Mixed Models for the Analysis of Optimization Algorithms

Marco Chiarandini
Department of Mathematics and Computer Science,
University of Southern Denmark, Odense, Denmark
marco@imada.sdu.dk

Yuri Goegebeur
Department of Mathematics and Computer Science,
University of Southern Denmark, Odense, Denmark,
and Research Group Quantitative Psychology and Individual Differences,
K.U.Leuven, Belgium
yuri.goegebeur@stat.sdu.dk

May 2009

Abstract

We review linear statistical models for the analysis of computational experiments on optimization algorithms. The models offer the mathematical framework to separate the effects of algorithmic components and instance features included in the analysis. We regard test instances as drawn from a population and we focus our interest not on those single instances but on the whole population. Hence, instances are treated as a *random factor*. Overall these experimental designs leads to *mixed effects linear models*. We present both the theory to justify these models and a computational example in which we analyze and comment several possible experimental designs. The example is a component-wise analysis of local search algorithms for the 2-edge-connectivity augmentation problem. We use standard statistical software to perform the analysis and report the R commands. Data sets and the analysis in SAS are available in an online compendium.

1 Introduction

Linear statistical models are well developed mathematical tools for the separation of effects in the observed results of an experiment. Among them, there is the classical analysis of variance (ANOVA). Such models have proved useful in many scientific disciplines and also in the field of optimization. In operation research, application examples to test mathematical programming software go back to the late 1970s, see,

e.g., Zanakis (1977); Lin and Rardin (1979); Coffin and Saltzman (2000); while in computer science and in testing heuristic and evolutionary computation methods their use can be traced back to the late 1990s. Prominent articles in this case are Barr et al (1995); McGeoch (1996); Rardin and Uzsoy (2001); Czarn et al (2004). However, only a very small number of articles, relatively to those published in these fields, report about using these statistical methods. This fact might be explained by two factors: the need of a background in statistics and experimental design techniques in order to correctly apply and fully understand the results provided; and the presence of underlying assumptions that make the researcher in computer science or operations research sceptical about the real applicability of these methods in the field of optimization. The aim of this chapter is to introduce the reader to the use of linear statistical models in the cases where they can be applied. We aim at showing the basic theory behind the methods, the practical application by means of publically available software and the possible outcomes. We go perhaps at a deeper level of detail with respect to previous publications in this field, hoping to facilitate future applications. Yet, we do not aim at removing completely the two barriers above: understanding of statistics and a careful investigation of applicability to each specific case are necessary preconditions. Knoweldge of the material in the appendix of this book might be required to follow this chapter.

We emphasize that our intention is presenting these tools as complementary and not substitutive to the current practice of reporting numerical results on benchmark instances with appropriate tables. This practice is indeed helpful to guarantee comparability and verifiability of results. The methods in this chapter are however desirable for scientific experimental analysis, where the interest is in explaining the causes of success of a certain optimization approach rather than in mere comparative studies (see Hooker, 1996 for a discussion on these guidelines).

To illustrate the application of the statistical tools we use a case example in which we study heuristic algorithms for a graph problem: finding the cheapest augmentation of arcs that make a network 2-edge-connected (Bang-Jensen et al, 2009). The heuristics are local search algorithms (Michiels et al, 2007) obtained by the combination of some specific components, which may be *qualitative*, like for the presence or not of an algorithmic step or *numerical*, like for parameters that assume real values. Our interest is in understanding the contribution of these components.

In statistical terms, these components are called *factors*. The interest is in the effects of the specific *levels* chosen for these factors. Hence, we say that the levels and consequently the factors are *fixed*. Moreover, when for two factors, every factor level of a factor appears with every factor level of another factor we say that the two factors are *crossed*. We restrict ourselves to analyze the effect of these factors on a univariate measure of performance, namely the quality of the solutions returned by the algorithm at termination. Multivariate analysis are however also possible by extensions of these methods, we refer to Johnson and Wichern (2007) for an overview of these.

Typically, the researcher takes a few instances for the problem at hand and collects the results of some runs of the algorithms on these instances. The instances are treated as *blocks* and all algorithms are run on each single instance. Results are therefore *grouped* per instance. The instances are chosen at random from a large set of possible instances of the problem and the interest of the researcher is not just on the performance of the algorithms on those specific instances chosen, but rather on the

generalization of the results to the entire population of instances. In statistical terms, instances are also levels of a factor. However, this factor is of a different nature than the fixed algorithmic factors described above. Indeed, the levels are chosen at random and the interest is not in these specific levels but in the population from which they are sampled. We say that the levels and the factor are *random*.

Further, it might be possible to stratify the instances according to some characteristics or features easily retrievable. The researcher might then be interested in studying the influence of these characteristics on the performance of the algorithms. Instance characteristics can be regarded as fixed factors, because we can control them and the interest is on the specific levels. However, in such a study another issue arises: the instances at different levels of the instance factors are different, that is, they are sampled from different populations. In other terms, the random factor does not cross like all other factors, but it is instead *nested* within some of them.

In statistics, the effects described are modeled as linear combinations and mathematical theory has been developed to make inference about the populations on the basis of the results observed in the samples. The mixed nature of the factors leads to so-called *nested linear mixed models*, see for instance Molenberghs and Verbeke (1997); Montgomery (2005); Pinheiro and Bates (2000). These designs, which are typical of the context of optimization, are non-trivial designs and go beyond the classical multi-factorial ANOVA, where all factors are instead treated as fixed. As we will see, the mathematical formula involved and the inference derived are different in the case of mixed-effects models and this may lead to a different inference. In our practical application we will give an example where this difference clearly arise. To the best of our knoweldge, only Lin and Rardin (1979) make a clear reference to the nesting issue while in all other articles that we reviewed the mixed nature of the factors is not emphasized or ignored.

The whole chapter is based on the assumption that additive linear models and normal distributions are appropriate to describe the experimental data. This is clearly a strong assumption that is often not met in experiments involving optimization algorithms. In fact, the example that we develop in the second part of the chapter was selected out of three, where the other two did not pass a diagnostic analysis on the assumptions. The arguments in defense of these tools also when assumptions are not met are the proven robustness of F-ratio tests in the analysis of variance method (Montgomery, 2005) and that small adjustments of the data, like increase in the number of observations, removal of outliers and opportune data transformations (e.g., log transformation) may contribute to meet the assumptions. Our point of view is that even when assumptions are not met, these tools can be a very useful exploratory device to look into the data. Extensions and generalizations that remove the need for these assumptions exist but for reasons of space we will not review them here.

The approach that we take to statistical inference is the classical one from statistics in which experiments are fully designed *a priori*. Even though differences among the entities studied always exist, we assume as correct the conservative hypothesis of no differences, and distinguish between *statistical differences* and *practically meaningful differences*. In this sense, we define a minimal effect size that is relevant in practice and derive the amount of data necessary to achieve a statistical power of 0.80 at a given level of significance of 0.05. We acknowledge that there are other ways to address the issue of sample size determination in experimental design.

The chapter is organized as follows. In Section 2, we formalize the problem of inference and the experimental designs, and we provide analytical support for the use of mixed models. We then review the theoretical background of the analysis. A reader primarily interested in the practical application of these methods may skip this part or consider it only when referenced back in Section 4. In Section 3, we introduce the application example on the 2-edge-connectivity problem. In Section 4, we develop the example producing an extended numerical analysis that reflects the mathematical background and the organization of Section 2. With the aim of facilitating reproduction, we report explicitly, in this section, the commands for the analysis in R, the free software environment for statistical computing (R Development Core Team, 2008). We conclude in Section 5 with a summary and pointers to further developments that could be helpful in similar studies.

2 Experimental design and statistical analysis

In the most basic design, the researcher wishes to assess the performance of an *optimization algorithm* on a single problem instance π. Since optimization algorithms are, in many cases, randomized, their performance Y on one instance is a random variable that might be described by a probability/density function $p(y|\pi)$.

Most commonly, we aim at drawing conclusions about a certain *class* or *population* of instances Π. In this case, the performance Y of the algorithm on the class Π is described by the probability function

$$p(y) = \sum_{\pi \in \Pi} p(y|\pi)p(\pi), \tag{1}$$

with $p(\pi)$ being the probability of sampling instance π. In other terms, we are interested in the distribution of Y marginalized over the population of instances. This modeling approach is described also by McGeoch (1996), Wolpert and Macready (1997) and Birattari (2004).

In experiments, we sample the population of instances and on each sampled instance we collect sample data on the performance of the algorithm. If on an instance π we run the algorithm r times then we have r replicates of the performance measure Y, denoted $Y_1, \ldots, Y_r$, which are, conditionally on the sampled instance and given the random nature of the algorithm, independent and identically distributed (i.i.d.), i.e.,

$$p(y_1, \ldots, y_r|\pi) = \prod_{j=1}^{r} p(y_j|\pi). \tag{2}$$

Marginally (over all the instances) the observed performance measures may show dependence, as is seen from

$$p(y_1, \ldots, y_r) = \sum_{\pi \in \Pi} p(y_1, \ldots, y_r|\pi)p(\pi). \tag{3}$$

The model (3) can be easily extended to the case where several algorithms are applied to the same instance by incorporating fixed effects in the conditional structure of (2). Next, we illustrate how this leads naturally to a mixed model.

We organize our presentation in different cases according to the number and type of factors involved. For the sake of conciseness, we identify the cases with the following notation:

$$\left\langle \begin{array}{c}\text{algorithm}\\ \text{factors}\end{array} , \begin{array}{c}\text{number of}\\ \text{instances}\end{array} \left(\begin{array}{c}\text{instance}\\ \text{factors}\end{array} \right), \begin{array}{c}\text{number of}\\ \text{runs}\end{array} \right\rangle$$

For example, $\langle N, q(M), r\rangle$ means that we are in the presence of an experimental design with N algorithmic factors, q instances sampled from each combination of M instance factors and r runs of the algorithm per instance. We use small letter when referring to number of factors and capital letter when referring to number of levels. We indicate the absence of fixed factors by a dash -. The round parenthesis indicates nesting and its meaning is better explained in Section 2.3.

2.1 Case $\langle$-, q(-), $r\rangle$: Random effect design

We start with the simplest experiment where one algorithm is evaluated on q instances randomly sampled from a class Π. The experiment is performed as follows. In a first stage an instance is randomly drawn from a population of instances, whereafter the single algorithm is run r times on the instance. Given the stochastic nature of the algorithm, this produces, *conditionally on the sampled instance*, r replications of the performance measure that are i.i.d. We use Y_{ij} to denote the random performance measure obtained in the jth replication of the algorithm on the ith instance.

The instances included in the study are randomly drawn from some population of instances, and the interest is in inferring about this larger population of instances, not just on those included in the experiment. The above considerations lead us to propose the following random effects model

$$Y_{ij} = \mu + \tau_i + \varepsilon_{ij}, \tag{4}$$

where

μ is an overall mean,

τ_i is a random variable representing the effect of instance i,

ε_{ij} is a random error term for replication j on instance i.

As such, the stochastic behavior of the response variable originates from both the instance and the algorithm. Concerning the random elements in the right-hand-side of (4) we assume the following:

- $\tau_1, \ldots, \tau_q$ are i.i.d. $N(0, \sigma_\tau^2)$,
- ε_{ij}, $i = 1, \ldots, q$, $j = 1, \ldots, r$, are i.i.d. $N(0, \sigma^2)$,
- all τ_i and ε_{ij} are independent of each other.

Note that the postulated random effects model satisfies the structure of the conditional and marginal models given by (2) and (3). In particular, the conditional distribution of the performance measure given the instance is given by

$$Y_{ij}|\tau_i \sim N(\mu + \tau_i, \sigma^2), \quad j = 1, \ldots, r.$$

Furthermore, conditionally on the random effect τ_i, the random variables $Y_{i1}, \ldots, Y_{ir}$ are independent. Integrating out the random effects we obtain the unconditional model

$$Y_{ij} \sim N(\mu, \sigma^2 + \sigma_\tau^2), \qquad i = 1, \ldots, q,\ j = 1, \ldots, r.$$

The use of random instance effects yields dependency between the performance measurements obtained on a specific instance, while performances are independent if they pertain to different instances. Hence, the covariance structure of model (4) is

$$\mathrm{Cov}(Y_{ij}, Y_{i'j'}) = \begin{cases} \sigma^2 + \sigma_\tau^2, & \text{if } i = i' \text{ and } j = j', \\ \sigma_\tau^2, & \text{if } i = i' \text{ and } j \neq j', \\ 0, & \text{if } i \neq i', \end{cases} \tag{5}$$

which is the compound symmetric covariance structure. The parameters σ^2 and σ_τ^2 determine the variance of the individual Y_{ij} as well as the covariance between the Y_{ij}, and therefore are called the *variance components*.

Collecting the performance measurements $Y_{i1}, \ldots, Y_{ir}$ into the vector $\mathbf{Y}_i$, and denoting by $\mathbf{1}$ the r-dimensional vector of ones and by $\boldsymbol{\Sigma}$ the $(r \times r)$ covariance matrix of $\mathbf{Y}_i$, i.e.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma^2 + \sigma_\tau^2 & \sigma_\tau^2 & \cdots & \sigma_\tau^2 \\ \sigma_\tau^2 & \sigma^2 + \sigma_\tau^2 & \cdots & \sigma_\tau^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_\tau^2 & \sigma_\tau^2 & \cdots & \sigma^2 + \sigma_\tau^2 \end{bmatrix},$$

we can summarize the above as

$$\mathbf{Y}_i \sim N_r(\mu\mathbf{1}, \boldsymbol{\Sigma}), \qquad i = 1, \ldots, q,$$

independently.

Note that in ordinary ANOVA, the *instance* factor in model (4) would be considered a fixed factor, i.e., non-random, yielding

$$Y_{ij} \sim N(\mu + \tau_i, \sigma^2), \quad i = 1, \ldots, q,\ j = 1, \ldots, r,$$

with Y_{ij} being independent, i.e., unlike model (4), this model does not take into account dependencies arising from applying an algorithm repeatedly to the same instances.

