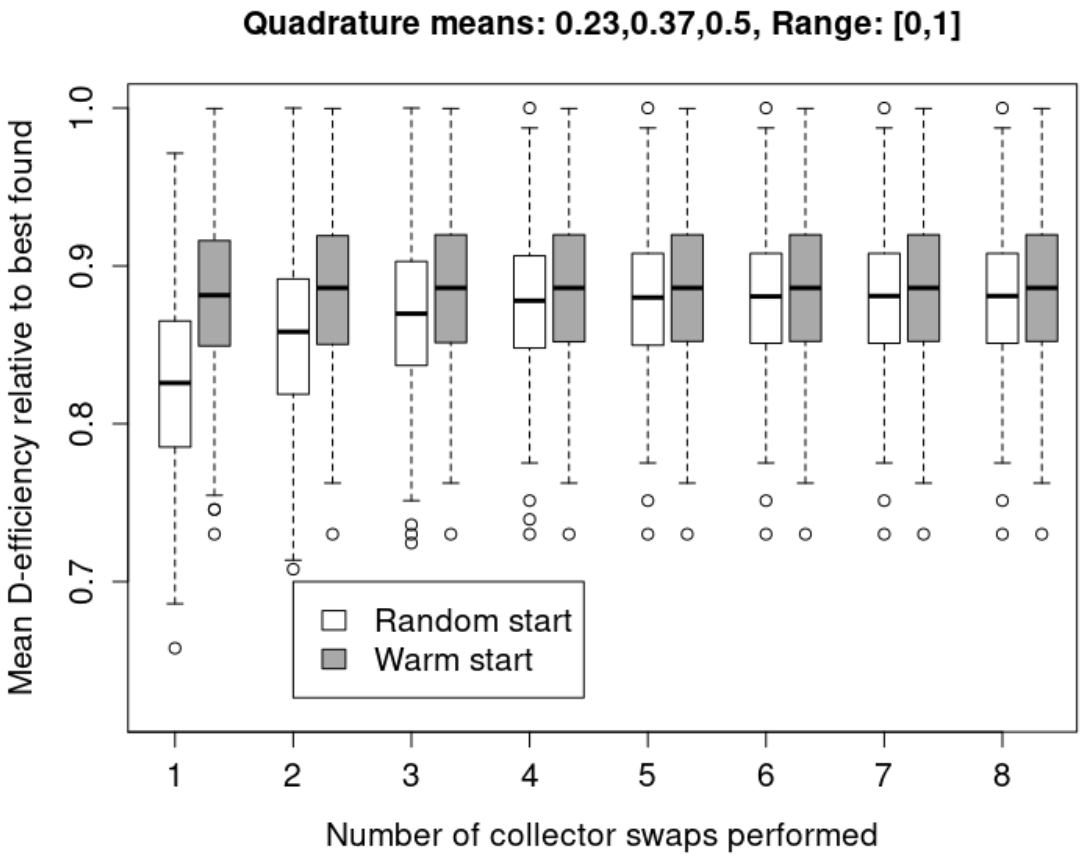

Figure 4.11: Simulation Type A_2 plot 8: D-efficiencies achieved relative to best-found design.

- Use covariances of 0.2 and variances of 1 for the parameters, so that the means are the only potential discrepancies between the parameters' prior distributions.
- Use two different sets of prior means for the parameters in turn: $\mu = c(0.1, 0.1, 0.1, 0.1)$ and $\mu = (0.5, 0.5, 0.5, 0.5)$. Note again that non-intercept predictors for the Type A simulations have no inherent meaningful ordering, given their identical variances and covariances.
- Generate 10 simulated datasets for each set of prior means; and
- Apply the algorithm 16 times per simulated dataset (parallel processing is utilised here).

Figures 4.12 and 4.13 show results for, as with the Type A simulations, both a random and a warm start for each collector swap. In both cases the algorithm converges after approximately five collector swaps, and benefits from the use of a warm start. It can be seen that the parameters' prior means affect the variance of the results, with a higher variance for the 0.1 means in comparison with the 0.5 means. It is possible that this can be explained by the 0.1 means causing a spanning of zero which induces greater uncertainty in the points' expectations. In practice, the prior means used will be informed by the context and cannot be controlled.

4.4.3 Application Data

The Nested Fedorov algorithm is applied, for Poisson designs, to the application dataset described in Section 3.4.3. The following characteristics are used:

- Use 3 predictors in the model, including the intercept. This means that each potential design point contains two attributes from the application dataset, and a 1 for the intercept;
- Stipulate that the algorithm is to choose 20 sensor locations from the available 2417 potential design points, and 5 collectors to be chosen from the 32 available collector sites;

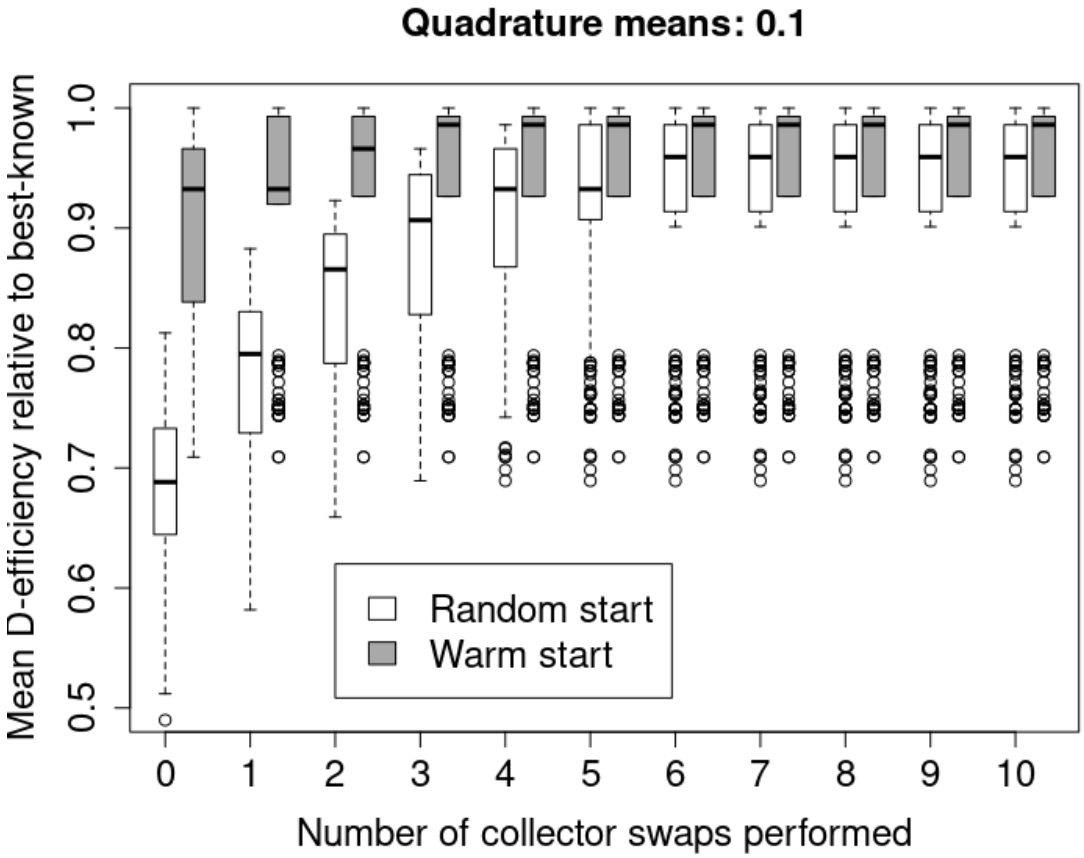

Figure 4.12: Simulation Type B plot 1: D-efficiencies achieved relative to best-found design.

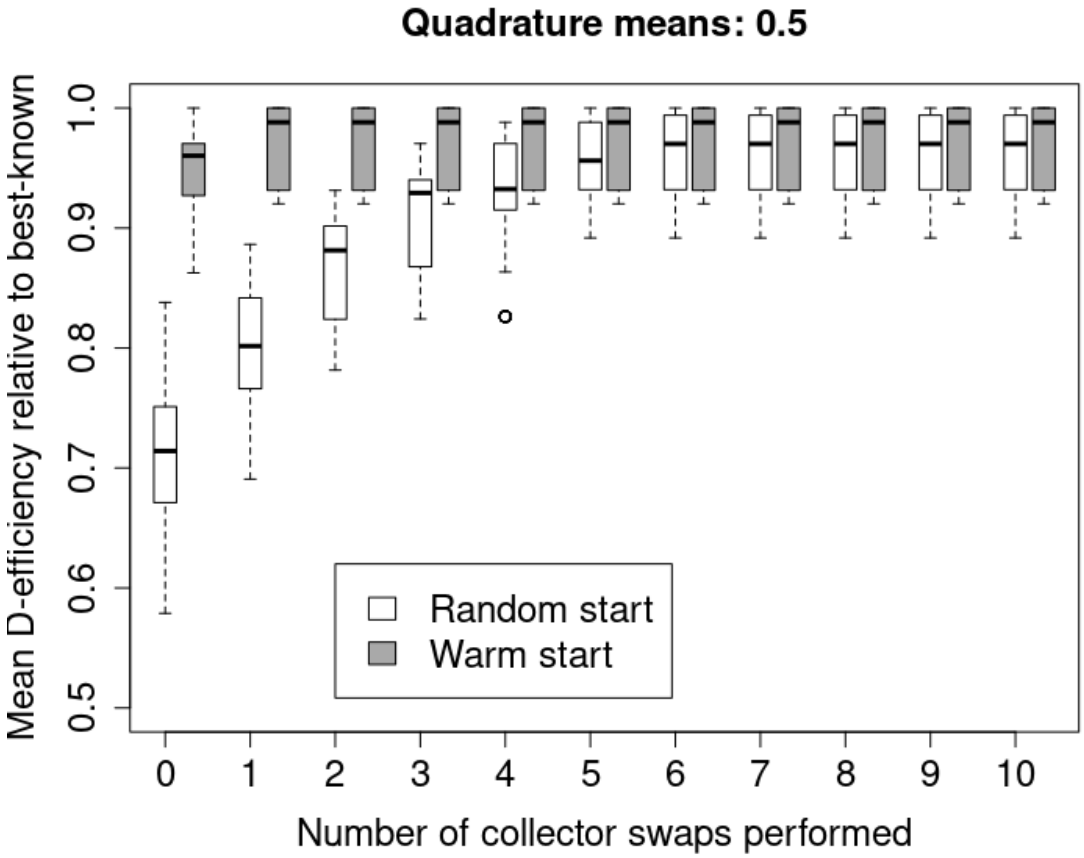

Figure 4.13: Simulation Type B plot 2: D-efficiencies achieved relative to best-found design.

