Performance Analysis of Iterative Decoding Algorithms with Memory over Memoryless Channels

Authors: Emil Janulewics, Amir H. Banihashemi

Publication: IEEE T. Comm, Dec 2012
Speaker: Jeong-Min Ryu

Short summary:

In this work, they propose a model for iterative decoding algorithms with **memory** which covers successive relaxation (SR) version of belief propagation and differential decoding with binary message passing (DD-BMP) algorithms as special cases. Based on this model, they derive a Bayesian network for iterative algorithms with memory over memoryless channels and use this representation to analyze the performance of the algorithms using density evolution.

I. INTRODUCTION

Iterative decoding algorithm → Decoding algorithm of LDPC codes

Low-density parity-check (LDPC) codes are known to have good performance when decoded with iterative decoding algorithms, also known as message-passing algorithms.

Density Evolution → An analytical tool of LDPC codes

An analytical tool called *density evolution* can be used to find the threshold of a particular code ensemble under a given iterative decoding algorithm. The threshold is an asymptotic measure of performance and is defined as the worst channel parameter (e.g., largest noise variance) for which the probability of error still converges to zero as the number of iterations tends to infinity

Density Evolution → A technique for constructing irregular LDPC codes

Density evolution is also a powerful technique for constructing irregular LDPC codes through the optimization of the degree distributions.

All message-passing algorithms analyzed by density Evolution → Memoryless

To the best of our knowledge however, all the message-passing algorithms analyzed by density evolution in the literature are memoryless, i.e., the output message of a variable node (check node) at iteration l is only a function of the input messages to that node at iteration l (l -1) and also of the initial message of the channel in the case of variable nodes.

Iterative decoding algorithms with memory → Exist

There exist however a number of iterative decoding algorithms, such as successive relaxation (SR) variants of BP and MS and DD-BMP (differential decoding with binary message-passing), that have memory.

The presence of memory in algorithms → Improves the performance but makes the density evolution analysis much more complex.

In this paper, they develop the framework for the density evolution analysis of iterative extrinsic message-passing algorithms with memory which includes DD-BMP and SR algorithms.

They employ the *Bayesian network representation* via a directed acyclic graph (DAG), to capture the dependences among different messages and memory contents in a space with two dimensions: iteration l and the depth of the decoding tree d.

Independent

Incoming messages to a node along different edges

Dependencies

A message passed along a given edge at iteration l

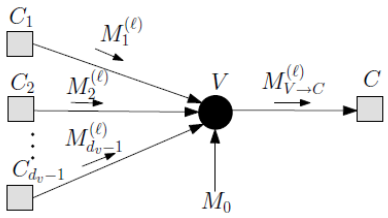
All the messages passed along that edge at previous iteration $l'<l$.

Such dependencies cause the complexity of density evolution to grow at least exponentially with l. They derive the density evolution equations and use techniques to make them tractable.

II. ITERATIVE DECODING ALGORITHMS WITH MEMORY

A. General Model

1) Memoryless decoding algorithm



(a) Memoryless

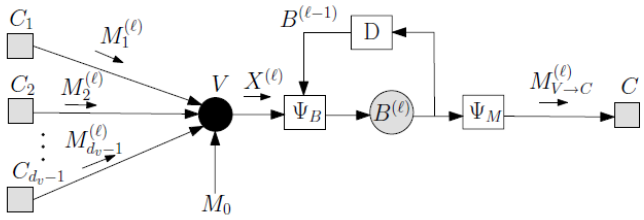
The figure (a) shows a snapshot of the Tanner graph of an LDPC code at iteration l for a memoryless decoding algorithm, where variable and check nodes are represented by circles and squares, respectively.

Under cycle-free assumption and based on the principle of extrinsic message passing,

- Incoming messages $M_1^{(l)},\ldots,M_{d_v-1}^{(l)}$ to node V are independent of each other and of the channel message M_0.
- The outgoing message $M_{V\to C}^{(l)}$ of node V to node C at iteration l is a function of d_v-1 i.i.d. random variables and the channel message.
- The outgoing message of a check node is a function of d_c-1 i.i.d. random variables corresponding to the extrinsic incoming message.

→ The distribution of a function of independent random variables is relatively easy to find since the joint distribution of these variables is the product of their marginal distribution. In this case, one can recursively derive the distribution of messages at iteration l as a deterministic function of the distribution at iteration l-1 with a complexity that is independent of l.

2) Memory decoding algorithm



(b) With Memory

Similar to figure (a), we have a set of i.i.d. extrinsic incoming messages to a variable node *V*.

The outgoing message from V: $X^{(l)}$

Memory units: Ψ_B, Ψ_M, $B^{(l)}$, D.

$B^{(l)}$ is updated by $\Psi_B\left(X^{(l)}, B^{(l-1)}\right)$, where Ψ_B is a deterministic function of $X^{(l)}$ and the content of the memory at iteration *l*-1.

The incoming message $M_{V\to C}^{(l)}$ to node *C* from *V* is obtained by $\Psi_M\left(B^{(l)}\right)$

Note that while the message $X^{(l)}$ is a function of independent random variables, the outgoing messages, $M_{V\to C}^{(l)}$, is a function of dependent random variables $X^{(l)}$ and $B^{(l-1)}$.

Our focus will be on the link from variable nodes to check nodes and on finding the distribution of $B^{(l)}$ and $M_{V\to C}^{(l)}$.

B. SR and DD-BMP Algorithms

1) SR Algorithms: Any standard memoryless iterative algorithm, such as BP or MS, can be turned into an SR algorithm by proper introduction of memory. SR algorithms can be performed in different message domains. In this work, we assume log-likelihood ratio (LLR) domain for messages (SRLLR). Based on the model of Fig. 1(b), the SR version is defined by the following variable node map

$$\begin{aligned} B^{(l)} &= \Psi_B\left(B^{(l-1)}, X^{(l)}\right) = (1-\beta)B^{(l-1)} + \beta X^{(l)}, \\ M_{V\to C}^{(l)} &= \Psi_M\left(B^{(l)}\right) = B^{(l)}, \end{aligned} \tag{1}$$

where $X^{(l)} = \Psi_V\left(M_0, M_1^{(l)}, ..., M_{d_v-1}^{(l)}\right)$. In (1), β is called the *relaxation factor*, and can be optimized for the best performance. The optimal value of β is usually in the interval (0, 1).

2) DD-BMP: Differential decoding with binary message passing (DD-BMP) was introduced as an attractive alternative to purely hard-decision algorithms. This algorithm combines the simplicity of binary message-passing with the good performance of soft-decision algorithms, where the soft information is stored in edge- or node-based memories. In the former case, studied in this paper, the variable node map, following the model of Fig. 1(b), is defined by

$$\begin{aligned} B^{(l)} &= \Psi_B\left(B^{(l-1)}, X^{(l)}\right) = B^{(l-1)} + X^{(l)}, \\ M_{V\to C}^{(l)} &= \Psi_M\left(B^{(l)}\right) = \mathrm{sgn}_r\left(B^{(l)}\right), \end{aligned} \tag{2}$$

where $\mathrm{sgn}_r(x) = 1$ for $x > 0$, and $= -1$ for $x < 0$. For $x = 0$, $\mathrm{sgn}_r(x)$ takes +1 or -1 randomly with equal probability. In (2), $X^{(l)} = \Psi_V\left(M_0, M_1^{(l)}, ..., M_{d_v-1}^{(l)}\right)$, which for the BIAWGN channel reduces to $X^{(l)} = \sum_{i=1}^{d_v-1} M_i^{(l)}$.

Both the variable and the check node operations (particularly the latter) are simpler for DD-BMP compared to BP and MS algorithms.

C. Symmetry of the Decoder and Error Probability

The analysis of iterative decoders is greatly simplified assuming that both the channel and the decoder are symmetric.

In particular, the variable node symmetry condition has some implications on the choices of the mappings Ψ_B and Ψ_M: Ψ_B should be sign inversion invariant, and $\Psi_M(-x) = -\Psi_M(x)$. As it can be seen in (1) and (2), both conditions are satisfied for SRLLR and DD-BMP algorithms.

With both the channel and the decoder being symmetric, we can assume, without loss of generality, that the all-zero codeword is transmitted. In this case, the average fraction of incorrect messages passed at iteration *l* from variable nodes to check nodes is calculated by

$$P_e^{(l)} = P\left(B^{(l)} < 0\right) + \frac{1}{2} P\left(B^{(l)} = 0\right). \quad (3)$$

We refer to $P_e^{(l)}$ in (3) as the probability of bit error at iteration *l*.

III. Bayesian Network Representation of Iterative Decoding Algorithms with Memory

A. Bayesian Networks and Conditional Independence

In this work, they use a Bayesian network to represent the dependencies among different messages and memory contents of an iterative algorithm with memory.

The *conditional independence* between two sets of random variables $\mathcal{X}$ and $\mathcal{Y}$ given a third set $\mathcal{Z}$ is defined by

$$\mathcal{X} \perp \mathcal{Y} \mid \mathcal{Z} \Leftrightarrow \mathbb{P}\left(\mathcal{X}, \mathcal{Y} \mid \mathcal{Z}\right) = \mathbb{P}\left(\mathcal{X} \mid \mathcal{Z}\right) \mathbb{P}\left(\mathcal{Y} \mid \mathcal{Z}\right)$$

where $\mathbb{P}\left(\mathcal{X}\right)$ and $\mathbb{P}\left(\mathcal{X} \mid \mathcal{Z}\right)$ are the marginal distribution of $\mathcal{X}$ and the conditional distribution of $\mathcal{X}$ given $\mathcal{Z}$, respectively.

B. Bayesian Networks of Iterative Decoders with Memory

Based on the principle of extrinsic message-passing, one can see that $X^{(l)}$ is a deterministic function of $B_1^{(l-1)}$ and M_0. Moreover, as it can be seen in Fig. 1(b), $B^{(l)}$ is a function of $B^{(l-1)}$ and $X^{(l)}$. In addition, $B_i^{(l)}$ depends on $B_i^{(l-1)}$ and $B_{i+1}^{(l-1)}$.

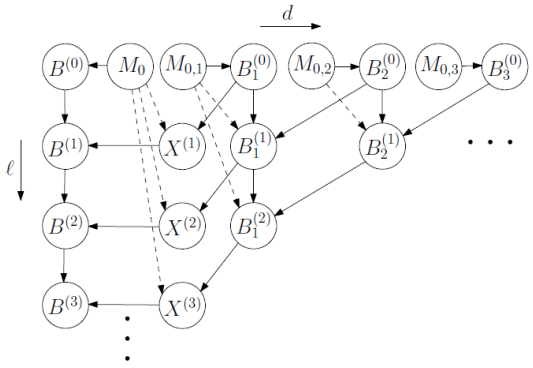


Fig. 3. Bayesian network based on the model of Fig. 1(b).

IV. DENSITY EVOLUTION

Based on (3), to obtain the error probability $P_e^{(l)}$ at each iteration l, we need to compute $\mathbb{P}\left(B^{(k)}\right)$.

A efficient approach is to compute $\mathbb{P}\left(B^{(k)}\right)$ is:

$$P\left(B^{(l)}=b\right)=\sum_{(b',x)\in S_B\times S_X : b=\Psi_B(b',x)} P\left(B^{(l-1)}=b', X^{(l)}=x\right),\ \forall b\in S_B\ .$$

where S_B is sample space of $B^{(l)}$ and $\mathbb{P}\left(B^{(l-1)}, X^{(l)}\right)$ is

$$\begin{aligned}\mathbb{P}\left(B^{(l-1)}, X^{(l)}\right)&=\sum_{x^{(1\to l-1)}}\mathbb{P}\left(B^{(l-1)}\middle| X^{(1\to l-1)}\right)\mathbb{P}\left(X^{(l)}\middle| X^{(1\to l-1)}\right)\mathbb{P}\left(X^{(1\to l-1)}\right)\\&=\sum_{x^{(1\to l-1)}}\mathbb{P}\left(B^{(l-1)}\middle| X^{(1\to l-1)}\right)\mathbb{P}\left(X^{(1\to l)}\right)\end{aligned}$$

where $X^{(1\to k)}=\left\{X^{(1)}, X^{(2)},\cdots, X^{(k)}\right\}$ for $k\geq 1$.

