รายการอ้างอิง

ภาษาไทย

มานะ ศรียุทธศักดิ์, เลอศักดิ์ พร้อมสงฆ์, "ผลของก๊าซพาหะที่มีต่อการตอบสนองของหัวตรวจวัดก๊าซจากสารกึ่งตัวนำ", การประชุมวิชาการวิศวกรรมไฟฟ้า ครั้งที่ 15, ภาควิชาวิศวกรรมไฟฟ้า สถาบันอุดมศึกษา, สถาบันเทคโนโลยีพระจอมเกล้าธนบุรี, 3-4 ธันวาคม 2535

ภาษาอังกฤษ

Anonymous., "Figaro Gas Sensor", Figaro Engineering Inc., 1993.

Bart Kosko, "Neural Networks and Fuzzy Systems", Prentice Hall,1992.

Cios. K. J., K. Chen and R. A. Langenderfer., "Use of neural networks in detecting cardiac disease from echocardiographic images", IEEE Eng. in Medicie & Biology Magazine.9(3), pp.58-60, 1990.

Don R. Hush and Bill G. Horne, "Progress in Supervised Neural Networks", IEEE Signal Processing Magazine,pp. 8-38, 1993.

Fukuda T.,T Shibata .,M Sato. and F Harashima., "Neuromorphic Control", IEEE Industrial Electronics, Vol 39 , No. 6 ,1992.

Hakim. P. S. and W. J. Tompkins., "Quantitative Investigation of QRS Detection Rules Using the MIT/BTH Arrhythmia Database", IEEE Trans. Biomed. Eng., BME-33, pp. 1157-1165,1986.

Hiraiwa. A., K. Shimohara and Y. Tokunaga.,"EEG topography recognition by neural networks" ,IEEE Eng. in Medicie & Biology Magazine.9(3), pp. 39-42, 1990.

Iwata. A., Y. Nagasaka and N. Suzumura., "Data compression of the ECG using neural network for digital holter monitor", IEEE Eng. in Medicie & Biology Magazine.9(3), pp. 53-57, 1990.

Jacek M. Zurada, "Introduction to Artificial Neural Systems", West Info. Access, 1992

Jansen. B. H., "Artificial neural nets for K-Complex detection", IEEE Eng. in Medicie & Biology Magazine.9(3), pp. 50-52, 1990.

John Hertz,Anders Krogh and Richard G.Palmer, "Introduction to the theory of neural computation", Addison Wesley Pulishing Company, 1991.

Khoshaba. T., K. Badie and R.M. Hashemi., "EMG pattern classification based on back-propagation neural for prosthesis control", Proc. Annual Intl. Conf. IEEE Eng. Med. & Biol. Soc., pp. 1474-1475, 1990.

Nanto H.,Kawai T. and Tsubakino S., "Gas Discrimination Using ZnO Thin Film Gas Sensor in Conjunction with Neural Network Pattern Recognition", Solid State Sensors and Actuators, 1993.

Paul M. Embree and Bruce Kimble, "C language algorithms for digital signal processing", Prentice Hall,1991.

Qiuzhen Xue, "Biomedical signal processing and Pattern recognition by Artificial Neural Networks",University of Wisconsin Madison.,1991.

R.P. Lippmann, "An introduction to computing with neural networks", IEEE ASSP. Mag,Vol 4,no 2,Apr 1987..

Robert S. Scalero and Nazif Tepedelenlioglu, "A Fast New Algorithm for Training Feedforward Neural Networks", IEEE Trans. on signal processing, Vol 40, no 1 ,1992.

Russell C.Eberhart and Roy W.Dobbins, "Neural Network PC Tools A practical Guide", Acedemic Press. Ino. Ma.,1990.

Vogia. M. J. and E. Micheli-Tzanakou, "Neural network modeling of the visual system", Proc. Annual Intl. Conf. IEEE Eng. Med. & Biol. Soc., pp. 1425-1426, 1990.

Yabuta T. and T Yamada., "Neural Network Controller Characteristics with Regard to Adaptive Control", IEEE System, Man ,and Cybernetics,Vol 22,No.1,1992.

Yeap. T. H.,F. Johnson and M. Rachniowski., "ECG beat classfication by a neural network", Proc. Annual Intl. Conf. IEEE Eng. Med. & Biol. Soc., pp. 1457-1458, 1990.

Yoh-Han Pao, "Adaptive Pattern Recognition and Neural Networks", Addison-Wesley Publishing Company. Inc., 1989.

ภาคผนวก

ภาคผนวก ก

โปรแกรมการวัดสัญญาณจากระบบตรวจวัดก๊าซ

โปรแกรมการวัดสัญญาณจากระบบตรวจวัดก๊าซ แบ่งออกเป็น 3 ส่วนคือ

1. ไฟล์ MATOD.H และ MENU.CPP เป็นไฟล์สำหรับเชื่อมทุกไฟล์เข้าด้วยและเป็นไฟล์เมนู

2. ไฟล์ ATOD.CPP เป็นไฟล์การเก็บข้อมูลจากสัญญาณที่วัดและการนำข้อมูลที่เก็บไว้มาแสดงบนจอคอมพิวเตอร์

3. ไฟล์ DATOD1.CPP เป็นไฟล์ชุดข้อมูลสำหรับเข้าระบบนิวรอลเน็ตเวอร์ก

1. ไฟล์ MATOD.H และ MENU.CPP

1.1 MATOD.H

```
#include <dos.h>
#include <graphics.h>
#include <bios.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <iostream.h>
#include <bcd.h>
#include <ctype.h>
#include <mem.h>
#include <time.h>
#include <dos.h>

typedef struct{   int      maxx, maxy, Xmax, Ymax,ym;
                  int      x,z,y, x1, dx, y1, dy, xd, yd;
                  float    xratio;
                  int      sw1,sw2,sw3,sw4,sw5;
                  int      selectx ,selecty;
                  float    pattern,flow,measure;
                  char     *names1,*names2,*names3,*names4,*names5;
                  float    r1,r2,r3,r4,r5;
                  int      colors1,colors2,colors3,colors4,colors5;

                     }adplot;

extern void initgphc();
extern void bkgraph(adplot *atod,int num);
extern void selectchannel(adplot *atod);
extern void selectxy(adplot *atod);
extern void adcon(adplot *atod);
extern void getdata(adplot *atod,int cens1,int csen2,int csen3,int csen4,int csen5);
```

```
extern void dataplot(adplot *atod,float data,int aa,int csen,int *vsen,int *jsen);
extern void filetonn();
```

1.2 MENU.CPP

```
#include "matod.h"

#define NOT_MOVED 0
#define RIGHT     1
#define LEFT      2
#define UP        3
#define DOWN      4
#define LEFTB 1
#define RIGHTB 2
#define MENU_MAX 20

void initialize();
void mode();
void mouse_position(int *x,int *y);
void set_mouse_position(int x,int y);
void mouse_motion(int *deltax,int *deltay);
void goto_xy(int x,int y);
void cursor_on(),cursor_off(),mouse_reset();
void set_hor(),set_ver();
void wait_on(int button);
void draw_menu(),highlight(int ac,int co);
void status(adplot *atod,int choice,char sm);
int read_mouse();
int read_kb();
menu();
mouse_menu();
void mainmenu(adplot *atod,char sm);
void circuitgraph(adplot *atod,int cx,int cy,int scolor,int sensor);
int  rightb_pressed();
int  leftb_pressed();
int  arrow_choice=0,count=6;
int x = 10,y = 10;

void main()
{
 adplot  *atod;
 int done =0;
 int deltax,deltay;
 char sm;
 atod = (adplot *)calloc(1,sizeof(adplot));

while((sm != 'y') && (sm != 'Y') &&(sm != 'n') && (sm != 'N'))
{
printf("Do you have mouse ?(y/n)");
scanf("%s",&sm);
};
atod->colors1 = 13;
atod->colors2 = 15;
atod->colors3 = 10;
atod->colors4 = 12;
atod->colors5 = 11;
atod->selectx = 2;
atod->pattern = 4;
atod->flow = 0.5;
atod->measure = 1.5;
atod->sw1 = 1;
atod->sw2 = 1;
atod->sw3 = 1;
atod->sw4 = 0;
```

```
atod->sw5 = 0;
atod->names1 = "TGS-800";
atod->names2 = "TGS-813";
atod->names3 = "TGS-822";
atod->names4 = "TGS-823";
atod->names5 = "TGS-824";
atod->r1 = 82;
atod->r2 = 47;
atod->r3 = 47;
atod->r4 = 47;
atod->r5 = 47;

mainmenu(atod,sm);
do{
  if((sm == 'y') || (sm == 'Y'))
  {
    mouse_motion(&deltax,&deltay);
    if(deltax || deltay) read_mouse();
    if(leftb_pressed() || rightb_pressed()) {
    done=  read_mouse();
    if(done)
    status(atod,done,sm);
    }
   }
    if(kbhit()) {
          done = read_kb();
          done +=1;
          if(done)
          status(atod,done,sm);
    }
    }while(done<6);
closegraph();
}
void mainmenu(adplot *atod,char sm)
{
  int i,j,k;
  char pa[10],fl[10],mea[10];

  initialize();
  setbkcolor(3);


  setcolor(15);
  setlinestyle(0,1,1);
  line(0,0,0,getmaxy());
  line(0,getmaxy(),getmaxx(),getmaxy());
  line(0,0,getmaxx(),0);
  line(getmaxx(),0,getmaxx(),getmaxy());
  line(0+5,0+5,0+5,getmaxy()-5);
  line(0+5,getmaxy()-5,getmaxx()-5,getmaxy()-5);
  line(0+5,0+5,getmaxx()-5,0+5);
  line(getmaxx()-5,0+5,getmaxx()-5,getmaxy()-5);
  setcolor(14);
  settextstyle(0,0,5);
  outtextxy(5,15," A/D  converter ");
  settextstyle(0,0,3);
  outtextxy(250,80," for ");
  settextstyle(0,0,4);
  outtextxy(120,140,"  5 Sensor");
  if (atod->sw1 == 1)
  circuitgraph(atod,30,80,atod->colors1,0);
  if (atod->sw2 == 1)
  circuitgraph(atod,450,80,atod->colors2,1);
  if (atod->sw3 == 1)
  circuitgraph(atod,30,240,atod->colors3,2);
  if (atod->sw4 == 1)
```

```
  circuitgraph(atod,450,240,atod->colors4,3);
  if (atod->sw5 == 1)
  circuitgraph(atod,230,350,atod->colors5,4);

  settextstyle(1,0,1);
if (sm == 'y')
{
  mouse_reset();
  set_hor();
  set_ver();
  set_mouse_position(x,y);
};
 draw_menu();
 if (atod->selecty == 1)
 outtextxy(getmaxx()/2+177,getmaxy()/2+140,"Resistance-Y axis");
 else
 outtextxy(getmaxx()/2+200,getmaxy()/2+140,"Voltage-Y axis");
 if (atod->selectx == 1)
 outtextxy(getmaxx()/2+200,getmaxy()/2+160," 10 min-X axis");
 else
 if (atod->selectx == 2)
 outtextxy(getmaxx()/2+200,getmaxy()/2+160," 20 min-X axis");
 else
 outtextxy(getmaxx()/2+200,getmaxy()/2+160," 60 min-X axis");
 outtextxy(getmaxx()/2+176,getmaxy()/2+180," Pattern      min");
 outtextxy(getmaxx()/2+176,getmaxy()/2+200," Flow gas     min");
 outtextxy(getmaxx()/2+176,getmaxy()/2+220," Measure      min");
 gcvt(atod->pattern,4,pa);
 gcvt(atod->flow,4,fl);
 gcvt(atod->measure,4,mea);
 setcolor(4);
 outtextxy(getmaxx()/2+250,getmaxy()/2+180,pa);
 outtextxy(getmaxx()/2+250,getmaxy()/2+200,fl);
 outtextxy(getmaxx()/2+255,getmaxy()/2+220,mea);
}
void circuitgraph(adplot *atod,int cx,int cy,int scolor,int sensor)
{
  char *fname[]  = {"Vi","Vo"};
  char *fname2[] = {"R1","R2","R3","R4","R5"};
  char *fname1 = NULL,*fname3;
  char sty[10],sty2[10],sty3[10];
  fname1 = (char *) calloc(20,sizeof(char));

  setfillstyle(1,scolor);
  bar(cx+95,cy-22,cx+115,cy-12);
  setcolor(12);
  settextstyle(0,0,1);
  strcpy(sty,fname[0]);
  outtextxy(cx-22,cy+45,sty);
  strcpy(sty,fname[1]);

  outtextxy(cx+162,cy+45,sty);
  if(sensor == 0)
  {
  strcpy(fname1,atod->names1);

  gcvt(atod->r1,5,sty3);
  }
  if(sensor == 1)
  {
  strcpy(fname1,atod->names2);
  gcvt(atod->r2,5,sty3);
  }
  if(sensor == 2)
  {
  strcpy(fname1,atod->names3);
```

```
    gcvt(atod->r3,5,sty3);
    }
    if(sensor == 3)
    {
    strcpy(fname1,atod->names4);
    gcvt(atod->r4,5,sty3);
    }
    if(sensor == 4)
    {
    strcpy(fname1,atod->names5);
    gcvt(atod->r5,5,sty3);
    }

    outtextxy(cx+30,cy-20,fname1 );
    outtextxy(cx+56,cy-2,"Rs");
    strcpy(sty2,fname2[sensor]);
    outtextxy(cx+70,cy+45,sty2);
    fname3 = strcat(sty3,"K-ohm");
    outtextxy(cx+50,cy+55,fname3);

    line(cx,cy,cx+50,cy);
    line(cx,cy+100,cx+170,cy+100);
    circle(cx+62,cy,12);
    circle(cx+173,cy,3);
    circle(cx+173,cy+100,3);
    line(cx+74,cy,cx+170,cy);

    line(cx,cy,cx,cy+40);
    line(cx,cy+55,cx,cy+100);
    line(cx-5,cy+40,cx+5,cy+40);
    line(cx-3,cy+45,cx+3,cy+45); /* battery */
    line(cx-5,cy+50,cx+5,cy+50);
    line(cx-3,cy+55,cx+3,cy+55);

    line(cx+110,cy+30,cx+113,cy+35);
    line(cx+107,cy+40,cx+113,cy+35);
    line(cx+107,cy+40,cx+113,cy+45);
    line(cx+107,cy+50,cx+113,cy+45);
    line(cx+107,cy+50,cx+113,cy+55); /* resistance */
    line(cx+107,cy+60,cx+113,cy+55);
    line(cx+107,cy+60,cx+113,cy+65);
    line(cx+110,cy+70,cx+113,cy+65);
    line(cx+110,cy,cx+110,cy+30);
    line(cx+110,cy+70,cx+110,cy+100);
    free(fname1);
};

void  status(adplot *atod,int choice,char sm)
{

      switch(choice){
          case 1:
                    selectchannel(atod);
                    mainmenu(atod,sm);
                    break;
          case 2:
                    selectxy(atod);
                    mainmenu(atod,sm);
                    break;
          case 3:
                    adcon(atod);
                    mainmenu(atod,sm);
                    break;
          case 4:
                    getdata(atod,atod->colors1,atod->colors2,
                    atod->colors3,atod->colors4,atod->colors5);
```

```
                mainmenu(atod,sm);
                break;
        case 5:
                filetonn();
                mainmenu(atod,sm);
                break;
        case 6:
                break;


    }
}

int read_mouse()
{
  int choice;
  setcolor(0);
  if(rightb_pressed() || leftb_pressed()) {
    choice = menu();
    return(choice);
  }
return(choice);
  }

int  read_kb()
  {
    union k{
        char c[2];
        int i;

         } c;

   c.i = bioskey(0);
   cursor_off();
   highlight(arrow_choice,0);
   cursor_on();
   if(c.c[0]) {

   /* check for enter or space bar */
   switch(c.c[0]) {
     case '\r' : cursor_off();
                 highlight(arrow_choice,1);
                 cursor_on();
                 return arrow_choice;
     case ' '  :arrow_choice++;
         break;
     case 29   :return -1;
     }
     }
     else { /* check for special key */
        switch(c.c[1]) {
          case 72: arrow_choice--; /* up arrow */
            break;
          case 80: arrow_choice++; /* down arrow */
            break;
     }
     }
     if(arrow_choice==count) arrow_choice=0;
     if(arrow_choice<0) arrow_choice = count-1;

     /* highlight the next selection */
     cursor_off();
     highlight(arrow_choice,1);
     cursor_on();
     return -1;
```

```
   }

void mode(int mode_code)
{
   union REGS r;
   r.h.al = mode_code;
   r.h.ah = 0;
   int86(0x10,&r,&r);
}

/* set cursor on */

menu()
{
   int x,y,choice;
   while(rightb_pressed() || leftb_pressed());
   cursor_on();

   choice = mouse_menu();
   return choice;
}

mouse_menu()
{

   int mousex,mousey;


      while(rightb_pressed() || leftb_pressed());

      mouse_position(&mousex,&mousey);

      if(mousex>=220 && mousex<420){
          cursor_off();
          highlight(arrow_choice,0);

          if(mousey>250 && mousey<262) {
             arrow_choice = 0;
             highlight(arrow_choice,1);
             cursor_on();
             return 1;
          }

          if(mousey>262 && mousey<274) {
             arrow_choice =1;
             highlight(arrow_choice,1);
             cursor_on();
             return 2;
          }
          if(mousey>274 && mousey<286) {
             arrow_choice = 2;
             highlight(arrow_choice,1);
             cursor_on();
             return 3;
          }
          if(mousey>286 && mousey<298) {
             arrow_choice = 3;
             highlight(arrow_choice,1);
             cursor_on();
             return 4;
          }

          if(mousey>298 && mousey<310) {
             arrow_choice = 4;
             highlight(arrow_choice,1);
             cursor_on();
```

```
            return 5;
          }
          if(mousey>310 && mousey<322) {
            arrow_choice = 5;
            highlight(arrow_choice,1);
            cursor_on();
            return 6;
          }

          highlight(arrow_choice,1);
          cursor_on();
          return -1;
      }
      else
      highlight(arrow_choice,1);
      cursor_on();
      return -1;
}

void wait_on(int button)
{
   if(button==LEFTB)
      while(leftb_pressed());
   else
      while(rightb_pressed());
}

void cursor_on()
{
  union REGS r;

  r.x.ax = 1;
  int86(0x33,&r,&r);
}

/* set min and max cursor position  (vertical)*/
void set_ver()
{
  union REGS r;
  r.x.ax = 8;
  r.x.cx = 0;
  r.x.dx = getmaxy()-5;
  int86(0x33,&r,&r);
}

/* set min and max cursor position (horizontal) */
void set_hor()
{
  union REGS r;
  r.x.ax = 7;
  r.x.cx  = 0;
  r.x.dx  = 636;
  int86(0x33,&r,&r);
}

/* hide the cursor */
void cursor_off()
{
  union REGS r;
  r.x.ax = 2;
  int86(0x33,&r,&r);
}

/* get mouse position */
void mouse_position(int *x,int *y)
{
```

```
 union REGS r;
 r.x.ax = 3;
 int86(0x33,&r,&r);
 *x = r.x.cx;
 *y = r.x.dx;
}

/* set position */
void set_mouse_position(int px,int py)
{
   union REGS r;
   r.x.ax = 4;
   r.x.cx = px;
   r.x.dx = py;
   int86(0x33,&r,&r);
}

/*  check if right button is pressed */
int rightb_pressed()
{
   union REGS r;
   r.x.ax = 3;
   int86(0x33,&r,&r);
   if((r.x.bx & 0x0a) == 1) return 1;

}

/*  check if left button is pressed */
int leftb_pressed()
{
   union REGS r;
   r.x.ax = 3;
   int86(0x33,&r,&r);
   if((r.x.bx & 0x01)== 1) return 1;

}

/* Return the direction of travel */
void mouse_motion(int *deltax,int *deltay)
{
   union REGS r;
   int c,d;
   r.x.ax = 11;
   int86(0x33,&r,&r);

   if(0< r.x.cx < 32767) *deltax = RIGHT;
   if(r.x.cx > 32767) *deltax = LEFT;
   if( r.x.cx == 0) *deltax = NOT_MOVED;


   if(0< r.x.dx < 32767) *deltay = DOWN;
   if(r.x.dx > 32767) *deltay = UP;
   if( r.x.dx == 0) *deltay = NOT_MOVED;
}


/* init graphic mode */
void initialize(void)
{
      int errorcode;
      int xasp, yasp;            /*  Used to read the aspect ratio */
      int g_driver,g_mode;

      detectgraph(&g_driver,&g_mode);
      g_mode = VGAHI;
      initgraph( &g_driver, &g_mode, "c:\tc" );
```

```
        errorcode = graphresult();  /* Read result of initialization*/

        if( errorcode != grOk ){              /* Error occured during init       */
            printf(" Graphics System Error: %s\n", grapherrormsg( errorcode ) );
            exit(1);
        }

}

/* initialize mouse */
void mouse_reset()
{
   union REGS r;
   r.x.ax = 0;
   int86(0x33,&r,&r);

   if(r.x.bx != 2) {
      printf(" two-button mouse is required ");
      exit(1);
   }
}
/* send the cursor to the specified x,y position */
void goto_xy(int x,int y)
{
   union REGS r;
   r.h.ah = 2;
   r.h.dl = y;
   r.h.dh = x;
   r.h.bh = 0;
   int86(0x10,&r,&r);
}

void draw_menu()
{
            cursor_off();
            setcolor(14);
            setfillstyle(1,9);
            bar(220,250,420,262);
            rectangle(220,250,420,262);
            outtextxy(224,252,"Select channel");
            bar(220,262,420,274);
            rectangle(220,262,420,274);
            outtextxy(224,264,"Select axis X-Y");
            bar(220,274,420,286);
            rectangle(220,274,420,286);
            outtextxy(224,276,"A/D converter");
            bar(220,286,420,298);
            rectangle(220,286,420,298);
            outtextxy(224,288,"Get data");
            bar(220,298,420,310);
            rectangle(220,298,420,310);
            outtextxy(224,300,"File to Network");
            bar(220,310,420,322);
            rectangle(220,310,420,322);
            outtextxy(224,312,"Quit");

            highlight(arrow_choice,1); /* highlight the first selection */
            cursor_on();

}

void   highlight(int ac,int co)
{
            setfillstyle(1,12);
            if(co == 0)
               setfillstyle(1,9);
```

```
            bar(220,250+ac*12,420,262+ac*12);
            setcolor(14);
            rectangle(220,250+ac*12,420,262+ac*12);

            switch(ac) {
            case 0:
                outtextxy(224,252,"Select channel");
                break;
            case 1:
                outtextxy(224,264,"Select axis X-Y");
                break;
            case 2:
                outtextxy(224,276,"A/D converter");
                break;
            case 3:
                outtextxy(224,288,"Get data");
                break;
            case 4:
                outtextxy(224,300,"File to Network");
                break;
            case 5:
                outtextxy(224,312,"Quit");
                break;

            }
}
```

