Parallel and Flexible Sampling from Autoregressive Models via Langevin Dynamics

Vivek Jayaram and John Thickstun

PAUL G. ALLEN SCHOOL OF COMPUTER SCIENCE & ENGINEERING

Smoothing a Discretized Autoregressive Model

$$p(\mathbf{x}) = \prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{x}_{<i}), \text{ where } \mathbf{x}_i \in \{e_1, \ldots, e_d\} \subset \mathbb{R}.$$

We want to parallelize posterior sampling from an autoregressive model of data $\mathbf{x}_i$ taken from some finite set of d values (e.g. 8-bit audio samples or 24-bit pixels).

$$\mathbf{x}^{(t+1)} \equiv \mathbf{x}^{(t)} + \eta\nabla_{\mathbf{x}} \log p(\mathbf{x}^{(t)}|\mathbf{y}) + \sqrt{2\eta}\varepsilon_t$$
$$= \mathbf{x}^{(t)} + \eta\nabla_{\mathbf{x}} \left(\log p(\mathbf{x}^{(t)}) + \log p(\mathbf{y}|\mathbf{x}^{(t)})\right) + \sqrt{2\eta}\varepsilon_t.$$

Using Langevin dynamics, we can generate approximate posterior samples from an unconditional generative model $p(\mathbf{x})$, controlled by a conditional likelihood model $p(\mathbf{y}|\mathbf{x})$.

$$p_\sigma(\tilde{\mathbf{x}}) = (\phi_\sigma * p)(\tilde{\mathbf{x}}) = \int \phi_\sigma(\tilde{\mathbf{x}} - \mathbf{x})p(\mathbf{x})\,d\mathbf{x}.$$

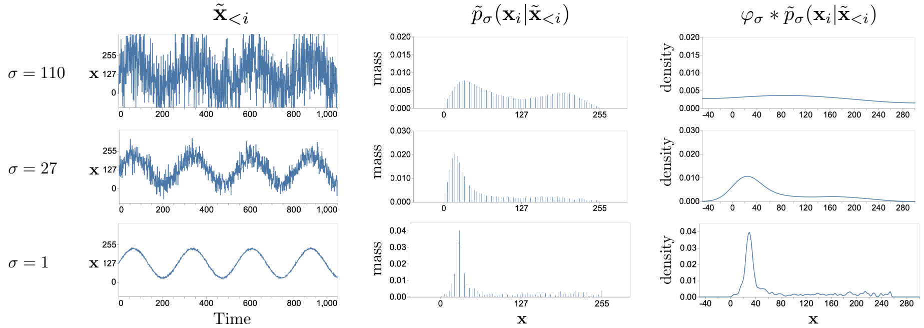
We can smooth a discrete distribution over discretized data by convolving with a Gaussian distribution $\phi_\sigma = \mathcal{N}(0, \sigma^2 I)$. This allows us to compute the gradients used for Langevin dynamics.

$$p_\sigma(\tilde{\mathbf{x}}) = \prod_{i=1}^{n} p_\sigma(\tilde{\mathbf{x}}_i|\tilde{\mathbf{x}}_{<i}) = \prod_{i=1}^{n} \left(\varphi_\sigma * \tilde{p}_\sigma(\,\cdot\,|\tilde{\mathbf{x}}_{<i})\right)(\tilde{\mathbf{x}}_i).$$

We can express the smoothed distribution as a product of Gaussian convolutions with fine-tuned models $\tilde{p}_\sigma(\mathbf{x}_i|\tilde{\mathbf{x}}_{<i})$ trained to predict values $\mathbf{x}_i$ given noisy history $\tilde{\mathbf{x}}_{<i} \sim \mathcal{N}(\mathbf{x}_{<i}, \sigma^2 I)$.

$$\left(\varphi_\sigma * \tilde{p}_\sigma(\,\cdot\,|\tilde{\mathbf{x}}_{<i})\right)(\tilde{\mathbf{x}}_i) = \sum_{k=1}^{d} \tilde{p}_\sigma(e_k|\tilde{\mathbf{x}}_{<i})\varphi_\sigma(\tilde{\mathbf{x}}_i - e_k).$$

The Gaussian convolutions can be calculated in a closed form given by a Gaussian mixture model.

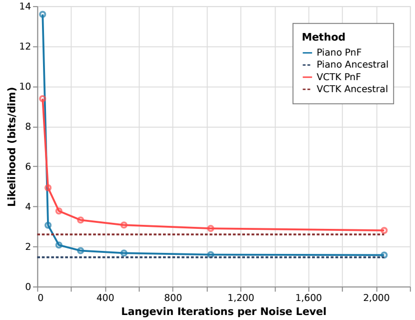


Given a noisy history $\tilde{\mathbf{x}}_{<i} = \mathbf{x}_{<i} + \varepsilon_{<i}$ (left column) where $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$, we train a model to predict the un-noised distribution over $\mathbf{x}_i \in \mathbb{R}$ (middle column). This distribution is discrete and non-differentiable in $\tilde{\mathbf{x}}$; we convolve with a Gaussian $\varphi_\sigma(t) = \mathcal{N}(t; 0, \sigma^2)$ to produce a continuous estimate of $\tilde{\mathbf{x}}_i$ (right column). We run Langevin dynamics on the continuous distribution, and gradually anneal the amount of smoothing σ (the noise level) to approximate the target distribution.

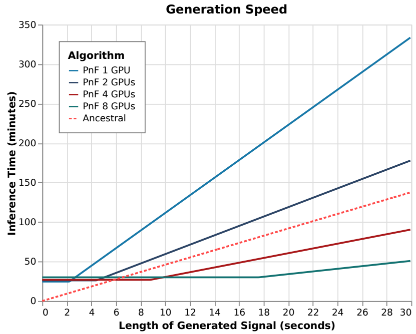
Project Webpage: https://grail.cs.washington.edu/projects/pnf-sampling/

GitHub Repo: https://github.com/vivjay30/pnf-sampling

Unconditional WaveNet Audio Sampling Results



As the number of iterations grows, the log-likelihood of our unconditional PnF sampler asymptotically matches the log-likelihood of ancestral samples.



For a fixed number of Langevin iterations, PnF sampling time is linear in the length of the generated sequence, but inverse-proportional to the number of parallel computing devices.

Conditional WaveNet Audio Sampling Results

Source Separation

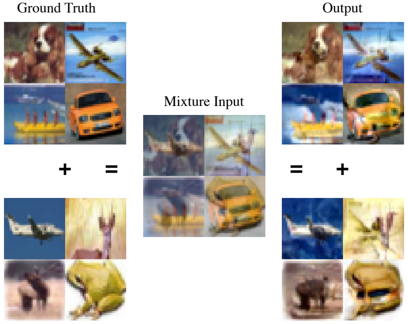
Algorithm	Test SI-SDR (dB) All	Piano	Voice
PnF (WaveNet)	17.07	13.92	20.25
Conv-Tasnet	17.48	20.02	15.50
Demucs	14.18	16.67	12.75

Super-Resolution

	Piano			Voice		
Ratio	Spline	KEE	PnF	Spline	KEE	PnF
4x	23.07	22.25	29.78	15.8	16.04	15.47
8x	13.58	15.79	23.49	10.7	11.15	10.03
16x	7.09	6.76	14.23	6.4	7.11	5.32

Qualitative PixelCNN++ Visual Sampling Results

Source Separation



2x Super-Resolution