Journal of Statistical Software

June 2020, Volume 94, Issue 2. doi: 10.18637/jss.v094.i02

R Package OBsMD for Follow-Up Designs in an Objective Bayesian Framework

Laura Deldossi
Universitá Cattolica del Sacro Cuore

Marta Nai Ruscone
Universitá degli Studi di Genova

Abstract

Fractional factorial experiments often produce ambiguous results due to confounding among the factors; as a consequence more than one model is consistent with the data. Thus, the practical problem is how to choose additional runs in order to discriminate among the rival models and to identify the active factors. The R package **OBsMD** solves this problem by implementing the objective Bayesian methodology proposed by Consonni and Deldossi (2016). The main feature of this approach is that the follow-up designs are obtained through the use of just two functions, `OBsProb()` and `OMD()` without requiring any prior specifications, being fully automatic. Thus **OBsMD** provides a simple tool for conducting a design of experiments to solve real world problems.

Keywords: Bayesian design of experiments, screening experiments, Bayesian model selection, model discrimination.

1. Introduction

Fractional factorial designs (often referred to as screening experiments) are generally used during the early stage of an investigation when there is limited scientific knowledge. Unfortunately, such designs often do not lead to unequivocal conclusions regarding the model structure and the combination of factors that influence the response variable. From a Bayesian perspective, this means that the posterior distributions will be flat on both model and factor spaces. In these circumstances, a follow-up design is needed, as the aim is to choose extra runs in order to resolve this ambiguity.

The strategy proposed by Consonni and Deldossi (2016) and implemented in the **OBsMD** package (Nai Ruscone and Deldossi 2020) for R (R Core Team 2020), is explained in Section 2 and illustrated via a tutorial in Section 5.

This proposal represents the *objective* Bayesian version of the criterion proposed by Meyer, Steinberg, and Box (1996), implemented in the R package **BsMD** (Barrios 2020) based on the

Fortran 90 bundle **mdopt** written by Meyer (1996). The **OBsMD**'s core is written in a Fortran 90 bundle obtained by modifying in a suitable way, because of the adoption of the objective priors, the package **mdopt** of Meyer (1996). The program was converted to subroutines to be run from the R package. The package is available from the Comprehensive R Archive Network (CRAN) at `https://CRAN.R-project.org/packages=OBsMD`. The adoption of an *objective* Bayesian approach (Berger and Pericchi 2001), where the priors are derived by default methods based on the statistical model, implies that no prior specifications are required to be provided by the users so that the approach is fully automatic.

The paper is organized as follows. After a brief recall of classical strategies for augmenting an experimental design, Section 2 presents the *objective* Bayesian approach implemented in the R package **OBsMD** through the use of an example. A discussion of the choice of some relevant arguments for the functions `OBsProb()` and `OMD()` is reported in Section 3. Section 4 schematically introduces the functions `OBsProb()` and `OMD()` and their arguments. Finally a tutorial using the `MetalCutting` dataset included in **OBsMD** is reported in Section 5.

2. Follow-up design in an objective Bayesian framework

Several different techniques and strategies exist for augmenting an experimental design: foldover (Montgomery and Runger 1996), semifolding (Mee and Peralta 2000), the D-optimal design approach to augmentation (Goos and Jones 2011), a Bayesian strategy in the context of model discrimination (Meyer *et al.* 1996 and later Consonni and Deldossi 2016). The package **OBsMD** implements the last of these proposals. To introduce the reader to the approach, Section 2.1 briefly explains – through the use of an example – when the researcher needs to augment a design and how he/she can choose the additional runs. The main results of Consonni and Deldossi (2016) are recalled in Section 2.2 together with the main output of the package **OBsMD** applied to the previous example. Finally Section 2.3 illustrates how the original **mdopt** program written by Meyer (1996) has been modified because of the adoption of the objective priors.

2.1. Aim of the package

Screening experiments are employed in the initial stage of investigation to identify the significant factors, i.e., those which affect the response variable from a list of many potential ones. We refer to this kind of factors as *active factors*; see Box, Hunter, and Hunter (1978) or Montgomery (2006) for references on this topic.

Consider for instance a response variable which is potentially influenced by k categorical factors. To discover which factors are active, the experimenter usually considers the designs 2^k, i.e., designs where all the k factors can take on only two values (*low* $= -1$, *high* $= +1$). A *run* specifies the level of each factor at which the experiment is conducted. Ideally, one would want to experiment with all possible 2^k runs (i.e., all the combination of the levels of the k factors). However, as the number k of the factors increases, the amount of experimental effort requested may be too high for an initial experiment. For example, a complete replicate of the 2^6 design requires 64 runs. In this design only 6 of the 63 degrees of freedom correspond to main effects, and only 15 to two factor interactions. The remaining 42 degrees of freedom are associated with interactions higher than 2. If the experimenter can reasonably assume that certain high-order interactions are negligible, information on

Factors	Low (-1)	High $(+1)$
A – Tool speed (rpm)	2700	3200
B – Workpiece speed (mm/min)	203	330
C – Depth of cut(mm)	0.5	1.0
D – Coolant	Off	On
E – Direction of cut	Conventional	Climbing
F – Number of cut	1	2

Table 1: The factors and their levels in the `MetalCutting` dataset.

main effects and low-order interactions may be obtained by running only a "fraction" of the complete factorial experiment.

These are the so-called fractional factorial experiments 2^{k-q}, corresponding to one half fraction when $q = 1$, one quarter fraction when $q = 2$, ... and so on. Refer, e.g., to the package **FrF2** (Grömping 2014, 2020) to create and analyze fractional factorial 2-level designs.

The reduction of runs (from 2^k to 2^{k-q}) introduces aliasing structures in the estimation process with the result that main effects are confounded with some interaction terms; the interested reader is referred to Box *et al.* (1978) and Montgomery (2006). As a consequence, this kind of design works well and gives clear results as long as the interactions among factors, especially those of high order, can be considered negligible. When this is not the case, the aliasing structure underlying these designs can lead to ambiguous conclusions regarding which combinations of factors are really active. Thus it is necessary to augment the design with n_f extra runs. This kind of experiment is called a "follow-up design".

Consider for instance the dataset `MetalCutting` included in the **OBsMD** package. This dataset was used in Mønness, Linsley, and Garzon (2007) and re-examined in Edwards, Weese, and Palmer (2014) with the aim to compare methods to design follow-up experiments. It is introduced here just for illustrative purposes and it is analyzed in Section 5 with the aim to identify the oracle model and active factors.

It is a six-factor (A, B, C, D, E, F) full factorial experiment designed to determine effects having an impact on metal surface finish in a cutting operation. Table 1 displays the $k = 6$ factors involved in the experiment and their levels. The full factorial design is composed of $2^6 = 64$ runs obtained by combining in all possible ways the levels of the 6 factors. It is reported in Table 2 together with the value of the *block*[1] variable and that of the response variable (`Ytransformed`) obtained by a reciprocal transformation of the original `y`; see Mønness *et al.* (2007). Each row of the `MetalCutting` dataset represents a *run* which defines, apart from the block, the level of each factor at which the experiment is conducted. For instance, $run = 1$ with $(-1, -1, -1, -1, 1, -1)$ defines the experiment done when the fifth factor (E) is fixed at high level, while all the other factors (A, B, C, D, F) are set at low level. Often the cost per run and time constraints limit the number of combinations that can be included in the experiment, thus the researcher usually performs only a fraction of the full factorial experiment; as a consequence, he/she will know the value of the response variable only relative to this fraction of runs. Consider for instance the fractional factorial design 2^{6-2}_{IV} with generators

[1]Observe that in Table 2 the variable *block* is a constant (it presents only a level equal to -1). This happens because Table 2 refers to the full data experiment performed on a batch. When it is impossible to perform all the runs of the experiment under homogeneous conditions, we can take into account this circumstance setting the *block* variable at a different level (for instance equal to $+1$).

run	*block*	*A*	*B*	*C*	*D*	*E*	*F*	`Ytransformed`	*run*	*block*	*A*	*B*	*C*	*D*	*E*	*F*	`Ytransformed`
1	−1	−1	−1	−1	−1	1	−1	0.959693	33	−1	1	−1	−1	−1	1	−1	1.064963
2	−1	−1	−1	−1	−1	1	1	0.937207	34	−1	1	−1	−1	−1	1	1	0.744602
3	−1	−1	−1	−1	−1	−1	−1	0.623830	35	−1	1	−1	−1	−1	−1	−1	0.537634
4	−1	−1	−1	−1	−1	−1	1	0.150376	36	−1	1	−1	−1	−1	−1	1	0.069252
5	−1	−1	−1	−1	1	1	−1	1.016260	37	−1	1	−1	−1	1	1	−1	0.986193
6	−1	−1	−1	−1	1	1	1	0.886525	38	−1	1	−1	−1	1	1	1	1.022495
7	−1	−1	−1	−1	1	−1	−1	1.145475	39	−1	1	−1	−1	1	−1	−1	1.157407
8	−1	−1	−1	−1	1	−1	1	1.095290	40	−1	1	−1	−1	1	−1	1	1.222494
9	−1	−1	−1	1	−1	1	−1	1.019368	41	−1	1	−1	1	−1	1	−1	1.012146
10	−1	−1	−1	1	−1	1	1	1.054852	42	−1	1	−1	1	−1	1	1	0.852515
11	−1	−1	−1	1	−1	−1	−1	0.327654	43	−1	1	−1	1	−1	−1	−1	0.307220
12	−1	−1	−1	1	−1	−1	1	0.071225	44	−1	1	−1	1	−1	−1	1	0.046970
13	−1	−1	−1	1	1	1	−1	1.021450	45	−1	1	−1	1	1	1	−1	0.925069
14	−1	−1	−1	1	1	1	1	1.054852	46	−1	1	−1	1	1	1	1	0.975610
15	−1	−1	−1	1	1	−1	−1	0.948767	47	−1	1	−1	1	1	−1	−1	0.987167
16	−1	−1	−1	1	1	−1	1	1.029866	48	−1	1	−1	1	1	−1	1	1.283697
17	−1	−1	1	−1	−1	1	−1	0.961538	49	−1	1	1	−1	−1	1	−1	1.001001
18	−1	−1	1	−1	−1	1	1	0.857633	50	−1	1	1	−1	−1	1	1	0.841043
19	−1	−1	1	−1	−1	−1	−1	0.702741	51	−1	1	1	−1	−1	−1	−1	0.736377
20	−1	−1	1	−1	−1	−1	1	0.173913	52	−1	1	1	−1	−1	−1	1	0.069156
21	−1	−1	1	−1	1	1	−1	0.957854	53	−1	1	1	−1	1	1	−1	1.027749
22	−1	−1	1	−1	1	1	1	0.814996	54	−1	1	1	−1	1	1	1	0.904159
23	−1	−1	1	−1	1	−1	−1	0.898473	55	−1	1	1	−1	1	−1	−1	1.172333
24	−1	−1	1	−1	1	−1	1	1.044932	56	−1	1	1	−1	1	−1	1	0.976563
25	−1	−1	1	1	−1	1	−1	0.807103	57	−1	1	1	1	−1	1	−1	0.961538
26	−1	−1	1	1	−1	1	1	0.962464	58	−1	1	1	1	−1	1	1	0.931099
27	−1	−1	1	1	−1	−1	−1	0.246063	59	−1	1	1	1	−1	−1	−1	0.349406
28	−1	−1	1	1	−1	−1	1	0.039124	60	−1	1	1	1	−1	−1	1	0.042373
29	−1	−1	1	1	1	1	−1	1.150748	61	−1	1	1	1	1	1	−1	1.039501
30	−1	−1	1	1	1	1	1	0.970874	62	−1	1	1	1	1	1	1	1.050420
31	−1	−1	1	1	1	−1	−1	1.102536	63	−1	1	1	1	1	−1	−1	1.206273
32	−1	−1	1	1	1	−1	1	1.148106	64	−1	1	1	1	1	−1	1	1.091703

