第4章　電気・電子工学編

目に見えない電気工学や電子工学の現象を目に見えるようにすれば，これらの分野を理解するのに大きな助けになります．この章では，このような観点から電気・電子工学の分野の中からいくつかのテーマを選びました．これ以外にも電気・電子の分野でコンピュータを使った計算に向いているテーマはたくさんあるので，みなさんも自分でプログラムを組んでみてください．そうすることでよりいっそう電気や電子が好きになるはずです．

4.1　交流波形

家庭に送られて来る電気は時刻と共に周期的に方向が反転しています．このような電気を交流と言いいます．交流にも様々な波形がありますが，家庭の電気は正弦波と呼ばれ，これを数式で表すと，

$$v(t) = V_m \sin(\omega t - \theta)$$

となります．ここで $\omega = 2\pi f$　　(f：周波数),θ: 位相, V_m: 最大値) です．表 4.1 は，実効値，周波数，位相を与えて正弦波交流波形を描くプログラムです．正弦波交流で 100 ボルトと言えば普通は最大値ではなく実効値を指します．最大値は実効値の $\sqrt{2}$ 倍です．図 4.1 は実効値を 100[V]，周波数を 50[Hz]，位相を 0[°] とした時の交流波形です．

表 4.1: 交流波形

```
DEF v(x)=SQR(2)*Ve*SIN(2*PI*f*t+ph)  !  交流波形を定義
LET Ve=100                           !  実効値
LET f=60                             !  周波数
LET ph=0                             !  位相 (単位はラジアン)
LET Ts=1/f                           !  周期
LET Ti=Ts/25                         !  時間間隔は周期の 25 分の 1
LET Tend=3*Ts                        !  範囲は周期の 3 倍
SET WINDOW -Tend/10,Tend,-2*Ve,2*Ve
DRAW GRID(0.01,25)
FOR t=0 TO Tend STEP Ti
   PLOT LINES: t,v(t);
NEXT t
END
```

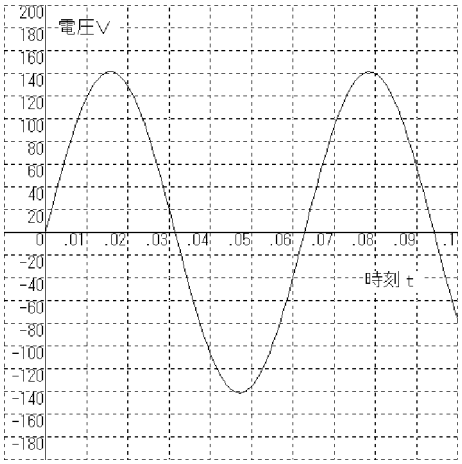

図 4.1: 交流波形（電圧 100V，周波数 50Hz，位相 0 °）

4.2 ベクトル軌跡

交流回路で周波数を変化させると電圧や電流，インピーダンスなどはもちろん変化します．この時，これらの値が周波数を徐々に変えた時にどのような値 (ベクトル値で) をとるか順に計算し図にしたものをベクトル軌跡と言います．言い換えると，周波数を少しずつ変化させたときの電流や電圧などのベクトルの終端がどのように変化していくかを図にしたものです．なおベクトルの始点は原点です．図 4.2 のような RL 直列回路に角周波数ωの正弦波交流 (実効値 E) を加えると流れる電流 $\dot{I}$ は

$$\dot{I} = \frac{E}{R + j\omega L} = \left(\frac{R}{R^2 + \omega^2 L^2} - j \frac{\omega L}{R^2 + \omega^2 L^2} \right) E$$

となります．ωを 0 から∞まで変化させると電流ベクトルの軌跡は中心の座標 $(\frac{E}{2R}, 0)$，半径 $\frac{E}{2R}$ の円の下半分の半円となります（証明は各自で試みて欲ください．方法は，$\dot{I} = x + jy$ とおいて x, y をそれぞれωの関数として求め $(x = f(\omega), y = g(\omega))$，これらの式を連立させてωを消去して x と y の関係を求めればよい.）．表 4.2 には，上記で述べた電流İ のベクトル軌跡を描くプログラムを，図 4.3 に実行結果を示します．

