Peak Estimator Design for Detector/Front-end Signals
Summary

• Peak estimator design
• Cost function visualization
• Lagrange multiplier review
• Example based on simulation
Peak Estimator Design
Signal Modeling

- Equation 1 describes the digitized signal $x[k]$ acquired at the front-end output. $h(t_k)$ is the signal and $n(t_k)$ is the additive noise.

- The signal $h(t_k)$ might be described in terms of its peak amplitude $A$ and the its shape normalized by the peak $g(t_k)$.

$$x[k] = h(t_k) + n(t_k) = Ag(t_k) + n(t_k)$$
Signal Modeling

- Equation 1 describes the digitized signal $x[k]$ acquired at the front-end output. $h(t_k)$ is the signal and $n(t_k)$ is the additive noise.

- The signal $h(t_k)$ might be described in terms of its peak amplitude $A$ and the its shape normalized by the peak $g(t_k)$.

$$x[k] = h(t_k) + n(t_k) = Ag(t_k) + n(t_k)$$

Therefore we are supposing the signal shape is known.
Peak Estimator Design

• Now we can propose a simple linear equation to be used as a peak amplitude estimator

\[ \hat{A} = \sum_{k=1}^{N} a[k]x[k] \]

\[ \hat{A} = A \sum_{k=1}^{N} a[k]g[k] + \sum_{k=1}^{N} a[k]n[k] \]

• To make this equation an estimator of \( A \) we should have the first summation equal to ONE and the second equal to ZERO

• In terms of its Expected Value, if we want an unbiased estimator

\[ E[\hat{A}] = A \]

\[ E[\hat{A}] = A \sum_{k=1}^{N} a[k]g[k] + \sum_{k=1}^{N} a[k]E[n[k]] \]
Peak Estimator Design

• Now we can propose a simple linear equation to be used as a peak amplitude estimator

\[ \hat{A} = \sum_{k=1}^{N} a[k] x[k] \]

\[ \hat{A} = A \sum_{k=1}^{N} a[k] g[k] + \sum_{k=1}^{N} a[k] n[k] \]

• To make this equation an estimator of \( A \) we should have the first summation equal to ONE and the second equal to ZERO

• In terms of its Expected Value, if we want an unbiased estimator

\[ E[\hat{A}] = A \]

\[ E[\hat{A}] = A \sum_{k=1}^{N} a[k] g[k] + \sum_{k=1}^{N} a[k] E[n[k]] \]

\[ = 1 \]

\[ = 0 \]
Peak Estimator Design

• Therefore TWO restriction are necessary

\[ E[\hat{A}] = A \sum_{k=1}^{N} a[k]g[k] + \sum_{k=1}^{N} a[k]E[n[k]] \]

\[ \sum_{k=1}^{N} a[k]g[k] = 1 \quad \sum_{k=1}^{N} a[k] = 0 \]
Peak Estimator Design

• Therefore two restrictions are necessary

\[
E[\hat{A}] = A \sum_{k=1}^{N} a[k] g[k] + \sum_{k=1}^{N} a[k] E[n[k]]
\]

\[
\sum_{k=1}^{N} a[k] g[k] = 1 \quad \sum_{k=1}^{N} a[k] = 0
\]

Our estimator is defined....

\[
\hat{A} = \sum_{k=1}^{N} a[k] x[k]
\]

Next step would be finding the coefficients \(a[k], k=1,2,3,...N\).
Peak Estimator Design

- The coefficients $a[k]$ will be found minimizing the estimator variance

$$Var[\hat{A}] = A \sum_{k=1}^{N} a[k]g[k] + \sum_{k=1}^{N} a[k]n[k]$$

$$Var[\hat{A}] = Var\left[ \sum_{k=1}^{N} a[k]n[k] \right]$$

$$Var[\hat{A}] = \sum_{i=1}^{N} \sum_{j=1}^{N} a[i]a[j]C_{ij}$$

Cost function to be minimized

Not forgetting both restrictions...

$$\sum_{k=1}^{N} a[k]g[k] = 1 \quad \sum_{k=1}^{N} a[k] = 0$$
Visualizing the cost function
Visualizing the cost function

• The model leads us to the following estimator variance

\[ \text{Var} [\hat{A}] = \text{Var} [a^T \mathbf{n}] = a^T \mathbf{C} a \]

  – \( \mathbf{C} \) is the covariance matrix and \( a \) the filter weights

• Our goal is to minimize the estimator variance
  – Let's suppose there are two weights only...

\[ \text{Var}[\hat{A}] = [a_1 \ a_2] \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} [a_1] \]

\[ \text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \]

