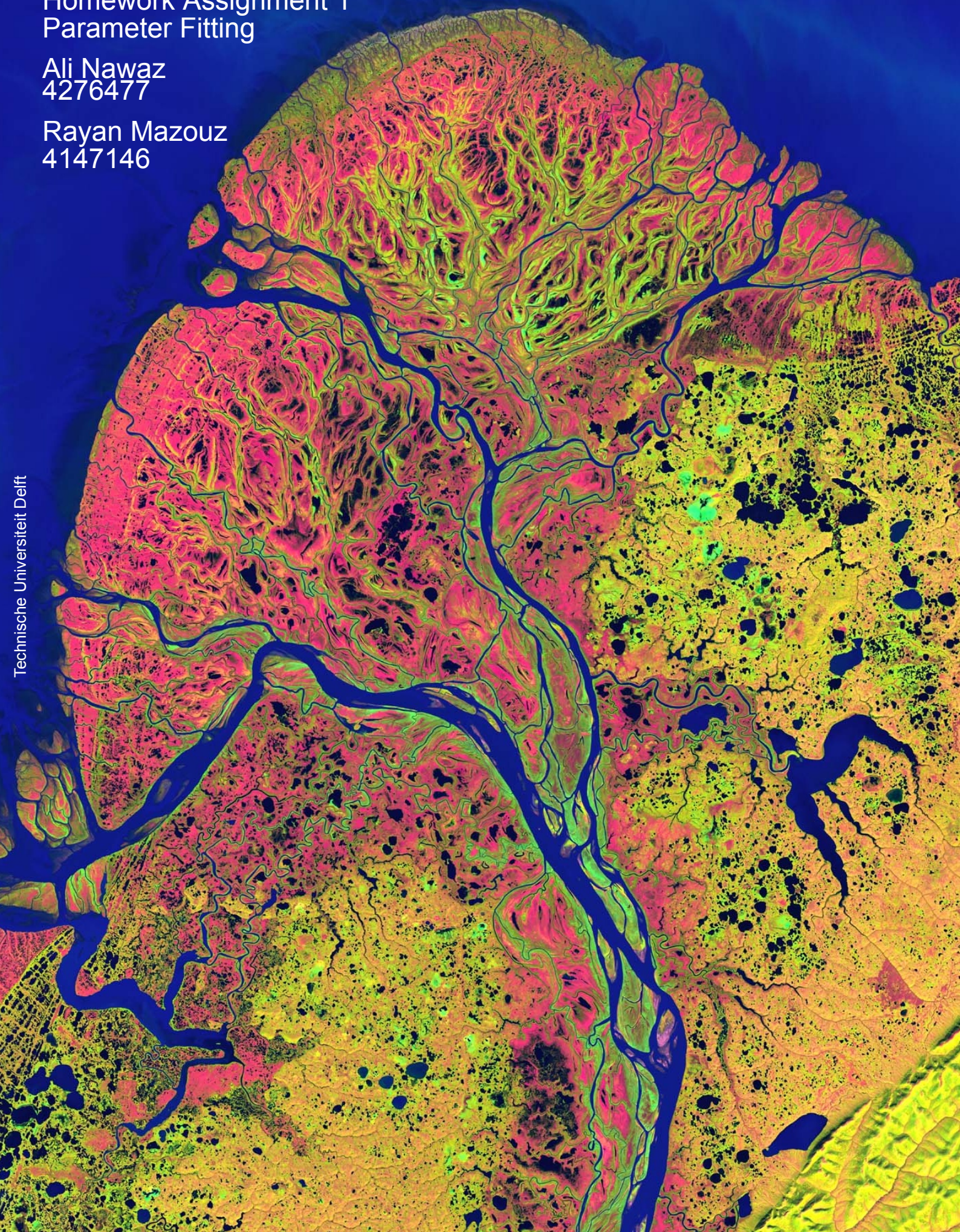


AE4872
Satellite Orbit Determination

Homework Assignment 1
Parameter Fitting

Ali Nawaz
4276477

Rayan Mazouz
4147146



List of symbols

Table 1: List of symbols

Symbol/Acronym	Description	Unit
μ	Freq sample mean	Hz
Λ	Diagonal matrix containing the singular values	-
A	Information/Data Matrix	-
CDF	Cumulative Distributed Frequency	-
f	Frequency	Hz
FCA	Frequency of Closest Approach	Hz
LSQ	Least Squares	-
MSL	Matlab Script Lines	-
n	Number of data points	-
pinv.	Pseudo Inverse	-
R^2	Co-efficient of Determination	-
RMS	Root Mean Square	Hz
RSS	Residual Sum Squared	Hz^2
SSE	Sum Squared Error	Hz^2
SST	Sum of Squared Total	Hz^2
std. (σ)	Standard Deviation	Hz
SVD	Singular Value Decomposition	-
t	Time	s
TCA	Time of Closest Approach	s
var.	Variance	Hz^2
x	State vector	-
y	Observation vector	-

Contents

1	Data Analysis with LSQ	1
1.1	Polynomial fitting with LSQ	1
1.2	Residual analysis	4
1.3	N-order polynomial fitting with LSQ	6
2	Data analysis with F-Test	10
2.1	Optimal n-order with F-Test	10
2.2	Time and frequency of closest approach, TCA and FCA	12
A	Matlab Script	14

Data Analysis with LSQ

The transmitted radio signal of Delfi-C3 measured by DopTrack can be transformed to obtain time [sec], frequency [Hz] and range-rate [m/s]. The goal of this assignment is to generate a polynomial model which fits the range rate measurements. Later the model is used to analyse the frequency and time of closest approach of Delfi-C3. This is compared with the actual observations at the ground station.

To facilitate the fitting of observation with the aid of a model, the least squares approach is utilised. Observation data is obtained from the file *Delfi - C3_32789_201602210946.rre'*. First part of this chapter conducts a low order polynomial fitting through the observed dataset. Afterwards, the residual behaviour is analysed and the model is improved by increasing the order of the polynomial. Since increasing the order of polynomial endlessly is not feasible, both computationally and temporally, statistical analysis is conducted on the residuals to obtain an optimal order of polynomial for the model.

1.1. Polynomial fitting with LSQ

To model the observations least-squares approach is applied on a Chebyshev function of the first kind. Primary function used as the model is represented in Equation 1.1. In this equation $a_0 - a_3$ are the estimated parameters. Furthermore, x is the time of observation and $f_3(x)$ the observed frequency.

$$f_3(x) = a_0 + a_1x + a_2x^3 + a_3x^5 \quad (1.1)$$

Equation 1.2 represents the least-squares problem. The observation vector is denoted as \bar{f} , the information matrix as A and the parameter vector as \bar{x} . The difference between the modelled and observed data is ϵ .

$$\bar{f} = A \cdot \bar{x} + \bar{\epsilon} \quad (1.2)$$

For different epoch the frequency and time observations represented by the model in Equation 1.1 can be expressed in the matrix notation form of Equation 1.2. Equation 1.3 illustrates the Chebyshev model in form of matrix notation. The size of the matrices depend on the number of measurements n . The aim of the least-squares method is to minimize a certain cost function expressed in terms of the residuals known as the Jacobian, $J = \epsilon^T \bar{\epsilon}$. For this assignment it is assumed that all observations are equally weighted.

$$\begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^3 & x_1^5 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & x_n^3 & x_n^5 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} + \bar{\epsilon} \quad (1.3)$$

The rank of data matrix A must be conserved throughout the LSQ estimation as loss in rank indicates loss of valuable information. Loss of rank is common for matrices with high values that undergo inverse or pseudo inverses. The rank of the information matrix A was observed to be 3 so clearly there was a loss of rank. To avoid this the frequency and time observations had to be normalized. There are different ways of normalizing the observations. The method indicated in the lecture notes normalizes observations around the pseudo-mean $\mu = \frac{\max(f) - \min(f)}{2}$. This process, as formulated with the aid of Equation 1.4, scales the frequency and time data in the range [-0.5, 0.5]. However, while conducting later estimations it was observed that this does not result in the optimal fit of the model. This was especially the case around the edges and the reason for that could be due the presence of negative normalizations or the definition of the used mean. To improve the model fitting, normalization around two different points were further analysed. These are indicated by

Equations 1.5 and 1.6. The first represents a normalization around the actual mean (still in the range [-0.5, 0.5]) while the latter represents a normalization around the minimum observation (in the range [0, 1]). The results of all three normalizations are illustrated in Figure 1.1.

$$f_{norm} = \frac{f - \frac{f_{max} - f_{min}}{2}}{f_{max} - f_{min}} \quad t_{norm} = \frac{t - \frac{t_{max} - t_{min}}{2}}{t_{max} - t_{min}} \quad (1.4)$$

$$f_{norm} = \frac{f - f_{mean}}{f_{max} - f_{min}} \quad t_{norm} = \frac{t - t_{mean}}{t_{max} - t_{min}} \quad (1.5)$$

$$f_{norm} = \frac{f - f_{min}}{f_{max} - f_{min}} \quad t_{norm} = \frac{t - t_{min}}{t_{max} - t_{min}} \quad (1.6)$$

