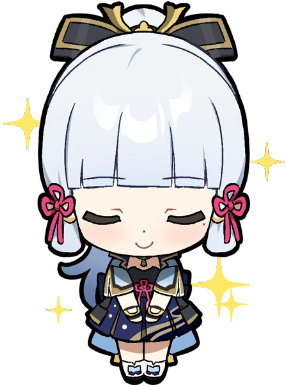
# Sparse Cholesky Factorization by Greedy Conditional Selection


Stephen Huan

Theory Club


February 28, 2022

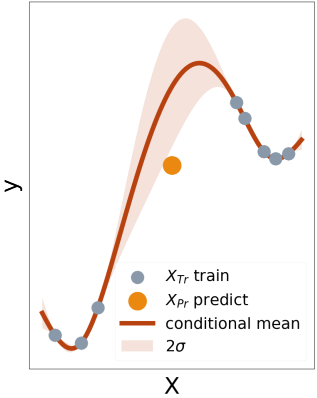


# Table of Contents



# The Problem: Gaussian Process Regression

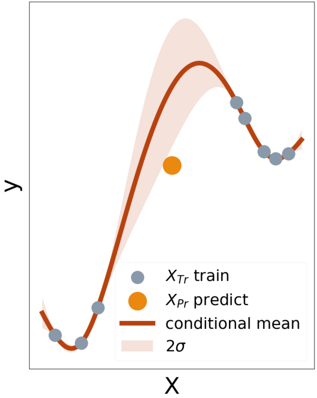
Measurements $\boldsymbol{y}_{\mathsf{Tr}}$ at $N$ points $X_{\mathsf{Tr}}$



# The Problem: Gaussian Process Regression

Measurements $\boldsymbol{y}_{\mathrm{Tr}}$ at $N$ points $X_{\mathrm{Tr}}$

Estimate unseen data $\boldsymbol{y}_{\mathrm{Pr}}$ at $X_{\mathrm{Pr}}$

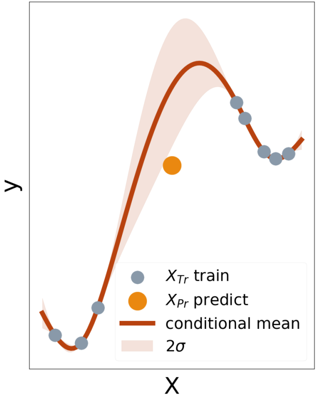


# The Problem: Gaussian Process Regression

Measurements $\boldsymbol{y}_{\mathrm{Tr}}$ at $N$ points $X_{\mathrm{Tr}}$

Estimate unseen data $\boldsymbol{y}_{\mathrm{Pr}}$ at $X_{\mathrm{Pr}}$

Model as Gaussian process
$\rightarrow$ condition on $\boldsymbol{y}_{\mathrm{Tr}}$



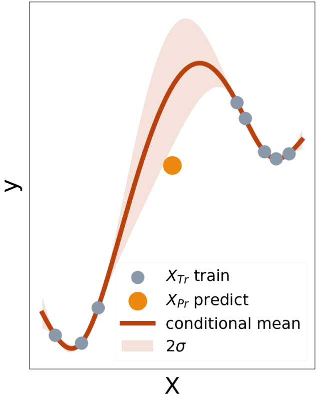
# The Problem: Gaussian Process Regression

Measurements $\boldsymbol{y}_{\text{Tr}}$ at $N$ points $X_{\text{Tr}}$

Estimate unseen data $\boldsymbol{y}_{\text{Pr}}$ at $X_{\text{Pr}}$

Model as Gaussian process
$\rightarrow$ condition on $\boldsymbol{y}_{\text{Tr}}$

Computational cost scales as $N^3$



# The Problem: Gaussian Process Regression

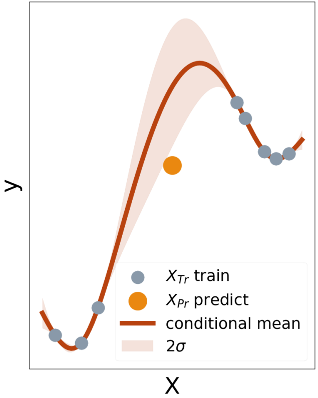
Measurements $\boldsymbol{y}_{\mathrm{Tr}}$ at $N$ points $X_{\mathrm{Tr}}$

Estimate unseen data $\boldsymbol{y}_{\mathrm{Pr}}$ at $X_{\mathrm{Pr}}$

Model as Gaussian process
$\rightarrow$ condition on $\boldsymbol{y}_{\mathrm{Tr}}$

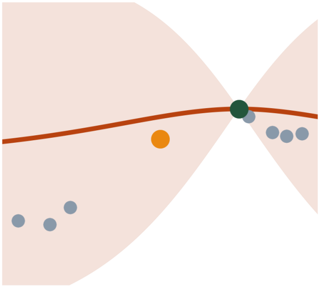
Computational cost scales as $N^3$

Choose $k$ most informative points!



# Conditional $k$-th Nearest Neighbors

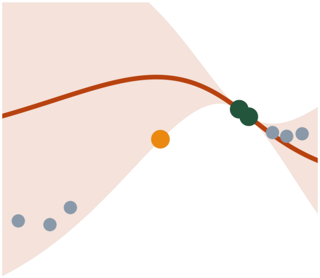
Naive: select $k$ closest points



# Conditional $k$-th Nearest Neighbors

Naive: select $k$ closest points

Chooses redundant information

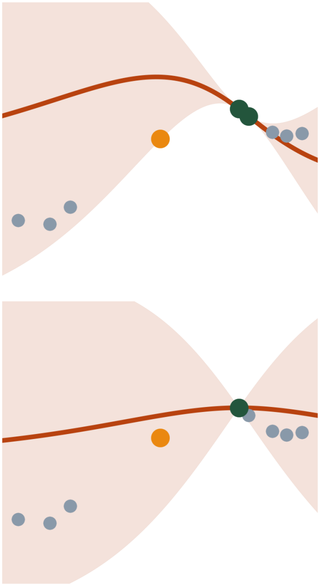


# Conditional $k$-th Nearest Neighbors

Naive: select $k$ closest points

Chooses redundant information

Maximize *mutual information*!

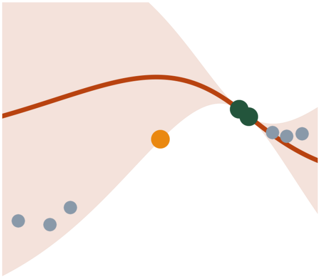


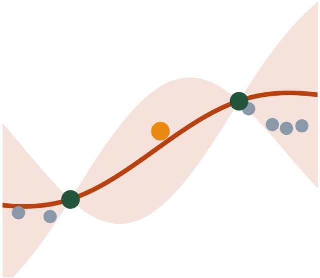
# Conditional $k$-th Nearest Neighbors

Naive: select $k$ closest points

Chooses redundant information

Maximize *mutual information*!





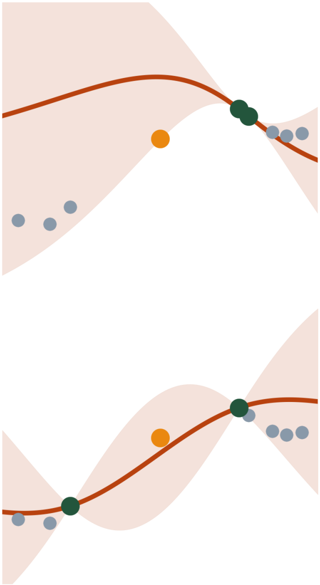
# Conditional $k$-th Nearest Neighbors

Naive: select $k$ closest points

Chooses redundant information

Maximize *mutual information*!

Direct computation: $\mathcal{O}(Nk^4)$



# Conditional $k$-th Nearest Neighbors

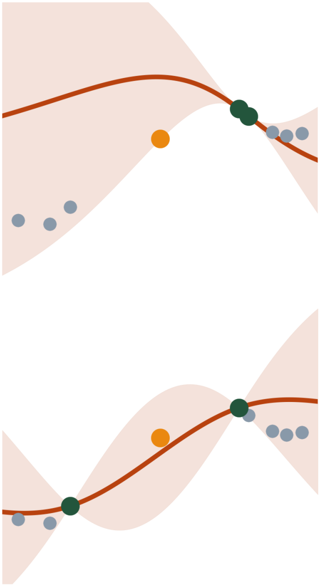
Naive: select $k$ closest points

Chooses redundant information

Maximize *mutual information*!

