International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

# Error Matrix

**Shahbaj Ahmad**

MS Computer Science, International University of Applied Sciences
Email I'd: shahbaj.ahmad[at]iu-study.org
Tutor's name: Lino Antoni Giefer

**Abstract:** ***Error calculation is extremely important. It gives feedback about the quality of the scientific procedure, whether it is an experiment or any other technique. Finding the right kind of error metric, however, can be challenging. In the case of mesh simplification, losing even a small amount of accuracy might cause problems, for example, the incapability of 3D printing.***

**Keywords:** Mean Squared Error, Root Mean Square Error, Mean Absolute Error, Mean Absolute Scaled Error, Mean Absolute Percentage Error, and Symmetric Mean Absolute Error

## 1. Symbols and abbreviations

| Label | Name | Symbol |
|---|---|---|
| Abbreviation | Mean Squared Error | MSE |
| | Root Mean Square Error | RMSR |
| | Mean Absolute Error | MAE |
| | Mean Absolute Scaled Error | MASE |
| | Mean Absolute Percentage Error | MAPE |
| | Symmetric Mean Absolute Error | SMAE |
| | Quadric Error Metric | QEM |
| | Quadric Error Collapse Decimation | QWCD |
| | Discrete Differential Error Metric | DDEM |

| | | |
|---|---|---|
| Mathematical object | Quadric error metric matrix | Q |
| | Vertex coordinates | v |
| | Plane | P |
| | Error metric | Δ |
| | Tangential error metric matrix | T |
| | Discrete curvature error matrix | C |

| | | |
|---|---|---|
| Miscellaneous | matrix transpose | T |

## 2. Introduction

### 2.1 Measurement vs. ideality

Nothing is without flaws in reality. There is always some uncertainty and some errors that can and will occur. Every scientific experiment, regardless of high technology, has its limits. When measurements are made, scientists can approach them in two different ways: they either already know what to expect to some certain extent, and they need certainty by making a theory practice, or they try to explain their results by building a hypothesis around the experiment. In the first scenario, the measured data set has to be fitted to a presumed relation between events, in other words, to an already existing function. This is a more likely approach, since experiments are very demanding, and without proper reasoning, they do not materialize. The second approach ends up having a bunch of data where scientists have to guess what kind of pattern they follow. The function that was at least assumed in the other case, is missing here, which complicates things.

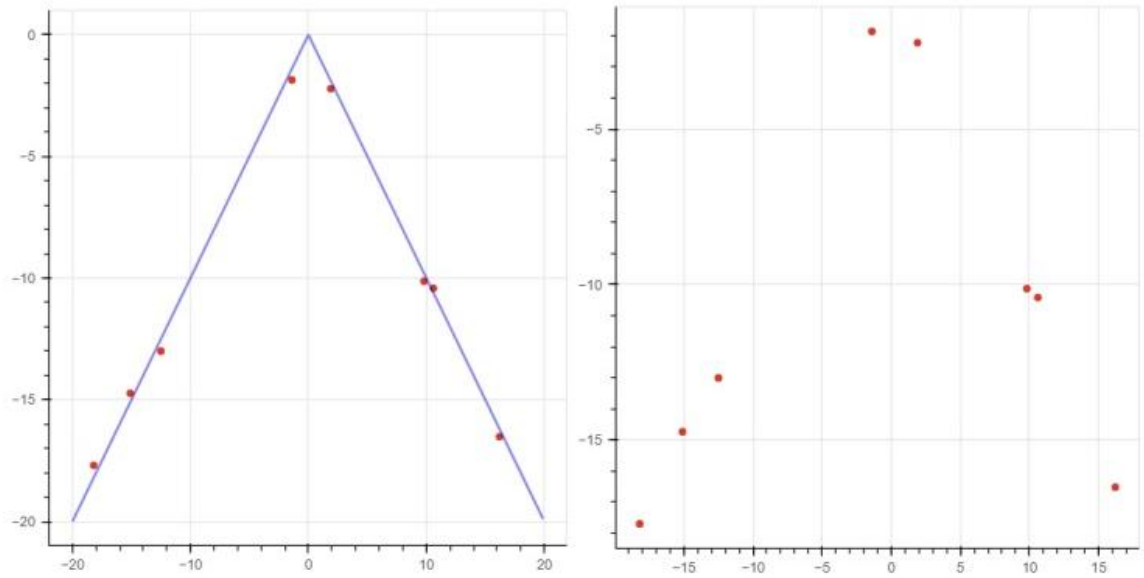


**Figure 1:** The two cases of experiments. The measured data points (red dots) are the same on both plots, but with the assumption of a pre-existing relation, it is easy to check the validity and correctness of the experiment via examination of the errors of the data set. After introducing the mathematical background, it turns to one specific error metric. The main goal is to find a good error metric to define the quality of a data set but on a 3D surface. Let us go back to Figure 1a. The dots are not perfect. They do not lie on the blue line which is caused by measurement errors. These errors in real practice can come from

equipment, environment, human limitations, or human negligence. In this current study, the origins of errors are irrelevant, and their discussions are omitted. What matters is that they are present, and their extension should be somehow qualified.

### 2.2 Error metrics

A way to measure the error of a model is the error metric. There are several types, such as **Mean Squared Error (MSE):** is probably the most well-known metric. It measures the mean of the squares of the deviations.

$$MSE = 1/n \Sigma (yi – y˜i) ^2$$

where n is the number of data points, yi is the measured value (or vector of measured values),and y˜i is the predicted value (vector of values). In later section, matrix notation will be used. In order to make comparison easier, let us write MSE as a product of matrices.

$$MSE = 1/n e^Te$$

where e is an n × 1 matrix with elements ei =yi - ˜yi. This metric is used in this assignment's programming task as well, when the ideal function for a specific train data set was determined. An ideal function was selected for a specific data set if it had the lowest MSE.

**Root Mean Square Error (RMSE):** is the square root of MSE.

$$RMSE = \sqrt{MSE}$$

The advantage of using this metric is its dimension. It is calculated in the same unit as the values are measured. It is more intuitive to calculate the average deviation than the average squared deviation.

**Mean Absolute Error (MAE):** averages out the absolute deviations.

$$MAE = 1/n \Sigma |yi − y˜i |$$

- Mean Absolute Scaled Error (MASE)
- Mean Absolute Percentage Error (MAPE)
- Symmetric Mean Absolute Percentage Error (SMAPE)

## 3. Measurements

### 3.1 In two dimensions

What all of these error metrics have in common is that their definitions are based on a distance measure. In the case of Figure 1a, the distance measured is the distance between a red dot (measured value) and the blue line at the same x-coordinate. Let us look at this from a different perspective. Let us consider a random curve that is the ideal result of a measurement.

- The curve that would be ideally measured without errors.
- A measured curve with some errors.
- A measured curve with higher errors.

The rougher the curve, the higher the errors that are made during measurement. Calculating an error metric on a rougher curve gives a higher result.

### 3.2 In three dimensions

Similarly, to a curve - that is a two-dimensional object -, a surface in three dimensions can also be the desired result of a measurement. Let us consider the following: the roughness of a brand-new aluminium plate is to be checked before it goes out of the factory. For the eye, a well-manufactured product of such is flat, but it does indeed have irregularities on its surface for a coloured plot of the result of such measurement. The error is related to the distance between the position of the new vertex and the faces connecting to that vertex. What if, however, a vertex is on a perfectly flat surface? In this case, it does not matter, where the vertex is positioned, since it would always be the closest possible to all faces: on them directly. With that, the matrix is unfortunately not invertible. If roughness is too much, the surface is to be smoothed out. The effort to put into the process would be higher with higher roughness which is, in fact, the error.

## 4. Quadric Error Metric

### 4.1 Modelling

What do all of these mean in practice? There are cases when it is clear what the outcome of a measurement is supposed to be. In this case, measured data must be fitted to a given curve/surface/etc. A good example of such use of this error measure is 3D modelling. In this field, people work with so-called meshes that represent an object in three-dimensional space. A common type of mesh is triangular. The error is related to the distance between the position of the new vertex and the faces connecting to that vertex. In this case, it does not matter, where the vertex is positioned, since it would always be the closest possible to all faces: on them directly. A real sphere is as smooth as no mesh can be. A triangle in a mesh of a ball is just an approximation of the real surface. It should be divided into infinitely many smaller triangles in order to have the same resolution as a mathematically perfect sphere. In this case, the difference between them is the error; but how can this error be qualified? Here comes the Quadric Error Metric (QEM) into the picture.

### 4.2 Application simplification

In 3D modelling, it has been a big struggle to find a way to simplify a mesh without losing too much of its integrity and resolution. The more vertices a mesh contains, the slower any computation is on it. Especially in real-time rendering, this can be unacceptable. So, what can one do? There are many ways to simplify a mesh. There exist many algorithms with all kinds of benefits of using them. Each one, however, has its own disadvantages as well. In this section, the already mentioned QEM will be explained in more detail. The algorithm using QEM is called Quadric Error Collapse

Decimation (QECD). It is based on contracting edges. At the beginning of the process, the required number of faces/edges has to be given, so that the algorithm knows how many iterations have to be done. It starts with calculating the error for all vertices and then for all edges using the faces connected to each vertex and edge. Then by going in ascending order - from lowest error to highest -, each edge is collapsed into one ideally positioned vertex. In order to find the ideal position of the new vertex, Δ has to be minimalised. In mathematics, that is differentiation. Since a surface is quadratic, the error function is also quadratic, therefore finding its minimum is linear.

#### 4.2.1 New ideal vertex position

The very first question could be: what does an ideal position really mean? A position is considered ideal to a vertex if it is the closest possible to all connecting faces at once. In the case of an edge, that consists of two vertices, the collapsed position is ideal if it is the closest possible to all faces touching the edge. Defining the cost of an edge contraction is calculating the error. Let us describe this error with a 4 × 4 matrix Q. To each vertex exists a symmetric Qv in a specific form. It can be derived from the faces connected to the vertex. There exist many algorithms with all kinds of benefits of using them. Each one, however, has its own disadvantages as well. In this section, the already mentioned QEM will be explained in more detail. Given a face described as a plane.

P : ax + by + cz + d = 0,
where a^2 + b^2 + c^2 = 1.

The distance between a given point in space v = (v1, v2, v3, 1) and plane P = (a, b, c, d) is

Pov = (av1, bv2, cv3, d)

It is considered best to use its square, since error may be negative this way. The new equation with matrix notation is the following:

(P^Tv)^2 = (P^Tv)^T(P^Tv) = (vP^T)(P^Tv) = v^T(PP^T)v
= v^TKpv

This gives the error metric matrix of v with respect to plane P. Calculating the error metric matrix of a given vertex is to sum up all its faces' error metric matrices.

Qv = Σ ∀P KP

For a given edge, the error metric matrix is the sum of its two vertices' error metric matrices.

Q = Q1 + Q2

The error metric of an edge collapse is therefore

Δ = v^TQv

In order to find the ideal position of the new vertex, Δ has to be minimalised. In mathematics, that is differentiation. Since a surface is quadratic, the error function is also quadratic, therefore finding its minimum is linear.

$$\partial\Delta/\partial x = \partial\Delta/\partial y = \partial\Delta/\partial z = 0$$

This, however, implies that Q has to be invertible. If this is not the case, another method is to be used.

#### 4.2.2 Noninvertible error metric matrix

Noninvertible error metric matrix in simpler mesh simplification algorithms, an edge is collapsed into one of its vertices or in the middle. This, however, might not be where the error is the smallest. In the case of a noninvertible QEM matrix, this might be the only solution. A common type of mesh is triangular. The error is related to the distance between the position of the new vertex and the faces connecting to that vertex. In this case, it does not matter, where the vertex is positioned, since it would always be the closest possible to all faces: on them directly. A real sphere is as smooth as no mesh can be. The process is quite similar, though, instead of calculating Q first, the ideal position (the positions of the two vertices and the middle of the edge) is calculated and from that the QEM matrix and the error itself. In this scenario, the position with the lowest QEM is to be the position of the new vertex after edge collapse.

#### 4.2.3 Result and Limitation

*Result*: - It is visible that more edge contractions were done on the sole than on the toes. It is quite predictable since the sole of a foot is relatively flat, meanwhile, toes are more rounded, so any change in those vertices would disrupt the integrity of the mesh more. In other words, changing the mesh in the toes will cost more, so the errors are also bigger than on the sole. The error is related to the distance between the position of the new vertex and the faces connecting to that vertex. What if, however, a vertex is on a perfectly flat surface? In this case, it does not matter, where the vertex is positioned, since it would always be the closest possible to all faces: on them directly. With that, the matrix is unfortunately not invertible.

*Limitation***: -** As it was mentioned in Section 4.2.2, there is a chance that the QEM matrix is not invertible. The question is: why can that be? Understanding the reason is really just understanding what actually the error is that is to be calculated. The error is related to the distance between the position of the new vertex and the faces connecting to that vertex. What if, however, a vertex is on a (fairly) perfectly flat surface? In this case, it does not matter, where the vertex is positioned, since it would always be the closest possible to all faces: on them directly. With that, the QEM matrix is unfortunately not invertible. That means, there is no error to be calculated. This is, however, little consolation when edges have to be contracted and the mesh does not become any simpler.

### 4.3 Further possibilities

There is another mesh simplification method that uses QEM as its base, the Discrete Differential Error Metric (DDEM). It is partially a weighted QEM, where each faceis weighted with its area. It also contains two more members, the Tangential Error Metric and the Discrete Curvature Error Metric.