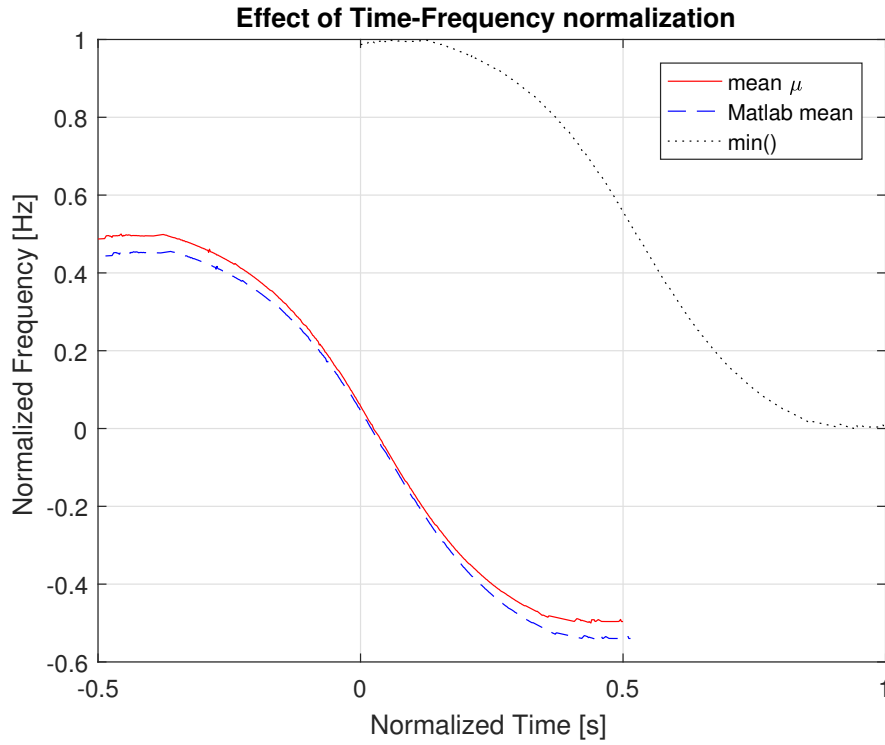


Figure 1.1: Time and frequency normalizations around μ , mean and min. parameter values.

It can be clearly observed from Figure 1.1 that the edge performance while normalizing with the aid of Equation 1.4 far exceeds the edge performance with other 2 normalisation techniques. The effect is relatively small when observed on this normalized scale, however the effect is quite prominent when the observations are de-normalized.

Once the time-freq. observations are normalized, the information matrix is observed to possess a full rank of 4. To aid visualisation of further processes undertaken, a simplified flowchart of the processes is illustrated in Figure 1.2. Interested readers are recommended to check the script provided in Appendix A.

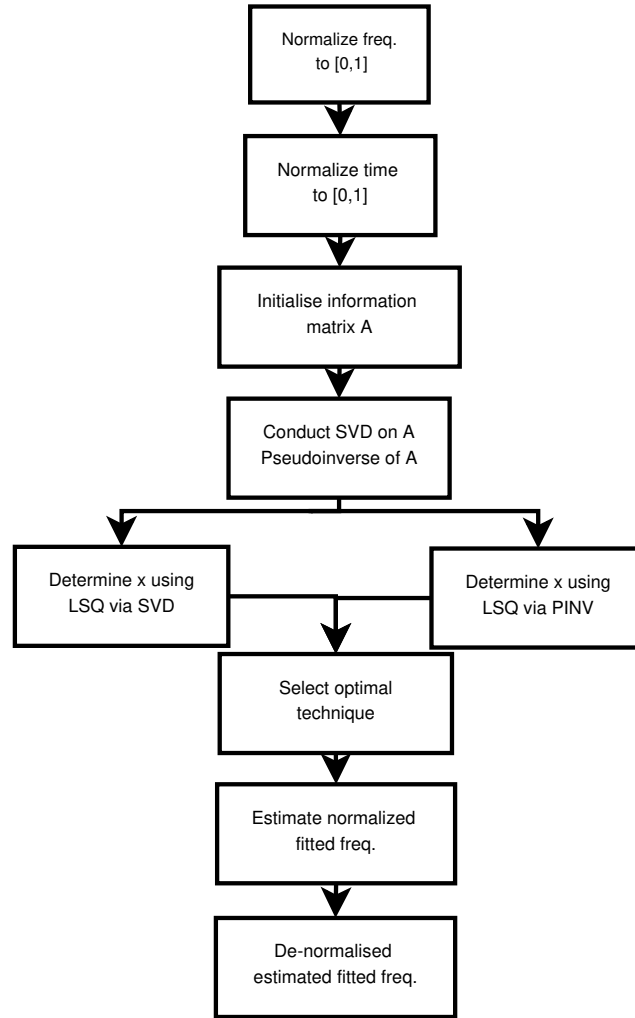


Figure 1.2: Flowchart illustrating steps undertaken to fit observed freq. through a fifth-order polynomial model.

It can be observed from the flowchart that a trade-off is conducted in the LSQ method. Nominally, Equation 1.8 is used to estimate the parameters. Since A is not a square matrix, conducting pseudo-inverse results in singularity errors while evaluating Equation 1.8 numerically in Matlab. Even though the parameters are estimated correctly, Matlab indicates presence of singularity errors. To avoid singularity errors SVD is conducted on the information matrix A . The process of obtaining the LSQ expression in terms of SVD is expressed in Equations 1.9 - 1.12.

$$\bar{y} = A \cdot \hat{x} \quad (1.7)$$

$$\hat{x} = (A^t \cdot A)^{-1} \cdot A^t \cdot \bar{y} \quad (1.8)$$

$$A^t A = V \Lambda^2 U^t U \Lambda V^t = V \Lambda^2 V^t \quad (1.9)$$

$$A^t \bar{y} = V \Lambda U^t \bar{y} \quad (1.10)$$

$$V \Lambda^2 V^t \bar{x} = V \Lambda U^t \bar{y} \rightarrow V^t \bar{x} = U^t \bar{y} \quad (1.11)$$

$$\bar{x} = V \Lambda^{-1} U^t \bar{y} \quad (1.12)$$

In these equations U and V are orthonormal matrices so that $U^t U = I$ and $V^t V = I$. Furthermore, Λ is a diagonal matrix containing the singular values. For the computation, Matlab's `svd` algorithm has been used.

In Table 1.1 the results for both methods are presented. The conclusion is drawn that both methods produce identical results up until eleven digits after the decimal point. However, LSQ method involving SVD operation produces no singularity warnings in Matlab operations.

Table 1.1: Coefficient (a_i) values for pseudo-inverse and singular value decomposition

Coefficients	Pseudo-inverse	Singular value decomposition
a_0	1.039772409710317	1.039772409710330
a_1	-0.322757964905084	-0.322757964905161
a_2	-3.077585132129835	-3.077585132129630
a_3	2.469923989445452	2.469923989445299

Once the parameters are obtained, frequency of fit is estimated with the aid of Equation 1.7. It is important to note that this frequency is still normalised and requires de-normalization using Equation 1.13. Figure 1.3 illustrates the observed frequency against estimated frequency obtained via pseudo inverse and SVD techniques.

$$f = f_{norm} \cdot (f_{max} - f_{min}) + f_{min} \quad (1.13)$$

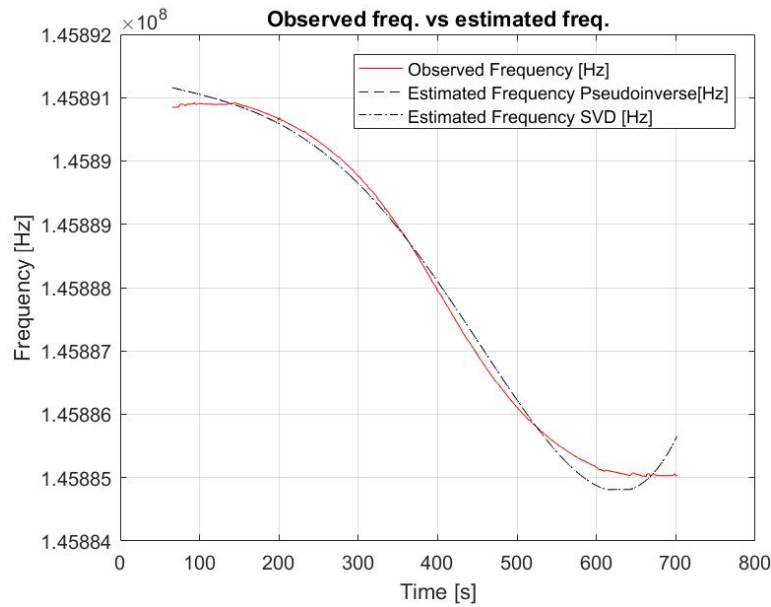


Figure 1.3: Estimated frequency function plotted over the observed frequency.