2. ไฟล์ ATOD.CPP

```
#include "matod.h"

#define b0  0X02E0
#define b1  0X02E1
#define b2  0X02E2
#define b3  0X02E3
#define b8  0X02E8
#define b12 0X02EC
#define b13 0X02ED
#define b14 0X02EE
#define b15 0X02EF
#define ENTER 13
#define right 50
#define left 50
#define top 80
#define bottom 50
#define origin 0

void initgphc();
void bkgraph(adplot *atod,int num);
void selectchannel(adplot *atod);
void selectxy(adplot *atod);
void adcon(adplot *atod);
void getdata(adplot *atod,int cens1,int csen2,int csen3,int csen4,int csen5);
void dataplot(adplot *atod,float data,int aa,int csen,int *vsen,int *jsen);

void adcon(adplot *atod)
{
  char     *fname,str[20];              /* file name */
  float    *dataf1,*dataf2,*dataf3,*dataf4,*dataf5;     /* data file */
  int      i,num;                       /* point number */
  long     ptime,ntime,prtime;          /* bios delay */
  time_t   first,second;
  int      timecarry,carry = 0;
```

```
int       lock = 1,pattern,off,on;          /* control valves  */
int       selectc;                  /* select channels  */
int       v1,v2,v3,v4;               /* A/D data */
int       data1,data2,data3,data4,data5;
float     vf,jf;
int       vi1,j1,vi2,j2,vi3,j3,vi4,j4,vi5,j5;
int       vi11,j11,vi21,j21,vi31,j31,vi41,j41,vi51,j51;
int       key,dkey;                 /* check key */
int       savei,stopi;
int       range;                    /* maximum voltage */
int       measuretime;               /* measure time 10 min or 20 min */
int yym;                            /* graph */
float xr;
FILE      *fsen;


          initgphc();
          setfillstyle(SOLID_FILL,0);
          bar(0,0,getmaxx(),getmaxy());
          gotoxy(2,2);
          printf ("Input File name to be saved : ");
          scanf ("%s",&str);
          fname = strcat(str,".dat");

          if ((fsen = fopen(fname,"w")) == NULL)
              {
                fprintf(stderr,"\nError! Can not open file! ");
              }
          range = 10;

          outportb(b15, 0X34);    /*  Initialize A/D  converter */
          outportb(b12, 0X36);
          outportb(b12, 0X00);
          outportb(b3,0X20);
          outportb(b8,  0X00);

          num = 2000;
          atod->selecty = 2;
          bkgraph(atod,num);
          yym = atod->ym;
          xr = atod->xratio;
          dataf1 = (float *)calloc(num,sizeof(float));
          dataf2 = (float *)calloc(num,sizeof(float));
          dataf3 = (float *)calloc(num,sizeof(float));
          dataf4 = (float *)calloc(num,sizeof(float));
          dataf5 = (float *)calloc(num,sizeof(float));

          ptime = biostime(0,0L);
          first = time(NULL);
          i = 0;

          while (i < num)
          {
     /* delay(574.5);*/
          second = time(NULL);
          ntime = biostime(0,0L);
          prtime =ntime-ptime;

          if (atod->selectx == 1)     /*   10 minute / 2000 point  */
          {
          pattern = (int) atod->pattern*200;
          on  = (int) atod->measure*200;
          off = (int) atod->flow*200;
          if (carry == 0)
          {
```

```
            timecarry = 5;
            };
            if(carry == 1 )
            {
            timecarry = 6;
            };
            }else            /*   20 minute / 2000 point   */
            if (atod->selectx == 2)
            {
            pattern = (int) atod->pattern*100;
            on  = (int) atod->measure*100;
            off = (int) atod->flow*100;
            if (carry == 0)
            {
            timecarry = 11;
             };
            if(carry == 7 )
            {
            timecarry = 10;
            };
            }else          /*   60 minute / 2000 point   */
            {
            pattern = (int) atod->pattern*100/3;
            on  = (int) atod->measure*100/3;
            off = (int) atod->flow*100/3;
            if (carry == 0)
            {
            timecarry = 32;
            };
            if (carry == 1)
            {
            timecarry = 33;
            };
            }

if  (prtime >= timecarry)
 {
  i++;
  carry++;

  if (lock == 1)              /*  if lock = 0 ->Sensor scanning */
   {
   if (i == on)               /*  measure    */
    {
     outportb(b8,0X80);
     on = on + pattern;
    };
   if (i == off)
    {
     outportb(b8,0X00);        /*  flow gas */
     off = off + pattern;
    };
};

if (atod->sw1 == 1)
        {
          outportb(b2 , 0X07);
          outportb(b0,0);
          while((inportb(b3) & 0X80) != 0);
          v1 = inportb(b1);
          v2 = inportb(b0);
          v3 = v1<<4;
          v4 = v2>>4;
          data1  = v3+ v4 ;
          dataf1[i] = (float)data1;
```

```
            dataf1[i] = (dataf1[i] - origin)*range/4096.0;
            vf = (float) data1*yym/4096.0;
            vf = (float)yym - vf;
            vf = (float)vf +top;
            vi1 = (int)vf - 1;
            setfillstyle(1,atod->colors1);
            bar(300,20, 320,30);
            gotoxy(4,2);
   printf("%s Data No.= %d,V1 = %.3f ",atod->names1,i,dataf1[i]);
            jf =  (float) i* xr + left;
            j1  =  (int) jf;
            setcolor(atod->colors1);
            setlinestyle(0,0,1);
            if (i <= 1)
            {
            line(j1,vi1,j1,vi1);
            }else
            line(j11,vi11,j1,vi1);
            j11 = j1;
            vi11 = vi1;
            ptime = ntime;
}
if (atod->sw2 == 1)
          {
            outportb(b2 , 0X08);
            outportb(b0,0);
            while((inportb(b3) & 0X80) != 0);
            v1 = inportb(b1);
            v2 = inportb(b0);
            v3 = v1<<4;
            v4 = v2>>4;
            data2  = v3+ v4 ;
            dataf2[i] = (float)data2;
            dataf2[i] = (dataf2[i] - origin)*range/4096.0;
            vf = (float) data2*yym/4096.0;
            vf = (float)yym - vf;
            vf = (float)vf +top;
            vi2 = (int)vf - 1;
            setfillstyle(1,atod->colors2);
            bar(300,35, 320,45);
            gotoxy(4,3);
   printf("%s Data No.= %d,V2 = %.3f ",atod->names2,i,dataf2[i]);
            jf =  (float) i* xr + left;
            j2  =  (int) jf;
            setcolor(atod->colors2);
            setlinestyle(0,0,1);
            if (i <= 1)
            {
            line(j2,vi2,j2,vi2);
            }else
            line(j21,vi21,j2,vi2);
            j21 = j2;
            vi21 = vi2;
            ptime = ntime;
}
if(atod->sw3 == 1)
          {
            outportb(b2 , 0X09);
            outportb(b0,0);
            while((inportb(b3) & 0X80) != 0);
            v1 = inportb(b1);
            v2 = inportb(b0);
            v3 = v1<<4;
            v4 = v2>>4;
            data3  = v3+ v4 ;
            dataf3[i] = (float)data3;
```

```
        dataf3[i] = (dataf3[i] - origin)*range/4096.0;
        vf = (float) data3*yym/4096.0;
        vf = (float)yym - vf;
        vf = (float)vf +top;
        vi3 = (int)vf - 1;
        setfillstyle(1,atod->colors3);
        bar(300,50, 320,60);
        gotoxy(4,4);
  printf("%s Data No.= %d,V3 = %.3f ",atod->names3,i,dataf3[i]);
        jf =  (float) i* xr + left;
        j3  =  (int) jf;
        setcolor(atod->colors3);
        setlinestyle(0,0,1);
        if (i <= 1)
        {
        line(j3,vi3,j3,vi3);
        }else
        line(j31,vi31,j3,vi3);
        j31 = j3;
        vi31 = vi3;
        ptime = ntime;
}
if(atod->sw4 == 1)
        {
        outportb(b2 , 0X0A);
        outportb(b0,0);
        while((inportb(b3) & 0X80) != 0);
        v1 = inportb(b1);
        v2 = inportb(b0);
        v3 = v1<<4;
        v4 = v2>>4;
        data4  = v3+ v4 ;
        dataf4[i] = (float)data4;
        dataf4[i] = (dataf4[i] - origin)*range/4096.0;
        vf = (float) data4*yym/4096.0;
        vf = (float)yym - vf;
        vf = (float)vf +top;
        vi4 = (int)vf - 1;
        setfillstyle(1,atod->colors4);
        bar(605,20, 625,30);
        gotoxy(42,2);
  printf("%s Data No.= %d,V4 = %.3f ",atod->names4,i,dataf4[i]);
        jf =  (float) i* xr + left;
        j4  =  (int) jf;
        setcolor(atod->colors4);
        setlinestyle(0,0,1);
        if (i <= 1)
        {
        line(j4,vi4,j4,vi4);
        }else
        line(j41,vi41,j4,vi4);
        j41 = j4;
        vi41 = vi4;
        ptime = ntime;
}
if(atod->sw5 == 1)
        {
        outportb(b2 , 0X0B);
        outportb(b0,0);
        while((inportb(b3) & 0X80) != 0);
        v1 = inportb(b1);
        v2 = inportb(b0);
        v3 = v1<<4;
        v4 = v2>>4;
        data5  = v3+ v4 ;
        dataf5[i] = (float)data5;
```

```
        dataf5[i] = (dataf5[i] - origin)*range/4096.0;
        vf = (float) data5*yym/4096.0;
        vf = (float)yym - vf;
        vf = (float)vf +top;
        vi5 = (int)vf - 1;
        setfillstyle(1,atod->colors5);
        bar(605,35, 625,45);
        gotoxy(42,3);
 printf("%s Data No.= %d,V5 = %.3f ",atod->names5,i,dataf5[i]);
        jf =  (float) i* xr + left;
        j5  =  (int) jf;
        setcolor(atod->colors5);
        setlinestyle(0,0,1);
        if (i <= 1)
        {
        line(j5,vi5,j5,vi5);
        }else
        line(j51,vi51,j5,vi5);
        j51 = j5;
        vi51 = vi5;
        ptime = ntime;
}

        stopi = num;

        if (bioskey(1) != 0)
        {
          key = bioskey(0);
          key = toascii(key);
          if(key == 's')
          { gotoxy(2,2);
            printf("Sensor scanning");
            outportb(b8,0X00);
            lock = 0;
          }else
          if (key == 'q')
          {
            gotoxy(2,2);
            printf("QUIT TO DOS!   ");
             stopi = i;
             i = num;
          }
          else
          {
           do
             dkey = getch();
             while (dkey != ENTER);
          }
          }
        if (atod->selectx == 1)          /*  delay  */
        {
        if (timecarry == 6)
        {
        carry = 0;
        };
        }else
        if (atod->selectx == 2)
        {
        if (timecarry == 10)
        {
        carry = 0;
        };
        }else
        {
        if (timecarry == 33)
        {
```

```
            carry = 0;
            };
            }
            }
          gotoxy (67,4);
          printf("Time :%.0f s",difftime(second,first));

         }


          fprintf(fsen,"%d ",stopi);
          if (atod->selectx == 1)
          measuretime = 10;
          else
          if (atod->selectx == 2)
          measuretime = 20;
          else
          measuretime = 60;
          fprintf(fsen,"%d\n",measuretime);
fprintf(fsen,"%d %d %d %d %d",atod->sw1,atod->sw2,atod->sw3,atod->sw4,atod->sw5);
          if(atod->sw1 ==1)
          fprintf(fsen,"%s ",atod->names1);
          if(atod->sw2 ==1)
          fprintf(fsen,"%s ",atod->names2);
          if(atod->sw3 ==1)
          fprintf(fsen,"%s ",atod->names3);
          if(atod->sw4 ==1)
          fprintf(fsen,"%s ",atod->names4);
          if(atod->sw5 ==1)
          fprintf(fsen,"%s ",atod->names5);
          fprintf(fsen,"\n");
          for (savei=1; savei<=stopi; savei++)
          {     fprintf(fsen,"\t%d",savei);
          if (atod->sw1 == 1)
                fprintf(fsen,"\t%.3f",dataf1[savei]);
          if (atod->sw2 == 1)
                fprintf(fsen,"\t%.3f",dataf2[savei]);
          if (atod->sw3 == 1)
                fprintf(fsen,"\t%.3f",dataf3[savei]);
          if (atod->sw4 == 1)
                fprintf(fsen,"\t%.3f",dataf4[savei]);
          if (atod->sw5 == 1)
                fprintf(fsen,"\t%.3f",dataf5[savei]);
                fprintf(fsen,"\n");
          }
          getch();
          lock = 1;
          outportb(b8,0X00);
          fclose(fsen);
          closegraph();
          free(dataf1);
          free(dataf2);
          free(dataf3);
          free(dataf4);
          free(dataf5);
          return;
}

void initgphc()
{
  int gd = DETECT,gm,errorcode;

  clrscr();
  detectgraph(&gd,&gm);
  initgraph(&gd,&gm,"");
  errorcode = graphresult();
```

```
        if (errorcode != grOk)
        {
                printf("Graphics error %d\n",errorcode);
                printf("Press any key to halt");
                getch();
                exit(1);
        }
 }
void selectchannel(adplot *atod)
{
int sw1,sw2,sw3,sw4,sw5;
float r1,r2,r3,r4,r5;
char names1[15],names2[15],names3[15],names4[15],names5[15];
char *nams1,*nams2,*nams3,*nams4,*nams5;
atod->names1 = (char *) calloc(20,sizeof(char));
atod->names2 = (char *) calloc(20,sizeof(char));
atod->names3 = (char *) calloc(20,sizeof(char));
atod->names4 = (char *) calloc(20,sizeof(char));
atod->names5 = (char *) calloc(20,sizeof(char));
nams1 = (char *) calloc(20,sizeof(char));
nams2 = (char *) calloc(20,sizeof(char));
nams3 = (char *) calloc(20,sizeof(char));
nams4 = (char *) calloc(20,sizeof(char));
nams5 = (char *) calloc(20,sizeof(char));
initgphc();
do
{
printf("Channel 7 ( off(0)/on(1) (default = on)):");
scanf("%d",&sw1);
atod->sw1 = sw1;
}while((sw1 != 0) && (sw1 != 1));
if (sw1 == 1)
{
printf("Name and number of sensor(default = TGS-800):");
scanf("%s",&names1);
strcpy(nams1,names1);
atod->names1 = nams1;
printf(" Name of sensor is %s\n",atod->names1);
printf("Load resistance((default = 82 K-ohm)K-ohm):");
scanf("%f",&r1);
atod->r1 = r1;
};
do
{
printf("Channel 8 ( off(0)/on(1) (default = on)):");
scanf("%d",&sw2);
atod->sw2 = sw2;
}while((sw2 != 0) && (sw2 != 1));
if (sw2 == 1)
{
printf("Name and number of sensor(default = TGS-813):");
scanf("%s",&names2);
strcpy(nams2,names2);
atod->names2 = nams2;
printf(" Name of sensor is %s\n",atod->names2);
printf("Load resistance((default = 47 K-ohm)K-ohm):");
scanf("%f",&r2);
atod->r2 = r2;
};
do
{
printf("Channel 9 ( off(0)/on(1) (default = on)):");
scanf("%d",&sw3);
atod->sw3 = sw3;
}while((sw3 != 0) && (sw3 != 1));
if (sw3 == 1)
```

```
{
printf("Name and number of sensor(default = TGS-822):");
scanf("%s",&names3);
strcpy(nams3,names3);
atod->names3 = nams3;
printf(" Name of sensor is %s\n",atod->names3);
printf("Load resistance((default = 47 K-ohm)K-ohm):");
scanf("%f",&r3);
atod->r3 = r3;
};
do
{
printf("Channel 10 ( off(0)/on(1) (default = off)):");
scanf("%d",&sw4);
atod->sw4 = sw4;
}while((sw4 != 0) && (sw4 != 1));
if (sw4 == 1)
{
printf("Name and number of sensor(default = TGS-823):");
scanf("%s",&names4);
strcpy(nams4,names4);
atod->names4 = nams4;
printf(" Name of sensor is %s\n",atod->names4);
printf("Load resistance((default = 47 K-ohm)K-ohm):");
scanf("%f",&r4);
atod->r4 = r4;
};
do
{
printf("Channel 11 ( off(0)/on(1) (default = off)):");
scanf("%d",&sw5);
atod->sw5 = sw5;
}while((sw5 != 0) && (sw5 != 1));
if (sw5 == 1)
{
printf("Name and number of sensor(default = TGS-824):");
scanf("%s",&names5);
strcpy(nams5,names5);
atod->names5 = nams5;
printf(" Name of sensor is %s\n",atod->names5);
printf("Load resistance((default = 47 K-ohm)K-ohm):");
scanf("%f",&r5);
atod->r5 = r5;
};
closegraph();
}

void selectxy(adplot *atod)
{
int ssy,ssx;
float pat,flow,measure;
int minx;
initgphc();
printf("Do you want to resistance (1) or voltage (2) on Y axis?(1 or 2):");
scanf("%d",&ssy);
atod->selecty = ssy;
printf("Do you want to  measure 10 min(1),20 min (2)or 60 min(3) on X axis ?(1,2,3):");
scanf("%d",&ssx);
atod->selectx = ssx;
if (ssx == 1)
minx = 10;
else
if (ssx == 2)
minx = 20;
else
minx = 60;
```

```
do{
printf("How many minute per one pattern ?(<= %d minute):",minx);
scanf("%f",&pat);
}while(minx < pat);
atod->pattern = pat;
do{
printf("How many minute to  flow gas?(< %.2f minute):",pat);
scanf("%f",&flow);
}while(pat < flow);
atod->flow = flow;
do{
printf("How many minute to measurement?(< %.2f minute):",pat-flow);
scanf("%f",&measure);
}while(pat-flow < measure);
atod->measure = measure;

closegraph();
}


void bkgraph(adplot *atod,int num)
{
  char stextx[10],stexty[10];
  int  xx = 0,yy = 0,textx = 0,texty,xloop,yloop;
  int  bcol,dkey;

          bcol = 1;                       /*  blue background  */
          if (atod->selecty == 1)
          {
          texty = 100;                    /*  resistance  */
          }
          else
          {
          texty = 10;                     /*  voltage     */
          }
          setbkcolor(bcol);
          atod->maxx = getmaxx();
          atod->maxy = getmaxy();
          atod->Xmax = atod->maxx+1;
          atod->Ymax = atod->maxy+1;
          atod->xratio = (float) (atod->Xmax-left-right)/num;

        /*   Scaling Axis   */

          setcolor(14);
          rectangle(0, 0,atod->maxx,atod->maxy);
          atod->x = atod->maxx-right;
          atod->y = atod->maxy-bottom;
          rectangle(left,top,atod->x,atod->y);
          atod->yd = atod->Ymax-top-bottom;
          atod->dy = atod->yd/10;
          atod->y1 = top;
          atod->ym = atod->Ymax -top -bottom;
          setlinestyle(1,1,1);
          do
          {
             line(left,atod->y1,atod->x,atod->y1);
             atod->y1 = atod->y1+atod->dy;
          }
          while (atod->y1<=atod->y);

          atod->xd = atod->Xmax-left-right;
          atod->dx = atod->xd/10;
          atod->x1 = left;

          do
```

```
            {
                line(atod->x1,top,atod->x1,atod->y);
                atod->x1 = atod->x1+atod->dx;
            }
            while (atod->x1<=atod->x);

            outtextxy(2,2,"Press ENTER key to start");
            atod->y1 = atod->maxy -bottom/2;
            for (xloop = 0;xloop < 6;xloop++)
            {
            itoa(textx,stextx,10);
            outtextxy(left + xx*atod->dx,atod->y1,stextx);
            xx = xx + 2;
            if (atod->selectx == 1)
            textx = textx + 2;
            else
            if (atod->selectx == 2)
            textx = textx + 4;
            else
            textx = textx + 12;

            }
            outtextxy(left+160+6*atod->dx,atod->y1+bottom/4, "Time[min]");

            atod->x1 =left/4;
            for (yloop = 0;yloop < 6;yloop++)
            {
            itoa(texty,stexty,10);
            outtextxy(atod->x1,top + yy*atod->dy, stexty);
            yy = yy + 2;
            if (atod->selecty == 1)
            {
            texty = texty - 20;
            }else
            {
            texty = texty-2;
            }

            }
            if (atod->selecty == 1)
            outtextxy(atod->x1,top*.8, "R[K-ohm]");
            else
            outtextxy(atod->x1,top*.8, "V[volt]");
            do
             dkey = getch();
             while (dkey != ENTER);
             setfillstyle(1,0);
             bar(2, 2, 250, 40);
}

void getdata(adplot *atod,int csen1,int csen2,int csen3,int csen4,int csen5)
{
 char   fdataf[13],*fnamef;
 int    nump,measuretime,nnum,aa;
 float  sensor1,sensor2,sensor3,sensor4,sensor5;
 float  datas1,datas2,datas3,datas4,datas5;
 int    *vsen1,*jsen1,*vsen2,*jsen2,*vsen3,*jsen3,*vsen4,*jsen4,*vsen5,*jsen5;
 int    sw1,sw2,sw3,sw4,sw5;
 char   names1[10],names2[10],names3[10],names4[10],names5[10];

 FILE   *fdata;


 initgphc();
 setfillstyle(SOLID_FILL,0);
 bar(0,0,getmaxx(),getmaxy());
```

```
gotoxy(2,2);
printf("What your data file =");
scanf("%s",&fdataf);
fnamef = strcat(fdataf,".dat");

fdata = fopen(fnamef,"r");

   fscanf(fdata,"%d",&nump);
   fscanf(fdata,"%d",&measuretime);
   fscanf(fdata,"%d",&sw1);
   fscanf(fdata,"%d",&sw2);
   fscanf(fdata,"%d",&sw3);
   fscanf(fdata,"%d",&sw4);
   fscanf(fdata,"%d",&sw5);
   if (measuretime == 10)
   atod->selectx = 1;
   else
   if (measuretime == 20)
   atod->selectx = 2;
   else
   atod->selectx = 3;
         bkgraph(atod,2000);
         if (sw1 == 1)
         {
         fscanf(fdata,"%s",&names1);
         setfillstyle(1,csen1);
         bar(390,20,410,30);
         outtextxy(250,22,names1);
         outtextxy(325,22,"SENSOR1");
         }
         if (sw2 == 1)
         {
         fscanf(fdata,"%s",&names2);
         setfillstyle(1,csen2);
         bar(390,35, 410,45);
         outtextxy(250,37,names2);
         outtextxy(325,37,"SENSOR2");
         }
         if (sw3 == 1)
         {
         fscanf(fdata,"%s",&names3);
         setfillstyle(1,csen3);
         bar(390,50, 410,60);
         outtextxy(250,52,names3);
         outtextxy(325,52,"SENSOR3");
         }
         if (sw4 == 1)
         {
         fscanf(fdata,"%s",&names4);
         setfillstyle(1,csen4);
         bar(570,20, 590,30);
         outtextxy(430,22,names4);
         outtextxy(505,22,"SENSOR4");
         }
         if (sw5 == 1)
         {
         fscanf(fdata,"%s",&names5);
         setfillstyle(1,csen5);
         bar(570,35, 590,45);
         outtextxy(430,37,names5);
         outtextxy(505,37,"SENSOR5");
         }
         vsen1 = (int *)calloc(nump,sizeof(int));
         jsen1 = (int *)calloc(nump,sizeof(int));
         vsen2 = (int *)calloc(nump,sizeof(int));
         jsen2 = (int *)calloc(nump,sizeof(int));
```

```
            vsen3 = (int *)calloc(nump,sizeof(int));
            jsen3 = (int *)calloc(nump,sizeof(int));
            vsen4 = (int *)calloc(nump,sizeof(int));
            jsen4 = (int *)calloc(nump,sizeof(int));
            vsen5 = (int *)calloc(nump,sizeof(int));
            jsen5 = (int *)calloc(nump,sizeof(int));

        for (aa = 0;aa < nump;aa++)
        {
         fscanf(fdata,"%d",&nnum);
         if(sw1 == 1)
         {
         fscanf(fdata,"%f",&sensor1);
         datas1 = (float)sensor1;
         dataplot(atod,datas1,aa,csen1,vsen1,jsen1);
         }
         if(sw2 == 1)
         {
         fscanf(fdata,"%f",&sensor2);
         datas2 = (float)sensor2;
         dataplot(atod,datas2,aa,csen2,vsen2,jsen2);
         }
         if(sw3 == 1)
         {
         fscanf(fdata,"%f",&sensor3);
         datas3 = (float)sensor3;
         dataplot(atod,datas3,aa,csen3,vsen3,jsen3);
         }
         if(sw4 == 1)
         {
         fscanf(fdata,"%f",&sensor4);
         datas4 = (float)sensor4;
         dataplot(atod,datas4,aa,csen4,vsen4,jsen4);
         }
         if(sw5 == 1)
         {
         fscanf(fdata,"%f",&sensor5);
         datas5 = (float)sensor5;
         dataplot(atod,datas5,aa,csen5,vsen5,jsen5);
         }
        }
         getch();
         fclose(fdata);
         closegraph();
         free(vsen1);
         free(jsen1);
         free(vsen2);
         free(jsen2);
         free(vsen3);
         free(jsen3);
         free(vsen4);
         free(jsen4);
         free(vsen5);
         free(jsen5);
         return;
}

void dataplot(adplot *atod,float data,int aa,int csen,int *vsen,int *jsen)
{
   float  maxvolt = 9,resistance;
   float  vs,js;
   int    range,yym,ssy;
   float  xr;

   yym = atod->ym;
   ssy = atod->selecty;
```

```
xr =  atod->xratio;
          if (csen == 13 )
          resistance = atod->r1;
          if (csen == 15)
          resistance = atod->r2;
          if  (csen == 10)
          resistance = atod->r3;
          if (csen == 12)
          resistance = atod->r4;
          if  (csen == 11)
          resistance = atod->r5;
          if (ssy == 1)
          {
          data = ((maxvolt *resistance)/data)-resistance;
          range = 100;
          }else
          {
          range = 10;
          }
          data = (data*4096.0/range)+origin;
          vs = (float) data*yym/4096.0;
          vs = (float)yym - vs;
          vs = (float)vs +top;
          vsen[aa] = (int)vs - 1;
          js =  (float) aa *xr + left;
          jsen[aa]  =  (int) js;
          setcolor(csen);
          setlinestyle(0,0,1);
          if (aa <= 1)
          {
          line(jsen[aa],vsen[aa],jsen[aa],vsen[aa]);
          }else
          line(jsen[aa-1],vsen[aa-1],jsen[aa],vsen[aa]);
          return;
}
```

