Better Inversion of Diffusion Models for Generative Steganography

Aurélien Noirault
UMR 9189 CRIStAL, CNRS
Lille, France

Tomáš Pevný
Artificial Intelligence Center, Czech Technical University
Prague, Czech Republic

Jan Butora
UMR 9189 CRIStAL, CNRS
Lille, France

Vincent Itier
Centre for Digital System, IMT Nord Europe,
UMR 9189 CRIStAL, CNRS
Lille, France

Patrick Bas
UMR 9189 CRIStAL, CNRS
Lille, France

Abstract

Traditional inversion algorithms attempt to directly invert the diffusion sampling equation. In this work, built on Latent Diffusion Models (LDMs), we propose a family of algorithms with varying time complexities that perform the search of an antecedent within the latent space and/or the Variational Autoencoder (VAE) decoder.

This antecedent search involves optimizing the input to each step of the generation pipeline, such as a single diffusion step. The goal is to minimize the distance between the output of the given step and the variable initially observed by the receiver. We also introduce a novel algorithm, Global Greedy Gradient Descent (3GD), which extends this search to multiple pipeline operations, enhancing both precision and stability.

Our results, using Stable Diffusion 2.1 and the widely-used s-SS [12] scheme, demonstrate that our step-by-step antecedent search achieves error-free performance in 56% of cases at a rate of 1 bit per sample (bps) when using PNG encoding. Furthermore, with Gaussian Shading [23] at 1 bps, our algorithm reduces the average error by 50%. We show a clear trade-off between computational complexity and faithful inversion across our algorithms.

The code for this work is available in the following gitlab repository https://gitlab.cristal.univ-lille.fr/noirault/better-inversion.git

CCS Concepts

• **Security and privacy** → *Information-theoretic techniques.*

Keywords

Data Hiding, Steganography, Generative Steganography, Diffusion Model, Rectified flow Model

ACM Reference Format:
Aurélien Noirault, Tomáš Pevný, Jan Butora, Vincent Itier, and Patrick Bas. 2026. Better Inversion of Diffusion Models for Generative Steganography. In *ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '26), June 17–19, 2026, Firenze, Italy.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3785353.3815088

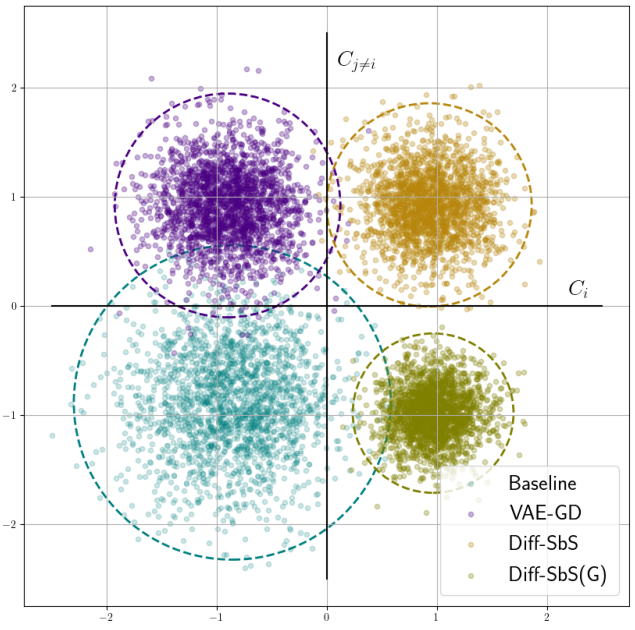

Figure 1: Visualization of the inversion noise of Stable Diffusion 2.1 with a PNG compressed image, and s-SS by projecting the inverted latent image on 2 carriers C_i and $C_{j\neq j}$. The corrected samples show the effect of the correction strategies presented in Section 5.

1 Introduction

Steganography has two main requirements: (1) to embed the largest possible payload and (2) to do so in the least detectable manner, particularly within popular content like digital images. With this in mind, a new category of steganographic algorithms based on generative models and grounded in the principle of generative steganography [6] has recently emerged. A key distinction from

IH&MMSec '26, Firenze, Italy

ACM ISBN 979-8-4007-2376-6/2026/06
https://doi.org/10.1145/3785353.3815088

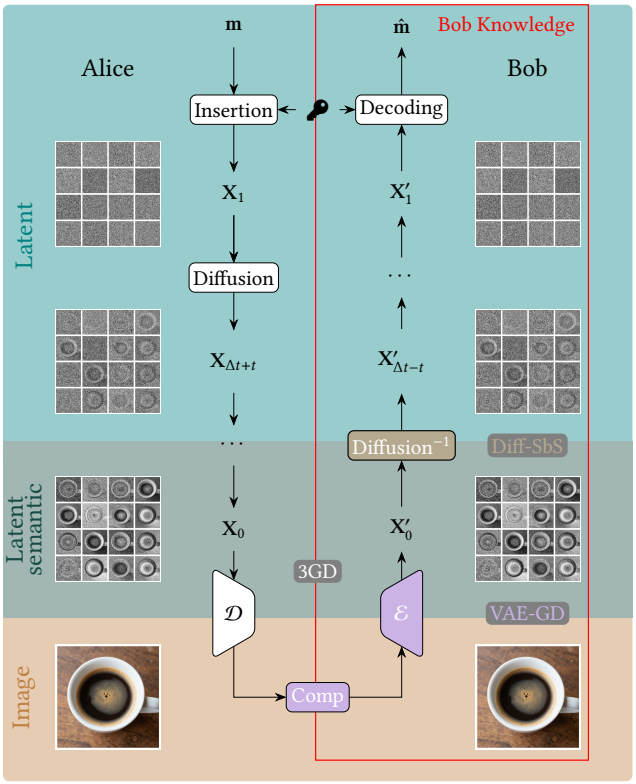

Figure 2: Generation and inversion pipelines, and associated knowledge accessible to the receiver (Bob), as well as our proposed algorithms. The sender (Alice) has access to the whole pipeline. "Comp" stands for image compression, either "lossless" like PNG or lossy like JPEG. 3GD, Diff-SbS, and VAE-GD are the algorithm names used to improve the performance of the operation in the corresponding colors.

classical steganography is that, rather than embedding the payload in the pixel or JPEG space, generative approaches embed it in the latent space. This approach is appealing because the latent space typically follows a prescribed distribution, often modeled as an i.i.d. Gaussian. If this distribution is maintained during embedding, the steganographer gains a compelling argument for undetectability [12]. It is also worth noting that, prior to the advent of generative AI, generative steganography was a niche field, limited to embeddings that mimicked random weak signals such as sensor noise [21] or random movements in generated animated backgrounds [4]. Today, however, it has become an active area of research, with numerous new embedding schemes being proposed [11, 24, 23, 25, 8].

