

# SBGM: Score-Based Generative Models in JAX.

Jed Homer  1,2\*

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

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

## Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

## License

Authors of papers retain copyright  
and release the work under a  
Creative Commons Attribution 4.0  
International License ([CC BY 4.0](#)).

## In partnership with



This article and software are linked  
with research article DOI  
[10.3847/xxxxx](https://doi.org/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

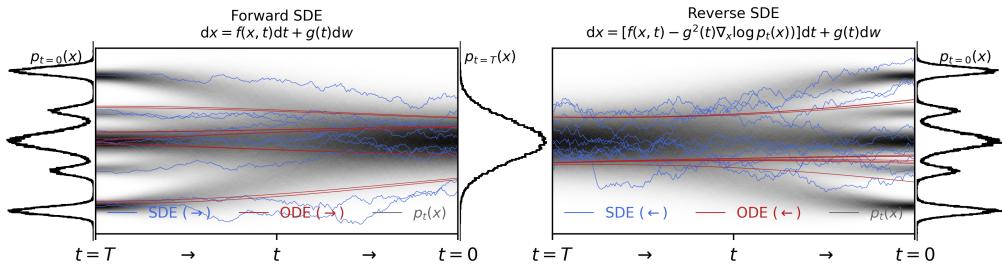
Diffusion models (Ho et al., 2020; Sohl-Dickstein et al., 2015; Song, Sohl-Dickstein, et al., 2021) have emerged as the dominant paradigm for generative modelling based on performance in a variety of tasks (Peebles & Xie, 2023a; Rombach et al., 2022). The advantages of accurate density estimation and 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 generative 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 inaccurate 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 diffraex (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 computations for optimisation and SDE/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 routines within the code (with demonstration examples provided). 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 seamless integration of diffusion models to these applications by providing data-generating models with easy conditioning of the data on any modality (e.g. images, audio

<sup>43</sup> or model parameters). Furthermore, the implementation in equinox ([Kidger & Garcia, 2021](#))  
<sup>44</sup> 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

<sup>46</sup> Diffusion in the context of generative modelling describes the process of adding small amounts  
<sup>47</sup> of noise sequentially to samples of data  $x$  ([Sohl-Dickstein et al., 2015](#)). A generative model  
<sup>48</sup> for the data arises from training a neural network to reverse this process by subtracting the  
<sup>49</sup> noise added to the data.

<sup>50</sup> Score-based diffusion models ([Song, Sohl-Dickstein, et al., 2021](#)) model a forward diffusion  
<sup>51</sup> process with Stochastic Differential Equations (SDEs) of the form

$$dx_t = f(x_t, t)dt + g(t)d\omega_t,$$

<sup>52</sup> where  $f(x_t, t)$  is a vector-valued function called the drift coefficient,  $g(t)$  is the diffusion  
<sup>53</sup> coefficient and  $d\omega_t$  is a sample of noise  $d\omega_t \sim \mathcal{G}[d\omega_t | \mathbf{0}, \mathbf{I}]$ . This equation describes the  
<sup>54</sup> infinitely many samples of noise along the diffusion time  $t$  that perturb the data. The diffusion  
<sup>55</sup> path, defined by the SDE, begins at  $t = 0$  and ends at  $t = T$  where the resulting distribution is  
<sup>56</sup> then a multivariate Gaussian with mean zero and covariance  $\mathbf{I}$ . The code implements various  
<sup>57</sup> SDEs known in the diffusion model literature.

<sup>58</sup> The reverse of the SDE, mapping from multivariate Gaussian samples  $x_T$  to samples of data  
<sup>59</sup>  $x_0$ , is of the form

$$dx_t = [f(x_t, t) - g^2(t)\nabla_{x_t} \log p_t(x_t)]dt + g(t)d\omega_t,$$

<sup>60</sup> where the score function  $\nabla_{x_t} \log p_t(x_t)$  is substituted with a neural network  $s_\theta(x(t), t)$  for  
<sup>61</sup> the sampling process. The network is fit by score-matching ([Hyvärinen, 2005; Vincent, 2011](#))  
<sup>62</sup> across the time span  $[0, T]$ . This network predicts the noise added to the image at time  
<sup>63</sup>  $t$  with the forward diffusion process, in accordance with the SDE, and removes it. With a  
<sup>64</sup> data-dimensional sample of Gaussian noise from the prior  $p_T(x_T)$  (see Figure 1) one can  
<sup>65</sup> reverse the diffusion process to generate data.

<sup>66</sup> The reverse SDE may be solved with Euler-Maruyama sampling ([Song, Sohl-Dickstein, et al., 2021](#))  
<sup>67</sup> (or other annealed Langevin sampling methods) which is featured in the code.

## 69 Likelihood calculations with diffusion models

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

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

76 This ODE can be solved with an initial-value problem to sample new data or estimate its  
 77 density. Starting with a data point  $\mathbf{x}_0 \sim p(\mathbf{x}) = p_0(\mathbf{x}_0)$ , this point is mapped along the  
 78 probability flow ODE path (see the right-hand side of Figure 1) shows a mapping to a sample  
 79 from the multivariate Gaussian prior  $\mathbf{x}_T \sim p_T(\mathbf{x}_T)$ . This inherits the formalism of continuous  
 80 normalising flows (Chen et al., 2019; Grathwohl et al., 2018) without the expensive ODE  
 81 simulations used to train these models - allowing for a likelihood estimate based on diffusion  
 82 models (Song, Durkan, et al., 2021). The initial value problem provides a solution  $\mathbf{x}_T$  and the  
 83 change in probability along the path  $\Delta = \log p_0(\mathbf{x}_0) - \log p_T(\mathbf{x}_T)$  where  $p_T(\mathbf{x}_T)$  is a simple  
 84 multivariate Gaussian distribution. Various ODE solvers of different orders are available (for  
 85 a user to balance speed and accuracy of sampling) which are provided by diffraax (Kidger,  
 86 2022).

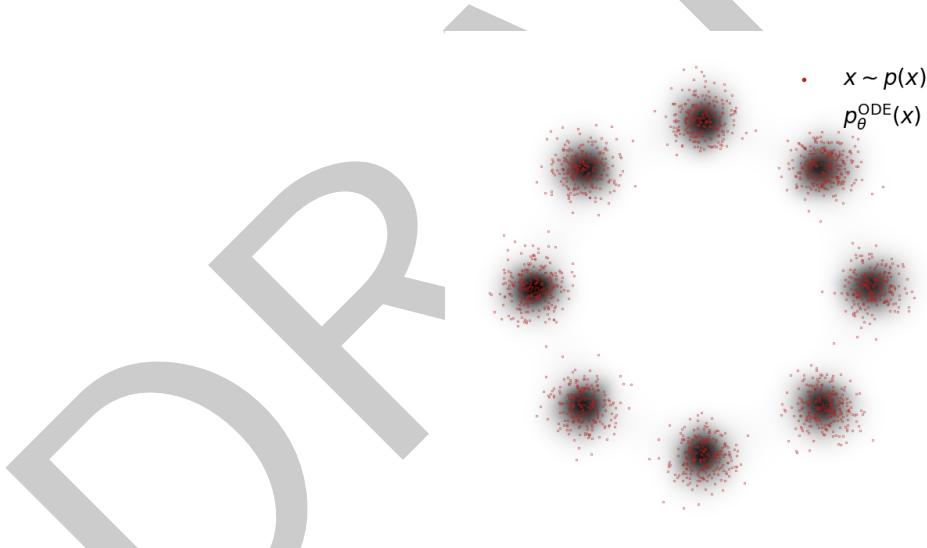


Figure 2: A diagram showing a log-likelihood calculation over the support of a Gaussian mixture model with eight components. Data is drawn (shown in red) from this mixture to train the diffusion model that gives the likelihood (defined by the diffusion model) in gray. The log-likelihood is calculated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

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

$$\frac{\partial}{\partial t} \log p_t(\mathbf{x}_t) = -\nabla_{\mathbf{x}_t} \cdot f(\mathbf{x}_t, t),$$

89 which gives the log-likelihood of a single datapoint  $\mathbf{x}_0$  as

$$\log p(\mathbf{x}_0) = \log p(\mathbf{x}_T) - \int_{t=0}^{t=T} dt \nabla_{\mathbf{x}_t} \cdot f(\mathbf{x}_t, t).$$

90 The code implements these calculations also for the Hutchinson trace estimation method and  
 91 (1990) that reduces the computational expense of the estimate. Figure 2 shows an example of  
 92 a data-likelihood calculation using a trained diffusion model with the ODE associated from an  
 93 SDE.

## 94 **Implementations and future work**

95 Diffusion models are defined in sbgm via a score-network model  $s_\theta$  and an SDE. All the  
 96 available SDEs (variance exploding (VE), variance preserving (VP) and sub-variance preserving  
 97 (SubVP) (Song, Sohl-Dickstein, et al., 2021)) in the literature of score-based diffusion models  
 98 are available. We provide implementations for UNet (Ronneberger et al., 2015), Diffusion  
 99 Transformers (Peebles & Xie, 2023b), MLP-Mixer (Tolstikhin et al., 2021) and Residual  
 100 Network (He et al., 2015) models which are state-of-the-art for diffusion tasks. It is possible to  
 101 fit score-based diffusion models to a conditional distribution  $p(\mathbf{x}|\pi, \mathbf{y})$  where in typical inverse  
 102 problems  $\mathbf{y}$  would be an image and  $\pi$  a set of parameters in a physical model for the data  
 103 (Batzolis et al., 2021) (e.g. to solve inverse problems). The code is compatible with any model  
 104 written in the equinox (Kidger & Garcia, 2021) framework. We recently extended the code  
 105 to provide transformer-based diffusion models (Peebles & Xie, 2023a) and plan to extend to  
 106 latent diffusion models (Rombach et al., 2022) and flow matching (Lipman et al., 2023).

## 107 **Acknowledgements**

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

## 111 **References**

- 112 and, M. F. H. (1990). A stochastic estimator of the trace of the influence matrix for laplacian  
 113 smoothing splines. *Communications in Statistics - Simulation and Computation*, 19(2),  
 114 433–450. <https://doi.org/10.1080/03610919008812866>
- 115 Andrews, A., Jasche, J., Lavaux, G., & Schmidt, F. (2023). Bayesian field-level inference of  
 116 primordial non-gaussianity using next-generation galaxy surveys. *Monthly Notices of the  
 117 Royal Astronomical Society*, 520(4), 5746–5763. <https://doi.org/10.1093/mnras/stad432>
- 118 Batzolis, G., Stanczuk, J., Schönlieb, C.-B., & Etmann, C. (2021). *Conditional image  
 119 generation with score-based diffusion models*. <https://arxiv.org/abs/2111.13606>
- 120 Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G.,  
 121 Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable  
 122 transformations of Python+NumPy programs* (Version 0.3.13). <http://github.com/jax-ml/jax>
- 124 Cabezas, A., Corenflos, A., Lao, J., & Louf, R. (2024). *BlackJAX: Composable Bayesian  
 125 inference in JAX*. <https://arxiv.org/abs/2402.10797>
- 126 Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2019). *Neural ordinary  
 127 differential equations*. <https://arxiv.org/abs/1806.07366>
- 128 Chung, H., Kim, J., Kim, S., & Ye, J. C. (2022). *Parallel diffusion models of operator and  
 129 image for blind inverse problems*. <https://arxiv.org/abs/2211.10656>

- 130 Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference.  
131 *Proceedings of the National Academy of Sciences*, 117(48), 30055–30062. <https://doi.org/10.1073/pnas.1912789117>
- 133 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>
- 135 DeepMind, Babuschkin, I., Baumli, K., Bell, A., Bhupatiraju, S., Bruce, J., Buchlovsky, P.,  
136 Budden, D., Cai, T., Clark, A., Danihelka, I., Dedieu, A., Fantacci, C., Godwin, J., Jones,  
137 C., Hemsley, R., Hennigan, T., Hessel, M., Hou, S., ... Viola, F. (2020). *The DeepMind  
138 JAX Ecosystem.* <http://github.com/google-deepmind>
- 139 Feng, B. T., & Bouman, K. L. (2024). *Variational bayesian imaging with an efficient surrogate  
140 score-based prior.* <https://arxiv.org/abs/2309.01949>
- 141 Feng, B. T., Smith, J., Rubinstein, M., Chang, H., Bouman, K. L., & Freeman, W. T.  
142 (2023). *Score-based diffusion models as principled priors for inverse imaging.* <https://arxiv.org/abs/2304.11751>
- 144 Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville,  
145 A., & Bengio, Y. (2014). *Generative adversarial networks.* <https://arxiv.org/abs/1406.2661>
- 146 Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., & Duvenaud, D. (2018).  
147 *FFJORD: Free-form continuous dynamics for scalable reversible generative models.* <https://arxiv.org/abs/1810.01367>
- 149 He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition.*  
150 <https://arxiv.org/abs/1512.03385>
- 151 Ho, J., Jain, A., & Abbeel, P. (2020). *Denoising diffusion probabilistic models.* <https://arxiv.org/abs/2006.11239>
- 153 Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching.  
154 *Journal of Machine Learning Research*, 6(24), 695–709. <http://jmlr.org/papers/v6/hyvarinen05a.html>
- 156 Kidger, P. (2022). *On neural differential equations.* <https://arxiv.org/abs/2202.02435>
- 157 Kidger, P., & Garcia, C. (2021). Equinox: Neural networks in JAX via callable PyTrees and  
158 filtered transformations. *Differentiable Programming Workshop at Neural Information  
159 Processing Systems 2021.*
- 160 Kingma, Diederik P., Salimans, T., Poole, B., & Ho, J. (2023). *Variational diffusion models.*  
161 <https://arxiv.org/abs/2107.00630>
- 162 Kingma, Diederik P., & Welling, M. (2022). *Auto-encoding variational bayes.* <https://arxiv.org/abs/1312.6114>
- 164 Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., & Le, M. (2023). *Flow matching for  
165 generative modeling.* <https://arxiv.org/abs/2210.02747>
- 166 Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B.  
167 (2021). *Normalizing flows for probabilistic modeling and inference.* <https://arxiv.org/abs/1912.02762>
- 169 Peebles, W., & Xie, S. (2023a). *Scalable diffusion models with transformers.* <https://arxiv.org/abs/2212.09748>
- 171 Peebles, W., & Xie, S. (2023b). *Scalable diffusion models with transformers.* <https://arxiv.org/abs/2212.09748>
- 173 Remy, B., Lanusse, F., Jeffrey, N., Liu, J., Starck, J.-L., Osato, K., & Schrabbach, T. (2023).  
174 Probabilistic mass-mapping with neural score estimation. *Astronomy & Astrophysics*,  
175 672, A51. <https://doi.org/10.1051/0004-6361/202243054>

- 176 Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). *High-resolution*  
177 *image synthesis with latent diffusion models*. <https://arxiv.org/abs/2112.10752>
- 178 Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-net: Convolutional networks for biomedical*  
179 *image segmentation*. <https://arxiv.org/abs/1505.04597>
- 180 Rozet, F. (2024). *Azula: Diffusion models in PyTorch* (Version 0.2.04). <https://github.com/probabilists/azula>
- 182 Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). *Deep unsuper-*  
183 *vised learning using nonequilibrium thermodynamics*. <https://arxiv.org/abs/1503.03585>
- 184 Song, Y., Durkan, C., Murray, I., & Ermon, S. (2021). *Maximum likelihood training of*  
185 *score-based diffusion models*. <https://arxiv.org/abs/2101.09258>
- 186 Song, Y., Shen, L., Xing, L., & Ermon, S. (2022). *Solving inverse problems in medical imaging*  
187 *with score-based generative models*. <https://arxiv.org/abs/2111.08005>
- 188 Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2021).  
189 *Score-based generative modeling through stochastic differential equations*. <https://arxiv.org/abs/2011.13456>
- 191 Spurio Mancini, A., Piras, D., Alsing, J., Joachimi, B., & Hobson, M. P. (2022). <Scp>Cos-  
192 *moPower</scp>*: Emulating cosmological power spectra for accelerated bayesian inference  
193 from next-generation surveys. *Monthly Notices of the Royal Astronomical Society*, 511(2),  
194 1771–1788. <https://doi.org/10.1093/mnras/stac064>
- 195 Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J.,  
196 Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., & Dosovitskiy, A. (2021). *MLP-mixer:*  
197 *An all-MLP architecture for vision*. <https://arxiv.org/abs/2105.01601>
- 198 Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural*  
199 *Computation*, 23(7), 1661–1674. [https://doi.org/10.1162/NECO\\_a\\_00142](https://doi.org/10.1162/NECO_a_00142)