1.2. Residual analysis

To analyse the accuracy of the estimated frequency model, statistical analysis is conducted on the residuals (difference between observation and estimation). This section focuses on generating residual information and their histogram plots for LSQ estimation via pseudo-inverse and SVD . Furthermore, the mean, median, standard deviation and co-efficient of determination R^2 are analysed.

1.2.1. Residual comparison: data vs estimated function

Residual is the difference between estimated and observed frequency which is obtained using Equation 1.14.

$$residual = observed\ frequency - estimated\ frequency \quad (1.14)$$

Figure 1.4 illustrates residual information for both pseudo-inverse and SVD guided LSQ estimation. It can be observed that the residuals obtained via both techniques are very similar. However, it is still important to evaluate their numerical statistical parameters, which is conducted in the next section. To conclude this section, histogram plots of the residuals for both the techniques used are illustrated in Figure 1.5.

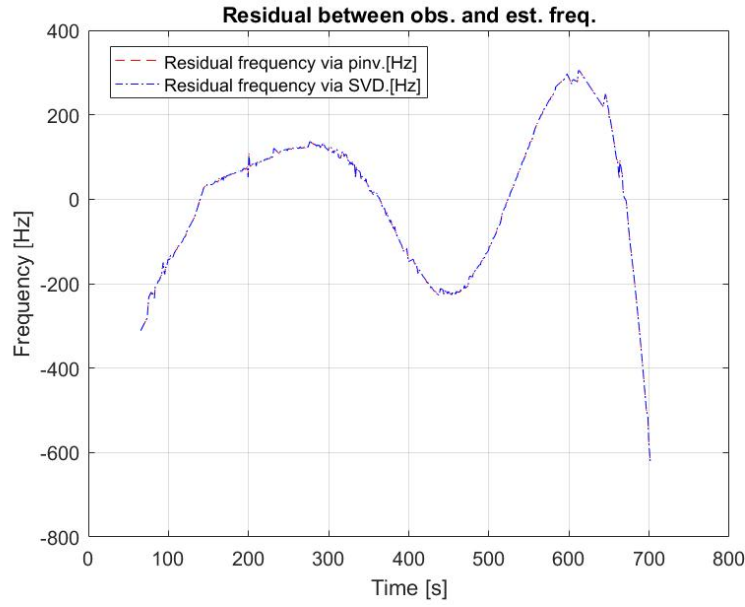


Figure 1.4: Residual analysis for different techniques of LSQ used.

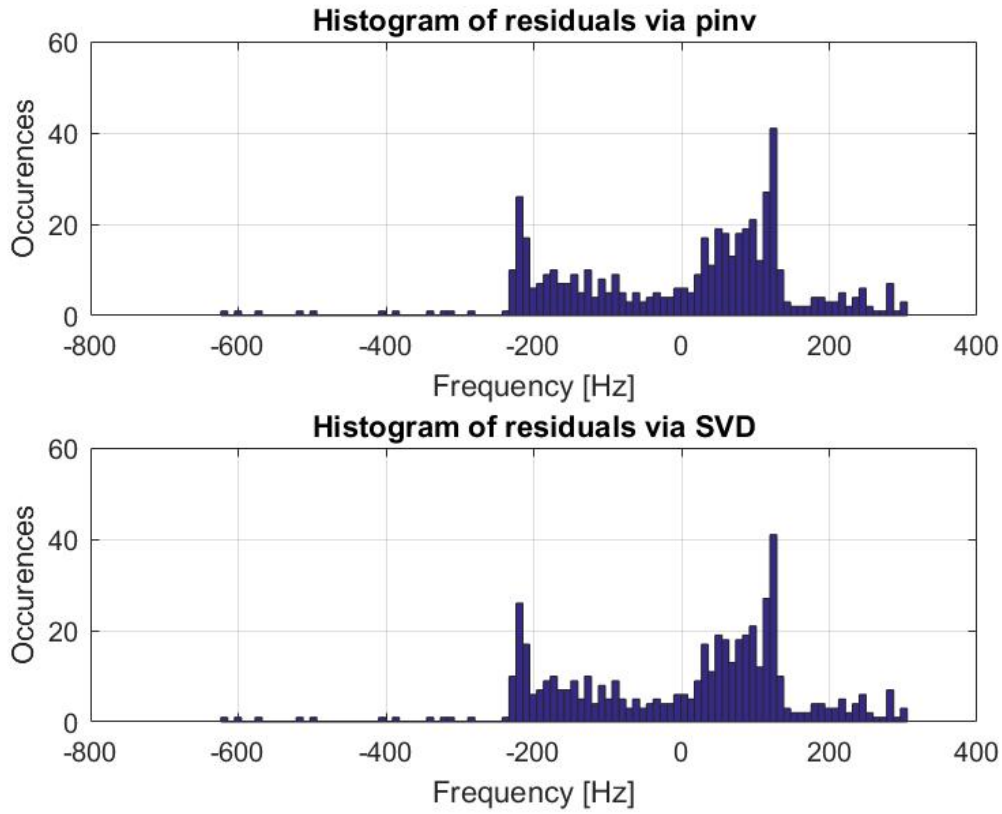


Figure 1.5: Histogram plot for residuals obtained via different LSQ techniques.

1.2.2. Residual statistical parameters

Statistical parameters of the residuals are determined with the aid of Equations 1.15- 1.18. Table 1.2 summarises the statistical parameters obtained for residuals using both LSQ techniques. Again, both methods produce similar results, but the advantage of using SVD lies in the absence of singularity warnings.

$$\text{Mean} = \mu = \frac{\sum \bar{r}}{n} = -1.5823 \cdot 10^{-8} \quad (1.15)$$

Where \bar{r} indicates the residuals and n the no. of data points involved.

$$\text{Standard Deviation} = \sigma = \sqrt{\frac{\sum(\bar{r} - \mu)^2}{n}} \quad (1.16)$$

$$\text{Root Mean Squared} = r_{rms} = \sqrt{\frac{1}{n}(r_1^2 + r_2^2 + \dots + r_n^2)} \quad (1.17)$$

$$R^2 = 1 - \frac{SSE}{SST}$$

Where, SSE = Sum of Square Error and SST = Total sum of squares

$$SSE = \sum (\bar{r})^2 \quad (1.18)$$

$$SST = \sum (freq - \mu_{freq})^2$$

Table 1.2: Statistical parameters for the residuals.

	μ [Hz]	Median [Hz]	σ [Hz]	r_{rms}	R^2
PINV	4.16e-10	47.27	154.52	154.37	0.9949
SVD	4.16e-10	47.27	154.52	154.37	0.9949

1.3. N-order polynomial fitting with LSQ

Upon inspecting the residual plot illustrated in fig. 1.4 it can be observed that there is still some information in the data that is not captured by the model; this could be due to the low order of polynomial used to fit the data through the model. Thereby, this section focuses on the influence of fitting observed data-set through a model of higher order of polynomials. First a script is written to fit data through desired "2·n-1"-order polynomial. Value of n is varied in the range [3,15]. While constructing the script precaution is taken to handle instabilities due to singularities, arising due to rank loses in information matrix. With the aid of this script mean, median and std of the residuals is estimated. Finally co-efficient of determination R^2 is estimated before the section concludes on a discussion for preferred value of n.

1.3.1. Generating n-order polynomial fitting

To increase the order of fitting function, Equation 1.19 is used. Equation illustrated in Equation 1.19 allows the user to manipulate the value of n to increase the order of polynomial model to fit the data. However, caution must be taken to avoid singularity errors and loss of information due to increasing order of polynomial.

$$f_n(x) = a_0 + \sum_{i=1}^n a_i \cdot x^{2 \cdot i - 1} \quad (1.19)$$

To avoid singularity errors and loss of information from the A matrix, observed frequency and time is primarily normalised in the range [0,1]. Figure 1.6 represents a flowchart indicating the sequences of steps undertaken to estimate frequency fitted from the desired order of polynomial. Interested reader is recommended to view MSL presented in appendix A.

