Uncertainty Quantification via Neural Posterior Principal Components: Supplementary Material

Elias Nehme
Technion - Israel Institute of Technology
seliasne@campus.technion.ac.il

Omer Yair
Technion - Israel Institute of Technology
omeryair@campus.technion.ac.il

Tomer Michaeli
Technion - Israel Institute of Technology
tomer.m@ee.technion.ac.il

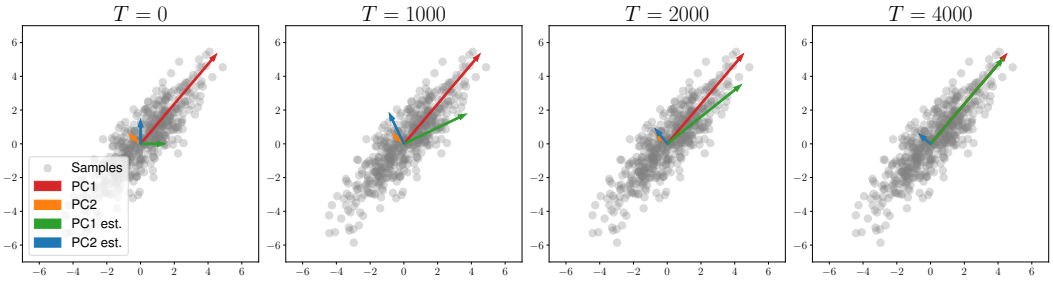
A Experimental details

Architectures In all experiments, we used a simple U-Net [2, 11] architecture with four downsampling/upsampling levels. Downsampling was achieved using 2×2 Maxpooling, and upsampling was performed using resize-convolutions [10] with nearest-neighbor interpolation. The input image is first projected to 32 feature channels using two conv layers. Then, at each level, the encoder halves the spatial resolution and doubles the number of channels using two conv layers to result in 32, 64, 128, and 256 channels at levels 1 to 4 with the output of 4 being the encoder's bottleneck. The final output is given by a 1×1 convolution reducing the number of channels to 1 for gray-scale images and to 3 for RGB images. Throughout the network, we used LeakyReLU activations with a negative slope of 0.1 for non-linearities and instance normalization to z-score feature activations. For training NPPC to wrap around pre-trained mean estimators, we used the same U-Net and only changed the number of output filters at the last layer to be K or $3K$ for gray-scale or RGB images, respectively. The resulting output channels were then reshaped to K independent images constituting $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$. While we are aware of more advanced U-Net variants such as [3, 9, 12], these improvements are orthogonal to NPPC. Therefore, in this work, we kept it as simple as possible.

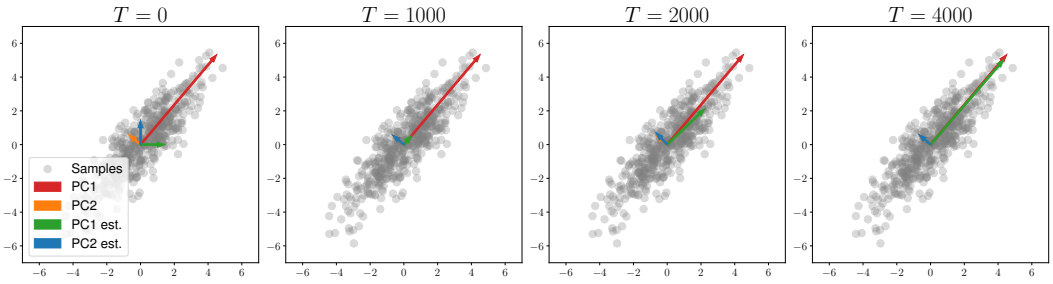
Per-task details In the super-resolution task we upsampled the low-resolution image using nearest neighbor interpolation prior to feeding it to the network. For the posterior mean prediction, we only learned the required residual from the input to produce the prediction. Similarly, for the inpainting task, we only learned the missing part and produced the prediction by summing the input with the masked output to not waste capacity. For all face models, we learned the directions on full images of resolution 256×256. On the other hand, for the biological image-to-image translation task, we did not use residual learning as the input and output are not similar. In addition, we learned the directions on 64×64 patches cropped from the full 1024×1024 images, as cell information tends to be local. Our architecture is fully convolutional, and hence at test time, we tested on bigger patches of 128×128 as shown in Fig. 6 in the main text, with more examples in Appendix D.

Optimization For all experiments with used the Adam optimizer [6] with an initial learning rate of 0.001, and $\beta_1 = 0.9, \beta_2 = 0.999$. For the CelebA-HQ experiments, the learning rate was dropped by a factor of 10 if the validation loss stagnated for more than 5 epochs, and the minimum learning rate was $5 \cdot 10^{-6}$. For other tasks, we did not use this scheduler, as the results were roughly the same with or without dropping the learning rate. The batch size and the number of epochs were tuned per task according to available GPU memory: batch size of 128 and 300 epochs for MNIST, 16 and 50 for CelebA-HQ, and 64 and 1800 for the biological dataset. When wrapping around a fixed posterior mean predictor, it took about 20 minutes, 5 hours, and 2 hours to train NPPC on MNIST, CelebA-HQ, and the biological data, respectively. In cases where NPPC also predicts the mean jointly with the

37th Conference on Neural Information Processing Systems (NeurIPS 2023).



(a) Standard.



(b) Ramping up λ_2 at T=1000.

Figure A1: **Scheduling λ_2 for a fixed $\hat{\boldsymbol{x}} = \mathbf{0}$.** In (a) we visualize the optimization steps while estimating the PCs and the variances at the same time, as opposed to ramping up λ_2 after 1000 steps in (b). Notice that at step $T = 1000$, the estimated PCs in (a) (blue and green arrows) are still not aligned with the ground truth PCs compared to (b), due to the optimization of their norms. Eventually, after $T = 4000$ steps both strategies converge to the same solution.

directions, it took roughly $2\times$ longer (*e.g.,* $\approx$40 minutes for MNIST) due to a more expressive model and the scheduling explained next.

Jointly predicting $\hat{\boldsymbol{x}}$ and scheduling λ_1, λ_2 Training NPPC to predict both the posterior mean $\hat{\boldsymbol{x}}$ and the PCs $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$ with their variances $\sigma_1^2, \ldots, \sigma_k^2$ requires delicate scheduling. The reason for this instability is that for PCA to work properly the data needs to be centered with zero mean. Hence, during the early iterations when the mean estimate hasn't converged yet, estimating the PCs and the variances along them is challenging. In the extreme case of a trivial mean estimate $\hat{\boldsymbol{x}} = \mathbf{0}$, the first PC converges to the mean and hence bears little meaning with respect to our purpose. To remedy this, we can ramp up the loss functions sequentially; *i.e.,* initially only $\mathcal{L}_{\boldsymbol{\mu}}$ is on, and after 20 epochs we turn on the PCs and variances loss functions. However, turning both losses $\mathcal{L}_{\boldsymbol{w}}$ and $\mathcal{L}_{\sigma}$ at the same time would lead to a similar conflicting behavior, as in the early iterations the model will try to optimize the variances along random directions. Hence, while we empirically find this to work given enough time for convergence, we opted to ramp up λ_2 after another 20 epochs such that the PCs are almost converged, and only their norm is being optimized. In Fig. A1 we visualize this effect in a 2D toy example of correlated Gaussian samples. Note that predicting both the mean and the directions requires an architecture with enough capacity to handle both tasks (see Fig. A2), especially for severely ill-posed problems. Therefore, for our CelebA-HQ and biological dataset experiments, we focused on a two-step setting where the mean estimate is first learned using a separate U-Net, and afterward, the result is wrapped around with NPPC predicting only the PCs and the variances. We found this strategy to require a significantly lighter model (*e.g.,* $4\times$ on MNIST denoising), while also stabilizing training and mitigating the need for scheduling λ_1 and λ_2.

Loss normalization Learning the PCs and the variances using (4) and (5) is straightforward. Nonetheless, to standardize the loss values across tasks and datasets, we used a slightly modified loss by reweighting the terms for the i^{th} example with the error norm $\|\boldsymbol{e}_i\|$; *i.e.,* for (4) we divided the term $|\boldsymbol{w}_k\left(\boldsymbol{y}_i, \hat{\boldsymbol{x}}_i; \varphi\right)^\top \boldsymbol{e}_i|^2$ by $\|\boldsymbol{e}_i\|^2$, and for (5) we divided the inner term by $\|\boldsymbol{e}_i\|^4$. This normalization has two benefits: 1. It standardizes the loss values by the difficulty of the task at hand on a per-sample basis. Meaning, for harder samples the PCs are expected to capture more of the error norm while

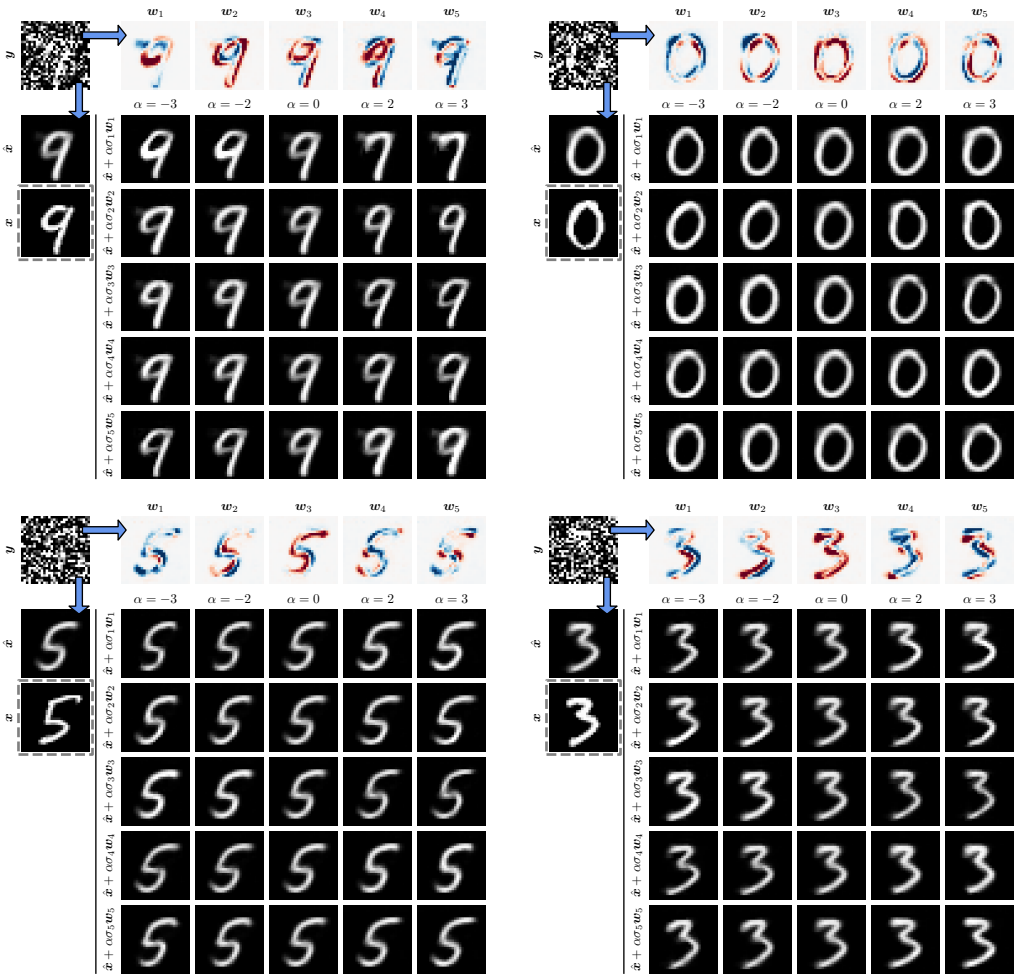


Figure A2: **Joint learning of $\hat{\boldsymbol{x}}$ and $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$.** Here we present the results of training a single model to predict both the mean $\hat{\boldsymbol{x}}$ and the PCs $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K$ on the task of denoising handwritten digits. While the resulting model is able to achieve similar results to two separate models (one for estimating the mean and one for estimating the PCs), this necessitated a U-net with double the number of channels compared to the model used in Fig. 5a, leading to an architecture with approximately $4\times$ the number of parameters.

being allowed an error margin in estimating the variances. 2. This facilitates a constant set of weights $\lambda_1 = 1, \lambda_2 = 1$ across tasks, thus overcoming the need for expensive hyperparameter tuning.

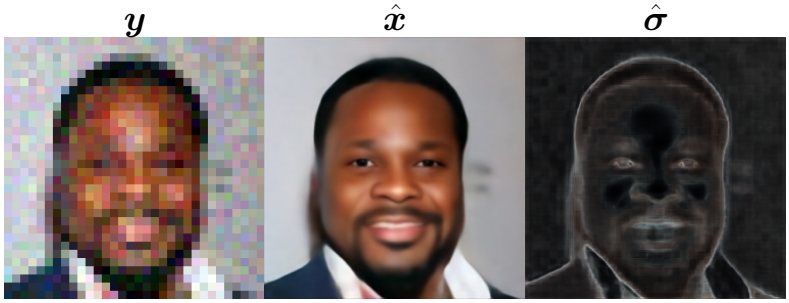
Predicting per-pixel variance For jointly predicting a mean estimate $\hat{\boldsymbol{x}}$ and a per-pixel variance map $\hat{\boldsymbol{\sigma}}_i^2$ (*e.g.,* in Fig. 1a), we trained a U-Net with two outputs to minimize the loss function

$$\mathcal{L}_{\text{per-pixel}}\left(\mathcal{D}, \boldsymbol{\theta}\right) = \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathcal{D})} \|\hat{\boldsymbol{x}}_i\left(\boldsymbol{y}_i; \boldsymbol{\theta}\right) - \boldsymbol{x}_i\|_2^2 + \|\hat{\boldsymbol{\sigma}}_i^2\left(\boldsymbol{y}_i; \boldsymbol{\theta}\right) - \left(\hat{\boldsymbol{x}}_i\left(\boldsymbol{y}_i; \boldsymbol{\theta}\right) - \boldsymbol{x}_i\right)^2\|_2^2. \tag{A1}$$

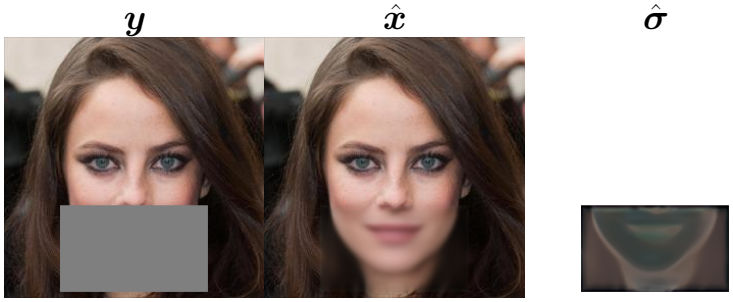
Note that this is slightly different from the standard maximum likelihood approach for predicting $\hat{\boldsymbol{\sigma}}$ (*e.g.,* as in [5]), where the loss is given by

$$\mathcal{L}_{\text{NLL}}\left(\mathcal{D}, \boldsymbol{\theta}\right) = \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathcal{D})} \frac{\|\hat{\boldsymbol{x}}_i\left(\boldsymbol{y}_i; \boldsymbol{\theta}\right) - \boldsymbol{x}_i\|_2^2}{\hat{\boldsymbol{\sigma}}_i^2\left(\boldsymbol{y}; \boldsymbol{\theta}\right)} + \log \hat{\boldsymbol{\sigma}}_i^2\left(\boldsymbol{y}; \boldsymbol{\theta}\right). \tag{A2}$$

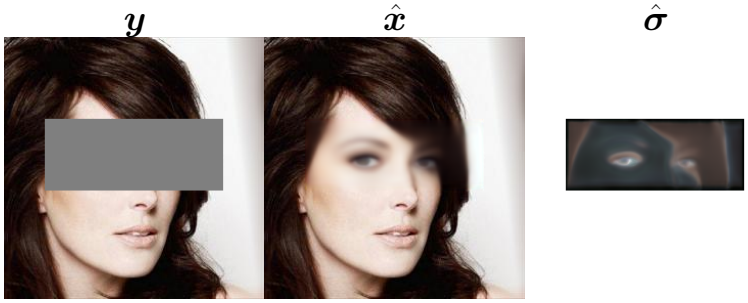
Nonetheless, as previously suggested in numerous works (*e.g.,* [1]), we found the MLE approach not stable and difficult to optimize even when reparameterizing and predicting $\boldsymbol{s}_i = \log \hat{\boldsymbol{\sigma}}_i^2$. Therefore,



(a) Noisy 8× super-resolution.



(b) Mouth inpainting.



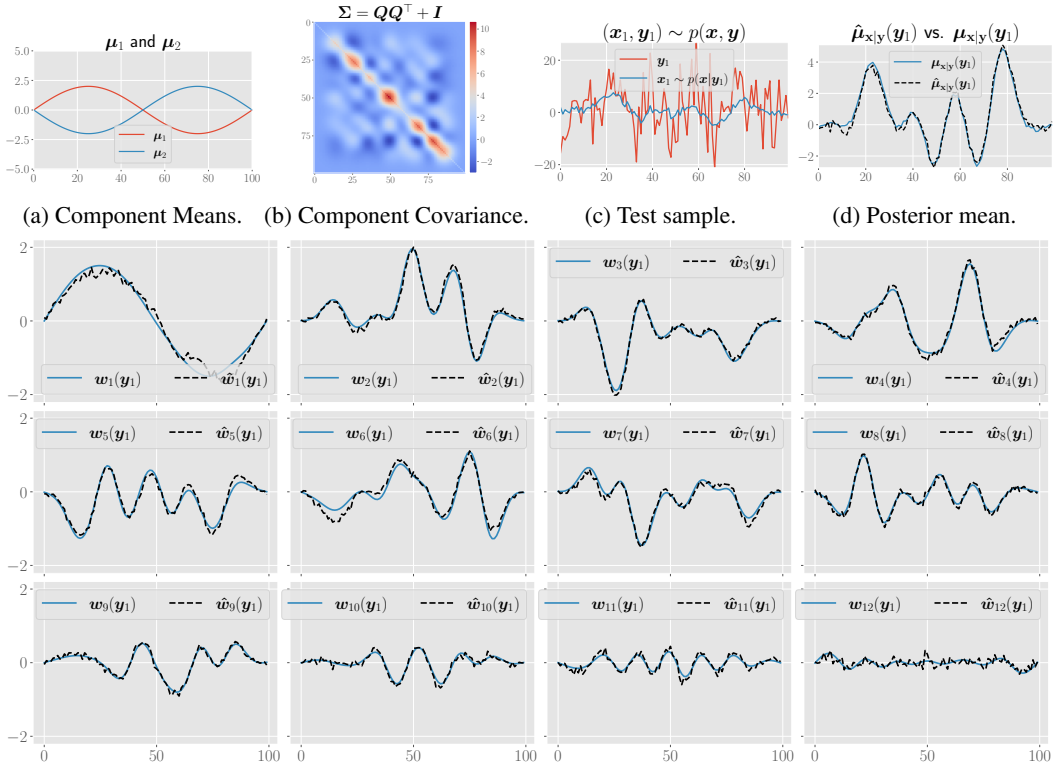
(c) Eyes inpainting.

Figure A3: **Per-pixel variance maps for CelebA-HQ**. Here we show representative per-pixel standard deviation maps for different tasks on CelebA-HQ. Intuitively, areas with high variance correspond to edges (*e.g.,* (a)), and uncertain shapes/colors (*e.g.,* (b) and (c)). However, as explained earlier, these on their own convey little information regarding the different possibilities.

in similar spirits to the loss employed to predict the error norm in [1], we optimized (A1) instead. In Fig. A3 we show representative estimated $\hat{\boldsymbol{\sigma}}$ maps on the tasks of 8× noisy super-resolution and inpainting from the CelebA-HQ experiments.

B Analytical posterior for Gaussian mixture denoising

In Figure 3 and Table 1, we compared NPPC to the GT posterior for a Gaussian mixture prior with 2 components. Here we provide the closed-form expressions for the analytically derived posterior and its two first moments for completeness. The denoising task we assumed was $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{x}$ comes from a mixture of $L = 2$ Gaussians, $p(\boldsymbol{x}) = \sum_{\ell=1}^{L} \pi_\ell \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$, and $\mathbf{n} \sim \mathcal{N}(\cdot; \mathbf{0}, \sigma_\varepsilon^2 \boldsymbol{I})$ is a white Gaussian noise. Let $\mathbf{c}$ be a random variable taking values in $\{1, \dots, L\}$ with probabilities $\{\pi_1, \dots, \pi_L\}$. Then we can write the posterior by invoking the law of total probability conditioned



(a) Component Means. (b) Component Covariance. (c) Test sample. (d) Posterior mean.

(e) PCs comparison.

Figure A4: **Denoising samples from a 100-dimensional Gaussian mixture model**. (a) Gaussian mixture means. (b) Covariance matrix of each component. (c) Examined test sample $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ (note that the GT posterior sample $\boldsymbol{x}_i$ is only presented for illustration and is unknown to NPPC). (d) Comparison of the estimated and GT posterior means for the test input $\boldsymbol{y}_i$ in (c). (e) Comparison of the GT (blue) and estimated (dashed black) first 12 PCs scaled by their (respective) GT/estimated σ.

on the event $\mathbf{c} = \ell$,

$$\begin{aligned} p(\boldsymbol{x}|\boldsymbol{y}) &= \sum_{\ell=1}^{L} p_{\mathbf{x}|\mathbf{y},\mathbf{c}}(\boldsymbol{x}|\boldsymbol{y},\ell) p_{\mathbf{c}|\mathbf{y}}(\ell|\boldsymbol{y}) \\ &= \sum_{\ell=1}^{L} p_{\mathbf{x}|\mathbf{y},\mathbf{c}}(\boldsymbol{x}|\boldsymbol{y},\ell) \frac{p_{\mathbf{y}|\mathbf{c}}(\boldsymbol{y}|\ell) p_{\mathbf{c}}(\ell)}{p_{\mathbf{y}}(\boldsymbol{y})} \\ &= \sum_{\ell=1}^{L} \mathcal{N}(\boldsymbol{x}; \tilde{\boldsymbol{\mu}}_\ell, \tilde{\boldsymbol{\Sigma}}_\ell) \frac{q_\ell \pi_\ell}{\sum_{\ell'=1}^{L} q_{\ell'} \pi_{\ell'}}, \end{aligned} \tag{A3}$$

where we denoted

$$\begin{aligned} q_i &= \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i + \sigma_\varepsilon^2 \boldsymbol{I}), \\ \tilde{\boldsymbol{\mu}}_i &= \boldsymbol{\mu}_i + \boldsymbol{\Sigma}_i (\boldsymbol{\Sigma}_i + \sigma_\varepsilon^2 \boldsymbol{I})^{-1} (\boldsymbol{y} - \boldsymbol{\mu}_i) \\ \tilde{\boldsymbol{\Sigma}}_i &= \boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_i (\boldsymbol{\Sigma}_i + \sigma_\varepsilon^2 \boldsymbol{I})^{-1} \boldsymbol{\Sigma}_i. \end{aligned} \tag{A4}$$

As evident in Fig. 3 and also in Equation (A3), the GT posterior is itself a Gaussian mixture with updated parameters. However, to simplify calculations for the Wasserstein 2-distance comparisons in Table 1, we approximated the GT posterior using a Gaussian with the same first 2 moments,

i.e., $p(\boldsymbol{x}|\boldsymbol{y}) \approx \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}})$, where,

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} &= \sum_{\ell=1}^{L} \tilde{\boldsymbol{\mu}}_\ell \frac{q_\ell \pi_\ell}{\sum_{\ell'=1}^{L} q_{\ell'} \pi_{\ell'}}, \\ \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}} &= \sum_{\ell=1}^{L} \{(\tilde{\boldsymbol{\mu}}_\ell - \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}})^2 + \tilde{\boldsymbol{\Sigma}}_\ell\} \frac{q_\ell \pi_\ell}{\sum_{\ell'=1}^{L} q_{\ell'} \pi_{\ell'}}.\end{aligned} \tag{A5}$$

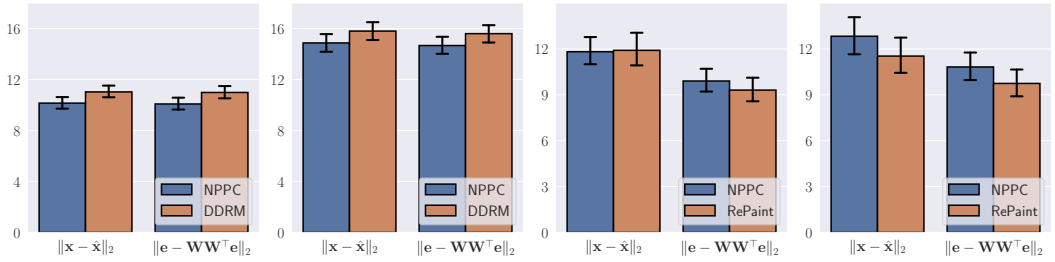
With this approximation, the Wasserstein 2-distance to the estimated Gaussian constructed by NPPC, $\hat{p}(\boldsymbol{x}|\boldsymbol{y}) = \mathcal{N}(\hat{\boldsymbol{\mu}}_{\mathbf{x}|\mathbf{y}}, \hat{\boldsymbol{\Sigma}}_{\mathbf{x}|\mathbf{y}} = \boldsymbol{W}_\star \boldsymbol{W}_\star^\top)$, where $\boldsymbol{W}_\star$ has $\hat{\sigma}_k \boldsymbol{w}_k$ in its kth column, can be efficiently computed using the closed-form expression

$$d_{\mathcal{W}}(p(\boldsymbol{x}|\boldsymbol{y}), \hat{p}(\boldsymbol{x}|\boldsymbol{y}))^2 = \|\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} - \hat{\boldsymbol{\mu}}_{\mathbf{x}|\mathbf{y}}\|_2^2 + \operatorname{Tr}\left(\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}} + \hat{\boldsymbol{\Sigma}}_{\mathbf{x}|\mathbf{y}} - 2(\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}}^{1/2} \hat{\boldsymbol{\Sigma}}_{\mathbf{x}|\mathbf{y}} \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}}^{1/2})^{1/2}\right). \tag{A6}$$

Figure A4 visualizes the 100-dimensional Gaussian mixture with $L = 2$ components referred to in Table 1. The component means (A4a) were chosen such that $\boldsymbol{\mu}_1 = -\boldsymbol{\mu}_2$, both with equal weights $\pi_1 = \pi_2 = 0.5$, and the same low-rank covariance matrix $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{Q}\boldsymbol{Q}^\top + \boldsymbol{I}$ with $\operatorname{rank}(\boldsymbol{Q}) = 12$. Similar to the 2D case, the noise is assumed to be white Gaussian with a per-dimension variance of $\sigma_\varepsilon^2 = 100$. As demonstrated previously in 2D, NPPC is also able to recover the top $K = 12$ PCs and variances accurately in the high-dimensional case.

C Comparison to posterior samplers

In Figure A5, we plot representative comparisons from Table 2 using $K = 5$ PCs as bar plots. The error bars signify a 95% confidence interval constructed using the standard error of the mean. Note that in all tested experiments, NPPC achieved comparable results within the confidence interval of the respective baseline. Moreover, for the task of noisy super-resolution, we observe that only 5 PCs are not enough to capture a significant portion of the error norm (neither by NPPC nor by DDRM [4]). On the other hand, for the task of inpainting, both NPPC and RePaint [8] were able to span a significant portion of the error. To gain a better insight into this phenomenon, in Fig. A6 we plot the fraction of residual error variance against the number of PCs using the posterior samples of DDRM and RePaint. These plots were generated by computing PCA on 100 samples from DDRM and RePaint for each input image. The result suggests that for the task of noisy super-resolution, the error covariance has a wider spread, and the decay rate is significantly slower than for inpainting. For example, using $K = 10$ PCs, the fraction of unexplained variance for super-resolution is ≈ 0.8 compared to ≈ 0.4 for inpainting. This result is not surprising as the optimal number of PCs K is task-dependent and is expected to vary based on the complexity of the posterior. To better exemplify this point, in appendix D.1 we demonstrate the result of NPPC on the task of (face) image colorization.



(a) Noisy $4\times$ SR (b) Noisy $8\times$ SR (c) Mouth inpainting (d) Eyes inpainting

Figure A5: **Comparison to posterior samplers**. Error bars denote a 95% confidence interval constructed using the standard error of the mean. In all tasks (including noiseless super-resolution omitted here for brevity), the performance of NPPC was within a 95% confidence interval around the mean of the respective baseline, sometimes even leading to better results (*e.g.,* (a) and (b)).

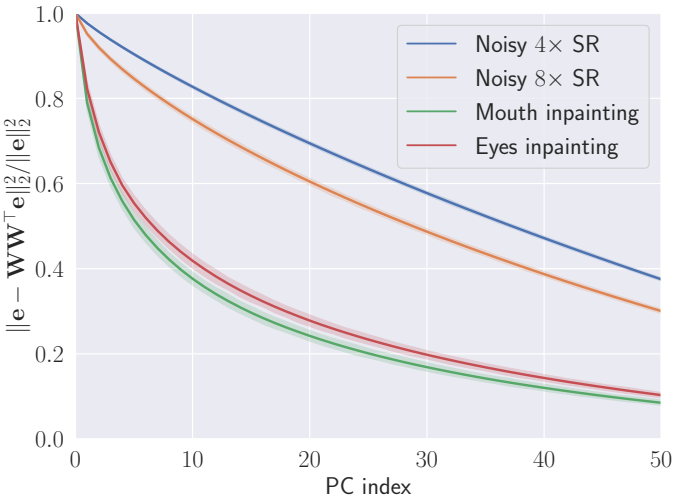


Figure A6: **Fraction of residual error magnitude vs the number of PCs**. Here we visualize the decay in the fraction of unexplained variance (of the error) as more principal components are added. Note that for the task of super-resolution, the decay is significantly slower compared to the task of inpainting. This suggests that multiple PCs roughly capture the same amount of variance, hence explaining the orthogonality of the subspace found by NPPC compared to posterior sampling.

D Additional results

D.1 Image colorization

Inverse problems in imaging vary in their level of difficulty, dictating the complexity of the respective posterior. NPPC is particularly suited for tasks that exhibit strong correlations between output pixels, such that few PCs faithfully span the posterior covariance. One example of such a task is image colorization, where given a grayscale image $\boldsymbol{y}$, a model is trained to restore the RGB color image $\boldsymbol{x}$. While this task is severely ill-posed, with proper cues and priors encoded in the form of training examples, neural models are able to predict plausible colorizations [7, 13]. Nonetheless, for image regions with multiple possible colorizations, a model trained to minimize the MSE will inevitably regress to the mean producing shades of gray and reflecting dataset bias (*e.g.,* average skin and hair color). Figures A7 and A8 demonstrate the application of NPPC to the problem of face colorization on the CelebA-HQ dataset. First, we trained a model to regress the RGB image $\boldsymbol{x}$ from the corresponding grayscale input $\boldsymbol{y}$. Afterward, we trained NPPC to wrap around the mean estimate $\hat{\boldsymbol{x}}$, and output the first $K = 5$ PCs. Here, the resulting PCs efficiently capture the majority of the error variance, enabling high-quality reconstructions of the ground truth on test samples, by adding the projected error $\hat{\boldsymbol{e}}_i = \boldsymbol{W}_i \boldsymbol{W}_i^\top \boldsymbol{e}_i$ to the mean estimate $\hat{\boldsymbol{x}}_i$. Note that this result uses privileged information (*i.e.,* $\boldsymbol{e}_i$) unavailable at test time, and is presented here merely to reinforce our intuition that NPPC is efficient for posteriors with a spectrally-concentrated covariance. In general, using PCs to visualize uncertainty is very appealing as we can efficiently communicate the different possibilities to the user with a few sliders. However, for a large number of PCs K, this strategy becomes impractical and tedious, warranting further research of condensed alternative representations of uncertainty.

D.2 Sequential vs joint PC learning

As explained in Sec. 3.2, we use `Stopgrad` to recover the solution of sequential training while learning the PCs jointly. Although this is guaranteed in theory, whether the two are equivalent in practice still warrants empirical validation. To realize sequential training without significantly altering the number of parameters used to predict the PCs, we trained multiple models with an increasing number of PCs $K = 1, 2, \ldots, 5$, and compared the PCs backward across different models. The results for the task of image colorization on CelebA-HQ confirm that end-to-end training leads to approximately the same PCs as sequential training (see Fig. A9). In addition, the resulting first 5 PCs when training two NNs with 5/10 components were very similar (up to a flipped sign) with an

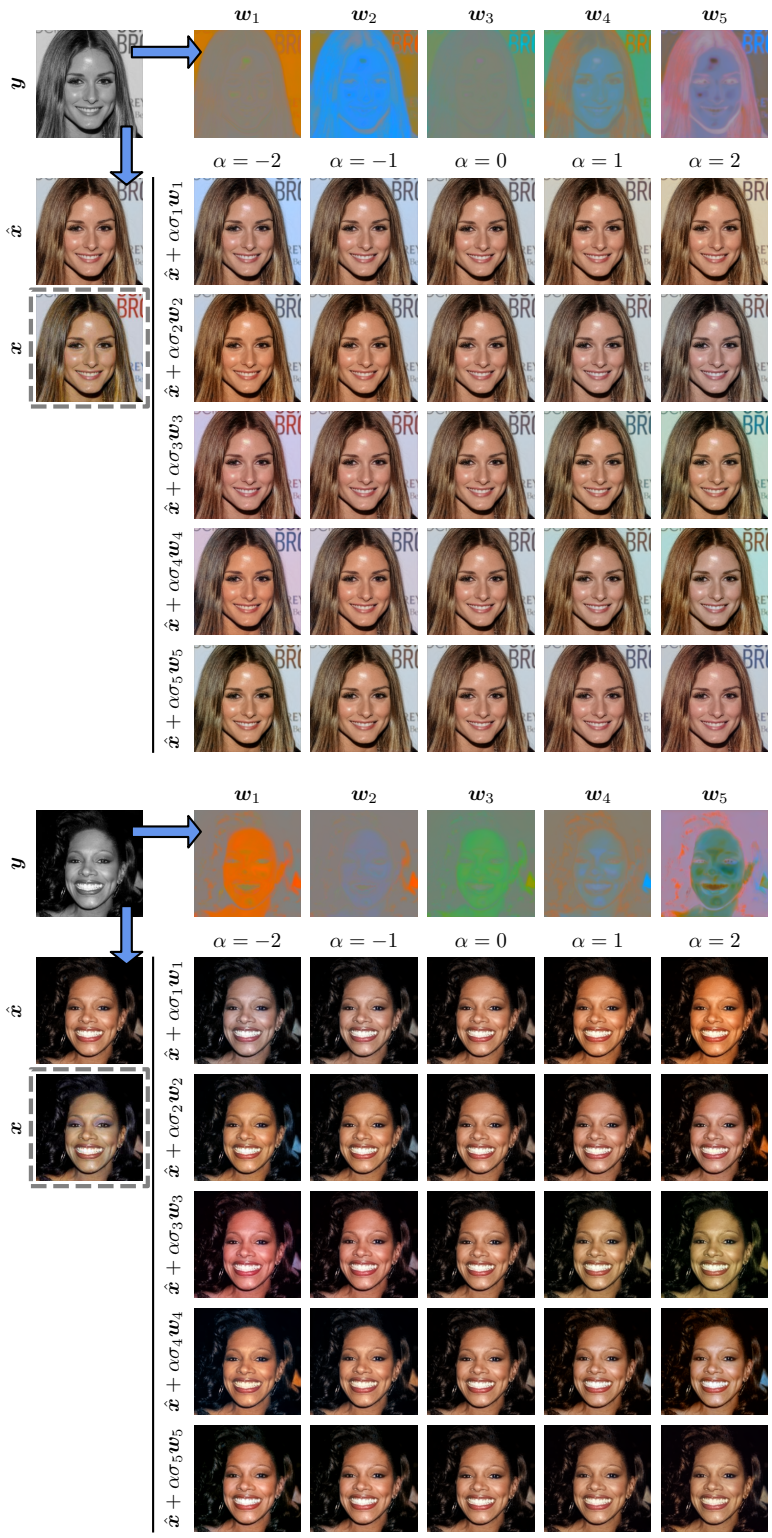


Figure A7: **Face colorization on CelebA-HQ**. Here we showcase the result of NPPC applied to the task of image colorization, *i.e.,* going from a grayscale measurement $\boldsymbol{y}$ to an RGB image x. The resulting PCs span various semantic changes including hair, skin and background colors.

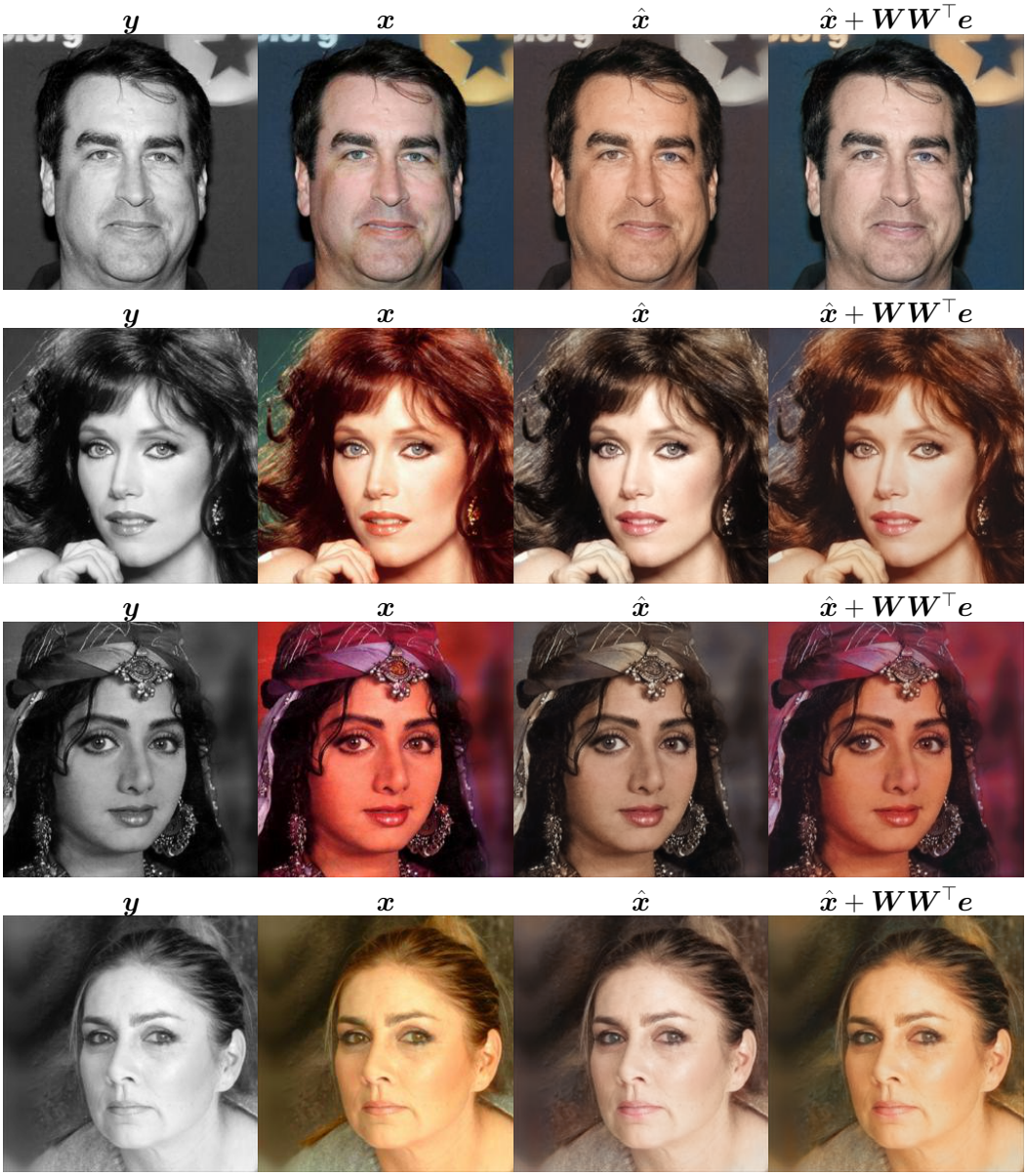
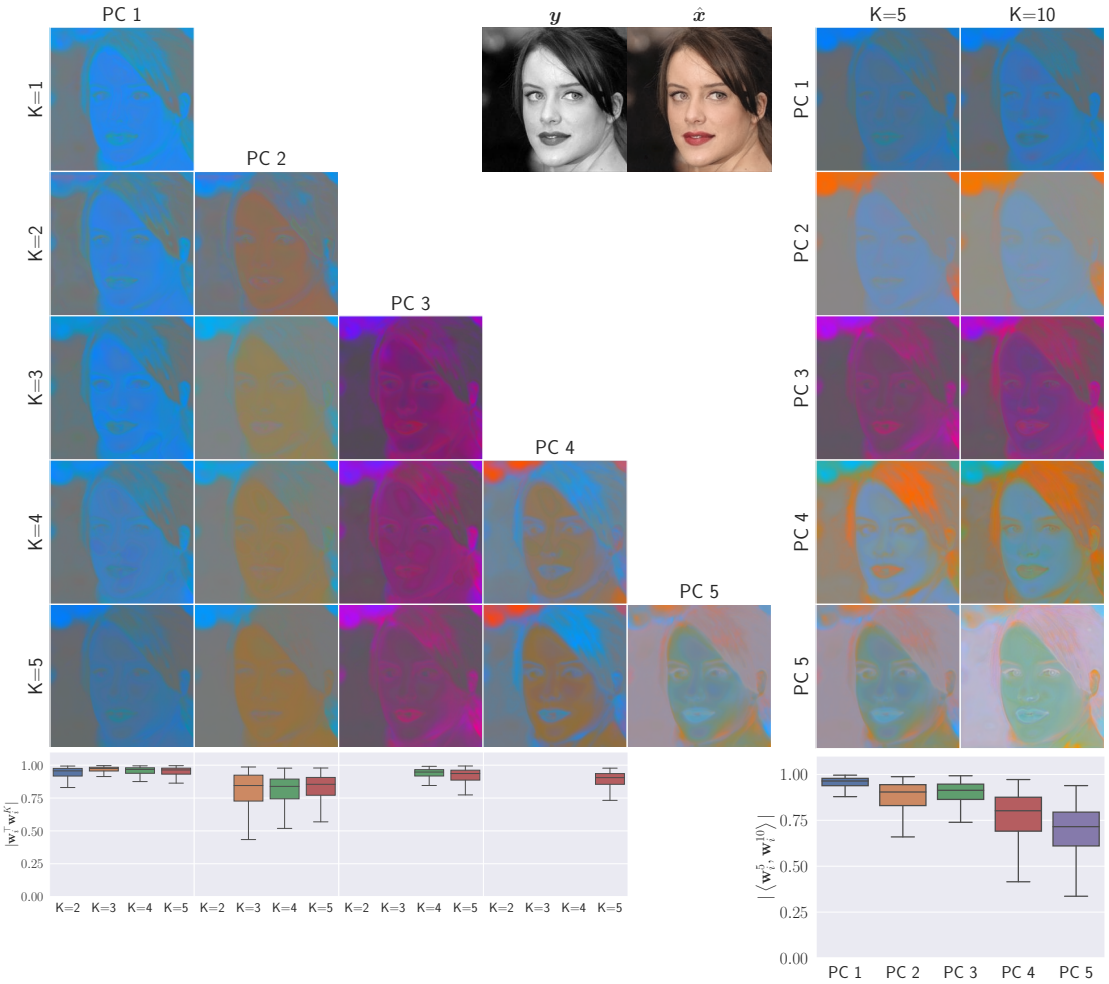


Figure A8: **Reconstructing test samples using 5 PCs**. Here we test the approximation error of the first $K = 5$ PCs found by NPPC by reconstructing the ground truth error $\boldsymbol{e}_i = \hat{\boldsymbol{x}}_i - \boldsymbol{x}_i$ of test samples. The reconstructed error $\hat{\boldsymbol{e}}_i = \boldsymbol{W}_i \boldsymbol{W}_i^\top \boldsymbol{e}_i$ is added to the mean estimate $\hat{\boldsymbol{x}}_i$ to approximate the ground truth RGB image $\boldsymbol{x}_i$. Note that at test time we do not have access to $\boldsymbol{e}_i$, therefore this result is only a sanity check to verify our PCs.

average cosine similarity of 0.9 across the first 3 PCs, and 0.83 overall. For later PCs (e.g. the 4^{th} and 5^{th}) the similarity slightly drops as the error variance along multiple PCs is roughly the same (see Fig. A10), and hence the ordering of the PCs becomes less distinct and prune to optimization errors. These results further validate that NPPC consistently outputs the PCs in the correct order.

D.3 Predicted variance validation

As mentioned in Sec. 3, there exists no ground truth uncertainty in image restoration datasets. This is because each element in the dataset is comprised of a *single* posterior sample $\boldsymbol{x}_i$ from $p_{\mathbf{x}|\mathbf{y}}(\mathbf{x}|\mathbf{y} = \boldsymbol{y}_i)$ for each observed image $\boldsymbol{y}_i$. This is the reason why beyond toy examples we evaluated the quality of



(a) Sequential training vs end-to-end.

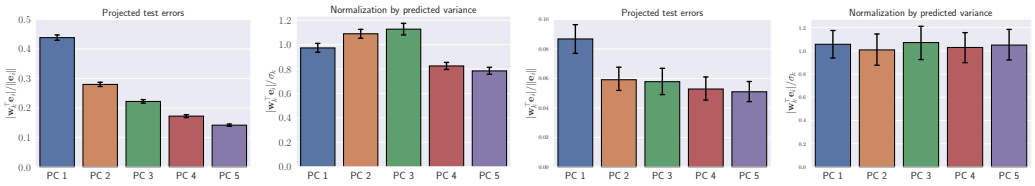
(b) $K = 5$ vs $K = 10$.

Figure A9: PC consistency as a function of K in image colorization. (a) Sequential and end-to-end PC training are compared using 5 models outputting an increasing number of PCs $K = 1, \ldots, 5$. Each row presents K PCs predicted by a model with K outputs applied to the inputs at the top right. The PCs were backward consistent as judged by their absolute cosine similarity over the entire test set (see boxplots at the bottom). (b) The results were similar when comparing the first 5 PCs of two models with $K = 5/10$ outputs, with the last 2 PCs being slightly less consistent due to the remaining PCs having roughly the same variance.

the PCs indirectly using the residual error magnitude $\|\boldsymbol{e} - \boldsymbol{W}\boldsymbol{W}^\top \boldsymbol{e}\|_2$, which is a function of the subspace spanned by the PCs $\boldsymbol{W}$. Similarly, for the k^{th} predicted variance σ_k^2, we only have the norm of a single projected error $|\boldsymbol{w}_k^\top \boldsymbol{e}_i|$ per measurement $\boldsymbol{y}_i$, and hence no ground truth variance either. Please note that, unlike per-pixel methods, in our case, we cannot compare the aleatoric uncertainty and the test error directly (e.g. RMSE vs. fraction of pixels above an uncertainty threshold), because our method does not assume pixels are independent (an incorrect assumption in images). Nonetheless, we further verified the predicted variances by comparing the projected test error $\boldsymbol{w}_k^\top \boldsymbol{e}_i$ to the predicted variance σ_k^2 for every test point $\boldsymbol{y}_i$ (see Fig. A10). The results indicate that NPPC estimates the standard deviation with high accuracy across tasks and datasets.

D.4 More examples

Here we provide more results on each of the tasks demonstrated in Section 4.



(a) (b) (c) (d)

Figure A10: Predicted variance validation. The PC ordering/predicted variances are validated using the projected test errors $|\mathbf{w}_k^\top \mathbf{e}_i|/\|\mathbf{e}_i\|$, and their normalization by the predicted standard deviation $|\mathbf{w}_k^\top \mathbf{e}_i|/\sigma_k$, both for image colorization on CelebA-HQ (a)-(b) and for image denoising on MNIST (c)-(d). As can be seen in (a) and (c) the PCs are correctly ordered from 1 to 5. Furthermore, (b) and (d) show that the predicted variances are accurate with an average standard deviation of 0.96 for CelebA-HQ and 0.99 for MNIST, where 1 is the ground truth.

Handwritten digits Figure A11 demonstrates more denoising examples for handwritten digits. At severe noise levels of $\sigma_\varepsilon = 1.0$, the mean reconstruction is often ambiguous with regards to digit identity (*e.g.,* a "7" can become a "9" as in the upper left panel). Similarly, for extreme inpainting of 70% of the pixels in Fig. A12, a "5" can become a "3" (top right), a "4" can become a "1" (bottom left), etc.

Faces Figures A13 shows additional examples for the task of noisy $8\times$ super-resolution. The resulting PCs capture different semantic properties such as eye/mouth shape, eyebrows position, and chin/jawline placement. Figures A14 and A15 show additional results for inpainting of the eyes and the mouth, respectively. The corresponding PCs manipulate the content in the missing pixels such that the eyes (Fig. A14) are more open/closed, the eyebrows are thicker/thinner etc. Similarly, for mouth inpainting (Fig. A15), the mouth could be open/closed, the jaw could be lower/higher, and the neck area is manipulated to be wider/thinner, etc.

Biological image-to-image translation Figures A16 and A17 show additional results on the biological image-to-image translation task. Note that the first PC is essentially a bias component, suggesting that the absolute intensity of the nuclear stains is uncertain and cannot be inferred accurately using the measurement $\boldsymbol{y}$. The remaining PCs capture more semantic content by highlighting uncertain cell shapes *e.g.,* in the case of dividing (small ellipsoidal) cells and navigating the existence/disappearance of ambiguous cells.

References

[1] Anastasios N Angelopoulos, Amit P Kohli, Stephen Bates, Michael I Jordan, Jitendra Malik, Thayer Alshaabi, Srigokul Upadhyayula, and Yaniv Romano. Image-to-image regression with distribution-free uncertainty quantification and applications in imaging. In *International Conference on Machine Learning*, 2022.

[2] Thorsten Falk, Dominic Mai, Robert Bensch, Özgün Çiçek, Ahmed Abdulkadir, Yassine Marrakchi, Anton Böhm, Jan Deubner, Zoe Jäckel, Katharina Seiwald, et al. U-net: deep learning for cell counting, detection, and morphometry. *Nature methods*, 16(1):67–70, 2019.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[4] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems*, 2022.

[5] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 3, 2015.

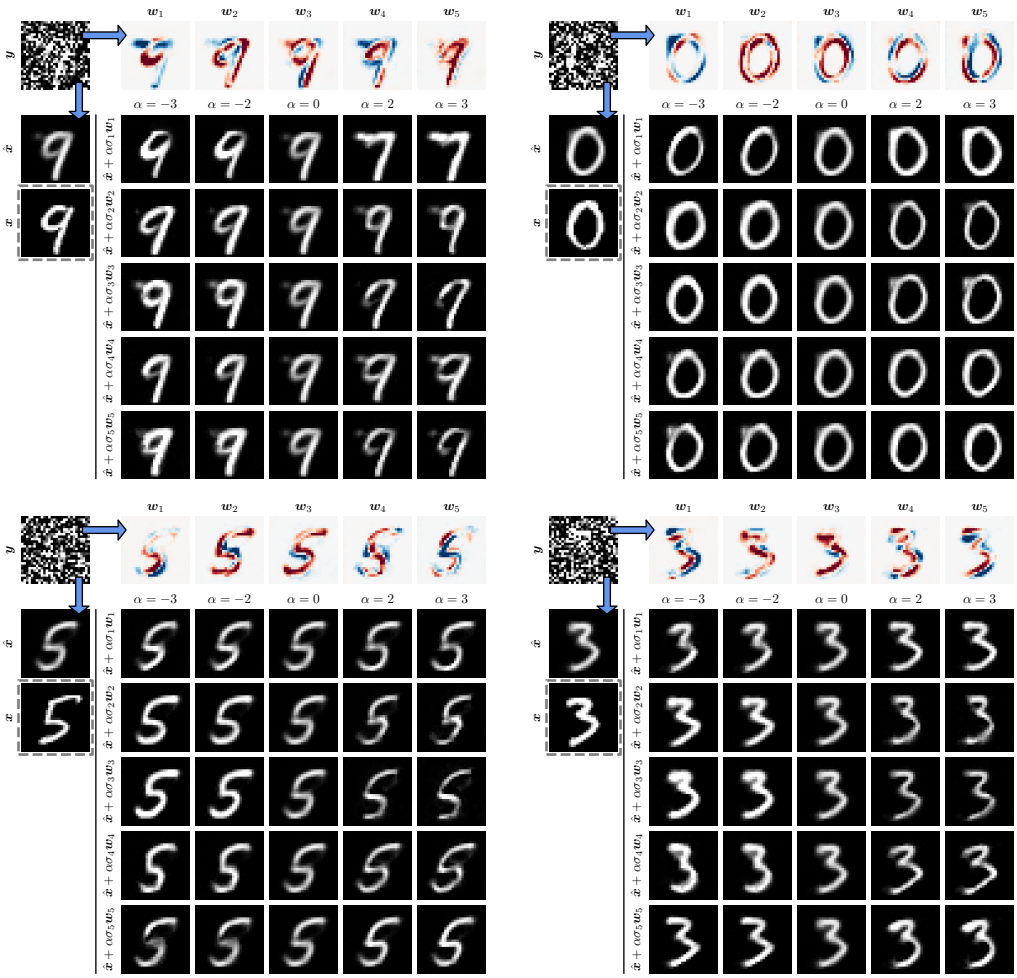


Figure A11: **MNIST denoising**.

[7] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 577–593. Springer, 2016.

[8] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[9] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[10] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[12] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

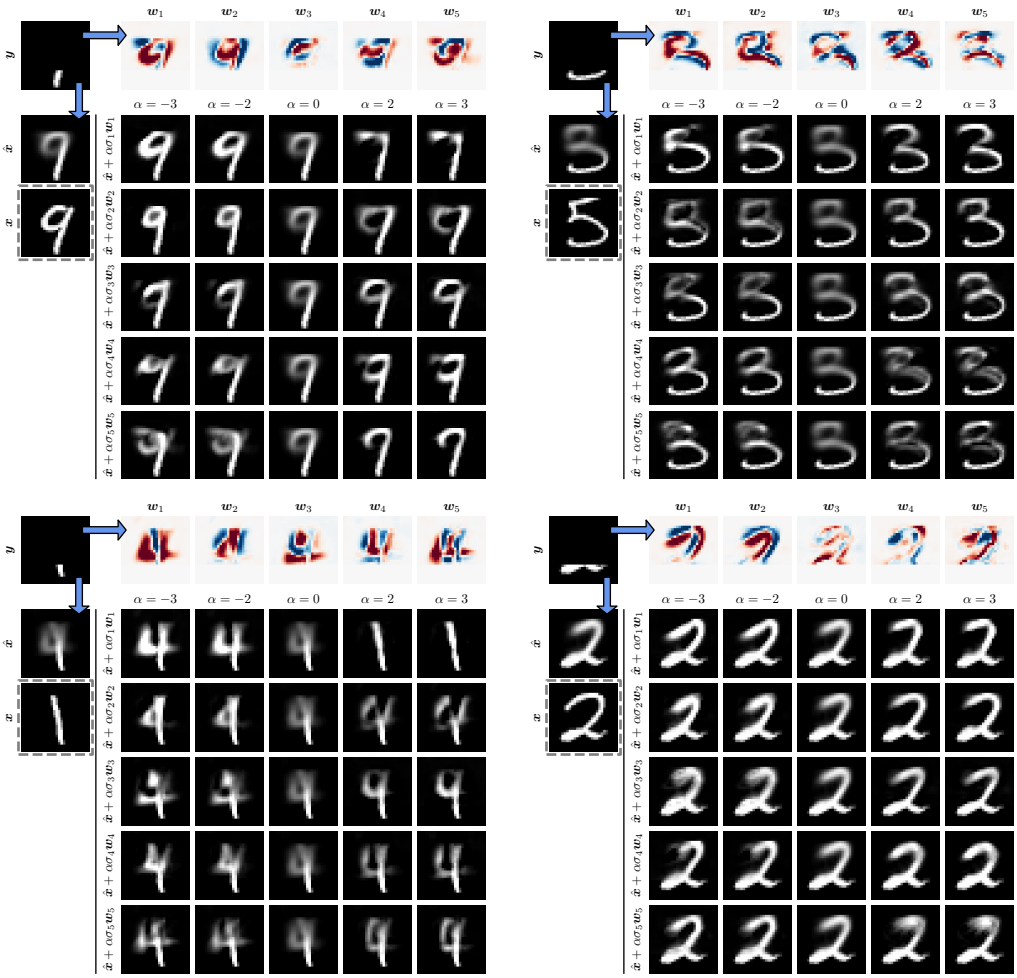


Figure A12: **MNIST inpainting**.

[13] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016.

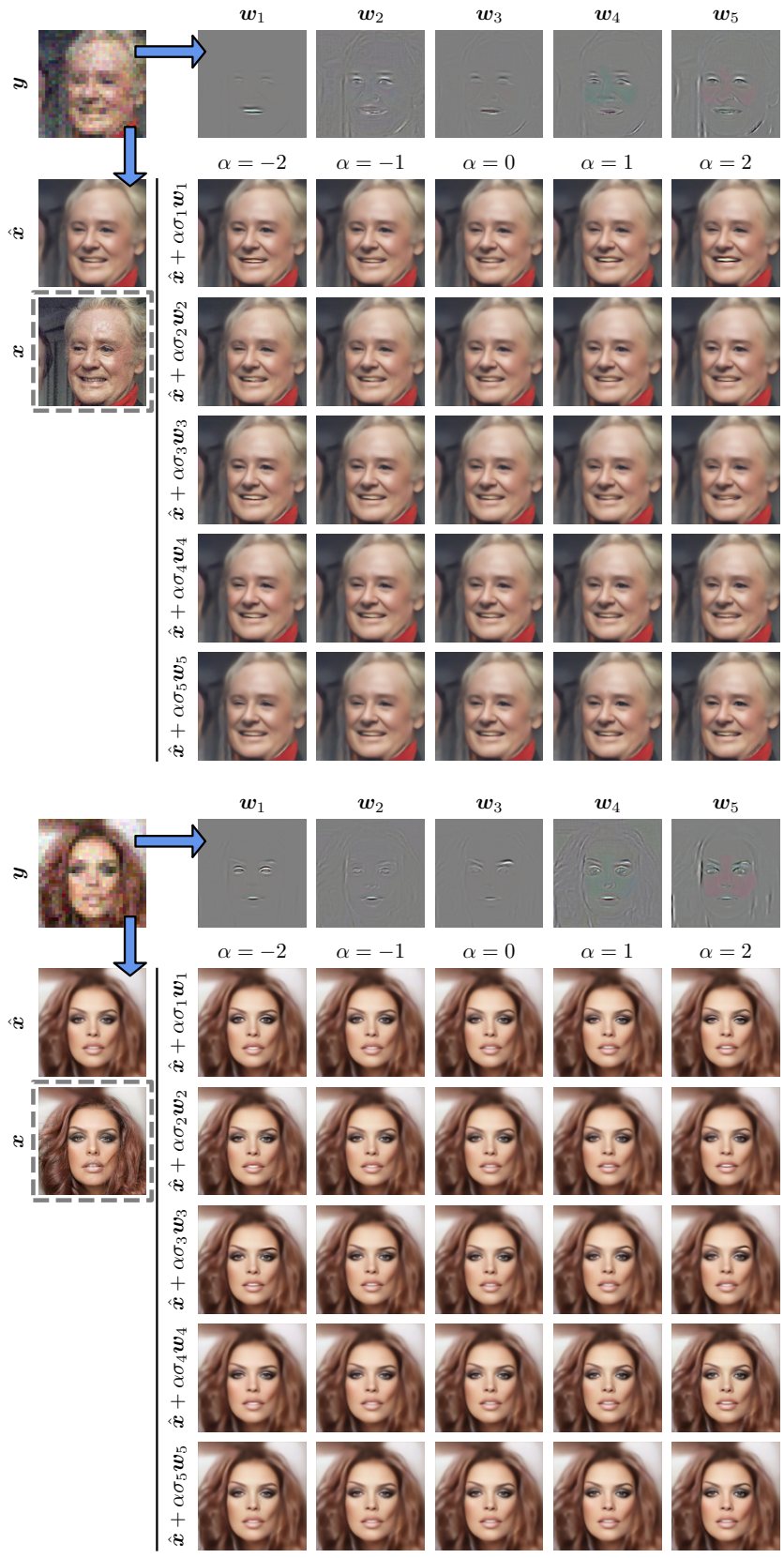


Figure A13: **CelebA-HQ noisy** $8\times$ **super-resolution**.

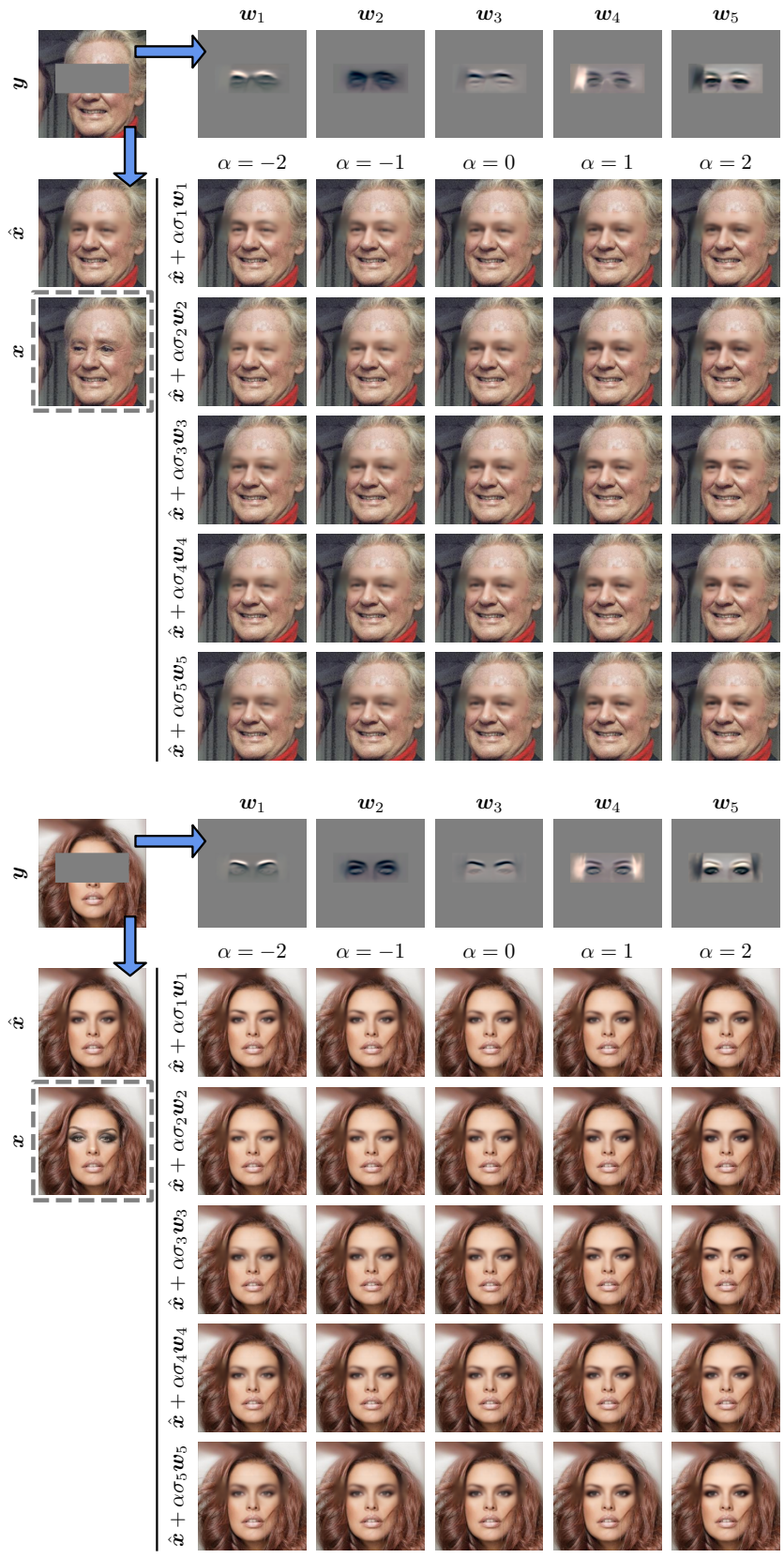


Figure A14: **CelebA-HQ eyes inpainting**.

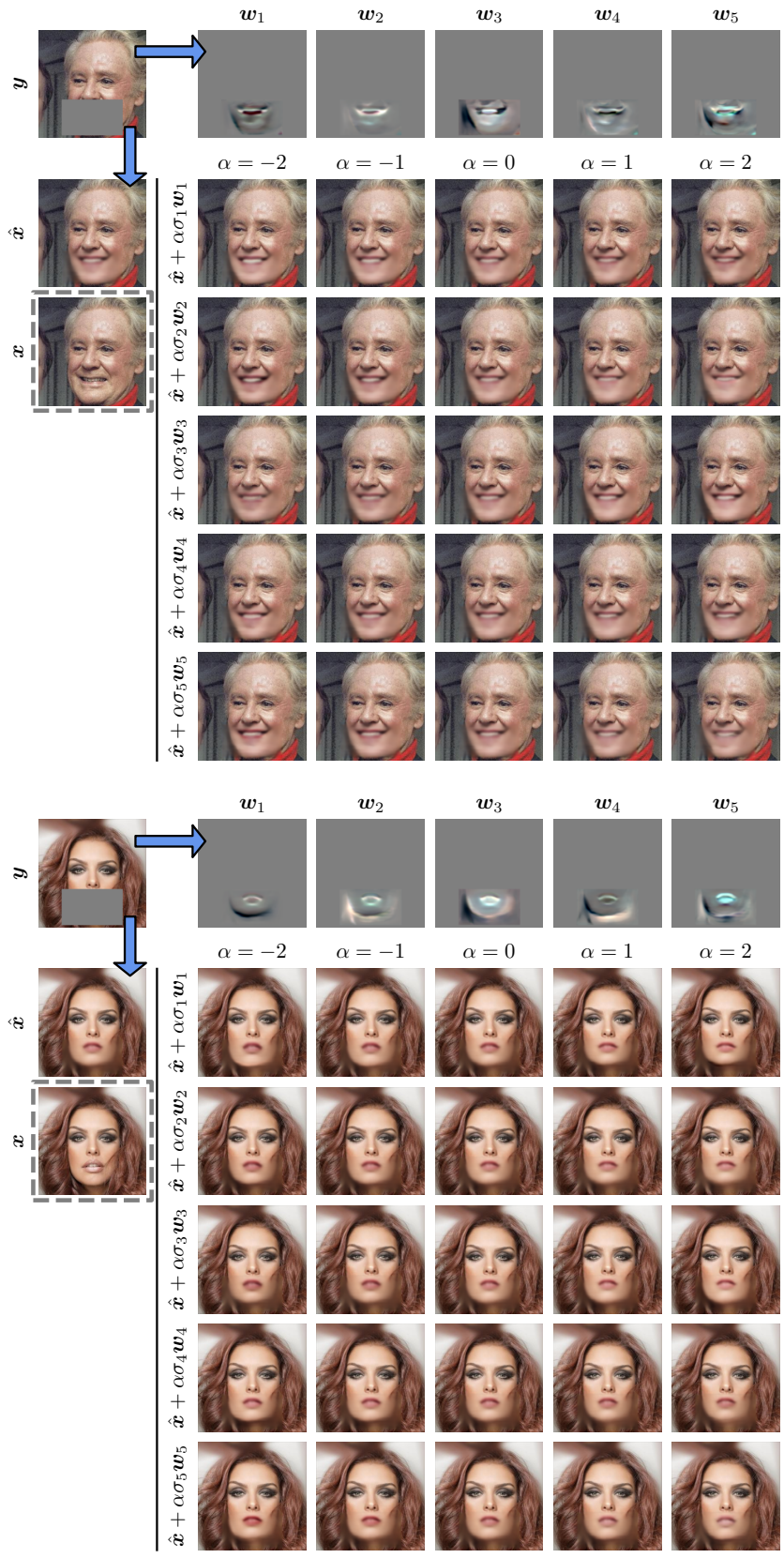


Figure A15: **CelebA-HQ mouth inpainting**.

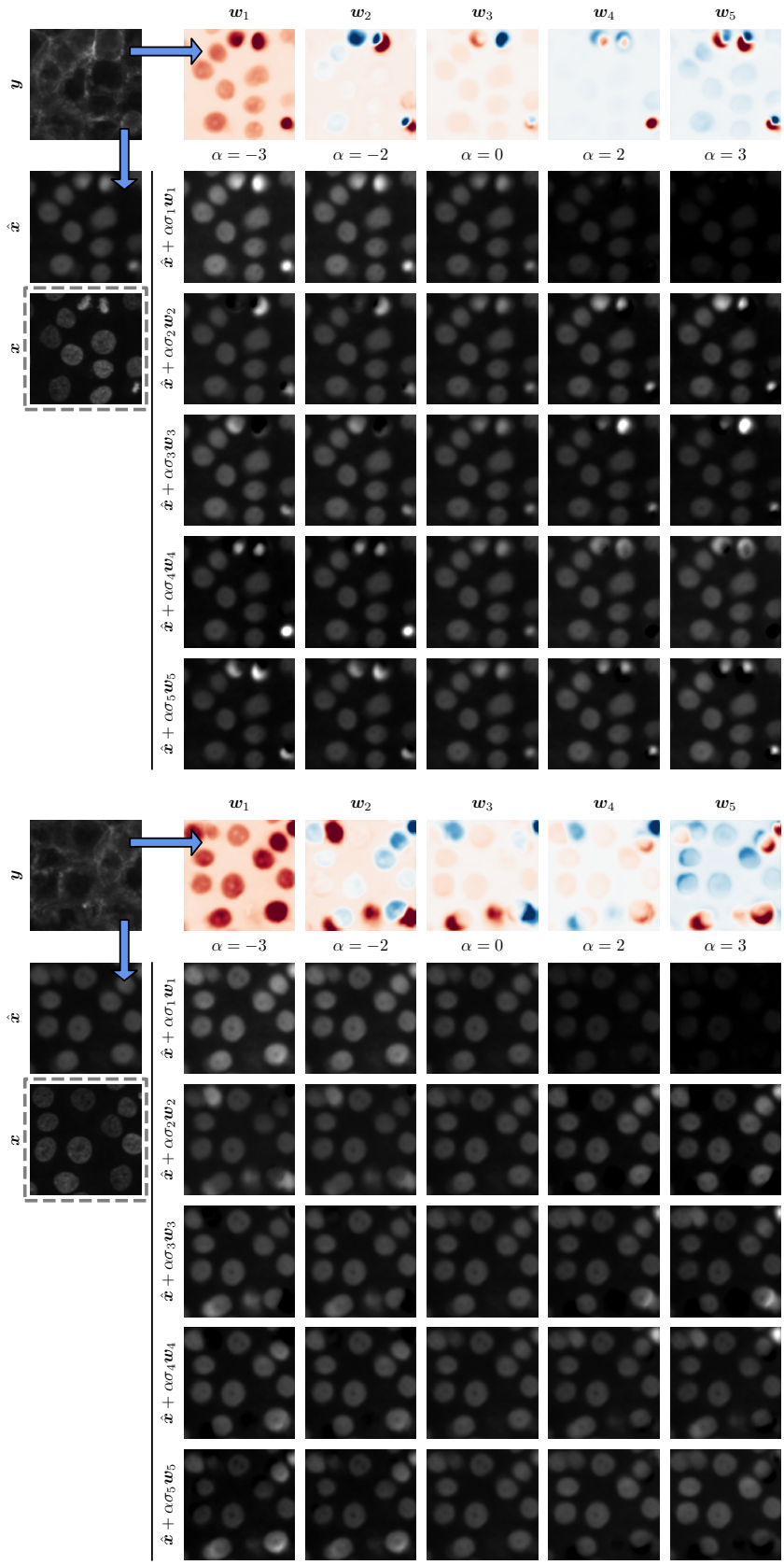


Figure A16: **Biological image-to-image translation results**.

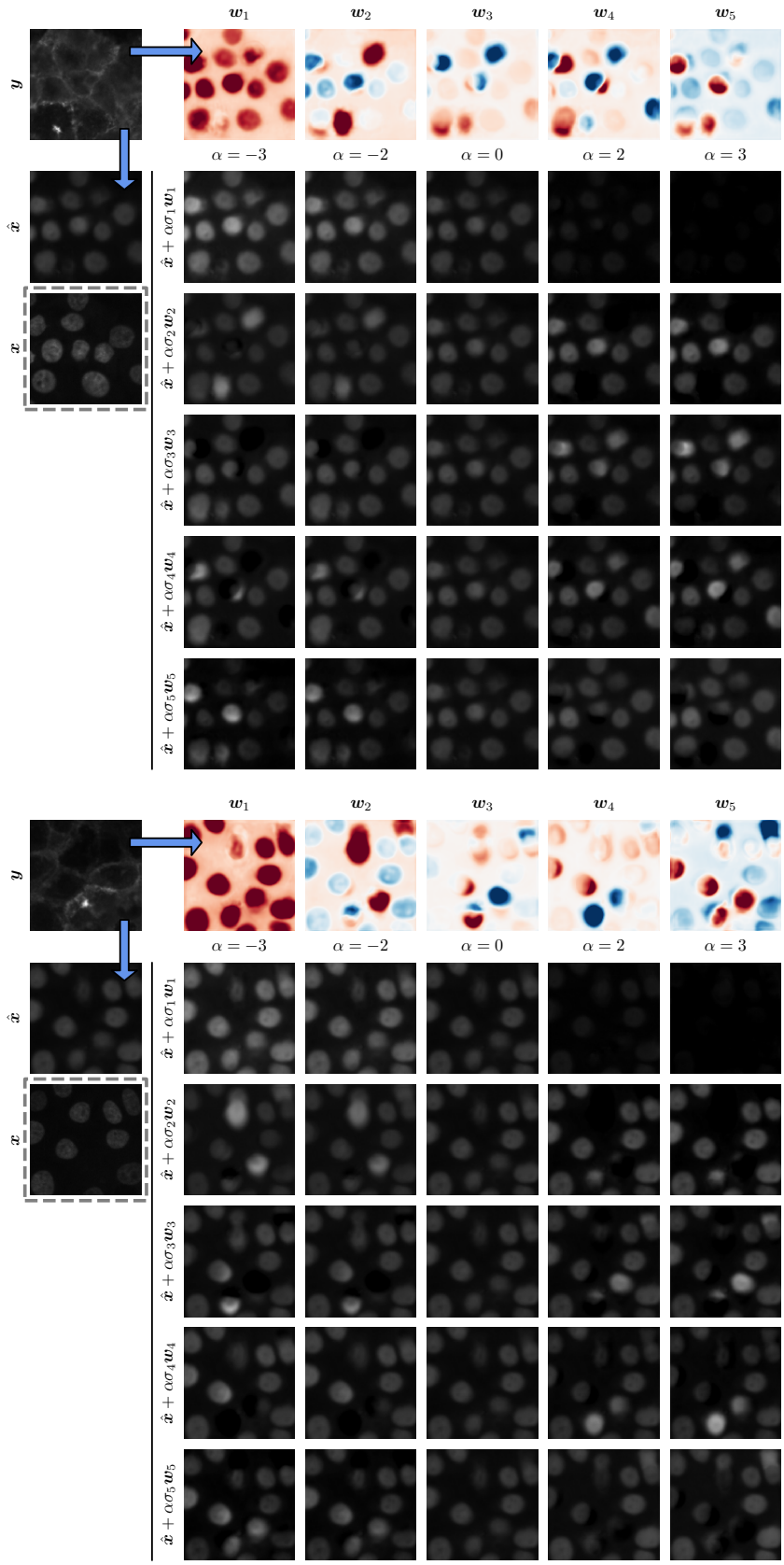


Figure A17: **More biological image-to-image translation results**.