Given that the instances are here considered as samples from some larger population of instances, we are not interested in performing a hypothesis test about the particular levels included in the study. Instead, the interest is in the whole population of instances and hence the hypothesis of interest is one involving the variance component σ_τ^2, in particular

$$H_0 : \sigma_\tau^2 = 0 \qquad \text{versus} \qquad H_1 : \sigma_\tau^2 > 0. \tag{6}$$

Clearly, if H_0 is true then the instance distribution reduces to a point mass at zero, implying that all possible instance parameters are fixed and equal to zero, which corresponds to no instance effect.

Intuitively, tests concerning σ_τ^2, as specified in (6), and tests involving a comparison of the algorithmic variance σ^2 and the instance variance σ_τ^2 should be based on a

comparison of the between instance variability, measured by $r\sum_{i=1}^{q}(\bar{Y}_{i.}-\bar{Y}_{..})^2$, and the within instance variability, measured by $\sum_{i=1}^{q}\sum_{j=1}^{r}(Y_{ij}-\bar{Y}_{i.})^2$ (the dot and the bar in $\bar{Y}_{i.}$ indicate averages over the index j and in $\bar{Y}_{..}$ average over both indices i and j). Statistical theory motivates the use of the ratio

$$F = \frac{\frac{r\sum_{i=1}^{q}(\bar{Y}_{i.}-\bar{Y}_{..})^2}{(q-1)(\sigma^2+r\sigma_\tau^2)}}{\frac{\sum_{i=1}^{q}\sum_{j=1}^{r}(Y_{ij}-\bar{Y}_{i.})^2}{q(r-1)\sigma^2}}, \tag{7}$$

as it can be shown that under the above model assumptions $F \sim F(q-1, q(r-1))$, where $F(\nu_1, \nu_2)$ is used to denote the F distribution with ν_1 and ν_2 degrees of freedom. We distinguish three specific uses of (7):

– Test for an instance effect: $H_0 : \sigma_\tau^2 = 0$ versus $H_1 : \sigma_\tau^2 > 0$.

Under $H_0 : \sigma_\tau^2 = 0$ we have

$$F_1 = \frac{MSI}{MSE},$$

where

$$\begin{aligned} MSI &= \frac{r\sum_{i=1}^{q}(\bar{Y}_{i.}-\bar{Y}_{..})^2}{q-1}, \\ MSE &= \frac{\sum_{i=1}^{q}\sum_{j=1}^{r}(Y_{ij}-\bar{Y}_{i.})^2}{q(r-1)}, \end{aligned}$$

and $F_1 \sim F(q-1, q(r-1))$, leading to the decision rule to reject H_0 at the significance level α if $f_1 > F(1-\alpha; q-1, q(r-1))$, where f_1 is the realization of F_1 from the observed data. An intuitive motivation for the form of statistic F_1 can be obtained from the expected mean squares. It can be shown that

$$\mathrm{E}[MSI] = \sigma^2 + r\sigma_\tau^2, \text{ and} \tag{8}$$

$$\mathrm{E}[MSE] = \sigma^2, \tag{9}$$

so under H_0 both MSI and MSE estimate σ^2 in an unbiased way, and F_1 can be expected to be close to one. On the other hand, large values of F_1 give evidence against H_0.

– Test involving a comparison of the instance and the algorithmic variance: $H_0 : \sigma_\tau^2/\sigma^2 = c$ versus $H_1 : \sigma_\tau^2/\sigma^2 \neq c$.

Under $H_0 : \sigma_\tau^2/\sigma^2 = c$ we have

$$F_2 = \frac{\frac{r\sum_{i=1}^{q}(\bar{Y}_{i.}-\bar{Y}_{..})^2}{(q-1)(1+rc)}}{\frac{\sum_{i=1}^{q}\sum_{j=1}^{r}(Y_{ij}-\bar{Y}_{i.})^2}{q(r-1)}} \sim F(q-1, q(r-1)),$$

leading to the decision rule to reject H_0 if $f_2 < F(\alpha/2; q-1, q(r-1))$ or $f_2 > F(1-\alpha/2; q-1, q(r-1))$.

– Power calculations. Power calculations can be useful at the design stage of the experiment when one has to decide on the number of instances q and the number of replicates r. The power of a statistical test is the probability that the test will reject the null hypothesis when in fact the alternative hypothesis is true. The power of the F test for testing $H_0 : \sigma_\tau^2 = 0$ vs $H_1 : \sigma_\tau^2 > 0$ can be computed from

$$POWER = \Pr\left\{ F > \frac{F(1-\alpha; q-1, q(r-1))}{1 + r\tilde{\sigma}_\tau^2/\sigma^2} \right\}, \tag{10}$$

where $\tilde{\sigma}_\tau^2$ is a value for σ_τ^2 from H_1. We refer to Section 4.1 for an illustration on the use of power calculations.

It might be also relevant to *estimate* the overall mean μ. Since we have $E[Y_{ij}] = \mu$ then an unbiased estimator of μ is:

$$\hat{\mu} = \bar{Y}_{..}$$

It can be shown that an unbiased estimator[1] of $\sigma^2[\bar{Y}_{..}]$ is $s^2[\bar{Y}_{..}] = MSI/qr$ and

$$\frac{\bar{Y}_{..} - \mu}{\hat{s}[\bar{Y}_{..}]} \sim t(q-1)$$

Hence, we obtain the *confidence limits* for μ by

$$\bar{y}_{..} \pm t(1-\alpha/2; q-1)s[\bar{Y}_{..}]. \tag{11}$$

Confidence intervals on the variance components σ_τ^2 and σ^2 can also be derived (see Kutner et al, 2005).

2.2 Case $\langle N, q(\text{-}), r\rangle$: Mixed effect design

We now consider the case where h *algorithms* are evaluated on q *instances* randomly sampled from a class Π. The experiment is performed as follows. In a first stage an instance is sampled from a population of instances. Next, each algorithm is run r times on the instance. Again, *conditionally on the instance* and for a given algorithm, we obtain r i.i.d. replications of the performance measure. We use Y_{ijk} to denote the random performance measure obtained in replication k of algorithm j on instance i. Note that we are here, for simplicity of exposition, dealing with a special case of design $\langle N, q(\text{-}), r\rangle$, namely the case where $N = 1$, which corresponds to having one single factor representing the different algorithms.

The algorithms included in the study are the ones in which we are particularly interested, and hence they can be considered as levels of a *fixed factor*. As before, the instances are drawn randomly from some population of instances and the interest is in inferring about this global population of instances, not just those included in the study. Hence, we assume that the performance measure can be decomposed according to the following *mixed effects ANOVA model*

$$Y_{ijk} = \mu + \alpha_j + \tau_i + \gamma_{ij} + \varepsilon_{ijk}, \tag{12}$$

where

[1] We adopt the convention of using the same symbol for esimators and estimates.

μ is an overall performance level common to all observations,

α_j is a fixed effect due to the algorithm j,

τ_i is a random effect associated with instance i,

γ_{ij} is a random interaction between instance i and algorithm j,

ε_{ijk} is a random error for replication k of algorithm j on instance i.

For identification purposes we impose the usual sum constraint on the factor level effects, i.e., $\sum_{j=1}^{h} \alpha_j = 0$. The assumptions imposed on the random elements are

- τ_i are i.i.d. $N(0, \sigma_\tau^2)$,
- γ_{ij} are i.i.d. $N(0, \sigma_\gamma^2)$,
- ε_{ijk} are i.i.d. $N(0, \sigma^2)$,
- the τ_i, γ_{ij} and ε_{ijk} are mutually independent random variables.

Also here the postulated model satisfies the structure of the conditional and marginal models given by (2) and (3). In particular, the conditional distribution of the performance measure given the instance and the instance-algorithm interaction is given by

$$Y_{ijk}|\tau_i, \gamma_{ij} \sim N(\mu + \alpha_j + \tau_i + \gamma_{ij}, \sigma^2), \quad i = 1, \ldots, q,\ j = 1, \ldots, h,\ k = 1, \ldots, r.$$

Furthermore, conditionally on the random effects τ_i and γ_{ij}, $i = 1, \ldots, q$, $j = 1, \ldots, h$, all responses are independent. Integrating out the random effects we obtain the marginal model for the response variables:

$$Y_{ijk} \sim N(\mu + \alpha_j, \sigma^2 + \sigma_\tau^2 + \sigma_\gamma^2), \qquad i = 1, \ldots, q,\ j = 1, \ldots, h,\ k = 1, \ldots, r.$$

The use of random instance effects and random instance-algorithm interactions yields dependency between the performance measurements obtained on a specific instance, while observations are independent if they pertain to different instances. The covariance between any two observations under model (12) is

$$\mathrm{Cov}(Y_{ijk}, Y_{i'j'k'}) = \begin{cases} \sigma^2 + \sigma_\tau^2 + \sigma_\gamma^2, & \text{if } i = i',\ j = j',\ k = k', \\ \sigma_\tau^2 + \sigma_\gamma^2, & \text{if } i = i',\ j = j',\ k \neq k', \\ \sigma_\tau^2, & \text{if } i = i',\ j \neq j', \\ 0, & \text{if } i \neq i'. \end{cases} \tag{13}$$

The mixed model (12) with its assumptions forms the natural basis for testing hypotheses about both fixed and random factors, and their interactions. Concerning the fixed factors, the interest is usually in testing whether there is a difference between the factor level means $\mu + \alpha_1, \ldots, \mu + \alpha_h$. Formally, one tests the hypothesis

$$\begin{aligned} H_0 &: \alpha_1 = \alpha_2 = \ldots = \alpha_h = 0, \\ H_1 &: \text{at least one } \alpha_j \text{ not equal to } 0. \end{aligned}$$

Effects	Mean Squares	df	Expected Mean Squares	Test Statistics
Fixed Factor	MSF	$h-1$	$\sigma^2 + r\sigma^2_\gamma + rp\frac{\sum_{j=1}^{h}\alpha_j^2}{h-1}$	$MSF/MSFR$
Random Factor	MSR	$p-1$	$\sigma^2 + r\sigma^2_\gamma + rh\sigma^2_\tau$	$MSR/MSFR$
Interaction	$MSFR$	$(h-1)(p-1)$	$\sigma^2 + r\sigma^2_\gamma$	$MSFR/MSE$
Error	MSE	$hq(r-1)$	σ^2	

TABLE 1: Expected mean squares and consequent appropriate test statistics for a mixed two-factor model. For a generalization to multifactorial cases see rules in Montgomery (2005); Kutner et al (2005)

Similarly to the random effects model, for the random effects, tests about the particular levels included in the study are meaningless. Instead we test hypotheses about the variance components σ^2_τ and σ^2_γ, reflecting that the ultimate interest is in the whole population of instances:

$$\begin{array}{lcl} H_0 & : & \sigma^2_\tau = 0, \\ H_1 & : & \sigma^2_\tau > 0, \end{array} \qquad \text{and} \qquad \begin{array}{lcl} H_0 & : & \sigma^2_\gamma = 0, \\ H_1 & : & \sigma^2_\gamma > 0, \end{array}$$

respectively. In balanced designs, the test statistics for these hypotheses are ratios of mean squares that are chosen such that the expected mean squares of the numerator differs from the expected mean squares of the denominator only by the variance components of the random factor in which we are interested. We report the resulting analysis of variance in Table 1. Formal procedures that automatize the derivation of these tables are described by Montgomery (2005, pag. 502), Kutner et al (2005) and Molenberghs and Verbeke (1997, 2005).

Estimators for the fixed effects of balanced mixed models have been also derived. Estimators of the fixed effects α_j are $\hat{\alpha}_j = \bar{Y}_{\cdot j \cdot} - \bar{Y}_{\cdots}$. Perhaps more interesting for our purposes in the analysis of optimization algorithms is the marginal mean $\mu_{\cdot j} = \mu + \alpha_j$ whose best estimator is $\hat{\mu}_{\cdot j} = \bar{Y}_{\cdots} + (\bar{Y}_{\cdot j \cdot} - \bar{Y}_{\cdots}) = \bar{Y}_{\cdot j \cdot}$. Unbiased estimators for the variances $\sigma^2[\hat{\alpha}_j]$ and $\sigma^2[\hat{\mu}_{\cdot j}]$ are

$$s^2[\hat{\alpha}_j] = \frac{1}{qr} MSFR \quad \text{and} \quad s^2[\hat{\mu}_{\cdot j}] = \frac{h-1}{hqr} MSFR + \frac{1}{hqr} MSR,$$

respectively. We can then compute exact confidence limits on pairwise comparisons of fixed effects, $D = \mu_{\cdot j} - \mu_{\cdot j'} = \alpha_j - \alpha_{j'}$, by the fact that

$$\frac{\hat{D} - D}{s[\hat{D}]} \sim t\big((h-1)(p-1)\big) \ ,$$

where $s^2[\hat{D}] = 2s^2[\hat{\alpha}_j]$.

Tukey's multiple comparison procedure can be used to guarantee a *family confidence coefficient* $1-\alpha$ when multiple comparisons are to be performed. In other terms, if we perform $\binom{h}{2}$ pairwise comparisons for the fixed factor, we want each of them to be correct $(1-\alpha)100$ percent of times. Tukey's procedure consists in substituting

the t distribution with the studentized range distribution. More precisely, we can make explicit the multiple comparisons confidence limits for all pairwise comparisons $D = \mu_{.j} - \mu_{.j'}$, or equivalently $D = \alpha_j - \alpha_{j'}$, with family confidence coefficient of at least $1 - \alpha$ as follows

$$\hat{D} \pm Ts[\hat{D}], \qquad T = \frac{1}{\sqrt{2}} t'(1-\alpha; h, (h-1)(q-1))$$

where $t'(p; \nu_1, \nu_2)$ denotes quantile p of the studentized range distribution with ν_1 and ν_2 degrees of freedom.

