Prof. Dr. Hauke Schramm
Department of Computer Science
Kiel University

Pattern Recognition

Solution to Exercises

Practice Sheet 1

Exercise L-1.1 (Octave basics, random variables and statistics)

a. Generate two N x N matrices `A` and `B` containing integer random values (set `N = 100`). The numbers in `A` must be uniformly distributed in the interval [0,10) and in `B` normally distributed with mean 0 and standard deviation $\sigma = 10$. Utilize the Octave functions `rand`, `randn`, `fix` and `round`. For the generation of `A` you need `fix` and for `B` `round`. Why?

b. Determine the minimum and maximum element of `A` and `B` (functions min and max). Store the results in `amin`, `amax`, `bmin` and `bmax`.

c. Determine the frequency of each matrix element in `A` (respectively `B`) and store it as follows in vector `a` (resp. `b`): The first element of `a` (`b`) contains the frequency of the smallest element in `A` (`B`) and the last element of `a` (`b`) contains the frequency of the largest element in `A` (`B`).

   Hints:

   i. For this task you may use the functions `find` and `length` and work with a `for`-loop.
   ii. `A == i` returns a N x N logical matrix with values 1 only at elements in `A` which equal `i`.
   iii. You may append a number `x` as follows to a vector `a`: `a = [a  x];`

d. Draw the frequency distributions in vector `a` and `b` into two separate figures (Figure 1 and Figure 2) using the functions `bar` and `figure`. Label them appropriately. Note that this graphical representation of the distribution of data is called a ***histogram***.

   Useful commands: `title`, `xlabel`, `ylabel`, `axis`

e. Since the data in matrix `A` is uniformly distributed, all components of vector `a` should have similar values which must be evenly distributed around the mean frequency value. Illustrate this fact by plotting the mean of all components in `a` as horizontal line into Figure 1. Use the command "`hold on`" to retain the current figure when plotting a new object.

   Useful commands: `line, hold on`

f. Generate probability distributions from the histograms using relative frequencies and plot them into Figure 3 (vector `a`) and 4 (vector `b`). Show that the probabilities in each distribution sum up to 1.

g. The data in matrix `B` is normally distributed with mean value $\mu = 0$ and standard deviation $\sigma = 10$. Its probability in Figure 3 can be described by the following function:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

   Draw this function as red line into Figure 4, using for the abscissa a stepsize of 0.1 between the end points bmin and bmax.

h. Check the following statements in your program:

   a. 68.3% of all elements in B lie in the interval [$-\sigma$, σ]
   b. 95.5% of all elements in B lie in the interval [-2σ, 2σ]
   c. 99.7% of all elements in B lie in the interval [-3σ, 3σ].

