Nuclear Instruments and Methods in Physics Research A270 (1988) 110–117
North-Holland, Amsterdam

# HOW TO COMBINE CORRELATED ESTIMATES OF A SINGLE PHYSICAL QUANTITY

Louis LYONS and Duncan GIBAUT
*Nuclear Physics Laboratory, Keble Road, Oxford, UK*

Peter CLIFFORD
*Mathematical Institute, St. Giles, Oxford, UK*

Received 4 January 1988

Experiments to measure a single physical quantity often produce several estimates based on the same data, and which are hence correlated. We describe how to combine these correlated estimates in order to provide the best single answer, and also how to check whether the correlated estimates are mutually consistent.

We discuss the properties of our technique, and illustrate its application by using it for a specific experiment which measured the lifetime of charmed particles.

# Uncorrelated Errors

The standard method for combining different experiments consists in weighting each result $y_i \pm \sigma_i$ by a factor inversely proportional to that measurement's variance, i.e.

$$\hat{y} = \sum \left( y_i / \sigma_i^2 \right) \Big/ \sum \left( 1/\sigma_i^2 \right). \qquad (1)$$

The corresponding error $\sigma$ is given by

$$1/\sigma^2 = \sum \left( 1/\sigma_i^2 \right). \qquad (2)$$

This, however, applies only when the individual measurement errors are uncorrelated. In the situation we

# BLUE technique (Best Linear Unbiassed Estimate)

We employ a BLUE technique [1] (i.e., Best Linear Unbiassed Estimate) which consists in looking for an estimate $\hat{y}$ which

(1) is a linear combination of the individual estimates;
(2) provides an unbiassed estimate of $y$; and
(3) has the minimum possible variance $\sigma^2$.

The first condition is that

$$\hat{y} = \sum \alpha_i y_i , \tag{3}$$

where the $\alpha_i$ are constants, the weighting factors for the various estimates $y_i$, and which we want to determine.

For eq. (3) to be unbiassed, we require

$$\sum \alpha_i = 1; \tag{4}$$

this follows from our assumption that the $y_i$ are themselves unbiassed.

From eq. (3), we deduce the variance of $\hat{y}$ as

$$\sigma^2 = \tilde{\boldsymbol{\alpha}} \mathbf{E} \boldsymbol{\alpha} \tag{5}$$

where $\boldsymbol{\alpha}$ is the vector of the weighting factors $\alpha_i$, and $\tilde{\boldsymbol{\alpha}}$ is its transpose. In terms of components

$$\sigma^2 = \sum_i \sum_j E_{ij} \alpha_i \alpha_j . \tag{5'}$$

The BLUE technique consists simply of finding the $n$ values of $\alpha_i$ which minimise $\sigma^2$, subject to the constraint (4). This can be achieved by using eq. (4) to eliminate one particular $\alpha_i$ (e.g. $\alpha_1$) and then minimising $\sigma^2$ with respect to the remaining $(n-1)$ $\alpha$'s, regarded as independent. Alternatively, we can use the method of Lagrangian multipliers to give

$$\boldsymbol{\alpha} = \mathbf{E}^{-1} U / (\tilde{U} \mathbf{E}^{-1} U), \tag{6}$$

where $U$ is a vector whose $n$ components are all unity, and $\mathbf{E}^{-1}$ is the inverse error matrix.

These values of $\alpha_i$ can then be substituted back into eq. (3) to provide our best value $\hat{y}$, and into eq. (5) to obtain its variance.

# An example: the lifetime of D mesons

The results of the individual methods described in the previous section are as follows [7]

$$\tau_1 = \left(9.5^{+1.7}_{-1.2}\right) \times 10^{-13} \text{ s},$$
$$\tau_2 = \left(11.9^{+1.5}_{-1.3}\right) \times 10^{-13} \text{ s},$$
$$\tau_3 = \left(11.1^{+1.8}_{-1.2}\right) \times 10^{-13} \text{ s},$$
$$\tau_4 = \left(8.9^{+1.6}_{-1.2}\right) \times 10^{-13} \text{ s}. \qquad (14)$$

We are now going to combine these by the technique described in section 2, in order to produce our best estimate $\hat{\tau}$.

