直交計画法を用いた局所探索法

田中秀俊
新情報処理開発機構 並列応用三菱研究室 (三菱電機 情報総研)

概要

直交計画法を近傍定義と局所的最適方向決定に用いる局所探索法は、従来の差分近似によるものに比べ、特にノイズを含む目的関数の最適化問題において良い性能を発揮する。この問題は、評価値に影響する要因を全部は考慮できない状態をモデル化していると見なせ、一般性が高い。直交計画法を用いた最適方向決定の頑健さは、各変数の方向決定に近傍全点による統計的推定量を用いていることによる。本稿では方式の特徴と性能をノイズを含んだ簡単な二次関数を用いて示す。

Local Search Using Orthogonal Design of Experiment

Hidetoshi Tanaka
Parallel Application Mitsubishi Laboratory, Real World Computing Partnership
Information Technology R&D Center, Mitsubishi Electric Corp.

Abstract

The local search with orthogonal design of experiment in its neighborhood determination (ODLS) outperforms the local search with the conventional neighborhood when the objective function includes noise. This models practical optimization problems that contains uncontrolled or unobserved variables. ODLS is robust and efficient since it shares all evaluations for direction determination of each variable. We illustrate the characteristics and demonstrate its performance in simple quadratic function plus random noise, and discuss their improved capability from the conventional local search.

1 Introduction

Local search (LS) is the basic algorithm when we solve optimization problems, which contain e.g. quasi-Newton method with definite difference approximation[1], simulated annealing[2] and tabu search[3]. Multi-start local search (MLS), e.g. genetic algorithm[4], runs multiple searches in parallel, and sometimes exchange search reports, to score the excellent performance and quality. Function optimization, e.g. sequential quadratic programming and conjugate gradient method, solve problems whose derivatives are available or easily estimated. LS also solves ones whose derivatives are unavailable, such as integer programming, combinatorial optimization, and black-box optimization. To our knowledge, function optimization methods like conjugate gradient method have never tried to solve protein folding optimization problems[5, 6], since its derivatives are hard to describe. Whereas, several LS such as genetic algorithm[7] tried to solve it. Practical optimization problems often ignore several variables which may affect the objective function, since they cannot be controlled or observed. It can be modeled by the objective function with noise, which is another example of the function whose derivatives are unavailable.

LS usually consists of four stages: (1) determination of the initial point; (2) definition of the neighborhood points of the current point and their evaluation; (3) replacement of the current point; and (4) decision of the termination of the search. LS repeats (2) neighborhood definition plus evaluation, and (3) current point replacement, to proceed the search. We introduce orthogonal design of experiment (OD)[8] to the neighborhood definition, as well as singular factor analysis based on OD to the current point replacement.

LS sometimes employs a simple determination for its neighborhood points (central difference approximation): one variable changes by plus or minus one unit, keeping the other variables unchanged. Let us explain this method using an example of the protein folding optimization problem of the length 30 (which means 30 dihedral angles). A unit of the dihedral angle is 5 degree, thus each variable (dihedral angle) varies 72 discrete values. The current point is set at $(x_1, x_2, ..., x_{30})$. Therefore, the next current point is chosen out of 60 points. This method examines biased 60 points out of $2^{30} = 1$ billion candidates. Whereas, OD examines non-biased 32 points. Every point contains 30 variables each of which is minus-point (-5) or plus-point $(+5)$, such that every variable has 16 plus-points and 16 minus-points, and all two variable-pairs have 8 plus-plus, 8 minus-plus, 8 plus-minus, and 8 minus-minus point pairs. Therefore, we can estimate the better direction (plus or minus) by comparing the means of the evaluation of the 16 plus- and the 16 minus-points. It stabilizes direction determinations of this local search.

This paper shows the LS using OD and partial OD in Section 2, illustrates their characteristics and demonstrates its performance in simple quadratic function plus random noise in Section 3, and discusses their improved parallel processing capability from the conventional local search and future work in Section 4.

2 LS using OD

OD is applied to the second and third stages of LS using the following 6 steps:

- s1: build a 2-level n-variable OD;
- s2: substitute 2 levels by $+d$ and $-d$ to define the neighborhood points;
- s3: evaluate the objective function at each neighborhood point;
- s4: calculate the means of $+d$ and $-d$ of each variable from all neighborhood points;
- s5: choose the next value at each variable based on these means; and
- s6: replace the current point.

(s1) How to build OD:

1. (1) The number of points is $m = 2^q$ where $2^{q-1} \leq n < 2^q$, q is an integer, thus both n and m are represented in q digits of binary or gray code.
2. (2) The level L_{ij} of the i-th variable of the j-th point is determined by the gray code representation of i $(g_0 g_1 \dots g_{q-1})$ and the binary representation of j $(b_0 b_1 \dots b_{q-1})$:

$$L_{ij} = \mathrm{mod}_2 \sum_{s=0}^{q-1} g_s b_{q-s-1} \qquad (1)$$

(s2) How to define neighborhood points N_j ($f(N_j)$ is evaluated at m neighborhood points):

$$\begin{aligned} N_j &= (N_{j0}, N_{j1}, \dots, N_{j(n-1)}) \\ N_{ji} &= \begin{cases} x_i - d & (L_{ij} = 0) \\ x_i + d & (L_{ij} = 1) \end{cases} \end{aligned} \qquad (2)$$

(s4) The 2-level OD is built such that every variable has the same number $(m/2)$ of plus- and minus-neighborhood points. Thus we can compare the means of neighborhood points for i-th variable (μ_i^+ and μ_i^-).

$$\begin{aligned} \mu_i^- &= \frac{2}{m} \sum_{j=0}^{m-1} (1 - L_{ij}) f(N_j) \qquad (3) \\ \mu_i^+ &= \frac{2}{m} \sum_{j=0}^{m-1} L_{ij} f(N_j) \end{aligned}$$

(s5) μ_i^+ and μ_i^- are compared considering a constant B to obtain choices of the direction at i-th variable e_i as follows (an example of maximization):

$$e_i = \begin{cases} +d & (\mu_i^- + B < \mu_i^+) \\ -d & (\mu_i^+ + B < \mu_i^-) \\ 0 & (\text{otherwise}) \end{cases} \qquad (4)$$

B might be determined with respect to the variances of $+d$ and $-d$, however, B is treated as a previously determined constant.

(s6) Thus the level of each variable e_i is determined as $e_i = +d$, 0 or $-d$, to replace the current point.

3 Experiment

3.1 Local searches

We briefly describe four local searches considered in this paper (which are called SD, SDI, ODLS, and P-ODLS), their determinations of neighborhood points and the replacements of the current point. The current point is described as $(x_1, x_2, \ldots, x_n)$, unit d is common to every variable and the following definitions are for maximization problems.

SD determines the direction using the single-variable changed neighborhoods (central difference approximation). It evaluates 2 points (f_i^+ and f_i^-) for i-th variable, as SW does. Then SD determines the direction e by Eq.(5), and replaces the current point.

$$e_i = \begin{cases} +d & (f_i^+ > f_i^-) \\ -d & (f_i^+ < f_i^-) \end{cases} \qquad (5)$$

SDI determines the direction using the iterated evaluation of single-variable changed neighborhoods. It evaluates each point k times where k is a previously determined constant. SDI then determines the direction $e = (e_1, \ldots, e_n)$ by these means and replaces the current point.

ODLS determines the direction using OD. See Section 2.

P-ODLS determines the direction using part of OD. It builds 2-level n-variable OD, selects E points randomly out of the OD to make points P_j^+ where $P_{ij}^+ = L_{ij}$, and provides its reverse points P_j^- where $P_{ij}^- = 1 - L_{ij}$, such that the occurrences of $P_{ij} = 0$ and $P_{ij} = 1$ are the same for each i where $P_j = P_j^+ + P_j^-$. Thus, P-ODLS neighborhood points N_j are defined by Eq.(6):

$$N_{ji} = \begin{cases} x_i - d & (P_{ij} = 0) \\ x_i + d & (P_{ij} = 1) \end{cases} \qquad (6)$$

P-ODLS evaluates the objective function $f(N_j)$ at these $2E$ neighborhood points, so that we can obtain the means of neighborhood points for i-th variable (μ_i^+ and μ_i^-). Then, P-ODLS chooses levels by Eq.(4) and replaces the current point.

3.2 Quadratic function with noise

This experiment uses a quadratic function with random number:

$$\text{maximize} \quad f^1(x) = -\sum_{i=1}^{n} [(x_i - 10)^2 + R]$$

where $n = 100$ and R is a random integer ($0 \leq R \leq 1$, or $0 \leq R \leq 99$). The unit is $d = 1$, and the initial point is generated by random integer ($-50 \leq x_i < 50$).

SD and SDI: SD is unstable since it determines the direction of each variable by only 2 points. SDI repeats evaluation k times for each point. If $n = 100$ and $k = 5$, 1000 evaluations are executed per iteration. Figure 1 shows the relations of iterations (x-axis) and f^1 (y-axis) of SD ($k = 1$) (R1S1), SDI of $k = 5$, (R1S5), and $k = 10$ (R1S10), where R is 0 or 1 in 1/2 probability.

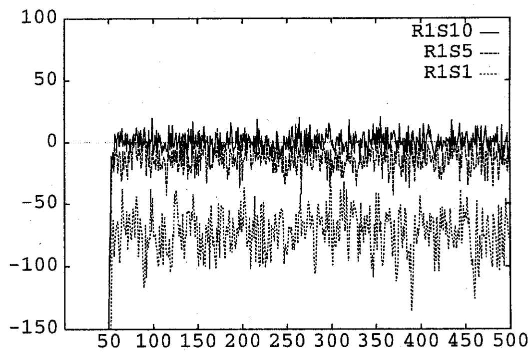

Figure 1: SD and SDI

ODLS and P-ODLS: Figure 2 shows the relations of iterations (x-axis) and f^1 (y-axis) of ODLS of $B = 1$ (OB1), P-ODLS of $E = 10$ (E10B1), $E = 20$ (E20B1), $E = 50$ (E50B1), and SD (S1) for comparison, where R is 0 or

1 in 1/2 probability. E50B1 outperforms S1, which implies 100 points per iteration outperforms 200 points per iteration.

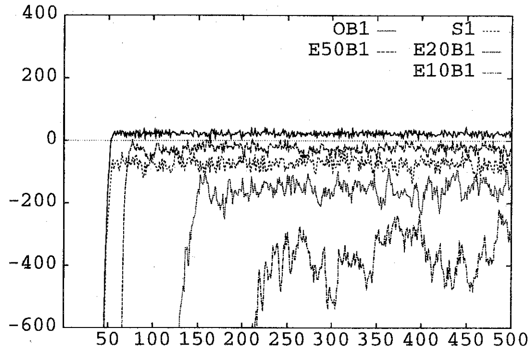

Figure 2: ODLS and P-ODLS

4 Consideration

When the objective function has no noise, ITR's (the number of iterations from the same initial point to the optimal point) of ODLS and SD are the same. Whereas, ODLS outperforms SD in precision when the objective function has noise.

ODLS determines direction of each variables using m points' evaluations, where m is nearly equal to the number of variables of the objective function. ODLS is robust and efficient since all these m evaluations are shared and used by direction determinations of all variables. SDI provides similar result, however, it takes more evaluations than ODLS, since it does not share any evaluations.

The objective function with noise models the real world situation, where it does not cover all the variables to affect the objective. ODLS works in such situation better than SD or SDI.

5 Conclusion

The local search with orthogonal design of experiment in its neighborhood determination outperforms the local search with the conventional neighborhood when the objective function includes noise. It determines direction of each variables using only m points, where m is nearly equal to the number of variables of the objective function. It is robust and efficient since it uses all m evaluations for each variable's direction determination. It efficiently solves practical optimization problems which are modeled by the objective function with noise.

References

[1] Nemhauser, G.L., Rinnooy Kan, A.H.G., and Todd, M.J. *Handbooks in Operations Research and Management Science* Vol.1. Elsevier Science Publishers B.V., 1989.

[2] Kirkpatrick, S. et al. Optimization by Simulated Annealing. *Science* vol.220, pp.671-680. 1983.

[3] Glover, F., and Laguna, M. *Tabu Search.* Kluwer Academic Publishers, 1997.

[4] Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, 1989.

[5] Zhang, X. and Waltz, D. Developing Hierarchical Representations for Protein Structures: An Incremental Approach. *An Artificial Intelligence and Molecular Biology.* AAAI Press, 1993.

[6] Tanaka, H., Shiraishi, M., Sato, H., and Nakajima, K. A Parallel Search Algorithm for Discrete Space and Its Application to Protein Folding Problem. *Proc. of 1998 RWC Symposium.* RWC TR-98001, 1998.

[7] Del Carpio, C.A., and Gogonea, V. Modelling Proteins Conformation in Solution. *Proc. of the 7th Workshop on Genome Informatics.* Universal Academy Press, Inc., 1996.

[8] Montgomery, D.C. *Design and Analysis of Experiment.* John Wiley & Sons, Inc., 1997.