2o. Trabalho Computacional

Prof. Dr. Guilherme de Alencar Barreto
gbarreto@ufc.br

06/01/2026

Departmento de Engenharia de Teleinformática (DETI)
Curso de Graduação em Engenharia de Computação (EngComp)
Programa de Pós-Graduação em Engenharia de Teleinformática (PPGETI)
Universidade Federal do Ceará (UFC), Campus do Pici, Fortaleza-CE

Objetivo - O objetivo deste trabalho é fazer com que o(a) aluno(a) implemente comandos básicos essenciais para geração (simulação) e estimação dos parâmetros de um processo autorregressivo de ordem 1 - AR(1)no Matlab/Octave. Para isso, iremos apresentar as seqüências de comandos e etapas necessárias para simular e estimar os parâmetros associados ao modelo de um processo AR(1). O(a) aluno(a) deve implementar na linguagem de programação de preferência e gerar um relatório com os resultados obtidos.

1 Fundamentação Teórica

Vimos em sala de aula e documentamos no material de apoio (slides) que um processo AR(1) possui as seguintes características:

(i) Modelo Matemático:

$$x(t) = a_0 + a_1 x(t-1) + \varepsilon(t), \tag{1}$$

em que $a_0 \in \mathbb{R}$ e $|a_1| < 1$. Além disso, a variável aleatória $\varepsilon(t)$ é gaussiana, de média nula e variância σ_ε^2 e não possui correlação serial. Resumidamente, temos que

$$\varepsilon(t) \sim N(0, \sigma_\varepsilon^2) \quad \text{e} \quad R_\varepsilon(\tau) = \begin{cases} \varepsilon, & \tau = 0 \\ 0, & \tau \neq 0 \end{cases} \tag{2}$$

(ii) Média Teórica:

$$\mu_x = \frac{a_0}{1 - a_1} \tag{3}$$

(ii) Variância Teórica:

$$\sigma_x^2 = \frac{\sigma_\varepsilon^2}{1 - a_1^2} \tag{4}$$

(ii) FAC Teórica (não normalizada):

$$R_x(\tau) = \sigma_x^2 a_1^{|\tau|} = \frac{\sigma_\varepsilon^2}{1 - a_1^2} a_1^{|\tau|}, \quad \tau = 0, \pm 1, \ldots, \pm \tau_{max}. \tag{5}$$

(ii) FAC Teórica (normalizada):

$$\rho_x(\tau) = \frac{R_x(\tau)}{\sigma_x^2} = a_1^{|\tau|} \quad \tau = 0, \pm 1, \ldots, \pm \tau_{max}. \tag{6}$$

2 Comandos no Matlab/Octave

- **Passo 1**: Definir valores adequados para a_0 e a_1. Definir também a variância do ruído σ_ε^2.

```
>> a0=2; a1=0.8; % Coeficientes do modelo
>> vet=0.1; % variancia do ruido
```

Comentário 1: Caso vocês desejem visualizar os valores definidos em `a0`, `a1` e `ve` basta retirar o ponto-e-vírgula que aparece ao final de cada comando. Caso você já tenha digitado com o ponto-e-vírgula, basta chamar o vetor correspondente digitando seu símbolo na linha de comando, individualmente (e.g. `>> a0`) ou separados por vírgulas (`>> a0,a1`). Experimente!

- **Passo 2**: Calcular a média (μ_x) e a variância (σ_x^2) teóricas do processo AR(1) usando as Eqs. (3) e (4), respectivamente.

```
>> mxt = a0/(1-a1) % media teorica do processo AR1
mxt = 10.000
>> vxt = vet/(1-a1*a1) % variancia teorica do processo AR1
vxt = 0.2778
```

- **Passo 3**: Calcular a FAC teórica $R_x(\tau)$ do processo AR(1) usando a Eq. (5).

```
>> LAGmax=20; % Valor maximo para os atrasos (lag)
>> tau=0:LAGmax; % Usa apenas tau >= 0
>> Rxt = vxt*(a1.^tau); % calcula fac p/ diferentes tau's
>> figure; stem(tau,Rxt); % plota a FAC
>> xlabel('Espacamento temporal (lag)')
>> ylabel('FAC'); title('FAC teorica de um processo AR(1)')
```

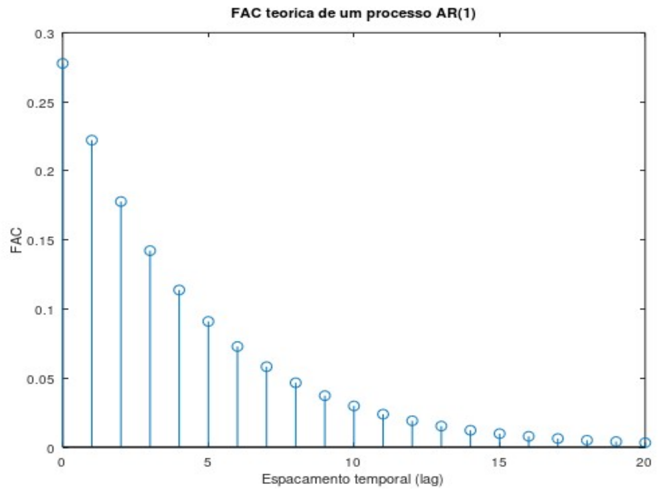
Comentário 2: Note o decaimento exponencial da FAC do processo AR(1). Este comportamento é típico de processos estocásticos (i.e., sistemas dinâmicos) estacionários.

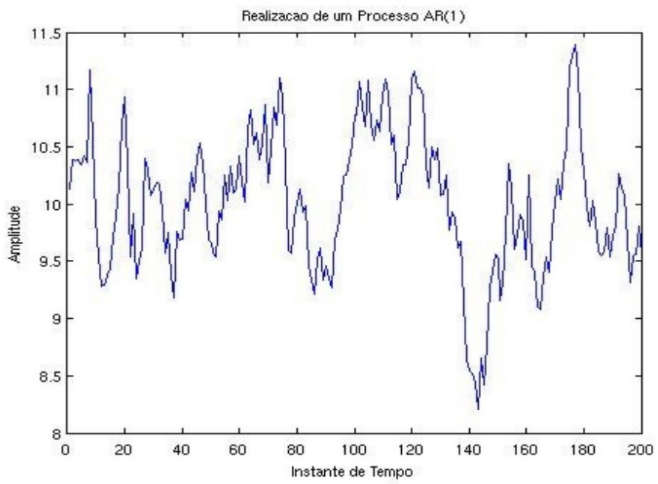
- **Passo 4**: Definir o número de amostras a serem simuladas e a condição inicial.

```
>> N=5500; % No.  De pontos a amostras a serem geradas
>> x=randn; % Condição inicial
```

- **Passo 5**: Simular o processo AR(1) por $2N$ iterações.

```
>> for t=2:2*N, ...
       x(t) = a0 + a1*x(t-1)+ normrnd(0,sqrt(vet)); ...
   end
```





Comentário 3: O comando `normrnd` gera um número aleatório normalmente distribuído de média 0 e desvio-padrão igual $\sqrt{vet}$. Note que especificamos anteriormente a variância do ruído. Para obter o desvio-padrão, basta extrair a raiz quadrada da variância. Para mais detalhes do comando `normrnd`, digite `>> help normrnd`.

Comentário 4: As reticências ao final da linha de comando permite a digitação de laços em várias linhas de comandos diretamente no prompt do ambiente.

- **Passo 6**: Desprezar as primeiras amostras geradas para mitigar a influência da condição inicial no processo de geração da realização $\{x(t)\}$.

```
>> x=x(N+1:2*N); % Extrai a metade final do sinal
```

- **Passo 7**: Visualizar um trecho do sinal gerado.

```
>> figure; plot(x(1:500));
>> xlabel('Instante de Tempo')
>> ylabel('Amplitude')
>> title('Realizacao de um Processo AR(1)')
```

Comentário 5: Note que o gráfico gerado emula (ie., imita) um sinal analógico, ou seja, de tempo contínuo, porque o comando `plot` conecta as amplitudes sequencialmente. Para visualizar o sinal de tempo discreto corretamente, deveríamos usar o comando `stem`.

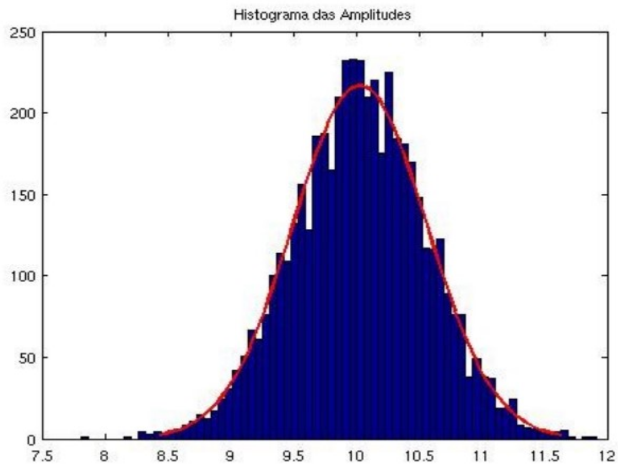
- **Passo 8**: Estimar a média amostral ($\bar{x}$) e a variância amostral (σ_x^2) das amplitudes do sinal gerado.

```
>> xbar=mean(x) % estimacao da media de x(t)

xbar = 9.9984

>> vxh=var(x) % estimacao da variancia de x(t)

vxh = 0.2798
```

Comentário 6: Note que estes valores são bem próximos dos valores teóricos calculados no Passo 2.

