第４部：　『魔方陣の最新研究』　摂田　寛二
第４章：　解説『魔方陣研究の諸技法』II
第11-３節：　五進法で対称五方陣の全数個を構成する

０．今回は，五進法を用いて，補数対称型五方陣の標準解の全数 48544 個を構成する。
　企図においても方法においても，第８章第３節及び第 10 章第３節を継承しているので，もし未読ならば，先に是非お読みいただきたい。というのは，「**オイラー方陣**」を構成するように見えていささか違い，**ラテン方陣**を作ろうというわけでもないからだ。
　今までの研究では，『完全オイラー型』の補数対称五方陣は，たったの 32 個しか作れなかった。主対角線２本から**ラテン条件**（0，1，2，3，4 の５数で作るという）を外して，ただ定和を定義した場合でも，384 個しか作れなかった。対称型は，総じてラテン方陣と相性が極端に悪いという印象で，自分の不勉強を省みず，そうとしか考えなかった。
　対称五方陣に対して，五進法ははたして何の威力も無いのか。汎五方陣や対称汎五方陣ではあんなに見事な成果を上げたというのに，どうしたことであろう。
　ここは少し角度を変えて見なければいけないのではないか。この際，ラテン条件に余りとらわれずに，別の条件を与えて対称五方陣を作ってみてはどうだろうか。
　第８章での四方陣での成功に励まされて，五方陣でもう一踏ん張りしてみよう。
　試行錯誤の全経過を書くのは，煩わしい。ここでは二段階に整理して，私の挑戦を説明しよう。その方が，対称五方陣の「構造」が良くわかるように思うからだ。

１．先ず「**新オイラー法**」を使った最初の挑戦を説明しよう。最初から百パーセント成功した方法では決してないが，良い足がかりを作ってくれたように思うから。
　次の概念図を見てほしい。五進法分解図の高位と低位を統一的に扱い「**単位方陣**」と考える。その同類を別途に構成し，組み合わせて計算・合成するのが，この方法の骨子だ。

**「新オイラー法」の概念図：五進法分解図と構成図 **

（古典的表記と数学的表記：五進法分解図：高位/低位/；サム・チェック付き）

```
          420#\65/65| [Math]        \60/60|  /D5i    H\10/10|          L\10/10|
 1 22 24 15  3|65|65|   0 21 23 14  2|60|60|  0 4 4 2 0|10|10|  0 1 3 4 2|10|10|
20  5 16  7 17|65|65|  19  4 15  6 16|60|60|  3 0 3 1 3|10|10|  4 4 0 1 1|10|10|
12  8 13 18 14|65|65|  11  7 12 17 13|60|60|  2 1 2 3 2|10|10|  1 2 2 2 3|10|10|
 9 19 10 21  6|65|65|   8 18  9 20  5|60|60|  1 3 1 4 1|10|10|  3 3 4 0 0|10|10|
23 11  2  4 25|65|65|  22 10  1  3 24|60|60|  4 2 0 0 4|10|10|  2 0 1 3 4|10|10|
```

（計算：古典#－１＝数学# -> ５で割った整商を高位に，剰余を低位に）

（構成図：単位方陣：高位/低位/　数学的表記#と古典的表記# サム・チェック付き）

```
/EU    1/H\10/10|      339/L\10/10|  [Math]        \60/60|          1#\65/65|
 0 4 4 2 0|10|10|  0 1 3 4 2|10|10|   0 21 23 14  2|60|60|   1 22 24 15  3|65|65|
 3 0 3 1 3|10|10|  4 4 0 1 1|10|10|  19  4 15  6 16|60|60|  20  5 16  7 17|65|65|
 2 1 2 3 2|10|10|  1 2 2 2 3|10|10|  11  7 12 17 13|60|60|  12  8 13 18 14|65|65|
 1 3 1 4 1|10|10|  3 3 4 0 0|10|10|   8 18  9 20  5|60|60|   9 19 10 21  6|65|65|
 4 2 0 0 4|10|10|  2 0 1 3 4|10|10|  22 10  1  3 24|60|60|  23 11  2  4 25|65|65|
```

（計算：単位方陣の高位×５＋低位 -> 数学# ＋１＝古典#）

　単位方陣を構成する時の条件としては，とりあえず次のように仮定することにしよう。
（１）相互に対称の位置にある二数の間に，a+b=4 が成り立つ。つまり，その二数は４の補数を成す。(0,4)，(1,3)，(2,2)，　(3,1)，(4,0)の４ペアのいずれかが，必ず対称の位置に来る。対象となる補数対称方陣の特性を単位方陣も持つと仮定するのだ。
（２）どの行・列でも五数の和が定和の 10 に等しいとする。つまり，i+j+k+l+m=10；対象方陣の特性を単位方陣も持つと仮定する。ただし，ラテン条件にはとらわれない。
（３）どの単位方陣でも，(0,1,2,3,4) の五数がいずれも５度ずつ現れる。過不足無く平等に５度ずつ現れなければならない。

２．特に（１）の条件を仮定すると，主対角線２本と中央の行と列の計４本は，それぞれ内部に相互に対称の位置にある補数を２組ずつ当然持つことになり，常に中央にある n13 にはいつも 2 が来る。そして必ず五数定和を成立させる。

この三条件を盛り込んだプログラムを，次のように書いた。英語版で申し訳ないが，要点のみを示したリストを掲げる。

```
//**  Self-complementary Magic Squares of Order 5: Reconstructed  **
//**  by the 5-th Increment Number System and New Euler's Method  **
//**  File: 'NewEulerMS55SC.c'; Dictated by Kanji Setsuda  **
//**  on Dec.18, 2011, Jan.25, Feb. 9 and Apr. 3, 2012  **
//**  Working with MacOSX 10.7.3 & Xcode 4.2.1  **
//
//** Definitions for Self-complementary Magic Squares of Order 5 **
//.--.--.--.--.--. * Basic Form and Equations: C=10; *
//| 1| 2| 3| 4| 5|  n1+n2+n3+n4+n5=C      ...sm1;
//|--+--+--+--+--|  n6+n7+n8+n9+n10=C     ...sm2;
//| 6| 7| 8| 9|10|  n11+n12+n13+n14+n15=C ...sm3;
//|--+--+--+--+--|  n16+n17+n18+n19+n20=C ...sm4;
//|11|12|13|14|15|  n21+n22+n23+n24+n25=C ...sm5;
//|--+--+--+--+--|  n1+n6+n11+n16+n21=C   ...sm6;
//|16|17|18|19|20|  n2+n7+n12+n17+n22=C   ...sm7;
//|--+--+--+--+--|  n3+n8+n13+n18+n23=C   ...sm8;
//|21|22|23|24|25|  n4+n9+n14+n19+n24=C   ...sm9;
//'--'--'--'--'--'  n5+n10+n15+n20+n25=C  ...sm10;
//  n1+n7+n13+n19+n25=C ...sm11;  n5+n9+n13+n17+n21=C ...sm12;
//* Self-complementary Conditions: SC=4; *
//  n1+n25=n2+n24=n3+n23=n4+n22=n5+n21
// =n6+n20=n7+n19=n8+n18=n9+n17=n10+n16
// =n11+n15=n12+n14=n13+n13=SC
//
#include <stdio.h>
//
short int cnt, cnt2;
short eucnt[8];
short u1,u2;
short SC, LSM;
short ecnt, cnt3;
short nm[30], uflg[26];
short anm[8][76];
char teu[2625][26];
char mtc[2625][2625];
short solt[7137][26];
short cntr[7137][3];
//
void estp01(void), estp02(void), estp03(void), estp04(void), estp05(void);
void estp06(void), estp07(void), estp08(void), estp09(void), estp10(void);
void estp11(void), estp12(void);
void eansrecord(void), preunit(short x);
//
void tblmtc(void);
void cmbcmp(void), solsave(void);
void prans(short x), pr2ans(void), pr1ans(void);
//
void makemod(void);
void defmod(short x, short y);
//
//* Main Program *
int main(void){
 short n;
  printf("\n");
  printf("** Self-complementary Magic Squares of Order 5: **\n");
  printf(" ** Made by our 'New Euler's Method' with /D5i **\n");
  for(n=0;n<5;n++){uflg[n]=0;}
```

```
  nm[13]=2; uflg[2]=1;
  SC=4; LSM=10; cnt=0;
  printf("\n");
  printf(" ** List of Euler Units made with /D5i **\n");
  estp01();                    // Make the Euler Units with /D5i
  if(cnt3>0){preunit(cnt3);}   // Print the Rest One
  ecnt=cnt; printf(" [Count of Euler Units = %d]\n",ecnt);
  tblmtc();
  printf("\n");
  printf("** Standard Solutions of Self-complementary Magic Squares 5x5: made by our\n");
  printf("   'New Euler's Method' with Decompositions by the 5th Increment Number System **\n");
  printf("   (Euler Units: High/Low/  Sol_Number#  Check Sums:\\Diagonals/Rows|Columns|)\n");
  cnt=0; cnt3=0;
  cmbcmp();                    // Combine and Compose
  prans(500);                  // Print the Solutions with that Interval
  if(cnt3==1){pr1ans();}       // Print the Rest One
  printf(" [Solution Counts = %d]\n",cnt);
  printf(" [OK!]\n");
  printf("** Calculated and listed by Kanji Setsuda\n");
  printf("    on Apr. 3, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **\n");
  printf("\n");
 return 0;
}
//
// Make the Euler Units by /D5i
// Level #1:
// Set N1 & n25
void estp01(){
 short i,j;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[1]=i; nm[25]=j;
      uflg[i]++; uflg[j]++;
      estp02();
      uflg[i]--; uflg[j]--;}
  }
}
// Set N5 & n21
void estp02(){
 short i,j;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){cnt2=0;
      nm[5]=i; nm[21]=j;
      uflg[i]++; uflg[j]++;
      estp03();
      uflg[i]--; uflg[j]--;}
  }
}
// Set N7 & n19
void estp03(){
 short i,j,sm;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){if(nm[1]==0){cnt2=0;}
      nm[7]=i; nm[19]=j;
      uflg[i]++; uflg[j]++;
      sm=nm[1]+nm[7]+nm[13]+nm[19]+nm[25];
      if(sm==LSM){estp04();}
      uflg[i]--; uflg[j]--;}
  }
}
// Set N9 & n17
void estp04(){
 short i,j,sm;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
```

```
        nm[9]=i; nm[17]=j;
        uflg[i]++; uflg[j]++;
        sm=nm[5]+nm[9]+nm[13]+nm[17]+nm[21];
        if(sm==LSM){estp05();}
        uflg[i]--; uflg[j]--;}
   }
}
// Level #2:
// Set N2 & n24
void estp05(){
 short i,j;
   for(i=4;i>=0;i--){j=SC-i;
     if((uflg[i]<5)&&(uflg[j]<5)){
        nm[2]=i; nm[24]=j;
        uflg[i]++; uflg[j]++;
        estp06();
        uflg[i]--; uflg[j]--;}
   }
}
// Set N4 & n22
void estp06(){
 short i,j;
   for(i=0;i<5;i++){j=SC-i;
     if((uflg[i]<5)&&(uflg[j]<5)){
        nm[4]=i; nm[22]=j;
        uflg[i]++; uflg[j]++;
        estp07();
        uflg[i]--; uflg[j]--;}
   }
}
// Set N3 & n23
void estp07(){
 short i,j,sm1,sm2;
   for(i=0;i<5;i++){j=SC-i;
     if((uflg[i]<5)&&(uflg[j]<5)){
        nm[3]=i; nm[23]=j;
        uflg[i]++; uflg[j]++;
        sm1=nm[1]+nm[2]+nm[3]+nm[4]+nm[5];
        sm2=nm[25]+nm[24]+nm[23]+nm[22]+nm[21];
        if((sm1==LSM)&&(sm2==LSM)){estp08();}
        uflg[i]--; uflg[j]--;}
   }
}
// Set N8 & n18
void estp08(){
 short i,j,sm;
   for(i=0;i<5;i++){j=SC-i;
     if((uflg[i]<5)&&(uflg[j]<5)){
        nm[8]=i; nm[18]=j;
        uflg[i]++; uflg[j]++;
        sm=nm[3]+nm[8]+nm[13]+nm[18]+nm[23];
        if(sm==LSM){estp09();}
        uflg[i]--; uflg[j]--;}
   }
}
// Set N12 & n14
void estp09(){
 short i,j,sm1,sm2;
   for(i=0;i<5;i++){j=SC-i;
     if((uflg[i]<5)&&(uflg[j]<5)){
        nm[12]=i; nm[14]=j;
        uflg[i]++; uflg[j]++;
        sm1=nm[2]+nm[7]+nm[12]+nm[17]+nm[22];
        sm2=nm[24]+nm[19]+nm[14]+nm[9]+nm[4];
        if((sm1==LSM)&&(sm2==LSM)){estp10();}
```

```
      uflg[i]--; uflg[j]--;}
  }
}
// Set N6 & n20
void estp10(){
 short i,j;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[6]=i; nm[20]=j;
      uflg[i]++; uflg[j]++;
      estp11();
      uflg[i]--; uflg[j]--;}
  }
}
// Set N10 & n16
void estp11(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[10]=i; nm[16]=j;
      uflg[i]++; uflg[j]++;
      sm1=nm[6]+nm[7]+nm[8]+nm[9]+nm[10];
      sm2=nm[20]+nm[19]+nm[18]+nm[17]+nm[16];
      if((sm1==LSM)&&(sm2==LSM)){estp12();}
      uflg[i]--; uflg[j]--;}
  }
}
// Set N11 & n15
void estp12(){
 short i,j,sm1,sm2,sm3;
  for(i=0;i<5;i++){j=SC-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[11]=i; nm[15]=j;
      uflg[i]++; uflg[j]++;
      sm1=nm[1]+nm[6]+nm[11]+nm[16]+nm[21];
      sm2=nm[25]+nm[20]+nm[15]+nm[10]+nm[5];
      sm3=nm[11]+nm[12]+nm[13]+nm[14]+nm[15];
      if((sm1==LSM)&&(sm2==LSM)&&(sm3==LSM)){eansrecord();}
      uflg[i]--; uflg[j]--;}
  }
}
//
// Record Low Units
void eansrecord(){
 short n;
  teu[cnt][0]=cnt+1;
  for(n=1;n<26;n++){teu[cnt][n]=nm[n];}
  cnt++; cnt2++;
  if(cnt2==1){anm[cnt3][0]=cnt;
    for(n=1;n<26;n++){anm[cnt3][n]=nm[n];}
    cnt3++; if(cnt3==7){preunit(cnt3); cnt3=0;}
  }
}
//
// Print Euler Units
void preunit(short x){
 short l,m,m5,n;
  printf("//");
  for(l=0;l<x;l++){printf("  %9d/",anm[l][0]);}
  printf("\n");
  for(m=0;m<5;m++){m5=m*5;
    printf("//");
    for(l=0;l<x;l++){printf("  ");
      for(n=1;n<6;n++){printf("%2d",anm[l][m5+n]);}}
    printf("\n");
```

```
  }
}
//
// Check how well they match and Save the Data to the Table
void tblmtc(){
 short h,l;
 short n,d1,d2;
  for(h=0;h<ecnt;h++){
    for(l=0;l<ecnt;l++){
      d1=0; for(n=1;n<26;n++){if(teu[h][n]==teu[l][n]){d1++;}}
      d2=0; for(n=1;n<26;n++){if(teu[h][n]+teu[l][n]==SC){d2++;}}
      if(d1<d2){d1=d2;}
      mtc[h][l]=d1;}
  }
}
//
//* Combine and Compose *
void cmbcmp(){
 short n,d,fc;
  for(u1=0;u1<(ecnt/2);u1++){
    for(u2=0;u2<ecnt;u2++){
      if(mtc[u1][u2]==5){
        for(n=0;n<26;n++){uflg[n]=0;}
        for(n=1;n<26;n++){
          d=teu[u1][n]*5+teu[u2][n]+1;
          nm[n]=d; uflg[d]++;
        }
        fc=0;
        for(n=1;n<26;n++){if(uflg[n]==0){fc++;}}
        if(fc==0){
          if((nm[1]<nm[5])&&(nm[5]<nm[21])&&(nm[1]<nm[25])){solsave();}}
      }
    }
  }
}
//
// Save Answers to the Solution List
void solsave(){
 short n;
  for(n=1;n<26;n++){solt[cnt][n]=nm[n];}
  cntr[cnt][0]=cnt+1; cntr[cnt][1]=u1;  cntr[cnt][2]=u2;
  cnt++;
}
//
// Print the Solution List
void prans(short x){
 short s;
 short m,n,sm1,sm2;
  for(s=0;s<cnt;s=s+x){
    eucnt[cnt3*3]=s+1;
    for(n=1;n<26;n++){nm[n]=solt[s][n]; anm[cnt3][n]=nm[n]; anm[cnt3+1][n]=0;}
    for(m=0;m<5;m++){sm1=0; sm2=0;
      for(n=0;n<5;n++){sm1=sm1+nm[m*5+n+1]; sm2=sm2+nm[n*5+m+1];}
      anm[cnt3][m+26]=sm1; anm[cnt3][m+31]=sm2;}
    anm[cnt3][36]=nm[1]+nm[7]+nm[13]+nm[19]+nm[25];
    anm[cnt3][37]=nm[5]+nm[9]+nm[13]+nm[17]+nm[21];
    eucnt[cnt3*3+1]=cntr[s][1]+1; eucnt[cnt3*3+2]=cntr[s][2]+1;
    cnt3++;
    if(cnt3==2){pr2ans(); cnt3=0;}
  }
}
//
// Print 2 Answers with Sum Checks
void pr2ans(){
 short lu1,lu2;
```

```
 short l,l5,m,n;
  printf("%5d/H%5d/L%15d# \\%2d/%2d|  %5d/H%5d/L%15d# \\%2d/%2d|\n",
        eucnt[1],eucnt[2],eucnt[0],anm[0][36],anm[0][37],
        eucnt[4],eucnt[5],eucnt[3],anm[1][36],anm[1][37]);
  for(l=0;l<5;l++){l5=l*5;
    for(m=0;m<2;m++){
      lu1=eucnt[m*3+1]-1; lu2=eucnt[m*3+2]-1;
      printf(" ");
      for(n=0;n<5;n++){printf("%d",teu[lu1][l5+n]);}
      printf("  ");
      for(n=0;n<5;n++){printf("%d",teu[lu2][l5+n]);}
      printf("  ");
      for(n=1;n<6;n++){printf("%3d",anm[m][l5+n]);}
      printf(" |%2d|%2d|",anm[m][l+26],anm[m][l+31]);
      if(m<1){printf("  ");}
    }
    printf("\n");
  }
}
//
// Print the Rest One
void pr1ans(){
 short lu1,lu2;
 short l,l5,n;
  lu1=eucnt[1]-1; lu2=eucnt[2]-1;
  printf("%5d/H%5d/L%15d# \\%2d/%2d|\n",
       lu1+1,lu2+1,eucnt[0],anm[0][36],anm[0][37]);
  for(l=0;l<5;l++){l5=l*5;
    printf(" ");
    for(n=0;n<5;n++){printf("%d",teu[lu1][l5+n]);}
    printf("  ");
    for(n=0;n<5;n++){printf("%d",teu[lu2][l5+n]);}
    printf("  ");
    for(n=1;n<6;n++){printf("%3d",anm[0][l5+n]);}
    printf(" |%2d|%2d|\n",anm[0][l+26],anm[0][l+31]);
  }
}
//
```