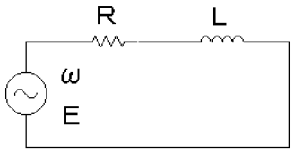

図 4.2: RL 直列回路

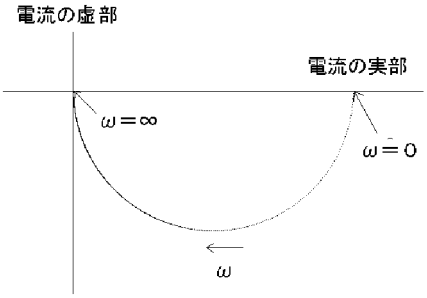

図 4.3: ベクトル軌跡

表 4.2: ベクトル軌跡のプログラム

```
DEF Ir(omega)=R/(R^2+(omega*l)^2)*E          !  電流の実数部
DEF Im(omega)=-omega*l/(R^2+(omega*l)^2)*E   !  電流の虚数部
LET R=10                                     !  抵抗は 10 オーム
LET L=0.1                                    !  コイルは 0.1 ヘンリー
LET E=1                                      !  電圧は 1V
SET POINT STYLE 1
SET WINDOW -0.125,0.125,-0.125,0.125
DRAW GRID
FOR omega=0 TO 10000 STEP 10                 !  周波数は 0 から 10kHz
   PLOT POINTS: Ir(omega),Im(omega)          !  まで
NEXT omega
END
```

4.3　共振回路

交流回路において電源の周波数を変化させると電流の大きさはある周波数で極大になったり (直列共振回路の場合) 極小になったり (並列共振回路の場合) します．図 4.4 のように抵抗 R[Ω]，インダクタンス L[H]，静電容量 C[F] を直列に接続し交流電圧を加えると電流 I が流れて各素子両端の電圧は

$$V_R = RI\ ,\quad V_L = \omega LI\ ,\quad V_C = \frac{1}{\omega C}I$$

となります．ここで V_R は I と同相，V_L は I より $\pi/2$[rad] 進み，V_C は I より $\pi/2$[rad] 遅れています．したがって電圧のベクトル図は図 4.5 のようになり ($V_L > V_C$ のとき)，V は

$$\begin{aligned} V &= \sqrt{V_R^2 + (V_L + V_C)^2} = \sqrt{(RI)^2 + (\omega LI - \frac{1}{\omega C}I)^2} \\ &= I\sqrt{R^2 + \left(\omega L - \frac{1}{\omega C}\right)^2} \end{aligned}$$

となります．よってこの回路のインピーダンス Z[Ω] は

$$Z = \sqrt{R^2 + \left(\omega L - \frac{1}{\omega C}\right)^2}$$

となります．ところで，この回路において電源の周波数を変化させると $\omega L = \frac{1}{\omega C}$ となって Z が最小となる周波数 f_0[Hz] があるはずです．この様子を図 4.6 に示します．

このとき Z は

$$Z = \sqrt{R^2 + \left(\omega L - \frac{1}{\omega C}\right)^2} = \sqrt{R^2 + (0)^2} = R$$

であり，電流は最大値

$$I = \frac{V}{Z} = \frac{V}{R}$$

となります．このような周波数と電流の関係を図示すると図 4.7 のようになります．これを共振特性と呼びます．また共振周波数 f_0 を考えると，$\omega L = \dfrac{1}{\omega C}$ より $\omega^2 = \dfrac{1}{LC}$ ですから，$\omega = 2\pi f_0$ より

$$f_0 = \frac{1}{2\pi\sqrt{LC}}$$

となります．f_0 を共振周波数といい，このような状態になったとき「共振した」と言います．表 4.3 に図 4.4 の共振回路の電流を求めるプログラムを，またその結果を図 4.7 に示します．この図を見ると角周波数 1000[rad/s](すなわち周波数 159Hz) で電流が最大 ($I = \frac{V}{R} = \frac{1}{5} = 0.2$[A]) になっていることがわかります．