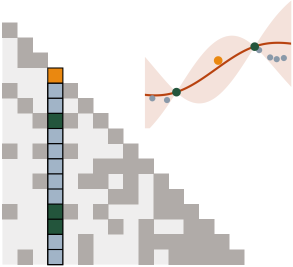
Direct computation: $\mathcal{O}(Nk^4)$

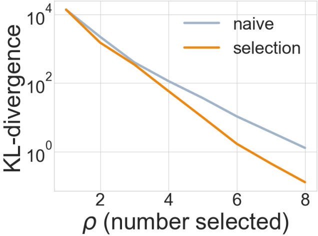
Store Cholesky factor $\to \mathcal{O}(Nk^2)$!



# Cholesky Factorization by Selection

Apply column-wise
→ sparse approx. of GP

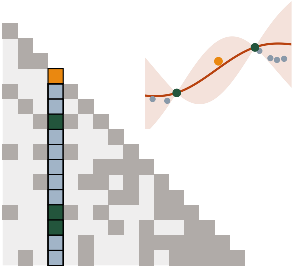


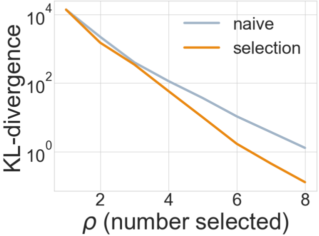


# Cholesky Factorization by Selection

Apply column-wise
$\rightarrow$ sparse approx. of GP

Maximum mutual information
$\rightarrow$ minimum KL-divergence



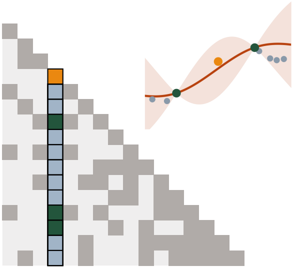


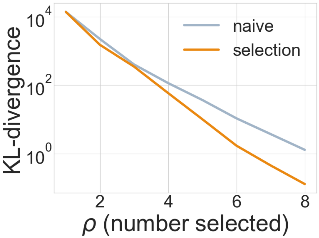
# Cholesky Factorization by Selection

Apply column-wise
$\rightarrow$ sparse approx. of GP

Maximum mutual information
$\rightarrow$ minimum KL-divergence

Improves approx. algorithm of [1]






[1] F. Schäfer, M. Katzfuss, and H. Owhadi, "Sparse Cholesky factorization by Kullback-Leibler minimization," *arXiv preprint arXiv:2004.14455*, 2020

# Table of Contents



# LU Decomposition

... and its symmetric counterpart

$M = LU$ where $L$ is lower triangular and $U$ is upper triangular

# LU Decomposition

... and its symmetric counterpart

$M = LU$ where $L$ is lower triangular and $U$ is upper triangular

Not always possible, need $PLU$ in general!

# LU Decomposition

... and its symmetric counterpart

$LU$ where $L$ is lower triangular and $U$ is upper triangular

...ways possible, need $PLU$ in general!

Special case for (square) symmetric matrices:

**Theorem**

*If $M = M^\top$ and $\det(M) \neq 0$, then $M = LDL^T$ where $L$ is from the LU decomposition of $M$ and $D$ is the diagonal of $U$.*

# LU Decomposition

... and its symmetric counterpart

$LU$ where $L$ is lower triangular and $U$ is upper triangular

Not always possible, need $PLU$ in general!

Special case for (square) symmetric matrices:

**Theorem**

*If $M = M^\top$ and $\det(M) \neq 0$, then $M = LDL^T$ where $L$ is from the LU decomposition of $M$ and $D$ is the diagonal of $U$.*

**Proof sketch.**

(MATH3406 Fall 2021, Prof. Wing Li) Let $M = LDK$. Just do matrix multiplication on $M = M^\top \implies (LDK) = (LDK)^T$. From matrix multiplication, able to see $K = L^\top$. □

# Cholesky Factorization

Let $M$ be (symmetric) *positive definite*.

# Cholesky Factorization

be (symmetric) *positive definite*.

Then $M = LDL^\top$ becomes $LL^\top$:

$$
\begin{aligned}
M &= LDL^\top \\
&= LD^{\frac{1}{2}}D^{\frac{1}{2}}L^\top \\
&= LD^{\frac{1}{2}}(LD^{\frac{1}{2}})^\top \\
&= L'L'^\top
\end{aligned}
$$

## Cholesky Factorization

$M$ be (symmetric) *positive definite*.

Then $M = LDL^\top$ becomes $LL^\top$:

$$
\begin{aligned}
M &= LDL^\top \\
&= LD^{\frac{1}{2}} D^{\frac{1}{2}} L^\top \\
&= LD^{\frac{1}{2}} (LD^{\frac{1}{2}})^\top \\
&= L' L'^\top
\end{aligned}
$$

This is the Cholesky factorization!

# Why Do We Care?

$\Theta = LL^\top$, $L$ has $N$ columns, $s$ non-zero entries per column

$L\boldsymbol{v}$ and $L^{-1}\boldsymbol{v}$ both cost $\mathcal{O}(Ns)$

Matrix-vector product $\Theta\boldsymbol{v} \to L(L^\top\boldsymbol{v})$

$N^2 \to Ns$

Solving linear system $\Theta^{-1}\boldsymbol{v} \to L^{-\top}(L^{-1}\boldsymbol{v})$

$N^3 \to Ns$

Log determinant $\operatorname{logdet}\Theta \to 2\operatorname{logdet}L = 2\sum_{i=1}^{N}\log L_{ii}$

$N^3 \to N$

Sampling from $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Theta) \to \boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I), \boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$

??? $\to Ns$

# Why Do We Care?

$\Theta = LL^\top$, $L$ has $N$ columns, $s$ non-zero entries per column

$L\boldsymbol{v}$ and $L^{-1}\boldsymbol{v}$ both cost $\mathcal{O}(Ns)$

Matrix-vector product $\Theta\boldsymbol{v} \to L(L^\top\boldsymbol{v})$

$N^2 \to Ns$

Solving linear system $\Theta^{-1}\boldsymbol{v} \to L^{-\top}(L^{-1}\boldsymbol{v})$

$N^3 \to Ns$

Log determinant $\operatorname{logdet}\Theta \to 2\operatorname{logdet} L = 2\sum_{i=1}^N \log L_{ii}$

$N^3 \to N$

Sampling from $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Theta) \to \boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, I), \boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$

??? $\to Ns$

# Computing the Cholesky Factorization

## Down-looking

Like LU

Gaussian elimination downwards

```
def down_cholesky(theta: np.ndarray) -> np.ndarray:
    M, n = np.copy(theta), len(theta)
    L = np.identity(n)
    for i in range(n):
        for j in range(i + 1, n):
            L[j, i] = M[j, i]/M[i, i]
            # zero out everything below
            M[j] -= L[j, i]*M[i]
        # update L
        L[:, i] *= np.sqrt(M[i, i])
    return L
```

# Computing the Cholesky Factorization

## Down-looking

Like LU

Gaussian elimination downwards

```python
def down_cholesky(theta: np.ndarray) -> np.
    M, n = np.copy(theta), len(theta)
    L = np.identity(n)
    for i in range(n):
        for j in range(i + 1, n):
            L[j, i] = M[j, i]/M[i, i]
            # zero out everything below
            M[j] -= L[j, i]*M[i]
        # update L
        L[:, i] *= np.sqrt(M[i, i])
    return L
```

# Computing the Cholesky Factorization

## Up-looking

Let $L'$ be blocked according to:

$$L' = \begin{pmatrix} L & \mathbf{0} \\ \boldsymbol{r}^\top & d \end{pmatrix}$$

$$L'L'^\top = \begin{pmatrix} L & \mathbf{0} \\ \boldsymbol{r}^\top & d \end{pmatrix} \begin{pmatrix} L^\top & \boldsymbol{r} \\ \mathbf{0}^\top & d \end{pmatrix}$$

$$= \begin{pmatrix} LL^\top & L\boldsymbol{r} \\ \boldsymbol{r}^\top L^\top & \boldsymbol{r}^\top \boldsymbol{r} + d^2 \end{pmatrix}$$

So if we have a Cholesky factor for a principle submatrix of $\Theta$, we can extend it inductively by reading off the appropiate data!

