# Igor Pro实用教程代码

# 第1章

### 1.1.7

Variable v1,v2

 String s1=”hello”,s2

 Make/N=100 data1

 Print 2+sqrt(3)+5^3+6^0.3

 Print sin(1),ln(3),exp(2)

 Make/O/N=100 data1

 Setscale/I x,0,2\*pi,data1

 Data1=sin(x)

 Fft data1

 Make/O/N=100 data1

 Data1=x+gnoise(1)

 Curvefit line data1/D

 Display data1

 Edit data1

 Display data1

ModifyGraph mode(data1)=3,marker(data1)=19

 Sin(1) //error

 Print sin(1)

### 1.2.1

Make/O/N=100 data

Setscale/I x,0,1,data

Data=x

Edit data //只显示data内容

Edit data.xy //显示data内容和x坐标信息

Make/O/N=200 sinx;

Setscale/I x,0,2\*pi,sinx;

sinx=sin(x);

Display sinx;

Make/O/N=200/C cmpwave;

Make/O/T/N=200 twave;

Setdatafolder root;//设定当前目录

Newdatafolder tmp;//创建一个名为tmp的目录

Make/O/N=200 root:tmp:cmpwave;

setscale/P x,0,2\*pi/199,”s”,sinx

setscale/I x,0,2\*pi,”s”,sinx

Duplicate/O oldwave,newwave;

dupplicate/O/R=(x1,x2) oldwave,newwave;

dupplicate/O/R=[p1,p2] oldwave,newwave;

Make/O data={1,2,3}

Make/O data

Setscale/P x,0,0.001,data

Variable tau=0.01

Data=exp(-x/0.01)

Make/O data1

Data1=data[p]

### 1.2.5

Variable v1; //申明一个普通变量

Variable/C v1; //申明一个复数变量

Variable/G v1; //申明一个全局变量

String str; //申明一个字符串变量

### 1.2.6

Function func()

 String curr=getdatafolder(1) //获取当前目录

 Setdatafloder mydestfd //mydestfd应该存在，存放了要处理的数据

 //some thing to be done

 Setdatafolder curr //恢复当前目录

 End

### 1.2.11

Integrate1D(UserFunctionName, min\_x, max\_x [, options [, count ]])

Integrate [/DIM = d /METH=m /P/T][typeFlags ] yWaveA [/X = xWaveA ][/D = destWaveA ] [, yWaveB [/X = xWaveB ][/D = destWaveB ][, ...]]

# 第2章

### 2.1.1

make/N=100/O gaussfun,lorfun

setscale/I x,-1,1,gaussfun,lorfun

gaussfun=exp(-x\*x/0.01)

lorfun=1/((x\*x)+0.04)

make/N=10 XY\_Y={4,6,8,4,6,7,9,1,4,3}

make/N=10 XY\_X={3,5,1,4,7,5,8,6,9,2}

Display gaussfun

// 绘制gaussfun曲线

Display XY\_Y vs XY\_X as “mygraph”

//绘制XY曲线并且设定graph标题为mygraph

Display/L=newLeft/B=NewBottomgaussfun as "mygraph"

//以新建自由坐标轴newLeft和NewBottom绘制gaussfun曲线。

Preferences 0;

Display/L=newLeft/B=NewBottom gaussfun as "mygraph";

Graph1Style();

Preferences 1;

//以新建坐标轴newLeft和NewBottom为坐标轴绘制gaussfun曲线，并采用预定义样式 Graph1Style。

### 2.1.2

AppendToGraph/L=newLeft/B=newBottomlorfun

//向当前graph添加lorfun，坐标轴为newLeft和newBottom。

AppendToGraph/W=Graph0 lorfun//向名为“Graph0”的窗口添加lorfun曲线

2.2.2 Make/N=50 sinx,cosx; //创建两个wave

Setscale/I x,0,2\*pi,sinx,cosx; //设置两个wave的x坐标为0到2π

Sinx=sin(x); //设置wave sinx为它的横坐标的正弦

Cosx=cos(x); //设置wavecosx为它的横坐标的余弦

Make/N=(50,2) arrowinfo; //创建Arrowinfo信息wave

Setscale/I x,0,2\*pi,arrowinfo;//设置Arrowinfo wave的x坐标为0到2π，代表一圈

Arrowinfo[][0]=10; //设置所有的箭头长度都是10；

Arrowinfo[][1]=x; //设置随着角度的增加箭头偏转的方向

Display sinxvscosx; //显示sinx，并且以cosx为横坐标

ModifyGrapharrow Marker(sinx)={arrowinfo,1,5,0.5,1}

 //设置Marker为Arrow。这一步可以也可以通过曲线 //外观设置对话框设置箭头marker完成，注意Arrow //Info 要选择arrowinfo

Make/N=100 datax,datay,dataz

datax=enoise(2); datay=enoise(2); dataz=exp(-(datax^2+datay^2))

Display datay vs datax; ModifyGraph mode=3,marker=8

ModifyGraph zmrkSize(datay)={dataz,\*,\*,1,10}

### 2.2.3

make/O data

setscale/P x 0,1,data

data=exp(-x/60)

display data

ModifyGraph manTick(bottom)={0,30,0,0},manMinor(bottom)={3,2}

make/O InverseTemperature={ 30,20,14.2857,10,5,3.0303,2.22222,1.25}

make/O Mobility={ 0.211521,0.451599,0.612956,0.691259,0.886406,0.893136,0.921083,1}

Display Mobility vs InverseTemperature //绘制曲线，使用默认坐标轴

ModifyGraph mode=3,marker=8

make/N=5/T TickLable={“20”,”30”,”50”,”100”,”400”} //刻度线标签

make/N=5 TickPosition

TickPosition=450/str2num(TickLable)

ModifyGraph userticks(bottom)={TickPosition,TickLable}

### 2.2.4

make/O gaussdata

setscale/I x,-1,1,gaussdata

gaussdata=gauss(x,0,0.2)

display gaussdata

ModifyGraph zColor(gaussdata)={gaussdata,\*,\*,Rainbow256,0}

ModifyGraph mode=4,marker=19

ModifyGraph mskip=5

TextBox/C/N=text0/D={2,2,0}/A=MC "\\JC\\Z15\\F'Times New Roman'Gauss Function\re\\S(-x\\S2\\M\\Z14\\S/2\\F'symbol's\\F'Times New Roman'\\S2\\M\\Z14\\S)"

Twice \{K0} is \{K0\*2}

\{"Twice K0 is %g, and today is %s", 2\*K0, date()}

\{“x value=%g”,tagval(2)} //输出x的大小，每一个参数表示一种预定义类型

\{“mean value=%g”,mean(tagwaveref(),-inf,+inf} //输出引用wave的平均值，tagwaveref返回wave的引用。

### 2.4.3

make/O/N=(128,128) gaussdata=gauss(x,64,10,y,64,10)

function xyzcontour()

 make/O/N=(100,3) xyzwave

 variable i

 for(i=0;i<100;i+=1)

 xyzwave[i][0]=gnoise(1)

 xyzwave[i][1]=gnoise(1)

 xyzwave[i][2]=gnoise(1)

 endfor

 display;appendxyzcontour xyzwave

end

### 2.4.5

Make/O/N=(200,30) fdfun;

Setscale/I x,-1,0.2,fdfun;

fdfun=1/(exp(x\*1.6\*10000/1.38/30)+1);

duplicate/O fdfun,lorfun1,lorfun2;

lorfun1=0.005/((x+0.1)^2+0.01);

lorfun2=0.008/((x+0.3)^2+0.01);

fdfun=fdfun+lorfun1+lorfun2

newwaterfall fdfun;//绘制waterfall图

 Newwaterfall/Host=graphname wavename

### 2.4.6

Duplicate/O fdfun fdcolor;

Fdcolor=y;

ModifyGraph zColor(fdfun)={fdcolor,\*,\*,Rainbow256,0};

Modifywaterfall angle=45

### 2.4.7

Make/N=(100,100) gaussfun;

Setscale/I x,-0.5,0.5,gaussfun;

Gaussfun=exp(-x\*x/0.04-y\*y/0.04);

### 2.5.2

make/O/N=(100,100) data

setscale/I x,-pi,pi,data

setscale/I x,-pi,pi,data

data=sin(x)\*cos(y)

make/O/N=(20,20,3) dsphere

setscale/I x,0,pi,dsphere

setscale/I y,0,2\*pi,dsphere

dsphere[][][0]=1\*sin(x)\*cos(y)

dsphere[][][1]=1\*sin(x)\*sin(y)

dsphere[][][2]=1\*cos(x)

NewGizmo/N=Gizmo0/T="Gizmo0"

ModifyGizmo startRecMacro

AppendToGizmo Surface=root:dsphere,name=surface0

ModifyGizmo ModifyObject=surface0 property={ surfaceColorType,1}

ModifyGizmo ModifyObject=surface0 property={ fillMode,3}

ModifyGizmo ModifyObject=surface0 property={ srcMode,4}

ModifyGizmo ModifyObject=surface0 property={ frontColor,0.250004,0.996109,0.250004,1}

ModifyGizmo ModifyObject=surface0 property={ backColor,0.996109,0.664073,0,1}

ModifyGizmo setDisplayList=0, object=surface0

ModifyGizmo SETQUATERNION={0.488416,0.302182,0.455614,0.680110}

Make/O/N=(20,3) data=gnoise(5)

data[][2]=2\*data[p][0]-3\*data[p][1]+data[p][0]^2+gnoise(0.05)

make/N=(100,100)/O data

setscale/I x,-2,2,data

setscale/I y,-2,2,data

data=gauss(x,-1,0.5,y,-1,0.5)+gauss(x,-1,0.5,y,1,0.5)

data=data+gauss(x,1,0.5,y,-1,0.5)+gauss(x,1,0.5,y,1,0.5)

# 第3章

### 3.1.2

Make/O/N=100 trialwave;

Setscale/I x,-1,3,trialwave;

Trialwave=exp(-0.1\*x)+gnoise(0.02);

Display trialwave;

ModifyGraph mode=3;

function fitfun(coef,x):Fitfunc

 wave coef

 variable x

 <Exressions>

end

make/O coef={a1,a2,a3,…}

funcfit fitfun,coef,ydata/D

### 3.1.3

make/O LorF

 setscale/I x,-10,10,LorF

 setrandomseed 0.5

 LorF=1/(x\*x+1)+gnoise(0.05)

 display LorF

 ModifyGraph mode=3,marker=8

### 3.1.5

Make/O/N=(200,200) w2d

ModifyGraph lsize(fit\_LorF)=2,rgb(fit\_LorF)=(0,0,65280)

### 3.2.2

Make w

f(x)=a\*x\*x+b\*x+c

Make/O/N=200 trialdata

Setscale/P x,-2,0.02,trialdata

Setrandomseed 0.5

Trialdata=x\*x-2\*x-1+gnoise(0.2)

Display trialdata

ModifyGraph mode=3,marker=8,rgb=(0,0,65280)

### 3.2.4

Function f(w,x):FitFunc

 Wave w

 Variable x

End

Function f(w,x,y):FitFunc

 Wave w

 Variable x

 Variable y

 // function body

End

### 3.3.1

CurveFit [ flags ] fitType, [kwCWave=coefWaveName ,] waveName [ flag parameters ]

CurveFit fitType wavename

Make/O/N=20 trialdata

Setscale/I x,0,10,trialdata

Setrandomseed 0.5

Trialdata=exp(-4\*(x-5)^2)

Display trialdata

Modifygraph mode=3,marker=8

Variable V\_FitOptions=8 //control fitting

Curvefit gauss trialdata

Edit M\_iterates

K0=1;K1=2;K2=3;K3=4

CurveFit/G gauss trialdata

Make/O/N=200 trialdata

Setscale/I x 0,5,trialdata

Setrandomseed 0.5

Trialdata=exp(-(x-2)/2)+gnoise(0.1)

Display trialdata

Modifygraph mode=3,marker=8

Curvefit/TBOX=768 exp\_Xoffset trialdata /D

Curvefit/K={2} /TBOX=768 exp\_Xoffset trialdata /D

SetAxis left 0,4

SetAxis bottom -1,8

Curvefit/X/TBOX=896 exp\_Xoffset trialdata /D

make/O/N=20 trialdata

setscale/I x,0,2,trialdata

setrandomseed 0.5

trialdata=exp(-x)+gnoise(0.1)

display trialdata

modifygraph mode=3,marker=8

Curvefit/TBOX=768 exp trialdata /D/F={0.95,6,ErrorBar}

Make/O/N=200 trialdata

Setscale/I x,0,2,trialdata

Setrandomseed 0.5

Trialdata=2\*exp(-x)+gnoise(0.3)

Display trialdata

Modifygraph mode=3,marker=8

Curvefit/W=0 line trialdata /D/R

### 3.3.3

Make/O/N=200 data1

Setscale/I x,0,2\*pi,data1

Setrandomseed 0.5

Data1=2\*sin(x)+gnoise(0.2)

Display data1

Modifygraph mode=3,marker=8

K1 + K2 > 3

K3/5 < 2\*K1

K1\*K1 < 2。

-1 0 0 0

 1 0 0 0

0 -1 0 0

0 1 0 0

0 0 -1 0

0 0 1 0

0 0 0 -1

0 0 0 1

K0,K1,K2,K3

1,1,0,3,0,2,1,1

Edit M\_FitConstraint,W\_FitConstraintwave

### 3.4.1

Function FitEllipse(w,x,y):FitFunc

 Wave w

 Variable x

 Variable y

 //w[0]=a

 //w[1]=b

 //w[2]=x0

 //w[3]=y0

 //equition:x^2/a^2+y^/b^2=1

 Return (x-w[2])^2/w[0]^2+(y-w[3])^2/w[1]^2-1

end

Make/O/N=20 theta,ellipseX,ellipseY

Theta=2\*pi/20\*p

ellipseX=2\*cos(theta)+2

ellipseY=3\*sin(theta)+1

Display ellipseY vs ellipseX

Modifygraph mode=3, marker=8

Modifygraph width= {perUnit,72,bottom},height={perUnit,72,left}

Modifygraph width= {perUnit,72,bottom},height={perUnit,72,left}

Setrandomseed 0.5

ellipseX += gnoise(.3)

ellipseY += gnoise(.3)

Duplicate/O ellipseY, ellipseYFit, ellipseXFit

Make/D/O ellipseCoefs={3,2,1,2} // a, b, x0, y0

FuncFit/ODR=3 FitEllipse, ellipseCoefs /X={ellipseX, ellipseY}/XD={ellipseXFit,ellipseYFit}

AppendToGraph ellipseYFit vs ellipseXFit

Make/O/N=(100,100) elllipseContour

SetScale/I x -3,4.5,elllipseContour

SetScale/I y -3, 5, elllipseContour

elllipseContour = FitEllipse(ellipseCoefs, x, y)

AppendMatrixContour elllipseContour

ModifyContour elllipseContour labels=0,autoLevels={\*,\*,0},moreLevels=0,moreLevels={0}

ModifyContour elllipseContour rgbLines=(0,0,0)

### 3.4.2

Make/O/N=20 data1

Setscale/I x, -2,2,data1

Setrandomseed 0.5

Data1=abs(x)+gnoise(0.1)

Display data1

Modifygraph mode=3,marker=8

Function myfit(w,x):FitFunc

 Wave w

 Variable x

 If(x>=0)

 Return w[0]+w[1]\*x

 Endif

 If(x<0)

 Return w[2]-w[3]\*x

 Endif

End

Make/O/N=4 co

Co={0,1,0,1}

FuncFit/W=0/NTHR=0 myfit,co,data1 /D

### 3.4.3

Function myfitfunc(pw,yw,xw):FitFunc

 Wave pw,yw,xw

 Yw=<expression involving pw and xw>

End

duplicate yw tmp

<do interpolation or extrapolation work>

yw=tmp(x)

Make/O/N=200 data1

Setscale/I x,-2,2,data1

Setrandomseed 0.5

Data1=1/(x\*x+0.04)+gnoise(0.8)

Display data1

Modifygraph mode=3,marker=8

Proc proc1()

 Make/O/N=101 gsfun

 Setscale/I x,-0.5,0.5,gsfun

 Variable width=0.2/2/ln(2)

 Gsfun=exp(-x\*x/width^2)

 Variable v=sum(gsfun,-inf,inf)

 Gsfun=gsfun/v

 Duplicate/O data1 data2

 Convolve/A gsfun data1

 Appendtograph data2

End

Function myfunc(pw,yw,xw):FitFunc

 Wave pw,yw,xw

 Wave w=gsfun

 Yw=pw[0]+pw[1]/((xw-pw[2])^2+pw[3]^2)

 Convolve/A w yw

End

Make/O/N=4 co

Co={0,10,0,0.2}

FuncFit/W=0/NTHR=0 myfunc co data1 /D

Duplicate/O data2 data3

Data3=co[0]+co[1]/((x-co[2])^2+co[3]^2)

