0.1 Estrutura PLL Multitaxa

Como vimos na seção referente ao PLL, esse método é capaz de minimizar o erro quadrático médio entre um sinal $y(t)$ e uma componente senoidal para ao menos um mínimo local. Nas figuras seguintes podemos ver como converge o algoritmo em diferentes situações, todas foram simuladas para um sinal de 180 V de amplitude e constantes 300, 500, e 6, com frequência de amostragem igual a 7680 Hz, partindo de condições inicias $f_i = 60Hz$ e $A = 0$. Podemos perceber pela simulação que o algoritmo converge rapidamente mesmo com ruído, entretanto, na presença de harmônicos com a mesma quantidade de energia, a convergência já é bastante comprometida. Ainda assim, em valor médio, a estimação se mostra correta. Podemos concluir que é possível estimar harmônicos diretamente com o algoritmo PLL obtido, entretanto é conveniente aliá-lo a outras técnicas para que se diminua o erro da estimação.

0.1.1 banco de filtros

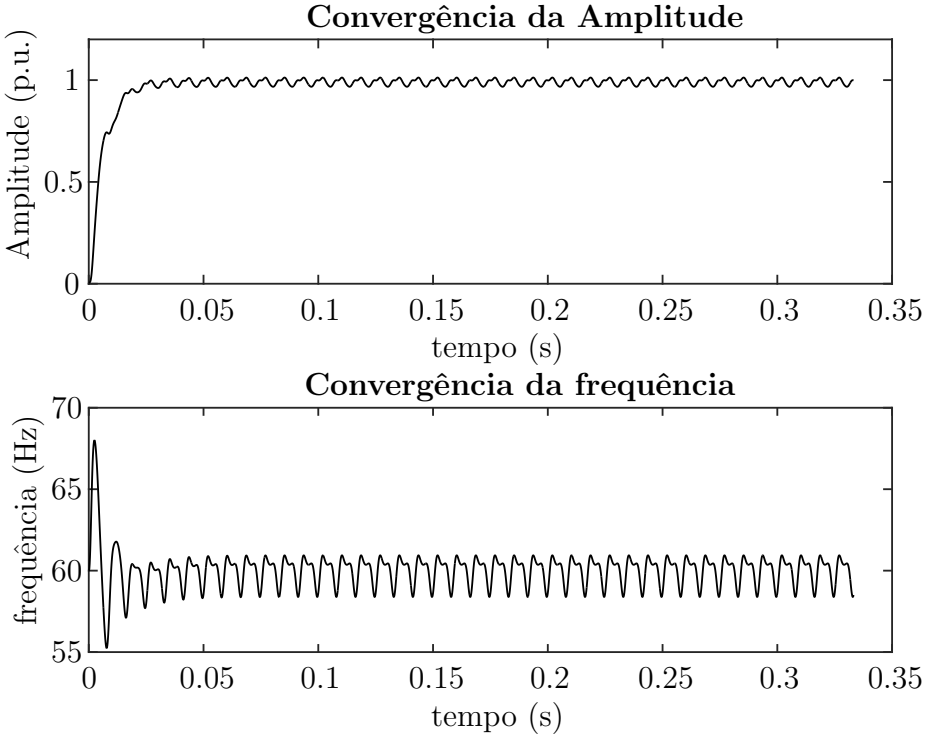

Figura 1: Convergência na presença do 3º e 5º harmônicos; SNR=10 dB

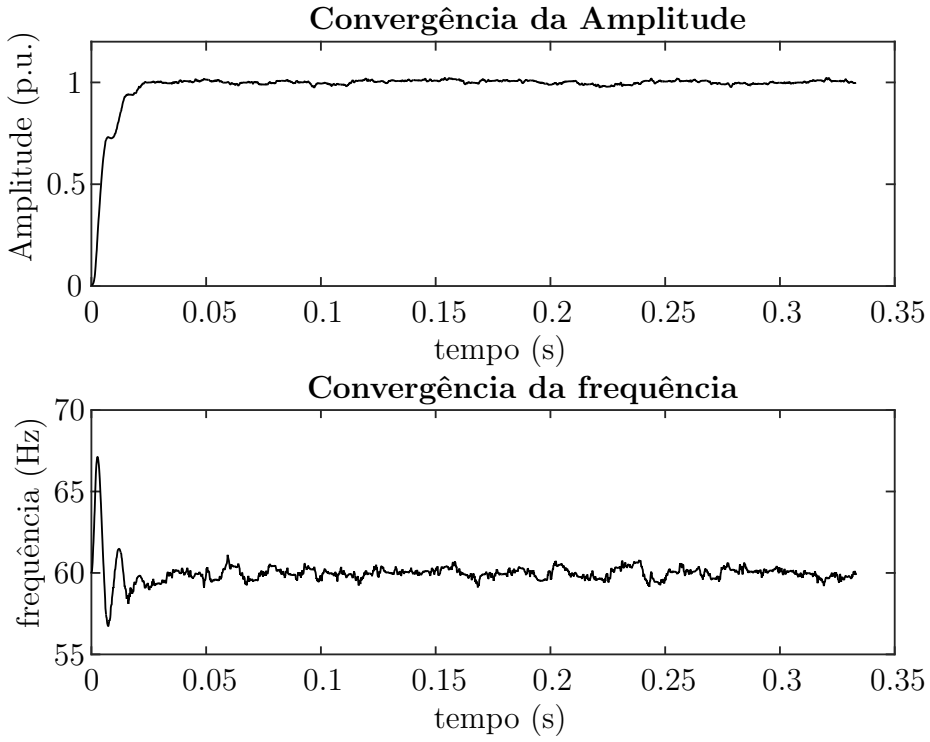

Figura 2: Convergência na presença de ruído; SNR=10 dB

A solução encontrada em [2] é o uso de um preprocessamento com filtros passa-banda, para melhorar a relação sinal ruído, e posteriormente subamostragem, para diminuir a complexidade computacional, de modo também que não seja necessário mudar as constantes do algoritmo.

O conjunto de filtros utilizado é uma cascata de dois filtros IIR com a seguinte função de transferência:

$$H_{bp}(z) = \frac{1-\alpha}{2}\frac{1-z^{-2}}{1-\beta(1-\alpha)z^{-1}+\alpha z^{-2}} \tag{1}$$

$$\beta = cos(w_0) \tag{2}$$

O parâmetro α modifica a seletividade do filtro e está entre 0 e 1, para que este seja estável. Quanto mais próximo de 1, mais seletivo é o filtro. O parâmetro β modifica a frequência central do filtro de acordo com a equação 2, onde w_0 é a frequência normalizada de acordo com a amostragem.

Este filtro é uma boa escolha por alguns motivos:

- Ele rechaça completamente a componente DC do sinal.

- Tem atraso de fase nulo na frequência central.
- É paramétrico, suas características dependem dos parâmetros α e β, os quais modificam propriedades muito específicas do filtro, sendo entAõ muito fácil utilizá-lo e modificá-lo em tempo real.

