# Bootstrapping to solve the limited data problem in production control: an application in batch process industries

V. Cristina Ivănescu[*], J. Will M. Bertrand[*], Jan C. Fransoo[*] and Jack P.C. Kleijnen[†]

**Abstract**

Batch process industries are characterized by complex precedence relationships among operations, which makes the estimation of an acceptable workload very difficult. Previous research indicated that a regression-based model that uses aggregate job set characteristics may be used to support order acceptance decisions. Applications of such models in real life assume that sufficient historical data on job sets and the corresponding makespans are available. In practice, however, historical data may be very limited and may not be sufficient to produce accurate regression estimates. This paper shows that such a lack of data significantly impacts the performance of regression-based order acceptance procedures. To resolve this problem, we devised a method that uses the bootstrap principle. A simulation study shows that performance improvements are obtained when using the parameters estimated from the bootstrapped data set, demonstrating that this bootstrapping procedure can indeed solve the limited data problem in production control.

*Keywords:* order acceptance, batch process industries, statistics

## Introduction

Regression models have been used to estimate job set makespans and job flowtimes in manufacturing environments. One of the key assumptions is that a sufficient amount of (historical) data is available to construct such regression models. In reality, it is unlikely that this assumption holds. We

[*]Department of Technology Management, Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

[†]Department of Information Systems and Management/ Center for Economic Research (CentER), Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands

denote this problem as the limited data problem in production control. In this paper, we solve this problem through a method based on the bootstrapping principle. We further apply this method in a batch chemical plant setting.

Statistical methods have a long tradition in the field of Operations Research (OR). Empirical studies by Ford et al. (1987) and Lane et al. (1993) indicate that statistical methods - especially regression analysis - is one of the top-three OR techniques that OR educators teach and practitioners find most useful. Regression analysis is used in forecasting and prediction problems in many areas, including manufacturing planning and control, projecting workforce requirements, and development of project costs and cash flows. For example, simple and multiple linear regression analyse are the most common approaches in the due-date assignment literature. Researchers used numerous job-related and shop-related factors as independent variables to predict the flowtime (i.e. the total throughput time of a job in the production system, which consists of processing time and waiting time) for arriving jobs, and for setting the due date accordingly (see e.g. Ragatz and Mabert , 1984; Cheng and Gupta , 1989; Vig and Dooley , 1991, 1993; Gee and Smith , 1993).

A companion of the flowtime estimation problem is the estimation of the makespan of a set of jobs (i.e., the completion time of the last job). Raaymakers et al. (2000a) identified five aggregate job set characteristics that were used - in addition to the workload of a job set - to accurately predict the job set's makespan by means of a multiple linear regression model. This regression model is used to support order acceptance decisions in batch process industries, featuring complex job and resource structures, in settings with deterministic processing times. Ivanescu et al. (2002) extended this model by considering Erlang distributed processing times.

The findings in these studies indicate that regression modelling does provide the decision makers with a powerful and relatively straightforward tool for setting due-dates and/or supporting order acceptance decisions. However, the question then arises whether this theoretical approach can be applied in practice. In all these studies, the most common methodology for obtaining the appropriate regression data is simulation. This allows a very large amount of data to be explored to determine the coefficients of the regression models. Although some authors indicate that historical information may also be analyzed, application of such models in real life situations assumes that sufficient historical data is available. Unfortunately, this assumption does not always hold: the quantity of historical data at hand may be limited, or not all the available data may be relevant. It is well known that a regression model can be used reliably if and only if the relationship between the independent variables and the dependent variable does not change. Changes in technology, raw materials, customer attitudes and needs can have a lasting impact on established relationships. If there are indications that the relationship between independent and dependent variables changes, it becomes necessary to collect a new set of data in order to re-estimate the regression coefficients. This implies that predictions must be made from a small historical base. To our knowledge, insights have not yet been developed as to what extent a limited amount of historical data affects the performance of the regression-based order acceptance models, and to what extent simulation can help.

In this paper, we address this issue for batch chemical plants, and we investigate whether the performance of a regression-based order acceptance procedure - namely the *regression policy* developed by Ivanescu et al. (2002) - is affected by limited availability of historical data. Although the problem we address is of practical interest, the research presented in this paper is simulation based; i.e. we do not use real data but simulated data, allowing us to conduct controlled experiments. We develop a simulation model that represents a hypothetical batch process industries production department, and we generate shop-specific historical data by mimicking the planner's actions. After investigating the magnitude of the limited data problem, we develop a procedure based on the bootstrap principle. The bootstrap was introduced by Efron (1979) as a computer-based method for measuring the accuracy of statistical estimates. It implies resampling - with replacement - of a given (limited) sample of i.i.d. observations. In our paper, we use resampling to generate additional data (namely job sets) with the right variety across the job sets and with similar characteristics as the observed historical data. We then investigate whether performance improvements are obtained through this bootstrapped data set.

