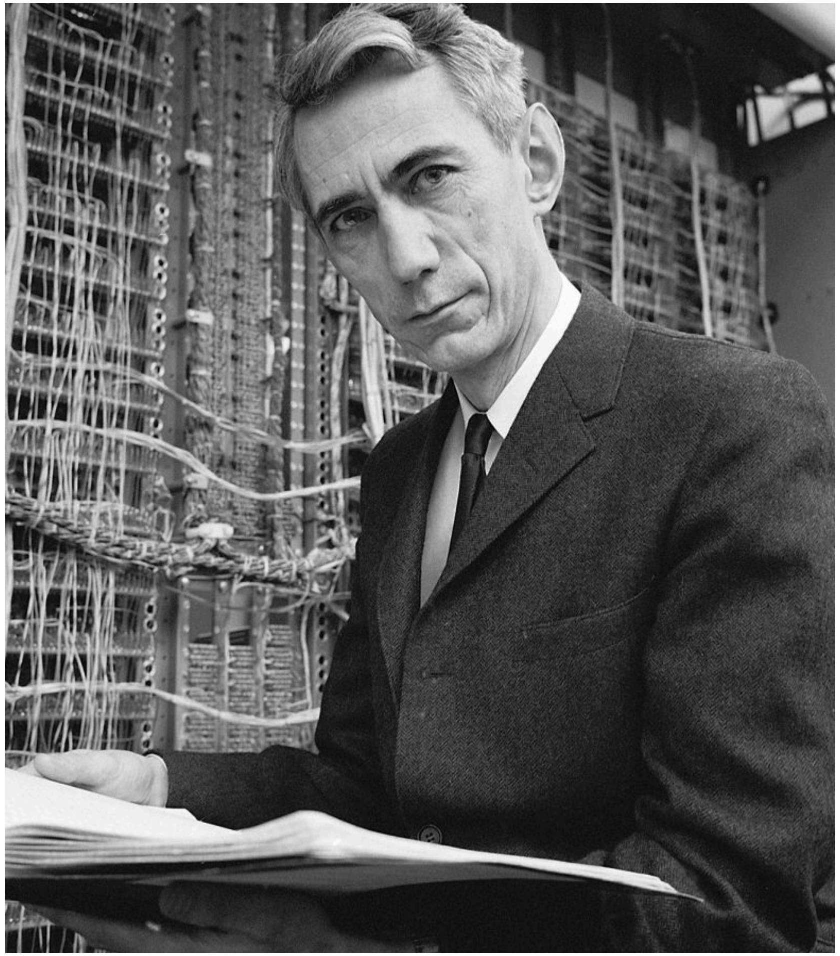
On Duality, Encryption, Sampling and Learning: the power of *codes*

Kannan Ramchandran
University of California, Berkeley

Berkeley Laboratory for Information and System Sciences

Shannon’s incredible legacy

- A mathematical theory of communication
- Channel capacity
- Source coding
- Channel coding
- Cryptography
- Sampling theory
- …



(1916-2001)

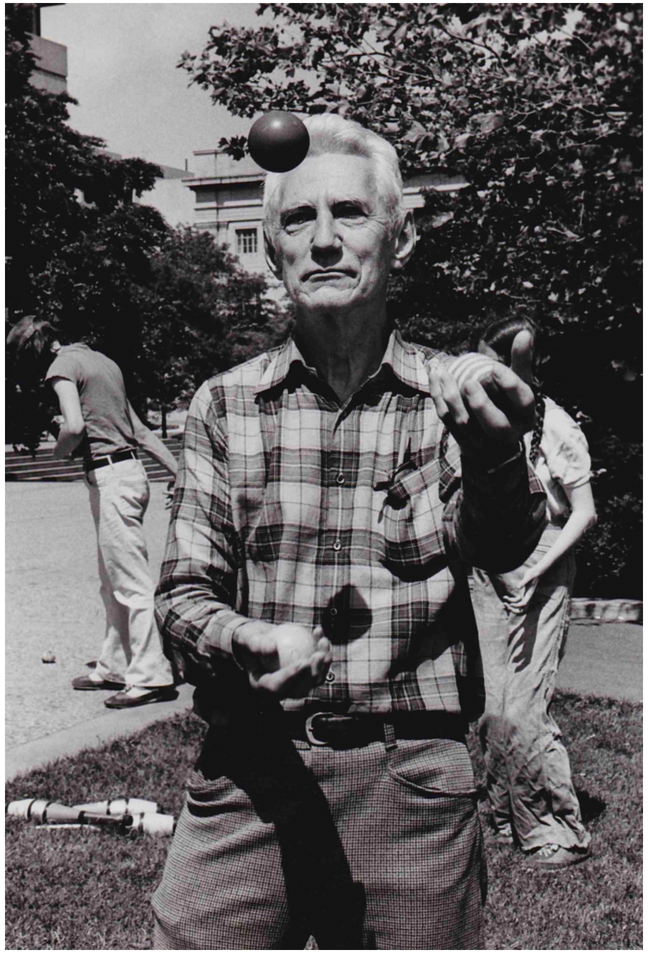
And many more…

- Boolean logic for switching circuits (MS thesis 1937)

- Juggling theorem:
  H(F+D)=N(V+D)

```
F: the time a ball spends in the air,
D: the time a ball spends in a hand,
V: the time a hand is vacant,
N: the number of balls juggled,
H: the number of hands.
```

- …



(1916-2001)

Story: Shannon meets Einstein

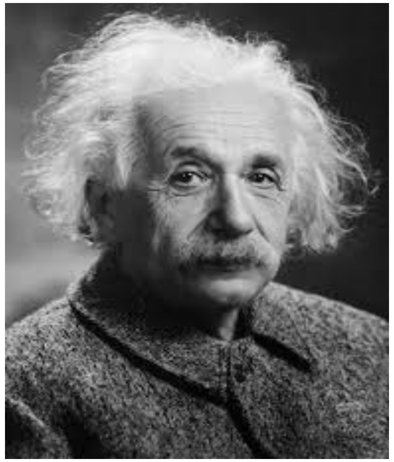
As narrated by Arthur Lewbel (2001)

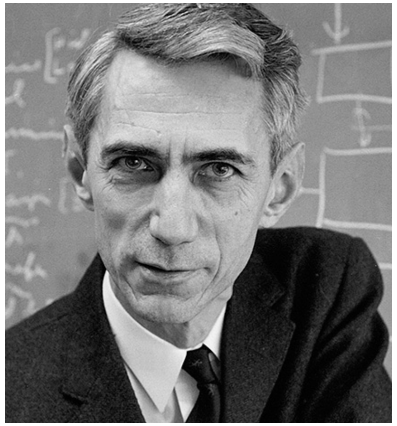
"

The story is that Claude was in the middle of giving a lecture to mathematicians in Princeton, when the door in the back of the room opens, and in walks **Albert Einstein**.

Einstein stands listening for a few minutes, whispers something in the ear of someone in the back of the room, and leaves. At the end of the lecture, Claude hurries to the back of the room to find the person that Einstein had whispered too, to find out what the great man had to say about his work.

The answer: Einstein had asked directions to the men's room.
"





Outline

Five “personal” Shannon-inspired research threads:

Chapter 1: **Duality** between source coding and channel coding – with side-information (2003)

Chapter 2: **Encryption** and **Compression** – swapping the order (2003)

Chapter 3: **Sampling** below Nyquist rate and efficient reconstruction (2014)

Chapter 4: **Learning** and inference exploiting sparsity – sub-linear time algorithms (2015-Present)

Chapter 5: **Codes** for distributed computing & machine learning (2017-Present)

Sandeep Pradhan

Jim Chou

Chapter 1

Duality

- source & channel coding
- with side-information

Shannon's celebrated 1948 paper

The Bell System Technical Journal

Vol. XXVII *July, 1948* *No. 3*

A Mathematical Theory of Communication

By C. E. SHANNON

Introduction

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have *meaning*; that is they refer to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem. The significant aspect is that the actual message is one *selected from a set* of possible messages. The system must be designed to operate for each possible selection, not just the one which will actually be chosen since this is unknown at the time of design.

If the number of messages in the set is finite then this number or any monotonic function of this number can be regarded as a measure of the information produced when one message is chosen from the set, all choices being equally likely. As was pointed out by Hartley the most natural choice is the logarithmic function. Although this definition must be generalized considerably when we consider the influence of the statistics of the message and when we have a continuous range of messages, we will in all cases use an essentially logarithmic measure.

The logarithmic measure is more convenient for various reasons:

1. It is practically more useful. Parameters of engineering importance

[1] Nyquist, H., "Certain Factors Affecting Telegraph Speed," *Bell System Technical Journal*, April 1924, p. 324; "Certain Topics in Telegraph Transmission Theory," *A. I. E. E. Trans.*, v. 47, April 1928, p. 617.

[2] Hartley, R. V. L., "Transmission of Information," *Bell System Technical Journal*, July 1928, p. 535.

general theory of communication

communication system as source/channel/destination

abstraction of the concept of message

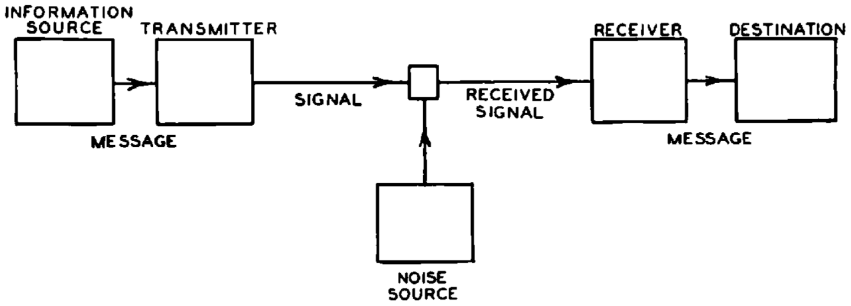
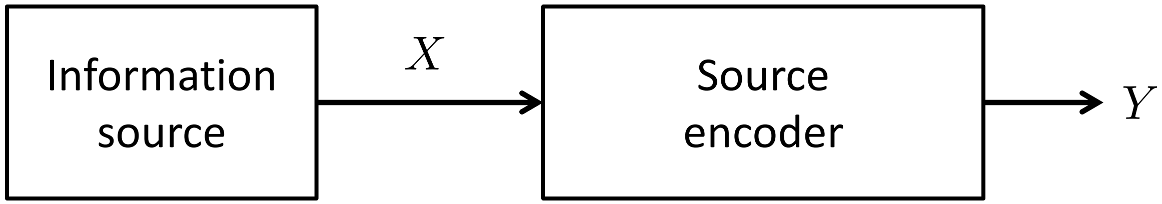


Fig. 1—Schematic diagram of a general communication system.

Source coding



$$H(X) = \mathbb{E}_X\left[\log\left(\frac{1}{p(X)}\right)\right]$$

Entropy of a random variable
= minimum number of bits required to represent the source

Rate-distortion theory - 1948

- Trade-off between ***compression rate*** and the ***distortion***

PART V: THE RATE FOR A CONTINUOUS SOURCE

27. Fidelity Evaluation Functions

In the case of a discrete source of information we were able to determine a definite rate of generating information, namely the entropy of the underlying stochastic process. With a continuous source the situation is considerably more involved. In the first place a continuously variable quantity can assume an infinite number of values and requires, therefore, an infinite number of binary digits for exact specification. This means that to transmit the output of a continuous source with *exact recovery* at the receiving point requires, in general, a channel of infinite capacity (in bits per second). Since, ordinarily, channels have a certain amount of noise, and therefore a finite capacity, exact transmission is impossible.

This, however, evades the real issue. Practically, we are not interested in exact transmission when we have a continuous source, but only in transmission to within a certain tolerance. The question is, can we assign a definite rate to a continuous source when we require only a certain fidelity of recovery, measured in a suitable way. Of course, as the fidelity require-

Mutual information:

$\mathcal{H}(X)$-$\mathcal{H}(X|Y)$

$$R(D) = \min_{P_{Y|X}(y|x)} I(X;Y)$$

$$\text{subject to} \quad \mathbb{E}\left[d(X,Y)\right] \leq D$$

distortion measure

Channel coding

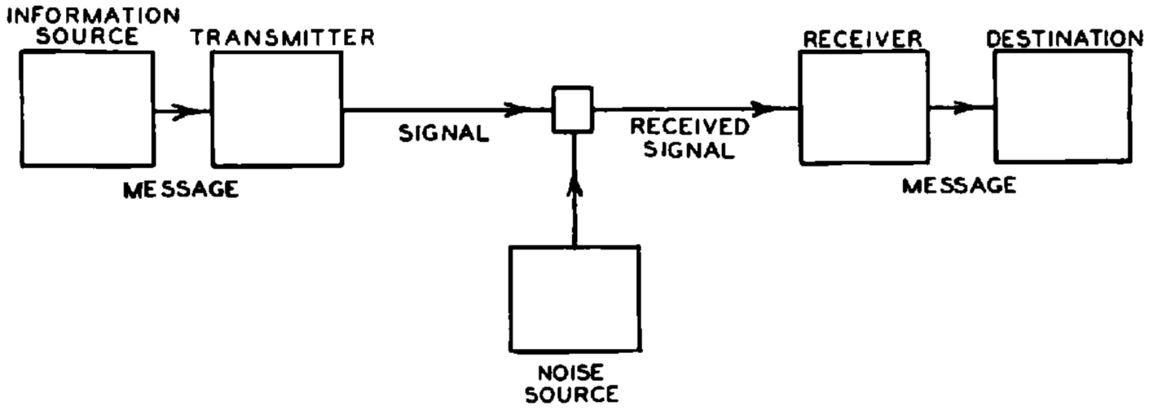


Fig. 1—Schematic diagram of a general communication system.

- For rates $R < C$, can achieve arbitrary small error probabilities
- Used to be thought one needs $R \rightarrow 0$

capacity

$$C(W) = \max_{P_X(x)} \quad I(X;Y)$$

$$\text{subject to} \quad \mathbb{E}\left[w(X)\right] \leq W$$

cost measure

Shannon's breakthrough

- Communication before Shannon:
  - ***Linear filtering*** (Wiener) at receiver to remove noise
- Communication after Shannon:
  - Designing codebooks
  - ***Non-linear estimation*** (MLE) at receiver

Reliable transmission at rates approaching channel capacity

Shannon (1959)

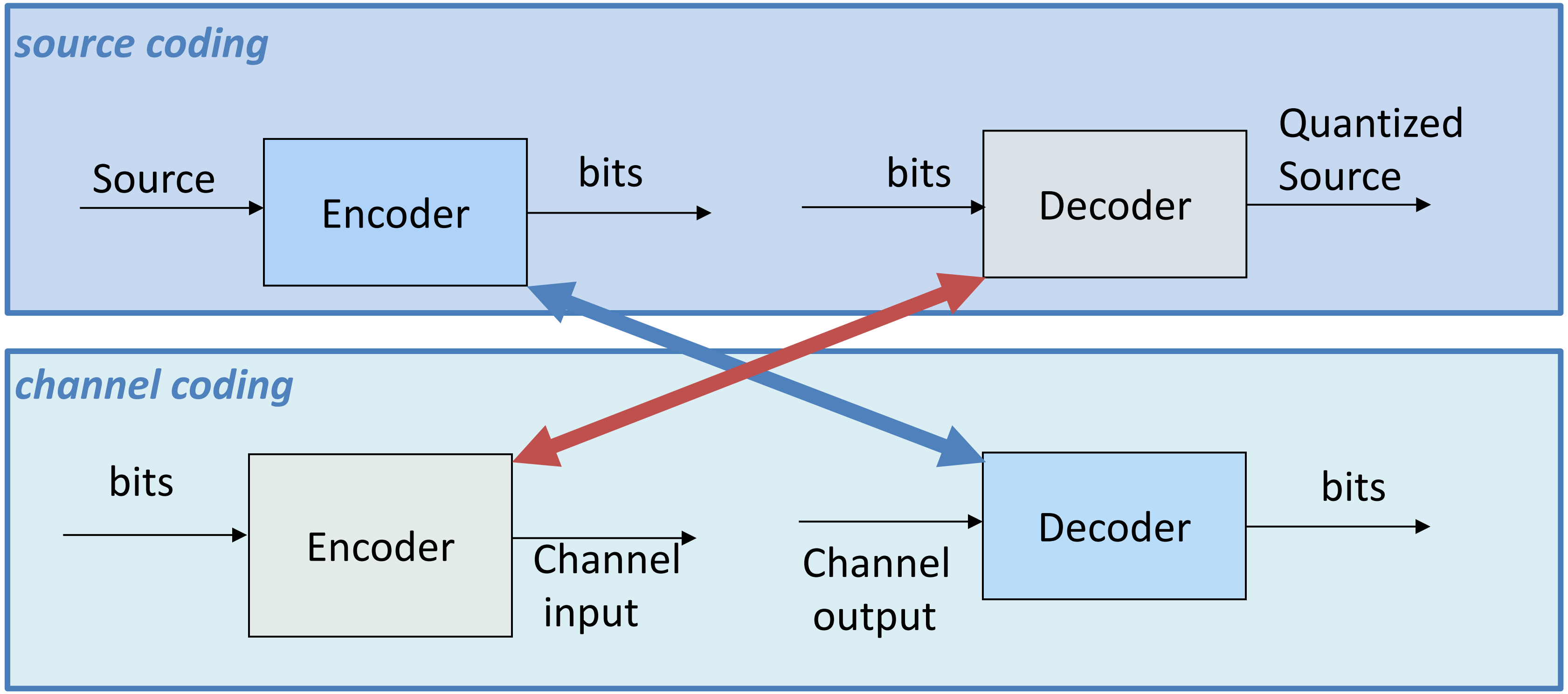
“***There is a curious and provocative duality between the properties of a source with a distortion measure and those of a channel.*** *This duality is enhanced if we consider channels in which there is a* ***cost*** *associated with the different input letters, and it is desired to find the capacity subject to the constraint that the expected cost not exceed a certain quantity…..*

Shannon (1959)

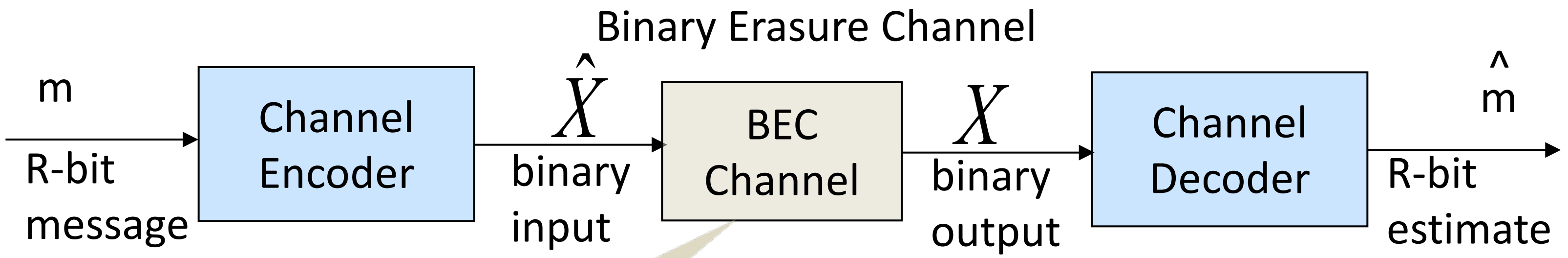
…This duality can be pursued further and is related to a duality between past and future and the notions of control and knowledge. ***Thus, we may have knowledge of the past but cannot control it; we may control the future but not have knowledge of it.***"

Functional duality

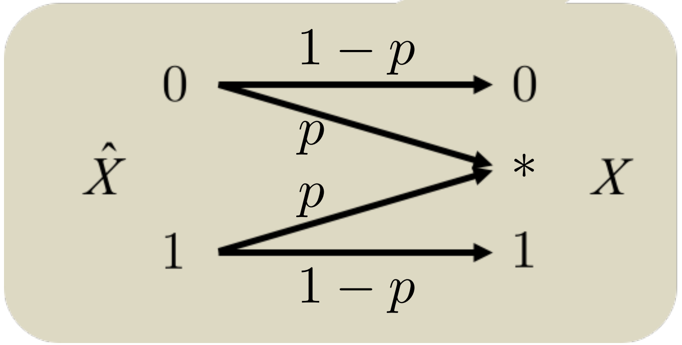
When is the ***optimal encoder*** for one problem functionally identical to the ***optimal decoder*** for the dual problem?



Duality example: Channel coding



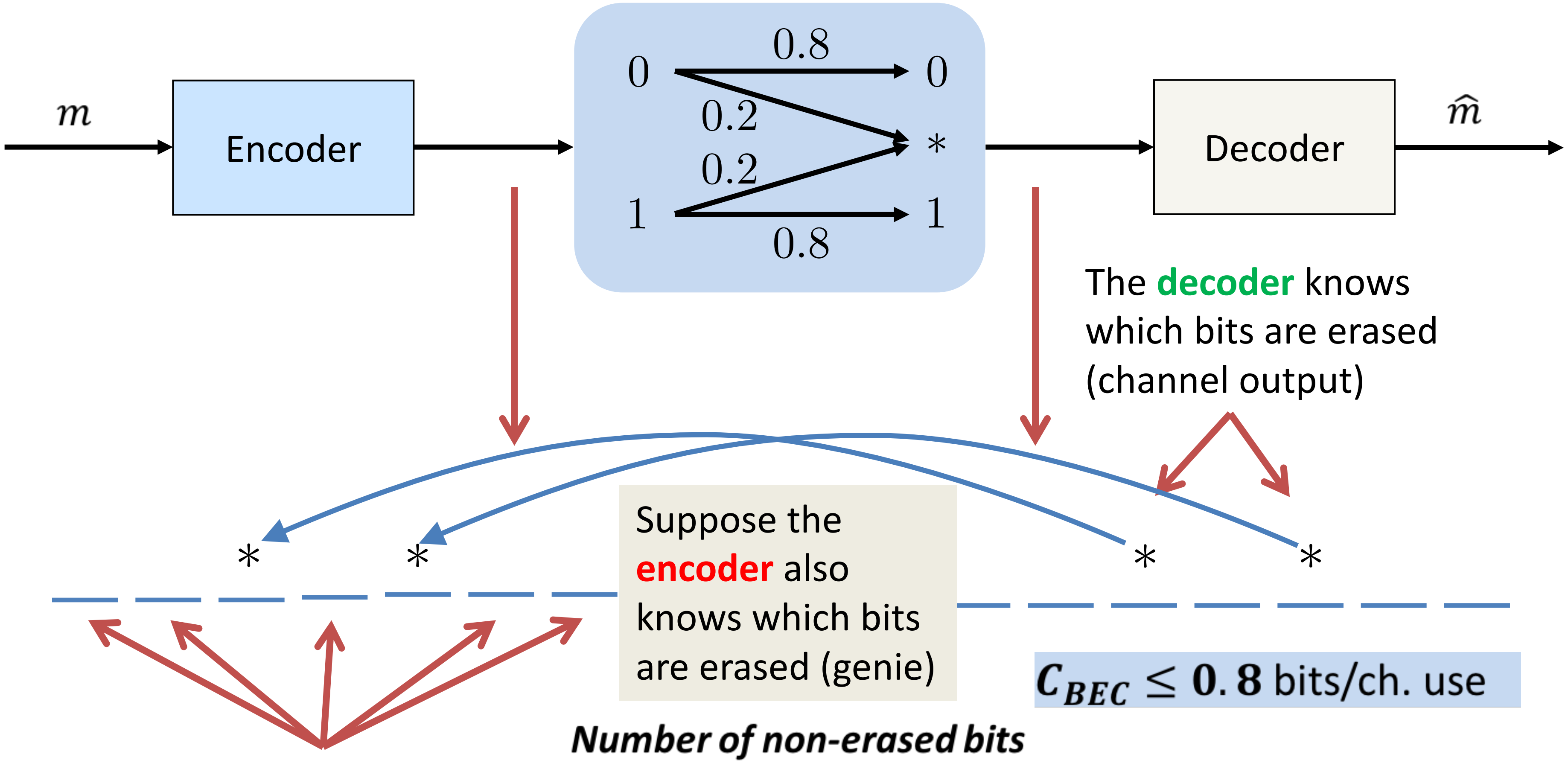
You want to send message m: how big can you make R?



Shannon's result:
C_{BEC}=(1-p) bits
per channel use

$$p = 0.2$$
$$\boldsymbol{Cost}\ (\mathbf{0}) = 1\,;\ \boldsymbol{Cost}\ (\mathbf{1}) = 1$$
$$\boldsymbol{Total\ budget} \leq 10{,}000$$

What is the Shannon capacity?



Number of non-erased bits

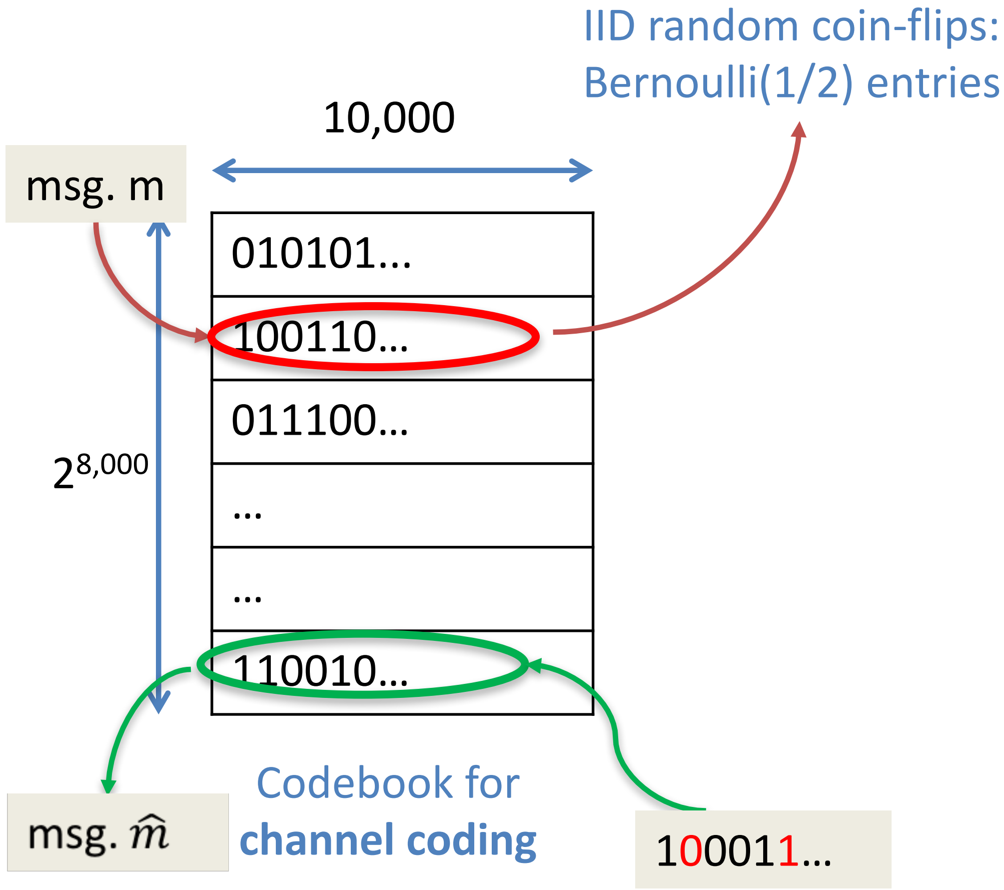
$\approx \mathbf{10,000 \times (1-p)}$

$= \mathbf{10,000 \times 0.8 = 8,000}$

$C_{BEC} \leq \mathbf{0.8}$ bits/ch. use

Surprise: ***the encoder does not need to know which bits are erased!***

Shannon's prescription: random coding



1) **Encoder & Decoder agree on a random codebook**

Shannon's random coding argument

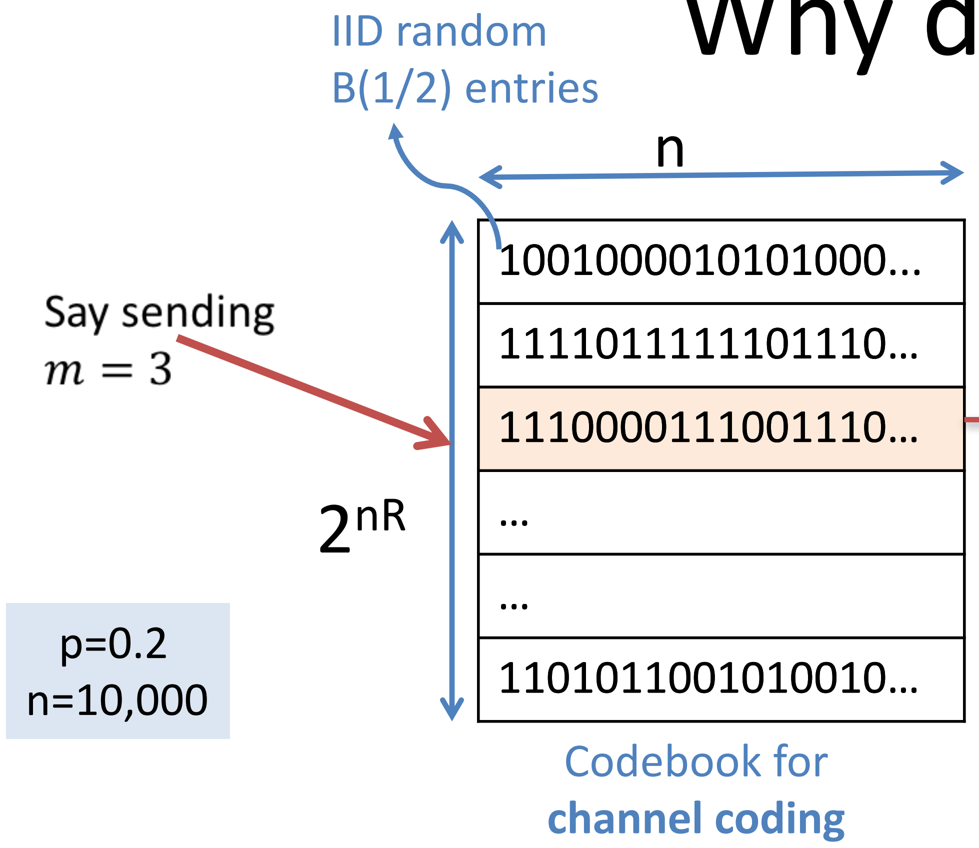
2) **Encoder encodes message**

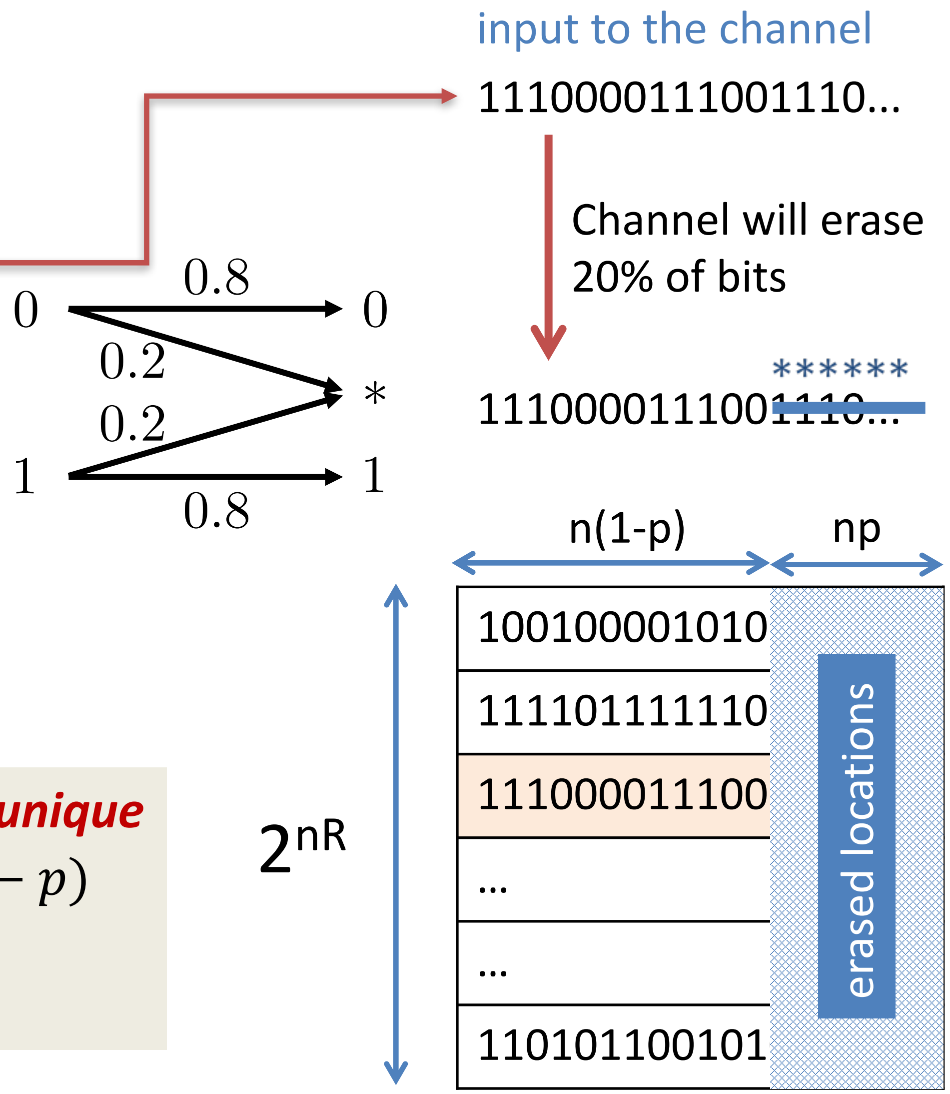
Output the codeword corresponding to the index

3) **Decoder decodes message**

*Output the index corresponding to the **closest** codeword*

Why does it work?

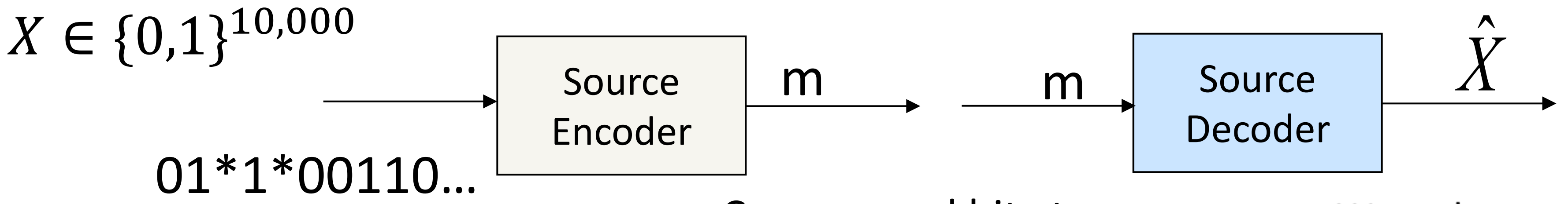




- Decoding successful if the non-erased string is ***unique***
- Pr.{***not unique***} $\leq 2^{-n(1-p)}$ x 2^{nR} ➔ 0 if R $\leq (1-p)$
- *8,000 bits will induce unique match if (random) codebook size is $\leq 2^{8{,}000}$ w.h.p.*

Source Coding Dual to the BEC: BEQ

(Binary Erasure Quantization)



Compressed bit-stream
8,000 bits

Want the average distortion to be ≤ 0.2

$$\boldsymbol{p(0) = p(1) = 0.4;}$$
$$\boldsymbol{p(*) = 0.2}$$

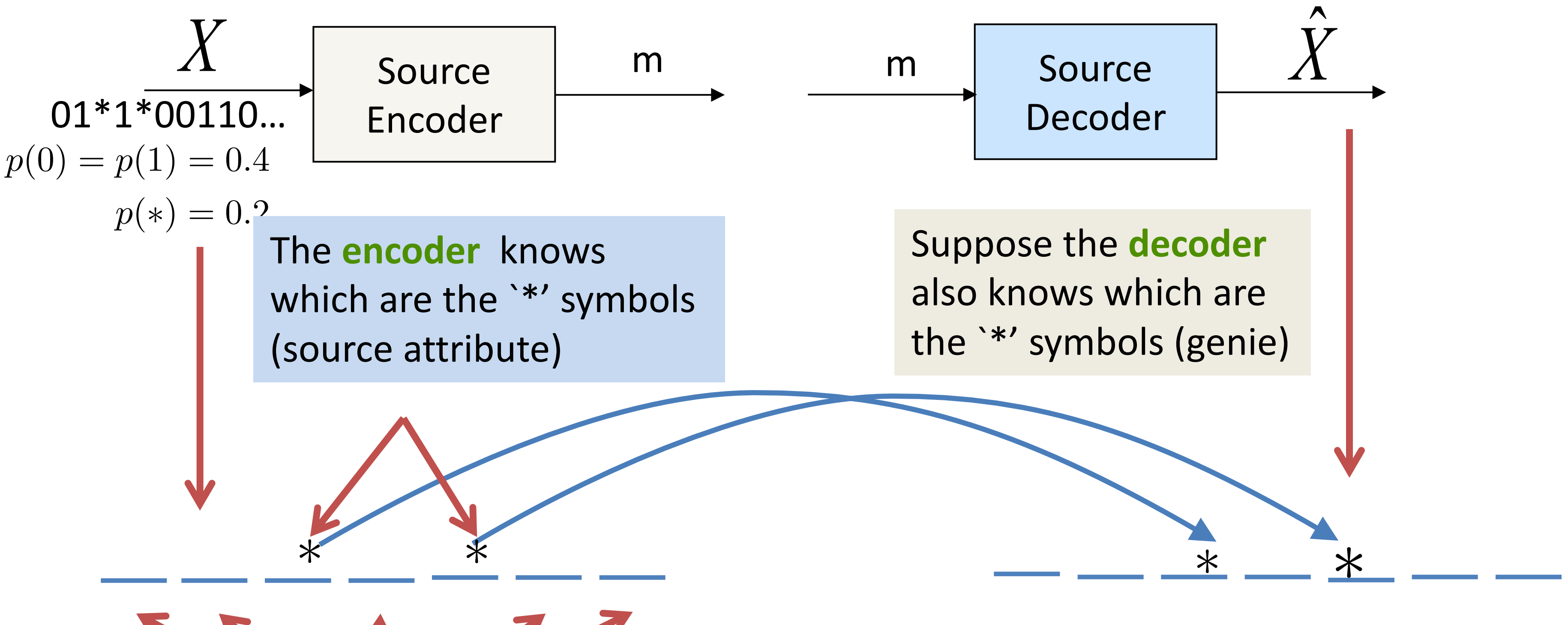
$$d(x,\hat{x}) = \begin{cases} 0 & \text{if } \hat{x} = x \text{ for } x \in \{0,1\} \\ \infty & \text{if } \hat{x} \neq x \text{ for } x \in \{0,1\} \\ 1 & \text{if } x = * \end{cases}$$

x:	1	0	$*$	$*$	0	1
$\hat{x}$:	1	0	1	0	1	0
cost:	0		1		∞	

$*$ is like a "don't care" symbol (e.g., perceptually masked symbols). How can we exploit this for compression?

Martinian and Yedidia, 2004

Source Coding Dual to the BEC: BEQ



$R_{BEQ}(0.2) \geq 0.8\ bits/symbol$

Send the non-* bits:

01100110…

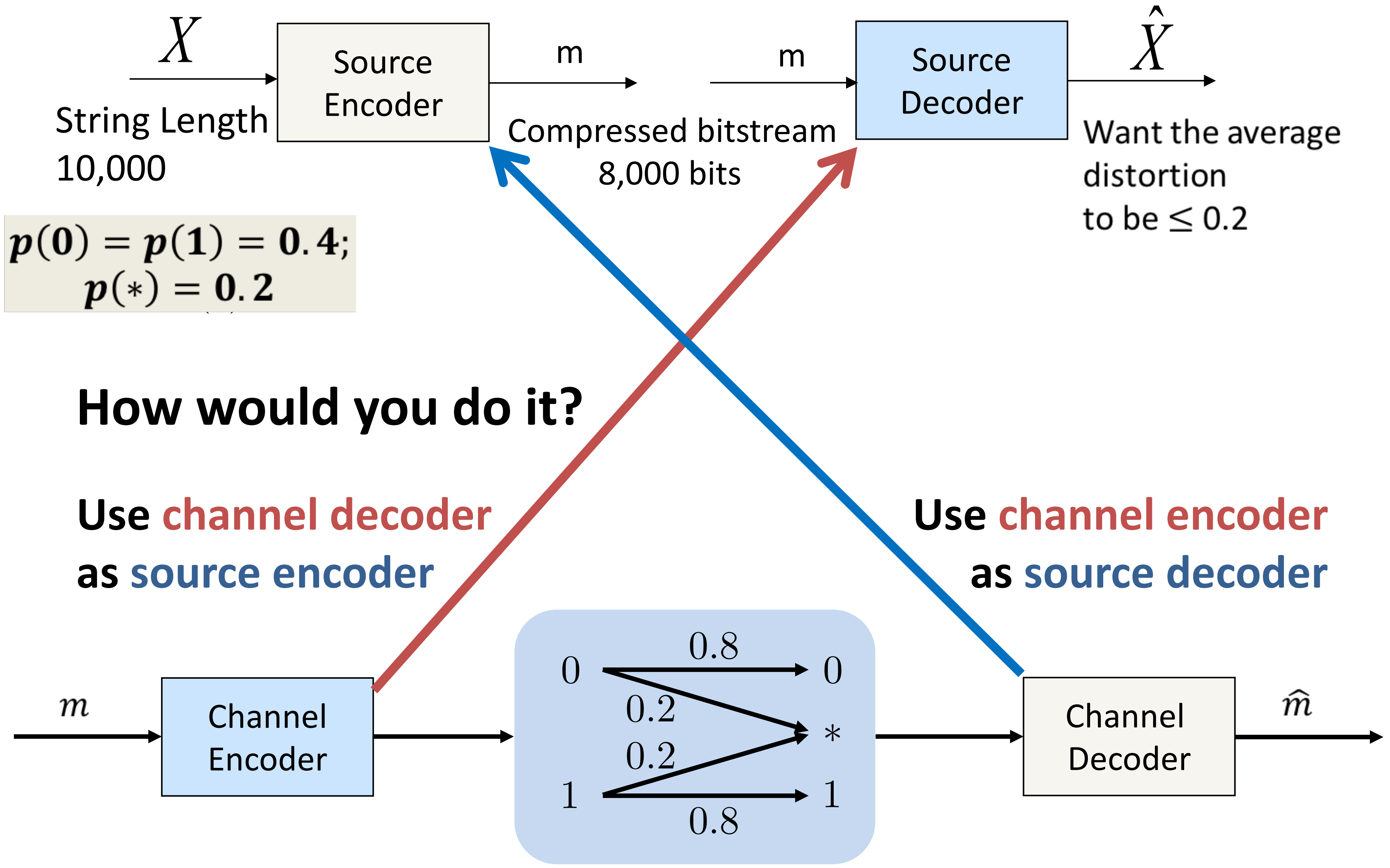
Number of non '$*$' symbols to send

$\approx \mathbf{10,000 \times (1 - p(*))}$

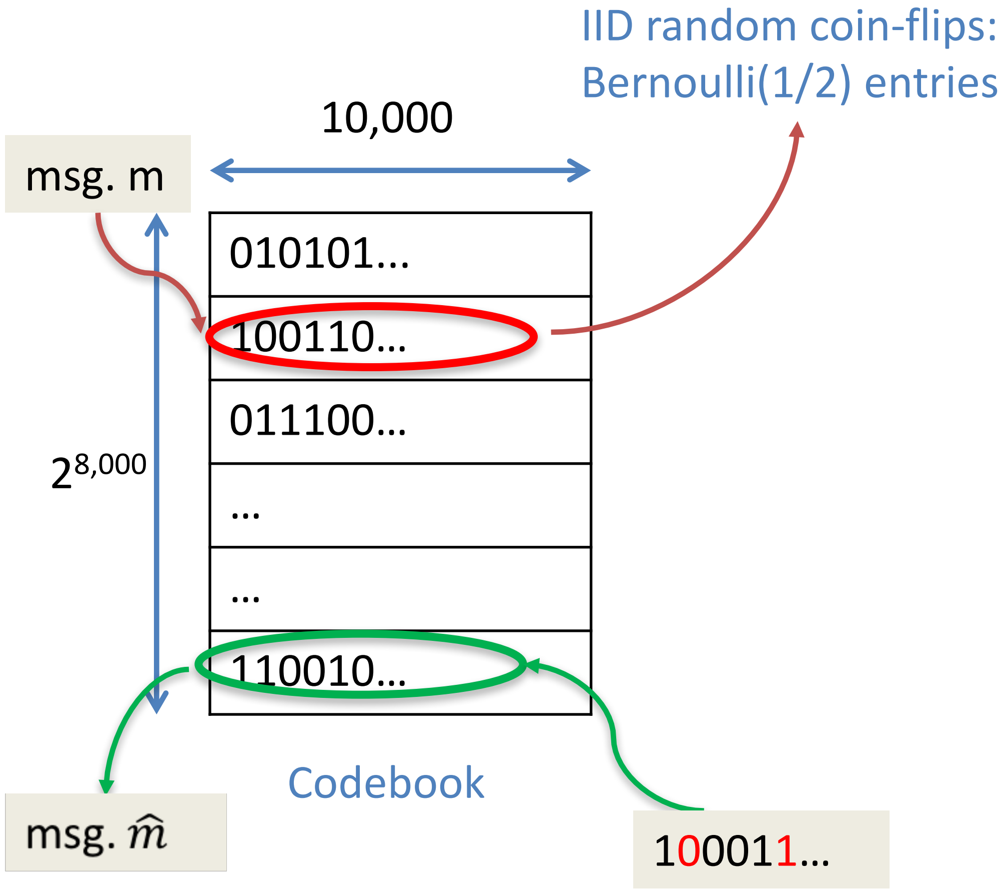
$\mathbf{= 10,000 \times 0.8 = 8,000}$

<u>Surprise:</u> ***the decoder does not need to know which symbols are '$*$'!***

Source Coding Dual to the BEC: BEQ



Shannon’s prescription: random coding



1) Encoder & Decoder agree on a random codebook

Shannon’s random coding argument

2) Encoder encodes message

~~*Output the codeword corresponding to the index*~~

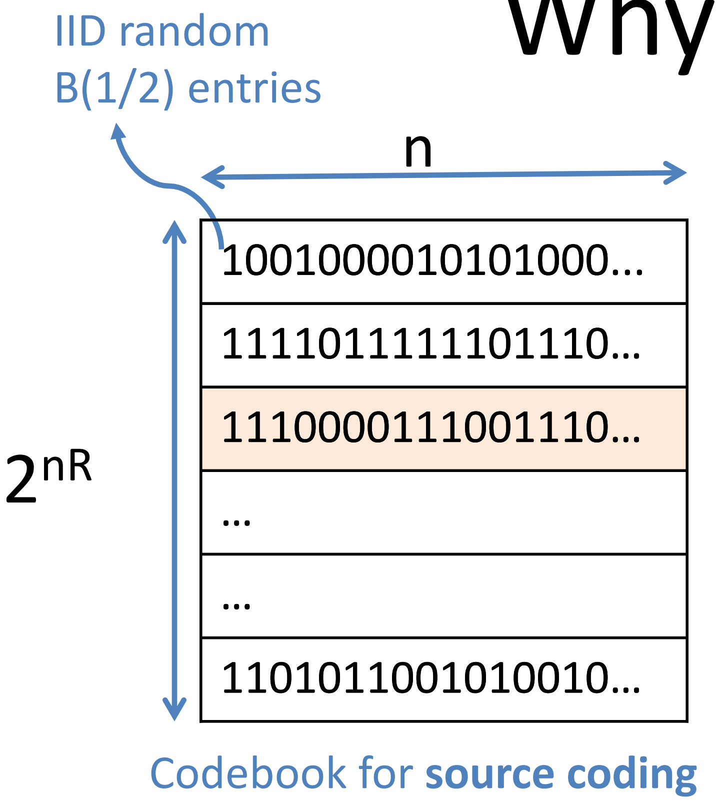
Output the index corresponding to the ***closest*** *codeword*

3) Decoder decodes message

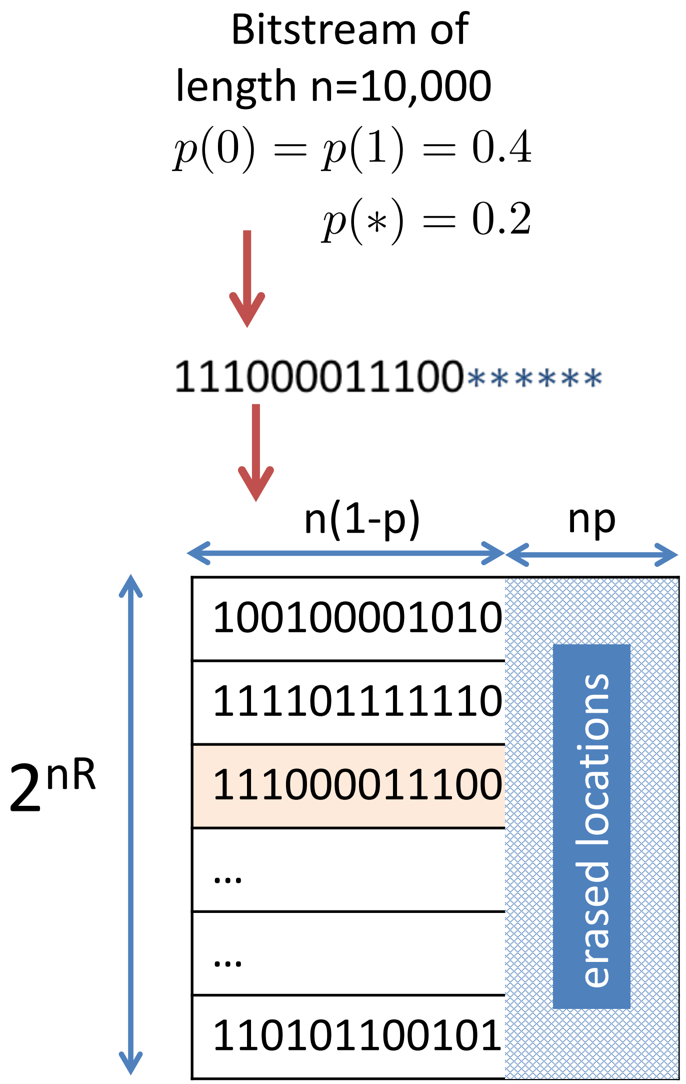
~~*Output the index corresponding to the **closest** codeword*~~

Output the codeword corresponding to the index

Why does it work?

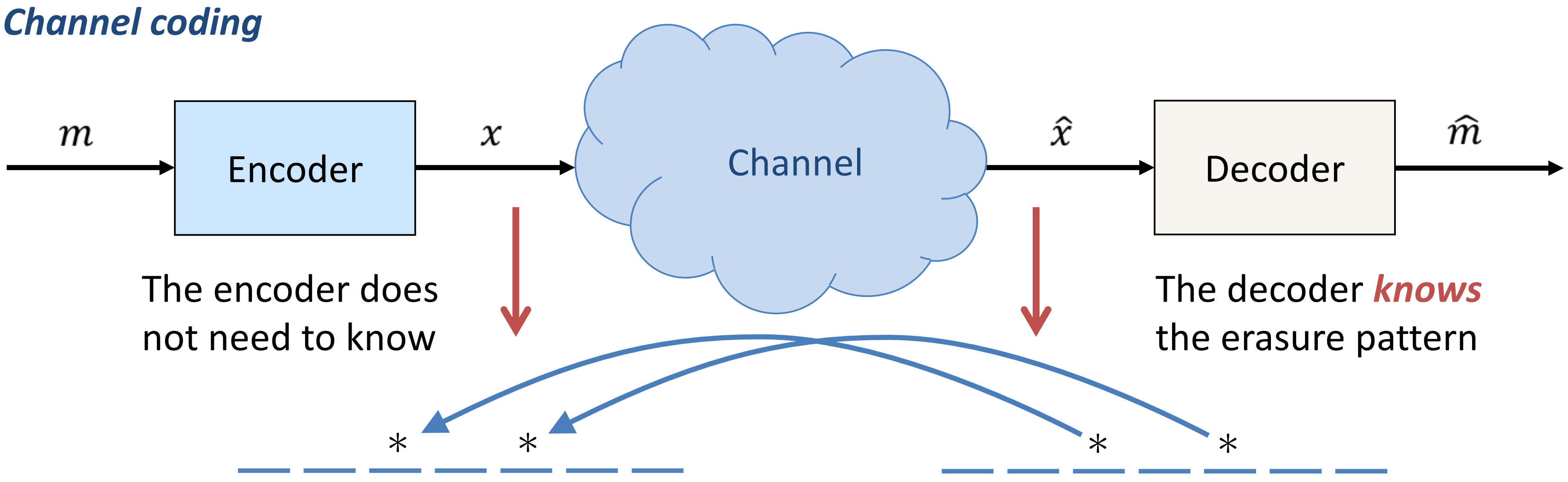


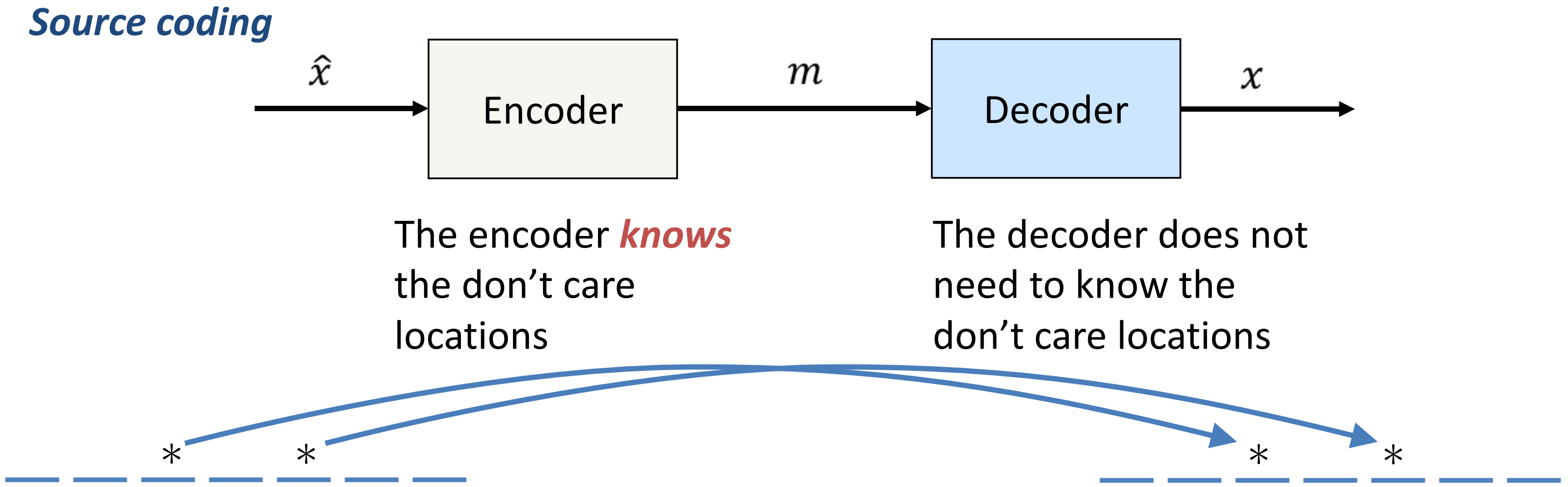
p=0.2
n=10,000



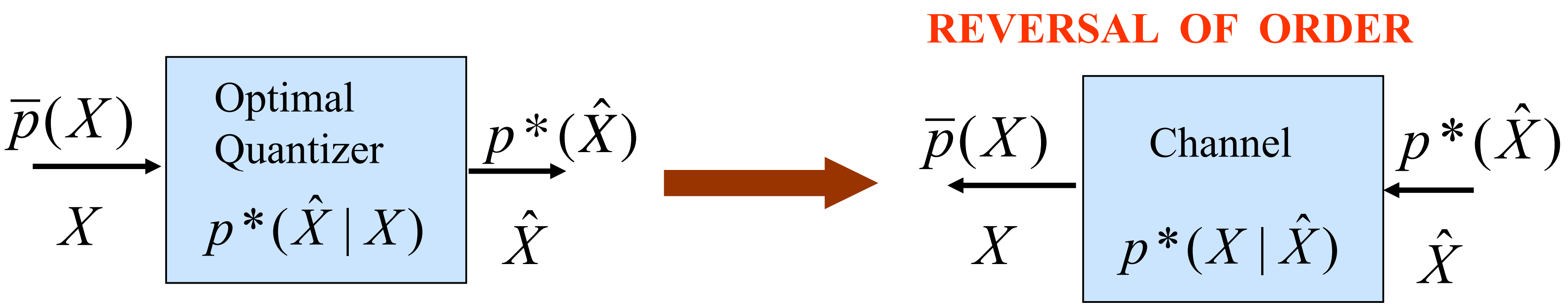
- Encoding successful if there exists an ***exact match*** for the non-* part of input string
- Pr.{***no exact match***} $\leq (1 - 2^{-n(1-p)})$ ^ 2^{nR} ➔ 0 if R $\geq (1-p)$
- *8,000 source bits will induce an exact match w.h.p. if random codebook size is at least $2^{8,000}$*

Knowledge of the erasure pattern





Duality between source and channel coding:



Given a source coding problem with source distr. $\bar{p}(X)$, optimal quantizer $p^*(\hat{X} \mid X)$ distortion measure $d(x, \hat{x})$ and distortion constraint **D**, (left) ,

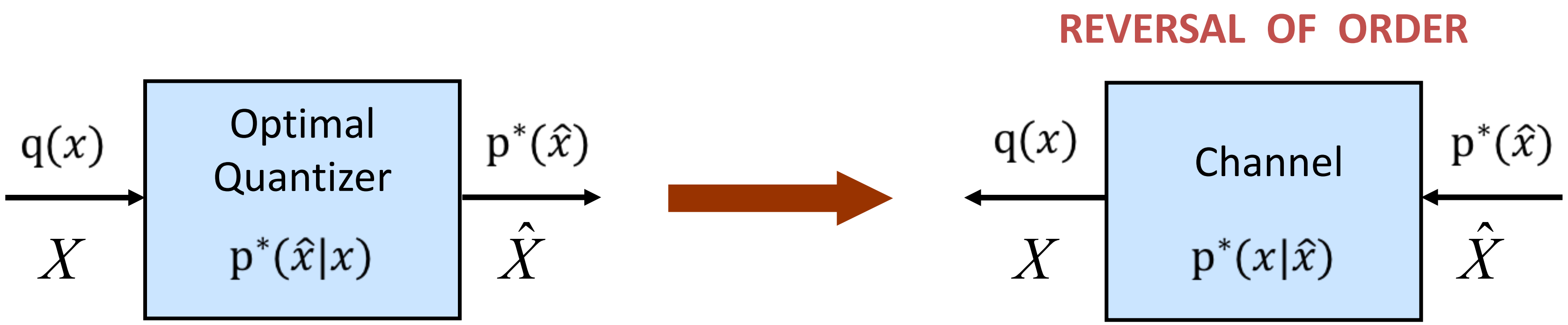
$\exists$ a **dual** channel coding problem with channel $p^*(x \mid \hat{x})$, cost measure $w(\hat{x})$, and cost constraint **W** (right) s.t.:

(i) R(**D**)=C(**W**);

(ii) $$p^*(\hat{x}) = \arg\max_{p(\hat{x}):X|\hat{X} \sim p^*(x|\hat{x}), Ew \le W} I(X; \hat{X}),$$

where $\boxed{w(\hat{x}) \stackrel{\Delta}{=} c_1 D(p^*(x \mid \hat{x}) \,\|\, \bar{p}(x)) + \theta}$ and $\boxed{W = E_{p^*(\hat{x})} w(\hat{X}).}$

Duality between source and channel coding



Given a ***source coding problem*** with source distribution $\mathrm{q}(x)$, optimal quantizer $\mathrm{p}^*(\hat{x}|x)$, distortion measure $d(x,\hat{x})$ and distortion constraint **D**

There is a ***dual channel coding problem*** with channel $\mathrm{p}^*(x|\hat{x})$ cost measure $w(\hat{x})$ and cost constraint **W** such that

$$\boldsymbol{R(D) = C(W)}$$

$$w(\hat{x}) = c_1 D(p^*(x|\hat{x})\,||\,q(x)) + \theta \qquad W = E_{p^*(\hat{x})} w(\hat{X}).$$

Pradhan, Chou and Ramchandran, 2003

Interpretation of functional duality

For ***any*** given source coding problem, there is a ***dual*** channel coding problem such that:

- both problems induce the ***same optimal joint distribution***
- the ***optimal encoder*** for one is ***functionally identical*** to the ***optimal decoder*** for the other
- an appropriate ***channel-cost measure*** is associated

Key takeaway

Source coding

distortion measure is as important as the *source distribution*

Channel coding

channel cost measure is as important as the *channel conditional distribution*

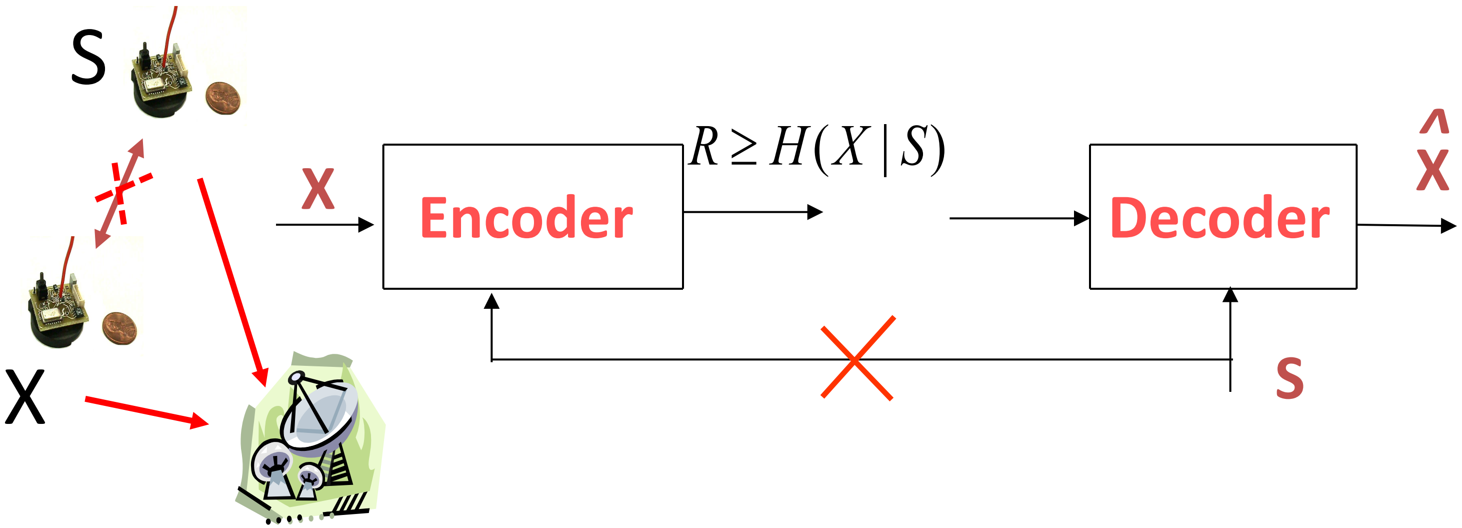
Duality between

source coding with side information

and

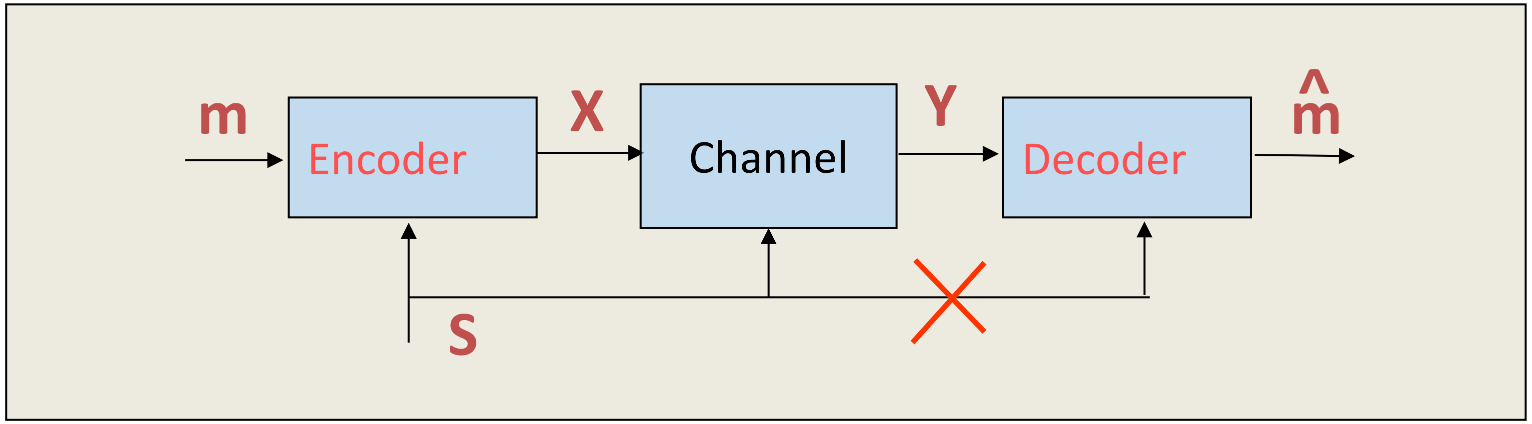
channel coding with side information

Source coding with side information (SCSI):



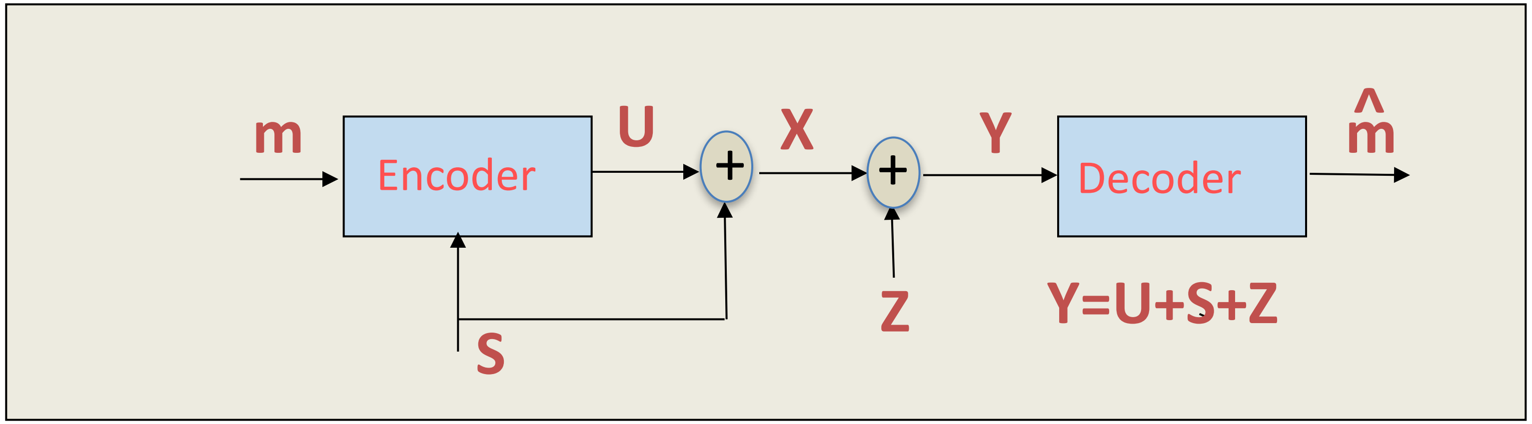
- (Only) decoder has access to side-information S
- Studied by Slepian-Wolf '73, Wyner-Ziv '76, Berger '77
- Applications: sensor networks (IoT), digital upgrade, secure compression.
- **No performance loss in some important cases**

Channel coding with side information (CCSI):



- (Only) encoder has access to ``interfering” side-information S
- Studied by Gelfand-Pinsker ‘81, Costa ‘83, Heegard-El Gamal ’85
- Applications: data hiding, watermarking, precoding for known interference, writing on dirty paper, MIMO broadcast.
- **No performance loss in some important cases**

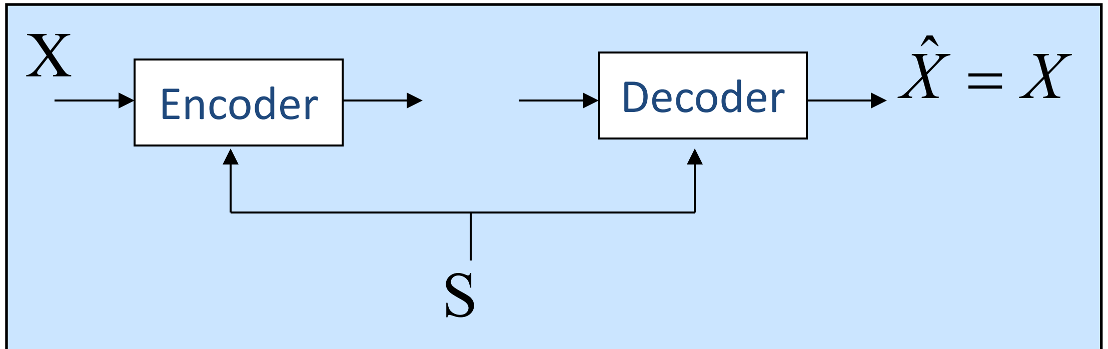
Channel coding with side information (CCSI):



- Encoder (only) has access to ``interfering” side-information S
- Studied by Gelfand-Pinsker ‘81, Costa ‘83, Heegard-El Gamal ’85
- Applications: data hiding, watermarking, precoding for known interference, writing on dirty paper, MIMO broadcast.
- **No performance loss in some important cases**

SCSI: binary example of noiseless compression

- X and S=> length-3 binary data (equally likely),
- Correlation: Hamming distance between X and S at most 1
- E.g.: when X=[0 1 0],   S => [0 1 0], [0 1 1], [0 0 0], [1 1 0].

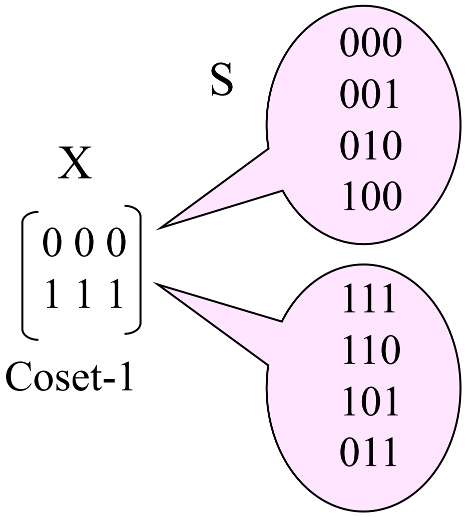


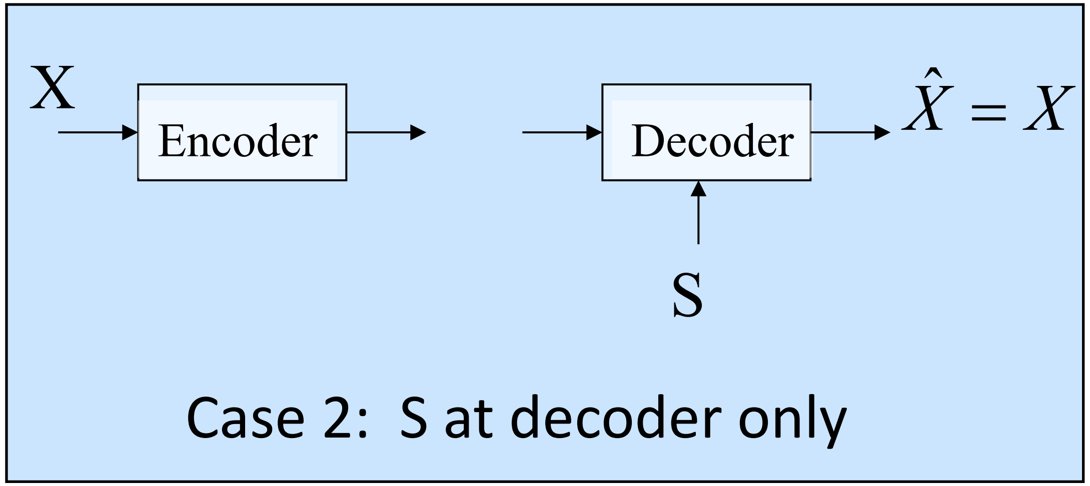
Case 1 (S at both ends)

Encoder computes   e=S+X (mod 2) and sends using 2 bits

Decoder outputs X=S+e (mod 2)

00 → 000
01 → 001
10 → 010
11 → 100
=X+S





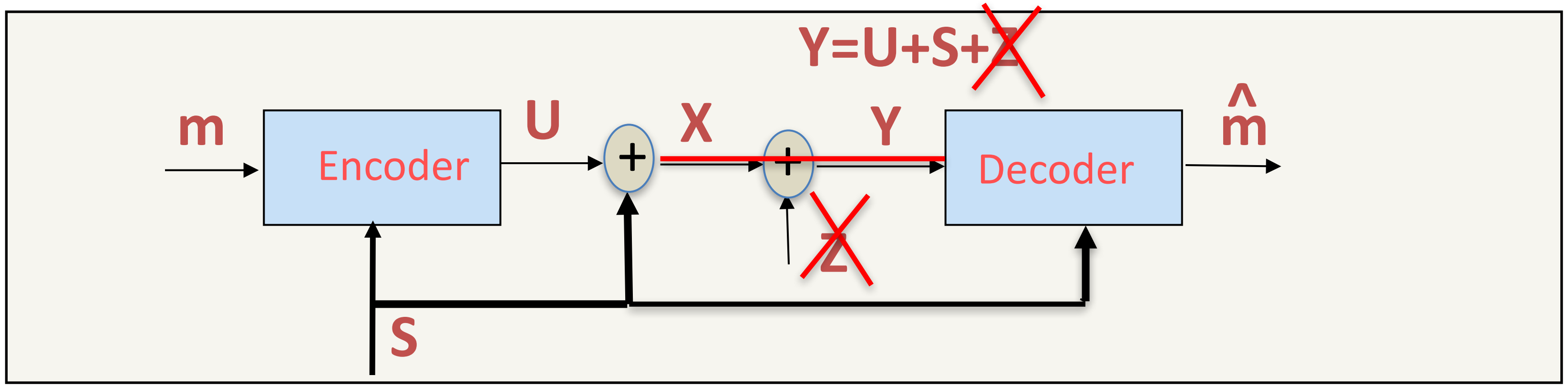
Coset-1 (00) $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$ Coset-2 (01) $\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

Coset-3 (10) $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$ Coset-4 (11) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$

- Transmission at **2 bits/sample achievable**
- Encoder => send index of the coset containing X.
- Decoder => find a codeword in given coset closest to S

Example: X=010, S=110 => Encoder sends message 10

CCSI: illustrative example *(Binary data-embedding/watermarking)*

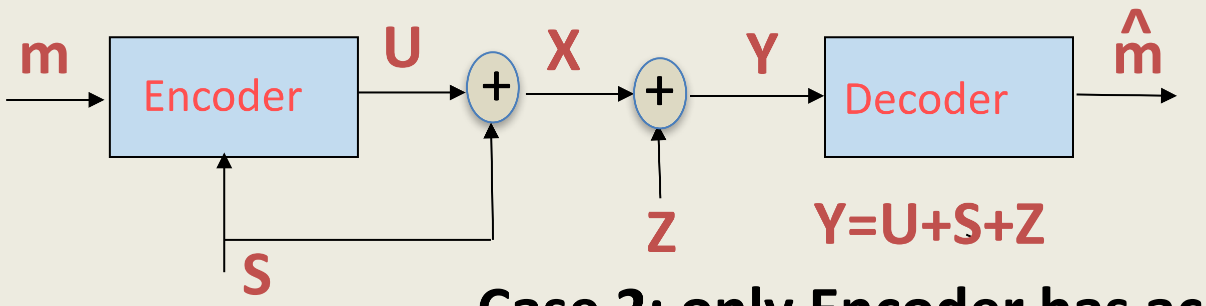


- **S**: **3-bit** (uniformly random) **host** signal (e.g. binary fax)
- **m**: message bits to be embedded in the host signal
- Max. allowed distortion between **S** and embedded host X is **1**: d_H(X,S)$\leq$ 1
- Clean channel (no attack) model: (**Z=0**); received signal **Y=X**

Case: 1: Both encoder and decoder have access to host signal

- Q) **How many bits can m be?**
- A) **2 bits**

m	U
00 →	000
01 →	001
10 →	010
11 →	100



Case 2: only Encoder has access to S

Q) Can we still embed a 2 bit message in S while satisfying $d_H(S,X) \leq 1$?