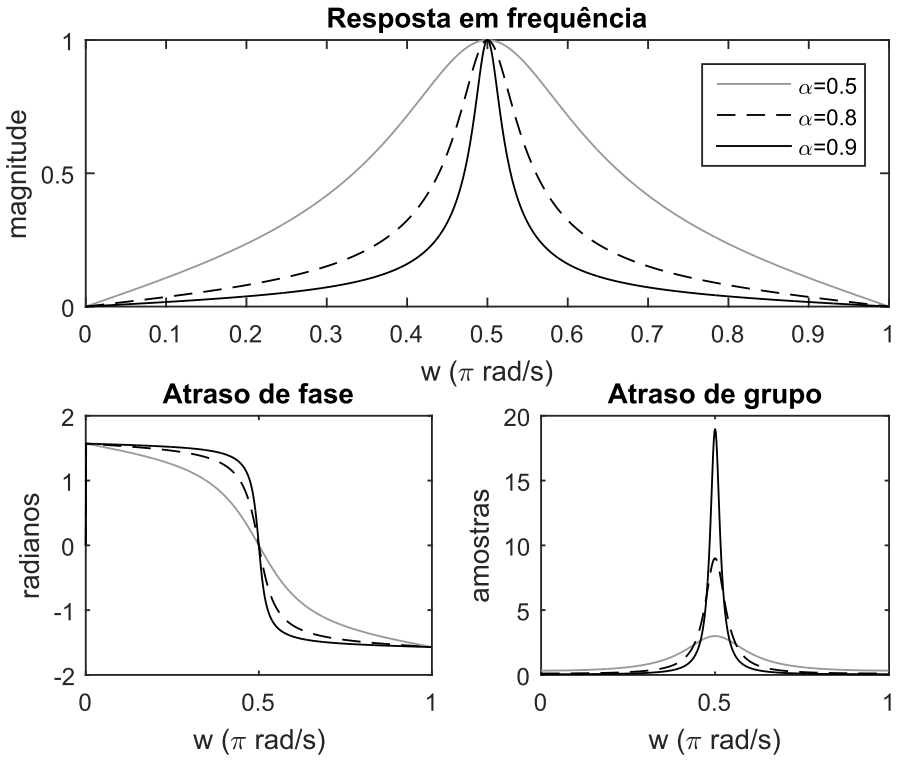

Figura 3: Características do filtro com w_0=0.25

Temos talvez como única desvantagem o atraso de grupo que é máximo para a frequência central, e quanto mais seletivo é o filtro, maior é esse atraso. Devemos também ter atenção com a frequência de amostragem, pois quanto maior, menos seletivo um mesmo α seria. Se por exemplo, mantemos um α e aumentamos f_s, frequências que antes estavam normalizadas mais longe de nossa frequência central anterior, agora estariam mais próximas, por assim dizer. Dessa forma quanto maior é f_s mais seletivo deve ser o filtro para a separação das mesmas frequências. Isso acaba se tornando um jogo de balanceamento destes parâmetros, pois como vimos anteriormente, um α maior também eleva o atraso de grupo, entretanto temos mais amostras em menos tempo devido ao aumento em f_s. Todos estes efeitos estão muito

bem descritos por J. Rodrigues em seu trabalho [13]. Ao final utilizaremos $\alpha = 0.975$ e dois filtros em cascata para o restante das simulações.

0.1.2 Uso da estrutura multitaxa

Depois de passado pelo filtro passa-banda, podemos realizar a subamostragem do sinal, uma vez que em tese eliminamos os harmônicos mais distantes quase que completamente, e estes não interferirão em nossa estimação. Na revisão bibliográfica foi discutido o perfil desta interferência e também como encontrar a posição de uma frequência depois de esta sofrer aliasing. Agora faremos a exposição do algoritmo em C de uma função simples capaz de calcular esta frequência, que pode ser até mais explicativo que o texto anterior:

```
float freq_finder(const float Fs,
