MMSE Approximation for the Sparse Prior

Dror Simon

Computer Science, Technion, Israel

January 9, 2019

Joint work with Jeremias Sulam, Yaniv Romano, Yue M. Lu and Michael Elad
Why denoising?
A simple testing ground for novel concepts in signal processing.
Can be generalized to other, more complicated applications.
Noise Removal

Why denoising?
A simple testing ground for novel concepts in signal processing.
Can be generalized to other, more complicated applications.

Dror Simon (Technion)

MMSE for Sparse Prior

January 9, 2019 2 / 50
Why denoising?
Why denoising?

- A simple testing ground for novel concepts in signal processing.
Why denoising?

- A simple testing ground for novel concepts in signal processing.
- Can be generalized to other, more complicated applications.
Noise Removal

Noisy Signal
Noise Removal

Noisy Signal

Signal Restoration

Clean Estimate
Noise Removal

Signal Restoration

Clean Estimate

Prior Knowledge

Noisy Signal
Denoising
Noise Removal – Bayesian Standpoint

Denoising

Signal Estimation
Noise Removal – Bayesian Standpoint

Denoising → Signal Estimation → Approx.
Noise Removal – Bayesian Standpoint

Denoising → Suboptimal Signal Estimation → Approx.
Noise Removal – Bayesian Standpoint

Denoising

Suboptimal Signal Estimation

Optimal Signal Estimation

Approx.
Outline

1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
Outline

1 Bayesian Framework
   • The Generative Model
   • Bayesian Estimators

2 MMSE Approximation
   • Previous Work

3 Stochastic Resonance
   • Can Noise Help Denoising?

4 Our Proposed Method
   • The Algorithm
   • Unitary Case Analysis
   • The General Dictionary Case
   • Image Denoising

5 Conclusions
\( D \in \mathbb{R}^{n \times m} \) is a dictionary with normalized columns.
The Generative Model

Each element $i$ in $\alpha$ is non zero with probability $p_i \ll 1$. 

$D_{\alpha}$
The Generative Model

The non-zero elements of the sparse representation, denoted by $\alpha_s$, are sampled from a Gaussian distribution $\alpha_s | s \sim N(0, \sigma^2 \mathbf{I}_|s|)$. 

$$D \alpha^m \quad n$$
The product $D\alpha$ leads to a signal $x$. 

$$D\alpha = x$$
The Generative Model

We are given noisy measurements \( y = D\alpha + \nu \), where \( \nu \) is a white Gaussian noise \( \nu \sim \mathcal{N} (0, \sigma^2 \nu I_n) \).
The Generative Model – Results

The prior probability of a support (Bernoulli):

\[ p(s) = \prod_{i \in s} p_i \prod_{j / \in s} (1 - p_j). \]

When the support is known, \( y \) and \( \alpha_s \) are jointly Gaussian:

\[ y = D_s \alpha_s + \nu, \]

leading to \( y \mid s \) is Gaussian:

\[ y \mid s \sim N(0, C_s). \]

\( y \mid \alpha_s, s \) is Gaussian:

\[ y \mid \alpha_s, s \sim N(D_s \alpha_s, \sigma^2 \nu I_n). \]

\( \alpha_s \mid y, s \) is Gaussian:

\[ \alpha_s \mid y, s \sim N(1/\sigma^2 \nu Q_s^{-1} s D_s^T y, Q_s^{-1} s). \]

\[ C_s = \sigma^2 \alpha_s D_s D_s^T + \sigma^2 \nu I_n, \]

\[ Q_s = \frac{1}{\sigma^2 \nu} |s| I_n + \frac{1}{\sigma^2 \nu} D_s^T D_s. \]

---

\(^1\)Turek, Javier S., Irad Yavneh, and Michael Elad, 2011. "On MMSE and MAP denoising under sparse representation modeling over a unitary dictionary."
The prior probability of a support (Bernoulli):
\[ p(s) = \prod_{i \in s} p_i \prod_{j \notin s} (1 - p_j). \]
The prior probability of a support (Bernoulli):
\[ p(s) = \prod_{i \in s} p_i \prod_{j \notin s} (1 - p_j). \]

When the support is known, \( \mathbf{y} \) and \( \mathbf{\alpha}_s \) are jointly Gaussian
\[ \mathbf{y} = \mathbf{D}_s \mathbf{\alpha}_s + \mathbf{\nu}, \text{ leading to}^{1} \]

---

^{1} Turek, Javier S., Irad Yavneh, and Michael Elad, 2011. "On MMSE and MAP denoising under sparse representation modeling over a unitary dictionary."
The prior probability of a support (Bernoulli):
\[ p(s) = \prod_{i \in s} p_i \prod_{j \notin s} (1 - p_j). \]

When the support is known, \( y \) and \( \alpha_s \) are jointly Gaussian
\[ y = D_s \alpha_s + \nu, \] leading to
\[ y | s \] is Gaussian: \( y | s \sim \mathcal{N}(0, C_s). \)

---

\(^1\)Turek, Javier S., Irad Yavneh, and Michael Elad, 2011. "On MMSE and MAP denoising under sparse representation modeling over a unitary dictionary."
The prior probability of a support (Bernoulli):
\[ p(s) = \prod_{i \in s} p_i \prod_{j \notin s} (1 - p_j). \]

When the support is known, \( y \) and \( \alpha_s \) are jointly Gaussian
\[ y = D_s \alpha_s + \nu, \]
leading to\(^1\)
\[ y | s \text{ is Gaussian: } y | s \sim \mathcal{N}(0, C_s). \]
\[ y | \alpha_s, s \text{ is Gaussian: } y | \alpha_s, s \sim \mathcal{N}(D_s \alpha_s, \sigma^2 \nu I_n). \]

---

\(^1\)Turek, Javier S., Irad Yavneh, and Michael Elad, 2011. "On MMSE and MAP denoising under sparse representation modeling over a unitary dictionary."
The prior probability of a support (Bernoulli):
\[ p(s) = \prod_{i \in s} p_i \prod_{j \notin s} (1 - p_j). \]

When the support is known, \( y \) and \( \alpha_s \) are jointly Gaussian
\[ y = D_s \alpha_s + \nu, \]
leading to\(^1\)

- \( y|s \) is Gaussian: \( y|s \sim \mathcal{N}(0, C_s) \).
- \( y|\alpha_s, s \) is Gaussian: \( y|\alpha_s, s \sim \mathcal{N}(D_s \alpha_s, \sigma^2 \nu I_n) \).
- \( \alpha_s|y, s \) is Gaussian: \( \alpha_s|y, s \sim \mathcal{N}\left(\frac{1}{\sigma^2 \nu} Q^{-1}_s D^T_s y, Q^{-1}_s\right) \).

\[ C_s = \sigma^2 \alpha D_s D^T_s + \sigma^2 \nu I_n, \quad Q_s = \frac{1}{\sigma^2 \alpha} |s| + \frac{1}{\sigma \nu} D^T_s D_s \]

\(^1\) Turek, Javier S., Irad Yavneh, and Michael Elad, 2011. "On MMSE and MAP denoising under sparse representation modeling over a unitary dictionary."
1. **Bayesian Framework**
   - The Generative Model
   - Bayesian Estimators

2. **MMSE Approximation**
   - Previous Work

3. **Stochastic Resonance**
   - Can Noise Help Denoising?