Messages index one of 4 cosets of U:

Coset-1 (00) $\begin{bmatrix} 0\,0\,0 \\ 1\,1\,1 \end{bmatrix}$

Coset-3 (10) $\begin{bmatrix} 0\,1\,0 \\ 1\,0\,1 \end{bmatrix}$

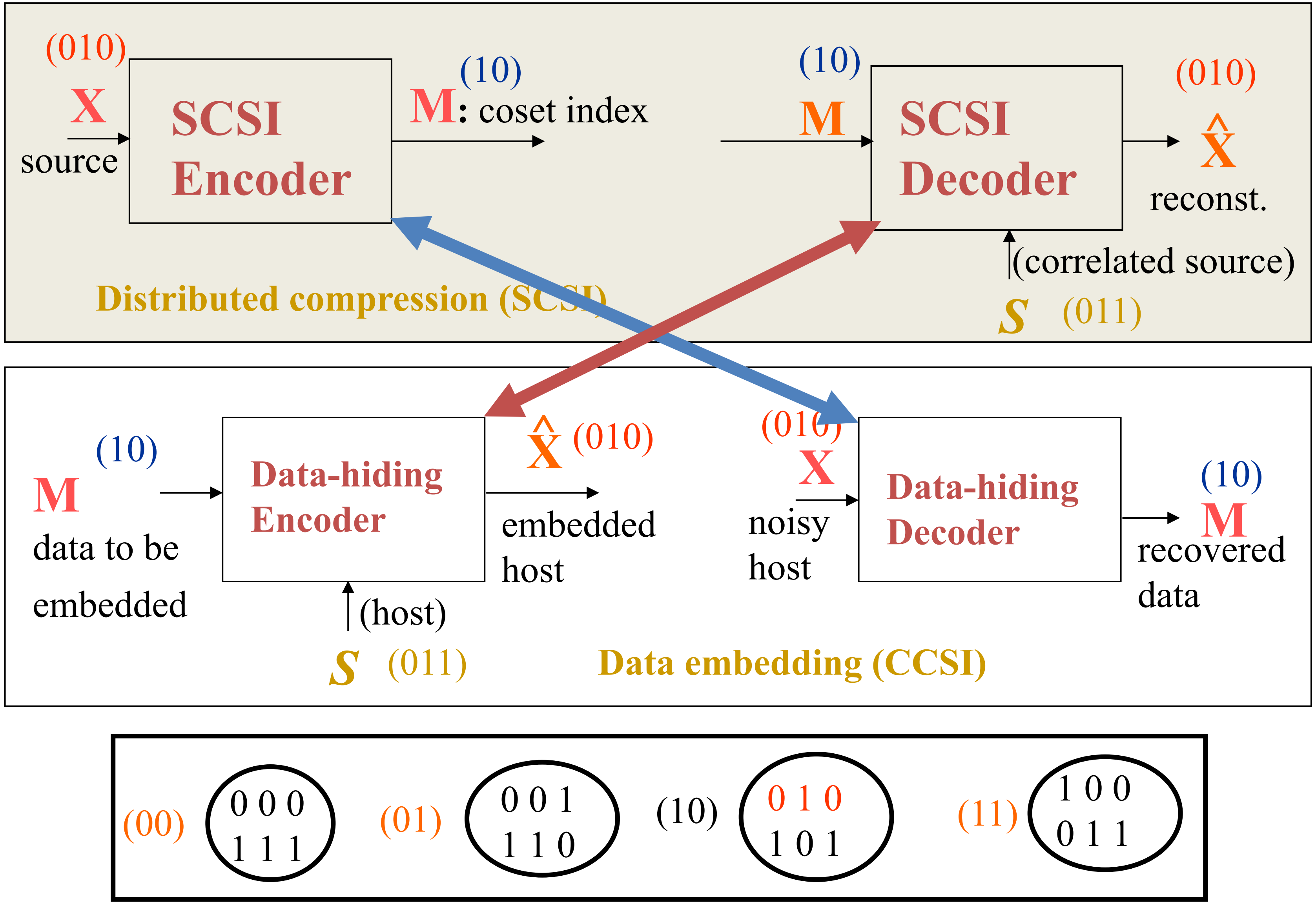
Coset-2 $\begin{bmatrix} 0\,0\,1 \\ 1\,1\,0 \end{bmatrix}$ 01

Coset-4 (11) $\begin{bmatrix} 1\,0\,0 \\ 0\,1\,1 \end{bmatrix}$

Example: S=011, m=01;
X=001 (off in $\leq$ 1 bit)

- **Codebook: partition U into 4 cosets**
- **Each of 4 messages indexes a coset in U.**
- **Encoder "nudges" S to closest entry X in desired coset of U:** $d_H(S,X) \leq 1$
- **Decoder receives Y=X and declares coset index of Y as message sent.**

Toy example of duality between SCSI and CCSI



Duality (loose sense)

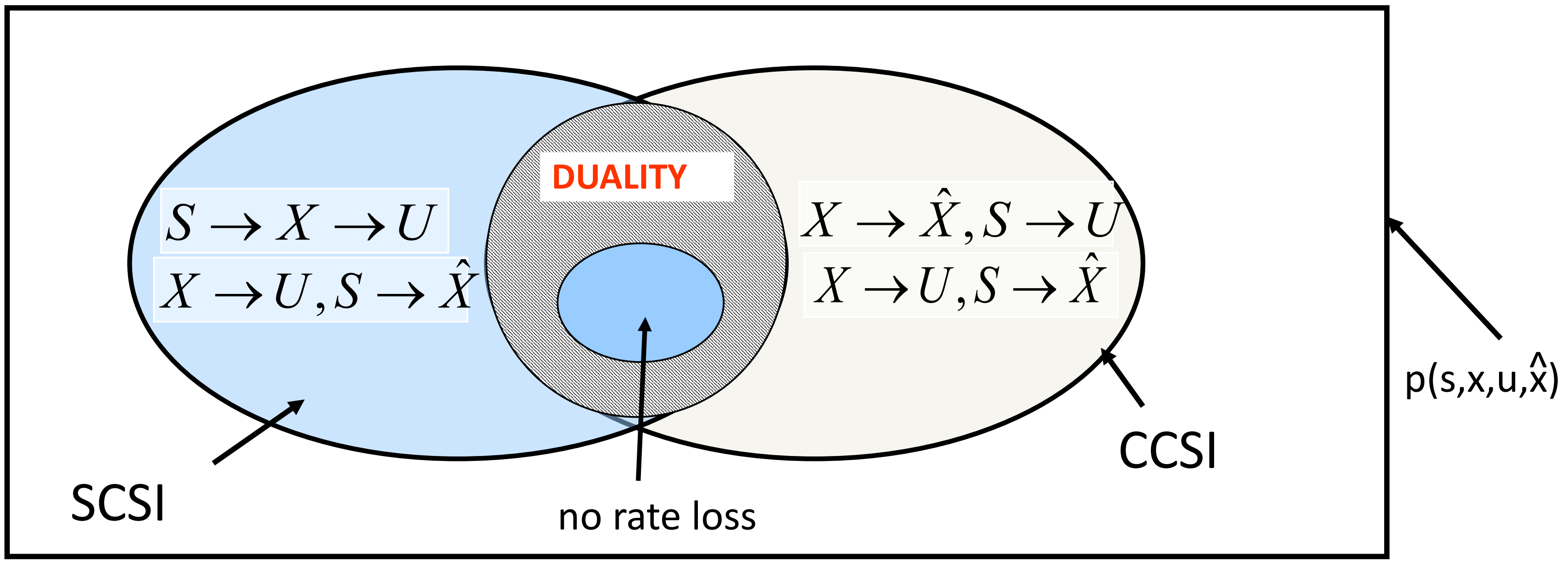
CCSI

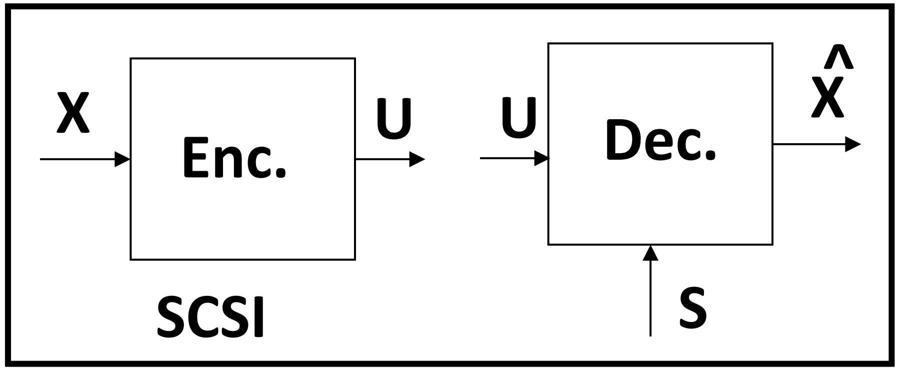
- Side information at encoder only
- Channel code is "partitioned" into a bank of source codes
- No performance loss in some important cases w.r.t. presence of side information at both ends

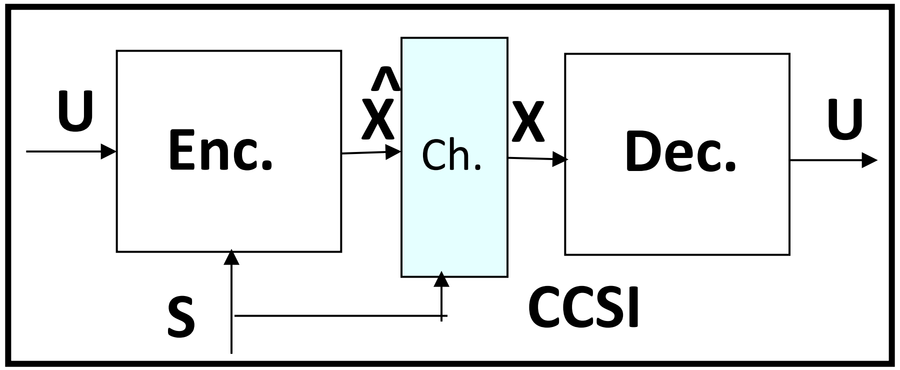
SCSI

- Side info. at decoder only
- Source code is "partitioned" into a bank of channel codes
- No performance loss in some important cases w.r.t. presence of side information at both ends

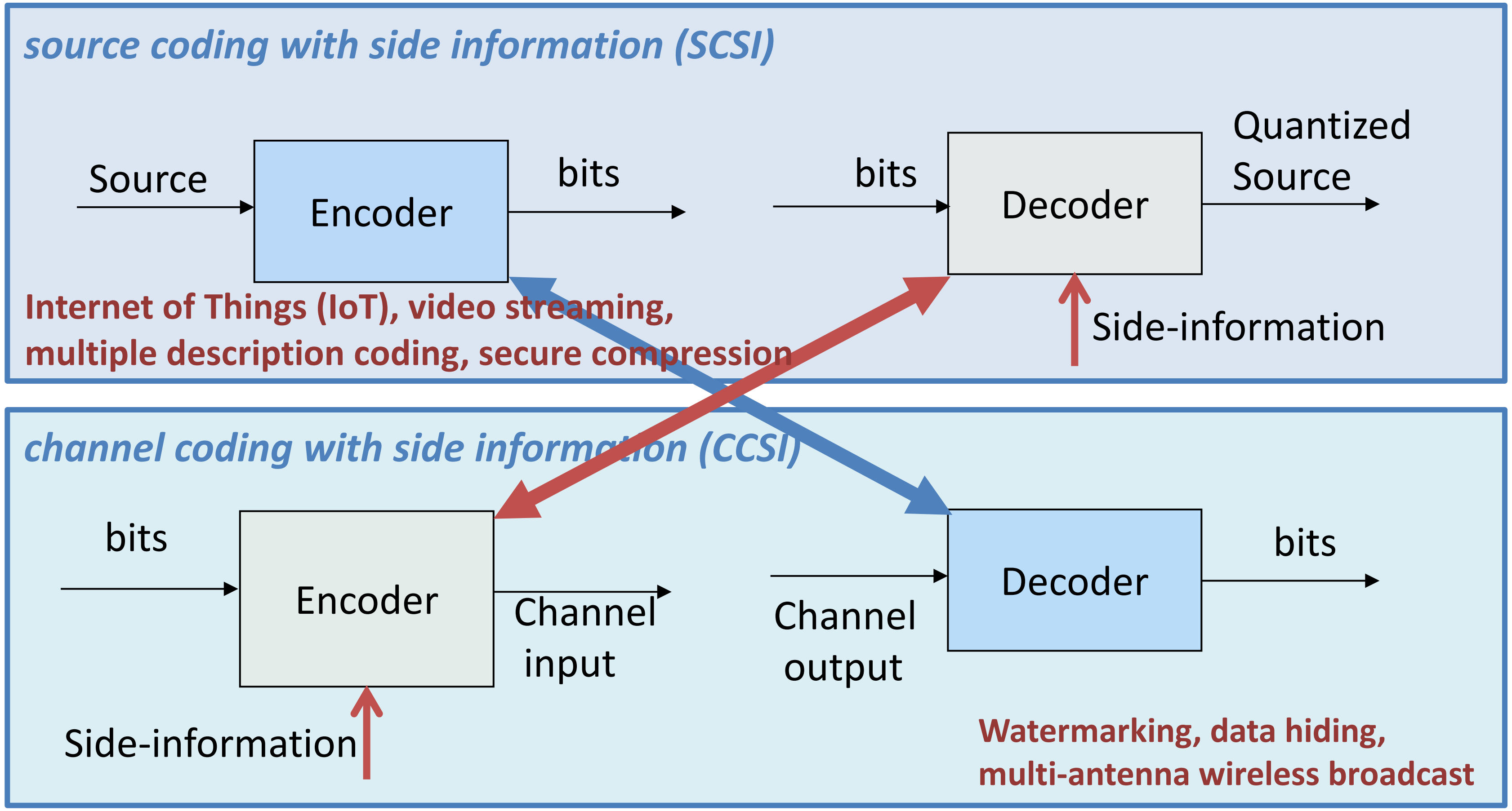
Markov chains, duality and rate loss







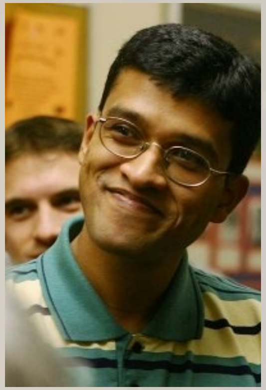
Duality between *source coding* & *channel coding with side information*



Pradhan, Chou and Ramchandan, 2003

Mark Johnson

Prakash Ishwar



Vinod Prabhakaran

Chapter 2

Cryptography

- Compressing encrypted data

Cryptography – 1949

- Foundations of ***modern cryptography***
- All theoretically unbreakable ciphers must have the properties of one-time pad

Communication Theory of Secrecy Systems*

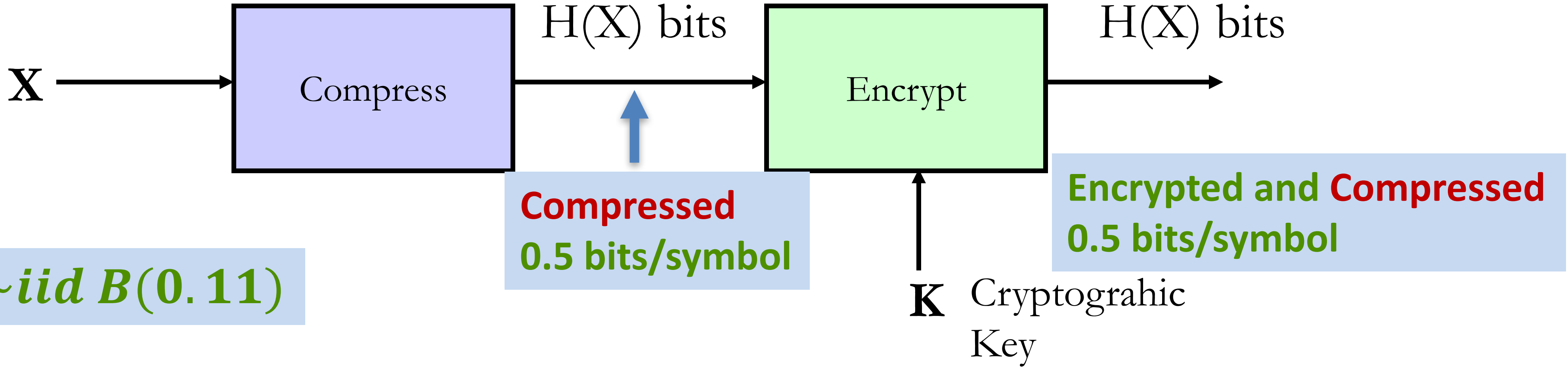
By C. E. SHANNON

1. INTRODUCTION AND SUMMARY

THE problems of cryptography and secrecy systems furnish an interesting application of communication theory.[1] In this paper a theory of secrecy systems is developed. The approach is on a theoretical level and is intended to complement the treatment found in standard works on cryptography.[2] There, a detailed study is made of the many standard types of codes and ciphers, and of the ways of breaking them. We will be more concerned with the general mathematical structure and properties of secrecy systems.

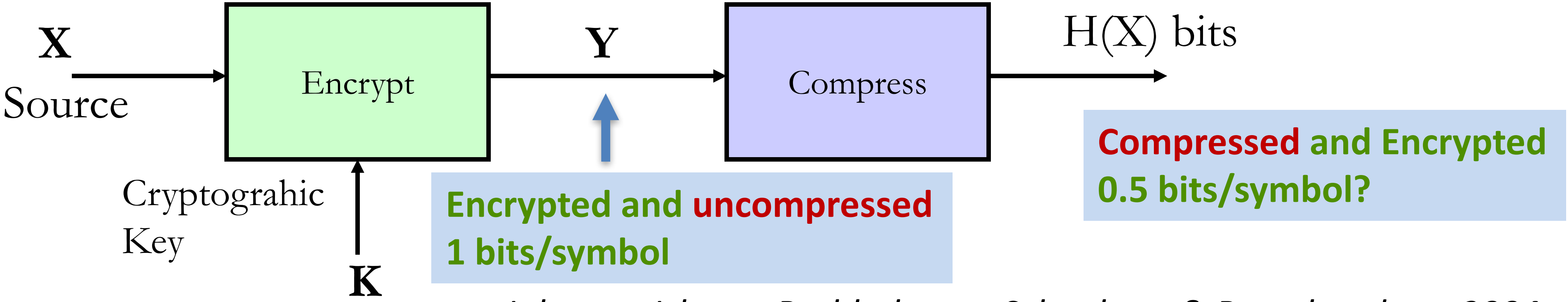
Compression of Encrypted Data

“Correct” order



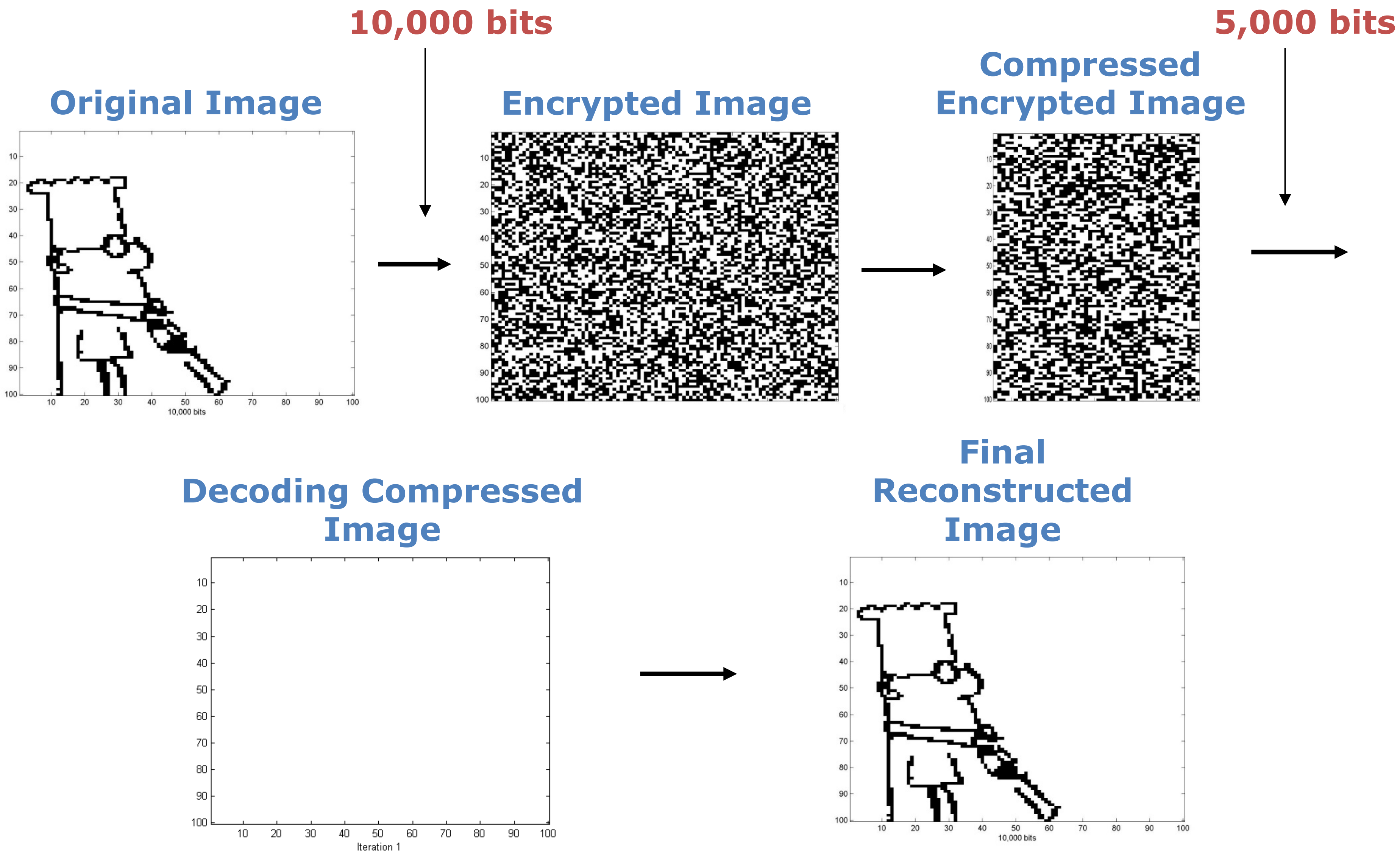
Source $\sim iid\ B(0.11)$

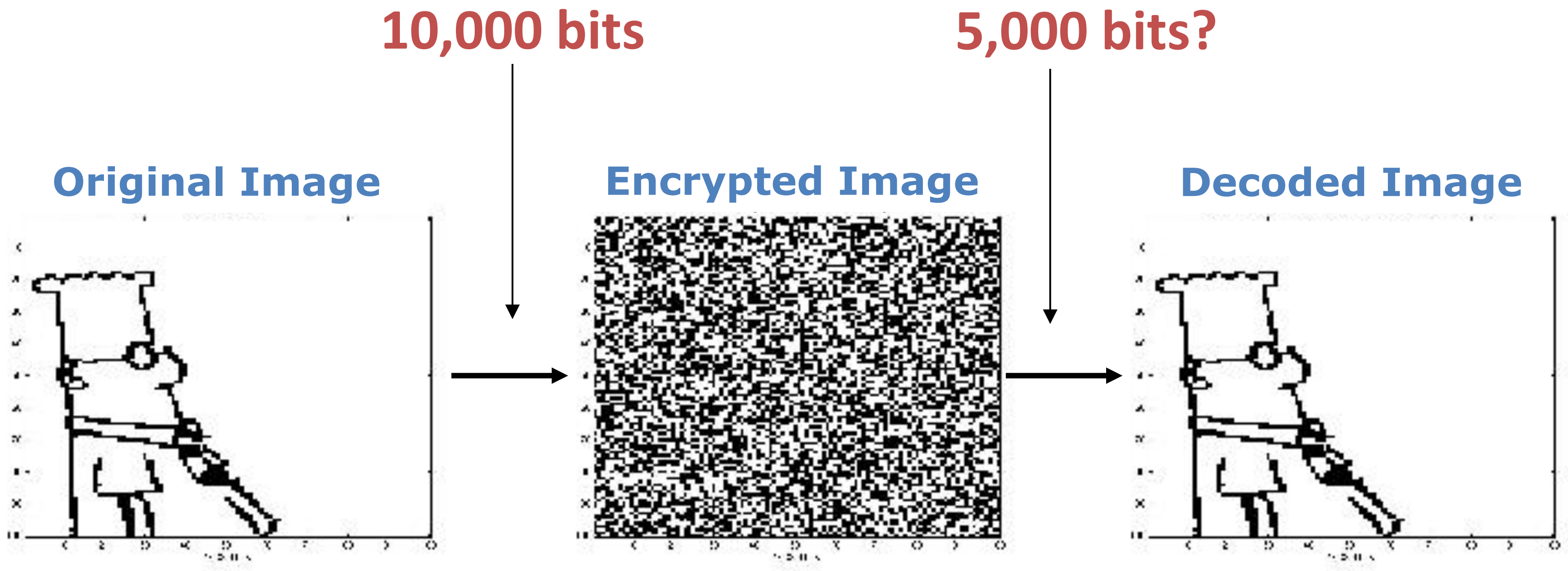
Wrong order?



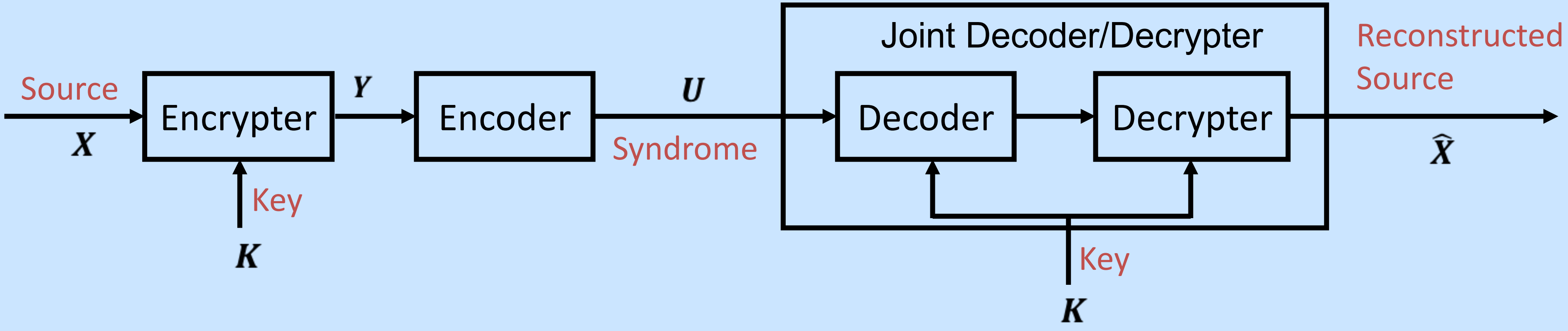
Johnson, Ishwar, Prabhakaran, Schonberg & Ramchandran, 2004

Example


10,000 bits
5,000 bits
Original Image
Encrypted Image
Compressed
Encrypted Image
10,000 bits
Decoding Compressed
Image
Iteration 1
Final
Reconstructed
Image
10,000 bits



Key Insight!

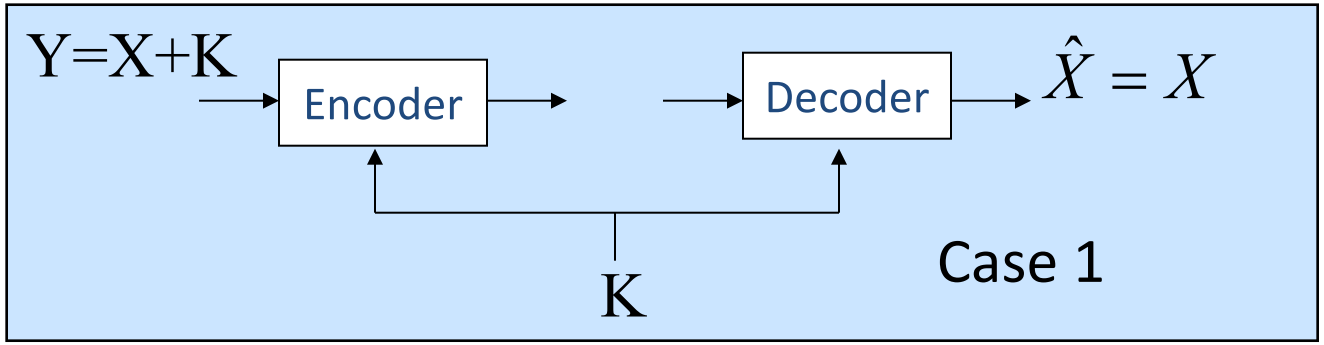


- Y = X + K where X is independent of K
- **Slepian-Wolf theorem**:
  can send X at rate H(Y|K) = H(X)

SCSI: binary example of noiseless compression

(Slepian-Wolf ’73)

- **X** is uniformly chosen from {[000], [001], [010], [100]}
- **K** is a length-3 random key (equally likely in $\{0,1\}^3$)
- Correlation: Hamming distance between **Y** and **K** at most 1
- Example: when **K**=[0 1 0], **Y** => [0 1 0], [0 1 1], [0 0 0], [1 1 0]

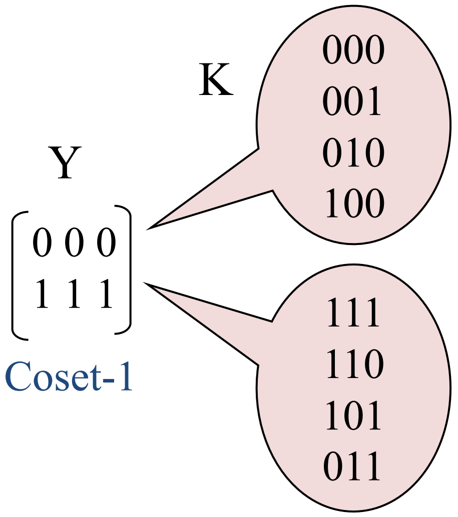


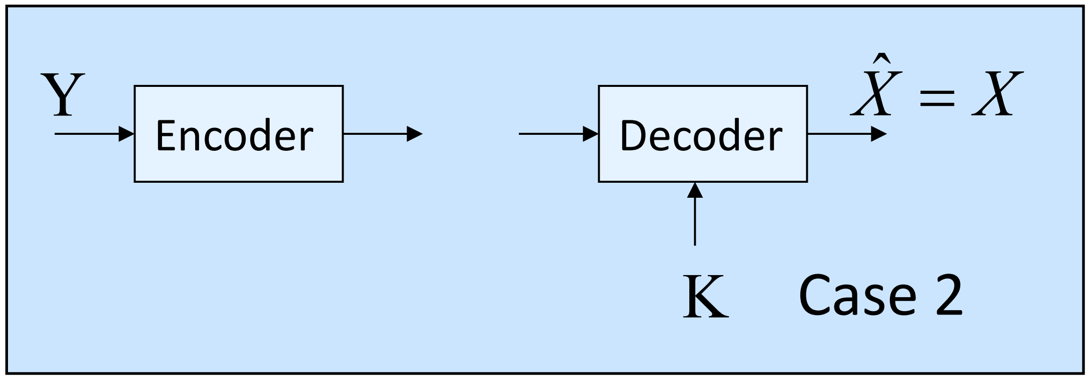
- **Encoder computes X=Y+K (mod 2)**
- **Encoder represents X using 2 bits**
- **Decoder outputs X (mod 2)**

00 → 000
01 → 001
10 → 010
11 → 100

=Y+K

(Slepian-Wolf ’73)



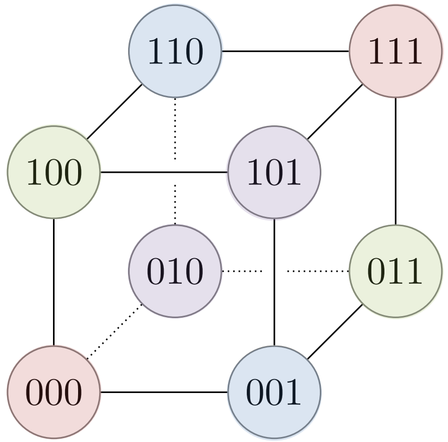


Coset-1 (00) $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

Coset-2 (01) $\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$

Coset-3 (10) $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$

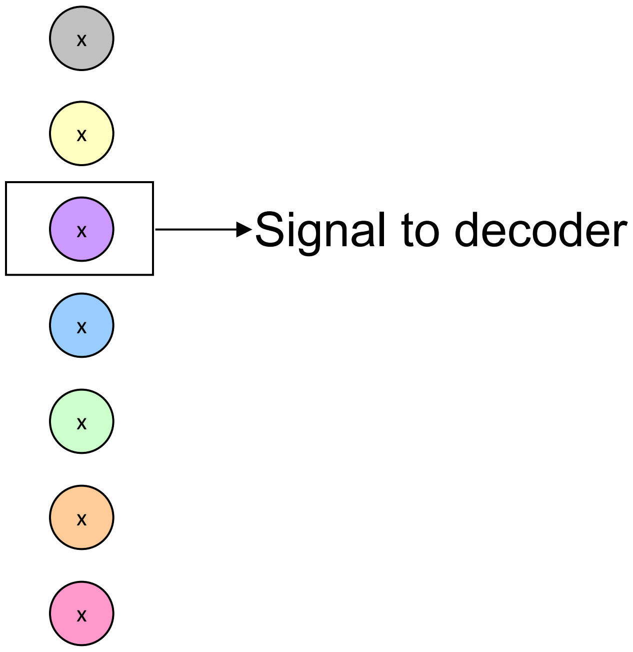
Coset-4 (11) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}$



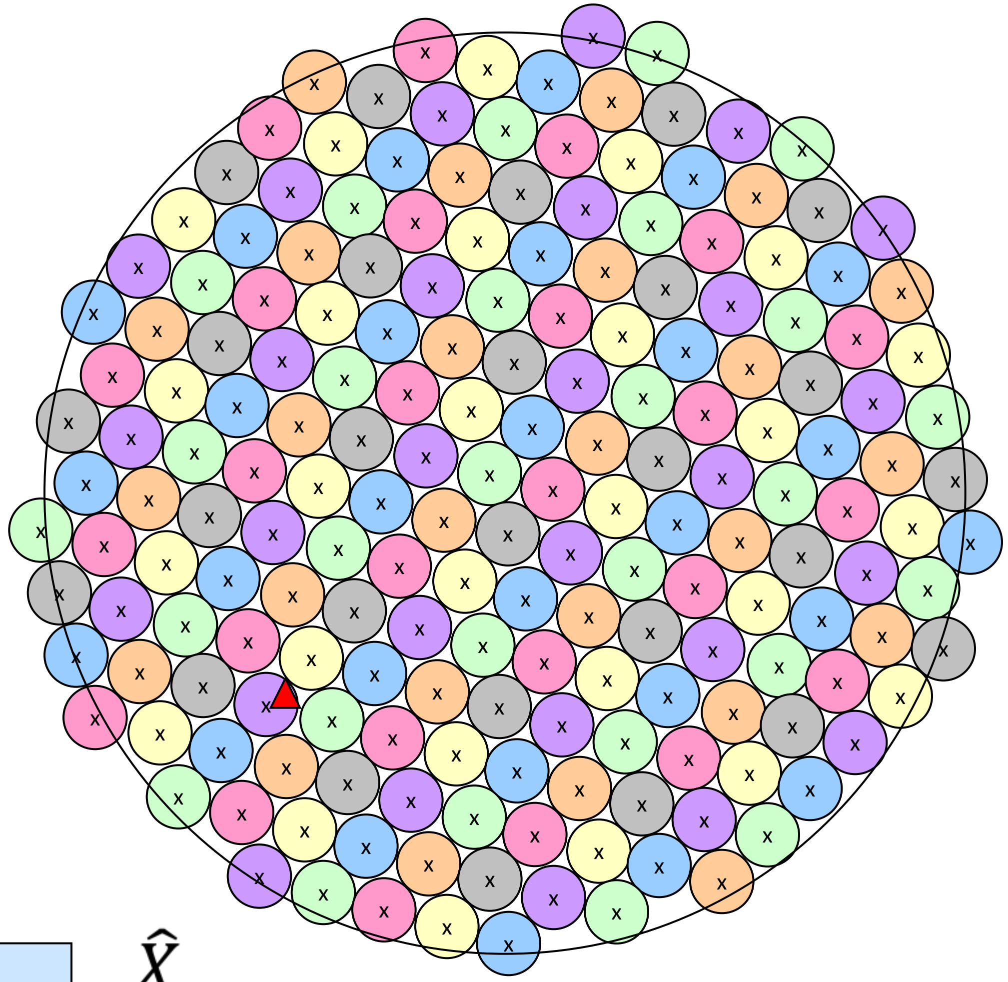
- Transmission at 2 bits/sample
- Encoder => send index of the coset containing X.
- Decoder => find a codeword in given coset closest to K

Example: Y=010 (K=110) => Encoder sends message 10

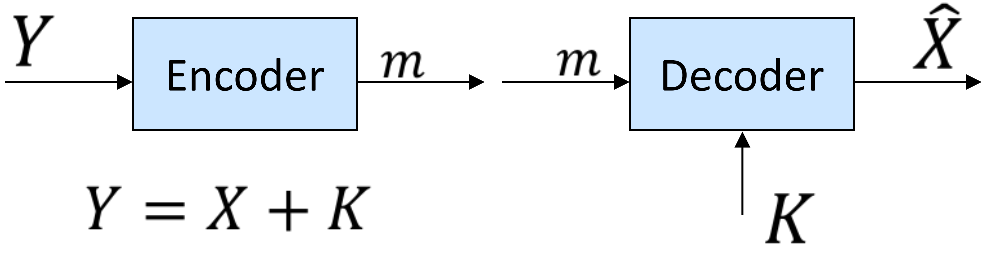
Geometric illustration



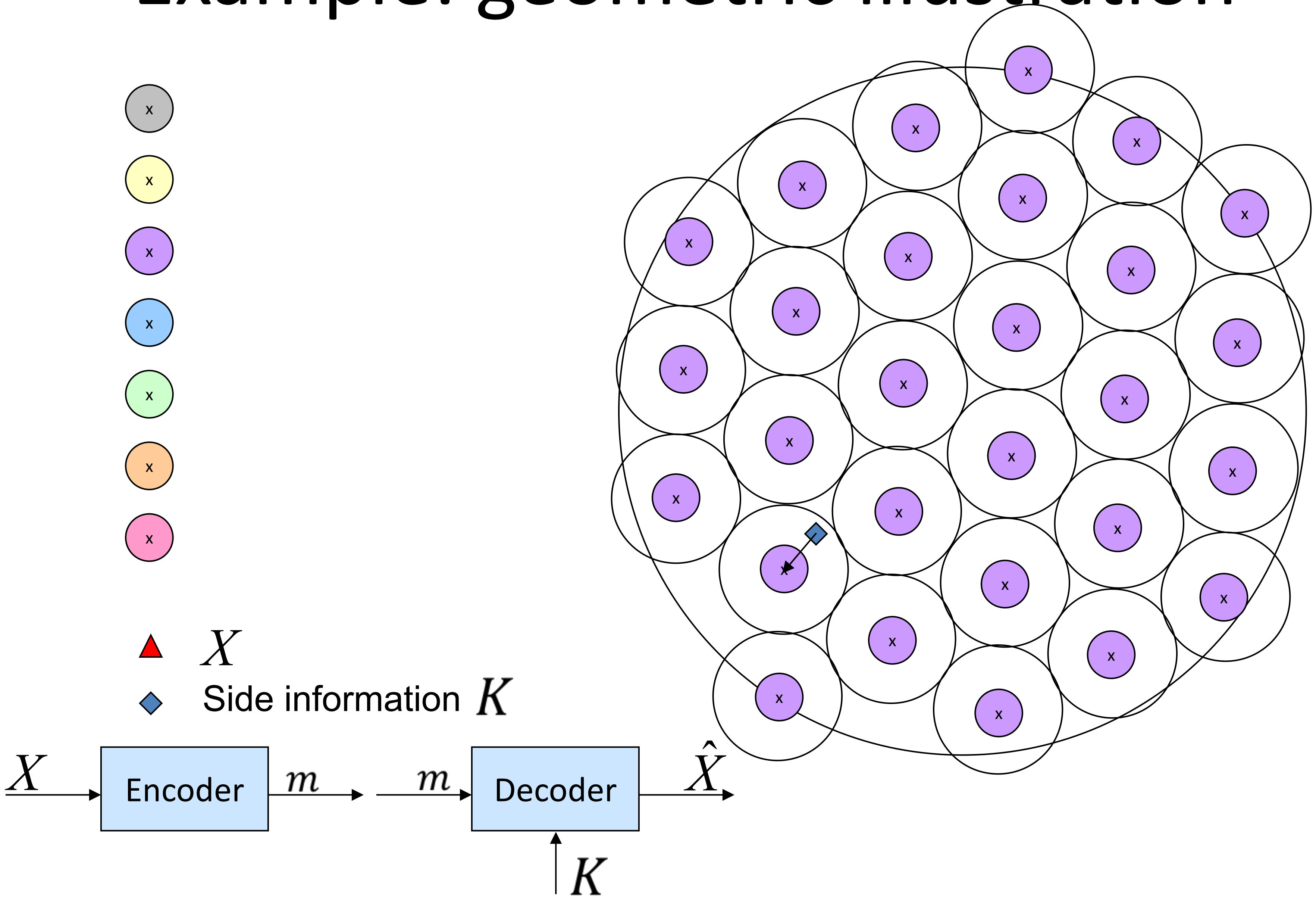
▲ Y (encrypted)



X (unencrypted & compressible)



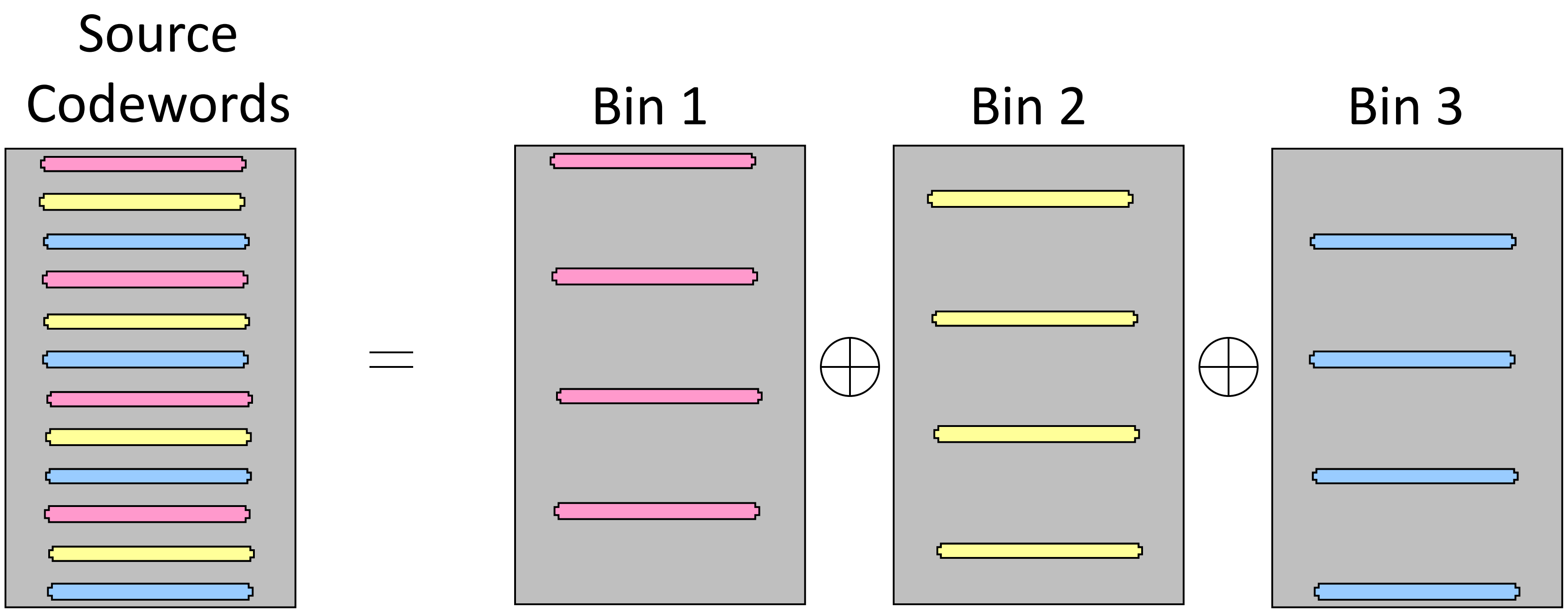
Example: geometric illustration



Practical Code Constructions

- Use a linear transformation (hash/bin)
- Design cosets to have maximal spacing
  - State of the art linear codes (LDPC codes)
- Distributed Source Coding Using Syndromes (DISCUS)*

*Pradhan & Ramchandran, '03



Orhan Ocal

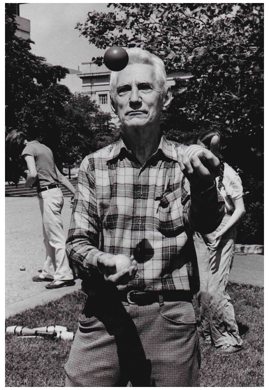
Xiao Li

Chapter 3

Sampling theory

- Sample and compute efficient sampling (and connections to learning)

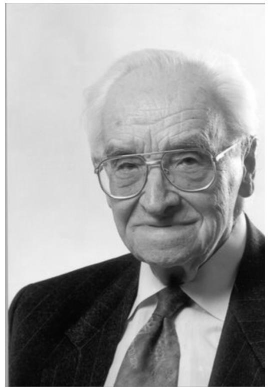
Sampling theorem



Shannon
1949

Nyquist
1928

Whittaker
1915



Kotelnikov
1933

Communication in the Presence of Noise

CLAUDE E. SHANNON, MEMBER, IRE

Theorem 1: If a function $f(t)$ contains no frequencies higher than W cps, it is completely determined by giving its ordinates at a series of points spaced 1/2 W seconds apart.

pointwise sampling!

...

Mathematically, this process can be described as follows. Let x_n be the nth sample. Then the function $f(t)$ is represented by

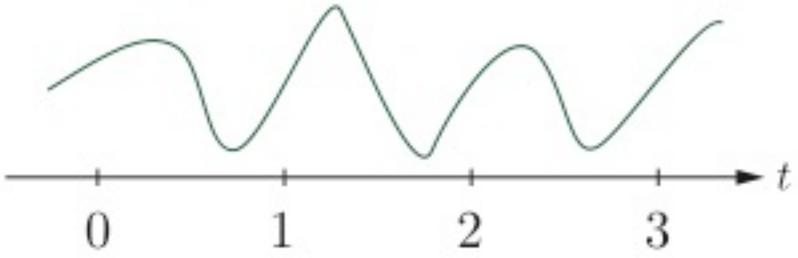
$$f(t) = \sum_{n=-\infty}^{\infty} x_n \frac{\sin \pi(2Wt - n)}{\pi(2Wt - n)}. \qquad (7)$$

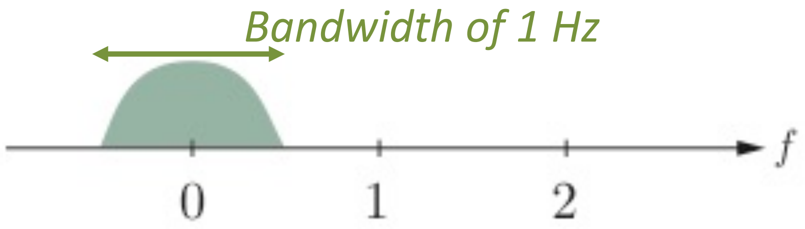
linear interpolation!

Aliasing phenomenon

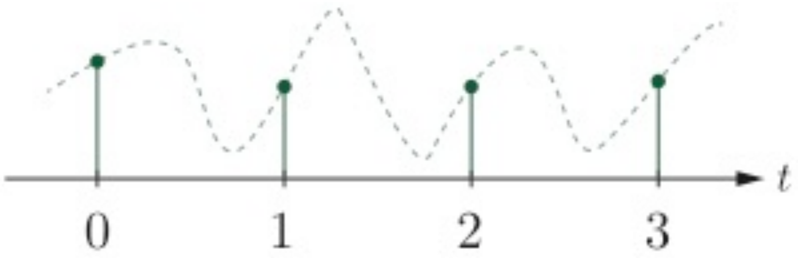
Time domain | **Frequency domain**

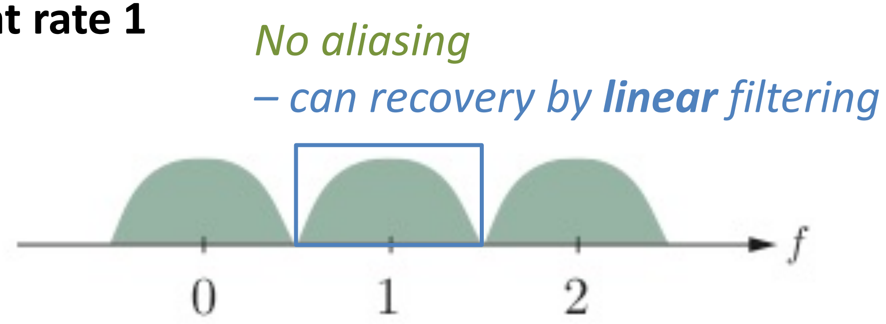
Input signal



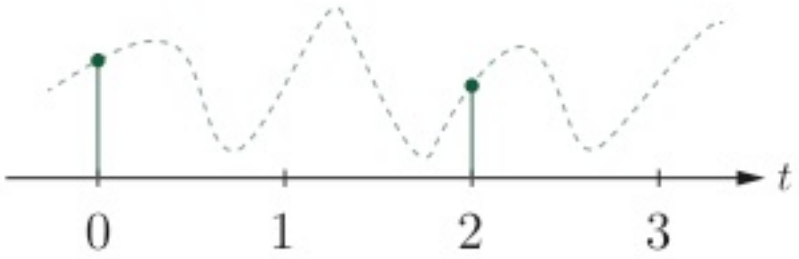


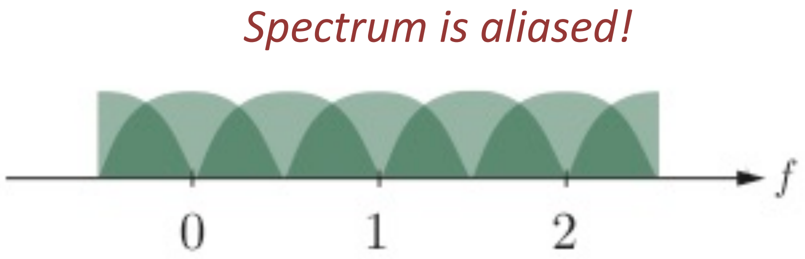
Sampling at rate 1



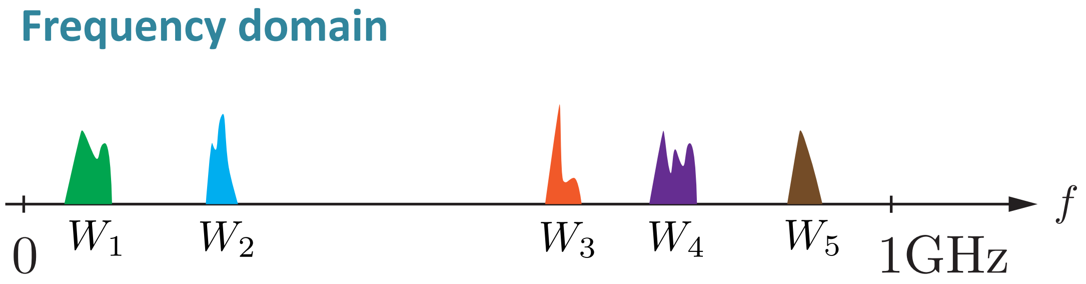


Sampling at rate 1/2





But what if the spectrum is sparsely occupied?



$$f_{occ} = \sum_{i=1}^{5} W_i = 100\text{MHz}$$

Henry Landau, 1967

– Know the frequency support

– Sample at rate *"occupied bandwidth"* f_{occ} ***(Landau rate)***

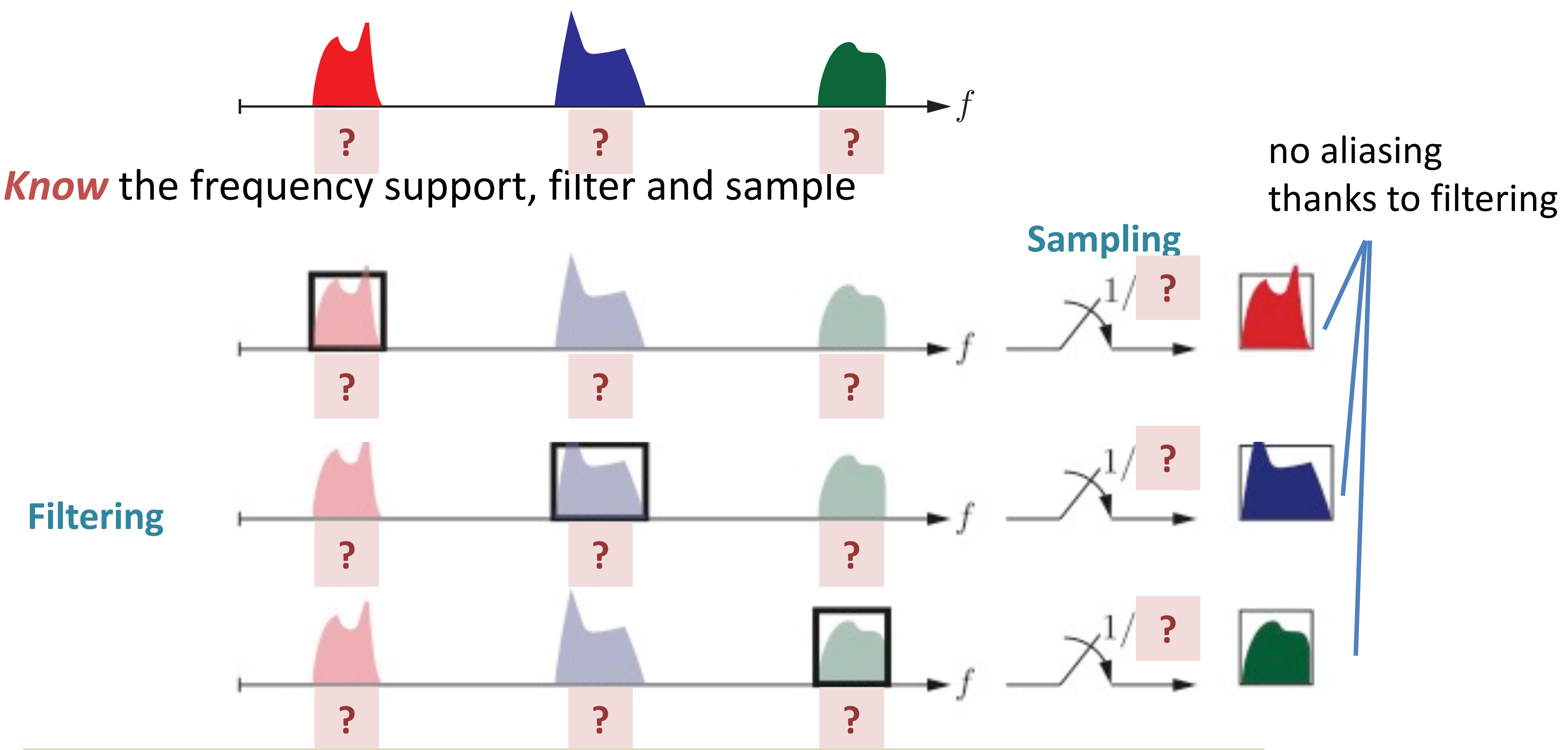
When you do not know the support?

- *Feng and Bresler, 1996*
- *Lu and Do, 2008*
- *Mishali, Eldar, Dounaevsky and Shoshan, 2011*
- *Lim and Franceschetti, 2017*

Filter bank approach

Input in frequency domain

Know the frequency support, filter and sample



Sampling ***spectrum-blind?***

Requires $2f_{occ}$. ***Can we design a constructive scheme?***

Lu and Do, 2008

Puzzle: Gold thief

100 grams each

- One unknown thief
- Steals unknown but fixed amount from each coin
- What is min. no. of weighings needed ?
  - 2 are enough!

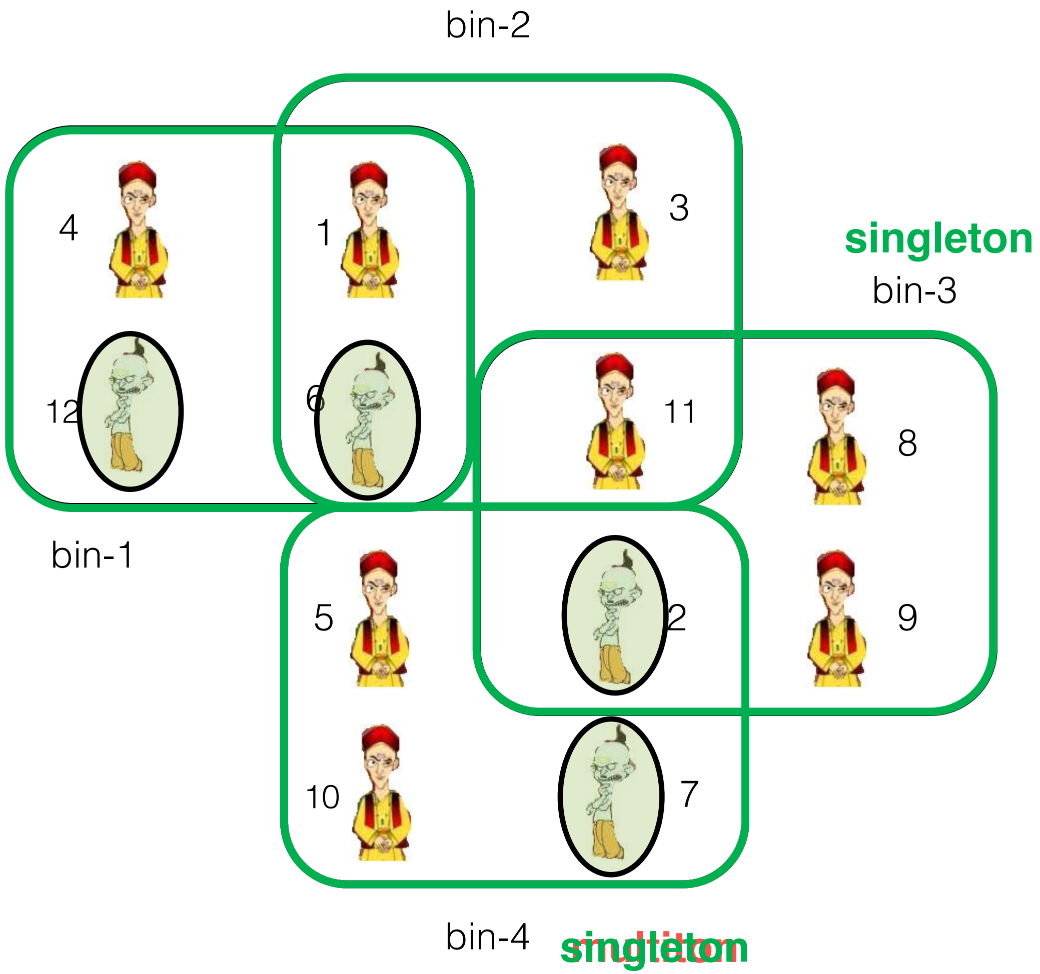
$$\frac{y2}{y1} \quad \boxed{4}$$

Differential weight

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{bmatrix} = \begin{bmatrix} -5 \\ -20 \end{bmatrix} = \begin{bmatrix} y1 \\ y2 \end{bmatrix}$$

Ratio-test identifies the location

4-thieves among 12-treasurers



Key Ideas:

1. **Randomly group the treasurers.**
2. **If there is a single thief problem**
   - ✓ **Ratio test**
   - ✓ **Iterate.**

Questions:

1. **How many groups needed?**
2. **How to form groups?**
3. **How to identify if a group has a single thief?**

Main result

Any bandlimited signal $x(t) \in \mathbb{C}$ whose spectrum has occupancy f_{occ} can be sampled asymptotically at rate $f_s = 2f_{occ}$ by a randomized "*sparse-graph-coded filter bank*" with probability 1 using $O(f_{occ})$ operations per unit time.

Remarks

- Computational cost $O(f_{occ})$ ***independent of bandwidth***
- Requires mild assumptions (genericity)
- Can be made robust to sampling noise

Ocal, Li & Ramchandran, 2016

Key insight for spectrum-blind sampling

- To reduce sampling rate, ***subsample judiciously***

subsampling $\Rightarrow$ aliasing

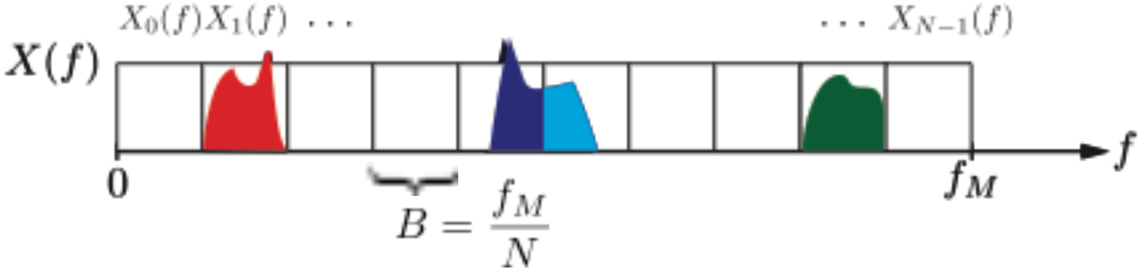
“judicious” filtering/subsampling $\Rightarrow$ “good” aliasing

- Introduces aliasing (***structured noise***)

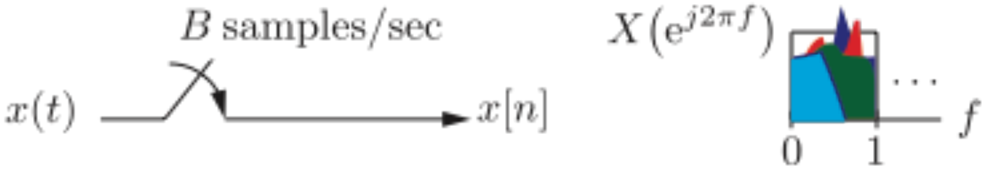
- ***Filter bank*** derived from ***capacity-achieving codes for the*** BEC: (irregular LDPC codes)

- ***Non-linear recovery*** instead of linear interpolation

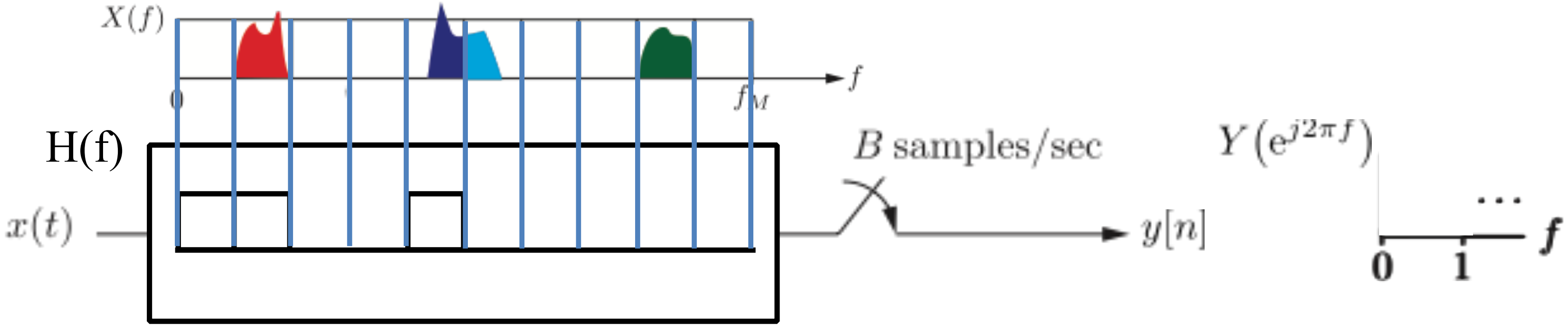
Filter bank for sampling



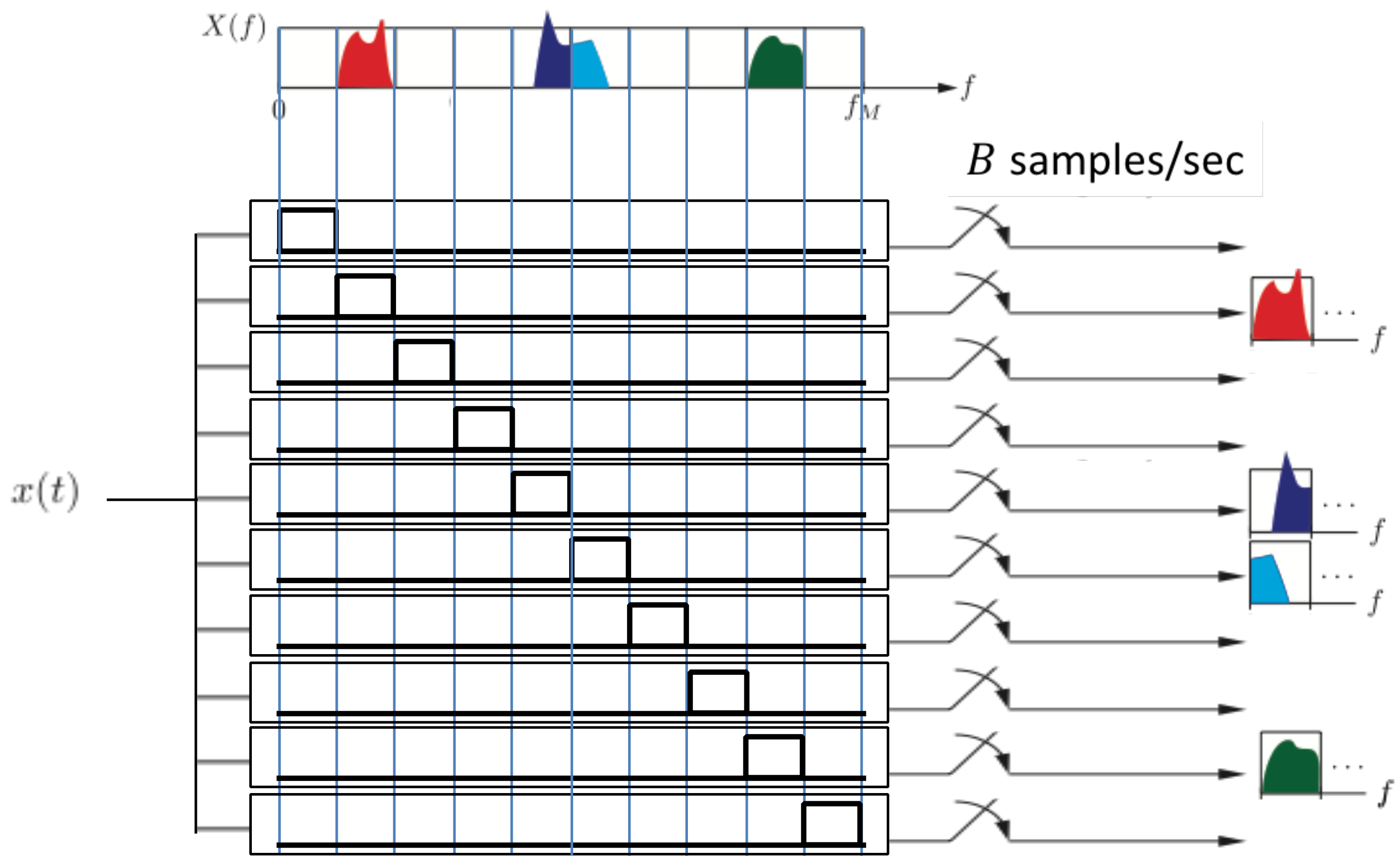
- Sample the signal at rate B



- **Filter and then sample at rate B**



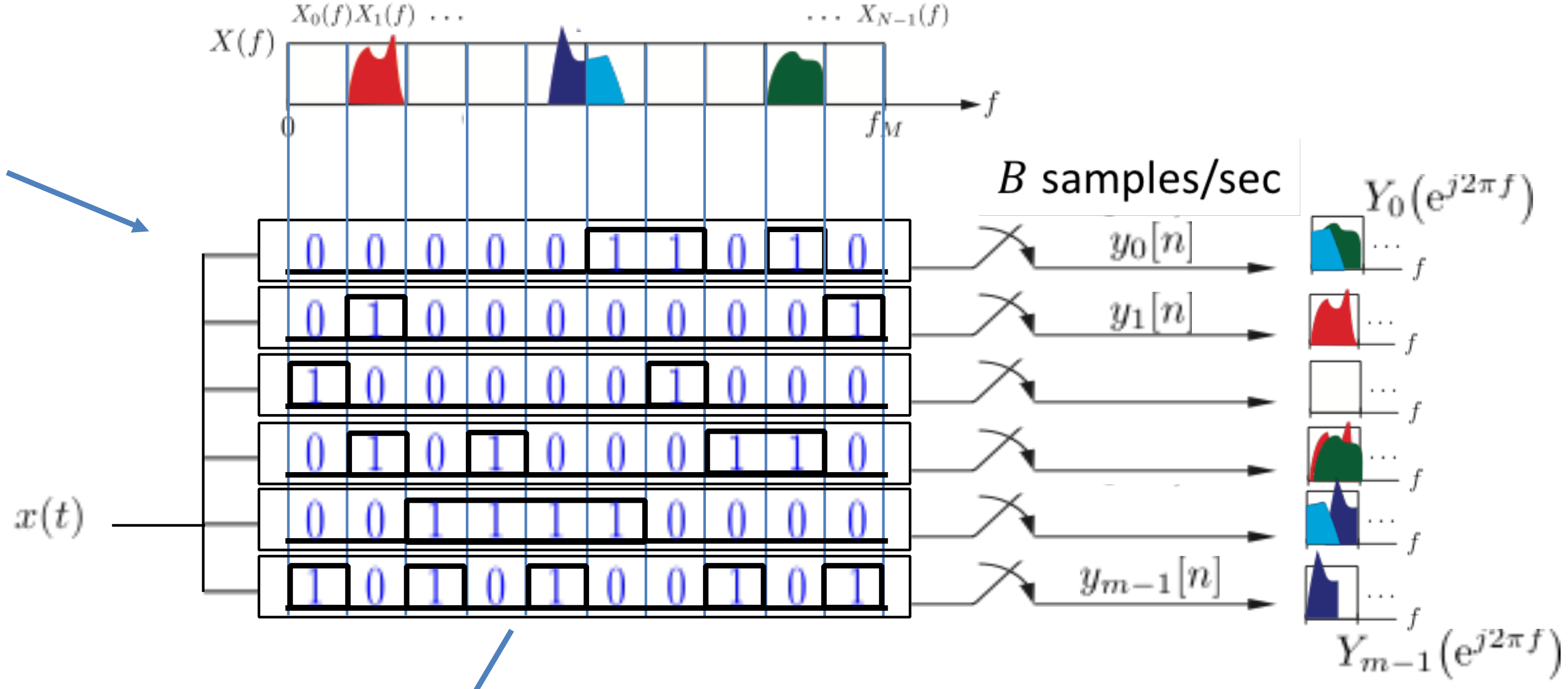
Filter bank for sampling



Aggregate sampling rate: $N\frac{f_M}{N} = f_M$ = Nyquist rate for $x(t)$

‘*Sparse-graph-coded*’ filter bank

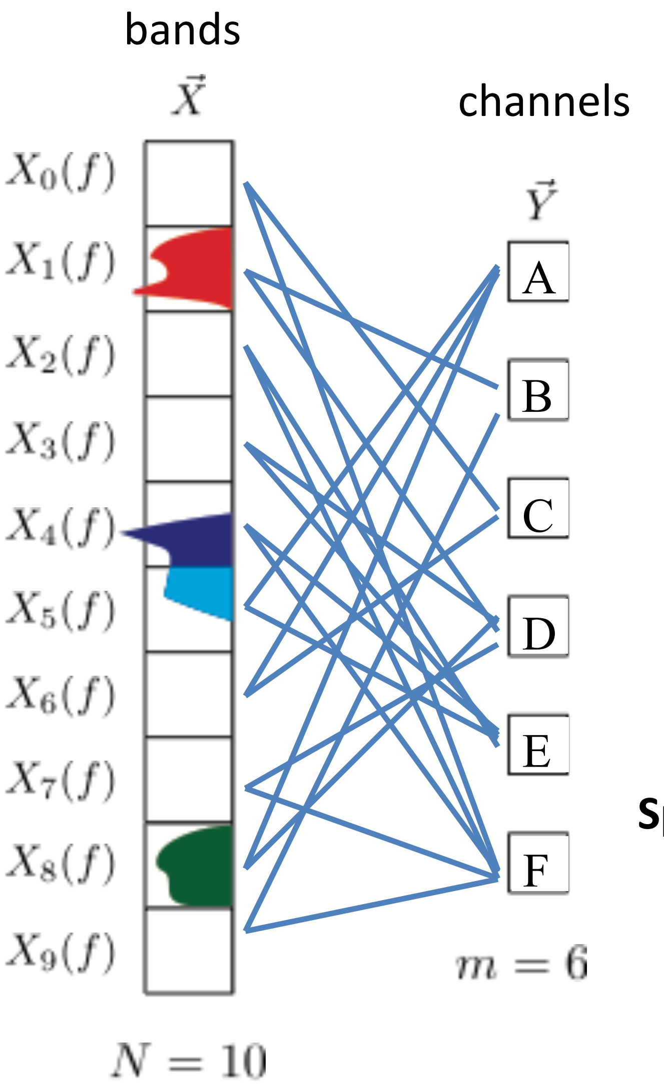
m filters
N bands



$$\vec{Y}(e^{j2\pi f}) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \vec{X}(Bf) \quad \text{where} \quad \vec{X}(f) = \begin{pmatrix} X_0(f) \\ \vdots \\ X_{n-1}(f) \end{pmatrix}$$

$m \times N$ matrix

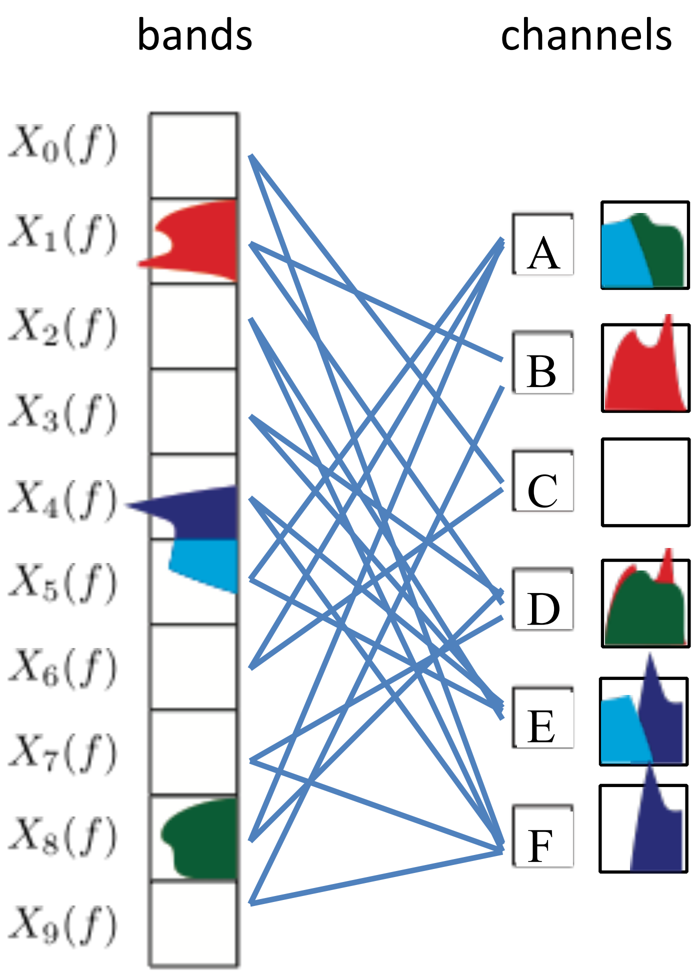
Example — sparse graph underlying the measurements



$$\vec{Y}(e^{j2\pi f}) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \vec{X}(Bf)$$

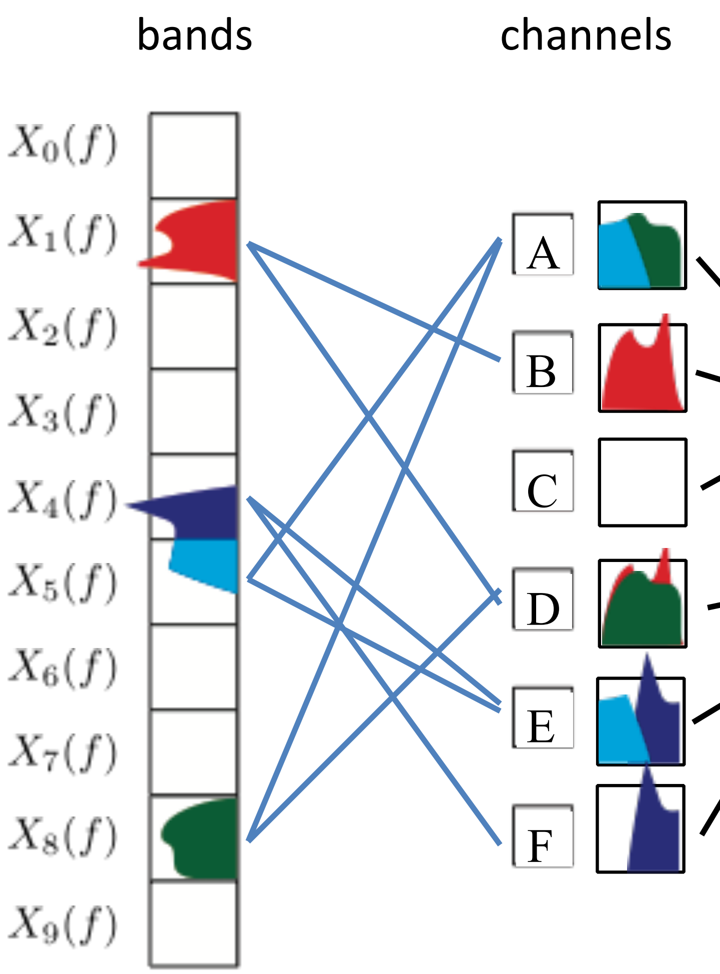
Sparse bipartite graph

Example — sparse graph underlying the measurements



visual cleaning for presentation:
remove edges that connect to non-active bands

Example — peeling



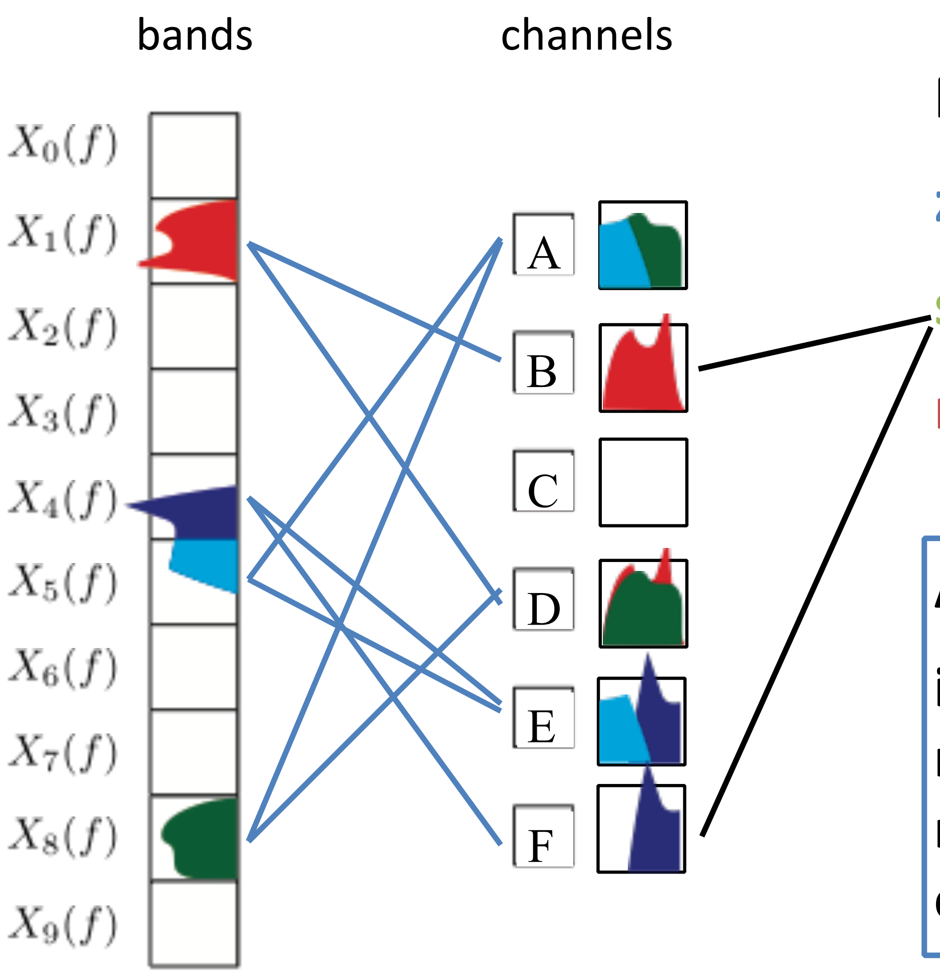
Measurement classification

zero-ton: **no signal**

single-ton: **no aliasing**

multi-ton: **aliasing**

Example — peeling



Measurement classification

zero-ton: **no signal**

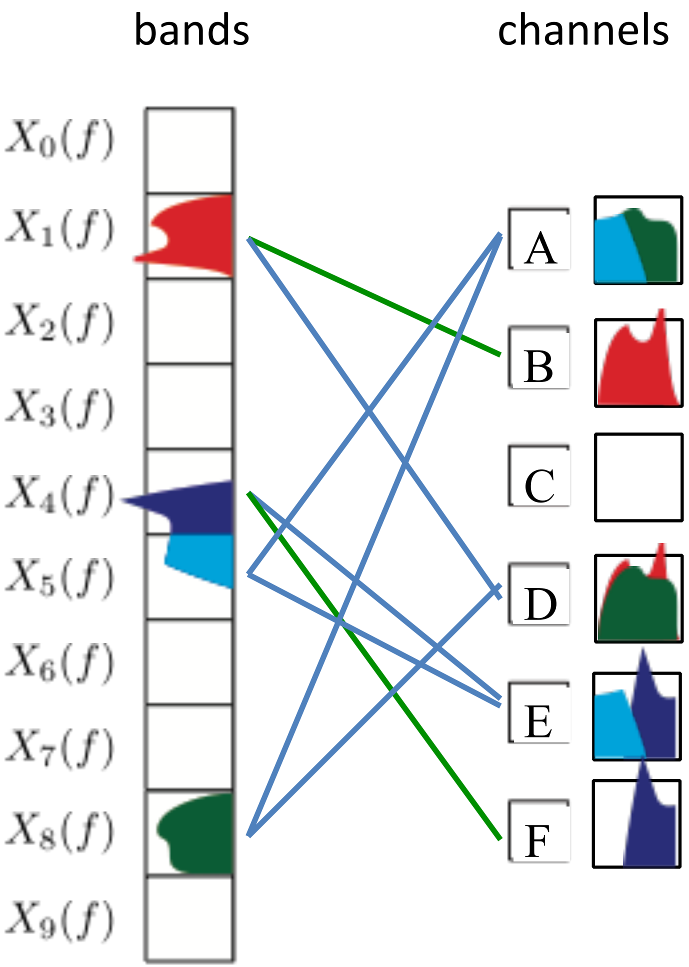
single-ton: **no aliasing**

multi-ton: **aliasing**

Assume a *mechanism*:

identifies which channels have no aliasing (here B and F) and maps them to which bands they came from (here 1 and 4 resp.)

Example — peeling



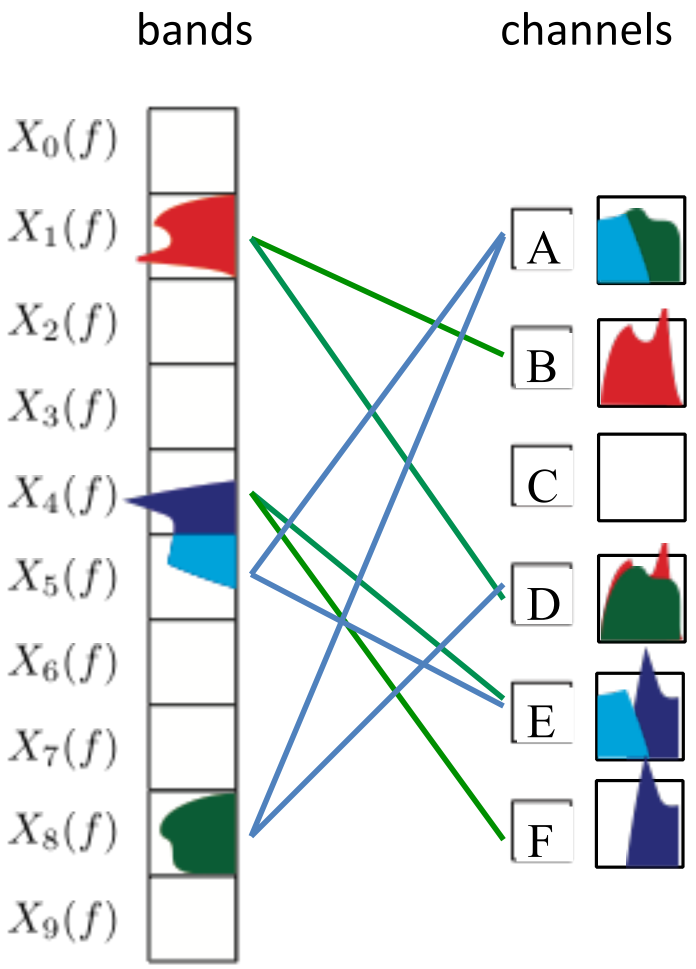
***mechanism*:**

identifies which channels have no aliasing and maps them to which bands they came from

output:

channel B: (red, index = 1)
channel F: (blue, index = 4)

Example — peeling



***mechanism*:**

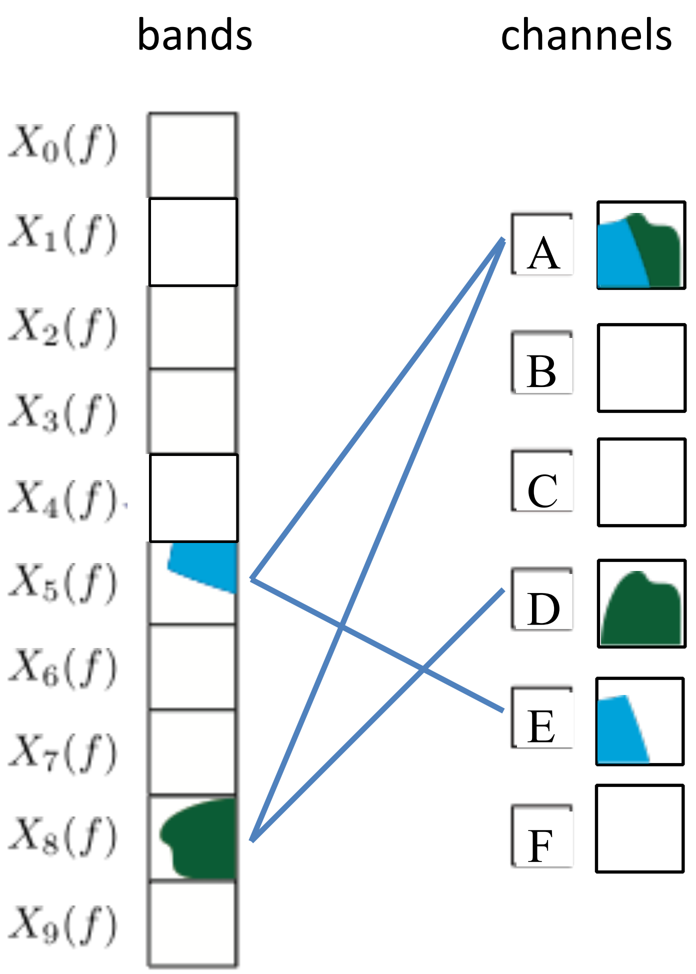
identifies which channels have no aliasing and maps them to which bands they came from

output:

channel B: (red, index = 1)
channel F: (blue, index = 4)

peel from channels they alias into!

