Covariance Matrix Adaptation
CMA-ES

CJ Chung

Fall 2025

Covariance Matrix

- https://statisticsbyjim.com/basics/covariance/
- https://towardsdatascience.com/understanding-the-covariance-matrix-92076554ea44/
- https://youtu.be/152tSYtiQbw?si=EI367ppk9iTolcQO
- https://youtu.be/2bcmklvrXTQ?si=VI9K7uX5F48eNbvF
- …

Mean Centering

For a dataset (or a vector of values), **mean centering** means:

$$x_i^{\text{centered}} = x_i - \bar{x}$$

where $\bar{x}$ is the **mean** of all the values.

So you're shifting the data so that its mean becomes **0**.

Example: Suppose your data is: $[2, 4, 6, 8]$

- Mean = $(2 + 4 + 6 + 8)/4 = 5$.
- Mean-centered data =

$$[2-5,\ 4-5,\ 6-5,\ 8-5] = [-3, -1, 1, 3]$$

- New mean = 0.

Variance = spread (in squared units)

Variance

- Variance measures **how spread out** a set of numbers is from the mean.
- Formula (population variance):

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$$

where

- N = number of data points
- x_i = each data point
- μ = mean (average)

So variance is the **average of squared deviations** from the mean.

Standard Deviation

Standard Deviation (std)

- Standard deviation is simply the **square root of variance**.

$$\sigma = \sqrt{\sigma^2}$$

- It's more interpretable because it's in the **same units** as the original data. (Variance is in squared units.)

SD = Typical distance from the mean (in original units).

Covariance

- Covariance tells us how **two variables vary together**.
- Formula (population covariance between X and Y):

$$\mathrm{Cov}(X, Y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_X)(y_i - \mu_Y)$$

- If **Cov > 0**: when X increases, Y tends to increase (positive relationship).
- If **Cov < 0**: when X increases, Y tends to decrease (negative relationship).
- If **Cov ≈ 0**: no linear relationship.

Example:

- $X = [1, 2, 3]$, $Y = [2, 4, 6]$ → positive covariance (they grow together).
- $X = [1, 2, 3]$, $Y = [6, 4, 2]$ → negative covariance (one grows, the other shrinks).

Covariance Matrix

When you have **more than two variables**, we arrange all covariances in a matrix.

For variables $X_1, X_2, \ldots, X_n$:

$$\Sigma = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) & \cdots & \mathrm{Cov}(X_1, X_n) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) & \cdots & \mathrm{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(X_n, X_1) & \mathrm{Cov}(X_n, X_2) & \cdots & \mathrm{Var}(X_n) \end{bmatrix}$$

- Diagonal entries = variances of each variable.
- Off-diagonal entries = covariances between pairs.
- Always symmetric ($\mathrm{Cov}(X, Y) = \mathrm{Cov}(Y, X)$).

Covariance Matrix Example

$$\mathrm{Cov}(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_X)(y_i - \mu_Y)$$

Let's say:

- $X = [1, 2, 3]$
- $Y = [2, 4, 6]$

Mean of $X = 2$, mean of $Y = 4$.

- Var(X) = $\frac{(1-2)^2+(2-2)^2+(3-2)^2}{3} = \frac{2}{3}$ $= 0.67$
- Var(Y) = $\frac{(2-4)^2+(4-4)^2+(6-4)^2}{3} = \frac{8}{3}$ $= 2.67$
- Cov(X,Y) = $\frac{(1-2)(2-4)+(2-2)(4-4)+(3-2)(6-4)}{3} = \frac{4}{3}$ $= 1.33$

So covariance matrix is:

$$\Sigma = \begin{array}{cc} \begin{array}{cc} X & \quad Y \end{array} & \\ \begin{bmatrix} 0.67 & 1.33 \\ 1.33 & 2.67 \end{bmatrix} & \begin{array}{c} X \\ Y \end{array} \end{array}$$