3. ไฟล์ DATOD1.CPP

```
#include"matod.h"

#define ESC   27
void filetonn();
void filetonn()
{
char   fnamedelta[15] ,*fn;    /* create file to network */
FILE   *fdelta;
char   ftr[15],*fname;         /* read A/D converter */
FILE   *f;
float  timepattern,timestable,timerecover;
int    time,tpat = 400;
int    timemaxm = 400 ,timestam = 195 ,timerecom = 300,tmax,tsta,treco;
float  timemax1,timemax2,timemax3,timemax4,timemax5;
int    i,num,timegraph;
int    sw1,sw2,sw3,sw4,sw5;
char   ns1[10],ns2[10],ns3[10],ns4[10],ns5[10];
int    trte = 0,j,cslut;
int    b0 = 0,b1 = 1;
float  in1,in2,in3,in4,in5;
int    *tim;
float  *input1,*input2,*input3,*input4,*input5;
float  *datmax1,*datmax2,*datmax3,*datmax4,*datmax5;
float  *dat21,*dat22,*dat23,*dat24,*dat25;
float  *dat31,*dat32,*dat33,*dat34,*dat35;
```

```
initgphc();
printf("Filename for save Max value,Stable value,Recover value =  ");
scanf("%s",&fnamedelta);
fn = strcat(fnamedelta,".dat");

tim     = (int *)calloc(3,sizeof(int));
input1  = (float *)calloc(2000,sizeof(float));
input2  = (float *)calloc(2000,sizeof(float));
input3  = (float *)calloc(2000,sizeof(float));
input4  = (float *)calloc(2000,sizeof(float));
input5  = (float *)calloc(2000,sizeof(float));
datmax1 = (float *)calloc(100,sizeof(float));
datmax2 = (float *)calloc(100,sizeof(float));
datmax3 = (float *)calloc(100,sizeof(float));
datmax4 = (float *)calloc(100,sizeof(float));
datmax5 = (float *)calloc(100,sizeof(float));
dat21   = (float *)calloc(100,sizeof(float));
dat22   = (float *)calloc(100,sizeof(float));
dat23   = (float *)calloc(100,sizeof(float));
dat24   = (float *)calloc(100,sizeof(float));
dat25   = (float *)calloc(100,sizeof(float));
dat31   = (float *)calloc(100,sizeof(float));
dat32   = (float *)calloc(100,sizeof(float));
dat33   = (float *)calloc(100,sizeof(float));
dat34   = (float *)calloc(100,sizeof(float));
dat35   = (float *)calloc(100,sizeof(float));

do
{
fdelta = fopen(fn,"a"); if(fdelta == NULL)
 {
  fprintf(stderr,"Can't open file %s \n",fn);
 };

tmax  = 0;
tsta  = 0;
treco = 0;
timemax1 = 0;
timemax2 = 0;
timemax3 = 0;
timemax4 = 0;
timemax5 = 0;

printf("filename =  ");
scanf("%s",&ftr);
fname = strcat(ftr,".dat");

f = fopen(fname,"r"); if(f == NULL)
 {
  fprintf(stderr,"Can't open file %s \n",fname);
 };

 fscanf(f,"%d",&num);
 fscanf(f,"%d",&timegraph);
 fscanf(f,"%d %d %d %d %d",&sw1,&sw2,&sw3,&sw4,&sw5);
 if (sw1 == 1)
 fscanf(f,"%s",&ns1);
 if (sw2 == 1)
 fscanf(f,"%s",&ns2);
 if (sw3 == 1)
 fscanf(f,"%s",&ns3);
 if (sw4 == 1)
 fscanf(f,"%s",&ns4);
 if (sw5 == 1)
 fscanf(f,"%s",&ns5);
 do
```

```
{
printf("\nHow many minute per pattern ?(<= %d):",timegraph);
scanf("%f",&timepattern);
}while(timegraph < timepattern);
do
{
printf("\nHow many minute to stable value ?(<= %.2f):",timepattern);
scanf("%f",&timestable);
}while(timepattern < timestable);
 do
{
printf("\nHow many minute to recover value ?(<= %.2f):",timepattern);
scanf("%f",&timerecover);
}while(timepattern < timerecover);

  if (timegraph == 10)
  {
  tpat = (int) timepattern*200;
  timemaxm = (int) timepattern*200;
  timestam = (int) timestable*200;
  timerecom = (int) timerecover*200;
  }else
  if (timegraph == 20)
  {
  tpat = (int) timepattern*100;
  timemaxm = (int) timepattern*100;
  timestam = (int) timestable*100;
  timerecom = (int) timerecover*100;
  }
  else
  {
  tpat = (int) timepattern*100/3;
  timemaxm = (int) timepattern*100/3;
  timestam = (int) timestable*100/3;
  timerecom = (int) timerecover*100/3;
  }

for(i = 0;i < num;i++)
{

  fscanf(f,"%d",&time);
  *(tim+i) = time;

if (sw1 == 1)
{
  fscanf(f,"%f",&in1);
  *(input1+i) = in1;
   if (input1[i] > timemax1)
 {
  timemax1 = input1[i];

 }
 };
if (sw2 == 1)
{
  fscanf(f,"%f",&in2);
  *(input2+i) = in2;
  if (input2[i] > timemax2)
 {
  timemax2 = input2[i];

 }
 };
if (sw3 == 1)
{
    fscanf(f,"%f",&in3);
```

```
  *(input3+i) = in3;
  if (input3[i] > timemax3)
 {
  timemax3 = input3[i];

 }
 };
if (sw4 == 1)
{
    fscanf(f,"%f",&in4);
  *(input4+i) = in4;
  if (input4[i] > timemax4)
 {
  timemax4 = input4[i];

 }
 };
if (sw5 == 1)
{
    fscanf(f,"%f",&in5);
  *(input5+i) = in5;
  if (input5[i] > timemax5)
 {
  timemax5 = input5[i];

 }
 };
   if ((tim[i] == timemaxm) || (tim[i] == tim[num]))
   {
      if (sw1 == 1)
      datmax1[tmax] = timemax1/10;
      if (sw2 == 1)
      datmax2[tmax] = timemax2/10;
      if (sw3 == 1)
      datmax3[tmax] = timemax3/10;
      if (sw4 == 1)
      datmax4[tmax] = timemax4/10;
      if (sw5 == 1)
      datmax5[tmax] = timemax5/10;
      timemax1 = 0;
      timemax2 = 0;
      timemax3 = 0;
      timemax4 = 0;
      timemax5 = 0;
      timemaxm = timemaxm+tpat;
      tmax = tmax++;

           }
  if (tim[i] == timestam)
 {
      if (sw1 == 1)
      dat21[tsta] = input1[i]/10;
      if (sw2 == 1)
      dat22[tsta] = input2[i]/10;
      if (sw3 == 1)
      dat23[tsta] = input3[i]/10;
      if (sw4 == 1)
      dat24[tsta] = input4[i]/10;
      if (sw5 == 1)
      dat25[tsta] = input5[i]/10;
      timestam = timestam+tpat;
      tsta = tsta++;
 }
   if (tim[i] == timerecom)
 {
      if (sw1 == 1)
```

```
        dat31[treco] = input1[i]/10;
        if (sw2 == 1)
        dat32[treco] = input2[i]/10;
        if (sw3 == 1)
        dat33[treco] = input3[i]/10;
        if (sw4 == 1)
        dat34[treco] = input4[i]/10;
        if (sw5 == 1)
        dat35[treco] = input5[i]/10;
        timerecom = timerecom+tpat;
        treco = treco++;
   }
}

while((trte != 1) && (trte != 2))
{
 printf("Data to 1 Training or 2 Testing(1,2)");
 scanf("%d",&trte);
};
if (trte == 1)
{
 printf("Pattern is 1.water\n");
 printf("            2.EtOH \n");
 printf("            3.Acetone \n");
 printf("            4.Ammonia \n");
 printf("            5.Acetaldehyde \n");
 printf("            6.Solution1 \n");
 printf("            7.Solution2 \n");
}
if (timegraph == 10)
 printf("\n TIME = %d \n",tim[num-1]/200);
else
if (timegraph == 20)
printf("\n TIME = %d \n",tim[num-1]/100);
else
printf("\n TIME = %d \n",tim[num-1]/100/3);

  for(j = 0;j <tmax;j++)
{
 printf("Pattern %d = Max value,Stable value,Recover value\n",j+1);
 if (sw1 == 1)
 {
 printf(" %s  = %.4f , %.4f , %.4f\n",ns1,datmax1[j],dat21[j],dat31[j]);
 fprintf(fdelta,"%.4f %.4f %.4f ",datmax1[j],dat21[j],dat31[j]);
 };
 if (sw2 == 1)
 {
 printf(" %s  = %.4f , %.4f , %.4f\n",ns2,datmax2[j],dat22[j],dat32[j]);
 fprintf(fdelta,"%.4f %.4f %.4f ",datmax2[j],dat22[j],dat32[j]);
 };
 if (sw3 == 1)
 {
 printf(" %s  = %.4f , %.4f , %.4f\n",ns3,datmax3[j],dat23[j],dat33[j]);
 fprintf(fdelta,"%.4f %.4f %.4f ",datmax3[j],dat23[j],dat33[j]);
 };
 if (sw4 == 1)
 {
 printf(" %s  = %.4f , %.4f , %.4f\n",ns4,datmax4[j],dat24[j],dat34[j]);
 fprintf(fdelta,"%.4f %.4f %.4f ",datmax4[j],dat24[j],dat34[j]);
 };
 if (sw5 == 1)
 {
 printf(" %s  = %.4f , %.4f , %.4f\n",ns5,datmax5[j],dat25[j],dat35[j]);
 fprintf(fdelta,"%.4f %.4f %.4f ",datmax5[j],dat25[j],dat35[j]);
 };
 fprintf(fdelta,"\n");
```

```
if (trte == 1)                  /* Training file */
{
  printf("Select (choice 1-7):");
  scanf("%d",&cslut);
  switch(cslut){
  case 1:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b0,b0,b1);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b0,b0,b1);
           break;
  case 2:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b0,b1,b0);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b0,b1,b0);
           break;
  case 3:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b0,b1,b1);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b0,b1,b1);
           break;
  case 4:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b1,b0,b0);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b1,b0,b0);
           break;
  case 5:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b1,b0,b1);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b1,b0,b1);
           break;
  case 6:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b1,b1,b0);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b1,b1,b0);
           break;
  case 7:printf ("Output %d = %d,%d,%d,%d,%d",j+1,b0,b0,b1,b1,b1);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b1,b1,b1);
           break;
 default:printf ("Output Unknown = %d,%d,%d,%d,%d",b0,b0,b0,b0,b0);
           fprintf(fdelta,"%d %d %d %d %d",b0,b0,b0,b0,b0);
           break;

 }
   printf("\n");
    fprintf(fdelta,"\n");
 }
 }
fclose(fdelta);
fclose(f);
printf("\n Press key ESC to QUIT \n\n");
}while(( getch() != ESC));
closegraph();
return;
}
```

