A. Implementation details

A.1. Architecture

Our GAN architecture is composed of an RRDB based generator and a ResNet critic. The generator largely follows the architecture proposed in Real-ESRGAN [54, 55], where we use a pixel-unshuffle block to rearrange a $m \times n \times 3$ input into a $m/2 \times n/2 \times 12$ input to reduce the computational complexity of the following 23 RRDB blocks, followed by an upscale of the result to restore the original input shape. One major difference from [54] is the injection of noise channels along the network – Each "Noise Injection" block concatenates a new channel filled with Gaussian noise, which is used by the network to generate stochastic details (this is Z, mentioned in the loss formulation). In cases supporting multiple QFs we implement a FiLM [40] block that modulates the results of an RRDB or upsampling block using a learned affine transformation, conditioned on the quantization table that is embedded in the JPEG file. We present the generator architecture in Figure 5. The critic is a plain ResNet34 [20].

A.2. Training

We train our models using the Adam [27] optimizer with batch-size of 32 to alternately update the generator and the critic networks, using Eq. 10 and Eq. 9. We use exponential moving average on the generator weights with decay factor of 0.999.

FFHQ-128: The learning rate starts at 1×10^{-4} and annealed to 1×10^{-7} after $400,000$ steps using cosine annealing. We scale $V(D_\omega, G_\theta)$ in Eq. 10 by 1×10^{-3} and set $\lambda_{R_1} = 1$, $\lambda_{FM} = 1$ and λ_C according to the version reported in Figure 2. For the versions denoted as **Ours** and **Ours-P** we use cosine annealing of λ_C from 1×10^{-1} to 1×10^{1}.

General-Content: The learning rate starts at 1×10^{-4} and annealed to 1×10^{-6} after $400,000$ steps using cosine annealing. The weights are initialized from a model trained for $50,000$ steps as a regression model using an MSE loss, as this was found to be important for stabilizing the training. We scale $V(D_\omega, G_\theta)$ in Eq. 10 by 1×10^{-3} and set $\lambda_{R_1} = 1$, $\lambda_{FM} = 1$, $\lambda_P = 1 \times 10^{-2}$, $\lambda_{SM} = 1 \times 10^{3}$ and use cosine annealing of λ_C from 1×10^{0} to 1×10^{3}.

To estimate the mean and variance of generated images for use in $FM(G_\theta)$ and $SM(G_\theta)$ we generate 16 different restorations (using different Zs) for each of the first 8 images in a batch. In order to save train time we preform this every 8 iterations as we found negligible performance differences compared to performing this each iteration.

As the reference MMSE estimator for $SM(G_\theta)$ we use a regression model with the same architecture, trained for $650,000$ steps using a simple MSE loss.

As training data, we extract square patches with random scale (between 128×128 and the image resolution) from the training set at random position and rescale them to 128×128 pixels. This is inspired by [9] and it exposes our GAN to more diverse set of patches.

B. MMSE estimator is consistent

Theorem 1. *Let $\bar{X}(Y)$ be an MMSE estimator for JPEG artifact removal, i.e $\bar{X}(Y) = \mathbb{E}[X|Y]$. Then $\bar{X}(Y)$ is necessarily perfectly consistent with the compressed input Y.*

Proof. Denote by D the matrix that performs block-wise 2D-DCT and elementwise division by the matrix Q. Then $\bar{X}(Y)$ is consistent with Y iff $\left\|D\bar{X}(Y) - DY\right\|_\infty \leq \frac{1}{2}$. And indeed,

$$\begin{aligned}\left\|D\bar{X}(Y) - DY\right\|_\infty &= \|D\mathbb{E}[X|Y] - DY\|_\infty \\ &= \|\mathbb{E}[DX - DY|Y]\|_\infty \\ &\leq \mathbb{E}[\|DX - DY\|_\infty |Y] \\ &\leq \mathbb{E}\left[\frac{1}{2}|Y\right] = \frac{1}{2},\end{aligned}$$

where we used the triangle inequality and the fact that at any point where $p(X|Y) > 0$, the maximal difference in the DCT domain, before rounding, is $\frac{1}{2}$. □

C. Second-moment penalty

The **per-pixel** conditional variance $\sigma^2_{X|Y}$ of a random-variable X can be estimated from samples $\{x_i\}_{i=1}^n$ sampled from $p_{X|Y}$:

$$\sigma^2_{X|Y} \approx \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{X|Y})^2, \tag{12}$$

where $\mu_{X|Y}$ is the mean of X conditioned on Y. In practice we do not have access to this mean, hence we can approximate it either using an MMSE estimator $\bar{X}(Y)$ (that at optimality becomes the conditional mean) or using the sample mean:

$$\tilde{\mu}_{X|Y} = \frac{1}{n}\sum_{i=1}^{n} x_i. \tag{13}$$

In our case, we have two different variables for which we would like to compare their variances – the ground-truth images X and our reconstructed images $\hat{X}$, both conditioned on the compressed images Y.

Recall that for a fixed Y we have only a single ground-truth sample X and thus we cannot compute a useful sample conditional mean, hence we opt to use a pre-trained regression model as our MMSE estimator $\bar{X}(Y)$:

$$\tilde{\sigma}^2_{X|Y} = (x - \bar{X}(Y))^2. \tag{14}$$

Intuitively, $\bar{X}(Y)$ averages all possible values of each pixel in the reconstructed image based on the probability of the

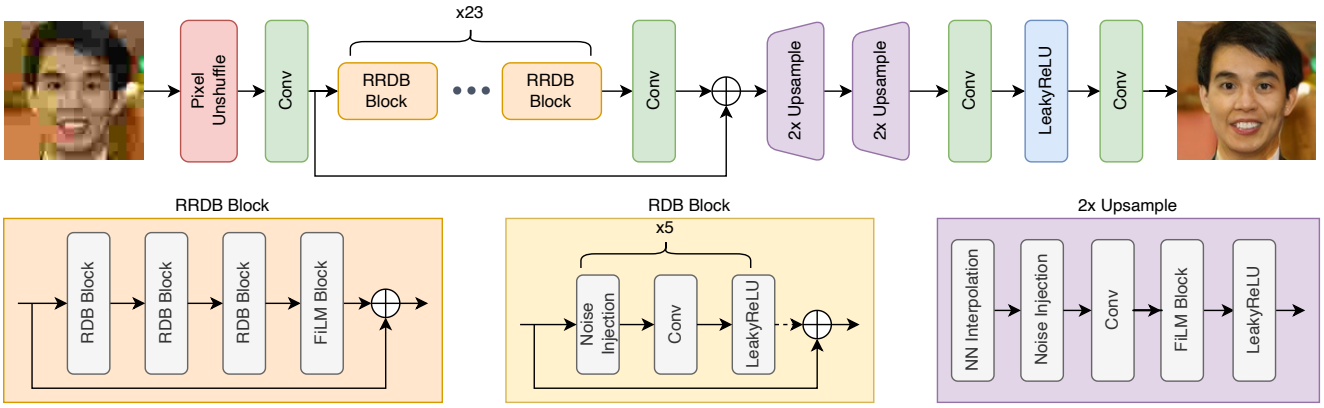

Figure 5. Our generator's architecture is based on [54], where Residual-in-Residual Dense Blocks (RRDBs) are used to restore a degraded input. Due to the fact that we work on full resolution inputs, we trade spatial resolution with depth using a pixel-unshuffle block that rearrange the input from $m \times n \times 3$ tensor into a $m/2 \times n/2 \times 12$ tensor and in the end we upsample the resulted tensor. To achieve stochasticity, we inject a noise channel to the features at the beginning of every RDB block and in the upsampling blocks. To be agnostic to the QF of the input image, we condition the RRDB and upsampling blocks on the quantization table used to create the image using a FiLM [40] block.

value. Hence, pixels with similar values in $\bar{X}(Y)$ and in x are pixels with small ambiguity – all of the possible reconstructions agree on the pixel's value, and thus we can expect small variance at such locations. On the other hand, pixels with large discrepancy between $\bar{X}(Y)$ and x are not necessarily an indication for large variance as the value in x might be rare and thus far from the mean. To better analyze the latter case we need further assumption on the conditional probability $X|Y$.

As $p_{\hat{X}|Y}$ changes during training, we cannot use a pre-trained regression model that was trained on $p_{X|Y}$, but we can generate as much samples as needed, hence we can compute a sample conditional mean that approximates well the true mean:

$$\tilde{\sigma}^2_{\hat{X}|Y} = \frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - \tilde{\mu}_{\hat{X}|Y})^2, \tag{15}$$

To create the penalty mentioned in Subsection 4.2 we just compute the distance between those two, per-pixel, variance approximations over a batch of realizations of Y:

$$SM(\hat{X}) = \lambda_{SM}\mathbb{E}_{X,\hat{X},Y}\left[\left\|\tilde{\sigma}^2_{\hat{X}|Y} - \tilde{\sigma}^2_{\hat{X}|Y}\right\|\right]. \tag{16}$$

In Figure 6 we present a visual illustration of the variance estimator $\tilde{\sigma}^2_{\hat{X}|Y}$ using QGAC as our MMSE estimator alongside per-pixel sample variance of Ours' and Bahat *et al.*'s methods.

D. JPEG numerical errors

As mentioned in Subsection 5.1 and Table 4, numerical approximations in the JPEG algorithm result in near, but not perfect, consistent results. To showcase that this phenomena is not unique to the differentiable JPEG implementations used by us and [5] we test *libjpeg-9d* [1], a standard JPEG implementation used in packages such as OpenCV [7] and PIL [12]. We compress the FFHQ test set with QF=100 which corresponds to no quantization, at block size of 1 which means the DCT values equal the color values, and without chroma sub-sampling, meaning we should expect a perfect reconstruction of the inputed images theoretically. Table 2 present the RMSE between the ground-truth images and the compressed images, using different configurations of the encoder-decoder. As it can be clearly seen, as long as we do not skip the YCbCr color-space conversion, true lossless compression is not achieved due to numerical approximations. Table 3 presents the actual Consistency RMSE results of the projected methods. We can see that while all the methods are not perfectly consistent, they are extremely quite to zero and perform better than *libjpeg-9d* when operating at the YCbCr color space.

To reproduce the *libjpeg-9d* images, use the following snippet:

```
# 2D-DCT=float, Color-Space=YCbCr, Block-Size=1
cjpeg -block 1 -quality 100 -sample 1x1,1x1,1x1 -
    dct float <input_image> | djpeg -dct float >
    <output_image>

# 2D-DCT=float, Color-Space=RGB, Block-Size=1
cjpeg -block 1 -quality 100 -sample 1x1,1x1,1x1 -
    dct float -rgb <input_image> | djpeg -dct
    float > <output_image>
```

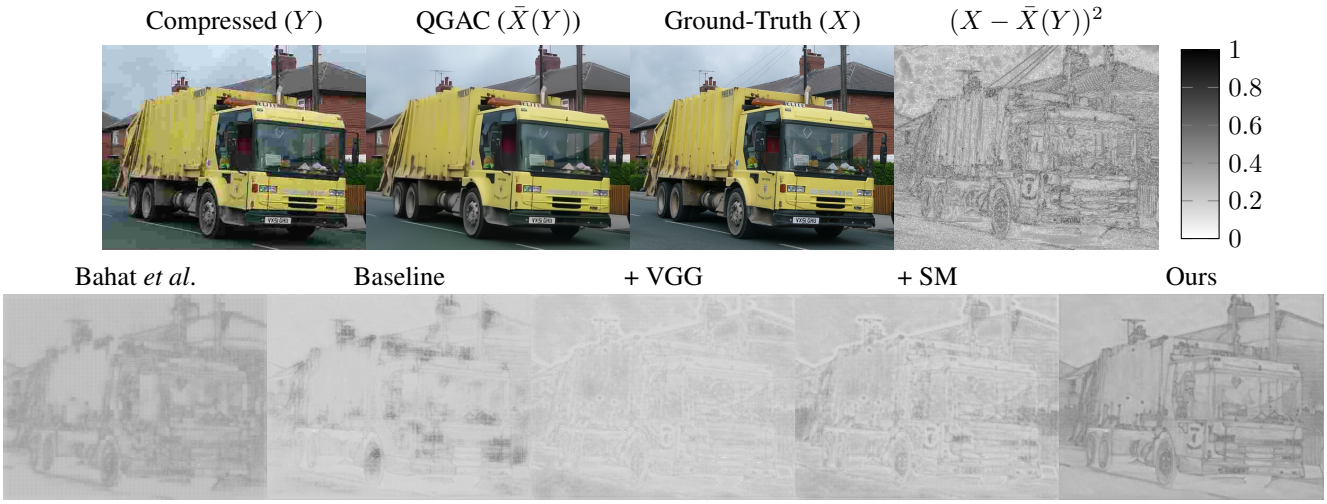

Figure 6. We approximate the conditional variance of clean images on a compressed input (Y) using a single ground-truth image (X) and an MMSE estimator ($\bar{X}(Y)$). Here we use QGAC [15] as our MMSE estimator on a JPEG compressed image with QF=10. On the bottom row we present per-pixel variance maps calculated on 64 samples of different methods ($\hat{X}$). For visualization purposes we use the 8th root of the variance and add a color bar (white and black correspond to low and high variance, respectively). The variance estimation $(X-\bar{X}(Y))^2$ indicates that we should expect more variance in regions with sharp transitions and this is indeed what we see in the variance maps of the different methods. Our method produce more variance compared to Bahat *et al.*'s method in regions with sharp transition and less variance in smooth regions. This is also supported quantitatively in Table 1 where we see larger average per-pixel standard deviation while achieving better FID.

Table 2. RMSE between clean FFHQ test images and compressed-decompressed FFHQ test images using the *libjpeg-9d* library. We use no chroma sub-sampling and quantization table of ones, hence, the process should be invertible. In practice, we see deviations due to numerical approximations.

2D-DCT	Color-Space	Block-Size	RMSE
float	YCbCr	1	0.5465
float	RGB	1	0

Table 3. Consistency results of different constrained methods on the FFHQ test dataset at QF=5. The inconsistencies stem from numerical approximations.

Method	Consistency
Bahat	0.0867
λ_C=0-P	0.9466
Ours-P	0.1664
Ground Truth (theoretical)	0

E. More results

E.1. FFHQ

In Table 4 we present quantitative results of the different methods on the FFHQ test set at QF=5.

In Figure 7 we visually demonstrate the perceptual quality and the stochastic variation of our method by presenting several realizations of a given input and comparing them to the other methods. As expected, the regression model generates overly-smoothed results and is unable to recover fine details such as hairs and wrinkles. Bahat's method successfully recovers some fine details but suffers from severe color and grid-like artifacts. We find that the training of this method is highly unstable, and we hypothesize that this is partly due to the overly constrained optimization with perfect consistency. The results denoted Ours are highly visually appealing – fine details such as hair and wrinkles are generated in a reasonable manner.

In Figure 8 we present a couple of compressed images from the test set of FFHQ and the corresponding recompressed restoration from our method with and without consistency regularization and with projection. This showcases the effectiveness of our consistency regularization in improving the consistency of the reconstructed images without detoriating their perceptual quality.

In Figure 9 we present the stochastic nature of the different methods (except the deterministic regression models). We expect to see different plausible details generated for the same compressed input image Y given different noise injection Z. Indeed, we see that our methods generate slight variations in the expression, in the beard and hair structure, in the background details and in the skin colors. Those variations are also indicated by the per-pixel standard deviation map we present for each method, where darker values represent more varying pixels in the restored images. While Bahat's method also produces stochastic results, it can be

clearly seen that most of the variation comes from color artifacts.

In Figure 10 we present more results of the different methods on the test set of FFHQ.

Table 4. Quantitative results of different methods on the FFHQ test set at QF=5. The consistency of constrained methods, marked by *, are practically zero – please refer to Subsection 5.1 and Appendix D for more details.

Method	FID (↓)	Consistency (↓)	PSNR (↑)
Regression	66.14 ± 0.00	5.2217	25.6579
Ours-A	35.26 ± 0.00	0.3203	25.4529
Bahat	65.86 ± 0.28	≈ 0*	22.6807
λ_C=0	16.46 ± 0.16	11.8848	23.3457
λ_C=0-P	20.60 ± 0.16	≈ 0*	23.4896
Ours	16.70 ± 0.14	0.7481	23.7595
Ours-P	18.54 ± 0.14	≈ 0*	23.7767
Ground Truth	10.28 ± 0.00	0	∞

E.2. ImageNet

In Table 5 we present quantitative results of the different methods on ImageNet-ctest10k at QF=10.

In Figure 11 and Figure 12 we present more results of the different methods on ImageNet-ctest10k. Note that QGAC and QGAC-GAN are not trained on QFs lower than 10, hence we do not show their results on QF=5 for fair comparison. The visual results further corroborate the quantitative results shown in Figure 3 – Our method provides the best perceptual results, creating more fine details and less artifacts and projecting our results does not detoriate their perceptual quality. Note that while QGAC-GAN provide better visual results compared to Bahat's, they are not consistent with the compressed inputs. This means that those are not valid reconstructions in the sense that they could not have created the compressed images.

E.3. LIVE1 & BSDS500

In Figure 13 and Figure 14 we present visual results of different methods on LIVE1 [44, 45] and BSDS500 [4] datasets.

Such small datasets (29 and 500 images, respectively) cannot be used for reliable deep-features-based, ensemble perceptual quality assessments (FID, KID, IS, etc.). From the official FID implementation[4] "IMPORTANT: The number of samples [...] should be greater than the dimension of the coding layer, here 2048 [...]". Hence, we do not include quantitative results for this datasets.

Table 5. Quantitative results of different methods on ImageNet-ctest10k at QF=10. The consistency of constrained methods, marked by *, are practically zero – please refer to subsection 5.1 and Appendix D for more details.

Method	FID (↓)	Consistency (↓)	PSNR (↑)
SwinIR	13.93 ± 0.00	2.5858	27.8662
FBCNN	14.85 ± 0.00	3.9929	27.6000
QGAC	16.20 ± 0.00	2.6551	27.4091
QGAC-GAN	6.93 ± 0.00	1.8640	27.0681
Bahat	13.71 ± 0.03	0.3598*	25.4243
Ours	4.76 ± 0.01	0.9696	25.5758
Ours-P	4.78 ± 0.01	0.6411*	25.6008
Ground Truth	2.67 ± 0.00	0	∞

[4] https://github.com/bioinf-jku/TTUR

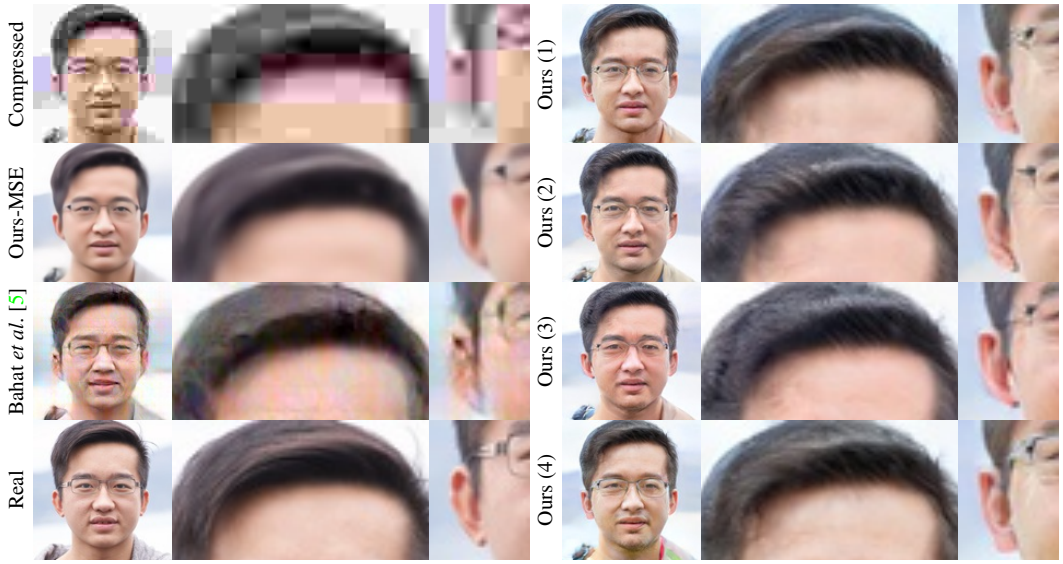

Figure 7. Zoom-in on the results of different recovery methods. Left column: The decompression of a single image from the FFHQ data set compressed using QF=5. Notice the smoothed result of Ours-MSE and the artifacts in Bahat's solution, while our method produces sharp and realistic results. Right column: Four realizations from our method that further show the stochastic nature of the results. Note the different hair patterns and ear shapes.

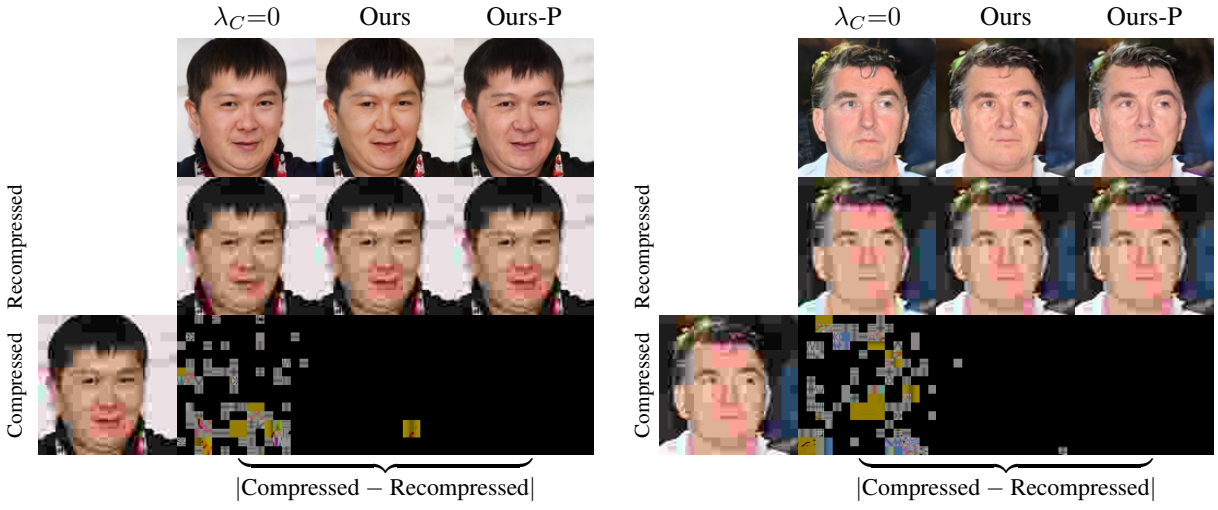

Figure 8. The difference between a compressed input and the recompressed outputs of our method with and without explicit consistency penalty and with projection. Values has been rescaled by taking the 4th root for visualization purposes. By adjusting λ_C we are able to produce reconstruction with near-perfect consistency (Ours). This allows us to project the results to achieve perfect consistency with minimal impact on the perceptual quality (Ours-P). Quantitative results can be seen in Table 4.

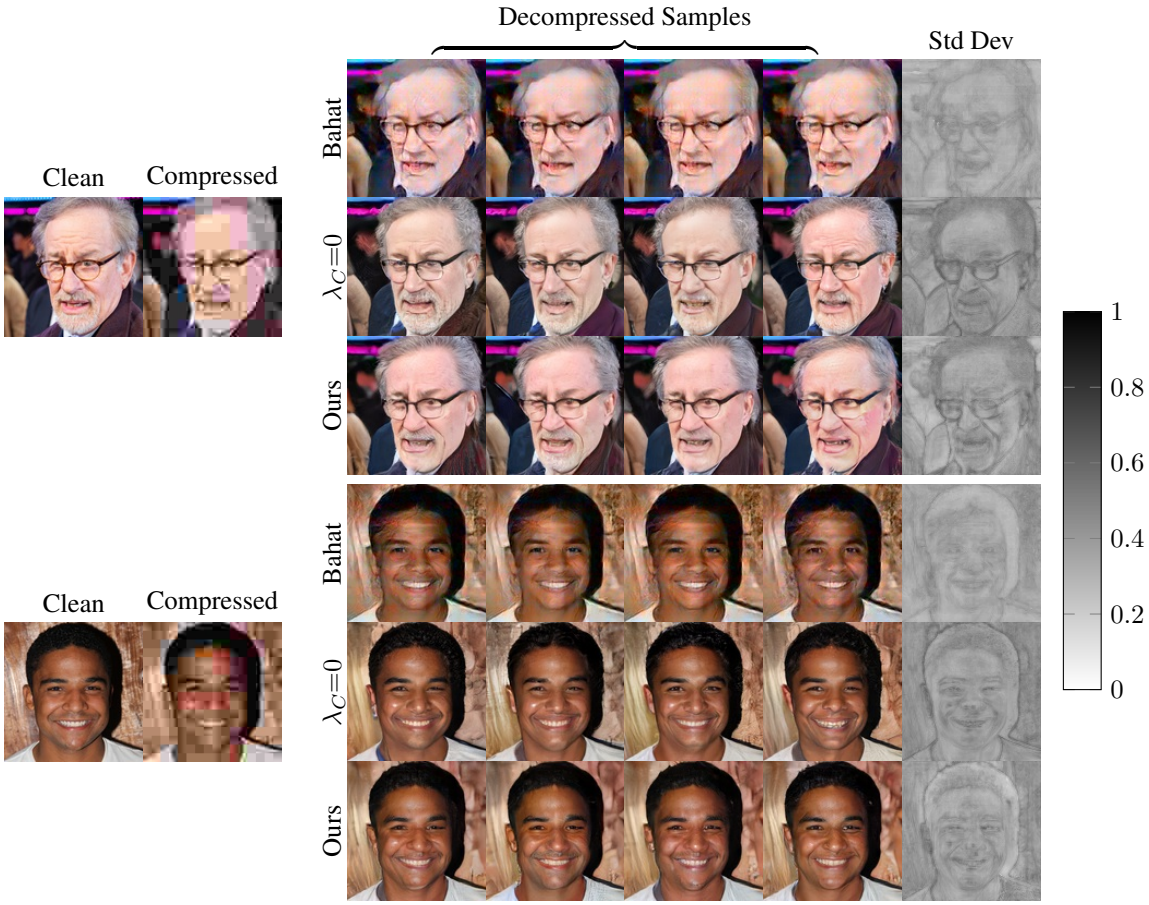

Figure 9. Stochastic variation of decompressed images using Bahat's and our methods. To the left we present two clean images and their corresponding compressed JPEG version using QF=5. To the right we present 4 realizations using each method, along with per-pixel standard deviation map calculated on 32 samples. For visualization purposes we use the 4th root of the standard deviation and add a color bar (white and black correspond to low and high standard deviations, respectively). All decompressed images were obtained using the default noise injection scheme ($z \sim \mathcal{U}(-1, 1)$ for Bahat's method and $z \sim \mathcal{N}(0, I)$ for the others).

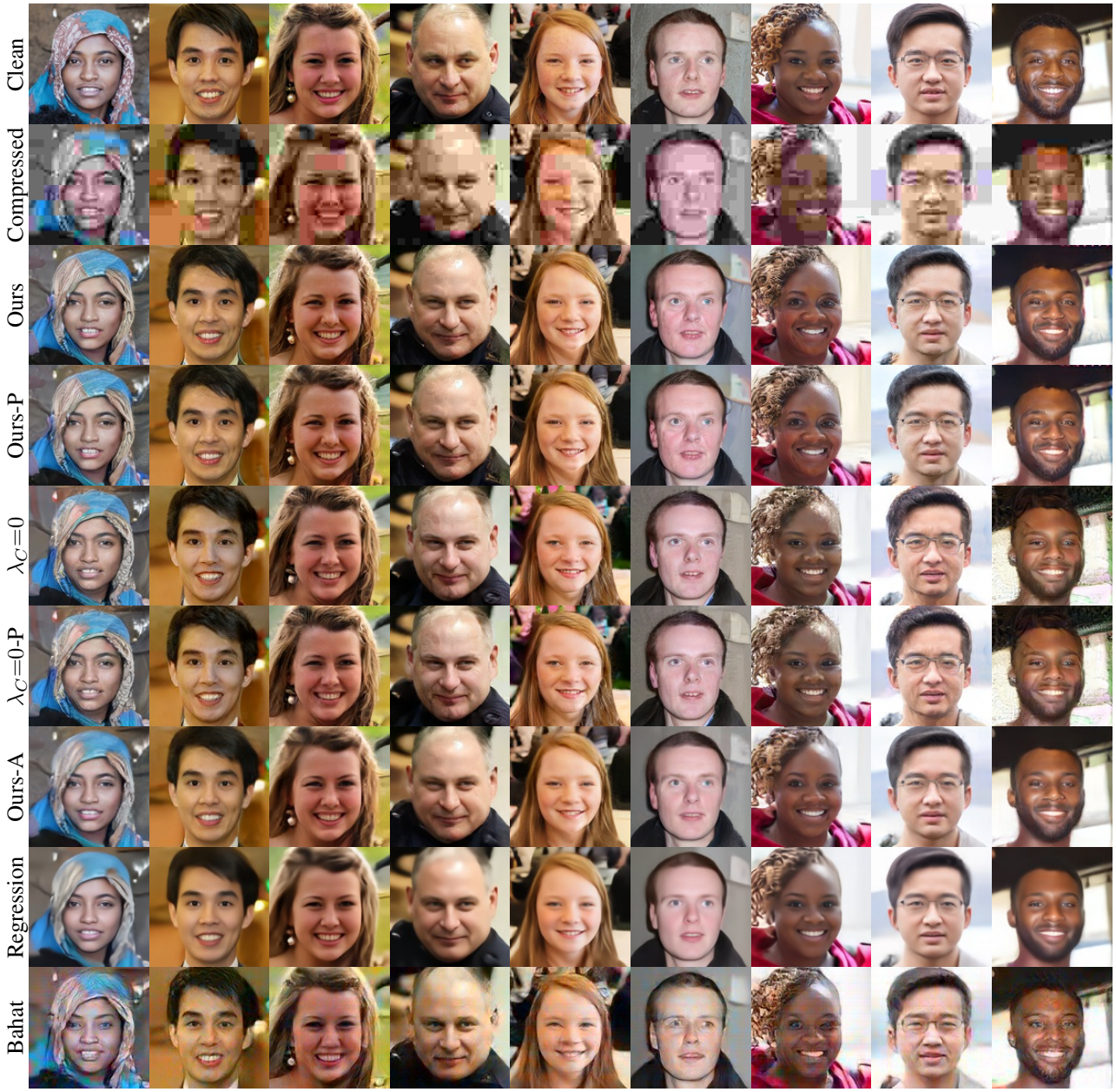

Figure 10. Decompression results using different methods on FFHQ images compressed using JPEG with QF=5. For stochastic methods (all except for Regression), the default noise injection scheme ($z \sim \mathcal{U}(-1, 1)$ for Bahat's method and $z \sim \mathcal{N}(0, I)$ for the others) was used during inference.

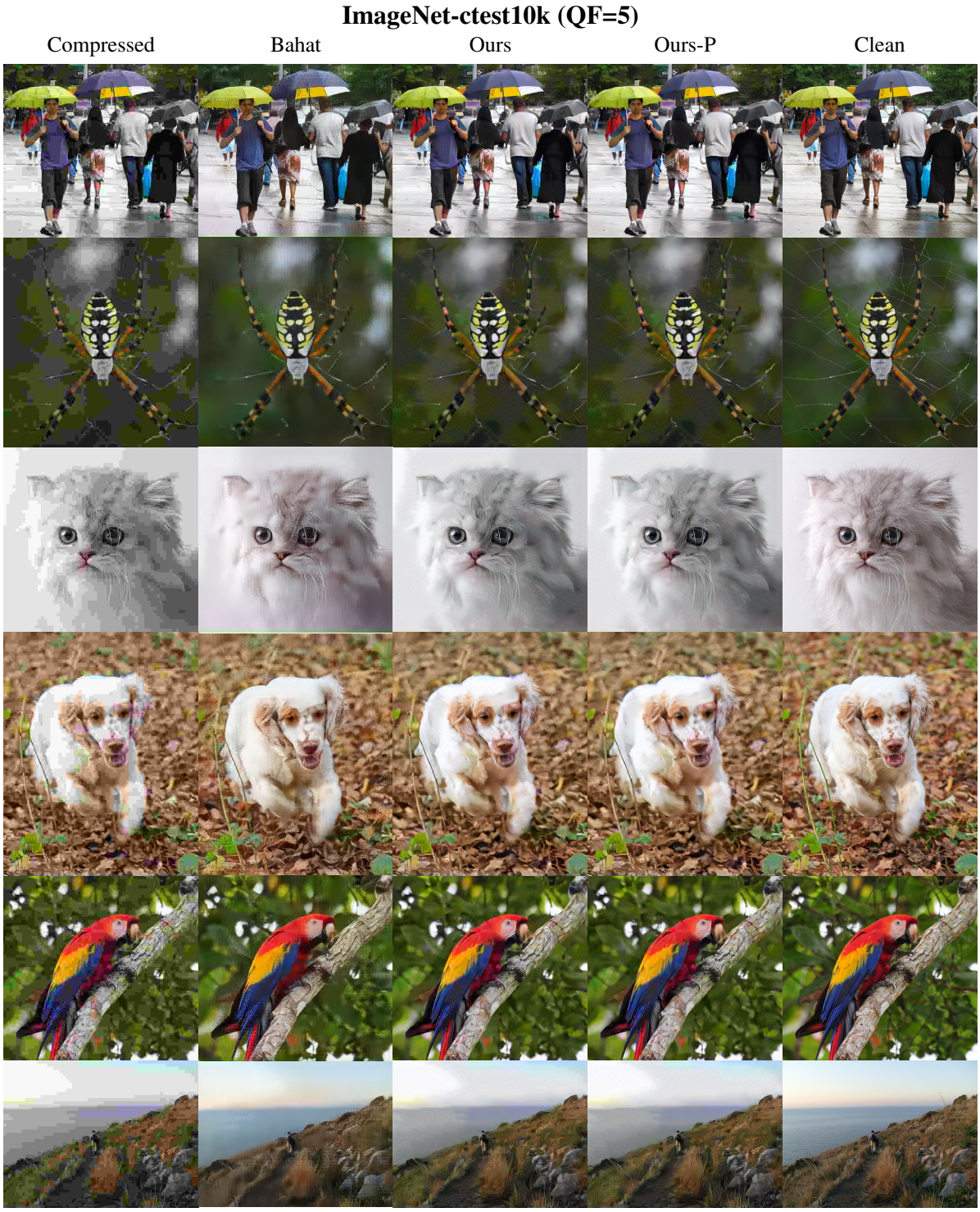

Figure 11. Decompression results using our method and Bahat *et al.*'s on ImageNet JPEG compressed images with QF=5.

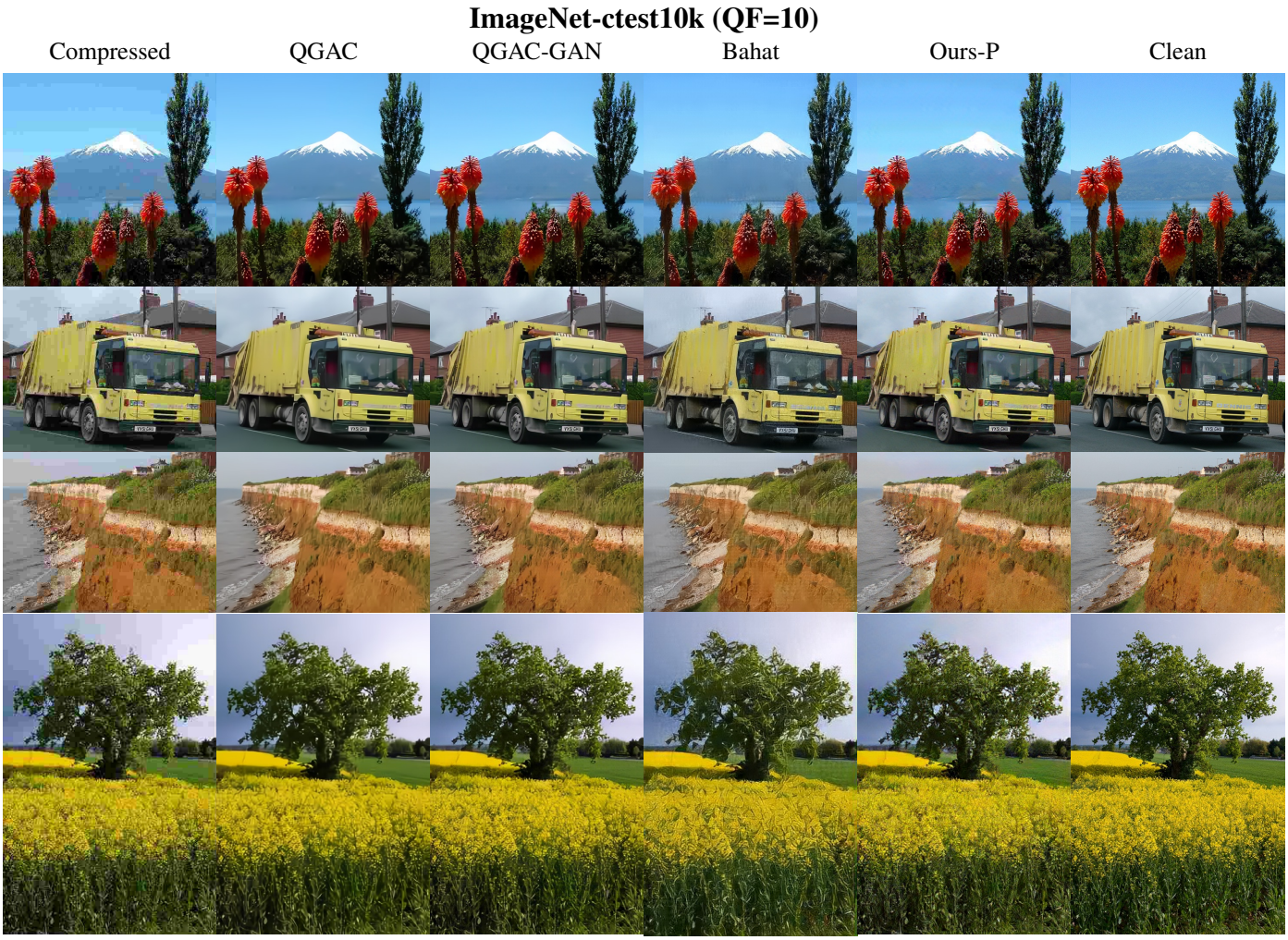

Figure 12. Decompression results using different methods on ImageNet JPEG compressed images with QF=10.

Figure 13. Decompression results using different methods on LIVE1 JPEG compressed images with QF=10.

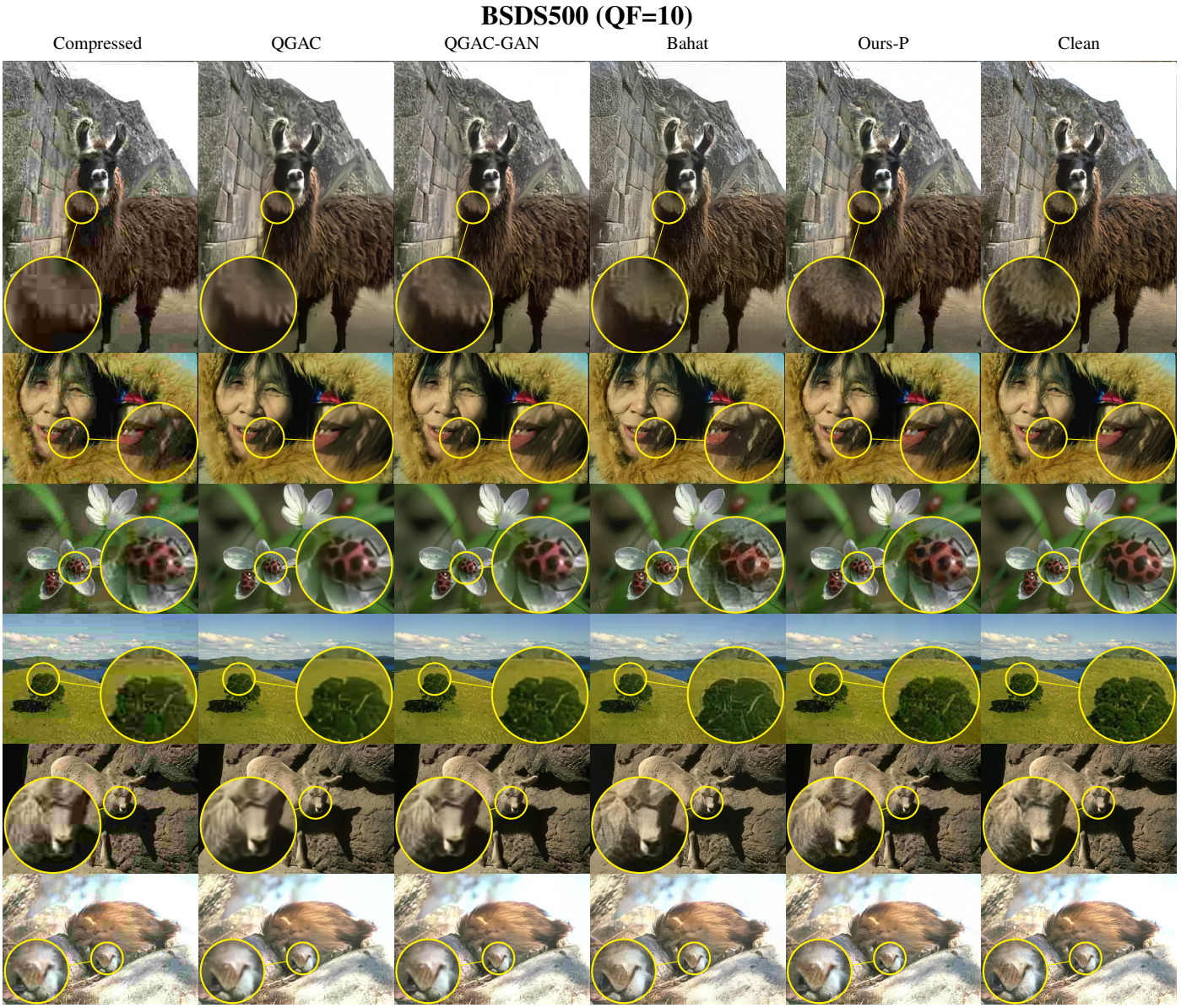

Figure 14. Decompression results using different methods on BSDS500 JPEG compressed images with QF=10.