### 4.4 Final words

Error calculation is extremely important. It gives feedback about the quality of the scientific procedure, whether it is an experiment or any other technique. Finding the right kind of error metric, however, can be challenging. In the case of

mesh simplification, losing even a small amount of accuracy might cause problems, for example,the incapability of 3D printing.

## References

[1] M. Botsch et al. - Geometric Modelling Based on Triangle Meshes, 2006, ACM DL 10.1145/1185657.1185839.
[2] M.Garland et al. – Surface Simplification Using Quadric Error Metrics, 1997, ACM DL 10.1145/258734.258849.
[3] https://graphics.stanford.edu/courses/cs468-10-fall/LectureSlides/08_ Simplification.pdf.
[4] S.Kim et al. - Discrete Differential Error Metric for Surface Simplification, 2002, IEEE 10.1109/PCCGA.2002.1167871.
[5] W.Griffith - The physics of everyday phenomena: a conceptual introduction to physics, 1992, ISBN 0072328371.
[6] F.Wilchzek - Fantastic realities, 2006, ISBN 9789812566492.
[7] https://manoa.hawaii.edu/exploringourfluidearth/physical/world-ocean/map-distortion/ practices -science-scientific-error.
[8] D.Wackerlyetal.-Mathematical Statistics with Application, 2008, ISBN 9780495385080.
[9] https://www.comsol.com/blogs/how-to-generate-random-surfaces-in-comsol-multiphysic/
[10] R. Tobler et al. - A Mesh Data Structure for Rendering and Subdivision, 2006, ISBN 8086943054.

### Appendix

#### 6.1 Python program code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
train_data=pd.read_csv("train.csv")
train_data.head()
```

```
test_data=pd.read_csv("test.csv")
test_data.head()
```

```
ideal_function=pd.read_csv("ideal.csv")
ideal_function.head()
```

```
# analyzing sidtribution of x
train_data['x'].hist(bins=50)
```

```
# analyzing training data
ax1 = train_data.plot(kind='scatter', x='x', y='y1', color='r')
ax2 = train_data.plot(kind='scatter', x='x', y='y2', color='g', ax=ax1)
ax3 = train_data.plot(kind='scatter', x='x', y='y3', color='b', ax=ax1)
ax4 = train_data.plot(kind='scatter', x='x', y='y4', color='y', ax=ax1)
print(ax1 == ax2 == ax3==ax4)
```

```
# analyzing distribution of y1
train_data['y1'].hist(bins=50)
```

```
# analyzing distribution of y2
train_data['y2'].hist(bins=50)
```

```
# analyzing distribution of y3
train_data['y3'].hist(bins=50)
```

```
# analyzing distribution of y4
train_data['y4'].hist(bins=50)
```

```
# analyzing testing  data
ax1 = test_data.plot(kind='scatter', x='x', y='y', color='r')
print(ax1)
```

```
## Defining function to calculate mean square error
def calculate_mean_sq_error(y,y_t):
    diff_sum = 0  #variable to store the summation of differences
    n = len(y) #finding total number of items in list
    for i in range (0,n):  #looping through each element of the list
      diff = y[i] - y_t[i]  #finding the difference between observed and predicted value
      squared_difference = diff**2  #taking square of the differene
      diff_sum = diff_sum + squared_difference  #taking a sum of all the differences
    MSE = diff_sum/n
    return MSE
```

```
#uses training data to choose the four ideal functions which are the
#best fit out of the fifty provided
```

```
def calculate_mean_sq_error_training(y_ideal):
    mean_er_y1= calculate_mean_sq_error(list(train_data['y1']),y_ideal)
    mean_er_y2= calculate_mean_sq_error(list(train_data['y2']),y_ideal)
    mean_er_y3= calculate_mean_sq_error(list(train_data['y3']),y_ideal)
    mean_er_y4= calculate_mean_sq_error(list(train_data['y4']),y_ideal)
    mean_error_training = (mean_er_y1 + mean_er_y2 + mean_er_y3 + mean_er_y4)/4
    return mean_error_training
```

```
ideal_fit_score=[]
for i in range(50):
    idle_y = 'y'+str(i+1)
    train_mean_score = calculate_mean_sq_error_training(list(ideal_function[idle_y]))
    ideal_fit_score.append(train_mean_score)
```

```
plt.hist(ideal_fit_score)
```

```
# get the four idle which has the lowest mean squared score which has the best fit
np.array(ideal_fit_score).argsort()[:4]
```

```
# for my given dataset y30,y32,y35, and y44 are the four ideal functions for which training data has the best fit
```