ภาคผนวก ข

โปรแกรมระบบนิวรอลเน็ตเวอร์ก

โปรแกรมระบบนิวรอลเน็ตเวอร์ก แบ่งออกเป็น 6 ส่วนคือ

1. ไฟล์ INCMAIN.H และ MENU.CPP เป็นไฟล์เมนู
2. ไฟล์ RECORDNN.H และ ALLOCDAT.CPP เป็นไฟล์สำหรับการสร้างและการจัดเก็บข้อมูล
3. ไฟล์ SIGMOID.CPP เป็นไฟล์ฟังก์ชันซิกมอยด์สำหรับนิวรอลเน็ตเวอร์ก
4. ไฟล์ ADAPTW.CPP เป็นไฟล์สำหรับการปรับค่าน้ำหนักและค่าพารามิเตอร์ต่างๆ
5. ไฟล์ GRAPHDAT.H และ GRAPHDAT.CPP เป็นไฟล์สำหรับแสดงกราฟในการเรียนรู้และผลในการทดสอบ
6. ไฟล์ BKPLEARN.H และ BKPLEARN.CPP เป็นไฟล์สำหรับการเรียนรู้และการทดสอบ

1. ไฟล์ INCMAIN.H และ MENU.CPP

1.1 ไฟล์ INCMAIN.H

```
#include <alloc.h>
#include <bios.h>
#include <dos.h>
#include <conio.h>
#include <ctype.h>
#include <graphics.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
```