We conclude this introduction with an outline of the paper. The next section details the problem setting. To make the paper self-contained, we briefly describe the regression policy in the third section. In the fourth section we investigate to what extent limited data affects the performance of this policy. The fifth section proposes a bootstrap method to generate additional data, while the sixth section investigates the performance of this method in a setting with dynamic order arrivals and stochastic processing times. The final section presents our conclusions.

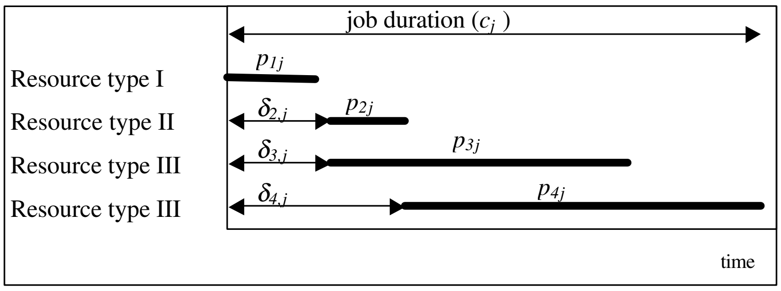
## Problem setting

The setting for this research is a real life inspired production department in the batch process industries (see Raaymakers et al., 2000b). We consider a hierarchical production control structure consisting of the following two levels: (1) a *planner* who accepts or rejects orders that are requested by the market for delivery at a specified date, and (2) a *scheduler* who allocates the processing steps of a job (the order accepted by the planner) to specific resources, and determines the exact sequencing and timing of the planned execution of the processing steps on the resources.

The planning horizon (say) $H$ is divided into $n_H$ time buckets or planning periods. A planning period $t$ $(t = 1, ..., n_H)$ always starts with an empty system; at the end of each period the system must be empty again.

In each planning period $t$, orders arrive at the planner randomly. The planner has to decide online whether to accept or reject an incoming order. We assume that each order consists of exactly one job and has a non-negotiable, known due date equal to the end of the next planning period. Each job $j$ requires $s_j$ processing steps. These processing steps have an overlap in time when two resources are needed simultaneously. In batch process industries, the product being processed is generally a fluid that needs containerization in order to be stored. If there are two consecutive processing steps, these could be decoupled by storing the product in a silo or bin - but this is often not possible due to the instability of the product. As a consequence, the product is transferred from one resource to the next (e.g., from a reactor to a distillation unit). The product will then occupy both units during some time. This overlap of processing steps is modelled as the time delay $\delta_{i,j}$ between the start time of processing step $i$ $(i = 1, 2, ..., s_j)$ relative to the start time of job $j$. An illustration of such a job is given in Figure 1.

Figure 1: Job definition



The processing time $P_{ij}$ of processing step $i$ of job $j$ is a random variable generated by a two-step sampling procedure (we use capital letters to denote random variables, and lower case letters to denote their realizations, e.g. $p_{ij}$). First, a value is generated from an uniform distribution on the interval $[a, b]$. This value is rounded up to an integer value and represents the expected processing times $(E[P_{ij}])$. Second, another value is generated by sampling from an Erlang distribution with the mean $E[P_{ij}]$ and the shape parameter $k$. The result is made integer to give the actual number of time units required for processing step $i$ of job $j$. We assume that the processing times are statistically independent of each other. Note that the shape parameter $k$ is the same for all jobs in the job set.

The jobs that arrive and are accepted in period $t$, must be executed in period $t + 1$; they form a job set - denoted by $JS(t)$ $(t = 1, 2, ..., n_H)$. Furthermore, there are no precedence relations among the jobs in the job set.

At the start of each period $t + 1$, $JS(t)$ is released to be scheduled and executed by the production system. The shop consists of five resource types, with two identical resources per type. Two reasons determined our choice of this particular resource configuration. First, ten resources seem a realistic size of a production department. Second, the size of the scheduling problem remains reasonable with respect to the computational time for the simulation experiments.

The scheduler constructs a schedule by using deterministic processing times equal to their expectation (i.e., the expected processing times). We refer to this schedule as the *ex ante schedule*, denoted by $S_{JS(t)}$. To construct this schedule, we use the simulated annealing algorithm for no-wait job shops developed by Raaymakers and Hoogeveen (2000). The resulting makespan, called the *ex ante makespan*, is denoted by $C_{\text{max}}^{\text{ex ante}}(S_{JS(t)})$.

