```
    if J<4 then begin J:=J+1; go to
      LOOP 1 end;
OUTPUT: WRITE PRINTER (,, '△△△T:=',
          TT-5.0*KK, TT-4.0*KK,
          TT-3.0* KK, TT-2.0*KK,
          TT-KK);
    J:=0;
    for I:=0 step 1 until M do
      begin
        WRITE PRINTER (I, C[I, o], C[I, 1],
          C[I, 2], C[I, 3], C[I, 4)];
        C[I, 0]:=C[I, 5]
      end;
    ADVANCE PRINTER (0);
LOOP 1: end MAIN END;
          WRITE PRINTER (,,'△△△T:='),
            TT-5.0*KK,
            TT-4.0*KK, TT-3.0*KK,
            TT-2.0*KK, TT-KK);
          for I:=0 step 1 until M do
            begin switch SW:=P 0, P 1, P 2,
                              P 3, P 4;
go to SW[J+1];
  P 0: WRITE PRINTER (I, C[I, 0]);
  P 1: WRITE PRINTER (I, C[I, 0], C[I, 1]);
  P 2: WRITE PRINTER (I, C[I, 0], C[I, 1],
    C[I, 2]);
  P 3: WRITE PRINTER (I, C[I, 0], C[I, 1],
    C[I, 2], C[I, 3]);
  P 4: WRITE PRINTER (I, C[I, 0], C[I, 1],
    C[I, 2], C[I, 3], C[I, 4])
    end
  end
```

注 identifier の説明および数値例については，次のプログラムのあとの記述を参照のこと．

6309. 偏微分方程式の Crank-Nicolson 解法

三浦 大亮（東洋レーヨン）

問題は前に同じ．

Crank-Nicolson の方法

（1）式を次のような形の差分方程式に直すことができる．

$$\frac{\partial C}{\partial t}=\frac{C_{i,j+1}-C_{i,j}}{k} \tag{9}$$

$$\left.\begin{aligned}&K\left(\frac{\partial^2 C}{\partial r^2}+\frac{1}{r}\frac{\partial C}{\partial r}\right)\\&=\frac{K}{2}\left(\frac{C_{i+1,j+1}-2C_{i,j+1}+C_{i-1,j+1}}{h^2}\right.\\&+\frac{1}{ih}\frac{C_{i+1,j+1}-C_{i-1,j+1}}{2h}\\&+\frac{C_{i+1,j}-2C_{i,j}+C_{i-1,j}}{h^2}\\&\left.+\frac{1}{ih}\frac{C_{i+1,j}-C_{i-1,j}}{2h}\right)\end{aligned}\right\} \tag{10}$$

これから（7）に相当するものを導くと

$$\left.\begin{aligned}&\left(1+\frac{1}{2i}\right)C_{i+1,j+1}-2\left(1+\frac{1}{\beta}\right)C_{i,j+1}\\&+\left(1-\frac{1}{2i}\right)C_{i-1,j+1}\\&=-\left(1+\frac{1}{2i}\right)C_{i+1,j}\\&+2\left(1-\frac{1}{\beta}\right)C_{i,j}-\left(1-\frac{1}{2i}\right)C_{i-1,j}\end{aligned}\right\} \tag{11}$$

境界条件，初期条件は（8）と同じである．ただしこの場合には，前進型の場合より，やや精度は低くなるが $\beta\equiv\frac{Kh}{k^2}$ に対する制限はない． また $r=0$ では次の式を用いる．

$$r=0 \text{ で } 4C_{1,j+1}-2\left(2+\frac{1}{\beta}\right)C_{0,j+1}$$
$$=-4C_{1,j}+2\left(2-\frac{1}{\beta}\right)C_{0,j} \tag{12}$$

（11）および（12）には，左辺の未知数として現われるものが，三つまたは，二つであるために，（7）のように初期値から1点づつ計算していくことができないで，$j+1$ の線上の各点の値 $C_{i,j+1}$ ($i=1, 2, \cdots\cdots, m-1$) についての連立方程式を解きながらステップを進める．

（11）と（12）によって表わされる連立方程式はステップごとに $m-1$ 元のものであるが，この係数行列は tridiagonal form となることを利用して解くことができる．