4. **Our Proposed Method**
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. **Conclusions**
The goal: Estimate $\alpha$ given the noisy measurements $y$, i.e. denoise the signal.
The goal: Estimate $\alpha$ given the noisy measurements $y$, i.e. denoise the signal.

Many estimators can be proposed. We focus our attention on three:
The goal: Estimate $\alpha$ given the noisy measurements $y$, i.e. denoise the signal.

Many estimators can be proposed. We focus our attention on three:

1. The oracle estimator.
The goal: Estimate $\alpha$ given the noisy measurements $y$, i.e. denoise the signal.

Many estimators can be proposed. We focus our attention on three:

1. The oracle estimator.
2. The Maximum A-posteriori Probability (MAP) estimator.
Bayesian Estimators

The goal: Estimate $\alpha$ given the noisy measurements $y$, i.e. denoise the signal.

Many estimators can be proposed. We focus our attention on three:

1. The oracle estimator.
2. The Maximum A-posteriori Probability (MAP) estimator.
3. The Minimum Mean Square Error (MMSE) estimator.
The Oracle Estimator

The Oracle Estimator assumes knowledge of the support $\alpha_s | s, y$, which is Gaussian $\Rightarrow$ it's MAP and MMSE estimators are identical.

$\hat{\alpha}_{\text{Oracle}} = E\{\alpha_s | s, y\} = \frac{1}{\sigma^2} \nu Q^{-1} s^T y$

Cannot be obtained in practice. We will use it as a basic building block later in this talk.
The Oracle Estimator

- Assumes knowledge of the support $s$. 

\[
\hat{\alpha}_{\text{Oracle}}(s) = E\{\alpha_s | s, y\} = \frac{1}{\sigma^2} \nu \Omega^{-1} s D^T s y
\]

Cannot be obtained in practice. We will use it as a basic building block later in this talk.
The Oracle Estimator

- Assumes knowledge of the support $s$.
- $\alpha_s | y, s$ is Gaussian $\implies$ it’s MAP and MMSE estimators are identical.
The Oracle Estimator

- Assumes knowledge of the support $s$.
- $\alpha_s | y, s$ is Gaussian $\implies$ it’s MAP and MMSE estimators are identical.

Oracle Estimator

$$\hat{\alpha}_s^{\text{Oracle}} = \mathbb{E} \{ \alpha_s | s, y \} = \frac{1}{\sigma^2_Y} Q_s^{-1} D_s^T y$$

Cannot be obtained in practice. We will use it as a basic building block later in this talk.
The Oracle Estimator

- Assumes knowledge of the support $s$.
- $\alpha_s | y, s$ is Gaussian $\implies$ it’s MAP and MMSE estimators are identical.

Oracle Estimator

\[ \hat{\alpha}_s^{\text{Oracle}} = \mathbb{E} \{ \alpha_s | s, y \} = \frac{1}{\sigma^2} Q_s^{-1} D_s^T y \]

- Cannot be obtained in practice.
The Oracle Estimator

- Assumes knowledge of the support $s$.
- $\alpha_s|y, s$ is Gaussian $\implies$ it’s MAP and MMSE estimators are identical.

**Oracle Estimator**

\[
\hat{\alpha}_s^{\text{Oracle}} = \mathbb{E}\{\alpha_s|s, y\} = \frac{1}{\sigma^2} Q_s^{-1} D_s^T y
\]

- Cannot be obtained in practice.
- We will use it as a basic building block later in this talk.
The MAP Estimator

Instead of approximating the representation $\alpha$, we approximate the support itself.

**MAP Support Estimator**

$$\hat{s}_{\text{MAP}} = \arg \max_s p(s|y) = \arg \max_s p(s)p(y|s) = \arg \max_s -\frac{1}{2}y^T C^{-1} s - \frac{1}{2} \log \det (C^s) + \sum_{i \in s} \log (p_i) + \sum_{j \neq i \in s} \log (1 - p_j)$$

$s \in \{0, 1\}$

The sparse representation $\hat{\alpha}_{\text{MAP}}$, is obtained using the oracle estimator on the recovered support:

$$\hat{\alpha}_{\text{MAP}} = \hat{\alpha}_{\text{Oracle}} s_{\text{MAP}}.$$
The MAP Estimator

Instead of approximating the representation $\alpha$, we approximate the support itself.
Instead of approximating the representation $\alpha$, we approximate the support itself.

MAP Support Estimator

$\hat{s}_{\text{MAP}} = \arg \max_s p(s|y)$
The MAP Estimator

Instead of approximating the representation $\alpha$, we approximate the support itself.

**MAP Support Estimator**

\[ \hat{s}_{\text{MAP}} = \arg \max_s p(s|y) = \arg \max_s p(s) p(y|s) \]
The MAP Estimator

Instead of approximating the representation $\alpha$, we approximate the support itself.

**MAP Support Estimator**

\[
\hat{s}_{\text{MAP}} = \arg\max_s p(s|y) = \arg\max_s p(s)p(y|s)
\]

\[
= \arg\max_s -\frac{1}{2} y^T C_s^{-1} y - \frac{1}{2} \log \det(C_s)
\]

\[
+ \sum_{i \in s} \log(p_i) + \sum_{j \notin s} \log(1 - p_j)
\]

The sparse representation $\hat{\alpha}_{\text{MAP}}$, is obtained using the oracle estimator on the recovered support: $\hat{\alpha}_{\text{MAP}} = \hat{\alpha}_{\text{Oracle}} s_{\text{MAP}}$. 

Dror Simon (Technion)
Instead of approximating the representation $\alpha$, we approximate the support itself.

**MAP Support Estimator**

\[
\hat{s}_{\text{MAP}} = \arg \max_s p(s|y) = \arg \max_s p(s)p(y|s)
\]

\[
= \arg \max_s -\frac{1}{2} y^T C_s^{-1} y - \frac{1}{2} \log \det (C_s)
\]

\[
+ \sum_{i \in s} \log (p_i) + \sum_{j \notin s} \log (1 - p_j)
\]

$s \in \{0, 1\}^m$
The MAP Estimator

Instead of approximating the representation $\alpha$, we approximate the support itself.

