疲労亀裂伝播寿命の確率分布の実験的決定法

（研究課題番号　02402038）

平成３年度科学研究費補助金（一般研究Ａ）研究成果報告書

平成４年３月

研究代表者　板　垣　浩

（横浜国立大学工学部生産工学科教授）

はしがき

本研究は平成２年４月より平成４年３月に至る２年間にわたって、文部省科学研究費（一般研究Ａ）の補助により行なわれたものである。研究課題は海洋構造部材の疲労亀裂伝播寿命の確率分布をいかに推定するかと言う問題である。疲労亀裂伸長線上の材料強度（疲労亀裂伝播に対する材料の抵抗）の確率分布を実験的に求め、統計的解析と、確率過程のシミュレーションの応用により、比較的小数の試験結果から、疲労寿命の確率分布関数を決定する方法について検討した。その結果、構造部材の疲労亀裂伝播寿命解析に非常に有効な方法を与えるものと言えよう。

研究組織

研究代表者	板垣　浩	（横浜国立大学工学部教授）
研究分担者	森下　信	（横浜国立大学工学部助教授）
研究分担者	石塚鐵夫	（横浜国立大学工学部助手）

研究経費

平成２年度	10,900千円
平成３年度	2,700千円
計	13,600千円

研究発表

（１）学会誌等

1) 板垣　浩、石塚鐵夫、関　隆広：”狭帯域ランダム荷重下の疲労亀裂伝播（バンド幅の影響）”、第１０回設計における信頼性シンポジュウム（JCOSSAR '90）、1990, p 89 - 94

2) 板垣　浩、石塚鐵夫、関　隆広：”ランダム荷重による疲労亀裂伝播、日本造船学会論文集、VOl. 168, １９９０, p 583 - 591

3) 板垣　浩、石塚鐵夫、関　隆広、藤波 貢：”疲労亀裂伝播への狭帯域ランダム荷重波形の影響, 第１１回設計における信頼性シンポジュウム（JCOSSAR '91）, 1991, p 411 - 418

4) 板垣　浩、石塚鐵夫, 金 善振：”材質の空間的変動の材料試験結果に及ぼす影響について, 日本造船学会論文集、VOl. 170, 1991, p 665 - 671

研究成果　疲労亀裂伝播寿命の確率分布の実験的決定法

1．はじめに

疲労破壊が安全性、信頼性に深く関与している場合には、材料の疲労特性の確率分布について、解析に用いるのに十分な情報が必要である。一般に、その様な情報を集めようとすれば、かなりの量の実験が必要である。ただ単に、平均値、分散等を知るだけでなく、材料特性、例えば疲労寿命、の確率分布関数を定めようということであれば、データ収集に要する時間と費用は多大なものとなろう。特に、Gumbelの極値統計学に在るような3母数のワイブル分布関数を適用しようとするならばなおさらである。とはいえ、信頼性解析には、材料特性の確率分布の推定は不可欠である。

材料あるいは構造物の疲労破壊寿命分布を知るには、統計的手法を用い寿命分布のすべてのパラメータを実験的に推定する方法があるが、多数の疲労試験の繰返しが必要である。このような試験は時間的かつ経済的制約から考えて現実的ではない。この欠点を多少でも補いうる方法の一つに、ベイズ統計法を用いた疲労寿命分布のパラメータを推定する方法がある。これは、過去に蓄積られたデータを利用し、自分の経験的判断も加え小数の疲労試験結果からパラメータを推定する方法である。この方法で推定した寿命の分布特性を用いた設計では、必然的に不確実要因を含んでいるので、安全係数を大きめにとらねばならないという欠点がある。

疲労亀裂伝播のシミュレーション方法は種々提案されている[1〜3]。　その一つに、いわゆるParis則[4]の疲労亀裂伝播係数Cと伝播指数mを確率変数として取り扱い、試験データからCの分布関数$F_C(c)$及びCとmの相関関係を定め、それよりCとmをシミュレートする方法がある。この方法で得られた亀裂長さ〜繰返し数関係曲線（以下a〜N曲線という）は、CとmはCとmは一定であるので、平滑な曲線となる。これは、多くの実験的事実と一致しないことが知られている。一方、伝播指数mは亀裂経路に沿って一定値であり、かつ伝播係数Cは空間的確率変数であると仮定し、a〜N曲線をシミュレーションする方法もあり、この方法で得られるa〜N曲線はジグザグな曲線となる。

上述した二つのシミュレーションに用いられる確率変数Cは全く異なるものであることに注意しなくてはならない。すなわち、後者のCはの平均値が前者のCに対応している。前者の方法を採る時は、Cの分布を定めるためにN個の標本を用いるとすれば、N本の試験片を必要とする。後者では、亀裂進展経路に沿ってk個の標本を採取するとすれば、N本の試験片からk×N個の標本が取れることになる。従って、前者のシミュレーション方法で疲労亀裂伝播寿命分布の推定を行なうとすれば、基礎データを得るために多数の疲労試験を必要とする。これに反し、後者の方法を用いる場合、小数の疲労試験ですむ可能性が十分であると考えられる。

筆者等の一人は先に微視的な研究から、亀裂伝播に関する材料の抵抗を近似的に正規（空間）確率過程として扱い疲労寿命を考察する方法を提案した[5]。それは、非常に微細な亀裂には適用しうるが、巨視的寸法（数mm〜数cm）の亀裂には適用しがたいことが分かっている。巨視的亀裂を扱うためには、材料抵抗のデータを相応する距離にわたって収集しなくてはならず、実験的困難を伴う。また、材料抵抗の分布は必ずしも正規確率過程として扱えるとは限らない等の難点もあった。しかしながら、後述するように、非ガウス過程の簡易シミュレーション方法がShinozuka等[6]によって開発されていること、応力拡大係数ΔK制御試験による実験によって、長い亀裂に沿って同一条件の下でのデータ収集が可能になっていることなどで、巨視的立場から、モンテカルロ法を適用した疲労寿命の推定が可能になると考えられる。

このような観点から、本研究では、伝播係数Cを空間的確率変数として扱い、比較的長い疲労亀裂について伝播抵抗の統計的解析モデルを仮定し、試験データから材料の統計的特性、例えば確率分布関数、自己相関関数あるはパワースペクトル密度関数を推定することを試みる。亀裂伝播線上の材料抵抗を、局所的亀裂速度として捉え、速度の変動を材料抵抗の変動に変換する。この変動を統計的に解析することによって、亀裂伝播抵抗の空間的確率過程としての特性を抽出する。そのためには、定常亀裂成長過程を実験的に実現する必要があり、自動的に亀裂長さを計測し応力拡大係数ΔKを制御しながら、一定間隔で亀裂成長速度を観測する必要がある。従って、効率良くΔKを制御しなが

ら亀裂成長を自動計測するための試験法について、ファジィ理論を応用したΔK制御および実験の高速化の観点から検討した。

一般に、疲労亀裂伝播試験は、一定荷重振幅条件で行われている。ＡＳＴＭ規格の中でも、試験はΔＰを一定とし、他の負荷条件も固定して行うことが望ましい、と定められている。しかし、亀裂伝播過程において、外部荷重状態が不変でも亀裂が伸長するため応力状態が変化した時の破壊条件ΔＫも変化するので、荷重(または変位）の制御のみで亀裂先端の破壊時の条件を制御することはできない。亀裂伝播が応力拡大係数Ｋによって律せられることがよく知られているので、疲労亀裂伝播についての情報を得る方法として応力拡大係数を一定に制御（ΔＫ制御）するような試験法が望ましい。

また、疲労亀裂伝播速度 da/dN にばらつきがあることは今日一般に認められているところであり、機械、構造物の安全性、信頼性を確保するためには、種々の不確定要因を設計、試験、保守等において考慮する必要があり da/dN のばらつきはこうした不確定要因の一つとして重要である。ΔＫ制御試験はこのｄａ/ｄＮのばらつきを調べる一つの方法としても知られている。

当研究室においても一定振幅荷重下およびランダム荷重下におけるΔＫ値制御疲労試験がいままで行なわれてきた[7]。 従来当研究室で行なわれて来た一定振幅荷重下のΔＫ値制御疲労試験においては、ΔＫを一定に保つため、ほとんど常時ΔＫを算出し、計算上でΔＫの誤差が０.１％以上になると目標のΔＫになるように修正していた。ΔＫはＡＤ変換によって計測された亀裂長さと荷重から求められておりコンピュータの計算のほとんど全てがこのＡＤ変換に費やされていて効率がよくない。もしΔＫがずれる時期を予測することが可能であるとしたらその時だけΔＫを修正すればＡＤ変換をほとんど常に実行する必要はなくなりコンピュータの効率化がはかれ、ＡＤ変換を実行しない時間に他の計算が可能となる。

ΔＫの修正時期を予測するためには da/dN の変動を事前に予測する必要がある。予想した da/dN とΔＫがずれるまでの亀裂長さとから次に修正するまでのサイクル数が求まるのである。過去に当研究室で行なわれた実験データによると、ΔＫ値制御試験では、ΔＫの修正と修正の間の da/dN は場所によってかなり変動している。これは試験片の材質のむらまたは計測誤差が原因であると思われる。しかし、da/dN は同一試験片内での平均の da/dN を上下するような形で変動していると思われる。例えばある時点において da/dN が急に速くなったとすると次は現時点の da/dN よりも遅くなる傾向があり、また、他の時点においてやや遅くなった場合次は少し速くなる傾向がある。

そこで、本研究ではこのような理論的な判断が難しいｄａ/ｄＮの変動をファジィ推論を導入することによって予測し、ΔＫの修正時期を決定することを試みる。

一方，疲労亀裂伝播試験は通常多大な時間と労力がかかり，亀裂伝播寿命の予測を確率的に論じるためにはそれに応じた量のデータ収集が必要となる。したがって疲労試験の高速化はより多くのデータの収集が可能となり，その解析の精度に大きく貢献すると思われる。本研究では亀裂長さの自動計測に関与する亀裂開口変位を比較的高周波数でも精度良く測定できる無接触光学式変位計を使用し，これまでの CODゲージ測定との比較などして，どの程度まで簡便な実験装置で実用にたえうるかについても検討した.

また、得られた材料の統計的特性を用い、疲労試験のシミュレーションにより疲労亀裂伝播寿命分布を推定し、疲労亀裂伝播に関する構造物の信頼性を検討する方法を論じる。

2. 疲労亀裂伝播則のパラメータの変動

疲労亀裂伝播挙動を表すパラメータである疲労亀裂伝播速度 da/dNは、一般に、パリス則[4]

$$\frac{da}{dN} = C(\Delta K)^m \qquad (1)$$

で表されることが多い。ここに、a は亀裂長さ、Nは繰返し数、ΔKは応力拡大係数範囲であり、C, m は材料定数とされている（以下、Cを伝播係数、mを伝播指数と呼ぶ）。

疲労亀裂伝播の統計的特性の解析モデルとしては、幾つかのモデルが挙げられるが、式（１）を用い解析モデルを構成する際に、伝播係数Ｃ と伝播指数 mの取扱い方は、次の四つの仮説に分けられる。すなわち

１）C, m ともに定数とする（確率モデル）[4]；

２）C, m ともに確率変数とする[2,3]；

３）m は一定として、Cのみ確率変数とする；

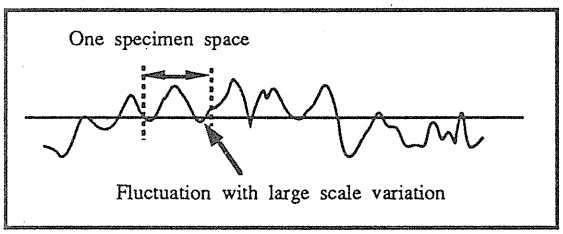
４）m のみ確率変数として、Cは定数とする。

このうち、１）は確定的モデルであるから、ここでは考えない。また、４）は例を見ない。

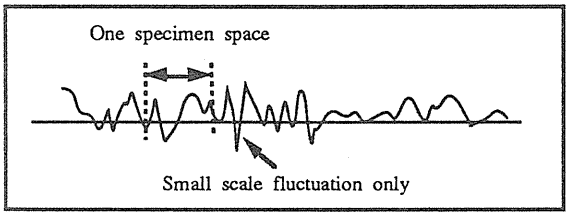
先の研究では、解析を簡単化するため伝播指数 m は亀裂経路に沿ったあらゆる場所で一定とし、伝播係数Cのみを空間的確率変数と仮定する問題を扱ったが、本研究ではmの変動をも考察の対象とすることとするので、さらにK_0なるパラメータを加えた次式を用いる。

$$\frac{da}{dN} = C\left(\frac{\Delta K}{K_0}\right)^m \qquad (2)$$

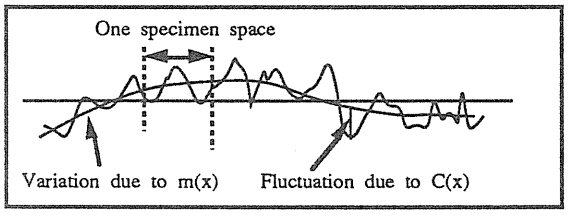
これは言うまでもなく指数 m を変えたときに得られるCの次元を一定にして、統計的取扱を可能にするためである。確定論的に材料定数 m と C との関係を論ずるときには式（1）でも差し支えはないが[2)]、確率変数としての C の値の確率密度などを検討するにはそれらが 同一母集団に属していなくてはならず、従って、少なくとも扱う次元は一定でなければならないからである。 応力拡大係数の振幅を一定にして充分な長さにわたって疲労亀裂伝播試験を行えば、図 2.1 - (a) または - (b) の様な結果が得られよう。



(a)



(b)



(c)

図 2．1 疲労亀裂伝播の変動概略図

-(a) 図は亀裂伝播率が細かく変動しながら全体としても大きく 変動している場合で、-(b) 図は 大きな変動がない場合である。図中の破線で囲った部分は通常使用されている試験片の寸法を表す。現実には図のような結果を得ることは不可能であるが、試験片毎の平均伝播率に著しい変動がなければ、-(b)図の場合で、そうでなければ - (a) 図 であろうと推定される。

いずれにしても C と m との空間的変動の様子が類似のものであればその変動を分離して解析することは困難である。破壊力学的には、Cは局所的な破壊に対する被害の限界値に対応するパラメータで、m は1サイクルで累積する被害と応力拡大係数との関係を表すパラメータと考えられるので、これらの確率的性格は大きく違うと考えられることも出来よう。例えば m を材料に依って決まる一定値とすることもある。しかし、m を一定として C のみが変動するとしても - (a) 図の様な場合には、実験的に変動の空間的相関などを知ることはできない。なぜならば、相関関数を求めるためのデータ間隔の最大値はせいぜい試験片幅の程度であるからである。最も解析のしやすいモデルは図 2.1 - (c) の様なものであろう。図に定性的に示したように、微小間隔で激しく変動する C に起因する変動と緩やかに変化する m による変動の重なった変動である。このようなモデルを想定すると、一試験片内では m は殆ど変化しない一定値をとり、細かく変動する C の試験片内での平均値もまたほぼ一定となると考えられる。 以下ではこのモデルの可能性について考えて行くこととする。なお、m が一定の場合はこのモデルの極端な場合である。

試験片 i の m_i、その試験片に負荷した応力拡大係数をΔK_i、亀裂先端 x における C を C(x) として、（2）式を書き直すと

$$\frac{da}{dN} = C(x)\left(\frac{\Delta K_i}{K_0}\right)^{m_i} \qquad (3)$$

さらに、亀裂伝播率及び C(x) の試験片内での亀裂線に沿った平均値をそれぞれおよびC_iとかくと

$$\left(\frac{da}{dN}\right)_i = C_i\left(\frac{\Delta K_i}{K_0}\right)^{m_i} \qquad (4)$$

となる。

式（2）の両辺の対数を取り,さらに期待値を取って

$$\overline{\log\left[\frac{da}{dN}\right]} = \overline{\log[C]} + \bar{m}\left\{\log[\Delta K_i] - \log[K_0]\right\} \qquad (5)$$

$\log\left(\frac{da}{dN}\right)$ の分散は

$$\sigma_v^2 = \sigma_{lc}^2 + \sigma_m^2\{\log[\Delta K] - \log[K_0]\}^2 + 2\sigma_{lcm}\{\log[\Delta K] - \log[K_0]\} \quad (6)$$

ここに、v = log[da/dN], lc = log[C] また、σ_{lcm}は lc と m との共分散を表す。

上式よりも、もしm に変動がなければ亀裂伝播率の変動はCの変動のみに起因し、従って、ΔKに依らないことが判る。もしも、C(x) の試験片についての平均値と、全体の平均値とがほぼ等しければ、$\log[C_i] = \overline{\log[C]}$（以下、これを$\log C_0$と記することにする）であるから、式（4）から

$$\log\left[\left(\frac{da}{dN}\right)_i\right] = \log[C_i] + m_i\{\log[\Delta K_i] - \log[K_0]\} \approx \log C_0 + m_i\{\log[\Delta K_i] - \log[K_0]\} \quad (7)$$

両辺の期待値を取って

$$\overline{\log\left[\left(\frac{da}{dN}\right)_i\right]} = \log C_0 + \overline{m}\{\log[\Delta K_i] - \log[K_0]\} \quad (8)$$

$\log\left[\left(\frac{da}{dN}\right)_i\right]$ の標準偏差をσ_{vi}とかけば、

$$\sigma_{vi} = \sigma_m|k_i - k_0| \quad (9)$$

ここに, $k_i - k_0 = \log[\Delta K_i] - \log[\Delta K_0]$

式（9）から、もしも、m_iが変動しなければ亀裂伝播率の試験片毎の平均値も変動しないことになる。逆に、C(x) の試験片毎の平均値が必ずしも一定でなくて、mが一定のときには、

$$\sigma_{vi} = \sigma_{Ci} \quad (10)$$

以上のような考察から、前述した4仮説のいずれが真実かを実験的に検証できよう。まず、幾種かの応力拡大係数振幅を用いて疲労亀裂伝播試験を数個の試験片ずつについて行って、da/dNのデータを収集し、log[da/dN]の分散、σ_v^2,を求めlog[ΔK]に対しプロットする。もしも、σ_v^2が0でなければ、仮説1）は成立し難い。もしもσ_v^2が応力拡大係数に依存するならば、仮説2）は否定されよう。 σ_v^2 が常に正であれば、仮説3）は有り得ない。1）、2）および3）に疑問があれば4）が最も妥当な考え方と言うことになろう。

また、da/dNの試験片毎の平均$\left(\frac{da}{dN}\right)_i$の対数 $\log\left[\left(\frac{da}{dN}\right)_i\right]$, の標準編差$\sigma_{vi}$を求めlog[ΔK]に対してプロットし、式（9）か式（10）の何れが実験に近いかを見ればmとCの空間的変動の様子について何らかの結論を得ることができよう。 ただし、いわゆるパリス則と呼ばれる式（1）が成立することを前提としるものであって、もし、他のの伝播則が成立する理由があるときは、その伝播則について考察し直す必要がある。

なお、実験的に得られる材料特性、ここではda/dN、はある範囲での平均値であるから、計測にさいしては一定長さだけ亀裂が伸長するに要するサイクル数を計測する必要がある。また、これは後述する材料特性の自己相関関数を求めるにも必要なことである。

3．ファジィ理論を用いたΔK値制御試験と実験の高速化

3．1　供試材と試験片

実験に使用した供試材は、海洋構造物用、高張力鋼BS4360　及びL36Eで何れも50kg級ＨＴであり、その化学成分を表3．1に、機械的性質を表3．2に示す。また、試験片形状および寸法は、図3．1に示すように,BS4360,L36Eの厚さがそれぞれ 30, 25mmの鋼板を削って, 18mmに仕上げた標準C.T（コンパクトテンション）型とした。

表3.1　化学成分　（%）

Steel	C	Si	Mn	P	S	Nb
BS4360	0.15	0.40	1.32	0.016	0.004	0.26
L36E	0.12	0.20	1.05	0.011	0.003	-

表3.2 機械的性質

Steel	Y. P	T.S	El.
BS4360	371MPa	522MPa	30 %
L36E	353MPa	530MPa	22%

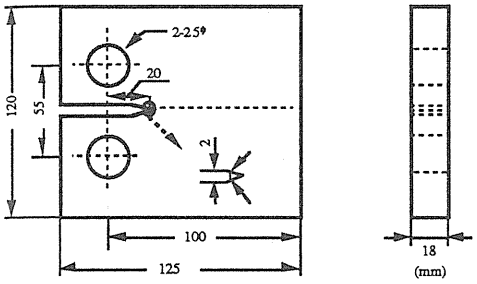


図３．１　試験片形状

３．２　従来のΔＫ値制御疲労試験システ[8,9)]

材料強度のばらつきによる”疲労亀裂伝播速度da/dNの変動”を調べる方法として、亀裂伝播長さ(a)と繰り返し数(N)の関係をグラフ上にプロットしてその変動を見るという方法がある。この場合、グラフが直線状になれば疲労亀裂伝播速度の変化をグラフから直接読みとることができるので、疲労亀裂伝播速度を一定にする試験方法が望ましい。従来のシステムを図３．２に示す。この図に示すように電気油圧サーボ式疲労試験機（ＭＴＳ社製）と２台のマイクロコンピューター(ＰＣ－９８００シリーズ）が本試験システムの主なものである。この２台のマイクロコンピューターにより、本試験システムはすべての操作を自動的に行なうことができる。システムの概略を以下に示す。

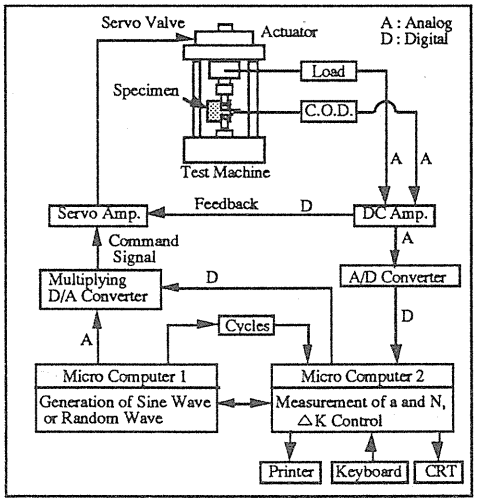


図３．２　ΔＫ値制御試験システム

マイクロコンピューター１により、荷重指令信号(正弦波）を発生させる。この信号と、次節で説明するマイクロコンピューター２で計算された乗数を、マルチプライングＤ／Ａコンバータへ入力し、アナログ信号に変換された荷重指令信号は、適当なレベルに調節され試験機へ出力される。そして、マイクロコンピューター２で、自動的に各種のデータを記録する。

切り欠き開口部に取り付けられたＣ.Ｏ.Ｄゲージによって測られた亀裂開口変位（crack opening displacement：Ｖ）と、ロードセルから検出された荷重（load：Ｐ）がマイクロコンピューター２へ入力される。マイクロコンピューター２は、この変位と荷重（Ｖ/Ｐ）の値を計算し、次式(11)、(12)を用いて亀裂長さａを算出する。

$$\frac{a}{W} = c_0 + c_1(U_x) + c_2(U_x)^2 + c_3(U_x)^3 + c_4(U_x)^4 + c_5(U_x)^5 \qquad (11)$$

$$U_x = \frac{1}{1 + \sqrt{\frac{BEV}{P}}} \qquad (12)$$

Ｃ0＝1.0010：Ｃ1＝-4.6655：Ｃ2＝18.460

Ｃ3＝-236.82：Ｃ4＝1214.9：Ｃ5＝-2143

ａ：crack length (mm)　Ｗ：width of specimen (mm)

Ｂ：thickness of specimen (mm)

Ｅ：Young's modulus (kgf/mm2)

Ｖ：crack opening displacement (mm)　Ｐ：load (kgf)

この理論式の亀裂長さａとＶ／Ｐの関係は、図３．３(BS4360) に示す様に、ΔＰ値一定の試験結果とよく合っている。

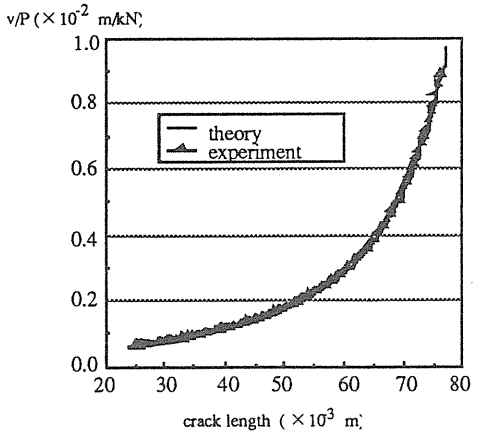


図３．３　a-v/P 線図(BS4360)

このようにして求められた亀裂長さより、ΔK値は式(13)（ＡＳＴＭ規格による）を用いて求めることができる。

$$K=\frac{P}{B\sqrt{W}}\frac{2+\frac{a}{W}}{\left(1-\frac{a}{W}\right)^{\frac{3}{2}}}\left\{C_0+C_1\left(\frac{a}{W}\right)+C_2\left(\frac{a}{W}\right)^2+C_3\left(\frac{a}{W}\right)^3+C_4\left(\frac{a}{W}\right)^4\right\} \quad (13)$$

ここで、

C_0=0.886：C_1=4.64：C_2=-13.32：C_3=14.72：C_4=-5.6

a：crack length (mm)

W：width of specimen (mm)

B：thickness of specimen (mm)　　P：load (kgf)

K：stress intensity factor (kgf/mm$^{3/2}$)

$$\Delta K=K_{max}-K_{min} \quad (14)$$

なお,V／Pを求めるためのデータは、連続した２周期からVおよびPを500ポイント、サンプリングする。そして最小２乗法（least squre error method）によりV／Pを求める。このようにして得られた10個のサンプルの平均値をV／Pのデータとして用いる。

このような過程をマイクロコンピューター２が終始連続的に行なう。もし、式(13)により得られたΔK値が規定のΔK値より、0.1%以上ずれたら、マルチプライングD／Aコンバータの乗数を調節する事によって自動的にΔK値の制御を行なう。

3.3　データの記録

亀裂長さの計測に関しては、一定長さだけ亀裂が伸びるのに要するサイクル数などのデータを計測する必要がある。

一枚の試験片でできるだけ多くのデータを収集するためには、Δａをできる限り小さくすることが望ましいが、Δａは試験システムの測定精度とマイクロコンピュータの計算精度などによって制限される。当研究室では、亀裂長さが0.4mm増加したときに、制御コンピュータが、亀裂長さａ、荷重の繰り返し数N、その時点での荷重振幅を求めるための最大及び最小荷重（P_{max}, P_{min}）、亀裂開口変位ｖ等をデータとして自動的にディスクとプリンターに記録するようにした。

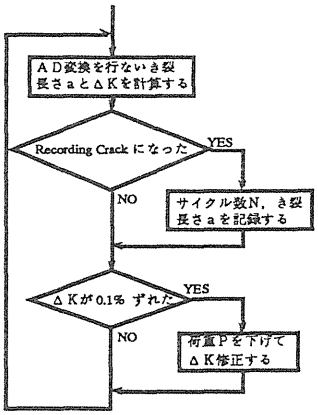


図３．４　従来の実験プログラムのフローチャート

３．４　ファジィ推論法による制御

(1)　ファジィ推論法適用の可能性

ΔＫを制御するには、ΔＫの計算、そして修正が必要となる。ΔＫの計算にはそのときの亀裂長さ（ａ）と荷重振幅（ΔＰ）を用いる。亀裂長さが伸びるにつれΔＫは徐々に増加する。よってΔＫをある精度で一定に保つため今までは、先に述べたようにほとんど常にΔＫの計算を行いΔＫがある範囲（０．１%）を越えたら修正してきた。しかし、もし、次にΔＫがずれるのに要する亀裂長さに達するまでのサイクル数（Ｎ）が予想できれば常にΔＫを計算する必要はなくなる。サイクル数を予測するには亀裂伝播速度 da/dN の予測が必要となる。da/dN と次にΔＫがずれるのに要する亀裂長さΔａが与えられると次に修正するまでのサイクル数N_fが求められるのである。da/dN が一定であれば次に修正するまでのサイクル数N_fは常に一定でよいのであるが、da/dN のばらつきを調べるためにΔＫ値一定試験を行うのであるから da/dN は当然のことながらばらつく。図3.5　に示すように過去に当研究室で行われたΔＫ値一定試験のデータ（BS4360）によると、ばらつきにいくつかの傾向があることがわかる。たとえばあるときda/dNが急激に増加した後はつぎは急激に減少したり、da/dNが急激に減少した後はつぎは急激に増加したりする。また、da/dNはだいたいにおいて上下に変化を繰り返していることにも気付く。このような傾向から、現時点までのda/dNから次にΔＫを修正するまでのda/dNが予想できれば次に修正するまでのサイクル数N_fの予測は可能となる。そこで、本研究ではda/dNの傾向を把握するというようなあいまいさを含むものを処理するためファジィ理論の適用を試みた。

