Appendix C: Component Matrix using Drawn Phonemes

File: constructBaseVector.m

```
allPhonemes = {'p' 't' 'k' 'pcl' 'tcl' 'kcl' 'dx' 'm' 'n' 'ng' 'nx' 's'...
    'z' 'ch' 'th' 'f' 'l' 'r' 'y' 'pau' 'hh' 'eh' 'ao' 'aa' 'uw' 'er'...
    'ay' 'ey' 'aw' 'ax' 'ix' 'b' 'd' 'g' 'bcl' 'dcl' 'gcl' 'q' 'em' 'en'...
    'eng' 'sh' 'zh' 'jh' 'dh' 'v' 'el' 'w' 'h#' 'epi' 'hv' 'ih' 'ae'...
    'ah' 'uh' 'ux' 'oy' 'iy' 'ow' 'axr' 'ax-h' ... up to here from web
    'sil' 'oh' 'ia' 'ea' 'ua'}; % these last few from inspection
timeslice = 4e-2; %40ms slices
wavSamplesPerPhn = 10;

%get all training speaker folders
trainingDir = '/Users/Ash/Documents/ThesisData/wsjcam0/rawdat/si_tr/';
speakerDirs = dir(trainingDir);

% randomly draw a speaker, get files
speakerID = datasample(speakerDirs(4:end),1);
[phnFiles,wavFiles] = getSpeakerFiles(trainingDir,speakerID.name);

% draw a random set of phoneme samples
[phonemeSamples,fs] = drawPhnSamples([trainingDir speakerID.name '/'],...
    timeslice,wavSamplesPerPhn,allPhonemes);
sliceT = timeslice * fs;

% construct a plot of the number of samples drawn by phoneme
numSamples = cellfun(@(x) x(2),...
    cellfun(@size,phonemeSamples,'UniformOutput',false));

% Get V - spectal component matrix
Vspeaker = constructVMatrix(phonemeSamples,ceil(sliceT));

% Get X - output, by concatenating all samples together, taking FFTs
```

```
Y = [];
for a = 1:numel(wavFiles)
    Y = [Y;readwav([trainingDir  speakerID.name '/'...
        wavFiles(a).name])];
end
X = zeros(ceil(sliceT/2+1),floor(numel(Y))/sliceT);
Xindex = 1;
for Yindex = 1:sliceT:numel(Y)-sliceT
    X(:,Xindex) = rfft(Y(Yindex:Yindex+sliceT-1),sliceT);
    Xindex = Xindex+1;
end

% Calculate W - occurences matrix
%W = ones(size(Vspeaker,2),size(X,1));
%W=W.*(Vspeaker'*(X./(Vspeaker*W))./(Vspeaker');



presentPhonemes = allPhonemes(numSamples~=0);
for (i=1:numel(presentPhonemes))
    presentPhonemes{i} = sprintf('/%s/', presentPhonemes{i});
end
presentPhonemes{end+1} = '';

figure()
hold all
p = 10*log10(abs(Vspeaker(1:ceil(sliceT)/2,:)));
p(p<-30)=-30;
%subplot(2,1,1);
surf(p,'EdgeColor','none');
axis xy; axis tight; colormap(jet); view(0,90);
set(gca,'XTick',[1;unique(cumsum(numSamples))],...
    'XTickLabel',presentPhonemes)
title('Vspeaker');
xlabel('components');
ylabel('Frequency Bin');
```

```
% p = 10*log10(abs(X));
% p(p<-30)=-30;
% subplot(2,1,2);
% surf(p,'EdgeColor','none');
% axis xy; axis tight; colormap(jet); view(0,90);
% title('Speech');
% xlabel('Time');
% ylabel('Frequency Bin');
```

File: constructVMatrix.m

```
function [V] = constructVMatrix(phonemeRecordings,pointsPerSample)
%CONSTRUCTVMATRIX construct spectral component matrix (V) from
%phonemeRecordings.
%   PhonemeRecordings are short recordings of phonemes to contruct V from
%   in the format of a cell array of cell arrays of waveforms, where each
%   item in the outer cell array corresponds to a phoneme.
%   PointsPerSample is the max number of points in a given recording.

% get the sizes to preallocate
numSamples = cellfun(@(x) x(2),...
    cellfun(@size,phonemeRecordings,'UniformOutput',false));
V = zeros(pointsPerSample/2+1,sum(numSamples));

% step through every phoneme recording, take the FFT, and construct a
% matrix
c=1;
for a = 1:numel(phonemeRecordings)
    for b = 1:numel(phonemeRecordings{a})
        phonemeRecordings{a}{b} = [phonemeRecordings{a}{b};...
            zeros(pointsPerSample-numel(phonemeRecordings{a}{b}),1)];
        V(:,c) = rfft(phonemeRecordings{a}{b});
        c=c+1;
    end
```

```
end
```

File: DrawPhnSamples.m

```
function [phonemeSamples,fs] = drawPhnSamples(directory,time,...
    wavSamplesPerPhn,phnList)
%DRAWPHNSAMPLES Draws a given number of randon samples of phonemes
%   Directory is the directory in which phn and wav files are kept.
%   Time is the length (in seconds) of the samples taken for each phoneme,
%   actual lengths may be shorter.
%   WavSamplesPerPhn is the number of waveform samples to take for each
%   phoneme in phnList.
%   phonemeSamples is the resultant samples drawn.
phonemeSamples = cell(numel(phnList),1);
[phnFiles,wavFiles] = getSpeakerFiles(directory);

%randomly draw a recording, open files
for k = randperm(numel(phnFiles))
    currentRecPhnFilename = phnFiles(k); %for all files, rand order
    [recordingPhns,startstops] = getPhnData([directory...
        currentRecPhnFilename.name]);
    [y,fs,wmode,fidx]=readwav([directory...
        currentRecPhnFilename.name(1:end-4) '.wav']);

    numSamples = ceil(time * fs);

    for phonemeNum = 1:numel(phnList)
        phoneme = phnList{phonemeNum};
        % find any present occurences of current phone
        phoneOccurences = find(strcmp(phoneme,recordingPhns));
        for l = randperm(numel(phoneOccurences))
            %do we need more of this phoneme?
            if numel(phonemeSamples{phonemeNum}) < wavSamplesPerPhn
```

```
                index = phoneOccurences(l); %for all occurences, rand order
                %choose a random point to sample phoneme
                firstSamp = startstops(index,1);
                lastSamp = startstops(index,2);
                if lastSamp-firstSamp > numSamples
                    samplestart = randi([firstSamp lastSamp-numSamples]);
                    samplePoints = samplestart:samplestart+numSamples-1;
                else
                    samplePoints = firstSamp:lastSamp;
                end
                phonemeSamples{phonemeNum}{end+1} = y(samplePoints);
            end
        end
    end
end
```

File: getPhnData.m

```
function [phones,startstops] = getPhnData(fname)
% GETPHNDATA get a file list of phn and wav files for a speaker.
%   directory is the directory speaker folders are located in.
%   speakerName is the id names of the speaker
%   phnFiles and wavFiles are arrays of directory structures.
currentRecPhnFID = fopen(fname);
phnData = textscan(currentRecPhnFID,'%u %u %s','delimiter','\t');
fclose(currentRecPhnFID);
phones = phnData{3};
startstops = [phnData{1}+1 phnData{2}+1];
```

File: getSpeakerFiles.m

```
function [phnFiles,wavFiles] = getSpeakerFiles(directory,speakerName)
% GETSPEAKERFILES get a file list of phn and wav files for a speaker.
%   directory is the directory speaker folders are located in.
%   speakerName is the id names of the speaker
%   phnFiles and wavFiles are arrays of directory structures.
if nargin < 2
    speakerName = [];
end
phnFiles = dir([directory speakerName '/*.phn']);
wavFiles = dir([directory speakerName '/*.wav']);
```