UNIVERSIDADE FEDERAL FLUMINENSE
PROGRAMA DE PÓS-GRADUÇÃO EM COMPUTAÇÃO
ESTIMAÇÃO DE ESTADO

Luiz Phillip Quintanilha da Silva

Neste relatório será apresentado o usa da aplicação desenvolvida para a Estimação de Estado (EE). O programa contém as principais ferramentas estudadas na disciplina de EE ministrada pelo professor Julio Cesar Stacchini De Souza no segundo semestre de 2020. O progama foi testado no sistema do IEEE 14 Barras para sistemas de transmissão.

A Estimação de Estado é um processo iterativo em que dado um conjuto de medições instaladas um valor e designado para as variáveis de estado do sistema com base em algum critério. Como os erros são inerentes ao processo de medição a redundância torna-se necessária. Na EE para sistemas elétricos de potência é usual minimizar a função de custo conhecida como Mínimos Quadrados Ponderados.

Mínimos Quadrados Ponderados

Conforme mencionado anteriormente a função que se deseja minimizar para a EE é a Mínimos Quadrados Ponderados. Esta função, designada pela letra J está representada na Equação (1)

$$J(x) = \sum_{i=1}^{N} \frac{(z_i - h_i(x))^2}{R_{ii}} \quad (1)$$

onde z_i representa o valor medido, h_i é uma função não linear que relaciona o estado estimado x com a medida e R_{ii} representa a variância da medida.

A Equação (1) difere dos mínimos quadrados comum por considerar a variância da medição. Esta função é considerada uma função de maximização da verossimilhança quando avaliada sobre uma distribuição normal.

É interessante notar que $J(x) \sim \tilde{\chi}^2$. Desta forma, pode-se avaliar a qualidade da EE a partir de um teste de hipótese. Se ao final da estimação a função de custo está acima do valor da $\tilde{\chi}^2$ para um nível de confiança α, então pode-se investigar a presença de dados errôneos ou de erro(s) no(s) parâmetros/topologia da rede. Vale ressaltar que em geral este método não é muito utilizado.

Processo de Estimação de Estado

A EE utiliza as medições disponíveis no sistema para estimar as variáveis de estado mais prováveis da rede. Este processo utiliza a Equação (1) para maximizar a verossimilhança entre as medições recebidas e as estimadas. Para tal tarefa defini-se uma função $g(x)$ que será responsável pela minimização de $J(x)$. Para esta minimização $g(x)$ será dada por:

$$g(x) = \frac{\partial J}{\partial x} = -H(x)^T R^{-1}[z - h(x)] \quad (2)$$

em que $H(x) = \frac{\partial h}{\partial x}$

Expandindo $g(x)$ em série de Taylor e considerando somente o termo de primeira ordem obtêm-se a Equação (3):

$$[G(x^k)]\Delta x^{k+1} = H^T(x^k)R^{-1}[z - h(x^k)] \quad (3)$$

em que $G(x)$ é conhecida como matriz de ganho e definida como $G(x) = \frac{\partial g}{\partial x}$

Por ser um processo iterativo, o estado estimado é determinado quando $|\Delta x|$ é menor que uma tolerância determinada. Neste caso é dito que o processo convergiu. A Equação (4) apresenta esse critério de parada.

$$\left|\Delta x^k\right| \leqslant \epsilon \quad (4)$$

Criticalidade - Medidas e Conjuntos

As medidas críticas e os conjuntos críticos podem ser obtidos a partir do EE linear. A criticalidade é um problema de característica topológica, então os valores das medidas e das admitâncias não são necessários. A principal característica das medidas críticas é que tanto o resíduo da estimação quanto o desvio padrão são iguais a zero. Desta forma, não é possível identificar erros presentes nas medidas críticas, além disso caso esta medida torne-se indisponível o sistema será dito não observável. Já as medidas pertencentes aos conjuntos críticos possuem tanto correlação máxima entre os resíduos quanto igualdade entre os resíduos normalizados. Neste caso, é possível detectar a presença de medidas errôneas, porém a identificação não pode ser realizada.

Outra característica que vale ressaltar é que quando se perde uma medida pertencente a algum conjunto crítico, todas as medidas daquele conjuto tornam-se críticas.

Em resumo, para a identificação das medidas críticas e dos conjuntos críticos:

- Medidas Críticas:

$$r_i = r_i^N = 0$$

- Conjuntos Críticos:

$$\rho = \frac{r_i^N}{r_k^N} = 1$$
$$\gamma = \frac{|E_{ik}|}{\sqrt{E_i}\sqrt{E_k}} = 1$$

Bibliotecas e Funções

- Pandas - Biblioteca padrão do python para trabalhar com dados tabulares
- Numpy - Biblioteca padrão do python para trabalhar com dados numéricos
- Matplotlib - Biblioteca padrão do python para trabalhar com gráficos
- PSSS - Função criada para o processo da Estimação de Estado