（2）　ファジィ推論式の作成

da/dNを予想するためのファジィ制御規則を作成するにあたり，過去に当研究室で行われたΔK値制御疲労試験の実験データから亀裂長さaが0.2mm伸展する間の平均の亀裂伝播速度da/dNを調べた（図3.5）。

ここで、亀裂長さaの伸展が0.2mmというのは、本研究においてのΔK値の修正時期は設定値に対して0.5％ずれた時を目安としていて、この時の亀裂長さaの伸展がほぼ0.2mmであるためである。

図3.5によると、ある時点でda/dNが増加すればおおむね次はda/dNが減少していて、反対にda/dNが減少すると次は増加している。

さらに、da/dNの増減の度合いも次のda/dNに影響しているようである。

そこで、過去に当研究室で行なわれてきたΔK値制御試験のデータから、$(da/dN)_{i-1}$と$(da/dN)_i$の変化の割合$(d^2a/dN^2)_i = \{((da/dN)_i - (da/dN)_{i-1}\} / (dN_i - dN_{i-1})$に対して、$(da/dN)_{i+1}/(da/dN)_i$がどのようになるかをΔK=70, 110 ,140 [kgf/mm3/2]の3つの実験データについて3－Dグラフで図3.6(BS4360)に示した。

図3.6の断面図を図3.7にそれぞれ示す。

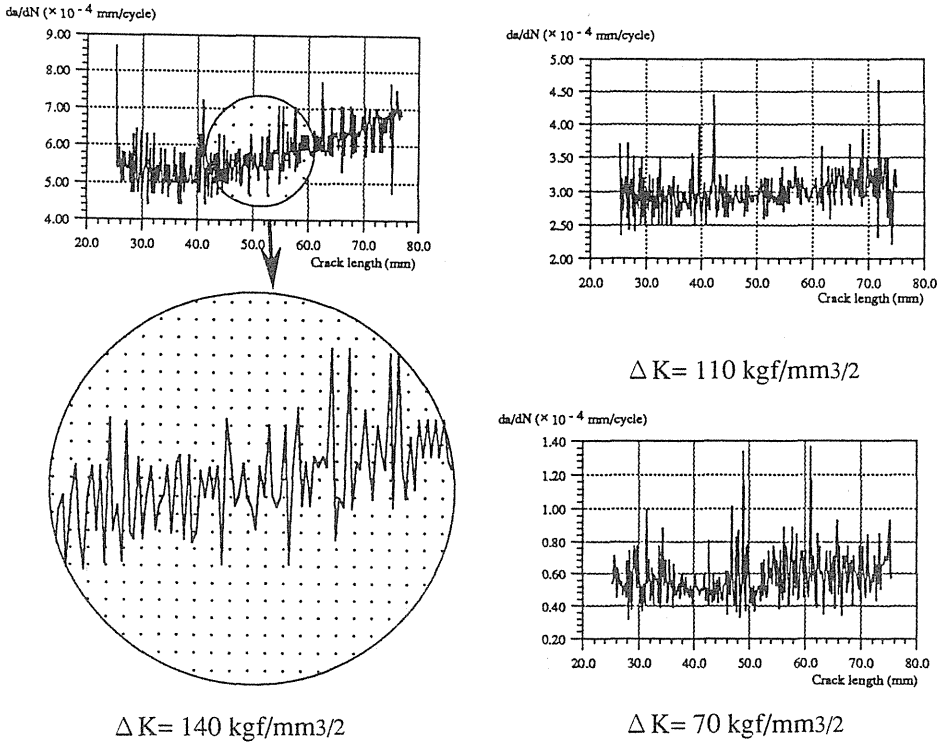


図 3．5 da/dNのバラツキ

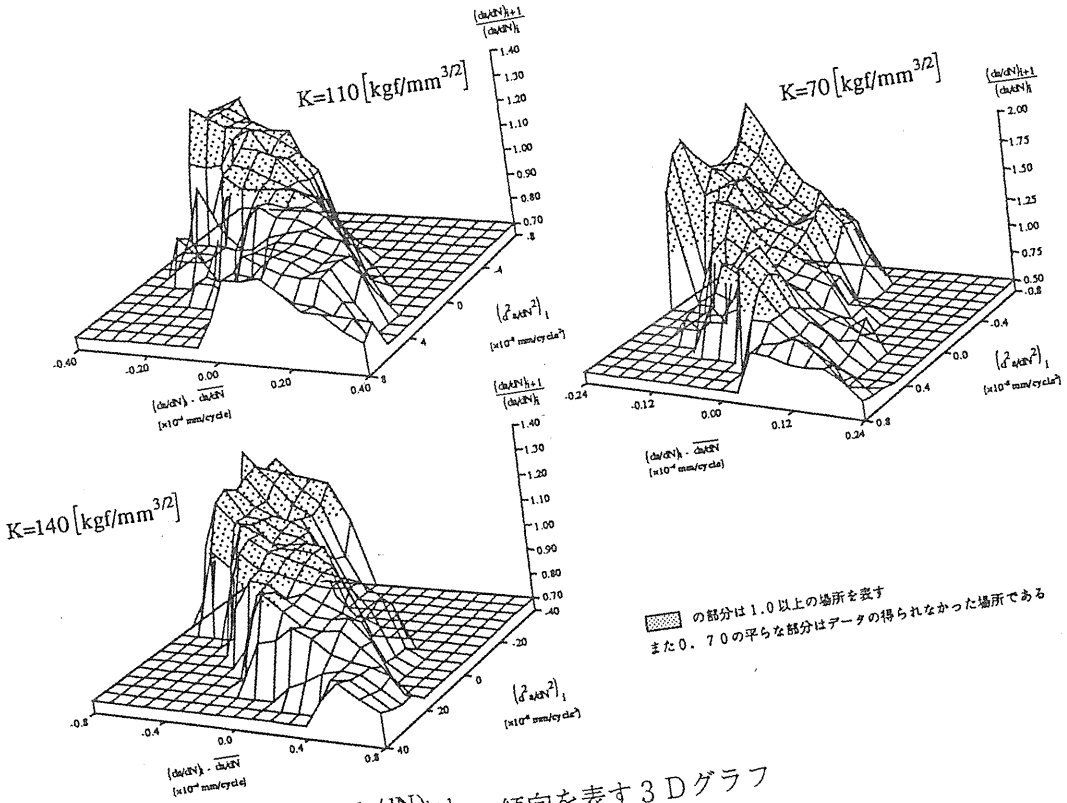
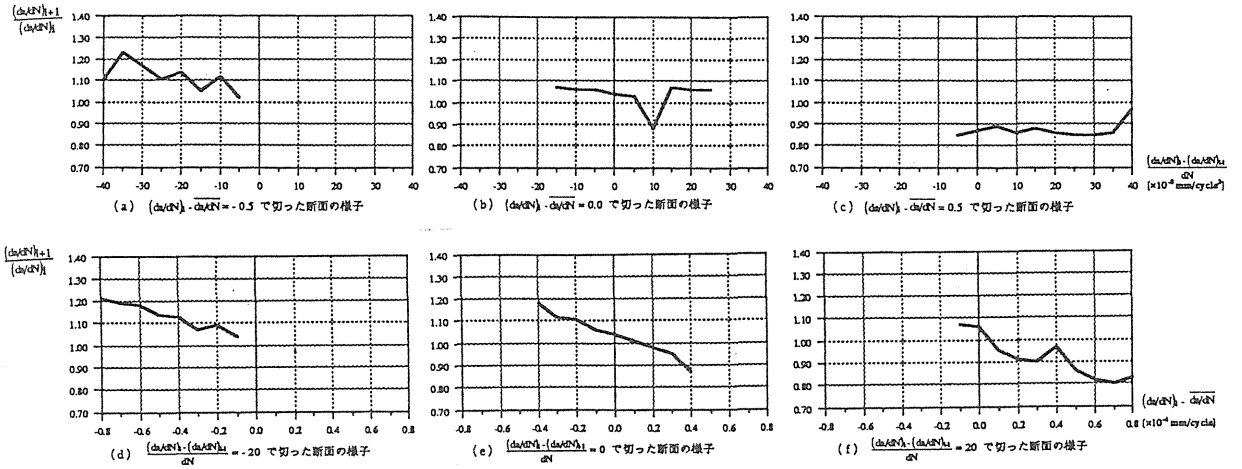
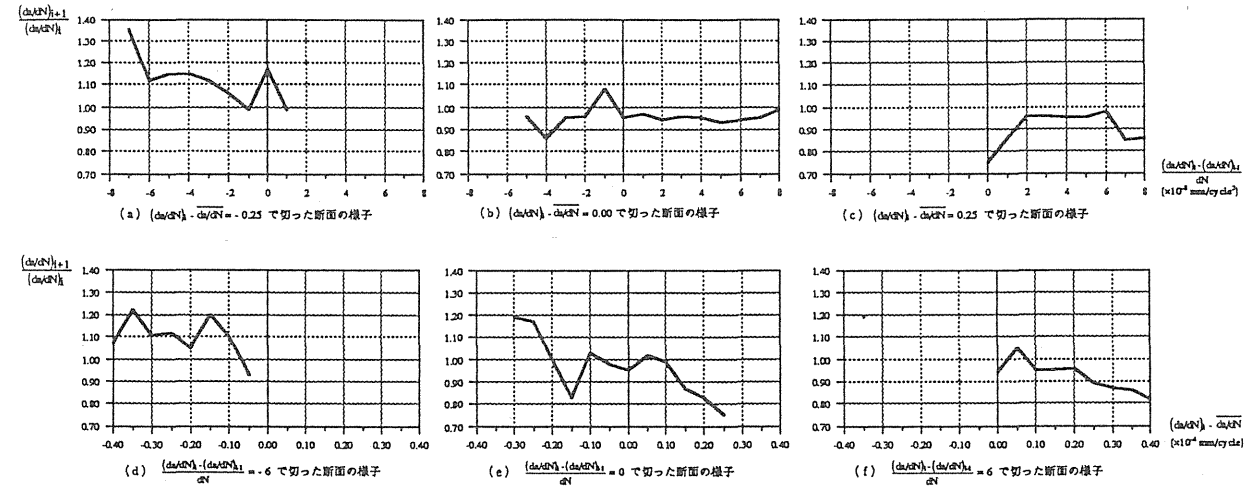


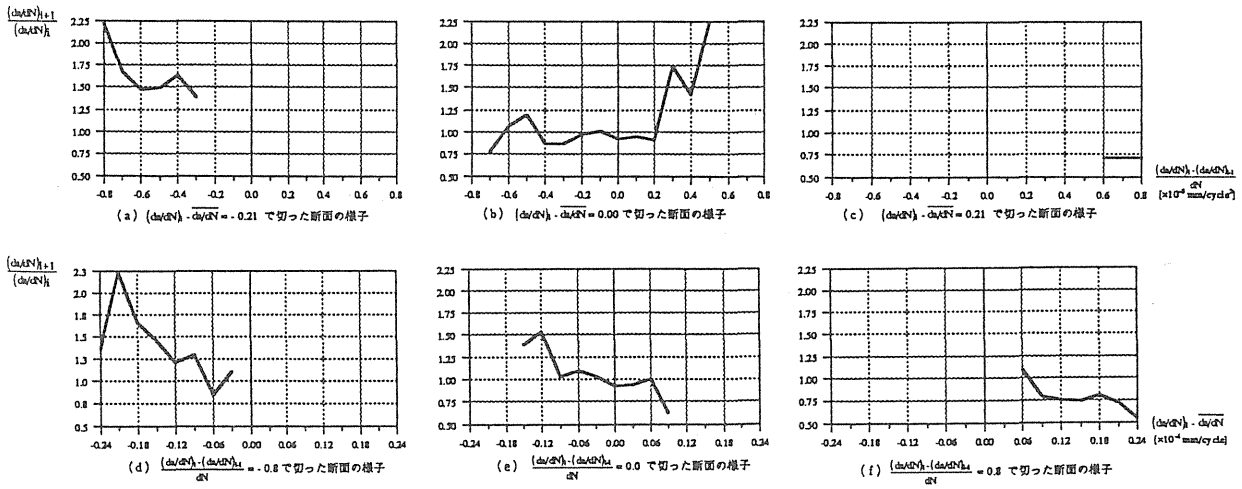
図３．６ $\frac{(da/dN)_{i+1}}{(da/dN)_i}$ の傾向を表す３Ｄグラフ



Δ K=140 kgf/mm3/2



Δ K=110 kgf/mm3/2



Δ K=70 kgf/mm3/2

図３．７　３－Ｄグラフの断面図

図３．６、図３．７から、例えばおおむね次のことがいえる。

＊ $(da/dN)_i$ が $\overline{da/dN}$ に対してかなり遅くてなおかつ$(d^2a/dN^2)_i$が負のとき、$(da/dN)_{i+1}/(da/dN)_i$ は１よりもかなり大きくなる。

＊ $(da/dN)_i$が$\overline{da/dN}$に対してかなり遅くてなおかつ$(d^2a/dN_2)_i$がほぼ零のとき、$(da/dN)_{i+1}/(da/dN)_i$ は１よりもやや大きくなる。

＊ $(da/dN)_i$が$\overline{da/dN}$ とほぼ同じでなおかつ $(d^2a/dN^2)_i$ が零のとき、$(da/dN)_{i+1}/(da/dN)_i$ はだいたい１になる。

上記の例などからファジィ推論規則を作成し、$(da/dN)_{i+1}/(da/dN)_i$ の推論値ごとにｉｆ～ｔｈｅｎ形式の推論文を用いて整理したのが表３．３である。略号の意味は表３．４に示す通りである。

ここで亀裂伝播速度 $(da/dN)_i$ が $\overline{da/dN}$ に対してかなり速い、ほぼ同じ、かなり遅いといった漠然とした表現は、メンバーシップ関数を定義することによって定量的に評価できるようになる。本研究では、これを図３．８-ａ～図３．８-ｃのように決めた。

亀裂伝播速度 $(da/dN)_i$ については、過去１２回のΔＫの修正と修正の間の $(da/dN)_i$ $(i = i\text{-}12,i\text{-}11,\cdots,i\text{-}1)$ を小さい値から順に並べたときの真中４つの平均値を中位としてMediumと表わし、Slow, Medium, Fast の３つのグレードで評価する。なお、図中の s_1 は最も小さい４つの $(da/dN)_i$ の平均値で、average はその次の４つの平均値、fa は残り４つの平均値とした。

表３．３　（ｄａ／ｄＮ）のファジィ規則

Case	Condition	Reasoning
1-a	If { $(da/dN)_i$ is SL and $(d^2a/dN^2)_i$ is NE }	then r is VL
2-a	If { $(da/dN)_i$ is SL and $(d^2a/dN^2)_i$ is ZO } or	
2-b	{ $(da/dN)_i$ is SL and $(d^2a/dN^2)_i$ is PO }	then r is LA
3-a	If { $(da/dN)_i$ is ME and $(d^2a/dN^2)_i$ is NE } or	
3-b	{ $(da/dN)_i$ is ME and $(d^2a/dN^2)_i$ is ZO } or	
3-c	{ $(da/dN)_i$ is ME and $(d^2a/dN^2)_i$ is PO } or	
3-d	{ $(da/dN)_i$ is FA and (d^2a/dN^2) is NE }	then r is ST
4-a	If { $(da/dN)_i$ is FA and $(d^2a/dN^2)_i$ is ZO } or	
4-b	{ $(da/dN)_i$ is FA and $(d^2a/dN^2)_i$ is PO }	then r is SM

表３．４　略号とその意味

Symbol	Meaning	Symbol	Meaning	Symbol	Meaning
				r	$(da/dN)_{i+1}/(da/dN)_i$
SL	Slow	NE	Negative	VL	Very Large
ME	Medium	ZO	Zero	LA	Large
FA	Fast	PO	Positive	ST	Standard
				SM	Small

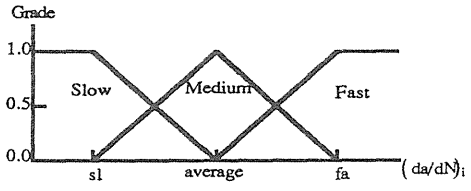


図３．８-ａ　(da/dN)i のメンバーシップ関数

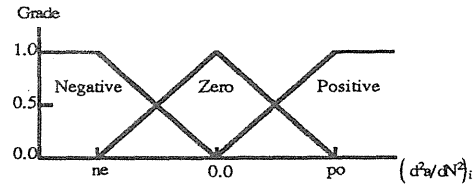


図３．８-ｂ　(d2a/dN2)i のメンバーシップ関数

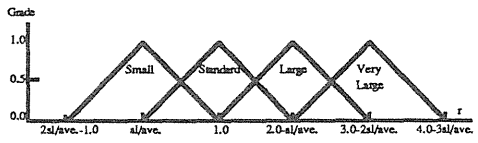


図３．８-ｃ　推論値ｒのメンバーシップ関数

亀裂伝播速度の変化の割合 $(d^2a/dN^2)_i$ は、Negative , Zero , Positive の３つのグレードに分類した。図中の ne は過去１２回のΔＫ値の修正時における亀裂伝播速度の変化の割合 $(d^2a/dN^2)_i$ $(i = i\text{-}11,i\text{-}10,\cdots,i\text{-}1)$ の最も小さいもの４つの値の平均値、po は最も大きい値４つの平均値とした。

亀裂伝播速度の比 $(da/dN)_{i+1}/(da/dN)_i$ の推論値については、Small , Standard , Large , Vere Large の４つのグレードに分類した。

表３．５　亀裂長さ（ａ）の伸展の実例

き裂長さａ [mm]	繰り返し数N [cycle]	da/dN [×10^{-4} mm/cycle]	d^2a/dN^2 [×10^{-8} mm/cycle^2]
48.27	98771		
48.49	99603	2.64	
48.75	100722	2.32	-2.87
48.90	101594	1.72	-6.92
49.21	102700	2.80	9.79
49.39	103527	2.18	-7.57
49.62	104547	2.25	0.77
49.86	105662	2.15	-0.92
50.10	106715	2.27	1.20
50.38	107827	2.52	2.15
50.60	108937	1.98	-4.83
50.83	109881	2.44	4.81
51.08	110987	2.26	-1.59
51.30	111892	2.43	1.88

具体的に $(da/dN)_{i+1}/(da/dN)_i$ ＝ｒの推論手順を示すと次のようになる。例えば亀裂長さ(ａ)と繰り返し数(N)が表３．５のようになっていて、亀裂長さａ＝51.08[mm]、繰り返し数Ｎ＝110987[cycles]のΔＫの修正を行なった時点では

ｉ）da/dNのメンバーシップ関数は　図３．８－ａにおいて、

sl ＝ (1.72+1.98+2.15+2.18)/4＝ 2.0

me ＝ (2.25+2.26+2.28+2.32)/4＝ 2.3

fa ＝ (2.52+2.44+2.64+2.80)/4＝ 2.6

となる。

ii）d^2a/dN^2のメンバーシップ関数は図３．８－ｂにおいて、

ne＝{-7.57+(-6.92)+(-4.83)+(-2.87)}/4＝ -5.6

po ＝ (1.20+2.15+4.81+9.79)/4＝ 4.5

となる。

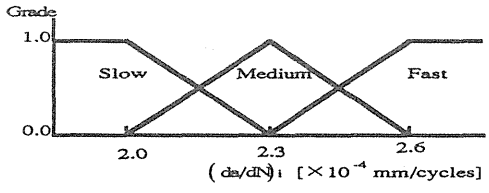


図３．９－ａ　$(da/dN)_i$のメンバーシップ関数の例

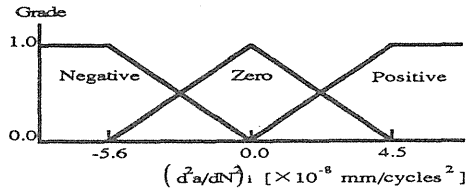


図3．9-b　$(d^2a/dN^2)_i$のメンバーシップ関数の例

da/dN = 2.26 [×10-4mm/cycles] , d2a/dN2 = -2.72 [×10^{-8}mm/cycles]　であるので、図３．９－ａ、ｂよりda/dN、d^2a/dN^2のメンバーシップ関数は次のようになる。da/dNに関しては Slow, Medium, Fastそれぞれに対する帰属度を$\mu SL\{(da/dN)_i\}, \mu ME\{(da/dN)_i\}, \mu FA\{(da/dN)_i\}$とし、$d^2a/dN^2$に関しては、Negative,Zero,Positiveそれぞれに対する帰属度を$\mu NE\{(d^2a/dN^2)_i\}, \mu ZO\{(d^2a/dN^2)_i\}, \mu PO\{(d^2a/dN^2)_i\}$として

$$\mu SL\{(da/dN)_i\} = 0.2, \quad \mu ME\{(da/dN)_i\} = 0.8,$$

$$\mu FA\{(da/dN)_i\} = 0.0, \quad \mu NE\{(d^2a/dN^2)_i\} = 0.3,$$

$$\mu ZO\{(d^2a/dN^2)_i\}=0.7, \quad \mu PO\{(d^2a/dN^2)_i\}=0.0 \qquad (15)$$

となる。

表3.3のファジィ規則で式（15）に関係するのはCase 1-a, 2-a, 3-a および3-b である。

Case 1-a より

If$\{(da/dN)_i$ is SL and $(d^2a/dN^2)_i$ is NE$\}$, then r is VL $=(0.2 \wedge 0.3) \wedge \mu VL(r) = 0.2 \wedge \mu VL(r)$

$=\mu VL^*(r)$ (16)

Case 2-a より

If$\{(da/dN)_i$ is SL and $(d^2a/dN^2)_i$ is ZO$\}$, then r is LA $=(0.2 \wedge 0.7) \wedge \mu LA(r) = 0.2 \wedge \mu LA(r)$

$=\mu LA^*(r)$ (17)

Case 3-a, b より

If$[\{(da/dN)_i$ is ME and $(d^2a/dN^2)_i$ is NE$\}$ or $\{(da/dN)_i$ is ME and $(d^2a/dN^2)_i$ is ZO$\}]$, then r is ST$=\{(0.8 \wedge 0.3) \vee (0.8 \wedge 0.7) \wedge \mu ST(r)$

$=(0.3 \vee 0.7) \wedge \mu ST(r)$

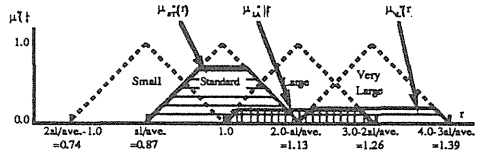
$=0.7 \wedge \mu ST(r) = \mu ST^*(r)$ (18)

上式で∧、∨は min , max 演算を表す。

結局、ｒの最終的なメンバーシップ関数は次式で表せる。

$$\mu^*(r) = \mu VL^*(r) \vee \mu LA^*(r) \vee \mu ST^*(r) \quad (19)$$

図３．１０はこれを図示したもので、ｒのファジィ推論値は３つの斜線を施した部分の重心位置（Ｇ）におけるｒで与えられることとする。



(G=) r = 1.1

図３．１０　推論の一例

$(da/dN)_{i+1}/(da/dN)_i$ が推論できれば、次回のΔＫの修正時期までの予想サイクル数 $(N_f)_{i+1}$ は(Ne)iに対して

$$(N_f)_{i+1} = FACTOR \times (Ne)_i, \quad FACTOR = 1/r \quad (20)$$

と計算することができる。ここで $(Ne)_i$ は $(da/dN)_i$ と次にΔＫが0.5%ずれるときまでの亀裂の増分Δａから次式から計算される。

$$(Ne)_i = \Delta a / (da/dN)_i \quad (21)$$

ここで、Δaは（１３）式を変形して

$$\Delta a = C_0 + C_1 a + C_2 a^2 + C_3 a^3 + C_4 a^4 + C_5 a^5,$$

$$C_0 = 0.13075,\ C_1 = 2.9724 \times 10^{-4},$$

$$C_2 = 5.7223 \times 10^{-4},\ C_3 = -2.1155 \times 10^{-5},$$

$$C_4 = 2.6348 \times 10^{-7},\ C_5 = -1.1273 \times 10^{-9} \quad (22)$$

上式よりa=51.08[mm]のときΔa=0.23、$(da/dN)_i$=2.26×10^{-4}[mm/cycle]のとき$(Ne)_i$＝1006[cycle]、FACTOR = 1/r=1/1.1=0.91より$(N_f)_{i+1}$=FACTOR×$(Ne)_i$=0.91×1006=915[cycle]となる（実際のFACTORは、0.93だった）。

(3) 実験手順

本研究の試験システムは３．２節に述べた従来のシステムとほとんど変りはないが、２台のマイクロコンピュータを１つにまとめ１台で荷重指令信号を発生させ、ａとＮを記録し、ΔＫをコントロールできるようにした。なおv/Ｐを求めるためのデータとして連続した２周期からｖおよびＰを５０ポイントサンプリングし、最小２乗法によりｖ／Ｐを求める。そしてこのように得られた１０個のサンプルの平均をｖ／Ｐの値とする。

従来のΔＫ値制御法は常時ΔＫを計算し、ΔＫが規定の値より０．１％以上ずれたらマルチプライングＤ／Ａコンバータの乗数を調節する制御法であった。しかしΔＫのずれが０．５％以内であればda/dNのずれに換算すると１～２％程度であり、da/dNの変動を調べる際にはほとんど影響がないと考えられる。そこで本研究で行なった制御法ではΔＫが規定の値より０．５％ずれるまでのサイクル数N_fを前述したファジィ推論法で予想し、その時にだけΔＫを修正することによりコンピュータの効率化を図った。

da/dNの変動を調べるためのデータとして、亀裂が0.4mm増加したときの亀裂長さａ、荷重の繰り返し数Ｎ、荷重振幅ΔＰ、亀裂開口変位ｖ等をデータとして自動的にディスクとプリンターに記録するようにした。

正確に0.4mmごとのデータを記録するためには、Recording crack （0.4mm 毎）の少し手前からRecording crack になるまで亀裂長さａを計測し続ける必要がある。それには、同一試験片で最も速く0.4mm伸展するサイクル数をあらかじめ求めておかなければならない。当研究室で今まで行ったいくつかの実験データから

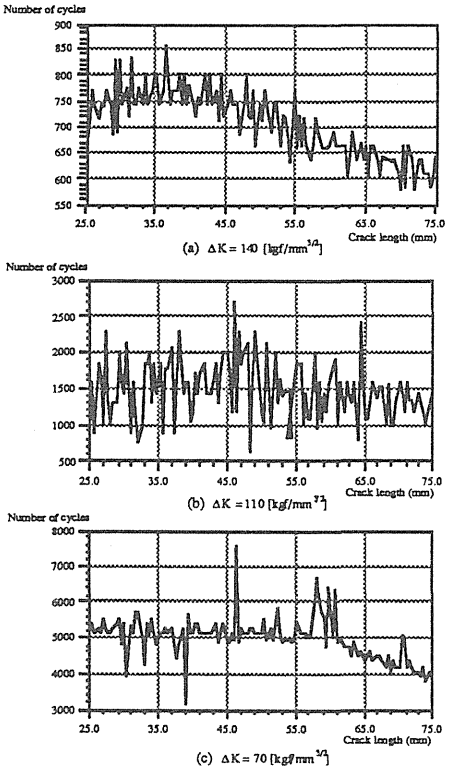


図３.. １１亀裂が0.4mm伸長するのにかかるcycle数

0.4mm伸展するまでのサイクル数を図３．１１に示す。これを検討してみると亀裂長さが25.0mm～28.0mmにあるときの0.4mm伸展するまでのサイクル数の平均に対して、最速の 0.4mm 伸展するのに要するサイクル数はそれの６０％以上であることがわかった。