The results (14) contain the error estimates on the individual measurements but not their correlations, which arise from the fact that the same events contribute to the separate determinations. Thus if a particular $D^{\pm}$ lives for much longer than the average, it tends to raise each of the estimates to which it contributes. The correlation is not complete, however, for the following reasons:

(1) The data samples for the different determinations are not identical, because of the different selection criteria for the various methods.

(2) Different features of the $D^{\pm}$ are important for the separate methods. Thus $\tau_2$ depends almost entirely on the production characteristics of the $D^{+}$, and $\tau_3$ on its decay properties.

We have estimated the full error matrix by a Monte Carlo method. We have created a series of 100 "experiments", in which the $D^{\pm}$ are generated in accordance with the known production and decay properties, and are subjected to the same selection criteria as were used for the real data. We also ensured that the numbers of events available for each of the 4 analyses and the numbers which were common to each possible combination of methods were identical to those in the actual data. We then analysed each of these "experiments" in exactly the same way as described in section 4 for the real data. In this way we obtained 100 sets of four lifetime estimates $\tau_{ik}$, where the subscript $k$ denotes the Monte Carlo experiment number and $i$ refers to the method used to determine $\tau$ (i.e. $i$ goes from 1 to 4, and $k$ from 1 to 100).

We then calculate the elements of the error matrix **E** as

$$E_{ij} = \frac{1}{100} \sum_{k=1}^{100} (\tau_{ik} - \bar{\tau}_i)(\tau_{jk} - \bar{\tau}_j), \qquad (15)$$

where $\bar{\tau}_i$ is the Monte Carlo average for the $i$th method, i.e.

$$\bar{\tau}_i = \frac{1}{100} \sum_{k=1}^{100} \tau_{ik}. \qquad (16)$$

The above procedure provides us with the following error matrix

$$\mathbf{E} = \begin{pmatrix} 2.66 & 1.15 & 0.86 & 1.31 \\ 1.15 & 1.45 & 0.82 & 1.32 \\ 0.86 & 0.82 & 1.06 & 1.05 \\ 1.31 & 1.32 & 1.05 & 2.56 \end{pmatrix}, \tag{17}$$

Our corrective procedure is thus simply to leave unchanged the off-diagonal elements of $\mathbf{E}$, but to increase the diagonal elements to be equal to the square of the errors as given in (1) above. We thus obtain

$$\mathbf{E} = \begin{pmatrix} 2.74 & 1.15 & 0.86 & 1.31 \\ 1.15 & 1.67 & 0.82 & 1.32 \\ 0.86 & 0.82 & 2.12 & 1.05 \\ 1.31 & 1.32 & 1.05 & 2.93 \end{pmatrix}. \tag{17'}$$

This error matrix determines the weights of the various estimates as

$$\alpha_1 = 0.14,$$

$$\alpha_2 = 0.47,$$

$$\alpha_3 = 0.35,$$

$$\alpha_4 = 0.04. \tag{18}$$

Together with the individual results (14), these give a best estimate

$$\tau = (11.2 \pm 1.1) \times^{-13} \text{ s}. \tag{19}$$

```python
measurements = [9.5, 11.9, 11.1, 8.9]
covariance_matrix = [
    [2.74, 1.15, 0.86, 1.31],
    [1.15, 1.67, 0.82, 1.32],
    [0.86, 0.82, 2.12, 1.05],
    [1.31, 1.32, 1.05, 2.93]
]

x_hat, sigma_x_hat, weights = blue_combination(measurements, covariance_matrix)

print(f"Combined estimate: {x_hat:.3f} ± {sigma_x_hat:.3f}")
print(f"BLUE weights: {weights}")
```

```
PS C:\Users\Wen-Chen Chang\SynologyDrive\python_working> & "C
error_lyons.py"\
Combined estimate: 11.160 ± 1.134
BLUE weights: [0.14507476 0.46957738 0.34729705 0.03805081]
```