Display data2

Modifygraph mode=3,marker=8

Appendtograph data3

Legend/C/N=text0/A=MC

//普通拟合函数结构体变量

### 3.4.4

Structure myBasicFitStruct

 Wave coefw

 Variable x

 …

EndStructure

//all at once拟合函数结构变量

Structure myAllAtOnceFitStruct

 Wave coefw

 Wave yw

 Wave xw

 …

EndStructure

function myfitfunc(s):Fitfunc

 struct mystruct &s //mystruct为预先定义的结构体类型

 <do something>

end

Make/D/O/N=100 expData,expDataX

expDataX = enoise(0.5)+100.5

expData = 1.5+2\*exp(-(expDataX-100)/0.2) + gnoise(.05)

Display expData vs expDataX

ModifyGraph mode=3,marker=8

// The structure definition

Structure expFitStruct

 Wave coefw // required coefficient wave

 Variable x // required X value input

 Variable x0 // constant

EndStructure

// The fitting function

Function fitExpUsingStruct(s) : FitFunc

 Struct expFitStruct &s

 return s.coefw[0] + s.coefw[1]\*exp(-(s.x-s.x0)/s.coefw[2])

End

// The driver function that calls FuncFit:

Function expStructFitDriver(pw, yw, xw, xOff)

 Wave pw //coefficient wave

 Wave yw //data

 Wave xw //x value

 Variable xOff

 Variable doODR

 STRUCT expFitStruct fs

 fs.x0 = xOff

 FuncFit fitExpUsingStruct, pw, yw /X=xw /D /STRC=fs

 print pw

 Wave W\_sigma

 print W\_sigma

End

Make/D/O expStructCoefs = {1.5, 2, .2}

expStructFitDriver(expStructCoefs, expData, expDataX, 100)

Structure expFitStruct

 Wave coefw // 拟合参数wave，必须

 Variable x //x值，必须

 STRUCT WMFitInfoStruct fi // WMFitInfoStruct结构成员，可选

 Variable x0 // 常量，可选

EndStructure

//普通拟合函数：

Structure My2DFitStruct

 Wave coefw

 Variable x[2]

 …

EndStructure

//All at once拟合函数：

Structure My2DAllAtOnceFitStruct

 Wave coefw

 Wave yw

 Wave xw[2]

 …

EndStructure

### 3.4.6

Function FitManyGaussian(w, x) : FitFunc

 WAVE w

 Variable x

 Variable returnValue = w[0]

 Variable i

 Variable numPeaks = floor((numpnts(w)-1)/3)

 Variable cfi

 for (i = 0; i < numPeaks; i += 1)

 cfi = 3\*i+1

 returnValue += w[cfi]\*exp(-((x-w[cfi+1])/w[cfi+2])^2)

 endfor

 return returnValue

End

3.4.7 Make/O/N=200 line1;

Setscale/I x,0,10,line1;

Line1=x+enoise(0.1);

Display line1;

ModifyGraph marker=16;

Display newtonring vs ringnum;

ModifyGraph mode=3,marker=16;

Display resistance vs temperature

# 第4章

### 4.1.1

Make/O/N=100 w

Setscale/I x,0,2\*pi,w

w=sin(x)

Print w(0.1),sin(0.1)

Duplicate w1 wdest

Setscale/I x,x1,x2,wdest

Wdest=w1(x)+w1(2\*x0-x)

Make/O data={3,2,5,3,1,5,8,9}

Make/O xw={1,2.1,3,3.9,5,5.8,7,8}

Print interp(2,xw,data)

Make/O/N=10 xData, yData // Make source data

xData = p; yData = 3 + 4\*xData + gnoise(2) // Put sample data in them

Display yData vs xData // Make a graph

Modify mode=2, lsize=3 // Display source data as dots

Interpolate2/N=200/T=2/E=2/Y=ydata\_CS xData, ydata

AppendToGraph yData\_CS; Modify rgb(yData\_CS)=(0,0,65535)

Redimension/N=500 xData, yData

xData = p/50; yData = 10\*sin(xData) + gnoise(1.0)

Modify lsize(yData)=1

Interpolate2/N=200/T=2/E=2/Y=ydata\_CS xdata, ydata

AppendToGraph yData\_CS; Modify rgb(yData\_CS)=(0,0,65535)

Make/N=(100,100)/O ydata

Setscale/I x,-1,1,ydata

Setscale/I y,-1,1,ydata

Ydata=x\*x+y\*y

Print interp2D(ydata,0.1,0.2), 0.1\*0.1+0.2\*0.2

Make/O/N=(10,20,30) ddd=gnoise(10)

Print interp3d(ddd,1,0,0)

Make/O/N=(10,4) ddd=gnoise(10)

Triangulate3D/OUT=1 ddd

Print interp3d(ddd,1,0,0,M\_3DVertexList)

Interp3DPath 3dWave tripletPathWave

Ydata\_new=interp2d(ydata,M-1(x,y))

### 4.1.2

Make/O/N=100 xData = .01\*x + gnoise(.01)

Make/O/N=100 yData = 1.5 + 5\*exp(-((xData-.5)/.1)^2)

Display yData vs xData

 Sort xdata,xdata,ydata

Duplicate/O yData, wData

SetScale/I x 0, 1, wData

wData = interp(x, xData, yData)

AppendToGraph wData

interp(x1, xwaveName, ywaveName )

Interpolate2/T=2/N=200/E=2/Y=yData\_CS xData, yData

Make/n=(tot,4) a3ddata

Make/N=(n1,n2,n3) new3d

//set new3d x,y and z scale

//here needs some loops to get x0,y0,z0

//loop begins

TransformXYZ (x1,y1,z1,x0,y0,z0)

A3ddata[i][0]=x1

A3ddata[i][1]=y1

A3ddata[i][2]=z1

A3ddata[i][3]=v //v表示（x0,y0,z0）处的测量值

i=i+1

//loop ends

Triangulate3D a3ddata

Wave M\_3DVertexList

New3d=interp3d(srcwave,x,y,z, M\_3DVertexList)

ImageInterpolate/S={-3,0.01,1,-2,0.01,2} Voronoi myTripletWave

ImageInterpolate/STW/S={0,1,0,1,1,1} Voronoi myTripletWave

ImageInterpolate/PTW=W\_TriangulationData/S={-3,0.02,1,-2,0.02,2} Voronoi my1DZWave

My1DZWave= myTripletWave[2][p]

ContourZ(graphNameStr, contourNameStr, instance, x, y)

空字符串，

can use ContourZ to convert XYZ triplet data into gridded data.

// Make example XYZ triplet contour data

Make/O/N=50 wx,wy,wz

wx= enoise(2) // x = -2 to 2

wy= enoise(2) // y = -2 to 2

wz= exp(-(wx[p]\*wx[p] + wy[p]\*wy[p])) // XY gaussian, z= 0 to 1

// ContourZ requires a displayed contour data set

Display; AppendXYZContour wz vs {wx,wy};DelayUpdate

ModifyContour wz autoLevels={\*,\*,0}

ModifyContour wz xymarkers=1 // show the X and Y locations

// Set the null (out-of-XY domain) value

ModifyContour wz nullValue=NaN // default is min(wz) - 1

// Convert to grid: Make matrix that spans X and Y

Make/O/N=(30,30) matrix

SetScale/I x, -2, 2, "", matrix

SetScale/I y, -2, 2, "", matrix

matrix= ContourZ("","wz",0,x,y) // or = ContourZ("","",0,x,y)

AppendImage matrix

### 4.1.3

BinarySearch(waveName,val)

Make/O data = {1, 2, 3.3, 4.9} // Monotonic increasing

Print BinarySearch(data,3)

BinarySearchInterp(waveName,val)

Make/O data = {1, 2, 3.3, 4.9} // Monotonic increasing

Print BinarySearchInterp(data,3)

Print data[1.76923]

duplicate w,w1

w1=leftx(w)+deltax(w)\*p

variable x0=interp(y0,w,w1)

FindLevel [/B=box /EDGE=e /P/Q/R=(startX,endX ) ] waveName, level

Make/O/N=100 data1

Setscale/P x,0,0.01,data1

Data1=2\*sin(2\*pi\*3\*x)+gnoise(0.1)

Display data1

Findlevel/B=5/R=(0.6,0.8) data1,1

Make/O/N=2 ylevel

Setscale/I x,0,1,ylevel

Ylevel=1

Make/N=2/O xcross\_x,xcross\_y={-2,2}

Xcross\_x=V\_LevelX

Appendtograph ylevel

Appendtograph xcross\_y vs xcross\_x

FindLevels [/B=box /D=destWaveName /DEST=destWaveName /EDGE=e /M=minWidthX /N=maxLevels /P/Q /R=(startX,endX)]waveName, level

findlevels/B=5 data1,1

edit W\_FindLevels

### 4.1.4

Smooth [ /B[=b ] /DIM = d /E=endEffect /F[=f ] /M=threshold /MPCT=percentile /R=replacement /S=sgOrder ] num, waveName [,waveName ]...

Smooth num,wavename

Make/O data={1,2,4,3,6,5,7}

Duplicate/O data data1

Display data

Smooth/B 3,data1

Appendtograph data1

ModifyGraph mode(data)=3,marker(data)=19

Edit data,data1

Make/O/N=100 data= enoise(1)>.9 ? NaN : sin(x/8) // signal with NaNs

Duplicate/O data, dataMedian

Smooth/M=(NaN) 5, dataMedian // replace (only) NaNs with 5-point median

Display dataMedian, data; ModifyGraph rgb(data)=(0,0,65535); Legend

ModifyGraph offset(data)={0,1}

Loess [/CONF={confInt, ciPlusWaveName [,ciMinusWaveName]} /DEST=destWaveName /DFCT[={destFactorWaveName1 [,destFactorWaveName...]}] /E=extrapolate /N=neighbors /ORD=order /PASS=passes /R=robust /SMTH= sf /TIME=secs /V=verbose /Z=z ] srcWave=srcWaveName [, factors=factorWaveName1 [, factorWaveName2...]]

Loess sorWave=srcWaveName

Make/O/N=200 wv=2\*sin(x/8)+gnoise(1)

KillWaves/Z smoothed // 如果该wave存在，就删除掉

Loess/DEST=smoothed/N=50 srcWave=wv

Display wv; ModifyGraph mode=3,marker=19

AppendtoGraph smoothed; ModifyGraph rgb(smoothed)=(0,0,65535)

Make/O/N=100 short=2\*cos(x/4)+gnoise(1)

Make/O/N=300 out; SetScale/I x, 0, 99, "" out

Loess/DEST=out/DFCT/N=30 srcWave=short

Display short; ModifyGraph mode=3,marker=19

AppendtoGraph out

ModifyGraph rgb(out)=(0,0,65535),mode(out)=2,lsize(out)=2

// NOx

Make/O/D NOx = {4.818, 2.849, 3.275, 4.691, 4.255, 5.064, 2.118, 4.602, 2.286, 0.97, 3.965, 5.344, 3.834, 1.99, 5.199, 5.283, 3.752, 0.537, 1.64, 5.055, 4.937, 1.561};

// X wave (注意坐标值大小并没有排序)

Make/O/D EquivRatio = {0.831, 1.045, 1.021, 0.97, 0.825, 0.891, 0.71, 0.801, 1.074, 1.148, 1, 0.928, 0.767, 0.701, 0.807, 0.902, 0.997, 1.224, 1.089, 0.973, 0.98, 0.665};

Make/O/D/N=100 fittedNOx

WaveStats/Q EquivRatio

SetScale/I x, V\_Min, V\_max, "", fittedNOx

Loess/CONF={0.99, cp, cm}/DEST=fittedNOx/DFCT/SMTH=(2/3) srcWave=NOx, factors={EquivRatio}

Display NOx vs EquivRatio; ModifyGraph mode=3,marker=19

AppendtoGraph fittedNOx, cp,cm // fit and confidence intervals

ModifyGraph rgb(fittedNOx)=(0,0,65535)

ModifyGraph mode(fittedNOx)=2,lsize(fittedNOx)=2

### 4.2.1

Differentiate ywave

function calgauss(inx)

 variable inx

 return gauss(inx,0,1)

end

make/N=200/O pDistribution

setscale/I x,-20,20,pDistribution

pDistribution=integrate1d(calgauss,-100,x,0)

duplicate/O pDistribution fDistribution

differentiate fDistribution

Differentiate [/DIM = d /EP=e /METH=m /P ][typeFlags ] yWaveA [/X=xWaveA ] [/D=destWaveA ][, yWaveB [/X=xWaveB ][/D=destWaveB ][, ...]]

Differentiate/METH=0 wave0 /D=w1

Differentiate/METH=1 wave0 /D=w2

Differentiate/METH=2 wave0 /D=w3

Appendtotable w1,w2,w3

Display w1,w2,w3

ModifyGraph mode=4,marker(w1)=8,marker(w2)=5,rgb(w2)=(0,0,0),marker(w3)=6

ModifyGraph rgb(w3)=(0,0,65280)

Legend/C/N=text0/A=MC

integrate ywave

Make/O/N=200 data1

Setscale/I x,0,2\*pi,data1

Data1=sin(x)

Integrate data1 /D=inte\_data1

Duplicate/O data1 data2

Data2=1-cos(x)

Display inte\_data1,data2

ModifyGraph mode(inte\_data1)=3,marker(inte\_data1)=8,rgb(data2)=(0,0,0)

ModifyGraph lsize(data2)=2

function diracfun(inX)

 variable inX

 nvar/Z a

 variable a1=0.00001

 if(nvar\_exists(a))

 a1=a

 endif

 return 1/pi\*a1/(a1\*a1+inX\*inX)

end

variable a=0.000001

print diracfun(0.001),diracfun(0)

print integrate1D(diracfun,-20,20,2)

function inte\_f(x0)

 variable x0

 return (2\*x0\*x0-3\*x0+1)\*diracfun(x0-2)

end

a=0.0000001

print integrate1D(inte\_f,-20,20,2)

mean(w,[x1,x2])

### 4.2.2

WaveStats /Q [/R = (startX, endX )]/Z waveName

Make/O/N=200 data

Data=x

Wavestats data

function fun1()

 wave w=data

 wavestats/Q w

 print V\_avg

end

WaveStats/R=(12.75,13.32) wave

V\_avg = (55+88+100+87)/4 = 82.5

Sum(wave,12.75,13.32) =55+88+100+87 = 330

mean(wave,12.75,13.32) = (55+88+100+87)/4 = 82.5

area(wave,12.75,13.32) = 0.05 · (43+55) / 2 first trapezoid

 + 0.20 · (55+88) / 2 second trapezoid

 + 0.20 · (88+100) / 2 third trapezoid

 + 0.12 · (100+92.2) / 2 fourth trapezoid

 = 47.082

faverage(wave,12.75,13.32) = area(wave,12.75,13.32) / (13.32-12.75)

 = 47.082/0.57 = 82.6

### 4.2.3

Make/D/O polycoef={1.5,-1.25,-3.75,0,1}

Findroots/P=polycoef

Edit w\_polyroots

Function myFunc(w,x)

 Wave w

 Variable x

 return <an arithmetic expression>

End

Function mySinc(w, x)

 //w[0]:a

 //w[1]:b

 //w[2]:c

 //w[3]:x0

 Wave w

 Variable x

 return w[0]+w[1]\*sinc(w[2]\*(x-w[3]))

End

Make/D/O SincCoefs={0, 1, 2, .5}

Make/D/O SincWave

SetScale/I x -10,10,SincWave

SincWave = mySinc(SincCoefs, x)

Display SincWave

ModifyGraph zero(left)=1

ModifyGraph minor(bottom)=1

FindRoots/L=1/H=3 mySinc, SincCoefs

FindRoots/L=0/H=5 mySinc, SincCoefs

FindRoots mySinc, SincCoefs

Function myFunc1(w, x1, x2)

 Wave w

 Variable x1, x2

 return <an arithmetic expression>

End

Function myFunc2(w, x1, x2)

 Wave w

 Variable x1, x2

 return <an arithmetic expression>

End

Function myf1(w, xx, yy)

 Wave w

 Variable xx,yy

 return w[0]\*sin(xx/w[1])\*cos(yy/w[2])

End

Function myf2(w, xx, yy)

 Wave w

 Variable xx,yy

 return w[0]\*cos(xx/w[1])\*tan(yy/w[2])

End

Make/D/O params2D={1,5,4}

Findroots myf1,params2D, myf2,params2D

FindRoots/X={7.7,6.3} myf1,params2D, myf2,params2D

FindRoots [/H=highBracket /L=lowBracket ] [/I=maxIters ] [/Q] [/T=tol ] [/X=startXSpec ] funcspec, pWave [, funcspec, pwave [, ...]]

