Dana Joaquin

ECE 3512: Signals – Continuous and Discrete

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19121

# Problem Statement

In this assignment, the concept of covariance and correlation are introduced by manipulating the audio signal sampled at 8 kHz. For the correlation part, 240 sample points of the audio signal will be used to observe and understand the concept. This will be done by taking two 240 by 1 vectors, one of them being shifted from 0 to 512 samples, creating two 240 by 512 matrices. For the covariance part, a vector containing 240 samples of the signal will be used to understand the idea of covariance.

# Approach and Results

Two 240 by 1 vectors were made: vector **x** and vector **y**. Starting at 0.9 seconds of the audio data, 240 samples were stored in **x**. A function “edit\_sig” was written to do this job, which took in the wanted time to start and created a vector that is 240 samples long. This function was iterated 512 times to make matrix **X**, which is a 240 by 512 matrix. This matrix contained the same 240 sample-long vector, 512 times. For vector **y**, it also contained 240 samples of the audio file, however, the sample point in which it starts to store the audio data, it is shifted by value k, where k = 0, 1, 2, …, 512. This is also done 512 times, creating the matrix **Y**, except that each column of sample starts at a different point of the audio file. The function that creates vector **y** and handles the shifting is the function “shift\_sig.”

Once both matrices were made, the correlation of corresponding column of data was computed using the coded function “compute\_corr.” This function does this by using the correlation equation:

Equation 1: Correlation Equation

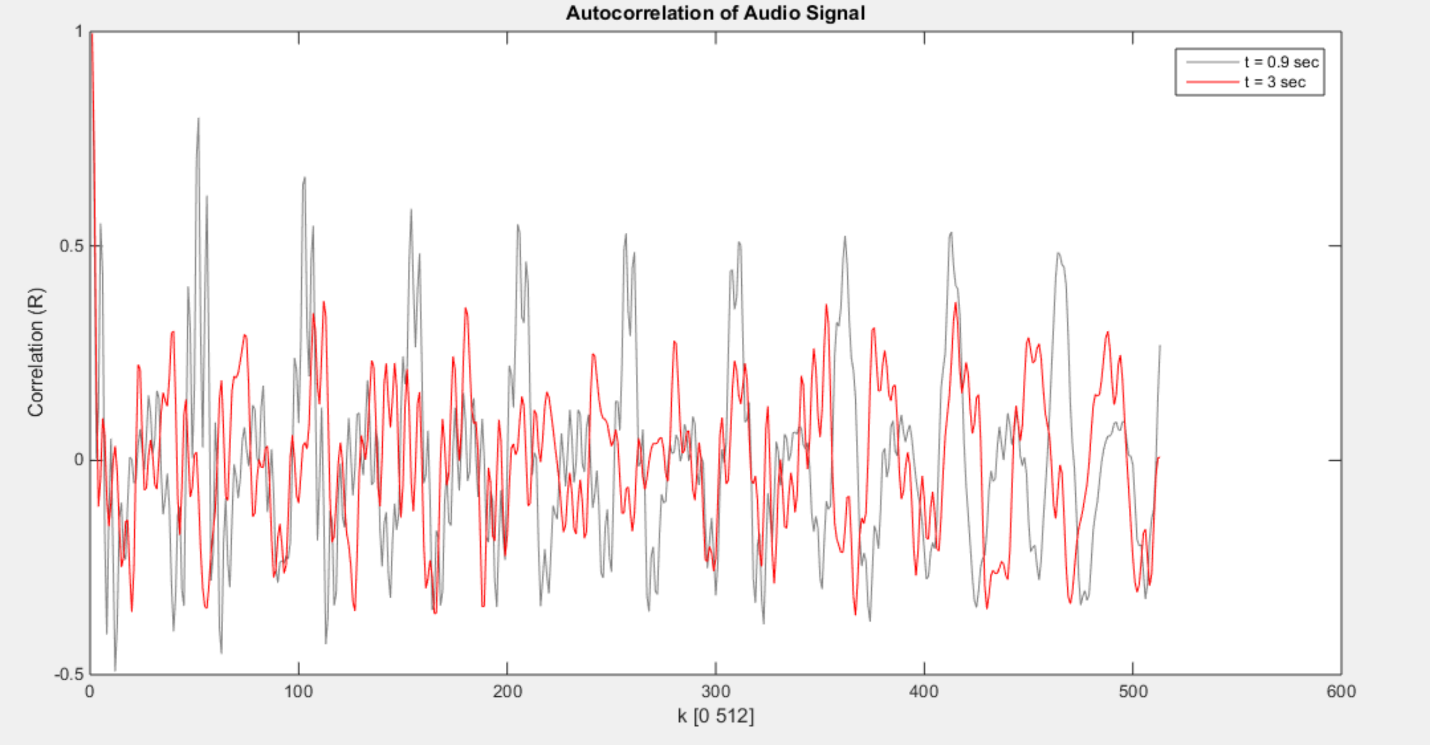


Figure 1. Autocorrelation data of the audio file for when t = 0.9 seconds and t = 3 seconds.

To compute the covariance of each column of data from matrix **X** and **Y**, the function “compute\_cov” was written to do this. “Compute\_cov” does this by using the covariance formula:

Equation 2: Covariance Equation

The mean function in MATLAB was used to compute the mean of a column in matrix **X**, in matrix **Y**, and the product of a column of data from both matrices. This iterated 512 times, creating a 512 by 1 vector containing the covariance value of each column pair. The resulting vector was returned to the “compute\_cov” function to finish calculating the correlation for each column pair of data. The variance of each column was computed using the MATLAB var function. All 512 pairs had their correlation value calculated, creating a 512 by 1 vector. This vector was returned to the main function and plotted with respect to k = 0, 1, 2, …, 512. This same procedure was done starting at t = 3 seconds of the audio signal. Figure 1 displays the results for t = 0.9 seconds and t = 3 seconds.

For the second part of the assignment, the same 240 by 1 vector **x** was used. A 16 by 16 matrix was created to store the covariance values computed using vector **x**. The function “compute\_covv2” does the calculations using the following equation:

Equation 3: Covariance Equation

Variables i and j are equal to 16, so Equation 3 is iterated 16 times to create the matrix containing the computed covariance values.

