MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
(Real Academia de Artilharia, Fortificação e Desenho, 1792)
CURSO DE GRADUAÇÃO EM ENGENHARIA CARTOGRÁFICA

GABRIEL SANTOS DO AMARAL

ÍTALO DE LANNA ARCANJO

MENDSON CABRAL AMENTA BOAZ

IMPLEMENTAÇÃO DE SISTEMA COMPUTACIONAL DE CÁLCULOS GEODÉSICOS:

AJUSTAMENTO DE REDE GNSS UTILIZANDO MMQ

Rio de Janeiro

2018

INSTITUTO MILITAR DE ENGENHARIA

GABRIEL SANTOS DO AMARAL

ÍTALO DE LANNA ARCANJO

MENDSON CABRAL AMENTA BOAZ

IMPLEMENTAÇÃO DE SISTEMA COMPUTACIONAL DE CÁLCULOS GEODÉSICOS:

AJUSTAMENTO DE REDE GNSS UTILIZANDO MMQ

Projeto de Fim de Curso (PFC) apresentado ao Curso de Graduação em Engenharia Cartográfica do Instituto Militar de Engenharia, para a obtenção de grau na Verificação Final (VF) de 2018.

Orientadores:
Prof. Francisco Roberto da Rocha Gomes
Prof. Leonardo Castro de Oliveira

Rio de Janeiro

2018

INSTITUTO MILITAR DE ENGENHARIA

GABRIEL SANTOS DO AMARAL

ÍTALO DE LANNA ARCANJO

MENDSON CABRAL AMENTA BOAZ

IMPLEMENTAÇÃO DE SISTEMA COMPUTACIONAL DE CÁLCULOS GEODÉSICOS:

AJUSTAMENTO DE REDE GNSS UTILIZANDO MMQ

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia Cartográfica do Instituto Militar de Engenharia, como requisito parcial para a obtenção de grau na Verificação Final (VF).

Orientadores: Prof. Francisco Roberto da Rocha Gomes
Prof. Leonardo Castro de Oliveira

Avaliado em 2018 pela seguinte Banca Examinadora:

_____
Francisco Roberto da Rocha Gomes Ms.C.

_____
Heloísa Alves Silva Marques D.E.

_____
Luís Antônio de Andrade Ms. C.

Rio de Janeiro
2018

INSTITUTO MILITAR DE ENGENHARIA

Praça General Tibúrcio, 80 – Praia Vermelha

Rio de Janeiro – RJ CEP: 22290-270

Este exemplar é de propriedade do Instituto Militar de Engenharia, que poderá incluí-lo em base de dados, armazenar em computador, microfilmar ou adotar qualquer forma de arquivamento.

É permitida a menção, reprodução parcial ou integral e a transmissão entre bibliotecas deste trabalho, sem modificação de seu texto, em qualquer meio que esteja ou venha a ser fixado, para pesquisa acadêmica, comentários e citações, desde que sem finalidade comercial e que seja feita a referência bibliográfica completa.

Os conceitos expressos neste trabalho são de responsabilidade do(s) autor(es) e do(s) orientador(es).

526	Boaz, Mendson Cabral Amenta. Amaral, Gabriel Santos do. Arcanjo, Ítalo de Lanna.
A485i	IMPLEMENTAÇÃO DE SISTEMA COMPUTACIONAL DE CÁLCULOS GEODÉSICOS: AJUSTAMENTO DE REDE GNSS UTILIZANDO MMQ/ Gabriel Santos do Amaral, Ítalo de Lanna Arcanjo, Mendson Cabral Amenta Boaz; orientado por Francisco Roberto da Rocha Gomes e Leonardo Castro de Oliveira– Rio de Janeiro: Instituto Militar de Engenharia, 2018. 20p.il. Projeto de Fim de Curso (PROFIC) – Instituto Militar de Engenharia, Rio de Janeiro, 2018. 1. Curso de Engenharia Cartográfica – Projeto de Fim de Curso. 2. Ajustamento de Rede GNSS. 3. Implementação. 4. Ambientes computacionais livres. I. Gomes, Francisco Roberto da Rocha; de Oliveira, Leonardo Castro. II. Título. III. Instituto Militar de Engenharia.

SUMÁRIO

LISTA DE ILUSTRAÇÕES.....6
LISTA DE ABREVIATURAS E SIGLAS.....7
1. INTRODUÇÃO.....10
1.1 CONTEXTUALIZAÇÃO.....10
1.2 OBJETIVO.....11
1.3 JUSTIFICATIVA.....11
1.4 ORGANIZAÇÃO.....12
2. REVISÃO BIBLIOGRÁFICA.....14
2.1 OBSERVAÇÕES GNSS.....14
2.2 POSICIONAMENTO PELO GNSS.....18
2.3 MMQ NO AJUSTE DE OBSERVAÇÕES.....20
2.3.1 PRINCÍPIOS DO MMQ.....20
2.3.2 MODELO PARAMÉTRICO.....21
2.4 AJUSTAMENTO DE REDES GNSS.....25
2.5 CONTROLE DE QUALIDADE.....26
2.5.1 TESTE ESTATÍSTICO QUI-QUADRADO.....27
2.5.2 ELIPSE DE ERRO.....28
2.6 IMPLEMENTAÇÃO COMPUTACIONAL.....31
2.6.1 LINGUAGEM PYTHON ADAPTADA AO AJUSTAMENYO.....31
2.6.2 DESENVOLVIMENTO WEB.....32
3. DESCRIÇÃO DA PLATAFORMA.....33
3.1 ESPECIFICAÇÕES TÉCNICAS.....33
3.2 ENTRADA DE DADOS.....34
3.3 INTERFACE DA PLATAFORMA *ONLINE*.....36
3.4 AJUSTAMENTO ONLINE.....37
3.5 HOSPEDAGEM 39
4. CONCLUSÃO.....39
5. REFERÊNCIAS BIBLIOGRÁFICAS.....39
APÊNDICE A. EXEMPLO: AJUSTAMENTO DE REDE GNSS.....40
APÊNDICE B. CÓDIGO PYTHON IMPLEMENTADO PARA SOLUÇÃO DE UM AJUSTAMENTO DE REDE GNSS.....44

LISTA DE ILUSTRAÇÕES

Figura 1 Rede GNSS 17

Figura 2 Esquematização do Posicionamento Relativo 18

Figura 3 Teste qui-quadrado para valor de significância de 5% 26

Figura 4 Retângulo de Erros 28

Figura 5 Elipse de erros em relação aos sistemas XY e X'Y' 29

Figura 6 Exemplo de planilha, com valores referentes ao problema do Apêndice A, que é disponibilizada online 30

Figura 7 Interface da página inicial do site desenvolvido 33

Figura 8 Exemplo de um resultado sem evidências de erros grosseiros 34

Figura 9 Exemplo de um resultado com evidências de erros grosseiros 34

LISTA DE ABREVIATURAS E SIGLAS

DoD	Department of Defense
GLONASS	*Global'naya Navigatsionnaya Sputnikowaya Sistema*
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IME	Instituto Militar de Engenharia
MMQ	Método dos Mínimos Quadrados
MVC	Matriz variância-covariância
NAVSTAR-GPS	*Navigation Satellite with Time and Ranging – Global Position System*
PSF	*Python Software Foundation*

RESUMO

Partindo do pressuposto da inevitável presença de erros nas observações de qualquer medida, sejam eles humanos, provenientes dos aparelhos utilizados para realizar tais observações ou erros de natureza aleatória, surge a necessidade de buscar os melhores valores que se adaptem – ou se ajustem – às condições nas quais tais medidas foram aferidas. Obter o valor exato de uma grandeza medida é algo impraticável, visto que, quando se realiza uma medição repetidas vezes é habitual que estas observações tenham valores distintos. Para o caso de uma rede GNSS, as observações de medidas dos vetores linhas de base são utilizadas a fim de se realizar um ajustamento da rede GNSS. É possível ao fim deste projeto compreender toda a teoria por trás do ajuste de uma rede GNSS, além dos testes de controle de qualidade para detecção de erros. O produto final é uma ferramenta online que realiza os cálculos referentes ao ajuste. A plataforma online recebe como entradas as observações de linhas de base e seus desvios padrões além dos valores das coordenadas de estações que são consideradas fixas. Como saída, disponibiliza os valores ajustados das estações envolvidas, análise da qualidade ajustamento através do teste qui-quadrado e também elipses de erros das coordenadas ajustadas das estações. Toda plataforma online e seus algoritmos são implementados na linguagem Python, por se tratar de uma linguagem alto nível, e utilizam o webframework Flask.

Palavras-chave: rede GNSS, ajustamento, python, flask

ABSTRACT

Due to the inevitable presence of errors in observation of any measure, being human or not, originated from devices used to perform such observations or randomly natured errors, a need to pursue better values which can adapt – or adjust – to conditions in which those measures were made is at hand. To obtain the exact value of a quantity is impractical, since it´s common to have different values when performing a measurement repeatedly. In the case of a GNSS network, observations of measurements of the baseline vectors are used in order to perform an adjustment of the GNSS network. It is possible, at the end of this Project, to understand all the theory behind the adjustment of a GNSS network and also the quality control tests for error detection. The final product is an online tool that performs calculations regarding the adjustment. The online platform receives as inputs the baseline observations and their standard deviations in addition to the coordinates of stations that are considered fixed. As an output, it provides the adjusted values of the stations involved, analysis of quality adjustment through the Chi-square test and also ellipses of errors of the coordinates of the stations. Every online platform and its algorithms are implemented in the Python language, because it is a high level language, and use the Flask webframework.