*Function to be minimized... Find \( a_i \) for \( \min \text{Var} [\hat{A}] \)
Visualizing the cost function

- Lets suppose $C = I$

\[
Var[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \quad \Rightarrow \quad Var[\hat{A}] = a_1^2 + a_2^2
\]

Function to be minimized...

```matlab
a1 = -1:0.01:1;
a2 = -1:0.01:1;
C = [1 0; 0 1];

for i=1:length(a1)
    for j=1:length(a2)
        a = [a1(i);a2(j)];
        Var(i,j) = a'*C*a;
    end
end

figure;
hold on;
[X Y] = meshgrid(a1,a2);
surf(X, Y, Var');
contour(X, Y, Var');
axis([min(a1) max(a1) min(a2) max(a2)]);
xlabel('a1')
ylabel('a2')
zlabel('Var[\hat{A}]','interpreter','latex');
```
Visualizing the cost function

- Let's suppose $C = \text{Diagonal}$

$$Var[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \quad \Rightarrow \quad Var[\hat{A}] = 5a_1^2 + a_2^2$$

```matlab
a1 = -1:0.01:1;
a2 = -1:0.01:1;
C = [5 0; 0 1];

for i=1:length(a1)
    for j=1:length(a2)
        a = [a1(i);a2(j)];
        Var(i,j) = a'*C*a;
    end
end

figure;
hold on;
[X Y] = meshgrid(a1,a2);
surf(X, Y, Var);
contour(X, Y, Var);
axis([min(a1) max(a1) min(a2) max(a2)]);
xlabel('a1')
ylabel('a2')
zlabel('Var[\hat{A}]','interpreter','latex');
```

*Function to be minimized...*
Visualizing the cost function

• Lets suppose \( C \neq I \)

\[
\text{Var}[\hat{A}] = a_1^2C_{11}a_2C_{12} + a_1C_{21}a_2^2C_{22}
\]

\[
\text{Var}[\hat{A}] = 2a_1^2a_2 + a_10.1a_2^2
\]

Function to be minimized...

```matlab
a1 = -1:0.01:1;
a2 = -1:0.01:1;
C = [2 1; 1 0.1];

for i=1:length(a1)
    for j=1:length(a2)
        a = [a1(i);a2(j)];
        Var(i,j) = a'*C*a;
    end
end

figure;
hold on;
[X Y] = meshgrid(a1,a2);
surf(X, Y, Var');
contour(X, Y, Var');
axis([min(a1) max(a1) min(a2) max(a2)]);
xlabel('a1')
ylabel('a2')
zlabel('\text{Var[}\hat{A}\text{]}', 'interpreter', 'latex');
```
Visualizing the cost function

• The model lead us to the following estimator variance

\[ \text{Var}[\hat{A}] = \text{Var}[a^T n] = a^T Ca \]

– \( C \) is the covariance matrix and \( a \) the filter weights

• Our goal is to minimize the estimator variance

\[
\text{Var}[\hat{A}] = [a_1 \ a_2 \ a_3 \ \ldots \ a_n] \begin{bmatrix} C_{11} & \cdots & C_{1n} \\ \vdots & \ddots & \vdots \\ C_{n1} & \cdots & C_{nn} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{bmatrix}
\]

\[
\text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} a_3 C_{13} \ldots a_n C_{1n} + a_1 C_{21} a_2^2 C_{22} a_3 C_{23} \ldots a_n C_{2n} + \ldots
\]

\text{Function to be minimized...}
\text{Find } a_i \text{ for } \min \text{Var}[\hat{A}]
Visualizing the cost function

• Now applying restriction...
  
  \[- \mathbf{C} \neq \mathbf{I} \rightarrow \text{Var}[\hat{\mathbf{A}}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \quad \rightarrow \quad \text{Var}[\hat{\mathbf{A}}] = 2a_1^2 + a_2^2\]
  
  – Lets suppose the following restriction: \[a_1 + a_2 = 1\]

```matlab
clear all; close all
a1 = -1:0.01:1;
a2 = -1:0.01:1;
C = [2 0; 0 1];

for i=1:length(a1)
  for j=1:length(a2)
    a = [a1(i);a2(j)];
    Var(i,j) = a'*C*a;
  end
end
surf(a1, a2, Var);
hold on;
[X,Y] = meshgrid(a1,a2);
contour(a1, a2, X.^2+Y.^2);
plot(a1, 1-a1, '--r');
axis([min(a1) max(a1) min(a2) max(a2)]);
xlabel('a1')
ylabel('a2')
zlabel('\text{Var}[\hat{\mathbf{A}}]', 'interpreter','latex');
```

\[a_1 + a_2 - 1 = 0\]
Visualizing the cost function

• Now applying restriction...
  - \( C \neq I \) \( \Rightarrow \) \( \text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \) \( \Rightarrow \) \( \text{Var}[\hat{A}] = 2a_1^2 + a_2^2 \)
  - Lets suppose the following restriction: \( a_1 + a_2 = 1 \)
Visualizing the cost function

