ITLS

WORKING PAPER

ITLS-WP-14-22

Constrained stated choice experimental designs

By
Andrew T. Collins, Michiel C.J. Bliemer and John M. Rose[1]

[1] The University of South Australia, Australia

October 2014

ISSN 1832-570X

INSTITUTE of TRANSPORT and LOGISTICS STUDIES
The Australian Key Centre in
Transport and Logistics Management

The University of Sydney
Established under the Australian Research Council's Key Centre Program.

NUMBER:	Working Paper ITLS-WP-14-22
TITLE:	**Constrained stated choice experimental designs**
ABSTRACT:	While a significant literature exists on the generation of experimental designs for stated choice experiments, little work has been done to effectively accommodate constraints on these designs. Constraints may include the prevention of dominated alternatives, the exclusion of specific combinations of attribute levels on plausibility and realism grounds, and the imposition of attribute level balance. We argue that most such constraints are fundamental to the success of the experiment, and should always, at least where feasible, be respected. The notable exception is level balance, which can instead be imposed as a soft constraint, using a level balance measure introduced herein. Various rule structures for specifying the constraints are proposed, as are two new algorithms that can conform to any such rules, whilst allowing the analyst to control the level of importance placed on level balance. From the two case studies investigated, it is determined that the type of constraints specified may influence which algorithm performs best, or even if the algorithm finds a solution at all. Excluding specific combinations of attribute levels reduces the statistical efficiency of the design but is nonetheless warranted. By contrast, the reduction in efficiency resulting from level balance may be excessive, and the analyst should test the trade-offs between these two design properties, by varying the weight placed on level balance in the objective function. The proposed techniques allow researchers to focus not just on the statistical efficiency of these experiments, but behavioural realism and plausibility as well.
KEY WORDS:	***Experimental design, constraints, attribute level balance, d-error, stated choice experiments***
AUTHORS:	**Collins, Bliemer and Rose**
CONTACT:	INSTITUTE OF TRANSPORT AND LOGISTICS STUDIES (C13) The Australian Key Centre in Transport and Logistics Management The University of Sydney NSW 2006 Australia Telephone: +612 9114 1824 E-mail: business.itlsinfo@sydney.edu.au Internet: http://sydney.edu.au/business/itls
DATE:	October 2014

1. Introduction

Stated choice experiments are commonly employed in the field of transportation to collect data which are used to model preferences or other outputs of interest to transport planners, such as the value of travel time. Such experiments typically require the generation of an experimental design, which controls what combinations of attribute levels are presented in each choice task that a survey respondent completes.

The attribute levels in the experimental design can be created in many ways, e.g., randomly, by adopting orthogonal arrays, or by using efficient design procedures (Rose et al., 2008). Surprisingly, little attention has been placed on constraints that may need to be imposed that invalidate certain combinations of attribute levels. Such constraints may include

- The elimination of choice tasks in which one alternative dominates the others; and
- Rules that prevent or require certain combinations of attributes levels, either within or across choice alternatives, to ensure realism, plausibility, or logical consistency.

These constraints, and their potential importance, are best illustrated with a number of examples.

In the first example, consider three unlabelled motorway alternatives, each described by travel time, running cost, toll, and toll payment options. The example choice task depicted in Table 1 presents a number of problems. Motorway B is an implausible alternative, as it has a longer travel time on the motorway than both A and C, and yet a toll is charged for B, but not A or C. For Motorway A, payment is via E-tag only, yet no toll need be paid, making the alternative logically inconsistent. Finally, C is equal or better than A and B on all attributes, and so dominates the other two alternatives.

Table 1: Motorway example

	Motorway A	**Motorway B**	**Motorway C**
Travel time on motorway (time)	25 minutes	35 minutes	25 minutes
Running costs (run)	\$4.50	\$5	\$4
Toll (toll)	\$0	\$2	\$0
Toll payment options (pay)	E-tag only	Cash or E-tag	N/A

A number of constraints could be formally specified with a set of logical statements, to make the scenario more plausible.

- If (A.toll=0) and (A.time<B.time), then reject the choice task. Repeat the same logic for all pairs of alternatives.
- If (A.toll=0) then (A.pay={N/A}). Repeat for B and C.
- If (A.toll>0) then (A.pay={Cash or E-tag, E-tag only}). Repeat for B and C.
- If (C.time<=A.time) and (C.run<=A.run) and (C.toll<=A.toll) then reject due to dominance. Repeat for all pairs of attributes.

Next, consider mode choice between train, bus and light rail. Several aspects of the choice task in Table 2 are implausible or impossible. Despite train seat occupancy of only 40 percent, 30 people are standing. The train has no transfers, yet five minutes is spent transferring, and the bus has one transfer but no transfer time. Finally, light rail dominates train and bus with respect to all of the attributes. Whilst it is feasible that modal preferences might break this dominance, it may be desirable from the analyst’s perspective to eliminate dominance in the attributes.

Table 1: Mode choice example

	Train	Bus	Light rail
Seats occupied (seats)	40%	75%	10%
People standing (standing)	30 per carriage	0	0
Number of transfers (transferNumber)	0	1	0
Time spent transferring (transferTime)	5 minutes	0 minutes	0 minutes
Travel time (travelTime)	20 minutes	15 minutes	10 minutes
Fare (fare)	\$5	\$7	\$3

The following constraints could be applied for each alternative.

- If (seats<90%) then (standing=0).
- If (transferNumber=0) then (transferTime=0).
- If (transferNumber>0) then (transferTime>0).
- Optionally, a dominance check could be applied.

A final example is drawn from Daly et al. (2012), who conducted a stated choice experiment to investigate rail security. Numerous combinations of attributes were infeasible in their choice task. For example, metal detector/x-ray security checks are not possible if the time to pass through security is less than four minutes. Further, it is not plausible to have a uniformed military presence without a certain level of other security measures first being in place, such as closed circuit television with automatic identification of individuals. Whereas they implemented a near orthogonal design, we specify constraints within an efficient design.