The BLUE technique consists simply of finding the $n$ values of $\alpha_i$ which minimise $\sigma^2$, subject to the constraint (4). This can be achieved by using eq. (4) to eliminate one particular $\alpha_i$ (e.g. $\alpha_1$) and then minimising $\sigma^2$ with respect to the remaining $(n-1)$ $\alpha$'s, regarded as independent. Alternatively, we can use the method of Lagrangian multipliers to give

$$\alpha = \mathbf{E}^{-1}U/(\tilde{U}\mathbf{E}^{-1}U), \qquad (6)$$

where $U$ is a vector whose $n$ components are all unity, and $\mathbf{E}^{-1}$ is the inverse error matrix.

These values of $\alpha_i$ can then be substituted back into eq. (3) to provide our best value $\hat{y}$, and into eq. (5) to obtain its variance.

```python
import numpy as np

def blue_combination(measurements, covariance_matrix):
    """
    Combine measurements using the BLUE method.

    Parameters:
    - measurements: array-like, shape (N,)
    - covariance_matrix: array-like, shape (N, N)

    Returns:
    - x_hat: float, combined estimate
    - sigma_x_hat: float, uncertainty of the combined estimate
    - weights: array-like, the BLUE weights for each measurement
    """
    x = np.array(measurements)
    V = np.array(covariance_matrix)
    V_inv = np.linalg.inv(V)

    ones = np.ones(len(x))

    # Compute weights
    denominator = ones @ V_inv @ ones
    weights = (V_inv @ ones) / denominator

    # Compute combined estimate and its variance
    x_hat = weights @ x
    sigma_x_hat = np.sqrt(1.0 / denominator)

    return x_hat, sigma_x_hat, weights
```

We employ a BLUE technique [1] (i.e., Best Linear Unbiassed Estimate) which consists in looking for an estimate $\hat{y}$ which

(1) is a linear combination of the individual estimates;
(2) provides an unbiassed estimate of $y$; and
(3) has the minimum possible variance $\sigma^2$.

The first condition is that

$$\hat{y} = \sum \alpha_i y_i, \tag{3}$$

where the $\alpha_i$ are constants, the weighting factors for the various estimates $y_i$, and which we want to determine.

For eq. (3) to be unbiassed, we require

$$\sum \alpha_i = 1; \tag{4}$$

this follows from our assumption that the $y_i$ are themselves unbiassed.

From eq. (3), we deduce the variance of $\hat{y}$ as

$$\sigma^2 = \tilde{\alpha} \mathbf{E} \alpha \tag{5}$$

where $\alpha$ is the vector of the weighting factors $\alpha_i$, and $\tilde{\alpha}$ is its transpose. In terms of components

$$\sigma^2 = \sum_i \sum_j E_{ij} \alpha_i \alpha_j. \tag{5'}$$

The BLUE technique consists simply of finding the $n$ values of $\alpha_i$ which minimise $\sigma^2$, subject to the constraint (4). This can be achieved by using eq. (4) to eliminate one particular $\alpha_i$ (e.g. $\alpha_1$) and then minimising $\sigma^2$ with respect to the remaining $(n-1)$ $\alpha$'s, regarded as independent. Alternatively, we can use the method of Lagrangian multipliers to give

$$\alpha = \mathbf{E}^{-1} U / (\tilde{U} \mathbf{E}^{-1} U), \tag{6}$$

where $U$ is a vector whose $n$ components are all unity, and $\mathbf{E}^{-1}$ is the inverse error matrix.

These values of $\alpha_i$ can then be substituted back into eq. (3) to provide our best value $\hat{y}$, and into eq. (5) to obtain its variance.

