

Chapter 1

Least Squares

1.1 Basic Statistics

1.2 Confidence Intervals, Covariance Matrices, Error Ellipses

1.3 Error Propagation (GLOPOV/SLOPOV)

1.4 Derivations of Least Squares

Least Squares regression solves an over-constrained set of equations as a minimization of the residuals squared(v_i). There are a few different least squares regression methods, which will be described in this section.

$$\min \sum_{i=1}^n v_i^2$$

1.4.1 Ordinary and Weighted Least Squares

Ghalani provides a good basic example explanation of a simple ordinary least squares problem (Section 11.6.1 Ghalani). Consider the 3 equations provided below.

Equations

The system is overconstrained and there is no solution that satisfies each equation.

$$\begin{aligned}x + y &= 3.0 \\2x - y &= 1.5 \\x - y &= 0.2\end{aligned}$$

Observation Equations

In order to solve the solution, we add a *residual* for each equation. This residual captures any error in the measurements. These equations are called the observation equations, and are a foundation of least squares.

$$\begin{aligned}x + y &= 3.0 + v_1 \\2x - y &= 1.5 + v_2 \\x - y &= 0.2 + v_3\end{aligned}$$

Residual Equations

The observation equations can be rewritten as residual equations, shown below.

$$\begin{aligned}v_1 &= x + y - 3.0 \\v_2 &= 2x - y - 1.5 \\v_3 &= x - y - 0.2\end{aligned}$$

The goal of least squares is to determine a solution which minimizes the sum of the square of the residuals... or to get the **least squares solution!**

$$\min \left(\sum_{i=1}^n v_i^2 \right)$$

Sum of Squares Equation

$$f(x, y) = (x + y - 3.0)^2 + (2x - y - 1.5)^2 + (x - y - 0.2)^2$$

The minimum of the sum of squares is where the derivative in the x and y dimension is equal to 0. Therefore we derive a normal equation as the partial derivative with respect to each unknown variable, in this case (x,y).

Normal Equation Derivation

$$\begin{aligned}\frac{\delta f(x, y)}{\delta x} &= 2(x + y - 3.0) + 2(2x - y - 1.5)(2) + 2(x - y - 0.2) \\ \frac{\delta f(x, y)}{\delta y} &= 2(x + y - 3.0) + 2(2x - y - 1.5)(-1) + 2(x - y - 0.2)(-1)\end{aligned}$$

Normal Equations

$$\begin{aligned}6x - 2y - 6.2 &= 0 \\ -2x + 3y - 1.3 &= 0\end{aligned}$$

Solving for the normal equations yields the 'most probable solution'.

Solving Normal Equations Using Matrices

$$\begin{aligned}\begin{bmatrix} 6 & -2 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 6.2 \\ 1.3 \end{bmatrix} \\ \begin{bmatrix} x \\ y \end{bmatrix} &= inv \left(\begin{bmatrix} 6 & -2 \\ -2 & 3 \end{bmatrix} \right) \begin{bmatrix} 6.2 \\ 1.3 \end{bmatrix} \\ \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 1.514 \\ 1.443 \end{bmatrix}\end{aligned}$$

Derive Normal Equations Using Matrices

If we write our initial observation equations as matrices...

$$AX = L + V$$
$$A = \begin{bmatrix} 1 & 1 \\ 2 & -1 \\ 1 & -1 \end{bmatrix} \quad X = \begin{bmatrix} x \\ y \end{bmatrix} \quad L = \begin{bmatrix} 3.0 \\ 1.5 \\ 0.2 \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

Pre-multiplying both sides of the equation by A^T , we get the same Normal Equations as above! (V goes away because the normal equation can be solved without residuals). $A^T A$ is sometimes rewritten as N , or the Normal Matrix.

$$\begin{aligned}AX &= L + V \\ A^T AX &= A^T L \rightarrow \text{sometimes written as } NX = A^T L \\ \begin{bmatrix} 1 & 2 & 1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 2 & -1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 1 & 2 & 1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 3.0 \\ 1.5 \\ 0.2 \end{bmatrix} \\ \begin{bmatrix} 6 & -2 \\ -2 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} &= \begin{bmatrix} 6.2 \\ 1.3 \end{bmatrix}\end{aligned}$$

Notice that this is the same as the normal equations derived above.

Derive Matrix Least Squares Solution (Ordinary Least Squares)

Putting the pieces together, we can now solve for the least squares solution using matrices where X represents the most probable values which are the result of a minimization of the squared residuals of each observation.

$$\begin{aligned}AX &= L + V \\ A^T AX &= A^T L \\ X &= inv(A^T A) A^T L\end{aligned}$$

Add a Weight Term (Stochastic Model)

The weight matrix, W , is used to weight the importance of each observation equation. This is used when you have higher confidence in certain observations, and want those observations to more significantly influence your result. While these weights can be related to anything, it is most common in geomatics that the W matrix is equal to the covariance matrix for all of the observation equations. Normally each observation is assumed to be statistically independent, which would yield a diagonal matrix with no covariances. This is often referred to as Weighted Least Squares (WLS). However, in an example using GPS surveying, 3 observation equations (X,Y,Z) with covariances are created for each measurement. When Covariances are present, this is often referred to as General Least Squares (GLS). The equations to solve the system are exactly the same. Note: Ghilani Textbook does not use the WLS-GLS nomenclature, as he describes GLS the same way others describe TLS online, so I've chosen to follow the "internet" method for this document.

An example weight matrix for 4 observations equations (a,b,c,d) is shown below for WLS and GLS.

$$\begin{aligned} \text{Weighted Least Squares (WLS) } W = \Sigma^{-1} &= \begin{bmatrix} \sigma_{aa}^2 & 0 & 0 & 0 \\ 0 & \sigma_{bb}^2 & 0 & 0 \\ 0 & 0 & \sigma_{cc}^2 & 0 \\ 0 & 0 & 0 & \sigma_{dd}^2 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{\sigma_{aa}^2} & 0 & 0 & 0 \\ 0 & \frac{1}{\sigma_{bb}^2} & 0 & 0 \\ 0 & 0 & \frac{1}{\sigma_{cc}^2} & 0 \\ 0 & 0 & 0 & \frac{1}{\sigma_{dd}^2} \end{bmatrix} \\ \text{General Least Squares (WLS) } W = \Sigma^{-1} &= \begin{bmatrix} \sigma_{aa}^2 & \sigma_{ab}^2 & \sigma_{ac}^2 & \sigma_{ad}^2 \\ \sigma_{ab}^2 & \sigma_{bb}^2 & \sigma_{bc}^2 & \sigma_{bd}^2 \\ \sigma_{ac}^2 & \sigma_{bc}^2 & \sigma_{cc}^2 & \sigma_{cd}^2 \\ \sigma_{ad}^2 & \sigma_{bd}^2 & \sigma_{cd}^2 & \sigma_{dd}^2 \end{bmatrix}^{-1} \end{aligned}$$

Weighted Linear Least Squares and General Least Squares Equation

Incorporating the Weight matrix results in the following, where $A^T W A$ is the Normal Matrix.

$$\begin{aligned} WAX &= WL + WV \\ A^T WAX &= A^T WL \\ X &= \text{inv}(A^T W A) A^T WL \end{aligned}$$

1.4.2 Nonlinear Least Squares

Each of the previous examples only considered linear observation equations, so the values in the A matrix were constants. When this is not the case, an iterative nonlinear least squares approach must be used. Instead of writing the observation equation as $AX = L + V$ as in the linear case, we write the observation equations as $JX = K + V$. Where J is a matrix of partial derivatives of each observation with respect to each unknown. This method is based on Taylor series expansion, and it requires a guess at the initial X (X_0).

The X , in this case, is sometimes written as a ΔX , as it represents a delta from the initial guess. The delta is applied to the initial guess, and a new delta is solved... which is then applied. This iterative approach is performed until it stabilizes on a solution. The exit criteria for the loop is normally based on when the reference variance S_0^2 increases. This either means, your solution is diverging to a poor solution, or the deltas have become so small, that they are "bouncing" up and down about the solution. Other exit criteria are a finite number of iterations, or a small absolute value of the delta values for each unknown. Incorporating a maximum loop iterations is a good idea so that the while loop does not spin off into oblivion.

Weighted Nonlinear Least Squares

Incorporating the Weight Matrix results in the following nonlinear least squares solution.

$$\begin{aligned} WJ\Delta X &= WK + WV \\ J^T WJ\Delta X &= J^T WK \\ \Delta X &= inv(J^T WJ)J^T WK \end{aligned}$$

1.4.3 Total Least Squares (TLS)

*This section is derived from the "General Least Squares" section of Ghilani... From reading online, I believe this is actually called TLS by most other communities.

It is important to recognize that the weight in all of the previous examples is associated with each observation equation, and not each individual variable. For example, when solving a coordinate transformation, it is impossible to have error in both the reference and transformed coordinates. Total Least Squares is required to allow residuals in each of the variables.

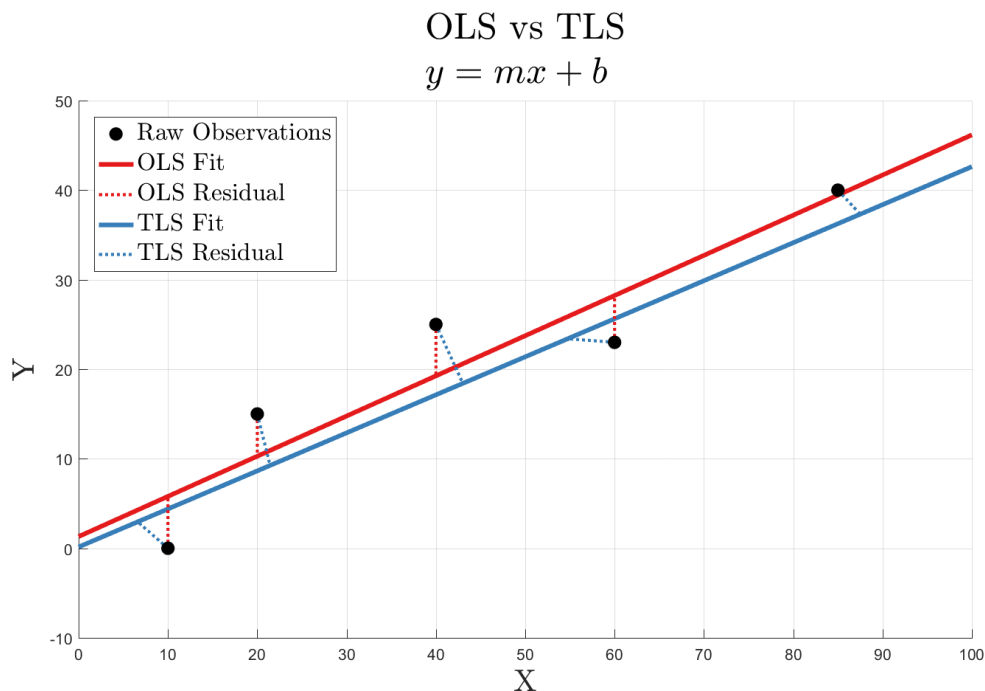
Consider the least squares fit of a line with the observation equation to fit a linear line through a set of data points. Notice that the residual is associated with the y coordinate.

