A Regularized Conditional GAN for Posterior Sampling in Inverse Problems

Matthew C. Bendel ¹ Rizwan Ahmad ² Philip Schniter ¹

Abstract

In inverse problems, one seeks to reconstruct an image from incomplete and/or degraded measurements. Such problems arise in magnetic resonance imaging (MRI), computed tomography, deblurring, super-resolution, inpainting, and other applications. Given a training set of signal/measurement pairs, we design a method to generate posterior samples rapidly and accurately. Our interest in posterior sampling is inspired by linearization comprises an $\ell_1$ penalty and an adaptively weighted standard-deviation reward. Using quantitative evaluation metrics like conditional Fréchet inception distance, we demonstrate that our method produces state-of-the-art posterior samples in both multicoil MRI and large-scale inpainting applications.

1. Introduction

Given a training dataset of signal/measurement pairs $\{(x_t, y_t)\}_{t=1}^T$, which defines a joint distribution $p_{x,y}$, our goal is to learn a generating function $G_\theta$ such that, for a given $y$, maps random code vectors $z \sim \mathcal{N}(0, I)$ to posterior samples $\tilde{x} = G_\theta(z, y)$ which defines a joint distribution $p_{x|y}$. Our interest in posterior sampling is inspired by linear inverse problems, where the goal is to recover a signal or image $x$ from a measurement $y$ of the form

$$ y = Ax + w, $$

where $A$ is a known matrix and $w$ is unknown noise. Such problems arise in deblurring, super-resolution, inpainting, computed tomography (CT), magnetic resonance (MR) imaging, and other fields. When $A$ does not have full column rank, $Ax$ filters out any components of $x$ that lie in the nullspace of $A$, and so prior information about the true $x$ is needed for accurate recovery (e.g., we may know that $x$ is an MR image). But even after taking such prior information into account, there may be many hypotheses of $x$ that yield equally good explanations of $y$. Thus, rather than settling for a single “best” estimate of $x$ from $y$, our goal is to efficiently sample from the posterior $p_{x|y}(\cdot|y)$.

There exist several approaches that learn to sample from the posterior given training samples $\{(x_t, y_t)\}_{t=1}^T$, including conditional generative adversarial networks (cGANs) (Isola et al., 2017; Adler & Öktem, 2018; Zhao et al., 2021a), conditional variational autoencoders (cVAEs) (Edupuganti et al., 2021; Tonolini et al., 2020), conditional normalizing flows (cNFs) (Ardizzone et al., 2019; Winkler et al., 2019), and score-based generative models (Jalal et al., 2021a; Song et al., 2021; 2022a). We focus on cGANs, which are typically regarded as generating samples of high quality but low diversity.

We propose a cGAN that addresses the aforementioned lack-of-diversity issue by training with a novel regularization that aims to enforce consistency with the true posterior mean and covariance. Our proposed regularization consists of supervised-$\ell_1$ loss plus an adaptively weighted standard-deviation (SD) reward.

We demonstrate our approach on accelerated MR image recovery and large-scale image completion/inpainting. To quantify performance, we focus on conditional Fréchet inception distance (CFID) (Soloveitchik et al., 2021), but we also consider FID (Heusel et al., 2017), PSNR, SSIM (Wang et al., 2004), and average pixel-wise SD (APSD). Our results show the proposed regularized cGAN (rcGAN) outperforming existing cGANs (Adler & Öktem, 2018; Ohayon et al., 2021a; Zhao et al., 2021a) and the score-based generative models from (Jalal et al., 2021a) and (Song et al., 2021) in all tested metrics.

2. Problem formulation and background

2.1. Conditional Wasserstein GAN

We build on the Wasserstein cGAN framework from (Adler & Öktem, 2018). The goal is to design a generator network $G_\theta : \mathcal{Z} \times \mathcal{Y} \rightarrow \mathcal{X}$ such that, for fixed $y$, the random variable $\tilde{x} = G_\theta(z, y)$ induced by $z \sim p_z$ has a distribution that best matches the posterior $p_{x|y}(\cdot|y)$ in Wasserstein-1 distance.
Here, $z$ is drawn independently of $y$, and $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$ denote the sets of $x$, $y$, and $z$, respectively.

The Wasserstein-1 distance can be expressed as

$$W_1(p_{x|y} \cdot y), p_{z|y} \cdot y) = \sup_{D \in \mathcal{L}_1} E_{x|y} [D(x,y)] - E_{z|y} [D(\tilde{x}, y)],$$

(2)

where $L_1$ denotes functions that are 1-Lipschitz with respect to their first argument and $D : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a "critic" or "discriminator" that tries to distinguish between true and generated data.

Although issues with stability and mode collapse are also present in unconditional GANs (uGANs), the causes of mode-collapse in uGANs or discretized cGANs (Mirza & Osindero, 2014) are fundamentally different than in continuously conditioned cGANs. In the latter case, there is only one example of a valid $x_i$ for each given $y_i$, whereas in the former case there are either many $x_i$ or many $x_i$ for each conditioning class. As a result, most strategies used to combat mode-collapse in uGANs, like (Schonfeld et al., 2020; Karras et al., 2020; Zhao et al., 2021b), are not well suited to cGANs. For example, mini-batch discrimination strategies like MBSD (Karras et al., 2018), where the discriminator aims to distinguish a mini-batch of true samples $\{x_i\}$ from a mini-batch of generated samples $\{\tilde{x}_i\}$, do not work with cGANs because the statistics of the posterior are expected to significantly differ from those of the prior.

To combat mode collapse in the cGAN case, Adler et al. (Adler & Öktem, 2018) proposed to use a three-input discriminator $D_{\phi} : \mathcal{X} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ and replace $L_{adv}$ from (5) with the loss

$$L_{adv}^\text{adler}(\theta, \phi) \triangleq E_{x,z_1,z_2,y} \left\{ \frac{1}{2} D_{\phi}(x, G_{\theta}(z_1, y)) + \frac{1}{2} D_{\phi}(G_{\theta}(z_2, y), x) - D_{\phi}(G_{\theta}(z_1, y), G_{\theta}(z_2, y), y) \right\},$$

(6)

which rewards variation between the first and second inputs to $D_{\phi}$. They then proved that minimizing $L_{adv}^\text{adler}$ in place of $L_{adv}$ does not compromise the Wasserstein cGAN objective, i.e., $\arg \min_{\theta} L_{adv}^\text{adler}(\theta, \phi) = \arg \min_{\theta} L_{adv}(\theta, \phi)$.

Ohayon et al. (2021a) proposed to fight mode collapse via supervised-\(\ell_2\) regularization\(^1\) of $L_{adv}$, i.e., by solving $\arg \min_{\theta} \{ L_{adv}(\theta, \phi) + \lambda L_2(\theta) \}$ with some $\lambda > 0$ and

$$L_2(\theta) \triangleq E_{x,y} \{ ||x - E_{z}(G_{\theta}(z, y))||_2^2 \}$$

(7)

when training the generator. This regularization is consistent with the cGAN objective in that, if there exists some $\theta$ for which $p_{z|y}$ matches the true posterior $p_{z|y}$, then both $L_{adv}$ and $L_2$ are minimized by the same $\theta$. This is because $L_2(\theta)$ is minimized when $E_{z}(G_{\theta}(z, y))$ is the minimum mean-square error (MSE) estimate of $x$ from $y$, or equivalently the posterior mean $E_{z|y}(x|y)$. In practice, such a $\theta$ may not exist, in which case $L_{adv}$ and $L_2$ will act in complementary ways to drive $G_{\theta}$ towards the true-posterior generator.

However, it is important to note that practical application requires the use of a (finite) $P$-sample average with $P \geq 2$ (e.g., $P = 8$ in Ohayon et al., 2021b), i.e.,

$$L_{2,P}(\theta) \triangleq E_{x,z_1,\ldots,z_P,y} \{ ||x - \frac{1}{P} \sum_{i=1}^{P} G_{\theta}(z_i, y)||_2^2 \},$$

(8)

in place of the expectation $E_{z}$ in (7). As we show below, $L_{2,P}$ has the potential to induce mode collapse rather than prevent it, and the potential grows larger as $P$ grows smaller.

### The problem with supervised-\(\ell_2\) regularization:

To understand why supervised-\(\ell_2\) regularization using a finite-sample average can lead to mode collapse, we write (8) as

$$L_{2,P}(\theta) = E_{y} \{ E_{x,z_1,\ldots,z_P|y} \{ ||x - \tilde{x}_{i|y}||_2^2 \} \}$$

(9)

$$= E_{y} \{ ||\tilde{x}_{\text{mse}} - x||_2^2 + \frac{1}{P} E_{z|y} \{ ||d_i||_2^2 \} \} + E_{x|y} \{ ||e_{\text{mse}}||_2^2 \}$$

(10)

\(^1\)The idea to regularize a cGAN with supervised-\(\ell_2\) loss was proposed long before (Ohayon et al., 2021a); see, e.g., (Isola et al., 2017).
A Regularized Conditional GAN for Posterior Sampling

(see Appendix A for a derivation) using

\[ \hat{x}_i \triangleq G_\theta(x_i, y_i), \quad \hat{x}_{(p)} \triangleq \frac{1}{P} \sum_{i=1}^{P} \hat{x}_i, \]
\[ \overline{x} \triangleq E_{z|y}[\hat{x}_i|y], \quad \overline{x} = E_{z_1,...,z_P}[\hat{x}_{(p)}|y], \]
\[ d_i \triangleq \hat{x}_i - \overline{x}, \quad d_{(p)} \triangleq \frac{1}{P} \sum_{i=1}^{P} d_i, \]
\[ \overline{x}_{\text{mse}} \triangleq E_{z|y}[x|y], \quad e_{\text{mse}} \triangleq \overline{x} - \overline{x}_{\text{mse}}. \]

noting that \( E_{z|y}[\hat{x}_i|y] \) is invariant to \( i \) since \( \{z_i\} \) are i.i.d.

Let us now examine (10). First, note that only \( \overline{x} \) and \( d_i \) in (10) depend on the generator parameters \( \theta \). The first term in (10) encourages \( \overline{x} \) to match the MMSE estimate \( \hat{x}_{\text{mse}} \), while the second term in (10) encourages \( d_i = 0 \) or equivalently \( \hat{x}_i = \overline{x} \), which corresponds to mode collapse. Due to the \( 1/P \) term in (10), we see that, as \( P \) decreases, \( L_{2,p} \) gives a stronger incentive to mode collapse.

Experimentally, we observe that supervised-\( \ell_2 \) regularization does indeed lead to mode collapse when \( P = 2 \). Although mode collapse may not occur with larger values of \( P \), there is a high cost to using large \( P \) as a result of GPU memory constraints: as \( P \) doubles, the batch size must halve, and so training time increases linearly with \( P \). For example, in our MRI experiment, we found that \( P = 2 \) takes \( \approx 2.5 \) days to train for 100 epochs on a 4\times A100 GPU server, while \( P = 8 \) takes \( \approx 10 \) days.