このような理由から本研究では亀裂長さが 25.0mm～28.0mmにあるときに毎回ａとＮを測りΔＫを計算してΔＫの修正と修正の間の da/dNを１２回求め、それを平均したものを 0.4mm伸展するのに要するサイクル数に換算した値に０．６を乗じたサイクル数をＮｒとし、前回のRecording crackからＮｒ経過した時点から毎回亀裂長さａの計測を行うことにする。

実験プログラムは付録に示し、そのフローチャートを図３．１２に示す。

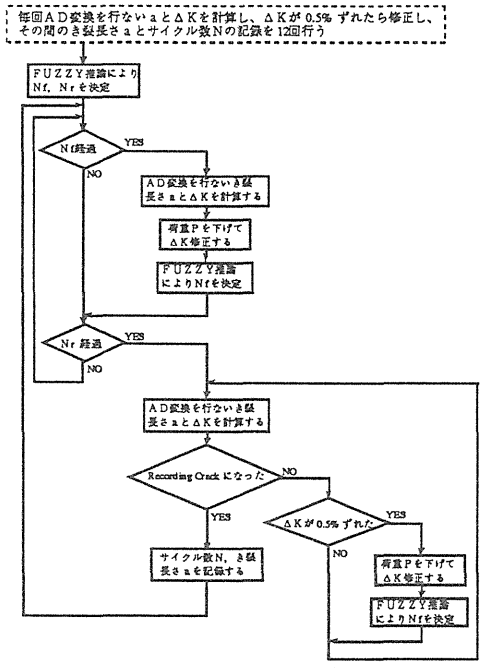


図３．１２　荷重制御プログラムのフローチャート
（実際のプログラムは付録に示す。）

（４）試験結果と考察

本研究では、ΔＫ＝110,140 [kgf/mm$^{3/2}$] でそれぞれ１枚の試験片について実験を行なった。

(a) 亀裂長さａと繰り返し数Ｎの関係

ａ－Ｎ線図を図３．１３に示す。この図からも分かるようにａとＮの関係はほぼ直線的ではあるが、微視的には細かく変動していることから同一試験片内でも亀裂伝播速度（da/dN）がばらついていることがわかる。

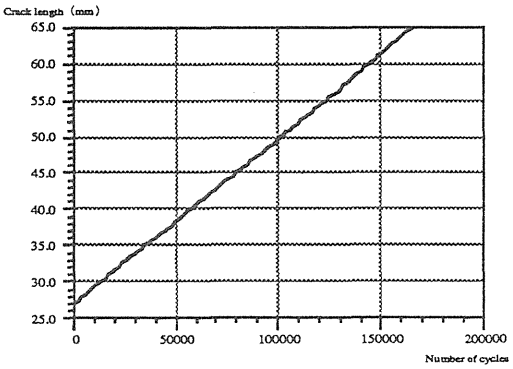


図３．１３-１　ａ－Ｎ曲線
（ΔＫ＝１１０［ｋｇｆ／ｍｍ３／２］）

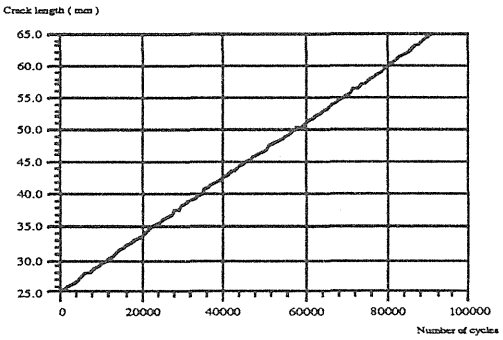


図３．１３-２　ａ－Ｎ曲線
（ΔＫ＝１４０［ｋｇｆ／ｍｍ３／２］）

(b) 応力拡大係数範囲ΔＫと亀裂伝播速度da/dNの変動

図３．１４にΔＫと da/dN の変動をａに対してプロットした。この図から分かるように、ΔＫはよく制御されているが、da/dNは大きく変動している。

(c) ファジィ推論法の適用に対する検討

図３．１４のΔＫの変動を拡大して示したのが図３．１５である。この図から分かるように、ΔＫは制御の目標であるばらつき０.５％以内にほぼ収まっている。

ファジィを用いた予測を実際の結果と比較してみると、図３．１６に示すようにおおまかに見て da/dN（またはΔＫが 0.5% ずれるまでのサイクル数）の変動の傾向を捉えいるように思える。

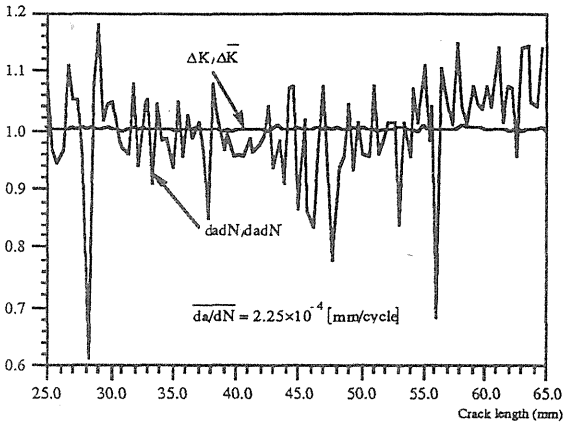


図３．１４－１　亀裂伝播経路に沿ったda/dNの変動（ΔK=110 kg/mm$^{3/2}$）

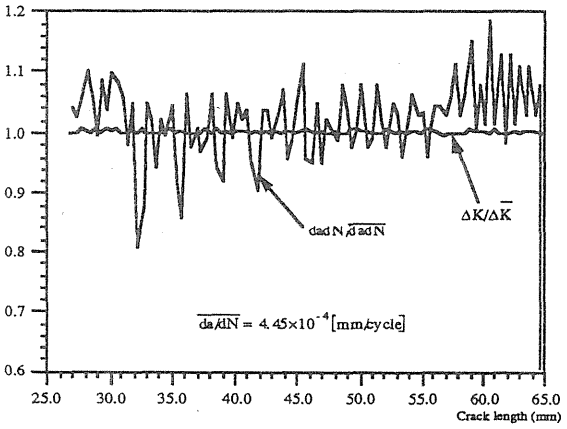
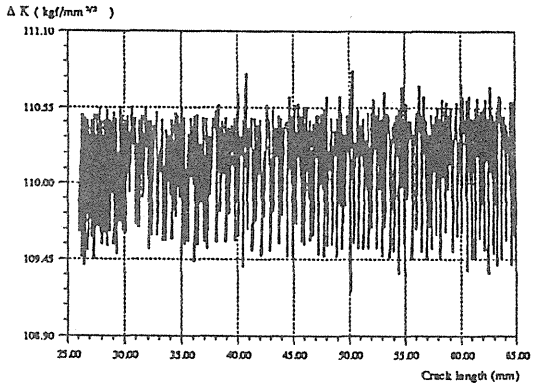


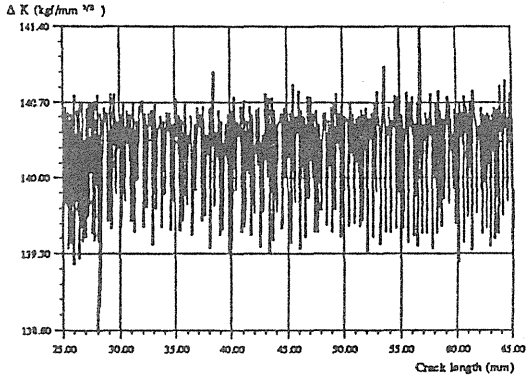
図３．１４－２　亀裂伝播経路に沿ったda/dNの変動（ΔK=140 kg/mm$^{3/2}$）

$(N_f)_i/(Ne)_{i-1}$と$(Ne)_i/(Ne)_{i-1}$はある程度同じような変動をしており、両方が＋側または両方が一側にある場合はΔK=110[kgf/mm 3/2]で全体の75%、ΔK=140[kgf/mm 3/2]では約 80% 存在し、ΔKが 0.5% ずれるまでのサイクル数の予想が前回のそれより減るか増えるかという点に関してはファジィでの予想はだいたい当たっていた。しかしそれは da/dN（またはΔKが 0.5% ずれるまでのサイクル数）がジグザグに上下しているときに限っている場合がほとんどであり、もし２回続けて増加したり減少したときにはほとんどの場合はずしている。このことに関しては da/dN の動向を完全に把握することの困難さから来るもので、やむを得ないことであると考えられる。また、図３．１７に $(Ne)_i/(Ne)_{i-1}$ と $(N_f)_i/(Ne)_{i-1}$の関係を示す。これを見るとプロットしたデータの多くは直線$(N_f)_i/(Ne)_{i-1}=(Ne)_i/(Ne)_{i-1}$ の周辺に分布している。直線から大きくはずれている点もいくつか見られるが、そのほとんどは$(N_f)_i/(Ne)_{i-1}$の方が$(Ne)_i/(Ne)_{i-1}$ よりも小さい値となっている。これは、実際にΔKが 0.5% ずれる時点よりも早めにΔKを修正することを意味しているので安全側であると言える。

ファジィ推論法を適用した制御で行なった実験のＡＤ変換の回数（ａとΔＫの算出回数）はΔＫ＝140 [kgf/mm$^{3/2}$] の場合今までの制御の49.1%、ΔＫ＝110 [kgf/mm$^{3/2}$] の場合には42.9%になり半分以下に減った。これについては、Nr （亀裂長さ計測開始サイクル数）について検討していけば今以上にＡＤ変換の回数を少なくすることができいっそうコンピュータの効率化が計れるであろう。



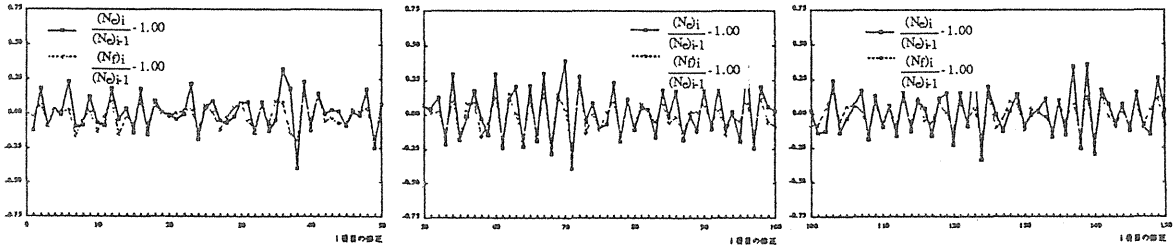
ΔK = 110 kgf / mm$^{3/2}$



ΔK = 140 kgf / mm$^{3/2}$

図3.15 疲労試験中のΔKの変動

$\Delta K = 110\ kgf / mm^{3/2}$



$\Delta K = 140\ kgf / mm^{3/2}$

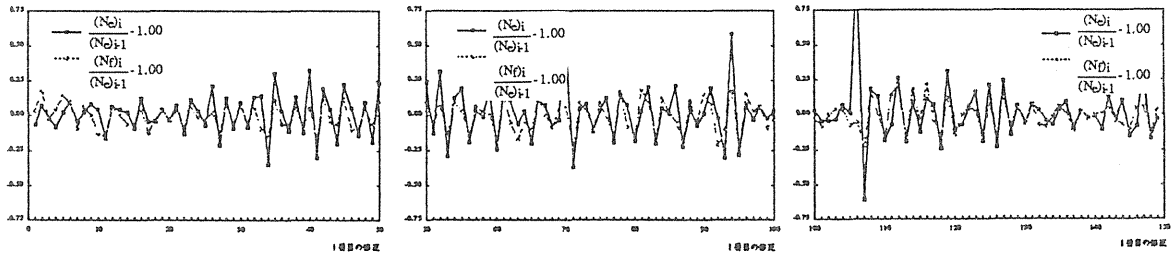


図3.16 i 番目の修正時と i+1 番目の修正時でのΔKが0.5 %ずれるまでに
かかったサイクル数のファジイ予測で求めた比と実際の比の比較

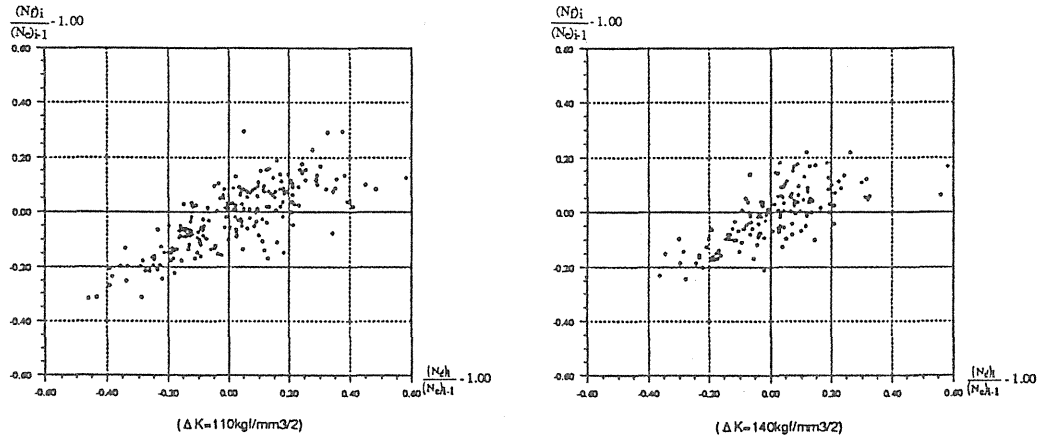


図3.17 (Ne)i / (Ne) i-1 と(Nf) i /(Ne)i-1の関係

3．5　試験の高速化

（1）高速化の問題点

従来より本研究室にて行われてきた疲労試験システムでは高速化を行う際に，

1）ＣＯＤ測定装置の周波数特性が悪い
2）使用コンピュータの性能上，計算が間に合わない
3）油圧サーボ試験機の周波数特性が悪い等の問題があった。

1）は，後で詳しく述べるが、試験の際に亀裂長さの自動計測のために精度よく亀裂開口変位（ＣＯＤ値）を測定することが必要となる。しかしながら従来からこの測定に用いられているクリップゲージは周波数特性が悪く，負荷する荷重の周波数は2Hz程度が限界であった。そのため今回周波数特性のよい無接触光学式変位計を導入し、それの検査を行った結果、クリップゲージに比べ2Hz以上の周波数においても非常に精度良くＣＯＤ値を測定することができることがわかった。したがって実際の試験ではこれを用いてＣＯＤ値の測定を行った。

2）はシステムの自動化や亀裂長さの算出などによってコンピュータに負担がかかり、プログラム中でタイマー割り込みを行っているので試験の高速化により計算が終わらずに割り込みがかかる恐れがあった。この問題についてはより高性能なコンピュータの利用によっても解決できるが，コストの面から従来から使用しているコンピュータを用い，既存のプログラムに改良を加え、より高速化することで解決した。

3）については２つの問題を含んでいた。１つは以前から指摘されていたが，入力信号に対してタイムラグがあり，周波数が高くなるとこれが大きくなり正確な荷重が出力されなくなる。また周波数に対してどの程度出力が出力が変わるのかわからなかった。もう１つは上昇荷重よりも下降荷重の追従性が悪いという問題であった。これらは実際の実験を行う前に，予備実験として試験機のタイムラグならびに，各周波数に対する荷重の減少率を調べた。

以上により３つの問題は解決でき，従来2Hzで行ってきた試験を５Hzまで周波数を上げることができ試験時間の短縮化に大きく貢献した。

（2）　油圧サーボ試験機の荷重応答の遅れ

疲労試験に先駆けて，MTS社製電気油圧式サーボ試験機とZimmer社製の光学変位計の性能検査を行った。本研究のような，鋼を用いた疲労試験の結果は，数 Hz 程度の負荷周波数の違いによる影響はほとんどないと思われるので，試験機の性能が可能な限りなるべく高い周波数の荷重振幅で試験を行う。そのために，まず試験装置の性能を以下の要領にて検査した。またこれに用いた機器は表３．６に示す。

表３．６ 使用機器一覧

装置名	モデル番号	備考
ファンクション ジェネレータ	MTS 410 (MTS)	サイン波出力
ＡＤ変換器 （１２ビット）	AD12-16A (CONTEC)	２系統のアナログ入力に対し同時にサンプリングしデジタル信号に変換
データ記録	PC-9801 (NEC)	
ＡＤ変換器 制御プログラム	ADLABO	ＡＤ変換器の内部タイマを利用してサンプリングタイミング等を制御する
クリップゲージ	632.02C-20 （ＭＴＳ）	
光学変位計	Model 200 X (Zimmer)	

試験機の特性上，ある指令信号を試験機に入力しても，実際に出力される荷重は入力された信号に対して時間的にやや遅れが生じる。そこで，周波数が1.0，2.0，5.0　Hzの一定振幅のサイン波を入力し，その入力信号と実際に出力された荷重の応答の関係を調べた。

a) 検査方法

試験装置は図 3.18 に示すように，まず入力信号用ファンクションジェネレータから0-10(V)のサイン波を出力し，ゲインアンプにて荷重がDP＝17600Nになるように調整し，それに対する試験機の実出力の反応を見た。試験機に送られた荷重信号と試験機のロードセルから得られた実際に出力された荷重の信号をＡＤ変換器に送り，２つの信号を同じ時刻にサンプリングし，そこでデジタル信号に変換する。

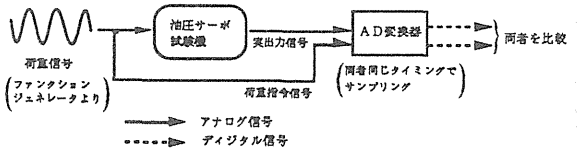


図３．１８　試験における各信号の流れ

そしてマイクロコンピュータでその時の荷重信号と実際に出力された信号をそれぞれ荷重値に換算しそれらをフロッピーディスクに記録した。 ここでは実際に疲労試験を行う場合と同じように試験機に試験片を取付け，指令信号，応答信号ともに疲労試験で用いるノイズフィルタを通した信号を検査した。使用した試験片は，50kgf級の高張力鋼（BS4360）を使った標準ＣＴ試験片で，亀裂長さは約50mmである。これに，荷重振幅DP＝17,600Nを負荷した。これは数百サイクルの繰り返しでは亀裂成長にほとんど影響がでない規模の荷重である。

b) 検査結果

表３．７の条件で試験を行ったところ。図 3.19 のような結果が得られた。ただし、同図は，荷重応答の時間的遅れを明確にするために，グラフに示す際に荷重応答の振幅を指令信号の振幅と同等になるようにある係数を乗じた値である。指令信号の周波数を1.0Hz, 2.0Hz, 5.0Hzの３種類について行ったが，実験を行った範囲では周波数による出力信号の時間的遅れはほぼ一定していた。具体的には，信号の最大値，最小値付近では約0.04秒程度，荷重の平均値付近では荷重が上昇している場合には約0.06秒，荷重が下降している場合には約0.03秒程度遅れることがわかった。

なお、応答信号の位相を調節し，指令信号と位相を合わせたグラフを図 3.20 に示す。グラフより判断すれば，指令信号と応答信号の波形との間に若干の相違が見られるが，疲労破壊を考える場合，主に荷重の最大値と最小値の差が大きな影響をおよぼし，波形の違いによる亀裂伝播の影響は小さいことを考慮すれば，この程度の波形の違いは亀裂伝播にほとんど影響しないと思われる。 つまり指令信号がサイン波の場合，応答に遅れがでるものの，波形の形状そのものにはほとんど影響がでないので，一定のサイン波を出力している限り，応答荷重の最大値，最小値を調べれば，目標の荷重（サイン波）が出力されているか否かの判断が可能であると思われる。

表３．７ 試験条件の一覧

入力信号	1.0 2.0 5.0 Hz のサイン波
荷重	ΔＰ＝17600 N
サンプリング間隔	10 msec
トータルサンプリング時間	2 sec

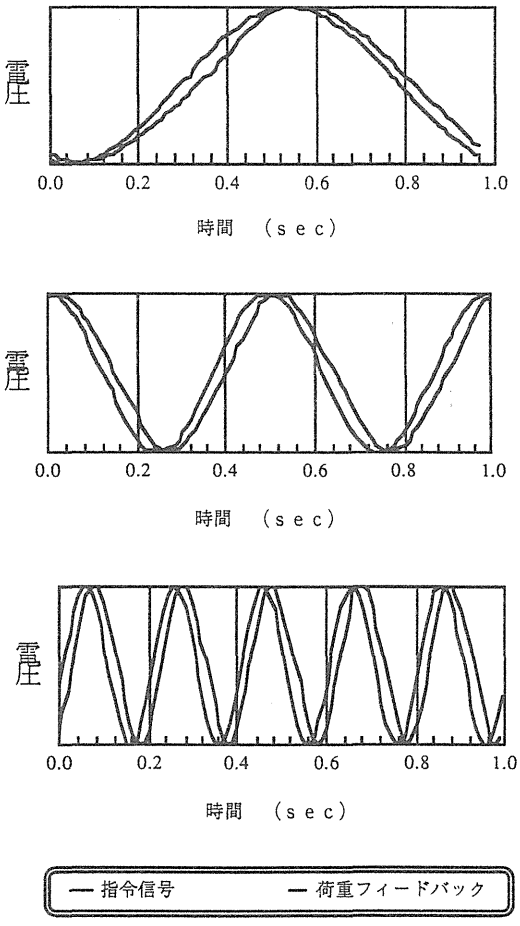


図３．１９ 荷重の遅れ(上から 1Hz,2Hz,5Hz)

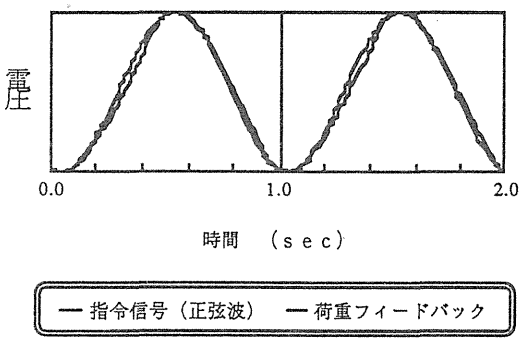


図３．２０ 荷重信号の形状（１Ｈｚ）

（３） 周波数，荷重，変位による応答特性

前述の検査の結果よりわかるように試験機の入力信号に対する出力荷重の応答が遅いため，周波数が高くなるとロードセルが反応する前に次の信号が試験機に入力され，目標の荷重よりも小さい荷重が出力される傾向がある。またその他の荷重減少の要因として周波数，荷重規模，ロードセルの変位の違いなどが考えられる。もしも諸条件によって生じる荷重の減少の度合い（荷重減少率）がわかれば，負荷する荷重の波形がサイン波の場合には，あらかじめ周波数，ロードセルの変位の違いなどによる荷重振幅の減少を考慮し，目標の荷重よりも大きめの指令信号を試験機に送るなどの手段を講ずれば目標の荷重を得ることが可能であろう。（図 3.21）

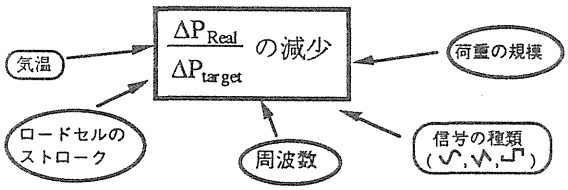


図３．２１ 減少率の原因

a) 検査方法

使用機器は表３．６に示したとおりである。

まず試験機にある一定の振幅で５種類の周波数を含んだサイン波群の目標荷重信号を試験機に入力し，その時出力される荷重信号を調べ，十分に低い周波数（1Hz）で信号を送った場合との違いを見た。実際に用いたサイン波は図 3.22 のようなもので１回の目標荷重群の中に異なった５種類の周波数のサイン波を含んでいる。このようなサイン波群を，低い周波帯（荷重群１）と高い周波帯（荷重群２）の２種類を用意し，荷重振幅を変えて試験機に入力する。またロードセルのストロークの違いによる特性を見るために，試験片の亀裂長さが25mm，45mmの２種類の場合について調べた。表3.8に実験に用いた波形の詳細を示す。

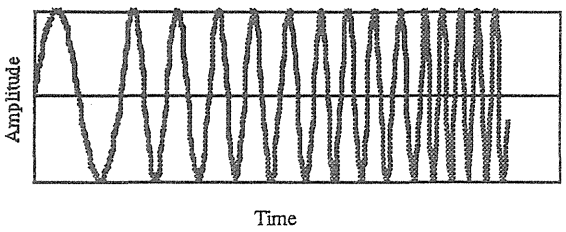


図３．２２ 検査のためのサイン波形群

表３．８ 荷重応答の特性検査条件

	周波数(f)	荷重(ΔP)	き裂長さ(a)
荷重群１	1.0 2.0 2.5 3.5 5.0 Hz	600 1200 1800 2400 3000 kgf	25 45 mm
荷重群２	5.0 6.6 8.0 10.0 13.3 16.0 Hz	600 1200 1800 2400 kgf	25 45 mm

以上により試験機の周波数特性，荷重特性，ストロークによる特性が知ることができる。各条件による試験機の特性は，1Hzの時の荷重出力に対して各条件の時出力された荷重値の割合を荷重減少率としDP/DP_{1Hz}で評価する。

b) 検査結果

まず問題を単純化するために荷重，亀裂長さを一定とし，周波数のみ変化させた時の荷重減少率を図 3.23 に示す。このグラフは荷重振幅DP＝5,880（N），亀裂長さが 25mm の時の周波数に対する荷重の減少率を示している。グラフよりわかるように周波数が増加するにつれて荷重減少率も大きくなっている。具体的には，5Hzの時には目標荷重の約80%しか出力されないことがわかる。また，5Hzまでについての荷重減少率について示しているが，5Hz以上も減少傾向が続くと思われる。

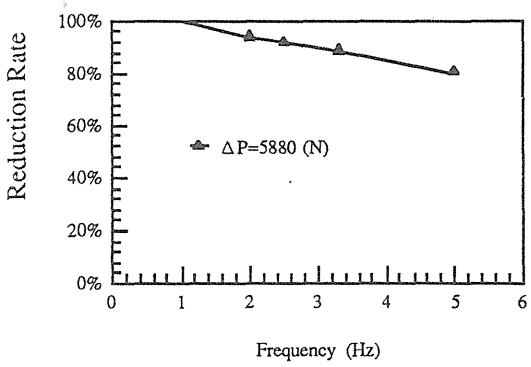


図３．２３ 周波数に対する荷重減少率

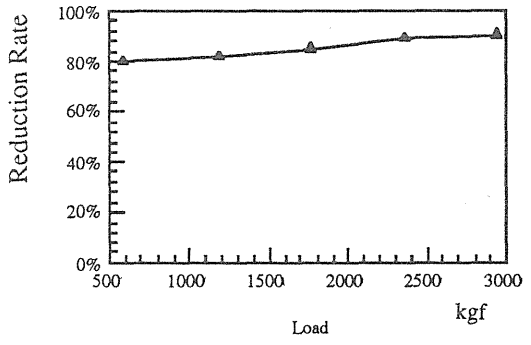


図３．２４ 荷重に対する荷重減少率

図 3.24 は，周波数，き裂長さを一定として荷重振幅のみを変化させた時の荷重減少率のグラフである。き裂長さは図 3.23 と同様に25mmで，周波数は f = 5.0Hz のものであるが，これによれば荷重が大きくなればそれとは反対に荷重減少率は小さくなる傾向がある。つまり目標荷重が大きくなれば出力される荷重もその目標荷重に近くなることを示している。