In [1]:
```
from PSSS import *
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Parâmetros das Linhas da rede teste

In [2]:
```
line_params = 'Dados_linha_XIV_barras.xlsx'
pd.read_excel(line_params)
```

Out[2]:

	De	Para	R	X	C	Tap
0	1	2	0.01938	0.05917	18.939394	0
1	1	5	0.05403	0.22304	20.325203	0
2	2	3	0.04699	0.19797	22.831050	0
3	2	4	0.05811	0.17632	26.737968	0
4	2	5	0.05695	0.17388	29.411765	0
5	3	4	0.06701	0.17103	28.901734	0
6	4	5	0.01335	0.04211	78.125000	0
7	4	7	0.00000	0.20912	0.000000	0
8	4	9	0.00000	0.55618	0.000000	0
9	5	6	0.00000	0.25202	0.000000	0
10	6	11	0.09498	0.19890	0.000000	0
11	6	12	0.12291	0.25581	0.000000	0
12	6	13	0.06615	0.13027	0.000000	0
13	7	8	0.00000	0.17615	0.000000	0
14	7	9	0.00000	0.11001	0.000000	0
15	9	10	0.03181	0.08450	0.000000	0
16	9	14	0.12711	0.27038	0.000000	0
17	10	11	0.08205	0.19207	0.000000	0
18	12	13	0.22092	0.19988	0.000000	0
19	13	14	0.17093	0.34082	0.000000	0

Medições disponíveis na rede

In [3]:
```
meansured_file = 'Meansured_data_XIV_bus.xlsx'
pd.read_excel(meansured_file)
```

Out[3]:

	Tipo	De	Para	Valor	Desvio_Pad
0	P	1	-	1.2660	0.016125
1	P	2	-	0.1564	0.012247
2	P	2	1	-0.8613	0.017321
3	P	1	5	0.4354	0.014491
4	P	2	3	0.4628	0.014832
5	P	2	4	0.3406	0.013784
6	P	5	2	-0.2439	0.013038
7	P	3	4	-0.1982	0.013038
8	P	4	5	-0.3882	0.014491
9	P	7	4	-0.2079	0.013038
10	P	4	9	0.1308	0.012247
11	P	5	6	0.3203	0.013784
12	P	11	6	-0.0291	0.011832
13	P	6	12	0.0432	0.011832
14	P	6	13	0.0629	0.012247
15	P	7	8	-0.0049	0.011832
16	P	9	7	-0.1922	0.013038
17	P	9	10	0.0328	0.011832
18	P	14	9	-0.0408	0.011832
19	P	10	11	-0.0186	0.011832
20	P	13	12	0.0004	0.011832
21	P	13	14	0.0244	0.011832
22	Q	1	-	-0.0759	0.011832
23	Q	2	-	-0.1262	0.012247
24	Q	2	1	0.0015	0.011832
25	Q	1	5	-0.0486	0.011832
26	Q	2	3	0.0766	0.012247
27	Q	2	4	-0.1233	0.012247
28	Q	5	2	0.0832	0.012247
29	Q	3	4	-0.1012	0.012649
30	Q	4	5	0.0097	0.011832
31	Q	7	4	0.1381	0.012649
32	Q	4	9	-0.0640	0.011832
33	Q	5	6	-0.0922	0.012247
34	Q	11	6	-0.0361	0.011832
35	Q	6	12	0.0276	0.011832
36	Q	6	13	0.0380	0.011832
37	Q	7	8	-0.1227	0.012649
38	Q	9	7	-0.0297	0.011832
39	Q	9	10	0.0126	0.011832
40	Q	14	9	-0.0111	0.011832
41	Q	10	11	0.0055	0.011832
42	Q	13	12	0.0027	0.011832
43	Q	13	14	0.0259	0.011832
44	V	1	-	1.0551	0.003162
45	V	2	-	1.0468	0.003162

Executando a função state_estimation(). Esta função retorna:

- y_bus_matrix - Matriz Y Barra do sistema
- criticality_data - Dados Relacionados a Criticalidade das medidas
- J_list - Função de custo ao longo do processo iterativo
- J_critical - Valor crítico da função de custo
- State_dataframe - Dados do Estado estimado com as medidas disponíveis
- data_SE - Dados Relacionados as medidas estimadas

In [4]:
```
y_bus_matrix, criticality_data, J_list, J_critical, State_dataframe, data_SE = state_estimation(line_pa
rams = line_params, meansured_file = meansured_file)
```

Visualização da Matriz Y_{bus} obtida pelo programa state_estimation

$$Y_{bus} = Z_{bus}^{-1}$$

In [5]:
```
amount_bus = y_bus_matrix.shape[0]
bus = range(1, amount_bus+1)
pd.DataFrame(y_bus_matrix, index = bus, columns = bus).style.format("{:,.1f}".format).set_properties(**
{'font-size': '8pt'})
```

Out[5]:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	6.0-19.4j	-5.0+15.3j	0.0+0.0j	0.0+0.0j	-1.0+4.2j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