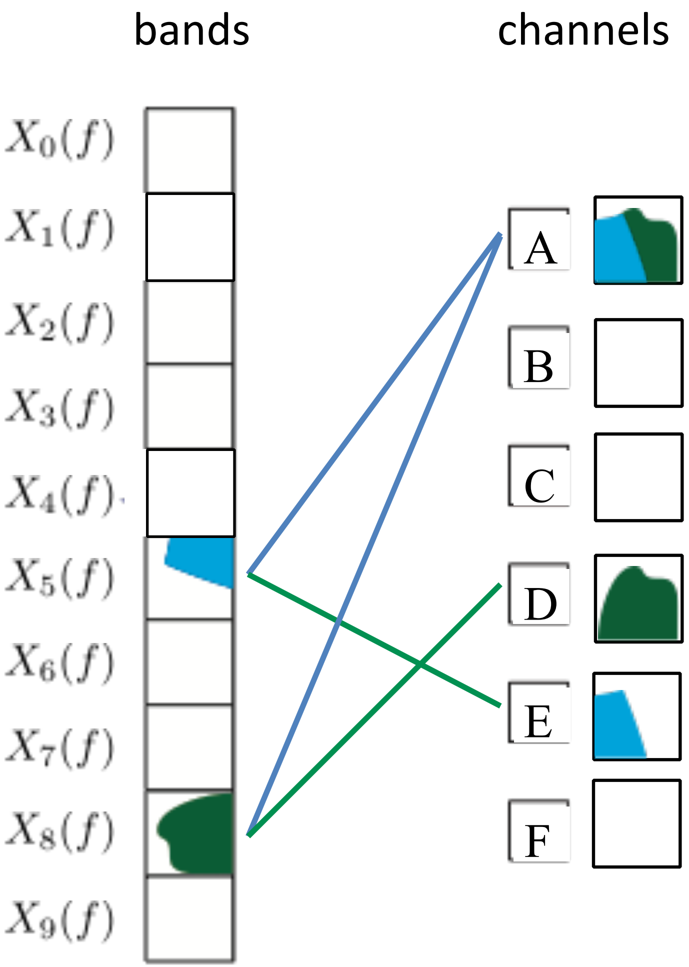
Example — peeling



***mechanism*:**

identifies which channels have no aliasing and maps them to which bands they came from

Example — peeling



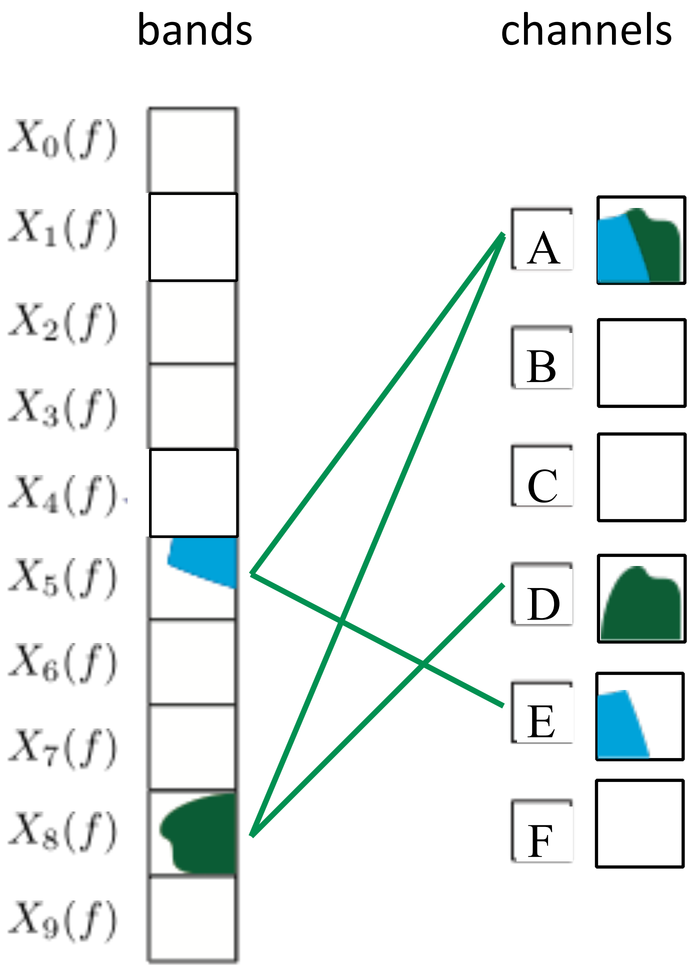
***mechanism*:**

identifies which channels have no aliasing and maps them to which bands they came from

output:

channel D: (green, index = 8)
channel E: (cyan, index = 5)

Example — peeling



***mechanism*:**

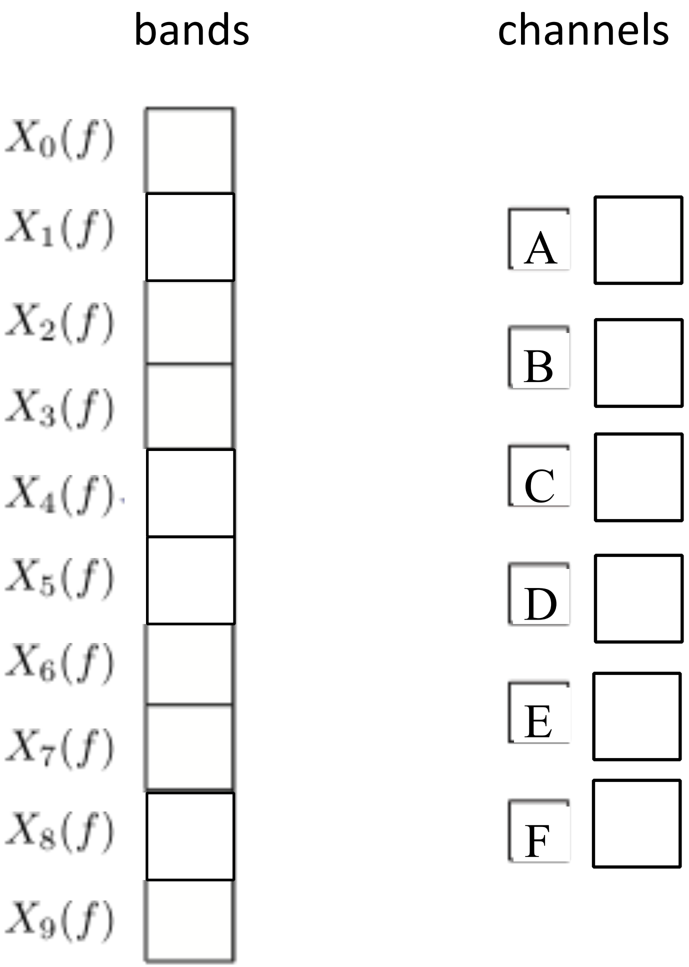
identifies which channels have no aliasing and maps them to which bands they came from

output:

channel D: (green, index = 8)
channel E: (cyan, index = 5)

peel from channels they alias into!

Example — peeling



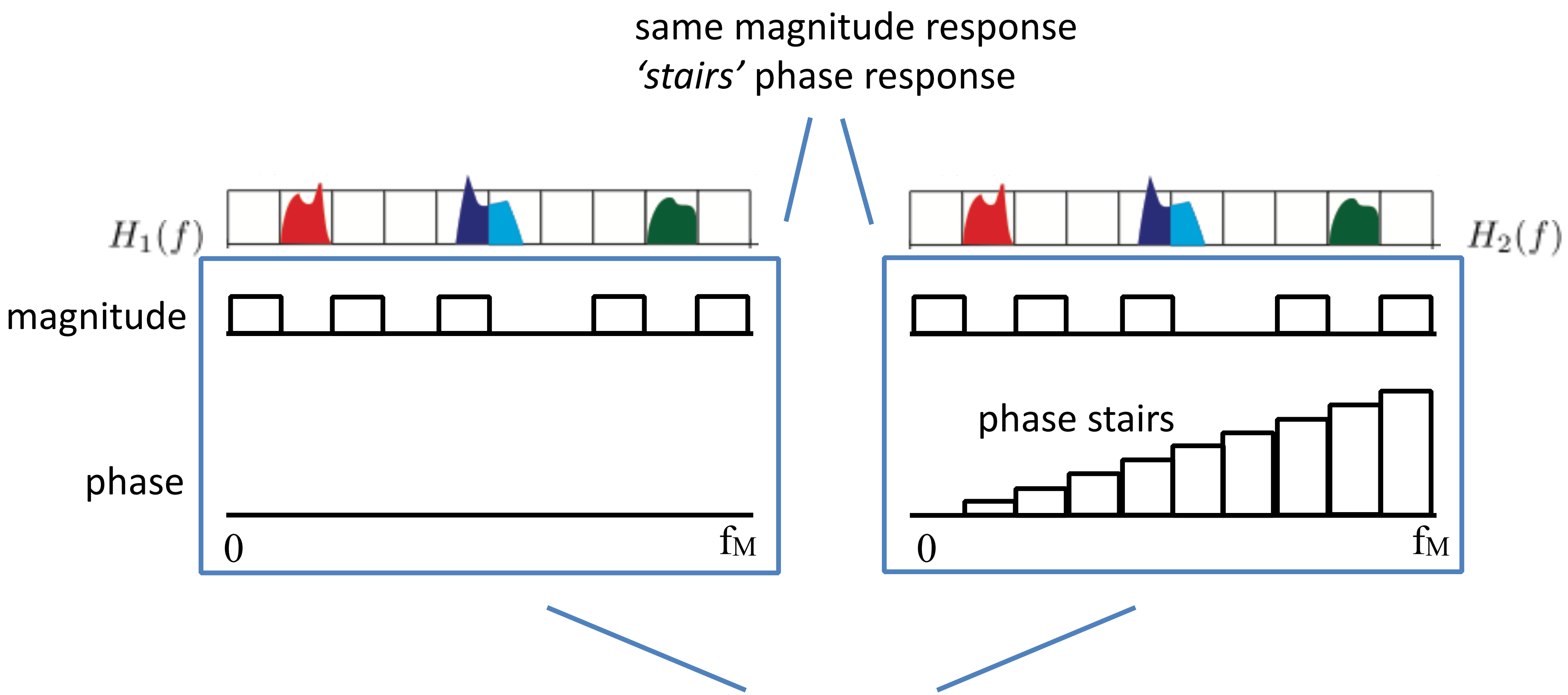
mechanism:

identifies which channels have no aliasing and maps them to which bands they came from

signal is completely recovered!

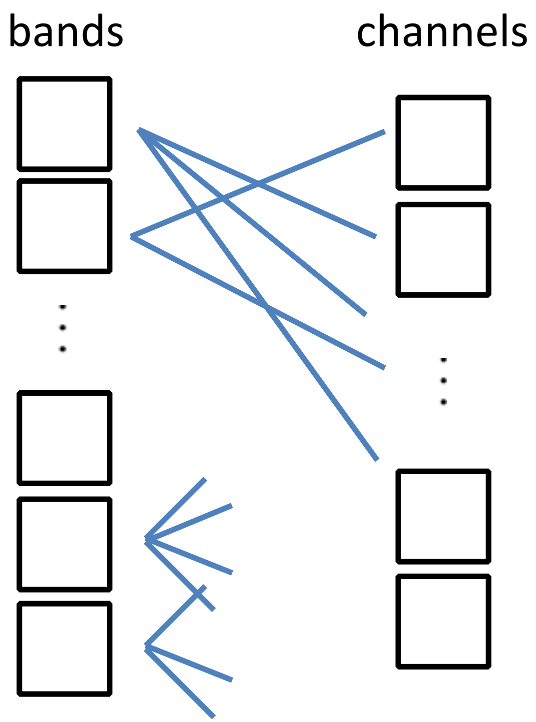
Realizing the *mechanism*

Identify which channels have no aliasing and map them to bands



identifies dark blue band as a singleton

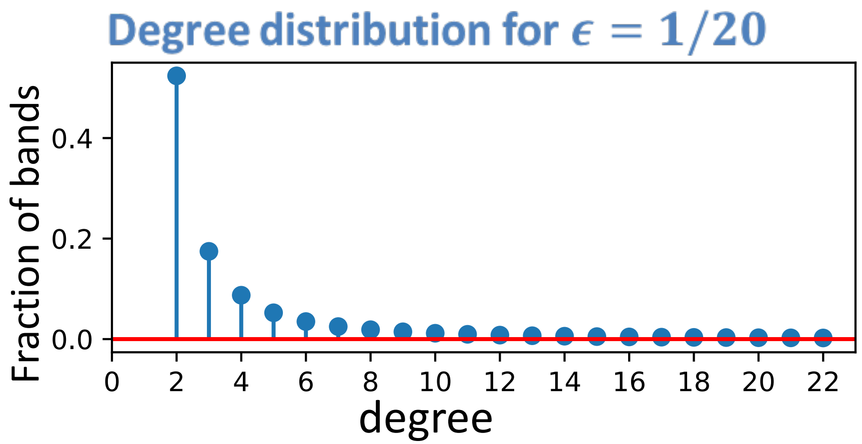
Construction of the sparse-graph code



- Designed through ***capacity-approaching sparse-graph codes***
- Connect each ***band*** to ***channels*** at random according to a carefully chosen degree distribution.
- Asymptotically, ***number of channels*** is $(1+\epsilon)$ times the ***number of active bands***

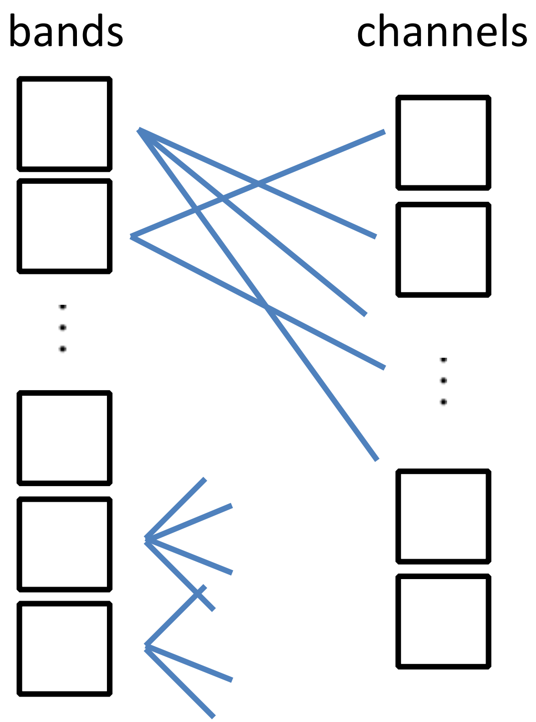
$$P(\text{degree} = j) \propto \frac{1}{j(j-1)}, \text{ for } j = 2, \ldots, D+1$$

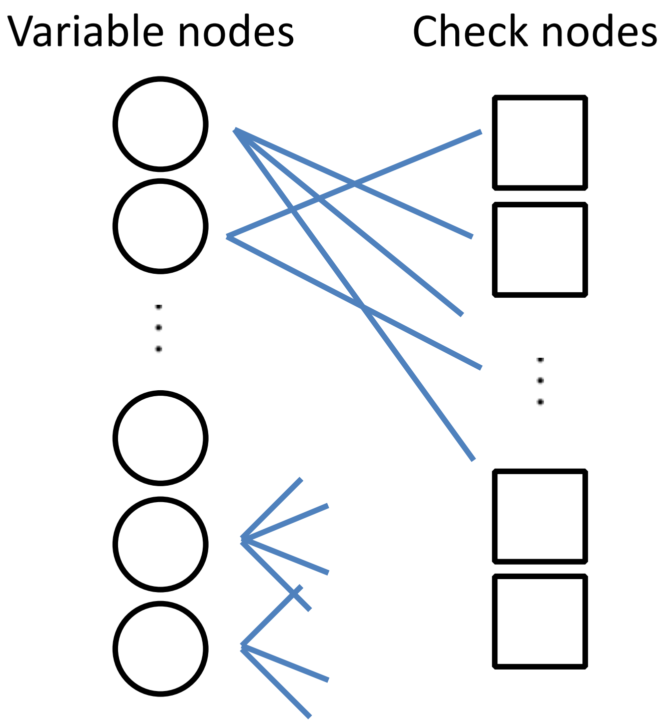
$$D > 1/\epsilon$$



Luby et al. 2001

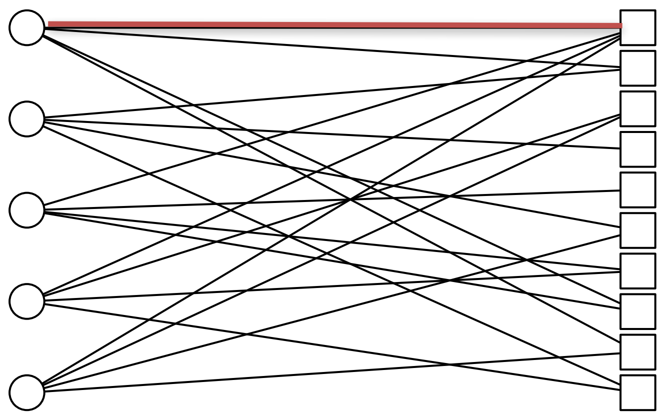
Construction of the sparse-graph code





Regular graph construction:
Connect every variable node to d check nodes chosen uniformly at random

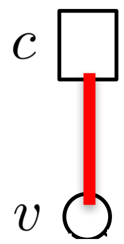
Density evolution

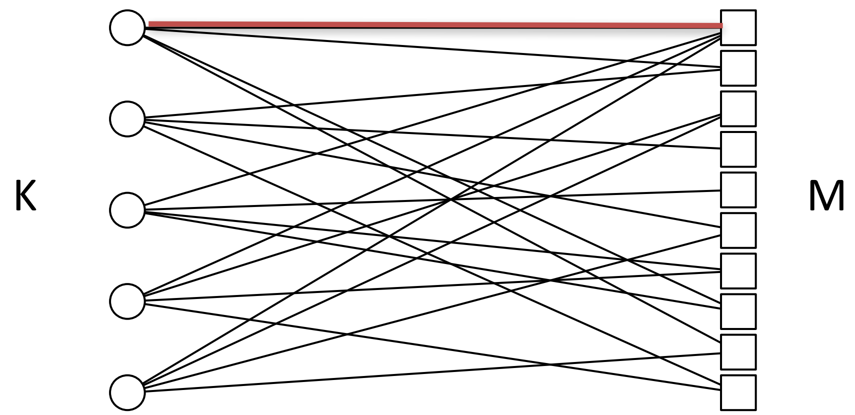


$\text{example: } d = 4$

Regular graph construction:
Connect every variable node to d check nodes chosen uniformly at random

Density evolution



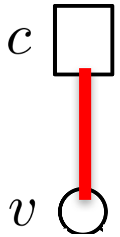


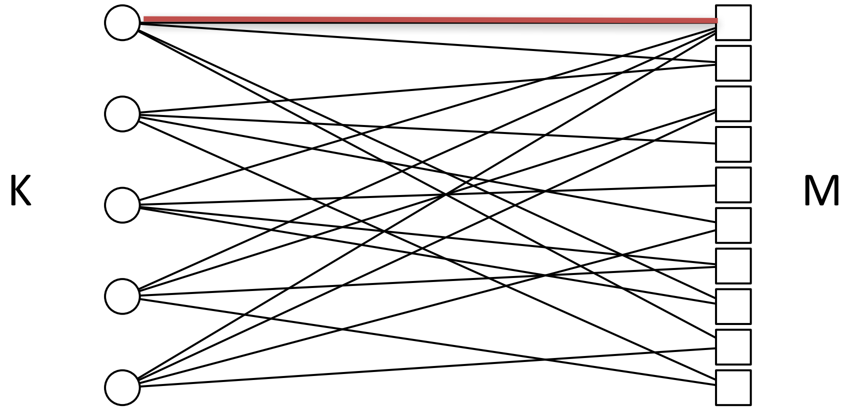
example: $d = 4$

- Pick an arbitrary edge in the graph (c, v).

Regular graph construction:
Connect every variable node to d check nodes chosen uniformly at random

Density evolution

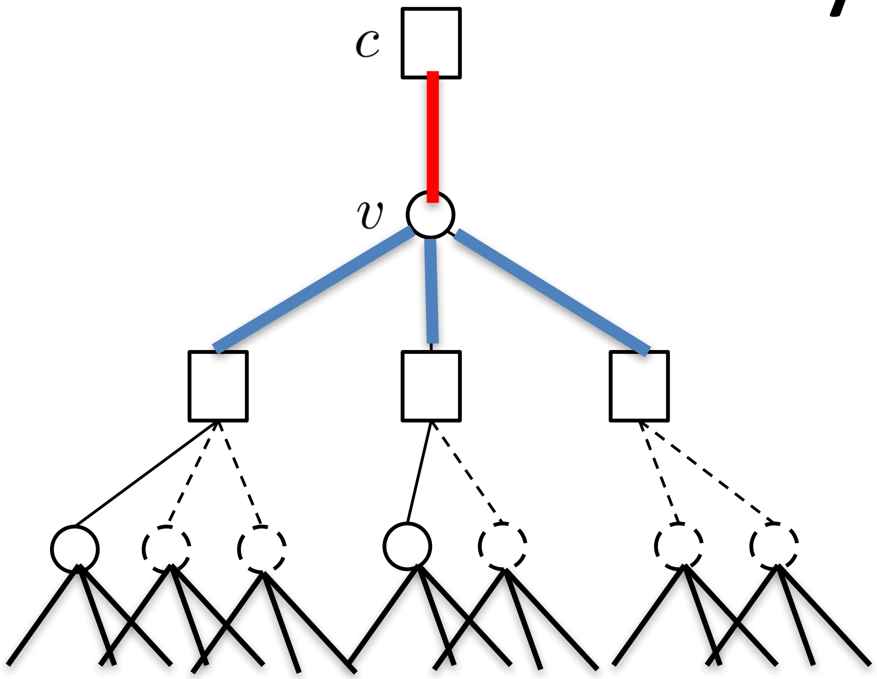


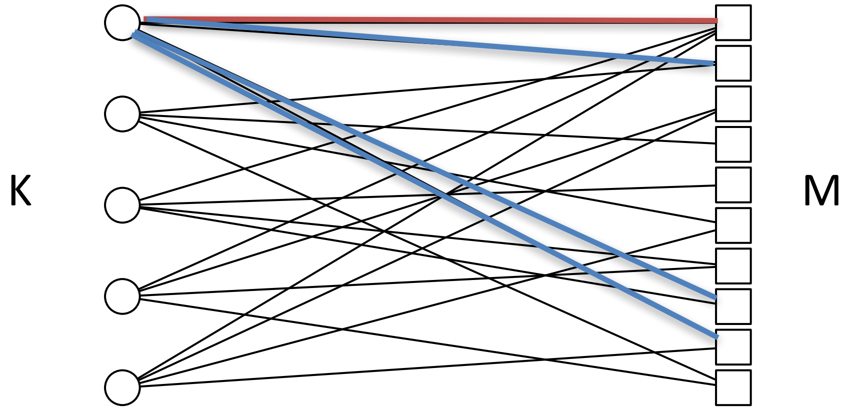


example: $d = 4$

- Examine its directed neighborhood at depth-2ℓ

Density evolution

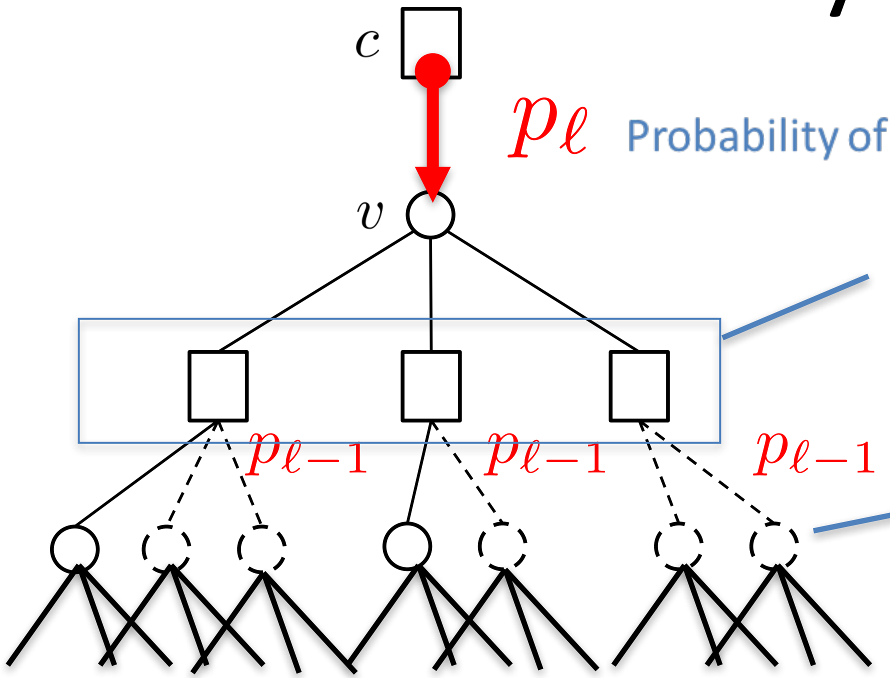




example: $d = 4$

- Examine its directed neighborhood at depth-2ℓ

Density evolution

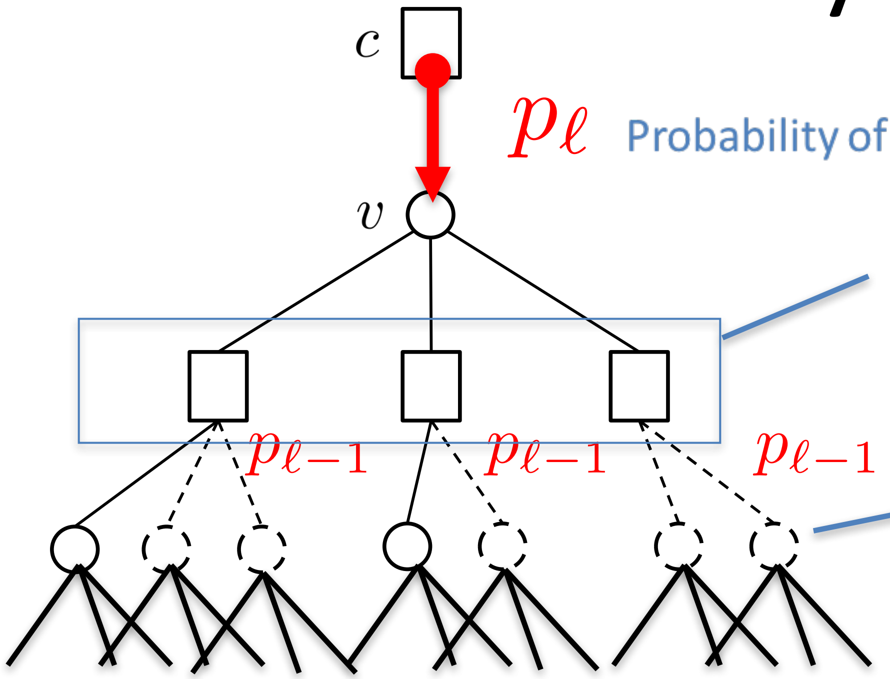


Probability of being present at depth 2ℓ

The variable node v can be resolved if ***any*** of these check nodes can be resolved

A check node is resolved from below if ***all*** of the variable nodes connected to it from below are resolved

Density evolution



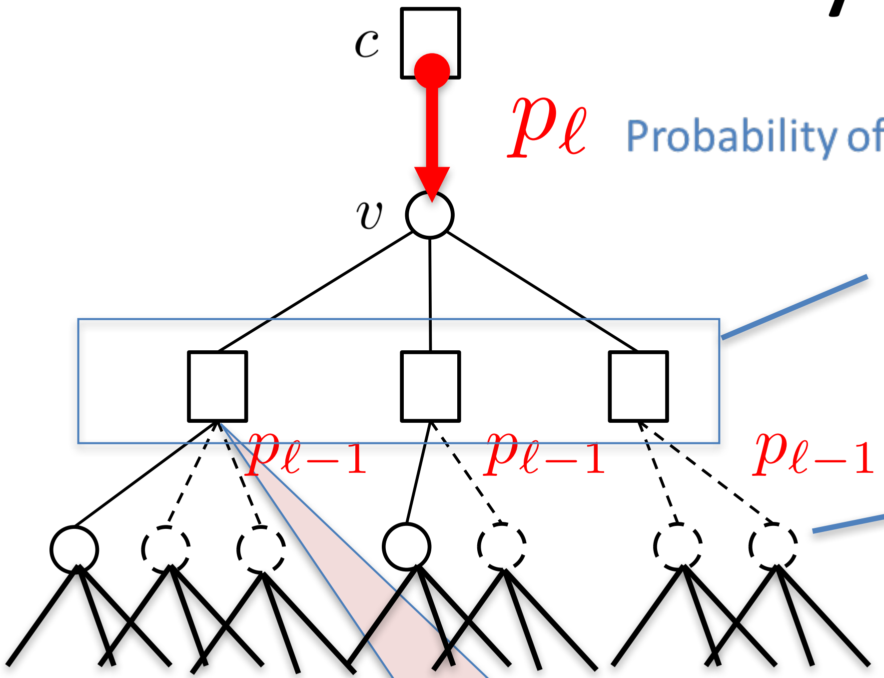
Probability of being present at depth 2ℓ

The variable node v can be resolved if ***any*** of these check nodes can be resolved

A check node is resolved from below if ***all*** of the variable nodes connected to it from below are resolved

$p_\ell =$

Density evolution



Probability of being present at depth 2ℓ

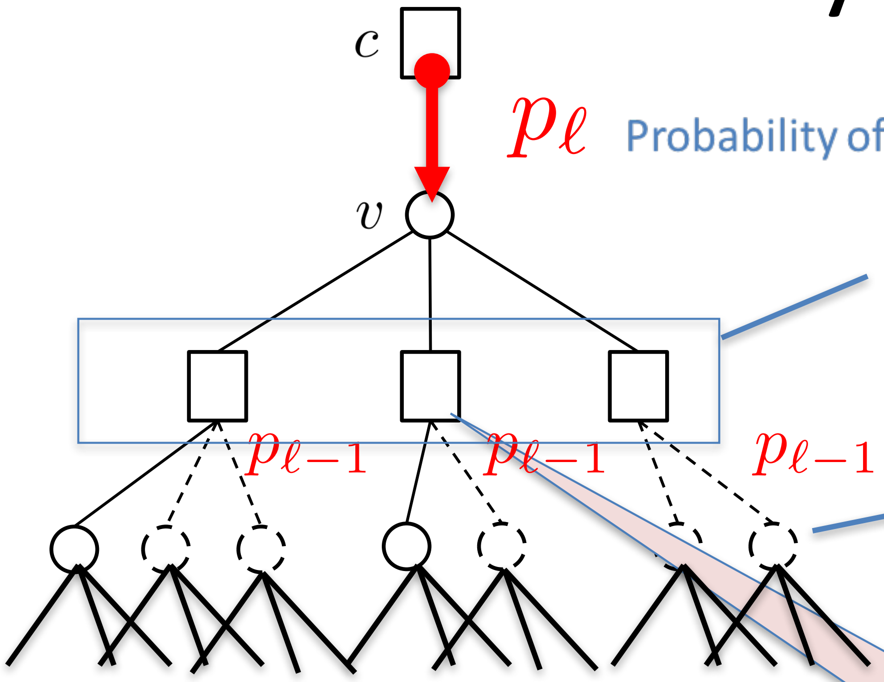
The variable node v can be resolved if ***any*** of these check nodes can be resolved

A check node is resolved from below if ***all*** of the variable nodes connected to it from below are resolved

$$p_\ell = \boxed{\left[1-(1-p_{\ell-1})^3\right]} \times$$

Power 3 is because the check node has 3 variable nodes as children

Density evolution



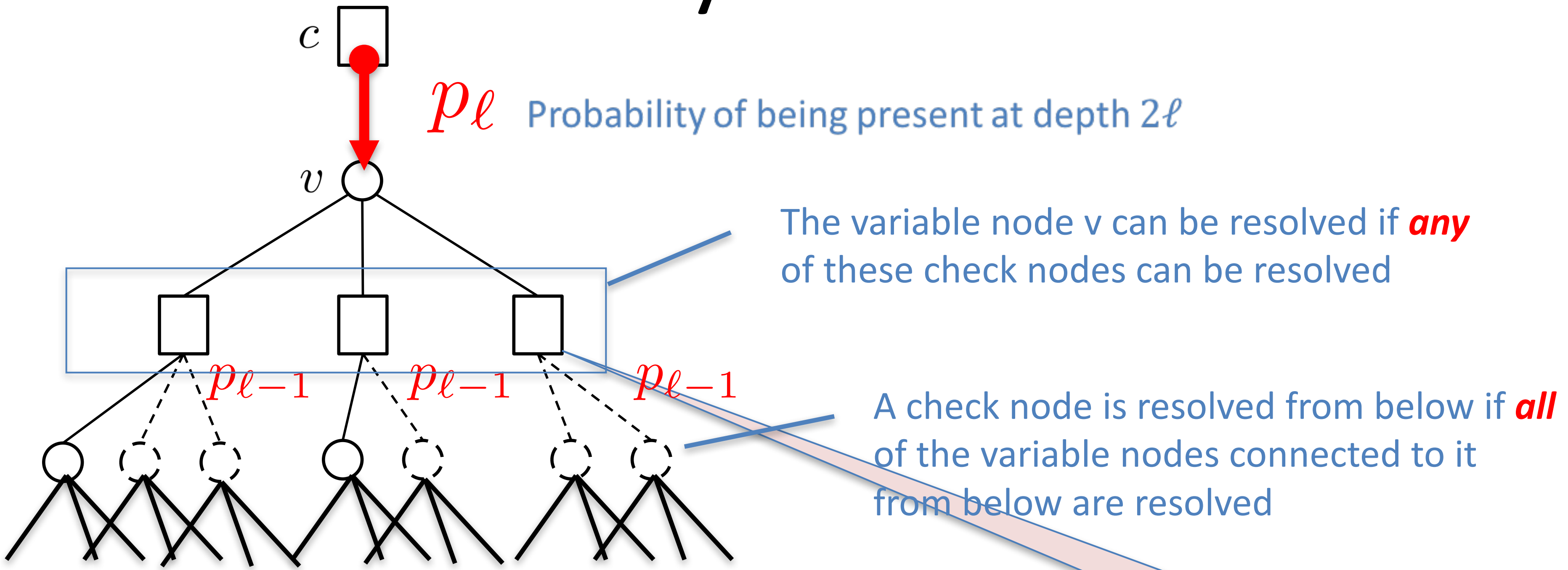
Probability of being present at depth 2ℓ

The variable node v can be resolved if ***any*** of these check nodes can be resolved

A check node is resolved from below if ***all*** of the variable nodes connected to it from below are resolved

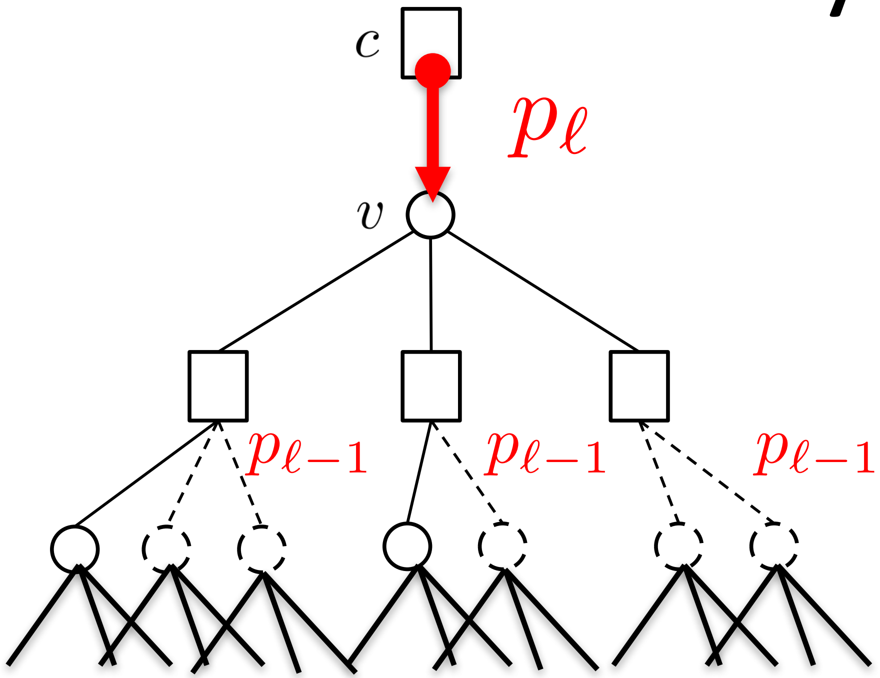
$$p_\ell = \left[1-(1-p_{\ell-1})^3\right] \times \boxed{\left[1-(1-p_{\ell-1})^2\right]} \times$$

Density evolution



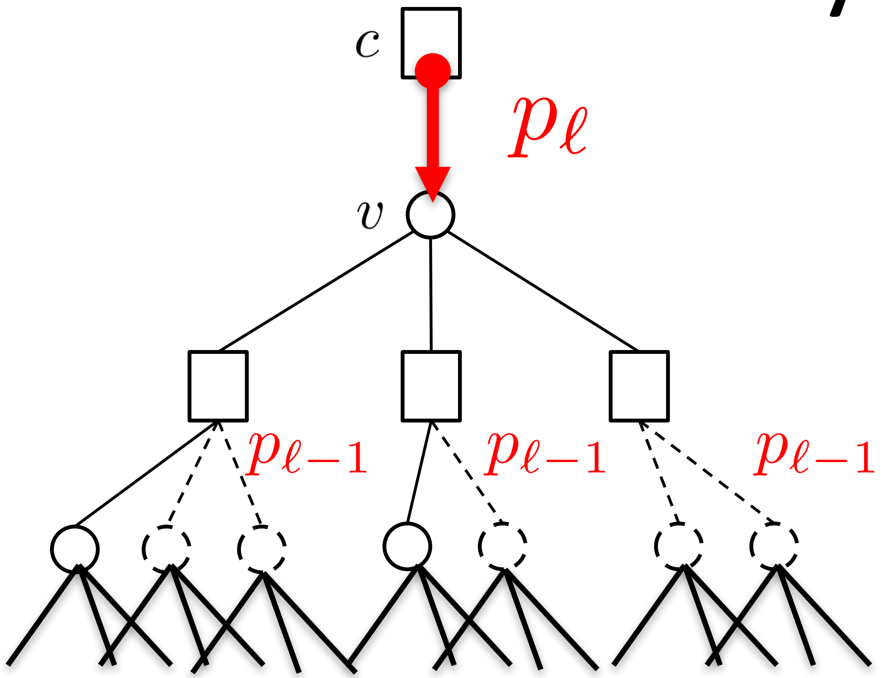
$$p_\ell = \left[1-(1-p_{\ell-1})^3\right] \times \left[1-(1-p_{\ell-1})^2\right] \times \boxed{\left[1-(1-p_{\ell-1})^2\right]}$$

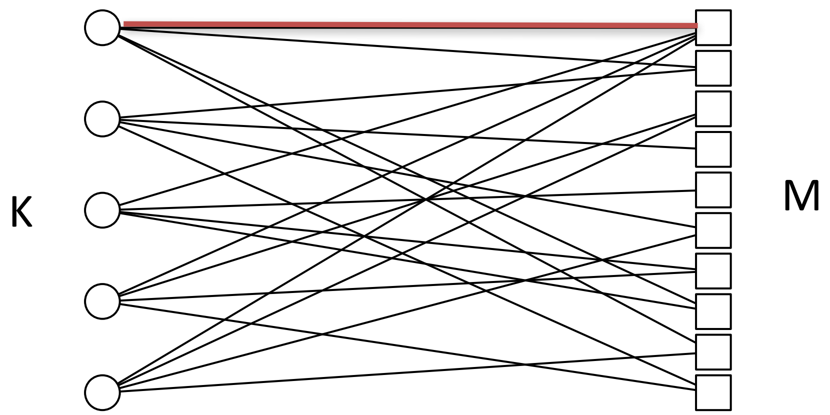
Density evolution



$$p_\ell = \left[1-(1-p_{\ell-1})^3\right] \times \left[1-(1-p_{\ell-1})^2\right] \times \left[1-(1-p_{\ell-1})^2\right]$$

Density evolution





example: $d = 4$

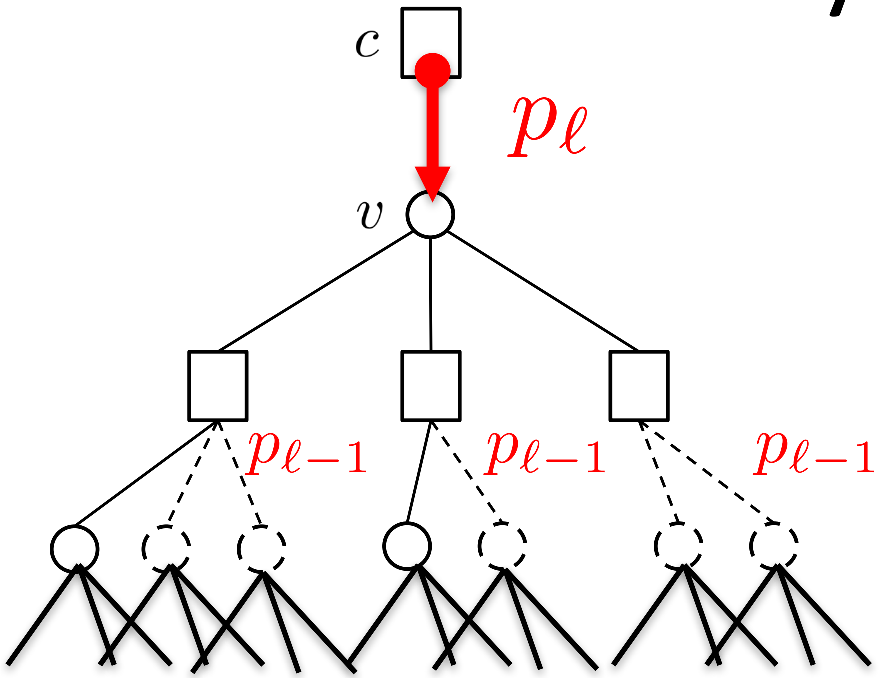
$$p_\ell = \left[1-(1-p_{\ell-1})^3\right] \times \left[1-(1-p_{\ell-1})^2\right] \times \left[1-(1-p_{\ell-1})^2\right]$$

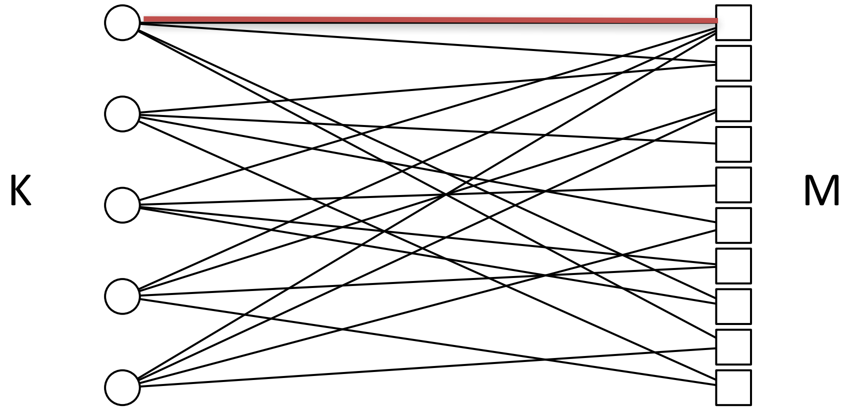
Regular graph construction:
Connect every variable node to **d** check nodes chosen uniformly at random

Number of children of check nodes has Poisson distribution with mean Kd/M

$$\Pr\{\text{a check node is resolved}\} = \sum_c e^{-\frac{Kd}{M}} \frac{\left(\frac{Kd}{M}\right)^c}{c!} (1-p_{\ell-1})^c = e^{-\frac{Kd}{M} p_{\ell-1}}$$

Density evolution





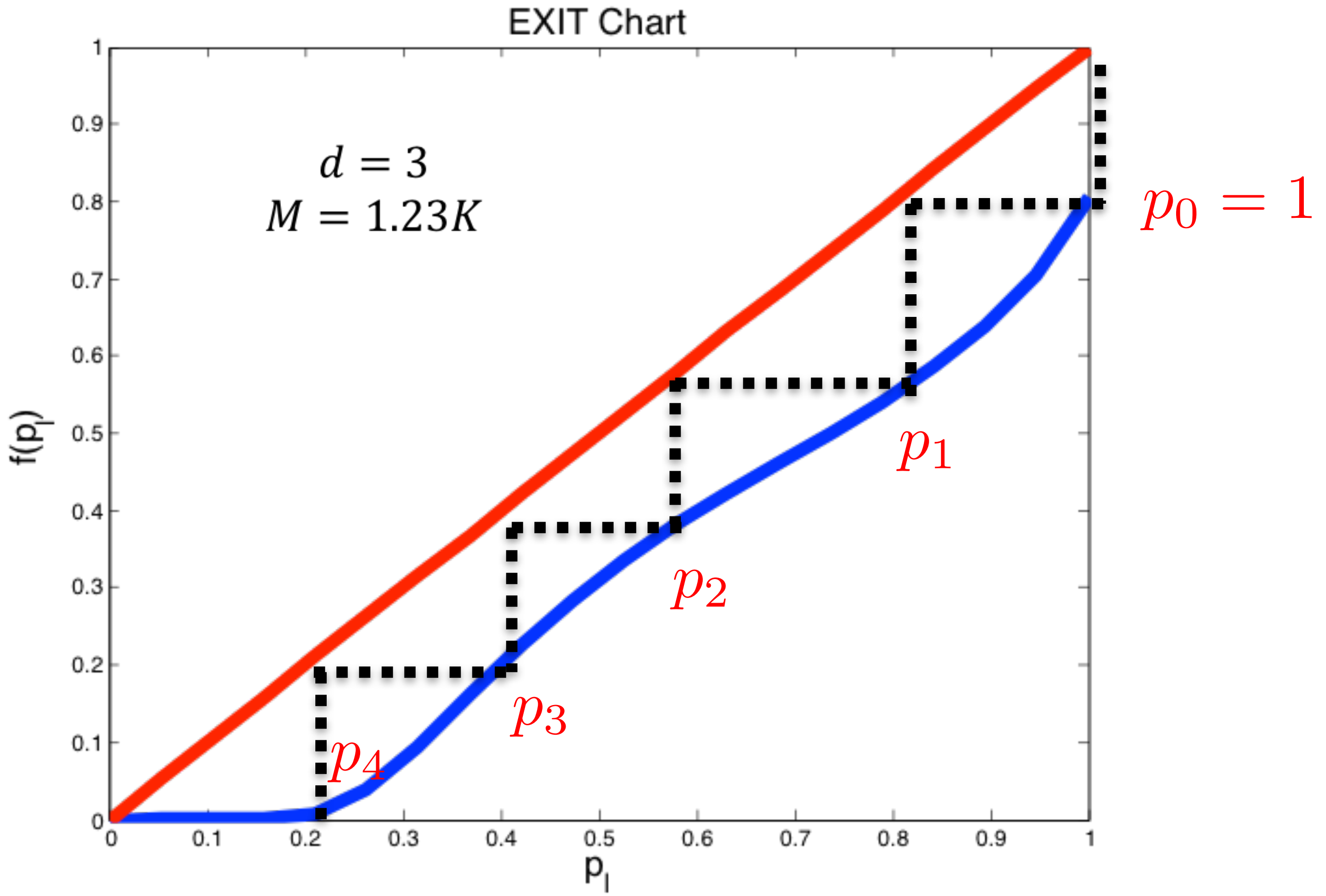
example: $d = 4$

$$p_\ell = \left[1-(1-p_{\ell-1})^3\right] \times \left[1-(1-p_{\ell-1})^2\right] \times \left[1-(1-p_{\ell-1})^2\right]$$

$$\boxed{p_\ell = \left(1 - e^{-\frac{Kd}{M}p_{\ell-1}}\right)^{d-1}}$$

- *Need p_ℓ → 0 as ℓ → ∞.*
- *Choose K, M and d so that p_ℓ goes to zero!*

Density evolution

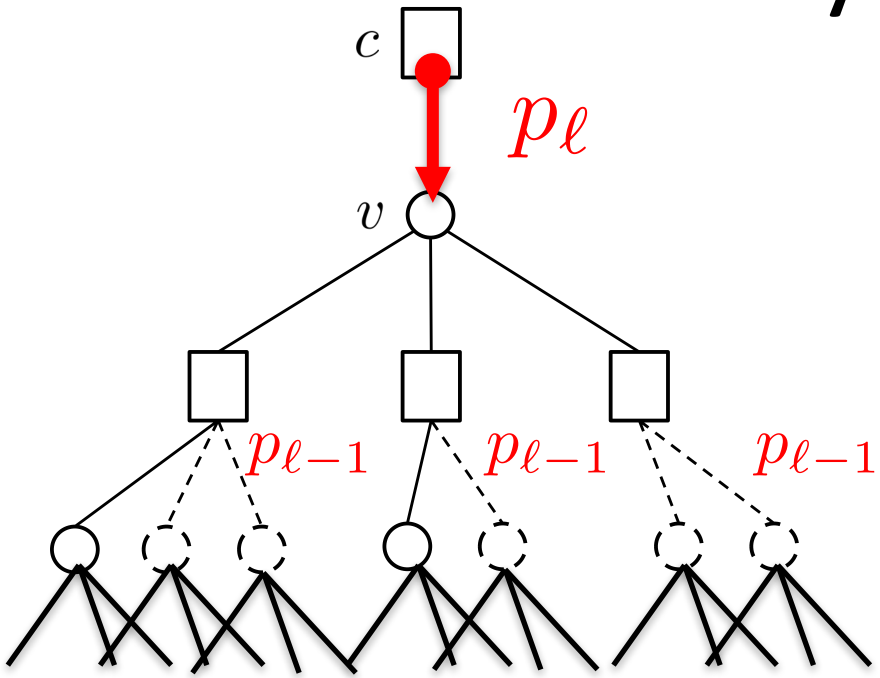


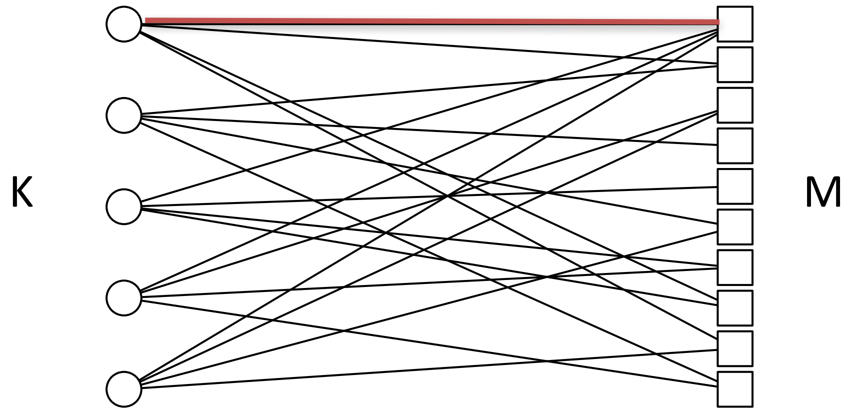
$$p_\ell = \left(1 - e^{-\frac{Kd}{M} p_{\ell-1}}\right)^{d-1}$$

- K=# of active bands
- M= # of channels
- d= left degree (# of edges from bands to channels)

Stephan ten Brink '99
Richardson & Urbanke '08

Density evolution





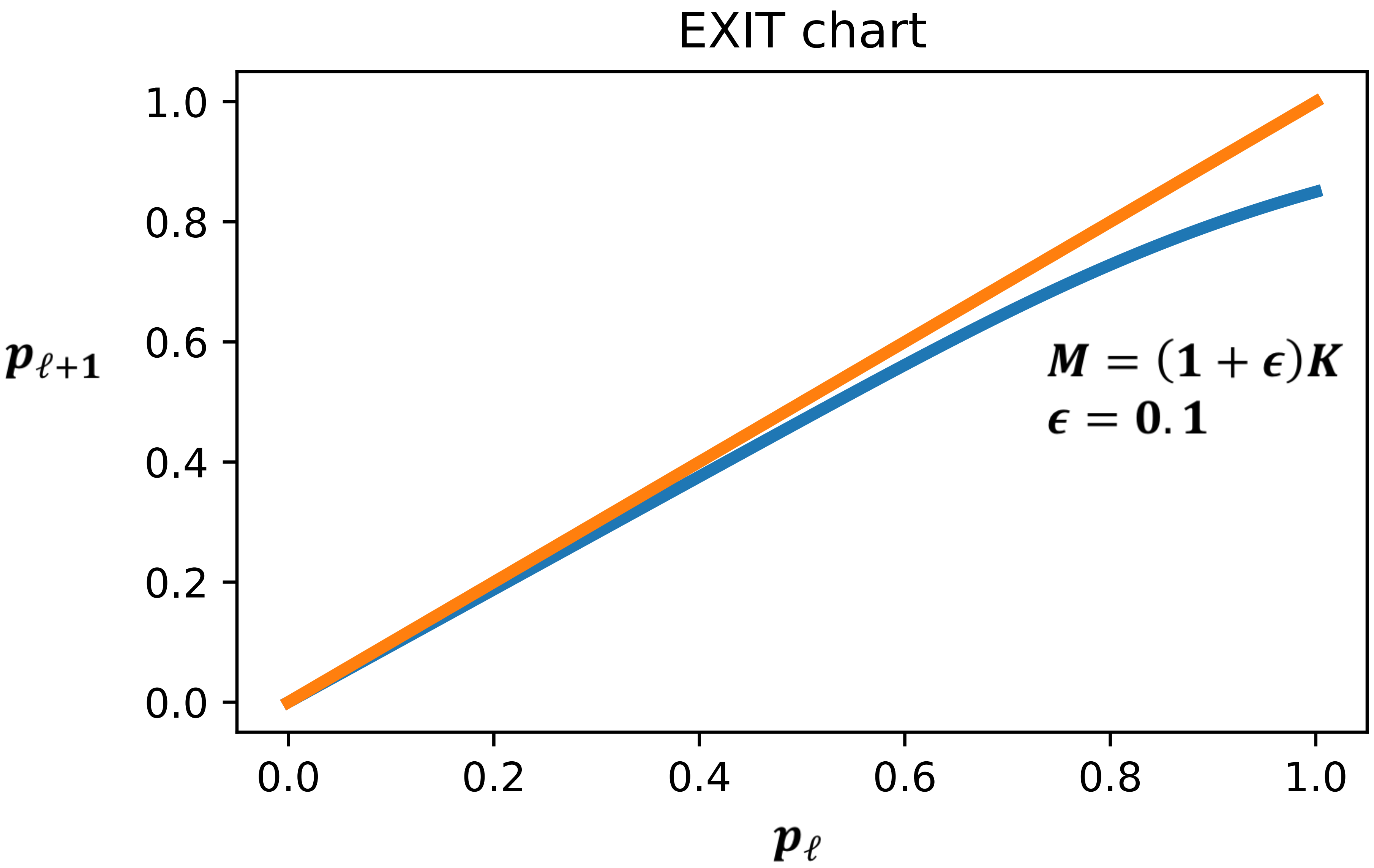
Set:

- $M = (1+\epsilon)K$
- $D > 1/\epsilon$
- Node degree distribution $\mathrm{P}(\text{degree} = j) = \frac{D+1}{D}\frac{1}{j(j-1)}$, for $j = 2, \dots, D+1$

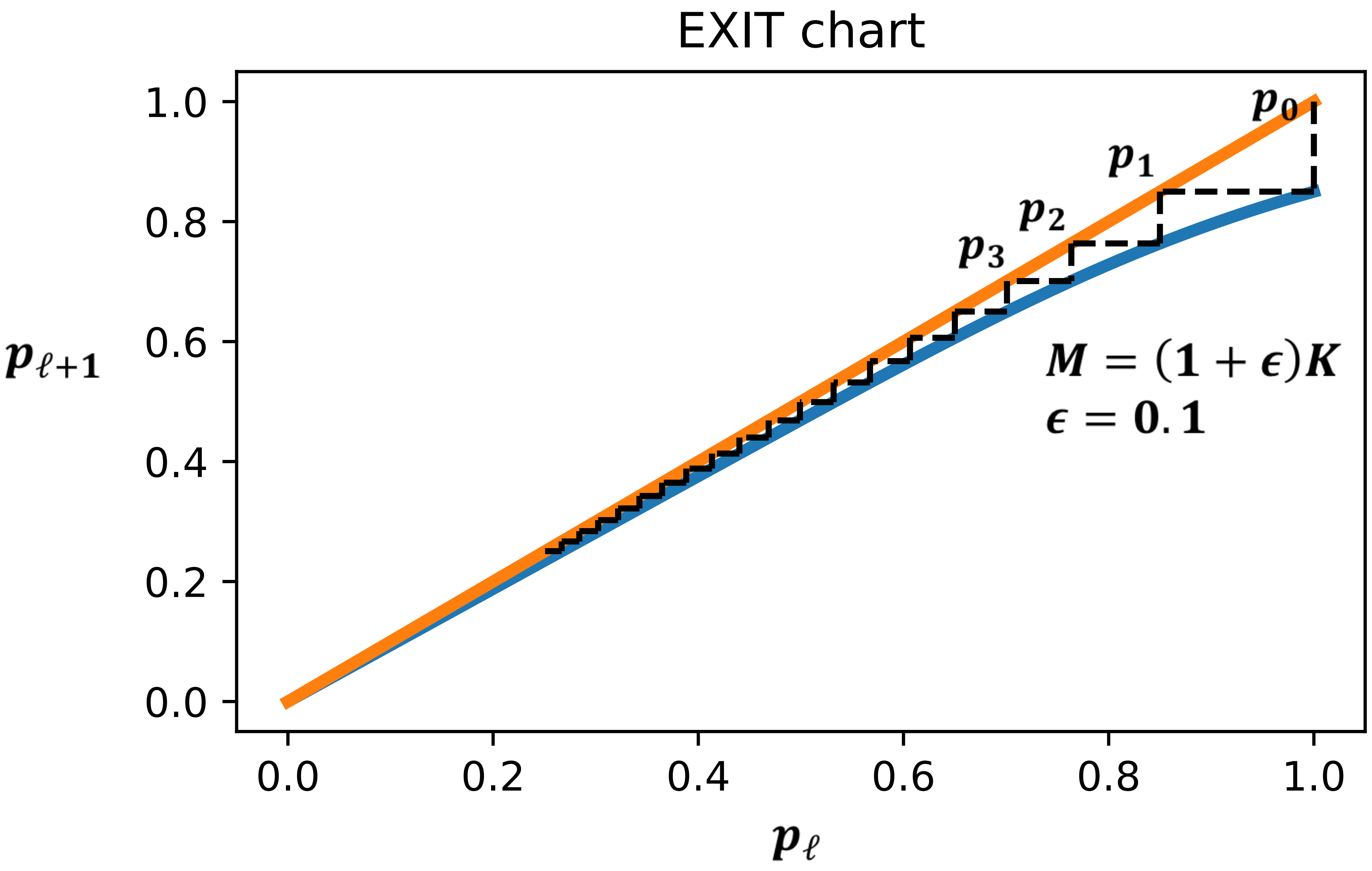
$$\boxed{p_\ell = \frac{1}{H(D)} \sum_{j=2}^{D+1} \frac{1}{j-1} \left(1 - e^{-\frac{\bar{d}}{1+\epsilon} p_{\ell-1}}\right)}$$

p_ℓ goes to zero!

Density evolution



Density evolution



Algorithm analysis

- **Density Evolution**
  - assumes that the directed neighborhood is a tree
  - tree-based average analysis

  Density evolution equations

  p_ℓ can be made arbitrarily small ***with O(1) number of iterations***

Algorithm analysis

- **Density Evolution**
  - assumes that the directed neighborhood is a tree
  - tree-based average analysis

  Density evolution equations

  p_ℓ can be made arbitrarily small ***with O(1) number of iterations***

$Kd(1 - p_\ell)$ **edges** removed

Algorithm analysis

Performance concentration:

- Actual performance concentrated around the density evolution
- $P(|\text{\#of actual remaining edges} - Kdp_\ell| > \epsilon_2) \to 0, \forall \epsilon_2 > 0$

$Kd(1 - p_\ell)$ **edges** removed

Algorithm analysis

?

$Kd(1 - p_\ell)$ **edges** removed

Kdp_ℓ **edges** remain

Kd edges to be removed

Algorithm analysis

- **Expander Graph**
  - the remaining Kdp_ℓ edges form an **expander graph**
  - expander graphs guarantee steady supplies of **single-tons**

ALL non-zero coefficients recovered w.h.p.

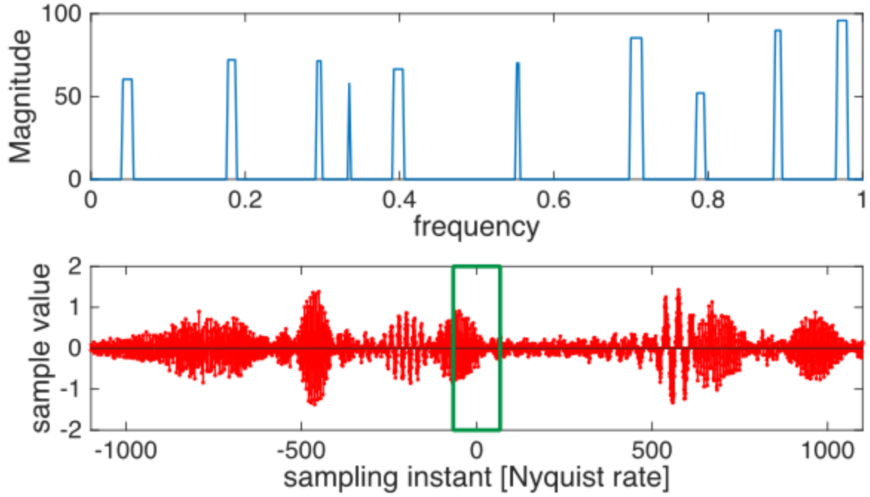
$Kd(1-p_\ell)$ **edges** removed

Kdp_ℓ **edges** remain

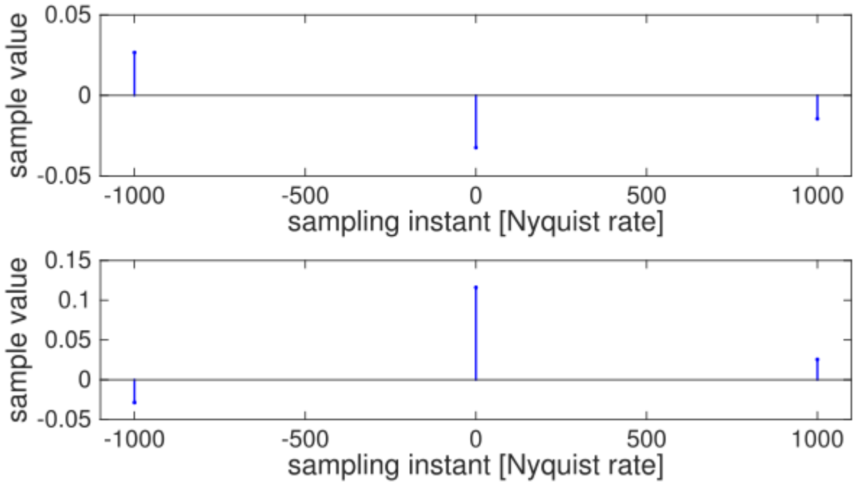
Kd edges to be removed

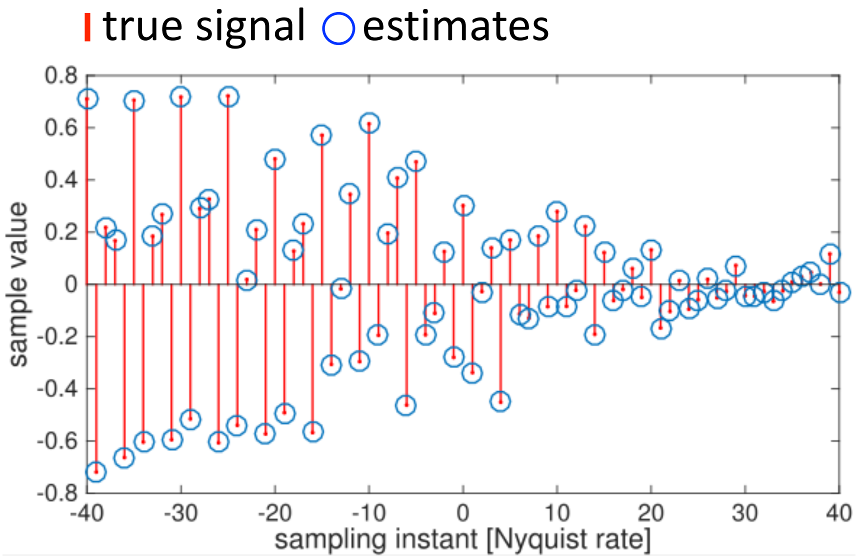
Back to sub-Nyquist sampling: Numerical experiment

Input spectrum and time domain signal



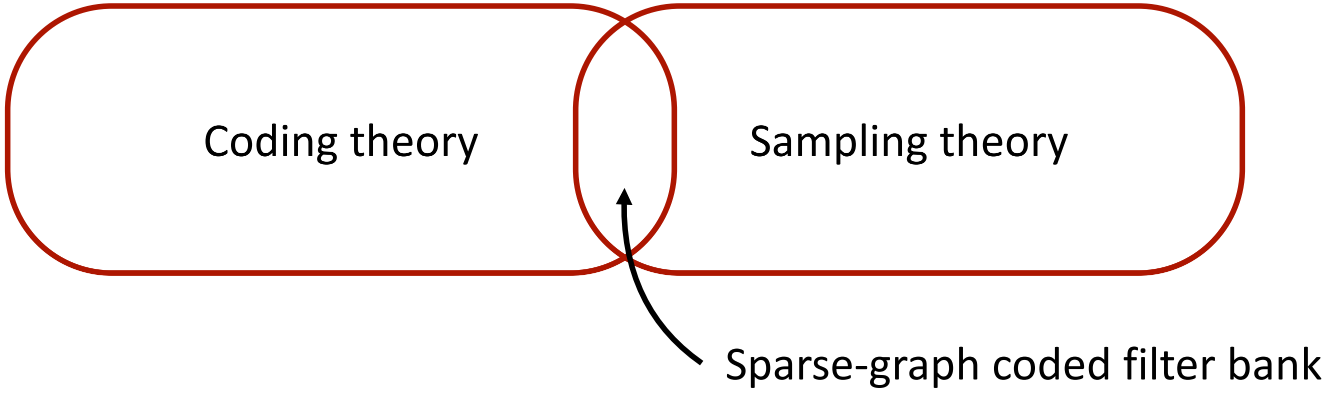
Output from two sample channels



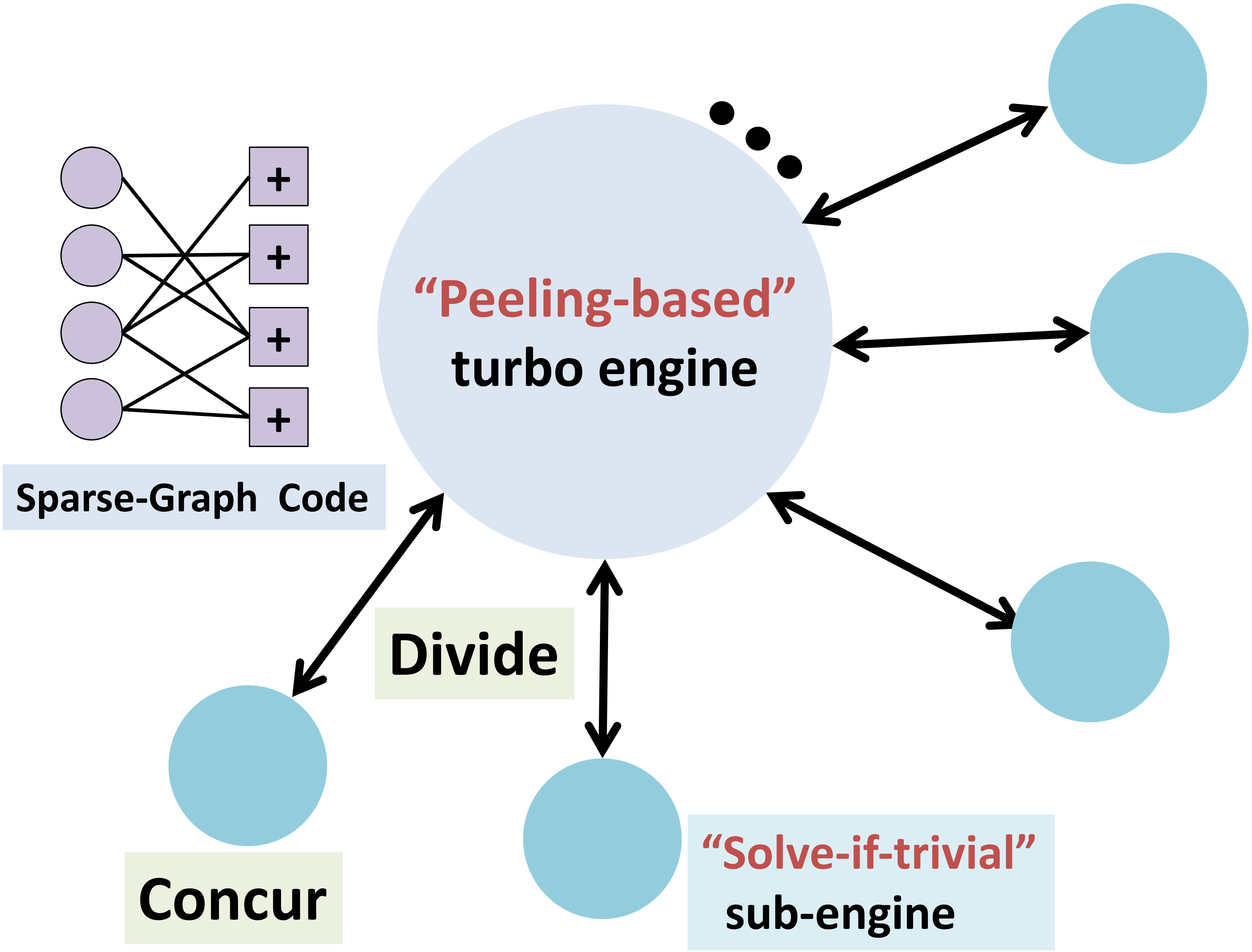


- Lebesgue measure $f_L = 0.1$
- Number of slices $N = 1000$
- Number of channels $M = 284$
- Sampling rate $f_S = 0.284$

Interesting connection

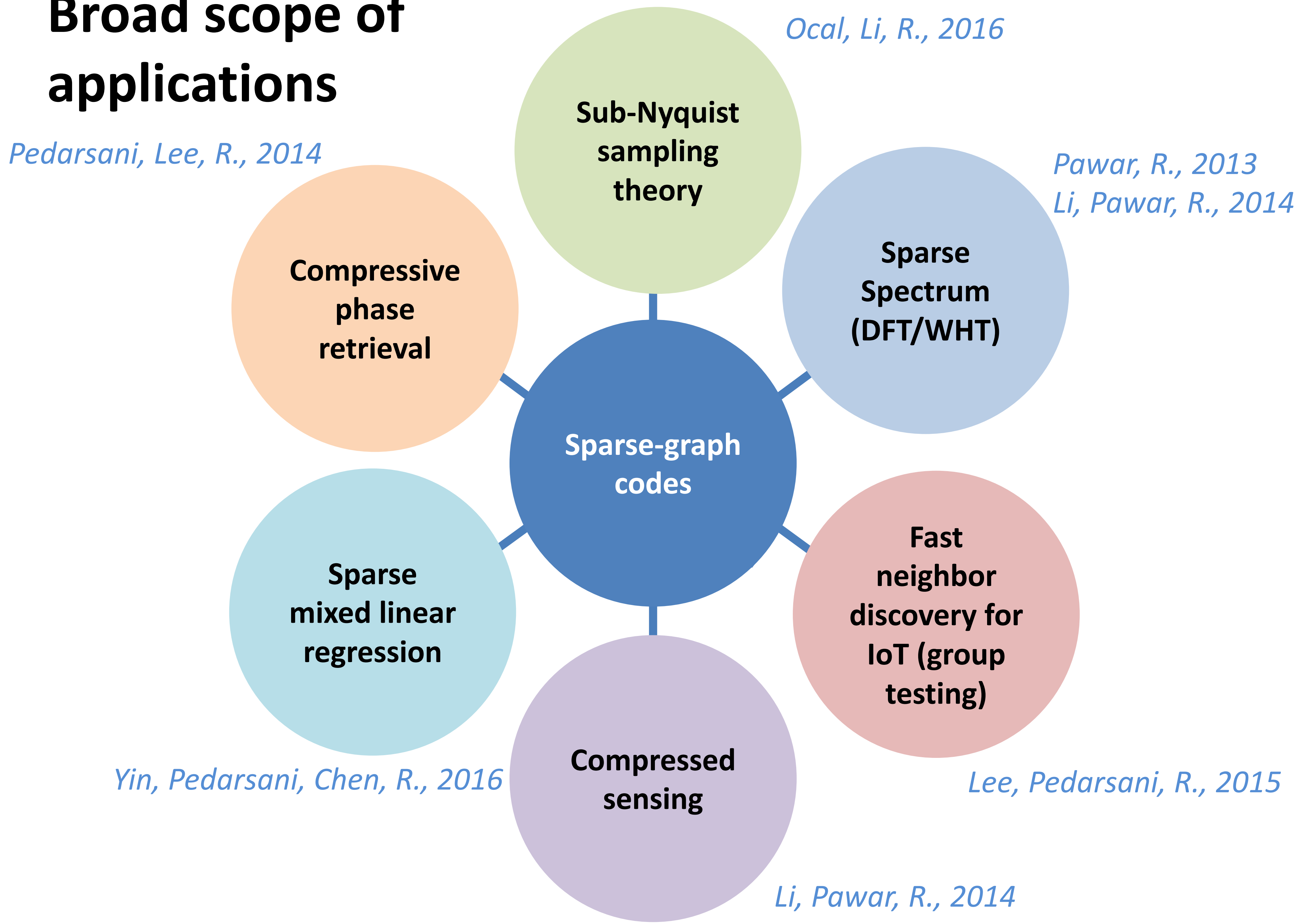


- *Minimum-rate spectrum-blind* sampling
- *Coding theory* and *sampling theory*
  - Capacity-approaching codes for erasure channels
  - Filter banks that approach Landau rate for sampling

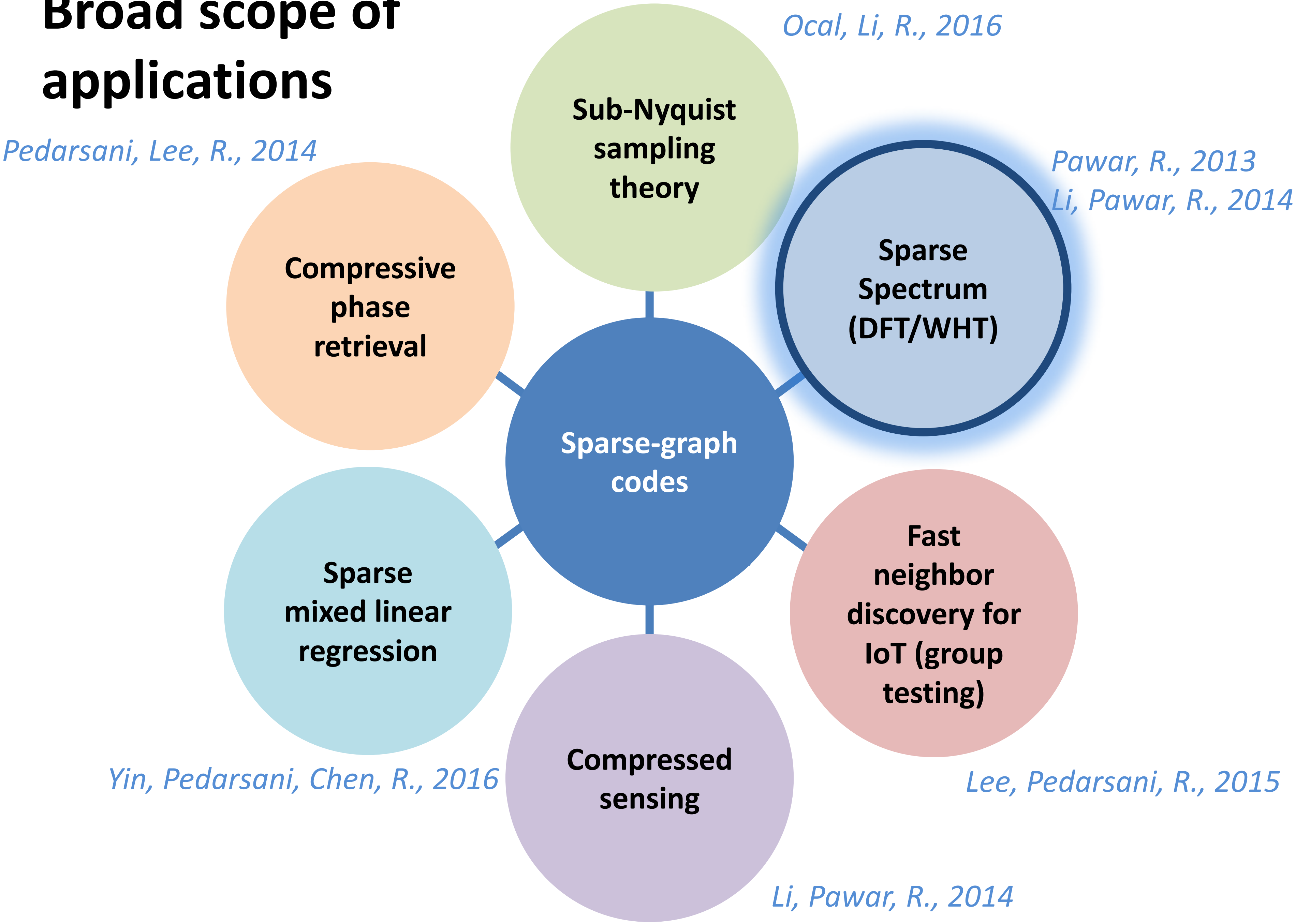


+
+
+
+
Sparse-Graph Code
"Peeling-based"
turbo engine
Divide
Concur
"Solve-if-trivial"
sub-engine

Broad scope of applications

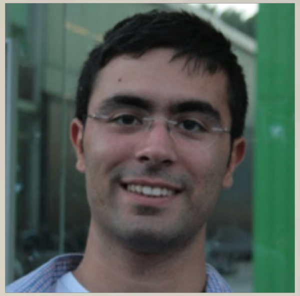


Broad scope of applications



Sameer Pawar

Simon Li



Orhan Ocal

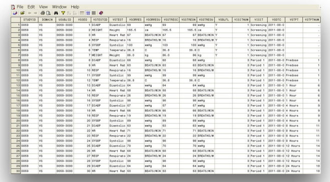
Chapter 4

Speeding up learning and sparse recovery

Motivation

- Given training data points (x, y), our goal is to learn

$(x : \text{feature}, y : \text{label})$

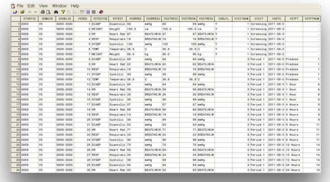


Dataset

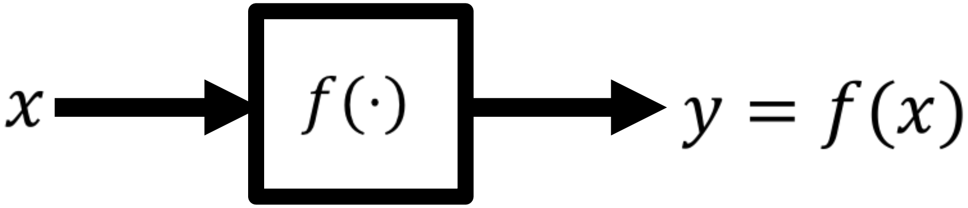
Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



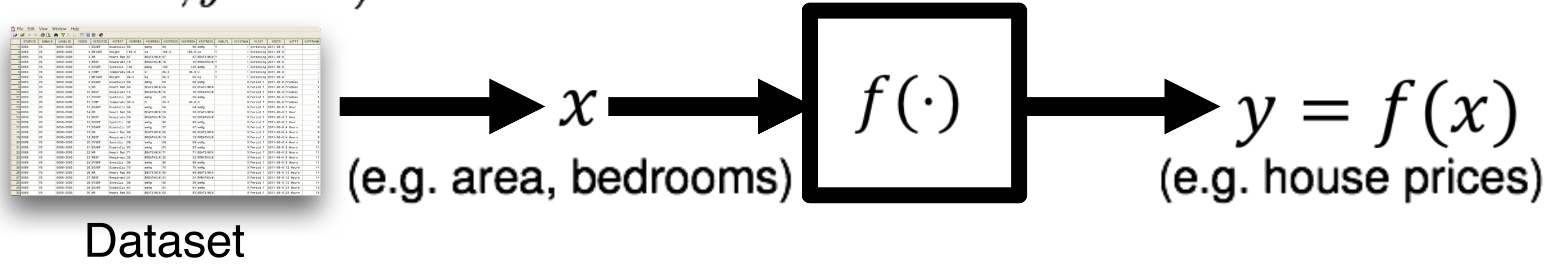
Dataset



Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

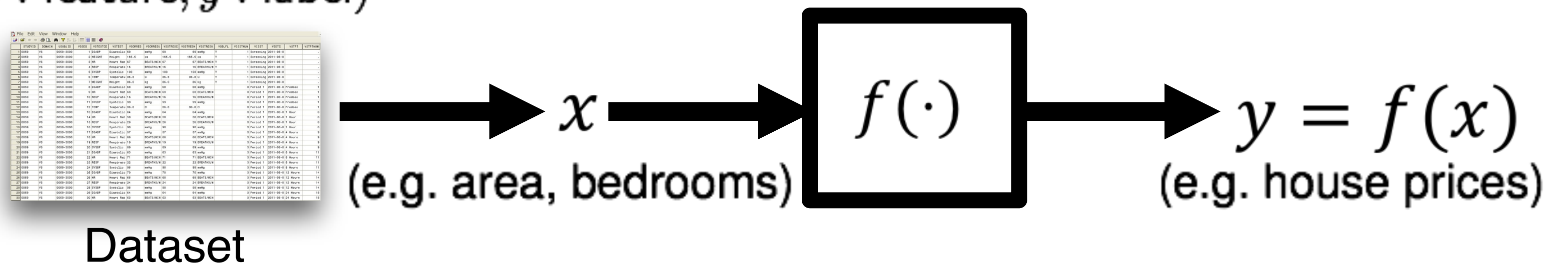
$(x : \text{feature}, y : \text{label})$



Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$

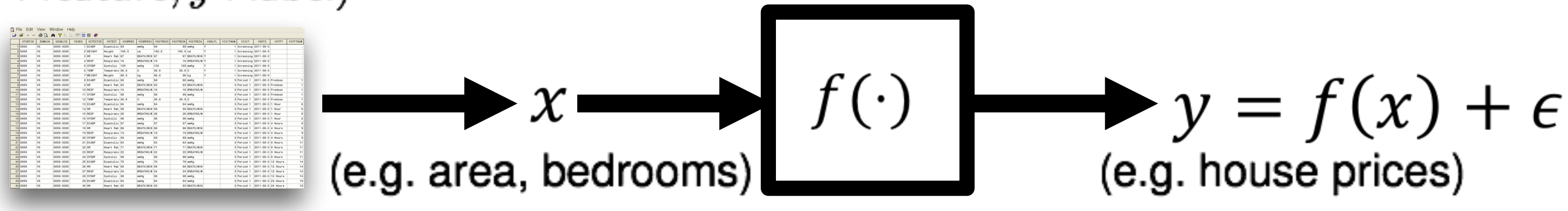


- Questions of interest
  - Sample complexity: how many data points do we need?
  - Computational complexity: how much time does it take?

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



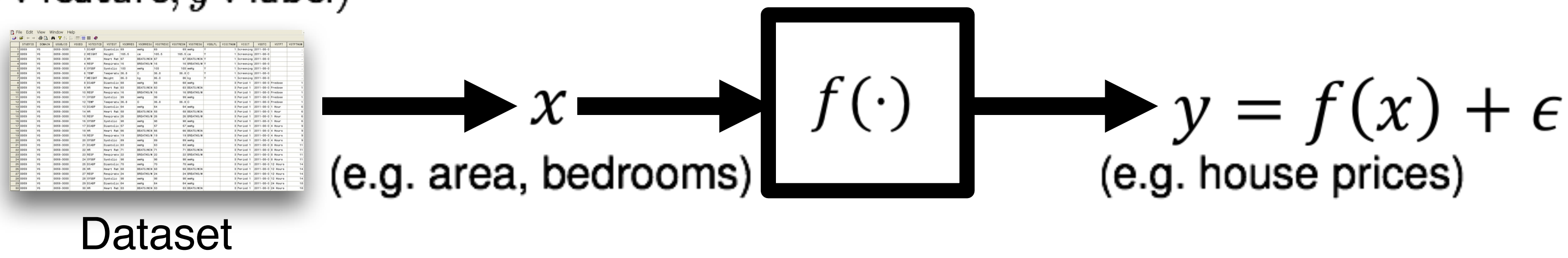
Dataset

- Questions of interest
  - Sample complexity: how many data points do we need?
  - Computational complexity: how much time does it take?
  - Robustness: how accurate and stable is it?

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



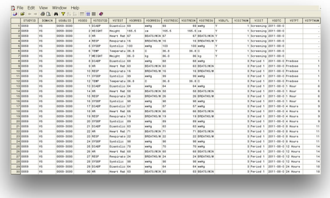
e.g. $f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b$

Problem Dimension $N = 3$

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x$: feature, y : label$)$



Dataset

$\longrightarrow x \longrightarrow \boxed{f(\cdot)} \longrightarrow y = f(x) + \epsilon$

(e.g. area, bedrooms) (e.g. house prices)

e.g. $f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b$

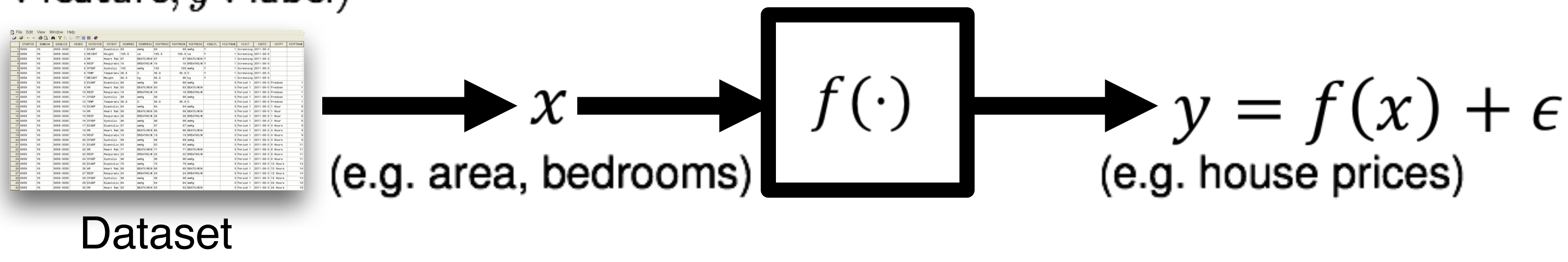
EASY!