Prof. Dr. Hauke Schramm
Department of Computer Science
Kiel University

```matlab
% Solution to Practical Exercise 1

% Part (a): Generating Matrices A and B
% Matrix A: For a uniform distribution of integer values in the
% interval [0,10), we generate random values in [0,10) using rand, then
% apply fix to round down each element to the nearest integer. We use fix
% here because it always rounds toward zero.

% Matrix B: For a normal distribution of integer values with mean 0 and
% standard deviation 10, we use randn to generate normally distributed
% values and then scale it by 10. We use round to round these values to
% the nearest integer.

% Set matrix size
N = 100;

% Generate matrix A with uniform distribution in the interval [0,10)
A = fix(10 * rand(N));

% Generate matrix B with normal distribution (mean 0, std 10)
B = round(10 * randn(N));

% Part (b): Determine the minimum and maximum elements
% Next, we calculate the minimum and maximum values of A and B using the
% min and max functions, storing them in amin, amax, bmin, and bmax.
% Find minimum and maximum elements in A
amin = min(A(:));
amax = max(A(:));

% Find minimum and maximum elements in B
bmin = min(B(:));
bmax = max(B(:));

% Part (c): Determine frequency of each matrix element

% Initialize frequency vector 'a' as an empty array
a = [];
for i=amin:amax
  % Find all occurrences of the current value 'i' in matrix 'A'
  % 'find(A == i)' returns indices of elements in 'A' that are equal to 'i'
  % 'length(find(A == i))' gives the count of occurrences of 'i' in 'A'
  % Append this count to the frequency vector 'a'
  a = [a length(find (A==i))];
end

b = [];
for i=bmin:bmax
  % see comment above
  b = [b length(find (B==i))];
end

figure(1);
bar(amin:amax, a);

% draw the mean frequency line
hold on;
line([min(A(:))-0.5 max(A(:))+0.5],[mean(a(:)) mean(a(:))], 'Color','r');
```

```
% axis scaling
axis([min(A(:)) - 1, max(A(:)) + 1, 0, max(a) * 1.1]);

figure(2);
bar(bmin:bmax, b);

% axis scaling
axis([min(B(:))-1, max(B(:))+1, 0, max(b) * 1.1]);

% convert histogram into probability distribution
a_norm = a./N^2;
b_norm = b./N^2;

figure(3);
bar(amin:amax, a_norm);

% axis scaling
axis([min(A(:))-1 max(A(:))+1, 0, max(a_norm) * 1.1]);

figure(4);
bar(bmin:bmax, b_norm);

% axis scaling
axis([min(B(:))-1 max(B(:))+1, 0, max(b_norm) * 1.1]);

% check if probabilities sum up to 1
a_sum = sum(a_norm);
b_sum = sum(b_norm);

fprintf("Probability mass of frequency of elements in A: %.2f\n", a_sum);
fprintf("Probability mass of frequency of elements in B: %.2f\n", b_sum);

% add normal distribution function to Figure 4
t = bmin:0.1:bmax;

% determine mean and variance
m = mean(B(:));
s = std(B(:));

% equation of normal function with mean m and standard deviation s
f = exp(-0.5*((t-m)./s).^2)./(sqrt(2*pi)*s);

hold on;
plot(t, f, 'r');

% determine percentage of probability mass lying in the range mean +/-
standard deviation
% get the index of the element positioned at mean = 0 (note: indexing
starts at 1)
zero_index = -bmin+1

% sum up elements around mean (+/- sigma)
sum_plusminus_s = sum(b_norm(zero_index-10:zero_index+10))

% sum up elements around mean (+/- 2*sigma)
sum_plusminus_2s = sum(b_norm(zero_index-20:zero_index+20))
% sum up elements around mean (+/- 3*sigma)
sum_plusminus_3s = sum(b_norm(zero_index-30:zero_index+30))
```

Prof. Dr. Hauke Schramm
Department of Computer Science
Kiel University

Exercise L-2.1 (Octave basics, joint and marginal probabilities)

Suppose we consider two random variables:

- x: Weather condition, where x=1 represents sunny weather and x=2 represents rainy weather.
- y: Driving behavior, where y=1 represents cautious driving and y=2 represents fast driving.

The joint probabilities are defined as follows:

- P(x=1,y=1)=0.2: The probability that the weather is sunny and driving is cautious.
- P(x=1,y=2)=0.3: The probability that the weather is sunny and driving is fast.
- P(x=2,y=1)=0.4: The probability that the weather is rainy and driving is cautious.
- P(x=2,y=2)=0.1: The probability that the weather is rainy and driving is fast.

For all other combinations, the probability P(x,y)=0.

This joint distribution can be represented as a matrix:

	$Y=1$	$Y=2$
$X=1$	0.2	0.3
$X=2$	0.4	0.1

Tasks

1. Calculate the marginal probabilities P(x=1) and P(x=2).
2. Calculate the marginal probabilities P(y=1) and P(y=2).
3. Verify if the sum of all joint probabilities is equal to 1.

Solve this using an Octave script.

Solution

```
% Solution to Practical Sheet 1, Exercise 2
% Define the joint probabilities as a matrix
P_XY = [0.2, 0.3; 0.4, 0.1];

% Calculate marginal probabilities for X
P_X1 = sum(P_XY(1, :));  % P(X=1)
P_X2 = sum(P_XY(2, :));  % P(X=2)

% Calculate marginal probabilities for Y
P_Y1 = sum(P_XY(:, 1));  % P(Y=1)
P_Y2 = sum(P_XY(:, 2));  % P(Y=2)

% Sum of joint probabilities
P_sum = sum(P_XY(:));

% Display results
fprintf("Marginal probability P(X=1): %.2f\n", P_X1);
fprintf("Marginal probability P(X=2): %.2f\n", P_X2);
fprintf("Marginal probability P(Y=1): %.2f\n", P_Y1);
fprintf("Marginal probability P(Y=2): %.2f\n", P_Y2);
fprintf("Sum of joint probabilities: %.2f\n", P_sum);
```