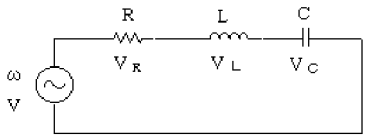

図 4.4: 共振回路

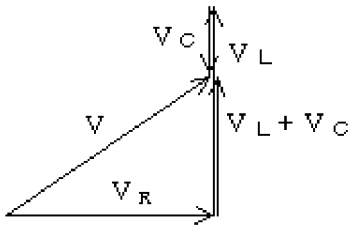

図 4.5: 共振回路のベクトル図

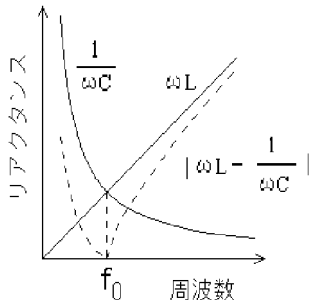

図 4.6: リアクタンスの周波数特性

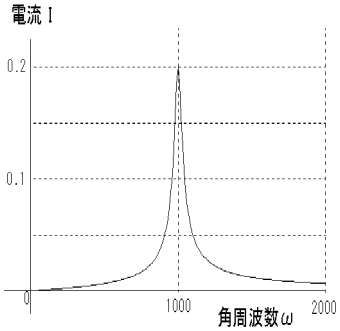

図 4.7: 共振回路の計算結果

表 4.3: 共振回路のプログラム

```
DEF x(omega)=omega*L-1/(omega*C)          !  リアクタンス
DEF i(omega)=E/SQR(R^2+x(omega)^2)        !  電流
LET R=5                                   !  抵抗値
LET L=0.1                                 !  インダクタンス
LET C=0.00001                             !  キャパシタンス
LET Oend=2000                             !  角周波数の最大値
LET E=1                                   !  電圧は 1V
SET WINDOW -Oend/10,Oend,-0.05*2/R,2/R
DRAW GRID(1000,0.25/R)
FOR omega=10 TO Oend STEP 10
   PLOT LINES: omega,i(omega);
NEXT omega
END
```

4.4 過渡現象（コンデンサの充電）

スイッチの開閉など回路の状態の変化に伴う電圧や電流の変動を過渡現象と言います．このような過渡現象は一般に微分方程式で表されることが多く，ここでは，その解法によく用いられる 4 次のルンゲ-クッタ法を使うことにします．

4.4.1 ルンゲ・クッタ法による過渡現象（コンデンサの充電）の解法

図 4.8 に示すような回路でのコンデンサの充電を考えます．初めコンデンサには電荷が無いとし，時刻 $t = 0$ でスイッチを入れ充電を始めます．このとき流れる電流を $i(t)$ とすると

抵抗の電圧＋コンデンサの電圧＝電源電圧

ですから，

$$Ri(t) + \frac{1}{C}\int i(t)dt = V$$

が成り立ちます．電流と電荷の間には $\frac{dq(t)}{dt} = i(t)$ が成り立ちますので，この式を

$$R\frac{dq(t)}{dt} + \frac{1}{C}q(t) = V$$

と書き直し 4 次のルンゲ・クッタ法で解きましょう．初期条件は $q(0) = 0$ です．図 4.9 に計算結果を示します．

図 4.9 には解析解 (すなわち式 4.4.1 を数学的に解いた答え)

$$i(t) = \frac{V}{R}e^{-t/CR}$$

も同時に示しています．解析解は図中 · で表しています．この図からルンゲ-クッタ法で精度良く微分方程式が解けていることがわかります．表 4.4 がコンデンサの充電を 4 次のルンゲ・クッタ法で解くプログラムです．

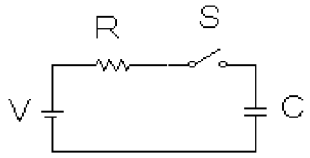

図 4.8: 過渡現象（コンデンサの充電）の回路図

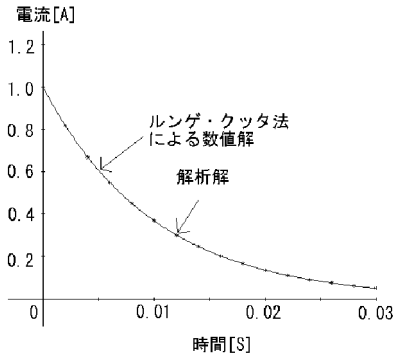

図 4.9: 過渡現象（コンデンサの充電）の計算結果

表 4.4: 過渡現象（コンデンサの充電）のプログラム

```
DEF fq(t,q)=V/R-q/R/C                    !  fq(t,q)=dq/dt=i
DEF i(t)=V/R*EXP(-t/tt)                  !  解析解
LET V=10                                 !  電源電圧
LET R=10                                 !  抵抗値
LET C=0.001                              !  容量
LET tt=C*R                               !  時定数
LET h=0.01*tt                            !  刻み幅
LET t=0                                  !  初めの時間はゼロ
LET q=0                                  !  初めの電荷はゼロ
SET WINDOW -tt/3,3*tt,-0.25*V/R,1.5*V/R
DRAW axes(tt/2,0.2)
```

```
DO UNTIL t>5*tt                  !  計算は時定数の 5 倍まで
   LET k1=h*fq(t,q)
   LET k2=h*fq(t+h/2,q+k1/2)
   LET k3=h*fq(t+h/2,q+k2/2)
   LET k4=h*fq(t+h,q+k3)
   LET dq=(k1+2*(k2+k3)+k4)/6
   LET q=q+dq
   LET t=t+h                     !  時間を刻み幅分だけ進める
   PLOT LINES: t,fq(t,q);
   IF MOD(t,tt/5)=0 THEN         !  時定数の間に 5 つの解析解を表示
      PLOT POINTS: t,i(t)
   END IF
LOOP
END
```