Many models, such as Glow [11] or Latent Diffusion Models (LDM) [19], generate images by starting from a latent space, where the latent vector is sampled from an i.i.d. Gaussian distribution characterized by potentially high entropy. Unlike embedding schemes that operate on image pixels or DCT coefficients and make small ±1 perturbations, generative steganography allows for significant semantic changes in the image while remaining potentially undetectable.

For generative steganography, payload embedding in the latent space is usually performed using a mapping function, which can be seen as a mapping of a label (a codeword from the payload to be embedded) to different parts of the Gaussian distributions. For instance, with Gaussian Shading [23], the message is held in the sign of the sample of the latent variable (in the binary case). For the Spread Spectrum (SS) embedding [9], the payload is mapped to binary antipodal locations on which a secret rotation is applied. Crucially, all these schemes require an inversion process at the receiver: transforming the received image back into its representation within the latent space in order to extract the payload.

Consequently, generative steganography methods embedded in the latent space must face specific constraints:

- First, they need to be robust to lossy coding (*i.e. quantization in the image space (PNG) or DCT domain (JPEG)*) and to the inversion process, *i.e.* the noisy process that transforms the generated image into a vector in the latent space, noise that can be rather important for classical LDM [23]. Note that decoding errors are very problematic since the payload needs, for security reasons, to be encrypted. Error correcting codes can be used, but they usually decrease a lot the capacity of the steganographic channel, especially if the channel error rate is important.
- Secondly, they need to be practically secure, *i.e.* being undetectable, both when performing steganalysis in the latent space [12] or in the image space. The undetectability can, however, be reached if the embedding is *distribution preserving*, *i.e.* if one can prove that the distribution in the latent space is not altered by the embedding.

This paper proposes a set of methods, highlighted in Fig. 2, designed to decrease the noise of the channel and improve the decoding performances by performing an antecedent search of non invertible operation.

From a detectability perspective, any operations that the sender may perform might lead to detectable artifacts. Therefore, the number of operations the sender can perform to improve the robustness of his message is very limited. However, since the image was already transmitted through the canal, the receiver is not limited by such constraints. Following this principle, every algorithm presented in this work are post-hoc algorithm and does not impact the detectability of the embedding scheme use for the communication, only its robustness.

We begin by examining prior works on inversion methods and the security scenario relevant to this work Sec. 2. Subsequently, in Sec. 3, we will delve deeper into the transmission canal and further motivate our choice of performing an antecedents search. Then, in Sec. 4, we present two robust embedding algorithms that will be used for evaluating the algorithms presented in Sec. 5. Finally, the results of our experiments are presented and discussed in Sec. 6.

1.1 Notations

In this work, we use lowercase classical fonts to denote scalar values, lowercase bold fonts for vectors, and uppercase bold fonts

for matrices. Variables that are the lower-case versions of upper-case variables represent unfolded matrices, for example $\mathbf{x}_1$ is the unfolded version of $\mathbf{X}_1$.

We refer to variables existing before the generative process and after the final inversion step as *latent* variables. We name the *latent semantic space* the latent space, which is just before the decoding of the VAE. The *image space* is the RGB space after the decoding of the VAE, details are provided in Fig. 2.

The embedding rates will either be in *bits per pixel* (bpp) or in *bits per sample* (bps), *i.e.* the payload size divided by the number of samples of a latent variable. The latent variable is typically 4×64^2 for SD2.1, 16×64^2 or 16×128^2 for Z-Image, and 16×128^2 for SD3.5.

2 State of the Art and Security Setup

2.1 Inversion Methods

Generating images iteratively from a simple distribution can be modeled as an inverse problem or an optimal transport problem. These approaches have given rise to two distinct generative model families: diffusion models and rectified flow models. The core objective is to transform a basic distribution, typically a standard Gaussian, into a more complex one, such as the distribution of all images. Given the high dimensionality of image data, this presents a significant challenge. To address this complexity, Rombach *et al.* [19] proposed performing diffusion within the latent space of a Variational Autoencoder (VAE).

Diffusion models are trained to reconstruct images from noisy observations, with each step of the process corresponding to a specific noise level. Through iterative denoising, it is possible to "reconstruct" an image starting from purely random noise. Various algorithms, known as solvers, exist to combine the output of each diffusion step and effectively guide a trajectory through the latent space towards the latent semantic space.

The first deterministic one, Denoising Diffusion Implicit Model (DDIM) [20] proposes a way to return to the latent space from the latent semantic by inverting the diffusion equation:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}/\alpha_t}\left(\mathbf{x}_t - \sqrt{1-\alpha_t}\epsilon_\theta^t(\mathbf{x}_t)\right) + \sqrt{1-\alpha_{t-1}}\epsilon_\theta^t(\mathbf{x}_t), \quad (1)$$

with α_t scheduling factors and $\epsilon_\theta^t(\cdot)$ the diffusion model conditioned on a time step t.

The reverted equation, however, relies on future knowledge to have the exact inversion of a step. For instance, we would need the latent variable at step 49 for step 50. To tackle this problem, like nearly all future methods, the author states that a diffusion step is small enough that we can consider $\mathbf{X}_t \approx \mathbf{X}_{t+1}$. The inverted equation then becomes:

$$\mathbf{x}_t = \sqrt{\alpha_{t-1}/\alpha_t}\left(\mathbf{x}_{t-1} - \sqrt{1-\alpha_{t-1}}\epsilon_\theta^t(\mathbf{x}_{t-1})\right) + \sqrt{1-\alpha_t}\epsilon_\theta^t(\mathbf{x}_{t-1}). \quad (2)$$

Later works like DPM[16] or DPM++[17] improved those results by proposing "better" invertible sampling equations, but every one of them is still introducing non-negligible errors during the inversion.

For rectified flow models, instead of predicting a noise to be removed from the latent variable, the diffusion model parameterizes a velocity field to follow in order to map the latent variable to the latent semantic variable. Rectified flow models allow for better trajectory modeling but still rely on the same approximation for the inversion and therefore are bound to sub-optimality.

Despite extensive research into optimal inversion methods for the diffusion process in image editing, all approaches rely on knowledge that is inaccessible to the receiver and therefore unusable in a secure scenario. Moreover, few works have addressed the issue of the VAE. If the model has been trained for minimizing $||\mathcal{D}(\mathcal{E}(\mathbf{X})) - \mathbf{X}||^2$, it is not the case for the reverse $||\mathcal{E}(\mathcal{D}(\mathbf{X})) - \mathbf{X}||^2$.

2.2 Discussion: relations with previous works

Note that rectified flow models (e.g. SD3.5 and Z-Image) and diffusion models (e.g. SD2.1) represent distinct approaches. However, within the scope of this study, we utilize "diffusion" to refer to both rectified flow models and diffusion models when describing the process of inferring a trajectory from a latent space to a latent semantic space, in order to simplify our notation.