**MAP Support Estimator**

\[
\hat{s}_{\text{MAP}} = \arg \max_s p(s|y) = \arg \max_s p(s) p(y|s)
\]

\[
= \arg \max_s -\frac{1}{2} y^T C_s^{-1} y - \frac{1}{2} \log \det (C_s)
\]

\[
+ \sum_{i \in s} \log (p_i) + \sum_{j \notin s} \log (1 - p_j)
\]

$s \in \{0, 1\}^m$

The sparse representation $\hat{\alpha}_{\text{MAP}}$, is obtained using the oracle estimator on the recovered support: $\hat{\alpha}_{\text{MAP}} = \hat{\alpha}_{s_{\text{MAP}}}$. 
The MMSE Estimator

The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[ \hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}} \mathbb{E} \left\{ \| \hat{\alpha}(y) - \alpha \|^2 \mid y \right\} = \mathbb{E} \{ \alpha \mid y \} = \mathbb{E}_{s \mid y} \{ \mathbb{E}_{\alpha \mid y, s} \{ \hat{\alpha} \mid y, s \} \} = \sum_{s \in \{0,1\}} m_p(s \mid y) \hat{\alpha}_{\text{Oracle}} \]

The MMSE is the sum of all the possible oracle estimators, weighted by the probability of the support.

The MMSE estimator is not sparse at all!
The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[
\hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E} \left\{ \| \hat{\alpha}(y) - \alpha \|_2^2 \mid y \right\}
\]

The MMSE is the sum of all the possible oracle estimators, weighted by the probability of the support.

The MMSE estimator is not sparse at all!
The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[ \hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E} \left\{ \left\| \hat{\alpha}(y) - \alpha \right\|_2^2 \mid y \right\} = \mathbb{E} \{ \alpha \mid y \} \]
The MMSE Estimator

The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

**MMSE Estimator**

\[
\hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E} \left\{ \| \hat{\alpha}(y) - \alpha \|_2^2 \bigg| y \right\} = \mathbb{E} \left\{ \alpha \bigg| y \right\}
\]

\[
= \mathbb{E}_{s \mid y} \left\{ \mathbb{E}_{\alpha \mid y, s} \{ \alpha \mid y, s \} \right\}
\]
The MMSE Estimator

The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[ \hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E}\{ \| \hat{\alpha}(y) - \alpha \|_2^2 \mid y \} = \mathbb{E}\{ \alpha \mid y \} \]

\[ = \mathbb{E}_{s \mid y} \{ \mathbb{E}_{\alpha \mid y, s}\{ \alpha \mid y, s \} \} = \sum_{s \in \{0,1\}^m} p(s \mid y) \hat{\alpha}_s^{\text{Oracle}} \]
The MMSE Estimator

The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[
\hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E} \left\{ \| \hat{\alpha}(y) - \alpha \|^2 \mid y \right\} = \mathbb{E} \left\{ \alpha \mid y \right\}
\]

\[
= \mathbb{E}_{s \mid y} \left\{ \mathbb{E}_{\alpha \mid y, s} \left\{ \alpha \mid y, s \right\} \right\} = \sum_{s \in \{0,1\}^m} p(s \mid y) \hat{\alpha}_{\text{Oracle}}^s
\]

The MMSE is the sum of all the possible oracle estimators, weighted by the probability of the support.
The MMSE Estimator

The Minimum Mean Square Error (MMSE) estimator minimizes the MSE of the estimator.

\[ \hat{\alpha}_{\text{MMSE}} = \arg \min_{\hat{\alpha}(y)} \mathbb{E} \left\{ \| \hat{\alpha}(y) - \alpha \|^2 \mid y \right\} = \mathbb{E} \left\{ \alpha \mid y \right\} \]

\[ = \mathbb{E}_{s \mid y} \left\{ \mathbb{E}_{\alpha \mid y, s} \left\{ \alpha \mid y, s \right\} \right\} = \sum_{s \in \{0,1\}^m} p(s \mid y) \hat{\alpha}_s^{\text{Oracle}} \]

The MMSE is the sum of all the possible oracle estimators, weighted by the probability of the support.

The MMSE estimator is not sparse at all!
Both estimators are practically impossible to obtain.
The MMSE and Map Estimators

- Both estimators are practically impossible to obtain.
  - MAP – compute the posterior probability of each of the $2^m$ supports and pick the most probable one.
  - MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.

How is this issue resolved?
- MAP – use an approximation algorithm (greedy or relaxed) to recover a likely support, and then use the oracle.
- MMSE – usually avoided.

Can we do better?
Both estimators are practically impossible to obtain.

- MAP – compute the posterior probability of each of the $2^m$ supports and pick the most probable one.
- MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.
The MMSE and Map Estimators

Both estimators are practically impossible to obtain.
- MAP – compute the posterior probability of each of the \( 2^m \) supports and pick the most probable one.
- MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.

How is this issue resolved?
- MAP – use an approximation algorithm (greedy or relaxed) to recover a likely support, and then use the oracle.
- MMSE – usually avoided.

Can we do better?
Both estimators are practically impossible to obtain.
- MAP – compute the posterior probability of each of the $2^m$ supports and pick the most probable one.
- MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.

How is this issue resolved?
- MAP – use an approximation algorithm (greedy or relaxed) to recover a likely support, and then use the oracle.
Both estimators are practically impossible to obtain.
- MAP – compute the posterior probability of each of the $2^m$ supports and pick the most probable one.
- MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.

How is this issue resolved?
- MAP – use an approximation algorithm (greedy or relaxed) to recover a likely support, and then use the oracle.
- MMSE – usually avoided.
Both estimators are practically impossible to obtain.
- MAP – compute the posterior probability of each of the $2^m$ supports and pick the most probable one.
- MMSE – compute the posterior probability of each support and use them as weights for all the possible oracle estimators.

How is this issue resolved?
- MAP – use an approximation algorithm (greedy or relaxed) to recover a likely support, and then use the oracle.
- MMSE – usually avoided.

Can we do better?
Outline

1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
1. **Bayesian Framework**
   - The Generative Model
   - Bayesian Estimators

2. **MMSE Approximation**
   - Previous Work

3. **Stochastic Resonance**
   - Can Noise Help Denoising?

4. **Our Proposed Method**
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. **Conclusions**
Random OMP

A modified version of the OMP algorithm. OMP: picks the atom most correlated with the current residual. RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly. Repeated many times leading to a variety of solutions. Averages the solutions to retrieve a final estimate. Asymptotically converges with the MMSE estimator when: The dictionary is unitary. The cardinality of the sparse representation is 1. Empirically achieves better MSE than OMP even when these conditions are not met.

Other methods exist (Schniter, P. et al. 2008 “Fast Bayesian matching pursuit”).

Elad, Michael, and Irad Yavneh, 2009. ”A plurality of sparse representations is better than the sparsest one alone.”
Random OMP

- A modified version of the OMP algorithm.

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.

Other methods exist (Schniter, P. et al. 2008 “Fast Bayesian matching pursuit”).

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.

Repeated many times leading to a variety of solutions. Averages the solutions to retrieve a final estimate. Asymptotically converges with the MMSE estimator when:
  - The dictionary is unitary.
  - The cardinality of the sparse representation is 1.

Empirically achieves better MSE than OMP even when these conditions are not met.

Other methods exist (Schniter, P. et al. 2008 “Fast Bayesian matching pursuit”).

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.
- Repeated many times leading to a variety of solutions.

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.
- Repeated many times leading to a variety of solutions.
- Averages the solutions to retrieve a final estimate.

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.

