SBGM: Score-Based Generative Models in JAX.

Jed Homer [1,2*]

1 University Observatory, Faculty for Physics, Ludwig-Maximilians-Universität München, Scheinerstrasse 1, München, Deustchland. **2** Munich Center for Machine Learning. * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

In partnership with

AMERICAN ASTRONOMICAL SOCIETY

This article and software are linked with research article DOI 10.3847/xxxxx <- update this with the DOI from AAS once you know it., published in the Astrophysical Journal <- The name of the AAS journal..

Summary

Diffusion models (Ho et al., 2020; Sohl-Dickstein et al., 2015; Song, Sohl-Dickstein, et al., 2021) have emerged as the dominant paradigm for generative modelling based on performance in a variety of tasks (Peebles & Xie, 2023a; Rombach et al., 2022). The advantages of accurate density estimation and high-quality samples of normalising flows (Grathwohl et al., 2018; Papamakarios et al., 2021), VAEs (Diederik P. Kingma & Welling, 2022) and GANs (Goodfellow et al., 2014) are subsumed into this method. Significant limitations exist on implicit and neural network based likelihood models with respect to modelling normalised probability distributions and sampling speed. Score-matching diffusion models are more efficient than previous generative model algorithms for these tasks. The diffusion process is agnostic to the data representation meaning different types of data such as audio, point-clouds, videos and images can be modelled.

Statement of need

Diffusion-based generative models (Ho et al., 2020; Sohl-Dickstein et al., 2015) are a method for sampling from high-dimensional distributions. A sub-class of these models, score-based diffusion generatives models (SBGMs, (Song, Sohl-Dickstein, et al., 2021)), permit exact-likelihood estimation via a change-of-variables associated with the forward diffusion process (Song, Durkan, et al., 2021). Diffusion models allow fitting generative models to high-dimensional data in a more efficient way than normalising flows since only one neural network model parameterises the diffusion process as opposed to a sequence of neural networks in typical normalising flow architectures. Whilst existing diffusion models (Ho et al., 2020; Diederik P. Kingma et al., 2023) allow for sampling, they are limited to innaccurate variational inference approaches for density estimation which limits their use for Bayesian inference. This code provides density estimation with diffusion models using GPU enabled ODE solvers in `jax` (Bradbury et al., 2018) and `diffrax` (Kidger, 2022). Similar codes (e.g. (Rozet, 2024)) exist for diffusion models but they do not implement log-likelihood calculations, various network architectures and parallelised ODE-sampling.

The software we present, `sbgm`, is designed to be used by researchers in machine learning and the natural sciences for fitting diffusion models with custom architectures for their research. These models can be fit easily with multi-accelerator training and inference within the code. Typical use cases for these kinds of generative models are emulator approaches (Spurio Mancini et al., 2022), simulation-based inference (Cranmer et al., 2020), field-level inference (Andrews et al., 2023) and general inverse problems (Feng et al., 2023; Feng & Bouman, 2024; Remy et al., 2023; Song et al., 2022) (e.g. image inpainting (Song, Sohl-Dickstein, et al., 2021) and denoising (Chung et al., 2022; Daras et al., 2024)). This code allows for seemless integration of diffusion models to these applications by providing data-generating models with easy conditioning of the data on any modality. Furthermore, the implementation in `equinox`

([Kidger & Garcia, 2021](#)) guarantees safe integration of `sbgm` with any other sampling libraries (e.g. BlackJAX Cabezas et al. (2024)) or `jax` (Bradbury et al., 2018) based codes.

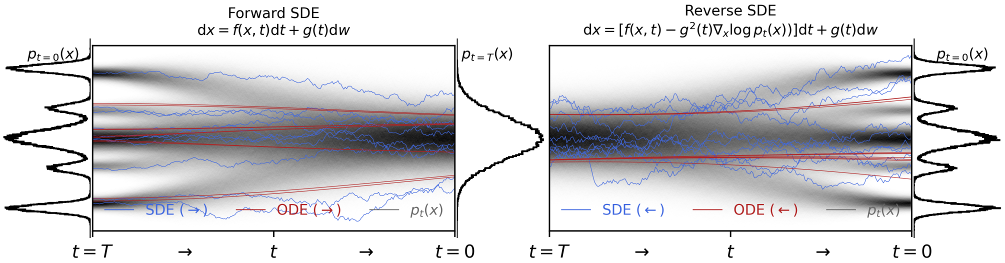

Figure 1: A diagram showing how to map data to a noise distribution (the prior) with an SDE, and reverse this SDE for generative modeling. One can also reverse the associated probability flow ODE, which yields a deterministic reverse process. Both the reverse-time SDE and probability flow ODE can be obtained by estimating the score.

Diffusion

Diffusion in the context of generative modelling describes the process of adding small amounts of noise sequentially to samples of data $\boldsymbol{x}$ (Sohl-Dickstein et al., 2015). A generative model for the data arises from training a neural network to reverse this process by subtracting the noise added to the data.

Score-based diffusion models (Song, Sohl-Dickstein, et al., 2021) model a forward diffusion process with Stochastic Differential Equations (SDEs) of the form

$$\mathrm{d}\boldsymbol{x} = f(\boldsymbol{x}, t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w},$$

where $f(\boldsymbol{x}, t)$ is a vector-valued function called the drift coefficient, $g(t)$ is the diffusion coefficient and $\mathrm{d}\boldsymbol{w}$ is a sample of noise $\mathrm{d}\boldsymbol{w} \sim \mathcal{G}[\mathrm{d}\boldsymbol{w}|\mathbf{0}, \mathbf{I}]$. This equation describes the infinitely many samples of noise along the diffusion time t that perturb the data. The diffusion path, defined by the SDE, begins at $t = 0$ and ends at $T = 0$ where the resulting distribution is then a multivariate Gaussian with mean zero and covariance $\mathbf{I}$.

The reverse of the SDE, mapping from multivariate Gaussian samples $\boldsymbol{x}(T)$ to samples of data $\boldsymbol{x}(0)$, is of the form

$$\mathrm{d}\boldsymbol{x} = [f(\boldsymbol{x}, t) - g^2(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})]\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w},$$

where the score function $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ is substituted with a neural network $\boldsymbol{s}_\theta(\boldsymbol{x}(t), t)$ for the sampling process. The network is fit by score-matching (Hyvärinen, 2005; Vincent, 2011) across the time span $[0, T]$. This network predicts the noise added to the image at time t with the forward diffusion process, in accordance with the SDE, and removes it. With a data-dimensional sample of Gaussian noise from the prior $p_T(\boldsymbol{x})$ (see Figure 1) one can reverse the diffusion process to generate data.

The reverse SDE may be solved with Euler-Murayama sampling (Song, Sohl-Dickstein, et al., 2021) (or other annealed Langevin sampling methods) which is featured in the code.

Likelihood calculations with diffusion models

Many of the applications of generative models depend on being able to calculate the likelihood of data. Song, Sohl-Dickstein, et al. (2021) show that any SDE may be converted into an ordinary differential equation (ODE) without changing the distributions, defined by the SDE, from which the noise is sampled from in the diffusion process (denoted $p_t(x)$ and shown in grey in Figure 1). This ODE is known as the probability flow ODE (Song, Sohl-Dickstein, et al., 2021; Song, Durkan, et al., 2021) and is written

$$\mathrm{d}\boldsymbol{x} = [f(\boldsymbol{x}, t) - g^2(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})]\mathrm{d}t = f'(\boldsymbol{x}, t)\mathrm{d}t.$$

This ODE can be solved with an initial-value problem. Starting with a data point $\boldsymbol{x}(0) \sim p(\boldsymbol{x})$, this point is mapped along the probability flow ODE path (see the right-hand side of Figure 1) to a sample from the multivariate Gaussian prior. This inherits the formalism of continuous normalising flows (Chen et al., 2019; Grathwohl et al., 2018) without the expensive ODE simulations used to train these models - allowing for a likelihood estimate based on diffusion models (Song, Durkan, et al., 2021). The initial value problem provides a solution $\boldsymbol{x}(T)$ and the change in probability along the path $\Delta = \log p(\boldsymbol{x}(0)) - \log p(\boldsymbol{x}(T))$ where $p(\boldsymbol{x}(T))$ is a simple multivariate Gaussian distribution.

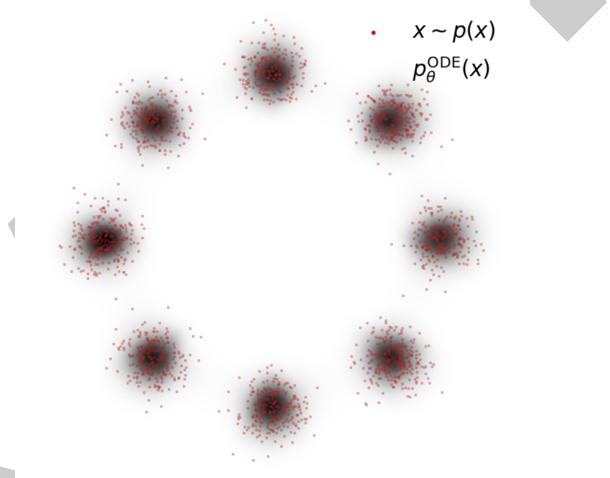

Figure 2: A diagram showing a log-likelihood calculation over the support of a Gaussian mixture model with eight components. Data is drawn (shown in red) from this mixture to train the diffusion model that gives the likelihood in gray. The log-likelihood is calculated using the ODE and a trained diffusion model.

The likelihood estimate under a score-based diffusion model is estimated by solving the change-of-variables equation for continuous normalising flows.

$$\frac{\partial}{\partial t} \log p(\boldsymbol{x}(t)) = \nabla_{\boldsymbol{x}} \cdot f(\boldsymbol{x}(t), t),$$

which gives the log-likelihood of a single datapoint $\boldsymbol{x}(0)$ as

$$\log p(\boldsymbol{x}(0)) = \log p(\boldsymbol{x}(T)) + \int_{t=0}^{t=T} \mathrm{d}t \; \nabla_{\boldsymbol{x}} \cdot f(\boldsymbol{x}, t).$$

The code implements these calculations also for the Hutchinson trace estimation method and (1990) that reduces the computational expense of the estimate. Figure 2 shows an example of a data-likelihood calculation using a trained diffusion model with the ODE associated from an SDE.

Implementations and future work

Diffusion models are defined in `sbgm` via a score-network model s_θ and an SDE. All the availble SDEs (variance exploding (VE), variance preserving (VP) and sub-variance preserving (SubVP) (Song, Sohl-Dickstein, et al., 2021)) in the literature of score-based diffusion models are available. We provide implementations for UNet (Ronneberger et al., 2015), Diffusion Transformers (Peebles & Xie, 2023b), MLP-Mixer (Tolstikhin et al., 2021) and Residual Network (He et al., 2015) models which are state-of-the-art for diffusion tasks. It is possible to fit score-based diffusion models to a conditional distribution $p(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{y})$ where in typical inverse problems $\boldsymbol{y}$ would be an image and $\boldsymbol{\pi}$ a set of parameters in a physical model for the data (**?**) (e.g. to solve inverse problems). The code is compatible with any model written in the `equinox` (Kidger & Garcia, 2021) framework. We are extending the code to provide transformer-based (Peebles & Xie, 2023a) and latent diffusion models (Rombach et al., 2022).

Acknowledgements

We thank the developers of the packages `jax` (Bradbury et al., 2018), `optax` (DeepMind et al., 2020), `equinox` (Kidger & Garcia, 2021) and `diffrax` (Kidger, 2022) for their work and for making their code available to the community.

References

and, M. F. H. (1990). A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, *19*(2), 433–450. https://doi.org/10.1080/03610919008812866

Andrews, A., Jasche, J., Lavaux, G., & Schmidt, F. (2023). Bayesian field-level inference of primordial non-gaussianity using next-generation galaxy surveys. *Monthly Notices of the Royal Astronomical Society*, *520*(4), 5746–5763. https://doi.org/10.1093/mnras/stad432

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs* (Version 0.3.13). http://github.com/jax-ml/jax

Cabezas, A., Corenflos, A., Lao, J., & Louf, R. (2024). *BlackJAX: Composable Bayesian inference in JAX*. https://arxiv.org/abs/2402.10797

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2019). *Neural ordinary differential equations*. https://arxiv.org/abs/1806.07366

Chung, H., Kim, J., Kim, S., & Ye, J. C. (2022). *Parallel diffusion models of operator and image for blind inverse problems*. https://arxiv.org/abs/2211.10656

Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, *117*(48), 30055–30062. https://doi.org/10.1073/pnas.1912789117

Daras, G., Dimakis, A. G., & Daskalakis, C. (2024). *Consistent diffusion meets tweedie: Training exact ambient diffusion models with noisy data*. https://arxiv.org/abs/2404.10177

DeepMind, Babuschkin, I., Baumli, K., Bell, A., Bhupatiraju, S., Bruce, J., Buchlovsky, P., Budden, D., Cai, T., Clark, A., Danihelka, I., Dedieu, A., Fantacci, C., Godwin, J., Jones, C., Hemsley, R., Hennigan, T., Hessel, M., Hou, S., … Viola, F. (2020). *The DeepMind JAX Ecosystem*. http://github.com/google-deepmind

Feng, B. T., & Bouman, K. L. (2024). *Variational bayesian imaging with an efficient surrogate score-based prior*. https://arxiv.org/abs/2309.01949

Feng, B. T., Smith, J., Rubinstein, M., Chang, H., Bouman, K. L., & Freeman, W. T. (2023). *Score-based diffusion models as principled priors for inverse imaging*. https://arxiv.org/abs/2304.11751

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). *Generative adversarial networks*. https://arxiv.org/abs/1406.2661

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., & Duvenaud, D. (2018). *FFJORD: Free-form continuous dynamics for scalable reversible generative models*. https://arxiv.org/abs/1810.01367

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition*. https://arxiv.org/abs/1512.03385

Ho, J., Jain, A., & Abbeel, P. (2020). *Denoising diffusion probabilistic models*. https://arxiv.org/abs/2006.11239

Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, *6*(24), 695–709. http://jmlr.org/papers/v6/hyvarinen05a.html

Kidger, P. (2022). *On neural differential equations*. https://arxiv.org/abs/2202.02435

Kidger, P., & Garcia, C. (2021). Equinox: Neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming Workshop at Neural Information Processing Systems 2021*.

Kingma, Diederik P., Salimans, T., Poole, B., & Ho, J. (2023). *Variational diffusion models*. https://arxiv.org/abs/2107.00630

Kingma, Diederik P., & Welling, M. (2022). *Auto-encoding variational bayes*. https://arxiv.org/abs/1312.6114

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2021). *Normalizing flows for probabilistic modeling and inference*. https://arxiv.org/abs/1912.02762

Peebles, W., & Xie, S. (2023a). *Scalable diffusion models with transformers*. https://arxiv.org/abs/2212.09748

Peebles, W., & Xie, S. (2023b). *Scalable diffusion models with transformers*. https://arxiv.org/abs/2212.09748

Remy, B., Lanusse, F., Jeffrey, N., Liu, J., Starck, J.-L., Osato, K., & Schrabback, T. (2023). Probabilistic mass-mapping with neural score estimation. *Astronomy &Amp; Astrophysics*, *672*, A51. https://doi.org/10.1051/0004-6361/202243054

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). *High-resolution image synthesis with latent diffusion models*. https://arxiv.org/abs/2112.10752

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-net: Convolutional networks for biomedical image segmentation*. https://arxiv.org/abs/1505.04597

Rozet, F. (2024). *Azula: Diffusion models in PyTorch* (Version 0.2.04). https://github.com/probabilists/azula

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). *Deep unsupervised learning using nonequilibrium thermodynamics*. https://arxiv.org/abs/1503.03585

Song, Y., Durkan, C., Murray, I., & Ermon, S. (2021). *Maximum likelihood training of score-based diffusion models*. https://arxiv.org/abs/2101.09258

Song, Y., Shen, L., Xing, L., & Ermon, S. (2022). *Solving inverse problems in medical imaging with score-based generative models*. https://arxiv.org/abs/2111.08005

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021). *Score-based generative modeling through stochastic differential equations*. https://arxiv.org/abs/2011.13456

Spurio Mancini, A., Piras, D., Alsing, J., Joachimi, B., & Hobson, M. P. (2022). <Scp>CosmoPower</scp>: Emulating cosmological power spectra for accelerated bayesian inference from next-generation surveys. *Monthly Notices of the Royal Astronomical Society*, *511*(2), 1771–1788. https://doi.org/10.1093/mnras/stac064

Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., & Dosovitskiy, A. (2021). *MLP-mixer: An all-MLP architecture for vision*. https://arxiv.org/abs/2105.01601

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, *23*(7), 1661–1674. https://doi.org/10.1162/NECO_a_00142