BUBBLE ENTROPY

Presentation for the Biomedical Signal Processing course
in the Master degree in Computer Science
Università degli Studi di Milano – AA 2024/2025

by Andrea Fumagalli – 46930A

What is entropy?

- In time series analysis is a measure of complexity/regularity
- For a discrete random variable X

$$H(X) = -\sum_{x \in \mathcal{X}} p(x) \log p(x)$$

- Entropy rate

$$H(\mathcal{X}) = \lim_{n \to \infty} \frac{1}{n} H(X_1, X_2, \ldots, X_n)$$

Common definitions

- Sample Entropy and Approximate Entropy
  - *Embed the time series in m dimension*
  - *Calculate the distance between each pair of vectors*
  - *Count the number of vectors that differ for less than r*
  - *Repeat the process for m+1*
  - *Calculate entropy according to each method*
- They both depend heavily on m, r and N

Alternative solutions

- Permutation entropy
  - *We get rid of r by ordering each vector X_i and the building another vector J_i composed by the positions of the elements of X_i before ordering*

    Ex: $X_i = (6, 2, 1, 4)$ -> $J_i \text{ is } (3, 2, 4, 1)$
  - *We then compute the probability of each pattern ad calculate the entropy as*

$$peEn = H = -\sum_{i=1}^{n} p(J_i) \log(p(J_i))$$

- Rènyi permutation entropy
  - *Instead of Shannon Entropy we use Rènyi entropy of order 2*

$$H_2(X) = -\log \sum_{i=1}^{n} p_i^2$$

- Conditional Rènyi permutation Entropy
  - *We use a conditional measure to have a more solid and descriptive metric*

$$cRpeN = (H_2^{m+1} - H_2^{m})/\log(m+1)$$

Bubble Entropy

- It is based on Conditional Rènyi Permutation Entropy, but differently from before the measure is applied on the number of swaps necessary for *bubbleSort* to order each vector.
  - *Sort each vector X_i of m elements in ascending order and count the number of swaps n_i necessary*
  - *Make an histogram of n_i and normalize by N – m + 1 to obtain p_i*
  - *Compute H using Rènyi entropy of order 2*
  - *Repeat for m + 1*
  - *Compute bEn as:* $bEn = (H^{m+1}_{swaps} - H^{m}_{swaps}) / \log\left({}^{m+1}/_{m-1}\right)$
- Complexity of *bubbleSort is $O(n^2)$, but we are ordering partially ordered vectors, so in this case the complexity is O(m). In conclusion the total complexity of Bubble Entropy is O(Nm)*

Implementation

- All the implementation has been done in MATLAB R2024b

```matlab
function out = bubbleEntropy(serie, m)
    p_m = swap_freq(serie,m);
    H_m = -log(sum(p_m.^2));

    p_m1 = swap_freq(serie,m + 1);
    H_m1 = -log(sum(p_m1.^2));

    out = (H_m1 - H_m)/log((m+1)/(m-1));
return
```

```matlab
function out = embed(serie, dim)
    out = zeros(dim, size(serie,1) - dim + 1);
    for i=1:dim
        out(i,:) = serie(i:end-(dim-i))';
    end
return
```

```matlab
function out = swap_freq(serie,m)
    embedded = embed(serie,m);
    nSwaps = count_swaps(embedded);
    uv = unique(nSwaps);
    hist = histc(nSwaps,uv);
    out = hist/(size(serie,1)-m+1);
return
```

Implementation

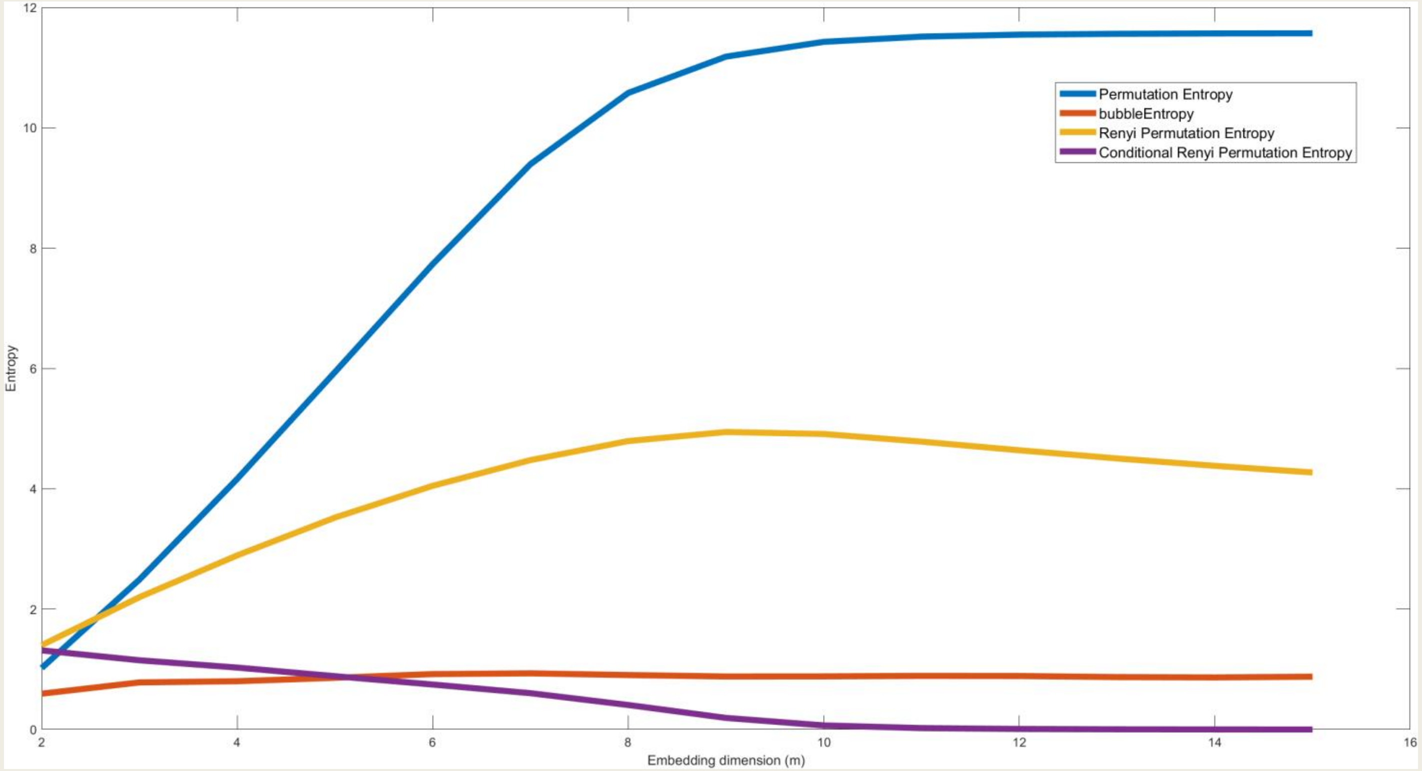
```
function out = count_swaps(serie)
    m = size(serie,1);
    x = serie(:,1)';
    [sorted, nswaps] = bubbleSort(x);
    out = [nswaps];
    for i=2:size(serie,2)
        index = find(sorted==serie(1,i-1));
        index = index(1);
        nswaps = nswaps - index + 1;
        x = swap(sorted,index);
        x = [x(2:end), serie(m,i)];
        [sorted, new_nswaps] = quicklySort(x);
        nswaps = nswaps + new_nswaps;
        out = [out, nswaps];
    end
return
```

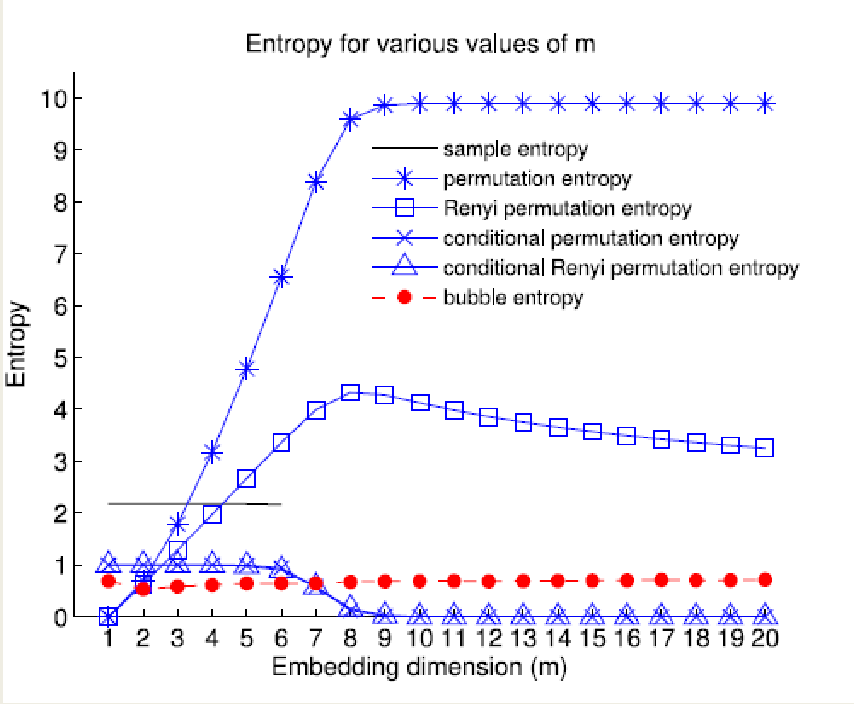
```
function [ordered,swaps] = quicklySort(a)
    ordered = a;
    swaps = 0;
    m = size(a,2);
    for i=m:-1:2
        if ordered(i)<ordered(i-1)
            temp = ordered(i);
            ordered(i) = ordered(i-1);
            ordered(i-1) = temp;
            swaps = swaps + 1;
        else
            break
        end
    end
return;
```

```
function [ordered,swaps] = bubbleSort(a)
    ordered = a;
    swaps = 0;
    swapped = true;
    n = size(a,2) - 1;
    while(swapped)
        swapped = false;
        for i=1:n
            if ordered(i)>ordered(i+1)
                temp = ordered(i);
                ordered(i) = ordered(i+1);
                ordered(i+1) = temp;
                swaps = swaps + 1;
                swapped = true;
            end
        end
        n = n-1;
    end
return;
```

Results

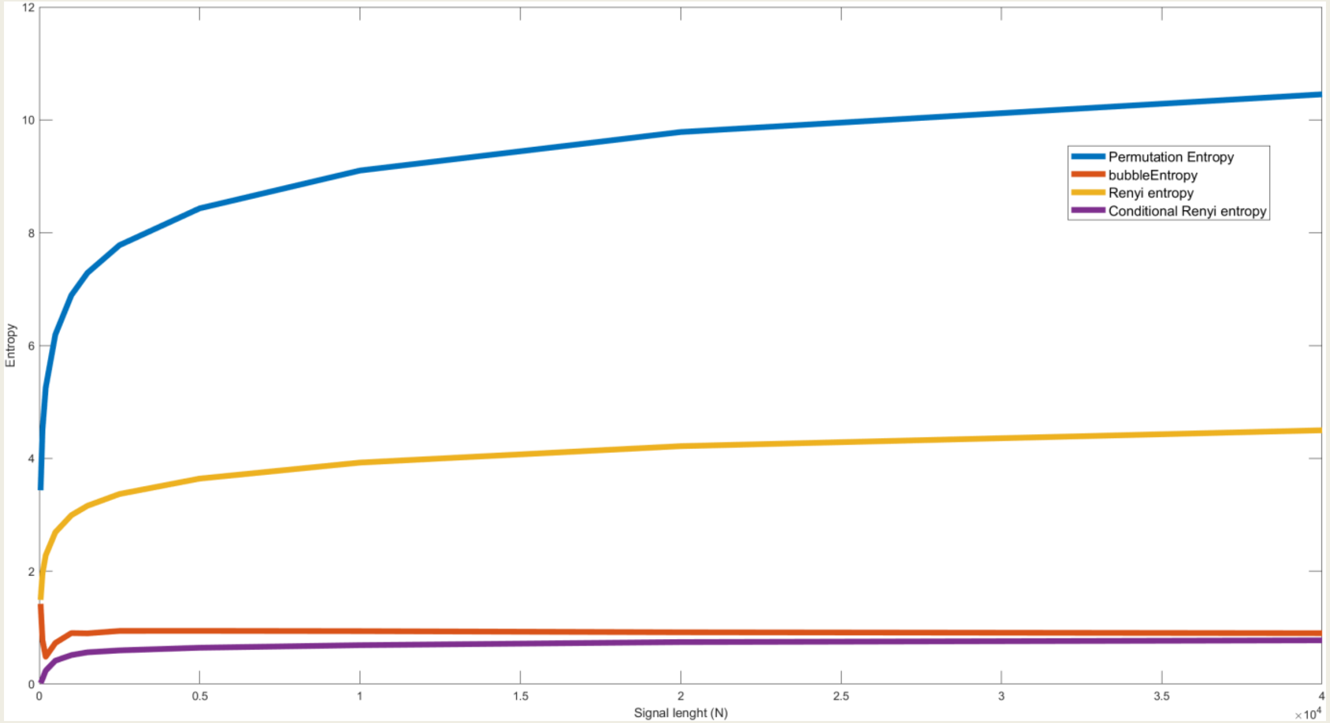
- The experiments have been done on two datasets from Physionet : *Congestive Heart Failure RR Interval Database (chf)* and *Normal Sinus Rhythm RR Interval Database (nsr)*
- *Dependance from m*

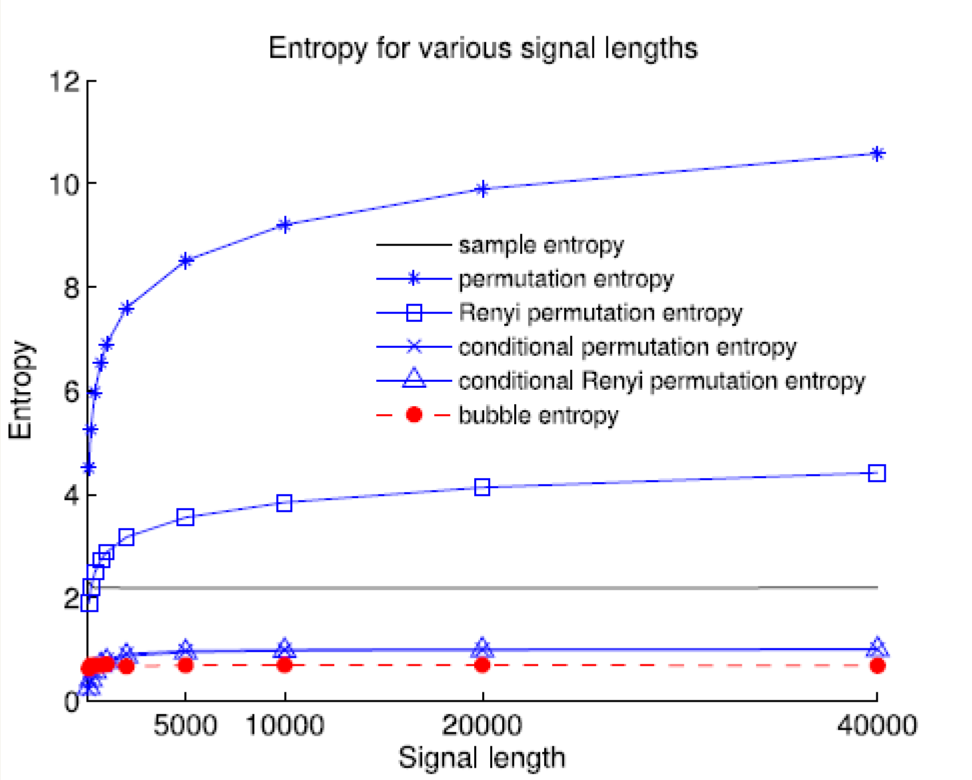




Results

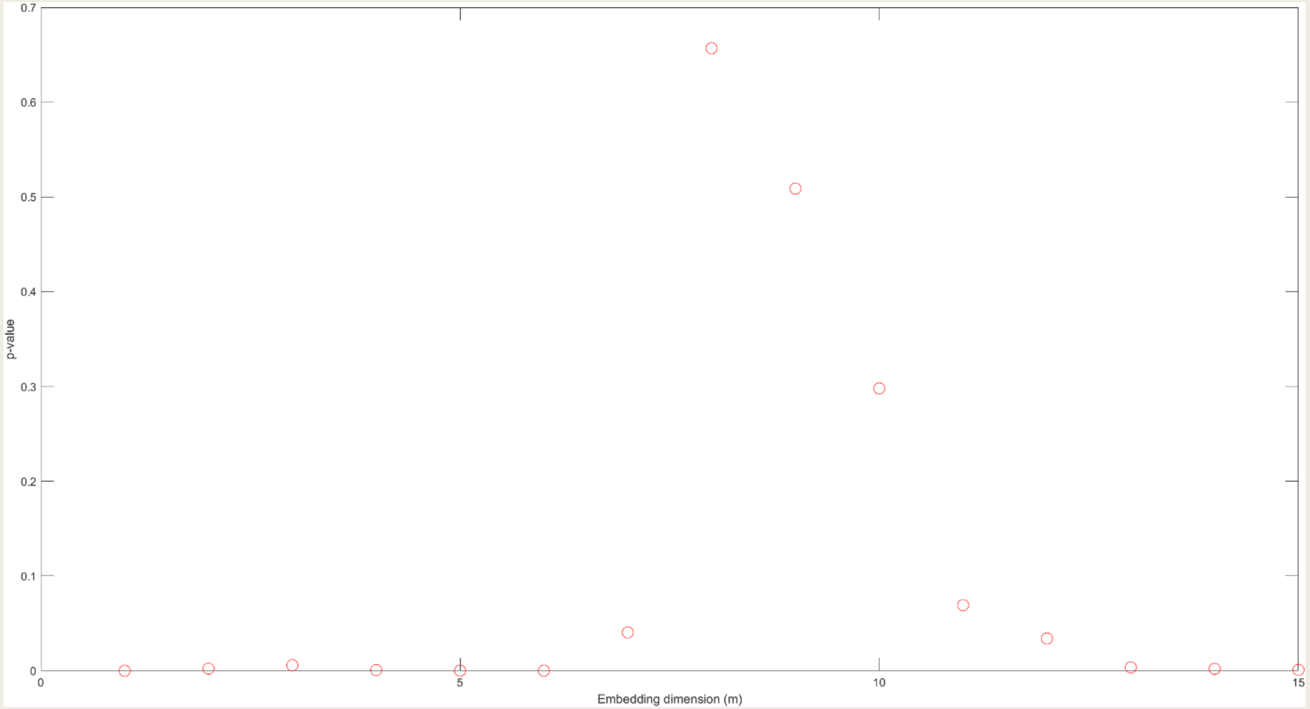
- Dependance from N

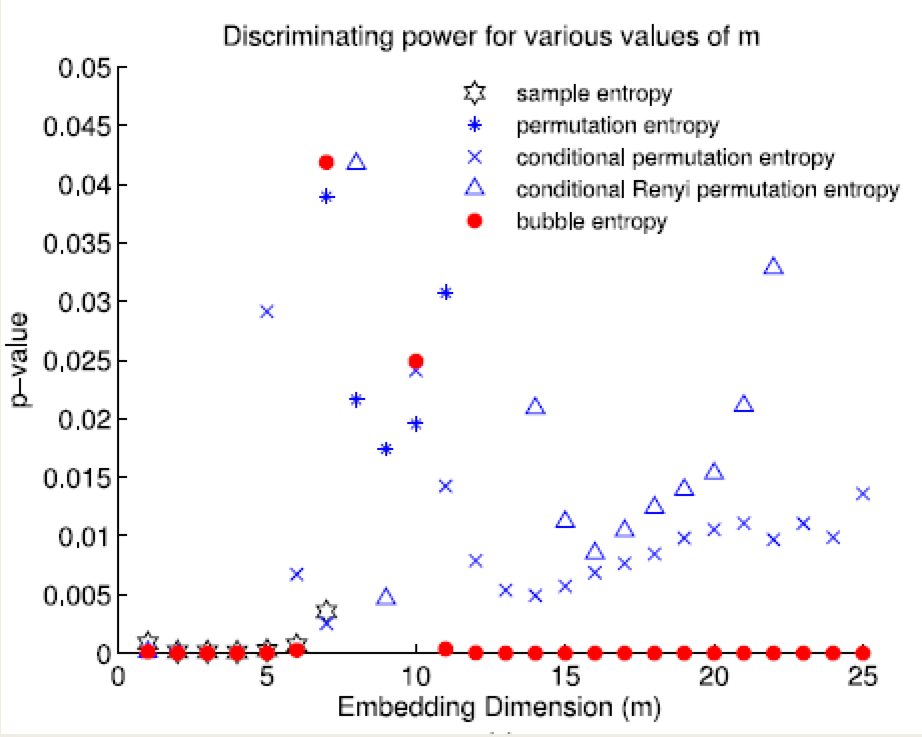




Results

- Discriminating power





Conclusions

- Bubble Entropy is a definition of entropy in which the parameter r isn't necessary
- It is almost completetly independent from the embedding dimension m and the lenght N of the time serie
- It has a complexity of order O(Nm), which is quite reasonable
- It has a strong discriminating power