Keyword: GNSS network, adjustment, python, flask

1. INTRODUÇÃO

1.1 Contextualização

Desenvolvido pelo Departamento de Defesa dos Estados Unidos (DoD), o *Navigation Satellite with Time and Ranging – Global Positioning System* (NAVSTAR-GPS), ou simplesmente *Global Positioning System* GPS, é uma evolução do sistema TRANSIT. Com uma constelação de seis satélites, o TRANSIT usava o princípio *Doppler* para o posicionamento preciso e navegação. No entanto, o TRANSIT possuía uma precisão de cerca de 200 metros e devido à pouca quantidade de satélites em operação, não era frequente obter posições com frequência (WOLF, 2007).

Motivados, então, pela eficiência do sistema TRANSIT, o DoD desenvolveu o sistema GPS, que conta com uma constelação de 24 satélites e com um grau de acurácia e precisão superior a seu antecessor.

Semelhantemente ao sistema GPS, a extinta União Soviética iniciou o desenvolvimento de um projeto próprio que objetivava o posicionamento tridimensional e determinação de velocidade, o GLONASS (em russo, *Global'naya Navigatsionnaya Sputnikowaya Sistema)*. Com o projeto iniciado no princípio da década de 1970, o GLONASS foi oficialmente declarado operante em 1995, com uma constelação de 24 satélites. No entanto, por falta de novos lançamentos ocasionou um decrescimento na quantidade de satélites GLONASS em funcionamento (MONICO, 2008).

Impulsionada pela decisão americana de manter outras nações fora do controle do sistema GPS, a União Europeia iniciou o desenvolvimento do seu próprio sistema de posicionamento, o Galileo, nome dado em homenagem ao pai das observações astronômicas, o italiano Galileo Galilei. As etapas de desenvolvimento do sistema europeu se iniciaram no ano de 1999, com seu primeiro satélite teste sendo lançado em 2005.

A China também iniciou em 1983 o desenvolvimento do seu sistema de posicionamento próprio, o Beidou. Foi declarado operacional em 2011, sendo utilizado para serviços de navegação na China e em nações vizinhas. O Beidou tem previsão para ser totalmente concluído até o ano de 2020 (MONICO, 2008).

Os sistemas de navegação GPS, GLONASS, Galileo e Beidou podem ser agrupados em um amplo grupo, o GNSS. A sigla GNSS (*Global Navigation Satellite System*) foi apresentada pela primeira vez em 1991 pela ICAO (*International Civil Aviation Organization*) para designar qualquer sistema de posicionamento por satélites artificiais.

A navegação GNSS é possível através de cálculos de distâncias entre o receptor dos sinais GNSS e uma constelação formada por quatro satélites. Uma vantagem interessante do GNSS é não ser necessária a intervisibilidade entre estações e a tecnologia pode ser utilizada sob quaisquer condições climáticas (MONICO, 2008).

Em conjunto com este contexto de busca das coordenadas de determinados pontos que melhor se adaptem à geometria real, surge a motivação central deste trabalho que consiste em ajustar uma rede GNSS com o uso do MMQ.

A evolução computacional, avanço da matemática e desenvolvimento de linguagens de alto nível de programação, fazem o MMQ ser uma eficaz ferramenta no ajustamento de observações, justificando assim a escolha do método para este projeto.

Ainda no quesito computacional e maior procura por ferramentas *online* que solucionem específicos problemas matemáticos e inspirando-se no Projeto de Fim de Curso de Graduação em Engenharia Cartográfica de Alves (2016) que soluciona um ajustamento de uma rede de nivelamento altimétrico, vislumbrou-se a criação de uma plataforma *online* que solucione um ajustamento de uma rede GNSS.

1.2 Objetivo

Criar uma plataforma *online* de cálculos geodésicos do Instituto Militar de Engenharia (IME) em linguagem *Python* e utilizando o *framework Flask*, possibilitando que esta realize o ajustamento de uma rede GNSS e exiba dados complementares referentes à qualidade do ajustamento.

1.3 Justificativa

Com a plataforma *online* de ajustamentos altimétricos criada em "*Implementação de sistema para cálculos geodésicos em ambiente computacional livre: Ajustamento de rede de nivelamento geométrico"* (ALVES, 2016), vislumbrou-se a possibilidade de evoluir a plataforma, agregando a ela uma atualização que também permita o ajustamento de uma rede GNSS utilizando como princípio matemático o MMQ.

Apesar de já existirem *softwares* comerciais que façam o ajuste de redes GNSS, como por exemplo, o *Topcon Tools* e o *GNSS Solutions*, alguns fatores os tornam de restrito acesso ao público geral, como por exemplo, o valor para a compra de licenças.

Ainda sobre *softwares* que realizem ajustes de redes GNSS, destaca-se o AJURGPS, desenvolvido na Universidade Estadual Paulista. Ressalta-se o caráter científico e livre do *software.* O AJURGPS leva em consideração informações estocásticas das coordenadas conhecidas, considerando-as como injunções relativas (SILVA, 2007).

Diferentemente dos *softwares* citados acima, a plataforma implementada é *online* e não de forma executável. Com a ferramenta de cálculos geodésicos disponível *online*, torna-se mais fácil e rápido o acesso a ela. Essa forma de acesso também faz da ferramenta um recurso mais transparente, ou seja, os desenvolvedores poderão ter acesso ao perfil dos que a utilizam e, também, receber relatórios sobre o desempenho da plataforma. Ainda é interessante ressaltar que a ferramenta implementada é gratuita.

Por fim, é possível editar a plataforma sempre que for detectado algum problema ou quando for necessária uma nova atualização. Isso se deve ao fato de que os arquivos estarão instalados no servidor, então basta um acesso à internet para que seja possível alterá-los.

1.4 Organização

No capítulo 2 deste trabalho, é discutida a teoria por trás das observações GNSS, os tipos de erros encontrados nessas observações, os princípios do MMQ, controles de qualidade dos resultados obtidos, ajustamento de uma rede GNSS, entre outros assuntos relacionados à teoria necessária ao desenvolvimento deste projeto.

Os capítulos 3 e 4 deste Projeto de Fim de Curso são desenvolvidos, explicados e apresentados ao longo do segundo semestre do ano de 2018. Sendo, o capítulo 3 reservado para a explicação geral da linguagem de programação utilizada: o *Python*. É abordada, inclusive, uma explicação acerca dos módulos *Python* usados e suas funções principais. E no capítulo 4, poder-se-á encontrar a apresentação final da plataforma implementada, constando suas características e simulações de resultados.

2. REVISÃO BIBLIOGRÁFICA

2.1 Observações GNSS

Conceitualmente, o funcionamento prático do GNSS se assemelha ao de instrumentos eletrônicos de medição de distâncias: o sistema de posicionamento global opera por meio de seus satélites, que têm suas posições conhecidas com precisão (pontos de controle), observa distâncias até os receptores localizados em estações no solo, cujas posições não são necessariamente conhecidas (WOLF, 2007).

As distâncias são determinadas no monitoramento GNSS tomando-se observações nestes sinais transmitidos pelos satélites em diferentes frequências. Dois procedimentos observacionais diferentes são utilizados: através das observações de pseudodistância e/ou pelas observações de fase da onda portadora (WOLF, 2007).

No GNSS, a pseudodistância é determinada através da correlação entre o código PRN (*pseudorandom noise*) gerado pelo satélite no instante da transmissão do sinal (t^t) e a sua réplica gerada no receptor no instante de recepção do sinal (t_r). Segundo Monico (2008), a fórmula da pseudodistância é dada pela Eq. 2.1.1:

$$PD_r^s = \rho_r^s + c(dt_r - dt^s) + \mathrm{Ion}_r^s + Trop_r^s + dm_r^s + Orb_r^s + \nu_{PD_r^s} \qquad \text{Eq. 2.1.1}$$

Sendo:

PD_r^s – pseudodistância entre o satélite *s* e o receptor *r*;

ρ_r^s – distância geométrica entre o satélite e o receptor;

c – velocidade da luz no vácuo;

dt_r – erro do relógio do receptor;

dt^s – erro do relógio do satélite;

Ion_r^s – erro causado pelos efeitos da ionosfera;

$Trop_r^s$ – erro caudado pelos efeitos da troposfera;

dm_r^s – erro causados pelo multicaminho;

Orb_r^s – erros provenientes da órbita do satélite;

$\nu_{PD_r^s}$ – erros devidos aos efeitos aleatórios e não modelados.

A medida da fase da onda portadora é obtida a partir da diferença entre a fase gerada pelo satélite e a fase da sua réplica gerada pelo receptor. Segundo Monico (2008), a fórmula da fase da onda portadora é dada pela Eq. 2.1.2:

$$\phi_r^s(t) = f\left(\frac{\rho_r^s - \mathrm{Ion}_r^s + Trop_r^s + dm_r^s + Orb_r^s}{c}\right) + f(dt_r - dt^s) + \left(\phi^s(t_0) - \phi_r(t_0)\right) + N_r^s + \nu_{\phi_r^s} \qquad \text{Eq. 2.1.2}$$

Sendo:

f – frequência nominal da fase;

$\phi^s(t_0)$ – fase inicial no satélite na época de referência t_0;

$\phi_r(t_0)$ – fase inicial no receptor na época de referência t_0;

N_r^s – número inteiro desconhecido de ciclos da onda portadora, denominado ambiguidade;

$\nu_{\phi_r^s}$ – erro da fase da portadora devido aos efeitos aleatórios e não modelados.