A *paired comparison plot* (see Figure 5 in Section 4.2) can be used to visualize Tukey's multiple comparisons when the design is perfectly balanced. It consists of plotting around each estimated mean, e.g., $\hat{\mu}_{.j}$, an interval whose limits are $\bar{y}_{.j} \pm Ts[\hat{D}]/2$. When intervals overlap in this plot we conclude that there is not significant difference between the means compared. The advantage of this plot is that it shows at the same time the significance and the entity of the differences.

2.3 Case $\langle$-, $q(M), r\rangle$: Nested effects design

In the previously considered designs the instances were assumed to be sampled from some (homogeneous) population of instances, whereafter they were solved by all algorithms. Each instance was combined with all possible levels of the algorithmic factors and hence the instance factor and the algorithmic factors were crossed. It is clear that, besides the algorithmic factors, also the characteristics of the instances may affect the performance measure, and that to study these formally we have to include them as fixed factors in the experimental design. Assume that the instances can be characterized by M factors, each of them having a given number of levels. The combination of the levels of these factors defines an instance class and the instances are then randomly sampled from it. As such, instances are specific for the given combination of levels of the instance factors, meaning that they are not crossed with the instance factors, but nested within them.

We consider the simplest design where there is only one instance factor with m possible levels, defining m instance classes. From each instance class (factor level), q instances are randomly sampled and subsequently solved r times by a single algorithm. We denote here by Y_{ijk} the random performance measure obtained in the kth replication of the algorithm on the ith instance sampled from the jth class. Hereafter, we use the subscript $i(j)$ to indicate that the ith factor level of the random factor is nested within the jth factor level of the fixed factor. A possible model for this design is

$$Y_{ijk} = \mu + \beta_j + \tau_{i(j)} + \epsilon_{ijk} \tag{14}$$

where

μ is an overall performance level common to all observations,

β_j is a fixed effect due to the instance class j,

$\tau_{i(j)}$ is a random effect associated with instance i sampled from class j,

Class 1			Class 2		
Instance			Instance		
1	2	3	1	2	3
Y_{111}	Y_{211}	Y_{311}	Y_{121}	Y_{221}	Y_{321}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Y_{11r}	Y_{21r}	Y_{31r}	Y_{12r}	Y_{22r}	Y_{32r}

FIGURE 1: An illustration of nested factor design with $m = 2$ and $q = 3$. The instances within the two classes are different and should be more appropriately identified by $1, 2, 3$ and $4, 5, 6$.

Effects	Square Mean	df	Expected Square Mean	Test Statistics
Fixed Factor	MSF	$m-1$	$\sigma^2 + r\sigma_\tau^2 + rp\frac{\sum_{j=1}^{m}\beta_j^2}{m-1}$	$MSF/MSI(F)$
Nested Factor	$MSI(F)$	$m(p-1)$	$\sigma^2 + r\sigma_\tau^2$	$MSI(F)/MSE$
Error	MSE	$mq(r-1)$	σ^2	

TABLE 2: Expected mean squares and consequent appropriate test statistics for a two-factor nested model. For a generalization to multifactorial cases see rules in Montgomery (2005); Kutner et al (2005)

ε_{ijk} is a random error for replication k on instance i in class j.

The assumptions on the random effects are as follows:

- $\tau_{i(j)}$ are i.i.d. $N(0, \sigma_\tau^2)$,
- ε_{ijk} are i.i.d. $N(0, \sigma^2)$,
- $\tau_{i(j)}$ and ε_{ijk} are independent random variables.

The principle of nesting is illustrated in Fig. 1. Under the above model the response variables are linear combinations of independent normal random variables and hence they follow a normal distribution. To be specific

$$Y_{ijk} \sim N(\mu + \beta_j, \sigma^2 + \sigma_\tau^2), \;\; i = 1, \ldots, q, \;\; j = 1, \ldots, m, \;\; k = 1, \ldots, r.$$

The above model forms the basis for performing inference about the factor effects β_j. On the other hand the instances are randomly drawn from some larger population of instances and we focus on the variability by testing the variance component σ_τ^2 of the instances similarly to what seen in the previous two cases. The quantities needed for developing the tests, and the test statistics themselves are presented in Table 2.

2.4 Case $\langle N, q(M), r\rangle$: General mixed effects design

In this case, the researcher wishes to assess how the performance measure Y is affected by several parameters of the algorithms and of the instances. Ideally, we fix those

parameters that are most important and that we can control, and randomize those properties that we do not understand or cannot control. The parameters controlled may be both categorical or numerical. We consider the following setting:

- Factors $A_1, \ldots, A_N$ represent the parameters of the algorithms. Each combination of these factors gives rise to an instantiated algorithm.
- Factors $B_1, \ldots, B_M$ represent the parameters of the instances (or the *stratification factors* of the whole space of instances). Each combination of these factors gives rise to a different class of instances Π_l.
- From each class of instances Π_l, q instances are sampled randomly and on each of them, each instantiated algorithm is run r times.

The factors A_i, $i = 1, \ldots, N$, and B_j, $j = 1, \ldots, M$, are fixed factors and the factor instance is a random factor. Since the instances within each class Π_l are different the design is *nested*. This yields a linear mixed model that can be written as

$$\begin{aligned} Y_{i_1,\ldots,i_N,j_1,\ldots,j_M,k} = \mu + \alpha_{i_1} + \ldots + \alpha_{i_N} + \beta_{j_1} + \ldots + \beta_{j_M} + \\ + \tau_{k(j_1,\ldots,j_M)} + \varepsilon_{i_1,\ldots,i_N,j_1,\ldots,j_M,k} \end{aligned} \tag{15}$$

with

$\{i_1, \ldots, i_N\}$ an index set referring to the levels of the algorithmic factors $A_1, \ldots, A_M$;

$\{j_1, \ldots, j_M\}$ an index set referring to the levels of the instance factors $B_1, \ldots, B_N$;

$\alpha_{i_1}, \ldots, \alpha_{i_N}$ the main effects and interactions of the algorithmic factors $A_1, \ldots, A_M$;

$\beta_{j_1}, \ldots, \beta_{j_M}$ the main effects and the interactions of the instance factors $B_1, \ldots, B_M$,

$\tau_{k(j_1,\ldots,j_M)}$ the random effect of instance k in setting $\{j_1, \ldots, j_M\}$ of the instance factors,

$\varepsilon_{i_1,\ldots,i_N,j_1,\ldots,j_M,k}$ a random error term.

and where, for brevity, we omitted the interaction terms between all fixed factors.

The analysis of this model is a generalization of those outlined in the previous cases. However, it is more convenient to cast the current model into the framework of the linear mixed model (LMM), where (15) can be rewritten as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b} + \boldsymbol{\varepsilon} \tag{16}$$

with

$\mathbf{Y}$ a n-vector that contains the response variables,

$\mathbf{X}$ a known $n \times k$ matrix, the design matrix associated with the fixed regression coefficients,

$\boldsymbol{\beta}$ is a k-vector that contains the fixed regression coefficients,

$\mathbf{Z}$ a known $n \times q$ matrix, the design matrix associated with the random regression coefficients,

$\mathbf{b}$ a q-vector that contains the random regression coefficients,

$\boldsymbol{\varepsilon}$ a n-vector of error terms.

The terms $\mu + \alpha_{i_1} + \ldots + \alpha_{i_N} + \beta_{j_1} + \ldots + \beta_{j_M}$ for all combinations of indices are now represented by $\mathbf{X}\boldsymbol{\beta}$ and the model is more general because it allows to include both qualitative and quantitative variables while in the models encountered above variables were bounded to be qualitative. Model (16) contains two random components, namely $\mathbf{b}$ and $\boldsymbol{\varepsilon}$, for which we make the following distributional assumptions:

$$\mathbf{b} \sim N_q(\mathbf{0}, \mathbf{D}) \qquad \text{and} \qquad \boldsymbol{\varepsilon} \sim N_n(\mathbf{0}, \boldsymbol{\Sigma})$$

with $\mathbf{b}$ and $\boldsymbol{\varepsilon}$ independent random vectors, and where $N_\ell(\boldsymbol{\mu}, \boldsymbol{\Psi})$ denotes the ℓ-variate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Psi}$. Clearly, the LMM satisfies the conditional and marginal structures outlined in the introduction of Section 2; in particular, conditionally,

$$\mathbf{Y}|\mathbf{b} \sim N_n(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}, \boldsymbol{\Sigma}) \tag{17}$$

and, marginally,

$$\mathbf{Y} \sim N_n(\mathbf{X}\boldsymbol{\beta}, \mathbf{V}[\boldsymbol{\alpha}]), \tag{18}$$

where $\mathbf{V}[\boldsymbol{\alpha}] = \mathbf{Z}\mathbf{D}\mathbf{Z}' + \boldsymbol{\Sigma}$. We use the notation $\mathbf{V}[\boldsymbol{\alpha}]$ here in order to indicate explicitly the dependence of the marginal covariance matrix on an unknown vector $\boldsymbol{\alpha}$ of parameters in the covariance matrices $\mathbf{D}$ and $\boldsymbol{\Sigma}$. Although several methods are available to estimate the LMM, the classical approach is based on maximum likelihood estimation of the marginal model (18). According to the latter, one maximizes the likelihood function, given by

$$L(\boldsymbol{\theta}) = \frac{1}{(2\pi)^{n/2}|\mathbf{V}[\boldsymbol{\alpha}]|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{V}^{-1}[\boldsymbol{\alpha}](\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})\right],$$

with respect to the vector $\boldsymbol{\theta} = (\boldsymbol{\beta}, \boldsymbol{\alpha})$ of unknown model parameters, leading to the maximum likelihood estimator (MLE) $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\alpha}})$. For a given fixed $\boldsymbol{\alpha}$, the MLE for $\boldsymbol{\beta}$ can be obtained explicitly, and is given by

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}^{-1}[\boldsymbol{\alpha}]\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}[\boldsymbol{\alpha}]\mathbf{Y},$$

the well known generalized least squares estimator for the marginal model (18). However, in practice $\boldsymbol{\alpha}$ is typically unknown and hence it must be replaced by an estimate. For this one often uses the restricted maximum likelihood (REML) method, which allows one to estimate $\boldsymbol{\alpha}$ without having to estimate the parameters of $\boldsymbol{\beta}$ first. The basic idea of the REML method is to form linear combinations $\mathbf{K}'\mathbf{Y}$, where $\mathbf{K}$ is a matrix of full column rank, such that the joint distribution of these transformed data no longer depends on $\boldsymbol{\beta}$. This is achieved by constructing $\mathbf{K}$ having columns orthogonal to the columns of $\mathbf{X}$, i.e., $\mathbf{K}'\mathbf{X} = \mathbf{0}$. Another motivation for the REML method stems from the fact that it produces estimates that are less biased than the MLEs.

An appealing feature of the likelihood framework is that it provides a general procedure for testing hypotheses about model parameters, by simply comparing two likelihood values: the likelihood of a restricted model, the null model, and that of an unrestricted model, also referred to as the full model. This approach to hypothesis testing is especially useful in complex and unbalanced designs, where exact tests as the F tests described above are typically unavailable. Formally, consider the above introduced parameter vector $\boldsymbol{\theta}$ and its associated parameter space $\boldsymbol{\Theta}$ (the set of possible values for $\boldsymbol{\theta}$), so $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, and let $\boldsymbol{\theta}_1$ denote the subvector of $\boldsymbol{\theta}$ that is of interest for testing. In other terms, let the vector $\boldsymbol{\theta}_1$ contain the parameters of the unrestricted model that are not contained in the restricted model. Under the null hypothesis $H_0 : \boldsymbol{\theta}_1 = \boldsymbol{\theta}_{10}$, the parameter vector $\boldsymbol{\theta}$ is restricted to lie in some subset $\boldsymbol{\Theta}_0$ of $\boldsymbol{\Theta}$. To test the hypothesis about $\boldsymbol{\theta}_1$ one computes the likelihood ratio test statistic, expressed by the ratio between the maximum likelihood of the sample data under the restricted model and the one under the unrestricted model, i.e.,

$$\Lambda = \frac{\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_0} L(\boldsymbol{\theta})}{\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} L(\boldsymbol{\theta})},$$

and rejects H_0 if Λ is too small. It can be shown that under certain regularity conditions and assuming $H_0 : \boldsymbol{\theta}_1 = \boldsymbol{\theta}_{10}$ holds, $-2 \ln \Lambda$ is approximately distributed as χ^2_ν, provided the sample size is large. The degrees of freedom ν of the approximating chi-squared distribution are given by the difference of the dimensions of the parameter spaces $\boldsymbol{\Theta}_0$ and $\boldsymbol{\Theta}$. One of the assumptions for having a chi-squared limiting distribution for $-2 \ln \Lambda$ is that the parameters in the null hypothesis are not on the boundary of the parameter space. In the LMM we are often interested in testing a hypothesis about a random effect that takes the form $H_0 : \sigma^2 = 0$, which constitutes a violation of this assumption. If one uses in such a case the chi-squared approximation with its usual degrees of freedom, then the test will be conservative. For a detailed description of the asymptotic properties of the maximum likelihood method and the likelihood ratio test statistic we refer the reader to Lehmann (2003) and Lehmann and Romano (2008).

3 Augmenting a tree to a 2-edge-connected graph

In this section we briefly describe the application example that we will develop in the next section. The example is extracted from a study on heuristic and exact algorithms for the so-called E1-2AUG problem (Bang-Jensen et al, 2009). Here, we focus on an intermediate result of that work concerning local search algorithms. We first describe the problem and, then, sketch the local search schemes from which the algorithms are derived; finally we introduce the test instances.

3.1 Definitions and problem formulation

In graph theory terminology (see, for example, Bondy and Murty 2008), an edge uv in a connected graph $G = (V, E)$ is a bridge if we can partition V into two sets $S, V - S$ so that uv is the only edge from E with endpoints in both S and $V - S$. A graph is *2-edge-connected* if it is connected and has no bridges.