The problem with variance rewards: To mitigate \( L_{2,p} \)'s incentive for mode-collapse, one might try to incorporate a variance reward (e.g., to counteract the second term in (10)). In particular, to train the generator, one could solve

\[ \arg \min_{\theta} \{ \beta_{\text{adv}} L_{\text{adv}}(\theta, \phi) + L_{2,\text{var},P}(\theta, \beta_{\text{var}}) \} \tag{12} \]

with some \( \beta_{\text{adv}}, \beta_{\text{var}} > 0 \) and \( P \geq 2 \) using

\[ L_{2,\text{var},P}(\theta, \beta_{\text{var}}) \triangleq L_{2,P}(\theta) - \beta_{\text{var}} L_{\text{var},P}(\theta) \tag{13} \]
\[ L_{\text{var},P}(\theta) \triangleq \frac{1}{P-1} \sum_{i=1}^{P} E_{z_i,...,z_P}[\|\hat{x}_i - \overline{x}_{(p)}\|_2^2]. \tag{14} \]

We show in Appendix B that (14) can be rewritten as

\[ L_{\text{var},P}(\theta) = E_y \{ E_{z|y}[\|d_i\|_2^2|y] \}, \tag{15} \]

with \( d_i \) from (11), so that

\[ L_{2,\text{var},P}(\theta, \beta_{\text{var}}) = E_y \left\{ \frac{1}{P-1} \sum_{i=1}^{P} E_{z_i,...,z_P}[\|\hat{x}_i - \overline{x}_{(p)}\|_2^2] \right\} + \frac{\beta_{\text{var}}}{P-1} E_{z|y}[\|d_i\|_2^2|y] + E_{z|y}[\|e_{\text{mse}}\|_2^2|y]. \tag{16} \]

From (16), we can see that minimizing the regularization \( L_{2,\text{var},P}(\theta, \beta_{\text{var}}) \) with \( \beta_{\text{var}} = 1/P \) encourages \( \overline{x} \) to match the MMSE estimate \( \hat{x}_{\text{mse}} \) without encouraging or discouraging mode collapse. To discourage mode collapse, one might consider using \( \beta_{\text{var}} > 1/P \), but in this case \( L_{2,\text{var},P}(\theta, \beta_{\text{var}}) \) would encourage the norm of \( d_i \) (i.e., trace covariance of \( \hat{x}_i \)) to be as large as possible, which is not our aim. We want a regularizer that encourages the covariance of \( \hat{x}_i \) to match the true posterior covariance, but it does not seem possible to do this using \( L_{2,\text{p}} \)-based regularization.

2.2. Quantifying performance using CFID

As previously stated, our goal is to train a generator \( G_\theta \) so that, for typical fixed values of \( y \), the distribution \( p_{\text{ cyl}} \) matches the true posterior \( p_{\text{ cyl}}(\cdot|y) \). It is essential to have a quantitative metric for evaluating performance with respect to this goal. For example, it is not enough that the generated samples are “accurate” in the sense that \( \hat{x}_i \) or \( \overline{x}_{(p)} \) are close to the ground truth \( x_i \), nor is it enough that \( \hat{x}_i \) are “diverse” in the sense of having a large element-wise SD.

We quantify the performance of posterior approximation using the conditional Fréchet Inception distance (CFID) (Solvei et al., 2021), which is a computationally efficient approximation to the conditional Wasserstein distance

\[ \text{CWD} \triangleq E_y \{ W_2(p_{\text{ cyl}}(\cdot, y), p_{\text{ cyl}}(\cdot, y)) \}. \tag{17} \]

In (17), \( W_2(p_a, p_b) \) denotes the Wasserstein-2 distance between distributions \( p_a \) and \( p_b \), defined as

\[ W_2(p_a, p_b) \triangleq \min_{p_{ab} \in \Pi(p_a,p_b)} E_{a,b}[\|a - b\|_2^2] \tag{18} \]
\[ \Pi(p_a,p_b) \triangleq \{ p_{ab} : p_a = \int p_{ab} \, db \text{ and } p_b = \int p_{ab} \, da \}, \]

where \( \Pi(p_a,p_b) \) denotes the set of joint distributions \( p_{ab} \) with prescribed marginals \( p_a \) and \( p_b \). Similar to how FID—a popular uGAN metric—is computed, CFID approximates CWD (17) as follows: i) the random vectors \( x \), \( \hat{x} \), and \( y \) are replaced by low-dimensional embeddings \( x, \hat{x} \), and \( y \), generated by the convolutional layers of a deep network, and ii) the embedding distributions \( p_{\text{ cyl}} \) and \( p_{\text{ cyl}} \) are approximated by Gaussians \( N(\mu_{\text{ cyl}}, \Sigma_{\text{ cyl}}) \) and \( N(\mu_{\text{ cyl}}, \Sigma_{\text{ cyl}}) \). With these approximations, the CWD reduces to

\[ \text{CFID} \triangleq E_y \left\{ \|\mu_{\text{ cyl}} - \mu_{\text{ cyl}}\|_2^2 + \text{tr} \left( \Sigma_{\text{ cyl}} + \Sigma_{\text{ cyl}} \right) - 2\left( \Sigma_{\text{ cyl}}^{1/2} \Sigma_{\text{ cyl}}^{1/2} \right)^{1/2} \right\}. \tag{19} \]

In practice, the expectations, means, and covariances in (19) are replaced by sample averages using samples \( \{(x_i, y_i)\} \) from a test set. For more on CFID, please see Appendix E.

3. Proposed method

3.1. Regularization: Supervised-\( \ell_1 \) plus SD reward

We now propose a different form of cGAN regularization that, unlike those previously discussed, encourages the samples \( \hat{x}_i \) to match the true posterior in both mean and covariance. In particular, when training the generator, we propose to solve

\[ \arg \min_{\theta} \{ \beta_{\text{adv}} L_{\text{adv}}(\theta, \phi) + L_{1,SD,P}(\theta, \beta_{\text{SD}}) \} \tag{20} \]

with some \( \beta_{\text{adv}}, \beta_{\text{SD}} > 0 \) and \( P \geq 2 \), where regularizer

\[ L_{1,SD,P}(\theta, \beta_{\text{SD}}) \triangleq L_1,P(\theta) - \beta_{\text{SD}} L_{SD,P}(\theta) \tag{21} \]
We note that regularizing a cGAN with supervised-\(\ell_1\) loss and SD reward:

\[
L_{1,P}(\theta) \triangleq E_{x,z_1,...,z_P,y} \left\{ \| x - \hat{x}_{i(P)} \|_1 \right\}
\tag{22}
\]

\[
L_{\text{SD},P}(\theta) \triangleq \frac{2}{\pi P(P+1)} \sum_{i=1}^{P} E_{z_1,...,z_P,y} \left\{ \| \hat{x}_i - \hat{x}_{i(P)} \|_1 \right\},
\tag{23}
\]

where \(\hat{x}_i\) and \(\hat{x}_{i(P)}\) were defined in (11).

In practice, we find that \(P = 2\) works best (in accordance with Figure 1). From a computational perspective, this is highly advantageous because, as discussed earlier, the time to train the network increases linearly with \(P\).

We note that regularizing a cGAN with supervised-\(\ell_1\) loss alone is not new; see, e.g., (Isola et al., 2017). In fact, for image recovery in general, the use of supervised-\(\ell_1\) loss is often preferred over \(\ell_2\) because it results in sharper, more visually pleasing results (Zhao et al., 2017). But regularizing a cGAN using supervised-\(\ell_1\) loss alone can push the generator towards mode collapse, for reasons similar to the supervised-\(\ell_2\) case discussed earlier. For example, \(\ell_1\)-induced mode collapse was observed in (Isola et al., 2017), which led the authors to use dropout instead of random codes \(z\) to induce variation in the generator output.

Our contribution is to use a properly weighted SD reward in conjunction with supervised-\(\ell_1\) loss, as in (21). As we show below, under certain assumptions, the SD reward works together with the \(\ell_1\) loss to enforce the correctness of both the posterior mean and the posterior covariance. This stands in contrast to the case of \(\ell_2\) loss with a variance reward, which enforces only the correctness of the posterior mean.

**Proposition 3.1.** Assume that \(P \geq 2\) and that \(\theta\) has complete control over the \(y\)-conditional mean and covariance of \(\hat{x}_i\). Then the regularization-minimizing parameters \(\theta^*_s = \text{arg min}_\theta L_{1,SD,P}(\theta, \beta^s_{SD})\) with

\[
\beta_{SD}^N \triangleq \sqrt{\frac{2 \pi P(P+1)}{\pi P(P+1)}}
\tag{24}
\]

yield generated statistics

\[
E_{z_i | y} \{ \hat{x}_i(\theta_s) | y \} = E_{z_i | y} \{ x | y \} = \hat{x}_{\text{mmse}}
\tag{25a}
\]

\[
\text{Cov}_{z_i | y} \{ \hat{x}_i(\theta_s) | y \} = \text{Cov}_{z_i | y} \{ x | y \}
\tag{25b}
\]

when the elements of \(\hat{x}_i\) and \(x\) are independent Gaussian conditioned on \(y\). Thus, the \(L_{1,SD,P}\) regularization encourages the \(y\)-conditional mean and covariance of \(\hat{x}_i\) to match those of the true \(x\).

**Proof.** See Appendix C. \(\square\)

In most applications, \(\hat{x}_i\) and \(x\) are not expected to be independent Gaussian conditioned on \(y\), as assumed in Proposition 3.1. So, using \(\beta_{SD}^N\) from (24) may not work well in practice. We thus propose a method to automatically determine the correct \(\beta_{SD}\) in Section 3.2.

### Scalar-Gaussian Illustration

To illustrate the behavior of the previously described regularizers, we consider the simple scalar-Gaussian case, where the generator is \(G_\theta(z,y) = \mu + \sigma z\), with code \(z \sim N(0,1)\) and parameters \(\theta = [\mu, \sigma]^\top\). In this case, the generated posterior is \(p_{\hat{x}|y}(x|y) \sim N(x; \mu, \sigma^2)\), and we assume that the true posterior is \(p_{x|y}(x|y) \sim N(x; \mu_0, \sigma_0^2)\) for some \(\mu_0\) and \(\sigma_0 > 0\).

Figure 1(a) plots \(P\)-sample supervised-\(\ell_2\) loss \(L_{2,P}(\theta)\) from (8) versus \(\theta\) for \(P = 8\). The plot shows that the minimizing \(\theta = \text{arg min}_\theta L_{2,P}(\theta)\) yields \(\hat{\mu} = \mu_0\) and \(\hat{\sigma} = 0\), which corresponds to mode collapse. Figure 1(b) plots \(P\)-sample supervised-\(\ell_2\) loss plus variance reward \(L_{2,\text{var},P}(\theta, \beta_{\text{var}})\) from (13) versus \(\theta\) for \(\beta_{\text{var}} = 1/P\) and \(P = 8\). The choice of \(\beta_{\text{var}}\) cancels the middle term in (10), which is responsible for mode collapse. As can be seen in Figure 1(b), \(\theta = \text{arg min}_\theta L_{2,\text{var},P}(\theta, \beta_{\text{var}})\) is no longer unique, and there is no incentive for mode collapse (i.e., \(\hat{\sigma} = 0\)) but no incentive to match the true variance (i.e., \(\hat{\sigma} = \sigma_0\)). Figure 1(c) plots \(P\)-sample supervised-\(\ell_1\) loss plus SD reward \(L_{1,SD,P}(\theta, \beta_{SD})\) versus \(\theta\) from (21) for \(\beta_{SD} = \beta_{SD}^N\) from (24) and \(P = 8\). The plots shows that \(\theta = \text{arg min}_\theta L_{1,SD,P}(\theta, \beta_{SD})\) recovers the true parameters (i.e., \(\theta = [\mu_0, \sigma_0]^\top\)) as predicted by Proposition 3.1.

Finally, Figure 1(d) repeats the experiment from Figure 1(c), but with \(P = 2\). The plot shows \(\theta = [\mu_0, \sigma_0]^\top\) as predicted by Proposition 3.1. But, comparing Figure 1(d) to Figure 1(c), we can see that the cost surface becomes steeper for smaller \(P\), i.e., the regularization becomes stronger.

In real-world experiments, we find that \(L_{1,SD,P}(\theta, \beta_{SD})\) regularized cGANs work best when trained with the minimum \(P\), i.e., \(P = 2\).

### 3.2 Auto-tuning of \(\beta_{SD}\)

We now propose a method to auto-tune \(\beta_{SD}\) in (21) for a given training dataset. Our approach is inspired by the observation that larger values of \(\beta_{SD}\) tend to yield samples \(\hat{x}_i\) with more variation. But more variation is not necessarily better; we want samples with the correct amount of variation. To assess variation, we will compare the expected \(\ell_2\) error of the \(P\)-sample average \(\hat{x}_{i(P)}\) to that of \(\hat{x}_{i(1)}\). In the case of mode collapse, these errors are identical. But when \(\{\hat{x}_i\}\) are true posterior samples, these errors follow a particular relationship, as established by Proposition 3.2.

**Proposition 3.2.** Given generator outputs \(\{\hat{x}_i\}\) and their \(P\)-sample average \(\hat{x}_{i(P)} \triangleq \frac{1}{P} \sum_{i=1}^{P} \hat{x}_i\), let us define the expected \(\ell_2\) error on \(\hat{x}_{i(P)}\) as

\[
\mathcal{E}_P \triangleq E \{ \| \hat{x}_{i(P)} - x \|_2^2 | y \}.
\tag{26}
\]

If \(\{\hat{x}_i\}\) are independent samples of the true posterior (i.e., \(\hat{x}_i \sim p_{\hat{x}|y}(x|y)\)), then

\[
\mathcal{E}_P = \frac{P+1}{P} \mathcal{E}_{\text{mmse}} \text{ and so } \frac{\mathcal{E}_P}{\mathcal{E}_{\text{mmse}}} = \frac{P+1}{P}.
\tag{27}
\]
Figure 1. Scalar-Gaussian illustration of four different regularizers: (a) supervised-ℓ₂ at P = 8, (b) supervised-ℓ₂ plus variance reward with βSD = 1/P at P = 8, (c) supervised-ℓ₁ plus SD reward with βSD = βSD,N at P = 8, and (d) supervised-ℓ₁ plus SD reward with βSD = βSD,N at P = 2. The cost contours show the regularization value versus θ = [μ, σ]ᵀ, and the red star shows the true posterior parameters [μ₀, σ₀]ᵀ. Subplot (a) shows that supervised-ℓ₂ promotes mode collapse (i.e., ̂σ = 0). Subplot (b) shows that adding a variance reward to supervised-ℓ₂ discourages mode collapse but does not recover the true parameters [μ₀, σ₀]ᵀ. Subplots (c)-(d) show that supervised-ℓ₁ plus SD reward recovers the true parameters [μ₀, σ₀]ᵀ for any P ≥ 2. Comparing subplots (c) and (d), a smaller P yields a stronger effect.

Figure 2. Example PSNR of ̂x_i(P) versus P, the number of averaged outputs, for several values of βSD used during training. Also shown is the theoretical behavior for true-posterior samples.

Proof. See Appendix D.

Figure 2 shows that, for any P ≥ 2, E₁/E_P grows with βSD. Together, Proposition 3.2 and Figure 2 suggest to adjust βSD so that E₁/E_P ratio obeys (27). In practice, at each epoch t, we approximate E_P and E₁ using a validation pair (x_t, y_t) as follows:

\[
\begin{align*}
\tilde{E}_P &= \frac{1}{P} \sum_{i=1}^{P} G_θ(z_i, y_i) - x_i \|_2^2 \\
\tilde{E}_1 &= \frac{1}{P} \sum_{i=1}^{P} G_θ(z_i, y_i) - x_i \|_2^2,
\end{align*}
\]

(28) (29)

where i.i.d. codes \{z_i\}_i=1^P are drawn independently of x_t and y_t. We then update βSD using gradient descent:

\[
β_{SD,t+1} = β_{SD,t} - μ_{SD} \left( \frac{\tilde{E}_P}{P} \frac{1}{\tilde{E}_1} \right) \left[ 2 \frac{P}{1+P} \right] \beta_{SD}^N
\]

(30)

with β_{SD,0} = β_{SD,N}, some μ_{SD} > 0, and [x]db ≡ 10 log_{10}(x).

3.3. Enforcing data consistency

In this section we describe a data-consistency procedure that can be optionally used with our cGAN. The motivation is that, in some applications such as medical imaging or inpainting, the end user may feel comfortable knowing that all generated reconstructions ̂x_i of x from y = Ax + w (recall (1)) are consistent with the measurements in that

\[
y = A ̂x_i.
\]

When (31) holds, A⁺y = A⁺A ̂x_i must also hold, where (·)⁺ denotes the pseudo-inverse. The quantity A⁺ can be recognized as the orthogonal projection matrix associated with the row space of A. So, (31) requires the components of ̂x_i in the row space of A to equal A⁺y, while placing no constraints on the components of ̂x_i in the null space of A. This suggests the following data-consistency procedure:

\[
\tilde{x}_i = (I - A⁺A) ̂x_i^{raw} + A⁺y.
\]

(32)

where ̂x_i^{raw} is the raw generator output.

By employing the data-consistency procedure (32), the overall recovery method will attempt to recover only the components of x that lie in the nullspace of A. Consequently, (32) is admissible only when A has a non-trivial nullspace. Also, because no attempt is made to remove the noise w in y, this approach is recommended for low-noise applications.

4. Numerical experiments

4.1. MRI

Data: In the MRI version of (1), x is a complex-valued multicoil image (see Appendix F for details). For the training
\{ x_i \}, we used the first 8 slices of all fastMRI (Zbontar et al., 2018) T2 brain training volumes with at least 8 coils, cropping them to 384 \times 384 pixels and compressing to 8 virtual coils (Zhang et al., 2013), yielding 12,200 training images. Then 2,376 testing and 784 validation images were obtained in the same manner from the fastMRI T2 brain testing volumes. From the 2,376 testing images, we randomly selected 72 from which to compute performance metrics, in order to limit the generation time of our Langevin competitor (Jalal et al., 2021a) to roughly 6 days. To create measurement data \( y_t \), we transformed \( x_t \) to the Fourier domain, sampled using the pseudo-random GRO pattern (Ahmad et al., 2015) at acceleration \( R = 4 \), and transformed the zero-filled k-space measurements back to the (complex, multicoil) image domain.

**Architecture:** We used a U-Net (Ronneberger et al., 2015) for the generator and a standard CNN for the discriminator (see Appendix H for more details). The discriminator was patch-based (Isola et al., 2017) since that gave slightly improved performance. Also, we used the data-consistency processing described in Appendix F. We then converted the images to the magnitude domain before computing FID, CFID, PSNR, SSIM, and APSD were evaluated from the P-averaged outputs \( \hat{x}_{(r)} \) (recall (11)), while FID and CFID were computed from the un-averaged outputs \( \hat{x}_i \). We used \( P = 8 \) for validation and \( P = 32 \) for testing, and computed FID and CFID using VGG-16 (not Inception-v3) for better agreement with radiologist’s perceptions (Krastraulyin et al., 2022).

**Competitors:** We compare our cGAN to the cGAN from (Adler & Öktem, 2018), the cGAN from (Ohayon et al., 2021a), and the fastMRI Langevin approach from (Jalal et al., 2021a). The cGAN from (Adler & Öktem, 2018) uses generator loss \( \beta_{adv}\mathcal{L}_{adv}(\theta, \phi) \) and discriminator loss \( -\mathcal{L}_{adv}(\theta, \phi) + \alpha_1 \mathcal{L}_{gp}(\phi) + \alpha_2 \mathcal{L}_{drift}(\phi) \), while the cGAN from (Ohayon et al., 2021a) uses generator loss \( \beta_{adv}\mathcal{L}_{adv}(\theta, \phi) + \mathcal{L}_{2,p}(\theta) \) and discriminator loss \( -\mathcal{L}_{adv}(\theta, \phi) + \alpha_1 \mathcal{L}_{gp}(\phi) + \alpha_2 \mathcal{L}_{drift}(\phi) \). Each used the value of \( \beta_{adv} \) specified in the original paper. All cGANs used the same generator and discriminator architectures, except that (Adler & Öktem, 2018) used extra discriminator input channels to facilitate the 3-input loss (6). For the fastMRI Langevin approach (Jalal et al., 2021a), we did not modify the authors’ implementation in (Jalal et al., 2021b) except for the undersampling mask.

**Reconstruction results:** Table 1 shows performance results and generation time (for 4 samples). CFID, FID, PSNR, SSIM, APSD were evaluated on only 72 test images due to the long generation time of the Langevin competitor (Jalal et al., 2021a). But because CFID and FID are known to be biased with small sample sizes (Soloveitchik et al., 2021), we re-evaluated CFID and FID on all 2,376 test images, and re-evaluated CFID on all 14,576 training and test images, for all but the Langevin method (due to computational constraints).

Table 1: Average results for \( R = 4 \) MRI reconstruction of 384 \times 384 T2 brain images. \( \pm \) shows standard error. (C)FID\(^2\), PSNR, SSIM, and APSD used 72 test samples and \( P = 32 \). (C)FID\(^2\) used 2,376 test samples and \( P = 8 \). CFID\(^3\) used all 14,576 samples and \( P = 1 \).

| Model | CFID\(^1\) | CFID\(^2\) | CFID\(^3\) | FID\(^1\) | FID\(^2\) | PSNR\(^\dagger\) | SSIM\(^\dagger\) | APSD | Time (4)\(^\dagger\) |
|-------|------------|------------|------------|----------|----------|----------------|----------------|------|-----------------|
| Langevin (Jalal et al., 2021a) | 5.29 | - | - | 6.12 | - | 37.88 ± 0.41 | 0.9042 ± 0.0062 | 5.96-6 | 14 min |
| cGAN (Adler & Öktem, 2018) | 6.39 | 4.27 | 3.82 | 5.25 | 2.86 | 37.28 ± 0.19 | 0.9412 ± 0.0031 | 3.96-6 | 217 ms |
| cGAN (Ohayon et al., 2021a) | 4.06 | 3.27 | 2.95 | 6.45 | 3.61 | 39.40 ± 0.24 | 0.9486 ± 0.0031 | 7.2-8 | 217 ms |
| cGAN (Ours) | **3.10** | **1.54** | **1.29** | **3.75** | **1.22** | **39.70 ± 0.24** | **0.9507 ± 0.0030** | **3.8e-6** | **217 ms** |

**Validation/Testing:** To evaluate performance, we converted the multicoil generator outputs to complex-valued images using SENSE-based coil combining (Pruessmann et al., 1999) with ESPiRiT-estimated (Uecker et al., 2014) coil sensitivity maps (via SigPy (Ong & Lustig, 2019)) as described in Appendix F. We then converted the images to the magnitude domain before computing FID, CFID, PSNR, SSIM, and APSD. We used the same generator and discriminator architectures, except for the undersampling mask.

From the 2,376 testing images, we randomly selected 72 for validation. From the 2,376 testing images, we randomly selected 72 from which to compute performance metrics, in order to limit the generation time of our Langevin competitor (Jalal et al., 2021a) to roughly 6 days. To create measurement data \( y_t \), we transformed \( x_t \) to the Fourier domain, sampled using the pseudo-random GRO pattern (Ahmad et al., 2015) at acceleration \( R = 4 \), and transformed the zero-filled k-space measurements back to the (complex, multicoil) image domain.
A Regularized Conditional GAN for Posterior Sampling

Figure 3. MRI reconstruction of a test image. Column one: reconstruction \( \hat{x}^{(P)} \), two: pixel-wise absolute error \( |\hat{x}^{(P)} - x| \), three: pixel-wise SD \( \left( \frac{1}{P} \sum_{i=1}^{P} (\hat{x}_i - \hat{x}^{(P)}_i)^2 \right)^{1/2} \).

that the APSD of Ohayon et al.’s cGAN was almost two orders of magnitude lower than the other APSDs, indicating mode collapse. Table 1 also quantifies the bias of CFID and FID at small sample sizes, and Appendix G.1 demonstrates that the CFID bias is mostly due to the covariance component of CFID. The cGANs generated samples 3 800 times faster than the Langevin approach.

Figure 3 shows example test-image reconstructions for the four different methods under test, along with the corresponding pixel-wise absolute errors \( |\hat{x}^{(P)} - x| \) and pixel-wise SD \( \left( \frac{1}{P} \sum_{i=1}^{P} (\hat{x}_i - \hat{x}^{(P)}_i)^2 \right)^{1/2} \). The mode collapse of Ohayon et al.’s cGAN is evident from the dark pixel-wise SD image. The fact that the cGAN errors are less than the Langevin errors near the image corners is a consequence of minor differences in sensitivity-map estimation relative to (Jalal et al., 2021b). Appendix I shows several other test-image reconstructions.

\( \beta_{SD} \) auto-tuning results: Figure 4 shows the dB difference between \( P \)-sample PSNR gain \( \hat{E}_{1,t}/\hat{E}_{P,t} \) and the theoretical value \( 2P/(P+1) \) versus the training epoch \( t \) for \( P = 8 \), as used during validation. As discussed in Section 3.2, the observed \( P \)-sample PSNR gain \( \hat{E}_{1,t}/\hat{E}_{P,t} \) is dependent on

\[ \beta_{SD} \text{, which is adapted according to (30). Figure 5 shows the PSNR of } \hat{x}^{(P)} \text{ versus } P \text{ after auto-tuning. The figure shows that the observed curve closely matches the theoretical curve corresponding to true-posterior samples.} \]

4.2. Inpainting

Data: For our inpainting experiment, our objective was to complete the missing centered 128 × 128 square of a 256 × 256 CelebA-HQ face image (Karras et al., 2018). We randomly split the dataset, yielding 27 000 images for training, 2 000 for validation, and 1 000 for testing.

Architecture: For our cGAN, we use the CoModGAN architecture from (Zhao et al., 2021a) along with our proposed \( L_{1,SD,P} \) regularization, but unlike (Zhao et al., 2021a) we did not use MBSD (Karras et al., 2018) at the discriminator.

Training/Validation/Testing: We use the same general training and testing procedure described in Section 4.1, but with \( \beta_{adv} = 5e-3 \), \( n_{batch} = 100 \), and 110 epochs of cGAN training. Running PyTorch on a server with 4 Tesla A100 GPUs, each with 82 GB of memory, the cGAN training takes approximately 2 days. FID, PSNR, and SSIM were evaluated on 1 000 test images using \( P = 32 \) samples per measurement. To avoid the bias that would result from evaluating CFID on only 1 000 images (see Appendix G.1), CFID was evaluated on all 30 000 images with \( P = 1 \).

Competitors: We compare with the state-of-the-art CoModGAN (Zhao et al., 2021a) and Score-based SDE (Song et al., 2021) approaches. For CoModGAN, we use the PyTorch implementation from (Zeng, 2022). CoModGAN differs from our cGAN only in its use of MBSD and lack of \( L_{1,SD,P} \) regularization. For Song et al.’s SDE, we use the
Table 2. Average results for inpainting a 128 × 128 centered square on a 256 × 256 celebA-HQ image. FID, PSNR, and SSIM were computed from 1 000 test images with \( P = 32 \), while CFID was computed from all 30 000 images with \( P = 1 \).

| Model                        | CFID↓ | FID↓ | PSNR↑ | SSIM↑ | Time (128)↓ |
|------------------------------|-------|------|-------|-------|-------------|
| Score-SDE (Song et al., 2021) | 5.11  | 7.92 | 26.11 ± 0.09 | 0.9025 ± 0.0007 | 48 min |
| CoModGAN (Zhao et al., 2021a) | 5.29  | 8.50 | 25.45 ± 0.06 | 0.8882 ± 0.0006 | 217 ms |
| Ours                         | 4.69  | 7.45 | 26.61 ± 0.07 | 0.9037 ± 0.0007 | 217 ms |

Reconstruction results: Table 2 shows the test CFID, FID, PSNR, SSIM, and generation time (for 128 samples). The table shows that our approach gave the best results in all metrics. Overall, the cGANs generated samples 13 272 times faster than the score-based method.

Figure 6 shows five generated samples for the three methods under test. There, the samples generated by our approach outperform CoModGAN in terms of sample quality and are more diverse than those generated by Song et al.’s SDE. Additional examples can be found in Appendix I.

5. Conclusion

We proposed a novel regularization technique for cGANs which consists of supervised-\( \ell_1 \) loss plus an appropriately weighted SD reward, i.e., \( \mathcal{L}_{1,P}(\theta) - \beta_{SD} \mathcal{L}_{SD,P}(\theta) \). For the simple case of an independent Gaussian posterior, we showed that, with appropriate \( \beta_{SD} \), minimizing our regularization yields generator samples that agree with the true posterior samples in both mean and covariance. For practical data, we proposed a method to auto-tune \( \beta_{SD} \).

For multicoil MR reconstruction and large-scale image inpainting, experiments showed our proposed method (with appropriate choice of generator and discriminator architecture) outperforming all cGAN and score-based competitors in CFID as well as in accuracy metrics like PSNR and SSIM and perceptual metrics like FID. Compared to the score-based approaches, our method generated samples thousands of times faster.

One limitation of our current implementation is that our cGAN is trained for a specific \( A \) in (1). In ongoing work, we are extending our approach so that it can be trained to handle a wide range of \( A \) matrices. In addition, we hope to extend our approach to other applications, such as CT, super-resolution, deblurring, compressive sensing, and denoising.

Figure 6. Example of inpainting a 128 × 128 centered square on a 256 × 256 resolution celebA-HQ test image.
References

Adler, J. and Öktem, O. Deep Bayesian inversion. *arXiv:1811.05910*, 2018. 1, 2, 6, 7, 16, 17, 18, 19, 20, 21, 22

Ahmad, R., Xue, H., Giri, S., Ding, Y., Craft, J., and Simonetti, O. P. Variable density incoherent spatiotemporal acquisition (VISTA) for highly accelerated cardiac MRI. *Magn. Reson. Med.*, 74, 2015. 6

Ardizzone, L., Lüth, C., Kruse, J., Rother, C., and Köthe, U. Guided image generation with conditional invertible neural networks. *arXiv:1907.02392*, 2019. 1

Deora, P., Vasudeva, B., Bhattacharya, S., and Pradhan, P. M. Structure preserving compressive sensing MRI reconstruction using generative adversarial networks. In *Proc. IEEE Conf. Comp. Vision Pattern Recog. Workshop*, pp. 2211–2219, June 2020. 17

Edupuganti, V., Mardani, M., Vasanawala, S., and Pauly, J. Uncertainty quantification in deep MRI reconstruction. *IEEE Trans. Med. Imag.*, 40(1):239–250, January 2021. 1

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. Improved training of Wasserstein GANs. In *Proc. Neural Inf. Process. Syst. Conf.*, pp. 5769–5779, 2017. 2, 6

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Proc. Neural Inf. Process. Syst. Conf.*, volume 30, 2017. 1

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pp. 1125–1134, 2017. 1, 2, 4, 6, 17

Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A., and Tamir, J. Robust compressed sensing MRI with deep generative priors. In *Proc. Neural Inf. Process. Syst. Conf.*, 2021a. 1, 6, 7, 16, 17, 18, 19, 20, 21, 22

Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A., and Tamir, J. csrgm-mri-langevin. https://github.com/utcsilab/csrgm-mri-langevin, 2021b. Accessed: 2021-12-05. 6, 7

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. Int. Conf. on Learn. Rep.*, 2018. 2, 6, 7

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. Training generative adversarial networks with limited data. In *Proc. Neural Inf. Process. Syst. Conf.*, volume 33, pp. 12104–12114, 2020. 2

Kastryulin, S., Zakirov, J., Pezzotti, N., and Dylov, D. V. Image quality assessment for magnetic resonance imaging. *arXiv:2203.07809*, 2022. 6, 15

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proc. Int. Conf. on Learn. Rep.*, 2015. 6

Leone, F. C., Nelson, L. S., and Nottingham, R. The folded normal distribution. *Technometrics*, 3(4):543–550, 1961. 12, 13

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014. 2

Ohayon, G., Adrai, T., Vaksman, G., Elad, M., and Milanfar, P. High perceptual quality image denoising with a posterior sampling CGAN. In *Proc. IEEE Conf. Comput. Vis. Workshops*, volume 10, pp. 1805–1813, 2021a. 1, 2, 6, 7, 16, 18, 19, 20, 21, 22

Ohayon, G., Adrai, T., Vaksman, G., Elad, M., and Milanfar, P. High perceptual quality image denoising with a posterior sampling CGAN. Downloaded from https://github.com/theoad/pscgan, July 2021b. 2

Ong, F. and Lustig, M. SigPy: A python package for high performance iterative reconstruction. In *Proc. Annu. Meeting ISMRM*, volume 4819, 2019. 6

Pruessmann, K. P., Weiger, M., Scheidegger, M. B., and Boesiger, P. SENSE: Sensitivity encoding for fast MRI. *Magn. Reson. Med.*, 42(5):952–962, 1999. 6, 15

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Proc. Intl. Conf. Med. Image Comput. Comput. Assist. Intervent.*, pp. 234–241, 2015. 6, 16

Schonfeld, E., Schiele, B., and Khoreva, A. A U-Net based discriminator for generative adversarial networks. In *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pp. 8207–8216, 2020. 2

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 15

Soloveitchik, M., Diskin, T., Morin, E., and Wiesel, A. Conditional Frechet inception distance. *arXiv:2103.11521*, 2021. 1, 3, 6, 14, 15, 16

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *Proc. Int. Conf. on Learn. Rep.*, 2021. 1, 7, 8, 16, 17, 23, 24, 25, 26

Song, Y., Shen, L., Xing, L., and Ermon, S. Solving inverse problems in medical imaging with score-based generative models. In *Proc. Int. Conf. on Learn. Rep.*, 2022a. 1
A Regularized Conditional GAN for Posterior Sampling

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. Downloaded from https://github.com/yang-song/score_sde_pytorch, October 2022b.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In Proc. IEEE Conf. Comp. Vision Pattern Recog., 2016.

Tonolini, F., Radford, J., Turpin, A., Faccio, D., and Murray-Smith, R. Variational inference for computational imaging inverse problems. J. Mach. Learn. Res., 21(179):1–46, 2020.

Uecker, M., Lai, P., Murphy, M. J., Virtue, P., Elad, M., Pauly, J. M., Vasanawala, S. S., and Lustig, M. ESPIRiT—an eigenvalue approach to autocalibrating parallel MRI: Where SENSE meets GRAPPA. Magn. Reson. Med., 71(3):990–1001, 2014.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: From error visibility to structural similarity. IEEE Trans. Image Process., 13(4):600–612, April 2004.

Winkler, C., Worrall, D., Hoogeboom, E., and Welling, M. Learning likelihoods with conditional normalizing flows. arXiv preprint arXiv:1912.00042, 2019.

Zhontar, J., Knoll, F., Sriram, A., Muckley, M. J., Bruno, M., Defazio, A., Parente, M., Geras, K. J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdal, M., Romero, A., Rabbat, M., Vincent, P., Pinkerton, J., Wang, D., Yakubova, N., Owens, E., Zitnick, C. L., Recht, M. P., Sodickson, D. K., and Lui, Y. W. fastMRI: An open dataset and benchmarks for accelerated MRI. arXiv:1811.08839, 2018.

Zeng, Y. co-mod-gan-pytorch. Downloaded from https://github.com/zengxianyu/co-mod-gan-pytorch, September 2022.

Zhang, T., Pauly, J. M., Vasanawala, S. S., and Lustig, M. Coil compression for accelerated imaging with Cartesian sampling. Magn. Reson. Med., 69(2):571–582, 2013.

Zhao, H., Gallo, O., Frosio, I., and Kautz, J. Loss functions for image restoration with neural networks. IEEE Trans. Comput. Imag., 3(1):47–57, March 2017.

Zhao, S., Cui, J., Sheng, Y., Dong, Y., Liang, X., Chang, E. I.-C., and Xu, Y. Large scale image completion via co-modulated generative adversarial networks. In Proc. Int. Conf. on Learn. Rep., 2021a.

Zhao, Z., Singh, S., Lee, H., Zhang, Z., Odena, A., and Zhang, H. Improved consistency regularization for GANs. In Proc. AAAI Conf. Artificial Intell., volume 35, pp. 11033–11041, 2021b.
A. Derivation of (10)

Using the definitions of $\hat{x}_{(p)}$, $\mathbf{x}$, $d_{(p)}$, $\hat{x}_{\text{mmse}}$, $e_{\text{mmse}}$ from (11), and leveraging the fact that $\hat{x}_{\text{mmse}}$ and $\mathbf{x}$ are deterministic given $y$, we can write the inner term in (9) as

$$E_{x,z_1,\ldots,z_p | y} \{ \| x - \hat{x}_{(p)} \|^2_2 \}$$

$$= E_{x,z_1,\ldots,z_p | y} \{ \| \hat{x}_{\text{mmse}} + e_{\text{mmse}} - x - d_{(p)} \|^2_2 \}$$

$$= E_{x,z_1,\ldots,z_p | y} \{ \| \hat{x}_{\text{mmse}} - x \|^2_2 \} + 2 \text{Re} E_{x,z_1,\ldots,z_p | y} \{ (\hat{x}_{\text{mmse}} - x)^H (e_{\text{mmse}} - d_{(p)}) | y \} + E_{x,z_1,\ldots,z_p | y} \{ \| e_{\text{mmse}} - d_{(p)} \|^2_2 \}$$

$$= \| \hat{x}_{\text{mmse}} - x \|^2_2 + 2 \text{Re} \left[ (\hat{x}_{\text{mmse}} - x)^H E_{x,z_1,\ldots,z_p | y} \{ (e_{\text{mmse}} - d_{(p)}) | y \} \right] + E_{x,z_1,\ldots,z_p | y} \{ \| e_{\text{mmse}} - d_{(p)} \|^2_2 \}$$

(A.1)

Finally, we can leverage the fact that $\{z_i\}$ are i.i.d. to write

$$E_{x,z_1,\ldots,z_p | y} \{ \| e_{\text{mmse}} - d_{(p)} \|^2_2 \} = E_{x,z_1,\ldots,z_p | y} \{ \| \frac{1}{p} \sum_{j=1}^P d_j \|^2_2 \}$$

(A.2)

$$= \frac{1}{p} \sum_{j=1}^P E_{x | y} \{ \| d_j \|^2_2 \}$$

(A.3)

where in (A.2) we used the fact that $d_{(p)}$ and $e_{\text{mmse}}$ are both zero-mean when conditioned on $y$. Furthermore we can leverage the fact that $\{z_i\}$ are independent of $x$ and $y$ to write

$$E_{x,z_1,\ldots,z_p | y} \{ \| e_{\text{mmse}} - d_{(p)} \|^2_2 \} = E_{x,z_1,\ldots,z_p | y} \{ \| \frac{1}{p} \sum_{j=1}^P d_j \|^2_2 \}$$

(A.4)

$$= \frac{1}{p} E_{x | y} \{ \| d_i \|^2_2 \}$$

(A.5)

B. Derivation of (15)

To show that the expression for $\mathcal{L}_{\text{var}, P}$ from (15) holds, we first rewrite (14) as

$$\mathcal{L}_{\text{var}, P}(\theta) = \frac{1}{P-1} \sum_{i=1}^P E_y \{ \| \hat{x}_i - \hat{x}_{(p)} \|^2_2 \}$$

(B.1)

where the definitions in (11) imply

$$E_{z_1,\ldots,z_p | y} \{ \| \hat{x}_i - \hat{x}_{(p)} \|^2_2 \} = E_{z_1,\ldots,z_p | y} \{ \| \hat{x}_i + d_i - d_{(p)} - \mathbf{x} \|^2_2 \}$$

(B.2)

$$= E_{z_1,\ldots,z_p | y} \{ \| d_i - \frac{1}{P} \sum_{j=1}^P d_j \|^2_2 \}$$

(B.3)

$$= E_{z_1,\ldots,z_p | y} \{ \| (1 - \frac{1}{P}) d_i - \frac{1}{P} \sum_{j \neq i} d_j \|^2_2 \}$$

(B.4)

$$= \left(1 - \frac{1}{P}\right)^2 E_{z_1 | y} \{ \| d_i \|^2_2 \} + \frac{P-1}{P^2} E_{z_1 | y} \{ \| d_i \|^2_2 \}$$

(B.5)

For (B.5), we leveraged the zero-mean and i.i.d. nature of $\{e_i\}$ conditioned on $y$. By plugging (B.6) into (B.1), we get

$$\mathcal{L}_{\text{var}, P}(\theta) = \frac{1}{P} \sum_{i=1}^P E_y \{ \| d_i \|^2_2 \}$$

(B.6)

For (B.5), we leveraged the zero-mean and i.i.d. nature of $\{e_i\}$ conditioned on $y$. By plugging (B.6) into (B.1), we get

$$\mathcal{L}_{\text{var}, P}(\theta) = \frac{1}{P} \sum_{i=1}^P E_y \{ \| d_i \|^2_2 \}$$

(B.7)

where (B.8) follows because $\{d_i\}$ are i.i.d. when conditioned on $y$. Note that (B.8) equals the trace of the covariance of $\hat{x}_i$. 

Supplementary Materials
C. Proof of Proposition 3.1

Here we prove Proposition 3.1. To begin, we rewrite (22)-(23) as

\[ L_{1,P}(\theta) = \sum_{j=1}^{N} E_{y} \{ |x_j - \frac{1}{P} \sum_{i=1}^{P} \hat{x}_{ij}| |y \} \]  \hspace{1cm} (C.1)

\[ L_{SD,P}(\theta) = \sum_{j=1}^{N} E_{y} \{ \gamma_{P} \sum_{i=1}^{P} |\hat{x}_{ij} - \frac{1}{P} \sum_{k=1}^{P} \hat{x}_{kj}| |y \} \],  \hspace{1cm} (C.2)

where \( x_j \triangleq [x]_j \), \( \hat{x}_{ij} \triangleq [\hat{x}]_{ij} \), and

\[ \gamma_{P} \triangleq \sqrt{\frac{\pi P}{2(P-1)}}. \]  \hspace{1cm} (C.3)

To simplify the notation in the sequel, we will consider an arbitrary fixed value of \( j \) and use the abbreviations

\[ x_j \rightarrow X, \hspace{1cm} \hat{x}_{ij} \rightarrow \hat{X}_i. \]  \hspace{1cm} (C.4)

Recall that \( x \) and \( \{ \hat{x}_i \} \) are mutually independent when conditioned on \( y \) because the code vectors \( \{ z_i \} \) are generated independently of both \( x \) and \( y \). In the context of Proposition 3.1, we also assume that the vector elements \( x_j \) and \( \hat{x}_{ij} \) are independent Gaussian when conditioned on \( y \). This implies that we can make the notational shift

\[ p_{k|x}(x_j|y) \rightarrow N(X; \mu_0, \sigma_0^2), \hspace{1cm} p_{\hat{x}_i|x}(\hat{x}_{ij}|y) \rightarrow N(\hat{X}_i; \mu, \sigma^2), \]  \hspace{1cm} (C.5)

where \( X \) and \( \{ \hat{X}_i \} \) are mutually independent. With this simplified notation, we note that \([\hat{x}_{mmse}]_j \rightarrow \mu_0\), and that mode collapse corresponds to \( \sigma = 0 \).

Furthermore, if \( \theta \) can completely control the statistics of \( \hat{X}_i \), which are now parameterized by \( (\mu, \sigma) \), then (25) can be rewritten as

\[ (\mu_*, \sigma_*) = \arg \min_{\mu, \sigma} \{ L_{1,P}(\mu, \sigma) - \beta_{SD}L_{SD,P}(\mu, \sigma) \} \Rightarrow \begin{cases} \mu_* = \mu_0 \\ \sigma_* = \sigma_0 \end{cases} \]  \hspace{1cm} (C.6)

with

\[ L_{1,P}(\mu, \sigma) = E_{X,\hat{X}_1,...,\hat{X}_P} \{|X - \frac{1}{P} \sum_{i=1}^{P} \hat{X}_i|\} \]  \hspace{1cm} (C.7)

\[ L_{SD,P}(\mu, \sigma) = E_{\hat{X}_1,...,\hat{X}_P} \{ \gamma_{P} \sum_{i=1}^{P} |\hat{X}_i - \frac{1}{P} \sum_{k=1}^{P} \hat{X}_k| \}. \]  \hspace{1cm} (C.8)

Although \( \sigma_* \) must be positive, it turns out that we do not need to enforce this in the optimization because it will arise naturally.

To further analyze (C.7) and (C.8), we define

\[ \hat{\mu} \triangleq \frac{1}{P} \sum_{i=1}^{P} \hat{X}_i, \]  \hspace{1cm} (C.9)

\[ \hat{\sigma} \triangleq \frac{\gamma_{P}}{\sqrt{P}} \sum_{i=1}^{P} |\hat{X}_i - \hat{\mu}|. \]  \hspace{1cm} (C.10)

The quantity \( \hat{\mu} \) can be recognized as the unbiased estimate of the mean \( \mu \) of \( \hat{X}_i \), and we now show that \( \hat{\sigma} \) is an unbiased estimate of the SD \( \sigma \) of \( \hat{X}_i \), in the case of Gaussian \( \hat{X}_i \). To see this, first observe that the i.i.d. \( N(\mu, \sigma^2) \) property of \( \{ \hat{X}_i \} \) implies that \( \hat{X}_i - \hat{\mu} \sim (1 - \frac{1}{P}) \hat{X}_i - \frac{1}{P} \sum_{k \neq i} \hat{X}_k \) is Gaussian with mean zero and variance \( 1 - \frac{1}{P} \)^2 \sigma^2 + \frac{P-1}{P} \sigma^2 = \frac{P-1}{P} \sigma^2 \).

The variable \( |\hat{X}_i - \hat{\mu}| \) is thus half-normal distributed (Leone et al., 1961) with mean \( \sqrt{\frac{2(P-1)}{P}} \sigma^2 \). Because \( \{ \hat{X}_i \} \) are i.i.d., the variable \( \frac{1}{P} \sum_{i=1}^{P} |\hat{X}_i - \hat{\mu}| \) inherits the same mean. Finally, multiplying that variable by \( \gamma_{P} \) yields \( \hat{\sigma} \) from (C.10), and multiplying its mean using the expression for \( \gamma_{P} \) from (C.3) implies

\[ E(\hat{\sigma}) = \sigma, \]  \hspace{1cm} (C.11)

and so \( \hat{\sigma} \) is an unbiased estimator of \( \sigma \), the SD of \( \hat{X}_i \).
With the above definitions of $\bar{\mu}$ and $\bar{\sigma}$, the optimization cost in (C.6) can be written as

$$L_{1,P}(\mu, \sigma) - \beta_{SD} L_{SD,P}(\mu, \sigma) = E_{X, \hat{X}_1, \ldots, \hat{X}_P} \{ |X - \bar{\mu}| \} - \beta_{SD} E_{\hat{X}_1, \ldots, \hat{X}_P} \{ \bar{\sigma} \}$$  \hspace{1cm} (C.12)

$$= E_{X, \hat{X}_1, \ldots, \hat{X}_P} \{ |X - \bar{\mu}| \} - \beta_{SD} \sigma,$$  \hspace{1cm} (C.13)

where in the last step we exploited the unbiased property of $\bar{\sigma}$. To proceed further, we note that under the i.i.d. Gaussian assumption on $\{\hat{X}_i\}$ we get $\bar{\mu} \sim \mathcal{N}(\mu, \sigma^2/P)$, after which the mutual independence of $\{\hat{X}_i\}$ and $X$ yields

$$X - \bar{\mu} \sim \mathcal{N}(\mu_0 - \mu, \sigma_0^2 + \sigma^2/P).$$  \hspace{1cm} (C.14)

Taking the absolute value of a Gaussian random yields a folded-normal random variable (Leone et al., 1961). Using the mean and variance in (C.14), the expressions in (Leone et al., 1961) yield

$$E_{X, \hat{X}_1, \ldots, \hat{X}_P} \{ |X - \bar{\mu}| \} = \sqrt{\frac{2(\sigma_0^2 + \sigma^2/P)}{\pi}} \exp \left( - \frac{(\mu_0 - \mu)^2}{2(\sigma_0^2 + \sigma^2/P)} \right) + (\mu_0 - \mu) \text{erf} \left( \frac{\mu_0 - \mu}{\sqrt{2(\sigma_0^2 + \sigma^2/P)}} \right).$$  \hspace{1cm} (C.15)

Thus the optimization cost (C.13) can be written as

$$J(\mu, \sigma) \triangleq \sqrt{\frac{2(\sigma_0^2 + \sigma^2/P)}{\pi}} \exp \left( - \frac{(\mu_0 - \mu)^2}{2(\sigma_0^2 + \sigma^2/P)} \right) + (\mu_0 - \mu) \text{erf} \left( \frac{\mu_0 - \mu}{\sqrt{2(\sigma_0^2 + \sigma^2/P)}} \right) - \beta_{SD} \sigma.$$  \hspace{1cm} (C.16)

Since $J(\cdot, \cdot)$ is convex, the minimizer $(\mu_*, \sigma_*) = \arg \min_{\mu, \sigma} J(\mu, \sigma)$ satisfies $\nabla J(\mu_*, \sigma_*) = (0, 0)$. To streamline the derivation, we define

$$c \triangleq \sqrt{\frac{2(\sigma_0^2 + \sigma^2/P)}{\pi}}, \hspace{1cm} s \triangleq \sqrt{\frac{\sigma_0^2 + \sigma^2}{P}}$$  \hspace{1cm} (C.17)

so that

$$J(\mu, \sigma) = c \exp \left( - \frac{(\mu - \mu_0)^2}{2s^2} \right) + (\mu_0 - \mu) \text{erf} \left( \frac{\mu_0 - \mu}{\sqrt{2s^2}} \right) - \beta_{SD} \sigma.$$  \hspace{1cm} (C.18)

Because $c$ and $s$ are invariant to $\mu$, we get

$$\frac{\partial J(\mu, \sigma)}{\partial \mu} = -c \exp \left( - \frac{(\mu - \mu_0)^2}{2s^2} \right) \frac{\mu - \mu_0}{s^2} + \text{erf} \left( \frac{\mu_0 - \mu}{\sqrt{2s^2}} \right) + (\mu_0 - \mu) \frac{2}{\sqrt{\pi}} \exp \left( - \frac{(\mu - \mu_0)^2}{2s^2} \right),$$  \hspace{1cm} (C.19)

which equals zero if and only if $\mu = \mu_0$. Thus we have determined that $\mu_* = \mu_0$. Plugging $\mu_* = \mu_0$ into (C.16), we find

$$J(\mu_*, \sigma) = \sqrt{2(\sigma_0^2 + \sigma^2/P)} - \beta_{SD} \sigma.$$  \hspace{1cm} (C.20)

Taking the derivative with respect to $\sigma$, we get

$$\frac{\partial J(\mu_*, \sigma)}{\partial \sigma} = \frac{2}{\pi P(\sigma_0^2/\sigma^2 + 1)} - \beta_{SD}$$  \hspace{1cm} (C.21)

$$= \sqrt{\frac{2}{\pi P(\sigma_0^2/\sigma^2 + 1)}} - \sqrt{\frac{2}{\pi P(P + 1)}},$$  \hspace{1cm} (C.22)

where in the last step we applied the value of $\beta_{SD}$ from (24). It can now be seen that $\frac{\partial J(\mu_*, \sigma)}{\partial \sigma} = 0$ if and only if $\sigma = \sigma_0$, which implies that $\sigma_* = \sigma_0$. Thus we have established (C.6), which completes the proof of Proposition 3.1.

**D. Proof of Proposition 3.2**

Here we prove Proposition 3.2. Recall from (11) that $\hat{x}_{mmse} \triangleq E\{x|y\}$. Let's define

$$\hat{e}_i \triangleq \hat{x}_i - \hat{x}_{mmse}$$  \hspace{1cm} (D.1)

$$e \triangleq x - \hat{x}_{mmse}$$  \hspace{1cm} (D.2)
Then from (26),
\[ E_P = E\{\|\hat{x}_{(p)} - x\|^2|y\} \]
\[ = E\{\frac{1}{P} \sum_{i=1}^{P} \|\hat{x}_i - x\|^2|y\} \] (D.3)
\[ = E\{\frac{1}{P} \sum_{i=1}^{P} \|\hat{x}_i - x\|^2|y\} \] (D.4)
\[ = \frac{1}{P^2} E\{\sum_{i=1}^{P} (\hat{x}_i - \hat{x}_{\text{mmse}} + \hat{x}_{\text{mmse}} - x)\|^2|y\} \] (D.5)
\[ = \frac{1}{P^2} E\{\sum_{i=1}^{P} (\hat{e}_i - e)\|^2|y\} \] (D.6)
\[ = \frac{1}{P^2} E\{\|\hat{e}_i - e\|_2^2 \sum_{j=1}^{P} (\hat{e}_j - e)|y\} \] (D.7)
\[ = \frac{1}{P^2} \sum_{i=1}^{P} \sum_{j=1}^{P} E\{((\hat{e}_i - e)^\top (\hat{e}_j - e))|y\} \] (D.8)
\[ = \frac{1}{P^2} \sum_{i=1}^{P} \sum_{j=1}^{P} E\{((\hat{e}_i - e)^\top (\hat{e}_j - e))|y\} \] (D.9)
\[ = \frac{1}{P^2} \sum_{i=1}^{P} E\{((\hat{e}_i - e)^\top (\hat{e}_i - e))|y\} + \frac{1}{P^2} \sum_{i=1}^{P} \sum_{j\neq i} E\{((\hat{e}_i - e)^\top (\hat{e}_j - e))|y\} \] (D.10)
\[ = \frac{1}{P^2} \sum_{i=1}^{P} \left[ E\{\|\hat{e}_i\|^2|y\} - 2E((\hat{e}_i)^\top e|y) + E\{\|e\|^2|y\} \right] \] (D.11)
\[ + \frac{1}{P^2} \sum_{i=1}^{P} \sum_{j\neq i} \left[ E\{(\hat{e}_i)^\top \hat{e}_j|y\} - E((\hat{e}_i)^\top e|y) - E(e^\top \hat{e}_j|y) + E\{\|e\|^2|y\} \right] \] (D.12)

where certain terms vanished because the i.i.d. and zero-mean properties of \( \{e, \hat{e}_1, \ldots, \hat{e}_P\} \) imply
\[ E\{(\hat{e}_i)^\top \hat{e}_j|y\} = E\{(\hat{e}_i)^\top y\} E\{(\hat{e}_j)^\top y\} = 0 \] (D.13)
\[ E\{(\hat{e}_i)^\top e|y\} = E\{(\hat{e}_i)^\top y\} E\{e|y\} = 0 \] (D.14)
\[ E\{e^\top \hat{e}_j|y\} = E\{e^\top y\} E\{\hat{e}_j|y\} = 0. \] (D.15)

Finally, note that \( E\{\|e\|^2|y\} = \mathcal{E}_{\text{mmse}} \) from (11) and (D.2). Furthermore, because \( \{x, x_1, \ldots, x_P\} \) are independent samples of \( p_{x|y}(\cdot|y) \) under the assumptions of Proposition 3.2, we have \( E\{\|e\|^2|y\} = E\{\|\hat{e}_i\|^2|y\} \) and so (D.12) becomes
\[ E_P = \frac{1}{P^2} \sum_{i=1}^{P} \mathcal{E}_{\text{mmse}} + \frac{1}{P} \mathcal{E}_{\text{mmse}} + \frac{P(P-1)}{P^2} \mathcal{E}_{\text{mmse}} = \frac{P + 1}{P} \mathcal{E}_{\text{mmse}}. \] (D.16)

This result holds for any \( P \geq 1 \), which implies the ratio
\[ \frac{\mathcal{E}_1}{E_P} = \frac{2P}{P+1}. \] (D.17)

**E. CFID implementation details**

Recall the CFID definition from (19):
\[ \text{CFID} \triangleq E_x \{\|\mu_{x|y} - \mu_{x|y}'\|^2 + \text{tr}\left[ (\Sigma_{x|x} - \Sigma_{x'|x}) - 2(\Sigma_{x'|x}^{-1} (\Sigma_{x'|x}^{-1} - \Sigma_{x'|x})\right] \}, \] (E.1)

The values used to compute (E.1) are (Soloveitchik et al., 2021)
\[ \mu_{x|y} = \mu_x + \Sigma_x \Sigma_{y|x}^{-1} (y - \mu_y) \] (E.2)
\[ \Sigma_{x|x} = \Sigma_x - \Sigma_x \Sigma_{y|x}^{-1} \Sigma_{y'|x} \] (E.3)
\[ \mu_{x'|y} = \mu_x + \Sigma_x \Sigma_{y'|x}^{-1} (y - \mu_y) \] (E.4)
\[ \Sigma_{x'|x} = \Sigma_{x|y} - \Sigma_{x'|y} \Sigma_{y'|x}^{-1} \Sigma_{y}'. \] (E.5)

Plugging the above values into (E.1), the CFID can be written as in Lemma 2 of (Soloveitchik et al., 2021):
\[ \text{CFID} = \|\mu_x - \mu_{\hat{x}}\|^2 + \text{tr}\left[ (\Sigma_{x|x} - \Sigma_{x|x}) - 2(\Sigma_{x|x}^{-1} - \Sigma_{x'|x}^{-1})\right] + \text{tr}\left[ (\Sigma_{x'|x} + \Sigma_{x'|x}^{-1} - 2(\Sigma_{x'|x}^{-1} - \Sigma_{x'|x}^{-1})), \right] \] (E.6)
where $\Sigma_{yy}^{-1}$ is typically implemented using a pseudo-inverse.

In practice, when computing CFID, we have a set of test data, $\{(x_t, y_t)\}_{t=1}^{P}$ and, for each $y_t$, we have a set of generated test samples $\{\hat{x}_t\}_{i=1}^{n_t}$. We repeat $x_t$ and $y_t$ $P$ times, which yields a new set of $P$ tuples, $\{(x_{ti}, y_{ti}, \hat{x}_{ti})\}_{i=1}^{P}$ for $t = 1 \ldots n$. We embed all $x_{ti}$, $y_{ti}$, and $\hat{x}_{ti}$ using a feature-generating network, yielding embedding matrices $X, Y$, and $\hat{X}$ of dimension $Pn \times d$, respectively. We used the VGG-16 network (Simonyan & Zisserman, 2014) for our MRI experiments, since (Kastryulin et al., 2022) found that it gave results that correlated much better with radiologists’ perceptions, while we used the standard Inception-v3 network (Szegedy et al., 2016) for our inpainting experiments. The embeddings are then used to compute the sample mean values

$$\mu_{\hat{x}} \triangleq \frac{1}{Pn} 1^T X, \quad \mu_{\hat{y}} \triangleq \frac{1}{Pn} 1^T Y, \quad \mu_{\hat{X}} \triangleq \frac{1}{Pn} 1^T \hat{X},$$  

(E.7)

We then subtract the sample mean from each row of $X, Y$, and $\hat{X}$ to give the zero-mean embedding matrices $X_m \triangleq X - 1\mu_{\hat{X}}, Y_m \triangleq Y - 1\mu_{\hat{y}},$ and $\hat{X}_m \triangleq \hat{X} - 1\mu_{\hat{X}}$, Then, $X_{zm}, Y_{zm},$ and $\hat{X}_{zm}$ are used to compute the sample auto-covariance matrices

$$\Sigma_{xx} \triangleq \frac{1}{Pn} X_m X_m^T, \quad \Sigma_{yy} \triangleq \frac{1}{Pn} Y_m Y_m^T, \quad \Sigma_{\hat{X}\hat{X}} \triangleq \frac{1}{Pn} \hat{X}_m \hat{X}_m^T,$$  

(E.8)

and the sample cross-covariance matrices

$$\Sigma_{xy} \triangleq \frac{1}{Pn} X_m Y_m^T, \quad \Sigma_{\hat{X}\hat{y}} \triangleq \frac{1}{Pn} \hat{X}_m \hat{y}_m^T.$$  

(E.9)

We plug the sample statistics from (E.8) and (E.9) into (E.3) and (E.5) to compute the conditional covariance matrices, which are then used in conjunction with the values from (E.7), (E.8), and (E.9) to compute the CFID as written in (E.6). In (Soloveitchik et al., 2021), the authors use $P = 1$ in all of their experiments. To be consistent with how we evaluated the other metrics, we use $P = 32$ unless otherwise noted.

### F. MR imaging details

We now present some details about the MR imaging linear inverse problem. Suppose that the goal is to recover the $N$-pixel MR image $i \in \mathbb{C}^N$ from the multicoil measurements $\{k_c\}_{c=1}^C$, where (Pruessmann et al., 1999)

$$k_c = MF S_c i + n_c.$$  

(F.1)

In (F.1), $C$ refers to the number of coils, $k_c \in \mathbb{C}^M$ are the measurements from the $c$th coil, $M \in \mathbb{R}^{M \times N}$ is a sub-sampling operator containing rows from $I_N$—the $N \times N$ identity matrix, $F \in \mathbb{C}^{N \times N}$ is the unitary 2D discrete Fourier transform, $S_c \in \mathbb{C}^{N \times N}$ is a diagonal matrix containing the sensitivity map of the $c$th coil, and $n_c \in \mathbb{C}^M$ is noise. From (F.1), it can be seen that the MR measurements are collected in the spatial Fourier domain, otherwise known as the “k-space.” We assume that the sensitivity maps $\{S_c\}$ are estimated from $\{k_c\}$ using SPIRIT (Uecker et al., 2014), which yields maps with the property $\sum_{c=1}^C S_c^H S_c = I_N$. The ratio $R \triangleq \frac{N}{M}$ is known as the acceleration rate.

There are different ways that one could apply the generative posterior sampling framework to multicoil MR image recovery. One is to configure the generator to produce posterior samples $\hat{i}$ of the complex image $i$. Another is to configure the generator to produce posterior samples $\hat{x}$ of the stack $x \triangleq [x_1, \ldots, x_c]^T$ of “coil images” $x_c \triangleq S_c i$ and later coil-combining them to yield a complex image estimate $\hat{i} \triangleq [S_1^H, \ldots, S_C^H] \hat{x}$. We take the latter approach. Furthermore, rather than feeding our generator with k-space measurements $k_c$, we choose to feed it with aliased coil images $y_c \triangleq F^H M^T k_c$. Writing (F.1) in terms of the coil images, we obtain

$$y_c = F^H M^T M F x_c + w_c,$$  

(F.2)

where $w_c \triangleq F^H M^T n_c$. Then we can stack $\{y_c\}$ and $\{w_c\}$ column-wise into vectors $y$ and $w$, and set $A = I_C \otimes F^H M^T M F \in \mathbb{C}^{NC \times NC}$, to obtain the formulation in (1).

To train our generator, we assume to have access to paired training examples $\{(x_t, y_t)\}$, where $x_t$ is a stack of coil images and $y_t$ is the corresponding stack of k-space coil measurements. The fastMRI multicoil dataset (Zhontar et al., 2018) provides $\{(x_t, k_t)\}$, from which we can easily obtain $\{(x_t, y_t)\}$.

To implement the data-consistency method in (32), we note that $A = I_C \otimes F^H M^T M F$ is an orthogonal projection matrix, and so $I - A^+ A = I - A = I \otimes F^H (I - M^T M) F$. 

A Regularized Conditional GAN for Posterior Sampling
Table G.1. The mean and covariance components of CFID, along with the total CFID, for the MRI and inpainting experiments. For the MRI experiment, \( \text{CFID}^1 \) used 72 test samples and \( P = 32 \), \( \text{CFID}^2 \) used 2 376 test samples and \( P = 8 \), and \( \text{CFID}^3 \) used all 14,576 samples and \( P = 1 \). For the inpainting experiment, \( \text{CFID}^1 \) used 1 000 test images and \( P = 32 \), \( \text{CFID}^2 \) used 3 000 test and validation images and \( P = 8 \), and \( \text{CFID}^3 \) used all 30,000 images and \( P = 1 \).

| Experiment | Model                          | \( \text{CFID}_{\text{mean}} \) | \( \text{CFID}_{\text{cov}} \) | \( \text{CFID}^1 \) | \( \text{CFID}_{\text{mean}} \) | \( \text{CFID}_{\text{cov}} \) | \( \text{CFID}^2 \) | \( \text{CFID}_{\text{mean}} \) | \( \text{CFID}_{\text{cov}} \) | \( \text{CFID}^3 \) |
|------------|--------------------------------|-------------------------------|-------------------------------|-------------------|-------------------------------|-------------------------------|-------------------|-------------------------------|-------------------------------|-------------------|
| MRI        | Langevin (Jalal et al., 2021a) | 1.89                          | 3.40                          | 3.29              | -                             | -                             | -                 | -                             | -                             | -                 |
|            | cGAN (Adler & Öktem, 2018)    | 3.12                          | 3.27                          | 6.39              | 2.79                          | 1.48                          | 4.27              | 2.71                          | 1.10                          | 3.82              |
|            | cGAN (Ohayon et al., 2021a)   | 1.94                          | 2.12                          | 4.06              | 2.27                          | 1.00                          | 3.27              | 2.29                          | 0.66                          | 2.95              |
|            | Ours                          | 0.98                          | 2.12                          | 3.10              | 0.86                          | 0.68                          | 1.54              | 0.86                          | 0.43                          | 1.29              |
| Inpainting | Score SDE (Song et al., 2021) | 0.97                          | 38.69                         | 39.66             | -                             | -                             | -                 | 0.90                          | 4.21                          | 5.11              |
|            | CoModGAN (Zhao et al., 2021a) | 0.42                          | 41.21                         | 41.63             | 0.35                          | 25.39                         | 25.74             | 0.32                          | 4.98                          | 5.29              |
|            | Ours                          | 0.32                          | 39.41                         | 39.73             | 0.25                          | 22.32                         | 22.58             | 0.24                          | 4.45                          | 4.69              |

G. Additional results

G.1. CFID decomposition into mean and covariance components

We can write the CFID from (19) as a sum of two terms: a term that quantifies the conditional-mean error and a term that quantifies the conditional-covariance error:

\[
\text{CFID} = \text{CFID}_{\text{mean}} + \text{CFID}_{\text{cov}} \tag{G.1}
\]

\[
\text{CFID}_{\text{mean}} \triangleq \mathbb{E}_y \{ \| \hat{\mu}_y - \mu_y \|_2^2 \} \tag{G.2}
\]

\[
\text{CFID}_{\text{cov}} \triangleq \text{tr} \left[ \Sigma_{x|y} + \Sigma_{y|x} - 2 \left( \Sigma_{y|x} \Sigma_{y|x} \Sigma_{y|x} \right)^{1/2} \right]. \tag{G.3}
\]

In Table G.1, we report \( \text{CFID}_{\text{mean}} \) and \( \text{CFID}_{\text{cov}} \) for the MRI and inpainting experiments, in addition to the total CFID (also shown in Table 1 and Table 2). As before, we computed CFID on three data subsets for each experiment, motivated by the fact that the CFID bias decreases as the number of data samples increases (Soloveitchik et al., 2021). Importantly, due to the slow sample-generation time of the two score-based methods (Jalal et al., 2021a; Song et al., 2021), we did not have the computational resources to evaluate them on all six datasets considered in the table, and that is why certain entries of Table G.1 (and of Table 1) are missing.

For the MRI experiment, Table G.1 shows that our proposed method outperformed the competing methods in both the mean and covariance components of CFID (and thus the total CFID) in all cases.

For the inpainting experiment, Table G.1 shows that our proposed method outperformed CoModGAN in both the mean and covariance components (and thus the total CFID) for all three evaluation datasets. Table G.1 also shows that our proposed method outperformed Song et al.’s SDE in the mean component, but was outperformed in the covariance component, on both datasets for which Song et al.’s SDE was evaluated. Finally, our method gave the best total CFID on the 30,000-sample dataset, while Song et al.’s SDE gave the best total CFID on the 1,000-sample dataset.

When CFID is evaluated on small datasets, the results are known to be biased towards larger values (Soloveitchik et al., 2021). Consequently, the results in the last three columns of Table G.1 are the least biased. In particular, Table G.1 shows that the CFID bias is mainly concentrated in the variance component. On the 1,000-sample dataset, our technique outperformed Song et al.’s SDE by \( 3 \times \) in the mean component but was outperformed by \( 1.02 \times \) in the covariance component. In this case, Song et al.’s SDE outperformed our technique in total CFID by \( 1.002 \times \) because the covariance components were so inflated due to bias. On the 30,000-sample dataset, our technique outperformed Song et al.’s SDE by \( 3.75 \times \) in the mean component but was outperformed by \( 1.06 \times \) in the covariance component. In this case, our technique outperformed Song et al.’s SDE in total CFID by \( 1.09 \times \) because the covariance components were much less biased.

H. Network implementation details

H.1. MRI

**Generator architecture:** For our MRI experiments, we take inspiration from the U-Net architecture (Ronneberger et al., 2015), using it as the basis for our generator. The primary input, \( y \) is concatenated with the code vector \( z \) and fed through the U-Net. The network consists of 4 pooling layers with 128 initial channels. However, instead of pooling, we opt to use convolutions with kernels of size \( 3 \times 3 \), “same” padding, and a stride of 2 when downsampling. Likewise, we upsample
using transpose convolutions, again with kernels of size $3 \times 3$, “same” padding, and a stride of 2. All other convolutions utilize kernels of size $3 \times 3$, “same” padding, and a stride of 1.

Within each encoder and decoder layer we include a residual block, the architecture of which can be found in (Adler & Öktem, 2018). We use instance-norm for all normalization layers and parametric ReLUs as our activation functions, in which the network learns the optimal “negative slope.” Finally, we include 5 residual blocks at the base of the U-Net, in between the encoder and decoder. This is done in an effort to artificially increase the depth of the network and is inspired by (Deora et al., 2020). Our generator has 86,734,334 trainable parameters.

**Discriminator architecture:** Our discriminator is a standard CNN with 5 layers. In the first 3 layers, we use convolutions with kernels of size $4 \times 4$, “same” padding, and a stride of 2 to reduce the image resolution. The remaining two convolutional layers use the same parameters, with the stride modified to be 1. We use batch-norm as our normalization layer and leaky ReLUs with a “negative-slope” of 0.2 as our activation functions.

The final convolutional layer does not have a normalization layer or activation function, and outputs a 1 channel “prediction map.” This prediction map gives a Wasserstein score for a patch of the image. Patch-based discrimination has been known to improve the high-frequency information in reconstructions (Isola et al., 2017). Our discriminator has 693,057 trainable parameters.

cGANs: For all 3 cGANs, we use the architectures described above and a similar training and testing procedure. We adapt the model’s regularization and $\beta_{adv}$ to match the authors’ original implementation. In particular, when training Ohayon et al.’s cGAN we set $\beta_{adv} = 1e^{-3}$ and use $L_{2,1}$ regularization while training the generator. When training Adler et al.’s cGAN, we set $\beta_{adv} = 1$ and do not apply any regularization to the generator. Instead we modify the number of input channels to the discriminator and slightly modify the training logic to be consistent with the loss proposed in (6).

**Langevin:** For Jalal et al.’s MRI approach (Jalal et al., 2021a), we do not modify the original implementation from (Jalal et al., 2021b) other than replacing the default sampling pattern with the GRO undersampling mask. We generated 32 samples for 72 different test images using a batch-size of 4, which took roughly 6 days. These samples were generated on a server with 4 NVIDIA V100 GPUs, each with 32 GB of memory. We used 4 samples per batch (and recorded the time to generate 4 samples in Table 1) because the code from (Jalal et al., 2021b) is written to generate one sample per GPU.

**H.2. Inpainting**

**CoModGAN:** We use the PyTorch implementation of CoModGAN from (Zeng, 2022) and train the model to inpaint a $128 \times 128$ centered square of $256 \times 256$ celebA-HQ images. The total training time on a server with 4 NVIDIA A100 GPUs, each with 82 GB of memory, was roughly 2 days.

**Score-based SDE:** For the inpainting experiment in Section 4.2, we compare against Song et al.’s more recent SDE technique (Song et al., 2021), for which we use the publicly available pretrained weights and suggested settings for the $256 \times 256$ celebA-HQ dataset and the code from the official PyTorch implementation (Song et al., 2022b). We generate 32 samples for all 1,000 images in our test set, using a batch-size of 20 and generating 32 samples for each batch element concurrently. The total running time on a server with 4 NVIDIA A100 GPUs, each with 82 GB of memory, is roughly 9 days.
I. Additional reconstruction plots

I.1. MRI Reconstruction

**Figure I.1.** Example posterior samples of a zoomed region. The first column for each method contains five samples, and the second contains the corresponding error map. The purpose of the error map is to better identify variation across samples.
Figure I.2. Example posterior samples of a zoomed region. Visible variation in an anatomical feature is highlighted by yellow arrows.
Figure 1.3. MRI reconstruction of two test images. Column one: reconstruction $\hat{x}(\rho)$, two: pixel-wise absolute error $|\hat{x}(\rho) - x|$, three: pixel-wise SD $(\frac{1}{N} \sum_{i=1}^{N} (\hat{x}_i - \hat{x}(\rho))^2)^{1/2}$. 
Figure I.4. MRI reconstruction of two test images. Column one: reconstruction $\hat{x}(\rho)$, two: pixel-wise absolute error $|\hat{x}(\rho) - x|$, three: pixel-wise SD $(\frac{1}{P} \sum_{i=1}^{P} (\hat{x}_i - \hat{x}(\rho))^2)^{1/2}$.
Figure 1.5. MRI reconstruction of two test images. Column one: reconstruction $\hat{x}(\rho)$, two: pixel-wise absolute error $|\hat{x}(\rho) - x|$, three: pixel-wise SD $(\frac{1}{P} \sum_{i=1}^{P} (\hat{x}_i - \hat{x}(\rho))^2)^{1/2}$. 
I.2. Inpainting

Figure I.6. Two examples of inpainting a $128 \times 128$ centered square on an $256 \times 256$ resolution celebA-HQ test image.
Figure 1.7. Two examples of inpainting a $128 \times 128$ centered square on an $256 \times 256$ resolution celebA-HQ test image.
Figure I.8. Two examples of inpainting a $128 \times 128$ centered square on an $256 \times 256$ resolution celebA-HQ test image.
Figure I.9. Two examples of inpainting a $128 \times 128$ centered square on a $256 \times 256$ resolution celebA-HQ test image.