2	-5.0+15.3j	9.5-30.3j	-1.1+4.8j	-1.7+5.1j	-1.7+5.2j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
3	0.0+0.0j	-1.1+4.8j	3.1-9.8j	-2.0+5.1j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
4	0.0+0.0j	-1.7+5.1j	-2.0+5.1j	10.5-38.3j	-6.8+21.6j	0.0+0.0j	0.0+4.8j	0.0+0.0j	0.0+1.8j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
5	-1.0+4.2j	-1.7+5.2j	0.0+0.0j	-6.8+21.6j	9.6-34.9j	0.0+4.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
6	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+4.0j	6.6-17.3j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-2.0+4.1j	-1.5+3.2j	-3.1+6.1j	0.0+0.0j
7	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+4.8j	0.0+0.0j	0.0+0.0j	0.0-19.5j	0.0+5.7j	0.0+9.1j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
8	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+5.7j	0.0-5.7j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j
9	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+1.8j	0.0+0.0j	0.0+0.0j	0.0+9.1j	0.0+0.0j	5.3-24.3j	-3.9+10.4j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-1.4+3.0j
10	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-3.9+10.4j	5.8-14.8j	-1.9+4.4j	0.0+0.0j	0.0+0.0j	0.0+0.0j
11	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-2.0+4.1j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-1.9+4.4j	3.8-8.5j	0.0+0.0j	0.0+0.0j	0.0+0.0j
12	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-1.5+3.2j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	4.0-5.4j	-2.5+2.3j	0.0+0.0j
13	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-3.1+6.1j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-2.5+2.3j	6.8-10.7j	-1.2+2.3j
14	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-1.4+3.0j	0.0+0.0j	0.0+0.0j	0.0+0.0j	-1.2+2.3j	2.6-5.4j

Medidas Críticas e Conjuntos Críticos calculados pelo programa a partir do estimador de estado linear.

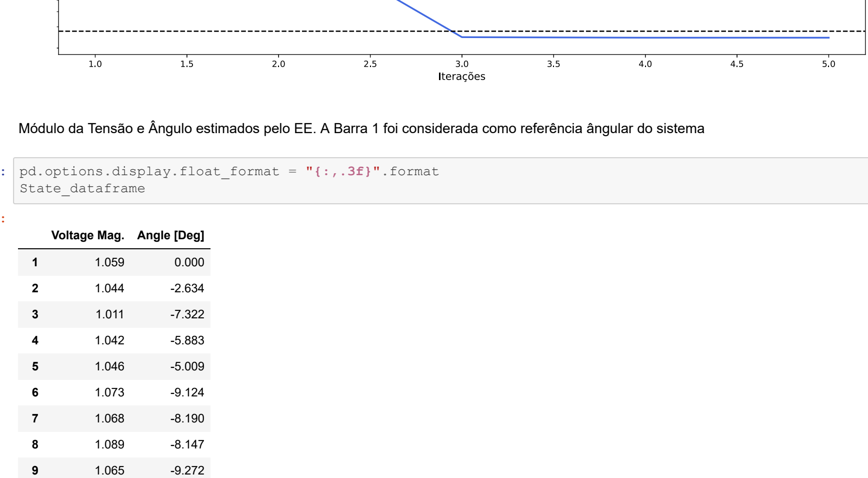
In [6]:
```
criticality_data.dropna().sort_values(by= 'Cricality')
```

Out[6]:

	Meansured_Type	Location	Cricality
9	P_flow	(7, 4)	CSets_1
16	P_flow	(9, 7)	CSets_1
12	P_flow	(11, 6)	CSets_2
17	P_flow	(9, 10)	CSets_2
19	P_flow	(10, 11)	CSets_2
13	P_flow	(6, 12)	CSets_3
20	P_flow	(13, 12)	CSets_3
18	P_flow	(14, 9)	CSets_4
21	P_flow	(13, 14)	CSets_4
15	P_flow	(7, 8)	Critical Meansured

Evolução da função de custo ao longo do processo iterativo da EE. Pode-se notar que o valor de $J(x)$ está abaixo do valor crítico. Desta forma, não há motivos para se rejeitar a hipótese nula H_0 de que não há medidas ruins utilizando $\alpha = 0.05$.

In [7]:
```
plt.figure(figsize = (17, 8))
iteracoes = range(1, len(J_list)+1)
plt.plot(iteracoes, J_list, color = 'royalblue', lw = 2, label = 'Função de Custo')
plt.axhline(J_critical, color = 'black', ls = '--', label = 'Valor Crítico da Função de Custo')
plt.yscale('log')
plt.legend()
plt.xlabel('Iterações', fontsize = 12)
plt.ylabel('Função de Custo - Escala Log', fontsize = 12)
plt.title('Evolução da Função de Custo ao longo das Iterações', fontsize = 12)
plt.show()
```



Módulo da Tensão e Ângulo estimados pelo EE. A Barra 1 foi considerada como referência ângular do sistema

In [8]:
```
pd.options.display.float_format = "{:,.3f}".format
State_dataframe
```

Out[8]:

	Voltage Mag.	Angle [Deg]