$$\begin{pmatrix} LL^\top & L\boldsymbol{r} \\ \boldsymbol{r}^\top L^\top & \boldsymbol{r}^\top \boldsymbol{r} + d^2 \end{pmatrix} = \begin{pmatrix} \Theta & \boldsymbol{c} \\ \boldsymbol{c}^\top & C \end{pmatrix}$$

$$\boldsymbol{r} = L^{-1}\boldsymbol{c}$$

$$d = \sqrt{C - \boldsymbol{r}^\top \boldsymbol{r}}$$

# Computing the Cholesky Factorization

## Up-looking

Let $L'$ be blocked according to:

$$L' = \begin{pmatrix} L & \mathbf{0} \\ \boldsymbol{r}^\top & d \end{pmatrix}$$

$$L'L'^\top = \begin{pmatrix} L & \mathbf{0} \\ \boldsymbol{r}^\top & d \end{pmatrix} \begin{pmatrix} L^\top & \boldsymbol{r} \\ \mathbf{0}^\top & d \end{pmatrix}$$

$$= \begin{pmatrix} LL^\top & L\boldsymbol{r} \\ \boldsymbol{r}^\top L^\top & \boldsymbol{r}^\top \boldsymbol{r} + d^2 \end{pmatrix}$$

So if we have a Cholesky factor for a principle submatrix of $\Theta$, we can extend it inductively by reading off the appropiate data!

$$\begin{pmatrix} LL^\top & L\boldsymbol{r} \\ \boldsymbol{r}^\top L^\top & \boldsymbol{r}^\top \boldsymbol{r} + d^2 \end{pmatrix} = \begin{pmatrix} \Theta & \boldsymbol{c} \\ \boldsymbol{c}^\top & C \end{pmatrix}$$

$$\boldsymbol{r} = L^{-1}\boldsymbol{c}$$

$$d = \sqrt{C - \boldsymbol{r}^\top \boldsymbol{r}}$$

# Computing the Cholesky Factorization

## Up-looking

```python
def Lsolve(L: np.ndarray, y: np.ndarray) -> np.ndarray:
    """ Solves Lx = y for lower triangular L. """
    n = len(y)
    x = np.zeros(n)
    for i in range(n):
        x[i] = (y[i] - L[i, :i].dot(x[:i]))/L[i, i]
    return x

def up_cholesky(theta: np.ndarray) -> np.ndarray:
    n = len(theta)
    L = np.zeros((n, n))
    for i in range(n):
        row = Lsolve(L, theta[:i, i])
        L[i, :i] = row
        L[i, i] = np.sqrt(theta[i, i] - row.dot(row))
    return L
```

# Computing the Cholesky Factorization

## Up-looking

```
def Lsolve(L: np.ndarray, y: np.ndarray) -> np.ndarray:
    """ Solves Lx = y for lower triangular L. """

    x       os(n)
            e(n):
            (y[i] - L[i, :i].dot(x[:i]))/L[i, i]
    re

def up_cholesky(theta: np.ndarray) -> np.ndarray:
    n = len(theta)
    L = np.zeros((n, n))
    for i in range(n):
        row = Lsolve(L, theta[:i, i])
        L[i, :i] = row
        L[i, i] = np.sqrt(theta[i, i] - row.dot(row))
    return L
```

# Computing the Cholesky Factorization

## Right-looking

$$L = \begin{pmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_N \end{pmatrix}$$

$$LL^\top = \begin{pmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_N \end{pmatrix} \begin{pmatrix} \boldsymbol{l}_1^\top \\ \boldsymbol{l}_2^\top \\ \vdots \\ \boldsymbol{l}_N^\top \end{pmatrix}$$

$$= \boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top = \Theta$$

From lower triangularity, nested submatrices!

# Computing the Cholesky Factorization

## Right-looking

$$L = \begin{pmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_N \end{pmatrix}$$

$$LL^\top = \begin{pmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_N \end{pmatrix} \begin{pmatrix} \boldsymbol{l}_1^\top \\ \boldsymbol{l}_2^\top \\ \vdots \\ \boldsymbol{l}_N^\top \end{pmatrix}$$

$$= \boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top = \Theta$$

From lower triangularity, nested submatrices!

# Computing the Cholesky Factorization

## Right-looking

$$
\begin{aligned}
\boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top &= \Theta \\
\boldsymbol{l}_1\boldsymbol{l}_1^\top &= \Theta_1 \\
l_1^2 &= \Theta_{11} \\
l_1 &= \sqrt{\Theta_{11}} \\
\boldsymbol{l}_1 &= \frac{\Theta_1}{l_1} = \frac{\Theta_1}{\sqrt{\Theta_{11}}} \\
\boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top &= \Theta - \left(\frac{\Theta_1}{\sqrt{\Theta_{11}}}\right)\left(\frac{\Theta_1}{\sqrt{\Theta_{11}}}\right)^\top \\
&= \Theta - \frac{\Theta_1\Theta_1^\top}{\Theta_{11}}
\end{aligned}
$$

Proceed inductively on rank-one update

# Computing the Cholesky Factorization

## Right-looking

$$
\begin{aligned}
\boldsymbol{l}_1 \boldsymbol{l}_1^\top + \boldsymbol{l}_2 \boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N \boldsymbol{l}_N^\top &= \Theta \\
\boldsymbol{l}_1 \boldsymbol{l}_1^\top &= \Theta_1 \\
l_1^2 &= \Theta_{11} \\
l_1 &= \sqrt{\Theta_{11}} \\
\boldsymbol{l}_1 &= \frac{\Theta_1}{l_1} = \frac{\Theta_1}{\sqrt{\Theta_{11}}} \\
\boldsymbol{l}_2 \boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N \boldsymbol{l}_N^\top &= \Theta - \left( \frac{\Theta_1}{\sqrt{\Theta_{11}}} \right) \left( \frac{\Theta_1}{\sqrt{\Theta_{11}}} \right)^\top \\
&= \Theta - \frac{\Theta_1 \Theta_1^\top}{\Theta_{11}}
\end{aligned}
$$

Proceed inductively on rank-one update

# Computing the Cholesky Factorization

## Right-looking

```python
def right_cholesky(theta: np.ndarray) -> np.ndarray:
    M, n = np.copy(theta), len(theta)
    L = np.zeros((n, n))
    for i in range(n):
        L[:, i] = M[:, i]/np.sqrt(M[i, i])
        M -= np.outer(L[:, i], L[:, i])
    return L
```

# Computing the Cholesky Factorization

## Left-looking

Recall:

$$\boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top = \Theta$$

Look at $\boldsymbol{l}_i$:

$$\begin{aligned}
\boldsymbol{l}_i\boldsymbol{l}_i^\top &= \left(\Theta - (\boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_{i-1}\boldsymbol{l}_{i-1}^\top)\right)_i \\
&= \Theta_i - (l_{1i}\boldsymbol{l}_1 + l_{2i}\boldsymbol{l}_2 + \cdots + l_{i-1,i}\boldsymbol{l}_{i-1}) \\
&= \Theta_i - \begin{pmatrix}\boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_{i-1}\end{pmatrix} \begin{pmatrix} l_{1i} \\ l_{2i} \\ \vdots \\ l_{i,i-1} \end{pmatrix} \\
&= \Theta_i - L_{:,:i}L_{i,:i}
\end{aligned}$$

Don't need to store modified $\Theta$ in memory!

# Computing the Cholesky Factorization

Left-looking

Recall:

$$\boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_N\boldsymbol{l}_N^\top = \Theta$$

Look at $\boldsymbol{l}_i$:

$$\begin{aligned}
\boldsymbol{l}_i\boldsymbol{l}_i^\top &= \left(\Theta - (\boldsymbol{l}_1\boldsymbol{l}_1^\top + \boldsymbol{l}_2\boldsymbol{l}_2^\top + \cdots + \boldsymbol{l}_{i-1}\boldsymbol{l}_{i-1}^\top)\right)_i \\
&= \Theta_i - (l_{1i}\boldsymbol{l}_1 + l_{2i}\boldsymbol{l}_2 + \cdots + l_{i-1,i}\boldsymbol{l}_{i-1}) \\
&= \Theta_i - \begin{pmatrix}\boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_{i-1}\end{pmatrix} \begin{pmatrix} l_{1i} \\ l_{2i} \\ \vdots \\ l_{i,i-1}\end{pmatrix} \\
&= \Theta_i - L_{:,:i}L_{i,:i}
\end{aligned}$$

Don't need to store modified $\Theta$ in memory!

