

Relatório de E.T.

**Laboratório nº1**

**Curso:** METI

**Turno:** 3ª feira 15:00 > 16:30

**Grupo:** Bancada 8

|  |
| --- |
| **Trabalho realizado por:** |
|  |
| Luís Pereira, nº77984 |
| Ruben Condesso, nº 81969 |

**Índice**

**1 -** [**Introdução**](#_oau7gxiep0f4) **2**

**2 -** **Geração do Processo de Poisson 3**

2.1 - [Desenvolvimento do programa](#_bo1nfgz6t65f) 3

2.2 - [Resultados](#_2hte9d7daa55) obtidos 5

2.3 - [Conclus](#_4m25jaypn5os)ões 7

**3 - Sobreposição do Processo de Poisson 8**

3.1 - [Desenvolvimento do programa](#_95qpjdtu0h75) 8

3.2 - [Resultado](#_v9se7uv92jw6)s obtidos 9

3.3 - [Conclus](#_1sag2c8ms62t)ões 9

**4 - Simulação de Queue** [**M/M/1**](#_lazn8ulmoos6)  **10**

4.1 - [Desenvolvimento do programa](#_esnly7wxw30i) 10

4.2 - R[esultados](#_sikra85h0gvp) obtidos 11

4.3 - [Conclus](#_g8fbrtxyq5im)ões 13

**5 -** [**Conclusão**](#_5vad5ql7m8xp) **14**

# 

# 

# **Introdução**

A finalidade do primeiro trabalho laboratorial da disciplina de Engenharia de Tráfego, tem por base simular e analisar o processo de chegada de *Poisson*, qual a sua aplicação no processamento de fila e discutir os resultados obtidos.

O relatório está dividido em três tópicos, referentes aos três exercícios do enunciado do laboratório: 1 - geração do processo de chegada de *Poisson*; 2 - Sobreposição do processo de *Poisson*; 3 - Simulação do M/M/1 *queue*. É disponibilizado em anexo o código para cada exercício.

No relatório, para cada exercício, é feita uma explicação do desenvolvimento do programa, é feita uma discussão dos resultados obtidos e as conclusões que se podem retirar. Tal irá ser feito com o auxílio de gráficos e resultados de variáveis apropriadas que vão ser igualmente evidenciados ao longo do relatório.

# **Geração do Processo de Poisson**

O objetivo do primeiro exercício é simular um processo de *Poisson.* Irá ser usada uma sequência de N eventos, com intervalos de tempo entre eventos que apresentam uma distribuição exponencial, segundo um parâmetro lambda.

Para retirar conclusões do exercício, os valores de lambda irão ser: [0.5;1.0;5.0;10.0] e os valores de N irão variar entre 10 e 20000.

Irão ser gerados gráficos adequados que servirão pela razão explícita no parágrafo acima.

### *2.1 - Desenvolvimento do programa*

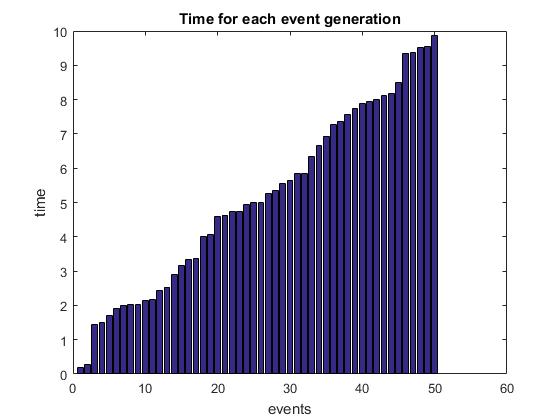
 Começamos por criar um vetor, *vector\_eventos*, que contém os valores gerados aleatoriamente segundo uma distribuição exponencial, graças ao uso da variável *delta\_t* que representa uma *pseudo* variável aleatória com esse tipo de distribuição, como se pode verificar pelo enunciado. Esse vetor foi gerado para 50 eventos numa primeira instância. Pode-se consultar o gráfico que o representa em baixo:

Figura 1 - Instante de tempo t em que cada evento foi gerado

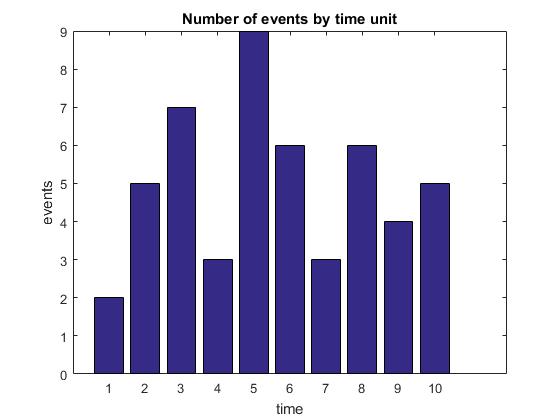
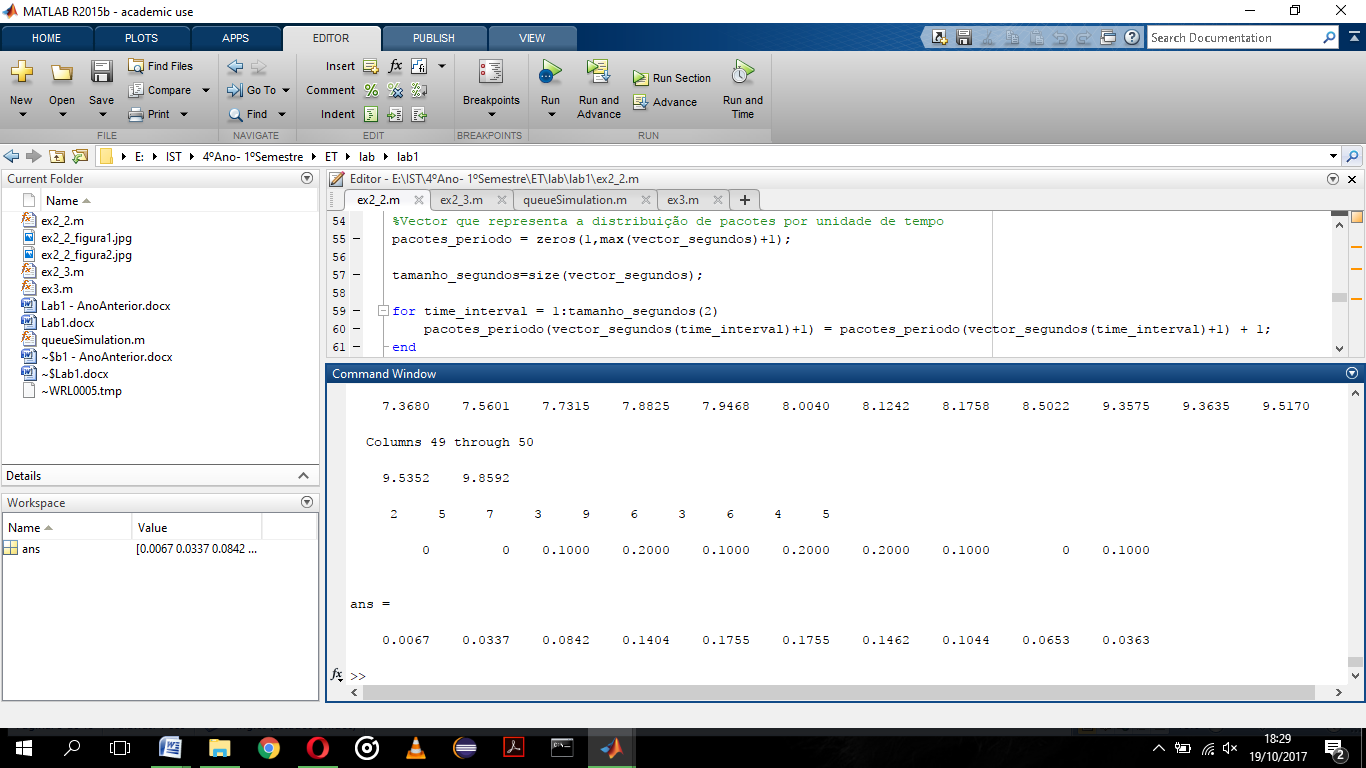
 De seguida, gerámos um segundo vetor, *vector\_segundos*, que tem como finalidade ver, dado a distribuição dos intervalos de tempo gerados anteriormente, quantos pacotes podiam ser colocados em cada segundo. Tal é ilustrado pela figura 2:

Figura 2 - Número de Pacotes por instante de tempo t

Gerámos depois o vetor, *vector\_periodo*, que ilustra a distribuição de pacotes por unidade de tempo, ou seja, a sua probabilidade, dado o número de pacotes pela mesma unidade de tempo, como vimos no gráfico anterior. O *display* do vetor poderá ser *visualizado em baixo:*



Posteriormente é criado um vetor, *Poisson*, que irá dar origem ao vetor que representa a distribuição para um dado lambda, usando a variável *poisson*, onde esta impõe a probabilidade de observar um certo número de eventos, num período unitário, num processo de chegada de *Poisson*, dado pela fórmula que se encontra no enunciado. Dá-se então origem ao último gráfico deste exercício:

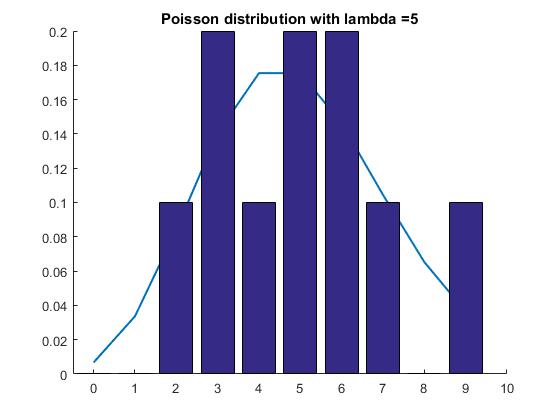


Figura 3 - Histograma final: Distribuição de Poisson, para lambda=5 e N=50

Pode-se verificar que o centro do gráfico converge para 5, sendo este o valor de lambda.

*2.2 - Resultados obtidos*

Iremos agora figurar vários gráficos que representam a distribuição de *Poisson*, para vários valores de lambda e do número de eventos, que nos permitirão chegar a conclusões mais concretas e robustas.

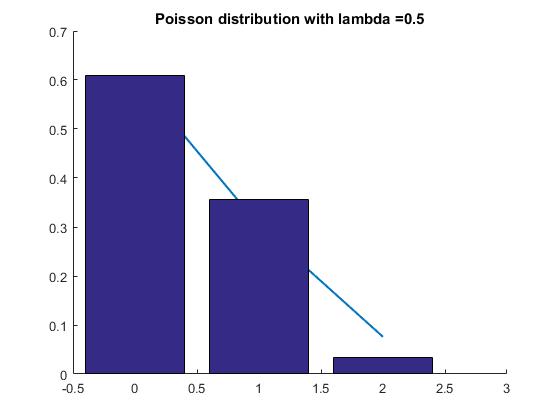
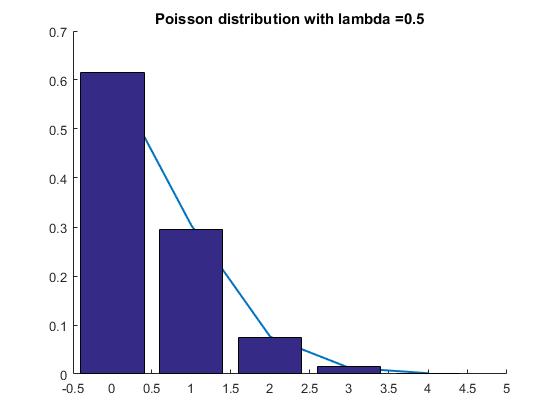


Figura 5 - Distribuição de Poisson, para lambda=0.5 e N=1000

Figura 4 - Distribuição de Poisson, para lambda=0.5 e N=50

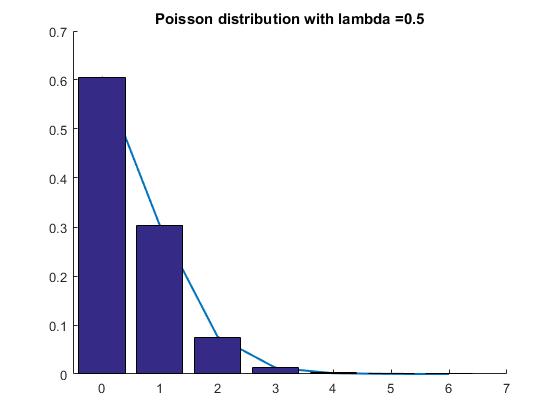
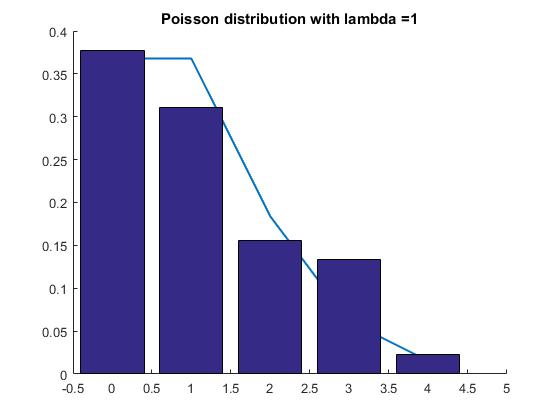


