Deep Generative Models

Fall 2025

Sharif University of Technology

Alireza Mirrokni - 401106617

Problem Set 1

Problem 1: Properties of Gaussian Distributions

We first find the Moment Generating Function (MGF) of a Gaussian random vector. Let $\boldsymbol{X} \in \mathbb{R}^d$ be Gaussian with mean $\boldsymbol{\mu}$ and covariance Σ. Choose a matrix $L \in \mathbb{R}^{d\times d}$ such that $LL^\top = \Sigma$ and let $\boldsymbol{Z} \sim \mathcal{N}(\mathbf{0}, I_d)$ with independent coordinates. Then $\boldsymbol{X}$ has the representation $\boldsymbol{X} \overset{d}{=} \boldsymbol{\mu} + L\boldsymbol{Z}$. For any $\boldsymbol{t} \in \mathbb{R}^d$,

$$\begin{aligned} M_{\boldsymbol{X}}(\boldsymbol{t}) &= \mathbb{E}\left[\exp\left(\boldsymbol{t}^\top \boldsymbol{X}\right)\right] = \mathbb{E}\left[\exp\left(\boldsymbol{t}^\top(\boldsymbol{\mu} + L\boldsymbol{Z})\right)\right] \\ &= \exp\left(\boldsymbol{t}^\top \boldsymbol{\mu}\right) \mathbb{E}\left[\exp\left(\left(L^\top \boldsymbol{t}\right)^\top \boldsymbol{Z}\right)\right]. \end{aligned}$$

Let $\boldsymbol{u} = L^\top \boldsymbol{t}$. Since $Z_1, \ldots, Z_d$ are independent standard normals,

$$\mathbb{E}\left[\exp\left(\boldsymbol{u}^\top \boldsymbol{Z}\right)\right] = \mathbb{E}\left[\exp\left(\sum_{i=1}^d u_i Z_i\right)\right] = \prod_{i=1}^d \mathbb{E}\left[\exp\left(u_i Z_i\right)\right].$$

For scalar $Z \sim \mathcal{N}(0,1)$ and $u \in \mathbb{R}$,

$$\begin{aligned} \mathbb{E}\left[\exp\left(uZ\right)\right] &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(uz - \frac{1}{2}z^2\right) dz \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}(z-u)^2 + \frac{1}{2}u^2\right) dz = \exp\left(\frac{1}{2}u^2\right), \end{aligned}$$

because $\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2}(z-u)^2\right) dz = 1$. Therefore

$$\mathbb{E}\left[\exp\left(\boldsymbol{u}^\top \boldsymbol{Z}\right)\right] = \prod_{i=1}^d \exp\left(\frac{1}{2}u_i^2\right) = \exp\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{u}\right).$$

Substituting $\boldsymbol{u} = L^\top \boldsymbol{t}$ yields

$$M_{\boldsymbol{X}}(\boldsymbol{t}) = \exp\left(\boldsymbol{t}^\top \boldsymbol{\mu}\right) \exp\left(\frac{1}{2}\boldsymbol{t}^\top LL^\top \boldsymbol{t}\right) = \exp\left(\boldsymbol{t}^\top \boldsymbol{\mu} + \frac{1}{2}\boldsymbol{t}^\top \Sigma \boldsymbol{t}\right).$$

1. For any $\boldsymbol{s} \in \mathbb{R}^m$,

$$\begin{aligned} M_{\boldsymbol{Y}}(\boldsymbol{s}) &= \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top \boldsymbol{Y}\right)\right] = \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top (A\boldsymbol{X}+\boldsymbol{b})\right)\right] \\ &= \exp\left(\boldsymbol{s}^\top \boldsymbol{b}\right) \mathbb{E}\left[\exp\left(\left(A^\top \boldsymbol{s}\right)^\top \boldsymbol{X}\right)\right] = \exp\left(\boldsymbol{s}^\top \boldsymbol{b}\right) M_{\boldsymbol{X}}\left(A^\top \boldsymbol{s}\right) \\ &= \exp\left(\boldsymbol{s}^\top \boldsymbol{b} + \left(A^\top \boldsymbol{s}\right)^\top \boldsymbol{\mu} + \frac{1}{2}\left(A^\top \boldsymbol{s}\right)^\top \Sigma \left(A^\top \boldsymbol{s}\right)\right) \\ &= \exp\left(\boldsymbol{s}^\top (A\boldsymbol{\mu}+\boldsymbol{b}) + \frac{1}{2}\boldsymbol{s}^\top \left(A\Sigma A^\top\right)\boldsymbol{s}\right). \end{aligned}$$

   This is the mgf of a multivariate normal distribution with mean $A\boldsymbol{\mu}+\boldsymbol{b}$ and covariance $A\Sigma A^\top$. By uniqueness of the mgf, $\boldsymbol{Y}$ is Gaussian with

$$\mathbb{E}[\boldsymbol{Y}] = A\boldsymbol{\mu}+\boldsymbol{b}, \qquad \mathrm{Cov}(\boldsymbol{Y}) = A\Sigma A^\top.$$

   Equivalently, the cumulant generating function $K_{\boldsymbol{Y}}(\boldsymbol{s}) = \log M_{\boldsymbol{Y}}(\boldsymbol{s}) = \boldsymbol{s}^\top (A\boldsymbol{\mu}+\boldsymbol{b}) + \frac{1}{2}\boldsymbol{s}^\top \left(A\Sigma A^\top\right)\boldsymbol{s}$ satisfies

$$\nabla_s K_{\boldsymbol{Y}}(\boldsymbol{0}) = A\boldsymbol{\mu}+\boldsymbol{b}, \qquad \nabla_s^2 K_{\boldsymbol{Y}}(\boldsymbol{0}) = A\Sigma A^\top.$$

2. For any $\boldsymbol{s} \in \mathbb{R}^d$,

$$\begin{aligned} M_{\boldsymbol{Z}}(\boldsymbol{s}) &= \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top \boldsymbol{Z}\right)\right] = \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top (a\boldsymbol{X}+b\boldsymbol{Y})\right)\right] \\ &= \mathbb{E}\left[\exp\left(a\boldsymbol{s}^\top \boldsymbol{X}\right)\exp\left(b\boldsymbol{s}^\top \boldsymbol{Y}\right)\right] = \mathbb{E}\left[\exp\left(a\boldsymbol{s}^\top \boldsymbol{X}\right)\right]\mathbb{E}\left[\exp\left(b\boldsymbol{s}^\top \boldsymbol{Y}\right)\right] \\ &= M_{\boldsymbol{X}}(a\boldsymbol{s})\, M_{\boldsymbol{Y}}(b\boldsymbol{s}) \\ &= \exp\left((a\boldsymbol{s})^\top \boldsymbol{\mu}_X + \frac{1}{2}(a\boldsymbol{s})^\top \Sigma_X (a\boldsymbol{s})\right)\exp\left((b\boldsymbol{s})^\top \boldsymbol{\mu}_Y + \frac{1}{2}(b\boldsymbol{s})^\top \Sigma_Y (b\boldsymbol{s})\right) \\ &= \exp\left(\boldsymbol{s}^\top (a\boldsymbol{\mu}_X + b\boldsymbol{\mu}_Y) + \frac{1}{2}\boldsymbol{s}^\top \left(a^2\Sigma_X + b^2\Sigma_Y\right)\boldsymbol{s}\right). \end{aligned}$$

   This is the mgf of a multivariate normal distribution with mean $a\boldsymbol{\mu}_X + b\boldsymbol{\mu}_Y$ and covariance $a^2\Sigma_X + b^2\Sigma_Y$. Again by uniqueness of the mgf, $\boldsymbol{Z}$ is Gaussian with

$$\boldsymbol{\mu}_Z = a\boldsymbol{\mu}_X + b\boldsymbol{\mu}_Y, \qquad \Sigma_Z = a^2\Sigma_X + b^2\Sigma_Y.$$

   Equivalently, the cumulant generating function $K_{\boldsymbol{Z}}(\boldsymbol{s}) = \log M_{\boldsymbol{Z}}(\boldsymbol{s}) = \boldsymbol{s}^\top (a\boldsymbol{\mu}_X + b\boldsymbol{\mu}_Y) + \frac{1}{2}\boldsymbol{s}^\top \left(a^2\Sigma_X + b^2\Sigma_Y\right)\boldsymbol{s}$ satisfies

$$\nabla_s K_{\boldsymbol{Z}}(\boldsymbol{0}) = a\boldsymbol{\mu}_X + b\boldsymbol{\mu}_Y, \qquad \nabla_s^2 K_{\boldsymbol{Z}}(\boldsymbol{0}) = a^2\Sigma_X + b^2\Sigma_Y.$$

