Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Faculdade de Estatística

Relatório sobre modelagem de série temporal utilizando um modelo de decomposição

Caio Sanches Bentes
201306840059

Belém-Pará
2016

Universidade Federal do Pará
Instituto de Tecnologia
Faculdade de Engenharia da Computação e Telecomunicações

Neste trabalho modelo uma série que contén os Índices volumes de vendas no varejo no Rio Grande do Norte de janeiro de 2000 a outubro de 2016 utilizando uma modelagem de decomposição em têndecia e sazonalidade

Caio Sanches Bentes
201306840059

Belém-Pará
2016

1 – INTRODUÇÃO

Em estatística, econometria, matemática aplicada e processamento de sinais, uma série temporal é uma coleção de observações feitas sequencialmente ao longo do tempo. Em modelos de regressão linear com dados cross-section a ordem das observações é irrelevante para a análise, em séries temporais a ordem dos dados é fundamental. Uma característica muito importante deste tipo de dados é que as observações vizinhas são dependentes e o interesse é analisar e modelar esta dependência.

O modelos estatísticos são representados por equações derivadas diretamente dos dados disponíveis, sem recorrer a qualquer teoria, exceto no caso da escolha dos dados a serem utilizados. Já os modelos econométricos, levam em conta uma dada teoria econômica, a qual quase sempre não pode ser convenientemente modelada matematicamente.

2 – OBJETIVO

Neste relaorio objetivo modelar a serie de Índices volumes de vendas no varejo no Rio Grande do Norte de janeiro de 2000 a outubro de 2016, me utilizando do método de decomposição para assim modelar sua tendência utilizando Tendência polinomial e sua sazonalidade usandoo Método da regresão para sazonalidades determinísticas.

3 – PROCEDIMENTOS EXPERIMENTAIS

3.1 – MATERIAIS

- Índices volumes de vendas no varejo no Rio Grande do Norte de janeiro de 2000 a outubro de 2016
-MATLAB

3.2 – EQUIPAMENTOS

-MATLAB

3.3 – MÉTODOS

-Tendência polinomial

O método clássico de análise de regressão pode ser utilizado para estimar os parâmetros do modelo assim como suas tendências. Nós vamos considerar precisamente a tendência linear.

Considere o modelo com tendência determinística linear

$Zt = \beta 0 + \beta 1 t + Xt$

$Q(\beta 0, \beta 1) = \sum_{t=1}^{n} (Zt - \beta 0 - \beta 1 t)^2$

onde β0 e β1 são o intercepto e a inclinação, respectivamente, e os parâmetros desconhecidos. O método clássico de minímos quadrados (ou regressão) é utilizado para estimar os parâmentros β0 e β1, que minimizam

A solução pode ser obtida calculando as derivadas parciais com relação a β0 e β1 igualando-as a zero, encontrando as equações normais

Daí encontraremos os seguintes estimadores de minímos quadrados.

teremos então o seguinte estimador de minímos quadrados $\hat{\beta} = (Y^T Y)^{-1-1} Y^T Z$.

-Sazonalidade determinística - método de regressão

Os métodos de regressão são ótimos para séries que apresentam sazonalidade determinística, ou seja, esta pode ser prevista perfeitamente a partir de meses anteriores.

Supondo a sazonalidade constante, α_j não depende de T. Pode-se ter, por exemplo,

dij = 1, se o período t corresponde ao mês j, j = 1, ..., 12.
0, caso contrário.

de modo que a matriz de regressão não é de posto completo, mas de posto m + 12 (temos m + 13 parâmetros). Impondo-se a restrição adicional:

X 12 j=1 αj = 0

$$\sum_{j=1}^{12} aj = 0$$

obtém-se um modelo de posto completo:

$$Zt = \sum_{j=0}^{m} \beta j\, t^j + \sum_{1}^{1} \alpha j Djt + at$$

onde agora

Djt = 1, se o período t corresponde ao mês j,
−1, se o período t corresponde ao mês 12
0, caso contrário.

Utilizando o método de mínimos quadrados pode-se obter os estimadores de αj e βj , ou seja, a partir de uma amostra Z1, ..., ZN obtém-se o modelo matricial:

Z = Cβ + Dα + a

A equação pode ser escrita na forma:

Z = Xγ + a
onde
X = [C : D]

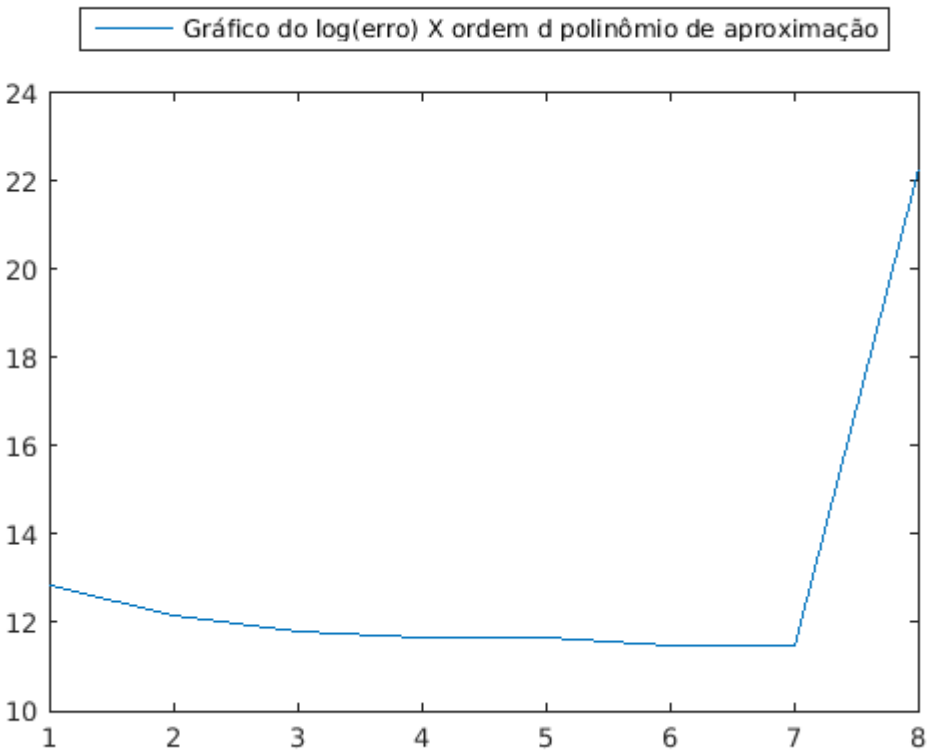
e γ = •| β |
| α |

de modo que:

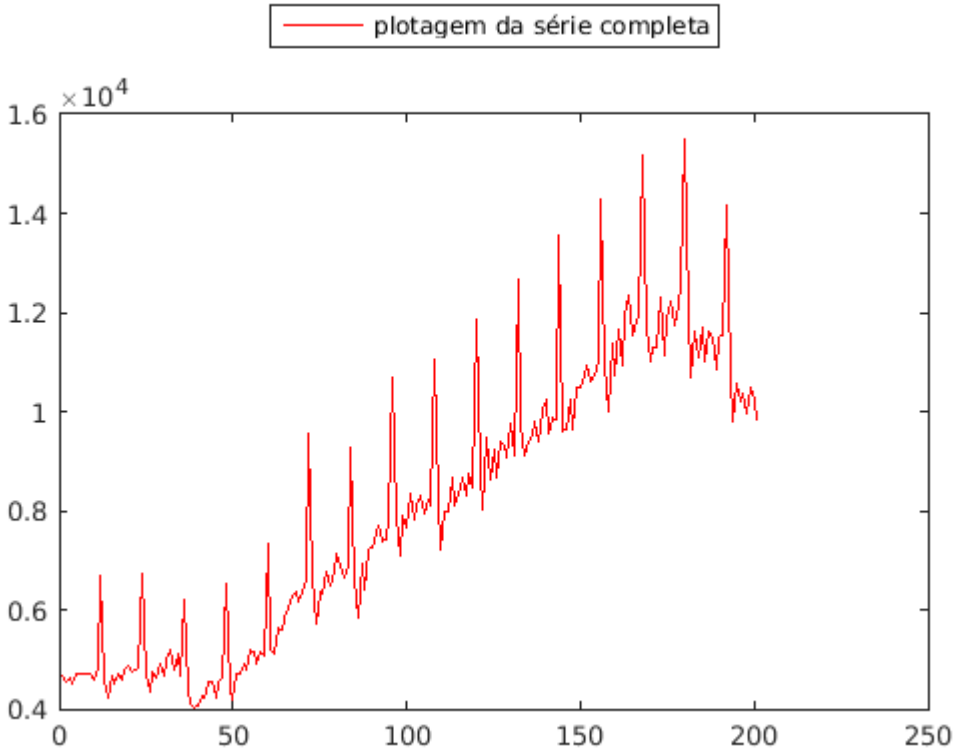
$\hat{\gamma} = [X' X\}^{-1} X' Z$
são os estimadores de mínimos quadrados

4 – RESULTADOS E DISCUSSÕES

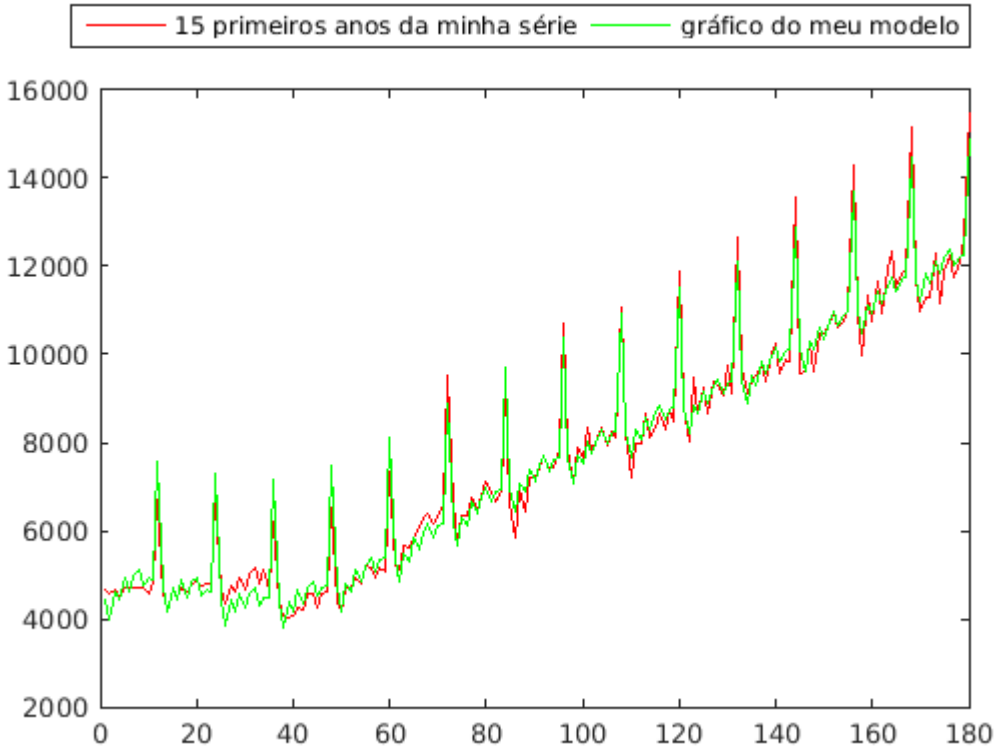
Criei uma função que gera um vetor com o log neperiano do erro médio quadrático de de minha modelagem em relação a original. Ela me retornou o seguinte gráfico.



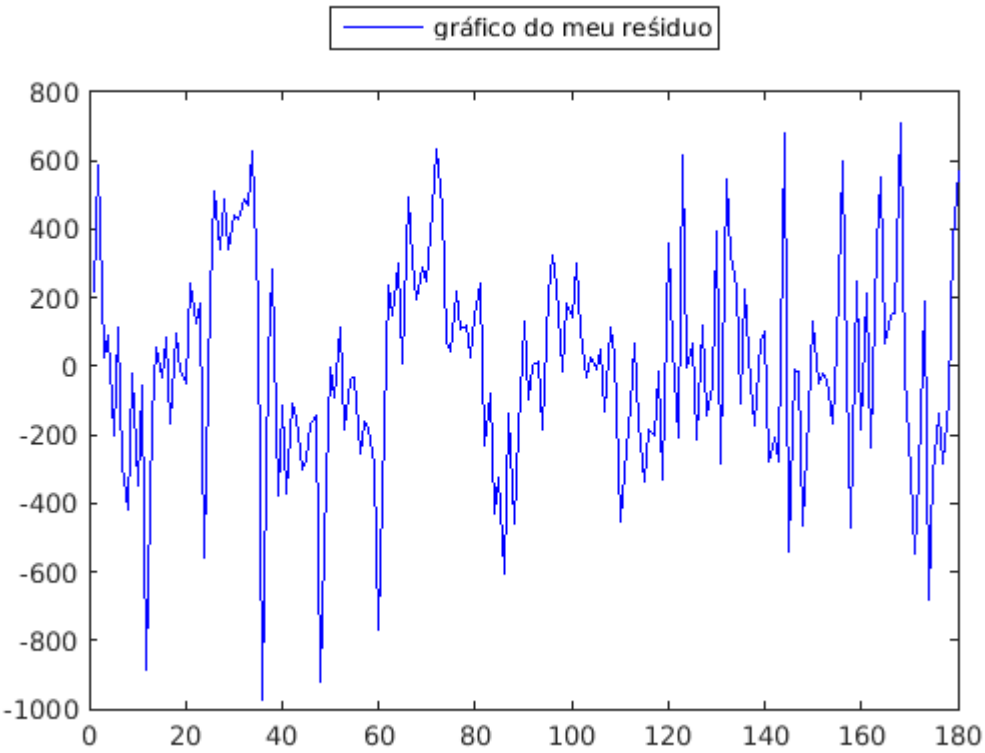
Partindo disso me utilizei da 7ª ordem para modelagem. Segue imagens com os passos:

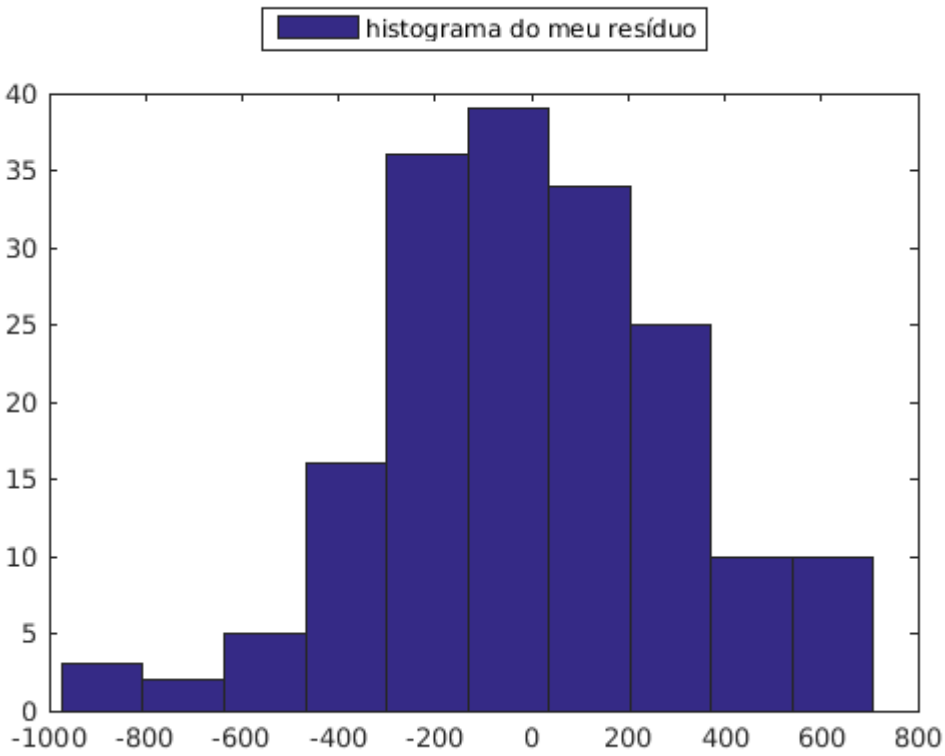


Meu modelo de 7ª ordem.



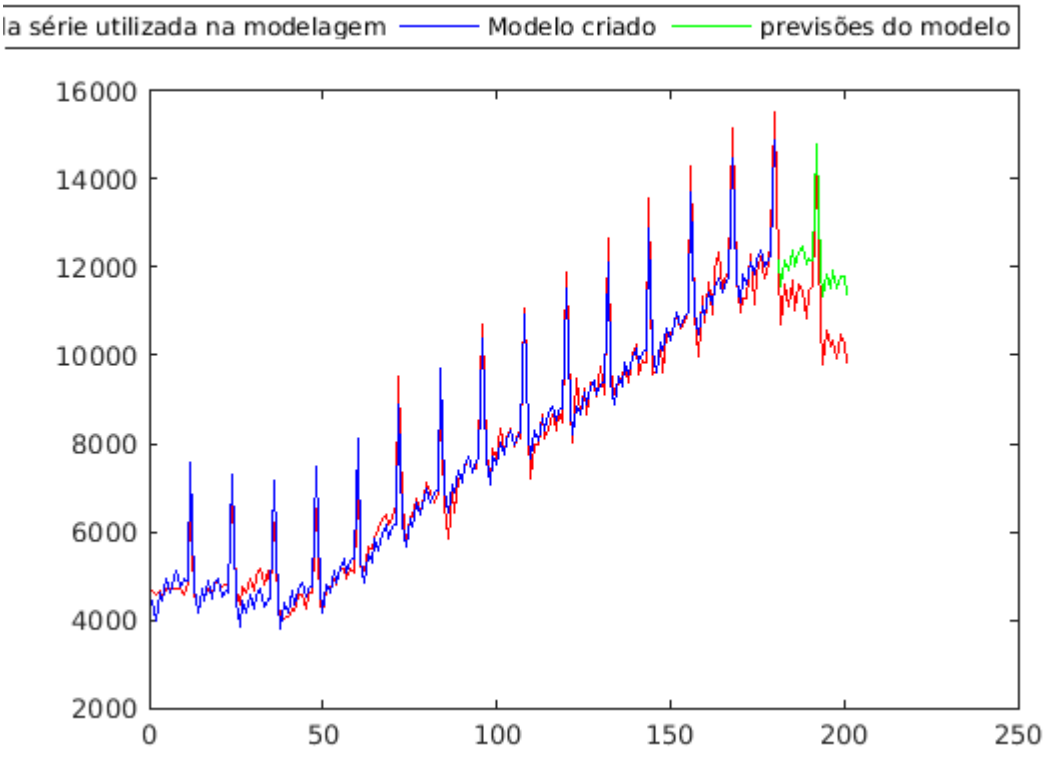
Resíduo obtido.





Histograma do resídio(que teve media muito próxima de zero)

Etapa de previsões



5 – CONCLUSÃO

Concluí para os seguintes casos que se utilizando de um função de 7ª ordem obtenho o menor erro na modelagem e previsão da minha série. Além de que ela melhor representa a queda nos dois ultimos anos.

6 – BIBLIOGRAFIA

MORETTIN, Pedro A. e TOLOI, Clélia M. Séries Temporais. 2ª ed. São Paulo: Atual, 1987.

MORETTIN, Pedro A. e TOLOI, Clélia M. Previsão de Séries Temporais. 2ª ed. São Paulo: Atual, 1987.

CHATFIELD, C. The Analysis of Time Series: An Introduction. Chapman-Hall, 1984.

ANEXO A: Códigos em matlab

Código principal

```
st = transpose(importfile('STP-20161121152409392.csv', 2, 202)); % importa dados da minha Série temporal
Z = st(1:180); %Parte da série que será utilizada para modelagem.
N= 7; %Ordem da função que modela a tendência
T = minimos(length(Z),N); % Obtenção da matriz T
anos = 15; %anos utilizados para modelagem
D = obtemD(anos); %obtenção da matriz D
X = [T D]; %Monto a matriz X

gama = inv(transpose(X)*X)*transpose(X)*transpose(Z); % obtenho gama

figure(1)
plot(st,'r')%plotagem da série original com todos os pontos
legend('plotagem da série completa', 'Location','northoutside','Orientation','horizontal')
Zest = gama'*X'; % Obtenho meu Z estimado
figure(2)
plot(Z,'r');hold on;plot(Zest, 'g'); %Ploto na mesma figura a serie original(em verde) e a série estimada(em vermelho)
legend('15 primeiros anos da minha série','gráfico do meu modelo','Location','northoutside','Orientation','horizontal')
figure(3)
residuo = Z-Zest; plot(residuo, 'b');%gráfico do resíduo
legend('gráfico do meu reśiduo','Location','northoutside','Orientation','horizontal')
erro = mean((log(Z)-log(Zest)).^2)
figure(4)
plot
hist(residuo);%histograma do meu resíduo
legend('histograma do meu resíduo','Location','northoutside','Orientation','horizontal')
mean(residuo);

% aqui eu vou fazer a minha previsão.
T2 = minimos(204,N);
D2 = obtemD(17); %obtenção uma nova matriz D
X2= [T2 D2]; %Monto a=uma nova matriz X
Zest2 = gama'*X2'; % Obtenho meu Z2 estimado

figure(6)
plot(st,'r');hold on;plot(Zest2(1:180),'B');plot( [181:201],Zest2(181:201),'g')
%ploto a serie, a serie estimada e a previsão
legend('Parte da série utilizada na modelagem','Modelo criado','previsões do modelo','Location','northoutside','Orientation','horizontal')
```