- Calculation of $\mathbb{P}\left(B^{(l)}\middle| X^{(1\to l)}\right)$

$$\begin{aligned}\mathbb{P}\left(B^{(l)}\middle| X^{(1\to l)}\right)&=\sum_{b^{(l-1)}}\mathbb{P}\left(B^{(l)}\middle| X^{(1\to l)}, B^{(l-1)}\right)\mathbb{P}\left(B^{(l-1)}\middle| X^{(1\to l)}\right)\\&=\sum_{b^{(l-1)}}\mathbb{P}\left(B^{(l)}\middle| X^{(l)}, B^{(l-1)}\right)\mathbb{P}\left(B^{(l-1)}\middle| X^{(1\to l-1)}\right),\ \ l\geq 2\end{aligned}$$

- Calculation of $\mathbb{P}\left(X^{(1\to l)}\right)$

The variables $M_i^{(l)}$ are i.i.d.. Since $X^{(l)}=\Psi_v\left(M_1^{(l)}, M_2^{(l)},\ldots, M_{d_v-1}^{(l)}\right)$, $X^{(l)}$ is conditionally independent of all other random variable given $M_{1\to d_v-1}^{(l)}$. We thus have

$$\begin{aligned}\mathbb{P}\left(X^{(1\to l)}\right)&=\sum_{m_{1\to d_v-1}^{(1\to l)}}\mathbb{P}\left(X^{(1\to l)}\middle| M_{1\to d_v-1}^{(1\to l)}\right)\mathbb{P}\left(M_{1\to d_v-1}^{(1\to l)}\right)\\&=\sum_{m_{1\to d_v-1}^{(1\to l)}}\prod_{i=1}^{l}\mathbb{P}\left(X^{(i)}\middle| M_{1\to d_v-1}^{(i)}\right)\prod_{j=1}^{d_v-1}\mathbb{P}\left(M_j^{(1\to l)}\right)\end{aligned}\qquad (4)$$

V. Memory Truncation

A. Main Idea

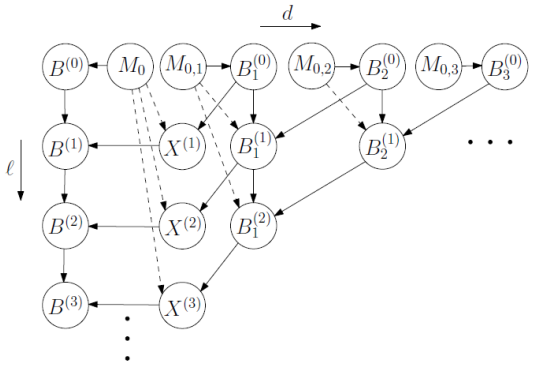


Fig. 3. Bayesian network based on the model of Fig. 1(b).

To explain the approximation, we consider the calculation of the joint distribution $\mathbb{P}\left(B^{(l-1)}\middle|X^{(l)}\right)$. This computation uses the fact that $B^{(l-1)} \perp X^{(l)} \mid X^{(1\to l-1)}$. The problem lies in the fact that the size of the sample space of the conditioning set $X^{(1\to l-1)}$ grows exponentially with *l*. Now consider making the following approximation:

$$B^{(l-1)} \perp X^{(l)} \mid X^{(l-n+1\to l-1)}, n \geq 2 . \quad (5)$$

Regardless of *l*, the conditioning set in (5) will always have a sample space with size $\left|S_X\right|^{n-1}$. Consider the sequence $B^{(0\to l)}$. This sequence, in general, is not a Markov process of some finite order. The memory truncation approximates $B^{(0\to l)}$ by a Markov process of order *n*, $n \in \mathbb{N}$. This is represented by the following:

$$\mathbb{P}\left(B^{(n+k)}\middle|B^{(n+k-1)},\ldots,B^{(0)}\right) \approx \mathbb{P}\left(B^{(n+k)}\middle|B^{(n+k-1)},\ldots,B^{(k)}\right) (6)$$

and corresponds to removing the edges between $X^{(i)}$ and $B^{(i)}$ for $i = 1,...,k$, in the Bayesian network of Fig.3.

We refer to the approximation of (6) as memory truncation of order $n\left(MT^n\right)$.

B. Analysis

We consider a memory truncation of order n, and assume that we have already calculated (approximated) $\mathbb{P}\left(B^{(k)}\right)$, $k \geq n$. For $k = 1, ..., n$, we have the following distributions available:

- $\mathbb{P}\left(B^{(k-1)}, X^{(k)}\right)$
- $\mathbb{P}\left(B^{(k-1)} \middle| X^{(k-n+1 \to k-1)}\right)$
- $\mathbb{P}\left(X^{(k-n+1 \to k)}\right)$

We now derive $\mathbb{P}\left(B^{(k+1)}\right)$. To perform this, we will use the calculation of the joint distribution $\mathbb{P}\left(B^{(k)}, X^{(k+1)}\right)$.

$$\begin{aligned}\mathbb{P}\left(B^{(k)}, X^{(k+1)}\right) &= \sum_{x^{(k-n+2 \to k)}} \mathbb{P}\left(B^{(k)} \mid X^{(k-n+2 \to k)}\right) \mathbb{P}\left(X^{(k+1)} \mid X^{(k-n+2 \to k)}\right) \mathbb{P}\left(X^{(k-n+2 \to k)}\right) \\ &= \sum_{x^{(k-n+2 \to k)}} \mathbb{P}\left(B^{(k)} \mid X^{(k-n+2 \to k)}\right) \mathbb{P}\left(X^{(k-n+2 \to k+1)}\right)\end{aligned} \tag{7}$$

In (7), the distribution $\mathbb{P}\left(B^{(k)} \mid X^{(k-n+2 \to k)}\right)$ is calculated by

$$\mathbb{P}\left(B^{(k)} \mid X^{(k-n+2 \to k)}\right) = \sum_{b^{(k-1)}} \mathbb{P}\left(B^{(k)} \mid X^{(k)}, B^{(k-1)}\right) \mathbb{P}\left(B^{(k-1)} \mid X^{(k-n+2 \to k)}\right),$$

where

$$\begin{aligned}&\mathbb{P}\left(B^{(k-1)} \mid X^{(k-n+2 \to k)}\right) \\ &= \sum_{x^{(k-n+1)}} \mathbb{P}\left(B^{(k-1)} \mid X^{(k-n+1 \to k)}\right) \mathbb{P}\left(X^{(k-n+1)} \mid X^{(k-n+2 \to k)}\right) \\ &= \frac{1}{\mathbb{P}\left(X^{(k-n+2 \to k)}\right)} \sum_{x^{(k-n+1)}} \mathbb{P}\left(B^{(k-1)} \mid X^{(k-n+1 \to k-1)}\right) \mathbb{P}\left(X^{(k-n+1 \to k)}\right) \quad \because \mathbb{P}\left(X^{(k-n+1)} \mid X^{(k-n+2 \to k)}\right) = \frac{\mathbb{P}\left(X^{(k-n+1 \to k)}\right)}{\mathbb{P}\left(X^{(k-n+2 \to k)}\right)}\end{aligned}$$

VI. Simulation Results

In general, the accuracy of $\mathbb{P}\left(B^{(l)}\right)$ increases with increasing the memory truncation order *n*, and so does the complexity. It is however expected that after increasing *n* beyond a certain order n_0, the accuracy improvement would be negligible. The goal is thus to find n_0.

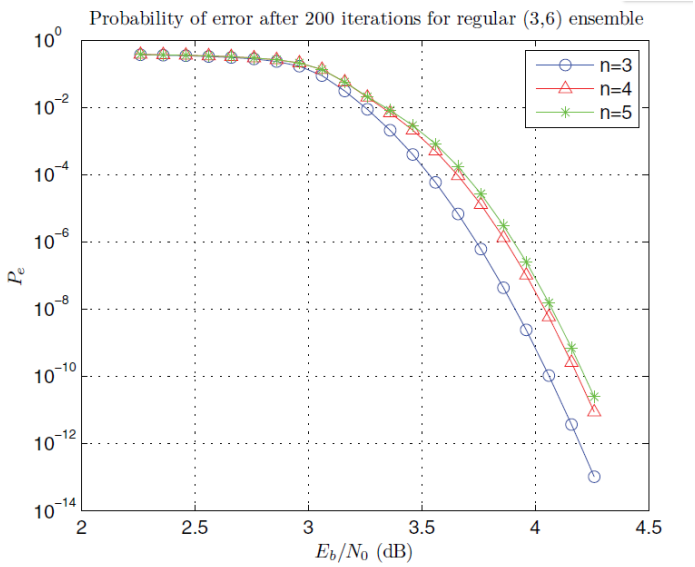


Fig. 4. P_e after 200 iterations vs. E_b/N_0 for the (3,6) ensemble with memory truncation orders 3, 4, and 5.

In Fig. 4, we have shown $P_e^{(l)}$ of the (3, 6) LDPC code ensemble for $l = 200$ vs. E_b / N_0 for different values of memory truncation order *n*. The curves demonstrate a convergence behavior as *n* is increased. In particular, the two curves for *n* = 4 and *n* = 5 are very close. We have also tried a number of other ensembles and observed a similar trend

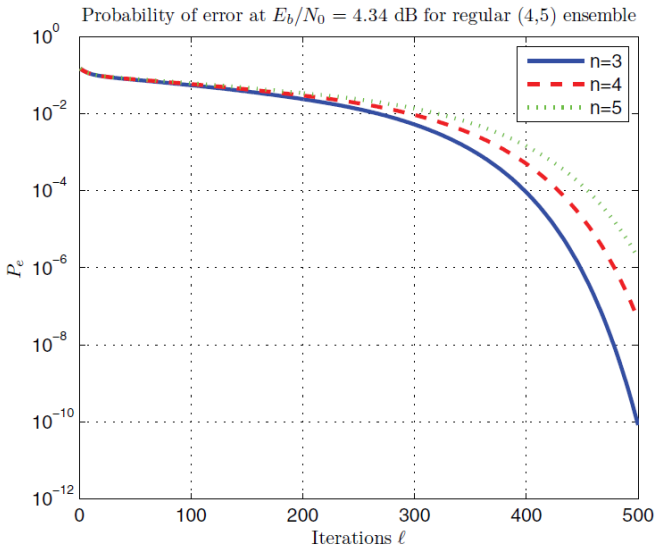


Fig. 5. $P_e^{(\ell)}$ of the (4,5) ensemble vs. ℓ at $E_b/N_0 = 4.34$ dB (above threshold) for memory truncation orders 3, 4, and 5.

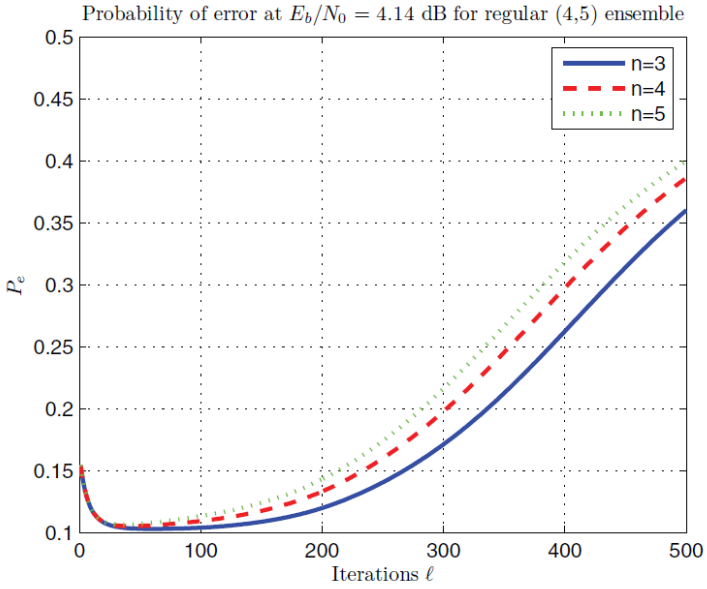


Fig. 6. $P_e^{(\ell)}$ of the (4,5) ensemble vs. ℓ at $E_b/N_0 = 4.14$ dB (below threshold) for memory truncation orders 3, 4, and 5.

For the ensemble of (4, 5) codes, we have plotted $P_e^{(l)}$ vs. l for truncation orders 3, 4 and 5, and for E_b / N_0 values 4.34 dB and 4.14 dB in Figures 5 and 6, respectively. The figures suggest that the ensemble threshold is between the two SNR values.

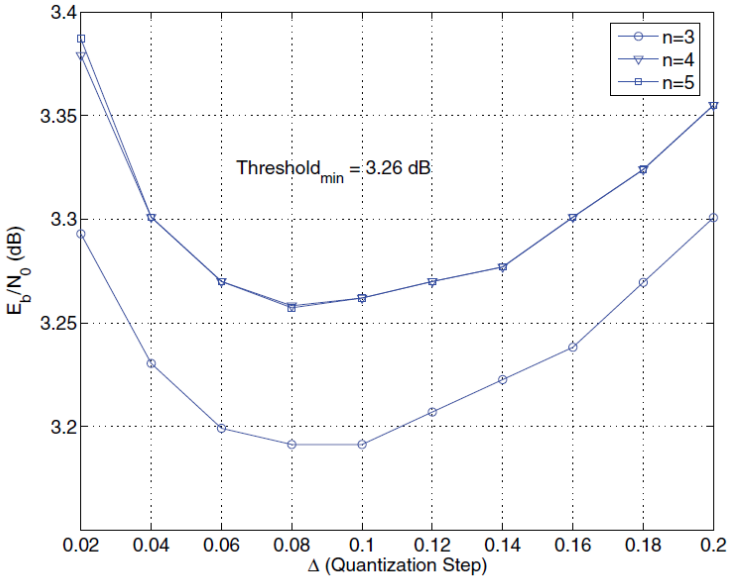


Fig. 7. Threshold values of the $(3, 6)$ ensemble vs. Δ for different memory truncation orders $(q = 8)$.

To clearly see the effect of memory truncation on the calculated thresholds, in Fig. 7, we show the threshold values of the (3, 6) ensemble for different memory truncation orders *n*. The thresholds for each truncation order are plotted versus the quantization step Δ for q = 8.the calculated thresholds for n = 4 and n = 5 are practically identical for different values of Δ. From Fig. 7, the optimal threshold of the (3, 6) ensemble (as a function of Δ) is seen to be about 3.26 dB. Based on the above results, in the following, we use n_0 = 4 to derive the thresholds. In all cases, we use q = 8 and the optimal value of Δ that minimizes the threshold.