3. For any $\boldsymbol{s} \in \mathbb{R}^{d_y}$,

$$M_{\boldsymbol{Y}}(\boldsymbol{s}) = \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top \boldsymbol{Y}\right)\right] = \mathbb{E}\left[\ \mathbb{E}\left[\exp\left(\boldsymbol{s}^\top \boldsymbol{Y}\right) \mid \boldsymbol{X}\right]\ \right].$$

   Given $\boldsymbol{X}$, the vector $\boldsymbol{Y}$ is Gaussian with mean $A\boldsymbol{X}+\boldsymbol{b}$ and covariance $\Sigma_{Y|X}$, hence its conditional mgf is

$$\mathbb{E}\left[\exp\left(\boldsymbol{s}^\top \boldsymbol{Y}\right) \mid \boldsymbol{X}\right] = \exp\left(\boldsymbol{s}^\top (A\boldsymbol{X}+\boldsymbol{b}) + \frac{1}{2}\boldsymbol{s}^\top \Sigma_{Y|X}\boldsymbol{s}\right).$$

Taking expectation over $\boldsymbol{X}$ and using the Gaussian mgf for $\boldsymbol{X}$ at argument $A^\top \boldsymbol{s}$ gives

$$\begin{aligned} M_{\boldsymbol{Y}}(\boldsymbol{s}) &= \exp\left(\boldsymbol{s}^\top \boldsymbol{b} + \frac{1}{2}\boldsymbol{s}^\top \Sigma_{Y|X} \boldsymbol{s}\right) \mathbb{E}\left[\exp\left(\left(A^\top \boldsymbol{s}\right)^\top \boldsymbol{X}\right)\right] \\ &= \exp\left(\boldsymbol{s}^\top \boldsymbol{b} + \frac{1}{2}\boldsymbol{s}^\top \Sigma_{Y|X} \boldsymbol{s}\right) \exp\left(\left(A^\top \boldsymbol{s}\right)^\top \boldsymbol{\mu}_X + \frac{1}{2}\left(A^\top \boldsymbol{s}\right)^\top \Sigma_{XX}\left(A^\top \boldsymbol{s}\right)\right) \\ &= \exp\left(\boldsymbol{s}^\top (A\boldsymbol{\mu}_X + \boldsymbol{b}) + \frac{1}{2}\boldsymbol{s}^\top \left(\Sigma_{Y|X} + A\Sigma_{XX}A^\top\right)\boldsymbol{s}\right). \end{aligned}$$

This is the mgf of a multivariate normal distribution with mean $\boldsymbol{\mu}_Y = A\boldsymbol{\mu}_X + \boldsymbol{b}$ and covariance $\Sigma_Y = \Sigma_{Y|X} + A\Sigma_{XX}A^\top$. By uniqueness of the mgf, the marginal $\boldsymbol{Y}$ is Gaussian with

$$\boldsymbol{\mu}_Y = A\boldsymbol{\mu}_X + \boldsymbol{b}, \qquad \Sigma_Y = \Sigma_{Y|X} + A\Sigma_{XX}A^\top.$$

As a consistency check, the same formulas follow from the laws of total expectation and total covariance:

$$\mathbb{E}[\boldsymbol{Y}] = \mathbb{E}[\,\mathbb{E}[\boldsymbol{Y} \mid \boldsymbol{X}]\,] = \mathbb{E}[A\boldsymbol{X} + \boldsymbol{b}] = A\boldsymbol{\mu}_X + \boldsymbol{b},$$

$$\begin{aligned} \mathrm{Cov}(\boldsymbol{Y}) &= \mathbb{E}[\,\mathrm{Cov}(\boldsymbol{Y} \mid \boldsymbol{X})\,] + \mathrm{Cov}(\,\mathbb{E}[\boldsymbol{Y} \mid \boldsymbol{X}]\,) \\ &= \Sigma_{Y|X} + \mathrm{Cov}(A\boldsymbol{X} + \boldsymbol{b}) = \Sigma_{Y|X} + A\Sigma_{XX}A^\top. \end{aligned}$$

4. Write $\delta\boldsymbol{x} = \boldsymbol{x} - \boldsymbol{\mu}_X$ and $\delta\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{\mu}_Y$. The joint density is

$$f_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{x},\boldsymbol{y}) = (2\pi)^{-\frac{d_x+d_y}{2}} (\det\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\begin{pmatrix}\delta\boldsymbol{x}\\ \delta\boldsymbol{y}\end{pmatrix}^\top \Sigma^{-1}\begin{pmatrix}\delta\boldsymbol{x}\\ \delta\boldsymbol{y}\end{pmatrix}\right).$$

The marginal density of $\boldsymbol{X}$ is

$$f_{\boldsymbol{X}}(\boldsymbol{x}) = (2\pi)^{-\frac{d_x}{2}} (\det\Sigma_{XX})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\delta\boldsymbol{x}^\top \Sigma_{XX}^{-1}\delta\boldsymbol{x}\right).$$

Hence

$$f_{\boldsymbol{Y}|\boldsymbol{X}=\boldsymbol{x}}(\boldsymbol{y}) = \frac{f_{\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{x},\boldsymbol{y})}{f_{\boldsymbol{X}}(\boldsymbol{x})} = (2\pi)^{-\frac{d_y}{2}} \left(\frac{\det\Sigma_{XX}}{\det\Sigma}\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}Q(\boldsymbol{x},\boldsymbol{y})\right),$$

where

$$Q(\boldsymbol{x},\boldsymbol{y}) = \begin{pmatrix}\delta\boldsymbol{x}\\ \delta\boldsymbol{y}\end{pmatrix}^\top \Sigma^{-1}\begin{pmatrix}\delta\boldsymbol{x}\\ \delta\boldsymbol{y}\end{pmatrix} - \delta\boldsymbol{x}^\top \Sigma_{XX}^{-1}\delta\boldsymbol{x}.$$

Introduce the Schur complement

$$\Sigma_{Y|X} = \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}.$$

By the block matrix inversion identity proved below,

$$\Sigma^{-1} = \begin{pmatrix} \Sigma_{XX}^{-1} + \Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{Y|X}^{-1}\Sigma_{YX}\Sigma_{XX}^{-1} & -\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{Y|X}^{-1} \\ -\Sigma_{Y|X}^{-1}\Sigma_{YX}\Sigma_{XX}^{-1} & \Sigma_{Y|X}^{-1} \end{pmatrix}.$$

Substituting this into $Q(\boldsymbol{x}, \boldsymbol{y})$ and expanding gives

$$\begin{aligned} Q(\boldsymbol{x}, \boldsymbol{y}) &= \delta\boldsymbol{y}^\top \Sigma_{Y|X}^{-1} \delta\boldsymbol{y} - 2\delta\boldsymbol{y}^\top \Sigma_{Y|X}^{-1} \Sigma_{YX} \Sigma_{XX}^{-1} \delta\boldsymbol{x} + \delta\boldsymbol{x}^\top \Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{Y|X}^{-1} \Sigma_{YX} \Sigma_{XX}^{-1} \delta\boldsymbol{x} \\ &= \left(\delta\boldsymbol{y} - \Sigma_{YX}\Sigma_{XX}^{-1}\delta\boldsymbol{x}\right)^\top \Sigma_{Y|X}^{-1} \left(\delta\boldsymbol{y} - \Sigma_{YX}\Sigma_{XX}^{-1}\delta\boldsymbol{x}\right). \end{aligned}$$

Therefore with $\boldsymbol{u} = \boldsymbol{y} - \boldsymbol{\mu}_Y - \Sigma_{YX}\Sigma_{XX}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_X)$,

$$f_{\boldsymbol{Y}|\boldsymbol{X}=\boldsymbol{x}}(\boldsymbol{y}) = (2\pi)^{-\frac{d_y}{2}} \left(\frac{\det \Sigma_{XX}}{\det \Sigma}\right)^{\frac{1}{2}} \exp\left(-\frac{1}{2}\boldsymbol{u}^\top \Sigma_{Y|X}^{-1} \boldsymbol{u}\right).$$

