A Novel Approach to Quantifying the Influence of Software Process on Project Performance

Jia-kuan Ma[1,2], Xiao-fan Tong[1,2], Ya-sha Wang[1,2] and Gang Li[3,4]

[1] Key Laboratory of High Confidence Software Technologies, Ministry of Education
Beijing, China
[2] Software Institute, School of Electronics Engineering and Computer Science
Peking University, Beijing, China

[3] Shandong Computer Science Center, Jinan, China
[4] Shandong Provincial Key Laboratory of Computer Network, Jinan, China

Abstract. Determining the appropriate process to be used is a key ingredient of project management. To this end, understanding the influence of activities on the project performance can facilitate the project management. However, quantifying such a relationship via traditional Multiple Linear Regression method tends to be challenging, for the amount of independent variables (activities in software process) is usually larger than the size of dataset. Aiming at such a problem, in this paper we propose a novel approach. By combing the Dantzig selector and Ordinary Least Squares (OLS) regression method, our approach can derive the regression model in such challenging situations, which further set the theoretical stage for studying the quantitive influences of software process on project performance.

1 Introduction

Inherent in every software project there is a process (whether known or unknown, whether good or bad, and whether stable or erratic) [1]. Nowadays, it has been widely accepted that for a given software project, the employed process can have significant influence on project performance (e.g. schedule, budget, quality of deliverables) in general [2-3]. Specifically, [4] found evidence, in a sample of 61 organizations, that higher CMM process maturity is associated with better project performance.

Therefore, determining the appropriate process to be used is a key ingredient of project management. However, the conclusion that software process can influence project performance does not tell us details of this relationship. For instance:

- Among various activities in a given process, are certain activities more likely to influence the final project performance than other activities? The answer to this question can facilitate the project manager to cast more focus on the more crucial activities.
- Which activities can have significant positive impact on the final project performance? Which activities can have significant negative impact on the final project performance? Answers to these questions can facilitate the project manager to better

Ma J., Tong X., Wang Y. and Li G..
A Novel Approach to Quantifying the Influence of Software Process on Project Performance.
DOI: 10.5220/0003584000440050
In *Proceeding of the 1st International Workshop on Evidential Assessment of Software Technologies* (EAST-2011), pages 44-50
ISBN: 978-989-8425-58-4
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

customize the applied process. That is, tries to include positive-impact activities while exclude negative-impact activities.

Theoretically, answering these questions calls for establishing quantitive relation-ship between software process and project performance. In particular, taking each activity in a given process as an independent variable, project performance as the dependent variable, we need to quantify the influence of independent variables on a dependent variable.

In this light, the most commonly employed approach is Multiple Linear Regression model, which can be expressed as follow:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \ldots + \beta_p X_{pi} + \varepsilon_i \qquad i=1,2,\ldots,n$$

where

Y_i = the *i*th observation of the dependent variable *Y*. In our context, Y_i denotes the project performance of a given project (labeled i) in the dataset.

X_{ji} = the *i*th observation of the independent variable X_j $(j = 1,2,\ldots,p)$. In our context, X_{ji} denotes an activity (labeled j) in the process of a given project (labeled i) in the dataset.

β_0 = the intercept of the equation; ε_i = the error term.

$\beta_1, \beta_2, \ldots, \beta_p$ = the slope coefficients for each of the independent variables. In our context, β_p denotes the correlation between an activity X_p and the final project performance *Y.* Such a correlation is formulated from the dataset.

n = the number of observations. In our context, n denotes the number of projects we collected, namely size of the dataset.

In general, Multiple Linear Regression estimates $\beta_0, \ldots, \beta_p$ through the Ordinary Least Squares (OLS) criteria, which minimizes the sum of squared residuals:

$$\min \sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1 X_{1i} - \beta_2 X_{2i} - \ldots - \beta_p X_{pi})^2$$

When the number of observations (n) is larger than the number of independent variables (p), namely n > p, the criteria above is equivalent to the p+1 first order conditions. Solving such equations can draw the estimations of β_0 to β_p, and therefore formulate the Multiple Linear Regression model.

