DOA and Mutual Coupling Estimation using Score-Guided Diffusion

ZhaoJie Luo, *Student Member, IEEE*, and Carrson C. Fung*, *Member, IEEE*

Abstract

Index Terms

I. Introduction

II. Methodology

A. Fine Beam Alignment Via DoA/DoD and MCM Estimation

Suppose T time symbols with S frequency bins are used to estimate the parameters of the channel. Using the ray tracing model

$$\mathbf{H}=\sum_{p=1}^{P}\alpha_p\mathbf{C}_R\left(\theta_p,\phi_p\right)\mathbf{a}_R\left(\theta_p,\phi_p\right)\mathbf{a}_T\left(\theta_p,\phi_p\right)^H\mathbf{C}_T^H\left(\theta_p,\phi_p\right), \tag{1}$$

where $\mathbf{C}_R\left(\theta_p,\phi_p\right)\in\mathbb{C}^{N_R\times N_R}$ and $\mathbf{C}_T\left(\theta_p,\phi_p\right)\in\mathbb{C}^{N_T\times N_T}$ denote the mutual coupling matrices (MCMs) at the receiver and transmitter, respectively. Since the elements of the antenna array may not be uniformly distributed, hence, the MCMs need not be Hermitian symmetric. θ_p and ϕ_p denote the azimuth and elevation angle of the pth path. α_p denotes the complex channel gain of the pth path. $\mathbf{a}_R\left(\theta_p,\phi_p\right)\in\mathbb{C}^{N_R}$ and $\mathbf{a}_T\left(\theta_p,\phi_p\right)\in\mathbb{C}^{N_T}$ denote the array factors at the receiver and transmitter, respectively. Then the received signal at time t and subcarrier s is

$$y_t^s=\mathbf{w}_t^{s\,H}\widetilde{\mathbf{C}}_R\widetilde{\mathbf{A}}_R\mathbf{\Gamma}^s\widetilde{\mathbf{A}}_T^H\widetilde{\mathbf{C}}_T^H\mathbf{f}_t^s s_t^s+n_t^s, \tag{2}$$

This work has been partially supported by Google exploreCSR Grant 111Q90004C.

ZhaoJie Luo and Carrson C. Fung are both affiliated with the Department of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan. (e-mail: janny00kevin@gmail.com@gmail.com, c.fung@ieee.org)

where $\mathbf{w}_t^s \in \mathbb{C}^{N_R}$ is the analog linear combiner, $\mathbf{f} \in \mathbb{C}^{N_T}$ is the analog linear beamformer, $\boldsymbol{\Gamma}^s \triangleq \begin{bmatrix} \alpha_1^s & & \\ & \ddots & \\ & & \alpha_P^s \end{bmatrix} \in \mathbb{C}^{P\times P}$ is the channel gain matrix, which consists of different values at different subcarriers (even if the angles do not change). $\widetilde{\mathbf{C}}_R \triangleq \begin{bmatrix} \mathbf{C}_{R,1} & \cdots & \mathbf{C}_{R,P} \end{bmatrix} \in \mathbb{C}^{N_R\times PN_R}$ and $\widetilde{\mathbf{C}}_T \triangleq \begin{bmatrix} \mathbf{C}_{T,1} & \cdots & \mathbf{C}_{T,P} \end{bmatrix} \in \mathbb{C}^{N_T\times PN_T}$, with $\mathbf{C}_{R,p} \triangleq \mathbf{C}_{R,p}(\theta_p, \phi_p)$ and $\mathbf{C}_{T,p} \triangleq \mathbf{C}_{T,p}(\theta_p, \phi_p)$ contain mutual coupling matrices at the receiver and transmitter of all p paths, for $p = 1, \ldots, P$, respectively. $\widetilde{\mathbf{A}}_R \triangleq \text{blkdiag}\left(\mathbf{a}_{R,1}(\theta_1, \phi_1), \ldots \mathbf{a}_R(\theta_P, \phi_P)\right) \in \mathbb{C}^{PN_R\times P}$ and $\widetilde{\mathbf{A}}_T \triangleq \text{blkdiag}\left(\mathbf{a}_{T,1}(\theta_1, \phi_1), \ldots \mathbf{a}_T(\theta_P, \phi_P)\right) \in \mathbb{C}^{PN_T\times P}$ are block diagonal matrices that contain the array factors of all p paths at the receiver and transmitter, respectively. Assuming $s_t^s = 1$, using the property of Kronecker product $vec(\mathbf{ABC}) = (\mathbf{B}^T \otimes \mathbf{A})vec(\mathbf{b})$, the column-wise Khatri-Rao product property $(\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\circledast\mathbf{D}) = (\mathbf{AC})\circledast(\mathbf{BD})$, (2) can be rewritten as

$$\begin{aligned} y_t^s &= vec\left(\mathbf{w}_t^{s\,H}\widetilde{\mathbf{C}}_R\widetilde{\mathbf{A}}_R\boldsymbol{\Gamma}^s\widetilde{\mathbf{A}}_T^H\widetilde{\mathbf{C}}_T^H\mathbf{f}_t^s\right) + n_t^s \\ &= \left(\mathbf{f}_t^{s\,T}\widetilde{\mathbf{C}}_T^*\widetilde{\mathbf{A}}_T^* \circledast \mathbf{w}_t^{s\,H}\widetilde{\mathbf{C}}_R\widetilde{\mathbf{A}}_R\right)\boldsymbol{\alpha}^s + n_t^s \\ &= \left(\mathbf{f}_t^{s\,T}\widetilde{\mathbf{C}}_T^* \otimes \mathbf{w}_t^{s\,H}\widetilde{\mathbf{C}}_R\right)\left(\widetilde{\mathbf{A}}_T^* \circledast \widetilde{\mathbf{A}}_R\right)\boldsymbol{\alpha}^s + n_t^s \\ &= \underbrace{\left(\mathbf{f}_t^{s\,T} \otimes \mathbf{w}_t^{s\,H}\right)}_{1\times N_TN_R}\underbrace{\left(\widetilde{\mathbf{C}}_T^* \otimes \widetilde{\mathbf{C}}_R\right)}_{N_TN_R\times P(N_T+N_R)}\underbrace{\left(\widetilde{\mathbf{A}}_T^* \circledast \widetilde{\mathbf{A}}_R\right)}_{P(N_T+N_R)\times P}\boldsymbol{\alpha}^s + n_t^s, \end{aligned}$$

where $\boldsymbol{\alpha}^s = \begin{bmatrix} \alpha_1^s & \ldots & \alpha_P^s \end{bmatrix}^T \in \mathbb{C}^P$. Stacking T number of received signals in time, the received signal at the sth subcarrier is written as