Table 2: `MetalCutting` dataset (where *run* is the row number in the dataset).

run	A	B	C	D	E	F	`Ytransformed`
62	1	1	1	1	1	1	1.05042017
28	−1	1	1	−1	−1	1	0.03912363
51	1	1	−1	−1	−1	−1	0.73637703
16	−1	−1	1	1	−1	1	1.02986612
64	1	1	1	1	−1	1	1.09170306
21	−1	1	−1	1	1	−1	0.95785441
26	−1	1	1	−1	1	1	0.96246391
42	1	−1	1	−1	1	1	0.85251492
44	1	−1	1	−1	−1	1	0.04697041
23	−1	1	−1	1	−1	−1	0.89847260
39	1	−1	−1	1	−1	−1	1.15740741
1	−1	−1	−1	−1	1	−1	0.95969290
14	−1	−1	1	1	1	1	1.05485232
49	1	1	−1	−1	1	−1	1.00100100
37	1	−1	−1	1	1	−1	0.98619329
3	−1	−1	−1	−1	−1	−1	0.62383032

Table 3: Runs and response variable `Ytransformed` corresponding to the 2^{6-2}_{IV} fractional factorial design with generators $E = ABC$ and $F = ABD$.

$E = ABC$ and $F = ABD$, where only $n = 16$, over the 64 runs, are performed; see Edwards *et al.* (2014).

These 16 runs are identified according to a rule defined by the generator functions[2]. Thus, the experiment corresponding to the runs 62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14, 49, 37, 3 of Table 2 are performed in order to observe the response variable (see Table 3). This design has resolution IV, this means that the main factors are aliased with three-factors interactions. Thus, if we suspect that three-factors interactions are not negligible, i.e., their effect is not null, the results can lead to ambiguous conclusions regarding which of the main factors is really active, since in the estimate of E and F is included also, the effect of the interaction ABC and ABD, respectively. Accordingly the augmentation of the design may help in providing clear responses.

Several different techniques and strategies exist for augmenting an experimental design; their selection depends essentially on the results of the initial experiment and the availability of resources.

Foldover is a classic approach for design augmentation; see Montgomery and Runger (1996). It consists of a follow-up design with a number of runs equal to that of the initial experiment ($n_f = n$), where one or more factors have opposite levels compared with those of the initial experiment.

Semifolding (Mee and Peralta 2000) is a strategy where only half of the foldover runs are performed ($n_f = n/2$). It is usually obtained by selecting from the foldover design the runs related to a desirable level of a prominent factor revealed in the initial experiment.

[2]In this case the generators are $E = ABC$ and $F = ABD$, thus the fractional factorial design corresponds to the runs for which the level of factor E and F are equal to the level of the three-factors interaction ABC and ABD in the design matrix; see Chapter 8 in Montgomery (2006) respectively.

The D-optimal design approach to augmentation (Goos and Jones 2011) is driven by the best model identified in the screening experiment. Conditionally to this model, the additional $n_f > 0$ follow-up runs are chosen to minimize the generalized variance of the parameter estimates. One criticism of this approach regards the follow-up runs which are chosen solely on the basis of improving estimation for the single model identified as the best one in the initial experiment.

To overcome this approach Meyer *et al.* (1996) and later Consonni and Deldossi (2016) propose a Bayesian strategy in the context of model discrimination based on predictive densities for competing models. The idea is the following: since, due to confounded and/or limited data, there are generally multiple models (see Section 2.2) that provide similar or identical response predictions, follow-up runs have to be selected to allow maximum discrimination among all these plausible models. Within the Bayesian construct this means searching for runs able to produce strong diversification of posterior model probabilities. This result is obtained maximizing a model discrimination criterion. To compute the posterior probability of each model, one requires a prior on the model space, as well as a prior on the space of parameters, conditionally on each single model. The above proposals (Meyer *et al.* 1996; Consonni and Deldossi 2016) differ in the choice of these priors. The R package **OBsMD** implements the *objective* Bayesian methodology introduced by Consonni and Deldossi (2016) for this aim. The main feature of this approach is that it does not require prior specifications, being fully automatic.

2.2. Basic formulation, objective priors, main outputs

In the context of experimental designs, it is customary to assume that the response is normally distributed with expectation having a linear regression structure. Specifically, consider k categorical factors, and n experimental runs extracted from a full factorial experiment.

Let M_i be a model which specifies a set of f_i active factors $(0 \leq f_i \leq k)$. Following Consonni and Deldossi (2016) we assume the *effect forcing* assumption (Chipman and Hamada 1996) according to which an interaction AB is active, and hence included in the model, if and only if both its corresponding main effects are active. As a consequence each model M_i will contain a certain number of main factors and all their interactions up to a certain fixed order, typically not higher than three. The assumption of effect forcing was adopted by Consonni and Deldossi (2016) to simplify the illustration of their theory, and also for comparison with the work of Meyer *et al.* (1996). The package could be extended by relaxing this assumption, therefore considers a more flexible approach. For instance two common ways of specifying heredity relationships are strong heredity, where an interaction can be active with probability $p \leq 1$ if both its corresponding main effects are active (under the effect forcing assumption $p = 1$), and weak heredity, where all interactions have at least one parent main effect in the model; see Chipman and Hamada (1996). Another possibility is to assume that interactions may be present in a model only with a certain probability as advocated in Bingham and Chipman (2007). In this way the user is able to incorporate prior opinions on structural aspects of effects; see also Wolters and Bingham (2011).

Notice that the total number of distinct models M_i will be equal to 2^k, including the null model M_0 (no factor is active), and the full model (all factors and interactions are active). For instance, consider the case of $k = 3$ active factors $\{A, B, C\}$. Then the $2^3 = 8$ distinct models are those reported in Table 4: the null model, three models each with only one main

	Active factors	Model M_i	f_i
1	none	1	0
2	A	$1 + A$	1
3	B	$1 + B$	1
4	C	$1 + C$	1
5	A, B	$1 + A + B + A * B$	2
6	A, C	$1 + A + C + A * C$	2
7	B, C	$1 + B + C + B * C$	2
8	A, B, C	$1 + A + B + C + A * B + A * C + B * C + A * B * C$	3

Table 4: List of the total number of possible distinct models under the effect forcing assumption with $k = 3$ and interactions up to the third order.

effect, three models each with two main effects and the related two-factor interactions, the full model that contains, besides the three main effects, also all the two-factor interactions AB, AC, BC, as well as the three-factor interaction ABC, if interactions up to third order have been chosen. Thus a model is fully defined only fixing its main possibly active factors. We will assume that under model M_i

$$y|\beta_0, \beta_i, \sigma, M_i \sim N_n(X_0\beta_0 + X_i\beta_i, \sigma^2 I_n), \tag{1}$$

where X_0 represents an $n \times t_0$ design matrix containing all columns which are common to all models (typically the intercept and possibly the block factors), while X_i represents an $n \times t_1$ matrix whose columns are related to each specific effect under model M_i, i.e., the main factors and their interactions up to the desired order.

With reference to the prior on the model space, it is customary to assume that each factor is included in any particular model with some probability π independently of the other factors. If M_i contains f_i specific active factors, $f_i \in \{0, 1, \ldots, k\}$, then

$$\mathsf{P}(M_i|\pi) = \pi^{f_i}(1-\pi)^{k-f_i}. \tag{2}$$

In the full objective Bayesian perspective of Consonni and Deldossi (2016), π should be regarded as an uncertain quantity with its own distribution that they assume equal to $\pi \sim Beta(a, b)$. Then integrating (2) with respect to this prior yields

$$\mathsf{P}(M_i) = \int_0^1 \pi^{f_i}(1-\pi)^{k-f_i} p(\pi) d\pi = B(a + f_i, b + k - f_i)/B(a, b), \tag{3}$$

where $B(\cdot, \cdot)$ is the usual beta function.

This kind of prior has been studied by Scott and Berger (2010), who show that it enjoys the attractive property of "multiplicity adjustment", i.e., it provides control – unlike prior (2) – over the addition of spurious covariates, when performing variable selection in a linear model. Usually the values $(a = 1, b = 1)$, corresponding to the uniform distribution, are adopted. The alternative choice $(a = 1, b = k + 1)$ has been advocated to achieve a stronger sparse model effect. Moreover, the main difference with such a prior is that the choice $(a = 1, b = k + 1)$ gives more weight to more parsimonious models, relative to $(a = 1, b = 1)$.

At this point, conditionally to a specific model M_i, a prior distribution on the parametric space is needed. In the following the intercept β_0 and the error variance σ^2 are regarded as "common" parameters across models, while β_i is the model-specific vector of regression parameters.

Consonni and Deldossi (2016) adopt a robust prior that does not require the user to specify any tuning parameters and that satisfies all the *desiderata* a prior should have from an objective model selection perspective; see Bayarri, Berger, Forte, and García-Donato (2012). According to their approach, a non-informative prior proportional to 1 over σ is given to the common parameters β_0 and σ, whereas the prior of the specific parameters β_i of the model is a mixture of normal distributions with weight given by a new (proper) distribution prior. The analytic expression of this hierarchical g-prior for model selection is

$$p^R(\beta_0, \beta_i, \sigma\, M_i) \quad = \quad p(\beta_0, \sigma) p^R(\beta_i\, \beta_0, \sigma\, M_i) = \sigma^{-1} \times \int_0^\infty N_{t_i}(\beta_i\, 0, g\Sigma_i) p^R(g\, M_i) dg, \quad (4)$$

where $p(\beta_0, \sigma)$ is the prior on the common parameters shared by all models, $\Sigma_i = \sigma^2 (V_i^\top V_i)^{-1}$, $V_i = (I_n - X_0(X_0^\top X_0)^{-1} X_0^\top) X_i$ and

$$p^R(g\, M_i) = \frac{1}{2} \left[\frac{1+n}{t_i + t_0} \right]^{1/2} (g+1)^{-3/2} 1_{\left(\frac{1+n}{t_i+t_0} - 1, \infty\right)}(g),$$

with $1_A(t) = 1$ if $t \in A$ and 0 otherwise; refer to Consonni and Deldossi (2016) for further details.