- Repeated many times leading to a variety of solutions.
- Averages the solutions to retrieve a final estimate.
- Asymptotically converges with the MMSE estimator when:
  - The dictionary is unitary.
  - The cardinality of the sparse representation is 1.

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.

- Repeated many times leading to a variety of solutions.
- Averages the solutions to retrieve a final estimate.
- Asymptotically converges with the MMSE estimator when:
  - The dictionary is unitary.
  - The cardinality of the sparse representation is 1.

- Empirically achieves better MSE than OMP even when these conditions are not met.

Elad, Michael, and Irad Yavneh, 2009. "A plurality of sparse representations is better than the sparsest one alone."
Random OMP

- A modified version of the OMP algorithm.
  - OMP: picks the atom most correlated with the current residual.
  - RandOMP: weights the unpicked atoms according to their correlation with the residual, and chooses randomly.
- Repeated many times leading to a variety of solutions.
- Averages the solutions to retrieve a final estimate.
- Asymptotically converges with the MMSE estimator when:
  - The dictionary is unitary.
  - The cardinality of the sparse representation is 1.
- Empirically achieves better MSE than OMP even when these conditions are not met.

Other methods exist (Schniter, P. et al. 2008 ”Fast Bayesian matching pursuit”)

Elad, Michael, and Irad Yavneh, 2009. ”A plurality of sparse representations is better than the sparsest one alone.”
Why do these methods work?
Why do these methods work?

- $p(s|y)$ has an exponential nature.
Why do these methods work?

- $p(s|y)$ has an exponential nature. $\Rightarrow$ The estimator is dominated by a small set of solutions.
MMSE Approximation Methods

Why do these methods work?

- $p(s|y)$ has an exponential nature. $\Rightarrow$ The estimator is dominated by a small set of solutions.

$$\hat{\alpha}_{\text{MMSE}} = \sum_{s \in \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}} \approx \sum_{s \in \omega \subset \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}}$$

These methods find a “dominant” subset $\omega$ of supports and approximate their weights (posterior probabilities).

These methods operate in a greedy fashion.

$\Rightarrow$ They are impractical for high dimensional signals.
Why do these methods work?

- $p(s|y)$ has an exponential nature. $\Rightarrow$ The estimator is dominated by a small set of solutions.

\[
\hat{\alpha}_{\text{MMSE}} = \sum_{s \in \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}} \approx \sum_{s \in \omega \subset \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}}
\]

- These methods find a “dominant” subset $\omega$ of supports and approximate their weights (posterior probabilities).
Why do these methods work?

- \( p(s|y) \) has an exponential nature. \( \Rightarrow \) The estimator is dominated by a small set of solutions.

\[
\hat{\alpha}_{\text{MMSE}} = \sum_{s \in \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}} \approx \sum_{s \in \omega \subset \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}}
\]

- These methods find a “dominant” subset \( \omega \) of supports and approximate their weights (posterior probabilities).

These methods operate in a greedy fashion.
MMSE Approximation Methods

Why do these methods work?

- $p(s|y)$ has an exponential nature. $\Rightarrow$ The estimator is dominated by a small set of solutions.

$$\hat{\alpha}_{\text{MMSE}} = \sum_{s \in \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}} \approx \sum_{s \in \omega \subseteq \{0,1\}^m} p(s|y) \hat{\alpha}_s^{\text{Oracle}}$$

- These methods find a “dominant” subset $\omega$ of supports and approximate their weights (posterior probabilities).

These methods operate in a greedy fashion.

$\Rightarrow$ They are impractical for high dimensional signals.
1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
Outline

1 Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2 MMSE Approximation
   - Previous Work

3 Stochastic Resonance
   - Can Noise Help Denoising?

4 Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5 Conclusions
Stochastic Resonance

**Definition**

- Originally, was suggested as an explanation to the periodic recurrence of ice ages.
- Today, broadly applied to describe a more general phenomenon where presence of noise in a *nonlinear* system provides a better response.
Stochastic Resonance

Definition

- Originally, was suggested as an explanation to the periodic recurrence of ice ages.
- Today, broadly applied to describe a more general phenomenon where presence of noise in a nonlinear system provides a better response.

Noise improves system performance?
Dither

- In signal quantization, additive noise is used to create stochastic quantization error.
- For example in images, dither prevents color banding which creates unpleasant images.
Dither

- In signal quantization, additive noise is used to create stochastic quantization error.
- For example in images, dither prevents color banding which creates unpleasant images.
Dither – Noise Has a Constructive Value

Dither

- In signal quantization, additive noise is used to create stochastic quantization error.
- For example in images, dither prevents color banding which creates unpleasant images.

![Image showing dither effect in images]
Dither – Noise Has a Constructive Value

Dither

- In signal quantization, additive noise is used to create stochastic quantization error.
- For example in images, dither prevents color banding which creates unpleasant images.
Dither – Noise Has a Constructive Value

Dither

- In signal quantization, additive noise is used to create stochastic quantization error.
- For example in images, dither prevents color banding which creates unpleasant images.
Outline

1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
1 Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2 MMSE Approximation
   - Previous Work

3 Stochastic Resonance
   - Can Noise Help Denoising?

4 Our Proposed Method
   - The Algorithm
     - Unitary Case Analysis
     - The General Dictionary Case
     - Image Denoising

5 Conclusions
The Proposed Algorithm

\[ y \]
The Proposed Algorithm

n

y

y
The Proposed Algorithm

\[ y + n \]

Dror Simon (Technion)

MMSE for Sparse Prior

January 9, 2019 25 / 50
The Proposed Algorithm

\[ y + n \rightarrow \text{MAP} \]
The Proposed Algorithm

\[ y + n \rightarrow \text{MAP} \rightarrow S \]
The Proposed Algorithm

\[ y + n \rightarrow \text{MAP} \]

\[ y + n \rightarrow \text{Oracle} \]

\[ \text{Oracle} \rightarrow S \]
The Proposed Algorithm

Dror Simon (Technion)

MMSE for Sparse Prior

January 9, 2019 25 / 50
Algorithm 2: Stochastic Resonance MMSE Approximation

input : $y, D, \text{PursuitMethod}, \sigma_n, K$

for $k \in 1 \ldots K$
do
$
n_k \leftarrow \text{SampleNoise}(\sigma_n)$

$
\tilde{\alpha}_k \leftarrow \text{PursuitMethod}(y + n_k, D)$

$
\hat{S}_k \leftarrow \text{Support}(\tilde{\alpha}_k)$

$
\hat{\alpha}_k \leftarrow \hat{\alpha}_{\text{Oracle}}(\hat{S}_K, y)$
endo

$\hat{\alpha} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k$
Algorithm 2: Stochastic Resonance MMSE Approximation

input : $y, D, \text{PursuitMethod}, \sigma_n, K$

output: $\hat{\alpha}$
Algorithm 2: Stochastic Resonance MMSE Approximation

**input**: $y, D, \text{PursuitMethod}, \sigma_n, K$

**output**: $\hat{\alpha}$

**for** $k \in 1...K$ **do**

- $n_k \leftarrow \text{SampleNoise}(\sigma_n)$
- $\tilde{\alpha}_k \leftarrow \text{PursuitMethod}(y + n_k, D)$
- $\hat{S}_k \leftarrow \text{Support}(\tilde{\alpha}_k)$
- $\hat{\alpha}_k \leftarrow \hat{\alpha} \text{Oracle} \hat{S}_K(y)$