Problem Dimension $N = 3$

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



e.g. $f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b$

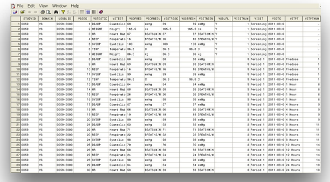
However…

Problem Dimension $N = 3$

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



Dataset

$$\longrightarrow x \longrightarrow \boxed{f(\cdot)} \longrightarrow y = f(x) + \epsilon$$

(e.g. area, bedrooms)

(e.g. house prices)

e.g. $f(x_1, x_2) = a_1 x_1 + a_2 x_2 + b$

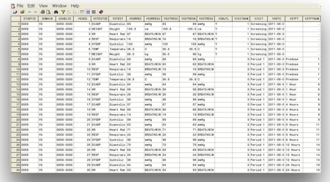
in reality…

Problem Dimension $N = 3$

Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



Dataset

$$x \longrightarrow \boxed{f(\cdot)} \longrightarrow y = f(x) + \epsilon$$

in reality...

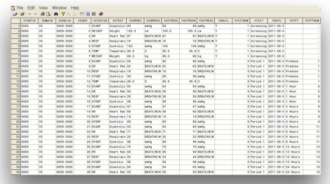
e.g.
$$\begin{aligned}
f(x) = {} & \frac{D}{Dt}\overline{w'^i w'^j} + \overline{w'^i w'^\alpha}\nabla_\alpha \bar{u}^j + \overline{w'^j w'^\alpha}\nabla_\alpha \bar{u}^i - \alpha\left(g^{i\alpha}\overline{w'^j \frac{T'}{\bar{T}}} + g^{j\alpha}\overline{w'^i \frac{T'}{\bar{T}}}\right)\left(\nabla_\alpha \bar{\Phi} + \frac{D\bar{u}_\alpha}{Dt}\right) \\
& + \frac{1}{\bar{\rho}}\nabla_\alpha[\overline{\bar{\rho} u'^\alpha w'^i w'^j} + \overline{(g^{i\alpha} w'^j + g^{j\alpha} w'^i)P'} - \overline{w'^i \sigma^{j\alpha}(u')} - \overline{w'^j \sigma^{i\alpha}(u')}] \\
& + \frac{1}{\bar{\rho}}\overline{w'^i w'^j \nabla_\alpha(\bar{\rho} u'^\alpha)} - \overline{P'(g^{i\alpha}\nabla_\alpha w'^j + g^{j\alpha}\nabla_\alpha w'^i)} = -\frac{1}{\bar{\rho}}[\overline{\sigma^{i\alpha}(u')\nabla_\alpha w'^j} + \overline{\sigma^{j\alpha}(u')\nabla_\alpha w'^i}] = -\epsilon_2^{ij}\,, \\
(1+e_4)\frac{D}{Dt}\overline{\left(\frac{T'}{\bar{T}}\right)^2} & - 2f(t)\overline{\left(\frac{T'}{\bar{T}}\right)^2} - 2\overline{w'^\alpha \frac{T'}{\bar{T}}}D_\alpha + \frac{1}{(1+e_4)\bar{\rho}C_p^2}\nabla_\alpha\left[(1+e_4)^2 C_p^2 \overline{\bar{\rho} w'^\alpha \left(\frac{T'}{\bar{T}}\right)^2}\right] + \frac{1+e_4}{\bar{\rho}}\overline{\left(\frac{T'}{\bar{T}}\right)^2 \nabla_\alpha(\rho u'^\alpha)} \\
& + \frac{2}{\bar{\rho}\bar{T}C_p}\overline{\frac{T'}{\bar{T}}\left[P'\nabla_\alpha w'^\alpha - \nabla_\alpha(P'_g w'^\alpha) - \frac{DP'_g}{Dt}\right]} = \frac{2}{\bar{\rho}\bar{T}C_p}\overline{\frac{T'}{\bar{T}}[\sigma^{\alpha\beta}(u')\nabla_\alpha u'_\beta - \nabla_\alpha F_r'^\alpha]} = -\epsilon_2\,, \\
(1+e_4)\Bigg[\frac{D}{Dt}\left(\overline{w'^i \frac{T'}{\bar{T}}}\right) & + \overline{w'^\alpha \frac{T'}{\bar{T}}}\nabla_\alpha \bar{u}^i - \alpha\overline{\left(\frac{T'}{\bar{T}}\right)^2} g^{i\alpha}\left(\nabla_\alpha \bar{\Phi} + \frac{D\bar{u}_\alpha}{Dt}\right)\Bigg] - f(t)\overline{w'^i \frac{T'}{\bar{T}}} - \overline{w'^i w'^\alpha}D_\alpha \\
& + \frac{1}{\bar{\rho}C_p}\nabla_\alpha\left[(1+e_4)C_p \overline{\bar{\rho} w'^i w'^\alpha \frac{T'}{\bar{T}}}\right] + \frac{1+e_4}{\bar{\rho}}\overline{w'^i \frac{T'}{\bar{T}}\nabla_\alpha(\rho u'^\alpha)} + \frac{1}{\bar{\rho}\bar{T}C_p}\overline{w'^i\left[P'\nabla_\alpha w'^\alpha - \nabla_\alpha(P'_g w'^\alpha) - \frac{DP'_g}{Dt}\right]}
\end{aligned}$$

Problem Dimension $N \to \infty$

Motivation

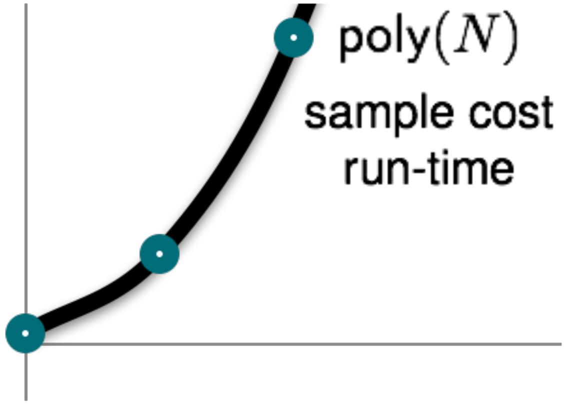
- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



Dataset

$$\longrightarrow x \longrightarrow \boxed{f(\cdot)} \longrightarrow y = f(x) + \epsilon$$



Problem Dimension N

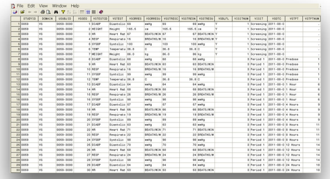
e.g.
$$
\begin{aligned}
f(x) = {} & \frac{D}{Dt}\overline{w'^i w'^j} + \overline{w'^i w'^\alpha}\nabla_\alpha \bar u^j + \overline{w'^j w'^\alpha}\nabla_\alpha \bar u^i - \alpha\left(g^{i\alpha}\overline{w'^j\frac{T'}{\bar T}} + g^{j\alpha}\overline{w'^i\frac{T'}{\bar T}}\right)\left(\nabla_\alpha\bar\Phi + \frac{D\bar u_\alpha}{Dt}\right) \\
& + \frac{1}{\bar\rho}\nabla_\alpha\left[\overline{\bar\rho u'^\alpha w'^i w'^j} + \overline{(g^{i\alpha}w'^j + g^{j\alpha}w'^i)P'} - \overline{w'^i\sigma^{j\alpha}(u')} - \overline{w'^j\sigma^{i\alpha}(u')}\right] \\
& + \frac{1}{\bar\rho}\overline{w'^i w'^j\nabla_\alpha(\bar\rho u'^\alpha)} - \overline{P'(g^{i\alpha}\nabla_\alpha w'^j + g^{j\alpha}\nabla_\alpha w'^i)} = -\frac{1}{\bar\rho}\left[\overline{\sigma^{i\alpha}(u')\nabla_\alpha w'^j} + \overline{\sigma^{j\alpha}(u')\nabla_\alpha w'^i}\right] = -\epsilon_2^{ij}\,, \\
& (1+e_4)\frac{D}{Dt}\overline{\left(\frac{T'}{\bar T}\right)^2} - 2f(t)\overline{\left(\frac{T'}{\bar T}\right)^2} - 2\overline{w'^\alpha\frac{T'}{\bar T}}D_\alpha + \frac{1}{(1+e_4)\bar\rho C_p^2}\nabla_\alpha\left[(1+e_4)^2 C_p^2\bar\rho\overline{w'^\alpha\left(\frac{T'}{\bar T}\right)^2}\right] + \frac{1+e_4}{\bar\rho}\overline{\left(\frac{T'}{\bar T}\right)^2\nabla_\alpha(\rho u'^\alpha)} \\
& + \frac{2}{\bar\rho\bar T C_p}\overline{\frac{T'}{\bar T}\left[P'\nabla_\alpha w'^\alpha - \nabla_\alpha(P'_g w'^\alpha) - \frac{DP'_g}{Dt}\right]} = \frac{2}{\bar\rho\bar T C_p}\overline{\frac{T'}{\bar T}\left[\sigma^{\alpha\beta}(u')\nabla_\alpha u'_\beta - \nabla_\alpha F'^\alpha_r\right]} = -\epsilon_2\,, \\
& (1+e_4)\left[\frac{D}{Dt}\left(\overline{w'^i\frac{T'}{\bar T}}\right) + \overline{w'^\alpha\frac{T'}{\bar T}}\nabla_\alpha\bar u^i - \alpha\overline{\left(\frac{T'}{\bar T}\right)^2}g^{i\alpha}\left(\nabla_\alpha\bar\Phi + \frac{D\bar u_\alpha}{Dt}\right)\right] - f(t)\overline{w'^i\frac{T'}{\bar T}} - \overline{w'^i w'^\alpha}D_\alpha \\
& + \frac{1}{\bar\rho C_p}\nabla_\alpha\left[(1+e_4)C_p\bar\rho\overline{w'^i w'^\alpha\frac{T'}{\bar T}}\right] + \frac{1+e_4}{\bar\rho}\overline{w'^i\frac{T'}{\bar T}\nabla_\alpha(\rho u'^\alpha)} + \frac{1}{\bar\rho\bar T C_p}\overline{w'^i\left[P'\nabla_\alpha w'^\alpha - \nabla_\alpha(P'_g w'^\alpha) - \frac{DP'_g}{Dt}\right]}
\end{aligned}
$$

Problem Dimension $N \to \infty$

Motivation

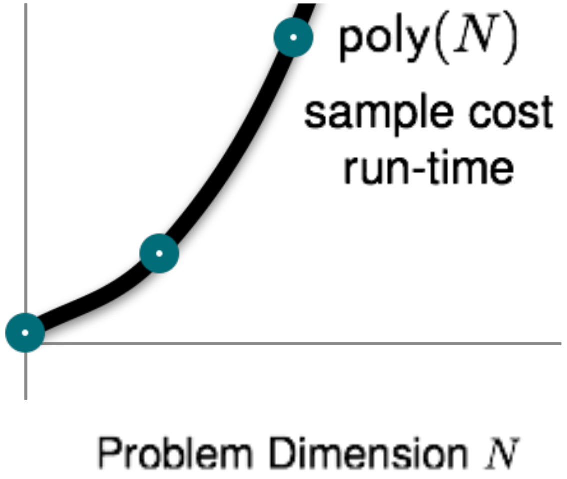
- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

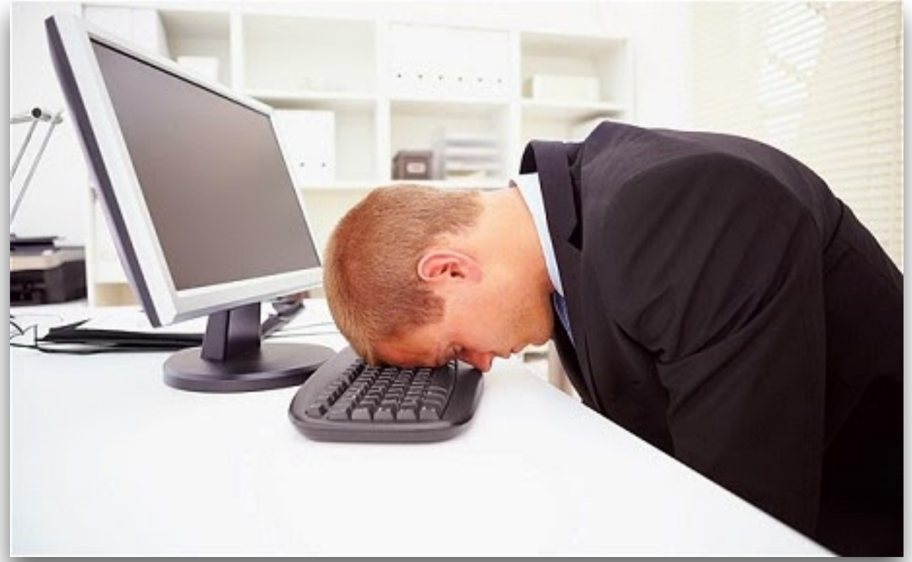
$(x : \text{feature}, y : \text{label})$



Dataset

$$\rightarrow x \rightarrow \boxed{f(\cdot)} \rightarrow y = f(x) + \epsilon$$

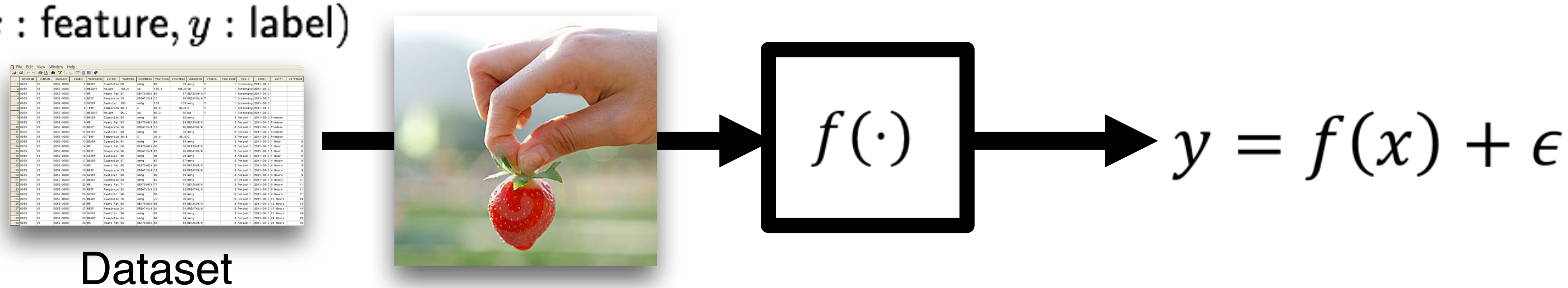




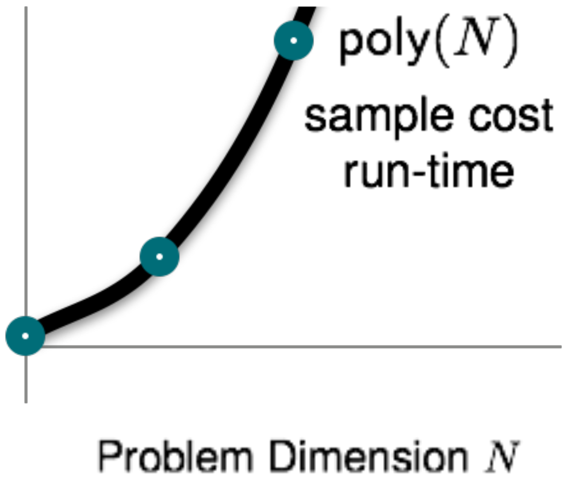
Motivation

- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



Dataset



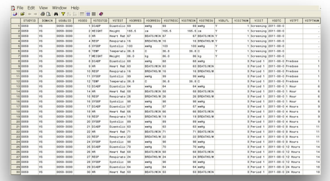
What **if**

- we can **actively choose** training data
- the model has **sublinear d.o.f**

Motivation

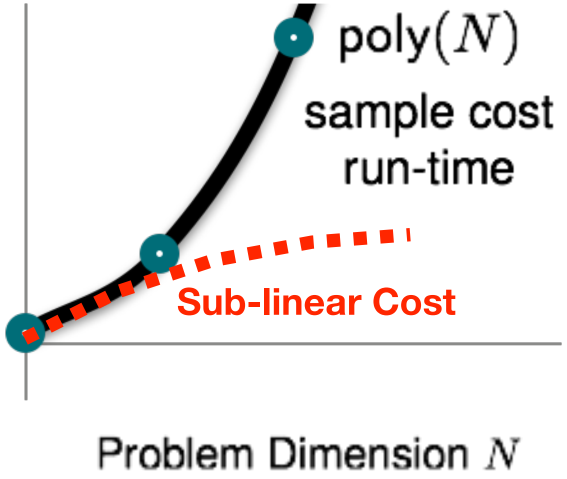
- Given training data points (x, y), our goal is to learn
  - **a certain rule** f that explains the label y based on features x:

$(x : \text{feature}, y : \text{label})$



Dataset

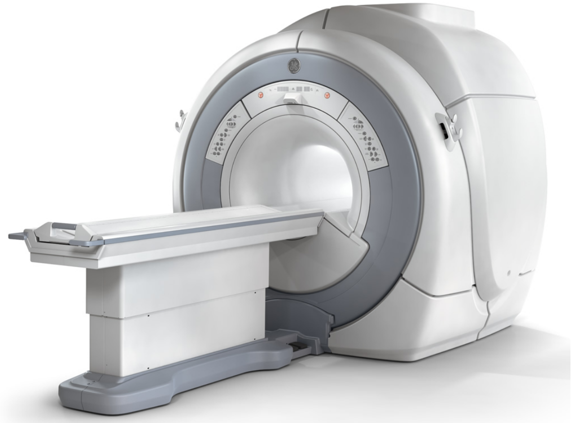
$$\longrightarrow \boxed{f(\cdot)} \longrightarrow y = f(x) + \epsilon$$



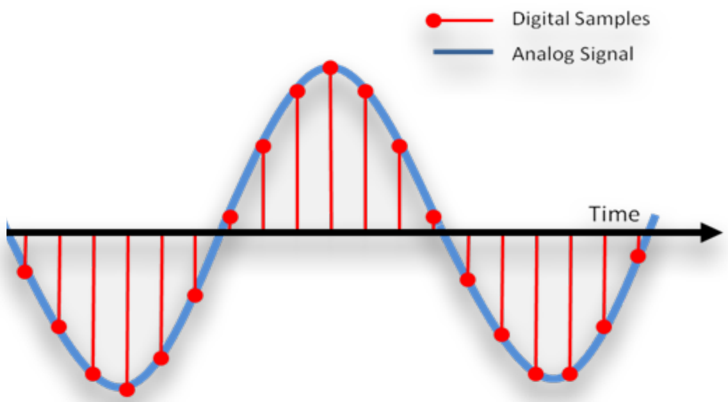
What **if**
- we can **actively choose** training data
- the model has **sublinear d.o.f.**

Can we achieve fast & robust learning with active sampling + coding theory?

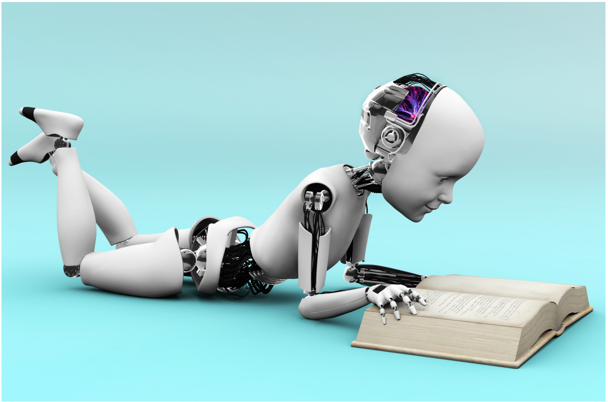
Applications



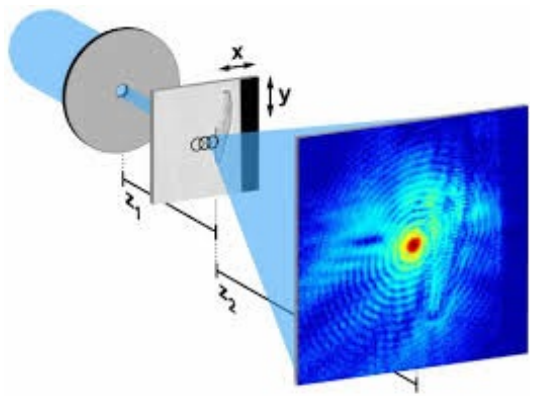
MRI



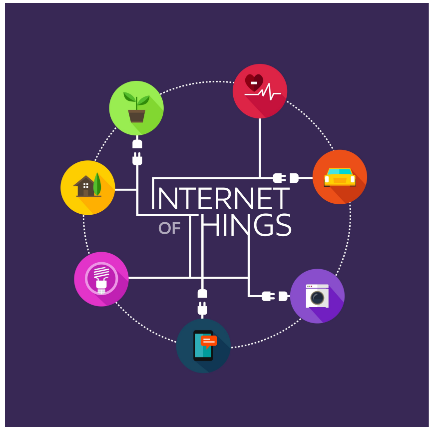
Sub-Nyquist Sampling



Machine Learning



Computational Imaging



IoT

Sparse Spectrum (DFT/WHT)

Sparse-graph codes

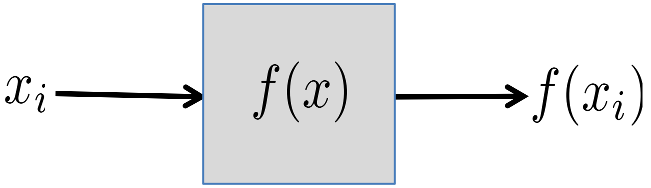
Sameer Pawar

Xiao (Simon) Li

Orhan Ocal

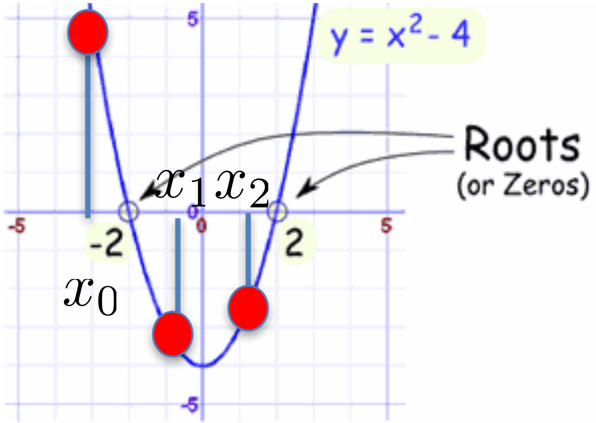
Learning polynomials: HS algebra edition

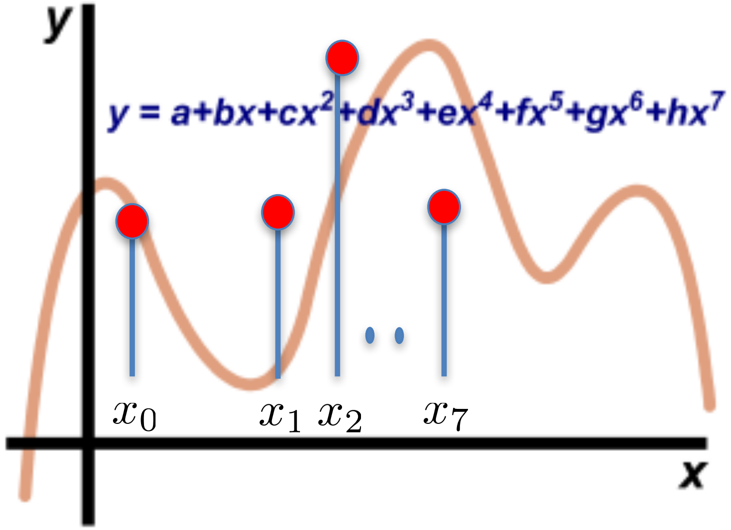
- Given $f(x) = \sum_{n=0}^{N-1} F_n x^n$
- Find coefficients $\{F_n\}_{n=0}^{N-1}$



Q. How many evaluations do we need?

A. N evaluations

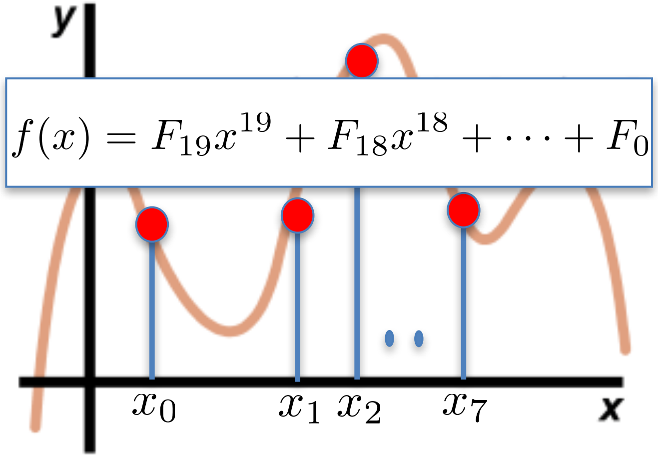




Recovering the coefficients

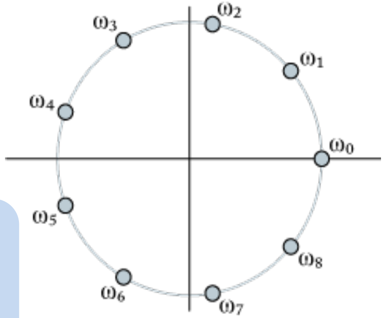
- Given $f(x) = \sum_{n=0}^{N-1} F_n x^n$
- Find coefficients $\{F_n\}_{n=0}^{N-1}$

$$x_i \longrightarrow \boxed{f(x)} \longrightarrow f(x_i)$$



$$f(x) = F_{19}x^{19} + F_{18}x^{18} + \cdots + F_0$$

$$\begin{bmatrix} f(X_0) \\ f(X_1) \\ f(X_2) \\ \vdots \\ f(X_{19}) \end{bmatrix} = \begin{bmatrix} 1 & X_0 & \cdots & X_0^{19} \\ 1 & X_1 & \cdots & X_1^{19} \\ 1 & X_2 & \cdots & X_2^{19} \\ \vdots & & & \\ 1 & X_{19} & \cdots & X_{19}^{19} \end{bmatrix} \begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ \vdots \\ F_{19} \end{bmatrix}$$



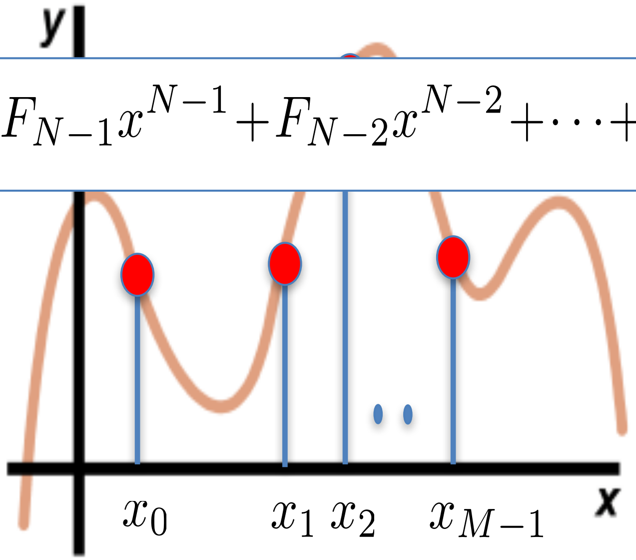
inverse Discrete Fourier Transform (DFT)
if $X_m = e^{\mathrm{i}\frac{2\pi}{N}m}$

What if only **K** of **N** coeffs. non-zero?

$$x_i \longrightarrow \boxed{f(x)} \longrightarrow f(x_i)$$

K sublinear in **N** (**K**/**N**➔0)

$$f(x) = F_{N-1}x^{N-1} + F_{N-2}x^{N-2} + \cdots + F_0$$



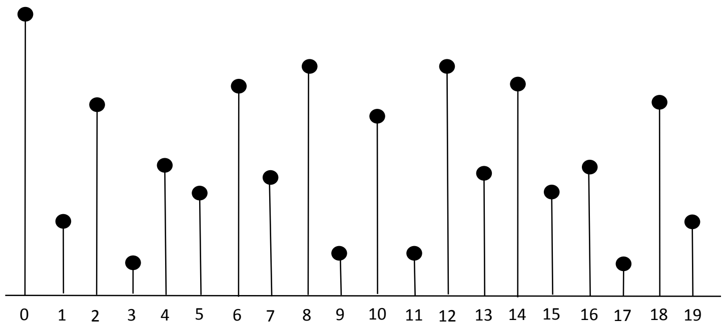
Example:
Degree **N= 1 million**
Sparsity **K = 200**

(spoiler alert)
*# evalulations = **616** ($\approx 3K$)*
computations = O(K log K)

Discrete Fourier Transform (DFT)

Compute the DFT of $x \in \mathbb{C}^N$:

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} X[k] e^{j\frac{2\pi}{N}kn}, \qquad n = 0, \cdots, N-1$$



FFT Algorithm

Sample complexity:	N
Computational cost:	O(N log N)

What if only K out of N Fourier coefficients are non-zero?

Example:

Length N = 1 million

Sparsity K = 200

(spoiler alert)

 # evalulations = 616 ($\approx 3K$)

computations = O(K logK)

Problem Formulation / Results

Compute the K-sparse DFT of $x \in \mathbb{C}^N$ with $K \ll N$:

$$x[n] = \frac{1}{\sqrt{N}} \sum_{k \in \mathcal{K}} X[k] e^{i \frac{2\pi k}{N} n} \qquad n = 0, \cdots, N-1$$

Support $\mathcal{K}$ chosen from $[N]$ uniformly at random

FFAST (Fast Fourier Aliasing-Based Sparse Transform)

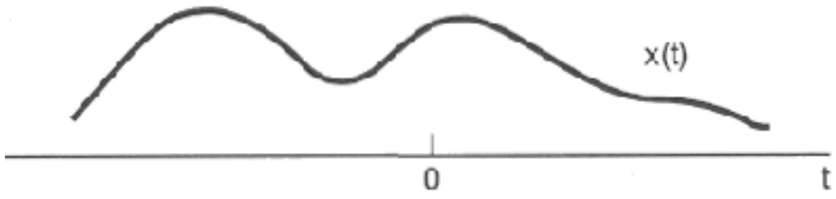
- ***Noiseless***: For **K** sublinear in **N**
  - Uses **fewer than 4K** samples
  - **O(K log K)** computation time
- ***Robust to noise***: **O(K $\log^{4/3}$ N)** samples in **O(K $\log^{7/3}$ N)** time

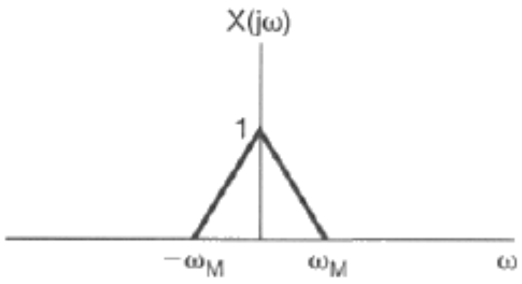
Sub-linear time recovery when d.o.f. sublinear!

Pawar, **R**, IEEE Trans. Inf. Theory , 2018

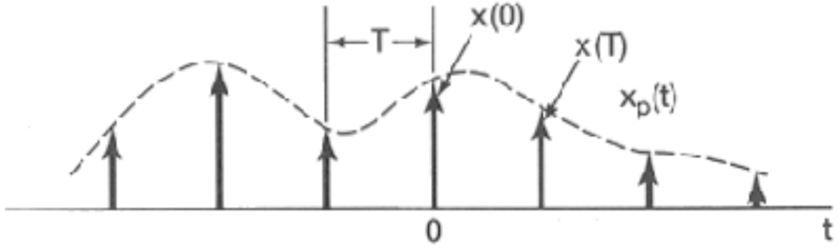
Aliasing

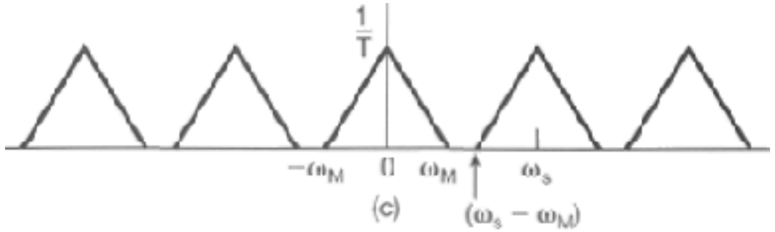
Signal and its spectrum



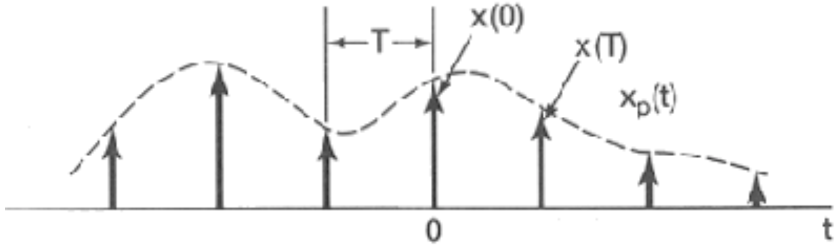


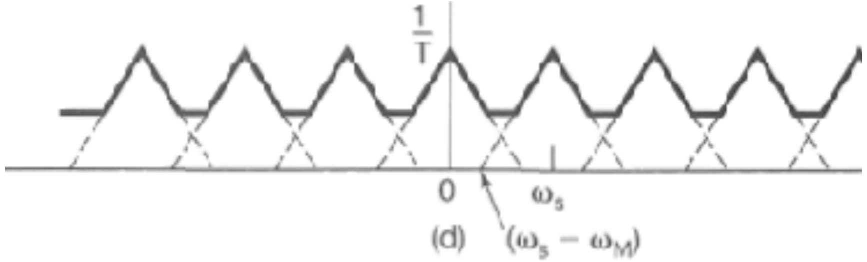
Sampling





Sub-sampling





Insights

Sub-sampling below Nyquist rate → **Aliasing** in the frequency domain

Clever sub-sampling (for **sparse** case) → **Good** "alias" code

Chinese-Remainder-Theorem guided subsampling — ***Sparse graph codes***

We use coding-theoretic tools

Design:

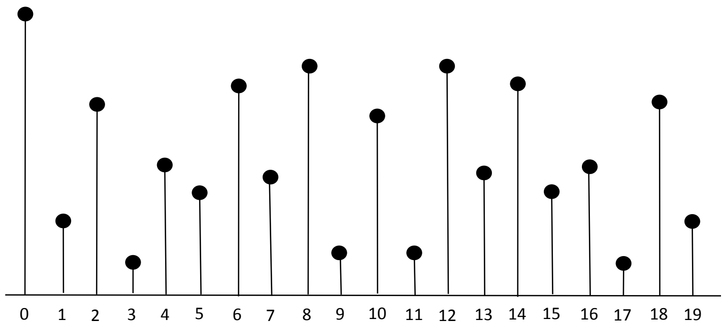
- Randomized constructions of good sparse-graph codes

Analysis:

- Density evolution, Martingales, Expander graph theory…

Main idea

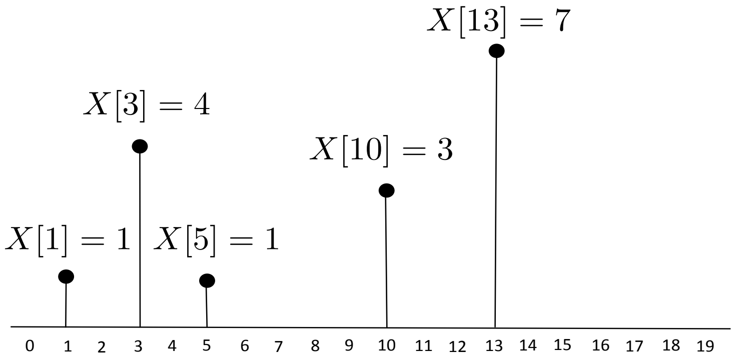
time-domain $x[n]$, length $N = 20$



$\Longleftarrow$ **DFT** $\Longrightarrow$

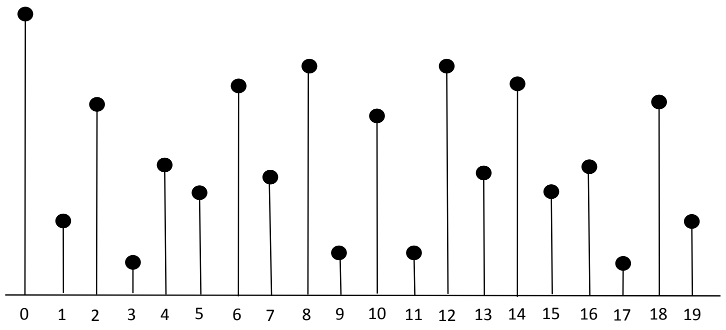
(length = 20)

frequency-domain $X[k]$, sparsity $K = 5$

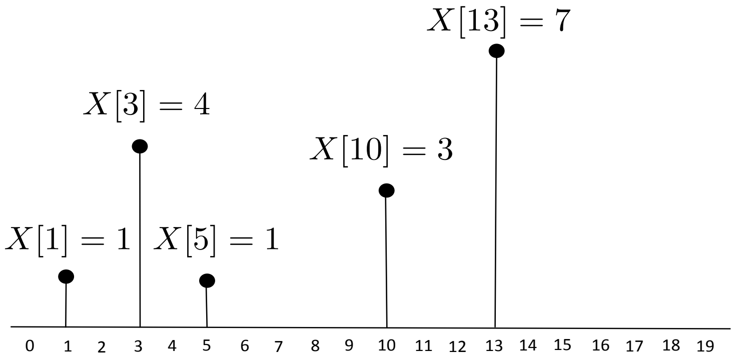


Main idea

time-domain $x[n]$, length $N = 20$



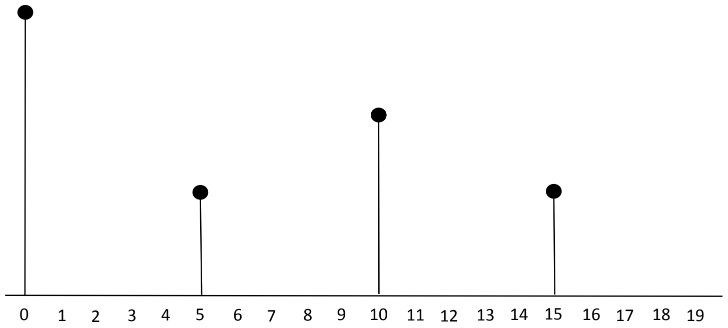
frequency-domain $X[k]$, sparsity $K = 5$



$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 20)

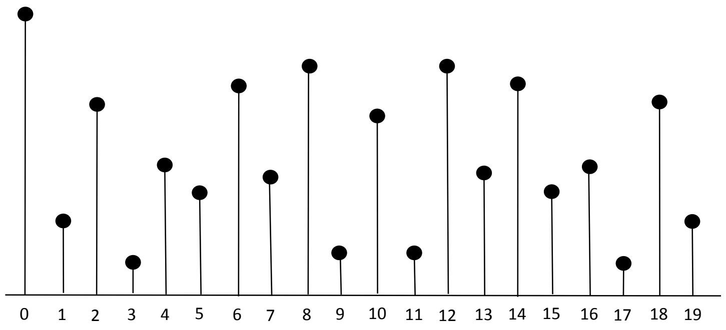
$\downarrow 5$

subsample by 5

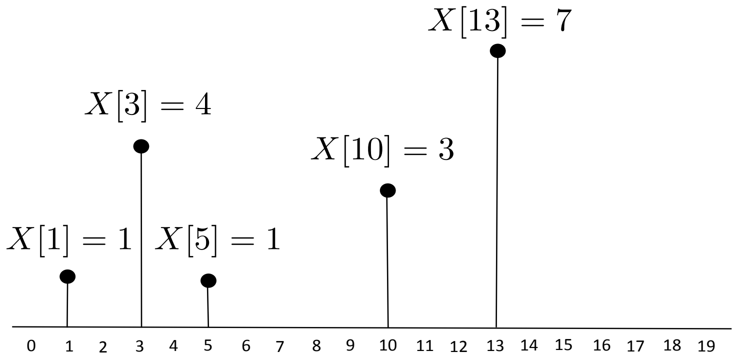


Main idea

time-domain $x[n]$, length $N = 20$



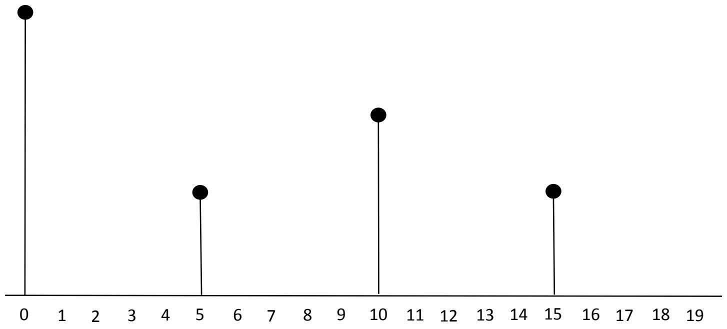
frequency-domain $X[k]$, sparsity $K = 5$



$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 20)

$\downarrow 5$

subsample by 5

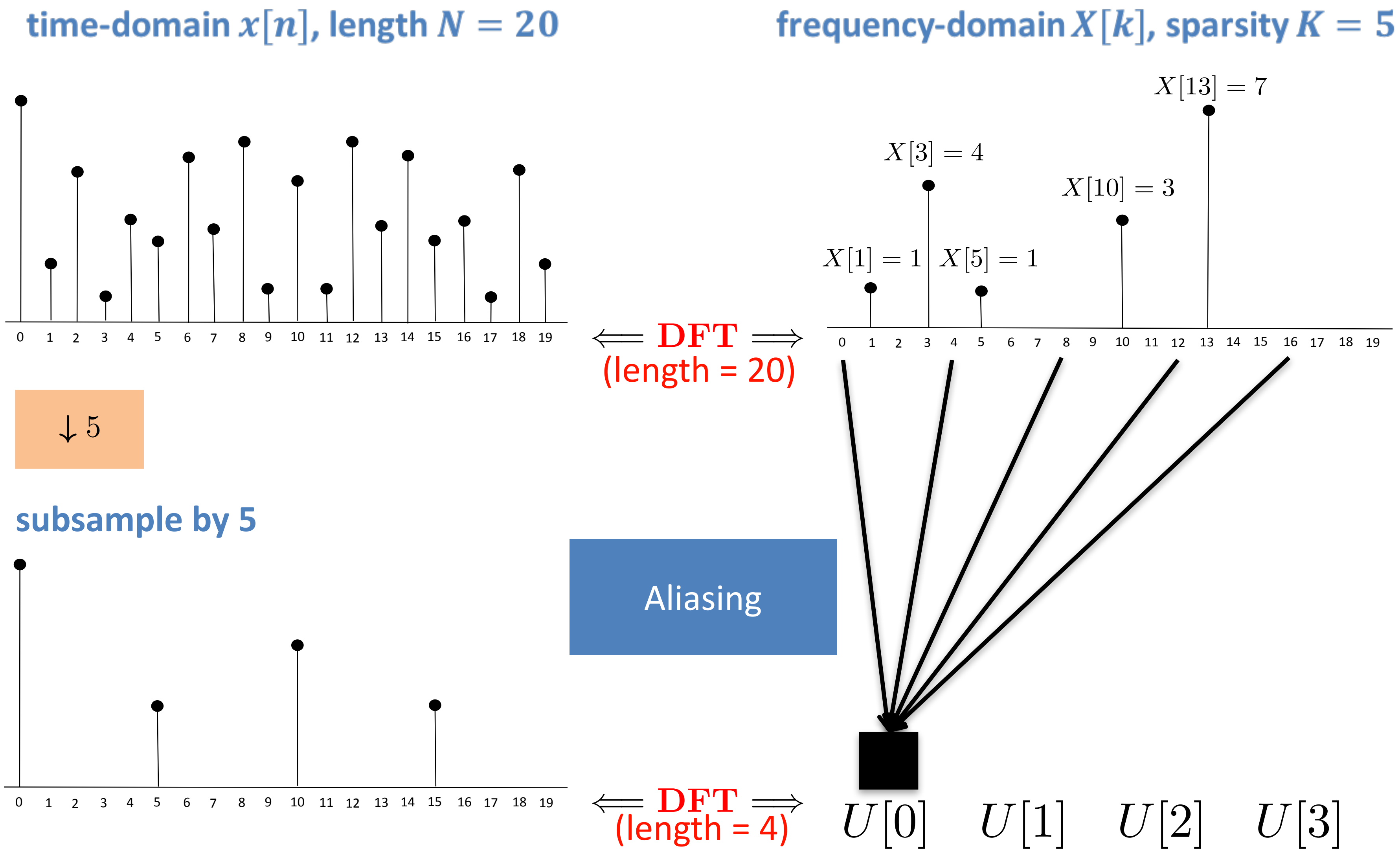


$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 4)

Our Measurements

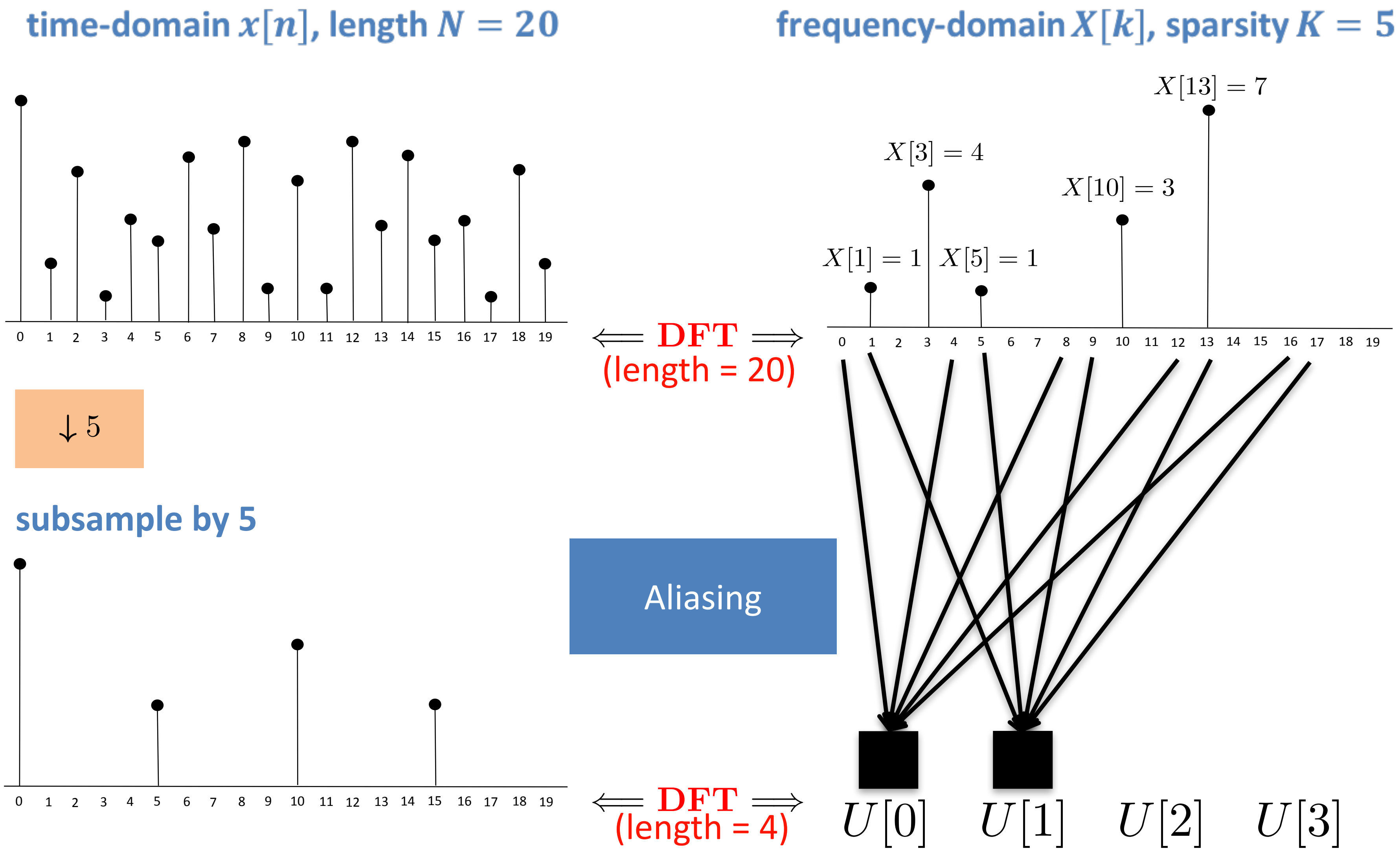
$U[0] \quad U[1] \quad U[2] \quad U[3]$

Main idea



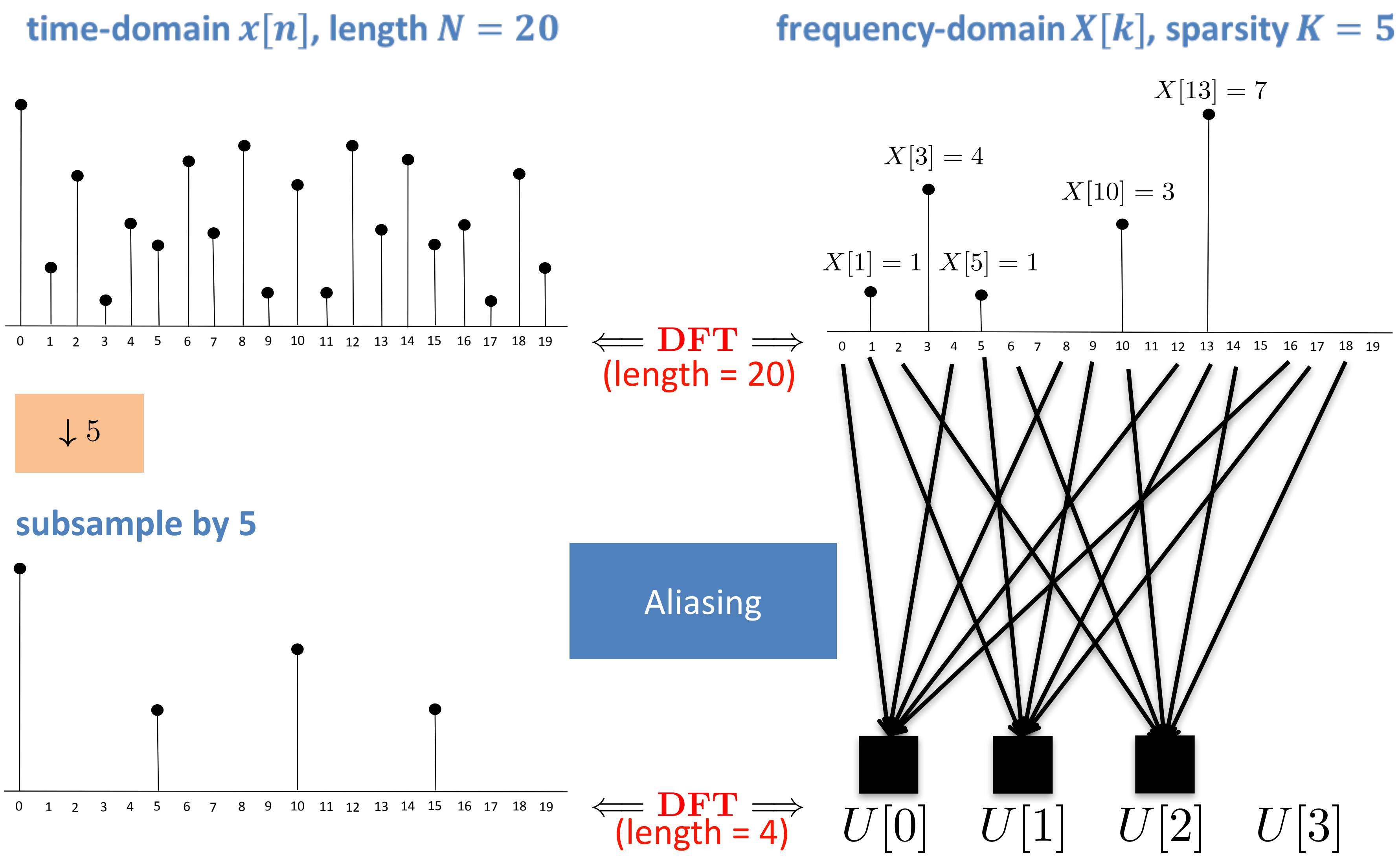
time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
Aliasing
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$ $U[1]$ $U[2]$ $U[3]$

Main idea



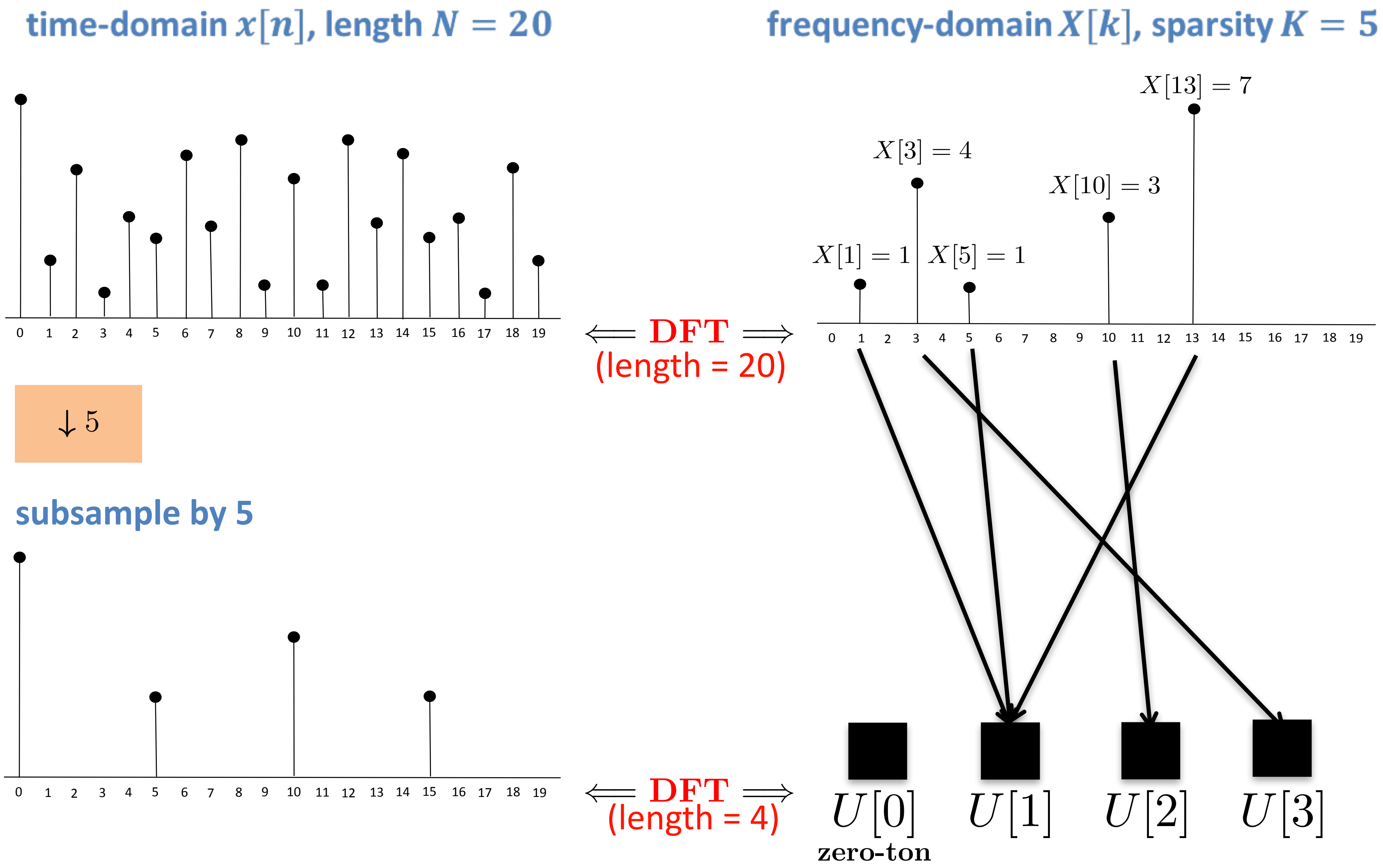
time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
Aliasing
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$ $U[1]$ $U[2]$ $U[3]$

Main idea



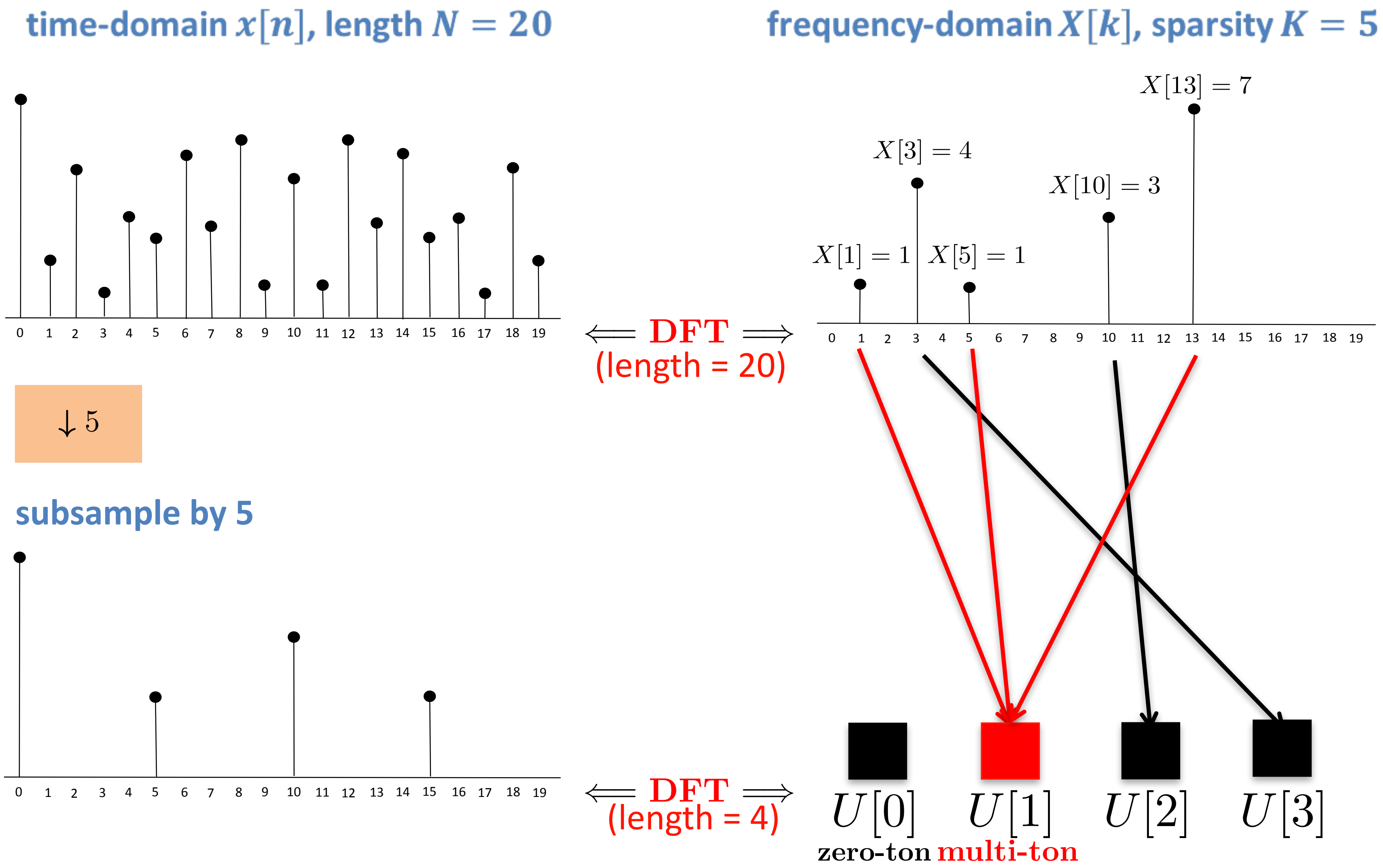
time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
Aliasing
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$
$U[1]$
$U[2]$
$U[3]$

Main idea

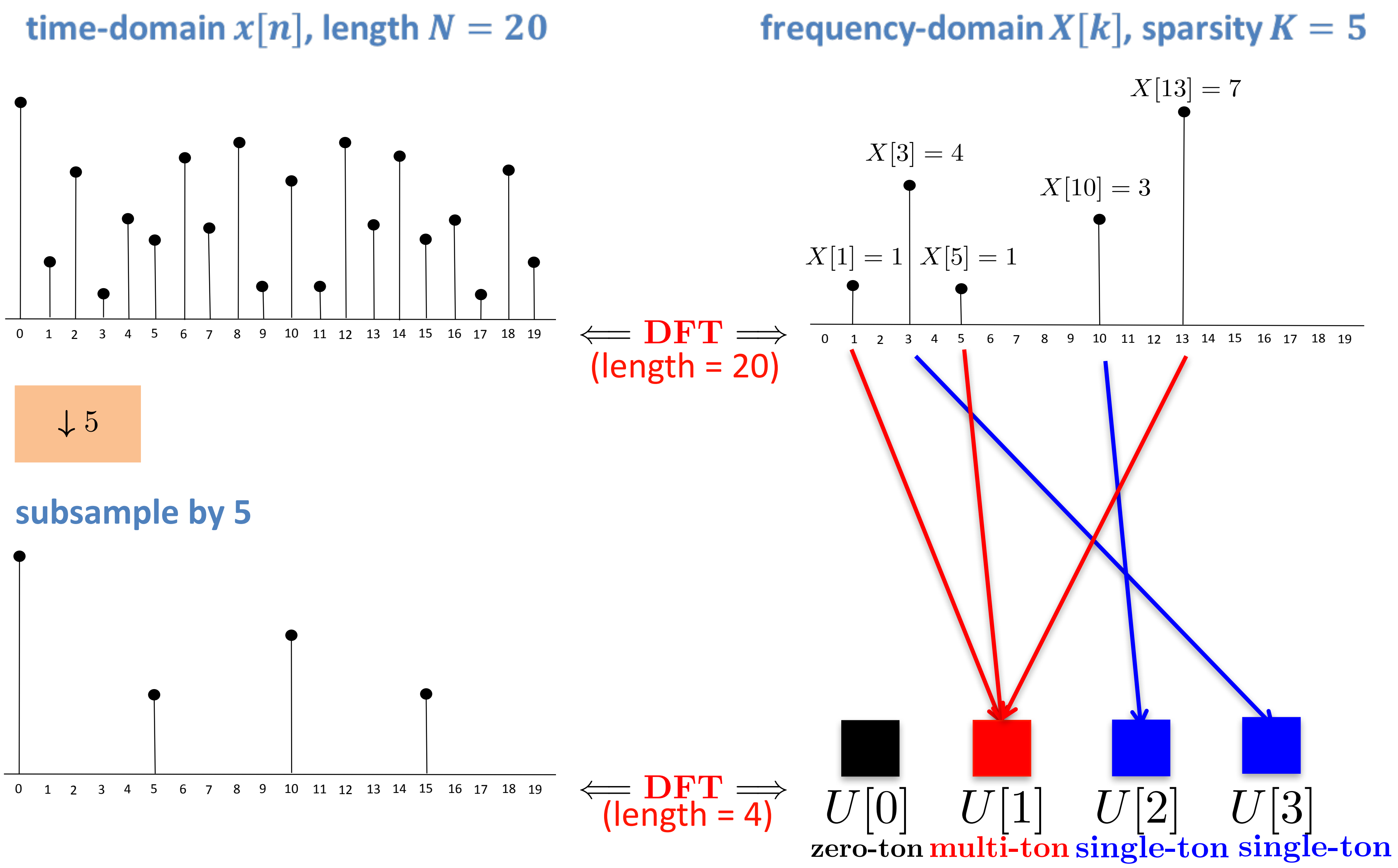


time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$
zero-ton
$U[1]$
$U[2]$
$U[3]$

Main idea

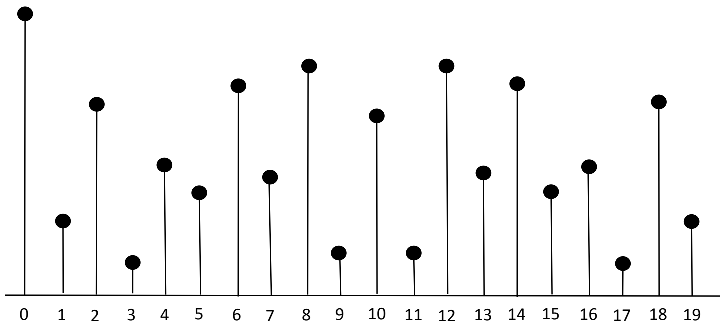

time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$ $U[1]$ $U[2]$ $U[3]$
zero-ton multi-ton

Main idea

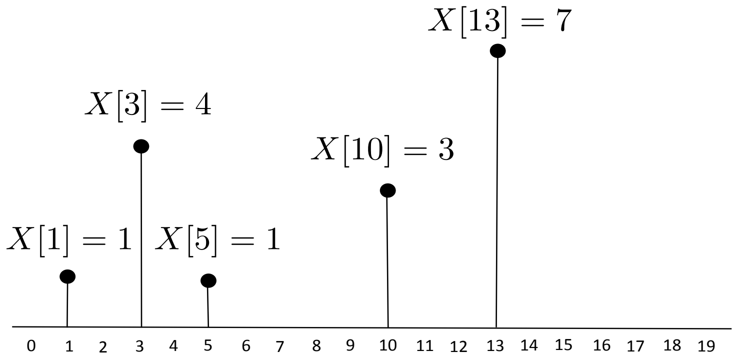

time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
subsample by 5
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U[0]$ $U[1]$ $U[2]$ $U[3]$
zero-ton multi-ton single-ton single-ton

Main idea

time-domain $x[n]$, length $N = 20$



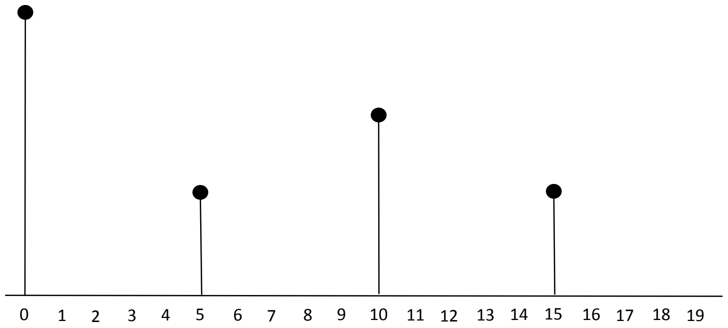
frequency-domain $X[k]$, sparsity $K = 5$



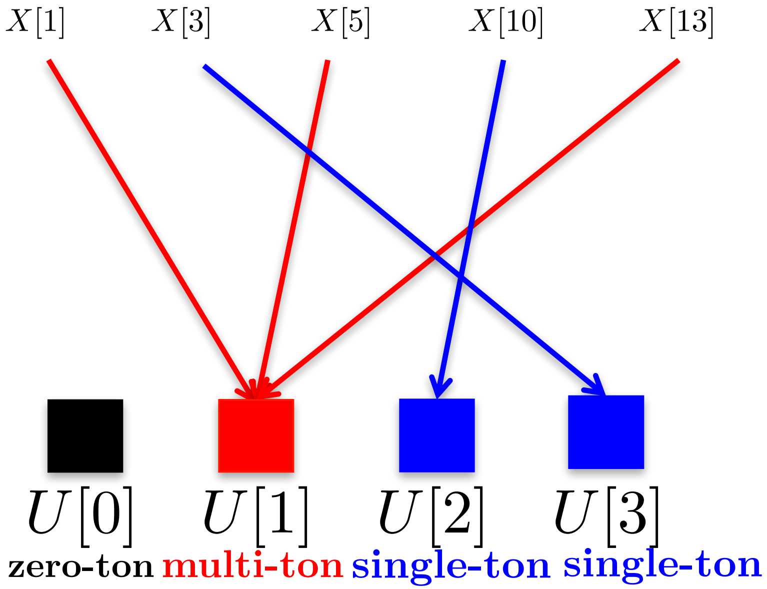
$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 20)

$\downarrow 5$

subsample by 5

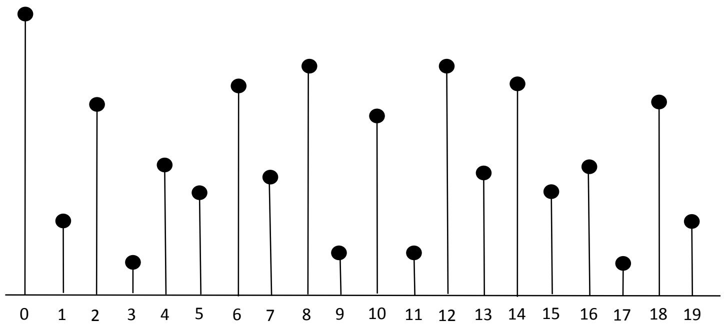


$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 4)

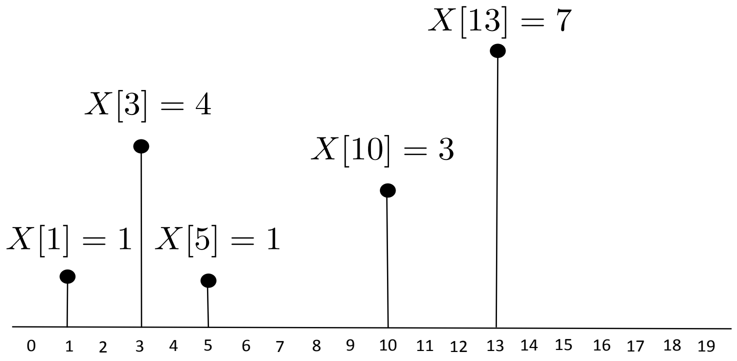


Main idea

time-domain $x[n]$, length $N = 20$



frequency-domain $X[k]$, sparsity $K = 5$

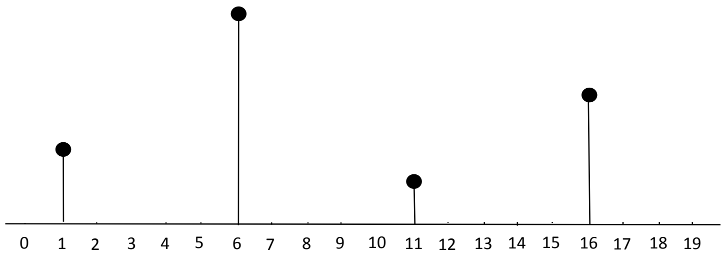


$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 20)

$\downarrow 5$

$X[1]$ $X[3]$ $X[5]$ $X[10]$ $X[13]$

shift and subsample by 5



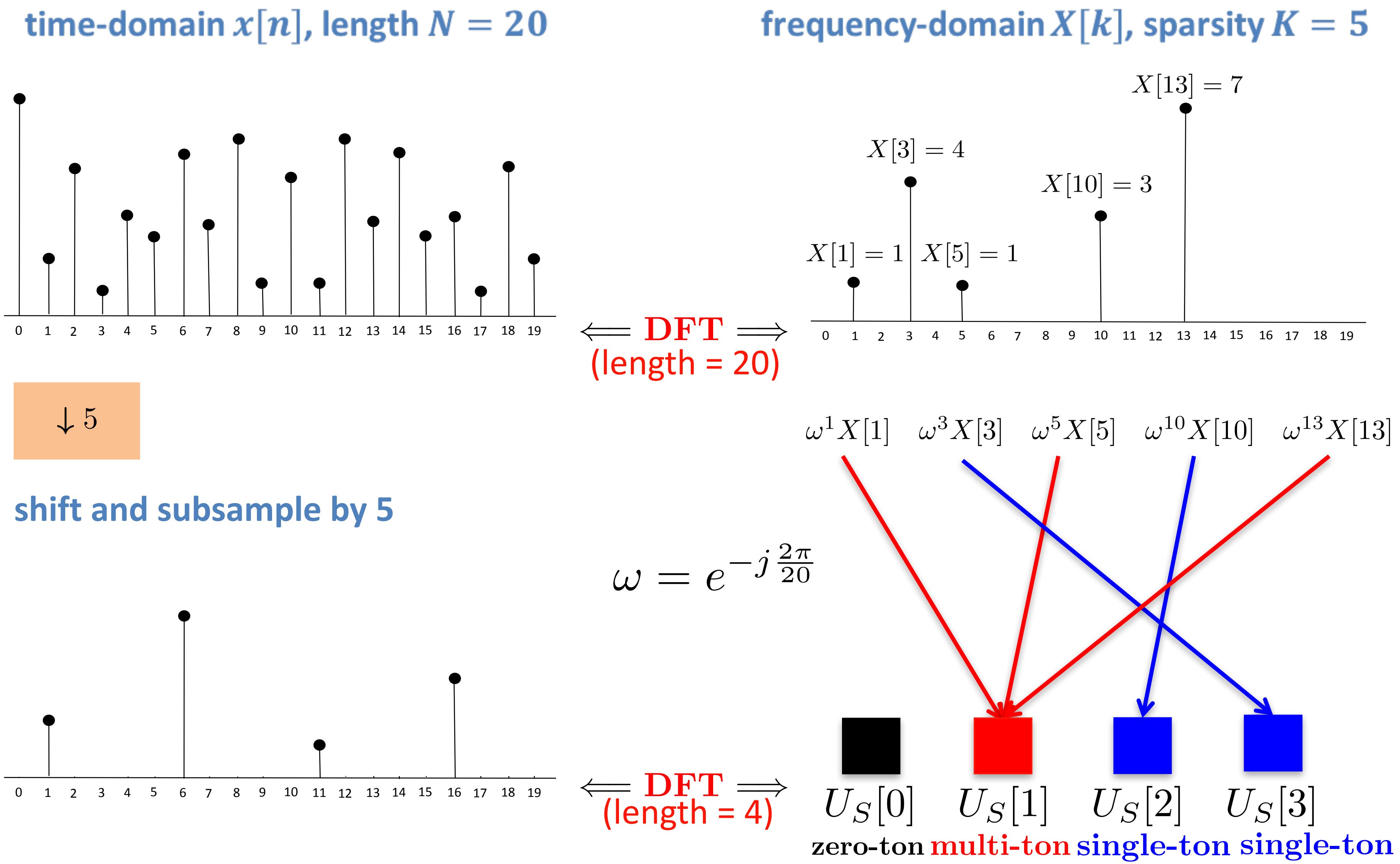
$\Longleftarrow$ **DFT** $\Longrightarrow$
(length = 4)

Our Measurements

$U_S[0]$ $U_S[1]$ $U_S[2]$ $U_S[3]$

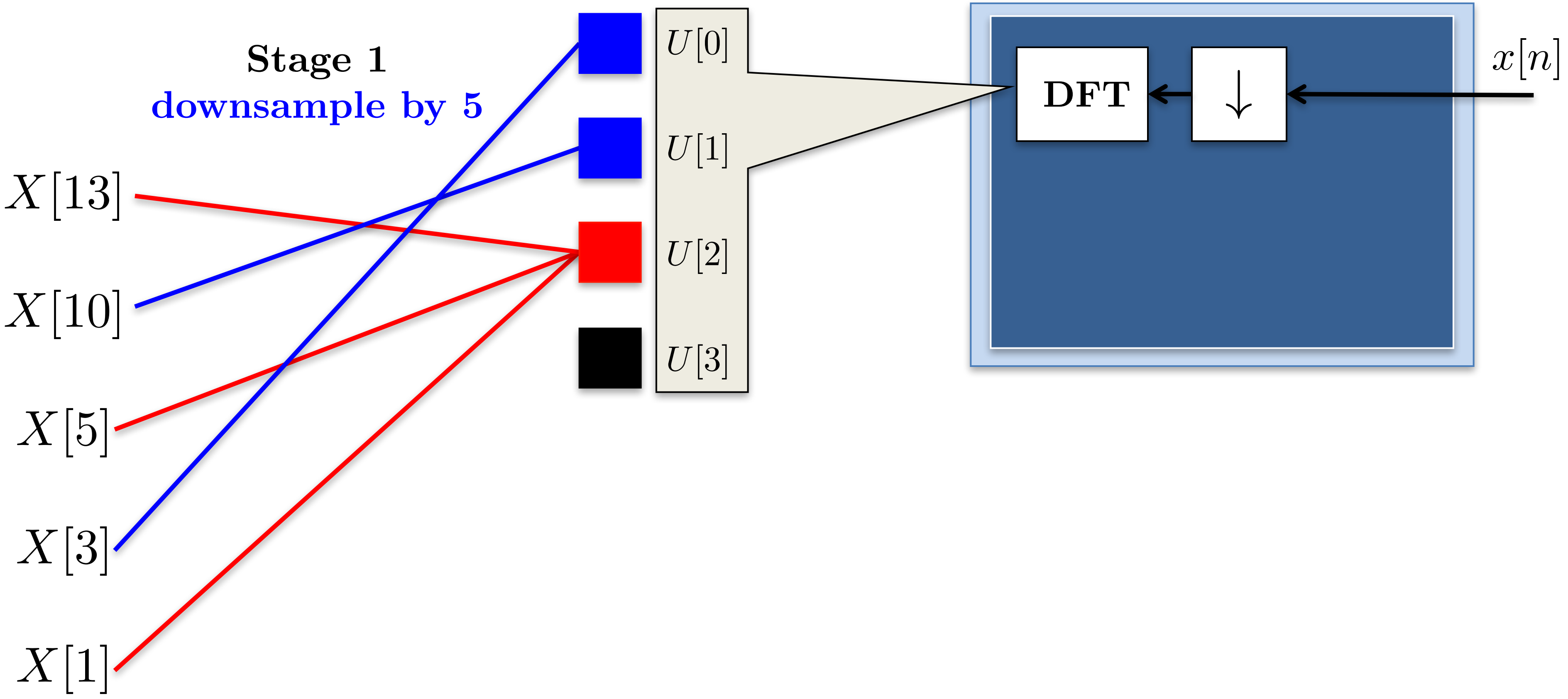
subscript U_S suggests **shift**

Main idea

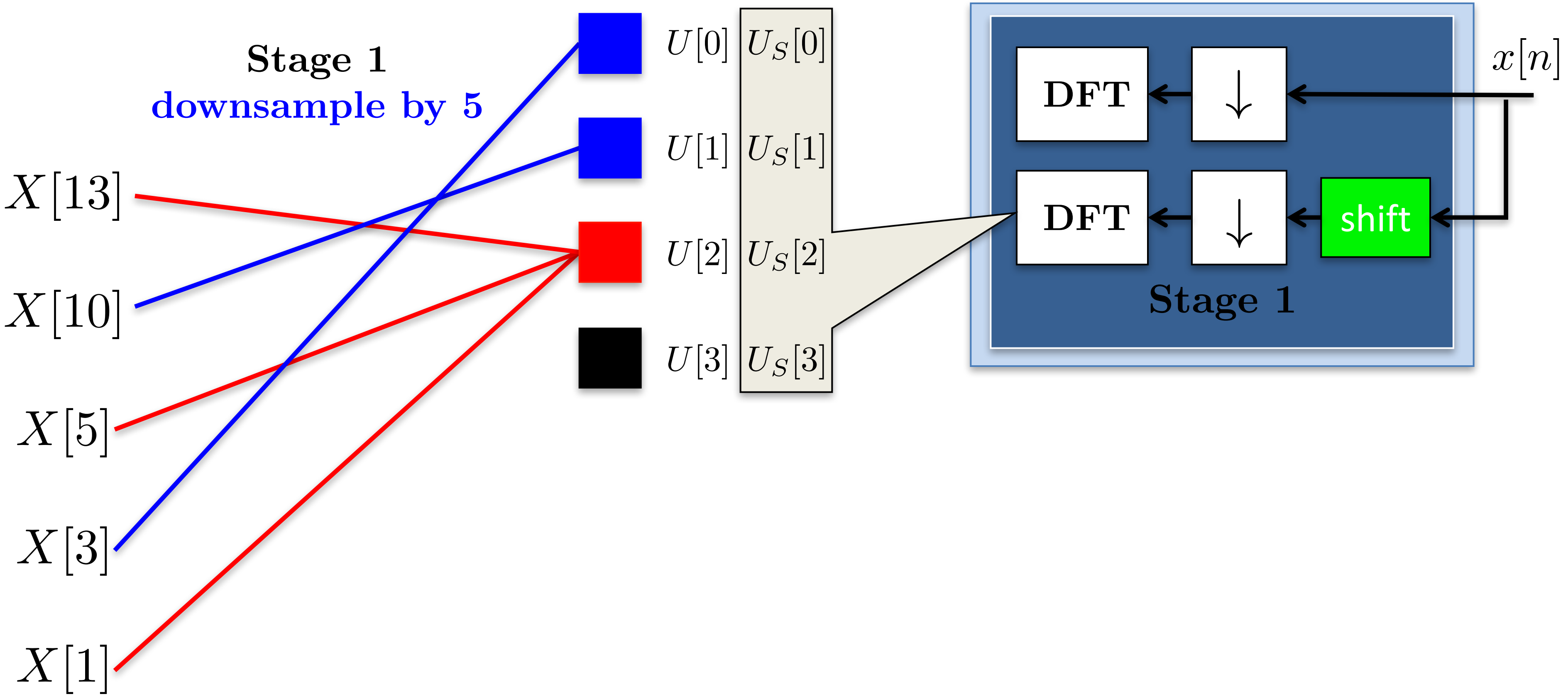


time-domain $x[n]$, length $N = 20$
frequency-domain $X[k]$, sparsity $K = 5$
$X[13] = 7$
$X[3] = 4$
$X[10] = 3$
$X[1] = 1$
$X[5] = 1$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 20)
$\downarrow 5$
shift and subsample by 5
$\omega = e^{-j\frac{2\pi}{20}}$
$\omega^1 X[1]$ $\omega^3 X[3]$ $\omega^5 X[5]$ $\omega^{10} X[10]$ $\omega^{13} X[13]$
$\Longleftarrow$ DFT $\Longrightarrow$
(length = 4)
$U_S[0]$ $U_S[1]$ $U_S[2]$ $U_S[3]$
zero-ton multi-ton single-ton single-ton

