研究員報告書

第６・７号

(1986.3)

岡山理科大学
情報処理センター資料室

目　　次

ブロック符号の誤りの統計的性質の解析

電子理学科教授　宮　垣　嘉　也
電子理学科助手　小　西　憲　一

1. はじめに

ディジタル技術の急速な進展に伴い，データ伝送の需要が急増している。データ伝送では高品質，すなわち低いビット誤り率が要求され，誤り率特性改善のために誤り制御符号が用いられる。その際，どのような符号が最適なのかという符号の選択の問題は，難しい問題であるが工学上重要である。そのために，ある一定長の符号ブロック内で何個のビット誤りがどのくらいの確率で生起するのかを知っておくことは基礎的に重要なことのひとつと考えられる。一方，移動無線通信においては，移動体の走行に伴ってマルチパスフェージングが常時発生し，このためブロック内のビット間では相互に相関をもつことになる。このような記憶のある通信路において，複数個のビットの誤り率を理論的に計算することは非常に困難になる。従来，実験やシミュレーションによる結果が若干得られてはいるが，極端な仮定の下での計算であったり，多くのパラメータの組合せの影響が捉えにくいといった難点がある。[(1)(2)]

ここでは，ブロック長が100ビット以下ぐらいでフェージングが周波数非選択性ならば，ビット間で任意の相関をもつという一般的条件の下で，誤り個数の分布を計算機で効率よく計算するアルゴリズムがあるので，[(3)(4)] それによって，誤り個数の確率分布を数値的に得た。その結果の一例を示す。

2. 誤り個数分布の計算

ブロック内誤り個数分布の理論計算の詳細については文献を参照していただくとして，ここでは割愛する。大雑把に言うと以下の如く計算する。長さnビットのブロック内で，mビット以上が誤る確率分布$P\ (\geqq m \mid n)$は，

$$P\ (\geqq m \mid n)=\sum_{k=m}^{n}(-1)^{k+m}\binom{k-1}{m-1}Q_k \qquad (1)$$

によって計算される。ここで，$Q_k=\sum_{1\leqq n_1<n_2<\cdots<n_k\leqq n}P_{n_1 n_2\cdots n_k}$ は，nビットのうち少なくとも第n_1番目，n_2番目，……，n_k番目のビットで誤りが起る確率$P_{n_1\ n_2\ \cdots\ n_k}$の総和である。

ここでは非同期2進FSK信号の場合を考え，受信機では平均電力N〔W〕の白色ガウス雑音が相加し，独立L重検波前最大比合成ダイバーシチ受信を採用しているとする。通信路のフェージングは周波数非選択性高速レイリーフェージングであるとする。以上の仮定の下で$P_{n_1\ n_2\ \cdots\ n_k}$は

$$P_{n_1 n_2 \cdots n_k} = (\frac{1}{2})^k \{ \prod_{j=1}^{k} \frac{1}{(1+\frac{S}{2N}\lambda_j)} \}^L \qquad (2)$$

で与えられる。ここで，λ_j は各ダイバーシチブランチにおける，（n_1，n_2，…，n_k）で示される各ビットにおけるフェージングの標本値間の相関係数行列の固有値である。

3. 計算結果の例

ブロック内の隣接ビット間のフェージング相関係数ρは，

$$\rho = \exp(-2\pi f_D I T_b) \qquad (3)$$

で与えられるとする。ただし，f_D はフェージングのドプラスペクトルの半値幅，I はインタリーブ距離，T_b はビット間隔である。

表Ⅰに示すn，ρ，ダイバーシチ枝の平均S/Nの値について計算を行った。計算結果の一例として，$n=31$，$S/N=10\ dB$ の場合の結果を$L=1$～4の場合について，各々図1から図4に示す。さらに，これらの図より$P(\geqq m \mid n) \geqq 10^{-4}$ となるブロック内誤り個数の最大数を読みとってρ及びLについてまとめると表Ⅱのようになる。ダイバーシチ受信により大幅に減少することがわかる。