**end**

$\hat{\alpha} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k$
Algorithm 2: Stochastic Resonance MMSE Approximation

**input**: $y, D, \text{PursuitMethod}, \sigma_n, K$

**output**: $\hat{\alpha}$

for $k \in 1...K$ do

$n_k \leftarrow \text{SampleNoise}(\sigma_n)$

end
Algorithm 2: Stochastic Resonance MMSE Approximation

input : y, D, PursuitMethod, $\sigma_n$, K
output: $\hat{\alpha}$

for $k \in 1...K$ do
  $n_k \leftarrow $ SampleNoise($\sigma_n$)
  $\tilde{\alpha}_k \leftarrow $ PursuitMethod($y + n_k$, D)

$\hat{\alpha} \leftarrow \frac{1}{K}\sum_{k=1}^{K} \hat{\alpha}_k$
Algorithm 2: Stochastic Resonance MMSE Approximation

input : \( y, D, \text{PursuitMethod}, \sigma_n, K \)

output: \( \hat{\alpha} \)

for \( k \in 1...K \) do

\( n_k \leftarrow \text{SampleNoise}(\sigma_n) \)

\( \tilde{\alpha}_k \leftarrow \text{PursuitMethod}(y + n_k, D) \)

\( \hat{S}_k \leftarrow \text{Support}(\tilde{\alpha}_k) \)

end

\( \hat{\alpha} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k \)
The Proposed Algorithm

Algorithm 2: Stochastic Resonance MMSE Approximation

**input**: $y$, $D$, PursuitMethod, $\sigma_n$, $K$

**output**: $\hat{\alpha}$

for $k \in 1...K$ do

    $n_k \leftarrow$ SampleNoise($\sigma_n$)
    $\tilde{\alpha}_k \leftarrow$ PursuitMethod($y + n_k$, $D$)
    $\hat{S}_k \leftarrow$ Support($\tilde{\alpha}_k$)
    $\hat{\alpha}_k \leftarrow \hat{\alpha}_{\hat{S}_K}^\mathrm{Oracle}(y)$

end

$\hat{\alpha} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k$
Algorithm 2: Stochastic Resonance MMSE Approximation

\textbf{input}: y, D, PursuitMethod, σ_n, K

\textbf{output}: \hat{\alpha}

\textbf{for} k \in 1 \ldots K \ \textbf{do}

\hspace{1em} n_k \leftarrow \text{SampleNoise}(\sigma_n)

\hspace{1em} \tilde{\alpha}_k \leftarrow \text{PursuitMethod}(y + n_k, D)

\hspace{1em} \hat{S}_k \leftarrow \text{Support}(\tilde{\alpha}_k)

\hspace{1em} \hat{\alpha}_k \leftarrow \hat{\alpha}_{\hat{S}_K}^{\text{Oracle}}(y)

\textbf{end}

\hat{\alpha} \leftarrow \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k
The pursuit itself is independent from the rest of the process $\Rightarrow$ Relaxation methods are just as applicable.
The pursuit itself is independent from the rest of the process
⇒ Relaxation methods are just as applicable.
The Proposed Algorithm – Does It Work?

- $D \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
The Proposed Algorithm – Does It Work?

- $D \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
- $\|\alpha\|_0 = 1$, $\alpha_s \sim \mathcal{N}(0, 1)$.
The Proposed Algorithm – Does It Work?

- \( D \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \| \alpha \|_0 = 1, \alpha_s \sim \mathcal{N}(0, 1) \).
- \( \nu \sim \mathcal{N}(0, \sigma^2_\nu I_{50}) , \sigma_\nu = 0.2. \)
The Proposed Algorithm – Does It Work?

- $D \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
- $\|\alpha\|_0 = 1$, $\alpha_s \sim \mathcal{N}(0, 1)$.
- $\nu \sim \mathcal{N}(0, \sigma_\nu^2 I_{50})$, $\sigma_\nu = 0.2$.
- 100 iterations of stochastic resonance.
The Proposed Algorithm – Does It Work?

- \( D \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \| \alpha \|_0 = 1, \alpha_s \sim \mathcal{N}(0, 1) \).
- \( \nu \sim \mathcal{N}(0, \sigma^2 \nu I_{50}), \sigma_\nu = 0.2 \).
- 100 iterations of stochastic resonance.
1 Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2 MMSE Approximation
   - Previous Work

3 Stochastic Resonance
   - Can Noise Help Denoising?

4 Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5 Conclusions
When $D$ is a unitary matrix ($D^T D = I$), the oracle, MAP and MMSE estimators are element-wise shrinkage operators:

$c$ and $\lambda_{\text{MAP}}$ depend on $p_i$, $\sigma_\alpha$ and $\sigma_\nu$. 
When $D$ is a unitary matrix ($D^T D = I$), the oracle, MAP and MMSE estimators are element-wise shrinkage operators:

**Oracle**

$$\hat{\alpha}_s^{\text{Oracle}}(y) = c^2 D_s^T y = c^2 \beta_s$$

$c$ and $\lambda_{\text{MAP}}$ depend on $p_j$, $\sigma_\alpha$ and $\sigma_\nu$.  

Dror Simon (Technion)
Unitary Case - Estimators

When \( \mathbf{D} \) is a unitary matrix \( (\mathbf{D}^T \mathbf{D} = \mathbf{I}) \), the oracle, MAP and MMSE estimators are element-wise shrinkage operators:

\[
\hat{\alpha}_{\text{MAP}} (\mathbf{y}) = \mathcal{H}_{\lambda_{\text{MAP}}} \left( \mathbf{D}^T \mathbf{y} \right) = \mathcal{H}_{\lambda_{\text{MAP}}} (\mathbf{\beta})
\]

\( \hat{\alpha}_{\text{MAP}} \) depends on \( p_i, \sigma_{\alpha} \) and \( \sigma_{\nu} \).
When \( \mathbf{D} \) is a unitary matrix \( (\mathbf{D}^T \mathbf{D} = \mathbf{I}) \), the oracle, MAP and MMSE estimators are element-wise shrinkage operators:

\[
\hat{\alpha}_{\text{MAP}} (\mathbf{y}) = \mathcal{H}_{\lambda_{\text{MAP}}} (\mathbf{D}^T \mathbf{y}) = \mathcal{H}_{\lambda_{\text{MAP}}} (\beta)
\]

\[
\mathcal{H}_{\lambda_{\text{MAP}}} (\beta_i) = \begin{cases} 
  c^2 \beta_i & \text{if } |\beta_i| \geq \lambda_{\text{MAP}}, \\
  0 & \text{else}
\end{cases}
\]

\( c \) and \( \lambda_{\text{MAP}} \) depend on \( p_i, \sigma_\alpha \) and \( \sigma_\nu \).
When $\mathbf{D}$ is a unitary matrix ($\mathbf{D}^T \mathbf{D} = \mathbf{I}$), the oracle, MAP and MMSE estimators are element-wise shrinkage operators:

\[
\hat{\alpha}_i^{\text{MMSE}}(\beta_i) = \frac{\exp \left( \frac{c^2}{2\sigma^2_\nu} \beta_i^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}}{1 + \exp \left( \frac{c^2}{2\sigma^2_\nu} \beta_i^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}} c^2 \beta_i
\]

$c$ and $\lambda_{\text{MAP}}$ depend on $p_i$, $\sigma_\alpha$ and $\sigma_\nu$. 
Unitary Case – Stochastic Resonance

To use our proposed algorithm we need to provide a pursuit.
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable.
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable. \[\implies \text{Use it!}\]
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable. \(\Rightarrow\) Use it!