Combining the prior on the model space with the parameters' prior, it is possible to obtain the posterior distribution of each model in closed form

$$\mathsf{P}(M_i|y) = \frac{BF_{i0}(y) P_{i0}}{1 + \sum_{j \neq 0} BF_{j0}(y) P_{j0}}, \quad (5)$$

where P_{j0} is the prior odds of model M_j relative to M_0 implied by (3), and $BF_{j0}(y)$ is defined as

$$BF_{i0}(y) = \left[\frac{n+1}{t_i + t_0} \right]^{-t_i/2} \times \frac{Q_{i0}(y)^{-(n-t_0)/2}}{t_i + 1} {}_2F_1 \left[\frac{t_i+1}{2}; \frac{n-t_0}{2}; \frac{t_i+3}{2}; \frac{(1 - Q_{i0}(y)^{-1})(t_i + t_0)}{n+1} \right], \quad (6)$$

where ${}_2F_1$ is the standard hypergeometric function (see Abramowitz and Stegun 1965, p. 555) and $Q_{i0}(y) = SSE_i(y)/SSE_0(y)$ is the ratio of the sum of squared errors of models M_i and M_0. Summing the probabilities $\mathsf{P}(M_i|y)$, $i = 1, \ldots, 2^k$ over all models that include a specific factor, it is possible to obtain the posterior probability that it is active. For instance the posterior probability that factor j is active, $P_j(y)$, is given by

$$P_j(y) = \sum_{\{M_i: \text{factor j is active}\}} \mathsf{P}(M_i|y). \quad (7)$$

Function `OBsProb()` of the R package **OBsMD** computes the posterior probabilities of models and factors based on objective priors implementing formula (5) and (7), respectively. Two further outputs of `OBsProb()` are:

(a) The value of the normalized Shannon entropy index (hereafter named "Shannon index")

$$\left(-\sum_{i=1}^{s} \mathsf{P}(M_i|y) \cdot \ln(\mathsf{P}(M_i|y))\right) / \ln(s),$$

where s is the total number of distinct models[3]. The Shannon index will assume values close to zero when there is low model uncertainty, i.e., there is one model with high posterior probability (thus, in the model discrimination framework, low values of the Shannon index are better).

(b) The value of the coefficient of variation (hereafter named "CV") that is the ratio of the standard deviation to the mean of the posterior probability of the factors (see (7)). The CV among posterior probabilities of factors is high when we have a clear evidence in favor of some factors (thus, high values of CV are better).

Just to give an idea of how `OBsProb()` works, consider for instance to have performed the 2_{IV}^{6-2} fractional factorial design of Table 3 and thus to know the values of the response variable only for these 16 combinations of factor levels.

```
R> library("OBsMD")
R> data("MetalCutting", package = "OBsMD")
R> X <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 2:7]
R> y <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 8]
```

The code reported in the following will produce the posterior probabilities for each of the 2^6 distinct models (`nTop = 64`) and the posterior probabilities for the factors $\{A, B, C, D, E, F\}$ (namely 1, 2, 3, 4, 5, 6 in the last column of the model probabilities output), under the assumption of considering models with interactions up to the second order (`mInt = 2`) and $Beta$(`abeta` $= 1$, `bbeta` $= k + 1 = 7$) as prior on the model space.

```
R> es3.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 7, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
```

By changing the `mInt` input to 3, interactions up to the third order will be included. We refer the reader to Section 4 for an extensive illustration of the arguments of `OBsProb()`. Using the `print` method one can inspect the `OBsProb()` output. By suitably setting the argument `nTop`, which is the number of the top ranked models to print, you will get a glimpse of the output.

```
R> print(es3.OBsProb, nTop = 64)
```

```
 Design Matrix:
    A  B  C  D  E  F
62  1  1  1  1  1  1
28 -1  1  1 -1 -1  1
```

[3]If a model M_i has $\mathsf{P}(M_i|y) = 0$, then the corresponding product $\mathsf{P}(M_i|y) \cdot \ln(\mathsf{P}(M_i|y))$ is set to 0.

```
51  1  1 -1 -1 -1 -1
16 -1 -1  1  1 -1  1
64  1  1  1  1 -1  1
21 -1  1 -1  1  1 -1
26 -1  1  1 -1  1  1
42  1 -1  1 -1  1  1
44  1 -1  1 -1 -1  1
23 -1  1 -1  1 -1 -1
39  1 -1 -1  1 -1 -1
1  -1 -1 -1 -1  1 -1
14 -1 -1  1  1  1  1
49  1  1 -1 -1  1 -1
37  1 -1 -1  1  1 -1
3  -1 -1 -1 -1 -1 -1

 Response vector:
1.05 0.039 0.736 1.03 1.092 0.958 0.962 0.853 0.047 0.898 1.157 0.96 1.055
1.001 0.986 0.624

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
    16      6      0      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.144
1      A 0.014
2      B 0.011
3      C 0.345
4      D 0.835
5      E 0.801
6      F 0.345

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.329 0.013  3      4,5,6
M2  0.329 0.013  3      3,4,5
M3  0.144 0.114  0      none
M4  0.117 0.039  2      4,5
M5  0.034 0.082  1      4
M6  0.008 0.101  1      5
M7  0.003 0.014  4      1,3,4,5
M8  0.003 0.014  4      1,4,5,6
M9  0.003 0.074  2      4,6
M10 0.003 0.074  2      3,4
M11 0.003 0.116  1      3
M12 0.003 0.116  1      6
```

```
M13 0.002 0.049  3      1,4,5
M14 0.002 0.050  3      2,4,5
M15 0.002 0.122  1      1
M16 0.002 0.019  4      2,3,4,5
M17 0.002 0.019  4      2,4,5,6
M18 0.002 0.122  1      2
M19 0.001 0.063  4      3,4,5,6
M20 0.001 0.081  3      3,4,6
M21 0.001 0.094  2      1,4
M22 0.001 0.094  2      2,4
M23 0.001 0.116  2      3,6
M24 0.000 0.102  2      5,6
M25 0.000 0.102  2      3,5
M26 0.000 0.050  4      1,2,4,5
M27 0.000 0.115  2      1,5
M28 0.000 0.102  3      3,5,6
M29 0.000 0.117  2      2,5
M30 0.000 0.087  4      1,3,4,6
M31 0.000 0.096  3      1,3,4
M32 0.000 0.096  3      1,4,6
M33 0.000 0.088  4      2,3,4,6
M34 0.000 0.097  3      2,4,6
M35 0.000 0.097  3      2,3,4
M36 0.000 0.128  2      1,2
M37 0.000 0.105  3      1,2,4
M38 0.000 0.133  2      1,3
M39 0.000 0.133  2      1,6
M40 0.000 0.135  2      2,3
M41 0.000 0.135  2      2,6
M42 0.000 0.124  3      1,3,6
M43 0.000 0.125  3      2,3,6
M44 0.000 0.107  4      1,2,4,6
M45 0.000 0.107  4      1,2,3,4
M46 0.000 0.109  4      1,3,5,6
M47 0.000 0.110  4      2,3,5,6
M48 0.000 0.132  3      1,3,5
M49 0.000 0.132  3      1,5,6
M50 0.000 0.135  3      1,2,5
M51 0.000 0.135  3      2,3,5
M52 0.000 0.135  3      2,5,6
M53 0.000 0.158  3      1,2,6
M54 0.000 0.158  3      1,2,3
M55 0.000 0.202  4      1,2,3,5
M56 0.000 0.202  4      1,2,5,6
M57 0.000 0.134  4      1,2,3,6
M58 0.000 0.000  5      1,3,4,5,6
M59 0.000 0.000  5      2,3,4,5,6
```

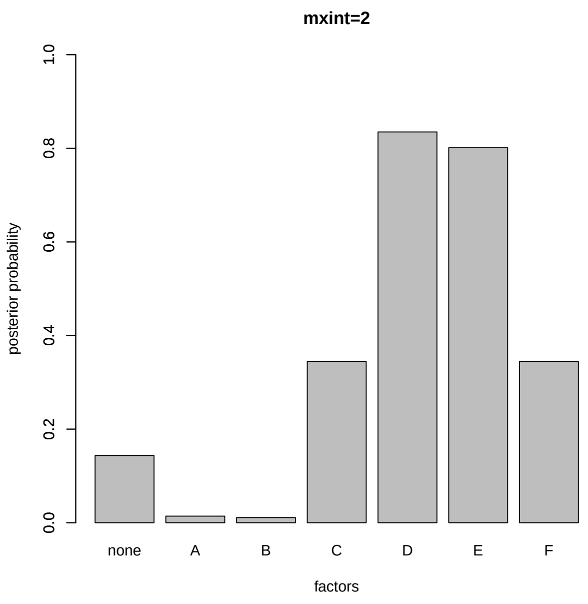

Figure 1: Posterior probabilities of factors for screening the 2^{6-2}_{IV} design with generators $E = ABC$ and $F = ABD$ (prior $Beta(1, 7)$).

```
M60 0.000 0.000  5      1,2,4,5,6
M61 0.000 0.000  5      1,2,3,5,6
M62 0.000 0.000  5      1,2,3,4,5
M63 0.000 0.000  5      1,2,3,4,6
M64 0.000 0.000  6      1,2,3,4,5,6

 Shannon index:
[1] 0.401

 CV:
[1] 0.844
```

```
R> plot(es3.OBsProb, main = "mxint = 2")
```

The `OBsProb()` output shows the posterior probabilities that factors are active ("Factor probabilities", (7)), the posterior distribution of the models ("Model probabilities", (5)), the normalized Shannon entropy index of the model posterior probabilities ("Shannon index") and the coefficient of variation of the posterior probabilities of the factors ("CV").

With regard to the previous example we can observe that none of the models has a posterior probability greater than 0.5. In particular the same mass of probability (0.329) is centered on two models, the first containing as active factors D, E, F (namely 4, 5, 6 in the output) and the second the factors C, D, E (3, 4, 5 in the output). The Shannon index is equal to 0.401. Looking at the factor probabilities reported in Figure 1 we could conclude that factors D and E are active (their posterior probabilities are greater than 0.8), but we have less evidence in favor of C and F since their posterior probabilities are around 0.35. The CV index is 0.844. Thus, augmenting the design is recommended.

To choose the n_f follow-up runs which best discriminate between competing models, we look for those runs corresponding to the maximum value of the following model discrimination criterion (MD):

$$MD = \sum_{i \neq j} \mathsf{P}(M_i|y)\mathsf{P}(M_j|y) KL(m(\cdot|y, M_i), m(\cdot|y, M_j)), \tag{8}$$

where $m(\cdot|y, M_i)$ is the (posterior) predictive density for the vector of follow-up observations, and

$$KL(f, g) = \int f(x) \log \frac{f(x)}{g(x)} dx \tag{9}$$

is the Kullback-Leibler (KL) divergence of the density f from g. Notice that MD is a weighted average of the KL-divergences between all pairs of predictive distributions for the follow-up observation introduced by Meyer *et al.* (1996). The closed form expression for the objective MD (OMD), obtained by Consonni and Deldossi (2016) adopting a standard reference prior $p^N(\beta_0, \beta_i, \sigma\, M_i) \propto 1/\sigma$ for prediction purposes is:

$$OMD = \sum_{i \neq j} \mathsf{P}(M_i|y)\mathsf{P}(M_j|y) \frac{1}{2} \left\{ tr({V_j^*}^{-1} V_i^*) + \frac{n - t_i - t_0}{SSE_i} (\hat{y}_i^* - \hat{y}_j^*)^\top {V_j^*}^{-1} (\hat{y}_i^* - \hat{y}_j^*) - n^* \right\}, \tag{10}$$

where $\mathsf{P}(M_i|y)$ is defined in (5), SSE_i is the usual residual sum of squares under model M_i, $\hat{y}_i^* = Z_i^* \hat{\gamma}_i, \quad V_i^* = I_{n^*} + Z_i^* (Z_i^\top Z_i)^{-1} {Z_i^*}^\top$, having set $Z_i^* = [X_0 \vdots X_i^*]$, where X_i^* is the design matrix for the additional runs.