# Computing the Cholesky Factorization

## Left-looking

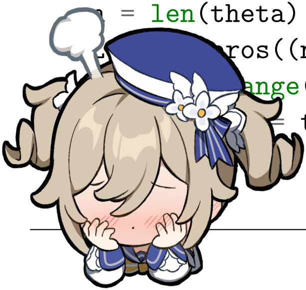
```python
def left_cholesky(theta: np.ndarray) -> np.ndarray:
    n = len(theta)
    L = np.zeros((n, n))
    for i in range(n):
        L[:, i] = theta[:, i] - L[:, :i]@L[i, :i]
        L[:, i] /= np.sqrt(L[i, i])
    return L
```

# Computing the Cholesky Factorization

## Left-looking

```
def left_cholesky(theta: np.ndarray) -> np.ndarray:
    n = len(theta)
    L   eros((n, n))
          ange(n):
              = theta[:, i] - L[:, :i]@L[i, :i]
                np.sqrt(L[i, i])

```



# Table of Contents



# Schur Complement

## or recursive Cholesky factorization

Block $\Theta$ as follows:

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$$

Then proceed by one step of Gaussian elimination:

$$\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \mathbf{0} & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix}$$

Thus,

$$= \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix} \begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

so we see the Cholesky factorization of $\Theta$ is

$$\begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \mathrm{chol}(\Theta_{11}) & 0 \\ 0 & \mathrm{chol}(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}) \end{pmatrix}$$

The term in blue is the *Schur complement* of $\Theta$ on $\Theta_{11}$

# Schur Complement

## or recursive Cholesky factorization

Block $\Theta$ as follows:

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$$

Then proceed by one step of Gaussian elimination:

$$\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \mathbf{0} & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix}$$

Thus,

$$= \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix} \begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

so we see the Cholesky factorization of $\Theta$ is

$$\begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \operatorname{chol}(\Theta_{11}) & 0 \\ 0 & \operatorname{chol}(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}) \end{pmatrix}$$

The term in blue is the *Schur complement* of $\Theta$ on $\Theta_{11}$

# Proper Determinant of Block Matrix

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$$

$$\begin{aligned} \det(\Theta) &= \text{?} \\ &= \det(\Theta_{11})\det(\Theta_{22}) - \det(\Theta_{21})\det(\Theta_{12})\text{?} && \text{wrong!} \\ &= \det(\Theta_{11}\Theta_{22} - \Theta_{21}\Theta_{12})\text{?} && \text{wrong!} \end{aligned}$$

Schur complement gives proper answer:

$$\Theta = \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix} \begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

$$\det(\Theta) = \det(\Theta_{11})\det(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12})$$

# Proper Determinant of Block Matrix

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$$

$$\det(\Theta) = ?$$

$$= \det(\Theta_{11})\det(\Theta_{22}) - \det(\Theta_{21})\det(\Theta_{12})? \qquad \text{wrong!}$$

$$= \det(\Theta_{11}\Theta_{22} - \Theta_{21}\Theta_{12})? \qquad \text{wrong!}$$

Schur complement gives proper answer:

$$\Theta = \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix}\begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix}\begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

$$\det(\Theta) \ldots \det(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12})$$

# Proper Submatrix of Inverse

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$$

$$(\Theta^{-1})_{22} = \text{?}$$

$$= (\Theta_{22})^{-1}\text{?} \qquad \text{wrong!}$$

Schur complement to the rescue again!

# Proper Submatrix of Inverse

$$\Theta = \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix} \begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

For notational convenience, we denote the Schur complement $\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}$ as $\Theta_{22|1}$. Inverting both sides of the equation,

$$\Theta^{-1} = \begin{pmatrix} I & -\Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix} \begin{pmatrix} \Theta_{11}^{-1} & 0 \\ 0 & \Theta_{22|1}^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -\Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix}$$

$$= \begin{pmatrix} \Theta_{11}^{-1} + \left(\Theta_{11}^{-1}\Theta_{12}\right)\Theta_{22|1}^{-1}\left(\Theta_{21}\Theta_{11}^{-1}\right) & -\left(\Theta_{11}^{-1}\Theta_{12}\right)\Theta_{22|1}^{-1} \\ -\Theta_{22|1}^{-1}\left(\Theta_{21}\Theta_{11}^{-1}\right) & \Theta_{22|1}^{-1} \end{pmatrix}$$

So $(\Theta^{-1})_{22}$ can be read off as $\Theta_{22|1}^{-1}$,

$$= \left(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}\right)^{-1}$$

# Proper Submatrix of Inverse

$$\Theta = \begin{pmatrix} I & 0 \\ \Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} \Theta_{11} & 0 \\ 0 & \Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12} \end{pmatrix} \begin{pmatrix} I & \Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix}$$

For notational convenience, we denote the Schur complement $\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}$ as $\Theta_{22|1}$. Inverting both sides of the equation,

$$\Theta^{-1} = \begin{pmatrix} I & -\Theta_{11}^{-1}\Theta_{12} \\ 0 & I \end{pmatrix} \begin{pmatrix} \Theta_{11}^{-1} & 0 \\ 0 & \Theta_{22|1}^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -\Theta_{21}\Theta_{11}^{-1} & I \end{pmatrix}$$

$$= \begin{pmatrix} \Theta_{11}^{-1} + (\Theta_{11}^{-1}\Theta_{12})\Theta_{22|1}^{-1}(\Theta_{21}\Theta_{11}^{-1}) & -(\Theta_{11}^{-1}\Theta_{12})\Theta_{22|1}^{-1} \\ -\Theta_{22|1}^{-1}(\Theta_{21}\Theta_{11}^{-1}) & \Theta_{22|1}^{-1} \end{pmatrix}$$

So $(\Theta^{-1})_{22}$ can be read off as $\Theta_{22|1}^{-1}$

$$= \left(\Theta_{22} - \Theta_{21}\Theta_{11}^{-1}\Theta_{12}\right)^{-1}$$

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

## A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

Is Schur complementing transitive?

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

Is Schur complementing transitive?

i.e. suppose we have $\Theta$ blocked as

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} & \Theta_{13} \\ \Theta_{21} & \Theta_{22} & \Theta_{23} \\ \Theta_{31} & \Theta_{32} & \Theta_{33} \end{pmatrix}$$

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

Is Schur complementing transitive?

i.e. suppose we have $\Theta$ blocked as

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} & \Theta_{13} \\ \Theta_{21} & \Theta_{22} & \Theta_{23} \\ \Theta_{31} & \Theta_{32} & \Theta_{33} \end{pmatrix}$$

Is $\Theta$ complemented on $\Theta_{11}$ and then on $\Theta_{22}$ the same as $\Theta$ complemented on $\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$?

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

Is Schur complementing transitive?

i.e. suppose we have $\Theta$ blocked as

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} & \Theta_{13} \\ \Theta_{21} & \Theta_{22} & \Theta_{23} \\ \Theta_{31} & \Theta_{32} & \Theta_{33} \end{pmatrix}$$

Is $\Theta$ complemented on $\Theta_{11}$ and then on $\Theta_{22}$ the same as $\Theta$ complemented on $\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$?

Intuitively, it should be, but tedious to prove

# A Few Important Questions...

Is the Schur complement symmetric positive definite (s.p.d.)?

If it isn't, we're kinda screwed — have been assuming so

Is Schur complementing transitive?

i.e. suppose we have $\Theta$ blocked as

$$\Theta = \begin{pmatrix} \Theta_{11} & \Theta_{12} & \Theta_{13} \\ \Theta_{21} & \Theta_{22} & \Theta_{23} \\ \Theta_{31} & \Theta_{32} & \Theta_{33} \end{pmatrix}$$

Is $\Theta$ complemented on $\Theta_{11}$ and then on $\Theta_{22}$ the same as $\Theta$ complemented on $\begin{pmatrix} \Theta_{11} & \Theta_{12} \\ \Theta_{21} & \Theta_{22} \end{pmatrix}$?

Intuitively, it should be, but tedious to prove

New perspective which changes everything!

# Table of Contents



# The Multivariate Gaussian

Recall: Gaussian (or normal) distribution:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

# The Multivariate Gaussian

Recall: Gaussian (or normal) distribution:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

Important (defining?) property: completely determined by mean and variance, all higher-order cumulants zero.