Using the block determinant identity $\det\Sigma = \det\Sigma_{XX} \det\left(\Sigma_{Y|X}\right)$ proved below, the leading constant simplifies to $(2\pi)^{-\frac{d_y}{2}} \left(\det\Sigma_{Y|X}\right)^{-\frac{1}{2}}$. Hence the conditional distribution is Gaussian with

$$\boldsymbol{Y} \mid \boldsymbol{X} = \boldsymbol{x} \sim \mathcal{N}\left(\ \boldsymbol{\mu}_Y + \Sigma_{YX}\Sigma_{XX}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_X)\ ,\ \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}\ \right).$$

1. Proof of the block matrix inversion identity:
Assume Σ_{XX} is invertible and set $\Sigma_{Y|X} = \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}$. Factor Σ via block Gaussian elimination

$$\Sigma = \begin{pmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{pmatrix} = \underbrace{\begin{pmatrix} I & 0 \\ \Sigma_{YX}\Sigma_{XX}^{-1} & I \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} \Sigma_{XX} & 0 \\ 0 & \Sigma_{Y|X} \end{pmatrix}}_{D} \underbrace{\begin{pmatrix} I & \Sigma_{XX}^{-1}\Sigma_{XY} \\ 0 & I \end{pmatrix}}_{U}.$$

Inverting the triangular factors,

$$L^{-1} = \begin{pmatrix} I & 0 \\ -\Sigma_{YX}\Sigma_{XX}^{-1} & I \end{pmatrix}, \quad D^{-1} = \begin{pmatrix} \Sigma_{XX}^{-1} & 0 \\ 0 & \Sigma_{Y|X}^{-1} \end{pmatrix}, \quad U^{-1} = \begin{pmatrix} I & -\Sigma_{XX}^{-1}\Sigma_{XY} \\ 0 & I \end{pmatrix}.$$

Thus $\Sigma^{-1} = U^{-1}D^{-1}L^{-1}$. A direct multiplication gives

$$\Sigma^{-1} = \begin{pmatrix} \Sigma_{XX}^{-1} + \Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{Y|X}^{-1}\Sigma_{YX}\Sigma_{XX}^{-1} & -\Sigma_{XX}^{-1}\Sigma_{XY}\Sigma_{Y|X}^{-1} \\ -\Sigma_{Y|X}^{-1}\Sigma_{YX}\Sigma_{XX}^{-1} & \Sigma_{Y|X}^{-1} \end{pmatrix}.$$

2. Proof of the block determinant identity:
With the same L, D, U factorization as above, both L and U are block unit triangular, hence $\det L = 1$ and $\det U = 1$. Therefore

$$\begin{aligned} \det\Sigma = \det L \det D \det U = \det D &= \det\Sigma_{XX} \det\left(\Sigma_{Y|X}\right) \\ &= \det\Sigma_{XX} \det\left(\Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}\right). \end{aligned}$$

5.

$$p_i(\boldsymbol{x}) = (2\pi)^{-d/2} (\det\Sigma_i)^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^\top \Sigma_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right),\ i \in \{0, 1\}.$$

By definition,

$$D_{KL}(\mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0) \| \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1)) = \mathbb{E}_{p_0}\left[\log p_0(\boldsymbol{X}) - \log p_1(\boldsymbol{X})\right].$$

Using the Gaussian log density and cancelling the $\left(-\frac{d}{2}\log(2\pi)\right)$ terms,

$$\log p_0(\boldsymbol{x}) - \log p_1(\boldsymbol{x}) = \frac{1}{2}\log\frac{\det\Sigma_1}{\det\Sigma_0} - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_0)^\top\Sigma_0^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_0) + \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_1)^\top\Sigma_1^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_1).$$

Taking expectation under $\boldsymbol{X}\sim\mathcal{N}(\boldsymbol{\mu}_0,\Sigma_0)$,

$$D_{KL} = \frac{1}{2}\log\frac{\det\Sigma_1}{\det\Sigma_0} - \frac{1}{2}\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_0)^\top\Sigma_0^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_0)\right] + \frac{1}{2}\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_1)^\top\Sigma_1^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_1)\right].$$

The first expectation equals d because

$$\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_0)(\boldsymbol{X}-\boldsymbol{\mu}_0)^\top\right] = \Sigma_0 \implies \mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_0)^\top\Sigma_0^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_0)\right] = \operatorname{tr}\left(\Sigma_0^{-1}\Sigma_0\right) = d.$$

For the second expectation, write

$$\begin{aligned}\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_1)(\boldsymbol{X}-\boldsymbol{\mu}_1)^\top\right] &= \mathbb{E}\left[((\boldsymbol{X}-\boldsymbol{\mu}_0)+(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1))((\boldsymbol{X}-\boldsymbol{\mu}_0)+(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1))^\top\right]\\ &= \mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_0)(\boldsymbol{X}-\boldsymbol{\mu}_0)^\top\right] + (\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^\top\\ &= \Sigma_0 + (\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^\top,\end{aligned}$$

since $\mathbb{E}[\boldsymbol{X}-\boldsymbol{\mu}_0]=\boldsymbol{0}$ kills the cross terms. Hence

$$\begin{aligned}\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_1)^\top\Sigma_1^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_1)\right] &= \operatorname{tr}\left(\Sigma_1^{-1}\mathbb{E}\left[(\boldsymbol{X}-\boldsymbol{\mu}_1)(\boldsymbol{X}-\boldsymbol{\mu}_1)^\top\right]\right)\\ &= \operatorname{tr}\left(\Sigma_1^{-1}\Sigma_0\right) + \operatorname{tr}\left(\Sigma_1^{-1}(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^\top\right)\\ &= \operatorname{tr}\left(\Sigma_1^{-1}\Sigma_0\right) + (\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)^\top\Sigma_1^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0),\end{aligned}$$

where the last step uses $\operatorname{tr}(AB)=\operatorname{tr}(BA)$ and $\operatorname{tr}\left(\boldsymbol{a}\boldsymbol{a}^\top B\right)=\boldsymbol{a}^\top B\boldsymbol{a}$. Substituting,

$$\begin{aligned}&D_{KL}(\mathcal{N}(\boldsymbol{\mu}_0,\Sigma_0)\|\mathcal{N}(\boldsymbol{\mu}_1,\Sigma_1))\\ &\quad= \frac{1}{2}\left(\operatorname{tr}\left(\Sigma_1^{-1}\Sigma_0\right) + (\boldsymbol{\mu}_1-\boldsymbol{\mu}_0)^\top\Sigma_1^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_0) - d + \log\frac{\det\Sigma_1}{\det\Sigma_0}\right),\end{aligned}$$

valid when Σ_0 and Σ_1 are symmetric positive definite so that all terms are finite.

For $d=1$, set $\Sigma_0=\sigma_0^2$ and $\Sigma_1=\sigma_1^2$. Then $\operatorname{tr}\left(\Sigma_1^{-1}\Sigma_0\right)=\sigma_0^2/\sigma_1^2$, $\log\det$ becomes $\log\left(\sigma_1^2/\sigma_0^2\right)$, and the quadratic form becomes $(\mu_1-\mu_0)^2/\sigma_1^2$. Therefore

$$D_{KL}\left(\mathcal{N}\left(\mu_0,\sigma_0^2\right)\|\mathcal{N}\left(\mu_1,\sigma_1^2\right)\right) = \frac{1}{2}\left(\frac{\sigma_0^2}{\sigma_1^2} + \frac{(\mu_1-\mu_0)^2}{\sigma_1^2} - 1 + \log\frac{\sigma_1^2}{\sigma_0^2}\right).$$

Problem 2: Autoregressive (AR) Models of Order p

1,2. Let $\{y_t\}_{t=1}^n$ follow the AR(p) model

$$y_t = \sum_{j=1}^{p}\phi_j y_{t-j} + \varepsilon_t, \qquad \varepsilon_t \overset{i.i.d.}{\sim} \mathcal{N}\left(0,\sigma^2\right).$$

Define one–step–ahead innovations

$$e_t(\boldsymbol{\phi}) = y_t - \sum_{j=1}^{p} \phi_j y_{t-j}, \qquad t = p+1, \ldots, n,$$

where $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_p)^\top$. Conditional on the initial lag vector $(y_1, \ldots, y_p)$, the variables $e_t(\boldsymbol{\phi})$ are independent $\mathcal{N}(0, \sigma^2)$. Hence the conditional likelihood is

$$L_c(\boldsymbol{\phi}, \sigma^2 \mid y_{1:n}) = \prod_{t=p+1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{e_t(\boldsymbol{\phi})^2}{2\sigma^2}\right) = \left(2\pi\sigma^2\right)^{-\frac{n-p}{2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{t=p+1}^{n} e_t(\boldsymbol{\phi})^2\right).$$