上のＣプログラムには，三つの仮定の下に構成した単位方陣の出力，相性表の作成，組み合わせて計算・合成する過程，最終的には構成解の出力までの全部が含まれている。ぜひ根気良く追跡してみていただきたい。

** 五進法による新オイラー法で作った「単位方陣」のリスト（抜粋） **

```
        1/        12/        45/        69/       102/       113/       143/
0 4 4 2 0  0 4 2 4 0  0 3 4 3 0  0 3 3 4 0  0 3 3 4 0  0 4 2 3 1  0 4 2 3 1
3 0 3 1 3  3 1 3 0 3  2 2 3 0 3  2 3 2 0 3  1 4 2 1 2  4 0 3 0 3  4 1 3 0 2
2 1 2 3 2  2 1 2 3 2  3 0 2 4 1  3 0 2 4 1  3 0 2 4 1  2 1 2 3 2  1 0 2 4 3
1 3 1 4 1  1 4 1 3 1  1 4 1 2 2  1 4 2 1 2  2 3 2 0 3  1 4 1 4 0  2 4 1 3 0
4 2 0 0 4  4 0 2 0 4  4 1 0 1 4  4 0 1 1 4  4 0 1 1 4  3 1 2 0 4  3 1 2 0 4

      163/       189/       209/       239/       257/       289/       301/
0 4 1 4 1  0 3 2 4 1  0 2 3 4 1  0 3 3 2 2  0 4 1 3 2  0 4 1 3 2  0 3 1 4 2
3 2 2 0 3  4 3 1 0 2  3 4 2 0 1  3 0 4 0 3  4 1 2 0 3  4 2 0 1 3  4 3 2 0 1
3 0 2 4 1  1 0 2 4 3  1 0 2 4 3  4 1 2 3 0  3 0 2 4 1  3 0 2 4 1  1 0 2 4 3
1 4 2 2 1  2 4 3 1 0  3 4 2 0 1  1 4 0 4 1  1 4 2 3 0  1 3 4 2 0  3 4 2 1 0
3 0 3 0 4  3 0 2 1 4  3 0 1 2 4  2 2 1 1 4  2 1 3 0 4  2 1 3 0 4  2 0 3 1 4

      333/       351/       381/       401/       427/       447/       477/
0 1 4 3 2  0 3 2 2 3  0 3 0 4 3  0 3 0 4 3  0 2 2 3 3  0 2 1 4 3  0 3 2 1 4
3 4 1 0 2  4 0 3 0 3  4 1 3 0 2  3 2 3 0 2  4 3 3 0 0  3 4 2 0 1  4 0 3 1 2
3 0 2 4 1  4 1 2 3 0  3 2 2 2 1  4 1 2 3 0  1 0 2 4 3  3 0 2 4 1  4 1 2 3 0
2 4 3 0 1  1 4 1 4 0  2 4 1 3 0  2 4 1 2 1  4 4 1 1 0  3 4 2 0 1  2 3 1 4 0
2 1 0 3 4  1 2 2 1 4  1 0 4 1 4  1 0 4 1 4  1 1 2 2 4  1 0 3 2 4  0 3 2 1 4
```

```
      488/        521/        545/        578/        589/        715/        775/
0 2 1 3 4   0 2 1 3 4   0 2 1 3 4   0 2 1 3 4   1 4 2 3 0   1 4 2 2 1   1 4 1 2 2
4 1 4 0 1   3 2 4 0 1   4 3 2 0 1   3 4 2 1 0   3 0 3 0 4   3 0 3 0 4   3 0 4 0 3
3 2 2 2 1   4 1 2 3 0   3 0 2 4 1   3 0 2 4 1   2 1 2 3 2   3 0 2 4 1   3 0 2 4 1
3 4 0 3 0   3 4 0 2 1   3 4 2 1 0   4 3 2 0 1   0 4 1 4 1   0 4 1 4 1   1 4 0 4 1
0 1 3 2 4   0 1 3 2 4   0 1 3 2 4   0 1 3 2 4   4 1 2 0 3   3 2 2 0 3   2 2 3 0 3

      895/        955/       1081/       1193/       1313/       1433/       1545/
1 3 1 2 3   1 2 2 1 4   2 3 3 2 0   2 4 1 2 1   2 3 1 1 3   2 2 1 1 4   3 3 2 2 0
4 0 2 0 4   4 0 3 0 3   3 0 4 0 3   3 0 4 0 3   4 0 2 0 4   3 0 4 0 3   3 0 3 0 4
4 1 2 3 0   4 1 2 3 0   0 1 2 3 4   1 0 2 4 3   3 0 2 4 1   4 1 2 3 0   0 1 2 3 4
0 4 2 4 0   1 4 1 4 0   1 4 0 4 1   1 4 0 4 1   0 4 2 4 0   1 4 0 4 1   0 4 1 4 1
1 2 3 1 3   0 3 2 2 3   4 2 1 1 2   3 2 3 0 2   1 3 3 1 2   0 3 3 2 2   4 2 2 1 1

      1671/       1731/       1851/       1911/       2037/       2149/       2275/
3 3 1 2 1   3 3 1 1 2   3 2 2 0 3   3 2 0 1 4   4 3 2 1 0   4 2 2 1 1   4 2 1 1 2
4 0 2 0 4   4 0 2 0 4   3 0 3 0 4   4 0 3 0 3   2 0 3 1 4   3 0 3 0 4   3 0 4 0 3
0 1 2 3 4   1 0 2 4 3   3 0 2 4 1   2 1 2 3 2   0 1 2 3 4   0 1 2 3 4   0 1 2 3 4
0 4 2 4 0   0 4 2 4 0   0 4 1 4 1   1 4 1 4 0   0 3 1 4 2   0 4 1 4 1   1 4 0 4 1
3 2 3 1 1   2 3 3 1 1   1 4 2 2 1   0 3 4 2 1   4 3 2 1 0   3 3 2 2 0   2 3 3 2 0

      2387/       2513/
4 2 0 1 3   4 2 0 0 4
3 0 3 0 4   3 0 3 1 3
2 1 2 3 2   2 1 2 3 2
0 4 1 4 1   1 3 1 4 1
1 3 4 2 0   0 4 4 2 0     [Count of Euler Units = 2624]
```

3．単位方陣が何と 2624 個も出来た。この中から任意の２個を選んで組み合わせる場合の数は，688 万個を超える。標準解を考慮して高位を半分に減らしたとしても，344 万個の場合を調査・検定しなければならない。最新式コンピューターにやらせるから良いようなものの，手計算でやっていたら大変だ。考えただけでも，悪夢のような作業だ。…
　とにもかくにも下表のような結果を得ることができたことを報告する。