The 2-edge-connectivity augmentation (E1-2AUG) problem asks for a given undirected 2-edge-connected graph $G = (V, E)$, a fixed spanning connected subgraph of G,

$S = (V, F)$, and a non-negative weight function ω on $E' = E - F$, to find a subset X of E' of minimal total weight so that $A(G) = (V, F \cup X)$ is 2-edge-connected.

We restrict ourselves to only the cases where the graph G is a simple graph and S is a tree. An edge $uv \in E$ which is not in S is said to *cover* those edges of S which correspond to the unique uv-path P_{uv} in S. We assume that every edge uv in F is covered by at least two edges in E'. We call a subset X of E' a *proper augmentation* of S if $A(G) = (V, F \cup X)$ is 2-edge-connected.

Every optimal augmentation X is minimal, that is, no edge can be deleted from X without leaving at least one edge of S uncovered. If a given augmentation is not minimal it can be made so by means of a *trimming* procedure that removes edges from X without leaving any edge of S uncovered.

It can be shown that the E1-2AUG problem is a special case of the general set covering problem (Conforti et al, 2004). That is, the minimal weight augmentation corresponds to the minimal weight selection of edges such that every edge of F is covered by at least an edge from E'.

3.2 Local search algorithms

Three construction heuristics named lightest addition (`la`), shortest path (`sp`) and greedy covering (`gc`) have been designed in Bang-Jensen et al (2009). To improve the solution provided by these heuristics, three local search schemes are used. They are based on a first improvement strategy and on three different neighborhood structures.

Addition neighborhood (`addn`) Neighboring augmentations are obtained by adding k edges from $E' - X$ and trimming the resulting augmentation.

Destruct-reconstruct neighborhood (`gcn`) Neighboring augmentations are obtained by removing k edges from the current augmentation and reconstructing the resulting improper augmentation by means of the greedy set covering heuristic by Chvatal (Cormen et al, 2001, pag. 1035).

Shortest path neighborhood (`spn`) It consists of deleting k edges and finding the shortest path between pairs of their ending vertices in a suitable digraph. The digraph is constructed considering edges available for the augmentation and not allowing to reinsert deleted edges. After the insertion of the new edges the augmentation is trimmed to make it again minimal.

Our task is to assess empirically the impact of three factors: the construction heuristic, the local search scheme identified by its neighborhood and the parameter k common to all neighborhoods.

3.3 Problem instances

In the experiments, we sample the space of instances of the E1-2AUG problem by restricting ourselves to only a portion of it and by stratifying this portion according to three instance characteristics: *type* of graphs, *edge density* and *distribution of weights*.[2]

[2]Note that the process of sampling should be designed carefully in order to avoid pitfalls like bias towards some instances rather than others. For example, the possible non-isomorphic graphs of size 800 are more than those of size 200, hence they should be given more probability to appear. This

The type distinguishes between uniform graphs (Type `U`), geometric graphs (Type `G`) and small world graphs (Type `sm`) (see Bang-Jensen et al, 2009 for definitions). In all types of graphs, the spanning tree S is chosen randomly. All graphs may have random weights on their edges (`r`) or have uniform weights (`1`). The edge density is a measure of the amount of edges present in the graph and we consider three possibilities, high, medium and low, {`h`,`m`,`l`}.

4 Experimental analysis

We measure the performance of a run of an algorithm on an instance π by the *gap* or *percent error* of the lower bound approximation, i.e., $(z(\pi) - z^*(\pi))/z^*(\pi) \cdot 100$, where $z(\pi)$ is the observed solution cost in that run of the algorithm and $z^*(\pi)$ is the lower bound on the solution costs for that instance. This measure is feasible in our example problem because a good lower bound can be determined for several instances in relatively short time by integer programming. In fact, for most of the cases the lower bound used is also proved to be the optimal solution. Other measures of solution quality are possible. Zemel (1981) points out that a criterion for judging quality measures is the invariance to simple transformation of the instances. Another measure of interest, not based on solution quality, might be the computation time, since the algorithms in the study have all a natural termination condition (the attainment of a local optimum).[3]

We now develop the analysis on the local search algorithms for the E1-2AUG problem proceeding case by case in the same order as in Section 2. The analysis is conducted with the statistical package R (R Development Core Team, 2008) and in the text we give the main commands to execute this analysis. In the online compendium `http://www.imada.sdu.dk/~marco/Mixed/` we report the data, the full code in R and the same analysis in the statistical software package SAS.

4.1 Case $\langle$-, q(-), $r\rangle$: Random effects design

The goal of this simple case is to illustrate the decomposition of the variance of the response observations in two components, namely, the variability of the results due to the stochasticity of the algorithm and the variability due to the instances sampled from the population. Moreover, we derive an estimation of solution quality with associated confidence intervals.

We collect the results of one algorithm run a number of times on a set of instances. Precisely, the algorithm is determined by the choices `gc`, `addn` and `k3` and the instance class by the choices `G`, `m` and `1`.

In the design of the experiment, we decide the number of runs and the number of instances on the basis of considerations on the level of significance and on the power. In particular, we fix the level of significance to 0.05 and aim at a statistical power of 0.8. (These values are maintained throughout the remainder of the chapter.) We then

problem is solved if stratification is applied and the stratifying factor, in the example the size of the graph, is included in the analysis.

[3]Often local search algorithms are enhanced by metaheuristics (Glover and Kochenberger, 2002) and do not have anymore a natural termination condition. In this case, computation time can be seen as an external parameter and treated as an algorithmic fixed factor in the models here discussed.

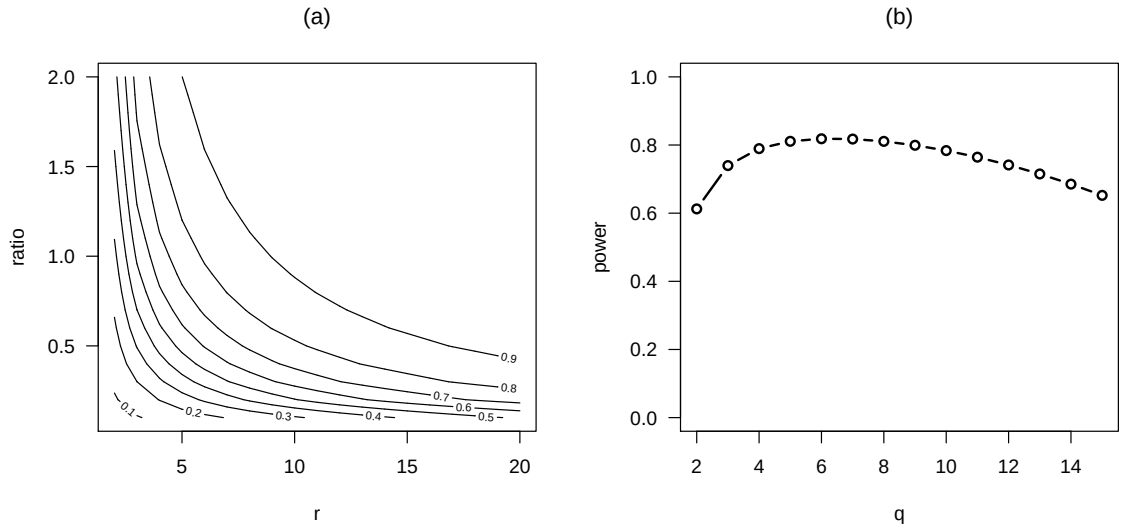

FIGURE 2: Statistical power for the case $\langle -, q(-), r\rangle$. The power is a function of three variables: $\tilde{\sigma}^2_\tau/\sigma^2, q, r$. The plot on the left shows a contour plot of the power surface as a function of r and $\tilde{\sigma}^2_\tau/\sigma^2$ when $q = 5$. The plot on the right shows the power as a function of q for a total number of experiments $qr = 30$, when $\tilde{\sigma}^2_\tau/\sigma^2 = 1$.

use Formula (10) to compute the value of power as a function of r, q and $\tilde{\sigma}^2_\tau/\sigma^2$ and we visualize this function in two alternative ways in Figure 2.

The two plots represent different views of $POWER(\tilde{\sigma}^2_\tau/\sigma^2, q, r)$. In Figure 2 (a) we show the contour plot of the $POWER$ surface when considered as a function of $\tilde{\sigma}^2_\tau/\sigma^2$ (called ratio) and r, for a fixed value of q (here $q = 5$). Each curve in this plot corresponds to a specific power level, and represents the $(r, \tilde{\sigma}^2_\tau/\sigma^2)$ combinations for which this power is achieved. For instance, if one wants to detect with a 5% significance test a $\tilde{\sigma}^2_\tau$ which is of the same magnitude as σ^2 (corresponding to ratio=1) with a probability of 0.8 in an experiment with 5 instances, then one has to collect 6 replicates to achieve this level. Figure 2 (b) contains an alternative representation of the $POWER$ function. Here we show the power of the test when $\tilde{\sigma}^2_\tau/\sigma^2 = 1$ as a function of the number of instances q, when the total number of experiments is fixed at 30, i.e. when $qr = 30$. This bound on the number of experiments can be posed by consideration on the computational time available. This plot has a peak around $q = 6$ hinting at a $POWER$-optimal design under the conditions stated. We use this observation to conclude that in order to have a power of 0.8 when $\tilde{\sigma}^2_\tau/\sigma^2 = 1$, which we deem a relevant value for practical purposes, then we need to collect at least $r = 30/q = 5$ runs on $q = 6$ instances.[4]

In a second step, we analyze the results of an experiment in which 6 instances were randomly sampled from the class `G-m-1`, whereafter they were solved 5 times with the algorithm `gc-addn-k3`. We load the data in R stored in a data frame and check its content by means of the command `str`. The data frame is organized in two columns,

[4]In the next designs we omit the details of power computations. A computer program by Lenth (2006) for these computations is available online.

`instance` and `algorithm`, that indicate the factors of each observation, a column `run` that reports the replicate number and a column `gap` that gives the response variable:

```
> load("Data/OPOR.dataR")
> str(OPOR, strict.width = "cut", width = 70)
Soutput
'data.frame':        30 obs. of  4 variables:
 $ instance : Factor w/ 6 levels "G-800-0.5-1-pre.ins",..: 1 1 1 1 1..
 $ run      : int  5 1 3 2 4 5 3 4 2 1 ...
 $ algorithm: Factor w/ 1 level "gc-addn-3": 1 1 1 1 1 1 1 1 1 1 ...
 $ gap      : num  4.07 4.05 4.07 4.07 4.07 ...
```

Before presenting the results of the analysis and performing hypothesis testing using the random effects model described in Section 2.1, we comment on the validity of the model assumptions. Under model (4) the response variables are normally distributed with mean μ and variance $\sigma^2 + \sigma_\tau^2$, and we validate this assumption using a normal quantile plot (QQ plot). In such a plot we compare the empirical quantiles (the ordered data) with the corresponding quantiles of a standard normal model. In case the normality assumption holds then the points on the QQ plot will show a straight line pattern (see also the appendix of this book). For the experiment under consideration we show this plot in Figure 3 (a). Clearly, the points are quite tightly concentrated along a straight line, indicating that a normal model is plausible for our data. In Figure 3 (b) and (c), we report the quantile plots also for two other cases that we examined. More precisely, the second QQ plot is obtained for a continuous optimization problem, namely, the least median of squares, a robust way to estimate parameters in linear regression analysis (Rousseeuw, 1984). The problem with this plot is that the tails are very far from being normally distributed, with the tails of the empirical distribution being lighter than normal tails. The third plot is based on a study for the graph coloring problem (Chiarandini, 2005). In this case, the major problem is that data, corresponding to the minimal number of colors used, are discrete and distributed among only few values. The whole methodology developed in the chapter works for continuous objective functions. When data are discrete but, contrary to the case of the third plot, have many possible values, data can still be reasonably approximated by a continuous distribution.

We now turn to fitting the random effects model (4) and to the estimates and inference about its parameters. Random (and mixed) effects models can be fitted with the function `lmer` of the package `lme4` (Bates et al, 2008).[5] Below we show the resulting R output.

```
> library(lme4)
> fm1a <- lmer(gap ~ 1 + (1 | instance), data = OPOR)
> print(fm1a, digits = 3, corr = FALSE)
Soutput
Linear mixed model fit by REML
Formula: gap ~ 1 + (1 | instance)
   Data: OPOR
  AIC   BIC logLik deviance REMLdev
 -101 -96.6   53.4     -105    -107
```

[5]The package `nlme` (Pinheiro et al, 2008) can also treat mixed-effects models.