Therefore the conditional log-likelihood is

$$\ell_c(\boldsymbol{\phi}, \sigma^2) = -\frac{n-p}{2} \log(2\pi) - \frac{n-p}{2} \log\left(\sigma^2\right) - \frac{1}{2\sigma^2} S(\boldsymbol{\phi}),$$
$$S(\boldsymbol{\phi}) = \sum_{t=p+1}^{n} \left(y_t - \sum_{j=1}^{p} \phi_j y_{t-j}\right)^2.$$

It is convenient to write this in matrix form. Let

$$\boldsymbol{y} = \begin{pmatrix} y_{p+1} \\ y_{p+2} \\ \vdots \\ y_n \end{pmatrix}, \qquad X = \begin{pmatrix} y_p & y_{p-1} & \cdots & y_1 \\ y_{p+1} & y_p & \cdots & y_2 \\ \vdots & \vdots & \ddots & \vdots \\ y_{n-1} & y_{n-2} & \cdots & y_{n-p} \end{pmatrix}, \qquad \boldsymbol{\phi} = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{pmatrix}.$$

Then $\boldsymbol{e}(\boldsymbol{\phi}) = \boldsymbol{y} - X\boldsymbol{\phi}$ and

$$S(\boldsymbol{\phi}) = (\boldsymbol{y} - X\boldsymbol{\phi})^\top (\boldsymbol{y} - X\boldsymbol{\phi}).$$

The log-likelihood becomes

$$\ell_c(\boldsymbol{\phi}, \sigma^2) = -\frac{n-p}{2} \log(2\pi) - \frac{n-p}{2} \log\left(\sigma^2\right) - \frac{1}{2\sigma^2} (\boldsymbol{y} - X\boldsymbol{\phi})^\top (\boldsymbol{y} - X\boldsymbol{\phi}).$$

Maximization with respect to $\boldsymbol{\phi}$. For fixed σ^2, maximizing ℓ_c in $\boldsymbol{\phi}$ is equivalent to minimizing $S(\boldsymbol{\phi})$. Expand

$$S(\boldsymbol{\phi}) = \boldsymbol{y}^\top \boldsymbol{y} - 2\boldsymbol{\phi}^\top X^\top \boldsymbol{y} + \boldsymbol{\phi}^\top X^\top X \boldsymbol{\phi}.$$

Differentiate and set the gradient to zero:

$$\nabla_{\boldsymbol{\phi}} S(\boldsymbol{\phi}) = -2X^\top \boldsymbol{y} + 2X^\top X\boldsymbol{\phi} = \boldsymbol{0} \iff X^\top X\boldsymbol{\phi} = X^\top \boldsymbol{y}.$$

When $X^\top X$ is invertible, the unique solution of the normal equations is

$$\widehat{\boldsymbol{\phi}} = \left(X^\top X\right)^{-1} X^\top \boldsymbol{y}.$$

Equivalently, compute the log-likelihood gradient directly:

$$\nabla_{\boldsymbol{\phi}} \ell_c\left(\boldsymbol{\phi}, \sigma^2\right) = -\frac{1}{2\sigma^2} \nabla_{\boldsymbol{\phi}} S\left(\boldsymbol{\phi}\right) = -\frac{1}{\sigma^2}\left(X^\top X \boldsymbol{\phi} - X^\top \boldsymbol{y}\right),$$

so the stationary condition is the same. The Hessian with respect to $\boldsymbol{\phi}$ is

$$\nabla^2_{\boldsymbol{\phi}} \ell_c\left(\boldsymbol{\phi}, \sigma^2\right) = -\frac{1}{\sigma^2} X^\top X,$$

which is negative definite when $X^\top X$ is positive definite. Hence $\widehat{\boldsymbol{\phi}}$ is the unique maximizer in $\boldsymbol{\phi}$ for fixed σ^2.

Maximization with respect to σ^2. For fixed $\boldsymbol{\phi}$, differentiate ℓ_c with respect to σ^2:

$$\frac{\partial}{\partial \sigma^2} \ell_c\left(\boldsymbol{\phi}, \sigma^2\right) = -\frac{n-p}{2}\frac{1}{\sigma^2} + \frac{1}{2}\frac{S\left(\boldsymbol{\phi}\right)}{\left(\sigma^2\right)^2}.$$

Set to zero and solve for σ^2:

$$-\left(n-p\right)\sigma^2 + S\left(\boldsymbol{\phi}\right) = 0 \implies \widehat{\sigma}^2\left(\boldsymbol{\phi}\right) = \frac{1}{n-p} S\left(\boldsymbol{\phi}\right).$$

To verify a maximum, compute the second derivative

$$\frac{\partial^2}{\partial\left(\sigma^2\right)^2} \ell_c\left(\boldsymbol{\phi}, \sigma^2\right) = \frac{n-p}{2}\frac{1}{\left(\sigma^2\right)^2} - \frac{S\left(\boldsymbol{\phi}\right)}{\left(\sigma^2\right)^3},$$

which, evaluated at $\sigma^2 = \widehat{\sigma}^2\left(\boldsymbol{\phi}\right) = S\left(\boldsymbol{\phi}\right)/(n-p)$, equals

$$-\frac{n-p}{2}\frac{1}{\left(\sigma^2\right)^2} < 0.$$

Thus $\widehat{\sigma}^2\left(\boldsymbol{\phi}\right)$ maximizes ℓ_c in σ^2. Combining with $\widehat{\boldsymbol{\phi}}$ above and substituting $\boldsymbol{\phi} = \widehat{\boldsymbol{\phi}}$ yields the joint maximizers

$$\widehat{\boldsymbol{\phi}} = \left(X^\top X\right)^{-1} X^\top \boldsymbol{y}, \qquad \widehat{\sigma}^2 = \frac{1}{n-p}\left(\boldsymbol{y} - X\widehat{\boldsymbol{\phi}}\right)^\top \left(\boldsymbol{y} - X\widehat{\boldsymbol{\phi}}\right).$$

3. Let $\mathcal{F}_{t-1} = \sigma\left(y_{t-1}, y_{t-2}, \ldots\right)$. The model is

$$y_t = \phi\, y_{t-1} + \varepsilon_t, \qquad \varepsilon_t \text{ independent of } \mathcal{F}_{t-1}, \quad \varepsilon_t \sim \mathcal{N}\left(0, \sigma^2\right).$$

For any Borel set $B \subset \mathbb{R}$,

$$\mathbb{P}\left(y_t \in B \mid \mathcal{F}_{t-1}\right) = \mathbb{P}\left(\phi\, y_{t-1} + \varepsilon_t \in B \mid \mathcal{F}_{t-1}\right) = \mathbb{P}\left(\varepsilon_t \in B - \phi\, y_{t-1}\right) = \mathbb{P}\left(y_t \in B \mid y_{t-1}\right),$$

since conditioning on $\mathcal{F}_{t-1}$ fixes y_{t-1} and ε_t is independent of $\mathcal{F}_{t-1}$. Thus

$$\mathbb{P}\left(y_t \in B \mid y_{t-1}, y_{t-2}, \ldots\right) = \mathbb{P}\left(y_t \in B \mid y_{t-1}\right),$$

so $\{y_t\}$ is a time-homogeneous Markov process of order 1.

(a) Conditional on $y_{t-1} = x$,

$$(y_t \mid y_{t-1} = x) = \phi x + \varepsilon_t \sim \mathcal{N}\left(\phi x, \sigma^2\right).$$

Hence, for $y \in \mathbb{R}$,

$$p\left(y_t = y \mid y_{t-1} = x\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - \phi x)^2}{2\sigma^2}\right),$$

which depends on t only through the constant parameters ϕ, σ^2, so the chain is time-homogeneous.