- **Passo 9**: Visualizar o histograma das amplitudes do sinal gerado.

```
>> figure; histfit(x);
>> title('Histograma das Amplitudes')
```

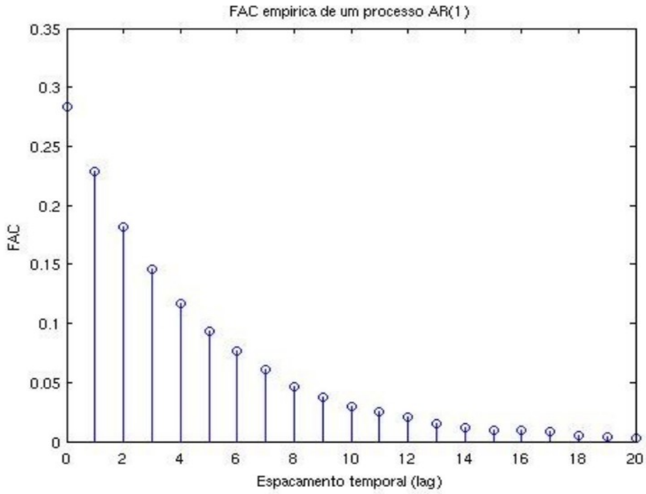


Comentário 7: Percebe-se que as amplitudes da realização $\{x(t)\}$ parecem seguir uma distribuição normal. Isso era esperado? Por quê? Nota-se também que as amplitudes concentram-se em torno do valor médio $\mu_x = 10$.

- **Passo 10**: Estimar e visualizar a função de autocorrelação amostral $\hat{R}_x(\tau)$ a partir do sinal gerado.

```
>> Rxh=[];
>> for tau=0:LAGmax, ...
       aux1=x(1:end-tau)-xbar; ...
       aux2=x(tau+1:end)-xbar; ...
       aux3=dot(aux1,aux2)/(N-tau); ...
       Rxh=[Rxh; aux3]; ...
```

```
    end
>> figure; stem(0:LAGmax,Rxh)
>> xlabel('Espacamento temporal (lag)')
>> ylabel('FAC'); title('FAC empirica de um processo AR(1)')
```



- **Passo 11**: Estimar os parâmetros a_0 e a_1, bem como a variância do ruído σ_ε^2 a partir do sinal gerado. Para isso, a partir da Eq. (5) podemos fazer $\tau = 1$, de modo que chegamos à seguinte expressão que nos permite estimar a_1:

$$\hat{a}_1 = \frac{\hat{R}_x(1)}{\hat{\sigma}_x^2} \tag{7}$$

Comentário 8: O símbolo (^) sobre as variáveis indica um valor estimado a partir dos dados.

Comentário 9: Este método é chamado de "estimação de parâmetros pelo método dos momentos", já que a autocorrelação é um momento estatístico (assim como a média e a variância).

Assim, utilizamos o seguinte comando:

```
>> a1h=Rxh(2)/vxh

a1h = 0.8065
```

Comentário 10: Note que o vetor Rxh contém os valores da FAC para $\tau = 0, 1, \ldots, \tau_{max}$ sendo representado matematicamente como

$$\texttt{Rxh} = \begin{bmatrix} \hat{R}_x(0) \\ \hat{R}_x(1) \\ \hat{R}_x(2) \\ \vdots \\ \hat{R}_x(\tau_{max}) \end{bmatrix} = \begin{bmatrix} \hat{\sigma}_x^2 \\ \hat{\sigma}_x^2 a_1 \\ \hat{\sigma}_x^2 a_1^2 \\ \vdots \\ \hat{\sigma}_x^2 a_1^{\tau_{max}} \end{bmatrix}, \tag{8}$$

sendo que este vetor é representado pela variável `fac` no Octave/Matlab sendo indexado a partir de 1. Por isso, pegamos a segunda componente do vetor `fac` (ou seja, `fac(2)`) que contém $\hat{R}_x(1) = \hat{\sigma}_x^2 a_1$.

Para determinar uma estimativa do parâmetro a_0 usamos a Eq. (3). Daí, deduzimos a seguinte expressão: $\hat{a}_0 = \bar{x}(1 - \hat{a}_1)$, em que $\bar{x}$ é a média amostral das amplitudes de $x(t)$. Para o cálculo, utilizamos os seguintes comandos:

```
>> a0h=xbar*(1-a1h)

a0h = 1.9411
```

Finalmente, para obter uma estimativa da variância do ruído usamos a Eq. (4), de onde chegamos ao seguinte comando:

```
>> veh=vxh*(1a1h*a1h)

veh = 0.0990
```

Comentário 11: Compare os valores estimados de $\hat{a}_0$ (`a0h`), $\hat{a}_1$ (`a1h`) e $\hat{\sigma}_\varepsilon^2$ com os respectivos valores reais especificados no Passo 1. Pode-se notar que são bem próximos.

Comentário 12: Por fim, vale comentar que na sua simulação os valores de a_0 e a_1 não serão os mesmos que os mostrados acima, pois os valores do ruído gerados pelo comando `normrnd` serão diferentes cada vez que você rodar a simulação.

DIVIRTAM-SE!!