• Now applying restriction...
  – $C \neq I \rightarrow Var[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22}$
  – $Var[\hat{A}] = 2a_1^2 + a_2^2$
  – Lets suppose the following restriction: $a_1 + a_2 = 1$
  – Now a second restriction: $a_1 - a_2 = 0$

$\begin{align*}
  a_1 + a_2 &= 1 \\
  a_1 - a_2 &= 0
\end{align*}$

plot(a1, 1-a1, 'r');
plot(a1, a1, 'r');
Visualizing the cost function

- Now applying restriction...
  - $C \neq I \Rightarrow Var[\hat{A}] = a_1^2 C_{11} + a_2 C_{12} + a_1 C_{21} + a_2 C_{22} = 2a_1 + a_2^2$
  - Let's suppose the following restriction:
  - Now a second restriction: $a_1 - a_2 = 0$

Lagrange multiplier technique is used for minimization with constrains

```matlab
plot(a1, 1-a1, '--r');
plot(a1, a1, '--r');
```
Lagrange multiplier review
Lagrange multiplier review

Candidates for absolute maximum and minimum of $f(x, y)$ subjected to the constraint $g(x, y)$ are the points on $g(x, y) = 0$ where gradients of $f(x, y)$ and $g(x, y)$ are parallel.

To solve for these points, we find all $x$, $y$ and $\lambda$ such that $\nabla f(x, y) = \lambda \nabla g(x, y)$ and $g(x, y)$ hold simultaneously.

Lagrange equation is given by $L(x, y, \lambda) = f(x, y) - \lambda g(x, y)$, where $\nabla L(x, y, \lambda) = 0$. 

Illustration of the constrained optimization problem
Lagrange multiplier review

- **C ≠ I**  \[ \Rightarrow \]  \[ \text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \]  \[ \Rightarrow \]  \[ \text{Var}[\hat{A}] = 5 a_1^2 + a_2^2 \]

- Lets suppose the following restriction:  \[ a_1 + a_2 = 1 \]

```matlab
a1 = -1:0.01:1;
a2 = -1:0.01:1;
C = [5 0; 0 1];
for i=1:length(a1)
    for j=1:length(a2)
        a = [a1(i);a2(j)];
        Var(i,j) = a'*C*a;
    end
end
figure;
hold on;
[X Y] = meshgrid(a1,a2);
contour(X, Y, Var');
plot(a1, 1-a1, '--r');
axis([min(a1) max(a1) min(a2) max(a2)]);
xlabel('a1')
ylabel('a2')
zlabel('Var[\hat{A}]', 'interpreter', 'latex');
```
Lagrange multiplier review

- **C ≠ I** \(\rightarrow\) \(\text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22}\) \(\rightarrow\) \(\text{Var}[\hat{A}] = 5a_1^2 + a_2^2\)

- Let's suppose the following restriction: \(a_1 + a_2 = 1\)

\[\nabla(5a_1^2 + a_2^2) = \lambda \nabla (a_1 + a_2 - 1)\]

- \(10a_1 = \lambda\)
- \(2a_2 = \lambda\)
- \(a_1 + a_2 = 1\)

\[
\begin{bmatrix}
10 & 0 & -1 \\
0 & 2 & -1 \\
1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda
\end{bmatrix}
=
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]

\[
\begin{bmatrix}
2C_{11} & C_{12} & -1 \\
C_{21} & 2C_{22} & -1 \\
1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda
\end{bmatrix}
=
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]

\[
\begin{bmatrix}
10 \\
2 \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2
\end{bmatrix}
=
\begin{bmatrix}
\lambda \\
\lambda
\end{bmatrix}
\]

\[
\begin{bmatrix}
\lambda \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2
\end{bmatrix}
=
\begin{bmatrix}
1 \\
0
\end{bmatrix}
\]

\[
\begin{bmatrix}
10 \\
2 \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda
\end{bmatrix}
=
\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]

\[
\begin{bmatrix}
10 \\
2 \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2
\end{bmatrix}
=
\begin{bmatrix}
\lambda \\
\lambda
\end{bmatrix}
\]

\[
\begin{bmatrix}
\lambda \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2
\end{bmatrix}
=
\begin{bmatrix}
1 \\
0
\end{bmatrix}
\]