問　抵抗とコイルの直列回路 (図 4.8 で C の替わりに L を入れた直流回路) でスイッチを入れた時にどのように電流が流れるかの微分方程式を立てて，4 次のルンゲ・クッタ法で解いて求めてみよう．

4.5　変調

音声のような低い周波数の信号は遠くまで届きませんが 100kHz 以上の高周波信号は空間を伝わって何キロメートルも遠方まで届きます．従って低周波信号を遠くまで届けるには，低周波信号で高周波信号に変化をつけて送れば良いわけです．この操作を変調と言います．また逆の操作，すなわち変調された高周波信号から低周波信号を取り出すことを復調と言います．例えばラジオ放送では，放送局で音楽や人の声を変調しアンテナから電波として送り出します．AM 放送ならおおむね 550kHz から 1600kHz の電波を使っています．家庭ではラジオ受信機で受信した電波を復調して音声信号を取り出しスピーカーを鳴ならしています．高周波の電波のことを情報を運ぶという意味で搬送波と言います．変調には，搬送波の振幅を変

化させる AM 変調，周波数を変化させる FM 変調などがあります．

4.5.1　ＡＭ 変調

AM 変調では，搬送波の振幅を信号波で変化させます．搬送波を $v_0 = V_0 \sin \omega_0 t$，信号波を $v_s = V_s \sin \omega_s t$ とすれば，AM 波は

$$v(t) = V_0(1 + m \sin \omega_s t) \sin \omega_0 t$$

と表せられます．ここで $m = \dfrac{V_s}{V_0}$ は変調度です．表 4.5 に AM 変調のプログラムを，図 4.10 に AM 変調した波形を示します．図 4.10 では変調の様子が見やすいように搬送波の周波数を 1kHz，信号波の周波数を 100Hz，変調度を 0.5 としました．各自でこれらの値を変化させて見てください．

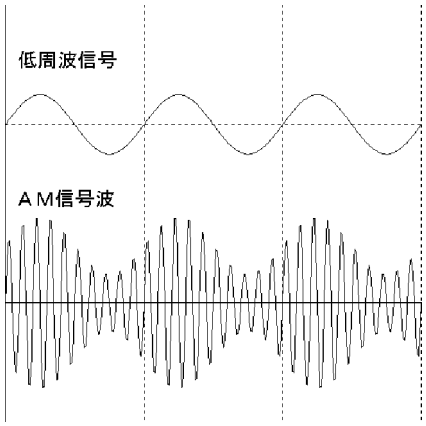

図 4.10: AM 変調の計算結果

表 4.5: AM 変調のプログラム

```
DEF v1(t)=m*SIN(omegas*t)   !  信号波
DEF v2(t)=V0*SIN(omega0*t)  !  搬送波
LET f0=1000
LET fs=100
LET omega0=2*PI*f0          !  搬送波の角周波数
LET omegas=2*PI*fs          !  信号波の角周波数
LET V0=1
LET m=0.5                   !  変調度
SET WINDOW 0,3/fs,-2*V0,5*V0
DRAW GRID0(1/fs,3*V0)
FOR t= 0 TO 3/fs STEP 0.01/fs
   PLOT LINES: t,V0*(1+v1(t))*v2(t);  !  AM 波形
NEXT t
PLOT LINES                             !  余分な線を消すため
FOR t= 0 TO 3/fs STEP 0.01/fs
   PLOT LINES: t,v1(t)+3;              !  信号波の波形
NEXT t
END
```

