ONE-STEP ERROR PROBABILITY

Table of Contents

Initial setup

```
clear, clc
N = 120;
number_of_trials = 1e5;
p_array = [12,24,48,70,100,120];
```

With zero diagonals

```
error_probability_array_zero_diag = zeros(1,size(p_array,2));
for i = 1:size(p_array,2)
    p = p_array(i);
    errors = 0;
    for trial = 1:number_of_trials
        patterns = generate_patterns(p, N);
        weight_matrix = generate_weight_matrix_zero_diag(patterns);
        pattern_number = randi([1 p]);
        neuron_number = randi([1 N]);
        old_neuron_value = patterns(neuron_number,pattern_number);
        new_neuron_value =
 signum(weight_matrix(neuron_number,:)*patterns(:,pattern_number));
        errors = errors + double(old_neuron_value ~=
 new_neuron_value);
    end
    error_probability_array_zero_diag(i) = errors / number_of_trials;
end
disp("The one step error probabilities with zero diagonals are:")
disp(num2str(error_probability_array_zero_diag))
```

```
The one step error probabilities with zero diagonals are:
0.00035     0.01147     0.05612     0.09402     0.13625     0.15764
```

Without zero diagonals

```
error_probability_array = zeros(1,size(p_array,2));
for i = 1:size(p_array,2)
    p = p_array(i);
    errors = 0;
    for trial = 1:number_of_trials
```

```
        patterns = generate_patterns(p, N);
        weight_matrix = generate_weight_matrix(patterns);
        pattern_number = randi([1 p]);
        neuron_number = randi([1 N]);
        old_neuron_value = patterns(neuron_number,pattern_number);
        new_neuron_value =
 signum(weight_matrix(neuron_number,:)*patterns(:,pattern_number));
        errors = errors + double(old_neuron_value ~=
 new_neuron_value);
    end
    error_probability_array(i) = errors / number_of_trials;
end
disp("The one step error probabilities without zero diagonals are:")
disp(num2str(error_probability_array))
```

```
The one step error probabilities without zero diagonals are:
0.00022     0.00304     0.01242      0.019     0.02119     0.02308
```

Functions

```
disp('')
function weight_matrix = generate_weight_matrix(patterns)
    weight_matrix = patterns*patterns';
end

function weight_matrix = generate_weight_matrix_zero_diag(patterns)
    weight_matrix = patterns*patterns'/size(patterns,1);
    weight_matrix = weight_matrix - diag(diag(weight_matrix));
end

function sign = signum(x)
    sign = 2*(x >= 0) - 1;
end

function patterns = generate_patterns(p, N)
    % Generates a p by N matrix with p N-bit patterns as columns
    patterns = 2.*randi([0 1],N, p) - 1;
end
```

Published with MATLAB® R2020a