\[
\begin{bmatrix}
\lambda \\
1
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2
\end{bmatrix}
=
\begin{bmatrix}
1 \\
0
\end{bmatrix}
\]

```matlab
>> M = [10 0 -1; 0 2 -1; 1 1 0];
>> R = [0; 0; 1];
>> A = inv(M)*R
A =
 0.1667
 0.8333
 1.6667
>> plot(A(1),A(2),'.r')
```

\[
\begin{bmatrix}
\frac{1}{6} \\
\frac{5}{6}
\end{bmatrix}
\]

\[
\text{MIN}
\]

\[
\text{plot}(1/6, 5/6, 'r*')
\]
Lagrange multiplier review

- **C ≠ I** $\rightarrow$ $Var[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22}$ $\rightarrow$ $Var[\hat{A}] = 5a_1^2 + a_2^2$

- Let's suppose the following restriction: $a_1 + a_2 = 1$

$\nabla (5a_1^2 + a_2^2) = \lambda \nabla (a_1 + a_2 - 1)$

\[
\begin{align*}
10a_1 &= \lambda \\
2a_2 &= \lambda \\
a_1 + a_2 &= 1
\end{align*}
\]

Lagrange expression...

$L(a_1, a_2, \lambda) = (5a_1^2 + a_2^2) - \lambda (a_1 + a_2 - 1)$

$\nabla L(a_1, a_2, \lambda) = 0$
Lagrange multiplier review

• $C \neq I \rightarrow \text{Var}[\hat{A}] = a_1^2 c_{11} a_2 c_{12} + a_1 c_{21} a_2^2 c_{22} \quad \Rightarrow \quad \text{Var}[\hat{A}] = 5a_1^2 + a_2^2$

• Lets suppose the following restriction: $a_1 + a_2 = 1$

• And a new restriction: $a_1 = a_2$

$\nabla(5a_1^2+a_2^2) = \lambda \nabla(a_1+a_2-1)+k \nabla(a_1-a_2)$
Lagrange multiplier review

- \( C \neq I \) → \( \text{Var}[\hat{A}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} \)  \( \text{Var}[\hat{A}] = 5a_1^2 + a_2^2 \)

- Let's suppose the following restriction: \( a_1 + a_2 = 1 \)

- And a new restriction: \( a_1 = a_2 \)

\[ \nabla (5a_1^2 + a_2^2) = \lambda \nabla (a_1 + a_2 - 1) + k \nabla (a_1 - a_2) \]

\[
\begin{bmatrix}
10 & 0 & -1 & -1 \\
0 & 2 & -1 & 1 \\
1 & 1 & 0 & 0 \\
1 & -1 & 0 & 0 
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k 
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
0 \\
0 
\end{bmatrix}
\]

>> M = [10 0 -1; 0 2 -1 +1; 1 1 0 0; 1 -1 0 0];
>> R = [0; 0; 1; 0];
>> A = inv(M)*R
A =
 0.5000
 0.5000
 3.0000
 2.0000
>> plot(A(1), A(2), 'r*');
Lagrange multiplier review

• \( C \neq I \) \( \Rightarrow \) \( Var[\hat{A}] = a_1^2 c_{11} a_2 c_{12} + a_1 c_{21} a_2^2 c_{22} \) \( \Rightarrow \) \( Var[\hat{A}] = 5a_1^2 + a_2^2 \)

• Lets suppose the following restriction: \( a_1 + a_2 = 1 \)

• And a new restriction: \( a_1 = a_2 \)

\[
\nabla(5a_1^2 + a_2^2) = \lambda \nabla(a_1 + a_2 - 1) + k \nabla(a_1 - a_2)
\]

\[
\begin{bmatrix}
10 & 0 & -1 & -1 \\
0 & 2 & -1 & 1 \\
1 & 1 & 0 & 0 \\
1 & -1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k
\end{bmatrix} =
\begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

\[
\begin{bmatrix}
2c_{11} & c_{12} & -1 & -1 \\
c_{21} & 2c_{22} & -1 & 1 \\
1 & 1 & 0 & 0 \\
1 & -1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k
\end{bmatrix} =
\begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix}
\]

\[
\begin{align*}
10a_1 &= \lambda + k \\
2a_2 &= \lambda - k \\
a_1 + a_2 &= 1 \\
a_1 - a_2 &= 0
\end{align*}
\]

>> M = [10 0 1 -1; 0 2 -1 1; 1 1 0 0; 1 -1 0 0];
>> R = [0; 0; 1; 0];
>> A = inv(M)*R
A =
0.5000
0.5000
3.0000
2.0000