1	1.059	0.000
2	1.044	-2.634
3	1.011	-7.322
4	1.042	-5.883
5	1.046	-5.009
6	1.073	-9.124
7	1.068	-8.190
8	1.089	-8.147
9	1.065	-9.272
10	1.063	-9.396
11	1.064	-9.216
12	1.063	-9.461
13	1.065	-9.434
14	1.056	-9.712

Medidas Estimadas, Erro e Erro Normalizado para o sistema 14 barras do IEEE. Pode-se notar que todos os erros encontram-se abaixo do limiar padrão de 3. Assim, não há motivo para acreditar que há problemas no processo de medição.

OBS: Uma coloração divergente foi utilizada na coluna de erro normalizado. Tendo erros mais altos tendendo a cor vermelho e erros mais baixos tendendo a cor azul.

In [9]:
```
import seaborn as sns
cmap=sns.diverging_palette(250, 5, as_cmap=True)
data_SE.style.background_gradient(cmap, subset=['Error_Normalize'])
```

Out[9]:

	Meansured Type	Location	Meansured	Meansured Estimated	Error	Error_Normalize
0	P_inj	1	1.266000	1.288484	-0.022484	2.237543
1	P_inj	2	0.156400	0.165357	-0.008957	1.798094
2	P_flow	(2, 1)	-0.861300	-0.847507	-0.013793	0.919651
3	P_flow	(1, 5)	0.435400	0.428190	0.007210	0.532682
4	P_flow	(2, 3)	0.462800	0.456335	0.006465	0.520301
5	P_flow	(2, 4)	0.340600	0.323045	0.017555	1.371557
6	P_flow	(5, 2)	-0.243900	-0.230285	-0.013615	1.131389
7	P_flow	(3, 4)	-0.198200	-0.195457	-0.002743	0.361335
8	P_flow	(4, 5)	-0.388200	-0.389885	0.001685	0.473014
9	P_flow	(7, 4)	-0.207900	-0.214161	0.006261	0.869415
10	P_flow	(4, 9)	0.130800	0.117923	0.012877	1.123025
11	P_flow	(5, 6)	0.320300	0.319576	0.000724	0.083010
12	P_flow	(11, 6)	-0.029100	-0.026667	-0.002433	0.415762
13	P_flow	(6, 12)	0.043200	0.038190	0.005010	0.653540
14	P_flow	(6, 13)	0.062900	0.065509	-0.002609	0.512071
15	P_flow	(7, 8)	-0.004900	-0.004900	0.000000	0.000000
16	P_flow	(9, 7)	-0.192200	-0.195387	0.003187	0.860409
17	P_flow	(9, 10)	0.032800	0.033810	-0.001010	0.421017
18	P_flow	(14, 9)	-0.040800	-0.040855	0.000055	0.008730
19	P_flow	(10, 11)	-0.018600	-0.016266	-0.002334	0.419118
20	P_flow	(13, 12)	0.000400	0.006370	-0.005970	0.736957
21	P_flow	(13, 14)	0.024400	0.024045	0.000355	0.044332
22	Q_inj	1	-0.075900	-0.076828	0.000928	0.124640
23	Q_inj	2	-0.126200	-0.131709	0.005509	0.948037
24	Q_flow	(2, 1)	0.001500	0.006869	-0.005369	0.540205
25	Q_flow	(1, 5)	-0.048600	-0.050591	0.001991	0.180608
26	Q_flow	(2, 3)	0.076600	0.063055	0.013545	1.422349
27	Q_flow	(2, 4)	-0.123300	-0.101080	-0.022220	1.963577
28	Q_flow	(5, 2)	0.083200	0.073174	0.010026	0.882339
29	Q_flow	(3, 4)	-0.101200	-0.122075	0.020875	2.466130
30	Q_flow	(4, 5)	0.009700	0.008359	0.001341	0.495699
31	Q_flow	(7, 4)	0.138100	0.141234	-0.003134	0.457693
32	Q_flow	(4, 9)	-0.064000	-0.040920	-0.023080	2.073785
33	Q_flow	(5, 6)	-0.092200	-0.099579	0.007379	0.993687
34	Q_flow	(11, 6)	-0.036100	-0.036533	0.000433	0.070567
35	Q_flow	(6, 12)	0.027600	0.024901	0.002699	0.351719
36	Q_flow	(6, 13)	0.038000	0.035604	0.002396	0.499622
37	Q_flow	(7, 8)	-0.122700	-0.122700	0.000000	0.000000
38	Q_flow	(9, 7)	-0.029700	-0.028059	-0.001641	0.504346
39	Q_flow	(9, 10)	0.012600	0.012498	0.000102	0.040658
40	Q_flow	(14, 9)	-0.011100	-0.018935	0.007835	1.215057
41	Q_flow	(10, 11)	0.005500	0.005163	0.000337	0.058140
42	Q_flow	(13, 12)	0.002700	0.003331	-0.000631	0.077339
43	Q_flow	(13, 14)	0.025900	0.016087	0.009813	1.214079
44	V	1	1.055100	1.059136	-0.004036	1.786382
45	V	2	1.046800	1.044309	0.002491	1.104458