Next, the jobs in the job set are executed by the production department according to this schedule. The actual processing times may differ from the estimated durations used to obtain the ex ante schedule. Moreover, the processing times are not known until realized; therefore, non-feasibility problems may occur during execution, so rescheduling may be needed. The rescheduling procedure used in this paper is a "right-shift" control policy (Leon et al., 1994) that entails a right-shifting of the schedule in order to restore the feasibility on the resources while maintaining the original sequence. It can be used for locally revising the schedule in real time. For details on the rescheduling procedure we refer to Ivanescu et al. (2002). One of the difficulties in no-wait job shop scheduling is that changes in the job sequence on one resource are likely to affect several other resources. Because the actual processing times are not exactly known at the time the schedule is determined, it is impossible to satisfy all the no-wait restrictions throughout the job set execution. Violations of the no-wait restrictions may cause product quality problems, therefore violations are undesirable. This is measured by the feasibility performance, defined as one minus the fraction of processing steps that violate the no-wait restrictions. At the end of the planning period $t+1$, the *ex post makespan* is obtained - denoted by $C_{max}(S_{JS(t)})$.

## Regression policy

The makespan of a job set $JS(t)$ is clearly influenced by its workload. More specifically, the workload on the bottleneck resource type imposes a lower bound (say) $LB(JS(t))$ on the makespan. This bound is a single resource lower bound on the makespan which is computed for each job set by dividing the workload on the bottleneck resource type by the number of resources of that type (see Carlier 1987). We round this number upwards, because all processing times are integer values and no pre-emption is allowed.

The minimal makespan of a feasible schedule will often exceed this lower bound because of job interactions (timing and no-wait sequencing constraints) at the scheduling level. Job interactions result from relations among capacity requirements on different resources and from scarcity of capacity. The capacity requirements for different resources have a fixed offset in time for each job, due to the fixed time delay for each processing step. To obtain a feasible schedule, some idle time on the resources is unavoidable. Job interaction is measured by the *interaction margin* defined as follows. Assuming the existence of a number of scheduled job sets (i.e. historical data), for each scheduled job set $JS(t)$ the interaction margin $I(S_{JS(t)})$ is defined as the relative difference between the realized makespan and the lower bound on the makespan:

$$I(S_{JS(t)}) = \frac{C_{\max}(S_{JS(t)}) - LB(JS(t))}{LB(JS(t))} \tag{1}$$

Ivanescu et al. (2002) estimated the relationship between the interaction margin and a number of aggregate job set characteristics. The following aggregate job set characteristics proved to significantly influence the amount of job interaction: (i) average number of processing steps per job $\mu_s$; (ii) average overlap of processing steps $\mu_g$; (iii) squared coefficient of variation of the expected processing time $cv^2_{E[p]}$; (iv) workload balance over the resource types $\rho_{\max}$; (v) number of jobs in the job set $n_{jobs}$; (vi) squared coefficient of variation of the actual processing time $cv^2_p$. For detailed definitions of these characteristics we refer to Ivanescu et al. (2002).

By using these six characteristics as regressors, Ivanescu et al. (2002) developed a multiple linear regression model to predict the mean interaction margin defined in (1). In this paper, however, we aim at a job set service level (defined as the probability of on-time job set completion) equal to $\alpha$, so estimating only the mean interaction margin is not sufficient. Therefore, we regress the $(100 \cdot \alpha)$th quantile of the distribution of $I(S_{JS(t)})$ of a given job set $JS(t)$. To estimate this quantile - denoted by $I^\alpha(S_{JS(t)})$ - we use the empirical distribution of $I(S_{JS(t)})$ and the quantile estimator implemented in the SPSS statistical package:

$$I^\alpha(S_{JS(t)}) = I_{(\lceil n \cdot \alpha \rceil)}(S_{JS(t)}) \tag{2}$$

where $\lceil x \rceil$ denotes the smallest integer that is greater than or equal to $x$ and $I_{(1)}(S_{JS(t)}) \leq I_{(2)}(S_{JS(t)}) \leq ... \leq I_{(n)}(S_{JS(t)})$ are the order statistics obtained by sorting the observations $\{I_{(i)}(S_{JS(t)}) : i = 1,..,n\}$ in ascending order.

Then, the regression model has the following form:

$$\widehat{I^\alpha}(S_{JS(t)}) = \hat{\beta}_0 + \hat{\beta}_1 \cdot \mu_s + \hat{\beta}_2 \cdot \mu_g + \hat{\beta}_3 \cdot cv^2_{E[p]} + \hat{\beta}_4 \cdot \rho_{\max} + \hat{\beta}_5 \cdot n_{jobs} + \hat{\beta}_6 \cdot cv^2_p \tag{3}$$

where $\hat{\beta}_0, \hat{\beta}_1, ..., \hat{\beta}_6$ are the ordinary least squares (OLS) estimates of the first-order effects of the six characteristics listed above.

Next, this estimate $\widehat{I^\alpha}(S_{JS(t)})$ is used to determine the $\alpha$-reliable completion time estimate of a given job set $JS(t)$:

$$\widehat{C}^{\alpha}_{\max}(JS(t)) = (1 + \widehat{I^{\alpha}}(S_{JS(t)})) \cdot LB(JS(t)) \tag{4}$$

Finally, under the regression policy, the planner accepts orders as long as the $\alpha$-reliable completion time estimate for the resulting job set is shorter than the period length:

$$\widehat{C}^{\alpha}_{\max}(JS(t)) \leq T \tag{5}$$

## The effect of limited data

The regression policy described in the previous section requires the estimation of the regression coefficients. To determine these estimates, shop-specific historical data are required. As we mentioned in the introduction, in real life we may have only limited data. Therefore, we now investigate the sensitivity of the order acceptance model is to the size of the database. We do so by means of simulation; i.e., we generate shop-specific historical data by using the production department described in the second section.

The simulation experiments are carried out in two stages: (1) data collection and estimation of the policy's parameters, and (2) performance evaluation of the order acceptance policy in a setting with dynamic order arrivals. These stages are addressed in the following sub-sections.

### Data collection

We generate a sufficiently large "historical" database of the production department described in the second section, by means of simulation. This database contains information on the customer orders and the production system for a large number of planning periods. We refer to this database as the *construction data set*.

We made the following additional assumptions. In each planning period, orders arrive according to a Poisson arrival process. The arrival rate is controlled by the demand/capacity ratio. We assume that the average demand requirements for capacity are equal to the total available capacity per planning period, which represents a situation where demand effectively exceeds available capacity. The arriving orders are generated with a high job mix variety (i.e. $s_j \sim U(1, 10)$ and $E[P_{ij}] \sim U(1, 49)$) and high uncertainty in the processing times (i.e. $P_{ij} \sim$ Erlang-2). We further assume a job set service level target of 0.95 ($\alpha = 0.95$) and that the shop has a balanced workload.

The length of a planning period is chosen such that the job set consists of a realistic number of jobs. The empirical study of Raaymakers et al. (2000b) showed that a job set of 40 to 50 jobs is realistic for this type of industrial process. To realize such job sets, we fix the length of the planning period at 1300 time units (as additional experiments indicate). The absolute length of the planning period depends on the average processing time per job, which is chosen arbitrarily in the simulation experiments.

Given that the construction data set contains the jobs accepted in a certain planning period, there must exist a mechanism to accept or reject arriving jobs. We use the following workload-based rule: orders are accepted as long as (i) the total workload does not exceed a specified maximum workload, and (ii) the workload per resource type does not exceed the available capacity per resource type.

The size of the construction data set is given by the number of planning periods, namely $n_H$. We expect the size of the construction data set to affect the performance of the regression policy. To investigate this effect, we consider six construction data sets with sizes ranging from 12 to 175 planning periods, namely $n_H \in \{12, 25, 50, 100, 150, 175\}$. A construction data set of size 12 corresponds with one year production, 25 with about two years, etc. These construction data sets are obtained by randomly sampling from the large "historical" database we generated (see the first paragraph of this subsection).

Each job set $JS(t)$ in each of the construction data sets is scheduled and executed by the production department (see Section ). All the processing times in $JS(t)$ are stochastic (namely, Erlang distributed with the same shape parameter $k = 2$). A single simulated execution of the job set would not be sufficient to properly capture the effect of stochastic processing times when developing the regression models. Therefore, we repeat the execution of each job set several times. Additional experiments showed that 250 replications were necessary to reduce the variability in the results. Therefore, we performed 250 independent realizations of the processing times, which yields i.i.d. observations $\{C_{max,i}(S_{JS(t)});\ i = 1, ..., 250\}$ for the ex post makespan for each job set.

## Regression parameters estimation

Each of the six construction data sets are now used to determine parameter estimates for the regression policy. Note that the sixth regressor in (3), $cv_p^2$, is constant in our experiments since the processing times $P_{ij}$ in each job set from each construction data set are identically distributed, namely Erlang-2. Therefore, we consider only the first five characteristics. In the following we detail our regression models. We used the SPSS software package for our regression analyses.

For developing the regression model, we use a two-step process that can be found in most statistics textbooks (e.g. Montgomery and Peck (1992)): (i) model building and (ii) model evaluation. In the model building step, a regression equation is determined that provides the best fit to the data in the construction data set. At the model evaluation step we distinguish between model adequacy checking and model validation. Model adequacy checking use residual analysis to investigate the fit of the regression model to the construction data set. However, there is no assurance that the equation with the best fit will be a successful predictor. Therefore, the predictive performance of the model has to be tested. For this purpose, new data are considered that were not used in the model building step, called the *testing data set*.

The estimation models are generated by means of multiple linear regression techniques. We denote by $Regr_{n_H}$ the regression policy that uses the parameters estimated from a construction data set of size $n_H$. To measure the fit of the regression equations to the data in the construction data set we computed the adjusted coefficient of multiple determination (adj. $R^2$), and the residual standard error ($RSE$, also called the standard error of the estimate). Whereas the adj. $R^2$ is a measure of relative fit and indicates how much of the total variation in the dependent variable is explained by the estimated regression equation, the standard error of estimate is an absolute measure of the fit because its value depends on the scale of the response variable. It specifies the amount of error incurred when the least-squares regression equation is used to predict values of the dependent variable. Therefore, the smaller the $RSE$ value, the better the prediction is.

We check the regression models for multicollinearity, because a high degree of multicollinearity gives high variance estimates. Multicollinearity exists whenever a regressor variable is highly correlated with one or more of the other regressor variables; it can be detected by using *variance inflation factors* (VIF's)(see Montgomery and Peck (1992)). A rule of thumb is that VIF's larger than 10 imply serious multicollinearity. For all the models, the VIF value for the $n_{jobs}$ variable was larger than 10; therefore, we eliminate this variable from the list of regressors.

Next, we perform a residual analysis to test the adequacy of the regression models. The standardized residuals versus the predicted values plots we examined indicate no violation of the basic regression assumptions.

Table 1 gives the name of the models, the adjusted coefficient of multiple determination (adj. $R^2$), residual standard error ($RSE$) and the regression parameters' least squares estimates.

Table 1: Regression models

| Model | adj. $R^2$ | $RSE$ | Regressors | | | | |
|---|---|---|---|---|---|---|---|
| | | | Intercept | $\mu_s$ | $\mu_g$ | $cv^2_{E[p]}$ | $\rho_{\max}$ |
| $Regr_{12}$ | 0.85 | 0.044 | -0.277 | 0.010 | -0.766 | -1.637 | 3.209 |
| $Regr_{25}$ | 0.82 | 0.065 | -1.707 | 0.121 | -0.025 | 0.842 | 2.722 |
| $Regr_{50}$ | 0.79 | 0.062 | -0.896 | 0.117 | 0.110 | -0.418 | 2.209 |
| $Regr_{100}$ | 0.85 | 0.061 | -1.575 | 0.136 | 0.287 | 0.407 | 2.443 |
| $Regr_{150}$ | 0.79 | 0.068 | -1.469 | 0.116 | 0.177 | 0.406 | 2.515 |
| $Regr_{175}$ | 0.80 | 0.069 | -1.278 | 0.133 | 0.347 | -0.269 | 2.340 |

The relatively high adj. $R^2$ values and low $RSE$ values indicate that the models we developed can be used to obtain accurate predictions. The predictive performance of the models is further evaluated on a testing data set. This data set contains 100 new job sets that were generated independently from the job sets in the construction data set. The quality of the estimation models was evaluated using the mean prediction error ($ME$) and the square root of the mean square prediction error ($\sqrt{MSE}$). Additionally, the percentage of variability in the new data explained by the model ($R^2_{pred}$) is compared with the adj. $R^2$ of the building model. The results are presented in Table 2.

Table 2: Predictive quality of the regression models in the testing data set

| *Model* | $ME$ | $\sqrt{MSE}$ | $R^2_{pred}$ |
|---|---|---|---|
| $Regr_{12}$ | 0.0061 | 0.1199 | 0.35 |
| $Regr_{25}$ | 0.0262 | 0.0910 | 0.62 |
| $Regr_{50}$ | 0.0242 | 0.0863 | 0.66 |
| $Regr_{100}$ | 0.0253 | 0.0835 | 0.68 |
| $Regr_{150}$ | 0.0244 | 0.0839 | 0.68 |
| $Regr_{175}$ | 0.0251 | 0.0839 | 0.68 |

The mean prediction error is nearly zero for all models, so the regression models seem to produce unbiased predictions. Comparing Tables 1 and 2, we observe that $RSE$ is smaller than $\sqrt{MSE}$, for all models. Furthermore, the percentage of variability in the new data explained by the models is less than the adj. $R^2$ in the construction phase. This indicates that the regression models do not predict new data as well as they fit the existing (historical) data. However, except for models $Regr_{12}$ and $Regr_{25}$, the degradation of performance is not severe. We conclude that the regression models we developed are likely to be successful predictors, except for the models $Regr_{12}$ and $Regr_{25}$. We expect a poorer performance of these two models when used to support customer order acceptance decisions.

## Performance evaluation

In this section, we perform experiments to examine if the performance of the regression policy is indeed affected by the size of the construction data set. The estimated parameters are now frozen and used by the regression policy. We repeat the experimental setting of the data collection phase. This time, however, the arriving orders are accepted/rejected according to each of the six regression policies. We simulate a planning horizon of one year, i.e. 12 replications of a planning period.

Given that the parameters were determined under a 95% job set service level target, when we investigate the effect of the construction data set size we are interested in the ability of effectively meeting this target. This may be quantified through the average percentage of job sets completed on time (POT) measure. The results are presented in Table 3.

Table 3: The average and 95% confidence interval for the actual % of job sets on-time

| Policy | POT | 95% conf. interval |
|---|---|---|
| $Regr_{12}$ | 84.93 | [77.07, 92.80] |
| $Regr_{25}$ | 88.27 | [83.89, 92.65] |
| $Regr_{50}$ | 88.37 | [83.52, 93.22] |
| $Regr_{100}$ | 88.43 | [84.60, 92.27] |
| $Regr_{150}$ | 88.67 | [84.90, 92.43] |
| $Regr_{175}$ | 89.63 | [84.99, 94.28] |

Table 3 shows that none of the proposed policies reaches the pre-specified target. However, the larger the construction data set, the better the performance (i.e. the average *POT* is closer to the 95% target and the confidence interval is tighter). We observe that the model developed in the case of limited data ($n_H = 12$) realizes the poorest performance, as expected. Such a small sample size is likely to be encountered in practice, because relevant historical information about the accepted orders may be available only over a horizon of one year or less (which means 12 observations or less).

## Limited data problem: bootstrap solution

In the previous section we saw that the size of the construction data set affects the performance of the regression policy. Therefore, in a situation with limited historical data (e.g., $n_H \leq 12$), application of the regression policy may be jeopardized. What is required is a large number of job sets with the right variety across the jobs and similar characteristics as the observed historical data. We propose a method based on the bootstrap principle to generate these additional data.

The bootstrap principle consists of repeated random re-sampling of the original observations with replacement, which is performed by a computer (Efron and Tibshirani , 1993). Bootstrapping is an approach to statistical inference that makes few assumptions about the underlying probability distribution that describes the data. This approach assumes that the empirical cumulative distribution function is a reasonable estimate of the unknown, population cumulative distribution function (in other words, the empirical density function approximates the population density function). Using the data as an approximation to the population density function, data is re-sampled with replacement from the observed sample to create an empirical sampling distribution for the statistic under consideration.

In this paper, however, we use the bootstrap principle not for statistical inference, but for generating additional job sets, denoted *bootstrap job sets.*

Bootstrap assumes that the observed data is a good estimate of the population density function. Following this principle, we assume that the sample consisting of the jobs from all the accepted job sets is a good estimate of the population consisting of the jobs that are accepted. Therefore, we re-sample with replacement from this observed sample and we generate a number of $B$ additional job sets. The bootstrap principle assure us that this set of $B$ bootstrap job sets is a proxy for a set of $B$ independent real job sets.

Let $\{j_i : i = 1, ..., n\}$ denote the sample of all accepted jobs, where $n = \sum_{t=1}^{n_H} n_{JS(t)}$ and $n_{JS(t)} = |JS(t)|$ denotes the number of jobs in a job set $JS(t)$ $(t = 1, ..., n_H)$. Since we consider independent planning periods, and assume that each job arriving into the system may be different and there are no precedence relations among jobs, the job arriving stream across all the planning periods is a sample with independent observations. However, given that a workload-based order acceptance procedure is used to accept/reject the arriving jobs, the accepted jobs contained in each job set may not be a random sample from the arriving job stream. Nevertheless, we do not expect this characteristic to violate the independence assumption. The proposed procedure is as follows.

1. Arrival moments are generated, according to a Poisson arrival process.

2. At each arrival moment, we randomly select a job - with replacement - from the observed sample $\{j_1, ..., j_n\}$ of accepted jobs. A decision to either accept or reject the job has to be made. This decision is based on the same acceptance procedure that was used to accept the jobs in the original historical data set (i.e., a workload based procedure). The result is a new job set denoted by $JS^*$. This job set is next scheduled and we denote its corresponding ex ante makespan by $C_{\max}^{exante}(S_{JS^*})$.

3. We simulate the execution of the bootstrap job set $JS^*$. As in the data collection section, 250 independent realizations of the processing times are made that yield i.i.d. observations $\{C_{max,i}(S_{JS^*}); i = 1, ..., 250\}$ for the ex post makespan.

4. We repeat steps 1, 2 and 3 $B$ times; we take $B = 500$. The replicate $b$, $b = 1, ..., B$ gives the bootstrap job set $JS_b^*$.

This procedure generates $B$ bootstrap job sets. These job sets, together with the original job sets forming the historical data set, form the new construction data set. Based on this construction data set, we estimate the parameters of the regression policy. We hypothesize that the regression policy that uses parameter estimates based on this bootstrapped construction data set performs better. We test this hypothesis in the next section.

## Evaluating the bootstrap solution

In this section we investigate to what extend a regression model estimated from a bootstrapped construction data set improves system performance when used to support order acceptance decisions as compared to a regression model induced on a small size construction data set. Let the smallest construction data set (size 12) in the fourth section be the original historical data set. Applying the bootstrap procedure described in the previous section, we generate $B = 500$ additional job sets. Adding these job sets to the initial construction data set gives the bootstrapped construction data set, with 512 job sets. Applying the procedure of Section , we compute the parameter estimates for the regression policy. We obtain the following regression model:

$$\widehat{I^{0.95}}(JS) = -1.480 + 0.125 \cdot \mu_s + 0.582 \cdot \mu_g - 0.043 \cdot cv^2_{E[p]} + 2.434 \cdot \rho_{max} \quad (6)$$

Because our "real" historical data set is simulated, we can also create an ideal situation with a construction data set with 512 independent job sets. The regression model estimated from this data set is:

$$\widehat{I^{0.95}}(JS) = -1.425 + 0.130 \cdot \mu_s + 0.337 \cdot \mu_g + 0.087 \cdot cv^2_{E[p]} + 2.402 \cdot \rho_{max} \quad (7)$$

The adj. $R^2$ ($RSE$) is 0.76 (0.076 ) for (6) and 0.79 (0.068) for (7). Comparing these values with the values in Table 1, we observe a smaller adj. $R^2$ and a higher $RSE$ for the regression model given by (6). However, the decrease in performance is small and gives us no reason for concern with respect to the predictive performance of this model.

We denote the regression policy that uses estimates based on the bootstrapped construction data set by $Regr_b$. We compare the performance of the three policies - namely $Regr_b$, $Regr_{12}$ and $Regr_{512}$ - by means of simulation experiments - using the same experimental setting as in the fourth section.

We use the following performance measures: ($i$) realized capacity utilization ($RCU$), ($ii$) percentage of job sets on time ($POT$), ($iii$) job set tardiness ($JST$), and (iv) feasibility ($FEB$). The realized capacity utilization per period is measured by

$$RCU = \frac{\sum_{j=1}^{n_J} \sum_{i=1}^{s_j} \theta_{p_{ij}}}{N \cdot T} \quad (8)$$

where

$$\theta_{p_{ij}} = \begin{cases} p_{ij} & \text{if } st_{ij} \leq T \text{ and } c_{ij} \leq T \\ p_{ij} - (c_{ij} - T) & \text{if } st_{ij} \leq T \text{ and } c_{ij} > T \\ 0 & \text{if } st_{ij} > T \end{cases} \quad (9)$$

where $p_{ij}$ denotes the processing time, $st_{ij}$ the start time, and $c_{ij}$ the completion time of the processing step $i$ of the job $j$.

Job set tardiness occurs when the ex post makespan ($C_{\max}(S_{JS})$) of the job set $JS$ exceeds its due-date (T):

$$JST = (C_{\max}(S_{JS}) - T)^+ \tag{10}$$

The feasibility performance is one minus the fraction of processing steps that violate the no-wait restrictions throughout the entire planning period. Unavoidable non-feasibility problems will occur due to the no-wait restrictions between the processing steps of a job, and because the actual processing times differ from the expected processing times. Obviously, a smaller number of violations of the no-wait constraints is preferred.

Table 4 reports these performance measures averaged over the 12 planning periods.

Table 4: Computational results

| Policy | *RCU* | *POT* | *JST* | *FEB* |
|---|---|---|---|---|
| $Regr_b$ | 0.3357 | 92.67 | 3.84 | 0.805 |
| $Regr_{12}$ | 0.3432 | 84.93 | 9.57 | 0.802 |
| $Regr_{512}$ | 0.3439 | 88.70 | 6.28 | 0.804 |

To compare the performance of $Regr_b$ with the performance of $Regr_{12}$ and $Regr_{512}$, we use paired $t$-test. That is, for each performance measure, we make two pairwise comparisons. The individual test level is set at 95%. The results showed that for all performance measures, except $FEB$, there are significant differences between $Regr_b$ and both $Regr_{512}$ and $Regr_{12}$. Table 4 reveals that with respect to the ability of meeting customer requirements, the $Regr_b$ performs best: this policy manages to determine job sets that result in a delivery performance very close to the 95% target, and a much smaller job set tardiness. Furthermore, $Regr_b$ realizes lower $RCU$ than either $Regr_{12}$ or $Regr_{512}$. However, this difference is very small. We conclude that, in case of limited historical data, our re-sampling procedure gives a similar or even improved system performance compared with a situation with a very large historical data set.

## Summary and conclusions

Regression models show good performance when used to support customer order acceptance decisions in complex plants with dynamic order arrivals and either deterministic or stochastic processing times (e.g. Raaymakers et al., 2000a; Ivanescu et al., 2002). These models have been tested extensively through simulation. These simulations, however, used a large variety of shops and job sets to estimate the coefficients of the regression models. Applications of such models in real life assume that sufficient historical data (regarding customer orders and production system) is available for estimating these coefficients with acceptable accuracy. In practice, however, relevant historical data may be limited. In this study, we investigated to what extent this limited data problem in production control impacts the performance of a regression-based acceptance procedure.

For batch process industries, featuring complex job and resource structures, we first generated shop-specific historical data by simulation. This initial simulation study shows that a regression policy performs less well if the size of the construction data set is small. To overcome the limited data problem, we developed a procedure based on the bootstrap principle. In general, bootstrap randomly re-samples the original observations with replacement. Our procedure bootstraps the original set of accepted jobs, to generate additional job sets. We assessed the performance of our bootstrap procedure by means of simulation. The results showed that the performance of the bootstrap regression policy clearly improves. Moreover, we found that this policy reaches performance levels similar to those of a regression policy that uses a very large historical data to estimate its parameters.

The results clearly demonstrate the power of extending the bootstrap principle by applying this to the basic elements of a data set in production control, namely the individual jobs. Rather than re-sampling the job sets, we used the individual jobs for re-sampling. This allowed us to construct new jobs sets. We believe that this principle can be further extended to other production control environments where the limited data problem may occur, and encourage further research to investigate the possible impact and limitations beyond the boundaries of our study.

## References

Ford FN, Bradbard DA, Ledbetter WN, and Cox JF (1987). Use of operations research in production management. *Production and Inventory Management Journal* 3rd Qtr.: 59–62.

Lane MS, Mansour AH, and Harpell JL (1993). Operations research techniques: a longitudinal update 1973-1988. *Interfaces* **23**: 63–68.

Ragatz GL and Mabert VA (1984). A simulation analysis of due date assignment rules. *J Oper Manag* **5**(1): 27–39.

Cheng TCE and Gupta MC (1989). Survey of scheduling research involving due date determination decisions. *Eur J Oper Res* **38**: 156–166.

Vig MM and Dooley KJ (1991). Dynamic rules for due date assignment. *Int J Prod Res* 29(**7**): 1361–1377.

Vig MM and Dooley KJ (1993). Mixing static and dynamic estimates for due date assignment. *J Oper Manag* **11**: 67–79.

Gee ES and Smith CH (1993). Selecting allowance policies for improved job shop performance. *Int J Prod Res* **31**(8): 1839–1852.

Raaymakers WHM, Bertrand JWM, and Fransoo JC (2000a). Using aggregate estimation models for order acceptance in a descentralized production control for batch chemical manufacturing. *IIE Trans* **32**: 989–998.

Ivanescu CV, Fransoo JC, and Bertrand JWM (2002). Makespan estimation and order acceptance in batch process industries when processing times are uncertain. *OR Spectrum* **24**: 467–495.

Efron B (1979). Bootstrap methods: Another look at the jackknife. *Ann Stat* **7**: 1–27.

Raaymakers WHM, Bertrand JWM, and Fransoo JC (2000b). The performance of workload rules for order acceptance in batch chemical manufacturing. *J Intell Manuf* **11**: 217–228.

Raaymakers WHM and Hoogeveen JA (2000). Scheduling no-wait job shops by simulated annealing. *Eur J Oper Res* **126**: 131–151.

Leon VJ, Wu SD, and Storer RH (1994). Robustness measures and robust scheduling for job shops. *IIE Trans* **26**: 32–43.

Carlier J (1987). Scheduling jobs with release dates and tails on identical machines to minimize the makespan. *Eur J Oper Res* **29**: 298–306.

Montgomery DC and Peck EA (1992). *Introduction to linear regression analysis.* Wiley: New York.

Efron B and Tibshirani RJ (1993). *An introduction to the bootstrap.* Chapman & Hall: New York.