「エクセルマクロ」による拡散問題の数値解法

1. 「エクセルマクロ」とは

「エクセルマクロ」とは，エクセルのワークシートをデータの入出力先としたプログラムを作成することができる機能である. C 言語や Visual Basic 言語によってプログラムを作成する場合，入力データが書かれたテキストファイル等を予め用意し，あるいはプログラム中に入力データを直接書き込み，入力データを読み込んでプログラムを実行した後，実行結果をテキストファイルや CSV ファイル（エクセルで開くことができるファイル）に出力するのが一般的である．この場合，出力されたファイルの中身は数字や文字の羅列であり，得られた結果の正当性や視覚的な内容を直ちに判断することができない．これに対し,「エクセルマクロ」の機能を使ってプログラムを作成する場合，入出力データがエクセルの 1 つのファイル内に集約されるだけでなく，エクセルのグラフ作成機能によって，出力データをリアルタイムで確認することができる．したがって，プログラムの不備や誤りを早急に発見することができ，非常に効率よくプログラムを完成させることができる.

2. 「エクセルマクロ」を使うための準備

「エクセルマクロ」の機能を使うためには，以下の設定をする必要がある.

1) エクセルを起動し，[ツール]→[マクロ]→[セキュリティ]をクリックして，**図 1** の画面を起動する.
2) セキュリティ※) レベルを「高」，「中」，「低」のいずれかに選択し, OKボタンをクリックする.

※）これは，マクロ機能によって作成されたウイルスが，ファイルを開いたと同時に実行されるのを未然に防ぐための機能である.

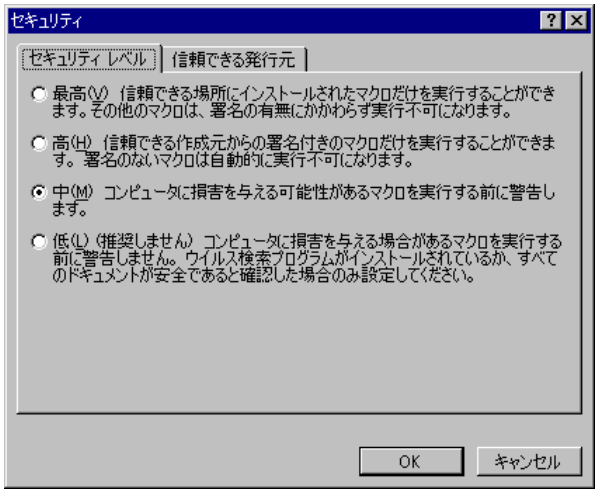

図 1　マクロ機能を使うための初期設定

3. 「エクセルマクロ」の使い方

エクセルを起動し，[ツール]→[マクロ]→[Visual Basic Editor]をクリックする．Visual Basic Editor が起動したら，[挿入]→[標準モジュール]をクリックする．すると，**図 2** のような画面が起動する．右側の白い部分にプログラムを記述することにより，エクセルのワークシートとの関連付けや，数値計算を行う.