すなわち

$$B_0=-2\left(2+\frac{1}{\beta}\right),\quad C_0=4,$$
$$D_0=-4C_{1,j}+2\left(2-\frac{1}{\beta}\right)C_{0,j}$$
$$A_i=\left(1-\frac{1}{2i}\right),\quad B_i=-2\left(1+\frac{1}{\beta}\right),$$
$$C_i=\left(1+\frac{1}{2i}\right),$$
$$D_i=-\left(1+\frac{1}{2i}\right)C_{i+1,j}$$

$$\left.\begin{aligned}
&+2\left(1-\frac{1}{\beta}\right)C_{i,j}-\left(1-\frac{1}{2i}\right)C_{i-1,j}\\
&(i=1,2,\cdots,m-2)\\
&A_{m-1}=\left(1-\frac{1}{2(m-1)}\right),\\
&B_{m-1}=-2\left(1+\frac{1}{\beta}\right),\\
&D_{m-1}=-\left(1+\frac{1}{2(m-1)}\right)C_{m,j+1}\\
&\quad-\left(1+\frac{1}{2(m-1)}\right)C_{m,j}\\
&\quad+2\left(1-\frac{1}{\beta}\right)C_{m-1,j}\\
&\quad-\left(1-\frac{1}{2(m-1)}\right)C_{m-2,j}
\end{aligned}\right\}(13)$$

これから T_i を未知数として次のように整理すれば

$$\left.\begin{aligned}
&B_0T_0+C_0T_1=D_0\\
&A_iT_{i-1}+B_iT_i+C_iT_{i+1}=D_i\\
&\qquad(i=1,2,\cdots,m-2)\\
&A_{m-1}T_{m-2}+B_{m-1}T_{m-1}=D_{m-1}
\end{aligned}\right\}(14)$$

以下の方法によって，(14) のような形の tridiagonal form の一般の連立方程式を解くことができる．

$$\left.\begin{aligned}
&b_0=C_0/B_0,\ b_i=C_i/(B_i-A_ib_{i-1})\\
&\qquad(i=1,2,\cdots,m-2)\\
&q_0=D_0/B_0,\ q_i=(D_i-A_iq_{i-1})\\
&\qquad/(B_i-A_ib_{i-1})\ (i=1,2,\cdots,m-1)\\
&T_{m-1}=q_{m-1},\ T_i=q_i-b_iT_{i+1}\\
&\qquad(i=m-2,\ m-3,\ \cdots,\ 0)
\end{aligned}\right\}(15)$$

<Crank-Nicolson-Tridiagonal の方法のプログラム>

```
begin real B0, B1, B2, H, KK, IIF, AA, K,
        Co, CA, MF, TT, To, TL. BETA;
   integer I, J, M;
 real array A, C, D[0 : 20], T[0 : 20, 0 : 4];
 procedure TRID;
   comment TO SOLVE THE LINEAR SYSTEM
          WITH TRIDIAGONAL FORM;
   begin real X;
      real array QQ, BB [0 : 20];
     BB[0]:=C[0]/B0;
     QQ[0]:=D[0]/B0;
    for I:=1 step 1 until M-1 do
     begin X:=B1-A[I]*BB(I-1);
       BB[I]:=C[I]/X;
       QQ[I]:=(D[I]-A[I]*QQ[I-1])/X
     end;
    X:=T[M-1, J]:=QQ[M-1];
    for I:=M-2 step-1 until 0 do
      X:=T[I, J]:=QQ[I]-BB[I]*X
       end TRID;
 procedure OUTPUT;
     begin J:=0;
     WRITE PRINTER ('△△△T:=', TT-5.0*KK,
       TT-4.0*KK, TT-3.0*KK,
       TT-2.0*KK, TT-KK);
     for I:=0 step 1 until M do
       WRITE PRINTER (I, T[I, 0],
         T[I, 1], T[I, 2], T[I, 3], T[I, 4]);
     T[M, 0]:=CA; ADVANCE PRINTER (0)
   end;
START: READ CARD (M, To, TL, KK, K,
          AA, Co, CA);
        WRITE PRINTER ('△△△K:=', K,
         '△△△A:=',AA);
        MF:=M; H:=AA/MF; TT:=To;
        BETA:=K*KK/H↑2;
        B0:=-2.0*(2.0+1.0/BETA);
        B1:=-2.0*(1.0+1.0/BETA);
        B2:=2.0*(1.0-1.0/BETA);
        C[0]:=4.0;
 BOUNDARY:
      for I:=1 step 1until 4 do T[M, I]:=CA;
      for I:=0 step 1 until M do T[I, 0]:=Co;
      J:=0;
 INITIALIZATION: comment DEFINED BY
   YOUR PROBLEM;
     for I:=1 step 1 until M do
       begin IIF:=I;
         A[I]:=1.0-0.5/IIF;
         C[I]:=1.0+0.5/IIF
       end;
 COEFD: comment TO CALCULATE D[I],
          DEFINED BY YOUR PROBLEM;
          D[0]:=-4.0*T[1, J]+2.0*
            (2.0-1.0/BETA)*T[0, J];
        for I:=1 step 1 until M-1 do
          D[I]:=-C[I]*T[I+1, J]+B2*
           T[I, J]-A[I]*T[I-1, J];
        D[M-1]:=D[M-1]-C[M-1]*T[M,J];
 MAIN: TT:=TT+KK;
```