次に周波数，荷重，き裂長さの３点を変化させた場合についての結果を図 3.25 に示す。結果は，周波数については高くなるにつれ減少率が大きくなり16Hz付近ではき裂長さが25mmの時，45mmの時ともに出力された荷重は目標の30%以下であった。き裂長さ（ロードセルの変位の大きさ）の影響は，全体的な傾向（荷重規模による影響，周波数の影響）は変わらないが，き裂が伸びると減少率も大きくなることがわかった。また，周波数の影響に比べそれほど大きくないが，荷重が小さくなるにつれ，目標の荷重に対して出力された荷重の割合（荷重減少率）は，小さくなることがわかった。

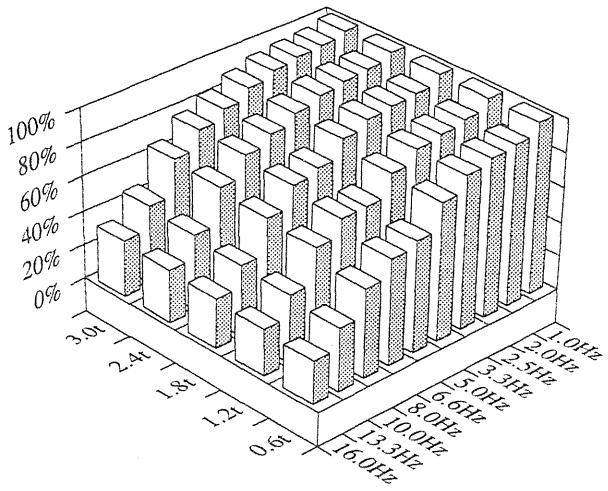


図３．２５-a 荷重，周波数に対する減少(25mm)

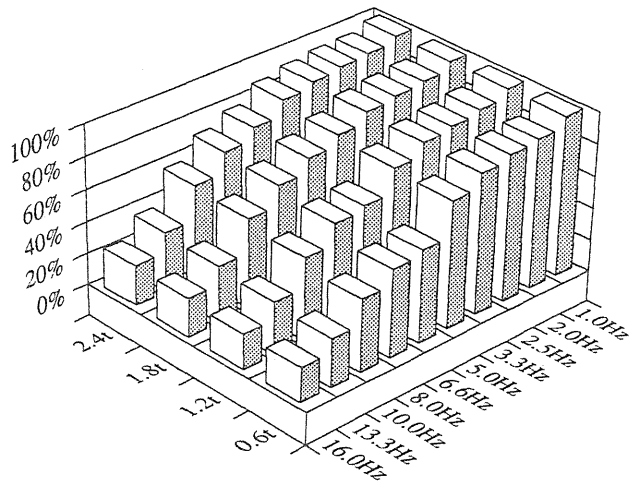


図３．２５-b 荷重，周波数に対する減少(45mm)

（4）　ＣＯＤ測定装置の性能比較

亀裂長さを自動計測するために，ＣＯＤ値（試験片の切り欠きのある側面の亀裂開口変位）の測定を行うが，その測定装置の性能を検査する。亀裂の亀裂閉口などが十分小さい場合（試験片のコンプライアンスが一定の場合）には，フックの法則より応答荷重とＣＯＤ値は比例関係になっているはずである。しかしながら，変位の測定装置の影響，油圧サーボ試験機の特性，亀裂閉口，ロードセルと試験片の固定方法など様々な要因のため，必ずしも比例関係にならない。特に周波数が高くなるとそれが顕著になる傾向があることがわかっている。主な原因として変位の測定装置であるクリップゲージを用いた時，これよりより得られたＣＯＤ値は荷重に対して時間的に遅れる傾向があるためであろうと予測した。そのため周波数特性に優れた光学変位計を用いてＣＯＤ値を測定した場合にどの程度結果が改善されるかを調べた。

○クリップゲージ（632.02C-20）

図 3.26 のように試験片の切り欠き側の側面に取付用の金具をつけ，そこにクリップゲージを取り付ける。ＣＯＤ値は，ストレインゲージと同様に，歪によって生じた電位差をひずみアンプによってその時のＣＯＤ値に応じた電圧に変換して出力される。

○無接触光学変位計（Model 200X）

光学的に明るい部分と暗い部分の境界をターゲットに，その変位を光の干渉を利用して測定する。測定対象物に触れないので対象物の動きに影響を与えずにターゲットの変位を計測でき，特に周波数特性に優れている。（図3.27）

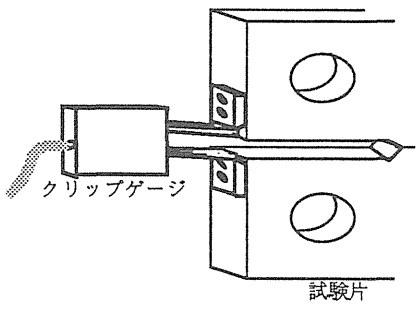


図３．２６ クリップゲージの試験片への取付図

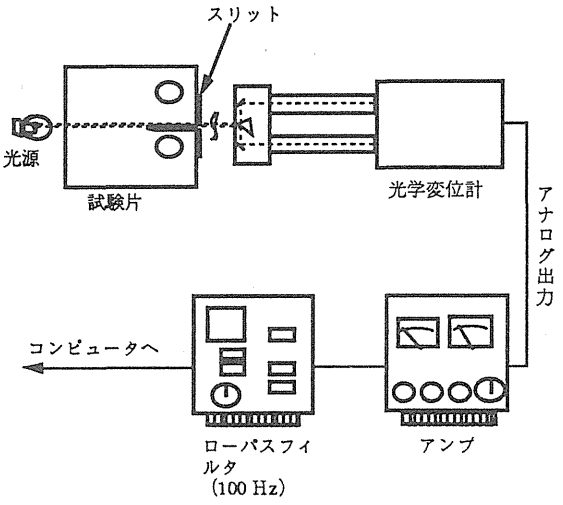


図３．２７ 光学式変位計による計測の信号の流れ

a) 検査方法

従来から用いられていたクリップゲージと今回導入した無接触光学変位計の２種類にてＣＯＤ値測定しそれを比較した。

表3.6の装置を使用し，実際の疲労試験と同じ要領にて試験片にサイン波荷重を負荷しその時の荷重応答とＣＯＤ値の関係を上記の２種類のＣＯＤ値測定装置について調べた。周波数は，両測定装置において十分に周波数特性がよいと思われる1Hzの場合，実際の実験に用いる予定の周波数と同じ値の5Hz，そして比較的周波数が大きい10Hzの場合の３つの場合について行った。

b) 検査結果

横軸に時間，縦軸にその時の荷重応答とＣＯＤ値を同時にサンプリングした時のグラフを図3.25に示す。これよりわかるように，周波数が1Hzの場合には，応答荷重に対するＣＯＤ値の遅れは，クリップゲージ，光学変位計共に見られない。しかし，周波数を10Hzにした場合には，光学変位計によるＣＯＤ値は1Hzの場合と変わらないが，クリップゲージによって得られたＣＯＤ値は，荷重に対して遅れて出力されていることがわかる。つまり従来より使用されてきたクリップゲージは周波数特性が悪く，周波数が高くなると精度がよくない事がわかった。

次に縦軸にＣＯＤ値，横軸に荷重値をとり，連続した２周期の荷重について１周期256ポイントずつサンプリングしたものを示す。（図 3.29 ）　クリップゲージ

を用いた場合には，5Hzの場合には荷重に対するＣＯＤ値の線形性は失われ，正確にコンプライアンスを測定することは難しい。それに対し，光学変位計の場合には，5Hzまでははぼ比例関係にあり，精度よくコンプライアンスを測定することが可能である。

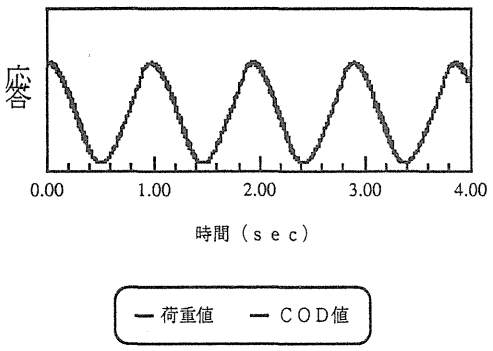


図３．２８-a 荷重値とＣＯＤ値の応答
（クリップゲージ使用）

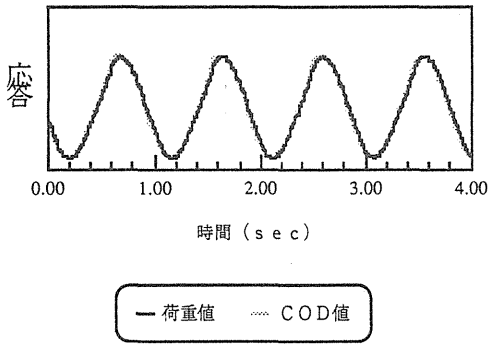


図３．２８-b 荷重値とＣＯＤ値の応答
（光学変位計使用）

またそれ以上の周波数ではクリップゲージの場合には全体的にＣＯＤ値が遅れるのに対し，光学変位計の場合は荷重が大きい方が荷重の遅れが顕著になる。

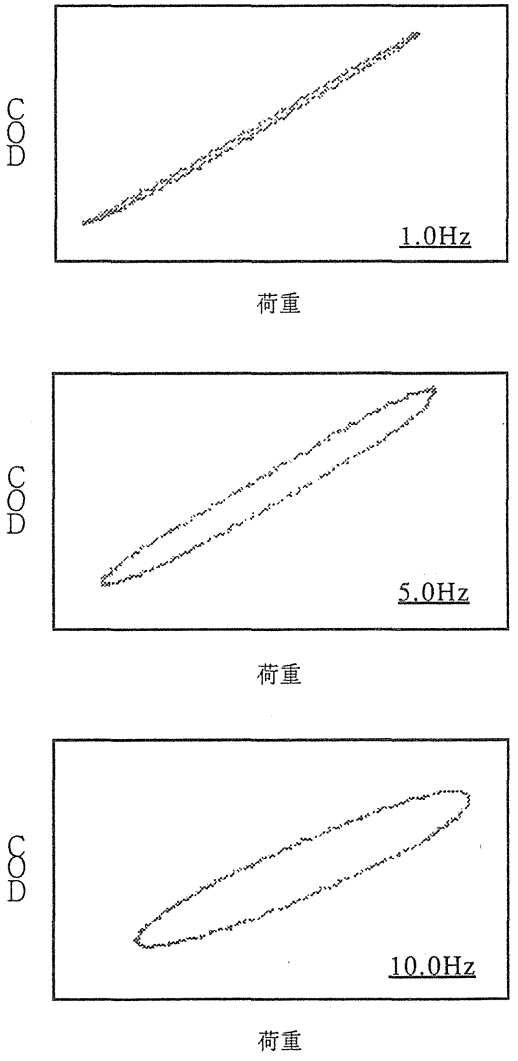


図３．２９-a 荷重に対するＣＯＤ値（クリップゲージ使用）

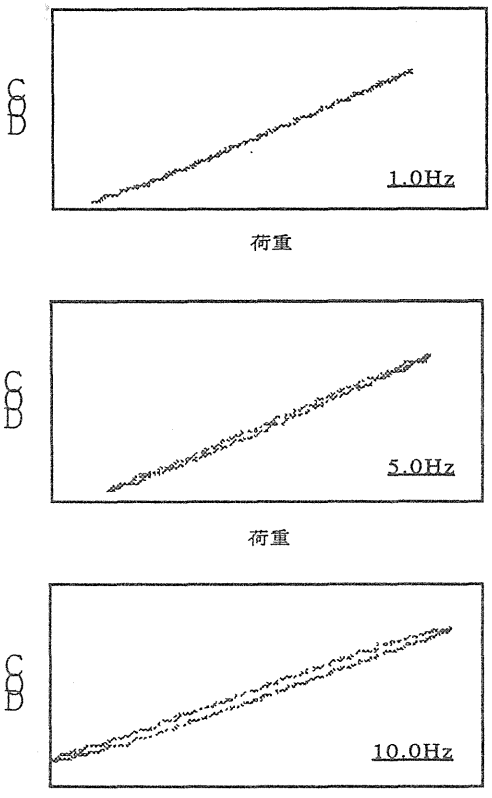


図３．２９-b 荷重に対するＣＯＤ値
（光学式変位計使用）

４．試験結果とその考察

試験条件と試験結果をBS4360とL36E材についてそれぞれ表4.1,4.2に示す。得られたa～N線図は図4.1-a,-bに示す。図から分かるように、a～N曲線はほぼ直線である。同一のΔK値に対してもその傾きが違い、試験片毎の平均伝播率あるいは平均伝播抵抗に変動があることが明らかである。

また、図4.2に応力拡大係数範囲ΔK、疲労き裂伝播率da/dNの変動を亀裂長さaに対してプロットしてある。図から分かるように、ΔK値はよく制御されているにも関わらず、亀裂伝播率が大きく変動していることが分かる。これは、微小間隔で激しく変動するCに起因する変動と穏やかに変化するmによる変動の重なった変動であると考えられる。なお、他の全ての試験片およびL36E材でも同様の関係が得られている。

表４．１　疲労試験条件と試験結果（BS4360）

specimen no.	ΔK MPa$\sqrt{m}$	mean of $\frac{da}{dN}$ (m/cycle)	c.o.v of $\frac{da}{dN}$ (%)	m	c.o.v of Z(x) (%)*
1BS08	21.8	6.26×10^{-8}	13.7	3.60	13.9
1BS09	21.8	5.61	15.2	3.88	14.2
1BS10	21.8	5.52	14.6	3.92	14.0
1BS11	21.8	6.58	14.0	3.48	14.7
4BS01	21.8	7.65	15.7	3.10	15.4
4BS02	21.8	7.98	2.78	2.99	15.5
2BS01	28.0	1.75×10^{-7}	11.6	2.79	11.4
2BS02	28.0	1.83	11.4	2.49	11.8
2BS03	28.0	1.80	10.1	2.60	10.9
2BS04	28.0	1.85	9.5	2.42	9.4
2BS05	28.0	1.90	8.9	2.24	9.1
2BS06	28.0	1.85	10.4	2.42	11.1
2BS07	28.0	1.84	10.0	2.45	10.3
2BS09	28.0	1.90	12.7	2.24	12.6
2BS10	28.0	1.78	12.5	2.68	13.2
2BS11	28.0	1.76	14.5	2.75	14.4
3BS02	34.2	3.06×10^{-7}	8.5	2.75	8.4
3BS04	34.2	3.00	10.9	2.41	10.5
3BS05	34.2	3.17	9.5	3.49	8.8
3BS06	34.2	3.10	7.9	3.05	6.9
3BS07	34.2	3.08	10.9	2.93	10.6
3BS08	34.2	2.96	7.1	2.15	6.9
3BS10	34.2	2.99	6.6	2.35	6.2
5R01	43.5	5.76×10^{-7}	9.5	2.65	9.4
5R02	43.5	5.12	9.1	2.25	8.4
5R03	43.5	5.74	9.5	2.64	9.1
1BS02	43.5	6.42	7.5	3.03	7.3
1BS04	43.5	6.36	9.2	2.99	8.6
1BS05	43.5	6.02	15.1	2.81	11.1

* mean of $Z(x) = 3.81 \times 10^{6}$

表４．２疲労試験結果と試験結果（L36E）

Specimen ID	ΔK (MPam1/2)	mean of (da/dN) (m/cycle)$\times 10^{-7}$	COV of (da/dN) (%)	m
2HT1	28.0	0.61	23.0	3.54
2HT2	28.0	0.25	15.1	5.18
2HT6	28.0	1.14	7.5	2.32
3HT1	28.0	0.78	23.9	3.08
3HT7	28.0	1.19	12.1	2.40
5HT4	28.0	0.70	36.2	3.79
1HT4	34.2	2.18	22.3	1.85
1HT5	34.2	1.68	30.4	1.07
1HT6	34.2	2.78	8.5	2.69
1HT7	34.2	1.79	12.4	2.39
4HT3	34.2	1.63	17.8	2.68
4HT5	34.2	2.11	15.6	1.92
2HT3	43.5	3.74	13.9	4.70
2HT4	43.5	3.21	11.9	0.73
2HT5	43.5	2.60	22.9	2.29
3HT6	43.5	3.71	9.4	0.75
4HT1	43.5	2.42	16.7	5.32
5HT1	43.5	2.97	9.9	3.08
5HT2	43.5	3.85	10.9	0.38
5HT3	43.5	2.95	11.7	3.17
3HT2	55.9	4.86	20.6	1.16
3HT3	55.9	7.18	7.1	3.78
3HT4	55.9	5.41	16.4	1.90
3HT5	55.9	5.01	12.7	1.44
4HT2	55.9	6.64	11.4	3.26
4HT4	55.9	6.60	13.0	3.20

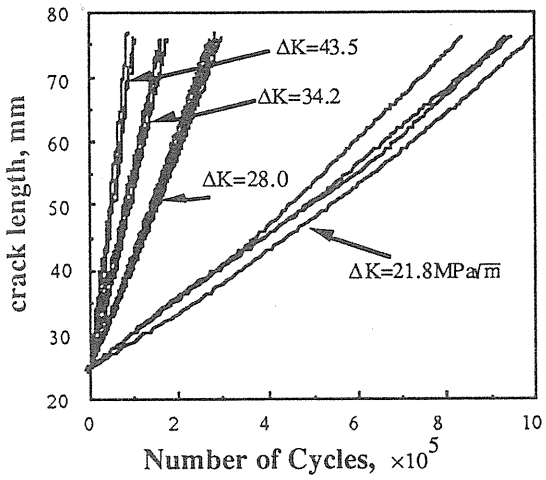


図４．１-a ΔＫ値一定試験における a～Ｎ曲線
(BS4360)

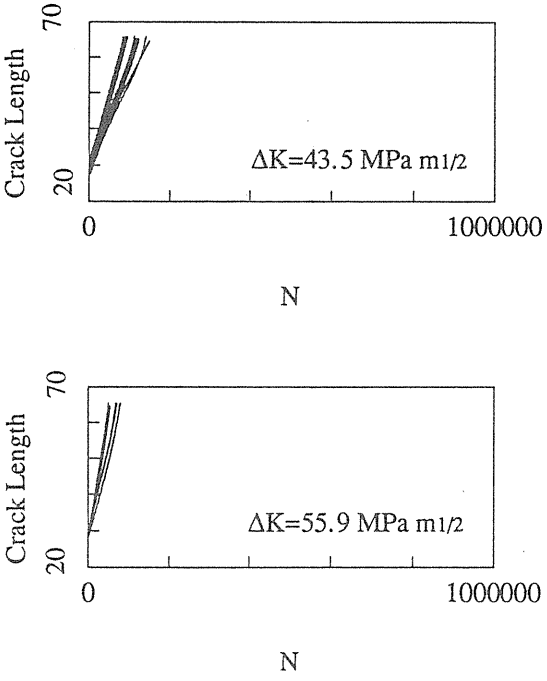


図４．１-b(2)ΔＫ値一定試験におけるa～Ｎ曲線
(L36E)

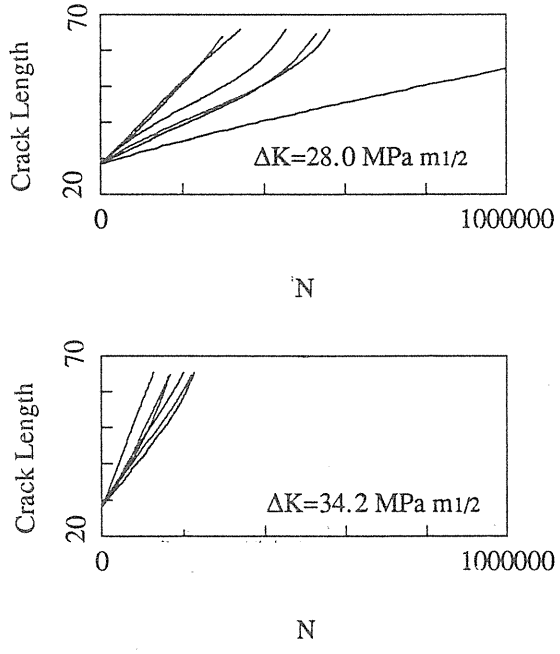


図４．１-b(1)ΔＫ値一定試験におけるa～Ｎ曲線
(L36E)

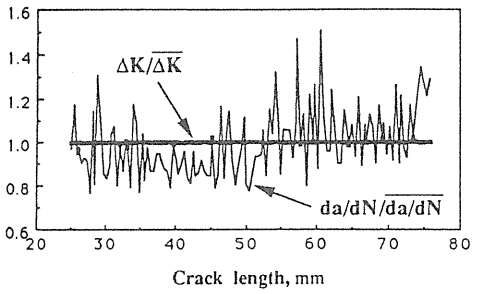


図４．２ 亀裂伝播経路に沿った亀裂伝播率の
変動例(BS4360)

4．1パラメータK_0とC_0の検討

図4.3-a,-b は亀裂伝播率da/dNの標準偏差σ_{vi}をlogΔKに対してプロットしたものである。図から式(9)が成立しているように見受けられる。図中には$\Delta K < K_0$と見られるΔKについては、σ_{vi}を負にした数値もプロットしてある。これらの点を用いて最小二乗法で直線のあてはめを行なったのが図中の直線である。
この直線と横軸との交点K_0の一つの推定値を与えよう

ここで、K_0の値はBS4360とL36Eについてそれぞれ32.5, 46.0 $MPam^{1/2}$として得られた。

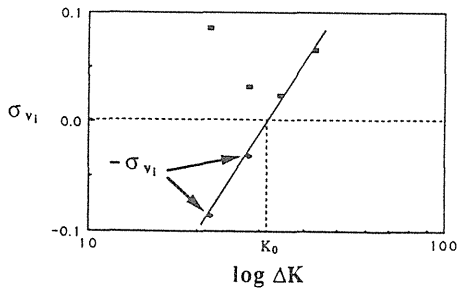


図４．３-a 平均亀裂伝播率の標準偏差とlogΔKの関係（Bs4360）

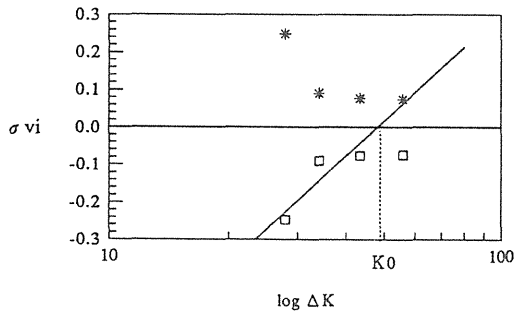


図４．３-b 平均亀裂伝播率の標準偏差とlogΔKの関係（L36E）

K_0が推定できれば、C_0は$\Delta K = K_0$の点の平均伝播率$\overline{da/dN}$として与えられることになるが、試験片毎にm_iが異なると考えられるので、この点で数回実験を繰り返さなくてはならず実際的とは考えられない。そこで、近似的にm_iが平均 $\bar{m}$,分散σ^2_mの正規分布に従うと仮定して次のようにしてC_0を推定する。

式（７）より試験片毎の平均伝播率の対数$\log\left[\left(\frac{da}{dN}\right)_i\right] = v_i$は平均が$\log C_0 + \bar{m}(k_i - k_0)$が$\sigma_m^2(k_i - k_0)^2$の正規分布に従う。

今、n回の試験を行なって、
$v_1, v_2, v_3, \cdots\cdots, v_{n-1}, v_n$を得たとすると、likelihoodは

$$L = \left\{\frac{1}{\sqrt{2\pi}\sigma^m}\right\}^n \left\{\prod_{i=1}^{n}\frac{1}{|k_i - k_0|}\right\} \times\exp\left\{-\sum_{i=1}^{n}\frac{\left[v_i - \log C_0 - (k_i - k_0)\bar{m}\right]^2}{2(k_i - k_0)^2\sigma_m^2}\right\} \quad (23)$$

$\log C_0$と$\bar{m}$の最尤推定値は

$$\log C_0 = \frac{\sum_{i=1}^{n} v_i\left[\frac{n}{(k_i - k_0)^2} - \frac{S_1}{k_i - k_0}\right]}{nS_2 - S_1^2}$$

$$\bar{m} = \frac{\sum_{i=1}^{n} v_i\left[\frac{S_2}{k_i - k_0} - \frac{S_1}{(k_i - k_0)^2}\right]}{nS_2 - S_1^2}$$

ここに、

$$S_1 = \sum_{i=1}^{n}\frac{1}{k_i - k_0} \quad 、 \quad S_2 = \sum_{i=1}^{n}\frac{1}{(k_i - k_0)^2}$$

C_0が決まれば式 (7)から個々の試験片のm_iを求めることができる。

4．2 パラメータmの確率分布

前節のように求めた m_i の確率分布を図4.4-a,-b に示す。m_iについてはすでに平均と、標準偏差が求まっているが、個々の実験値から改めて求めた値と比べると、平均値はよくあっているが標準偏差は必ずしも一致しないようである。

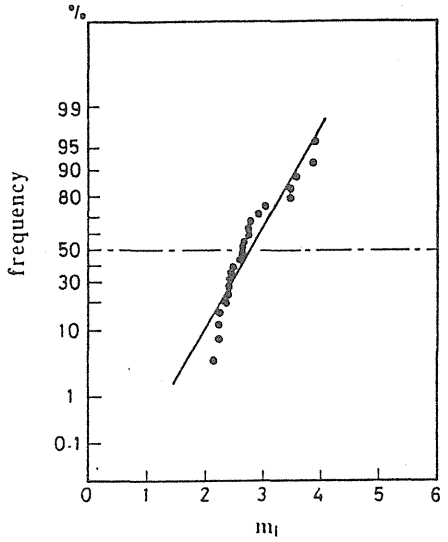


図４．４-a 試験片毎のm_iの正規確率紙プロット
(BS4360)

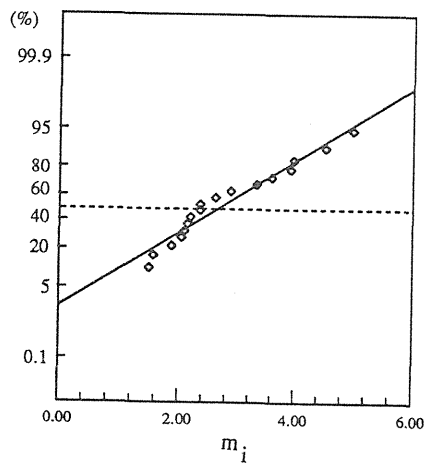


図４．４-b 試験片毎のm_iの正規確率紙プロット
(L36E)

４．３　伝播抵抗係数の確率分布

K_0と各々の試験片のm_i　とが求められれば、式(3)から各々の試験片でのC(x)を各点での伝播率から求めることができる。材料特性を表す数値としては　C(x)よりもその逆数 Z(x) = 1/C(x) の方が適当であるので、以下では伝播抵抗係数として Z(x) を用いる。このようにして求めた亀裂伝播経路に沿うZ(x)　の一例を図4.5　に示す。図より、Z(x)が試験片内で激しく変動することが分かる。

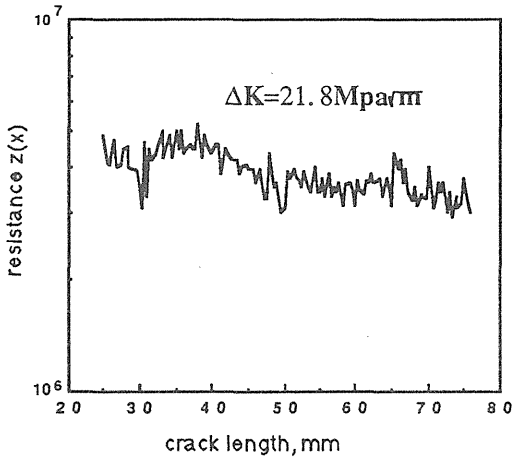


図４．５ 亀裂伝播経路に沿ってのZ(x)の変動
(BS4360)