                  const float f_init, int *flag){

  float f_aparente=f_init;
  *flag=1; //valor padrao para a variavel

  //verifica se a frequencia investigada sofre aliasing
  if(f_init>(Fs/2)){
    //calcula o resto da divisao entre as frequencias
    f_aparente=fmod(f_init/Fs, 1);
    if(f_aparente<=0.5){
      //se o resto e menor ou igual a 1/2, estamos
      //no semicirculo positivo
      f_aparente=f_aparente*Fs;
    }
    else{
      //caso contrario, estamos no semicirculo negativo
      f_aparente=(1-f_aparente)*Fs;
      *flag=-1;
    }
  }

  return f_aparente;
}
```

A função recebe como argumentos a frequência de amostragem Fs, a frequência rastreada f_init, e o ponteiro para a variável flag, que indica se a frequência encontrada estava no semicírculo positivo, quando vale 1, ou negativo, quando vale -1 ($\theta < \pi$ ou $\theta > \pi$). É importante guardar esta

informação porque necessitamos dela para recuperar a frequência original estimada.

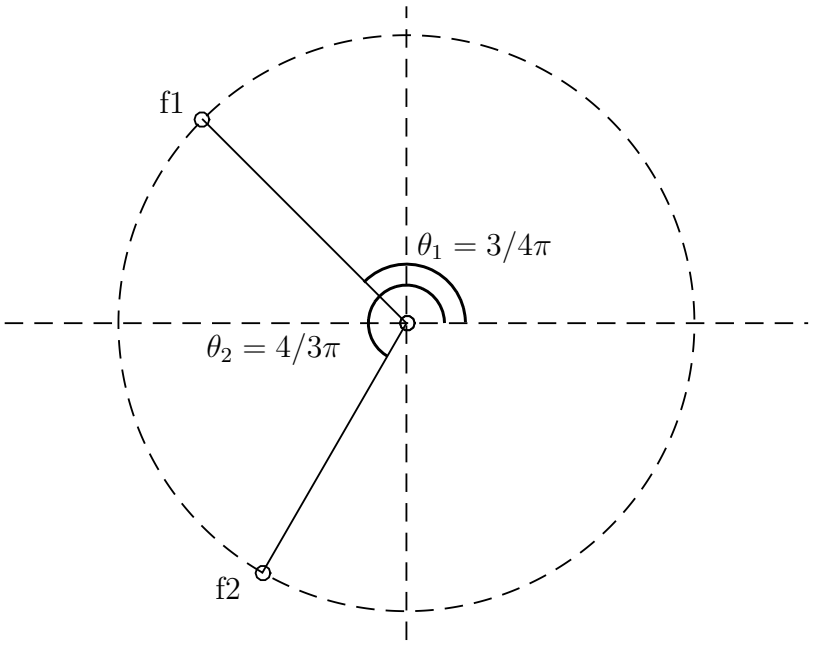

Figura 4: círculo de frequências

Podemos observar na figura 4 a localização de duas frequências f1 e f2 no círculo. Suponhamos que a frequência de amostragem é $f_s = 240\ Hz$, $f_2 = \frac{240}{2}\frac{3}{4}\ Hz = 90\ Hz$ e $f_2 = \frac{240}{2}\frac{4}{3}\ Hz = 160\ Hz$, encontramos facilmente suas colocações no círculo utilizando o algoritmo citado. Agora reparemos que diferentes frequências serão mapeadas nas mesmas posições do círculo, por exemplo $f_1 = \frac{240}{2}\frac{11}{4}\ Hz = 330\ Hz$ também é mapeada no mesmo ângulo, então não temos uma função inversa que devolva as frequências mapeadas para seus valores reais.