|  |
| --- |
| Covariance for t = 0.9 seconds  Columns 1 through 8  753424 824383 937657 916174 898163 705901 403403 314867  621705 753424 824383 937657 916174 898163 705901 403403  226331 621705 753424 824383 937657 916174 898163 705901  -351974 226331 621705 753424 824383 937657 916174 898163  -688107 -351974 226331 621705 753424 824383 937657 916174  -424452 -688107 -351974 226331 621705 753424 824383 937657  -133889 -424452 -688107 -351974 226331 621705 753424 824383  -551180 -133889 -424452 -688107 -351974 226331 621705 753424  -1331946 -551180 -133889 -424452 -688107 -351974 226331 621705  -1422001 -1331946 -551180 -133889 -424452 -688107 -351974 226331  -484344 -1422001 -1331946 -551180 -133889 -424452 -688107 -351974  344379 -484344 -1422001 -1331946 -551180 -133889 -424452 -688107  94612 344379 -484344 -1422001 -1331946 -551180 -133889 -424452  -547274 94612 344379 -484344 -1422001 -1331946 -551180 -133889  -310527 -547274 94612 344379 -484344 -1422001 -1331946 -551180  921165 -310527 -547274 94612 344379 -484344 -1422001 -1331946  Columns 9 through 16  436821 597184 376278 -12803 -242389 -59892 255192 341124  314867 436821 597184 376278 -12803 -242389 -59892 255192  403403 314867 436821 597184 376278 -12803 -242389 -59892  705901 403403 314867 436821 597184 376278 -12803 -242389  898163 705901 403403 314867 436821 597184 376278 -12803  916174 898163 705901 403403 314867 436821 597184 376278  937657 916174 898163 705901 403403 314867 436821 597184  824383 937657 916174 898163 705901 403403 314867 436821  753424 824383 937657 916174 898163 705901 403403 314867  621705 753424 824383 937657 916174 898163 705901 403403  226331 621705 753424 824383 937657 916174 898163 705901  -351974 226331 621705 753424 824383 937657 916174 898163  -688107 -351974 226331 621705 753424 824383 937657 916174  -424452 -688107 -351974 226331 621705 753424 824383 937657  -133889 -424452 -688107 -351974 226331 621705 753424 824383  -551180 -133889 -424452 -688107 -351974 226331 621705 753424  Covariance for t = 1.1 seconds  1.0e+04 \*  Columns 1 through 9  4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510 -1.5588 -1.7469 -5.5470  6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510 -1.5588 -1.7469  5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510 -1.5588  4.1925 5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510  2.2844 4.1925 5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781  3.6658 2.2844 4.1925 5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225  2.6714 3.6658 2.2844 4.1925 5.7674 6.6973 4.6225 2.1930 0.0269  0.7901 2.6714 3.6658 2.2844 4.1925 5.7674 6.6973 4.6225 2.1930  1.1610 0.7901 2.6714 3.6658 2.2844 4.1925 5.7674 6.6973 4.6225  1.5695 1.1610 0.7901 2.6714 3.6658 2.2844 4.1925 5.7674 6.6973  0.3279 1.5695 1.1610 0.7901 2.6714 3.6658 2.2844 4.1925 5.7674  -0.0484 0.3279 1.5695 1.1610 0.7901 2.6714 3.6658 2.2844 4.1925  0.4461 -0.0484 0.3279 1.5695 1.1610 0.7901 2.6714 3.6658 2.2844  1.0213 0.4461 -0.0484 0.3279 1.5695 1.1610 0.7901 2.6714 3.6658  -1.4351 1.0213 0.4461 -0.0484 0.3279 1.5695 1.1610 0.7901 2.6714  0.2473 -1.4351 1.0213 0.4461 -0.0484 0.3279 1.5695 1.1610 0.7901  Columns 10 through 16  -4.2839 -0.8869 0.3279 -1.1718 -2.5316 -0.8063 -0.7901  -5.5470 -4.2839 -0.8869 0.3279 -1.1718 -2.5316 -0.8063  -1.7469 -5.5470 -4.2839 -0.8869 0.3279 -1.1718 -2.5316  -1.5588 -1.7469 -5.5470 -4.2839 -0.8869 0.3279 -1.1718  -2.4510 -1.5588 -1.7469 -5.5470 -4.2839 -0.8869 0.3279  -1.4781 -2.4510 -1.5588 -1.7469 -5.5470 -4.2839 -0.8869  -0.3225 -1.4781 -2.4510 -1.5588 -1.7469 -5.5470 -4.2839  0.0269 -0.3225 -1.4781 -2.4510 -1.5588 -1.7469 -5.5470  2.1930 0.0269 -0.3225 -1.4781 -2.4510 -1.5588 -1.7469  4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510 -1.5588  6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781 -2.4510  5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225 -1.4781  4.1925 5.7674 6.6973 4.6225 2.1930 0.0269 -0.3225  2.2844 4.1925 5.7674 6.6973 4.6225 2.1930 0.0269  3.6658 2.2844 4.1925 5.7674 6.6973 4.6225 2.1930  2.6714 3.6658 2.2844 4.1925 5.7674 6.6973 4.6225  Covariance for t = 3 seconds  1.0e+05 \*  Columns 1 through 9  2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084 0.3276 -0.0399 -0.8283  2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084 0.3276 -0.0399  1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084 0.3276  0.0610 1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084  -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278  -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847  0.5705 -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131 0.3513  0.5493 0.5705 -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131  0.0286 0.5493 0.5705 -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825  0.2367 0.0286 0.5493 0.5705 -0.2230 -0.4870 0.0610 1.2768 2.1362  -0.1881 0.2367 0.0286 0.5493 0.5705 -0.2230 -0.4870 0.0610 1.2768  -0.5693 -0.1881 0.2367 0.0286 0.5493 0.5705 -0.2230 -0.4870 0.0610  -0.8146 -0.5693 -0.1881 0.2367 0.0286 0.5493 0.5705 -0.2230 -0.4870  -0.3949 -0.8146 -0.5693 -0.1881 0.2367 0.0286 0.5493 0.5705 -0.2230  0.3650 -0.3949 -0.8146 -0.5693 -0.1881 0.2367 0.0286 0.5493 0.5705  0.6141 0.3650 -0.3949 -0.8146 -0.5693 -0.1881 0.2367 0.0286 0.5493  Columns 10 through 16  -1.1485 -0.7200 -0.6751 -0.7785 -0.0822 0.6540 1.0575  -0.8283 -1.1485 -0.7200 -0.6751 -0.7785 -0.0822 0.6540  -0.0399 -0.8283 -1.1485 -0.7200 -0.6751 -0.7785 -0.0822  0.3276 -0.0399 -0.8283 -1.1485 -0.7200 -0.6751 -0.7785  0.1084 0.3276 -0.0399 -0.8283 -1.1485 -0.7200 -0.6751  -0.6278 0.1084 0.3276 -0.0399 -0.8283 -1.1485 -0.7200  -0.7847 -0.6278 0.1084 0.3276 -0.0399 -0.8283 -1.1485  0.3513 -0.7847 -0.6278 0.1084 0.3276 -0.0399 -0.8283  1.6131 0.3513 -0.7847 -0.6278 0.1084 0.3276 -0.0399  2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084 0.3276  2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278 0.1084  1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847 -0.6278  0.0610 1.2768 2.1362 2.4825 1.6131 0.3513 -0.7847  -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131 0.3513  -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825 1.6131  0.5705 -0.2230 -0.4870 0.0610 1.2768 2.1362 2.4825 |