The generation of efficient designs involves a search over what is typically a very large design space, even once that space has been reduced through the imposition of constraints. With one exception (ChoiceMetrics, 2012), existing generation algorithms do not give explicit consideration to the types of constraints suggested above. This paper first demonstrates that existing algorithms do not handle these constraints effectively. In particular, the problem is complicated by some constraints being imposed within choice task (dominance checks and realism constraints), and others across choice tasks (attribute level balance). The paper then proposes two algorithms that effectively handle all of these constraint types, and evaluates their performance on several examples. The solution is an improvement on the algorithm currently contained in the Ngene software package (ChoiceMetrics, 2012). Transportation researchers and practitioners will benefit, as it will allow them to specify complex constraints that allow for behaviorally plausible choice tasks, whilst also minimising sample size requirements through the generation of an efficient design.

2. Background

The typical experimental design task is to determine an experimental design x which consists of the attribute levels presented in each choice task. Instead of randomly selecting a design, in most cases a certain optimality criterion is used to evaluate the design. The *d*-error is often used to determine its efficiency, such that the aim is to find an optimal design $x^* \in X$ that minimises the *d*-error and as such maximises the efficiency (i.e., Fisher information), where X is the set of feasible attribute levels.

An experimental design is in all cases restricted by some design dimensions, namely the number of choice tasks, and the total number of attributes. Let *S* denote the number of choice tasks, and let K_j be the number of attributes in alternative *j*, $j = 1,\dots,J$. Let $K = \sum_j K_j$ be the total number of attributes presented in each choice task. Hence, $X \subseteq R^{S\times K}$. Let $x \equiv [x_{jks}]$, where x_{jks} is the attribute level for attribute *k*, $k = 1,\dots,K_j$, in alternative *j*, $j = 1,\dots,J$, and choice task *s*, $s = 1,\dots,S$. Each attribute level x_{jks} has to be selected from a pre-defined set of feasible levels. Let L_{jk} denote the set of feasible levels for attribute *k*, $k = 1,\dots,K_j$, in alternative *j*, $j = 1,\dots,J$.

This set can consist of a finite number of discrete elements, such as $L_{jk} = \{1, 2, 3\}$, but can also be described by a continuous range of values, such as $L_{jk} = \{l \in R : 1 \le l \le 3\}$.

The basic experimental design optimisation problem therefore can be formulated as follows:

$$x^* = \arg\min_{x \in X} f(x) \tag{1}$$

where $X = \{x \in R^{S \times K} : x_{jks} \in L_{jk}, j = 1, \ldots, J; k = 1, \ldots, K_j; s = 1, \ldots, S\}$.

This problem is well-defined and has a solution, but finding the optimal solution is often difficult due to the fact that it is a (mixed) integer programming problem with a huge number of feasible solutions. Therefore, one mostly aims to find a feasible experimental design that is as good as possible while realising that it may not be optimal.

This paper discusses additional constraints that can be imposed on the attribute levels and therefore on set *X* and how optimisation problem (1) can be solved. Additional constraints result in a smaller set of feasible solutions, such that in some cases finding the optimal solution may become somewhat easier (although each additional constraint also makes the design less optimal). However, in most cases, either the number of feasible solutions is still extremely large, or the set of feasible solutions becomes relatively small such that finding a feasible solution turns out to be very challenging and sometimes even impossible.

Well-known examples of additional constraints include attribute level balance and orthogonality. When requiring attribute level balance (referred to henceforth as just level balance), each level in set L_{jk} has to appear an equal number of times over the choice tasks. This ensures that data points are evenly spread. A more strict constraint is orthogonality, in which each column in matrix *x* is uncorrelated with any other column. This constraint is often so strict that problem (1) no longer has a solution (i.e., $X = \emptyset$). Orthogonality is of importance in linear models, but less relevant for nonlinear models such as discrete choice models, particularly when the population parameter estimates are expected to be non-zero. Therefore, in recent years orthogonality is often no longer imposed.

While level balance and orthogonality were imposed for statistical reasons, in recent years there has been a significant shift towards the desire to include constraints for behavioural reasons. Additional constraints can make the choice tasks often more realistic or plausible by removing impossible or implausible choice tasks from the set of feasible designs, *X*. The next section discusses the different types of constraints that can be imposed.

3. Constraints in stated choice surveys

Constraints in stated choice surveys can be classified into constraints *across* choice tasks, and constraints *within* a choice task. Each class of constraints will be discussed next.

3.1 Constraints across choice tasks

In the past, analysts have mainly put constraints over multiple rows in the design, i.e., across choice tasks (ACT[1]). Level balance is a good example of a column-based constraint over choice tasks (see e.g., Huber and Zwerina, 1996), where the number of occurrences of each of an attribute's levels should be as equal as possible. Level balance is implicit for orthogonal designs, but has sometimes been enforced for efficient designs as well. For continuous attributes, this prevents cases in which nonlinearities cannot be identified, while for categorical attributes it prevents cases in which there are insufficient occurrences of a category to estimate an associated taste parameter. However, whilst too much imbalance may be problematic, complete balance may not be necessary. One mechanism for achieving the appropriate amount of balance is through the parameter priors and the efficiency measure, by using dummy or

[1] A glossary is presented at the end of the paper.

effects coding (Sándor and Wedel, 2001). The levels will be balanced to the extent that information gain is maximised. Whilst this works well for categorical attributes, the parametric cost may be too great for continuous attributes.