Unfortunately, this is commonly not the case when conducting such studies in practice. On one hand, a software process usually embodies a host of activities. For instance, the international standard IEEE Std 12207:2008 [5], which serves as a major process framework, contains 123 activities in total. As a result, when taking activities in a software process as independent variables, the amount of independent variables (p) is usually large. On the other hand, the fact that collecting software process and project related data is costly and challenging [6] limits the size of feasible dataset. Consequently, the number of observations is usually relatively smaller compared to the amount of activities. Under such a condition, the OLS criteria become no longer applicable.

Aiming at such a problem, we propose in this paper a new approach to formulate the quantitive relationship. Our approach first performs variable selection based on the Dantzig selector. Then, with the n > p criteria satisfied, we iteratively apply the OLS regression following the Backward Elimination method. In this way, the quantitive relationship between software process and project performance can be derived from a relatively small dataset. Meanwhile, the underlying rigorous mathematical properties of our approach guarantee the accuracy and reliability of the result.

The rest of this paper is organized as follows. Section 2 presents a brief introduction to the Dantzig selector. Section 3 proposes our novel approach of combing the Dantzig selector with traditional Multiple Linear Regression. Finally, we conclude and discuss future work in Section 4.

2 The Dantzig Selector

In many statistical applications, the number of independent variables p is larger than the number of observations n. Suppose we have the following linear regression model:

$$y = X\beta + \varepsilon \tag{1}$$

where $\beta = (\beta_1, \beta_2, ..., \beta_p)^T \in R^p$ is the associated regression coefficients, and ε_i's are i.d.d $N(0, \sigma^2)$. X is a data matrix with possibly fewer rows than columns, i.e. $n < p$. $X = (x^1, x^2, ... , x^n)$, where $x^i = (x_{i1}, x_{i2}, ... x_{ip})^T$, $i = 1, 2, ..., n$, are predictor variables. Besides, $y = (y_1, y_2, ..., y_p)^T \in R^p$ is a vector of observation.

When $n < p$, OLS criterion cannot provide a unique performance. To deal with this challenge, Candes and Tao recently proposed a new approach, namely the Dantzig selector [7], which can generate a sparse estimate of β. The sparse nature means many coefficients in the result are exactly 0. In this sense, the Dantzig selector provides a reliable method for variable selection.

Specifically, the Dantzig selector is solution to the following l_1-regularization problem (2).

$$\min_{\tilde{\beta}\in R^p} || \tilde{\beta} ||_{\ell 1} \text{ , s.t. } ||X^* r||_{\ell\infty} \le (1 + t^{-1})\sqrt{2\log p} \cdot \sigma \tag{2}$$

where r is the residual vector $y - X\tilde{\beta}$ and t is a positive scalar. Candes and Tao indicates that if X obeys a uniform uncertainty principle (with unit-normed columns) and if the true parameter vector is sufficiently sparse (which here roughly guarantees that the model is identifiable), then we can get the following result (3) with very large probability.

$$|| \tilde{\beta} - \beta ||_{\ell 2} \le C^2 \cdot 2\log p \cdot (\sigma^2 + \sum_i \min(\beta_i^2, \sigma^2)\,) \tag{3}$$

To further estimate β with noisy data ε, for some $\lambda_p > 0$, consider solving the following convex program,

$$\min_{\tilde{\beta}\in R^p} || \tilde{\beta} ||_{\ell 1} \text{ , s.t. } ||X^* \boldsymbol{r}||_{\ell\infty} := \sup_{i \le i \le p} |(X^* \boldsymbol{r})_i| \le \lambda_p \cdot \sigma \tag{4}$$

where $r = y - X\tilde{\beta}$. In other words, the estimator $\tilde{\beta}$ is with minimum complexity (as measured by the l_1-norm) among all objects that are consistent with the data. The estimator (4) is called the Dantzig selector.

Since (4) is convex, it can easily be recast as a linear program (5),

$$\min \sum_i u_i \text{, s.t.} -u \le \tilde{\beta} \le u \text{ and } -\lambda_p \sigma 1 \le X^*(\boldsymbol{y} - X\tilde{\beta}) \le \lambda_p \sigma 1 \tag{5}$$

where the optimization variables are u, $\tilde{\beta} \in R^p$ and σ1 is a p-dimensional vector of ones. Accordingly, this estimation procedure is computationally tractable. Candes and Tao proved that the Dantzig selector is surprisingly accurate at the same time.

Ever since its birth, the Dantzig selector algorithm has drawn enormous attentions. There have been much useful research work, such as a generalized Dantzig selector [8], DASSO [9] method used to solve the Dantzig selector. Analyzing data sets with the sample size n smaller than the number of variables p, such challenges arise in many other fields ranging from health sciences to economics. For instance, in disease classification using microarray gene expression data [10], the number of arrays is usually in order of tens but the total amount of gene expression profiles is often tens of thousands. The Dantzig selector has been applied respectively as an effective solution to various specific problems.

3 Our Approach

Our approach consists of two steps. First, we leverage the Dantzig Selector to select the most correlative dependent variables from the original massive candidates. After the selection, the amount of picked variables will drop below the number of samples. Then, we apply the OLS regression iteratively to derive the ultimate regression model, as indicated by the Backward Elimination method.

3.1 Variable Selection via the Dantzig Selector

As a variable selection method, the Dantzig Selector itself does not designate the number of variables to pick out. In regard of accuracy and reliability of the selection result, we use a statistical method called cross-validation to discover the best amount of variables to pick out. It has been shown in [11] that the cross-validated choice of the penalty parameter is consistent for model selection in general conditions.

Specifically, we use fivefold cross-validation to make the decision. The details of the cross-validation procedure are listed as follows.

(a) Standardize the data; denote the full sample set by T; divide it randomly and equally into 5 parts, then get subsets T^v, v=1, 2, 3, 4, 5.

(b) Define the fivefold cross-validation training set as $T - T^v$, and test set as T^v.

(c) For each v, apply the Dantzig selector [1] on training data set $T - T^v$ to select J parameters which is denoted as a set called S_J^v; do this repeatedly increasing J from 1 to k , a certain positive integer, and get sets S_0^v, S_1^v, ..., S_p^v.

(d) Let $PE^v(J)$ be the prediction error when S_J^v is applied to the test data set T^v, and form the estimate $PE(J) = \frac{1}{5}\sum_{v=1}^{5} PE^v(J)$.

(e) Find the $\hat{J}$ that minimizes PE(J) and our selected model is $S_{\hat{J}}$.

[1] An implementation of the Dantzig Selector is available at: http://www.acm.caltech.edu/l1magic/

This process is called fivefold cross-validation indicating that we divide the full sample set into five parts. Note that this fivefold cross-validation is not the same as estimating the prediction error of the fixed models S_0, S_1, ..., S_P and then choosing the one with the smallest prediction error. This latter procedure is described in [12], and can lead to inconsistent model selection unless the cross-validation test set T^v grows at an appropriate asymptotic rate.

It needs to be emphasized that the fivefold cross-validation is taken once after a running loop from steps (a) to (e). To strengthen the stability of result, we can repeat this kind of cross-validation plenty of times and choose the stable $S_{\hat{J}}$.

3.2 OLS Regression Afterwards

After variable selection, we are able to pick $\hat{J}$ significant variables and have $\hat{J} < n$. Suppose that the smallest significant level we set for the model is *s*. Now we can apply the regression analysis to derive the needed model.

We first do the OLS using the picked out dependent variables $S_{\hat{J}}$ against y. Then, with the first round regression result in hand, we can examine the p-value of each dependent variable. For a certain dependent variable X_k, its corresponding p-value tells us the smallest significance level at which the H_0: $\beta_k=0$ (null hypothesis) would be rejected given the observed value of the t statistic. Therefore, a large p-value indicates that the corresponding parameter is not significant in this model. Such insignificant parameters should be filtered out of the ultimate model.

According to the Backward Elimination method, we implement the above analysis as the follows. Find the largest p-value from OLS regression result and see if it is larger than the smallest significant level *s* we set. If this p-value meets this condition, filter the corresponding parameter out and run OLS regression once again using the rest of parameters until there is no p-value larger than *s*.

Now we are able to obtain a regression model, where the independent variables are activities that have significant impact to the dependent variable, namely project performance. The coefficient denotes the impact of corresponding activity to project performance, including the direction (+ for positive impact, - for negative impact) and relative strength (indicated by the absolute value).

4 Conclusions and Future Work

Understanding the influence of certain activities on the project performance can facilitate the project management. However, the fact that the amount of independent variables (activities in software process) is usually larger than the size of dataset fabricates a barrier for quantitive analysis. The approach we proposed in this paper provides a theoretical basis for solving this problem.

Recently, we are applying this approach to study the influence of acquirer's participation process on project performance. The amount of activities in the investigated acquirer participation process is 84. Under collaboration with several companies in ShanDong province of China, we are able to collect 25 projects. The preliminary result (depicted in Fig. 1 and Fig. 2) shows that our approach works well in such an

84 independent variables, 25 samples situation. Further evaluation and improvement of our approach is still under discussion.

```
      Source |       SS       df       MS              Number of obs =      25
-------------+------------------------------           F(  5,    19) =  144.15
       Model |  .270468179     5  .054093636           Prob > F      =  0.0000
    Residual |  .007129813    19  .000375253           R-squared     =  0.9743
-------------+------------------------------           Adj R-squared =  0.9676
       Total |  .277597992    24  .011566583           Root MSE      =  .01937

------------------------------------------------------------------------------
           y |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
          Da |  -.3447594   .0219539   -15.70   0.000    -.3907095   -.2988093
          Db |  -.0977238   .0069554   -14.05   0.000    -.1122817   -.0831659
          Dc |   .0248959   .0026504     9.39   0.000     .0193486    .0304432
          Dd |   .2343965   .0240759     9.74   0.000     .1840051    .2847878
          Df |  -.0190572   .0023798    -8.01   0.000    -.0240381   -.0140764
       _cons |   .9256581   .0137656    67.24   0.000     .8968463    .9544698
------------------------------------------------------------------------------
```

Fig. 1. The final regression result. Note that the Adj R-squared indicating the explanation of the model reaches nearly 96.8%.

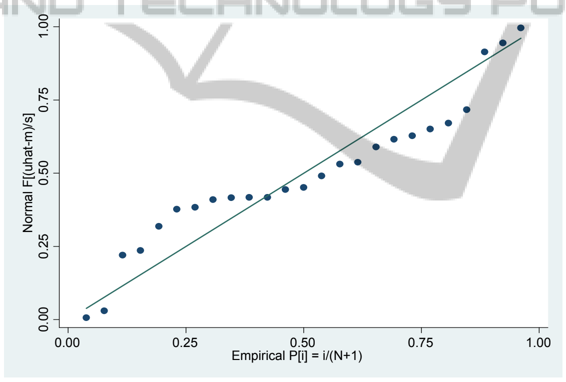

Fig. 2. The result of normal distribution of residuals test. As we can see, plots standing for residuals are distributed around the 45 degree line, which tells that we can generally consider that residuals approximately obey the law of normal distribution.

References

1. Jonathan E. Cook, Alexander L. Wolf, Automating process discovery through event-data analysis. ICSE '95: Proceedings of the 17th international conference on Software engineering (1995), pp. 73-82.
2. Humphrey, W. S., Managing the Software Process. 1989: Addison-Wesley. Software Process: A Roadmap.
3. A. Rai, H. Al-Hindi, The effects of development process modeling and task uncertainty on development quality performance, Information & Management 37, 2000, pp.335–346J.D.

4. Herbsleb, D. R. Goldenson, A system survey of CMM experience and results, Proceedings of ICSE 18, 1996, pp.323–330.
5. IEEE Std 12207:2008. 2008. Systems and software engineering-Software life cycle processes Information technology-Software life cycle processes.
6. Zhihao Chen, Daniel Port, Yue Chen, Barry Boehm, Evolving an Experience Base for Software Process Research, International Workshop on Software Process. 2005.
7. Emmanuel Candes, Terence Tao. The Dantzig selector: statistical estimation when p is much larger than n. The Annals of Statistics, 2007,35(6):2313-2351.
8. Gareth M. James, Peter Radchenko. 2009. A generalized Dantzig selector with shrinkage tuning. *Biometrika*, 2009,96(2):323–337.
9. Gareth M. James, Peter Radchenko, Jinchi Lv. DASSO: connections between the Dantzig selector and Lasso. *Journal of the Royal Statistical Scoiety* 2009, 71(1):127-142.
10. R. Tibshirani, T. Hastie, B. Narasimhan, G. Chu. Class prediction by nearest shrunken centroids, with applications to DNA microarrays. Statistical Science, 2003, 18(1):104-117.
11. Bunea, F., Tsybakov, A. B. and Wegkamp, M. H. Aggregation for Gaussian regression. Ann. Statist. 35 1674–1697. MR2351101. 2007.
12. Shao, J. Linear model selection by cross-validation. Journal of the American Statistical Association 88 486–494. 1992.