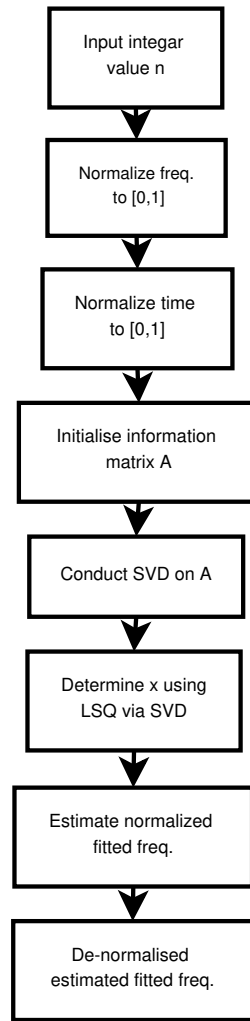


Figure 1.6: Flowchart illustrating steps undertaken to fit observed freq. through a desired polynomial model.

After normalising frequency observation, information matrix A is constructed as indicated in Equation 1.20. Upon construction of A matrix from normalised time observation, the aim is to minimize the Jacobian(cost function) as previously indicated by Section 1.1. After normalization, the information matrix appears to have full rank. Since the information matrix is full rank, theoretically LSQ both with the aid of pinv. and SVD can be used. However, Matlab indicated warnings regarding the possibility of the presence of singularities during parameter estimation with pinv. LSQ. To avoid this, LSQ with SVD is applied, (similar analogy to that of Section 1.1 is applicable here). Estimation of the unknown parameters with the aid of SVD guided LSQ, can be conducted with the aid of previously mentioned Equations 1.9-1.12.

$$\begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & \dots & x_1^{2n-1} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & \dots & x_n^{2n-1} \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} + \epsilon \quad (1.20)$$

Once the parameter is estimated, normalized fitted frequency is obtained with the aid of Equation 1.6. This normalised fitted frequency is now set back to original values by de-normalising the results with the aid of Equation 1.13.

1.3.2. N-order residual statistical parameters

Using the algorithm presented in Section 1.3.1, frequency is estimated for n in the range [3,15]. These estimated frequencies are compared with the observed frequencies to obtain residuals. Table 1.3 summarises statistical properties of the residuals for varying n . Statistical properties include value of n and its corresponding polynomial order, mean, median, standard deviation and co-efficient of determination R^2 . Based on the data presented in Table 1.3 it can be observed that for $n=11$ (i.e. polynomial of order 21) onward the fit reaches a fixed standard deviation and R^2 (to 5 decimal places). Based on this $n=11$ is preferred. Of course increasing n further decreases the mean and brings it closer to 0, however the corresponding increase in the required

computational work doesn't add extra value in terms of Std. and R^2 . Next section focuses on obtaining the optimal value of n by conducting an F-test on the residuals. Figure 1.7 and 1.8 illustrate the estimated and observed frequency for $n=3$ and $n=11$. While Figure 1.9 and 1.10 illustrate their respective residuals.

Table 1.3: Residual statistical parameters for varying n .

n	Polynomial order	Mean	Median	Std.	R^2
3	5	4.16e-10	4.16e-10	154.52	0.99485
4	7	1.19e-10	1.19e-10	36.11	0.99972
5	9	3.57e-10	3.57e-10	35.84	0.99972
6	11	-5.95e-11	-5.95e-11	22.41	0.99989
7	13	2.97e-10	2.97e-10	20.68	0.99991
8	15	2.02e-09	2.02e-09	16.94	0.99994
9	17	5.12e-09	5.12e-09	9.48	0.99998
10	19	-7.45e-08	-7.45e-08	8.53	0.99998
11	21	-4.34e-08	-4.34e-08	8.31	0.99999
12	23	5.71e-07	5.71e-07	8.21	0.99999
13	25	-8.12e-07	-8.12e-07	8.12	0.99999
14	27	-2.11e-05	-2.11e-05	8.12	0.99999
15	29	1.24e-04	1.24e-04	8.12	0.99999

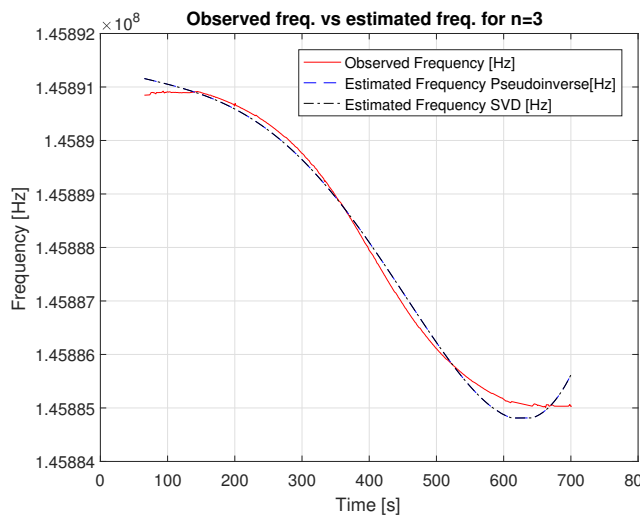


Figure 1.7: Estimated frequency and observed frequency for $n=3$.

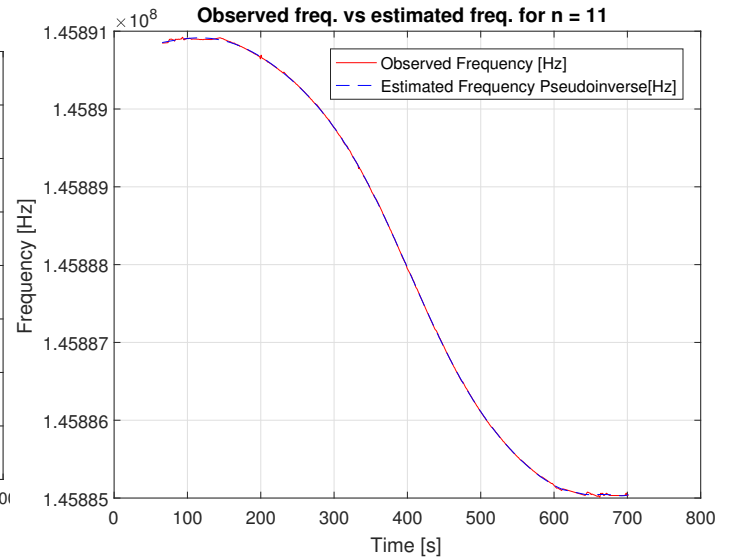


Figure 1.8: Estimated frequency and observed frequency for $n = 11$.

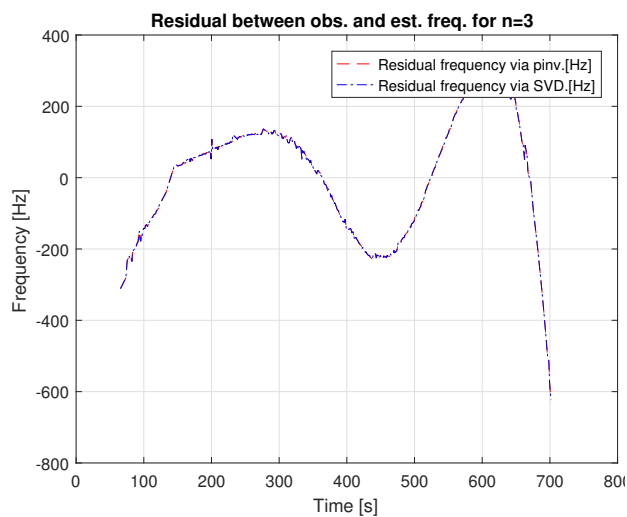


Figure 1.9: Residual between estimated and observed freq. for $n=3$.

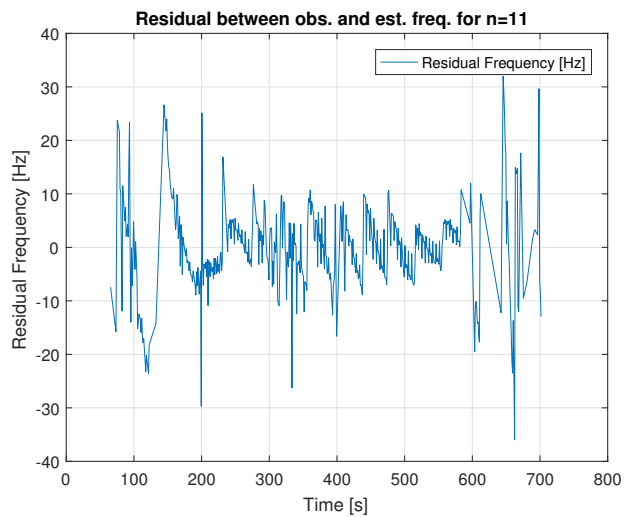


Figure 1.10: Residual between estimated and observed freq. for $n=11$.

By observing the residuals, it is clear that polyfit for $n=13$ functions way better than the polyfit for $n=3$. Thus, optimal n is chosen to be 11.