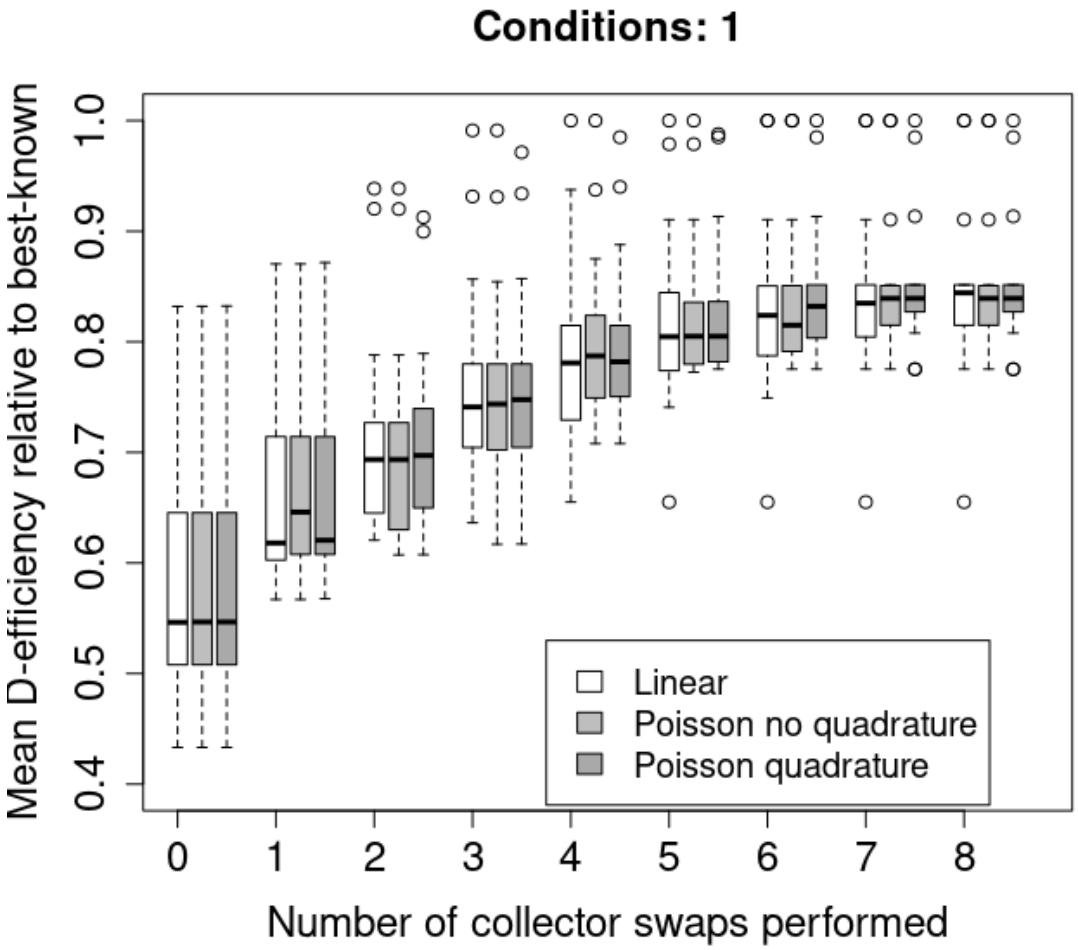

Figure 4.14: Application data results, conditions '1' (prior parameter covariances 0.5, prior means of parameters $(0.05, 0.1, 0.15)$, $\sqrt{2}$km grid)

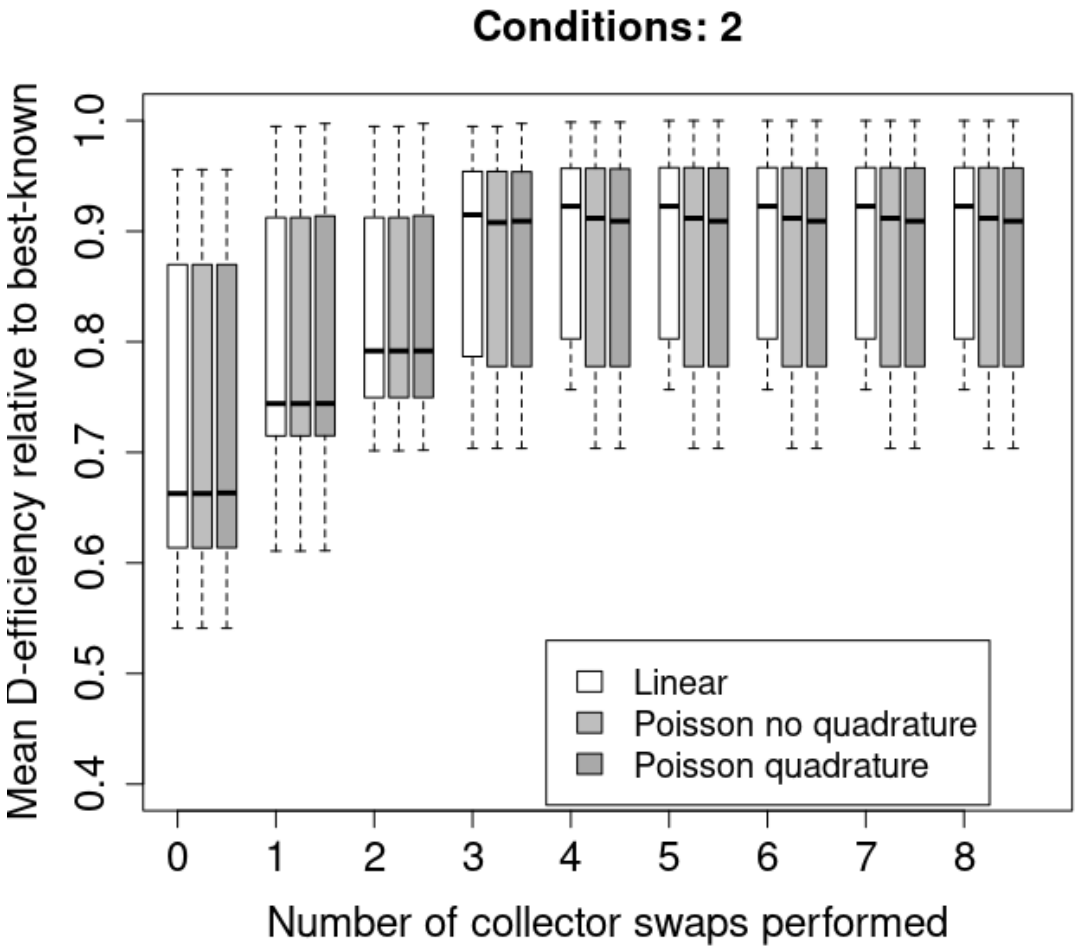

Figure 4.15: Application data results, conditions '2' (prior parameter covariances 0.5, prior means of parameters $(0.15, 0.1, 0.05)$, 2km grid)

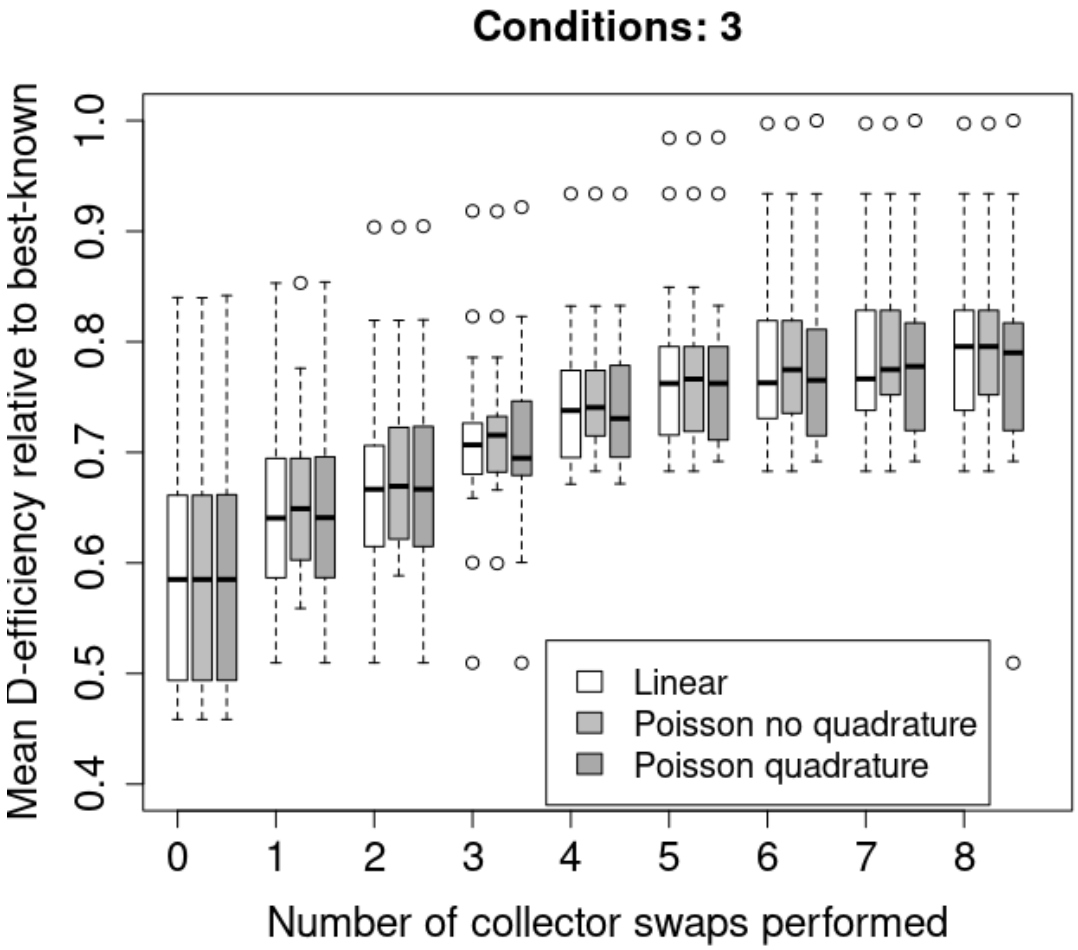

Figure 4.16: Application data results, conditions '3' (prior parameter covariances 0.5, prior means of parameters $(0.05, 0.1, 0.15,)$, $\sqrt{2}$km grid)