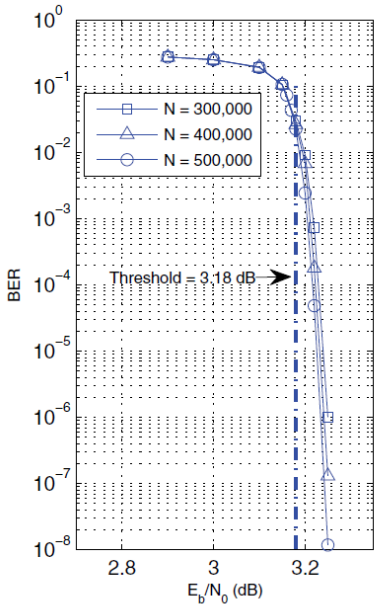


Fig. 8. BER curves of $(5, 10)$ LDPC codes with block lengths 300,000; 400,000; and 500,000 decoded by DD-BMP, and the corresponding threshold $(q = 8)$.

To verify the calculated threshold, for the (5, 10) ensemble, we compare the performance of randomly constructed (5, 10) codes of large block length (N = 300, 000; 400, 000 and 500, 000) with the threshold value of the ensemble (3.18 dB) in Fig. 8.

TABLE I
THRESHOLDS OF BP AND DD-BMP FOR REGULAR ENSEMBLES WITH $d_v = 4$ AND $5 \leq d_c \leq 18$

d_c	$rate$	$\left(\frac{E_b}{N_0}\right)_{BP}$ (dB)	$\left(\frac{E_b}{N_0}\right)_{DD-BMP}$ (dB)	Gap (dB)
5	1/5	2.57	4.24	1.67
6	1/3	1.73	3.18	1.45
7	3/7	1.27	2.88	1.61
8	1/2	1.59	2.80	1.21
9	5/9	1.67	2.82	1.15
10	3/5	1.78	2.87	1.09
11	7/11	1.89	2.93	1.04
12	4/6	1.99	2.99	1.00
13	9/13	2.10	3.06	0.96
14	5/7	2.20	3.13	0.93
15	11/15	2.29	3.20	0.91
16	3/4	2.39	3.27	0.88
17	13/17	2.47	3.33	0.86
18	7/8	2.55	3.40	0.85

TABLE II
THRESHOLDS OF BP AND DD-BMP FOR REGULAR ENSEMBLES WITH $d_v = 6$ AND $7 \leq d_c \leq 18$

d_c	$rate$	$\left(\frac{E_b}{N_0}\right)_{BP}$ (dB)	$\left(\frac{E_b}{N_0}\right)_{DD-BMP}$ (dB)	Gap (dB)
7	1/7	5.14	6.46	1.32
8	1/4	3.52	4.71	1.19
9	1/3	2.90	4.00	1.10
10	2/5	2.61	3.63	1.02
11	5/11	2.47	3.45	0.98
12	1/2	2.41	3.34	0.93
13	7/13	2.39	3.26	0.87
14	6/7	2.40	3.24	0.84
15	3/5	2.43	3.24	0.81
16	5/8	2.46	3.24	0.78
17	11/17	2.50	3.26	0.76
18	2/3	2.54	3.28	0.74

These results show that for a fixed dv, the threshold gap between DD-BMP and BP decreases with increasing the rate. As it can be seen, at higher rates the performance gap is less than 1 dB. In comparison with MS, for codes with larger degrees, DD-BMP outperforms MS

TABLE III
THRESHOLDS OF MS AND DD-BMP FOR RATE-1/2 ENSEMBLES

d_v	d_c	$\left(\frac{E_b}{N_0}\right)_{MS}$ (dB)	$\left(\frac{E_b}{N_0}\right)_{DD-BMP}$ (dB)	Gap (dB)
3	6	1.70	3.26	1.56
4	8	2.49	2.80	0.31
5	10	3.09	3.18	0.09
6	12	3.54	3.34	-0.21
7	14	3.91	3.61	-0.30

These results show that by increasing the degrees, the performance gap between MS and DD-BMP, which is to the advantage of MS for smaller degrees, disappears and then reverses to the advantage of DD-BMP.

These performance results for DD-BMP are impressive considering that both the check node operations and the message-passing for DDBMP are much simpler than those of BP and MS. They also demonstrate the potential of iterative decoding algorithms with memory in achieving better performance/complexity tradeoffs compared to memoryless algorithms.

Faster STORM using compressed sensing

Lei Zhu et al.

Nature method. (2012.04)

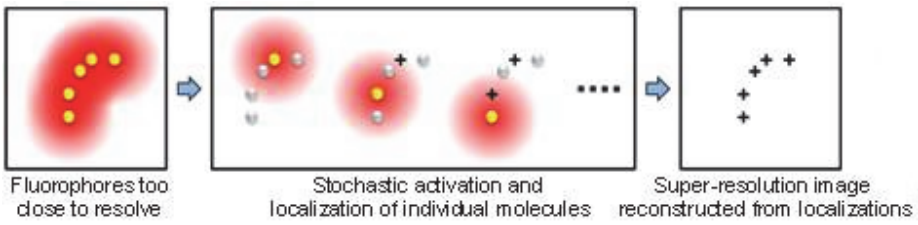
Presenter : Eunseok Jung

GIST, Dept. of Mechatronics , Bioscopy Lab.

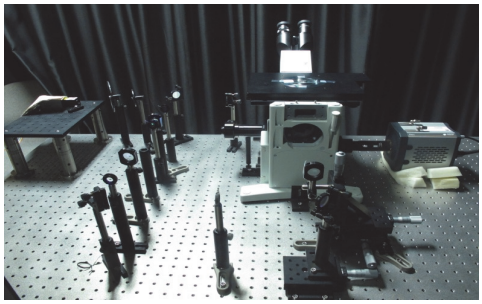
Gwangju Institute of Science and Technology

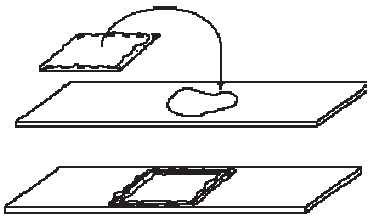
Background

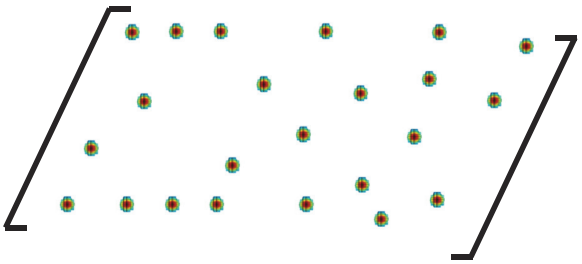
- STORM (Stochastic optical reconstruction microscopy)
  - STORM is a super-resolution optical microscopy technique based on stochastic switching of single-molecule fluorescence signal.
  - STORM utilizes fluorescent probes that can switch between fluorescent and dark states so that in every snapshot, only a small, optically resolvable fraction of the fluorophores is detected.
  - This enables determining their positions with high precision from the center positions of the fluorescent spots.



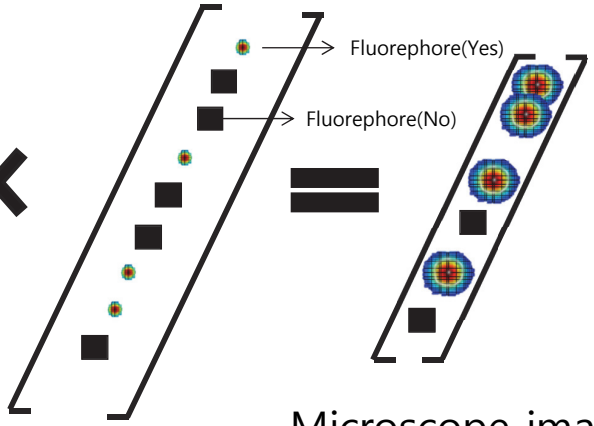
Cited form : http://huanglab.ucsf.edu







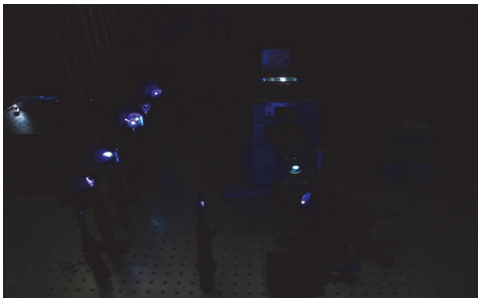
Matrix(**A**)
Optic system - LTI

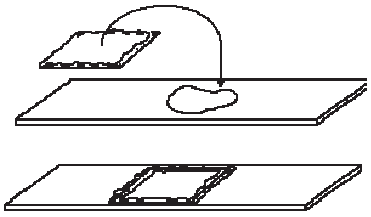


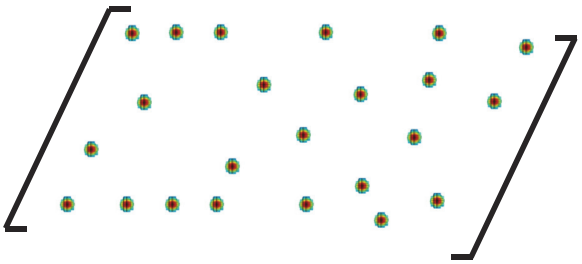
Original
Image(**X**)

Microscope image
-Blurred picture

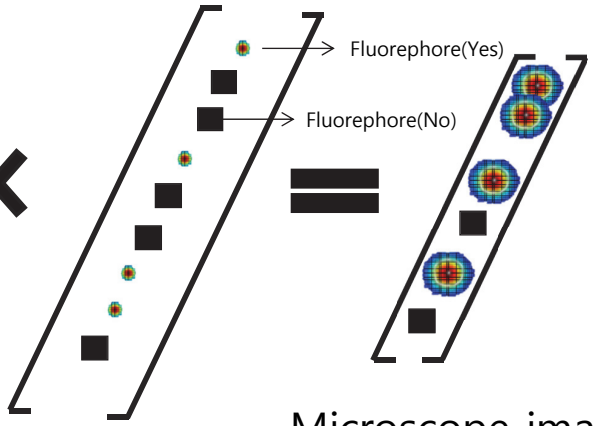
From : http://www.ruf.rice.edu/







Matrix(**A**)
Optic system - LTI



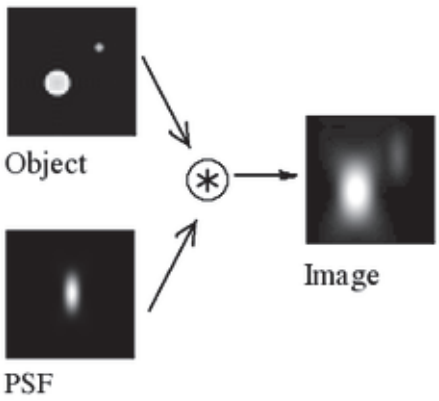
Original
Image(**X**)

Microscope image
-Blurred picture

From : http://www.ruf.rice.edu/

Background

- Point spread function(PSF)
  - The **point spread function** (**PSF**) describes the response of an imaging system to a point source or point object.
  - A more general term for the PSF is a system's impulse response, the PSF being the impulse response of a focused optical system.



From : http://en.wikipedia.org

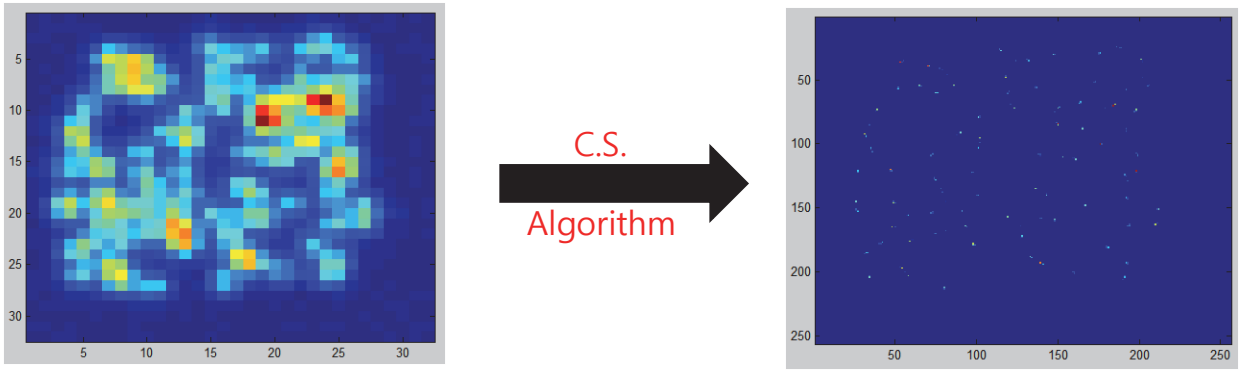
Introduction & Motivation

- Super resolution microscope overcome traditional optical microscope limit.(According to Abbe's theory : $\frac{Light\ wave\ length}{2} \approx 200\text{nm}$)
- The super resolution microscope and fluorescence technique make spatial resolution closer to the molecular scale.(Approximately : 30nm)
- **Now we can see a ten nanometer scale cell structure.**

From : Imaging Intracellular Fluorescent Proteins at Nanometer Resolution, E. Beitzig, science, 2006

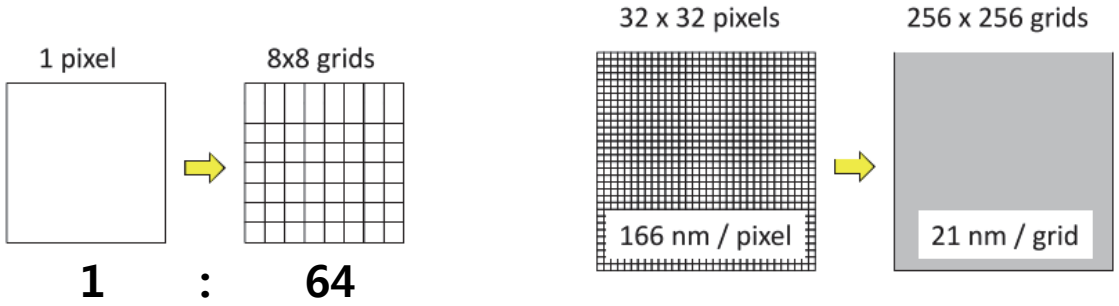
Introduction & Motivation

- Benefit of compressed sensing in fluorescence imaging.