# The Multivariate Gaussian

Recall: Gaussian (or normal) distribution:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

Important (defining?) property: completely determined by mean and variance, all higher-order cumulants zero.

We're going to extend this to higher dimensions. Consider

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

where $\boldsymbol{x}$ ("variables") is a $N \times 1$ vector, $\boldsymbol{\mu}$ ("mean vector") is a $N \times 1$ vector, and $\Sigma$ ("covariance matrix") is a $N \times N$ matrix

# Defining Everything

Naturally,

$$
\begin{aligned}
\mu_i &= \mathrm{E}[x_i] \\
\boldsymbol{\mu} &= \mathrm{E}[\boldsymbol{x}] \\
\Sigma_{ij} &= \mathrm{Cov}[x_i, x_j] \\
&= \mathrm{E}[(x_i - \mathrm{E}[x_i])(x_j - \mathrm{E}[x_j])] \\
&= \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top]
\end{aligned}
$$

## Defining Everything

Naturally,

$$
\begin{aligned}
\mu_i &= \mathrm{E}[x_i] \\
\boldsymbol{\mu} &= \mathrm{E}[\boldsymbol{x}] \\
\Sigma_{ij} &= \mathrm{Cov}[x_i, x_j] \\
&= \mathrm{E}[(x_i - \mathrm{E}[x_i])(x_j - \mathrm{E}[x_j])] \\
&= \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top]
\end{aligned}
$$

Two natural (and fundamental) questions from here:

1. What is the probability density function $f(\boldsymbol{x})$?
2. How can we sample from $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$?

# Defining Everything

Naturally,

$$
\begin{aligned}
\mu_i &= \mathrm{E}[x_i] \\
\boldsymbol{\mu} &= \mathrm{E}[\boldsymbol{x}] \\
\Sigma_{ij} &= \mathrm{Cov}[x_i, x_j] \\
&= \mathrm{E}[(x_i - \mathrm{E}[x_i])(x_j - \mathrm{E}[x_j])] \\
&= \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top]
\end{aligned}
$$

Two natural (and fundamental) questions from here:

1. What is the probability density function $f(\boldsymbol{x})$?
2. How can we sample from $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$?

Surprisingly enough, Cholesky factorization answers both!

Naturally,

$$
\begin{aligned}
\mu_i &= \mathrm{E}[x_i] \\
\boldsymbol{\mu} &= \mathrm{E}[\boldsymbol{x}] \\
\Sigma_{ij} &= \mathrm{Cov}[x_i, x_j] \\
&= \mathrm{E}[(x_i - \mathrm{E}[x_i])(x_j - \mathrm{E}[x_j])] \\
&= \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top]
\end{aligned}
$$

Two natural (and fundamental) questions from here:

1. What is the probability density function $f(\boldsymbol{x})$?
2. How can we sample from $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$?

Surprisingly enough, Cholesky factorization answers both!

# Independent Variables

Gaussian has the (unique?) property if $\Sigma_{ij} = 0$, then $x_i$ and $x_j$ are statistically independent. This is not true in general!

# Independent Variables

Gaussian has the (unique?) property if $\Sigma_{ij} = 0$, then $x_i$ and $x_j$ are statistically independent. This is not true in general!

Key property we will make heavy use of: moment matching. If we know $\boldsymbol{\mu}$ and $\Sigma$, distribution is determined.

# Independent Variables

Gaussian has the (unique?) property if $\Sigma_{ij} = 0$, then $x_i$ and $x_j$ are statistically independent. This is not true in general!

Key property we will make heavy use of: moment matching. If we know $\boldsymbol{\mu}$ and $\Sigma$, distribution is determined.

Consider: if $x_i$ and $x_j$ *were* independent, then $\Sigma_{ij} = 0$. So suppose $x_i$ and $x_j$ are not independent but $\Sigma_{ij} = 0$. It's the same $\Sigma$ as when they were independent. So $x_i$ and $x_j$ must be distributed like they're independent. By contradiction, they must have been independent in the first place!

# Independent Variables

Gaussian has the (unique?) property if $\Sigma_{ij} = 0$, then $x_i$ and $x_j$ are statistically independent. This is not true in general!

Key property we will make heavy use of: moment matching. If we know $\boldsymbol{\mu}$ and $\Sigma$, distribution is determined.

Consider: if $x_i$ and $x_j$ *were* independent, then $\Sigma_{ij} = 0$. So suppose $x_i$ and $x_j$ are not independent but $\Sigma_{ij} = 0$. It's the same $\Sigma$ as when they were independent. So $x_i$ and $x_j$ must be distributed like they're independent. By contradiction, they must have been independent in the first place!

# Completely Independent Variables

Well, if $\Sigma$ has particular structure, it's actually trivial:

$$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$$

$$z_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$$

$$\begin{aligned} f(\boldsymbol{z}) &= \prod_{i=1}^{N} f(z_i) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} z_i^2} \\ &= \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}(z_1^2 + z_2^2 + \cdots + z_N^2)} \\ &= \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{z}} \end{aligned}$$

# Completely Independent Variables

Well, if $\Sigma$ has particular structure, it's actually trivial:

$$\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$$

$$z_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$$

$$\begin{aligned} f(\boldsymbol{z}) &= \prod_{i=1}^{N} f(z_i) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} z_i^2} \\ &= \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}(z_1^2 + z_2^2 + \cdots + z_N^2)} \\ &= \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{z}} \end{aligned}$$

## Moment Matching

How can we generalize to arbitrary $\Sigma$?

Moment match!

$$\begin{aligned}
\boldsymbol{z} &\sim \mathcal{N}(\mathbf{0}, I_N) \\
\boldsymbol{x} &= L\boldsymbol{z} + \boldsymbol{\mu} \\
\mathrm{E}[\boldsymbol{x}] &= \mathrm{E}[L\boldsymbol{z} + \boldsymbol{\mu}] = L\,\mathrm{E}[\boldsymbol{z}] + \boldsymbol{\mu} = \boldsymbol{\mu} \\
\mathrm{Cov}[\boldsymbol{x}] &= \mathrm{E}[(\boldsymbol{x} - \mathrm{E}[\boldsymbol{x}])(\boldsymbol{x} - \mathrm{E}[\boldsymbol{x}])^\top] \\
&= \mathrm{E}[L\boldsymbol{z}(L\boldsymbol{z})^\top] \\
&= \mathrm{E}[L\boldsymbol{z}\boldsymbol{z}^\top L^\top] \\
&= L\,\mathrm{E}[\boldsymbol{z}\boldsymbol{z}^\top]L^\top \\
&= LL^\top
\end{aligned}$$

so $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, LL^\top)$. We want $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, so $\Sigma = LL^\top$

# Moment Matching

How can we generalize to arbitrary $\Sigma$?

Moment match!

$$
\begin{aligned}
\boldsymbol{z} &\sim \mathcal{N}(\mathbf{0}, I_N) \\
\boldsymbol{x} &= L\boldsymbol{z} + \boldsymbol{\mu} \\
\mathrm{E}[\boldsymbol{x}] &= \mathrm{E}[L\boldsymbol{z} + \boldsymbol{\mu}] = L\,\mathrm{E}[\boldsymbol{z}] + \boldsymbol{\mu} = \boldsymbol{\mu} \\
\mathrm{Cov}[\boldsymbol{x}] &= \mathrm{E}[(\boldsymbol{x} - \mathrm{E}[\boldsymbol{x}])(\boldsymbol{x} - \mathrm{E}[\boldsymbol{x}])^\top] \\
&= \mathrm{E}[L\boldsymbol{z}(L\boldsymbol{z})^\top] \\
&= \mathrm{E}[L\boldsymbol{z}\boldsymbol{z}^\top L^\top] \\
&= L\,\mathrm{E}[\boldsymbol{z}\boldsymbol{z}^\top]L^\top \\
&= LL^\top
\end{aligned}
$$

so $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, LL^\top)$. We want $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, so $\Sigma = LL^\top$

# Sampling with Cholesky Factorization

As we just saw, we can sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by instead sampling $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$ and computing $\boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$.

# Sampling with Cholesky Factorization

As we just saw, we can sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by instead sampling $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$ and computing $\boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$.

Since $LL^\top = \Sigma$, a natural pick is $L = \mathrm{chol}(\Sigma)$.

# Sampling with Cholesky Factorization

As we just saw, we can sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by instead sampling $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$ and computing $\boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$.

