FCTUC FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Análise e Transformação de Dados

Trabalho Prático nº 4

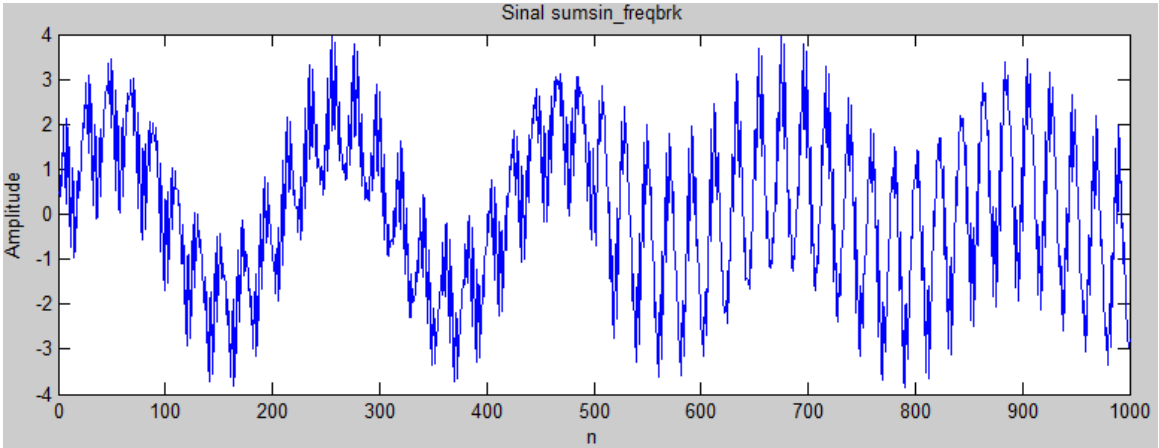
Grupo 23:

João Miguel Rodrigues Jesus nº 2008111667

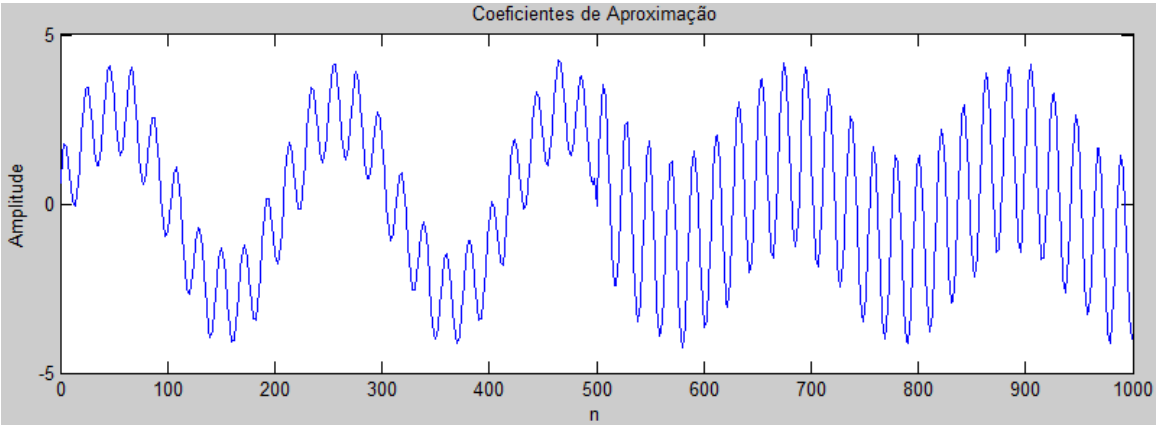
Rosa Manuela Rodrigues de Faria nº 2005128014

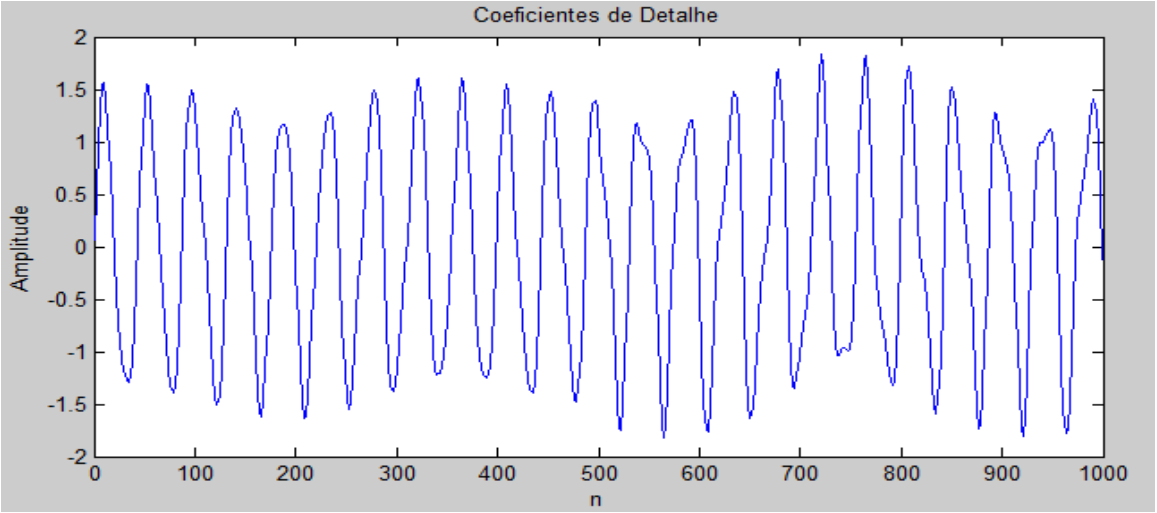
1. O código a que se refere este exercício encontra-se no ficheiro tp4_1.m

1.1. Ao carregarmos a matriz correspondente ao sinal *sumsin_freqbrk* e ao representarmos esse sinal através da função *plot* do MATLAB obtivemos a seguinte representação gráfica do sinal:

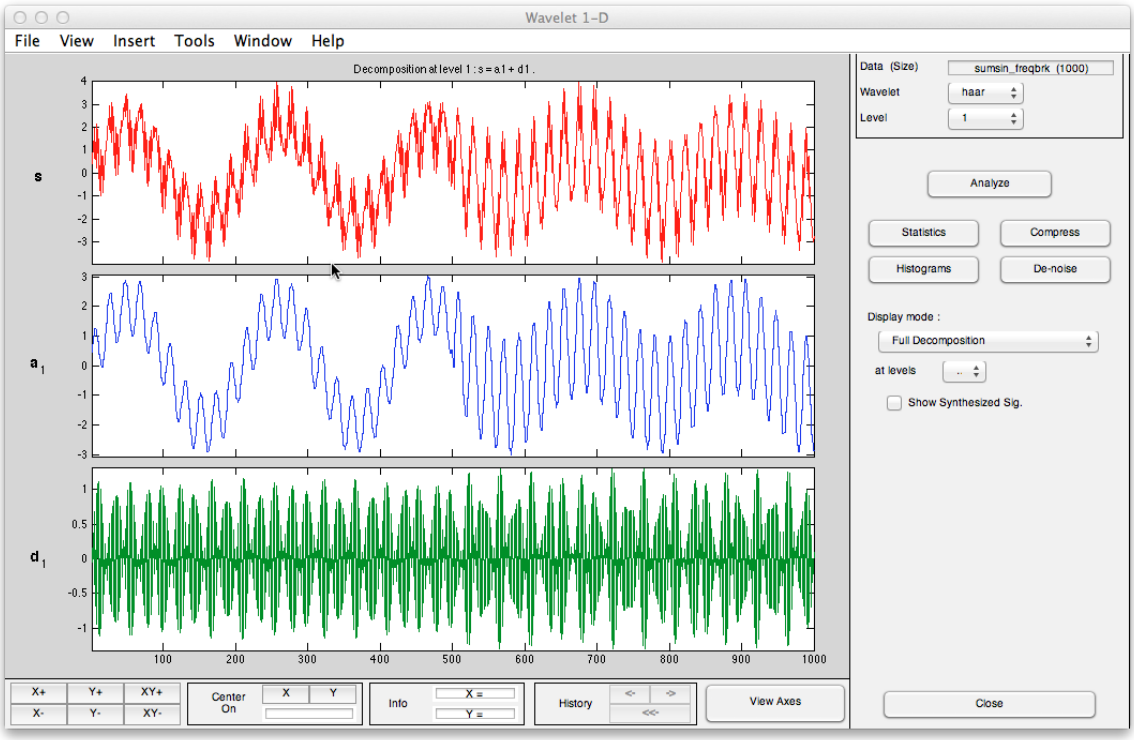


1.2. Neste exercício, para fazermos a decomposição do sinal com um nível de resolução em coeficientes de detalhe e de aproximação, utilizámos a Wavelet da Família de Haar, que corresponde a uma wavelet não continua, simétrica e simples. Para decompormos o sinal no MATLAB utilizámos a função *dwt*, que nos irá devolver os Coeficientes de Aproximação (CA) e os Coeficientes de Detalhe (CD). Ao representarmos esses coeficientes graficamente obtivemos as seguintes representações gráficas:



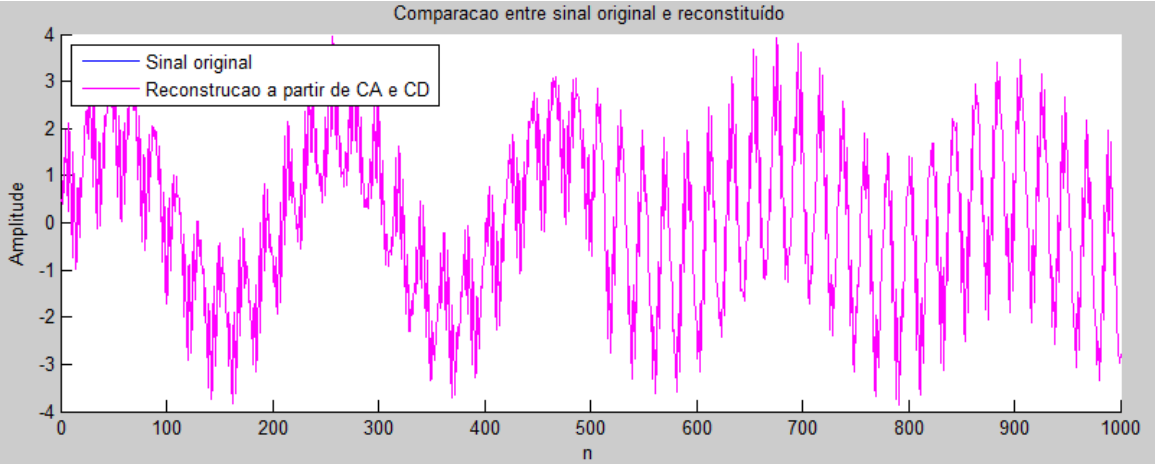


Para confirmar estas representações, utilizámos o plugin de wavelet do MATLAB, através da chamada da função *wavemenu* e escolhemos uma wavelet de 1 dimensão da família Haar e com apenas um nível de detalhe, obtendo os seguintes resultados para comparação:

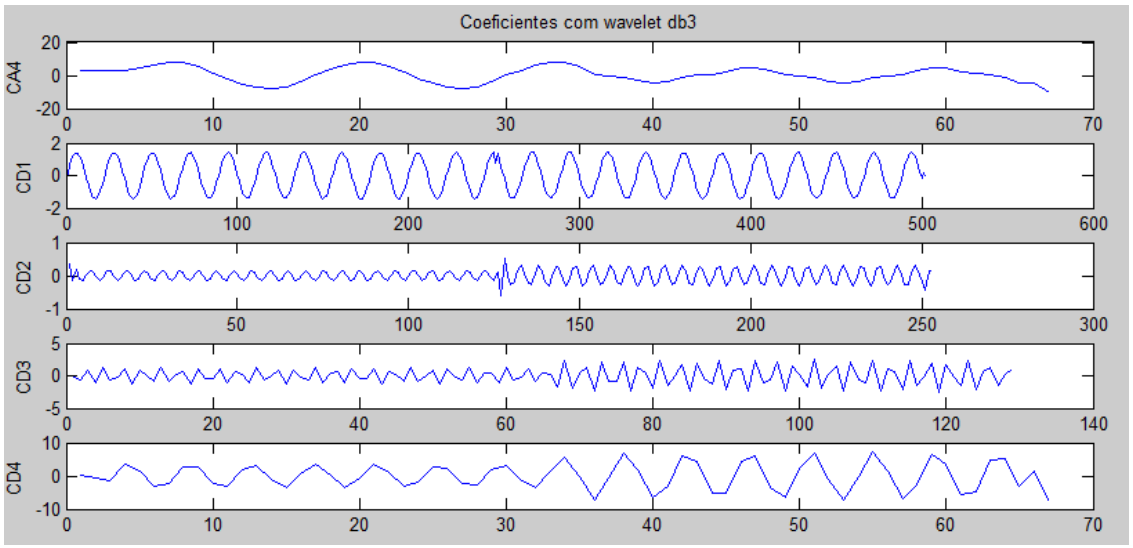


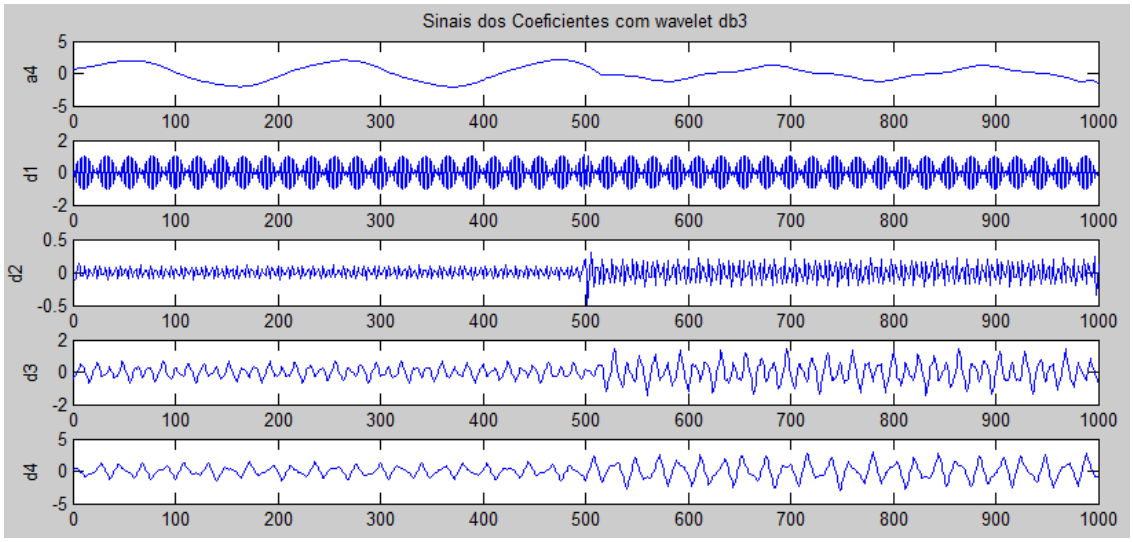
1.3.

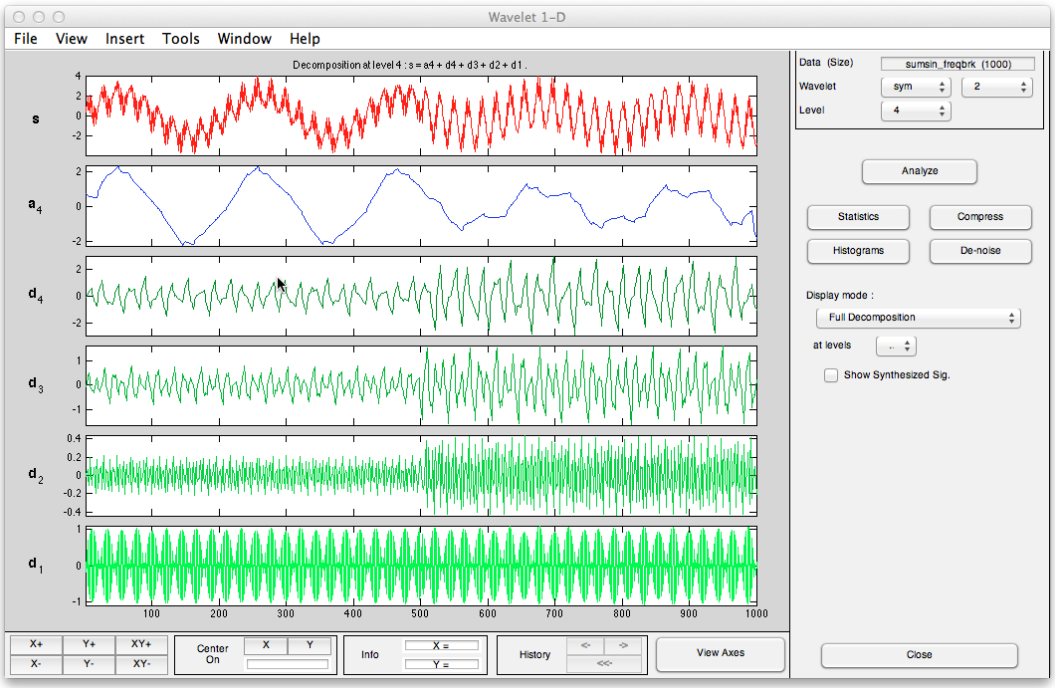
Ao termos decomposto o sinal anteriormente, neste exercício iremos fazer a sua reconstrução a partir dos coeficientes de detalhe (CD) e dos coeficientes de aproximação através da função *idwt* em que teremos de usar a mesma família de wavelet, ou seja, Haar, para a sua reconstrução, já que na sua decomposição foi esta que foi utilizada. Ao representarmos graficamente o sinal reconstruído, iremos obter o sinal que está em baixo representado;



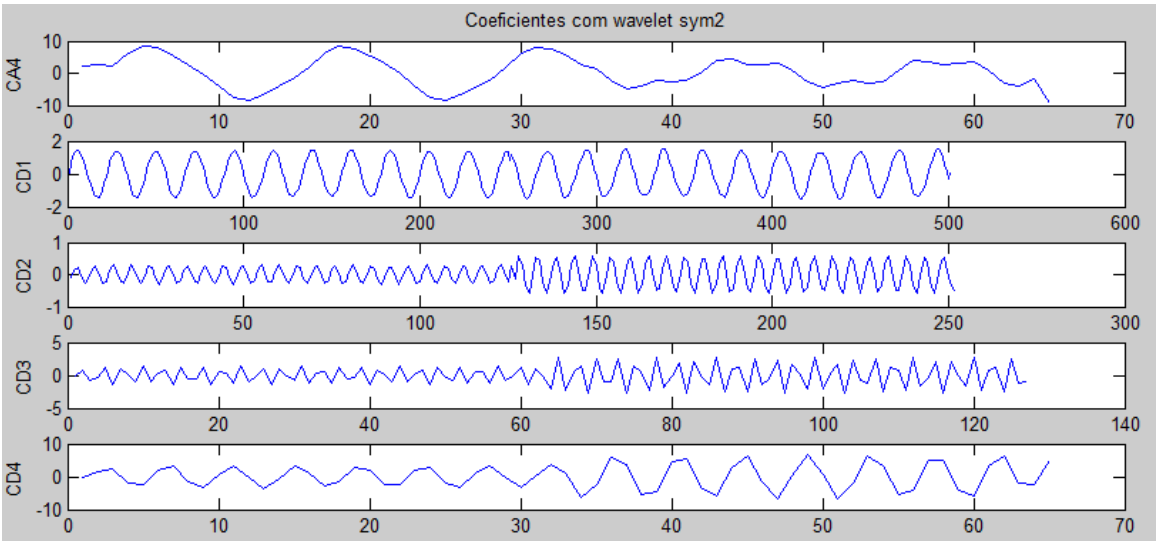
1.4. Neste exercício iremos proceder à decomposição do sinal original, desta vez com 4 níveis de resolução, e utilizaremos a wavelet mais comum da família de Daubechies, que se trata da terceira wavelet de Daubechies (db3). Esta é uma wavelet com pouca simetria para o comprimento dado. Depois de decompormos o sinal através da função de MATLAB *wavedec* iremos obter todos os Coeficientes de Detalhe(CD) e de Aproximação (CA), assim como o tamanho em dois vectores distintos; estes irão ser utilizados para calcular e representar cada nível de detalhe e de aproximação através de uma representação gráfica. Para determinarmos esses coeficientes, utilizámos um ciclo que calcula o coeficiente de detalhe de cada nível e o coeficiente de aproximação, através da função *detcoef*, e a função *wrcoef* para reconstruir o sinal apenas com o coeficiente desejado. Depois de estarem todos os coeficientes calculados, fomos fazer a comparação com os resultados obtidos com a função *wavmenu*, que mostrou que os nossos resultados são o que seria esperado. Obtivemos os seguintes gráficos:

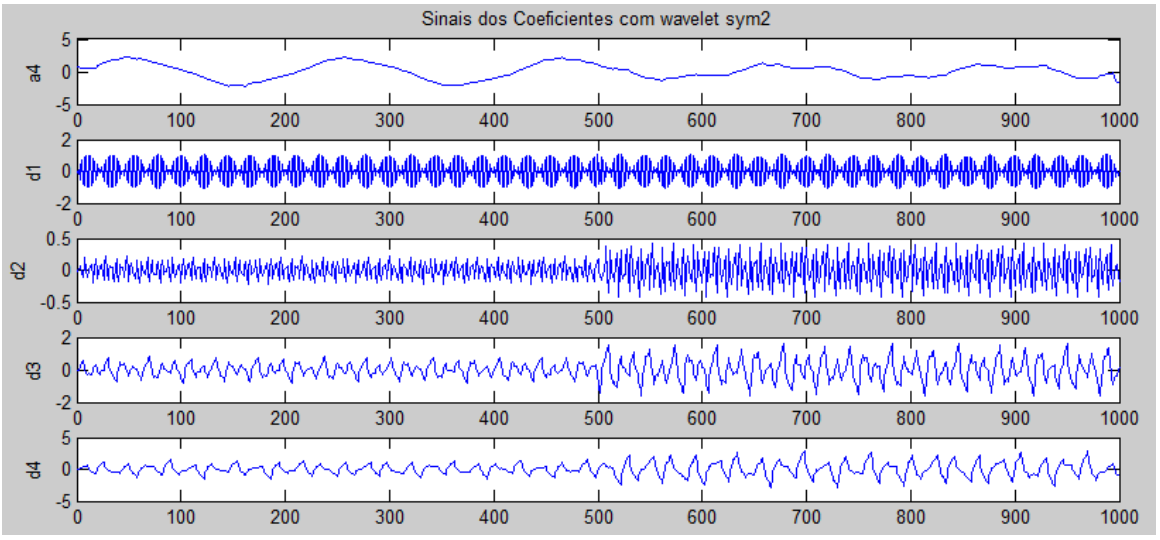

Coeficientes com wavelet db3
CA4
CD1
CD2
CD3
CD4


Sinais dos Coeficientes com wavelet db3
a4
d1
d2
d3
d4

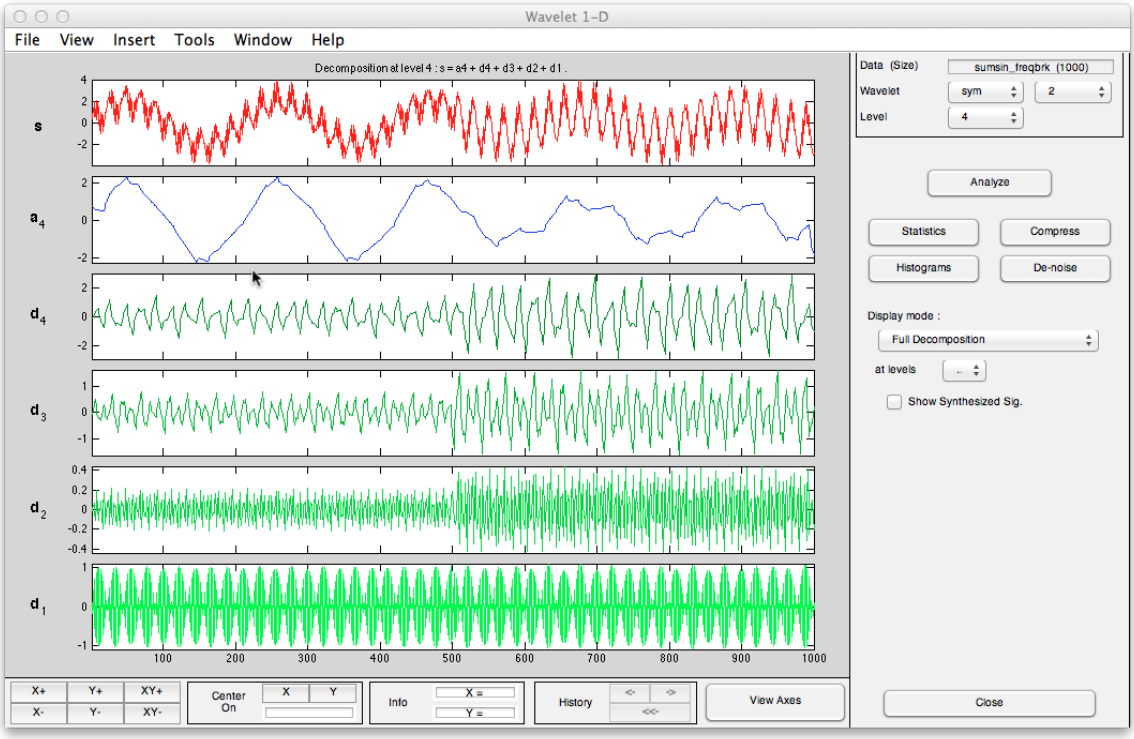

Wavelet 1-D
File View Insert Tools Window Help
Decomposition at level 4 : s = a4 + d4 + d3 + d2 + d1 .
Data (Size)
sumsin_freqbrk (1000)
Wavelet
sym
2
Level
4
Analyze
Statistics
Compress
Histograms
De-noise
Display mode :
Full Decomposition
at levels
Show Synthesized Sig.
Center On
Info
History
View Axes
Close

1.5. Neste exercício iremos proceder novamente à decomposição do sinal original, também com 4 níveis de resolução, mas utilizaremos a segunda wavelet de Symlet (sym2). Através do mesmo procedimento realizado na alínea anterior obtivemos:

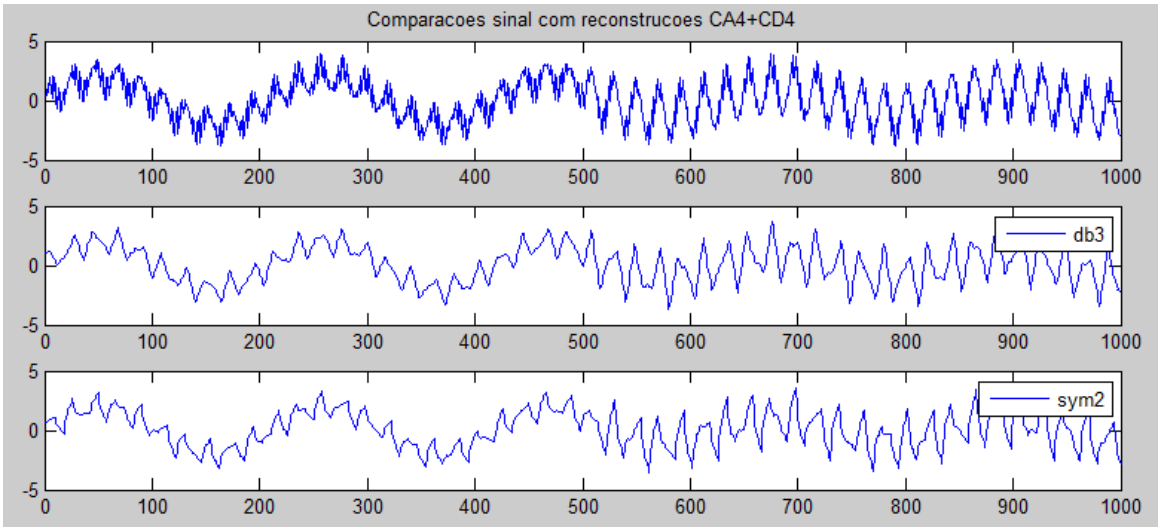




Comparando mais uma vez com a função *wavemenu*, vemos que obtivemos os resultados pretendidos.