**** 五進法による新オイラー法で作った補数対称型五方陣の標準解のリスト（抜粋）****

（使用した単位方陣: 高位/H 低位/L　解の番号#　サム・チェック:\対角線/行|列|）

```
  1/H  339/L              1# \65/65|    7/H  233/L             51# \65/65|
04420  01342    1 22 24 15  3 |65|65| 03430  04231    1 20 23 19  2 |65|65|
30313  44011   20  5 16  7 17 |65|65| 20323  14140   12  5 17 15 16 |65|65|
21232  12223   12  8 13 18 14 |65|65| 34201  21232   18 22 13  4  8 |65|65|
13141  33400    9 19 10 21  6 |65|65| 12142  40303   10 11  9 21 14 |65|65|
42004  20134   23 11  2  4 25 |65|65| 41014  31204   24  7  3  6 25 |65|65|

 10/H 1935/L             101# \65/65|   14/H 1433/L             151# \65/65|
04330  32104    4 23 17 16  5 |65|65| 03340  22114    3 18 17 22  5 |65|65|
20233  10432   12  1 15 19 18 |65|65| 41203  30403   24  6 15  1 19 |65|65|
34201  43210   20 24 13  2  6 |65|65| 12223  41230   10 12 13 14 16 |65|65|
11242  21043    8  7 11 25 14 |65|65| 14230  14041    7 25 11 20  2 |65|65|
41104  04321   21 10  9  3 22 |65|65| 40114  03322   21  4  9  8 23 |65|65|

 19/H  580/L             201# \65/65|   23/H  926/L             251# \65/65|
04420  01234    1 22 23 14  5 |65|65| 04240  12403    2 23 15 21  4 |65|65|
31312  44110   20 10 17  7 11 |65|65| 41113  42130   25  8  7  9 16 |65|65|
10243  21232    8  2 13 24 18 |65|65| 12223  01234    6 12 13 14 20 |65|65|
23131  43300   15 19  9 16  6 |65|65| 13330  41320   10 17 19 18  1 |65|65|
42004  01234   21 12  3  4 25 |65|65| 40204  14023   22  5 11  3 24 |65|65|

 26/H 1469/L             301# \65/65|   31/H  962/L             351# \65/65|
02440  20314    3 11 24 22  5 |65|65| 04330  12034    2 23 16 19  5 |65|65|
21313  31420   14  7 20  8 16 |65|65| 41023  40312   25  6  4 12 18 |65|65|
34201  14203   17 25 13  1  9 |65|65| 12223  34201    9 15 13 11 17 |65|65|
13132  42031   10 18  6 19 12 |65|65| 12430  23140    8 14 22 20  1 |65|65|
40024  03142   21  4  2 15 23 |65|65| 41104  01423   21  7 10  3 24 |65|65|
```

```
  42/H  262/L            401# \65/65|    45/H  584/L            451# \65/65|
04420 01342   1 22 24 15  3 |65|65| 03430 02134   1 18 22 19  5 |65|65|
11341 41203  10  7 18 21  9 |65|65| 22303 34021  14 15 16  3 17 |65|65|
23212 34201  14 20 13  6 12 |65|65| 30241 41230  20  2 13 24  6 |65|65|
30133 14230  17  5  8 19 16 |65|65| 14122 32401   9 23 10 11 12 |65|65|
42004 20134  23 11  2  4 25 |65|65| 41014 01324  21  7  4  8 25 |65|65|

  53/H  474/L            501# \65/65|    87/H  513/L            751# \65/65|
03340 03223   1 19 18 23  4 |65|65| 04420 02314   1 23 24 12  5 |65|65|
32014 14140  17 15  2 10 21 |65|65| 23131 41032  15 17  6 19  8 |65|65|
32221 41230  20 12 13 14  6 |65|65| 10243 43210  10  4 13 22 16 |65|65|
03421 40303   5 16 24 11  9 |65|65| 31312 21430  18  7 20  9 11 |65|65|
40114 12214  22  3  8  7 25 |65|65| 42004 03124  21 14  2  3 25 |65|65|

 107/H  509/L           1001# \65/65|   122/H 1777/L           1251# \65/65|
04240 03124   1 24 12 23  5 |65|65| 04141 34102   4 25  7 21  8 |65|65|
14131 41032  10 22  6 19  8 |65|65| 40213 21034  23  2 11  9 20 |65|65|
21232 43210  15  9 13 17 11 |65|65| 23212 30241  14 16 13 10 12 |65|65|
31303 21430  18  7 20  4 16 |65|65| 13240 01432   6 17 15 24  3 |65|65|
40204 02314  21  3 14  2 25 |65|65| 30304 24301  18  5 19  1 22 |65|65|

 136/H 1213/L           1501# \65/65|   144/H 1136/L           1751# \65/65|
04321 23401   3 24 20 11  7 |65|65| 04141 21340   3 22  9 25  6 |65|65|
40231 10324  22  1 14 18 10 |65|65| 41203 02413  21  8 15  2 19 |65|65|
03214 41230   5 17 13  9 21 |65|65| 21232 34201  14 10 13 16 12 |65|65|
31240 02143  16  8 12 25  4 |65|65| 14230 13024   7 24 11 18  5 |65|65|
32104 34012  19 15  6  2 23 |65|65| 30304 40132  20  1 17  4 23 |65|65|

 154/H 1354/L           2001# \65/65|   163/H 1535/L           2251# \65/65|
04141 20413   3 21 10 22  9 |65|65| 04141 20044   3 21  6 25 10 |65|65|
41023 41104  25  7  2 11 20 |65|65| 32203 34111  19 15 12  2 17 |65|65|
23212 32221  14 18 13  8 12 |65|65| 30241 23212  18  4 13 22  8 |65|65|
12430 04330   6 15 24 19  1 |65|65| 14221 33301   9 24 14 11  7 |65|65|
30304 13042  17  4 16  5 23 |65|65| 30304 00442  16  1 20  5 23 |65|65|

 178/H  863/L           2501# \65/65|   190/H 2183/L           2751# \65/65|
04321 10432   2 21 20 14  8 |65|65| 02431 40321   5 11 24 18  7 |65|65|
02431 24103   3 15 22 16  9 |65|65| 43102 21403  23 17 10  1 14 |65|65|
41230 41230  25  7 13 19  1 |65|65| 10243 03214   6  4 13 22 20 |65|65|
31024 14302  17 10  4 11 23 |65|65| 24310 14032  12 25 16  9  3 |65|65|
32104 21043  18 12  6  5 24 |65|65| 31024 32140  19  8  2 15 21 |65|65|

 201/H 1228/L           3001# \65/65|   221/H  543/L           3501# \65/65|
04321 23401   3 24 20 11  7 |65|65| 02431 03124   1 14 22 18 10 |65|65|
13042 20341   8 16  4 25 12 |65|65| 34120 32041  19 23  6 15  2 |65|65|
41230 03214  21  9 13 17  5 |65|65| 01234 43210   5  9 13 17 21 |65|65|
20413 30142  14  1 22 10 18 |65|65| 42301 30421  24 11 20  3  7 |65|65|
32104 34012  19 15  6  2 23 |65|65| 31024 02314  16  8  4 12 25 |65|65|

 254/H  147/L           4001# \65/65|   294/H  135/L           4501# \65/65|
03322 03241   1 19 18 15 12 |65|65| 01342 04321   1 10 19 23 12 |65|65|
30430 41104  20  2 22 16  5 |65|65| 32410 40033  20 11 21  9  4 |65|65|
14203 32221   9 23 13  3 17 |65|65| 14203 23212   8 24 13  2 18 |65|65|
41041 04330  21 10  4 24  6 |65|65| 43021 11440  22 17  5 15  6 |65|65|
22114 30214  14 11  8  7 25 |65|65| 20134 32104  14  3  7 16 25 |65|65|

 332/H 1585/L           5001# \65/65|   753/H  266/L           5501# \65/65|
02332 33040   4 14 16 20 11 |65|65| 14041 03322   6 24  4 23  8 |65|65|
33040 21124  18 17  2 23  5 |65|65| 03322 41014   5 17 16 12 15 |65|65|
14203 14203   7 25 13  1 19 |65|65| 41230 41230  25  7 13 19  1 |65|65|
40411 02332  21  3 24  9  8 |65|65| 22114 03430  11 14 10  9 21 |65|65|
21124 40411  15  6 10 12 22 |65|65| 30403 22114  18  3 22  2 20 |65|65|
(使用した単位方陣: 高位/H 低位/L   解の番号#  サム・チェック:\対角線/行|列|)
```

```
 797/H  165/L           6001# \65/65|  837/H 1217/L           6501# \65/65|
14302  03241    6 24 18  5 12 |65|65| 14032  24301    8 25  4 16 12 |65|65|
40033  42301   25  3  4 16 17 |65|65| 02341  20431    3 11 20 24  7 |65|65|
21232  01234   11  7 13 19 15 |65|65| 43210  01234   21 17 13  9  5 |65|65|
11440  34120    9 10 22 23  1 |65|65| 30124  31042   19  2  6 15 23 |65|65|
24103  30214   14 21  8  2 20 |65|65| 21403  34102   14 10 22  1 18 |65|65|
 877/H 2243/L           7001# \65/65|
10432  42031   10  3 21 19 12 |65|65|
04321  03142    1 24 17 15  8 |65|65|
43210  14203   22 20 13  6  4 |65|65|
32104  20314   18 11  9  2 25 |65|65|
21043  31420   14  7  5 23 16 |65|65|   [Solution Counts = 7136]  [OK!]
** Calculated and listed by Kanji Setsuda
    on Apr. 3, 2012 with MacOSX 10.7.2 & Xcode 4.2.1 **
```

構成解がだいぶ増えたが，それでも標準解の総数 48544 個には程遠い。
先の三つの仮定には，まだ何か重大な見落としがあると推理される。**ラテン条件**を放棄したくらいでは足りなかった，と考えざるを得ない。

4．ここは，やはり原点に戻るしか無い。通常の方法で作った標準解の１つ１つを五進法分解して，細かく観察・考究しなくてはならない。次表に具体的な十数例を掲げる。
これらの五進法分解図から何がわかるであろうか。行列をざっと見渡しただけでも，和が 10 になっていないものが幾つも見つかる。かなりの頻度で出現している。