Orthogonality can also be considered a column-based constraint, in which the levels over choice tasks are constructed such that the entire column is orthogonal to all other columns. Further, one usually does not want repeating choice tasks in the survey. In many cases such duplicates are not noticed easily, especially in an unlabelled experiment in which swapping the levels of two alternatives essentially yields the same choice question. Filtering out such duplicate choice tasks requires additional constraints.

More recent are constraints to reduce complexity and fatigue in the survey. In particular, reducing the number of alternatives or attributes presented in each choice task has been subject of study. In labelled experiments, in which the label has a specific meaning like a brand name (e.g., Qantas, Virgin Airlines) or refers to a specific product (e.g., bus, train), sometimes the number of possible alternatives is too large to present all of them in each choice task. In that case it is possible to show only a subset of alternatives (e.g., Rose and Hensher, 2006; Rose et al., 2013). In other cases, the number of attributes is too large to present in each choice task. In this case, partial profile designs can assist in which only a subset of attribute levels change values from choice task to choice task in order to reduce the information burden on the respondent (e.g., Kessels et al., 2011).

None of these constraints serve the purpose of avoiding unrealistic, implausible, or silly choice tasks in terms of the attribute levels presented (The constraining of alternatives only to a relevant set may assist in making certain choices more relevant however. For example, Rose and Hensher (2006) restrict the alternatives presented to each respondent in the experiment to reflect only those alternatives that they would have available to them in real life, such that if a respondent did not own a car, they would not have a car alternative available to them).

3.2 Constraints within a choice task

In the past few years, an increasing number of analysts have questioned the plausibility and realism of choice tasks as commonly represented in stated choice experiments, and have indicated a desire to focus more on the behavioural interpretation of the choice tasks, as opposed to the statistical properties of the underlying design which is used to construct them. We will illustrate the issues raised with several silly choice tasks that often reside in experimental designs that are computer generated using existing techniques. Consider a travel choice in which the respondent has to choose between Route A and Route B, or decide to Stay home. Both unlabelled route alternatives are described by several attributes, namely whether it is a toll road (levels: Yes, No), travel time (levels: 10, 20 minutes), and toll costs (levels: \$0, \$1, \$2). Further, there is a scenario variable that describes the weather conditions (levels: Sunny, Rain). The level of the scenario variable is expected to be constant across the alternatives (that is, it may vary between choice tasks, but must stay the same for all alternatives within a task). The labelled alternative Stay home is a no-choice option and does not have any attributes. In total there are $(2 \cdot 2 \cdot 2 \cdot 3)^2 = 24^2 = 576$ possible choice tasks that could be constructed for this example, however, many of these 576 possible tasks are not behaviourally sensible.

For example, consider the choice task in Table 2. Route A is not a toll road, but there is a positive toll cost, which is inconsistent. Hence, we should impose a constraint within each route alternative that if it is not a toll road, then the toll cost should be zero. Note that a zero toll cost is not per se inconsistent with using a toll road, hence in the example we do not remove any route alternatives that use a toll road. We shall refer to these types of constraints as plausibility and realism (PAR) constraints.

Table 2: Silly choice task: inconsistent alternative

	Route A	Route B	Stay home
Weather	Rain	Rain	
Uses toll road?	No	Yes	
Travel time	30 min.	20 min.	
Toll cost	$1	$2	
Your choice:	☐	☐	☐

Even if inconsistent alternatives are removed, choice tasks may still be silly even if the alternatives themselves are plausible. The next three examples illustrate problematic combinations of alternatives.

Consider Table 3 in which Route A and Route B are identical. These duplicated alternatives not only mean that little information will be captured by this question, but also that the respondent may have doubts about the survey and not take it seriously. Therefore, we typically want to impose the constraint that the attribute levels across alternatives are not completely overlapping.

Table 3: Silly choice task: duplicated alternative

	Route A	Route B	Stay home
Weather	Rain	Rain	
Uses toll road?	Yes	Yes	
Travel time	20 min.	20 min.	
Toll cost	$1	$1	
Your choice:	☐	☐	☐

In Table 4 we illustrate a choice task in which the scenario variable between the routes is different, while it is clear that the weather should be the same across the two route alternatives. This was also the case in the example shown in Table 2. This requires yet another constraint on the set of feasible attribute levels.

Table 4: Silly choice task: inconsistent scenario

	Route A	Route B	Stay home
Weather	Rain	Sunny	
Uses toll road?	No	Yes	
Travel time	30 min.	20 min.	
Toll cost	$1	$2	
Your choice:	☐	☐	☐

Finally, Table 5 shows yet another silly choice task in the form of a strictly dominant alternative. It is clear that Route B is strictly preferred over Route A, since its attribute levels are all better than or equal to the attribute levels of Route B. Not only may the respondent doubt the survey, but having one or more strictly dominant alternative in the data is also problematic in estimation (see Bliemer et al., 2014). Therefore, in case of an unlabelled experiment it is important to add constraints in order to avoid strictly dominant alternatives.

Table 5: Silly choice task: dominant alternative

	Route A	Route B	Stay home
Weather	Rain	Rain	
Uses toll road?	Yes	Yes	
Travel time	30 min.	20 min.	
Toll cost	$2	$1	
Your choice:	☐	☐	☐

Out of the 24 possible attribute level combinations for each of the route alternatives, we should first remove inconsistent alternatives. This means removing eight combinations (Sunny/Rain, No, 20/30 minutes, $1/$2), such that 16 possible alternatives remain. The number of possible choice tasks is now $16 \cdot 16 = 256$. If we further would like to remove duplicate alternatives, this number further decreases to $16 \cdot (16-1) = 240$. Removing inconsistent scenarios (i.e., removing Sunny/Rain and Rain/Sunny) yields $16 \cdot (16-8-1) = 112$ feasible choice tasks. Finally, removing choice tasks with strictly dominant alternatives results in only 8 feasible choice tasks as listed in Table 6.