Oversampling

- For increasing spatial resolution, the author used grid method.

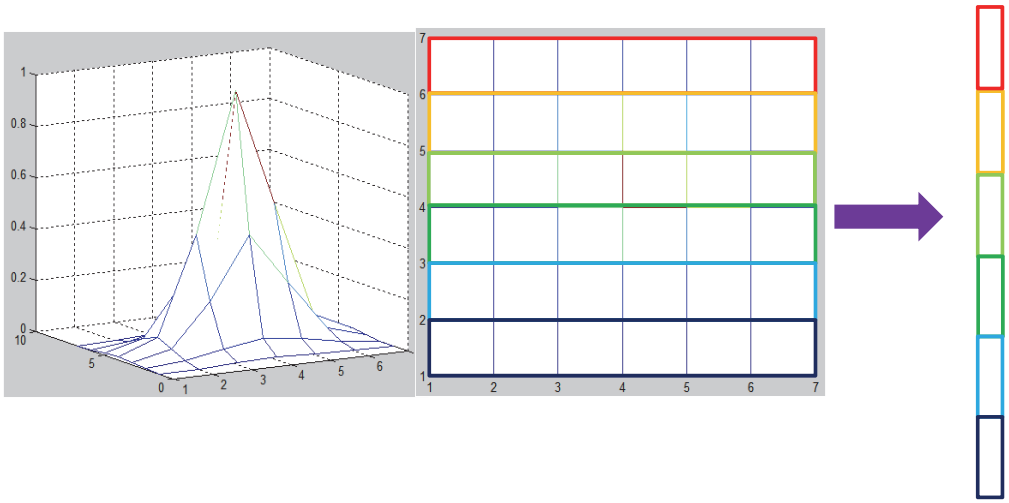


- According to the nyquist theorem, we need 2 times more sampling for reconstruct singal. In here, 21nm*2=42nm.

- Finally, the reconstruction image can get a 42nm spatial resolution.

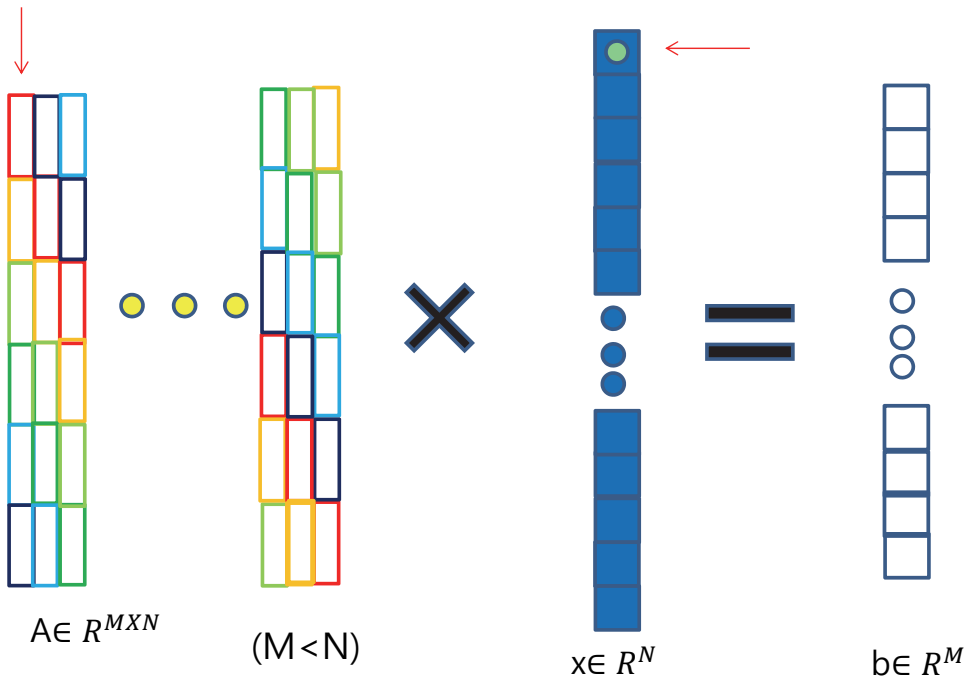
Measurement matrix (1)

- Measure the PSF of optic system.
- Fit the Gaussian function.



- Optic system is LTI system.

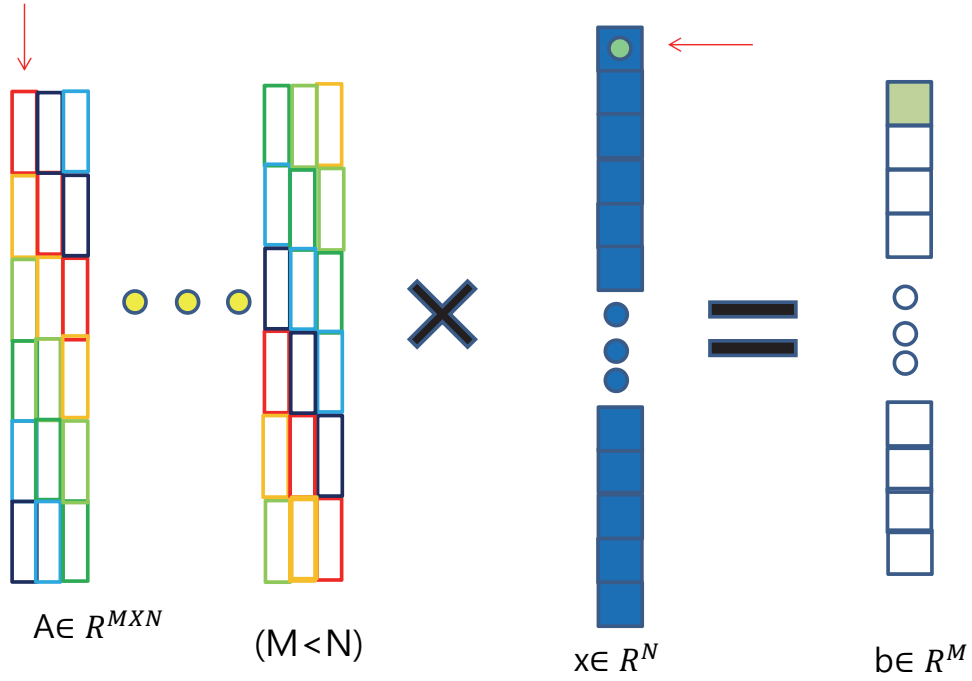
Measurement matrix (2)



A = Measurement matrix **x** = Reconstruct image

Measurement matrix (3)

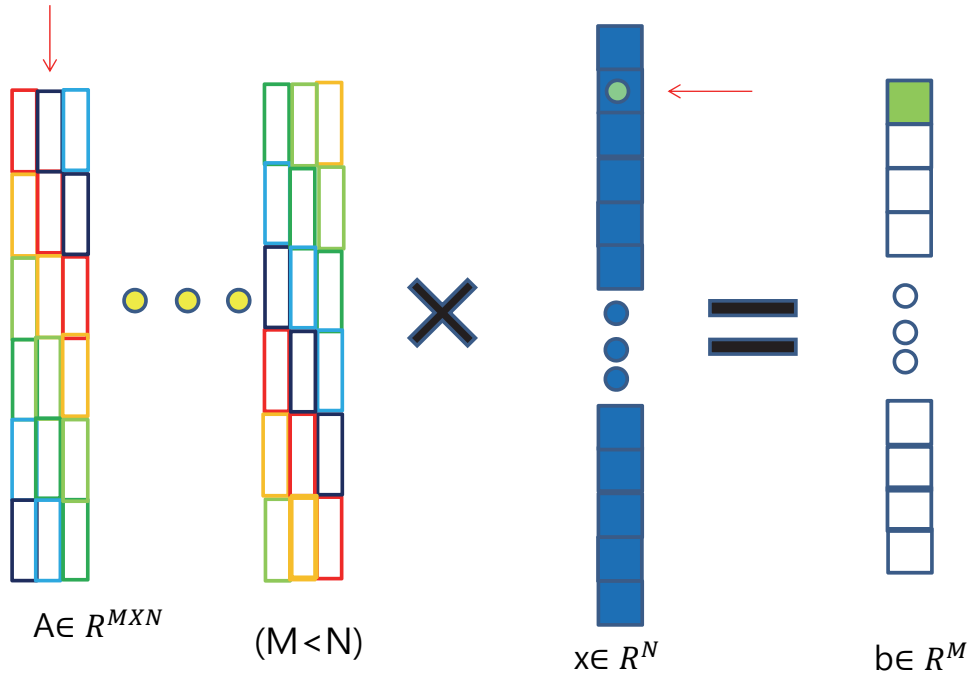
- Measurement matrix A



A = Measurement matrix **x** = Reconstruct image

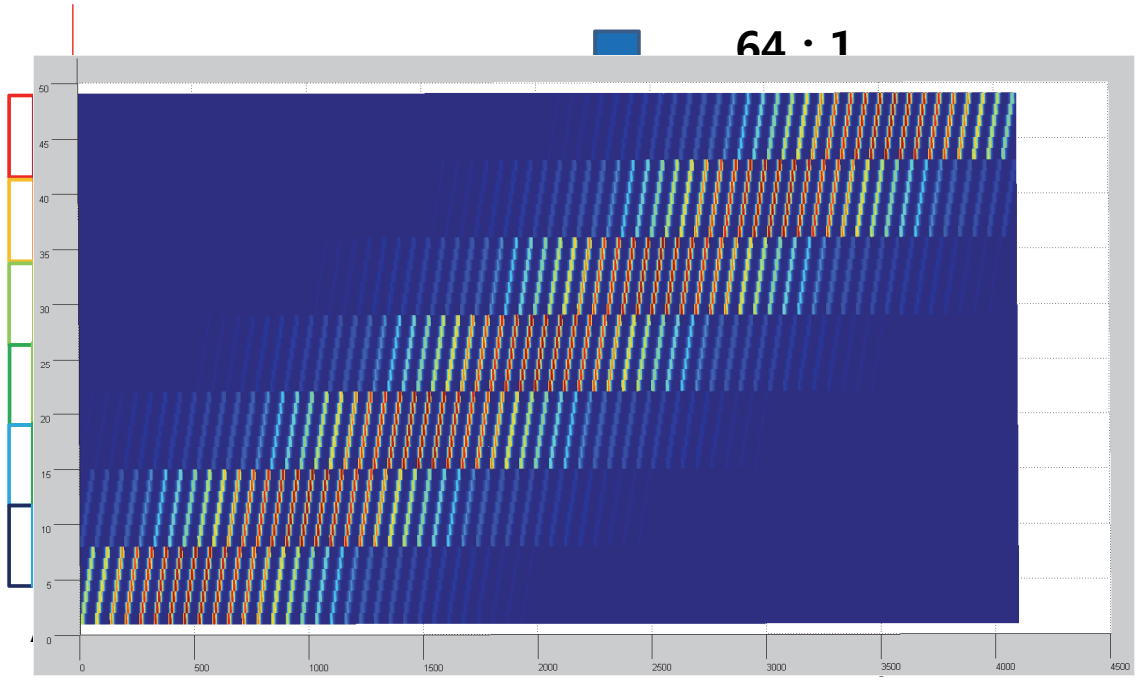
Measurement matrix (4)

- Measurement matrix A

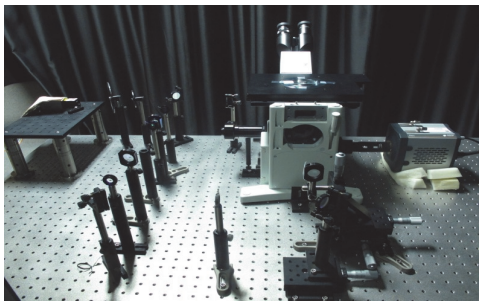


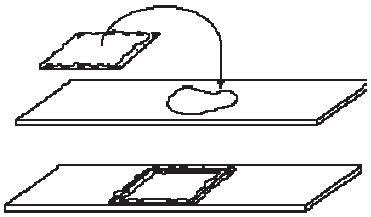
A = Measurement matrix **x** = Reconstruct image