**** 通常の方法で作った対称五方陣（抜粋）：五進法分解図 サム・チェック付き ****

```
            1#\65/65| [Math]        \60/60|  /D5i    H\10/10|         L\10/10|
 1 22 21 19  2|65|65|   0 21 20 18  1|60|60|  0 4 4 3 0|11|10|  0 1 0 3 1| 5|10|
20  3 15  9 18|65|65|  19  2 14  8 17|60|60|  3 0 2 1 3| 9|11|  4 2 4 3 2|15| 5|
12 16 13 10 14|65|65|  11 15 12  9 13|60|60|  2 3 2 1 2|10|10|  1 0 2 4 3|10|10|
 8 17 11 23  6|65|65|   7 16 10 22  5|60|60|  1 3 2 4 1|11| 9|  2 1 0 2 0| 5|15|
24  7  5  4 25|65|65|  23  6  4  3 24|60|60|  4 1 0 0 4| 9|10|  3 1 4 3 4|15|10|
            201#\65/65| [Math]        \60/60|  /D5i    H\10/10|         L\10/10|
 1 21 22 19  2|65|65|   0 20 21 18  1|60|60|  0 4 4 3 0|11|10|  0 0 1 3 1| 5|10|
17 14 10  6 18|65|65|  16 13  9  5 17|60|60|  3 2 1 1 3|10|10|  1 3 4 0 2|10|10|
15  3 13 23 11|65|65|  14  2 12 22 10|60|60|  2 0 2 4 2|10|10|  4 2 2 2 0|10|10|
 8 20 16 12  9|65|65|   7 19 15 11  8|60|60|  1 3 3 2 1|10|10|  2 4 0 1 3|10|10|
24  7  4  5 25|65|65|  23  6  3  4 24|60|60|  4 1 0 0 4| 9|10|  3 1 3 4 4|15|10|
            401#\65/65| [Math]        \60/60|  /D5i    H\10/10|         L\10/10|
 1 24 19 18  3|65|65|   0 23 18 17  2|60|60|  0 4 3 3 0|10|10|  0 3 3 2 2|10|10|
20  4  9 11 21|65|65|  19  3  8 10 20|60|60|  3 0 1 2 4|10| 9|  4 3 3 0 0|10|15|
16 14 13 12 10|65|65|  15 13 12 11  9|60|60|  3 2 2 2 1|10|10|  0 3 2 1 4|10|10|
 5 15 17 22  6|65|65|   4 14 16 21  5|60|60|  0 2 3 4 1|10|11|  4 4 1 1 0|10| 5|
23  8  7  2 25|65|65|  22  7  6  1 24|60|60|  4 1 1 0 4|10|10|  2 2 1 1 4|10|10|
            601#\65/65| [Math]        \60/60|  /D5i    H\10/10|         L\10/10|
 1 24 15 22  3|65|65|   0 23 14 21  2|60|60|  0 4 2 4 0|10|10|  0 3 4 1 2|10|10|
19  8  5 17 16|65|65|  18  7  4 16 15|60|60|  3 1 0 3 3|10| 9|  3 2 4 1 0|10|15|
12 20 13  6 14|65|65|  11 19 12  5 13|60|60|  2 3 2 1 2|10|10|  1 4 2 0 3|10|10|
10  9 21 18  7|65|65|   9  8 20 17  6|60|60|  1 1 4 3 1|10|11|  4 3 0 2 1|10| 5|
23  4 11  2 25|65|65|  22  3 10  1 24|60|60|  4 0 2 0 4|10|10|  2 3 0 1 4|10|10|
            801#\65/65| [Math]        \60/60|  /D5i    H\10/10|         L\10/10|
 1 22 21 18  3|65|65|   0 21 20 17  2|60|60|  0 4 4 3 0|11|10|  0 1 0 2 2| 5|10|
10 12 15  9 19|65|65|   9 11 14  8 18|60|60|  1 2 2 1 3| 9|11|  4 1 4 3 3|15| 5|
24  6 13 20  2|65|65|  23  5 12 19  1|60|60|  4 1 2 3 0|10|10|  3 0 2 4 1|10|10|
 7 17 11 14 16|65|65|   6 16 10 13 15|60|60|  1 3 2 2 3|11| 9|  1 1 0 3 0| 5|15|
23  8  5  4 25|65|65|  22  7  4  3 24|60|60|  4 1 0 0 4| 9|10|  2 2 4 3 4|15|10|
```

```
          1001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 20 19 22  3|65|65|   0 19 18 21  2|60|60|  0 3 3 4 0|10|10|  0 4 3 1 2|10|10|
 5 17 11 18 14|65|65|   4 16 10 17 13|60|60|  0 3 2 3 2|10|10|  4 1 0 2 3|10|10|
24 16 13 10  2|65|65|  23 15 12  9  1|60|60|  4 3 2 1 0|10|10|  3 0 2 4 1|10|10|
12  8 15  9 21|65|65|  11  7 14  8 20|60|60|  2 1 2 1 4|10|10|  1 2 4 3 0|10|10|
23  4  7  6 25|65|65|  22  3  6  5 24|60|60|  4 0 1 1 4|10|10|  2 3 1 0 4|10|10|
          2001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 12 24 23  5|65|65|   0 11 23 22  4|60|60|  0 2 4 4 0|10|10|  0 1 3 2 4|10|10|
15  9 16  7 18|65|65|  14  8 15  6 17|60|60|  2 1 3 1 3|10|10|  4 3 0 1 2|10|10|
20 22 13  4  6|65|65|  19 21 12  3  5|60|60|  3 4 2 0 1|10|10|  4 1 2 3 0|10|10|
 8 19 10 17 11|65|65|   7 18  9 16 10|60|60|  1 3 1 3 2|10|10|  2 3 4 1 0|10|10|
21  3  2 14 25|65|65|  20  2  1 13 24|60|60|  4 0 0 2 4|10|10|  0 2 1 3 4|10|10|
          3001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 23 16 19  6|65|65|   0 22 15 18  5|60|60|  0 4 3 3 1|11|10|  0 2 0 3 0| 5|10|
14 18  4 24  5|65|65|  13 17  3 23  4|60|60|  2 3 0 4 0| 9|10|  3 2 3 3 4|15|10|
 9 15 13 11 17|65|65|   8 14 12 10 16|60|60|  1 2 2 2 3|10|10|  3 4 2 0 1|10|10|
21  2 22  8 12|65|65|  20  1 21  7 11|60|60|  4 0 4 1 2|11|10|  0 1 1 2 1| 5|10|
20  7 10  3 25|65|65|  19  6  9  2 24|60|60|  3 1 1 0 4| 9|10|  4 1 4 2 4|15|10|
          4001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 24 12 20  8|65|65|   0 23 11 19  7|60|60|  0 4 2 3 1|10|10|  0 3 1 4 2|10|10|
19  3 22  5 16|65|65|  18  2 21  4 15|60|60|  3 0 4 0 3|10|11|  3 2 1 4 0|10| 5|
17 11 13 15  9|65|65|  16 10 12 14  8|60|60|  3 2 2 2 1|10|10|  1 0 2 4 3|10|10|
10 21  4 23  7|65|65|   9 20  3 22  6|60|60|  1 4 0 4 1|10| 9|  4 0 3 2 1|10|15|
18  6 14  2 25|65|65|  17  5 13  1 24|60|60|  3 1 2 0 4|10|10|  2 0 3 1 4|10|10|
          5001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 14 21 20  9|65|65|   0 13 20 19  8|60|60|  0 2 4 3 1|10|11|  0 3 0 4 3|10| 5|
24  8  3 11 19|65|65|  23  7  2 10 18|60|60|  4 1 0 2 3|10|10|  3 2 2 0 3|10|10|
16 22 13  4 10|65|65|  15 21 12  3  9|60|60|  3 4 2 0 1|10|10|  0 1 2 3 4|10|10|
 7 15 23 18  2|65|65|   6 14 22 17  1|60|60|  1 2 4 3 0|10|10|  1 4 2 2 1|10|10|
17  6  5 12 25|65|65|  16  5  4 11 24|60|60|  3 1 0 2 4|10| 9|  1 0 4 1 4|10|15|
          6001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1 24 22  8 10|65|65|   0 23 21  7  9|60|60|  0 4 4 1 1|10|11|  0 3 1 2 4|10| 5|
12 15  6 23  9|65|65|  11 14  5 22  8|60|60|  2 2 1 4 1|10| 9|  1 4 0 2 3|10|15|
19  5 13 21  7|65|65|  18  4 12 20  6|60|60|  3 0 2 4 1|10|10|  3 4 2 0 1|10|10|
17  3 20 11 14|65|65|  16  2 19 10 13|60|60|  3 0 3 2 2|10|11|  1 2 4 0 3|10| 5|
16 18  4  2 25|65|65|  15 17  3  1 24|60|60|  3 3 0 0 4|10| 9|  0 2 3 1 4|10|15|
          7001#\65/65| [Math]          \60/60|  /D5i     H\10/10|          L\10/10|
 1  8 23 22 11|65|65|   0  7 22 21 10|60|60|  0 1 4 4 2|11|10|  0 2 2 1 0| 5|10|
 9 21 19 14  2|65|65|   8 20 18 13  1|60|60|  1 4 3 2 0|10|10|  3 0 3 3 1|10|10|
16 20 13  6 10|65|65|  15 19 12  5  9|60|60|  3 3 2 1 1|10|10|  0 4 2 0 4|10|10|
24 12  7  5 17|65|65|  23 11  6  4 16|60|60|  4 2 1 0 3|10|10|  3 1 1 4 1|10|10|
15  4  3 18 25|65|65|  14  3  2 17 24|60|60|  2 0 0 3 4| 9|10|  4 3 2 2 4|15|10|
```

. . . . .

（１）２本の主対角線と中央行と中央列に限って見ると，和が 10 のものばかりである。

n1+n7+n13+n19+n25=10; n5+n9+n13+n17+n21=10; n3+n8+n13+n18+n23=10; n11+n12+n13+n14+n15=10;

また中央の n13 には，必ず 2 が来ている。例外は無い。

（２）相互に対称の位置にある２数の間には，a+b=4 が成り立つ。ペアは，必ず 4 の補数を成す。中央 n13 は，自己自身に対して対称なので，n13+n13=4; 故に n13=2; 対称方陣の特性の一つが単位方陣にも見られる。上の４本の定和は，全て (a+b)+n13+(c+d)=4+2+4=10 の構成になっている。（１）の性質は，全て対称方陣の定義から導き出される。

（３）他の行列の和は，必ずしも定和 10 に等しいとは限らない。もちろん 10 のものもあるが，高位の単位方陣には，9 のものも 11 もある。8 や 12 のものは無いようだ。低位の単位方陣には，和が 5, 10, 15 のものしか見当たらない。高位と低位との間には和に関して相関関係があり，高位 9 の時は低位 15，高位 10 の時は低位 10，高位 11 の時は低位 5 と決まっているようだ。他の組み合せは無い。これはどういうことか。

これは，単位方陣の場合はいざ知らず，対象方陣全体で行列の定和が必ず成り立たねばなら

ないからだ。行列各５数の和は，最終の数学的表記において 60 でなければならず，それを満たす可能性はあくまで次のような組み合せの時に限られるからだ。

（高位の５数の和×５＋低位の５数の和＝60）

```
[1]  8 x 5 + 20 = 60;   [2]  9 x 5 + 15 = 60;   [3] 10 x 5 + 10 = 60;
[4] 11 x 5 +  5 = 60;   [5] 12 x 5 +  0 = 60;
```

低位単位方陣内に 4+4+4+4+4=20 や 0+0+0+0+0=0 の行列は出現しない。出現すると，最終的に使用数字の重複・欠落が起きてしまう。だから，上の[2],[3],[4]の場合だけを考えれば良い。

（４）どの単位方陣内でも，(0, 1, 2, 3, 4)の５数は，いずれも平等に５度ずつ現れなければならない。これも使用数字の重複・欠落を防ぐ約束から来る常に不変の性質である。

５．以上の考察をふまえて，全ての単位方陣を作らなければならない。それをプログラムにどう表現するか。頭を使う時だ。

（１），（２）および（４）は，以前と同じに考えてプログラムを組めば良い。

問題は，（３）だ。これだけでも，高位方陣と低位方陣とを別々に作る必要が生じる。行列の和に関して，与える条件が違うからだ。一方は，8＜和＜12 であり，他方は和＝5, 10, or 15 だからだ。単位方陣を統一的に扱うという新オイラー法の骨子を崩すことにはなってしまうが，今はやむを得まい。… 結局次のようにプログラムを組んだ。