表Ⅰ　$P(\geqq m \mid n)$ 計算のパラメータ

ブロック長 n	7,　15,　23,　31
ρ	0.77 ～ 0.9995
ダイバーシチ枝の平均 S/N	−10 dB ～ 40dB

表Ⅱ　$P(\geqq m \mid n) \geqq 10^{-4}$ となるブロック内誤り最大個数

ρ ＼ L	1	2	3	4
0.77	11	5	3	2
0.95	15	7	4	2
0.995	20	14	8	4
0.9995	22	17	11	6

$n=31$，$S/N=10\ dB$

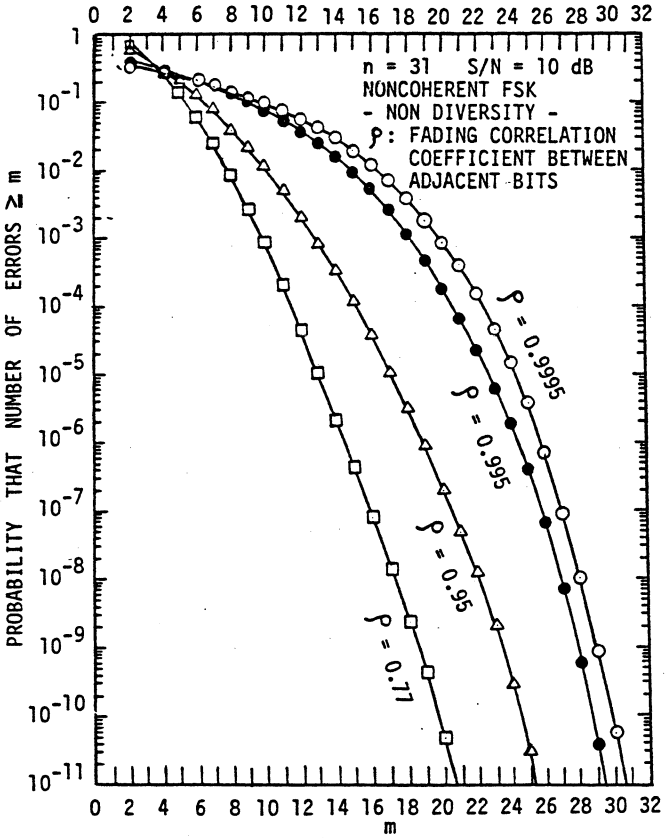


図1. P（$\geqq m \mid n$）計算例
$n=31$, ダイバーシチなし

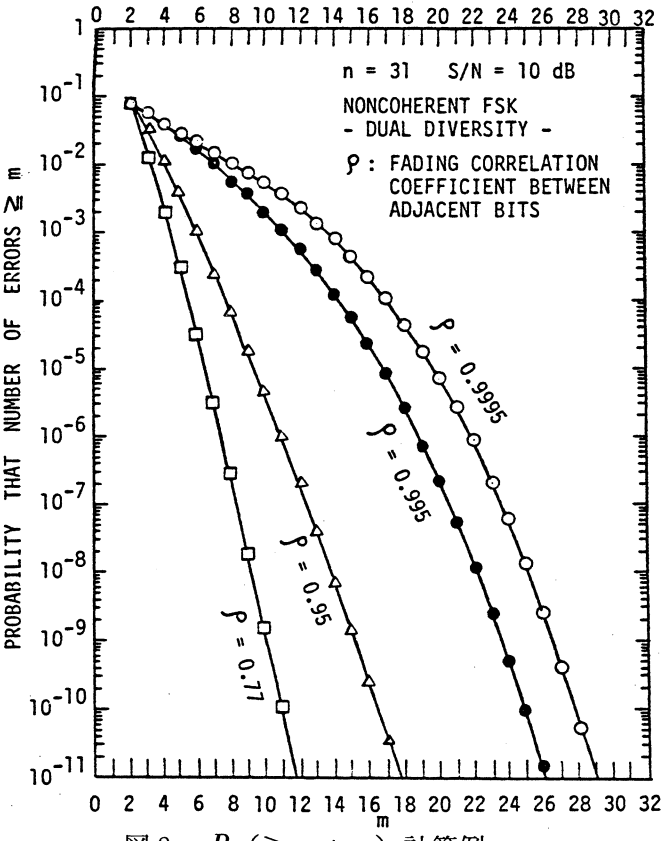


図2. P（$\geqq m \mid n$）計算例
$n=31$, 2重ダイバーシチ

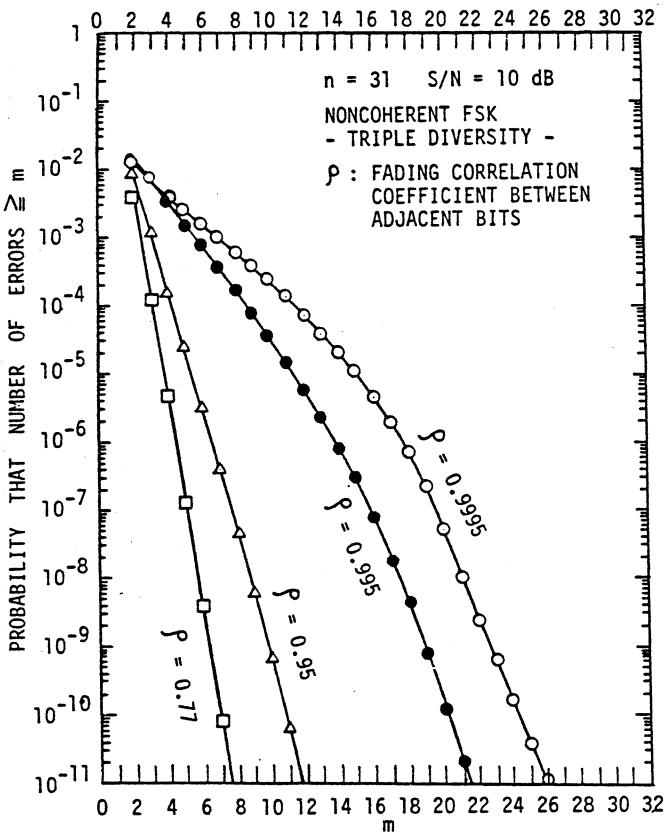


図3. P（$\geqq m \mid n$）計算例
$n=31$, 3重ダイバーシチ

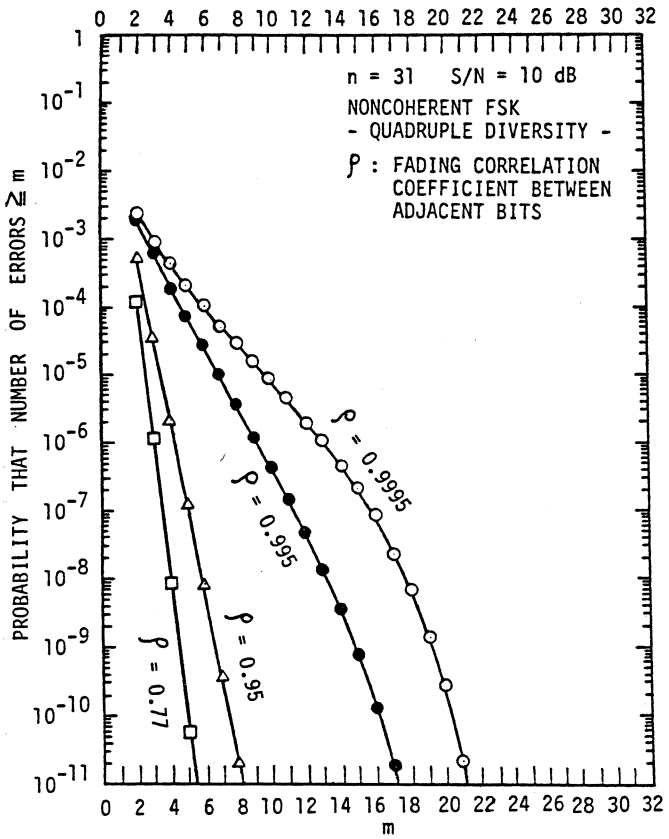


図4. P（$\geqq m \mid n$）計算例
$n=31$, 4重ダイバーシチ

4. むすび

長さnビットのブロック内でmビット以上が誤る確率分布を，フェージングがビット間で任意の相関をもつという一般的条件下で，ダイバーシチ受信した場合に計算し，その結果の一例を示した。

5. 文　　献

（1） P. J. Mabey：“Mobile radio data transmission…… coding for error control”, IEEE Trans. Veh. Technol., VT-27, 3, pp. 99-109 (Aug. 1978).

（2） D. Arnstein：“Correlated error statistics on troposcatter channels”, IEEE Trans. Commun. Technol., COM-19, 4, pp. 225-228 (April 1971).

（3） 宮垣：“フェージング通信路における誤り訂正符号の検出誤り率の計算法に関する一考察”, 信学論（B), 56-B, 6, pp. 274-275（昭48-06）.

（4） 宮垣，森永，滑川：“フェージング通信回線における誤り訂正符号の効果について”，信学技報，CS 73-131（昭 49-01）.

プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	
プログラム名			
作成者氏名	宮垣嘉也，小西憲一		
形式	ⓐ コンプリート・プログラム　c 関数副プログラム b サブルーチン・副プログラム　d その他		
使用機器	a 磁気テープ　入力（　）本，　出力（　）本 b CRTの種類（　） c 磁気ディスク・ファイル（　）個 d その他（　）		
プログラムサイズ	a ソース・プログラム（　）行 b オブジェクト・プログラム（　）KW		
使用言語	a FORTRAN 77　c COBOL　e PL/1 b PASCAL　d LISP　f ASSEMBLER g その他（FORTRAN Ⅳ　）		
処理形態	a バッチ処理専用　c バッチ・TSS兼用 b TSS専用		
使用条件等	4倍精度 （枠内に入りきらない場合は，Ａ４サイズのもので説明を添付してください）		

手書き漢字画像のフィードバック型定線幅2値化法

A Feedback Binarization Method with Constant Line Width for Handprinted KANJI Patterns.

電子理学科助教授　塩野　充

あらまし

手書き漢字画像の2値化において，方向コード化を用いた矩形の領域分割を行い，各領域について2値化の閾値にフィードバックをかけ，原画像の濃淡や運筆方向による筆圧の違いに影響されない一定線幅の漢字パターンを得る方法を提案する。

1. はじめに

画像入力装置によってディジタル入力された手書き文字パターンはその儘では多値の濃淡画像ゆえ，文字認識等の処理をするには2値化して線図形に変換しなければならない。文字パターンの2値化法としてはもっとも単純な固定閾値法をはじめ，濃度微分法[1]，濃度ヒストグラム法[1]，局所的濃度分布法[1]，判別閾値法[2]などが提案されている。しかしこれらの手法は，いずれも一般的な画像の2値化におけるノイズの低減を主眼としており，対象を文字パターンに限定しているわけではない。一方，対象を文字パターンに限定してみた場合，ノイズを少なくすることは当然であるが，それ以外に文字線の線幅が揃っていることが望まれる。特に最近は細線化を行わないで，原図形のまま認識を始める認識手法が多く提案されつつあるので，原図形の文字パターンに細い太いのバラつきのない方が望ましい。これはサンプル間のバラつきだけでなく，同一サンプルにおけるストローク間のバラつきをも含む。本研究では手書き漢字画像の2値化において，方向コード化[3]を用いた矩形の領域分割を行い，各領域について2値化の閾値にフィードバックをかけ，原画像の濃淡や運筆方向による筆圧の違いに影響されない一定線幅の漢字パターンを得る方法を提案し，基礎的な実験を行う。

2. 処理の流れ

図1に本方式のジェネラルフローを示す。ステップ（a）ではまず判別閾値法の閾値θ_0にて画面全体の初期2値化を行う。（b）では4方向の方向コード化[3]を行う。方向コード化とは各黒画素が4つの方向（コード1：右上がり，コード2：垂直，コード3：左上がり，コード4：水平）のいずれの文字線に属しているかを4方向に延ばしたゾンデで調べ，各黒画素の画素値をコードで置き換える処理である。（c）では同一方向コードの画素のみからなる4枚の方向画面G_γ（$\gamma=1\sim4$）を作成する。（d）ではG_γ内の各セグメント（棒状の単連結領域）に外接し，長辺がγ方向の直線に平行な矩形領域を切りだす。（e）では4つの方向画面のすべての矩形領域について個々