# MATLAB Code

|  |
| --- |
| % Computer Assignment 5  % Dana Joaquin    function computer\_assignment5 = ca5\_v00  clear; clc  close all    % opening and reading audio file  filename = 'rec\_01\_speech.raw';  fileID = fopen(filename);  signaldata = fread(fileID, inf, 'int16');  fclose(fileID);    % global parameters  fs = 8000; % sampling frequency  shiftsize = 513; % k = 0,1,2,...,512  nsize = 240; % length of vector  matsize = 16; % square matrix size    % global vectors  xaxe = 1:shiftsize; % x-axis vector  x = zeros(nsize, 1); % stores signal data  y = zeros(nsize, 1); % stores shifted signal data  so\_hxc1 = zeros(shiftsize, 1); % stores corr. results when t = 0.9sec  so\_hxc2 = zeros(shiftsize, 1); % stores corr. results when t = 3sec      % ---------------------------------------------------------  % plotting audio file being used  time = 1:length(signaldata);  time = time./fs;  figure(1) % audio signal plot  plot(time, signaldata)  xlabel('Time (seconds)')  ylabel('Amplitude')  title('Audio Signal')  % ---------------------------------------------------------        % ----------------------- PART 1 --------------------------  % ---------------------------------------------------------  % starting t = 0.9 sec  starttime = 0.9; % time in audio where to start storing data  x = edit\_sig(signaldata, starttime, fs, nsize);  % making a 240 x 512 matrix of X with sample values  for i=1:shiftsize  X(:,i) = x;  end  % making a 240 x 512 matrix of Y with shifted sample values  for j=1:shiftsize  y = shift\_sig(signaldata, starttime, fs, nsize, j);  Y(:,j) = y;  end    so\_hxc1 = compute\_corr(X, Y, shiftsize);  % ---------------------------------------------------------  % ---------------------------------------------------------  % starting t = 3 sec  starttime = 3; % time in audio where to start storing data  x = edit\_sig(signaldata, starttime, fs, nsize);  % making a 240 x 512 matrix of X with sample values  for i=1:shiftsize  X(:,i) = x;  end  % making a 240 x 512 matrix of Y with shifted sample values  for j=1:shiftsize  y = shift\_sig(signaldata, starttime, fs, nsize, j); % 240 long vector  Y(:,j) = y; % stores each 240 long vector into a 240 x 512 matrix  end    so\_hxc2 = compute\_corr(X, Y, shiftsize);    % ---------------------------------------------------------  % ---------------------------------------------------------  % Plot Results Correlation Results  figure(2)  plot(xaxe, so\_hxc1)  hold on  plot(xaxe, so\_hxc2)  xlabel('k [0 512]')  ylabel('Correlation (R)')  title('Autocorrelation of Audio Signal')  legend('t = 0.9 sec', 't = 3 sec')  % ---------------------------------------------------------        % ----------------------- PART 2 --------------------------  % ---------------------------------------------------------  % starting t = 0.9 sec  starttime = 0.9; % time in audio where to start storing data  x = edit\_sig(signaldata, starttime, fs, nsize);  xcovv1 = compute\_covv2(x, matsize);  disp('Covariance for t = 0.9 seconds')  disp(xcovv1)  % ---------------------------------------------------------  % ---------------------------------------------------------  % starting t = 1.1 sec  starttime = 1.1; % time in audio where to start storing data  x = edit\_sig(signaldata, starttime, fs, nsize);  xcovv2 = compute\_covv2(x, matsize);  disp('Covariance for t = 1.1 seconds')  disp(xcovv2)  % ---------------------------------------------------------  % ---------------------------------------------------------  % starting t = 3 sec  starttime = 3; % time in audio where to start storing data  x = edit\_sig(signaldata, starttime, fs, nsize);  xcovv3 = compute\_covv2(x, matsize);  disp('Covariance for t = 3 seconds')  disp(xcovv3)  % ---------------------------------------------------------    end      % ---------------------------------------------------------  % ---------------------- FUNCTIONS ------------------------  % ---------------------------------------------------------    % function name: compute\_covv2  % coded by Dana Joaquin  % input argument(s):  % (1) vect: signal data (array)  % (2) nmatrix: size of the desired square matrix  % output argument(s):  % (1) coven: 16x16 matrix of covariance values  % objective:  % This function computes the covariance of input data array 'vect'  % and returns a n x n square matrix containing the covariance values.  % The covariance is calculated using this implemented equation:  %  % i = j = n, N = length of vect  %  % N-1  % \_\_\_\_\_  % Cov(i, j) = 1 \  % ----- / x(n-i)\*x(n-j)  % n -----  % N = 0  %  function coven = compute\_covv2(vect, nmatrix)  n = nmatrix; % desired square matrix size  datam = zeros(n, n); % initializing the n x n matrix  vl = length(vect); % length of inputted vector  % will loop through vect data and compute covar. until  % n x n matrix made  for i=0:n-1  for j=0:n-1  datam(i+1, j+1) = (1/vl).\*sum(vect(vl-i).