>> plot(A(1), A(2), 'r*');
Lagrange multiplier review

• \( \mathbf{C} \neq \mathbf{I} \rightarrow \text{Var}[\hat{A}] = a_1^2C_{11}a_2C_{12} + a_1C_{21}a_2^2C_{22} \rightarrow \text{Var}[\hat{A}] = 5a_1^2 + a_2^2 \)

• Lets suppose the following restriction: \( a_1 + a_2 = 1 \)

• And a new restriction: \( a_1 = a_2 \)

\[
\nabla(5a_1^2 + a_2^2) = \lambda \nabla(a_1 + a_2 - 1) + k \nabla(a_1 - a_2)
\]

\[
\begin{align*}
10a_1 &= \lambda + k \\
2a_2 &= \lambda - k \\
a_1 + a_2 &= 1 \\
a_1 - a_2 &= 0
\end{align*}
\]

\[
\text{Lagrange expression...}
\]

\[
L(a_1, a_2, \lambda, k) = (5a_1^2 + a_2^2) - \lambda(a_1 + a_2 - 1) - k((a_1 - a_2)
\]

\[
\nabla L(a_1, a_2, \lambda, k) = 0
\]

\[
\text{plot}(1/2, 1/2, 'r*')
\]

\[
\text{MIN}
\]
Lagrange multiplier review

- Previous example

\[ C_{12} = C_{21} = 0 \]

\[ \text{Var}[^\hat{\mathbf{A}}] = a_1^2 C_{11} a_2 C_{12} + a_1 C_{21} a_2^2 C_{22} = 5a_1^2 + a_2^2 \]

\[ \nabla (5a_1^2 + a_2^2) = \lambda \nabla (a_1 + a_2 - 1) + k \nabla (a_1 - a_2) \]

\[
\begin{align*}
10a_1 &= \lambda + k \\
2a_2 &= \lambda - k \\
a_1 + a_2 &= 1 \\
a_1 - a_2 &= 0
\end{align*}
\]

\[
\begin{bmatrix}
2C_{11} & C_{12} & -1 & -1 \\
C_{21} & 2C_{22} & -1 & 1 \\
1 & 1 & 0 & 0 \\
1 & -1 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
0 \\
0 \\
0
\end{bmatrix}
\]

- Generalization....

\[
\begin{align*}
f_{c1} &= a_1 + a_2 - 1 \\
f_{c2} &= a_1 - a_2
\end{align*}
\]

\[
\begin{align*}
a_1 + a_2 &= 1 \\
a_1 - a_2 &= 0
\end{align*}
\]

Considering linear equations...
• Considering known $C$

$$C = [5 \ 0; \ 0 \ 1]$$; %noise covariance matrix

$$f_{c1} = a_1 + a_2 - 1$$
$$f_{c2} = a_1 - a_2$$

MATLAB code

```matlab
%DEFINE CONSTRAINT FUNCTIONS
Ncoef = length(diag(C)); %number of dimensions/coeffs (a1, a2, ...)
a = sym('a', [1 Ncoef]);
Ncons = 2; %number of constraint functions
fc1 = a(1)+a(2)-1; %constraint function 1
fc2 = a(1)-a(2); %constraint function 2

% BUILD-UP MATRIX AND VECTOR TO FIND COEFFs.

% CONSTRAINTS FUNCTIONS PART
for j = 1:Ncons
    s = sprintf('fc%d', j);
    AA = diff(eval(s),a(1));
    for i = 2:Ncoef
        AA = [AA diff(eval(s),a(i))];
    end
    if j==1 Mconst = AA;
    else Mconst = [Mconst; AA]; end
end
Mconst = double(Mconst); %transform symbolic to numeric

% FINAL MATRIX
M = [C -Mconst; Mconst zeros(Ncons, Ncons)]; %build up final matrix

% OBTAIN COEFFs.
A = inv(M)*vector
```
Lagrange multiplier review

\[ \begin{align*}
   f_{c1} &= a_1 + a_2 - 1 \\
   f_{c2} &= a_1 - a_2
\end{align*} \]

**MATLAB code**

%DEFINE CONSTRAINT FUNCTIONS
Ncoef = length(diag(C)); %number of dimensions/coffs (a1, a2, ...)
a = sym('a', [1 Ncoef]);
Ncons = 2; %number of constraint functions
fc1 = a(1)+a(2)-1; %constraint function 1
fc2 = a(1)-a(2); %constraint function 2