(b) Iterate the recursion:

$$y_t = \phi^t y_0 + \sum_{k=0}^{t-1} \phi^k \varepsilon_{t-k}.$$

If $|\phi| < 1$, then $\phi^t y_0 \to 0$ almost surely. The sum is a linear combination of independent zero-mean Gaussians, hence Gaussian with mean 0 and variance

$$\mathrm{Var}\left(y_t\right) = \sum_{k=0}^{t-1} \phi^{2k}\, \sigma^2 = \sigma^2 \sum_{k=0}^{t-1} \phi^{2k} = \sigma^2 \frac{1 - \phi^{2t}}{1 - \phi^2} \xrightarrow[t\to\infty]{} \frac{\sigma^2}{1 - \phi^2}.$$

Therefore the unique stationary distribution is

$$\pi\left(y\right) = \mathcal{N}\left(0,\ \frac{\sigma^2}{1 - \phi^2}\right),$$

with mean 0 and variance $\sigma^2 / \left(1 - \phi^2\right)$.

Problem 3: Nonlinear and Multi-Step Autoregressive Models

1. • Conditioning on the initial window $x_{1:p}$, the likelihood factors as

$$p_\theta\left(x_{p+1:T} \mid x_{1:p}\right) = \prod_{t=p+1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(x_t - f_\theta\left(x_{t-1:t-p}\right)\right)^2}{2\sigma^2}\right).$$

Hence the negative log likelihood is

$$\mathcal{L}\left(\theta\right) = -\log p_\theta\left(x_{p+1:T} \mid x_{1:p}\right) = \frac{T - p}{2} \log\left(2\pi\sigma^2\right) + \frac{1}{2\sigma^2} \sum_{t=p+1}^{T} \left(x_t - f_\theta\left(x_{t-1:t-p}\right)\right)^2.$$

Define the per step residual $r_t = f_\theta\left(x_{t-1:t-p}\right) - x_t$. Then

$$\mathcal{L}\left(\theta\right) = \frac{T - p}{2} \log\left(2\pi\sigma^2\right) + \frac{1}{2\sigma^2} \sum_{t=p+1}^{T} r_t^2.$$

- For a windowed feedforward f_θ that takes $(x_{t-1}, \ldots, x_{t-p})$ as inputs, the gradient is a sum of per step contributions with shared weights

$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{2\sigma^2} \sum_{t=p+1}^{T} \frac{\partial r_t^2}{\partial r_t} \cdot \frac{\partial r_t}{\partial \theta} = \frac{1}{\sigma^2} \sum_{t=p+1}^{T} r_t \nabla_\theta f_\theta(x_{t-1:t-p}).$$

  Computationally, one runs a forward pass to compute all $f_\theta(x_{t-1:t-p})$, accumulates the residuals r_t, and backpropagates through the network of f_θ at each time step, summing the parameter gradients because the same θ is reused.

- When training with teacher forcing, each $f_\theta(x_{t-1:t-p})$ consumes ground truth inputs, so there is no dependency of x_t on earlier model outputs within the loss. Gradients do not have to traverse across time through the data sequence; they only traverse within the network that defines f_θ. The only coupling across time is weight sharing, which results in summation of per step gradients.

  When f_θ is recurrent or when training objectives unroll the model over its own predictions for multi step horizons, gradients propagate through time via products of Jacobians such as $J_{g,h}$. For long sequences or large effective horizons this can cause vanishing or exploding gradients depending on the spectral norms of these Jacobians. If the norms are mostly below one, products tend to zero and gradients vanish, hampering learning of long range dependencies. If they are above one, products can blow up and gradients explode. Common mitigations include residual or gating architectures, gradient clipping, careful initialization, normalization, and contracting regularization that keeps the state update near a contraction. For large p with a purely feedforward f_θ, the input dimension grows but the gradient path length in time does not, so the main concern shifts to optimization conditioning and input scaling rather than temporal vanishing or exploding.

2. - When the order p is very large, the nonlinear function f_θ receives a long input vector $\boldsymbol{x}_{t-1:t-p} = (x_{t-1}, \ldots, x_{t-p})$. A fully connected multilayer perceptron (MLP) whose first layer directly connects to all p lags has $O(p)$ inputs and quickly becomes expensive and prone to overfitting as p grows. In this regime it is natural to exploit structure in time.

     One important design is a one dimensional convolutional network over the lag axis. A 1D CNN with kernel size k and shared weights across time can model local temporal patterns with $O(k)$ parameters per filter instead of $O(p)$. Stacking layers increases the effective receptive field. With dilated convolutions the receptive field can grow exponentially in depth, so a deep dilated CNN (as in WaveNet type architectures) can cover a very large context with logarithmic depth. This yields

$$f_\theta(x_{t-1}, \ldots, x_{t-p}) = \mathrm{Conv1D}_\theta(x_{t-p:t-1}),$$

     where the convolution is causal and may include residual connections and normalization to improve training stability.

     Self attention based architectures can also be used when p is large. A transformer style block takes all lags $x_{t-1}, \ldots, x_{t-p}$ as a sequence and uses attention to adaptively focus on a subset of relevant time steps. This can capture long range dependencies and interactions that are hard for local convolutions. However, the naive attention

cost is $O\left(p^2\right)$ and memory usage also scales quadratically in p, which becomes prohibitive when p is very large. Practical use requires sparse or low rank attention, truncation of context, or streaming approximations.

Recurrent neural networks (RNNs) such as LSTMs or GRUs can be used to summarize the long history into a hidden state $\boldsymbol{h}_{t-1}$, and one can set

$$f_\theta\left(x_{t-1}, \ldots, x_{t-p}\right) = g_\theta\left(\boldsymbol{h}_{t-1}\right), \qquad \boldsymbol{h}_{t-1} = \mathrm{RNN}_\theta\left(\boldsymbol{h}_{t-p-1}, x_{t-p:t-1}\right).$$

This avoids a p dimensional input layer, but gradients through long sequences may vanish or explode, and training cost scales with sequence length. Gated units and techniques like truncated backpropagation, gradient clipping, and normalization help but there remain practical limits on the effective horizon.

In summary, for $p \gg 1$ it is usually preferable to choose architectures with

- parameter sharing across time (convolutions, RNNs, attention),
- controlled growth of receptive field (dilated convolutions or attention),
- good inductive bias for temporal structure (causal convolutions or causal attention).

Fully connected MLPs with dense connections to all lags become inefficient and unstable as p grows.

- When p is small, for example $p = 1, 2, 3$, the input dimension to f_θ is low and the model does not need to handle very long temporal dependencies. In this regime simple architectures are often preferable.

  A shallow MLP with a few hidden layers and nonlinearities is a natural choice. The mapping

  $$f_\theta\left(x_{t-1}, \ldots, x_{t-p}\right) = \mathrm{MLP}_\theta\left(x_{t-1}, \ldots, x_{t-p}\right)$$

  has

  - good approximation power for smooth nonlinear relationships between a small number of lags,
  - low computational cost per time step, since the first layer has only p inputs,
  - simple and stable training behavior, because gradients do not need to propagate through time and the network is relatively shallow.

  For $p = 1$, even a single hidden layer with a modest number of units is often sufficient, and adding recurrent or attention mechanisms usually brings little benefit while increasing complexity and risk of overfitting.

  Simple recurrent models like GRUs or LSTMs can still be used for small p, but they are typically unnecessary compared to MLPs unless one wants to process variable length contexts or share a single recurrent state across many different predictive tasks. Transformers and attention mechanisms are generally excessive in this setting: the sequence length seen by f_θ is tiny, so the advantages of flexible long range attention do not outweigh their computational and implementation overhead.

3. - Define deterministic point predictions as conditional means. First step:

$$\hat{x}_{t+1} = f_\theta\left(x_t, x_{t-1}, \ldots, x_{t-p+1}\right).$$

For later steps we feed back previous predictions. For $j \geq 2$,

$$\hat{x}_{t+j} = f_\theta\left(x^*_{t+j-1}, x^*_{t+j-2}, \ldots, x^*_{t+j-p}\right),$$

where

$$x^*_s = \begin{cases} x_s & s \leq t, \\ \hat{x}_s & s > t. \end{cases}$$

Then $\hat{x}_{t+1}, \hat{x}_{t+2}, \ldots, \hat{x}_{t+k}$ are defined recursively for any k.