4.5.2 FM 変調

FM 変調では，搬送波の周波数を信号波で変化させます．搬送波と信号波を上記と同様とすると FM 波は，

$$v(t) = V_0 \sin\left(\omega_0 t + m_f \sin \omega_s\right)$$

で表されます．ここで m_f は信号波の振幅に相当し変調指数と言います．表 4.6 にプログラムを，図 4.11 に計算結果を示します．

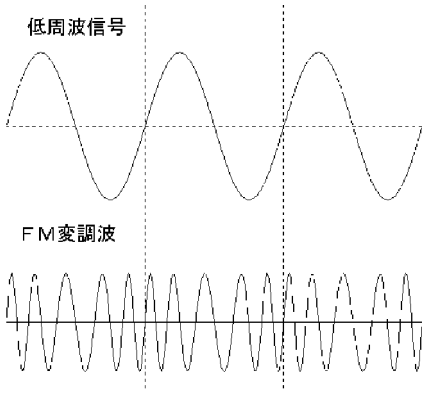

図 4.11: FM 変調の計算結果

表 4.6: FM 変調のプログラム

```
DEF v1(t)=m*SIN(omegas*t)          !  信号波
DEF v2(t)=V0*SIN(omega0*t+v1(t))   !  変調波
LET f0=1000
LET fs=200
LET omega0=2*PI*f0
LET omegas=2*PI*fs
LET V0=1
LET m=1                            !  変調度
SET WINDOW 0,3/fs,-2*V0,5*V0
DRAW GRID(1/fs,3*V0)
FOR t= 0 TO 3/fs STEP 0.01/fs
   PLOT LINES: t,v2(t);         !  FM 波形
NEXT t
PLOT LINES                      !  余分な線を消すため
FOR t= 0 TO 3/fs STEP 0.01/fs
   PLOT LINES: t,v1(t)+3;       !  信号波の波形
NEXT t
END
```

4.6 キルヒホッフ則

電気回路の一般的な解法を与えるのがキルヒホッフ則です．キルヒホッフ則を電気回路に適応すると連立方程式ができます．一旦，連立方程式ができてしまうと後は数学的にそれを解けば良いことになります．連立方程式を解く方法はいくつかありますが，ここでは “Gauss-Jordan 法” と呼ばれる方法を紹介します．

4.6.1 Gauss-Jordan 法 (掃出法)

$x_1, x_2, x_3, ..., x_n$ を解とする連立 1 次方程式

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}$$

において

$$\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b'_1 \\ b'_2 \\ b'_3 \\ \vdots \\ b'_n \end{pmatrix}$$

と変形できたとすると $(b'_1, b'_2, b'_3, \cdots, b'_n)$ が解 $(x_1, x_2, x_3, \cdots, x_n)$ となります．計算量は Gauss の消去法より多いけれども手順が簡単です．

計算手順は次の 2 ステップです．

1. 第 k 行を a_{kk} で割る ($k = 1,2,\cdots,n$ の順に)
2. $i \neq k$ の時だけ第 i 行から第 k 行の a_{ik} 倍を引く ($i = 1,2,\cdots,n$ の順に)

4.6.2　Gauss-Jordan 法を用いたキルヒホッフ則の解法

図 4.12 の直流回路網で電流 I_1, I_2, I_3 を求めることを考えます．点 c に

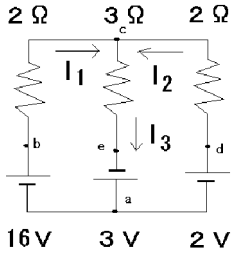

図 4.12: キルヒホッフ則のための直流回路網

キルヒホッフの第一法則を適用すると

$$I_1 + I_2 - I_3 = 0$$

が得られ，閉回路 a-b-c-e-a に第二法則を適用すると

$$2I_1 + 3I_3 = 19$$

が得られ，閉回路 a-d-c-e-a に第二法則を適用すると

$$2I_2 + 3I_3 = 5$$

が得られます．これらの式を連立させると電流 I_1, I_2, I_3 は

$$\begin{aligned} I_1 &= 5 \\ I_2 &= -2 \\ I_3 &= 3 \end{aligned}$$

となります．表 4.7 にプログラムを示しますが，実際にはゼロで割り算しないようにするような工夫が必要です．

表 4.7: キルヒホッフ則のプログラム

```
Dim a(9,9)
Dim b(9)
INPUT PROMPT "次数":  n
FOR i=1 TO n
   FOR j=1 TO n
   INPUT PROMPT "係数a":a(i,j)
   NEXT j
   INPUT PROMPT "係数b":b(i)
NEXT i
FOR k = 1 to n
   LET p = a(k, k)
   FOR j = 1 to n
     LET a(k,j) = a(k,j) / p
   NEXT j
   LET b(k) = b(k) / p
   FOR i = 1 To n
      IF i <> k THEN
         LET aik = a(i,k)
         FOR j = 1 To n
            LET a(i,j) = a(i,j) - a(k,j) * aik
         NEXT j
         LET b(i) = b(i) - b(k) * aik
      END IF
   NEXT i
NEXT k
FOR i = 1 TO n
   PRINT " b( ";i;" )= ";b(i)
NEXT i
```

プログラムの説明

- 1 行目で係数の，2 行目で定数および解の配列を宣言しています．配

列については次章 (4.7) を見てください.