Since $LL^\top = \Sigma$, a natural pick is $L = \mathrm{chol}(\Sigma)$.

Why is $\Sigma$ s.p.d.? Because it's a covariance/Gram matrix!

$$\begin{aligned}
\Sigma &= \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top] \\
\boldsymbol{y}^\top \Sigma \boldsymbol{y} &= \boldsymbol{y}^\top \mathrm{E}[(\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top]\boldsymbol{y} \\
&= \mathrm{E}[\boldsymbol{y}^\top (\boldsymbol{x} - \boldsymbol{\mu})(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{y}] \\
&= \mathrm{E}[((\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{y})^\top (\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{y}] \\
&= \mathrm{E}[\|(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{y}\|^2] \geq 0
\end{aligned}$$

# Sampling with Cholesky Factorization

As we just saw, we can sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by instead sampling $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_N)$ and computing $\boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$.

Since $LL^\top = \Sigma$, a natural pick is $L = \operatorname{chol}(\Sigma)$.

Why is $\Sigma$ s.p.d.? Because it's a covariance/Gram matrix!

$$\begin{aligned}
\Sigma &= \mathrm{E}[(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top] \\
\boldsymbol{y}^\top \Sigma \boldsymbol{y} &= \boldsymbol{y}^\top \mathrm{E}[(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top]\boldsymbol{y} \\
&= \mathrm{E}[\boldsymbol{y}^\top(\boldsymbol{x}-\boldsymbol{\mu})(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{y}] \\
&= \mathrm{E}[((\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{y})^\top (\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{y}] \\
&= \mathrm{E}[\|(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{y}\|^2] \geq 0
\end{aligned}$$

# Probability Density Function from Sampling

What's the probability density function $f(\boldsymbol{x})$?

# Probability Density Function from Sampling

What's the probability density function $f(\boldsymbol{x})$?

Idea: view $\boldsymbol{x}$ resulting from a invertible transformation from $\boldsymbol{z}$.

# Probability Density Function from Sampling

What's the probability density function $f(\boldsymbol{x})$?

Idea: view $\boldsymbol{x}$ resulting from a invertible transformation from $\boldsymbol{z}$.

We know $f(\boldsymbol{z})$, so $f(\boldsymbol{x})$ should be similar!

# Probability Density Function from Sampling

What's the probability density function $f(\boldsymbol{x})$?

Idea: view $\boldsymbol{x}$ resulting from a invertible transformation from $\boldsymbol{z}$.

We know $f(\boldsymbol{z})$, so $f(\boldsymbol{x})$ should be similar!

In scalars:

$$
\begin{aligned}
z &\sim \mathcal{N}(0, 1) \\
x &= \sigma z + \mu \\
x &\sim \mathcal{N}(\mu, \sigma^2) \\
z &= \frac{x - \mu}{\sigma}
\end{aligned}
$$

# PDF from Sampling — Scalar Edition

Since $f(z)$ is a valid probability density function,

$$1 = \int_{-\infty}^{\infty} f(z)\,\mathrm{d}z = \int_{-\infty}^{\infty} f(z)\frac{\mathrm{d}z}{\mathrm{d}x}\,\mathrm{d}x$$

We now perform the change of variables $z = \frac{x-\mu}{\sigma}$

$$= \int_{-\infty}^{\infty} \underbrace{f\left(\frac{x-\mu}{\sigma}\right)\frac{1}{\sigma}}_{\text{PDF of } x}\,\mathrm{d}x$$

$$f(z) = \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}z^2}$$

$$\frac{1}{\sigma}f\left(\frac{x-\mu}{\sigma}\right) = \frac{1}{\sigma}\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

# PDF from Sampling — Vector Edition

$$\boldsymbol{x} = L\boldsymbol{z} + \boldsymbol{\mu}$$
$$\boldsymbol{z} = L^{-1}(\boldsymbol{x} - \boldsymbol{\mu})$$

Since $f(\boldsymbol{z})$ is a valid probability density function,

$$\begin{aligned}
1 &= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(\boldsymbol{z}) \, \mathrm{d}\boldsymbol{z} \\
&= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(\boldsymbol{z}) \frac{\mathrm{d}\boldsymbol{z}}{\mathrm{d}\boldsymbol{x}} \, \mathrm{d}\boldsymbol{x} && \text{(informal)} \\
&= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f(\boldsymbol{z}) |\det(J_{\boldsymbol{z}})| \, \mathrm{d}\boldsymbol{x} && \text{(formal)} \\
&= \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \underbrace{f(L^{-1}(\boldsymbol{x} - \boldsymbol{\mu})) \det(L^{-1})}_{\text{PDF of } \boldsymbol{x}} \, \mathrm{d}\boldsymbol{x}
\end{aligned}$$

# PDF from Sampling — Vector Edition

$$f(\boldsymbol{z}) = \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{z}}$$

Expanding $\det(L^{-1}) f(L^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))$,

$$= \frac{1}{\det(L)} f(L^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))$$

$$= \frac{1}{\det(L)} \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}(L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))^\top (L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))}$$

Since $LL^\top = \Sigma$, $\det(\Sigma) = \det(L)^2$

$$= \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top L^{-T} L^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

$$= \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

# PDF from Sampling — Vector Edition

$$f(\boldsymbol{z}) = \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}\boldsymbol{z}^\top \boldsymbol{z}}$$

Expanding $\det(L^{-1}) f(L^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))$,

$$= \frac{1}{\det(L)} f(L^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))$$

$$= \frac{1}{\det(L)} \frac{1}{\sqrt{(2\pi)^N}} e^{-\frac{1}{2}(L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))^\top (L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))}$$

Since $LL^\top = \Sigma$, $\det(\Sigma) = \det(L)^2$

$$= \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top L^{-T} L^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

$$= \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1} (\boldsymbol{x}-\boldsymbol{\mu})}$$

Compare PDFs of multivariate normal and scalar normal:

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$
$$f(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

Compare to scalar:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

Compare PDFs of multivariate normal and scalar normal:

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

$$f(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

Compare to scalar:

$$x \sim \mathcal{N}(\mu, \sigma^2)$$

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

Remarkable similarity!

# Cholesky Factorization for Gaussians

Sampling: $\boldsymbol{x} = L\boldsymbol{z} + \mu$, matrix-vector product, $\mathcal{O}(Ns)$

Density computation:

$$\begin{aligned}(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) &= (\boldsymbol{x}-\boldsymbol{\mu})^\top L^{-\top} L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \\ &= (L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))^\top L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \\ &= \|L^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\|^2\end{aligned}$$

Back-substitution, $O(Ns)$

# Cholesky Factorization for Gaussians

Sampling: $\boldsymbol{x} = L\boldsymbol{z} + \mu$, matrix-vector product, $\mathcal{O}(Ns)$

Density computation:

$$
\begin{aligned}
(\boldsymbol{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) &= (\boldsymbol{x}-\boldsymbol{\mu})^\top L^{-\top} L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \\
&= (L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}))^\top L^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \\
&= \|L^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\|^2
\end{aligned}
$$

Back-substitution, $O(Ns)$

# Closure of Multivariate Gaussians

Many statistical operations preserve distribution

# Closure of Multivariate Gaussians

Many statistical operations preserve distribution

Affine transformation

# Closure of Multivariate Gaussians

Many statistical operations preserve distribution

Affine transformation

Joint distribution & marginalization:

$$\begin{aligned} \boldsymbol{x}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_{11}) \\ \boldsymbol{x}_2 &\sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_{22}) \\ \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{pmatrix} &\sim \mathcal{N}\left( \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \right) \end{aligned}$$

# Closure of Multivariate Gaussians

Many statistical operations preserve distribution

Affine transformation

Joint distribution & marginalization:

$$
\begin{aligned}
\boldsymbol{x}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_{11}) \\
\boldsymbol{x}_2 &\sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_{22}) \\
\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{pmatrix} &\sim \mathcal{N}\left( \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \right)
\end{aligned}
$$

Conditioning

# Closure of Multivariate Gaussians

Many statistical operations preserve distribution

Affine transformation

Joint distribution & marginalization:

$$\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_{11})$$
$$\boldsymbol{x}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_{22})$$
$$\begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}\right)$$

Conditioning

# Conditioning

Assume $\boldsymbol{\mu} = \mathbf{0}$ and use precision instead of covariance!