Function `OMD()` of the R package **OBsMD** computes the n_f extra-runs according to this approach.

Going on with the example of Table 3, we want to select the optimal $n_f = 4$ follow-up experimental runs using function `OMD()` with the aim to discriminate among the `nMod = 57` estimable models. The list of the possible follow-up runs in decreasing order according to the value of the function (10) can be obtained using the following code. Again, we refer to Section 4 for an extensive illustration of the function's arguments.

```
R> Xcand <- MetalCutting[, 2:7]
R> Mbest <- as.matrix(combinations(64, 4, 1:64, repeats = TRUE))
R> es3.pp_omd <- OMD(OBsProb = es3.OBsProb, nFac = 6, nBlk = 0, nMod = 57,
+    nFoll = 4, Xcand = Xcand, mIter = 0, nStart = nrow(Mbest),
+    startDes = Mbest, top = 10)
R> summary(es3.pp_omd)
```

```
 Base:
 nRuns    nFac    nBlk maxInt    nMod
    16       6       0       2     57


 OMD:
  nCand    nRuns  maxIter  nStart
     64        4        0  766480
```

```
Top 10 runs:
      OMD r1 r2 r3 r4
1  11.530 12 36 52 59
2  11.530  4 43 52 60
3  11.530 28 36 43 52
4  11.529 20 36 43 60
5  11.529  4 36 43 60
6  11.529 12 36 43 52
7  11.528  4 28 43 52
8  11.528  4 12 52 59
9  11.527 12 20 36 59
10 11.527 28 36 36 43
```

Looking at the output we can see that the first $n_f = 4$ runs corresponding to the highest value of *OMD* are (12 36 52 59). After the augmentation of the design, function `OBsMD()` may be called again, and the posterior probabilities of the models and of the factors recomputed.

```
R> X <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 1:7]
R> y <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 8]
R> TOP_DES <- cbind(blk = rep(1, 4), MetalCutting[es3.pp_omd$TOPDES[1, ],
+    c(-1, -8)])
R> y_TOP_DES <- MetalCutting[es3.pp_omd$TOPDES[1, ], 8]
R> X <- rbind(X, TOP_DES)
R> y <- c(y, y_TOP_DES)
R> es3.aug.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 7, blk = 1,
+    mFac = 6, mInt = 2, nTop = 64)
R> print(es3.aug.OBsProb, nTop = 10)
```

```
[...]

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
    20      6      1      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.034
1      A 0.008
2      B 0.007
3      C 0.055
4      D 0.960
5      E 0.947
6      F 0.861

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.800 0.012  3      4,5,6
M2  0.078 0.035  2      4,5
M3  0.048 0.012  4      3,4,5,6
M4  0.034 0.098  0      none
M5  0.013 0.071  1      4
M6  0.005 0.014  4      1,4,5,6
M7  0.004 0.033  3      3,4,5
M8  0.004 0.015  4      2,4,5,6
M9  0.003 0.086  1      5
M10 0.002 0.060  2      4,6

 Shannon index:
[1] 0.206

 CV:
[1] 0.954
```

Looking at factor probabilities, also displayed in Figure 2 (on the left side), we can observe more clearly which factors appear as active (D, E and F) and which not (C). Furthermore, there is only one model with posterior probability greater than 0.5 and it is just that containing factors D, E and F (4, 5, 6 in the output). Observe that now the Shannon index is lower (0.207 against 0.401) and the CV is greater (0.954 against 0.844), reflecting a reduced heterogeneity among competing models and an increased concentration of probabilities on really active factors, respectively.

The same whole procedure replicated for `mInt = 3` (`OBsProb()` plus `OMD()` with the augmentation of runs 4 11 52 59; see the code in the supplementary material) generates the posterior probability of factors shown in Figure 2 (on the right side). We can observe that the results do not change and they are consistent to those reported in Figure 5 of Edwards *et al.* (2014) who adopted different augmentation approaches for the same dataset. In conclusion this package allows, within the objective Bayesian approach, to reach two different goals. First, it produces the posterior probability of models and factors using the data observed on an initial experiment; then, it provides the identification of the optimal follow-up runs in order to discriminate among competing models.

2.3. Changes in the Fortran code

Our Fortran code is the result of an extensive modification of the original **mdopt** program written by Meyer (1996). In order to adopt the objective priors we had to substantially modify both the expressions of the posterior probability of the models and the model discrimination criterion.

Moreover, in the objective Bayesian approach, the model matrix $[X_0 \vdots X_i]$ is assumed to be of full column rank, so that the number of linearly independent terms in the regression structure cannot exceed n. Then only estimable (non aliased) interactions can be introduced in the models besides the main effects, conditionally on the constraints that $n > t_0 + t_i$. This constraint is not present in the original code.

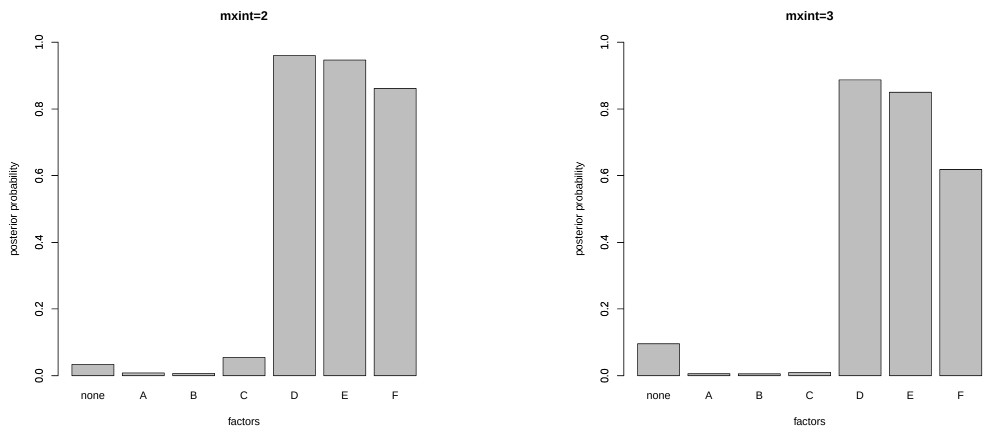

Figure 2: Posterior probabilities of factors for the 2^{6-2}_{IV} design after the addition of the extra runs: (on the left side) $(12-36-52-59)$ associated to the highest *OMD* value (prior $Beta(1,7)$ and `mInt = 2`); (on the right side) $(4-11-52-59)$ associated to the highest *OMD* value (prior $Beta(1,7)$ and `mInt = 3`).

Thus only the original general structure of the program is maintained, i.e., the computation of the OLS estimates of β_i and the corresponding residual sum of squares SSE_i under each model M_i and the exchange algorithm.

As far as the prior assumptions are concerned, we need only to fix the parameter a and b of the *Beta* distribution on the model space (see (3)). On the contrary, the approach of Meyer *et al.* (1996) requires the user to assign a value to π in expression (2) (with the recommended choice being $\pi = 0.25$ to induce factor sparsity in model selection). Furthermore, with reference to the parametric space, they assign a weakly independent $N(0, \gamma^2\sigma^2)$ prior to the components of the model-specific vector of regression parameters β_i, γ being a scale parameter that needs to be fixed.

As a consequence, their expressions of the model posterior probability (see Formula (A.1) in Meyer *et al.* 1996) are different from our expressions (5) and (6), as well as their model discrimination criterion (see Formula (A.9) in Meyer *et al.* 1996) is different from (10) since we adopt a reference prior for prediction purposes. Thus with our proposal the well-known problem of sensitivity to the tuning parameters (π and γ) is overcome.

3. How the package works

In this section the main arguments of the functions `OBsProb()` and `OMD()` are introduced together with a discussion of the theoretical implication of their setting.

Assume that an experimenter has to identify which, among k factors each at two levels, are active on the basis of $n < 2^k$ runs. Due to the *effect forcing* assumption he/she knows that there are 2^k distinct models able to capture the dependence between the response variables and the potential active factors.

To obtain the posterior probability of the competing models through the function `OBsProb()`, the experimenter needs to fix:

- The parameters a and b of the *Beta* distribution for the prior on the model space.

The standard assumption is $a = 1$ and $b = 1$; this choice (equivalent to *uniform* distribution) implies that (3) will be equal to:

$$\mathsf{P}(M_i) = \frac{1}{k+1}\binom{k}{f_i}^{-1}. \tag{11}$$

If we think of the competing models as grouped in $k + 1$ boxes, each defined according to the number of the active factors (from 0 to k), Equation 11 means that a box is selected uniformly at random from the $(k + 1)$ boxes available; next, given the chosen box, a model is selected uniformly at random from those in that box (Scott and Berger 2010). In other words this prior gives the same probability to all the boxes as well as to all the models in each box. Recently, an alternative choice ($a = 1$, $b = k + 1$) has been advocated to achieve a stronger sparse modeling effect. This option gives more weight to more parsimonious models; see the discussion in Consonni and Deldossi (2016).

- The maximum order of interaction among factors to be considered in the model (typically 2 or 3).

  It has to be set according to considerations related to the number k of factors and the dimension n of the screening design. In fact, as we have just mentioned, in the objective Bayesian approach, the model matrix $[X_0 \vdots X_i]$ is assumed to be of full column rank, so that the number $(t_0 + t_i)$ of linearly independent terms in the regression structure cannot exceed n. This condition, along with the effect forcing assumption, implies that sometimes it is not possible to compute the posterior probability for all the 2^k different models. The number of models for which this is possible also depends on the maximum order of interactions since it influences the value of t_i. For instance if $k = 5$ and the initial screening experiment consist of a fractional design 2^{5-2}, depending on whether the maximum order of interactions is set at 2 or 3, we can estimate 26 or 16 distinct models, respectively. In fact the ten models with three active factors are not estimable if the maximum order of interactions is set equal to 3 because $n = t_0 + t_1 = 8$.

  In this context also the presence of a block factor – incrementing the value of t_0 – could represent a restriction in the exploration of potential different models whenever the dimension of the initial experiment n is low.

Analyzing the output of `OBsProb()`, the researcher can observe:

1. None of the factors stands out either as clearly active or clearly inactive. Extra runs are needed in order to resolve this ambiguity. In this case it is possible to apply the function `OMD()` in order to identify, through the *OMD* criterion (10), the optimal n_f runs that allow maximum discrimination among the competing models and factors.

2. There are some factors with zero posterior probabilities (inert factors). In this case the user can decide to drop the inert factors and collapse the initial fractional design on the remaining active factors, so obtaining a replicated fractional factorial design defined only on those factors identified as active. At this point `OBsProb()` has to be called again in order to obtain the posterior probability of models and factors related to the experiment based only on the active factors. If the result does not resolve ambiguity, then the `OMD()` function can be applied (see the example reported in Consonni and

Deldossi 2016). Dropping inactive factors can reduce or eliminate the need of extra runs. This strategy has not been applied in this paper to preserve comparison with follow-up designs in Mønness *et al.* (2007).

3. There are some factors with posterior probability greater than 0.5. In this case the user can decide to stop the procedure or to continue until all factors have been determined to be either inactive or active.

To obtain the optimal follow-up design, some results obtained as output of `OBsProb()` are required. By default they are passed as input to the function `OMD()`, together with the matrix `Xcand` containing the list of the 2^k possible combinations of the levels of the k factors to be considered as potentially active after the previous step. Furthermore the following relevant arguments have to be fixed:

- the dimension n_f of the follow-up design, i.e., the number of additional runs;
- the number of competing models among which the experimenter wants to discriminate. Generally the investigator chooses models with the highest posterior probabilities by examining the output of the function `OBsProb()` where they are ranked in decreasing order.

Moreover the researcher can decide to identify the optimal follow-up design corresponding to the highest *OMD* value using either the exhaustive search strategy or the exchange algorithm.

- Adopting the exhaustive search over all the possible follow-up designs of n_f runs chosen from the 2^k candidates, the experimenter has to give as input to the function `OMD()` the matrix `Mbest` containing all the combinations (with replication) of n_f runs chosen from `Xcand`. `Xcand` is a matrix of dimension $p \times n_f$ obtained using the R function `combinations()`.
- Adopting the exchange algorithm (proposed in Meyer *et al.* 1996) it is possible to avoid the intensive calculation due to the previous choice. According to this option, a design is generated at random from the set of candidate points and the corresponding *OMD* value (10) is computed. Then this initial design is improved by adding that run which most increases the value of *OMD* and removing that which results in the smallest reduction in the criterion. This add/remove procedure is continued until the algorithm converges. In the `OMD()` function the exchange algorithm is set as default option. For general problems there will be a large number of possible follow-up designs and the adoption of the exhaustive search would be excessively time-consuming.

`OMD()` produces the optimal follow-up designs and the corresponding *OMD* values, ranked in decreasing order. After having collected the value of the response variable corresponding to the follow-up design, the function `OBsProb()` can be re-applied in order to obtain the new posterior probabilities of models and factors.

Our procedure should appeal to practitioners because the number n_f of additional runs can be freely chosen. Furthermore it is fully automatic not requiring prior specifications unlike the Meyer *et al.* (1996) approach whose sensitivity to the tuning parameters (π and γ) may represent a problematic issue.

4. Package description

In this section a detailed list of all the arguments of the functions `OBsProb()` and `OMD()` is provided.

Function `OBsProb()` produces posterior probabilities of models and factors based on the objective priors defined in Section 2.2; its arguments are listed in Table 5. The function returns an '`OBsProb`' class object. Function `OMD()` produces follow-up designs, i.e., the extra runs which maximize the model discrimination criterion (10) represented by weighted average of Kullback-Leibler divergences between all pairs of models; its arguments are listed in Table 6. The function returns an '`OMD`' class object. In order to compute `OMD()` we require some output from `OBsProb()`. As we have previously observed, in using `OMD()` the researcher can decide to adopt the exhaustive search strategy or the exchange algorithm. For the exchange algorithm

Arguments	Description
`X`	Design matrix, containing in the first columns block factors if present.
`y`	Response vector.
`abeta`	First parameter of the *Beta* prior distribution on the model space.
`bbeta`	Second parameter of the *Beta* prior distribution on the model space.
`blk`	Number of blocking factors (≥ 0), accommodated in the first columns of matrix `X` (`ncol(X)` $-$ `blk` $= k =$ number of factors).
`mFac`	Maximum number of factors considered in the models.
`mInt`	Maximum order of interactions (≤ 3) among factors included in the models.
`nTop`	Number of models (≤ 100) for which the posterior probability could be computed.

Table 5: Arguments of the function `OBsProb()`.

Arguments	Description
`OBsProb`	Return object of function `OBsProb()`.
`nFac`	Number of factors in the initial experiment.
`nBlk`	Number of blocking factors in the initial experiment.
`nMod`	Number of competing models considered to compute *OMD*.
`nFoll`	Number of additional runs n_f in the follow-up experiment.
`Xcand`	Matrix [`N` $\times$(`nBlk` + `nFac`)] of candidate runs for the follow-up design where `N` generally represents the full $2^{\texttt{nFac}}$ design.
`mIter`	Maximum number of iterations (≥ 0) in the exchange algorithm. When `mIter = 0`, the exchange algorithm is not used.
`nStart`	Number of different designs of dimension `nFoll` to be evaluated by the *OMD* criterion. When the exchange algorithm is used, `nStart` represents the number of random starts to initialize the algorithm; otherwise `nStart = nrow(startDes)`.
`startDes`	Input matrix [`nStart` $\times$ `nFoll`] containing different `nStart` designs to be evaluated by the *OMD* criterion. If the exchange algorithm is used `startDes = NULL`.
`top`	Number of highest *OMD* follow-up designs recorded. By default set to 20.

Table 6: Arguments of the function `OMD()`.

the following setting is required:

- `mIter` > 0;
- `nStart` $=$ number of random starts to initiate the algorithm (default value is 25);
- `startDes = NULL`.

On the other hand, to explore all possible follow-up runs the matrix `Mbest` is previously needed. It is obtained using the Rfunction `combinations()` added to the **OBsMD** package. Furthermore this setting is required:

- `mIter = 0`;
- `nStart = nrow(Mbest)`;
- `startDes = Mbest`.

5. Tutorial using the `MetalCutting` dataset

This tutorial uses the dataset `MetalCutting` included in the **OBsMD** package. See Section 2.1 for its description. It corresponds to a full factorial design 2^6 that a researcher rarely has at his/her disposal, due to the cost per run and time limit constraints. In the following we consider and analyze this dataset only with the aim to have a benchmark to compare the performance of our approach with that of the other methods to specify follow-up designs (see Edwards *et al.* 2014).

The data reported in Table 2 is loaded with

```
R> library("OBsMD")
R> data("MetalCutting", package = "OBsMD")
```

The `ID` column indicates the row number of the `MetalCutting` dataset in Table 2 and it specifies the *run* number. The first column of the `MetalCutting` dataset corresponds to the *block* variable. It is generally used when it is impossible to perform all the runs under homogeneous conditions, for instance on the same day or using a single batch of raw material. In Table 2 all the levels of the *block* variable are set equal to "-1", thus meaning that all the runs are performed under the same conditions. It follows that we do not consider it to analyze the full design, so we put `blk = 0` as input of the `OBsProb()` function and we define `X` as the columns from 2 to 7 of the `MetalCutting` dataset.

```
R> X <- MetalCutting[, 2:7]
R> y <- MetalCutting[, 8]
R> es2.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
```

The function `OBsProb()` computes the posterior probabilities of all the distinct models using data from the full design according to the objective Bayesian framework.

This example exhibits most of the output of the `OBsProb()` function. The function calls a Fortran subroutine. Most of the output is included in `OBsProb`'s output list. This is a list of class '`OBsProb`' with `print`, `plot` and `summary` methods. The design factors are accommodated in the matrix X while y includes the value of the response variable. As reported in Table 5, in the call of the function `OBsProb()` the experimenter can fix:

- The parameters `abeta` and `bbeta` of the *Beta* prior distribution on model space. These parameters are set by default equal to 1 (equivalent to the *uniform* distribution).
- The number `blk` of block in the experiment; this parameter is set by default equal to 0.
- The maximum number `mFac` of factors that the model has to contain (in our example `mFac = 6`, i.e., we are interested in models with possibly all the 6 factors (A, B, C, D, E, F)).
- The maximum order of interactions `mInt` among factors to be considered in the model (in this case its two-factor interactions ($2FIs$), but it is also possible to fix `mInt = 3` in case of interest in $3FIs$).
- The number `nTop` of models for which the posterior probability is reported in the output. This number cannot be greater than the value of `totMod`, shown in the output, which represents the number of models for which the posterior probability has been computed. Observe that `totMod` is less or equal to $2^{\texttt{nFac}}$, where `nFac` represents the number of factors involved in the experiment (in our example $\texttt{totMod} = 2^{\texttt{nFac}} = 64$).

The `print` method returns the results of `OBsProb()`:

```
R> print(es2.OBsProb, nTop = 3)

[...]

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
    64      6      0      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.000
1      A 0.001
2      B 0.000
3      C 0.779
4      D 1.000
5      E 1.000
6      F 1.000

 Model probabilities:
    Prob  Sigma2 NumFac Factors

M1  0.779 0.011  4      3,4,5,6
```

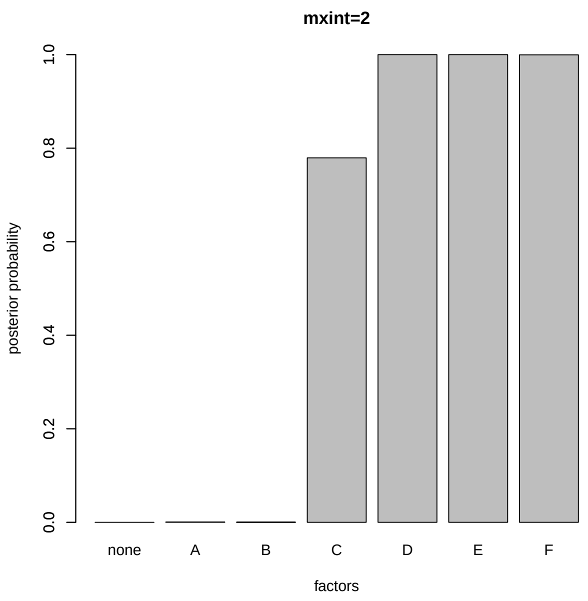

Figure 3: Posterior probabilities of factors using the full design of Table 2.

```
M2  0.220 0.014  3      4,5,6
M3  0.000 0.011  5      1,3,4,5,6

 Shannon index:
[1] 0.129

 CV:
[1] 0.717
```

From the posterior probabilities of factors we can observe that factors C, D, E, F are active while A and B are inert since $P_A = P_B = 0$, while $P_C = 0.77$ and $P_D = P_E = P_F = 1$. These posterior probabilities can be exhibited as in Figure 3 using the code

```
R> plot(es2.OBsProb, main = "mxint = 2")
```

where `es2.OBsProb` is the output of the function `OBsProb()`. Observe that only two models have a non-null posterior probabilities; specifically model M1 has the highest posterior probabilities (0.779). It contains the active factors, in the output enumerated from 3 (C) to 6 (F).

Herein we assume that model M1 is the "oracle" model and that the posterior probabilities of the factors' importance is the one reported in Figure 3 since they are obtained from a full factorial design. As a matter of fact, such a full experiment is very rarely conducted in practice due to the high number of trials involved and the consequently required amount of time and resources. Thus the researchers usually perform a fractional factorial design, reducing the number of runs. This kind of design often produces ambiguous results and thus extra runs are required to identify active factors. The collection of such additional runs defines a follow-up experiment. The question then becomes: how to efficiently choose such extra runs.

run	*A*	*B*	*C*	*D*	*E*	*F*	`Ytransformed`
2	−1	−1	−1	−1	1	1	0.93720712
25	−1	1	1	−1	1	−1	0.80710250
37	1	−1	−1	1	1	−1	0.98619329
62	1	1	1	1	1	1	1.05042017
15	−1	−1	1	1	−1	−1	0.94876660
24	−1	1	−1	1	−1	1	1.04493208
44	1	−1	1	−1	−1	1	0.04697041
51	1	1	−1	−1	−1	−1	0.73637703

Table 7: Runs, design matrix X and response variable `Ytransformed` corresponding to the 2^{6-3}_{III} fractional factorial design with generators $D = ABC$, $E = BC$, $F = AC$.

In order to illustrate the **OBsMD** package, we consider as initial experiment the same used for comparing different methods for design follow-up in Edwards *et al.* (2014). This will allow us to compare the results obtained adopting the objective Bayesian framework with the other approaches analyzed (foldover, semifoldover, D-optimal designs, MD-criterion of Meyer *et al.* 1996).

5.1. First 2^{6-3}_{III} fractional factorial design

We begin by considering a 2^{6-3}_{III} fractional factorial design according to the generators $D = ABC$, $E = BC$, $F = AC$. Given the generators, the experimenter knows the aliasing structure of the plan and its resolution equal to III, meaning that the main factors are aliased with two-factor interactions. The runs (ID) of Table 2 corresponding to this fractional factorial design are $(2, 25, 37, 62, 15, 24, 44, 51)$. The design matrix X and the related response variable are shown in Table 7.

The code

```
R> X <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 2:7]
R> y <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 8]
```

loads the data as documented in Table 7. A preliminary analysis based on the objective Bayesian approach is conducted using function `OBsProb()` with the command

```
R> es7.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
```

We set `mFac = 6`, `mInt = 2` and `abeta` $=$ `bbeta` $= 1$, thus requiring to consider all the models with a number of factors starting from zero (the null model) to `mFac` $= 6$, including all the possible two-factor interactions (`mInt` $= 2$) and to adopt the uniform distribution (`abeta` $=$ `bbeta` $= 1$) as prior on the model space. Using the `print` method reported below we require to print only the top 10 (`nTop = 10`) highest non-null model posterior probabilities.

```
R> print(es7.OBsProb, nTop = 10)
```

```
 Design Matrix:
    A  B  C  D  E  F
```

```
2  -1 -1 -1 -1  1  1
25 -1  1  1 -1  1 -1
37  1 -1 -1  1  1 -1
62  1  1  1  1  1  1
15 -1 -1  1  1 -1 -1
24 -1  1 -1  1 -1  1
44  1 -1  1 -1 -1  1
51  1  1 -1 -1 -1 -1

 Response vector:
0.937 0.807 0.986 1.05 0.949 1.045 0.047 0.736

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
     8      6      0      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.429
1      A 0.209
2      B 0.163
3      C 0.160
4      D 0.276
5      E 0.227
6      F 0.143

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.429 0.109  0      none
M2  0.068 0.081  1      4
M3  0.029 0.107  1      5
M4  0.027 0.110  1      1
M5  0.025 0.113  1      3
M6  0.024 0.020  3      1,2,5
M7  0.024 0.020  3      3,4,5
M8  0.023 0.063  2      1,4
M9  0.023 0.063  2      1,5
M10 0.023 0.063  2      4,5

 Shannon index:
[1] 0.64

 CV:
[1] 0.236
```

```
R> plot(es7.OBsProb, main = "mxint = 2")
```

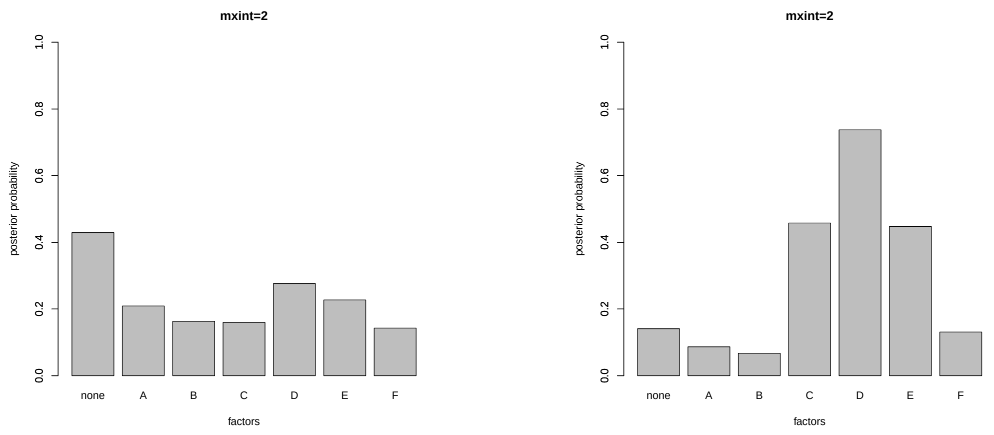

Figure 4: Posterior probabilities of factors for first 2^{6-3}_{III} design with generators $D = ABC$, $E = BC$, $F = AC$ (prior $Beta(1, 1)$) (on the left side); posterior probabilities of factors after the addition of the extra runs $(28 - 40 - 44 - 44)$ (prior $Beta(1, 1)$) (on the right side).

This fractional factorial design does not lead to a clear conclusion about which factors are truly active. In fact, looking at Figure 4 (on the left side), the factors with the largest effect are D and E, even if $\mathsf{P}(D) = 0.276$ and $\mathsf{P}(E) = 0.227$. Likewise, it indicates substantial ambiguity as none of the other factors clearly stands out as active. Actually there is no model whose posterior probability is really emerging from the others, apart from the null model. The value of the Shannon index is 0.64, thus confirming that the entropy is high, i.e., the posterior probabilities of the models are homogeneous and they do not allow predictability. As a consequence follow-up experimentation is warranted, namely it is necessary to augment the design with extra runs. Follow-up runs are selected using function `OMD()` in the package **OBsMD** in R. The function `OMD()` calls a Fortran subroutine. The output of the `OMD()` function is a list of class '`OMD`' with associated `print` and `summary` methods.

After having fixed the number of runs of the follow-up design, we compute the function `OMD()` for all the possible combinations with replacement of n_f runs from the full 2^6 design. For instance, if $n_f = 4$, the number of all the possible different additional runs is 766480. An automatic routine allows us to make a list of all the possible combination of n_f follow-up runs to allocate in the matrix `Mbest`:

```
R> Mbest <- as.matrix(combinations(64, 4, 1:64, repeats = TRUE))
```

To implement the procedure we first need to insert the matrix `Xcand` which contains the runs of the original 2^6 design. When the argument `blk` in the `OBsProb()` function is not set equal to 0, in the first columns `Xcand` will include the blocking factors.

```
R> Xcand <- MetalCutting[, 2:7]
```

As reported in Table 6, in the call of the function `OMD()` there are several arguments that have to be fixed coherently with those of `OBsProb()` such as: `OBsProb`, `nFac`, `nBlk` and `Xcand`. On the contrary the arguments `mIter`, `nStart` and `startDes` are to be set according to the choice of whether or not to use the exchange algorithm (see Section 4).

Furthermore the researcher has to fix the number of additional runs n_f to use as input (argument `nFoll`) of `OMD()` and the number `nMod` of competing models according to which the discrimination criterion *OMD* has to be computed.

In this example `nFoll = 4` and `nMod = 42`, since 42 are the number of models whose posterior probabilities are non-null according to the output of `OBsProb()`[4]. Obviously we could choose to discriminate only among the models in the list with a posterior probability greater than a certain value. For instance we could consider relevant to discriminate only among the models with posterior probability greater or equal to 0.025 (`nMod = 5`).

We fix `mIter = 0`, `nStart = nrow(Mbest)`, `startDes = Mbest` and we compute the function `OMD()` for all the possible 766480 follow-up runs (exhaustive search). The design with the largest *OMD* will be preferred.

```
R> es7.pp_omd <- OMD(OBsProb = es7.OBsProb, nFac = 6, nBlk = 0, nMod = 42,
+    nFoll = 4, Xcand = Xcand, mIter = 0, nStart = nrow(Mbest),
+    startDes = Mbest, top = 8)
```

The `summary` method gives an overview of the result obtained with `OMD()`:

```
R> summary(es7.pp_omd)


 Base:
 nRuns   nFac   nBlk maxInt   nMod
     8      6      0      2     42

 OMD:
  nCand   nRuns maxIter  nStart
     64       4       0  766480

Top 8 runs:
    OMD r1 r2 r3 r4
1 2.567 28 40 44 44
2 2.563 28 40 43 44
3 2.551 28 43 44 48
4 2.543 28 44 44 48
5 2.540 12 40 43 44
6 2.526 12 40 44 44
7 2.519 12 43 44 48
8 2.498 40 43 44 44
```

The runs (28 40 44 44) associated with the highest *OMD* value are added to the experiment. As a consequence an updating of the vector y and the matrix X is needed. To take into account the different time of the experimentation, a block factor (`blk = 1`) can be inserted in the first column of X. Observe that the level of the block factor is equal to -1 for the initial experiment and $+1$ for the follow-up one. The function `OBsProb()` is called again to recompute the posterior probabilities of the models.

[4]You can see these results with `print(es7.OBsProb, nTop = 42)`

```
R> X <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 1:7]
R> y <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 8]
R> TOP_DES <- cbind(blk = rep(1, 4), MetalCutting[es7.pp_omd$TOPDES[1, ],
+    c(-1, -8)])
R> y_TOP_DES <- MetalCutting[es7.pp_omd$TOPDES[1, ], 8]
R> X <- rbind(X, TOP_DES)
R> y <- c(y, y_TOP_DES)
R> es7.aug.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 1,
+    mFac = 6, mInt = 2, nTop = 64)
R> print(es7.aug.OBsProb, nTop = 10)

 Design Matrix:
   blk  A  B  C  D  E  F
2   -1 -1 -1 -1 -1  1  1
25  -1 -1  1  1 -1  1 -1
37  -1  1 -1 -1  1  1 -1
62  -1  1  1  1  1  1  1
15  -1 -1 -1  1  1 -1 -1
24  -1 -1  1 -1  1 -1  1
44  -1  1 -1  1 -1 -1  1
51  -1  1  1 -1 -1 -1 -1
28   1 -1  1  1 -1 -1  1
40   1  1 -1 -1  1 -1  1
44   1  1 -1  1 -1 -1  1
44   1  1 -1  1 -1 -1  1

 Response vector:
0.937 0.807 0.986 1.05 0.949 1.045 0.047 0.736 0.039 1.222 0.047 0.047

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
    12      6      1      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.141
1      A 0.087
2      B 0.067
3      C 0.458
4      D 0.737
5      E 0.448
6      F 0.131

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.271 0.007  3      3,4,5
M2  0.153 0.093  1      4
```

```
M3  0.141 0.181  0      none
M4  0.080 0.056  2      4,5
M5  0.049 0.066  2      3,4
M6  0.035 0.132  1      3
M7  0.029 0.034  3      3,4,6
M8  0.027 0.081  2      4,6
M9  0.026 0.036  3      4,5,6
M10 0.025 0.051  3      1,4,5

 Shannon index:
[1] 0.619

 CV:
[1] 0.766
```

The posterior probabilities of factors are exhibited in Figure 4 on the right side. We can observe how the augmentation of the design allows the identification of D as an active factor ($\mathsf{P}(D) = 0.737$), C and E as potentially active factors ($\mathsf{P}(C)$ and $\mathsf{P}(E)$ are around 0.45), while there is little evidence for factor F. A new augmented design might be performed. However, the result obtained applying the approach proposed in Consonni and Deldossi (2016) is comparable with the other methods with $n_f = 4$ in Edwards *et al.* (2014, see Figure 2 herein).

We can also observe that variability among the factors increases in the augmented design (CV = coefficient of variation is now 0.766 against 0.236), while the heterogeneity among the model posterior probabilities decreases (Shannon index is 0.619 against 0.64).

To adopt the exchange algorithm, some arguments of the function `OMD()` (in this example `es7.pp_omd`) have to be changed. In particular:

- `mIter` has to be set greater then 0 since it represents the maximum number of iterations of the exchange algorithm;
- `nStart` represents the number of random starts of the exchange algorithm (and not the number of rows of the matrix `Mbest`);
- `startDes` has to be set equal to `NULL` (and not to `Mbest`).

For instance in the sequel we fix `mIter` = 20 and `nStart` = 25. Using the code:

```
R> X <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 2:7]
R> y <- MetalCutting[c(2, 25, 37, 62, 15, 24, 44, 51), 8]
R> es7.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
R> Mbest <- as.matrix(combinations(64, 4, 1:64, repeats = TRUE))
R> Xcand <- MetalCutting[, 2:7]
R> es7.pp_omd_ex <- OMD(OBsProb = es7.OBsProb, nFac = 6, nBlk = 0,
+    nMod = 42, nFoll = 4, Xcand = Xcand, mIter = 20, nStart = 25,
+    startDes = NULL, top = 8)
R> summary(es7.pp_omd_ex)
```

run	A	B	C	D	E	F	`Ytransformed`
62	1	1	1	1	1	1	1.05042017
15	−1	−1	1	1	−1	−1	0.94876660
6	−1	−1	−1	1	1	1	0.88652482
17	−1	1	−1	−1	1	−1	0.96153846
41	1	−1	1	−1	1	−1	1.01214575
28	−1	1	1	−1	−1	1	0.03912363
36	1	−1	−1	−1	−1	1	0.06925208
55	1	1	−1	1	−1	−1	1.17233294

Table 8: Runs, design matrix X and response variable `Ytransformed` corresponding to the 2^{6-3}_{III} fractional factorial design with generators $D = AB$, $E = AC$, $F = BC$.

we obtain the following results:

```
 Base:
 nRuns   nFac   nBlk maxInt   nMod
     8      6      0      2     42

 OMD:
  nCand   nRuns maxIter  nStart
     64       4      20      25

   Top 8 runs:
     OMD r1 r2 r3 r4
1 2.567 28 40 44 44
2 2.563 28 40 43 44
3 2.551 28 43 44 48
4 2.543 28 44 44 48
5 2.540 12 40 43 44
6 2.519 12 43 44 48
7 2.498 40 43 44 44
8 2.487 40 43 44 60
```

Observe that the exchange algorithm identifies the same top 8 runs as the exhaustive search strategy.

5.2. Second 2^{6-3}_{III} fractional factorial design

As a second example now consider the plan 2^{6-3}_{III} with generators $D = AB$, $E = AC$, $F = BC$. The following code loads the data as displayed in Table 8.

```
R> X <- MetalCutting[c(62, 15, 6, 17, 41, 28, 36, 55), 2:7]
R> y <- MetalCutting[c(62, 15, 6, 17, 41, 28, 36, 55), 8]
```

A preliminary analysis is conducted using the following code:

```
R> es8.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
R> print(es8.OBsProb, nTop = 10)

[...]

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
     8      6      0      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.073
1      A 0.056
2      B 0.055
3      C 0.046
4      D 0.646
5      E 0.633
6      F 0.638

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.206 0.010  2      5,6
M2  0.206 0.010  2      4,6
M3  0.206 0.010  2      4,5
M4  0.155 0.010  3      4,5,6
M5  0.073 0.201  0      none
M6  0.014 0.000  3      1,4,6
M7  0.014 0.000  3      2,4,5
M8  0.012 0.147  1      6
M9  0.011 0.001  3      1,2,3
M10 0.011 0.001  3      3,5,6

 Shannon index:
[1] 0.533

 CV:
[1] 0.848

R> plot(es8.OBsProb, main = "mxint = 2")
```

Figure 5 on the left side indicates that factors D, E, F are active. Then, with the exception of factor C the posterior probabilities obtained with `OBsProb()` seem to highlight the true active factors (see Figure 3 representing the oracle of the effects importance). Note that this resolution III screening design shows a really different result compared to the previous one. The reason is that we are considering a highly fractional factorial design; as a consequence the choice of generators may influence the analysis of the screening design. Because $D = EF$,

$E = DF$, and $F = DE$, some caution is needed before considering factors others than D, E, and F as inert and dropping them off. In order to understand the potential effective relevance of the other factors a follow-up experimental design might be run. Fixed equal to 4 the dimension n_f of the follow-up design (`nFoll = 4`), the function `OMD()` is computed for all the possible designs of `nFoll` runs chosen from the 2^6 candidates of the full design (`Xcand`).

```
R> Xcand <- MetalCutting[, 2:7]
R> Mbest <- as.matrix(combinations(64, 4, 1:64, repeats = TRUE))
R> es8.pp_omd <- OMD(OBsProb = es8.OBsProb, nFac = 6, nBlk = 0, nMod = 8,
+    nFoll = 4, Xcand = Xcand, mIter = 0, nStart = nrow(Mbest),
+    startDes = Mbest, top = 8)
```

In this case we decide to fix `nMod` = 8. This means that the function `OMD()` is computed with the aim of discriminating among the eight models with the highest posterior probabilities after the initial experiment.

```
R> summary(es8.pp_omd)

 Base:
 nRuns   nFac   nBlk maxInt   nMod
     8      6      0      2      8

 OMD:
  nCand   nRuns maxIter  nStart
     64       4       0  766480

   Top 8 runs:
     OMD r1 r2 r3 r4
1 88.748 10 51 59 64
2 88.748 10 51 56 59
3 88.748  2 51 59 64
4 88.748 10 51 51 64
5 88.748 10 56 59 59
6 88.748 10 59 59 64
7 88.748 10 51 51 56
8 88.748  2 59 59 64
```

The first design with largest OMD appears to be $(10 - 51 - 59 - 64)$ with $OMD = 88.748$. After the addition of the extra runs associated with the highest $OMD()$ value, function `OBsMD` is called again.

```
R> X <- MetalCutting[c(62, 15, 6, 17, 41, 28, 36, 55), 1:7]
R> y <- MetalCutting[c(62, 15, 6, 17, 41, 28, 36, 55), 8]
R> TOP_DES <- cbind(blk = rep(1, 4), MetalCutting[es8.pp_omd$TOPDES[1, ],
+    c(-1, -8)])
R> y_TOP_DES <- MetalCutting[es8.pp_omd$TOPDES[1, ], 8]
R> X <- rbind(X, TOP_DES)
```

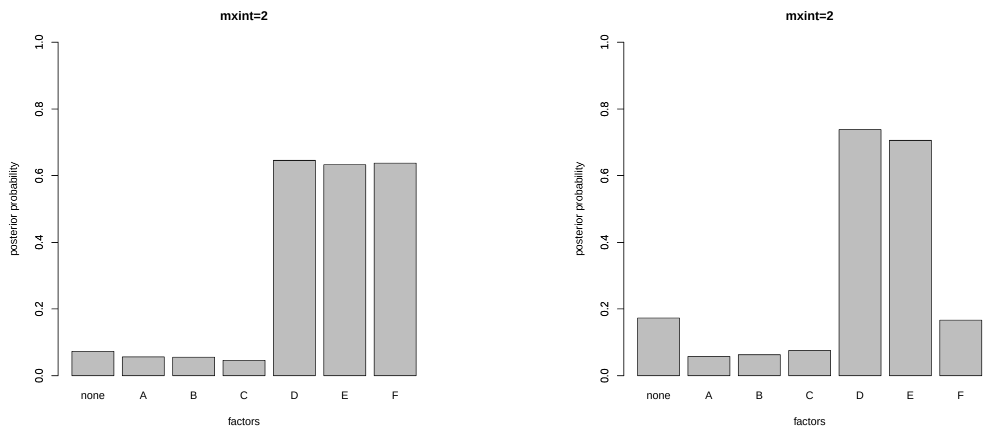

Figure 5: Posterior probabilities of factors for the second 2^{6-3}_{III} design with generators $D = AB$, $E = AC$, $F = BC$ (prior $Beta(1, 1)$) (on the left side); posterior probabilities of factors after the addition of the extra runs $(10 - 51 - 59 - 64)$ (prior $Beta(1, 1)$) (on the right side).

```
R> y <- c(y, y_TOP_DES)
R> es8.aug.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 1, blk = 1,
+    mFac = 6, mInt = 2, nTop = 64)
R> print(es8.aug.OBsProb, nTop = 10)
```

```
[...]

 Calculations:
  nRun   nFac   nBlk   mFac   mInt totMod
    12      6      1      6      2     64

 Factor probabilities:
  Factor  Prob
    none 0.173
1      A 0.058
2      B 0.063
3      C 0.076
4      D 0.738
5      E 0.706
6      F 0.167

 Model probabilities:
    Prob  Sigma2 NumFac Factors
M1  0.453 0.032  2      4,5
M2  0.173 0.176  0      none
M3  0.132 0.023  3      4,5,6
M4  0.040 0.131  1      4
M5  0.037 0.032  3      3,4,5
M6  0.023 0.148  1      5
M7  0.023 0.044  3      1,4,5
```

```
M8  0.020 0.049  3      2,4,5
M9  0.009 0.187  1      6
M10 0.007 0.091  3      1,2,3

 Shannon index:
[1] 0.476

 CV:
[1] 0.996
```

```
R> plot(es8.aug.OBsProb, main = "mxint = 2")
```

The update posterior probability of factors is displayed in Figure 5 on the right side. The result is comparable with that reported in Edwards *et al.* (2014).

5.3. 2^{6-2}_{IV} fractional factorial design one-run-at-a-time

In this section we consider the same fractional experimental design 2^{6-2}_{IV} analyzed in Section 2.2 (`abeta` $= 1$, `bbeta` $= 7$ and `mInt = 2`) but assuming that the follow-up experimentation is performed in one-run-at-a-time fashion.