Main Idea

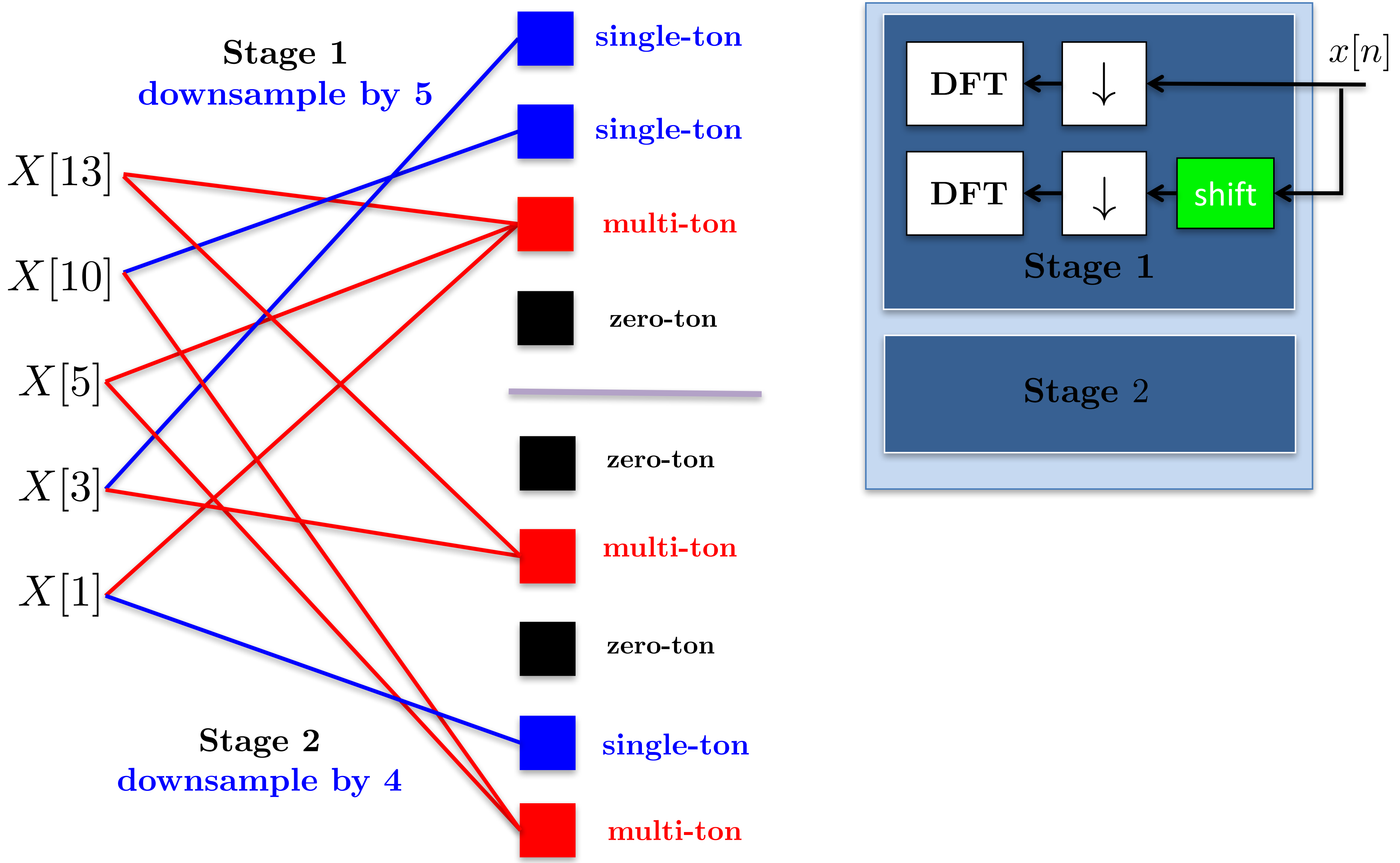


Main Idea



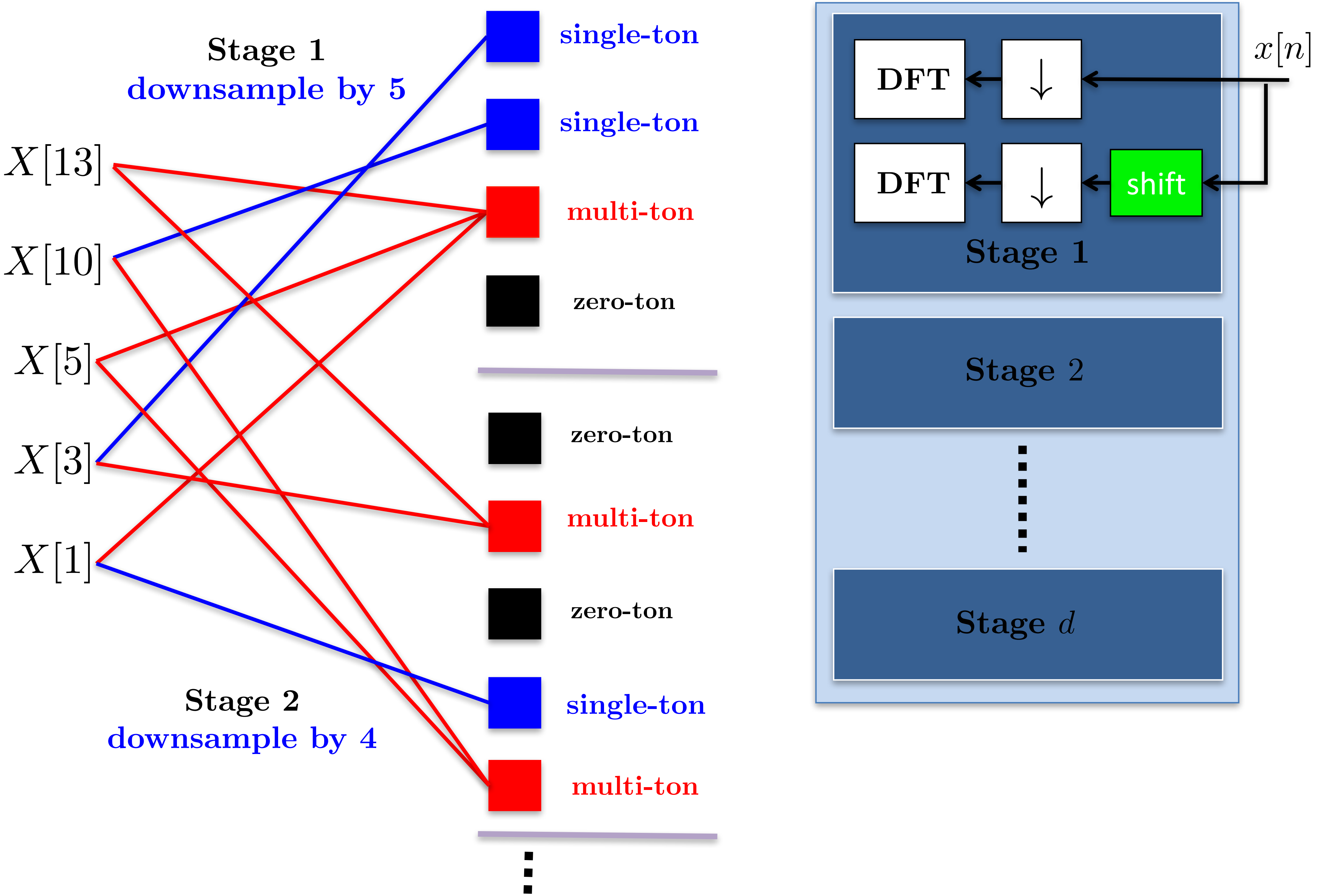
Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
$U[0]$ $U_S[0]$
$U[1]$ $U_S[1]$
$U[2]$ $U_S[2]$
$U[3]$ $U_S[3]$
DFT
DFT
shift
Stage 1
$x[n]$

Main Idea



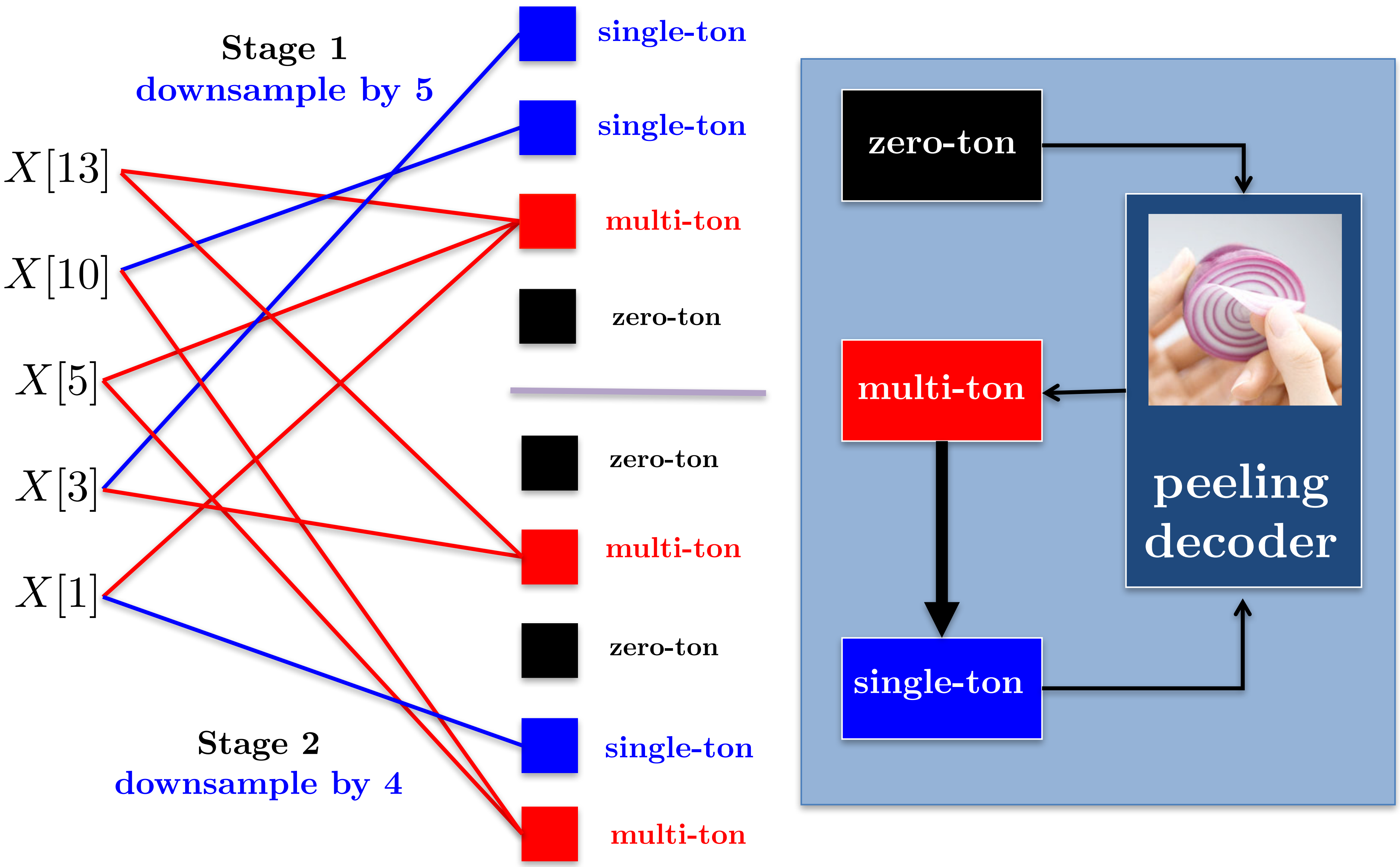
Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
single-ton
multi-ton
zero-ton
zero-ton
multi-ton
zero-ton
single-ton
multi-ton
Stage 2
downsample by 4
DFT
DFT
shift
Stage 1
Stage 2
$x[n]$

Main Idea

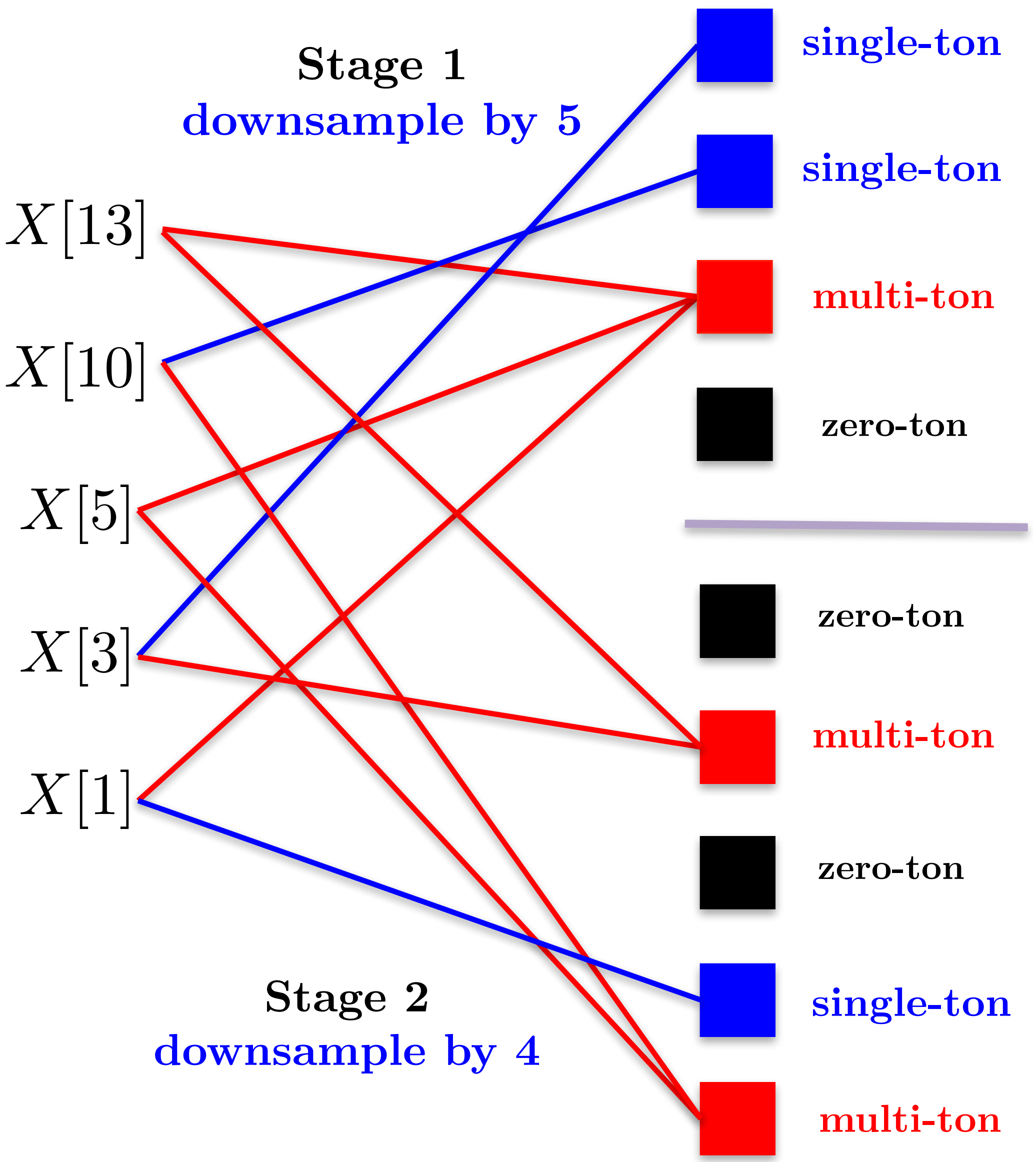


Stage 1
downsample by 5
single-ton
single-ton
multi-ton
zero-ton
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
zero-ton
multi-ton
zero-ton
single-ton
multi-ton
Stage 2
downsample by 4
DFT
DFT
shift
$x[n]$
Stage 1
Stage 2
Stage d

Main Idea

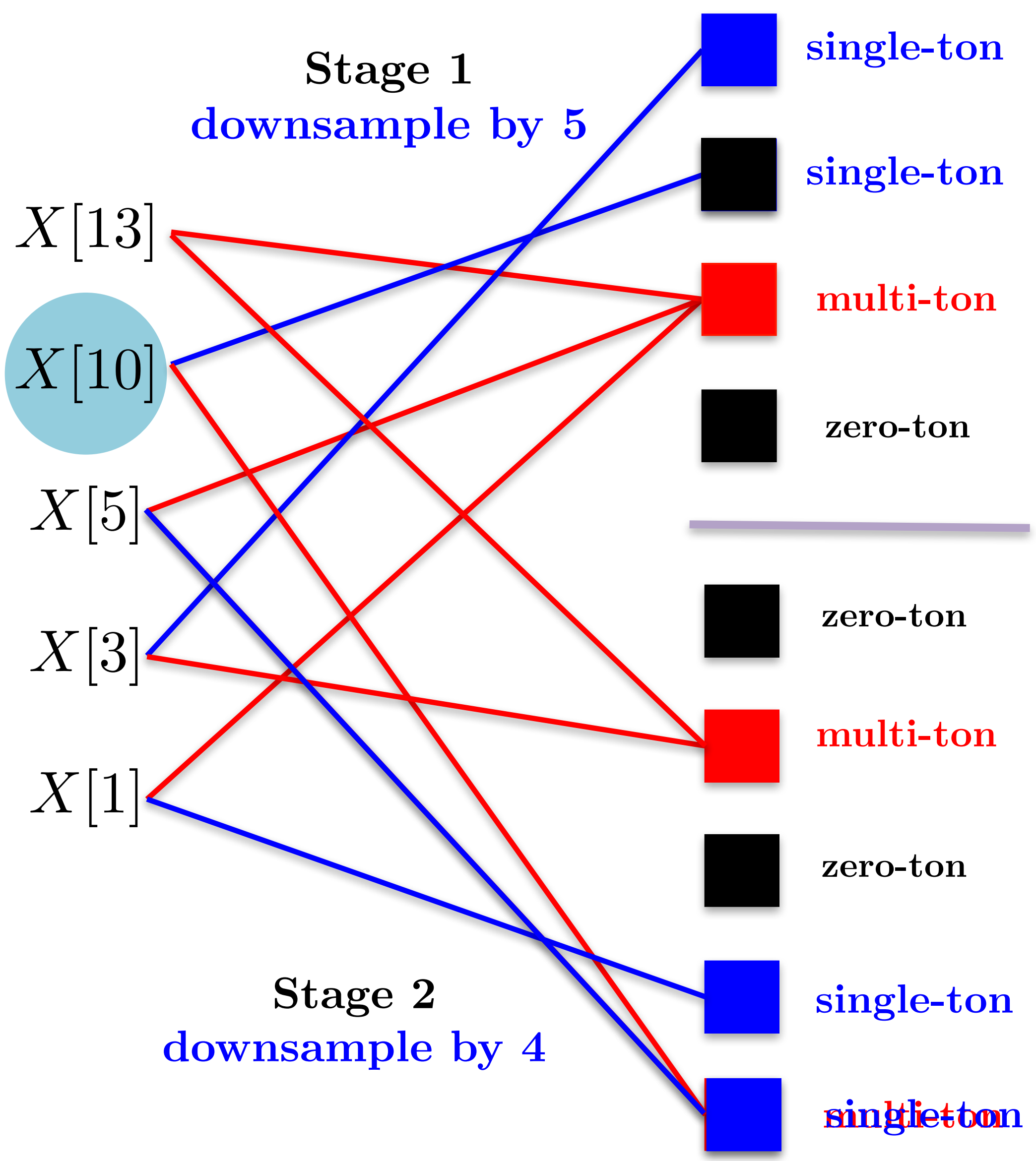

Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
single-ton
multi-ton
zero-ton
zero-ton
multi-ton
zero-ton
single-ton
multi-ton
Stage 2
downsample by 4
zero-ton
multi-ton
single-ton
peeling decoder

Main Idea


Stage 1
downsample by 5
Stage 2
downsample by 4
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
single-ton
multi-ton
zero-ton
zero-ton
multi-ton
zero-ton
single-ton
multi-ton

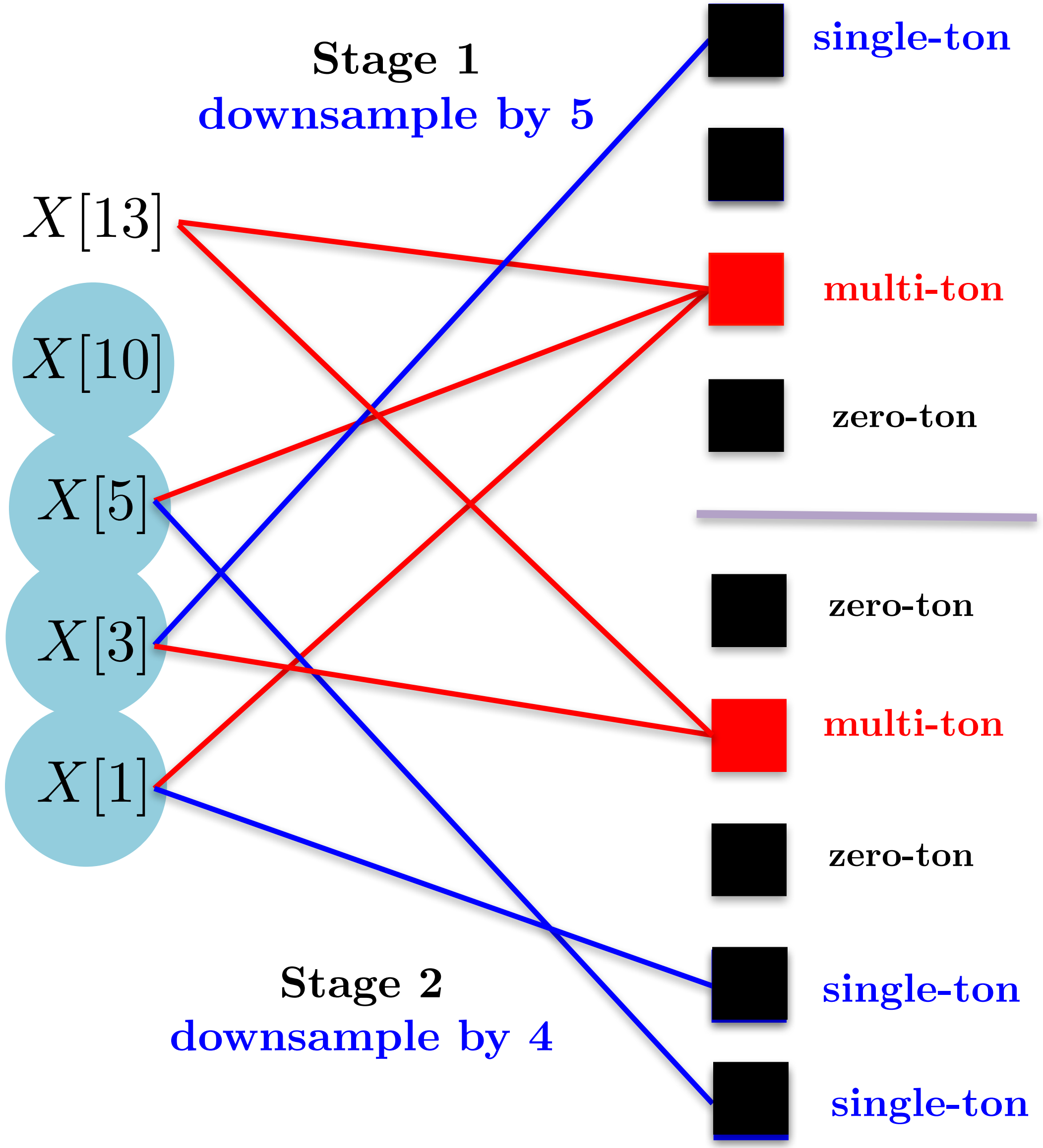

peeling decoder

Main Idea


Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
single-ton
multi-ton
zero-ton
zero-ton
multi-ton
zero-ton
single-ton
single-ton
multi-ton
Stage 2
downsample by 4

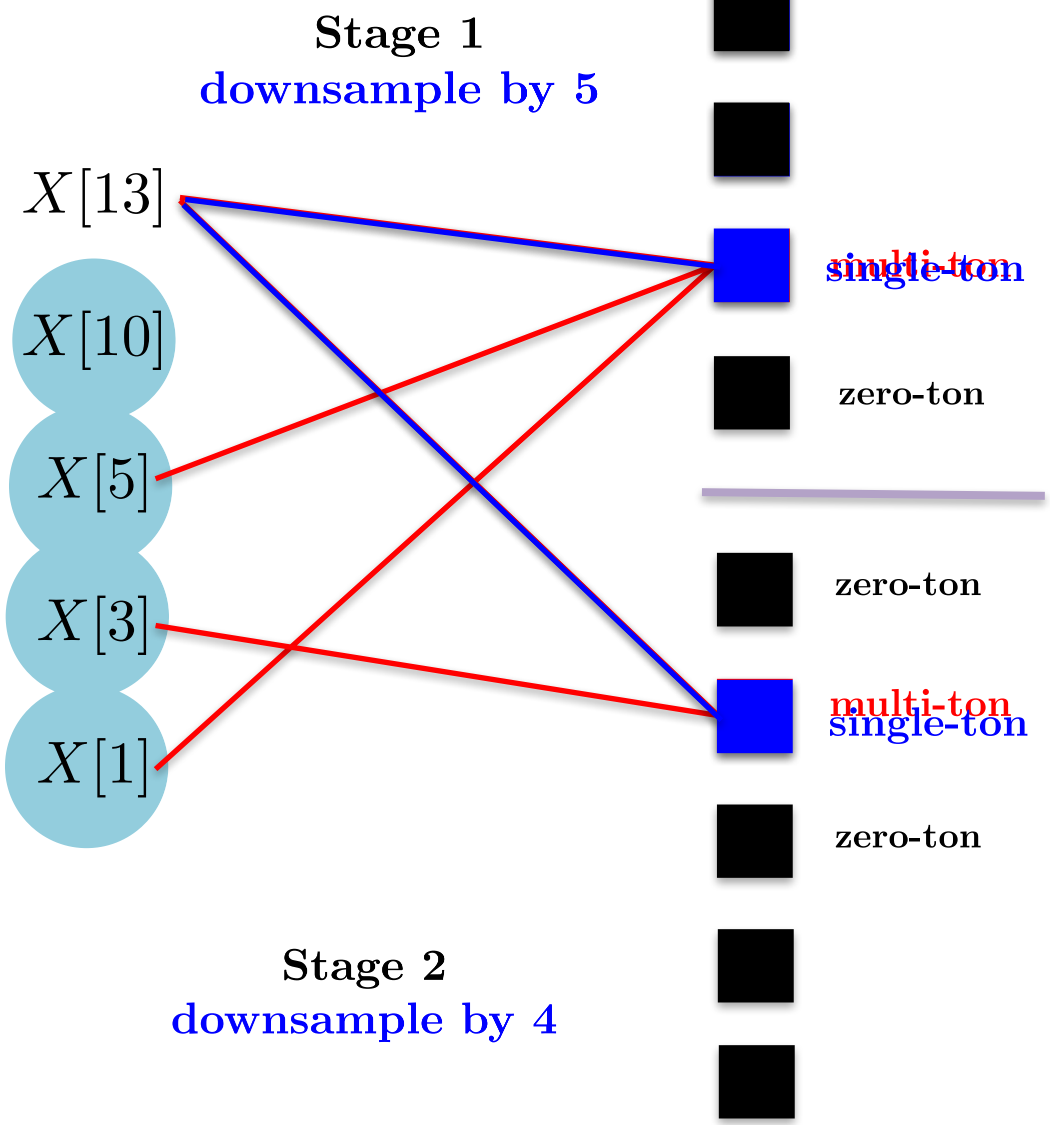

peeling decoder

Main Idea


Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
multi-ton
zero-ton
zero-ton
multi-ton
zero-ton
single-ton
single-ton
Stage 2
downsample by 4

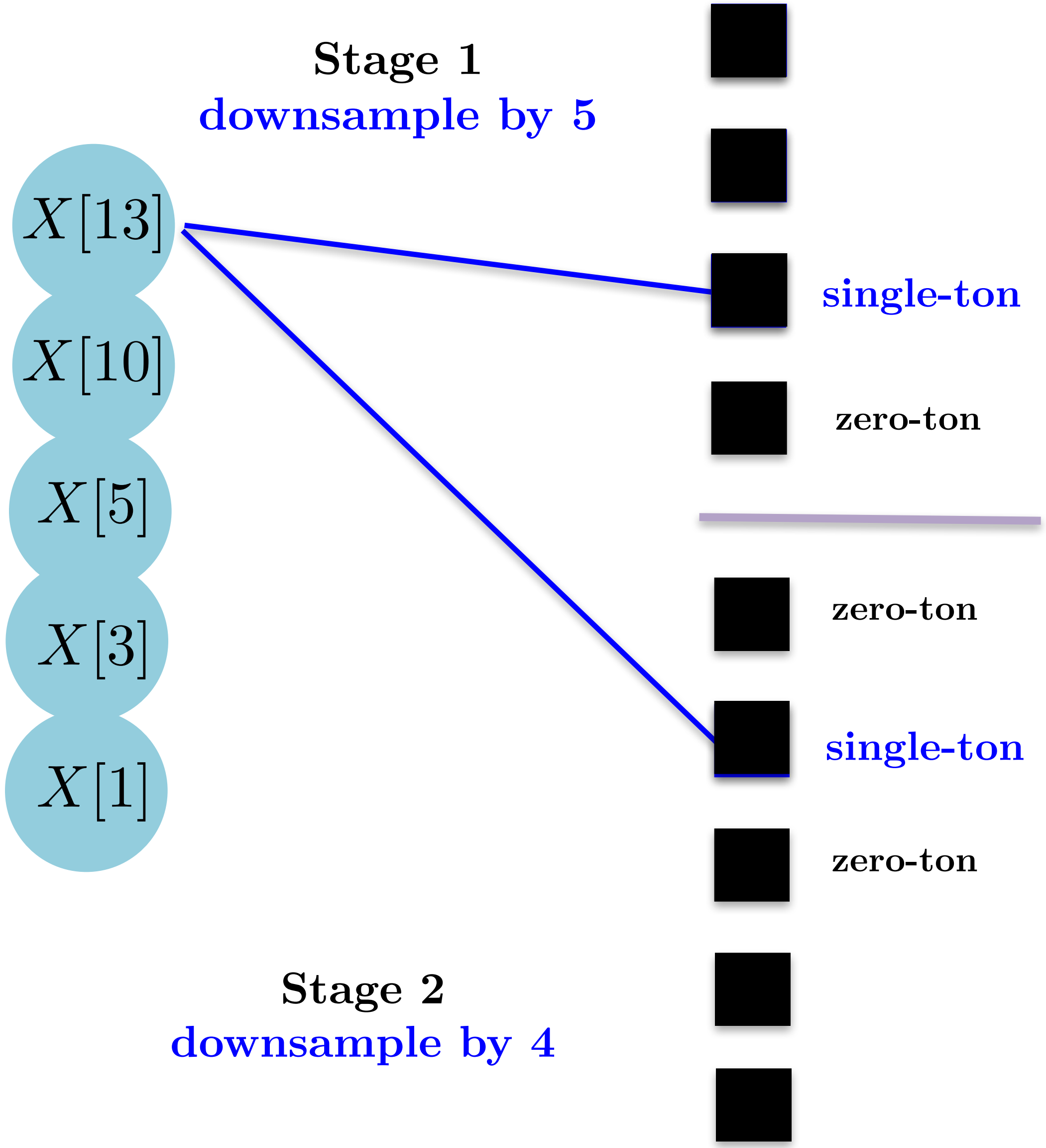

peeling decoder

Main Idea


Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
multi-ton
single-ton
zero-ton
zero-ton
multi-ton
single-ton
zero-ton
Stage 2
downsample by 4


peeling decoder

Main Idea


Stage 1
downsample by 5
$X[13]$
$X[10]$
$X[5]$
$X[3]$
$X[1]$
single-ton
zero-ton
zero-ton
single-ton
zero-ton
Stage 2
downsample by 4


peeling decoder

Main Idea

Stage 1

downsample by 5

$X[13]$

$X[10]$

$X[5]$

$X[3]$

$X[1]$

zero-ton

zero-ton

zero-ton

zero-ton

zero-ton

zero-ton

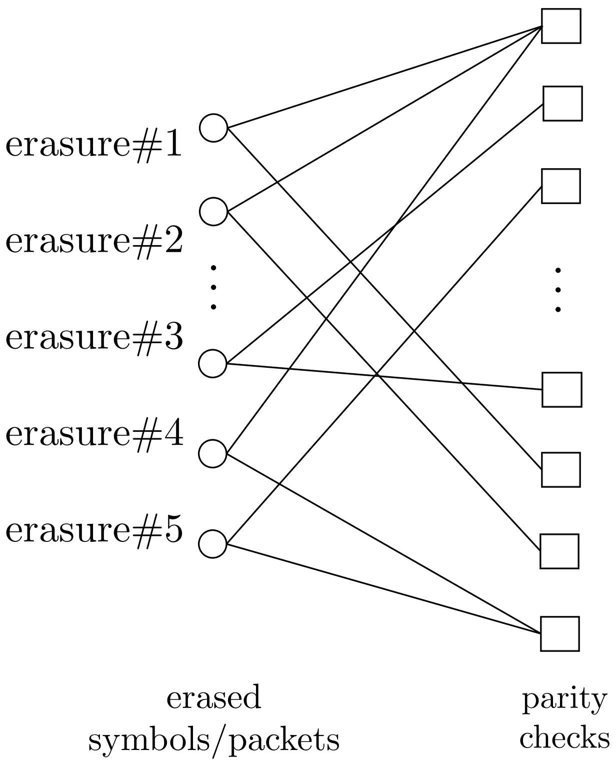
zero-ton

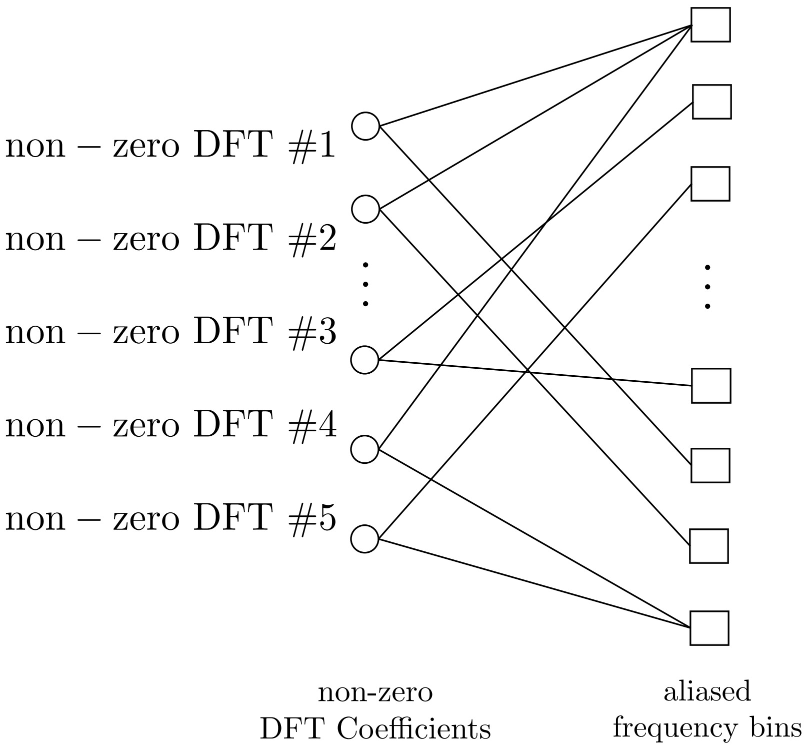
zero-ton

peeling decoder

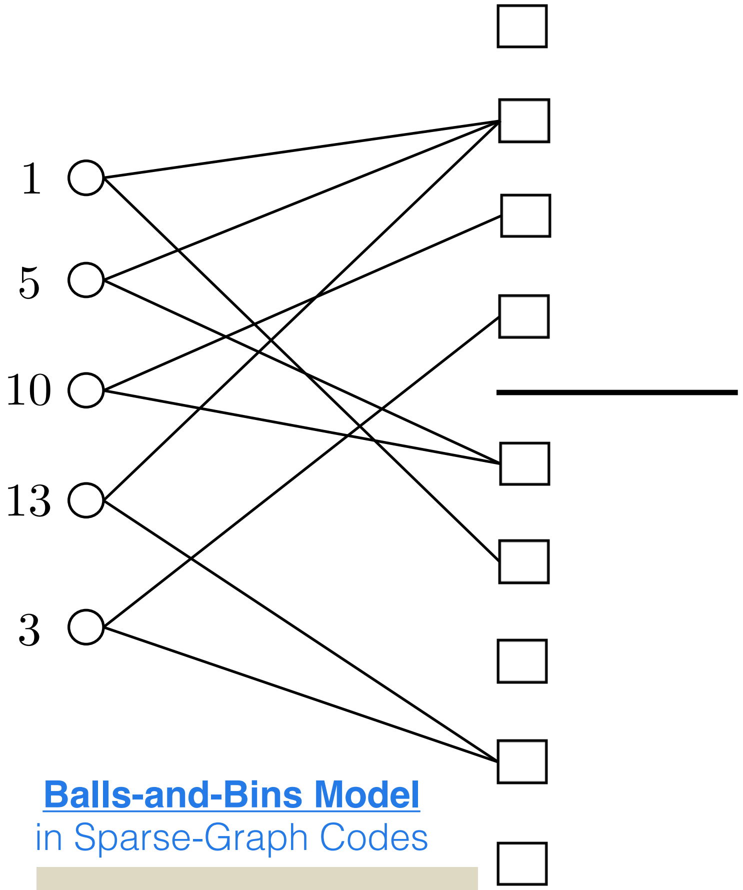
How do we induce good graphs that will work?

Sparse DFT Computation = Decoding over Sparse Graphs

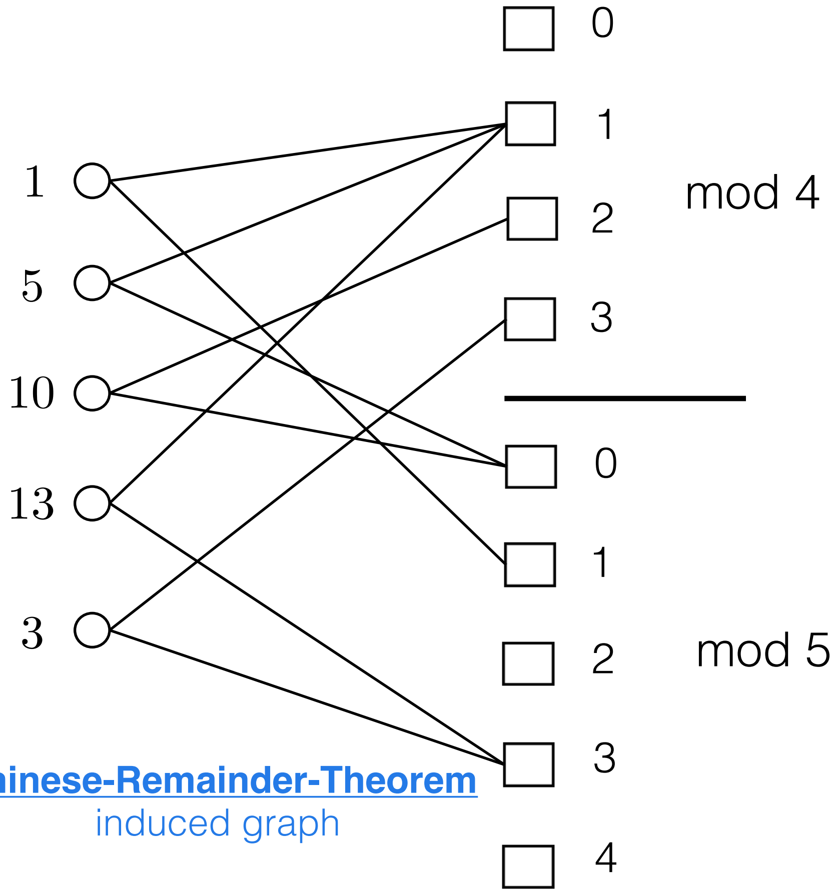




CRT-guided Subsampling Induces Good Graphs



Balls-and-Bins Model
in Sparse-Graph Codes



Chinese-Remainder-Theorem
induced graph

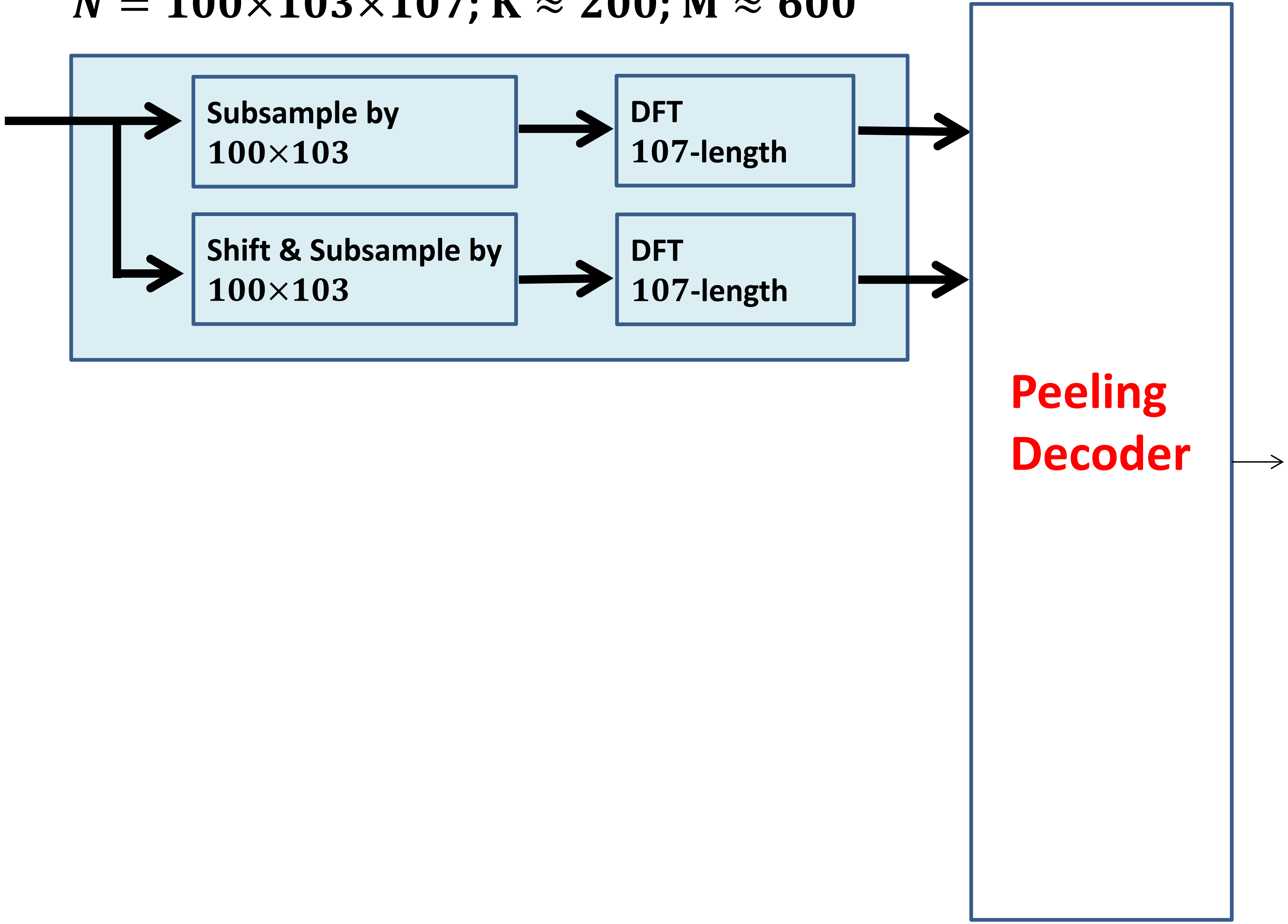
LDPC codes

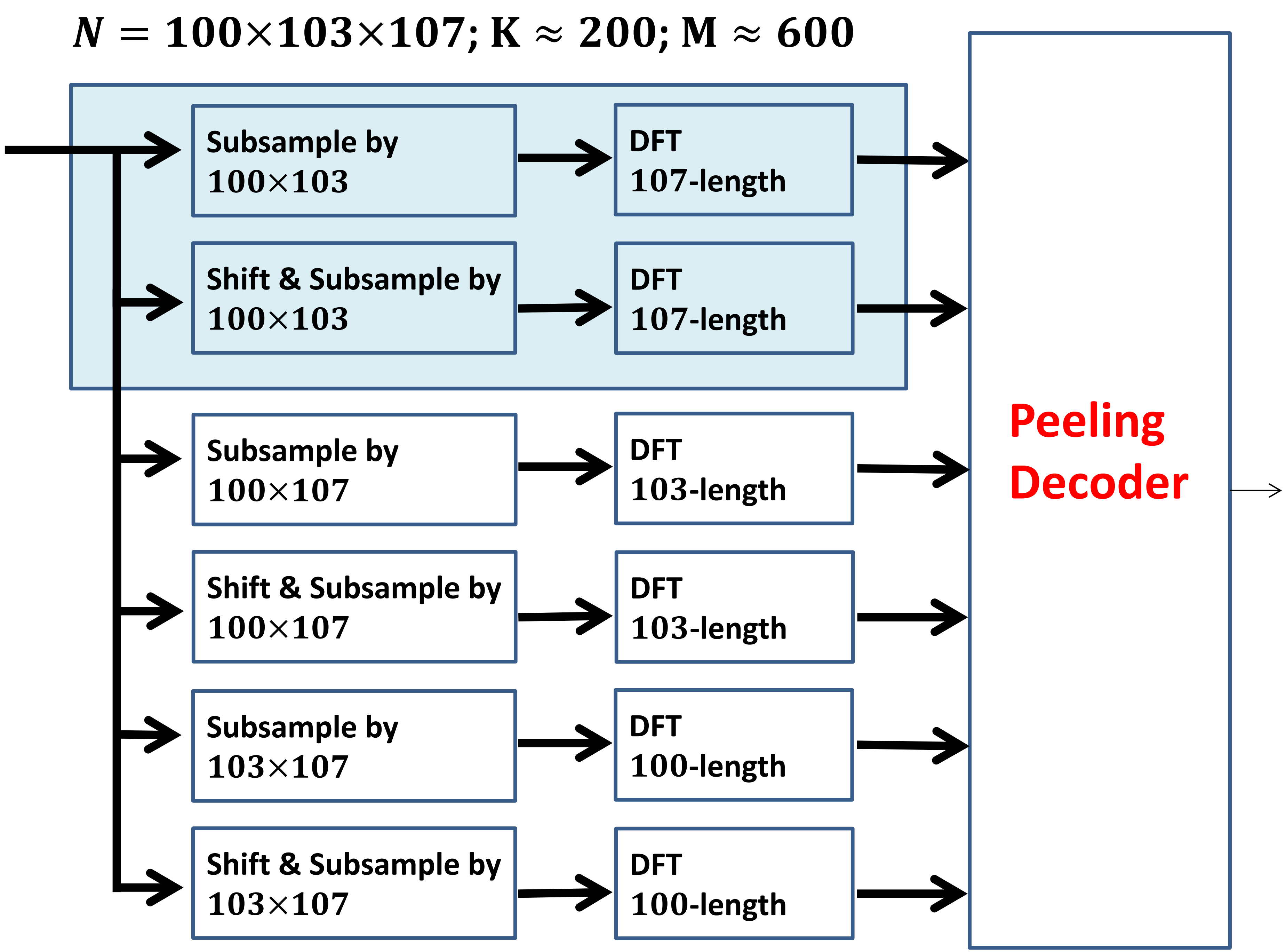
Density evolution, Martingales, Expander graph theory

Chinese-Remainder-Theorem:
A number between 0-19 is uniquely represented by its remainders modulo (4,5)

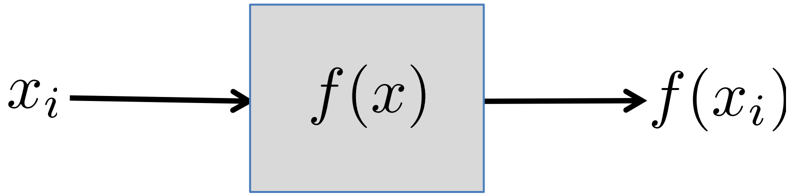
> The two graph ensembles are **identical.**

$$N = 100\times103\times107;\ \mathrm{K} \approx 200;\ \mathrm{M} \approx 600$$

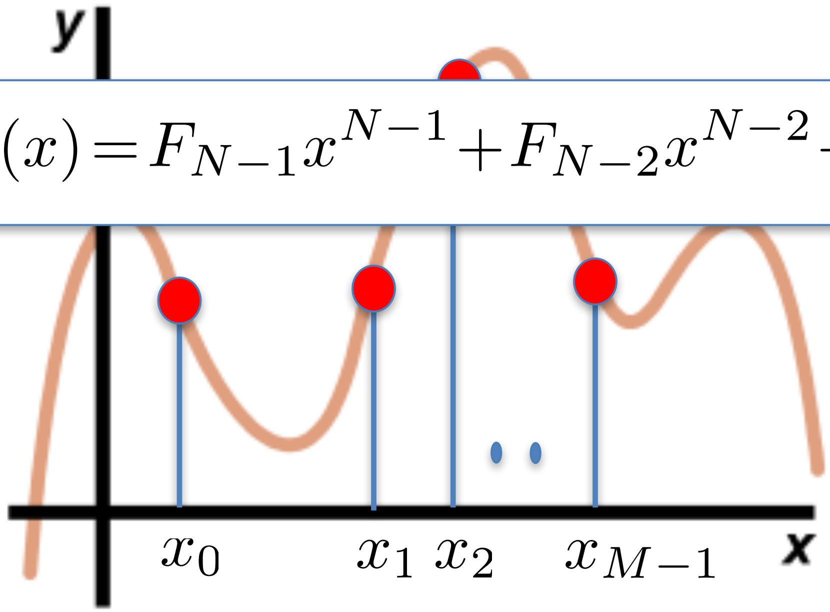



$N = 100\times103\times107; K \approx 200; M \approx 600$
Subsample by 100×103
DFT 107-length
Shift & Subsample by 100×103
DFT 107-length
Subsample by 100×107
DFT 103-length
Shift & Subsample by 100×107
DFT 103-length
Subsample by 103×107
DFT 100-length
Shift & Subsample by 103×107
DFT 100-length
Peeling Decoder

Sparse polynomial learning



E.g. deg. N=1 million
Sparsity K= 200



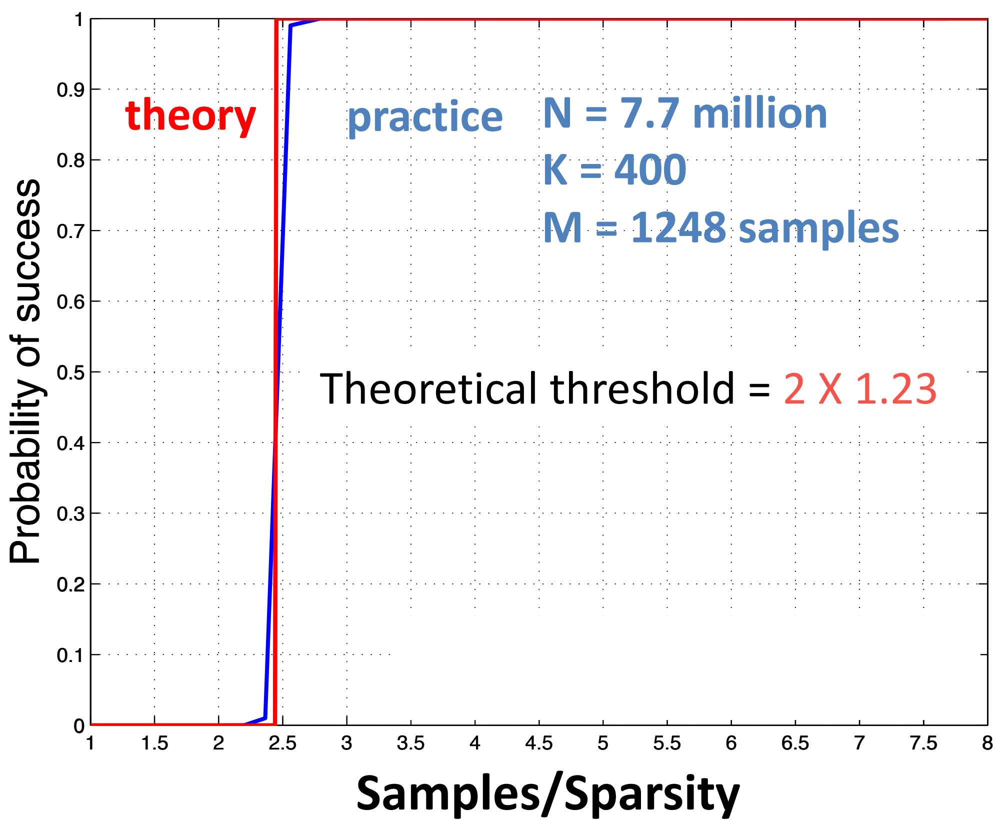
$$f(x) = F_{N-1}x^{N-1} + F_{N-2}x^{N-2} + \cdots + F_0$$

 # evals. $M = 616$

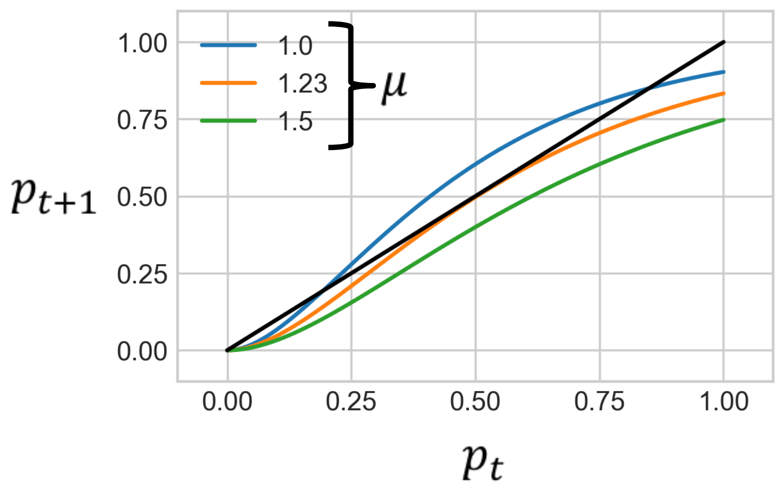
- **N=100x103x107**
- **K≅ $\mathbf{N^{1/3}}$**
- ***M=2*(100+103+107)-4***

What if only (very few) K of the N **polynomial coeffs.$\{F_n\}$ are non-zero?**

Noiseless setting: Theory vs. practice



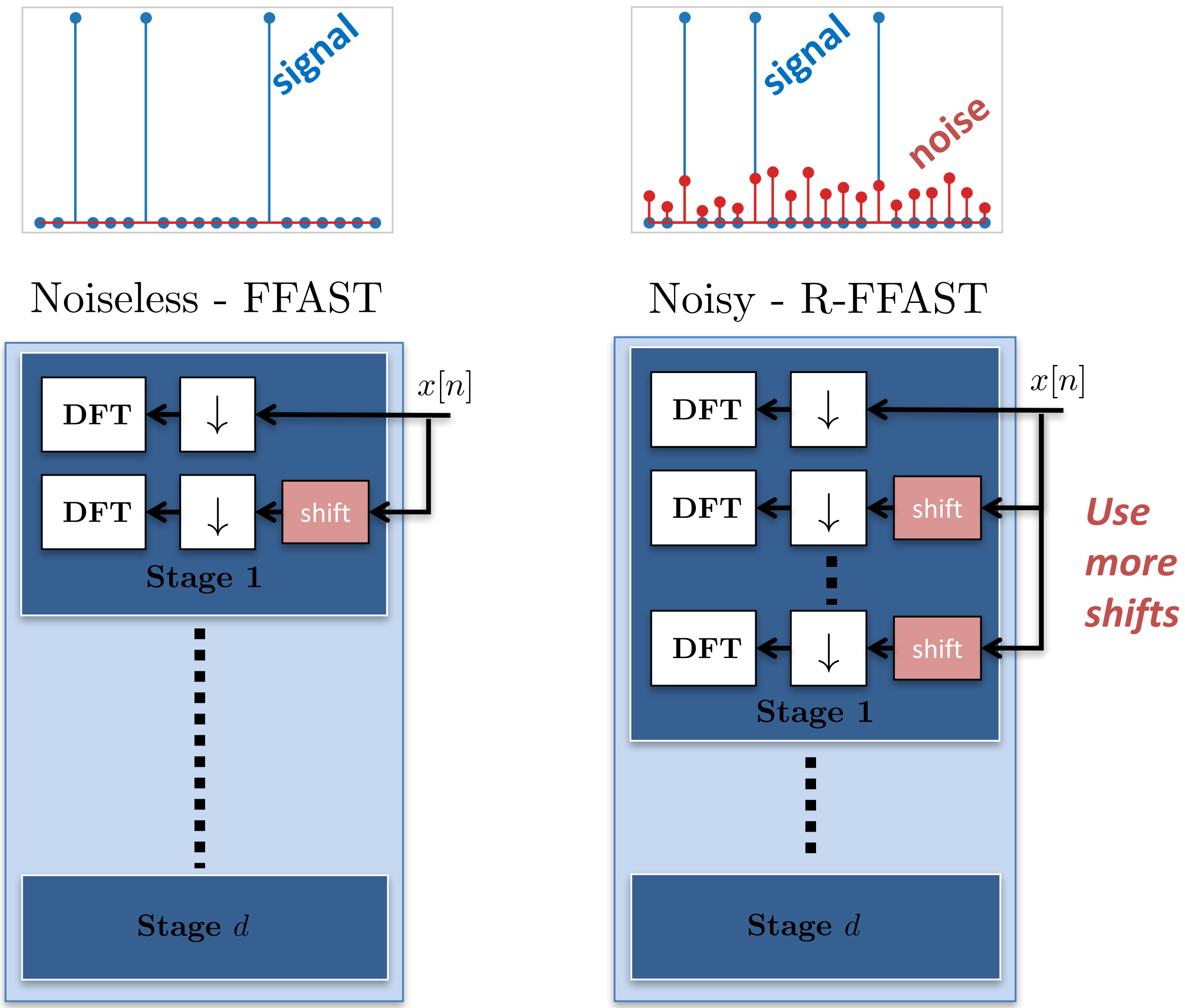
$$p_{t+1} = \left(1 - e^{-3p_t/\mu}\right)^2$$



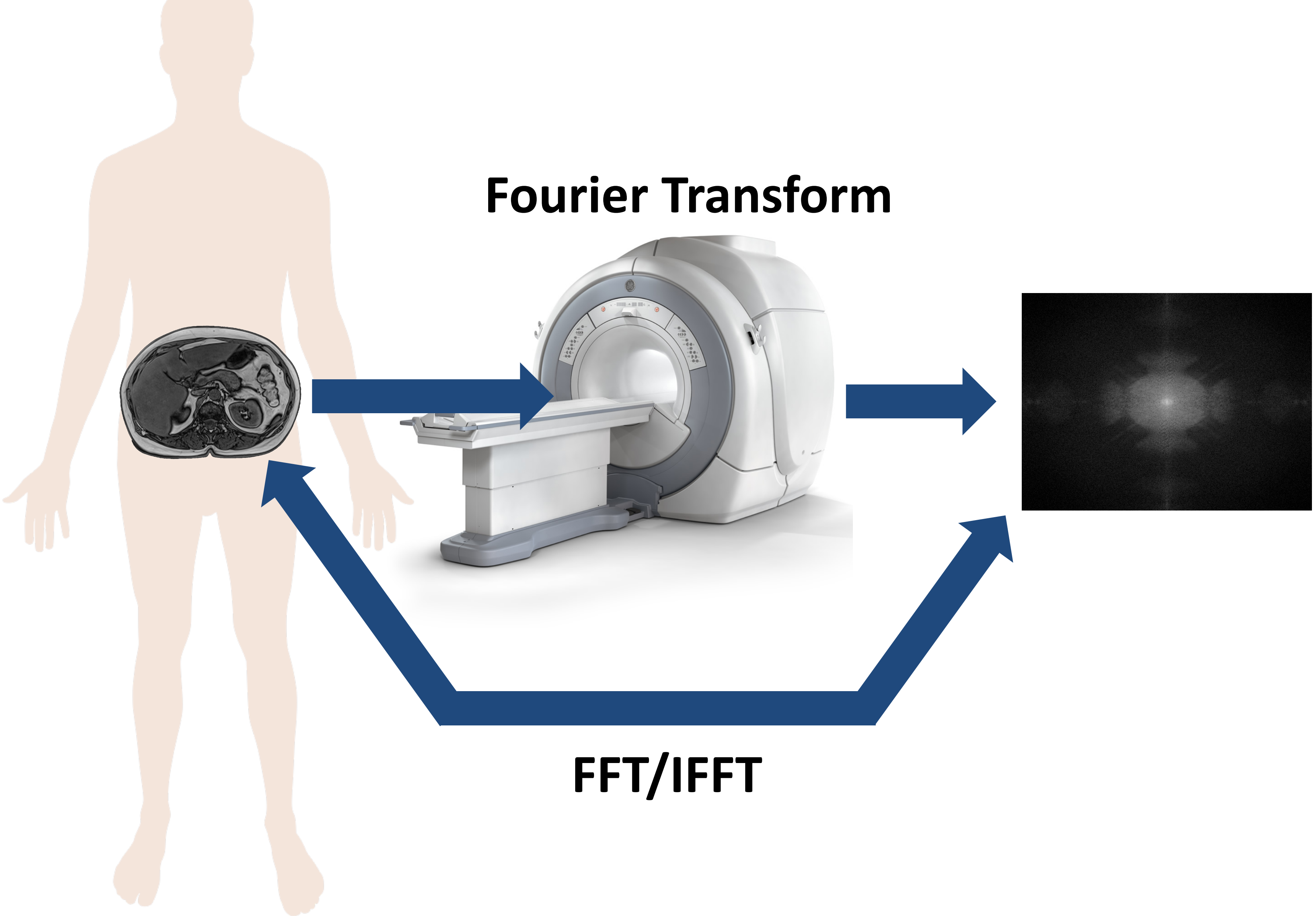
Fraction of non-zero coefficients not recovered at time t

Theory is by using ***density evolution equations***

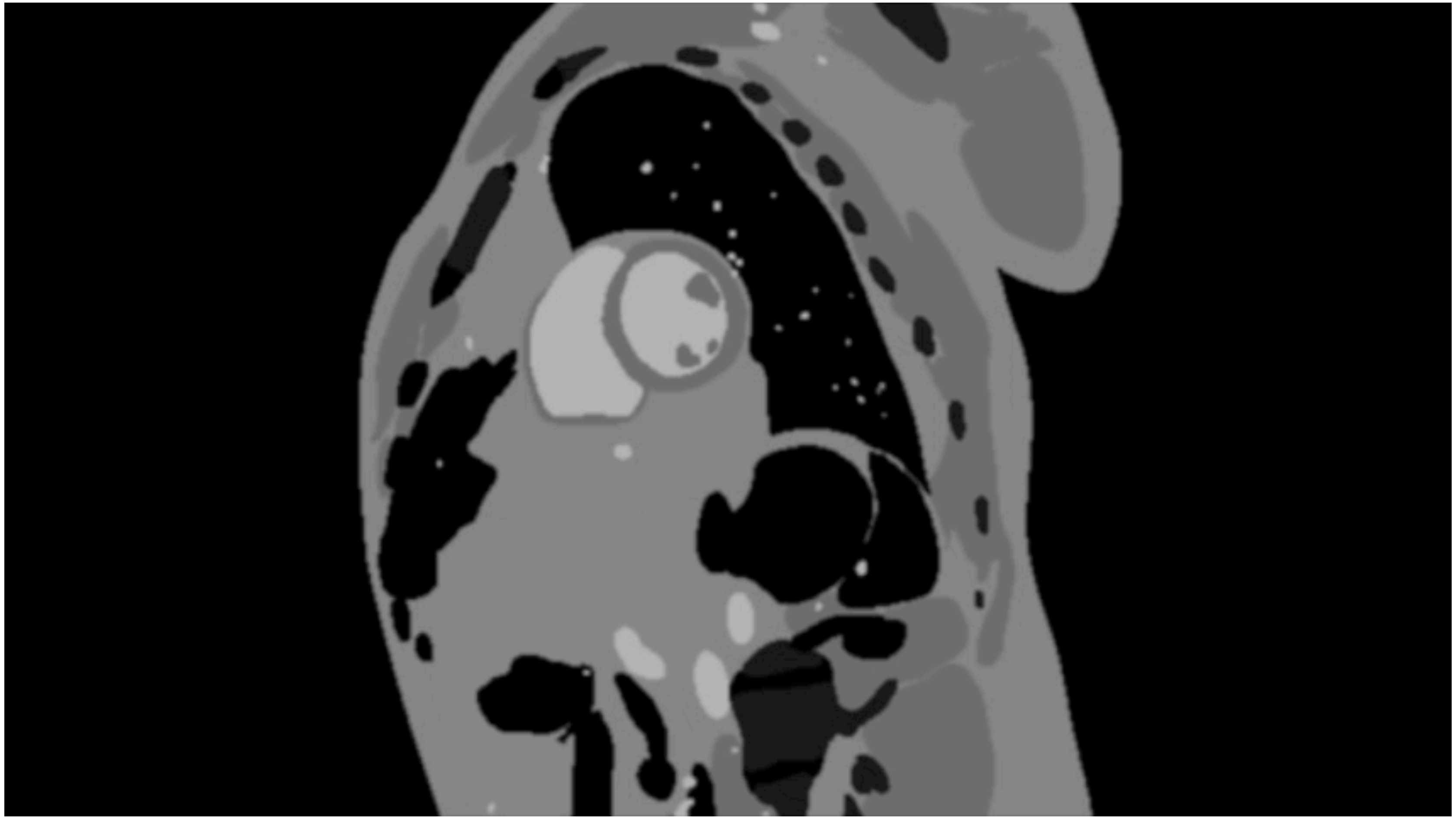
From Noiseless to Noisy



Magnetic resonance imaging



Numerical Phantoms for Cardiovascular MR



http://www.biomed.ee.ethz.ch/research/bioimaging/cardiac/mrxcat

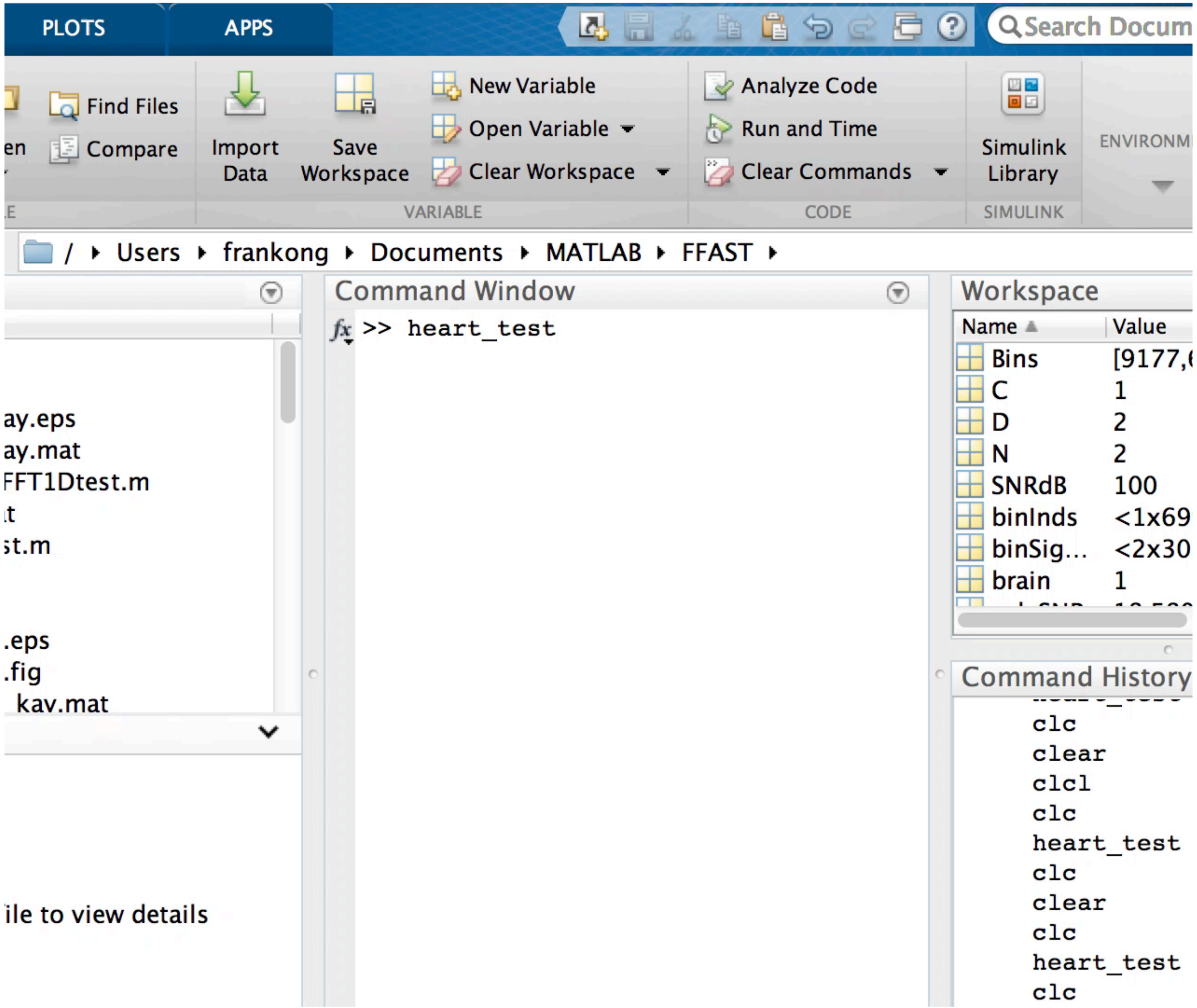
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

$$336 = 16 \times 21$$
$$323 = 17 \times 19$$

Numerical Phantoms for Cardiovascular MR

temporal difference across different frames of the phantom

Real Time Reconstruction in MATLAB on a Macbook



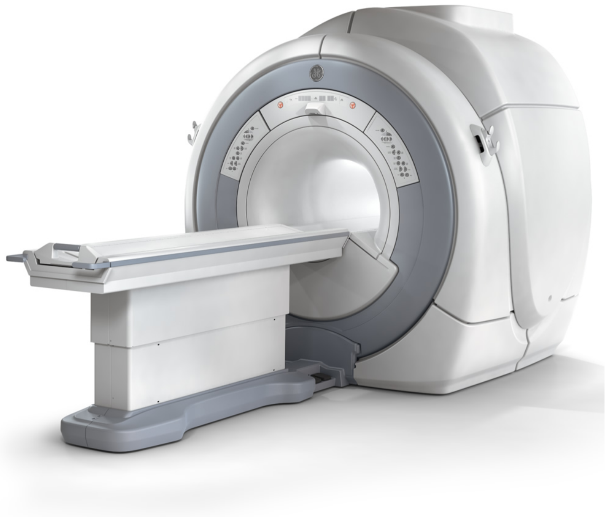
Measurements: 35.33% of Nyquist rate

MRI Viewfinder

Kodak, 1975

Viewing the photograph

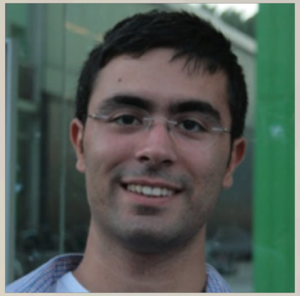
Canon, 2000



Real-time MRI with viewfinder

Sameer Pawar

Simon Li



Orhan Ocal

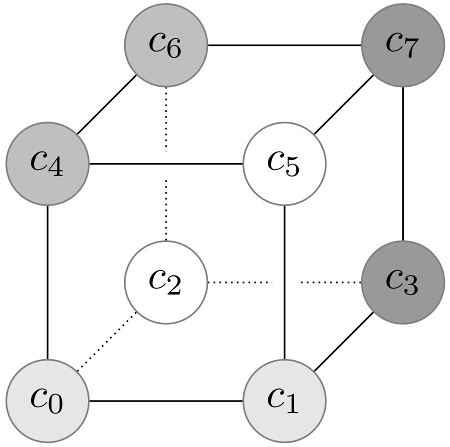
Chapter 4 (part 2)

Speeding up learning and recovery of pseudo-Boolean functions

Walsh-Hadamard Transform (WHT)

- N-point Discrete Fourier Transform (DFT)

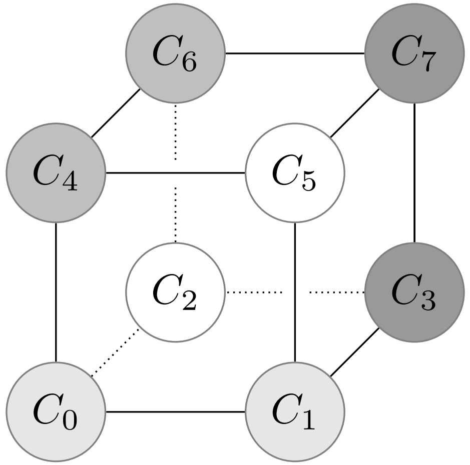
$$f[m]=\frac{1}{N}\sum_{k=0}^{N-1}F[k]e^{\mathrm{i}\frac{2\pi k}{N}m},\quad m=0,\cdots,N-1$$



- N-point Walsh-Hadamard (WHT) with $N=2^n$

$$f[\mathbf{m}]=\sum_{\mathbf{k}\in\{0,1\}^n}F[\mathbf{k}](-1)^{\langle\mathbf{k},\mathbf{m}\rangle},\quad \mathbf{m}\in\{0,1\}^n$$

Equivalent to a high-dim. DFT over the hyper-cube



- *F*[k] is sparse in many machine learning applications:
  - Decision tree and regression tree
  - Evolutionary biology
  - Hypergraph sketching

WHT: polynomial interpretation

$$f[\mathbf{m}] = \sum_{\mathbf{k}\in\{0,1\}^n} F[\mathbf{k}](-1)^{\langle \mathbf{k},\mathbf{m}\rangle}, \quad \mathbf{m} \in \{0,1\}^n$$

- Set $x_i = (-1)^{m_i}$ to get a multlinear polynomial $f: \{-1,1\}^n \to \mathbb{R}$

Ex. $n = 2$:

$$f(x_1, x_2) = F_0 1 + F_1 x_1 + F_2 x_2 + F_3 x_1 x_2$$

Ex. $n = 3$:

$$f(x_1, x_2, x_3) = F_0 1 + F_1 x_1 + F_2 x_2 + F_3 x_1 x_2 + F_4 x_3 + F_5 x_1 x_3 + F_6 x_2 x_3 + F_7 x_1 x_2 x_3$$

1-1 mapping between WHT coeffs. F_i's and the evaluations of $f(x_1, x_2, x_3)$ at $x_i = (-1)^{m_i}$

Recovering the function

Example for
$f : \{-1,1\}^2 \to \mathbb{R}$

$$\begin{bmatrix} f(1,1) \\ f(-1,1) \\ f(1,-1) \\ f(-1,-1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} F_{\{\}} \\ F_{\{1\}} \\ F_{\{2\}} \\ F_{\{1,2\}} \end{bmatrix}$$

Polynomial recovery

Recover the polynomial $f : \{-1,1\}^n \to \mathbb{R}$

Example for $n = 3$:

$$f(x_1, x_2, x_3) = F_0 1 + F_1 x_1 + F_2 x_2 + F_3 x_1 x_2 + F_4 x_3 + F_5 x_1 x_3 + F_6 x_2 x_3 + F_7 x_1 x_2 x_3$$

input	*f*
(1,1,…,1)	$f(1,1,\ldots,1)$
(1,1,…,-1)	$f(1,1,\ldots,-1)$
…	…

Evaluate the function at every point

Sample complexity: $N = 2^n$

What if only K out of N WHT coeffs. are non-zero?

Ex: **No. of variables** **n = 30**

No. of input combinations N= 1 billion

Sparsity **K = 64**

 # evalulations $= 2600\ (\approx 1.23Kn)$

Main Result

Example for $n = 3$:

$$f(x_1, x_2, x_3) = F_0 1 + F_1 x_1 + F_2 x_2 + F_3 x_1 x_2 + F_4 x_3 + F_5 x_1 x_3 + F_6 x_2 x_3 + F_7 x_1 x_2 x_3$$

We can learn $f: \{-1,1\}^n \to \mathbb{R}$ whose spectrum is K-sparse:

- with a sample complexity of $\boldsymbol{O(nK)}$
- with a computational complexity of $\boldsymbol{O(nK\,logK)}$
- can be made robust to noise

$$\boldsymbol{n = log(N)}$$

Insights:

Sub-sampling → **Aliasing** in the WHT domain

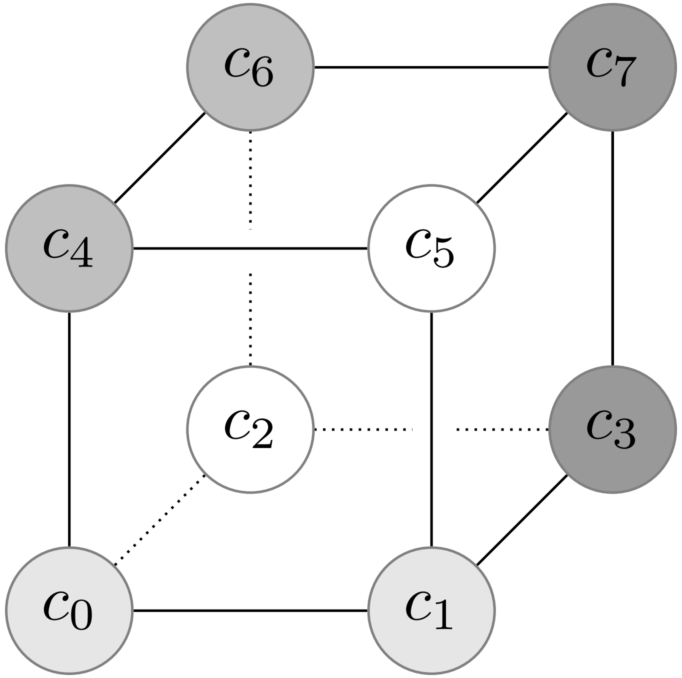
Clever sub-sampling (for **sparse** case) → **Good** "alias" code ***(Sparse graph codes)***

Li, Pawar, Bradley, Ramchandran, 2015

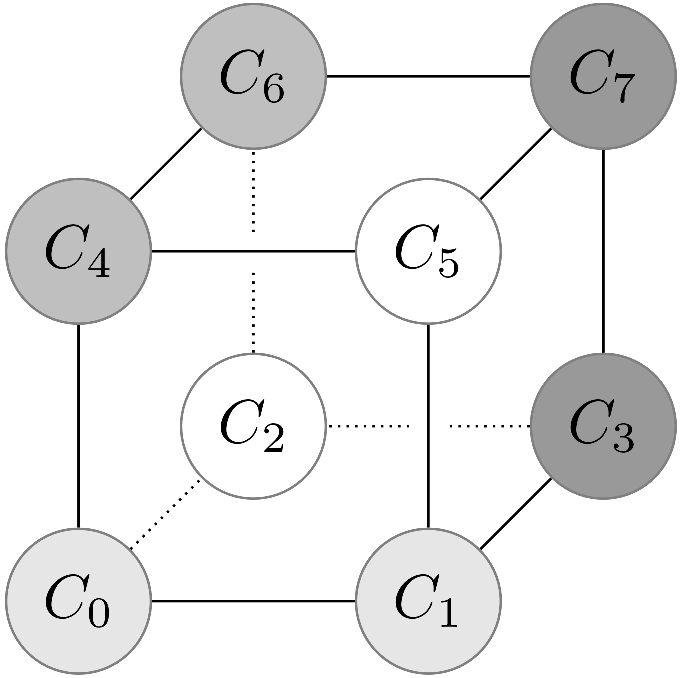
Walsh-Hadamard Transform

Equivalent to a high-dim. DFT
over the hyper-cube

"time" domain

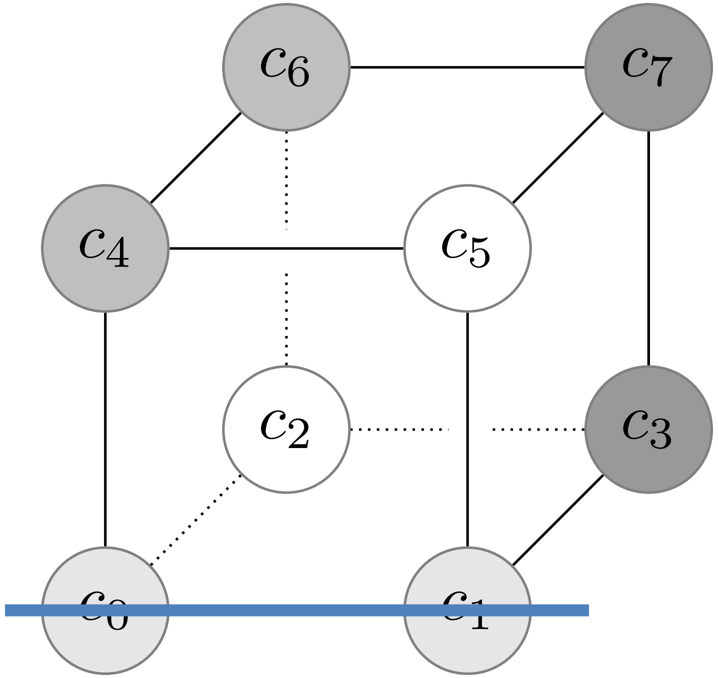


WH domain

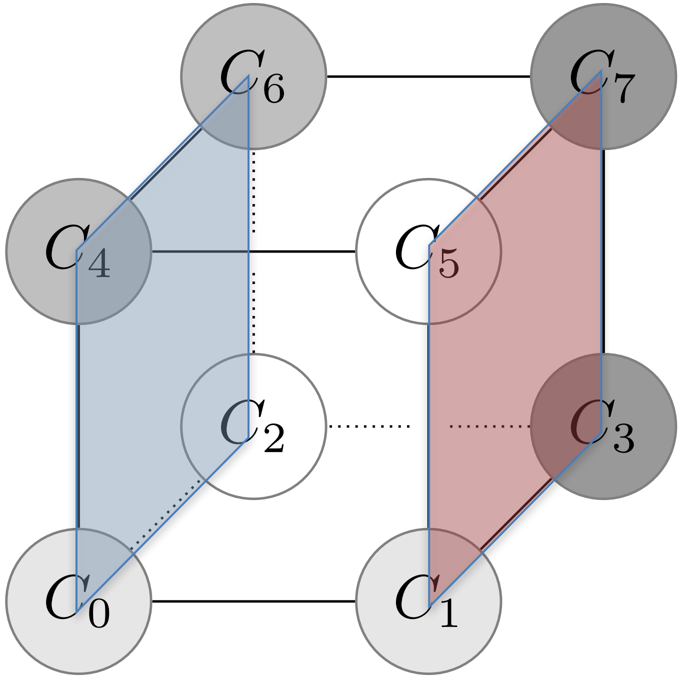


Walsh-Hadamard Transform

“time” domain

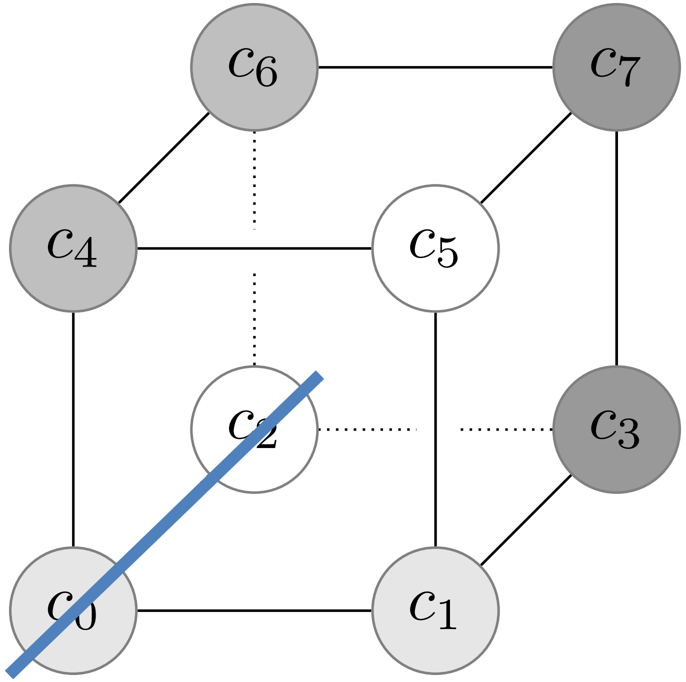


WH domain

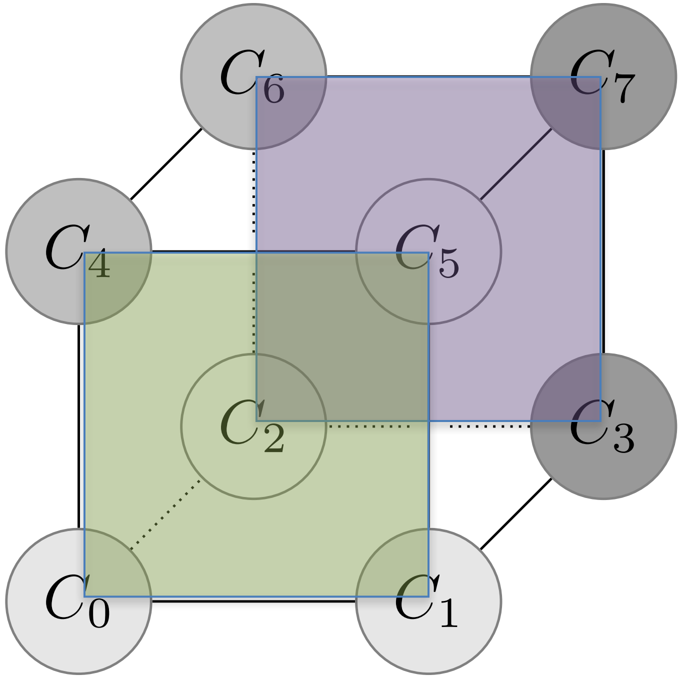


Walsh-Hadamard Transform

“time” domain

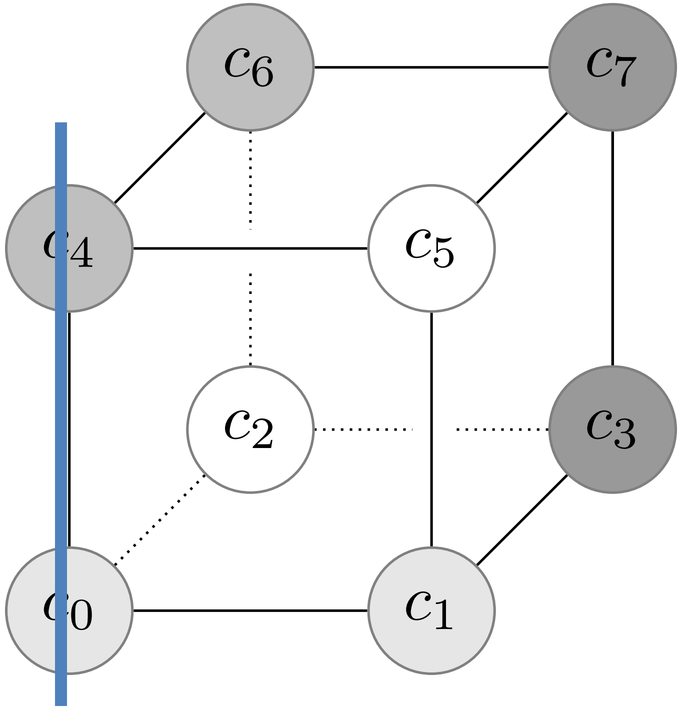


WH domain

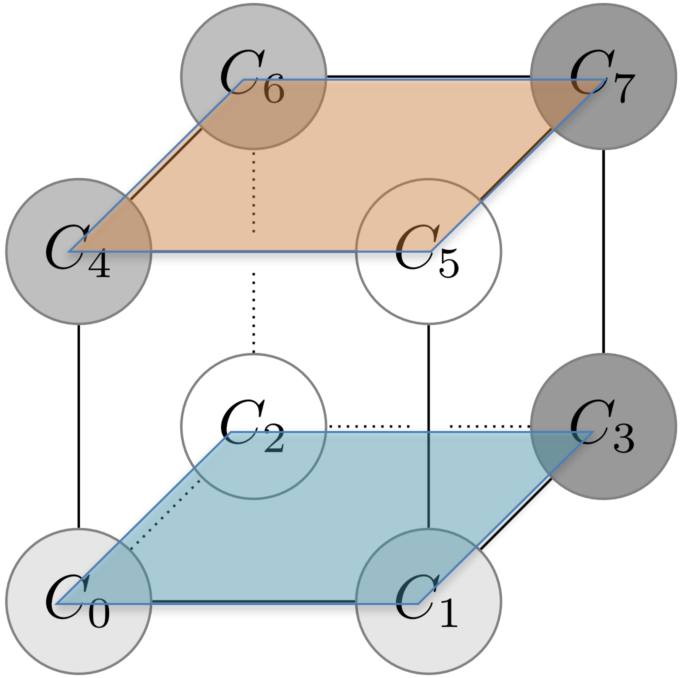


Walsh-Hadamard Transform

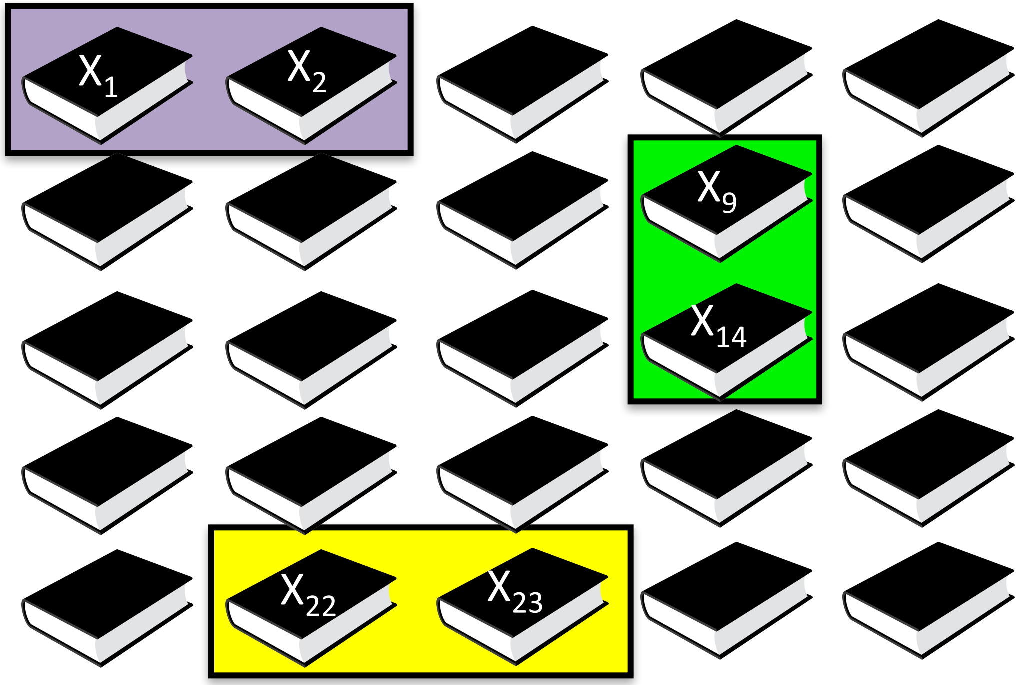
“time” domain



WH domain



WHT – Hypergraph Sketching



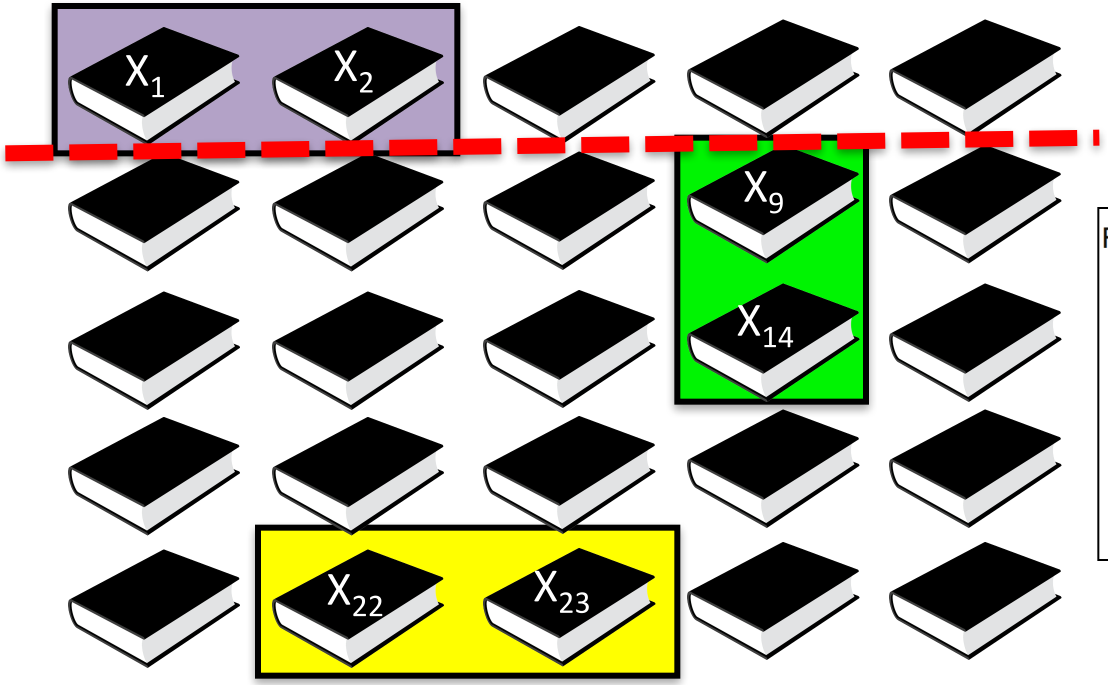
$n = \#$ of books

$s = \#$ of sale patterns

$2^n \approx 10^9$ possible hyperedges
if $n = 30$

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the **cuts** of the graph instead!