FindRoots /P=PolyCoefsWave

Function myCombinedFunc(w, xW, yW)

 Wave w, xW, yW

 yW[0] = f1(w[…][0], xW[0], xW[1],..., xW[N-1])

 yW[1] = f2(w[…][1], xW[0], xW[1],..., xW[N-1])

 ...

 yW[N-1] = fN(w[…][N-1], xW[0], xW[1],..., xW[N-1])

End

Make/N=(M,N) paramWave

<填充参数>

Make/N=(N) guessWave

guessWave = {x0, x1, ... , xN-1}

FindRoots /X=guessWave myCombinedFunc, paramWave

function myf3(w,xw,yw)

 wave w,xw,yw

 yw[0]=w[0][0]\*xw[0]+w[1][0]\*xw[1]+w[2][0]\*xw[2]+w[3][0]

 yw[1]=w[0][1]\*xw[0]+w[1][1]\*xw[1]+w[2][1]\*xw[2]+w[3][1]

 yw[2]=w[0][2]\*xw[0]+w[1][2]\*xw[1]+w[2][2]\*xw[2]+w[3][2]

end

make/O/N=(4,3) pw={{1,1,1,-6},{2,-1,3,-9},{1,2,-1,-2}}

Make/N=3 guesswave={0,0,0}

Findroots/X=guesswave myf3,pw

### 4.2.4

Function D(pw, xx, yw, dydx)

 Wave pw // 参数wave

 Variable xx //x值

 Wave yw // y值，wave表示多个微分方程

 Wave dydx // dyi/dx

 dydx[0] = < expression for one derivative>

 dydx[1] = < expression for next derivative>

 < etc.>

 return 0

End

Function FirstOrder(pw, xx, yw, dydx)

 Wave pw // pw[0] 保存a

 Variable xx // 本例没有使用

 Wave yw // 只有一个元素，因为只有一个微分方程

 Wave dydx //只有一个元素，因为只有一个微分方程

 dydx[0] = -pw[0]\*yw[0] //只有一个表达式

 return 0

End

Make/D/O/N=101 YY //存放解

YY[0] = 10 // 初始条件 y0=10

Display YY // 显示解

Make/D/O PP={0.05} // 设置参数为a=0.05

IntegrateODE FirstOrder, PP, YY

SetScale/P x 0,3,YY //x坐标从0开始，间隔为3

IntegrateODE FirstOrder, PP, YY

IntegrateODE/X={0,3} FirstOrder, PP, YY

Make/D/O/N=101 XX

XX = exp(p/20)

Display YY vs XX

ModifyGraph mode=2

IntegrateODE/X=XX FirstOrder, PP, YY

Function chaosfun(pw, tt, yw, dydt)

 Wave pw

 Variable tt

 Wave yw

 Wave dydt

 Variable gv1

gv1=(-0.409091\*yw[0]-0.174235\*(abs(yw[0]+1.9565)-abs(yw[0]-1.9565)))/1000

 //pw[0]:1/C1

 //pw[1]:R0

 //pw[2]:1/C2

 //pw[3]:1/L

 dydt[0] = pw[0]/pw[1]\*(yw[1]-yw[0])-pw[0]\*gv1

 dydt[1] = pw[2]\*yw[2]-pw[2]/pw[1]\*(yw[1]-yw[0])

 dydt[2] = -pw[3]\*yw[1]

 return 0

End

make/O coef={100000000,1856,10000000,55.556}

make/N=(20000,3) iuu

setscale/P x,0,0.001,iuu

iuu[0][0]=1 //设置初始值

iuu[0][1]=0

iuu[0][2]=0

display iuu[][1] vs iuu[][0]

ModifyGraph mode=2,rgb=(0,0,0)

integrateode chaosfun,coef,iuu

Coef[1]=1970

integrateode chaosfun,coef,iuu

Function Harmonic(pw, tt, yy, dydt)

 Wave pw // pw[0]=λ, pw[1]=ω// pw[2]=F, pw[3]=F frequency

 Variable tt

 Wave yy // yy[0] = v, yy[1] = y

 Wave dydt

 Variable Force = pw[2]\*sin(pw[3]\*tt) // 正弦驱动力

 dydt[0] = -2\*pw[0]\*yy[0] - pw[1]\*pw[1]\*yy[1]+Force

 dydt[1] = yy[0]

 return 0

End

Make/D/O/N=(300,2) HarmonicOsc

SetDimLabel 1,0,Velocity,HarmonicOsc

SetDimLabel 1,1,Displacement,HarmonicOsc

HarmonicOsc[0][%Velocity] = 5 //初速度

HarmonicOsc[0][%Displacement] = 0 // 初始位移

Make/D/O HarmPW={.01,.5,.1,.45} // 阻尼,固有频率,驱动力幅度，驱动力频率

Display HarmonicOsc[][%Displacement]

IntegrateODE Harmonic, HarmPW, HarmonicOsc

Make/D/O/N=1000 FreeRunY

FreeRunY = NaN

FreeRunY[0] = 10 // 初始条件 y0=10

Make/O/D/N=1000 FreeRunX

FreeRunX = NaN

FreeRunX[0] = 0 //初始值y0对应的x坐标

Display FreeRunY vs FreeRunX

ModifyGraph mode=3, marker=19

Make/D/O PP={0.05}

IntegrateODE/M=1/X=FreeRunX/XRUN={1,100} FirstOrder, PP, FreeRunY

Redimension/N=(V\_ODETotalSteps+1) FreeRunY, FreeRunX

IntegrateOD/R=[n1,n2]…

Redimension/N=(500,2) HarmonicOsc

Display HarmonicOsc[][%Displacement]

IntegrateODE/M=1/R=[,300] Harmonic, HarmPW, HarmonicOsc

IntegrateODE/M=1/R=[300,400] Harmonic, HarmPW, HarmonicOsc

IntegrateODE/M=1/R=[400,500] Harmonic, HarmPW, HarmonicOsc

IntegrateODE [/E=eps /F=errMethod /M=m /Q= quiet /R=(startX, endX) /S=waveName /STOP={stopWave, mode} /U=u /X=xvaluespec /XRUN={dx0, Xmax} /CVOP={solver, jacobian, extendedErrors}] derivFunc, cwaveName, ywaveSpec

### 4.2.5

Make/N=4/O scorewithgrade={2,11,11,8,4}

Make/T/N=4/O gradeinfo={“A+”,”A”,”B”,”C”,”D”}

Display scorewithgrade vs gradeinfo

SetAxis left 0,12

Make/O score={0,15,18,23,30,31,35,40,41,43,45,47,50,51,52,58,60,63,67,69,70,71,73,75,77,79,80,81,83, 89,90,92,95,98,100,100}

Reverse score\_Hist

Display score\_hist vs gradeinfo

Histogram [/A/B={binStart,binWidth,numBins}/B=mode /P/R=(startX, endX) /C/N/P/R=(startX,endX )/R=[startP,endP ]/W=weightWave ] srcWaveName, destWaveName

Make/N=1024 noise = gnoise(1) // make raw data

Make hist // make destination for histogram

Histogram/B={-3, (3 - -3)/100, 100} noise, hist // do histogram

Display hist; Modify mode(hist)=1

setscale/P x leftx(hist)+deltax(hist)/2, deltax(hist),hist

CurveFit gauss hist /D // curve fit to histogram

4.2.6Sort [ /A /DIML /C /R ] sortKeyWaves, sortedWaveName [, sortedWaveName ]...

Make/O data={1,2,3,4,5}

Make/O sortkey={2,1,3,4,5}

Duplicate/O data data1

Sort sortkey data

Edit data sortkey data1

Make/O data={3,5,1,2,7,4}

Duplicate/O data data1

Sort data data

Edit data data1

make/O data={1,2,3,4,5}

make/O sortkey={7,6,3,3,8}

duplicate/O data data1

sort sortkey data

edit data1 sortkey data

make/O data={1,2,3,4,5}

make/O sortkey1={7,6,3,3,8}

make/O sortkey2={1,1,1,2,1}

duplicate/O data data1

sort {sortkey1,sortkey2} data

edit data1 sortkey1 sortkey2 data

4.3.1 Make /O/N=120 data

Setscale/P x 0,0.001,”s”,data

Data=cos(100\*pi\*x)

Display data

Print dimsize(data,0)

fft data

Print dimsize(data,0)

Make /O/N=120 data

Setscale/P x 0,0.001,”s”,data

Data=cos(100\*pi\*x)

Display data

fft data

Print deltax(data)

Make/O/N=128 wave0

SetScale/P x 0,1e-3,"s",wave0 // 采样间隔=1ms,奈奎斯特频率=500Hz

wave0= 1 - cos(2\*Pi\*125\*x) // 信号频率=125Hz,振幅= -1

Display wave0;ModifyGraph zero(left)=3

Make/O/N=128 wave0

SetScale/P x 0,1e-3,"s",wave0 // 采样间隔=1ms,奈奎斯特频率=500Hz

wave0= 1 - cos(2\*Pi\*150\*x) // 信号频率=150Hz,振幅= -1

Display wave0;ModifyGraph zero(left)=3

Fft wave0

 FFT /RP=[startPoint,endPoint ] /RX=(startX,endX ) /PAD={dim1 [, dim2, dim3, dim4 ]} /Z [/COLS/HCC/HCS/ROWS/REAL/MAG/MAGS/OUT=mode ] /WINF=windowKind /DEST=destWave srcWave

### 4.3.2

Make/O/N=128 cosWave

Setscale/P,x,0,0.001,coswave

cosWave =cos(2\*pi\*125\*x)

Display cosWave

ModifyGraph mode=4,marker=8

Rotate 3,cosWave // 将末尾的数据依次接到开头

SetAxis bottom -0.005,0.01

Make/O/N=128 cosWave

Setscale/P x,0,0.001,coswave

cosWave=cos(2\*pi\*125\*x)

Display cosWave

fft coswave

Make/O/N=128 cosWave

Setscale/P x,0,0.001,cosWave //采样间隔1ms

cosWave=cos(2\*pi\*x\*137) //频率为137Hz

Display cosWave

ModifyGraph mode=4,marker=8

 Rotate 3,cosWave

SetAxis bottom -0.005,0.01

Make/O/N=128 cosWave

Setscale/P x,0,0.001,cosWave //采样间隔1ms

cosWave=cos(2\*pi\*x\*137) //频率为137Hz

Display cosWave

Fft coswave

SetAxis/A

ModifyGraph cmplxMode=3

Make/O/N=128 cosWave

Setscale/P x,0,0.001,cosWave //采样间隔1ms

cosWave=cos(2\*pi\*x\*137) //频率为137Hz

Display cosWave

Hanning cosWave

Fft coswave

SetAxis/A

ModifyGraph cmplxMode=3

Make/O/N=128 cosWave

Setscale/P x,0,0.001,cosWave //采样间隔1ms

cosWave=cos(2\*pi\*x\*137) //频率为137Hz

Display cosWave

fft/winf=hanning coswave

display w\_fft

 ModifyGraph mode=4,marker=8,cmplxMode=3

### 4.3.3

HilbertTransform [ /Z][/O][/DEST= destWave ] srcWave

Make/O/N=512 cosWave=cos(2\*pi\*x\*20/512)

HilbertTransform/Dest=hCosWave cosWave

Display cosWave,hCosWave

ModifyGraph rgb(hCosWave)=(0,0,65535)

Duplicate/O coswave sinwave

Sinwave=-sin(2\*pi\*x\*20/512)

Display sinwave,hcoswave

Make/O/N=512 cosWave=cos(2\*pi\*x\*20/512)

Fft/dest=fftcoswave coswave

Duplicate/O fftcoswave wave1

Wave1=-cmplx(0,1)\*cmplx(sign(x),0)

Fftcoswave=fftcoswave\*wave1

Ifft/dest=wave2 fftcoswave

Display wave2

### 4.3.4

Make/O data1={1,3,2,5,7} //N1=5

Make/O srcdata={2,5,3} //N2=3

Duplicate/O data1 data2,data3

Convolve srcdata data1

Convolve/C srcdata data2

Redimension/N=7 data3 //N1+N2-1

Convolve/C srcdata data3

Edit data1 data2 data3

Convolve [/A/C] srcWaveName, destWaveName [, destWaveName ]...

Make/O/N=100 gsfun

Make/O/N=100 f1,f2

Setscale/I x,-0.2,0.2,gsfun

Setscale/I x,-0.8,0.2,f1,f2

Gsfun=exp(-x\*x/2/0.0001)

K1=sum(gsfun,-inf,inf)

Gsfun=gsfun/K1 //归一化

f1=1/(exp(11670\*x/30)+1)

f2=1/(exp(11670\*x/30)+1)

convolve gsfun,f1

convolve/A gsfun,f2

display f1

display f2

Make/O/N=100 f3

Setscale/I x,-0.8,0.2,f3

f3=f1[p+50]

display f3

Correlate [/AUTO/C/NODC] srcWaveName, destWaveName [, destWaveName ]...

### 4.3.5

Make/O/N=100 gs1,gs2

Setscale/I x,-0.3,0.3,gs1,gs2

Gs1=exp(-(x-0.1)^2/0.01)

Gs2=exp(-(x+0.1)^2/0.01)

Display gs1

appendtograph gs2

ModifyGraph rgb(gs2)=(0,0,65280)

Legend/C/N=text0/A=MC

Duplicate/O gs2 gs3

Display gs2

Correlate gs1 gs2

print leftx(gs2),numpnts(gs2)

Print -0.3-deltax(gs2)\*99

Duplicate/O gs3 gs2 // restore gs2

Correlate/auto gs1 gs2

Make/O/N=256 reverse\_lUK=255-x

### 4.4.1

Duplicate/O MRI R\_MRI

R\_MRI=reverse\_LUK[R\_MRI]

Newimage MRI

Newimage M\_MRI

Make/O/N=256 reverse\_lUK=1-x/255

ModifyImage MRI ctabAutoscale=0,lookup= reverse\_lUK

Variable Gamma

 Make/O/N=100 contY,contX

 contX= p/99

 contY= contX^(1/Gamma) //Lookup table:contY

Make/O/N=3 userX,userY

userX=x/2

userY=x/2

make/O/N=1000 luk\_w

setscale/I x,0,1,luk\_w

display userY vs userX

GraphWaveEdit /M userY

Luk\_w= userY(BinarySearchInterp(userX, x))

Newimage MRI

ModifyImage MRI ctabAutoscale=0,lookup= luk\_w

### 4.5.2

Newimage MRI

Imagehistmodification/W MRI

NewImage M\_ImageHistEq

Imagehistogram MRI

Duplicate W\_ImageHist ori\_hist

Imagehistogram M\_ImageHistEq

duplicate W\_ImageHist after\_hist

display ori\_hist

ModifyGraph axisEnab(left)={0,0.4}

AppendToGraph/L=L1 after\_hist

ModifyGraph axisEnab(L1)={0.6,1},freePos(L1)={0,bottom}

ModifyGraph rgb(after\_hist)=(0,0,0)

Legend/C/N=text0/F=0/A=MC

ImageHistModification MRI

Duplicate/O M\_ImageHistEq, globalHist

NewImage globalHist

ImageTransform/N={2,7} padImage MRI

ImageHistModification/A/C=10/H=2/V=2 M\_paddedImage

NewImage M\_ImageHistEq

### 4.5

make/O/N=10000 noisedata=enoise(1)

make/O/N=50 pdis

histogram/B=1 noisedata,pdis

integrate pdis

pdis/=pdis[49]

differentiate pdis

display pdis

ModifyGraph mode=1

make/O/N=10000 noisedata=gnoise(1)

make/O/N=50 pdis

histogram/B=1 noisedata,pdis

integrate pdis

pdis/=pdis[49]

differentiate pdis

display pdis

ModifyGraph mode=1

CurveFit/M=2/W=0 gauss, pdis/D

Print k3/sqrt(2)

# 第5章

### 5.1.1

function myfun()

 string s="This ia my first function in Igor\r"

 s+="Please give two numbers, I'll calculate the sum."

 doalert 0,s

 variable v1,v2

 prompt v1,"Input the first number"

 prompt v2,"Input the second number"

 doprompt "Calculate Sum",v1,v2

 string s1="The sum of "+num2str(v1)+" and "+num2str(v2)+" is "+num2str(v1+v2)

 doalert 0,s1

end

### 5.1.4

Macro macroname(parameters list)

 body

end

proc procname(parameters list)

 body

end

function functionname(parameters list)

 body

end

macro test1()

 variable i

 variable t0,t1

 t0=ticks

 i=0

 silent 1

 do

 i+=1

 while(i<1e5)

 t1=ticks

 print/D "Time used is ",(t1-t0)/60,"s"

end

function test2()

 variable i

 variable t0,t1

 t0=ticks

 i=0

 silent 1

 do

 i+=1

 while(i<1e5)

 t1=ticks

 print/D "Time used is ",(t1-t0)/60,"s"

end

### 5.2.1

variable v=sin(x)

make/N=100 gsfun=gauss(x,50,1)+gnoise(0.1)

curvefit gauss,data/D

myfun()

variable sin

make cos

newpanel/N=sin //系统会自动将sin改为sin0

variable 1v, \_v

make ‘a wave’

### 5.2.2

Variable [/C/D/G] varName [=numExpr ][, varName [=numExpr ]]...