Table 6: Feasible choice tasks

	Road A				Road B			
s	Weather	Toll road	Travel time	Toll cost	Weather	Toll road	Travel time	Toll cost
1	Sunny	No	30 min.	\$0	Sunny	Yes	20 min.	\$1
2	Sunny	No	30 min.	\$0	Sunny	Yes	20 min.	\$2
3	Sunny	Yes	30 min.	\$0	Sunny	Yes	20 min.	\$1
4	Sunny	Yes	30 min.	\$0	Sunny	Yes	20 min.	\$2
5	Rain	No	30 min.	\$0	Rain	Yes	20 min.	\$1
6	Rain	No	30 min.	\$0	Rain	Yes	20 min.	\$2
7	Rain	Yes	30 min.	\$0	Rain	Yes	20 min.	\$1
8	Rain	Yes	30 min.	\$0	Rain	Yes	20 min.	\$2

4. Existing algorithms

There are numerous stated choice experimental design algorithms in existence, where these can broadly be classified as column-based or row-based. With column-based algorithms (CBAs), in each iteration attribute levels are changed within a certain column (i.e., attribute). In contrast, during each iteration, row-based algorithms (RBAs) change two or more levels within a choice task, but make no changes across choice tasks. It is useful to conceptualise all algorithms as having three key steps. In step 1, associated data structures, such as candidate sets, may be populated, although not all algorithms require this step. In step 2, an initial design is populated. In step 3 the design is updated successively over many iterations. Optionally, these steps may be rerun, with multiple global iterations, where this may help overcome local optima.

4.1 Column-based algorithms

There exist several CBAs for generating experimental designs, including the RSC algorithm (Huber and Zwerina, 1996), the coordinate exchange algorithm (Kuhfeld and Tobias, 2005), and the randomised exchange algorithm (Quan et al., 2011). Step 1 is typically not required under the column-based approach, although the column-based, constraint friendly algorithm we propose in this paper will require the generation of one or more candidate sets.

Column-based algorithms can typically handle constraints across choice tasks quite well. For example, the level balance constraint can be easily imposed by starting with an initial level balanced design in step 2 and in step 3 only permitting swaps or permutations within each column, thereby preserving level balance. Similarly, starting with an orthogonal design populated from an orthogonal table, orthogonality can be preserved by swapping columns within a design (between attributes that have the same number of attribute levels), swapping columns between the design and any spare columns in the orthogonal table (again if the number of attribute levels match), or by performing these swaps with orthogonal columns that are merged to create columns with the appropriate number of levels. However, CBAs usually cannot deal with constraints within choice tasks. Any change within a certain column may violate one or more constraints that exist between attributes.

4.2 Row-based algorithms

A well-known RBA is the Modified Federov algorithm (Cook and Nachtsheim, 1980). The basis of this algorithm is building a candidate set of feasible choice tasks[2] in step 1, populating the experimental design from this candidate set in step 2, and in step 3 iteratively swapping all candidate set choice tasks with all tasks in the design, testing for improvements in the optimality criterion. Due to the dimensions of the design, it may not be feasible to generate a candidate set that contains all possible combinations of attributes, thus it is common to randomly select a fractional factorial for this purpose. The within choice task (WCT) constraints are applied in the first stage when constructing the candidate set by testing all combinations of attributes which are constrained in some way. This guarantees that any design based on these candidates will always satisfy the WCT constraints. However, ACT constraints, such as level balance and orthogonality, can no longer be guaranteed and will in most cases be violated.

4.3 Assessment of existing algorithms

Although several algorithms have been proposed in the literature, none of them are able to effectively handle constraints that are both within and across choice tasks. However, we believe that WCT constraints, which are typically imposed to ensure consistency, plausibility and realism, are most important and therefore need to strictly be satisfied, while ACT constraints, which are typically imposed for statistical reasons or for choice task complexity, are less important and may be relaxed. In the next section we propose both a new CBA and a new RBA. In both, the WCT constraints are strictly satisfied, while ACT constraints may be weakly satisfied, i.e., they aim to satisfy them as much as possible but allow some flexibility.

5. Novel constrained design algorithm

5.1 Specifying constraints

The integration of constraints into stated choice experimental designs requires an unambiguous specification of what constraints are required, and may also require some additional information beyond the various dimensions of the design (attributes, levels, etc.).

It must be made clear whether the design is unlabelled, as this will require the checking of repeated alternatives within a choice task, and have implications when testing for repeated choice tasks (i.e., the order of the alternatives will be irrelevant). If dominated alternatives are to be avoided, it must be clear whether an increase in an attribute's magnitude leads to an increase or a decrease in utility[3] (see Bliemer et al., 2014).

We suggest that PAR constraints be specified using one of three rule structures, where A, B, C and D are logical expressions:

`Reject if A`
Any choice task in which logical expression A evaluates to true must be excluded from the design.
e.g., Reject if number of train transfers is greater than zero and train transfer time is zero.
The next two structures can also be expressed using the reject logic.

`Require B`
Any choice task in which logical expression B evaluates to false must be excluded from the design.
e.g., Require busway travel time to be less than or equal to regular bus travel time.

---

[2] While candidate sets may also be generated for each alternative, in this paper we only consider choice task candidate sets.
3 This can be easily determined if non-zero parameter priors are specified.

`If C then D`
Any choice task in which C is true but D is false is rejected.
e.g., If bus seat occupancy is less than 90 percent, then zero people are standing.