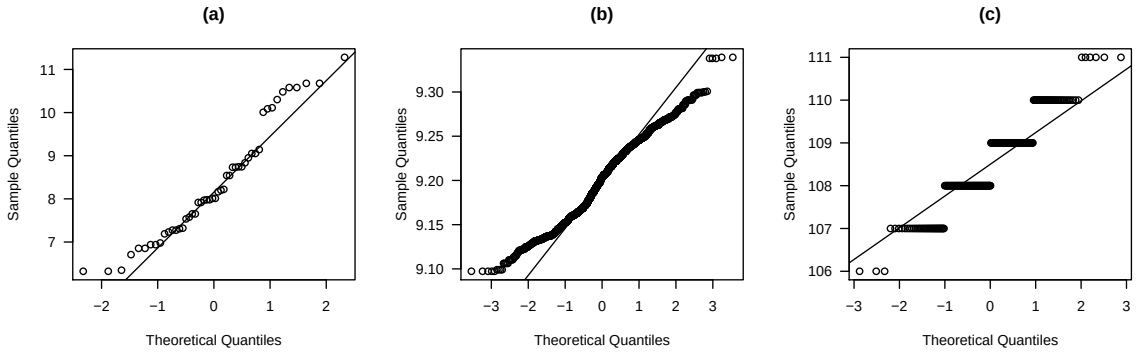

FIGURE 3: The distribution of data. The left most plot shows the quantile distribution of `gc-addn-k3` when run 5 times on 10 instances of the class `G-800-0.5`. The plot in the center shows 5 runs on 500 instances on a different problem where the distribution shows strong deviance from normality even after data transformation. The right most plot shows the distribution of quantiles from 5 runs on 60 instances. Here the problem is discreteness of data.

```
Random effects:
 Groups   Name        Variance Std.Dev.
 instance (Intercept) 4.362580 2.0887
 Residual             0.000173 0.0131
Number of obs: 30, groups: instance, 6
Fixed effects:
            Estimate Std. Error t value
(Intercept)    4.559      0.853    5.35
```

Since there are no fixed effects, the model (4) passed to `lmer` contains only `1` that represents the intercept μ. The random effect is expressed by `(1 | instance)` indicating that the data is grouped by `instance` and that the random effect is constant within each group, `1`. By default `lmer` uses the restricted maximum likelihood (REML) method to fit the model. The output provides information about some of the measures of the fitting such as the log-likelihood (53.4), the deviance for the maximum likelihood criterion (-105), the deviance for the REML criterion (-107), Akaike's Information Criterion (AIC=-101) and Schwartz's Bayesian Information Criterion (BIC=-96.6). Under the header `Fixed effects`, we find the estimate for the intercept μ while under `Random effects` we find the estimates for the parameters related to the random effects and the error distributions, here the standard deviations for τ (`instance`) and ε (`Residuals`), respectively. For our experiment we obtain $\hat{\sigma}_\tau = 2.0887$ and $\hat{\sigma} = 0.0131$, which indicates that the variability in the response observations can be mainly attributed to the variability of the instances.

By default the `lmer` function does not report the test on the hypothesis about the variance components σ_τ^2 and σ^2. This is because in general for unbalanced data the computation of the test is not trivial. However, in the cases of perfectly balanced experiments, like ours, we can proceed to compute the F statistic and the p-value on the basis of (8) and (9), by plugging the estimates into the equations for the expected

mean squares. We get

$$\begin{aligned} MSI &= \hat{\sigma}^2 + r\hat{\sigma}_\tau^2 \\ &= 0.0131^2 + 5(2.0887)^2 \\ MSE &= \hat{\sigma}^2 \\ &= 0.0131^2, \end{aligned}$$

and hence $f_1 = 126283$, with p-value≈ 0 thus the null hypothesis is to be rejected. In R:

```
> VC <- VarCorr(fm1a)
> sigma.tau <- as.numeric(attr(VC$instance, "stddev"))
> sigma <- as.numeric(attr(VC, "sc"))
> q <- nlevels(OPOR$instance)
> r <- length(unique(OPOR$run))
> MSI <- sigma^2 + r * sigma.tau^2
> MSE <- sigma^2
> 1 - pf(MSI/MSE, q - 1, q * (r - 1))
Soutput
[1] 0
```

We can compute the test on the random effects also by using the likelihood ratio test. In this case, for the likelihood of the model without fixed effects we have to use the function `lm`

```
> fm1a <- lmer(gap ~ 1 + (1 | instance), data = OPOR, REML = FALSE)
> fm1a.0 <- lm(gap ~ 1, data = OPOR)
> LRT <- as.numeric(2 * (logLik(fm1a) - logLik(fm1a.0)))
> 1 - pchisq(LRT, 1)
Soutput
[1] 0
```

The test confirms the rejection of the null hypothesis. Note that we perform here a test where the parameter of the null hypothesis is on the boundary of the parameter space, and hence, as noted before, the classical chi-squared approximation to the null distribution of the likelihood ratio test is inappropriate. For this particular case where we test the importance of a single variance component, the limiting distribution of the likelihood ratio statistic is a mixture of a point mass at zero and a chi-squared distribution with one degree of freedom, where both components of the mixture have probability one. This implies that the usual p-value needs to be divided by two, or, otherwise stated, that the classical test is conservative (Stram and Lee, 1994, 1995).

Finally, if we wish to predict the performance of the algorithm on a new instance, the best we can do is to give $\hat{\mu} = 4.559$ and to give the 95% confidence interval. According to (11) of Section 2.1:

```
> s <- sqrt(MSI/(q * r))
> Y.. <- mean(OPOR$gap)
> qsr <- qt(1 - 0.025, 5)
> Y.. - qsr * s
```

```
Soutput
[1] 2.37
> Y.. + qsr * s
Soutput
[1] 6.75
```

hence μ in $[2.37; 6.75]$.

4.2 Case $\langle N, q(\text{-}), r\rangle$: Mixed effects design

We discuss two designs within this case: $\langle 1, q(-), r\rangle$ and $\langle N, q(-), r\rangle$. The focus is, in the first design, on the visualization of the results and, in the second design, on the comparison of replicated *vs* unreplicated designs.

$\langle 1, q(-), r\rangle$ In the first design we aim at comparing the performance of the addition neighborhood at different values of k, over an instance class. More precisely, we have the following factors:

- `algorithm`: three algorithms, starting from the solution produced by greedy covering (`gc`) and using the k-addition neighborhood (`addn`) with $k = \{1, 3, 5\}$, hence levels in `{gc-addn-1,gc-addn-3,gc-addn-5}`;
- `instance`: 5 instances randomly sampled from the class `G-m-1`;
- `replicates`: 5.

```
> load("Data/YPOR.dataR")
> str(YPOR, strict.width = "cut", width = 70)
Soutput
'data.frame':        75 obs. of  5 variables:
 $ instance : Factor w/ 5 levels "G-800-0.5-1-pre.ins",..: 1 1 1 1 1..
 $ k        : Factor w/ 3 levels "1","3","5": 3 2 1 2 1 3 3 1 2 2 ...
 $ algorithm: Factor w/ 3 levels "gc-addn-1","gc-addn-3",..: 3 2 1 2..
 $ run      : int  5 2 4 1 1 1 5 5 4 3 ...
 $ gap      : num  3.67 4.07 4.07 4.05 4.05 ...
```

Relevant questions for this design are

- Is there an instance effect, i.e., do the instances contribute significantly to the variability of the responses?
- Do the mean performances of the algorithms with different k differ? If yes, how different are they?
- Do the instance-algorithm interactions contribute significantly to the variability of the responses?

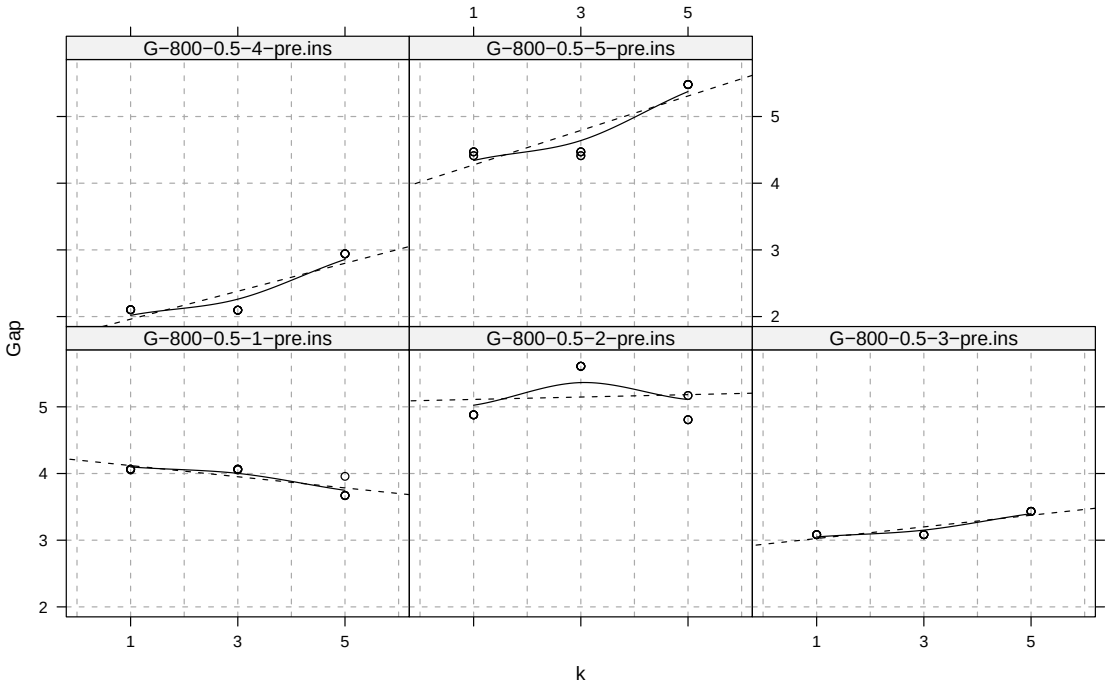

FIGURE 4: The data in the design $\langle 1, q(-), r\rangle$. The three algorithms, `gc-addn-1`, `gc-addn-3`, `gc-addn-5`, correspond to greedy covering construction (`gc`) followed by k-addition neighborhood (`addn`) with $k = \{1, 3, 5\}$. A local regression line and a least square linear regression line (dashed) are superimposed.

We treat k as if it were a qualitative factor, even though k is a numerical value with a clear order. Treating discrete numerical factors as qualitative factors gives more freedom, because in this way we do no assume that the change in the mean response for k from 1 to 3 has to be the same as for k from 3 to 5.

A way to inspect the data is by plotting the percentage error of the algorithms within each instance, which is interpreted as a different group of results. This is shown in Figure 4. We observe that there are differences in the slopes and intercepts of the linear regressions within each group. This hints at the presence of random effects and interaction effects between the random and the fixed factors. We will therefore test the inclusion of both a random instance intercept and a random instance-algorithm interaction in the model that describes these data.

The results of the analysis of the mixed model are:

```
> op <- options(contrasts = c("contr.sum", "contr.poly"))
> fm2a <- lmer(gap ~ k + (1 | instance) + (1 | instance:k),
     data = YPOR)
> print(fm2a, digits = 3)
Soutput
Linear mixed model fit by REML
Formula: gap ~ k + (1 | instance) + (1 | instance:k)
   Data: YPOR
 AIC   BIC logLik deviance REMLdev
 -95 -81.1   53.5     -111    -107
```

```
Random effects:
 Groups     Name        Variance Std.Dev.
 instance:k (Intercept) 0.15795  0.3974
 instance   (Intercept) 1.23514  1.1114
 Residual               0.00387  0.0622
Number of obs: 75, groups: instance:k, 15; instance, 5
Fixed effects:
            Estimate Std. Error t value
(Intercept)   3.8939     0.5075    7.67
k1           -0.1786     0.1455   -1.23
k2           -0.0342     0.1455   -0.23

Correlation of Fixed Effects:
   (Intr) k1
k1  0.000
k2  0.000 -0.500
```

We have specified sum contrasts, here, as a way to identify parameters in the model instead of the default treatment contrasts for `lmer`. This will make later results comparable with `lm`. The estimated variances for the instance and the instance-algorithm interaction random effects are $\hat{\sigma}^2_\tau = 1.23514$ and $\hat{\sigma}^2_\gamma = 0.15795$, respectively. The section `Fixed effects` reports the estimates of the fixed effects model parameters from which we obtain the point estimates for the mean performance of the algorithms $E[Y_{ijk}] = \mu + \alpha_j$. The sum contrasts specified before implies that $\sum \alpha_j = 0$. Hence, for $\alpha_{\texttt{k1}} = -0.1786$ and $\alpha_{\texttt{k2}} = -0.0342$, we have $\alpha_{\texttt{k3}} = 0.2128$, with `k1` representing $k = 1$, `k2`, $k = 3$ and `k3`, $k = 5$. The last column in this section gives the t statistics for the hypotheses that the jth level of the factor is not different from the mean response.

Let's look at the acceptance or rejection of the null hypothesis that the variance components of the random effects are zero. The exact test is via the F-ratio from Table 1 of Section 2.2. If we do not want to look up the table the likelihood ratio test can be computed more easily.

```
> fm2a.1 <- lmer(gap ~ k + (1 | instance), data = YPOR,
     REML = FALSE)
> fm2a.2 <- lmer(gap ~ k + (1 | instance) + (1 | instance:k),
     data = YPOR, REML = FALSE)
> anova(fm2a.2, fm2a.1)
Soutput
Data: YPOR
Models:
fm2a.1: gap ~ k + (1 | instance)
fm2a.2: gap ~ k + (1 | instance) + (1 | instance:k)
       Df   AIC   BIC logLik Chisq Chi Df Pr(>Chisq)
fm2a.1  5  71.0  82.6  -30.5
fm2a.2  6 -99.2 -85.3   55.6   172      1     <2e-16
```

As is clear, the instance-algorithm interactions contribute significantly to the variability of the performance measure, and hence, given (13), measurements obtained on a particular instance show dependence. A similar test can be performed for the instance variance, by fitting a model without an instance random effect:

```
> fm2a.3 <- lmer(gap ~ k + (1 | instance:k), data = YPOR,
     REML = FALSE)
> anova(fm2a.2, fm2a.3)
Soutput
Data: YPOR
Models:
fm2a.3: gap ~ k + (1 | instance:k)
fm2a.2: gap ~ k + (1 | instance) + (1 | instance:k)
       Df   AIC   BIC logLik Chisq Chi Df Pr(>Chisq)
fm2a.3  5 -84.6 -73.0   47.3
fm2a.2  6 -99.2 -85.3   55.6  16.6      1    4.5e-05
```

Hence also this term is significant and should be included in the model.