1.2 ไฟล์ MENU.CPP

```
#include "incmain.h"
```

```
#include "recordnn.h"
#include "bkplearn.h"
#include "graphdat.h"

#define NOT_MOVED 0
#define RIGHT     1
#define LEFT      2
#define UP        3
#define DOWN      4
#define LEFTB 1
#define RIGHTB 2
#define MENU_MAX 20

void initialize();
void mode();
void mouse_position(int *x,int *y);
void set_mouse_position(int x,int y);
void mouse_motion(int *deltax,int *deltay);
void goto_xy(int x,int y);
void cursor_on(),cursor_off(),mouse_reset();
void set_hor(),set_ver();
void wait_on(int button);
void draw_menu(),highlight(int ac,int co);
void status(bpta *bp,int choice,char sm);
int read_mouse();
int read_kb();
menu();
mouse_menu();
void mainmenu(char sm);
int  rightb_pressed();
int  leftb_pressed();
int  arrow_choice=0,count=9;
int x = 10,y = 10;

void main()
{
   bpta  *nn1;
   int done =0;
   int deltax,deltay;
   char sm;
   nn1 = (bpta *)calloc(1,sizeof(bpta));
while((sm != 'y') && (sm != 'Y') &&(sm != 'n') && (sm != 'N'))
{
printf("Do you have mouse ?(y/n)");
scanf("%s",&sm);
};
mainmenu(sm);
do{
   if((sm == 'y') || (sm == 'Y'))
   {
      mouse_motion(&deltax,&deltay);
      if(deltax || deltay) read_mouse();
      if(leftb_pressed() || rightb_pressed()) {
      done=  read_mouse();
      if(done)
```

```
        status(nn1,done,sm);
        }
      }
        if(kbhit()) {
                done = read_kb();
                done +=1;
                if(done)
                status(nn1,done,sm);
        }
        }while(done<9);
closegraph();
}

void mainmenu(char sm)
{
   int i,j,k,xx = 0,yy =0,yy1,xx1= 0,yy2;
   char *fname[] = {"Input","Hidden 1","Hidden 2","Hidden n","Output"};
   char *fname1[] = {"i","j","k","l","m"};
   char sty[10],sty1[10];
  initialize();
   setbkcolor(3);

   setcolor(15);
   setlinestyle(0,1,1);
   line(0,0,0,getmaxy());
   line(0,getmaxy(),getmaxx(),getmaxy());
   line(0,0,getmaxx(),0);
   line(getmaxx(),0,getmaxx(),getmaxy());
   line(0+5,0+5,0+5,getmaxy()-5);
   line(0+5,getmaxy()-5,getmaxx()-5,getmaxy()-5);
   line(0+5,0+5,getmaxx()-5,0+5);
   line(getmaxx()-5,0+5,getmaxx()-5,getmaxy()-5);
   setcolor(14);
   settextstyle(0,0,5);
   outtextxy(5,15," Neural Network ");
   settextstyle(0,0,3);
   outtextxy(250,80," for ");
   settextstyle(0,0,4);
   outtextxy(120,140,"Gas Sensing");
   settextstyle(0,0,2);
   outtextxy(120,430,"Back propagation Network");

   setcolor(12);
   settextstyle(0,0,1);
   for (i = 0;i < 5;i++)
  {
   yy = 0;
   strcpy(sty1,fname1[i]);
   outtextxy(130+xx,370,sty1);
   outtextxy(142+xx,370,"node");
   strcpy(sty,fname[i]);
   outtextxy(100+xx,390,sty);
   outtextxy(100+xx,400,"layer");
   line(30+xx,364,105+xx,364);
```

```
  for (j = 0;j < 4;j++)
  {
  circle(118+xx,235+yy,12);
  yy = yy+30;
  }
   circle(118+xx,365,12);
   outtextxy(115+xx,342,":");
   xx = xx+100;
 }
 yy = 0;
 for (i = 0;i < 4;i++)
 { yy1 =0;
  outtextxy(30,225+yy,"Input");
  outtextxy(550,225+yy,"Output");
   circle(160+xx1,210,12);
  line(172+xx1,210,212+xx1,225);
  outtextxy(158+xx1,208,"b");

  line(30,234+yy,105,234+yy);
  line(30+xx,234+yy,105+xx,234+yy);
  yy = yy+30;
 for (j = 0;j < 4;j++)
 {
  yy2 = 0;
  for (k = 0;k < 4;k++)
  {
  line(130+xx1,234+yy2,205+xx1,234+yy1);
  line(130+xx1,234+yy2,205+xx1,234+130);
  line(130+xx1,364,205+xx1,234+yy2);
  yy2 = yy2 +30;
  }
  yy1 = yy1+30;
 }
  xx1 = xx1 +100;

 }
  outtextxy(30,235+yy,fname[0]);
  outtextxy(550,235+yy,fname[4]);
 line(30+xx,232+yy+10,105+xx,232+yy+10);
 yy = 0;
 setfillstyle(1,3);
 bar(370,225,380,365);
 for (j =0; j < 7;j++)
 {
  outtextxy(372,225+yy,":");
  yy = yy +28;
 }
  settextstyle(1,0,1);
if (sm == 'y')
{
  mouse_reset();
  set_hor();
  set_ver();
  set_mouse_position(x,y);
};
```

```
   draw_menu();
}


void  status(bpta *bp,int choice,char sm)
{

      switch(choice){
          case 1:
                    training(bp);
                    mainmenu(sm);
                    break;
          case 2:
                    testing(bp);
                    mainmenu(sm);
                    break;
          case 3:
                    getdatasensor(bp);
                    mainmenu(sm);
                    break;
          case 4:
                    newdemo(bp);
                    mainmenu(sm);
                    break;
          case 5:
                    loaddemo(bp);
                    mainmenu(sm);
                    break;
          case 6:
                    savedemo(bp);
                    mainmenu(sm);
                    break;
          case 7:
                    saveyy(bp);
                    mainmenu(sm);
                    break;
          case 8:
                    saveTyy(bp);
                    mainmenu(sm);
                    break;
          case 9:
                    break;

      }
}

int read_mouse()
{
   int choice;
   setcolor(0);
   if(rightb_pressed() || leftb_pressed()) {
      choice = menu();
      return(choice);
   }
return(choice);
```

```
  }


int  read_kb()
  {
    union k{
        char c[2];
        int i;

         } c;

   c.i = bioskey(0);
   cursor_off();
   highlight(arrow_choice,0);
   cursor_on();
   if(c.c[0]) {

   /* check for enter or space bar */
   switch(c.c[0]) {
     case '\r' : cursor_off();
                 highlight(arrow_choice,1);
                 cursor_on();
                 return arrow_choice;
     case ' '  :arrow_choice++;
         break;
     case 29   :return -1;
     }
     }
     else { /* check for special key */
        switch(c.c[1]) {
          case 72: arrow_choice--; /* up arrow */
            break;
          case 80: arrow_choice++; /* down arrow */
            break;
     }
     }
     if(arrow_choice==count) arrow_choice=0;
     if(arrow_choice<0) arrow_choice = count-1;

     /* highlight the next selection */
     cursor_off();
     highlight(arrow_choice,1);
     cursor_on();
     return -1;

  }

void mode(int mode_code)
{
  union REGS r;
  r.h.al = mode_code;
  r.h.ah = 0;
  int86(0x10,&r,&r);
}
```

```
/* set cursor on */

menu()
{
   int x,y,choice;
   while(rightb_pressed() || leftb_pressed());
   cursor_on();

   choice = mouse_menu();
   return choice;
}

mouse_menu()
{

   int mousex,mousey;


      while(rightb_pressed() || leftb_pressed());

      mouse_position(&mousex,&mousey);

      if(mousex>=220 && mousex<420){
           cursor_off();
           highlight(arrow_choice,0);
           if(mousey>250 && mousey<262) {
              arrow_choice = 0;
              highlight(arrow_choice,1);
              cursor_on();
              return 1;
           }
           if(mousey>262 && mousey<274) {
              arrow_choice =1;
              highlight(arrow_choice,1);
              cursor_on();
              return 2;
           }
           if(mousey>274 && mousey<286) {
              arrow_choice = 2;
              highlight(arrow_choice,1);
              cursor_on();
              return 3;
           }
           if(mousey>286 && mousey<298) {
              arrow_choice = 3;
              highlight(arrow_choice,1);
              cursor_on();
              return 4;
           }
           if(mousey>298 && mousey<310) {
              arrow_choice = 4;
```

```
            highlight(arrow_choice,1);
            cursor_on();
            return 5;
          }
          if(mousey>310 && mousey<322) {
            arrow_choice = 5;
            highlight(arrow_choice,1);
            cursor_on();
            return 6;
          }
          if(mousey>322 && mousey<334) {
            arrow_choice = 6;
            highlight(arrow_choice,1);
            cursor_on();
            return 7;
          }
          if(mousey>334 && mousey<346) {
            arrow_choice = 7;
            highlight(arrow_choice,1);
            cursor_on();
            return 8;
          }
          if(mousey>346 && mousey<358) {
            arrow_choice = 8;
            highlight(arrow_choice,1);
            cursor_on();
            return 9;
          }
          highlight(arrow_choice,1);
          cursor_on();
          return -1;
      }
      else
      highlight(arrow_choice,1);
      cursor_on();
      return -1;
}

void wait_on(int button)
{
  if(button==LEFTB)
     while(leftb_pressed());
  else
     while(rightb_pressed());
}

void cursor_on()
{
 union REGS r;

 r.x.ax = 1;
 int86(0x33,&r,&r);
}
```

```
/* set min and max cursor position  (vertical)*/
void set_ver()
{
  union REGS r;
  r.x.ax = 8;
  r.x.cx = 0;
  r.x.dx = getmaxy()-5;
  int86(0x33,&r,&r);
}
/* set min and max cursor position (horizontal) */
void set_hor()
{
  union REGS r;
  r.x.ax = 7;
  r.x.cx  = 0;
  r.x.dx  = 636;
  int86(0x33,&r,&r);
}
/* hide the cursor */
void cursor_off()
{
  union REGS r;
  r.x.ax = 2;
  int86(0x33,&r,&r);
}
/* get mouse position */
void mouse_position(int *x,int *y)
{
  union REGS r;
  r.x.ax = 3;
  int86(0x33,&r,&r);
  *x = r.x.cx;
  *y = r.x.dx;
}
/* set position */
void set_mouse_position(int px,int py)
{
   union REGS r;
   r.x.ax = 4;
   r.x.cx = px;
   r.x.dx = py;
   int86(0x33,&r,&r);
}
/*  check if right button is pressed */
int rightb_pressed()
{
   union REGS r;
   r.x.ax = 3;
   int86(0x33,&r,&r);
   if((r.x.bx & 0x0a) == 1) return 1;
}
```

```
/*  check if left button is pressed */
int leftb_pressed()
{
   union REGS r;
   r.x.ax = 3;
   int86(0x33,&r,&r);
   if((r.x.bx & 0x01)== 1) return 1;

}
/* Return the direction of travel */
void mouse_motion(int *deltax,int *deltay)
{
   union REGS r;
   int c,d;
   r.x.ax = 11;
   int86(0x33,&r,&r);

   if(0< r.x.cx < 32767) *deltax = RIGHT;
   if(r.x.cx > 32767) *deltax = LEFT;
   if( r.x.cx == 0) *deltax = NOT_MOVED;


   if(0< r.x.dx < 32767) *deltay = DOWN;
   if(r.x.dx > 32767) *deltay = UP;
   if( r.x.dx == 0) *deltay = NOT_MOVED;
}

/* init graphic mode */
void initialize(void)
{
      int errorcode;
      int xasp, yasp;              /*  Used to read the aspect ratio */
      int g_driver,g_mode;

      detectgraph(&g_driver,&g_mode);
      g_mode = VGAHI;
      initgraph( &g_driver, &g_mode, "c:\tc" );
      errorcode = graphresult();  /* Read result of initialization*/
      if( errorcode != grOk ){              /* Error occured during init      */
         printf(" Graphics System Error: %s\n", grapherrormsg( errorcode ) );
         exit(1);
      }
}

/* initialize mouse */
void mouse_reset()
{
   union REGS r;
   r.x.ax = 0;
   int86(0x33,&r,&r);
   if(r.x.bx != 2) {
```

```
      printf(" two-button mouse is required ");
      exit(1);
    }
}
/* send the cursor to the specified x,y position */
void goto_xy(int x,int y)
{
   union REGS r;
   r.h.ah = 2;
   r.h.dl = y;
   r.h.dh = x;
   r.h.bh = 0;
   int86(0x10,&r,&r);
}

void draw_menu()
{
            cursor_off();
            setcolor(14);
            setfillstyle(1,9);
            bar(220,250,420,262);
            rectangle(220,250,420,262);
            outtextxy(224,252,"Training ");
            bar(220,262,420,274);
            rectangle(220,262,420,274);
            outtextxy(224,264,"Testing ");
            bar(220,274,420,286);
            rectangle(220,274,420,286);
            outtextxy(224,276,"Load data ");
            bar(220,286,420,298);
            rectangle(220,286,420,298);
            outtextxy(224,288,"New Network ");
            bar(220,298,420,310);
            rectangle(220,298,420,310);
            outtextxy(224,300,"Load Network ");
            bar(220,310,420,322);
            rectangle(220,310,420,322);
            outtextxy(224,312,"Save Network ");
            bar(220,322,420,334);
            rectangle(220,322,420,334);
            outtextxy(224,324,"Save train output ");
            bar(220,334,420,346);
            rectangle(220,334,420,346);
            outtextxy(224,336,"Save test output ");
            bar(220,346,420,358);
            rectangle(220,346,420,358);
            outtextxy(224,348,"Quit ");

            highlight(arrow_choice,1); /* highlight the first selection */
            cursor_on();

}

void   highlight(int ac,int co)
{
```

```
            setfillstyle(1,12);
            if(co == 0)
               setfillstyle(1,9);

            bar(220,250+ac*12,420,262+ac*12);
            setcolor(14);
            rectangle(220,250+ac*12,420,262+ac*12);

            switch(ac) {
            case 0:
                 outtextxy(224,252,"Training ");
                 break;
            case 1:
                 outtextxy(224,264,"Testing ");
                 break;
            case 2:
                 outtextxy(224,276,"Load data ");
                 break;
            case 3:
                 outtextxy(224,288,"New Network ");
                 break;
            case 4:
                 outtextxy(224,300,"Load Network ");
                 break;
            case 5:
                 outtextxy(224,312,"Save Network ");
                 break;
            case 6:
                 outtextxy(224,324,"Save train output ");
                 break;
           case 7:
                 outtextxy(224,336,"Save test output ");
                 break;
           case 8:
                 outtextxy(224,348,"Quit ");
                 break;
            }
}
```

2. ไฟล์ RECORDNN.H และ ALLOCDAT.CPP

2.1 ไฟล์ RECORDNN.H

```
#include "incmain.h"

typedef  struct {
                  unsigned long    niterstart,niterstop;
                  int              npattern,npattest;
                  int              layer;
                  float far        ***w,***old_w,**wbias,**old_wbias;
                  float far        **out,**bout,**net;
                  float far        ***dokdoj;
                  int              *nodes;
                  int              *fsigmoid;
                  float            l_rate[8],b_rate[8];m_rate[8];
```

```
        char            name[14];
        FILE            *file;
    }  bpta ;

extern void   Allocate(bpta *bp);
extern void   getdatasensor(bpta *bp);
extern void   newdemo(bpta *bp);
extern void   chkmemnn(bpta *bp);
extern void   chkmem(float *bp);
extern void   saveyy(bpta *bp);
extern void   saveTyy(bpta *bp);
extern void   savedemo(bpta *bp);
extern void   loaddemo(bpta *bp);
extern void   Free(bpta *bp);
extern int    chkbioskey(void);
extern float  get(char textc[],int x,int y);
extern int    getpat(char textc[],int x,int y);
extern int    chkeofnn(bpta *bp);
```