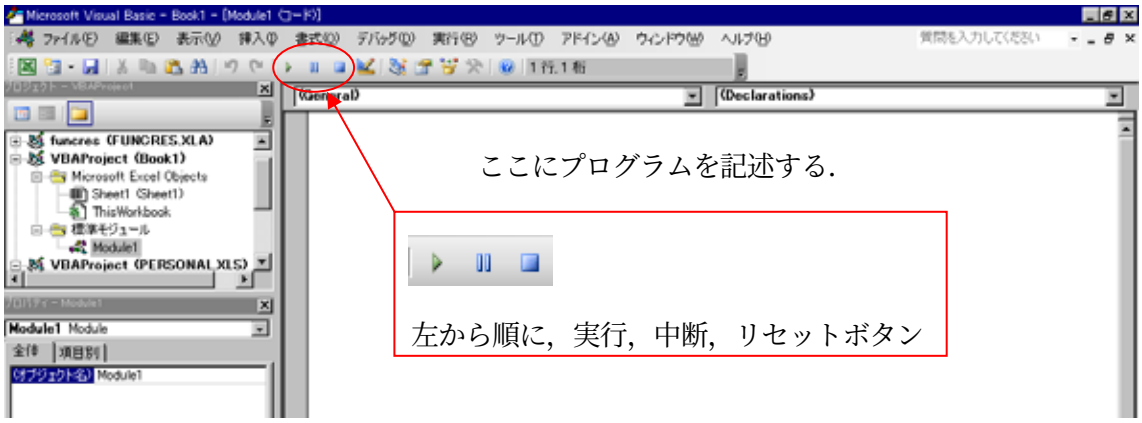

図 2　Visual Basic Editor の基本画面

プログラムを記述した後，プログラムを実行するには実行ボタン を押す.

プログラムを実行し，いつまで経っても結果が出力されない場合，プログラムが間違っている可能性があるので，中断ボタンを押し，プログラムを修正する.

プログラムを実行し, 何らかのエラーメッセージが表示されたら, リセットボタンを押し, プログラムを修正する.

4. 陽解法による拡散問題の数値解法（演習の(2)）

4.1　演習内容

厚さ $l=1$ の壁があり，壁厚方向の温度分布の経時変化を求めたい．壁を構成している材料の密度を $c=1$，比熱を $\rho=1$，熱伝導率を $\kappa=1$ とする．時刻tにおける壁厚方向の温度分布を$u(x,t)$と書く（xは壁の一方の面からの距離）．初期温度は$u(x,0)=u_0=1$ とする．境界条件は，時刻$t>0$において，壁表面から流出する熱流束が$q=m(u_s-u_{ext})$（mは熱伝達係数で$m=2$，u_s は壁表面の温度，u_{ext}は外気温度で$u_{ext}=0$）で表されるとする.

支配方程式：$\dfrac{\partial u}{\partial t}=\dfrac{\partial^2 u}{\partial x^2}$　　$(0<t, 0<x<l)$

初期条件：$u(x,0)=u_0=1$　　$(0\leqq x\leqq l)$

境界条件：$q=m(u_s-u_{ext})=2\,u_s$　　$(0\leqq t, x=0,l)$

本問題を差分法で解くために，壁を厚さ方向にM等分し，$\Delta x=l/M$と書く．時間に関する刻みを Δt として，$t=n\cdot\Delta t$ と書く．また，$u(i\cdot\Delta x, n\cdot\Delta t)$を$u_{i,n}$と書く.

ここでは，壁厚の分割数を $M=10$，時間の刻みを $\Delta t=0.004$ とする.

陽解法によって解くための解法公式（漸化式）は次のようになる.

$$u_{0,n+1} = (1 - 4r \cdot \Delta x - 2r) \cdot u_{0,n} + 2r \cdot u_{1,n} \qquad (i=0) \qquad (1)$$

$$u_{i,n+1} = r \cdot u_{i-1,n} + (1 - 2r) \cdot u_{i,n} + r \cdot u_{i+1,n} \qquad (i=1 \sim M\text{-}1) \qquad (2)$$

$$u_{M,n+1} = 2r \cdot u_{M-1,n} + (1 - 4r \cdot \Delta x - 2r) \cdot u_{M,n} \qquad (i=M) \qquad (3)$$

ここに， $r = \Delta t / (\Delta x)^2$

※注意

$r > \dfrac{1}{2 + 4 \cdot \Delta x}$ のとき，(1)式の右辺第 1 項，あるいは (3)式の右辺第 2 項が負の値となる．

また， $r > \dfrac{1}{2}$ のとき，(2)式の右辺第 2 項が負の値となる．

$\dfrac{1}{2 + 4 \cdot \Delta x} < \dfrac{1}{2}$ であるから，$r < \dfrac{1}{2 + 4 \cdot \Delta x}$ の範囲であれば，(1)～(3)式の全ての項は正の値となる．

したがって， $r > \dfrac{1}{2 + 4 \cdot \Delta x}$ のとき，数値計算によって得られた解は不安定となる．

4.2　計算条件の入力とデータの出力

図 3 に示すように，エクセルのワークシートに **4.1** の条件を入力する．なお，ワークシートの名前は「input」とする．また，計算結果を出力するワークシート「output」も作成しておく．出力先のワークシート「output」は空欄でよい．

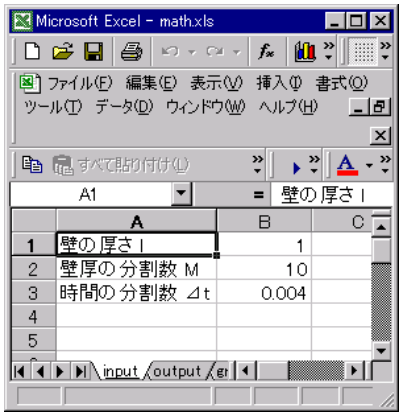

図 3　計算条件の入力

4.3　プログラムの記述

Visual Basic Editor を起動して標準モジュールを挿入（**3.**参照）し，プログラムを記述する．

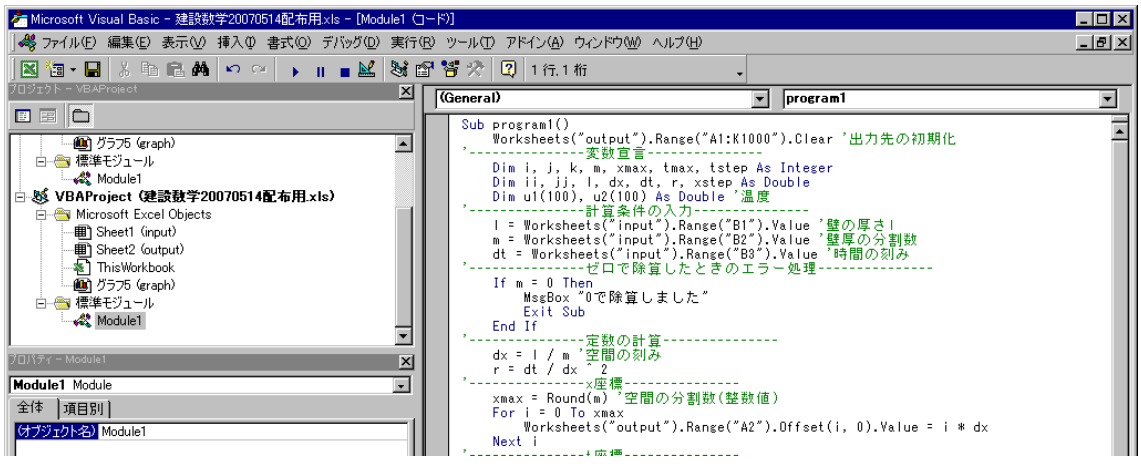

図 4　プログラムの記述

陽解法によるプログラムソース

```
--------------------------------------------------------------------------------
1:        Sub program1()
2:            Worksheets("output").Range("A1:K12").Clear '出力先の初期化
3:        '--------------変数宣言--------------
4:            Dim i, j, k, m, xmax, tmax, tstep As Integer
5:            Dim ii, jj, l, dx, dt, r, xstep As Double
6:            Dim u1(100), u2(100) As Double '温度
7:        '--------------計算条件の入力--------------
8:            l = Worksheets("input").Range("B1").Value '壁の厚さ l
9:            m = Worksheets("input").Range("B2").Value '壁厚の分割数
10:           dt = Worksheets("input").Range("B3").Value '時間の刻み
11:       '--------------ゼロで除算したときのエラー処理--------------
12:           If l * m = 0 Then
13:               MsgBox "0 で除算しました"
14:               Exit Sub
15:           End If
16:       '--------------定数の計算--------------
17:           dx = l / m '空間の刻み
18:           r = dt / dx ^ 2
19:       '--------------x 座標--------------
20:           xmax = Round(m) '空間の分割数(整数値)
21:           For i = 0 To xmax
22:               Worksheets("output").Range("A2").Offset(i, 0).Value = i * dx
23:           Next i
24:       '--------------t 座標--------------
25:           tmax = 200 '時間の分割数(整数値)
26:           tstep = Round(tmax / 10) '時間の出力間隔
27:           For jj = tstep To tmax Step tstep
28:               j = Round(jj / tstep)
29:               Worksheets("output").Range("B1").Offset(0, j - 1).Value = jj * dt
30:           Next jj
31:       '--------------初期条件 u(x,0)=u0=1--------------
32:           For i = 0 To xmax
33:               u1(i) = 1
34:           Next i
35:       '--------------時間を更新して計算開始--------------
36:           For j = 1 To tmax
37:           '境界条件 1 u(0,t)
38:               u2(0) = (1 - 4 * r * dx - 2 * r) * u1(0) + 2 * r * u1(1)
39:           '差分方程式
40:               For i = 1 To xmax - 1
41:                   u2(i) = r * u1(i - 1) + (1 - 2 * r) * u1(i) + r * u1(i + 1)
42:               Next i
43:           '境界条件 2 u(l,t)
44:               u2(xmax) = 2 * r * u1(xmax - 1) + (1 - 4 * r * dx - 2 * r) * u1(xmax)
45:           '温度分布の更新
46:               For k = 0 To xmax
47:                   u1(k) = u2(k)
48:               Next k
49:           '出力
50:               If j Mod tstep = 0 Then
51:                   For i = 0 To xmax
52:                       k = Round(j / tstep)
53:                       Worksheets("output").Range("B2").Offset(i, k - 1).Value = u1(i)
```

```
54:                     Next i
55:                 End If
56:             Next j
57:         '--------------解の安定評価--------------
58:             If r * (2 + 4 * dx) - 1 > 0 Then
59:                 MsgBox ("解が不安定です")
60:             End If
61:         End Sub
```

4.4　実行結果のグラフ化

4.3 で作成したプログラムを実行し，何もエラーがなかった場合，実行結果がワークシート「output」に出力される．最後に，実行結果をグラフ化し，結果が正しいかどうかを判断する．結果は**図 5** のようになる．

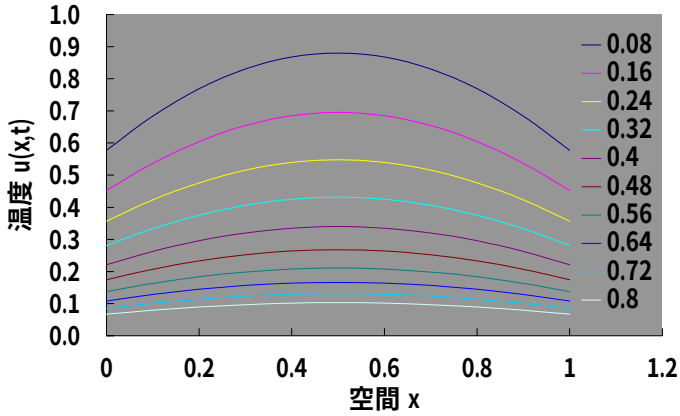

図 5　実行結果

5.　プログラムの書き換え（演習の(5)）

4.3 で作成したプログラム内容を書き換えることにより，初期温度や熱伝達係数の違いを考慮した計算を行うことができる．具体的に変更する項目は以下の通りである．

図 6　入力値の追加

1)　計算条件を追加する．

・ワークシート「input」に入力値を追加：
図 6 参照

・プログラムソースの 5 行目：
「**Dim ii, jj, l, dx, dt, r, xstep As Double**」→
「**Dim ii, jj, l, dx, dt, r, xstep, u0, uext1, uext2, c, rho, kappa, m1, m2 As Double**」

・プログラムソースの 10 行目と 11 行目の間に以下の記述を追加：

「**u0 = Worksheets("input").Range("B4").Value '初期条件 u(x,0)=u0**」
「**uext1 = Worksheets("input").Range("B5").Value '外気温 1**」
「**uext2 = Worksheets("input").Range("B6").Value '外気温 2**」

「c = Worksheets("input").Range("B7").Value '材料の密度 c」

「rho = Worksheets("input").Range("B8").Value '比熱 ρ」

「kappa = Worksheets("input").Range("B9").Value '熱伝導率 κ」

「m1 = Worksheets("input").Range("B10").Value '熱伝達係数 m1」

「m2 = Worksheets("input").Range("B11").Value '熱伝達係数 m2」

2) 初期条件を変数にする.

・プログラムソースの 33 行目：

「u1(i) = 1」→「u1(i) = u0」

3) 境界条件式を密度，比熱，熱伝導率，熱伝達係数，外気温度を考慮した形にする．特に，両端で熱伝達係数と外気温度が異なるように，境界条件式を与える必要がある.

・プログラムソースの 38 行目：

「u2(0) = (1 - 4 * r * dx - 2 * r) * u1(0) + 2 * r * u1(1)」→

「u2(0) = 2 * m1 / c / rho * r * dx * uext1 + (1 - 2 * m1 / c / rho * r * dx - 2 * kappa / c / rho * r) * u1(0) + 2 * kappa / c / rho * r * u1(1)」

・プログラムソースの 44 行目：

「u2(xmax) = 2 * r * u1(xmax - 1) + (1 - 4 * r * dx - 2 * r) * u1(xmax)」→

「u2(xmax) = 2 * kappa / c / rho * r * u1(xmax - 1) + (1 - 2 * m2 / c / rho * r * dx - 2 * kappa / c / rho * r) * u1(xmax) + 2 * m2 / c / rho * r * dx * uext2」

4) 差分方程式を密度，比熱，熱伝導率を考慮した形にする.

・プログラムソースの 41 行目：

「u2(i) = r * u1(i - 1) + (1 - 2 * r) * u1(i) + r * u1(i + 1)」→

「u2(i) = kappa / c / rho * r * u1(i - 1) + (1 - 2 * kappa / c / rho * r) * u1(i) + kappa / c / rho * r * u1(i + 1)」

5) エラー処理に，密度あるいは比熱がゼロの場合を追加する.

・プログラムソースの 12 行目：

「If l * m = 0 Then」→「If l * m * c * rho = 0 Then」

6) 解の安定評価をする判定式を，密度，比熱，熱伝導率，熱伝達係数を考慮した形にする.

・プログラムソースの 58 行目：

「If r * (2 + 4 * dx) - 1 > 0 Then」→

「If 2 * r * (m1 * dx + kappa) / c / rho - 1 > 0
 Or 2 * r * (m2 * dx + kappa) / c / rho - 1 > 0 Then」

6. 補足

プログラム中に「**'時間の刻み**」のような記述があるが，Visual Basic 言語では「 ' 」（上付きカンマ）以降に書かれた文字列は，プログラムの命令文として認識されない.「 ' 」はプログラム中に注釈を加えたいときによく用いる.

プログラム中の数式において，「**r = dt / dx ^ 2**」のように，演算子の前後には必ずスペースが挿入されている．これは，「**r=dt/dx^2**」と記述して改行すれば自動的に挿入されるものなので，キーボードからわざわざ入力しなくてもよい．

プログラム中に数式などの長い記述をする場合，「_」（アンダーバー）を利用して改行するとよい．例えば，以下の記述は全く同じ内容である．

「**e = a + b + c + d**」

「**e = a + b _**
+ c + d」

7. プログラミングについて

プログラミングの第 1 歩は，他人が書いたプログラムを真似することである．ただし，電子コピーしたプログラムをそのまま利用するだけでは全く勉強にならない．プログラムソースが書かれた資料をスキャナで読み取り，画像データを OCR ソフトで電子化してプログラムを利用するのも論外である．プログラムの内容が全くわからなくても，一文字ずつキーボードを叩くうちに，プログラムの全体の流れを徐々に把握できるようになる．一通り真似をしてみて，どの部分を書き換えれば自分が必要とする計算結果が得られるのかと考えるだけでも，十分勉強になる. プログラムや数学が嫌いな人, 苦手な人にとってはかなり酷なことかもしれないが，是非この課題にチャレンジして欲しい．