Video denoising via Bayesian modeling of patches

Pablo Arias

joint work with Thibaud Ehret and Jean-Michel Morel

CMLA, ENS Paris-Saclay

IHP workshop: Statistical modeling for shapes and imaging

Paris - 14/03/2019

Why are we still researching in denoising

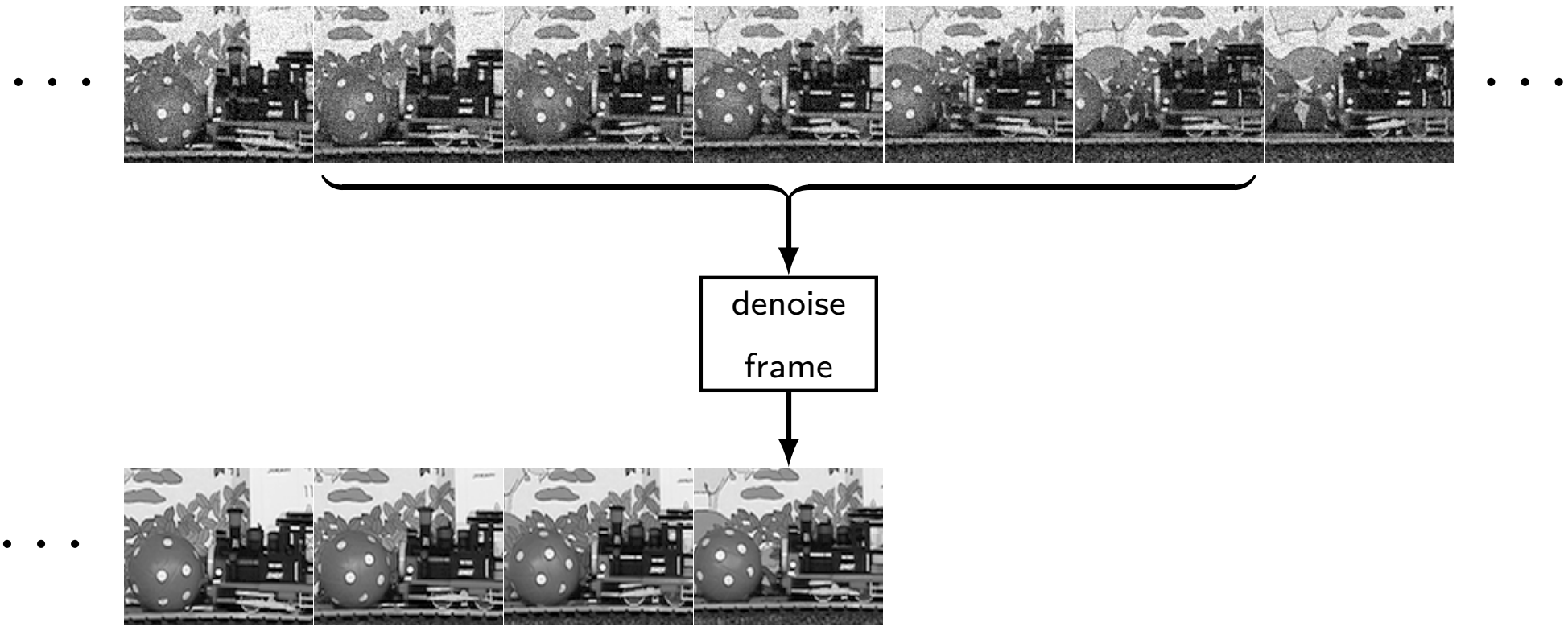
Denoising is mainly for people (i.e. not as much as a pre-processing for a CV task)

- Photography
- Smartphone cameras
- Film industry
- Night vision cameras
- Surveillance cameras
- Medical imaging
- Astronomy

A lot of research has been devoted to still image denoising,

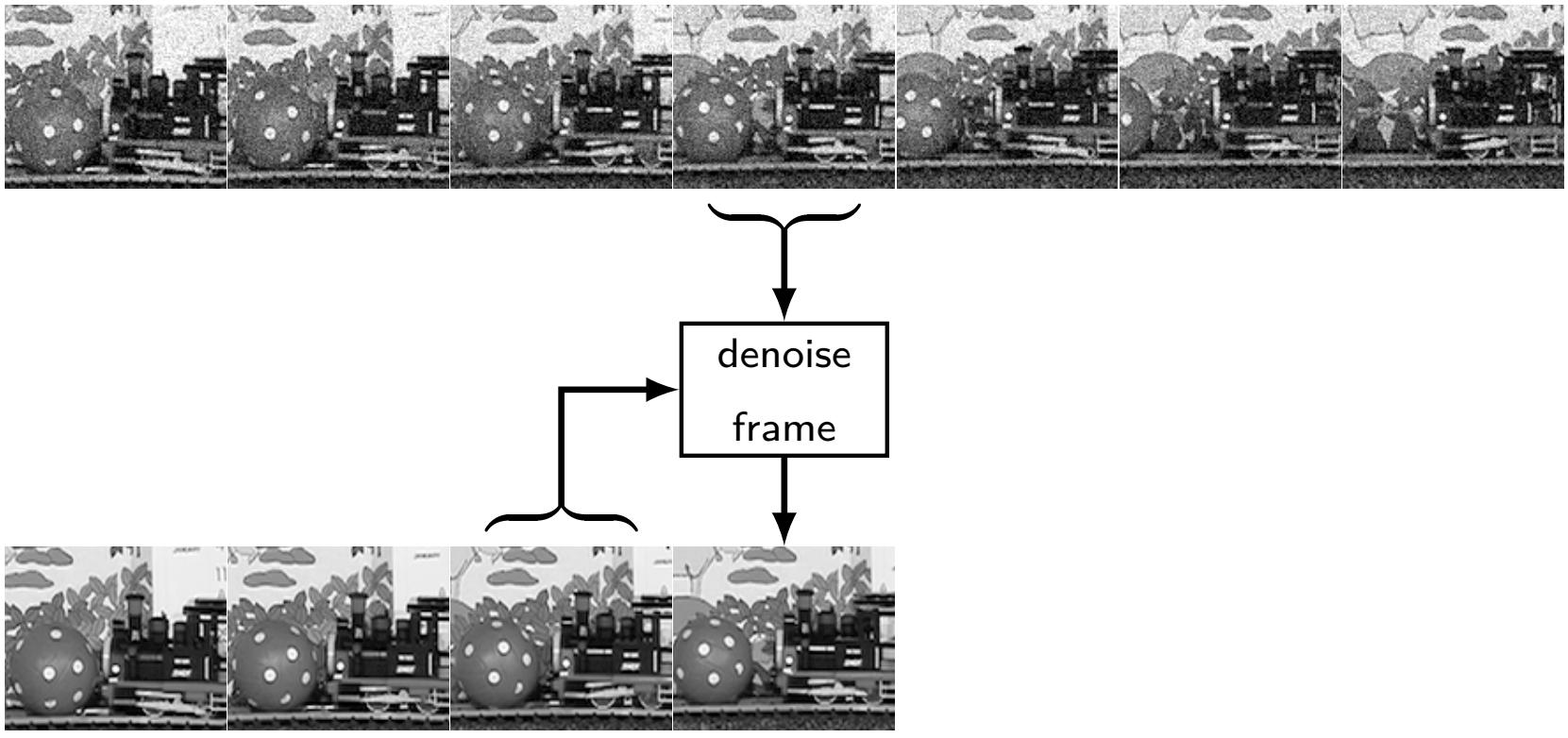
but there is still room for improvement in video denoising.

Recursive vs. non-recursive methods



In this talk: video denoising via Bayesian estimation of patches, non-recursive and recursive approaches.

Recursive vs. non-recursive methods



In this talk: video denoising via Bayesian estimation of patches, non-recursive and recursive approaches.

State of the art

Non-recursive

- Non-local means [Buades et al.'05], [Liu, Freeman '10]
- K-SVD [Protter, Elad '07]
- V-BM3D [Dabov et al.'07]
- Graph regularization [Ghoniem et al. '10]
- BM4D, V-BM4D [Maggioni et al.'12]
- SPTWO [Buades et al.'17]
- VNLB [Arias, Morel'18]
- VIDOSAT [Wen et al.'19]
- SALT [Wen et al.'19]
- VNLnet [Davy et al.'19]

Recursive

- Bilateral + Kalman filter [Zuo et al.'13]
- Bilateral + Kalman filter [Pfleger et al.'17]
- Recursive NL-means [Ali, Hardie'17]
- Gaussian-Laplacian fusion [Ehmann et al.'18]

State of the art

Non-recursive

- Non-local means [Buades et al.'05], [Liu, Freeman '10]
- K-SVD [Protter, Elad '07]
- V-BM3D [Dabov et al.'07]
- Graph regularization [Ghoniem et al. '10]
- BM
- SPT
- VNI
- VID
- SALT [Wen et al.'19]
- VNLnet [Davy et al.'19]

Recursive

- Bilateral + Kalman filter [Zuo et al.'13]
- Bilateral + Kalman filter [Pfleger et al.'17]
- Recursive NL-means [Ali, Hardie'17]
- Gaussian Laplacian fusion [Ehmann et

Non-recursive/mixed approaches good results at high computational cost.

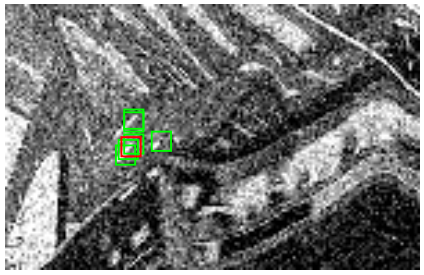
Recursive methods: real-time performance but worse results.

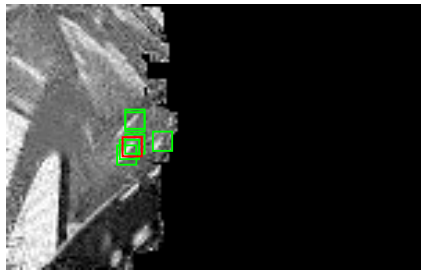
Contents

Contents

Attention! Strange diagrams ahead

Video denoising framework





Extract n similar patches

$[\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$

Wiener filtering

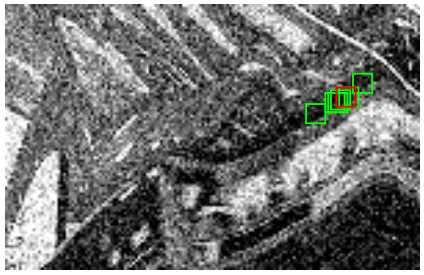
$[\widehat{\boldsymbol{p}}_1, \ldots, \widehat{\boldsymbol{p}}_n]$