いつも和を監視すべきは，２行・２列だけで良い。その４本が決まれば，対称型補数のおかげで，それに応じて他の２行２列が自動的に決まるからだ。

```
//** 'New-Euler Type' of Self-complementary Magic Squares 5x5 **
//** Composed by "New Euler's Method" with /D5i **
//** 'NEMS55SC.c': Dictated by Kanji Setsuda  **
//** on Apr.11, 2006, Dec.18, 2011 and Jan. 7, 2012 **
//** Working with MacOSX 10.7.2 & Xcode 4.2.1 **
//
//** Definitions for Self-complementary Magic Squares of Order 5 **
//.--.--.--.--.--. * Basic Form and Equations: C=65; *
//| 1| 2| 3| 4| 5|  n1+n2+n3+n4+n5=C      ...sm1;
//|--+--+--+--+--|  n6+n7+n8+n9+n10=C     ...sm2;
//| 6| 7| 8| 9|10|  n11+n12+n13+n14+n15=C ...sm3;
//|--+--+--+--+--|  n16+n17+n18+n19+n20=C ...sm4;
//|11|12|13|14|15|  n21+n22+n23+n24+n25=C ...sm5;
//|--+--+--+--+--|  n1+n6+n11+n16+n21=C   ...sm6;
//|16|17|18|19|20|  n2+n7+n12+n17+n22=C   ...sm7;
//|--+--+--+--+--|  n3+n8+n13+n18+n23=C   ...sm8;
//|21|22|23|24|25|  n4+n9+n14+n19+n24=C   ...sm9;
//'--'--'--'--'--'  n5+n10+n15+n20+n25=C  ...sm10;
//  n1+n7+n13+n19+n25=C ...sm11;  n5+n9+n13+n17+n21=C ...sm12;
//* Self-complementary Conditions: SC=26; *
//  n1+n25=n2+n24=n3+n23=n4+n22=n5+n21
// =n6+n20=n7+n19=n8+n18=n9+n17=n10+n16
// =n11+n15=n12+n14=n13+n13=SC
//
#include <stdio.h>
//
long int cnt, hcnt, lcnt, cnt2;
long lucnt[8];
long u1,u2;
short cnt3;
short nm[30], uflg[26];
short anm[8][76];
char htlu[39917][29];
char ltlu[30097][29];
char mtc[39917][30097];
short solt[48545][26];
long cntr[48545][3];
//
void hstp01(void), hstp02(void), hstp03(void), hstp04(void), hstp05(void);
```

```
void hstp06(void), hstp07(void), hstp08(void), hstp09(void), hstp10(void);
void hstp11(void), hstp12(void);
void hansrecord(void), prhunit(short x);
//
void lstp01(void), lstp02(void), lstp03(void), lstp04(void), lstp05(void);
void lstp06(void), lstp07(void), lstp08(void), lstp09(void), lstp10(void);
void lstp11(void), lstp12(void);
void lansrecord(void), prlunit(short x);
//
void tblmtc(void);
void cmbcmp(void), solsave(void);
void prans(short x), pr2ans(void), pr1ans(void);
//
void makemod(void);
void defmod(short x, short y);
//
//* Main Program *
int main(void){
 short n;
  printf("\n");
  printf("** Self-complementary Magic Squares of Order 5: **\n");
  printf(" ** Made by our 'New Euler's Method' with /D5i **\n");
  for(n=0;n<5;n++){uflg[n]=0;}
  nm[13]=2; uflg[2]=1;
  cnt=0;
  printf("\n");
  printf(" ** List of Euler Units for High Positions by /D5i **\n");
  hstp01();                       // Make the Euler Units for High Positions
  if(cnt3>0){prlunit(cnt3);}      // Print the Rest One
  hcnt=cnt; printf(" [Count of Euler Units for High Positions = %ld]\n",hcnt);
  for(n=0;n<5;n++){uflg[n]=0;}
  nm[13]=2; uflg[2]=1;
  cnt=0; cnt3=0;
  printf("\n");
  printf(" ** List of Euler Units for Low Positions by /D5i **\n");
  lstp01();                       // Make the Euler Units for Low Positions
  if(cnt3>0){prlunit(cnt3);}      // Print the Rest One
  lcnt=cnt; printf(" [Count of Euler Units for Low Positions = %ld]\n",lcnt);
  tblmtc();
  printf("\n");
  printf("** Standard Solutions of Self-complementary Magic Squares 5x5: made by our\n");
  printf("   'New Euler's Method' with Decompositions by the 5th Increment Number System **\n");
  printf("   (Euler Units: High/Low/  Sol_Number#  Check Sums:\\Diagonals/Rows|Columns|)\n");
  cnt=0; cnt3=0;
  cmbcmp();                       // Combine and Compose
  prans(1000);                    // Print the Solutions with that Interval
  if(cnt3==1){pr1ans();}          // Print the Rest One
  printf(" [Solution Counts = %ld]\n",cnt);
  printf(" [OK!]\n");
  printf("** Calculated and listed by Kanji Setsuda\n");
  printf("    on Jan. 7, 2012 with MacOSX 10.7.2 & Xcode 4.2.1 **\n");
  printf("\n");
 return 0;
}
//
// Make the Euler Units for High Positions
// Level #1:
// Set N1 & n25 & N1<=n25
void hstp01(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((j>=i)&&(uflg[i]<5)&&(uflg[j]<5)){
    nm[1]=i; nm[25]=j;
    uflg[i]++; uflg[j]++;
    hstp02();
```

```
    uflg[i]--; uflg[j]--;}
  }
}
// Set N5 & n21 & N1<=N5<=n21
void hstp02(){
 short i,j;
  for(i=nm[1];i<5;i++){j=4-i;
    if((j>=i)&&(uflg[i]<5)&&(uflg[j]<5)){
    nm[5]=i; nm[21]=j;
    uflg[i]++; uflg[j]++;
    hstp03();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N7 & n19
void hstp03(){
 short i,j,sm;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){cnt2=0;
    nm[7]=i; nm[19]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[1]+nm[7]+nm[13]+nm[19]+nm[25];
    if(sm==10){hstp04();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N9 & n17
void hstp04(){
 short i,j,sm;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[9]=i; nm[17]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[5]+nm[9]+nm[13]+nm[17]+nm[21];
    if(sm==10){hstp05();}
    uflg[i]--; uflg[j]--;}
  }
}
// Level #2:
// Set N2 & n24
void hstp05(){
 short i,j;
  for(i=4;i>=0;i--){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[2]=i; nm[24]=j;
    uflg[i]++; uflg[j]++;
    hstp06();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N4 & n22
void hstp06(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[4]=i; nm[22]=j;
    uflg[i]++; uflg[j]++;
    hstp07();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N3 & n23
void hstp07(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
```

```
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[3]=i; nm[23]=j;
    uflg[i]++; uflg[j]++;
    sm1=nm[1]+nm[2]+nm[3]+nm[4]+nm[5];
    sm2=nm[25]+nm[24]+nm[23]+nm[22]+nm[21];
    if((8<sm1)&&(sm1<12)&&(sm1+sm2==20)){nm[26]=sm1; hstp08();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N8 & n18
void hstp08(){
 short i,j,sm;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[8]=i; nm[18]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[3]+nm[8]+nm[13]+nm[18]+nm[23];
    if(sm==10){hstp09();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N12 & n14
void hstp09(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[12]=i; nm[14]=j;
    uflg[i]++; uflg[j]++;
    sm1=nm[2]+nm[7]+nm[12]+nm[17]+nm[22];
    sm2=nm[24]+nm[19]+nm[14]+nm[9]+nm[4];
    if((8<sm1)&&(sm1<12)&&(sm1+sm2==20)){nm[28]=sm1; hstp10();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N6 & n20
void hstp10(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[6]=i; nm[20]=j;
    uflg[i]++; uflg[j]++;
    hstp11();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N10 & n16
void hstp11(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[10]=i; nm[16]=j;
    uflg[i]++; uflg[j]++;
    sm1=nm[6]+nm[7]+nm[8]+nm[9]+nm[10];
    sm2=nm[20]+nm[19]+nm[18]+nm[17]+nm[16];
    if((8<sm1)&&(sm1<12)&&(sm1+sm2==20)){nm[29]=sm1; hstp12();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N11 & n15
void hstp12(){
 short i,j,sm1,sm2,sm3;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[11]=i; nm[15]=j;
    uflg[i]++; uflg[j]++;
```

```
     sm1=nm[1]+nm[6]+nm[11]+nm[16]+nm[21];
     sm2=nm[25]+nm[20]+nm[15]+nm[10]+nm[5];
     sm3=nm[11]+nm[12]+nm[13]+nm[14]+nm[15];
     if((8<sm1)&&(sm1<12)&&(sm1+sm2==20)){
       if(sm3==10){nm[27]=sm1; hansrecord();}}
     uflg[i]--; uflg[j]--;}
   }
}
//
// Record the High Units
void hansrecord(){
 short n;
  for(n=0;n<29;n++){htlu[cnt][n]=nm[n+1];}
  cnt++; cnt2++;
  if(cnt2==1){lucnt[cnt3]=cnt;
    for(n=0;n<25;n++){anm[cnt3][n]=nm[n+1];}
    cnt3++; if(cnt3==7){prlunit(cnt3); cnt3=0;}
  }
}
//
//** Possible Conditions for our New Euler's Units **
// ** How is the Constant Sum of Lines available? **
//   [1]  8 x 5 + 20 = 60;    [2]  9 x 5 + 15 = 60;
//   [3] 10 x 5 + 10 = 60;    [4] 11 x 5 +  5 = 60;
//   [5] 12 x 5 +  0 = 60;
//
// Make the Euler Units for Low Positions
// Level #1:
// Set N1 & n25
void lstp01(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[1]=i; nm[25]=j;
    uflg[i]++; uflg[j]++;
    lstp02();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N5 & n21
void lstp02(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){cnt2=0;
    nm[5]=i; nm[21]=j;
    uflg[i]++; uflg[j]++;
    lstp03();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N7 & n19
void lstp03(){
 short i,j,sm;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){if(nm[1]==0){cnt2=0;}
    nm[7]=i; nm[19]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[1]+nm[7]+nm[13]+nm[19]+nm[25];
    if(sm==10){lstp04();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N9 & n17
void lstp04(){
 short i,j,sm;
```

```
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[9]=i; nm[17]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[5]+nm[9]+nm[13]+nm[17]+nm[21];
    if(sm==10){lstp05();}
    uflg[i]--; uflg[j]--;}
  }
}
// Level #2:
// Set N2 & n24
void lstp05(){
 short i,j;
  for(i=4;i>=0;i--){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[2]=i; nm[24]=j;
    uflg[i]++; uflg[j]++;
    lstp06();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N4 & n22
void lstp06(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[4]=i; nm[22]=j;
    uflg[i]++; uflg[j]++;
    lstp07();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N3 & n23
void lstp07(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[3]=i; nm[23]=j;
      uflg[i]++; uflg[j]++;
      sm1=nm[1]+nm[2]+nm[3]+nm[4]+nm[5];
      sm2=nm[25]+nm[24]+nm[23]+nm[22]+nm[21];
      if((sm1==5)||(sm1==10)||(sm1==15)){
        if(sm1+sm2==20){nm[26]=sm1; lstp08();}}
      uflg[i]--; uflg[j]--;}
  }
}
// Set N8 & n18
void lstp08(){
 short i,j,sm;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[8]=i; nm[18]=j;
    uflg[i]++; uflg[j]++;
    sm=nm[3]+nm[8]+nm[13]+nm[18]+nm[23];
    if(sm==10){lstp09();}
    uflg[i]--; uflg[j]--;}
  }
}
// Set N12 & n14
void lstp09(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[12]=i; nm[14]=j;
      uflg[i]++; uflg[j]++;
```

```
      sm1=nm[2]+nm[7]+nm[12]+nm[17]+nm[22];
      sm2=nm[24]+nm[19]+nm[14]+nm[9]+nm[4];
      if((sm1==5)||(sm1==10)||(sm1==15)){
        if(sm1+sm2==20){nm[28]=sm1; lstp10();}}
      uflg[i]--; uflg[j]--;}
  }
}
// Set N6 & n20
void lstp10(){
 short i,j;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
    nm[6]=i; nm[20]=j;
    uflg[i]++; uflg[j]++;
    lstp11();
    uflg[i]--; uflg[j]--;}
  }
}
// Set N10 & n16
void lstp11(){
 short i,j,sm1,sm2;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[10]=i; nm[16]=j;
      uflg[i]++; uflg[j]++;
      sm1=nm[6]+nm[7]+nm[8]+nm[9]+nm[10];
      sm2=nm[20]+nm[19]+nm[18]+nm[17]+nm[16];
      if((sm1==5)||(sm1==10)||(sm1==15)){
        if(sm1+sm2==20){nm[29]=sm1; lstp12();}}
      uflg[i]--; uflg[j]--;}
  }
}
// Set N11 & n15
void lstp12(){
 short i,j,sm1,sm2,sm3;
  for(i=0;i<5;i++){j=4-i;
    if((uflg[i]<5)&&(uflg[j]<5)){
      nm[11]=i; nm[15]=j;
      uflg[i]++; uflg[j]++;
      sm1=nm[1]+nm[6]+nm[11]+nm[16]+nm[21];
      sm2=nm[25]+nm[20]+nm[15]+nm[10]+nm[5];
      sm3=nm[11]+nm[12]+nm[13]+nm[14]+nm[15];
      if((sm1==5)||(sm1==10)||(sm1==15)){
        if((sm1+sm2==20)&&(sm3==10)){nm[27]=sm1; lansrecord();}}
      uflg[i]--; uflg[j]--;}
  }
}
//
// Record Low Units
void lansrecord(){
 short n;
  for(n=0;n<29;n++){ltlu[cnt][n]=nm[n+1];}
  cnt++; cnt2++;
  if(cnt2==1){lucnt[cnt3]=cnt;
    for(n=0;n<25;n++){anm[cnt3][n]=nm[n+1];}
    cnt3++; if(cnt3==7){prlunit(cnt3); cnt3=0;}
  }
}
//
// Print Euler Units
void prlunit(short x){
 short l,m,m5,n;
  for(l=0;l<x;l++){printf("  %9ld/",lucnt[l]);}
  printf("\n");
  for(m=0;m<5;m++){m5=m*5;
```

```
    for(l=0;l<x;l++){printf("  ");
      for(n=0;n<5;n++){printf("%2d",anm[l][m5+n]);}}
    printf("\n");
  }
}
//
// Check how well they match and Save the Data to the Table
void tblmtc(){
 long h,l;
 short n,d1,d2;
  for(h=0;h<hcnt;h++){
    for(l=0;l<lcnt;l++){
      d1=0; for(n=1;n<26;n++){if(htlu[h][n]==ltlu[l][n]){d1++;}}
      d2=0; for(n=1;n<26;n++){if(htlu[h][n]+ltlu[l][n]==4){d2++;}}
      if(d1<d2){d1=d2;}
      mtc[h][l]=d1;}
  }
}
//
//* Combine, Compose and Print *
void cmbcmp(){
 short n,d,fc;
 short s1,s2,s3,s4,s5,s6,s7,s8;
  for(u1=0;u1<hcnt;u1++){
    s1=htlu[u1][25]; s2=htlu[u1][26]; s3=htlu[u1][27]; s4=htlu[u1][28];
    if((8<s1)&&(s1<12)&&(8<s2)&&(s2<12)&&(8<s3)&&(s3<12)&&(8<s4)&&(s4<12)){
      for(u2=0;u2<lcnt;u2++){
        s5=ltlu[u2][25]; s6=ltlu[u2][26]; s7=ltlu[u2][27]; s8=ltlu[u2][28];
        if((s1*5+s5==60)&&(s2*5+s6==60)&&(s3*5+s7==60)&&(s4*5+s8==60)){
          if(mtc[u1][u2]==5){
            for(n=1;n<26;n++){uflg[n]=0;}
            for(n=0;n<25;n++){
              d=htlu[u1][n]*5+ltlu[u2][n]+1;
              nm[n+1]=d; uflg[d]++;
            }
            fc=0;
            for(n=1;n<26;n++){if(uflg[n]==0){fc++;}}
            if(fc==0){
              if((nm[1]<nm[5])&&(nm[5]<nm[21])&&(nm[1]<nm[25])){solsave();}}
          }}
      }
    }}
}
//
// Save Answers to the Solution List
void solsave(){
 short n;
  for(n=1;n<26;n++){solt[cnt][n]=nm[n];}
  cntr[cnt][0]=cnt+1; cntr[cnt][1]=u1;  cntr[cnt][2]=u2;
  cnt++;
}
//
// Print the Solution List
void prans(short x){
 long s;
 short m,n,sm1,sm2;
  for(s=0;s<cnt;s=s+x){
    lucnt[cnt3*3]=s+1;
    for(n=1;n<26;n++){nm[n]=solt[s][n]; anm[cnt3][n]=nm[n]; anm[cnt3+1][n]=0;}
    for(m=0;m<5;m++){sm1=0; sm2=0;
      for(n=0;n<5;n++){sm1=sm1+nm[m*5+n+1]; sm2=sm2+nm[n*5+m+1];}
      anm[cnt3][m+26]=sm1; anm[cnt3][m+31]=sm2;}
    anm[cnt3][36]=nm[1]+nm[7]+nm[13]+nm[19]+nm[25];
    anm[cnt3][37]=nm[5]+nm[9]+nm[13]+nm[17]+nm[21];
    lucnt[cnt3*3+1]=cntr[s][1]+1; lucnt[cnt3*3+2]=cntr[s][2]+1;
```

```
    cnt3++;
    if(cnt3==2){pr2ans(); cnt3=0;}
  }
}
// Print 2 Answers with Sum Checks
void pr2ans(){
 long lu1,lu2;
 short l,l5,m,n;
  printf("%6ld/H%6ld/L%15ld# \\%2d/%2d| %6ld/H%6ld/L%15ld# \\%2d/%2d|\n",
       lucnt[1],lucnt[2],lucnt[0],anm[0][36],anm[0][37],
       lucnt[4],lucnt[5],lucnt[3],anm[1][36],anm[1][37]);
  for(l=0;l<5;l++){l5=l*5;
    for(m=0;m<2;m++){
      lu1=lucnt[m*3+1]-1; lu2=lucnt[m*3+2]-1;
      printf("  ");
      for(n=0;n<5;n++){printf("%d",htlu[lu1][l5+n]);}
      printf("   ");
      for(n=0;n<5;n++){printf("%d",ltlu[lu2][l5+n]);}
      printf("  ");
      for(n=1;n<6;n++){printf("%3d",anm[m][l5+n]);}
      printf(" |%2d|%2d|",anm[m][l+26],anm[m][l+31]);
      if(m<1){printf("  ");}
    }
    printf("\n");
  }
}
// Print the Rest One
void pr1ans(){
 long lu1,lu2;
 short l,l5,n;
  lu1=lucnt[1]-1; lu2=lucnt[2]-1;
  printf("%6ld/H%6ld/L%15ld# \\%2d/%2d|\n",
       lu1+1,lu2+1,lucnt[0],anm[0][36],anm[0][37]);
  for(l=0;l<5;l++){l5=l*5;
    printf("  ");
    for(n=0;n<5;n++){printf("%d",htlu[lu1][l5+n]);}
    printf("   ");
    for(n=0;n<5;n++){printf("%d",ltlu[lu2][l5+n]);}
    printf("  ");
    for(n=1;n<6;n++){printf("%3d",anm[0][l5+n]);}
    printf(" |%2d|%2d|\n",anm[0][l+26],anm[0][l+31]);
  }
}
```