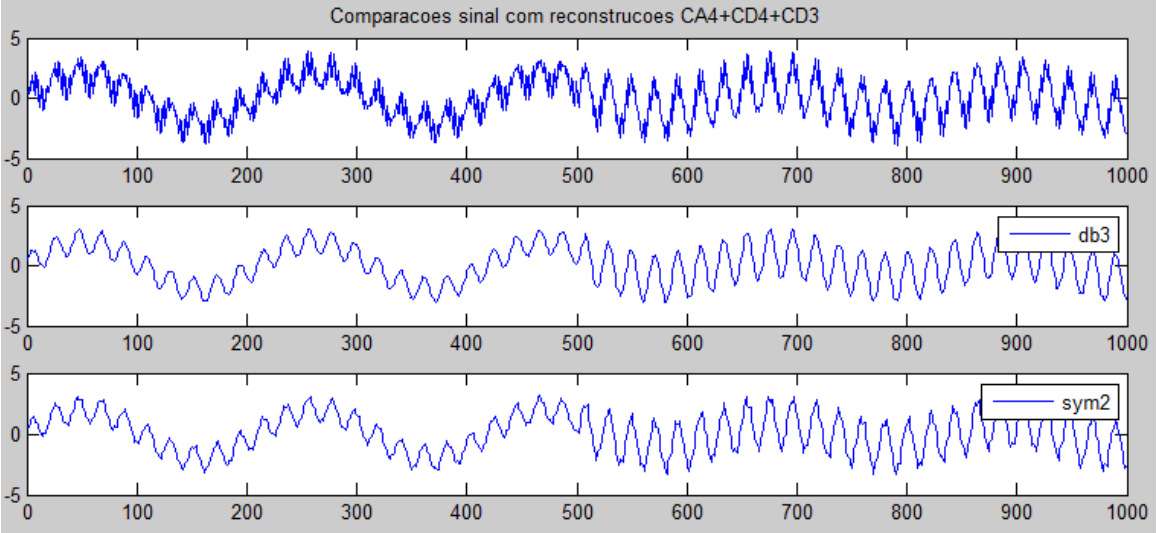


1.6. Para fazermos a reconstrução das duas wavelets consideramos primeiro o coeficiente de Aproximação de nível 4 (CA4) e o de Detalhe (CD4), em que obtivemos as seguintes representações gráficas:

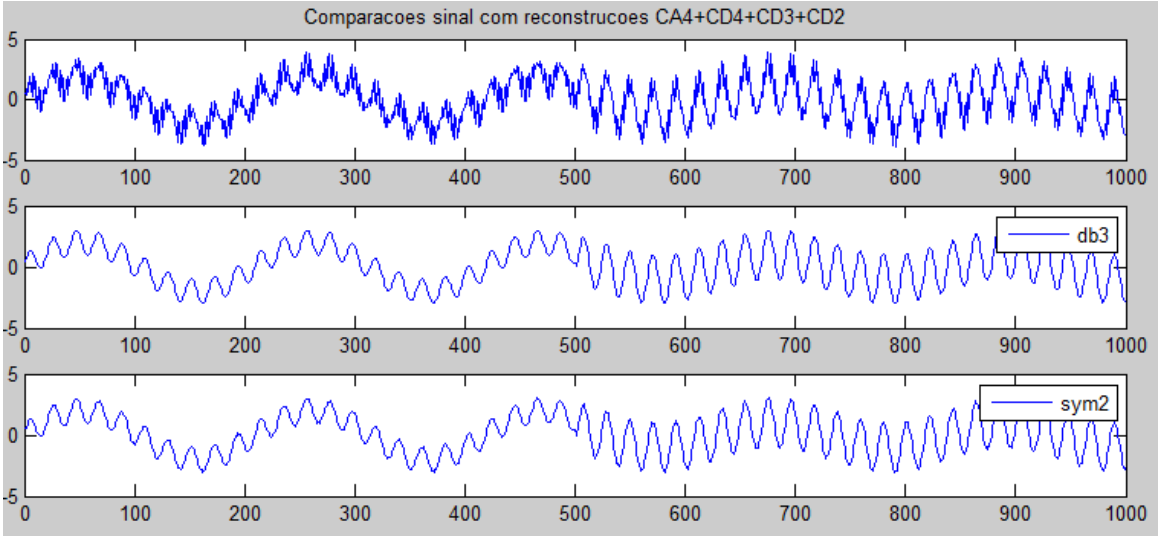


Como podemos ver o sinal reconstruído apenas com estes coeficientes de nível 4 utilizando as duas wavelets apresentam distorções e algumas diferenças tanto a nível de amplitude como de frequência em relação ao sinal original. Vemos também que há diferenças entre os coeficientes das duas wavelets.

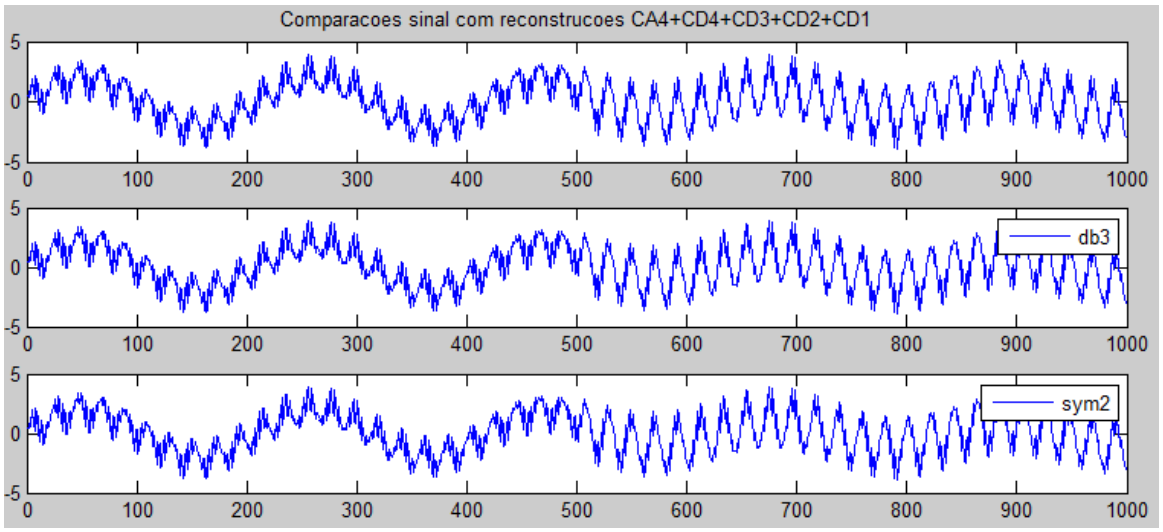
A seguir adicionaremos mais um nível de coeficiente de detalhe que corresponde ao 3 nível, que embora esteja significativamente melhor em relação ao anterior, contém ainda falhas e distorções em relação ao sinal original.

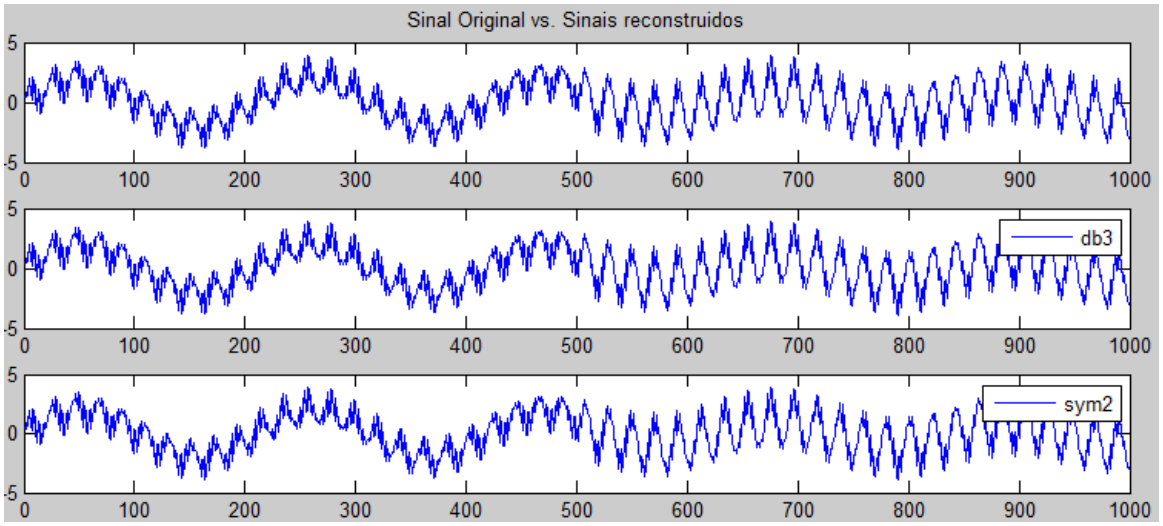


Adicionando mais um nível de coeficiente de detalhe, neste caso o CD2, já se nota um ligeiro melhoramento em ambos sinais reconstruídos em relação ao original, havendo novamente quase nenhuma diferença entre o db3 e o sym2.



Por fim, se adicionarmos todos os coeficientes, ou seja CD1+CD2+CD3+CD4+CA4, iremos obter a reconstrução do sinal original, sem quaisquer diferenças visíveis, tal como seria esperado.

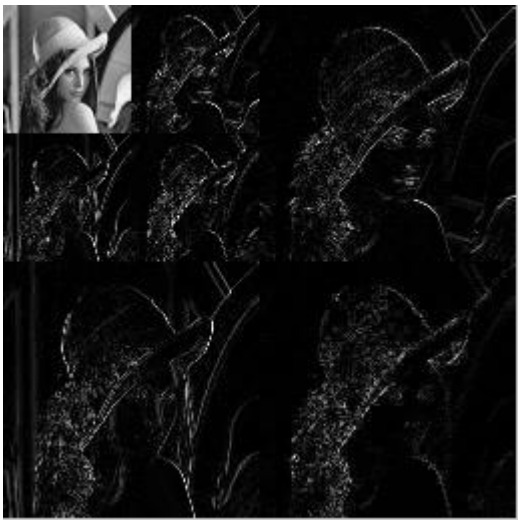

Comparacoes sinal com reconstrucoes CA4+CD4+CD3+CD2+CD1
db3
sym2


Sinal Original vs. Sinais reconstruidos
db3
sym2

2. O código respeitante a este exercício encontra-se no ficheiro tp4_2.m

   2.1. Para decompor a imagem foi utilizada a função wavedec2, tal como sugerido, após a leitura da imagem. De seguida foram determinados os coeficientes dessa mesma decomposição com as funções appcoef2 e detcoef2, à semelhança do exercício anterior. Fazendo um imshow das imagens obtidas através duma matriz com todas as decomposições obtivemos o seguinte:

Dada a dificuldade em interpretar os resultados desta forma, às matrizes correspondentes à decomposição foi aplicada a função *wcodemat*, permitindo a ligação entre os valores dessa matriz e os valores do mapa de cores, obtendo uma nova imagem mais facilmente interpretável. Nesta podemos verificar o que já era esperado: o canto superior esquerdo, onde se encontra a imagem produzida com o coeficiente de detalhe, é o mais semelhante à imagem original, dado que corresponde a um filtro passa-baixo, que apenas esbate os contornos, que correspondem às frequências mais altas. Assim, nas restantes imagens dentro desta figura, podemos apenas observar os contornos da imagem, por serem correspondentes aos coeficientes de detalhe, e portanto, às frequências mais altas, já que estes são filtros passa-alto, e que as frequências mais altas correspondem a transições mais bruscas.



2.2. Para podermos reconstruir a imagem através dos coeficientes de aproximação e de detalhe obtidos com a *wavelet* de Haar, aplicámos a função *wrcoef2* sucessivamente aos resultados obtidos no início da alínea anterior, tendo como parâmetros o coeficiente que pretendíamos calcular (aproximação, horizontal, vertical ou detalhe) e o nível (2 ou 1). Os coeficientes de nível 2 foram depois somados de forma a obter a seguinte imagem:

De seguida foi somada a imagem anterior aos coeficientes de nível 1, e no final a imagem resultante à aplicação da função *waverec2*:

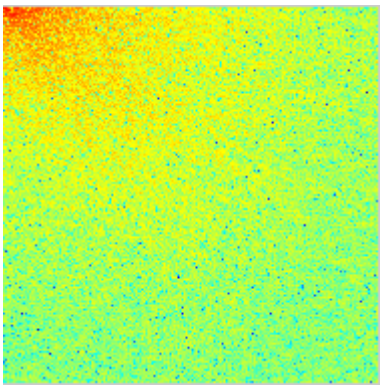
Observando todos estes resultados podemos ver que utilizando os coeficientes de nível 2 se obtém uma representação razoável da imagem original, que, embora pixelizada, nos permite distinguir perfeitamente os detalhes da imagem, sendo um bom modo de compressão da imagem.

Ao somar todos os coeficientes, podemos ver que a imagem obtida é extremamente parecida com a imagem original, não nos sendo possível identificar quaisquer diferenças a olho nu. Considerando a imagem obtida pela função waverec2 acontece precisamente o mesmo fenómeno, não sendo possível encontrar diferenças entre esta, a imagem obtida com todos os coeficientes e a imagem original.

3. O código respeitante a este exercício encontra-se no ficheiro tp4_3.m, sendo que, para a alínea 3.5, há também código no ficheiro tp4_2.m, no fim do ficheiro. Há ainda uma função PickCoef.m que também pertence a esta resolução.

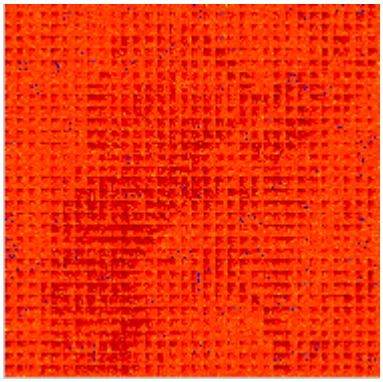
3.1. Para ler a imagem contida no ficheiro lenna.jpg foi utilizada a função *imread*, e para a visualizar a função *imshow*, obtendo o seguinte resultado:

3.2. De modo a representar a transformada DCT da imagem, foi aplicada a função *dct2* à imagem originalmente lida, fazendo-se depois uma transformação logarítmica, e substituindo os valores não-válidos por 0. Finalmente foi representada a imagem:



Esta imagem mostra que, tal como previsto, os coeficientes mais significativos das imagens (a vermelho) se acumulam no início da matriz dos dados, ficando o restante com valores muito pequenos (e portanto pouca informação). Esta "selecção" de irá depois permitir a compressão da imagem, excluindo os coeficientes pouco importantes, e reduzindo o tamanho da mesma.

3.3. Para o processamento de imagens de grande dimensão, calcular os coeficientes DCT das imagens pode ser uma operação demasiado pesada. Embora não seja este o caso, para mostrar a alternativa, foi calculada a DCT em blocos de 8x8 pixeis, utilizando a função blkproc com a opção 'dct2', e obtendo a seguinte imagem:



Vemos aqui que cada bloco tem o seu máximo no início do mesmo, o que permitirá a reconstrução da imagem seleccionando os coeficientes mais significativos de cada bloco, como se verá na alínea seguinte.

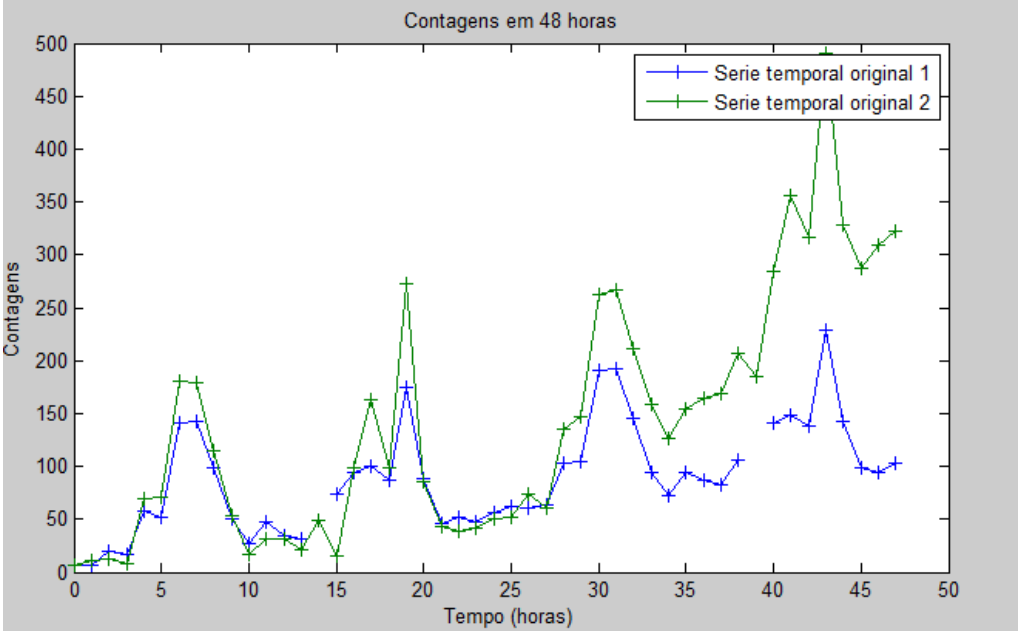
3.4. Nesta alínea é feita a reconstrução da imagem usando um número específico de coeficientes de cada bloco, escolhendo sempre o(s) coeficiente(s) mais significativo(s). Assim, foi criada uma função para fazer esta selecção (PickCoef.m), que para cada bloco calcula o máximo, guarda numa nova matriz, e põe o elemento correspondente a esse máximo da imagem a NaN, de modo a não ser novamente escolhido na iteração seguinte. O número de iterações será o número de coeficientes a usar, dado que em cada iteração é escolhido um. Podemos observar claramente que quanto mais coeficientes são utilizados, mais parecida a reconstrução fica com a imagem inicial, tal como esperado, dado que estamos a incluir frequências mais altas, correspondentes aos detalhes das imagens (e portanto aos contornos).

3.5. Podemos ver que, tanto com a DCT e a DWT, o resultado da reconstrução da imagem com todas as suas componentes é muito semelhante à imagem original, não nos sendo possível, a olho nu, detectar qualquer diferença entre qualquer das reconstruções e a imagem inicial.

Sabemos, no entanto, que a DCT é uma derivada das transformadas de Fourier, e portanto esta não contém informação temporal no seu resultado; já a DWT, que é ideal precisamente para a análise e decomposição de frequências ao longo do tempo, já que nos dá informação tanto relativa à frequência como relativa ao tempo. Este facto não parece, no entanto, interferir na reconstrução de imagens, dado que nestas as frequências apenas variam espacialmente e não temporalmente. No caso de sinais sonoros, por exemplo, já seria bastante melhor usar a DWT em vez da DCT.

4. O código desenvolvido para a realização deste exercício encontra-se no ficheiro tp4_4.m

4.1. Para proceder à representação das séries dadas foram carregados os dados através da função *load*, e fazendo um *plot* do resultado.



Podemos verificar que as 2 séries têm algumas semelhanças (provavelmente devidas à sazonalidade, já que se repetem de 12 em 12 horas aproximadamente), embora, crescendo lineamente ao longo do tempo, uma tenha uma direcção de 1ª ordem, a outra de 2ª.

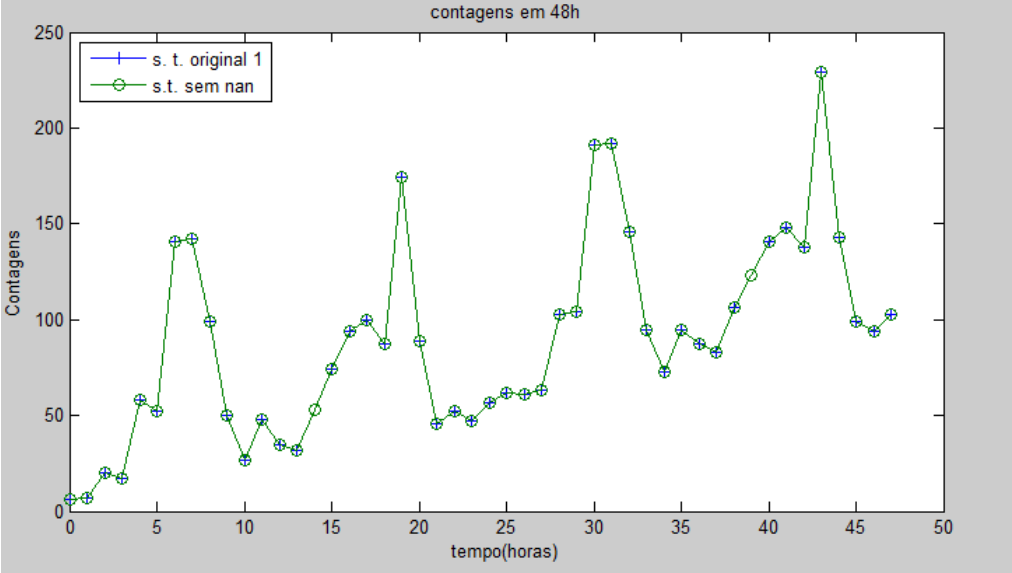
4.2. É possível verificar, tanto pela representação gráfica como pelos dados originais, que a 2ª sequência temporal está completa, tendo a primeira, porém, 2 lacunas. Para normalizar a frequência e completar as falhas temos duas opções:

A primeira, que consiste em apenas eliminar aqueles dados, tem como consequência uma translação do resto dos dados, o que alteraria completamente a análise temporal da série.

A segunda será a interpolação, escolhida neste caso, pois preenche as lacunas consoante o critério escolhido: levando em conta apenas os mais próximos, levando em conta toda a série, etc. Neste caso será usada uma interpolação linear, utilizando todos os dados da série.

Assim, para resolver esta alínea, procura-se saber qual a coluna que contém valores NaN, indo depois procurar os índices específicos, e eliminando-os. Seguidamente é reconstruída a linha, usando a função *interp1*, e feito o *plot*.

Podemos ver na imagem que os 2 pontos que anteriormente eram inexistentes, às 14 e 39 horas, têm agora "contagens" estimadas, fruto da interpolação realizada.



4.3.

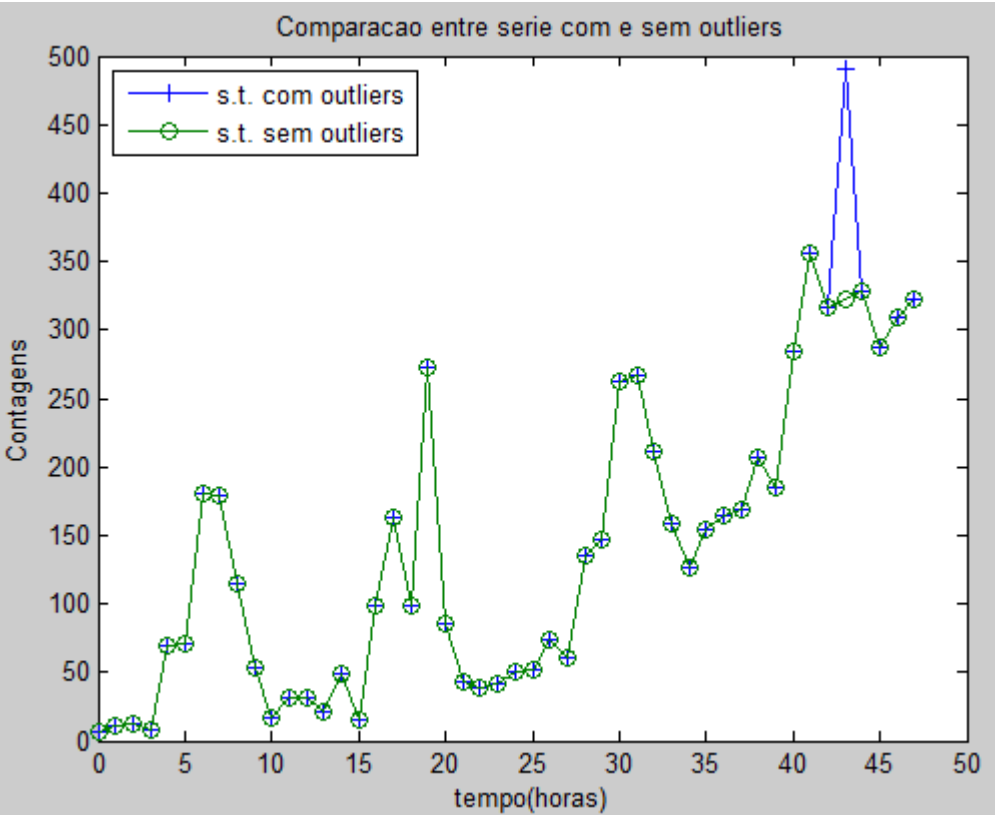
```
Media:
   8.9302e+01   1.4229e+02
Desvio-Padrao:
   5.0477e+01   1.1590e+02
Correlacao:
   1.0000e+00   8.7321e-01
   8.7321e-01   1.0000e+00
Nº de outliers por coluna:
     0     1
```

Dada a matriz de correlação obtida, podemos verificar que as duas séries estão muito próximas, sendo o seu valor de correlação é de 0.87, muito próximo de 1. Se este fosse negativo as séries estariam muito correlacionadas, mas inversamente, ou seja, quando uma crescesse a outra decresceria.

Olhando para o desvio-padrão podemos verificar que, dado que este nos dá a variação em relação à média, no 2º caso há maior dispersão.

4.4. Observando o gráfico inicial é facilmente detectado um *outlier* (um valor anormal provavelmente devido a um erro de contagem). De modo a calcular o mesmo, são feitas 2 matrizes com o tamanho das séries originais, e é calculada a diferença entre a matriz inicial e a matriz com a média. Se a diferença for maior que o triplo do desvio-padrão (valor mais habitualmente usado, mas não a única hipótese), então estamos perante um *outlier*. O procedimento a seguir é agora semelhante ao realizado na alínea

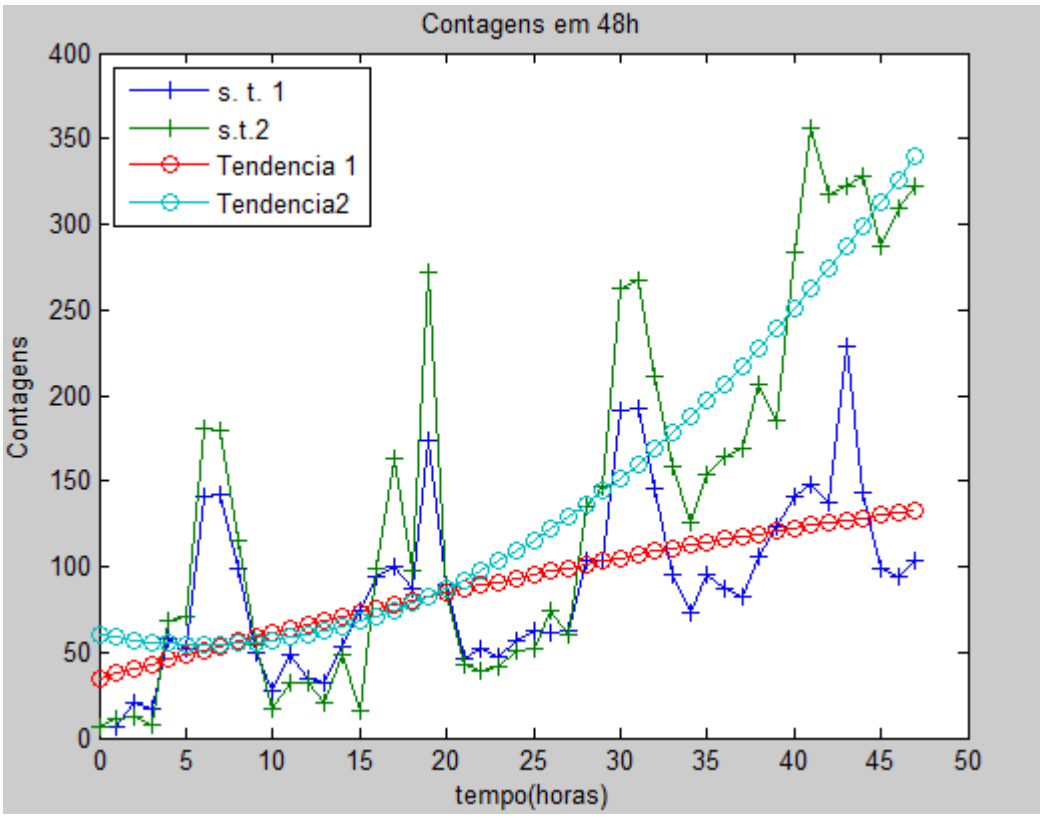
4.2, para interpolar os pontos anteriormente inválidos. Assim sendo, foi obtida a seguinte figura, que nos mostra que às 43h ocorreu um *outlier*, e a respectiva interpolação. Seria de esperar que esta interpolação resultasse num valor mais elevado, sendo o valor dado um valor demasiado baixo tendo em consideração a sazonalidade da série e os elementos anteriores.

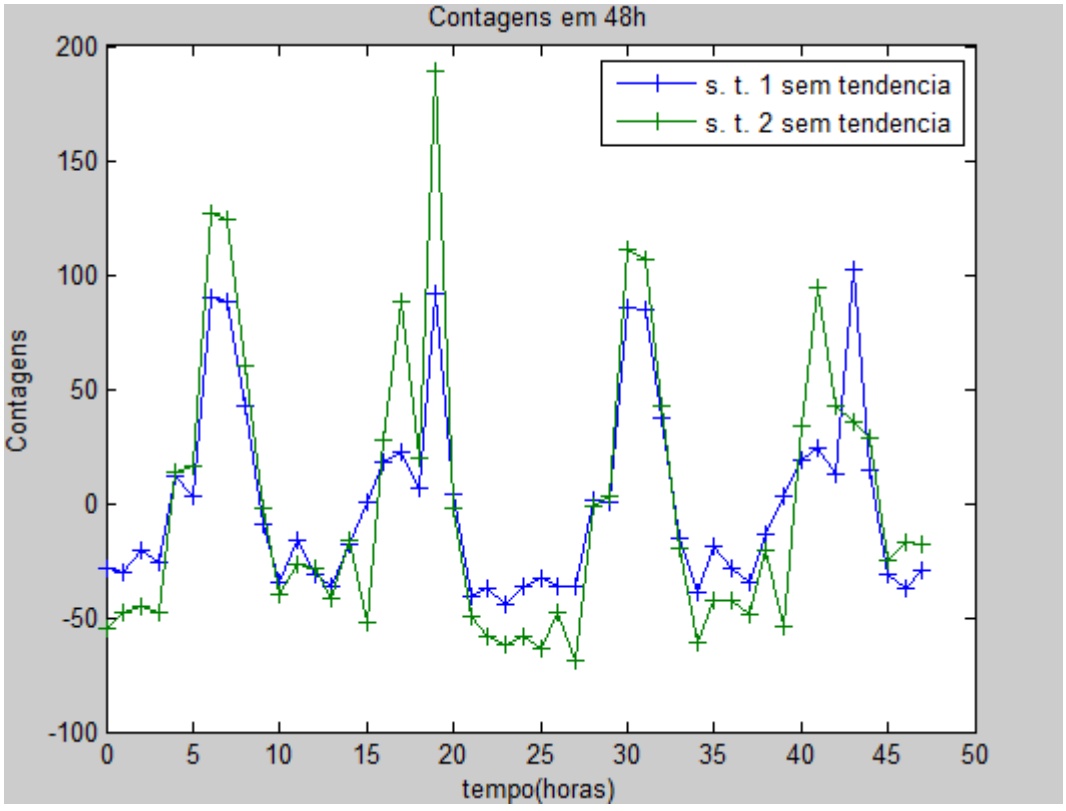


4.5. Sabemos que a componente cíclica de uma série, ou é relacionada com a tendência, ou é menosprezada. Assim, o valor da tendência será o valor em torno do qual esta varia. Calculamos a tendência de cada série temporal usando as funções *polyfit* e *polyval*, que aproximam as séries com polinómios e calculam os valores desses mesmos polinómios em diferentes pontos.

Assim, foram obtidas as imagens que se seguem: na primeira podemos ver a comparação das tendências com as séries originais, e na segunda as séries originais uma vez subtraídas as suas tendências correspondentes.

Nestes gráficos conseguimos verificar que todos os resultados estão de acordo com o esperado, ou seja, que as tendências mostram o crescimento, num caso de 2ª ordem, no outro de 1ª ordem, como referido previamente, e que, uma vez removidas as tendências, as séries apenas variam em torno do eixo dos xx, mostrando ainda mais claramente a relação entre as duas séries e a sua sazonalidade.





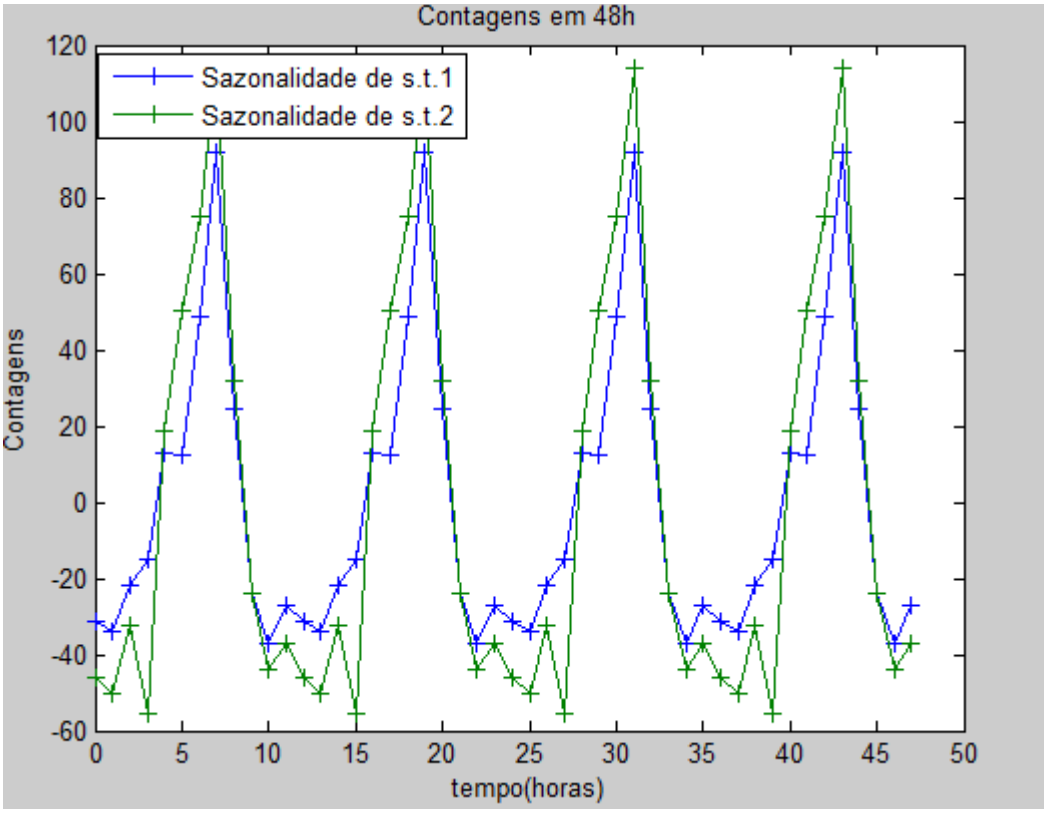
4.6. Ao calcular o modelo temos como objectivo poder prever o que acontecerá no futuro para além dos dados que temos. Tendo obtido as tendências das funções na alínea

anterior pudemos considera-las como modelos, já que, ao calcular os polinómios ajustados, podemos calcular também os seus valores no futuro. Assim, se considerarmos por exemplo a série 1 e lhe aplicarmos a função *polyfit*, obtemos os coeficientes do polinómio que a aproxima: [-1.5126e-02 2.7934e+00 3.4914e+01]. Utilizando um procedimento semelhante para a série 2, obtêm-se os valores [1.7034e-01 -2.0709e+00 6.0685e+01].

Usando agora a função *polyval* para, por exemplo, encontrar os valores para as 72h, vemos que na série 1 obteríamos o valor 1.5763e+02 e para a série 2 o valor 7.9463e+02.

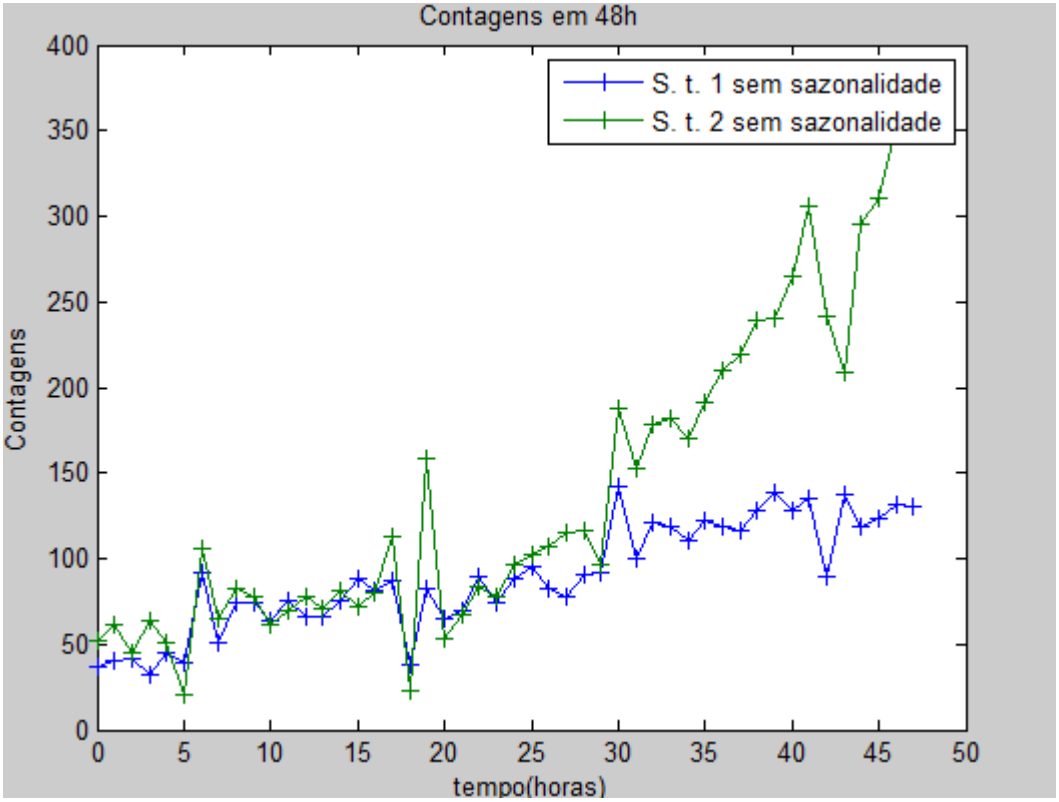
Não podemos esquecer, no entanto, que ao calcular um modelo linear deste modo, ou seja, considerando as tendências das séries como modelo, estamos a ignorar a componente sazonal das mesmas, que podemos ver que tem grande influência em determinados pontos.

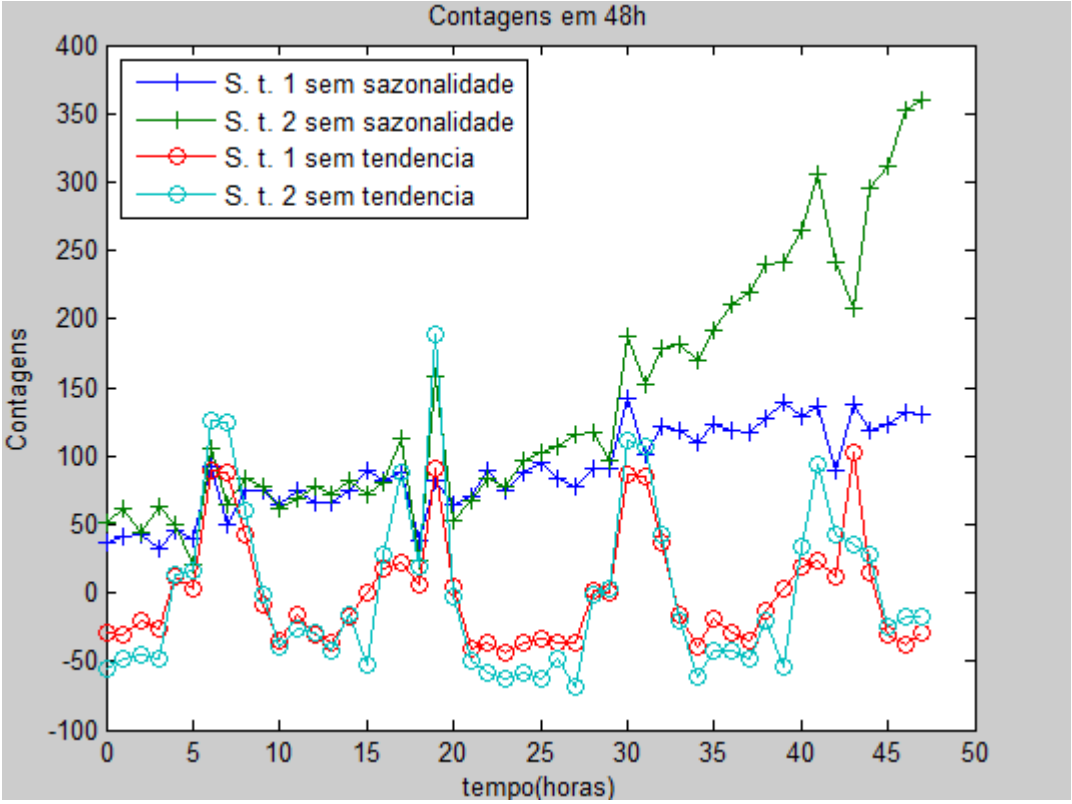
4.7. Nesta alínea considerámos períodos de 12h como sendo os períodos mínimos de repetição do padrão de cada série. Assim, considerando o cálculo da sazonalidade sobre as séries uma vez retiradas as tendências, foi obtido o seguinte gráfico de sazonalidades:



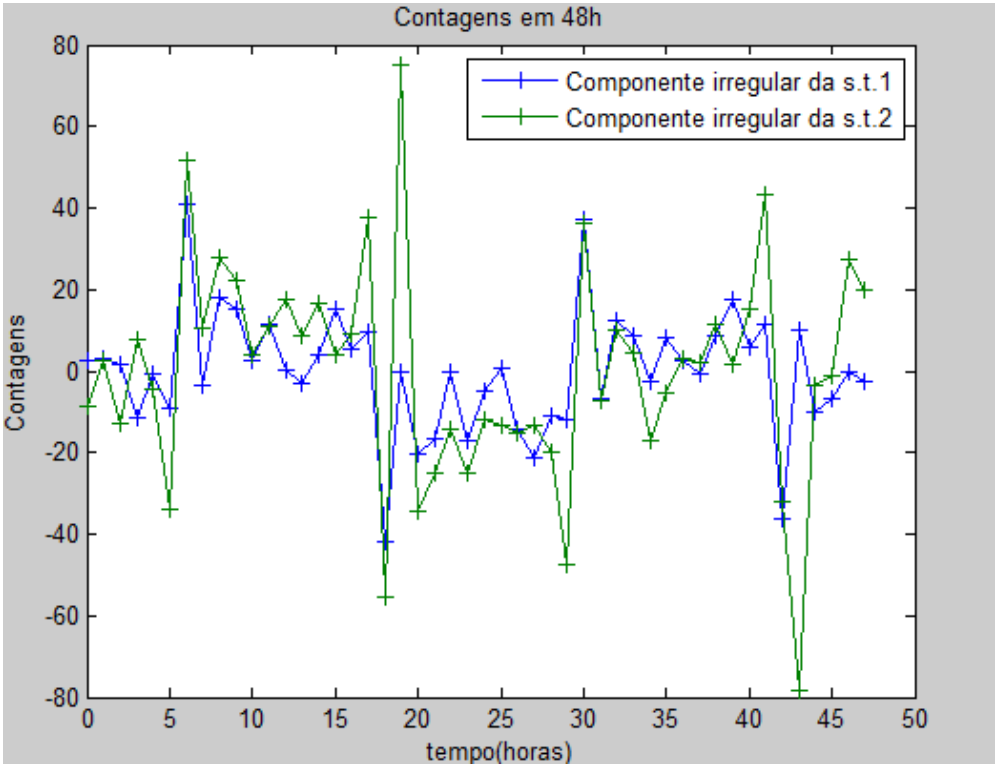
Adicionalmente, foi também feito um gráfico com as séries uma vez subtraída a sua componente sazonal, e outro com a comparação entre as séries sem componente sazonal e as séries sem componente tendencial.

Nestes verificamos que, ao retirar a componente sazonal às séries, estas se tornam bastante idênticas aos gráficos das suas tendências; do mesmo modo, ao subtrair as tendências às séries originais, vemos que o resultado é muito semelhante às componentes sazonais. Concluímos assim, facilmente, que as séries como um todo são resultantes da soma da componente sazonal e da tendência, adicionando ainda uma componente irregular, a calcular na próxima alínea.

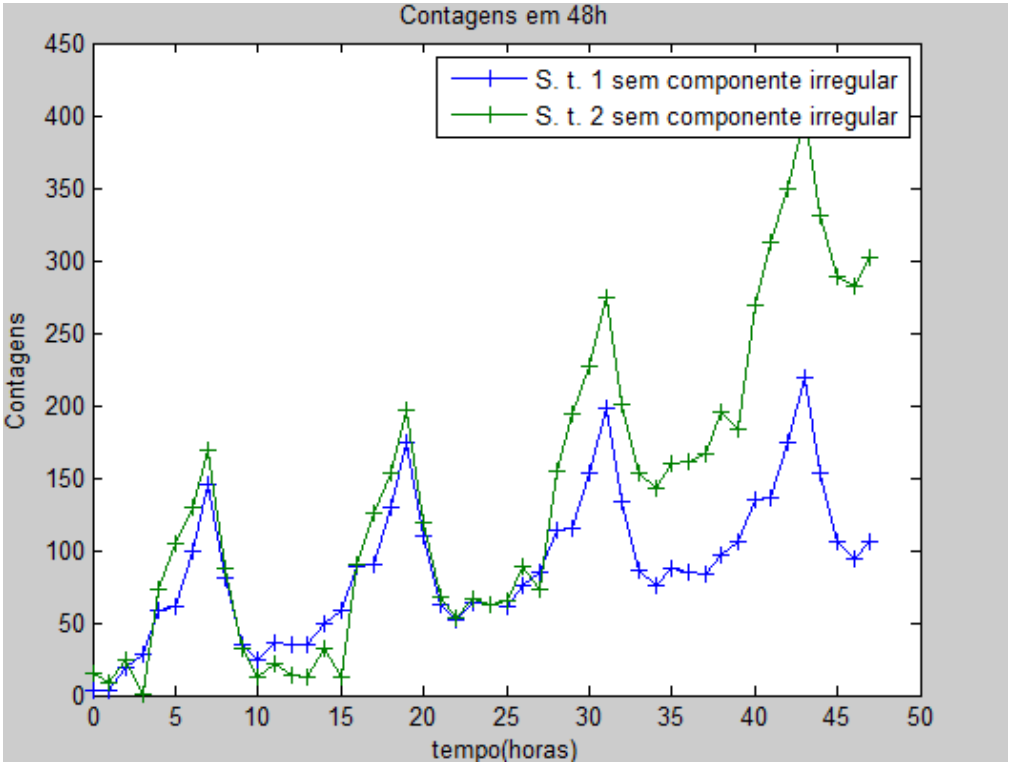




4.8. Tal como mencionado anteriormente, a terceira componente destas séries é a componente irregular. Esta é obtida simplesmente pela subtracção das duas anteriores componentes à série original, obtendo-se o seguinte gráfico:



Se virmos as séries sem a componente irregular é-nos fácil observar que nos restam as outras duas componentes, dado que não se verificam picos a não ser os relativos à sazonalidade.



4.9. Ao aplicarmos um filtro baseado na média móvel vemos que todos os picos são bastante suavizados, sejam estes devidos à componente irregular ou à sazonalidade. Aliás, a componente irregular torna-se impossível de detectar na forma das séries filtradas, sendo os únicos picos visíveis devidos à sazonalidade. Podemos assim ver uma certa periodicidade, e uma menor variação nos valores em relação à média.