**P\_frequency; Number of peaks per millisecond over a window of 4096 samples**

halfWindow = runParams.clusterSize;

sigma = halfWindow/2;

slidingFreqTrace = zeros([length(trace) 1]);

midPeak =round( (startIndexs+endIndexs)/2);

slidingFreqTrace(midPeak)=1;

impulse =exp(-1\*( (-halfWindow:halfWindow) ./sigma).^2);

slidingFreqTrace=conv(slidingFreqTrace,impulse,'same');

**P\_maxAmplitude**

max\_amplitude = max(chunk);

**P\_averageAmplitude**

averageAmplitude = mean(chunk(2:(length(chunk)-2)));

**P\_topAverage; average of peak above half maximum**

topAmp=mean(chunk(top));

**P\_peakWidth**

top=find(chunk>averageAmplitude);

peakwidth = length(top )/50000\*1000;

**P\_roughness; standard deviation of the spike above half maximum height**

roughness = std(chunk(top)/topAmp);

**P\_totalPower; total fft power (square root of the sum of power spectrum)**

cE = vertcat(cE, ones([FFTSize - length(cE),1])\*cE(end));

cEFFT =cEFFT + abs(fft(cE));

cc=cc+1;

sigma =cEFFT / cc;

alpha=.5;

N = size(chunk,1);

Yf = fft(chunk)/N;

Pyf = abs(Yf).^2;

sigma=abs(sigma).^2/N^2;

W=((1-alpha)\*Pyf-alpha\*sigma)./Pyf;

W(W<0)=0;

spec = (W.\*Yf);

spec=abs(spec(1:round(end/2)));

spec=spec.^2;

spec(1)=0;

TotalPower=sum(spec(1:end));

TotalPower=(TotalPower)^.5;

**P\_iFFTLow; average of three points within the first frequency band (0.9, 1.8, 2.7kHz)**

spec = (W.\*Yf);

rms =(sum( (chunk- mean(chunk)).^2 )/length(chunk)) ^.5;

spec=abs(spec(1:round(end/2)));

spec=spec.^2;

spec(1)=0;

powerspec=(spec./TotalPower);

specLength =length(powerspec);

n1=round(specLength/2);

n2=specLength-5;

powerspec(5)+powerspec(6)+powerspec(7);

**P\_iFFTMedium; average of three points within the middle frequency band (9.3, 10.2, 11.1kHz)**

specLength =length(powerspec);

n1=round(specLength/2);

n2=specLength-5;

powerspec(n1)+powerspec(n1+1)+powerspec(n1+2);

**P\_iFFTHigh; average of three points within the highest frequency band (23.2, 24.1, 25kHz)**

powerspec(n2)+powerspec(n2+1)+powerspec(n2+2);

**P\_Even\_FFT; Sum of all even frequencies from the non downsampled FFT**

sum( powerspec(1:2:specLength));

**P\_Odd\_FFT; Sum of all odd frequencies from the non downsampled FFT**

sum( powerspec(2:2:specLength));

**P\_OddEvenRatio; Ratio of the odd to the even FFT sums.**

peakParams.P\_Odd\_FFT/peakParams.P\_Even\_FFT;

**P\_peakFFT\_Whole**

[TotalPowerW,powerspecW, misMatch,tilt]=DenoiseSpecWhole(chunk,chunkempty,256);

misMatch=spec(1)-min(chunk);

tilt= real(spec(2));

third = sum(powerspec(1:3:specLength));

halfs = sum(powerspec(1:floor(end/2)))/sum(powerspec(floor(end/2):end));

powerspecW(end) = halfs;

powerspecW(end-1) = third;

powerspecW(end-2) = misMatch;

powerspecW(end-3) = tilt;

**P\_peakFFT; Downsampled FFT spectrum.**

peakCoef(cc3) = mean(powerspec(indxs(k):indxs(k+1)));

**P\_highLow\_Ratio; Ratio of FFT amplitude in the 22.3-25kHz band to that in the 0-2.7kHz band**

P\_highLow\_Ratio = peakCoef(round(end\*.75))/peakCoef(1);

**C\_peaksInCluster; Number of peaks in the cluster.**

assignmentTrace = zeros(size(slidingFreqTrace));

midPeak =round( (startIndexs+endIndexs)/2);

for I=1:clusterIndex

assignmentTrace(clusterStartI(I):clusterEndI(I))=I;

end

clusterAssignment = assignmentTrace(midPeak);

clusterOccupancy = histc(clusterAssignment,1:max(clusterAssignment));

C\_peaksInCluster = clusterOccupancy;

**C\_frequency; Number of peaks in cluster divided by ms length of cluster.**

peaksInCluster/length(chunk)\*1000;

**C\_averageAmplitude; Average amplitude of all cluster peaks.**

averageAmplitude = mean(chunk);

**C\_topAverage; Average amplitude of all peaks above half maximum.**

top=find(chunk>averageAmplitude);

topAmp=mean(chunk(top));

**C\_clusterWidth; Cluster time in ms.**

(length(top)/50000\*1000);

**C\_roughness; Standard deviation of whole cluster signal.**

roughness = std(chunk(top)/topAmp);

**C\_maxAmplitude; Average of the max of all the spikes in cluster.**

max(chunk(2:(length(chunk)-2))) - min(chunk(2:(length(chunk)-2)));

**C\_clusterFFT\_Whole**

FFTSize = 256;

powerspec=zeros([FFTSize 1]);

cc3=1;

for k=1:FFTSize

try

powerspec(cc3) = mean( spec(indxs(k):indxs(k+1) )) ;

catch mex

disp (mex);

end

cc3=cc3+1;

end

powerSpec2= zeros([1 51]);

step = floor(length(powerspec)/51);

for I=1:51

powerSpec2(I)= powerspec(I\*step);

end

powerspec = powerSpec2;

**C\_totalPower; Square root of the sum of the power spectrum.**

cE = vertcat(cE, ones([FFTSize - length(cE),1])\*cE(end));

cEFFT =cEFFT + abs(fft(cE));

cc=cc+1;

sigma =cEFFT / cc;

alpha=.5;

N = size(chunk,1);

Yf = fft(chunk)/N;

Pyf = abs(Yf).^2;

sigma=abs(sigma).^2/N^2;

W=((1-alpha)\*Pyf-alpha\*sigma)./Pyf;

W(W<0)=0;

spec = (W.\*Yf);

rms =(sum( (chunk- mean(chunk)).^2 )/length(chunk)) ^.5;

spec=abs(spec(1:round(end/2)));

spec=spec.^2;

spec(1)=0;

TotalPower=sum(spec(1:end));

powerspec=(spec./TotalPower);

TotalPower=(TotalPower)^.5;

**C\_iFFTLow; Average of thress points within the first frequency band (0.136, 0.273, 0.410 kHz)**

powerspec(5)+powerspec(6)+powerspec(7);

**C\_iFFTMedium; Average of thress points within the middle frequency band (12.710, 12.847, 12.983 kHz)**

specLength =length(powerspec);

n=specLength;

n1=round(specLength/2);

n2=n-5;

powerspec(n1)+powerspec(n1+1)+powerspec(n1+2);

**C\_iFFTHigh; Average of thress points within the highest frequency band (24.726, 24.863, 25 kHz)**

powerspec(n2)+powerspec(n2+1)+powerspec(n2+2);

**C\_freq\_Maximum\_Peaks; Frequency of 4 dominant peaks in spectrum, ordered by the height of the peaks.**

**(Cluster Frequency Location of Maximum Peaks)**

Lspec=powerspec(3:end-15);

Lspec = smooth(Lspec,31);

Lspec=Lspec-Lspec(end)+.01;

Lspec=Lspec./sum(Lspec);

f=smooth(Lspec,201)';

e=mean(powerspec(end-20:end))\*2;

Lspec= (e + Lspec)./ ( e + f)'-1;

[pks loc]=findpeaks(Lspec);

locs=loc;

KK=2;

while KK<length(locs)

if abs(locs(KK)-locs(KK-1))<51

locs(KK)=[];

pks(KK)=[];

else

KK=KK+1;

end

end

[v idx]=sort(pks,'descend');

locs=locs(idx);

%make sure to put it into the form of a ratio to deal with the different run lengths.

locs = vertcat(locs ,zeros([4 1]));

locs = locs./length(Lspec);

clusterParams.C\_freq\_Maximum\_Peaks=locs(1:4)\*25;

**C\_clusterFFT; Downsampled FFT spectrum of cluster.**

specLength =length(powerspec);%round(length(powerspec)/2);

peakCoef = zeros([nComponents 1]);

%reduce the complexity to just a few parameters. Since the spacing is only dependant on.

indxs = round( specLength \* (0: (1/nComponents) :1).^2);

if indxs(1)==0;

indxs=indxs+1;

end

for iI=2:length(indxs)

if indxs(iI)==indxs(iI-1)

indxs(iI:end)=indxs(iI:end) + 1;

end

end

indxs(indxs>length(powerspec))=length(powerspec);

cc3=1;

for k=1:nComponents

try

peakCoef(cc3) =mean( powerspec(indxs(k):indxs(k+1) ));

catch mex

disp (mex);

end

cc3=cc3+1;

end

**C\_highLow**

mean(peakCoef(round(end\*.75):end))/ mean( peakCoef(1:round(end\*.25)));

**C\_clusterCepstrum; Spectrum of the power spectrum of the cluster, downsampled to 61 points.**

%now handle the cepstrum to get the underlying structure of the data.

peakCoef = zeros([nComponents 1]);

cepstrum = fft(powerspec(2:end));

cepstrum = cepstrum /length(cepstrum);

cepstrum=cepstrum(1:round(end/2));

cepstrum=abs(cepstrum).^2;

cepstrum(1)=0;

%reduce the complexity to just a few parameters. Since the spacing is

%only dependant on

specLength=length(cepstrum);

for k=1:(specLength)

bin =floor( k/(specLength)\*(nComponents-1))+1;

peakCoef(bin) = peakCoef(bin)+cepstrum(k);

end