上のCプログラムには，単位方陣の構成・出力だけでなく，それを組み合わせて計算・合成・出力する過程も含まれている。根気良くたどって行っていただきたい。

早速にも実行結果を出力させよう。先ずは，単位方陣から。

** 五進法による新オイラー法で補数対称五方陣を作る **

** 五進法による新オイラー法で作った単位方陣（高位用）のリスト：抜粋 **

```
      1/       682/      2513/      3929/      5760/      6441/      8048/
0 4 3 2 0  0 4 3 2 0  0 4 2 3 0  0 3 3 3 0  0 2 3 4 0  0 4 2 2 1  0 4 2 2 1
3 0 3 0 3  3 1 2 0 3  3 2 1 0 3  2 3 2 0 3  2 4 1 0 3  3 0 3 0 3  3 1 1 0 4
2 1 2 3 2  1 0 2 4 3  1 0 2 4 3  4 0 2 4 0  3 1 2 3 1  3 0 2 4 1  3 0 2 4 1
1 4 1 4 1  1 4 2 3 1  1 4 3 2 1  1 4 2 1 2  1 4 3 0 2  1 4 1 4 1  0 4 3 3 1
4 2 1 0 4  4 2 1 0 4  4 1 2 0 4  4 1 1 1 4  4 0 1 2 4  3 2 2 0 4  3 2 2 0 4

     9774/     11592/     13318/     14925/     16299/     18291/     19051/
0 4 1 3 1  0 4 1 4 1  0 3 1 4 1  0 4 2 1 2  0 4 1 2 2  0 4 0 3 2  0 4 1 4 2
4 2 1 0 2  3 3 1 0 2  2 4 1 0 2  3 0 3 0 3  4 1 1 0 3  3 2 1 0 3  3 3 1 0 2
1 0 2 4 3  2 0 2 4 2  3 0 2 4 1  3 0 2 4 1  3 0 2 4 1  3 0 2 4 1  3 0 2 4 1
2 4 3 2 0  2 4 3 1 1  2 4 3 0 2  1 4 1 4 1  1 4 3 3 0  1 4 3 2 1  2 4 3 1 1
3 1 3 0 4  3 0 3 0 4  3 0 3 1 4  2 3 2 0 4  2 2 3 0 4  2 1 4 0 4  2 0 3 0 4
```

21043/
0 3 1 4 2
3 4 1 0 1
2 0 2 4 2
3 4 3 0 1
2 0 3 1 4

22417/
1 4 2 1 1
3 0 2 0 4
3 0 2 4 1
0 4 2 4 1
3 3 2 0 3

24419/
1 4 1 2 1
4 1 1 0 4
2 0 2 4 2
0 4 3 3 0
3 2 3 0 3

25389/
1 4 0 3 1
2 2 1 0 4
3 0 2 4 1
0 4 3 2 2
3 1 4 0 3

26909/
1 4 0 4 1
2 3 1 0 3
2 0 2 4 2
1 4 3 1 2
3 0 4 0 3

27879/
1 3 0 4 1
2 4 1 0 2
1 0 2 4 3
2 4 3 0 2
3 0 4 1 3

29881/
1 4 1 1 2
3 0 2 0 4
3 0 2 4 1
0 4 2 4 1
2 3 3 0 3

31769/
1 4 0 2 2
3 1 1 0 4
3 0 2 4 1
0 4 3 3 1
2 2 4 0 3

33127/
1 4 0 3 2
3 2 1 0 3
4 0 2 4 0
1 4 3 2 1
2 1 4 0 3

33855/
1 4 0 4 2
2 3 1 0 3
3 0 2 4 1
1 4 3 1 2
2 0 4 0 3

35213/
1 3 0 4 2
2 4 1 0 3
3 0 2 4 1
1 4 3 0 2
2 0 4 1 3

37101/
2 4 0 1 2
3 0 3 0 3
1 0 2 4 3
1 4 1 4 1
2 3 4 0 2

37805/
2 4 0 3 2
3 1 1 0 4
3 0 2 4 1
0 4 3 3 1
2 1 4 0 2

38509/
2 3 0 3 2
3 3 0 0 4
3 0 2 4 1
0 4 4 1 1
2 1 4 1 2

39213/
2 3 0 4 2
1 4 1 0 3
3 0 2 4 1
1 4 3 0 3
2 0 4 1 2

[高位用単位方陣の総数 = 39916]

** 五進法による新オイラー法で作った単位方陣（低位用）のリスト：抜粋 **

1/
0 4 3 3 0
1 0 2 0 2
3 1 2 3 1
2 4 2 4 3
4 1 1 0 4

165/
0 4 1 0 0
1 1 1 0 2
3 2 2 2 1
2 4 3 3 3
4 4 3 0 4

438/
0 4 1 0 0
1 2 1 0 1
2 1 2 3 2
3 4 3 2 3
4 4 3 0 4

668/
0 4 0 1 0
2 3 2 0 3
3 1 2 3 1
1 4 2 1 2
4 3 4 0 4

941/
0 4 3 3 0
2 4 1 0 3
3 2 2 2 1
1 4 3 0 2
4 1 1 0 4

1105/
0 4 2 3 1
0 0 2 0 3
1 1 2 3 3
1 4 2 4 4
3 1 2 0 4

1401/
0 4 0 0 1
1 1 2 0 1
3 2 2 2 1
3 4 2 3 3
3 4 4 0 4

1697/
0 4 0 0 1
3 2 2 0 3
3 1 2 3 1
1 4 2 2 1
3 4 4 0 4

1985/
0 4 4 1 1
2 3 2 0 3
4 1 2 3 0
1 4 2 1 2
3 3 0 0 4

2281/
0 4 3 2 1
2 4 1 0 3
4 1 2 3 0
1 4 3 0 2
3 2 1 0 4

2577/
0 4 3 1 2
3 0 1 0 1
2 4 2 0 2
3 4 3 4 1
2 3 1 0 4

2791/
0 4 3 1 2
3 1 2 0 4
0 3 2 1 4
0 4 2 3 1
2 3 1 0 4

3080/
0 4 1 3 2
1 2 1 0 1
4 4 2 0 0
3 4 3 2 3
2 1 3 0 4

3182/
0 4 3 1 2
0 3 0 0 2
1 1 2 3 3
2 4 4 1 4
2 3 1 0 4

3471/
0 4 3 1 2
3 4 1 0 2
3 0 2 4 1
2 4 3 0 1
2 3 1 0 4

3685/
0 4 3 0 3
1 0 2 0 2
1 3 2 1 3
2 4 2 4 3
1 4 1 0 4

3981/
0 4 3 0 3
1 1 0 0 3
2 2 2 2 2
1 4 4 3 3
1 4 1 0 4

4277/
0 4 3 0 3
1 2 0 0 2
1 1 2 3 3
2 4 4 2 3
1 4 1 0 4

4565/
0 4 3 0 3
2 3 2 0 3
1 0 2 4 3
1 4 2 1 2
1 4 1 0 4

4861/
0 4 2 1 3
2 4 1 0 3
1 0 2 4 3
1 4 3 0 2
1 3 2 0 4

5157/
0 3 1 2 4
3 0 2 0 0
3 1 2 3 1
4 4 2 4 1
0 2 3 1 4

5321/
0 4 2 0 4
3 1 3 0 3
1 2 2 2 3
1 4 1 3 1
0 4 2 0 4

5594/
0 4 2 0 4
1 2 1 0 1
1 1 2 3 3
3 4 3 2 3
0 4 2 0 4

5824/
0 4 1 1 4
3 3 2 0 2
0 1 2 3 4
2 4 2 1 1
0 3 3 0 4

6097/
0 3 2 1 4
3 4 1 0 2
0 1 2 3 4
2 4 3 0 1
0 3 2 1 4

6261/
1 4 2 3 0
3 0 2 0 0
3 1 2 3 1
4 4 2 4 1
4 1 2 0 3

7733/
1 4 4 0 1
1 0 2 0 2
3 3 2 1 1
2 4 2 4 3
3 4 0 0 3

8841/
1 4 3 0 2
0 0 2 0 3
1 3 2 1 3
1 4 2 4 4
2 4 1 0 3

9945/
1 4 2 0 3
3 0 0 0 2
3 3 2 1 1
2 4 4 4 1
1 4 2 0 3

11053/
1 4 1 0 4
1 0 2 0 2
1 3 2 1 3
2 4 2 4 3
0 4 3 0 3

12525/
2 4 3 1 0
1 0 1 0 3
2 4 2 0 2
1 4 3 4 3
4 3 1 0 2

13633/
2 4 3 0 1
3 0 2 0 0
3 3 2 1 1
4 4 2 4 1
3 4 1 0 2

14737/
2 4 1 1 2
1 0 1 0 3
4 4 2 0 0
1 4 3 4 3
2 3 3 0 2

15281/
2 4 1 0 3
0 0 2 0 3
1 3 2 1 3
1 4 2 4 4
1 4 3 0 2

16385/
2 4 3 2 4
1 0 1 0 3
1 0 2 4 3
1 4 3 4 3
0 2 1 0 2

17493/
3 4 3 0 0
2 0 2 0 1
3 3 2 1 1
3 4 2 4 2
4 4 1 0 1

18965/
3 4 2 0 1
2 0 0 0 3
1 3 2 1 3
1 4 4 4 2
3 4 2 0 1

20073/
3 4 1 0 2
3 0 2 0 0
3 3 2 1 1
4 4 2 4 1
2 4 3 0 1

21177/
3 4 0 0 3
1 0 2 0 2
3 3 2 1 1
2 4 2 4 3
1 4 4 0 1

22285/
3 4 3 1 4
2 0 1 0 2
3 4 2 0 1
2 4 3 4 2
0 3 1 0 1

23757/
4 3 1 2 0
0 0 2 0 3
1 1 2 3 3
1 4 2 4 4
4 2 3 1 0

24861/
4 4 1 0 1
2 0 2 0 1
3 3 2 1 1
3 4 2 4 2
3 4 3 0 0

26333/
4 4 3 2 2
3 0 1 0 1
3 0 2 4 1

27441/
4 4 3 1 3
2 0 1 0 2
1 4 2 0 3

28913/
4 3 3 1 4
1 0 2 0 2
3 0 2 4 1

```
３４３４１　２４３４２　２４２４３
２２１００　１３１００　０３１１０　［低位用単位方陣の総数 = 30016］
```

何とも凄い数あるものだ。組み合わせてチェックしなければならない場合の数は，全部で 39916×30016通り，約十二億個にもなる。相性表を作ったり，組み合わせて対象方陣を作ったり，さらに検定試験を施したりすると，いかに強力なコンピューターといえども，さぞかし時間が掛かる。内部メモリーも大量に使用する。私の古いマシンでは，とても対応し切れなかった（仕方なく，最新のマシンを調達・導入することにしたが）。