Figura 7 - Distribuição de Poisson, para lambda=1 e N=50

Figura 6 - Distribuição de Poisson, para lambda=0.5 e N=20000

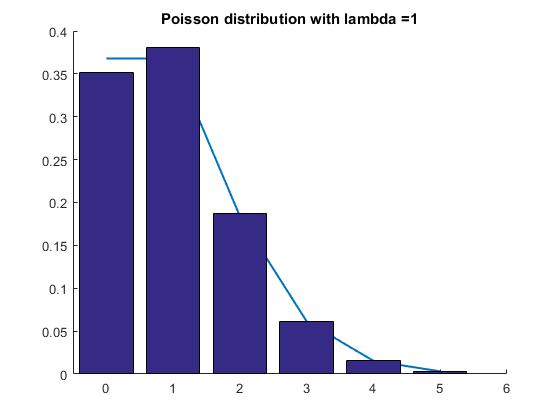
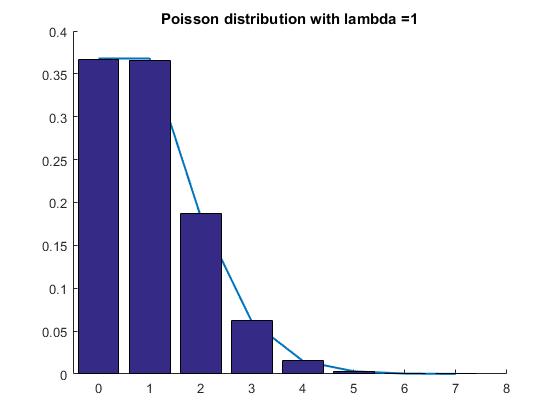


Figura 8 - Distribuição de Poisson, para lambda=1 e N=1000

Figura 9 - Distribuição de Poisson, para lambda=1 e N=20000

Figura 7 - Distribuição de Poisson, para lambda=1 e N=50

Figura 7 - Distribuição de Poisson, para lambda=1 e N=50

### *ex2_2_figura11.jpgex2_2_figura10.jpg*

Figura 11 - Distribuição de Poisson, para lambda=5 e N=1000

Figura 10 - Distribuição de Poisson, para lambda=5 e N=50

### *ex2_2_figura12.jpgex2_2_figura13.jpg*

Figura 13 - Distribuição de Poisson, para lambda=10 e N=50

Figura 12 - Distribuição de Poisson, para lambda=1 e N=20000

### *ex2_2_figura15.jpgex2_2_figura14.jpg*

Figura 15 - Distribuição de Poisson, para lambda=10 e N=20000

Figura 14 - Distribuição de Poisson, para lambda=10 e N=1000

### *2. 3 - Conclusões*

# A partir dos gráficos gerados, para os vários valores lambda e de N, podemos verificar, primeiro, que quanto maior for o número de eventos mais se aproximará o centro do gráfico do valor de lambda. Isto provêm da definição de probabilidade, ao nível da frequência, que quanto maior o universo de elementos testados, mais aproximados são os resultados da simulação comparativamente com o valor teórico .

# Também é percetível, por análise dos gráficos acima, que para valores de lambda superior ou igual a 5, existe uma maior aproximação a uma distribuição de *Poisson*. Esta é mais próxima quanto maior for o número de eventos, tal como foi explicado acima. Tal leva a crer que quanto maior o número de eventos e maior o valor de lambda, maior será a aproximação do gráfico gerado a uma distribuição de *Poisson*.