ΔK値毎のZ(x)の累積頻度を図 4.6 に示す。　Z(x)の確率分布はΔK値に依らず、よく一致しているので、先のようなパリス則を仮定する場合は、このようなデータ整理法が妥当であるといえよう。全てのZ(x) のデータをワイブル確率紙上にプロットしたものを図4.7に示す。直接探索法を用い確率分布関数F_z(z | α、β、γ）の母数α、βおよびγを推定し、BS4360の場合4.2, 3.96×10^6および2.34×10^6を得た。関数F_z(z | α、β、γ）を計算し同図に実線で示す。図から判るように、伝播抵抗係数Z(x)は3母数のワイブル分布に従うといえよう。

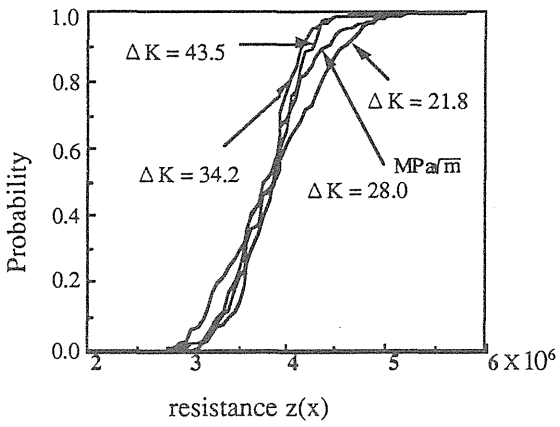


図４．６ΔK毎のZ(x)の確率分布(BS4360)

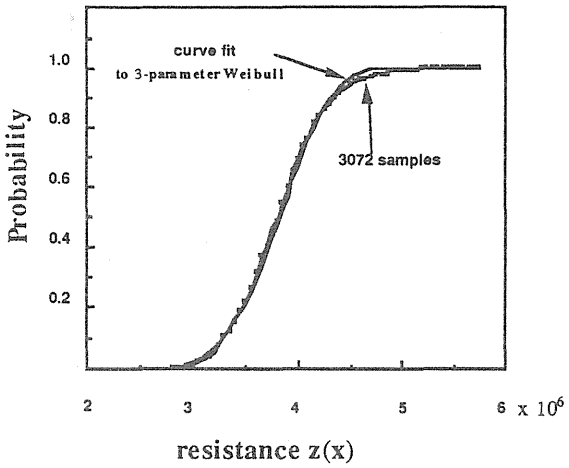


図４．７全データのZ(x)の確率分布（BS4360）

Z(x)のデータから求めた平均自己相関関数を図4.8-a,-b に示す。自己相関関数を図中に示した式で近似し適当に定めた係数を用いた計算値を同図に示してある。

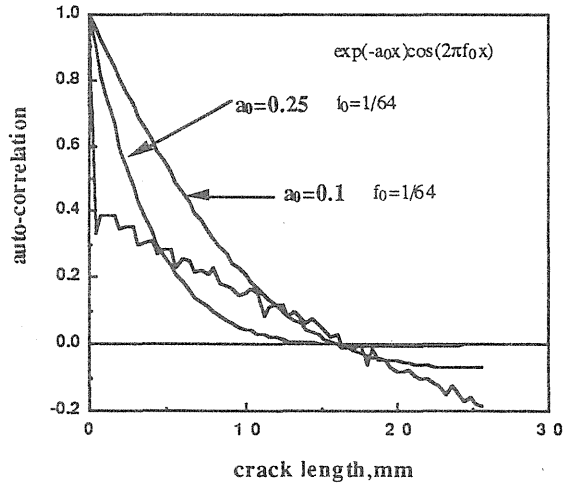


図４．８-a Z(x)の平均自己相関関数（BS4360）

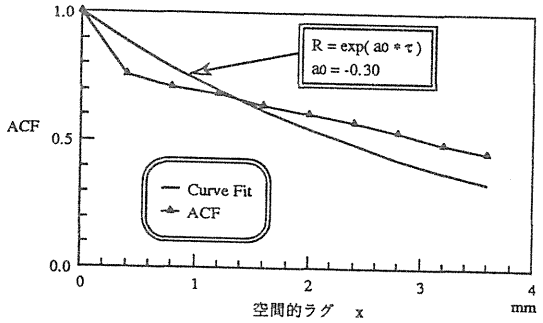


図４．８-b Z(x)の平均自己相関関数（L36E）

５．信頼性解析への応用

以上で得た結果を信頼性解析に適用する一例として疲労亀裂伝播寿命の確率分布の推定法について述べる。

５．１ 亀裂伝播のシミュレーション

先に決めたm_iの平均値 $\bar{m}$と分散 σ_m^2を用い、ガウス分布に従う伝播指数m_iをシミュレートし、それを一つの試験片 i の m_i とする。また、さきに求めた自己相関関数と確率分布関数を持つ非ガウス確率過程Z(x)を、Shinozukaらの開発したシミュレーション方法[6]を用いて、シミュレートする。図5.1に概略のフローチャートを示す。

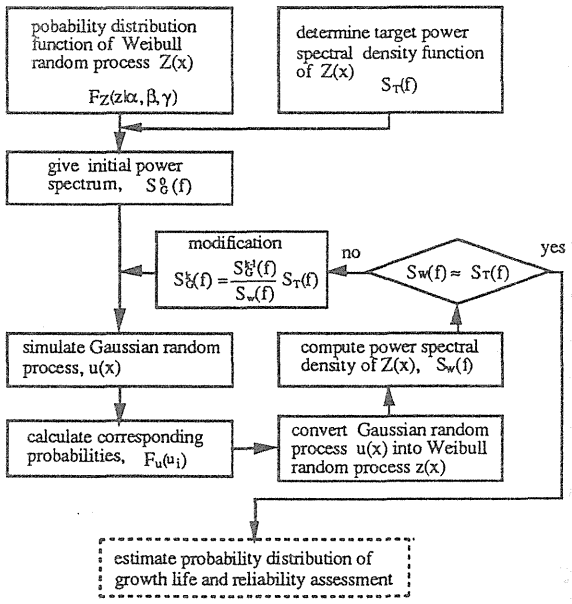


図５．１ Z(x)のシミュレーション方法のフローチャート

シミュレートしたｍとZ(x)のデータより、初期亀裂長さa_0を与え、式（２）を用いaとNを求めると、a～N曲線が得られる。図5.2-a,-b にa_0=25mmとして、ΔK値毎に１００回（L36Eでの場合は２０回）のシミュレーションを行なって得られた a～N曲線の範囲を示す。実

際に実験を行なって得られたデータである図４．１-a, -b と比較すると、二鋼種ともシミュレーションと実験値はほぼ一致することが分る。

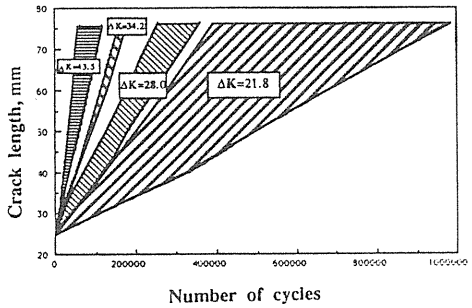


図５．２-a シミュレーションによる a～N曲線
（BS4360, ΔK値一定）

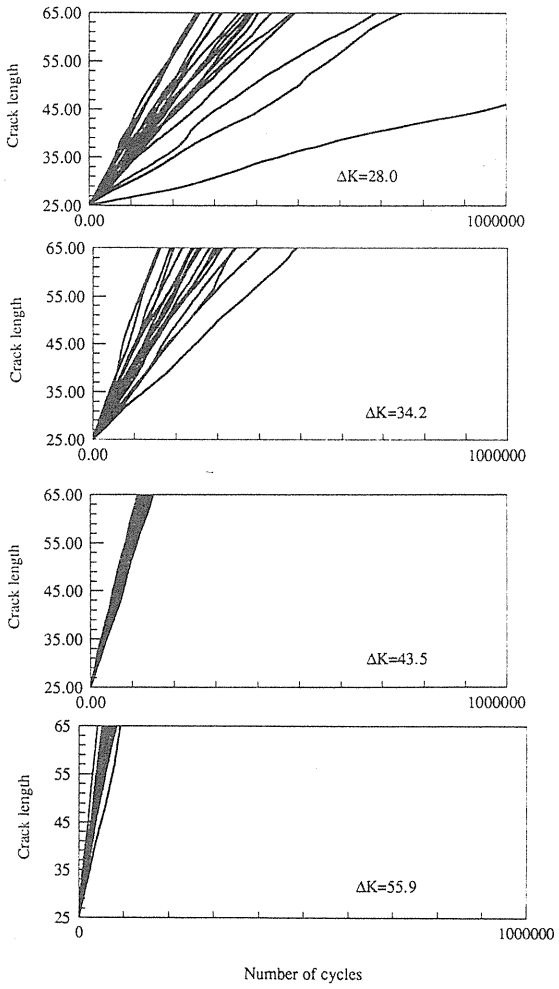


図５．２-b シミュレーションによる a～N曲線
（L36E, ΔK値一定）

図5.3-a,-b は荷重一定試験における場合でΔK値一定試験の場合と類似した結果が得られた。

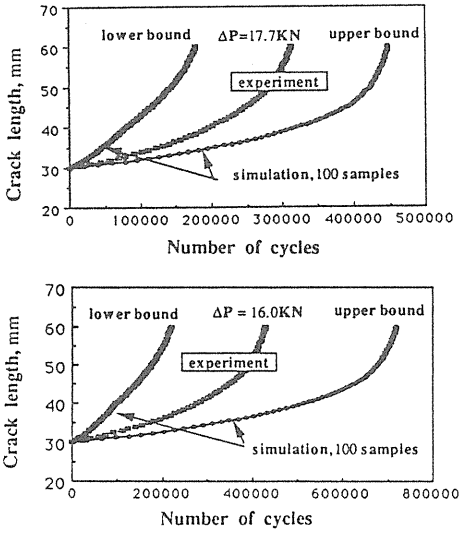


図５．３-a シミュレーションによる a～N曲線
（BS4360, 荷重一定）

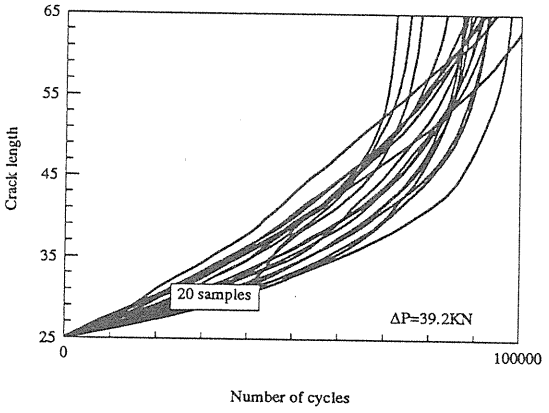


図５．３-b シミュレーションによる a～N曲線
（L36E, 荷重一定）

5．2　伝播寿命分布と信頼性

以後、BS4360材の場合の解析例について述べる。伝播指数m_iと伝播抵抗係数Z(x)のシミュレーションを繰り返し、荷重制御下のa～N関係を多数求め、亀裂長さがそれぞれ40mm,50mmおよび60mmに達した時の伝播寿命N(a)を求め、ワイブル確立紙上にプロットしてみると図 5.4 のようになる。図中の実線はN(a) が3母数のワイブル分布に従うとして定めたものである。疲労亀裂伝播寿命N(a)は3 母数のワイブル分布に従うと考えられる。

構造物の信頼度R(n)を、ある亀裂長さに達するまでの寿命Nが与えられた値 n より大なる確率と定義すれば、推定された伝播寿命の分布関数を用い、所定の信頼度に対応する伝播寿命が求められる。例として ΔP = 1,800kgf (17.7kN)の荷重下の幾つかの計算値を表5.1 に示す。

所定の亀裂長さ a_1 に達するに必要な最小繰り返し数 Ns （以下、最小寿命と呼ぶことにする）は、推定された疲労亀裂伝播寿命の分布関数中の位置母数γ値であるので、初期亀裂長さ a_0 = 30mm,所定亀裂長さがそれぞれ 40mm,50mmおよび60mmの時の表5.1 に示す。

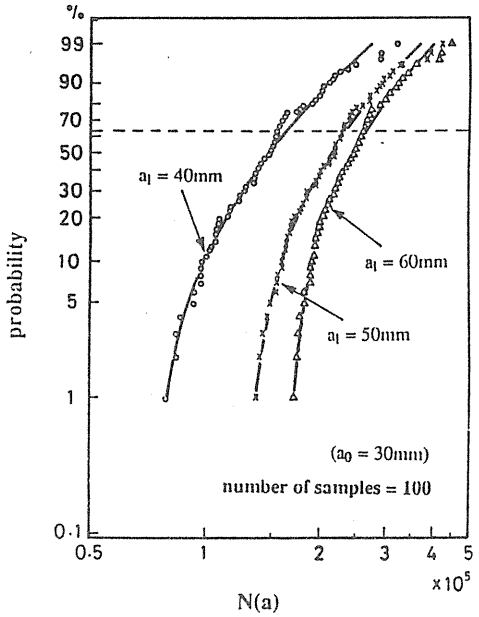


図５．４　シミュレーションによる疲労亀裂伝播寿命のワイブルプロット

一方、亀裂伝播経路に沿ったあらゆる場所において伝播抵抗係数 Z(x) が最小値を取った時の伝播寿命をN_0とし、表5.1 に示す。ただし、m は平均値を用いた。先に求めた最小寿命 N_s は N_0 に対応するものである。図5.1から判るように、シミュレーションによって推定された最小寿命の数値 N_s は計算値 N_0 とほぼ一致している。

表５．１　シミュレーションによる信頼性計算結果 (BS4360材)

crack length a (mm)	parameters of P.D.F for N(a)			pobability (%)	growth life ($\times 10^4$)	smallest life	
	α	β	γ			N_{S_4} ($\times 10^4$)	N_{0_4} ($\times 10^4$)
		($\times 10^5$)	($\times 10^5$)				
40	1.92	1.62	0.714	99.0 99.5 99.9 99.99 99.999	7.96 7.71 7.39 7.21 7.16	7.14	9.40
50	1.89	2.34	1.27	99.0 99.5 99.9 99.99 99.999	13.6 13.3 13.0 12.8 12.7	12.7	13.9
60	1.79	2.64	1.65	99.0 99.5 99.9 99.99 99.999	17.3 17.0 16.7 16.6 16.5	16.5	15.7

6．結論

疲労を考慮せねばならぬ構造物の安全性、信頼性の推定に不可欠である疲労特性の確率分布につて、単に、平均値、分散等を知るだけでなく、例えば3母数のワイブル分布関数を小数の実験とシミュレーションとによって推定する方法を考察した。即ち、いわゆるパリス則が成立するとして、実験データを整理すれば、パラメータC及び m はともに確率変数とせねばならないこと、もしも、Cは亀裂経路に沿って激しく変動し、m は試験片内ではほぼ一定であるとすれば、二つの確率変数Cと m とを分離して解析しうること、その時、伝播抵抗 1/C の確率分布はΔK値に依らず同一の3母数ワイブル分布に従うことなどを示した。

本研究で用いた材料は５０kg級高張力鋼のBS4360 及び L36E の二種である。

また、本方法は、実験的に応力拡大係数範囲制御の疲労試験に頼っているので、効率よく実験を行えること、制御の精度の向上が必要であること等が重要である。そこで、ファジィ理論を応用した応力拡大係数範

囲制御の試験法を検討し、制御ΔK値のばらつきが0.5％以内の制御であれば十分な制御が復雑な実験手順なしに可能となった。また、亀裂伸長さ計測に関与する亀裂開口変位の挙動について、無接触光学式変位計とC.O.D. ゲージを使った実験との比較を行って試験システムの応答特性等について検討した結果、より高速な実験を可能にした。

以上の結果を用いて、疲労亀裂伝播寿命の確率分布をシミュレーションで求める方法を提案した。提案した方法は疲労亀裂伝播に関するもので、亀裂成長率と応力拡大係数範囲との間に式（1）が成立つ場合であり、亀裂成長率が極端に低くなるいわゆる下限界応力拡大係数以下及び極端に高くなる上限界応力拡大係数以上は対象としていない。殆ど不安定に近くなるところは伝播寿命のばらつきには影響しないであろう。しかし、亀裂発生寿命と下限界とは全寿命に大きく影響する。特に、発生寿命のばらつきには全寿命のばらつきに直接影響するので別途考慮しなければならない。この問題はベイジアンアプローチなどにより解決する方法が試みられているので、これらと組み合わせて解決する必要があろう。　しかし、下限界を無視すれば、安全側の寿命推定が可能である。

最後に、本研究をまとめるにあたり、本学大学院学生の金　善振、藤波　貢、浪岡保宏君等に手伝っていただいた。ここに、謝意を表する

参考文献

(1) 岡村弘之、板垣　浩、強度の統計的取扱い、培風館、(1975)

(2) 北川英夫、他３名、機論、52-480, (昭61), 1749.

(3) 市川昌弘、他３名、材料、33-364, 34-378, 35-398, (昭59,60,61)

(4) Paris,P.C. amd Erdogam,F., ASME J. Basic Eng. 85,(1936),528.

(5) Itagaki,H. and Shinozuka, M., Special Technical Publication 511,ASTM,(1972),168.

(6) Yamazaki,F., and Shinozuka, M., Stochastic Mechanics, 1, (1986), 211.

(7) 板垣　浩、石塚鉄夫、黄　倍彦、日本造船学会論文集、Vol.165,(1989), 253.

(8) Itagaki,H., Ishizuka, T., Huang, P., amd Ueki, S.,INALCO'88 Organizing Committee, Tokyo, (1988), 3-38

(9) Itagaki, H., Ishizuka, T. , and Huang, P., ICAEM, Tianjn, China, (1988),F24.