Given:

- Measurements $\vec{x} = (x_1, x_2, \ldots, x_N)$
- Covariance matrix $V$
- Weight vector $\vec{w}$, where

$$\vec{w} = \frac{V^{-1} \cdot \vec{1}}{\vec{1}^\top V^{-1} \vec{1}}$$

Then the combined estimate is:

$$\hat{x} = \vec{w}^\top \vec{x}$$

And its **variance** (i.e. squared uncertainty) is:

$$\sigma_{\hat{x}}^2 = \vec{w}^\top V \vec{w}$$

Given:

- Measurements $\vec{x} = (x_1, x_2, \ldots, x_N)$
- Covariance matrix $V$
- Weight vector $\vec{w}$, where

$$\vec{w} = \frac{V^{-1} \cdot \vec{1}}{\vec{1}^\top V^{-1} \vec{1}}$$

Then the combined estimate is:

$$\hat{x} = \vec{w}^\top \vec{x}$$

And its **variance** (i.e. squared uncertainty) is:

$$\sigma_{\hat{x}}^2 = \vec{w}^\top V \vec{w}$$

### 🧠 Alternative Expression (Without Explicit Weights)

You can also directly compute the variance using:

$$\sigma_{\hat{x}}^2 = \frac{1}{\vec{1}^\top V^{-1} \vec{1}}$$

This is what we used earlier in code. It's mathematically equivalent to $\vec{w}^\top V \vec{w}$, just more efficient in practice.

```python
import numpy as np

def blue_combination(measurements, covariance_matrix):
    """
    Combine measurements using the BLUE method.

    Parameters:
    - measurements: array-like, shape (N,)
    - covariance_matrix: array-like, shape (N, N)

    Returns:
    - x_hat: float, combined estimate
    - sigma_x_hat: float, uncertainty of the combined estimate
    - weights: array-like, the BLUE weights for each measurement
    """
    x = np.array(measurements)
    V = np.array(covariance_matrix)
    V_inv = np.linalg.inv(V)

    ones = np.ones(len(x))

    # Compute weights
    denominator = ones @ V_inv @ ones
    weights = (V_inv @ ones) / denominator

    # Compute combined estimate and its variance
    x_hat = weights @ x
    sigma_x_hat = np.sqrt(1.0 / denominator)

    return x_hat, sigma_x_hat, weights
```

Let:

- $V$ be the covariance matrix (NxN, symmetric and positive-definite),
- $\vec{1}$ be a column vector of ones (shape Nx1),
- $V^{-1}$ be the inverse of the covariance matrix.

---

## Expression 1:

Variance via the **weight vector** $\vec{w}$:

$$\sigma_{\hat{x}}^2 = \vec{w}^\top V \vec{w}$$

---

## Expression 2:

Direct variance formula (used in code):

$$\sigma_{\hat{x}}^2 = \frac{1}{\vec{1}^\top V^{-1} \vec{1}}$$

---

## ✅ Goal: Show that both are equal

**Step 1: Write BLUE weights**

$$\vec{w} = \frac{V^{-1}\vec{1}}{\vec{1}^\top V^{-1}\vec{1}}$$

---

**Step 2: Plug weights into Expression 1**

$$\sigma_{\hat{x}}^2 = \vec{w}^\top V \vec{w} = \left(\frac{V^{-1}\vec{1}}{\vec{1}^\top V^{-1}\vec{1}}\right)^\top V \left(\frac{V^{-1}\vec{1}}{\vec{1}^\top V^{-1}\vec{1}}\right)$$

**Step 3: Simplify**

Let's denote $d = \vec{1}^\top V^{-1}\vec{1}$. Then:

$$\sigma_{\hat{x}}^2 = \frac{1}{d^2} \cdot \left(\vec{1}^\top V^{-1}\right) V \left(V^{-1}\vec{1}\right)$$

Now use associativity:

$$\sigma_{\hat{x}}^2 = \frac{1}{d^2} \cdot \vec{1}^\top V^{-1} V V^{-1} \vec{1}$$

Since $V^{-1}V = I$:

$$\sigma_{\hat{x}}^2 = \frac{1}{d^2} \cdot \vec{1}^\top V^{-1}\vec{1} = \frac{d}{d^2} = \frac{1}{d}$$

---

## ✅ Conclusion:

$$\boxed{\vec{w}^\top V \vec{w} = \frac{1}{\vec{1}^\top V^{-1}\vec{1}}}$$

Hence, **both expressions for the variance are mathematically equivalent** — one is more intuitive (via weights), the other is computationally compact.