```
//MODEL BOUTON CLOUD USING CONOURS FROM NEUROLUCIDA//
//These scripts will attempt to model a cloud of boutons using the
//outlines of the cloud as drawn on serial sections in Neurolucida


// *** sepWaves *** //
// Function to seperate input contour(from neurolucida) into different
// waves based on Z level
// By Jeremy Atherton, 30 July 2006
Function sepWaves(xWave,yWave,zWave)
Wave xWave,yWave,zWave

Variable i,k=-100000
String newNameX,newNameY,newNameZ
for(i=0;i<numpnts(zWave);i+=1)
if(zWave[i]!=k)
k=zWave[i]
newNameX="contX" + num2str(k)
newNameY="contY" + num2str(k)
newNameZ="contZ" + num2str(k)
Make/N=0 $newNameX $newNameY $newNameZ
Wave contX = $newNameX
Wave contY = $newNameY
Wave contZ = $newNameZ
endif
InsertPoints numpnts(contX),1,contX
contX[numpnts(contX)-1]=xWave[i]
InsertPoints numpnts(contY),1,contY
contY[numpnts(contY)-1]=yWave[i]
InsertPoints numpnts(contZ),1,contZ
contZ[numpnts(contZ)-1]=k
endfor
End


// *** insidePolygon *** //
// Determine if a point is inside a polygon
//
```

```
// Returns 1 for 'inside', 0 for 'outside'
//
// This script is based on C code by Paul Bourke found at
// at http://local.wasp.uwa.edu.au/~pbourke/geometry/insidepoly/
// which is a version of the 'crossings test', an application of
// Jordan Curve Theorem. See Haines, Eric, "Point in Polygon Strategies,"
// Graphics Gems IV, ed Paul Heckbert, Academic Press, p.24-46 1994.
Function insidePolygon(xIN,yIN,xP,yP)
Wave xIN,yIN
Variable xP,yP

Duplicate xIN xWave //Don't want to alter the original data, so duplicate it
Duplicate yIN yWave

//Some variables
Variable counter = 0,i,xinters,x1,y1,x2,y2,N=numpnts(xWave)

//Add the last point in the contour to the beginning so that the contour is closed
InsertPoints 0,1,xWave
xWave[0]=xWave[numpnts(xWave)-1]
InsertPoints 0,1,yWave
yWave[0]=yWave[numpnts(yWave)-1]

x1 = xWave[0]
y1 = yWave[0]

//Count vertex crossings
for (i=1;i<=N;i+=1)
x2 = xWave[mod(i,N)]
y2 = yWave[mod(i,N)]
if (yP > min(y1,y2))
if (yP <= MAX(y1,y2))
if (xP <= MAX(x1,x2))
if (y1 != y2)
xinters = (yP-y1)*(x2-x1)/(y2-y1)+x1
if (x1 == x2 || xP <= xinters)
counter+=1
endif
```

```
endif
endif
endif
endif
x1 = x2
y1 = y2
endfor

KillWaves xWave,yWave

if (mod(counter,2) == 0)
return(0)
else
return(1)
endif
End

// *** modelCont *** //
// This is the main modelling script that generates the random data points, finds
// the correct contour (as generated by sepwaves() to test each point against,
// and runs the insidePolygon script on that contor
//
// By Jeremy Atherton, 30 July 2006

// IMPORTANT! Run sepwaves() to get the contours before using this function
// failure to run sepwaves() causes this script to loop infinitely--there is no fail-safe code!
Function modelCont(xCont,yCont,zCont,num)
Wave xCont,yCont,zCont
Variable num //number of points to model (should be number of points in real cloud)

// Get min, max, and range from X values
Variable xMin,xMax,xRange
Wavestats/Q xCont
xMin=V_min
xMax=V_max
xRange=xMax-xMin

// Get min, max, and range from XYvalues
```

```
Variable yMin,yMax,yRange
Wavestats/Q yCont
yMin=V_min
yMax=V_max
yRange=yMax-yMin

// Work out the z steps
Variable i,z0,stepTemp,step=10000
Make/N=0 zSteps
for(i=0;i<numpnts(zCont);i+=1)
if(zCont[i]!=z0)
if(zCont[i]-z0<step)
step=abs(zCont[i]-z0)
endif
insertpoints numpnts(zSteps),1,zSteps
z0=zCont[i]
zSteps[numpnts(zSteps)-1]=z0
endif
endfor

// Sanity check: print out the z steps
Sort/R zSteps,zSteps
print zSteps
print step

Variable zMin,zMax,zRange
Wavestats/Q zCont
zMin=V_min-step
zMax=V_max
zRange=zMax-zMin
print zMin
print zMax

Variable xCentre=xMin+(xRange/2)
Variable yCentre=yMin+(yRange/2)
Variable zCentre=zMin+(zRange/2)

// the main body
```

```
Variable k,zLevel
Variable newX,newY,newZ
String newNameX,newNameY
Make/N=(num) xModel,yModel,zModel //waves to dump our new x, y & z coordinates into
do
newX=enoise(xRange/1.9)+xCentre //random x
newY=enoise(yRange/1.9)+yCentre //random y
newZ=enoise(zRange/1.9)+zCentre // random z

// Find the contour that corresponds to the random z value
zLevel=10000
for(i=0;i<numpnts(zSteps);i+=1)
if(newZ<=zSteps[0] && newZ>=zSteps[numpnts(zSteps)-1]-step)
if(newZ<=zSteps[i] && newz>zSteps[i]-step)
zLevel=zSteps[i]
endif
else
break
endif
endfor

newNameX = "contX" + num2istr(zLevel)
newNameY = "contY" + num2istr(zLevel)

// If we found a contour check if the x and y values are within that contour
if(zLevel!=10000)
if(insidePolygon($newNameX,$newNameY,newX,newY))
xModel[k]=newX
yModel[k]=newY
zModel[k]=newZ
k+=1
endif
endif
while(k<num) // loop until the number of model points is equal to the number of boutons
KillWaves zSteps
End

// *** repeatModel *** //
```

```
// This script will run modelCont() an arbitrary number of times
// Each time, the output from the model is given nearest neighbour and 'autocorrelogram'
// analysis (see Boutons.ipf) - the results of this analysis are averaged using the WavesAverage() script

// IMPORTANT! Run sepwaves() to get the contours before using this function
Function repeatModel(xCont,yCont,zCont,num,repeats)
Wave xCont,yCont,zCont
Variable num,repeats

Make/N=(repeats) statsWaveMean statsWaveSD

Variable i
for(i=0;i<repeats;i+=1)
modelCont(xCont,yCont,zCont,num)

nearest(xModel,yModel,zModel, nograph=1)
String nS=makeName("nearest")
Duplicate nearestdistance_Hist $nS
WaveStats/Q nearestdistance
statsWaveMean[i]=V_avg
statsWaveSD[i]=V_sdev
killwaves nearestdistance,nearestbouton,nearestdistance_Hist
autocorr(xModel,yModel,zModel, nograph=1)
nS=makeName("auto")
Duplicate distancewave_Hist $nS
killwaves distancewave, distancewave_Hist
killwaves xModel,yModel,zModel
endfor

WavesAverage("near", "near")
WavesAverage("auto", "auto")

WaveStats/Q statswaveMean
print "Mean = " & V_avg
WaveStats/Q statswaveSD
print "SD = " & V_avg
End
```

```
// *** WavesAverage(baseName, destName) *** //
// Produces a new wave, each point of which contains the average
// of the corresponding points of a number of source waves.
// All waves whose name starts with the specified base name
// are source waves.

// WARNING - This function assumes that all waves that start with the base
// name have the same number of points and that there is at least
// one such wave. There is no fail-safe code.
Function WavesAverage(baseName, destName)
String baseName // name for source wave
String destName // name for destination wave
String wn // contains the name of a particular wave
String wl // contains a list of wave names
Variable index=0
// get list of waves whose names start with baseName
wl = WaveList(baseName+"*", ";", "")
// Make destination wave based on the first source wave
wn = StringFromList(0, wl)
Duplicate/O $wn, $destName
Wave/D dest = $destName//create wave reference for destination
dest = 0
do
wn = StringFromList(index, wl) // get next wave
if (strlen(wn) == 0) // no more names in list?
break // break out of loop
endif
Wave/D source = $wn // create wave reference for source
dest += source // add source to dest
index += 1
killwaves source
while (1) // do unconditional loop
dest /= index // divide by number of waves
End



// *** modelAv() *** //
```

```
// Unused script -- ignore!
Function modelAv(num)
Variable num //number of times that the model was run

make/N=(numpnts(xModel0)) xModel,yModel,zModel

Variable i
String wxS,wyS,wzS
for(i=0;i<num;i+=1)
wxS="xModel"+num2istr(i)
wyS="yModel"+num2istr(i)
wzS="zModel"+num2istr(i)

//if(waveExists($wxS)==1)
Wave wx=$wxS
Wave wy=$wyS
Wave wz=$wzS

xModel+=wx
yModel+=wy
zModel+=wz

killwaves wx,wy,wz
//endif
endfor

xModel=xModel/num
yModel=yModel/num
zModel=zModel/num

make/N=(numpnts(xModel),3) modelData
modelData[][0]=xModel[p]
modelData[][1]=yModel[p]
modelData[][2]=zModel[p]
End
```