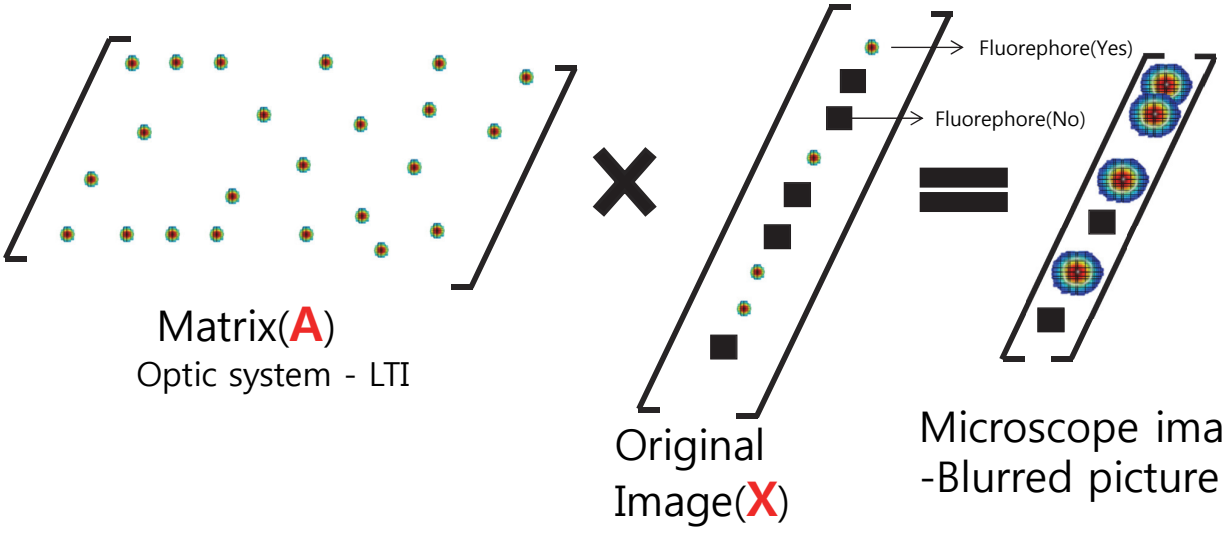
Measurement matrix (5)



A = Measurement matrix **x** = Reconstruct image







Matrix(**A**)
Optic system - LTI

Original
Image(**X**)

Microscope image
-Blurred picture

From : http://www.ruf.rice.edu/

Reconstruction

$$\text{Minimize}: c^{\mathrm{T}}\boldsymbol{x}$$

$$\text{Subject to}: \boldsymbol{x}_i \geq 0 \text{ and } \| \boldsymbol{Ax} - \boldsymbol{b} \|_2 \leq \varepsilon \cdot (\Sigma \boldsymbol{b}_j)^{1/2}$$

- The matrix **A** is determined by the point-spread function (PSF) of the imaging system.
- The i^{th} column of **A** corresponds to the acquired raw image if only one molecule emits fluoroscopic photons at the position index i of **x.**
- The weight vector **c** is to account for the difference of the total contribution to the camera image from one fluorescent molecule at different locations.
- The value of the i^{th} element of **c** equals the summation of the i^{th} column of **A.**
- The minimization term $\mathbf{c}^{\mathrm{T}}\mathbf{x}$ is equivalent to a weighted L1 norm of **x** because **x** is non-negative.

Reconstruction

$$\text{Minimize}: c^{\mathrm{T}}\boldsymbol{x}$$

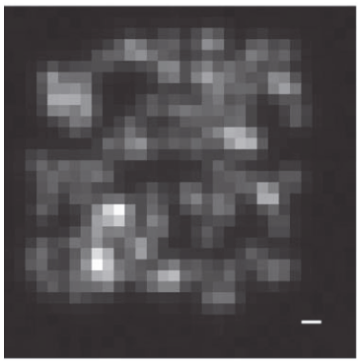
$$\text{Subject to}: \boldsymbol{x}_i \geq 0 \text{ and } \| \boldsymbol{Ax} - \boldsymbol{b} \|_2 \leq \varepsilon \cdot (\Sigma \boldsymbol{b}_j)^{1/2}$$

```
b = img_raw(:);

n = len;

cvx_begin
    variable x(n)
    minimize(c*x)
    subject to
        x >= 0;
        norm( A * x - b, 2 )<=eps;
cvx_end

img_est = x;
```

- CVX converts the above problem to an SOCP (Second-order cone programming), and solves it.

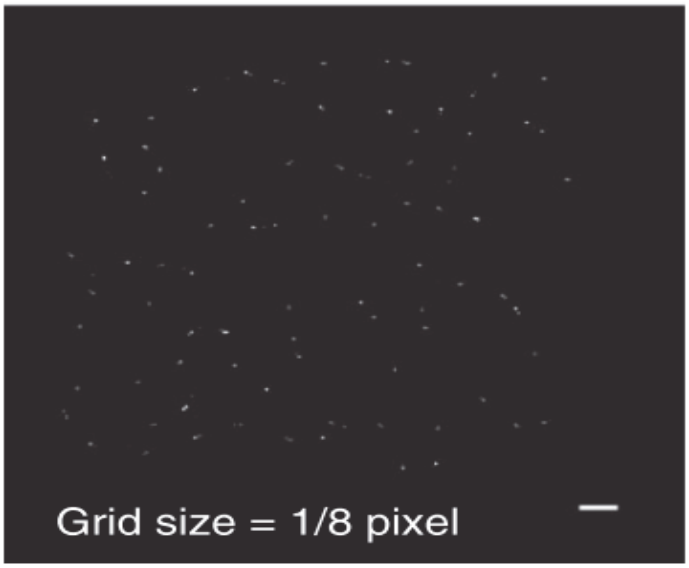
Results

100 molecules in simulation



C.S.
Algorithm

Compressed sensing

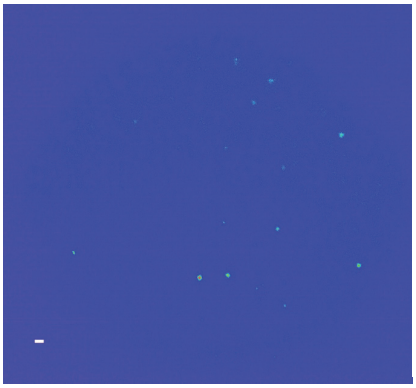


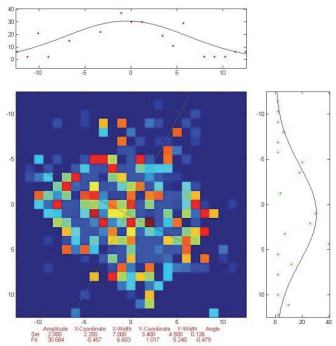
Scale bar : 300nm
FOV : 4um*4um
Pixel : 32*32

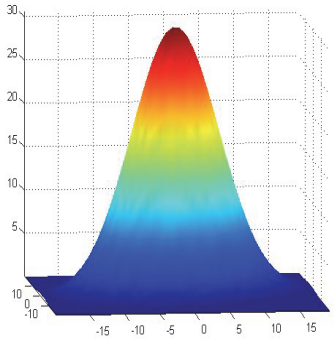
Scale bar : 300nm
FOV : 4um*4um
Pixel : 256*256
The pixel size of 166nm, the 21nm grid size should be able to support a final image resolution of 42nm.

Results

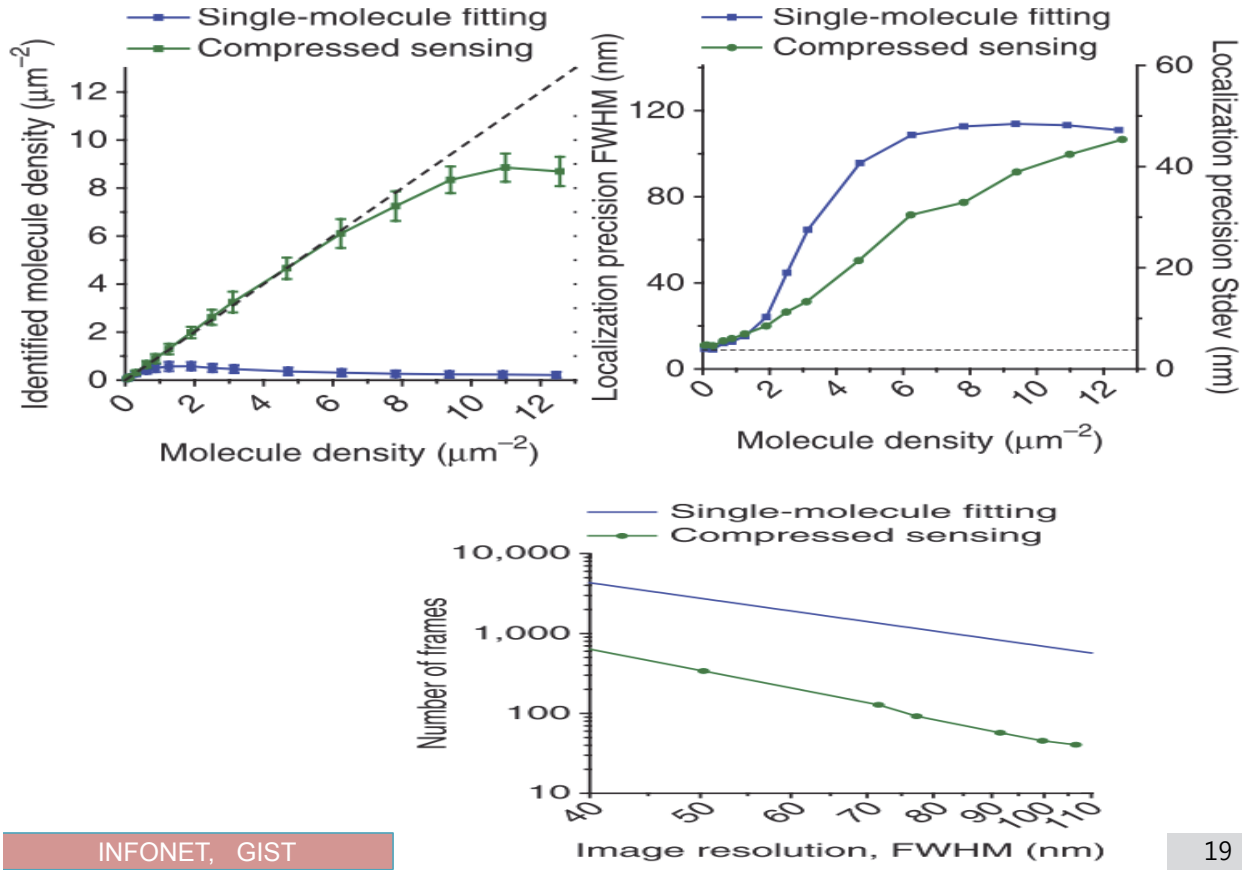
- Conventional fitting



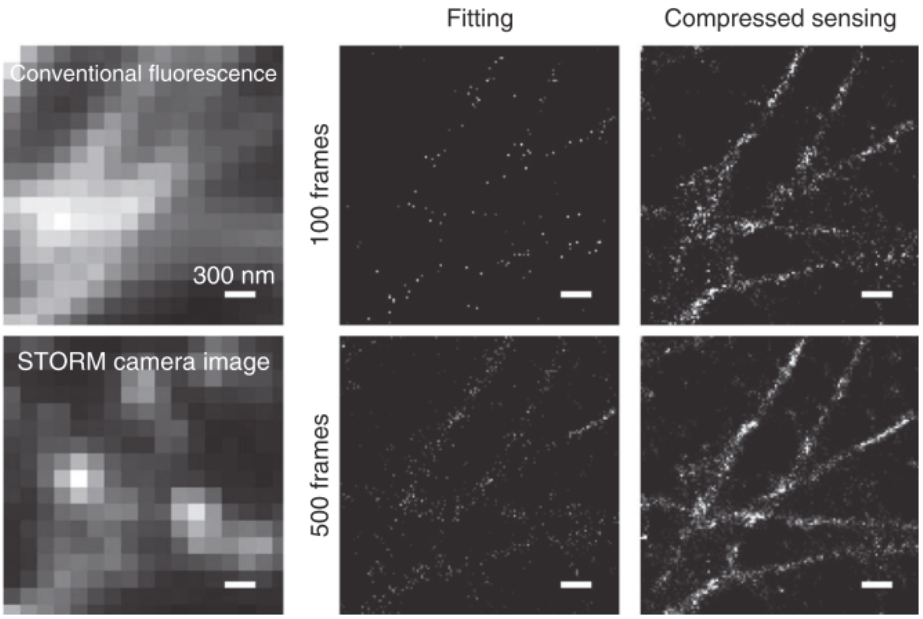




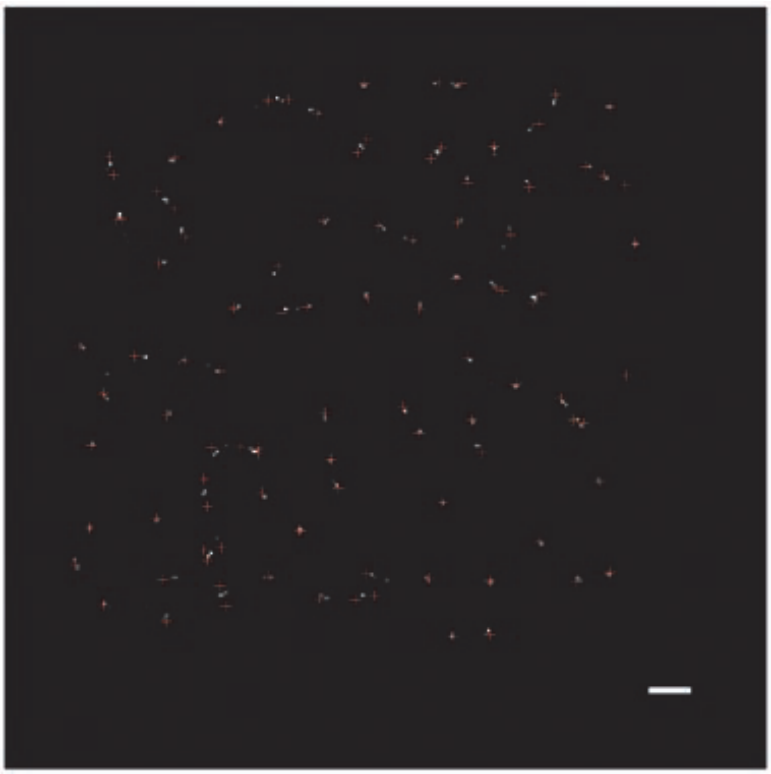
Results



Results



Data Analysis and Discussions



- The red-cross is the reconstructed image using compressed sensing
- The white one is fluorescence position (original image).
- The maximum difference between two positions is 60nm.
- Scale bar is 300nm.