WHT – Hypergraph Sketching



$n = \#$ of books

$s = \#$ of sale patterns

$2^n \approx 10^9$ possible hyperedges
if $n = 30$

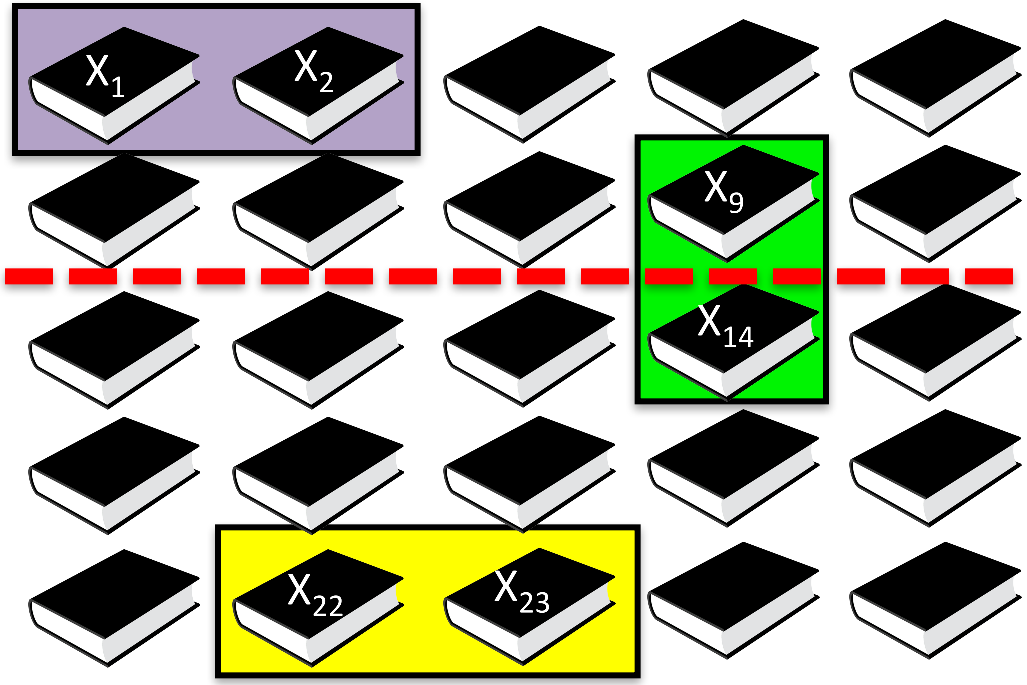
- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the **cuts** of the graph instead!

consider a **cut** :

$$x_1 = \cdots = x_5 = +1$$
$$x_6 = \cdots = x_{25} = -1$$

$$\implies \text{cut value } f(\boldsymbol{x}) = 0$$

WHT – Hypergraph Sketching



$n = \#$ of books

$s = \#$ of sale patterns

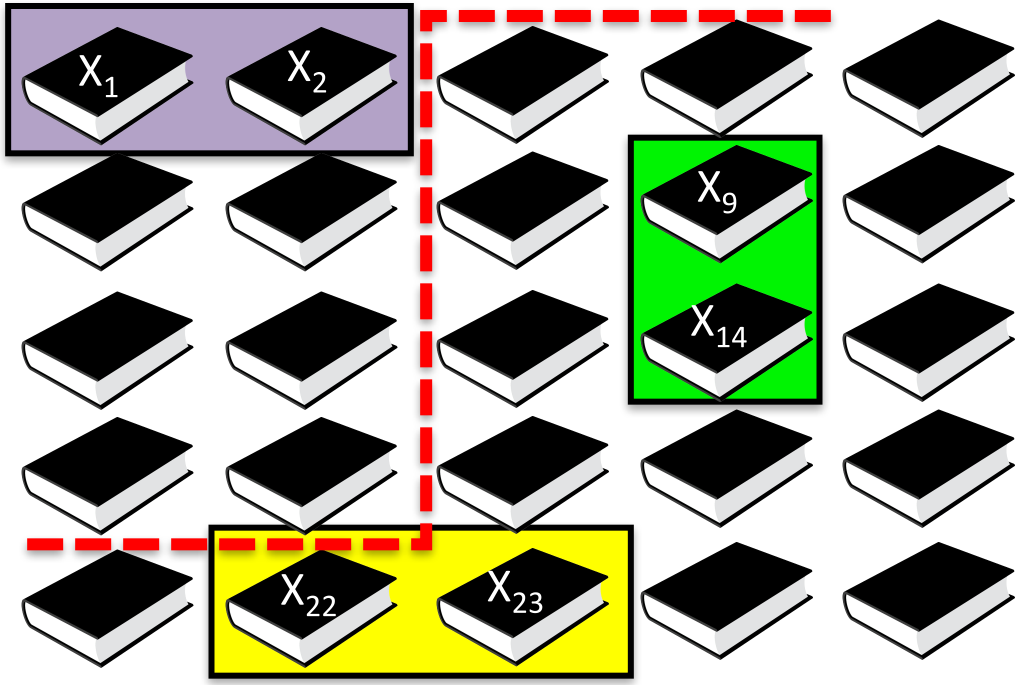
$2^n \approx 10^9$ possible hyperedges if $n = 30$

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the **cuts** of the graph instead!

consider a **cut** :

$$x_1 = \cdots = x_{10} = +1$$
$$x_{11} = \cdots = x_{25} = -1$$
$$\implies \text{cut value } f(\boldsymbol{x}) = 1$$

WHT – Hypergraph Sketching



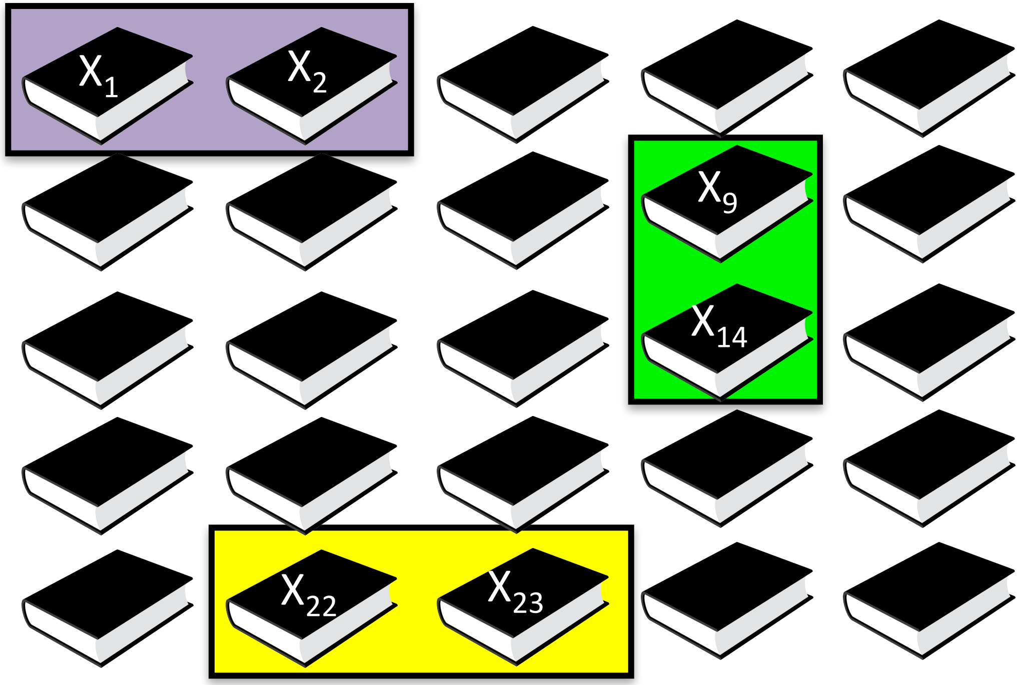
$n = \#$ of books

$s = \#$ of sale patterns

$2^n \approx 10^9$ possible hyperedges if $n = 30$

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the **cuts** of the graph instead!

WHT – Hypergraph Sketching



$n = \#$ of books

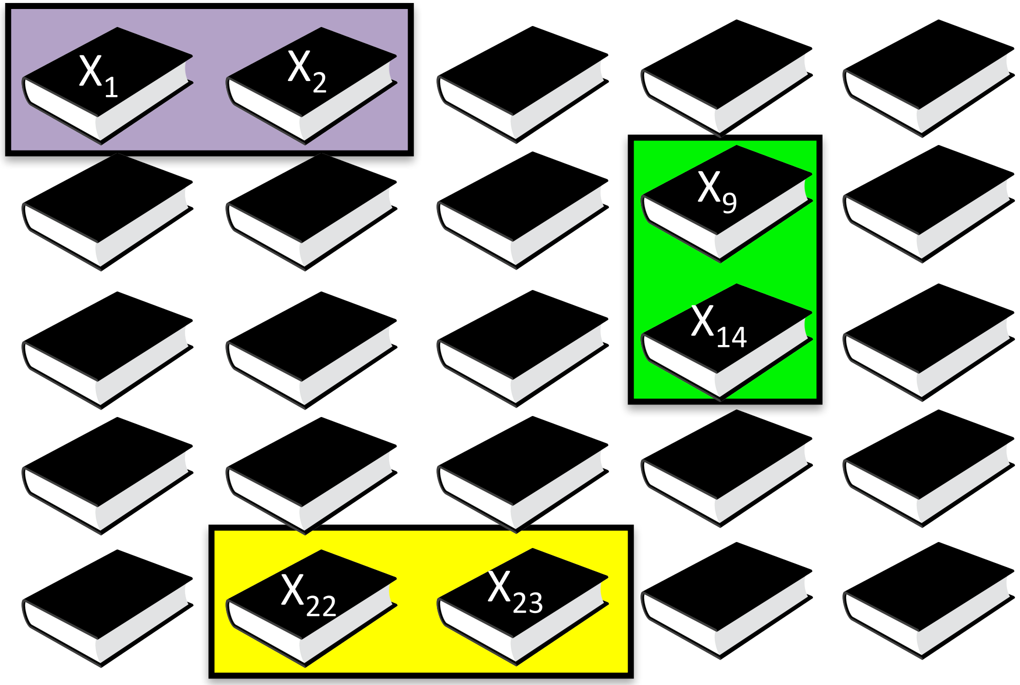
$s = \#$ of sale patterns

$2^n \approx 10^9$ possible hyperedges if $n = 30$

- recover all sale patterns (hyperedges) without logging every transaction?
- sketch the **cuts** of the graph instead!
- Generally speaking, we have the **cut function**

$$f(\boldsymbol{x}) = \frac{3}{2} - \frac{1}{2}x_1x_2 - \frac{1}{2}x_9x_{14} - \frac{1}{2}x_{22}x_{23}$$

WHT – Hypergraph Sketching



$n = \#$ of books

$s = \#$ of sale patterns

$2^n \approx 10^9$ possible hyperedges
if $n = 30$

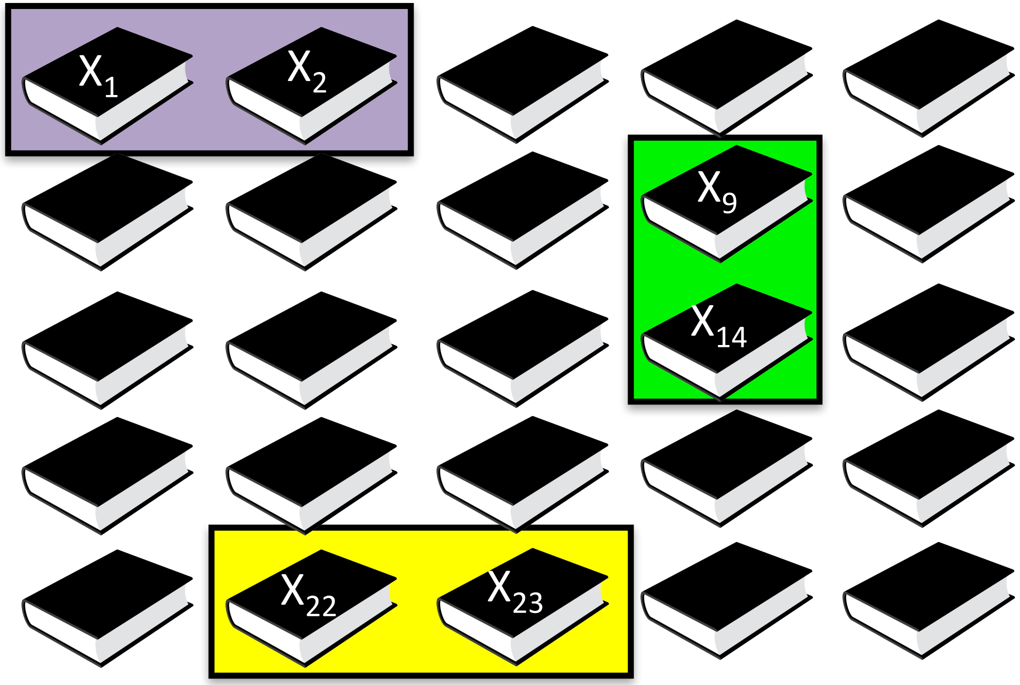
- small # of sale patterns $s \ll n$
- small # of items per sale $d \ll n$

- K-sparse polynomial
- $K \leq s2^{d-1}$

- Generally speaking, we have the **cut function**

$$f(\boldsymbol{x}) = \frac{3}{2} - \frac{1}{2}x_1x_2 - \frac{1}{2}x_9x_{14} - \frac{1}{2}x_{22}x_{23}$$

WHT – Hypergraph Sketching



$n = \#$ of books

$s = \#$ of sale patterns

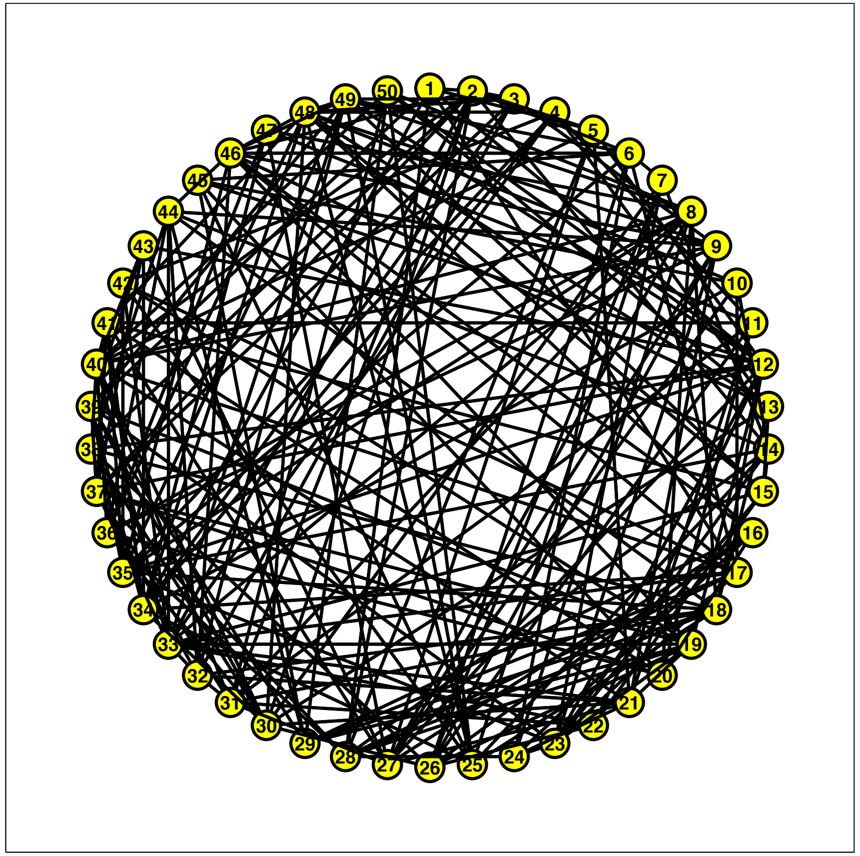
$2^n \approx 10^9$ possible hyperedges
if $n = 30$

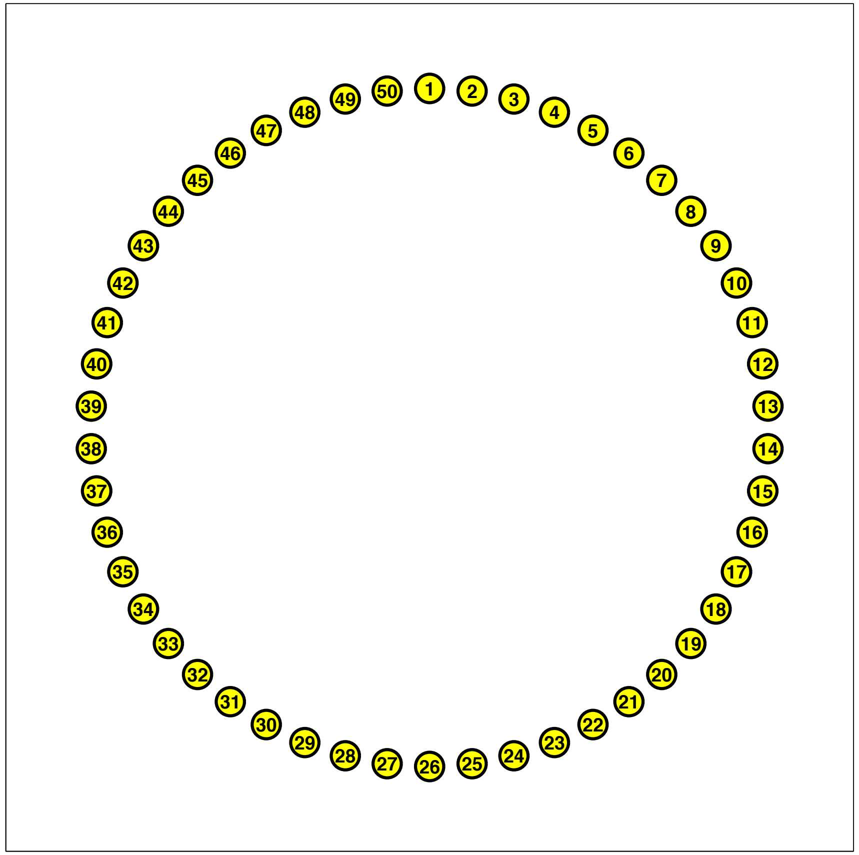
sub-sample **cuts** → recover **hyperedges**

- Generally speaking, we have the **cut function**

$$f(\boldsymbol{x}) = \frac{3}{2} - \frac{1}{2}x_1x_2 - \frac{1}{2}x_9x_{14} - \frac{1}{2}x_{22}x_{23}$$

WHT – Hypergraph Sketching





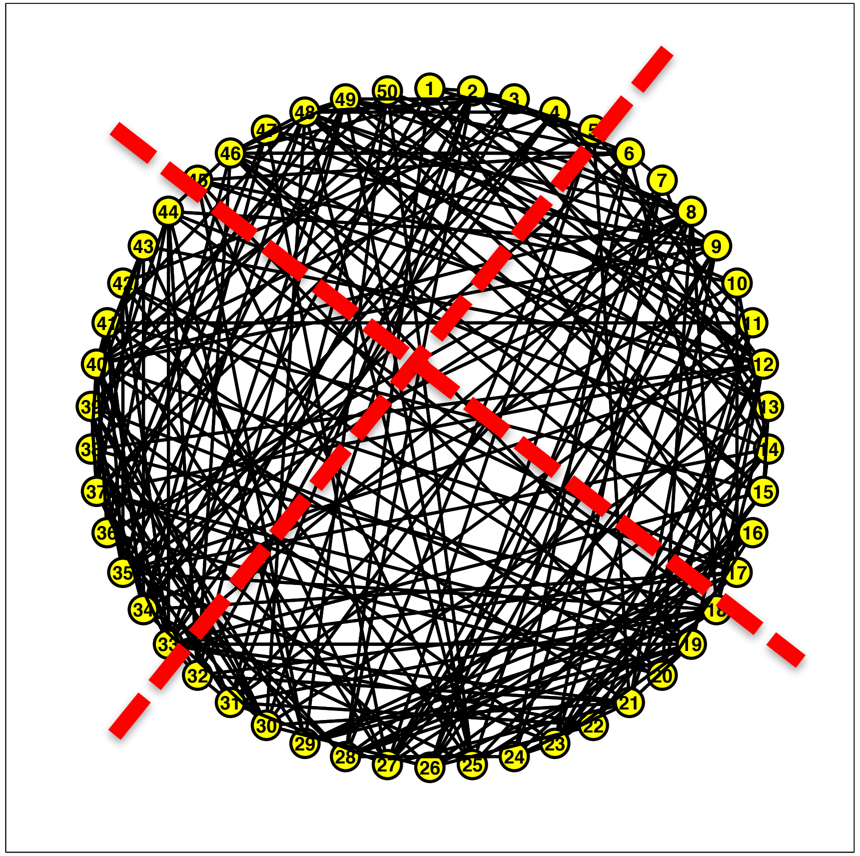
$n = 50$ books

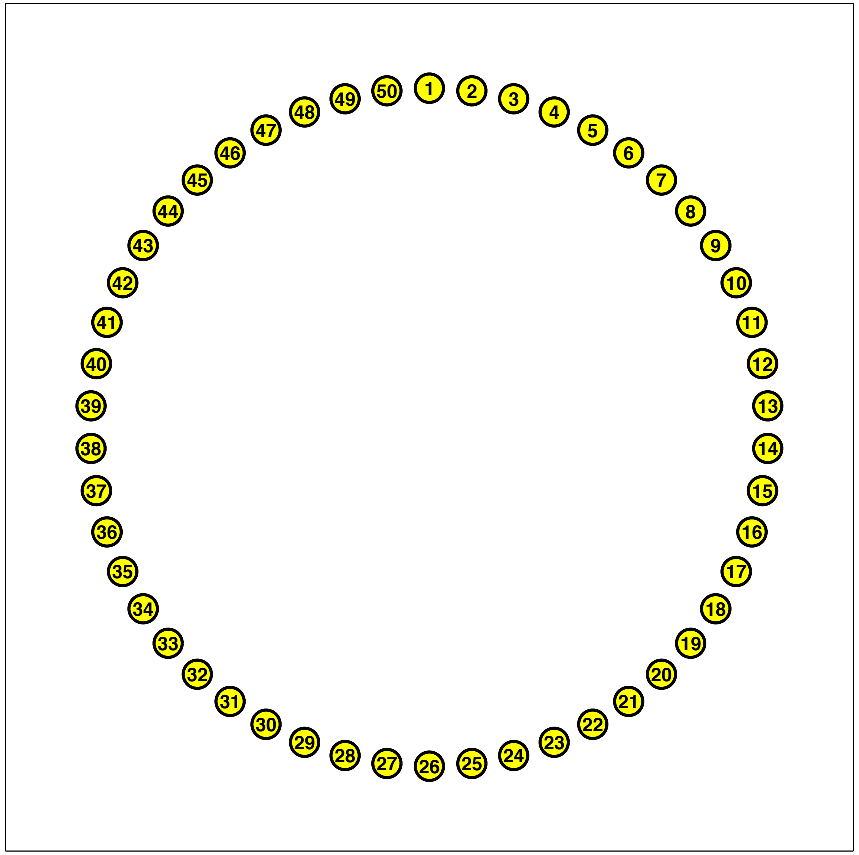
$d = 2$ items/sale

$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$

WHT – Hypergraph Sketching





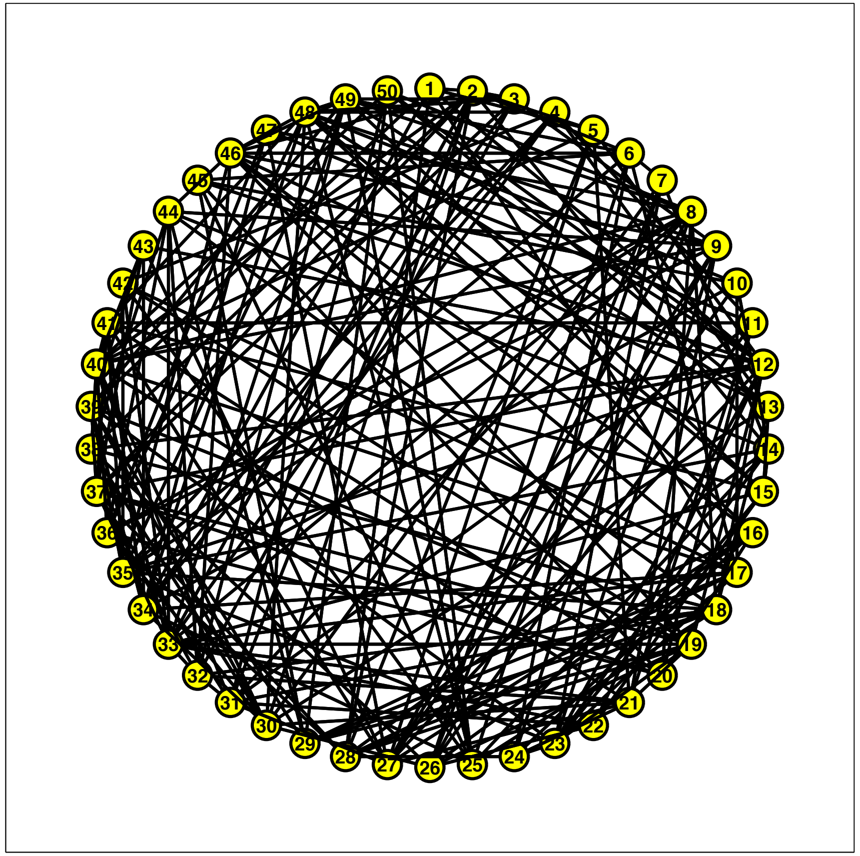
$n = 50$ books

$d = 2$ items/sale

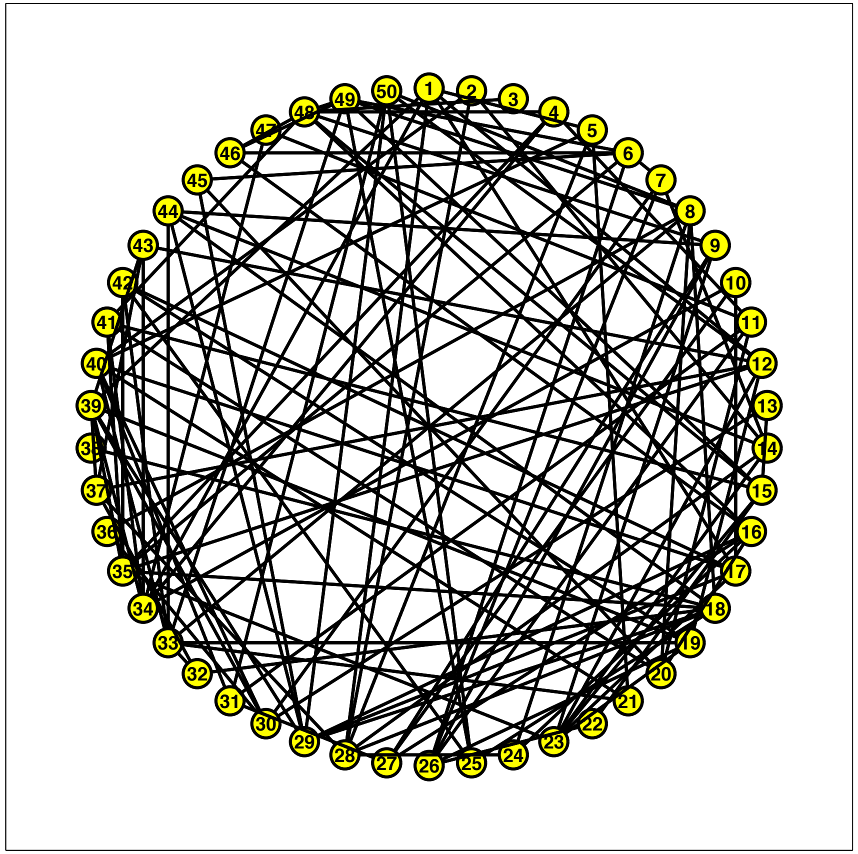
$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$

WHT – Hypergraph Sketching



Hyperedge at iteration 1



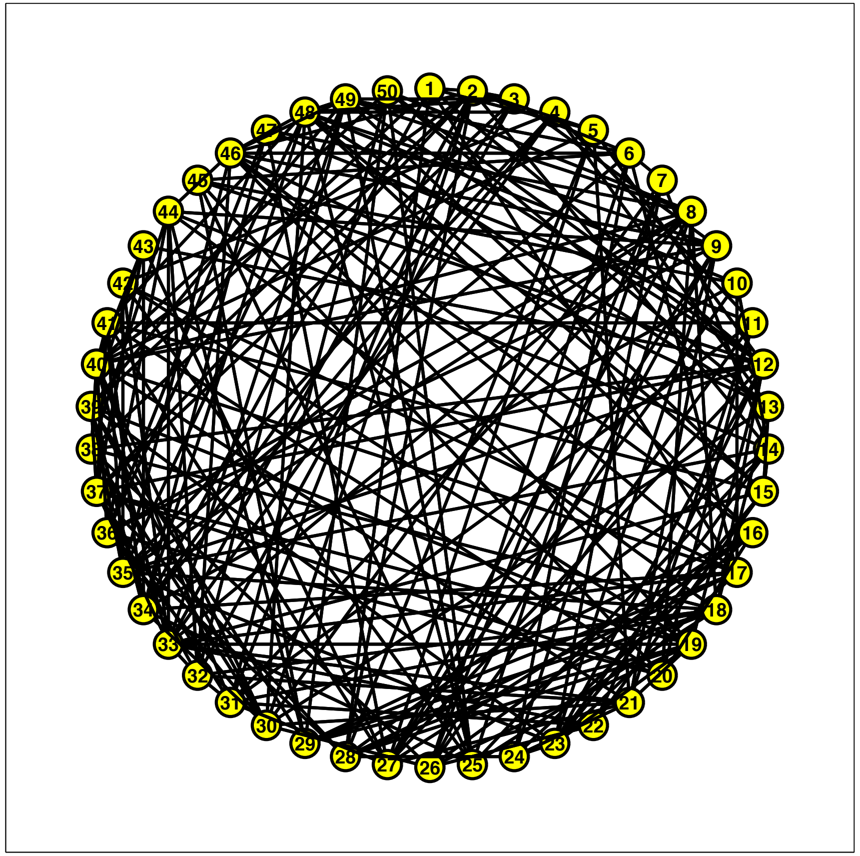
$n = 50$ books

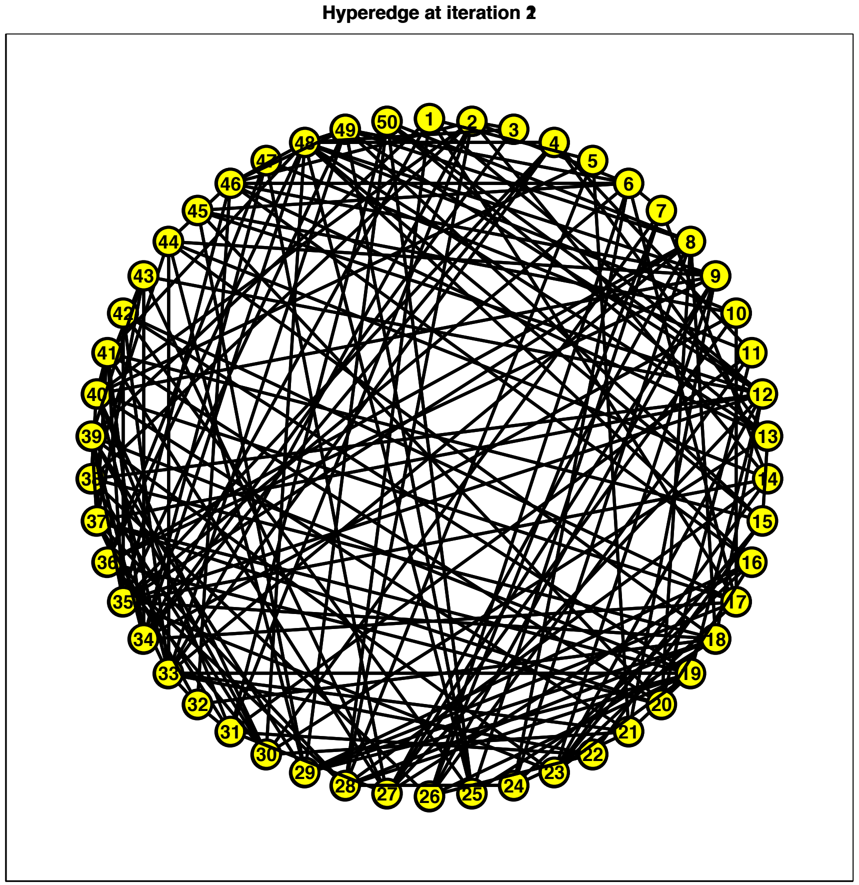
$d = 2$ items/sale

$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- $\#$ of cut queries $O(Kn) \approx 25000$

WHT – Hypergraph Sketching





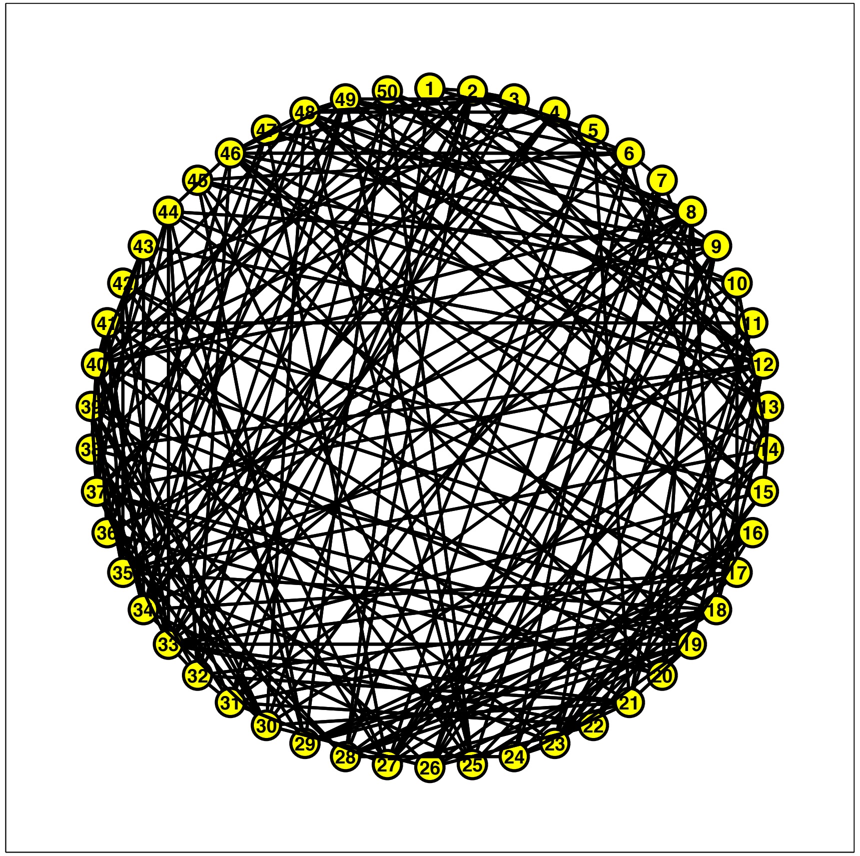
$n = 50$ books

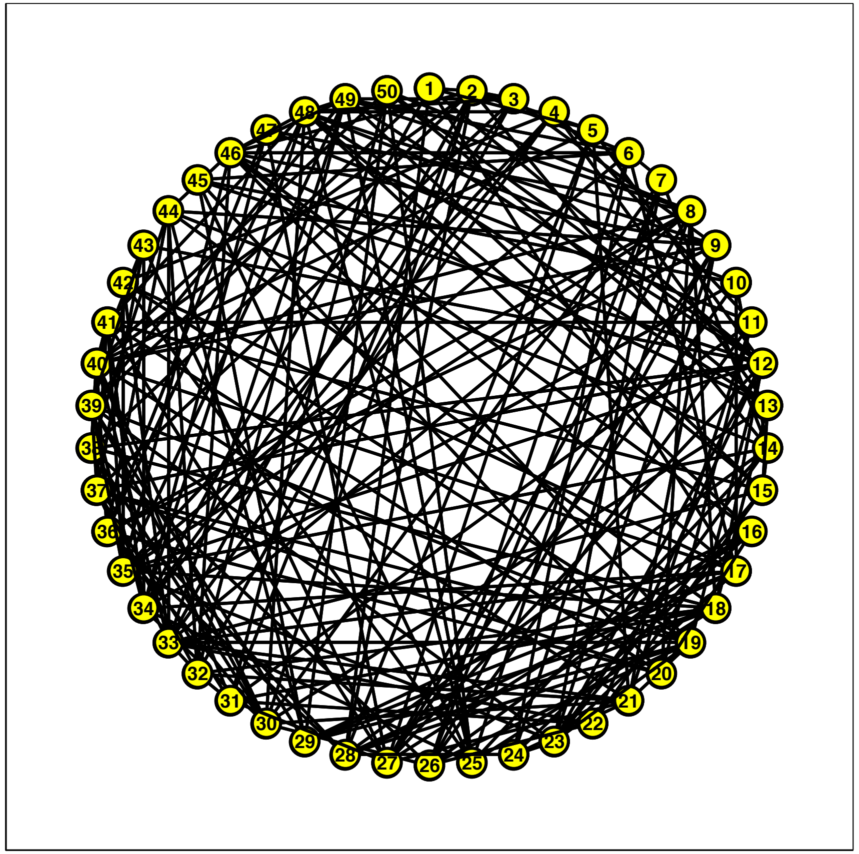
$d = 2$ items/sale

$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- # of cut queries $O(Kn) \approx 25000$

WHT – Hypergraph Sketching





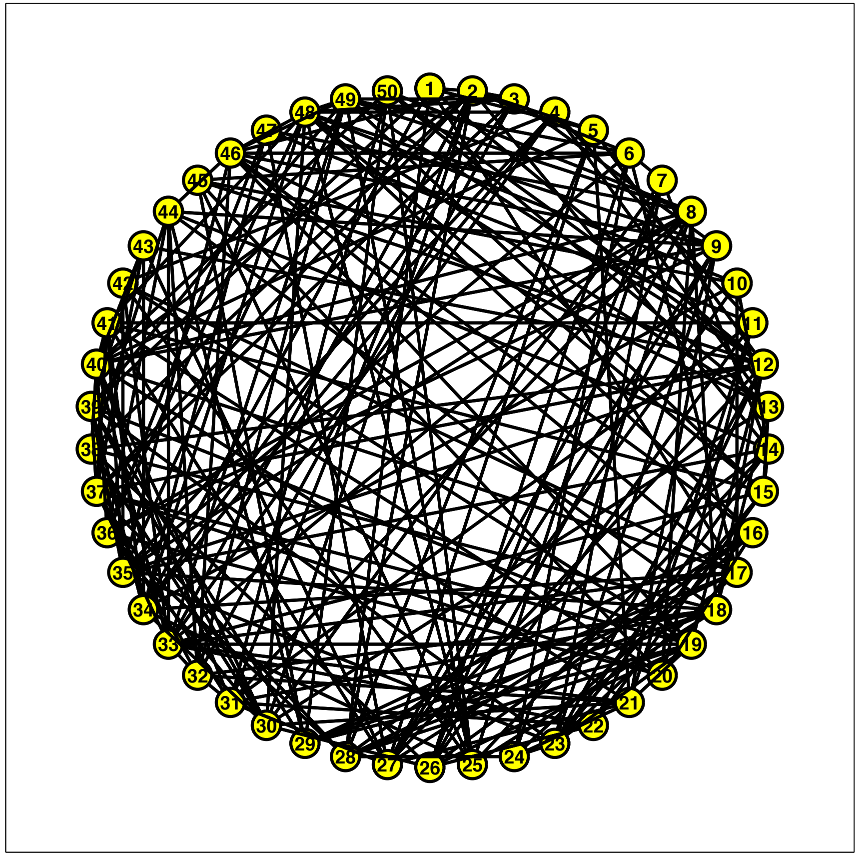
$n = 50$ books

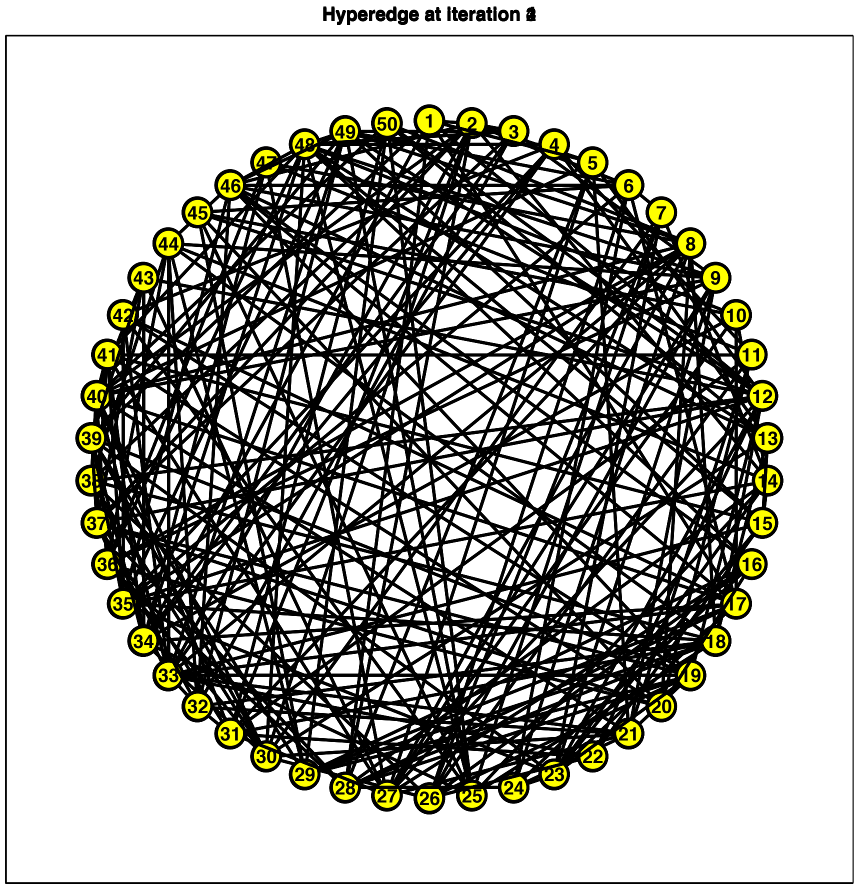
$d = 2$ items/sale

$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- $\#$ of cut queries $O(Kn) \approx 25000$

WHT – Hypergraph Sketching





$n = 50$ books

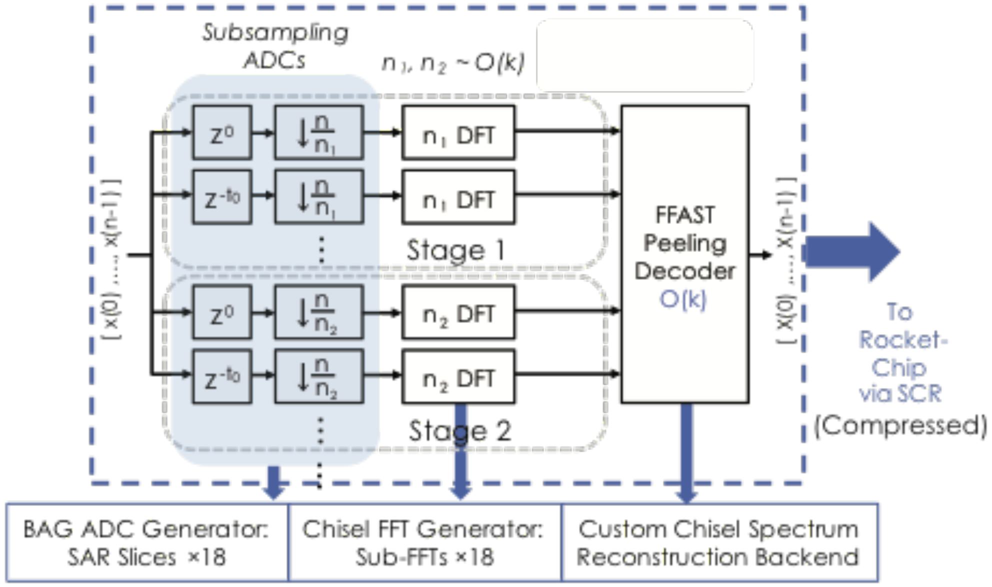
$d = 2$ items/sale

$s = 250$ sale patterns

- total cut values $2^n = 2^{50}$
- sparsity $K \leq s2^{d-1} = 500$
- $\#$ of cut queries $O(Kn) \approx 25000$

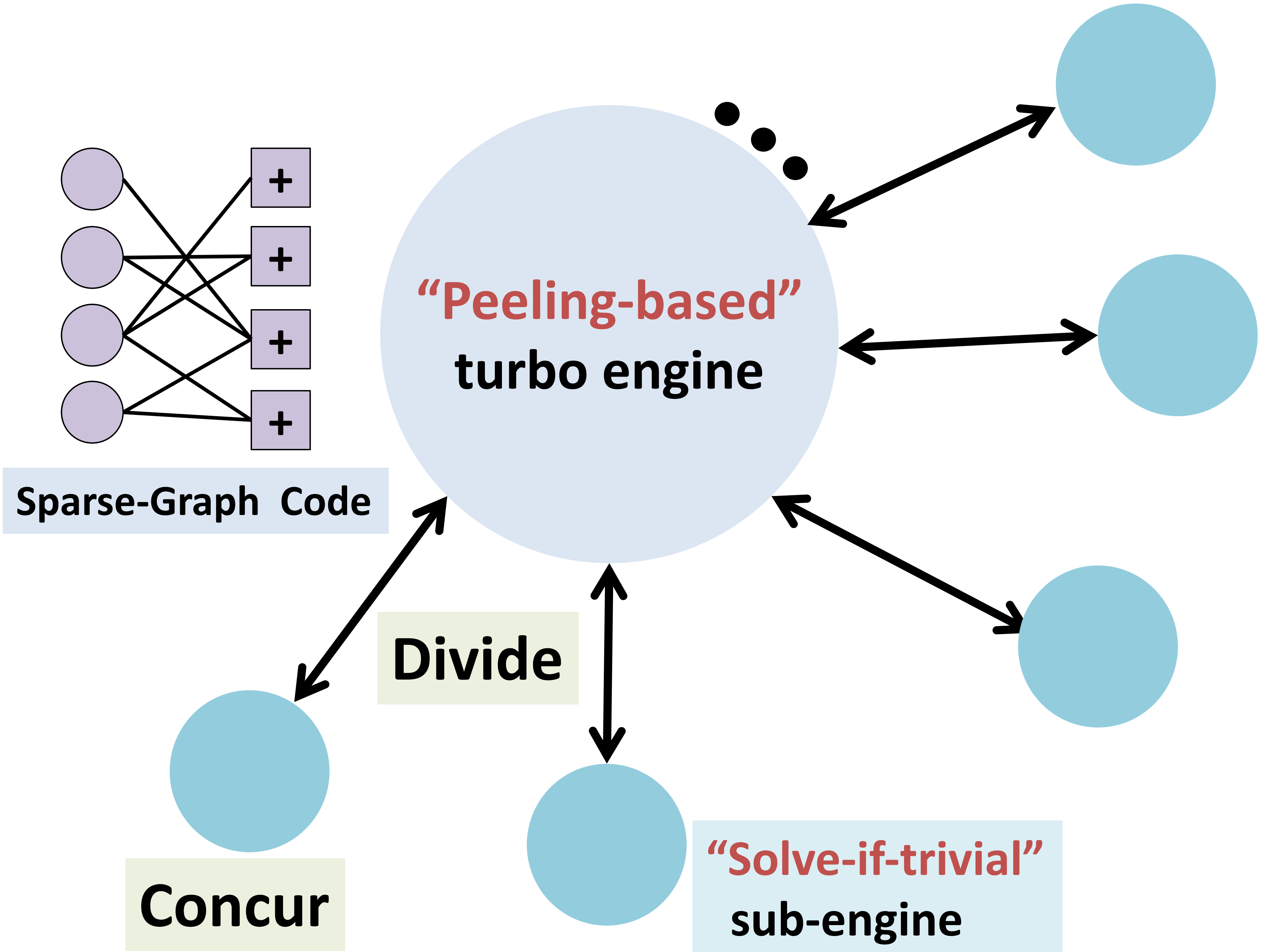
Open source implementations

- **Sparse FFT and Sparse WHT** implemented in C++
- Publicly available on **GitHub**
  https://github.com/ucbasics
- Hardware implementation of sparse FFT



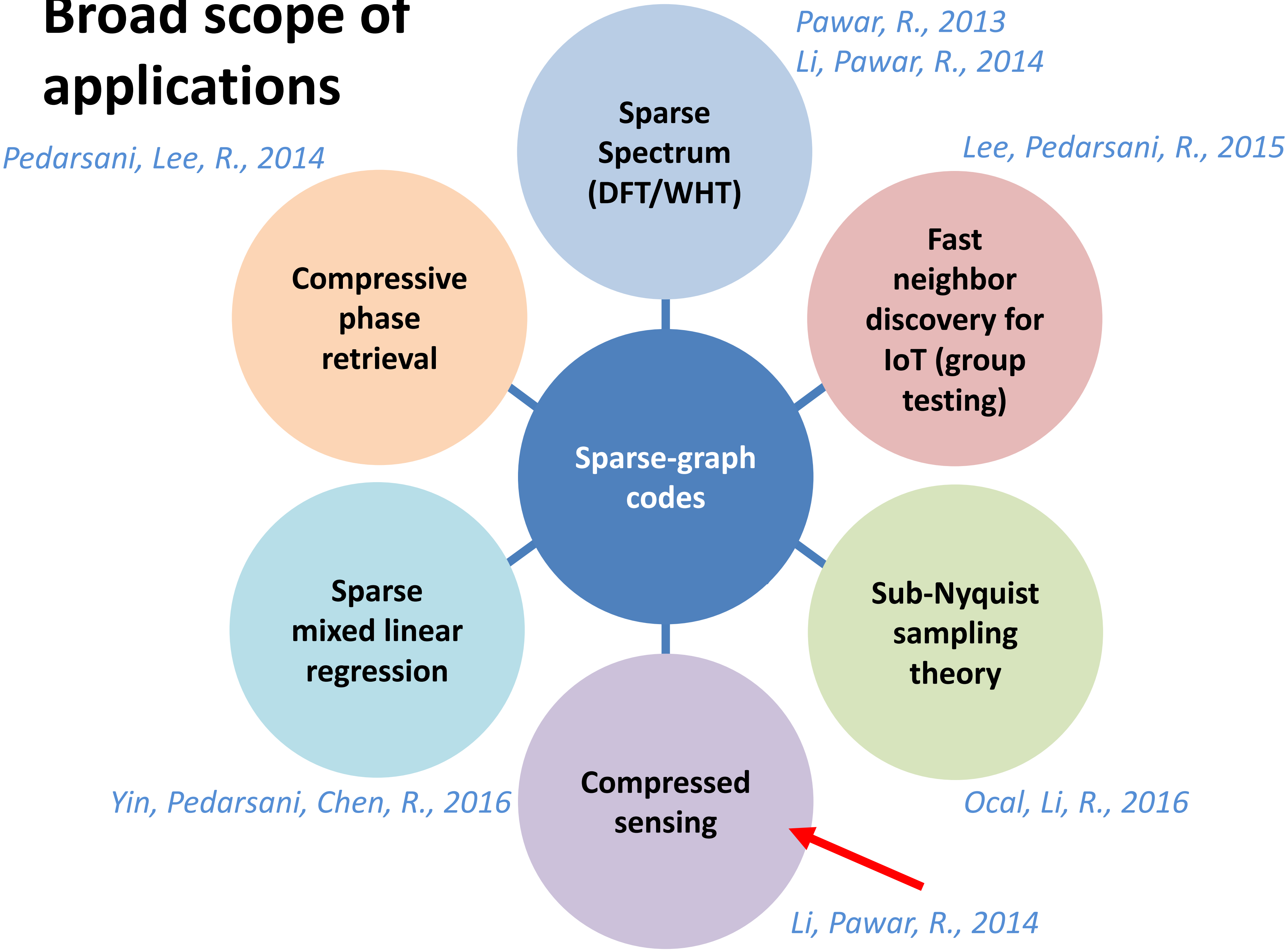
Technology	16nm FinFET
Bandwidth	2GHz
Analysis time	0.02ms
Compression	65%

IEEE JSSC, 2019 ***A Real-Time, 1.89-GHz Bandwidth, 175-kHz Resolution Sparse Spectral Analysis RISC-V SoC in 16-nm FinFET***
A. Wang , W. Bae , J. Han , S. Bailey , O. Ocal , P. Rigge , Z. Wang , K. Ramchandran , E. Alon , B. Nikolic



Sparse-Graph Code
"Peeling-based"
turbo engine
Divide
Concur
"Solve-if-trivial"
sub-engine

Broad scope of applications



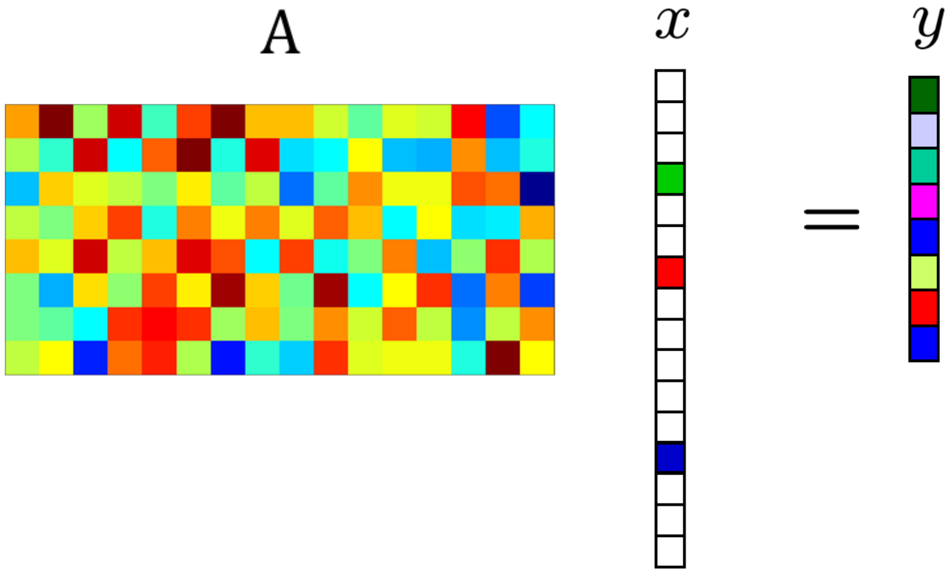
Compressed sensing

Estimate the K-sparse signal $\boldsymbol{x} \in \mathbb{C}^N$, which has only $K \ll N$ non-zero coefficients, from linear measurements in the presence of noise

$$\boldsymbol{y} = \mathbf{A}\boldsymbol{x} + \mathbf{w}$$

Methods based on convex relaxation

- Measurement matrix A has random design (e.g., random Gaussian matrix)
- Solve the the convex optimization problem Minimize $||Ax - y|| + \lambda ||x||_1$
- Measurements: $O\left(K \log \frac{N}{K}\right)$
- Computations: $O(\text{poly}(N))$

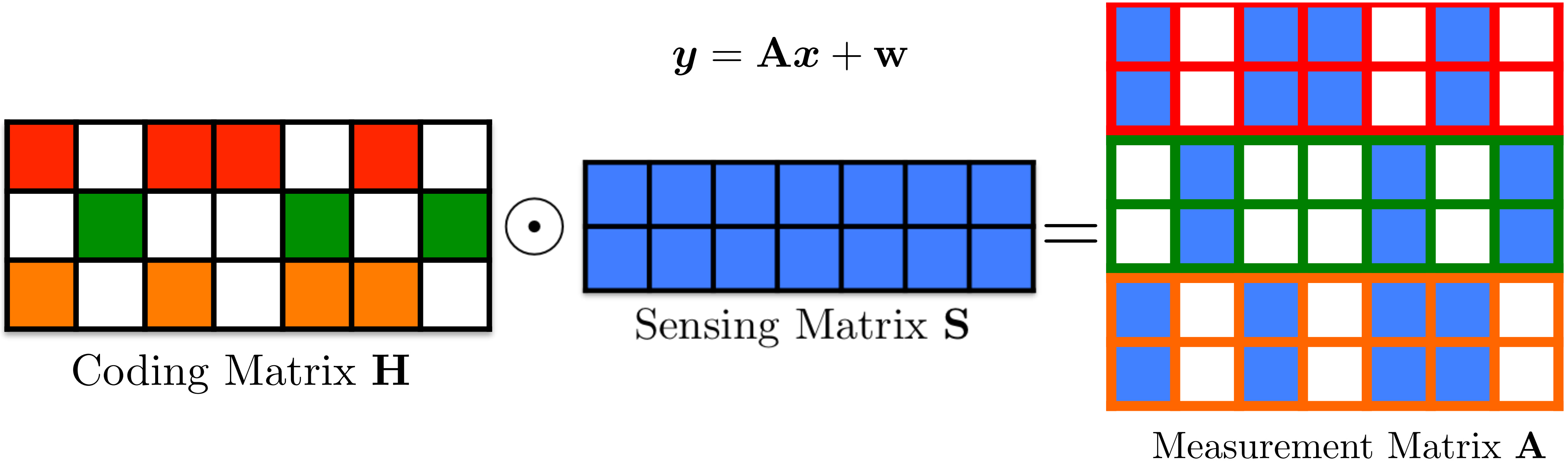


Candes 2006, Donoho 2006

Compressed sensing

Estimate the K-sparse signal $\boldsymbol{x} \in \mathbb{C}^N$, which has only $K \ll N$ non-zero coefficients, from linear measurements in the presence of noise

$$\boldsymbol{y} = \mathbf{A}\boldsymbol{x} + \mathbf{w}$$



Recovery w.h.p. using

Noiseless: $O(K)$ samples, $O(K)$ computations

Noisy: $O(K \log N)$ samples, $O(K \log N)$ computations

Li, Pawar, R., 2014 – Yin et al., 2019

Generic method to make algorithm robust to noise

Recall how we find locations and values of singletons in the noiseless setting.

Ex.: a singleton with non-zero element *b* at index 4

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 \\ r_1 & r_2 W & r_3 W^2 & r_4 W^3 & r_5 W^4 & r_6 W^5 & r_7 W^6 & r_8 W^7 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ b \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} r_4 \\ r_4 W^3 b \end{bmatrix}$$

Location information is encoded in the ***relative phase*** between y_2 and y_1.

- What if we have $y_1 = r_4 + w_1$ and $y_2 = r_4 W^3 b + w_2$?
- $\angle\left(\frac{y_2}{y_1}\right) =?$

Generic method to make algorithm robust to noise

It is ***not*** robust to encode the **location** information in the relative phase! Alt. choice?

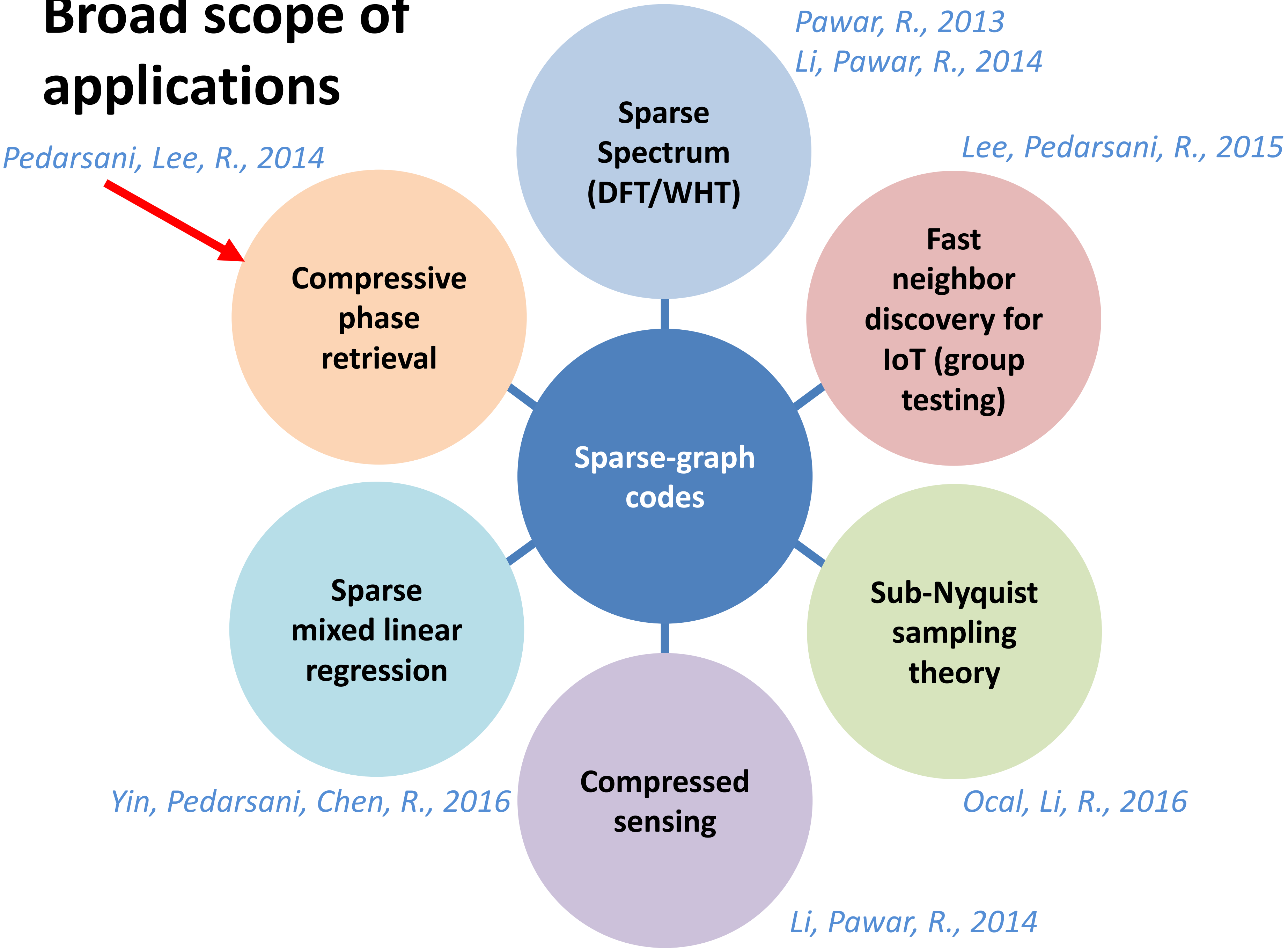
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

1. Represent each element by its binary index string: **(log N)**
2. Encode it using an error correcting code matched to the noise of the channel: **(C_1 log N)**
3. Add a unique random signature vector to each column to identify the element the column represents: **(C_2 log N)**.
4. Total cost (per measurement bin) is **O(log N)**.
5. No. of measurement bins is **O(K)** (using sparse graph codes).
6. Total measurement cost is **O(K log N)**.

Guess-and-check algorithm:

A. **Guess** that a received bin measurement corresponds to a **singleton**.
B. Find ML estimate of singleton **value and location** index (using coded representation).
C. **Verify** using signature vector if singleton hypothesis is correct.
D. **If yes**, “peel” singleton node from the other measurement bins it belongs to, and continue.
E. **If no**, continue to next measurement bin.

Broad scope of applications

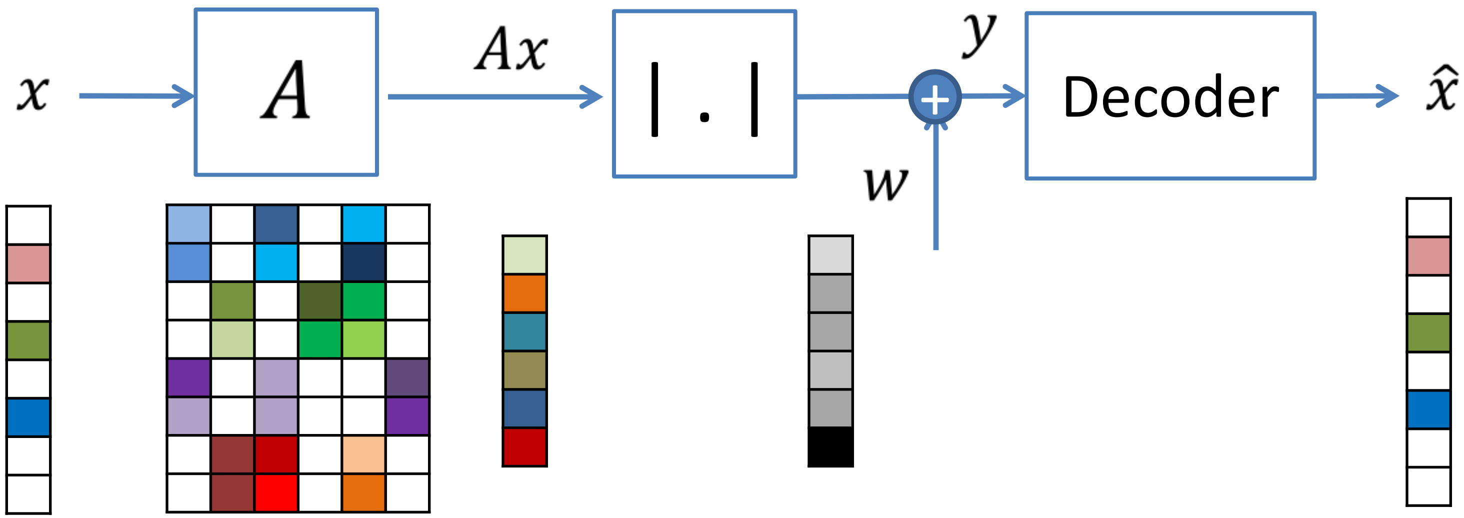


Compressive Phase Retrieval (CPR)

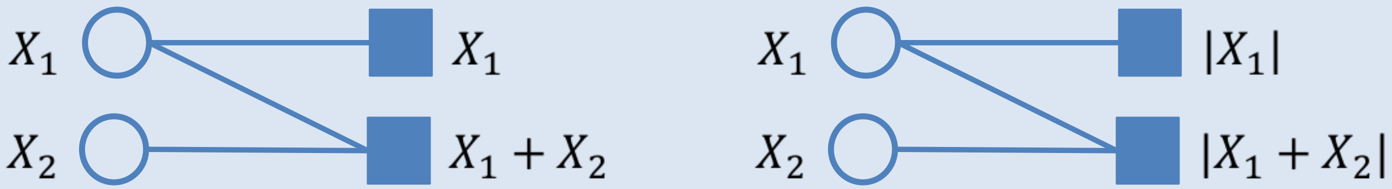
Recover a ***K-sparse*** signal x $\in \mathbb{C}^n$ from ***m magnitude*** measurements:

$$y = |Ax| + w,$$

where $A \in \mathcal{C}^{m \times n}$ is the measurement matrix



Nonlinearity makes peeling challenging



Main Results

- ***Sparse-graph*** codes for Compressive Phase Retrieval: ***PhaseCode***

- ***Fast*** & ***efficient***: *first 'capacity-approaching' results*

	Sample complexity	Computational complexity
Noiseless	$\mathbf{4K}$ (or $14K$)	$O(K)$
Noisy (almost-linear)	$O(K \log n)$	$O(N \log n)$
Noisy (sub-linear)	$O(K \log^3 n)$	$\boldsymbol{O(K \log^3 n)}$

- Design can be made ***'Optics-Friendly'***

- Extensive ***simulations*** validate close tie between theory & practice

Pedarsani, Yin, Lee, R., 2014

Simulation Results

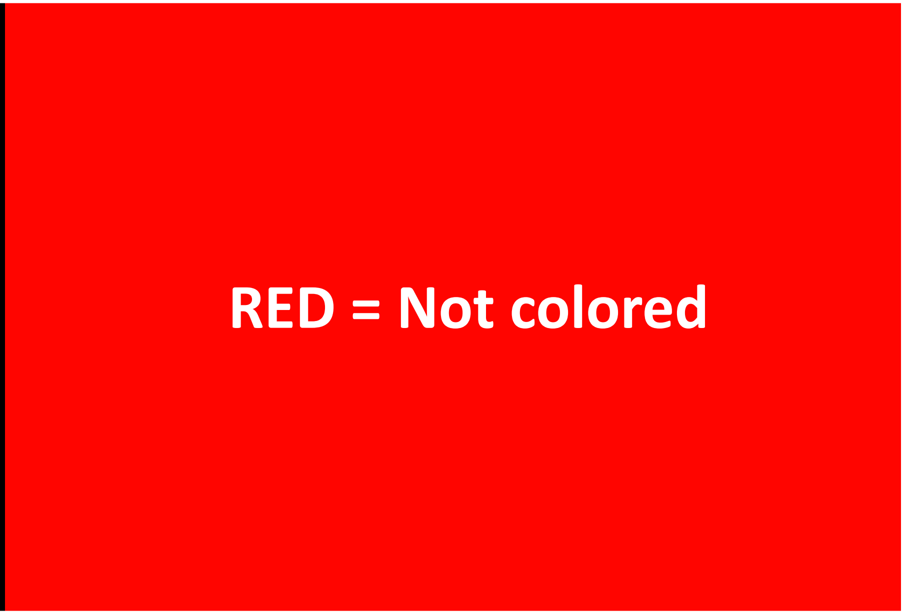
x = 2D FFT coefficients of

$$x \rightarrow \boxed{A} \xrightarrow{Ax} \boxed{|.|} \xrightarrow{y} \boxed{\text{Decoder}} \rightarrow \hat{x}$$

Simulation Results

Iteration 0

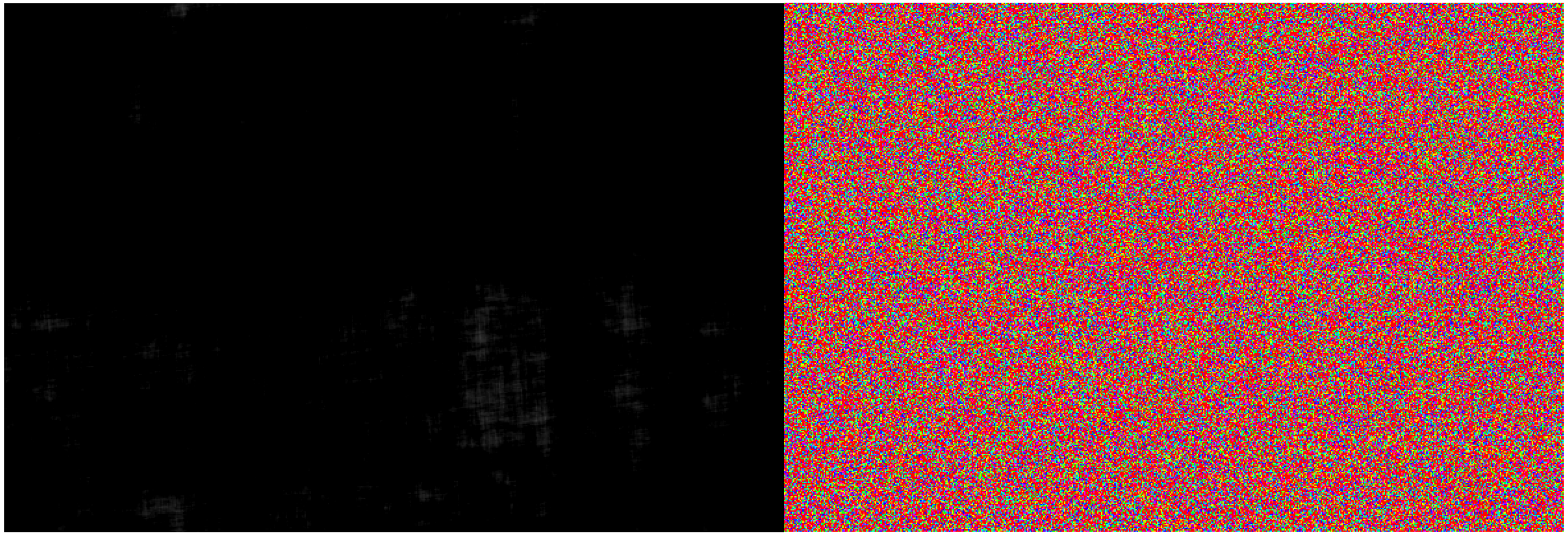
IFFT of recovered FFT coefficients



Color of balls

Simulation Results

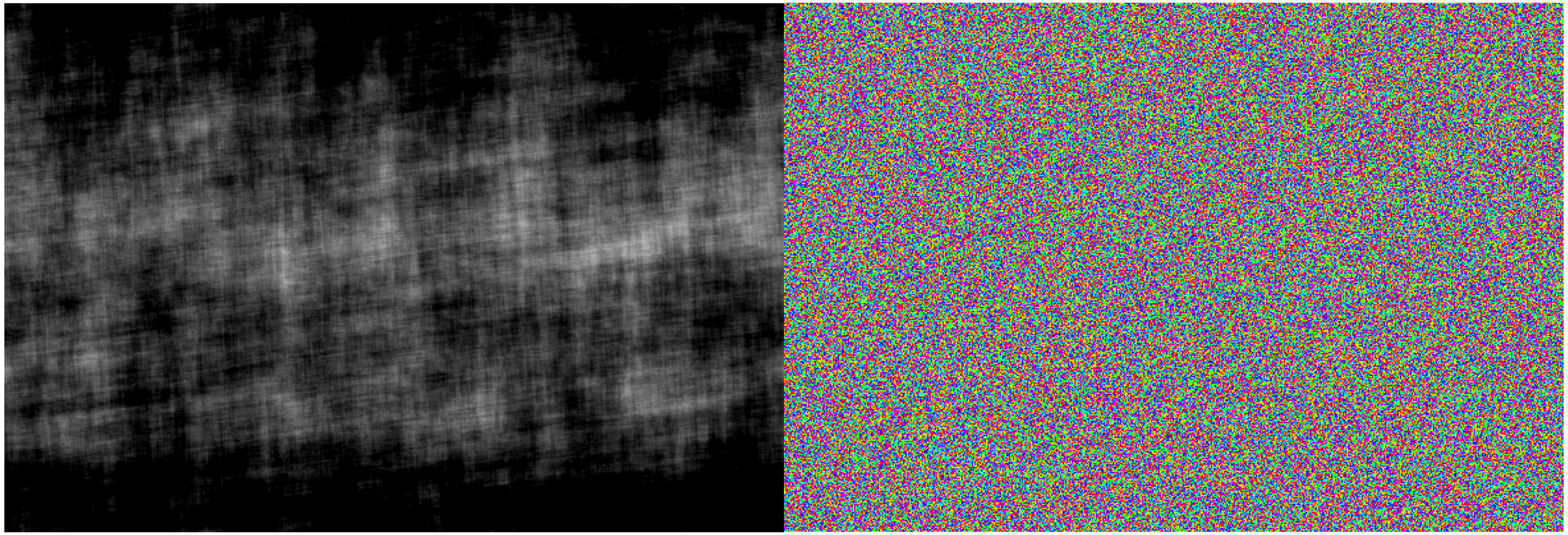
Iteration 1



Some balls are colored!

Simulation Results

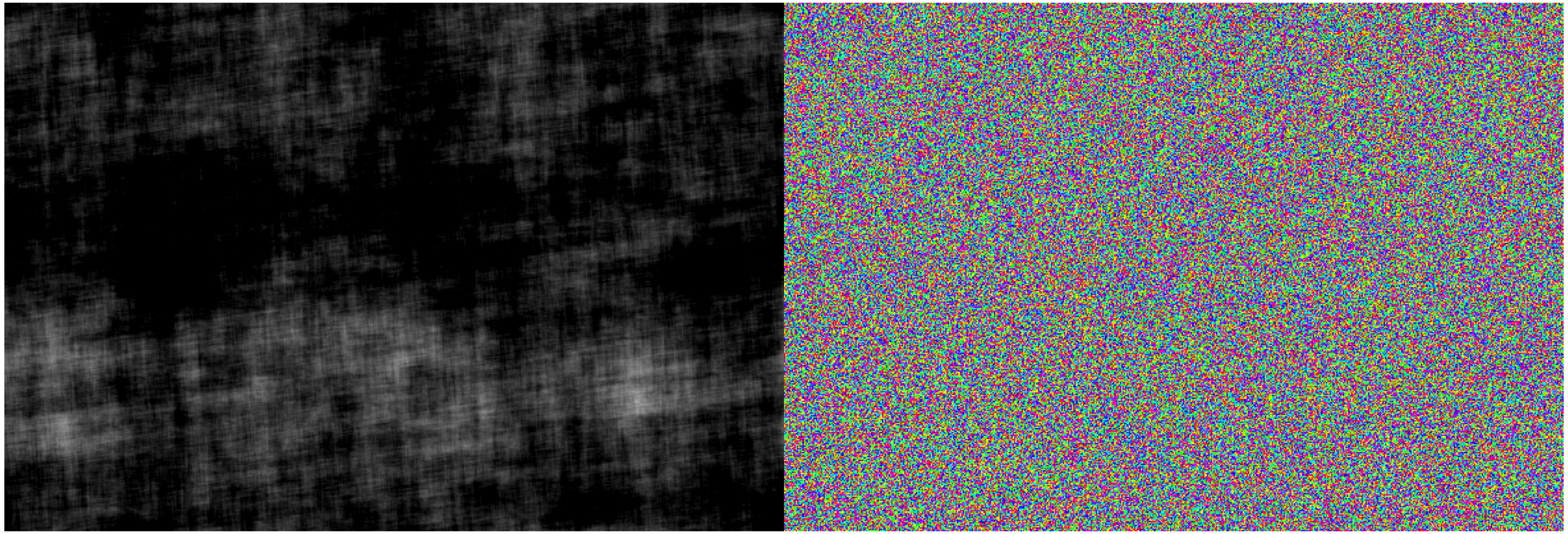
Iteration 2



More balls are colored!

