Exercise 2: Half Quadratic Splitting and Contrastive Divergence Learning (Due 3/5/2017)*

Statistical Methods in Image Processing 048926

Half Quadratic Splitting (60 points)

As we saw in class, using the pairwise-cliques model with potential $\rho(\cdot)$, the MAP estimate of an image $x$ from its degraded version $y = Hx + n$ can be computed by solving

$$\arg \min_{x,\{z_{i,j}\}} \frac{1}{2\sigma^2} \|y - Hx\|^2 + \sum_f \sum_{(i,j) \in c} b_{ij} \rho(z^f_{ij}) + \frac{\beta}{2} \sum_f \sum_{\{i,j\} \in C} \|z^f_{ij} - (d^f \ast x)_{ij}\|^2,$$

where $\{z_{i,j}\}$ are auxiliary variables and $f$ represents the filter index. In our case, $h$ stand for horizontal derivatives and $v$ for vertical derivatives ($d^h = [1, -1]$, $d^v = [1, -1]^T$ respectively). This problem is solved iteratively, by alternating between minimizing the objective with respect to $x$ and $z$ while gradually increasing $\beta$.

1. Implement the $x$-step:

   (a) Write the solution of (1) with respect to $x$ while regarding $z$ as fixed.

   (b) Since the solution involves convolutions, it is easier (and faster) to implement it in the frequency domain. Write this step in the frequency domain.

   (c) Implement this $x$ update step. Pay attention: frequency domain operations require additional processes. Pad the image using the matlab function `padarray` using ‘replicate’ mode. Use the function `fftshift` if required. As a sanity check, after returning to the spatial domain, you should get a real valued image. Don’t use `real` or `abs` to suppress non-negligible imaginary components! Make sure your code produces real values to begin with.

*Please send your solutions to Tamar
2. Implement the $z$-step:

(a) Minimizing (1) with respect to $z$ boils down to:

$$\arg \min_{\{z_{i,j}\}} \sum_f \sum_{(i,j) \in c} b_{ij} \rho(z^f_{ij}) + \frac{\beta}{2} \sum_f \sum_{(i,j) \in c} \|z^f_{ij} - (d^f \ast x)_{ij}\|^2,$$

which can be solved separately for each $z_{ij}$. Using the TV prior $\rho(\Delta) = |\Delta|/\sigma_x$, write the solution for each $z_{ij}$.

(b) Implement this $z$ update step by updating each $z^f_{ij}$ independently.

3. Implement the full HQS scheme by iterating between updating $z$, updating $x$ and increasing $\beta$.

4. Show the results of the HQS optimization scheme in the following settings. Use the two images in the supplementary zip file. Compare the results in terms of PSNR

$$\text{PSNR}(x, y) = 10 \log_{10} \left( \frac{255^2}{\sum_i (x_i - y_i)^2} \right).$$

Use the estimation for $\hat{\sigma}_x$ from the clean images using the function from HW1.

(a) Denoising: show the denoising results using the TV prior for $\sigma = 15, 25, 35$.

(b) Deblurring: show the deblurring results using the TV prior for the given blur kernels and with a noise variance of $\sigma_n = 2.5$.

5. Discuss and compare the differences between the Majorization Minimization scheme of HW1 and Half Quadratic Splitting.

**Contrastive Divergence Learning (40 points)**

1. Given a set of realizations $\{I\}$ (training examples) drawn from the product of Student-$t$ (PoT) distributions

$$p(x; \theta) = \frac{1}{Z(\theta)} \prod_{n=1}^N \left(1 + \frac{1}{2}(J_n^T x)^2\right)^{-\alpha_n},$$

we would like to estimate the parameters $\theta = \{J_n, \alpha_n\}$. Assuming we know how to draw samples from this model, write a closed form expression for the $\theta$ update step in the Contrastive Divergence algorithm.

2. Matlab Implementation

Consider the distribution

$$p(x; \{\mu_i\}) = \sum_{i=1}^N \frac{1}{N} \frac{1}{2 \pi} \exp \left\{ -\frac{1}{2} ||x - \mu_i||^2 \right\},$$

where $x, \mu_i \in \mathbb{R}^2$. 
(a) Initialization

i. Describe how to draw a sample \(x\) from \(p(x; \{\mu_i\})\) given \(\{\mu_i\}\). Write a function that accepts \(\{\mu_i\}\), and returns a sample \(x\) from \(p(x; \{\mu_i\})\).

ii. Use \(N = 4\) and \(\mu = \{(0,0)^T, (0,2)^T, (2,0)^T, (2,2)^T\}\) and draw 1000 samples \(x\) from \(p(x; \{\mu_i\})\).

(b) Contrastive divergence estimation of \(\{\mu_i\}\)

**From now on we will refer to \(\{\mu_i\}\) as unknowns** and we will estimate them using the contrastive divergence algorithm.

i. Randomly select an initial guess \(\{\tilde{\mu}_i\}\).

ii. Draw 1000 samples \(\tilde{x}\) from \(p(x; \{\mu_i\})\) using \(\{\tilde{\mu}_i\}\).

iii. Update \(\{\tilde{\mu}_i\}\) using the gradient descent step:

\[
\tilde{\mu}_i^{k+1} = \tilde{\mu}_i^k + \eta \left( \langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_x - \langle \nabla_{\mu_i} \log p(x; \{\mu_i\}) \rangle_{\tilde{x}} \right),
\]

where \(\langle \cdot \rangle_x\) denotes averaging over the input samples from 2(a)ii and \(\langle \cdot \rangle_{\tilde{x}}\) denotes averaging over the synthetically generated samples from 2(b)ii.

iv. Repeat 2(b)ii and 2(b)iii until convergence.