Data analysis with F-Test

Statistical data presented in Section 1.3.2 alone does not provide insight into the optimal value of n . To provide further insight into the significance of proceeding with further polynomials, F-test is conducted. The aim of this chapter is to conduct F-test on residual dataset and obtain the optimum value of n . Based on this optimum n value, frequency is fitted to the observed dataset. Fitted frequency is used to obtain the Time of Closest Approach (TCA) and Frequency of Closest Approach (FCA) of the satellite.

2.1. Optimal n -order with F-Test

To provide further insight into the optimal value of n , an F-test is conducted. Underlying principle of an F-test is to hypothesise that a data set in a regression analysis follows the simpler of two proposed linear models that are nested within each other. This is similar to the "*varTest2(x,y)*" function in Matlab which returns a test decision for the null hypothesis that the data in vectors x and y comes from normal distribution with the same variance, using two-sample F-test. The alternative hypothesis is that they come from normal distributions with different variances. If the result of hypothesis testing is 1, then the test rejects null hypothesis at 5% significance level, and 0 otherwise. Primarily F-test is conducted with the aid of Equation 2.1. Where RSS is the residual sum square of the residual data set for n -th value, p is the order of polynomial and n is the no. of data points used. To verify the outcomes of F-test, *varTest2* from Matlab is used.

$$F = \frac{\frac{RSS_1 - RSS_2}{p_2 - p_1}}{\frac{RSS_2}{n - p_2}} \quad (2.1)$$

F-test results in a value which if below a certain threshold (F_{crit} to be discussed later on) then it can be assumed that the newer model does not provide any more information or better fit than model 1. F_{crit} indicates that model 2 is better than model 1. It is indicated to use $p < 0.05$ as the selection criteria. Which can be obtained by plotting a cumulative frequency distribution for the F-test. Figure 2.1 illustrates cumulative frequency distribution of the F-test. A line at $1 - p = 0.95$ is marked to provide the threshold value, F_{crit} . A value of 3.86 is obtained for F_{crit} at this intersection point. Table 2.1 summarises the values of F-test for varying values of n . While, Figure 2.2 illustrates a zoomed in view of the F-test. From this it can be observed that for $n = 4, 5$, the value of F-test for the first time reaches below 3.86. At $n = 4, 5$ F-test results in a value of 3.734. As mentioned earlier, if the value of F-test results in a value below F_{crit} , then newer model does not provide any more information or better fit than needed. Thus the optimal value of n is 4. This results in a polynomial of order 7.

Table 2.1: F-test for n in the range [2, 15]

<i>n</i>	<i>Polynomial order</i>	<i>F-test</i>
1, 2	1, 3	1.26e+04
2, 3	3, 5	2.06e+03
3, 4	5, 7	4.28e+03
4, 5	7, 9	3.73
5, 6	9, 11	381.34
6, 7	11, 13	42.63
7, 8	13, 15	119.12
8, 9	15, 17	530.22
9, 10	17, 19	57.01
10, 11	19, 21	12.98
11, 12	21, 23	5.72
12, 13	23, 25	5.29
13, 14	25, 27	0.02
14, 15	27, 29	4.28e-04

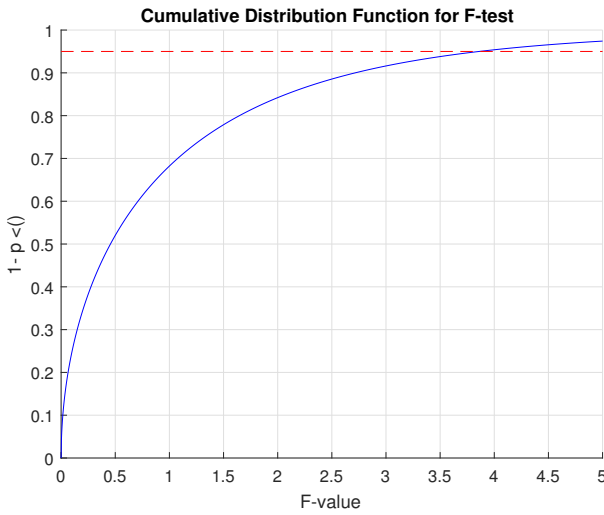
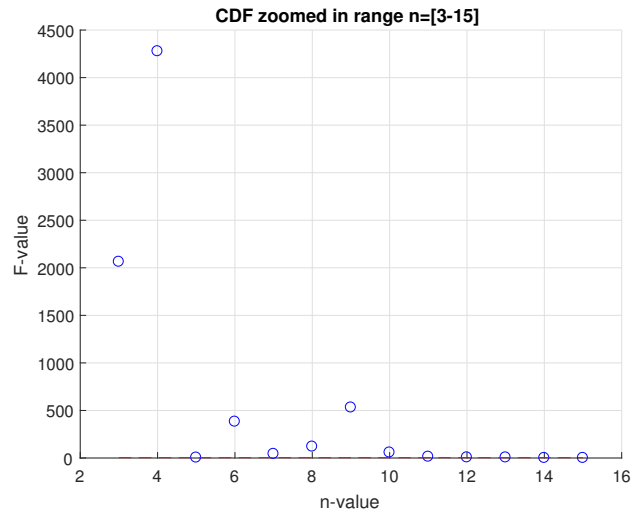
Figure 2.1: CDF to evaluate F_{crit} .

Figure 2.2: F-test with the aid of CDF technique, in the range [3, 15].

To verify the results obtained via F-test, *vartest2* is implemented with *Alpha* parameter set to 0.05. As mentioned earlier, the fundamental basis of the *vartest2* function lies in hypothesis testing. The function relies on testing two hypothesis. First hypothesis is the null hypothesis, hypothesis that two vectors of data are from normal distribution with similar variance. Second hypothesis is the alternate hypothesis, hypothesis that two vectors of data are from normal distribution with different variance. The function conducts a test between these two hypothesis, if the result of the test is 1, then test rejects null hypothesis (hypothesis that data is from normal distribution with similar variances). From Table 2.2 it can be observed that, while testing the residuals of fit obtained for $n=4$ and $n=5$ the null hypothesis is accepted for the first time in increasing values of n . This means that residual data for $n=4$ and $n=5$ have normal distribution with similar variances, i.e. model 5 adds no further value (at least in terms of variance) when compared to model 4. It can thus be verified that optimum value of n is 4, with corresponding polynomial order of 7.

Table 2.2: F-test with the aid of Matlab function "vartest2".

n_1, n_2	Polynomial order for n_1, n_2	Hypothesis value $H=0$ if null hyp. can't be rejected	Prob. of observing the result if null hyp. is true
1, 2	1, 3	1	6.16e-259
2, 3	3, 5	1	5.01e-116
3, 4	5, 7	1	7.63e-179
4, 5	7, 9	0	0.87
5, 6	9, 11	1	6.65e-25
6, 7	11, 13	0	0.07
7, 8	13, 15	1	8.87e-06
8, 9	15, 17	1	1.48e-36
9, 10	17, 19	1	0.018
10, 11	19, 21	0	0.56
11, 12	21, 23	0	0.79
12, 13	23, 25	0	0.81
13, 14	25, 27	0	1.0
14, 15	27, 29	0	1.0

Figure 2.3 illustrates a visual comparison between F-test with the aid of Equation 2.1 and matlab function vartest2. From the first plot it is clear that for $n = 4$, $h=0$ for the first time. Similarly, from the second plot it can be observed that for the first time at $n=5$ (i.e. comparing $n=4$ and $n=5$) the value drops below F_{crit}

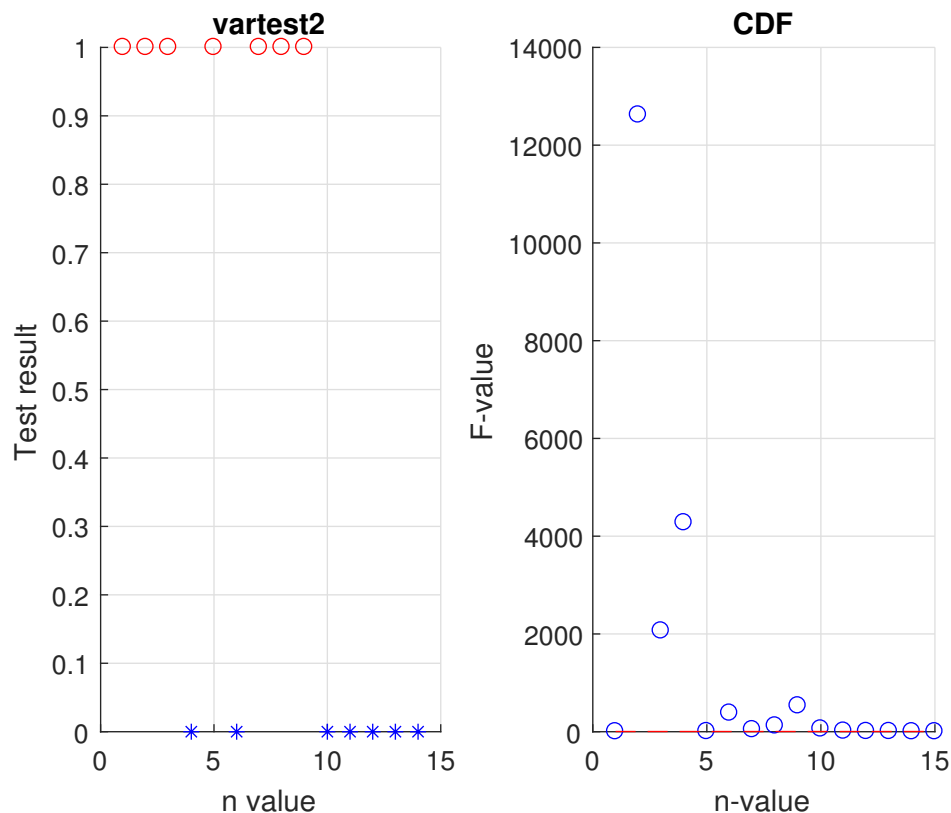


Figure 2.3: Comparison of F-test with the aid of vartest2 and cumulative freq. distribution.