5.2 A measure of level balance

In order to facilitate the weak satisfaction of level balance, we propose a measure of attribute level (im)balance over K attributes, which may be a subset of all attributes. Consider an experimental design x, which contains S choice tasks. Each attribute k has L_k levels. We calculate the level imbalance (LIB) as:

$$LIB = \frac{1}{K}\sum_{k}^{K}\left[\frac{\sum_{l}^{L_k}\left(\frac{S}{L_k}-\sum_{s}^{S}1_{\{x_{ks}=l\}}\right)^2}{\left(\frac{S}{L_k}-S\right)^2+\left(L_k-1\right).\left(\frac{S}{L_k}\right)^2}\right]$$

In essence, for each attribute, the difference is calculated between the number of times each attribute level occurs, and the number of times it should occur under level balance. Imbalance of greater magnitude is penalised by squaring the difference. The denominator represents the balance measure in the worst case. The final level imbalance measure lies between 0 (full balance) and 1 (full imbalance), and is useful for combining with an efficiency measure in the optimisation function. Alternatively, the level balance LB can be expressed as 1-LIB.

Another measure which may be useful in some contexts is one which considers the average 'distance' from level balance, without any comparison to the worst balance outcome, or penalisation of larger distances. Again over the K attributes of interest, the average distance from level balance (ADFLB) is:

$$ADFLB = \frac{1}{K}\sum_{k}^{K}\sum_{l}^{L_k}\left|\frac{S}{L_k}-\sum_{s}^{S}1_{\{x_{ks}=l\}}\right|$$

As an example of ADFLB for one attribute only, if that attribute has three levels and the design 12 choice tasks, balance is achieved if each level occurs four times. If the frequency of each of the levels is actually 2, 4 and 6, then $ADFLB = |4-2|+|4-4|+|4-6| = 4$. When comparing the level balance properties of alternative designs in this paper, we will report both LB and ADFLB.

5.3 Step 1: Generation of candidate set(s) of constrained attributes

When PAR constraints are specified, we propose that one or more candidate sets be generated for the PAR constrained attributes, irrespective of whether the CBA or RBA are employed. Define a cluster as all attributes linked together logically by one or more PAR constraints. For example, if there are three PAR constraints, concerning attributes A and B, C and D, and A and E, respectively, then there will be two clusters: ABE and CD. For each cluster, we wish to determine the full factorial of PAR compliant attribute combinations. One approach, which we call naïve iteration, is to generate the full factorial of all combinations, then eliminate any combination that violates a constraint. However, if each cluster has many attributes, such an enumeration may be prohibitively large and intractable, even if the resulting factorial of valid combinations is not very large.

As an alternative, we propose a bottom up approach. Maintain a collection of sets, where each set contains one or more of the K attributes[4]. For each set, maintain a full factorial of valid

[4] Note that an attribute is actually an alternative-attribute combination, but will be referred to simply as an attribute here.

attribute levels combinations. A combination is valid if it does not violate any of the constraints *considered thus far*. Initially, generate K unique sets, each populated solely with one of the K attributes. Then, for each rule, generate a set which contains only the attributes referenced in the rule. Generate the full factorial of all level combinations for these attributes. Eliminate all combinations that violate the constraint. Then merge this set and all sets that contain these attributes. This process is repeated for each of the rules, where the final sets will be the clusters that we seek.

This process is best illustrated with an example. We have a design with the following attributes and levels, partitioned into sets.

Initial sets

A	B	C	D	E
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4				

Additionally, the following PAR constraints are specified:

1. Require A>=B
2. If (C=1) then (D={2,3})
3. Require A+E<5

For constraint 1, we combine A and B, generate the full factorial, and remove three combinations that violate the constraint. A similar process is performed for constraint 2.

Constraint 1

A	B	C	D	E
1	1	1	1	1
1	2	2	2	2
1	3	3	3	3
2	1			
2	2			
2	3			
3	1			
3	2			
3	3			
4	1			
4	2			
4	3			

Constraint 2

A	B	C	D	E
1	1	1	1	1
2	1	1	2	2
2	2	1	3	3
3	1	2	1	
3	2	2	2	
3	3	2	3	
4	1	3	1	
4	2	3	2	
4	3	3	3	

For constraint 3, even though A and B exist in a set, a full factorial for A and E is first generated and three combinations removed. Then this set AE is merged with set AB, due to the overlap in attribute A. The final result is two clusters, ABE with 18 combinations and CD with six combinations. The efficiency of this algorithm may not be apparent here, but the improvement over the naïve iteration approach will become greater as the number and overlap of PAR constraints increases.

Constraint 3, step 1						**Constraint 3, step 2**				
A	B	C	D	A	E	A	B	E	C	D
1	1	1	2	1	1	1	1	1	1	2
2	1	1	3	1	2	1	1	2	1	3
2	2	2	1	1	3	1	1	3	2	1
3	1	2	2	2	1	2	1	1	2	2
3	2	3	1	2	2	2	1	2	3	1
3	3	3	2	2	3	2	1	3	3	2
4	1			3	1	2	2	1		
4	2			3	2	2	2	2		
4	3			3	3	2	2	3		
				4	1	3	1	1		
				4	2	3	1	2		
				4	3	3	2	1		
						3	2	2		
						3	3	1		
						3	3	2		
						4	1	1		
						4	2	1		
						4	3	1		

What results from the above algorithm is, for each cluster of attributes which are related through constraints, a full factorial of attribute combinations which do not violate any of the specified PAR constraints. For the CBA, nothing further is done in step 1. For the RBA, a single candidate set is generated, where each candidate is comprised of all attributes in each choice task. For smaller designs, this can constitute all combinations of attribute levels that do not violate the PAR constraints. For most design specifications, it is necessary to draw randomly with replacement from each of the clusters of attributes (which satisfy all PAR constraints). The choice task thus generated is then only accepted into the candidate set if it is not a repetition of choice tasks already drawn, and, if applicable, it does not contain any dominated or repeated alternatives. The analyst decides how many such successful draws are required, i.e., the size of the final candidate set. Global iterations of the RBA may overcome local optima that result from the stochastic nature of the candidate set generation for larger designs.