No método da fase da onda portadora, como as observações pautam-se nas mudanças de fase ocorridas durante a viagem do sinal desde o satélite até o receptor, há a necessidade de se usar as técnicas de diferença entre as observações de fase (MONICO, 2008).

2.2 Posicionamento Relativo pelo GNSS

Segundo Monico (2008), o posicionamento se refere à determinação da posição de objetos com relação a um referencial específico. O posicionamento é dito absoluto quando as coordenadas estão associadas ao geocentro, enquanto no posicionamento relativo as coordenadas são determinadas com

relação a um referencial materializado por um ou mais vértices com coordenadas “conhecidas”.

O método de posicionamento GNSS utilizado em redes geodésicas para determinar os vetores linhas de base entre duas ou mais estações é o *Posicionamento Relativo*.

Uma vantagem do posicionamento relativo é a facilidade em reduzir os erros existentes nas observações por meio de combinações de diferenças entre as observações das estações (MONICO, 2008). As principais delas são: simples diferença e dupla diferença. A simples diferença é feita através da diferença entre duas observações simultâneas de duas estações, a fim de eliminar o erro do relógio do satélite. Esta técnica permite a eliminação do erro do relógio do satélite. A dupla diferença, por sua vez, é realizada por meio da subtração entre duas simples diferenças de dois receptores. Esta técnica elimina o erro do relógio do receptor e outros erros sistemáticos (WOLF, 2007).

Na prática, são observados quatro satélites ou mais simultaneamente usando dois ou mais receptores em estações no solo. E estas observações são repetidas inúmeras vezes, gerando uma quantidade numerosa de observações redundantes.

O posicionamento utilizando as observações de pseudodistância é preferível para fins de navegação, pois estas geram posições de forma rápida e com precisão satisfatória. No entanto, para trabalhos de levantamento que exijam uma maior precisão, opta-se pelas observações da fase da onda portadora. Desta forma, quando o objetivo é realizar o ajustamento de uma rede GNSS, é habitual que se utilize as medidas provenientes das observações de fase da onda portadora, justamente por esta fornecer uma maior precisão nas observações (WOLF, 2007).

O posicionamento relativo pode ser subdividido em distintos métodos. Por exemplo, o posicionamento relativo estático, cinemático, semicinemático, *stop and go* , etc. Tais métodos utilizam a pseudodistância e/ou a fase da onda portadora para determinar a posição de um ponto em relação a outros pontos de coordenadas conhecidas.

O vetor que liga duas estações GNSS pode ser expressado matematicamente como a diferença entre as coordenadas das posições ocupadas por estas estações. Esse vetor é chamado de *vetor linha de base*.

Então, partindo de uma coordenada conhecida, são determinadas as três componentes (ΔX, ΔY e ΔZ) da linha de base que liga o ponto conhecido ao ponto cujo se desconhece as coordenadas. Com as componentes da linha de base estimada, de acordo com a Figura 1, o ponto desconhecido é estimado acrescentando estas componentes às coordenadas do ponto da estação de referência com coordenadas conhecidas.

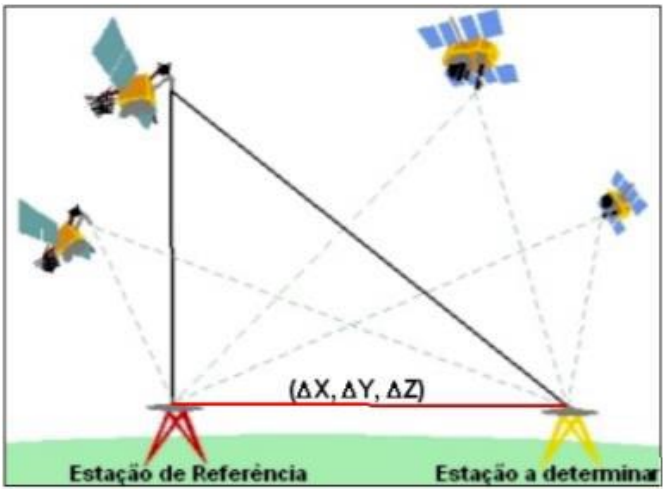

Figura 1 Esquematização do Posicionamento Relativo. Fonte: IBGE (2015)

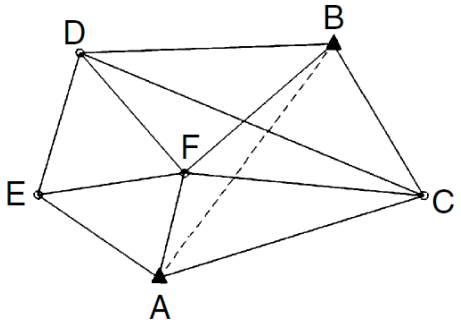

Figura 2 Rede GNSS. Fonte: WOLF (2007)

Como o exemplo apresentado na Figura 1, tomem-se duas estações A e B usadas em uma sessão de observação, sendo A um ponto de controle, e B sendo um ponto de posição desconhecida. A sessão de trabalhos gera as

componentes ΔX_{AB}, ΔY_{AB} e ΔZ_{AB} do vetor linha de base. As coordenadas da estação B podem então ser determinadas adicionando-se as componentes da linha de base às coordenadas do ponto de controle A. Este procedimento é conhecido como posicionamento relativo, já que as observações da fase da portadora e pseudodistância não fornecem diretamente a posição da estação desejada, mas parte-se de um ponto conhecido e distâncias relativas a este ponto de controle. As coordenadas do ponto B do exemplo, podem ser obtidas através do sistema de equações:

$$X_B = X_A + \Delta X_{AB} \qquad \text{Eq. 2.2.1}$$

$$Y_B = Y_A + \Delta Y_{AB} \qquad \text{Eq. 2.2.2}$$

$$Z_B = Z_A + \Delta Z_{AB} \qquad \text{Eq. 2.2.3}$$

É habitual que na prática mais de dois receptores sejam utilizados simultaneamente no posicionamento relativo, fornecendo mais de uma linha de base. Após a primeira sessão de observação, outros pontos são ocupados movendo os receptores a estações próximas, mantendo pelo menos um receptor na posição anterior. Desta forma, cria-se uma rede GNSS, exemplificada na figura 2, onde os pontos A e B são de controle e os pontos C, D, E e F são pontos de coordenadas desconhecidas. (WOLF, 2007).

2.3 MMQ no ajuste de observações

2.3.1 Princípios do MMQ

Seja a seguinte equação linear:

$$A.X = L = L_0 - L_b \qquad \text{Eq. 2.3.1.1}$$

Em que:
A representa a matriz jacobiana;
X representa o vetor de correção dos parâmetros;

L_0 representa a função dos parâmetros de aproximação inicial;
L_b representando o vetor de observações.

No entanto, como as observações possuem erros de medidas que fazem o sistema inconsistente com o modelo matemático, inclui-se ao vetor de observações o vetor V de resíduos, transformando a equação 2.3.1.1 em:

$$AX = L + V = L_a \qquad \text{Eq. 2.3.1.2}$$

Em que:
L_a representa o vetor de observações ajustadas.

Desta forma, proposto por Gauss e Legendre, o objetivo do MMQ é assumir como a melhor estimativa para o vetor de parâmetros X o valor que torna *mínima a soma dos quadrados dos resíduos*. Escrevendo isto na forma matricial, tem-se:

$$V^T PV = mín \qquad \text{Eq. 2.3.1.3}$$

Com P representando a matriz P que contém informações e características da estatística dos resíduos.

2.3.2 Modelo Paramétrico

Sejam as seguintes equações,

$$V = L_a - L_b \qquad \text{Eq. 2.3.2.1}$$

e

$$X_0 = X_a - X \qquad \text{Eq. 2.3.2.2}$$

Em que:

V representa o vetor de resíduos;

L_a representa o vetor de observações ajustadas;

X_0 representa o vetor de valores aproximados para os parâmetros;

X_a representa o vetor de parâmetros ajustados;

O modelo paramétrico do MMQ se configura quando é possível expressar o vetor de observações ajustadas através de uma expressão matemática:

$$F(X_a) = L_a \qquad \text{Eq. 2.3.2.3}$$

Substituindo os valores da Eq. 2.5.2.2 na Eq. 2.5.2.3:

$$F(X_0 + X) = L + V \qquad \text{Eq. 2.3.2.4}$$

Sabendo que uma aproximação linear de uma função pode ser dada pela fórmula de Taylor:

$$F(X) = F(X') + \frac{\partial F}{\partial X}|_{X'} . (X - X') \qquad \text{Eq. 2.3.2.5}$$

Desenvolvendo a Eq.2.4.2.4 utilizando a fórmula de Taylor, tem-se:

$$F(X_0 + X) = L + V = F(X_0) + \frac{\partial F}{\partial X}|_{X_0} . X \qquad \text{Eq. 2.3.2.6}$$

Por fins de simplificação, pode-se substituir os termos $F(X_0)$ e $\frac{\partial F}{\partial X}|_{X_0}$, respectivamente por, L_0 e A. Em que L_0 representa a função dos parâmetros de aproximação inicial e A representa a matriz de derivadas parciais ou a matriz Jacobiana.

Assim, o lado direito da Eq.2.4.2.6 se reduz a:

$$L + V = L_0 + AX \qquad \text{Eq. 2.3.2.7}$$

$$V = AX + L_0 - L \qquad \text{Eq. 2.3.2.8}$$

Substituindo $L_0 - L$ por L_b representando o vetor de observações, obtêm-se o modelo matemático linearizado:

$$V = AX + L_b \qquad \text{Eq. 2.3.2.9}$$

Seguindo no objetivo do MMQ de minimizar a expressão $V^T PV$:

$$\lambda = V^T PV = (AX + L_b)^T P(AX + L_b) \qquad \text{Eq. 2.3.2.10}$$
$$= X^T A^T PAX + 2X^T A^T PL_b + {L_b}^T PL_b$$

Fazendo

$$\frac{d\lambda}{dX} = 0 \qquad \text{Eq. 2.3.2.11}$$

$$A^T PAX + A^T PL_b = 0 \qquad \text{Eq. 2.3.2.12}$$

E finalmente

$$X = -(A^T PA)^{-1} A^T PL_b \qquad \text{Eq. 2.3.2.13}$$

Fazendo

$$N = A^T PA \qquad \text{Eq. 2.3.2.14}$$

$$U = A^T PL_b \qquad \text{Eq. 2.3.2.15}$$

Tem-se que

$$X = -N^{-1} U \qquad \text{Eq. 2.3.2.16}$$

Seja a MVC das observações Σ_{L_b}, a matriz que sumariza todas as variâncias e covariâncias entre as observações do vetor L_b. A partir desta e do valor da variância a priori σ_0^2, pode-se definir a matriz de pesos como:

$$P = \sigma_0^2 \Sigma_{L_b}^{-1} \qquad \text{Eq. 2.3.2.17}$$

Ao fim do ajustamento, através da lei de propagação de variâncias, pode-se definir as MVC's das variáveis $X, X_a, V\ e\ L_a$, representadas pelas Eq. 2.3.2.18 a Eq. 2.3.2.19. Para mais detalhes da demonstração, consultar GEMAEL (2008).

$$\Sigma_X = \sigma_0^2 N^{-1} \qquad \text{Eq. 2.3.2.18}$$

$$\Sigma_{X_a} = \sigma_0^2 N^{-1} \qquad \text{Eq. 2.3.2.19}$$

$$\Sigma_V = \sigma_0^2 (AN^{-1}A^T - P^{-1}) \qquad \text{Eq. 2.3.2.20}$$

$$\Sigma_{L_a} = \sigma_0^2 (AN^{-1}A^T) \qquad \text{Eq. 2.3.2.21}$$

A variância a priori σ_0^2 é habitualmente arbitrada, já que esta definição não influenciará no vetor das parâmetros X. Por isso, é interessante realizar uma escolha de variância a priori que traga vantagens computacionais (GEMAEL, 2008).

Já a variância a posteriori para o modelo paramétrico é definida como:

$$\hat{\sigma}_0^2 = \frac{V^T P V}{n - u} \qquad \text{Eq. 2.3.2.22}$$

Em que:

n e u representam, respectivamente, o número de observações e o número de parâmetros.

2.4 Ajustamento de Redes GNSS

Uma rede GNSS pode ser definida como o conjunto de estações que foram ocupadas em diferentes sessões de um projeto de levantamento de observações GNSS. Como uma rede GNSS possui observações consideradas redundantes, faz-se necessário o ajustamento destas observações de tal forma que se garanta uma diferença de coordenadas (geometria) consistente (WOLF, 2007). Os vetores linha de base são utilizadas como entrada (observações) no ajustamento da rede GNSS. O ajustamento utilizará as observações independentes destes vetores linha de base e as coordenadas conhecidas de estações serão consideradas livre de erros e fixas, ou seja, injunções absolutas.

No entanto, em uma rede GNSS com um número superior a duas estações, a combinação de linhas de bases possíveis inclui linhas de bases dependentes. Ou seja, no espaço amostral há a presença de linhas de bases que podem ser expressas como uma combinação linear de outras linhas de bases.

Segundo Seeber (2003), uma regra geral para determinar o número de linhas de base totais e independentes é definido pelas seguintes expressões matemáticas:

$$\frac{r.(r-1)}{2} = n° \; de \; linhas \; de \; base \; totais \qquad \text{Eq. 2.4.1}$$

$$(r-1) = n° \; de \; linhas \; de \; base \; independentes \qquad \text{Eq. 2.4.2}$$

Para um levantamento multi-sessão, o número de sessões necessárias é dado por:

$$s = \frac{n-m}{r-m} \qquad \text{Eq. 2.4.3}$$

Em que:

s - número de sessões;

r - número de receptores;

n - número de estações;

m - número de estações com mais de um observação entre duas sessões diferentes.

Por fim, ao se obter as observações de linhas de base ao final do processamento dos dados GNSS, realiza-se o ajustamento pelo modelo paramétrico do MMQ (SILVA, 2007). O modelo matemático do levantamento de rede GNSS é representado pelo seguinte conjunto de equações:

$$\Delta X_{ij} = X_j - X_i \qquad \text{Eq. 2.4.4}$$

$$\Delta Y_{ij} = Y_j - Y_i \qquad \text{Eq. 2.4.5}$$

$$\Delta Z_{ij} = Z_j - Z_i \qquad \text{Eq. 2.4.6}$$

Em que do lado esquerdo das equações se encontram as observações e do lado direito se encontram os parâmetros. Com X, Y e Z representando as componentes do sistema de referência e *i* e *j* representando estações.

2.5 Controle de Qualidade

O controle de qualidade tem como objetivo assegurar a qualidade de um produto para determinadas condições de utilização. Com base nas técnicas de ajustamento, é o possível detectar a presença de erros grosseiros. Além disso, com uso de computação para operar matrizes de grandes dimensões, pode-se obter informações sobre confiabilidade e qualidade de resultados (CAMARGO, 2000).

O modelo matemático envolvido em um ajustamento é composto pelo modelo estocástico, descrevendo as propriedades estatísticas das observações, e o modelo funcional, relacionando as observações e os parâmetros desconhecidos do modelo (SILVA, 2007).

A precisão das coordenadas obtidas em um levantamento GNSS é expressa pela MVC, obtida através da propagação de erros aleatórios. Já a confiabilidade dos resultados refere-se à capacidade em detectar a presença de erros na modelagem e no próprio levantamento (TEUNISSEN, 1998).

Após terem sido realizados testes, como por exemplo os testes Baarda e Tau, busca-se retirar a influência do erro do resultado do ajustamento, esta etapa é denominada *adaptação* (SILVA, 2007).

2.5.1 Teste estatístico qui-quadrado

Para analisar a qualidade do ajustamento, passa-se pelas etapas de detecção, identificação e adaptação. Ou seja, busca-se detectar a presença de erros grosseiros, identificar estes erros e, por fim, adaptá-los a fim de eliminá-los ou reduzi-los. Neste texto, dar-se-á ênfase a etapa de detecção de erros grosseiros através do teste qui-quadrado.

Ao se arbitrar um valor para a variância (σ_0^2), não se ocasiona efeitos ao ajustamento. Como este valor é atribuído antes do ajustamento com o objetivo de fixar os pesos, denomina-se *variância a priori.* Com o ajustamento concluído, estima-se um valor para a variância em função dos resíduos, esta variância é denominada *variância a posteriori* ($\hat{\sigma}_0^2$) (GEMAEL, 2003).

O teste qui-quadrado baseia-se em fazer uma análise de qualidade do ajustamento, utilizando os valores de σ_0^2 e $\hat{\sigma}_0^2$, com o objetivo de detectar erros presentes nas observações e/ou nos modelos. O teste leva a elaboração de duas hipóteses, a básica (H_0) e a alternativa (H_1), apresentadas da seguinte forma:

$$H_0 : E\{\hat{\sigma}_0^2\} = \sigma_0^2 \qquad contra \qquad H_1 : E\{\hat{\sigma}_0^2\} > \sigma_0^2 \qquad \text{Eq. 2.5.1.1}$$

Então, calcula-se o valor de:

$$\chi_c^2 = \frac{\hat{\sigma}_0^2}{\sigma_0^2} \upsilon \qquad \text{Eq. 2.5.1.2}$$

Em que υ representa o número de graus de liberdade.

Rejeitar-se-á a hipótese se:

$$\chi_c^2 \leq \chi^2_{\upsilon,\frac{\alpha}{2}} \qquad ou \qquad \chi_c^2 \geq \chi^2_{\upsilon,1-\frac{\alpha}{2}} \qquad \text{Eq. 2.5.1.3}$$

Habitualmente utiliza-se o nível de significância $\alpha = 5\%$. A aceitação da hipótese é representada na Figura 3.

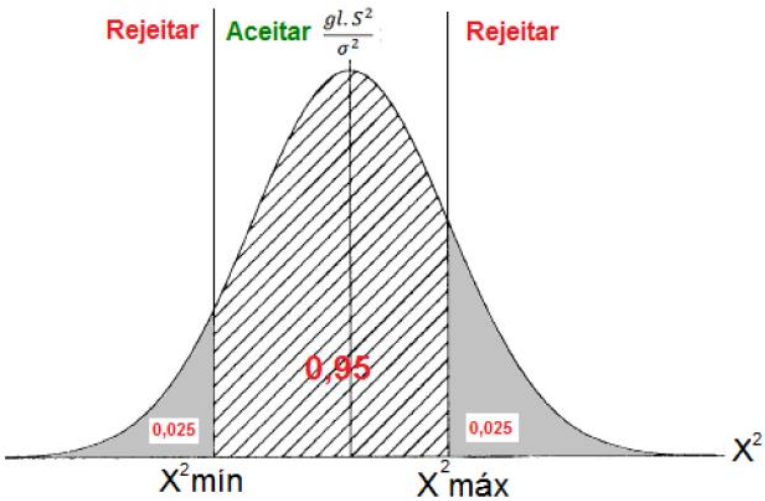

Figura 3 Teste qui-quadrado para valor de significância de 5%. Fonte: ALVES (2016)

2.5.2 Elipse e Elipsoide de Erro

Depois de realizado um ajustamento utilizando o MMQ, os desvios-padrões das coordenadas de uma estação podem ser calculados a partir da MVC. Estes desvios-padrões fornecem os erros estimados em cada direção do referencial utilizado. Representando graficamente, pode-se criar um retângulo, cujas dimensões sejam o dobro dos desvios-padrões. Este retângulo é denominado *retângulo de erros,* conforme mostrado na Fig. 4 (WOLF, 2007).

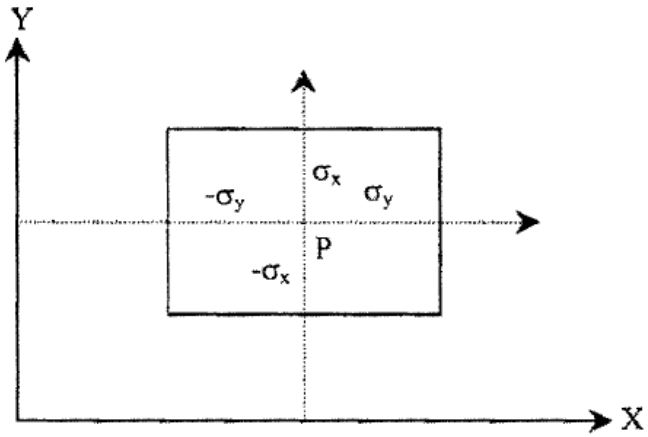

Figura 4 Retângulo de Erros. Fonte: CAMARGO (2000)

Na teoria, busca-se uma elipse que seja internamente tangente aos lados do retângulo e que, ainda, possua os semieixos maior e menor, respectivamente, máximo e mínimo. Esta elipse é denominada *elipse padrão* (CAMARGO, 2000).

A orientação da elipse padrão pode ser obtida através da MVC, segundo as expressões:

$$\Sigma_{xy} = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{yx} & \sigma_y^2 \end{bmatrix} \qquad \text{Eq. 2.5.2.1}$$

Aplicando a propagação de variâncias, a fim de se obter a MVC das novas coordenadas:

$$\Sigma_{x'y'} = D\Sigma_{xy}D^T \qquad \text{Eq. 2.5.2.2}$$

Sendo, a matriz de rotação de eixos:

$$D = \begin{bmatrix} \cos(t) & sen(t) \\ -sen(t) & \cos(t) \end{bmatrix} \qquad \text{Eq. 2.5.2.3}$$

Após o produto matricial da Eq. 2.7.2.3.2 e algumas operações algébricas e trigonométricas, estas detalhadas em Camargo (2000), chega-se em:

$$\sigma_{x'} = \sqrt{\frac{1}{2}[\sigma_x^2 + \sigma_y^2 + 4\sigma_{xy}^2 + \left(\sigma_x^2 - \sigma_y^2\right)^2]}$$ Eq. 2.5.2.4

$$\sigma_{y'} = \sqrt{\frac{1}{2}[\sigma_x^2 + \sigma_y^2 - 4\sigma_{xy}^2 - \left(\sigma_x^2 - \sigma_y^2\right)^2]}$$ Eq. 2.5.2.5

As Eq. 2.7.2.3.4 e Eq. 2.7.2.3.5 fornecem, respectivamente, os semieixos maiores e menores da *elipse de erros* (Fig. 5).

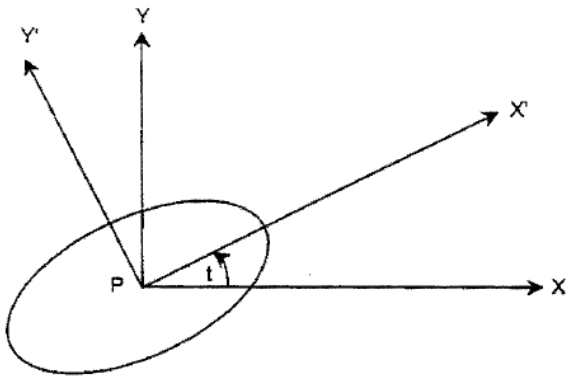

Figura 5 Elipse de erros em relação aos sistemas XY e X'Y'. Fonte: CAMARGO (2000)

A elipse de erro pode fornecer informações importante acerca da precisão da posição de uma estação. Outra vantagem da elipse de erros é que ela fornece um método comparativo entre as precisões relativas entre duas estações. Comparação que pode ser feita de maneira visual, analisando as orientações, tamanhos e formas das elipses em questão (WOLF, 2007).

Quando se tratar das coordendas ajustadas de uma estação, tem-se valores de variância para as três componentes desta específica coordenada. Sendo assim, os erros para a posição da estação são melhores representados através de um elipsoide de erro.

Seja a MVC da coordenada ajustada da estação dada pela Eq. 2.5.2.6:

$$\Sigma_A = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_y^2 & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_z^2 \end{bmatrix} \qquad \text{Eq. 2.5.2.6}$$

Os semieixos do elipsoide de erro são, segundo Camargo (2000), dados por:

$$\begin{bmatrix} a_A \\ b_A \\ c_A \end{bmatrix} = \begin{bmatrix} \sqrt{\lambda_1} \\ \sqrt{\lambda_2} \\ \sqrt{\lambda_3} \end{bmatrix} \qquad \text{Eq. 2.5.2.7}$$

Em que $\lambda_1, \lambda_2, \lambda_3$ são soluções de:

$$\det(\Sigma_A - I\lambda) = 0 \qquad \text{Eq.2.5.2.8}$$

2.6 Implementação Computacional

2.6.1 Linguagem Python adaptada ao ajustamento

Python é uma linguagem de programação orientada a objeto de altíssimo nível, interativa, interpretada e está categorizada entre os softwares livres. Seguindo a definição de softwares livres, a *Python Software Foundation* (PSF) exige apenas o aviso de créditos durante o uso e distribuição, sejam elas comerciais ou recreativos, das aplicações desenvolvidas com Python (PSF, 2018).

Já em relação a sua caracterização como linguagem interpretada, diferentemente das linguagens compiladas, elas transformam o código fonte em uma linguagem intermediária (específica de cada linguagem), que é interpretada pela máquina virtual. O número de interpretadores disponíveis pela PSF é tão vasto quanto às possíveis aplicações de Python, desde web e bancos de dados, até aplicações acadêmico-científicas (PSF, 2018).

Apesar da vantagem da implementação livre, a busca de determinadas funcionalidades exige um proporcional trabalho de prospecção de bibliotecas.

Este fator se torna ainda mais crítico se levarmos em conta que não existe um repositório oficial.

Módulos Python utilizados

Para o desenvolvimento do código utilizado neste trabalho, são utilizados os seguintes módulos Python:

- Numpy – para operações matemáticas, principalmente para operações matriciais;
- Matplotlib – para construções de gráficos e *plotagem* de elipses de erros;
- Scipy – para testes estatísticos;
- Pandas – para manipulação de arquivos *.csv*.

2.6.2 Desenvolvimento Web

De acordo com o maior site sobre desenvolvimento web, *W3Schools*, o Web *Development* é um termo que abrange o amplo espectro relacionado ao desenvolvimento de uma página para a Internet (*Word Wide Web*) ou uma Intranet (rede privada), ou seja, toda a programação que possibilita a funcionalidade de um website, de acordo com os requerimentos do proprietário. Normalmente o termo é utilizado em um sentido mais estrito, para se referir a todas as atividades não relacionada diretamente com o aspecto do design.

No desenvolvimento da aplicação *web* em questão, optou-se pelo Flask por se tratar de um *framework* com vasta teoria disponível e amplo uso entre os desenvolvedores de plataformas web. Além de ser considerado mais simples que outros *frameworks,* como, por exemplo, o *Web2Py.*

Flask é um *webframework* baseado na biblioteca *WSGI Werkzeug* e na biblioteca de *Jinja2*. Ele fornece aos usuários ferramentas, bibliotecas e tecnologias que lhe permitem construir aplicações web. Essas aplicações podem ser páginas *web*, *blog*, *wiki* ou utilitários como uma aplicação agenda online e sites comerciais.

O Flask é categorizado como *microframework*, pois tem um núcleo que é simples, porém estendível. Nele, não há camada de abstração do banco de dados, validação de formulários, nem outro componente onde bibliotecas de terceiros existem para prover funcionalidade, ou seja, o Flask tem pouca dependência de bibliotecas externas. Há vantagens e desvantagens nesta característica. Entre as vantagens, podemos citar o fato do framework ser leve, e, portanto, há pouca ou nenhuma necessidade de atualizar e monitorar bugs de segurança. Entre as desvantagens, ressalta-se o fato de em determinado momento demandar mais trabalho para a adição de *plugins*. Ainda de acordo com o *site* W3Schools, o *Flask* está apto a receber extensões que adicionem essas funcionalidades na aplicação final.

3. Descrição da plataforma

3.1 Especificações Técnicas

O produto final deste trabalho corresponde a uma página na *internet* cujo objetivo é realizar um ajustamento de uma rede GNSS e exibir online determinadas características do ajuste realizado.

Detalhes sobre o site:

I) Título: ajustgnssnet

II) Domínio do site: http://ajustgnssnet.pythonanywhere.com/

III) Data de conclusão de elaboração: 30/09/2018

IV) Responsáveis pela elaboração:

a) Gabriel Santos do Amaral

b) Ítalo de Lanna Arcanjo

c) Mendson Cabral Amenta Boaz

V) Organização: Instituto Militar de Engenharia

VI) Tamanho total do site: 5,13Mb

VII) Requisitos de software no servidor:

a) Interpretador: Python 3.6.4

b) Framework: Flask

VIII) Mapa do site:

a) Sobre

b) Teoria e Exemplo

c) Ajuste de Rede GNSS

d) Código

3.2 Entrada de dados

Já na plataforma *online*, é disponibilizada uma tabela padrão para *download* e em formato .csv. Conforme o exemplo da Figura 6:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	From	To	DX	VX	DY	VY	DZ	VZ	CtrlSt	X	Y	Z	Var_a_priori
2	1	3	11644.2232	0.00669	3601.2165	0.00203	3399.255	0.03082	1	402.3509	-4652995.301	4349760.778	1
3	1	5	-5321.7164	0.02645	3634.0754	0.00582	3173.665	0.01068	2	8086.032	-4642712.847	4360439.083	
4	2	3	3960.5442	0.00478	-6681.2467	0.01153	-7279.01	-0.00511					
5	2	4	-11167.6076	-0.00731	-394.5204	-0.00136	-907.959	-0.00194					
6	4	3	15128.1647	-0.00081	-6286.7054	-0.00801	-6371.06	-0.00037					
7	4	5	-1837.7459	-0.01005	-6253.8534	0.00268	-6596.67	0.00109					
8	6	1	-1116.4523	0.00198	-4596.161	0.00524	-4355.9	-0.01563					
9	6	3	10527.7852	-0.00563	-994.9377	0.00047	-956.625	-0.0014					
10	6	5	-6438.1364	-0.00387	-962.0694	-0.00514	-1182.23	-0.00545					
11	6	4	-4600.3787	-0.00561	5291.7785	-0.00232	5414.431	0.00156					
12	6	2	6567.2311	-0.00051	5686.2926	0.00534	6322.392	0.0022					
13	2	6	-6567.231	0.00041	-5686.3033	0.00536	-6322.38	-0.0132					
14	1	6	1116.4577	-0.00738	4596.1553	0.00046	4355.914	-0.00227					

Figura 6 Exemplo de planilha, com valores referentes ao problema do Apêndice A, que é disponibilizada online.

Efetuado o *download* do arquivo *.csv,* o usuário deverá manipular a tabela e inserir valores de:

- Identificação das estações;
- Componentes dos vetores linha de base entre as estações;
- Desvio-padrão de cada componente do vetor linha de base;
- Componentes das coordenadas das estações de controle;
- Variância a priori.

Após o preenchimento do arquivo *.csv*, o usuário retornará a planilha à plataforma online, via *upload,* para o ajustamento da rede GNSS ser concluído.

Por fim, são exibidas as coordenadas ajustadas das estações, as análises referentes ao teste qui-quadrado e plotagem das elipses de erros, com o objetivo de proporcionar ao usuário uma avaliação da qualidade do ajustamendo realizado.

Para que a ferramenta online funcione em sua totalidade, algumas prescrições do uso da tabela devem ser pré-definidas. São elas:

- As estações devem ser identificadas de forma numérica. Esta condição foi definida para facilitar a leitura de dados e também para admitir um valor indefinido de estações;
- As casas decimais dos valores inseridos devem ser necessariamente separadas por ponto (.) e não há restrição na quantidade de decimais utilizados;
- A variância *a priori* padrão estabelecida é igual ao valor unitário. Podendo, no entanto, ser alterada pelo usuário.

O preenchimento da planilha (em formato *.csv*) utilizada para o ajuste, disponibilizada em http://ajustgnssnet.pythonanywhere.com/exemplo, deverá seguir o seguinte padrão:

- As colunas A e B da planilha deverão ser preenchidas com os identificadores das estações. Sendo a coluna A para as estações de início da linha de base e a coluna B para as estações de fim da linha de base;
- As colunas C, E e G abrigam, respectivamente, as componentes X, Y e Z de cada linha de base;
- As colunas D, F e H são preenchidas, respectivamente, com os valores dos desvios-padrões das componentes X, Y e Z de cada uma das linhas de base observadas;
- A coluna I é reservada para a identificação numérica das estações de controle, aquelas que são consideradas injunções absolutas no ajuste;
- As colunas J, K e L são, respectivamente, as componentes X, Y e Z das coordenadas das estações de controle;
- A coluna M é preenchida com o valor da variância *a priori* a ser utilizada no ajuste.

3.3 Interface da plataforma *online*

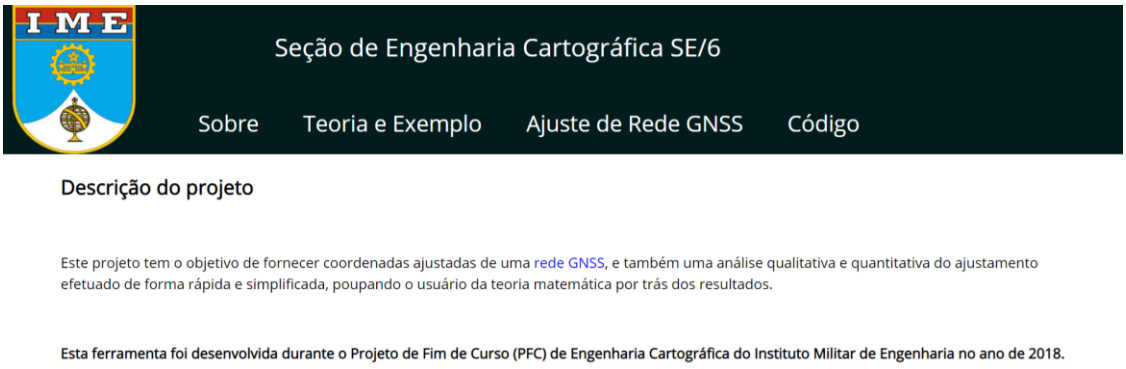

Figura 7. Interface da página inicial do site desenvolvido

Procurou-se desenvolver uma interface mais simples possível para possibilitar que os mais distintos usuários utilizassem a plataforma sem grandes dificuldades.

No menu superior, exposto na Figura 7, encontra-se as seguintes opções de página:

- Sobre – onde o usuário encontrará uma breve descrição do projeto;
- Teoria e Exemplo – onde o usuário poderá ler uma breve teoria relacionada ao ajuste e também realizar o download da planilha padrão;
- Ajuste de Rede GNSS – onde o usuário realizará o upload da planilha já preenchida com os valores de observações;
- Código – onde o usuário poderá ter acesso livre ao código *python* desenvolvido.

3.4 Ajustamento *online*

Após o usuário realizar o upload da planilha já preenchida, os resultados do ajustamento aparecerão de maneira automática na tela. Desta forma, são listados os seguintes conteúdo:

- Coordenadas ajustadas – este é o objetivo principal do ajuste, ou seja, determinar as coordenadas após o ajustamento da rede GNSS.
- Teste Qui-Quadrado – é realizado um teste de hipótese, com o objetivo de detectar a presença de erros grosseiros no ajuste a uma significância de 5%. Desta forma, se o ajuste for considerado livre de erros grosseiros, aparecerá tal informação em azul (Figura 11). Caso haja detecção de erros, aparecerá uma mensagem em vermelho (Figura 12).
- Elipses de erros – também são exibidas as elipses de erros relacionadas a cada uma das estações com coordenadas ajustada, permitindo o usuário a perceber de forma visual qual das estações apresentaram um maior erro durante o ajuste.
- MVC dos parâmetros ajustados – permitindo o usuário identificar os valores de variância atribuidos a cada componente das coordenadas ajustadas.

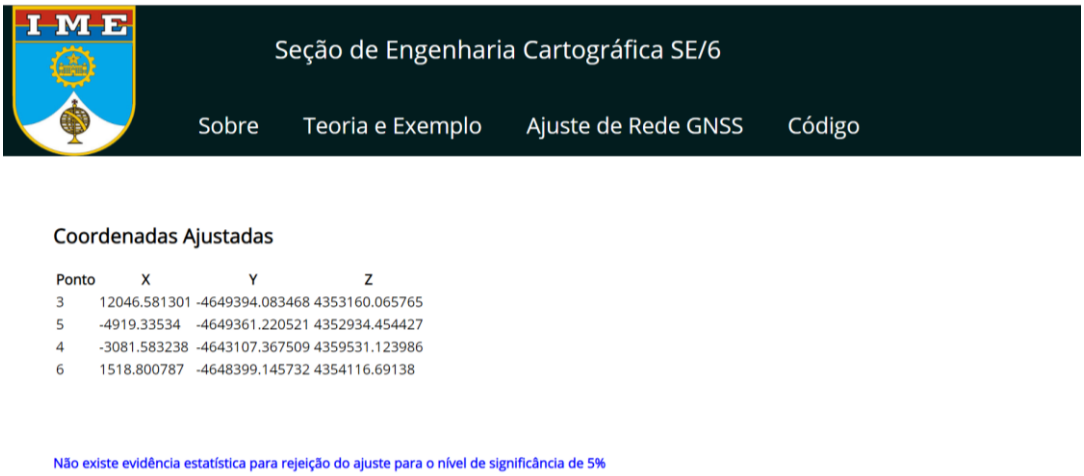

Figura 8. Exemplo de um resultado sem evidências de erros grosseiros

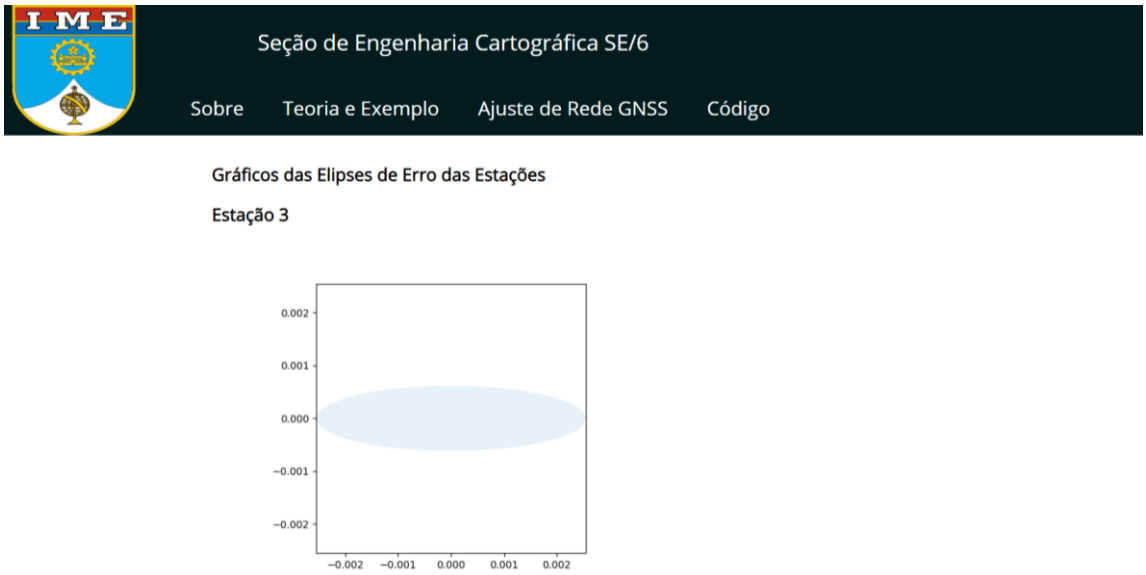

Figura 9. Elipse de erro encontrada para uma das estações utilizadas no exemplo da Fig. 8

Figura 10. Exemplo de um resultado com evidências de erros grosseiros.

Ainda, caso haja algum erro durante o processo, principalmente erros advindos de um incorreto preenchimento da planilha padrão, surgirá uma página informando erro.

3.5 Hospedagem

Optou-se pela hospedagem utilizando os serviços gratuitos da plataforma *PythonAnywhere.com.* Além disso, oferece-se suporte Python a todas bibliotecas utilizadas no algoritmo de ajustamento.

Todo o projeto se encontra disponibilizado no endereço http://ajustgnssnet.pythonanywhere.com/, permitindo que o usuário utilize suas ferramentas de ajustamento de redes GNSS e obtenha dados distintos sobre a estatística e qualidade do processo.

Além disso, o usuário encontra disponível no *site* o código, em *python*, utilizado e também este texto.

4. CONCLUSÃO

Ao fim deste projeto, obteve-se uma ferramenta que poderá ser útil a diversos estudantes da área de Geodésia e Ajustes de Observações, servindo de apoio na compreensão da teoria que envolve um ajustamento de uma rede GNSS. Apesar da teoria por trás da plataforma online esteja fortemente relacionada à área de Geodésia, observou-se um forte viés computacional no desenvolvimento deste projeto.

Para trabalhos futuros, sugere-se a complementação dos testes de qualidade desenvolvidos. Ou seja, adicionar à ferramenta os testes Tau e Baarda para identificar os erros grosseiros e também agregar a etapa de adaptação de erros na análise de qualidade.

5. REFERÊNCIAS BIBLIOGRÁFICAS

ALVES, H. D. A. **Implementação de sistema para cálculos geodésicos em ambiente computacional livre: Ajustamento de rede de nivelamento geométrico**. Rio de Janeiro, 2016. Projeto de Fim de Curso de Graduação em Engenharia Cartográfica. IME/RJ.

CAMARGO, P. O. **Ajustamento de Observações**. Presidente Prudente, 2000. Notas de Aula do Curso de Graduação em Engenharia Cartográfica. FCT/UNESP.

GEMAEL, Camil. **Introdução ao ajustamento de observações: Aplicações geodésicas**. Curitiba. UFPR, 1994. Reimpressão 2004. 428p.

GEMAEL, Camil; ANDRADE, José Bittencourt. **Geodésia Celeste**. Curitiba. UFPR, 2003. 389p.

MONICO, João Francisco Galera. **Posicionamento pelo GNSS**. 2ª ed. Unesp, 2008. 476p.

PSF: Portal Python software Foundation. Disponível em <https://www.python.org/ > Acesso em 11/09/2018.

SILVA, H. A. **Adequação de softwares comerciais às exigências da Lei 10.267/2001**. Relatório de iniciação científica. Faculdade de Ciências e Tecnologia, Universidade Estadual Paulista, Presidente Prudente**,** 2007.

SEEBER, G. **Satellite Geodesy: Foundations, Methods, and Applications.** Berlin, New York: Walter de Gruyter, 2003.

TEUNISSEN, P. J. G., KLEUSBERG, A. **GPS for Geodesy.** 2. ed. Berlin: Springer-Verlag, 1998.

WOLF, Paul R.; GHILANI, Charles D. **Adjustment Computatios: Spatial Data Analysis**. 4ª ed. JOHN WILEY & SONS, 2007. 611p.

W3Schools – The world's largest web developer site. Disponível em <http://www.w3schools.com/default.asp > Acesso em 11/09/2018.

APÊNDICE A. Exemplo de ajustamento de rede GNSS (Fonte: WOLF, 2007)

Seja a rede GNSS representada na Figura 10. Foram realizadas 13 observações de linhas de base, e como cada linha de base possui 3 componentes, gera-se um total de 39 observações.

Sabendo ainda que as coordenadas A e B são conhecidas e são consideradas fixas e perfeitas (injunções absolutas) ao longo do ajustamento, tem-se um total de $39 - 12 = 27$ observações redundantes nesta rede.

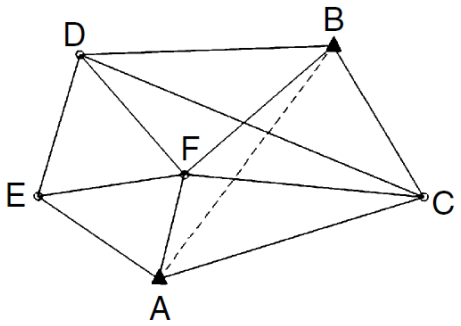

Figura 11 Exemplo de rede GNSS

Tabela A.1 Observações de linha de base

De	Para	ΔX	ΔY	ΔZ
A	C	11644,2232	3601,2165	3399,2550
A	E	–5321,7164	3634,0754	3173,6652
B	C	3960,5442	–6681,2467	–7279,0148
B	D	–11167,6076	–394,5204	–907,9593
D	C	15128,1647	–6286,7054	–6371,0583
D	E	–1837,7459	–6253,8534	–6596,6697
F	A	–1116,4523	–4596,1610	–4355,9062
F	C	10527,7852	–994,9377	–956,6246
F	E	–6438,1364	–962,0694	–1182,2305
F	D	–4600,3787	5291,7785	5414,4311
F	B	6567,2311	5686,2926	6322,3917
B	F	–6567,2310	–5686,3033	–6322,3807
A	F	1116,4577	4596,1553	4355,9141

Tabela A.2 Coordenadas das estações A e B

Estação	ΔX	ΔY	ΔZ
A	402,35087	-4652995,30109	4349760,77753
B	8086,03178	-4642712,84739	4360439,08326

Solucionando o ajustamento com o código implementado na linguagem Python, contido no Anexo B, obteve-se as seguintes coordenadas ajustadas:

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ X_D \\ Y_D \\ Z_D \\ X_E \\ Y_E \\ Z_E \\ X_F \\ Y_F \\ Z_F \end{bmatrix} = \begin{bmatrix} 12046{,}5813 \\ -4649394{,}08 \\ -4362397{,}94 \\ -3081{,}58324 \\ -4643107{,}37 \\ -4356104{,}68 \\ -4919{,}33534 \\ -4649361{,}22 \\ -4362482{,}23 \\ 1518{,}80079 \\ -4648399{,}15 \\ -4359961{,}18 \end{bmatrix} m$$

Ainda elaborou-se a MVC, utilizando a variância a posteriori para os cálculos, dos parâmetros ajustados:

$$\Sigma_{X_a} = \begin{bmatrix} 6.4894*10^{-6} & 0 & 0 & \cdots & 4.4985*10^{-8} & 0 & 0 \\ 0 & 3.7942*10^{-7} & 0 & \cdots & 0 & 1.8060*10^{-7} & 0 \\ 0 & 0 & 1.7258*10^{-6} & \cdots & 0 & 0 & 1.1770*10^{-6} \\ 8.3054*10^{-7} & 0 & 0 & \cdots & 9.0396*10^{-8} & 0 & 0 \\ 0 & 7.0808*10^{-8} & 0 & \cdots & 0 & 7.1947*10^{-8} & 0 \\ 0 & 0 & 1.6028*10^{-6} & \cdots & 0 & 0 & 1.1608*10^{-6} \\ 6.2481*10^{-6} & 0 & 0 & \cdots & 4.5820*10^{-8} & 0 & 0 \\ 0 & 5.5973*10^{-8} & 0 & \cdots & 0 & 5.4832*10^{-8} & 0 \\ 0 & 0 & 1.6365*10^{-6} & \cdots & 0 & 0 & 1.1573*10^{-6} \\ 4.4985*10^{-8} & 0 & 0 & \cdots & 9.8941*10^{-8} & 0 & 0 \\ 0 & 1.8060*10^{-7} & 0 & \cdots & 0 & 1.9101*10^{-7} & 0 \\ 0 & 0 & 1.1770*10^{-6} & \cdots & 0 & 0 & 1.5492*10^{-6} \end{bmatrix}$$

De acordo com as Eq. 2.5.2.6 e Eq.2.5.2.7, pode-se obter os semineixos os elipsoides de erros referentes às coordenadas dos pontos C, D, E e F:

$$\begin{bmatrix} a_C \\ b_C \\ c_C \end{bmatrix} = \begin{bmatrix} 0{,}00254744 \\ 0{,}00061597 \\ 0{,}00131372 \end{bmatrix} m$$

$$\begin{bmatrix} a_D \\ b_D \\ c_D \end{bmatrix} = \begin{bmatrix} 0{,}00091134 \\ 0{,}00026610 \\ 0{,}00126604 \end{bmatrix} m$$

$$\begin{bmatrix} a_E \\ b_E \\ c_E \end{bmatrix} = \begin{bmatrix} 0{,}00249964 \\ 0{,}00023659 \\ 0{,}00127930 \end{bmatrix} m$$

$$\begin{bmatrix} a_F \\ b_F \\ c_F \end{bmatrix} = \begin{bmatrix} 0{,}00021210 \\ 0{,}00042498 \\ 0{,}00108492 \end{bmatrix} m$$

APÊNDICE B. Código Python implementado para solução de um ajustamento de rede GNSS

```
def calc_ajuste_rede(observation_matrix, std_dev_matrix, ctrl_matrix,
var_a_priori):
    # Variância a priori
    v_a_p = var_a_priori
    # Matriz de Pontos de Controle
    M_ctrl = ctrl_matrix

    # Cria vetor dos pontos de controle
    v_ctrl=[]
    for ctrl in M_ctrl:
        v_ctrl.append(ctrl[0])

    # Matriz de Observações
    M_obs = observation_matrix

    # Lista dos pontos que não são de controle
    v_nctrl=[]
    for observ in M_obs:
        if (observ[0] not in v_nctrl and observ[0] not in v_ctrl):
            v_nctrl.append(observ[0])
        if (observ[1] not in v_nctrl and observ[1] not in v_ctrl):
            v_nctrl.append(observ[1])

    # Matriz dos desvios padrões
    M_dev_pad = std_dev_matrix

    def ajuste(A, P, L):
        N = np.dot(A.T, np.dot(P, A))
        U = np.dot(A.T, np.dot(P, L))
        Ninv = np.linalg.inv(N)
        delta_X = np.dot(Ninv, U)
        return delta_X

    #Aproximação inicial
    X0 = []
    for nctrl in v_nctrl:
        for obs in M_obs:
            if(obs[0] == nctrl and obs[1] in v_ctrl):
                for ctrl in M_ctrl:
                    if(ctrl[0]==obs[1]):

X0.append([obs[0],ctrl[1]+obs[2],ctrl[2]+obs[3],ctrl[3]+obs[4]])
                break
            elif (obs[1] == nctrl and obs[0] in v_ctrl):
                for ctrl in M_ctrl:
                    if(ctrl[0]==obs[0]):

X0.append([obs[1],ctrl[1]+obs[2],ctrl[2]+obs[3],ctrl[3]+obs[4]])
                break

    for ctrl in M_ctrl:
        X0.append([ctrl[0],ctrl[1], ctrl[2], ctrl[3]])

    #Matriz Jacobiana
    A_dev = []
    zero_line = []
```

```
    for x in v_nctrl:
        zero_line.append(0)
        zero_line.append(0)
        zero_line.append(0)
    for x in v_ctrl:
        zero_line.append(0)
        zero_line.append(0)
        zero_line.append(0)

    for obs in M_obs:
        A_dev.append(zero_line.copy())
        A_dev.append(zero_line.copy())
        A_dev.append(zero_line.copy())
        for i in range(len(X0)):
            if(obs[1] == X0[i][0]):
                A_dev[len(A_dev)-3][i*3] = 1
                A_dev[len(A_dev)-2][i*3+1] = 1
                A_dev[len(A_dev)-1][i*3+2] = 1
                break
        for i in range(len(X0)):
            if(obs[0] == X0[i][0]):
                A_dev[len(A_dev)-3][i*3] = -1
                A_dev[len(A_dev)-2][i*3+1] = -1
                A_dev[len(A_dev)-1][i*3+2] = -1
                break
    for ctrl in M_ctrl:
        A_dev.append(zero_line.copy())
        A_dev.append(zero_line.copy())
        A_dev.append(zero_line.copy())
        for i in range(len(X0)):
            if(ctrl[0] == X0[i][0]):
                A_dev[len(A_dev)-3][i*3] = 1
                A_dev[len(A_dev)-2][i*3+1] = 1
                A_dev[len(A_dev)-1][i*3+2] = 1
                break
    A_dev = np.array(A_dev)

    # Vetor de observações
    Lb = []
    for obs in M_obs:
        Lb.append([obs[2]])
        Lb.append([obs[3]])
        Lb.append([obs[4]])
    for ctrl in M_ctrl:
        Lb.append([ctrl[1]])
        Lb.append([ctrl[2]])
        Lb.append([ctrl[3]])

    # Vetor de observações estimadas com base na aproximação inicial
(X0)
    L0 = []
    for obs in M_obs:
        valor = [0,0,0]
        for x in X0:
            if (x[0]==obs[1]):
                valor[0] += x[1]
                valor[1] += x[2]
                valor[2] += x[3]
            if (x[0]==obs[0]):
                valor[0] -= x[1]
                valor[1] -= x[2]
```

```
                valor[2] -= x[3]
        L0.append([valor[0]])
        L0.append([valor[1]])
        L0.append([valor[2]])
    for ctrl in M_ctrl:
        L0.append([ctrl[1]])
        L0.append([ctrl[2]])
        L0.append([ctrl[3]])

    # Vetor L
    #L = L0 - Lb
    L = np.array(L0) - np.array(Lb)

    # Matriz de Varianca e Covariancia
    diag_abs = np.array([])
    for dev_pad in M_dev_pad:
        diag_abs = np.concatenate((diag_abs, np.array([dev_pad[0]]),
np.array([dev_pad[1]]),np.array([dev_pad[2]])))
    for ctrl in M_ctrl:
        diag_abs = np.concatenate((diag_abs, np.array([10**-12]),
np.array([10**-12]),np.array([10**-12])))
    mvc_abs = np.diag(diag_abs ** 2)

    # Matriz Peso
    P_abs = np.linalg.inv(mvc_abs)

    # Aplicação do ajustamento
    delta_X = ajuste(A_dev, P_abs, L)
    #X = X0 - delta_X

    X1 = X0.copy()
    for i in range(len(X0)):
        X0[i]=[X0[i][1],X0[i][2],X0[i][3]]

    X0 = np.array(X0).reshape(3*len(X0),1)
    X = np.array(X0) - np.array(delta_X)

    # Matriz de variancia e covariancia
    N = np.dot(A_dev.T,np.dot(P_abs,A_dev))

    # Retorna matriz de parametros ajustados
    M_par_aj = []
    for i in range(len(X)):
        if i%3 == 0:
            if X1[i//3][0] not in v_ctrl:
                M_par_aj.append([X1[i//3][0], X[i][0], X[i+1][0],
X[i+2][0]])

    # Retorna Matriz das Variancas e Covariancas dos parâmetros
ajustados (para exibicao das elipses)
    M_var = (np.linalg.inv(N))
    sqrt_m_var = (np.linalg.inv(N))**0.5
    M_elipsoides_de_erro = []
    for i in range(len(M_var)):
        if i%3 == 0 and i < len(v_nctrl)*3:
            M_elipsoides_de_erro.append([X1[i//3][0],
sqrt_m_var[i][0], sqrt_m_var[i+1][1], sqrt_m_var[i+2][2]])

    # Retorna a Matriz de Variância de Covariância
    MVC_final = []
    for i in range(len(M_var)):
```

```
        if(i < len(v_nctrl)*3):
            mvc_line = [v_nctrl[i//3]]
            for j in range(len(v_nctrl)):
                mvc_line.append(M_var[i][j])
                mvc_line.append(M_var[i][j+1])
                mvc_line.append(M_var[i][j+2])
            MVC_final.append(mvc_line.copy())

    # Realiza o Teste Qui-Quadrado
    V = np.dot(A_dev,X) - Lb

    quiquadrado_calc =
np.array(np.dot(V.T,np.dot(P_abs,V))/v_a_p)[0][0]

    quitabmax = chi2.ppf(1-0.05/2, len(Lb) - len(X0))
    quitabmin = chi2.ppf(0.05/2, len(Lb) - len(X0))

    teste_hipotese_aceito = False
    if(quiquadrado_calc < quitabmax and quiquadrado_calc > quitabmin):
        teste_hipotese_aceito = True

    return M_par_aj, MVC_final, M_elipsoides_de_erro,
teste_hipotese_aceito
```