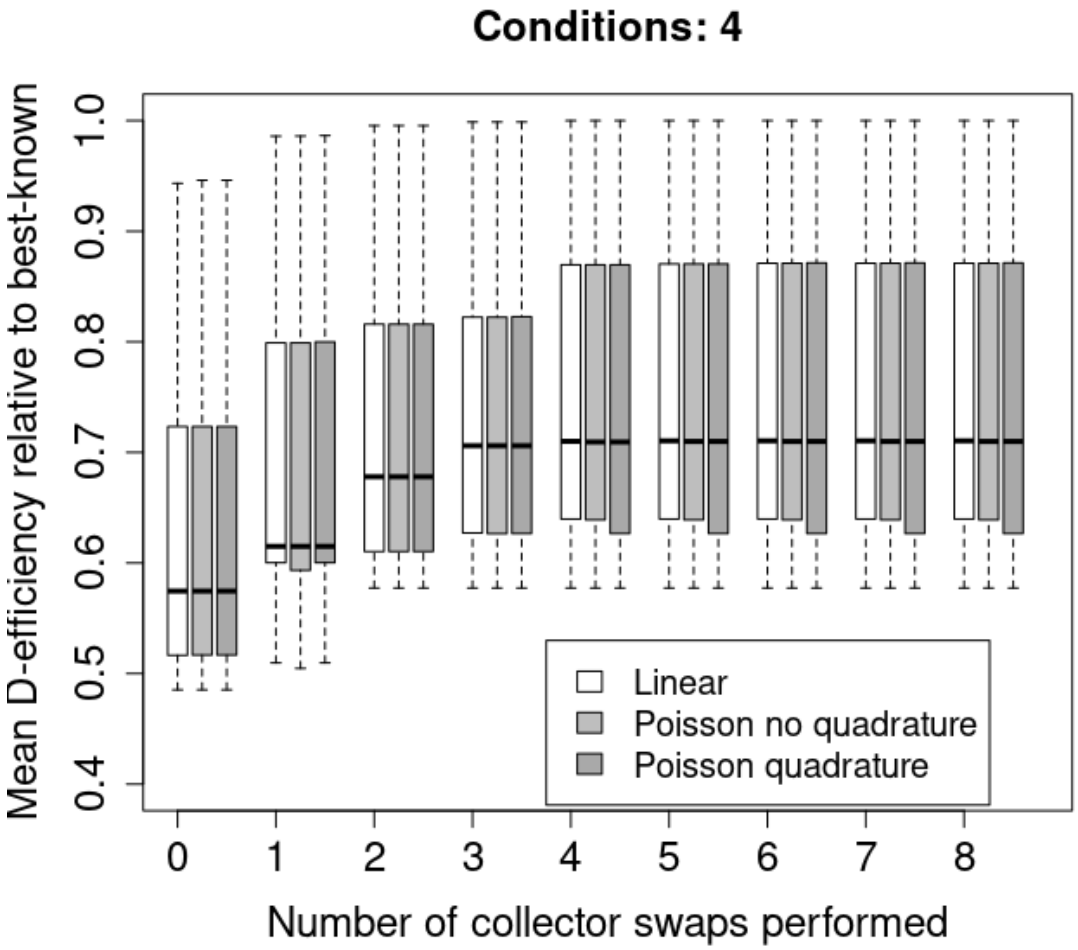

Figure 4.17: Application data results, conditions '4' (prior parameter covariances 0.5, prior means of parameters $(0.15, 0.1, 0.05)$, 2km grid)

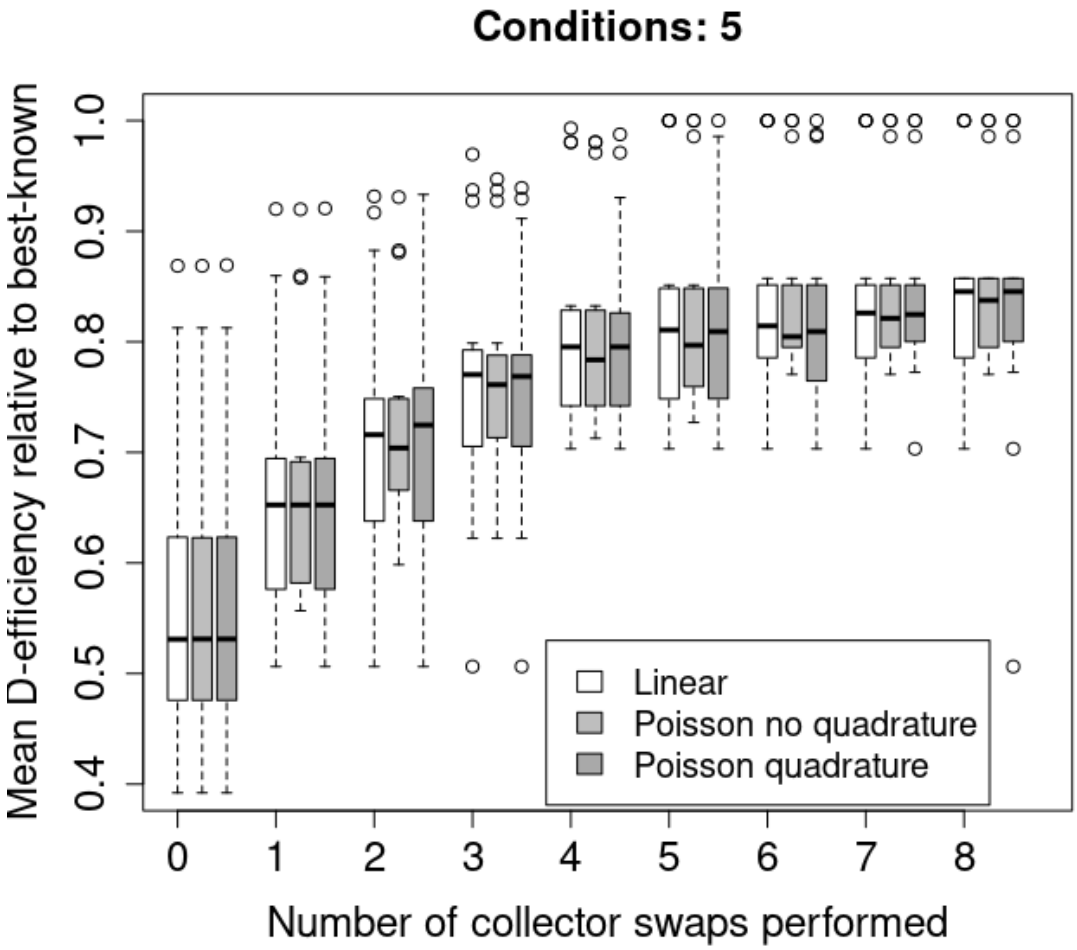

Figure 4.18: Application data results, conditions '5' (prior parameter covariances 0.1, prior means of parameters $(0.05, 0.1, 0.15,)$, $\sqrt{2}$km grid)

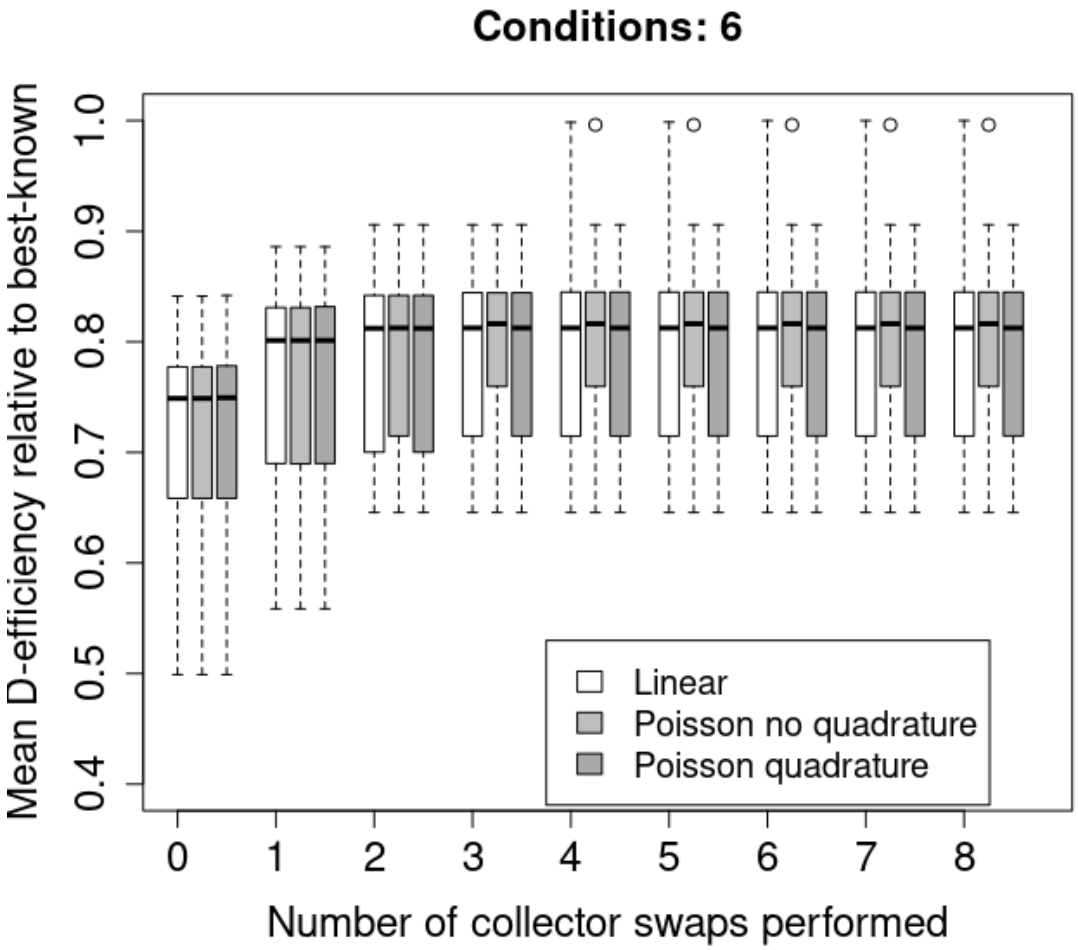

Figure 4.19: Application data results, conditions '6' (prior parameter covariances 0.1, prior means of parameters $(0.15, 0.1, 0.05)$, 2km grid)

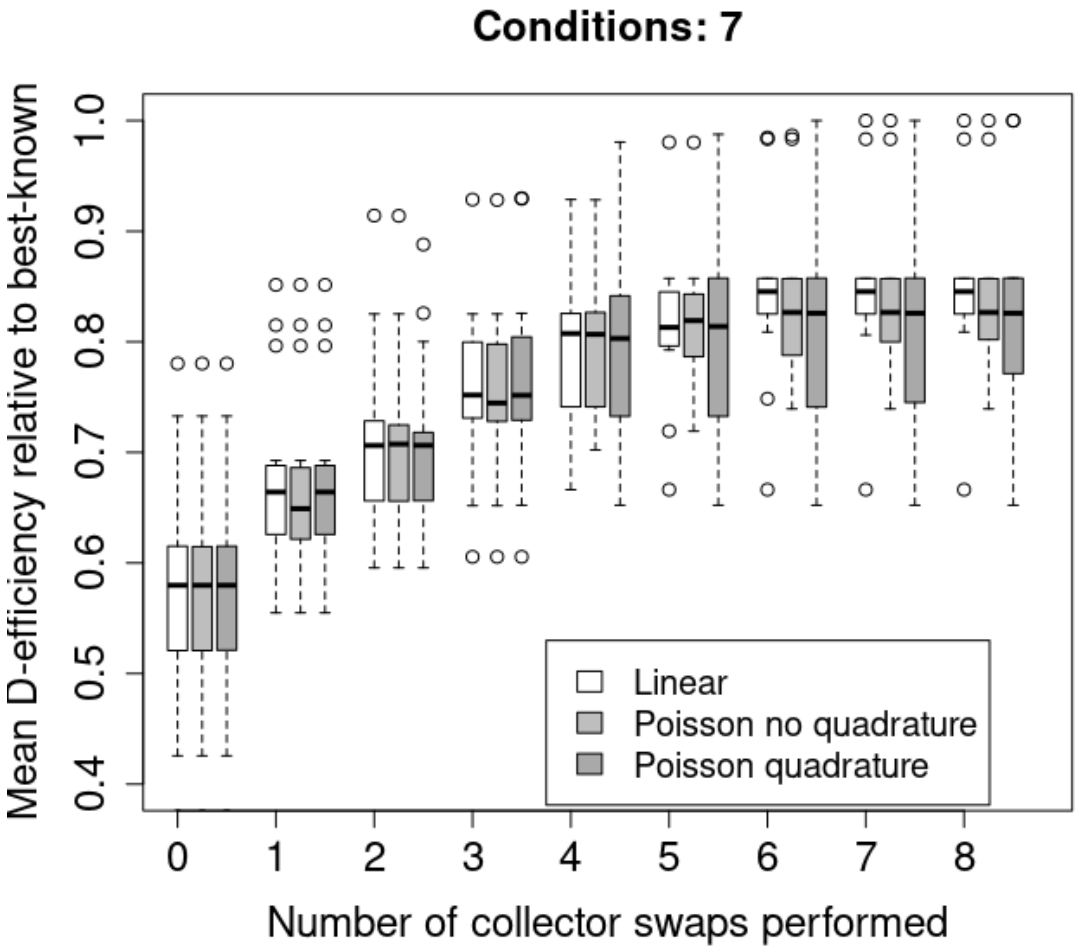

Figure 4.20: Application data results, conditions '7' (prior parameter covariances 0.1, prior means of parameters $(0.05, 0.1, 0.15,)$, $\sqrt{2}$km grid)

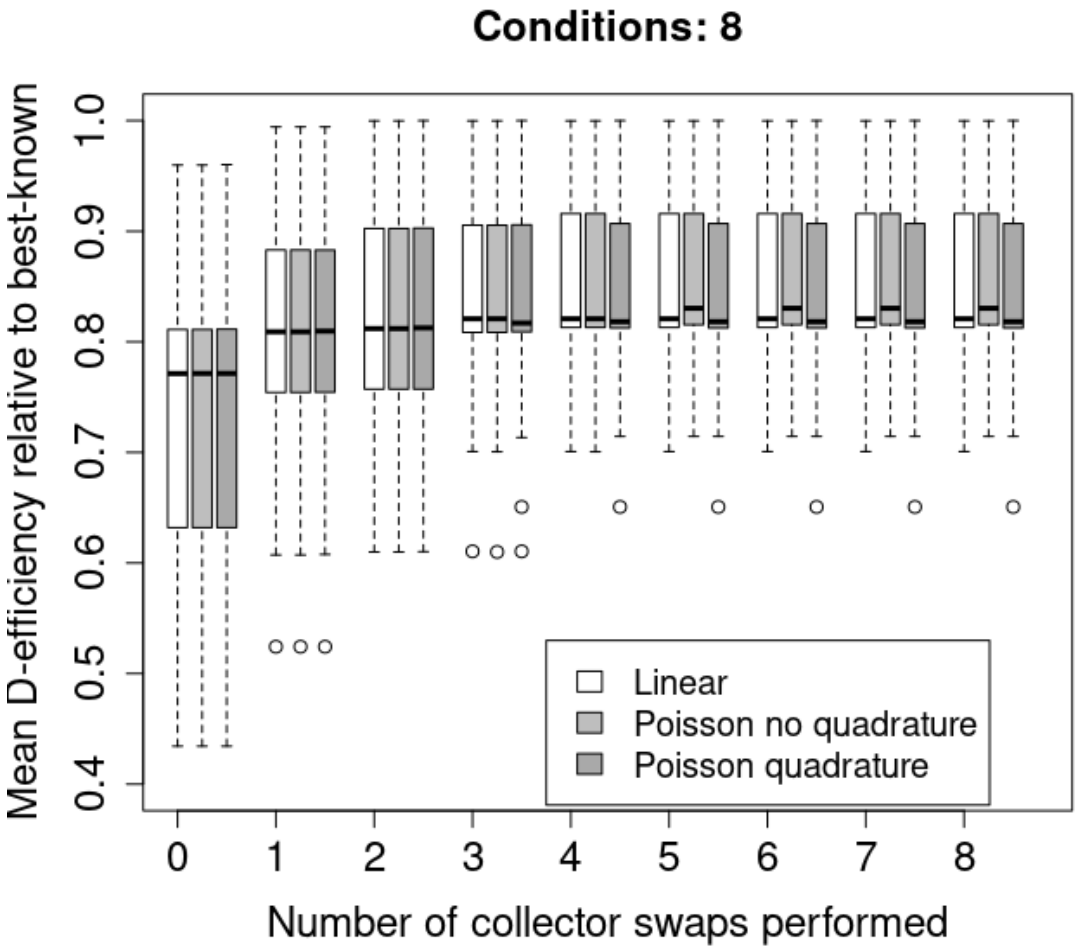

Figure 4.21: Application data results, conditions '8' (prior parameter covariances 0.1, prior means of parameters $(0.15, 0.1, 0.05)$, 2km grid)

- Set the parameters' prior means to $(0.05, 0.1, 0.15)$; and
- Use 5 quadrature points per dimension.

The following parameters are varied, for a total of eight sets of conditions:

- The covariances in the covariance matrix for the parameters' priors are set at 0.5 for the first four sets of conditions, and 0.1 for the last four;
- Two different allocations of potential sensor sites (design points) to potential collector sites are generated from the dataset; for odd numbered conditions, based upon a grid of $\sqrt{2} \times \sqrt{2}$km squares, and for even-numbered conditions with 2×2km squares.

Figures 4.14-4.21 give results for 16 algorithm repetitions (parallel processing is utilised here) for each of these eight sets of conditions. For each collector swap, three boxplots are shown side-by-side;

1. the first (white) for the Nested Fedorov algorithm, finding a Poisson design using the specified quadrature;
2. the second (pale grey) for the Nested Fedorov algorithm finding a linear design;
3. the third (dark grey) for the Nested Fedorov algorithm finding a Poisson design, but with the Poisson parameters assumed to be at their prior means (effectively a 'one-point quadature').

For each of the 16 repetitions within each condition, the same starting design (16 randomly generated) is used for each of the Nested Fedorov algorithm runs. The D-efficiencies are calculated relative to the best-found design in any of the three types of algorithm run, and are based upon determinants calculated using three times as many quadrature points per dimension ($3 \times 5 = 15$) as used in the actual algorithm runs. Thus even the linear-model algorithm runs have their results displayed in terms of the Poisson model, to enable comparison. The results demonstrate that a change in covariances from 0.1 to 0.5 for the quadrature results in decreases in variance of the achieved D-efficiencies. Also, a wider grid allocation of potential collector sites to potential sensor sites (more potential sensor sites

per potential collector), tends to cause higher variance in results, but slightly fewer required collector swaps for convergence. This may be due to the number of potential collector swaps possible at any given stage- when this is lower, one might expect a lower number of higher-gain swaps to occur in the process of convergence. Each potential swap can cause a higher increase in determinant (because of the higher number of potential sensor sites associated with each potential collector). Thus, it is easier for the algorithm to become trapped in a local maximum because of the restricted choice of collectors. However, in the process of so-doing the algorithm can make some high gains in determinant. The values for the first collector swap tend to be similar, suggesting that there may not be a large benefit in the extra computation required for 3 quadrature points per dimension over only 1 point. Additionally, for this simulated type at least, searching for a linearly optimal model gives a design that serves well for the Poisson case too, although this is dependent upon the assumptions made for the true parameter values. Recall that for the Type B simulations, some increases in efficiency were achieved near the beginning of warm starts. These increases would not have been possible through linear application of the Nested Fedorov algorithm alone since these warm starts began with the best design that could be found for the linear model. Recall also that for the Type A_1 simulations, the algorithm differentiated correctly between points belonging to the optimal linear and Poisson designs. This shows that the Nested Fedorov method does have the capacity to distinguish between designs that are optimal for linear and for Poisson scenarios, when such designs consist of different sets of points.

4.4.4 Discussion

The expansion of our aims to Poisson models increases the complexity of the optimisation with the use of Gauss-Hermite quadrature to estimate an integral for a multivariate distribution of the prior parameters. This significantly affects computation time, owing to the need to account for possible parameter values in the calculated determinants. Higher variances and higher expected values in the parameters' prior distributions tend to lead to lower variances in the achieved D-efficiences. Designs found for the linear model case tend to perform well for the Poisson model too, but some simulations have shown that the best-found linear designs can still be improved upon through application of the Nested Fedorov

algorithm again for a Poisson design.

Other distributions

The assumption of a Poisson model for the design may be too restrictive in terms of mean and variance, given its use of a single parameter to describe both of these. In reality, it may be that the variance for observed counts is greater than the mean (overdispersion) because of unavailability of important attributes at the sensor sites. The assumption could be refined to accommodate this through the use of a quasi-Poisson or negative Binomial model instead. A quasi-Poisson model allows, through the inclusion of an extra parameter, for a linear depedence of variance on mean rather than a simple equality (Ver Hoef and Boveng, 2007). Meanwhile, the negative Binomial goes a step further by allowing a quadratic relationship. With this latter option would come additional complexity in using the distribution and, inevitably, a greater demand for computing power. The information matrix would depend upon the regression parameters but also on the dispersion parameter. From an implementation perspective, if the counts were instead assumed to be Binomial/binary, and the link function to be logit, the information matrix would have the same $X^T W X$ form, but with diagonal elements $\mu_i * (1 - \mu_i)$ instead of the μ_i elements considered in this chapter. This would mean that the Poisson methodology could be adapted for the Binomial case with relative ease.

Discrete multivariate distribution of parameters

A key element of the methodology for the Poisson-based adaptations made in this chapter is the characterisation of the distribution of the prior parameters. One issue is the point at which this distribution is estimated. The approach taken in this chapter has been to specify a continuous multivariate Normal distribution for the prior parameters, and approximate the resulting integral using a quadrature. An alternative to this approach would be to assume a discrete distribution at the outset, and obtain an exact expectation from this discrete distribution, thus making the distribution estimation at an earlier point, in the manner of Cook et al (2004).

Developing the existing methodology

The results have demonstrated the potential utility of the linear-model Nested Fedorov algorithm in providing a warm start for the Poisson version. A possible method for further study is to develop the linear model option to allow 'backtracking' from a best found design to several designs of comparable, if slightly lower, determinant which may have several different choices of collectors, and using these as starting points for application of the Poisson Nested Fedorov algorithm.

Chapter 5

Design point reallocation

Chapter 3 described methodology developed for the generation of optimal linear designs for the nested sensor placement problem. Chapter 4 developed this work through the generation of Poisson model designs. This chapter addresses a different extension of the original problem by again considering the generation of linear models (as in Chapter 3), but with the inclusion of a time-dependent element. The original assumption was that the chosen design points were to form an exact design with precisely one usage of each point; that is, that a response variable would be recorded precisely once for each design point. In practice however, the experiment being conducted may continue over time in such a way that the response variable is recorded at more than one time point. For example, sensors are placed and collect data over time. The question then arises whether the optimal design(s) requires moves of sensors over time. The primary concern here is the possible relocation of sensors, and time effects are not considered. It should also be noted that the methodoloy uses the D-criterion to design a linear model for all time points prior to the experiment, so that data from one time point are not used to inform the design at subsequent time points. The discussion of Poisson models in Section 5.6.3 notes the possibility of using collected data to inform the design for later time points.

5.1 The temporal element in the Sensor Placement Problem

The sensor placement problem may be taken as an example to illustrate this idea with a motivation. The assumption in Chapter 3 is that the response variable being recorded is of such a nature that its value must be assessed over some sufficiently long, but defined, time period at each sensor location. The accumulation of material in a roadside gully is an example of such a response variable. The introduction of the time element makes possible the move of a given sensor at some point in time to a new location. For instance, ideally there would be the option to move any given sensor on any given day. However, it is not feasible to allow sensors to be moved on this individual basis, and the notion of time is therefore restricted to a lower granularity, with a batch approach to sensor moves. For instance, the unit of time epochs might be a month, a quarter or a year. Every epoch, the whole set of sensors are candidates to be moved to new locations, although each one might alternatively remain where it is. The experiment then accommodates data from multiple epochs. There is the possibility of moving some or all of the sensors whilst keeping the collectors fixed, or of moving both sensors and collectors. Practicalities dictate that any moves be conducted together over a short time period at the end of each epoch, as this minimises resources required for deinstallation, transport and reinstallation. Let T be the number of time epochs permitted by the available experimental resources. There is an assumption that each epoch would make possible the recording of the response variable at a maximum of n points (n sensors), in the same manner as the fixed n points of the static single-epoch arrangement; thus, a total of nT responses are to be recorded during the course of the experiment. The problem then becomes one of choosing:

1. The collector sites at each epoch.

2. The design points (sensor sites) at each epoch.

3. The reallocation of sensors and collectors at the beginning of each epoch subsequent to the first one; this reallocation should attempt to minimise the total cost of moving the requisite equipment, if such a cost is to be considered, although this minimisation may compete with the optimality requirement of the design.

These choices may equivalently be expressed with primary emphasis on the holistic design choices, rather than on the changes between epochs:

1. The collector sites to be used collectively across all epochs;

2. The design points to be used collectively across all epochs;

3. The allocation of these two sets of choices to individual epochs.

This distinction is made for a practical reason. Suppose that it is not possible to guarantee that the experiment will in fact be conducted for the entirety of its intended timespan, for instance owing to financial constraints or the risk of equipment malfunction. This consideration, if entertained, makes it prudent to take a relatively greedy approach, prioritising information gain in earlier epochs over the minimisation of equipment relocation costs during later epochs. Whatever the emphasis in making these choices, the procedure may now be formalised, with the relevant notation being chosen as follows.

5.2 Notation

As specified in Section 5.1, n design points are to be used in each time epoch, with a total of T epochs and therefore nT points. Suppose there are a total of h_0 potential collector sites available for use. As in Chapter 3, let the set $\{1, \ldots, h_0\}$ be treated as a fixed indexing of these potential collector sites.

Let the set C consist of the chosen collectors across all T epochs; that is, it can be partitioned as $C = \cup_{t \in \{1,\ldots,T\}} C_t$ where C_t consists of the chosen collectors at epoch t. As in Chapter 3, let there be an integer h_1 number of collectors available for use during each epoch, so that this value is fixed over time. Then each subset C_t (the choices of collector for epoch t) has cardinality $|C_t| \leq h_1$, where the inequality indicates that not all collectors need be deployed during a given epoch. Each potential collector site may be utilised up to T times, subject of course to the numbers of usages of the other potential collector sites together with the requirement that $|C| = h_1 T$. A potential collector site being used T times would correspond to being used during every epoch of the experiment. Let ξ_H be the design consisting of the sensor sites to be used, across all epochs, with design points belonging to

the finite subset $\mathcal{A}$ of the design space χ. Thus $\mathcal{A}$ represents the points that may be used during any epoch, and this set is effectively duplicated for every new epoch; that is, it is no longer (necessarily) the case that $\xi_H(x) = 1\ \forall x \in \mathcal{A}$ since design points may now be used more than once, provided that no two such usages occur during the same epoch. The design must however still be *exact*; for each $x \in \mathcal{A}$, $\xi_H(x) = q/n$, for some $q \in \{1, 2, \ldots, n\}$. Let X_t represent the set of design points used during epoch t. Then $|X_t| = n$, for each $t \in \{1, \ldots, T\}$.

5.2.1 Model specification

We assume the same model form as in Chapter 3, repeated here for convenience. We have a sequence $\mathbf{f}$ of p continuous real-valued functions $f = (f_1, f_2, \ldots, f_p)$ on χ, to give the coefficient structure of the linear model; write $f(x)$ for $(f_1(x), f_2(x), \ldots, f_p(x))$ when $x \in \chi$. Observations y_i are to be collected once the x_i have been chosen, with the observations assumed to be of the form:

$$y_i = \sum_{j=1}^{p} (f_j(x_i)\beta_j) + e_i, \tag{5.1}$$

for p unknown coefficients β_j, whose values are to be estimated by the experiment, and n error terms e_i, which are uncorrelated, each with Gaussian distribution; mean 0 and variance σ^2. The use of a Poisson model as an alternative to Gaussianity is considered in the discussion at the conclusion of this chapter.

5.3 Move costs

To maximise learning, one might expect that the use of as few duplications in design points as possible would be desirable. This may not be the case if only a small subset of the available design points are located near the vertices of the hypercuboid bounding the design space, but this will by no means always be the case, and the methodology developed for multiple time epochs must at least be capable of attempting to avoid duplications. This leads to the expectation that it will often be necessary not only to move design points to new locations between time epochs, but that entire collectors will need to be reassigned, with the concomitant relocation of all probes associated with them. It may be assumed that

it is more (economically) costly to relocate an entire collector and all its sensors than it is to relocate some or all of the sensors within the pool of available sensor sites associated with a given collector. Thus, there is a compromise to be made between the amount of relocation cost required by C, and the quality of the design that can be achieved by exploiting the potential for sensor and/or collector relocations.

A methodology is adopted based on a set of varying applications of the Nested Fedorov and standard Fedorov algorithms, and comparison of their results in terms of optimality.

5.4 Methodology

The following set of methods provides different approaches to striking balance between minimising reallocation costs (but only implicitly), and maximising not only the overall value of the design criterion, but its value for points used in the first time epoch. Each method produces a design independently of the others, except 4, which takes the result of 3 as its input. The preferred design can be selected on the basis of its optimality value, reallocation costs, or optimality during the first epoch; or a combination of these. As specified in Section 5.2, there are T epochs in total. Each method adheres to any restriction on the number of sensors per collector caused by the nesting structure. However, Methods 4 and 5 can result in designs that do not meet the requirement of n points per epoch. The methods are as follows:

1. Use the Nested Fedorov algorithm to construct a design of Tn points using only h_1 collectors, so that the chosen collectors will be fixed for the T epochs. The dataset input consists of the set of available points for one epoch, duplicated so that the number of available collector sites does not increase, but the number of available sensor sites allocated to each collector is T times its value.

2. As Method 1, with the same dataset of potential sites, but construct the design for only n points, intending these as the epoch 1 points; then allow the standard Fedorov algorithm to add n points to the design (fixing the initial n points), for the subsequent epochs with the collectors also fixed to be the same as for the first epoch.

3. Use the Nested Fedorov algorithm to choose h_1 collectors and n points, for the first epoch, then fix these collectors and points and allow the same algorithm to choose the collectors and points for the subsequent epochs, from any of the available collector and sensor sites.

4. Run the standard Fedorov algorithm on the output of Method 3, fixing the chosen collectors but allowing the sensors to vary. This may of course result in a distribution of probes that makes it difficult or impossible to adhere to the requirement that there be precisely n points per epoch. This method is included as a slight adjustment to Method 3 where the D-optimality may increase, but the practicality of assigned probes to epochs becomes more problematic.

5. Use the Hierarchical Fedorov algorithm to choose the collectors for all epochs simultaneously. As with Method 4, this can make the probe-to-epoch allocation problematic, in order to satisfy the requirement of precisely n points per epoch.

The Hierarchical Fedorov algorithm is adapted to allow the fixing of specified sensors and collectors, in order to make Methods 2 and 3 possible. Note that the methods do not directly consider move costs in the choices of pairwise swaps.

5.5 Results

Simulated datasets are again used to assess the methods' ability to identify a known optimal design, where optimality is assessed purely on the basis of D-optimality, not move costs. The application dataset is also used. In Chapters 3 and 4 it was possible to make comparisons of the progress of the algorithm(s) during a given run, between for example the use of the Hierarchical Fedorov algorithm with and without an informed starting design, compared at successive collector swaps. However, for the five methods described in 5.4, and the naïve sampling method, the work being performed at a given stage does not have the same meaning for different methods. For example, Methods 1 and 2 choose all the collectors to be used from the beginning, but with different numbers of sensors initially. However, efficiencies with respect to the D-optimality criterion may be used to compare the finally generated

Method	Mean Efficiency	Std. Dev. Efficiency
1	1.000	0.00181
2	1.000	0.00
3	0.995	0.0155
4	0.995	0.0155
5	0.991	0.0225

Table 5.1: Comparison of D-efficiencies achieved using time-based methods for simulated data (Type A) (3 sig. figures).

designs. Similarities in terms of the collectors chosen may also be assessed.

5.5.1 Simulated data

50 simulations are performed with a known hidden linearly-optimal design of 16 points augmented by Uniformly sampled points, as in Chapter 3. The data range is $[-1, 1]$ for these simulations. Two time epochs are used. There are 5 collectors to be chosen from 20 available collectors per epoch. For each method, this hidden design is made accessible through an appropriate choice of collectors. For Methods 1 and 2, it is accessible through a choice of 5 collectors, and for Methods 3-5, through a choice of 10 collectors. The methods are repeated 16 times for each of the 50 simulations. Table 5.1 gives achieved efficiencies for each of the methods. Methods 3, 4 and 5 are the only ones that can result in different collectors being chosen for the second epoch compared to the first. For the simulated data, these methods consistently choose to move all of the collectors to new locations for the second epoch, when there is an optimal design that may be accessed through such a choice. In this scenario, the optimal design requires moves of sensors over time in order to be accessed.

5.5.2 Application data

The application dataset as used in Chapters 3 and 4 is used to test the time-based methods. Two time epochs are used, with 16 algorithm repetitions, 20 design points chosen from 2047 potential points, and 5 collectors chosen from 55 sites. The achieved efficiencies are displayed in Table 5.2.

Method	Efficiency
1	0.900
2	1.00
3	0.828
4	0.829
5	0.862

Table 5.2: Comparison of efficiencies achieved using time-based methods for application data (3 sig. figures).

It would appear from these results that whilst all five of the non-naïve methods do perform well, Method 2 outperforms the others. Method 2 has appeal as an approach, in its combination of minimising reallocation costs by fixing the collectors in the epochs, with the greedy approach to information by using n initial points for the first epoch, in an attempt to concentrate the information in this first epoch.

5.6 Discussion

5.6.1 Allocation of sensors to epochs

Methods 4 and 5 do not display superiority in terms of D-optimality over Methods 1 and 2 in the simulation runs. This is despite the freedom that they are permitted in failing to observe the requirement for n points per epoch. Had they proven superior to Methods 4 and 5, there would be grounds for adapting their methods to control the allocation of sensors to epochs. This could be achieved by having the standard and Nested Fedorov algorithms only be permitted to make swaps of sensors and collectors within assigned epochs, so that the number of sensors used in any epoch did not change.

5.6.2 Move costs

Move costs could be incorporated in the implementations of the standard and Nested Fedorov algorithms, in order that these costs be considered during the design construction rather than as a criterion for deciding upon which of the five methods detailed in Section 5.4 to use. This would be achieved by altering the basis upon which the value of each po-

tential swap is assessed. The hypothetical change in D-optimality would still be calculated, but a move cost(s) would be subtracted if the swap would result in a move or moves. The ratio of move cost to D-optimality would be a user-defined input. The question of moving entire collectors becomes complex in this scenario, since the number of points per collector is not fixed. In such a scenario, it would also be of interest to explore the effect of varying the ratio of move cost to D-optimality value. This ratio would be vital in determining the behaviour of the algorithm subject to the cost and D-optimality requirements. However, its value may be dictated by external factors, depending on the relative importance of the parameter estimates and the costs of moving sensors over time.

5.6.3 Poisson models

An unrelated question is that of the use of a Poisson-based model. This would naturally add a great deal of complexity to the problem, not least because of the dependence of the parameter estimates upon the data. Should a Poisson model be assumed, it would become prudent to use any data collected during the first time epoch in subsequent ones in a Bayesian manner. The matter of allowing more than two time epochs (as used in the simulations) would also add significant complexity to the process, but could be useful for scenarios in which long term use and relocation of sensors is possible.

Chapter 6

Conclusions and future directions

6.1 Contributions

This thesis has presented new methodology for the construction of D-optimal designs subject to a nesting constraint. Allowing for this constraint in the construction of designs makes it possible to assign design points to higher-level members of the nesting hierarchy, which in practical terms means that sensors may be deployed in the natural environment around local data collecting devices, for relay to a server. The methodology presented allows for linear or Poisson models to be constructed with respect to D-optimality. The existing Fedorov exchange algorithm is adapted to permit swaps of collectors based upon assessment of their potential contribution to the design's D-optimality criterion value. This is performed for linear models in Chapter 3 and then for Poisson models in 4. The extension to Poisson models involves additional computation using a Gaussian quadrature approach. The successful implementation of these methods in R represents a considerable amount of work. Linear models may also be constructed for multiple time epochs with respect to D-optimality. Several potential methods are proposed and compared in Chapter 5, accounting for the possibility of moves of sensors over time. Results have demonstrated a clear ability of the Nested Fedorov algorithm to identify known optimal designs in all three of these scenarios, and the algorithm also demonstrates convergence and improvement upon naïve or alternative methods. In the linear case, the results have been shown to achieve designs with consistent prediction variance across most of the design space.

6.2 Future work

There are many possibilities for further work. As noted in the discussion at the end of Chapter 4, the use of a negative Binomial model rather than Poisson may help to account for possible overdispersion caused by important but unavailable attributes at the sensor sites. There would be likely to be an increase in the computation required, with a more complex information matrix than for the Poisson case. This information matrix would depend upon the model regression parameters but also upon the dispersion parameter.

Also based on the work in Chapter 4, an alternative to the quadrature method currently employed for Poisson models would be the assumption of a discrete prior distribution for the parameters. This might reduce computation times within the algorithm run itself by removing the need for the quadrature, but would rely upon appropriate specification of the prior distribution, which might in itself present more serious difficulties.

The use of a weight matrix for the Poisson model also raises the possibility of using a weight matrix to deal with possible spatial correlation in the candidate design points. The issue of dependent errors leads to a complex problem with a nonconvex space on which to optimise, even before imposing the nesting constraint, as noted for example by Dette et al. (2016). It may be necessary to start from a higher number of designs in order to increase the chance of finding an optimal or close-to-optimal design. The starting designs may also need to be more strategically chosen, using a grid search for example. The current implementation of the Nested Fedorov algorithm already allows for the use of a weight matrix that is not necessarily diagonal in the computation for linear model construction, and this means that there is potential for further work on this in the near future.

Section 5.6 notes a number of possible areas for further work in the implementation of the algorithms for use over multiple epochs.

One further area for exploration is the possibility of multiple membership of sensors to collectors. The current assumption is that there is no overlap and that each potential sensor (potential design point) has only one assigned collector. With multiple membership, it becomes possible to choose which sensor-collector memberships to use, based on their effect upon the optimality criterion. The Nested Fedorov algorithm in its current form then

becomes unsuitable for design construction, as it relies on a single membership of design points to collectors. With multiple membership, one possible methodology would be to choose a primary membership for each design point with more than one associated collector, in order to consider the situation as if there were no multiple memberships. Suppose that a point x_i had two associated collectors c_1 and c_2. Before constructing any design, we might consider the optimality criterion values that would arise by using all elements of each of the pools of potential design points associated with c_1 and c_2 in turn, and assess the changes in these two values that would be caused by associating x_i with each of them. x_i would then be assigned to the one whose value it best affected. However, this approach would be highly dependent upon the order in which points with multiple memberships were assigned to single collectors, and in any case would not directly take account of the actual design subsequently chosen by the Nested Fedorov algorithm.

An alternative approach would be to modify the Nested Fedorov algorithm methodology itself such that during the course of the algorithm's run, the multiple memberships were considered as part of the decision to make a collector swap. Suppose that a collector swap were to be made, removing collector c_1 and including collector c_2 in its place. Any design points already being used for c_1 that also had membership to c_2 could be retained. Any others would be removed and their sensors relocated to c_2, provided that they could be accommodated there. Before actually performing a collector swap, however, a choice would still need to be made as to the most appropriate swap to perform. In the current Nested Fedorov algorithm methodology, the consideration of a potential swap includes consideration of the potential removal of one collector's design points from the design, and the potential inclusion of another collector's design points. The potential removal and potential inclusion could now include bounds. For the potential removal,

- the worst case is the optimality criterion change that would result from excluding all of the points associated with the candidate-collector-for-removal, including those with membership to other currently used collectors;
- the best case is the optimality criterion change that would result from excluding only those of the candidate-collector-for-removal's points that have no membership with

any of the other currently used collectors.

For the potential inclusion,

- the worst case is the optimality criterion change that would result from including only those of the candidate-collector-for-inclusion's points that have no membership to a currently-used collector;
- the best case is the optimality criterion change that would result from including all of the candidate collector-for-inclusion's points, regardless of their membership to any of the currently-used collectors. For points currently used by some other currently-chosen collector, but with membership to the candidate-collector-for-inclusion, they are temporarily considered as if they belonged to the candidate collector rather than the current design.

These bounds would then permit calibration of the algorithm according to the degree of optimism to be employed in making collector swaps. The calibration would also depend upon the nature of the multiple memberships, which might include factors such as:

- Mean number of memberships per design point;
- Mean number of multiple memberships per collector;
- Quantities relating to a distance matrix for the collectors, constructed based upon the number of design points in common.

These potential areas of further exploration afford many opportunities for exciting developments of the methodology presented in this thesis.

Appendix A

Nested Fedorov algorithm in detail

This Appendix provides a more detailed account of the Nested Fedorov algorithm and its implementation in R; Algorithm 2 is replicated from Chapter 3 for convenience, and contains the main components, whilst Algorithm 3 describes the adaption of a given design to satisfy a constraint on the number of design points per collector, which is a possible practical requirement. Information on the implementation of the standard Fedorov algorithm is also provided.

A.1 Nested Fedorov algorithm

The algorithm inputs for the Nested Fedorov algorithm are:

- n (the prescribed number of design points) and h_1 (the prescribed number of collectors to be used)- assume that the h_0 potential collector sites are simply labelled $1, \ldots, h_0$;
- $\{(x, g(x))\}\ \forall x \in \mathcal{A}$ (the nesting structure);
- Criterion for minimum determinant before a matrix is deemed singular- this is not referred to explicitly in the algorithm description, but checks are made at the relevant points before altering the design, in order to avoid a singular matrix M.
- An option to calculate the D-optimality criterion either for a linear model, or for a Poisson one, using a specified Gaussian quadrature input. The implementation for the

Gaussian quadrature input makes use of code sourced from BioStatMatt (2015) which itself uses the Gauss Hermite function from Swihart and Lindsey (2022).

In addition to these inputs are various parameters governing the numbers of runs of the algorithm and its components. In the interests of clarity, these are not included explicitly in the algorithm description, but are as follows:

- Number of times to repeat the algorithm overall; the code permits repetitions of the algorithm with the same starting design, or with different specified starting designs. This feature can be useful in observing possible relationships between starting and output designs.

- Maximum number of times to attempt a collector swap, for each of the overall algorithm repetitions specified in the previous point. This is the only stopping criterion used by the Nested Fedorov algorithm in its current implementation. It can be used as a fail-safe against large computation times with small incremental improvements in the D criterion value, and also facilitates comparison of the progress of multiple repetitions of the algorithm at each collector swap;

- Number of additional times (beyond once) to run the standard Fedorov algorithm after a collector swap, and the number of times to attempt a swap of design points in these runs. This corresponds to line 23 in Algorithm 2.

- Number of additional times (beyond once) to run the standard Fedorov algorithm after having added a collector, in the case of there being too few used, and the number of times to attempt a swap in these runs. This corresponds to line 29 in Algorithm 2.

A.2 Standard Fedorov algorithm

The standard Fedorov algorithm is implemented in R, in order that it may be run at lines 23 and 29 of the Nested Fedorov algorithm (algorithm 2). At these points, it is run on $\{x : g(x) \in C\}$, with starting set of design points $\{x : \xi_H(x) = 1\} \cup E$ where E consists of $\min(n, |\{x : g(x) \in C\}|) - |\{x : \xi_H(x) = 1\}|$ randomly sampled points from

input : n, h_1, h_0,$\{(x, g(x))\}$, and minimum determinant criterion.

Create a random sample $C \in \{1, 2, \ldots h_0\}$, with $|C| = h_1$;

Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with a random starting set of $\min(n, |\{x : g(x) \in C\}|)$ design points- resultant design $\xi_H \in \Xi_H$;

while *an increase in* $|M(\xi_H)|$ *is achieved* **do**

  if $|\{x : g(x) = c \ AND \ \xi_H(x) = 1\}| > 0 \ \forall c \in C$ **then**

    for $c_i \in \{c : c \in C\}$ **do**

      Set the design for ξ_{H_i} to $\{x : g(x) \in (C \setminus c_i)$ AND $\xi_H(x) = 1\}$.;

      for $p_j \in \{g(x) : g(x) \notin C\}$ **do**

        if $|\{x : g(x) = p_j\}| \leq (n - |\{x : \xi_{H_i}(x) = 1\}|)$ **then**

          Set the design for $\xi_{H_{ij}}$ to $\{x : \xi_{H_i}(x) = 1$ OR $g(x) = p_j\}$

        else

          Set $\xi_{H_{ij}} = \xi_{H_i}$;

          for k *in* $1 : (n - |\{x : \xi_{H_i}(x) = 1\}|)$ **do**

            for $x_l \in \{x : g(x) = p_j \ AND \ \xi_{H_{ij}}(x) = 0\}$ **do**

              Set the design for $\xi_{H_{ijk}}$ to $\{x : \xi_{H_{ij}}(x) = 1\} \cup x_l$

            end

            $\xi_{H_{ij}} \leftarrow \arg\max_{\xi_{H_{ijk}}} |M(\xi_{H_{ijk}})|$.

          end

        end

      end

      $\xi_{H_i} \leftarrow \arg\max_{\xi_{H_{ij}}} |M(\xi_{H_{ij}})|$.

    end

    $\xi_H \leftarrow \arg\max_{\xi_{H_i}} |M(\xi_{H_i})|$ and then $C \leftarrow \{g(x) : \xi_H(x) = 1\}$;

    Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with starting design ξ_H.

  else

    for $p_i \in \{g(x) : g(x) \notin C\}$ **do**

      Set the design for ξ_{H_i} to $\{x : \xi_H = 1$ OR $g(x) = p_i\}$.

    end

    $C \leftarrow C \cup p_i$ where $i = \arg\max_i |M(\xi_{H_i})|$. Note that ξ_H is not altered here;

    Run standard Fedorov algorithm on $\{x : g(x) \in C\}$ with starting design ξ_H.

  end

  if *The current run of the while loop has not resulted in an increase in* $|M(\xi_H)|$, **then**

    undo its effects, and halt the while loop at this point.

  end

end

output: ξ_H (C may be inferred from this).

Algorithm 2: Nested Fedorov Algorithm

$\{x : \xi_H(x) = 0, g(x) \in C\}$. Note this standard Fedorov run does not affect C, the currently chosen collectors. If specified, at each of these two points (lines 23 and 29), the standard Fedorov algorithm may be run a further number of times, each time with a new randomly sampled starting set, of size $\min(n, |\{x : g(x) \in C\}|)$ sampled from $\{x : g(x) \in C\}$, with the best design always being chosen. The implementation of the standard Fedorov algorithm has certain bespoke features to aid its use within the Nested algorithm:

- An option to use a stopping criterion based on determinant increase, or to impose a maximum number of swaps. If the latter option is selected, the algorithm continues either until no increase in determinant is achieved, or the maximum number of swaps has been performed.

- An optional input of a starting design ξ_H from which to attempt pairwise swaps. If $|M(\xi_H)|$ and/or $M(\xi_H)^{-1}$ are known, these may also be input to eliminate unnecessary computation. If no starting design is specified, a random one is used; uniform random sampling of the required number of design points is performed from the candidate points until a design with non-singular M is found.