- 3 行目から 9 行目まででデータを入力しています.
- 10 行目では第 k 行を a_{kk} で割るための k を変えるための FOR 文です.
- 12 から 15 行目で第 k 行を a_{kk} で割っていますが，前の行で a_{kk} を p に代入しています (ピボット選択). なぜこのようなことをするのか考えてみてください.
- 16 行目で引き算をする行を決めています.
- 11 行目で $i \neq k$ の時だけにしています.
- 12 から 18 行目までで第 i 行から第 k 行の a_{ik} 倍を引いています．やはりピボット (変数 aik) を使っています.
- 20 から 22 行目で答えを表示しています.

4.7　配列

キルヒホッフ則を解く際に `Dim a(9,9)` のような命令を使っています．ここで定義されている a(9,9) のような番号つきの変数を配列と言います．すなわち，配列は多数のデータを格納できるように番号 (これを添字と言います) 付けされた変数で，容易に多数のデータの管理が可能です．例えば，A という変数には最大 1 個のデータしか格納できませんが，大きさが 4 と定義された配列 A には A(1)，A(2)，A(3)，A(4) の最大 4 個のデータを扱うことができます．しかも添字 (括弧の中の番号) は整数ですから計算によって添字を求めたりすることもできます．そのことによりデータの格納場所を指定したりするのが非常に容易になります．一つの例として，この大きさ 4 の配列 A と 4 個の変数 B，C，D，E を考えよう．両者とも最大 4 個のデータを取り扱うことができることは同じです．いま，これらの配列や変

数をすべてゼロにしたいとします．配列では，`LET A(I)=0` という式を `I` を 1 から 4 まで変化させながらループ構造で繰り返せばよいだけです．一方，変数では `LET B=0`， `LET C=0`， `LET D=0`， `LET E=0` と同じ様なコードを変数一つにつき一つずつ別々に記述しなければなりません．データ数が例えば 1000 個になると変数ではもはや対応できなくなりますが，配列では，繰り返し回数を 1000 にすればよいだけです．このように配列は多数のデータを取り扱うには極めて有効な手段です．ここでは添字の個数が 1 個の 1 次元配列と 2 個の 2 次元配列を取り扱います．

4.7.1 1 次元配列

1 次元配列は，図 4.13 のように 1 方向に並んだ，番号付きの変数です．それぞれの格納場所は配列の要素とよばれます．この例では，配列 A の 1 番目の要素には 2，2 番目の要素には 4... というように添え字の 2 倍の数が各要素に格納されています． 1 次元の場合の配列の書き方は以下のとおりです．このとき配列名の付け方は変数名と同じです．

構文 4.1 1 次元配列の書き方

配列名 (添字の番号)

配列の要素	A(1)	A(2)	A(3)	A(4)
格納されているデータ	2	4	6	8

図 4.13: 1 次元配列の例

1 次元配列の宣言 1 次元配列の場合の宣言の方法は次のようにします．

構文 4.2 1 次元配列の宣言

`Dim` 配列名 (大きさ)

ここで配列名の付け方は変数のときと同じですが，配列には次元と大きさがあります．大きさとは格納できるデータの個数であり，次元とは添え字の個数です．

例 4.1 大きさ 10 の配列 Idx を宣言

```
Dim Idx(10)
```

この例では Idx(1),Idx(2),...,Idx(9),Idx(10) の 10 個の要素が使えます．

配列への代入　配列へデータを代入するとき，何番目の要素に代入するかが重要です．代入のしかたは次のようになります．

構文 4.3 配列への代入

LET 配列名 (代入したい要素の添え字)=式

次に例を示します．

例 4.2 配列への代入の例

```
LET A(3) = 1       配列 A の 3 番目の要素を 1 にします．
LET B(2) = X + Y   配列 B の 2 番目の要素を X + Y の計算結果の
                   値にします．
LET X(I) = 0       配列 X の I 番目の要素を 0 にします．
```

この例のように添え字に変数を使うことにより，任意の配列要素を指定することができます．すなわち，添え字 I に入れる数を変化させることによって代入する X の場所を変えることができます．