In steganography, the possibility of reducing the inversion error of the VAE combined with a gradient descent is mentioned in [10, 22] following the methodology proposed in [7], but it is largely under-exploited since the proposed solution in [10] affects only the VAE, and the scheme presented in [22] is unclear on which components are optimized and how. Additionally, the proposed contribution proposes a greedy global approach that, contrary to [7], prevents the propagation of residual errors on the whole pipeline.

Specifically, we can find several works in line with our approach, either in machine learning or in steganography. Notably [18], tries to find an antecedent with Fixed Point Iteration algorithm but is limited to generators related to small guidance values (< 1). In [10], authors fine-tuned the encoder to mitigate the inversion error, but the optimization is focused on only reducing the error related to the VAE. Another way of solving this issue has been presented in [7], where authors propose to reduce the error of the VAE by minimizing the reconstruction error between the received image $\mathbf{X}_{Alice}$ and the regenerated image $\mathcal{D}(\mathcal{E}(\mathbf{X}_{Alice}))$ by doing a gradient descent. In the same paper, authors proposed to optimize each step of the diffusion process to minimize the reconstruction error of each step using DPM solvers. The original paper is unclear on how the `UPDATE` operation is performed. When discussing the method, the gradient descent seems to be more prevalent, but except for the decoder inversion, there is no gradient in the associated codes. The real impact of each algorithm in terms of complexity and precision is also unknown.

In this paper, we build upon this concept but improve the antecedent search by including the image compression algorithm in the optimization (VAE-GD Sec. 5.2).

We also conduct a more comprehensive study of the impact of each proposed algorithm on inversion errors and decoding error-rates, with or without using the gradient in the optimization processes (Diff-SbS Sec. 5.1, Diff-SbS(G) Sec. 5.3). Moreover, we show that by using a more agnostic approach to the diffusion steps, the method can be extended to rectified flow models, and we give insight into the complexity increase of each algorithm.

Note that previous works [7] also mentioned the possible harmful aspect of optimizing each step of the diffusion individually. To the

best of our knowledge, this is the first proposal to solve this issue by optimizing the reconstruction error in the image domain, and by taking into account the whole generation pipeline using a greedy approach (see 3GD, Sec. 5.4).

2.3 Security Setup

Before going deeper into steganographic algorithms and the minimization of the inversion error, we recall the specific security setup related to generative steganography.

In typical generative steganography scenarios, a sender, Alice, aims to transmit a secret binary message to a receiver, Bob. She achieves this by embedding the message within the latent space of a diffusion model, generating an image from this variable, and transmitting it using a standard format like PNG (RGB, 8 bits) or JPEG. Following the Kerckhoffs' principle, all generation parameters, including the generator weights, prompt, number of diffusion steps, and guidance factor, are publicly known, with security relying on a secret key. Unlike watermarking, where the key is typically constant [2] (the communication channel is rarely sequential), here the secret key can dynamically change between Alice and Bob based on a pre-defined sequence. This approach allows for perfect security, identical distribution of generated stego content, as demonstrated by embedding schemes like s-SS [12], a variant of [8] detailed in Section 4.2.

Even though all parameters are supposed to be public (except the secret key), we assume that the input of lossy operation (image compression), or operations that do not have a perfect inverse (VAE decoder and diffusion steps) are only accessible to Alice. Bob should only rely on the approximate inverse to be able to decode the message. The details of the information available to Bob are given in Fig. 2.

In an operational setting, we consider two equally realistic scenarios. One where the prompt is shared by Alice and Bob, like a secret key that they agreed beforehand. In this case, Bob can do exactly the same set of operations as Alice. In the other case, only Alice knows the real prompt, Bob would only have an approximate prompt via a descriptor such as BLiP [13]. In this work, we will mainly focus on the first scenario where the prompt is supposed to be known by Bob, either because it is public knowledge or because of a side-channel communication without compromising the detectability of the scheme.

Following this security scenario, except if stated otherwise, we will consider that every diffusion step, generation, or inversion will be performed with the same parameters as Alice (guidance, number of steps, prompt, image compression).

3 Transmission canal

This section aims to give more insight into the inversion error, and we will consider a transmission canal composed of the diffusion process, some compression operations such as PNG or JPEG, and the imperfect inversion of the diffusion process.

Looking at the marginal of the variables, a simple first guess would be to approximate the canal as a gaussian canal, *i.e.* $\mathbf{X}'_1 = f(\mathbf{X}_1) = \mathbf{X}_1 + \mathbf{n}$ with $\mathbf{n} \sim \mathcal{N}(0, \sigma_c)$. Such approach has already been successfully used in [12] to perform steganalysis in the latent space of Stable Diffusion 1.5.

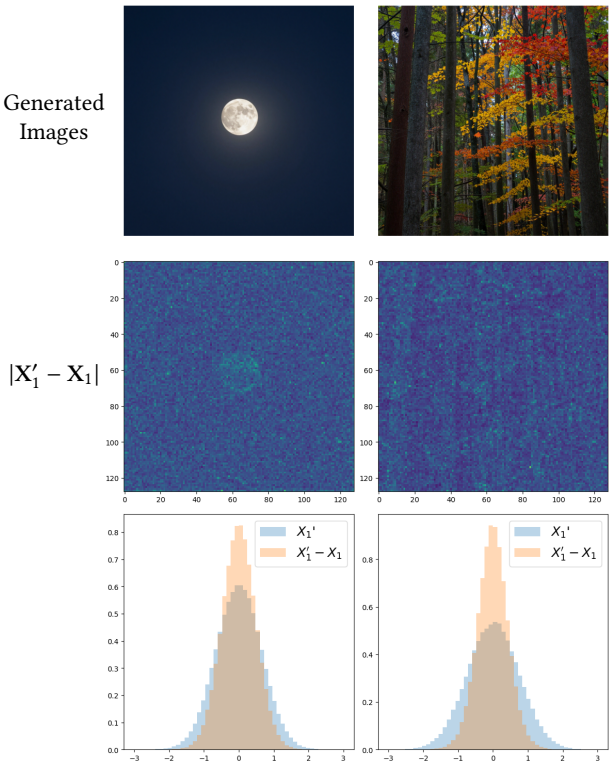

Figure 3: Two images generated with the latent variable $\mathbf{X}_1$ but different prompts with Z-Image and the averaged inversion error $|\mathbf{X}'_1 - \mathbf{X}_1|$ over the 16 latent channels and there histograms.

While sufficient if one is only interested in the norm of the latent variables, this approximation does not take into account the spatial correlation of the error. The inversion aims to retrieve a nosified version of semantics variables. Since the inversion is not perfect, the inversion error is prone to depend on the semantics of the said variable. Moreover, two images generated with the same latent variable but with different prompts will have very different inversion errors, which tends to strengthen this hypothesis. As it is shown in Fig. 3, in some extreme cases, the semantic can still be, at least, partly seen in the inverted latent $\mathbf{X}'_1$.

Furthermore, the inversion error tend to be higher in area corresponding to the smooth part of the image with high luminance. Those kind of area are the most different from random noise, and will require the most amount of modification during the generation and therefore will be the hardest to "remove" during the inversion, increasing the inversion error in the corresponding area of the latent variable. This correlation between inversion error and luminance of the image is further examplified in Fig. 4.

The non gaussianity of the canal and the clear link between the semantic of an image and the inversion error make any usefull modeling of the canal in a data hiding context non-trivial. A more effective way to face the noisy nature of the inversion process is,

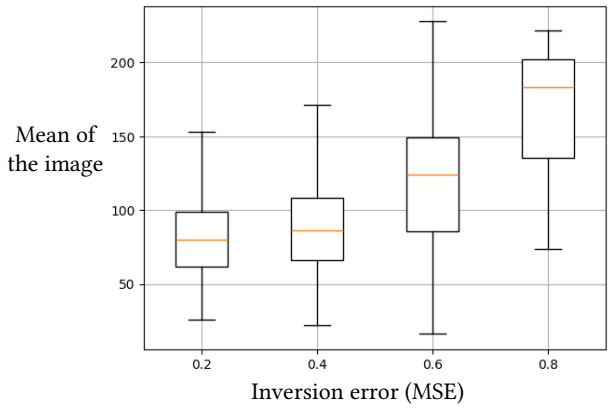

Figure 4: Boxplot of the norm of the average value of images against the inversion error. The higher the average, the more luminance in the image and, the less invertible the image tends to be.

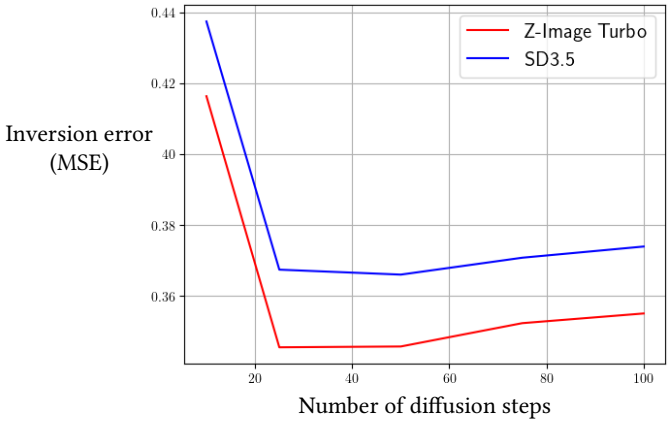

Figure 5: Evolution of the inversion error for a fix amount of generation steps (9 for Z-Image Turbo, 28 for Stable Diffusion 3.5) while changing the number of inversion steps. The results are the average inversion error for 20 images.

as presented in Sec. 5, to perform an antecedant search of the non invertible operations.

4 Data Hiding with Diffusion Models

Diffusion models generate images by sampling from a standard Normal distribution. Consequently, if a given payload can be mapped into this distribution, it is theoretically possible to embed data in an undetectable manner. This property is particularly appealing within security-sensitive contexts, offering potential applications for secret communication and watermarking.

A key limitation of these two frameworks is the ability to accurately recover the secret payload, which is a challenging task given the inversion error. To overcome this issue, most watermarking algorithms use error correcting codes, but this drastically decreases the capacity of such schemes, especially since the error rate fluctuates significantly from one image to another. The nominal capacity of most data hiding algorithms is 1bps, older LDM, like Stable Diffusion 2.1, using a latent space of dimension $(4 \times 64 \times 64)$ to generate RGB images of size $(3 \times 512 \times 512)$, gives an overall capacity of 0.02bpp. More recent models, like Z-Image Turbo or Stable Diffusion 3.5 are using a latent space of dimension $(16 \times 64 \times 64)$ and $(16 \times 128 \times 128)$ respectively, which increases the total capacity up to 0.08bpp. Further reducing capacity through error correction would render these schemes largely uncompetitive compared to standard data hiding algorithms operating directly on the image data.

A way to mitigate this issue is to limit the inversion error after lossless or lossy coding. In the remainder of this section, we present two well known data hiding algorithms which are used to evaluate the performance of our proposed solutions.

4.1 Gaussian Shading (GS)

Gaussian Shading [23] uses the q-ary algorithm to watermark generated images by hiding a payload in the latent variable $\mathbf{X}_1$. To increase the robustness, the payload is duplicated a given amount of time, and pseudo-randomly permuted to prevent any periodicity in the payload before the embedding.

Given a distribution, a standard Normal distribution here, the distribution is split into q bins of equal probabilities, q being the number of symbols of the embedding dictionary. To insert a given symbol, one just needs to sample from a specific portion of the distribution. If every symbol is equiprobable, the sampled vector perfectly follows the original distribution. In the binary case, and similarly to Natural Watermarking [1], the message is consequently held by the sign of the sample.

Instead of performing rejection sampling, the authors proposed to use the cumulative distribution function (cdf) and the percent point function (ppf) of the standard Normal distribution as well as random shifting to map a binary message in the latent variable. The i^{th} sample of the latent variable $\mathbf{x}_1$ becomes:

$$x_{1,i} = \text{ppf}\left(\frac{u + m_i}{2^q}\right), \tag{3}$$

where m_i is the next symbol of the message to be embedded and $u \sim \mathcal{U}(0, 1)$. The reverse operation is:

$$m_i = \left\lfloor 2^q \text{cdf}\left(x_{1_i}\right) \right\rfloor, \tag{4}$$

with $\lfloor \cdot \rfloor$ the floor rounding operation.

4.2 Scaled Spread Spectrum (s-SS)

The scheme proposed by [8] offers a better robustness (*i.e.* a lower bit error rate), and a relative security against steganalysis when it is performed in the image domain. Conceptually, we can see the embedding as binary spread-spectrum watermarking [3] applied in the latent space. Note that this embedding strategy is popular among watermarking algorithms.

We consider a payload matrix with the same number of elements as the latent variable $\mathbf{M}_c \in \{-1, +1\}^{n \times n}$ and its individual elements $m_{c,i,j}$. Given c random orthonormal matrices $\mathbf{Q}_c$ derived from a secret key, the orthonormal carrier associated with each bit $m_{c,i,j}$ is

consequently given by $\mathbf{C}_{i,j} = \mathbf{q}_i \cdot \mathbf{q}_j^T$, where $\mathbf{q}_k$ is the k^{th} column of the matrix $\mathbf{Q}_c$. The payload matrix $\mathbf{M}$ is transformed to a latent matrix $\mathbf{X}_1$ by the channel-wise operation:

$$\mathbf{X}_1 = \mathbf{Q} \cdot \mathbf{M} \cdot \mathbf{Q}^T. \tag{5}$$

After potential compression of the image, the noisy inversion process is applied, using the same parameters as for the generation (guidance scale, prompt, number of diffusion steps) to obtain the latent image $\mathbf{X}'_1$, the decoding is computed using the inverse transform:

$$\hat{\mathbf{M}} = \mathbf{Q}^T \cdot \mathbf{X}'_1 \cdot \mathbf{Q}. \tag{6}$$

By invocation of the Central Limit Theorem, the authors claim that each sample $x_{1,i,j}$ is asymptotically i.i.d. as the latent samples of cover images, *i.e.* $\mathcal{N}(0,1)$. Note that this assumption is wrong, as demonstrated in [12]. The embedded latent variable is distributed on a hypersphere where the latent variable is supposed to be Gaussian.

In other words, the norm of the cover variable are supposed to follow a χ_n distribution and the norm of the stego is a $\delta_{\sqrt{n}}$ a dirac distribution, with n the number of dimensions of the latent variable. Using the invertibility of diffusion models, authors construct a test based on the norm of the inverted latent variable.

To address this limitation, the authors also propose to scale the embedded latent variable by a norm s, sampled from a χ_n distribution. The "embedding" function becomes:

$$\mathbf{X}_1 = \frac{s}{\sqrt{n}} \mathbf{Q} \cdot \mathbf{M} \cdot \mathbf{Q}^T. \tag{7}$$

5 Improved Inversion Algorithms

We propose here a set of different algorithms which are designed to decrease both the inversion error and the decoding BER in steganography.

5.1 Gradient Free Algorithm

One easy way to reduce the inversion error is to increase the number of diffusion steps, even though increasing the number of generation steps is dangerous and could lead to detectable artifacts since the semantic content is intimately linked to the number of steps.

However one could use any number of inverse diffusion steps without compromising the security of the scheme. Fig. 5 demonstrates that increasing inverse diffusion steps while keeping a fixed number of generation steps reliably lowers the inversion error. Diffusion model estimates a trajectory between two distributions, the latent distribution (standard Normal distribution) and the latent semantic distribution. To be implemented, the trajectory needs to be discretized, which corresponds to the diffusion steps. Therefore, more diffusion steps correspond to a less discrete trajectory, a more faithful one.

The most widely used solver for diffusion models, DDIM [20] uses the approximation that $\epsilon_\theta^t(\mathbf{x}_t) \approx \epsilon_\theta^t(\mathbf{x}_{t-1})$ to reverse the sampling equation and approximate the reverse sampling trajectory. Using this same approximation, we instead propose to search for an antecedent of each diffusion step in order to decrease the inversion error.

Algorithm 1: Antecedent search of each Diffusion step, Diff-SbS and Diff-SbS(G)

Input: α_d (learning rate), T number of diffusion step, $\mathbf{X}_0$ (latent semantic variable), *Grad* (boolean, method to use)

$\mathbf{X}_t \leftarrow \mathbf{X}_0$
for $t \leftarrow 1$ **to** T **do**
  $\mathbf{X}_{target} \leftarrow \mathbf{X}_t$
  $\mathbf{X}_t \leftarrow \text{Diff}_t^{-1}(X_t)$
  while *not converged* **do**
    if *not Grad* **then**
      $\mathbf{X}_t \leftarrow \mathbf{X}_t - \alpha_d \left(\text{Diff}_t(\mathbf{X}_t) - \mathbf{X}_{target}\right)$
    end
    else
      $\mathbf{X}_t \leftarrow \mathbf{X}_t - \alpha_d \nabla_{\mathbf{X}_t} \mathcal{L}\left(\text{Diff}_t(\mathbf{X}_t), \mathbf{X}_{target}\right)$
    end
  end
end
$\mathbf{X}'_0 \leftarrow \mathbf{X}_t$
return $\mathbf{X}'_0$

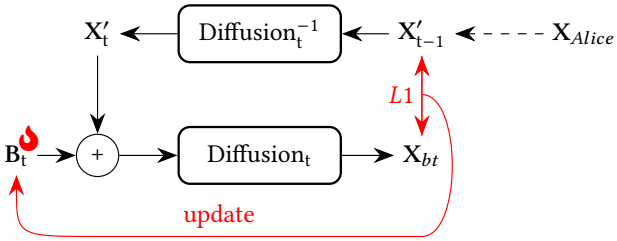

Figure 6: Process of the antecedent search for one step of diffusion. The starting point is initialized using an approximation of one diffusion step, then we learn a bias to correct the inversion error.

The naïve, gradient free version works as follows. If we consider that $\epsilon_\theta^t(\mathbf{x}_t) \approx \epsilon_\theta^t(\mathbf{x}_{t-1})$, by taking a black box approach independent of the solver as exemplified in Fig. 6, we can expand the approximation as $\mathbf{x}_t \approx \mathbf{x}_{t-1}$. We can learn a bias $\mathbf{b}_t$ by taking steps in the opposite direction of the reconstruction error, $[\text{Diff}_t(\mathbf{x}_t + \mathbf{b}_t) - \mathbf{x}_{t-1}]$, where $\text{Diff}_t(\cdot)$ is the diffusion step at time t. To reduce the computation time, $\mathbf{x}_t$ can be initialized by the approximation of the inverse diffusion step $\mathbf{x}_t \simeq \text{Diff}_t^{-1}(\mathbf{x}_{t-1})$. This is presented Alg. 1 (Diff-SbS) and produce only a marginal increase to the computational cost, as the number of diffusion steps required only become TN when the initial inversion require T steps and N is the number of optimization steps. However, this algorithm still heavily relies on the DDIM approximation and, therefore, is bound to yield suboptimal results. Moreover, this algorithm can only be applied to the diffusion part, leaving the error of the VAE unaddressed.

Even though it is possible to fix $\alpha_d = 1$, and use only one step of optimization, it is usually not enough. Since it is based on approximation, it will lead to suboptimal results. Empirically, we observed the best results on average with $\alpha_d = 0.1$.

Algorithm 2: Antecedent search of the VAE (VAE-GD)

Input: α_{vae} (learning rate), $\mathbf{X}_{Alice}$ (transmitted image)

$\mathbf{X}'_0 \leftarrow \mathcal{E}(\mathbf{X}_{Alice})$

while *not converged* **do**

  $\mathbf{X}_{CPR} \leftarrow \mathrm{Comp}(\mathcal{D}(\mathbf{X}'_0))$

  $\mathbf{X}'_0 \leftarrow \mathbf{X}'_1 - \alpha_{vae} \nabla_{\mathbf{X}'_0} \mathcal{L}(\mathbf{X}_{CPR}, \mathbf{X}_{Alice})$

end

return $\mathbf{X}'_1$

Table 1: Evaluation of the performance of the antecedent search on flow models, such as Stable Diffusion 3.5 and Z-ImageTurbo.