Conclusions

- Conclusion

1) This journal has potential benefit. It was first step of compressed sensing with super resolution microscope.
2) It had not been impossible to taking a living cell image without compressed sensing.
3) Now, it could be done with compressed sensing.
4) The author spent 3sec for taking living cell photo. And they get a same result when they spent over 30sec. They decrease experiment time 10times more.
5) According to this, now we can get a living cell image.

Improvement

- Algorithm

1) If we make measurement matrix A using difference method, we can decrease error and also increase spatial resolution.

2) The other reconstruction method, instead of SOCP, can decrease error.

- Hard ware

1) The author used EM-CCD camera, it is very expensive detect device.

2) But if we can make same result using cheap detect device like a sCMOS or CMOS, the system cost is more cheaper than before.

Journal Club Meeting, Feb. 14, 2013

Thank you

A Node-Based Time Slot Assignment Algorithm for STDMA Wireless Mesh Networks

Authors: W. Chen, and Chin-Tau Lea

Publication: IEEE Trans. Veh. Tech., Jan. 2013
Speaker: Asif Raza

Short summary: In this paper authors present a link capacity model for spatial time-division multiple access (STDMA) mesh networks. It makes use of a simplified transmission model that also considers channel fading. The model then forms the basis of a node-based slot-assignment and scheduling algorithm. This algorithm enables the user to exploit multiuser diversity that results in optimizes network throughput. The presented algorithm shows significant improvement in the throughput when compared with existing slot-assignment methods.

I. Introduction

In STDMA network the transmission time of a channel is divided into slots where multiple slots constitute a frame. These slots are assigned to potential users of the network. The goal of slot assignment scheme is to maximize network throughput. Existing assignment algorithms in STDMA make use of simplified transmission model which do not consider the time-varying fading behavior of a wireless channel. This results in slot wastage when link is in deep fade. The slot is also wasted if scheduled link has no traffic to transmit. This degrades the STDMA network throughput. Therefore a dynamic slot-assignment with that should exploit multiuser diversity is required. However sheer complexity involved in coordinating with all nodes and generating scheduling map in a reasonable time makes this approach impractical. In order to fix these issues the authors present a node-based slot-assignment scheme in which scheduling in each slot is done for nodes not for links. Their contributions include:

- Defining link capacity: a model that includes channel fading. It ensures that whichever link is used by a node will not change the interference profiles on the links selected by other users.
- Node-based time-slot assignment and scheduling algorithms.

II. System Model

Wireless STDMA mesh network with fixed routers.

Transmissions are organized in frames.

Synchronization among nodes provided through GPS.

Set of nodes are identified and assigned to a slot for their transmission.

Each node maintains a separate queue for each outgoing link and performs scheduling without coordination with other nodes.

Multiprotocol Label Switching (MPLS) multipath routing is used for routing however packets are transmitted in sequence.

Adaptive modulation and time varying fading channels are considered. It is also assumed that wireless channels undergo slow fading. Due to fading channel an instant channel gain will be fed back to transmitter. The duration for feedback is no longer than coherence time (the time for which channel conditions remain same)

Adaptive modulation is implemented that each data packet can be fragmented into multiple segments and each segment can be transmitted in with lowest data rate. If high data rate is available then multiple segments can be transmitted per slot duration.

III. Link Capacity Modeling

Each node has multiple links and it can exploit multiuser diversity i.e. different links have different traffic and fading conditions. A channel model is presented that includes shadowing and slow fading.

A. Signal to interference and noise ratio (SINR) Formulation

$h_{r,t}$: Channel response function from transmitter 't' and receiver 'r'

x_t: Signal from 't'

$I_r, n_I, t_i^{'}$: Set of transmitters causing interference to 'r', number of transmitters and i[th] transmitter in I_r respectively. Power control is not considered therefore transmission power of 't' is $pt = E\left(\left|x_t\right|^2\right)$. Let n_0 be thermal noise with power equal to k then received power at 'r' is

$$y_{r,t} = h_{r,t}x_t + \sum_{i=1}^{n_I} h_{r,t_i^{'}} x_{t_i^{'}} + n_0 \qquad (1)$$

SINR at receiver 'r' is expressed as:

$$\gamma_{r,t} = \frac{\left|h_{r,t}x_t\right|^2}{\sum_{i=1}^{n_I}\left|h_{r,t_i^{'}}x_{t_i^{'}}\right|^2 + \kappa} = \frac{\left|h_{r,t}\right|^2 p_t}{\sum_{i=1}^{n_I}\left|h_{r,t_i^{'}}\right|^2 p_{t_i^{'}} + \kappa} = \frac{s_0}{\sum_{i=1}^{n_I} s_i + \kappa} \qquad (2)$$

Here $s_0 = \left|h_{r,t}\right|^2 p_t$ *and* $s_i = \left|h_{r,t_i^{'}}\right|^2 p_{t_i^{'}}$. The Channel response function consists of three parts:

- Path loss
- Shadowing
- Fading

$$h_{r,t} = \sqrt{l_{r,t}^{-\alpha} 10^{\frac{f_{r,t}}{10}} \pi_{r,t}} \qquad (3)$$

Where $l_{r,t}$ is distance between 't' and 'r', $\alpha \mapsto [2-4]$ (constant), $10^{\frac{f_{r,t}}{10}}$ is shadowing effect and it is modeled as a log-normal distributed random variable. $\pi_{r,t}$ is fading effect and it is defined as complex Gaussian RV with mean and variance equal to 0 and 1 respectively. PDF of s_0 and s_i are defined as:

$$p_{s_0}(\alpha_0) = \tfrac{1}{\rho_0} e^{-\frac{\alpha_0}{\rho_0}} \qquad (4a)$$

$$p_{s_i}(\alpha_i) = \tfrac{1}{\rho_i} e^{-\frac{\alpha_i}{\rho_i}} \qquad (4b)$$

Here $\rho_0 = E(s_0) = l_{r,t}^{-\alpha} 10^{\frac{f_{r,t}}{10}} p_t$ and $\rho_i = E(s_i) = l_{r,t_i'}^{-\alpha} 10^{\frac{f_{r,t_i'}}{10}} p_{t_i'}$

B. PDF of SINR

Case 1: no interference is observed by receiver 'r' i.e. (I_r=0, n_I=0) then PDF of $\gamma_{r,t}$, is defined as: let $\delta = \gamma_0 / \kappa$

$$p_{\gamma_{r,t}}(z) = p_{s_0}(z\kappa) = \tfrac{1}{\delta} e^{-\frac{z}{\delta}} \qquad (5)$$

Probability that $\gamma_{r,t}$ is smaller than w is defined as:

$$Pr(\gamma_{r,t} \geq w) = \int_w^\infty p_{\gamma_{r,t}}(z)dz = e^{-\frac{w}{\delta}} \qquad (6)$$

Case 2: unit interference is observed by 'r' i.e. (I_r>0, n_I=1) then PDF of term (s_i+k i.e. denominator of equ.2) is defined as:

$$p_{s_1+\kappa}(v) = \tfrac{1}{\sigma_1} e^{-\frac{v-\kappa}{\sigma_1}} \qquad (7)$$

Finally PDF of $\gamma_{r,t}$ is defined as:

$$p_{\gamma_{r,t}}(z) = \int_\kappa^\infty v p_{s_0}(vz) p_{s_1+\kappa}(v)dv = \int_\kappa^\infty \tfrac{v}{\sigma_0} e^{-\frac{vz}{\sigma_0}} \tfrac{1}{\sigma_1} e^{-\frac{v-\kappa}{\sigma_1}} dv = \tfrac{\kappa + \frac{1}{u}}{\sigma_0 \sigma_1 u} e^{\left(\frac{\kappa}{\sigma_1} - u\kappa\right)} \qquad (8)$$

Probability that $\gamma_{r,t}$ is smaller than w is defined as:

$$Pr(\gamma_{r,t} \geq w) = \int_w^\infty p_{\gamma_{r,t}}(z)dz = \frac{\sigma_0}{\sigma_0 + w\sigma_1} e^{-\frac{w}{\delta}} \qquad (9)$$

Case 3: more than one interferers are present in I_r i.e. ($n_I > 1$) then PDF of $\sum_{i=1}^{n_I} s_i$ can be defined as:

$$p_I(v) = P_{\sum_{i=1}^{n_I} s_i}(v) \otimes p_{s_{n_I}}(v) = \sum_{i=1}^{n_I} \frac{b_i}{\sigma_i} e^{-\frac{v}{\sigma_i}} \qquad (10)$$

Here $b_i = \prod_{j=1, j\neq i}^{n_I} \left(\sigma_i / \sigma_i - \sigma_j \right) and \sum_{i=1}^{n_I} b_i = 1$. The PDF of term ($s_i$+k i.e. denominator of equ.2) is defined as: $p_{I+\kappa}(v) = p_I(v-\kappa)$. Finally the PDF of $\gamma_{r,t}$:

$$p_{\gamma_{r,t}}(z) = \int_{\kappa}^{\infty} v p_{s_0}(vz) p_{I+\kappa}(v) dv = \sum_{i=1}^{n_I} d_i \left(\frac{\kappa}{q_i} + \frac{\kappa}{q_i^2} \right) e^{-\kappa q_i} \qquad (11)$$

Here $q_i = \left(z / \sigma_0 \right) + \left(1 / \sigma_i \right)$ and $d_i = \left(b_i / \sigma_0 \sigma_i \right) e^{-\kappa q_i}$. Probability that $\gamma_{r,t}$ is smaller than w is defined as:

$$Pr\left(\gamma_{r,t} \geq w \right) = \int_{w}^{\infty} p_{\gamma_{r,t}}(z) dz = \sigma_0 e^{-\frac{w}{\delta}} \sum_{i=1}^{n_I} \frac{b_i}{\sigma_0 + \sigma_i w} \qquad (12)$$

Finally Link Capacity can then be determined as:

$$c_{r,t}(I_r) = \sum_{i=1}^{\xi-1} c_i \cdot Pr\left(\gamma_{thr}^{i} \leq \gamma_{r,t} < \gamma_{thr}^{i+1} \right) + c_{\xi} \cdot Pr\left(\gamma_{r,t} \geq \gamma_{thr}^{\xi} \right)$$

Where $c_{r,t}(I_r)$ is average data rate between 't' and 'r', given interference set I_r $c_{r,t}(I_r))$ $c_{r,t}(I_r)$

IV. Proposed Time-Slot Assignment Algorithm

TDMA frame consists of a fixed number of slots is considered. The set of transmitting links that are activated in a given slot is called a **link pattern**, and the set of nodes activated in a given slot is called a **node pattern**.

A. Formulation of Node-Based Time-Slot Algorithm

Notations:

V: set of nodes

E: set of links

NP: Node Pattern

$tx_e, rx_e; e \in E$ transmitter and the receiver of link e, respectively,

$E_{s,p} = \{ e \mid e \in E, p \in s, s \in NP, tx_e = p \}$ set of links that can be used at node p, where p∈s i.e p is activated in node pattern *s*);

μ_s = portion of time that is assigned to node pattern (s) in a frame, where,

$\sum_{s \in NP} \mu_s = 1$

$\left\{ò_{s,p,e} \mid e \in E_{s,p}\right\}$:portion of time that is assigned to each link of node p in node pattern s

F: set of flows in the system; where flow defines all traffic that belongs to (S, D) pair

h_f : traffic demand for flow f, where $f \in F$

S_f : source of flow f

D_f : destination of flow f

$x_{f,e}$: percentage of traffic that flow f passes through link e,

Calculations

Link congestion: it is total amount of traffic routed through the link 'e' over its average capacity (c_e) i.e. $r_e = \left(\sum_{f \in F} \frac{x_{f,e}}{c_e} \right)$ where link capacity (data rate between transmitter 't' and receiver 'r' is defined as: $c_e = \sum_{\{s \mid s \in NP, p \in s, e \in E_{s,p}\}} c_{s,e} ò_{s,p,e}$.