$$\begin{aligned} \mathbf{y}^s &= \underbrace{\begin{bmatrix} \mathbf{f}_1^{s\,T} \otimes \mathbf{w}_1^{s\,H} \\ \vdots \\ \mathbf{f}_T^{s\,T} \otimes \mathbf{w}_T^{s\,H} \end{bmatrix}}_{T\times N_TN_R}\left(\widetilde{\mathbf{C}}_T^* \otimes \widetilde{\mathbf{C}}_R\right)\left(\widetilde{\mathbf{A}}_T^* \circledast \widetilde{\mathbf{A}}_R\right)\boldsymbol{\alpha}^s + \mathbf{n}^s \\ &= \boldsymbol{\Psi}^s\boldsymbol{\alpha}^s + \mathbf{n}^s \in \mathbb{C}^T, \end{aligned}$$

where $\mathbf{\Psi}^s \triangleq \begin{bmatrix} \mathbf{f}_1^{s\,T} \otimes \mathbf{w}_1^{s\,H} \\ \vdots \\ \mathbf{f}_T^{s\,T} \otimes \mathbf{w}_T^{s\,H} \end{bmatrix} \left(\widetilde{\mathbf{C}}_T^* \otimes \widetilde{\mathbf{C}}_R\right)\left(\widetilde{\mathbf{A}}_T^* \circledast \widetilde{\mathbf{A}}_R\right) \in \mathbb{C}^{T\times P}$. As a result, stacking all S number of subcarriers (or snapshots) will result in

$$\mathbf{Y} = \begin{bmatrix} y_1^1 & \cdots & y_1^S \\ \vdots & \ddots & \vdots \\ y_T^1 & \cdots & y_T^S \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{\Psi}^1 & \cdots & \mathbf{\Psi}^S \end{bmatrix}}_{T\times PS} \underbrace{\begin{bmatrix} \boldsymbol{\alpha}^1 & & \\ & \ddots & \\ & & \boldsymbol{\alpha}^S \end{bmatrix}}_{PS\times S}$$
$$= \begin{bmatrix} \mathbf{\Psi}^1\boldsymbol{\alpha}^1 & \cdots & \mathbf{\Psi}^S\boldsymbol{\alpha}^S \end{bmatrix} \in \mathbb{C}^{T\times S}. \tag{3}$$

That is, if the autocorrelation of $\mathbf{y}^s$, $\mathbf{R}_\mathbf{y} = E[\mathbf{y}\mathbf{y}^H]$ needs to be estimated, this can be done by

$$\mathbf{R}_\mathbf{y} \approx \frac{1}{S}\mathbf{Y}\mathbf{Y}^H \in \mathbb{C}^{T\times T}.$$

We may be able to use the BSLB [3] (or a neural network version of it) to solve our estimation problem. Carrson suggests to first look at [4] and [5] for background on Bayesian compressive sensing and how it can be used for DoA estimation with planar arrays.

B. *Score-Guided Diffusion Approach for Mutual Coupling and DoA Estimation*

We will first approach the estimation in the context of array signal processing (instead of communications) by first considering estimating the mutual coupling matrix and P DoAs (at the receiver) ϕ_p. Assume the array of antennas are aligned in a uniform linear array (ULA) and only a single snapshot is used. Using the same notation as in (1) and (2), then received signal can be written as

$$\mathbf{y} = \sum_{p=1}^{P} \alpha_p \mathbf{C}_R(\phi_p)\mathbf{a}_R(\phi_p)\mathbf{s} + \mathbf{n}$$
$$=, \tag{4}$$

1) Assume $\mathbf{C}(\phi_p)$ *to be independent of* θ_p*:* In this case, (4) can be simplified as

$$\mathbf{y} = \mathbf{C}_R \sum_{p=1}^{P} \alpha_p \mathbf{a}_R(\phi_p)\mathbf{s} + \mathbf{n},$$
$$= \mathbf{C}_R\mathbf{A}_R\mathbf{\Gamma}\mathbf{s} + \mathbf{n} \in \mathbb{C}^{N_R} \tag{5}$$

where $\mathbf{A}_R \triangleq \begin{bmatrix} \mathbf{a}_{R,1}(\phi_1) & \cdots & \mathbf{a}_{R,P}(\phi_P) \end{bmatrix} \in \mathbb{C}^{N_R\times P}$, $\mathbf{C}_R \in \mathbb{C}^{N_R\times N_R}$ and $\mathbf{\Gamma} = Diag(\alpha_1, \ldots, \alpha_P)$. Define $\mathbf{\Theta} \triangleq \{\mathbf{C}_R, \phi_p, \forall p\}$ to be the parameter that needs to be estimated.

A score-guided diffusion model [1], [2] learns (during the training process) to model a probability distribution $p\left(\mathbf{x}_0\right)$ by 1) gradually adding white Gaussian noise to the measurement $\mathbf{x}_t|_{t=0} \triangleq \mathbf{C}_R\mathbf{A}_R\mathbf{\Gamma}\mathbf{s}$ in the forward process until the result $\mathbf{x}_T$ at $t = T$ follows a standard Gaussian distribution with zero mean and unit variance, and 2) learning to denoise $\mathbf{x}_T$ step by step in the reverse process using the score function

$$s_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t|\,\mathbf{x}_0\right), \tag{6}$$

where $\boldsymbol{\theta}$ denotes the parameter of a model that models $\nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t|\,\mathbf{x}_0\right)$. Note that in pratice $s_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ is actually used to approximate the marginal distribution $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$, but it has been reported that the approximation works well empirically. In addition, even though in the current problem $\mathbf{x}_t \in \mathbb{C}^{N_R}$, however, assuming $\mathbf{x}_t$ is a vector that contains the stacking of the real and imaginary part of $\mathbf{C}_R\mathbf{A}_R\mathbf{\Gamma}\mathbf{s}$, then the gradient can be taken with respect to $\mathbf{x}_t$ instead of $\mathbf{x}_t^*$. During inference, the diffusion process is reversed but guided by the score function $s_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ and the likelihood function $p\left(\mathbf{y}|\,\mathbf{x}_t, \mathbf{\Theta}\right)$ that enforces consistency with the measured received signal $\mathbf{y}$.

In continuous form, the forward diffusion process can be described by stochastic differential equation [1]

$$d\mathbf{x}_t = f(\mathbf{x}_t, t)dt + g(t)d\mathbf{w}_t, \tag{7}$$

where $\mathbf{x}_t$ is the noisy latent data at time t that had evolved from "clean" data $\mathbf{x}_0 = \mathbf{C}_R\mathbf{A}_R\mathbf{s}$ to Gaussian noise. $f(\mathbf{x}_t, t)$ is a deterministic drift term that pushes the data $\mathbf{x}_t$ along some deterministic flow. $g(t)$ denotes the diffusion coefficient that controls the noise intensity so that $g(t)d\mathbf{w}_t$ adds the Gaussian noise as time increases, with $d\mathbf{w}_t$ denoting a standard Weiner process. Over time, the distribution of $\mathbf{x}_t$ approaches a standard Gaussian distribution.