2.2ไฟล์ ALLOCDAT.CPP

```
#include "incmain.h"
#include "recordnn.h"
#include "graphdat.h"
#include "bkplearn.h"

void getdatasensor(bpta *bp);
void newdemo(bpta *bp);
void Allocate(bpta *bp) ;
void saveyy(bpta *bp);
void saveTyy(bpta *bp);
void savedemo(bpta *bp);
void loaddemo(bpta *bp);
void Free(bpta *bp);
void chkmemnn(bpta *bp);
void chkmem(float *bp);
void chkmemfile(FILE *bp);
float get(char textc[],int x,int y);
int   getpat(char textc[],int x,int y);
int   chkbioskey(void);

void  getdatasensor(bpta *bp)
{
char file[30],*rf;

clrscr();
initgphc();
setfillstyle(SOLID_FILL,0);

printf("File name of data sensor =    ");
scanf("%s",&file);
rf = strcat(file,".dat");
strcpy(bp->name,rf);
closegraph();
}
```

```
void  newdemo(bpta *bp)
{
unsigned long niterstart,niterstop;
int   i,l,j,*nodes,npattern,layer,fsigmoid;
int   h;
char  key='z',st;
float delta,***w,***old_w,**wbias,**old_wbias;
float minn=-1,maxx=1,wrd;
char  file[30];
FILE  *f;

Free(bp);
clrscr();
initgphc();

bp->nodes  = (int *)calloc(layer,sizeof(int));
bp->fsigmoid       = (int *)calloc(layer,sizeof(int));

setfillstyle(SOLID_FILL,0);
while ((st != 'y') && (st != 'Y')
        && (st != 'n') && (st != 'N'))
{
printf("Do you have text file ?(y/n)");
scanf("%s",&st);
};
if ((st == 'y') || (st == 'Y'))
{
gotoxy(30,30);
printf("File of Neural network =     ");
scanf("%s",&file);
f = fopen(file,"r");
if(f==NULL) goto end;
fscanf(f,"%lu",&niterstart);bp->niterstart = niterstart;
fscanf(f,"%lu",&niterstop);bp->niterstop = niterstop;
fscanf(f,"%d",&npattern);bp->npattern = npattern;
fscanf(f,"%d",&layer); bp->layer = layer ;
nodes = bp->nodes;
for(i=0;i<layer;i++)
  {
fscanf(f,"%d",nodes+i);};
for(i=0;i<layer;i++)
 {fscanf(f,"%d",&fsigmoid); bp->fsigmoid[i] = fsigmoid;
 };
}else
{
printf ("Start iteration =");
scanf("%lu",&niterstart);bp->niterstart = niterstart;
printf("\nStop iteration =");
scanf("%lu",&niterstop);bp->niterstop = niterstop;
printf("\nNumber patterns = ");
scanf("%d",&npattern);bp->npattern = npattern;
```

```
printf("\nNumber of layer =");
scanf("%d",&layer); bp->layer = layer ;
nodes = bp->nodes;
for(i=0;i<layer;i++)
{ printf("\nNumber node of layer [%d] =",i);
  scanf("%d",nodes+i);
};
for(i=0;i<layer;i++)
{
printf("\nFunction sigmoid of layer [%d] =",i);
scanf("%d",&fsigmoid); bp->fsigmoid[i] = fsigmoid;
}
};
Allocate(bp);
printf("\nStart iteration = %lu \n",bp->niterstart);
printf("\nStop iterstop = %lu \n",bp->niterstop);
printf("\nNumber of patterns = %d \n",bp-> npattern);
printf("\nNumber of layer = %d \n",bp->layer);
for(i=0;i<layer;i++)
  {printf("Number node of layer[%d] = %d  \n",i,nodes[i]);};

for(i=0;i<layer;i++)
 {printf("Function sigmoid of layer[%d] =  %d  \n",i,bp->fsigmoid[i]);
 };


w  =bp->w;
old_w =bp->old_w;
wbias  =bp->wbias;
old_wbias =bp->old_wbias;
while( (key != 'y') && (key != 'n'))
{  printf("\nDo you want to set range of random weight.? (Y/N)") ;
   key  =  getch();
};
if(key=='y')
{
  printf("\nSet range of Weight  :  ");
  printf("\n Minimum weights(default = -1) =  ");
  scanf("%f",&minn);
  printf("\n Maximum weights(default = 1) =  ");
  scanf("%f",&maxx);
};

  printf("\n\n\t Mininum weights = %f ,  Maximum weights = %f",minn,maxx);
  printf("\n\n\n\t\t              Press any key to menu!");
  getch();

  for(l=1;l<layer;l++)
  {
  h = l-1;
  wrd =  (maxx-minn)*200;
    for(i=0;i<nodes[h];i++)
    for(j=0;j<nodes[l];j++)
     {
       w[l][i][j]  =  (int) random((int)wrd) ;
```

```
      w[l][i][j]  = old_w[l][i][j]  =  w[l][i][j]/200.+minn ;
      if(w[l][i][j]==0)  old_w[l][i][j]  =w[l][i][j]=0.0001;
    };
  };
  for(l=1;l<layer;l++)
  {
if(nodes[l]>1)
{  wrd =  (maxx-minn)/((float)nodes[l]-1);
   for(i=0;i<nodes[l];i++)
    {
      wbias[l][i]  =  old_wbias[l][i]  =  wrd*(float)i+minn ;
      if(wbias[l][i]==0)
         { old_wbias[l][i]  = wbias[l][i] = 0.0001; }
    };
}else
{ wrd =  (maxx-minn)*200;
   for(i=0;i<nodes[l];i++)
    {
      wbias[l][i]  =  (int) random((int)wrd) ;
      wbias[l][i]  =  old_wbias[l][i]  =  wrd/200.+minn ;
      if(wbias[l][i]==0)
         { old_wbias[l][i]  = wbias[l][i] = 0.0001; }
    };
};

};
closegraph();
fclose(f);
end : return;
}

void Allocate(bpta *bp)
{
int   i=0,l=0,layer,*nodes,h;
float ***w,***old_w,***dokdoj;
float **omem;

layer = bp->layer ;
nodes = bp->nodes;

/*                        Allocate dokdoj                  */

bp->dokdoj  = (float far ***)farcalloc(layer,sizeof(float));
chkmem((float *) bp->dokdoj);
dokdoj = bp->dokdoj;
for(l=0;l<layer;l++)
{  *(dokdoj+l) = (float **)calloc(nodes[l],sizeof(float));
   chkmem((float *) *(dokdoj+l));
};
  for(l=1;l<layer;l++)
   { for(i=0;i<nodes[l];i++)
     {h = l-1;
      *(*(dokdoj+l)+i) = (float *)calloc(nodes[h],sizeof(float));
      chkmem(*(*(dokdoj+l)+i));
     };
```

```
   };


/*                    Allocate weight                    */

bp->w  = (float far ***)farcalloc(layer,sizeof(float));
chkmem((float *)bp->w);
w = bp->w;
for(l=1;l<layer;l++)
{ h = l-1;
  *(w+l) = (float **)calloc(nodes[h],sizeof(float));
  chkmem((float *)*(w+l));
};
  for(l=1;l<layer;l++)
   { h =l-1;
     for(i=0;i<nodes[h];i++)
     {*(*(w+l)+i) = (float *)calloc(nodes[l],sizeof(float));
      chkmem(*(*(w+l)+i));
     };
   };

/*                    Allocate old_weight                    */

bp->old_w = (float far ***)farcalloc(layer,sizeof(float));
chkmem((float *)bp->old_w);
old_w = bp->old_w;
for(l=1;l<layer;l++)
{ h =l-1;
*(old_w+l) = (float **)calloc(nodes[h],sizeof(float));
 chkmem((float *)*(old_w+l));
};
  for(l=1;l<layer;l++)
 { h =l-1;
   for(i=0;i<nodes[h];i++)
  {   *(*(old_w+l)+i) = (float *)calloc(nodes[l],sizeof(float));
     chkmem(*(*(old_w+l)+i));
  };
 };

/*                    Allocate weight-bias                    */

bp->wbias  = (float far **)farcalloc(layer,sizeof(float));
chkmem((float *)bp->wbias);
omem = bp->wbias;
for(l=0;l<layer;l++)
{*(omem+l) = (float *)calloc(nodes[l],sizeof(float));
chkmem(*(omem+l));
};

/*                Allocate old_weight-bias                    */

bp->old_wbias  = (float far **)farcalloc(layer,sizeof(float));
chkmem((float *)bp->old_wbias);
omem = bp->old_wbias;
```

```
for(l=0;l<layer;l++)
{*(omem+l) = (float *)calloc(nodes[l],sizeof(float));
 chkmem(*(omem+l));
};
/*                  Allocate output                  */

bp->out  = (float far **)farcalloc(layer,sizeof(float));
chkmem((float *)bp->out);
omem = bp->out;
for(l=0;l<layer;l++)
{
*(omem+l) = (float *)calloc(nodes[l],sizeof(float));
chkmem(*(omem+l));
};

/*                  Allocate boutput                 */

bp->bout = (float far **)farcalloc(layer,sizeof(float));
chkmem((float *) bp->bout);
omem = bp->bout;
for(l=0;l<layer;l++)
{
*(omem+l) = (float *)calloc(nodes[l],sizeof(float));
 chkmem((float *)*(omem+l));
};

/*                  Allocate net                     */

bp->net = (float far **)farcalloc(layer,sizeof(float));
chkmem((float *) bp->net);
omem = bp->net;
for(l=0;l<layer;l++)
{
*(omem+l) = (float *)calloc(nodes[l],sizeof(float));
chkmem(*(omem+l));
};

for (l=0;l<8;l++)
bp->l_rate[l]       = 0.15;
for (l=0;l<8;l++)
bp->b_rate[l]     = 0.15;
for (l=0;l<8;l++)
bp->m_rate[l] = 0.15;
return;
}

void saveyy(bpta *bp)
{
unsigned long niterstart;
int i,j,p,numpoint,h,pattern,re = 0;
float *y,nerr;
float *input,*output;
char finame[30],*rfi ;
FILE *fi;
```

```
      clrscr();
      initgphc();
      setfillstyle(SOLID_FILL,0);
      gotoxy(30,30);
      printf(" Save  output filename of Neural network =      ");
      scanf ("%s",&finame);
rfi = strcat(finame,"_o.dat");
bp->file = fopen(bp->name,"r");
h =(bp->layer)-1;
input =(float *) calloc(bp->nodes[0],sizeof(float));
output =(float *) calloc(bp->nodes[h],sizeof(float));
y =(float *) calloc(bp->nodes[h],sizeof(float));
numpoint = bp->nodes[h];
pattern = bp->npattern;
niterstart = bp->niterstart;

fi = fopen(rfi,"a");  chkmemfile(fi);
fprintf(fi,"\n Training output \n");
printf("\n Training output \n");

for(i = 0,nerr = 0.0;i < pattern;i++)
{
 listdata(bp,input,output,re);
 sigmoid(bp,input,y);
   p = i+1;
 fprintf(fi," \n %d  \t",p);
 printf(" \n %d  \t",p);
 for (j = 0;j < numpoint;j++)
{
  fprintf(fi," %f ",y[j]);
  printf(" %f ",y[j]);
  float tem = output[j]-y[j];
  nerr += tem * tem;
  }
 }
fprintf(fi,"\n Iteration = %lu",niterstart);
printf("\n Iteration = %lu",niterstart);
nerr /= (pattern * numpoint);
fprintf(fi,"\t Sum squared error = %e",nerr);
printf("\t Sum squared error = %e",nerr);
fclose(fi);
getch();
closegraph();
return;
}

void saveTyy(bpta *bp)
{
   int k,j,p= 0,numnode,h,pattern,ret = 0;
   float *yy;
   float *input,*output;
   char ftname[20],*rftname;
   FILE *fi;

   float low = 0.3,high = 0.7;
```

```
        clrscr();
        initgphc();
        setfillstyle(SOLID_FILL,0);
        gotoxy(30,30);
        printf(" Save  output filename of Neural network from testing =      ");
        scanf ("%s",&ftname);
rftname = strcat(ftname,"_t.dat");
bp->file = fopen(bp->name,"r");
h =(bp->layer)-1;
input =(float *) calloc(bp->nodes[0],sizeof(float));
yy =(float *) calloc(bp->nodes[h],sizeof(float));
numnode = bp->nodes[h];
pattern = bp->npattest;

fi = fopen(rftname,"a");  chkmemfile(fi);
fprintf(fi," \n Testing output \n");
printf(" \n Testing output \n");
for(k = 0;k < pattern;k++)
{
   listTdata(bp,input,ret);
   sigmoid(bp,input,yy);
   p = k+1;
 fprintf(fi," \n %d  \t",p);
 printf(" \n %d  \t",p);
 for (j = 0;j < numnode;j++)
{
  fprintf(fi," %f ",*(yy+j));
  printf(" %f ",*(yy+j));
}
if (numnode == 3)
{
if ((yy[0] < low) && (yy[1] < low) && (yy[2] > high))
{
  fprintf(fi," %s ","water");
printf(" %s ","water");
}else
if ((yy[0] < low) && (yy[1] > high) && (yy[2] < low))
{
  fprintf(fi," %s ","EtOH");
  printf(" %s ","EtOH");
}else
if ((yy[0] < low) && (yy[1] > high) && (yy[2] > high))
{
  fprintf(fi," %s ","acetone");
  printf(" %s ","acetone");
}else
if ((yy[0] > high) && (yy[1] < low) && (yy[2] < low))
{
  fprintf(fi," %s ","amonia");
  printf(" %s ","amonia");
}else
if ((yy[0] > high) && (yy[1] < low) && (yy[2] > high))
{
  fprintf(fi," %s ","acetaldehyde");
```

```
  printf(" %s ","acetaldehyde");
 }else
 if ((yy[0] > high) && (yy[1] > high) && (yy[2] < low))
 {
  fprintf(fi," %s ","solution1");
  printf(" %s ","solution1");
 }else
 if ((yy[0] > high) && (yy[1] > high) && (yy[2] > high))
 {
  fprintf(fi," %s ","solution2");
  printf(" %s ","solution2");
 }else
 {
  fprintf(fi," %s ","unknown");
  printf(" %s ","unknown");
 }
 }
 if (numnode == 5)
 {
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] < low) &&
     (yy[3] < low) && (yy[4] > high))
 {
  fprintf(fi," %s ","water");
  printf(" %s ","water");
 }else
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] < low) &&
     (yy[3] > high) && (yy[4] < low))
 {
  fprintf(fi," %s ","EtOH");
  printf(" %s ","EtOH");
 }else
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] < low) &&
     (yy[3] > high) && (yy[4] >= high))
 {
  fprintf(fi," %s ","acetone");
  printf(" %s ","acetone");
 }else
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] > high) &&
     (yy[3] < low) && (yy[4] < low))
 {
  fprintf(fi," %s ","ammonia");
  printf(" %s ","ammonia");
 }else
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] > high) &&
     (yy[3] < low) && (yy[4] > high))
 {
  fprintf(fi," %s ","acetaldehyde");
  printf(" %s ","acetaldehyde");
 }else
 if ((yy[0] < low) && (yy[1] < low) && (yy[2] > high) &&
     (yy[3] > high) && (yy[4] < low))
 {
  fprintf(fi," %s ","solution1");
  printf(" %s ","solution1");
 }else
```

```
if ((yy[0] < low) && (yy[1] < low) && (yy[2] > high) &&
    (yy[3] > high) && (yy[4] > high))
{
  fprintf(fi," %s ","solution2");
  printf(" %s ","solution2");
 }else
 {
  fprintf(fi," %s ","unknown");
  printf(" %s ","unknown");
 }
 }
 };
 fclose(fi);
 getch();
 closegraph();
 return;
 }

 void savedemo(bpta *bp)
 { FILE *f;
   char fname[20],*rname;
   int  l=0,i=0,j=0,layer=0,h;
   float ***w,**wbias,**y;
   int *nodes;

       clrscr();
       initgphc();
       setfillstyle(SOLID_FILL,0);
       gotoxy(30,30);
       printf(" Save filename of Neural network =       ");
       scanf ("%s",&fname);

 /*  start iteration
     stop  iteration
     pattern
     layer
     nodes
     function sigmoid
     weight
     weight bias
 */
 w =  bp->w;
 wbias =  bp->wbias;
 layer =bp->layer;
 nodes =bp->nodes;
 rname = strcat(fname,"_w.dat");
 f = fopen(rname,"w");  chkmemfile(f);
 fprintf(f,"%lu    \n"     ,bp->niterstart);
 fprintf(f,"%lu    \n"     ,bp->niterstop);
 fprintf(f,"%d     \n"     ,bp->npattern);
 fprintf(f,"%d     \n"     ,bp->layer);
 for(i=0;i<layer;i++)
 { fprintf(f,"%d \n",nodes[i]);};

 for(i=0;i<layer;i++)
```

```
fprintf(f,"%d \n",bp->fsigmoid[i]);

for(l=1;l<layer;l++)
 {h =l-1;
  for(i=0;i<nodes[h];i++)
  for(j=0;j<nodes[l];j++)
   fprintf(f," %f \n",w[l][i][j]);
 }
for(l=1;l<layer;l++)
 for(i=0;i<nodes[l];i++)
  { fprintf(f,"%f \n",wbias[l][i]);
  };

fclose(f);
closegraph();
return;
}

void loaddemo(bpta *bp)
{ FILE  *f;
  char  fname[20],*rname;
  int   i,l,j,*nodes,npattern,layer,fsigmoid = 0,h;
  unsigned long niterstart,niterstop;
  float ***w,***old_w,**wbias,**old_wbias;
      clrscr();
      initgphc();
      setfillstyle(SOLID_FILL,0);
      gotoxy(30,30);
      printf("Input filename =      ");
      scanf ("%s",&fname);
rname = strcat(fname,"_w.dat");
Free(bp);
/*  start iteration
    stop iteration
    pattern
    layer
    nodes
    function sigmoid
    weight
    weight bias
*/
f = fopen(rname,"r");   chkmemfile(f);
fscanf(f,"%lu",&niterstart);bp->niterstart = niterstart;
fscanf(f,"%lu",&niterstop);bp->niterstop = niterstop;
fscanf(f,"%d",&npattern);bp->npattern = npattern;
fscanf(f,"%d",&layer); bp->layer = layer ;
nodes =  bp->nodes = (int *)calloc(layer,sizeof(int));
bp->fsigmoid    = (int *)calloc(layer,sizeof(int));
chkmem((float *)nodes);
for(i=0;i<layer;i++)
{ fscanf(f,"%d",nodes+i);};

for(i=0;i<layer;i++)
{fscanf(f,"%d",&fsigmoid); bp->fsigmoid[i] = fsigmoid;
};
```

```
Allocate(bp);
printf("\nStart iteration = %lu \n",bp->niterstart);
printf("\nStop iteration = %lu \n",bp->niterstop);
printf("\nNumber of patterns = %d \n",bp->npattern);
printf("\nNumber of layer = %d \n",bp->layer);
for(i=0;i<layer;i++)
  {printf("Number nodes[%d] = %d  \n",i,nodes[i]);};

for(i=0;i<layer;i++)
 {printf("Function sigmoid layer[%d] =  %d  \n",i,bp->fsigmoid[i]);
 };
w =  bp->w;
old_w =  bp->old_w;
wbias =  bp->wbias;
old_wbias =  bp->old_wbias;
nodes =bp->nodes;
for(l=1;l<layer;l++)
 {h =l-1;
  for(i=0;i<nodes[h];i++)
  for(j=0;j<nodes[l];j++)
   { fscanf(f,"%f",&w[l][i][j]);
     old_w[l][i][j] = w[l][i][j];
   };
 };
for(l=1;l<layer;l++)
 for(i=0;i<nodes[l];i++)
  { fscanf(f,"%f",&wbias[l][i]);
    old_wbias[l][i]  = wbias[l][i] ;
  };
fclose(f);
getch();
closegraph();
return;
}

void Free(bpta *bp)
{
int  i=0,l=0,layer;
int  *nodes,h;
float ***w,***old_w,***dokdoj;
float **omem;
layer = bp->layer ;
nodes   = bp->nodes;

w = bp->w;

/*                    Free  weight                          */

  for(l=1;l<layer;l++)
   {h =l-1;
    for(i=0;i<nodes[h];i++)
      free(*(*(w+l)+i));
   };
  for(l=1;l<layer;l++)
    free(*(w+l));
```

```
free(w);

/*                  Free old_weight                  */

old_w = bp->old_w;
  for(l=1;l<layer;l++)
    { h =l-1;
      for(i=0;i<nodes[h];i++)
      free(*(*(old_w+l)+i));
    }
  for(l=1;l<layer;l++)
    free(*(old_w+l));

free(old_w);

/*                  Free weight-bias                 */

omem = bp->wbias;
for(l=0;l<layer;l++)
  free(*(omem+l));
free(bp->wbias);

/*                  Free old_weight-bias             */

omem = bp->old_wbias;
for(l=0;l<layer;l++)
  free(*(omem+l));
free(bp->old_wbias);

/*                  Free output                      */

omem = bp->out;
for(l=0;l<layer;l++)
  free(*(omem+l));
free(omem);

/*                  Free boutput                     */

omem = bp->bout;
for(l=0;l<layer;l++)
 free(*(omem+l));

free(omem);

/*                  Free net                         */

omem = bp->net;
for(l=0;l<layer;l++)
free(*(omem+l));
free(omem);

/*                  Free  dokdoj                     */
```

```
dokdoj = bp->dokdoj;
for(l=0;l<layer;l++)
 for(i=0;i<nodes[l];i++)
  free( *(*(dokdoj+l)+i) );
for(l=0;l<layer;l++)
 free( *(dokdoj+l) );
free(dokdoj);

/*                  Free nnodes                 */

free(bp->nodes);
}

float get(char textc[],int x,int y)
{ float  v;
  int i=0;
  int  wid=0;
  char  c[50]="",key;

    setfillstyle(SOLID_FILL,RED);
    bar(x-20,y,x+textwidth(textc)+100,y+8);

  setcolor(15);
  outtextxy(x,y,textc);
  wid=x+textwidth(textc)+15;

  while((key=getch())!='\r')
   {      if(key=='\b')
          {  setcolor(RED);  outtextxy(wid,y,c);
             *(c+i-1) = ' ';
             setcolor(15);  outtextxy(wid,y,c);
             i-=1;
          }
          else
          {
            if(i<50)
             {*(c+i)=key;
              outtextxy(wid,y,c);
              ++i;
              };
           };
   };

  v= atof(c);
      if (key == '\r')
       {
           setfillstyle(SOLID_FILL,0);
           bar(x-20,y,x+textwidth(textc)+100,y+8);
       };
  return(v);
}

int getpat(char textc[],int x,int y)
{ int  pat;
  int i=0;
```

```
int  wid=0;
char  c[50]="",key;

  setfillstyle(SOLID_FILL,14);
  bar(x-20,y,x+textwidth(textc)+100,y+8);
setcolor(9);
outtextxy(x,y,textc);
wid=x+textwidth(textc)+15;
while((key=getch())!='\r')
 {      if(key=='\b')
        {  setcolor(14);  outtextxy(wid,y,c);
           *(c+i-1) = ' ';
           setcolor(9);  outtextxy(wid,y,c);
           i-=1;
        }
        else
        {
          if(i<50)
           {*(c+i)=key;
            outtextxy(wid,y,c);
            ++i;
            };
         };
  };
 pat= atoi(c);
    if (key == '\r')
     {
         setfillstyle(SOLID_FILL,14);
         bar(x-20,y,x+textwidth(textc)+100,y+8);
     };
 return(pat);
}

int  chkbioskey(void)
{ int key;
  key = bioskey(1);
  if(key!=0) bioskey(0);
  return(key);
}

void chkmemfile(FILE *bp)
{
 if(bp==NULL)
  {
  sound(300); delay(5) ;nosound();
  printf("DATA ERROR \n \n");
  };
}

void chkmemnn(bpta *bp)
{ if(bp==NULL)
  {sound(300);delay(5);nosound();
  printf(" DATA ERROR  \n \n");
  printf(" Press key to dos \n");
  getch();
```

```
  exit(2);
  };
}

void chkmem(float *bp)
{ if(bp==NULL)
  {
  sound(300); delay(5);nosound();
  printf("DATA ERROR \n \n");
  printf(" Press key to dos \n");
  getch();
  exit(2);
  };
}

int chkeofnn(bpta *bp)
{
   if (feof(bp->file))
   { fclose(bp->file);
     bp->file = fopen(bp->name,"r");
     return(1);
   }
return(0);
}
```