% BUILD-UP MATRIX AND VECTOR TO FIND COEFFs.
% %%%%%%%% LEFT-HAND MATRIX
% % COVARIANCE PART
C = C*diag(2*ones(1, length(diag(C)))); %covariane matrix but diag*2

% CONSTRAINTS FUNCTIONS PART
for j = 1:Ncons
   s = sprintf('fc%d', j);
   AA = diff(eval(s),a(1));
   for i = 2:Ncoef
      AA = [AA diff(eval(s),a(i))];
   end
   if j==1 Mconst = AA;
   else Mconst = [Mconst; AA]; end
end
Mconst = double(Mconst); %transform symbolic to numeric

% FINAL MATRIX
M = [C -Mconst; Mconst zeros(Ncons, Ncons)]; %build up final matrix

% %%%%%%%% RIGHT-HAND VECTOR
for i = 1:Ncoef
   eval(sprintf('a%d=0', i));
end
vector = [0 0 -subs(fc1) -subs(fc2)];

% OBTAIN COEFFs.
A = inv(M)*vector

Considering known \( C \)
\( C = [5 0; 0 1]; \) %noise covariance matrix

\[
\begin{array}{ccc}
   2C_{11} & C_{12} & -\frac{\partial f_{c1}}{\partial a_1} & -\frac{\partial f_{c1}}{\partial a_2} \\
   C_{21} & 2C_{22} & -\frac{\partial f_{c2}}{\partial a_1} & -\frac{\partial f_{c2}}{\partial a_2}
\end{array}
\]

\[
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k
\end{bmatrix} =
\begin{bmatrix}
0 & 0 & -f_{c1} & 0 \\
0 & 0 & -f_{c2} & 0
\end{bmatrix}
\]

Considering linear equations...
**Lagrange multiplier review**

### MATLAB code

**Define constraint functions**

- \( N\text{coef} = \text{length(diag}(C)) \)
- \( a = \text{sym}(\text{'a'}, [1 \text{ N\text{coef}}]) \)
- \( \text{Ncons} = 2 \)
- \( fc1 = a(1)+a(2)-1 \)
- \( fc2 = a(1)-a(2) \)

**Build-up matrix and vector to find coeff**s.

- \( C = \text{C*diag}(2*\text{ones}(1, \text{length(diag}(C)))) \)
- \( \text{Mconst} = \text{double(Mconst)} \)

**Final matrix**

- \( M = [C -\text{Mconst}; \text{Mconst} \text{zeros(Ncons, Ncons)]} \)

**Obtain coeff**s.

- \( A = \text{inv}(M)*\text{vector} \)

---

- **Considering known** \( C \)

\[
\begin{align*}
  f_{c1} &= a_1 + a_2 - 1 \\
  f_{c2} &= a_1 - a_2
\end{align*}
\]

- **Considering known** \( C \)

\[
C = [5 \ 0; \ 0 \ 1] \%	ext{noise covariance matrix}
\]

- **Considering linear equations...**

\[
\begin{bmatrix}
  2C_{11} & C_{12} \\
  C_{21} & 2C_{22}
\end{bmatrix}
\begin{bmatrix}
  \frac{\partial f_{c1}}{\partial a_1} & -\frac{\partial f_{c2}}{\partial a_1} \\
  -\frac{\partial f_{c1}}{\partial a_2} & \frac{\partial f_{c2}}{\partial a_2}
\end{bmatrix}
\begin{bmatrix}
  a_1 \\
  a_2
\end{bmatrix}
= \begin{bmatrix}
  0 \\
  0
\end{bmatrix}
\]

\[
\begin{bmatrix}
  -f_{c1} | a_1=0 \\
  -f_{c2} | a_2=0
\end{bmatrix}
\]

- **Obtaining coeff**s.

\[
A = \text{inv}(M)*\text{vector}
\]
Lagrange multiplier review

MATLAB code

Considering known $C$

$C = [5 \ 0; \ 0 \ 1]$; %noise covariance matrix

\[ f_{c_1} = a_1 + a_2 - 1 \]
\[ f_{c_2} = a_1 - a_2 \]

%DEFINE CONSTRAINT FUNCTIONS
Ncoef = length(diag(C)); %number of dimensions/coeffs (a1, a2, ...)
a = sym('a', [1 Ncoef]);
Ncons = 2; %number of constraint functions
fc1 = a(1)+a(2)-1; %constraint function 1
fc2 = a(1)-a(2); %constraint function 2

% BUILD-UP MATRIX AND VECTOR TO FIND COEFFs.
%%%%%%%% LEFT-HAND MATRIX
% COVARIANCE PART
C = C*diag(2*ones(1, length(diag(C)))); %covariane matrix but diag*2