Teste da Retirada de Medidas Pertencentes a Conjuntos Críticos

Retirando-se a medida de fluxo de potência ativa entre as barras 9-10, a qual pertencia a um conjunto crítico, todas as medidas daquele conjunto tornam-se críticas.

In [10]:
```
meansured_file_cmeans = 'Meansured_data_XIV_bus_cset.xlsx'
_, criticality_data, _, _, _, _ = state_estimation(line_params = line_params, meansured_file = meansure
d_file_cmeans)
```

In [11]:
```
criticality_data.dropna().sort_values(by= 'Cricality')
```

Out[11]:

	Meansured_Type	Location	Cricality
9	P_flow	(7, 4)	CSets_1
16	P_flow	(9, 7)	CSets_1
11	P_flow	(5, 6)	CSets_2
17	P_flow	(14, 9)	CSets_2
20	P_flow	(13, 14)	CSets_2
13	P_flow	(6, 12)	CSets_3
19	P_flow	(13, 12)	CSets_3
12	P_flow	(11, 6)	Critical Meansured
15	P_flow	(7, 8)	Critical Meansured
18	P_flow	(10, 11)	Critical Meansured

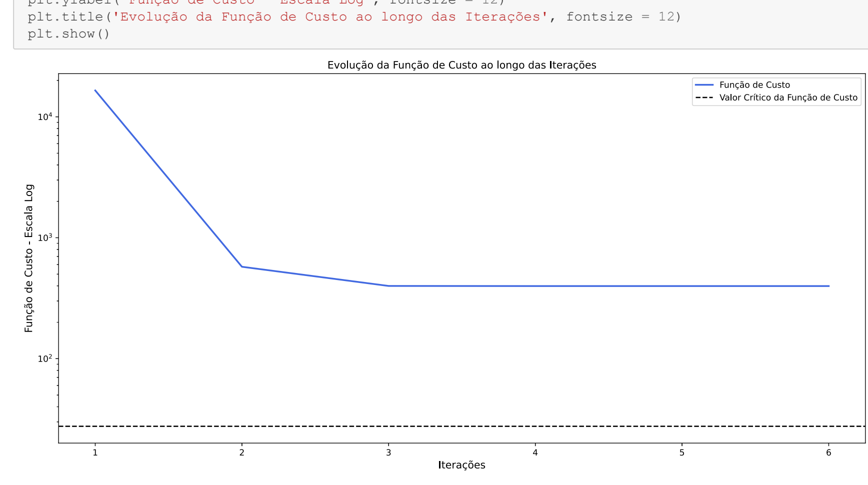
Teste da Detecção e Identificação de Medidas com Erros Grosseiros

Foi modificado o valor de P_{15} para $P_{15} = 0.725$. O que representa um erro na medida de 20 vezes o desvio padrão. Sendo, portanto, considerado um erro grosseiro. Este teste irá verificar a capacidade do programa detectar o erro nesta medida.

In [12]:
```
meansured_wrong = 'Meansured_data_XIV_bus_wrong.xlsx'
y_bus_matrix, criticality_data, J_list, J_critical, State_dataframe, data_SE = state_estimation(line_pa
rams = line_params, meansured_file = meansured_wrong)
```

Pode-se notar que o valor de $J(x)$ está acima do valor crítico. Desta forma, há motivos para se rejeitar a hipótese nula H_0 de que não há medidas ruins utilizando $\alpha = 0.05$.

In [13]:
```
plt.figure(figsize = (17, 8))
iteracoes = range(1, len(J_list)+1)
plt.plot(iteracoes, J_list, color = 'royalblue', lw = 2, label = 'Função de Custo')
plt.axhline(J_critical, color = 'black', ls = '--', label = 'Valor Crítico da Função de Custo')
plt.yscale('log')
plt.legend()
plt.xlabel('Iterações', fontsize = 12)
plt.ylabel('Função de Custo - Escala Log', fontsize = 12)
plt.title('Evolução da Função de Custo ao longo das Iterações', fontsize = 12)
plt.show()
```



Teste - Medida com Erro Grosseiro

Verificando os resíduos normalizados pode-se notar que o maior resíduo está na medida P_{15}. Sendo assim, o estimador pode determinar corretamente o erro na medida.

In [14]:
```
data_SE.style.background_gradient(cmap, subset=['Error_Normalize'])
```

Out[14]:

	Meansured Type	Location	Meansured	Meansured Estimated	Error	Error_Normalize
0	P_inj	1	1.266000	1.362147	-0.096147	9.586303
1	P_inj	2	0.156400	0.187691	-0.031291	6.286040
2	P_flow	(2, 1)	-0.861300	-0.884052	0.022752	1.517423
3	P_flow	(1, 5)	0.725228	0.464246	0.260981	19.287602
4	P_flow	(2, 3)	0.462800	0.457451	0.005349	0.430291
5	P_flow	(2, 4)	0.340600	0.349220	-0.008620	0.673506
6	P_flow	(5, 2)	-0.243900	-0.261039	0.017139	1.423738
7	P_flow	(3, 4)	-0.198200	-0.171767	-0.026433	3.488240
8	P_flow	(4, 5)	-0.388200	-0.368993	-0.019207	5.395679
9	P_flow	(7, 4)	-0.207900	-0.215102	0.007202	1.000281
10	P_flow	(4, 9)	0.130800	0.118375	0.012425	1.083620
11	P_flow	(5, 6)	0.320300	0.318022	0.002278	0.261080
12	P_flow	(11, 6)	-0.029100	-0.025973	-0.003127	0.534260
13	P_flow	(6, 12)	0.043200	0.038149	0.005051	0.658882
14	P_flow	(6, 13)	0.062900	0.065341	-0.002441	0.479073
