Primeira entrega de projeto

Karen Kaori Yonea - 10349471

1 Projeto 1

1.1 Geração de padrões 1D

Para realização do projeto foi utilizado o script a seguir responsável pela geração de um padrão 1D de tamanho n à partir de um autômato probabilístico com um vetor(nos) correspondente aos seus nós e uma matriz de transição(mt) característica:

```
generate_pattern <- function(nos, mt, n, e) {
  #Adicionar estado inicial ao vetor
  ve <-matrix(c(0,e), ncol = 2)
  #Loop para gerar o padrao
  for (i in 1:(n-1)) {
    #Gerar numero aleatorio entre 0 e 1
    x <-runif(1, 0, 1)
    #Contador que percorre as linhas do estado
        atual
    j = 1
    #Probabilidade de ir para primeiro nó
    p = mt[nos[j],e]
    #Teste da probabilidade
    while (x > p) {
      j = j + 1
      p = p + mt[nos[j],e]
    }
    #Atualizar o nó atual e adiciona ao vetor
    e = nos[j]
    ve <-rbind(ve, c(i, e))
  }
  #Retorna vetor de estados
  return(ve)
}
```

1.2 Parte A

1.2.1 Autômatos Probabilísticos

Nessa parte do projeto foram utilizados 3 autômatos probabilísticos com 2 nós.

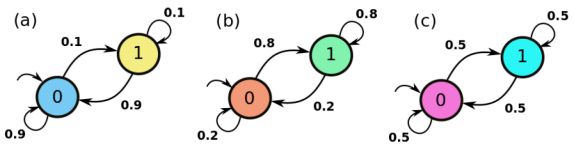

Figura 1: Figura retirada do CDT-22

1.2.2 Plots

A seguir temos três formas de apresentar os mesmos padrões gerados com 200 iterações cada pelos três autômatos:

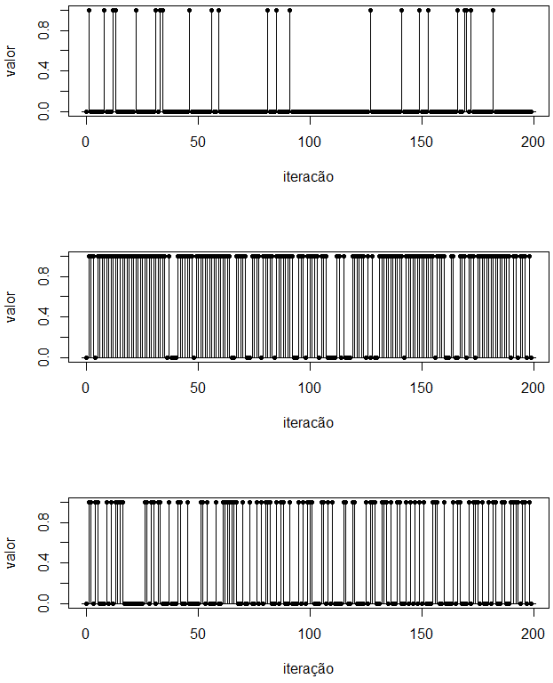

Figura 2: StemPlot dos padrões

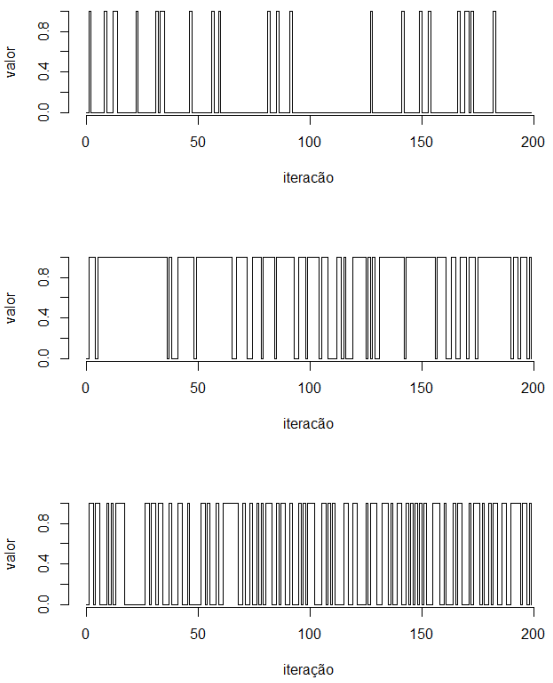

Figura 3: SquarePlot dos padrões

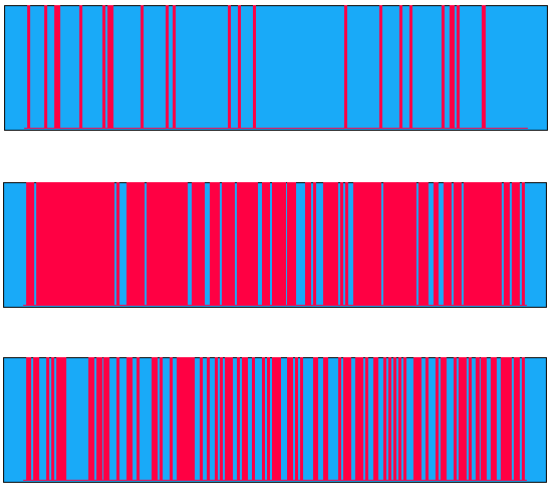

Figura 4: BarPlot dos padrões

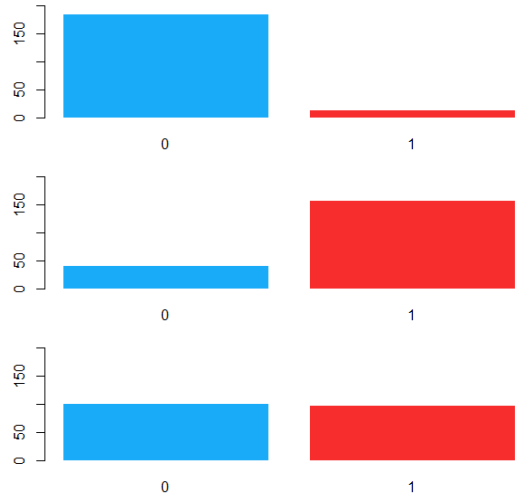

Figura 5: BarPlot da frequência dos valores dos autômatos

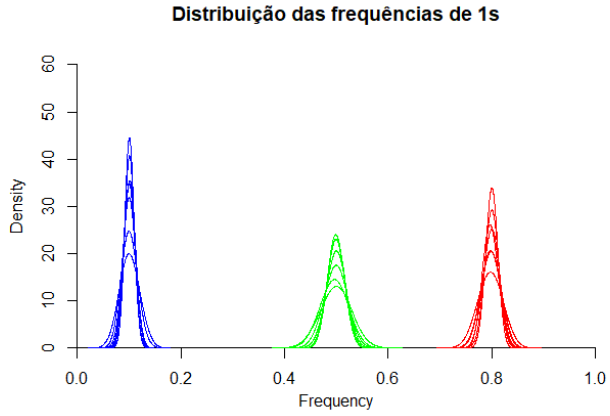

Figura 6: Densidade de 1s obtida com 200 execuções dos autômatos para diferentes valores de $M = 500, 750, ..., 2000$

1.3 Parte B

1.3.1 Autômatos Probabilísticos

Nessa parte do projeto foram utilizados 2 autômatos probabilísticos com 6 nós

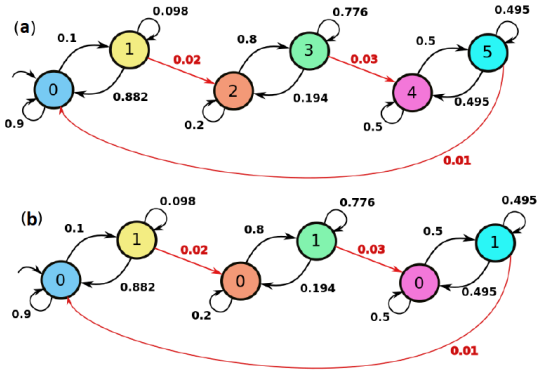

Figura 7: Figura retirada do CDT-22

1.3.2 Plots

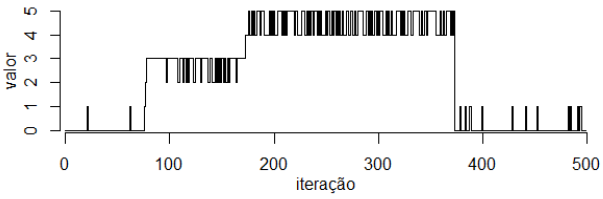

Figura 8: Square plot do padrão gerado pelo autômato 7.a

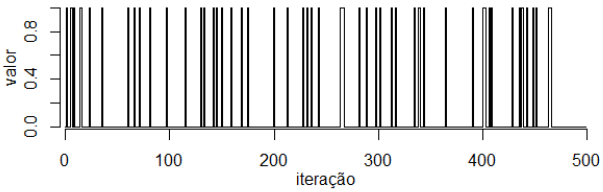

Figura 9: Square plot do padrão gerado pelo autômato 7.b

2 Projeto 2

Autômato utilizado na obtenção dos resultados:

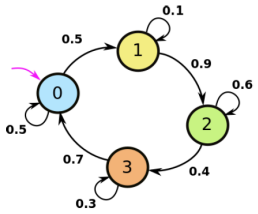

Figura 10: Figura retirada do CDT-23

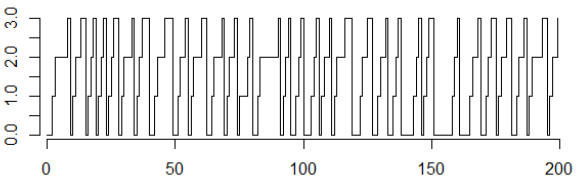

Figura 11: Square plot do padrão gerado

2.1 Split Signal

```r
split_signal <- function(L, S) {
  #Li é uma matrix cujas linhas são os splits
       para cada nó do autômato
  #Gera a matrix com zeros
  Li <-matrix(0, nrow=lenght(S), ncol=dim(L)[1])
  rownames(Li) <-S
  #Percorre o padrão
  for (i in 1:dim(L)[1]) {
    #Adiciona 1 no split correspondente ao sinal
    Li[L[i, 2], i] = 1
  }
  return(Li)
}
```

Separação do sinal por símbolos:

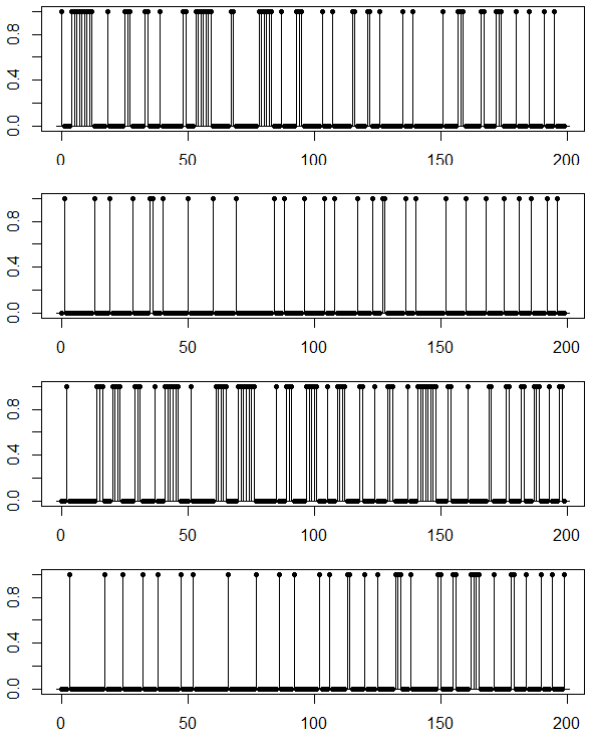

Figura 12: StemPlot dos splits

2.2 Bursts

```r
scan_bursts <- function(L, n) {
  #Inicializa o vetor de bursts
  Ls <-vector()
  for (i in 1:n) {
    #Encontra a ocorrência do sinal
    if (L[i] == 1) {
      i0 = 1
      #Incrementa enquanto há ocorrência do sinal
      while ((L[i] == 1) & (i < n)) {
        i = i + 1
      }
      #Caso do último ser o sinal
      if ((i == n) & (L[i] == 1)) {
        i = n + 1
      }
      bs = i - i0
      if (bs > 0) {
        Ls <- append(Ls,bs)
      }
    }
    i = i + 1
  }
  return(Ls)
}
```

Essa função recebe um vetor correspondente a um dos splits do sinal gerado pelo autômato e retorna as posições em que há ocorrência do valor 1, se há várias ocorrências juntas, a posição inicial da ocorrência se repete, sendo assim, a quantidade de bursts corresponde a quantidade de posições distintas no vetor, o tamanho do burst corresponde a quantidade de vezes que o mesmo valor aparece.

Sinal	Número de bursts	Tamanho dos bursts			
		Média	Desvio	Entropia	Eveness
0	31	2.26	1.57	2.16	4.48
1	30	1.10	0.31	1.23	2.35
2	30	2.03	1.47	1.92	3.79
3	30	1.20	0.41	1.32	2.49

Logicamente, um nó cuja probabilidade de se manter nele mesmo é pequena, possui média do tamanho dos bursts do símbolos menor.

2.3 Distância entre símbolos

```r
scan_intersymbols <- function(L, n) {
  #Inicializa o vetor de distâncias
  Ls <-vector()
  for (i in 1:n) {
    #Encontra a ocorrência do sinal
    if (L[i] == 1) {
      bs = 0
      #Incrementa enquanto não encontra outra
           ocorrência
      while ((L[i] == 0) & (i < n)) {
        i = i + 1
      }
      #Caso do último ser o sinal
      if ((i == n) & (L[i] == 1)) {
        bs = n
      #Caso do último ser um espaço
      }else if ((i == n) & (L[i] == 0)) {
        bs = 0
      }else if ((i < n) & (L[i] == 1)) {
        bs = i
      }
      if (bs > 0) {
        Ls <-c(Ls, bs)
      }
    }
    i = i + 1
  }
  return(Ls)
}
```

Essa função recebe um vetor correspondente a um dos splits do sinal gerado pelo autômato e retorna as posições em que há ocorrência do valor 1, sendo assim, as distâncias entre os símbolos corresponde à diferença entre o valor de uma posição e o valor da posição anterior no vetor.

Sinal	Inter símbolos	Tamanho dos Intersímbolos			
		Média	Desvio	Entropia	Eveness
0	52	3.50	2.75	2.16	4.48
1	32	6.13	2.45	1.23	2.35
2	64	3.06	2.42	1.92	3.79
3	44	4.43	2.86	1.32	2.49

Nesse caso, a média corresponde ao comportamento contrário da média do burst e se pegarmos a quantidade de valores intersímbolos e retirarmos as ocorrências da distância 1, temos o número de bursts.

2.4 Espectro de Potência - Transformada de Fourier

Foi utilizada a rotina FFT do R para calcular a transformada discreta de Fourier dos splits e o espectro de potência pelo método da multiplicação pelo conjugado.

Magnetudes	Sinal			
	0	1	2	3
Média	45.2	28.9	39.6	35.7
Desvio Padrão	5.2	3.6	5.4	3.9

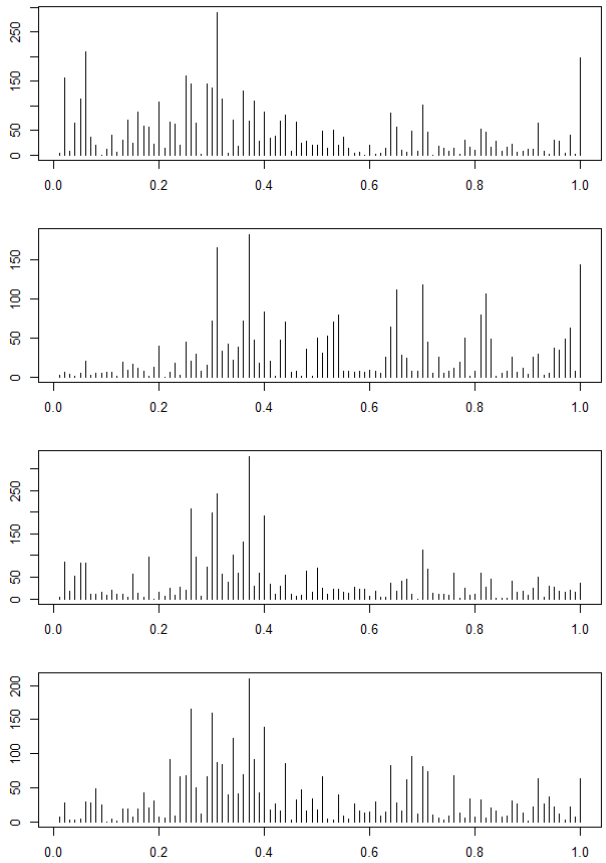

Figura 13: Espectro de potêcnia dos splits

2.5 Método de visibilidade

A função à seguir recebe o sinal gerado pelo autômano e transforma em uma matriz de visibilidade $(n : n)$:

```
visibility <- function(L, n) {
  A <-matrix( rep( 0, len=(n*n)), nrow = n)
  for (j in 2:n) {
    for (i in 1:(j-1)) {
      flag = 1
      k = i + 1
      while ((k <= (j-1)) & (flag == 1)) {
        aux = L[j] + (L[i] - L[j])*(j-k)/(j-i)
        if (L[k] >= aux) {
          flag = 0
        }
        k = k + 1
      }
      if (flag == 1) {
        A[i,j] = 1
        A[j,i] = 1
      }
    }
  }
  return(A)
}
```

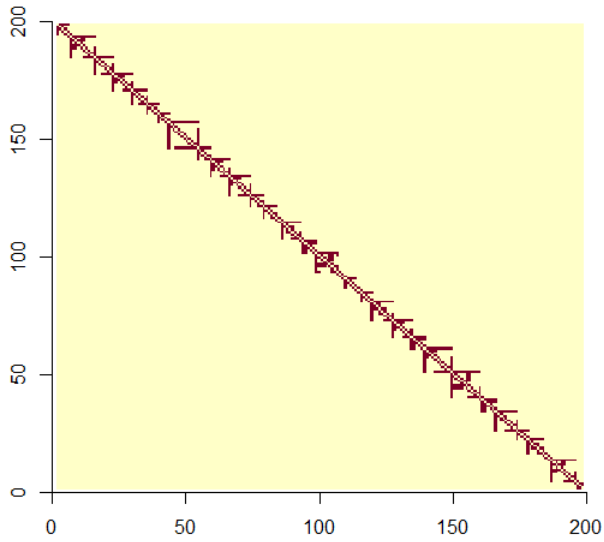

Figura 14: Matriz de visibilidade

Somando as linhas, temos um vetor $(1 : n)$, cuja média é o grau dos nós.

Grau Médio	Desvio Padrão	Coef. de Aglomeração
4.38	2.46	0.02

3 Análise de Componentes Principais (PCA)

Função geradora da distribuição uniforme circular:

```r
circlenormaldistribution <- function(n, r=1) {
  # inicializa os vetores x e y
  x <- c()
  y <- c()
  #Loop para preencher os vetores x e y
  while (length(x) < n) {
    #Gera um ponto aleatorio
    p = runif(2, min=-r, max=r)
    #Verifica r
    if ( p[1]**2 + p[2]**2 <= r**2 ) {
      # adiciona o ponto nos vetores x e y
      x <- c(x, p[1])
      y <- c(y, p[2])
    }
  }
  return(data.frame("x"=x, "y"=y))
}
```

Distribuição de pontos com $R \leq 1$:

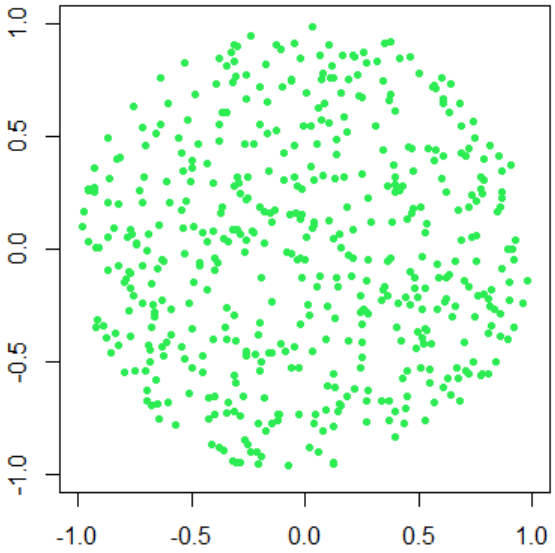

Figura 15: Distribuição circular

Função para deformar a distribuição:

```r
deformation <- function(x, y, xaxis=1, yaxis=1) {
  #Deforma x e y
  x = x*xaxis
  y = y*yaxis
  return(data.frame("x"=x, "y"=y))
}
```

Deformação $y = y \times 0.2$:

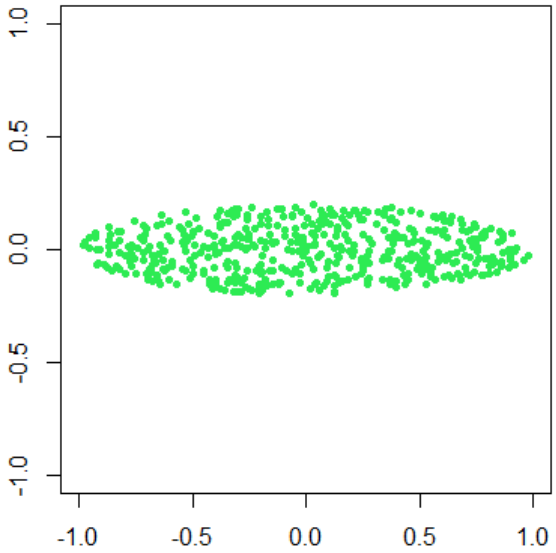

Figura 16: Distribuição circular deformada

Função para rotacionar a distribuição:

```r
rotation <- function(x, y, theta=0) {
  #Aplica uma rotação de angulo theta no sentido
      anti-horário
  x = x*cos(theta) + y*sin(theta)
  y = y*cos(theta) + x*sin(theta)
  return(data.frame("x"=x, "y"=y))
}
```

Matrix de Covariância	
0.2016693	0.1062295
0.1062295	0.0630756

Autovalor	Autovetor	
	x	y
0.005538863	0.4762561	-0.8793066
0.259206089	-0.8793066	-0.4762561

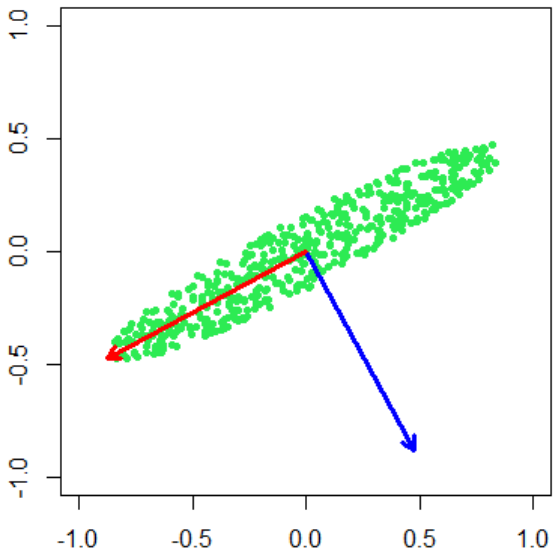

Figura 17: Distribuição circular deformada e rotacionada(30°) com os autovetores