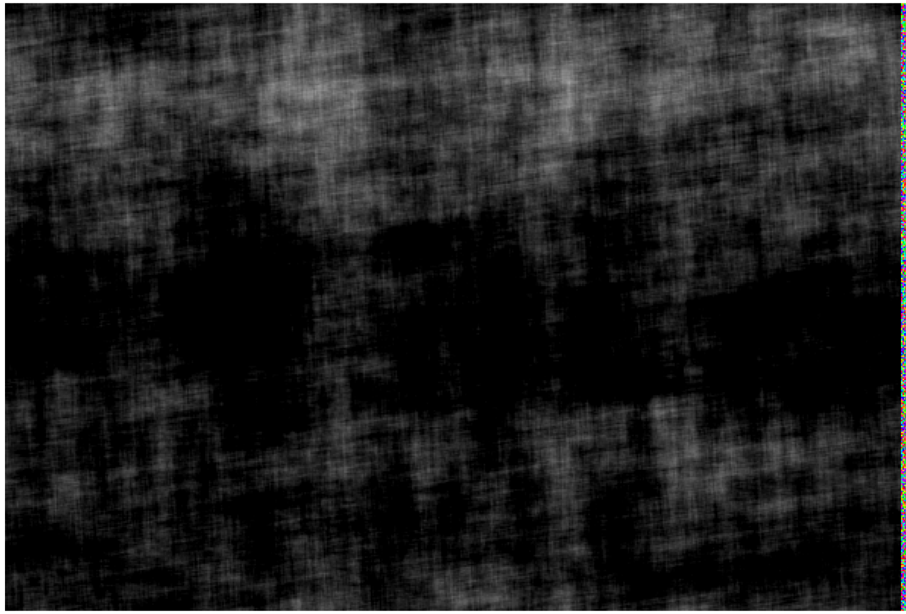
Simulation Results

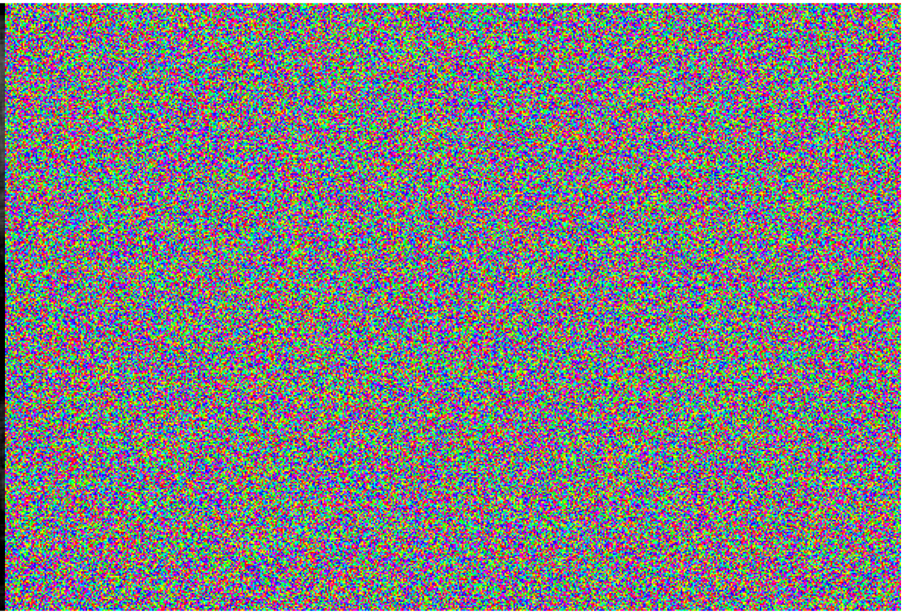
Iteration 3



Simulation Results

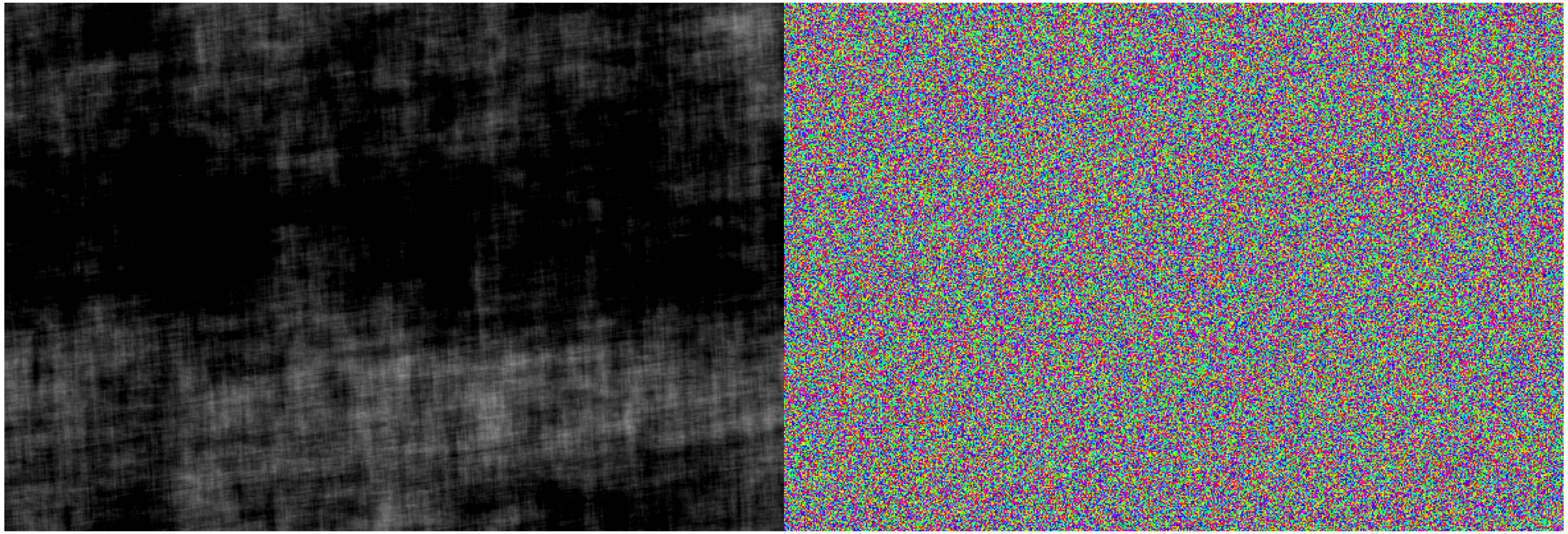
Iteration 4





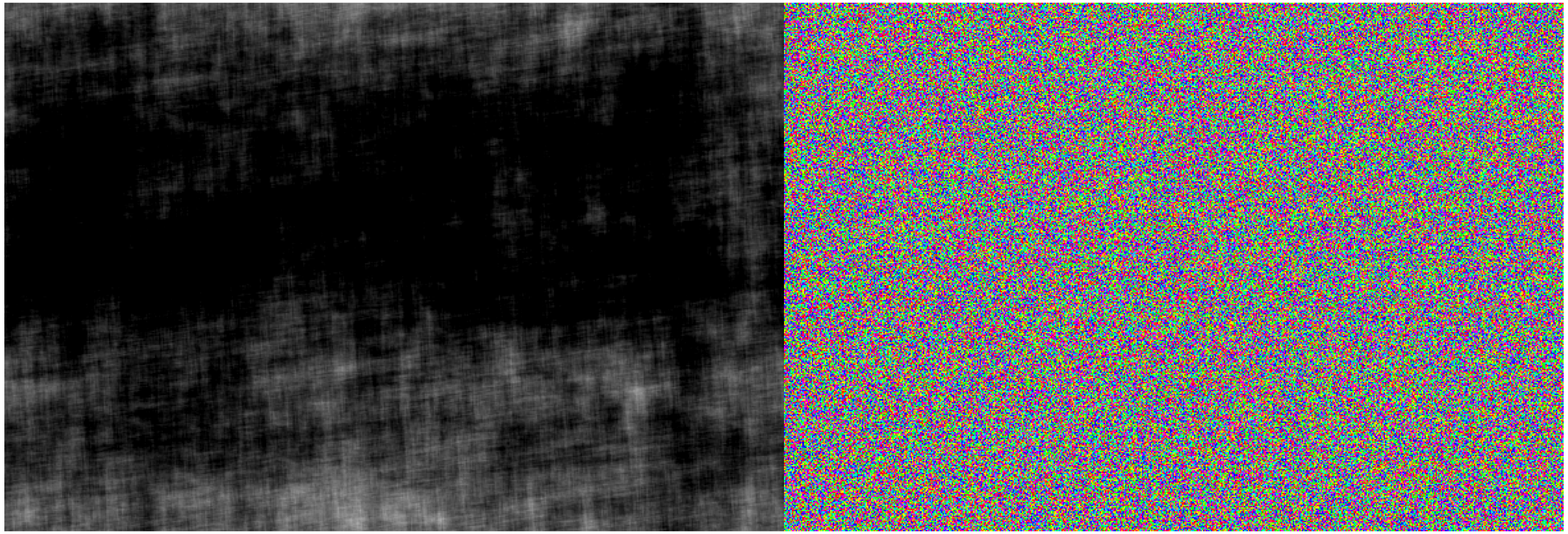
Simulation Results

Iteration 5



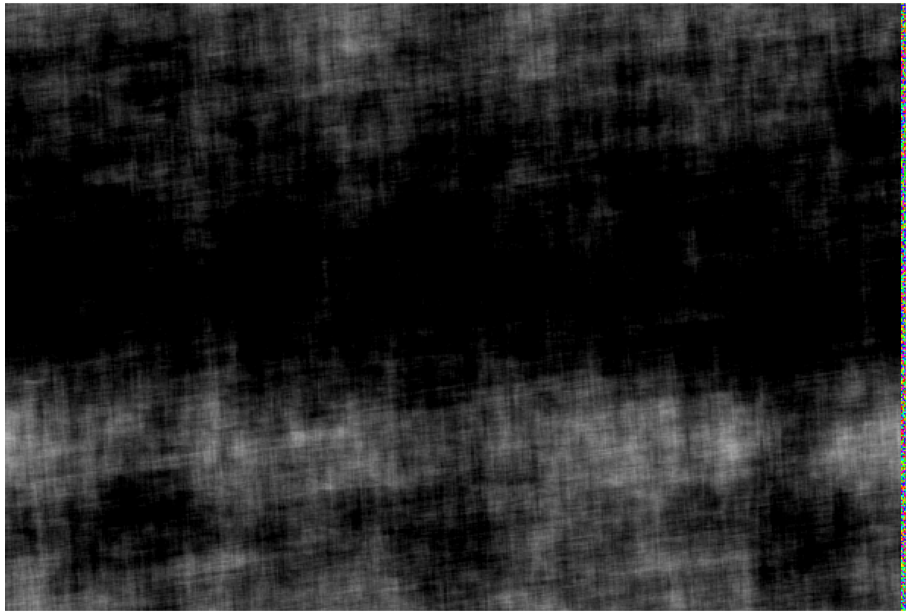
Simulation Results

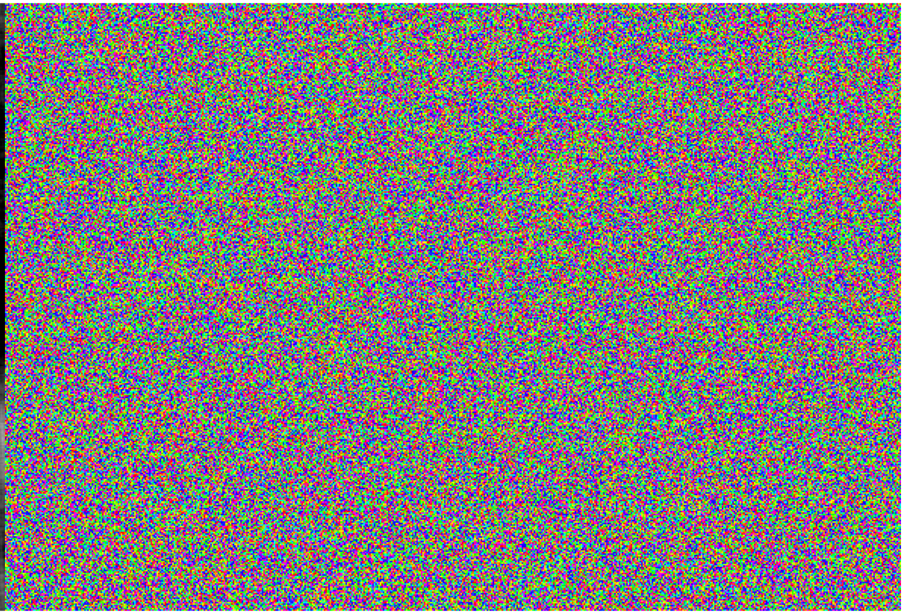
Iteration 6



Simulation Results

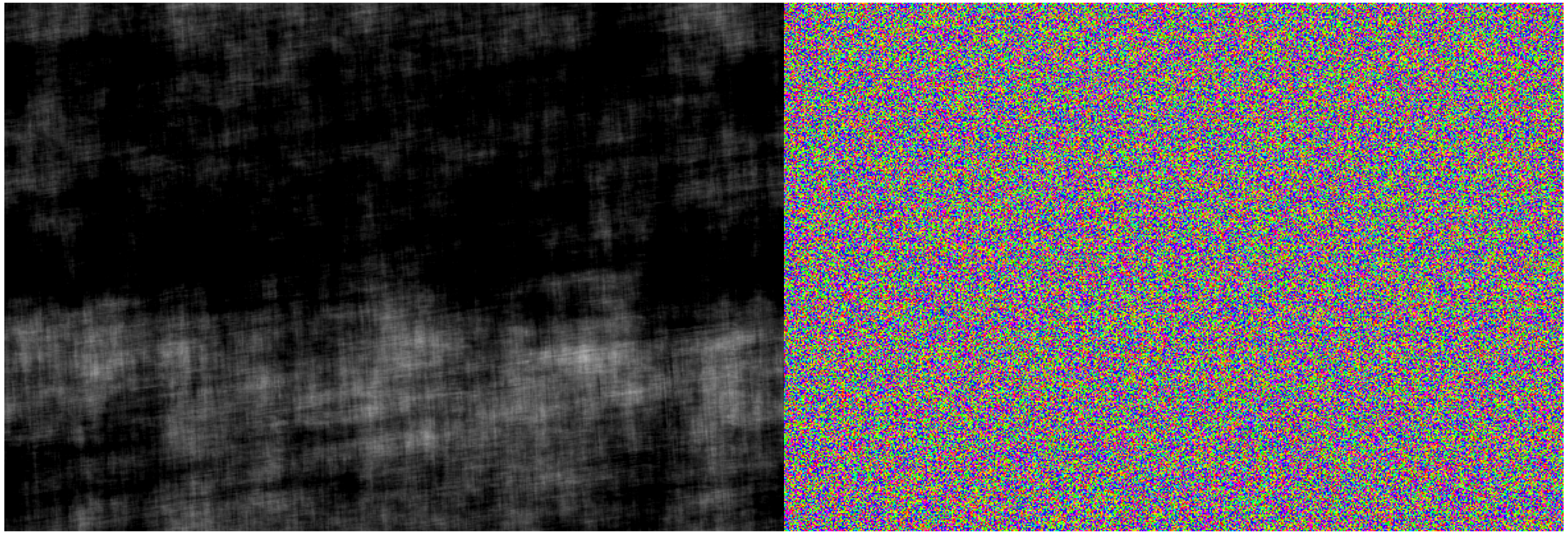
Iteration 7





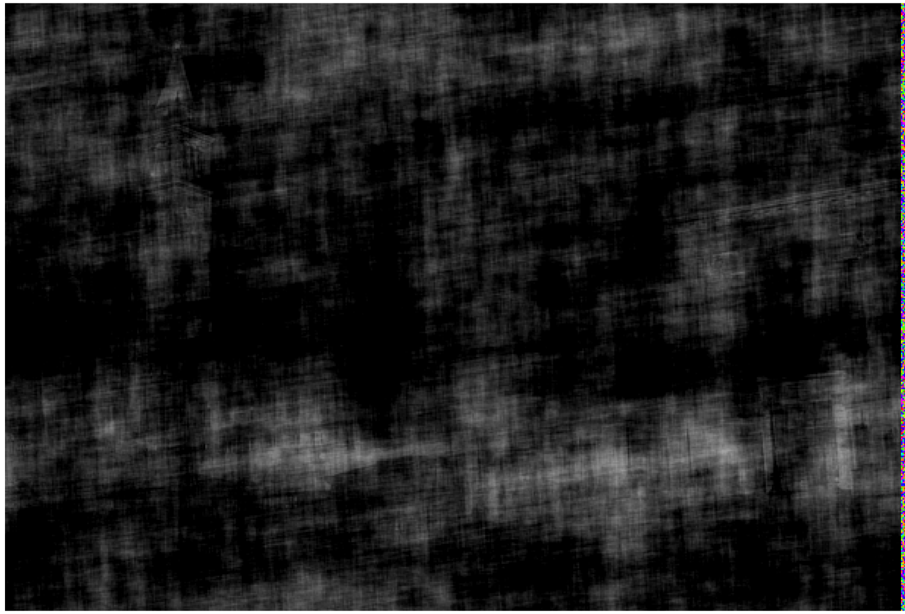
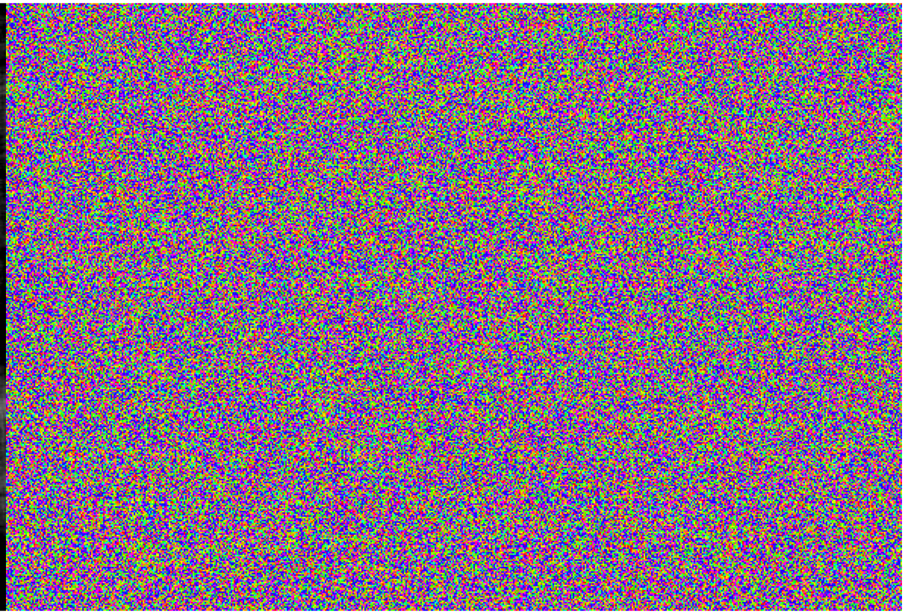
Simulation Results

Iteration 8



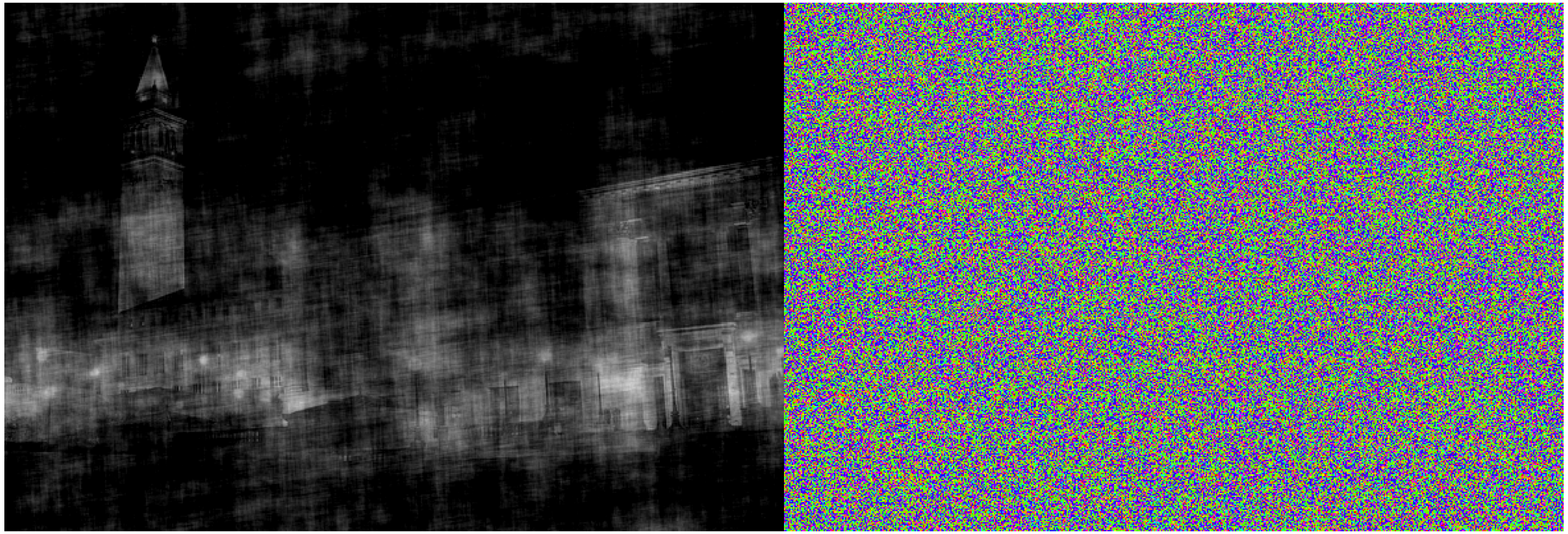
Simulation Results

Iteration 9

Simulation Results

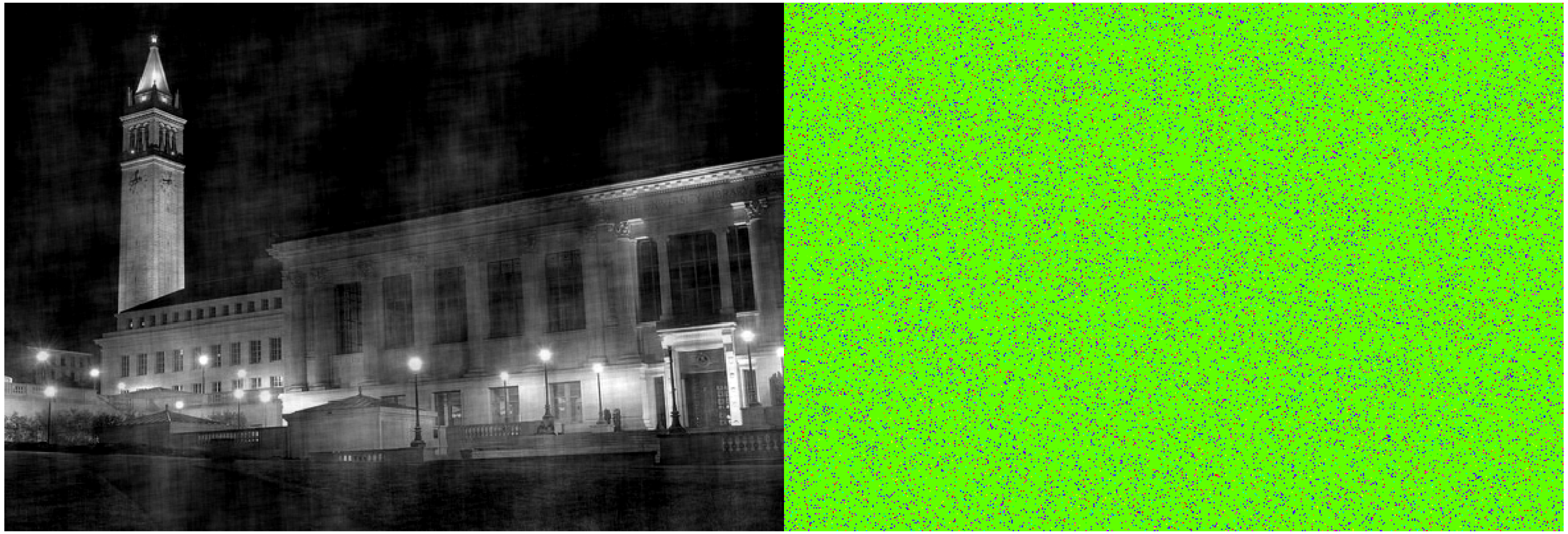
Iteration 10



GREEN becomes dominant?

Simulation Results

Iteration 11



Most balls are **GREEN**

Simulation Results

Iteration 12

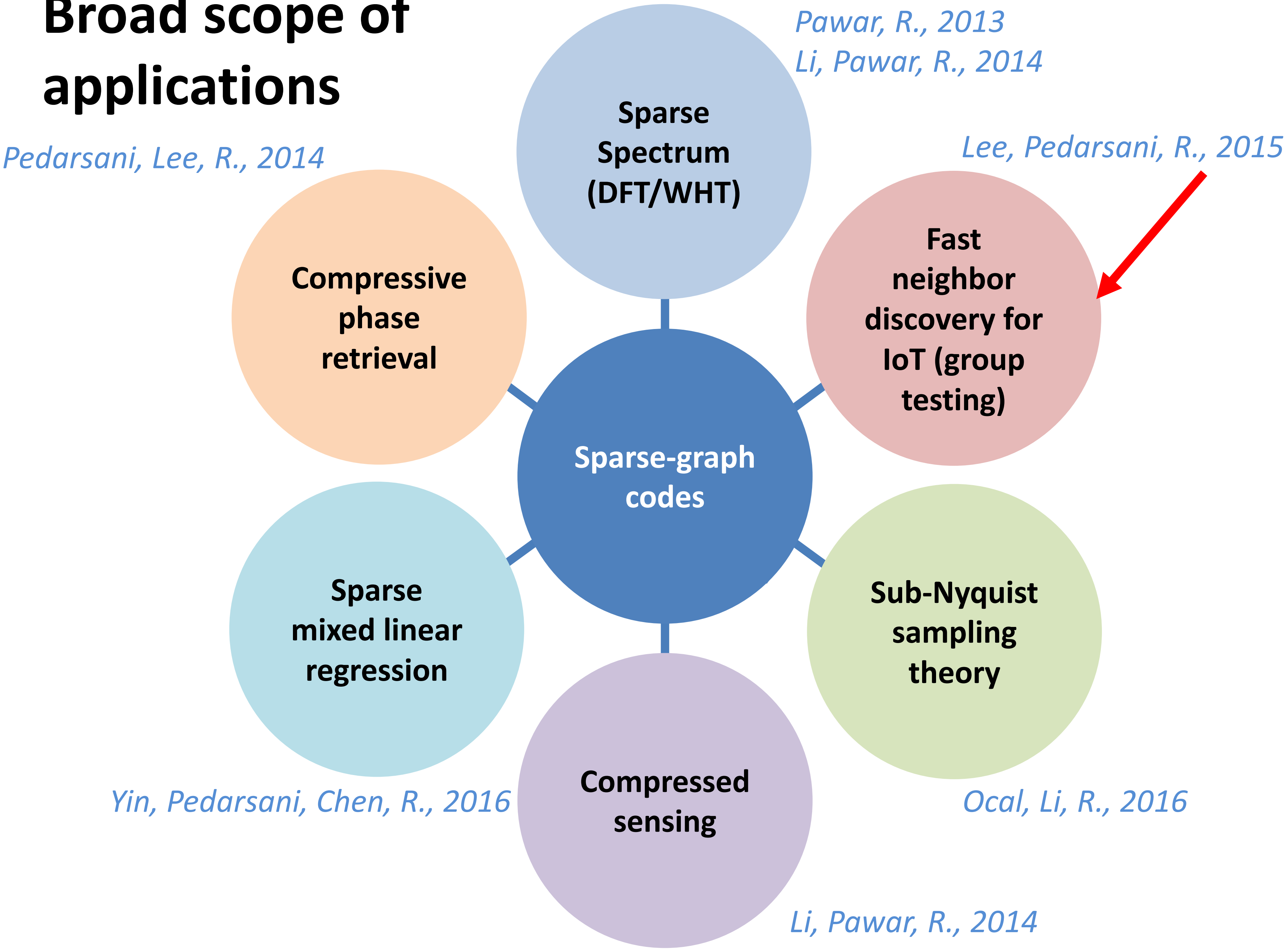
All but 1 ball are **GREEN**

Simulation Results

Iteration 13

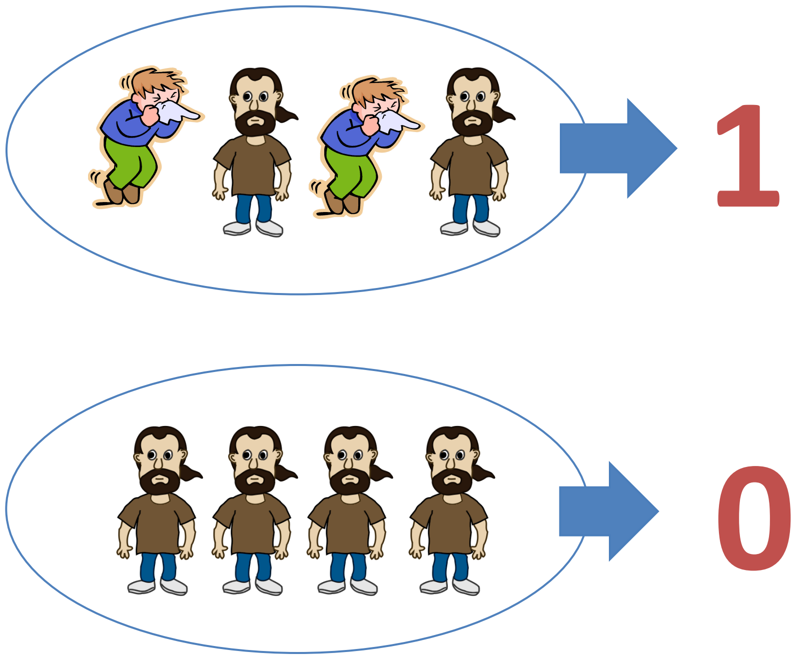
All balls are **GREEN!**

Broad scope of applications



Group testing

Find **K defective** from **n** items using ‘group’ measurements

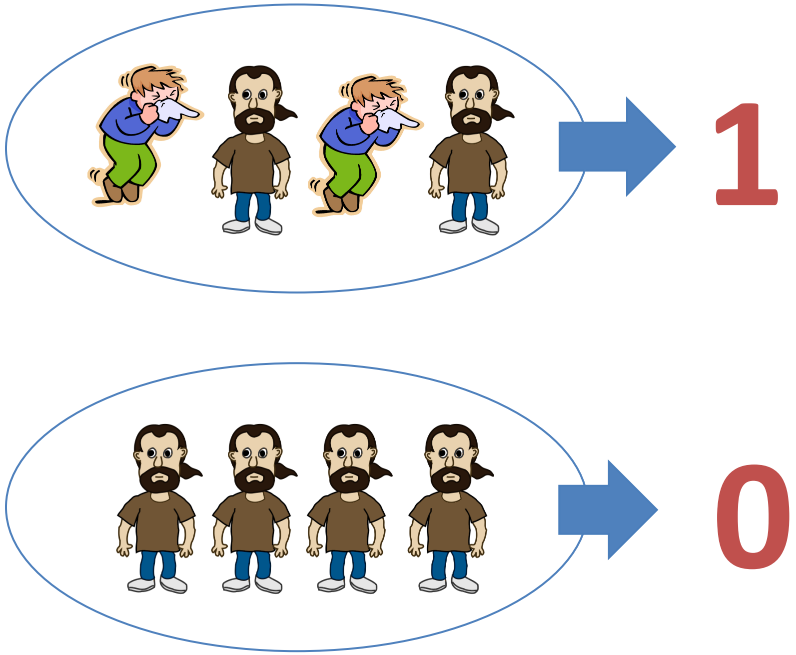


Jack Keil Wolf

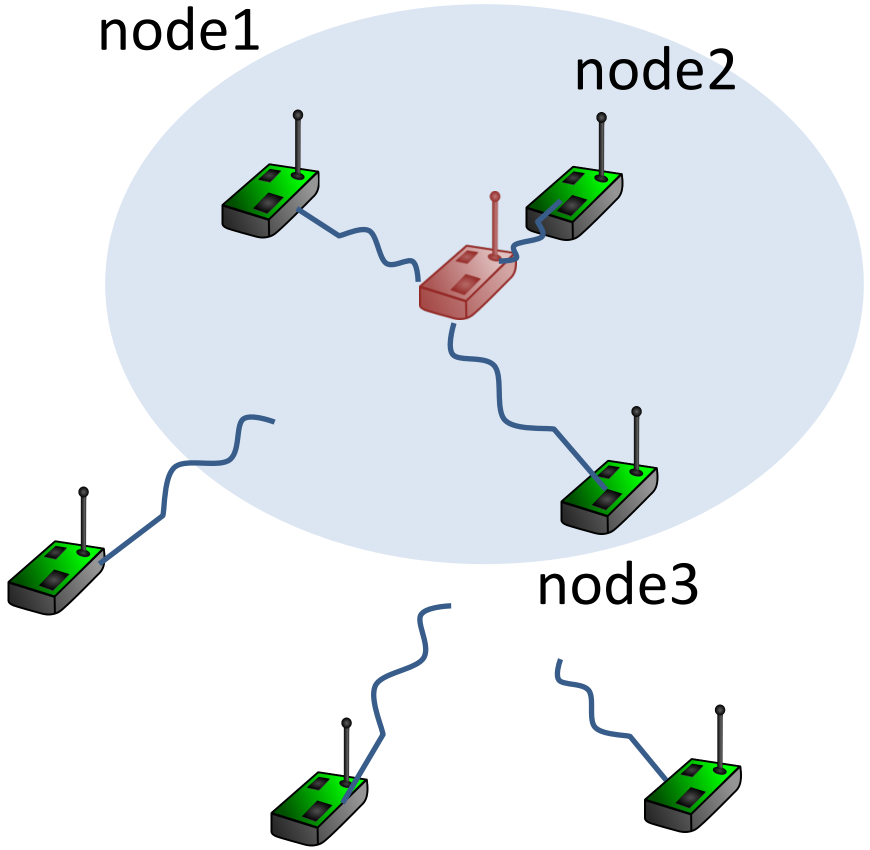
[85] Principles of **group testing** and an application of the design and analysis of multi-access protocols
[85] Born again **group testing**: multi-access communications
[84] Random multiple-access communications and **group testing**
[81] An Application of **Group Testing** to the Design of Multi-User Protocols

Group testing

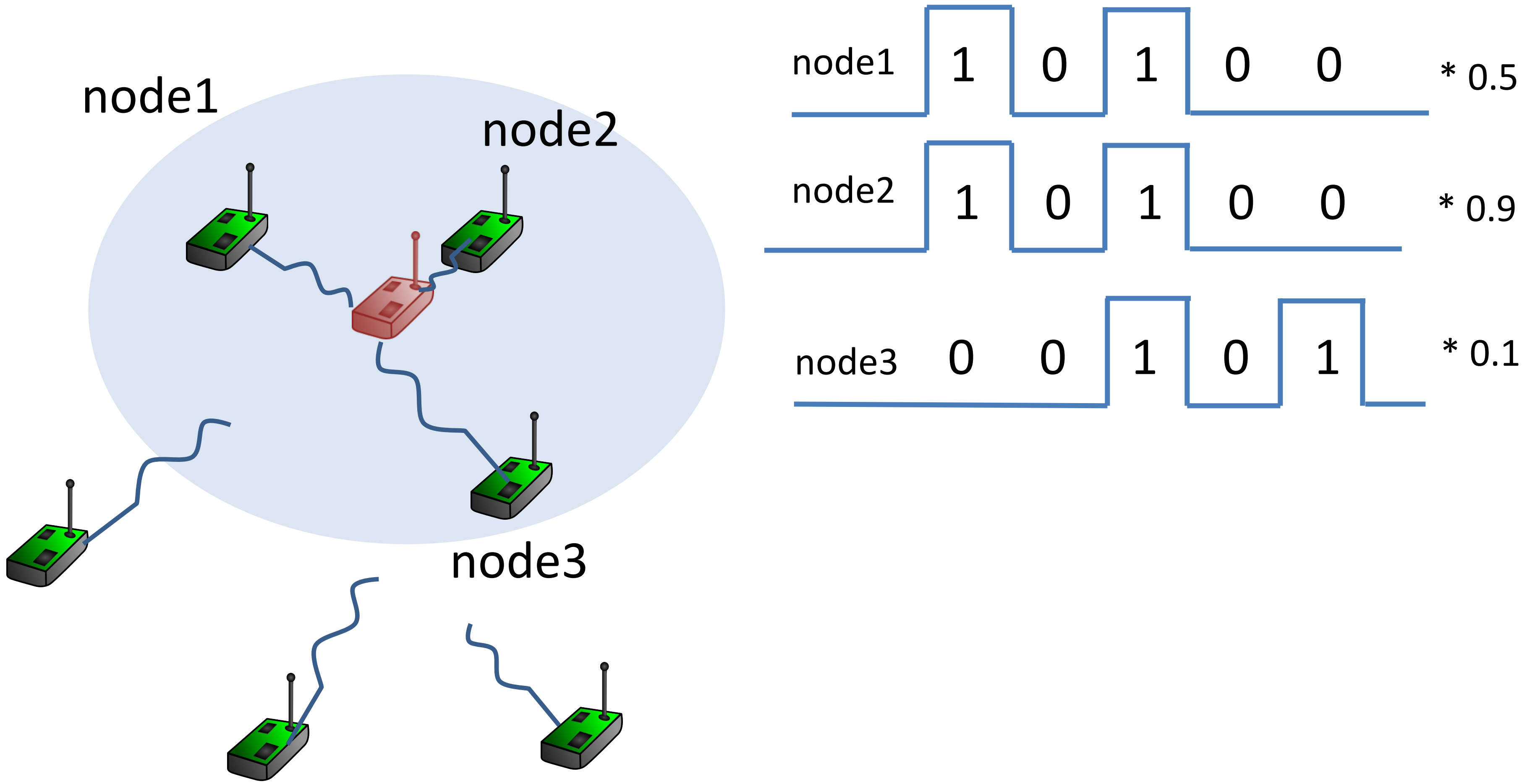
Find **K** defective from **n** items using 'group' measurements



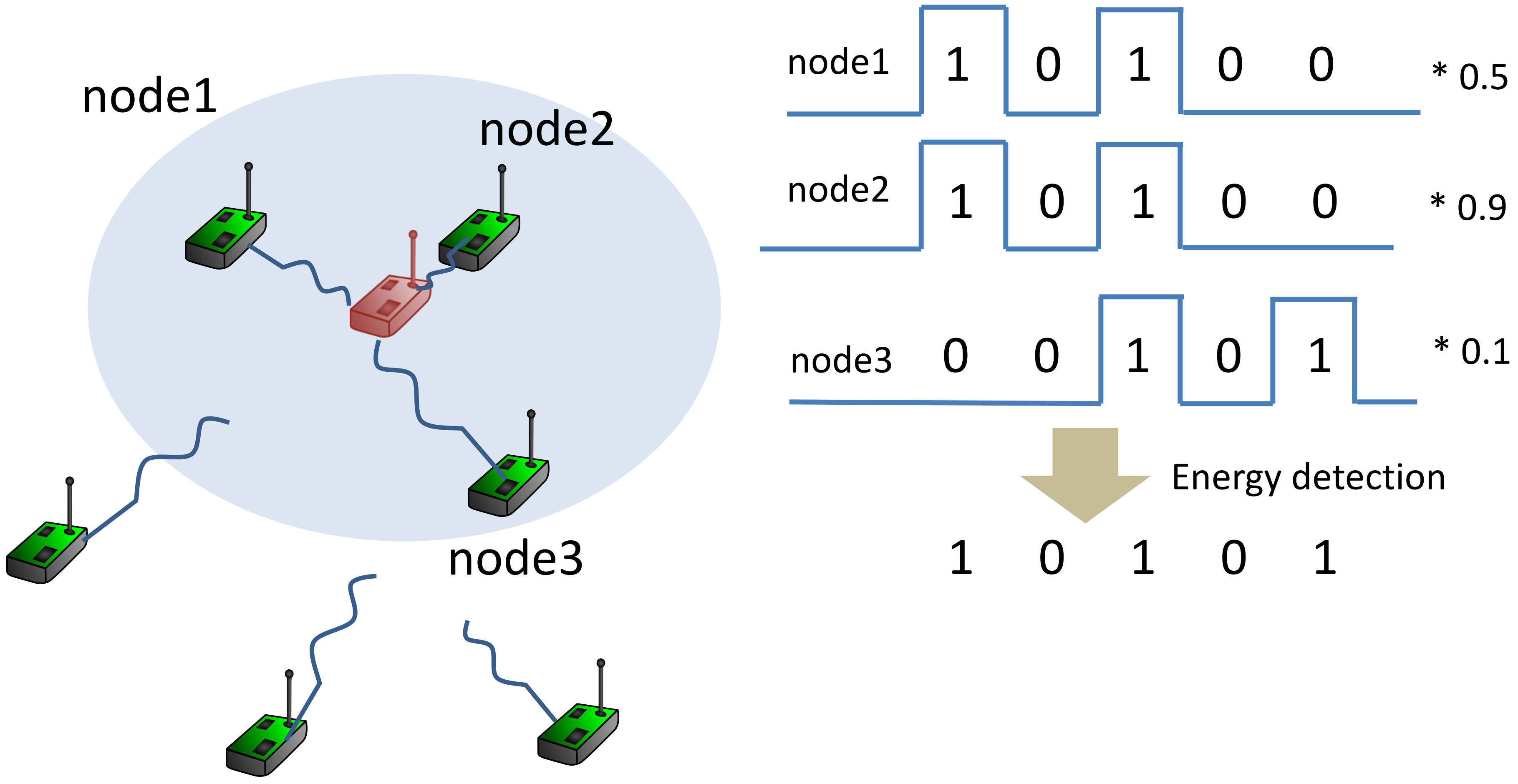
Group Testing for Neighbor Discovery



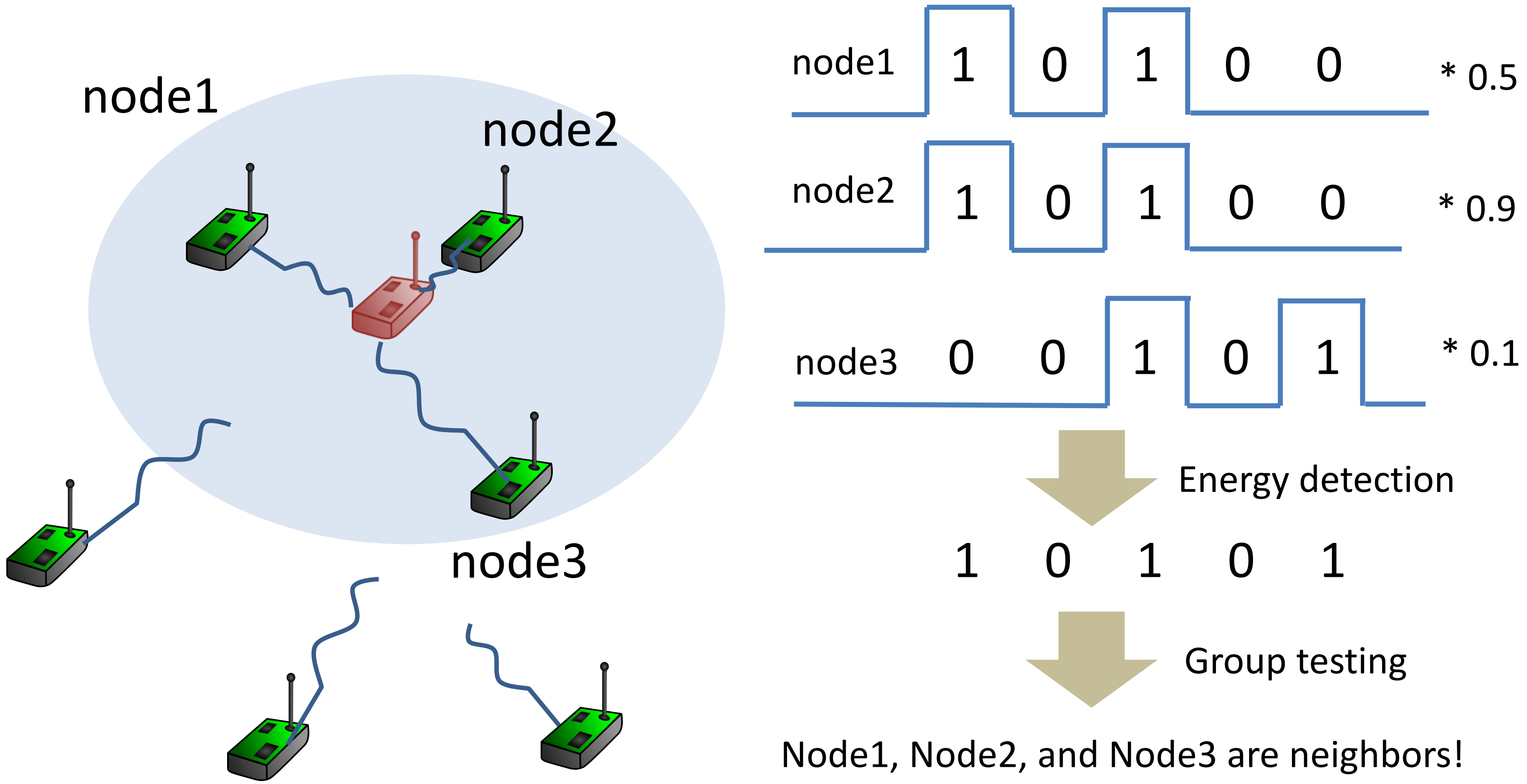
Group Testing for Neighbor Discovery



Group Testing for Neighbor Discovery



Group Testing for Neighbor Discovery



SAFFRON

(Sparse-grAph codes Framework For gROup testiNg)

Thm: With $6C(\epsilon)K\log_2 n$ tests, SAFFRON recovers at least $(1-\epsilon)K$ defective items with probability $1-O\left(\frac{K}{n^2}\right)$ by performing $O(K\log n)$ computations.

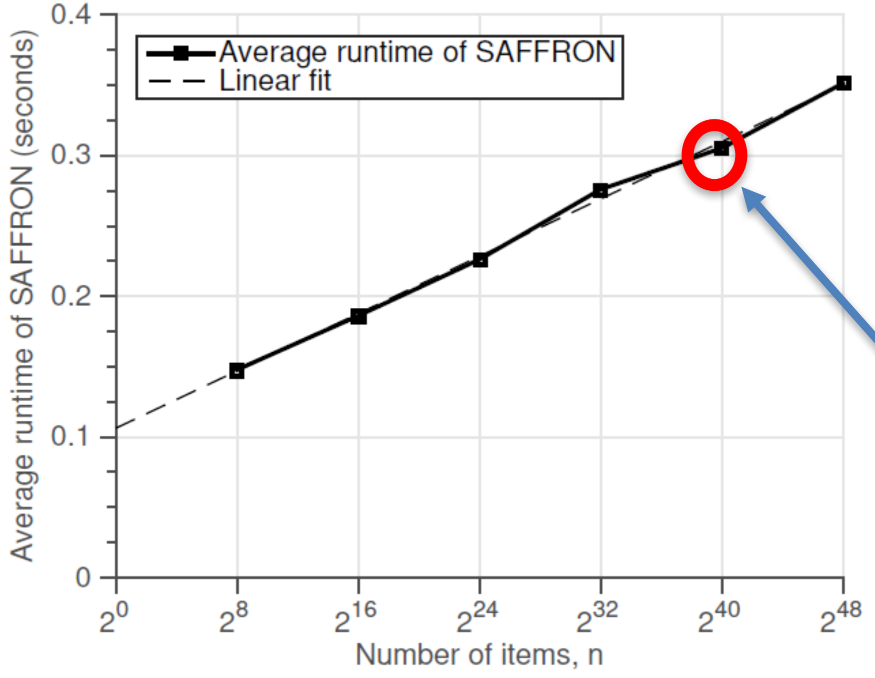
Example: SAFFRON $(\epsilon = 10^{-6}, C(\epsilon) = 11.3)$

With $68K\log_2 n$ tests, SAFFRON recovers at least $(1-10^{-6})K$ defective items with probability $1-O\left(\frac{K}{n^2}\right)$ with a decoding time complexity of $O(K\log n)$.

Lee, Pedarsani, Chandrasekher, R., 2015

SAFFRON

(Sparse-grAph codes Framework For gROup testiNg)



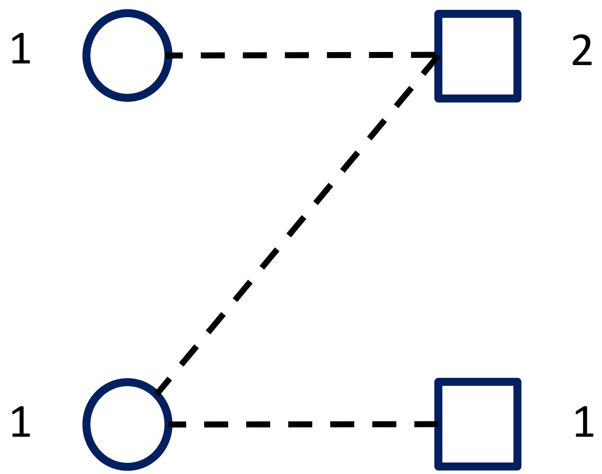
Run-time with $K = 2^5$ and varying n.

Simulation done on a regular MacBook Air laptop

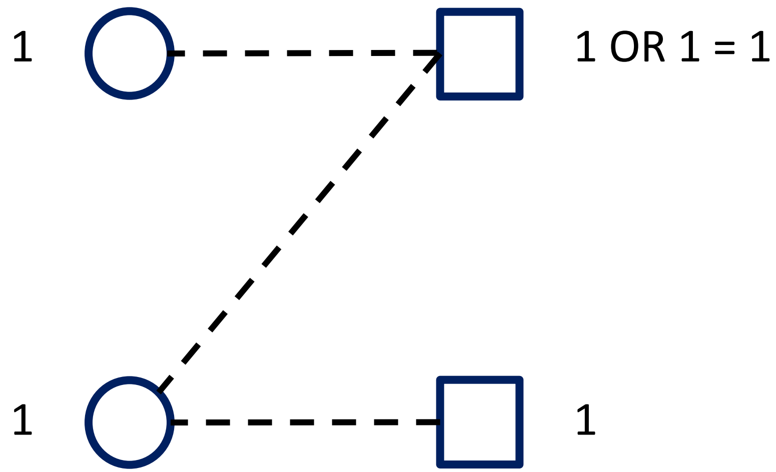
Finding 32 defective items from a population of size 1 trillion can be done with SAFFRON using ~87,000 tests in 0.3 second on a regular MacBook Air laptop!

Peeling with OR operation

Under field arithmetic

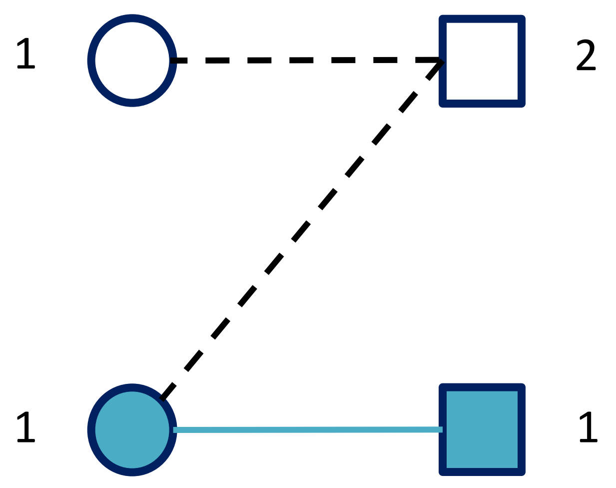


Under OR

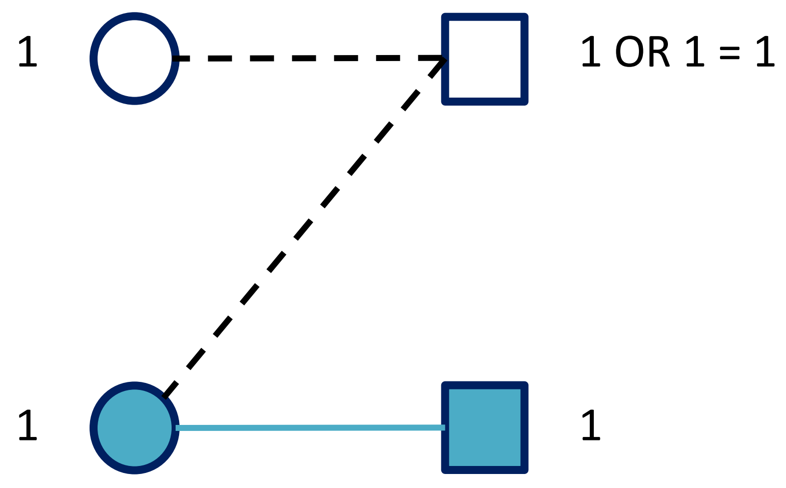


Challenge: Peeling with OR operation

Under field arithmetic

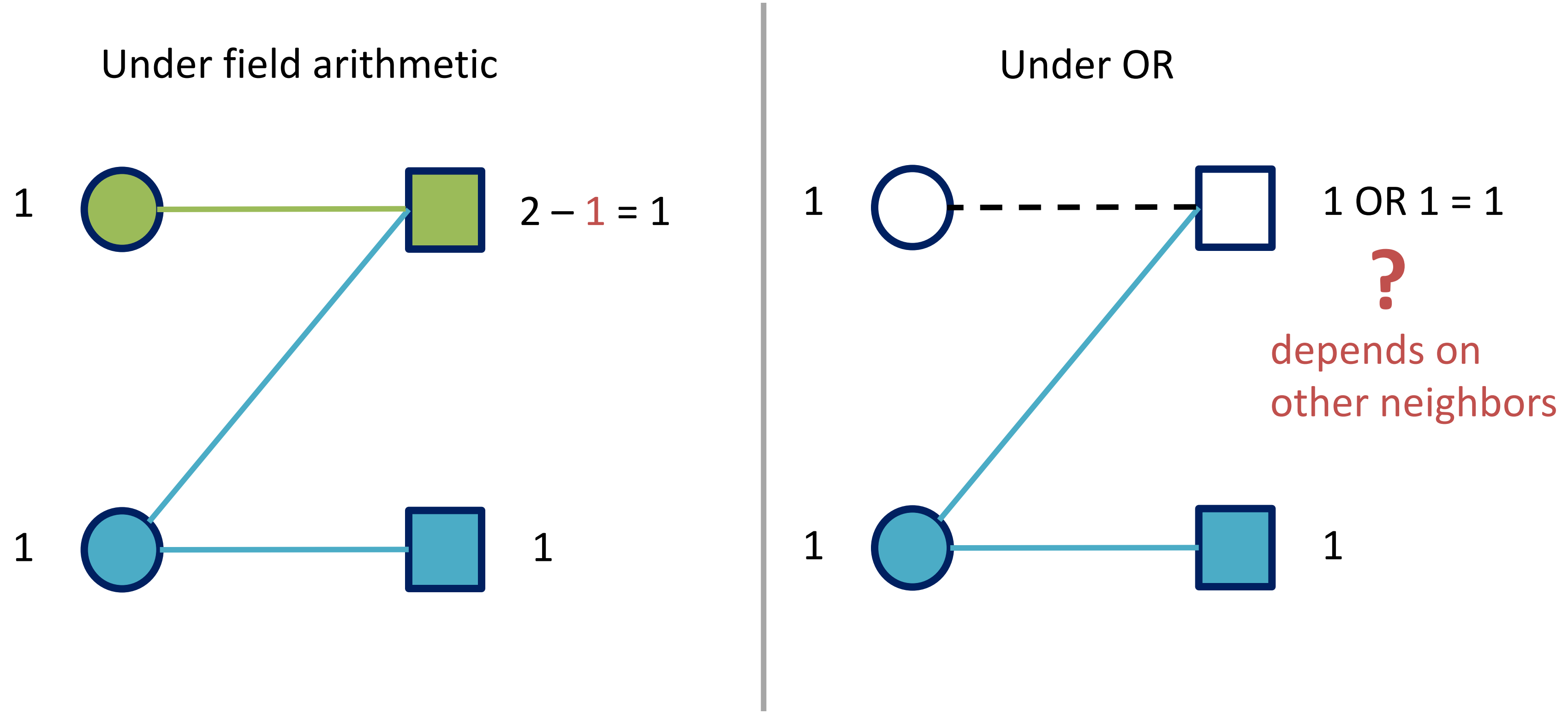


Under OR



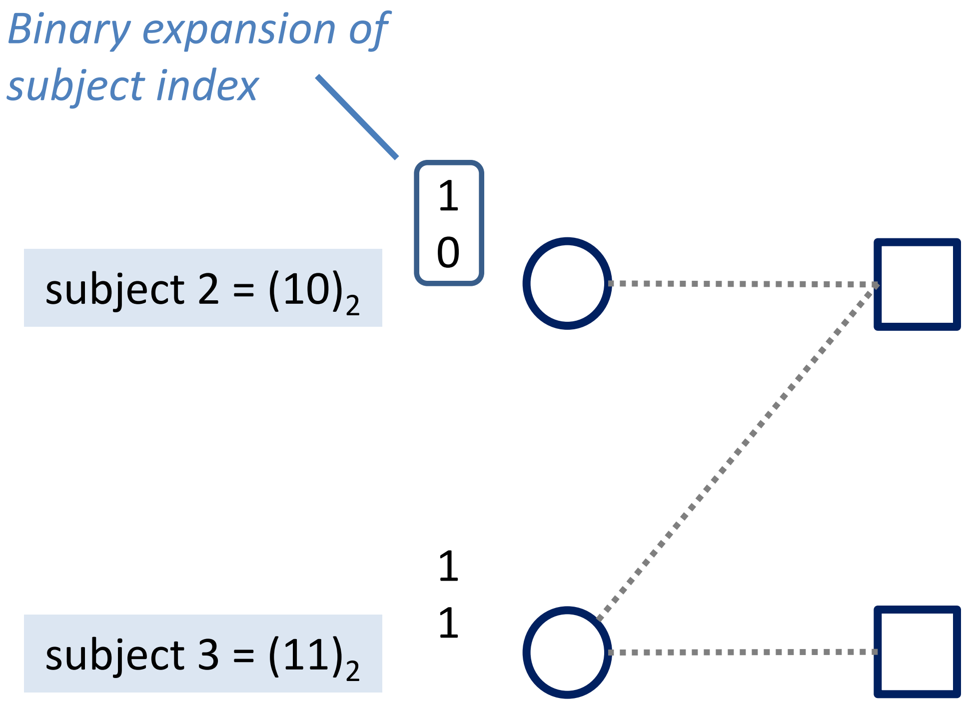
Find singleton measurement/test and recover the value

Peeling with OR operation

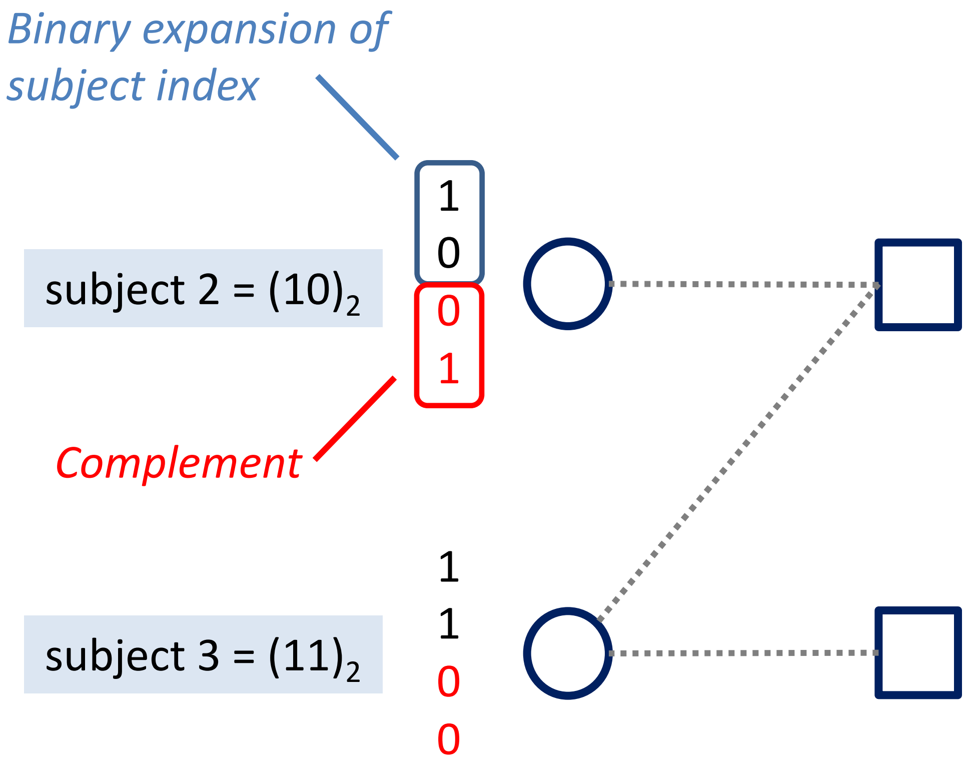


Nonlinearity makes peeling challenging

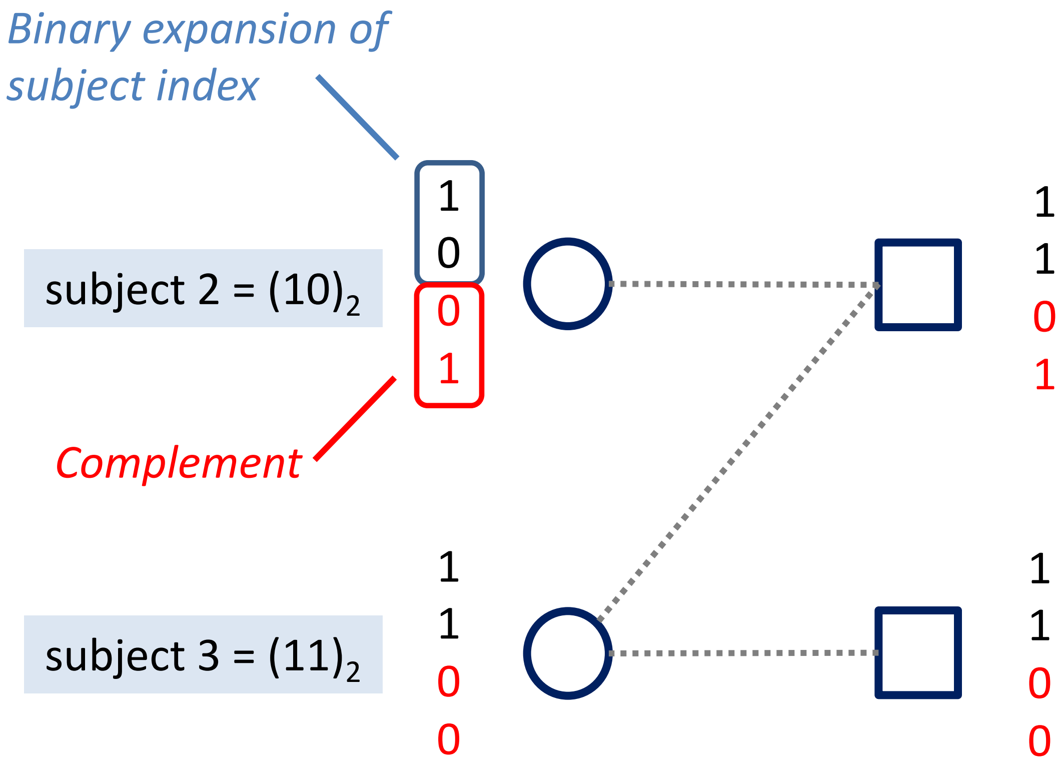
Solution – high level idea



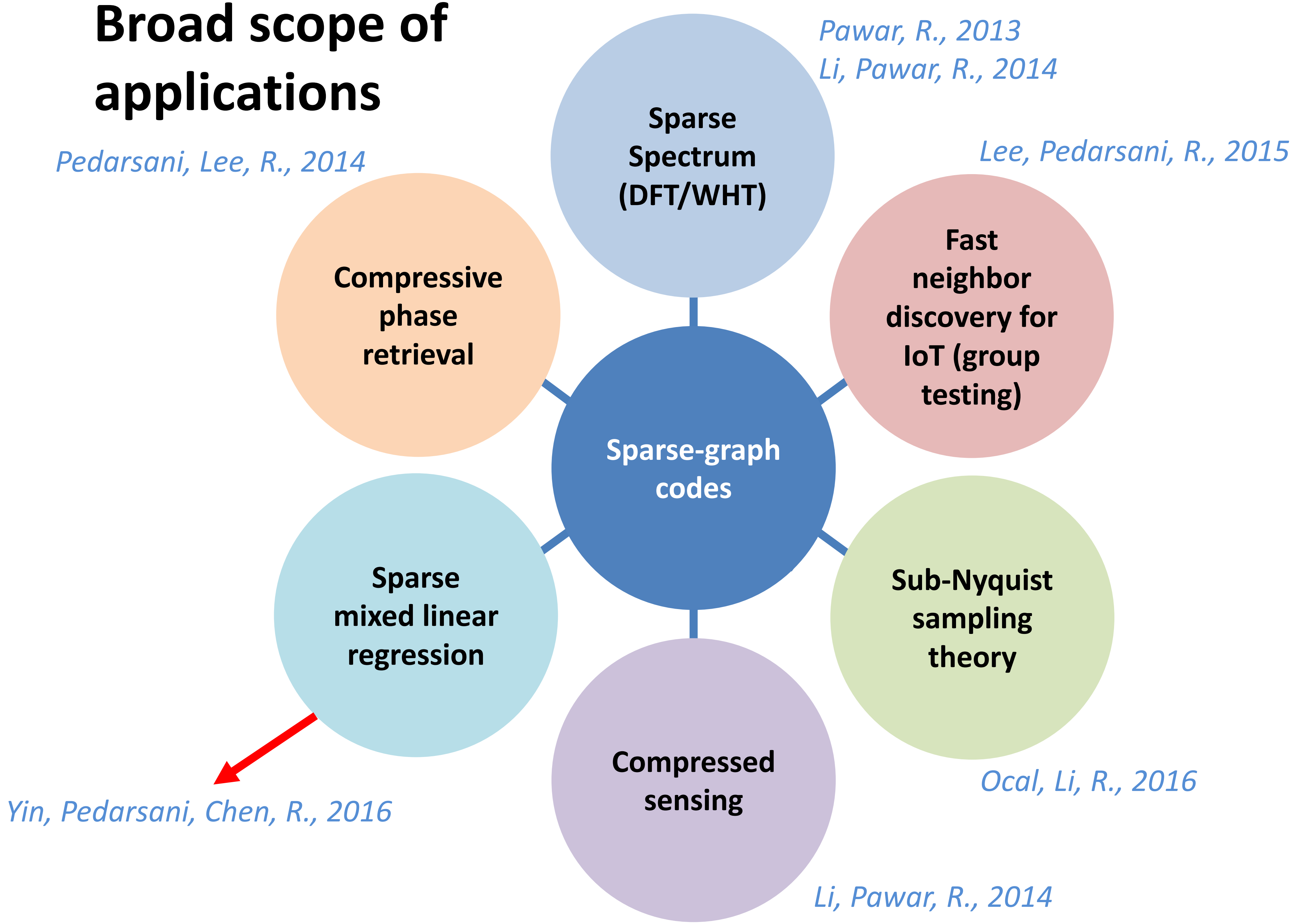
Solution – high level idea



Solution – high level idea

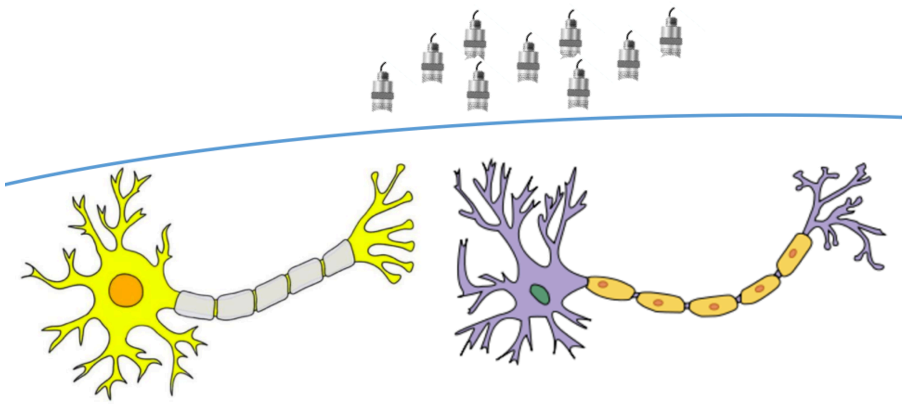


Broad scope of applications

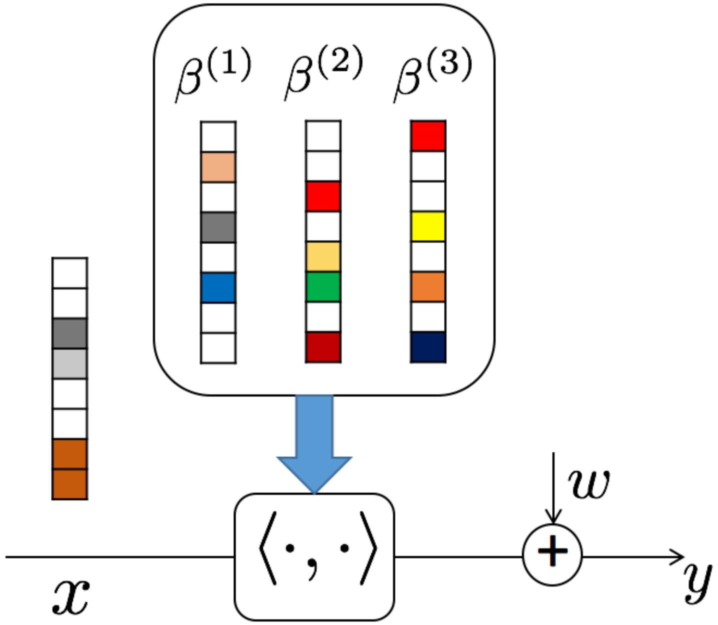


Motivation

- Compressive sensing: a powerful tool for sparse recovery.
- What if we have a *mixture* of sparse signals?
- Applications: Neuroscience, experiment design in biology…



Problem Formulation

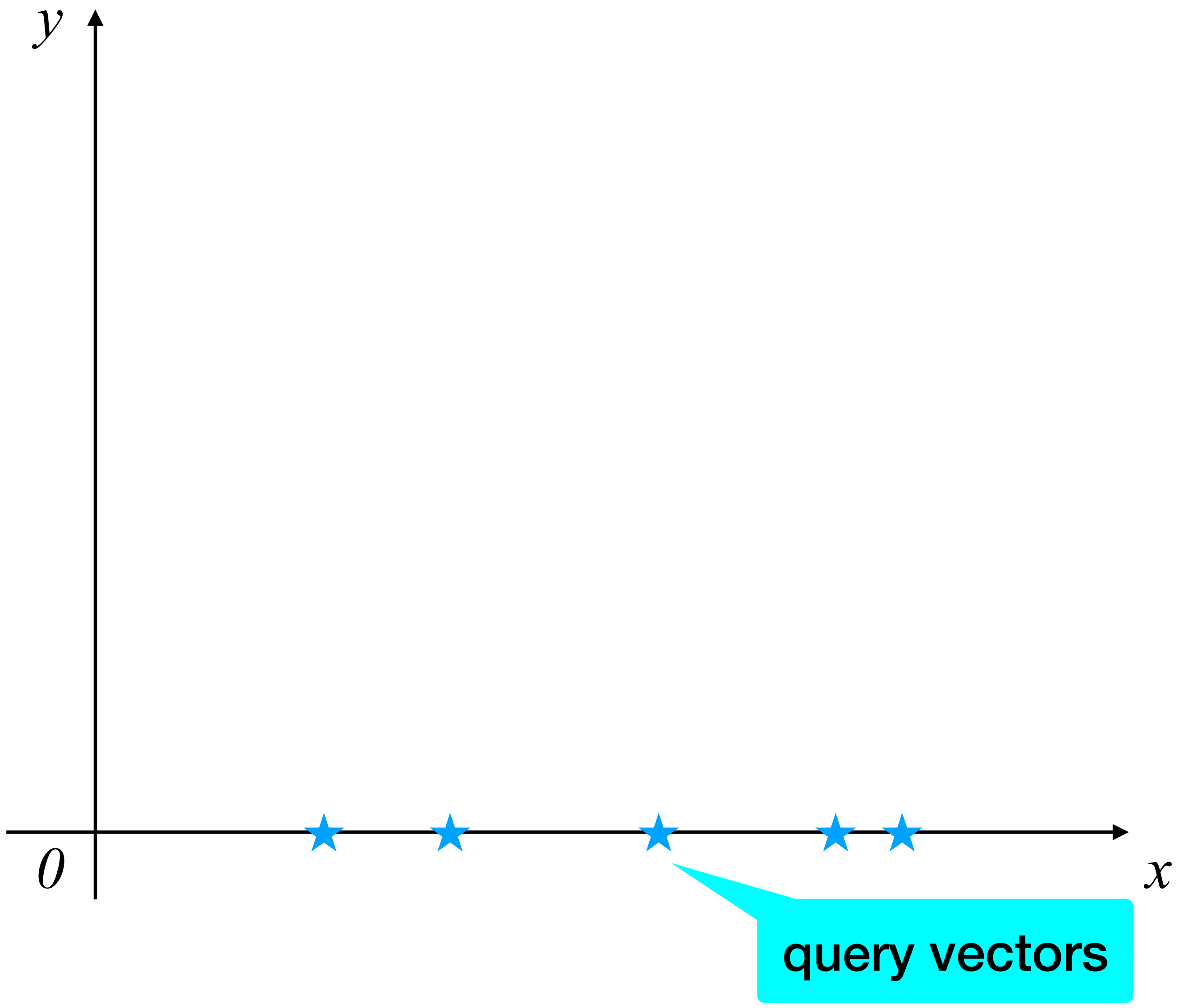


- L-class mixture of sparse linear regressions.
- Sparse parameter vectors $\beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(L)} \in \mathbb{C}^n$
- Total number of non-zero elements K ($\ll Ln$)
- Design query vectors $x_1, x_2, \ldots, x_m \in \mathbb{C}^n$.
- Obtain measurements $y_i = \langle x_i, \beta^{(\ell)} \rangle + w_i$ with probability q_ℓ.
- No knowledge of *which* $\beta^{(\ell)}$ is associated with each measurement.

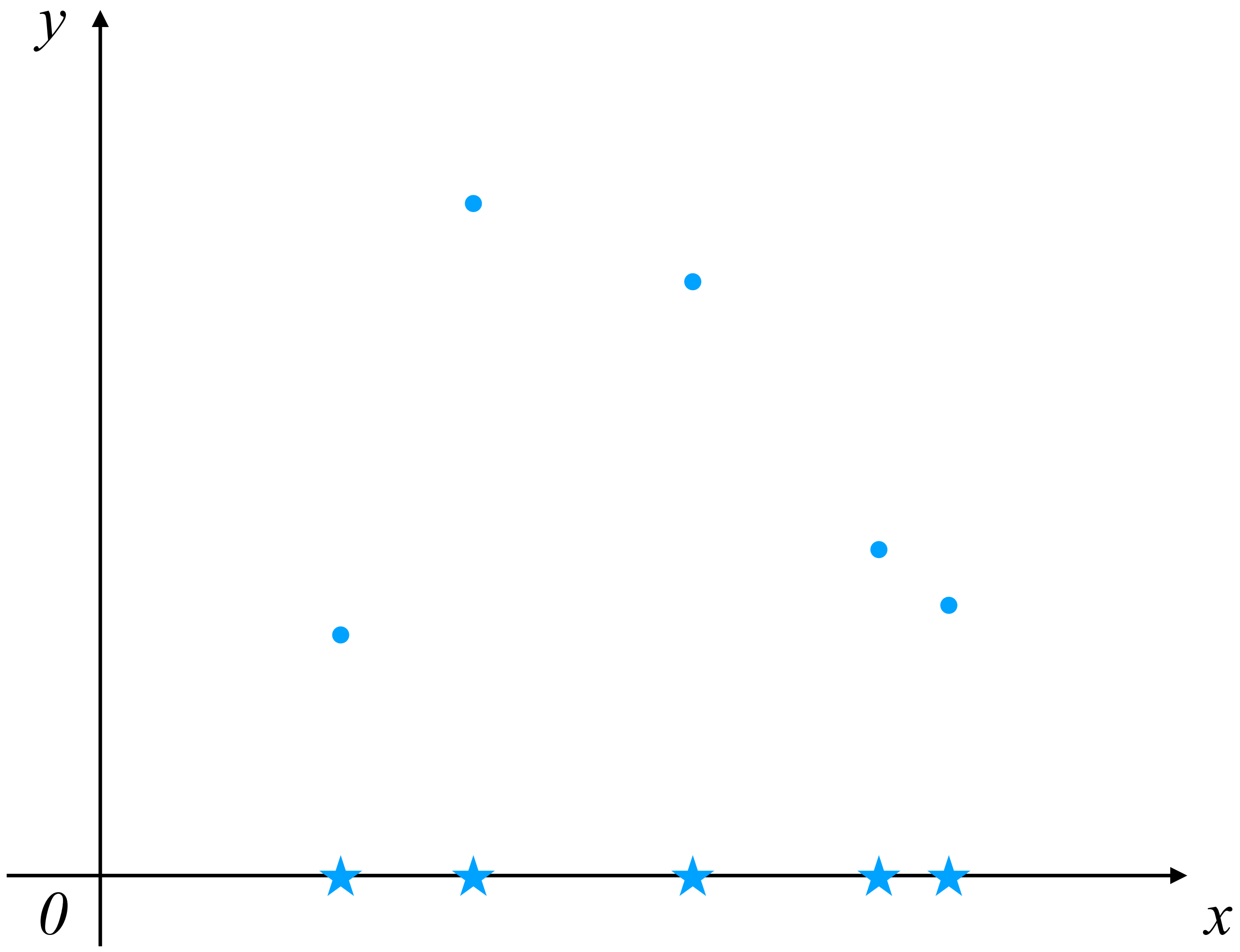
Simultaneous ***de-mixing*** and sparse parameter ***estimation*** problem!