ループと１次元配列　ループとは For 文などを用いて繰り返し処理をすることです．繰り返し処理については第 1 章第 2 節第 2 項 (1.2.2，7 ページ) を参照してください．前節の例 (例 4.2) の最後のように添え字に変数を使うことにより，任意の配列要素を指定することができます．このとき，添

え字の変数にループのカウンタ変数を用いればループが 1 回まわるごとに次の要素を参照することができます．このように，ループと配列は密接な関係があり，これらを自由に扱うことによって，より高度でわかりやすいプログラミングが可能です．以下に例を示します．

例 4.3 大きさ 10 の配列 A のすべての要素をゼロにします．

```
Dim A(10)          配列の宣言（配列名:A　大きさ:10）
For I = 1 To 10    ループ (I を 1 から 10 まで 1 ずつ増やし繰り返す
                   (J=1,2,...,10))
  LET A(I) = 0     配列 A の I 番目の要素をゼロにする
Next I             ループの終端
```

図 4.14: 例 4.3 のコード

例 4.4 大きさ 10 の配列 MtxA の奇数番目の要素を表示．

```
Dim MtxA(10)             配列の宣言（配列名:MtxA　大きさ:10）
For J = 1 To 10 Step 2   ループ J を 1 から 10 まで 2 ずつ変えて
                         繰り返す (J=1,3,...,9)
  Print MtxA(J)          配列 MtxA の J 番目の要素を表示する
Next J                   ループの終端
```

図 4.15: 例 4.4 のコード

例 4.5 配列 MtxB （大きさ:10）の 6 から 10 番目の要素を表示します．ただし，カウンタ変数を 1 から 5 まで変化させます．

カウンタ変数 I は 1 から 5 まで変化するので，配列の 6 から 10 番目の要素を表示するには添え字を I + 5 とすればよろしい．このように配列の括弧の中には式も書けます．

```
Dim MtxB(10)          配列の宣言（配列名:MtxB　大きさ:10）
For I = 1 To 5        ループ I を 1 から 5 まで 1 ずつ変え繰り返す
  Print MrxB(I + 5)   配列 MtxB の I + 5 番目の要素を表示する
Next I                ループの終端
```

図 4.16: 例 4.5 のコード

例 4.6 整数型で大きさ 10 の配列 MtxC に添え字の 2 倍の数値を代入し表示します.

```
Dim MtxC(10)            配列の宣言（配列名:MtxC　大きさ:10）
For I = 1 To 10         For...Next ループ I は 1 から 10 まで
  LET MtxC(I) = 2 * I   配列 MtxC の I 番目の要素を I*2 にする
Next I                  ループの終端
For I = 1 To 10         次のループ
  Print MrxC(I)         配列 MtxC の I 番目の要素を表示する
Next I                  ループの終端
```

図 4.17: 例 4.6 のコード

問題 4.1 大きさ 5 配列 Ary を宣言し，その 3 番目の要素に 1.7320508 を代入し，変数 x に Ary の 2 番目の要素を代入してみよう.

問題 4.2 大きさ 100 の配列 Net を宣言し，その I 番目の要素に 0 を代入し，変数 x に Ary の J 番目の要素を代入してみよう.

問題 4.3 大きさ 100 の配列 A の偶数番目の要素を表示するプログラムを Do...Loop または For...Next を用いて書いてみよう.

問題 4.4 次のように配列 A に代入するプログラムをループ構造を用いて書いてみよう.

1.

配列の要素	A(1)	A(2)	...	A(9)
データ	11	12	...	19

2.

配列の要素	A(1)	A(2)	...	A(10)
データ	9	8	...	0

3.

配列の要素	A(1)	A(2)	A(3)	A(4)	A(5)
データ	2	4	8	16	32

4.7.2 ２次元配列

１次元配列の場合，添字の個数は 1 個であり，データは横方向だけに並んでいます．添字の個数が 1 個で，並ぶ方向が 1 つですから１次元と言います．それに対して，２次元配列では添字の個数は 2 個あります．したがって２次元配列では，図 4.18 のように並ぶ方向が縦と横の 2 方向になり，それぞれ行と列と呼んでいます．行は上から 1 行目，２行目... と並び，列は左から 1 列目，2 列目... とならんでいます．だから配列の要素（位置）を指定するには 2 個の添字すなわち行番号と列番号の両方を決める必要があります．この例の場合，配列 A の 2 行 3 列目には −1 が代入されています．ところで，行と列を表わす添字に変数を用いる場合，例えば行番号を必ず変数 I で列番号を変数 J で表わすというように統一しておけば行と列の混乱を軽減することができます．このとき，変数 I の値が 1 のときは 1 行目を，2 のとき 2 行目を表わし，変数 J の値が 1 のときは 1 列目を 2 のときは 2 列目を表わします．