新しいマシンでさえ十余分間も待たされたが，とにもかくにも次の如く出力して来た。

**** 五進法による新「新オイラー法」で作った補数対称五方陣の標準解のリスト：抜粋 ****

（使用した単位方陣：高位/H 低位/L　解の番号#　サム・チェック：\対角線/|行|列|）

```
 35/H 16810/L                1# \65/65|   1967/H  9144/L             2001# \65/65|
04410  21444   3 22 25 10  5 |65|65|   04320  14442   2 25 20 15  3 |65|65|
30213  41433  20  2 15  9 19 |65|65|   31034  11300  17  7  4 16 21 |65|65|
21232  32221  14  8 13 18 12 |65|65|   32221  21232  18 12 13 14  8 |65|65|
13241  11030   7 17 11 24  6 |65|65|   01431  44133   5 10 22 19  9 |65|65|
43004  00032  21 16  1  4 23 |65|65|   42104  20003  23 11  6  1 24 |65|65|

 3985/H  3382/L             4001# \65/65|   6030/H  4562/L             6001# \65/65|
03340  01432   1 17 20 24  3 |65|65|   03430  00343   1 16 24 20  4 |65|65|
23301  33234  14 19 18  4 10 |65|65|   24120  32244  14 23  8 15  5 |65|65|
20242  04204  11  5 13 21 15 |65|65|   11233  13213   7  9 13 17 19 |65|65|
34112  01211  16 22  8  7 12 |65|65|   42302  00221  21 11 18  3 12 |65|65|
40114  21034  23  2  6  9 25 |65|65|   41014  10144  22  6  2 10 25 |65|65|

 7081/H  3137/L             8001# \65/65|   7917/H 29691/L            10001# \65/65|
02431  04042   1 15 21 20  8 |65|65|   04221  44304   5 25 14 11 10 |65|65|
40312  12331  22  3 19  9 12 |65|65|   30043  13123  17  4  2 23 19 |65|65|
14203  43210  10 24 13  2 16 |65|65|   33211  42220  20 18 13  8  6 |65|65|
23140  31123  14 17  7 23  4 |65|65|   10441  12313   7  3 24 22  9 |65|65|
31024  20404  18  6  5 11 25 |65|65|   32204  04100  16 15 12  1 21 |65|65|

 8811/H  3039/L            12001# \65/65|   9666/H 14433/L            14001# \65/65|
04411  03232   1 24 23  9  8 |65|65|   04331  20201   3 21 18 16  7 |65|65|
41023  11440  22  7  5 15 16 |65|65|   21240  43143  15  9 12 25  4 |65|65|
21232  30241  14  6 13 20 12 |65|65|   14203  03214   6 24 13  2 20 |65|65|
12430  40033  10 11 21 19  4 |65|65|   40232  10310  22  1 14 17 11 |65|65|
33004  21214  18 17  3  2 25 |65|65|   31104  34242  19 10  8  5 23 |65|65|

10563/H  9520/L            16001# \65/65|  11547/H 29635/L            18001# \65/65|
01441  13042   2  9 21 25  8 |65|65|   03141  44124   5 20  7 23 10 |65|65|
22411  43210  15 14 23  7  6 |65|65|   22340  03110  11 14 17 22  1 |65|65|
14203  41230  10 22 13  4 16 |65|65|   14203  23212   8 24 13  2 18 |65|65|
33022  43210  20 19  3 12 11 |65|65|   40122  43314  25  4  9 12 15 |65|65|
30034  20413  18  1  5 17 24 |65|65|   30314  02300  16  3 19  6 21 |65|65|

12444/H  3186/L            20001# \65/65|  13276/H  6377/L            22001# \65/65|
03241  04132   1 20 12 24  8 |65|65|   02431  14410   2 15 25 17  6 |65|65|
43120  03403  21 19 10 11  4 |65|65|   23140  30124  14 16  7 23  5 |65|65|
01234  23212   3  9 13 17 23 |65|65|   14203  21232   8 22 13  4 18 |65|65|
42310  14014  22 15 16  7  5 |65|65|   40312  02341  21  3 19 10 12 |65|65|
30214  21304  18  2 14  6 25 |65|65|   31024  43003  20  9  1 11 24 |65|65|

14159/H 27831/L            24001# \65/65|  14969/H 19851/L            26001# \65/65|
01431  42243   5  8 23 20  9 |65|65|   03322  34201   4 20 18 11 12 |65|65|
04321  11314   2 22 19 12 10 |65|65|   40303  24013  23  5 16  2 19 |65|65|
42220  44200  25 15 13 11  1 |65|65|   30241  10243  17  1 13 25  9 |65|65|
32104  03133  16 14  7  4 24 |65|65|   14140  13402   7 24 10 21  3 |65|65|
31034  10220  17  6  3 18 21 |65|65|   22114  34201  14 15  8  6 22 |65|65|
```

```
17023/H 19603/L          28001# \65/65|  19030/H   670/L          30001# \65/65|
 03242   31001    4 17 11 21 12 |65|65|   03332   03200    1 19 18 16 11 |65|65|
 31014   42243   20  8  3 10 24 |65|65|   42140   13001   22 14  6 21  2 |65|65|
 43210   43210   25 19 13  7  1 |65|65|   03214   21232    3 17 13  9 23 |65|65|
 03431   10220    2 16 23 18  6 |65|65|   40320   34413   24  5 20 12  4 |65|65|
 20214   34431   14  5 15  9 22 |65|65|   21114   44214   15 10  8  7 25 |65|65|
21250/H 12664/L          32001# \65/65|  23196/H 16658/L          34001# \65/65|
 03232   23140    3 19 12 20 11 |65|65|   12331   20134    8 11 17 19 10 |65|65|
 34310   20030   18 21 16  9  1 |65|65|   40412   11201   22  2 23  6 12 |65|65|
 00244   31231    4  2 13 24 22 |65|65|   04204   44200    5 25 13  1 21 |65|65|
 43101   41442   25 17 10  5  8 |65|65|   23040   34233   14 20  3 24  4 |65|65|
 21214   40312   15  6 14  7 23 |65|65|   31123   01342   16  7  9 15 18 |65|65|
25616/H  9803/L          36001# \65/65|  27975/H 12121/L          38001# \65/65|
 12431   11102    7 12 22 16  8 |65|65|   12141   14244    7 15  8 25 10 |65|65|
 12403   04321    6 15 24  3 17 |65|65|   34201   13330   17 24 14  4  6 |65|65|
 40240   44200   25  5 13 21  1 |65|65|   00244   42220    5  3 13 23 21 |65|65|
 14023   32104    9 23  2 11 20 |65|65|   34201   41113   20 22 12  2  9 |65|65|
 31023   24333   18 10  4 14 19 |65|65|   30323   00203   16  1 18 11 19 |65|65|
30109/H  7473/L          40001# \65/65|  31815/H 17520/L          42001# \65/65|
 13142   11210    7 17  8 22 11 |65|65|   14132   31100    9 22  7 16 11 |65|65|
 30304   04402   16  5 20  1 23 |65|65|   41402   00203   21  6 23  1 14 |65|65|
 42220   33211   24 14 13 12  2 |65|65|   10243   21232    8  2 13 24 18 |65|65|
 04141   24004    3 25  6 21 10 |65|65|   24030   14244   12 25  3 20  5 |65|65|
 20313   43233   15  4 18  9 19 |65|65|   21303   44331   15 10 19  4 17 |65|65|
33610/H  6633/L          44001# \65/65|  35364/H  1092/L          46001# \65/65|
 13042   14320    7 20  4 23 11 |65|65|   11432   02210    6  8 23 17 11 |65|65|
 32330   21011   18 12 16 17  2 |65|65|   24103   14140   12 25  7  5 16 |65|65|
 04204   00244    1 21 13  5 25 |65|65|   40240   11233   22  2 13 24  4 |65|65|
 41121   33432   24  9 10 14  8 |65|65|   14302   40303   10 21 19  1 14 |65|65|
 20413   42103   15  3 22  6 19 |65|65|   21033   43224   15  9  3 18 20 |65|65|
37037/H 24834/L          48001# \65/65|
 11332   41410   10  7 20 17 11 |65|65|
 04140   44232    5 25  8 24  3 |65|65|
 24202   11233   12 22 13  4 14 |65|65|
 40304   21200   23  2 18  1 21 |65|65|
 21133   43030   15  9  6 19 16 |65|65|   [標準解の総数 = 48544]  [OK!]
** Calculated and listed by Kanji Setsuda on Jan. 8, 2012 with MacOSX 10.7.2 & Xcode 4.2.1 **
```

ようやく総数48544個を網羅することができた！　試みた甲斐があった！
しかし，解の出力スタイルがあまり芳しくない。特に順番が狂っていて，見苦しい。
データを綺麗に並べ替えて出力した「清書版」が，次のリストだ。

**** 清書版：五進法によって作った補数対称五方陣の標準解のリスト：抜粋 ****

（使用した単位方陣：高位/H 低位/L　解の番号#　サム・チェック:\対角線/|行|列|）

```
 137/H  1901/L             1# \65/65|   2779/H  2051/L           201# \65/65|
04430   01031    1 22 21 19  2 |65|65|  04430   00131    1 21 22 19  2 |65|65|
30213   42432   20  3 15  9 18 |65|65|  32312   03200   16 14 18  6 11 |65|65|
23212   10243   12 16 13 10 14 |65|65|  10243   32221    9  3 13 23 17 |65|65|
13241   21020    8 17 11 23  6 |65|65|  23121   44214   15 20  8 12 10 |65|65|
41004   31434   24  7  5  4 25 |65|65|  41004   31344   24  7  4  5 25 |65|65|
 351/H  3197/L           401# \65/65|   2200/H  3110/L           601# \65/65|
04330   03322    1 24 19 18  3 |65|65|  04240   03412    1 24 15 22  3 |65|65|
30124   43300   20  4  9 11 21 |65|65|  21431   12011   12  8 21 17  7 |65|65|
32221   03214   16 14 13 12 10 |65|65|  13213   44200   10 20 13  6 16 |65|65|
02341   44110    5 15 17 22  6 |65|65|  31032   33423   19  9  5 18 14 |65|65|
41104   22114   23  8  7  2 25 |65|65|  40204   23014   23  4 11  2 25 |65|65|
```

2774/H	2991/L					801#	\65/65|	5351/H	2920/L					1001#	\65/65|
04430	01022	1	22	21	18	3	|65|65|		03340	04312	1	20	19	22	3	|65|65|
12213	41433	10	12	15	9	19	|65|65|		03232	41023	5	17	11	18	14	|65|65|
41230	30241	24	6	13	20	2	|65|65|		43210	30241	24	16	13	10	2	|65|65|
13223	11030	7	17	11	14	16	|65|65|		21214	12430	12	8	15	9	21	|65|65|
41004	22434	23	8	5	4	25	|65|65|		40114	23104	23	4	7	6	25	|65|65|

6033/H	2755/L					1201#	\65/65|	995/H	4906/L					1401#	\65/65|
03430	04312	1	20	24	17	3	|65|65|		04420	00343	1	21	24	15	4	|65|65|
14221	10144	7	21	12	15	10	|65|65|		21313	34201	14	10	18	6	17	|65|65|
30241	23212	18	4	13	22	8	|65|65|		30241	32221	19	3	13	23	7	|65|65|
32203	00343	16	11	14	5	19	|65|65|		13132	34201	9	20	8	16	12	|65|65|
41014	23104	23	9	2	6	25	|65|65|		42004	10144	22	11	2	5	25	|65|65|

5583/H	3895/L					1601#	\65/65|	462/H	5622/L					1801#	\65/65|
04330	02413	1	23	20	17	4	|65|65|		03430	01324	1	17	24	18	5	|65|65|
03241	40132	5	16	12	24	8	|65|65|		30322	32401	19	3	20	11	12	|65|65|
32221	34201	19	15	13	11	7	|65|65|		14203	41230	10	22	13	4	16	|65|65|
30214	21340	18	2	14	10	21	|65|65|		22141	34021	14	15	6	23	7	|65|65|
41104	13024	22	9	6	3	25	|65|65|		41014	02134	21	8	2	9	25	|65|65|

1458/H	5910/L					2001#	\65/65|	5786/H	5429/L					2501#	\65/65|
02440	01324	1	12	24	23	5	|65|65|		03430	00244	1	16	23	20	5	|65|65|
21313	43012	15	9	16	7	18	|65|65|		24311	01112	11	22	17	7	8	|65|65|
34201	41230	20	22	13	4	6	|65|65|		20242	31231	14	2	13	24	12	|65|65|
13132	23410	8	19	10	17	11	|65|65|		33102	23334	18	19	9	4	15	|65|65|
40024	02134	21	3	2	14	25	|65|65|		41014	00244	21	6	3	10	25	|65|65|

13045/H	604/L					3001#	\65/65|	10833/H	1364/L					3501#	\65/65|
04331	02030	1	23	16	19	6	|65|65|		04411	03241	1	24	23	10	7	|65|65|
03240	42333	5	18	14	24	4	|65|65|		12430	30043	9	11	21	20	4	|65|65|
32221	14203	17	15	13	11	9	|65|65|		21232	32221	14	8	13	18	12	|65|65|
40214	11120	22	2	12	8	21	|65|65|		41023	10441	22	6	5	15	17	|65|65|
31104	41424	20	7	10	3	25	|65|65|		33004	30214	19	16	3	2	25	|65|65|

6479/H	3174/L					4001#	\65/65|	9127/H	4292/L					5001#	\65/65|
04231	03142	1	24	12	20	8	|65|65|		02431	03133	1	14	22	19	9	|65|65|
40303	12043	22	3	16	5	19	|65|65|		41321	32000	24	8	16	11	6	|65|65|
32221	10243	17	11	13	15	9	|65|65|		04204	20242	3	21	13	5	23	|65|65|
14140	10423	7	21	10	23	4	|65|65|		32130	44421	20	15	10	18	2	|65|65|
31204	20314	18	6	14	2	25	|65|65|		31024	11314	17	7	4	12	25	|65|65|

11230/H	6178/L					6001#	\65/65|	21783/H	126/L					7001#	\65/65|
04411	03124	1	24	22	8	10	|65|65|		01442	02210	1	8	23	22	11	|65|65|
12241	34120	9	15	12	23	6	|65|65|		14320	30331	9	21	19	14	2	|65|65|
30241	34201	19	5	13	21	7	|65|65|		33211	04204	16	20	13	6	10	|65|65|
30223	42301	20	3	14	11	17	|65|65|		42103	31141	24	12	7	5	17	|65|65|
33004	02314	16	18	4	2	25	|65|65|		20034	43224	15	4	3	18	25	|65|65|

3047/H	9552/L					8001#	\65/65|	1235/H	11188/L					9001#	\65/65|
03440	11102	2	17	22	21	3	|65|65|		03420	13434	2	19	25	14	5	|65|65|
32311	43021	20	14	16	8	7	|65|65|		31214	10022	17	6	11	8	23	|65|65|
02224	00244	1	11	13	15	25	|65|65|		41230	14203	22	10	13	16	4	|65|65|
33121	32410	19	18	10	12	6	|65|65|		03231	22443	3	18	15	20	9	|65|65|
40014	24333	23	5	4	9	24	|65|65|		42014	01013	21	12	1	7	24	|65|65|

10260/H	6754/L					10001#	\65/65|	13794/H	8083/L					11001#	\65/65|
04331	12020	2	23	16	18	6	|65|65|		01441	14041	2	10	21	25	7	|65|65|
12412	11134	7	12	22	9	15	|65|65|		24310	41122	15	22	17	8	3	|65|65|
40240	44200	25	5	13	21	1	|65|65|		12223	03214	6	14	13	12	20	|65|65|
23023	01333	11	17	4	14	19	|65|65|		43102	22330	23	18	9	4	11	|65|65|
31104	42423	20	8	10	3	24	|65|65|		30034	30403	19	1	5	16	24	|65|65|

7644/H	10383/L					12001#	\65/65\	12200/H	11723/L					13001#	\65/65\
04321	14203	2	25	18	11	9	|65|65|	00441	14244	2	5	23	25	10	|65|65|
30134	32000	19	3	6	16	21	|65|65|	23312	12110	12	18	17	7	11	|65|65|
42220	11233	22	12	13	14	4	|65|65|	34201	41230	20	22	13	4	6	|65|65|
01341	44421	5	10	20	23	7	|65|65|	23112	43323	15	19	9	8	14	|65|65|
32104	14203	17	15	8	1	24	|65|65|	30044	00203	16	1	3	21	24	|65|65|
21963/H	7519/L					14001#	\65/65\	5479/H	15706/L					15001#	\65/65\
03232	12340	2	18	14	20	11	|65|65|	02440	21403	3	12	25	21	4	|65|65|
04330	34012	4	25	16	17	3	|65|65|	13231	01432	6	17	15	19	8	|65|65|
41230	01234	21	7	13	19	5	|65|65|	34201	03214	16	24	13	2	10	|65|65|
41104	23401	23	9	10	1	22	|65|65|	31213	21034	18	7	11	9	20	|65|65|
21214	40123	15	6	12	8	24	|65|65|	40024	14032	22	5	1	14	23	|65|65|
7822/H	12878/L					16001#	\65/65\	8821/H	14234/L					17001#	\65/65\
03421	23140	3	19	22	15	6	|65|65|	04411	20341	3	21	24	10	7	|65|65|
20333	11021	12	2	16	18	17	|65|65|	41221	12340	22	8	14	15	6	|65|65|
44200	04204	21	25	13	1	5	|65|65|	01234	03214	1	9	13	17	25	|65|65|
11142	32433	9	8	10	24	14	|65|65|	32230	40123	20	11	12	18	4	|65|65|
32014	40312	20	11	4	7	23	|65|65|	33004	30142	19	16	2	5	23	|65|65|
7377/H	15253/L					18001#	\65/65\	10322/H	15364/L					19001#	\65/65\
03411	24432	3	20	25	9	8	|65|65|	04141	23203	3	24	8	21	9	|65|65|
30124	04141	16	5	7	15	22	|65|65|	42211	40114	25	11	12	7	10	|65|65|
42220	31231	24	12	13	14	2	|65|65|	01234	30241	4	6	13	20	22	|65|65|
02341	30304	4	11	19	21	10	|65|65|	33220	03340	16	19	14	15	1	|65|65|
33014	21002	18	17	1	6	23	|65|65|	30304	14212	17	5	18	2	23	|65|65|
13205/H	16614/L					20001#	\65/65\	1673/H	23372/L					22001#	\65/65\
03141	24144	3	20	7	25	10	|65|65|	04140	32244	4	23	8	25	5	|65|65|
23240	31001	14	17	11	21	2	|65|65|	21421	03331	11	9	24	14	7	|65|65|
14203	21232	8	22	13	4	18	|65|65|	13213	44200	10	20	13	6	16	|65|65|
40212	34431	24	5	15	9	12	|65|65|	32032	31114	19	12	2	17	15	|65|65|
30314	00302	16	1	19	6	23	|65|65|	40304	00221	21	1	18	3	22	|65|65|
7459/H	21102/L					24001#	\65/65\	12327/H	23109/L					26001#	\65/65\
03331	30412	4	16	20	17	8	|65|65|	04141	33004	4	24	6	21	10	|65|65|
10422	14334	7	5	24	14	15	|65|65|	33220	12340	17	18	14	15	1	|65|65|
44200	42220	25	23	13	3	1	|65|65|	01234	21232	3	7	13	19	23	|65|65|
22043	01103	11	12	2	21	19	|65|65|	42211	40123	25	11	12	8	9	|65|65|
31114	23041	18	9	6	10	22	|65|65|	30304	04411	16	5	20	2	22	|65|65|
11296/H	24596/L					28001#	\65/65\	11829/H	26558/L					30001#	\65/65\
04131	44430	5	25	10	19	6	|65|65|	01431	44232	5	10	23	19	8	|65|65|
22140	43233	15	14	8	24	4	|65|65|	23204	31001	14	17	11	1	22	|65|65|
03214	21232	3	17	13	9	23	|65|65|	41230	30241	24	6	13	20	2	|65|65|
40322	11210	22	2	18	12	11	|65|65|	04212	34431	4	25	15	9	12	|65|65|
31304	41000	20	7	16	1	21	|65|65|	31034	21200	18	7	3	16	21	|65|65|
16575/H	24614/L					32001#	\65/65\	23194/H	3169/L					34001#	\65/65\
02142	43440	5	14	10	25	11	|65|65|	12331	04132	6	15	17	19	8	|65|65|
31104	43233	20	9	8	4	24	|65|65|	40014	42441	25	3	5	10	22	|65|65|
43210	23212	23	19	13	7	3	|65|65|	24202	13213	12	24	13	2	14	|65|65|
04331	11210	2	22	18	17	6	|65|65|	03440	30020	4	16	21	23	1	|65|65|
20324	40010	15	1	16	12	21	|65|65|	31123	21304	18	7	9	11	20	|65|65|
33306/H	214/L					36001#	\65/65\	24411/H	10624/L					38001#	\65/65\
10432	04420	6	5	25	18	11	|65|65|	12411	14343	7	15	24	10	9	|65|65|
42301	21313	23	12	19	2	9	|65|65|	30241	43102	20	4	12	21	8	|65|65|
03214	30241	4	16	13	10	22	|65|65|	04204	24202	3	25	13	1	23	|65|65|
34120	13132	17	24	7	14	3	|65|65|	30241	24310	18	5	14	22	6	|65|65|
21043	42004	15	8	1	21	20	|65|65|	33023	10103	17	16	2	11	19	|65|65|

（使用した単位方陣: 高位/H 低位/L　　解の番号#　サム・チェック:\対角線/|行|列|）

```
34396/H  7061/L          40001# \65/65|  23961/H 17232/L          42001# \65/65|
 10432   11440    7  2 25 20 11 |65|65|    13321   23104    8 19 17 11 10 |65|65|
 13410   42234   10 18 23  9  5 |65|65|    20430   33441   14  4 25 20  2 |65|65|
 24202   11233   12 22 13  4 14 |65|65|    04204   20242    3 21 13  5 23 |65|65|
 43013   01220   21 17  3  8 16 |65|65|    41042   30011   24  6  1 22 12 |65|65|
 21043   40033   15  6  1 24 19 |65|65|    32113   04312   16 15  9  7 18 |65|65|
36287/H 13946/L          44001# \65/65|  30988/H 19776/L          46001# \65/65|
 14032   23041    8 24  1 20 12 |65|65|    14302   32311    9 23 19  2 12 |65|65|
 24031   41401   15 22  5 16  7 |65|65|    20430   03444   11  4 25 20  5 |65|65|
 10243   32221    9  3 13 23 17 |65|65|    11233   42220   10  8 13 18 16 |65|65|
 31402   34030   19 10 21  4 11 |65|65|    41042   00014   21  6  1 22 15 |65|65|
 21403   30412   14  6 25  2 18 |65|65|    24103   33121   14 24  7  3 17 |65|65|
36269/H 26007/L          48001# \65/65|
 14022   42441   10 23  5 15 12 |65|65|
 04131   33342    4 24  9 20  8 |65|65|
 30241   30241   19  1 13 25  7 |65|65|
 31304   20111   18  6 17  2 22 |65|65|
 22403   30020   14 11 21  3 16 |65|65|   [標準解の総数 = 48544]   [OK!]
** Calculated and listed by Kanji Setsuda on Jan. 8, 2012 with MacOSX 10.7.2 & Xcode 4.2.1 **
```

ついでにこのリストを通常の方法で作った古いリストの解と１つ１つを比較照合してみた。完全に一致したことを報告する。

６．この成功には，どんな意味があるのだろうか。

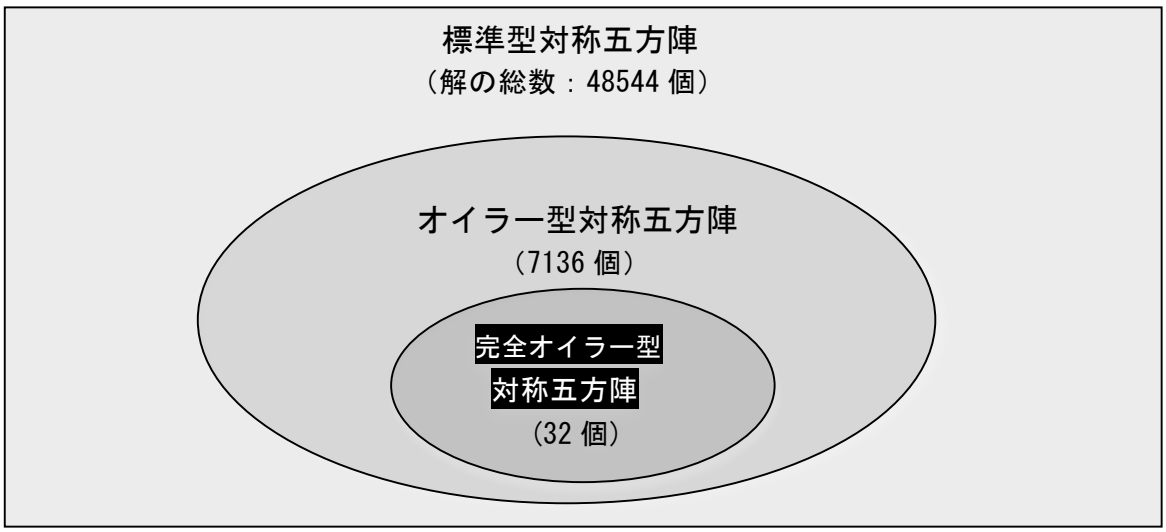
我々の新しい方法は，果たしてラテン方陣の時のような「美しくて，速い」ものだったか。それが，どうもそうではない。新しい「仮定」からして，複雑だ。場合分けが多い。高速でもない。途中で大きく膨らんで，余計なものを一杯抱え込んでいる感じがある。

しかし，余計な解を出力したわけではない。最終的には，標準解の総数 48544 個をドンピシャリで出して来ている。途中の条件を少し厳しくすると，全数を出力することは無くなる。それだからこそ，このドロくさい全過程が必要だったと，私は考えている。

では，メリットは何だったのか。

先ず五進法分解と構成の正逆双方向で，補数対称型五方陣の特性が確かめられたことが大きいと思う。単位方陣の存在は，本質的・運命的なもので，恣意的な仮定ではない。

そして，これが対称五方陣の**構造研究**に役立つと，私は考える。構成過程自体が，対象方陣がどういうふうに出来ているのかを，わかりやすく説明していたと思う。



三種類の解の集合どうしの関係には，上図のような包摂関係を仮定して良いと思う。

「オイラー型」と「完全オイラー型」とを分けたのは，新しい定義に従ったためだ。
単位方陣（別名オイラー・ユニット）と方陣全体との間に構造的な相似性があるものの全部を『**オイラー型**』と呼び，その中でも単位方陣があの美しい「ラテン条件」に従う場合にのみ『**完全オイラー型**』と呼ぶことにしたのである（2012 年定義）。
全てはその構造の故であり，必然的なものを感じて，私はそう再定義した。
そう考え直してみると，先に求めた 7136 個の解を持つ「オイラー型」の対称五方陣にも重要な意味があったと言える。単なる「中間点・通過点」では決してなかった。
48544 個の「対称五方陣」は，規則的な対応関係を持つものとそうでないものを併せ持った全体だ。混沌とした感じがあって，大変に研究し難い。
もう一度この研究をやりたいかと尋ねられたら，私は「否」と答えるだろう。この作業は，コンピューターにとっても大変だったが，私にとっても大変だった。
二度とやりたくないと正直思う。しかし，二度とやりたくはないが，一度は必ずやっておくべき仕事だったと，今では思っている。

魔方陣研究の世界にはそうした必須事項が，本来いろいろあるものだ。
その一つに，標準型五方陣の標準解を五進法で作る課題がまだ残っている。
しかし，これこそ本当にやりたくない仕事だ。
やるべきことの想像は付いても，手間の煩雑さと消費時間と使用メモリーがそれこそ半端なスケールではない。何しろ対象となる標準解が，２億 7530 万 5224 個もあるのだ。
作るべき単位方陣も，何千万個のレベルを超えるだろう。その自乗個ある「場合の数」をこなして構成・検定をするのは大変^{2乗}な作業だ。とても私の手に負える仕事ではない。それに私には，自由に使えるスーパー・コンピューターの持ち合わせが無い。
しかも，努力に対する報いが少ない。幾ら頑張っても，これは「新発見」とは言われない。本来が検証作業の一つに過ぎないからだ。

六次・七次 … と次数を上げてゆくと，ますます「不可能」の世界に踏み込む。どうやっても「五里霧中」から抜け出せなくなる。
もはや誰にも「標準型」の全貌は明かせない。事実上不可能なのだ。
高次方陣の世界では，従って，必然的に「珍しい方陣」を求める方向に進む。美しくて，希有の，珠玉のようなタイプの特別な方陣を何とか作って報告せざるを得ない。
そんな時に役立つのが，「ギリシャ・ラテン方陣」構成法だ。
例の「汎対角線型五方陣」3600 個と「連立型対称汎五方陣」16 個という２種類の『鈴木方陣』については，既に報告した。これらは，通常の方法で求めても，五進法を使って求めても，全く同じ解集合が得られる。上図のような包摂関係ではなく，全てが同値となるのである。五進法を使うと，すばやく成果が得られる。
四方陣，八方陣，九方陣では，二進法・三進法を使う方法が速い。… 十一次・十三次ともなると，さすがに時間を取られる。事実上「桂馬跳び構成法」しか使えない。
そして，この桂馬跳び法こそは，実は素数次の方陣作成の時に使えるギリシャ・ラテン方陣構成法のヴァリエーションの一つだ。手順は少し複雑だが，極めて高速だ。

(Written on Jan. 8 and Apr. 4, 2012 with MacOSX 10.7.3 & Xcode 4.2.1

by Kanji Setsuda: E-Mail Address <jag12001@nifty.com>
http://homepage2.nifty.com/KanjiSetsuda/)