対話型数値シミュレーションシステム：ビジュアル DEQSOL†

金野千里†† 梅谷征雄†† 太田忠††† 深田肇†††† 山賀晋††† 池田美以子†††

スーパコンピュータの普及に象徴されるように数値シミュレーションへの需要は近年特に高まっている．筆者らはかねてよりそのソフトウェア環境として，連続型のシミュレーション問題に対して数式レベルの記述から計算性能の良いシミュレーションプログラムを自動生成する高水準言語 DEQSOL (Differential EQuation SOlver Language) の研究開発を行ってきた．さらに近年普及の目覚ましいエンジニアリングワークステーションの対話環境と分散環境を活かして，シミュレーション工程を一貫して支援するビジュアルDEQSOLシステムへと発展させた．本システムは，シミュレーションに関連する情報群を物理的情報，数学的情報，数値的情報に階層分けし，それらをアイコンやボタンで代表させ，その押下によって現れるサブウィンドウやテンプレートやメニューの操作によって，プログラミング，実行，結果検証に至るすべての工程を可能としている．本システムの特徴は，領域形状や支配方程式の画面への直接入力方式，条件式やアルゴリズム展開のエキスパート的なガイダンス機能，標準的なウィンドウシステム上での実現による見え方や操作の統一性などである．本システムによれば，FORTRAN 等の従来言語による工程を 1/10～1/30 に短縮できるだけでなく，ユーザの数値シミュレーション実行上の前提知識が大幅に緩和される．

1. はじめに

スーパコンピュータの出現と普及に象徴されるように，数値シミュレーションの需要は近年特に高まっている．現在では，数値シミュレーションなくしては，先端技術の研究開発はありえないと言っても過言ではないまでに至っている．

計算能力を供給する半導体 LSI を始めとするハードウェアの目覚ましい進歩に比べ，その利用技術であるソフトウェアの面は立ち遅れが目立っていた．そこで著者らは，一般の科学技術者が自在に記述でき，かつ FORTRAN に代わる生産性の高いプログラミング言語として，偏微分方程式で記述される連続型のシミュレーション問題を対象に DEQSOL (Differential EQuation SOlver Language)を現究開発してきた[6)~9)]．本言語は，微分演算子を用いた数式レベルの記述と解析領域の記述から，実行性能（ベクトル化率）の高い FORTRAN プログラムを自動生成するものである．これにより，(スーパ) コンピュータの専門家でなくても容易に数値シミュレーションを実行することが可能である．DEQSOL は FORTRAN の 10 倍以上の記述効率を有し，自動生成されるシミュレーションコードは 90% 以上の高いベクトル化率を達成している．

DEQSOL と同様の目的で研究開発されているシステムとしては，欧米を中心とした大学，公的研究機関による ELLPACK[1)-3)], FIDISOL[4)], ALPAL[5)] などが挙げられる．システムの実現方式は，コード・ジェネレータを備えてプログラム生成を行うものと，多数のライブラリを備えてそこへのドライブを図るものに大別される．DEQSOL, ALPAL などは前者に，ELLPACK は後者に属する．DEQSOL はコード生成方式を活かし，マシンに最適なプログラムの生成に大きな特徴を有している．また，数値解析手法として差分法（バウンダリ・フィット法を含む[11)]）と有限要素法の両解析手法を備え，連立偏微分方程式の一括解法機能[10)]，離散化の精度制御や風上化機能[13)]を加え，ポテンシャル問題，移流拡散問題，流体問題，およびそれらの連成した問題など幅広い分野に適用可能である．

一方，ハードウェア面ではこの数年エンジニアリングワークステーションの普及と高性能化が目覚ましく，計算機との対話環境やグラフィックス環境および分散処理環境の充実に加え，個人の占有できる計算機資源も大幅に増大している．

これらの背景を踏まえ，言語システムである DEQSOL をさらに一歩進め，ウィンドウインタフェース

† Interactive Numerical Simulation System: Visual DEQSOL by CHISATO KONNO, YUKIO UMETANI (Central Research Laboratory, Hitachi Ltd.), TADASHI OOTA (Hitachi VLSI Engineering Ltd.), HAJIME FUKADA (Hitachi Software Engineering Ltd.), SUSUMU YAMAGA and MIIKO IKEDA (Hitachi VLSI Engineering Ltd.).

†† (株)日立製作所中央研究所第7部
††† 日立超 LSI エンジニアリング(株)
†††† 日立ソフトウェアエンジニアリング(株)

を備えシミュレーションを一貫して支援するビジュアル DEQSOL システムを提案し開発したので，それについて述べる．本論文では，まず第2章にて言語 DEQSOL の実問題への適用を通した課題を挙げ，第3章でそれらを解決するビジュアル DEQSOL のマンマシンインタフェースを示す．第4章ではその実現方式であるシステム構成を，第5章では本システムの1つの特徴的機能であるアルゴリズム展開などのガイダンス機能を，第6章では本システムの評価について述べる．

2. 言語システムの課題

FORTRAN などの従来言語の 10 倍以上の記述効率を有した DEQSOL においては，プログラミング工数それ自体は大幅に削減される．**図1**に DEQSOL によるプログラムの記述例（の一部）を示す．239 行の DEQSOL プログラムから 2,476 行の FORTRAN のシミュレーションプログラムが生成される[12]．

しかしながら，シミュレーションの全体工程およびシステムの使い勝手から省みた時，プログラミング工数以外に全体工数を長大化する因子のあるケース，形状やメッシュ疎密制御などの言語のみでは記述が困難なケース，言語のレベルの一層の高度化によってモデル記述が容易になるケース，あるいは初心者が言語を十分に使い熟せないケース等々が挙げられる．

本章ではこれらの工程分析[14),15)]などを踏まえ，言語 DEQSOL の課題と問題点を整理する．これは DEQSOL に特定したものではなく，言語によるシミュレーションシステムに共通している．

（1） 言語システムに起因する課題

（a） 文法習得の必要性

DEQSOL 自体の構文種別は解析手法に応じて約 32〜35 種で，問題よりの概念と数式レベルの記述を可能とするよう設計されているが，一般ユーザにとってみると文法の習得の要否自体が高い障壁となっている．

（b） 言語レベルへの問題の分解

言語の文法レパートリとして用意された要素に原問題を分解して記述する必要があり，ユーザによっては困難が伴う．特に DEQSOL においては，物理現象を表す偏微分方程式のアルゴリズムへの展開などが主要な困難さとなる．

（c） 表現上の限界

情報の中にはテキストによって表現しにくい，もしくはできないものが存在する．問題の定義されている領域の形状やそれと偏微分方程式とを関連付ける境界条件などがそれに該当する．

（d） 構文相互の関連の保証

上位で定義された情報を用いて下位の手続き部分が記述されるが，その無矛盾をユーザが保証する必要があり，容易に誤りが混入する．シミュレーションにおいてその典型的なのは，偏微分方程式と境界条件式の適合性などが挙げられる．

```
PROG EB;                                                    プログラム名
METHOD FEM;                                                 解析手法選択
DOMAIN X=[0:7.5], Y=[0;7.5];                                空間範囲指定
                                                            領域形状定義
POINT P01=(5.8,5.6), P02=(5.8,2.6), P03=(5.6,2.2),            点定義
      P04=(5.2,2.0), P05=(5.0,2.0), P06=(4.88,1.96),
      .... ;                                                  線定義
REGION L01=LN(P01,P02), L02=ARC(P02,P03,P04),
       L03=SPL(P04,P05,P06,P07,P08,P09,P10),
       L04=LN(P10,P11),
       .... ;                                                 面定義
       Q01=QUAD(L01,L29,L27,L28), Q02=QUAD(L02,L30,L26,L29),
       Q03=QUAD(L03,L31,L41,L30),
       .... ;
FREGION Q01+Q02+Q03+.... ;
BOUND L01+L02+L03+.... ;                                    境界宣言
                                                            メッシュ定義
MESH L01=D(5), L02=D(5), L03=D(5),                            辺分割指定
     .... ,
     Q01=AS(P01,P02), Q02=AS(P02,P04),                        面分割指定
     .... ;
VAR PHI;                                                    変数宣言
CONST EPS=0.01, VX=0.0,                                     定数定義
      VX=10.0 AT Q04+Q11+Q14,
      VY= 0.0,
      .... ;
CTENS CT=(VXX,VXY,VYX,VYY);                                 定数テンソル定義
CVEC CV=(VX,VY);                                            定数ベクトル定義
BCOND PHI=50000.0 AT L15+L16+L17+L18+L19+L20+               境界条件定義
                     L21+L22+L23+L24+L25+L26+L27,             一種条件
      PHI=0 AT L01+L02+L03+L04+L05+L06+L07+
               L08+L09+L10+L11+L12+L13,
      NGRAD(PHI)=0 AT L14;                                    二種条件
SCHEME;                                                     アルゴリズム定義
  SOLVE PHI OF                                                偏微分方程式
    DIV(CT..GRAD(PHI))+(1/(X+EPS))*CV..GRAD(PHI)=0              の求解
  BY 'PCG';
  PRINT PHI;                                                結果数値出力指示
  SAVE PHI;                                                 結果作画指示
END SCHEME;
END;
```

図1 DEQSOL プログラムの記述例
Fig. 1 An example of PDEQSOL description.