$$Q = \Sigma^{-1} = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix}$$

$$\begin{aligned} \pi(\boldsymbol{x}_2 \mid \boldsymbol{x}_1) &= \frac{\pi(\boldsymbol{x}_1 \mid \boldsymbol{x}_2)\pi(\boldsymbol{x}_2)}{\pi(\boldsymbol{x}_1)} = \frac{\pi(\boldsymbol{x}_1, \boldsymbol{x}_2)}{\pi(\boldsymbol{x}_1)} \\ &\propto \pi(\boldsymbol{x}_1, \boldsymbol{x}_2) \\ &\propto e^{-\frac{1}{2}\boldsymbol{x}_2^\top Q_{22}\boldsymbol{x}_2 - (Q_{21}\boldsymbol{x}_1)^\top \boldsymbol{x}_2} \end{aligned}$$

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(-Q_{22}^{-1}Q_{21}\boldsymbol{x}_1, Q_{22}^{-1}\right)$$

If $\boldsymbol{\mu} \neq \mathbf{0}$, shift $\boldsymbol{x}^* = \boldsymbol{x} - \boldsymbol{\mu}$, $\mathrm{E}[\boldsymbol{x}^*] = \mathbf{0}$

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 - Q_{22}^{-1}Q_{21}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), Q_{22}^{-1}\right)$$

# Conditioning with Schur Complements

$$
\begin{aligned}
\boldsymbol{x}_2 \mid \boldsymbol{x}_1 &\sim \mathcal{N}\left(\boldsymbol{\mu}_2 - Q_{22}^{-1} Q_{21}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), Q_{22}^{-1}\right) \\
Q &= \Sigma^{-1} = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \\
&= \begin{pmatrix} \Sigma_{11}^{-1} + \left(\Sigma_{11}^{-1}\Sigma_{12}\right)\Sigma_{22|1}^{-1}\left(\Sigma_{21}\Sigma_{11}^{-1}\right) & -\left(\Sigma_{11}^{-1}\Sigma_{12}\right)\Sigma_{22|1}^{-1} \\ -\Sigma_{22|1}^{-1}\left(\Sigma_{21}\Sigma_{11}^{-1}\right) & \Sigma_{22|1}^{-1} \end{pmatrix} \\
Q_{22}^{-1} &= (\Sigma_{22|1}^{-1})^{-1} = \Sigma_{22|1} \\
&= \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12} \\
Q_{22}^{-1} Q_{21} &= -\Sigma_{22|1}(\Sigma_{22|1}^{-1}\Sigma_{21}\Sigma_{11}^{-1}) \\
&= -\Sigma_{21}\Sigma_{11}^{-1} \\
\boldsymbol{x}_2 \mid \boldsymbol{x}_1 &\sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)
\end{aligned}
$$

# Conditioning with Schur Complements

$$
\begin{aligned}
\boldsymbol{x}_2 \mid \boldsymbol{x}_1 &\sim \mathcal{N}\left(\boldsymbol{\mu}_2 - Q_{22}^{-1} Q_{21}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), Q_{22}^{-1}\right) \\
Q = \Sigma^{-1} &= \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \\
&= \begin{pmatrix} \Sigma_{11}^{-1} + \left(\Sigma_{11}^{-1}\Sigma_{12}\right)\Sigma_{22|1}^{-1}\left(\Sigma_{21}\Sigma_{11}^{-1}\right) & -\left(\Sigma_{11}^{-1}\Sigma_{12}\right)\Sigma_{22|1}^{-1} \\ -\Sigma_{22|1}^{-1}\left(\Sigma_{21}\Sigma_{11}^{-1}\right) & \Sigma_{22|1}^{-1} \end{pmatrix} \\
Q_{22}^{-1} &= (\Sigma_{22|1}^{-1})^{-1} = \Sigma_{22|1} \\
&= \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12} \\
Q_{22}^{-1} Q_{21} &= -\Sigma_{22|1}(\Sigma_{22|1}^{-1}\Sigma_{21}\Sigma_{11}^{-1}) \\
&= -\Sigma_{21}\Sigma_{11}^{-1} \\
\boldsymbol{x}_2 \mid \boldsymbol{x}_1 &\sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)
\end{aligned}
$$

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

Schur complement $\iff$ conditional covariance!

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

Schur complement $\iff$ conditional covariance!

s.p.d. because covariance matrices s.p.d.

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

Schur complement $\iff$ conditional covariance!

s.p.d. because covariance matrices s.p.d.

Quotient rule statistically trivial:
$\pi((x_1 \mid x_2) \mid x_3) = \pi(x_1 \mid x_2, x_3)$

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

Schur complement $\iff$ conditional covariance!

s.p.d. because covariance matrices s.p.d.

Quotient rule statistically trivial:
$\pi((x_1 \mid x_2) \mid x_3) = \pi(x_1 \mid x_2, x_3)$

Conditioning in covariance $\iff$ marginalization in precision

# Statistical Interpretation

From conditioning,

$$\boldsymbol{x}_2 \mid \boldsymbol{x}_1 \sim \mathcal{N}\left(\boldsymbol{\mu}_2 + \Sigma_{21}\Sigma_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1), \Sigma_{22} - \Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\right)$$

Schur complement $\iff$ conditional covariance!

s.p.d. because covariance matrices s.p.d.

Quotient rule statistically trivial:
$\pi((x_1 \mid x_2) \mid x_3) = \pi(x_1 \mid x_2, x_3)$

Conditioning in covariance $\iff$ marginalization in precision

# Table of Contents



Probability distribution over *vectors*

# Gaussian Processes

Probability distribution over *vectors*

Extend to distribution over *functions*?

Probability distribution over *vectors*

Extend to distribution over *functions*?

Idea: for finite set of points, function simply vector

$$X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$$
$$\boldsymbol{y} = \{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_N)\}$$

Probability distribution over *vectors*

Extend to distribution over *functions*?

Idea: for finite set of points, function simply vector

$$X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$$
$$\boldsymbol{y} = \{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_N)\}$$

Idea: for points we're not given, marginalization is trivial

# Gaussian Processes

Probability distribution over *vectors*

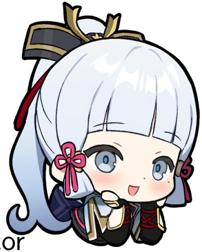
Extend to distribution over *functions*?

Idea: for finite set of points, function simply vector

$$X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$$
$$\boldsymbol{y} = \{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_N)\}$$

Idea: for points we're not given, marginalization is trivial

How to assign mean and covariance in a sensible way?



Probability distribution over *vectors*

Extend to distribution over *functions*?

Idea: for finite set of points, function simply vector

$$X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$$
$$\boldsymbol{y} = \{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \dots, f(\boldsymbol{x}_N)\}$$

Idea: for points we're not given, marginalization is trivial

How to assign mean and covariance in a sensible way?

# Gaussian Process Definition

Let $\mu(\boldsymbol{x})$ be the *mean function* and
$K(\boldsymbol{x}, \boldsymbol{x}')$ be the *covariance function* or *kernel function*

We say

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), K(\boldsymbol{x}, \boldsymbol{x}'))$$

If for all point sets $X$,

$$\begin{aligned} X &= \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\} \\ \boldsymbol{y} &= \{f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \ldots, f(\boldsymbol{x}_N)\} \\ \boldsymbol{y} &\sim \mathcal{N}(\boldsymbol{\mu}, \Theta) \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{\mu}_i &= \mu(\boldsymbol{x}_i) \\ \Theta_{ij} &= K(\boldsymbol{x}_i, \boldsymbol{x}_j) \end{aligned}$$

# Regression with Gaussian Processes

Simply condition prediction points on training points:

$$\Theta = \begin{pmatrix} \Theta_{\mathsf{Tr},\mathsf{Tr}} & \Theta_{\mathsf{Tr},\mathsf{Pr}} \\ \Theta_{\mathsf{Pr},\mathsf{Tr}} & \Theta_{\mathsf{Pr},\mathsf{Pr}} \end{pmatrix}$$

$$\mathrm{E}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] = \boldsymbol{\mu}_{\mathsf{Pr}} + \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} (\boldsymbol{y}_{\mathsf{Tr}} - \boldsymbol{\mu}_{\mathsf{Tr}})$$

$$\mathrm{Cov}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] = \Theta_{\mathsf{Pr},\mathsf{Pr}} - \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} \Theta_{\mathsf{Tr},\mathsf{Pr}}$$

# Regression with Gaussian Processes

Simply condition prediction points on training points:

$$
\begin{aligned}
\Theta &= \begin{pmatrix} \Theta_{\mathsf{Tr},\mathsf{Tr}} & \Theta_{\mathsf{Tr},\mathsf{Pr}} \\ \Theta_{\mathsf{Pr},\mathsf{Tr}} & \Theta_{\mathsf{Pr},\mathsf{Pr}} \end{pmatrix} \\
\mathrm{E}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] &= \boldsymbol{\mu}_{\mathsf{Pr}} + \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} (\boldsymbol{y}_{\mathsf{Tr}} - \boldsymbol{\mu}_{\mathsf{Tr}}) \\
\mathrm{Cov}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] &= \Theta_{\mathsf{Pr},\mathsf{Pr}} - \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} \Theta_{\mathsf{Tr},\mathsf{Pr}}
\end{aligned}
$$

Nonparametric! No training! Uncertainty quantification!