Thus network congestion ratio 'r' is the maximum of all link congestion ratios, i.e. $r = \max_{e \in E} r_e$

Optimal node-based slot assignment scheme is one which minimizes congestion 'r':

$$\min \quad r \qquad (13a)$$

$$s.t \quad \frac{\sum_{f \in F} x_{f,e} h_f}{\sum_{\{e \mid s \in NP, p \in s, e \in E_{s,p}\}} c_{s,e} ò_{s,p,e}} \le r \quad (13b)$$

$$\sum_{\{e \mid e \in E_{s,p}\}} ò_{s,p,e} \le \mu_s \qquad (13c)$$

$$\sum_{\{e \mid p \in s, e \in E_{s,p}, rx_e = q\}} ò_{s,p,e} \le \mu_s \qquad (13d)$$

$$\sum_{s \in NP} \mu_s = 1 \qquad (13e)$$

$$\mu_s \ge 0, ò_{s,p,e} \ge 0, r \ge 0 \qquad (13f)$$

$$\sum_{\{e \mid tx_e = v\}} x_{f,e} - \sum_{\{e \mid rx_e = v\}} x_{f,e} = 0 \quad (13g)$$

$$\sum_{\{e \mid tx_e = S_f\}} x_{f,e} - \sum_{\{e \mid rx_e = S_f\}} x_{f,e} = 1 (13h)$$

$$x_{f,e} \ge 0 \qquad (13i)$$

Problem 13 is the optimization problem, whose purpose is to find the set of $\grave{o}_{s,p,e}$ that will lead to the optimal objective function. Constraint 13c represents that in node pattern s, for any node p∈s, p can transmit to only one node at one time. 13d ensures that a node q can receive from only one node at one time while q ∉ s and p∈s. 13f and 13i ensures non-negativity constraints. Constraint 13b is non-linear therefore $\mu_s r$ and $\grave{o}_{s,p,e} r$ are replaced by $\theta_s \; and \; \rho_{s,p,e}$ respectively. Therefore final formulation is defined as:

$$\min \quad \sum_{s \in NP} \theta_s \qquad (14a)$$

$$s.t \quad \sum_{f \in F} x_{f,e} h_f \le \sum_{\{e | s \in NP, p \in s, e \in E_{s,p}\}} c_{s,e} \grave{o}_{s,p,e} \qquad (14b)$$

$$\sum_{\{e | e \in E_{s,p}\}} \rho_{s,p,e} \le \theta_s \qquad (14c)$$

$$\sum_{\{e | p \in s, e \in E_{s,p}, rx_e = q\}} \rho_{s,p,e} \le \theta_s \qquad (14d)$$

$$\theta_s \ge 0, \rho_{s,p,e} \ge 0 \qquad (14e)$$

$$\sum_{\{e | tx_e = v\}} x_{f,e} - \sum_{\{e | rx_e = v\}} x_{f,e} = 0 \qquad (14f)$$

$$\sum_{\{e | tx_e = S_f\}} x_{f,e} - \sum_{\{e | rx_e = S_f\}} x_{f,e} = 1 \qquad (14g)$$

$$x_{f,e} \ge 0 \qquad (14h)$$

Authors describe that the presented formulation can handle scheduling of node patterns by using Linear Programming approach. However for link based approach, listing all link patterns does not work by using LP formulation. Therefore column generation method is used to tackle the problem.

B. Frame Construction and Throughput Loss due to Frame Quantization

Frame is constructed as: $n_f = \sum_{s \in NP} \left[z \mu_s \right]$ here z is frame length and function [x] rounds 'x' to nearest integer.

The frame quantization will change the portion of time assigned to all patterns (μ_s). Therefore parameters like minimum congestion ratio r_z, the optimal link capacities (c_e) and the routing scheme $x_{f,e}$

will change. These parameters need to be recomputed as follows. let z_s be number of slots assigned to node pattern 's' in a frame.

$$\min \quad \varsigma_z \qquad (15a)$$

$$s.t \quad \sum_{f\in F} y_{f,e} h_f \le \sum_{\{s|s\in NP, p\in s, e\in E_{s,p}\}} c_{s,e} \grave{o}_{s,p,e} \qquad (15b)$$

$$\sum_{\{e|e\in E_{s,p}\}} \grave{o}_{s,p,e} \le \frac{z_s}{\sum_{s\in NP} z_s} \qquad (15c)$$

$$\sum_{\{e|p\in s, e\in E_{s,p}, rx_e=q\}} \grave{o}_{s,p,e} \le \frac{z_s}{\sum_{s\in NP} z_s} \qquad (15d)$$

$$\sum_{\{e|tx_e=v\}} y_{f,e} - \sum_{\{e|rx_e=v\}} y_{f,e} = 0 \qquad (15e)$$

$$\sum_{\{e|tx_e=S_f\}} y_{f,e} - \sum_{\{e|rx_e=S_f\}} y_{f,e} = h_f \varsigma_z \qquad (15f)$$

$$y_{f,e} \ge 0,\ \grave{o}_{s,p,e} \ge 0 \qquad (15g)$$

Here $\varsigma_z = 1/r_z, \mu_s = \left(z_s / \sum_{s\in NP} z_s \right), y_{f,e} = \left(x_{f,e} / r_z \right) = x_{f,e}\varsigma_z$

C. Column Generation Method

Column generation is an algorithm for solving large LP problems. Most of the variables are usually non-basic and assume zero values in the optimal solution, only a subset of variables are needed for solving the problem. Column generation method considers only the variables which have potential to improve the objective function. It splits the problem into master problem and subproblem. Master problem is the original problem with subset of variables being considered. In subproblem it uses duality approach to select new variables to be added to master problem to improve its result.

Master Problem: it is same as defined in problem 14 except that NP is replaced with NP' (subset of NP which is feasible for 14). Solution of master problem shall provide a routing and slot-assignment scheme.

Subproblem: is a new problem created to identify a new node pattern to add to master problem and it is defined as:

$$\min_{s\in\{NP\setminus NP'\}} rp_s \qquad (15)$$

Here rp_s is reduced cost of node pattern 's' in the column generation algorithm and it is optimal value of following problem:

$$\max\left(1-\sum_{\{p|p\in s\}}\omega_{s,p}-\sum_{\{q|p\in s,e\in E_{s,p},rx_e=q\}}\tau_{s,q}\right) \quad (16)$$
$$s.t \qquad \omega_{s,p}+\tau_{s,p}-\phi_e c_{s,e}\geq 0$$

Here $\omega_{s,p}\ and\ \tau_{s,q}$ are variables that are associated with the transmitter p and the receiver q in node patterns s. Well the question is which node pattern should be included into NP' ?

According to duality theory if master problem is optimal then rp_s is always non-negative for any pattern in NP. The node patterns with negative rp_s can improve the result if they are added into NP'. So algorithm will iterate between two phases until no more patterns can be added to NP'.

Algorithm steps are defined as follows:

Step 1: Set node pattern A= ϕ and rp_A =0,

Step 2: Identify $A^c = v$ and compute $rp_{A'_v}$ for node pattern $A'_v\, s.t\, A'_v = \{A, v\}$.

Step 3: select v from A^c with minimum $rp_{A'_v}$ and compute rp_A of A.

Step 4: If $rp_{A'_v} \geq rp_A$, node v will be deleted from A^c and add it A.

Step 5: If $A^c \neq \phi$ stop else go to step 3.

V. SCHEDULING ALGORITHMS

Two scheduling algorithms are proposed in which each node will locally schedule its link transmissions without inter-node coordination and without disturbing interference profiles of other nodes.

A. Scheme 1

Every node 't' in node pattern 's' assigns a transmission probability to every link associated with 't'. The set of transmission probabilities is then defined as:

$$P_{s,t}=\left\{p_{s,t,e}\,|\,e\in E_{s,t}, p_{s,t,e}=\frac{\partial_{s,t,e}}{\mu_s}\right\} \qquad (17)$$

The region [0,1] is then divided into subregions, one for each link in $|E_{s,t}|$, and length of regions is set according to $P_{s,t}$. The algorithm works as follows; Suppose a node pattern 's' is activated in slot x. Each node $t \in s$ will generate a RV w, uniformly distributed within [0, 1]. The node will then schedule

link into which subregion w falls. If selected link (e) is not usable (either due to fading or no traffic) the scheduler will check link next to 'e' one by one until a usable link is found.

B. Scheme 2

Scheme 1 does not consider link quality while scheduling the links. Therefore authors presented another scheduling mechanism.

Selection criteria = (queue length * link capacity)

Each node maintains two queues for each of its link:

1) a real data queue to store packets and

2) A shadow queue for scheduling.

These queues of link 'e' whose transmitter can be activated in slot 'x', are defined as:

$$q_e(x) = q_e(x-1) + a_e(x) - d_e(x)$$
$$q'_e(x) = q'_e(x-1) + a'_e(x) - d'_e(x)$$

Here $q_e(x)$ and $q'_e(x)$ are lengths of the real queue and shadow queue respectively. $a_e(x)$, $a'_e(x)$, $d_e(x)$ and $d'_e(x)$ are the number of arrivals and departures for the two queues in 'x', respectively. In shadow queue the term $a'_e(x)$ is defined as: $a'_e(x) = \left(1 + \gamma_x\right)\sum_{t=0}^{x} a_e(t)$ i.e. it is used to smooth the incoming traffic from source or previous hop.

Packets departing from link 'e' are defined as: $d_e(x) = \min\left\{\tilde{c}_e(x), q_e(x)\right\}$. Here $\tilde{c}_e(x)$ is instant capacity of link 'e' in slot 'x'. Thus scheduling, in slot 'x', the scheduler in node $t \in s$ will select the link from all its associated links with a maximum value of $q'_e(x)\tilde{c}_e(x)$. In doing so, it tries to strike the optimal balance between link quality and traffic backlog.

VI. Simulation and Results

A. Simulation Environment and Settings

Linear optimization toolbox of MATLAB is used for proposed routing and slot-assignment algorithm. C++ program is then used to inspect maximum achievable throughput for different scheduling schemes.

The physical-layer parameters are summarized as follows:

- Transmission power: 20 dBm.
- Thermal noise:−90 dBm.
- Path loss(α):3.5.
- Variance of shadow fading: 4 dBm.

• Minimal distance of two nodes: 15 m.

• Slot duration: 0.22 ms.

• Frame size: 100 slots. So frame length = 22 ms.

The mapping between the following data rates and SINR threshold is summarized as follows.

• 54 Mb/s: 24.56 dBm.

• 48 Mb/s: 24.05 dBm.

• 36 Mb/s: 18.80 dBm.

• 24 Mb/s: 17.04 dBm.

• 18 Mb/s: 10.79 dBm.

• 12 Mb/s: 9.03 dBm.

• 9 Mb/s: 7.78 dBm.

• 6 Mb/s: 6.02 dBm.

Network Topology: two networks 15-node and 30-node with two gateway nodes and three gateway nodes are considered, respectively.

The traffic load of each flow is assumed to be the same i.e.,h_f =1Mb/s,

Throughput loss due to Frame Quantization:

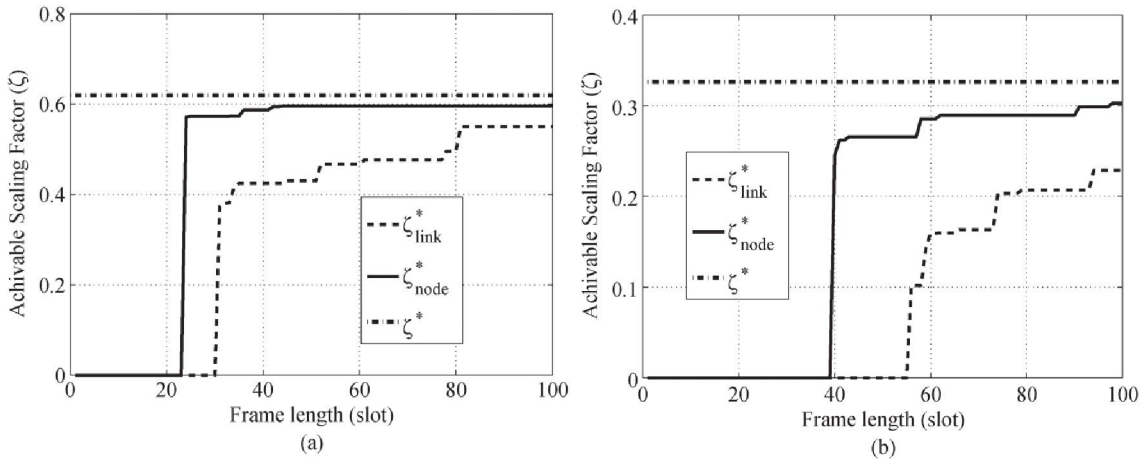


Fig. 1. Achievable throughput after frame generation for (a) 15- and (b) 30-node networks.

The solid (ζ^*_{node}), dashed (ζ^*_{link}) and dashed–dotted (ζ^*) lines indicate the achievable throughput in node-based, link based and before frame construction (I.e. upper bound on throughput) respectively. The flat area represents the range where the performance does not improve. Note that (ζ^*_{node}) and (ζ^*_{link}) are function of 'z' and are not always monotonically increasing due to the quantization involved in the process, and small oscillation occurs within a short range of z. This is why, in Fig. 1(a) and (b), the curves move up in steps.

TABLE I
PERFORMANCE COMPARISONS FOR THE LINK- AND NODE-BASED SCHEMES

N	ζ^*	ζ^*_{node} (ζ^*_{link})	n_{use}	$\zeta^*_{exp,1}$	$\zeta^*_{exp,2}$	$\zeta^*_{det,1}$	$\zeta^*_{det,2}$
15 (Node)	0.62	0.60	17	0.62	0.77	0.62	0.76
15 (Link)	0.62	0.55	26	NA	NA	NA	NA
30 (Node)	0.33	0.30	30	0.31	0.39	0.31	0.39
30 (Link)	0.33	0.23	54	NA	NA	NA	NA

From the table it is clear that ζ^*_{node} approaches ζ^* much faster than ζ^*_{link}. Moreover difference in throughput between ζ^*_{node} and ζ^*_{link} is also significant as shown in table 1.

The optimal scaling factors of the ζ^*_{node} for schemes 1 and 2 under the Poisson and a deterministic arrival process are denoted as shown by $\zeta^*_{\text{exp},1}$, $\zeta^*_{\text{exp},2}$, $\zeta^*_{\text{det},1}$, $\zeta^*_{\text{det},2}$ respectively.