２次元配列の宣言

２次元配列の宣言のしかたは次の通りです．

構文 4.4 ２次元配列の宣言

```
Dim 配列名 (行の大きさ , 列の大きさ)
```

行番号 I ＼ 列番号 J	1	2	3	4
1	0 (1,1)	–1 (1,2)	–2 (1,3)	–3 (1,4)
2	1 (2,1)	0 (2,2)	–1 (2,3)	–2 (2,4)
3	2 (3,1)	1 (3,2)	0 (3,3)	–1 (3,4)

図 4.18: 2 次元配列の例 括弧内は (行番号, 列番号)

添え字の個数が行と列の 2 つあることを除いて 1 次元配列と同じです. このとき，配列の要素の個数は “行の大きさ × 列の大きさ” となります．図 4.18 の例では $4 \times 3 = 12$ 個の要素があるので，最大 12 個のデータを取り扱うことができます.

例 4.7 大きさ 4×2 の配列 `MatrixA` を宣言します.

```
Dim MartixA(4,2)
```

例 4.8 大きさ 10×10 の配列 `MatrixB` を宣言します.

```
Dim MartixB(10,10)
```

問題 4.5 2 次元配列の宣言と代入

1. 大きさ 5×10 の 2 次元配列 `MatrixA` を宣言してみよう.
2. この配列の 3 行 4 列目に 5.0 を代入し，5 行 3 列目の値を変数 `A` に代入してみよう.
3. 大きさ 100×100 の 2 次元配列 `MatrixB` を宣言してみよう.
4. この配列の 35 行 77 列目に 5.0 を代入し，52 行 33 列目の値を変数 `X` に代入してみよう.

ネストしたループと 2 次元配列

２次元配列では行と列の２つがあるので，それぞれの値を決めないとデータを配列に代入したり参照したりできません．ループを用いて配列の添え字を変化させることは大切なテクニックですが，２次元配列の場合は添え字が 2 個あるのでループも 2 重にすると便利です．例えば，2 重のループの外側を行に，内側を列にしておけば行が 1 のとき列が 1,2,3,... と変化し，次に行が 2 になって列が 1,2,3,... となります．次に例を示します．

例 4.9 図 4.18 の配列 Ary を作る

回数	行（I の値）	列（J の値）	I-J の値
1	1	1	0
2	1	2	-1
3	1	3	-2
4	1	4	-3
5	2	1	1
6	2	2	0
7	2	3	-1
8	2	4	-2
9	3	1	2
10	3	2	1
11	3	3	0
12	3	4	-1

図 4.19: 配列の行と列の計算の順番

```
Dim Ary(3 , 4)                  3行4列の配列Aryの宣言
For I = 1 To 3                  Iは1から3まで1づつ増加
                                Iが変われば行が変わる
  For J = 1 To 4                Jは1から4まで1づつ増加
                                Jが変われば列が変わる
    LET Ary(I , J) = I - J      配列AryのI行J列目にI-Jを代入
  Next J                        Iのループの終端
Next I                          Iのループの終端
```

図 4.20: 図4.18の配列 Ary を作るコード

問題 4.6 9行9列の配列 QQ で九九の表を作り表示するプログラムを書いてみよう.

問題 4.7 4行3列の2次元配列 K に K(1,1)=11, K(2,3)=23 のように 10 の位が行番号, 1 の位が列番号となる数を代入し表示してみよう.

列番号 J / 行番号 I	1	2	3
1	11	12	13
2	21	22	23
3	31	32	33
4	41	42	43

問題 4.8 行列の和と差は各要素の和や差です. A, B, C, D を2行3列の行列とし, $C = A + B$, $D = A - B$ を計算し表示するプログラムを書いてみよう. (例えば $c_{11} = a_{11} + b_{11}$)

問題 4.9 2行3列の行列と3行2列の行列の積を計算するプログラムを書

いてみよう．ただし行列の積は次のように書けます．

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{pmatrix}$$
$$= \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} & a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} \\ a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31} & a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32} \end{pmatrix}$$

参考文献

須田・北原「BASIC による電気電子」共立出版 (1985)