$$mx + b = y + v$$

This is almost always the methodology performed when you do a linear regression with most software packages. If, however, you want to have error in both the x and y coordinate, you must use TLS. The observation equation would then be:

$$m(x + v_x) + b = (y + v_y)$$

A visual plot showing how the fit from TLS is different is shown below:



Total Least Squares Solution

Define your observation equations such that all of the observations and residuals are on the left side of the equation.

$$\begin{aligned} mx + b &= y \\ m(x + v_x) + b &= (y + v_y) \\ m(x + v_x) + b - (y + v_y) &= 0 \end{aligned}$$

Now we want to linearize the equation with respect to the unknowns and the observations.

$$F : m(x + v_x) + b - (y + v_y) = 0$$

$$\begin{aligned} \frac{\delta F}{\delta x}v_x + \frac{\delta F}{\delta y}v_y + \frac{\delta F}{\delta m}d_m + \frac{\delta F}{\delta b}d_b &= 0 - (m_0x + b_0 - y) \\ mv_x - v_y + xd_m + d_b &= 0 - (mx + b - y) \end{aligned}$$

Now if we had 4 points (A,B,C,D), we could write the following equations.

$$\begin{aligned} mv_{x_A} - v_{y_A} + x_Ad_m + d_b &= 0 - (mx_A + b - y_A) \\ mv_{x_B} - v_{y_B} + x_Bd_m + d_b &= 0 - (mx_B + b - y_B) \\ mv_{x_C} - v_{y_C} + x_Cd_m + d_b &= 0 - (mx_C + b - y_C) \\ mv_{x_D} - v_{y_D} + x_Dd_m + d_b &= 0 - (mx_D + b - y_D) \end{aligned}$$

Which can be rewritten in Matrix form:

$$BV + JX = K$$

Where:

$$\begin{aligned} B &= \begin{bmatrix} m & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & m & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & m & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & m & -1 \end{bmatrix} & V &= \begin{bmatrix} v_{x_A} \\ v_{y_A} \\ v_{x_B} \\ v_{y_B} \\ v_{x_C} \\ v_{y_C} \\ v_{x_D} \\ v_{y_D} \end{bmatrix} \\ J &= \begin{bmatrix} x_A & 1 \\ x_A & 1 \\ x_A & 1 \\ x_A & 1 \end{bmatrix} & X &= \begin{bmatrix} dm \\ db \end{bmatrix} & K &= \begin{bmatrix} -(mx_A + b - y_A) \\ -(mx_B + b - y_B) \\ -(mx_C + b - y_C) \\ -(mx_D + b - y_D) \end{bmatrix} \end{aligned}$$

With an optional Covariance Matrix for all of the observations equal to:

$$\Sigma = \begin{bmatrix} \sigma_{x_A x_A}^2 & \sigma_{x_A y_A}^2 & \sigma_{x_A x_B}^2 & \sigma_{x_A y_B}^2 & \sigma_{x_A x_C}^2 & \sigma_{x_A y_C}^2 & \sigma_{x_A x_D}^2 & \sigma_{x_A y_D}^2 \\ \sigma_{x_A y_A}^2 & \sigma_{y_A y_A}^2 & \sigma_{y_A x_B}^2 & \sigma_{y_A y_B}^2 & \sigma_{y_A x_C}^2 & \sigma_{y_A y_C}^2 & \sigma_{y_A x_D}^2 & \sigma_{y_A y_D}^2 \\ \sigma_{x_A x_B}^2 & \sigma_{x_A y_B}^2 & \sigma_{x_B x_B}^2 & \sigma_{x_B y_B}^2 & \sigma_{x_B x_C}^2 & \sigma_{x_B y_C}^2 & \sigma_{x_B x_D}^2 & \sigma_{x_B y_D}^2 \\ \sigma_{x_A y_B}^2 & \sigma_{y_A y_B}^2 & \sigma_{x_B y_B}^2 & \sigma_{y_B y_B}^2 & \sigma_{y_B x_C}^2 & \sigma_{y_B y_C}^2 & \sigma_{y_B x_D}^2 & \sigma_{y_B y_D}^2 \\ \sigma_{x_A x_C}^2 & \sigma_{x_A y_C}^2 & \sigma_{x_B x_C}^2 & \sigma_{y_B x_C}^2 & \sigma_{x_C x_C}^2 & \sigma_{x_C y_C}^2 & \sigma_{x_C x_D}^2 & \sigma_{x_C y_D}^2 \\ \sigma_{x_A y_C}^2 & \sigma_{y_A y_C}^2 & \sigma_{x_B y_C}^2 & \sigma_{y_B y_C}^2 & \sigma_{x_C y_C}^2 & \sigma_{y_C y_C}^2 & \sigma_{y_C x_D}^2 & \sigma_{y_C y_D}^2 \\ \sigma_{x_A x_D}^2 & \sigma_{x_A y_D}^2 & \sigma_{x_B x_D}^2 & \sigma_{y_B x_D}^2 & \sigma_{x_C x_D}^2 & \sigma_{y_C x_D}^2 & \sigma_{x_D x_D}^2 & \sigma_{x_D y_D}^2 \\ \sigma_{x_A y_D}^2 & \sigma_{y_A y_D}^2 & \sigma_{x_B y_D}^2 & \sigma_{y_B y_D}^2 & \sigma_{x_C y_D}^2 & \sigma_{y_C y_D}^2 & \sigma_{x_D y_D}^2 & \sigma_{y_D y_D}^2 \end{bmatrix}$$

Or with no covariances:

$$\Sigma = \begin{bmatrix} \sigma_{x_A}^2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_{y_A}^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{x_B}^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{y_B}^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{x_C}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{y_C}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{x_D}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{y_D}^2 \end{bmatrix}$$

This system of equations is solved by finding an "equivalent solution" by using an "equivalent weight matrix".

$$W_e = \text{inv}(B\Sigma B^T)$$

The "Equivalent Weight Matrix System" is:

$$\begin{aligned} J^T W_e J \Delta X &= J^T W_e K \\ \Delta X &= \text{inv}(J^T W_e J) J^T W_e K \end{aligned}$$

This is a nonlinear problem which must be solved iteratively. The solution is just like nonlinear least squares, except the Weight term is now also be calculated each iteration.

To calculate the residual for each observation, use the following equations, where \hat{X}_i is the current loop guess at the unknowns:

$$\begin{aligned} \text{Equivalent Residuals} &= V_{eq} = J\hat{X}_i - K \\ \text{Observation Residuals} &= V = \Sigma B^T W_{eq} V_{eq} \end{aligned}$$

1.5 Robust and Resistant Regression

Robust Regression attempts to minimize the weight of outliers. So instead of minimizing the square of the residuals (L2 Norm Regression aka Least Squares), one example would be to minimize the absolute value of the residuals (L1 Norm Regression aka Least Absolute Difference LAD). This minimizes the impact of outliers, as their residuals are no longer squared. A more sophisticated example would use "**Iterative Reweighted Least Squares**", where each point receives a new weight based on an equation for the residuals.

Resistant Least Squares attempts to remove the outlier observations altogether in an iterative manner.

These methods are useful for datasets where there are likely outliers or blunders.

*I'm not going to go into too much detail on this right now, but this link seems like a good resource for more a more detailed description: <https://onlinecourses.science.psu.edu/stat501/node/353>

1.6 Variables and Definitions

1.6.1 Calculating Least Squares Parameters

When solving the least squares problem, you can also report other information about the solution. The definition of each of these terms is provided here, and the equation to calculate each term for each least squares method is provided in the summaries.

Residuals (V)

The residuals are easy to compute for ordinary least squares. It is simply the predicted value minus the observed value.

For Ordinary Least Squares:

$$\text{Residuals} = V = AX - L$$

Unit Variance (S_0^2) (*mse* in matlab)

The unit variance is also called "variance of an observation of unit weight" or the "variance of unit weight." Assuming all of the variances used in the initial weight matrix are correct (accurate stochastic model), the a priori value for the unit variance is 1. With a correct stochastic model, the computer reference variance of the solution after a least squares adjustment should also be equal to 1. If it is too high, it indicates that the initial stochastic model was overly optimistic and that the covariance matrix of the observations needed to be scaled up to characterize the observed residuals. If the reference variance is too low, the initial stochastic model was too conservative, and the covariance matrix needed to be scaled down to characterize the observed residuals. In the case where the computed reference variance is too low, the test result is often ignored since correcting it will not change the adjusted coordinates significantly. This is basically saying you measured your points more accurately than you thought you did. If the computed reference variance is too high, this indicates that there may either be blunders in the data or you simply you didn't measure your points as accurately as you thought you did. To test if the computed reference variance is statistically different than the a priori reference variance (1), use the goodness of fit test.

For ordinary least squares is:

$$\text{Reference Variance} = S_0^2 = \frac{V'V}{dof}$$

χ^2 Goodness of Fit Test (is S_0^2 statistically equal to 1?)

A two tailed χ^2 Goodness of Fit Test is used to determine if there is an issue with your least squares adjustment at an α significance level. α values are normally low (eg. 0.01, 0.05).

$$\text{Null Hypothesis } H_0 : S_0^2 = 1$$

$$\text{Alternative Hypothesis } H_a : S_0^2 \neq 1$$

$$\text{Significance} : \alpha$$

Test Statistic:

$$\chi^2 = \frac{vS_0^2}{\sigma^2} = \frac{dof \times S_0^2}{1} = dof \times S_0^2$$

Rejection Region:

$$\begin{aligned} \chi^2 &> \chi_{(\alpha/2, dof)}^2 \\ \chi^2 &< \chi_{(1-\alpha/2, dof)}^2 \end{aligned}$$

If either of the rejection region tests fail at the alpha level, then the null hypothesis is rejected and we can say that the computed reference variance is not equal to the a priori reference variance of 1 at an α significance

level. **More importantly**, and somewhat counter-intuitively, is that if both rejection region tests pass, we say that the computed reference variance is not significantly different than the a priori reference variance at an *alpha* significance level, so **we set the computed reference variance equal to 1!** What we are saying here is that the computed reference variance is just an estimate, and it should be equal to 1. Since that estimate is not statistically different than 1 (which would indicate a blunder or poor stochastic model), we assume that there are no blunders and our stochastic model is correct. If there are no blunders and our stochastic model is correct, the computed reference variance is equal to 1.