		MSE		BER (%)	
		SD3.5	Z-Image	SD3.5	Z-Image
PNG	Baseline	0.38	0.36	9.08	8.95
	VAE-GD	0.25	0.35	4.94	8.06
	Diff-SbS	0.23	0.32	4.34	7.12
JPEG (QF95)	Baseline	0.70	0.73	19.4	23.4
	VAE-GD	0.64	0.76	16.86	23.06
	Diff-SbS	0.66	0.75	16.54	22.38

5.2 Decoder optimization through gradient descent

Variational autoencoders are trained to minimize the reconstruction error between an input image and the output of its decoder. However, in a data hiding scenario, the important part is the distance between the input of the said decoder and the encoder output, the distance being unconstrained during training. To overcome this limitation, we employ an antecedent search of the decoder, a very efficient way to do so is with a gradient descent. If this idea has already been used in [7], note, however, that it is suboptimal. Both the image space and the latent semantic space are high dimensional continuous space, depending on the structure of the latent semantic space, and finding a proper antecedent might be difficult. One can improve the search by reducing the number of possible antecedents by taking into account more operations, such as compression algorithms, using the straight through[1] techniques for non differentiable function, such as rounding or clipping. This algorithm, which will refer to VAE-GD for the remainder of this work, is described in Alg. 2.

5.3 Step by step gradient descent for each inverse diffusion process

Within a Latent Diffusion Model (LDM) framework, suboptimal antecedent search for even one operation can significantly degrade the inversion process due to error propagation throughout the pipeline. Relying on approximations can also introduce undesirable artifacts. Therefore, improving the inversion quality of any single operation directly enhances the stability and accuracy of the entire system.

[1]The regular function is used for the forward part, but its gradient is replaced with an identity.

Table 2: Improvement of VAE-GD by using differentiable coding compared to using the floating point output of the VAE decoder.

BER(%)	PNG	JPEG QF95	JPEG QF75
∅	0.67	2.59	12.83
Diff Comp	0.68	2.12	11.88

Using the approach of Sec. 5.2, we can remove the need for the approximation for the algorithm of Sec. 5.1 by finding the antecedent of a given step with gradient descent. As shown by Alg. 2 and Alg. 1, this version of the algorithm is a specialized version of the VAE-GD algorithm for a diffusion step.

This Diff-SbS(G) algorithm significantly increases the stability of the entire inversion process while only minimally increasing computational time complexity. The number of diffusion steps remains consistent with the standard Diff-SbS algorithm, however, the requirement to compute a gradient on the diffusion model does introduce an additional computational cost.

5.4 Global Greedy Gradient Descent (3GD)

It is conceptually possible to extend this approach to the entire pipeline by computing the distance in the image space and the gradient with respect to the original latent space. The results found in each step are likely to contain errors, those errors could be very small, but they can be propagated to the following steps and create a drift. Taking the entire pipeline into account solves this issue. However, in practice, such optimization would not easily converge towards the optimal solution.

Instead, we propose a hybrid approach where, after each optimization of an operation, the next operation is incorporated into the global optimization process as it is shown in Alg. 3. Computing the loss in the image space allows some built-in regularization where the error propagated by the optimization of step t could be corrected at time $t - \Delta t$, increasing the stability of the optimization.

For instance, at time 0, the optimization is equivalent to VAE-GD, the target function, or the function to find the antecedant from is $\mathrm{Diff}_T \circ \mathcal{D}(\cdot)$, at the next step $\mathrm{Diff}_{T-1} \circ \mathrm{Diff}_T \circ \mathcal{D}(\cdot)$ up to $\mathrm{Diff}_0 \circ \cdots \circ \mathrm{Diff}_T \circ \mathcal{D}(\cdot)$, the algorithm is also exemplified in Fig. 7.

Note that even if the potential for error correction is the highest of our proposed algorithm, the computational complexity is also the most important. If we compare to the number of diffusion steps required for Diff-SbS(G), which is TN, the number of steps for the 3GD algorithm is T^2N. Moreover, computing the gradient of the complete diffusion process is very expensive, one as to use gradient check-pointing between diffusion steps. Checkpoints allow reducing the requirements of VRAM but increase the number of call to diffusion steps depending on the implementation.

6 Experiments of the inversion error

First, note that since our methods are working after the reception of the image, there is no modification applied to the image, which means that we do not alter the detectability of the embedding

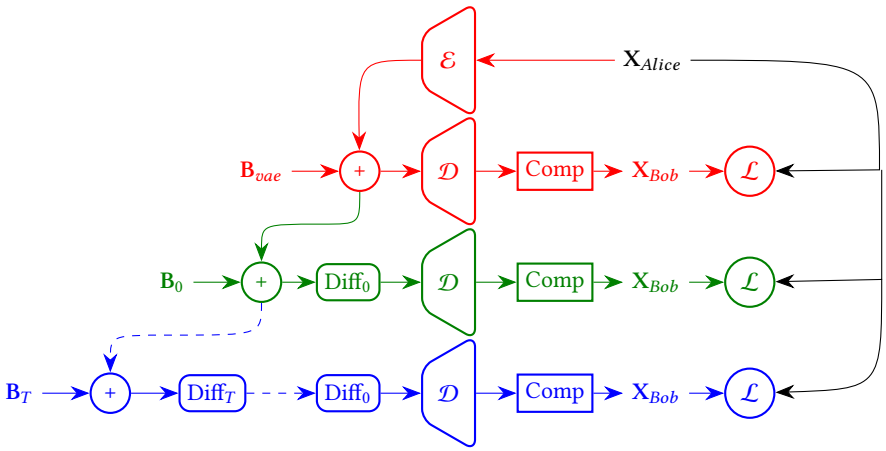

Figure 7: Process of the 3GD algorithm. At first, it is analogous to VAE-GD (red), then the diffusion part is added to the optimization pipeline (green) step by step until the complete generation is optimized (blue).

Table 3: Approximated computation time of the different algorithms for the S-SS, PNG experiment on a NVIDIA RTX A6000.

Time (s)	Z-Image	SD3.5	SD2.1
Baseline	11	20	11
VAE-GD	18	51	36
Diff-SbS	101	379	272
Diff-SbS(G)	-	-	2363
3GD	-	-	32346

Algorithm 3: Antecedent search of the generation pipeline with Global Greedy Gradient Descent.

Input: α_{diff} (learning rate), $\mathbf{X}_{Alice}$ (transmitted image)

$\mathbf{X}_t \leftarrow \text{VAE-GD}(\mathbf{X}_{Alice})$
for $t \leftarrow 1$ **to** T **do**
  $\mathbf{X}_t \leftarrow \text{Diff}_t^{-1}(X_t)$
  while *not converged* **do**
    $\mathbf{X}_{t'} \leftarrow \mathbf{X}_t$; // point to optimize
    /* Simulate the pipeline */
    for $t' \leftarrow t$ **to** T **do**
      $\mathbf{X}_{t'} \leftarrow \text{Diff}_{t'}(\mathbf{X}_{t'})$
    end
    $\mathbf{X}_{Bob} \leftarrow \text{Comp}(\mathcal{D}(\mathbf{X}_{t'}))$
    /* update */
    $\mathbf{X}_t \leftarrow \mathbf{X}_t - \alpha_d \nabla_{\mathbf{X}_t} \mathcal{L}(\mathbf{X}_{Bob}, \mathbf{X}_{Alice})$
  end
end
$\mathbf{X}'_0 \leftarrow \mathbf{X}_t$
return $\mathbf{X}'_0$

scheme algorithm used, and we do not need to conduct any security analysis.

Table 4: Evaluation of the performance of the antecedent search with Stable Diffusion 2.1, MSE is the means squared inversion error, BER is the average pourcentages of wrongly decoded bits, and SR, the success rate, the proportion of images reaching BER= 0.

PNG/JPEG	MSE		BER(%)		SR(%)	
Baseline	0.22	0.29	3.15	5.39	0	0
VAE-GD	0.11	0.17	0.68	2.12	15	6
Diff-SbS	0.09	0.15	0.58	1.91	36	13
Diff-SbS(G)	0.08	0.15	0.53	1.78	45	17
3GD	0.06	-	0.39	-	56	-

6.1 Parameters

In this section, we evaluate the gain in performance by using our algorithms compared to simply increasing the number of steps of the inverse diffusion process with three different image generators, Stable Diffusion 2.1, Stable Diffusion 3.5 [5], and Z-ImageTurbo [15] via the HuggingFace implementation and pre-trained weights. The generation parameters are kept to default, for 9 generation steps for Z-ImageTurbo, 28 steps, and a guidance of 4.5 for Stable Diffusion 3.5, 50 steps, and a guidance of 7.5 for Stable Diffusion 2.1. The image size is set to 512×512 for Z-ImageTurbo and Stable Diffusion 2.1 and to 1024×1024 for Stable Diffusion 3.5.

For all experiments, we used an $L1$ loss function. The VAE-GD algorithm was consistently optimized with Adam and a learning rate of $\alpha_{vae} = 0.01$. The Diff-SbS algorithm employed a learning rate of $\alpha_d = 0.1$, while the Diff-SbS(G) and the 3GD algorithms used the LBFGS optimizer [14].

For a fair evaluation, when evaluating the performance of the Baseline (no optimization) or only the VAE-GD algorithm, the number of inverse diffusion steps is set to 50 since more steps mean a lower inversion error, as shown in Section 5.1. All the results are the average of 100 events. Prior to optimizing the diffusion inversion

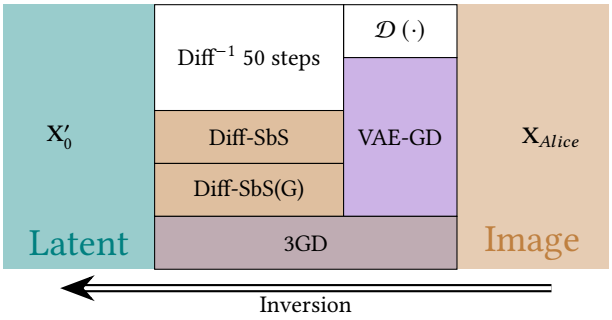

Figure 8: Test frameworks where Diff^{-1} 50 steps is the inverse diffusion of 50 steps and $\mathcal{D}(\cdot)$ the decoder of the VAE.

(Diff-SbS, Diff-SbS(G)), the VAE-GD is used to obtain the latent semantic vector.

The details of the evaluation pipeline and how each algorithm is used with one another are given in Fig. 8.

6.2 Inversion Error

When searching for an antecedent of a function, leveraging prior knowledge improves efficiency by reducing the search space. In the case of inverting the generation of images, one form of prior knowledge is the operation used to generate the said image, including the compression process of the image especially for non linear compression algorithm such as JPEG. In Tab. 2, we show that using a differentiable JPEG/PNG compression scheme in the optimization process enables us to decrease the inversion error.

A comparative study of the methods can be found in Tab. 4 using the s-SS insertion at 1bps capacity with Stable Diffusion 2.1. The results are also illustrated in Fig. 1. Tab. 1 shows that the antecedent search is also effective with rectified flow models.

Already, decreasing the inversion error of the VAE is very efficient. In the PNG case, the bit error rate of Z-Image decreases by almost one point, where it is halved on SD3.5 and is under 1% for SD2.1. and achieve descent performance on JPEG images.

The antecedent search is more effective than inverting the sampling equation. The naive antecedent search (Diff-SbS) is more effective than VAE-GD for Z-Image while providing still good improvement with SD2.1 and SD3.5 for both PNG and JPEG cases in terms of BER, while achieving perfect decoding twice more often with SD2.1.

The gradient based methods are, as expected, the most efficient ones. The method Diff-SbS(G) brings the success rate (the proportion of images achieving perfect decoding) close to 50% for PNG images and near 20% on JPEG images.

The greedy algorithm (3GD) achieves near perfect decoding $BER = 0.39\%$ on average, with the success rate above 50% (SR=56%). With a high success rate, it is therefore possible to generate several images with different prompts but the same message to find one that will be perfectly decoded. To minimize computational cost, we opted not to test 3GD on JPEG images. Nevertheless, based on its successful performance with PNG images and our previous results on JPEG images, it would also perform effectively.

Furthermore, we show that our algorithms are also effective with Gaussian Shading for different capacities with the Diff-SbS algorithm. As we can see in Tab. 5, even the fast algorithm can reach perfect decoding.

6.3 Time complexity

Performing an antecedent search to invert the generation increases the computation time of the inversion. If we consider the cost of making a call to the VAE decoder as low compared to calling one step of diffusion, the cost of optimizing the inversion error of the VAE will be minimal, as shown in Tab. 3. Looking for the antecedent of the diffusion is, however, way more costly, increasing the number of diffusion steps from T for the regular inversion to NT for Diff-SbS and Diff-SbS(G) and to NT^2 for 3GD. The time computation might sound prohibitive, but the increase in performance is far from being negligible, especially with 3GD. Please keep in mind that the given computation times are only indicative for computation done on NVIDIA RTX A6000, using a better GPU will, of course, make the computation faster, for instance, using an NVIDIA L40S decreases the computation time of 3GD from $\sim$ 9h to $\sim$ 4h. The use of gradient checkpointing also slows down the optimization process.

7 Conclusion

We have demonstrated, in this paper, that inverting the sampling equation is not the only approach to inverting a diffusion process. Indeed, for data hiding, it can significantly limit the capacity of embedding schemes. We evaluated four algorithms designed to improve this generation process.

Our experiments highlight the benefits of leveraging increased knowledge for antecedent search, whether through utilizing image coding algorithms (VAE-GD) or optimizing more stages of the generation process. This resulted in a success rate of up to 56% in the PNG case. Furthermore, our methods are agnostic to embedding schemes, achieving strong performance with both s-SS and Gaussian Shading. We also showed that a black box approach to the optimization process enables more faithful inversion, applicable not only to diffusion models but also to rectified flow models.

Our findings highlight a clear trade-off between precision and computational cost. Depending on the application and model used, VAE-GD already delivers very good performance. For more precise decoding, the Diff-SbS(G) algorithm or its gradient-free variant can be employed for larger models. In scenarios where precision is paramount, the 3GD algorithm's high success rate allows for efficient resampling of stego images to guarantee perfect message decoding. However, it's important to note that this approach could potentially introduce bias in the distribution of stego images, a factor requiring further investigation. Even though one might think that the computational cost of 3GD is prohibitive and ill-suited for real case applications, please remember that the cost of embedding the message does not change. One could imagine a case for highly susceptible information where the sender would be an individual without acces to expensive computation resources and the receiver a collective that could afford the computation cost.

However, these algorithms are primarily suited for PNG or low-compression factor JPEG images, as one would expect to optimize

Table 5: Evaluation of the performance of the Diff-SbS algorithm using Stable Diffusion 2.1, Gaussian Shading for different level of redundancy.

	BINARY					QUATERNARY				
$BER(\%)/SR(\%)$	1bps ×1	0.25bps ×4	0.125bps ×8	0.0625bps ×16	0.03125bps ×32	2bps ×1	1bps ×2	0.5bps ×4	0.25bps ×8	0.125bps ×16
None	13.75/0	5.63/0	1.17/0.5	0.08/64.5	0.006/98.5	33/0	30/0	18/0	8.9/0	5/1
Diff-SbS	7.34/0	1.95/0	0.19/33	0.002/97.5	0/100	18.9/0	16.5/0	7/0	1.54/1	0.93/61

a different diffusion trajectory for a significantly degraded image compared to the generation process.

Acknowledgments

This work received funding from the French Defense & Innovation Agency. This work was also supported by a French government grant managed by the Agence National de la Recherche under the France 2030 program, reference ANR-22-PECY-0011, and by the project referenced by ANR-23-IAS4-0004. This work was also supported by 25-17259K, Fundamental Tradeoffs for Information Hiding in Generated Media (DETERMINE). This work was granted access to the HPC resources of Lille University.

References

[1] Francois Cayre and Patrick Bas. "Kerckhoffs-based embedding security classes for woa data hiding". In: *IEEE Transactions on Information Forensics and Security* 3.1 (2008), pp. 1–15.

[2] François Cayre, Caroline Fontaine, and Teddy Furon. "Watermarking security: theory and practice". In: *IEEE Transactions on signal processing* 53.10 (2005), pp. 3976–3987.

[3] Ingemar J Cox et al. "Digital watermarking". In: *Morgan Kaufmann Publishers* 54.56-59 (2008), p. 2.

[4] Scott Craver et al. "A supraliminal channel in a videoconferencing application". In: *International Workshop on Information Hiding*. Springer. 2008, pp. 283–293.

[5] Patrick Esser et al. "Scaling rectified flow transformers for high-resolution image synthesis". In: *Forty-first international conference on machine learning*. 2024.

[6] Jessica Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge university press, 2009.

[7] Seongmin Hong et al. "On exact inversion of dpm-solvers". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 7069–7078.

[8] Xiaoxiao Hu et al. "Establishing robust generative image steganography via popular stable diffusion". In: *IEEE Transactions on Information Forensics and Security* 19 (2024), pp. 8094–8108.

[9] F. Huang et al. "Distortion Function Designing for JPEG Steganography with Uncompressed Side-Image". In: *1st ACM IH&MMSec. Workshop*. Ed. by W. Puech et al. Montpellier, France, June 2013.

[10] Jingyuan Jiang et al. "Generative image steganography based on text-to-image multimodal generative model". In: *IEEE Transactions on Circuits and Systems for Video Technology* (2025).

[11] Durk P Kingma and Prafulla Dhariwal. "Glow: Generative flow with invertible 1x1 convolutions". In: *Advances in neural information processing systems* 31 (2018).

[12] Etienne Levecque et al. "Targeted Pooled latent-space steganalysis applied to generative steganography, with a fix". In: *2026 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Barcelona, Spain, May 2026.

[13] Junnan Li et al. "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation". In: *International conference on machine learning*. PMLR. 2022, pp. 12888–12900.

[14] Dong C Liu and Jorge Nocedal. "On the limited memory BFGS method for large scale optimization". In: *Mathematical programming* 45.1 (1989), pp. 503–528.

[15] Dongyang Liu et al. "Decoupled DMD: CFG Augmentation as the Spear, Distribution Matching as the Shield". In: *arXiv preprint arXiv:2511.22677* (2025).

[16] Cheng Lu et al. "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps". In: *Advances in neural information processing systems* 35 (2022), pp. 5775–5787.

[17] Cheng Lu et al. "Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models". In: *Machine Intelligence Research* 22.4 (2025), pp. 730–751.

[18] Zhihong Pan et al. "Effective real image editing with accelerated iterative diffusion inversion". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 15912–15921.

[19] Robin Rombach et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[20] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models". In: *arXiv preprint arXiv:2010.02502* (2020).

[21] Théo Taburet et al. "Natural steganography in JPEG domain with a linear development pipeline". In: *IEEE Transactions on Information Forensics and Security* 16 (2020), pp. 173–186.

[22] Haozhong Yang et al. "SDS-TG: Secure Diffusion Steganography in Text-Guided Generative Images". In: *2025 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2025, pp. 1–6.

[23] Zijin Yang et al. "Gaussian shading: Provable performance-lossless image watermarking for diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 12162–12171.

[24] Zhili Zhou et al. "Secret-to-Image Reversible Transformation for Generative Steganography". In: *IEEE Transactions on Dependable and Secure Computing* 20.5 (2023), pp. 4118–4134. DOI: 10.1109/TDSC.2022.3217661.

[25] Jiahao Zhu et al. "Plug-and-Hide: Provable and Adjustable Diffusion Generative Steganography". In: *arXiv preprint arXiv:2409.04878* (2024).