# **Processo de Sobreposição de Poisson**

Neste exercício, foi-nos pedido para modificar o código apresentado no exercício anterior, de modo a gerar três sequências independentes, cada uma com um valor de lambda diferente. De seguida, é imposto que se combine as três sequências numa só, e por fim teremos de responder se essa mesma sequência é um processo de *Poisson*.

Mais uma vez, as conclusões e os resultados demonstrados irão ser ilustrados com o auxílio de gráficos (histogramas) e variáveis.

### *3.1 - Desenvolvimento do programa*

Para este exercício sabíamos à priori da existência da sobreposição dos processos de *Poisson*. Como tal, começamos por criar uma função que recebia o numero de eventos e um vetor de Lambdas que serão usados para calcular a sobreposição.

Neste caso apenas eram passados 3 valores de lambda e para cada um deles calculámos o tempo atribuído a cada lambda, que é dado por :

* (Nr\_eventos / max(lambda) )\* lambda

Para cada um destes lambdas voltamos a gerar, como no exercício anterior, um vector de eventos que é dado pela *pseudo* variável aleatória *delta\_t*. Devido à sobreposição do processo de *Poisson* o próximo passo é concatenar os 3 vetores e obter um vetor final para a nossa experiência.

Posteriormente, para saber quantos pacotes ocorrem em cada segundo, vamos verificar desde o 1º segundo até ao ultimo segundo ( numero natural ) que existe no vector final ( que corresponde à concatenação dos 3). De seguida, repetimos o processo do exercício 2.2 para verificar as probabilidades de ocorrer um determinado número de pacotes por segundo.

Por último, para calcular a *Poisson*, iremos fazer uma soma dos lambdas passados como argumento no início da função e gerar o gráfico que apresentaremos em baixo.

### *ex2_2_figura16.jpg3.2 - Resultados obtidos*

### 

Figura 16 - Distribuição de Poisson, para N=20000, lambdas =1;5;10

### *ex2_2_figura17.jpg*

Figura 17 - Distribuição de Poisson, para N=1000, lambdas =0.5;1;5

### *3.3 - Conclusões*

Sendo as três sequências geradas para cada valor de lambda dado, é esperado que a distribuição de *Poisson* da sequência que representa a junção das três anteriores seja idêntica à soma dos valores dos lambdas dados. Pode-se verificar tal facto nos gráficos ilustrados no ponto 3.2: No primeiro gráfico verifica-se que o centro do gráfico se encontra no valor 16 (1+5+10); o mesmo raciocínio que pode fazer no gráfico da figura 17.

# **Simulação de Queue M/M/1**

No último exercício deste laboratório, é proposto simular uma *queue* M/M/1. É imposto que o processo de chegada à *queue* seja feito segundo uma distribuição de Poisson com um parâmetro *lambda*.

O processamento na *queue* é feito tendo por base o modelo FIFO, e *t* é o tempo que demora a ser processado cada evento na mesma. Esta variável é aleatória, segundo uma distribuição aleatória com parâmetro *miu*.

Para a discussão de resultados e conclusões que irão ser, mais uma vez, auxiliados por gráficos e variáveis convenientes, vamos dar ênfase para as seguintes simulações:

* *miu* < *lambda*;
* *miu* = *lambda*;
* *miu* > *lambda*.

### *4.1 - Desenvolvimento do programa*

Para o desenvolvimento deste exercício, começamos por escrever a expressão do valor teórico do problema em causa, que servirá para comparar com o valor prático calculado ao longo do exercício.

Posteriormente, definimos como variável fixa e global o valor do tempo total em que decorrerá a simulação, por uma questão de facilidade. Depois assumindo o raciocínio implementando no ponto 2 deste relatório, mais concretamente na geração do vetor de eventos, foram criados dois vetores: *vetor\_eventosTempo* e *vetor\_eventosServico*.

No primeiro, ao chamar a variável *delta\_t* (explicada no ponto 2 também), passamos como input o valor de *lambda* e no outro vetor passamos o valor *de miu*. O primeiro representa o vetor com os tempos de chegada de cada pacote, ou seja, a distribuição exponencial dos intervalos de tempo gerados; o segundo, o tempo que cada pacote demora a ser "processado", logo têm ambos o mesmo tamanho. Resumindo, temos os tempos de partida e de chegada.

De seguida, segue-se o cálculo da média do valor prático e começamos por criar um ciclo que, enquanto uma variável chamada *tempo\_atual* fosse menor que o tempo total da simulação, iriamos fazer o seguinte: comparar os primeiros dois valores de cada vetor, onde o menor seria o que tinha chegado primeiro. Isso iria originar duas situações:

* caso o primeiro a chegar fosse do *vetor\_eventosTempo* o valor da *queue* seria incrementado com +1 no seu valor (variável que representa este valor é designada por *count*);
* caso pertence-se ao *vetor\_eventosServico* o valor da *queue* seria decrementado com -1 no seu valor. O menor valor em causa, seria retirado do vetor a que pertencesse e o *tempo\_atual* seria igual a esse mesmo valor.

Relativamente ao cálculo do valor prático do tamanho da *queue*, à medida que o valor da variável *tempo\_atual* for maior que o valor da variável *count*, é guardado esse mesmo valor no vetor *queue\_vector*, associado a esse instante, e vai sendo calculada a média de todos os tamanhos guardados nesse vetor. Para cada tempo unitário é guardado então no vetor *vector\_final*, onde o último valor representará o valor da média final. Esta explicação dada encontra-se no ficheiro *queueSimulation.*

No ficheiro *ex\_3* é chamada a função *queueSimulation* n vezes e é calculada a média de todos os resultados encontrados, tal ação vai ao encontro de tentar encontrar um resultado final mais robusto. Por fim, sim é comparada essa média com o valor teórico.

### *4.2 - Resultados obtidos*

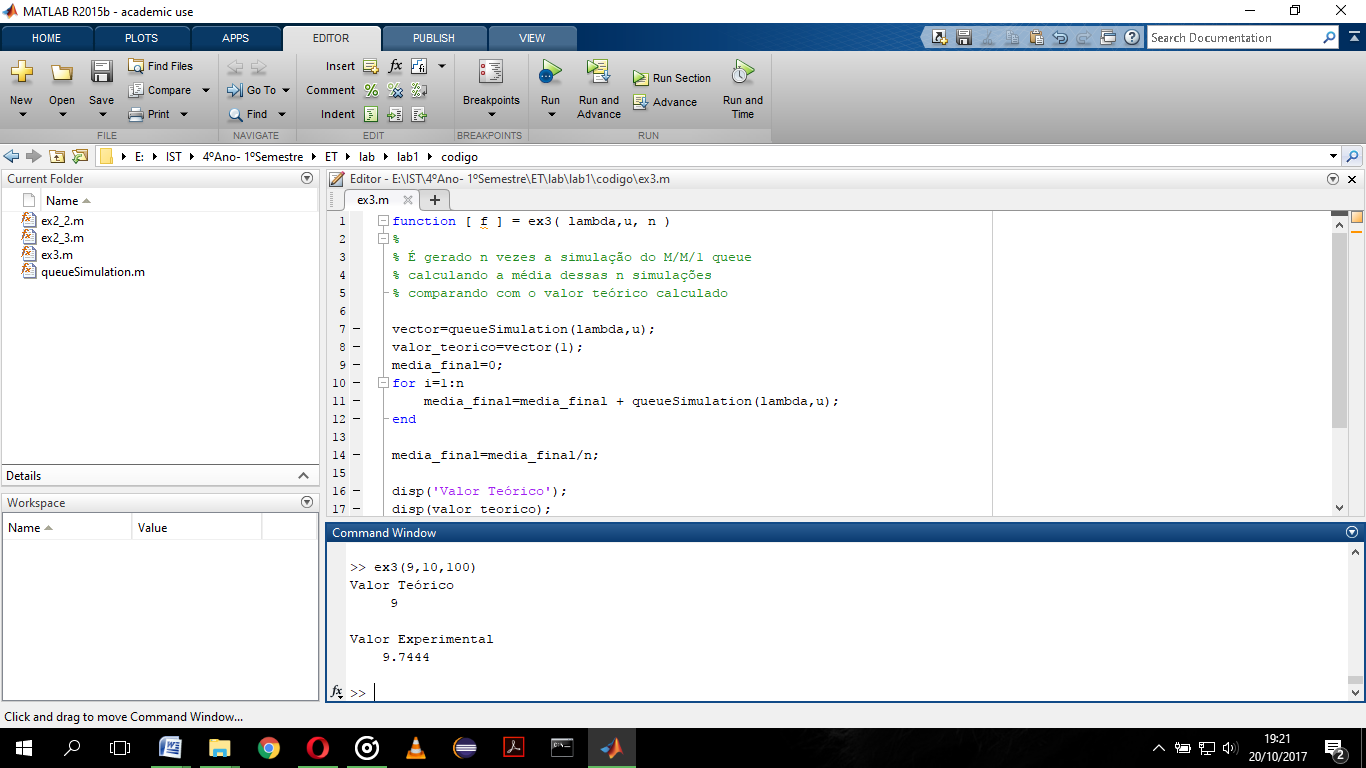


Figura 18 -Valor médio do tamanho da queue, para lambda=9;miu=10;N=100

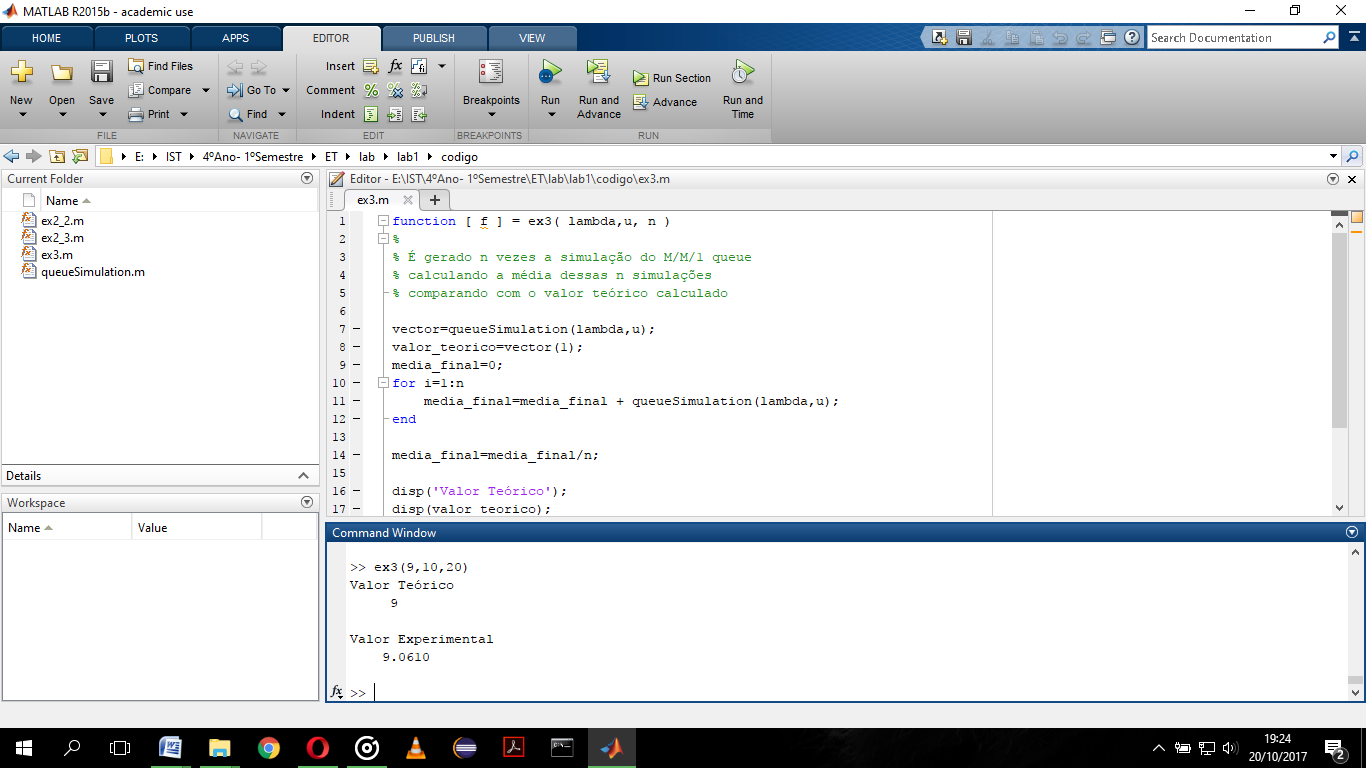


Figura 19 -Valor médio do tamanho da queue, para lambda=9;miu=10;N=20

# 

Figura 18 -Valor médio do tamanho da queue, para lambda=10;miu=10;N=20

# 

Figura 18 -Valor médio do tamanho da queue, para lambda=10;miu=10;N=100

# 

Figura 18 -Valor médio do tamanho da queue, para lambda=9;miu=10;N=20

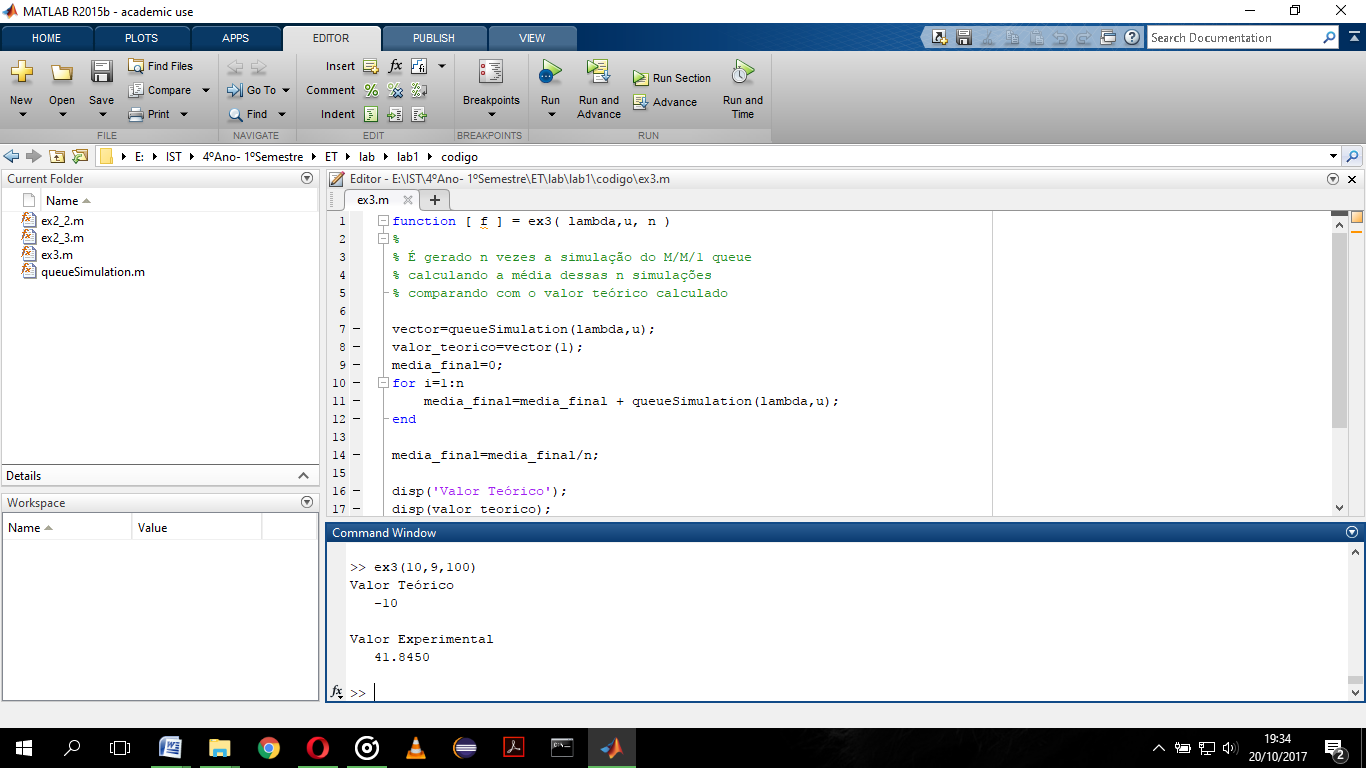


Figura 18 -Valor médio do tamanho da queue, para lambda=9;miu=10;N=100

# *4.3 - Conclusões*

Para cada relação entre *lambda* e *miu* podemos verificar através da comparação entre os valores práticos gerados e os valores teóricos associados, que quando o *lambda* é menor que *miu*, o valor experimental aproxima-se bastante do valor teórico, sendo então possível calcular o tamanho médio da *queue*. Para o caso de o *lambda* ser igual a *miu* não é possível tirar conclusões dado ao facto que o valor teórico é infinito, devido à sua formula (*lambda/(u-lambda*). Para o último caso, os resultados são inconclusivos sendo que o valor teórico é sempre negativo, como se pode ver nos *displays* gerados.