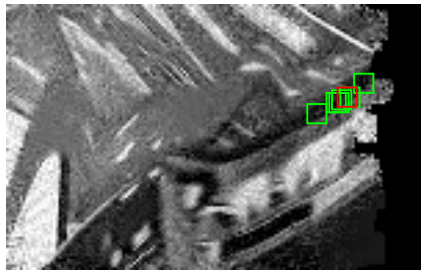
Aggregate by averaging

$[\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$

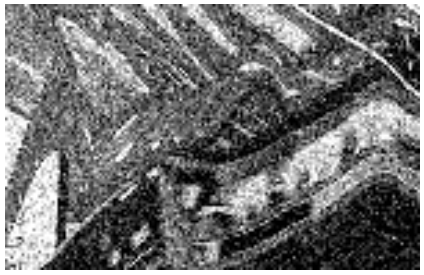
Estimate model parameters

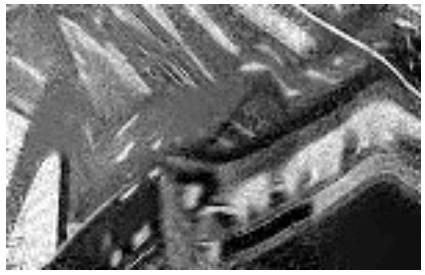
Video denoising framework

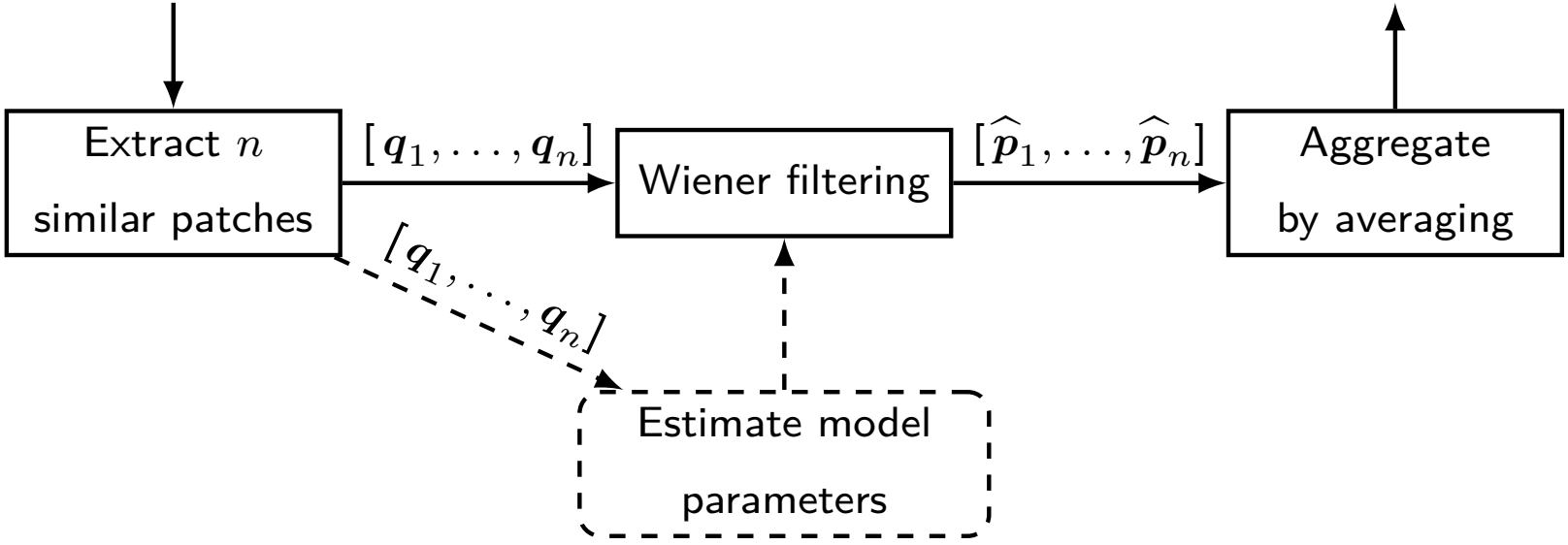




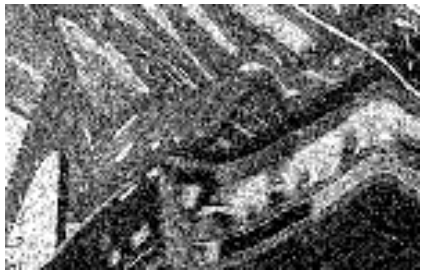
Video denoising framework

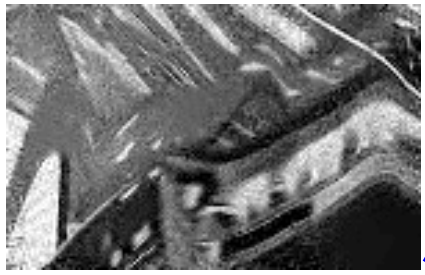






Video denoising framework





Extract n similar patches

$[\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$

Wiener filtering

$[\widehat{\boldsymbol{p}}_1, \ldots, \widehat{\boldsymbol{p}}_n]$

Aggregate by averaging

$[\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n]$

Estimate model parameters

set as guide

VNLB: Bayesian patch estimation with Gaussian prior

$\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$: n "similar" **clean** patches from u

$\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$: n "similar" **noisy** patches from v

Assumption: patches are IID samples of a Gaussian distribution

$$\mathbb{P}(\boldsymbol{p}_i) = \mathcal{N}(\boldsymbol{p}_i \,|\, \boldsymbol{\mu}, C)$$

$$\mathbb{P}(\boldsymbol{q}_i | \boldsymbol{p}_i) = \mathcal{N}(\boldsymbol{q}_i \,|\, \boldsymbol{p}_i, \sigma^2 I)$$

For each noisy patch $\boldsymbol{q}_i$ estimate $\boldsymbol{p}_i$ as the MAP/MMSE, given by the Wiener filter:

$$\hat{\boldsymbol{p}}_i = \boldsymbol{\mu} + C(C + \sigma^2 I)^{-1}(\boldsymbol{q}_i - \boldsymbol{\mu})$$

VNLB: Bayesian patch estimation with Gaussian prior

$\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$: n "similar" **clean** patches from u

$\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$: n "similar" **noisy** patches from v

Assumption: patches are IID samples of a Gaussian distribution

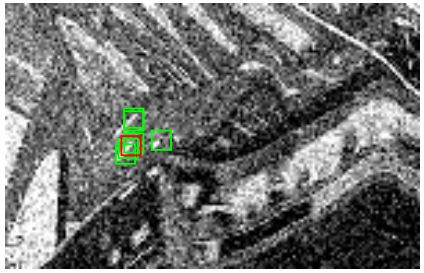
$$\mathbb{P}(\boldsymbol{p}_i) = \mathcal{N}(\boldsymbol{p}_i \,|\, \boldsymbol{\mu}, C)$$

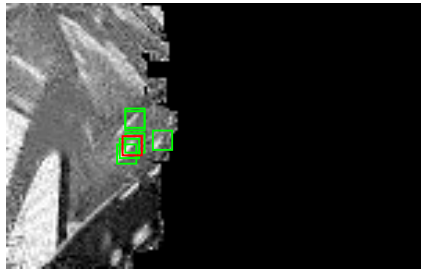
$$\mathbb{P}(\boldsymbol{q}_i | \boldsymbol{p}_i) = \mathcal{N}(\boldsymbol{q}_i \,|\, \boldsymbol{p}_i, \sigma^2 I)$$

For each noisy patch $\boldsymbol{q}_i$ estimate $\boldsymbol{p}_i$ as the MAP/MMSE, given by the Wiener filter:

$$\hat{\boldsymbol{p}}_i = \boldsymbol{\mu} + C(C + \sigma^2 I)^{-1}(\boldsymbol{q}_i - \boldsymbol{\mu})$$

Video denoising framework





Extract n similar patches

$[\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$

Wiener filtering

$[\widehat{\boldsymbol{p}}_1, \ldots, \widehat{\boldsymbol{p}}_n]$

Aggregate by averaging

$[\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n]$

Estimate model parameters

Estimating the prior parameters

How to estimate $\boldsymbol{\mu}$ and C?

STEP 2: Use the **guide** patches $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n \sim N(\boldsymbol{\mu}, C)$:

$$\widehat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{g}_i \qquad \widehat{C} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{g}_i - \widehat{\boldsymbol{\mu}})(\boldsymbol{g}_i - \widehat{\boldsymbol{\mu}})^T$$

STEP 1: Use the **noisy** pathes $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n \sim N(\boldsymbol{\mu}, C + \sigma^2 I)$:

$$\widehat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{q}_i \qquad \widehat{C} \approx \widehat{C}_{\boldsymbol{q}} - \sigma^2 I = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{q}_i - \widehat{\boldsymbol{\mu}})(\boldsymbol{q}_i - \widehat{\boldsymbol{\mu}})^T - \sigma^2 I$$

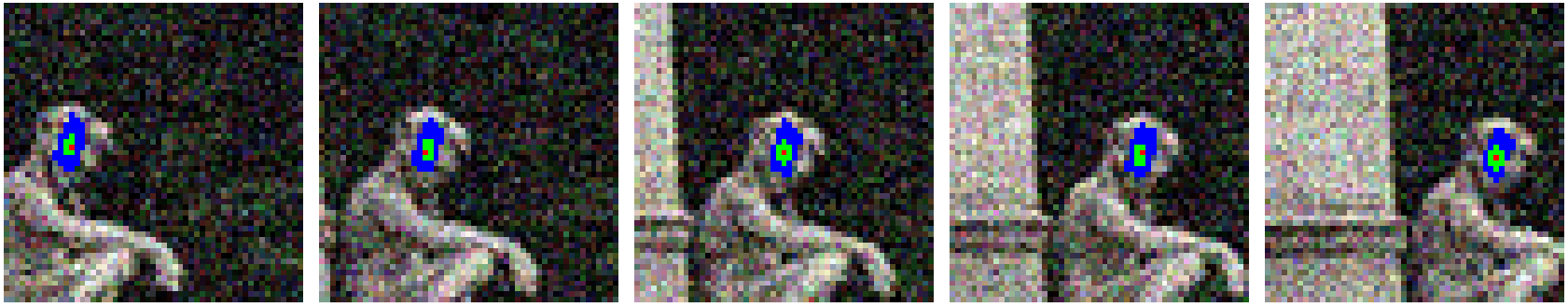
Estimating the prior parameters

We use the following hard-thresholding of the eigenvalues of the sample covariance matrix of the noisy patches:

$$\hat{\lambda}_i^{\mathsf{H}} = \mathrm{H}_\tau(\hat{\xi}_i - \sigma^2) = \begin{cases} \hat{\xi}_i - \sigma^2 & \text{if } \hat{\xi}_i \geqslant \tau\sigma^2 \\ 0 & \text{if } \hat{\xi}_i < \tau\sigma^2. \end{cases}$$

Based on the results of Debashis Paul (2007) on the aymptotic properties for the eigenvalues and eigenvectors of the noisy sample covariance matrix.

Examples of local Gaussian models



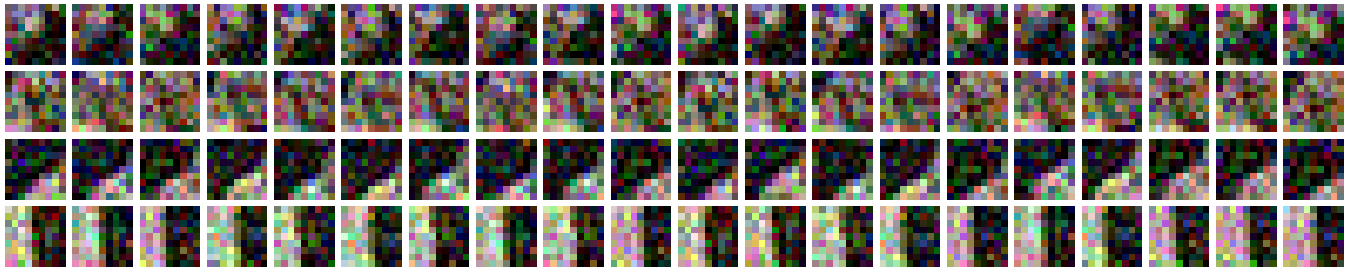
Groups of 200 similar $9 \times 9 \times 4$ patches, in a $37 \times 37 \times 5$ search region.

Nearest neighbors 1 to 5.

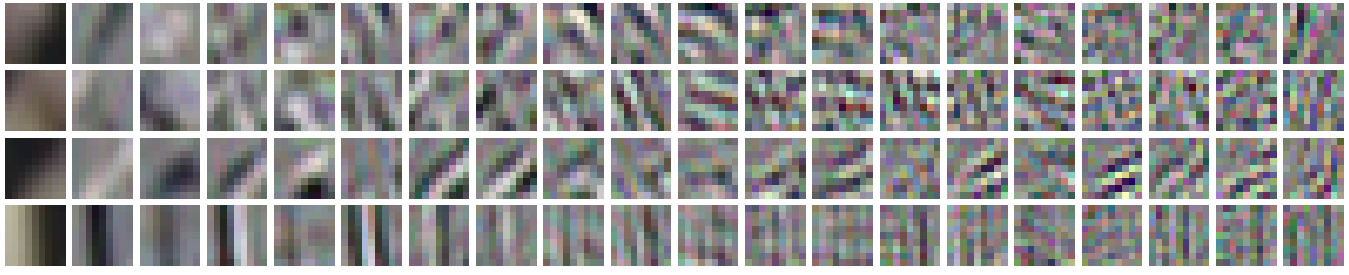
Nearest neighbors 6 to 45.

Nearest neighbors 46 to 200.

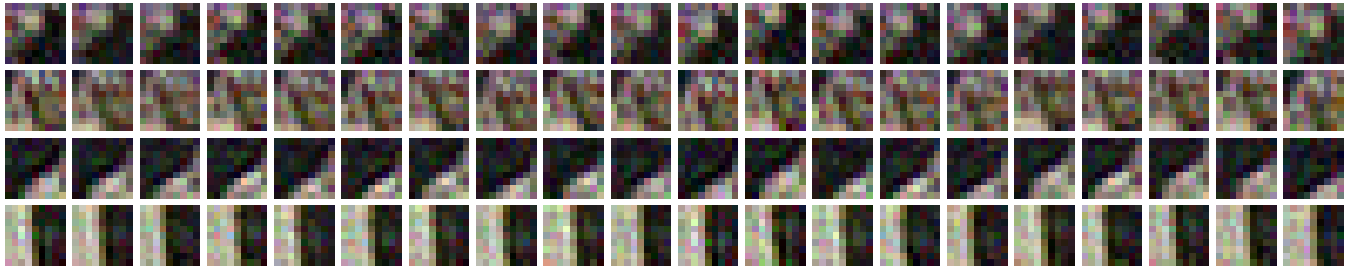
VLNB: Examples of local Gaussian models



20 nearest neighbors (noisy).



sample mean and first 19 principal directions.



corresponding MAP estimates.

Patch search and aggregation

- 3D rectangular patches, typical sizes are $10 \times 10 \times 2$.
- Motion compensated search region: a square tracing the motion trajectory of the target patch. Motion estimation using optical flow (TV-L1 [Zach et al. '07]).
- Search region size: $21 \times 21 \times 9$
- Two iterations over the whole video (as in BM3D [Dabov et al. '07]).

Contents

Motivation

$$u_t \;=\; \texttt{recursive-method}(f_t, u_{t-1})$$

Design constraints:

- reduce computational cost
- reduce memory cost ($M_t = \mathcal{O}(1)$)

Appealing properties:

- natural mechanism to enforce temporal consistency
- can aggregate an arbitrary number of past frames

Motivation

$$u_t, M_t \;=\; \texttt{recursive-method}(f_t, M_{t-1})$$

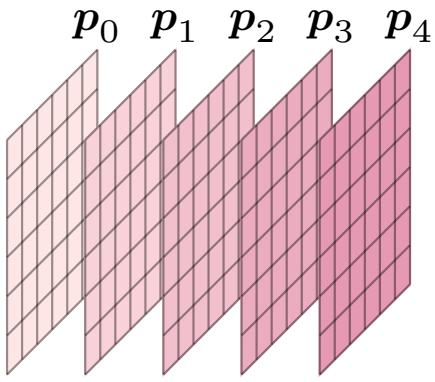
Design constraints:

- reduce computational cost
- reduce memory cost ($M_t = \mathcal{O}(1)$)

Appealing properties:

- natural mechanism to enforce temporal consistency
- can aggregate an arbitrary number of past frames

Dynamic Gaussian patch model

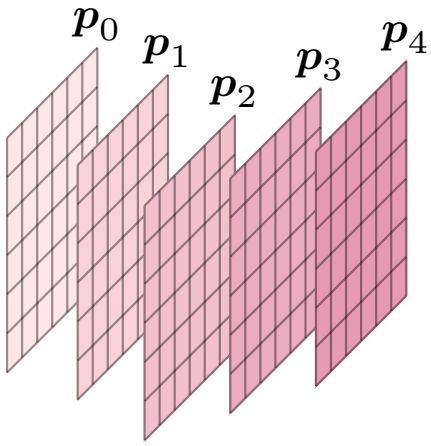
$\boldsymbol{p} =$ 

"Static" model: $\begin{cases} \boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}, C), \\ \boldsymbol{q} \sim \mathcal{N}(\boldsymbol{p}, \sigma^2 I). \end{cases}$

We assume a Gaussian linear dynamic model for a patch trajectory $\boldsymbol{p}_t$:

$$\begin{cases} \boldsymbol{p}_0 = \boldsymbol{\mu}_0 + \boldsymbol{w}_0, & \boldsymbol{w}_0 \sim \mathcal{N}(\mathbf{0}, P_0), \\ \boldsymbol{p}_t = \boldsymbol{p}_{t-1} + \boldsymbol{w}_t, & \boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, W_t), \\ \boldsymbol{q}_t = \boldsymbol{p}_t + \boldsymbol{r}_t, & \boldsymbol{r}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 I). \end{cases}$$

Dynamic Gaussian patch model

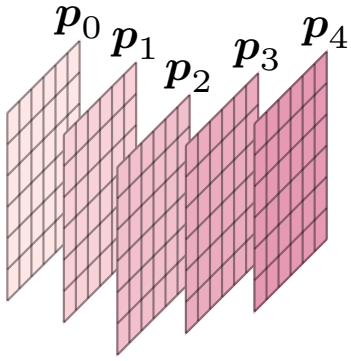
$\boldsymbol{p} =$ 

"Static" model: $\begin{cases} \boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}, C), \\ \boldsymbol{q} \sim \mathcal{N}(\boldsymbol{p}, \sigma^2 I). \end{cases}$

We assume a Gaussian linear dynamic model for a patch trajectory $\boldsymbol{p}_t$:

$$\begin{cases} \boldsymbol{p}_0 = \boldsymbol{\mu}_0 + \boldsymbol{w}_0, & \boldsymbol{w}_0 \sim \mathcal{N}(\mathbf{0}, P_0), \\ \boldsymbol{p}_t = \boldsymbol{p}_{t-1} + \boldsymbol{w}_t, & \boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, W_t), \\ \boldsymbol{q}_t = \boldsymbol{p}_t + \boldsymbol{r}_t, & \boldsymbol{r}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2 I). \end{cases}$$

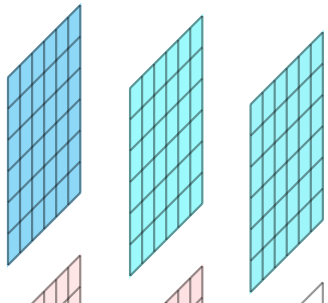
Dynamic Gaussian patch model

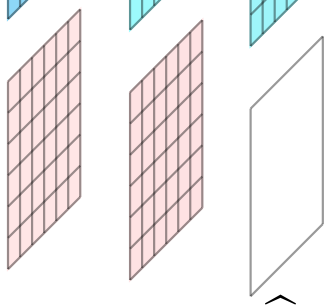
$\boldsymbol{p} =$ 

Full patch model: $\begin{cases} \boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}_K, Q_K^{-1}), \\ \boldsymbol{q} \sim \mathcal{N}(\boldsymbol{p}, \sigma^2 I). \end{cases}$

$$\boldsymbol{\mu}_K = \begin{pmatrix} \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_0 \\ \boldsymbol{\mu}_0 \end{pmatrix}, \quad Q_K = \begin{pmatrix} P_0^{-1}{+}W_1^{-1} & -W_1^{-1} & & & \\ -W_1^{-1} & W_1^{-1}{+}W_2^{-1} & -W_2^{-1} & & \\ & -W_2^{-1} & W_2^{-1}{+}W_3^{-1} & -W_3^{-1} & \\ & & -W_3^{-1} & W_3^{-1}{+}W_4^{-1} & -W_4^{-1} \\ & & & -W_4^{-1} & W_5^{-1} \end{pmatrix}$$

Recursive Bayesian patch estimation: filtering

$\boldsymbol{q} =$ 

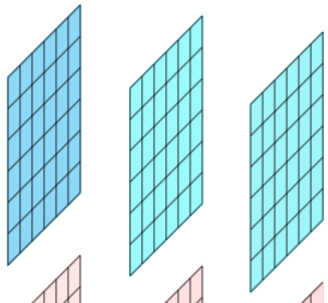
$\widehat{\boldsymbol{p}} =$ 

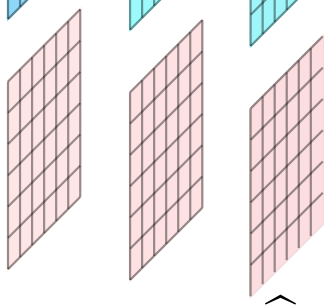
$$\widehat{\boldsymbol{p}}_2 = \mathbb{E}\{\boldsymbol{p}_2 | \boldsymbol{q}_2, \boldsymbol{q}_1, \boldsymbol{q}_0)\}$$

Kalman filter: recursive computation of $\mathbb{P}(\boldsymbol{p}_t | \boldsymbol{q}_t, \ldots, \boldsymbol{q}_1) \sim \mathcal{N}\left(\widehat{\boldsymbol{p}}_t, P_t\right)$

$$\begin{cases} \widehat{\boldsymbol{p}}_t = (I - K_t)\widehat{\boldsymbol{p}}_{t-1} + K_t \boldsymbol{q}_t & \textit{state mean} \\ P_t = (I - K_t)(P_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (P_{t-1} + W_t)(P_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{Kalman gain} \end{cases}$$

Recursive Bayesian patch estimation: filtering

$\boldsymbol{q} =$ 

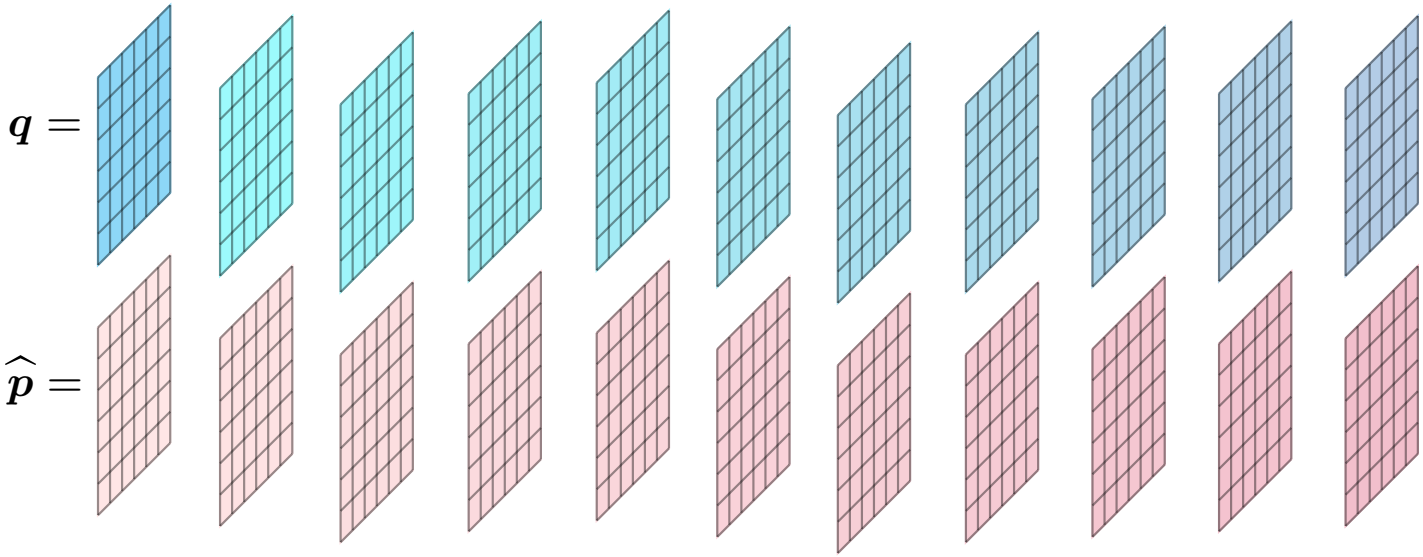
$\widehat{\boldsymbol{p}} =$ 

$$\widehat{\boldsymbol{p}}_2 = \mathcal{F}(\boldsymbol{q}_2, \widehat{\boldsymbol{p}}_1, P_1, W_2)\}$$

Kalman filter: recursive computation of $\mathbb{P}(\boldsymbol{p}_t|\boldsymbol{q}_t, \ldots, \boldsymbol{q}_1) \sim \mathcal{N}\left(\widehat{\boldsymbol{p}}_t, P_t\right)$

$$\begin{cases} \widehat{\boldsymbol{p}}_t = (I - K_t)\widehat{\boldsymbol{p}}_{t-1} + K_t \boldsymbol{q}_t & \textit{state mean} \\ P_t = (I - K_t)(P_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (P_{t-1} + W_t)(P_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{Kalman gain} \end{cases}$$

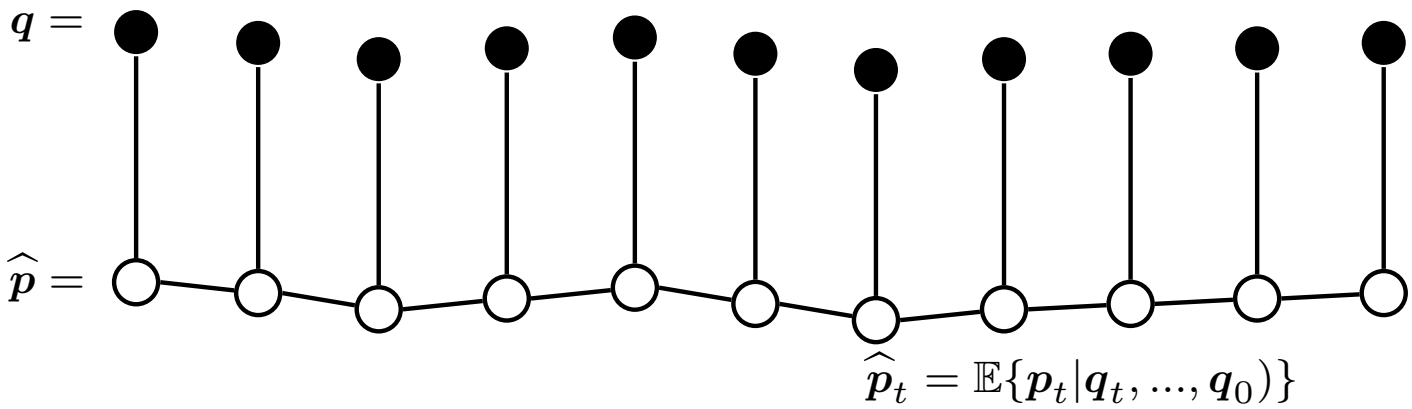
Recursive Bayesian patch estimation: filtering



Kalman filter: recursive computation of $\mathbb{P}(\boldsymbol{p}_t|\boldsymbol{q}_t,\ldots,\boldsymbol{q}_1) \sim \mathcal{N}\left(\widehat{\boldsymbol{p}}_t, P_t\right)$

$$\begin{cases} \widehat{\boldsymbol{p}}_t = (I - K_t)\widehat{\boldsymbol{p}}_{t-1} + K_t\boldsymbol{q}_t & \textit{state mean} \\ P_t = (I - K_t)(P_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (P_{t-1} + W_t)(P_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{Kalman gain} \end{cases}$$

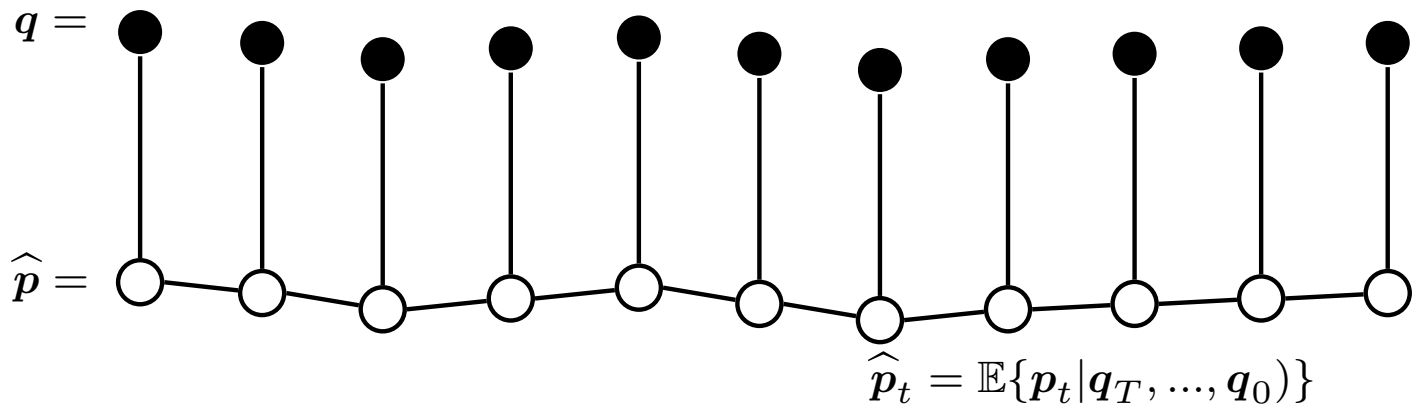
Recursive Bayesian patch estimation: filtering



Kalman filter: recursive computation of $\mathbb{P}(\boldsymbol{p}_t|\boldsymbol{q}_t, \ldots, \boldsymbol{q}_1) \sim \mathcal{N}\left(\widehat{\boldsymbol{p}}_t, P_t\right)$

$$\begin{cases} \widehat{\boldsymbol{p}}_t = (I - K_t)\widehat{\boldsymbol{p}}_{t-1} + K_t \boldsymbol{q}_t & \textit{state mean} \\ P_t = (I - K_t)(P_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (P_{t-1} + W_t)(P_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{Kalman gain} \end{cases}$$

Recursive Bayesian patch estimation: smoothing



Rauch-Tung-Streibel smoother: back-recursion for $\mathbb{P}(\boldsymbol{p}_t|\boldsymbol{q}_T, \ldots, \boldsymbol{q}_1) \sim \mathcal{N}\left(\widetilde{\boldsymbol{p}}_t, \widetilde{P}_t\right)$

$$\begin{cases} \widetilde{\boldsymbol{p}}_t = (I - S_t)\widehat{\boldsymbol{p}}_t + S_t\widetilde{\boldsymbol{p}}_{t+1} & \textit{state mean} \\ \widetilde{P}_t = P_t + S_t(\widetilde{P}_{t+1} - P_t - W_{t+1})S_t & \textit{state covariance} \\ S_t = P_t(P_t + W_{t+1})^{-1} & \textit{smoothing gain} \end{cases}$$

Parameter estimation

The only model parameters that need to be estimated are the state transition covariances W_t associated to the group.

$$\mathbb{E}\{(\boldsymbol{q}_t - \boldsymbol{q}_{t-1})(\boldsymbol{q}_t - \boldsymbol{q}_{t-1})^T\} = W_t + 2\sigma^2 I.$$

We assume that similar patches are iid realizations of the same dynamic model:

$$\boldsymbol{p}_{t,i} = \boldsymbol{p}_{t-1,i} + \boldsymbol{w}_{t,i} \text{ with } \boldsymbol{w}_{t,i} \sim \mathcal{N}(\mathbf{0}, W_t) \quad (1)$$

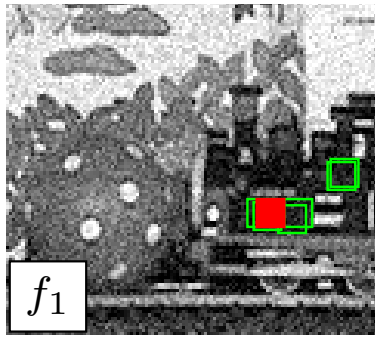
$$\boldsymbol{q}_{t,i} = \boldsymbol{p}_{t,i} + \boldsymbol{r}_{t,i} \text{ with } \boldsymbol{r}_{t,i} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I). \quad (2)$$

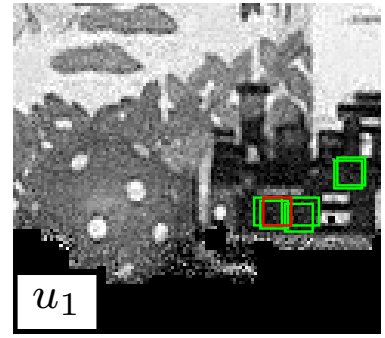
We estimate W_t via the sample covariance matrix of the innovations:

$$\widehat{W}_t = \beta \widehat{W}_{t-1} + (1-\beta) \left(\frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{q}_{t,i} - \boldsymbol{q}_{t-1,i})(\boldsymbol{q}_{t,i} - \boldsymbol{q}_{t-1,i})^T - 2\sigma^2 I \right)_+ .$$

A forgetting factor $\beta \in [0, 1]$ is introduced to increase temporal stability.

Forward NL-Kalman filter: frame 1





Extract n similar patches

$\{\boldsymbol{q}_{1,i}\}$

Wiener filtering

$\{\widehat{\boldsymbol{p}}_{1,i}\}$

Aggregate by averaging

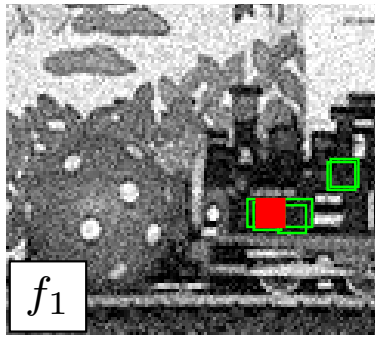
$\{\boldsymbol{q}_{1,i}\}$

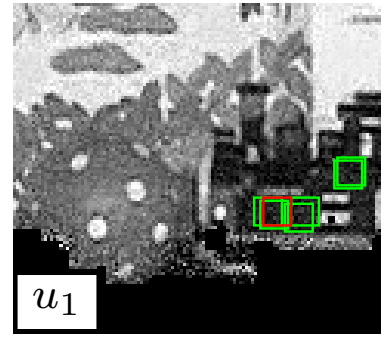
W_1

Estimate model parameters

Patch groups 1

Forward NL-Kalman filter: frame 1





Extract n similar patches

$\{\boldsymbol{q}_{1,i}\}$

Wiener filtering

$\{\widehat{\boldsymbol{p}}_{1,i}\}$

Aggregate by averaging

$\{\boldsymbol{q}_{1,i}\}$

W_1

$\{\widehat{\boldsymbol{p}}_{1,i}\}, P_1$

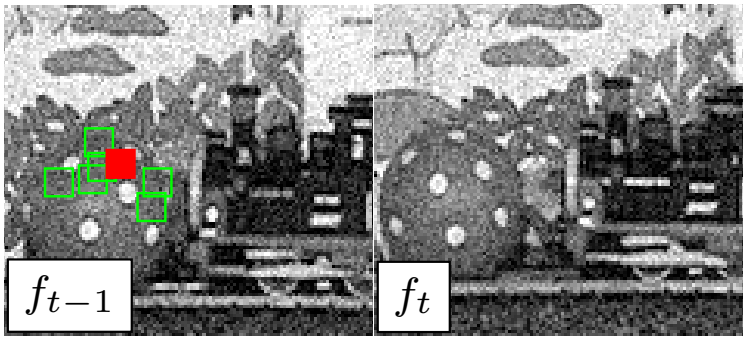
Estimate model parameters

W_1

coordinates

Patch groups 1

Forward NL-Kalman filter: frame t



Extract n similar patches

coordinates $t-1$

Kalman filtering

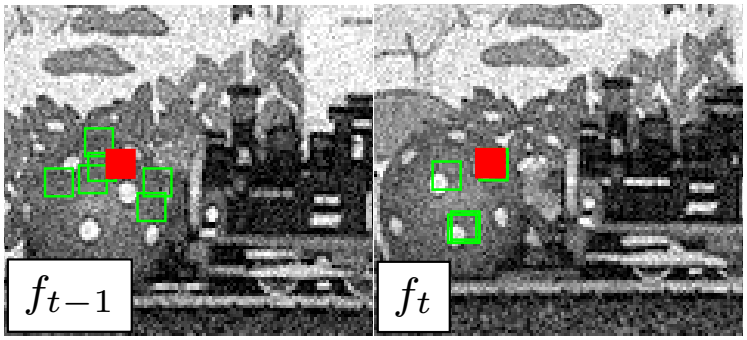
Aggregate by averaging

Estimate model parameters

Patch groups $t-1$

Patch groups t

Forward NL-Kalman filter: frame t



FOF
$t-1 \to t$

Extract n similar patches

Kalman filtering

Aggregate by averaging

coordinates $t-1$

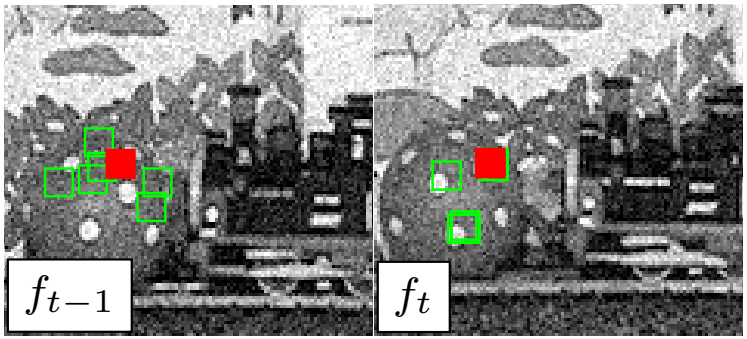
coordinates t

Estimate model parameters

Patch groups $t-1$

Patch groups t

Forward NL-Kalman filter: frame t



FOF
$t-1 \to t$

Extract n similar patches

Kalman filtering

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

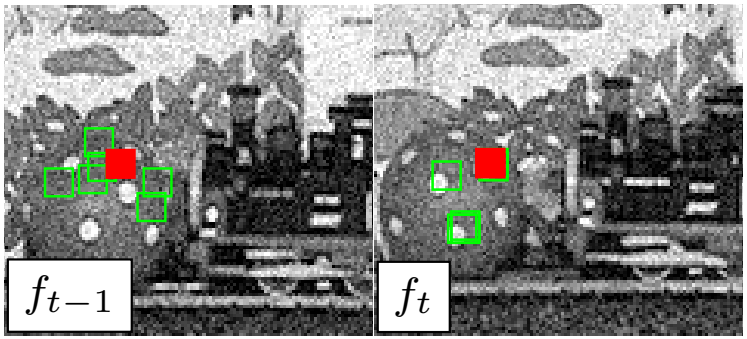
Estimate model parameters

W_{t-1}

Patch groups $t-1$

Patch groups t

Forward NL-Kalman filter: frame t



FOF
$t-1 \to t$

Extract n similar patches

Kalman filtering

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

W_t

Estimate model parameters

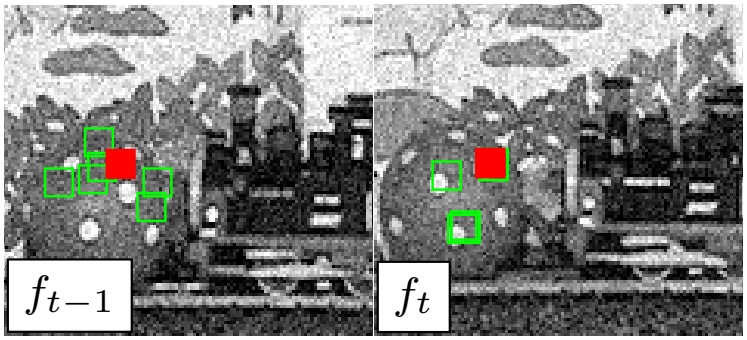
W_{t-1}

Patch groups $t-1$

W_t

Patch groups t

Forward NL-Kalman filter: frame t



FOF
$t-1 \to t$

Extract n similar patches

$\{q_{t,i}\}$

Kalman filtering

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

W_t

$\{\widehat{p}_{t,i}\}, P_t$

Estimate model parameters

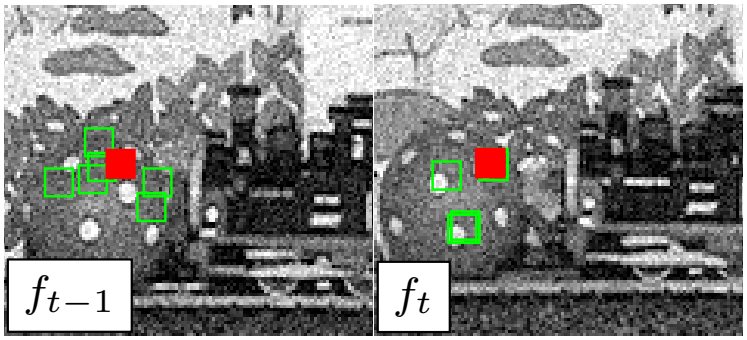
W_{t-1}

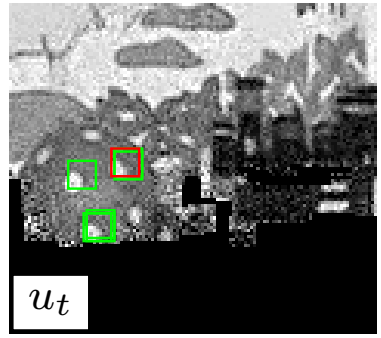
Patch groups $t-1$

$\{\widehat{p}_{t-1,i}\}, P_{t-1}$

Patch groups t

Forward NL-Kalman filter: frame t





FOF
$t-1 \to t$

Extract n similar patches

$\{q_{t,i}\}$

Kalman filtering

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

W_t

$\{\widehat{p}_{t,i}\}, P_t$

$\{\widehat{p}_{t,i}\}$

Estimate model parameters

W_{t-1}

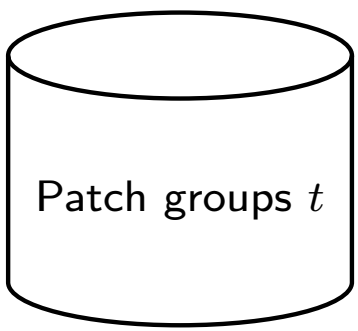
Patch groups $t-1$

$\{\widehat{p}_{t-1,i}\}, P_{t-1}$

Patch groups t

FNLK: Managing patch trajectories

- **Occlusions:** Terminate patch trajectories if an occlusion is detected.
- **Dis-occlusions:** Create groups for parts not covered by existing groups.

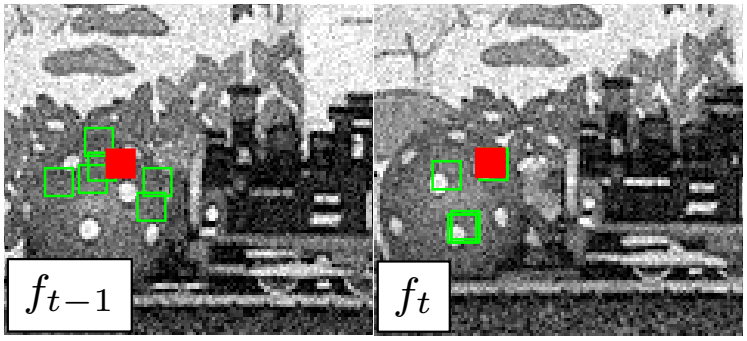


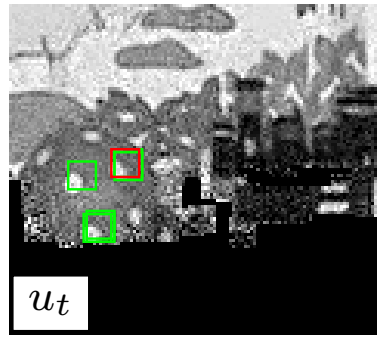
For each of these groups, we store

- coordinates of the patches in the group
- estimated clean patches $\widehat{\boldsymbol{p}}_{t,i}$
- covariance of estimated patches P_t
- transition covariance matrix $\widehat{W}_t$

Contents

Dropping the patch groups memory in FNLK





FOF
$t-1 \to t$

Extract n similar patches

$\{q_{t,i}\}$

Kalman filtering

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

W_t

$\{\widehat{p}_{t,i}\}, P_t$

$\{\widehat{p}_{t,i}\}$

Estimate model parameters

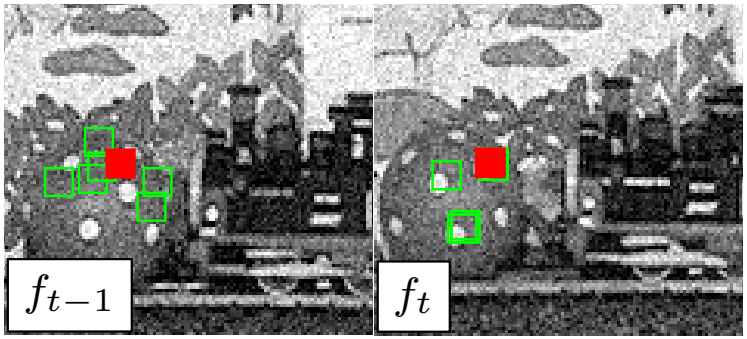
W_{t-1}

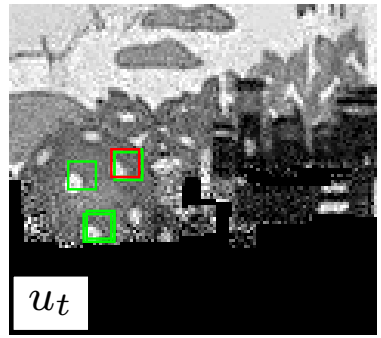
Patch groups $t-1$

$\{\widehat{p}_{t-1,i}\}, P_{t-1}$

Patch groups t

Dropping the patch groups memory in FNLK





FOF
$t-1 \to t$

Extract n similar patches

$\{q_{t,i}\}$

Kalman filtering

$\{\widehat{p}_{t,i}\}$

Aggregate by averaging

coordinates $t-1$

$\{q_{t,i}\}$
$\{q_{t-1,i}\}$

W_t

Estimate model parameters

W_{t-1}

Patch groups $t-1$

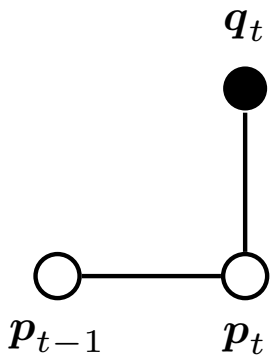
$\{\widehat{p}_{t-1,i}\}, P_{t-1}$

$\{\widehat{p}_{t,i}\}, P_t$

$\{\widehat{p}_{t,i}\}$

Patch groups t

Recursive Bayesian patch estimation w/out covariances



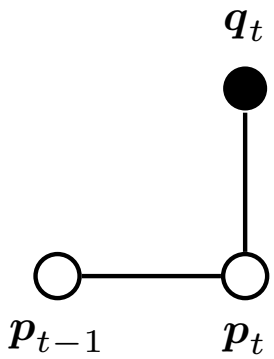
Given $\widehat{p}_{t-1}, P_{t-1}$, at time t we have

$$\begin{cases} p_{t-1} \sim \mathcal{N}(\widehat{p}_{t-1}, P_{t-1}), & \\ p_t = p_{t-1} + w_t, & w_t \sim \mathcal{N}(0, W_t), \\ q_t = p_t + r_t & r_t \sim \mathcal{N}(0, \sigma^2 I). \end{cases}$$

Kalman filter: Recursive computation of $\mathbb{P}(p_t | q_t, \dots, q_1) \sim \mathcal{N}\left(\widehat{p}_t, P_t\right)$

$$\begin{cases} \widehat{p}_t = (I - K_t)\widehat{p}_{t-1} + K_t q_t & \textit{state mean} \\ P_t = (I - K_t)(P_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (P_{t-1} + W_t)(P_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{Kalman gain} \end{cases}$$

Recursive Bayesian patch estimation w/out covariances



If we don't have $\widehat{p}_{t-1}$, P_{t-1} we introduce them as parameters

$$\begin{cases} p_{t-1} \sim \mathcal{N}(\,\mu_{t-1}, C_{t-1}\,), & \\ p_t = p_{t-1} + w_t, & w_t \sim \mathcal{N}(0, W_t), \\ q_t = p_t + r_t & r_t \sim \mathcal{N}(0, \sigma^2 I). \end{cases}$$

We have the following posterior $\mathbb{P}(p_t|q_t) \sim \mathcal{N}\left(\widehat{p}_t, P_t\right)$

$$\begin{cases} \widehat{p}_t = (I - K_t)\mu_{t-1} + K_t q_t & \textit{state mean} \\ P_t = (I - K_t)(C_{t-1} + W_t)(I - K_t)^T + \sigma^2 K_t^2 & \textit{state covariance} \\ K_t = (C_{t-1} + W_t)(C_{t-1} + W_t + \sigma^2 I)^{-1} & \textit{"Kalman" gain} \end{cases}$$

Parameter estimation for the memoryless model

We assume that similar patches are iid realizations of the same dynamic model:

$$\begin{cases} \boldsymbol{p}_{t-1,i} \sim \mathcal{N}(\boldsymbol{\mu}_{t-1}, C_{t-1}) & \\ \boldsymbol{p}_{t,i} = \boldsymbol{p}_{t-1,i} + \boldsymbol{w}_{t,i} & \boldsymbol{w}_{t,i} \sim \mathcal{N}(\mathbf{0}, W_t) \\ \boldsymbol{q}_{t,i} = \boldsymbol{p}_{t,i} + \boldsymbol{r}_{t,i} & \boldsymbol{r}_{t,i} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I). \end{cases}$$

$$\widehat{\boldsymbol{\mu}}_{t-1} = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{p}_{t-1,i}$$

$$\widehat{C}_{t-1} = \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{p}_{t-1,i} - \widehat{\boldsymbol{\mu}}_{t-1,i})(\boldsymbol{p}_{t-1,i} - \widehat{\boldsymbol{\mu}}_{t-1,i})^T$$

$$\widehat{W}_t = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{q}_{t,i} - \boldsymbol{p}_{t-1,i})(\boldsymbol{q}_{t,i} - \boldsymbol{p}_{t-1,i})^T - \sigma^2.$$

NOTE: We introduced m to control the spatial averaging in $\widehat{\boldsymbol{\mu}}_{t-1}$. Typically $m < n$.

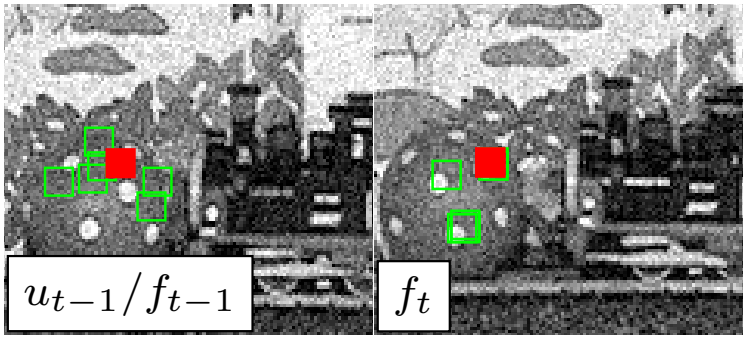
Additional simplification: work in DCT domain

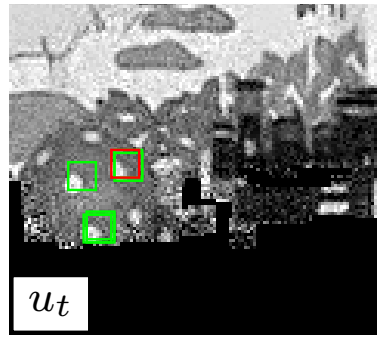
Assumption: $W_t = U\text{Diag}(\boldsymbol{\nu}_t)U^T$ where U is the DCT basis

Then:

1. $P_t = U\text{Diag}(\widehat{\boldsymbol{\nu}}_t)U^T$
2. The Kalman recursion separates in d scalar filters on each DCT component:

Backward NL-Kalman filter: frame t





FOF
$t-1 \to t$

Extract n similar patches

$\{ q_{t,i}\, ,\, \widehat{p}_{t-1,i} \}$

Kalman filtering

$\{ \widehat{p}_{t,i} \}$

Aggregate by averaging

$\{q_{t,i}\, ,\, \widehat{p}_{t-1,i}\}$

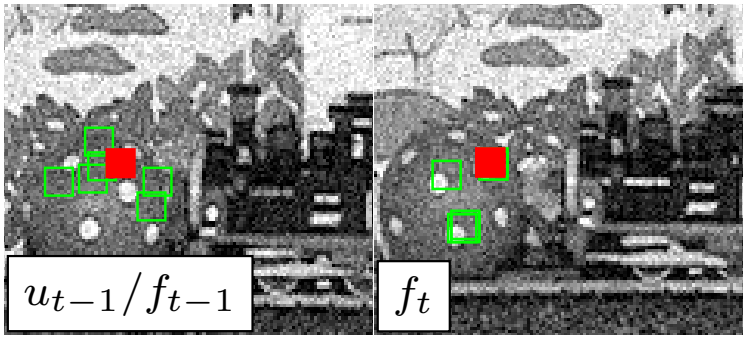
$\{q_{t-1,i}\}$

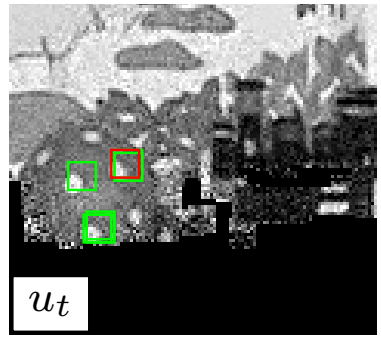
W_t

μ_{t-1}, C_{t-1}

Estimate model parameters

Backward NL-Kalman filter: frame t





BOF
$t \to t-1$

Extract n similar patches

$\{ \boldsymbol{q}_{t,i}\, , \widehat{\boldsymbol{p}}_{t-1,i} \}$

Kalman filtering

$\{ \widehat{\boldsymbol{p}}_{t,i} \}$

Aggregate by averaging

$\{ \boldsymbol{q}_{t,i}\, , \widehat{\boldsymbol{p}}_{t-1,i} \}$

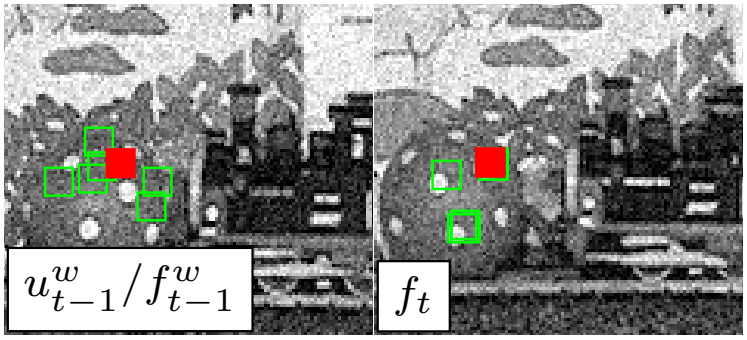
$\{ \boldsymbol{q}_{t-1,i} \}$

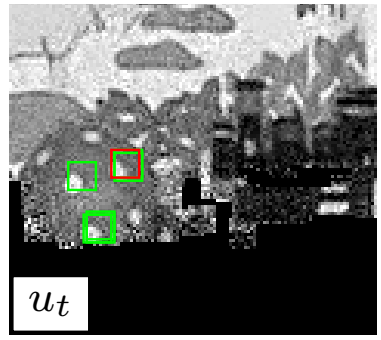
W_t

$\boldsymbol{\mu}_{t-1}, C_{t-1}$

Estimate model parameters

Backward NL-Kalman filter: frame t





Extract n similar patches

$\{q_{t,i}\,,\,\widehat{p}_{t-1,i}\}$

Kalman filtering

$\{\widehat{p}_{t,i}\}$

Aggregate by averaging

$\{q_{t,i}\,,\,\widehat{p}_{t-1,i}\}$

$\{q_{t-1,i}\}$

W_t

μ_{t-1}, C_{t-1}

Estimate model parameters

Backward NL-Kalman filtering and smoothing

Algorithm 1: Recursive video filtering

input : Noisy video f, noise level σ

output: Denoised video u

for $t = 1 \dots T$ **do**

- $\boldsymbol{v}_t^b \leftarrow \texttt{compute-optical-flow}(f_t, \hat{u}_{t-1}, \sigma)$
- $\hat{u}_{t-1}^w \leftarrow \texttt{warp-bicubic}(\hat{u}_{t-1}, \boldsymbol{v}_t^b)$
- $\hat{g}_t \leftarrow \texttt{bwd-nlkalman-filter}(f_t, \hat{u}_{t-1}^w, \sigma)$
- $\hat{u}_t \leftarrow \texttt{bwd-nlkalman-filter}(f_t, \hat{u}_{t-1}^w, \hat{g}_t, \sigma)$

Algorithm 2: Recursive video smoothing

input : Noisy video f, noise level σ

output: Denoised video u

for $t = 1 \dots T$ **do**

- $\boldsymbol{v}_t^f \leftarrow \texttt{compute-optical-flow}(\hat{u}_t, \tilde{u}_{t+1}, \sigma)$
- $\tilde{u}_{t+1}^w \leftarrow \texttt{warp-bicubic}(\tilde{u}_{t+1}, \boldsymbol{v}_t^b)$
- $\tilde{u}_t \leftarrow \texttt{bwd-nlkalman-smoother}(\hat{u}_t, \tilde{u}_{t-1}^w, \sigma)$

Three approaches based on Gaussian models of patches

VNLB

- Fixed 3D patch size
- No distinction between space and time
- Does not require OF
- Two iterations

FNLK

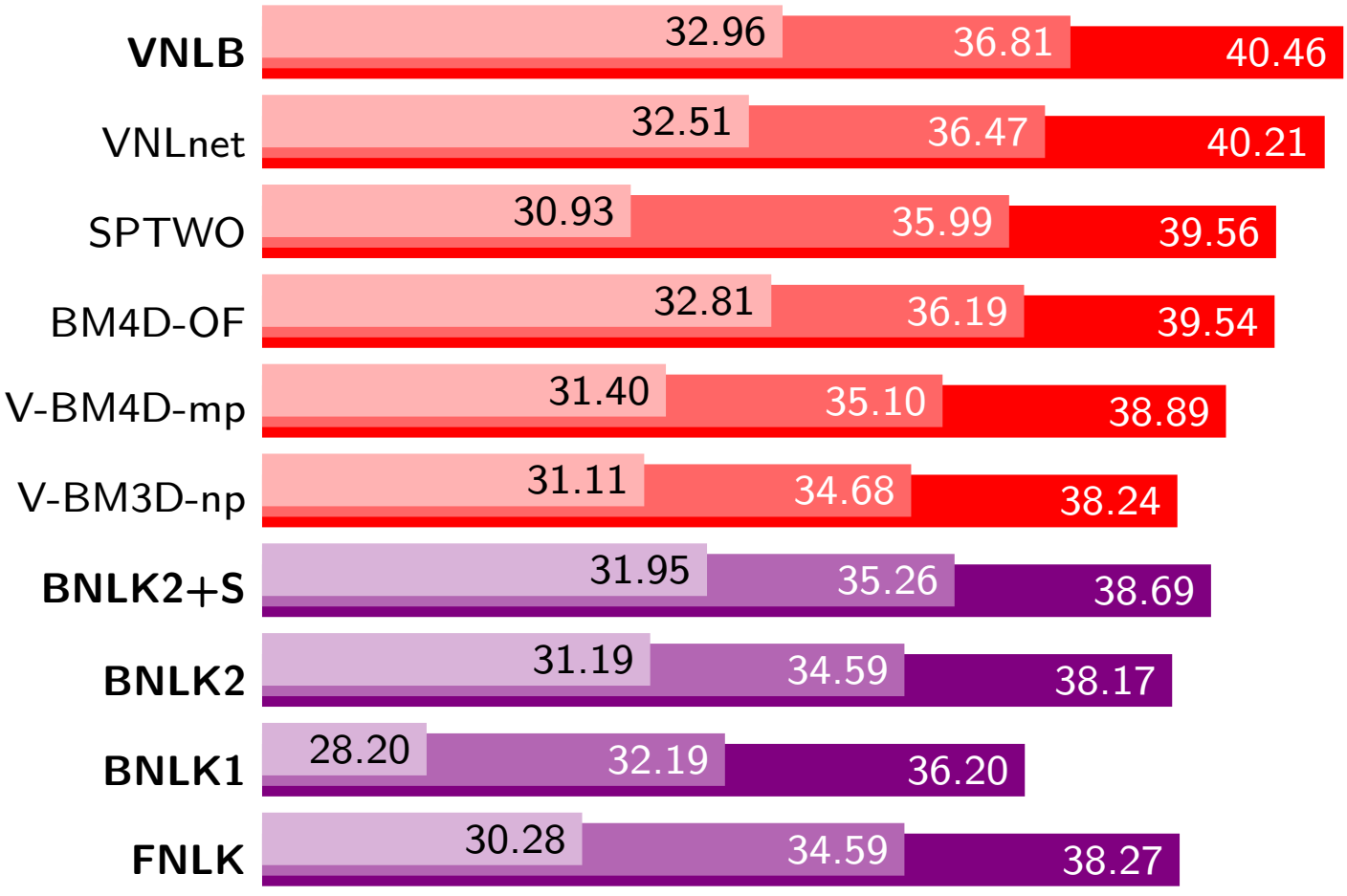
- Patch with arbitrary duration
- Processing organized by patch groups
- A lot of house-keeping
- Sensitive to OF
- Costly in memory

BNLK

- Patch with arbitrary duration (kind of)
- Very cheap in memory
- Processing by raster order
- DCT domain (for the moment)
- Two iterations
- Simple smoother
- Sensitive to OF

Contents

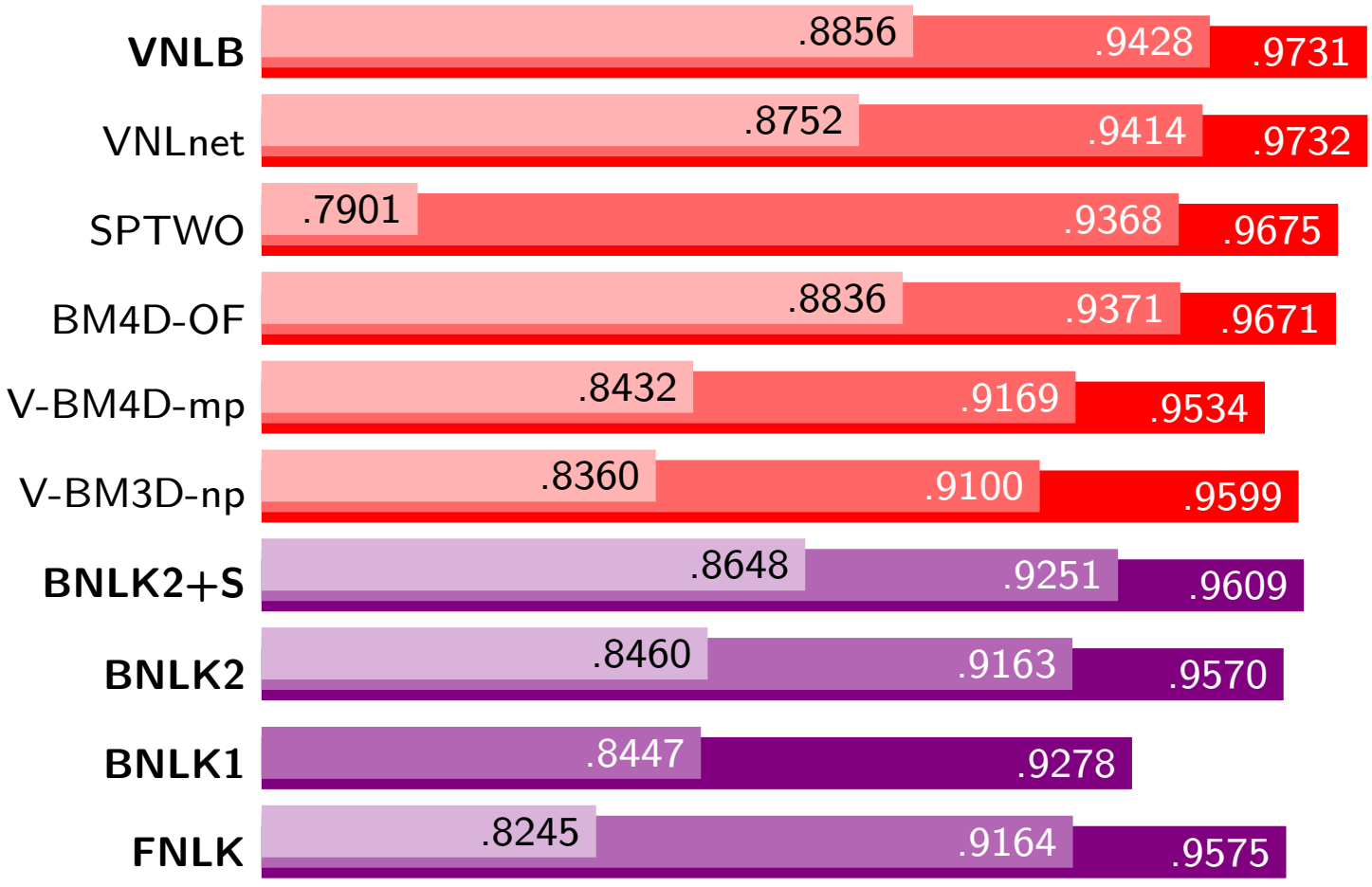
Quantitative denoising results (PSNR)



Average PSNR over 7 grayscale sequences $960 \times 540 \times 100$.

$\sigma = 10$ $\sigma = 20$ $\sigma = 40$

Quantitative denoising results (SSIM)



Average SSIM over 7 grayscale sequences $960 \times 540 \times 100$.

$\sigma = 10$ $\sigma = 20$ $\sigma = 40$

Conclusions and future work

- Current state-of-the-art in video denoising: either good results or fast results.
- Presented two recursive approaches bridging the gap between costly good methods and fast methods.
- They integrate information accross longer time ranges and are allow to recover many more details.
- Still very sensitive to optical flow.

Ongoing work

- Joint denoising and optical flow computation.
- Implement BNLK in an adaptive basis (instead of DCT).
- Fixed lag smoothers.
- Multiscale versions.

Thank you!

Reproducibility! Code and results:

- Non-recursive results:

  `http://dev.ipol.im/~pariasm/video_denoising_models/`
- VNLB: `http://github.com/pariasm/vnlb`
- SPTWO: `https://doi.org/10.5201/ipol.2018.224`
- BM4D-OF: `https://github.com/pariasm/vbm3d`
- BNLK: `http://github.com/pariasm/bwd-nlkalman`
- FNLK: `http://github.com/tehret/nlkalman`

References I

[Dabov'07] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian. *Image denoising by sparse 3D transform-domain collaborative filtering.* IEEE Trans. on Image Processing, 16, 2007.

[Mairal'09] J. Mairal, F. Bach, J. Ponce, G. Sapiro and A. Zisserman, *Non-local sparse models for image restoration.* CVPR 2009.

[Ji et al.'10] H. Ji, C. Liu, Z. Shen, Y. Xu, *Robust video denoising using low-rank matrix completion.* CVPR 2010.

[He et al.'11] Y. He, T. Gan, W. Chen, H. Wang, *Adaptive denoising by Singular Value Decomposition.* IEEE Signal Processing Letters, 18(4), 2011.

[Zoran'11] D. Zoran and Y. Weiss, *From learning models of natural image patches to whole image restoration,* ICCV 2011.

[Wang et al.'12] Wang S., Zhang L., Liang Y., *Nonlocal Spectral Prior Model for Low-Level Vision.* ACCV 2012.

[Yu et al.'12] G. Yu, G. Sapiro and S. Mallat, *Solving Inverse Problems With Piecewise Linear Estimators: From Gaussian Mixture Models to Structured Sparsity,* IEEE TIP, 21(5), 2012.

References II

[Dong et al.'13] Dong, W., Shi, G., Li, X., *Nonlocal Image Restoration With Bilateral Variance Estimation: A Low-Rank Approach*, IEEE TIP, 22(2), 2013.

[Lebrun'13] M. Lebrun, A. Buades and J.M. Morel. *A Nonlocal Bayesian image denoising algorithm.* SIAM Journal on Imaging Sciences, 6, 2013.

[Gu et al.'14] S. Gu, L. Zhang, W. Zuo, X. Feng, *Weighted Nuclear Norm Minimization with Application to Image Denoising*, CVPR 2014.

[Guo et al.'16] Q. Guo, C. Zhang, Y. Zhang and H. Liu, *An Efficient SVD-Based Method for Image Denoising*, IEEE Trans. on Circuits and Systems for Video Tech., 26(5), 2016.

[Badri et al.'16] H. Badri, H. Yahia and D. Aboutajdine, *Low-Rankness Transfer for Realistic Denoising*, in IEEE TIP, 25(12), 2016.

[Xie et al.'16] Y. Xie, S. Gu, Y. Liu, W. Zuo, W. Zhang and L. Zhang, *Weighted Schatten p-Norm Minimization for Image Denoising and Background Subtraction*," in IEEE TIP, 25(10), 2016.

Empirical Wiener filters in high dimensions

ML estimator of the inverse covariance matrix $Q_{\boldsymbol{y}} = C_{\boldsymbol{y}}^{-1}$ results from the following convex SDP:

$$\begin{aligned} &\max \ \log\det Q_{\boldsymbol{y}} - \operatorname{tr}(Q_{\boldsymbol{y}}\widehat{C}_{\boldsymbol{y}}) \\ &\text{subject to} \ \ 0 \prec Q_{\boldsymbol{y}} \preceq \sigma^{-2} I_d. \end{aligned}$$

Although there are efficient algorithms for solving such an SDP, they are still prohibitive for the present application.

Convergence of the sample covariance matrix

Spiked covariance model: Samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \sim \mathcal{N}(\mathbf{0}, C)$. We observe $\boldsymbol{y}_i \sim \mathcal{N}(\boldsymbol{x}_i, \sigma^2 I)$.

$$C = U \,\mathsf{Diag}(\lambda_1, \ldots, \lambda_m, 0, \ldots, 0)\, U^T.$$

Theorem (Paul 2007). Suppose that $d/n \to \gamma \in (0, 1)$ as $n \to \infty$. Let $\widehat{\xi}_i$ be the ith eigenvector of the sample covariance matrix $\widehat{C}_{\boldsymbol{y}}$. Then $\widehat{\xi}_i$ converges almost surely to

$$\widehat{\xi}_i \to \begin{cases} \sigma^2(1+\sqrt{\gamma})^2 & \text{if } \lambda_i \leqslant \sqrt{\gamma}\sigma^2, \\ f(\lambda_i) := (\lambda_i + \sigma^2)\left(1 + \dfrac{\gamma\sigma^2}{\lambda_i}\right) & \text{if } \lambda_i > \sqrt{\gamma}\sigma^2. \end{cases}$$

We can estimate λ_i as $\widehat{\lambda}_i^{\mathsf{S}}(\widehat{\xi}_i) = \begin{cases} 0 & \text{if } \widehat{\xi}_i \leqslant \sigma^2(1+\sqrt{\gamma})^2, \\ f^{-1}(\widehat{\xi}_i) & \text{if } \widehat{\xi}_i > \sigma^2(1+\sqrt{\gamma})^2. \end{cases}$

Convergence of the sample covariance matrix

Spiked covariance model: Samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \sim \mathcal{N}(\mathbf{0}, C)$. We observe $\boldsymbol{y}_i \sim \mathcal{N}(\boldsymbol{x}_i, \sigma^2 I)$.

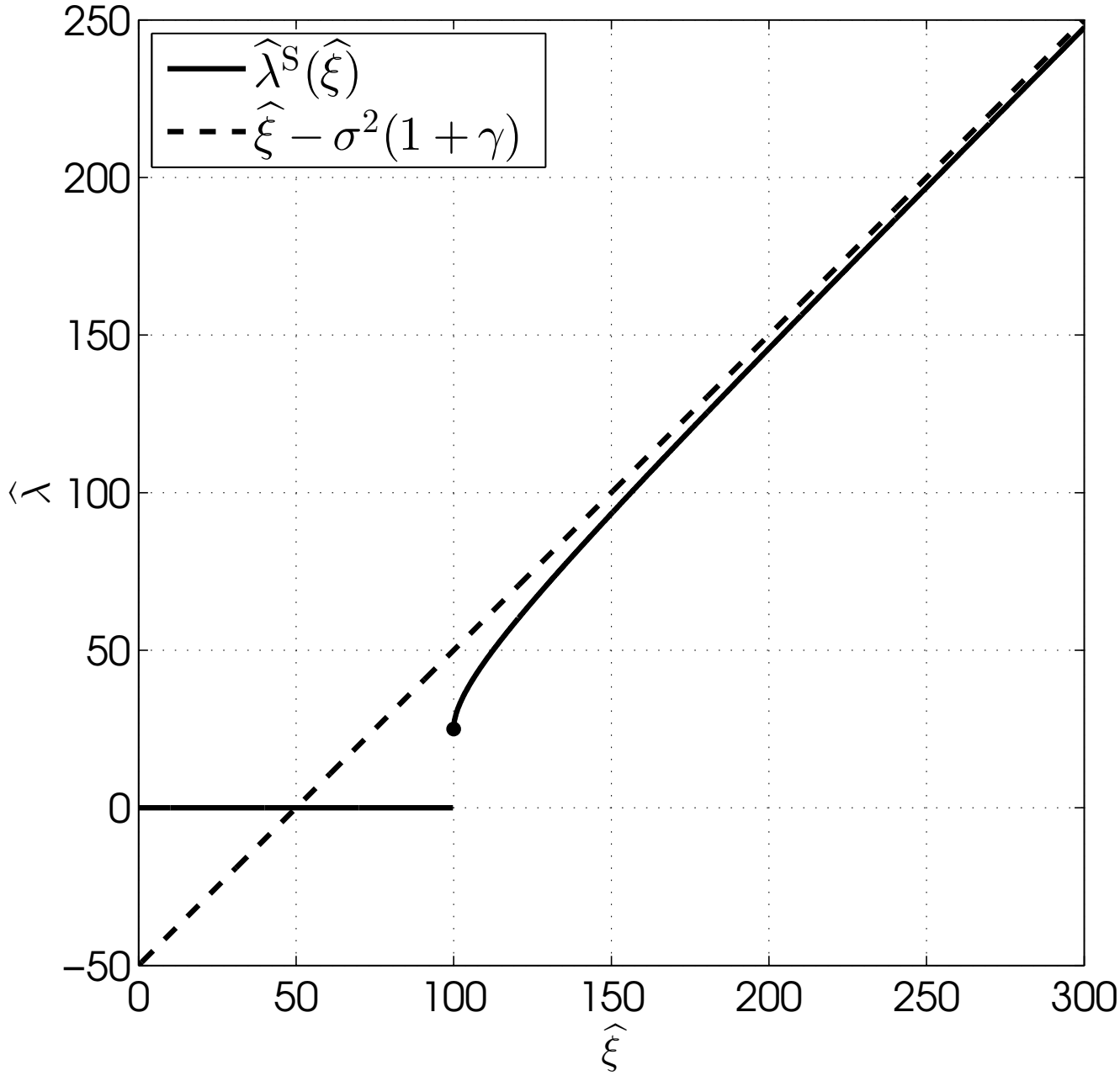
$$C = U \,\mathsf{Diag}(\lambda_1, \ldots, \lambda_m, 0, \ldots, 0)\, U^T.$$

Theorem (Paul 2007). Suppose that $d/n \to \gamma \in (0,1)$ as $n \to \infty$. Let $\widehat{\xi}_i$ be the ith eigenvector of the sample covariance matrix $\widehat{C}_{\boldsymbol{y}}$. Then $\widehat{\xi}_i$ converges almost surely to

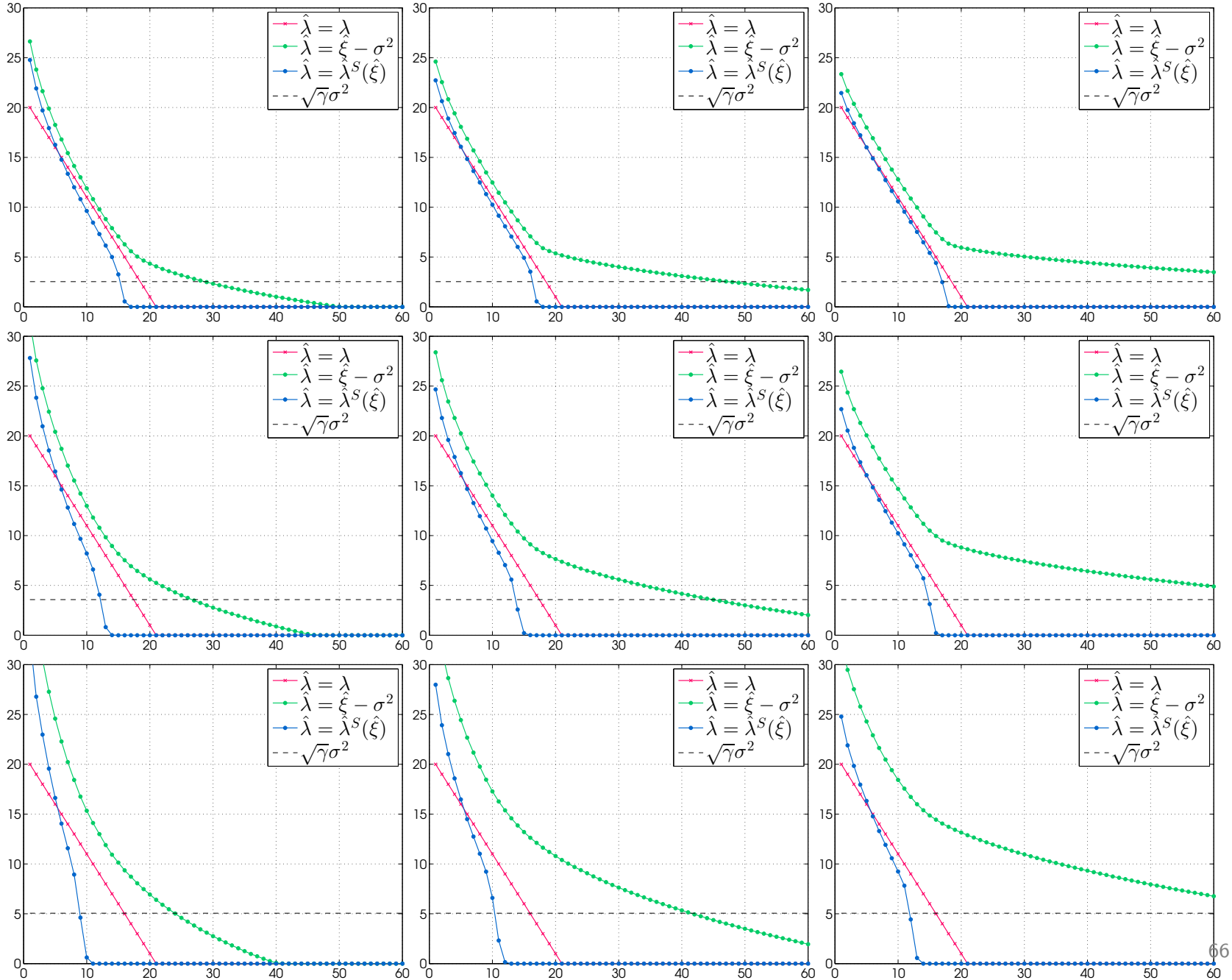
$$\widehat{\xi}_i \to \begin{cases} \sigma^2(1+\sqrt{\gamma})^2 & \text{if } \lambda_i \leqslant \sqrt{\gamma}\sigma^2, \\ f(\lambda_i) := (\lambda_i + \sigma^2)\left(1 + \dfrac{\gamma\sigma^2}{\lambda_i}\right) & \text{if } \lambda_i > \sqrt{\gamma}\sigma^2. \end{cases}$$

We can estimate λ_i as $\quad \widehat{\lambda}_i^{\mathsf{S}}(\widehat{\xi}_i) = \begin{cases} 0 & \text{if } \widehat{\xi}_i \leqslant \sigma^2(1+\sqrt{\gamma})^2, \\ f^{-1}(\widehat{\xi}_i) & \text{if } \widehat{\xi}_i > \sigma^2(1+\sqrt{\gamma})^2. \end{cases}$

Estimators for the eigenvalues of $C_{\boldsymbol{x}}$



Simulations



Variance threshold

We propose an additional estimator by hard thresholding the difference $\hat{\xi} - \sigma^2$. The value of the threshold is given by a parameter τ:

$$\hat{\lambda}_i^{\mathsf{H}} = \mathrm{H}_\tau(\hat{\xi}_i - \sigma^2) = \begin{cases} \hat{\xi} - \sigma^2 & \text{if } \hat{\xi} \geqslant \tau\sigma^2 \\ 0 & \text{if } \hat{\xi} < \tau\sigma^2. \end{cases}$$

Denoising performance of both estimators