(10) Ashok Saxena, and S. J.Hudak, Jr., Int.J.Fracture,14-5,(1978),453

付録　ファジィ推論法による疲労試験プログラム

<<<　初期条件設定プログラム　>>>

```
/***tmain.c************************************

        ⊿K一定疲労き裂伝播試験


    横浜国立大学　工学部　生産工学科

            板垣　研究室


            Jan.25,1991


        コンパイル       MS-C Ver5.1
                         /FPi /AS
************************************************/
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <math.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include "punch.h"
#include "p9128.h"
#include "value.h"


void main(void)
{
        int ika,check;
        int  cc;
        FILE *fp;

        simuortest(&iftest);
        k_or_p(&ctrl,&stress,&KVAL);
        check=setcondi(idnum,&thick,&icrack);
        strcpy(fname1,idnum); /*  intial data fname is : "specimen ID.cnd" */
        strcat(fname1,".cnd");  /*  intial data fname is : "specimen ID.cnd" */
        strcpy(fname2,idnum);   /*  recording fname is : "specimen ID.dat" */
        strcat(fname2,".dat");  /*  recording fname is : "specimen ID.dat" */

        while (1)
        {
                ika=specimenc(idnum,&width,&thick,&icrack,&crackstart,&crackend);
                if (ika==999) break;      /* ika==999 : ESC */
                ika=loadc(&freq,&ctrl);
                if (ika==999) break;
                ika=otherc(fname1,fname2,&YOUNG,&loadfact,&codfact,&iftest);
                if (ika==999) break;
                printf("¥n¥n       Everything Fine ?");
                cc=getch();
                if ((cc=='y')||(cc=='Y')||(cc==CR)) break;
        }

        if (NULL==(fp=fopen(fname1,"w")))
                printf("Cannot Open File  %s¥007¥n",fname1);

        fprintf(fp,"Specimen ID Number        : %s¥n",idnum);
```

```
        fprintf(fp,"Condition File Name      : %s¥n",fname1);
        fprintf(fp,"Test Data File Name      : %s¥n",fname2);
        fprintf(fp," ------ Specimen Size ------¥n");
        fprintf(fp,"Width                    : %lf¥n",width);
        fprintf(fp,"Thickness                : %lf¥n",thick);
        fprintf(fp,"Initial Crack            : %lf¥n",icrack);
        fprintf(fp," ------ Test Condition -----¥n");
        fprintf(fp,"First Recording Crack    : %lf¥n",crackstart);
        fprintf(fp,"Last Recording Crack     : %lf¥n",crackend);
        fprintf(fp,"1)Test     2)Simulation  : %d¥n",iftest);
        fprintf(fp,"1)K-Control  2)P-Control  : %d¥n",ctrl);
        fprintf(fp,"Load Frequency           : %lf¥n",freq);
        fprintf(fp,"YOUNG                    : %d¥n",YOUNG);
        fprintf(fp,"Delta P                  : %lf¥n",stress);
        fprintf(fp,"Delta K                  : %lf¥n",KVAL);
        fprintf(fp,"Load Factor              : %lf¥n",loadfact);
        fprintf(fp,"COD Factor               : %lf¥n",codfact);
        fclose(fp);
}


/***punch.h**************
punch.lib header file

************************/
extern  int setini(void );
extern  int simuortest(int *iftest);
extern  int k_or_p(int *ctrl,double *stress,double *KVAL);
extern  int kvalin(double *KVAL);
extern  int setcondi(unsigned char *idnum,double *thick,double *icrack);
extern  int conformini(void );
extern  int specimendsp(void );
extern  int specimenc(unsigned char *idnum,double *width,double *thick,double *icrack,double
*crackstart,double *crackend);
extern  int loaddsp(void );
extern  int loadc(double *freq,int *ctrl);
extern  int otherdsp(void );
extern  int otherc(unsigned char *fname1,unsigned char *fname2,int *YOUNG,double *loadfact,
double *codfact,int *iftest);
extern  int everyfine(void );


/***condi.c*********

setini()
simuortest()
k_or_p()
kvalin()
setcondi()

**********************/
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <math.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include "punch.h"
#include "p9128.h"
```

```
#include "value.h"

int setini(void)
{
        text_clr();
        locate(0,3);
        printf("    ---------- SET CONDITIONS ----------¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("                                        ¥n");
        printf("    ----------------------------------------¥n¥n");
        return(0);
}


int simuortest(int *iftest)  /** unresolved  **/
{
        int aa;

        text_clr();
        setini();
        locate(0,5);
        printf("      THIS PROGRAM APPLIES TO   (1 or 2)          ¥n¥n");
        printf("      1) Loading Test                             ¥n");
        printf("      2) Simulation                               ¥n");
        locate(44,5);
        for(aa='0'; ((aa!='1')&&(aa!='2'));)
        {
                aa=getch();
                if(aa==ESC) break;
        }
        switch(aa)
        {
                case '1':  *iftest=1; break;
                case '2':  *iftest=2; break;
                case ESC:          return(0);     /**   unresolved  **/
        }
        return(0);
}


int k_or_p(ctrl,stress,KVAL)

int *ctrl;
double *KVAL,*stress;
{
        int  aa;
        text_clr();
        setini();
        locate(0,5);
        printf("      CHOSE FOLLOWING TEST TYPE (1 or 2)  ¥n¥n");
        printf("      1) CONSTANT delta K CONTROL TEST                ¥n");
        printf("      2) CONSTANT STRESS CONTROL TEST                   ¥n");
```

```
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        locate(44,5);
        for(aa='0'; ((aa!='1')&&(aa!='2'));)
        {
                aa=getch();
                if(aa==ESC) break;
        }
        switch(aa)
        {
                case '1': *ctrl=1; kvalin(KVAL); break;
                case '2': *ctrl=2;
                                        locate(0,11);
                                        printf("     INPUT  delta P¥n");
                                        printf("       delta P =       [kg]");
                                        *stress = keyin(21,12);
                                        break;
                case ESC:       return(0);
        }
        return(0);
}

int kvalin(KVAL)
double *KVAL;
{
        int aa;

        locate(0,5);
        printf("     CHOSE A K-VALUE AND PRESS THE NUMBER (1-5)  ¥n¥n");
        printf("     1)  70 [kgf/ mm(2/3)]    2)  90 [kgf/ mm(2/3)]  ¥n");
        printf("     3) 110[kgf/ mm(2/3)]     4) 140 [kgf/ mm(2/3)]  ¥n");
        printf("      5) OTHER                                      ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        printf("                                                    ¥n");
        locate(49,5);
        for(aa='0'; ((aa<'1')||(aa>'5'));)
        {
                aa=getch();
                if(aa==ESC) break;
        }
        switch(aa)
        {
                case '1': *KVAL=70.0; break;
                case '2': *KVAL=90.0; break;
                case '3': *KVAL=110.0; break;
                case '4': *KVAL=140.0; break;
                case '5': locate(0,11);
                                        printf("     INPUT  K-VALUE ¥n");
                                        printf("       K-VALUE (double) = ");
                                        *KVAL=keyin(28,12);
                                        break;
                case ESC:       return(0);
        }
        return(0);
}
```

```
int setcondi(idnum,thick,icrack)
char idnum[];
double  *thick;
double  *icrack;
{
        char cbuf[20];
        int ika;
        double uni;
        setini();
        locate(0,5);
        printf("      INPUT FOLOWING CONDITIONS                    ¥n¥n");
        printf("      1) Specimen ID number   :                    ¥n");
        printf("      2) Specimen Thickness:      [mm]             ¥n");
        printf("      3) Initial Crack Length :     [mm]           ¥n");

        do{
                locate(34,7);  printf("           "); strcpy(cbuf,keyinstr(34,7));
                if((*cbuf)!=0) strcpy(idnum,cbuf);
                locate(34,7); space(15); printf("%s",idnum);

                locate(34,8);  printf("      "); uni=keyin(34,8);
                if(uni!=0.) *thick=uni;
                locate(34,8); space(7); printf("%4.2lf",*thick);

                locate(34,9);  printf("      ");  uni=keyin(34,9);
                if(uni!=0.) *icrack=uni;
                locate(34,9); space(7); printf("%4.2lf",*icrack);

                printf("¥n¥n      Everything Fine ? ");
                ika=getch();
        } while((ika!='y')&&(ika!='Y')&&(ika!=CR));
        return(0);
}


/***confrm.c**************************

conformini()
specimendsp()
specimenc()
loaddsp()
loadc()
otherdsp()
otherc()
everyfine()

************************************/
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <math.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include "p9128.h"
#include "punch.h"
#include "value.h"
```

```
int conformini(void)
{
        text_clr();
        locate(0,3);
        printf("  ********** CONFORMATION OF CONDITIONS **********¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf("  *                                                *¥n");
        printf(" ************************************************************¥n");
        printf("                                                   ¥n");
        printf("                                                   ¥n");
        printf("                                                   ¥n");
        printf("                                                   ¥n");
        printf("                                                   ¥n");
        printf("                                                   ¥n");
        return(0);
}


int specimendsp(void)
{
        locate(0,5);
        printf("  *   SPECIMEN                                        ¥n");
        printf("                                                  ¥n");
        printf("  *   1) Specimen ID number   :                     ¥n");
        printf("  *   2) Specimen Width       :       [mm]          ¥n");
        printf("  *   3) Specimen Thickness   :       [mm]          ¥n");
        printf("  *   4) Initial Crack Length :      [mm]           ¥n");
        printf("  *   5) 1st Recording Crack  :       [mm]          ¥n");
        printf("  *   6) last Recording Crack :       [mm]          ¥n");
        printf("  *   7) Next                                  ¥n");
        return(0);
}


int specimenc(idnum,width,thick,icrack,crackstart,crackend)
char idnum[];
double *width,*thick,*icrack,*crackstart,*crackend;
{
        int fine;
        char cbuf[20];
        double uni;

        text_clr();
        conformini();
        specimendsp();
        locate(34,7);       printf("%s",idnum);
        locate(34,8);       printf("%3.2lf",*width);
        locate(34,9);       printf("%3.2lf",*thick);
        locate(34,10);      printf("%3.2lf",*icrack);
        locate(34,11);      printf("%3.2lf",*crackstart);
```

```
        locate(34,12);      printf("%3.2lf",*crackend);

        while(1)
        {
                fine=everyfine();
                switch(fine)
                {
                        case 0:   return(999);
                        case 1: locate(6,15);printf("Specimen ID number =              ");
                                strcpy(cbuf, keyinstr(28,15));
                                if((*cbuf)!=0) strcpy(idnum,cbuf);
                                locate(34,7);       space(10);          printf("%s",idnum);
                                break;
                        case 2: locate(6,15);printf("Specimen Width =                 ");
                                uni=keyin(24,15); if(uni!=0.) *width=uni;
                                locate(34,8);       space(10);          printf("%3.2lf",*width);
                                break;
                        case 3: locate(6,15);printf("Specimen Thickness =             ");
                                uni=keyin(28,15); if(uni!=0.) *thick=uni;
                                locate(34,9);       space(10);          printf("%3.2lf",*thick);
                                break;
                        case 4: locate(6,15);printf("Initial Crack Length =           ");
                                uni=keyin(30,15); if(uni!=0.) *icrack=uni;
                                locate(34,10);      space(10);          printf("%3.2lf",*icrack);
                                break;
                        case 5: locate(6,15);printf("1st Recording Crack =            ");
                                uni=keyin(29,15); if(uni!=0.) *crackstart=uni;
                                locate(34,11);      space(10);          printf("%3.2lf",*crackstart);
                                break;
                        case 6: locate(6,15);printf("last Recording Crack =           ");
                                uni=keyin(30,15); if(uni!=0.) *crackend=uni;
                                locate(34,12);      space(10);          printf("%3.2lf",*crackend);
                                break;
                        case 7:   conformini(); return(0);
                        default:  conformini(); fine=999; break;
                }
        }
        return(0);
}


int loaddsp(void)
{
        locate(0,5);
        printf(" *  LOADING CONDITIONS                                  ¥n¥n");
        printf(" *  1) Frequency          :          [Hz]    ¥n");
        printf(" *  2) Controled by       :                  ¥n");
        printf(" *  3)                                       ¥n");
        printf(" *  4)                                       ¥n");
        printf(" *  5)                                       ¥n");
        printf(" *  6)                                       ¥n");
        printf(" *  7) Next                                    ¥n");
        return(0);
}

int loadc(freq,ctrl)
double *freq;
int *ctrl;
{
        int fine,ika;
```

```
        /* double uni; */

        text_clr();
        conformini();
        loaddsp();
        locate(34,7);       printf("%3.2lf",*freq);
        if (*ctrl==1) locate(34,8);  printf("¥"delta K¥"");
        if (*ctrl==2) locate(34,8);  printf("¥"delta P¥"");

        while(1)
        {
                fine=everyfine();
                switch(fine)
                {
                        case 0:   return(999);
                        case 1: locate(6,15);
                                        printf("cannot change!¥n");
                                        break;

                        /*              locate(6,15);printf("Load Frequency =                ");
                                        uni=keyin(24,15); if(uni!=0.) *freq=uni;
                                        locate(34,7);       space(10);          printf("%2.1lf",*freq);
                                        break;
                        */
                        case 2: locate(6,15);
                                        printf("Controled by  1) delta K  2) delta P¥n");
                                        locate(6,16);
                                        printf("PRESS  ( 1 or 2 ) ");
                                        ika=0;
                                        while((ika!='1')&&(ika!='2'))
                                        {
                                                locate(34,16);
                                                ika=getch();
                                                locate(34,16);space(10); /* delete eccor back num  */
                                        }
                                        locate(6,15); space(50);
                                        locate(6,16); space(50);
                                        locate(34,8); space(10);
                                        if (ika=='1')
                                        {
                                                *ctrl=1;
                                                printf("¥"delta K¥"");
                                        }
                                        if (ika=='2')
                                        {
                                                *ctrl=2;
                                                printf("¥"delta P¥"");
                                        }
                                        break;
                        case 3: break;
                        case 4: break;
                        case 5: break;
                        case 6: break;
                        case 7: conformini(); return(0);
                        default: fine=999; break;
                }
        }
        return(0);
}
```

```
int otherdsp(void)
{
        locate(0,5);
        printf("  *   OTHERS                                    ¥n¥n");
        printf("  *   1) Recording Filename    :                 ¥n");
        printf("  *   2) Condition Filename    :                 ¥n");
        printf("  *   3) Young's Module       :                 ¥n");
        printf("  *   4) Calibration of Load   :                ¥n");
        printf("  *   5) Calibration of COD    :                 ¥n");
        printf("  *   6) This program applies to :               ¥n");
        printf("  *   7) Next                                 ¥n");
        return(0);
}

int otherc(fname1,fname2,YOUNG,loadfact,codfact,iftest)
char fname1[],fname2[];
int  *YOUNG,*iftest;
double *loadfact,*codfact;
{
        int fine,ika;
        char cbuf[20];
        double uni;

        text_clr();
        conformini();
        otherdsp();

        locate(34,7);     printf("%s",fname1);
        locate(34,8);     printf("%s",fname2);
        locate(34,9);     printf("%5d",*YOUNG);
        locate(34,10);    printf("%4.2lf",*loadfact);
        locate(34,11);    printf("%4.2lf",*codfact);
        locate(34,12);    printf("    ¥"Loading Test¥"   ");

        while(1)
        {
                fine=everyfine();
                switch(fine)
                {
                        case 0:   return(999);
                        case 1: locate(6,15);printf("Recording Filename =            ");
                                strcpy(cbuf, keyinstr(28,15));
                                if((*cbuf)!=0) strcpy(fname1,cbuf);
                                locate(34,7);     space(10);     printf("%s",fname1);
                                break;
                        case 2: locate(6,15);printf("Condition Filename =            ");
                                strcpy(cbuf, keyinstr(28,15));
                                if((*cbuf)!=0) strcpy(fname2,cbuf);
                                locate(34,8);     space(10);     printf("%s",fname2);
                                break;
                        case 3: locate(6,15);printf("Young's Module =                ");
                                ika=keyinint(25,15); if(ika!=0) *YOUNG=ika;
                                locate(34,9);     space(10);     printf("%5d",*YOUNG);
                                break;
                        case 4: locate(6,15);printf("Load Calibration =              ");
                                uni=keyin(27,15); if(uni!=0.) *loadfact=uni;
                                locate(34,10);    space(10);     printf("%4.2lf",*loadfact);
                                break;
                        case 5: locate(6,15);printf("COD Calibration =               ");
                                uni=keyin(26,15); if(uni!=0.) *codfact=uni;
```

```
                                        locate(34,11);      space(10);          printf("%4.2lf",*codfact);
                                        break;
                        case 6: locate(6,15);
                                        printf("This program is used for                ¥n");
                                        locate(6,16);
                                        printf("1) Loading Test  2) Simulation  (1 or 2) ");
                                        ika=0;
                                        while((ika!='1')&&(ika!='2'))
                                        {
                                                locate(48,16);
                                                ika=getch();
                                                locate(48,16); space(5); /* delete eccor back num */
                                        }
                                        locate(6,15); space(50);
                                        locate(6,16); space(50);
                                        locate(38,12);      space(16);
                                        if (ika=='1')
                                        {
                                                *iftest=1;
                                                printf("¥"Loading Test¥"      ");
                                        }
                                        if (ika=='2')
                                        {
                                                *iftest=2;
                                                printf("¥"Simulation¥"        ");
                                        }
                                        break;
                        case 7: return(0);
                        default:  conformini(); fine=999; break;
                }
        }
        return(0);
}


int everyfine(void)
{
        char buf[2];
        int cc;

        locate(6,15);
        printf("Select a Number You Want to Change ");
        cc=0;
        while(cc<'1'||cc>'7')
        {
                locate(6,15);  printf("Select a Number You Want to Change      ");
                locate(41,15); cc=getch();
                if (cc==ESC) return(0);
        }
        buf[0]=(char)cc;
        buf[1]='¥0';
        cc=atoi(buf);
        return(cc);
}
```

```
<<<  実験制御プログラム  >>>
/****************start.c**********************
  ファジィ制御  ⊿K一定疲労き裂伝播試験

    横浜国立大学  工学部  生産工学科

          板垣  研究室


          Feb.1,1991

      プログラマー  パんち、浪岡、八田
      コンパイル    MS-C Ver5.1
                   /FPi /AS
      ライブラリー  P9128.LIB
*******************************************/
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <math.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include "start.h"
#include "p9128.h"
#include "value.h"
#include "fuz.h"

#define  GAMP  1
#define  DELAY(n) { register unsigned int i; i=n; while(i--); }

void main(void)
{
        int ika;
        int hatta;

        text_clr();
        readdat();
        if(freq==2.0) frequency=0x1d;
        else {
                printf("FREQENCY MUST BE 2.0 [Hz]");
                exit(999);
        }
        printpdini();

        locate(40,3); printf("  PRINTER OK ? ");
        for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
        {       if ((ika=='n')||(ika=='N')) exit(999);  }
        locatespace(40,3,40,1);

        crack = icrack;
        ccrack = icrack;
        rcrack = crackstart;
        kval = KVAL;
        setsine(sd);
        pvalue(&stress,crack,kval,width,thick);
        STRESS = stress;
        fact = 4095;  /*  Jan 8   rewrite  */

        locate(0,0);
```

```
        writedat();
        locate(40,3); printf("  CONDITIONS OK ? ");
        for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
        {       if ((ika=='n')||(ika=='N')) exit(999);  }
        locatespace(40,3,40,1);

        text_clr();
        crtonload();

#if GAMP==1
        if (1 == gain_ini())
        {
                vstr_set(40,11,"PGAMP initialized",RED|NORMAL);
                printf("¥7");
                gain=2;
                vstr_set(40,11,"                ",RED|NORMAL);
        }
        else
        {
                vstr_set(40,11,"cannot initialized PGAMP",RED|NORMAL);
                printf("¥7¥7¥7¥7");
                exit(999);
        }
#endif

        locate(40,3); printf(" ------ ADJUST 'SETPOINT' (MTS) ------ ");
        locate(40,5); printf("  INITIAL LOAD = 200 [kg]  ( 0.2 [v] ) ");
        locate(40,7); printf(" ---------------------------------- ");
        locate(40,8); printf("  OK ? ");
        for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
        {       if ((ika=='n')||(ika=='N')) exit(999);  }
        locatespace(40,3,40,7);

        locate(40,3); printf(" -------- ADJUST 'SPAN' (MTS) -------- ");
        locate(40,5); printf("  'SPAN' = 0                      ");
        locate(40,7); printf(" ---------------------------------- ");
        locate(40,8); printf("  OK ? ");
        for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
        {       if ((ika=='n')||(ika=='N')) exit(999);  }
        locatespace(40,3,40,7);

        factin(4095);
        starttimer2(frequency);
        locate(40,3); printf(" ------- ADJUST 'SPAN' (MTS) -------  ");
        locate(40,5); printf("  DELTA P = %4.0lf [kg]            ",STRESS);
        locate(40,7); printf(" ---------------------------------- ");
        locate(40,8); printf("  OK ? ");
        for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
        {       if ((ika=='n')||(ika=='N')) exit(999);  }
        locatespace(40,3,40,7);
        cursorsw(0);

        /***************** BEGINING OF TEST *****************/
        cycle = 0;
        xcyclecnt = 0;
        ycyclecnt = 0;

        /****** before first13 *****/
        FUZZY=0;
        while( ccrack<crackstart )  adloop();
```

```
        FUZZY=1;
        /*************************/

        if ( FUZZY ){ for (cnt=0; cnt<13; ) first13(); }

        /************ JOINT ****************************************/
        hatta=1;
        vstr_set(40,11,"        ",RED¦NORMAL);
        while( hatta )
        {
                vstr_set(40,10,"AD LOOP",RED¦NORMAL);
                printxycycle(cycle, xcyclecnt, ycyclecnt);
                locate(60,12); printf("cnt=%ld",cnt);
                if ( ccrack >= rcrack )
                {
                        ad();
                        xcyclecnt = 0;
                        rcrack+=0.4;
                        locate(1,4) ;printf("Next recording crack : %3.2lf[mm]",rcrack);
                        vstr_set(40,10,"        ",RED¦NORMAL);
                        hatta=0;
                }
                if (1==lookkey())
                {
                        ad();
                        changek(&fact,KVAL,kval);
                        factin(fact);
                }
                ad(); /**** AD conversion *****/

                if (kval >= (KVAL*1.004))
                {
                        ycyclecnt = 0;
                        changek(&fact,KVAL,kval);
                        factin(fact);
                        if ( FUZZY ){printf("¥7");fuzzy();}
                }
        }
        /*****************************************************************/

        while(1)
        {
                if ( FUZZY ) onload();
                else        adloop();
        }

        /****************** END OF TEST ******************/
        cursorsw(1);
        exit(0);
}


void scroll(long cycle, double crack, double stress, double kval)
{
        int i,j;
        static double dd[5],cc[5],bb[5];
        static long aa[5];

        for(i=1;i<5;i++)
        {
```

```
                dd[i-1] = dd[i];
                cc[i-1] = cc[i];
                aa[i-1] = aa[i];
                bb[i-1] = bb[i];
        }
        aa[4] = cycle;
        bb[4] = crack;
        cc[4] = stress;
        dd[4] = kval;

        for(j=0;j<5;j++){ locate(1,19+j);  printf("Cycle = %ld",aa[j]);}
        for(j=0;j<5;j++){ locate(21,19+j); printf("Crack = %2.2lf",bb[j]);  }
        for(j=0;j<5;j++){ locate(41,19+j); printf("Stress = %4.0lf",cc[j]); }
        for(j=0;j<5;j++){ locate(61,19+j); printf("K-value = %3.2lf",dd[j]);}

}


void crtonload()     /* 実験の初期画面 */
{
        locate(2,1)  ;printf(">>> SPECIMEN I.D. : %s <<<",idnum);
        locate(2,3)  ;printf("< RECORDING CRACK >");
        locate(1,4)  ;printf("Next recording crack  : %3.2lf[mm]",rcrack);
        locate(1,5)  ;printf("First recording crack : %3.2lf[mm]",crackstart);
        locate(1,6)  ;printf("Last recording crack  : %3.2lf[mm]",crackend);
        locate(2,7)  ;printf("    < CONTROL >");
        locate(1,8)  ;printf("Stress : ");
        locate(11,8) ;if(ctrl == 1) printf("K-control");
                                    else if (ctrl == 2 ) printf("P-control");
        locate(1,9) ;printf("AD     : ");
        locate(11,9);if(FUZZY == 1) printf("Fuzzy");
                     else if(FUZZY == 0) printf("Normal");
        locate(2,10) ;printf("    < FACTORS >");
        locate(1,11) ;printf("Young's module : %5d",YOUNG);
        locate(1,12) ;printf("Initial stress : %4.1lf[kgf]",STRESS);
        locate(2,13);printf("     < PGAMP >");
        locate(1,14);printf("Gain    : %d",gain);
        locate(2,15);printf("  < MTS CONTROL >");
        locate(1,16);printf("Frequency      : %1.1lf[Hz]",freq);
        locate(1,17);printf("Target K-value : %4.2lf[kgf/mm3/2]",KVAL);

        locate(42,13);printf("Cycle   :%7ld",cycle);
        locate(42,14) ;printf("     <NEXT K-CORRECTION CYCLE>");
        locate(42,15) ;printf("Ycycle :%5ld  Count :%5ld",ycycle,ycyclecnt);
        locate(42,16) ;printf("      <NEXT RECORDING CYCLE>");
        locate(42,17) ;printf("Xcycle :%5ld  Count :%5ld",xcycle,xcyclecnt);

        locate(9,24);printf("[cycle]");
        locate(29,24);printf("[mm]");
        locate(49,24);printf("[kgf]");
        locate(69,24) ;printf("[kgf/mm3/2]");
}

int printxycycle(long cycle, long xcyclecnt, long ycyclecnt)
{
        locate(51,13); printf("%7ld",cycle);
        locate(64,15); printf("%5ld",ycyclecnt);
        locate(64,17); printf("%5ld",xcyclecnt);
        return(0);
}
```

```
int printpdini()
{
        FILE *fp;
        char time[10],date[10];
        int ika;

        _strtime(time);
        _strdate(date);

/*-------- disket ----------*/
        fp=fopen(fname2,"r");
        if( fp != NULL )
        {
                locate(40,3); printf("  OVER WRITE '%s' OK ? ",fname2);
                for(ika=0;(ika!='y')&&(ika!='Y')&&(ika!=CR);ika=getch())
                {       if ((ika=='n')||(ika=='N')) exit(999);  }
                locate(40,3); printf("  WAIT !!                 ");
                if( fclose(fp)== EOF ) printf("¥n Can not close file¥n");
        }

        fp=fopen(fname2,"w+");
        if( fp == NULL )
        {
                printf("¥n Can not open file¥n");
                exit(999);
        }
        else {
                fprintf(fp,"¥n¥n");
                fprintf(fp,"/*************************************************¥n¥n");
                fprintf(fp,"    ファジィ制御   ⊿Ｋ一定疲労き裂伝播試験¥n¥n");
                fprintf(fp,"    横浜国立大学   工学部   生産工学科   板垣研究室¥n¥n");
                fprintf(fp,"    DATE %s    TIME %s ¥n¥n",date,time);