- Assume the true process is

$$x_{t+1} = f_\star\left(x_t, \ldots, x_{t-p+1}\right) + \varepsilon_{t+1}, \quad \mathbb{E}\left[\varepsilon_{t+1}\right] = 0.$$

Define the true state vector

$$s_t = \begin{pmatrix} x_t \\ x_{t-1} \\ \vdots \\ x_{t-p+1} \end{pmatrix} \in \mathbb{R}^p,$$

and the model state update map

$$F_\theta\left(s_t\right) = \begin{pmatrix} f_\theta\left(x_t, \ldots, x_{t-p+1}\right) \\ x_t \\ \vdots \\ x_{t-p+2} \end{pmatrix}.$$

Similarly the true update is

$$F_\star\left(s_t\right) = \begin{pmatrix} f_\star\left(x_t, \ldots, x_{t-p+1}\right) \\ x_t \\ \vdots \\ x_{t-p+2} \end{pmatrix}, \quad \eta_{t+1} = \begin{pmatrix} \varepsilon_{t+1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Then $s_{t+1} = F_\star\left(s_t\right) + \eta_{t+1}$. The free running model prediction obeys

$$\hat{s}_{t+1} = F_\theta\left(\hat{s}_t\right), \quad \hat{s}_t = \begin{pmatrix} x_t \\ x_{t-1} \\ \vdots \\ x_{t-p+1} \end{pmatrix}.$$

Define the state error $e_t = \hat{s}_t - s_t$. Then

$$\begin{aligned} e_{t+1} &= \hat{s}_{t+1} - s_{t+1} = F_\theta\left(\hat{s}_t\right) - F_\star\left(s_t\right) - \eta_{t+1} \\ &= F_\theta\left(\hat{s}_t\right) - F_\theta\left(s_t\right) + \underbrace{F_\theta\left(s_t\right) - F_\star\left(s_t\right)}_{\text{model bias}} - \eta_{t+1}. \end{aligned}$$

Assume F_θ is Lipschitz with constant L in some norm $\|\cdot\|$, and that the bias is bounded by $\delta \geq 0$,

$$\left\|F_\theta\left(u\right) - F_\theta\left(v\right)\right\| \leq L\left\|u - v\right\|, \quad \left\|F_\theta\left(s_t\right) - F_\star\left(s_t\right)\right\| \leq \delta.$$

Then

$$\|e_{t+1}\| \leq L \|e_t\| + \delta + \|\eta_{t+1}\| .$$

Iterating this recursion gives, for any $k \geq 1$,

$$\|e_{t+k}\| \leq L^k \|e_t\| + \sum_{j=0}^{k-1} L^j \left(\delta + \|\eta_{t+k-j}\|\right) .$$

Even if $\|e_t\| = 0$ initially, the expectation of $\|e_{t+k}\|$ grows with k as a geometric sum of model bias and noise terms. Thus multi step prediction errors are generally much larger than one step errors.

One way to reduce error accumulation is to train for multi step prediction. Fix a horizon K. For each starting time t, define free running predictions $\hat{x}_{t+1}, \ldots, \hat{x}_{t+K}$ by the recursion above and use the loss

$$\mathcal{L}_K(\theta) = \sum_t \sum_{j=1}^{K} \left(x_{t+j} - \hat{x}_{t+j}\right)^2 .$$

Backpropagation through time on $\mathcal{L}_K(\theta)$ forces the model to be robust to its own prediction errors and better aligned with the distribution of inputs it will see during multi step forecasting.

Another way is scheduled sampling. At training time, instead of always feeding the true past x_s into the model, randomly mix true and predicted values. For each step s define

$$z_s = m_s x_s + (1 - m_s)\, \hat{x}_s,$$

where $m_s \sim \text{Bernoulli}(q)$ is an independent mask with probability $q \in [0, 1]$ of using the true value. The model then uses windows built from z_s rather than x_s. Early in training one chooses $q \approx 1$ and gradually decreases q so that the model increasingly trains under its own predictions. This reduces the train test mismatch and mitigates error accumulation in long horizon forecasts.

4. • Conditioning on the first p values $x_{1:p}$, the conditional likelihood of $x_{p+1:T}$ given $c_{1:T}$ factorizes as

$$\begin{aligned} p_\theta\left(x_{p+1:T} \mid x_{1:p}, c_{1:T}\right) &= \prod_{t=p+1}^{T} p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right) \\ &= \prod_{t=p+1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(x_t - f_\theta\left(x_{t-1:t-p}; c_{1:t}\right)\right)^2}{2\sigma^2}\right) . \end{aligned}$$

The conditional log likelihood is

$$\begin{aligned} \log p_\theta\left(x_{p+1:T} \mid x_{1:p}, c_{1:T}\right) &= \sum_{t=p+1}^{T} \left[-\frac{1}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\left(x_t - f_\theta\left(x_{t-1:t-p}; c_{1:t}\right)\right)^2\right] \\ &= -\frac{T-p}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{t=p+1}^{T}\left(x_t - f_\theta\left(x_{t-1:t-p}; c_{1:t}\right)\right)^2 . \end{aligned}$$

For fixed σ^2, maximizing this with respect to θ is equivalent to minimizing

$$\mathcal{L}_{\text{cond}}(\theta) = \frac{1}{2\sigma^2} \sum_{t=p+1}^{T} \left(x_t - f_\theta\left(x_{t-1:t-p}; c_{1:t}\right)\right)^2,$$

or any positive constant multiple of this mean squared error. With a dataset of sequences indexed by n, the full training objective is the sum over sequences

$$\mathcal{L}_{\text{train}}(\theta) = \sum_{n} \sum_{t=p+1}^{T_n} \left(x_t^{(n)} - f_\theta\left(x_{t-1:t-p}^{(n)}; c_{1:t}^{(n)}\right)\right)^2.$$

- Given a trained model f_θ, a conditioning sequence $c_{1:T}$, and some initial values $x_{1:p}$ (either observed or sampled from an initial distribution), we can generate a conditional sequence by forward simulation.

  For $t = p+1, \ldots, T$ do:

  $$x_t \sim \mathcal{N}\left(f_\theta\left(x_{t-1}, \ldots, x_{t-p}; c_{1:t}\right), \sigma^2\right),$$

  feeding the newly generated x_t back into the inputs for future steps. Using the mean instead of sampling (that is, setting $x_t = f_\theta(\cdot)$) yields a deterministic conditional forecast.

  For example, in conditional speech generation, $c_{1:T}$ might be a sequence of text or linguistic features and $x_{1:T}$ an audio waveform. The model learns how the waveform evolves over time given the text. At generation time, for a new text sequence $c_{1:T}$, we sample or predict $x_{1:T}$ from the conditional model. Similarly, for motion generation, $c_{1:T}$ can be control commands or trajectory waypoints, and $x_{1:T}$ can be joint angles or positions.

- An unconditional autoregressive model for $x_{1:T}$ factorizes

  $$p_\theta(x_{1:T}) = \prod_{t=1}^{T} p_\theta\left(x_t \mid x_{<t}\right),$$

  so all dependence is internal to the sequence x. In the conditional case, the conditional distribution factorizes as

  $$p_\theta\left(x_{1:T} \mid c_{1:T}\right) = \prod_{t=1}^{T} p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right).$$

  Here x_t depends both on its own past $x_{<t}$ and on the conditioning signal $c_{1:t}$. If we regard c as exogenous, the joint density factorizes as

  $$p\left(x_{1:T}, c_{1:T}\right) = p\left(c_{1:T}\right) p_\theta\left(x_{1:T} \mid c_{1:T}\right),$$

  while in the unconditional case there is no explicit conditioning sequence.

  In terms of dependence structure, the unconditional AR model encodes

  $$x_t \perp\!\!\!\perp x_{>t} \mid x_{<t},$$

while the conditional AR additionally yields

$$x_t \perp\!\!\!\perp x_{>t} \mid x_{<t}, c_{1:T},$$

with direct functional dependence on $c_{1:t}$ at each step. This allows the model to explain variation in x_t caused by the external sequence c, rather than forcing all structure to be captured solely by past x.

Consequently, unconditional AR models are suitable for modeling a single sequence in isolation, such as raw audio or financial returns with no external covariates. Conditional AR models are designed for tasks where the sequence x is driven or modulated by another sequence c, such as speech given text, motion given control signals, or sensor readings given planned actions. The conditional formulation can capture richer input–output relationships and enables controllable generation by varying the conditioning sequence.