2.2. Time and frequency of closest approach, TCA and FCA

This section focuses on obtaining TCA and FCA using the optimal value of $n=4$ obtained in Section 2.1. Upon retrieving data from instrument it is necessary to avoid differentiating the data, since differentiating discrete time data leads to an increased noise. However, for this assignment fingers are crossed and data is differentiated, but with backward scheme. Based on Doppler effect the rate of change of frequency observation per unit time

reaches a point of inflection as the satellite passes over the ground station. First fitted frequency is estimated for $n=4$ using similar processes mentioned earlier in this assignment. Later $\frac{dfreq}{dt}$ is estimated using Equation 2.4. Using this FCA of 145.89 MHz was obtained, corresponding TCA=408.5.

$$\frac{dfreq}{dt} = \frac{freq_i - freq_{i-1}}{time_i - time_{i-1}} \quad (2.2)$$

This is verified visually by plotting $\frac{dfreq}{dt}$ against time as illustrated in Figure 2.4. At time 408.5s, min. point is obtained for $\frac{dfreq}{dt}$.

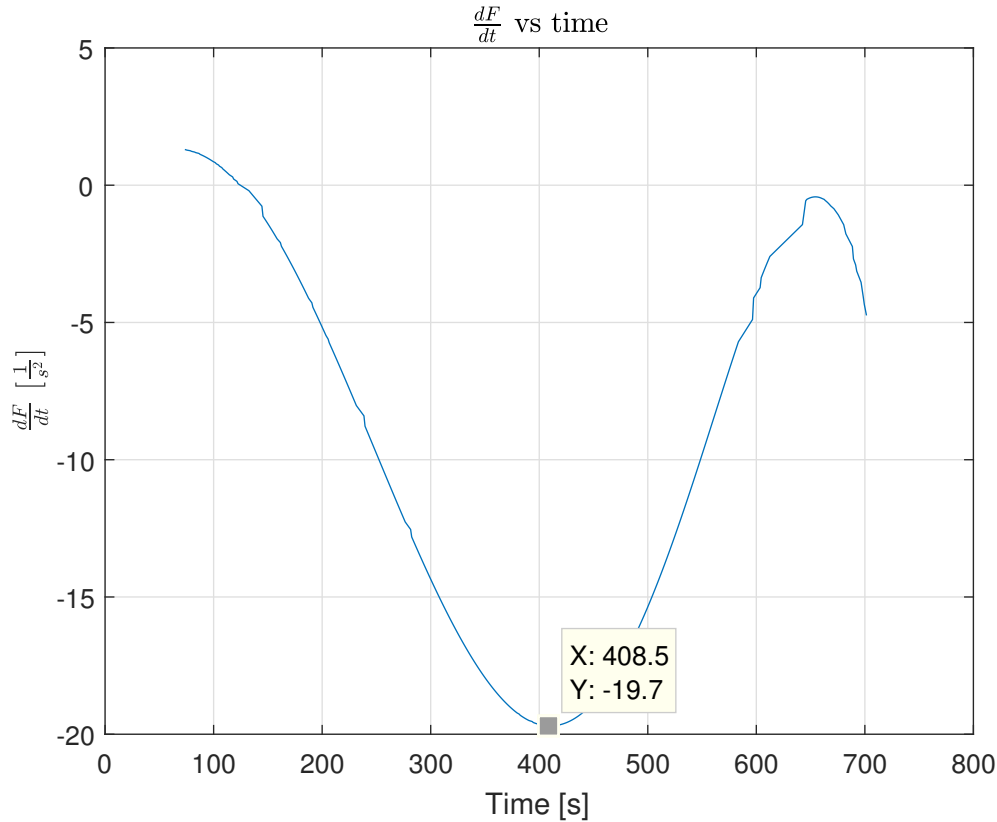
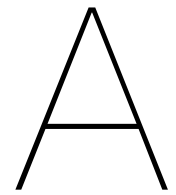


Figure 2.4: Visual guide to TCA by evaluation of rate of change of frequency against time.

However, header of datafile states that TCA=400.978258s and carrier_frequency=145887929.461154 Hz. Whereas for $n=4$, TCA value obtained from estimated frequency is 408.5s while FCA= 145887797.1344178Hz. The percentage offset and uncertainty in estimated against observed TCA/FCA is presented in Table 2.3.

Table 2.3: TCA and FCA for $n=4$ for estimated freq. compared against observed freq.

	TCA [s]	FCA [Hz]
Observed	400.978258	145887929.4611540
Estimated	408.5000	145887797.1344178
Offset	7.5217	132.3267
%Offset of Observed	1.8758	9.0704e-05



Matlab Script

Following script can be also be found at this Github Repository:

<https://github.com/Alixir/Satellite-Orbit-Determination>

```
1 % AE4872: Satellite Orbit Determination
2 % Homework Assignment 1: Parameter Fitting
3 % Authors : Ali Nawaz, Student number : 4276477
4 %          Rayan Mazouz, Student number: 4147146
5 % Facult of Aerospace Engineering, Delft University of Technology.
6
7 close all , clear all , clc;
8
9 file = importdata('Delfi-C3_32789_201602210946.rre '); % Loading data from the file
10 time = file.data(:,1); % time since recording initiated [s]
11 freq = file.data(:,2); % observed frequency [Hz]
12 range_rate = file.data(:,3); % range rate [m/s]
13
14
15 %% Part 1A
16
17 A = zeros(length(time),4); %initializing information matrix A
18 for a = 1:length(time);
19     A(a,1) = 1; % Column 1 elements of information matrix A
20     A(a,2) = time(a); % Column 2 elements of information matrix A
21     A(a,3) = time(a)^3; % Column 3 elements of information matrix A
22     A(a,4) = time(a)^5; % Column 4 elements of information matrix A
23 end
24
25 % Checking the rank of A matrix.
26 rank_A = rank(A); % Rank value of 3, A matrix loses rank.
27
28 % To prevent this rank loss we'll normalise freq and time; after LSQ
29 % they'll be restored to original states.
30
31 % Different normalisation technique tried and the best performing one is
32 % selected
33
34 % freq_norm = ( freq - (max(freq)+min(freq))/2 ) / ( max(freq) - min(freq)); % Normalised freq. i.e
35 % |freq_i| < 1
36 % time_norm = ( time - (max(time) +min(time))/2 )/( max(time) - min(time)); % Normalised time i.e
37 % |time_i| < 1
38
39 %
40 % freq_norm = ( freq - mean(freq) ) / ( max(freq) - min(freq)); % Normalised freq. i.e. |freq_i| < 1
41 % time_norm = ( time - mean(time))/ ( max(time) - min(time)); % Normalised time i.e. |time_i| < 1
42
43 % Comparing different normalizations
44 freq_norm1 = ( freq - (max(freq)+min(freq))/2 ) / ( max(freq) - min(freq)); % Normalised freq. i.e.
45 % |freq_i| < 1
46 time_norm1 = ( time - (max(time) +min(time))/2 )/( max(time) - min(time)); % Normalised time i.e.
47 % |time_i| < 1
48
49 freq_norm2 = ( freq - mean(freq) ) / ( max(freq) - min(freq)); % Normalised freq. i.e. |freq_i| < 1
50 time_norm2 = ( time - mean(time))/ ( max(time) - min(time)); % Normalised time i.e. |time_i| < 1
```



```

49
50 freq_norm3 = ( freq - min(freq) ) / ( max(freq) - min(freq)); % Normalised freq. i.e. |freq_i| < 1
51 time_norm3 = ( time - min(time)) / ( max(time) - min(time)); % Normalised time i.e. |time_i| < 1
52
53 % Plot for 3 different frequency time normalizations
54
55 figure(19)
56 plot(time_norm1, freq_norm1, 'r-', time_norm2, freq_norm2, 'b--', time_norm3, freq_norm3, 'k:');
57 title('Effect of Time-Frequency normalization');
58 legend({'mean $\mu$', 'Matlab mean', 'min()' }, 'Interpreter', 'latex');
59 xlabel('Normalized Time [s]');
60 ylabel('Normalized Frequency [Hz]');
61 grid on
62
63 % Reconstructing A matrix from the normalised dataset
64 A_norm = zeros(length(time),4); %initializing normalised information matrix A
65 for a = 1:length(time_norm);
66     A_norm(a,1) = 1; % Column 1 elements of information matrix A
67     A_norm(a,2) = time_norm(a); % Column 2 elements of information matrix A
68     A_norm(a,3) = time_norm(a)^3; % Column 3 elements of information matrix A
69     A_norm(a,4) = time_norm(a)^5; % Column 4 elements of information matrix A
70 end
71
72 % Rank check for A matrix
73
74 rank_A_norm = rank(A_norm); % Rank value of 4, no rank loss
75
76 % Since there is no rank loss, LSQ with psuedo-inverse can be applied
77
78 x1 = ((transpose(A_norm)*A_norm))\transpose(A_norm)*freq_norm; % Parameter estimation via LSQ with
    pinv.
79
80 a0_1 = x1(1); % a0
81 a1_1 = x1(2); % a1
82 a2_1 = x1(3); % a2
83 a3_1 = x1(4); % a3
84
85 freq_fit_norm1 = A_norm*x1; % Normalised plot-fitted freq plot
86 % freq_fit1 = freq_fit_norm1* ( max(freq) - min(freq) ) + mean(freq); % De-normalising fitted
    frequency
87 % freq_fit1 = freq_fit_norm1* ( max(freq) - min(freq) ) + (max(freq) + min(freq))/2; % De-
    normalising fitted frequency
88 freq_fit1 = freq_fit_norm1* ( max(freq) - min(freq) ) + min(freq); % De-normalising fitted
    frequency
89
90 %Application of SVD for verification of LSQ with pseudo-inverse
91
92 [U,S,V] = svd(A_norm,'econ'); % Singular value decomposition of A, to avoid singularity errors or
    solution manifold due to rank deficit
93 condition_number = S(1,1)/S(4,4); %2.5508e+14 We could lose upto 15 digits in any numerical
    calculations when A is computed directly.
94 S_inv_diag = [inv(S(1,1)), inv(S(2,2)), inv(S(3,3)), inv(S(4,4))]; % Diagonal elements of inv S
95 S_inv = diag(S_inv_diag);
96 x2 = V*S_inv*U'*freq_norm; % Parameter estimation via SVD aided LSQ
97 a0_2 = x2(1); % a0
98 a1_2 = x2(2); % a1
99 a2_2 = x2(3); % a2
100 a3_2 = x2(4); % a3
101 freq_fit_norm2 = A_norm*x2; % Normalised plot-fitted freq plot
102 % freq_fit2 = freq_fit_norm2* ( max(freq) - min(freq) ) + mean(freq);
103 % freq_fit2 = freq_fit_norm2* ( max(freq) - min(freq) ) + (max(freq) + min(freq))/2;
104 freq_fit2 = freq_fit_norm2* ( max(freq) - min(freq) ) + min(freq); % De-normalised esitimated
    frequency
105
106
107 figure(11)
108 plot(time, freq, 'r-', time, freq_fit1, 'b--', time, freq_fit2, 'k-.');
109 legend('Observed Frequency [Hz]', 'Estimated Frequency Pseudoinverse[Hz]', 'Estimated Frequency SVD [
    Hz]');
110 title('Observed freq. vs estimated freq. ');
111 xlabel('Time [s]');
112 ylabel('Frequency [Hz]');
113 grid on
114 %% Part 1B
115 % Residual plot between observed data and estimated function

```

```

116 res_pinv = freq - freq_fit1; %Residual for LSQ with pinv
117 res_svd = freq - freq_fit2; %Residual for LSQ with SVD
118
119 figure(12)
120 plot(time, res_pinv, 'r--', time, res_svd, 'b-.');
121 title('Residual between obs. and est. freq. ');
122 legend('Residual frequency via pinv.[Hz]', 'Residual frequency via SVD.[Hz]');
123 xlabel('Time [s]');
124 ylabel('Frequency [Hz]');
125 grid on
126
127 % Histogram plot of residuals
128 figure(13)
129 subplot(2,1,1);
130 hist(res_pinv,100);
131 title('Histogram of residuals via pinv');
132 xlabel('Frequency [Hz]');
133 ylabel('Occurences');
134 grid on
135 subplot(2,1,2);
136 hist(res_svd,100);
137 title('Histogram of residuals via SVD');
138 xlabel('Frequency [Hz]');
139 ylabel('Occurences');
140 grid on
141
142 % Residual information for LSQ with pinv. case
143 res_mean_p = mean(res_pinv); % mean of residuals
144 res_med_p = median(res_pinv); % median of residuals
145 res_std_p = std(res_pinv); % standard deviation of residuals
146 res_rms_p = rms(res_pinv); % RMS of residuals
147 SSE_p = sum( res_pinv.^2); % Sum squared error performance function
148 SST = sum( ( mean(freq)-freq).^2); % total sum of squares
149 res_r2_p = 1-SSE_p/SST; % Co-efficient of determination R^2 of residuals
150 % Residual information for LSQ with SVD case
151 res_mean_s = mean(res_svd); % mean of residuals
152 res_med_s = median(res_svd); % median of residuals
153 res_std_s = std(res_svd); % standard deviation of residuals
154 res_rms_s = rms(res_svd); % RMS of residuals
155 SSE_s = sum( res_svd.^2); % Sum squared error performance function
156 SST = sum( ( mean(freq)-freq).^2); % total sum of squares
157 res_r2_s = 1-SSE_s/SST; % Co-efficient of determination R^2 of residuals
158 %% Part 1C
159 prompt = ('Please enter the an integar value of n: ');
160 n = input(prompt);
161
162 An_norm = zeros(length(time_norm), n+1); % initialising normalised A matrix for nth order
    polynomial
163
164 An_norm(:,1) = 1;
165
166 for a = 1:length(time_norm) % Defining normalised A matrix for nth order polynomial
167     for b = 1:n
168         An_norm(a,b+1) = time_norm(a)^(2*b-1);
169     end
170 end
171 [Un,Sn,Vn] = svd(An_norm,'econ'); % Singular value decomposition of A, to avoid singularity errors
    or solution manifold due to rank deficit
172 condition_numbern = Sn(1,1)/Sn(end,end); % Indicates the number of potential digit that can be lost
    in any numerical calculations when A is computed directly.
173 Sn_inv_diag = [];
174 for nn=1:length(Sn)
175     Sn_inv_diag = [Sn_inv_diag, inv(Sn(nn,nn))];
176 end
177 Sn_inv = diag(Sn_inv_diag);
178 xn = Vn*Sn_inv*Un'*freq_norm; % Parameter estimation via SVD aided LSQ
179
180 % xn = ((transpose(An_norm)*An_norm))\transpose(An_norm)*freq_norm; % Parameter estimation via LSQ
    with pinv.
181
182 freq_fit_norm_n = An_norm*xn; % Normalised plot-fitted freq plot
183 % freq_fitn = freq_fit_norm_n* ( max(freq) - min(freq) ) + mean(freq); % De-normalising fitted
    frequency
184 % freq_fitn = freq_fit_norm_n* ( max(freq) - min(freq) ) + ( max(freq) + min(freq))/2; % De-
    normalising fitted frequency

```

```

185 freq_fitn = freq_fit_norm_n* ( max(freq) - min(freq) ) + min(freq); % De-normalising fitted
    frequency
186
187
188 figure(14)
189 plot(time, freq, 'r-', time, freq_fitn, 'b--');
190 legend('Observed Frequency [Hz]', 'Estimated Frequency Pseudoinverse[Hz]');
191 title(['Observed freq. vs estimated freq. for n = ' num2str(n)]);
192 xlabel('Time [s]');
193 ylabel('Frequency [Hz]')
194 grid on
195 % Residual information
196
197 resn = freq - freq_fitn; % Residual between estimated and observed frequencies.
198
199 % Residual plot for n
200 figure(20)
201 plot(time, resn);
202 legend('Residual Frequency [Hz]');
203 xlabel('Time [s]');
204 ylabel('Residual Frequency [Hz]');
205 title(['Residual between obs. and est. freq. for n=' num2str(n)]);
206 grid on
207
208 resn_mean = mean(resn); % Residual mean
209 resn_med = median(resn); % Residual median
210 resn_std = std(resn); % Residual standard deviation
211 resn_rms = rms(resn); % RMS of residuals
212 SSEn = sum( resn.^2); % Sum squared error performance function
213 SST = sum( ( mean(freq)-freq).^2); % total sum of squares
214 res_r2n = 1-SSEn/SST; % Co-efficient of determination R^2 of residuals
215
216 %% Part 1D
217 prompt = ( 'Please enter the max integar value of n for F-test: ');
218 n = input(prompt);
219
220 rss_n = [];
221 pol_n = [];
222 res_fl = [];
223 for aa = 1:n
224     Af = zeros(length(time_norm), aa+1); % initialising normalised A matrix for nth order
        polynomial
225     Af_norm(:,1) = 1;
226
227     for a = 1:length(time_norm) % Defining normalised A matrix for nth order polynomial
228         for b = 1:aa
229             Af_norm(a,b+1) = time_norm(a)^(2*b-1);
230         end
231     end
232     [Uf,Sf,Vf] = svd(Af_norm, 'econ'); % Singular value decomposition of A, to avoid singularity
        errors or solution manifold due to rank deficit
233     condition_numbern = Sf(1,1)/Sf(end,end); % Indicates the number of potential digit that can be
        lost in any numerical calculations when A is computed directly.
234     Sf_inv_diag = [];
235     for ff=1:length(Sf)
236         Sf_inv_diag = [Sf_inv_diag, inv(Sf(ff, ff))];
237     end
238     Sf_inv = diag(Sf_inv_diag);
239     xf = Vf*Sf_inv*Uf'*freq_norm; % Parameter estimation via SVD aided LSQ
240 %     xf = ((transpose(Af_norm)*Af_norm)\transpose(Af_norm)*freq_norm; % Parameter estimation via
        LSQ with pinv.
241
242     freq_fit_norm_f = Af_norm*xf; % Normalised plot-fitted freq plot
243     % freq_fitf = freq_fit_norm_f* ( max(freq) - min(freq) ) + mean(freq); % De-normalising fitted
        frequency
244     % freq_fitf = freq_fit_norm_f* ( max(freq) - min(freq) ) + ( max(freq) + min(freq))/2; % De-
        normalising fitted frequency
245     freq_fitf = freq_fit_norm_f* ( max(freq) - min(freq) ) + min(freq); % De-normalising fitted
        frequency
246     res_f = freq - freq_fitf; % Residual information
247     res_fl = [res_fl, res_f]; % Appending columns of residual per n-th order polyfit
248     rss = sum(res_f.^2); % Residual Sum Square
249     rss_n = [rss_n; rss]; % Appending residual sum square for nth residual
250     pol_n = [pol_n; 2*aa - 1]; % Appending the current order of polynomial
251 end

```

```

252 F_test = [];
253 for l = 2:length(rss_n)
254     F_test(l,1) = ( ( rss_n(l-1)-rss_n(l) )/(pol_n(l) - pol_n(l-1)) )/ ( rss_n(l)/ (length(
        time_norm) - pol_n(l))) );
255 end
256
257 % Plotting a CDF for F-Test
258 xx = 0:0.00001:5;
259 z = fcdf(xx,1,500);
260 for i = 1:length(z)
261     Fcrit = xx(i);
262     if z(i)>0.95
263         break
264     end
265 end
266
267 figure(15)
268 hold on;
269 plot(xx,z,'b-');
270 plot(xx, 0.95*ones(size(xx)), 'r--');
271 hold off
272 title('Cumulative Distribution Function for F-test');
273 xlabel('F-value');
274 ylabel('1- p <()');
275 grid on
276 % VALIDATION of F_test via vartest2 function
277
278 % Vartest2(x,y) returns a test decision for the null hypothesis that the data in vectors x and y
    comes from normal distributions with the same variance, using the two-sample F-test.
279 %The alternative hypothesis is that they come from normal distributions with different variances.
    The result h is 1 if the test rejects the null hypothesis at the 5% significance level, and 0
    otherwise.
280
281 h_res = []; % Hypothesis value
282 p_res = []; % Probability of Test
283 ci_res = []; % Confidence interval for the true variance ratio
284 stats_res = []; % Structure containing info. about test statistic
285
286 size_resfl = size(res_fl); % Size of appended residuals
287 col_resfl = size_resfl(2); % Total set of residual samples
288 for hh = 2:col_resfl
289     [h,p,ci,stats] = vartest2(res_fl(:,hh),res_fl(:,hh-1),'Alpha',0.05);
290     h_res=[h_res;h];
291     p_res=[p_res;p];
292     ci_res = [ci_res;ci];
293     stats_res = [stats_res;stats];
294 end
295 % f_val = [stats_res.fstat]';
296
297 % plotting probabilities to check which n satisfies p<0.5
298 figure(16)
299 title('F-test results');
300 subplot(1,2,1)
301 hold on
302 % plot(1:length(h_res), f_val, '*');
303 for zz=1:length(p_res)
304     if p_res(zz)>0.05
305         plot(zz, p_res(zz), 'ro');
306     else
307         plot(zz, p_res(zz), 'b*');
308     end
309 end
310 % legend('p>0.05','p<0.05');
311 title('vartest2');
312 ylabel('p <');
313 xlabel('n value');
314 hold off
315 grid on
316 subplot(1,2,2)
317 hold on
318 plot(1:n, F_test, 'bo');
319 index1 = find(z>0.95);
320 plot(1:0.1:n,xx(index1(1))*ones(size(1:0.1:n)), 'r--');
321 title('CDF');
322 xlabel('n-value');

```



```

323 ylabel('F-value')
324 hold off
325 grid on
326 % Zoomed in view of CDF based F-Test
327 figure(17)
328 hold on
329 plot(3:n, F_test(3:end), 'bo');
330 index1 = find(z>0.95);
331 plot(3:0.1:n, xx(index1(1))*ones(size(3:0.1:n)), 'r--');
332 title('CDF zoomed in range n=[3-15]');
333 xlabel('n-value');
334 ylabel('F-value')
335 hold off
336 grid on
337
338 %% Part 1E
339 % An optimal of n=5 is obtained.
340
341 A5_norm = zeros(length(time_norm), 5+1); % initialising normalised A matrix for 5th order
      polynomial
342 A5_norm(:,1) = 1;
343
344 for a = 1:length(time_norm) % Defining normalised A matrix for nth order polynomial
345     for b = 1:5
346         A5_norm(a,b+1) = time_norm(a)^(2*b-1);
347     end
348 end
349 [U5,S5,V5] = svd(A5_norm, 'econ'); % Singular value decomposition of A, to avoid singularity errors
      or solution manifold due to rank deficit
350 condition_number5 = S5(1,1)/S5(end,end); % Indicates the number of potential digit that can be lost
      in any numerical calculations when A is computed directly.
351 S5_inv_diag = [];
352 for nn=1:length(S5)
353     S5_inv_diag = [S5_inv_diag, inv(S5(nn,nn))];
354 end
355 S5_inv = diag(S5_inv_diag);
356 x5 = V5*S5_inv*U5'*freq_norm; % Parameter estimation via SVD aided LSQ
357
358 % x5 = ((transpose(A5_norm)*A5_norm))\transpose(A5_norm)*freq_norm; % Parameter estimation via LSQ
      with pinv.
359
360 freq_fit_norm_5 = A5_norm*x5; % Normalised plot-fitted freq plot
361 % freq_fitn = freq_fit_norm_n* ( max(freq) - min(freq) ) + mean(freq); % De-normalising fitted
      frequency
362 % freq_fitn = freq_fit_norm_n* ( max(freq) - min(freq) ) + ( max(freq) + min(freq))/2; % De-
      normalising fitted frequency
363 freq_fit5 = freq_fit_norm_5* ( max(freq) - min(freq) ) + min(freq); % De-normalising fitted
      frequency
364
365 dfreq = []; % Frequency difference between consecutive freq. elements
366 dtime = []; % Time difference between consecutive time elements
367 for c = 2:length(time)
368     dfreq = [dfreq; freq_fit5(c) - freq_fit5(c-1)];
369     dtime = [dtime; time(c) - time(c-1)];
370 end
371 dfreq_dtime = dfreq./dtime;
372 index_FCA = find( abs(dfreq_dtime)==max(abs(dfreq_dtime))); % Finding the inflection point for
      closest approach
373
374 FCA = freq_fit5(index_FCA); % Frequency of closest approach
375 TCA = time(index_FCA); % Time of closest approach
376
377 %Verification of numerical results with visual results
378
379 figure(18)
380 plot(time(2:end), dfreq_dtime);
381 title('$\frac{dF}{dt}$ vs time', 'Interpreter', 'latex');
382 xlabel('Time [s]');
383 ylabel('$\frac{dF}{dt}$ [$\frac{1}{s^2}$]', 'Interpreter', 'latex');
384 grid on

```