Também nos atentemos para o fato de que se aumentamos f_1 levemente, estaremos aumentando sua frequência aparente, mas se fazemos o mesmo com f_2 estaremos diminuindo sua frequência aparente. Por isso é importante guardar a variável 'flag', ela nos diz se acréscimos positivos em frequência aparente condizem à acréscimos ou decréscimos na frequência real, e uma vez que não temos uma função inversa que nos devolva a frequência real dada a aparente, temos que utilizar da variação na estimação aparente e o valor

inicial que mapeamos para recuperar a estimação real.

Uma coisa que devemos ter em mente é que algumas frequências se localizarão muito próximas a zero, terão frequência aparente muito menor que realmente tem. E isso dificulta muito a análise, é desejável testar a localização antes de iniciar o algoritmo e fazer uma mudança no valor de subamostragem caso necessário.

$$\hat{f} = f_{init} + \Delta f_{aparente}\, flag \tag{3}$$

0.1.3 variação da frequência central do banco de filtros

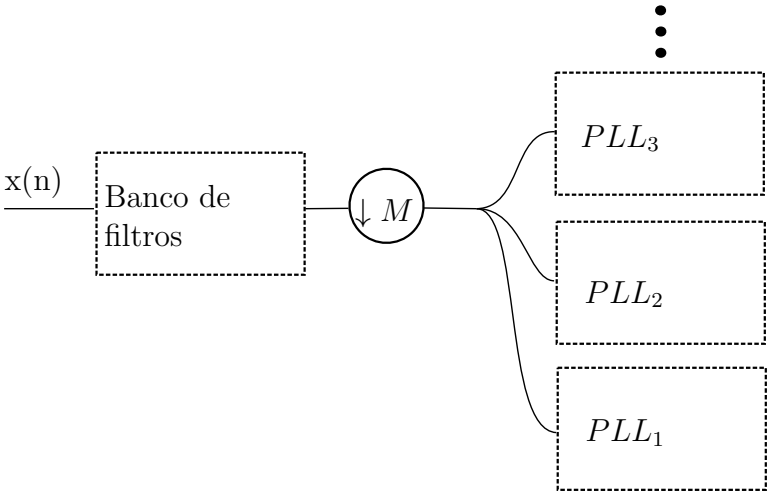

Figura 5: esquema multitaxa PLL

A estrutura multitaxa consiste em passar o sinal de entrada $x(t)$ pelo banco de filtros, sub-amostrar e passar este sinal para os respectivos PLLs. Como o filtro utilizado é bastante seletivo, a frequência estimada deve controlar o banco de filtros centrando a frequência corretamente. A maneira mais básica de realizar este controle seria alimentar o banco diretamente com as frequências obtida no PLL, entretanto este método é inviável. Vimos na seção sobre o algoritmo PLL utilizado que este é altamente não linear, e complexo de se analisar a convergência. Fazer uma realimentação deste tipo muda o sistema e pode torná-lo instável, ou prejudicar sua convergência. A alternativa encontrada por J. Rodrigues [13] é utilizar a média das últimas L estimações, assim o filtro passa-banda se move de maneira mais suave e não prejudica tanto o PLL. Desta maneira:

$$w_0[k] = \frac{1}{L} \sum_{i=k-L+1}^{k} \hat{w}[i] \tag{4}$$

Uma outra opção é fazer a estimação com uma série geométrica, que pode ser calculada de forma recursiva:

$$w_{rec}[k] = w_{rec}[k-1]\lambda + \hat{w}[k] \tag{5}$$

$$w_0[k] = (1-\lambda) w_{rec}[k] \tag{6}$$

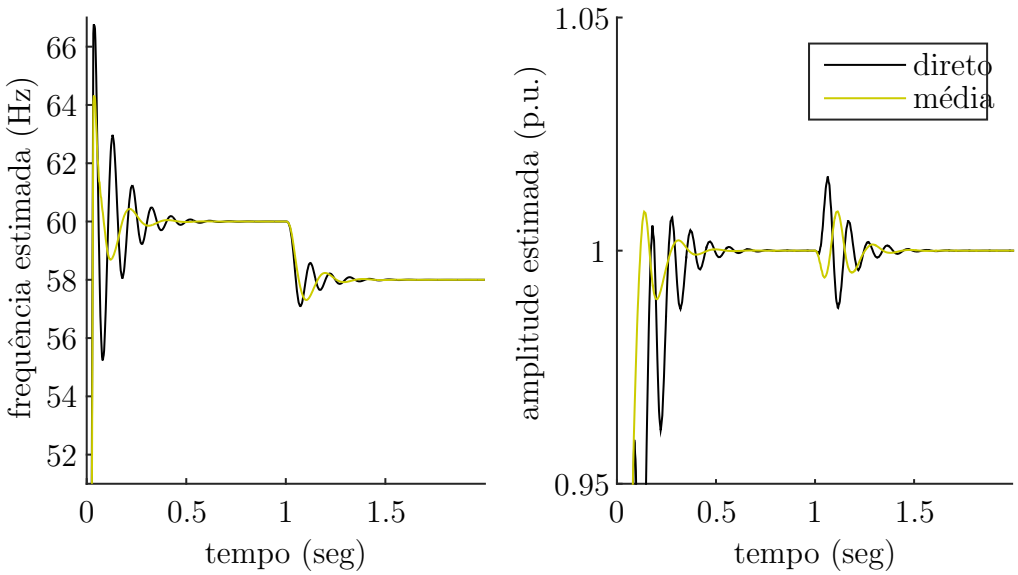

Figura 6: Comparação entre o método de média (com 24 amostras) e o de alimentação direta

Simulando as três opções, a alimentação direta da estimação realmente se mostra mais instável, e com maior tempo de convergência, enquanto que o método geométrico em geral é levemente mais estável e de mais rápida convergência que o de média, além de ser mais fácil de computar. Todos foram testados para as mesmas constantes do exemplo anterior, amplitude de 180 V e 60 Hz iniciais, a constante de subamostragem escolhida foi $M_k = 16$. Depois de 1 segundo de simulação, é aplicado um degrau na frequência de -2 Hz. Também é passado um filtro média móvel de 16 amostras nos resultados finais de $\hat{w}$ e $\hat{A}$.

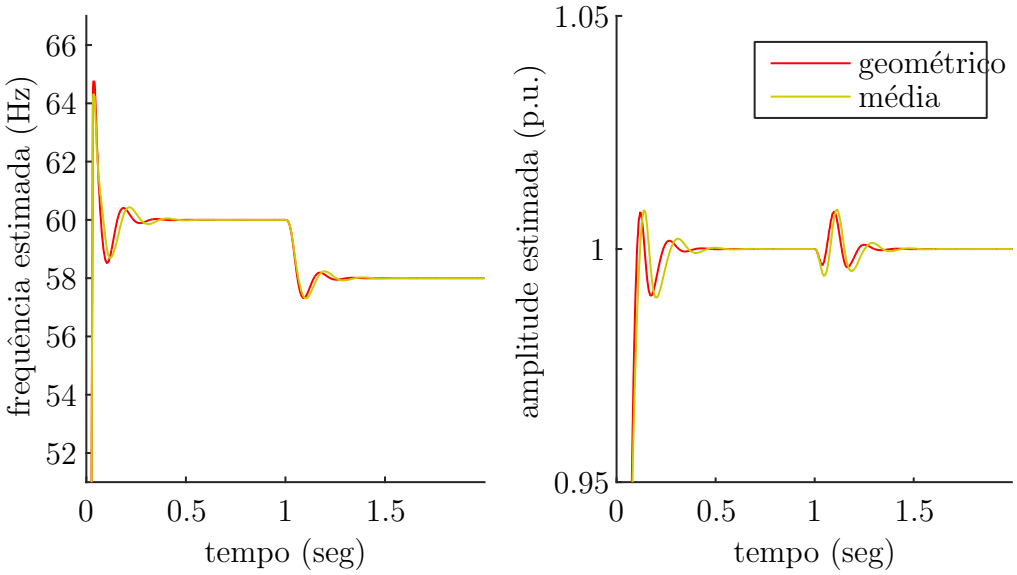

Figura 7: Comparação entre o método de média (com 24 amostras) e o geométrico ($\lambda = 0.9$)

0.1.4 Síntese da estrutura PLL Multitaxa

1. Passamos o sinal por um banco de filtros.
2. Abaixamos a frequência em M_k amostras.
3. Calculamos a frequência aparente f_a da componente a rastrear.
4. Inicializamos o PLL com esta f_a e a guardamos para recompor a original.
5. Fazemos as estimações de w e A, usamos a equação 2 e utilizamos a série geométrica para realimentar o banco de filtros com a frequência calculada.