（e） 誤り検出の難しさ

バッチ型の言語には共通であるが，エラー検出が翻訳時にのみ行われるので，エラーの主要因や主従関係を判定しにくい．

（2） システム形態に起因する課題

（a） 多種のジョブの連動

所望のシミュレーションを完了するには，プログラミング（データ入力を含む），コンパイル，他プログラムとのリンケージ，計算実行，結果の作画，検証等の異なったジョブもしくは作業を制御しなくてはならず，計算機環境への習熟が必須となる．

（b） 情報の多元管理

一般にプログラム，データ群（入力データ，解析結果など）は個別に管理されており，その対応はユーザ責任で管理しなくてはならない．特に数値シミュレーションでは同一問題に対して，データやメッシュや条件を変更して何種類もの計算を行うことが多く，プログラムとデータ群の管理は多くのユーザを煩わせている．

3. ビジュアル DEQSOL のマンマシンインタフェース

本章では，前章で挙げた課題を踏まえ，シミュレーションを統合的に支援するビジュアル DEQSOL システムのマンマシンインタフェースを提案する．以下では1回のシミュレーションの実行単位をシミュレーションモデルと呼ぶ．

3.1 設計の検討

インタフェースの設計に先立ち，対象となるシミュレーションモデルの操作も含んだ情報およびその構造について整理する．

（1） シミュレーションモデルの構成情報

シミュレーションを規定する情報は，上位の概念から大きく以下の3つのカテゴリに分類できる．

① 物理モデル：解析領域や支配方程式といった，対象とする問題の本来持っている，現象を規定するために入力される物理的情報．

② 数学モデル：メッシュやアルゴリズムといった数値シミュレーションを行うために付加される数学的情報．

③ 数値モデル：シミュレーションプログラムの作成や計算の実行などの処理プロセス，その実行結果として得られる数値的情報．

各モデルは，領域形状など図形情報で表現されるものと，数式などテキスト情報で表現されるものに大別される．その一覧を表1に示す．これらの個々の情報を容易かつ誤りなく定義し，シミュレーションを自然に実行できるインタフェースが必要となる．

（2） シミュレーション情報の構造

シミュレーションは通常1つの問題に対して，モデルの一部の修正，変更を施して何種類か実行して所期の目的を達成する．例えば，物理モデルでは物性値や条件式，数学モデルではメッシュ疎密やアルゴリズム，数値モデルではジョブの種類や計算機自体などが挙げられる．その都度，モデルやデータが生成され，モデル間の関連が発生する．図2にシミュレーションユーザが管理することになるデータの一般的な構成を示す．1つの問題はいくつかのシミュレーションモデルより成り，それぞれのシミュレーションモデルは固有の実行結果を有し，モデル変更のレベルによって対

表1 シミュレーションモデルの構成要素
Table 1 Construction list of simulation model.

カテゴリ	領域／図形情報	数式／テキスト情報	プロセス
物理モデル	次元 領域形状 時間領域	物理変数 物理定数（物性値） 支配方程式 初期条件 境界条件	
数学モデル	メッシュ タイムステップ	解法スキーム (アルゴリズム)	
数値モデル	実行結果 (グラフ/作画)	実行結果 (数値データ)	実行（計算）

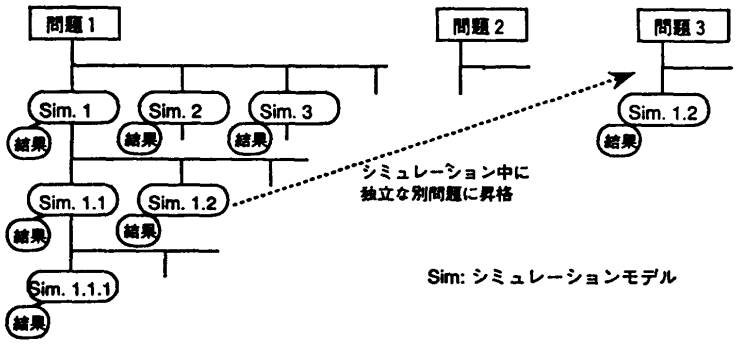


図2 シミュレーション情報の一般的構成
Fig. 2 General data construction of numerical simulation.

等もしくは子供の階層で管理される．この構造を管理できることが必要である．

3.2 マンマシンインタフェース

数値シミュレーションは，シミュレーションモデルの定義，その実行から成り，数値結果はシミュレーションモデルの付随情報である．そこで，シミュレーションモデルを主体として，すべての情報と操作を制御するオブジェクト指向的マンマシンインタフェースを提案する．

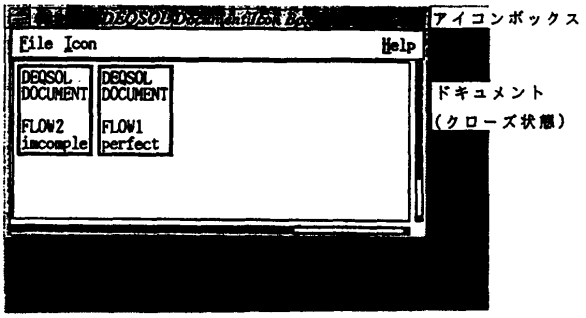
(1) 上位構造

ユーザの対象とする1つの問題は複数のシミュレーションモデルより構成され，ユーザは複数の問題を管理している構造を実現するため，ラン，ドキュメント，アイコンボックスという概念を導入する．これはそれぞれ図2のシミュレーションモデル，個々の問題，問題群に対応する．

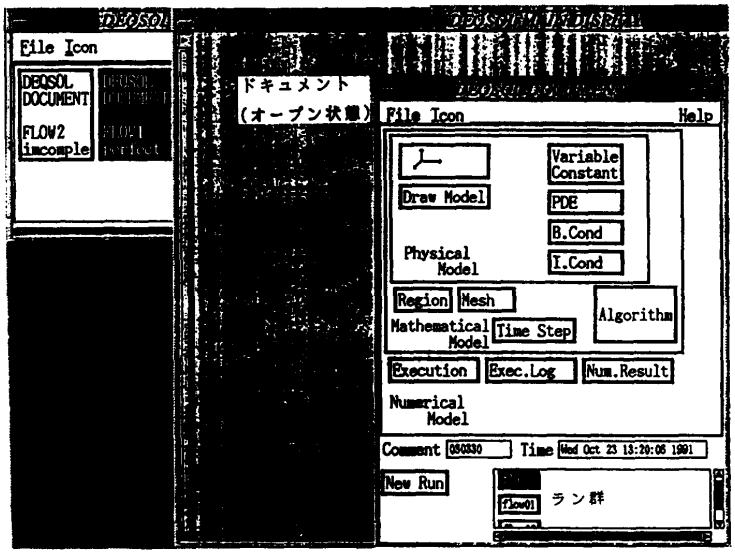
① ラン：1つのシミュレーションモデルを表す．

② ドキュメント：シミュレーションの対象となる1つの問題を表す．複数のランより成る．

③ アイコンボックス：個々の問題の全体集合を表す．複数のドキュメントから成る．



(a) アイコンボックスとドキュメント（クローズ状態）
(a) PDEQSOL document (close status).



(b) ドキュメント（オープン状態）とラン
(b) PDEQSOL document (open status).

図3 マンマシンインタフェース（上位部）
Fig. 3 Man-machine interface of visual PDEQSOL.