15	P_flow	(7, 8)	-0.004900	-0.004900	0.000000	0.000000
16	P_flow	(9, 7)	-0.192200	-0.195875	0.003675	0.992051
17	P_flow	(9, 10)	0.032800	0.034100	-0.001300	0.541576
18	P_flow	(14, 9)	-0.040800	-0.041229	0.000429	0.067694
19	P_flow	(10, 11)	-0.018600	-0.015598	-0.003002	0.538883
20	P_flow	(13, 12)	0.000400	0.006407	-0.006007	0.741495
21	P_flow	(13, 14)	0.024400	0.023576	0.000824	0.102868
22	Q_inj	1	-0.075900	-0.075945	0.000045	0.006094
23	Q_inj	2	-0.126200	-0.134393	0.008193	1.409993
24	Q_flow	(2, 1)	0.001500	0.010211	-0.008711	0.876186
25	Q_flow	(1, 5)	-0.048600	-0.049304	0.000704	0.063904
26	Q_flow	(2, 3)	0.076600	0.063593	0.013007	1.365774
27	Q_flow	(2, 4)	-0.123300	-0.104006	-0.019294	1.705509
28	Q_flow	(5, 2)	0.083200	0.079211	0.003989	0.350935
29	Q_flow	(3, 4)	-0.101200	-0.128845	0.027645	3.267180
30	Q_flow	(4, 5)	0.009700	0.005879	0.003821	1.410463
31	Q_flow	(7, 4)	0.138100	0.140985	-0.002885	0.421282
32	Q_flow	(4, 9)	-0.064000	-0.040792	-0.023208	2.085343
33	Q_flow	(5, 6)	-0.092200	-0.099942	0.007742	1.042568
34	Q_flow	(11, 6)	-0.036100	-0.036694	0.000594	0.096731
35	Q_flow	(6, 12)	0.027600	0.024909	0.002691	0.350634
36	Q_flow	(6, 13)	0.038000	0.035642	0.002358	0.491609
37	Q_flow	(7, 8)	-0.122700	-0.122700	0.000000	0.000000
38	Q_flow	(9, 7)	-0.029700	-0.028153	-0.001547	0.475353
39	Q_flow	(9, 10)	0.012600	0.012455	0.000145	0.058063
40	Q_flow	(14, 9)	-0.011100	-0.018852	0.007752	1.202310
41	Q_flow	(10, 11)	0.005500	0.005032	0.000468	0.080617
42	Q_flow	(13, 12)	0.002700	0.003307	-0.000607	0.074323
43	Q_flow	(13, 14)	0.025900	0.016204	0.009696	1.199422
44	V	1	1.055100	1.062174	-0.007074	3.126516
45	V	2	1.046800	1.046809	-0.000009	0.004104

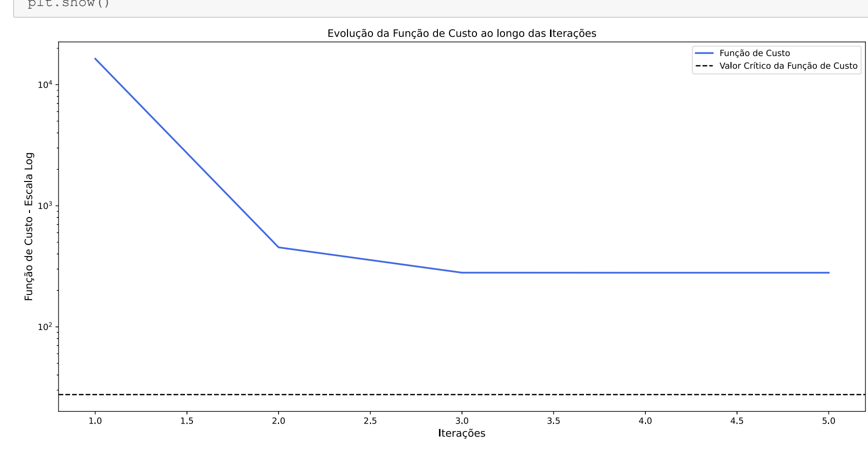
Teste da Detecção de Medidas com Erros Grosseiros em Conjuntos Críticos

Foram modificados os valores de $P_{(11,6)}$, $P_{(9,10)}$ e $P_{(10,11)}$ para $P_{(11,6)} = -0.265$, $P_{(9,10)} = 0.269$, $P_{(10,11)} = -0.255$. O que representa um erro nas medidas de 20 vezes o desvio padrão. Sendo, portanto, considerado um erro grosseiro. Este teste irá verificar a capacidade do programa detectar o erro em multiplas medidas.

In [15]:
```
meansured_wrong = 'Meansured_data_XIV_bus_wrong_cset.xlsx'
y_bus_matrix, criticality_data, J_list, J_critical, State_dataframe, data_SE = state_estimation(line_pa
rams = line_params, meansured_file = meansured_wrong)
```

Após a convergência do processo o valor de $J(x)$ encontra-se maior que o limiar estabelecido para este sistema. Indicando assim a presença de erros grosseiros.

In [16]:
```
plt.figure(figsize = (17, 8))
iteracoes = range(1, len(J_list)+1)
plt.plot(iteracoes, J_list, color = 'royalblue', lw = 2, label = 'Função de Custo')
plt.axhline(J_critical, color = 'black', ls = '--', label = 'Valor Crítico da Função de Custo')
plt.yscale('log')
plt.legend()
plt.xlabel('Iterações', fontsize = 12)
plt.ylabel('Função de Custo - Escala Log', fontsize = 12)
plt.title('Evolução da Função de Custo ao longo das Iterações', fontsize = 12)
plt.show()
```



Como pode-se observar na tabela abaixo todos os resíduos das medidas críticas estão acima de 3 e possuem valores próximos.

In [17]:
```
data_SE.style.background_gradient(cmap, subset=['Error_Normalize'])
```

Out[17]:

	Meansured Type	Location	Meansured	Meansured Estimated	Error	Error_Normalize
