Comparison of Learned Inference Approaches for Image Restoration

Jakob Kruse

Master's Thesis Defense, 23. 09. 2016

COMPUTER VISION LAB DRESDEN

supervised by **Uwe Schmidt, Prof. Carsten Rother**
Computer Vision Lab Dresden

Outline

Motivation and Background

- Image Restoration
- Model Parameterization
- Discriminative Training
- Truncated Optimization

Learned Inference Approaches

- Gradient Descent
- Half-quadratic Inference

Experiments

- Denoising
- Deblurring

Conclusion

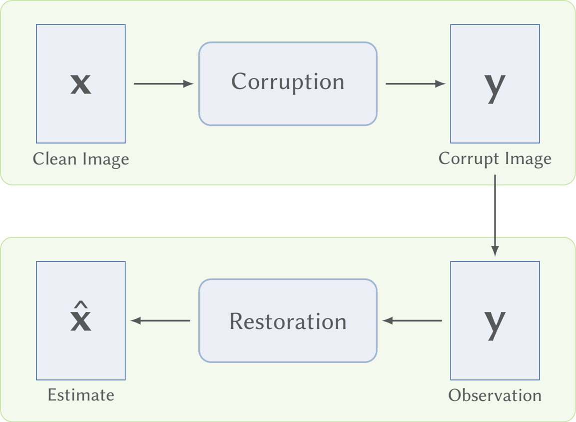
Motivation and Background

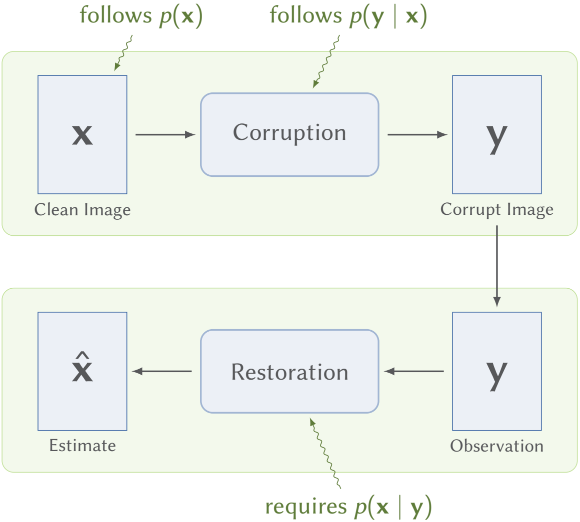
Clean Image

Noisy Image

Blurred Image

Clean image taken from *Berkeley Segmentation Data Set*
Blur kernel from Levin *et al.*, “Understanding and Evaluating Blind Deconvolution Algorithms”


x
Corruption
y
Clean Image
Corrupt Image
$\hat{x}$
Restoration
y
Estimate
Observation


follows $p(\mathbf{x})$
follows $p(\mathbf{y} \mid \mathbf{x})$
$\mathbf{x}$
Corruption
$\mathbf{y}$
Clean Image
Corrupt Image
$\hat{\mathbf{x}}$
Restoration
$\mathbf{y}$
Estimate
Observation
requires $p(\mathbf{x} \mid \mathbf{y})$

Bayes' Rule

Posterior probability of restored image $\mathbf{x}$ given observation $\mathbf{y}$ is

$$p(\mathbf{x} \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \mathbf{x}) \cdot p(\mathbf{x})}{p(\mathbf{y})}$$

Finding the Restored Image

Maximum a-posteriori solution (MAP) is the image $\mathbf{x}$ with the highest posterior probability given observation $\mathbf{y}$:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x} \mid \mathbf{y}) = \arg\max_{\mathbf{x}} \left(p(\mathbf{y} \mid \mathbf{x}) \cdot p(\mathbf{x}) \right)$$

Assumption for Image Corruption

Each pixel y_i of observation $\mathbf{y}$ depends on clean image $\mathbf{x}$ as

$$y_i = \Big(\sum_j K_{ij} x_j\Big) + r$$

with *blur matrix* $\mathbf{K}$ and pixel-independent *Gaussian noise* $r \sim \mathcal{N}(0, \sigma^2)$.

Assumption for Image Corruption

Each pixel y_i of observation $\mathbf{y}$ depends on clean image $\mathbf{x}$ as

$$y_i = \left(\sum_j K_{ij} x_j\right) + r$$

with *blur matrix* $\mathbf{K}$ and pixel-independent *Gaussian noise* $r \sim \mathcal{N}(0, \sigma^2)$.

