```
function [W, H] = nmfSS(V, K, W, MAXITER, beta, fixedInds)

% Supervised (or Semi-Supervised) NMF.
% [W, H] = nmfSS(V, K, W, MAXITER, beta, fixedInds)
%
% INPUT VARIABLES:
%   V : spectrogram of mixture
%   K : number of basis vectors
%   W : matrix of spectral basis functions. Insert empty matrix '[]'
 if
%       training or doing blind separation
%   MAXITER : number of update iterations
%   beta : divergence cost function ('EU', 'KL', or 'IS')
%   fixedInds : Once you have a trained W, use fixedInds to denote the
%               indeces that remain fixed. This can account for none
 of the
%               indeces (blind), indices for certain sources
%               (semi-supervised), or all indices (fully supervised)
%
% OUTPUT VARIABLES:
%   W : spectral basis functions for separated sources
%   H : time-activation gains for separated sources
%
% Function modified from "Single-Channel Source Separation Tutorial
% Mini-Series" by Nicholas Bryan, Dennis Sun, and Eunjoon Cho
%
% https://ccrma.stanford.edu/~njb/teaching/sstutorial/

F = size(V,1); T = size(V,2);

rng('shuffle')
if isempty(W)
    W = 1+rand(F, sum(K));
end
H = 1+rand(sum(K), T);

inds = setdiff(1:sum(K),fixedInds);
ONES = ones(F,T);

for i=1:MAXITER
    %=======================
    % I-S DIVERGENCE
    %=======================
    if strcmp(beta,'IS')

        % update activations
        H = H .* (W'*(V./((W*H+eps).^2)) ./ (W'*(ONES./(W*H))));

        % update dictionaries
        W(:,inds) = W(:,inds) .* (((V./((W*H).^2)) * H(inds,:)') ...
            ./ (ONES./(W*H) * H(inds,:)'));
```

```
    %=======================
    % KL DIVERGENCE
    %=======================
    elseif strcmp(beta,'KL')
        % update activations
        H = H .* (W'*( V./(W*H+eps))) ./ (W'*ONES);

        % update dictionaries
        W(:,inds) = W(:,inds) .* ((V./(W*H+eps))*H(inds,:)') ...
            ./ (ONES*H(inds,:)');

    %=======================
    % EUCLIDIAN DISTANCE
    %=======================
    elseif strcmp(beta,'EU')
        % update activations
        H = H .* ((W'*V) ./ (W'*W*H));

        % update dictionaries
        %W = W .* ((V*H') ./ (W*(H*H')));
        W(:,inds) = W(:,inds) .* ((V*H(inds,:)') ./ (W*H*H(inds,:)'));
    else

        % Return an error
        error('Please enter a valid beta value (EU, KL, or IS)');

    end


end

% normalize W to sum to 1
sumW = sum(W);
W = W*diag(1./sumW);
H = diag(sumW)*H;

Not enough input arguments.

Error in nmfSS (line 27)
F = size(V,1); T = size(V,2);
```

Published with MATLAB® R2019b