- Let $x_{1:T} = (x_1, \ldots, x_T)$ and $c_{1:T} = (c_1, \ldots, c_T)$. Consider two conditional autoregressive models

$$p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right), \qquad q_\phi\left(x_t \mid x_{<t}, c_{1:t}\right),$$

with corresponding sequence distributions

$$p_\theta\left(x_{1:T} \mid c_{1:T}\right) = \prod_{t=1}^{T} p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right), \qquad q_\phi\left(x_{1:T} \mid c_{1:T}\right) = \prod_{t=1}^{T} q_\phi\left(x_t \mid x_{<t}, c_{1:t}\right).$$

Fix a conditioning sequence $c_{1:T}$. The Kullback–Leibler divergence between the two conditional distributions is

$$\begin{aligned} D_{KL}\left(p_\theta\left(\cdot \mid c_{1:T}\right) \| q_\phi\left(\cdot \mid c_{1:T}\right)\right) &= \int p_\theta\left(x_{1:T} \mid c_{1:T}\right) \log \frac{p_\theta\left(x_{1:T} \mid c_{1:T}\right)}{q_\phi\left(x_{1:T} \mid c_{1:T}\right)} dx_{1:T} \\ &= \mathbb{E}_{p_\theta\left(x_{1:T} \mid c_{1:T}\right)}\left[\log p_\theta\left(x_{1:T} \mid c_{1:T}\right) - \log q_\phi\left(x_{1:T} \mid c_{1:T}\right)\right]. \end{aligned}$$

Using the autoregressive factorization,

$$\begin{aligned} \log p_\theta\left(x_{1:T} \mid c_{1:T}\right) &= \sum_{t=1}^{T} \log p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right), \\ \log q_\phi\left(x_{1:T} \mid c_{1:T}\right) &= \sum_{t=1}^{T} \log q_\phi\left(x_t \mid x_{<t}, c_{1:t}\right), \end{aligned}$$

so

$$D_{KL}\left(p_\theta \| q_\phi\right) = \sum_{t=1}^{T} \mathbb{E}_{p_\theta\left(x_{1:T} \mid c_{1:T}\right)}\left[\log p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right) - \log q_\phi\left(x_t \mid x_{<t}, c_{1:t}\right)\right].$$

In the Gaussian NAR case,

$$\begin{aligned} p_\theta\left(x_t \mid x_{<t}, c_{1:t}\right) &= \mathcal{N}\left(x_t \mid f_\theta\left(x_{t-1:t-p}; c_{1:t}\right), \sigma_p^2\right), \\ q_\phi\left(x_t \mid x_{<t}, c_{1:t}\right) &= \mathcal{N}\left(x_t \mid g_\phi\left(x_{t-1:t-p}; c_{1:t}\right), \sigma_q^2\right), \end{aligned}$$

the inner term is the KL between two univariate Gaussians for fixed history $(x_{<t}, c_{1:t})$,

$$D_{KL}\left(p_\theta\left(\cdot \mid x_{<t}, c_{1:t}\right) \| q_\phi\left(\cdot \mid x_{<t}, c_{1:t}\right)\right) = \frac{1}{2}\left(\log \frac{\sigma_q^2}{\sigma_p^2} + \frac{\sigma_p^2 + \left(f_\theta - g_\phi\right)^2}{\sigma_q^2} - 1\right),$$

where $f_\theta = f_\theta\left(x_{t-1:t-p}; c_{1:t}\right)$ and $g_\phi = g_\phi\left(x_{t-1:t-p}; c_{1:t}\right)$. The full sequence level KL is then

$$D_{KL}\left(p_\theta \| q_\phi\right) = \sum_{t=1}^{T} \mathbb{E}_{p_\theta\left(x_{1:T} \mid c_{1:T}\right)}\left[D_{KL}\left(p_\theta\left(\cdot \mid x_{<t}, c_{1:t}\right) \| q_\phi\left(\cdot \mid x_{<t}, c_{1:t}\right)\right)\right].$$

Computational challenges. Formally, the expectation is over the joint $p_\theta\left(x_{1:T} \mid c_{1:T}\right)$ on $\mathbb{R}^T$. Even though this distribution factorizes autoregressively, the histories $(x_{<t}, c_{1:t})$ enter nonlinearly through f_θ and g_ϕ, so there is no closed form for the expectation in general. Direct evaluation would require integrating over all $x_{1:T}$, which is intractable for high dimensional T. Dynamic programming does not help because the log ratio involves both models, and only one of them defines the sampling distribution p_θ. For neural network f_θ, g_ϕ, the integrand does not admit analytic marginalization.

Monte Carlo approximation. A practical approach is Monte Carlo estimation using ancestral sampling from p_θ. For a fixed conditioning sequence $c_{1:T}$:

1. Draw N independent trajectories $\boldsymbol{x}_{1:T}^{(n)} \sim p_\theta\left(x_{1:T} \mid c_{1:T}\right)$ by forward simulation

$$x_t^{(n)} \sim p_\theta\left(\cdot \mid x_{<t}^{(n)}, c_{1:t}\right), \qquad t = 1, \ldots, T.$$

2. For each sample compute

$$\log p_\theta\left(\boldsymbol{x}_{1:T}^{(n)} \mid c_{1:T}\right) = \sum_{t=1}^{T} \log p_\theta\left(x_t^{(n)} \mid x_{<t}^{(n)}, c_{1:t}\right),$$

$$\log q_\phi\left(\boldsymbol{x}_{1:T}^{(n)} \mid c_{1:T}\right) = \sum_{t=1}^{T} \log q_\phi\left(x_t^{(n)} \mid x_{<t}^{(n)}, c_{1:t}\right).$$

3. Approximate the KL divergence by the sample mean

$$D_{KL}\left(p_\theta \| q_\phi\right) \approx \frac{1}{N} \sum_{n=1}^{N} \left[\log p_\theta\left(\boldsymbol{x}_{1:T}^{(n)} \mid c_{1:T}\right) - \log q_\phi\left(\boldsymbol{x}_{1:T}^{(n)} \mid c_{1:T}\right)\right].$$

This estimator converges to the true KL as $N \to \infty$ by the law of large numbers. For random conditioning sequences, one can also average this estimate over a dataset of $c_{1:T}$ drawn from the empirical distribution, yielding an approximation of $\mathbb{E}_c\left[D_{KL}\left(p_\theta\left(\cdot \mid c\right) \| q_\phi\left(\cdot \mid c\right)\right)\right]$ in high dimensional autoregressive settings.

Problem 4: Real NADE Parameters

Let $\boldsymbol{h}_i \in \mathbb{R}^d$ be the hidden representation computed from $\boldsymbol{x}_{<i}$. We model the scalar conditional as a C component Gaussian mixture

$$p\left(x_i \mid \boldsymbol{x}_{<i}\right) = \sum_{c=1}^{C} \pi_i^c \mathcal{N}\left(x_i | \mu_i^c,\ \exp\left(s_i^c\right)\right),$$

where π_i^c are mixture weights, μ_i^c are means, and s_i^c are log variances so that the variance of component c is $\exp\left(s_i^c\right)$.

Matrix parameterization from $\boldsymbol{h}_i$. Introduce parameter matrices and biases

$$W_i^{(\pi)} \in \mathbb{R}^{C \times d}, \quad \boldsymbol{a}_i^{(\pi)} \in \mathbb{R}^C, \qquad V_i \in \mathbb{R}^{C \times d}, \quad \boldsymbol{b}_i \in \mathbb{R}^C, \qquad U_i \in \mathbb{R}^{C \times d}, \quad \boldsymbol{d}_i \in \mathbb{R}^C.$$

Compute componentwise logits, means, and log variances as

$$\boldsymbol{z}_i = W_i^{(\pi)} \boldsymbol{h}_i + \boldsymbol{a}_i^{(\pi)}, \qquad \boldsymbol{\mu}_i = V_i \boldsymbol{h}_i + \boldsymbol{b}_i, \qquad \boldsymbol{s}_i = U_i \boldsymbol{h}_i + \boldsymbol{d}_i.$$

Let z_i^c, μ_i^c, s_i^c denote the c th entries of $\boldsymbol{z}_i, \boldsymbol{\mu}_i, \boldsymbol{s}_i$. Map logits to mixture weights with a softmax so that weights are nonnegative and sum to one

$$\pi_i^c = \frac{\exp\left(z_i^c\right)}{\sum_{k=1}^{C} \exp\left(z_i^k\right)}, \qquad c = 1, \ldots, C.$$

This choice ensures $\sum_{c=1}^{C} \pi_i^c = 1$ automatically and permits unconstrained optimization of z_i^c.