Variable v1 //声明一个变量v1

Variable v1=0 //声明一个变量v1，并赋值为0

Variable pi //错误，pi是系统函数，返回圆周率值

Variable sin //错误，sin是系统数学函数

Variable/C c=cmplx(1,1) //声明一个复数型变量，并赋值1+i

Variable/G v2 //声明一个全局变量v2

Variable \_v3 //错误，不能以下划线开头

String [/G] strName [=strExpr ] [, strName [=strExpr ]... ]

String str1=”” //声明一个字符串str1,赋空值

String str1=”hello,world” //声明一个字符串str1，并赋值”hello,world”

String/G str1 //声明一个全局型字符串str1

Constant kName = literalNumber

### 5.2.3

Structure structurename

 Memtype memname [arraysize] [,memtype,memname [arraysize]]

 …

Endstructure

Struct structurename name

Structure mystruct

 Variable var1

 Variable var2[10]

EndStructure

Function myfunc()

 Struct mystruct ms

 Ms.var1=1

 Ms.var2[n]=20

 …

end

Constant kCaSize= 5

Structure substruct

 Variable v1

 Variable v2

EndStructure

Structure mystruct

 Variable var1

 Variable var2[10]

 String s1

 WAVE fred

 NVAR globVar1

 SVAR globStr1

 FUNCREF myDefaultFunc afunc

 STRUCT substruct ss1[3]

 char ca[kCaSize+1]

EndStructure

Struct mystruct ms

Wave ms.fred

Nvar ms.globvar1

Svar ms.globstr1

Funcref mydefaultfunct anotherfunc

Ms.fred=fred

Wave w=fred

Ms.fred=fred

Function subfunc(s)

 Struct structname &s

 …

End

### 5.2.4

If (<expression>)

 <True Part>

Else

 <False Part>

Endif

If (<expression>)

 <True Part1>

Elseif

 <True Part2>

Else

 <False Part>

endif

If (<expression>)

 <True Part>

endif

Function example()

 Variable a=2,b=1

 If(a > b)

 Print a

 Else

 Print b

 Endif

end

switch(<numerical expression>)

 case <literal><number constant>:

 <code>

 [break]

 case <literal><constant>:

 <code>

 [break]

 [default:

 <code>]

 Endswitch

strswitch(<string expression>)

 case <literal><string constant>:

 <code>

 [break]

 case <literal><string constant>:

 <code>

 [break]

 [default:

 <code>]

 Endswitch

Switch(a>b)

 Case 1:

 Print a

 break

 Case 0:

 Print b

 Break

 Default:

 Break

endswitch

Do

 <loop body>

While(<experssion>)

For(<initialize>;<continue test>;<update>)

 <loop body>

endfor

function example()

 variable I=0,sum=0

 do

 sum+=i

 i+=1

 while(i<=100)

 print sum

end

 Function example()

 Variable I,sum=0

 For(i=0;i<=100;i+=1)

 Sum+=i

 Endfor

 Print sum

 end

Function example()

 Variable I=0,sum=0

 For(;;)

 Sum+=i

 I+=1

 If(i>100)

 Break

 Endif

 Endfor

end

Function example()

 Variable i=0,sum=0

 Do

 i+=1

 If(mod(I,2)==0)

 Continue

 Endif

 Sum+=i

 While(i<=100)

 Print sum

End

### 5.2.5

Function myFun(a,b)

 Variable a,b

 Variable c

 C=sqrt(a^2+b^2)

 Return c

End

Function/S gettracename()

 String s,s1

 S=tracenamelist(“”,”;”,1)

 S1=stringfromlist(0,s)

 Return s

End

Make data1

Display data1

Print gettracename()

### 5.2.6

function mymarqueefun():graphmarquee

 getmarquee

 print V\_left,V\_right,V\_top,V\_bottom

end

make/O data=x

display data

### 5.2.7

Function mainFunc()

 Variable v=1

 String s=”hello”

 Subroutine(v,s)

 Print v,s

End

Function subroutine(v,s)

 Variable v

 String s

 Print v,s

 V=2

 S=”hello,IGOR”

end

Function mainFunc()

 Variable v=1

 String s=”hello”

 Subroutine(v,2,s)

 Print s,v

End

Function subroutine(num1,num2,s)

 Variable &num1,num2

 String &s

 Num1=num1+num2

 S=”The sum of the two num is”

 Print s,num1

end

Function f1(v)

 Variable &v

 F2(v)

End

Function f2(v)

 Variable &v

End

Variable v1

Variable &v=v1

Function routine()

 Make/O wave0=x

 Subroutine(wave0)

End

Function subroutine(w)

 Wave w

 W=1234

End

Waveclear w

### 5.2.8

function f(p1,p2,[p3,vout,s1,w1])

 variable p1,p2

 variable p3

 variable &vout

 string s1

 wave w1

end

f(1,2)

f(1,2,p3=3)

f(1,2,vout=v) //v是一个数值变量

f(1,2,p3=3,s1=s1 //s1是一个字符串变量

f(1,2,w1=w) //w是一个wave引用

if(paramisdefault(p1))

 p1=1

endif

看下面的例子：

function maxvalue(a,b,[c])

 variable a,b,c

 if(paramisdefault(c))

 return max(a,b)

 else

 return max(max(a,b),c)

 endif

end

print maxvalue(1,2)

print maxvalue(1,2,c=3) //注意变量名c不能省略

### 5.2.9

//calculate the sum of a and b

c=a+b

//this function is used to fit data using …

function lorf(x,cw):Fitfunc

 variable x //x value where the to be calculated

 wave cw //coefficient wave

 //cw[0]: some meaning full description 1

 //cw[1]: some meaning full description 1

 …

 <function body>

end

If(a>b)

 Num\_max=a

Else

 Num\_max=b

Endif

### 5.3.1

#include <procedurefilename>

#include “procedurefilename”

#include <Image Common>

#include “proc0”

#include “E:tmp:myproc”

#include “:categary1:proc1”

#include “:proc1”

### 5.3.2

#Pragmas keyword [ = parameter ]

#pragma rtGlobals = value

#pragma version = versionNumber

#pragma IgorVersion = versionNumber

#pragma ModuleName = name

#pragma IndependentModule = name

#pragmas rtglobals=0

#pragmas rtglobals=1

#pragmas rtglobals=2

#pragmas rtglobals=3

Function test()

 Display wave1

End

Function test()

 Wave wave1

 Display wave1

End

Function test()

 Make wave1

 Display wave1

End

Function test()

 Display wave1

End

Function test()

 Wave wave1

 Wave1=sin(x)

 Display wave2 // rtglobals=1时正确，rtglobals=3时错误

End

make data1={1,2,3,4,5}

data1[6]=7

#pragma version = 1.01

#include “a procedure file” version>=1.03

#pragma igorversion=5.03

#pragma ModuleName= name

#pragmas ModuleName = myGreateProcedure

Static function foo(a)

 Variable a

 Return a+100

End

myGreateProcedure#foo(0)

### 5.3.3

#pragma IndependentModule = imName

#pragma IndependentModule=myIM

Function test()

 Print “this function executes under an Independent Module”

End

myIM#test()

#pragma IndependentModule=myIM

 Function test()

 f1()

end

static function f1()

 print “Independent Module”

 end

myIM#test() //可以执行

myIM#f1() //不能执行

SetIgorOption IndependentModuleDev=1

#pragma independentmodule=myIM

#pragma rtGlobals=3

#pragma IndependentModule=myIM

menu "macros"

 "my menu item", myIM#f1()

 "create window",myIM#f2()

end

function f1()

 print 1

end

function f2()

 execute "panel0()"

end

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(150,77,412,164)

 PopupMenu popup0,pos={21,17},size={100,20},bodyWidth=100

 PopupMenu popup0,mode=1,popvalue="Yes"

 PopupMenu popup0,value= #GetIndependentModuleName()+"#popuplist()"

EndMacro

function/S popuplist()

 return "items;item2"

end

myIM#popuplist()

### 5.3.4

Function test()

 Execute “f1()”

 //由于f1()是一个proc，在函数中无法直接调用

End

Proc f1()

 Print 1

 End

String cmd

sprintf cmd, "GBLoadWave/P=%s/S=%d \"%s\"", pathName, skipCount, fileName

Execute cmd

Execute/P “dowindow/R PanelName”

Execute/P "INSERTINCLUDE <Peak Functions>

### 5.3.5

#define symbol

#undef symbol

#define用以定义一个符号标记，#undef用去取消一个符号标记定义。

#ifdef symbol

 <code1>

#else

 <code2>

#endif

#if expression

 <code1>

#else

 <code2>

#endif

**格式3：**

#if expression1

 <code1>

#elif expresion2

 <code2>

#else

 <code3>

#endif

#ifndef symbol

 <code1>

#else

 <code2>

#endif

和#ifdef刚好相反。

#ifdef WINDOWS

Function f()

 Print “I am running on windows.”

End

#endif

#ifdef MACINTOSH

Function f()

 Print “I am running on Mac OS.”

End

#endif

I am running on windows.

5.3.6 Funcref myprotofunc f=functionname

Function f0()

 Print “Naive!”

End

Function f1()

 Print “You get it!”

End

Function test(fref)

 Funcref f0 fref

 Function f0 fref1=f1

 Function f0 fref2=$”somefunc”

 Function f0 fref3=fref

 Fref1()

 Fref2()

 Fref3()

end

Test(f1)

Test($”aa”)

Function f3(v)

 Variable v

 Print “I am not the right one”

End

Function test(fref)

 Funcref f0 fref

 Funcref f0 fref1=f1

 Funcref f0 fref2=$”somefunc”

 Funcref f0 fref3=fref

 Funcref f0 fref4=f3 //------------>add this line

 Fref1()

 Fref2()

 Fref3()

end

Funcref myprotofunc f=functionname

Funcref myprotofunc f

F=functionname

### 5.3.7

Function myFun()

 Print v

End

NVAR [/C][/Z] localName [= pathToVar ][, localName1 [= pathToVar1 ]]...

nvar v //正确，此时引用名就是v，能直接访问v

nvar num=v //正确，引用名为num，能直接访问v

nvar num=root:v //正确，引用名为num，指明v的完整路径，能直接访问v

nvar v2 //错误，由于当前文件是root，root下不存在v2这样的全局变量，不能访问v2

nvar num=root:Folder1:v2 //正确，引用名为num，且指明了绝对路径，可以访问v2

nvar num=:Folder1:v2 //正确，引用名为num，且指明了相对路径，可以访问v2

nvar v=v //正确，引用名和全局变量名相同，可以访问v

WAVE [/C][/T][/Z] localName [= pathToWave ][, localName1 [= pathToWave1 ]]...

Wave w=wave1 //引用名为w

Wave wave1 //引用名为wave1

Wave w=root:wave1 //引用名为w，指明绝对路径

Wave w=:wave1 //引用名为w，指明相对路径

make/O wave1

variable v

string str

proc myProc()

 wave1=sin(x)

 v=1

 str="Hello,Igor"

end

macro myMacro()

 wave1=sin(x)

 v=1

 str="Hello,Igor"

end

### 5.3.8

Make wave1

Duplicate wave1 wave2

Fft/dest=w wave1

Function CreateaWave(namestrforwave)

 String namestrforwave

 Make $namestrforwave

 Wave w=$namestrforwave

 W=x^2

End

CsrWaveRef(cursorName [, graphNameStr ])

Wave w=Csrwaveref(A)

Wave w=CsrWaveRef(A,”graph0”)

Wave w=csrwaveref(A,namestr)

Print Csrwaveref(A)[0]

WaveRefIndexed(windowNameStr, index, type )

Wave w=waverefindexed(“”,0,1)

TraceNameToWaveRef(graphNameStr, traceNameStr )

string list=tracenamelist(“”,”;”,1)

wave w=TraceNametoWaveRef(“”,stringfromlist(0,list))

Nameofwave(wave)

GetWavesDataFolder(wave,kind)

Print nameofwave(w)

Function myFunc()

 Make/O data1

 Setscale/I x,0,2\*pi,data1

 Display data1

 Wave w=waverefindexed(“”,0,1)

 Print nameofwave(w)

 Print getwavesdatafolder(w,2)

End

结果如图5-38所示：

If(!waveexists(w)) //wave w存在

 <do somethings>

 Else //wave w不存在

 <coping with errors>

Endif

### 5.3.9

Nvar num=v

Wave w=$”wave1”

Function displaywaves()

 String s,s1

 S=wavelist(“\*”,”;”,”DIMS:1”)

 Variable I,n

 N=itemsinlist(s)

 display

 For(i=0;i<n;i+=1)

 S1=stringfromlist(i,s)

 Appendtograph $s1

 Endfor

End

String s=”somewave”

$s=sin(x)

String s=”somewave”

Wave w=$s

W=sin(x)

Variable var1=1

String str1=”Global string”

Function myFun()

 String vref,sref

 Vref=”var1”

 Sref=”str1”

 Nvar v=$vref

 Svar sG=$sref

 Print v,sg

End

Function GraphWindowExsts(s)

 String s

 Dowindow $s

 If(V\_Flag==0)

 Return 0

 Else

 Return 1

 Endif

End

### 5.3.10

Display

Dowindow/C myGraph

Dowindow myGraph

Print V\_flag

Dowindow somegraph

Print V\_flag

Function GraphWindowExsts(s)

 String s

 Dowindow $s

 If(V\_Flag==0)

 Return 0

 Else

 Return 1

 Endif

End

Function test()

 Print V\_flag

 Dowindow aa

end

### 5.3.11

Function testf(v1)

 Variable v1

 make/O/N=(3,4) data

 variable i

 for(i=0;i<v1;i+=1)

 matrixop/O w=col(data,i)

 endfor

 Wave w=w1

 Display w1

 print v1

end

execute/P “dowindow/R winname”

execute “macroname()”

# 第6章

### 6.1.4

ControlOperation Name [, keyword[=value] [,keyword[=value]]…]

Button button0 //创建button0（假设button0不存在）

Button button0 size={100,20} //设置大小

Button button0 title="MyButton" //命名

Button button0 title="\\F'Arial'\\Z14MyButton" //设置名字字体及字号

Button button0 size={100,30} //修改大小

### 6.2.1

Function ButtonProc(ctrlName) : ButtonControl

 String ctrlName

End

Function ButtonProc(ba) : ButtonControl

 STRUCT WMButtonAction &ba

 switch( ba.eventCode )

 case 2: // mouse up

 // click code here

 break

 case -1: // control being killed

 break

 endswitch

 return 0

End

Button [/Z] ctrlName [ keyword = value [, keyword = value ...] ]

Button button0,pos+={10,20},rename=”stop”

### 6.2.2

Variable var\_flag

Function CheckProc(ctrlName,checked) : CheckBoxControl

 String ctrlName

 Variable checked

End

Function CheckProc(cba) : CheckBoxControl

 STRUCT WMCheckboxAction &cba

 switch( cba.eventCode )

 case 2: // mouse up

 Variable checked = cba.checked

 break

 case -1: // control being killed

 break

 endswitch

 return 0

End

Checkbox checkbox1,value=1

Checkbox checkbox1,variable=K1

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(150,50,353,212)

 Variable/G gRadioVal= 1

 CheckBox check0,pos={52,25},size={78,15},title="Radio 1"

 CheckBox check0, value= 1,mode=1,proc=MyCheckProc

 CheckBox check1,pos={52,45},size={78,15},title="Radio 2"

 CheckBox check1,value= 0,mode=1,proc=MyCheckProc

 CheckBox check2,pos={52,65},size={78,15},title="Radio 3"

 CheckBox check2,value= 0,mode=1,proc=MyCheckProc

EndMacro

Function MyCheckProc(name,value)

 String name

 Variable value

 NVAR gRadioVal= root:gRadioVal

 strswitch (name)

 case "check0":

 gRadioVal= 1

 break

 case "check1":

 gRadioVal= 2

 break

 case "check2":

 gRadioVal= 3

 break

 endswitch

 CheckBox check0,value= gRadioVal==1

 CheckBox check1,value= gRadioVal==2

 CheckBox check2,value= gRadioVal==3

End

### 6.2.3

Variable var

SetVariable setvar0 value=var

string str1

Setvariable setvar1,size={150,20}

Setvariable setvar1,value=str1

Make/O aa

Setvariable setvar0,value=aa[0]

Setvariable setvar1,value=K1

Setvariable setvar0,value=\_NUM:0 //Internal Number

Setvariable setvar1,value=\_STR:”” //Internal string

Function SetVarProc(ctrlName,varNum,varStr,varName) : SetVariableControl

 String ctrlName

 Variable varNum

 String varStr

 String varName

End

Function SetVarProc(sva) : SetVariableControl

 STRUCT WMSetVariableAction &sva

 switch( sva.eventCode )

 case 1: // mouse up

 case 2: // Enter key

 case 3: // Live update

 Variable dval = sva.dval

 String sval = sva.sval

 break

 case -1: // control being killed

 break

 endswitch

 return 0

End

### 6.2.4

NewPanel/N=myPanel

Listbox list0

Newpanel

Make/O/T/N=(5,1) textwave={{”aaa”,”bbb”,”ccc”,”ddd”,”eee”}}

Setdimlabel 1,0, $"\\f01My Label",textwave

ListBox list0,size={90,163},listWave=root:textwave

Newpanel

Make/O/T/N=(5,1) textwave={{“aaa”,”bbb”,”ccc”,”ddd”,”eee”}}

Make/T/O/N=1 tiltletextwave

Tiltletextwave[0]= "\\JCThis is the title\\K(65535,0,0)\\k(65535,0,0)\\W523"

ListBox list0,size={161,167},listWave=root:textwave,titlewave=tiltletextwave

Make/O/T/N=(3,3) listw0={{“col1”,”col2”,”col3”},{“col1”,”col2”,”col3”},{“col1”,”col2”,”col3”}}

Make/O/N=(3,3,3) selw0

Newpanel

Listbox list0,listwave=listw0,selwave=selw0,mode=8,size={151,73}

Make/O/N=(3,3)/U colw={{65535,0,0},{0,65535,0},{0,0,65535}}

Matrixtranspose colw

SetDimLabel 2, 1, foreColors, selw0

SetDimLabel 2, 2, backColors, selw0

Function MyListboxProc(ctrlName,row,col,event) : ListboxControl

 String ctrlName // 控件名

 Variable row //被选中项的行数

 Variable col // 被选中项的列数

 Variable event // 事件

 ...

 return 0

End

Function newActionProcName(LB\_Struct) : ListboxControl

 STRUCT WMListboxAction &LB\_Struct

 ...

End

### 6.2.5

Newpanel

Popupmenu pop1

Display

Popupmenu p1,value=”cmd1;cmd2”,title=”Execute Commands”,mode=0

Newpanel

Popupmenu p1,value=”option1;option2”,title=”Options”,mode=2

Newpanel

PopupMenu popup0, value="Red;Green;Blue;"

NewPanel

PopupMenu popup0, value=WaveList("\*", ";", "")

Function/s mylist()

 Return “aa;bb;cc”

End

Newpanel

PopupMenu popup0, value=mylist()

Function PopupDemo() // Local string variable specifying items

 NewPanel

 String quote = "\""

 String list

 if (CmpStr(IgorInfo(2),"Windows") == 0)

 list = quote + "Windows XP; Windows VISTA;" + quote

 else

 list = quote + "Mac OS X 10.4;Mac OS X 10.5;" + quote

 endif

 PopupMenu popup0, value=#list

End

Windows XP; Windows VISTA;

Value= Windows XP; Windows VISTA;

“Windows XP; Windows VISTA;”

Function/S WindowsItemList()

 String list

 list = "Windows XP; Windows VISTA;"

 return list

End

Function/S MacItemList()

 String list

 list = "Mac OS X 10.4;Mac OS X 10.5;"

 return list

End

Function PopupDemo() // Local string variable specifying function

 String listFunc

 if (CmpStr(IgorInfo(2),"Windows") == 0)

 listFunc = "WindowsItemList()"

 else

 listFunc = "MacItemList()"

 endif

 NewPanel

 PopupMenu popup0, value=#listFunc

End

String ss=”aa;bb;cc”

Newpanel

Popupmenu p1,value=:ss

String ss=”aa;bb;cc”

Newpanel

Popupmenu p1,value=#”:ss”

Function popupdemo()

 String s1

 S1=”root:fd1:fd2:list0”

 Newpanel

 Popupmenu p1,value=#s1

End

Newpanel

Popupmenu p1,value="\*COLORPOP\*"

Newpanel

Popupmenu p1,value=\*COLORPOP\*

Function PopupMenuAction (ctrlName,popNum,popStr) : PopupMenuControl

 String ctrlName

 Variable popNum // which item is currently selected (1-based)

 String popStr // contents of current popup item as string

 ...

End

Newpanel

Popupmenu colorpop,value=”\*COLORPOP\*”,bodywidth=100

Function PopMenuProc(ctrlName,popNum,popStr) : PopupMenuControl

 String ctrlName

 Variable popNum

 String popStr

 string s1

 s1=imagenamelist("",";")

 if(0==strlen(s1))

 doalert/T="Warnning",0,"There is no image to set"

 endif

 variable i,n

 string ss

 n=itemsinlist(s1)

 for(i=0;i<n;i+=1)

 ss=stringfromlist(0,s1)

 modifyimage $ss,ctab={\*,\*,$popstr,0}

 endfor

End

Make/O/N=(100,100) gaussfun

setscale/I x,-3,3,gaussfun

setscale/I y,-3,3,gaussfun

display gaussfun

newimage gaussfun

Function PopupMenuAction(PU\_Struct) : PopupMenuControl

 STRUCT WMPopupAction &PU\_Struct

 ...

End

### 6.2.6

Newpanel

Slider sld0,size={60,15}

make/O/N=4 ticknumw={0,1,2,3}

make/O/T/N=4 ticklblw={“fast”,”media”,”slow”,”off”}

newpanel

slider sld0,vert=0,size={200,50},userticks={ticknumw,ticklblw},limits={0,3,1}

 Function MySliderProc(name, value, event) : SliderControl

 String name

 Variable value

 Variable event

 return 0

 End

Function newActionProcName(S\_Struct) : SliderControl

 STRUCT WMSliderAction &S\_Struct

 ...

End

### 6.2.7

newpanel

valdisplay valdisp0,,value=\_NUM:0

ValDisplay vdisp0,pos={90,86},size={323,48},title="ValDisplay",fSize=15

ValDisplay vdisp0,limits={-50,50,0},barmisc={15,50},value= \_NUM:20

function valdisplaydemo()

 newpanel

 valdisplay led0,pos={10,20},size={35,35},limits={0,1,0},mode=1

 valdisplay led0,barmisc={0,0},value=\_NUM:0

 button button0,pos={60,25},size={80,20},title="Open",userdata="1"

 button button0,proc=togglingled

end

function togglingled(s0)

 string s0

 string ss

 ss=getuserdata("","button0","")

 variable state

 state=str2num(ss)

 valdisplay led0,value=\_NUM:state

 button button0,title=selectstring(state,"Open","Close")

 button button0,userdata=num2str(!state)

end

value=2

value=\_NUM:n

variable/G val

valdisplay valdisp0,value=val

value=val1\*val2+val3

value=myfunc()

value=#”val1\*val2+val3”

string ss=”myfunc()”

value=#ss

### 6.2.8

newpanel

TabControl tab1,pos={5,10},size={183,154},appearance={os9,Win}

TabControl tab1,tabLabel(0)="Tab 0",value= 0

TabControl [/Z] ctrlName [ keyword = value [, keyword = value ...] ]

Function TabProc(ctrlName,tabNum) : TabControl

 String ctrlName

 Variable tabNum

 return 0

End

Function newActionProcName(TC\_Struct) : TabControl

 STRUCT WMTabControlAction &TC\_Struct

 ...

End

Function TabProc(ctrlName,tabNum) : TabControl

 String ctrlName

 Variable tabNum

 variable istab0,istab1

 istab0=tabNum==0?0:1

 checkbox check0,disable=istab0

 checkbox check1,disable=istab0

 checkbox check2,disable=istab0

 checkbox check3,disable=istab0

 popupmenu popup0,disable=istab0

 return 0

End

Function TabProc(ctrlName,tabNum) : TabControl

 String ctrlName

 Variable tabNum

 variable istab0,istab1

 //tab 0

 istab0=tabNum==0?0:1

 checkbox check0,disable=istab0

 checkbox check1,disable=istab0

 checkbox check2,disable=istab0

 checkbox check3,disable=istab0

 popupmenu popup0,disable=istab0

 //tab 1

 istab1=tabNum==1?0:1

 setvariable setvar0,disable=istab1

 setvariable setvar1,disable=istab1

 setvariable setvar2,disable=istab1

 button button0,disable=istab1

 return 0

End

string s0=Controlnamelist(“”,”;”,”tab0\_”) //当前程序面板中的控件，以分号分隔

ModifyControlList s0,disable=!(tabNum==0)

### 6.2.9

newpanel

customcontrol c1

CustomControl [/Z] ctrlName [ keyword = value [, keyword = value ...] ]

function CustomControlFunc(s)

 struct WMCustomControlAction &s

 …

end

Constant kCCE\_mousedown = 1

#include <CustomControl Definitions>

newpanel

customcontrol c1,picture={ProcGlobal#Mines\_Tiles,18},proc=minestilesfun

static constant kCCE\_mouseup= 2

static constant kCCE\_frame= 12

Structure CC\_CounterInfo

 Int32 theCount // 当前显示的状态图片次序

EndStructure

Function minestilesfun (s)

 STRUCT WMCustomControlAction &s

 STRUCT CC\_CounterInfo info

 if( s.eventCode==kCCE\_frame )

 StructGet/S info,s.userdata

 s.curFrame= mod(info.theCount+(s.curFrame),18)

 elseif( s.eventCode==kCCE\_mouseup )

 StructGet/S info,s.userdata

 info.theCount= mod(info.theCount+1,18)

 StructPut/S info,s.userdata // will be written out to control

 endif

 return 0

End

### 6.2.10

newpanel

TitleBox tb1,frame= 3,labelBack=(55000,55000,65000) // 3D内陷显示风格，设置背景颜色

TitleBox tb1,title= "\Z18\[020 log\\B10\\M|[1 + 2K(jwt) + (jwt)[\\S2\\M]|\\S-1](file:///%5C%5CS2%5C%5CM%5D%7C%5C%5CS-1)"

### 6.2.11

ControlUpdate [/A/W=winName][controlName ]

variable v

function fff()

 nvar v

 return v

end

newpanel

valdisplay vdisp0,value=fff()

killcontrol [/w=winname] controlname

### 6.2.12

ControlInfo [/W=winName ] controlName

controlinfo checkbox1 //checkbox1是一个复选框

controlinfo setvar0 //setvar0是一个文本框控件且显示一个数值

variable var=V\_value

controlinfo list0 //list0是一个listbox控件

string s0=S\_Value //获取listwave名字

string s1=S\_Datafolder //获取listwave所处数据文件夹

string lw\_fpath=s1+s0 //获取listwave完整路径

wave w=$lw\_fpath[V\_Value] //V\_Value表示当前选择的项次序（从0算起）

### 6.2.13

Function structureTest() //创建程序面板，添加一个名为b0的按钮

 NewPanel

 Button b0,proc= NewButtonProc

End

Structure MyButtonInfo //声明一个结构体类型，用于记录鼠标的状态

 Int32 mousedown //为1表示鼠标按下

 Int32 isLeft //为1表示鼠标位于按钮的左半部分

EndStructure

Function NewButtonProc(s)

 STRUCT WMButtonAction &s

 STRUCT MyButtonInfo bi

 Variable biChanged= 0 //当鼠标状态改变时置1

 StructGet/S bi,s.userdata //从用户数据中读取鼠标状态，并存入bi

 switch(s.eventcode)

 case 1: //鼠标按下事件

 bi.mousedown= 1 //鼠标按下，置1

 bi.isLeft= s.mouseLoc.h<(s.ctrlRect.left+s.ctrlRect.right)/2

 //判断鼠标是否位于按钮左侧，mouseLoc是一个point结构体类 //型变量，h表示鼠标的水平方向坐标，单位像素

 biChanged= 1 //鼠标按下状态改变了，置1

 break

 case 2: //鼠标在按钮上释放事件

 case 3: //鼠标在按钮外释放事件

 bi.mousedown= 0 //鼠标释放，置0

 biChanged= 1 //鼠标状态由按下到释放，状态改变，置1

 break

 case 4: //鼠标在按钮上移动事件

 if( bi.mousedown ) //如果鼠标按下去

 if( bi.isLeft ) //如果鼠标位于按钮左半边打印“L”

 printf "L"

 else //如果鼠标位于右半边打印“R”

 printf "R"

 endif

 else //如果鼠标没有按下去则打印“\*”

 printf "\*"

 endif

 break

 case 5: //鼠标进入按钮事件，打印“Enter button”

 print "Enter button"

 break

 case 6: //鼠标离开按钮事件，打印“Leave button”

 print "Leave button"

 break

 endswitch

 if( biChanged )

 StructPut/S bi,s.userdata //鼠标状态改变了，将改变写入控件用户数据

 //下次按钮某一事件发生时会使用此数据

 endif

 return 0

End

### 6.3.1

// PNG: width= 56, height= 44

Picture myPictName

 ASCII85Begin

 M,6r;%14!\!!!!.8Ou6I!!!!Y!!!!M#Qau+!5G;q\_uKc;&TgHDFAm\*iFE\_/6AH5;7DfQssEc39jTBQ

 =U!7FG,5u`\*!m?g0PK.mR"U!k63rtBW)]$T)Q\*!=Sa1TCDV\*V+l:Lh^NW!fu1>;(.<VU1bs4L8&@Q\_

 <4e(%"^F50:Jg6);j!CQdUA[dh6]%[OkHSC,ht+Q7ZO#.6U,IgfSZ!R1g':oO\_iLF.GQ@RF[/\*G98D

 bjE.g?NCte(pX-($m^\\_FhhfL`D9uO6Qi5c[r4849Fc7+\*)\*O[tY(6<rkm^)/KLIc]VdDEbF-n5&Am

 2^hbTu:U#8ies\_W<LGkp\_LEU1bs4L8&?fqRJ[h#sVSSz8OZBBY!QNJ

 ASCII85End

End

newpanel

drawpict 0,0,1,1,ProcGlobal#myPictName

ProcGlobal#gProcPictName

modName#ProcPictName

DrawPict 0,0,1,1, ProcGlobal#gProcPictName //公有图片程序代码

### 6.3.3

basename#sub1

#sub1

#

##

button btn0, win=mainwindow#mysubpanel,title=”button0”

appendtograph/w=mainwindow#mysubgraph data

modifypanel fixedsize=1 //Panel

modifygraph width=xx,height=xx //graph

newpanel/K=1 //panel

display/K=1 //graph

newpanel/flg

### 6.3.4

NewPanel /W=(315,81,615,281)

display /W=(315,81,615,281)

### 6.4.1

 Menu "Macros"

 "Load Data File/1"

 "Do Analysis/2"

 "Print Report"

End

Menu "Macros"

 "Load Data File/1", Beep; LoadWave/G

 "Do Analysis/2"

 "Print Report"

End

Menu "Macros"

 Submenu "Load Data File"

 "Text File"

 "Binary File"

 End

 Submenu "Do Analysis"

 "Method A"

 "Method B"

 End

 "Print Report"

End

menu “myproc”

 “open dataprocessing dialog”,dataprocessdg()

end

proc dataprocessdg()

 //do something to prepare for the dataprocessdg

 dataprocessdg\_panel() //Recreatoin Macro

end

Menu <Menu title string> [,<menu options>]

 [<Menu help strings>]

 <Menu item string> [,<menu item flags>] [,<execution text>]

 [<Item help strings>]

 …

 Submenu <Submenu title string>

 [<Submenu help strings>]

 <Submenu item string> [,<execution text>]

 [<Item help strings>]

 …

End

Menu "Test"

 help = {"This is the help for the Test menu."}

 "Load Data File"

 help = {"This is the help for the Load Data File item."}

 end

6.4.2 Function DoAnalysis()

 Print "Analysis Done"

End

Function ToggleTurboMode()

 Variable prevMode = NumVarOrDefault("root:gTurboMode", 0)

 Variable/G root:gTurboMode = !prevMode

End

Function/S MacrosMenuItem( itemNumber )

 Variable itemNumber

 Variable turbo = NumVarOrDefault("root:gTurboMode", 0)

 if (itemNumber == 1)

 if (strlen(WaveList("\*", ";", ""))==0) // any waves exist?

 return "(Do Analysis" // disabled state

 else

 return "Do Analysis" // enabled state

 endif

 endif

 if (itemNumber == 2)

 if (turbo)

 return "!"+num2char(18)+"Turbo" // Turbo with a check

 else

 return "Turbo"

 endif

 endif

End

Menu "Macros", dynamic

 MacrosMenuItem(1)

 help= {"Do analysis", "Not available because there are no waves."}

 MacrosMenuItem(2), /Q, ToggleTurboMode()

 help= {"When checked, turbo mode is on."}

End

Menu "Waves", dynamic

 WaveName("",0,4), DoSomething($WaveName("",0,4))

 WaveName("",1,4), DoSomething($WaveName("",1,4))

 WaveName("",2,4), DoSomething($WaveName("",2,4))

 WaveName("",3,4), DoSomething($WaveName("",3,4))

 WaveName("",4,4), DoSomething($WaveName("",4,4))

 WaveName("",5,4), DoSomething($WaveName("",5,4))

 WaveName("",6,4), DoSomething($WaveName("",6,4))

 WaveName("",7,4), DoSomething($WaveName("",7,4))

End

Function DoSomething(w)

 Wave/Z w

 if( WaveExists(w) )

 Print "DoSomething: wave's name is "+NameOfWave(w)

 endif

End

### 6.4.3

menu “tracepopup”

 “Print Trace Name”,printtracename()

end

function printtracename()

 getlastusermenuinfo

 print S\_tracename

end

### 6.4.4

 menu “character”

 "\*CHARACTER\*(Symbol,12) "

end

menu "specialmenus"

 submenu "Character"

 "\*CHARACTER\*(Arial,12)",/Q,getmenuinfo()

 end

 submenu "Colortable"

 "\*COLORTABLEPOP\*(YellowHot,1)",/Q,getmenuinfo()

 end

 submenu "ColorPop"

 "\*COLORPOP\*(0,65535,0)",/Q,getmenuinfo()

 end

 submenu "LineStyle"

 "\*LINESTYLEPOP\*(3)",/Q,getmenuinfo()

 end

 submenu "Markers"

 "\*MARKERPOP\*(8)",/Q,getmenuinfo()

 end

 submenu "Patterns"

 "\*PATTERNPOP\*(1)",/Q,getmenuinfo()

 end

end

function getmenuinfo()

 getlastusermenuinfo

 string str

 variable n=V\_flag

 dowindow/F char\_dlg

 if(!V\_Flag)

 display

 dowindow/C char\_dlg

 endif

 switch(n)

 case 3://"\*FONT\*"

 break

 case 6://"\*LINESTYLEPOP\*"

 popupmenu pop1,pos={1,80},value= "\*LINESTYLEPOP\*" popupmenu pop1,mode=V\_value+1

 break

 case 7://"\*PATTERNPOP\*"

 popupmenu pop2,pos={1,160},value= "\*PATTERNPOP\*",mode=V\_value

 break

 case 8://"\*MARKERPOP\*"

 popupmenu pop3,pos={1,120},value= "\*MARKERPOP\*"

 popupmenu pop3,mode=V\_value+1

 break

 case 9://"\*CHARACTER\*"

 str="\\F'Arial'\Z30"+S\_Value

 TextBox/C/N=text0/A=MC str

 break

 case 10://"\*COLORPOP\*"

 popupmenu pop4,pos={1,40},value= "\*COLORPOP\*"

 popupmenu pop4,popColor=(V\_Red,V\_green,V\_blue )

 break

 case 13://"\*COLORTABLEPOP\*"

 popupmenu pop5,pos={1,0},value= "\*COLORTABLEPOP\*"

 popupmenu pop5,mode=V\_value

 break

 endswitch

end

### 6.4.5

PopupContextualMenu [/C=(xpix, ypix) /N ] popupStr

function test()

 string poplist=”menu items 1;menu items 2”

 popupcontextualmenu poplist

end

menu “contextualmenu”,contextualmenu

 “Option1”,ff()

 “Option2”,ff()

end

function test()

 popupcontextualmenu/N “contextualmenu”

end

function ff()

 getlastusermenuinfo

 string str

 str=”the “+num2str(V\_Value+1)+” indexed item “+S\_Value+” is selected”

 print str

end

function demon()

 newpanel

 setwindow kwTopWin,hook(myhook)=myhook

end

function myhook(s)

 STRUCT WMWinHookStruct &s

 switch(s.eventcode)

 case 3:

 if(s.eventMod&~0x01)

 test()

 endif

 break

 endswitch

end

### 6.4.6

menu “test”

 “menu1/1”

 “!menu2”

 “m&enu3”

 “(menu4”

 “-“

 “menu5/O1”

 “menu6/S1”

 “menu7/OS1”

 “menu8/F5”

 “menu9/OF5”

 “menu10/CF5”

 “menu11/OCF5”

 “menu12/OSCF5”

end

newpanel

popupmenu p1,value=”menu1/1;!menu2”

menu “test”

 “km/h”

 “\\M0km/h”

end

newpanel

popupmenu p1,value="\\M0(First item;\\M1(Second item;\\M1!"+num2char(18)+"Third item"

# 第7章

### 7.1.1

wave w= NewFreeWave(2 , 4) //Free Wave，长度为4，数据类型float

make/Free/N=4 w //Free wave，长度为4

duplicate/Free awave,w

Function TestFreeWaveDeletion1()

 Wave w = NewFreeWave(2,3) // 创建free wave

 WaveClear w // w被清除

 //由于w被清除，free wave没有wave引用指向它，因此自动被删除

 Wave w = NewFreeWave(2,3) //创建free wave

 Wave w = root:wave0 // w被指向另外的wave

 // 没有wave引用指向此free wave，因此自动被删除

 Wave w = NewFreeWave(2,3) //创建free wave

 End // 函数结束，w的生存期结束，free wave也自动被删除

Function TestFreeWaveDeletion2()

 Make /D /N=3 /FREE jack // 创建一个free wave，wave引用自动被创建为jack

 Make /D /N=5 /FREE jack // 创建一个新的free wave，引用名仍然为jack

 // 由于jack被指向了新创建的具有5个元素的free wave，因此原来的

 //具有3个元素的free wave没有引用指向它，被自动删除

End // 函数结束，jack生存期结束，它指向的free wave也被删除（5个元素的wave）

function test()

 print nameofwave(newfreewave(2,4))

end

Function/WAVE Subroutine1()

 Make /D /N=3 /FREE jack=p // 创建Free wave

 return jack //返回free wave的引用

End

Function MainRoutine1()

 WAVE w= Subroutine1() // w指向由Subroutine1()返回的free wave

 Print w

End

wave w=newfreewave(2,4)

movewave w,root:mywave

Make/wave/N=1 wfree

Wfree[0]=newfreewave(2,10)

7.1.2 DFREF dfr=newfreedatafolder()

Function Test()

 DFREF dfrSave = GetDataFolderDFR()

 SetDataFolder NewFreeDataFolder() // 创建free data folder

 …

 SetDataFolder dfrSave

End

Function Test1()

 DFREF dfr= NewFreeDataFolder()

 KillDataFolder dfr // 引用dfr被删除已经不存在，free data folder也被删除

 DFREF dfr= NewFreeDataFolder()

 DFREF dfr= root:

 //引用dfr指向了其他数据文件夹，free data folder无引用指向它而被删除

 DFREF dfr= NewFreeDataFolder()

End //函数结束，dfr生存期结束，free data folder被删除。

function test1()

 setdatafolder newfreedatafolder()

 make jack //jack并不是free wave

 setdatafolder root: //free data folder 被删除，jack变为free wave

end

Make/df/N=1 wdf

Wdf[0]=newfreedatafolder()

Function test()

 Wave/df wdf

 Dfref df=wdf[0]

 <…>

end

### 7.2.1

Make wave1

Variable a=4

MultiThread wave1= sin(x/a)

wave1= wave1[p+1] - wave1[p-1]

Function TestMultiThread(n)

 Variable n // Number of wave points

 Make/O/N=(n) testWave

 testWave= 0

 Variable t1,t2

 Variable timerRefNum

 //单线程

 timerRefNum = StartMSTimer

 testWave= sin(x/8)

 t1= StopMSTimer(timerRefNum)

 // 多线程

 timerRefNum = StartMSTimer

 MultiThread testWave= sin(x/8)

 t2= StopMSTimer(timerRefNum)

 Variable processors = ThreadProcessorCount

 Print "On a machine with",processors,"cores,MultiThread is", t1/t2,"faster"

End

//worker thread

### 7.2.2

ThreadSafe Function/WAVE Worker(w3DIn, plane)

 WAVE w3DIn

 Variable plane

 DFREF dfSav= GetDataFolderDFR()

 SetDataFolder NewFreeDataFolder()

 ImageTransform/P=(plane) getPlane, w3DIn

 WAVE wOut= M\_ImagePlane

 MatrixFilter/N=21 gauss,wOut

 SetDataFolder dfSav

 return wOut // Return a reference to the free M\_ImagePlane wave

End

//Main Thread

Function Test()

 Variable numPlanes = 50

 Make/O/N=(200,200,numPlanes) srcData= (p==(2\*r))\*(q==(2\*r))

 Make/WAVE/N=(numPlanes) ww

 Variable timerRefNum = StartMSTimer

 MultiThread ww= Worker(srcData,p)

 Variable elapsedTime = StopMSTimer(timerRefNum) / 1E6

 Print "Assignment statement took ", elapsedTime, " seconds"

 WAVE w= ww[0]

 Duplicate/O w, out3D

 Variable i

 for(i=1;i<numPlanes;i+=1)

 WAVE w= ww[i]

 Concatenate {w}, out3D

 endfor

 killwaves ww

end

DFREF dfSav= GetDataFolderDFR()

SetDataFolder NewFreeDataFolder()

ImageTransform/P=(plane) getPlane, w3DIn

WAVE wOut= M\_ImagePlane

MatrixFilter/N=21 gauss,wOut

SetDataFolder dfSav

return wOut

//worker thread

ThreadSafe Function/DF Worker(w3DIn, plane)

 WAVE w3DIn

 Variable plane

 DFREF dfSav= GetDataFolderDFR()

 DFREF dfFree= NewFreeDataFolder()
 SetDataFolder dfFree

 ImageTransform/P=(plane) getPlane, w3DIn

 WAVE wOut= M\_ImagePlane

 MatrixFilter/N=21 gauss,wOut

 SetDataFolder dfSav

 return dfFree

End

//Main Thread

Function Test()

 Variable numPlanes = 50

 Make/O/N=(200,200,numPlanes) srcData= (p==(2\*r))\*(q==(2\*r))

 Make/DF/N=(numPlanes) dfw

 Variable timerRefNum = StartMSTimer

 MultiThread ww= Worker(srcData,p)

 Variable elapsedTime = StopMSTimer(timerRefNum) / 1E6

 Print "Assignment statement took ", elapsedTime, " seconds"

 DFREF df= dfw[0]

 Duplicate/O df:M\_ImagePlane, out3D

 Variable i

 for(i=1; i<numPlanes; i+=1)

 df= dfw[i]

 Concatenate {df:M\_ImagePlane}, out3D

 endfor

 KillWaves dfw

end

### 7.2.3

variable tgid= ThreadGroupCreate(4)

ThreadStart tgID, index, WorkerFunc (param1, param2,...)

threadstart tgID,1,WorkerFunc() //WorkerFunc没有参数

ThreadReturnValue(tgID, index )

ThreadGroupWait(tgID, waitms )

variable nt= ThreadProcessorCount

variable err=ThreadGroupRelease(tgID)

ThreadGroupPutDF tgID, datafolder

dfref dref=ThreadGroupGetDFR(tgID, waitms )

ThreadSafe Function MyWorkerFunc(w,col)

 WAVE w

 Variable col

 w[][col]= sin(x/(col+1))

 return stopMSTimer(-2)

End

Function MTFillWave(dest) //多线程

 WAVE dest

 Variable ncol= DimSize(dest,1)

 Variable i,col,nthreads= ThreadProcessorCount //处理器个数

 variable mt= ThreadGroupCreate(nthreads) //启动线程组

 for(col= 0;col<ncol;)

 for(i=0;i<nthreads;i+=1)

 ThreadStart mt,i,MyWorkerFunc(dest,col) //启动线程进行计算

 col+=1

 if( col>=ncol )

 break

 endif

 endfor

 do

 variable tgs= ThreadGroupWait(mt,100)//每个线程每次最多等待0.1s

 while( tgs != 0 ) //返回0表示没有线程在运行，即线程全部结束

 endfor

 variable dummy= ThreadGroupRelease(mt)

End

Function STFillWave(dest) //单线程

 WAVE dest

 Variable ncol= DimSize(dest,1)

 Variable col

 for(col= 0;col<ncol;col+=1)

 MyWorkerFunc(dest,col)

 endfor

End

Function ThreadTest(rows)

 Variable rows

 Variable cols=10

 make/o/n=(rows,cols) jack

 Variable i

 for(i=0;i<10;i+=1) // get any pending pause events out of the way

 endfor

 Variable ttime= stopMSTimer(-2)

 Variable t0= stopMSTimer(-2)

 MTFillWave(jack)

 Variable t1= stopMSTimer(-2)

 STFillWave(jack)

 Variable t2= stopMSTimer(-2)

 ttime= (stopMSTimer(-2) - ttime)\*1e-6

 // Times are in microseconds

 printf "ST: %d, MT: %d; ",t2-t1,t1-t0

 printf "speed up factor: %.3g; tot time= %.3gs\r",(t2-t1)/(t1-t0),ttime

End

Function MTFillWave (dest)

 WAVE dest

 Variable ncol= DimSize(dest,1)

 Variable col,nthreads= ThreadProcessorCount

 Variable threadGroupID= ThreadGroupCreate(nthreads)

 for(col=0; col<ncol; col+=1)

 Variable threadIndex = ThreadGroupWait(threadGroupID,-2) - 1

 if (threadIndex < 0)

 ThreadGroupWait(mt, 50)

 col -= 1

 continue

 endif

 ThreadStart threadGroupID, threadIndex, MyWorkerFunc(dest,col)

 endfor

 do

 Variable threadGroupStatus = ThreadGroupWait(threadGroupID,100)

 while(threadGroupStatus != 0)

 Variable dummy= ThreadGroupRelease(threadGroupID)

End

ThreadSafe Function MyWorkerFunc()

 do

 do

 DFREF dfr=ThreadGroupGetDFR(0,1000)//dfr是一个free data folder引用

 if (DataFolderRefStatus(dfr) == 0)

 if( GetRTError(2) ) // 线程被关闭

 Print "worker closing down due to group release"

 else

 Print "worker thread still waiting for input queue"

 endif

 else

 break

 endif

 while(1)

 SVAR todo = dfr:todo

 WAVE jack = dfr:jack

 NewDataFolder/S outDF

 Duplicate jack,outw // 新复制的outw，其引用名也是outw

 String/G did= todo

 if( CmpStr(todo,"sin") )

 outw= sin(outw)

 else

 outw= cos(outw)

 endif

 // 清除引用名outw，否则由于wave outw仍然有引用（就是outw）指向它，

 // ThreadGroupPutDF无法正常执行

 WAVEClear outw

 ThreadGroupPutDF 0,: // 将当前数据文件夹放入队列

 KillDataFolder dfr // 删除已经无用的源数据文件夹

 while(1)

 return 0

End

Function DemoThreadQueue()

 Variable i,ntries= 5,nthreads= 2

 Variable/G threadGroupID = ThreadGroupCreate(nthreads)

 for(i=0;i<nthreads;i+=1)

 ThreadStart threadGroupID,i,MyWorkerFunc()

 endfor

 for(i=0;i<ntries;i+=1)

 NewDataFolder/S forThread

 String/G todo

 if( mod(i,3) == 0 )

 todo= "sin"

 else

 todo= "cos"

 endif

 Make/N= 5 jack= x + gnoise(0.1)

 WAVEClear jack

 ThreadGroupPutDF threadGroupID,: // 将当前数据文件夹送入队列

 endfor

 for(i=0;i<ntries;i+=1)

 do

 // 获取处理结果，dfr是一个free data folder引用

 DFREF dfr= ThreadGroupGetDFR(threadGroupID,1000)

 if ( DatafolderRefStatus(dfr) == 0 )

 Print "Main still waiting for worker thread results."

 else

 break

 endif

 while(1)

 SVAR did = dfr:did

 WAVE outw = dfr:outw

 Print "task= ",did,"results= ",outw

 // 下面两行并非必要，引用被赋予新值时，原来指向的free 内容自动删除。

 WAVEClear outw

 KillDataFolder dfr

 endfor

 Variable tstatus= ThreadGroupRelease(threadGroupID)

 if( tstatus == -2 )

 Print "Thread would not quit normally, had to force kill it. Restart Igor."

 endif

End

print countobjects(“”,1)

print countobjects(“”,1)

### 7.2.4

CtrlNamedBackground taskName , keyword = value [, keyword = value …]

Function TestTask(s) // 后台任务要周期性执行的函数

 STRUCT WMBackgroundStruct &s

 Printf "Task %s called, ticks=%d\r", s.name, s.curRunTicks

 return 0 // 返回0表示后台任务持续运行，返回1立即结束

End

Function StartTestTask()

 Variable numTicks = 2 \* 60 //每120tics执行一次（大约2s）

 CtrlNamedBackground Test, period=numTicks, proc=TestTask

 //后台任务名为test，运行周期为numticks，程序为testtask

 CtrlNamedBackground Test, start //启动后台任务

End

Function StopTestTask()

 CtrlNamedBackground Test, stop //结束后台任务

End

### 7.2.5

threadsafe function workerfunc(w)

 wave w

 w=sin(x)\*cos(y)\*sin(z)

 return 0

end

Function ButtonProc(ctrlName) : ButtonControl

 String ctrlName

 if(numvarordefault("root:tgID",0)==0)

 variable/G root:tgID

 endif

 nvar tgID=root:tgID

 tgID=threadgroupcreate(1)

 button button1,disable=0

 button button2,disable=0

 button button0,disable=2

End

//按钮回调函数，结束后台线程

Function ButtonProc\_1(ctrlName) : ButtonControl

 String ctrlName

 nvar tgID=root:tgID

 if(!threadgrouprelease(tgID))

 print "Thread has been terminated"

 endif

 button button1,disable=2

 button button2,disable=2

 button button0,disable=0

End

//按钮回调函数，创建一个巨大的三维数据，并启动线程完成计算

Function ButtonProc\_2(ctrlName) : ButtonControl

 String ctrlName

 dfref dfr=root:package

 if(datafolderrefstatus(dfr)==0)

 newdatafolder/S root:package

 else

 setdatafolder dfr

 endif

 wave w=data

 nvar tgID=root:tgID

 if(!waveexists(w))

 make/N=(k0,k1,k2) data

 else

 redimension/N=(k0,k1,k2) w

 endif

 threadstart tgID,0,workerfunc(w)

 button button2,disable=2

 valdisplay valdisp0,value=\_Num:0

 ctrlnamedbackground task1,proc=checkcal,period=12

 ctrlnamedbackground task1,start

End

//程序面板，用于启动线程，给后台线程复杂计算任务，关闭线程

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(140,142,440,342)

 Button button0,pos={9,10},size={78,20},proc=ButtonProc,title="Start Thread"

 Button button1,pos={122,9},size={78,20},disable=2,proc=ButtonProc\_1,title="Stop Thread"

 SetVariable setvar0,pos={3,70},size={94,16},value= K0

 SetVariable setvar1,pos={3,128},size={94,16},value= K2

 SetVariable setvar2,pos={3,98},size={94,16},value= K1

 Button button2,pos={148,92},size={86,26},disable=2,proc=ButtonProc\_2,title="Calculate"

 ValDisplay valdisp0,pos={152,129},size={50,13},limits={0,0,0},barmisc={0,1000}

 ValDisplay valdisp0,value= #"0"

EndMacro

//ctrlnamedbackground后台任务，用于检查后台线程计算任务是否完成

//注意使用threadgroupwait(tgID,0)检查线程是否完成，threadgroupwait函数会立

//即返回，checkcal耗时几乎为0，对Igor前端没有任何影响，请读者仔细体会。

function checkcal(s)

 struct wmbackgroundstruct &s

 nvar tgID=root:tgID

 if(threadgroupwait(tgID,0)!=0)

 controlinfo valdisp0

 valdisplay valdisp0,value=\_Num:V\_Value+0.2

 return 0 //线程仍然在运行，返回0，后台任务继续执行

 endif

 button button2,disable=0

 return 1 //线程已经结束，返回1，后台任务被结束。

end

### 7.2.6

threadsafe function timer\_thd(i0)

 wave i0

 do

 sleep/T 20 //the timer's accuracy is about 1/3 s

 i0=i0+1

 while(1)

end

function workfun1()

 //do nothing,just print information

 print "time arrived, 1 s elapsed."

end

function f1(i0)

 wave i0

 if(mod(i0,3)==0) //触发“到时消息”

 workfun1()

 endif

end

function starttimer()

 variable/G v1,tgid

 make/O/N=1 i0=0

 execute "v1:=f1(i0)"

 tgid=threadgroupcreate(1)

 threadstart tgid,0,timer\_thd(i0)

end

function stoptimer()

 nvar tgid

 if(!nvar\_exists(tgid))

 print "no timer present"

 endif

 if(0==threadgrouprelease(tgid))

 print "timer "+num2str(tgid)+" is stopped"

 else

 print "Something wrong,please restart igor"

 endif

 nvar v1

 killvariables/Z v1,tgid

 wave i0

 killwaves/Z i0

end

### 7.3.1

function test()

 variable v1=1

 string s1="hello"

 prompt v1,"Input a number"

 prompt s1,"Input a string"

 doprompt "Input parameters",v1,s1

 print v1,s1

end

Prompt variableName, titleStr

DoPrompt dialogTitleStr, variable

prompt variableName,titlestr,popup, menuListStr

function test()

 variable v1=1

 string s1="hello"

 prompt v1,"select an option",popup,"option1;option2;option3"

 prompt s1,"select an option",popup,"option1;option2;option3"

 doprompt "Input parameters",v1,s1

 print "you have selected "+num2str(v1)+"th item"

 doprompt "Input parameters",v1,s1

 print "you have selected "+s1

end

### 7.3.2

PauseForUser [/C] mainWindowName [, targetWindowName ]

Function UserCursorAdjust(graphName,autoAbortSecs)

 String graphName

 Variable autoAbortSecs

 DoWindow/F $graphName

 if (V\_Flag == 0)

 Abort "UserCursorAdjust: No such graph."

 return -1

 endif

 NewPanel /K=2 /W=(187,368,437,531) as "Pause for Cursor"

 DoWindow/C tmp\_PauseforCursor

 AutoPositionWindow/E/M=1/R=$graphName

 DrawText 21,20,"Adjust the cursors and then"

 DrawText 21,40,"Click Continue."

 Button button0,pos={80,58},size={92,20},title="Continue"

 Button button0,proc=UserCursorAdjust\_ContButtonProc

 Variable didAbort= 0

 if( autoAbortSecs == 0 )

 PauseForUser tmp\_PauseforCursor,$graphName //暂停程序，等待用户输入

 else

 SetDrawEnv textyjust= 1

 DrawText 162,103,"sec"

 SetVariable sv0,pos={48,97},size={107,15},title="Aborting in "

 SetVariable sv0,limits={-inf,inf,0},value= \_NUM:10

 Variable td= 10,newTd

 Variable t0= ticks

 Do

 newTd= autoAbortSecs - round((ticks-t0)/60)

 if( td != newTd ) //请读者思考为什么要这个判断

 td= newTd

 SetVariable sv0,value= \_NUM:newTd,win=tmp\_PauseforCursor

 if( td <= 10 )

 SetVariable sv0,valueColor= (65535,0,0),win=tmp\_PauseforCursor

 endif

 endif

 if( td <= 0 )

 DoWindow/K tmp\_PauseforCursor

 didAbort= 1

 break

 endif

 PauseForUser/C tmp\_PauseforCursor,$graphName

 while(V\_flag)

 endif

 return didAbort

End

Function UserCursorAdjust\_ContButtonProc(ctrlName) : ButtonControl

 String ctrlName

 DoWindow/K tmp\_PauseforCursor

End

Function Demo(autoAbortSecs)

 Variable autoAbortSecs

 Make/O jack;SetScale x,-5,5,jack

 jack= exp(-x^2)+gnoise(0.1)

 DoWindow Graph0

 if( V\_Flag==0 )

 Display jack

 ShowInfo

 endif

 if (UserCursorAdjust("Graph0",autoAbortSecs) != 0)

 return -1

 endif

 if (strlen(CsrWave(A))>0 && strlen(CsrWave(B))>0)

 CurveFit gauss,jack[pcsr(A),pcsr(B)] /D

 endif

End

Function UserGetInputPanel\_ContButton(ctrlName) : ButtonControl

 String ctrlName

 DoWindow/K tmp\_GetInputPanel // 关闭输入窗口

End

Function DoMyInputPanel()

 NewPanel /W=(150,50,358,239)

 DoWindow/C tmp\_GetInputPanel

 DrawText 33,23,"Enter some data"

 SetVariable setvar0,pos={27,49},size={126,17},limits={-Inf,Inf,1}

 SetVariable setvar0,value= root:tmp\_PauseForUserDemo:numvar

 SetVariable setvar1,pos={24,77},size={131,17},limits={-Inf,Inf,1}

 SetVariable setvar1,value= root:tmp\_PauseForUserDemo:strvar

 Button button0,pos={52,120},size={92,20}

 Button button0,proc=UserGetInputPanel\_ContButton,title="Continue"

 PauseForUser tmp\_GetInputPanel //创建输入窗口（没有目标窗口）

End

Function Demo1()

 NewDataFolder/O root:tmp\_PauseForUserDemo

 Variable/G root:tmp\_PauseForUserDemo:numvar= 12

 String/G root:tmp\_PauseForUserDemo:strvar= "hello"

 DoMyInputPanel()

 NVAR numvar= root:tmp\_PauseForUserDemo:numvar

 SVAR strvar= root:tmp\_PauseForUserDemo:strvar

 printf "You entered %g and %s\r",numvar,strvar

 KillDataFolder root:tmp\_PauseForUserDemo

End

### 7.3.3

Function simpletest(indefinite, useIgorDraw)

 Variable indefinite

 Variable useIgorDraw //设置为1使用igor默认风格，否则使用操作系统风格

 NewPanel /N=ProgressPanel /W=(285,111,739,193)

 ValDisplay valdisp0,pos={18,32},size={342,18}

 ValDisplay valdisp0,limits={0,100,0},barmisc={0,0}

 ValDisplay valdisp0,value= \_NUM:0

 if( indefinite )

 ValDisplay valdisp0,mode= 4 //一致忙碌状态

 else

 ValDisplay valdisp0,mode= 3 // 按比例显示进度

 endif

 if( useIgorDraw )

 ValDisplay valdisp0,highColor=(0,65535,0) //操作系统风格进度条

 endif

 Button bStop,pos={375,32},size={50,20},title="Stop"

 DoUpdate /W=ProgressPanel /E=1 // 将此窗口设置为程序进度条窗口

 Variable i,imax= indefinite ? 10000 : 100

 for(i=0;i<imax;i+=1)

 Variable t0= ticks

 do

 while( ticks < (t0+3) )

 if( indefinite )

 ValDisplay valdisp0,value= \_NUM:1,win=ProgressPanel

 //每次“相位”改变1，进度条不断变化，显示忙碌状态

 else

 ValDisplay valdisp0,value= \_NUM:i+1,win=ProgressPanel

 //进度条按完成比例变化，显示进度

 endif

 DoUpdate /W=ProgressPanel

 if( V\_Flag == 2 ) //用户希望终止程序

 break

 endif

 endfor

 KillWindow ProgressPanel

End

Function spinnertest(nloops)

 Variable nloops

 //---开始创建进度条窗口

 NewPanel/FLT /N=myProgress/W=(285,111,739,193)

 //FLT表示创建一个浮动最前端显示程序面板

 ValDisplay valdisp0,pos={18,32},size={342,18}

 ValDisplay valdisp0,limits={0,100,0},barmisc={0,0}

 ValDisplay valdisp0,value= \_NUM:0

 ValDisplay valdisp0,mode= 4 // 一直忙碌模式

 Button bStop,pos={375,32},size={50,20},title="Abort"

 SetActiveSubwindow \_endfloat\_ //取消程序面板为活动窗口

 //---进度条窗口创建结束

 DoUpdate/W=myProgress/E=1 // 设置刚创建的窗口为程序进度条窗口

 SetWindow myProgress,hook(spinner)= MySpinner

 //给程序进度条窗口指定一个钩子函数MySpinner

 Variable t0= ticks,i

 for(i=0;i<nloops;i+=1) //复杂耗时的计算

 PerformLongCalc(1e6)

 endfor

 Variable timeperloop= (ticks-t0)/(60\*nloops)

 KillWindow myProgress //程序运行结束关闭程序进度条窗口

 print "time per loop=",timeperloop

End

Function MySpinner(s)

 STRUCT WMWinHookStruct &s

 if( s.eventCode == 23 ) //IGOR忙碌时向窗口发送eventcode=23的消息

 ValDisplay valdisp0,value= \_NUM:1,win=$s.winName

 DoUpdate/W=$s.winName //更新进度条窗口

 if( V\_Flag == 2 ) //如果进度条窗口鼠标点击按钮，则V\_flag=2，表示退出

 KillWindow $s.winName //关闭进度条窗口，程序也自动结束。

 return 1

 endif

 endif

 return 0

End

//模拟一个耗时的复杂函数

Function PerformLongCalc(nmax)

 Variable nmax

 Variable i,s

 for(i=0;i<nmax;i+=1)

 s+= sin(i/nmax)

 endfor

End

### 7.4.1

function aftercompiledhook()

 print "all procedures are compiled!"

end

Function BeforeFileOpenHook(refNum,fileName,path,type,creator,kind)

 Variable refNum,kind

 String fileName,path,type,creator

 variable handled=0

 return handled

end

Function beforeFileOpenHook (refNum,fileName,path,type,creator,kind)

 Variable refNum,kind

 String fileName,path,type,creator

 variable handled=1

 string s1= "The file refrence number is "+num2str(refNum)

 s1=s1+" and the find kind is thought as "+num2str(kind)

 print s1

 print "FIle info:",fileName,path,type,creator

 pathinfo $path

 print "Symbolic Path "+path+" points to "+S\_Path

 return handled

end

SetIgorHook [/K/L] [hookType [=procName ] ]

setigorhook BeforeFileOpenHook=myhook

### 7.4.2

Function MyWindowHook(s)

 STRUCT WMWinHookStruct &s

 Variable hookResult = 0 // 0表示没有处理，1表示已经处理

 switch(s.eventCode)

 case 11: // 键盘消息

 switch (s.keycode)

 case 28:

 Print "Left arrow key pressed."

 hookResult = 1

 break

 case 29:

 Print "Right arrow key pressed."

 hookResult = 1

 break

 case 30:

 Print "Up arrow key pressed."

 hookResult = 1

 break

 case 31:

 Print "Down arrow key pressed."

 hookResult = 1

 break

 endswitch

 break

 endswitch

 return hookResult // If non-zero, we handled event and Igor will ignore it.

End

Function DemoWindowHook()

 DoWindow/F DemoGraph // 窗口是否存在?

 if (V\_flag == 0)

 Display /N=DemoGraph // 创建窗口

 SetWindow DemoGraph, hook(MyHook) = MyWindowHook

 endif

End

setwindow winname,hook(hookname)=procname

GetWindow $s.winName activeSW //获取活动窗口，窗口名字保存在S\_Value中

String activeSubwindow = S\_value

if (CmpStr(activeSubwindow,"G0") != 0)

 return 0

endif

Structure WMWinHookStruct

 char winName[MAX\_WIN\_PATH+1] // 所属窗口

 STRUCT Rect winRect // 窗口坐标（局域）

 STRUCT Point mouseLoc // 鼠标位置

 Variable ticks //事件发生时的时间

 Int32 eventCode //事件消息代码

 char eventName[MAX\_OBJ\_NAME+1] // 事件名字

 Int32 eventMod // 组合键信息

 char menuName[255+1] //菜单名字

 char menuItem[255+1] // 菜单项

 char traceName[MAX\_OBJ\_NAME+1] //Cursor所处的曲线名字

 char cursorName[2] // Cursor名字（A或B）

 Variable pointNumber // Cursor在wave中位置（x方向或一维）

 Variable yPointNumber //Cursor在wave中位置（y方向）

 Int32 isFree // 1表示没有cursor

 Int32 keycode // 键盘字符对应的ASCII码

 char oldWinName[MAX\_OBJ\_NAME+1]// 窗口名字

 Int32 doSetCursor // 1表示设置鼠标形状

 Int32 cursorCode // 鼠标代码，每一个代码对应一个鼠标形状

 Variable wheelDx // 鼠标滚轮水平方向滚动大小

 Variable wheelDy // 鼠标滚轮竖直方向滚动大小

EndStructure

function ff(s)

 struct wmwinhookstruct &s

 variable handled=0

 switch(s.eventcode)

 case 11: //表示键盘键按下

 SetVariable setvar0,value=\_STR:num2char(s.keycode)

 ValDisplay valdisp0,value= \_NUM:s.keycode

 handled=1

 break

 endswitch

 return handled

end

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(150,77,423,188)

 SetVariable setvar0,pos={24,29},size={50,16},value= \_STR:""

 ValDisplay valdisp0,pos={144,32},size={50,13},limits={0,0,0},barmisc={0,1000}

 ValDisplay valdisp0,value= #"0"

 TitleBox title0,pos={26,10},size={54,12},title="Character",frame=0

 TitleBox title1,pos={140,10},size={30,12},title="ASCII",frame=0

 setwindow panel0,hook(myhook)=ff

EndMacro

swtich(eventCode)

 case 3://mosue down

 //do something

 break

 default:

 break

endswitch

function ff(s)

 struct wmwinhookstruct &s

 string s0

 variable n

 switch(s.eventcode)

 case 3:

 s0=getuserdata("","","s0")

 n=str2num(s0)

 n=n+1

 if(n>32)

 n=0

 endif

 s.dosetcursor=1

 s.cursorcode=n

 setwindow kwTopWin,userdata(s0)=num2str(n)

 break

 endswitch

end

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(150,77,450,277)

 setwindow kwTopWin,userdata(s0)="0",hook(myhook)=ff

EndMacro

function hookfunction(infostr)

 string infostr

 string eventstr=stringbykey(“EVENT”,infostr)

 …

 return statuscode //0表示由igor接着处理，1表示已经处理。

end

### 7.4.3

variable v1,v2

v1=v2

v1:=v2

make/O wave1

wave1:=sin(K0\*x/16)

objectname:=expression

setformula objectname,expression\_str

setformula v1,”v2”

setformula v1,”v1:=v2”

function f1(v)

 variable v

 return v+1

end

variable v1,v2

v1:=f1(v2)

function f1(v)

 variable v

 print “you have changed v2\r”

end

 K0=1
 Make/O wave1:=sin(K0\*x/16)
 Display /W=(4,53,399,261) wave1
 ControlBar 23
 SetVariable setvar0,size={60,15},value=K0

### 7.5.1

NewFIFO //创建新的FIFO对象

KillFIFO //删除FIFO对象

NewFIFOChan //创建FIFO通道

CtrlFIFO //操作已有的FIFO对象

FIFO2Wave //从FIFO对象中提取数据

AddFIFOData //向FIFO对象中添加数据

AddFIFOVectData //向FIFO对象中添加矢量数据

FIFOStatus //获取FIFO对象的状态

Chart //示波器，用于实时显示FIFO数据

ControlInfo //获取控件的信息

function test()

 fifostatus/Q dave

 if(0==V\_Flag)

 newfifo dave

 newfifochan/D dave,chan,0,1,-2,2,""

 ctrlfifo dave,deltaT=0.1

 make/O/N=2000,aa

 setscale/P x,0,0.1,aa

 aa=sin(x)

 variable/G timebase=0

 endif

 ctrlfifo dave,start

 execute "Panel0()"

 ctrlnamedbackground backtastk,period=6,proc=updatedata,start

end

function updatedata(s)

 struct wmbackgroundstruct &s

 wave w=aa

 nvar i0=timebase

 if(i0>=2000)

 execute "stopfifo(\”\”)"

 return 1

 endif

 addfifodata dave,w[i0]

 i0=i0+1

 return 0

end

Window Panel0() : Panel

 PauseUpdate; Silent 1 // building window...

 NewPanel /W=(195,271,675,596)

 Chart foo,pos={1,2},size={470,277},fifo= dave

 Chart foo,chans={0},omode=0,umode=3

 chart foo,linemode(0)=1

 Button button0,pos={4,287},size={104,23},proc=stopfifo,title="stop"

EndMacro

Function stopfifo(ctrlName) : ButtonControl

 String ctrlName

 ctrlnamedbackground backtastk,stop

 killfifo dave

End

### 7.5.2

#pragma rtGlobals=3 // Use modern global access method and strict wave access.

function vdttest()

 make/O data

 setscale/I x,0,2\*pi,data

 data=sin(x)

 variable i,N

 N=numpnts(data)

 vdtoperationsport2 COM3

 vdtopenport2 COM3

 for(i=0;i<N;i+=1)

 vdtwrite2 num2str(data[i])+","

 endfor

 vdtcloseport2 COM3

end

### 7.6.1

PlaySound [/A [=a ]] soundWave

Make/B/O/N=1000 sineSound // 数字化声音（8位）

SetScale/P x,0,1e-4,sineSound // 采样频率为10KHz

sineSound= 100\*sin(2\*Pi\*1000\*x) // 声音的频率为1KHz

PlaySound sineSound //播放声音

function f1()

 controlinfo popup0

 wave/Z w=$S\_Value,i0

 if(waveexists(w)==0)

 abort "please select a sound wave"

 endif

 if(strlen(tracenamelist("",";",1))==0)

 appendtograph w[0,4410][0]

 ModifyGraph mirror=1,noLabel=2,standoff=0,tick=3

 ModifyGraph axOffset(left)=-6.42857,axOffset(bottom)=1.53846

 SetAxis left -20000,20000

 ModifyGraph mode=2

 elseif(cmpstr(nameofwave(w),stringfromlist(0,tracenamelist("",";",1))))

 removefromgraph $stringfromlist(0,tracenamelist("",";",1))

 appendtograph w[0,4410][0]

 ModifyGraph mirror=1,noLabel=2,standoff=0,tick=3

 ModifyGraph axOffset(left)=-6.42857,axOffset(bottom)=1.53846

 SetAxis left -20000,20000

 ModifyGraph mode=2

 endif

 nvar tgID,imax,pouse

 imax=round(dimsize(w,0)/4410)

 if(pouse==0)

 i0=0

 playsound/A=1 w

 endif

 if(pouse==1)

 duplicate/O/R=[i0\*4410,imax\*4410][] w,w1

 playsound/A=1 w1

 pouse=0

 endif

 //crate movie

 newpath path1, "E:"

 newmovie/O/A/P=path1/S=w/F=10 as "mymovie.avi"

 //start new thread to get accurate time interval

 //start basktask to update sound wavefrom display

 tgID=threadgroupcreate(1)

 threadstart tgid,0,f2(i0)

 ctrlnamedbackground task,period=6,proc=f3,start

end

threadsafe function f2(i0)// worker func

 wave i0

 variable t0

 variable i=0,init=i0,t,dt=100

 t0=stopmstimer(-2)

 do

 sleep/T 3

 t=stopmstimer(-2)

 if(round((round(t/1000-t0/1000)-i\*dt)/1000)==0)

 i0=i+init

 i+=1

 endif

 while(1)

end

function f3(s) //backtask

 struct wmbackgroundstruct &s

 controlinfo popup0

 wave w=$S\_Value

 wave i0

 nvar imax

 if(i0>=imax)

 f4()

 return 1

 endif

 ReplaceWave trace= $nameofwave(w), w[i0\*4410,(i0+1)\*4410][0]

 slider slider0,value=i0

 doupdate/w=soundwave

 duplicate/O/R=[i0\*4410,(i0+1)\*4410] w,w1

 //add frame and audio

 addmovieframe

 addmovieaudio w1

 return 0

end

function f4() //stop

 nvar tgid

 variable var= threadgrouprelease(tgid)

 ctrlnamedbackground task,stop

 closemovie

end

Function ButtonProc(ctrlName) : ButtonControl //play

 String ctrlName

 f1()

End

Function ButtonProc\_1(ctrlName) : ButtonControl //pouse

 String ctrlName

 f4()

 nvar pouse

 pouse=1

 wave w=nonew

 playsound/A=2 w

End

Function ButtonProc\_2(ctrlName) : ButtonControl //stop

 String ctrlName

 f4()

 wave w=nonew

 nvar pouse

 pouse=0

 playsound/A=2 w

End

Window soundwave() : Graph

 PauseUpdate; Silent 1 // building window...

 Display /W=(280.5,235.25,672.75,442.25)

 Slider slider0,pos={30,223},size={456,6}

 Slider slider0,limits={0,2396,1},value= 513,side= 0,vert= 0

 Button button0,pos={236,242},size={50,20},proc=ButtonProc,title="Play"

 Button button1,pos={424,242},size={50,20},proc=ButtonProc\_2,title="stop"

 Button button2,pos={326,242},size={50,20},proc=ButtonProc\_1,title="pouse"

 PopupMenu popup0,pos={31,240},size={100,20},bodyWidth=100

 PopupMenu popup0,mode=1,popvalue="select wave",value= #"wavelist(\"\*\",\";\",\"DIMS:2\")"

EndMacro

function mp3player()

 dowindow/F soundwave

 if(!V\_flag)

 execute "soundwave()"

 make/N=1/O i0=0

 variable/G tgID,imax,pouse=0

 make/O/N=2 nonew

 setscale/P x,0,0.0001,"s",nonew

 endif

end

### 7.6.2

playmovie as filenamestr

7.7.1 if(aborttest)

 abort “some thing with wrong happened”

endif

function test()

 string s

 make/O a={1,2,3}

 a[5]=100 //错误：wave访问越界

 print s //错误：未赋值使用字符串

 variable i,n

 for(i=0;i<=100;i+=1)

 n+=i

 endfor

 print n

end

<expression>;abortonrte

7.7.2 function test()

 try

 abort

 print "try-catch"

 catch

 print "catch-endtry"

 print geterrmessage(V\_abortcode)

 endtry

end

function test()

 try

 string s

 print s;abortonrte

 print "try"

 catch

 print geterrmessage(V\_abortcode)

 print "catch"

 endtry

end

### 7.7.3

function test()

 NewNotebook/N=nb1/F=0

 variable i=1

 string s1

 do

 s1=geterrmessage(i)

 if(strlen(s1)==0)

 break

 endif

 notebook nb1,text=num2str(i)+": "+s1+"\r"

 i+=1

 while(1)

end

function test()

 string s

 print s

 variable v=getrterror(0)

 print geterrmessage(v)

end

function test()

 string s

 print s

 variable v=getrterror(1)

 print geterrmessage(v)

end

### 7.8.1

newpath path1,”E:Igor programming:”

pathinfo path1

string list=indexedfile(path1,-1,”.pxt”)

Open [ /A /C=creatorStr /D[=mode ] /F=fileFilterStr /M=messageStr /MULT=m /P=pathName /R /T=typeStr /Z[=z] ] refNum [as fileNameStr]

open ref //以读写方式打开文件，open会提供一个打开文件对话框

open ref as filenamestr //以读写方式打开

open/R ref as filenamestr //只读方式打开

open/A ref filenamestr //追加方式打开

open/R/D ref //只读方式打开文件，总是打开对话框，选择单个文件

open/D=2 ref filenamestr //保存文件，如果filenamestr信息不全，打开保存对话框

open/P=pathname ref as filenamestr//打开pathname指定路径下的文件filenamestr

open/R/F=filterstr //只读方式打开，filterstr提供过滤字符串

variable ref

open ref as “E:tmp.dat”

variable ref

open/D/R ref //从文件打开对话框选择一个文件（注意文件并没有打开）

open/R ref as S\_fileName //真正打开该文件

open/D ref //从文件保存对话框创建一个文件（注意文件并没有创建）

open ref as S\_filename //真正创建文件

fprintf ref,”%s:%g”,str,var

fsetpos ref,filepos

fstatus ref

fsetpos ref,V\_logEOF

fstatus ref

freadline ref,str

FBinRead [/F=f ] refNum, objectName

Fbinwrite [/F=f ] refNum, objectName

### 7.8.2

function filedemo1()

 variable v1,v2,v3

 open v1 as "E:tmp:data1.dat"

 open v2 as "E:tmp:data2.dat"

 open v3 as "E:tmp:data3.dat"

 make/N=100 w1,w2,w3

 setscale/P x,0,0.01,w1,w2,w3

 w1=sin(x)

 w2=x^2

 w3=x

 wfprintf v1,"%g\r\n",w1

 fbinwrite v2,w2

 fbinwrite v3,w3

 close/A

end

function filedemo2()

 variable v1,v2,v3

 open/R v1 as "E:tmp:data1.dat"

 open/R v2 as "E:tmp:data2.dat"

 make/O/N=1000 w1,w2 //long enough wave to hold the data

 variable i,v

 string s1

 for(i=0;i<1000;i+=1)

 freadline v1,s1

 if(!cmpstr(s1,""))

 break

 endif

 sscanf s1,"%g",v

 w1[i]=v

 endfor

 redimension/N=(i) w1

 fbinread v2,w2

 fstatus v2

 redimension/N=(round(V\_logEOF/8)) w2

 display w1

 display w2

 close/A

end

function filedemo3()

 variable v

 newpath/O path1,"E:tmp:"

 string list=indexedfile(path1,-1,".dat")

 print list

 string s1=stringfromlist(2,list)

 open/R/P=path1 v as s1

 make/O/N=1000 w

 fbinread v,w

 fstatus v

 redimension/N=(round(V\_filepos/8)) w

 display w

end

newpath pathname, “E:tmp:”

string list=indexedfile(pathname,-1,”.pxt”)

filenamestr=stringfromlist(i,list)

loaddata/P=pathname filenamestr

function filedemo4()

 variable v

 open/D v

 open v as S\_Filename

 make/O w

 fbinwrite v,w

 close v

end

### 7.9.2

 Window Panel0() : Panel //创建窗口生成macro

 PauseUpdate; Silent 1

 NewPanel /W=(207,92,443,184)

 Button button0,pos={153,33},size={50,20}

 SetVariable setvar0,pos={33,33},size={76,16}

 setvariable setvar0,value=root:demo\_package:var1

 //servar0绑定一个全局变量var1，这个变量在窗口创建时必须存在

EndMacro

function LoadmyPanel()

 string curr=getdatafolder(1) //获取当前数据文件夹并保存在curr中

 dowindow/F Panel0 //检查panel0是否已经打开

 if(0==V\_flag) //flag=0，没有打开；flag=1，已经打开

 dfref dfr=root:demo\_package

 if(!DataFolderRefStatus(dfr)) //检查root:demo\_package是否存在

 newdatafolder/S root:demo\_package

 //创建root:demo\_package

 variable/G var1 //创建var1变量

 endif

 execute "panel0()" //创建界面

 endif

 setdatafolder curr //恢复原来的数据文件夹

end

menu "mymenu"

 "load my panel",LoadmyPanel() //创建菜单，通过菜单打开程序

end

### 7.10.1

print ticks

variable t0=ticks

variable t0=stopmstimer(-2)

<do something>

variable t1=stopmstimer(-2)

print (t1-t0)/1000000

### 7.10.2

SetWindow graphName,hook(hookname)=userproc

function userproc(s)

 struct wmwinhookstruct &s

 switch(s.eventcode)

 case 7://Cusromoded event occurred

 //do something

 break

 endswitch

end

root:Winglobals:Graphname:S\_CursorAInfo

root:Winglobals:Graphname:S\_CursorBInfo

GRAPH:*graphName*;CURSOR:<A - J>;TNAME:*traceName*;MODIFIERS:*modifierNum*;
ISFREE:*freeNum*;POINT:*xPointNumber*;[YPOINT:*yPointNumber*;]

Function CursorGlobalsForGraph()

 String graphName= WinName(0,1)

 if( strlen(graphName) )

 String df= GetDataFolder(1);

 NewDataFolder/O root:WinGlobals

 NewDataFolder/O/S root:WinGlobals:$graphName

 String/G S\_CursorAInfo, S\_CursorBInfo

 SetDataFolder df

 endif

End

Function RemoveCursorGlobals()

 String graphName= WinName(0,1)

 if( strlen(graphName) )

 KillDataFolder root:WinGlobals:$graphName

 endif

End

Function CursorDependencyForGraph()

 String graphName= WinName(0,1)

 if( strlen(graphName) )

 String df= GetDataFolder(1);

 NewDataFolder/O root:WinGlobals

 NewDataFolder/O/S root:WinGlobals:$graphName

 String/G S\_CursorAInfo, S\_CursorBInfo

 Variable/G dependentA

 SetFormula dependentA, "CursorMoved(S\_CursorAInfo, 0)"

 Variable/G dependentB

 SetFormula dependentB,"CursorMoved(S\_CursorBInfo, 1)"

 SetDataFolder df

 endif

End

Function CursorMoved(info, isB)

 String info

 Variable isB // 0 if A cursor, nonzero if B cursor

 Variable result= NaN // error result

 String topGraph= WinName(0,1)

 String graphName= StringByKey("GRAPH", info)

 if( CmpStr(graphName, topGraph) == 0 )

 String tName= StringByKey("TNAME", info)

 if( strlen(tName) ) // cursor still on

 String cn

 Variable xVal

 if( isB )

 xVal= hcsr(B)

 cn= "Cursor B"

 else

 xVal= hcsr(A)

 cn= "Cursor A"

 endif

 WAVE w= TraceNameToWaveRef(graphName, tName)

 Variable pointNum= NumberByKey("POINT",info)

 Variable x1= pnt2x(w,pointNum-2)

 Variable x2= pnt2x(w,pointNum+2)

 result= mean(w,x1,x2)

 Print cn+" on "+tName+" moved to x= ",xVal,"mean= ",result

 endif

 endif

 return result

End

### 7.10.3

string s1=”hello,Igor”

print s1[0]

stringfromlist

strlen

strsearch

stringbykey

cmpstr

num2str

greplist

grepstring

grep

splitstring

wave w=$stringfromlist(0,s1) //获取第一个wave名字及其引用

modifycontrol $stringfromlist(0,s1),disable=0 //获取第一个控件名字及其引用，设为可用

funcref modfun f1=$stringfromlist(0,s1) //获取第一个函数名字及其引用

if(strlen(s1)!=0)

 //do something

endif

string s1=”””;

function/S test()

 controlinfo list0

 string s1= greplist(s\_recreation,"selWave",0,"\r")

 string s2,e1

 e1="(?<=selWave=)([a-zA-Z\d:]+)" //正则表达式，查找selWave对应的wave

 splitstring/E=e1 s1,s2

 return s2

end

root:wave0 //这里wave0位于root目录下

wave sw=$test()