Let's now analyze the significance of fixed effects. We use the F-ratio[6]

```
> anova(fm2a)
Soutput
Analysis of Variance Table
  Df  Sum Sq Mean Sq F value
k  2 0.00956 0.00478    1.23
```

The `lmer` function does not return the p-value for the test on fixed-effects terms but the F statistic computed by the `anova` function is the correct one for balanced designs. Hence, the observed F statistic is 1.23 on 2 (`Df`) and $(h-1)(q-1) = 8$ degrees of freedom and with a p-value of 0.341,

```
> p <- nlevels(YPOR$instance)
> h <- nlevels(YPOR$k)
> r <- length(unique(YPOR$run))
> 1 - pf(anova(fm2a)$"F value", h - 1, (h - 1) * (p - 1))
Soutput
[1] 0.341
```

We can conclude that there is not a significant effect of k and, hence, that the mean performance measure is not affected by this parameter.

Since we could not reject the global null hypothesis on k, the paired comparison plot will show overlapping confidence intervals for the three values of k. For the sake of completeness in our exposition, we derive this plot that we show in Figure 5, left panel.

```
> VC <- VarCorr(fm2a)
> sigma.gamma <- as.numeric(attr(VC$"instance:k", "stddev"))
> sigma <- as.numeric(attr(VC, "sc"))
> MSIK <- sigma^2 + p * sigma.gamma^2
> Yj. <- with(YPOR, aggregate(gap, list(alg = algorithm),
     mean))
> s <- sqrt(2) * sqrt(MSIK/(p * r))
> T <- qtukey(1 - 0.05, h, (h - 1) * (p - 1))/sqrt(2)
> Yj.$lower <- Yj.$x - 0.5 * T * s
> Yj.$upper <- Yj.$x + 0.5 * T * s
> intervals(alg ~ x, Yj.)
```

[6]Due to implementation issues in R and SAS the likelihood ratio test cannot be used for testing some of the fixed effects, as they remain unidentified (SAS Institute Inc., 2007).

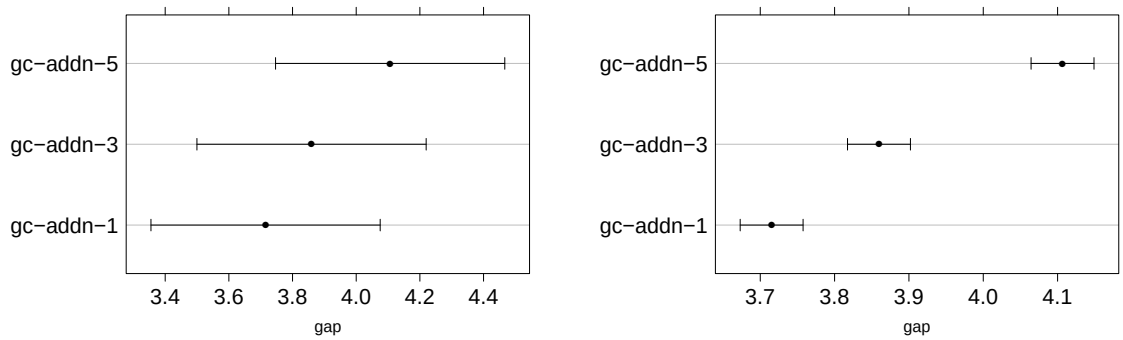

FIGURE 5: Paired comparison plots. On the left the one obtained by the mixed effects model, on the right the one obtained by ordinary ANOVA

The function `intervals` is a wrapper to `dotplot` available from the online compendium at `http://www.imada.sdu.dk/~marco/Mixed/`.

We check the diagnostic plots. We consider the conditional and marginal structure of the model (equation (17) and (18) of Section 2.4, respectively). In the standard diagnostic plots of residuals against fitted values we check the assumption of homoscedasticity of observations, whereas, in the QQplot we check if residuals meet the assumption of normality. Conditional residuals pertain to each instance individually taken and refer to the distances of observed points from the fitted conditional models. Aggregating these data for the 5 instances available we see that there might be some deviation from the assumptions, mainly due to the small variability of the responses within an instance. It might then be worth indicating the instances that cause the largest deviation from the assumptions. Things are instead much better for the marginal structure, which is the one we are mostly interested in our study. The plots seem to support quite well the assumptions of homoscedasticity and normality.

```
> plot(fitted(fm2a, type = "response"), residuals(fm2a,
     type = "response"), main = "Conditional residuals",
     xlab = "Predicted", ylab = "Residuals")
> res <- residuals(fm2a, type = "response")
> qqnorm(res, main = "Conditional residuals, QQplot")
> qqline(res)
> fm2a.0 <- lm(gap ~ k, data = YPOR)
> x <- model.matrix(fm2a.0)
> pred <- x %*% fixef(fm2a)
> res <- YPOR$gap - pred
> plot(pred, res, main = "Marginal residuals", xlab = "Predicted",
     ylab = "Residuals")
> qqnorm(res, main = "Marginal residuals, QQplot")
> qqline(res)
```

Finally, it is instructive to compare the results obtained here under a random effects model with those obtained by considering instances as fixed factors. In this latter case, the test for algorithmic differences is performed relative to the mean squared error, and

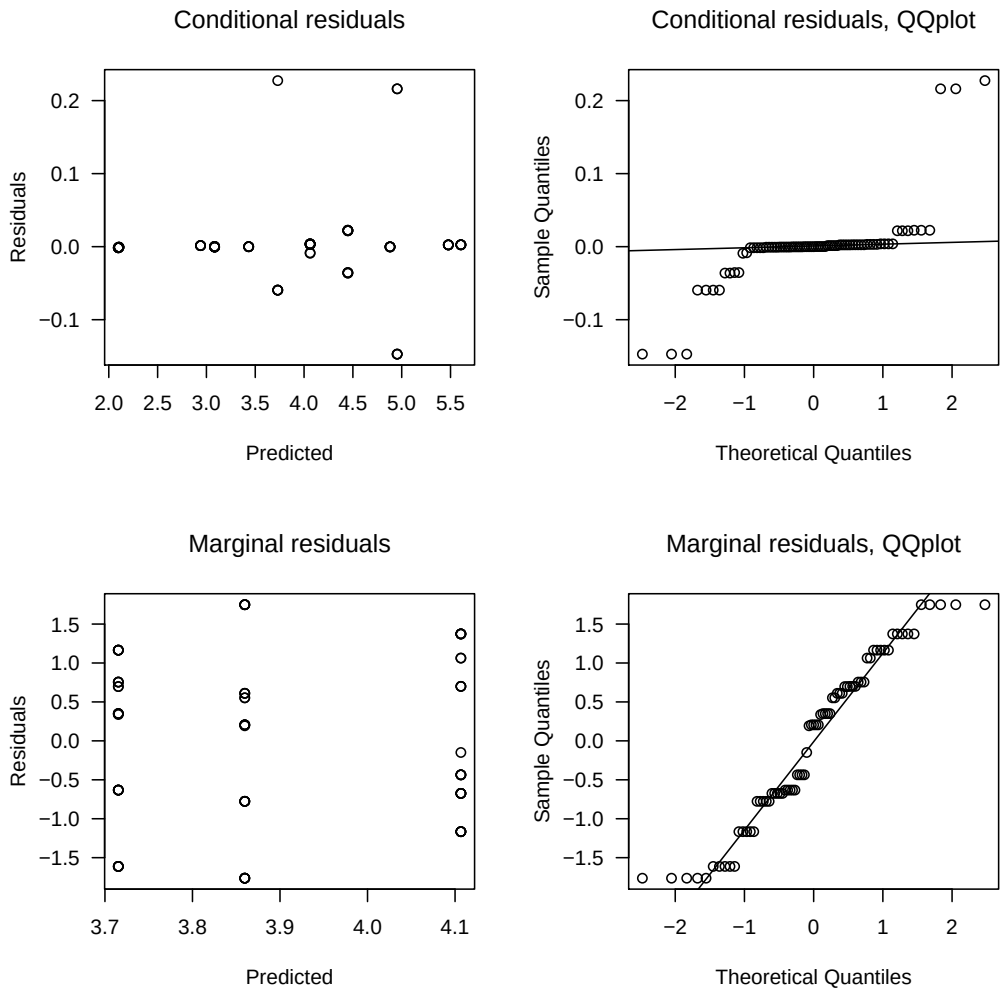

FIGURE 6: Diagnostic plots

not relative to the instance-algorithm interaction mean squares (see Table 1, Section 2.2). Moreover the F test has 60 degrees of freedom at the denominator, compared to 8 under a mixed model, and hence, for the same significance level it would reject sooner.

```
> fm2a.lm <- lm(gap ~ k * instance, data = YPOR)
> anova(fm2a.lm)
Soutput
Analysis of Variance Table
Response: gap
           Df Sum Sq Mean Sq F value Pr(>F)
k           2    2.0     1.0     253 <2e-16
instance    4   77.3    19.3    4986 <2e-16
k:instance  8    6.3     0.8     205 <2e-16
Residuals  60    0.2  0.0039
```

This would have led us to reject the hypothesis on `k` and conclude, mistakenly, that `k` is significant! The different conclusion is also shown in Figure 5, where on the right

we report the paired comparison plot that would arise from a Tukey pairwise analysis based on the fixed effect model of `lm`.

$\langle N, q(-), r\rangle$ We now discuss the case $\langle N, q(-), r\rangle$ and compare replicated and unreplicated designs. This case differs slightly from the previous in that we study three fixed factors and this leads us to a multi-factorial analysis. All fixed factors are algorithmic factors and are tested at three levels.

- `init.heur`: the starting solution generated by three different construction heuristics. It is a categorical factor in the levels `{gc,la,sp}`;
- `neigh`: the three local search schemes determined by their neighborhood structure as described above. It is a categorical factor in the levels `{addn,covn,spn}`;
- `k`: the parameter that determines the extension of the neighborhood. It is a categorical factor in the levels `{1,3,5}`.

The 27 possible combinations give rise to 27 algorithms to test. If our computational budget allows us to run 675 experiments then we can choose between a *replicated design* with 5 instances and 5 runs per instance, or an *unreplicated design* with one single run of each algorithm on 25 instances.

Let's analyze first the *replicated design*.

```
> load("Data/NPOR.dataR")
> str(NPOR, strict.width = "cut", width = 70)
Soutput
'data.frame':        675 obs. of  6 variables:
 $ instance : Factor w/ 5 levels "sm-800-h-w1",..: 1 1 1 1 1 1 1 1 1..
 $ init.heur: Factor w/ 3 levels "gc","la","sp": 1 1 1 2 2 2 2 3 3 1..
 $ neigh    : Factor w/ 3 levels "addn","gcn","spn": 2 2 1 3 2 2 2 1..
 $ k        : Factor w/ 3 levels "1","3","5": 3 2 1 2 3 2 1 2 3 2 ...
 $ run      : int  2 3 1 3 5 5 2 3 3 1 ...
 $ gap      : num  1.521 1.433 0.397 2.976 2.843 ...
```

We test the significance of the random effects and their interactions. The exponent of two in the `lmer` model statement indicates that all interactions of the second order are included.

```
> fm2bR.0 <- lm(gap ~ (k + init.heur + neigh)^2, data = NPOR)
> fm2bR.1 <- lmer(gap ~ (k + init.heur + neigh)^2 + (1 |
     instance), data = NPOR, REML = FALSE)
> fm2bR.2 <- lmer(gap ~ (k + init.heur + neigh)^2 + (1 |
     instance) + (1 | instance:k) + (1 | instance:neigh) +
     (1 | instance:init.heur), data = NPOR, REML = FALSE)
> LRT <- as.numeric(2 * (logLik(fm2bR.2) - logLik(fm2bR.0)))
> 1 - pchisq(LRT, 1)
Soutput
[1] 0
> anova(fm2bR.2, fm2bR.1)
```

```
Soutput
Data: NPOR
Models:
fm2bR.1: gap ~ (k + init.heur + neigh)^2 + (1 | instance)
fm2bR.2: gap ~ (k + init.heur + neigh)^2 + (1 | instance) + (1 | instance:k) +
fm2bR.2:     (1 | instance:neigh) + (1 | instance:init.heur)
        Df  AIC  BIC logLik Chisq Chi Df Pr(>Chisq)
fm2bR.1 21 1301 1396   -630
fm2bR.2 24  672  780   -312   636      3     <2e-16
```

The likelihood ratio test indicates again that the random factor instance is significant and also at least one of the random interaction terms between a fixed factor and the instance factor. For the fixed effects we have

```
> fm2bR <- lmer(gap ~ (k + init.heur + neigh)^2 + (1 |
     instance) + (1 | instance:k) + (1 | instance:neigh) +
     (1 | instance:init.heur), data = NPOR)
> anova(fm2bR)
Soutput
Analysis of Variance Table
                Df Sum Sq Mean Sq F value
k                2    0.4     0.2    1.56
init.heur        2   10.5     5.2   41.65
neigh            2    3.6     1.8   14.36
k:init.heur      4    0.3     0.1    0.69
k:neigh          4   39.6     9.9   78.54
init.heur:neigh  4   37.5     9.4   74.43
```

We omit here the details of the analysis of variance, which is similar to the previous case. It yields a p-value of 0.2675 for `k` and of 0.5984 for the interaction `k:init.heur` thus leading us to not reject the null hypothesis of no effect for these two factors. The latter result was expected, given that k does not alter the construction heuristics. All other effects are instead significant. To gain insight when interaction terms are significant one can use 2D or 3D interaction plots. In Figure 7 we visualize the interactions `neigh:init.heur` and `k:neigh`.

```
> with(NPOR, {
     interaction.plot(neigh, init.heur, gap, fixed = TRUE)
     interaction.plot(k, neigh, gap, fixed = TRUE)
 })
```

For later comparisons we report also the estimates of the fixed effects:

```
> summary(fm2bR)@coefs[1:10, ]
Soutput
              Estimate Std. Error t value
(Intercept)     2.3800     0.2509   9.485
k1              0.0340     0.0375   0.909
k2             -0.0662     0.0375  -1.767
init.heur1     -1.0076     0.1118  -9.016
init.heur2      0.3669     0.1118   3.283
neigh1         -1.1496     0.2180  -5.273
neigh2          0.3952     0.2180   1.813
k1:init.heur1   0.0131     0.0273   0.480
k2:init.heur1  -0.0147     0.0273  -0.538
k1:init.heur2  -0.0421     0.0273  -1.542
```

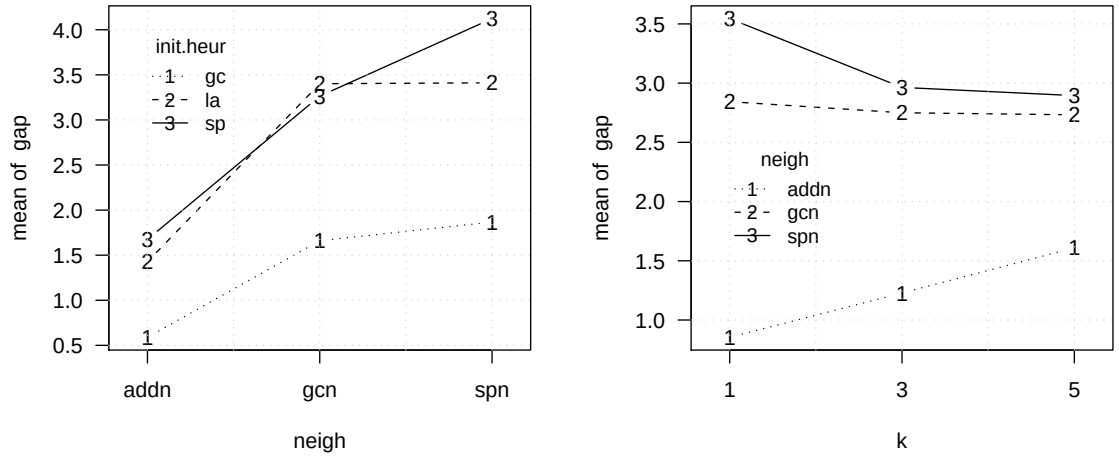

FIGURE 7: Interaction plots to visualize the impact of `neigh:init.heur` and `k:neigh` in the replicated model `fm2bR`

We now turn to the *unreplicated design*, i.e., $r = 1$.

```
> load("Data/NPOY.dataR")
> str(NPOY, strict.width = "cut", width = 70)
```

```
Soutput
'data.frame':        675 obs. of  6 variables:
 $ instance : Factor w/ 25 levels "1","10","11",..: 1 1 1 1 1 1 1 1 ..
 $ init.heur: Factor w/ 3 levels "gc","la","sp": 1 1 1 1 2 2 1 3 1 1..
 $ neigh    : Factor w/ 3 levels "addn","gcn","spn": 3 3 2 1 1 2 1 1..
 $ k        : Factor w/ 3 levels "1","3","5": 2 1 3 2 3 2 1 1 1 3 ...
 $ run      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ gap      : num  1.565 1.984 1.521 0.397 1.807 ...
```

We omit the likelihood ratio test analysis for the random effects, which is encoded in R exactly in the same way as for the replicated case and yields the same highly significant p-values.

The main point we want to make with this design pertains, instead, the fixed effects:

```
> fm2bU <- lmer(gap ~ (k + init.heur + neigh)^2 + (1 |
      instance) + (1 | instance:k) + (1 | instance:neigh) +
      (1 | instance:init.heur), data = NPOY)
> anova(fm2bU)
```

```
Soutput
Analysis of Variance Table
                Df Sum Sq Mean Sq F value
k                2   1.13    0.56    7.62
init.heur        2  10.32    5.16   69.82
neigh            2  10.80    5.40   73.11
k:init.heur      4   0.68    0.17    2.30
k:neigh          4  15.50    3.88   52.47
init.heur:neigh  4  20.07    5.02   67.92
```

Again the only p-values larger than 0.05 are those for `k` and `k:init.heur` (not shown). But the relevant observation is about the estimates of fixed effects means

```
> summary(fm2bU)@coefs[1:10, ]
Soutput
               Estimate Std. Error t value
(Intercept)      1.7642     0.1487  11.864
k1               0.0423     0.0148   2.861
k2              -0.0552     0.0148  -3.731
init.heur1      -0.7503     0.0639 -11.740
init.heur2       0.3007     0.0639   4.706
neigh1          -0.8072     0.0677 -11.929
neigh2           0.2878     0.0677   4.253
k1:init.heur1    0.0197     0.0209   0.941
k2:init.heur1   -0.0213     0.0209  -1.017
k1:init.heur2   -0.0604     0.0209  -2.889
```

The standard errors of these estimates in the unreplicated case are smaller than those in the replicated case, an observation which is consistent with Birattari (2004). For example, for `k1` we have $s[\alpha_{\texttt{k1}}] = 0.0375$ in the replicated case against $s[\alpha_{\texttt{k1}}] = 0.0148$ in the unreplicated case. As a consequence of this fact, the unreplicated case yields more powerful tests for differences between the levels of the fixed factors. Hence, when a limit on the total number of experiments is imposed, maximizing the number of tested instances should be preferred with respect to maximizing the number of replicates.

4.3 Case $\langle$-, $q(M), r\rangle$: Nested design

In this case we study the effect of instance parameters which are used to stratify the population of instances. We consider only one algorithm, as we are only interested in the instance parameters. Obviously, the conclusions on the instances will be valid only for the algorithm chosen. We have two instance factors under study:

- `type`: the type of graph with levels `{U,G,sm}`
- `weights`: the distribution of weights with levels `{w,1}`;

```
> load("Data/OPMR.dataR")
> str(OPMR, strict.width = "cut", width = 70)
Soutput
'data.frame':        150 obs. of  8 variables:
 $ weights  : Factor w/ 2 levels "1","w": 1 1 1 1 1 1 1 1 1 1 ...
 $ type     : Factor w/ 3 levels "G","U","sm": 1 1 1 1 1 1 1 1 1 1 ...
 $ algorithm: Factor w/ 1 level "gc-addn-1": 1 1 1 1 1 1 1 1 1 1 ...
 $ instance : Factor w/ 30 levels "G-800-l-11","G-800-l-12",..: 1 1 ..
 $ run      : int  1 5 2 2 4 3 5 2 1 3 ...
 $ gap      : num  3.3 3.3 5.5 5.5 3.3 ...
 $ class    : Factor w/ 6 levels "G-800-l-1","G-800-l-w",..: 1 1 1 1..
 $ inst.seed: Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 2 2 2..
```

Nesting is automatically handled appropriately in `lmer` as long as the levels of the instance factor are distinct (Bates, 2007). This might be not the case when the nesting is

implicit, that is, when the labels used for the levels of the variable at the inner state are incomplete. For example, this is the case if we identify the instances for each combination of the instance factors, type and weights, by the seeds used to generate them, say, 1, 2, 3, 4, 5. We might have 5 seeds but $3 \times 2 \times 5$ different instances. If so, then we just need to specify the seed as the random factor by `(1|type:weights:seed)` or relabel the instances as `G-800-w-1`, `G-800-w-2`, etc. and specify simply `(1|type:weights)`. Our data have both identifiers described: `instance` and `inst.seed`.

We first test the hypothesis on the nested random effects. Again we can choose between F-ratio and likelihood ratio test. The likelihood ratio test has the advantage that it does not require to recalculate the expected mean squares for the appropriate test statistic, so we try that first. In R we have

```
> fm3.1 <- lmer(gap ~ (type + weights)^2 + (1 | type:weights:inst.seed),
     data = OPMR, REML = FALSE)
> fm3.0 <- lm(gap ~ (type + weights)^2 + 1, data = OPMR)
> LRT <- as.numeric(2 * (logLik(fm3.1) - logLik(fm3.0)))
> 1 - pchisq(LRT, 1)
Soutput
[1] 1.71e-14
```

We see that we can reject $H_0 : \sigma_\tau^2 = 0$. As mentioned above, the likelihood ratio test is more conservative than the F-ratio test hence, since we reject already, there is no need to check the F-ratio test as well. We therefore include the random effect in the model.

The next step is considering the fixed factors that determine the instance classes. Again, since the experiment is balanced the p-values can be determined via the `anova` F statistics

```
> fm3 <- lmer(gap ~ (type + weights)^2 + (1 | type:weights:inst.seed),
     data = OPMR)
> fm3.aov <- anova(fm3)
> print(fm3.aov, digits = 3)
Soutput
Analysis of Variance Table
             Df Sum Sq Mean Sq F value
type          2  21.43   10.71   42.43
weights       1   0.10    0.10    0.38
type:weights  2   1.85    0.92    3.66
```

and manually derive the p-values adding the degrees of freedom of the denominator, that in this case are $(r-1)b_1b_2$, with b_1 and b_2 being the number of levels of the two instance factors

```
> type <- fm3.aov["type", ]
> 1 - pf(type$"F value", type$Df, (5 - 1) * 3 * 2)
Soutput
[1] 1.32e-08
> weights <- fm3.aov["weights", ]
> 1 - pf(weights$"F value", weights$Df, (5 - 1) * 3 * 2)
```

```
Soutput
[1] 0.544
> interaction <- fm3.aov["type:weights", ]
> 1 - pf(interaction$"F value", interaction$Df, (5 - 1) * 3 * 2)
Soutput
[1] 0.0409
```

We conclude that the type has a significant effect on the average performance of the algorithm while the weights not. If relevant to the analysis, one can proceed to consider the estimated effects of these two fixed factors. They are to be interpreted as the estimated change in the mean lower bound approximation of the algorithm caused by different characteristics of the instances.

4.4 Case $\langle N, q(M), r\rangle$: General design

In the last case, we aim at a general analysis of the influence on mean performance of local search components and different instance features. We consider a design with the following algorithm and instance factors

- `init.heur`: the construction heuristic with levels `{gc,la,sp}`;
- `neigh`: the neighborhood with levels `{addn,gcn,spn}`;
- `k`: the value of k in the neighborhoods with categorical levels `{1,3,5}`;
- `type`: the type of graphs with levels `{U,G,sm}`;
- `dens`: the edge density in the graph with levels `{l,m,h}`;
- `weights`: the distribution of weights with levels `{w,1}`.

All these factors are fixed factors. Each combination of the three instance factors gives rise to a class from which we sample 5 instances. The additional factor `instance`, or `inst.seed`, is, therefore, a random factor. The experiment has $3 \times 3 \times 2 \times 5 = 90$ experimental units (instances). Moreover we replicate each run of an algorithm 5 times leading to a total of 12150 runs over all.

```
> load("Data/NPMR.dataR")
> str(NPMR, strict.width = "cut", width = 70)
Soutput
'data.frame':        12150 obs. of  12 variables:
 $ weights  : Factor w/ 2 levels "1","w": 1 1 1 1 1 1 1 1 1 1 ...
 $ type     : Factor w/ 3 levels "G","U","sm": 1 1 1 1 1 1 1 1 1 1 ...
 $ dens     : Factor w/ 3 levels "h","l","m": 2 2 2 2 2 2 2 2 2 2 ...
 $ init.heur: Factor w/ 3 levels "gc","la","sp": 3 2 3 1 2 2 3 2 1 2..
 $ neigh    : Factor w/ 3 levels "addn","gcn","spn": 2 1 2 3 3 3 3 1..
 $ k        : Factor w/ 3 levels "1","3","5": 1 3 3 2 3 3 3 2 1 2 ...
 $ instance : Factor w/ 90 levels "G-800-h-1-1",..: 11 11 11 11 11 1..
 $ run      : int  4 4 5 2 3 2 3 3 3 3 ...
 $ gap      : num  11 3.3 3.3 5.5 6.6 ...
 $ class    : Factor w/ 18 levels "G-800-h-1","G-800-h-w",..: 3 3 3 ..
 $ algorithm: Factor w/ 27 levels "gc-addn-1","gc-addn-3",..: 22 12 ..
 $ inst.seed: Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1..
```

In a full nested factorial design of this kind we could study fixed effects interactions up to the sixth level. However, high order interactions are difficult to interpret and we, therefore, restrict ourselves to interactions of level three and to no interaction between algorithmic and instance factors.[7] Let's first test the significance of the instance factor.

```
> fm4.1 <- lmer(gap ~ (type + weights + dens)^3 + (init.heur +
      neigh + k)^3 + (1 | type:weights:dens:inst.seed),
      data = NPMR, REML = FALSE)
> fm4.0 <- lm(gap ~ (type + weights + dens)^3 + (init.heur +
      neigh + k)^3, data = NPMR)
> LRT <- as.numeric(2 * (logLik(fm4.1) - logLik(fm4.0)))
> 1 - pchisq(LRT, 1)
Soutput
[1] 0.224
```

In this case the p-value from the likelihood ratio test is not significant and it does not allow us to reject the null hypothesis that the two models are equal. However, as we mentioned the likelihood ratio test is rather conservative, hence we check also the exact F-ratio test. The terms in the F-ratio are the same as those provided in Table 2 of Section 2.3 but the derivation of the degrees of freedom require some more work. Calling `a1`, ..., `aN` and `b1`, ..., `bM` the levels of the algorithmic factors and instance factors, respectively, and using the rules for the degrees of freedom given by Montgomery (2005, pag. 502) we obtain

```
> fm4.1 <- lmer(gap ~ (type + weights + dens)^3 + (init.heur +
      neigh + k)^3 + (1 | instance), data = NPMR)
> VC <- VarCorr(fm4.1)
> sigma.tau <- as.numeric(attr(VC$instance, "stddev"))
> sigma <- as.numeric(attr(VC, "sc"))
> F.ratio <- (sigma^2 + (a1 * a2 * a3 * r) * sigma.tau^2)/sigma^2
> (df1 <- b1 * b2 * b3 * (q - 1))
Soutput
[1] 72
> (df2 <- as.numeric(fm4.1@dims["n"]) - 1 - sum(anova(fm4.1)["Df"]) -
      df1)
Soutput
[1] 12034
> 1 - pf(F.ratio, df1, df2)
Soutput
[1] 0.00469
```

The p-value is significant. Since the test is exact we give preference to this result and proceed to analyze the fixed effects using the mixed model rather than the ordinary ANOVA. In order to add the p-values to the ANOVA table we use a function written by ourselves and available from the online compendium. This function is simply a wrapper that uses the F-ratio from the `anova` method for `lmer` and the degrees of freedom derived by the rules of Montgomery (2005).

[7] Note that if high order interactions are not of interest, fractional factorial designs are a better choice than full factorial designs because they minimize the number of experiments.

```
> fm4 <- lmer(gap ~ (type + weights + dens)^3 + (init.heur +
     neigh + k)^3 + (1 | type:weights:dens:inst.seed),
     data = NPMR)
> anova.4lmer.balanced(fm4, c("type", "weights", "dens"),
     instance.id = "inst.seed")
Soutput
Analysis of Variance Table
                  Num. Def Sum Sq Mean Sq F value Den. Df Pr(>F)
type                     2  82058   41029  951.95      72 <2e-16
weights                  1  22441   22441  520.66      72 <2e-16
dens                     2   6882    3441   79.84      72 <2e-16
init.heur                2 142577   71288 1654.02   12034 <2e-16
neigh                    2 117939   58969 1368.19   12034 <2e-16
k                        2  76931   38465  892.47   12034 <2e-16
type:weights             2  90635   45318 1051.45      72 <2e-16
type:dens                4    313      78    1.82      72   0.14
weights:dens             2   9473    4736  109.89      72 <2e-16
init.heur:neigh          4  47798   11949  277.25   12034 <2e-16
init.heur:k              4  46994   11748  272.58   12034 <2e-16
neigh:k                  4  75897   18974  440.24   12034 <2e-16
type:weights:dens        4    209      52    1.21      72   0.31
init.heur:neigh:k        8  40766    5096  118.23   12034 <2e-16
```

The results indicate that all main effects are significant and that among the interactions only `type:weight:dens` and `type:dens` are not significant. The omission of effects that might be significant in the model may result in an overestimation of the denominator in the F-ratio and consequently in more conservative tests. However, these results are sufficient for us. They indicate that there is a significant effect of the nesting factors and this indicates that the analysis must be differentiated for each class.

Our final step is to split the data and to perform for each class an analysis similar to the one of case 2. In Figure 8 we report in a dotplot the average results of each algorithmic configuration on each instance class supported by confidence intervals (the function `intervals` to compute the overall plot is available online). The overall result is that the combination `gc-addn` is the one yielding the best performance, consitently over the different instance classes, while there does not seem to be a significant difference for this configuration in the value chosen for k.

5 Summary and Outlook

In this chapter, we described linear statistical models and their use in the specific task of analyzing the results of optimization algorithms. We put our emphasis on *mixed effects models* in which algorithmic components are treated as fixed factors and the test instances as random factors. We provided evidence that these models lead to different inferences with respect to ordinary ANOVA models where the instances are treated also as fixed factors. In addition, we argued that when instance factors are also subject of study then the models become nested or, alternatively, separate analyses have to be conducted.

We developed a detailed example for didactical purposes and showed how the results from the analysis of mixed models should be interpreted and how they may be

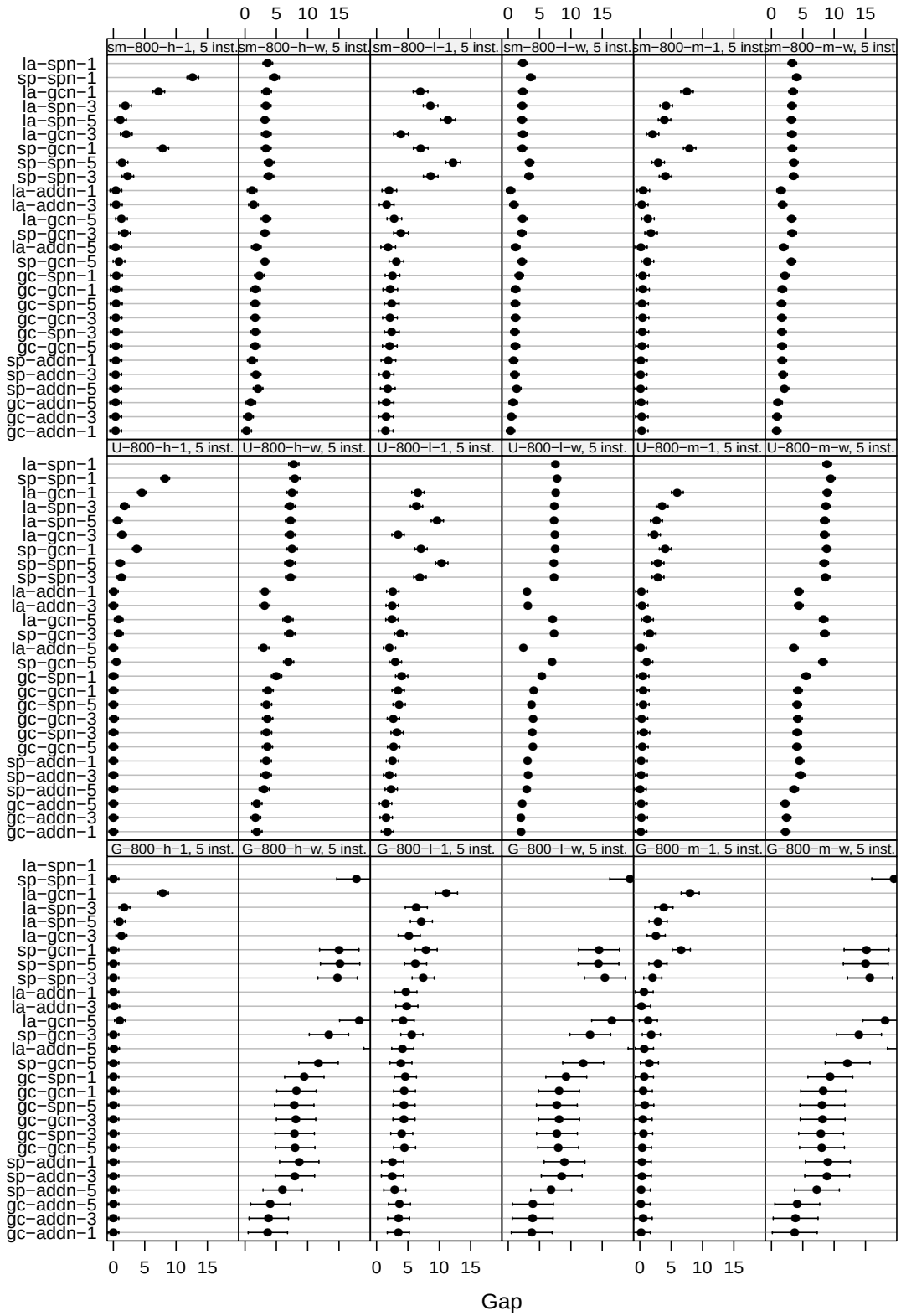

FIGURE 8: Confidence intervals derived as described in Section 2.2 for each instance class indepenedently taken. On the y-axis the labels of the algorithms indicate the composition with respect to the components described in the text. On the x-axis, the gap represents the percentage deviation from the lower bound

presented by means of tables and graphics. These might be used in articles in addition to the common practice of reporting numerical results on a few benchmark instances. The inferential analysis becomes more relevant to be reported as the amount of data decreses.

There are a number of issues that we left out and that may be included in this framework. Examples are conditional parameters for the algorithms, that can be modelled similarly to the nesting of the instances, and the permutation of instance data, a feature that might have a certain impact on the results, that can also be included yielding a design with two nested random factors and thus a hierarchical model (Fox, 2002).

There are also further developments that could be pursued. The graphical presentation of results constitutes one of the possible improvements of this work. Regression trees offer a nice and concise way to attain this. They consist of (binary) trees obtained by branching the data under analysis with branching higher in the tree for the factors responsible of the largest evidence for differences (identified by the entity of the p-value). However, all available packages of which we are aware do not include the possibility of treating random factors, nesting and neither blocking factors.

The whole chapter was based on the assumption of additive *linear* models and *normality* of data. A natural extension of this work is the use of non-linear mixed effects models and generalized linear mixed effects models that seem more appropriate in many cases of analysis of optimization algorithms. These models are often used in the study of repeated measurements over time of a certain response (longitudinal data). This could disclose a further development, that is, the analysis and comparison of optimization algorithms not only on the basis of their final response but also on the way performance changes over run time.

References

Bang-Jensen J, Chiarandini M, Morling P (2009) A computational investigation of heuristic algorithms for 2-edge-connectivity augmentation. Networks In print

Barr R, Golden B, Kelly J, Resende M, Stewart W (1995) Designing and reporting on computational experiments with heuristic methods. Journal of Heuristics 1(1):9–32, DOI 10.1007/BF02430363

Bates D (2007) Personal Communication

Bates D, Maechler M, Dai B (2008) lme4: Linear mixed-effects models using S4 classes. URL `http://lme4.r-forge.r-project.org/`, r package version 0.999375-28

Birattari M (2004) On the estimation of the expected performance of a metaheuristic on a class of instances. how many instances, how many runs? Tech. Rep. TR/IRIDIA/2004-01, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium

Bondy J, Murty U (2008) Graph Theory. Graduate Texts in Mathematics, Vol. 244, Springer London, DOI 10.1007/978-1-84628-970-5

Chiarandini M (2005) Stochastic local search methods for highly constrained combinatorial optimisation problems. PhD thesis, Computer Science Department, Darmstadt University of Technology, Darmstadt, Germany

Coffin M, Saltzman MJ (2000) Statistical analysis of computational tests of algorithms and heuristics. INFORMS Journal on Computing 12(1):24–44

Conforti M, Galluccio A, Proietti G (2004) Edge-connectivity augmentation and network matrices. In: Workshop on Graph-Theoretic Concepts in Computer Science, Springer Verlag, Berlin, Germany, Lecture Notes in Computer Science, vol 3353, pp 355–364

Cormen T, Leiserson C, Rivest R (2001) Introduction to algorithms, 2nd edn. MIT press

Czarn A, MacNish C, Vijayan K, Turlach B, Gupta R (2004) Statistical exploratory analysis of genetic algorithms. Evolutionary Computation, IEEE Transactions on 8(4):405–421, DOI 10.1109/TEVC.2004.831262

Fox J (2002) Linear mixed models. Appendix to An R and S-PLUS Companion to Applied Regression, URL `http://cran.r-project.org/doc/contrib/Fox-Companion/appendix-mixed-models.pdf`

Glover F, Kochenberger G (eds) (2002) Handbook of Metaheuristics, International Series in Operations Research & Management Science, vol 57. Kluwer Academic Publishers, Norwell, MA, USA

Hooker JN (1996) Testing heuristics: We have it all wrong. Journal of Heuristics 1:32–42

Johnson R, Wichern D (2007) Applied Multivariate Statistical Analysis, sixth edition edn. Prentice-Hall International Editions

Kutner MH, Nachtsheim CJ, Neter J, Li W (2005) Applied Linear Statistical Models, 5th edn. McGraw Hill

Lehmann E (2003) Theory of point estimation. Springer

Lehmann E, Romano J (2008) Testing statistical hypothesis. Springer

Lenth RV (2006) Java applets for power and sample size [computer software]. Retrieved 29 January 2009 from `http://www.stat.uiowa.edu/~rlenth/Power`

Lin BW, Rardin RL (1979) Controlled experimental design for statistical comparison of integer programming algorithms. Management Science 25(12):1258–1271, URL `http://www.jstor.org/stable/2630799`

McGeoch CC (1996) Toward an experimental method for algorithm simulation. INFORMS Journal on Computing 8(1):1–15, DOI 10.1287/ijoc.8.1.1, this journal issue contains also commentaries by Pierre L'Ecuyer, James B. Orlin and Douglas R. Shier, and a rejoinder by C. McGeoch

Michiels W, Aarts E, Korst J (2007) Theoretical Aspects of Local Search. Monographs in Theoretical Computer Science, An EATCS Series, Springer Berlin Heidelberg, DOI 10.1007/978-3-540-35854-1

Molenberghs G, Verbeke G (eds) (1997) Linear Mixed Models in Practice - A SAS-Oriented Approach. Verlag Springer

Molenberghs G, Verbeke G (2005) Models for Discrete Longitudinal Data. Verlag Springer

Montgomery DC (2005) Design and Analysis of Experiments, sixth edn. John Wiley & Sons

Pinheiro J, Bates D, DebRoy S, Sarkar D, the R Core team (2008) nlme: Linear and Nonlinear Mixed Effects Models. R package version 3.1-89

Pinheiro JC, Bates DM (2000) Mixed-Effects Models in S and S-Plus. Springer

R Development Core Team (2008) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, URL `http://www.R-project.org`, ISBN 3-900051-07-0

Rardin RL, Uzsoy R (2001) Experimental evaluation of heuristic optimization algorithms: A tutorial. Journal of Heuristics 7(3):261–304, DOI 10.1023/A:1011319115230

Rousseeuw PJ (1984) Least median of squares regression. Journal of the American Statistical Association 79(388):871–880, URL `http://www.jstor.org/stable/2288718`

SAS Institute Inc. (2007) SAS online documentation: Parameterization of mixed models. `http://www.webcitation.org/5h0u00trT`, retrieved on 2009-05-24

Stram D, Lee J (1994) Variance components testing in the longitudinal mixed effects model. Biometrics 50(4):1171–1177

Stram D, Lee J (1995) Correction to 'variance components testing in the longitudinal mixed effects model'. Biometrics 51(3):1196

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1):67–82

Zanakis SH (1977) Heuristic 0-1 linear programming: An experimental comparison of three methods. Management Science 24(1):91–104, URL `http://www.jstor.org/stable/2630731`

Zemel E (1981) Measuring the quality of approximate solutions to zero-one programming problems. Mathematics of operations research 6(3):319–332