                fprintf(fp,"    試験片番号   : %s¥n",idnum);
                fprintf(fp,"/*************************************************3**¥n");
                printf("¥007");
        }
        if( fclose(fp)== EOF ) printf("¥n Can not close file¥n");

/* --------- printer --------- */

        fp=fopen("PRN","a+");
        if( fp == NULL )
        {
                printf("¥n Can not open printer¥n");
                exit(999);
        }
        else {
                fprintf(fp,"¥n¥n¥n¥n");
                fprintf(fp,"*********************************************************¥n¥n");
                fprintf(fp,"    ファジィ制御   ⊿Ｋ一定疲労き裂伝播試験¥n¥n");
                fprintf(fp,
                                "      横浜国立大学   工学部   生産工学科   板垣研究室¥n¥n");
                fprintf(fp,"    DATE %s    TIME %s ¥n¥n¥n",date,time);
                fprintf(fp,"     試験片番号       : %s¥n¥n",idnum);
                fprintf(fp,"     試験機振動数     : %2.1lf¥n",freq);
                fprintf(fp,"     ヤング率         : %5d¥n",YOUNG);
                fprintf(fp,"     ⊿Ｋ              : %3.1lf¥n",KVAL);
                fprintf(fp,"¥n");
                fprintf(fp,"*********************************************************¥n");
```

```
                fprintf(fp,"¥n¥n¥n");
                printf("¥007");
        }
        if( fclose(fp)== EOF ) printf("¥n Can not close printer¥n");
        return(0);
}

int readdat()
{
        FILE *fp;
        char ika[30];

        locate(0,0); printf("INPUT SPECIMEN ID");
        locate(0,3); printf("ID number =");
        strcpy(fname1,keyinstr(13,3));
        strcat(fname1,".cnd");
        if (NULL==(fp=fopen(fname1,"r")))
        {
                printf("¥nCannot Open Condition File  %s¥n¥7¥7¥7¥7",fname1);
                exit(999);
        }
        fscanf(fp,"%29c %s",ika,idnum);
        fscanf(fp,"%29c %s",ika,fname1);
        fscanf(fp,"%29c %s",ika,fname2);
        fscanf(fp,"%29c",ika);
        fscanf(fp,"%29c %lf",ika,&width);
        fscanf(fp,"%29c %lf",ika,&thick);
        fscanf(fp,"%29c %lf",ika,&icrack);
        fscanf(fp,"%29c",ika);
        fscanf(fp,"%29c %lf",ika,&crackstart);
        fscanf(fp,"%29c %lf",ika,&crackend);
        fscanf(fp,"%29c %d",ika,&iftest);
        fscanf(fp,"%29c %d",ika,&ctrl);
        fscanf(fp,"%29c %lf",ika,&freq);
        fscanf(fp,"%29c %d",ika,&YOUNG);
        fscanf(fp,"%29c %lf",ika,&stress);
        fscanf(fp,"%29c %lf",ika,&KVAL);
        fscanf(fp,"%29c %lf",ika,&loadfact);
        fscanf(fp,"%29c %lf",ika,&codfact);
        fclose(fp);
        return(0);
}

int writedat()
{
        locate(0,0);
        printf("¥n¥n");
        printf("Specimen ID Number  : %s¥n",idnum);
        printf("Condition File Name : %s¥n",fname1);
        printf("Test Data File Name : %s¥n",fname2);
        printf("------ Specimen Size ------¥n");
        printf("Width       : %lf¥n",width);
        printf("Thickness   : %lf¥n",thick);
        printf("Initial Crack : %lf¥n",icrack);
        printf("------ Test Condition -----¥n");
        printf("First Recording Crack :%lf¥n",crackstart);
        printf("Last Recording Crack  :%lf¥n",crackend);
        printf("1)Test      2)Simulation  : %d¥n",iftest);
        printf("1)K-Control 2)P-Control   : %d¥n",ctrl);
        printf("Load Frequency : %lf¥n",freq);
```

```
        printf("YOUNG  :%d¥n",YOUNG);
        printf("Delta P : %lf¥n",stress);
        printf("Delta K : %lf¥n",KVAL);
        printf("Load Factor : %lf¥n",loadfact);
        printf("COD Factor  : %lf¥n",codfact);
        return(0);
}


int menu(void)
{
        int c,i;

        locatespace(40,1,41,9);
        locate(40,1); printf("---------- Change Menu ----------");
        locate(40,2); printf("  1: Correction Crack Length      ");
        locate(40,3); printf("  2: Abandon Test                 ");
        locate(40,4); printf("  3: Change Xcycle                ");
        locate(40,5); printf("  4: YOUNG'S MODULE                    ");
        locate(40,6); printf("  5: AD-conversion                ");
        locate(40,7); printf("----------------------------------");
        locate(40,8); printf(" Sellect (1-5) ");
        for(c='0'; ((c<'1')||(c>'5'));)
        {
                while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
                c=getch();
                if(c==ESC) break;
        }
        for(i=1; i<9; i++) {
                locate(40,i); space(40);
        }
        return(c);
}

int lookkey()
{
        FILE *fp;
        int keybuf;
        char end[10],dumy1[10],dumy2[10];

        if ( kbhit()==0 ) return(0);
        keybuf = getch();
        if (keybuf != ESC) return(0);
        switch(menu())
        {
        case '1':
                if( 1 == changec(crack,&crackfact,cycle,xcyclecnt,ycyclecnt))
                {
                        fp=fopen("PRN","a+");
                        if( fp == NULL )
                        {
                                vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                                printf("¥7¥7¥7¥7¥7");
                        }
                        else {
                                fprintf(fp," >>>>>>> Change correction factor !!! <<<<<<<<¥n");
                                fprintf(fp,"measured crack length = %2.2lf¥n", crack*crackfact);
                        }
                        if(fclose(fp)==EOF) vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                        return(1);
```

```
            }
            break;

    case '2':
            locate(40,1); printf("ABANDON TEST ? ");
            locate(40,3); printf("Type 'END' and press 'CR'. ");
            locate(40,4); printf("-->");
            while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
            strcpy(end,keyinstr(44,4));
            strcpy(dumy1,"end"); strcpy(dumy2,"END");
            if ((strcmp(dumy1,end)==0 )||(strcmp(dumy2,end)==0)) tstop(fact);
            else locatespace(40,1,30,4);
            break;

    case '3':
            if( 1 == changexcycle( &xcycle, cycle, xcyclecnt, ycyclecnt))
            {
                    locate(51,17) ;printf("%5ld",xcycle);
                    fp=fopen("PRN","a+");
                    if( fp == NULL )
                    {
                            vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                            printf("¥7¥7¥7¥7¥7");
                    }
                    else {
                            fprintf(fp," >>>>>>> Change XCYCLE  !!! <<<<<<<<¥n");
                    }
                    if(fclose(fp)==EOF) vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                    return(0);
            }
            break;

    case '4':
            if( 1 == changeyoung( &YOUNG, cycle, xcyclecnt, ycyclecnt))
            {m
                    locate(1,11) ;printf("Young's module : %5d",YOUNG);
                    fp=fopen("PRN","a+");
                    if(fp == NULL){
                            vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                            printf("¥7¥7¥7¥7¥7");
                    }
                    else {
                            fprintf(fp," >>>>>>> Change Young's Module  !!! <<<<<<<<¥n");
                            fprintf(fp,"New Young's Module = %5d¥n",YOUNG);
                    }
                    if(fclose(fp)==EOF) vstr_set(40,0,"PRINTER ERROR !!!",RED|NORMAL);
                    return(1);
            }
            break;

    case '5':
            adflag=1;
            ad();
            break;

    default : break;
    }
    return(0);
}
```

```
static void interrupt far dac98( void )
{
        int al, ah, data;
        static int ndata;
        _disable();                                     /*  全ての割り込みを禁止する */
        if( ndata >= 100)                       /* データバッファの先頭に戻る*/
        {
                ndata = 0;
                cycle++;
                xcyclecnt++;
                ycyclecnt++;
        }
        count++;
        data = sd[ndata++];                             /* サイン波のデータ*/
        al = ( data & 0x00ff );                         /* データの下位８ビット*/
        ah = ( (unsigned)data >> 8 );                   /* データの上位８ビット*/
        outp( CHSEL, CHANNEL0 );        /* チャネル０を選択*/
/*      DELAY(1);*/
        outp( LBYTE, al );                              /* 下位８ビットを出力*/
/*      DELAY(1);*/
        outp( HBYTE, ah );                              /* 上位８ビットを出力*/
        _enable();                                      /* 全ての割り込みを許可する */
        outp( IC, EOI );        /* 割り込み終了をインタラプトコントローラ に知らせる*/
        return;
}


int starttimer2(int frequency)
{
        int factor,i,ctrl;

        vectbuff2 = _dos_getvect( INT_NO ); /*割り込みベクタの保存*/
        _disable();
        _dos_setvect( INT_NO, dac98 );      /*割り込みルーチンのセット*/
        _enable();
        ctrl = 0x07;
        outp( CONTROL, ctrl );
/*      DELAY(1);*/
        outp( FREQSEL, frequency );
        factor=fact;
        locate(0,24);
        for(i=0; i<500; i++)                            /*  increase load  */
        {
                factor = (int)((float)factor*0.99);
                factin(fact-factor);
                printf("wait !¥r");
        }
        printf("     ¥r");
        return(0);
}


int stoptimer2(int fact)
{
        int i;

        locate(0,24);
        for(i=0; i<500; i++)                            /*  decrease load  */
        {
```

```
                fact = (int)((float)fact*0.99);
                factin(fact);
                printf("wait !¥r");
        }
        factin(0);
        outp( CONTROL, 0x00 );
        _disable();
        _dos_setvect( INT_NO, vectbuff2 );  /*割り込みベクタを元に戻す*/
        _enable();
        return(0);
}


static void interrupt far inttimer()            /* interrupt handler (PD8253C)  */
{
        if (timerflag!=1) timerflag=1;

        timer++;
        outp(mpd8253,intvcount & 0XFF);   /*  load to intarval timer  */
        DELAY(1);
        outp(mpd8253,intvcount >> 8);
        DELAY(1);
        outp(mpd8259,EOI);                          /*  end of interrupt  */
}

int starttimer1(int *timer)                         /*  for PD8253C  (AD-conversion)  */
{
        *timer=0;
        vectbuff1 = _dos_getvect(0x08);
        intvcount = 24576;
        _disable();
        _dos_setvect(0x08,inttimer);
        outp(mpd8253+6,0x36);
        DELAY(1);
        outp(mpd8253,intvcount & 0xff);
        DELAY(1);
        outp(mpd8253,intvcount >> 8);
        DELAY(1);
        outp(mpd8259+2,inp(mpd8259+2) & ~0x01);
        _enable();
        return(0);
}

int stoptimer1(void)                                /*  for PD8253C  (AD-conversion)  */
{
        _disable();
        outp(mpd8259+2,inp(mpd8259+2) | 1);
        _dos_setvect(0x08,vectbuff1);
        _enable();
        return(0);
}


ad(void)
{
        int i;
        static double stangent[20];
        static double sdeltap[20];

        tangent=0.0;
```

```
        deltap=0;

        for( i=0; i<=19; i++ )
        {
                printxycycle(cycle, xcyclecnt, ycyclecnt);
                if ( cnt==0 || cnt>12) lookkey();
                getpv();
                lsqm(samplep,samplev,point,&tangent,loadfact,codfact,adfact);
                stangent[i]=tangent;
                sdeltap[i]=(double)deltap;
        }
        sort1(stangent,20);
        sort1(sdeltap,20);

        tangent = (stangent[3]+stangent[4]+stangent[5]+stangent[6]+stangent[7]+stangent[8]+stangent[9]+stangent[10]
                +stangent[11]+stangent[12]+stangent[13]+stangent[14]+stangent[15]+stangent[16])/14.0;
        deltap = (int)((sdeltap[3]+sdeltap[4]+sdeltap[5]+sdeltap[6]+sdeltap[7]+sdeltap[8]+sdeltap[9]+sdeltap[10]
                +sdeltap[11]+sdeltap[12]+sdeltap[13]+sdeltap[14]+sdeltap[15]+sdeltap[16])/14.0 + 0.5);

        locate(60,11); printf("tangent=%1.3e",tangent);
        stress = (double)deltap*(double)loadfact/(double)adfact;
        avalue(tangent,&crack,width,thick,YOUNG);
        ccrack = crack*crackfact;
        kvalue(stress,ccrack,&kval,width,thick);
        printpd(cycle,ccrack,kval,stress,tangent,crackfact,cnt,fname2,preycycle,ycycle);
        scroll(cycle,ccrack,stress,kval);
        return(0);
}


int getpv(void)
{
        int p,v,pmax,pmin,vmax;

        pmax=0;
        vmax=0;
        pmin=4095;
        point=0;

        vstr_set(40,9,"AD CONVERSION                    ",RED|NORMAL);
        starttimer1(&timer);
        do{
                while (timerflag != 1) ;        /*  wait for timerflag changed  */
                adc(&p,&v);                     /*  p,v: integer (0-4095)  */
                if (pmax < p)   pmax = p;
                if (pmin > p)   pmin = p;
                if (vmax < v)       vmax = v;
                if ((p<lmth) && (p>lmtl))               /*  lmth: upper limit of p  */
                {
                        samplep[point] = p;
                        samplev[point] = (int)((float)v / (float)gain + 0.5);
                point++;
                }
                timerflag=0;
        } while (timer < 50);                   /* timer=50 for 1 cycle  when f=2[Hz] */
        stoptimer1();
        vstr_set(40,9,"                          ",BLACK|NORMAL);
        codmax = (double)vmax/adfact;  /* vmax: max cod value (0-4095)  */
        prog_gamp(codmax,&gain);  /* codmax: max voltage of cod   adfact=409.5 */
        locate(1,14);printf("Gain    : %d",gain);
```

```
        deltap = pmax - pmin;
        lmth = (int)((float)pmax * 0.8);
        lmtl = (int)((float)pmax * 0.1);
        return(0);
}


int fuzzy(void)
{
        adflag=1;
        ad();
        mcrack = crack;   /* mcrack: measured crack */
        mcycle = cycle;   /* mcycle: measured cycle */
        changek(&fact,KVAL,kval);
        factin(fact);
        fuzmain(mcrack,mcycle,&xcycle,&ycycle,cnt,&preycycle);
        cnt++;
        locate(51,15); printf("%5ld",ycycle);
        locate(51,17); printf("%5ld",xcycle);
        return(0);
}


int adloop(void)
{
        vstr_set(40,10,"AD LOOP",RED¦NORMAL);
        if ( ccrack > crackend ) tstop(fact);
        printxycycle(cycle, xcyclecnt, ycyclecnt);
        locate(60,12); printf("cnt=%ld",cnt);
        if ( ccrack >= rcrack )
        {
                ad();
                xcyclecnt = 0;
                rcrack+=0.4;
                locate(1,4)  ;printf("Next recording crack  : %3.2lf[mm]",rcrack);
                vstr_set(40,10,"        ",RED¦NORMAL);
                return(0);
        }
        if (1==lookkey())
        {
                ad();
                changek(&fact,KVAL,kval);
                factin(fact);
        }
        ad(); /****  AD conversion  *****/

        if (kval >= (KVAL*1.004))
        {
                ycyclecnt = 0;
                changek(&fact,KVAL,kval);
                factin(fact);
                if ( FUZZY ){printf("¥7");fuzzy();}
        }
        return(1);
}


int first13(void)
{
        vstr_set(40,11," FIRST13 ",RED¦NORMAL);
```

```
        if ( ccrack > crackend ) tstop(fact);
        printxycycle(cycle, xcyclecnt, ycyclecnt);
        locate(60,12); printf("cnt=%ld",cnt);
        ad();
        if ( ccrack >= rcrack )
        {
                rcrack+=0.4;
                locate(1,4) ;printf("Next recording crack  : %3.2lf[mm]",rcrack);
        }
        if  (kval >= (KVAL*1.004))
        {
                vstr_set(59,12,"FUZZY  ",RED¦NORMAL);
                printf("¥7");
                fuzzy();
                xcyclecnt = 0;
                ycyclecnt = 0;
                vstr_set(59,12,"       ",RED¦NORMAL);
        }
        return(0);
}


int onload(void)
{
        if ( ccrack > crackend ) tstop(fact);
        if (1==lookkey()) { while( 1==adloop() ); }
        printxycycle(cycle, xcyclecnt, ycyclecnt);
        locate(60,12); printf("cnt=%ld",cnt);
        if (ycyclecnt >= ycycle)
        {
                vstr_set(59,12,"FUZZY  ",RED¦NORMAL);
                printf("¥7");
                ycyclecnt = 0;
                fuzzy();
                vstr_set(59,12,"       ",RED¦NORMAL);
                return(0);
        }
        if (xcyclecnt < xcycle) return(0);
        else  while( 1==adloop() );  /* caution! adloop has 'return()'  */
        return(0);
}


/***start.h********

start header file

****************/
extern  int ad(void );
extern  int getpv(void );
extern  int onload(void );
extern  int menu(void );
extern  int fuzzy(void );
extern  int lookkey(void );
extern  int adloop(void );
extern  int first13(void );
extern  int printpdini(void );
extern  int readdat(void );
extern  int writedat(void );
static  void far _interrupt dac98(void );
```

```
extern  int starttimer2(int frequency);
extern  int stoptimer2(int fact);
static  void far _interrupt inttimer(void );
extern  int starttimer1(int *timer);
extern  int stoptimer1(void );
extern  int readdat(void );
extern  int writedat(void );
extern  void scroll(long cycle,double crack,double stress,double kval);
extern  void crtonload(void );
extern  int printxycycle(long cycle,long xcyclecnt,long ycyclecnt);
extern  int ad(void );
extern  int menu(void );
extern  int printpdini(void );
extern  int getpv(void );
extern  int fuzzy(void );
extern  int lookkey(void );
extern  int adloop(void );
extern  int first13(void );
extern  int onload(void );
extern  void main(void );
void (interrupt far *vectbuff1)();
void (interrupt far *vectbuff2)();

/***fuz.h**********

fuzmain  header file

*****************/
extern  int sort1(double *a,int n);
extern  void calgrade(double np2,double zop2,double pp2,double decp2,double incp2,double precpr,
double preccpr,double *nx,double *zox,double *px,double *decm,double
*nocm,double *incm,double *jjj);
extern  void calxcycle(double *boxa,long *boxn,long *xcycle);
extern  void center(double ans,double azo,double aps,double apb,double zop2,double jjj,long
*ycycle,double crack,long xcycle,double precpr,long *preycycle,double *cent);
extern  void datain(double crack,long cycle,double *boxa,long *boxn);
extern  void fuzmain(double crack,long cycle,long *xcycle,long *ycycle,long cnt,long *preycycle);
extern  int locate(int x,int y);
extern  int locatespace(int a,int b,int c,int d);
extern  void mkfs(double *boxa,long *boxn,double *np2,double *zop2,double *pp2,double
*decp2,double *incp2,double *precpr,double *preccpr);
extern  void fuzzyrule(double nx,double zox,double px,double decm,double nocm,double incm,
          double *ans,double *azo,double *aps,double *apb);
extern  int sort1(double *a,int n);
extern void printfuz(double np2,double zop2,double pp2,double decp2,double incp2,double jjj,
double precpr,double cent,double preccpr,long preycycle);


/***value.h*******************

試験制御プログラム変数一覧

*****************************/

#define         ESC                     0x1b
#define         BS                      0x08
#define         CR                      0x0d
#define         CHSEL           0xd8    /* チャネル選択ポート */
#define         CHANNEL0                0x00    /* チャネル０ */
#define         CONTROL                 0xda    /* ＤＡＣコントロールポート */
```

```
#define         FREQSEL             0xdb    /* 周波数選択ポート */
#define         LBYTE       0xdc    /* データ出力ポート（下位８ビット） */
#define         HBYTE       0xdd    /* データ出力ポート（上位８ビット） */
#define         IC                  0x00    /* インタラプトコントローラ */
#define         EOI                 0x26    /* 割り込み終了 */
#define         INT_NO      0x0e    /* 割り込み番号（拡張スロット＃２） */
#define         INPORT              0x02da  /* input port of MPDAC */
#define         OUTPORT             0X02d8  /* output port of MPDAC */
#define         mpd8253             0X71    /* timer controler */
#define         mpd8259             0X00    /* inter rupt controler */

/* カラーコード */
#define         BLACK       0x00
#define         BLUE                0x20
#define         RED                 0x40
#define         PURPLE              0x60
#define         GREEN       0x80
#define         SKYBLUE             0xa0
#define         YELLOW              0xc0
#define         WHITE       0xe0
/* 表示モードコード */
#define         NORMAL              0x01
#define         BLINK       0x03
#define         REVERSE             0x05
#define         UNDERLINE           0x09

/* specimen spec （試験片関連）    */
static double   width=100.0;            /* width: 試験片横幅 */
static double   thick=18.0;             /* thickness: 試験片厚さ */
static double   icrack=20.0;            /* initial crack: 初期亀裂長さ*/
static double   rcrack=25.0;            /* recording crack */
static double   ccrack;                 /* corrected crack: crack*crackfact */
static double   crackstart=25.0;    /* first recording crack: 記録開始亀裂長さ */
static double   crackend=60.0;      /* last recording crack: 最終亀裂長さ */

/* MTS controle （試験機制御関連）  */
static double   freq=2.0;           /* frequency: ＭＴＳ振動数 */
static double   KVAL=50.0;              /* target K-value: 制御⊿Ｋ値 */
static double   kval;                   /* feed back K-value: ＡＤ変換による⊿Ｋ値 */
static double   crack;                  /* crack length: 亀裂長さ */
static double   stress;                 /* stress: ⊿Ｐ    [kgf] */
static double   STRESS;                 /* STRESS: initial ⊿Ｐ  [kgf] */
static double   tangent;            /* tangent (P by V): [kg/mm] */
static int      frequency;          /* frequency: ＭＴＳ振動数CONSTANT */

/* factors （各定数、係数）  */
static double   loadfact=1000.;     /* 1[v] = 1000[kg], 0.125[mm] */
static double   codfact=0.125;      /* 1[v] = 0.125[mm] */
static double   adfact=409.5;       /* 4095 = 10 [v] */
static double   crackfact=1.0;              /* crack correction factor */
static int YOUNG=18000; /* young's module: ヤング率 */

/* files （ファイルネーム）  */
static char fname1[20]="ika.cnd"; /* cnd file: 初期条件ファイル */
static char fname2[20]="ika.dat"; /* data file: データ保存ファイル */
static char idnum[20];              /* spacimen ID: 試験片番号 */

/* flags （各フラグ）    */
static int adflag;                  /* ADC flag: ＡＤ変換フラグ */
static int timerflag=0;             /* PD8253C flag: ＡＤ変換タイミング用フラグ */
```

```
static int ctrl=1;                    /*  ctrl :  1)K-controle  2)P-control */
static int iftest=1;          /*  iftest : 1)loding test  2)simulation */
static int      FUZZY=1;

/* DAC  （ＤＡ変換、ＡＤ変換用） */
static int ndata;                           /* サイン波１周期分のデータナンバー */
static int sd[100];                   /* サイン波１周期分のデータバッファ */
static long count;                    /* interrupt number: ＤＡタイマーカウンター */
static long cycle;                    /* cycle number: サイクル数 */
static int fact;                            /* multiplyer of MPDAC */
static int intvcount;                       /* PD8253C timer: ＡＤタイマー間隔 */
static int timer;                           /* timer counter: ＡＤ変換タイマー用カウンター */
static int point;                           /* number of AD point: 上下カットした後のポイント数 */
static int samplep[128];              /* stress value sampled on AD: 荷重値 （0 - 4095）*/
static int samplev[128];              /* COD value sampled on AD: ＣＯＤ値（0 - 4095） */
static int deltap;                          /* delta P:  ⊿Ｐ（0 - 4095） */
static int lmth=4095;                       /* higher limit value of AD: LSQMデータ上限 */
static int lmtl=0;                    /* lower limit value of AD: LSQMデータ下限 */

/* PGAMP */
static int gain=4;            /*  PGAMP gain: 1,2,4,8,16 */
static double     codmax;           /* COD max [volt] */


/* FUZZY */
static long xcycle=1;                       /* adloop に入るまでのcycle数 */
static long ycycle=1;                       /* 次回ΔＫ修正までのcycle数 */
static long xcyclecnt=0;              /* xcycle counter */
static long ycyclecnt=0;              /* ycycle counter */
static long preycycle;                      /* 前回の実際の計算上の ycycle */
static long mcycle;                         /* measured cycle */
static double mcrack;                       /* measured crack */
static long cnt;                            /* fuzmain counter */



/***adc.c*************************************************************

adc()

関数が呼ばれるとＡＤボードに入力したアナログ値（ｐ，ｖ）をそれぞれ
（0-4095）の値に変換する
**********************************************************************/
#define         D0H             0xD0    /* I/O port d0h  MSB ON ADC*/
#define         D1H             0xD1    /* I/O port d0h  LSB ON ADC*/
#define         MSTART1         0x10    /* manual start command of channel 1 */
#define         MSTART2         0x11    /* manual start command of channel 2 */

#include <dos.h>
#include <conio.h>
#include "p9128.h"

int adc(int *p, int *v)
{
        int a1,a2,b1,b2;

        outp( D0H, MSTART1 ); /* PORT 0xd0: start AD-conversion of load */
        while( (inp( D1H )&0x80) !=0 );  /* bit 7 : conversion end signal */
        a1 = inp ( D1H );     /* the most significant 4 bits */
        a2 = inp ( D0H );     /* lsb */
```

```
        outp( D0H, MSTART2 );    /* AD-conversion of COD */
        while( (inp( D1H )&0x80) !=0 );
        b1 = inp ( D1H );
        b2 = inp ( D0H );

        *p =( (( a1<<8 ) & 0x0f00) | (a2 & 0x00ff) );
        *v =( ((b1 << 8) & 0x0f00) | (b2 & 0x00ff) );
        return(0);
}


/***avalue.c*************************************************

avalue()

tangent, width, thick, YOUNG よりき裂長さ（crack）を計算する
*********************************************************/
#define  C0  1.00104  /* constants of equation (1)    */
#define  C1  -4.72    /* which determines crack length  */
#define  C2  19.4
#define  C3  -249.
#define  C4  1320.
#define  C5  -2400.

#include <stdio.h>
#include <math.h>
#include  "p9128.h"

avalue(tangent,crack,width,thick,YOUNG)
        double tangent,*crack;    /*  tangent: [mm/kg]  */
        double width,thick;
        int    YOUNG;
{
        double  u, abyw;

        /*  computation of crack length, equation (1) */
        u=1. / ( sqrt(thick*(double)YOUNG*tangent)+1. ); /* tangent: [mm/kg] */
        abyw=C0+C1*u+C2*pow(u,2.)+C3*pow(u,3.)+C4*pow(u,4.)+C5*pow(u,5.);
        *crack=abyw*width;
        *crack = *crack * 100. + 0.5;
        *crack = floor(*crack) / 100.;
        return(0);
}


/***changec.c************************************************************

changec()

キーボードよりき裂長さの修正を行なう入力した場合１、しない場合０を返す
crackfactorのみ戻す
*************************************************************************/
#define         ESC             0x1b
#define         CR              0x0d
#define         RED             0x40
#define         NORMAL          0x01

#include <stdio.h>
#include <conio.h>
```

```
#include  "p9128.h"

int changec(crack, crackfact, cycle, xcyclecnt, ycyclecnt)
double crack,*crackfact;
long cycle, xcyclecnt, ycyclecnt;
{
        double ncrack;
        int ika;

        locate(40,1); printf("Input New Crack Length Measured    ¥n");
        locate(40,2); printf("Crack Length =");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        ncrack = keyin(55,2);
        locate(40,1); printf("Measured Crack Length = %5.2lf    ",ncrack);
        locatespace(40,2,38,2);
        printf("OK ? ");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        locatespace(40,1,38,3);
        ika=getch();
        if((ika!='y')&&(ika!='Y')&&(ika!=CR)) return(0);
        (*crackfact) = ncrack / crack;
        return(1);
}


/***changek.c**********

changek()

correction K-value
**********************/
#include  <math.h>
#include  "p9128.h"

changek(int *fact, double KVAL, double kval)
{
        *fact = (int)( (double)(*fact) * KVAL*0.997 / kval + 0.5);
        if(*fact>4095) *fact=4095;
        return(0);
}


/***changex.c***************

changex()

**************************/
#define         ESC             0x1b
#define         CR              0x0d

#include <stdio.h>
#include <conio.h>
#include  "p9128.h"

int changexcycle(long *xcycle, long cycle, long xcyclecnt, long ycyclecnt)
{
        int ika;
        long nami;

        locate(40,1); printf("Input Xcycle¥n");
```

```
        locate(40,2); printf("Xcycle = ");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        nami = keyinlong(48,2);
        locate(40,1); printf("New Xcycle = %6ld          ",nami);
        locatespace(40,2,38,2);
        printf("OK ? ");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        locatespace(40,1,38,3);
        ika=getch();
        if((ika!='y')&&(ika!='Y')&&(ika!=CR)) return(0);
        *xcycle = nami;
        return(1);
}


/***changey.c*************************************************************

changeyoung()

キーボードよりヤング率の修正を行なう入力した場合１、しない場合０を返す
*************************************************************************/
#define         ESC             0x1b
#define         CR              0x0d

#include <stdio.h>
#include <conio.h>
#include  "p9128.h"

int changeyoung(int *YOUNG, long cycle, long xcyclecnt, long ycyclecnt)
{
        int ika,young;

        locate(40,1); printf("Input Young's Module¥n");
        locate(40,2); printf("YOUNG = ");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        young = keyinint(48,2);
        locate(40,1); printf("New Young's Module = %4d     ",young);
        locatespace(40,2,38,2);
        printf("OK ? ");
        while ( !kbhit() ) printxycycle(cycle, xcyclecnt, ycyclecnt);
        locatespace(40,1,38,3);
        ika=getch();
        if((ika!='y')&&(ika!='Y')&&(ika!=CR)) return(0);
        *YOUNG = young;
        return(1);
}


/***cursor.c***********

cursorsw()

*********************/
#include <stdio.h>
#include  "p9128.h"

void cursorsw( int x )
{
        if(x==0)
```

```
        printf("¥x1b[>5h");  /* CURSOR OFF */
    else
        printf("¥x1b[>5l");  /* CURSOR ON */
}


/***factin.c*******************

factin()

input factor to MPDAC
*************************/
#include <conio.h>
#include "p9128.h"

#define         OUTPORT             0X02d8  /* output port of MPDAC */

int factin(int fact)   /*  input MP factor  */
{
        int factl,facth;

        if (fact>4095) fact=4095;
        factl = ( fact& 0x00ff );                   /*データの下位８ビット*/
        facth = ( (unsigned)fact>> 8 );     /*データの上位８ビット*/
        outp(OUTPORT,factl);
        while(1) break;
        outp(OUTPORT+1,facth);
        return(0);
}


/*** gain.c*********************************************

  gain_ini() : initialization of programable gain amp. (gain=1)
  prog_gamp(): the voltage of COD gage, xvolt
          (1) if 4<=xvolt<=8, then no change.
          (2) if 0<=xvolt<4, then increase gain.
          (3) if 8<xvolt<=10, then decrease gain.
          (4) if xvolt<0 or 10<xvolt,
                      then out of range
  gain_out() : send changed gain signal to programable  gain amp.

        << initial gain  2  >>
 ***************************************************/
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include "p9128.h"