3. ไฟล์ SIGMOID.CPP

```
#include "incmain.h"
#include "recordnn.h"
#include "bkplearn.h"

void sigmoid(bpta *bp,float *data,float *output);
float bsgm(long double x,int fsigmoid);
float sgm(long double x,int fsigmoid);

void sigmoid(bpta *bp,float *data,float *output)
{
 int i=0,j=0,l=0;
 int h1,h2,h3;
 int  layer;
 int *nodes;
 float **out,**bout,**net,***w,**wbias;
 long double sum=0;

 nodes = bp->nodes;
 layer     = bp->layer;
 out  = bp->out;
 net  = bp->net;
 bout = bp->bout;
 w  = bp->w;
 wbias  = bp->wbias;

for(i=0;i<nodes[0];i++)
    { out[0][i]=data[i];
```

```
    };
for(l=1;l<layer;l++)
{  for(j=0;j<nodes[l];j++)
    { sum=0;
      h1=l-1;
      for(i=0; i<nodes[h1];i++)
          { h2 =l-1;
            sum += w[l][i][j] * out[h2][i];
      .     };
      sum += wbias[l][j];

      net[l][j] = sum;
      out[l][j]     = sgm(sum,bp->fsigmoid[l]);
      bout[l][j]    = bsgm(sum,bp->fsigmoid[l]);
    };
};
h1 = layer -1;
for(i=0;i<nodes[h1];i++)
  { h3 = layer -1;
    output[i] = out[h3][i];};
return;
}

float bsgm(long double x,int fsigmoid)
{
  float y;

 switch(fsigmoid)
{
 case 0: y  = expl(-x)/(1+expl(-x))/(1+expl(-x));break;

 default: printf("\n function error \n");
};
return (y);
}

float sgm(long double x,int fsigmoid)
{
  float y;

 switch(fsigmoid)
{
 case 0: y  = 1/(1+expl(-x))   ;break;

 default: printf("\n function error  \n");

};
 return (y);
}
```

4. ไฟล์ ADAPTW.CPP

```
#include "incmain.h"
#include "recordnn.h"
#include "bkplearn.h"

void  adaptw(bpta *bp,float *data);
void outputsigmoid(bpta *bp);
float  dokdwbj(bpta *bp,int l,int k,int m,int i);
float dokdwj(bpta *bp,int l,int k,int m,int i,int j);

void adaptw(bpta *bp,float *data)
{
 int layer,ii=0,i=0,j=0,l=0;
 int h1,h2;
 int *nodes;
 float num1,num2;
 float **out,***w,***old_w,**wbias,**old_wbias;
 float *lr,*mr,*br,dEdw=0,m_term=0;

 nodes= bp->nodes;
 layer     = bp->layer;
 lr        = bp->l_rate;
 br        = bp->b_rate;
 mr        = bp->m_rate;
 out  = bp->out;
 w  = bp->w;
 wbias  = bp->wbias;
 old_wbias = bp->old_wbias;
 old_w = bp->old_w;

outputsigmoid(bp);
for (l=1;l<layer;l++)
 {h1 = l-1;
  for (i=0;i<nodes[h1];i++)
   for (j=0;j<nodes[l];j++)
     { dEdw=0;
       h2 = layer -1;
       for(ii=0; ii< nodes[h2] ;ii++)
           { num1  = -(data[ii]-out[h2][ii]);
             num2  = dokdwj(bp,layer-1,ii,l,i,j);
             dEdw +=  num1*num2;
           };
           m_term = mr[l] * (w[l][i][j] - old_w[l][i][j] );
           old_w[l][i][j] = w[l][i][j];
           w[l][i][j] += -dEdw*lr[l]+m_term;
     };
 };
for (l=1;l<layer;l++)
 for (i=0;i<nodes[l];i++)
     { dEdw=0;
       h1 = layer-1;
       for(ii=0;ii<nodes[h1];ii++)
          { num1  = -(data[ii]-out[h1][ii]);
            num2  = dokdwbj(bp,layer-1,ii,l,i);
```

```
            dEdw += num1*num2;
           };
           m_term      = mr[l] * (wbias[l][i] - old_wbias[l][i] );
           old_wbias[l][i]  = wbias[l][i];
           wbias[l][i]  += -dEdw*br[l]+m_term;
     };
 return;
 }

void outputsigmoid(bpta *bp)
{
int l,i,j;
int *nodes,h;
float ***dokdoj,***w,**bout;

nodes = bp->nodes;
dokdoj = bp->dokdoj;
bout = bp->bout;
w  = bp->w;

for (l=1;l<bp->layer;l++)
{ for (i=0;i<nodes[l];i++)
  {  h = l-1;
     for (j=0;j<nodes[h];j++)
      { dokdoj[l][i][j] = bout[l][i] * w[l][j][i] ;
      };
  };
};
return;
}

float dokdwj(bpta *bp,int l,int k,int m,int i,int j)
{
int   ii;
float dodw;
float ***dokdoj,**out,**bout;
int *nodes,h;

nodes = bp->nodes;
dokdoj= bp->dokdoj;
bout = bp->bout;
out = bp->out;

if(l==m)
 { h =l-1;
   if(k==j)
     dodw = bout[l][k]*out[h][i];
   else
     dodw = 0;
 } else
 {  if(l>m)
     { if(l-1==m)
       {  dodw = dokdoj[l][k][j] * dokdwj(bp,l-1,j,m,i,j);
       }else
       {  dodw    =  0;
```

```
            h  = l-1;
            for(ii=0;ii<nodes[h];ii++)
            {
             dodw += dokdoj[l][k][ii] * dokdwj(bp,l-1,ii,m,i,j);
            };
       };
      } else
      { printf(" error \n");
        sound(100);
        delay(100);
        nosound();
      };
 };
return(dodw);
}

float  dokdwbj(bpta *bp,int l,int k,int m,int i)
{
int ii,h;
int *nodes;
float ***dokdoj,**bout;
float dodw;

dokdoj= bp->dokdoj;
bout = bp->bout;
nodes = bp->nodes;

if(l==m)
 { if(k==i)
     dodw = bout[l][k];
   else
     dodw = 0;

 } else
 {  if(l>m)
     { if(l-1==m)
       {  dodw  = dokdoj[l][k][i] * dokdwbj(bp,l-1,i,m,i);
       }else
       {  dodw    = 0;
          h = l-1;
          for(ii=0;ii<nodes[h];ii++)
          dodw += dokdoj[l][k][ii] * dokdwbj(bp,l-1,ii,m,i);
       };
      } else
      { printf(" error \n");
        sound(100);
        delay(100);
        nosound();
      };
 };

return(dodw);
}
```

5. ไฟล์ GRAPHDAT.H และ GRAPHDAT.CPP

5.1 ไฟล์ GRAPHDAT.H

```
typedef struct {  float   minx,maxx,miny,maxy;
                  float   centerx,centery,diff;
                  float   minoutput,maxoutput;
               }  graph;

extern void   initgphc();
extern void   screengraph(graph pg,char textp[]);
extern void   screengtest(graph pg,char textp[]);
extern void   traingraph(float *target,float *outnn,int npixel,int itr,graph pg);
extern void   testgraph(float *outnn,int npixel,int itr,graph pg);
extern void   normalizedata(float *target,float *outnn,int npixel,graph pg);
extern void   normalizeTdata(float *outnn,int npixel,graph pg);
extern void   setgraph(graph *pg,float minpc,float maxpc,
                       float minoutput,float maxoutput);
```