0.1.5 simulações

Para as simulações computacionais foram seguidos determinados critérios:

- Convergência em amplitude: Foram consideradas as últimas 32 estimações, assim que a média do erro quadrático destas fosse menor que 4e-4 (que é um erro menor que 2% do sinal), consideramos que o algoritmo convergiu.

- O mesmo para a convergência em amplitude, mas com umbral de 1e-4.
- Os cálculos de erro quadrático médio são feitos considerando o valor das estimações depois de 1 segundo. Quando o algoritmo já convergiu.

Os parâmetros da simulação são $f0 = 60Hz$, 128 pontos por ciclo de $f0$, $Fs = 7380\,Hz$, $\alpha = 0.975$ e dois filtros em cascata no banco. A constante de subamostragem é 16, para todos os harmônicos e inter-harmônicos. A amplitude da fundamental é 180 V, e para todos os harmônicos suas amplitudes são a da fundamental dividido pela ordem do respectivo harmônico. Foi passado um filtro média móvel de ordem 16 em todas as estimações de modo a suavizar a visualização dos resultados.

harmônicos ímpares

Abaixo estão os resultados para a simulação com os harmônicos ímpares de 1 a 15. Verificamos algumas coisas:

- o valor de offset para a fundamental é relativamente grande, e isto se explica principalmente pela subamostragem, uma vez que alguns harmônicos não são completamente atenuados se sobrepõe à fundamental. Isso poderia ser melhorado com o aumento da seletividade do filtro, entretanto já discutimos os problemas que isto pode ocasionar. Seu uso ou não depende da natureza do problema em questão.
- o erro em frequência é em geral mais baixo que o de amplitude. E também é seu tempo de convergência.
- o tempo de convergência é difícil de prever, não é certo que os harmônicos de ordem mais elevada ou os de maior energia convergirão mais rapidamente que os demais.
- o algoritmo se comporta muito bem com ruído, principalmente a estimação de frequência.

inter-harmônicos

Foram simuladas estimações com inter-harmônicos, seguindo as mesmas prescrições do caso anterior. Vemos que os tempos de convergência não são muito afetados, entretanto o erro quadrático médio para a fundamental aumenta um pouco. Isto se deve a um leve batimento que o harmônico 3.2 causa na fundamental, fazendo com que a estimação oscile um pouco. Entretanto se olhamos uma média grande o suficiente destas estimações, podemos nos livrar do batimento.

Harmônico	Erro em frequência (%)	Erro em amplitude (%)	Tempo de convergência W (s)	tempo de convergência Amp. (s)
1	6,14E-07	1,344	0,144	0,098
3	2,08E-06	1,427	0,150	0,135
5	2,16E-06	1,338	0,035	0,088
7	3,05E-05	1,441	0,090	0,194
9	1,60E-05	0,913	0,035	0,090
11	4,93E-05	1,146	0,035	0,221
13	2,01E-04	0,136	0,035	0,090
15	4,57E-03	0,126	0,035	0,327

Tabela 1: Tabela para a simulação sem ruído

Harmônico	MSE em frequência (%)	MSE em amplitude (%)
1	4,21E-05	2,32E-04
3	6,05E-07	3,16E-04
5	4,23E-07	5,42E-04
7	3,70E-07	1,01E-03
9	5,56E-07	1,45E-03
11	3,26E-07	1,96E-03
13	7,37E-07	3,65E-03
15	7,76E-07	4,56E-03

Tabela 2: Tabela para a simulação com ruído (σ^2=10)

0.1.6 esforço computacional

Considerando um sinal amostrado em 7380 Hz e uma subamostragem de 16 amostras, realizaremos a estimativa do esforço computacional considerando operações realizadas a cada iteração de soma, multiplicação e funções trigonométricas, sem considerar deslocamentos de buffers e alocação de memória:

Banco de filtros

Cada estrutura correspondente ao filtro apresentado na equação [1] representa 5 multiplicações e 4 somas. Sendo dois filtros, totaliza 10 multiplicações e 8 somas por cada amostra.