% CONSTRAINTS FUNCTIONS PART
for j = 1:Ncons
    s = sprintf('fc%d', j);
    AA = diff(eval(s),a(1));
    for i = 2:Ncoef
        AA = [AA diff(eval(s),a(i))];
    end
    if j==1 Mconst = AA;
    else Mconst = [Mconst; AA]; end
end
Mconst = double(Mconst); %transform symbolic to numeric

% FINAL MATRIX
M = [C -Mconst; Mconst zeros(Ncons, Ncons)]; %build up final matrix

% RIGHT-HAND VECTOR
for i = 1:Ncoef
    eval(sprintf('a%d=0', i));
end
vector = [0 0 -subs(fc1) -subs(fc2)'];

% OBTAIN COEFFs.
A = inv(M)*vector

Considering linear equations...
**Lagrange multiplier review**

**MATLAB code**

**Considering known $C$**

\[
\begin{align*}
C &= \begin{bmatrix} 5 & 0 \\ 0 & 1 \end{bmatrix}; \quad \text{%noise covariance matrix} \\

f_{c1} &= a_1 + a_2 - 1 \\
f_{c2} &= a_1 - a_2
\end{align*}
\]

\[
\begin{bmatrix}
2C_{11} & C_{12} & -\frac{\partial f_{c1}}{\partial a_1} & -\frac{\partial f_{c2}}{\partial a_1} \\
C_{21} & 2C_{22} & -\frac{\partial f_{c1}}{\partial a_2} & -\frac{\partial f_{c2}}{\partial a_2}
\end{bmatrix}
\]

\[
\begin{bmatrix}
a_1 \\
a_2 \\
\lambda \\
k
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 \\
0 \\
-f_{c1} |_{a_1=0, a_2=0} \\
-f_{c2} |_{a_1=0, a_2=0}
\end{bmatrix}
\]

Considering linear equations...

Considering linear equations...
Lagrange multiplier review

MATLAB code

\[ f_{c1} = a_1 + a_2 - 1 \]
\[ f_{c2} = a_1 - a_2 \]

\%DEFINE CONSTRAINT FUNCTIONS
Ncoef = length(diag(C)); \%number of dimensions/coeffs (a1, a2, ...)
a = sym('a', [1 Ncoef]);
Ncons = 2; \%number of constraint functions
fc1 = a(1)+a(2)-1; \%constraint function 1
fc2 = a(1)-a(2); \%constraint function 2

\% BUILD-UP MATRIX AND VECTOR TO FIND COEFFs.
\%\%\%\%\%\%\%\%\%\% LEFT-HAND MATRIX
\% COVARIANCE PART
C = C*diag(2*ones(1, length(diag(C)))); \%covariante matrix but diag*2

\% CONSTRAINTS FUNCTIONS PART
for j = 1:Ncons
    s = sprintf('fc%d', j);
    AA = diff(eval(s),a(1));
    for i = 2:Ncoef
        AA = [AA diff(eval(s),a(i))];
    end
    if j==1 Mconst = AA;
    else Mconst = [Mconst; AA]; end
end
Mconst = double(Mconst); \%transform symbolic to numeric

\% FINAL MATRIX
M = [C -Mconst; Mconst zeros(Ncons, Ncons)]; \%build up final matrix

\%\%\%\%\%\%\%\%\%\% RIGHT-HAND VECTOR
for i = 1:Ncoef
    eval(sprintf('a%d=0', i));
end
vector = [0 0 -subs(fc1) -subs(fc2)];

\% OBTAIN COEFFs.
A = inv(M)*vector

Final result (coefficients)

- Considering known \( C \)
  \( C = [5 0; 0 1] \); \%noise covariance matrix

- Considering linear equations...

\( a_1 + a_2 = 1 \)
\( a_1 - a_2 = 0 \)
Example based on simulation
clear all; close all;

% Signal Generation
x = 0:1:50;
y = gaussmf(x,[5 25]);
N = 10000;

for i=1:N;
    truePeak(i) = rand+0.5;
    noise(i,:) = randn(1,51)*0.1;
    yt(:, i) = y*truePeak(i);
    yy(:, i) = y*truePeak(i)+noise(i,:);
end

plot(x,yy(:,1)); %signal with noise
hold on;
plot(x,yt(:,1), 'r'); %true signal
xlabel('Samples')
ylabel('Amplitude')
Example based on simulation

plot(x,yy(:,1)); %signal with noise
hold on;
plot(x,yt(:,1), 'r'); %true signal
xlabel('Samples')
ylabel('Amplitude')

% peak estimator based on max.
peakEst1 = max(yy);