In most score-based diffusion, a simple variance exploding (VE) or variance preserving (VP) SDE is used where (7) becomes

$$d\mathbf{x}_t = -\frac{1}{2}\beta(t)\mathbf{x}_t\, dt + \sqrt{\beta(t)}\, d\mathbf{w}_t, \tag{8}$$

where $\beta(t) > 0$ is a schedule that controls how fast noise is added. The drift term $-\frac{1}{2}\beta(t)\mathbf{x}_t\, dt$ shrinks $\mathbf{x}_t$ to $\mathbf{0}$, while the diffusion term $\sqrt{\beta(t)}\, d\mathbf{w}_t$ adds Gaussian noise. Then the closed-form transition probability distribution is

$$p_t\left(\mathbf{x}_t|\,\mathbf{x}_0\right) = \mathcal{N}\left(\alpha(t)\mathbf{x}_0, \sigma^2(t)\mathbf{I}\right),$$

where

$$\alpha(t) = \exp\left(-\frac{1}{2}\int_0^t \beta(u)\, du\right), \quad \sigma^2(t) = 1 - \alpha(t)^2,$$

so that at time t

$$\mathbf{x}_t = \alpha(t)\mathbf{x}_0 + \sigma(t)\boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{9}$$

From this equation, it is obvious that

$$\begin{aligned} p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) &= \frac{1}{\sqrt{2\pi\sigma^2(t)}} \exp\left\{-\frac{1}{2\sigma^2(t)}\left(\mathbf{x}_t - \alpha(t)\mathbf{x}_0\right)^T\left(\mathbf{x}_t - \alpha_t\mathbf{x}_0\right)\right\} \\ &= \frac{1}{\sqrt{2\pi\sigma^2(t)}} \exp\left\{-\frac{1}{2\sigma^2(t)}\left(\mathbf{x}_t^T\mathbf{x}_t - 2\alpha(t)\mathbf{x}_t^T\mathbf{x}_0 + \alpha(t)^2\mathbf{x}_0^T\mathbf{x}_0\right)\right\}. \end{aligned} \tag{10}$$

Then

$$\log p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) = \text{const} - \frac{1}{2\sigma^2(t)}\left(\mathbf{x}_t^T\mathbf{x}_t - 2\alpha(t)\mathbf{x}_t^T\mathbf{x}_0 + \alpha(t)^2\mathbf{x}_0^T\mathbf{x}_0\right).$$

Then taking the gradient of $p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)$ with respect to $\mathbf{x}_t$ yields

$$\begin{aligned} \nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) &= -\frac{1}{\sigma^2(t)}\mathbf{x}_t + \frac{\alpha(t)\mathbf{x}_0}{\sigma^2(t)} \\ &= \frac{\alpha(t)}{\sigma^2(t)}\mathbf{x}_0 - \frac{1}{\sigma^2(t)}\mathbf{x}_t. \end{aligned}$$

Therefore, the estimate of $\mathbf{x}_0$ based on $\mathbf{x}_t$ and t is

$$\begin{aligned} \widehat{\mathbf{x}}_0(\mathbf{x}_t, t) &= \frac{\mathbf{x}_t + \sigma^2(t)\nabla_{\mathbf{x}_t^*} \log p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)}{\alpha(t)} \\ &= \frac{\mathbf{x}_t + \sigma^2(t)s_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\alpha(t)}. \end{aligned}$$

In other words, once the score function has been learned (that carries information about the prior $p_t\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)$ (which again it is used to approximate $p_t(\mathbf{x}_t)$), then it can be sampled to get $\widehat{\mathbf{x}}_0(\mathbf{x}_t, t)$.

Going from the continuous SDE in (8) to discrete DDPM, the latter often assumes the forward transition distribution to be

$$q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) = \mathcal{N}\left(\sqrt{\bar{\alpha}(t)}\mathbf{x}_0, (1 - \bar{\alpha}(t))\mathbf{I}\right),$$

where $\bar{\alpha}(t) \triangleq \prod_{i=1}^{t} \alpha(t)$. Then from (9),

$$\alpha(t) \longrightarrow \sqrt{\bar{\alpha}(t)}, \quad \sigma^2(t) \longrightarrow 1 - \bar{\alpha}(t)$$

and (9) becomes

$$\mathbf{x}_t = \sqrt{\bar{\alpha}(t)}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}(t)}\boldsymbol{\varepsilon}. \tag{11}$$

Then

$$\widehat{\mathbf{x}}_0(\mathbf{x}_t, t) = \frac{\mathbf{x}_t + (1 - \bar{\alpha}(t))\, s_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}(t)}} \tag{12}$$

Using Bayes' rule, the reverse SDE

$$\begin{aligned} d\mathbf{x}_t &= \left[f(\mathbf{x}_t,t) - g(t)^2 \nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t | \, \mathbf{y}\right)\right] \, dt + g(t) \, d\bar{\mathbf{w}}_t \\ &= \left[f(\mathbf{x}_t,t) - g(t)^2 \left(\nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t\right) + \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)\right)\right] \, dt + g(t) \, d\bar{\mathbf{w}}_t \\ &= \left[f(\mathbf{x}_t,t) - g(t)^2 \left(s_{\boldsymbol{\theta}}\left(\mathbf{x}_t,t\right) + \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)\right)\right] \, dt + g(t) \, d\bar{\mathbf{w}}_t, \end{aligned} \tag{13}$$

where $\mathbf{s}_{\boldsymbol{\theta}}\left(\mathbf{x}_t,t\right) \approx \nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t\right)$ is the learned prior. The $-g(t)^2 \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)$ is the guidance that forces the reverse trajectory toward samples compatible with the measurements $\mathbf{y}$. $d\bar{\mathbf{w}}_t$ denotes Brownian motion in the reverse timed direction. Using variance preserving SDE, (13) becomes

$$\begin{aligned} d\mathbf{x}_t &= \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t) \nabla_{\mathbf{x}_t} \log p_t\left(\mathbf{x}_t | \, \mathbf{y}\right)\right] \, dt + \sqrt{\beta(t)} \, d\bar{\mathbf{w}}_t \\ &= \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)\left(\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)\right)\right] \, dt + \sqrt{\beta(t)} \, d\bar{\mathbf{w}}_t \\ &= \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t) s_{\boldsymbol{\theta}}(\mathbf{x}_t,t) - \beta(t) \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)\right] \, dt + \sqrt{\beta(t)} \, d\bar{\mathbf{w}}_t, \end{aligned} \tag{14}$$

where $\beta(t) \triangleq 1 - \alpha(t)$.

The gradient $\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right)$ can be obtained by using the chain rule

$$\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right) = \left(\frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t}\right)^T \nabla_{\mathbf{x}_0} \log p\left(\mathbf{y} | \, \widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right),$$

where $\frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t}$ is expressed using numerator layout (that is why there is a transpose), and

$$\nabla_{\mathbf{x}_0} \log p\left(\mathbf{y} | \, \widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right) = \sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0)$$

assuming $\mathbf{y} = \widehat{\mathbf{x}}_0 + \mathbf{n}$. From (12), the derivative

$$\begin{aligned} \frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t} &= \frac{1}{\sqrt{\bar{\alpha}(t)}}\left(\mathbf{I} + (1 - \bar{\alpha}(t))\frac{\partial s_{\boldsymbol{\theta}}(\mathbf{x}_t,t)}{\partial \mathbf{x}_t}\right) \\ &\approx \frac{1}{\sqrt{\bar{\alpha}(t)}}\mathbf{I}, \end{aligned}$$

where the approximation is made because it is difficult to compute $\frac{\partial s_{\boldsymbol{\theta}}(\mathbf{x}_t,t)}{\partial \mathbf{x}_t}$ FIND REFERENCE FOR THIS. Hence

$$\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right) = \frac{1}{\sqrt{\bar{\alpha}(t)}}\sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0).$$

Very often FIND REFERENCE, a tunable $\lambda(t) = \frac{\gamma g(t)^2}{\sqrt{\bar{\alpha}(t)}}$ is added so that gudiance term in (13) becomes

$$g(t)^2 \nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} | \, \mathbf{x}_t; \boldsymbol{\Theta}\right) = \lambda(t)\sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0).$$

However, $\lambda(t)$ requires careful tuning. A large $\lambda(t)$ will overfit on the measurement (i.e. ignore the prior) and a small $\lambda(t)$ will ignore the measurements. Hence, (13) becomes

$$d\mathbf{x}_t = \left[f(\mathbf{x}_t,t) - g(t)^2 s_{\boldsymbol{\theta}}\left(\mathbf{x}_t,t\right) - \lambda(t)\sigma^{-2}(\mathbf{y}-\widehat{\mathbf{x}}_0)\right]\, dt + g(t)\, d\bar{\mathbf{w}}_t \tag{15}$$

To discretize (14), supposed a discrete step size Δt is taken. Then going from t to $t-\Delta t$, using the Euler-Maruyama discrete reverse SDE with guidance [3], (14) becomes

$$\mathbf{x}_{t-\Delta t} = \mathbf{x}_t + \left[-\frac{1}{2}\beta(t)\mathbf{x}_t - \beta(t)s_{\boldsymbol{\theta}}(\mathbf{x}_t,t) - \beta(t)\nabla_{\mathbf{x}_t}\log p\left(\mathbf{y}|\,\mathbf{x}_t;\boldsymbol{\Theta}\right)\right]\Delta t + \sqrt{\beta(t)}\sqrt{\Delta t}\boldsymbol{\varepsilon}. \tag{16}$$

Unfortunately, the Euler-Maruyama is known to not render as precise estimation of $\mathbf{x}_0$ as the epsilon-net (DDIM), while the latter is also more stable and faster. It also provides deterministic refinement toward a clean estimate of $\mathbf{x}_0$. he ε-net (DDIM) approach removes the stochastic term to yield deterministic that traces a single trajectory from $\mathbf{x}_T$ to $\mathbf{x}_0$.

Expectation-Maximization (EM) method

Since the goal is to find $\boldsymbol{\Theta}$, there are several ways to do so. The expectation-maximization method is a natural choice as the $\widehat{\mathbf{x}}_0$ can be regarded as a "complete" data obtained above and treated as the E-step. Suppose S samples of $\widehat{\mathbf{x}}_0$ are drawn from the score-guided diffusion model, then the M-step is regarded as maximizing the average of the likelihood function and can be formulated as

$$\boldsymbol{\Theta}^{(i+1)} = \arg\max_{\boldsymbol{\Theta}} \frac{1}{2S}\sum_{\ell=1}^{S}\log p\left(\mathbf{y}|\,\widehat{\mathbf{x}}_0^{(\ell)},\boldsymbol{\Theta}\right) + p(\boldsymbol{\Theta}) = \mathcal{L}(\boldsymbol{\Theta}), \tag{17}$$

where i is the EM iteration index. (17) can be solved by taking the gradient of $\mathcal{L}(\boldsymbol{\Theta})$, which is

$$\nabla\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{S}\sum_{\ell=1}^{S} Re\left\{\left(\partial\mu(\boldsymbol{\Theta})^{(\ell)}\right)^H \boldsymbol{\Sigma}^{-1}\left(\mathbf{y}-\mu(\boldsymbol{\Theta})^{(\ell)}\right)\right\} + \nabla_{\boldsymbol{\Theta}}\log p(\boldsymbol{\Theta}),$$

where $\mu(\boldsymbol{\Theta}) \triangleq \mathbf{C}_R\mathbf{A}_R$, $\partial\mu(\boldsymbol{\Theta})$ is the Jacobian of μ with respect to $\boldsymbol{\Theta}$. During implementation, need to make all complex vectors/matrices real, and use autograd in Pytorch for simulation. (17) maximizes the posterior distribution, which implies the *a priori* distribution about $\boldsymbol{\Theta}$ is needed. This may be obtained using the diffusion model again, or we can just ignore it, and simply maximizes the likelihood distribution.

Score-Guided Diffusion Model for Finding $\mathbf{C}_R$ and MUSIC/ESPRIT/SBL to find DOAs

Instead of finding an estimate of $\mathbf{x}_0$, this method uses the score-guide diffusion model to directly find $\mathbf{C}_R$. Assuming again that S snapshots of $\widehat{\mathbf{x}}_0$ are collected so that (I have again assumed $\mathbf{x}_0$ is real, i.e. we stacked the real and imaginary part of the actual $\mathbf{x}_0$ into a real-valued vector)

$$\begin{aligned}\mathbf{R}_{\mathbf{x}_0} &= E\left[\mathbf{x}_0\mathbf{x}_0^T\right] \\ &= \mathbf{C}_R\mathbf{A}_R\boldsymbol{\Gamma}E\left[\mathbf{s}\mathbf{s}^T\right]\boldsymbol{\Gamma}^T\mathbf{A}_R^T\mathbf{C}_R^T + \sigma^2\mathbf{I}_{N_R} \\ &= \mathbf{C}_R\mathbf{A}_R\boldsymbol{\Gamma}\mathbf{R}_{\mathbf{s}}\mathbf{A}_R^T\boldsymbol{\Gamma}^T\mathbf{C}_R^T + \sigma^2\mathbf{I}_{N_R} \\ &\approx \frac{1}{S}\sum_{\ell=1}^{S}\mathbf{x}_0^{(\ell)}\mathbf{x}_0^{(\ell)\,T} \\ &= \widehat{\mathbf{R}}_{\mathbf{x}_0}.\end{aligned} \tag{18}$$

From (18),

$$\log p\left(\widehat{\mathbf{R}}_{\mathbf{x}_0}\middle|\,\mathbf{C}_R\right) = \left\|\widehat{\mathbf{R}}_{x_0} - \mathbf{C}_R\mathbf{A}_R\boldsymbol{\Gamma}\mathbf{R}_{\mathbf{s}}\mathbf{A}_R^T\boldsymbol{\Gamma}^T\mathbf{C}_R^T - \sigma^2\mathbf{I}_{N_R}\right\|_R^2$$

so that $\nabla_{\mathbf{c}_{R,t}}\log p\left(\widehat{\mathbf{R}}_{\mathbf{x}_0}\middle|\,\mathbf{c}_{R,t};\boldsymbol{\Theta}\right)$ can be easily computed.

Similar to (16), the inverse SDE can be discretized using the Euler-Maruyama reverse SDE, define $\mathbf{c}_R = \begin{bmatrix} vec\left(Re\{\mathbf{C}_R\}\right) \\ vec\left(Im\{\mathbf{C}_R\}\right)\end{bmatrix}$, $s_\gamma(\mathbf{c}_{R,t},t) \triangleq \nabla_{\mathbf{c}_t}\log p_t\left(\mathbf{c}_{R,t}\middle|\,\mathbf{c}_{R,0}\right)$, and $\log p_t\left(\widehat{\mathbf{R}}\middle|\,\mathbf{c}_{R,t}\right)$ denoting the likelihood function. Then $\mathbf{c}_{R,0}$ can be estimated using the inverse SDE as

$$\mathbf{c}_{R,t-\Delta t} = \mathbf{c}_t + \left[-\frac{1}{2}\beta(t)\mathbf{c}_{R,t} - \beta(t)s_\gamma(\mathbf{c}_{R,t},t) - \beta(t)\nabla_{\mathbf{c}_{R,t}}\log p\left(\widehat{\mathbf{R}}_{\mathbf{x}_0}\middle|\,\mathbf{c}_{R,t};\boldsymbol{\Theta}\right)\right]\Delta t + \sqrt{\beta(t)}\sqrt{\Delta t}\boldsymbol{\varepsilon}. \tag{19}$$

After this is done, MUSIC/ESPRIT/SBL [4], [5] can then be used to find the DOAs.

C. DDIM Score-Guided Deterministic Diffusion Approach for Mutual Coupling and DoA Estimation

Unfortunately the discretized Euler Maruyama sampling approach did not yield good estimation results. Herein, the DDPM/DDIM score-guided deterministic diffusion model is proposed instead.

Following the VP SDE, define

$$\alpha_t \triangleq 1-\beta_t, \quad \bar{\alpha}_t \triangleq \prod_{i=1}^{t}\alpha_i,$$

as the noise schedule and the cumulative noise schedule, respectively. Since a noisy sample at time t can be written as (using $\mathbf{x}_t$ reparameterization)

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}, \tag{20}$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}\left(\mathbf{0},\mathbf{I}\right)$. Then a given clean $\mathbf{x}_0$, the forward marginal distribution is

$$q\left(\mathbf{x}_t\middle|\,\mathbf{x}_0\right) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right). \tag{21}$$

Hence

$$\boldsymbol{\varepsilon} = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1-\bar{\alpha}_t}}. \tag{22}$$

Due to (21),

$$\log q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) = \text{constant} - \frac{1}{2(1-\bar{\alpha}_t)} \left\|\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0\right\|_2^2. \tag{23}$$

The gradient of (23) w.r.t. $\mathbf{x}_t$ is

$$\begin{aligned}\nabla_{\mathbf{x}_t} \log q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) &= -\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{1-\bar{\alpha}_t} \\ &= -\frac{\sqrt{1-\bar{\alpha}_t}}{1-\bar{\alpha}_t}\boldsymbol{\varepsilon} \\ &= -\frac{\boldsymbol{\varepsilon}}{\sqrt{1-\bar{\alpha}_t}},\end{aligned} \tag{24}$$

where the 2nd equality is due to (22).

Note that

$$p_t(\mathbf{x}_t) = \int_{\mathbf{x}_0} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) p(\mathbf{x}_0) d\mathbf{x}_0,$$

where $p(\mathbf{x}_0)$ is the latent signal distribution. Hence its gradient becomes

$$\nabla_{\mathbf{x}_t} p_t(\mathbf{x}_t) = \int_{\mathbf{x}_0} p(\mathbf{x}_0) \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) d\mathbf{x}_0.$$

From Bayes' rule, $q\left(\mathbf{x}_0 \middle| \mathbf{x}_t\right) = \frac{q(\mathbf{x}_t|\mathbf{x}_0)p(\mathbf{x}_0)}{p_t(\mathbf{x}_t)}$, and $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = \frac{\nabla_{\mathbf{x}_t} p_t(\mathbf{x}_t)}{p_t(\mathbf{x}_t)}$, then

$$\begin{aligned}\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) &= \frac{1}{p_t(\mathbf{x}_t)} \int_{\mathbf{x}_0} p(\mathbf{x}_0) \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) d\mathbf{x}_0 \\ &= \frac{1}{p_t(\mathbf{x}_t)} \int_{\mathbf{x}_0} \frac{q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) p(\mathbf{x}_0)}{q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)} \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) d\mathbf{x}_0 \\ &= \int_{\mathbf{x}_0} \frac{q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) p(\mathbf{x}_0)}{p_t(\mathbf{x}_t)} \frac{\nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)}{q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)} d\mathbf{x}_0 \\ &= \int_{\mathbf{x}_0} \log \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) q\left(\mathbf{x}_0 \middle| \mathbf{x}_t\right) d\mathbf{x}_0 \\ &= E_{q(\mathbf{x}_0|\mathbf{x}_t)}\left[\log \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)\right]\end{aligned} \tag{25}$$

Recall that the score network parameterizes the gradient of the log of the marginal (prior) distribution as $s_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$. Then combining (25) and (24), the score function can be written as

$$\begin{aligned}s_{\boldsymbol{\theta}}(x_t, t) &= E_{q(\mathbf{x}_0|\mathbf{x}_t)}\left[\log \nabla_{\mathbf{x}_t} q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right)\right] \\ &= -E_{q(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{\boldsymbol{\varepsilon}}{\sqrt{1-\bar{\alpha}_t}}\right] \\ &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} E\left[\boldsymbol{\varepsilon} \middle| \mathbf{x}_t\right],\end{aligned} \tag{26}$$

where the last equality is true because from (22), $\boldsymbol{\varepsilon}$ depends on $\mathbf{x}_t$. Similar to the way $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ was parameterized using the score function, the conditional (posterior) mean noise $E\left[\boldsymbol{\varepsilon} \middle| \mathbf{x}_t\right]$ can also be parameterized by the ε-network as

$$\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx E\left[\boldsymbol{\varepsilon} \middle| \mathbf{x}_t\right]. \tag{27}$$

Then

$$s_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx -\frac{\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}, \tag{28}$$

or

$$\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = -\sqrt{1-\bar{\alpha}_t}\, s_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \tag{29}$$

which shows the connection between the score function and the noise model, ε-net. Substituting (29) into (20), then the estimate of $\mathbf{x}_0$ can be expressed as

$$\begin{aligned} \widehat{\mathbf{x}}_0 &= \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\widehat{\boldsymbol{\varepsilon}}\right) \\ &= \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right) \\ &= \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t + (1-\bar{\alpha}_t)\, s_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right), \end{aligned} \tag{30}$$

which is identical to (12). Since $\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} \middle| \mathbf{x}_t; \boldsymbol{\Theta}\right)$ is needed in the $\boldsymbol{\varepsilon}$-guided diffusion model, this can be obtained using the chain rule

$$\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} \middle| \mathbf{x}_t; \boldsymbol{\Theta}\right) = \left(\frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t}\right)^T \nabla_{\mathbf{x}_0} \log p\left(\mathbf{y} \middle| \widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right),$$

where $\frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t}$ is expressed using numerator layout (that is why there is a transpose), and

$$\nabla_{\mathbf{x}_0} \log p\left(\mathbf{y} \middle| \widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right) = \sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0)$$

assuming $\mathbf{y} = \widehat{\mathbf{x}}_0 + \mathbf{n}$. From (30), the derivative

$$\begin{aligned} \frac{\partial \widehat{\mathbf{x}}_0}{\partial \mathbf{x}_t} &= \frac{1}{\sqrt{\bar{\alpha}(t)}} \left(\mathbf{I} - \sqrt{1-\bar{\alpha}(t)}\frac{\partial \varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\partial \mathbf{x}_t}\right) \\ &\approx \frac{1}{\sqrt{\bar{\alpha}(t)}}\mathbf{I}, \end{aligned}$$

where the approximation is made because it is difficult to compute $\frac{\partial \varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\partial \mathbf{x}_t}$. Hence

$$\nabla_{\mathbf{x}_t} \log p\left(\mathbf{y} \middle| \mathbf{x}_t; \boldsymbol{\Theta}\right) = \frac{1}{\sqrt{\bar{\alpha}(t)}}\sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0).$$

The guidance from the likelihood function is formed via gradient ascent

$$\begin{aligned} \widehat{\mathbf{x}}_0^{\text{guided}} &= \widehat{\mathbf{x}}_0 + \eta_{\mathbf{x}_0} \nabla_{\mathbf{x}_0} \log p\left(\mathbf{y} \middle| \widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right) \\ &= \widehat{\mathbf{x}}_0 + \eta_{\mathbf{x}_0} \sigma^{-2}(\mathbf{y} - \widehat{\mathbf{x}}_0). \end{aligned} \tag{31}$$

From (22) and using (31), define

$$\begin{aligned}\boldsymbol{\varepsilon}_{\text{mapped}} &\triangleq \boldsymbol{\varepsilon} \\ &= \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\widehat{\mathbf{x}}_0^{\text{guided}}}{\sqrt{1-\bar{\alpha}_t}}.\end{aligned} \tag{32}$$

Then one full step of the DDPM/DDIM-guided update becomes

$$\begin{aligned}\mathbf{x}_{t'} &= \sqrt{\bar{\alpha}_{t'}}\widehat{\mathbf{x}}_0^{\text{guided}} + \sqrt{1-\bar{\alpha}_{t'}}\boldsymbol{\varepsilon}_{\text{mapped}} \\ &= \sqrt{\bar{\alpha}_{t'}}\left(\widehat{\mathbf{x}}_0 + \eta_{\mathbf{x}_0}\sigma^{-2}(\mathbf{y}-\widehat{\mathbf{x}}_0)\right) + \sqrt{1-\bar{\alpha}_{t'}}\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\widehat{\mathbf{x}}_0^{\text{guided}}}{\sqrt{1-\bar{\alpha}_t}},\end{aligned} \tag{33}$$

where t' is a lower time index than t, which implies $\bar{\alpha}_{t'} < \bar{\alpha}_t$.

In summary, the algorithm is as follows

1. Train ε-net $\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ to predict noise
2. Compute $\widehat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right)$ as in (30)
3. Compute $\nabla_{\mathbf{x}_0} p\left(\mathbf{y}|\,\widehat{\mathbf{x}}_0; \boldsymbol{\Theta}\right) = \sigma^{-2}\left(\mathbf{y}-\widehat{\mathbf{x}}_0\right)$
4. Using gradient ascent $\widehat{\mathbf{x}}_0^{\text{guided}} = \widehat{\mathbf{x}}_0 + \eta_{\mathbf{x}_0}\sigma^{-2}\left(\mathbf{y}-\widehat{\mathbf{x}}_0\right)$ to update $\widehat{\mathbf{x}}_0^{\text{guided}}$ as stated in (31)
5. Compute $\boldsymbol{\varepsilon}_{\text{mapped}} = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\widehat{\mathbf{x}}_0^{\text{guided}}}{\sqrt{1-\bar{\alpha}_t}}$ as in (32)
6. Update $\mathbf{x}_{t'} = \sqrt{\bar{\alpha}_{t'}}\widehat{\mathbf{x}}_0^{\text{guided}} + \sqrt{1-\bar{\alpha}_{t'}}\boldsymbol{\varepsilon}_{\text{mapped}}$ as in (33)

After the model has predicted L snapshots of $\widehat{\mathbf{x}}_0$ (L snapshots of $\mathbf{y}$ is also collected during the training phase to train the model), it is used to generate the covariance of $\widehat{\mathbf{x}}_0$ denoted as $\mathbf{R}_{\widehat{\mathbf{x}}_0}$. Then define $\mathbf{R}_{\text{model}} \triangleq \mathbf{C}_R\mathbf{A}_R\mathbf{A}_R^H\mathbf{C}_R^H$. Then the estimate of the DOAs and $\mathbf{C}_R$ is found by solving

$$\min_{\mathbf{C}_R, \phi_p, \forall p} \quad \|\mathbf{R}_{\widehat{\mathbf{x}}_0} - \mathbf{R}_{\text{model}}\|_F^2 \tag{34}$$

using alternating minimization. This can be done by taking the gradient of the objective in (34) w.r.t. ϕ_p and $\mathbf{C}_R$ and using gradient descent to find the solution. To make the estimate of ϕ_p, $\forall p$ to be more accurate, MUSIC or ESPRIT can be used to initialize the gradient descent. This has proven to provide much better accuracy than randomly initializing the gradient descent.