\*vect(vl-j));  end  end    coven = datam; % transfer end result of for loop to return variable  end      % function name: compute\_cov  % coded by Dana Joaquin  % input argument(s):  % (1) mat1: inputted data matrix  % (2) mat2: inputted data matrix  % (3) n: # of columns in the inputted matrices, mat1 and mat2  % \*Note: mat1 and mat2 have to be the same size matrices  % output argument(s):  % (1) covv: vector of resulting covariance values of mat1 and mat2  % objective:  % This function computes the covariance of each corresponding columns  % in the inputted matrices, mat1 and mat2. The covariance is  % calculated by using this equation:  %  % E() = expectation/mean  %  % cov(X, Y) = E(XY) - E(X)E(Y)  %  function covv = compute\_cov(mat1, mat2, n)  for k=1:n  xmean = mean(mat1(:,k)); % mean of kth column in mat1  ymean = mean(mat2(:,k)); % mean of kth column in mat2  xymean = mean(mat1(:,k).\*mat2(:,k)); % mean of kth columns of mat1 and mat2  covk = xymean - xmean\*ymean; % compute covariance of kth iteration  covarray(k) = covk; % store each kth result in vector  end    covv = covarray; % finished result stored in returning variable  end      % function name: compute\_corr  % coded by Dana Joaquin  % input argument(s):  % (1) mat1: inputted data matrix  % (2) mat2: inputted data matrix  % (3) n: # of columns in the inputted matrices, mat1 and mat2  % \*Note: mat1 and mat2 have to be the same size matrices  % output argument(s):  % (1) hxc: vector of resulting correlation values of mat1 and mat2  % objective:  % This function computes the correlation of each corresponding columns  % in the inputted matrices, mat1 and mat2. The correlation is  % calculated by using this equation:  %  % cov(,) = covariance  % var() = variance  %  % cov(X,Y)  % corr(X,Y) = ---------------  % \_\_\_\_\_\_\_\_\_\_\_\_\_  % -/var(X)\*var(Y)  %  function hxc = compute\_corr(mat1, mat2, n)  covvect = compute\_cov(mat1, mat2, n); % cov. vector of each nth column  for k=1:n % gets var of each kth data vector in mat 1 and mat2  xvar = var(mat1(:,k)); % computes variance value of kth column of data  varX(k) = xvar; % variance vector of mat1  yvar = var(mat2(:,k)); % computes variance value of kth column of data  varY(k) = yvar; % variance vector of mat2  end  num = covvect;  den = sqrt(varX.\*varY);  hxc = num./den; % calculates correlation and stores in returning variable  end      % function name: edit\_sig  % coded by Dana Joaquin  % input argument(s):  % (1) sig: signal data  % (2) timestart: time where want to start storing data from sig  % (3) fsample: sampling frequency of sig  % (4) arraysize: desired arraysize of sig data  % output argument(s):  % (1) alteredsig: vector thats arraysize long containing sig data  % starting at t = timestart of sig  % objective:  % This function stores a certain amount of data from inputted signal  % sig in a vector that is arraysize long. The time to start storing  % sig data is multiplying the value of timestart with the fsample.  %  function alteredsig = edit\_sig(sig, timestart, fsample, arraysize)  nstart = timestart\*fsample; % start time in sig  nend = nstart+arraysize; % end time in sig  alteredsig = sig(nstart:nend-1); % return wanted vector of sig data  end      % function name: shift\_sig  % coded by Dana Joaquin  % input argument(s):  % (1) sig: signal data  % (2) timestart: time where want to start storing data from sig  % (3) fsample: sampling frequency of sig  % (4) arraysize: desired arraysize of sig data  % (5) shiftn: the amount of times want to shift in sig  % output argument(s):  % (1) alteredsig: vector that’s arraysize long containing sig data  % starting at t = timestart of sig shifted by shiftn  % objective:  % This function stores a certain amount of data from inputted signal  % sig in a vector that is arraysize long. The time to start storing  % sig data is multiplying the value of timestart with the fsample,  % then added by the shift value, shiftn.  %  function shiftdat = shift\_sig(sig, timestart, fsample, arraysize, shiftn)  shiftn = shiftn-1; % want to start with shift = 0  nstart = timestart\*fsample+shiftn; % start time of sig  nend = nstart+arraysize; % end time of sig  shiftdat = sig(nstart:nend-1); % return wanted vector of sig data with shift  end |

# Conclusions

Computing the correlation of a signal with itself is known as autocorrelation, which is what Figure 1 displays. An exponential decay is seen in the graph as the number of sample shifts increases, decaying to the value of 0. This shows as time progresses, specifically samples, the correlation value of a signal with itself starts to become uncorrelated and having no relation at all. Autocorrelation helps detect repeating patterns in a signal and for the audio signal in this assignment, it has a sinusoidal pattern.

Covariance is another way to see how two sets of data are related with each other. It tells the same type of information that correlation does. If the covariance value is positive, that means the data are directly related, therefore as one increases, the other does as well. For a negative value, it shows the data are inversely related. The magnitude of the covariance also shows how much they are related, so the larger the value, the more closely related they are. In the matrix, the diagonal contains the highest values because that is when the same element of data is being compared to itself.