#define         STAT        0x03da  /* PGAMP start port */
#define         LSB                 0x03d8  /* PGAMP gain data output port */
#define         MSB                 0x03d9  /* PGAMP gain data output port */
#define         SS1                 0x01    /* PGAMP zero bit on */
#define         SS4                 0x04    /* PGAMP 2 bit on */
#define         BLACK       0x00
#define         RED         0x40
#define         NORMAL              0x01

int gain_ini()
```

```
{
   int status,rdata;

   rdata=inp( LSB );
   rdata=inp( MSB );
   while(1) {
      status=inp( STAT );
      if( (status & SS1) == 0 ) break;
   }
   outp( LSB,128 );
   rdata=inp( LSB );
   while(1) {
      status=inp( STAT );
      if( (status & SS1) == 0 ) break;
   }
   outp( LSB,0 );
   rdata=inp( LSB );

   if( rdata != 0 ) {
      printf("error gain amp¥n");   /* GAIN-AMP ERROR, quit the test */
         exit(0);
         }
         return(1);
}



void gain_out( int gdata )
{
   int status,rdata;

   while(1) {
      status=inp( STAT );
      if( (status & SS1) == 0 ) break;
   }
   outp( LSB, gdata );
   rdata=inp( LSB );

   while(1) {
      status=inp( STAT );
      if( (status & SS1) == 0 ) break;
   }
   outp( LSB, gdata-128 );

   while(1) {
      status=inp( STAT );
      if( ( status & SS4 ) != 0 ) break;
   }
   rdata=inp( LSB );
   if( rdata != gdata-128 ) {
      printf("error gain amp¥n");        /* GAIN-AMP ERROR, quit the test */
                  exit(0);
   }
   return;
}



void prog_gamp(double codmax, int *gain)    /* vmax: max voltage of cod */
{
```

```
    static int g[]= { 1, 2, 4, 8, 16 };             /* gain */
    static int d[]= { 128,129,130,133,134 };    /* gain-data */
    static int gn=1;                                        /* gain=2 */       /* gain number 0,1,2,3,4 */
    double      vv;
          int       flag=0;

    while(1) {
       vv = codmax;
       codmax = 0.;
       if( vv >=4. && vv <= 8. ) {    /* no change */
          flag=0;
          break;
       }
       else if( vv >=0. && vv < 4. ) {   /* increase gain */
          if( gn == 4 ) {
             *gain=16; flag=1;
             break;
          }
          else {
             gn++; flag=0;
             break;
          }
       }
       else if( vv > 8. && vv <= 10. ) {   /* decrease gain */
          if( gn == 0 ) {
             *gain=1; flag=2;
             break;
          }
          else {
             gn--; flag=0;
             break;
          }
       }
       else if( vv < 0. || vv > 10.) {
          flag=3;
          break;
       }
    }

    switch( flag )
          {
       case 1 :vstr_set(40,0,"CANNOT INCREACE GAIN",RED|NORMAL);
                                       break;
       case 2 :vstr_set(40,0,"CANNOT DECREACE GAIN",RED|NORMAL);
                                       break;
       case 3 :vstr_set(40,0,"OUT OF RANGE OF COD",RED|NORMAL);
                                       break;

       default : gain_out(d[gn]);    /* output changed gain */
                *gain=g[gn];
                break;

          }
    return;
}
```

```
/***keyin.c*************************************************************

double keyin(int x,int y)

ＣＲＴのＸ列Ｙ行から８桁の数値（小数点も１桁とする）を入力し、返値として、
(double)の数値を返すただし、入力可能な文字は０－９および　’．’である
**********************************************************************/
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include  "p9128.h"

#define ESC 0x1b
#define  BS 0x08
#define  CR 0x0d

double keyin(int x,int y)
{
        int i;
        char cbuf[20],c;

        locate(x,y);
        i=0;
        for(i=0;(c=(char)getch())!=CR ; )
        {
                if((c==BS)&&(i>0))
                {
                        i--;
                        cbuf[i]='¥0';
                        locate(x+i,y);
                        printf(" ");
                        locate(x+i,y);
                }
                else if((c>='.')&&(c<='9')&&(i<8))
                {
                        locate(x+i,y);
                        cbuf[i]= c;
                        printf("%c",cbuf[i]);
                        i++;
                }
        }
        cbuf[i] = '¥0';
        return( atof(cbuf) );
}


/***keyinint.c**********************************************************

int keyinint(int x,int y)

ＣＲＴのＸ列Ｙ行から３２７６７までの正の整数値を入力し、返値として、
（ｉｎｔ）の整数値を返す。３２７６８以上の場合には（－１）を返す。
ただし、入力可能な文字は０－９である。（符号は使えない）
**********************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include  "p9128.h"
```

```
#define ESC 0x1b
#define  BS 0x08
#define  CR 0x0d

int keyinint(int x,int y)
{
        int i;
        char cbuf[20],c;

        locate(x,y);
        i=0;
        for(i=0;(c=(char)getch())!=CR ; )
        {
                if((c==BS)&&(i>0))
                {
                        i--;
                        cbuf[i]='¥0';
                        locate(x+i,y);
                        printf(" ");
                        locate(x+i,y);
                }
                else if((c>='0')&&(c<='9')&&(i<5))
                {
                        locate(x+i,y);
                        cbuf[i]= c;
                        printf("%c",cbuf[i]);
                        i++;
                }
        }
        cbuf[i] = '¥0';
        if(atoi(cbuf)<0||atoi(cbuf)>32767)
                return(-1);
        else
                return( atoi(cbuf) );

}


 /***keyinlon.c*************************************************************************

int keyinlong(int x,int y)

ＣＲＴのＸ列Ｙ行から２１４７４８３６４７までの正の整数値を入力し、返値として、
（ｌｏｎｇ　ｉｎｔ）の整数値を返す。２１４７４８３６４８以上の場合には（－１）
を返すただし、入力可能な文字は０－９である。（符号は使えない）
*************************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include  "p9128.h"

#define ESC 0x1b
#define  BS 0x08
#define  CR 0x0d

long keyinlong(int x,int y)
{
        int i;
        char cbuf[20],c;
```

```
        locate(x,y);
        i=0;
        for(i=0;(c=(char)getch())!=CR ; )
        {
                if((c==BS)&&(i>0))
                {
                        i--;
                        cbuf[i]='¥0';
                        locate(x+i,y);
                        printf(" ");
                        locate(x+i,y);
                }
                else if((c>='0')&&(c<='9')&&(i<10))
                {
                        locate(x+i,y);
                        cbuf[i]= c;
                        printf("%c",cbuf[i]);
                        i++;
                }
        }
        cbuf[i] = '¥0';
        if(atol(cbuf)<0||atol(cbuf)>2147483647)
                return(-1);
        else
                return( atol(cbuf) );

}


/***keyinstr.c*************************************************************************

char *keyinstr(int x,int y)

カーソルをＣＲＴのＸ列Ｙ行に移動して１２文字以内の（ピリオドも１文字とする）
文字列を取り込み、返値として、(char *)の値を返す
*************************************************************************************/
#include <stdio.h>
#include <conio.h>
#include  "p9128.h"

#define ESC 0x1b
#define  BS 0x08
#define  CR 0x0d

char *keyinstr(int x,int y)
{
        int i;
        static char cbuf[20],c;

        locate(x,y);
        for(i=0;(c=(char)getch())!=CR ; )
        {
                if((c==BS)&&(i>0))
                {
                        i--;
                        cbuf[i]='¥0';
                        locate(x+i,y);
                        printf(" ");
                        locate(x+i,y);
                }
```

```
                else if (i<12)
                {
                        locate(x+i,y);
                        cbuf[i]= c;
                        printf("%c",cbuf[i]);
                        i++;
                }
        }
        cbuf[i] = '¥0';
        return( cbuf );
}


/***kvalue.c************************

kvalue()

test program of calculation of K-value
*********************************/
#include <stdio.h>
#include <math.h>
#include  "p9128.h"

#define  B0  0.886   /* constants of equation (2)       */
#define  B1  4.64    /* which determines stress intensity */
#define  B2  -13.32
#define  B3  14.72
#define  B4  -5.6


kvalue(stress,crack,kval,width,thick)
        double stress,crack,*kval;    /*  tangent: [mm/kg]  */
        double width,thick;
{
        double abyw, xx;

        /*  computation of stress intensity, equation (2) */
        abyw = crack/width;
        xx = stress*(2.+abyw)/thick/sqrt(width)/pow((1.-abyw),1.5);
        *kval = xx*(B0+B1*abyw+B2*pow(abyw,2.)+B3*pow(abyw,3.)+B4*pow(abyw,4.));
        return(0);
}


/***locat.c**********

locate()
text_clr()

******************/
#define ESC 0x1b
#define  BS 0x08
#define  CR 0x0d
#include <stdio.h>
#include  "p9128.h"

int locate(int x,int y)
{
        if ( x<0 || x>80 || y<0 || y>25 ) return(0);
        putchar(ESC);
```

```
        putchar('[');
        putchar(y/10+'0');
        putchar(y%10+'0');
        putchar(';');
        putchar(x/10+'0');
        putchar(x%10+'0');
        putchar('H');
        return(0);
}

int text_clr(void)
{
        putchar(ESC);
        printf("[2J");
        return(0);
}


/***locatesp.c*****************

locatespace( x, y, space, line )

*************************/
#include <stdio.h>
#include <conio.h>
#include  "p9128.h"


int locatespace(int a, int b, int c, int d)
{
        int i,j;
        if ( (a+c)>80 ) c=81-a;
        if ( (b+d)>25 ) d=26-b;
        for(i=0; i<d; i++)
        {
                locate(a,b+i);
                for( j=0; j<c; j++)             printf(" ");
        }
        locate(a,b);
        return(0);
}


/***lsqm.c*********************************************

lsqm()

最小自乗法によりP(load), V(COD)の（tangent）を求める
*****************************************************/
#include <math.h>
#include  "p9128.h"

lsqm(samplep,samplev,point,tangent,loadfact,codfact,adfact)
        int samplep[],samplev[],point;
        double *tangent,loadfact,codfact;  /* loadfact=1000.  codfact=0.125 */
        double  adfact;                    /* 4095 = 10 [v]  */
{
        int i;
        double sump,sumv,sumpv,sump2,p,v;
```

```
        sump = 0.;
        sumv = 0.;
        sumpv = 0.;
        sump2 = 0.;

        for ( i=0; i<point; i++)
        {
                p = (double)samplep[i]*loadfact/adfact;  /*  p [kg]  */
                v = (double)samplev[i]*codfact/adfact;   /*  v [mm]  */
                sump += p;
                sumv += v;
                sumpv += (p*v) ;
                sump2 += (p*p) ;
        }
        *tangent = ((double)point*sumpv-sump*sumv)/
                                        ((double)point*sump2-sump*sump);  /* tangent : [kg/mm] */
        return(0);
}


/***printpd.c*********************************

printpd()

プリンター、ディスクへの出力プログラム
write to printer & disk "specimen ID.dat"
*****************************************/
#include <time.h>
#include <stdio.h>
#include  "p9128.h"

#define         RED             0x40
#define         NORMAL          0x01

int printpd(cycle,crack,kval,stress,tangent,crackfact,cnt,fname2,preycycle,ycycle)
long cycle,cnt,preycycle,ycycle;
double crack,kval,stress,tangent,crackfact;
char fname2[];
{
        FILE *fp;
        char time[10];

        _strtime(time);

/*-------- disket ----------*/
        fp=fopen(fname2,"a+");
        if( fp == NULL )
        {
                vstr_set(40,0,"CANNOT OPEN FILE !!!",RED|NORMAL);
                tstop(500);
        }
        else {
                fprintf(fp,"%7ld  %5.2lf  %6.2lf  %6.1lf %3ld %12.5e  %7.5lf  %s¥n",
                        cycle, crack, kval, stress, cnt, tangent, crackfact, time);
        }
        if( fclose(fp)== EOF )
        {
                vstr_set(40,0,"CANNOT CLOSE FILE !!!",RED|NORMAL);
                tstop(500);
        }
```

```
/*--------- printer ---------*/
        fp=fopen("PRN","a+");
        if( fp == NULL )
        {
                vstr_set(40,0,"CANNOT OPEN PRINTER !!!",RED!NORMAL);
                tstop(500);
        }
        else {
                fprintf(fp,
                        "CYCLES=%7ld A=%5.2lf dK=%6.2lf dP=%5.0lf V/P=%10.3e  %s %3ld %4ld %4ld¥n",
cycle, crack, kval, stress, tangent, time, cnt, preycycle, ycycle);
        }
        if( fclose(fp)== EOF )
        {
                vstr_set(40,0,"CANNOT CLOSE PRINTER !!!",RED!NORMAL);
                tstop(500);
        }
        return(0);
}


/***pvalue.c***************

pvalue()

test program of kval & crack
*************************/
#include <stdio.h>
#include <math.h>
#include "p9128.h"

#define C0 1.00104 /* constants of equation (1)     */
#define C1 -4.72   /* which determines crack length  */
#define C2 19.4
#define C3 -249.
#define C4 1320.
#define C5 -2400.
#define B0 0.886   /* constants of equation (2)       */
#define B1 4.64    /* which determines stress intensity */
#define B2 -13.32
#define B3 14.72
#define B4 -5.6


pvalue(stress,crack,kval,width,thick)
        double *stress,crack,kval;
        double width,thick;
{
        double abyw, xx;

        abyw = crack/width;
        xx = kval*thick*sqrt(width)*pow((1.-abyw),1.5)/(2.+abyw);
        *stress = xx/(B0+B1*abyw+B2*pow(abyw,2.)+B3*pow(abyw,3.)+B4*pow(abyw,4.));
        return(0);
}
```

```
/***setsin.c*****************************************************

setsin()

make sin curve data used by MPDAC
output : HAVER SINE data  (100 point)  lower 0   upper 32767   mean 16383
DAC setting  CMP BIP  -10[v] to +10[v]
************************************************************/
#include <stdio.h>
#include <math.h>
#include  "p9128.h"

#define  PAI                3.141592653
#define  NMAX  100          /*  number of sine data  */
#define  mean     0.0
#define  factor  16383.5    /*  32767: 10v    0<s<2   then   0kg-10000kg */

int setsine(sd)
int sd[];
{
        double  s;
        int i;

        for( i=0; i<NMAX; i++ ) {
                s = -cos( 2. * PAI * (double)i / (double)NMAX ) + 1. ;
                sd[i] = (int)( s * factor + mean + 0.5 );
        }
        return(0);
}


/***space.c*****************************************************

space(n)

カーソル位置からｎ文字分のスペースをあけもとのカーソル位置に戻る)
 ************************************************************/
#include <stdio.h>
#include  "p9128.h"

int space(int n)
{
        int i;
        for( i=0; i<n; i++)          printf(" ");
        for( i=0; i<n; i++)          printf("¥b");
        return(0);
}


 /***tstop.c****************
tstop()

**************************/
#include <stdio.h>
#include  "p9128.h"
extern  int stoptimer2(int fact);

int tstop(int fact)
{
```

```
        stoptimer2(fact);
        text_clr();
        locate(25,12);
        printf("PROGRAM TERMINATED");
        locate(25,14);
        printf(" TURN DOWN 'SPAN'¥n¥n¥n");
        cursorsw(1);
        exit(0);
}


/***vstr.c**********************************

vstr()

    テキストＶＲＡＭ直接出力（文字列）
  x, y      座標
  addres    文字列の先頭アドレス
  attr      文字アトリビュートコード
*****************************************/
#include <math.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <dos.h>
#include  "p9128.h"

/* カラーコード */
#define         BLACK           0x00
#define         BLUE            0x20
#define         RED             0x40
#define         PURPLE                  0x60
#define         GREEN           0x80
#define         SKYBLUE                 0xa0
#define         YELLOW                  0xc0
#define         WHITE           0xe0
/* 表示モードコード */
#define         SECRET                  0x00
#define         NORMAL                  0x01
#define         BLINK           0x03
#define         REVERSE                 0x05
#define         UNDERLINE       0x09
#define         VERTICALLINE0x11

extern char *keyinstr(int,int);

void vstr_set(int x, int y, unsigned char *addres, int attr)
{
        int position=(x<<1)+160*y;
        do{
                *(int far *)( 0xA0000000+position )= *addres;
                *(int far *)( 0xA2000000+position )=  attr;
                position+=2;
        }while( *++addres!='¥0' );
}
```

```
/***fuzmain.c*************************************************************

<<<  制御プログラムのファジィ推論部  >>>

fuzmain() : cntが１２になったときからcrack,cycleよりycycle,xcycleを求める
datain()  : 過去１２回のcrack,cycleを保存する
calxcycle() : xcycleを計算する
mkfs()    : メンバーシップ関数を作成する
sort1()   : 小さい順にソートする
calgrade() : 前件部の各グレードを計算する
fuzzyrule() : ファジィルールにより後件部のそれぞれのαカットレベルを決定する
center()  : 重心を計算してycycleを求める
printfuz()  : メンバーシップ関数等を画面に出力

*************************************************************************/
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <conio.h>
#include "fuz.h"

void fuzmain (crack,cycle,xcycle,ycycle,cnt,preycycle)
        double crack;
        long cycle,*xcycle,*ycycle,cnt,*preycycle;
{
        static long boxn[14];
        static double boxa[14],cent;
        static double np2,zop2,pp2,jjj;
        static double decp2,incp2,nx,zox,px,decm,incm,nocm,precpr,preccpr;
        static double ans,azo,aps,apb;

                datain(crack,cycle,boxa,boxn);

                if( cnt==12 )
                calxcycle( boxa, boxn, *(&xcycle) );

                if( cnt>=12 )
                {
                mkfs( boxa,boxn,&np2,&zop2,&pp2,&decp2,&incp2 ,&precpr,&preccpr);
                calgrade (np2,zop2,pp2,decp2,incp2,precpr,preccpr,
                        &nx,&zox,&px,&decm,&nocm,&incm,&jjj);
                fuzzyrule(nx,zox,px,decm,nocm,incm,&ans,&azo,&aps,&apb);
                center(ans,azo,aps,apb,zop2,jjj,*(&ycycle),crack,*xcycle,precpr,*(&preycycle),
&cent);
                printfuz(np2,zop2,pp2,decp2,incp2,jjj,precpr,cent,preccpr,*preycycle);

                }
                return;
}


void datain( crack,cycle,boxa,boxn )
double crack;
double boxa[]; /* 過去１２回のcrackのデータ */
long cycle;
long boxn[]; /* 過去１２回のcycleのデータ */
{
        int i;
        for(i=0; i<12; i++){
```

```
                boxa[i]=boxa[i+1];
                boxn[i]=boxn[i+1];
        }
        boxa[12]=crack;
        boxn[12]=cycle;

        return;
}

void calxcycle( boxa,boxn,xcycle)
long boxn[],*xcycle;
double boxa[];
{
        int i;
        long dn[13]; /* ΔN */
        double da[13]; /* Δa */
        double dnda[13]; /* da/dN */
        double sumdnda;

        sumdnda=0.;
        dn[0]=0l;
        da[0]=0.;
        dnda[0]=0.;

        for(i=0;i<12;i++){
                dn[i+1]=boxn[i+1]-boxn[i];
                da[i+1]=boxa[i+1]-boxa[i];
                if(da[i+1]<=0.0) da[i+1]=0.05;
        }
        for(i=0;i<12;i++){
                dnda[i+1]=(double)dn[i+1]/da[i+1];
                sumdnda=sumdnda+dnda[i+1];
        }
        /***0.4mm伸びる平均のcycle数に0.6を乗じる ***/
        *xcycle = (long)(sumdnda/12.*0.4*0.6);
        return;
}

void mkfs( boxa,boxn,np2,zop2,pp2,decp2,incp2,precpr,preccpr)
        long boxn[];
        double boxa[],*np2,*zop2,*pp2,*decp2,*incp2,*precpr,*preccpr;
{
        static double cpr[14]; /* Crack Propagtion Rate */
        static double ccpr[14]; /* Change of Crack Propagation Rate */
        static double scpr[12]; /* ソートのための cpr */
        static double sccpr[11]; /* ソートのための ccpr */
        double scpr1,scpr2,scpr3,sccpr1,sccpr2,sccpr3;
        int i;

        scpr[0] = sccpr[0] = sccpr[1] = *precpr = *preccpr = 0.0;

        for(i=0;i<12;i++)
        {
                if( (boxa[i+1]-boxa[i])>0.0 )
                                cpr[i+1]=(boxa[i+1]-boxa[i])/(double)(boxn[i+1]-boxn[i]);
                else    cpr[i+1]=0.05/(double)(boxn[i+1]-boxn[i]);
        }
        for(i=1;i<12;i++)
                ccpr[i+1]=(cpr[i+1]-cpr[i])/(double)(boxn[i+1]-boxn[i]);
```

```
        *precpr=cpr[12]; /* 前回のcpr */
        *preccpr=ccpr[12]; /* 前回のccpr */

        for(i=0;i<12;i++)
                scpr[i]=cpr[i+1];
        for(i=0;i<11;i++)
                sccpr[i]=ccpr[i+2];
        sort1(scpr,12);
        sort1(sccpr,11);

        scpr1=(scpr[0]+scpr[1]+scpr[2]+scpr[3])/4.;
        scpr2=(scpr[4]+scpr[5]+scpr[6]+scpr[7])/4.;
        scpr3=(scpr[8]+scpr[9]+scpr[10]+scpr[11])/4.;
        sccpr1=(sccpr[0]+sccpr[1]+sccpr[2]+sccpr[3])/4.;
        sccpr2=(sccpr[4]+sccpr[5]+sccpr[6])/3.;
        sccpr3=(sccpr[7]+sccpr[8]+sccpr[9]+sccpr[10])/4.;

        if(scpr1==scpr2)  scpr1=scpr1*0.9;
        if(scpr2==scpr3)  scpr3=scpr3*1.1;
        if(sccpr1>=0.00) sccpr1=-fabs(sccpr2)*0.1;
        if(sccpr3<=0.00) sccpr3= fabs(sccpr2)*0.1;

        *np2=scpr1; /* 前件部cprにおけるNegativeの頂点の値 */
        *zop2=scpr2; /* 前件部cprにおけるZeroの頂点の値 */
        *pp2=scpr3; /* 前件部cprにおけるPositiveの頂点の値 */
        *decp2=sccpr1; /* 前件部ccprにおけるDecreaseの頂点の値 */
        *incp2=sccpr3; /* 前件部ccprにおけるIncreaseの頂点の値 */

        return;
}



sort1(a,n)

  double a[] ;
  int n ;
{
  int i,j,k,l,m ;
  double w ;

  l = n / 2 ;
  for (k = l; k > 0 ; k /= 2)
  {
    for (i = k; i < n; i++)
    {
      j = i - k ;
      while(1)
      {
        m = j + k ;
        if (j < 0 || a[j] <= a[m]) break ;
        w = a[j] ;
        a[j] = a[m] ;
        a[m] = w ;
        j -= k ;
      }
    }
  }
  return(0) ;
}
```

```
void calgrade (np2,zop2,pp2,decp2,incp2,precpr,preccpr,nx,zox,px,decm,nocm,incm,jjj)
        double np2,zop2,pp2,decp2,incp2,precpr,preccpr;
        double *nx; /* Negative のグレード */
        double *zox; /* Zeroのグレード */
        double *px; /* Positiveのグレード */
        double *decm; /* Decreaseのグレード */
        double *nocm; /* No Cangeのグレード */
        double *incm; /* Increaseのグレード */
        double *jjj; /* ZeroとNegativeとの頂点の差 */
{
        if( precpr<np2 ) *nx=1.0;
        else *nx=(precpr-zop2)/(np2-zop2);

        if( precpr<zop2 ) *zox=(precpr-np2)/(zop2-np2);
        else *zox=(precpr-pp2)/(zop2-pp2);

        if(precpr>pp2 )    *px=1.0;
        else *px=(precpr-zop2)/(pp2-zop2);

        if( preccpr<decp2 ) *decm=1.0;
        else *decm=preccpr/decp2;

        if( preccpr>0. ) *nocm=(-preccpr/incp2)+1.0;
        else *nocm=-preccpr/decp2+1.0;

        if( preccpr>incp2 ) *incm=1.0;
        else *incm=preccpr/incp2 ;

        *jjj=(zop2-np2);

        if ( *nx<0.0 ) *nx=0.0;
        if ( *zox<0.0 ) *zox=0.0;
        if ( *px<0.0 ) *px=0.0;
        if (*decm<0.0 ) *decm=0.0;
        if (*nocm<0.0 ) *nocm=0.0;
        if (*incm<0.0 ) *incm=0.0;

        return;
}

void fuzzyrule (nx,zox,px,decm,nocm,incm,ans,azo,aps,apb)
        double *ans; /* NSのαカットレベル */
        double *azo; /* ZOのαカットレベル */
        double *aps; /* PSのαカットレベル */
        double *apb; /* PBのαカットレベル */
        double nx,zox,px,decm,nocm,incm;
{
        double aps1,aps2,azo0,azo1,azo2,azo3,azo4,ans1,ans2;

        aps1=aps2=0.0;
        azo0=azo1=azo2=azo3=azo4=0.0;
        ans1=ans2=0.0;
        *ans=*azo=*aps=*apb=0.0;

        if( nx>0.0 ){
                if( decm>0.0 ){
                        if( nx<decm ) *apb=nx;
                        else *apb=decm;
                }
```

```
        if( nocm>0.0 ){
                if( nx<nocm ) aps1=nx;
                else aps1=nocm;
        }
        if( incm>0.0 ){
                if( nx<incm ) aps2=nx;
                else aps2=incm;
        }
}

if( zox>0.0 ){
        if( decm>0.0 ){
                if( zox<decm ) azo1=zox;
                else azo1=decm;
        }
        if( nocm>0.0 ){
                if( zox<nocm ) azo2=zox;
                else azo2=nocm;
        }
        if( incm>0.0 ){
                if( zox<incm ) azo3=zox;
                else azo3=incm;
        }
}

if( px>0.0 ){
        if( decm>0.0 ){
                if( px<decm ) azo4=px;
                else azo4=decm;
        }
        if( nocm>0.0 ){
                if( px<nocm ) ans1=px;
                else ans1=nocm;
        }
        if( incm>0.0 ){
                if( px<incm ) ans2=px;
                else ans2=incm;
        }
}

if( aps1>aps2 )
        *aps=aps1;
else
        *aps=aps2;

if( ans1>ans2 )
        *ans=ans1;
else
        *ans=ans2;

if( azo1>azo2 ){
        if( azo1>azo3 ) azo0=azo1;
        else azo0=azo3;
}
else{
        if( azo2>azo3 ) azo0=azo2;
        else azo0=azo3;
}
if( azo0>azo4 ) *azo=azo0;
else *azo=azo4;
```

```
        return;
}

void center (ans,azo,aps,apb,zop2,jjj,ycycle,crack,xcycle,precpr,preycycle,cent)
        double ans,azo,aps,apb,zop2,jjj,crack,precpr;
        double *cent; /* 重心 */
        long *ycycle,xcycle,*preycycle;
{
        double dela; /* ΔKが０．４%ずれるまでのき裂長さ */

        if((azo>0.0 && aps>0.0 && apb>0.0) || (ans>0.0 && azo>0.0 && aps>0.0)){
                if(azo>0.0 && aps>0.0 && apb>0.0){
                        if(azo>=0.5)
                                *cent=(2.0*zop2*jjj*azo*(1.0-azo/2.0) +aps/2.0*jjj*(2.0*zop2+jjj*(3.0-aps))
                                 +apb*jjj*(2.0*zop2+(5.0-apb)*jjj)/2.0) /(2.0*jjj*azo*(1.0-azo/2.0)+(aps+apb)*jjj);
                        else if(aps>=0.5)
                                *cent=(2.0*jjj*(zop2+jjj)*aps*(1.0-aps/2.0)+azo/2.0*jjj*(2.0*zop2+jjj*(azo-1.0))
                                +apb*jjj*(2.0*zop2+(5.0-apb)*jjj)/2.0) /(2.0*jjj*aps*(1.0-aps/2.0)+(azo+apb)*jjj);
                        else
                                *cent=(aps*jjj*((aps+1.0)*jjj/2.0+zop2)+(2.0-apb)*apb*jjj*(zop2+2.0*jjj)
                                 +azo/2.0*jjj*(2.0*zop2+jjj*(azo-1.0))) /((2.0-apb)*apb*jjj+(azo+aps)*jjj);
                }
                if(ans>0.0 && azo>0.0 && aps>0.0){
                        if(ans>=0.5)
                                if(aps>=azo)
                                        *cent=((2.0-ans)*ans*jjj*(zop2-jjj) +(2.0-aps)*aps*jjj*(zop2+jjj)
                                        +azo*azo*jjj*zop2) /((2.0-ans)*ans*jjj+(2.0-aps)*aps*jjj +azo*azo*jjj);
                                else
                                        *cent=((2.0-ans)*ans*jjj*(zop2-jjj)+azo/2.0*jjj*(2.0*zop2+jjj*(1.0-azo))
                                         +aps/2.0*jjj*(2.0*zop2+jjj*(3.0-aps))) /(jjj*ans*(2.0-ans)+(azo+aps)*jjj);
                        else if(azo>=0.5)
                                *cent=(ans/2.0*jjj*(2.0*zop2-jjj*(3.0-ans))+2.0*zop2*jjj*azo*(1.0-azo/2.0)
                                +aps/2.0*jjj*(2.0*zop2+jjj*(3.0-aps)))/(2.0*jjj*azo*(1.0-azo/2.0)+(ans+aps)*jjj);
                        else if(aps>=0.5)
                                if(ans>=azo)
                                        *cent=((2.0-ans)*ans*jjj*(zop2-jjj)+(2.0-aps)*aps*jjj*(zop2+jjj)
                                        +azo*azo*jjj*zop2) /((2.0-ans)*ans*jjj+(2.0-aps)*aps*jjj+azo*azo*jjj);
                                else
                                        *cent=((2.0-aps)*aps*jjj*(zop2+jjj) +ans/2.0*jjj*(2.0*zop2-jjj*(3.0-ans))
                                        +azo/2.0*jjj*(2.0*zop2+jjj*(azo-1.0)))/((2.0-aps)*aps*jjj+(ans+azo)*jjj);
                }
        }else{
                if(ans>0.0 && azo>0.0 ){
                        if(ans<=azo)
                                *cent=(ans/2.0*jjj*(2.0*zop2-jjj*(3.0-ans)) +2.0*zop2*jjj*azo*(1.0-azo/2.0))
                                 /(ans*jjj+2.0*jjj*azo*(1.0-azo/2.0));
                        else
                                *cent=(2.0*jjj*(zop2-jjj)*ans*(1.0-ans/2.0) +azo/2.0*jjj*(2.0*zop2+jjj*(1.0-azo)))
                                 /(2.0*jjj*ans*(1.0-ans/2.0)+azo*jjj);
                }else if( azo>0.0 && aps>0.0 ){
                        if( azo<=aps )
                                *cent=(2.0*jjj*(zop2+jjj)*aps*(1.0-aps/2.0)+azo /2.0*jjj*(2.0*zop2+jjj*(azo-1.0)))
                                 /(2.0*jjj*aps*(1.0-aps/2.0)+azo*jjj);
                        else
                                *cent=(2.0*zop2*jjj*azo*(1.0-azo/2.0)+aps/2.0*jjj*(2.0*zop2+jjj*( 3.0-aps)))
                                /(2.0*jjj*azo*(1.0-azo/2.0)+aps*jjj);
                }else if( azo>0.0 && apb>0.0 ){
                        *cent=(2.0*jjj*(zop2+2.0*jjj)*apb*(1.0-apb/2.0)+2.0 *zop2*jjj*azo*(1.0-azo/2.0))
                        /(2.0*jjj*apb *(1.0-apb/2.0)+2.0*jjj*azo*(1.0-azo/2.0));
```

```
                }else if( ans>0.0 && aps>0.0 ){
                        *cent=((2.0-ans)*ans*jjj*(zop2-jjj)+(2.0-aps)*aps*jjj*(zop2+jjj))/((2.0-ans)*ans*jjj
                        +(2.0-aps) *aps*jjj);
                }else if( aps>0.0 && apb>0.0 ){
                        if( aps<=apb )
                                *cent=(aps*jjj*((aps+1.0)*jjj/2.0+zop2)+(2.0-apb)*apb*jjj*(zop2+2.0*jjj))
                                /(aps*jjj+(2.0-apb)*apb*jjj);
                        else
                                *cent=((2.0-aps)*aps*jjj*(zop2+jjj)+apb*jjj *(2.0*zop2+(5.0-apb)*jjj)/2.0)
                                /((2.0-aps)*aps*jjj+apb*jjj);
                }else{
                        if(ans>0.0)
                                *cent=zop2-jjj;
                        if(azo>0.0)
                                *cent=zop2;
                        if(aps>0.0)
                                *cent=zop2+jjj;
                        if(apb>0.0)
                                *cent=zop2+2*jjj;
                }
        }

                if(*cent<=0.0) *cent=zop2;

                                dela = 0.13075
                                        +2.9724*pow(10.0,-4.0)*crack
                                        +5.7223*pow(10.0,-4.0)*pow(crack,2.0)
                                        -2.1155*pow(10.0,-5.0)*pow(crack,3.0)
                                        +2.6348*pow(10.0,-7.0)*pow(crack,4.0)
                                        -1.1273*pow(10.0,-9.0)*pow(crack,5.0);


                *preycycle=(long)(dela/precpr);
                *ycycle =(long)((double)*preycycle*(zop2 / *cent));

                if (*ycycle<xcycle/4) *ycycle=(long)(xcycle/4);

                return;
}

void printfuz(np2,zop2,pp2,decp2,incp2,jjj,precpr,cent,preccpr,preycycle)
        double np2,zop2,pp2,decp2,incp2,jjj,precpr,cent,preccpr;
        long preycycle;
{
        locatespace(41,1,40,9);
        locate(41,1); printf("          < FUZZY REASONING >          ");
        locate(41,2); printf("cpr : %10.3e %10.3e %10.3e",np2,zop2,pp2);
        locate(41,3); printf("ccpr:%10.3e 0.000e-000 %10.3e",decp2,incp2);
        locate(41,4); printf("   NS       ZO       PS       PB    ");
        locate(41,5); printf("%9.2e %9.2e %9.2e %9.2e"
                        ,zop2-jjj,zop2,zop2+jjj,zop2+2*jjj);
        locate(41,6); printf("  %5.3lf    %5.3lf    %5.3lf    %5.3lf  "
                        ,(zop2-jjj)/zop2,zop2/zop2,(zop2+jjj)/zop2,(zop2+2*jjj)/zop2);
        locate(41,7); printf("  %5.3lf    %5.3lf    %5.3lf    %5.3lf  "
                        ,zop2/(zop2-jjj),zop2/zop2,zop2/(zop2+jjj),zop2/(zop2+2*jjj));
        locate(41,8); printf("precpr : %11.3e  FACTOR  :%1.3lf",precpr,zop2/cent);
        locate(41,9); printf("preccpr: %11.3e  PreYcycle:%ld",preccpr,preycycle);

        return;
}
```