# Regression with Gaussian Processes

Simply condition prediction points on training points:

$$\Theta = \begin{pmatrix} \Theta_{\text{Tr},\text{Tr}} & \Theta_{\text{Tr},\text{Pr}} \\ \Theta_{\text{Pr},\text{Tr}} & \Theta_{\text{Pr},\text{Pr}} \end{pmatrix}$$

$$\mathrm{E}[\boldsymbol{y}_{\text{Pr}} \mid \boldsymbol{y}_{\text{Tr}}] = \boldsymbol{\mu}_{\text{Pr}} + \Theta_{\text{Pr},\text{Tr}}\Theta_{\text{Tr},\text{Tr}}^{-1}(\boldsymbol{y}_{\text{Tr}} - \boldsymbol{\mu}_{\text{Tr}})$$

$$\mathrm{Cov}[\boldsymbol{y}_{\text{Pr}} \mid \boldsymbol{y}_{\text{Tr}}] = \Theta_{\text{Pr},\text{Pr}} - \Theta_{\text{Pr},\text{Tr}}\Theta_{\text{Tr},\text{Tr}}^{-1}\Theta_{\text{Tr},\text{Pr}}$$

Nonparametric! No training! Uncertainty quantification!

... $\mathcal{O}(N^3)$ to compute $\Theta_{\text{Tr},\text{Tr}}^{-1}$

# Regression with Gaussian Processes

Simply condition prediction points on training points:

$$\Theta = \begin{pmatrix} \Theta_{\mathsf{Tr},\mathsf{Tr}} & \Theta_{\mathsf{Tr},\mathsf{Pr}} \\ \Theta_{\mathsf{Pr},\mathsf{Tr}} & \Theta_{\mathsf{Pr},\mathsf{Pr}} \end{pmatrix}$$

$$\mathrm{E}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] = \boldsymbol{\mu}_{\mathsf{Pr}} + \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} (\boldsymbol{y}_{\mathsf{Tr}} - \boldsymbol{\mu}_{\mathsf{Tr}})$$

$$\mathrm{Cov}[\boldsymbol{y}_{\mathsf{Pr}} \mid \boldsymbol{y}_{\mathsf{Tr}}] = \Theta_{\mathsf{Pr},\mathsf{Pr}} - \Theta_{\mathsf{Pr},\mathsf{Tr}} \Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1} \Theta_{\mathsf{Tr},\mathsf{Pr}}$$

Nonparametric! No training! Uncertainty quantification!

... $\mathcal{O}(N^3)$ to compute $\Theta_{\mathsf{Tr},\mathsf{Tr}}^{-1}$

And we're back to the starting problem

# Screening Effect

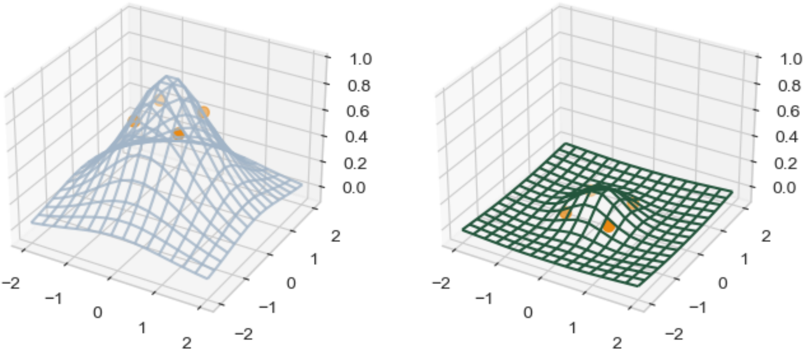



Figure: Conditional on nearby points, far away points have less covariance

# Table of Contents



# Cholesky Factorization by KL Minimization

Measure approximation error by KL-divergence:

$$L := \underset{\hat{L} \in S}{\operatorname{argmin}} \; \mathbb{D}_{\mathsf{KL}} \left( \mathcal{N}(\mathbf{0}, \Theta) \, \middle\| \, \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1}) \right)$$

# Cholesky Factorization by KL Minimization

Measure approximation error by KL-divergence:

$$L := \underset{\hat{L} \in S}{\operatorname{argmin}} \, \mathbb{D}_{\mathsf{KL}} \left( \mathcal{N}(\mathbf{0}, \Theta) \middle\| \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1}) \right)$$

Re-write KL-divergence:

$$2\mathbb{D}_{\mathsf{KL}} \left( \mathcal{N}(\mathbf{0}, \Theta_1) \middle\| \mathcal{N}(\mathbf{0}, \Theta_2) \right) =$$
$$\operatorname{trace}(\Theta_2^{-1}\Theta_1) + \operatorname{logdet}(\Theta_2) - \operatorname{logdet}(\Theta_1) - N$$

where $\Theta_1$ and $\Theta_2$ are both of size $N \times N$

# Cholesky Factorization by KL Minimization

Measure approximation error by KL-divergence:

$$L \coloneqq \underset{\hat{L} \in S}{\operatorname{argmin}} \, \mathbb{D}_{\mathsf{KL}} \left( \mathcal{N}(\mathbf{0}, \Theta) \,\middle\|\, \mathcal{N}(\mathbf{0}, (\hat{L}\hat{L}^\top)^{-1}) \right)$$

Re-write KL-divergence:

$$2\mathbb{D}_{\mathsf{KL}} \left( \mathcal{N}(\mathbf{0}, \Theta_1) \,\middle\|\, \mathcal{N}(\mathbf{0}, \Theta_2) \right) =$$

$$\operatorname{trace}(\Theta_2^{-1}\Theta_1) + \operatorname{logdet}(\Theta_2) - \operatorname{logdet}(\Theta_1) - N$$

where $\Theta_1$ and $\Theta_2$ are both of size $N \times N$

# Cholesky Factorization as GP Regression

## Theorem

*[1]. The non-zero entries of the $i$th column of $L$ are:*

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

# Cholesky Factorization as GP Regression

## Theorem

*[1]. The non-zero entries of the $i$th column of $L$ are:*

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

Plugging the optimal $L$ back into the KL-divergence, we obtain:

$$\sum_{i=1}^{N} \left[ \log \left( (\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1)^{-1} \right) \right] - \operatorname{logdet}(\Theta)$$

# Cholesky Factorization as GP Regression

## Theorem

*[1]. The non-zero entries of the $i$th column of $L$ are:*

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

Plugging the optimal $L$ back into the KL-divergence, we obtain:

$$\sum_{i=1}^{N} \left[ \log \left( (\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1)^{-1} \right) \right] - \operatorname{logdet}(\Theta)$$

But marginalization in covariance is conditioning in precision!

$$(\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1)^{-1} = \Theta_{ii|s_i - \{i\}}$$

# Cholesky Factorization as GP Regression

**Theorem**

*[1]. The non-zero entries of the $i$th column of $L$ are:*

$$L_{s_i,i} = \frac{\Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}{\sqrt{\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1}}$$

Plugging the optimal $L$ back into the KL-di

$$\sum_{i=1}^{N} \left[ \log \left( (\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1)^{-1} \right) \right] - \mathrm{l}$$

But marginalization in covariance is conditioning in precision!

$$(\boldsymbol{e}_1^\top \Theta_{s_i,s_i}^{-1} \boldsymbol{e}_1)^{-1} = \Theta_{ii|s_i - \{i\}}$$

This is precisely sparse Gaussian process regression!

# Table of Contents



# References

[1] F. Schäfer, M. Katzfuss, and H. Owhadi, "Sparse Cholesky factorization by Kullback-Leibler minimization," *arXiv preprint arXiv:2004.14455*, 2020.

# Thank You!