Equivalent per component vector form. For each c introduce $\boldsymbol{w}_i^c, \boldsymbol{v}_i^c, \boldsymbol{u}_i^c \in \mathbb{R}^d$ and scalars α_i^c, b_i^c, d_i^c and set

$$z_i^c = \left(\boldsymbol{w}_i^c\right)^\top \boldsymbol{h}_i + \alpha_i^c, \qquad \mu_i^c = \left(\boldsymbol{v}_i^c\right)^\top \boldsymbol{h}_i + b_i^c, \qquad s_i^c = \left(\boldsymbol{u}_i^c\right)^\top \boldsymbol{h}_i + d_i^c, \qquad \pi_i^c = \frac{\exp\left(z_i^c\right)}{\sum_{k=1}^{C} \exp\left(z_i^k\right)}.$$

This is exactly the same as the matrix form with rows of the matrices collected into the per component vectors.

Parameter count for a single conditional $p\left(x_i \mid \boldsymbol{x}_{<i}\right)$. The mapping $\boldsymbol{h}_i \mapsto \{\pi_i^c, \mu_i^c, s_i^c\}_{c=1}^C$ uses

$$\underbrace{C\,(d+1)}_{\text{logits } \boldsymbol{z}_i} + \underbrace{C\,(d+1)}_{\text{means } \boldsymbol{\mu}_i} + \underbrace{C\,(d+1)}_{\text{log variances } \boldsymbol{s}_i} = 3C\,(d+1)$$

learned parameters for that single conditional. Although the simplex of $\boldsymbol{\pi}_i$ has $C-1$ degrees of freedom, the softmax parameterization with C logits is standard, smooth, and avoids constraints. The upstream parameters that produce $\boldsymbol{h}_i$ in Real NADE are shared across i and are not included in this per conditional count.

Notes. If desired, one can tie parameters across c or across i to reduce the count, or add temperature scaling to $\boldsymbol{z}_i$ for sharper or flatter mixtures. For multivariate x_i, the same scheme applies but each component would output vector means and either diagonal or full covariance parameters.

Problem 5: Autoregressive Graph Neural Networks for Sequential Data

1. Let $\mathcal{V}$ be the set of nodes and for each $v \in \mathcal{V}$ let x_v^t be the observation at time t. Define hidden states h_v^t updated by a GNN as

$$h_v^t = \mathrm{GNN}\left(h_v^{t-1}, \left\{h_u^{t-1} : u \in N(v)\right\}\right), \qquad t = 2, \ldots, T,$$

with some initialization for h_v^1. Assume an emission model $p\left(x_v^t \mid h_v^t\right)$, and that, conditional on $\left\{h_v^t\right\}_{v\in\mathcal{V}}$, the observations at time t are independent across nodes:

$$p\left(x_t \mid h_t\right) = \prod_{v\in\mathcal{V}} p\left(x_v^t \mid h_v^t\right), \qquad x_t = \left(x_v^t\right)_{v\in\mathcal{V}},\ h_t = \left(h_v^t\right)_{v\in\mathcal{V}}.$$

First factorize over time using the chain rule:

$$p\left(x_{1:T}\right) = p\left(x_1\right) \prod_{t=2}^{T} p\left(x_t \mid x_{<t}\right), \qquad x_{1:T} = \left(x_1, \ldots, x_T\right).$$

For a fixed $t \geq 2$, by conditional independence of nodes given h_t,

$$p\left(x_t \mid h_t\right) = \prod_{v\in\mathcal{V}} p\left(x_v^t \mid h_v^t\right).$$

The hidden state h_v^t is a deterministic function of the previous hidden states of v and its neighbors:

$$h_v^t = F_v\left(h_v^{t-1}, \left\{h_u^{t-1} : u \in N(v)\right\}\right).$$

Unrolling this recursion shows that h_v^t is in fact a deterministic function of the past observations of v and its neighbors. Thus there exists a function H_v such that

$$h_v^t = H_v\left(x_v^{<t}, x_{N(v)}^{<t}\right),$$

where $x_v^{<t} = \left(x_v^1, \ldots, x_v^{t-1}\right)$ and $x_{N(v)}^{<t} = \left\{x_u^{1:t-1} : u \in N(v)\right\}$. Therefore

$$p\left(x_v^t \mid x_{<t}\right) = p\left(x_v^t \mid h_v^t\right) = p\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right),$$

and, given the past, the nodewise observations at time t are conditionally independent:

$$p\left(x_t \mid x_{<t}\right) = \prod_{v\in\mathcal{V}} p\left(x_v^t \mid x_{<t}\right) = \prod_{v\in\mathcal{V}} p\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

Substitute this into the temporal factorization to obtain

$$p\left(x_{1:T}\right) = p\left(x_1\right) \prod_{t=2}^{T} \prod_{v\in\mathcal{V}} p\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

If we absorb the initial time $t = 1$ into the same notation by defining suitable initial factors, we can write the joint likelihood factorization as

$$p\left(x_{1:T}\right) = \prod_{t=1}^{T} \prod_{v\in\mathcal{V}} p\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right),$$

which is the desired result.

2. Let $\mathcal{V}$ be the node set. At time t, node v has scalar value x_v^t and binary mask $m_v^t \in \{0,1\}$. The observed and missing sets are

$$O_t = \left\{v : m_v^t = 1\right\}, \qquad M_t = \left\{v : m_v^t = 0\right\}.$$

Assume an autoregressive GNN model with factorization

$$p_\theta(\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} \prod_{v \in \mathcal{V}} p_\theta\Big(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\Big),$$

where $\boldsymbol{x}_t = \left(x_v^t\right)_{v \in \mathcal{V}}$.

- When only $\boldsymbol{x}_{O_t}^t$ is observed, the conditional log likelihood contribution at time t is

$$\log p_\theta\left(\boldsymbol{x}_{O_t}^t \mid \boldsymbol{x}_{<t}\right) = \sum_{v \in O_t} \log p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

  Equivalently, using masks,

$$\log p_\theta\left(\boldsymbol{x}_{\text{obs}}\right) = \sum_{t=1}^{T} \sum_{v \in \mathcal{V}} m_v^t \log p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

  The training objective is to maximize this log likelihood over θ, or minimize the masked negative log likelihood

$$\mathcal{L}(\theta) = -\sum_{t=1}^{T} \sum_{v \in \mathcal{V}} m_v^t \log p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

  To encourage robustness to missing inputs, we can:

  - Inject artificial masks during training: randomly set some $m_v^t = 0$ even when the value is available, and exclude those terms from the loss via the mask.
  - Feed the mask into the GNN as an additional feature, for example updating node states using inputs of the form $\left(m_v^{t-1} x_v^{t-1}, m_v^{t-1}\right)$ and $\left(m_u^{t-1} x_u^{t-1}, m_u^{t-1}\right)$ for neighbors $u \in N(v)$.

  This forces the model to learn to make good predictions even when some neighbors or self values are missing or unreliable.

- At a given time t, suppose only $\boldsymbol{x}_{O_t}^t$ are observed, but the past $\boldsymbol{x}_{<t}$ is (possibly partially) known or has already been imputed. With the autoregressive factorization

$$p_\theta\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{<t}\right) = \prod_{v \in \mathcal{V}} p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right),$$

  the missing values at time t are conditionally independent of each other and of the observed values at time t, given the past, so

$$p_\theta\left(\boldsymbol{x}_{M_t}^t \mid \boldsymbol{x}_{O_t}^t, \boldsymbol{x}_{<t}\right) = \prod_{v \in M_t} p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right).$$

Thus each missing node $v \in M_t$ can be imputed independently from its predictive distribution. For example, in a Gaussian output model

$$p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right) = \mathcal{N}\left(x_v^t \mid \mu_v^t, \left(\sigma_v^t\right)^2\right),$$

we can either:

$$\hat{x}_v^t = \mu_v^t \quad \text{(conditional mean imputation)}, \qquad \text{or} \quad x_v^t \sim \mathcal{N}\left(\mu_v^t, \left(\sigma_v^t\right)^2\right) \text{ (sampling)}.$$

For sequences with missing values over many times, we proceed sequentially in t:

- At each time t, use the current history $\boldsymbol{x}_{<t}$ (containing true or previously imputed values) to compute the predictive distributions $p_\theta\left(x_v^t \mid x_v^{<t}, x_{N(v)}^{<t}\right)$.
- For $v \in O_t$ keep the observed x_v^t; for $v \in M_t$ replace x_v^t by its mean or a sample from its predictive distribution.

In this way, the learned autoregressive GNN provides a principled mechanism to infer or sample missing node values using the conditional distributions it has learned from data.