図3にそのマンマシンインタフェースを示す．四角はすべてボタンを表している．図3(a)のアイコンボックス内のドキュメントを表すボタンを押下すると，(b)に示すドキュメントが開かれる．ドキュメント内には複数のランがあり，その中の1つがアクティブになっている．ランの情報は，表1に挙げたカテゴリを代表する物理モデル，数学モデル，数値モデルに階層分けされたボタン群によって表示される．表2に各ボタンの目的・用途の一覧を示す．各ボタンはデータもしくは操作を表しており，その押下によって開かれるウィンドウによってそれぞれの情報の定義や参照が可能となる．実行等の指示操作（Execution ボタン）や実行結果のデータおよびその検証（Num. Result ボタン）もラン上のボタンによって代表される．左上から右下対角線方向に順次操作してくればシ

表2 Visual DEQSOL のボタン群一覧
Table 2 Buttons of visual PDEQSOL.

カテゴリ	ボタン名	機能
物理モデル	Dim.	領域の時空間次元を定義（画面上ではｱｲｺﾝ表示）
	Draw Model	領域形状を定義
	Variable/Constant	変数、定数を定義
	PDE	支配方程式を定義
	B. Cond	境界条件を定義
	I. Cond	初期条件を定義
数学モデル	Region	メッシュ分割の粗密制御単位を定義
	Mesh	自動メッシュ分割を指定
	Time Step	時間刻みを指定
	Algorithm	解法スキーム（アルゴリズム）を定義
数値モデル	Execution	実行を指定（実行形態、モードなど）
	Exec. Log	実行状態のロギングをアクセス
	Num. Result	実行結果へのアクセス、作画

ミュレーションの一連の定義や実行が終了するようにボタン群は配置している．

（2） 下位構造

シミュレーションモデルの定義や操作はドキュメント内の各ボタンを押下して現れるサブウィンドウを操作することにより行われるが，各サブウィンドウは2章の課題を解決する以下の目標の下に設計している．

① （ほとんど）学習なしで容易に使い熟せること．

② 既定義データの引用・参照と，データ間の無矛盾性を保証すること．

③ 入力に専門知識を要する事項はユーザレベルに応じガイダンスすること．

各ボタンとその操作例については付録に示す．また，上記③のガイダンスに関しては，数値解析の知識を必要とする境界条件入力やアルゴリズム展開を対象としており，第5章で述べる．

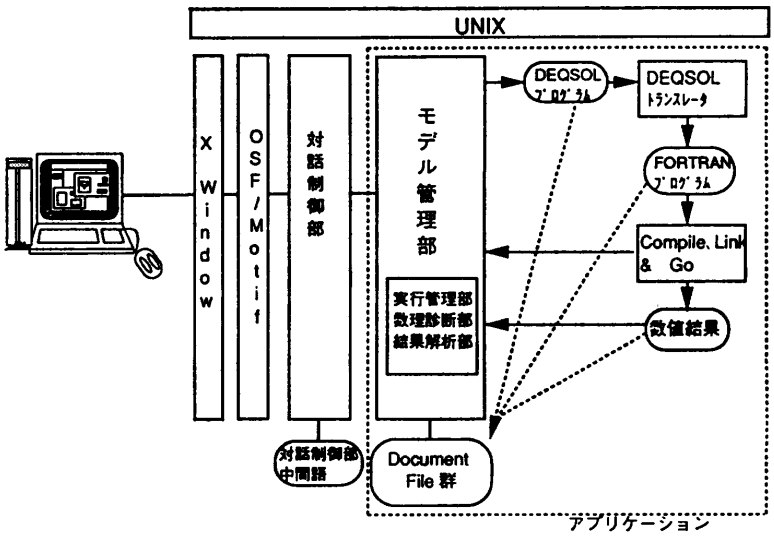


図 4 Visual DEQSOL のシステム構成
Fig. 4 System configuration of visual PDEQSOL.

4. システム構成

図4に本システムの構成を示す．前章で述べたインタフェースを介して入力されるデータは対話制御部を介してモデル管理部に与えられ，ドキュメントの定義やデータの追加，削除が行われる．シミュレーション情報が整うと，ユーザの指示でそれらは DEQSOL 言語に変換され，さらに DEQSOL トランスレータにより FORTRAN のシミュレーションプログラムに変換され，計算が実行される．新たに生成されるプログラムやデータはすべてシステム側で該当ドキュメントの付随情報として特殊ディレクトリで管理され，ユーザはまったく意識する必要はない．

システム設計にあたっては，対話操作の標準化，対話性能，拡張性，移植性が主要課題となる．対話操作の標準化および移植性の保証に対しては，GUI (Graphical User Interface) の標準化動向の1つである OSF/Motif (Open Software Foundation)[18] 上でのシステム構築を試みた．また，対話性能と拡張性に対しては，GUI 上に直接アプリケーションを構築するのではなく，アプリケーションに送信する必要のない対話情報のバッファリングやウィンドウの表示制御を行う対話制御部を GUI とアプリケーションの間に設ける UIMS (User Interface Management System)[17] の構成を採用した．またこの制御部に操作対象となっているランのみの情報（対話制御中間語）を持たせ，ウィンドウの変更や既定義情報の参照の高速化を図っている．これにより，対話制御で処理が閉じる場合は1秒以内，アプリケーションに処理が及ぶ場合は3秒以内の対話性能を達成している (CPU として MC 68030 使用のワークステーション上)．

5. ガイダンス機能

数値シミュレーションの情報の中には専門的な知識を有さずには正確な入力が困難もしくは不可能なものがある．本システムは，ユーザ入力の負担の軽減や使用前提となる知識の緩和を目的としたエキスパート的なガイダンス機能を有している．本章では境界条件入力とアルゴリズム展開のガイダンス方式について述べる．その特徴は，既入力情報を活かして無矛盾なモデル入力を誘導する点と必要最小限の入力による自動展開を行う点にある．

5.1 境界条件入力のガイダンス方式

シミュレーションモデルの定義における1つの特徴は，領域（形状）情報と式情報を連結する情報の存在にある．連結する情報としては，境界条件や初期条件，あるいは式中に含まれる領域に依存する物性値を表す材質定数などがある．境界条件を例にとれば，「何を（条件式）」にあたる部分は式情報に，「何処に

(設定箇所)」にあたる部分は領域情報に依存している.

(1) ガイダンス方式

条件の設定箇所や本来の与え方を検討すると以下のことがわかる.

(a) 設定箇所は,解析領域自体の性質によって本来決まっている.境界条件を例にとれば,入口や出口や壁,または電極や接地面といった物理的な性質で決まる部分領域が対象箇所となる.したがって偏微分方程式や対象定数・変数名称の種別が変わっても,箇所自体は共通なケースが多い.

(b) 同一条件を設定すべき箇所は,形状の定義可能単位より一般に大きく,しかも複連結領域や分断した領域になっているケースもある.

(c) 必要となる条件の種別(初期,境界条件など)や条件式の形式は,式情報が与えられた時点で本来決定している.

以上のことを考慮し,条件式の不適合や設定の過不足を回避する以下のガイダンス方式を考案した.

① 領域のグループ化の導入

物理的意味を与える手段として,形状の各単位に対して,そのグループ化を導入する.1つのグループは,形状単位の辺または面の集合であり,モデルに付随する物理的意味を表すのに必要な個数導入できるものとする.例えば,各グループは境界種別,材質種別,支配方程式の成立領域などを表す.

② 設定箇所の自動抽出

情報入力に先立って,どのグループを引用するかをあらかじめ指定させ,あとはシステムが該当情報の設定対象箇所をグループの構成要素単位ごとに自動抽出する.

③ 条件式の自動抽出

入力式に対して,その式が必要とする条件式の種別とその式に適合する条件式の形式を自動抽出する.抽出法は次項で述べる.

④ 条件式入力のガイダンス

②③の情報を画面上にわかりやすく表示し,必要な情報の選択と不足情報の入力をユーザに指示させる.一種/二種の形式の選択と左辺式の入力のみで境界条件の入力を可能とする.

(2) 境界条件式の抽出方法

対象変数と支配方程式が与えられた際の境界条件式の抽出方法は以下の2ステップより成る.

(a) 支配方程式と対象変数の一対一対応付け

これは各対象式と置換される一種境界条件式を決定する上で必要となる.単一式の場合は自明であるので,以下連立式に対する手順を述べる.ベクトル表示されている式に対しては,各成分に対する連立式に展開して処理する.この対応付けは,対象式にもっとも深く関わっている対象変数を決める観点(①②)と,数値計算上の不都合を回避する観点(③④)から,以下の手順で決定する.

① 対象式が対象変数の時間微分項を含む場合,その式にその変数を対応付ける.

② 対象式が微分のかかっていない対象変数を含む場合,それがなければ2階微分のかかっている対象変数を含む場合,その式にその変数を対応付ける.