Cofactor Matrix (Q)

The Cofactor Matrix is the inverse of the Normal Matrix. When the reference variance is equal to 1, it is the same as the covariance matrix.

need more
info here

Covariance Matrix or Unknowns (Σ_x)

The Covariance Matrix (also Variance-Covariance) describes the variance and covariances between the computed unknown parameters. The variance is the diagonal element, and the off diagonal elements represent how the computed unknown parameters are related. The standard deviation of the unknowns is just the square root of the variances along the diagonal.

Covariance Matrix of Observations (Σ_l)

The covariance matrix describes the variance and covariance of each of your observation equations. So for example, if you have observation equations for distances between a few points, and solve for the coordinates of the unknown points. You can then use this to determine the uncertainty in the distances. This is basically using Σ_x and GLOPOV.

Model Skill (R^2 , sometimes \hat{S})

The Skill of the model, or R^2 , is the amount of variance explained by the model. A R^2 equal to 1 indicates the model perfectly fits the data. A R^2 equal to 0 indicates that the model does not explain any of the variance of the data. **Do not use R^2 for nonlinear least squares**, as the value is not valid.

Extra Sum of Squares Test (Large N approximation)

The model skill will always increase as the number of unknowns increases, so the significant of adding another unknown to a model can be tested with a 'extra sum of squares test.' This test requires that a large number of N observations. The equation below shows the hypothesis test that is performed. to compare if the extra parameter added in model B (with M_2 parameters) is significant compared to model A (with M_1 parameters) at the α confidence level. If the test is true, the extra parameter is not significant.

$$\frac{R_B^2 - R_A^2}{1 - R_B^2} \leq \frac{1}{N} \chi_{(\alpha, M_2 - M_1)}^2$$

RMSE (Root Mean Square Error)

RMSE is another metric to summarize the residuals of the model. A value of 0 indicates a perfect fit. Higher values indicate higher residuals and a less accurate model.

1.7 Ordinary Least Squares (OLS)

1.7.1 Theory

Ordinary Least Squares solves a linear, overconstrained system of equations by minimizing the squared residuals of each observation equation.

$$\min \sum_i^n v_i^2$$

1.7.2 Assumptions

- No Outliers/Blunders OLS is not robust to outliers (consider RANSAC/Robust Weighting if outliers)
- System of equations is linear (eg. derivative wrt each unknown is not a function of any of the unknowns)
- System is over-constrained (eg. Number of Observation Equations > Number of Unknowns)
- Error only in dependent variable (eg. $mx+b = y + v \rightarrow$ error only in y dimension)

1.7.3 Equations

$$AX = L + V$$

$$m = \text{number of observations} \quad n = \text{number of unknowns}$$

$$dof = \text{degrees of freedom (\# of redundant observations)} = m - n$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad L = \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_m \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix}$$

$$\text{Unknowns} = \hat{X} = \text{inv}(A^T A) A^T L$$

$$\text{Residuals} = V = AX - L$$

$$\text{Reference Variance} = S_0^2 = \frac{V^T V}{dof}$$

$$\text{Cofactor Matrix} = Q_{xx} = \text{inv}(A^T A)$$

$$\text{Covariance Matrix of Unknowns} = \Sigma_{xx} = S_0^2 Q_{xx}$$

$$\text{Covariance Matrix of Observations} = \Sigma_{\hat{L}} = A \Sigma_{xx} A^T$$

$$\text{Standard Deviation of Solved Unknowns} = \sigma_{\hat{X}} = \sqrt{\text{diag}(\Sigma_{xx})}$$

$$\text{Predicted L} = \hat{L} = AX$$

$$R^2 \text{ (model skill)} = \frac{\text{var}(\hat{L})}{\text{var}(L)}$$

$$\text{RMSE} = \sqrt{\frac{V^T V}{m}}$$

1.7.4 Sample Problem

Given the points:

$$x = [0, 1, 2, 3, 4] \quad y = [5, 1, 7, 13, 24]$$

Fit a parabola given the observation equation:

$$y = ax^2 + bx + c$$

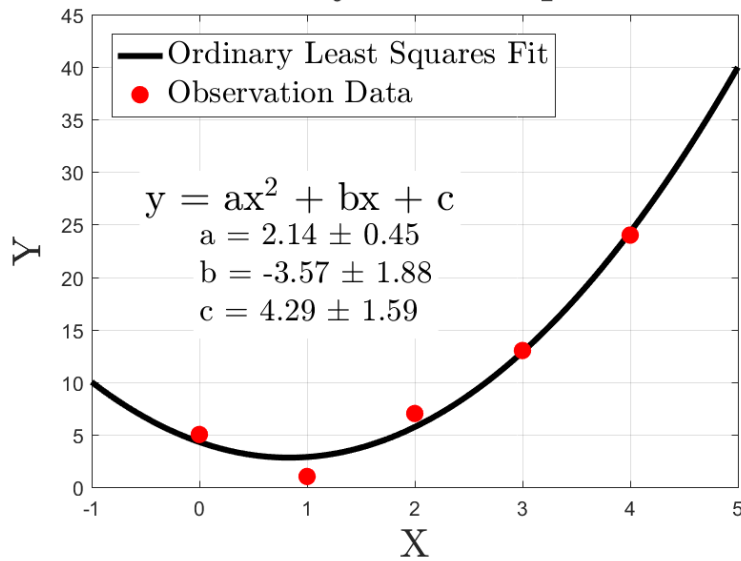
Note that the observation equation is linear. The x^2 term is a constant once observation values are substituted.

$$A = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ x_3^2 & x_3 & 1 \\ x_4^2 & x_4 & 1 \\ x_5^2 & x_5 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 4 & 2 & 1 \\ 9 & 3 & 1 \\ 16 & 4 & 1 \end{bmatrix} \quad X = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad L = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \\ 7 \\ 13 \\ 24 \end{bmatrix}$$

Use the Equations and solve:

$n = 3$	$m = 5$	$dof = 2$
$\hat{X} = \begin{bmatrix} 2.14 \\ -3.57 \\ 4.29 \end{bmatrix}$	$V = \begin{bmatrix} -0.71 \\ 1.86 \\ -1.29 \\ -0.14 \\ 0.29 \end{bmatrix}$	$S_0^2 = 2.86$
$\Sigma_{xx} = \begin{bmatrix} 0.20 & -0.82 & 0.41 \\ -0.82 & 3.55 & -2.20 \\ 0.41 & -2.20 & 2.53 \end{bmatrix}$	$\sigma_{\hat{X}} = \begin{bmatrix} 0.45 \\ 1.88 \\ 1.59 \end{bmatrix}$	$\Sigma_{\hat{U}} = \begin{bmatrix} 2.53 & 0.73 & -0.24 & -0.41 & 0.24 \\ 0.73 & 1.06 & 0.98 & 0.49 & -0.41 \\ -0.24 & 0.98 & 1.39 & 0.98 & -0.24 \\ -0.41 & 0.49 & 0.98 & 1.06 & 0.73 \\ 0.24 & -0.41 & -0.24 & 0.73 & 2.53 \end{bmatrix}$
$Q_{xx} = \begin{bmatrix} 0.07 & -0.29 & 0.14 \\ -0.29 & 1.24 & -0.77 \\ 0.14 & -0.77 & 0.89 \end{bmatrix}$	$\hat{L} = \begin{bmatrix} 4.29 \\ 2.86 \\ 5.71 \\ 12.86 \\ 24.29 \end{bmatrix}$	$R^2 = 0.9821 \quad RMSE = 1.07$

Ordinary Least Squares



1.7.5 Example Matlab Code

Algorithm 1.1: exampleOLS.m

```

1  %% Example Ordinary Least Squares Script
2  %% observations
3  x = [0 1 2 3 4];
4  y = [5 1 7 13 24];
5
6  %% Define A and L based on observation equation
7  A = [x(:).^2 x(:) ones(size(x(:)))];
8  L = y(:);
9
10 %% Calculate Ordinary Least Squares
11 m = numel(L);           % number of observations
12 n = size(A,2);          % number of unknowns
13 dof = m-n;              % degrees of freedom
14 X = (A'*A)\A'*L;        % unknowns(inv(A)*b' with 'A\b' per Matlab)
15 V = A * X - L;          % residuals
16 So2 = V'*V/dof;         % Reference Variance
17 Q = inv(A'*A);          % cofactor
18 Sx = So2 * Q;           % covariance of unknowns
19 Sl = A * Sx * A';        % covariance of observations
20 stdX = sqrt(diag(Sx));   % std of solved unknowns
21 Lhat = A * X;           % predicted L values
22 r2 = var(Lhat)/var(L);   % R^2 Skill
23 RMSE = sqrt(V'*V/m);    % RMSE

```

Algorithm 1.2: The Matlab built in function LSCOV generates the same results

```

25 %% Using Matlab Built in Function LSCOV
26 [matlab_X, matlab_stdX, matlab_So2, matlab_Sx] = lscov(A,L); %same results

```

1.8 Weighted Least Squares (WLS)

1.8.1 Theory

Weighted Least Squares solves a linear, overconstrained system of equations by minimizing the squared residuals of each observation equation, with a weight (W_i) applied to each observation.

$$\min \sum_i^n W_i \times v_i^2$$

*Note that if the scale of the variance in the weight matrix is known to be 1, then the computed reference variance should be inspected to ensure it passes the χ^2 goodness of fit test. If it passes the test, the Covariance matrix should NOT be multiplied by the reference variance. See definition of reference variance for the reasoning.

1.8.2 Assumptions

- No Outliers/Blunders WLS is not robust to outliers (consider RANSAC/Robust Weighting if outliers)
- System of equations is linear (eg. derivative wrt each unknown is not a function of any of the unknowns)
- System is over-constrained (eg. Number of Observation Equations > Number of Unknowns)
- Error only in dependent variable (eg. $mx+b = y + v \rightarrow$ error only in y dimension)
- No covariance between weight of each observation

1.8.3 Equations

$$WAX = WL + WV$$

$$m = \text{number of observations} \quad n = \text{number of unknowns}$$

$$dof = \text{degrees of freedom (\# of redundant observations)} = m - n$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad L = \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_m \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} \quad W = \text{diag} \left(\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} \text{ or } \begin{bmatrix} \sigma_1^2 \\ \sigma_2^2 \\ \vdots \\ \sigma_m^2 \end{bmatrix}^{-1} \right)$$

$$\text{Unknowns} = \hat{X} = \text{inv}(A^T W A) A^T W L$$

$$\text{Residuals} = V = AX - L$$

$$\text{Reference Variance} = S_0^2 = \frac{V' W V}{dof}$$

$$\text{Cofactor Matrix} = Q_{xx} = \text{inv}(A^T W A)$$

$$\text{Covariance Matrix of Unknowns} = \Sigma_{xx} = S_0^2 \times Q_{xx}$$

$$\text{Covariance Matrix of Observations} = \Sigma_{\hat{y}} = A \Sigma_{xx} A^T$$

$$\text{Standard Deviation of Solved Unknowns} = \sigma_{\hat{X}} = \sqrt{\text{diag}(\Sigma_{xx})}$$

$$\text{Predicted L} = \hat{L} = AX$$

$$R^2 \text{ (model skill)} = \frac{\text{var}(\hat{L})}{\text{var}(L)}$$

$$\text{RMSE} = \sqrt{\frac{V' V^T}{m}}$$

1.8.4 Sample Problem

Given the points(x, y) and weights($Weights$):

$$x = [0, 1, 2, 3, 4] \quad y = [5, 1, 7, 13, 24] \quad Weights = [1, 10, 100, 5, 1]$$

Fit a parabola given the observation equation:

$$y = mx + b$$

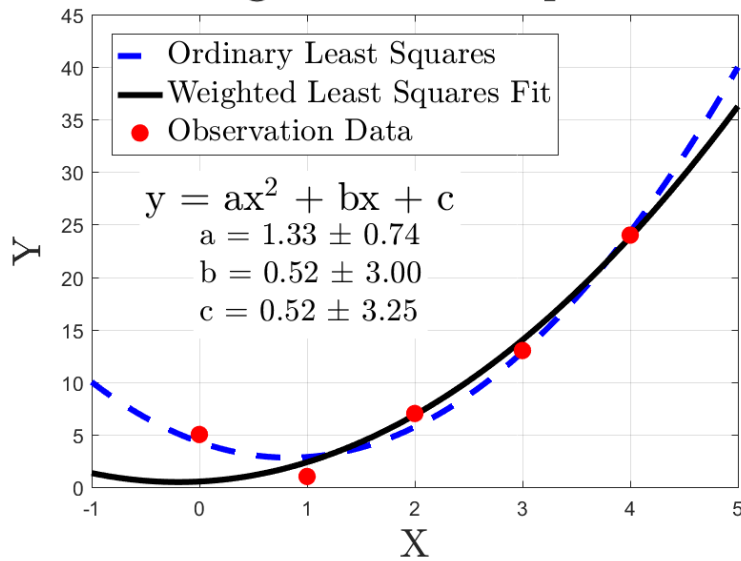
Note that the observation equation is linear, and there are no covariances to the weights. These weights have no units, and are just saying hey, I trust the third measurement 10 times more than I trust the second measurement.

$$A = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ x_4 & 1 \\ x_5 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{bmatrix} \quad X = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad L = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \\ 7 \\ 13 \\ 24 \end{bmatrix} \quad W = \begin{bmatrix} 15 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Use the Equations and solve:

$n = 3$	$m = 5$	$dof = 2$
$\hat{X} = \begin{bmatrix} 1.33 \\ 0.52 \\ 0.52 \end{bmatrix}$	$V = \begin{bmatrix} -4.48 \\ 1.36 \\ -0.14 \\ 1.01 \\ -0.19 \end{bmatrix}$	$S_0^2 = 22.88$
$\Sigma_{xx} = \begin{bmatrix} 0.55 & -2.09 & 1.89 \\ -2.09 & 9.01 & -9.21 \\ 1.89 & -9.21 & 10.59 \end{bmatrix}$	$\sigma_{\hat{X}} = \begin{bmatrix} 0.74 \\ 3.00 \\ 3.25 \end{bmatrix}$	$\Sigma_{\hat{L}} = \begin{bmatrix} 10.59 & 3.28 & -0.25 & 0.02 & 4.07 \\ 3.28 & 1.34 & 0.09 & -0.46 & -0.32 \\ -0.25 & 0.09 & 0.22 & 0.14 & -0.15 \\ 0.02 & -0.46 & 0.14 & 1.82 & 4.59 \\ 4.07 & -0.32 & -0.15 & 4.59 & 13.88 \end{bmatrix}$
$Q_{xx} = \begin{bmatrix} 0.02 & -0.09 & 0.08 \\ -0.09 & 0.39 & -0.40 \\ 0.08 & -0.40 & 0.46 \end{bmatrix}$	$\hat{L} = \begin{bmatrix} 0.52 \\ 2.36 \\ 6.86 \\ 14.01 \\ 23.81 \end{bmatrix}$	$R^2 = 1.1366 \quad RMSE = 2.15$

Weighted Least Squares



1.8.5 Example Matlab Code

Algorithm 1.3: exampleWLS.m

```

1  %% Example Weighted Least Squares Script
2  %% observations
3  x = [0 1 2 3 4];
4  y = [5 1 7 13 24];
5  weights = [1 10 100 5 1];
6
7  %% Define A, L, and W based on observation equation
8  A = [x(:).^2 x(:) ones(size(x(:)))];
9  L = y(:);
10 W = diag(weights);
11
12 %% Calculate Weighted Least Squares
13 m = numel(L);           % number of observations
14 n = size(A,2);          % number of unknowns
15 dof = m-n;              % degrees of freedom
16 X = (A'*W*A)\A'*W*L;    % unknowns(inv(A)*b' with 'A\b' per Matlab)
17 V = A * X - L;          % residuals
18 So2 = V'*W*V/dof;       % Reference Variance
19 Q = inv(A'*W*A);        % cofactor
20 Sx = So2 * Q;           % covariance of unknowns
21 Sl = A * Sx * A';       % covariance of observations
22 stdX = sqrt(diag(Sx));  % std of solved unknowns
23 Lhat = A * X;           % predicted L values
24 r2 = var(Lhat)/var(L);  % R^2 Skill
25 RMSE = sqrt(V'*V/m);    % RMSE

```

Algorithm 1.4: The Matlab built in function LSCOV generates the same results

```

25 %% Using Matlab Built in Function LSCOV
26 [mat_X, mat_stdX, mat_So2, mat_Sx] = lscov(A,L,weights); %same results

```

1.9 General Least Squares (GLS)

1.9.1 Theory

General Least Squares solves a linear, overconstrained system of equations by minimizing the squared residuals of each observation equation, with a weight matrix defining the variance AND covariance of each observation equation. Note that the equations are the same as WLS, except the Weight Matrix (W) contains covariances.

*Note that if the scale of the variance-covariance in the weight matrix is known to be 1, then the computed reference variance should be inspected to ensure it passes the χ^2 goodness of fit test. If it passes the test, the Covariance matrix should NOT be multiplied by the reference variance. See definition of reference variance for the reasoning.

1.9.2 Assumptions

- No Outliers/Blunders WLS is not robust to outliers (consider RANSAC/Robust Weighting if outliers)
- System of equations is linear (eg. derivative wrt each unknown is not a function of any of the unknowns)
- System is over-constrained (eg. Number of Observation Equations > Number of Unknowns)
- Error only in dependent variable (eg. $mx+b = y + v \rightarrow$ error only in y dimension)
- Covariance between weight of each observation

1.9.3 Equations

$$WAX = WL + WV$$

$$m = \text{number of observations} \quad n = \text{number of unknowns}$$

$$dof = \text{degrees of freedom (\# of redundant observations)} = m - n$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} \quad X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad L = \begin{bmatrix} l_1 \\ l_2 \\ \vdots \\ l_m \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} \quad W = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \dots & \sigma_{1m}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \dots & \sigma_{2m}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{m1}^2 & \sigma_{m2}^2 & \dots & \sigma_{mm}^2 \end{bmatrix}^{-1}$$

$$\text{Unknowns} = \hat{X} = \text{inv}(A^T W A) A^T W L$$

$$\text{Residuals} = V = AX - L$$

$$\text{Reference Variance} = S_0^2 = \frac{V' W V}{dof}$$

$$\text{Cofactor Matrix} = Q_{xx} = \text{inv}(A^T W A)$$

$$\text{Covariance Matrix of Unknowns} = \Sigma_{xx} = S_0^2 \times Q_{xx}$$

$$\text{Covariance Matrix of Observations} = \Sigma_{\hat{\hat{y}}} = A \Sigma_{xx} A^T$$

$$\text{Standard Deviation of Solved Unknowns} = \sigma_{\hat{X}} = \sqrt{\text{diag}(\Sigma_{xx})}$$

$$\text{Predicted L} = \hat{L} = AX$$

$$R^2 \text{ (model skill)} = \frac{\text{var}(\hat{L})}{\text{var}(L)}$$

$$\text{RMSE} = \sqrt{\frac{V V^T}{m}}$$

1.9.4 Sample Problem

Given the control coordinates (x_c, y_c) , corresponding measured coordinates (x, y) , and estimated covariance matrix of the control coordinates Σ_c :

$$x_c = [1, 2, 3] \quad y_c = [0, 5, 1] \quad x = [6, 1, 8] \quad y = [3, 12, 8]$$

$$\Sigma_c = \begin{bmatrix} \sigma_{x_{c1}x_{c1}}^2 & \sigma_{x_{c1}y_{c1}}^2 & \sigma_{x_{c1}x_{c2}}^2 & \sigma_{x_{c1}y_{c2}}^2 & \sigma_{x_{c1}x_{c3}}^2 & \sigma_{x_{c1}y_{c3}}^2 \\ \sigma_{x_{c1}y_{c1}}^2 & \sigma_{y_{c1}y_{c1}}^2 & \sigma_{y_{c1}x_{c2}}^2 & \sigma_{y_{c1}y_{c2}}^2 & \sigma_{y_{c1}x_{c3}}^2 & \sigma_{y_{c1}y_{c3}}^2 \\ \sigma_{x_{c1}x_{c2}}^2 & \sigma_{y_{c1}x_{c2}}^2 & \sigma_{x_{c2}x_{c2}}^2 & \sigma_{x_{c2}y_{c2}}^2 & \sigma_{x_{c2}x_{c3}}^2 & \sigma_{x_{c2}y_{c3}}^2 \\ \sigma_{x_{c1}y_{c2}}^2 & \sigma_{y_{c1}y_{c2}}^2 & \sigma_{x_{c2}y_{c2}}^2 & \sigma_{y_{c2}y_{c2}}^2 & \sigma_{y_{c2}x_{c3}}^2 & \sigma_{y_{c2}y_{c3}}^2 \\ \sigma_{x_{c1}x_{c3}}^2 & \sigma_{y_{c1}x_{c3}}^2 & \sigma_{x_{c2}x_{c3}}^2 & \sigma_{y_{c2}x_{c3}}^2 & \sigma_{x_{c3}x_{c3}}^2 & \sigma_{x_{c3}y_{c3}}^2 \\ \sigma_{x_{c1}y_{c3}}^2 & \sigma_{y_{c1}y_{c3}}^2 & \sigma_{x_{c2}y_{c3}}^2 & \sigma_{y_{c2}y_{c3}}^2 & \sigma_{x_{c3}y_{c3}}^2 & \sigma_{y_{c3}y_{c3}}^2 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.3 & 0 & 0 & 0 & 0 \\ 0.3 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.4 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.7 & 0.4 \\ 0 & 0 & 0 & 0 & 0.4 & 0.4 \end{bmatrix}$$

Calculate the 2D Conformal (Four Parameter Similarity Transform) to go from the measured coordinates to the control coordinates. (Note that the block diagonal structure of this covariance matrix describes that the x and y coordinate are correlated to each other for each point, but there is no correlation between points). The 2D Conformal equations are:

$$x_c = (S \cos(\theta))x - (S \sin(\theta))y + T_x$$

$$y_c = (S \sin(\theta))x - (S \cos(\theta))y + T_y$$

By substituting:

$$a = S \cos(\theta)$$

$$b = S \sin(\theta)$$

Where:

$$\theta = \tan^{-1}\left(\frac{b}{a}\right)$$

$$S = \frac{a}{\cos(\theta)}$$

The observation equations become:

$$x_c = ax - by + T_x$$

$$y_c = bx - ay + T_y$$

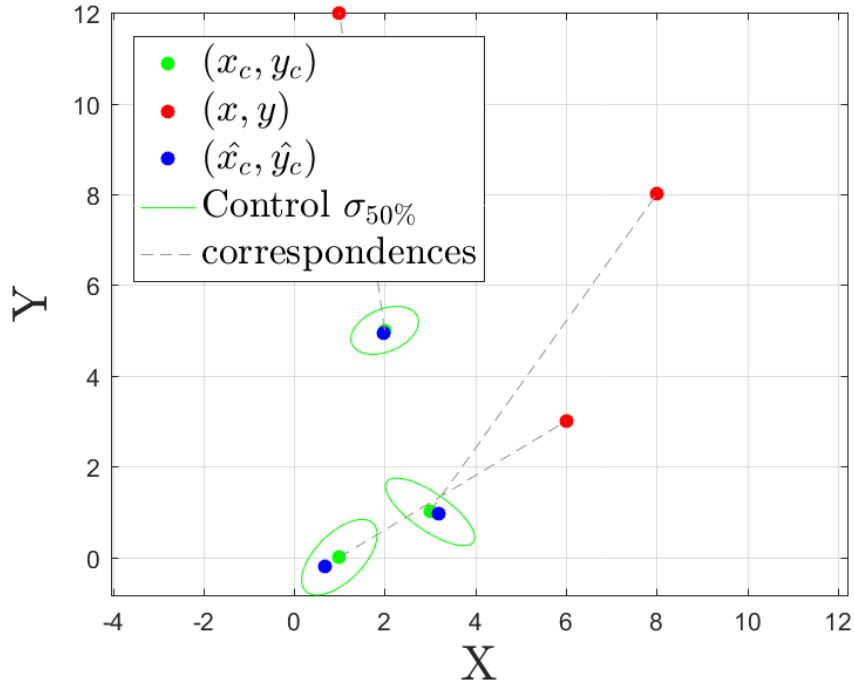
Note that every $(x_a, y_a) \rightarrow (x, y)$ correspondence produces 2 observation equations.

$$A = \begin{bmatrix} x_1 & -y_1 & 1 & 0 \\ y_1 & x_1 & 0 & 1 \\ x_2 & -y_2 & 1 & 0 \\ y_2 & x_2 & 0 & 1 \\ x_3 & -y_3 & 1 & 0 \\ y_3 & x_3 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 6 & -3 & 1 & 0 \\ 3 & 6 & 0 & 1 \\ 1 & -12 & 1 & 0 \\ 12 & 1 & 0 & 1 \\ 8 & -8 & 1 & 0 \\ 8 & 8 & 0 & 1 \end{bmatrix} \quad X = \begin{bmatrix} a \\ b \\ T_x \\ T_y \end{bmatrix} \quad L = \begin{bmatrix} x_{a1} \\ y_{a1} \\ x_{a2} \\ y_{a2} \\ x_{a3} \\ y_{a3} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 2 \\ 5 \\ 3 \\ 1 \end{bmatrix} \quad W = \text{inv}(\Sigma_c)$$

Use the Equations and solve:
Interpret the results:

- θ and S values and their covariances can be calculated using equations: $\theta = \tan^{-1}\left(\frac{b}{a}\right)$ and $S = \frac{a}{\cos(\theta)}$
- R^2 isn't really a valid metric here, the variance of the coordinate doesn't mean anything useful.
- Error is assumed to only be in the control coordinates
- We can use these results to transform other coordinates with uncertainty using GLOPOV
- The S_0^2 value is not near one, indicating the variance-covariance matrix may have over-estimated the predicted error of the control data. a χ^2 goodness of fit test should be done to confirm this.

GLS 2D Conformal Transformation



1.9.5 Example Matlab Code

Algorithm 1.5: exampleGLS.m

```

1  %% Example General Least Squares Script
2  %% observations
3  xc = [1 2 3];
4  yc = [0 5 1];
5  x = [6 1 8];
6  y = [3 12 8];
7
8  Sc = [0.5 0.3 0 0 0 0;
9        0.3 0.5 0 0 0 0;
10       0 0 0.4 0.1 0 0;
11       0 0 0.1 0.2 0 0;
12       0 0 0 0 0.7 -0.4;
13       0 0 0 0 -0.4 0.4]; %variance-covariance of control data
14
15 %% Define A, L, and W based on observation equation
16 A = nan(numel(x)*2,4);
17 A(1:2:end)=[x(:) -y(:) ones(size(x(:))) zeros(size(y(:)))];
18 A(2:2:end)=[y(:) x(:) zeros(size(y(:))) ones(size(x(:)))];
19
20 L = nan(numel(x)*2,1);
21 L(1:2:end) = xc;
22 L(2:2:end) = yc;
23
24 W = inv(Sc);
25
26 %% Calculate General Least Squares
27 m = numel(L);           % number of observations
28 n = size(A,2);          % number of unknowns
29 dof = m-n;              % degrees of freedom
30 X = (A'*W*A)\A'*W*L;    % unknowns(inv(A)*b' with 'A\b' per Matlab)
31 V = A * X - L;          % residuals
32 So2 = V'*W*V/dof;       % Reference Variance
33 Q = inv(A'*W*A);        % cofactor
34 Sx = So2 * Q;           % covariance of unknowns
35 Sl = A * Sx * A';       % covariance of observations
36 stdX = sqrt(diag(Sx));   % std of solved unknowns
37 Lhat = A * X;           % predicted L values
38 r2 = var(Lhat)/var(L);   % R^2 Skill
39 RMSE = sqrt(V'*V/m);     % RMSE

```

Algorithm 1.6: The Matlab built in function LSCOV generates the same results

```

41 %% Using Matlab Built in Function LSCOV
42 [mat_X, mat_stdX, mat_So2, mat_Sx] = lscov(A,L,Sc); %same results

```

1.10 Nonlinear Least Squares

1.10.1 Theory

Nonlinear Least Squares solves a nonlinear, overconstrained system of equations by minimizing the squared residuals of each observation equation. A Taylor Series expansion is utilized to linearize the equation and iteratively calculate the gradient of the function to determine a local minima. Optionally, a weight matrix(W) may be used. Remove the W term, or replace with an identity matrix if a Weight matrix is not desired. Covariances may be 0s if not known.

*Note that if the scale of the variance-covariance in the weight matrix is known to be 1, then the computed reference variance should be inspected to ensure it passes the χ^2 goodness of fit test. If it passes the test, the Covariance matrix should NOT be multiplied by the reference variance. See definition of reference variance for the reasoning.

1.10.2 Assumptions

- No Outliers/Blunders. Nonlinear Least Squares is not robust to outliers (consider RANSAC/Robust Weighting if outliers)
- System of equations is nonlinear (eg. derivative wrt at least one unknown is a function of one of the other unknowns)
- System is over-constrained (eg. Number of Observation Equations $>$ Number of Unknowns)
- Error only in dependent variable (eg. $mx+b = y + v \rightarrow$ error only in y dimension)
- X_0 must be a reasonable guess, otherwise the solution might settle on an incorrect local minima, rather than the global minimum.

1.10.3 Equations

$$AX = L + V$$

$$WJ\Delta X = WK + WV$$

$$m = \text{number of observations} \quad n = \text{number of unknowns}$$

$$dof = \text{degrees of freedom (\# of redundant observations)} = m - n$$

$$i = \text{loop iteration}$$

$$\text{Observation Equations} = F_m(x_1, x_2, \dots, x_n) = l_m \text{ (note: } AX \text{ is obs eqn, } L \text{ is } [l_1, l_2, \dots, l_m] \text{)}$$

$$J_i = \begin{bmatrix} \frac{\delta F_1}{\delta x_1} & \frac{\delta F_1}{\delta x_2} & \dots & \frac{\delta F_1}{\delta x_n} \\ \frac{\delta F_2}{\delta x_1} & \frac{\delta F_2}{\delta x_2} & \dots & \frac{\delta F_2}{\delta x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta F_m}{\delta x_1} & \frac{\delta F_m}{\delta x_2} & \dots & \frac{\delta F_m}{\delta x_n} \end{bmatrix} \quad \Delta X = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_n \end{bmatrix} \quad K_i = \begin{bmatrix} l_1 - F_1(\hat{X}_i) \\ l_2 - F_2(\hat{X}_i) \\ \vdots \\ l_m - F_m(\hat{X}_i) \end{bmatrix} \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix}$$

$$W = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \dots & \sigma_{1m}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \dots & \sigma_{2m}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{m1}^2 & \sigma_{m2}^2 & \dots & \sigma_{mm}^2 \end{bmatrix}^{-1} \quad \text{Initial Guess } \hat{X}_0 = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Loop Equations:

Loop until ΔX is small, or more robustly, loop until S_0^2 increases. S_0^2 will increase slightly when you get down to really really small numbers and the cpu starts rounding. Caveat: it will also increase if you have a really bad initial guess, and it starts diverging.

$$\text{Recalculate } K_i = K(\hat{X}_i)$$

$$\text{Recalculate } J_i = J(\hat{X}_i)$$

$$\text{Loop Delta Estimate} = \Delta X = \text{inv}(J_i^T W J_i) J_i^T W K_i$$

$$\text{Loop Estimate} = \hat{X}_i = X_{i-1} + \Delta X$$

$$\text{Residuals} = V = J_i \Delta \hat{X}_i - K_i$$

$$\text{Reference Variance} = S_0^2 = \frac{V^T W V}{dof}$$

Final Calculations

$$\text{Unknowns} = \hat{X} = \hat{X}_i \text{ (Final Loop Estimate)}$$

$$\text{Cofactor Matrix} = Q_{xx} = \text{inv}(J^T W J)$$

$$\text{Covariance Matrix of Unknowns} = \Sigma_{xx} = S_0^2 \times Q_{xx}$$

$$\text{Covariance Matrix of Observations} = \Sigma_{\hat{y}\hat{y}} = J \Sigma_{xx} J^T$$

$$\text{Standard Deviation of Solved Unknowns} = \sigma_{\hat{X}} = \sqrt{\text{diag}(\Sigma_{xx})}$$

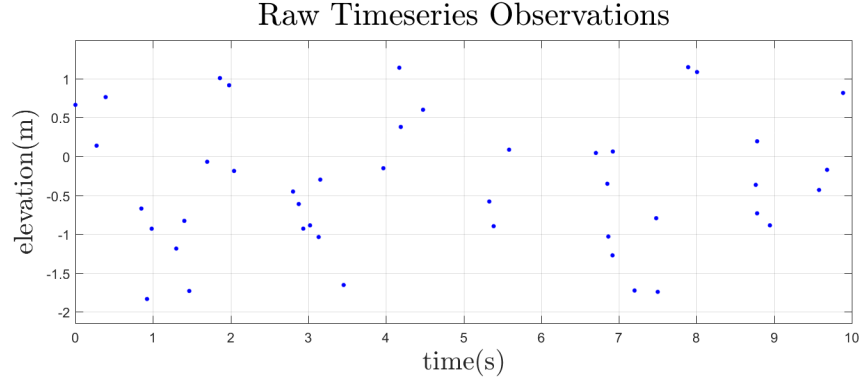
$$\text{Predicted L} = \hat{L} = A \hat{X}$$

$$R^2 \text{ (model skill)} = \text{Not valid for nonlinear least squares}$$

$$\text{RMSE} = \sqrt{\frac{V V^T}{m}}$$

1.10.4 Sample Problem

Given a timeseries dataset (ts.csv) of a harmonic signal with a known period and estimated variance for each measurement, calculate the amplitude and Phase.



The governing equation for the system is:

$$F : \quad A \times \sin\left(\frac{2\pi}{T}t + \phi\right) = y$$

Where:

$$\text{Amplitude(m)} = A$$

$$\text{Period(s)} = T = 2$$

$$\text{Phase(s)} = \phi$$

$$\text{Elevation Measurement(m)} = y$$

$$\text{Time Measurement(s)} = t$$

Note that this system is Nonlinear due to the Phase term. The Jacobian is solved using the partial derivatives.

$$J_i = \begin{bmatrix} \frac{\delta F_1}{\delta A} & \frac{\delta F_1}{\delta \phi} \\ \frac{\delta F_2}{\delta A} & \frac{\delta F_2}{\delta \phi} \\ \vdots & \vdots \\ \frac{\delta F_m}{\delta A} & \frac{\delta F_m}{\delta \phi} \end{bmatrix} = \begin{bmatrix} \sin\left(\frac{2\pi}{T}t_1 + \phi_i\right) & A_i \times \cos\left(\frac{2\pi}{T}t_1 + \phi_i\right) \\ \sin\left(\frac{2\pi}{T}t_2 + \phi_i\right) & A_i \times \cos\left(\frac{2\pi}{T}t_2 + \phi_i\right) \\ \vdots & \vdots \\ \sin\left(\frac{2\pi}{T}t_m + \phi_i\right) & A_i \times \cos\left(\frac{2\pi}{T}t_m + \phi_i\right) \end{bmatrix}$$

$$\Delta X = \begin{bmatrix} \Delta A \\ \Delta \phi \end{bmatrix} \quad K_i = \begin{bmatrix} l_1 - F_1(\hat{X}_i) \\ l_2 - F_2(\hat{X}_i) \\ \vdots \\ l_m - F_m(\hat{X}_i) \end{bmatrix} = \begin{bmatrix} y_1 - A_i \times \sin\left(\frac{2\pi}{T}t_1 + \phi_i\right) \\ y_2 - A_i \times \sin\left(\frac{2\pi}{T}t_2 + \phi_i\right) \\ \vdots \\ y_m - A_i \times \sin\left(\frac{2\pi}{T}t_m + \phi_i\right) \end{bmatrix}$$

$$\text{Initial guess from looking at the plot} = X_0 = \begin{bmatrix} A_0 \\ \phi_0 \end{bmatrix} = \begin{bmatrix} 1.5 \\ 1 \end{bmatrix}$$

Solving using Equations:

$n = 2$	$m = 50$	$dof = 48$
$\hat{X} = \begin{bmatrix} 1.43 \\ 0.90 \end{bmatrix}$	$S_0^2 = 2.77$	$Q_{xx} = \begin{bmatrix} 0.01 & 0.00 \\ 0.00 & 0.00 \end{bmatrix}$
$\Sigma_{xx} = \begin{bmatrix} 0.0202 & 0.0004 \\ 0.0004 & 0.0003 \end{bmatrix}$	$\sigma_{\hat{X}} = \begin{bmatrix} 0.1421 \\ 0.0174 \end{bmatrix}$	$RMSE = 0.66$

A two tailed χ^2 Goodness of Fit Test is performed at the 0.05 Significance level:

Null Hypothesis $H_0 : S_0^2 = 1$

Alternative Hypothesis $H_a : S_0^2 \neq 1$

Significance : $\alpha = 0.05$

Test Statistic:

$$\chi^2 = \frac{vS_0^2}{\sigma^2} = \frac{dof \times S_0^2}{1} = dof \times S_0^2 = 133.07$$

Rejection Region:

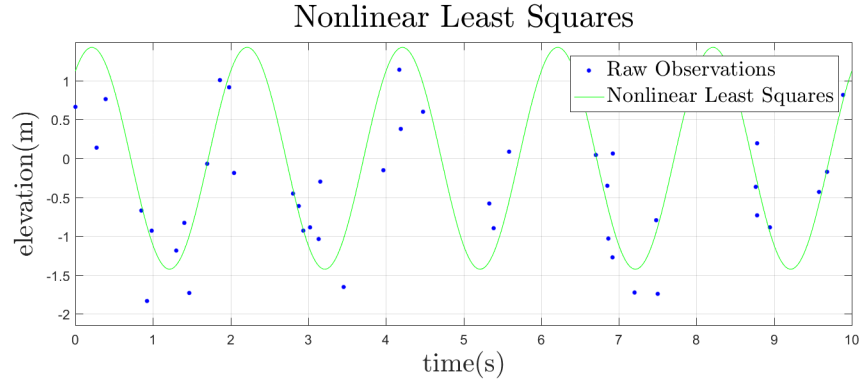
$$133.07 = \chi^2 > \chi_{(\alpha/2, dof)}^2 = 30.75 \quad (PASS)$$

$$133.07 = \chi^2 < \chi_{(1-\alpha/2, dof)}^2 = 69.02 \quad (FAIL)$$

Results:

The results fail the χ^2 Goodness of Fit Test at the 0.05 significance level, so check for blunders or a poor stochastic model

$S_0^2 = 1.00$	$\Sigma_{xx} = \begin{bmatrix} 0.0073 & 0.0001 \\ 0.0001 & 0.0001 \end{bmatrix}$	$\sigma_{\hat{X}} = \begin{bmatrix} 0.0853 \\ 0.0105 \end{bmatrix}$
----------------	--	---



Note: The previous example could have been calculated using Weighted Least Squares (WLS)
The Observation Equation can be rewritten as:

$$A \times \sin\left(\frac{2\pi}{T}t\right) + B \times \cos\left(\frac{2\pi}{T}t\right) = y$$

Where A , B , T are unknowns and:

$$Amp = \sqrt{A^2 + B^2}$$

$$\phi = \tan^{-1}(B/A)$$

1.10.5 Example Matlab Code

Algorithm 1.7: exampleNonlinearLS.m

```

1  %% Example Nonlinear Least Squares Script
2  %% read data
3  data=csvread('ts.csv');
4  t = data(:,1);           % raw time observations
5  y = data(:,2);           % raw elevation observations
6  stdy = data(:,3);        % reported std for each observation
7  W = inv(diag(stdy.^2));  % Create a Weight Matrix based on the stdy
8
9  %% Solve Least Squares
10 X = [1.5; 1];            % initial unknowns guess
11
12 So2 = inf; dSo2 = 1; iter = 0; % initialize for while loop
13
14 m = NPTS;                % number of observations
15 n = 2;                   % number of unknowns
16 dof = m-n;               % degrees of freedom
17 while dSo2>=0 && iter<100 %loop until So2 increases or exceed 100 iteration
18     J = [sin(2*pi/2.*t + X(2)) X(1)*cos(2*pi/2.*t + X(2))];
19     K = y - X(1)*sin(2*pi/2.*t + X(2));
20     dX = (J'*W*J)\J'*W*K; % Loop Delta Estimate
21     X = X + dX;           % Loop Estimate
22     V = J*dX-K;           % Residuals
23     dSo2 = So2 - V'*W*V/dof; % Change in Reference Variance
24     So2 = (V'*W*V)/dof;   % Reference Variance
25     iter = iter + 1;
26 end
27 Q = inv(J'*W*J);         % cofactor
28 Sx = So2 * Q;            % covariance of unknowns
29 Sl = J * Sx * J';        % covariance of observations
30 stdX = sqrt(diag(Sx));   % std of solved unknowns
31 Lhat = J * X;            % predicted L values
32 RMSE = sqrt(V'*V/m);     % RMSE
33
34 % Perform chi^2 Test
35 alphaval = 0.05;
36 chi2 = dof * So2;
37 chi2low = chi2inv(alphaval/2,dof);
38 chi2high = chi2inv(1-alphaval/2,dof);
39 % Recalculate Values with So2 = 1 because CHI2 passed
40 So2_b = 1;               % set reference variance equal to 1
41 Sx_b = So2_b * Q;        % covariance of unknowns w/ So2 = 1
42 Sl_b = J * Sx_b * J';    % covariance of observations w/ So2 = 1
43 stdX_b = sqrt(diag(Sx_b)); % std of solved unknowns w/ So2 = 1

```

Algorithm 1.8: The Matlab built in function NLINFIT generates the same results

```

45 %% Using Matlab Built in Function NLINFIT
46 beta0 = [1.5; 1];
47 modelfun = @(b,x)(b(1)*sin(2*pi/2*x+b(2)));
48 [mat_X,mat_V,mat_J,mat_Sx,mat_So2,ErrorModelInfo] = ...
49     nlinfit(t,y,modelfun,beta0,'Weights',1./noisyscale.^2);

```

1.11 Total Least Squares (TLS)

1.11.1 Theory

Total Least Squares solves a nonlinear, overconstrained system of equations by minimizing the squared residuals of each observation. A Taylor Series expansion is utilized to linearize the equation and iteratively calculate the gradient of the function to determine a local minima. A Weight Matrix, with a column/row for each observation rather than observation equation, is used to allow for error in all dimensions. Traditionally, this will be a predicted variance for each observation.

*Note that if the scale of the variance-covariance in the weight matrix is known to be 1, then the computed reference variance should be inspected to ensure it passes the χ^2 goodness of fit test. If it passes the test, the Covariance matrix should NOT be multiplied by the reference variance. See definition of reference variance for the reasoning.

1.11.2 Assumptions

- No Outliers/Blunders. Nonlinear Least Squares is not robust to outliers (consider RANSAC/Robust Weighting if outliers)
- System is over-constrained (eg. Number of Observation Equations > Number of Unknowns)
- Error is in all measurements variable (eg. $mx+b = y + v \rightarrow$ error only in x and y dimension)
- X_0 must be a reasonable guess, otherwise the solution might settle on an incorrect local minima, rather than the global minimum. If a linear problem, one method is to solve the unweighted OLS, then use that to initialize X_0 in TLS.

1.11.3 Equations

$$AX = L \quad (\text{note: residuals in both X and L})$$

$$BV + J\Delta X = K$$

$$m = \text{number of observations} \quad n = \text{number of unknowns}$$

$$dof = \text{degrees of freedom (\# of redundant observations)} = m - n$$

$$i = \text{loop iteration}$$

$$\text{Observation Equations} = F_m(x_1, x_2, \dots, x_n) = l_m \quad (\text{note: } AX \text{ is obs eqn, L is } [l_1, l_2, \dots, l_m])$$

$$B = \begin{bmatrix} \frac{\delta F_1}{\delta v_1} & \frac{\delta F_1}{\delta v_2} & \dots & \frac{\delta F_1}{\delta v_n} & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\delta F_2}{\delta v_1} & \frac{\delta F_2}{\delta v_2} & \dots & \frac{\delta F_2}{\delta v_n} & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & \frac{\delta F_m}{\delta v_1} & \frac{\delta F_m}{\delta v_2} & \dots & \frac{\delta F_m}{\delta v_n} \end{bmatrix} \quad V = \begin{bmatrix} v_{11} \\ v_{21} \\ \vdots \\ v_{n1} \\ v_{12} \\ v_{22} \\ \vdots \\ v_{n2} \\ \vdots \\ v_{1m} \\ v_{2m} \\ \vdots \\ v_{nm} \end{bmatrix}$$

$$J = \begin{bmatrix} \frac{\delta F_1}{\delta x_1} & \frac{\delta F_1}{\delta x_2} & \cdots & \frac{\delta F_1}{\delta x_n} \\ \frac{\delta F_2}{\delta x_1} & \frac{\delta F_2}{\delta x_2} & \cdots & \frac{\delta F_2}{\delta x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta F_m}{\delta x_1} & \frac{\delta F_m}{\delta x_2} & \cdots & \frac{\delta F_m}{\delta x_n} \end{bmatrix} \quad \Delta X = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_n \end{bmatrix} \quad K = \begin{bmatrix} l_1 - F_1(X_i) \\ l_2 - F_2(X_i) \\ \vdots \\ l_m - F_m(X_i) \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \cdots & \sigma_{1(n \times m)}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \cdots & \sigma_{2(n \times m)}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{(n \times m)1}^2 & \sigma_{(n \times m)2}^2 & \cdots & \sigma_{(n \times m)(n \times m)}^2 \end{bmatrix} \quad \text{Initial Guess } X_0 = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Loop Equations:

Loop until ΔX is small, or more robustly, loop until S_0^2 increases. S_0^2 will increase slightly when you get down to really really small numbers and the cpu starts rounding. Caveat: it will also increase if you have a really bad initial guess, and it starts diverging.

$$\begin{aligned} \text{Equivalent Weight} &= W_{eq} = inv(B \Sigma B^T) \\ \text{Loop Delta Estimate} &= \Delta X = inv(J^T W_{eq} J) J^T W_{eq} K \\ \text{Loop Estimate} &= \hat{X}_i = X_{i-1} + \Delta X \\ \text{Equivalent Residuals} &= V_{eq} = J \Delta \hat{X}_i - K \\ \text{Reference Variance} &= S_0^2 = \frac{V_{eq}^T W_{eq} V_{eq}}{dof} \end{aligned}$$

Final Calculations

$$\begin{aligned} \text{Observation Residuals} &= V = \Sigma B^T W_{eq} V_{eq} \\ \text{Unknowns} &= \hat{X} = \hat{X}_i \text{ (Final Loop Estimate)} \\ \text{Cofactor Matrix} &= Q_{xx} = inv(J^T W_{eq} J) \\ \text{Covariance Matrix of Unknowns} &= \Sigma_{xx} = S_0^2 \times Q_{xx} \\ \text{Covariance Matrix of Observations} &= \Sigma_{\hat{I}\hat{I}} = J \Sigma_{xx} J^T \\ \text{Standard Deviation of Solved Unknowns} &= \sigma_{\hat{X}} = \sqrt{diag(\Sigma_{xx})} \\ \text{RMSE} &= \sqrt{\frac{V_{eq}^T V_{eq}}{m}} \end{aligned}$$

1.11.4 Sample Problem

Given the points and covariance matrix for each observation:

$$x = [10, 20, 60, 40, 85] \quad y = [0, 15, 23, 25, 40]$$

$$\Sigma = \begin{bmatrix} \sigma_{x_1x_1}^2 & \sigma_{x_1y_1}^2 & \sigma_{x_1x_2}^2 & \sigma_{x_1y_2}^2 & \sigma_{x_1x_3}^2 & \sigma_{x_1y_3}^2 & \sigma_{x_1x_4}^2 & \sigma_{x_1y_4}^2 & \sigma_{x_1x_5}^2 & \sigma_{x_1y_5}^2 \\ \sigma_{x_1y_1}^2 & \sigma_{y_1y_1}^2 & \sigma_{y_1x_2}^2 & \sigma_{y_1y_2}^2 & \sigma_{y_1x_3}^2 & \sigma_{y_1y_3}^2 & \sigma_{y_1x_4}^2 & \sigma_{y_1y_4}^2 & \sigma_{y_1x_5}^2 & \sigma_{y_1y_5}^2 \\ \sigma_{x_1x_2}^2 & \sigma_{y_1x_2}^2 & \sigma_{x_2x_2}^2 & \sigma_{x_2y_2}^2 & \sigma_{x_2x_3}^2 & \sigma_{x_2y_3}^2 & \sigma_{x_2x_4}^2 & \sigma_{x_2y_4}^2 & \sigma_{x_2x_5}^2 & \sigma_{x_2y_5}^2 \\ \sigma_{x_1y_2}^2 & \sigma_{y_1y_2}^2 & \sigma_{x_2y_2}^2 & \sigma_{y_2y_2}^2 & \sigma_{y_2x_3}^2 & \sigma_{y_2y_3}^2 & \sigma_{y_2x_4}^2 & \sigma_{y_2y_4}^2 & \sigma_{y_2x_5}^2 & \sigma_{y_2y_5}^2 \\ \sigma_{x_1x_3}^2 & \sigma_{y_1x_3}^2 & \sigma_{x_2x_3}^2 & \sigma_{y_2x_3}^2 & \sigma_{x_3x_3}^2 & \sigma_{x_3y_3}^2 & \sigma_{x_3x_4}^2 & \sigma_{x_3y_4}^2 & \sigma_{x_3x_5}^2 & \sigma_{x_3y_5}^2 \\ \sigma_{x_1y_3}^2 & \sigma_{y_1y_3}^2 & \sigma_{x_2y_3}^2 & \sigma_{y_2y_3}^2 & \sigma_{x_3y_3}^2 & \sigma_{y_3y_3}^2 & \sigma_{y_3x_4}^2 & \sigma_{y_3y_4}^2 & \sigma_{y_3x_5}^2 & \sigma_{y_3y_5}^2 \\ \sigma_{x_1x_4}^2 & \sigma_{y_1x_4}^2 & \sigma_{x_2x_4}^2 & \sigma_{y_2x_4}^2 & \sigma_{x_3x_4}^2 & \sigma_{y_3x_4}^2 & \sigma_{x_4x_4}^2 & \sigma_{x_4y_4}^2 & \sigma_{x_4x_5}^2 & \sigma_{x_4y_5}^2 \\ \sigma_{x_1y_4}^2 & \sigma_{y_1y_4}^2 & \sigma_{x_2y_4}^2 & \sigma_{y_2y_4}^2 & \sigma_{x_3y_4}^2 & \sigma_{y_3y_4}^2 & \sigma_{x_4y_4}^2 & \sigma_{y_4y_4}^2 & \sigma_{x_4x_5}^2 & \sigma_{x_4y_5}^2 \\ \sigma_{x_1x_5}^2 & \sigma_{y_1x_5}^2 & \sigma_{x_2x_5}^2 & \sigma_{y_2x_5}^2 & \sigma_{x_3x_5}^2 & \sigma_{y_3x_5}^2 & \sigma_{x_4x_5}^2 & \sigma_{x_4y_5}^2 & \sigma_{x_5x_5}^2 & \sigma_{x_5y_5}^2 \\ \sigma_{x_1y_5}^2 & \sigma_{y_1y_5}^2 & \sigma_{x_2y_5}^2 & \sigma_{y_2y_5}^2 & \sigma_{x_3y_5}^2 & \sigma_{y_3y_5}^2 & \sigma_{x_4y_5}^2 & \sigma_{y_4y_5}^2 & \sigma_{x_5y_5}^2 & \sigma_{y_5y_5}^2 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 45 & -30 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -30 & 30 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 20 & -10 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -10 & 70 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 80 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 40 & -13 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -13 & 60 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & -25 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -25 & 30 & 30 \end{bmatrix}$$

Fit a line given the observation equation:

$$mx + b = y$$

With residuals for each observation:

$$F : \quad m(x + v_x) + b - (y + v_y) = 0$$

$$\frac{\delta F}{\delta v_x} = m$$

$$\frac{\delta F}{\delta v_y} = -1$$

Solving for Partial Derivatives:

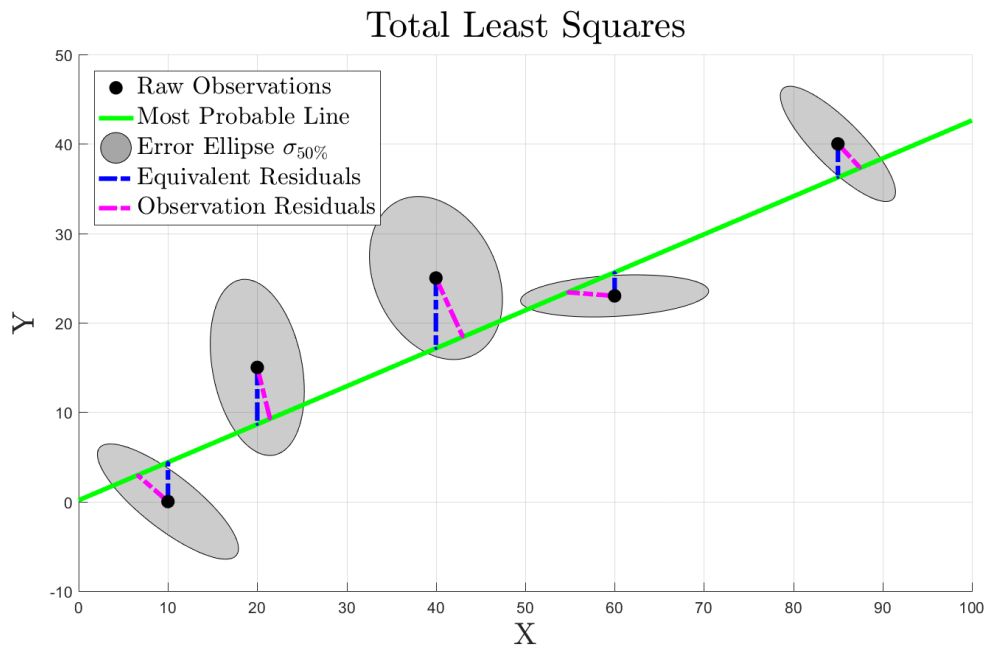
$$B = \begin{bmatrix} \frac{\delta F_1}{\delta v_{x_1}} & \frac{\delta F_1}{\delta v_{y_1}} & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & \frac{\delta F_2}{\delta v_{x_2}} & \frac{\delta F_2}{\delta v_{y_2}} & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\delta F_5}{\delta v_{x_5}} & \frac{\delta F_5}{\delta v_{y_5}} & 0 \end{bmatrix} = \begin{bmatrix} m & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & m & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & m & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & m & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & m & -1 \end{bmatrix}$$

$$V = \begin{bmatrix} v_{x_1} \\ v_{y_1} \\ v_{x_2} \\ v_{y_2} \\ v_{x_3} \\ v_{y_3} \\ v_{x_4} \\ v_{y_4} \\ v_{x_5} \\ v_{y_5} \end{bmatrix} \quad J_i = \begin{bmatrix} \frac{\delta F_1}{\delta m} & \frac{\delta F_1}{\delta b} \\ \frac{\delta F_2}{\delta m} & \frac{\delta F_2}{\delta b} \\ \frac{\delta F_3}{\delta m} & \frac{\delta F_3}{\delta b} \\ \frac{\delta F_4}{\delta m} & \frac{\delta F_4}{\delta b} \\ \frac{\delta F_5}{\delta m} & \frac{\delta F_5}{\delta b} \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ x_4 & 1 \\ x_5 & 1 \end{bmatrix} \quad \Delta X = \begin{bmatrix} \Delta m \\ \Delta b \end{bmatrix}$$

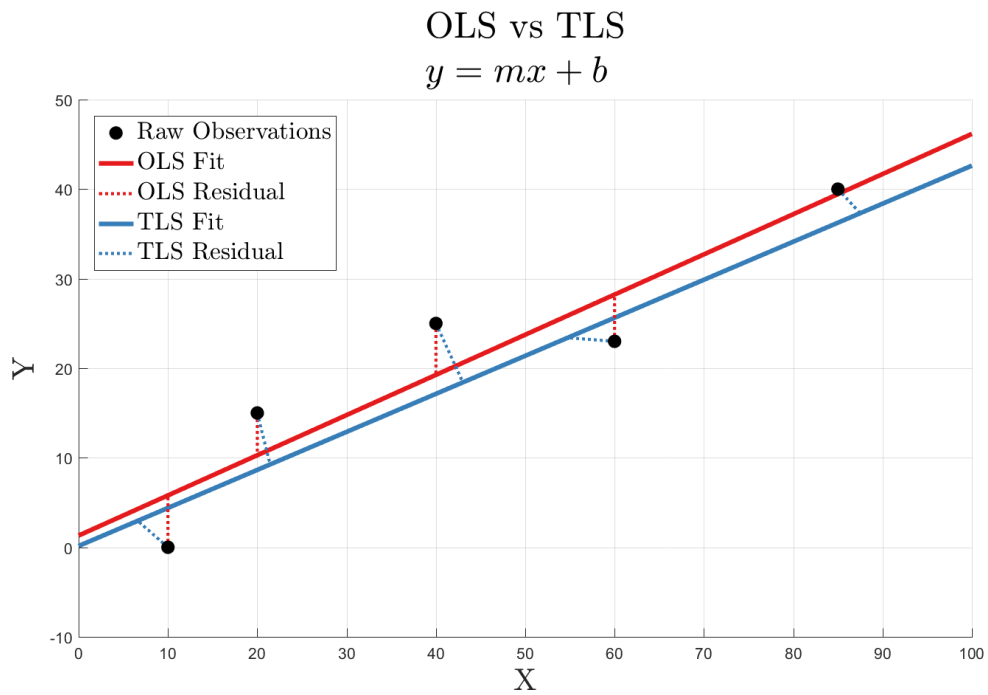
$$K_i = \begin{bmatrix} l_1 - F_1(X_i) \\ l_2 - F_2(X_i) \\ \vdots \\ l_m - F_m(X_i) \end{bmatrix} = \begin{bmatrix} 0 - (m_i x_1 + b_i - y_1) \\ 0 - (m_i x_2 + b_i - y_2) \\ 0 - (m_i x_3 + b_i - y_3) \\ 0 - (m_i x_4 + b_i - y_4) \\ 0 - (m_i x_5 + b_i - y_5) \end{bmatrix} \quad \text{Initial Guess } X_0 = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$$

Solve Using Equations:

$n = 2$	$m = 5$	$dof = 3$
$\hat{X} = \begin{bmatrix} 0.42 \\ 0.15 \end{bmatrix}$	$V_{eq} = \begin{bmatrix} 4.39 \\ -6.36 \\ 2.63 \\ -7.86 \\ -3.75 \end{bmatrix}$	$V = \begin{bmatrix} 3.39 \\ -2.95 \\ -1.43 \\ 5.75 \\ 5.25 \\ -0.40 \\ -3.01 \\ 6.58 \\ -2.50 \\ 2.69 \end{bmatrix}$
$\Sigma_{xx} = \begin{bmatrix} 0.01 & -0.60 \\ -0.60 & 36.87 \end{bmatrix}$	$\sigma_{\hat{X}} = \begin{bmatrix} 0.11 \\ 6.07 \end{bmatrix}$	$\Sigma_{\hat{U}} = \begin{bmatrix} 26.05 & 21.22 & 1.92 & 11.57 & -10.14 \\ 21.22 & 17.57 & 2.94 & 10.25 & -6.20 \\ 1.92 & 2.94 & 7.01 & 4.98 & 9.56 \\ 11.57 & 10.25 & 4.98 & 7.62 & 1.68 \\ -10.14 & -6.20 & 9.56 & 1.68 & 19.41 \end{bmatrix}$
$Q_{xx} = \begin{bmatrix} 0.02 & -0.78 \\ -0.78 & 48.19 \end{bmatrix}$	$\hat{L} = \begin{bmatrix} 4.39 \\ 8.64 \\ 25.63 \\ 17.14 \\ 36.25 \end{bmatrix}$	$S_0^2 = 0.76 \quad RMSE = 5.46$



Notice how the observation residuals are not just in the y dimension. OLS is calculated with the same data, and plotted below for comparison.



1.11.5 Example Matlab Code

Algorithm 1.9: exampleTLS.m

```

1 %% Example TLS Script
2 %% Input data
3 x = [10 20 60 40 85];
4 y = [0 15 23 25 40];
5 S = blkdiag([45 -30;-30 30],[20 -10;-10 70],[80 4;4 4],[40 -13;-13 60],[30 -25;-25 30]);
6
7 %% Solve TLS
8 X = [0.5; 0]; % initial unknowns guess
9
10 So2 = inf; dSo2 = 1; iter = 0; % initialize for while loop
11
12 m = numel(x); % number of observations
13 n = numel(X); % number of unknowns
14 dof = m-n; % degrees of freedom
15 while iter<100 %loop until So2 increases or exceed 100 iteration
16     B = blkdiag([X(1) -1],[X(1) -1],[X(1) -1],[X(1) -1],[X(1) -1]); %B
17     J = [x(:) ones(size(x(:)))]; % Jacobian
18     K = -(X(1)*x(:) + X(2) - y(:)); % K Matrix
19     Weq = inv(B*S*B');
20     dX = (J'*Weq*J)\J'*Weq*K; % Loop Delta Estimate
21     X = X + dX; % Loop Estimate
22     Veq = J*dX-K; % Residuals
23     dSo2 = So2 - Veq'*Weq*Veq/dof; % Change in Reference Variance
24     So2 = (Veq'*Weq*Veq)/dof; % Reference Variance
25     iter = iter + 1;
26 end
27
28 V = S * B' * Weq * Veq; % Observation Residuals
29 Q = inv(J'*Weq*J); % cofactor
30 Sx = So2 * Q; % covariance of unknowns
31 Sl = J * Sx * J'; % covariance of observations
32 stdX = sqrt(diag(Sx)); % std of solved unknowns
33 Lhat = J * X; % predicted L values
34 RMSE = sqrt(V'*V/m); % RMSE

```

Note: Matlab does not have a built in TLS function.

1.12 Example: TPU linear line fit

$$y = mx + b$$

TPU GLOPOV vs SLOPOV comparison of uncertainty from fit

1.13 Example: RANSAC Plane Fit

1.14 Example: 3D Conformal Transformation with TLS

1.15 Example: Space Resection

1.16 Example: TPU Ortho From Space Resection

1.17 Example: Space Intersection

1.18 Example: SfM

1.19 References