0	P_inj	1	1.266000	1.288268	-0.022268	2.215971
1	P_inj	2	0.156400	0.165458	-0.009058	1.818163
2	P_flow	(2, 1)	-0.861300	-0.848534	-0.012766	0.851150
3	P_flow	(1, 5)	0.435400	0.426917	0.008483	0.626772
4	P_flow	(2, 3)	0.462800	0.457404	0.005396	0.434272
5	P_flow	(2, 4)	0.340600	0.325116	0.015484	1.209765
6	P_flow	(5, 2)	-0.243900	-0.228318	-0.015582	1.294952
7	P_flow	(3, 4)	-0.198200	-0.194581	-0.003619	0.476782
8	P_flow	(4, 5)	-0.388200	-0.406980	0.018780	5.272054
9	P_flow	(7, 4)	-0.207900	-0.262241	0.054341	7.539376
10	P_flow	(4, 9)	0.130800	0.140903	-0.010103	0.880850
11	P_flow	(5, 6)	0.320300	0.239762	0.080538	9.351410
12	P_flow	(11, 6)	-0.265743	-0.171680	-0.094064	15.914797
13	P_flow	(6, 12)	0.043200	0.043313	-0.000113	0.014804
14	P_flow	(6, 13)	0.062900	0.086640	-0.023740	4.656694
15	P_flow	(7, 8)	-0.004900	-0.004900	0.000000	0.000000
16	P_flow	(9, 7)	-0.192200	-0.220692	0.028492	7.638415
17	P_flow	(9, 10)	0.269443	0.307141	-0.037698	15.860120
18	P_flow	(14, 9)	-0.040800	0.005418	-0.046218	7.305551
19	P_flow	(10, 11)	-0.255243	-0.166042	-0.089201	15.866996
20	P_flow	(13, 12)	0.000400	0.001272	-0.000872	0.107534
21	P_flow	(13, 14)	0.024400	0.082374	-0.057974	7.265468
22	Q_inj	1	-0.075900	-0.076970	0.001070	0.143732
23	Q_inj	2	-0.126200	-0.131718	0.005518	0.949443
24	Q_flow	(2, 1)	0.001500	0.006602	-0.005102	0.513303
25	Q_flow	(1, 5)	-0.048600	-0.051093	0.002493	0.226104
26	Q_flow	(2, 3)	0.076600	0.063367	0.013233	1.389638
27	Q_flow	(2, 4)	-0.123300	-0.100603	-0.022697	2.005783
28	Q_flow	(5, 2)	0.083200	0.073563	0.009637	0.848149
29	Q_flow	(3, 4)	-0.101200	-0.121912	0.020712	2.446610
30	Q_flow	(4, 5)	0.009700	0.005088	0.004612	1.706701
31	Q_flow	(7, 4)	0.138100	0.114691	0.023409	3.410817
32	Q_flow	(4, 9)	-0.064000	-0.026776	-0.037224	3.347746
33	Q_flow	(5, 6)	-0.092200	-0.140445	0.048245	6.493651
34	Q_flow	(11, 6)	-0.036100	-0.030853	-0.005247	0.854136
35	Q_flow	(6, 12)	0.027600	0.025133	0.002467	0.321893
36	Q_flow	(6, 13)	0.038000	0.036075	0.001925	0.402892
37	Q_flow	(7, 8)	-0.122700	-0.122700	0.000000	0.000000
38	Q_flow	(9, 7)	-0.029700	-0.039097	0.009397	2.887285
39	Q_flow	(9, 10)	0.012600	0.015336	-0.002736	1.100016
40	Q_flow	(14, 9)	-0.011100	-0.019110	0.008010	1.234728
41	Q_flow	(10, 11)	0.005500	0.008675	-0.003175	0.533906
42	Q_flow	(13, 12)	0.002700	0.005184	-0.002484	0.303814
43	Q_flow	(13, 14)	0.025900	0.015824	0.010076	1.262098
44	V	1	1.055100	1.059168	-0.004068	1.800815
45	V	2	1.046800	1.044305	0.002495	1.106294

Teste da Influência de Parâmetros Erroneos no Resíduo Normalizado

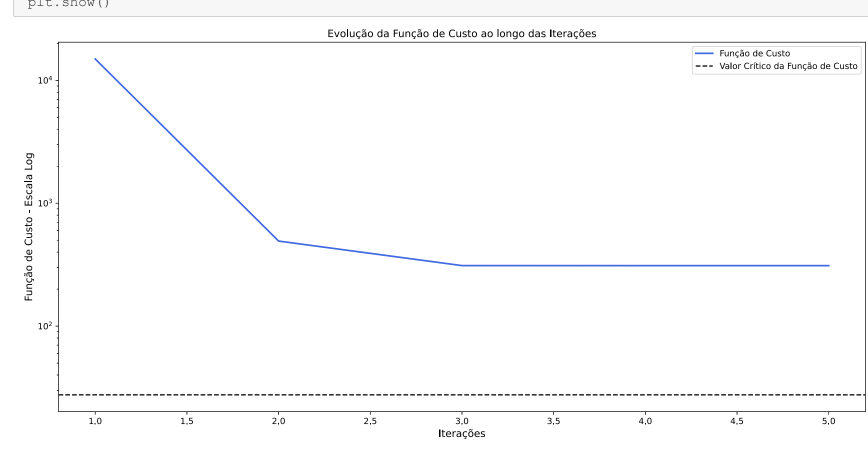
Neste teste os parâmetros de admitância relacionados a barra 2 foram alterados. O que ocorrerá é uma falsa impressão de que há medidas com erros grosseiros quando na verdade deve os parâmetros que causam erros no processo de estimação de estado. Os valores adotados para este teste foram os seguntes:

$$X_{1,2} = 0.090$$
$$X_{2,3} = 0.340$$
$$X_{2,4} = 0.244$$
$$X_{2,5} = 0.317$$

In [18]:
```
line_wrong = 'Dados_linha_XIV_barras_erro_barra_2.xlsx'
y_bus_matrix, criticality_data, J_list, J_critical, State_dataframe, data_SE = state_estimation(line_pa
rams = line_wrong, meansured_file = meansured_file)
```

Como pode-se observar abaixo o processo convergiu e resultou num valor de $J(x)$ acima do limiar estabelecido.