5.2 ไฟล์ GRAPHDAT.CPP

```
#include "incmain.h"
#include "graphdat.h"

void initgphc();
void screengraph(graph pg,char textp[]);
void screengtest(graph pg,char textp[]);
void traingraph(float *target,float *outnn,int npixel,int itr,graph pg);
void testgraph(float *outnn,int npixel,int itr,graph pg);
void normalizdata(float *target,float *outnn,int npixel,graph pg);
void normalizeTdata(float *outnn,int npixel,graph pg);
void  setgraph(graph *pg,float minpc,float maxpc,float minoutput
               ,float maxoutput );

void initgphc()
{  int gd=DETECT,gm;
   clrscr();
  detectgraph(&gd,&gm);
  initgraph(&gd,&gm," ");

}

void screengraph(graph pg,char textp[])
{
  setfillstyle(SOLID_FILL,0);
  bar(pg.minx-5,pg.miny-5,pg.maxx+5,pg.maxy+5);
  setfillstyle(SOLID_FILL,14);
  bar(pg.minx-10,pg.miny-5,pg.minx-5,pg.maxy+5);
  bar(pg.maxx+10,pg.maxy+5,pg.minx-10,pg.maxy+30);
  bar(pg.minx-10,pg.miny-5,pg.maxx+10,pg.miny-40);
  bar(pg.maxx+5,pg.miny-5,pg.maxx+10,pg.maxy+10);
  setcolor(YELLOW);
  setlinestyle(0,0,1);
  line(pg.maxx+5,pg.centery,pg.maxx+15,pg.centery);
```

```
  line(pg.minx-15,pg.centery,pg.minx-5,pg.centery);
  setcolor(0);
  settextstyle(TRIPLEX_FONT,HORIZ_DIR,5);
  outtextxy(pg.minx+120,pg.miny-20," TARGET-OUTPUT & LEARNING-OUTPUT");
  outtextxy(75,pg.maxy+10,textp);
  setcolor(9);
  outtextxy(pg.minx+2,pg.miny-35,"Iteration");
  outtextxy(pg.maxx-85,pg.miny-35,"s.s.error");
}

void screengtest(graph pg,char textp[])
{
  setfillstyle(SOLID_FILL,0);
  bar(pg.minx-5,pg.miny-5,pg.maxx+5,pg.maxy+5);
  setfillstyle(SOLID_FILL,14);
  bar(pg.minx-10,pg.miny-5,pg.minx-5,getmaxy()-30);
  bar(pg.maxx+10,pg.maxy+5,pg.minx-10,pg.maxy+30);
  bar(pg.minx-10,pg.miny-5,getmaxx()-65,pg.miny-30);
  bar(pg.minx-10,getmaxy()-5,getmaxx()-65,getmaxy()-30);
  bar(getmaxx()-70,pg.miny-5,getmaxx()-65,getmaxy()-30);
  bar(pg.maxx+5,pg.miny-5,pg.maxx+10,pg.maxy+10);
  setcolor(YELLOW);
  setlinestyle(0,0,1);
  line(pg.maxx+5,pg.centery,pg.maxx+15,pg.centery);
  line(pg.minx-15,pg.centery,pg.minx-5,pg.centery);
  setcolor(0);
  settextstyle(TRIPLEX_FONT,HORIZ_DIR,5);
  outtextxy(pg.minx+180,getmaxy()-25,textp);
  setcolor(9);
  settextstyle(0,HORIZ_DIR,4);
  outtextxy(pg.minx+20,getmaxy()-130,"Testing");
}

void traingraph(float *target,float *outnn,int npixel,int itr,graph pg)
{
  int i=0,tt=0,oo=0;

  for(i=0;i<npixel;i++)
   {tt = target[i];
    oo = outnn[i];
    setfillstyle(SOLID_FILL,i+3);
    setlinestyle(0, 0, 1);
    setcolor(i+9);
    line(itr+pg.minx,tt,itr+pg.minx,tt);
      /*  tt = Target output   */
    setlinestyle(0, 0, 1);
    setcolor(i+3);
    line(itr+pg.minx,oo,itr+pg.minx,oo);
    /* oo = output of neural network */
    }

    }
```

```
void testgraph(float *outnn,int npixel,int itr,graph pg)
{ int i=0;
  setlinestyle(0, 0, 1);
  for(i=0;i<npixel;i++)
   {
    setfillstyle(SOLID_FILL,i+3);
    setcolor(i+3);
    line(itr+pg.minx,*(outnn+i),itr+pg.minx,*(outnn+i));
    /* outnn = output of neural network */
   };
}

void normalizedata(float *target,float *outnn,int npixel,graph pg)
{
int i;

 for(i=0;i<npixel;i++)
  {
 *(target+i) = pg.diff*(-*(target+i))/ (pg.maxoutput-pg.minoutput)+pg.centery;
 if(*(target+i)>pg.maxy) *(target+i)=pg.maxy;
 if(*(target+i)<pg.miny) *(target+i)=pg.miny;
 *(outnn+i) = pg.diff*(-*(outnn+i))/ (pg.maxoutput-pg.minoutput)+pg.centery;
 if(*(outnn+i)>pg.maxy) *(outnn+i)=pg.maxy;
 if(*(outnn+i)<pg.miny) *(outnn+i)=pg.miny;
  }
}

void normalizeTdata(float *outnn,int npixel,graph pg)
{
int i;

 for(i=0;i<npixel;i++)
  {
 *(outnn+i) = pg.diff*(-*(outnn+i))/ (pg.maxoutput-pg.minoutput) +pg.centery;
 if(*(outnn+i)>pg.maxy) *(outnn+i)=pg.maxy;
 if(*(outnn+i)<pg.miny) *(outnn+i)=pg.miny;
  }
}

void  setgraph(graph *pg,float minpc,float maxpc,float minoutput
               ,float maxoutput )
{
 pg->maxx =  getmaxx();
 pg->maxy  =  getmaxy();
 pg->minx  =  pg->maxx*minpc;
 pg->miny  =  pg->maxy*minpc;
 pg->maxx  =  maxpc*pg->maxx;
 pg->maxy  =  maxpc*pg->maxy;
 pg->centerx =  (pg->maxx+pg->minx)/2;
 pg->centery =  (pg->maxy+pg->miny)/2;
 pg->diff =  pg->maxy-pg->miny;
 pg->minoutput = minoutput;
 pg->maxoutput = maxoutput;
};
```

6. ไฟล์ BKPLEARN.H และ BKPLEARN.CPP

6.1 ไฟล์ BKPLEARN.H

```
extern void   sigmoid(bpta *bp,float *data,float *output);
extern float  sgm(long double x,int fsigmoid);
extern float  bsgm(long double x,int fsigmoid);

extern void   training(bpta *bp);
extern void   testing(bpta *bp);
extern void   listdata(bpta *bp,float *input,float *output,int re);
extern void   listTdata(bpta *bp,float *input,int ret);
extern void   adaptw(bpta *bp,float *data);
extern void   outputsigmoid(bpta *bp);
extern float  dokdwbj(bpta *bp,int l,int k,int m,int i);
extern float  dokdwj(bpta *bp,int l,int k,int m,int i,int j);
extern void   getrate(char string[30],int layer,float *rate);
```

6.2 ไฟล์ BKPLEARN.CPP

```
#include "incmain.h"
#include "recordnn.h"
#include "graphdat.h"
#include "bkplearn.h"

void training(bpta *bp);
void testing(bpta *bp) ;
void listdata(bpta *bp,float *input,float *output,int re);
void listTdata(bpta *bp,float *input,int ret);
char squarederror(bpta *bp,float *target,float *outy,int outnode,
                  int it,int npattern,char key);

void training(bpta *bp)
{
int i=0,numnode,npattern;
unsigned long d = 0,itra,niterstop;
int outnode,re = 1;
int k,na;
char sit[25],serror[25],fyname[25],ryname[25];
char fnamemr[25] = "itXssrY";
double error = 0.0,mrerror,showerror;
float errorlevel = 4.120820e-11;
char key ='z';
float *out,*input,*target;
char  s[25];
graph  pg;
FILE *fyout;
FILE *fmr;

bp->file = fopen(bp->name,"r");
fmr = fopen(fnamemr,"a");
input  = (float *)calloc(bp->nodes[0],sizeof(float));
outnode  = (bp->layer)-1;
npattern = bp->npattern;
```

```
itra  = bp->niterstart;
niterstop = bp->niterstop;
target = (float *)calloc(bp->nodes[outnode],sizeof(float));
out    = (float *)calloc(bp->nodes[outnode],sizeof(float));

 showerror = itra;
 initgphc();
 numnode=bp->nodes[outnode];
 setgraph(&pg,.1,.9,-1,1);
 screengraph(pg," Momemtum-rate    Learn-rate    Bias-rate    Reset    Quit ");
 for( i = 0 ;  ;  i++)
{
 listdata(bp,input,target,re);
 d++;
 sigmoid(bp,input,out);
 adaptw(bp,target);
 setfillstyle(SOLID_FILL,14);
 bar(pg.minx,pg.miny-12,pg.minx+95,pg.miny-22);

  if (d % npattern == 0)
  {
    itra++;
   };
  ultoa(itra,s,10);
   if (itra== 0)
   {
    setcolor(9);
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);
    outtextxy(pg.minx+4,pg.miny-20,"0");
   }
    else
   {
    setcolor(9);
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);
    outtextxy(pg.minx+4,pg.miny-20,s);
   };
key = chkbioskey();
     for (k = 0;k < numnode;k++)
     {
        float  temp =  target[k]-out[k];
        error += temp * temp;
      };

     if ( d % npattern == 0)
     {
     error  /= (npattern * numnode);
     if ((itra == showerror+1) && (d % npattern == 0))
     {
     mrerror = error;
     gcvt(error,5,serror);
     fprintf(fmr," %lu \t",itra);
     fprintf(fmr,"%e\n",mrerror);
     outtextxy(pg.maxx-83,pg.miny-20,serror);
     showerror = showerror+100;
     };
```

```
        if (itra == showerror)
      {
     setfillstyle(SOLID_FILL,14);
     bar(pg.maxx-85,pg.miny-12,pg.maxx+10,pg.miny-22);
     };

       };
 if (((error !=0) &&(error <= errorlevel) &&( d % npattern ==0))
    || (itra == niterstop))
        {   setfillstyle(SOLID_FILL,4);.
             bar(getmaxx()/2-180,getmaxy()/2+40,getmaxx()/2+170,getmaxy()/2);
             setcolor(15);
             settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);
outtextxy(getmaxx()/2-100,getmaxy()/2+20," Press any key to quit! ");
            sound(300);delay(20);
            sound(300);delay(20);
            sound(300);delay(20);nosound();
            key = 'q';
            getch();
           };
           if ( d % npattern  == 0)
             error   = 0.0;

    normalizedata(target,out,numnode,pg);
    traingraph(target,out,numnode,i,pg);
    if(i== (int) (pg.maxx-pg.minx))
    {  setfillstyle(SOLID_FILL,0);
       bar(pg.minx-5+1,pg.miny-5+1,pg.maxx+5-1,pg.maxy+5-1);
       setlinestyle(0,0,1);
       setcolor(YELLOW);
       line(pg.maxx+5,pg.centery,pg.maxx+15,pg.centery);
       line(pg.minx-5,pg.centery,pg.minx-15,pg.centery);
       i=0;
     };
    switch(key)
    {
     case 'l': getrate("Learn_rate",bp->layer,bp->l_rate); break;
     case 'b': getrate("Bias_rate",bp->layer,bp->b_rate); break;
     case 'm': getrate("Momentum_rate",bp->layer,bp->m_rate); break;
     case 'r': fclose(bp->file);bp->file = fopen(bp->name,"r");itra = 0;break;
    };

   if(key=='q')break;
};
 bp->niterstart = itra;
    closegraph();
    fclose(fmr);
    return;
}
```

```
void testing(bpta *bp)
{
int i,pattern;
int outnode;
char key='i',snum[10];
float *y;
float *input;
graph  pg;
int numnode;
int mintexty = 0,textline = 0;
float low = 0.3,high = 0.7;

 bp->file = fopen(bp->name,"r");
 outnode  = (bp->layer)-1;
 input    =(float *) calloc(bp->nodes[0],sizeof(float));
  y       =(float *) calloc(bp->nodes[outnode],sizeof(float));

initgphc();
numnode=bp->nodes[outnode];
setgraph(&pg,.1,.5,-1,1);
screengtest(pg," Reset       Quit ");
setcolor(9);
settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);
pattern = getpat("How many your patterns to testing ?",pg.minx+20,pg.miny-20);
setcolor(13);
settextstyle(TRIPLEX_FONT,HORIZ_DIR,2);

/*                      Result  Testing                 */
for (i =0;i < pattern;i++)
 {
  itoa(i+1,snum,10);
  listTdata(bp,input,0);
  sigmoid(bp,input,y);
if ((mintexty >= getmaxy()-45 ) ||((i % 39 == 0) && (i >38)))
{
getch();
textline = 0;
setfillstyle(SOLID_FILL,0);
bar(getmaxx()/2+20,50,getmaxx()/2+240,getmaxy()-33);
}
mintexty =textline+50;
 if (numnode == 3)
{
 if ((y[0] < low)  && (y[1] < low) && (y[2] > high))
 {
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is water" );
}else
 if ((y[0] < low)  && (y[1] > high) && (y[2] < low))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is EtOH" );
}else
 if ((y[0] < low)  && (y[1] > high) && (y[2] > high))
{
```

```
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is acetone" );
}else
 if ((y[0] > high)  && (y[1] < low) && (y[2] < low))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is ammonia" );
}else
 if ((y[0] > high)  && (y[1] < low) && (y[2] > high))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is acetaldehyde" );
}else
 if ((y[0] > high)  && (y[1] > high) && (y[2] < low))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is solution1" );
}else
 if ((y[0] > high)  && (y[1] > high) && (y[2] > high))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is solution2" );
}else
{
outtextxy(getmaxx()/2+104,mintexty,snum );
outtextxy(getmaxx()/2+40,mintexty,"Pattern    is unknown" );
}
};

 if (numnode == 5)
{
 if ((y[0] < low)  && (y[1] < low) && (y[2] < low) &&
          (y[3] < low) && (y[4] > high))
 {
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is water" );
}else
 if ((y[0] < low)  && (y[1] < low) && (y[2] < low) &&
          (y[3] > high) && (y[4] < low))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is EtOH" );
}else
 if ((y[0] < low)  && (y[1] < low) && (y[2] < low) &&
          (y[3] > high) && (y[4] > high))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is acetone" );
}else
 if ((y[0] < low)  && (y[1] < low) && (y[2] > high) &&
          (y[3] < low) && (y[4] < low))
{
outtextxy(getmaxx()/2+94,mintexty,snum );
outtextxy(getmaxx()/2+30,mintexty,"Pattern      is ammonia" );
}else
```

```
void listdata(bpta *bp,float *input,float *target,int re)
{ int  layer,i,numinput,numoutput;
  int outnode;
  float in;
  FILE *f;

start : f = bp->file;
layer     = bp->layer;
numinput  = bp->nodes[0];
outnode = layer -1;
numoutput = bp->nodes[outnode];
for(i=0;i<numinput;i++)
  {fscanf(f,"%f",&in);
  *(input+i) =    in;

if ( re == 1)
{
  if(chkeofnn(bp)==1)
   goto start;
}
 };
for(i=0;i<numoutput;i++)
 { fscanf(f,"%f",&in);
   *(target+i) =  in;
if (re == 1)
{
   if(chkeofnn(bp)==1)
         goto start;
}
   };
 return;
    }

void listTdata(bpta *bp,float *input,int ret)
{ int  i,numinput;
  float in;
  FILE *f;

start : f = bp->file;
numinput  = bp->nodes[0];

for(i=0;i<numinput;i++)
  {fscanf(f,"%f",&in);
  *(input+i) =    in;

if (ret == 1)
{
  if(chkeofnn(bp)==1)
     goto start;
 }
 };
 return;
    }
```

```
void getrate(char str[],int layer,float *rate)
{ int l,i,pos;
  char *str1 = " of  ",string[30];
  float nra;
  strcpy(string,str);
  for(i=0;i<30;i++)
    { if(string[i]=='\x0')
       {pos =i;break;};
    };
strcpy(string+pos,str1);
itoa(layer,string+pos+5,10);
strcpy(string+pos+6," layer =");
 nra = get(string,100,getmaxy()-8);
for(l=1;l<layer;l++)
 {

rate[l] = nra;
 };
return;
}
```

ประวัติผู้เขียน

นาย นิรันดร์ เลิศนิมิตธรรม เกิดวันที่ 3 สิงหาคม พ.ศ.2513 ที่ อำเภอคลองสาน จังหวัดกรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตรบัณฑิต สาขาไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยรังสิต ในปีการศึกษา 2534 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย