Problem 1: Conditional Gaussian Distributions

We are given a partitioned Gaussian vector $\mathbf{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$:

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \\ \mathbf{x}_c \end{pmatrix}, \quad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \\ \boldsymbol{\mu}_c \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} & \Sigma_{ac} \\ \Sigma_{ba} & \Sigma_{bb} & \Sigma_{bc} \\ \Sigma_{ca} & \Sigma_{cb} & \Sigma_{cc} \end{pmatrix}, \quad \boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1} = \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} & \Lambda_{ac} \\ \Lambda_{ba} & \Lambda_{bb} & \Lambda_{bc} \\ \Lambda_{ca} & \Lambda_{cb} & \Lambda_{cc} \end{pmatrix}.$$

Since $\mathbf{x}_c$ is marginalized out, the joint distribution of $(\mathbf{x}_a, \mathbf{x}_b)$ depends only on the corresponding sub-blocks of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. The marginal is:

$$\begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix}, \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \right).$$

However, to find the conditional $p(\mathbf{x}_a \mid \mathbf{x}_b)$, we work with the precision matrix of this marginal distribution. Note that the inverse of the marginal covariance matrix is the corresponding sub-block of the joint precision matrix $\boldsymbol{\Lambda}$ only if the remaining variables are independent, which is not generally true.

Let us expand the quadratic form in the exponent of the Gaussian for the joint $(\mathbf{x}_a, \mathbf{x}_b)$ distribution. Let the precision of this marginal be denoted by the block matrix $\tilde{\boldsymbol{\Lambda}}$:

$$\tilde{\boldsymbol{\Lambda}} = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}^{-1}.$$

For notation in this specific derivation, we assume the problem implies working directly with the partitions of the precision matrix provided. We expand the exponent:

$$\begin{aligned} Q &= -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Lambda} (\mathbf{x} - \boldsymbol{\mu}) \\ &= -\frac{1}{2} \begin{pmatrix} \mathbf{x}_a - \boldsymbol{\mu}_a \\ \mathbf{x}_b - \boldsymbol{\mu}_b \end{pmatrix}^\top \begin{pmatrix} \Lambda_{aa} & \Lambda_{ab} \\ \Lambda_{ba} & \Lambda_{bb} \end{pmatrix} \begin{pmatrix} \mathbf{x}_a - \boldsymbol{\mu}_a \\ \mathbf{x}_b - \boldsymbol{\mu}_b \end{pmatrix} \\ &= -\frac{1}{2}(\mathbf{x}_a - \boldsymbol{\mu}_a)^\top \Lambda_{aa} (\mathbf{x}_a - \boldsymbol{\mu}_a) - (\mathbf{x}_a - \boldsymbol{\mu}_a)^\top \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b) - \frac{1}{2}(\mathbf{x}_b - \boldsymbol{\mu}_b)^\top \Lambda_{bb} (\mathbf{x}_b - \boldsymbol{\mu}_b). \end{aligned}$$

To find $p(\mathbf{x}_a \mid \mathbf{x}_b)$, we treat $\mathbf{x}_b$ as constant and complete the square for $\mathbf{x}_a$. The terms involving $\mathbf{x}_a$ are:

$$\begin{aligned} Q(\mathbf{x}_a) &= -\frac{1}{2}\mathbf{x}_a^\top \Lambda_{aa} \mathbf{x}_a + \mathbf{x}_a^\top \Lambda_{aa} \boldsymbol{\mu}_a - \mathbf{x}_a^\top \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b) + \dots \\ &= -\frac{1}{2}\mathbf{x}_a^\top \Lambda_{aa} \mathbf{x}_a + \mathbf{x}_a^\top \left(\Lambda_{aa} \boldsymbol{\mu}_a - \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b) \right) + \text{const.} \end{aligned}$$

Comparing this to the standard quadratic form $-\frac{1}{2}(\mathbf{x}_a - \boldsymbol{\mu}_{a|b})^\top \boldsymbol{\Sigma}_{a|b}^{-1} (\mathbf{x}_a - \boldsymbol{\mu}_{a|b})$, we identify:

$$\boldsymbol{\Sigma}_{a|b}^{-1} = \Lambda_{aa} \implies \boldsymbol{\Sigma}_{a|b} = \Lambda_{aa}^{-1}.$$

And the linear term coefficient is $\boldsymbol{\Sigma}_{a|b}^{-1} \boldsymbol{\mu}_{a|b}$. Thus:

$$\begin{aligned} \Lambda_{aa} \boldsymbol{\mu}_{a|b} &= \Lambda_{aa} \boldsymbol{\mu}_a - \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b) \\ \boldsymbol{\mu}_{a|b} &= \boldsymbol{\mu}_a - \Lambda_{aa}^{-1} \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b). \end{aligned}$$

Thus, the conditional distribution is:

$$p(\mathbf{x}_a \mid \mathbf{x}_b) = \mathcal{N}\left(\mathbf{x}_a \mid \boldsymbol{\mu}_a - \Lambda_{aa}^{-1} \Lambda_{ab} (\mathbf{x}_b - \boldsymbol{\mu}_b),\ \Lambda_{aa}^{-1} \right).$$

Problem 2: The Woodbury Matrix Identity

We wish to verify the identity:

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}.$$

Let $X = (A + BCD)$. We prove this by showing $X \cdot X^{-1} = I$. Let the RHS be Y.

$$\begin{aligned} XY &= (A + BCD)\left[A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}\right] \\ &= (A + BCD)A^{-1} - (A + BCD)A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1} \\ &= (I + BCDA^{-1}) - (B + BCDA^{-1}B)(C^{-1} + DA^{-1}B)^{-1}DA^{-1}. \end{aligned}$$

Factor out B from the second term:

$$\begin{aligned} &= I + BCDA^{-1} - B(I + CDA^{-1}B)(C^{-1} + DA^{-1}B)^{-1}DA^{-1} \\ &= I + BCDA^{-1} - B\left[C(C^{-1} + DA^{-1}B)\right](C^{-1} + DA^{-1}B)^{-1}DA^{-1} \\ &= I + BCDA^{-1} - BC\underbrace{\left[(C^{-1} + DA^{-1}B)(C^{-1} + DA^{-1}B)^{-1}\right]}_{I}DA^{-1} \\ &= I + BCDA^{-1} - BCDA^{-1} \\ &= I. \end{aligned}$$

This confirms the identity.

Computational Efficiency: Consider $\mathbf{A}$ as an $n \times n$ diagonal matrix, $\mathbf{B}$ is $n \times k$, $\mathbf{C}$ is $k \times k$, and $\mathbf{D}$ is $k \times n$, where $k \ll n$.

- Naively inverting $(A + BCD)$ requires $O(n^3)$ operations.
- Using Woodbury, we only invert $(C^{-1} + DA^{-1}B)$, which is a $k \times k$ matrix. This costs $O(k^3)$.
- Since A is diagonal, A^{-1} is trivial $O(n)$.
- The matrix multiplications cost approx $O(nk^2)$.

Thus, when $k \ll n$, the complexity drops from cubic in n to linear in n.

Problem 3: Bayesian Inference for Gaussian (Unknown Mean and Precision)

Likelihood for N i.i.d samples with precision $\lambda = 1/\sigma^2$:

$$p(\mathbf{X} \mid \mu, \lambda) \propto \lambda^{N/2} \exp\left(-\frac{\lambda}{2}\sum_{n=1}^{N}(x_n - \mu)^2\right).$$

The conjugate prior is Normal-Gamma:

$$p(\mu, \lambda) = \mathcal{N}(\mu \mid \mu_0, (\beta\lambda)^{-1})\mathrm{Gam}(\lambda \mid a, b) \propto \lambda^{1/2} e^{-\frac{\beta\lambda}{2}(\mu-\mu_0)^2} \lambda^{a-1} e^{-b\lambda}.$$

Combining terms in the prior:

$$p(\mu, \lambda) \propto \lambda^{a-1/2} \exp\left(-\frac{\lambda}{2}\left[\beta(\mu - \mu_0)^2 + 2b\right]\right).$$

Multiplying Prior $\times$ Likelihood:

$$p(\mu, \lambda \mid \mathbf{X}) \propto \lambda^{a-1/2+N/2} \exp\left(-\frac{\lambda}{2}\left[\beta(\mu-\mu_0)^2 + 2b + \sum(x_n-\mu)^2\right]\right).$$

We complete the square for μ inside the exponent. Let $\bar{x} = \frac{1}{N}\sum x_n$.

$$\sum(x_n-\mu)^2 = \sum x_n^2 - 2N\bar{x}\mu + N\mu^2.$$

Total term in bracket:

$$\begin{aligned}[\dots] &= \beta(\mu^2 - 2\mu\mu_0 + \mu_0^2) + 2b + \sum x_n^2 - 2N\bar{x}\mu + N\mu^2 \\ &= (\beta+N)\mu^2 - 2(\beta\mu_0 + N\bar{x})\mu + (\beta\mu_0^2 + 2b + \sum x_n^2).\end{aligned}$$

This is quadratic in μ, implying a new Gaussian component with precision $\lambda(\beta+N)$ and mean:

$$\mu_N = \frac{\beta\mu_0 + N\bar{x}}{\beta+N}.$$

The remaining terms form the new Gamma parameters.

$$a_N = a + \frac{N}{2}, \quad b_N = b + \frac{1}{2}\sum x_n^2 + \frac{\beta}{2}\mu_0^2 - \frac{\beta+N}{2}\mu_N^2.$$

This confirms the posterior is Normal-Gamma.

Problem 4: The Wishart Distribution

4a. Conjugacy Proof

The Wishart prior for precision matrix $\mathbf{\Lambda}$ is:

$$\mathcal{W}(\mathbf{\Lambda} \mid \mathbf{W}, \nu) \propto |\mathbf{\Lambda}|^{(\nu-D-1)/2} \exp\left(-\frac{1}{2}\mathrm{Tr}(\mathbf{W}^{-1}\mathbf{\Lambda})\right).$$

The likelihood for Multivariate Gaussian data $\mathbf{X}$ is:

$$p(\mathbf{X} \mid \boldsymbol{\mu}, \mathbf{\Lambda}) \propto |\mathbf{\Lambda}|^{N/2} \exp\left(-\frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu})^\top \mathbf{\Lambda}(\mathbf{x}_n - \boldsymbol{\mu})\right).$$

Using the trace trick $\mathbf{x}^\top A\mathbf{x} = \mathrm{Tr}(A\mathbf{x}\mathbf{x}^\top)$:

$$\sum(\mathbf{x}_n - \boldsymbol{\mu})^\top \mathbf{\Lambda}(\mathbf{x}_n - \boldsymbol{\mu}) = \mathrm{Tr}\left(\mathbf{\Lambda}\sum(\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^\top\right) = \mathrm{Tr}(\mathbf{\Lambda} \cdot N\mathbf{S}).$$

Multiplying Prior $\times$ Likelihood:

$$\begin{aligned}p(\mathbf{\Lambda} \mid \mathbf{X}) &\propto |\mathbf{\Lambda}|^{(\nu-D-1)/2}|\mathbf{\Lambda}|^{N/2} \exp\left(-\frac{1}{2}\mathrm{Tr}(\mathbf{W}^{-1}\mathbf{\Lambda}) - \frac{1}{2}\mathrm{Tr}(N\mathbf{S}\mathbf{\Lambda})\right) \\ &= |\mathbf{\Lambda}|^{(\nu+N-D-1)/2} \exp\left(-\frac{1}{2}\mathrm{Tr}\left([\mathbf{W}^{-1} + N\mathbf{S}]\mathbf{\Lambda}\right)\right).\end{aligned}$$

This is functionally identical to the Wishart form with updated parameters:

$$\nu_{\text{new}} = \nu + N, \quad \mathbf{W}_{\text{new}}^{-1} = \mathbf{W}^{-1} + N\mathbf{S}.$$

4b. Python Simulation

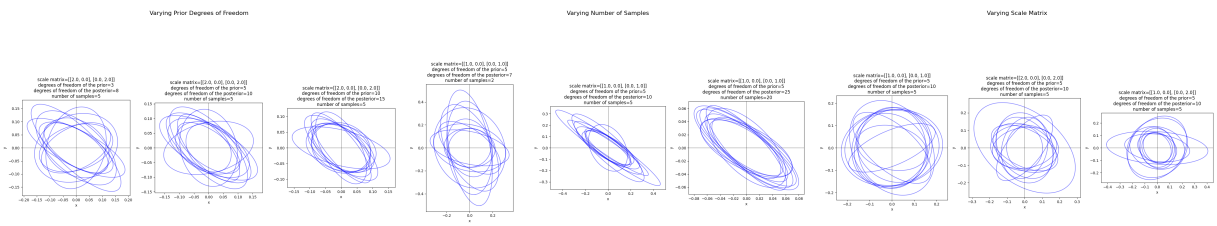

Figure 1: Wishart distributions showing precision ellipses for varying degrees of freedom, sample sizes, and scale matrices.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import wishart

def generate_data(n, cov):
    return np.random.multivariate_normal(mean=[0, 0], cov=cov, size=n)

def sample_wishart(df, scale, num_samples=5):
    samples = []
    for _ in range(num_samples):
        W = wishart.rvs(df=df, scale=scale)
        samples.append(W)
    return samples

def plot_precision_ellipses(precisions, ax, title):
    for p in precisions:
        eigvals, eigvecs = np.linalg.eigh(p)
        if np.any(eigvals <= 0):
            continue

        angles = np.linspace(0, 2*np.pi, 200)
        circle = np.stack([np.cos(angles), np.sin(angles)], axis=1)
        scale_matrix = np.diag(1.0 / np.sqrt(eigvals))
        ellipse_y = circle @ scale_matrix
        ellipse_x = ellipse_y @ eigvecs.T
        ax.plot(ellipse_x[:, 0], ellipse_x[:, 1], 'b', alpha=0.5)

    ax.set_aspect('equal', 'box')
    ax.set_title(title)
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.axhline(0, color='black', linewidth=0.5)
    ax.axvline(0, color='black', linewidth=0.5)

def variation_1_distribution():
    n = 5
    scale_matrix = np.eye(2) * 2.0
    cov = np.array([[1.0, 0.5], [0.5, 1.0]])
    X = generate_data(n, cov)
    sum_T = X.T @ X
    dfs = [3, 5, 10]

    fig, axes = plt.subplots(1, 3, figsize=(15, 4))
```

```python

    for i, df_prior in enumerate(dfs):
        df_post = df_prior + n
        scale_post = scale_matrix + sum_T
        posterior_samples = sample_wishart(df_post, scale_post, num_samples=10)

        title = (
            f"scale matrix={scale_matrix.tolist()}\n"
            f"degrees of freedom of the prior={df_prior}\n"
            f"degrees of freedom of the posterior={df_post}\n"
            f"number of samples={n}"
        )
        plot_precision_ellipses(posterior_samples, axes[i], title)

    fig.suptitle("Varying Prior Degrees of Freedom", fontsize=16)
    plt.tight_layout()
    plt.show()

def variation_2_distribution():
    df_prior = 5
    scale_matrix = np.eye(2)
    cov = np.array([[1.0, 0.5],[0.5, 1.0]])
    ns = [2, 5, 20]
    fig, axes = plt.subplots(1, 3, figsize=(15, 4))

    for i, n in enumerate(ns):
        X = generate_data(n, cov)
        sum_T = X.T @ X
        df_post = df_prior + n
        scale_post = scale_matrix + sum_T
        posterior_samples = sample_wishart(df_post, scale_post, num_samples=10)

        title = (
            f"scale matrix={scale_matrix.tolist()}\n"
            f"degrees of freedom of the prior={df_prior}\n"
            f"degrees of freedom of the posterior={df_post}\n"
            f"number of samples={n}"
        )
        plot_precision_ellipses(posterior_samples, axes[i], title)

    fig.suptitle("Varying Number of Samples", fontsize=16)
    plt.tight_layout()
    plt.show()

def variation_3_distribution():
    df_prior = 5
    n = 5
    scale_matrix_list = [np.eye(2), 2.0 * np.eye(2), np.diag([1.0, 2.0])]
    cov = np.array([[1.0, 0.5],[0.5, 1.0]])
    X = generate_data(n, cov)
    sum_T = X.T @ X

    fig, axes = plt.subplots(1, 3, figsize=(15, 4))

    for i, scale_matrix in enumerate(scale_matrix_list):
        df_post = df_prior + n
        scale_post = scale_matrix + sum_T
        posterior_samples = sample_wishart(df_post, scale_post, num_samples=10)

```

```python
        title = (
            f"scale matrix={scale_matrix.tolist()}\n"
            f"degrees of freedom of the prior={df_prior}\n"
            f"degrees of freedom of the posterior={df_post}\n"
            f"number of samples={n}"
        )
        plot_precision_ellipses(posterior_samples, axes[i], title)

    fig.suptitle("Varying Scale Matrix", fontsize=16)
    plt.tight_layout()
    plt.show()

def main():
    variation_1_distribution()
    variation_2_distribution()
    variation_3_distribution()

if __name__ == "__main__":
    main()
```

Listing 1: Simulation Code

Problem 5: Exponential Family Form of Multivariate Gaussian

The Multivariate Gaussian density is:

$$\begin{aligned} p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= (2\pi)^{-D/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \\ &= (2\pi)^{-D/2}|\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\mathbf{x}^\top \boldsymbol{\Sigma}^{-1}\mathbf{x} + \boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right). \end{aligned}$$

Standard Exponential Family form:

$$p(\mathbf{x} \mid \boldsymbol{\eta}) = h(\mathbf{x})g(\boldsymbol{\eta}) \exp\Big(\boldsymbol{\eta}^\top \mathbf{u}(\mathbf{x})\Big).$$

By inspection:

- Sufficient statistics: $\mathbf{u}(\mathbf{x}) = \begin{pmatrix} \mathbf{x} \\ \mathrm{vec}(\mathbf{x}\mathbf{x}^\top) \end{pmatrix}$.
- Natural parameters: $\boldsymbol{\eta} = \begin{pmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ -\frac{1}{2}\boldsymbol{\Sigma}^{-1} \end{pmatrix}$.

Base measure $h(\mathbf{x}) = (2\pi)^{-D/2}$.

Recovering Standard Parameters: From $\boldsymbol{\eta}_2 = -\frac{1}{2}\boldsymbol{\Sigma}^{-1}$, we get $\boldsymbol{\Sigma}^{-1} = -2\boldsymbol{\eta}_2 \implies \boldsymbol{\Sigma} = -\frac{1}{2}\boldsymbol{\eta}_2^{-1}$. From $\boldsymbol{\eta}_1 = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$, we get $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta}_1 = (-\frac{1}{2}\boldsymbol{\eta}_2^{-1})\boldsymbol{\eta}_1$.

Normalization Factor $g(\boldsymbol{\eta})$: The log-partition function logic leads to:

$$\begin{aligned} g(\boldsymbol{\eta}) &= |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}\boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right) \\ &= |-2\boldsymbol{\eta}_2|^{D/2} \exp\left(-\frac{1}{2}(\boldsymbol{\Sigma}\boldsymbol{\eta}_1)^\top(-2\boldsymbol{\eta}_2)(\boldsymbol{\Sigma}\boldsymbol{\eta}_1)\right) \\ &= |-2\boldsymbol{\eta}_2|^{D/2} \exp\left(\frac{1}{4}\boldsymbol{\eta}_1^\top \boldsymbol{\eta}_2^{-1}\boldsymbol{\eta}_1\right). \end{aligned}$$