% peak estimator based on position
peakEst2 = yy(25,:);
Example based on simulation

```matlab
plot(x,yy(:,1)); %signal with noise
hold on;
plot(x,yt(:,1), 'r'); %true signal
xlabel('Samples')
ylabel('Amplitude')

% peak estimator based on max.
peakEst1 = max(yy);

% peak estimator based on position
peakEst2 = yy(25,:);
```
clear all; close all;

% Signal generation
x = 0:1:50;
y = gaussmf(x,[5 25]);
N = 10000;

for i=1:N;
    truePeak(i) = rand+0.5;
    noise(i,:) = randn(1,51)*0.1;
    yt(:, i) = y*truePeak(i);
    yy(:, i) = y*truePeak(i)+noise(i,:);
end

plot(x,yy(:,1));
hold on;
plot(x,yt(:,1), 'r');
xlabel('Samples')
ylabel('Amplitude')

% Two simple peak estimators
peakEst1 = max(yy);

% peak estimator based on position
peakEst2 = yy(25,:);

% mean results
mean(peakEst1-truePeak)
std(peakEst1-truePeak)

mean(peakEst2-truePeak)
std(peakEst2-truePeak)
Example based on simulation

\[
\text{peakEst1} = \text{max}(yy); \\
\text{mean} = 0.0872 \\
\text{std} = 0.0666
\]

\[
\text{peakEst2} = yy(25,:); \\
\text{mean} = -0.0195 \\
\text{std} = 0.0995
\]
Example based on simulation

Next we should propose a new peak estimator based on estimator variance minimization.
Example based on simulation

• To find the estimator coefficients we need to know the signal shape and the noise covariance matrix since variance minimization depends on the following equations

\[
\text{Var}[\hat{A}] = \text{Var}[a^T n] = a^T C a
\]

\[
\sum_{k=1}^{N} a[k] g[k] = 1
\]

\[
\sum_{k=1}^{N} a[k] = 0
\]

• Usually, both \( g[k] \) and \( C \) should be estimated from data...
Example based on simulation

- Signal shape estimation $g[k]$

  $y_{\text{Mean}} = \text{mean}(yy')$;

  A simple solution. But it might be more complex...
  - Timing synchronization
  - Curve fitting
  - Pedestal removal
  - ...

- Noise covariance matrix estimation $C$

  $C = \text{cov}(\text{noise})$;
  \[ \text{bar3}(C); \]

  Supposing we have a noise database
Example based on simulation

- **Finding the coefficients vector $a$ ...**

```matlab
yMean = mean(yy');
C = cov(noise);

%constraint functions matrix
Fcons = [yMean -1; ones(1,length(diag(C))) 0];

%gives the filter coeffs. more lagrange coeffs.
A = lagrangeMin(C, Fcons);

%plot coefficients
plot(A(1:length(diag(C))))
ylabel('Amplitude')
xlabel('Coefficients')
xlim([0 length(diag(C))+1])
```

![Graph showing the coefficients and amplitude plot.](image)
Example based on simulation

- **Finding the coefficients vector** $\mathbf{a}$ ...

```matlab
yMean = mean(yy');
C = cov(noise);

% constraint functions matrix
Fcons = [yMean -1; ones(1,length(diag(C))) 0];

% gives the filter coeffs. more lagrange
A = lagrangeMin(C, Fcons);

% plot coefficients
plot(A(1:length(diag(C))));
ylabel('Amplitude');
xlabel('Coefficients');
xlim([0 length(diag(C))]);
```

Next we should apply this coeffs to estimate peak amplitude
Example based on simulation

• Apply coefficients!

%reorganize coefficients
coeffs = A(1:51,1)';

%estimate peaks
peakEstOpt = coeffs*yy;

%plot histogram
[n3,x3]=hist(peakEstOpt-truePeak, -0.5:0.02:0.5);
h3=bar(x3,n3,'hist');
set(h3,'FaceColor','no','EdgeColor','b')

%mean and std of error
mean(peakEstOpt-truePeak)
std(peakEstOpt-truePeak)
Example based on simulation

\textbf{peakEst1} = \texttt{max(yy)};
\begin{align*}
\text{mean} &= 0.0872 \\
\text{std} &= 0.0666
\end{align*}

\textbf{peakEst2} = \texttt{yy(25,:)};
\begin{align*}
\text{mean} &= -0.0195 \\
\text{std} &= 0.0995
\end{align*}

\textbf{peakEstOpt} = \texttt{coeffs*yy};
\begin{align*}
\text{mean} &= 0.0070 \\
\text{std} &= 0.0422
\end{align*}