Real-world Covariance Matrix Example

1. Data

Suppose we have 5 students:

Student	Hours Studied (X)	Test Score (Y)
1	2	65
2	4	70
3	6	75
4	8	80
5	10	85

2. Find the Means and variances

$$\mu_X = \frac{2+4+6+8+10}{5} = 6$$

$$\mu_Y = \frac{65+70+75+80+85}{5} = 75$$

Variance of X

$$\mathrm{Var}(X) = \frac{\sum (X_i - \mu_X)^2}{N}$$

Deviations: (-4, -2, 0, 2, 4) → Squares: (16, 4, 0, 4, 16) → Sum = 40

$$\mathrm{Var}(X) = \frac{40}{5} = 8$$

or 40/4=10

Variance of Y

Deviations: (-10, -5, 0, 5, 10) → Squares: (100, 25, 0, 25, 100) → Sum = 250

$$\mathrm{Var}(Y) = \frac{250}{5} = 50$$

or 250/4=62.5

3. Compute Deviations

Subtract the mean from each value:

6 75

Student	X	Y	$X - \mu_X$	$Y - \mu_Y$	Product $(X - \mu_X)(Y - \mu_Y)$
1	2	65	-4	-10	40
2	4	70	-2	-5	10
3	6	75	0	0	0
4	8	80	2	5	10
5	10	85	4	10	40

4. Apply Covariance Formula

For **population covariance**:

$$\mathrm{Cov}(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(X_i - \mu_X)(Y_i - \mu_Y)$$

$$= \frac{40 + 10 + 0 + 10 + 40}{5} = \frac{100}{5} = 20$$

For **sample covariance** (common in statistics, use $N-1$ in denominator):

$$\mathrm{Cov}(X,Y) = \frac{100}{4} = 25$$

Positive: more hours studied, higher test scores

Covariance is positive → as study hours increase, test scores also increase.
The magnitude (20 or 25) depends on whether we treat the data as population or sample.

$$\text{Covariance Matrix} = \begin{array}{c} \begin{array}{cc} X & Y \end{array} \\ \begin{bmatrix} 8 & 25 \\ 25 & 50 \end{bmatrix} \end{array}$$

Cov-CorrelMatrix.ipynb

```
import numpy as np
# Suppose we have 5 "sample" observations (rows) of 2 variables (columns)
# Each row = one observation [x, y] - x: Hours studied, y: Test score
data = np.array([
    [2, 65],
    [4, 70],
    [6, 75],
    [8, 80],
    [10, 85],
])

# Transpose the data so that variables are rows
# Each row = variable, each column = observation

cov_matrix = np.cov(data.T)

print("Data:\n", data)
print("Transposed Data: \n", data.T)
print("\nCovariance matrix:\n", cov_matrix)
```

```
Data:
 [[ 2 65]
 [ 4 70]
 [ 6 75]
 [ 8 80]
 [10 85]]
Transposed Data:
 [[ 2  4  6  8 10]
 [65 70 75 80 85]]

Covariance matrix:
 [[10.  25. ]
 [25.  62.5]]
```

```
# Compute the covariance matrix manually
# Formula: Cov(X, Y) = Σ[(xi - x̄)(yi - ȳ)] / (n - 1)
data_manual = np.array([
    [2, 65],     [4, 70],     [6, 75],     [8, 80],     [10, 85]        ])
n = data_manual.shape[0] # Number of observations

mean_x = np.mean(data_manual[:, 0]) # Calculate the means for each variable
mean_y = np.mean(data_manual[:, 1])

# Calculate the covariance between X and Y
cov_xy = np.sum((data_manual[:, 0] - mean_x) * (data_manual[:, 1] - mean_y)) / (n - 1)

# Calculate the variance of X (covariance of X with itself)
var_x = np.sum((data_manual[:, 0] - mean_x)**2) / (n - 1)

# Calculate the variance of Y (covariance of Y with itself)
var_y = np.sum((data_manual[:, 1] - mean_y)**2) / (n - 1)

# Construct the covariance matrix
covariance_matrix_manual = np.array([
    [var_x, cov_xy],
    [cov_xy, var_y]
])
print("\nManual Covariance matrix:\n", covariance_matrix_manual)
```

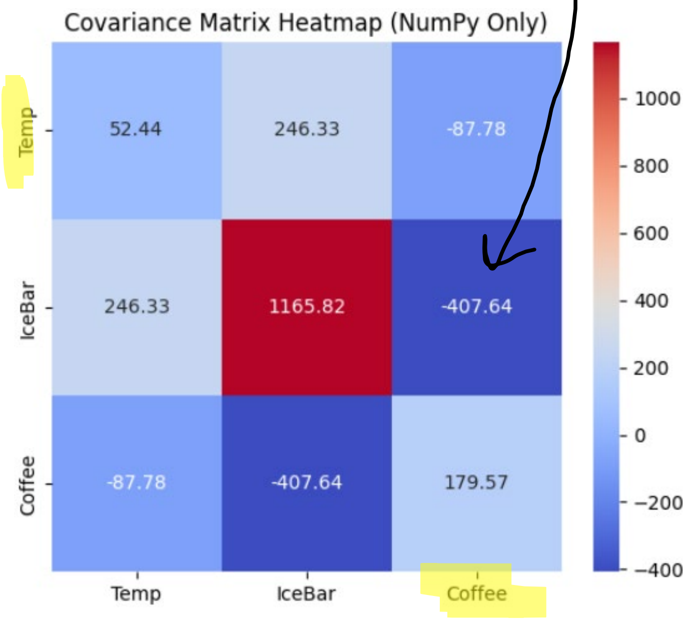
```
Manual Covariance matrix:
 [[10.  25. ]
 [25.  62.5]]
```

3 variables

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Example: 10 days. Variables are rows!
temperature =   np.array([15, 18, 20, 22, 25, 27, 30, 32, 35, 36])
iceBars =       np.array([2,  12, 26, 35, 40, 55, 70, 82, 96, 98])
hotCoffeeCups = np.array([50, 40, 30, 20, 18, 16, 14, 13, 12, 10])

# Stack into 2D array: each row = variable, each column = observation
data = np.vstack((temperature, iceBars, hotCoffeeCups))
print("Data:\n", data)

cov_matrix = np.cov(data) # var as rows; rowvar=True by default
print(f"Cov. matrix:\n{np.array2string(cov_matrix, precision=2)}")

plt.figure(figsize=(6, 5))
sns.heatmap(cov_matrix, annot=True, fmt=".2f", cmap="coolwarm",
            xticklabels=["Temp", "IceBar", "Coffee"],
            yticklabels=["Temp", "IceBar", "Coffee"])
plt.title("Covariance Matrix Heatmap (NumPy Only)")
plt.show()
```

```
Data:
 [[15 18 20 22 25 27 30 32 35 36]
 [ 2 12 26 35 40 55 70 82 96 98]
 [50 40 30 20 18 16 14 13 12 10]]
Covariance matrix:
[[  52.44  246.33  -87.78]
 [ 246.33 1165.82 -407.64]
 [ -87.78 -407.64  179.57]]
```

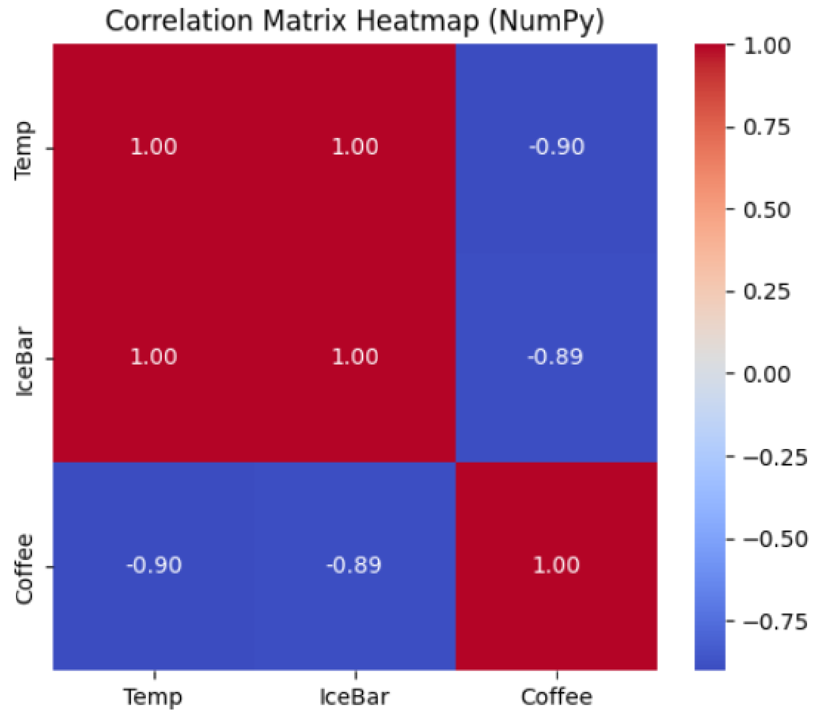
Negative linear relationship btn IceBar and Coffee sale



Correlation Matrix

```
# Compute the correlation matrix
correlation_matrix = np.corrcoef(data)

print(f"\nCorrelation matrix:\n{np.array2string(correlation_matrix, precision=2)}")

# Visualize the correlation matrix
plt.figure(figsize=(6, 5))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap="coolwarm",
            xticklabels=["Temp", "IceBar", "Coffee"],
            yticklabels=["Temp", "IceBar", "Coffee"])
plt.title("Correlation Matrix Heatmap (NumPy)")
plt.show()
```

```
Correlation matrix:
[[ 1.    1.   -0.9 ]
 [ 1.    1.   -0.89]
 [-0.9  -0.89  1.  ]]
```



Covariance matrix vs Correlation matrix (1 /4)

- **Scale:**
  - **Covariance:** The values in the covariance matrix are not standardized. Their magnitude depends on the scale of the variables. This means you can't easily compare covariances between different pairs of variables that have different units or scales.
  - **Correlation:** The values in the correlation matrix are standardized. They range from -1 to +1. This makes it easy to compare the strength and direction of the linear relationship between any pair of variables, regardless of their original scales.

Covariance matrix vs Correlation matrix (2 /4)

- **Interpretation:**
  - **Covariance:** Tells you the direction of the linear relationship (positive or negative) and gives an idea of the magnitude, but the magnitude is scale-dependent.
  - **Correlation:** Tells you the direction of the linear relationship (positive or negative) and the strength of the linear relationship on a standardized scale.
    - A correlation of +1 indicates a perfect positive linear relationship.
    - A correlation of -1 indicates a perfect negative linear relationship.
    - A correlation of 0 indicates no linear relationship.

Covariance matrix vs Correlation matrix (3 /4)

- **Calculation:**
  - **Covariance:** Calculated based on the deviations of each data point from the mean of its respective variable.

$$\mathrm{Cov}(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

where

- $\bar{X}$ = mean of X
- $\bar{Y}$ = mean of Y

  - **Correlation:** Calculated by dividing the covariance between two variables by the product of their standard deviations. This standardization process removes the effect of the variables' scales.

$$\rho_{XY} = \frac{\mathrm{Cov}(X,Y)}{\sigma_X \sigma_Y}$$

HW8: Your own Hybrid CMA-ES algorithm - Class Competition with extra credit opportunity

- 2+5 = 7 points (No Quiz #4)
- Due Monday Dec. 1 (syllabus defined late penalies apply)
- Class Competition:
  - Unknown Optimization function will be used to rank solutions
  - Best "mean" fitness found
  - Add your algorithm function in "ESalgorithms10funcs.ipynb"
  - 1st place winner: +5 points, 2nd place: +4 points, 3rd place: +3 points
  - Additional Prizes: LTU mugs, LTU caps, and many others

HW8: Create your own hybrid CMA-ES (mu+lamda) algorithm to solve function minimization problms

- Use 1/5 success rule for a global sigma
- A local sigma for each variable is adapted by using **correlation matrix**
  - First use the variance from the diagonal element of the correlation matrix to adapt the local sigma per variable
  - Second, each local sigma is also adapted using correlation matrix between variables. Any strong correlation (positive or negative) between variables should increase the stepsize for both variables.
- Combine global sigma and local sigma for the stepSize to mutate each var
- Correlation matrix needs to be calculated from the population with mu individuals right after mu parents were selected for the next generation.
- For the selection from mu+lamda candidates, you could use tournament selection algorithm
- Use generative AI tools effectively

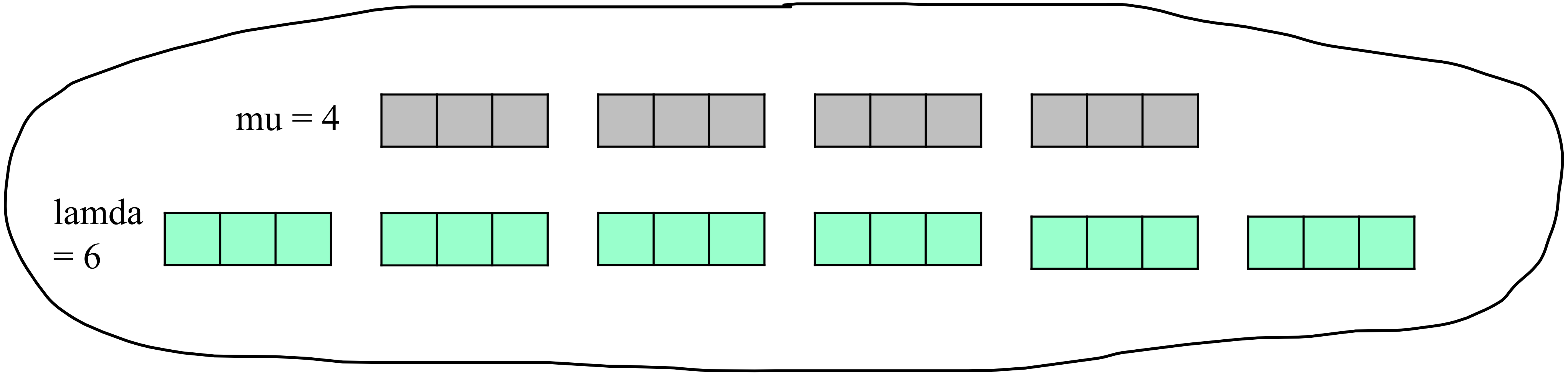
HW8: Create your own hybrid CMA-ES (mu+lamda) algorithm to solve function minimization problms

Global stepsize (sigma) maintained by 1/5 rule

Local sigma per variable

```
Correlation matrix:
[[ 1.     1.    -0.9 ]
 [ 1.     1.    -0.89]
 [-0.9   -0.89   1.  ]]
```

Calcualted from 4 observations of 3 variables



Some Useful Videos

- https://youtu.be/5qCAOyNJROg?si=AMgn0rP7EGh19ZLF (1)
- https://youtu.be/vWZgFokjON8?si=Nyy2gZw8BpGpAupS (2)
- https://youtu.be/hv2BXzjYeRw?si=Qv65pZOsJFI4EPzS (3)

- https://youtu.be/7VBKLH3oDuw?si=GiUPr99lY6-SX3m_