In [19]:
```
plt.figure(figsize = (17, 8))
iteracoes = range(1, len(J_list)+1)
plt.plot(iteracoes, J_list, color = 'royalblue', lw = 2, label = 'Função de Custo')
plt.axhline(J_critical, color = 'black', ls = '--', label = 'Valor Crítico da Função de Custo')
plt.yscale('log')
plt.legend()
plt.xlabel('Iterações', fontsize = 12)
plt.ylabel('Função de Custo - Escala Log', fontsize = 12)
plt.title('Evolução da Função de Custo ao longo das Iterações', fontsize = 12)
plt.show()
```



Os residuos normalizados também estão superiores a 3, neste caso poderia se suspeitar erroneamente de parâmetros com erros grosseiros.

In [20]:
```
data_SE.style.background_gradient(cmap, subset=['Error_Normalize'])
```
Out[20]:

	Meansured Type	Location	Meansured	Meansured Estimated	Error	Error_Normalize
0	P_inj	1	1.266000	1.291470	-0.025470	2.558429
1	P_inj	2	0.156400	0.148190	0.008210	1.610086
2	P_flow	(2, 1)	-0.861300	-0.713464	-0.147836	9.548050
3	P_flow	(1, 5)	0.435400	0.568971	-0.133571	10.287536
4	P_flow	(2, 3)	0.462800	0.349484	0.113316	8.336439
5	P_flow	(2, 4)	0.340600	0.317647	0.022953	1.771602
6	P_flow	(5, 2)	-0.243900	-0.192446	-0.051454	4.104606
7	P_flow	(3, 4)	-0.198200	-0.250874	0.052674	10.040389
8	P_flow	(4, 5)	-0.388200	-0.409873	0.021673	7.614646
9	P_flow	(7, 4)	-0.207900	-0.213213	0.005313	0.737952
10	P_flow	(4, 9)	0.130800	0.117482	0.013318	1.161486
11	P_flow	(5, 6)	0.320300	0.321090	-0.000790	0.090585
12	P_flow	(11, 6)	-0.029100	-0.027283	-0.001817	0.310420
13	P_flow	(6, 12)	0.043200	0.038226	0.004974	0.648835
14	P_flow	(6, 13)	0.062900	0.065656	-0.002756	0.540980
15	P_flow	(7, 8)	-0.004900	-0.004900	0.000000	0.000000
16	P_flow	(9, 7)	-0.192200	-0.194900	0.002700	0.728919
17	P_flow	(9, 10)	0.032800	0.033547	-0.000747	0.311464
18	P_flow	(14, 9)	-0.040800	-0.040526	-0.000274	0.043327
19	P_flow	(10, 11)	-0.018600	-0.016866	-0.001734	0.311323
20	P_flow	(13, 12)	0.000400	0.006341	-0.005941	0.733433
21	P_flow	(13, 14)	0.024400	0.024456	-0.000056	0.007004
22	Q_inj	1	-0.075900	-0.082491	0.006591	0.912183
23	Q_inj	2	-0.126200	-0.123691	-0.002509	0.419878
24	Q_flow	(2, 1)	0.001500	-0.019811	0.021311	2.082640
25	Q_flow	(1, 5)	-0.048600	-0.086075	0.037475	3.550576
26	Q_flow	(2, 3)	0.076600	0.043744	0.032856	3.050378
27	Q_flow	(2, 4)	-0.123300	-0.084453	-0.038847	3.388980
28	Q_flow	(5, 2)	0.083200	0.037457	0.045743	3.875753
29	Q_flow	(3, 4)	-0.101200	-0.104871	0.003671	0.586262
30	Q_flow	(4, 5)	0.009700	0.014174	-0.004474	2.046990
31	Q_flow	(7, 4)	0.138100	0.141262	-0.003162	0.461645
32	Q_flow	(4, 9)	-0.064000	-0.041013	-0.022987	2.065516
33	Q_flow	(5, 6)	-0.092200	-0.099501	0.007301	0.983387
34	Q_flow	(11, 6)	-0.036100	-0.036231	0.000131	0.021308
35	Q_flow	(6, 12)	0.027600	0.024884	0.002716	0.353905
36	Q_flow	(6, 13)	0.038000	0.035532	0.002468	0.514594
37	Q_flow	(7, 8)	-0.122700	-0.122700	0.000000	0.000000
38	Q_flow	(9, 7)	-0.029700	-0.028071	-0.001629	0.500588
39	Q_flow	(9, 10)	0.012600	0.012602	-0.000002	0.000840
40	Q_flow	(14, 9)	-0.011100	-0.019097	0.007997	1.240202
41	Q_flow	(10, 11)	0.005500	0.005430	0.000070	0.012096
42	Q_flow	(13, 12)	0.002700	0.003362	-0.000662	0.081155
43	Q_flow	(13, 14)	0.025900	0.015870	0.010030	1.240883
44	V	1	1.055100	1.059257	-0.004157	1.849314
45	V	2	1.046800	1.044991	0.001809	0.807090

In [ ]: