IMAGE RESTORATION VIA EFFICIENT GAUSSIAN MIXTURE MODEL LEARNING

Jianzhou Feng[⋆] *Li Song* [⋆] *Xiaoming Huo*[†] *Xiaokang Yang* [⋆] *Wenjun Zhang* [⋆]

[⋆] Shanghai Digital Media Processing and Transmission Key Lab, Shanghai Jiaotong University
[†] School of Industrial and Systems Engineering, Georgia Institute of Technology

ABSTRACT

Expected Patch Log Likelihood (EPLL) framework using *Gaussian Mixture Model* (GMM) prior for image restoration was recently proposed with its performance comparable to the state-of-the-art algorithms. However, EPLL uses generic prior trained from offline image patches, which may not correctly represent statistics of the current image patches. In this paper, we extend the EPLL framework to an adaptive one, named A-EPLL, which not only concerns the likelihood of restored patches, but also trains the GMM to fit for the degraded image. To efficiently estimate GMM parameters in A-EPLL framework, we improve a recent *Expectation- Maximization* (EM) algorithm by exploiting specific structures of GMM from image patches, like Gaussian Scale Models. Experiment results show that A-EPLL outperforms the original EPLL significantly on several image restoration problems, like inpainting, denoising and deblurring.

Index Terms— Image restoration, Expected patch log likelihood, Gaussian mixture model

1. INTRODUCTION

Nowadays, most image restoration algorithms use patch based prior since: (1) The high dimension of images makes learning, inference and optimization the prior for the whole image extremely hard. (2) Image patches are easy to model, and still contain local abundant structures, such as textures and edge patterns. The key issues that affect the patched based restoration algorithms are the accuracy of the used patch priors and how they are used for regularizing the whole image. For the first issue, priors adapt to the degraded images are shown to be better than the general priors in [1, 2, 3]. For the second issue, the recently proposed *Expected Patch Log Likelihood* (EPLL)[4] is verified to be better for regularization. This regularization process restore the patches based on both the given patch prior and the overlapping information of different patches. However, no state-of-the-art algorithms take advantage of both issues, either doesn't have adaptive priors like EPLL, or don't use the better regularization term, such as the ones in [1, 2, 3].

In this paper, we extend the EPLL framework to an adaptive one, named A-EPLL, which learns the online statistics from the degraded images, and retains the good regularization structure in EPLL. The *Gaussian Mixture Model* (GMM) is used in A-EPLL due to its good performance in image restoration tasks [4]. We propose an improved GMM learning algorithm to update the general GMM prior during the restoration process. The newly proposed learning algorithm treats a GMM as a hybrid Gaussian Scale Model, with each GSM modeling a certain pattern in image patches (texture or edge). Experiment results show that A-EPLL outperform the original EPLL significantly for image inpainting, denoising and deblurring tasks.

2. FROM EPLL TO A-EPLL

The general form of an image restoration task is $\mathbf{Y} = \mathbf{AX} + \varepsilon$, where $\mathbf{X}$ is a clean image, $\mathbf{A}$ is a corruption matrix corresponding to different tasks, ε is an i.i.d Gaussian random noise and $\mathbf{Y}$ is the final observation. We aim to derive a $\hat{\mathbf{X}}$ from $\mathbf{Y}$, the closer to $\mathbf{X}$ the better.

2.1. EPLL framework [4]

Under a prefixed prior p, EPLL aims to solve

$$\min_{\mathbf{X}} \frac{\lambda}{2}\|\mathbf{AX} - \mathbf{Y}\|_2^2 - EPLL_p(\mathbf{X}), \tag{1}$$

where λ is a tuning parameter related to the noise variance and

$$EPLL_p(\mathbf{X}) = \sum_{i=1}^{N} \log p(\mathbf{x}_i) \tag{2}$$

is the sum of log likelihood of all the N patches $\mathbf{x}_i$'s in $\mathbf{X}$. It is hard to solve (1) directly. An alternative optimization method called "Half Quadratic Splitting" can be used. It use the fact that

$$\begin{aligned}\min_{\mathbf{X},\{\mathbf{z}_i\}} &\frac{\lambda}{2}\|\mathbf{AX} - \mathbf{Y}\|_2^2 \\ &+ \sum_{i=1}^{N}\left(\frac{\beta}{2}\|\mathbf{z}_i - \mathbf{x}_i\|_2^2 - \log p(\mathbf{z}_i)\right)\end{aligned} \tag{3}$$

is equivalent to (1) as $\beta \to \infty$. Thus, we need only to solve (3) under a sequence of fixed β's, which approaches infinity. Algorithm 1 is proposed for iteratively solving (3).

Algorithm 1 One iteration for solving (3).

1: Solving for $\{\mathbf{z}_i\}$ given $\mathbf{X}$: For each i,

$$\mathbf{z}_i = \arg\min_{\mathbf{z}} \frac{\beta}{2}\|\mathbf{z} - \mathbf{P}_i\mathbf{X}\|_2^2 - \log p(\mathbf{z}), \tag{4}$$

where $\mathbf{P}_i$ is the matrix extract the i-th patch from the whole image.

2: Solving for $\mathbf{X}$ given $\{\mathbf{z}_i\}$:

$$\mathbf{X} = \left(\lambda\mathbf{A}^T\mathbf{A} + \beta\sum_{i=1}^{N}\mathbf{P}_i^T\mathbf{P}_i\right)^{-1} \left(\lambda\mathbf{A}^T\mathbf{Y} + \beta\sum_{i=1}^{N}\mathbf{P}_i^T\mathbf{z}_i\right). \tag{5}$$

2.2. A-EPLL framework

Denote the parameter of prior p as θ, we modify (1) to be

$$\min_{\mathbf{X},\theta} \frac{\lambda}{2}\|\mathbf{A}\mathbf{X} - \mathbf{Y}\|_2^2 - EPLL_p(\mathbf{X};\theta), \tag{6}$$

which optimizes the restored image $\mathbf{X}$ and the parameter θ simultaneously. If we know the clean $\mathbf{x}_i$'s, (6) turns to be $\hat{\theta} = \arg\max_\theta EPLL_p(\mathbf{X};\theta)$, where $\hat{\theta}$ is the maximum likelihood (ML) estimation. However, while solving (1), we could only obtain estimated $\mathbf{x}_i = \mathbf{P}_i\mathbf{X}$'s from Algorithm 1. They are much better than $\mathbf{y}_i = \mathbf{P}_i\mathbf{Y}$'s, but still degraded. In (4), $\mathbf{x}_i$'s are assumed to be degraded from a clean random sample $\mathbf{z} \sim p_\theta$ by adding noise $\varepsilon_i \sim \mathcal{N}(\mathbf{0}, \frac{1}{\beta}\mathbf{I}_n)$ and $\mathbf{z}_i$ is the MAP estimation. Thus, we update θ as the one generates noisy $\mathbf{x}_i$'s with the maximum probability

$$\hat{\theta} = \arg\max_\theta \sum_{i=1}^{N} \log p(\mathbf{x}_i; \frac{1}{\beta}\mathbf{I}_n, \theta). \tag{7}$$

It is clear that when $\beta \to \infty$, (7) also approaches to the ML estimation under the clean case. Thus, we simply extend EPLL to update the prior parameters by solving (7) at the beginning of Algorithm 1 using $\mathbf{X}$ from last iteration.

3. GMM LEARNING ALGORITHM

In [4], GMM is verified to outperform other priors for image restoration, so we propose an efficient GMM learning algorithm in this section and apply it to A-EPLL.

3.1. GMM prior

GMM describes local image patches with a mixture of Gaussian distributions. The parameter set of GMM is $\theta = \{\pi_k, \mu_k, \Sigma_k\}_{k=1}^K$. It assumes there are K components, each component C_k conforms to a Gaussian distribution $\mathcal{N}(\mu_k, \Sigma_k)$ and patch $\mathbf{x} \in \mathbb{R}^n$ is randomly generated from one of the K components, with probability π_k for C_k. Thus, the *probability density function* (pdf) $p(\mathbf{x};\theta)$ is

$$p(\mathbf{x};\theta) = \sum_{k=1}^{K} \pi_k \phi(\mathbf{x};\mu_k,\Sigma_k), \tag{8}$$

where

$$\phi(\mathbf{x};\mu_k,\Sigma_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\mu_k)^T\Sigma_k^{-1}(\mathbf{x}-\mu_k)\right)}{(2\pi)^{n/2}|\Sigma_k|^{1/2}} \tag{9}$$

is the Gaussian distribution pdf of component k. From the definition of π_k's, we can see

$$\sum_{k=1}^{K} \pi_k = 1. \tag{10}$$

The GMM learned from image patches has some special properties in addition to the aforementioned definition. It could be interpreted as a mixture of several GSMs and each GSM is used for modeling patches contain the same texture or edge pattern. This interpretation originates from the structured dictionary interpretation. In [4] and [5], the GMM is interpreted as a structured dictionary, which is composed by K sub-dictionaries. For the k-th sub-dictionary, its atoms are the top eigenvectors of Σ_k. Thus, each patch can be well sparsely represented by atoms from its corresponding sub-dictionary. However the structured dictionary interpretation doesn't consider the special situation that more than one Gaussian components share the same top eigenvectors with different scale on their eigenvalues, i.e. GSM. For natural image patches, it is shown in [6] and [7] that GSM is necessary and more suitable for modeling sharp distributions. Therefore, the properties that a patch based GMM should have are structured dictionary and multi-scale components for each sub-dictionary. These properties help us to understand the learned GMM more deeply as well as speed up GMM learning algorithms designed for general cases (e.g. the fast implementation in Section 3.2).

3.2. The efficient GMM learning algorithm

For solving (7) under GMM prior in A-EPLL, the uncertainty based EM algorithm [8] recently proposed (as listed in Algorithm 2) is a good candidate. However, the original algorithm is rather time consuming, which is unsuitable for being used frequently. Thus we propose a speed-up version for it. This implementation contains four techniques, the first two exploit the properties of patch based GMM as analyzed in Section 3.1 and the last two are general ones for all kinds of GMM.

1. **Utilizing the scale property:** Generally, for components with small scales, their mean vector μ_k's are also

Algorithm 2 One iteration of the uncertainty EM algorithm. $\{\mathbf{x}_i\}_{i=1}^N$ is the noisy data set and $\sigma^2 = 1/\beta$ is the noise level.

1: **E-step.** Conditional expectations of natural statistics:

$$\gamma_{k,i} \propto \pi_k \phi(\mathbf{x}_i; \mu_k, \Sigma_k + \sigma^2 \mathbf{I}_n), \tag{11}$$

and

$$\sum_{k=1}^{K} \gamma_{k,i} = 1, \tag{12}$$

$$\hat{\mathbf{x}}_{k,i} = \mathbf{W}_k(\mathbf{x}_i - \mu_k) + \mu_k, \tag{13}$$

$$\hat{\mathbf{R}}_{\mathbf{xx},k,i} = \hat{\mathbf{x}}_{k,i}\hat{\mathbf{x}}_{k,i}^T + (\mathbf{I}_n - \mathbf{W}_k)\Sigma_k, \tag{14}$$

where

$$\mathbf{W}_k = \Sigma_k(\Sigma_k + \sigma^2 \mathbf{I}_n)^{-1}. \tag{15}$$

2: **M-step.** Update GMM parameter set:

$$\hat{\pi}_k = \frac{1}{N}\sum_{i=1}^{N} \gamma_{k,i}, \tag{16}$$

$$\hat{\mu}_k = \frac{1}{\sum_{i=1}^N \gamma_{k,i}} \sum_{i=1}^{N} \gamma_{k,i}\hat{\mathbf{x}}_{k,i}, \tag{17}$$

$$\hat{\Sigma}_k = \frac{1}{\sum_{i=1}^N \gamma_{k,i}} \sum_{i=1}^{N} \gamma_{k,i}\hat{\mathbf{R}}_{\mathbf{xx},k,i} - \hat{\mu}_k\hat{\mu}_k^T. \tag{18}$$

small and close to $\mathbf{0}$. It implies these components can not be distinguished under high noise levels so that their μ_k's and Σ_k's need not be updated. These components can be found by $\Gamma = \{k \big| \|\Sigma_k\|_2 < C_1\sigma\}$, C_1 as the threshold. When the noise level is extremely large, all components belong to Γ and no one is updated, which is quite reasonable and time saving.

2. **Utilizing the structured dictionary property:** For patch i, most $\gamma_{k,i}$'s are very close to 0, only components corresponding to the correct sub-dictionary may have large $\gamma_{k,i}$'s. So we can add a hard thresholding process between (11) and (12), which set

$$\gamma_{k,i} < C_2 \max_{1 \le j \le K} \gamma_{j,i}$$

to be 0, where C_2 is a threshold no larger than 1.

3. **Seldomly used components are eliminated.** During the M-step, if $\hat{\pi}_k < C_3/N$, where C_3 restricts the minimum number of samples for updating a component, then component k is eliminated from the GMM. This process helps to reduce the component number K.

4. **Wiener filtering are applied together.** Instead of apply Wiener filtering to all $\mathbf{x}_i$'s to compute $\hat{\mathbf{x}}_{k,i}$'s and $\hat{\mathbf{R}}_{\mathbf{xx},k,i}$'s, we update $\hat{\mu}_k$ and $\hat{\Sigma}_k$ by preprocessing $\mathbf{x}_i$'s and apply Wiener filtering at last as

$$\hat{\mu}_k = \mathbf{W}_k\bar{\mathbf{x}}_k + (\mathbf{I}_n - \mathbf{W}_k)\mu_k, \tag{19}$$

and

$$\hat{\Sigma}_k = \mathbf{W}_k(\bar{\mathbf{R}}_{\mathbf{xx},k} - \bar{\mathbf{x}}_k\bar{\mathbf{x}}_k^T)\mathbf{W}_k + (\mathbf{I}_n - \mathbf{W}_k)\Sigma_k, \tag{20}$$

where

$$\bar{\mathbf{x}}_k = \frac{\sum_{i=1}^N \gamma_{k,i}\mathbf{x}_i}{\sum_{i=1}^N \gamma_{k,i}}, \tag{21}$$

and

$$\bar{\mathbf{R}}_{\mathbf{xx},k} = \frac{\sum_{i=1}^N \gamma_{k,i}\mathbf{x}_i\mathbf{x}_i^T}{\sum_{i=1}^N \gamma_{k,i}}. \tag{22}$$

This modification dramatically reduce the number of multiplications related to $\mathbf{W}_k$'s.

GMM learning experiment shows that the efficient version reduce the running time significantly without losing any restoration performance.

It should be noted that *Piecewise Linear Estimators* (PLE) [5] is another restoration algorithm with online GMM learning. However, it is based on direct EM learning method, which include the corruption matrix $\mathbf{A}$ in each iteration and has limitation for some restoration tasks, as stated in [5], it could not be applied for image deblurring with wide blur kernels.

4. EXPERIMENTAL RESULTS

The initial GMM is the one trained in [4] and parameters $C_1 = 0.5, C_2 = 0.1, C_3 = 20$ are fixed through all the experiments. Since different initialization and settings for EPLL are implemented in inpainting and deblurring, we denote EPLL [4] as the original one and EPLL* as the new implementation, equivalent to A-EPLL without GMM learning.

4.1. Image Inpainting

In [4], the authors didn't present the inpainting results and their source code online only implement inpainting by EPLL+ICA. Therefore we design the inpainting process as: (1) Restore each patch $\mathbf{x}_i = \arg\max_x p(\mathbf{U}_i\mathbf{x}|\mathbf{y}_i;\theta)$, where $\mathbf{U}_i$ is the random mask and $\mathbf{y}_i$ contains the remaining pixel values. (2) Initialize each pixel in $\mathbf{X}$ as the averaged value from the restored patches. (3) Only set a sufficient large value for β, since the initialized $\mathbf{X}$ is already a good estimation of the hidden image. We compare NL [9], EPLL, EPLL* and A-EPLL under uniform random masks with two data ratios. As listed in Table 1, EPLL* enhances EPLL by 1dB and A-EPLL outperforms EPLL* and the state-of-the-art algorithm NL significantly under various data ratios. The inpainting results in [5] show that NL is better than PLE under data ratio 80%, which means A-EPLL also outperforms PLE in this case.

Table 1. PSNR(dB) results of the inpainted images. For each image, uniform random masks with two data ratios are tested.

Data ratio		NL	EPLL	EPLL*	A-EPLL
Parrot	80%	35.68	35.86	36.11	**36.50**
	30%	26.08	26.31	26.99	**27.21**
House	80%	45.09	42.94	44.63	**46.23**
	30%	36.02	33.45	34.73	**36.80**
C.man	80%	36.28	36.14	36.43	**37.10**
	30%	26.42	26.44	27.10	**27.52**
Monarch	80%	38.15	38.44	38.79	**39.79**
	30%	28.55	27.54	28.46	**29.19**
Straw	80%	35.39	34.59	35.38	**36.37**
	30%	25.07	24.02	25.40	**26.63**
Average	80%	38.12	37.59	38.27	**39.20**
	30%	28.43	27.55	28.54	**29.47**

4.2. Image Denoising

The same as in [4], we initialize $\mathbf{X}$ as the noisy observation $\mathbf{Y}$ and set increasing β's. Under three noise levels, the PSNR values in Table 2 show that the embedded learning process lead to 0.25dB improvement in average. For images contain few structures like Straw, the improvements are the most significant. A-EPLL also outperforms the BM3D [2] algorithm in most cases.

Table 2. PSNR(dB) results of the denoised images.

Image		BM3D	EPLL	A-EPLL
Parrot	$\sigma = 5$	37.86	37.89	**37.95**
	$\sigma = 25$	28.91	29.00	**29.10**
	$\sigma = 50$	25.86	25.85	**26.04**
House	$\sigma = 5$	**39.82**	39.07	39.80
	$\sigma = 25$	**32.84**	32.30	32.72
	$\sigma = 50$	29.70	29.39	**29.71**
C.man	$\sigma = 5$	38.30	38.22	**38.33**
	$\sigma = 25$	29.44	29.22	**29.45**
	$\sigma = 50$	26.07	26.01	**26.35**
Monarch	$\sigma = 5$	38.19	38.41	**38.60**
	$\sigma = 25$	29.37	29.61	**29.81**
	$\sigma = 50$	25.76	25.93	**26.18**
Straw	$\sigma = 5$	35.39	35.45	**35.64**
	$\sigma = 25$	25.91	25.86	**26.24**
	$\sigma = 50$	22.47	21.72	**22.68**
Average	$\sigma = 5$	37.91	37.81	**38.06**
	$\sigma = 25$	29.29	29.20	**29.47**
	$\sigma = 50$	25.97	25.78	**26.19**

4.3. Image Deblurring

We initialize $\mathbf{X}$ by the deblurring result of BM3D [10] and adjust β and λ to achieve better results. The PSNR values in Table 3 show that EPLL* improve its initialization BM3D [10] a lot and the gain of GMM learning is more than 0.15dB. PLE [5] can not handle the 9×9 kernel since it is too wide.

Table 3. PSNR(dB) results of the deblurred images.

9×9 uniform blur		BM3D	EPLL	EPLL*	A-EPLL
Parrot	$\sigma = \sqrt{2}$	27.21	26.42	27.96	**28.06**
	$\sigma = 2$	26.50	25.47	27.13	**27.22**
House	$\sigma = \sqrt{2}$	32.95	32.55	33.85	**34.19**
	$\sigma = 2$	32.30	31.68	33.07	**33.42**
C.man	$\sigma = \sqrt{2}$	27.30	26.97	28.02	**28.10**
	$\sigma = 2$	26.65	26.13	27.21	**27.29**
Monarch	$\sigma = \sqrt{2}$	27.22	27.31	28.69	**28.81**
	$\sigma = 2$	26.51	26.34	27.87	**27.94**
Straw	$\sigma = \sqrt{2}$	22.65	21.59	22.46	**22.75**
	$\sigma = 2$	**22.02**	21.03	21.79	22.00
Average	$\sigma = \sqrt{2}$	27.47	26.97	28.20	**28.38**
	$\sigma = 2$	26.80	26.13	27.41	**27.57**

5. CONCLUSIONS

In this paper, we propose the A-EPLL framework for image restoration. This new framework contains an efficient GMM learning algorithm, which lead to better prior than the predefined one. By further optimizing the initialization and parameter settings, the performance under different tasks are enhanced significantly. Future work includes regularizing the whole image prior based on patch prior other than EPLL, for example, further restricting spatially close patches to share the same GMM component.

6. ACKNOWLEDGEMENT

This work was supported in part by 973 Program (2010CB731 401, 2010CB731406), 863 project (2012AA011703), NSFC (61221001), the 111 Project (B07022) and the Shanghai Key Laboratory of Digital Media Processing and Transmissions.

7. REFERENCES

[1] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006.

[2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[3] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration,"

in *Proc. IEEE Int. Conf. Computer Vision*, 2009, pp. 2272–2279.

[4] Daniel Zoran and Yair Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. IEEE Int. Conf. Computer Vision*, 2011, pp. 479–486.

[5] G. Yu, G. Sapiro, and S. Mallat, "Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity," *IEEE Trans. Image Process.*, pp. 2481–2499, 2012.

[6] Y. Weiss and W. T. Freeman, "What makes a good model of natural images?," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2007.

[7] Daniel Zoran and Yair Weiss, "Natural images, gaussian mixtures and dead leaves," in *Neural Information Processing Systems*, 2012.

[8] A. Ozerov, M. Lagrange, and E. Vincent, "Uncertainty-based learning of gaussian mixture models from noisy data," in *Inria Reserch report No 7862*, 2012.

[9] X. Li, "Patch-based image interpolation: Algorithms and applications," in *Int. Workshop Local Non-Local Approx.Image Process.*, 2008.

[10] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image restoration by sparse 3d transform-domain collaborative filtering," in *Proc. SPIE Electron. Imag.*, 2008.