

M-Ary Shanon Fano Encoding

Anubhav Rathore

Contents

- [Defaults](#)
- [Inputs](#)
- [M-ary Shanon-Fano Algorithm](#)
- [Results Viewing](#)

Defaults

```
clear; clc;
```

Inputs

```
%probabs = input("Enter probilites of symbols: ");
probabs = [0.1 0.2 0.1 0.3 0.1 0.1 0.1];
M = 4;
%M = input("Enter M = ");
N = length(probabs);

% Sort probabilities
[SortedProbabs, order] = sort(probabs, 'descend');

% Initialize codes
codes = repmat({''}, 1, N);
```

M-ary Shanon-Fano Algorithm

```
function codes = ShannonFano_Mary(probs, idx, codes, M)

n = length(idx);

% Base case: single symbol
if n == 1
    return;
end

% Base case: n <= M → assign digits and stop
if n <= M
    for i = 1:n
        codes{idx(i)} = [codes{idx(i)} num2str(i-1)];
    end
    return;
end

% ----- SPLITTING -----
totalProb = sum(probs);
target = totalProb / M;
```

```

groups = cell(1, M);
groupProbs = cell(1, M);

g = 1;
acc = 0;
startIdx = 1;

for i = 1:length(probs)
    acc = acc + probs(i);

    if acc >= target && g < M
        groups{g} = idx(startIdx:i);
        groupProbs{g} = probs(startIdx:i);
        startIdx = i + 1;
        acc = 0;
        g = g + 1;
    end
end

% Last group takes remaining symbols
groups{g} = idx(startIdx:end);
groupProbs{g} = probs(startIdx:end);

% Append Digits
for g = 1:length(groups)
    for i = groups{g}
        codes{i} = [codes{i} num2str(g-1)];
    end
end

% Recursion
for g = 1:length(groups)
    codes = ShannonFano_Mary(groupProbs{g}, groups{g}, codes, M);
end
end

```

Probabilities:

```
0.1000    0.2000    0.1000    0.3000    0.1000    0.1000    0.1000
```

Shannon–Fano Codes:

```
{'11'}    {'10'}    {'20'}    {'0'}     {'21'}    {'22'}    {'3'}
```

Results Viewing

```

codes = ShannonFano_Mary(SortedProbabs, order, codes, M);

% Display
disp('Probabilities:');
disp(probabs);
disp('Shannon–Fano Codes:');
disp(codes);

```