i	t=120 min		t=960 min		t=1440 min	
	前進型 k=30 min	C-N-T k=60 min	前進型	C-N-T	前進型	C-N-T
0	8.0000003	7.9999998	7.8683289	7.8630776	7.3596088	7.3791880
1	8.0000004	7.9999995	7.8374430	7.8347786	7.2963721	7.3192677
2	8.0000002	7.9999975	7.7347437	7.7406038	7.1014241	7.1338230
3	8.0000001	7.9999839	7.5266624	7.5489123	6.7585955	6.8050599
4	8.0000001	7.9998916	7.1610557	7.2082782	6.2473454	6.3094709
5	8.0000002	7.9992421	6.5762624	6.6537069	5.5510775	5.6262304
6	8.0000003	7.9945767	5.7188374	5.8227827	4.6667704	4.7476092
7	7.9554955	7.9605929	4.5678132	4.6819807	3.6133376	3.6888168
8	7.4184675	7.7104602	3.1572561	3.2554683	2.4358401	2.4938571
9	4.9895764	5.8547073	1.5851834	1.6412312	1.2033445	1.2343221
10	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

```
    J:=J+1;
    if J=5 then OUTPUT;
    TRID;
    if TT < TLthen go to COEFD;
    for J:=J+1 step 1 until 4 do
    for I:=0 step 1 until M do
      T[I, J]:=0.0;
    OUTPUT
  end
```

(注1) identifier の意味: M::=m(≦20), To::=to, T|TL::=tend, K::=K, A|AA::=a, KK::=k, Co::=Co, CA::=Ca, A[I]::=Ai, B0|B1|B2::=Bi, C[I]::=Ci, D[I]::=Di,C[I, J]|T[I, J]::=Ci,j である.

(注2) output の形: 5ステップづつまとめて i=0~m について縦にプリントする. ADVANCEPRINTER(0) でページを改める. △は space をコンマとコンマの間は5個分の space を示す.

(注3) input data: それぞれの identifier の type にしたがって card に穿孔する.

(注4) I/O の statement は standard procedure として扱う. format declaration はない.

(注5) テストは USSC のために作った TORAY ALGOL で行なった.

<計算例> $K=5.56\times10^{-7}$ cm²/min, a=0.1 cm, Co=8%, Ca=0%, m=10 の場合である（上表）.

6310. 路上駐車のモンテ・カルロ実験

森口 繁一（東大計数工学科），戸田 英雄（電試応用数学研）

1. 問題 ある道路の片側に駐車可能の区間 AB がある（**第1図**）．初めは全体が空いていて，そこへ1台の自動車が来て駐車する場合，A から B までの間に納る限りどこへでも**勝手次第**に駐車する．その位置を PQ とする．2台目は A から P まで，または Q から B までの間に勝手に駐車する．以下同様にして次々と駐車していくと，やがて何台か駐車したとき，もうあとはどこにも駐車する余地がない状態が現われる．このときこの駐車場はつまったと呼ぶことにする．問題は与えられた長さの駐車場は平均何台でつまるか？ ということである．

この問題は，森口[1]がすでに「応用数学夜話②」において考察し，JIS 乱数を用いて駐車の過程のモンテ・カルロ実験の結果（**第2図**）を与えている．（Dvoretzky の講演と A. Rényi の論文の紹介）その後戸田が JUSE ALGOL でプログラムを試みたものである．人間は駐車できるかどうかは**一目でわかる！** が機械は計算しないとわからない．

2. 計算機によるモンテ・カルロ実験 駐車の過程がどのように進行するかを見るため次のようにプログラムした．

START: たとえば X=10.0, PQ=1.0 とする．

INIT: X/PQ の整数部を L とし，P[1], P[2], …, P[L], までの L(=10) カ所を考えて，ここが初めは全部空いていることを示すため，負の数たとえば −1.0 をいれておく．X/PQ の小数部は P[L+1] にいれる．

RNDM: 0 から 1.0 までの一様に分布する乱数を発生させて（JUSE ALGOL には RANDOM という real procedure があって，プログラム中には parameter なしの real 型の function designator として一様乱数を発生する．），たとえば 0.62 が得られたとする．R=0.62×L=6.2 として 6.2 の位置から 7.2 の位置までの間に1台駐車せんとするわけだが，その前にできるかどうかを機械は計算しなければならない．（前を見，後を見て隣りの車や壁とぶつからないかを確める．）

PARK: 駐車可能ならば S=0.2（R の小数部分）を P[7] に入れて，6.2 の位置から 7.2 の位置までの区間がつまったことを示す．同時に駐車数 N を1だけ加える．駐車不可能のときは **RNDM** にもどって次の乱数を発生させて繰返す．以下同様にして第2図