にフィードバック２値化ルーチンを適用する。図２にフィードバック２値化ルーチンのフローを示す。（ａ）では各セグメントの平均線幅ｗを計算する。（ｂ）ではｗと目標線幅$\hat{w}$との誤差を計算し，許容誤差ε未満ならば終了し，ε以上ならば閾値θに補正幅δを増減してもう一度その矩形領域だけについて原画像Ｆを２値化し，Ｆ上のその矩形領域をＧγの同じ領域にコピーし，（ａ）に戻って再び線幅を計算し，ｗと$\hat{w}$の誤差を計算する。これを誤差がε未満に収束するまで繰り返す。収束せずに振動するような場合のために（ｃ）でフィードバック回数ｔをＴ回以内に制限している。最後に４枚のＧを１枚に合成して結果が得られる。

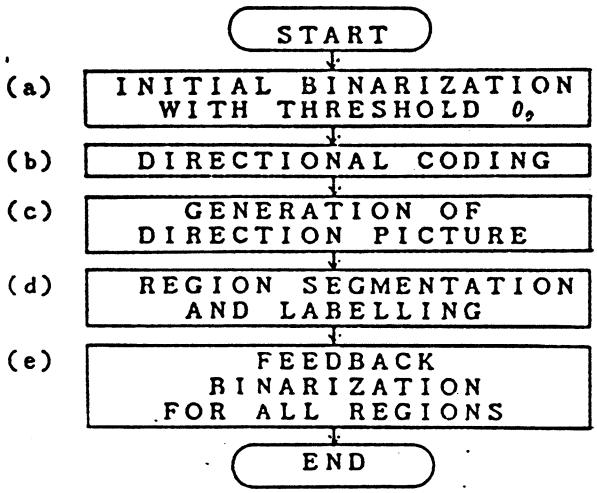


図1.　ジェネラルフローチャート

Fig. 1　The general flowchart.

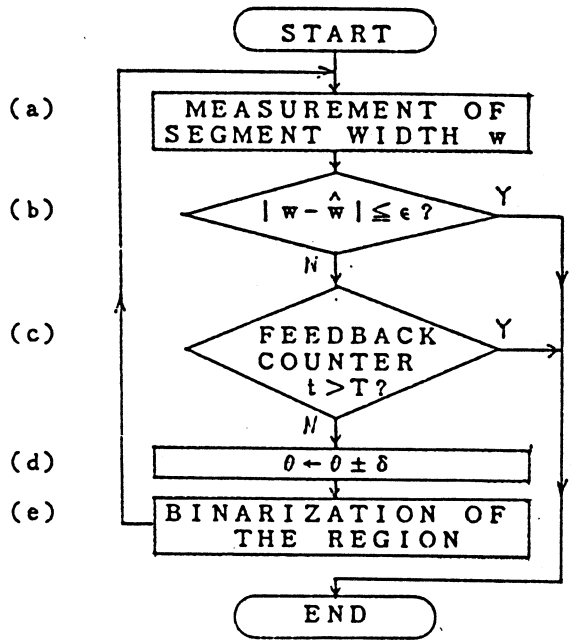


図２　フィードバック２値化ルーチンのフローチャート

Fig. 2　The flowchart of feedback binarization routine.

3. 線幅計算と2値化閾値の補正

1つのセグメントの線幅は面積を長で割って得られる。すなわち，セグメントの両端点の位置を（x_1, y_1），（x_2, y_2），面積をSとするとそのセグメントの線幅wは，

$$w = S / \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad (1)$$

となる。2値化の閾値θはフィードバックするたびに変化させてゆくが，式(2)の様にフィードバック回数tが増えるにつれて補正幅δを小さくしてゆく。

$$\left.\begin{aligned} &\theta(t) = \theta(t-1) \pm \delta(t) \\ &\delta(t) = 0.3 - 0.01t \\ &(t = 1,\ 2,\ \cdots\cdots,\ T\ ;\ \theta(o) = \theta_0) \end{aligned}\right\} \qquad (2)$$

複号＋は線幅を細める場合，－は太める場合である。tの上限値Tは30とする。

4. 2値化実験

実験は電総研手書き教育漢字データベースETL 8を用いて行った。使用したのはETL 8 第1巻の先頭からの漢字50カテゴリで各カテゴリ5サンプル計250サンプルについて行った。原画面サイズは127×128であるが64×64に縮少して用いた。使用言語はFORTRAN77，計算機は本学のMELCOM−COSMO 800 Ⅲである。その結果を表1に示す。目標線幅$\hat{w}$は画面一辺の約1/20より3とし，許容誤差$\varepsilon = 0.1$とした。処理後は各方向ともほぼ目標線幅に到達しており，線幅の分散も大幅に小さくなっている。フィードバック回数tの平均値は4.76回で，平均処理時間は約12秒であった。

表1　フィードバック型定線幅2値化法による実験結果

	方向コード別平均線幅				平均線幅	分　散
	コード1	コード2	コード3	コード4		
処理前	2.67	2.48	2.64	2.37	2.54	0.13179
処理後	3.00	2.99	3.00	2.99	2.99	0.00125

5. むすび

手書き漢字画像のフィードバック型定線幅2値化法を提案し，基礎的な実験を行った。その結果，5回未満のフィードバックによりほぼ目標線幅に修正されることが分かった。今後は他の画面サイズでの実験や，処理時間の短縮等の改善を行いたいと考える。最後に，本研究の実験を担当して頂いた昭和59年度塩野ゼミ卒研学生の酒井国博氏（現在，三菱電機東部コンピュータシステム株式会社），西野一寿氏（現在，三菱電機コントロールソフトウェア株式会社）に厚く感謝する。

文　献

（1）　中田和男編：パターン認識とその応用，コロナ社（昭53）。

（2）　大津展之：“判別および最小2乗基準に基づく自動しきい値選定法”，電子通信学会論文誌（D），J63－D，4，pp．349－356（昭55－04）。

（3）　塩野　充，馬場口　登，真田英彦，手塚慶一：“方向性マッチングによる常用手書き文字の認識”，電子通信学会論文誌（D），J63－D，5，pp．402－409（昭55－05）。

プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	59. 1
プログラム名	Ｐ４		
作成者氏名	塩野　　充，酒井国博，西野一寿		
形式	ⓐ コンプリート・プログラム　　c 関数副プログラム b サブルーチン・副プログラム　　d その他		
使用機器	a 磁気テープ　入力（ 1 ）本，出力（　　）本 b CRTの種類（　　） c 磁気ディスク・ファイル（　　）個 d その他（　　）		
プログラムサイズ	a ソース・プログラム（ 500 ）行 b オブジェクト・プログラム（　　）KW		
使用言語	ⓐ FORTRAN 77　　c COBOL　　e PL/1 b PASCAL　　d LISP　　f ASSEMBLER g その他（　　）		
処理形態	a バッチ処理専用　　ⓒ バッチ・ＴＳＳ兼用 b ＴＳＳ専用		
使用条件等	（枠内に入りきらない場合は，Ａ４サイズのもので説明を添付してください）		

モリブデン錯体のX線結晶解析

理学部化学科助教授　柴　原　隆　志

1. はじめに

モリブデンのクラスター錯体における電子状態と構造との関係を明らかにするため，一連の化合物のX線結晶構造解析を行った。

2. プログラム

プログラムとしては前年度までに作製もしくはconvertしたものの他に，直接法のプログラムMULTAN78を導入して用いた。

3. 結　果

Table 1に示すように，錯体が還元されるにつれて，Mo－Moの距離およびMo_4の体積が減少し，Mo－Mo間の結合が強くなっていることがわかる。このことより，還元の際，電子は結合性分子軌導に入っていくことがわかる。3つの錯体のうちの1つをFigur 1に示す。

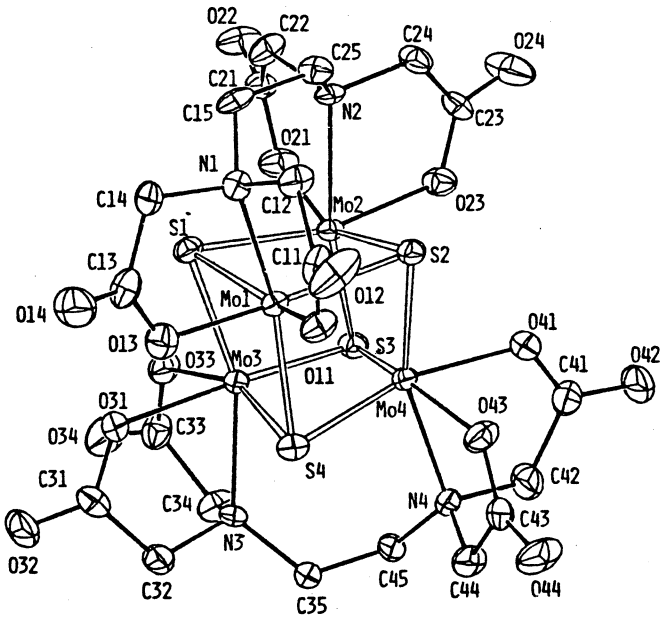


Figure 1. Perspective view of $[Mo_4S_4(edta)_2]^{3-}$. Bond distances (Å): Mo1–Mo2, 2.775 (2); Mo1–Mo3, 2.794 (2); Mo1–Mo4, 2.880 (2); Mo2–Mo3, 2.845 (2); Mo2–Mo4, 2.755 (2); Mo3–Mo4, 2.796 (2); Mo1–S1, 2.352 (3); Mo1–S2, 2.364 (4); Mo1–S4, 2.355 (3); Mo1–O11, 2.139 (8); Mo1–O13, 2.151 (10); Mo1–N1, 2.284 (11); Mo2–S1, 2.369 (4); Mo2–S2, 2.363 (3); Mo2–S3, 2.351 (3); Mo2–O21, 2.145 (8); Mo2–O23, 2.129 (10); Mo2–N2, 2.283 (11); Mo3–S1, 2.351 (3); Mo3–S3, 2.357 (4); Mo3–S4, 2.353 (3); Mo3–O31, 2.131 (11); Mo3–O33, 2.153 (9); Mo3–N3, 2.290 (11); Mo4–S2, 2.351 (4); Mo4–S3, 2.355 (3); Mo4–S4, 2.358 (4); Mo4–O41, 2.143 (11); Mo4–O43, 2.136 (9); Mo4–N4, 2.300 (11).

Table 1. Volumes and Bond distances of $[Mo_4S_4(edta)_2]^{n-}$

	$[Mo_4S_4(edta)_2]^{4-}$	$[Mo_4S_4(edta)_2]^{3-}$	$[Mo_4S_4(edta)_2]^{2-}$
Oxidation state of molybdenum	3.00	3.25	3.50
Volume ($\AA^3$)			
Mo_4	2.540 (5)	2.601 (6)	2.655 (5)
Mo_4S_4	10.78 (2)	10.87 (3)	10.97 (2)
Octahedron[a]	15.08	15.03	14.59
Bond distance ($\AA$)[b]			
Mo ··· Mo	2.783	2.807	2.826
Mo ··· S	2.355	2.356	2.355
Mo ··· N	2.28	2.29	2.27
Mo ··· O	2.18	2.14	2.09

a) mean value of four coordination octahedra around respective Mo's
b) mean value

プログラム登録表

（様式１）

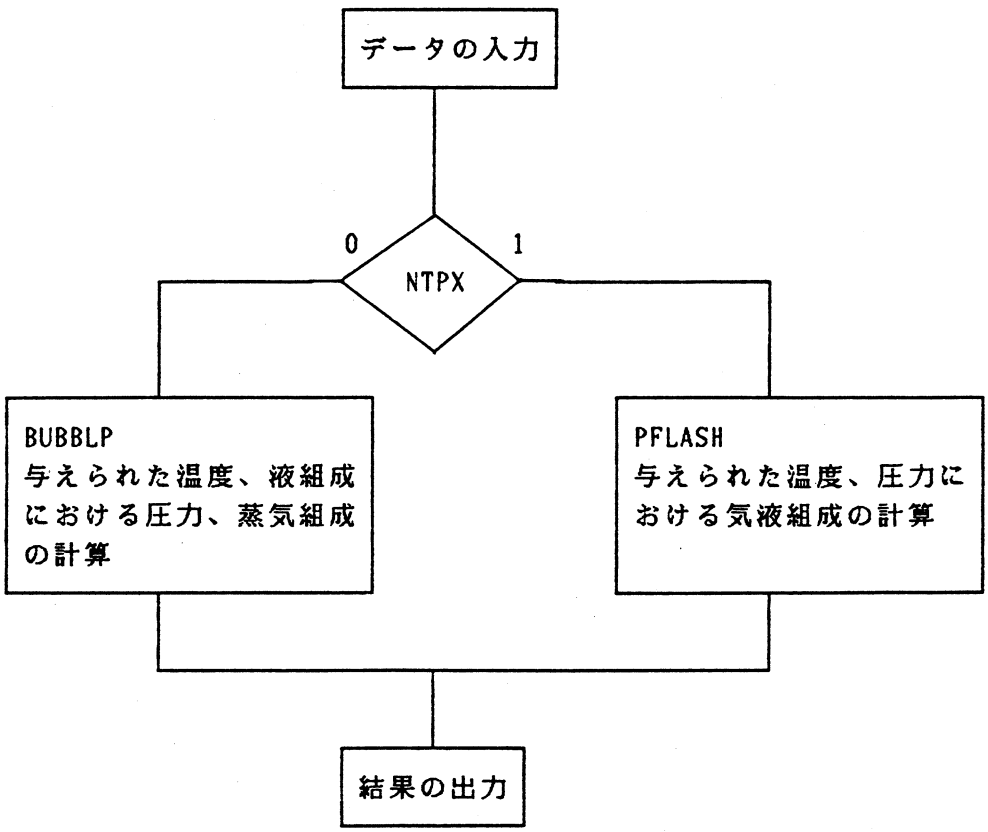
ライブラリ名	（センタ記入）	作成日付	
プログラム名	MULTAN		
作成者氏名	柴　原　隆　志（Conversionのみ）		
形式	ⓐ コンプリート・プログラム　　c 関数副プログラム b サブルーチン・副プログラム　　d その他		
使用機器	a 磁気テープ　入力（　　）本,　出力（　　）本 b CRTの種類（　　） c 磁気ディスク・ファイル（ 3 ）個 d その他（　　）		
プログラムサイズ	a ソース・プログラム（　　）行 b オブジェクト・プログラム（　　）KW		
使用言語	ⓐ FORTRAN 77　c COBOL　e PL/1 b PASCAL　d LISP　f ASSEMBLER g その他（　　）		
処理形態	a バッチ処理専用　　c バッチ・TSS兼用 ⓑ TSS専用		
使用条件等	（枠内に入りきらない場合は，Ａ４サイズのもので説明を添付してください）		

平衡物性推算プログラムの作製

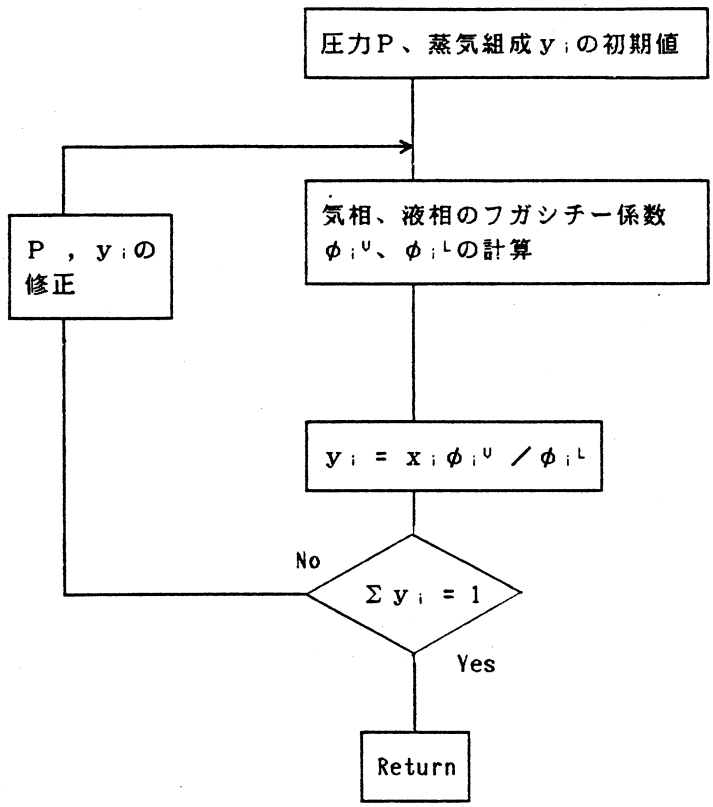
応用化学科講師　宮　野　善　盛

純流体および流体混合物の平衡物性（蒸気圧，露点，沸点，気液組成）を推算するプログラムを作成した。このプログラムで使用した推算法は，Soave−Redlich−Kwong 状態方程式に基づくものであり，低圧から高圧までの流体の平衡物性を推算することができる。また，その適用温度範囲は，混合物中の各成分のうち最も沸点の高い成分の融点から臨界点までである。

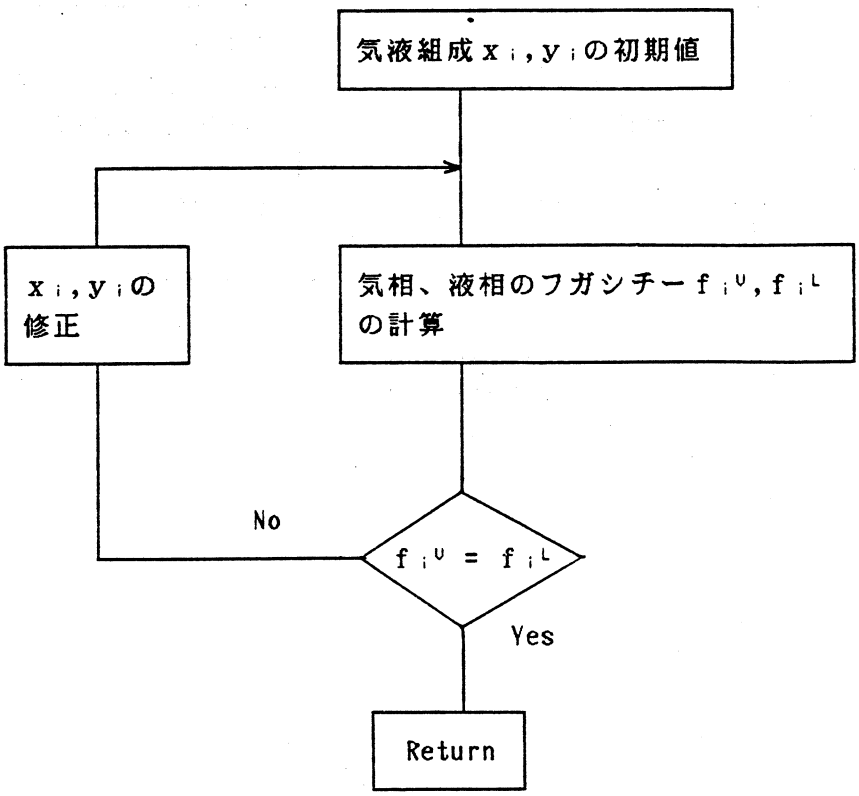
この推算のために必要な最小限のデータは，純物質の臨界温度，臨界圧力およびPitzer の偏心係数である。



SUBROUTINE　BUBBLP



SUBROUTINE　PFLASH



プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	59. 1
プログラム名	VLE. FORT 77		
作成者氏名	宮　野　善　盛		
形　　式	ⓐ コンプリート・プログラム　　c 関数副プログラム b サブルーチン・副プログラム　　d その他		
使用機器	a 磁気テープ　入力（　　）本,　　出力（　　）本 b CRTの種類（　　） c 磁気ディスク・ファイル（　　）個 d その他（　　）		
プログラムサイズ	a ソース・プログラム（ 444 ）行 b オブジェクト・プログラム（　　）KW		
使用言語	ⓐ FORTRAN 77　　c COBOL　　e PL/1 b PASCAL　　d LISP　　f ASSEMBLER g その他（　　）		
処理形態	a バッチ処理専用　　ⓒ バッチ・TSS兼用 b TSS専用		
使用条件等	（枠内に入りきらない場合は，Ａ４サイズのもので説明を添付してください）		

A DISCRETE SOFTWARE RELIABILITY GROWTH MODEL WITH NUMBER OF TEST RUNS

Shigeru YAMADA, Assistant Professor

Department of Electronic Engineering, Okayama University of Science, Ridai-cho 1-1, Okayama 700, Japan.

Abstract This paper investigates a software reliability growth model which provides a plausible description in case that the error detection process is characterized by two types of errors. The model is discussed by assuming a nonhomogeneous Poisson process of which the random variable is defined as the number of software errors by n test runs ($n=0,1,2,\cdots$).

Keywords Stochastic model applications, software reliability growth, software error, test run, nonhomogeneous Poisson process

1. INTRODUCTION

Modeling and analysis of software error occurrence phenomena are of major interest in software reliability. Many software reliability models have been proposed and discussed by many authors. The reliability models applicable during the testing phase in the software development are called software reliability growth models (see Ramamoorthy and Bastani [3], Yamada and Osaki [5]). The models can estimate software reliability in terms of the estimated number of software errors remaining in the software system. In this area, one of the key objectives is to describe an error detection process in which the software errors in a tested software are detected and all of them are corrected. The calendar time or the machine execution time has been often used as the unit of error detection period in the soft-

ware reliability growth models proposed so far. For software systems such as application programs, however, the appropriate unit of error detection period may be the number of test runs or executed test cases. Several works, which estimate software reliability from test data observed by test runs or executed test cases, have been by Brown and Lipow [1] and Nelson [2]. A software failure is defined as an unacceptable departure of program operation caused by a software error in the software system.

In this paper a software reliability growth model with the number of test runs or executed test cases is proposed. The model provides a plausible description in case that the error detection process is characterized by two types of errors: Some are easy to be detected and other are difficult to be. The underlying stochastic process is a nonhomogeneous Poisson process (NHPP) of which the random variable is defined as the number of errors detected by n test runs ($n = 0, 1, 2, \cdots$). The error detection rate per error for such an error detection process is not constant but depends on the number of test runs or executed test cases. The related quantities which can be used as assessment measures for software reliability and the maximum likelihood estimates of the model parameters are obtained. The application of the model, to actual software error data is also presented.

2, MODEL DEVELOPMENT

An implemented software system is tested in the software develoment. During the testing phase a software system such as an application program is executed with a sample of test cases to detect and correct errors in the system. It is assumed that the correction of errors does not introduce any new errors. Yamada and Osaki [4] proposed a software reliability growth model for an error detection process described by the cumulative number of errors detected by n test runs ($n = 0, 1, 2, \cdots$). The model is based on an NHPP and the mean value function representing the expected total number of errors detected by n test runs is given by

$$C(n) = a(1-(1-b)^n), \quad a>0, \quad 0<b<1, \tag{1}$$

where a is the expected number of errors to be eventually detected and b is the error detection rate per error (per test run).

In general, however, the chance of detecting an error on a given test run is not constant. The reason is that the errors detected early in the testing are different from those detected later on. This can be incorporated by assuming that there are two types of errors: Type 1 (Type 2) error is easy (difficult) to be detected.

Then, the NHPP model with C(n) defined in (1) is modified as follows: Let $\{N_d(n), n \geq 0\}$ $(n = 0, 1, 2, \cdots)$ denote a discrete counting process representing the cumulative number of Types 1 and 2 errors detected by n test runs. The error detection process with such two types of errors can be described by an NHPP as

$$\Pr\{N_d(n)=x\}=\frac{\{C_p(n)\}^x}{x!}\exp[-C_p(n)] \quad (x,\ n=0,\ 1,\ 2,\ \cdots), \tag{2}$$

$$C_p(n)=\sum_{i=1}^{2} c_i(n), \tag{3}$$

$$c_i(n)=p_i a\,(1-(1-b_i)^n) \quad (i=1,\ 2), \tag{4}$$

$$\sum_{i=1}^{2} p_i=1,\ p_i>0 \quad (i=1,\ 2), \tag{5}$$

$$a>0,\ 0<b_2<b_1<1, \tag{6}$$

where

p_i = the content proportion of Type i (i = 1, 2) error,

a = the total expected number of Types 1 and 2 errors to be eventually detected ($p_i a$ is the initial error content of Type i (i = 1, 2) error),

b_i = the error detection rate per error (per test run) for the error detection process of Type i (i = 1, 2) error, i.e.,

$$b_i=\frac{c_i(n+1)-c_i(n)}{p_i a-c_i(n)} \tag{7}$$

The mean value function $C_p(n)$ represents the expected total number of Types 1 and 2 errors detected by n test runs where $C_p(0)=0$ and $C_p(\infty)=a$. The error detection rate per error (per test run) is given by

$$z_p(n) \equiv \frac{C_p(n+1)-C_p(n)}{a-C_p(n)}$$

$$= \frac{\sum_{i=1}^{2} b_i p_i (1-b_i)^n}{\sum_{i=1}^{2} p_i (1-b_i)^n}, \qquad (8)$$

which is a monotone decreasing function in n with

$$z_p(0) = \sum_{i=1}^{2} p_i b_i, \quad z_p(\infty) = b_2. \qquad (9)$$

The equations (8) and (9) imply that most of remaining errors in the late phase of testing are Type 2 errors, i. e., errors difficult to be detected.

3. ASSESSMENT MEASURES

Let $\overline{N}_d(n)$ denote the total number of Types 1 and 2 errors remaining in the software system after n-th run. Then,

$$\overline{N}_d(n) = N_d(\infty) - N_d(n), \qquad (10)$$

and the expectation and variance of $\overline{N}_d(n)$ are

$$E[\overline{N}_d(n)] = a - C_p(n)$$

$$= a \sum_{i=1}^{2} p_i (1-b_i)^n$$

$$= \mathrm{Var}[\overline{N}_d(n)]. \qquad (11)$$

Suppose that n_c errors have been detected by n test runs. The conditional distribution of $\overline{N}_d(n)$, given that $N_d(n) = n_c$, is given by

$$\Pr\{\bar{N}_d(n)=y \mid N_d(n)=n_c\}=\frac{\{a-C_p(n)\}^y}{y!}\exp[-\{a-C_p(n)\}]$$

$$(y=0,\ 1,\ 2,\ \cdots), \tag{12}$$

which is a Poisson distribution with mean $\{a-C_p(n)\}$.

The probability of no errors detected between n-th and $(n+h)$-th test runs, given that n_0 errors have been detected by n test runs, is given by

$$R(n,h)\equiv\Pr\{N_d(n+h)-N_d(n)=0 \mid N_d(n)=n_0\}$$

$$=\exp[-\sum_{i=1}^{2}(1-b_i)^n c_i(h)], \tag{13}$$

which implies the reliability function of n, independent of n_o. The reliability function of (13) is called software reliability.

4. MAXIMUM LIKELIHOOD ESTIMATION

For the observed data in the testing of a software system, the parameters a and b_i $(i=1, 2)$ in the NHPP model with $C_p(n)$ can be estimated by the method of maximum likelihood. Suppose that the data set observed in the testing of the software system is available in the form of pairs (n_i, x_i) $(i=1, 2, \cdots, k; 0<n_1<n_2<\cdots<n_k)$ where x_i is the cumulative number of errors detected by n_i test runs. The joint probability mass function of $\{N_d(n_1)=x_1,\ N_d(n_2)=x_2,\ \cdots,\ N_d(n_k)=x_k\}$, i.e., the likelihood function, is given by

$$L_d\equiv\Pr\{N_d(n_1)=x_1,\ N_d(n_2)=x_2,\cdots,\ N_d(n_k)=x_k\}$$

$$=\prod_{j=1}^{k}\frac{\{C_p(n_j)-C_p(n_{j-1})\}^{x_j-x_{j-1}}}{(x_j-x_{j-1})}\exp[-\{C_p(n_j)-C_p(n_{j-1})\}], \tag{14}$$

where $n_o=0$ and $x_o=0$. The maximum likelihood estimates $\hat{a}$ and $\hat{b}_i$ $(i=1, 2)$ of the model parameters a and b_i $(i=1, 2)$ are given by a solution to the simultaneous likelihood equations $\partial \ln L_d/\partial a=\partial \ln L_d/\partial b_i=0$ $(i=1, 2)$:

$$\frac{x_k}{a} = 1 - \sum_{m=1}^{2} p_m (1-b_m)^{n_k}, \tag{15}$$

$$an_k (1-b_i)^{n_k - 1} = \sum_{j=1}^{k} \frac{(x_j - x_{j-1})\{n_j (1-b_i)^{n_j - 1} - n_{j-1}(1-b_i)^{n_{j-1}-1}\}}{\sum_{m=1}^{2} p_m \{(1-b_m)^{n_{j-1}} - (1-b_m)^{n_j}\}}$$

$$(i = 1, 2), \tag{16}$$

which can be solved numerically under condition that $0 < b_2 < b_1$.

5. NUMERICAL EXAMPLE

The data analyzed here were taken in the actual testing of an application program. The program consists of approximately 50,000 lines of code written in assembly and PL/I languages. The error detection period was measured in the number of executed test cases. The cumulative number x_i of errors detected by n_i test cases ($i = 1, 2, \cdots, 18$) were observed as illustrated in Fig. 1.

For the 18 data pairs, solving (15) and (16) numerically under condition that $0 < b_2 < b_1$ yields $\hat{a} = 85.22$, $\hat{b_1} = 0.00301$, and $\hat{b_2} = 0.00075$ where it is assumed that $p_1 = 0.9$ and $p_2 = 0.1$. The maximum likelihood estimates $\hat{C}_p(n)$ and $\hat{R}(n, h)$ for $C_p(n)$ and $R(n, h)$ are shown in Fig. 1 and Fig. 2, respectively.

ACKNOWLEDGMENT

The author is grateful to Mr. Mitsuru Ohba, Manager of Software Engineering, Science Institute in IBM Japan, Ltd., for the valuable software error data.

REFERENCES

[1] J.R. Brown and M. Lipow, "Testing for software reliability", Proc. International Conference on Reliable Software, Los Angels, CA (1975)

518 − 527 .

〔2〕 E. C. Nelson, "Estimating software reliability from test data", Microelectron. Reliab. 17, 67 − 74 (1978).

〔3〕 C.V. Ramamoorthy and F. B. Bastani, "Software reliability − Status and perspectives", IEEE Trans. Software Eng. SE−8, 354−371(1982).

〔4〕 S. Yamada and S. Osaki, "Discrete software reliability growth models", J. Applied Stochastic Models and Data Analysis 1, 65 − 77 (1985).

〔5〕 S. Yamada and S. Osaki, "Software reliability growth modeling: Models and applications", IEEE Trans. Software Eng. SE − 11, 1431 − 1437 (1985)

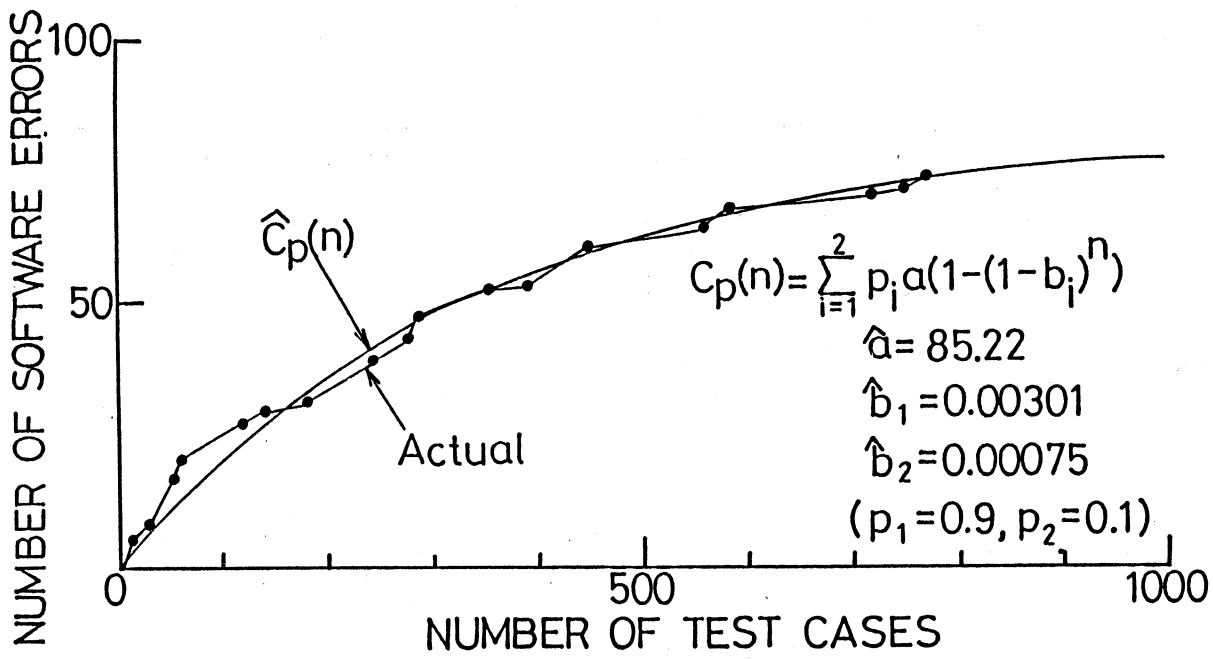


Fig. 1. Maximum likelihood estimation of Cp (n) for an actual data set.

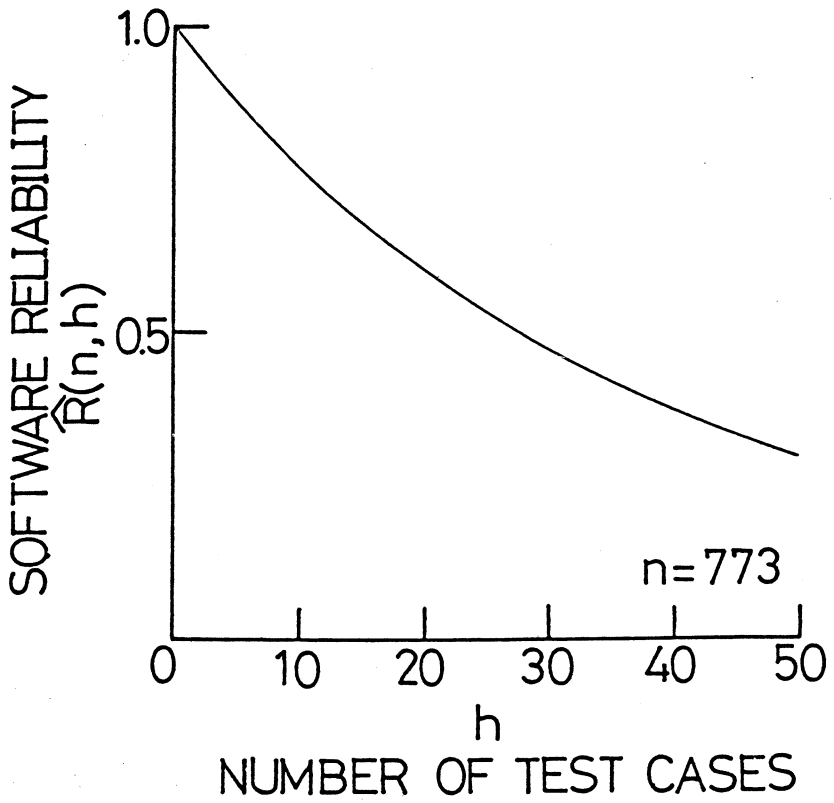


Fig. 2. Maximum likelihood estimation of R(n, h) for an actual data set.

プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	
プログラム名	O:NS－1, O:NSP－2, O:NSP－3,		
作成者氏名	山田　茂, 大寺　浩志		
形式	ⓐ コンプリート・プログラム　c 関数副プログラム b サブルーチン・副プログラム　d その他		
使用機器	a 磁気テープ　入力（　　）本, 出力（　　）本 b CRTの種類（　　） ⓒ 磁気ディスク・ファイル（ 1 ）個 d その他（　　）		
プログラムサイズ	a ソース・プログラム（ 500 ）行 b オブジェクト・プログラム		
使用言語	ⓐ FORTRAN 77　c COBOL　e PL/I b PASCAL　d LISP　f ASSEMBLER g その他（　　）		
処理形態	a バッチ処理専用　ⓒ バッチ・TSS兼用 b TSS専用		
使用条件等	（枠内に入りきらない場合は, A4サイズのもので説明を添付してください）		

金属錯体のX線結晶解析

理学部化学科助教授　柴原隆志

情報処理センターの機種変換（MELCOM COSMO 800 Ⅲ→FACOM 380 S）に伴い，X線結晶構造解析のためのプログラムを変更することが必要となった。

このプログラム変更は，化学科4年生，中島光喜，安田浩一の両君が卒業研究の一端として行ったものである。

変更に際し，従来のカード入力をTSS入力に改め，能率向上をはたした。

プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	
プログラム名	HBLSIV		
作成者氏名 （convens:ov）	中島光喜, 安田浩一, 柴原隆志		
形式	ⓐ コンプリート・プログラム　c 関数副プログラム b サブルーチン・副プログラム　d その他		
使用機器	a 磁気テープ入力（　）本, 出力（　）本 b CRTの種類（　） c 磁気ディスク・ファイル（ 4 ）個 d その他（　）		
プログラムサイズ	a ソース・プログラム（　）行 b オブジェクト・プログラム（　）KW		
使用言語	ⓐ FORTRAN 77　c COBOL　e PL/I h PASCAL　d LISP　f ASSEMBLER g その他（　）		
処理形態	a バッチ処理専用　c バッチ・TSS兼用 ⓑ TSS専用		
使用条件等	（枠内に入りきらない場合は, A4サイズのもので説明を添付してください）		

数理計画法の一般化

電子理学科　成久洋之

1. はじめに

数理計画法の中で非線形計画法は線形計画法の様な強力な解法アルゴリズムを作成することは極めて難しい。現状では問題のタイプにより各種の手法が提案されているが，うまく解ける場合もあるが一般的には必らずしも満足すべきものとは言えない。

線形計画問題に対する単体法はコンピュータコードとしてもかなり普及している現状から，非線形計画問題も線形近似法により求解させる事でより強靱な解法アルゴリズムとして君臨できるものと考える。アルゴリズムの効率性はその強靱性とある面では矛盾する事もあるが，計算論理の単純さはその収束効率以上に望まれる事も多い。

本研究は非線形計画問題を線形近似法によって解く事で数理計画法の統一的アルゴリズムを試みるものである。本来，非線形計画法としての線形近似法はGrifth and StewartによるMAP法（Method of Approximation Programming）として提案されているが，収束効率等の点で必らずしも普及するには至らなかった。しかしながら，効率性の改善等における工夫がなされれば，統一的アルゴリズム開発と云う点で極めて有望なものとなり得る。

非線形計画問題をつぎのように定義する。

$$\left.\begin{array}{lll} \min & f(\boldsymbol{x}) & \boldsymbol{x}\in E^n \\ \text{sub} & h_i(\boldsymbol{x})=0 & i=1,\cdots\cdots,\ m \\ & g_i(\boldsymbol{x})\geqq 0 & i=m+1,\ \cdots\cdots,\ p \end{array}\right\} \qquad (1)$$

この問題を点 $\boldsymbol{x}^{(k)}$ でテーラ近似すると，

$$\left.\begin{array}{lll} \min & f(\boldsymbol{x}^{(k)})+D^T f(\boldsymbol{x}^{(k)})\ (\boldsymbol{x}-\boldsymbol{x}^{(k)}) & \boldsymbol{x}\in E^n \\ \text{sub} & h_i(\boldsymbol{x}^{(k)})+D^T h_i(\boldsymbol{x}^{(k)})\ (\boldsymbol{x}-\boldsymbol{x}^{(k)})=0, & i=1,\cdots\cdots,\ m \\ & g_i(\boldsymbol{x}^{(k)})+D^T g_i(\boldsymbol{x}^{(k)})\ (\boldsymbol{x}-\boldsymbol{x}^{(k)})\geqq 0, & i=m+1,\ \cdots\cdots,\ p \end{array}\right\} \qquad (2)$$

で表わされ，これは明らかに線形計画問題となる。

2. 逐次解法の考え方

(2)式で表わされる線形化された問題を次のように書き直してみる。

$$
\left.
\begin{aligned}
\min \quad & f(\boldsymbol{x}) - f(\boldsymbol{x}^{(k)}) = \sum_{j=1}^{n} \frac{\partial f(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle x_j^{(k)} \\
\text{sub} \quad & \sum_{j=1}^{n} \frac{\partial h_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle x_j^{(k)} = -h_i(\boldsymbol{x}^{(k)}) \\
& \qquad\qquad i = 1, 2, \cdots\cdots, \ m \\
& \sum_{j=1}^{n} \frac{\partial g_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle x_j^{(k)} \geqq -g_i(\boldsymbol{x}^{(k)}) \\
& \qquad\qquad i = m+1, \cdots\cdots, \ p
\end{aligned}
\right\} \qquad (3)
$$

ただし，$\triangle x_j^{(k)} = (x_j - x_j^{(k)})$ とする。

これは線形計画問題であり，これを解くことにより，$(\boldsymbol{x} - \boldsymbol{x}^{(k)}) = (\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(k)})$ が求められたものと仮定する。ここで，

$$
\boldsymbol{x}(k+1) = \boldsymbol{x}^{(k)} + (\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(k)}) \qquad (4)
$$

として，(3)の線形計画問題を $k \leftarrow k+1$ として解かせる。以下この操作を繰り返して

$$
\| f(\boldsymbol{x})(k+1) - f(\boldsymbol{x})^{(k)} \| \leqq \rho \qquad (5)
$$

となったところで，それを最適解と看做そうとするものである。ただし，ρ は許容値とする。

与えられた非線形計画問題(1)を，テーラの1次近似により逐次線形計画問題として解く場合，実行可能領域外に各段階で求める点が出ない様にするために適当な線形条件式を付加しなければならない。そのために必要なつぎの各記号を説明する。

$m_j^{(k)}$ ：k段階において，x_j 方向での可能な最大許容変化量

L_j ：x_j における下限

U_j ：x_j における上限

$$
p_j^{(k)} : \max \left\{ 1, \ \frac{m_j^{(k)}}{U_j - x_j^{(k)}} \right\}
$$

$$
q_j^{(k)} : \max \left\{ 1, \ \frac{m_j^{(k)}}{x_j^{(k)} - L_j} \right\}
$$

$$
\triangle^{+} \boldsymbol{x}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)} \quad (\boldsymbol{x} - \boldsymbol{x}^{(k)} \geqq o)
$$

$$
\triangle^{-} \boldsymbol{x}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)} \quad (\boldsymbol{x} - \boldsymbol{x}^{(k)} \leqq o)
$$

これらの各量を用いて，k段階における解が実行可能解となるための条件を表わすと

$$p_j^{(k)} \triangle^+ x_j^{(k)} + q_j^{(k)} \triangle^- x_j^{(k)} \leqq m_j^{(k)} \qquad (6)$$

$$i = 1, \quad 2, \cdots\cdots, \quad n$$

となる。

各k段階で線形計画問題を解く場合，スラック変数ならびに人為変数を導入することにより，与えられた条件式を等式化し初期実行可能解を求めるものとする。すなわち，

$$p_j^{(k)} \triangle^+ x_j^{(k)} + q_j^{(k)} \triangle^- x_j^{(k)} + v_j^{(k)} = m_j \qquad (7)$$

$$i = 1, \quad 2, \cdots\cdots, \quad n$$

$$\sum_{j=1}^{n} \frac{\partial h_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle^+ x_j^{(k)} - \sum_{j=1}^{n} \frac{\partial h_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle^- x_j^{(k)} + w_j^{(k)} = -h_i(\boldsymbol{x}^{(k)})$$

$$i = 1, \cdots\cdots, \quad m \qquad (8)$$

$$\sum_{j=1}^{n} \frac{\partial g_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle^+ x_j^{(k)} - \sum_{j=1}^{n} \frac{\partial g_i(\boldsymbol{x}^{(k)})}{\partial x_j} \triangle^- x_j^{(k)} + U_i^{(k)} + w_i^{(k)} = -g_i(\boldsymbol{x}^{(k)})$$

$$i = m+1, \cdots\cdots, \quad p \qquad (9)$$

において，人為変数w_i（$\geqq 0$）の和を最小化することにより，

$$\min \Sigma w_i \rightarrow 0 \Longleftrightarrow w_i = 0 \quad (i = 1, \cdots\cdots, \quad p)$$

となって実行可能解が求められる。

3. アルゴリズムの概要

ステップ1　問題の読込み

ステップ2　問題が線形計画問題であるかどうかの判定

(i)　線形計画問題であればＬＰサブルーチンで解を求め，ステップ6へ

(ii)　線形計画問題でないとき，ステップ3へ

ステップ3　初期実行可能解の設定

ステップ4　k段階において，問題(1)を線形化して(3)，(7)，(8)，(9)を作る

ステップ5　線形化された問題をＬＰサブルーチンで解き(5)を満足すればステップ6へ

満足しないときはk＝k＋1としてステップ4へ

ステップ6　停止

4. 結果の考察と今後の問題点

数理計画問題を一般的に均一の解法アルゴリズムにより解かせる事自体，その収束効率をかなり

無視したものと言えよう。しかしながら，ユーザとしては常にその専門家が解法ルーチンを利用するものではない事を考え合せると，多少の効率性を低下させようとも，とに角与えられた問題が解けるような解法アルゴリズムの存在は極めて必要とされる事は事実である。

非線形計画問題を線形化し，線形計画問題として解かせる場合，逐次的解法手順で解くわけであるが，最適解に近づいたところでかなり収束性が低下する傾向が出てくる。特にこの傾向は逐次解を実行可能領域に保持しようと思えば，より顕著に現われるものとされている。その意味で許容解の精度を緩和して最適解そのものではなく，最適近似解が求まれば良いと言う現実的な立場での解法アルゴリズムの構築と言う点では一応納得の行くものではないかと考えられる。

今後，収束性の向上と言う観点で二次近似法でのアルゴリズムにする事も検討するが，何と言っても線形計画法が普及し高速処理が可能となった現在，線形近似法による数理計画法の一般化は大いに有効なしかも素人のユーザでも簡単に利用出来る強靱なアルゴリズムとなり得るものと考える。

プログラム登録表

（様式１）

ライブラリ名	（センタ記入）	作成日付	60. 12. 27
プログラム名	UNMP		
作成者氏名	成　久　洋　之		
形式	ⓐ コンプリート・プログラム　　c 関数副プログラム b サブルーチン・副プログラム　　d その他		
使用機器	a 磁気テープ　入力（　　）本,　出力（　　）本 b CRTの種類（　　） c 磁気ディスク・ファイル（　　）個 d その他（　　）		
プログラムサイズ	a ソース・プログラム（ 460 ）行 b オブジェクト・プログラム（ 512 ）KW		
使用言語	ⓐ FORTRAN 77　c COBOL　e PL/1 b PASCAL　d LISP　f ASSEMBLER g その他（　　）		
処理形態	a バッチ処理専用　　ⓒ バッチ・TSS兼用 b TSS専用		
使用条件等			

（枠内に入りきらない場合は，Ａ４サイズのもので説明を添付してください）

岡山理科大学情報処理センター

岡山市理大町１－１

TEL (0862) 52－3012