D. Differences between Score-Guided DDPM and DDIM

The main difference between DDPM and DDIM lies in the reverse process. First of all, suppose

- β_t denotes the forward variance at time t
- $\alpha_t \triangleq 1 - \beta_t$
- $\bar{\alpha}_t \triangleq \prod_{i=1}^{t} \alpha_i$ denotes the accumulated variance

- Let the forward marginal distribution be denoted as $q\left(\mathbf{x}_t \middle| \mathbf{x}_0\right) \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, \left(1-\bar{\alpha}_t\right)\mathbf{I}\right)$, implying that $q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_0\right) \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0, (1-\bar{\alpha}_{t-1})\mathbf{I}\right)$
- Let the forward Mardov step distribution be $q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}\right) \sim \mathcal{N}\left(\sqrt{\alpha_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right)$

To establish the Markovian relationship from $\mathbf{x}_0$ to $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$, from the last and second to last bullet,

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\varepsilon}_t, \tag{35}$$

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta} \tag{36}$$

$$\Longrightarrow \mathbf{x}_t = \sqrt{\alpha_t}\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{\alpha_t}\sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta} + \sqrt{\beta_t}\boldsymbol{\varepsilon}_t, \tag{37}$$

where $\boldsymbol{\varepsilon}_t$ and $\boldsymbol{\eta}$ follow $\mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$. (36) is obtained from (35) by recursively updating it until $t-1=0$ and $t=t-1$. Hence, it is clear that $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$, conditioned on $\mathbf{x}_0$, are jointly Gaussian. From (35) and (36), the conditional distributions are

$$q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}\right) = \frac{1}{\sqrt{2\beta_t}} \exp\left\{-\frac{1}{2\beta_t}\left\|\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1}\right\|_2^2\right\} \tag{38}$$

$$q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_0\right) = \frac{1}{\sqrt{2(1-\bar{\alpha}_{t-1})}} \exp\left\{-\frac{1}{2\left(1-\bar{\alpha}_{t-1}\right)}\left\|\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\right\|_2^2\right\}, \tag{39}$$

respectively. From Bayes' rule, the conditional distribution

$$\begin{aligned} q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_t, \mathbf{x}_0\right) &\propto q\left(\mathbf{x}_{t-1}, \mathbf{x}_t \middle| \mathbf{x}_0\right) \\ &= q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}, \mathbf{x}_0\right) q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_0\right) \\ &= q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}\right) q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_0\right), \end{aligned} \tag{40}$$

where the last equality is true because of the Markovian assumption. Since $q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_t, \mathbf{x}_0\right)$ is proportional to two forward conditional distributions, hence, it is another way to express the forward conditional distribution (even though it is the distribution of $\mathbf{x}_{t-1}$ given $\mathbf{x}_t$, which sounds like the reverse conditional distribution). In the DDPM and DDIM, the goal is to use $p_{\boldsymbol{\theta}}\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_t\right)$ is approximate $q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_t, \mathbf{x}_0\right)$.

The **forward** process conditional distribution $q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_t, \mathbf{x}_0\right)$ is also Gaussian distributed. This is because the distribution is a product of two Gaussians as highlighted in (40). Specially, since we need an expression for the distribution of $\mathbf{x}_{t-1}$, when multiplying $q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}\right)$ and $q\left(\mathbf{x}_{t-1} \middle| \mathbf{x}_0\right)$, we need to complete the squares based on the quadratic of $\mathbf{x}_{t-1}$ in the exponent.

That is, the exponent of the product is

$$
\begin{aligned}
&-\frac{1}{2\beta_t}\left\|\mathbf{x}_t-\sqrt{\alpha_t}\mathbf{x}_{t-1}\right\|_2^2-\frac{1}{2\left(1-\bar{\alpha}_{t-1}\right)}\left\|\mathbf{x}_{t-1}-\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\right\|_2^2\\
&=-\frac{\alpha_t}{2\beta_t}\mathbf{x}_{t-1}^T\mathbf{x}_{t-1}+\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_{t-1}^T\mathbf{x}_t-\frac{1}{2\beta_t}\mathbf{x}_t^T\mathbf{x}_t\\
&\quad-\frac{1}{2\left(1-\bar{\alpha}_{t-1}\right)}\mathbf{x}_{t-1}^T\mathbf{x}_{t-1}+\frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_{t-1}^T\mathbf{x}_0-\frac{\bar{\alpha}_{t-1}}{2\left(1-\bar{\alpha}_{t-1}\right)}\mathbf{x}_0^T\mathbf{x}_0\\
&=-\frac{1}{2}\left[\underbrace{\left(\frac{\alpha_t}{\beta_t}+\frac{1}{1-\bar{\alpha}_{t-1}}\right)}_{a}\left\|\mathbf{x}_{t-1}\right\|_2^2-2\mathbf{x}_{t-1}^T\underbrace{\left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t+\frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)}_{\mathbf{b}}\right]\\
&\quad+\text{constants involving } \mathbf{x}_t \text{ and } \mathbf{x}_0 \text{ which are unimportant and ignored)}\\
&=-\frac{1}{2}\left(a\left\|\mathbf{x}_{t-1}\right\|_2^2-2\mathbf{x}_{t-1}^T\mathbf{b}\right)+\text{const}\\
&=-\frac{1}{2a^{-1}}\left(\left\|\mathbf{x}_{t-1}\right\|_2^2-2\frac{\mathbf{b}}{a}\mathbf{x}_{t-1}^T+\frac{\|\mathbf{b}\|_2^2}{a^2}-\frac{\|\mathbf{b}\|_2^2}{a^2}\right)+\text{const}\\
&=-\frac{1}{2a^{-1}}\left\|\mathbf{x}_{t-1}-\frac{\mathbf{b}}{a}\right\|_2^2-\frac{\|\mathbf{b}\|_2^2}{a^2}+\text{const}\\
&=-\frac{1}{2a^{-1}}\left\|\mathbf{x}_{t-1}-\frac{\mathbf{b}}{a}\right\|_2^2+\text{const.}
\end{aligned}
$$

Since $\exp\{\text{const}\}$ is a constant that will only scale the amplitude of the Gaussian, so it does not affect the mean nor its variance, which are $\frac{\mathbf{b}}{a}$ and a^{-1}, respectively, according to the above derivation. Note that the inverse of the variance can be rewritten as

$$
\begin{aligned}
a &\triangleq \left(\frac{\alpha_t}{\beta_t}+\frac{1}{1-\bar{\alpha}_{t-1}}\right)\\
&=\frac{\alpha_t\left(1-\bar{\alpha}_{t-1}\right)+\beta_t}{\beta_t\left(1-\bar{\alpha}_{t-1}\right)}\\
&=\frac{1-\bar{\alpha}_t}{\beta_t\left(1-\bar{\alpha}_{t-1}\right)},
\end{aligned}
$$

where the last equality is true since $\beta\triangleq 1-\alpha_t$, so the numerator becomes $\alpha_t\left(1-\bar{\alpha}_{t-1}\right)+\beta_t=\alpha_t-\alpha\bar{\alpha}_{t-1}+1-\alpha_t=1-\alpha_t\bar{\alpha}_{t-1}=1-\bar{\alpha}_t$. Hence the variance of $q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t,\mathbf{x}_0\right)$ is

$$
a^{-1}=\frac{\beta_t\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\triangleq\widetilde{\beta}_t.
$$

Using $\widetilde{\beta}_t$ from above, the mean of $q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t,\mathbf{x}_0\right)$ can be rewritten as

$$
\begin{aligned}
\frac{\mathbf{b}}{a} &\triangleq \left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t+\frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)\times\frac{\beta_t\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\\
&=\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t+\frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}\mathbf{x}_0 \qquad (41)\\
&\triangleq\widetilde{\boldsymbol{\mu}}(\mathbf{x}_t,\mathbf{x}_0).
\end{aligned}
$$

Since $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}\right)$, $\widetilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0)$ can be rewritten as

$$\begin{aligned}\widetilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0) &= \frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t \\ &= \frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_t} \times \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}\right) + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t \\ &= \left(\frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}\left(1-\bar{\alpha}_t\right)} + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\right)\mathbf{x}_t - \frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{1-\bar{\alpha}_t}\sqrt{\bar{\alpha}_t}}\boldsymbol{\varepsilon}\end{aligned}$$

Hence

$$\begin{aligned}q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t, \mathbf{x}_0\right) &\sim \mathcal{N}(\widetilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t\mathbf{I}) \\ &= \mathcal{N}\left(\frac{\beta_t\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t,\quad \frac{\beta_t\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{I}\right).\end{aligned}$$

In the reverse process, the DDPM learns $p_{\boldsymbol{\theta}}\left(\mathbf{x}_t\middle|\,\mathbf{x}_t\right)$ to approximate $q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t, \mathbf{x}_0\right)$ by minimizing the their KL-divergence (or equivalent MSE). The learned model can either learned the noise that has been added in the reverse process by training what is known as the ϵ-net or learned the mean of $\mathbf{x}_{t-1}$ (in $q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t, \mathbf{x}_0\right)$).

the forward process for both is shown in (11) which is repeated here as

$$\mathbf{x}_t = \sqrt{\bar{\alpha}(t)}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}(t)}\boldsymbol{\varepsilon}.$$

Since $\boldsymbol{\varepsilon}$ is Gaussian, hence the forward process is linear and Gaussian, and the forward distribution

$$q\left(\mathbf{x}_{t-1}\middle|\,\mathbf{x}_t, \mathbf{x}_0\right) = \mathcal{N}\left(\widetilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t\mathbf{I}\right)$$

is also Gaussian with $\widetilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ and

$$\widetilde{\boldsymbol{\mu}}\left(\mathbf{x}_t, \mathbf{x}_0\right) = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1-\bar{\alpha}_t}}\right),$$

where the last term is just equal to $\boldsymbol{\varepsilon}$ in (22). Insert the forward expression for $\mathbf{x}_t$, which is

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon},$$

into $\widetilde{\boldsymbol{\mu}}\left(\mathbf{x}_t, \mathbf{x}_0\right)$, then $\widetilde{\boldsymbol{\mu}}\left(\mathbf{x}_t, \mathbf{x}_0\right)$ becomes

$$\widetilde{\boldsymbol{\mu}}\left(\mathbf{x}_t, \mathbf{x}_0\right) = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\varepsilon}.$$

Thus

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\varepsilon},$$

(because the mean of $\mathbf{x}_{t-1}$ equals $\widetilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0)$), which is just the same forward reparameterization equation as (11), except going to $t-1$ instead.

In DDPM, the true forward posterior distribution $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ cannot be

If $\mathbf{x}_t$ and the noise $\boldsymbol{\varepsilon}$ are known, then $\widehat{\mathbf{x}}_0$ is obtained via (30), which is repeated here as

$$\widehat{\mathbf{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \right),$$

where $\varepsilon_{\boldsymbol{\theta}}$ approximates $\boldsymbol{\varepsilon}$.

Suppose the forward process is described as (this is the same parameterization that is used in (11) except for relating $\mathbf{x}_{t-1}$ to $\mathbf{x}_0$)

$$\begin{aligned}\mathbf{x}_{t-1} &= \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1}} \boldsymbol{\varepsilon}' \\ &= \sqrt{\bar{\alpha}_{t-1}} \left(\frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon} \right) \right) + \sqrt{1 - \bar{\alpha}_{t-1}} \boldsymbol{\varepsilon}' \\ &= \frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}} \mathbf{x}_t + \sqrt{1 - \bar{\alpha}_{t-1}} \boldsymbol{\varepsilon}' - \frac{\sqrt{\bar{\alpha}_{t-1}}}{\sqrt{\bar{\alpha}_t}} \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\varepsilon}\end{aligned}$$

where the 2nd equality is obtained by substituting the estimate of $\mathbf{x}_0$ from (30) in place of $\mathbf{x}_0$ and $\boldsymbol{\varepsilon}' \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ (similar to $\boldsymbol{\varepsilon}$) is added to obtain $\mathbf{x}_{t-1}$ from $\mathbf{x}_0$.

In the DDPM sampler, it uses a stochastic reverse-time SDE

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \right) \boldsymbol{\varepsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \sigma_t \mathbf{z}, \tag{42}$$

where

$$\sigma_t^2 = \frac{1 - \alpha_{t-1}}{1 - \bar{\alpha}_t} (1 - \alpha_t)$$

with $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Since the noise term $\sigma_t \mathbf{z}$ is stochastic, hence, it makes the reverse process to be stochastic as well.

III. Conclusion

References

[1] Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon and B. Poole, “Score-based generative modeling through stochastic differential equations,” *Proc. of the Intl. Conf. on Learning Representation (ICLR)*, 2021.

[2] R. Zirvi, B. Tolooshams, and A. Anandkumar, “Diffusion state-guided projected gradient for inverse problems,” *Proc. of the Intl. Conf. on Learning Representations*, Singapore, Apr. 2025.

[3] W. Liu, X. Mao and Y. Wu, “The backward Euler-Maruyama method for invariant measures of stochastic differential equations with super-linear coefficients,” *Applied Numerical Mathematics*, vol. 184, pp. 137-150, Sep. 2022.

[4] S. Ji, Y. Xue and L. Carin, “Bayesian compressive sensing,” *IEEE Trans. on Signal Processing*, vol. 56(6), pp. 2346-2356, Jun. 2008.

[5] Z. Lin, Y. Chen, X. Liu, R. Jiang and B. Shen, “Adaptive beamforming design of planar array based on bayesian compressive sensing,” *IEEE Sensors Journal*, vol. 21(4), pp. 5185-8194, Feb. 2021.