{the ζ^*_{node} is derived from problem 14 and it does not include multi-user diversity gain. Therefore, it can be viewed as a lower bound of the two proposed scheduling schemes 1 and 2 As shown in the Table I.

It is also clear from the table that , both (posisson and deterministic arrival rates) $\zeta^*_{\text{exp},1}$, $\zeta^*_{\text{det},1}$ are only slightly larger than ζ^*_{node} for the 15- and 30-node networks. The difference is only about 3%. This is because scheme 1 tries to follow $\grave{o}_{s,p,e}$ i.e. portion of time that is assigned to each link of node p in node pattern s and does not select a link with the best quality. However, the situation is different in scheme 2, because link quality is part of the selection criteria. With scheme 2, $\zeta^*_{\text{exp},1}$, $\zeta^*_{\text{det},1}$ are about 26% larger than ζ^*_{node} for the 15-node network and 30% larger for the 30-node network.

A New TwIST: Two-Step Iterative Shrinkage/Thresholding Algorithms for Image Restoration

J. M. Bioucas-Dias, M. A. T. Figueiredo

IEEE transactions on image processing 2007.

Summary:

- ✓ It happens that the convergence rate of IST algorithms depends heavily on the linear observation operator, becoming very sow when this operator is ill-conditioned or ill-posed.
- ✓ In this paper, the authors introduce two-step IST (TwIST) algorithms, exhibiting much faster convergence rate than IST for ill-conditioned problems. They showed that TWIST converges to a minimizer of the objective function, for a given range of values of its parameters.

I. Introduction

Many approaches to LIPs define a solution $\hat{\mathbf{x}}$ as a minimizer of a convex objective function *f*:

$$f(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2 + \lambda\Phi(\mathbf{x}) \quad (1)$$

In a regularization framework, minimizing *f* is seen as a way of overcoming the ill-conditioned, or singular, nature of **K** which precludes inverting it. In this context, Φ is called the regularizer and λ the regularization parameter.

The current state-of-the-art regularizers for image restoration are nondifferentiable.

Examples of such choices are total-variation (TV) regularization and wavelet-based regularization.

The nondifferentiable nature of *f*, together with the huge dimension of its argument, place its minimization beyond the reach of standard off-the-shelf optimization methods.

Contribution

This paper is strictly concerned with algorithms for minimizing (1).

This paper introduces a new class of iterative schemes, bringing together the best of IRS and IST. For ill-conditioned (but invertible) linear observation operators, they prove (linear) convergence of TwIST to minima of the objective function f, for a certain range of the algorithm parameters, and derive bounds for the convergence factor.

II. Regularizers and Denoising

Denoising with convex regularizers

Denoising problems are LIPs in which **K** is the identity, **Kx**=**x**.

$$f_{den} = \frac{1}{2} d_y^2 + \lambda\Phi \text{ . where } \quad d_y^2 = \left\|\mathbf{y} - \mathbf{x}\right\|^2 \tag{2}.$$

With several standard assumptions about the regularizer $\Phi : \chi \to R$ (convex, lower semi-continuous (lsc), proper), its minimizer is unique; refer to Theorem 5 and Theorem 7 in appendix I. This allows defining the denoising function.

$$\mathbf{\Psi}_\lambda(\mathbf{y}) = \arg\min_{\mathbf{x}} \left\{ \frac{1}{2} d_y^2(\mathbf{x}) + \lambda\Phi(\mathbf{x}) \right\} \tag{3}.$$

Denoising with l-Homogeneous Regularizers

Let $\Upsilon(\chi)$ denote the set of functions $\Phi : \chi \to R$ that are convex, lsc, proper, and phd-1.

An important recent result states that denoising with regularizers from $\Upsilon(\chi)$ corresponds to the residual of the projection onto a convex set, as formalized in the following theorem.

Theorem 1: If $\Phi \in \Upsilon(\chi)$, then the denoising function $\mathbf{\Psi}_\lambda$ defined in (3) is given by

$$\mathbf{\Psi}_\lambda(\mathbf{y}) = \mathbf{y} - P_{\lambda C}(\mathbf{y}) \tag{4}$$

Where $C \subset \chi$ is a closed convex set depending on the regularizer Φ, and $P_A : \chi \to \chi$ denotes the orthogonal projection operator onto the convex set $A \subset \chi$.

Total variation

$\Phi_{iTV}, \Phi_{niTV} \in \Upsilon(R^m)$

Weighted lp norm

$$\Phi_{l_w^p}(\mathbf{x}) = \left\|\mathbf{x}\right\|_{p,w} = \left(\sum_i w_i \left| x_i \right|^p \right)^{1/p}$$

Being a norm, $\Phi_{l_w^p}$ clearly belongs to Υ.

The denoising function $\mathbf{\Psi}_\lambda$ under a $\Phi_{l_w^p}$ regularizer cannot be optained in a closed form, except in some particular cases, the most notable of which is p=1; in this case, $\mathbf{\Psi}_\lambda$ is the well known soft-thresholding function, that is $\mathbf{\Psi}_\lambda(\mathbf{z}) = \hat{\mathbf{x}}$ with

$$\hat{x}_i = sign(z_i) \max\left\{0, \left|z_i\right| - \lambda w_i\right\} \tag{8}.$$

Orthogonal representations

$$f(\mathbf{x}) = \frac{1}{2} d_{\mathbf{y}}^2(\mathbf{HWx}) + \lambda \Phi_{l_w^p}(\mathbf{x}) \quad (9).$$

III. Existence and uniqueness of solutions

Proposition 1: Let $f : X \to \bar{R}$ be defined as in (1), where operator **K** is linear and bounded, and Φ is a proper, lsc, convex function. Let G denote the set of minimizers of *f*. Then:

i) If Φ is coercive, then G is nonempty;

ii) If Φ is strictly convex or **K** is injective, then G contains at most one element;

iii) If **K** is bounded below, then G contains exactly one element.

Application of Proposition 1 to the several regularization function.

Weighted l-p norm and its p-th power

If all the weights are strictly positive, both are coercive; this ensures existence of minimizers of *f*. If K is injective, the minimizer is unique; otherwise, the minimizer is unique with $\Phi_{l_w^p}^p$, with *p*>1 (which is strictly convex).

Finite-dimensional cases.

Injectivity of K is sufficient to guarantee existence and uniqueness of the solution (under any convex regularizer, strictly or not, coercive or not). It is because any finite-dimensional injective operator is bounded below.

IV. Previous algorithms

Iterative shrinkage/thresholding (IST)

$$\mathbf{x}_{t+1} = (1-\beta)\mathbf{x}_t + \beta \boldsymbol{\Psi}_\lambda(\mathbf{x}_t + \mathbf{K}^T(\mathbf{y} - \mathbf{K}\mathbf{x}_t)) \quad (13)$$

Each iteration of the IST algorithm only involves sums, matrix-vector products by **K** and $\mathbf{K}^T$, and the application of the denoising operation $\boldsymbol{\Psi}_\lambda$.

Theorem 2: Let f be given by (1), where Φ is convex (and lsc) and $\|\mathbf{K}\|_2^2 < 2$. Let *G* be nonempty. Fix some $\mathbf{x}_1$ and let the sequence be produced by (13), with $\beta \in [0,1]$. Then the sequence converges to a point $\mathbf{x} \in G$.

Iterative Re-Weighted Shrinkage (IRS)

The IRS algorithm was specifically designed for wavelet-based problems of the form (9), where W contains an orthogonal or redundant wavelet basis and the regularizer is not necessarily a weighted lp norm.

$$\mathbf{x}_{t+1} = \text{solution}\left\{\mathbf{A}_t\mathbf{x} = \mathbf{b}\right\} = \text{solution}\left\{(\lambda\mathbf{D}_t + \mathbf{K}^T\mathbf{K})\mathbf{x} = \mathbf{K}^T\mathbf{y}\right\}$$

$\mathbf{D}_t$ is a diagonal matrix (of non-negative elements) that depends on $\mathbf{x}_t$ and Φ. Observe that matrix $\mathbf{D}_t$ shrinks the components of $\mathbf{x}_{t+1}$, thus the term iterative reweighted shrinkage.

The huge size of $\mathbf{A}_t = (\lambda\mathbf{D}_t + \mathbf{K}^T\mathbf{K})$ forces the use of iterative methods to implement. This is done with a two-step stationary iterative method, which we will next briefly review.

Two-Step Methods for Linear Systems

Considering the linear system $\mathbf{Ax} = \mathbf{b}$, with **A** positive definite; define a so-called splitting of **A** as **A=C-R**, such that **C** is positive definite and easy to invert (e.g., a diagonal matrix). A stationary two-step iterative method (TwSIM) for solving **Ax=b** is defined as

Two-step iterative method (TwSIM)

$$\mathbf{x}_1 = \mathbf{x}_0 + \beta_0\mathbf{C}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_0)$$

$$\mathbf{x}_{t+1} = (1-\alpha)\mathbf{x}_{t-1} + \alpha\mathbf{x}_t + \beta\mathbf{C}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_t) \qquad (15).$$

Theorem 3: Let $\{x_t, t \in \mathbb{N}\}$ be the sequence produced by (15), with arbitrary $\mathbf{x}_0$. Let λ_1 and λ_m denote the smallest and largest eigenvalues of matrix $\mathbf{C}^{-1}\mathbf{A}$, and $k = \lambda_1 / \lambda_m$. Then, $\{x_t, t \in \mathbb{N}\}$ converges to the solution of Ax=b if and only if $0 < \alpha < 2$ and $0 < \beta < 2\alpha / \lambda_m$. The optimal asymptotic convergence factor is $\rho = (1-\sqrt{k}) / (1+\sqrt{k})$.

Comparing IST with IRS

For ill conditioned systems, IRS is much faster than IST. On the other hand, when noise is the main factor, and the observation operator is not too ill-conditioned, IST outperforms IRS because it uses a closed-form denoising step in each iteration.

V. Two-Step IST (TwIST)

The TwIST method aims at keeping the good denoising performance of the IST scheme, while still being able to handle ill-posed problems as efficiently as the IRS algorithm.

Taking $\mathbf{C} = \mathbf{I} + \lambda\mathbf{D}_t$ and $\mathbf{R} = \mathbf{I} - \mathbf{K}^T\mathbf{K}$ in the splitting **A=C-R** of matrix $\mathbf{A} = \lambda\mathbf{D}_t + \mathbf{K}^T\mathbf{K}$, the two-step iteration (15) for the linear system $\mathbf{Ax} = \mathbf{K}^T\mathbf{y}$ becomes

$$\mathbf{x}_{t+1} = (1-\alpha)\mathbf{x}_{t-1} + (\alpha-\beta)\mathbf{x}_t + \beta\mathbf{C}^{-1}(\mathbf{x}_t + \mathbf{K}^T(\mathbf{y} - \mathbf{K}\mathbf{x}_t)) \quad (16)$$

Equation (13) can be obtained from (16) by setting α =1 and replacing the multiplication by matrix $\mathbf{C}^{-1}$ by the denoising operator $\mathbf{\Psi}_\lambda$. This similarity suggests a two-step version of IST (TwIST) as

$$\mathbf{x}_1 = \mathbf{\Gamma}_\lambda(\mathbf{x}_0) \quad (17)$$

$$\mathbf{x}_{t+1} = (1-\alpha)\mathbf{x}_{t-1} + (\alpha-\beta)\mathbf{x}_t + \beta\mathbf{\Gamma}_\lambda(\mathbf{x}_t) \quad (18)$$

for $t \geq 1$, where $\mathbf{\Gamma}_\lambda : R^m \to R^m$ is defined as

$$\mathbf{\Gamma}_\lambda(\mathbf{x}) = \mathbf{\Psi}_\lambda(\mathbf{x} + \mathbf{K}^T(\mathbf{y} - \mathbf{K}\mathbf{x})) \quad (19).$$

A key observation is that TwIST and IST all have the same fixed points.

VI. Experimental Results

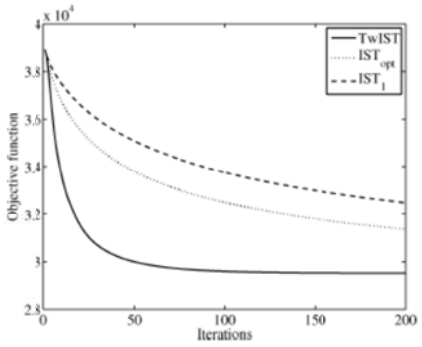


Fig. 1. TV-based deconvolution in a severely ill-conditioned problem (experiment 1). Evolution of the objective function $f(x_t)$ produced by TwIST, $\mathrm{IST}_{\mathrm{opt}}$, and IST_1.

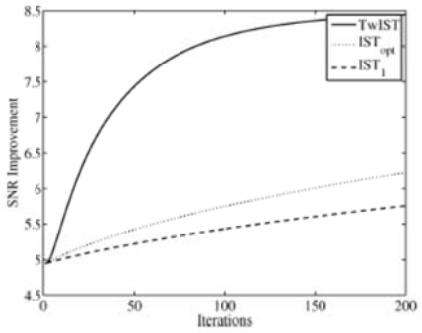


Fig. 2. TV-based deconvolution in a severely ill-conditioned problem (experiment 1). Evolution of the SNR improvement (ISNR) produced by TwIST, $\mathrm{IST}_{\mathrm{opt}}$, and IST_1.