Função que testa erro medío quadrático nos polinômios de orden de 1 a 10

```
st = transpose(importfile('STP-20161121152409392.csv', 2, 202)); % importa dados da minha Série temporal
Z = st(1:180); %Parte da série que será utilizada para modelagem.
anos = 15; %anos utilizados para modelagem
D = obtemD(anos); %obtenção da matriz D
```

```
for N=1:8
   T = minimos(length(Z),N); % Obtenção da matriz T
   X = [T D]; %Monto a matriz X
   gama = inv(transpose(X)*X)*transpose(X)*transpose(Z); % obtenho gama
   Zest = gama'*X'; % Obtenho meu Z estimado
   erro(N) = log(mean((Z-Zest).^2))
   T=0;X=0;gama=0;Zest=0;
end
plot(erro);
legend('Gráfico do log(erro) X ordem d polinômio de aproximação','Location','northoutside','Orientation','horizontal')
[valorMinimo ordem]= min(erro)
```

Função que gera matriz D

```
function D = obtemD(anos)
   D = [eye(11,11);-1*ones(1,11)];
   if anos>1
      for i=2:anos
         D = [D;[eye(11,11);-1*ones(1,11)]];
      end
   end
end
```

Função que gera matriz T

```
function T = minimos( tamanho, n )
   T = ones(tamanho, 1);
   T(1:tamanho,2)= 1:tamanho;
   if n>1
      for i=2:n
         T(1:tamanho,i+1)= (T(:,2)).^i;
      end
   end
end
```

Função que impota dados de meu arquivo .csv

```
function st = importfile(filename, startRow, endRow)
%IMPORTFILE Import numeric data from a text file as a matrix.

%% Initialize variables.
delimiter = ';';
if nargin<=2
   startRow = 2;
   endRow = 202;
end

%% Read columns of data as strings:
% For more information, see the TEXTSCAN documentation.
formatSpec = '%*s%s%[^\n\r]';

%% Open the text file.
```

```
fileID = fopen(filename,'r');

%% Read columns of data according to format string.
% This call is based on the structure of the file used to generate this
% code. If an error occurs for a different file, try regenerating the code
% from the Import Tool.
dataArray = textscan(fileID, formatSpec, endRow(1)-startRow(1)+1, 'Delimiter', delimiter, 'HeaderLines', startRow(1)-1, 'ReturnOnError', false);
for block=2:length(startRow)
   frewind(fileID);
    dataArrayBlock = textscan(fileID, formatSpec, endRow(block)-startRow(block)+1, 'Delimiter', delimiter, 'HeaderLines', startRow(block)-1, 'ReturnOnError', false);
   dataArray{1} = [dataArray{1};dataArrayBlock{1}];
end

%% Close the text file.
fclose(fileID);

%% Convert the contents of columns containing numeric strings to numbers.
% Replace non-numeric strings with NaN.
raw = repmat({''},length(dataArray{1}),length(dataArray)-1);
for col=1:length(dataArray)-1
   raw(1:length(dataArray{col}),col) = dataArray{col};
end
numericData = NaN(size(dataArray{1},1),size(dataArray,2));

% Converts strings in the input cell array to numbers. Replaced non-numeric
% strings with NaN.
rawData = dataArray{1};
for row=1:size(rawData, 1);
   % Create a regular expression to detect and remove non-numeric prefixes and
   % suffixes.
   regexstr = '(?<prefix>.*?)(?<numbers>([-]*(\d+[\,]*)+[\.]{0,1}\d*[eEdD]{0,1}[-+]*\d*[i]{0,1})|([-]*(\d+[\,]*)*[\.]{1,1}\d+[eEdD]{0,1}[-+]*\d*[i]{0,1}))(?<suffix>.*)';
   try
      result = regexp(rawData{row}, regexstr, 'names');
      numbers = result.numbers;

      % Detected commas in non-thousand locations.
      invalidThousandsSeparator = false;
      if any(numbers==',');
         thousandsRegExp = '^\d+?(\,\d{3})*\.{0,1}\d*$';
         if isempty(regexp(thousandsRegExp, ',', 'once'));
            numbers = NaN;
            invalidThousandsSeparator = true;
         end
      end
      % Convert numeric strings to numbers.
      if ~invalidThousandsSeparator;
         numbers = textscan(strrep(numbers, ',', ''), '%f');
         numericData(row, 1) = numbers{1};
         raw{row, 1} = numbers{1};
```

```
      end
   catch me
   end
end


R = cellfun(@(x) ~isnumeric(x) && ~islogical(x),raw); % Find non-numeric cells
raw(R) = {NaN}; % Replace non-numeric cells

%% variavel de saida
st = cell2mat(raw);
```