5.4 Step 2: Populating the initial design

For the RBA, the initial design is populated by drawing candidate choice tasks from the candidate set. No further constraints of any form are imposed at this stage – in particular, level balance will be sought during the iterations of step 3.

For the CBA, the population method will depend on whether level balance is sought. If not, attributes that are not associated with PAR constraints are randomly assigned, and combinations from the clusters of PAR constrained attributes from step 1 are assigned. Any remaining WCT constraints (duplicate alternatives, dominance) must be satisfied and choice tasks must not be repeated. This may necessitate an iterative approach until all constraints are met, and in particular if there are many WCT constraints, design generation may fail at step 2.

If level balance is sought under the CBA, attributes unassociated with PAR constraints are assigned by cycling through and assigning each level, thereby maximising balance. For PAR constrained attributes, each cluster can be assigned with a variant of the Modified Federov algorithm. Combinations of attributes are initially assigned from the clusters' candidate set that was generated in step 1. Each row is swapped with every candidate that remains in the candidate set. These swaps are accepted if there is a reduction in LIB, here calculated over the attributes in the cluster. The degree of level balance that can be achieved may depend on the nature of the PAR constraints, and some upper bound on the duration of balancing per cluster should be applied.

For the CBA, local optima might stem from the initial design population, and multiple global iterations are strongly recommended.

5.5 Step 3: Iterating over alternative designs

Under the CBA, pairs of attribute levels are swapped, as with the randomised exchange algorithm (Quan et al., 2011). However, if one attribute in a cluster of attributes linked by PAR constraints is swapped, the swap is also performed for all attributes in the cluster, ensuring that the associated PAR constraint is not violated. The swap is not accepted if dominance or duplicated alternative checks fail, nor if the swap results in a choice task repetition.

Under the RBA, the conventional Modified Federov algorithm is employed (see Section 4.2). However, if level balance is sought, then the level imbalance measure is calculated, and entered into the optimisation function together with the desired efficiency measure. A weighting α_{LIB} can be employed such that the final measure is $\alpha_{LIB} \cdot LIB + (1-\alpha_{LIB}) \cdot$ efficiency measure. The best weighting to use will be case specific, and will depend on the relative magnitude of the two component measures, and the degree to which the analyst wishes to trade off efficiency with level balance. This trade-off is investigated in this paper.

Table 7 presents a summary of how the constraints are handled under the RBA and CBA. For each constraint type, the table presents whether it is categorised as across or within choice task, the motivation for its implementation, the stage (i.e., step) of the RBA and CBA it is implemented in, and for each algorithm how difficult it is to implement.

Table 7: Integration of various constraints into row and column based algorithms

Constraint type	**Motivation**	**Stage of algorithm and difficulty to implement**	
		Row-based algorithm	**Column-based algorithm**
Across choice tasks			
Orthogonality	Alternative optimality criterion	Not possible	Per iteration. Can use orthogonal tables as candidate sets of columns for the design
Level balance (hard constraint)	Information gain	Per iteration. Hard (design rejections every iteration, could also accept near level balance using hard constraints)	Design population. Easy.
Level balance (soft constraint)	Information gain	Per iteration. Moderate difficulty.	Design population. Easy.
Prevent repeated choice tasks	Prevent duplication	Candidate generation (if a single candidate set across all alternatives). Easy.	Per iteration. Moderate difficulty (some designs rejected).
Subsets of alternatives	Complexity/fatigue	N/A	N/A
Subsets of attributes	Complexity/fatigue	N/A	N/A
Within choice tasks			
Within choice task constraints for realism and plausibility	Plausibility/consistency	Candidate generation. Moderate difficulty.	Candidate generation. Moderate difficulty.
Prevent duplicate unlabelled alternatives	Prevent duplication	Candidate generation. Easy.	Per iteration. Moderate difficulty (some designs rejected).
Scenario attributes	Plausibility/consistency	Candidate generation. Easy	Candidate generation. Moderate difficulty.
Prevent dominance	Plausibility and information gain	Candidate generation. Easy.	Per iteration. Moderate difficulty (some designs rejected).
Combination of across and within choice tasks			
Level balance (hard constraint) + any within choice task constraints	Various	Per iteration. Hard (as above, but (near) balance may be even harder due to constraints).	Design population. Hard.
Level balance (soft constraint) + any within choice task constraints	Various	Per iteration. Moderate difficulty.	Design population. Moderate difficulty.

6. Case studies

In this section we will present two case studies, drawing from two of the examples in the introduction, to illustrate the operation of the proposed row and column based algorithms. Each case study contains a selection of constraints of various types, where the impact of these constraints on the design efficiency, as measured by the *d*-error, will be examined, as will the trade-off between efficiency and a soft level balance constraint.

6.1 Case study 1: motorway choice

In this example, the choice tasks contain three unlabelled motorway alternatives {A, B, C}. The alternatives as described by the following attributes and levels:

- Travel time on motorway {20, 25, 30 minutes}
- Running costs {$4, $4.50, $5}
- Toll {$0, $2, $3}
- Toll payment options {electronic tag (E-tag) only, cash or E-tag, not applicable (N/A)}

Toll payment options is dummy coded, whilst the remaining attributes employ linear coding. The design is generated with 12 choice tasks.

Since the choice tasks are unlabelled, it is necessary to check for dominance and repeated alternatives within each choice task. Additional WCT constraints are specified on the grounds of plausibility:

- If (A.toll=0) and (A.time<B.time), then reject the choice task. Repeat the same logic for all pairs of alternatives.
- If (A.toll=0) then (A.pay={N/A}). Repeat for all alternatives.
- If (A.toll>0) then (A.pay={Cash or E-tag, E-tag only}). Repeat for all alternatives.

These suggest that the design attributes are highly interrelated within each choice task, in terms of logical constraints that need to be imposed. Toll, time and pay are linked not only within each alternative, but across alternatives as well. A full factorial of these nine attributes (three attributes by three alternatives) contains 19683 combinations, yet of these only 2475 satisfy the PAR constraints. It is in designs such as this that the benefits of the bottom up construction of the constrained factorial (as opposed to enumeration of the full factorial) are realised. As with all designs, an additional constraint is imposed that the same choice task cannot be repeated.

The CBA failed to generate an initial design, due to the very high number of WCT constraints. Whilst we remain agnostic as to the appropriateness of row and column based algorithms, and suggest that the best approach may be case specific, the CBA may not even be feasible if there are too many WCT constraints. The analyst generating an experimental design should be aware of this possibility.

The remaining designs were generated with RBAs. For each design and algorithm specification, six designs were generated and the average of the outputs are reported. The first specification is without WCT constraints. Four more specifications introduced WCT constraints, with varying weights α_{LIB} applied to the level balance criterion LIB in the objective function, and weights of 1-α_{LIB} applied to the *d*-error. In addition to the *d*-error, we report the LB and ADFLB measures introduced in Section 5.2. Level balance is not attempted for the dummy coded attributes, where we will investigate what amount of balance results with this approach.

As can be seen in Table 8, the RBA without constraints or level balance generates the most efficient design by some margin. The level balance is fairly poor, and an interrogation of the designs reveals a tendency for there to be an overrepresentation of the end point levels, and an underrepresentation of the middle level, consistent with Rose et al. (2011).

Table 8: Motorway choice results

Algorithm	WCT constraints	α_{LIB}	*d*-error	Level balance (LB)		Average distance per attribute from level balance (ADFLB)	
				Continuous	Dummy coded	Continuous	Dummy coded
RBA	No	0	0.1137	67.13%	75.69%	5.26	3.89
RBA	Yes	0	0.1912	65.51%	70.14%	5.52	4.78
RBA	Yes	0.25	0.1975	81.71%	77.78%	2.93	3.56
RBA	Yes	0.5	0.2037	89.58%	79.17%	1.67	3.33
RBA	Yes	0.75	0.2174	96.06%	80.56%	0.63	3.11

The imposition of WCT constraints leads to a large deterioration in statistical efficiency, with the average *d*-error increasing from 0.1137 to 0.1912. We argue that this should not be a concern, as the validity must be questioned of any conclusions reached from data that contain choice tasks that are logically inconsistent or implausible. Behavioural plausibility should be prioritised over statistical efficiency. The addition of the WCT constraints leads to a small drop in level balance, likely because the constraints disproportionately eliminate some attribute levels more than others.

The final three specifications of design and algorithm optimise jointly on the *d*-error and LIB, with α_{LIB} = 0.25, 0.5 and 0.75. The *d*-errors increase by a small amount, compared to when no weight is placed on level balance in the optimisation function. With $\alpha_{LIB} = 0.75$, the *d*-error is 13.7 percent larger than $\alpha_{LIB} = 0$. Level balance, however, improves dramatically, increasing from 65.51 percent to 96.06 percent. Thus, in this example, near level balance can be achieved with only a small decrease in efficiency.

For the dummy coded attributes, the balance level roughly follows the same pattern as the attributes with linear coding across specifications, however the difference in the magnitude of the balance is much more muted. The lower degree of balance suggests that an attempt to fully balance the levels may be misguided. The advantage of the dummy coding approach is that which levels are over or under represented (relative to the balance condition) will be guided by the associated information gain, rather than being merely arbitrary.

6.2 Case study 2: mode choice

This mode choice example contains three labelled alternatives: train, bus and light rail. The alternatives as described by the following attributes and levels, all with linear coding:

- Percent of seats occupied {50, 60, 70, 80, 90, 100%}
- Number standing (for train, light rail) {0, 15, 30, 45, 60, 75}
- Number standing (for bus) {0, 5, 10, 15, 20, 25}
- Number of transfer {0, 1}
- Transfer time {0, 5, 10, 15 minutes}
- Travel time {20, 25, 30, 35 minutes}
- Fare {$3.50, $4.00, $4.50, $5.00, $5.50, $6.00}

The design is generated with 24 choice tasks.

As the experiment is labelled, no repeated alternative constraints or dominance checks are imposed, although the dominance check might be plausible if modal preferences are not considered. The following PAR constraints are applied for each alternative:

- If (seats<90%) then (standing=0)
- If (transferNumber=0) then (transferTime=0)
- If (transferNumber>0) then (transferTime>0)

Unlike in case study 1, none of the clusters resulting from the PAR constraints span alternatives. The design is considerably less constrained within the choice tasks. Each alternative contains two clusters. The first links seats and standing, and has a full factorial of 36 combinations, of which 16 are valid. The second links the number of transfers and transfer time, has a full factorial of eight combinations, of which four are valid.

The results are presented in Table 9. Consider first the RBA findings. Once again, WCT constraints lead to a large deterioration in efficiency, with the *d*-error dropping from 0.00222 to 0.00280, and a drop in level balance, from 73.89 to 62.3 percent. Two new values of α_{LIB} were tested, 0.05 and 0.1, due to the small magnitude of the *d*-error. With $\alpha_{LIB} = 0.05$, level balance increases notably from 62.3 to 77.55 percent, although with a further large deterioration in *d*-error to 0.00335. Increasing α_{LIB} further does improve level balance by a small amount, but with a very high cost in terms of efficiency. This suggests that 100 percent balance cannot be achieved and the measure is asymptotic at some lesser value. The practical implication of this is that for any given case the analyst should test various α_{LIB} values and determine the best trade-off of efficiency and level balance.

Table 9: Mode choice results

Algorithm	WCT constraints	α_{LIB}	*d*-error	Level balance (LB)	Average distance per attribute from level balance (ADFLB)
RBA	No	0	0.00222	73.89	9.65
RBA	Yes	0	0.00280	62.30	13.52
RBA	Yes	0.05	0.00335	77.55	7.81
RBA	Yes	0.1	0.00357	79.66	7.06
RBA	Yes	0.25	0.00419	80.92	6.61
RBA	Yes	0.5	0.00474	81.16	6.52
RBA	Yes	0.75	0.00490	81.07	6.57
CBA	No	Full balance	0.00223	100.00	0.00
CBA	Yes	Full balance	0.00310	80.00	6.67

Unlike in the previous case study, the RBA algorithm does work here, due to there being fewer WCT constraints. When the WCT constraints are ignored, 100 percent level balance can be achieved for negligible loss of efficiency, compared to the RBA. With the WCT constraints imposed on the CBA, a level balance of 80 percent is achieved with a *d*-error of 0.0031, which compares favourably to the RBA with $\alpha_{LIB} = 0.1$ with a level balance of 79.66 percent and *d*-error of 0.00357. This suggests that the RBA may be more appropriate in some circumstances, likely when the WCT constraints are not heavy.

7. Discussion and conclusions

In this paper, we have detailed various constraints that often need to be imposed on stated choice experimental designs. These include level balance, the prevention of duplicated alternatives and choice tasks, dominance, and plausibility and realism constraints that need to be imposed within a choice task. For the latter, three rule structures are suggested, which allow these PAR constraints to be easily specified by the analyst. Row and column based algorithms are developed that respect all constraints, except for level balance, which is treated as a soft constraint. To that end, a level (im)balance measure is introduced, that can additionally be used both to interrogate the level balance properties of a design. Both of these algorithms are tested on two case studies.

We find that our CBA may perform well, or even better than our RBA, if a limited number of WCT constraints must be satisfied. As the number and complexity of these constraints increases, however, the CBA performs less well, and may even fail. In these circumstances, the RBA is superior. If level balance is sought, and dummy or effects coding is not desired or practical, then applying soft constraints on level balance is effective, within the confines that may be imposed by the WCT constraints. However, this will come at a cost of efficiency. The analyst can test different weights and make a decision, where this may also reveal the asymptotic properties of the level balance in the design. Our key recommendation is to always enforce all constraints, with the exception of level balance.

This paper focuses on the generation of experimental designs that respect constraints. Also of interest, but not covered here, is how to handle existing choice data that contain choice tasks which violate PAR or other constraints. The analyst may wish to remove the choice tasks entirely, or perhaps allow the scale of the model to vary as a function of the plausibility and realism of each choice task, as suggested by Bliemer et al. (2014) for the case of dominance. This will remain an area for future research, as will further refinement of the algorithms, and the development of rules that allow the algorithms to be run with minimum user input.

Glossary

ACT	across choice task
ADFLB	average distance from level balance
CBA	column-based algorithm
LB	level balance
LIB	level imbalance
PAR	plausibility and realism
RBA	row-based algorithm
WCT	within choice task

References

Bliemer, M.C.J., Rose, J.M. and Chorus, C. (2014) Dominancy in stated choice surveys and its impact on scale in discrete choice models. Paper submitted for *The 10th International Conference on Transport Survey Methods*, Leura, Australia, November 2014.

ChoiceMetrics (2012) Ngene 1.1.1 User Manual & Reference Guide, Australia.

Cook, R.D. and Nachtsheim, C.J. (1980) A comparison of algorithms for constructing exact D-optimal designs. *Techometrics* 22, pp. 315-324.

Daly, A., Hess, S., Patruni, B., Potoglou, D. and Rohr, C. (2012) Using ordered attitudinal indicators in a latent variable choice model: a study of the impact of security on rail travel behaviour. *Transportation*, 39, pp. 267-297.

Huber, J. and Zwerina, K. (1996) The importance of utility balance in efficient choice designs. *Journal of Marketing Research*, 33, pp. 307–317.

Kessels, R., Jones, B. and Goos, P. (2011) Bayesian optimal designs for discrete choice experiments with partial profiles. *Journal of Choice Modelling*, 4, pp. 52–74.

Kuhfeld, W. F. and Tobias, R.D. (2005) Large factorial designs for product engineering and marketing research applications. *Technometrics*, 47(2), pp. 132-141.

Rose, J.M., Bain, S. and Bliemer, M.C.J. (2011) Experimental Design strategies for Stated Preference Studies Dealing with Non Market Goods. In Bennett, J. (ed.), International Handbook on Non-Marketed Environmental Valuation, Edward Elgar, Cheltenham, pp. 273-279.

Rose, J.M., Bliemer, M.C.J., Hensher, D.A. and Collins, A.T. (2008) Designing Efficient Stated Choice Experiments in the Presence of Reference Alternatives. *Transportation Research Part B*, 42, pp. 395-406.

Rose, J.M. and Hensher, D.A. (2006) Handling individual specific availability of alternatives in stated choice experiments. In Stopher, P.R. and Stecher, C. (eds.), Travel Survey Methods: Quality and Future Directions, Elsevier Science, Oxford, pp. 347-371.

Rose, J.M., Louviere, J.J. and Bliemer, M.C.J. (2013) Efficient stated choice designs allowing for variable choice set sizes. *International Choice Modelling Conference*, Sydney, Australia, 3rd-5th July 2013.

Sándor, Z. and Wedel, M. (2001) Designing conjoint choice experiments using managers' prior beliefs. *Journal of Marketing Research*, 38, pp. 430–444.

Quan, W., Rose, J.M., Collins, A.T. and Bliemer, M.C.J. (2011) A comparison of algorithms for generating efficient choice experiments. Working paper ITLS-WP-11-19. The University of Sydney.