- If the starting design input option is exercised, a further option is also available, allowing a specified subset of the starting design to be fixed. This means that this subset will remain in the design, and the only points to be considered for replacement with other points as a result of pairwise swaps will be the elements of its complement in the starting design.

- As for the Nested Fedorov algorithm; an option to calculate the D-optimality criterion either for a linear model, or for a Poisson one, using a specified Gaussian quadrature input.

Both the standard and Nested Fedorov algorithm implementations include an option to make use of the computational shortcuts for determinant update and inverse update described in Section 3.3, for either linear or Poisson models.

A.3 Sensors-per-collector constraint

Algorithm 3 considers the possibility that, for practical reasons, the constraint $|\{x : g(x) = c\}| \in [a, b]\ \forall c \in C$ is to be imposed on the design, if possible, with $a, b \in \mathbb{N}$. This algorithm may be run separately on an output design from the Nested Fedorov algorithm, although its use is included as an option in the Nested Fedorov algorithm implementation as a final amendment to the output design, for ease.

if *there exists at least one rearrangement of the current design points given by* ξ_H*, subject to the current collector choice of* C*, that would satisfy the constraint* $|\{x : g(x) = c\}| \in [a, b]\ \forall c \in C$*, with* $a, b \in \mathbb{N}$ *;*
then
  while *the constraint is not met* **do**
    Define the following disjoint sets;
    $C_1 = \{c \in C : |\{x : g(x) = c\}| = a\}$;
    $C_2 = \{c \in C : |\{x : g(x) = c\}| = b\}$;
    $C_3 = \{c \in C : |\{x : g(x) = c\}| < a\}$;
    $C_4 = \{c \in C : |\{x : g(x) = c\}| > b\}$ and;
    $C_5 = \{c \in C : |\{x : g(x) = c\}| \in (a, b)\}$;
    and effect a single pairwise swap of design points in ξ_H; permissible swaps are those that either;
    ((decrease $|C_2 \cup C_5|$ by 1) AND (increase $|C_3|$ by 1)), OR;
    ((decrease $|C_4|$ by 1) AND (increase $|C_3|$ by 1)), OR;
    ((decrease $|C_4|$ by 1) AND (increase $|C_1 \cup C_5|$ by 1));
    (Note that the above three cases have been deliberately written so as to be disjoint, as this helps to avoid consideration of duplicate cases in the implementation). Choose the swap that results in the greatest change in the value of $|M(\xi_H)|$, even if this is negative. If no swap is possible, halt the *while* loop.
  end
else
  Leave ξ_H unchanged.
end

Algorithm 3: Greedy *design-points-per-collector* constraint modification.

In order to mitigate the risk of error propagation in real-world applications, the computational shortcut option in the Nested Fedorov algorithm includes the computation of determinants and inverses explicitly at regular strategic points in order to correct any errors. Similarly in the standard Fedorov algorithm implementation, an option is available that uses the Section 3.3 techniques in deciding upon a swap, but computes the resulting

determinant and inverse explicitly at the conclusion of each swap.

Bibliography

Arnouts, H. and P. Goos (2009). Update formulas for split-plot and block designs. *Computational Statistics and Data Analysis 54*, 3381–3391.

Bakhsh, S.T. (2017). Energy-efficient distributed relay selection in wireless sensor network for Internet of Things. *13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 1802–1807 DOI: 10.1109/IWCMC.2017.7986557.

Bartlett, M. S. (1951). An Inverse Matrix Adjustment Arising in Discriminant Analysis *Annals of Mathematical Statistics 22*(1), 107–111.

BioStatMatt (2015). Notes on Multivariate Gaussian Quadrature. *R bloggers* https://www.r-bloggers.com/2015/09/notes-on-multivariate-gaussian-quadrature-with-r-code/ Accessed on 25 March 2024.

Bliemer, M. C. J., J. M. Rose, and S. Hess (2009). Approximation of Bayesian Efficiency in Experimental Choice Designs. *Journal of Choice Modeling 1*, 98–127.

Chugh, A. and S. Panda (2018). Strengthening Clustering Through Relay Nodes in Sensor Networks. *Procedia Computer Science 132*, 689–695.

Cook, R. D. and C. J. Nachtsheim (1980). A Comparison of Algorithms for Constructing Exact D-Optimal Designs. *Technometrics 22*(3), 315–324.

Cook, R.J., G.Y. Yi and K-A. Lee (2004). A Conditional Markov Model for Clustered Progressive Multistate Processes under Incomplete Observation. *Biometrics 60*, 436–443.

Dette, H., A. Pepelyshev and A. Zhigljavsky (2016). Optimal Designs in Regression with Correlated Errors. *The Annals of Statistics 44*(1), 113–152.

Dormann, C. F., J. M. McPherson, M. B. Araújo, R. Bivand, J. Bolliger, G. Carl, R. G. Davies, A. Hirzel, W. Jetz, W. D. Kissling, I. Kühn, R. Ohlemüller, P. R. Peres-Neto, B. Reineking, B. Schröder, F. M. Schurr and R. Wilson (2007). Methods to account for spatial autocorrelation in the analysis of species distributional data: a review. *Ecography 30*, 609–628.

Golub, G. H. and J. H. Welsch (1969). Calculation of Gauss Quadrature Rules *Mathematics of Computation 23*(106), 221–230 + s1–s10.

Fedorov, V. V. (1972). Theory of Optimal Experiments. 1st ed. Academic Press, New York, NY.

Fedorov, V. (1996). Design Of Spatial Experiments: Model Fitting And Prediction *Handbook of Statistics 13*. DOI: 10.2172/231193

Ford, I., B. Torsney and C. F. J. Wu (1992). The Use of a Canonical Form in the Construction of Locally Optimal Designs for Non-Linear Problems *Journal of the Royal Statistical Society. Series B (Methodological) 54*, 569–583.

Goos, P. and B. Jones (2019). Design in the Presence of Nested Factors. *Technometrics 61*(4), 533–544. DOI: 10.1080/00401706.2018.1562986

Goos, P. and K. Mylona (2018). Quadrature Methods for Bayesian Optimal Design of Experiments With Nonnormal Prior Distributions. *Journal of Computational and Graphical Statistics 27*, 179–194.

Gotwalt, C. M., B. A. Jones and D. M. Steinberg (2009). Fast Computation of Designs Robust to Parameter Uncertainty for Nonlinear Settings. *Technometrics 51*, 88–95.

Harman, R. and L. Filová and P. Richtárik (2020). A Randomized Exchange Algorithm for Computing Optimal Approximate Designs of Experiments. *Journal of the American Statistical Association 115*, 348–361.

Harville, D. A. (1997). Matrix Algebra from a Statistician's Perspective, 1st ed. *Springer-Verlag, New York, NY.*

Hicks, C. R. and K. V. Turner (1999). Fundamental concepts in the design of experiments. 5th ed. Oxford University Press, New York, NY.

Johnson, M. E. and C. J. Nachtsheim (1983). Some Guidelines For Constructing Exact D-Optimal Designs on Convex Design Spaces. *Technometrics 25*(3), 271–277.

Kiefer, J. (1959). Optimum Experimental Designs *Journal of the Royal Statistical Society. Series B (Methodological) 21*(2), 272–319.

Khuri, A. I., B. Mukherjee, B. K. Sinha and M. Ghosh (2006). Design Issues for Generalized Linear Models: A Review *Statistical Science 21*(3), 376–399.

Labadi, L. A. (2015). Some Refinements on Fedorov's Algorithms For Constructing D-Optimal Designs. *Brazilian Journal of Probability and Statistics 29*(1), 53–70.

Lee, M. S. C. (1988). Constrained Optimal Designs. *Journal of Statistical Planning and Inference 18*, 377–389.

Li, H., C. Ao, Y. Xu, J. Tian and K. Yamashita (2017). Relay Node Position Optimization in Complex Environment. *IEEE Wireless Communications and Networking Conference (WCNC)*, 1–6 DOI: 10.1109/WCNC.2017.7925497

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

Morris, M. (2011). Design of experiments: an introduction based on linear models. 1st ed. Chapman & Hall/CRC, Boca Raton, FL.

Nguyen, Nam-Ky and A.J. Miller (1991). A review of some exchange algorithms for constructing discrete D-optimal designs. *Computational Statistics & Data Analysis 14*, 489–498.

Nishii, R. (1993). Optimality of experimental designs. *Discrete Mathematics 116*(1-3), 209–225.

Ray, P. P. (2018). A survey on Internet of Things architectures. *Journal of King Saud University - Computer and Information Sciences 30*(3), 291–319.

Rencher, A. C. and W. F. Christensen (2012). Methods of multivariate analysis. 3rd ed. Wiley, Hoboken, NJ.

Russell, K. G., D. C. Woods, S. M. Lewis and J. A. Eccleston (2009). D-Optimal Designs for Poisson Regression Models. *Statistica Sinica 19*(2), 721–730.

Sherman, J. and W.J. Morrison (1950). Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Annals of Mathematical Statistics. 21*(1), 124–127.

St. John, R. C. and N. R. Draper (1975). D-Optimality for Regression Designs: A Review. *Technometrics 17*(1), 15–23.

Swihart, B. and J. Lindsey (2022). rmutil: Utilities for Nonlinear Regression and Repeated Measurements Models *R package version 1.1.10 https://CRAN.R-project.org/package=rmutil*

Ver Hoef, J. M. and P. L. Boveng (2007). Quasi-Poisson Vs. Negative Binomial Regression: How Should We Model Overdispersed Count Data? *Ecology 88*(11), 2766–2772.

Yanping, W., R. H. Myers, E. P. Smith and K. Ye (2006). D-optimal designs for Poisson regression models. *Journal of Statistical Planning and Inference 136*, 2831–2845.

Zahran, A., C. M. Anderson-Cook and R. H. Myers (2003). Fraction of Design Space to Assess Prediction Capability of Response Surface Designs. *Journal of Quality Technology 35*(4), 377–386. DOI: 10.1080/00224065.2003.11980235