Resulting Gaussian Likelihood

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{x}) &= \prod_i \mathcal{N}\left(y_i; \sum_j K_{ij} x_j, \sigma^2\right) \\ &= \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}) \\ &\propto \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{K}\mathbf{x} - \mathbf{y}\|^2\right) \end{aligned}$$

Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Field of Experts Prior *(Roth and Black)*

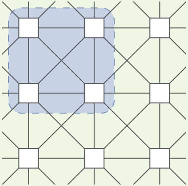
$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$

Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Field of Experts Prior *(Roth and Black)*

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$

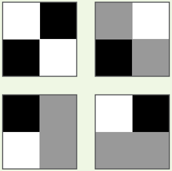


Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Field of Experts Prior *(Roth and Black)*

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$

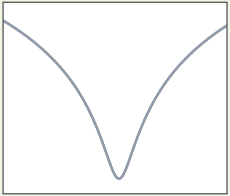


Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Field of Experts Prior *(Roth and Black)*

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$



Assumption for Restored Images

- local smoothness means low difference between neighbouring pixels
- can be extended to filter responses on image patches (*cliques*)

Field of Experts Prior *(Roth and Black)*

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$

$$\propto \exp\left(-\sum_{c} \sum_{i=1}^{N} \rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})\right)$$

Energy Formulation

Posterior probability $p(\mathbf{x} \mid \mathbf{y})$ can also be written as

$$\begin{aligned} p(\mathbf{x} \mid \mathbf{y}) &\propto p(\mathbf{y} \mid \mathbf{x}) \cdot p(\mathbf{x}) \\ &\propto \exp\left(-E(\mathbf{x} \mid \mathbf{y})\right) \end{aligned}$$

$$\text{with} \quad E(\mathbf{x} \mid \mathbf{y}) = \frac{1}{2\sigma^2} \cdot \|\mathbf{K}\mathbf{x} - \mathbf{y}\|^2 + \sum_c \sum_{i=1}^{N} \rho_i(\mathbf{f}_i^\top \mathbf{x}_{(c)})$$

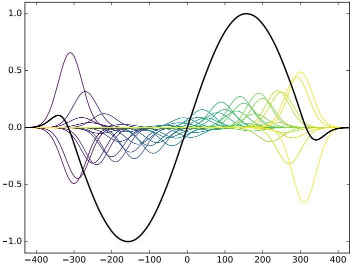
MAP Solution

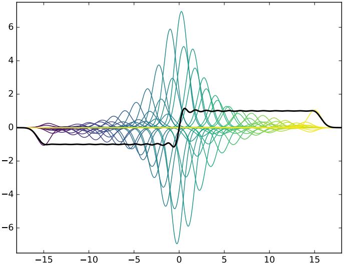
$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x} \mid \mathbf{y}) = \arg\min_{\mathbf{x}} E(\mathbf{x} \mid \mathbf{y})$$

- non-convex for *robust* penalty functions ρ_i
- use iterative minimization method

Model Parameterization

- linear filters $\mathbf{f}_i$ constructed from a basis of *zero-mean* filters
- functions ρ_i as flexible Gaussian *radial basis function mixtures*





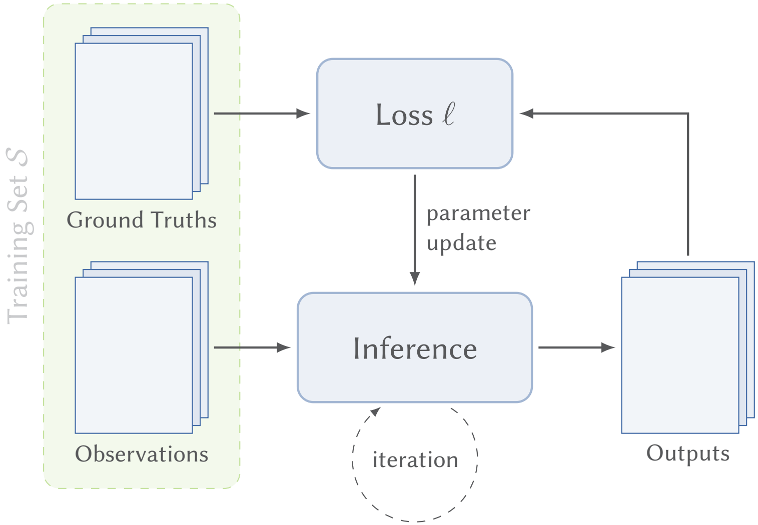
Discriminative Training of Parameters

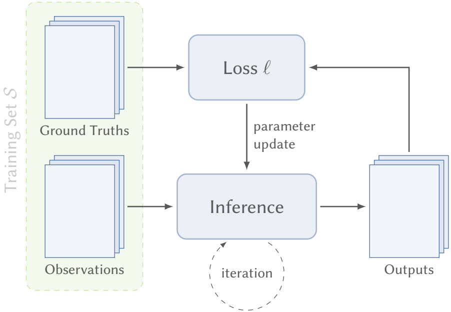
- find parameters Θ that minimize loss ℓ over training set $\mathcal{S}$
- supervised learning
- negative PSNR as loss function ℓ
- resulting models are application-specific

Bi-Level Optimization Task

$$\Theta^* = \arg\min_{\Theta} \sum_{k=1}^{|\mathcal{S}|} \ell(\hat{\mathbf{x}}^k, \mathbf{x}_{\mathrm{gt}}^k) \qquad \text{upper level}$$

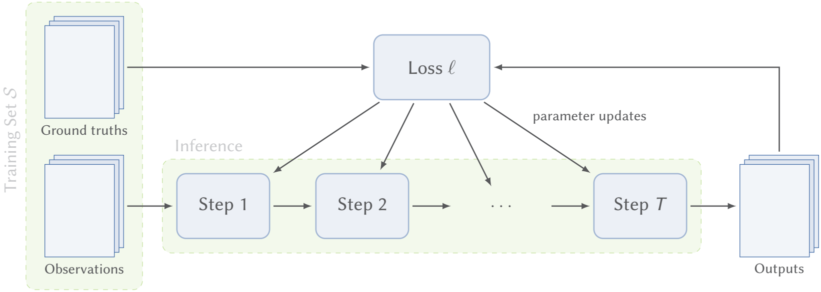
$$\hat{\mathbf{x}}^k = \arg\min_{\mathbf{x}} E(\mathbf{x} \mid \mathbf{y}^k; \Theta) \qquad \text{lower level (MAP)}$$


Training Set $\mathcal{S}$
Ground Truths
Observations
Loss ℓ
parameter update
Inference
iteration
Outputs



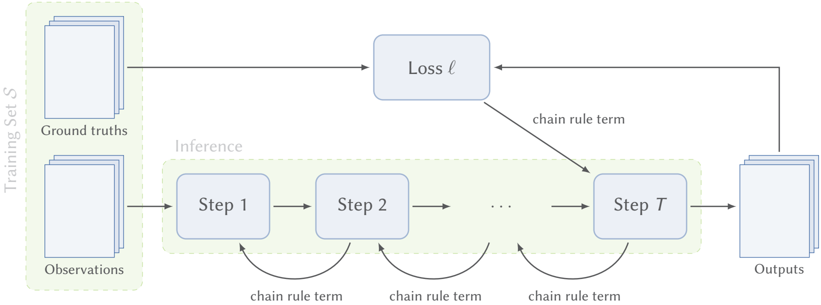
Problems

- inference can take unknown/variable number of iterations
- no easy closed-form derivative of ℓ wrt. model parameters



Alternative: Truncated Optimization

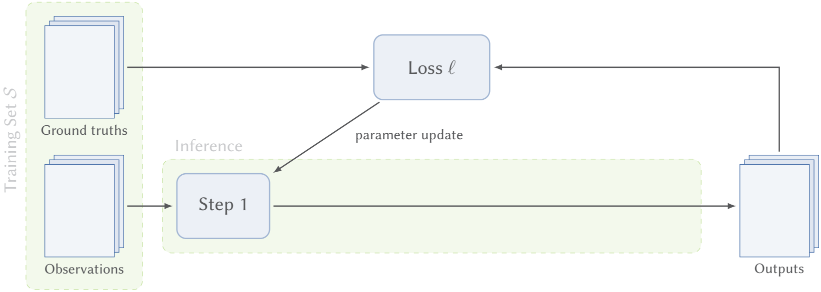
- *unroll* MAP inference for a small, fixed number of steps
- discriminative training makes up for “incomplete” inference
- better yet: each step gets *individual set of parameters* Θ_t
- different model at each step, not original problem anymore



End-to-End Training with Backpropagation

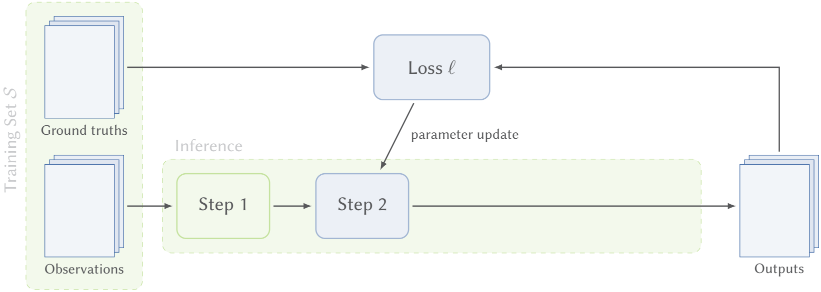
$$\frac{\partial \ell(\hat{\mathbf{x}}, \mathbf{x}_{\mathrm{gt}})}{\partial \Theta_t} = \frac{\partial \ell(\hat{\mathbf{x}}, \mathbf{x}_{\mathrm{gt}})}{\partial \hat{\mathbf{x}}_T} \cdot \frac{\partial \hat{\mathbf{x}}_T}{\partial \hat{\mathbf{x}}_{T-1}} \cdot \ldots \cdot \frac{\partial \hat{\mathbf{x}}_{t+1}}{\partial \hat{\mathbf{x}}_t} \cdot \frac{\partial \hat{\mathbf{x}}_t}{\partial \Theta_t}$$

- can be seen as *convolutional neural network* with special operations



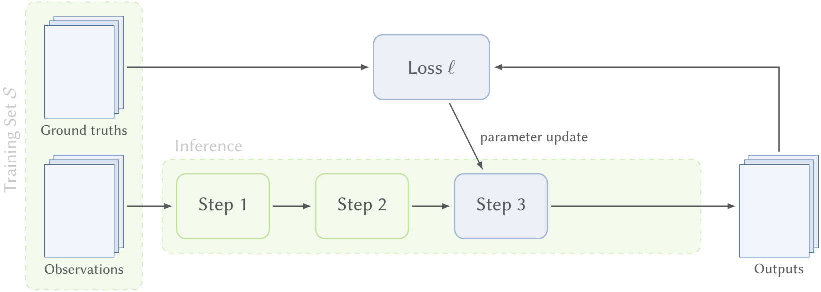
Better Results with Greedy Training

- train each step individually, keeping predecessors' parameters fixed
- output after each step t is a viable output $\hat{\mathbf{x}}_t$



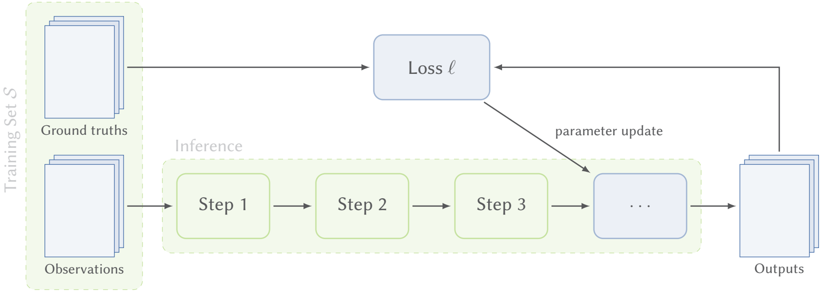
Better Results with Greedy Training

- train each step individually, keeping predecessors' parameters fixed
- output after each step t is a viable output $\hat{\mathbf{x}}_t$



Better Results with Greedy Training

- train each step individually, keeping predecessors' parameters fixed
- output after each step t is a viable output $\hat{\mathbf{x}}_t$



Better Results with Greedy Training

- train each step individually, keeping predecessors' parameters fixed
- output after each step t is a viable output $\hat{\mathbf{x}}_t$

Learned Inference Approaches

Approach 1: Truncated Gradient Descent

- done before, most recently by Chen and Pock
- very fast, state-of-the-art results for denoising
- not shown for deblurring before, maybe because of bad results

Approach 1: Truncated Gradient Descent

- done before, most recently by Chen and Pock
- very fast, state-of-the-art results for denoising
- not shown for deblurring before, maybe because of bad results

Approach 2: Truncated Half-quadratic Inference

- variants of this used before *e. g.* by Schmidt *et al.*
- very strong results for denoising and deblurring
- *Shrinkage Fields* also highly efficient and scalable
- HQ variant considered here is much less efficient

Gradient Descent

Minimize energy function by following its gradient, step by step:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\partial E(\mathbf{x}_{t-1} \mid \mathbf{y})}{\partial \mathbf{x}_{t-1}}$$

$$\widehat{=}\ \mathbf{x}_{t-1} - \left(\lambda_t \cdot \mathbf{K}^T(\mathbf{K}\mathbf{x}_{t-1} - \mathbf{y}) + \sum_{i=1}^{N} \mathbf{F}_{ti}^T \cdot \rho'_{ti}(\mathbf{F}_{ti}\mathbf{x}_{t-1})\right)$$

Gradient Descent

Minimize energy function by following its gradient, step by step:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\partial E(\mathbf{x}_{t-1} \mid \mathbf{y})}{\partial \mathbf{x}_{t-1}}$$

$$\widehat{=} \mathbf{x}_{t-1} - \left(\lambda_t \cdot \mathbf{K}^T(\mathbf{K}\mathbf{x}_{t-1} - \mathbf{y}) + \sum_{i=1}^{N} \mathbf{F}_{ti}^T \cdot \rho'_{ti}(\mathbf{F}_{ti}\mathbf{x}_{t-1})\right)$$

Gradient Descent

Minimize energy function by following its gradient, step by step:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\partial E(\mathbf{x}_{t-1} \mid \mathbf{y})}{\partial \mathbf{x}_{t-1}}$$

$$\widehat{=}\ \mathbf{x}_{t-1} - \left(\lambda_t \cdot \mathbf{K}^T(\mathbf{K}\mathbf{x}_{t-1} - \mathbf{y}) + \sum_{i=1}^{N} \mathbf{F}_{ti}^T \cdot \rho'_{ti}(\mathbf{F}_{ti}\mathbf{x}_{t-1}) \right)$$

Computation

- in all equations, $\mathbf{K}$ and $\mathbf{F}_{ti}$ only appear as $\mathbf{K} \cdot \mathbf{v}$ and $\mathbf{F}_{ti} \cdot \mathbf{v}$
- each represents a filter applied to all cliques in an image $\mathbf{v}$, can be done efficiently via convolution as in $\mathbf{f}_{ti} * \mathbf{v}$

Half-quadratic Inference: Idea

If penalty functions ρ_i are quadratic, the prior becomes *Gaussian*

$$\begin{aligned} p(\mathbf{x}) &= \frac{1}{Z} \prod_{c \in C} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^T \mathbf{x}_{(c)})\right) \\ &\propto \mathcal{N}\left(\mathbf{x}; \mathbf{0}, \mathbf{P}^{-1}\right), \end{aligned}$$

leading to a *Gaussian* posterior distribution

$$\begin{aligned} p(\mathbf{x} \mid \mathbf{y}) &\propto p(\mathbf{y} \mid \mathbf{x}) \cdot p(\mathbf{x}) \\ &\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}) \cdot \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{P}^{-1}) \\ &\propto \mathcal{N}\left(\mathbf{x};\ \boldsymbol{\Omega}^{-1}\boldsymbol{\eta},\ \boldsymbol{\Omega}^{-1}\right). \end{aligned}$$

Half-quadratic Inference: Idea

If penalty functions ρ_i are quadratic, the prior becomes *Gaussian*

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in C} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^T \mathbf{x}_{(c)})\right)$$
$$\propto \mathcal{N}\left(\mathbf{x}; \mathbf{0}, \mathbf{P}^{-1}\right),$$

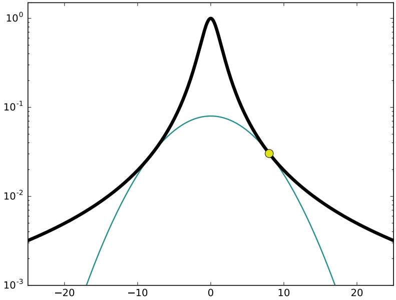
leading to a *Gaussian* posterior distribution

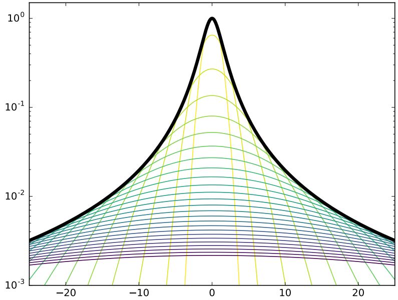
$$\begin{aligned} p(\mathbf{x} \mid \mathbf{y}) &\propto p(\mathbf{y} \mid \mathbf{x}) \cdot p(\mathbf{x}) \\ &\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}) \cdot \mathcal{N}\left(\mathbf{x}; \mathbf{0}, \mathbf{P}^{-1}\right) \\ &\propto \mathcal{N}\left(\mathbf{x};\ \boldsymbol{\Omega}^{-1}\boldsymbol{\eta},\ \boldsymbol{\Omega}^{-1}\right). \end{aligned}$$

MAP solution is the *mean* vector $\hat{\mathbf{x}} = \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}$ with $\begin{aligned} \boldsymbol{\Omega} &= \tfrac{1}{2\sigma^2}\mathbf{K}^\top\mathbf{K} + \mathbf{P} \\ \boldsymbol{\eta} &= \tfrac{1}{2\sigma^2}\mathbf{K}^\top\mathbf{y} \end{aligned}$

Half-Quadratic Augmentation *(multiplicative form)*

- approximate robust potentials $\exp\left(-\rho_i(u)\right)$ by *augmented* potentials $\exp\left(-\phi_i(u,z)\right)$ with auxiliary variable z
- $\phi_i(u,z)$ is a quadratic function if z is held fixed
- distinct values z_{ic} for each filter response $\mathbf{f}_i^T\mathbf{x}_{(c)} \Longrightarrow$ vector $\mathbf{z}$

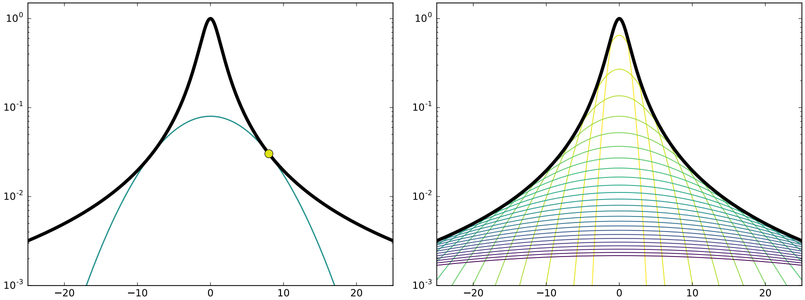




Half-Quadratic Augmentation *(multiplicative form)*

Prior probability is set to be *envelope* of all augmented potentials

$$p(\mathbf{x}) \propto \max_{\mathbf{z}} \left(\prod_{i=1}^{N} \prod_{c \in C} \exp\left(-\phi_i(\mathbf{f}_i^T \mathbf{x}_{(c)}, z_{ic})\right) \right)$$



MAP Estimation

- alternate between finding better $\hat{\mathbf{z}}$ for the approximation and solving resulting quadratic model for $\hat{\mathbf{x}}$

MAP Estimation

- alternate between finding better $\hat{\mathbf{z}}$ for the approximation and solving resulting quadratic model for $\hat{\mathbf{x}}$
- since best $\hat{\mathbf{z}}$ depends on current $\hat{\mathbf{x}}$, can be combined into one step

$$\hat{\mathbf{x}}_t = \boldsymbol{\Omega}_t^{-1} \cdot \boldsymbol{\eta}_t \quad \text{with}$$

$$\boldsymbol{\Omega}_t = \frac{\mathbf{K}^\top \mathbf{K}}{\sigma_t^2} + \sum_{i=1}^{N} \left(\mathbf{F}_{ti}^\top \cdot \mathrm{diag}\underbrace{\left\{ \rho_{ti}^*(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}) \right\}}_{\hat{\mathbf{z}}_{ti}} \cdot \mathbf{F}_{ti} \right) \qquad \boldsymbol{\eta}_t = \frac{\mathbf{K}^\top \mathbf{y}}{\sigma_t^2}$$

MAP Estimation

- alternate between finding better $\hat{\mathbf{z}}$ for the approximation and solving resulting quadratic model for $\hat{\mathbf{x}}$
- since best $\hat{\mathbf{z}}$ depends on current $\hat{\mathbf{x}}$, can be combined into one step

$$\hat{\mathbf{x}}_t = \boldsymbol{\Omega}_t^{-1} \cdot \boldsymbol{\eta}_t \quad \text{with}$$

$$\boldsymbol{\Omega}_t = \frac{\mathbf{K}^\top \mathbf{K}}{\sigma_t^2} + \sum_{i=1}^{N} \left(\mathbf{F}_{ti}^\top \cdot \operatorname{diag}\underbrace{\left\{ \rho_{ti}^*(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}) \right\}}_{\hat{\mathbf{z}}_{ti}} \cdot \mathbf{F}_{ti} \right) \qquad \boldsymbol{\eta}_t = \frac{\mathbf{K}^\top \mathbf{y}}{\sigma_t^2}$$

Computation

- need to solve linear equation system for each step, very costly
- construction of $\boldsymbol{\Omega}_t$ explicitly requires $\mathbf{K}^\top\mathbf{K}$ and $\mathbf{F}_{ti}^\top \cdot \operatorname{diag}\{\ldots\} \cdot \mathbf{F}_{ti}$
- functions ρ_{ti}^* are constrained non-negative so $\boldsymbol{\Omega}_t$ is *positive-definite*

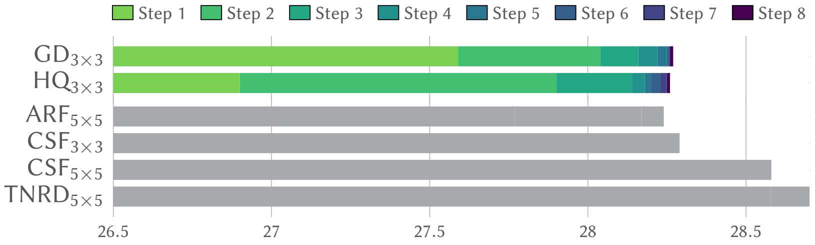
Experiments

Image Denoising

- both approaches trained greedily under identical conditions up to 8 inference steps, using 3×3 cliques
- 50 training images of size 128×128, noise level $\sigma^2 = 25.0$
- 100 iterations of *L-BFGS* per step to learn parameters

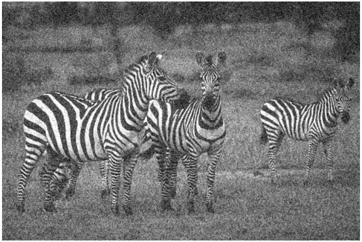
Image Denoising

- both approaches trained greedily under identical conditions up to 8 inference steps, using 3×3 cliques
- 50 training images of size 128×128, noise level $\sigma^2 = 25.0$
- 100 iterations of *L-BFGS* per step to learn parameters



68 image test set from Roth and Black (*Field of Experts*)

Ground Truth



Observation (20.20 dB)

$GD^{8}_{3\times3}$ (27.61 dB)

$HQ^{8}_{3\times3}$ (26.89 dB)

Ground truth taken from *Berkeley Segmentation Data Set*


Gradient descent (27.61 dB)


Half-quadratic (26.89 dB)

Ground Truth

Observation (20.16 dB)

$\mathrm{GD}^{8}_{3\times 3}$ (35.26 dB)

$\mathrm{HQ}^{8}_{3\times 3}$ (36.19 dB)

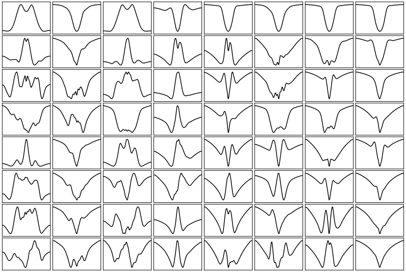
Ground truth taken from *Berkeley Segmentation Data Set*


Gradient descent (35.26 dB)

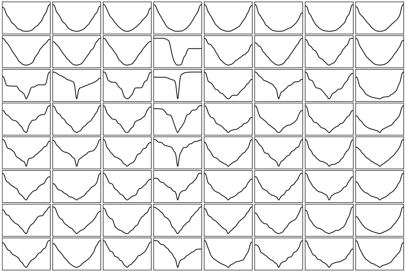

Half-quadratic (36.19 dB)

Learned Penalty Functions

- very flexible for GD model, many “inverted” penalty functions
- quite uniform for HQ model, resemble quadratic and hyper-Laplacian
- reason is the *positivity constraint* for HQ nonlinear functions



$GD^{8}_{3\times3}$ penalty functions



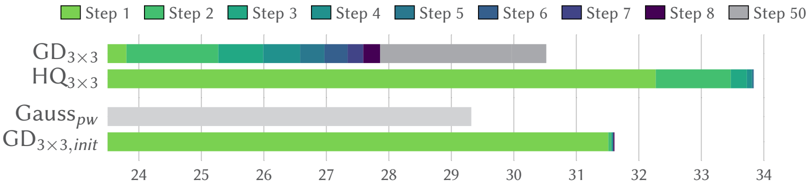
$HQ^{8}_{3\times3}$ penalty functions

Image Deblurring *(Non-blind Deconvolution)*

- both approaches trained greedily just like in denoising case
- 50 training images of size 128×128, kernels from Schmidt *et al.*
- 100 iterations of L-BFGS per step to learn parameters

Image Deblurring *(Non-blind Deconvolution)*

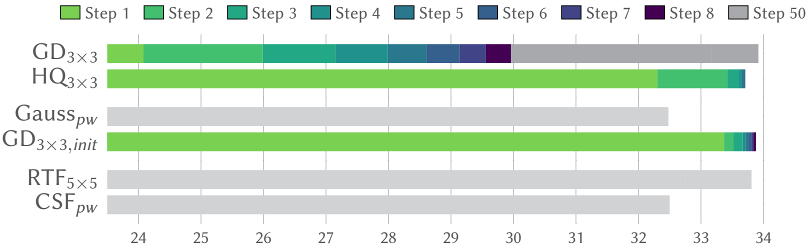
- both approaches trained greedily just like in denoising case
- 50 training images of size 128 × 128, kernels from Schmidt *et al.*
- 100 iterations of L-BFGS per step to learn parameters



50 image test set similar to training data

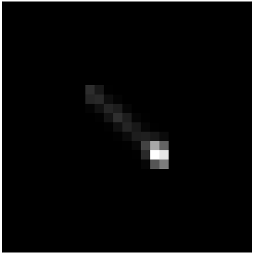
Image Deblurring *(Non-blind Deconvolution)*

- both approaches trained greedily just like in denoising case
- 50 training images of size 128 × 128, kernels from Schmidt *et al.*
- 100 iterations of L-BFGS per step to learn parameters

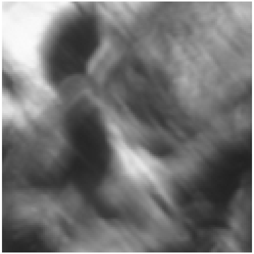


32 image test set from Levin *et al.*

Ground Truth



Blur Kernel



Observation (22.48 dB)

$\mathrm{GD}^{8}_{3\times3}$ (29.00 dB)

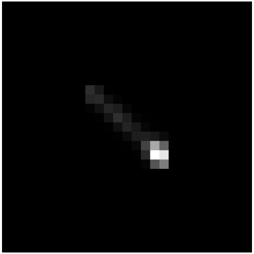
$\mathrm{HQ}^{8}_{3\times3}$ (33.33 dB)

Image from *Berkeley Segmentation Data Set*, blur kernel from Schmidt *et al.*

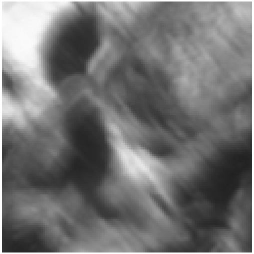

Gradient descent (29.00 dB)


Half-quadratic (33.33 dB)

Ground Truth

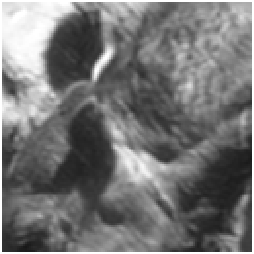


Blur Kernel



Observation (22.48 dB)

$GD^{50}_{3\times3}$ (30.88 dB)



$Gauss_{pw}$ (30.63 dB)

$GD_{3\times3,init}$ (32.07 dB)

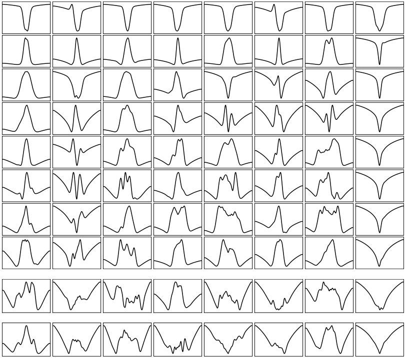
Image from *Berkeley Segmentation Data Set*, blur kernel from Schmidt *et al.*


Gradient descent 50 steps (30.88 dB)

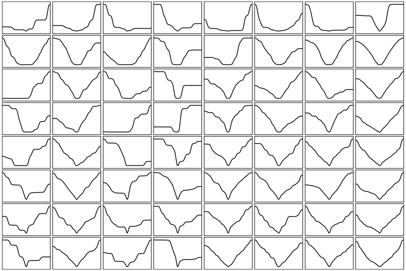

Gradient descent with Gaussian init (32.07 dB)

Learned Penalty Functions

- situation similar to denoising models
- later functions in HQ model tend towards ℓ_1-norm



$\mathrm{GD}^{50}_{3\times 3}$ penalty functions



$\mathrm{HQ}^{8}_{3\times 3}$ penalty functions

Conclusion

Findings

- both approaches achieved roughly same results for denoising
- truncated GD is less suitable for deblurring (depends of test set?)
- HQ variant is strong, but extremely slow
- constrained penalty functions restrict the model

Findings

- both approaches achieved roughly same results for denoising
- truncated GD is less suitable for deblurring (depends of test set?)
- HQ variant is strong, but extremely slow
- constrained penalty functions restrict the model

Possible Extensions and Further Research

- improve speed by parallelizing over images
- try solving $\mathbf{\Omega}_t \hat{\mathbf{x}}_t = \boldsymbol{\eta}_t$ iteratively instead of directly
- examine other applications and inference approaches

Findings

- both approaches achieved roughly same results for denoising
- truncated GD is less suitable for deblurring (depends of test set?)
- HQ variant is strong, but extremely slow
- constrained penalty functions restrict the model

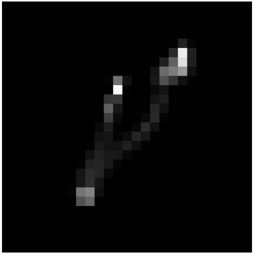
Possible Extensions and Further Research

- improve speed by parallelizing over images
- try solving $\boldsymbol{\Omega}_t \hat{\mathbf{x}}_t = \boldsymbol{\eta}_t$ iteratively instead of directly
- examine other applications and inference approaches

Thank you!

Bonus

Ground Truth



Blur Kernel

Observation (21.22 dB)

$\text{GD}^{8}_{3\times 3}$ (28.39 dB)

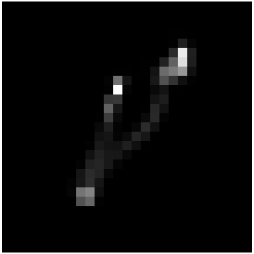
$\text{HQ}^{8}_{3\times 3}$ (32.90 dB)

Upper row from Levin *et al.*, “Understanding and Evaluating Blind Deconvolution Algorithms”


Gradient descent (28.39 dB)


Half-quadratic (32.90 dB)

Ground Truth



Blur Kernel

Observation (21.22 dB)

$\mathrm{GD}^{50}_{3\times 3}$ (33.64 dB)

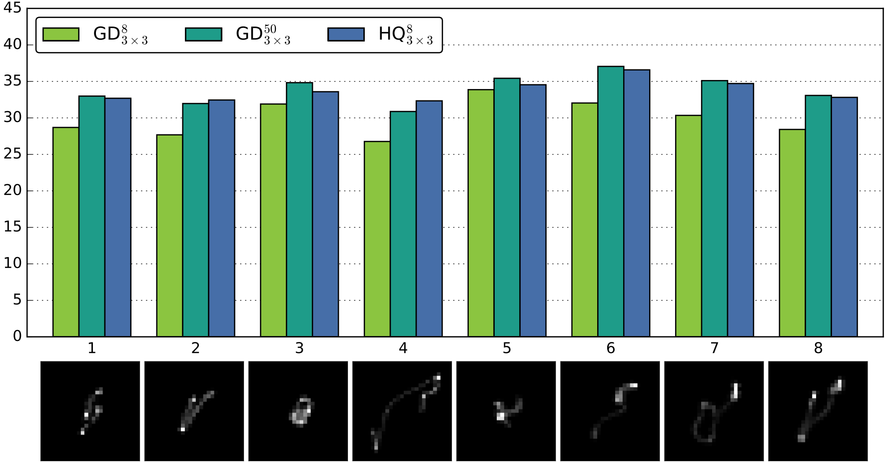
$\mathrm{Gauss}_{\mathrm{pw}}$ (32.34 dB)

$\mathrm{GD}^{8}_{3\times 3,\mathrm{init}}$ (33.88 dB)

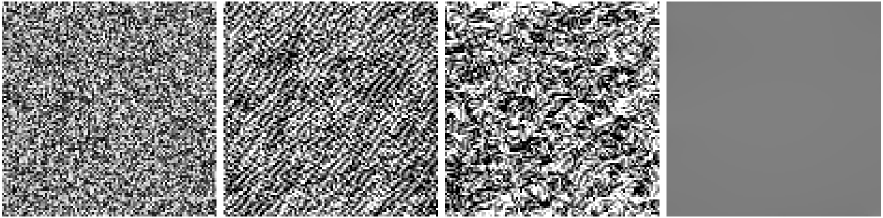
Upper row from Levin *et al.*, "Understanding and Evaluating Blind Deconvolution Algorithms"


Gradient descent 50 steps (33.64 dB)


Gradient descent with Gaussian init (33.88 dB)



Blur kernels from Levin *et al.*, "Understanding and Evaluating Blind Deconvolution Algorithms"



Uniform noise | $\mathrm{GD}^{8}_{3\times3}$, step 2 | $\mathrm{GD}^{8}_{3\times3}$, step 5 | $\mathrm{HQ}^{8}_{3\times3}$, any step

Pattern Sythesis Experiment *(after Chen and Pock)*

- Input is uniform random noise
- Repeatedly apply same step of a trained model until convergence
- Gradient descent steps encourage distinct patterns
- Half-quadratic steps quickly lead to uniform gray image