After having computed the posterior probability of factors (see Figure 1) using the same code used in Section 2.2

```
R> X <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 2:7]
R> y <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 8]
R> es3.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 7, blk = 0,
+    mFac = 6, mInt = 2, nTop = 64)
```

function `OMD()` is run for $n_f = 1$, after having coherently set the matrix `Mbest`.

```
R> Xcand <- MetalCutting[, 2:7]
R> Mbest <- as.matrix(combinations(64, 1, 1:64, repeats = TRUE))
R> es3_1.pp_omd <- OMD(OBsProb = es3.OBsProb, nFac = 6, nBlk = 0,
+    nMod = 57, nFoll = 1, Xcand = Xcand, mIter = 0, nStart = nrow(Mbest),
+    startDes = Mbest, top = 10)
```

Observe that only `Mbest` and the argument `nFoll` $(= 1)$ assume a different value with respect to the function `OMD()` used in Section 2.2 to obtain `es3.pp_omd`.

After having added the run corresponding to the highest OMD value ($run = 36$) to the initial design, `OBsProb()` is applied and the posterior probabilities of the factors updated:

```
R> X <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 1:7]
R> y <- MetalCutting[c(62, 28, 51, 16, 64, 21, 26, 42, 44, 23, 39, 1, 14,
+    49, 37, 3), 8]
```

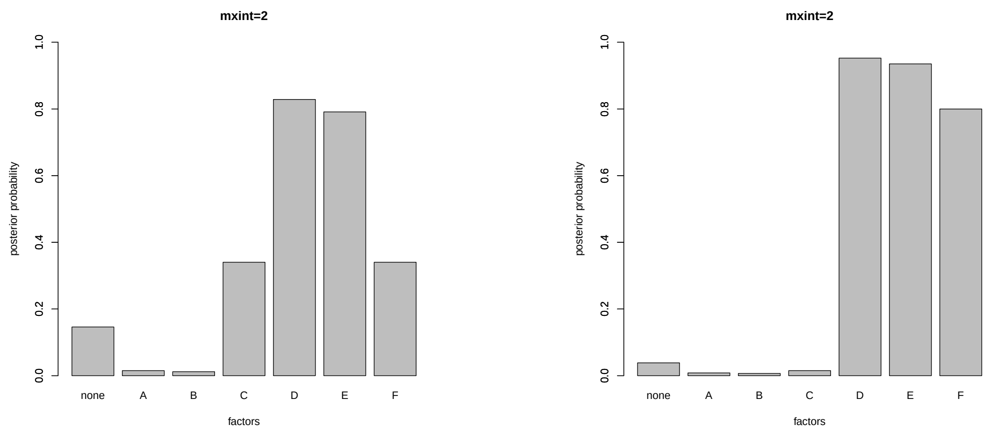

Figure 6: Posterior probabilities of factors for screening 2^{6-2}_{IV} design with generators $E = ABC$, $F = ABD$, (prior $Beta(1, 7)$) after the addition of the extra run 36 (on the left side) and $(36 - 55)$ (on the right side).

```
R> TOP_DES <- c(blk = c(1), MetalCutting[es3_1.pp_omd$TOPDES[1, ],
+    c(-1, -8)])
R> y_TOP_DES <- MetalCutting[es3_1.pp_omd$TOPDES[1, ], 8]
R> X <- rbind(X, TOP_DES)
R> y <- c(y, y_TOP_DES)
R> es3_1_1.aug.OBsProb <- OBsProb(X = X, y = y, abeta = 1, bbeta = 7,
+    blk = 1, mFac = 6, mInt = 2, nTop = 64)
R> print(es3_1_1.aug.OBsProb, nTop = 64)
```

After fixing `nMod = 10`, the second run ($run = 55$) is computed running the code:

```
R> es3_2_1.aug.pp_omd <- OMD(OBsProb = es3_1_1.aug.OBsProb, nFac = 6,
+    nBlk = 1, nMod = 10, nFoll = 1, Xcand = Xcand, mIter = 0,
+    nStart = nrow(Mbest), startDes = Mbest, top = 10)
R> summary(es3_2_1.aug.pp_omd)
```

With this second additional run we can conclude that D, E and F are active factors (see Figure 6 on the right side[5]). Observe that after just two runs, we obtain the same results reported in Figure 2 on the left side with $n_f = 4$.

Clearly the choice between running the follow-up design as a block or sequentially will depend on cost, feasibility, resources, and timeliness.

6. Conclusions

In this paper, the background and a tutorial are presented for the **OBsMD** package for the R environment. **OBsMD** implements the objective Bayesian methodology proposed in Consonni and Deldossi (2016) for identifying optimal follow-up runs to resolve model ambiguity. Unlike

[5]The code is available in the supplementary material.

other implementations, packages, or strategies, **OBsMD** should appeal to practitioners because of its flexibility and the use of the objective Bayesian approach which does not require specification of the prior parameters.

When applied to real data, it produces follow-up runs which discriminate among competing models better than the current methodology. Because the tutorial uses the dataset `MetalCutting` described in Mønness *et al.* (2007), the readers can compare the results obtained with our approach (and package) with those proposed in Edwards *et al.* (2014). Through the applications of this dataset, we have shown the performance and the usefulness of the **OBsMD** package. We believe that the availability of such a method will be appreciated by other R users as well.

Three examples have been discussed, along with some theoretical and practical issues. Flexibility is achieved by providing the user with many options. For example with regard to the prior of the model space, we adopted the values $(a = 1, b = 1)$, corresponding to the uniform distribution. The alternative choice $(a = 1, b = k + 1)$ has been advocated to achieve a stronger sparse model effect. Moreover the main difference with such a prior is that the choice $(a = 1, b = k + 1)$ gives more weight to more parsimonious models, relative to $(a = 1, b = 1)$, however, optimal follow-up runs are broadly similar in the two cases. The opportunity to add one-run-at-a-time represents a further key point, since, as in Section 5.3, we can discover which factors are active factors simply by augmenting the design with 1 or 2 runs.

The model discrimination criterion used in this package is based on the Kullback-Leibler divergence. Possible future extensions of this package may regard the use of alternative distances, like the Hellinger one (see Bingham and Chipman 2007) or the relaxation of the effect forcing assumption, which may sometimes represent a limitation.

We believe the **OBsMD** package may represent a useful tool since it does not require prior specifications, being fully automatic and not requiring prior specification unlike the Meyer *et al.* (1996) approach, implemented in the R package **BsMD**, whose sensitivity to the tuning parameters (π and γ) may represent a problematic issue.

References

Abramowitz M, Stegun IA (1965). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, volume 55 of *National Bureau of Standards Applied Mathematics Series*. U.S Government Printing Office, Washington, D.C.

Barrios E (2020). ***BsMD**: Bayes Screening and Model Discrimination.* R package version 2020.4.30, URL `https://CRAN.R-project.org/package=BsMD`.

Bayarri MJ, Berger JO, Forte A, García-Donato G (2012). "Criteria for Bayesian Model Choice with Application to Variable Selection." *The Annals of Statistics*, **40**(3), 1550–1577. `doi:10.1214/12-aos1013`.

Berger JO, Pericchi LR (2001). "Objective Bayesian Methods for Model Selection: Introduction and Comparison." In *Model Selection*, volume 38 of *IMS Lecture Notes*, pp. 135–207. Institute of Mathematical Statistics.

Bingham DR, Chipman HA (2007). "Incorporating Prior Information in Optimal Design for Model Selection." *Techonometrics*, **49**(2), 155–163. `doi:10.1198/004017007000000038`.

Box GEP, Hunter WG, Hunter JS (1978). *Statistics for Experimenters. An Introduction to Design, Data Analysis, and Model Building*. John Wiley & Sons.

Chipman H, Hamada MS (1996). "Discussion: Factor-Based or Effect-Based Modeling? Implications for Design." *Technometrics*, **38**(4), 317–320. doi:10.1080/00401706.1996.10484540.

Consonni G, Deldossi L (2016). "Objective Bayesian Model Discrimination in Follow-Up Design." *Test*, **25**(3), 397–412. doi:10.1007/s11749-015-0461-3.

Edwards DJ, Weese ML, Palmer GA (2014). "Comparing Methods for Design Follow-Up: Revisiting a Metal-Cutting Case Study." *Applied Stochastic Models in Business and Industry*, **30**(4), 464–478. doi:10.1002/asmb.1988.

Goos P, Jones BA (2011). *Optimal Design of Experiments: A Case Study Approach*. John Wiley & Sons. doi:10.1002/9781119974017.

Grömping U (2014). "R Package **FrF2** for Creating and Analyzing Fractional Factorial 2-Level Designs." *Journal of Statistical Software*, **56**(1), 1–56. doi:10.18637/jss.v056.i01.

Grömping U (2020). ***FrF2**: Fractional Factorial Designs with 2-Level Factors*. R package version 2.2-2, URL https://CRAN.R-project.org/package=FrF2.

Mee RW, Peralta M (2000). "Semifolding 2^{k-p} Designs." *Techometrics*, **42**(2), 122–134. doi:10.2307/1271444.

Meyer D (1996). ***mdopt**: Fortran Programs to Generate MD-Optimal Screening and Follow-up Designs, and Analysis of Data*. Statlib, Fortran package version 90, URL http://lib.stat.cmu.edu/.

Meyer RD, Steinberg DM, Box GEP (1996). "Follow-Up Designs to Resolve Confounding in Fractional Factorials." *Technometrics*, **38**(4), 303–313. doi:10.2307/1271297.

Mønness E, Linsley MJ, Garzon IE (2007). "Comparing Different Fractions of a Factorial Design: A Metal-Cutting Case Study." *Applied Stochastic Models in Business and Industry*, **23**(2), 117–128. doi:10.1002/asmb.641.

Montgomery DC (2006). *Design and Analysis of Experiments*. John Wiley & Sons.

Montgomery DC, Runger GC (1996). "Foldovers of 2^{k-p} Resolution IV Experimental Designs." *Journal of Quality Technology*, **28**(4), 446–450. doi:10.1080/00224065.1996.11979702.

Nai Ruscone M, Deldossi L (2020). ***OBsMD**: Objective Bayesian Model Discrimination in Follow-Up Designs*. R package version 6.1, URL https://CRAN.R-project.org/package=OBsMD.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Scott JG, Berger JO (2010). "Bayes and Empirical-Bayes Multiplicity Adjustment in the Variable-Selection Problem." *The Annals of Statistics*, **38**(5), 2587–2619. doi:10.1214/10-aos792.

Wolters MA, Bingham DR (2011). "Simulated Annealing Model Search for Subset Selection in Screening Experiments." *Technometrics*, **53**(3), 225–237. doi:10.1198/tech.2011.08157.

Affiliation:

Laura Deldossi
Departimento di Scienze Statistiche
Facultá di Economia
Universitá Cattolica del Sacro Cuore di Milano
20123 Milan, Italy
E-mail: laura.deldossi@unicatt.it

Marta Nai Ruscone
Dipartimento di Matematica – DIMA
Universitá degli Studi di Genova
Via Dodecaneso, 35
16146 Genova (GE), Italy
E-mail: marta.nairuscone@unige.it

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
June 2020, Volume 94, Issue 2 *Submitted:* 2017-04-20
doi:10.18637/jss.v094.i02 *Accepted:* 2019-05-12