**Algorithm 1**

**Subtractive hard-thresholding** \(\mathcal{H}^- (\cdot)\)

\[
\mathcal{H}^- (\beta, \tilde{n}) = \begin{cases} 
\text{c} & \text{if } |\beta + \tilde{n}| \geq \lambda_{\text{MAP}}, \\
0 & \text{otherwise}.
\end{cases}
\]
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable. \(\implies\) Use it!

**Subtractive hard-thresholding** \(\mathcal{H}^- (\cdot)\)

\[
\mathcal{H}^- (\beta, \tilde{n}) = \begin{cases} 
  c^2\beta & \text{if } |\beta + \tilde{n}| \geq \lambda_{\text{MAP}}, \\
  0 & \text{otherwise.}
\end{cases}
\]
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable. \( \implies \) Use it!

**Subtractive hard-thresholding \( \mathcal{H}^- (\cdot) \)**

\[
\mathcal{H}^- (\beta, \tilde{n}) = \begin{cases} 
  c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda_{\text{MAP}}, \\
  0 & \text{otherwise}. 
\end{cases}
\]
To use our proposed algorithm we need to provide a pursuit. The MAP is attainable. \( \Longrightarrow \) Use it!

**Subtractive hard-thresholding** \( \mathcal{H}^- (\cdot) \)

\[
\mathcal{H}^- (\beta, \tilde{n}) = \begin{cases} 
    c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\
    0 & \text{otherwise.}
\end{cases}
\]
What happens as \( K \to \infty \)?
Unitary Case – Asymptotic Estimator

What happens as $K \to \infty$?

$$\hat{\alpha}(\beta, \tilde{n}) = \mathcal{H}^{-}(\beta, \tilde{n}) = \begin{cases} c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\ 0 & \text{otherwise}. \end{cases}$$
What happens as $K \to \infty$?

$$\hat{\alpha} (\beta, \tilde{n}) = H^{-} (\beta, \tilde{n}) = \begin{cases} c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\ 0 & \text{otherwise}. \end{cases}$$

$$\hat{\alpha} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k = \mathbb{E}_n \{ \hat{\alpha}_k \}$$
What happens as $K \to \infty$?

$$\hat{\alpha}(\beta, \tilde{n}) = H^{-}(\beta, \tilde{n}) = \begin{cases} c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\ 0 & \text{otherwise.} \end{cases}$$

$$\hat{\alpha} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k = \mathbb{E}_n \{ \hat{\alpha}_k \} = \int_{-\infty}^{\infty} H^{-}(\beta, \tilde{n}) p(\tilde{n}) \, d\tilde{n}$$
Unitary Case – Asymptotic Estimator

What happens as $K \to \infty$?

$$\hat{\alpha} (\beta, \tilde{n}) = H^- (\beta, \tilde{n}) = \begin{cases} c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\ 0 & \text{otherwise.} \end{cases}$$

$$\hat{\alpha} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k = \mathbb{E}_n \{ \hat{\alpha}_k \} = \int_{-\infty}^{\infty} H^- (\beta, \tilde{n}) p (\tilde{n}) d\tilde{n} \quad = \ldots$$
Unitary Case – Asymptotic Estimator

What happens as $K \to \infty$?

$$\hat{\alpha} (\beta, \tilde{n}) = \mathcal{H}^-(\beta, \tilde{n}) = \begin{cases} c^2 \beta & \text{if } |\beta + \tilde{n}| \geq \lambda, \\ 0 & \text{otherwise.} \end{cases}$$

$$\hat{\alpha} = \lim_{K \to \infty} \frac{1}{K} \sum_{k=1}^{K} \hat{\alpha}_k = E_n \{\hat{\alpha}_k\} = \int_{-\infty}^{\infty} \mathcal{H}^- (\beta, \tilde{n}) p (\tilde{n}) d\tilde{n}$$

$$= c^2 \beta \left[ Q \left( \frac{\lambda + \beta}{\sigma_n} \right) + Q \left( \frac{\lambda - \beta}{\sigma_n} \right) \right]$$

$$Q(x) = \int_{x}^{\infty} e^{-\frac{t^2}{2}} dt$$
How does this estimator perform?
Unitary Case – Empirical Performance

How does this estimator perform?

![Graph: MMSE vs Subtractive SR](image-url)
How does this estimator perform?

![Graph showing MMSE vs Subtractive SR](image)
Stochastic Resonance vs. MMSE

Are the two the same?

Stochastic Resonance & MMSE

\[
\hat{\alpha}_{\text{stochastic}} = c^2 \beta \left[ Q \left( \frac{\lambda + \beta}{\sigma_n} \right) + Q \left( \frac{\lambda - \beta}{\sigma_n} \right) \right]
\]

\[
\hat{\alpha}_{\text{MMSE}} = \frac{\exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}}{1 + \exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}} c^2 \beta
\]

No... But are they close?

Empirically yes, but for the right choice of parameters.

We can set the parameters by using SURE.

More information in our paper.
Stochastic Resonance vs. MMSE

Are the two the same?

\[ \hat{\alpha}_{\text{stochastic}} = c^2 \beta \left[ Q \left( \frac{\lambda + \beta}{\sigma_n} \right) + Q \left( \frac{\lambda - \beta}{\sigma_n} \right) \right] \]

\[ \hat{\alpha}_{\text{MMSE}} = \frac{\exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}}{1 + \exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}} c^2 \beta \]

No... But are they close?
Are the two the same?

Stochastic Resonance & MMSE

\[ \hat{\alpha}_{\text{stochastic}} = c^2 \beta \left[ Q \left( \frac{\lambda + \beta}{\sigma_n} \right) + Q \left( \frac{\lambda - \beta}{\sigma_n} \right) \right] \]

\[ \hat{\alpha}_{\text{MMSE}} = \frac{\exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}}{1 + \exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1-c^2}} c^2 \beta \]

No... But are they close?
Empirically yes, but for the right choice of parameters.
Are the two the same?

**Stochastic Resonance & MMSE**

\[ \hat{\alpha}_{\text{stochastic}} = c^2 \beta \left[ Q \left( \frac{\lambda + \beta}{\sigma_n} \right) + Q \left( \frac{\lambda - \beta}{\sigma_n} \right) \right] \]

\[ \hat{\alpha}_{\text{MMSE}} = \frac{\exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1 - c^2}}{1 + \exp \left( \frac{c^2}{2\sigma_v^2} \beta^2 \right) \frac{p_i}{1-p_i} \sqrt{1 - c^2}} c^2 \beta \]

No... But are they close?
Empirically yes, but for the right choice of parameters.
We can set the parameters by using SURE.
Stochastic Resonance vs. MMSE

Are the two the same?

Stochastic Resonance & MMSE

\[ \hat{\alpha}_{\text{stochastic}} = c^2 \beta \left[ Q\left(\frac{\lambda + \beta}{\sigma_n}\right) + Q\left(\frac{\lambda - \beta}{\sigma_n}\right) \right] \]

\[ \hat{\alpha}_{\text{MMSE}} = \exp\left(\frac{c^2}{2\sigma^2} \beta^2\right) \frac{p_i}{1-p_i} \sqrt{1-c^2} \]

No... But are they close?
Empirically yes, but for the right choice of parameters.
We can set the parameters by using SURE.
More information in our paper.
What about non-unitary cases?
Applicable when the closed form solution of the MMSE is not attainable (i.e. when $p_i$ is not known).

What about non-unitary cases?
Unitary Case – Summary

Applicable when the closed form solution of the MMSE is not attainable (i.e. when $p_i$ is not known).
Applicable when the closed form solution of the MMSE is not attainable (i.e. when $p_i$ is not known).

What about non-unitary cases?
1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
The General Dictionary Case

The general case is harder to analyze. The MAP estimator is exhaustive, \( \Rightarrow \) use a pursuit algorithm instead. The performance (and the analysis) depends on the pursuit used.

We separate two cases:

- The generative model's parameters are known.
- The generative model's parameters are not known.
The General Dictionary Case

- The general case is harder to analyze.
The General Dictionary Case

- The general case is harder to analyze.
- The MAP estimator is exhaustive $\implies$ use a pursuit algorithm instead.
The General Dictionary Case

- The general case is harder to analyze.
- The MAP estimator is exhaustive $\implies$ use a pursuit algorithm instead.
- The performance (and the analysis) depends on the pursuit used.
The General Dictionary Case

- The general case is harder to analyze.
- The MAP estimator is exhaustive $\implies$ use a pursuit algorithm instead.
- The performance (and the analysis) depends on the pursuit used.
- We separate to two cases:
The general case is harder to analyze.

The MAP estimator is exhaustive \(\Rightarrow\) use a pursuit algorithm instead.

The performance (and the analysis) depends on the pursuit used.

We separate to two cases:

- The generative model’s parameters are known.
The general case is harder to analyze.

The MAP estimator is exhaustive \(\implies\) use a pursuit algorithm instead.

The performance (and the analysis) depends on the pursuit used.

We separate to two cases:

- The generative model’s parameters are known.
- The generative model’s parameters are not known.
General Case – Known Parameters

The parameters are known ⇒ use them.

Replace the arithmetic mean in the proposed algorithm with a weighted sum.

Similar to FBMP.

But unlike FBMP - no greedy measures required.

⇒ Applicable for large dimensions.

In our paper we prove it is equivalent to a Monte Carlo importance sampling simulation.

⇒ Asymptotically converges to the MMSE!
The parameters are known $\implies$ use them.
The parameters are known $\Rightarrow$ use them.

Replace the arithmetic mean in the proposed algorithm with a weighted sum.
General Case – Known Parameters

- The parameters are known → use them.
- Replace the arithmetic mean in the proposed algorithm with a weighted sum.
- Similar to FBMP.
General Case – Known Parameters

- The parameters are known $\implies$ use them.
- Replace the arithmetic mean in the proposed algorithm with a weighted sum.
- Similar to FBMP.
- But unlike FBMP - no greedy measures required.
The parameters are known \( \Rightarrow \) use them.

Replace the arithmetic mean in the proposed algorithm with a weighted sum.

Similar to FBMP.

But unlike FBMP - no greedy measures required.

\( \Rightarrow \) Applicable for large dimensions.
The parameters are known \( \implies \) use them.

Replace the arithmetic mean in the proposed algorithm with a weighted sum.

Similar to FBMP.

But unlike FBMP - no greedy measures required.

\( \implies \) Applicable for large dimensions.

In our paper we prove it is equivalent to a Monte Carlo importance sampling simulation.
The parameters are known $\Rightarrow$ use them.

Replace the arithmetic mean in the proposed algorithm with a weighted sum.

Similar to FBMP.

But unlike FBMP - no greedy measures required.

$\Rightarrow$ Applicable for large dimensions.

In our paper we prove it is equivalent to a Monte Carlo importance sampling simulation.

$\Rightarrow$ Asymptotically converges to the MMSE!
General Case – Known Parameters

- $\mathbf{D} \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
- $\|\alpha\|_0 = 1, \alpha_s \sim \mathcal{N}(0, 1)$.
- $\nu \sim \mathcal{N}(0, \sigma^2_\nu I_{50})$, $\sigma_\nu = 0.2$.
- 35 iterations of stochastic resonance.
General Case – Known Parameters

- \( D \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \|\alpha\|_0 = 1, \alpha_s \sim \mathcal{N}(0, 1) \).
- \( \nu \sim \mathcal{N}(0, \sigma^2 \nu I_{50}) \), \( \sigma_\nu = 0.2 \).
- 35 iterations of stochastic resonance.

![Graph showing SR Estimators]
Parameters’ values are lacking.
Parameters’ values are lacking. $\implies$ the same MMSE is no longer attainable.
• Parameters’ values are lacking. $\implies$ the same MMSE is no longer attainable.
• How can we estimate a support?
General Case – Unknown Parameters

- Parameters’ values are lacking. \( \Rightarrow \) the same MMSE is no longer attainable.
- How can we estimate a support? Use any pursuit (MAP approximation).
Parameters’ values are lacking. \(\implies\) the same MMSE is no longer attainable.

How can we estimate a support? Use any pursuit (MAP approximation).

How can we obtain the oracle estimator?
Parameters’ values are lacking. \(\Longrightarrow\) the same MMSE is no longer attainable.

How can we estimate a support? Use any pursuit (MAP approximation).

How can we obtain the oracle estimator? Simply use least squares
\[
\hat{\alpha}(s, y)_{\text{oracle}} = \left(D_s^T D_s\right)^{-1} D_s^T y.
\]
General Case – Unknown Parameters

\[ y + n = S \]

MAP

Oracle

S
General Case – Unknown Parameters

\[ y + n \]

Pursuit Algo.

Least Squares

\[ S \]
General Case – Unknown Parameters

- \( \mathbf{D} \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \| \mathbf{\alpha} \|_0 = 1, \mathbf{\alpha}_s \sim \mathcal{N} (0, 1). \)
- \( \nu \sim \mathcal{N} (\mathbf{0}, \sigma^2_{\nu} \mathbf{I}_{50}) , \sigma_{\nu} = 0.2. \)
- 100 iterations of stochastic resonance.
General Case – Unknown Parameters

- $\mathbf{D} \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
- $\|\alpha\|_0 = 1$, $\alpha_s \sim \mathcal{N}(0, 1)$.
- $\nu \sim \mathcal{N}(0, \sigma^2 \nu I_{50})$, $\sigma^2 = 0.2$.

100 iterations of stochastic resonance.

![Graph showing SR Estimators](image-url)
General Case – Unknown Parameters

- \( D \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \| \alpha \|_0 = 1 \, p_i = 0.05, \alpha_s \sim \mathcal{N}(0, 1) \).
- \( \nu \sim \mathcal{N}(0, \sigma_\nu^2 I_{50}) \), \( \sigma_\nu = 0.2 \).
- 100 iterations of stochastic resonance.
General Case – Unknown Parameters

- \( \mathbf{D} \in \mathbb{R}^{50 \times 100} \) a normalized random dictionary.
- \( \|\alpha\|_0 = 1 \), \( p_i = 0.05 \), \( \alpha_s \sim \mathcal{N}(0, 1) \).
- \( \mathbf{v} \sim \mathcal{N}(\mathbf{0}, \sigma^2_\nu \mathbf{I}_{50}) \), \( \sigma_\nu = 0.2 \).
- 100 iterations of stochastic resonance.

Use bounded noise formulation for the pursuit:

\[
\text{(OMP)} \quad \min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{D}\alpha\|_2 \leq \epsilon, \\
\text{(BP)} \quad \min_{\alpha} \|\alpha\|_1 \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{D}\alpha\|_2 \leq \epsilon.
\]
- $D \in \mathbb{R}^{50 \times 100}$ a normalized random dictionary.
- $\|\alpha\|_0 = 1$, $p_i = 0.05$, $\alpha_s \sim \mathcal{N}(0, 1)$.
- $\nu \sim \mathcal{N}(0, \sigma^2_\nu I_{50})$, $\sigma_\nu = 0.2$.
- 100 iterations of stochastic resonance.

![Graph showing MSE for different SR estimators: SR with OMP, OMP, SR with BP, BP. The x-axis represents n, the y-axis represents MSE.](image-url)
Outline

1. **Bayesian Framework**
   - The Generative Model
   - Bayesian Estimators

2. **MMSE Approximation**
   - Previous Work

3. **Stochastic Resonance**
   - Can Noise Help Denoising?

4. **Our Proposed Method**
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. **Conclusions**
Method Used:

---


6Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Image Denoising

Method Used:

- Dataset containing facial images.

---

6 Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Method Used:

- Dataset containing facial images.
- Trained a Trainlet\(^5\) dictionary on clean facial images.


\(^6\)Dai, Wei, and Olgica Milenkovic, 2009. ”Subspace pursuit for compressive sensing signal reconstruction.”
Image Denoising

Method Used:

- Dataset containing facial images.
- Trained a Trainlet\(^5\) dictionary on clean facial images.
- Pursuit used: Subspace Pursuit (SP)\(^6\).

---


\(^6\) Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Method Used:

- Dataset containing facial images.
- Trained a Trainlet\textsuperscript{5} dictionary on clean facial images.
- Pursuit used: Subspace Pursuit (SP)\textsuperscript{6}.

Experiment:


\textsuperscript{6}Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Method Used:

- Dataset containing facial images.
- Trained a Trainlet\(^5\) dictionary on clean facial images.
- Pursuit used: Subspace Pursuit (SP)\(^6\).

Experiment:

- Added noise to an unseen image.

---


\(^6\)Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Method Used:

- Dataset containing facial images.
- Trained a Trainlet\textsuperscript{5} dictionary on clean facial images.
- Pursuit used: Subspace Pursuit (SP)\textsuperscript{6}.

Experiment:

- Added noise to an unseen image.
- Use the dictionary and SP to denoise. Tune L parameter to obtain optimal denoising performance.

---


\textsuperscript{6}Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Image Denoising

Method Used:

- Dataset containing facial images.
- Trained a Trainlet\textsuperscript{5} dictionary on clean facial images.
- Pursuit used: Subspace Pursuit (SP)\textsuperscript{6}.

Experiment:

- Added noise to an unseen image.
- Use the dictionary and SP to denoise. Tune L parameter to obtain optimal denoising performance.
- Use SR algorithm using the same SP configuration used in the previous step.


\textsuperscript{6}Dai, Wei, and Olgica Milenkovic, 2009. "Subspace pursuit for compressive sensing signal reconstruction."
Image Denoising – Results

Noisy image.

PSNR = 16.1 dB.

Subspace Pursuit.

PSNR = 26.88 dB.

Stochastic Resonance.

PSNR = 28.76 dB.

Clean Image.

∼ 2 dB better.
Image Denoising – Results

Noisy image.
PSNR=16.1 dB.

Clean Image.
∼2dB better.
Image Denoising – Results

Noisy image.  PSNR=16.1 dB.
Subspace Pursuit.  PSNR=26.88 dB.
Clean Image.  ∼2dB better.
Image Denoising – Results

Noisy image. PSNR = 16.1 dB.
Subspace Pursuit. PSNR = 26.88 dB.
Stochastic Resonance. PSNR = 28.76 dB.
Clean Image. 

∼ 2dB better.
Image Denoising – Results

Noisy image. PSNR = 16.1 dB.
Subspace Pursuit. PSNR = 26.88 dB.
Stochastic Resonance. PSNR = 28.76 dB.
Clean Image.

~ 2dB better.
Subspace pursuit vs. stochastic resonance

![Graph showing comparison between Subspace pursuit (SP) and Stochastic resonance (SR) with PSNR on the y-axis and $\sigma_\nu$ on the x-axis]
1. Bayesian Framework
   - The Generative Model
   - Bayesian Estimators

2. MMSE Approximation
   - Previous Work

3. Stochastic Resonance
   - Can Noise Help Denoising?

4. Our Proposed Method
   - The Algorithm
   - Unitary Case Analysis
   - The General Dictionary Case
   - Image Denoising

5. Conclusions
Conclusions

MMSE estimator is desired, but involves an exhaustive computation. MAP is hard as well, but many approximation methods are available. We can use synthetic noise and any MAP estimator approximation to achieve an MMSE estimator approximation. MMSE estimator approximation is attainable, even for large dimensions.
Conclusions

- MMSE estimator is desired, but involves an exhaustive computation.
Conclusions

- MMSE estimator is desired, but involves an exhaustive computation.
- MAP is hard as well, but many approximation methods are available.
Conclusions

- MMSE estimator is desired, but involves an exhaustive computation.
- MAP is hard as well, but many approximation methods are available.
- We can use synthetic noise and any MAP estimator approximation to achieve an MMSE estimator approximation.
Conclusions

- MMSE estimator is desired, but involves an exhaustive computation.
- MAP is hard as well, but many approximation methods are available.
- We can use synthetic noise and any MAP estimator approximation to achieve an MMSE estimator approximation.
- MMSE estimator approximation is attainable, even for large dimensions.
Thank You