③ ②までで対応が付かなかった対象式に対しては(該当式は対象変数群に対して一階微分項しか含まないはず),残っている対象変数のうち,式に含まれる変数を優先して対応付ける.

④ 上記手順で決まった対応付けの候補に対して,各ステップで1式に対し2変数以上の候補がある場合には,全体で一対一の対応パターンが存在するかを調べ,それを最終的な対応付けとする.

例えば,流体の挙動を表すナビエ・ストークス方程式では,①の規則から運動方程式には流速を表す変数が,連続の方程式には③の規則から圧力を表す変数が割り当てられる.

(b) 境界条件式左辺の抽出

各対象式に対して,(a)によって対応付けられた対象変数に対する一種条件式と,式に適合する二種条件式(該当式の境界からのフラックス項)を抽出する.

① 一種条件式

(a)によって対応付けられた対象変数に対して,

(対象変数)=

を一種条件式の形式とする.

② 二種条件式

対象式のすべての項を左辺に移項した式の左辺式の,ラプラシアン演算子(DEQSOL では lapl と表記)とdiv を含む項のみをその項の前に付く符号も合わせて出現順に抽出した式に対して,

(i) lapl の部分を ngrad に置換

(ii) div の部分を n・(境界外向き単位法線ベクトルの内積)に置換

して得られた式によって,

(置換式)=

を二種条件式の形式とする．例えば

$$A*\mathrm{div}(T\cdot\mathrm{grad}(N))+V\cdot\mathrm{grad}(N)+\mathrm{div}(U)=B*\mathrm{lapl}(M)$$

に対して，

$$A*n\cdot(T\cdot\mathrm{grad}(N))+n\cdot(U)-B*\mathrm{ngrad}(M)=$$

を導出する．

以上，境界条件入力のガイダンス方式について述べたが，初期条件設定の場所依存の物性値の設定なども，まったく同様の方式でガイダンスが可能である．そのマンマシンインタフェースについては付録の図9(c)に一例がある．

5.2 アルゴリズム展開のガイダンス方式

数値計算を行う場合，一般に原式に含まれる以下の，

Ⓐ 非定常性（時間微分項の有無）
Ⓑ 連立性
Ⓒ 非線形性

に対するアルゴリズム展開が必要になる．手順的にはいずれの場合でも線形の偏微分方程式を繰り返し解くことによってアルゴリズムが構成される．これはFORTRAN 等の一般的なプログラミング言語を用いる場合もまったく同等である．

図5に DEQSOL 言語によるアルゴリズムの記述例を示す．これは非定常，非線形の原式(a)に対して，非定常性はバックワードオイラー法，非線形性はニュートンラフソン法[19]によってアルゴリズム展開した計算手順を示している．DEQSOL における SOLVE 文は，線形の空間偏微分方程式の求解を機能としており，その繰り返しを指示する制御文（ITERation から END ITERation までのイテレーションブロック）の二重のネスト構造でアルゴリズムを記述している．

（1） ガイダンス方式

原式のアルゴリズムへの展開を検討すると以下のことがわかる．

① 上記Ⓐ～Ⓒに対するアルゴリズム展開に関する専門的知識が必要となる．さらにこれらが複合的に含まれている場合には複雑となる．知識は，アルゴリズムのバリエーションやその展開方法，それに必要となる記憶容量や計算量などの計算機リソースの推定，さらにその安定性や収束性などの数理的な性質などから成る．

② アルゴリズムの中には，非圧縮性流体解析の SMAC 法[19]や半導体デバイスシミュレーションのシャーフェッター・ガンメル法[19]のように応用分野におけるモデルの数理的性質から固有に開発されたものと，上記Ⓐ～Ⓒのそれぞれに対する要素アルゴリズムを組み合せて構成されるものがある．

③ アルゴリズム展開の手順には，Ⓐ～Ⓒの順序性が存在する．上位の要素アルゴリズムの選択によっては，下位の展開要因は自動的に解消するケースがある．例えば，非定常／非線形式のアルゴリズム展開で非定常性に陽的なアルゴリズムを採用すると非線形性は自動的に解消する．

④ アルゴリズムの構築は，大きく分けて，構築に必要となって派生する変数の追加（例えば時間反復での1ステップ前の関数値を保持する変数），原式のアルゴリズム展開，初期／境界条件式の展開・追加より成

```
変数　　: P
原方程式 : ∂P／∂t = div((A0+A1*P)*grad(P))
境界条件 : n..((A0+A1*P)*grad(P)) = 0  AT  L1,
           P=0                         AT  L2;
初期条件 : P=P1
```

（a） 原モデル
（a） Original model.

```
VAR  P, POLD, DP;                                  ・・・変数宣言
SVAR  PEPS;                                        ・・・スカラ変数宣言
COUNT  CTL;                                        ・・・カウンタ変数宣言
ICOND  POLD=P1;                                    ・・・初期条件定義
BCOND  n..((A0+A1*P)*grad(P))=0     AT  L1,        ・・・境界条件定義
       n..((A0+A1*P)*grad(DP))= !?  AT  L1,          (*)
       n..(A1*DP*grad(P))= !?       AT  L1,          (*)
       P = 0                        AT  L2,
       DP = 0                       AT  L2;          (*)

SCHEME;
  ITER NT UNTIL NT  GE  1000;                      ・・アルゴリズム定義
    ITER CTL UNTIL [PEPS LE 0.001] OR [CTL GE 200];
      SOLVE  DP  OF
        (P+DP-POLD)/dlt= div((A0+A1*P)*grad(P)) + div((A0+A1*P)*grad(DP))
                        +div(A1*DP*grad(P))
      BY 'PCG';
      P= P+DP;
      CALL NORM2(PEPS,DP);
    END ITER;
    POLD=P;
  END ITER;
END SCHEME;
```

（b） DEQSOL プログラム展開例
（b） Example of numerical algorithm expansion by PDEQSOL.

図5 原モデルと DEQSOL によるアルゴリズム記述例
Fig. 5 Original model and its numerical algorithm by PDEQSOL.

る．

⑤ 通常は（物理）変数を定義し，その変数を用いて式を定義するというトップダウンの定義フローが自然だが，アルゴリズム展開の場合，アルゴリズムの選択によって派生する変数が異なるため，アルゴリズム選択・展開後の式に必要な変数や条件式を追加するという定義の逆戻りが生じる．

以上のことを考慮し，アルゴリズム展開を原情報のみから可能とする以下のガイダンス方式を提案する．

① アルゴリズム構築要因の自動抽出

ユーザは物理モデルが本来もっている原情報のみを入力し，入力された支配方程式に対して，非定常性，連立性，非線形性をシステムが抽出する．この際，連立式の場合，全式が非定常項を有するか，非線形性は双一次線形ではないか，なども検出する．

② アルゴリズム選択のガイダンス

アルゴリズム展開にあたって，分野固有のアルゴリズムを用いるか，要因ごとに展開する一般的なアルゴリズムを指定させる．前者の場合にはその一覧からの選択を，後者の場合には①で抽出された各要因に対して，要素アルゴリズムの選択肢をⒶ～Ⓒの順に表示し，選択を促進する．選択肢は計算機リソース負荷とアルゴリズムの確実性（安定性，収束性など）の２観点から，非定常性には陽解法と陰解法，連立性には逐次解法と一括解法[10]，非線形性には逐次解法（割線反復法）とニュートン法，さらに必要に応じて行列解法には反復法と直接法など，代表的な数種をレパートリーとする．上位の選択で自動的に解消される要因は省き最短のガイダンスを行う．その具体的なフローを**図6**に示す．非定常性に陽解法が選択され全式が非定常式の時や，連立性に逐次解法が選択され非線形性が双一次線形であった時など，ガイダンスを省略できる．

（2） アルゴリズムの自動展開方法

前項の選択に従ってシステムが，アルゴリズム展開を自動的に行う．本項では要因ごとのアルゴリズム展開の全体フローについて述べる．展開の具体的手順については付録に示す．

展開はⒶ～Ⓒの順に実現できる．それぞれの展開は後述するように変数の追加，初期・境界条件の展開，対象式の展開からなり，各展開の出力を下位の展開の入力として順次行えば良い．図５で挙げた例で，各展開フローの結果を**図7**に示す．展開の各段で，変数の追加と前段の条件式の展開，対象式（もしくはSOLVE文）がITERationの制御ブロックよりなる展開式への置換が行われ，結果として最終的なSOLVE文の対象式は線形式になる．

（3） マンマシンインタフェース

アルゴリズム展開のガイダンスのマンマシンインタフェースを示す．ユーザがアルゴリズムの入力を指定すると（SCHEMEウィンドウ中のSOLVEボタンの押下），以下のフローで入力がガイダンスされる．

① 入力方式の選択

図8（a）に示すように，まず，システムのアルゴリズム展開のガイダンスを受けるか否か（Guidance/User）を指定する．ユーザ自身でアルゴリズム構築する場合には，Userを選択すれば，通常のSOLVE文の入力テンプレートが現れる．ガイダンスを選択すると以下のフローとなる．

② 方程式の選択

図8（b）に示すように，どの式に対してアルゴリズム展開をするかシステムが問い合わせてくる（Select EQU）．ユーザは式ウィンドウの任意の既定義式を選

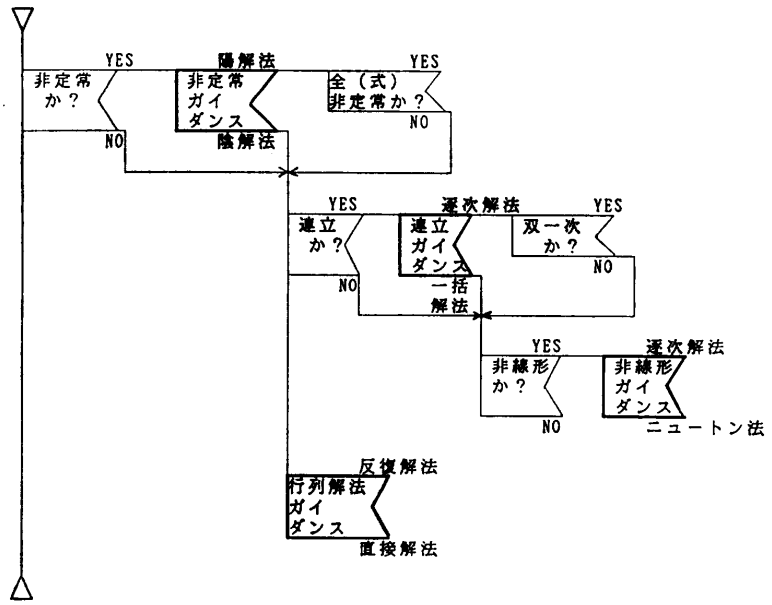


図6 アルゴリズム展開ガイダンスフロー
Fig. 6 Guidance flow of numerical algorithm expansion.

択する．

③ 展開方式の選択

アルゴリズムの中には，前節で述べたアルゴリズム構築が必要となる個々の要因（非定常性／連立性／非線形性）を手法に沿って展開していって得られるものと，非圧縮性流体解析における SMAC 法や半導体デバイスシミュレーションにおけるシャーフェッター・ガンメル展開のように式や分野に特定されているものに分類される．前者は Element，後者は Embedded で選択する．後者は，分野ごとに定型化したアルゴリズムのテンプレートをシステムが内蔵し，そのレパートリーの中から選択が行われる．以下は前者を選択した場合のフローである．

④ アルゴリズム構築手法の選択

図8（c）に示すように，選択した式のアルゴリズム構築が必要となる個々の要因をシステムが自動抽出し，その上位の要因から，それぞれの解法の選択をガイダンスする．テンプレートが随時拡張され，図7のフローに沿ってまだ残っている要因が表示される．上位の選択で下位の要因が解消した場合は，その時点でガイダンスが完了する．例えば本例では，非定常性に陽解法（Explicit）が選択されると，それでアルゴリズム展開が完了し，それ以降の選択ガイダンスは現れない．

以上，アルゴリズム展開のガイダンス方式について述べた．本方式およびマンマシンインタフェースの特徴は，ユーザが入力するのは原式と原変数のみで，そのアルゴリズム展開によって派生する変数や式展開はシステムが完全に保証する点と，ユーザによる最小限の選択でアルゴリズム展開をガイダンスする点にあ

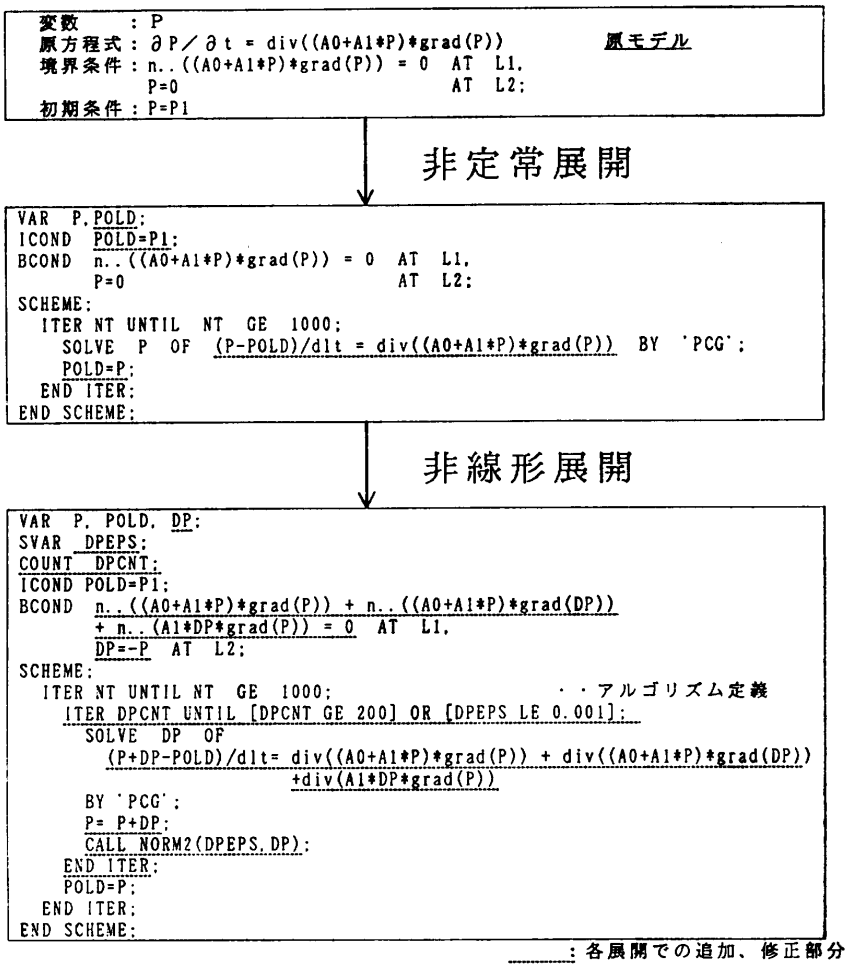
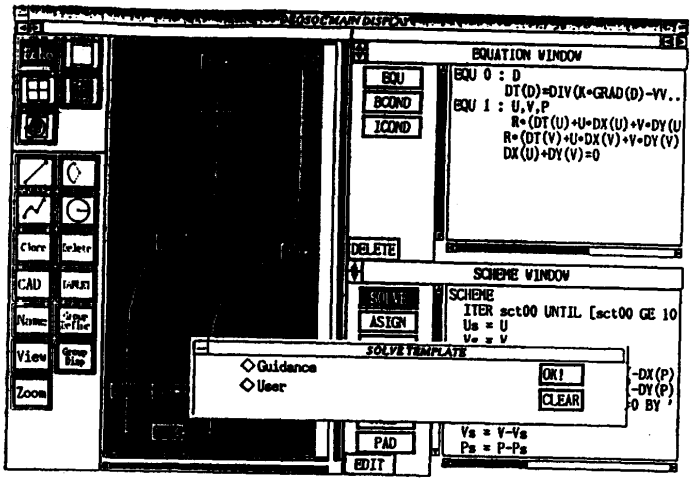
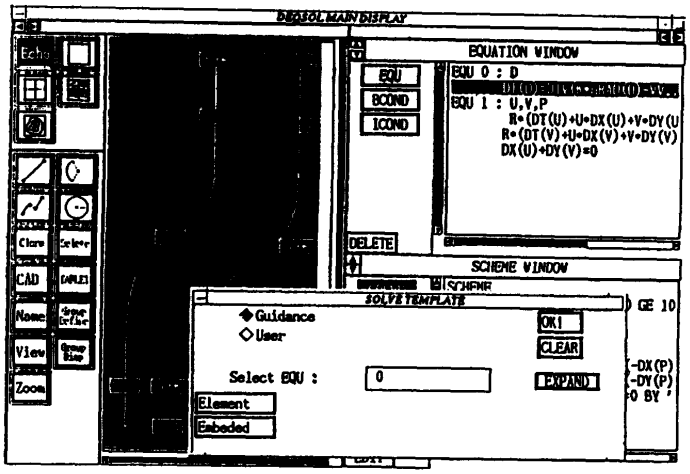


図 7　アルゴリズム展開フローの結果例（非定常／非線形）
（[バックワード・オイラー]+[ニュートン・ラフソン]）

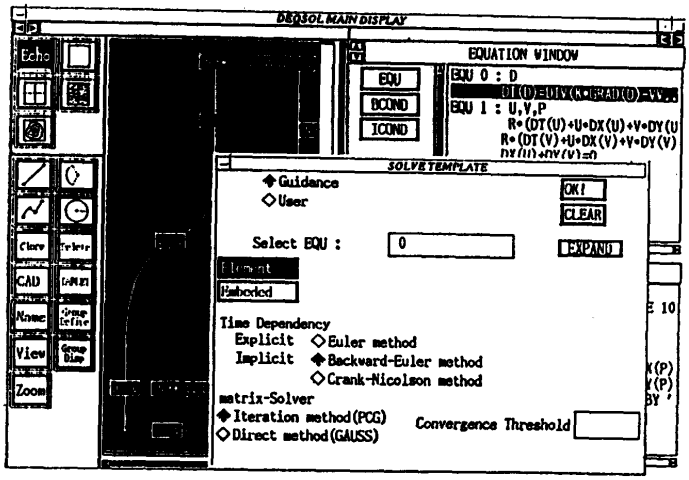
Fig. 7　Expansion flow of numerical algorithm. (Time dependent and non-linear equation.)



(a) 入力方式の選択
(a) Selection of inputting mode.



(b) 方程式の選択
(b) Selection of objective equation.



(c) アルゴリズム構築手法の選択
(c) Selection of algorithm expansion method.

図 8 アルゴリズム展開の入力ガイダンス画面
Fig. 8 Guidance windows for expanding numerical algorithms.

る.

6. 評　　価

本システムは，可視環境を利用したマンマシンインタフェースを備え，計算機やプログラミングの知識なしに，数値シミュレーションを一貫して実行できることを目標としている．マンマシンインタフェースの評価は客観的には難しいが，対話システムの一般的評価指標・要件として挙げられている[16]，表現のわかりやすさ，操作学習の容易性，ガイダンス機能，画面レイアウト，画面表示順序などについては前章までに述べた．以下では，記述性，工数削減効果について評価する．

(1) 記述性の評価

シミュレーションモデルの定義を最小限の言語セットで可能としている例として，DEQSOL 言語を対象に本システムの記述(操作) 性の向上について述べる．記述性は記述量的なものと，入力者の前提知識の緩和や信頼性の向上につながるものが挙げられる．

① 形状記述部は DEQSOL プログラム行数の 50% 以上を占めているが，本システムでは領域の定義や引用は可視情報とグループによって行うので，その労力は大幅に削減される．

② 数式情報の定義では，原変数と原式から条件式入力やアルゴリズム展開をガイダンスするので，入力は大幅に容易になる．ユーザの入力量は，変数の数が 1/2～1/4，条件式は設定箇所や式左辺が自動抽出されるので 1/2 以下に，アルゴリズムは自動展開されるので 1/10 程度となる．しかも，条件式の過不足や不適合，アルゴリズムの入力ミスなどが起こらない．

(2) 工数削減効果

ベンチマークとして，DEQSOL で解析したことのある数題を評価した．このうち，FORTRAN で 90 人日，DEQSOL 言語で 13 人日要した Electron Beam 装置の電位分布解析は[12]，本システムでは 2 人日で解析を終えた．領域形状やメッシュが

複雑なのか，アルゴリズム構築が大変なのかによって削減効果は依存するが，本評価では，DEQSOL 言語使用時の約 1/5，FORTRAN 等の従来言語使用時の 1/10～1/30 の削減が可能であった．

7. おわりに

数値シミュレーション向き高水準言語 DEQSOL を発展させ，ワークステーション環境を活かし数値シミュレーションを一貫して支援する対話型システム・ビジュアル DEQSOL を試作した．本システムは，一般的なシミュレーションモデルに対する階層的なウィンドウインタフェースを備え，その上でモデル定義，計算実行，結果検証のすべての工程を行うことができる．

本システムは計算機やプログラミングに関する特別な知識なしにシミュレーションを実行できる環境を念頭に設計している．また，ユーザの数値計算に関する知識を緩和するためのガイダンス機能を持たせている．シミュレーションの全体工数への効果として，ベンチマークテストでは DEQSOL 言語使用時の約 1/5，FORTRAN 等の従来言語使用等の 1/10～1/30 の削減を達成した．

現在，本システムは 2 次元問題を対象としているが，3 次元への拡張で特に影響を受けるのは領域形状の定義とそのメッシュ分割機能部分のみである．DEQSOL 自体はすでに 3 次元機能を有しており，インタフェース仕様，システム構成，さらにガイダンス機能等はそのまま適用できるものである．本システムでは 2 次元 CAD を内蔵させたが，3 次元では既存の CAD システムからの形状情報データの入力が現実的なアプローチとなろう．

謝辞 本システムのマンマシンインタフェースの初期設計にあたっては，当時日立製作所中央研究所の客員研究員であったジム・デラハント氏にご協力いただいた．

参考文献

1) Rice, J.: *Solving Elliptic Problem Using ELLPACK*, p. 497, Springer-Verlag (1984).
2) Rice, J.: ELLPACK—An Evolving Problem Solving Environments for Scientific Computing, *Proc. IFIP TC 2/WG 22. 5*, pp. 233-245 (1985).
3) Dyksen, W. R.: Elliptic Expert: An Expert System for Elliptic Partial Equations, Purdue Univ. Tech. Report, CER-88-48, CSD-TR-840 (1988).
4) Schonauer, W. and Schnepf, E.: Software Considerations for "Black Box" Solver FIDISOL for Partial Differential Equations, *ACM Trans. Math. Softw.*, Vol. 13, No. 4, pp. 333-350 (1987).
5) Cook, G. O.: ALPAL: *A Tool for the Development for Large-Scale Simulation Codes, UCID-21482*, Lawrence Livermore National Laboratory (1988).
6) 梅谷征雄：数値シミュレーション用プログラミング言語 DEQSOL，情報処理学会論文誌，Vol. 26, No. 1, pp. 168-180 (1985).
7) Umetani, Y.: DEQSOL—A Numerical Simulation Language for Vector/Parallel Processors, *Proc. IFIP TC 2/WG 22. 5*, pp. 147-164 (1985).
8) Konno, C.: A High Level Programming Language for Numerical Simulation: DEQSOL, *IEEE Denshi Tokyo*, No. 25, pp. 50-53 (1986).
9) Konno, C.: Automatic Code Generation Method of DEQSOL, *J. Inf. Process.*, Vol. 11, No. 1, pp. 15-21 (1987).
10) Konno, C.: Advanced Implicit Solution Function of DEQSOL and Its Evaluation, *Proc. Fall Joint Computer Conf.*, pp. 1026-1033 (1986).
11) Konno, C.: *The BF (Boundary-Fitted) Coordinate Transformation Technique of DEQSOL*, SIAM, ISBN 0-89871-228-9, pp. 322-326 (1988).
12) 平山裕之：DEQSOL による電子線描画装置内電位分布解析，第 34 回（昭和 62 年前期）情報処理学会全国大会論文集，pp. 81-82 (1987).
13) 佐川暢俊：有限要素法 DEQSOL の流体シミュレーション向き機能の検討，第 37 回（昭和 63 年後期）情報処理学会全国大会論文集，pp. 42-43 (1988).
14) 金野千里：ビジュアル DEQSOL システム 1：工程分析および機能構成，第 38 回（昭和 64 年前期）情報処理学会全国大会論文集，pp. 66-67 (1989).
15) Konno, C.: Interactive/Visual DEQSOL: Interactive Creation, Debugging, Diagnosis and Visualization of Numerical Simulation, *Mathematics and Computers in Simulation*, Vol. 31, pp. 353-369 (1989).
16) Shneiderman, B. (東 元基 (訳))：ユーザー・インタフェースの設計，p. 385，日経 BP 社 (1987).
17) Hayes, P. J.: Design Alternatives for User Interfaces Management Systems Based on Experience with Cousin, *SSIGCHI '85: Human Factors in Computing Systems*, pp. 169-175 (1985).

18) Young, D. A.: *The X Window System—Programming and Applications with Xt*, OSF/Motif ed., p. 533, Prentice-Hall (1990).
19) 村田健郎ほか(編):工学における数値シミュレーション, p. 322, 丸善 (1988).

付　　録

(1) マンマシンインタフェース

以下では，シミュレーションフローに沿ったマンマシンインタフェースの例を示す．

各ボタンが押下されると，そのボタンが代表する既定義の情報内容を表示するウィンドウであるメインディスプレイと，定義や操作指示の入力を促進するメニューやテンプレートなどのサブウィンドウが自動的に現れる．シミュレーションモデルを表す情報は，表1に示したように大きく分けて形状に関するものと式などのテキストで表されるものから成る．メインディスプレイは，形状情報を表示する部分とテキストを表示する部分をタイリングして表示する．形状かテキストのいずれか一方で十分なボタンの押下では一方だけが，両方に関する情報の時は両方がシステム側の判断で表示される．例えば形状(Draw Model)入力では形状のみが，境界条件(B. Cond)入力では両方が表示される．

メニューやテンプレートなどのサブウィンドウはメインディスプレイにオーバラップして表示され，自由に画面内を移動できる．

シミュレーションのフローに沿ったいくつかの例を示す．図9は各ボタンが押下された時に表示されるメインディスプレイと，現れるメニュー，テンプレートの例を示している．(a)に示す形状の入力では，現れたメニューを利用して，既定義形状上の点などを定義して入力を進める．(b)に示す偏微分方程式の入力では，既定義の変数と登録されている演算子を引用して入力を進める．(c)に示す境界条件の入力では，テンプレート上に適合する条件パターンが表示され，それを選択すれば良い．メッシュ定義，アルゴリズム定義も同様に行われる．計算の実行は，(d)に示す Execution ボタンによって現れるテンプレート上で，モデルのエラーチェックか本実行か，対話実行かバックグランドジョブによるバッチ実行かを指定する．計算結果は Num. Result ボタンによって現れる出力変数一覧からの選択とグラフ化仕様をテンプレート上で指定すると，(e)に示す等高線図等が表示される．

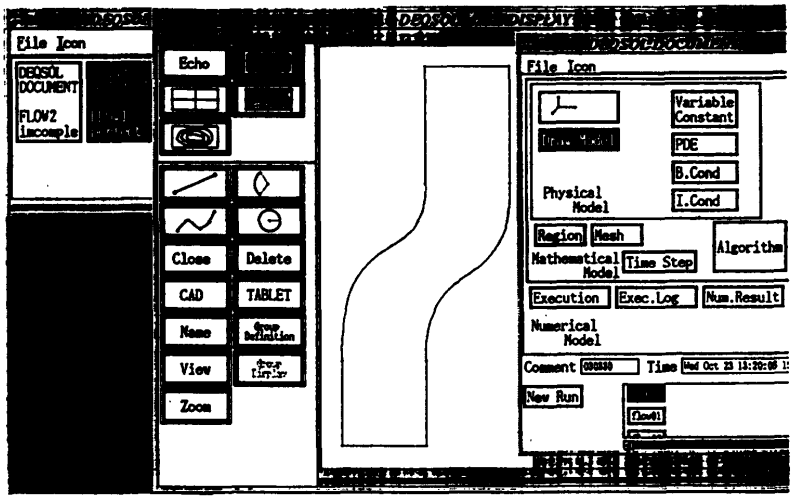


図 9 (a) 領域形状入力 (DRAW Model 押下時)
Fig. 9 (a) Inputting figure of domain.

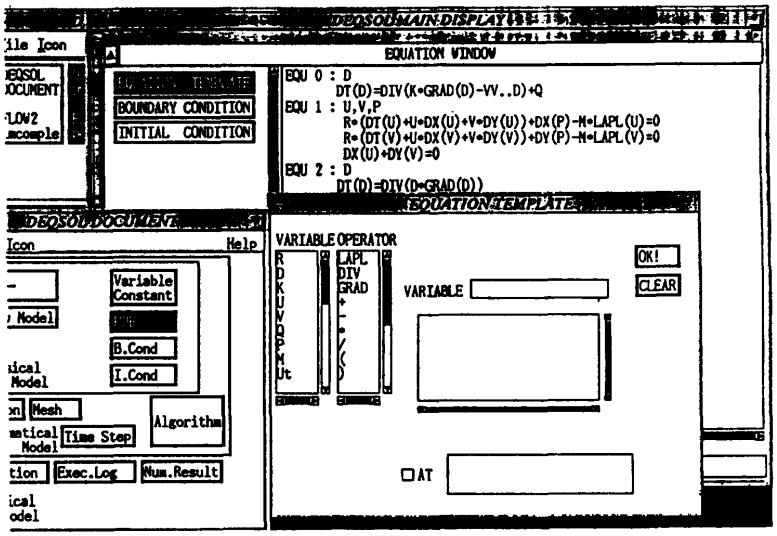


図 9 (b) 支配方程式入力 (PDE 押下時)
Fig. 9 (b) Inputting dominating equations.

（2） アルゴリズムの展開法

以下ではアルゴリズムの具体的な自動展開法として，非線形性に対するニュートン法の展開方式について述べる．非定常性，連立性の展開も同様の方式で実現できる．なお，連立性の逐次解法における展開では，各式をどの変数に関して解くかは，5.1 節(2)(a)で示した式と変数の対応付けを用いる．

ニュートンラフソン法は原偏微分方程式の与えられた変数値の微小変分から成る一次式を導出し，その微小変分を求めて変数値を繰り返し修正しながら，解への収束を図る解法である．その自動展開の処理方式について，図 10 の具体例を引用して説明する．

（a） 変数追加

アルゴリズム展開に必要とする以下の変数を追加する．

① 収束反復計算を行うので，対象変数の変分を保持しておく新規変数を対象変数分追加する（図 10 (a)-①）．

② 収束判定用に，①の変分のノルムを保持するスカラ変数を追加する（図 10 (a)-②）．

③ 収束反復のイテレーションの回数を制御するカウンタ変数を追加する（図 10 (a)-③）．

（b） 境界条件式の展開

① 一種条件式の展開

一般的な一種条件の形式を $P=P1$ とすると（変数名を P，(a)-① で導入された対応する変分を表す変数名を仮に DP とした），本式を，

$$DP=P1-P$$

に置換すれば良い．求められる解は $P=P+DP$ により更新されるので，本条件式により原一種条件が常に保証されることになる．

② 二種条件式の展開

対象式の発散項（div を含む項）に非線形性をもっている非線形偏微分方程式の境界条件式も非線形となる．

したがって該当する境界条件式の線形化の展開が必要となる．本展開は次に述べる対象式の展開と同じ処理を行えば良い．ただし，div が $n\cdot$(境界外向き法線ベクトルとの内積）に置き換わっているとみなす．これにより，本来適合する境界条件式は，本展開後も，アルゴリズム展開された式に適合することが保証される．

（c） 対象式の展開

① イテレーションブロックを生成する．収束条件はユーザより指定された収束判定値を用いて生成する．なお，イテレーション条件には無限ループを回避する

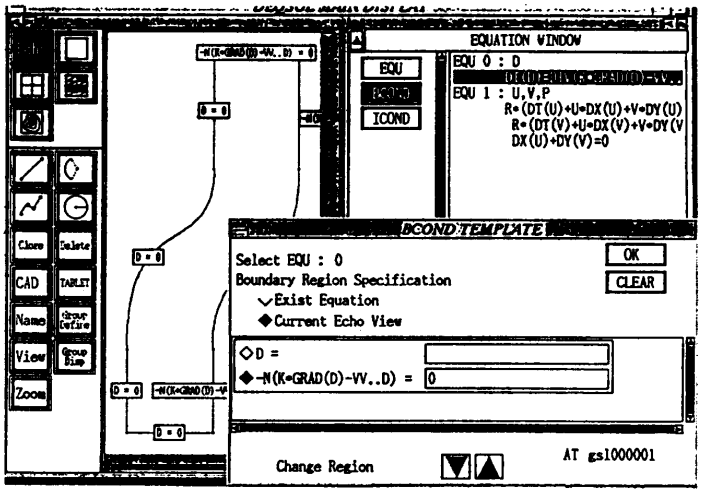


図 9 （c） 境界条件入力 (B. Cond 押下時)
Fig. 9 (c) Inputting boundary conditions.

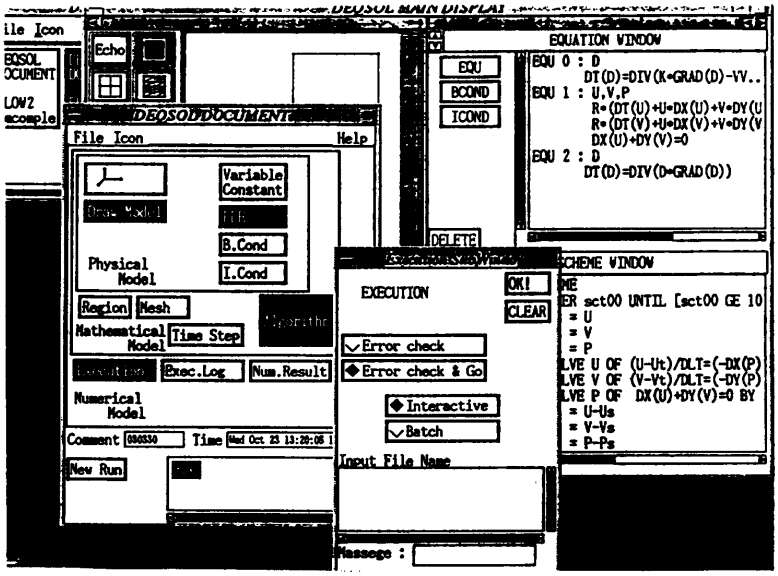


図 9 （d） シミュレーション実行指示 (Execution 押下時)
Fig. 9 (d) Directions of simulation execution.

条件が付加されている．本イテレーションブロック内に②以降の実行文が生成される．

② 繰り返し解かれる SOLVE 文を生成する（図10（c）-②）．SOLVE 文の対象変数は（a）-①で導入した変分を表す変数で，対象式は変分に対して原偏微分方程式から導出した線形化した式である．この線形化は以下の方式で行う．

（i） 対象式の中から，対象変数を含んだ非線形項を抽出する．

（ii） 各項に対して，

F*div (G·grad (H))

G·grad (H)

E

の各 E, F, G, H の各部に該当する項を検出する．

（iii） E, F, G, Hの各項に対して，対象変数に関する一階の偏微分を求める．連立であれば各対象変数ごとに求める．

（iv） 各 E, F, G, Hに対して（iii）で得た一階偏微分を用い，変分に対する一次近次式を求める．ここで対象変数を仮に P, Q, その微小変分を DP, DQ とすると，具体的には，

$F(P+DP) \rightarrow F(P)+F_P*DP$

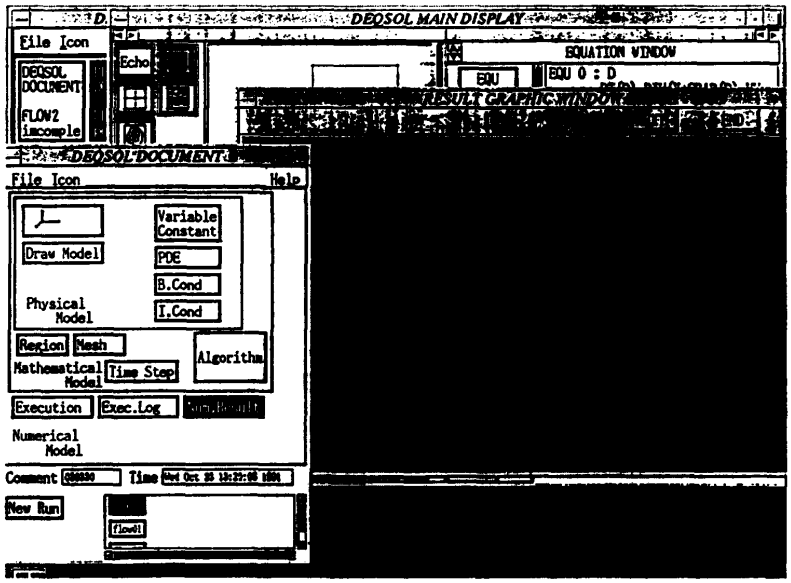
$F(P+DP, Q+DQ) \rightarrow F(P,Q)+F_P*DP+F_Q*DQ$（連立のケース）を求める．ここで F_P, F_Q は F の P 偏微分，Q 偏微分である．

（v） （ii）の各項の対象変数を微小変分を加えた式に変形し，すなわち，例えば，

$F(P+DP)$*div($G(P+DP)$ ·grad($H(P+DP)$))

を得，各 $F(P+DP)$, $G(P+DP)$, $H(P+DP)$ を（iv）で得た式に置換する．

（vi） （v）の各項を，内側より，grad の線形展開，内積の乗算展開，div の線形展開，全体の乗算展開を順次行う．



（e） 計算結果検証 (Num. Result 押下時)
（e） Verification of simulation result.

図 9 マンマシンインタフェース（シミュレーションフロー）
Fig. 9 Man-machine interface of simulation flow.

```
VAR  P;
BCOND  P=P1  AT  L1,
       n..((A0+A1*P)*grad(P))=M*(P-P0)  AT  L2, ・・・;

SCHEME;
  SOLVE  P  OF  div((A0+A1*P)*grad(P))=0   BY  'PCG';
END SCHEME;
```

↓

```
VAR  P, DP;                                                    (a)-①
SVAR  DPEPS;                                                   (a)-②
COUNT  DPCNT;                                                  (a)-③
BCOND  DP = P1-P  AT  L1,                                      (b)
       n..((A0+A1*P)*grad(P)) + n..((A0+A1*P)*grad(DP))
        + n..(A1*DP*grad(P)) = M*(P+DP-P0)   AT  L2, ・・・;

SCHEME;
  /* 必要であれば、Pの初期値の設定式を挿入する */

  ITER DPCNT UNTIL [DPCNT GE 100] OR [DPEPS LE 0.001];         (c)-①
    SOLVE  DP  OF  div((A0+A1)*grad(P))
         + div((A0+A1*P)*grad(DP)) + div(A1*DP*grad(P))=0      (c)-②
    BY 'PCG';
    P = P + DP;                                                (c)-③
    CALL NORM2(DPEPS, DP);                                     (c)-④
  END ITER;
END SCHEME;
```

図 10 ニュートン法による非線形式の展開
Fig. 10 Mechanism of algorithm expansion flow for non-linearity by Newton method.

（vii） （vi）で得られた式の各項のうち，微小変分を2つ以上含んでいる項を削除し，最終式を得る．

③ 対象変数の値を微小変分を加えることによって更新する代入文を生成する（図10（c）-③）．

④ 収束判定用の微小変分のノルムを計算する CALL

文を生成する（図 10（c）-④）.

なお，本反復解法のスタート値（初期値）はシステムによって適切な値を自動設定することはできないので，必要ならユーザによって，本手順で生成されるイテレーションブロックの前で記述されることを想定している.

(平成 4 年 1 月 8 日受付)
(平成 4 年 5 月 14 日採録)

金野 千里（正会員）

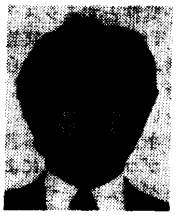
昭和 27 年生．昭和 50 年東京工業大学理学部情報科学科卒業．昭和 52 年同大学院修士課程修了．同年(株)日立製作所入社．以来，中央研究所にて，図形処理システム，数値計算技術の研究・開発に従事．応用数理学会会員.

梅谷 征雄（正会員）

昭和 19 年生．昭和 43 年東京大学理学部数学科卒業．同年(株)日立製作所入社．以来，同社中央研究所にて，デジタルシステムの故障診断，ベクトル化コンパイラ，高水準言語 DEQSOL，並列記述言語などの研究に従事して現在に至る．工学博士．応用数理学会，ACM 各会員.

太田 忠（正会員）

昭和 37 年生．昭和 59 年法政大学工学部電気工学科卒業．同年日立超 LSI エンジニアリング(株)入社．以来，数値シミュレーション向き高水準言語 DEQSOL の開発に従事.



深田 肇

昭和 39 年生．昭和 63 年北海道大学理学部地質学鉱物学科卒業．同年日立ソフトウェアエンジニアリング(株)に入社．以来，数値計算プログラムの開開発に従事.

山賀 晋

昭和 41 年生．平成元年青山学院大学理工学部物理学科卒業．同年日立超 LSI エンジニアリング(株)に入社．以来，数値シミュレーション向き高水準言語 DEQSOL の開発に従事.

池田美以子

昭和 37 年生．昭和 60 年図書館情報大学同学部同学科卒業．同年日立超 LSI エンジニアリング(株)入社．以来，数値シミュレーション向き高水準言語 DEQSOL の開発に従事.