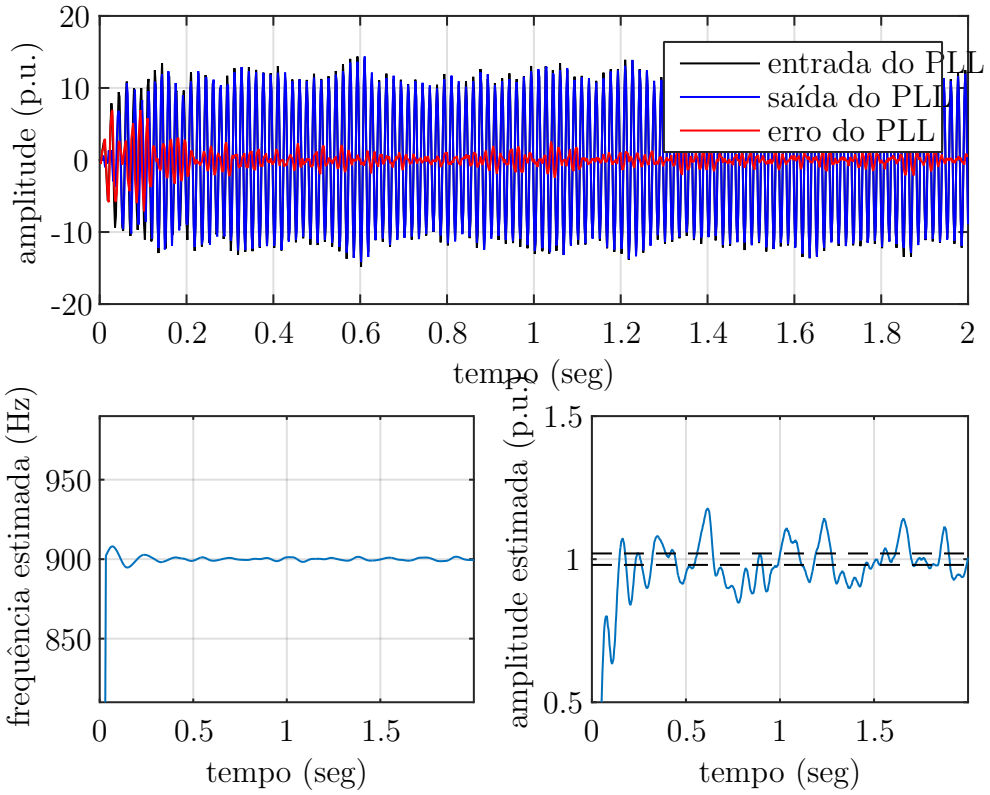

Figura 8: convergência do 15º harmônico na presença de ruído

PLL

Cada estrutura PLL como apresentada na equação [só dá pra citar quando juntar os capítulos] representa 11 multiplicações, 5 somas e 4 trigonométricas, que são executadas a cada 16 amostras.

Atualização de frequência

Cada atualização de frequência custa 2 multiplicações, uma soma e uma trigonométrica.

Totais

Abaixo temos uma tabela com todos os cálculos necessários para 1 segundo de sinal, por cada frequência rastreada. Percebemos que mais de 90% dos cálculos são gastos pelo filtro digital.

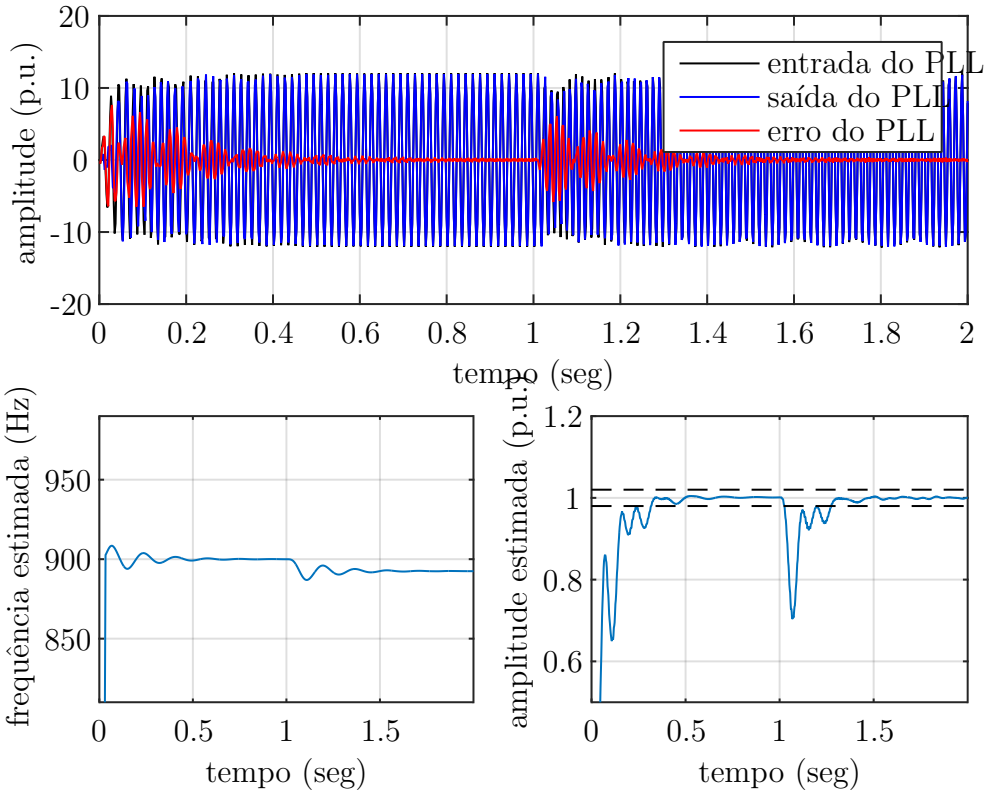

Figura 9: convergência do 15º harmônico com degrau em frequência

Estrutura	Somas	Multiplicação	Trigonométricas
Banco	61440	76800	0
PLL	2400	5280	1920
Atualização	480	920	480
Total	64320	83040	2400

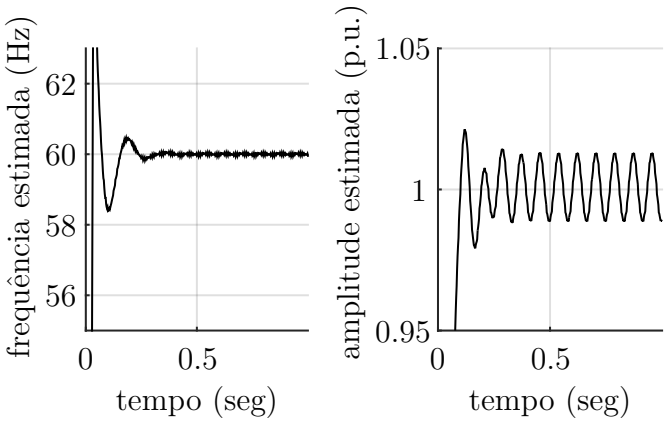

Figura 10: batimento observado na fundamental

Harmônico	convergencia em freq. (s)	convergencia em amp (s)	MSE em frequência (%)	MSE em amplitude (%)
1	0,177083	0,1375	2,91E-05	1,66E-04
3,2	0,179167	0,175	6,73E-08	1,06E-05
6,4	0,114583	0,19375	2,46E-09	3,65E-06
11,3	0,11875	0,352083	5,04E-08	3,59E-06

Tabela 3: Tabela para a simulação de inter-harmônicos

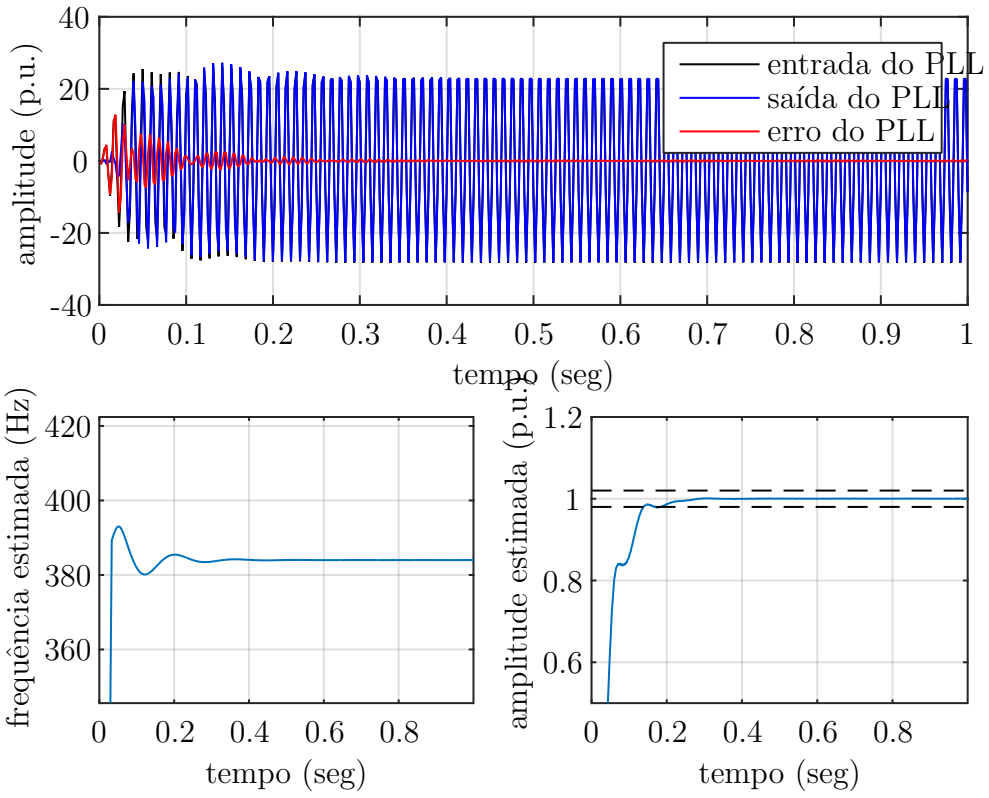

Figura 11: convergência do inter-harmônico 6.4