Problem Formulation

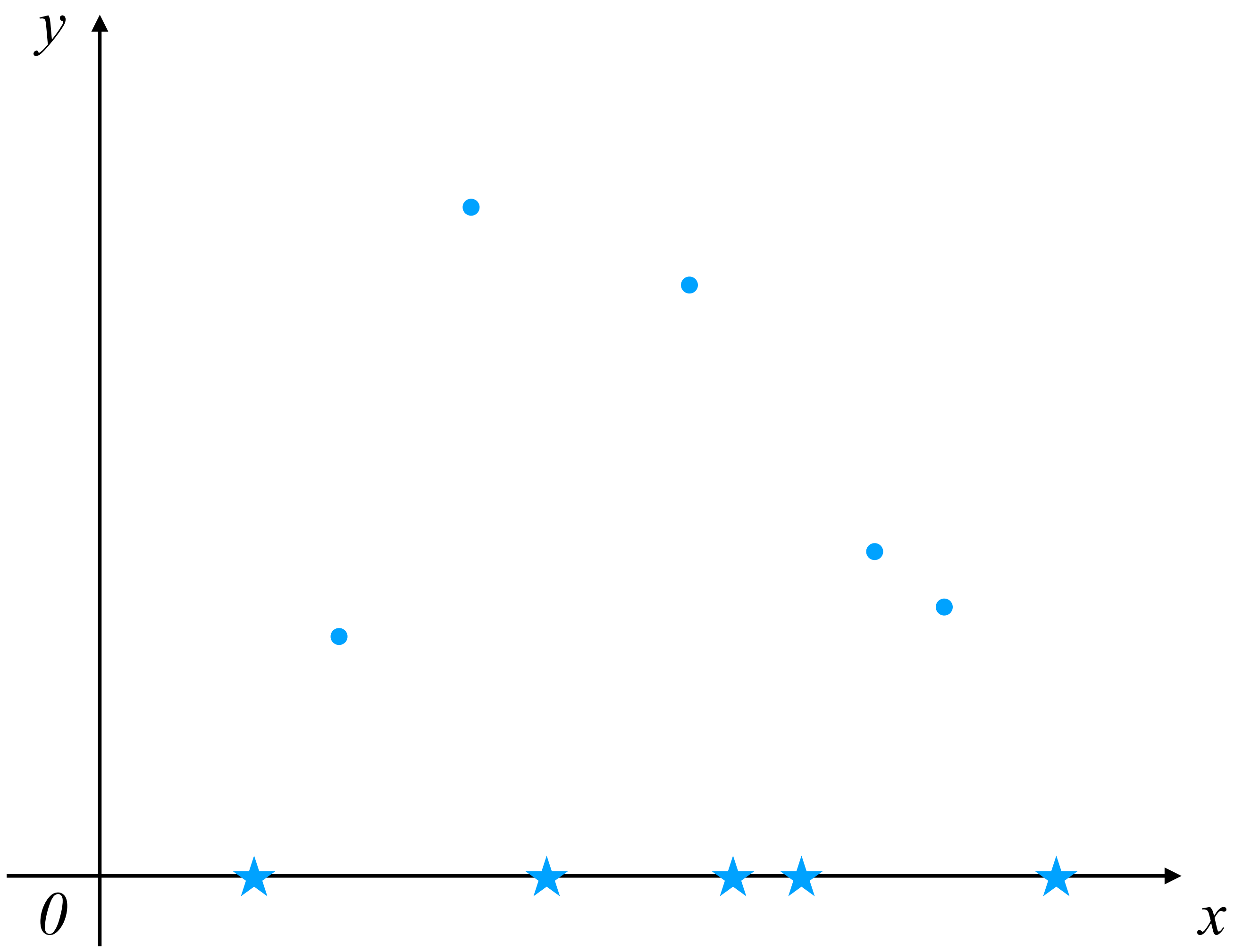
Problem Formulation



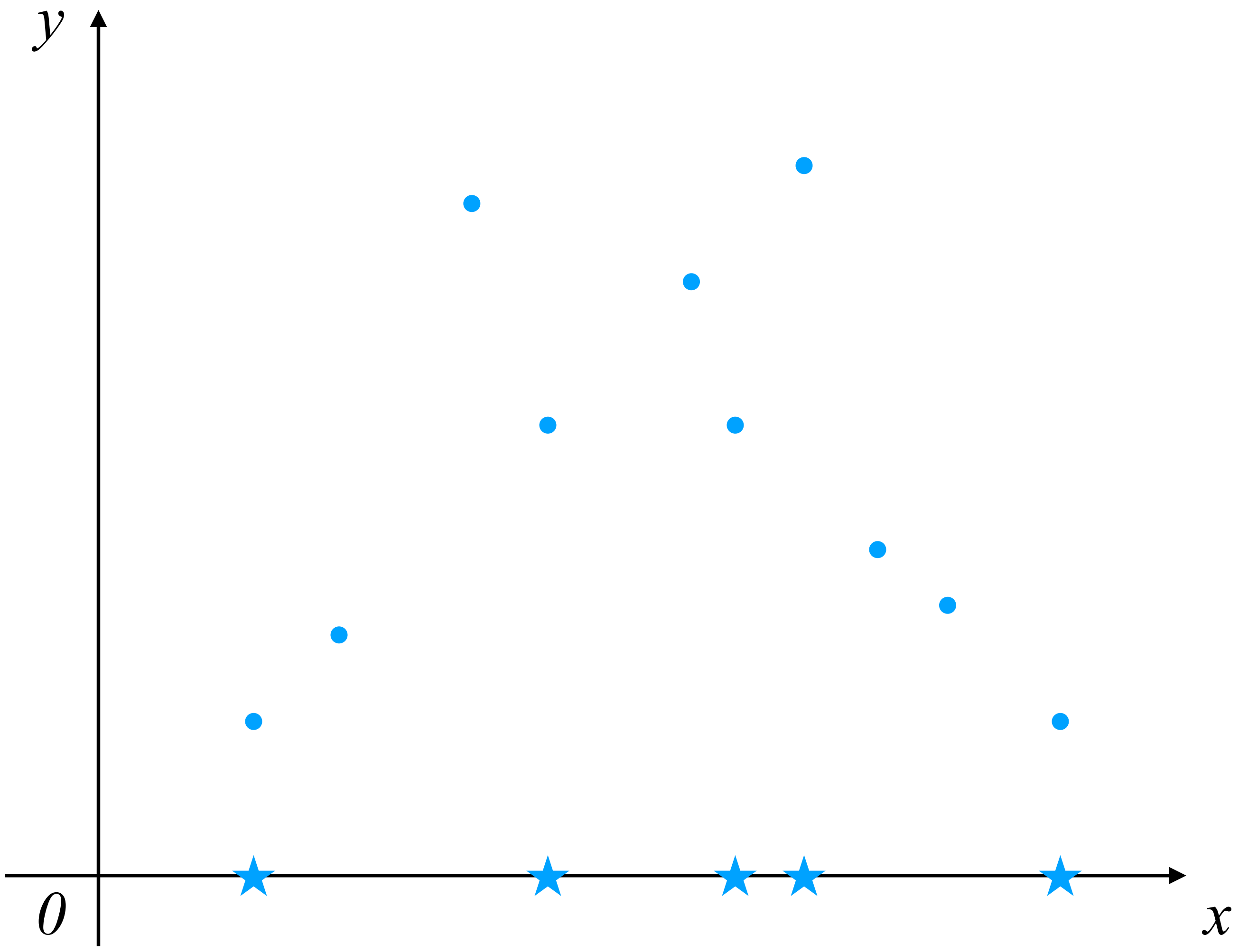
Problem Formulation



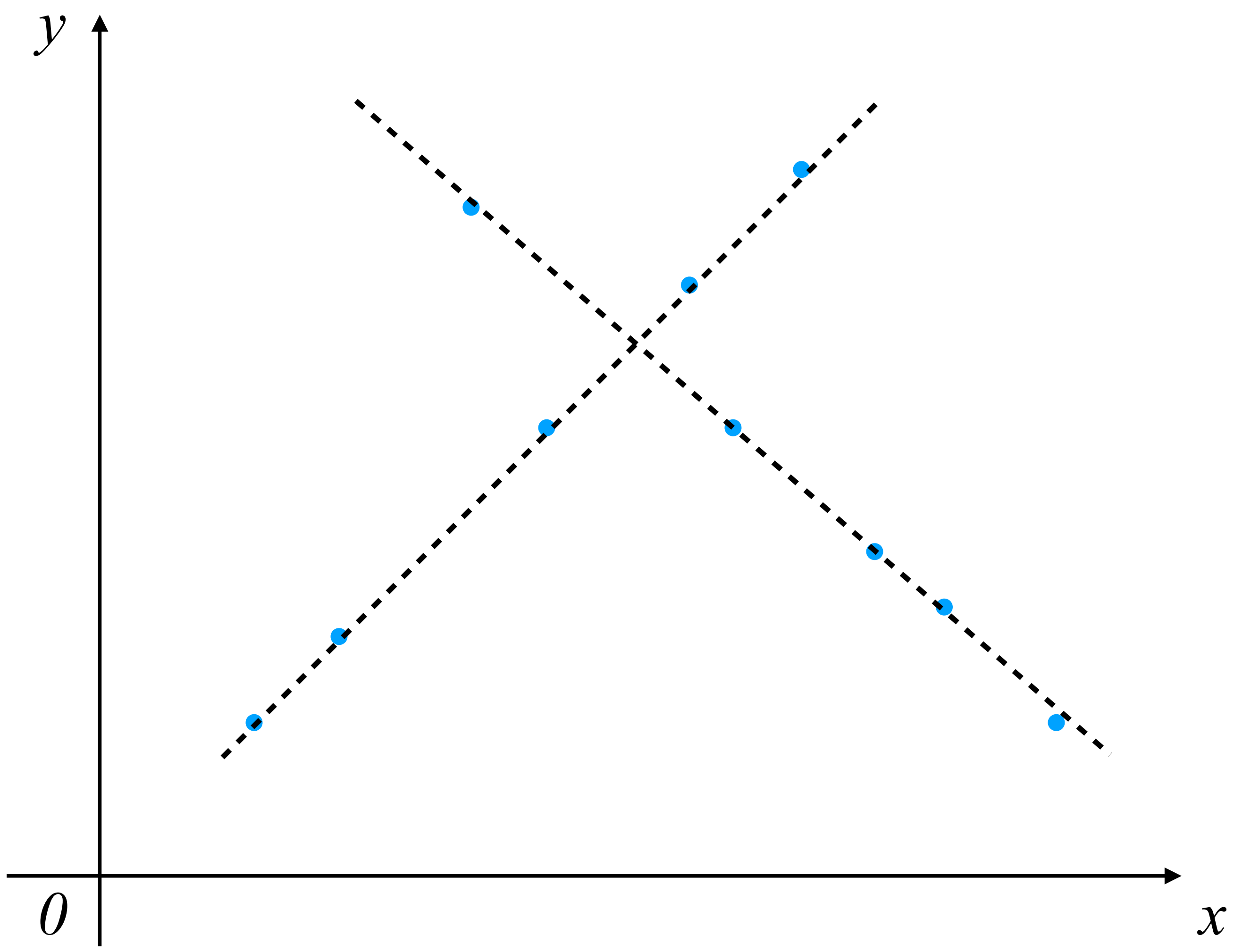
Problem Formulation



Problem Formulation



Problem Formulation



Problem Formulation

Goal:

- Output accurate estimates $\hat{\beta}^{(1)}, \hat{\beta}^{(2)}, \ldots, \hat{\beta}^{(L)}$.
- Minimize sample and time complexities.

Can we get sample and time complexities sublinear in n?

Related Work (incomplete list)

- **Tensor decomposing:** Chaganty, and Liang. "Spectral Experts for Estimating Mixtures of Linear Regressions." *ICML* 2013.
- **Convex relaxation:** Chen, Yi, and Caramanis. "A Convex Formulation for Mixed Regression with Two Components: Minimax Optimal Rates." *COLT*. 2014.
- **Alternating minimization (EM):** Yi, Caramanis, and Sanghavi. "Alternating Minimization for Mixed Linear Regression." *ICML*. 2014.

  Städler, Bühlmann, and van de Geer. "L_1 Penalization for Mixture Regression Models." *TEST*, 2012.

Sparse mixed linear regression: main results

Mixed-Coloring algorithm

For any fixed $p^* \in (0,1)$, for $\boldsymbol{m = \theta(K)}$, the Mixed-Coloring algorithm satisfies these properties for each $\ell \in [L]$:

- No false discovery
- Recover $1 - p^*$ fraction of the support of each $\beta^{(\ell)}$ w.p. $1 - O(1/K)$.
- Recovered support is uniform
- Time complexity: $\Theta(K)$ *(optimal)*

Yin, Pedarsani, Chen, R., 2017

Main Results

- Precise characterization of the constants in the sample complexity.

L	2	3	4
p^*	5.1×10^{-6}	8.8×10^{-6}	8.1×10^{-6}
$m = CK$	$33.39K$	$37.80K$	$40.32K$

L: # of parameter vectors
K: sparsity
p^* : error floor
m: # of measurements

- Time complexity: $\Theta(K)$ (optimal)

- $C = \Theta(\log\frac{1}{p^*})$.

Primitives

Summation check:

- Goal: find measurements generated by the same $\beta^{(\ell)}$
- Generate $x_1, x_2 \in \mathbb{C}^n$ from some continuous distribution
- Generate the third vector of the form $x_3 = x_1 + x_2$
- Get measurements y_1, y_2, y_3
- $y_3 = y_1 + y_2$?
- If so, the three measurements come from the same $\beta^{(\ell)}$
- Consistent pair (y_1, y_2)

Ratio test

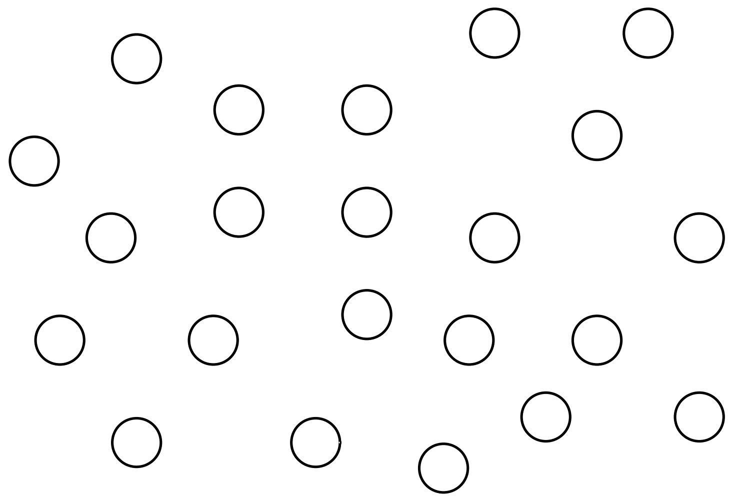
- Find location of a singleton

Peeling

- Remove contribution to other measurements

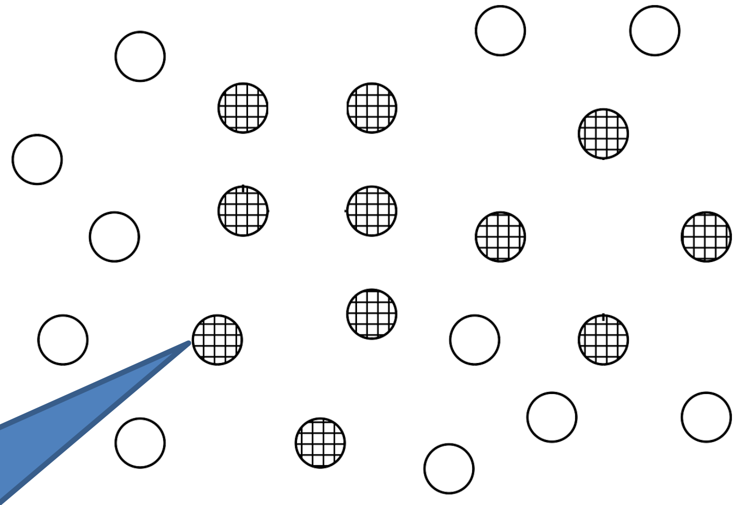
Decoding Algorithm

- Find consistent pairs



Decoding Algorithm

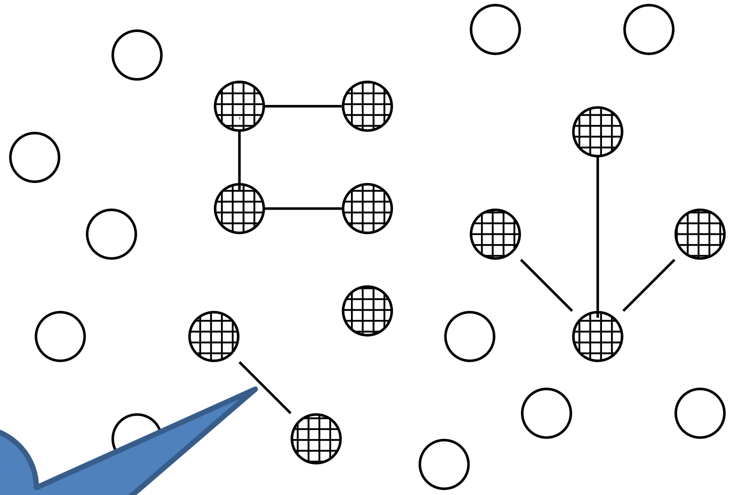
- Find consistent pairs
- Find singletons



Singleton balls
At this stage, we have got some non-zero elements but we don't know which parameter vectors they belong to.

Decoding Algorithm

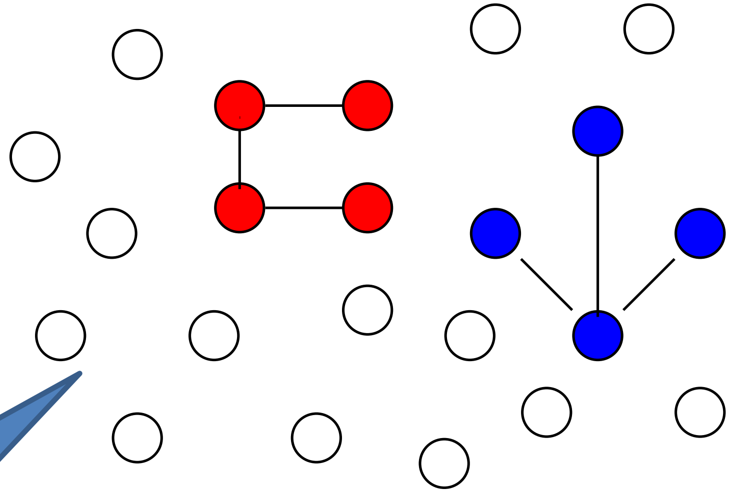
- Find consistent pairs
- Find singletons
- Find strong doubletons



- **Strong doubletons:** consistent pairs that are only associated with two singleton balls found in the first stage.
- Can be found by guess-and-check.
- The two singleton balls must be in the same parameter vector.

Decoding Algorithm

- Find consistent pairs
- Find singletons
- Find strong doubletons

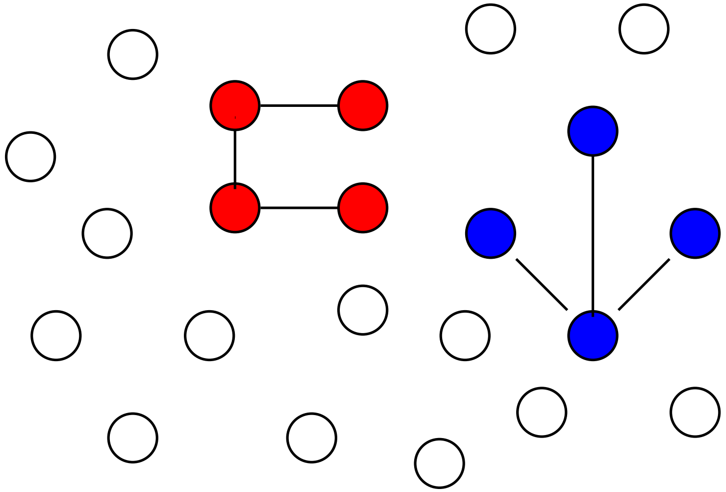


Theorem: As long as M/K > const., the L largest connected components of the graph are of size O(K), and correspond to different parameter vectors. Other connected components are of size O(log K).

[Follows from E-R (n,p) random graphs: if np>1, then component size is O(n), else it is O(log n).]

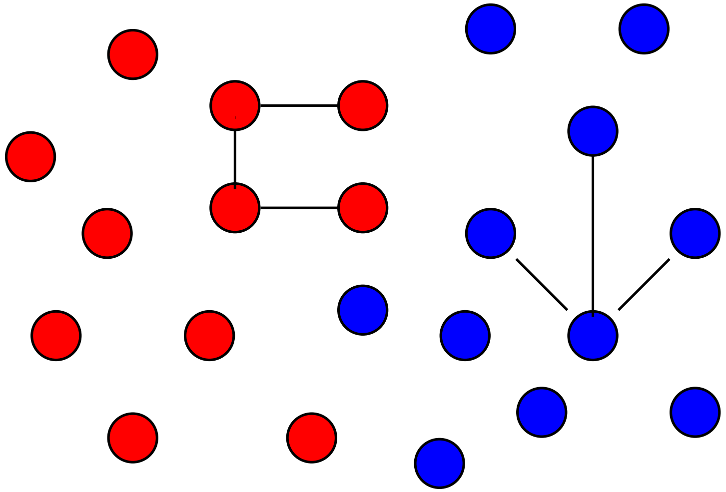
Decoding Algorithm

- Find consistent pairs
- Find singletons
- Find strong doubletons
- Recover a subset of size $\Theta(K)$



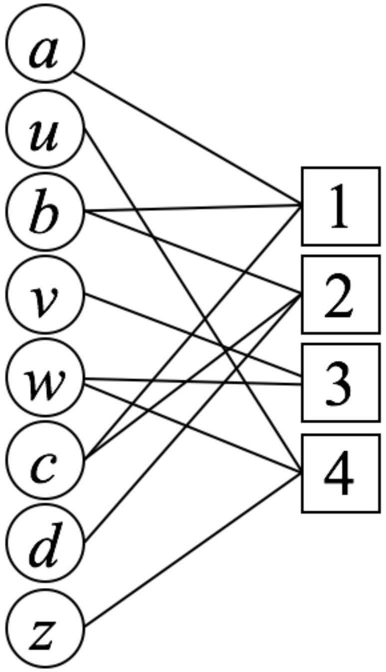
Decoding Algorithm

- Find consistent pairs
- Find singletons
- Find strong doubletons
- Recover a subset of size $\Theta(K)$
- Iterative decoding



Decoding Algorithm

Iterative decoding:

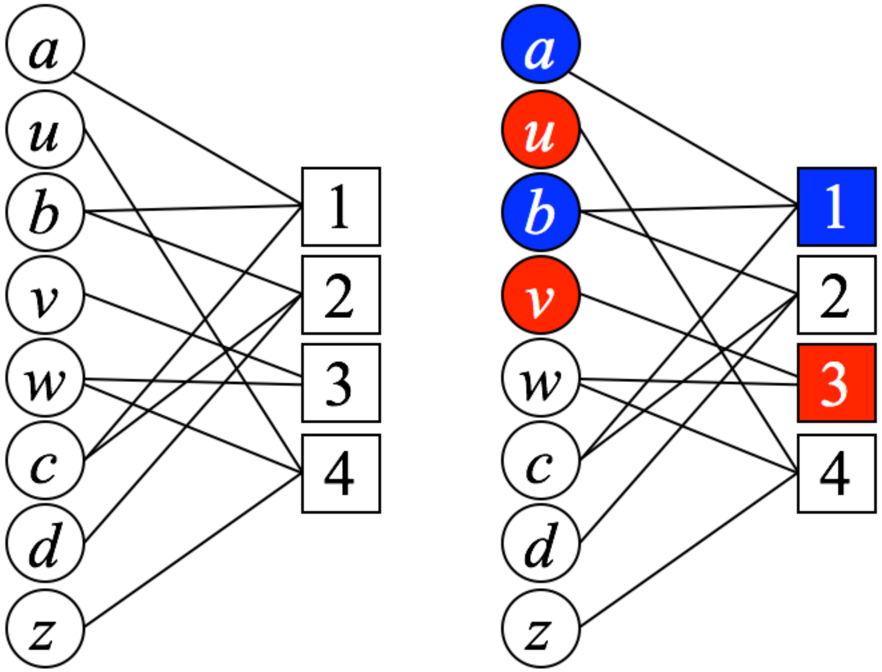


Consistent pairs (bins). Each bin is either blue or red.

Non-zero elements from two parameter vectors, either blue or red

Decoding Algorithm

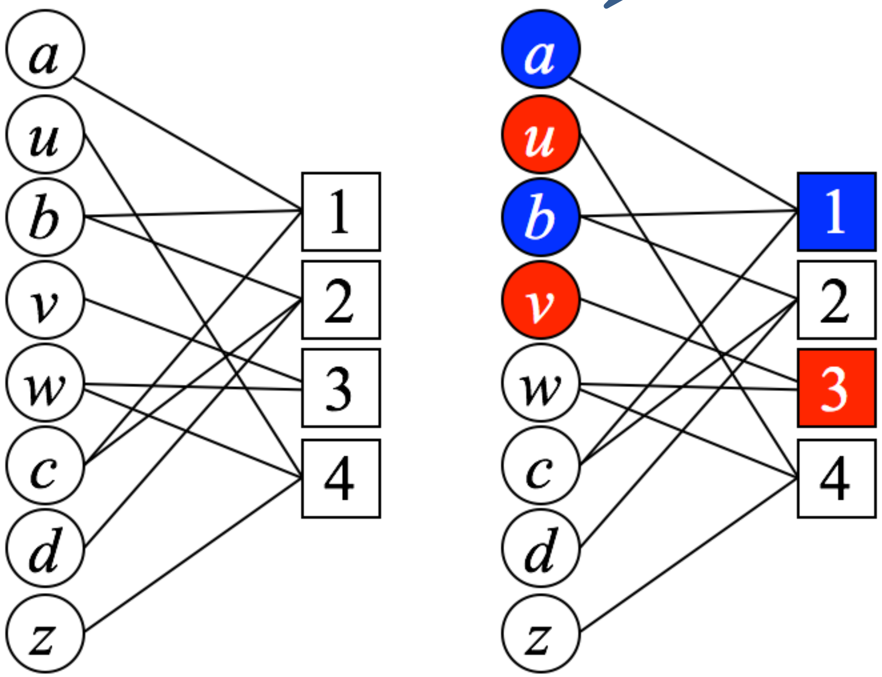
Iterative decoding:



Decoding Algorithm

Iterative decoding:

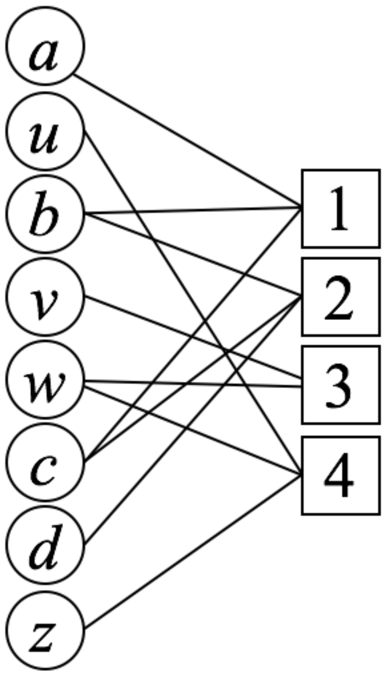
By finding strong doubletons and largest connected components, we have already **recovered** a fraction of non-zero elements. Say *a*, *b* (blue) and *u*, *v* (red).

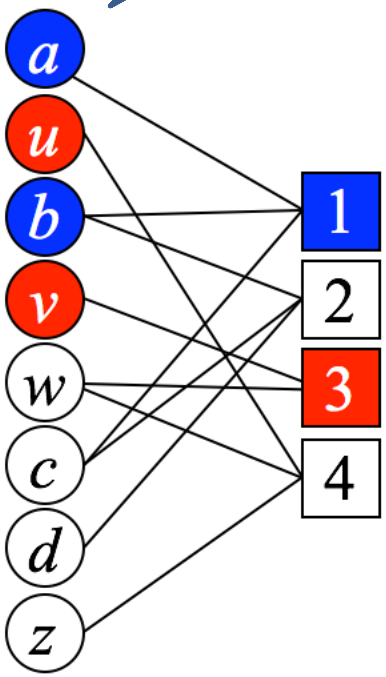


Decoding Algorithm

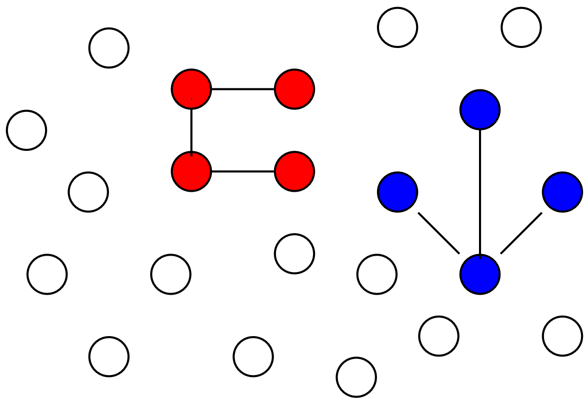
Iterative decoding:

By finding strong doubletons and largest connected components, we have already **recovered** a fraction of non-zero elements. Say *a*, *b* (blue) and *u*, *v* (red).



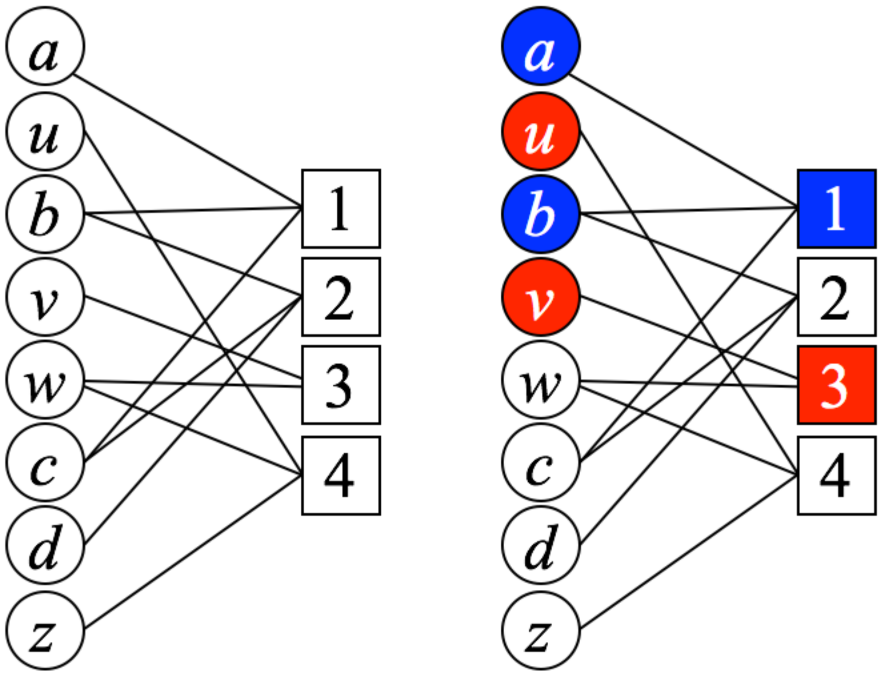


Recall:



Decoding Algorithm

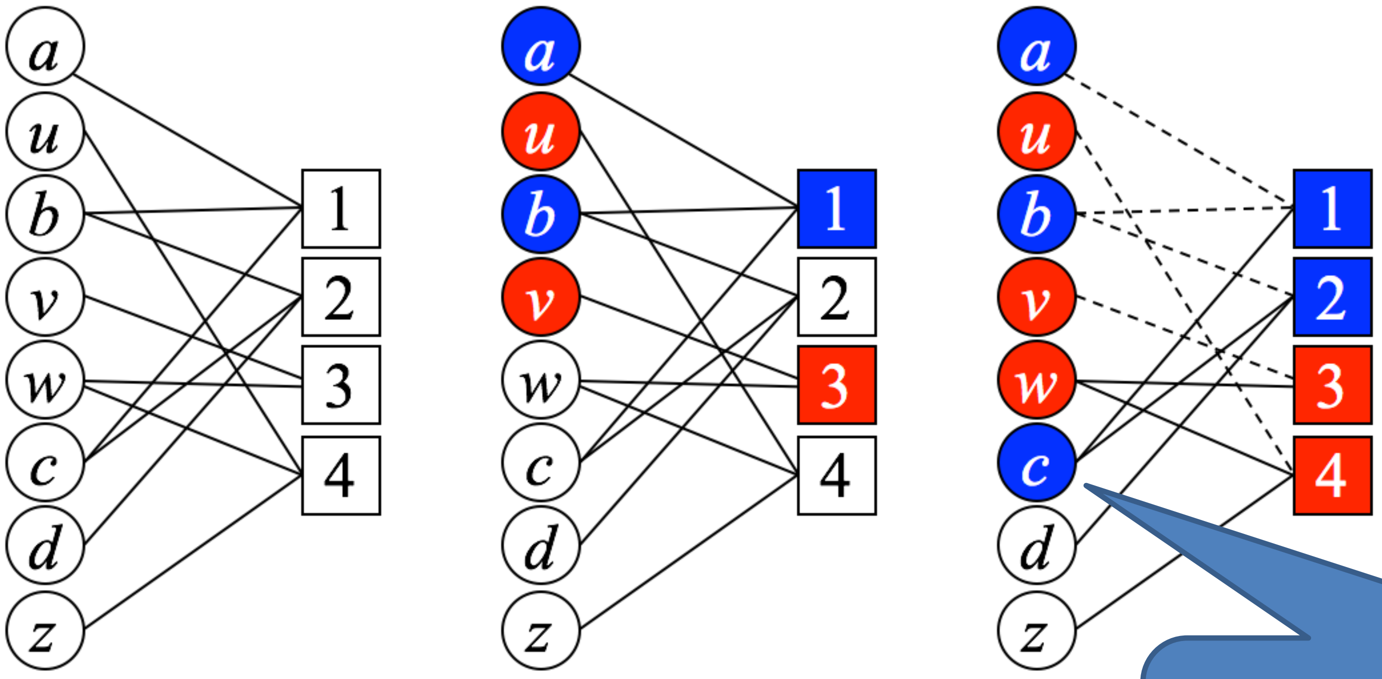
Iterative decoding:



Guess-and-check: try to subtract *a* and *b* from bin 1, and *v* from bin 3.

Decoding Algorithm

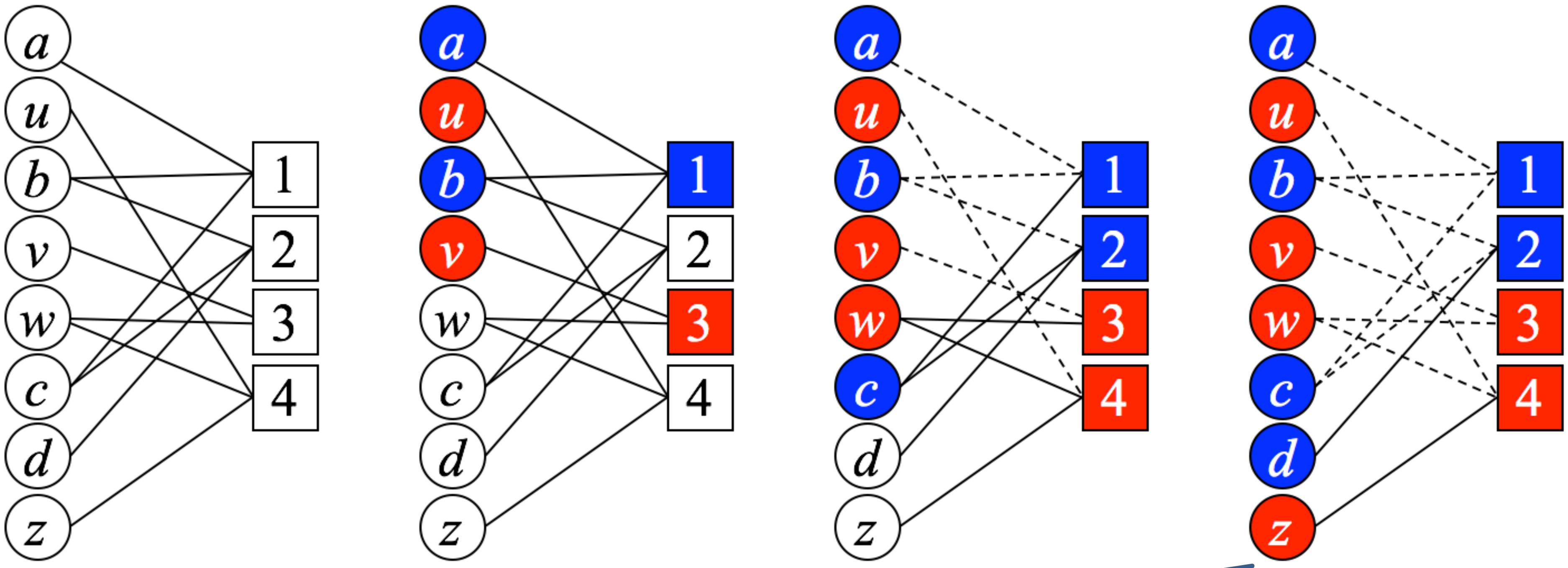
Iterative decoding:



The remaining measurements pass ratio test. Recover *c* using bin 1 and recover *w* using bin 3.

Decoding Algorithm

Iterative decoding:



Iterate this procedure and recover all the non-zero elements.

Decoding Algorithm

Density evolution:

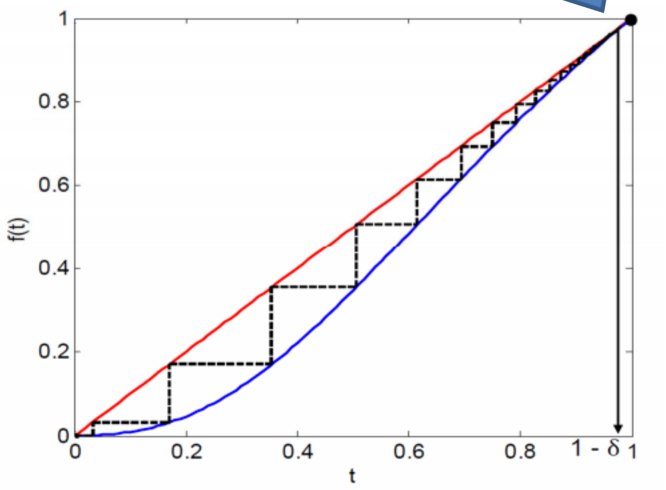
- Consider one particular parameter vect
- p_j: the fraction of non-zero elements th
  not recovered after the j-th iteration.

- Bad news: $p_0 = 1$ is a fixed point!
- Good news: If we can start at $p_0 = 1 - \delta$, we are good to go!

- Summation starts from 2 because singletons are not useful for iterative decoding as we don't know their "color."

ee i.

$$p_{j+1} = f(p_j) = \left(1 - \sum_{i=2}^{\infty} \rho_i (1 - p_j)^{i-1}\right)^{d-1}$$



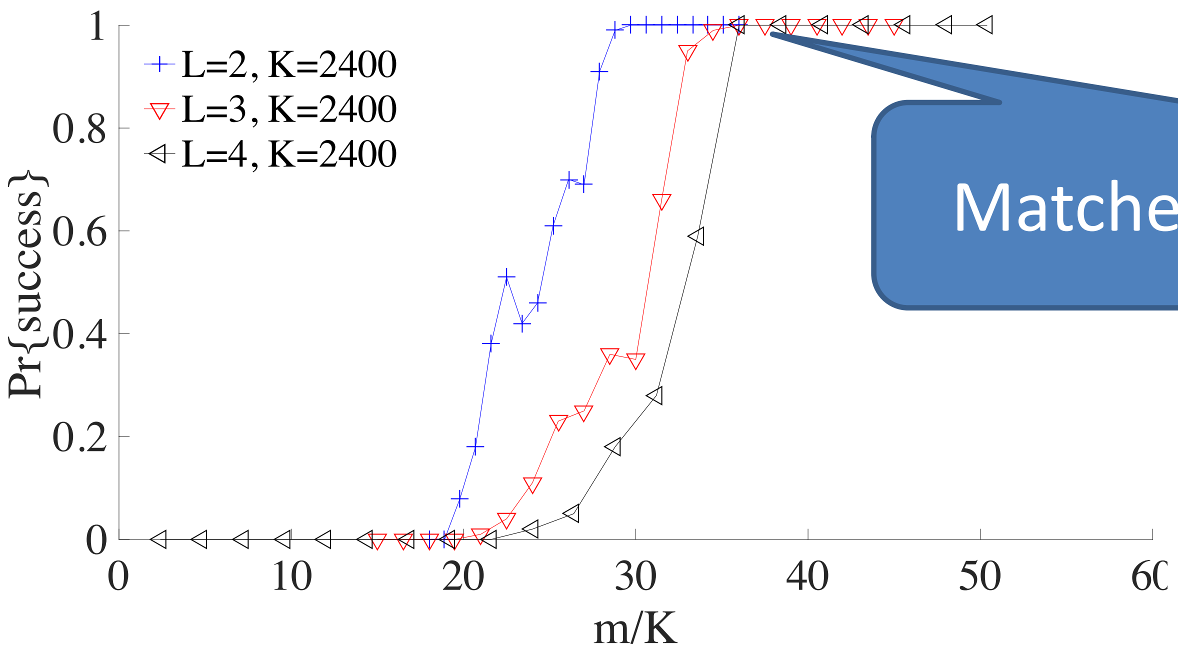
- p_j can converge to an arbitrarily small constant.

Experimental Results

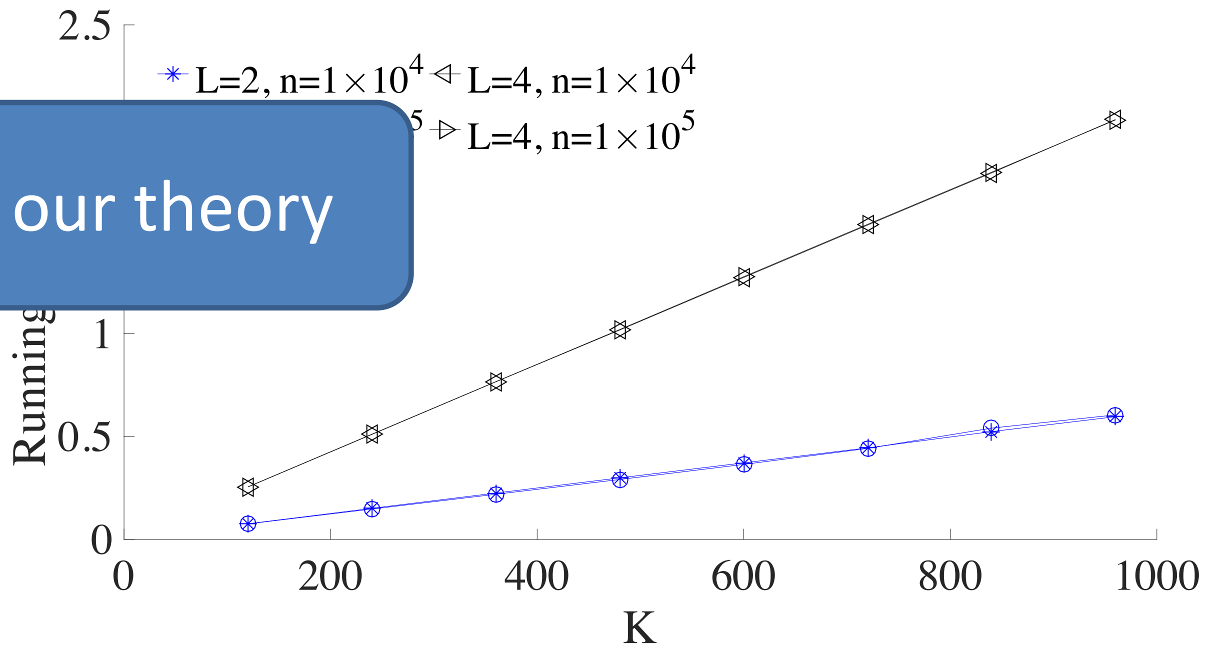
L	2	3	4
p^*	5.1×10^{-6}	8.8×10^{-6}	8.1×10^{-6}
$m = CK$	$33.39K$	$37.80K$	$40.32K$

Noiseless setting: sample and time complexities:

- Optimal parameters (d, R, V) computed from density evolution.
- Success: exact recovery of all non-zero elements.
- Empirical success probability/average running time over 100 trials.



$n = 10^5$



Generic method to make algorithm robust to noise

Recall how we find locations and values of singletons in the noiseless setting.

Ex.: a singleton with non-zero element b at index 4

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 \\ r_1 & r_2 W & r_3 W^2 & r_4 W^3 & r_5 W^4 & r_6 W^5 & r_7 W^6 & r_8 W^7 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ b \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} r_4 \\ r_4 W^3 b \end{bmatrix}$$

Location information is encoded in the ***relative phase*** between y_2 and y_1.

- What if we have $y_1 = r_4 + w_1$ and $y_2 = r_4 W^3 b + w_2$?
- $\angle\left(\frac{y_2}{y_1}\right) =?$

Robust Mixed-Coloring Algorithm

It is not robust to encode the location information in the relative phase!

Alternative choice?

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- It is still possible to recover the binary pattern of the measurements by a simple thresholding.
- Of course we may make mistakes.
- This procedure can be robustified by simply **repeating** each bit or using an **error correcting code**.

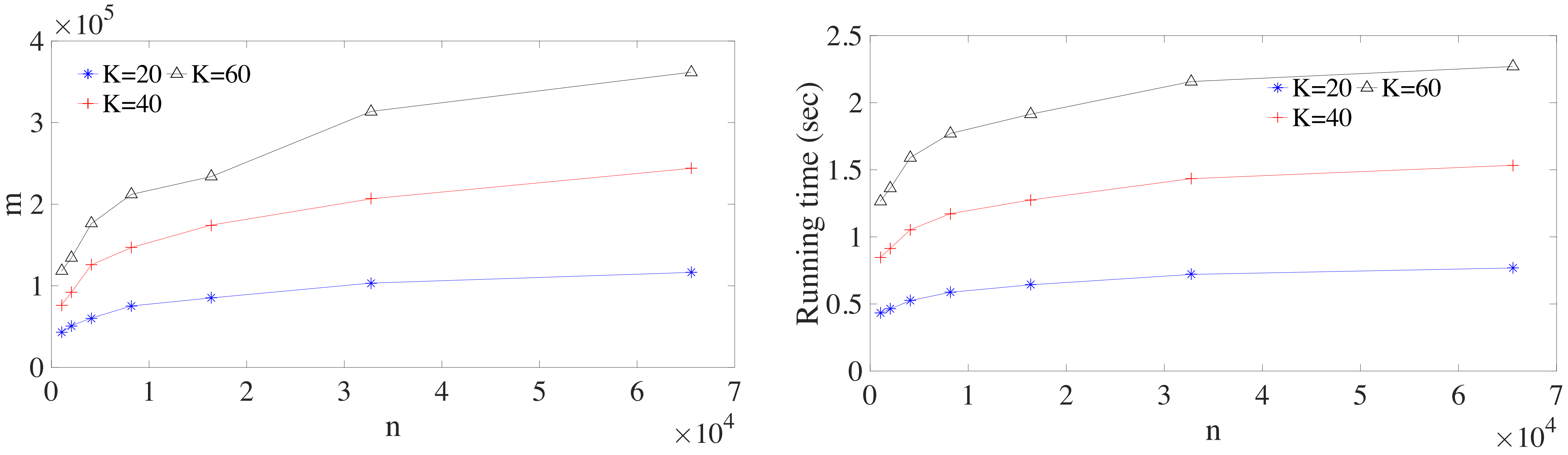
- We can also encode the lo

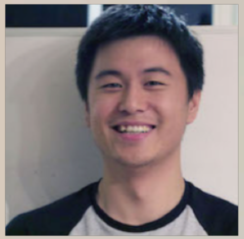
- What if $\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 + w_1 \\ b + w_2 \\ b + w_3 \end{bmatrix}$?

Experimental Results

Noisy setting: sample and time complexities:

- $\Delta = 1, \sigma = 0.2.$
- Record the minimum number of measurements to achieve 100 consecutive success.
- Sublinear sample and time complexities.

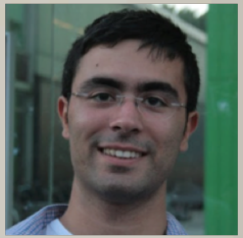




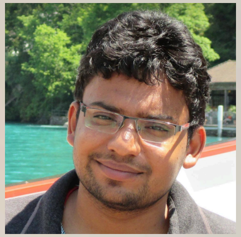
Kangwook Lee

Ramtin P.

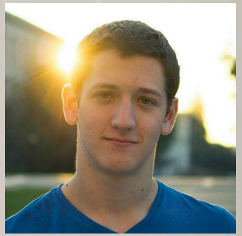
Dimitris P.



Orhan Ocal



Vipul Gupta

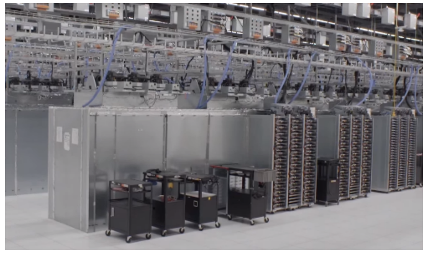


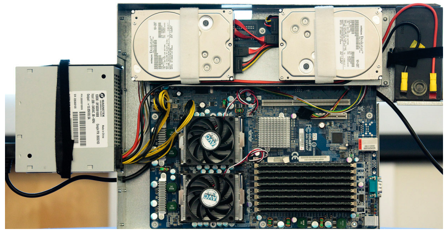
Tavor Baharav

Chapter 5

Speeding up distributed computing on the cloud

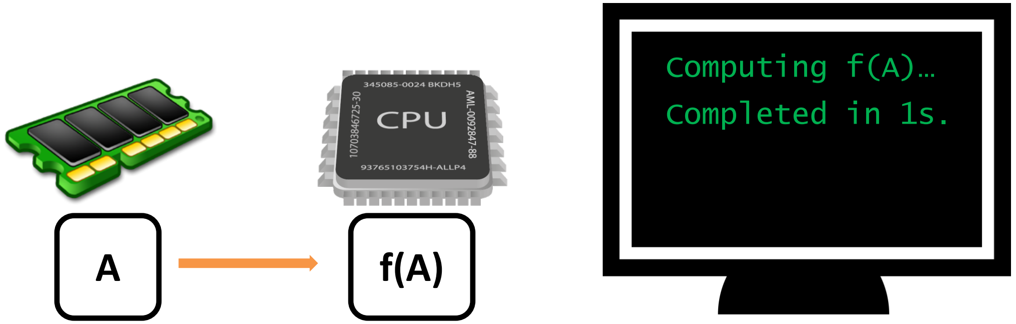
System Noise


Network bottlenecks

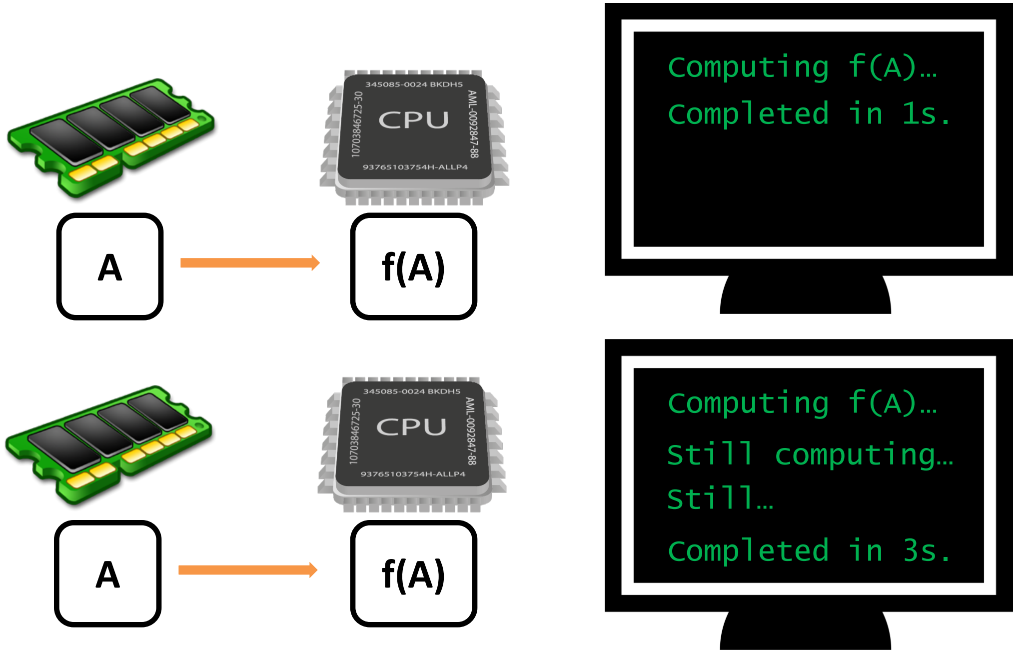

HW failures


Maintenance, etc.

System Noise = Latency Variability

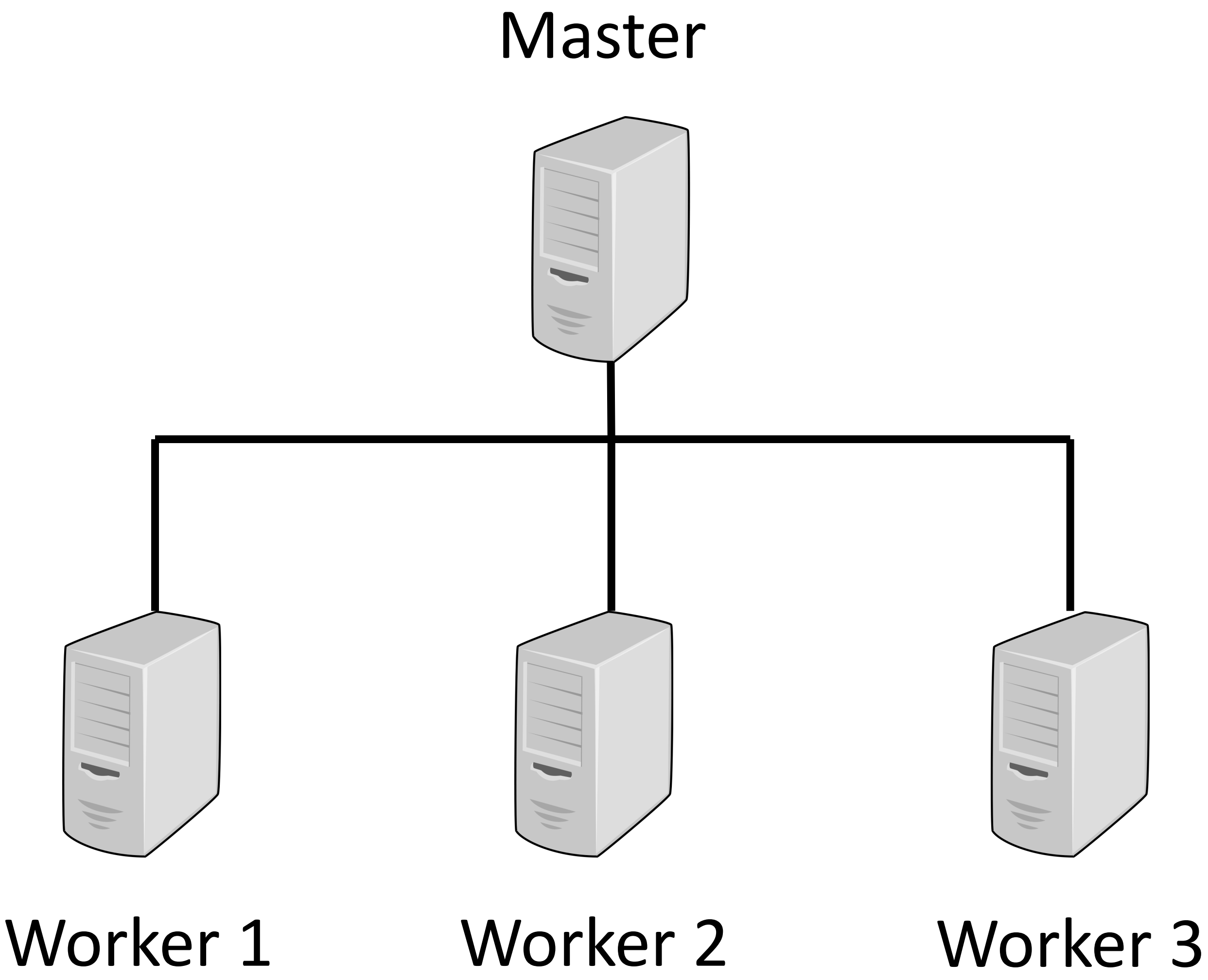


System Noise = Latency Variability



Distributed Matrix-Vector Multiplication

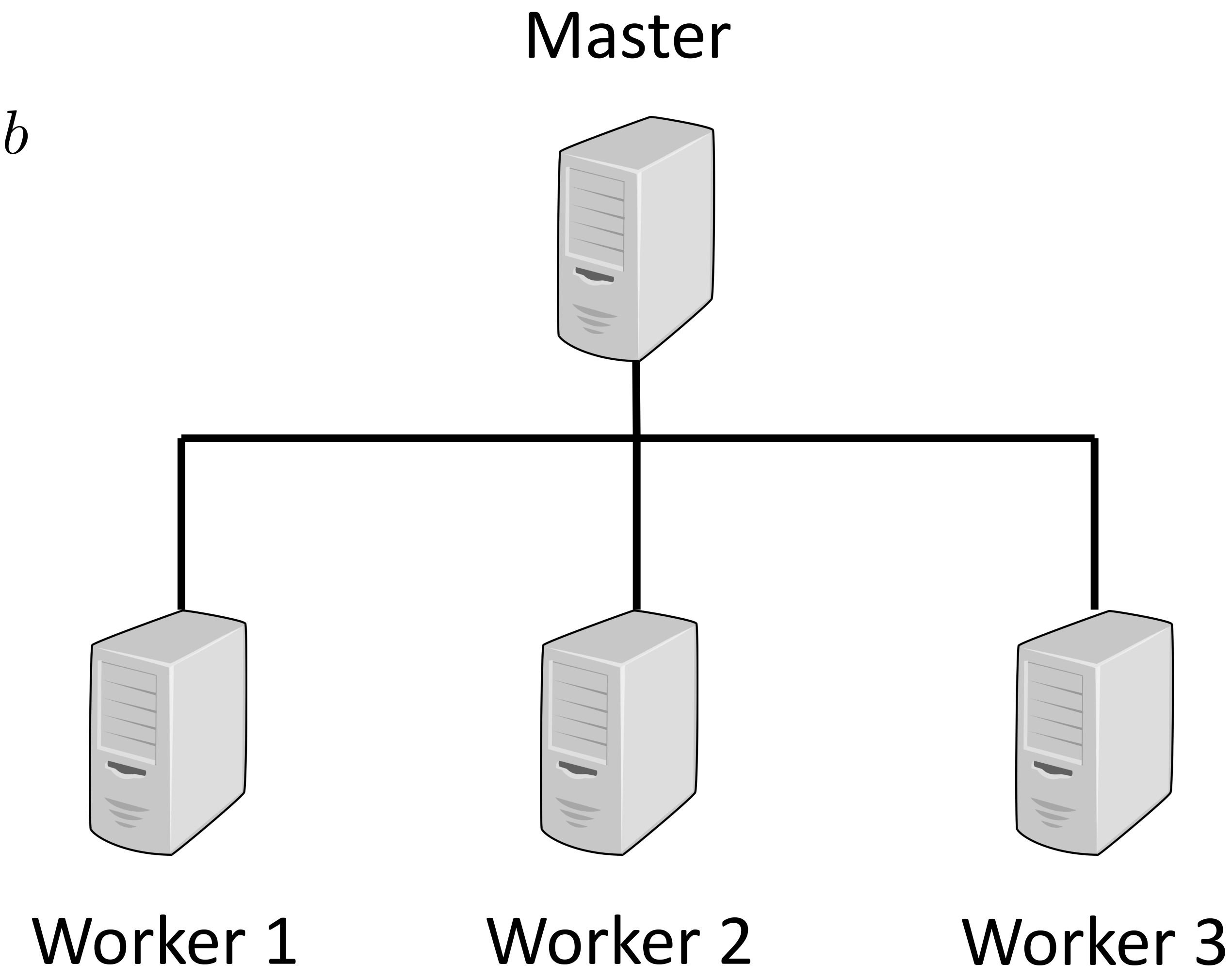
$A \times b$



Distributed Matrix-Vector Multiplication

$$A \times b$$

$$= \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b$$

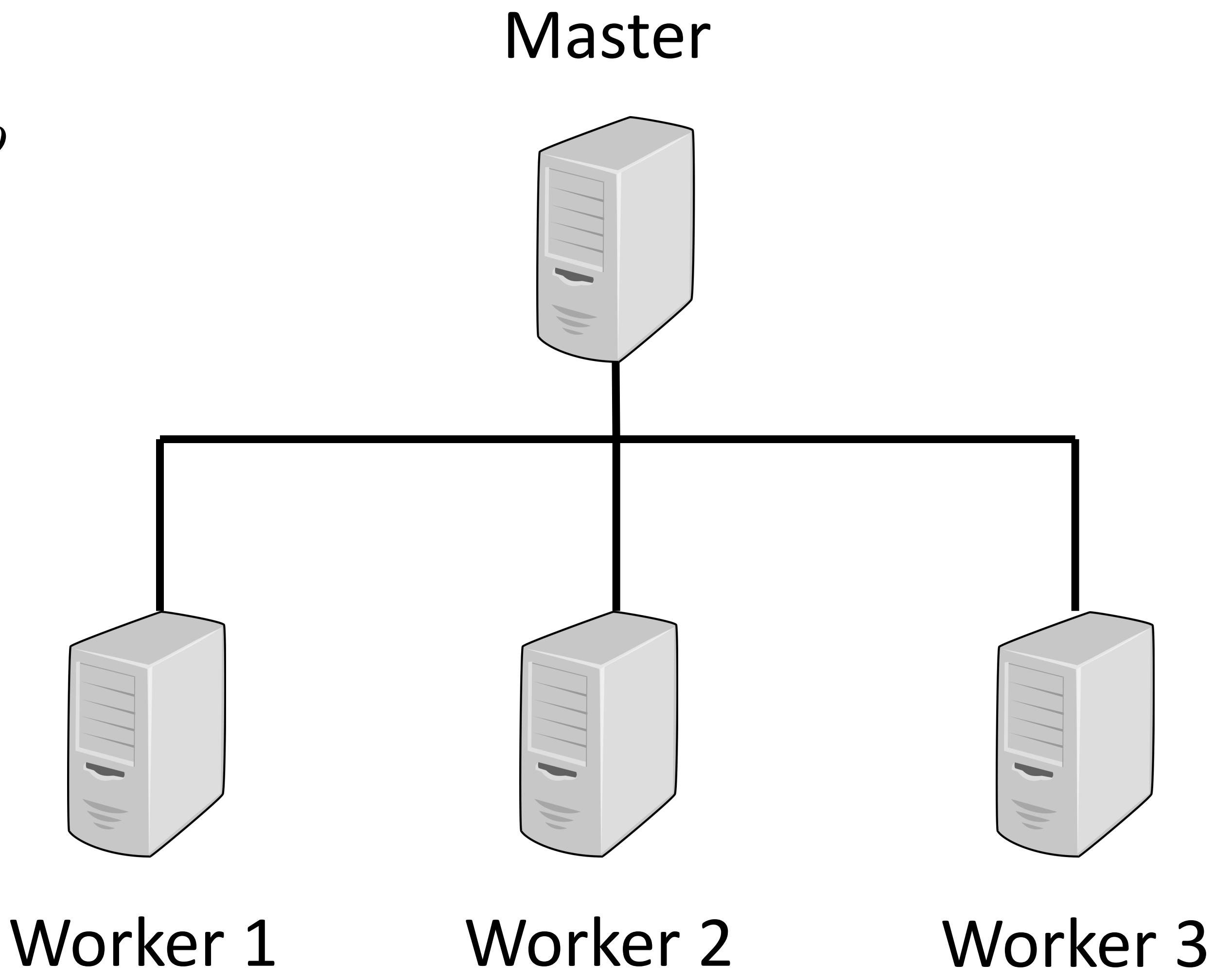


Distributed Matrix-Vector Multiplication

$$A \times b$$

$$= \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b$$

$$= \begin{pmatrix} A_1 \times b \\ A_2 \times b \\ A_3 \times b \end{pmatrix}$$



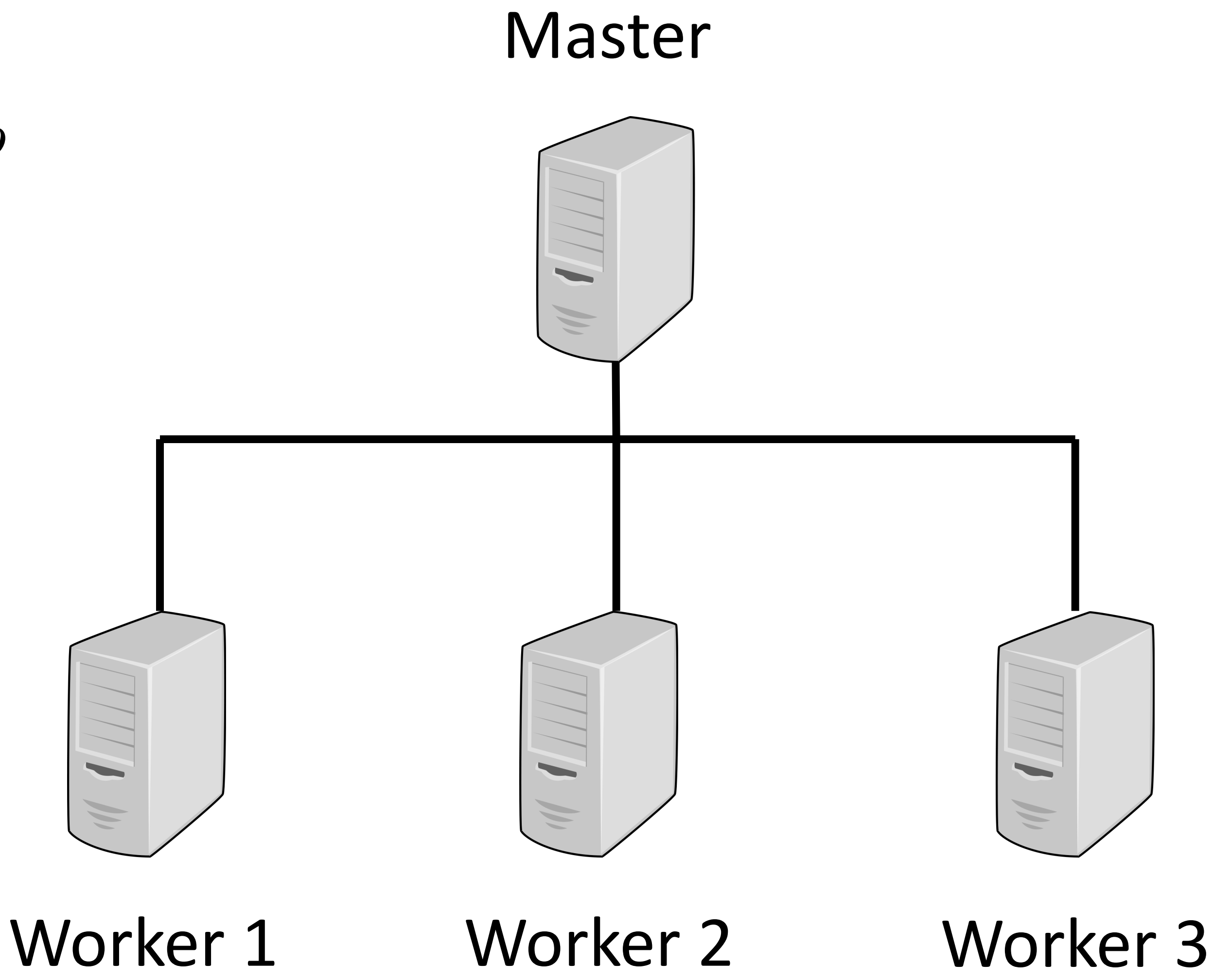
Distributed Matrix-Vector Multiplication

$$A \times b$$

$$= \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b$$

$$= \begin{pmatrix} A_1 \times b \\ A_2 \times b \\ A_3 \times b \end{pmatrix}$$

$$:= \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$



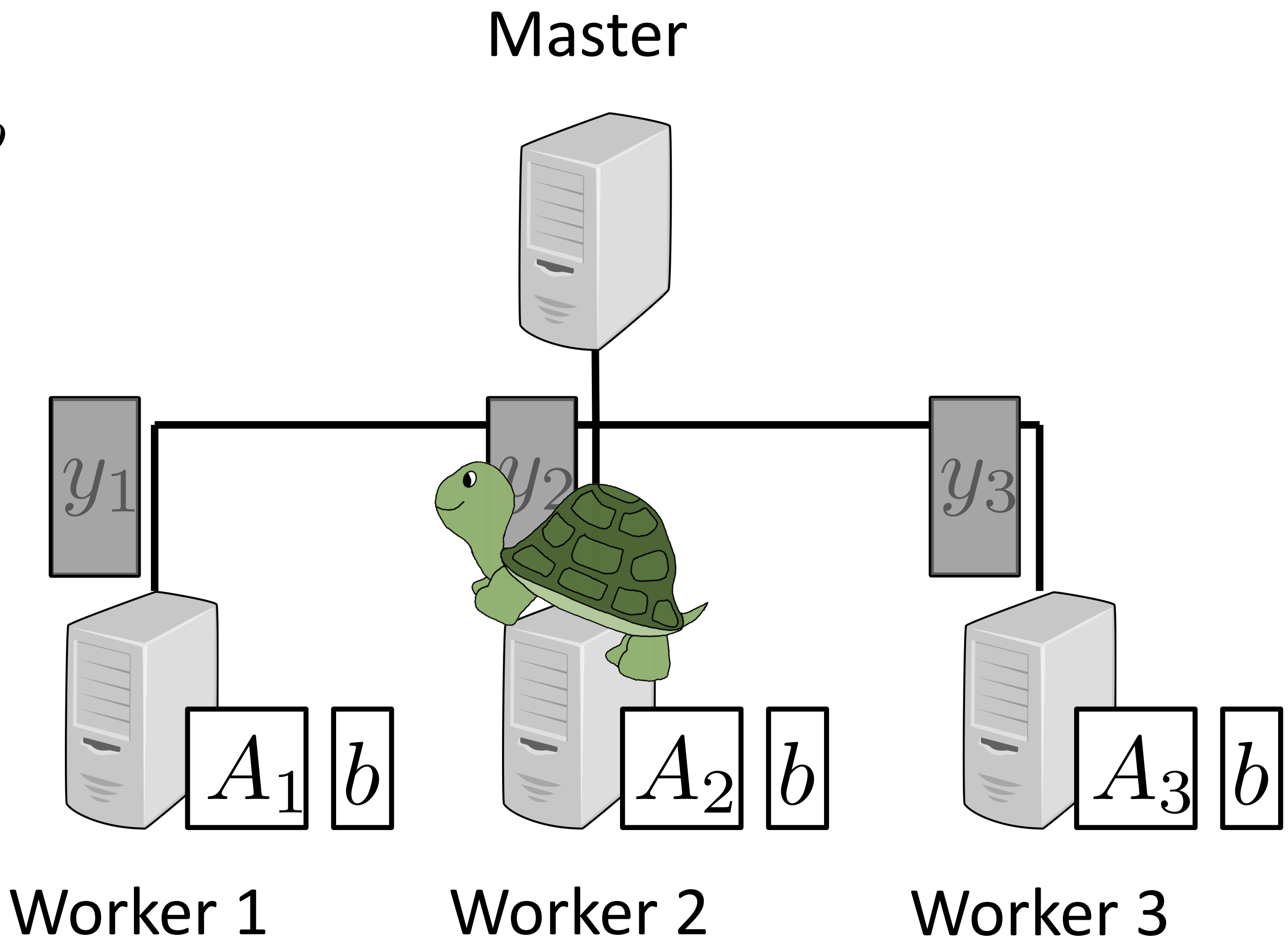
Distributed Matrix-Vector Multiplication

$$A \times b$$

$$= \begin{pmatrix} A_1 \\ A_2 \\ A_3 \end{pmatrix} \times b$$

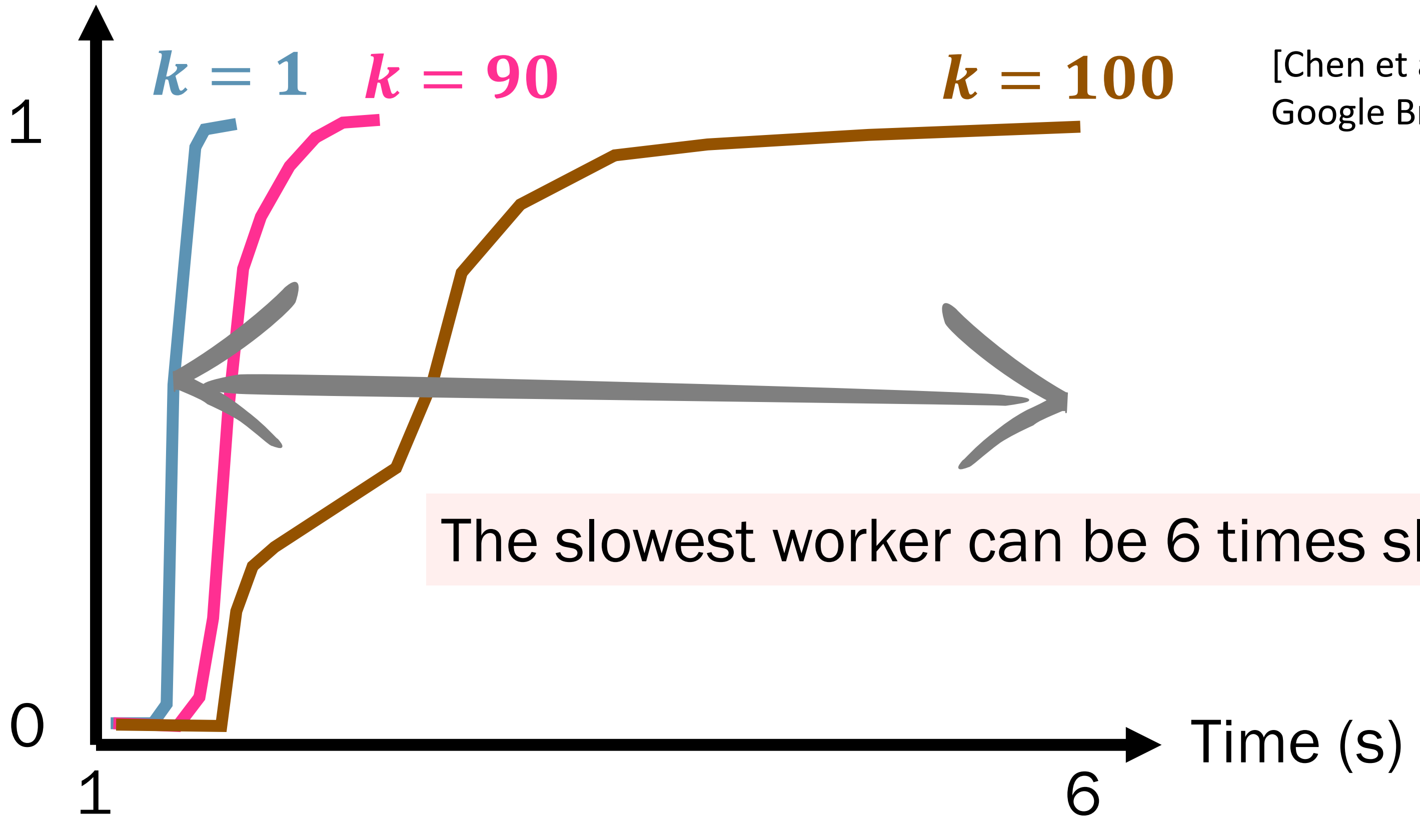
$$= \begin{pmatrix} A_1 \times b \\ A_2 \times b \\ A_3 \times b \end{pmatrix}$$

$$:= \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$



Straggler Problem

CDF of time to collect results from k workers

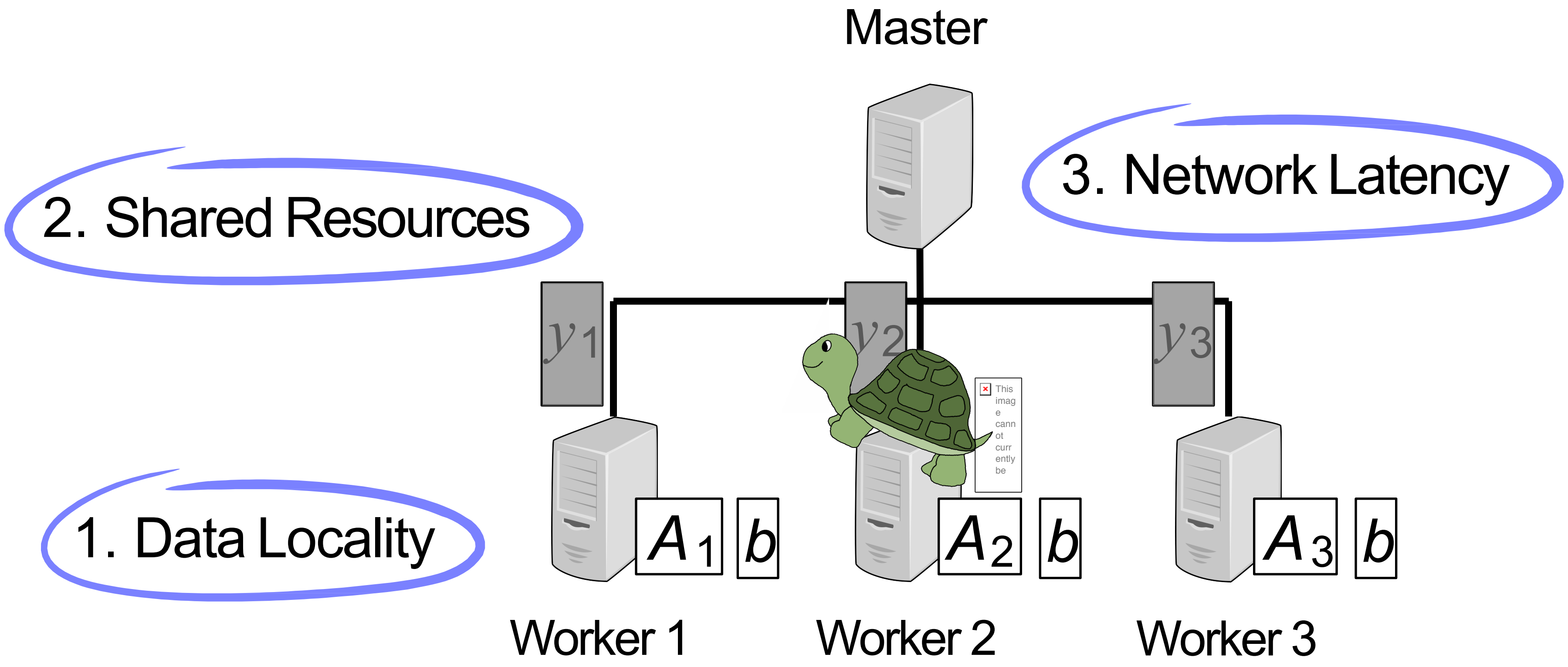


[Chen et al., 2016]
Google Brain

The slowest worker can be 6 times slower

Why Do We Have Stragglers?

> *"… infeasible to eliminate all latency variability."*
>
> [Dean, Barroso, Comm. ACM'2013]



Coded Matrix-Vector Multiplication

$$A \times b$$

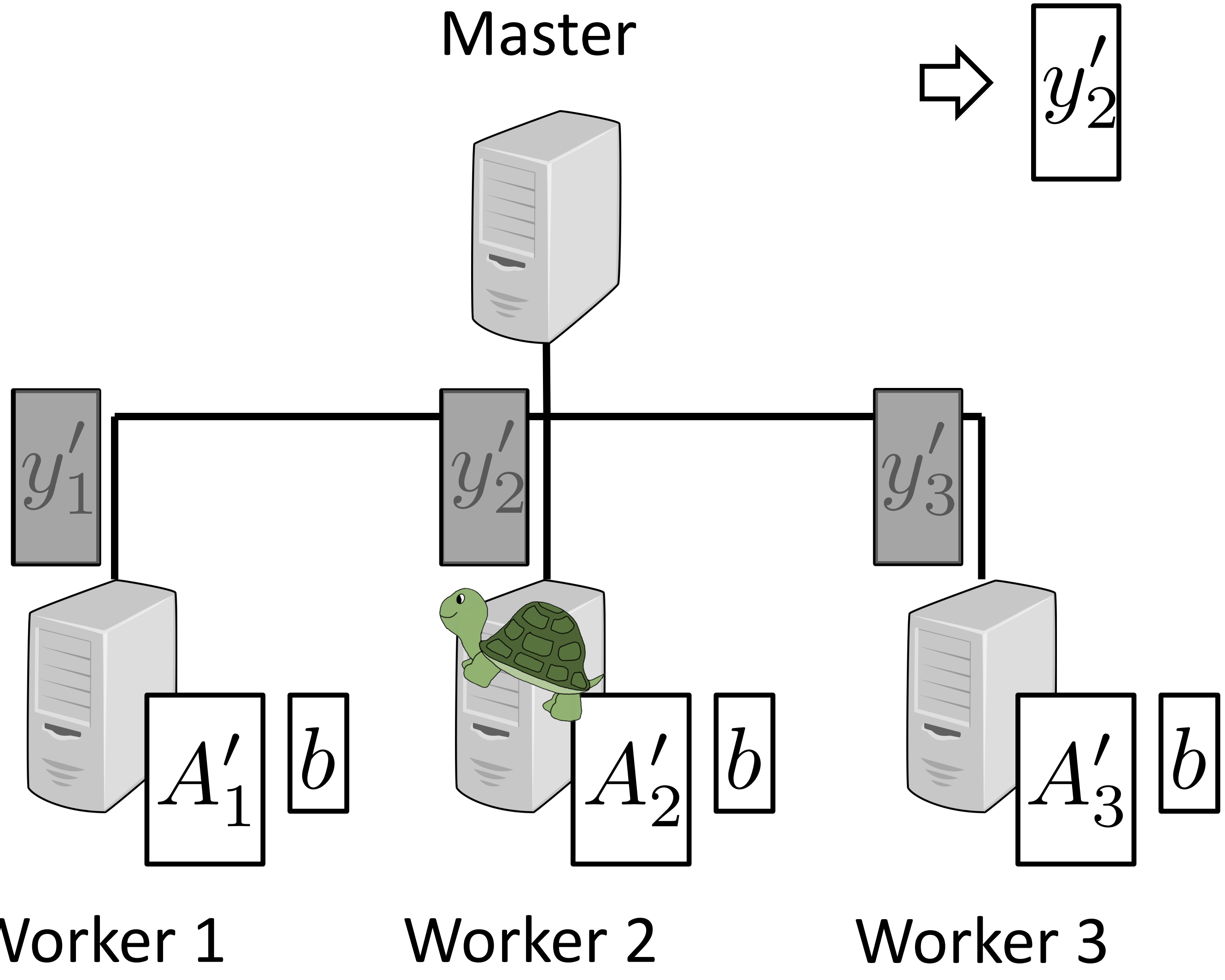
$$= \begin{pmatrix} A'_1 \\ A'_2 \end{pmatrix} \times b$$

$$= \begin{pmatrix} A'_1 \times b \\ A'_2 \times b \end{pmatrix}$$

$$:= \begin{pmatrix} y'_1 \\ y'_2 \end{pmatrix}$$

$$A'_3 := A'_1 + A'_2$$

$$y'_3 := y'_1 + y'_2$$



Coded Computation for Linear Operations

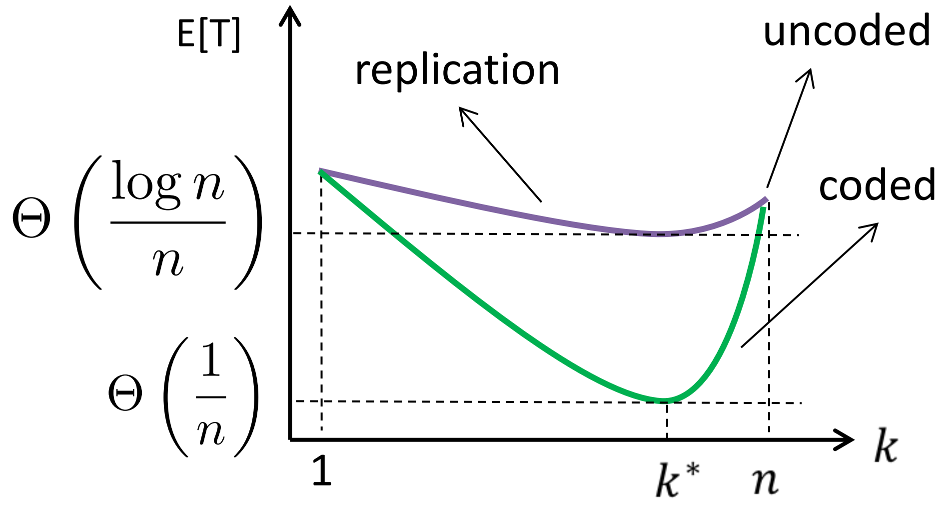
Assumptions:

- n workers
- k subtasks
- **Computing time of each worker: constant + exponential RV (i.i.d.)**
- Average computing time is proportional to $1/k$

Theorem:

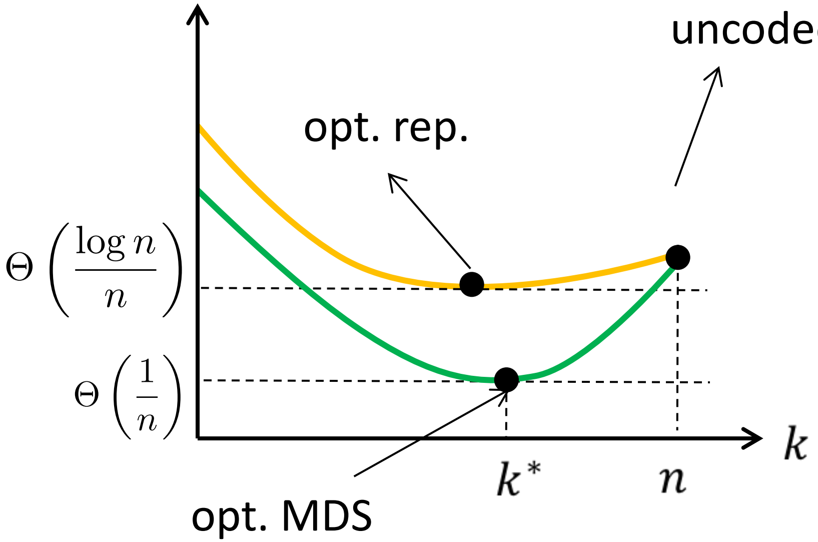
$$\mathrm{E}[T_{\text{uncoded}}] = \Theta\left(\frac{\log n}{n}\right) \qquad \mathrm{E}[T^{\star}_{\text{replication}}] = \Theta\left(\frac{\log n}{n}\right)$$

$$\mathrm{E}[T^{\star}_{\text{MDS-coded}}] = \Theta\left(\frac{1}{n}\right)$$

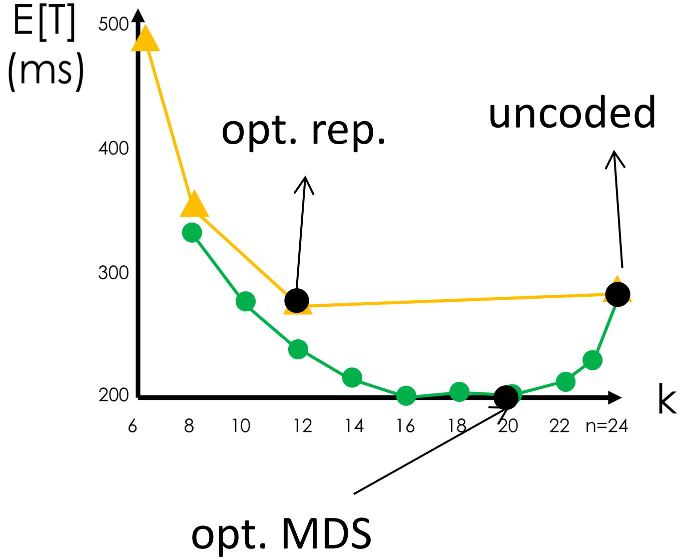


Lee, Lam, Pedarsani, Papailiopoulos, R. 2015

MDS-Coded Matrix-Vector Multiplication



Under exponential latency model



On Amazon AWS

Codes provide 30% speedup compared uncoded and replicated jobs for fixed number of workers

[LLPP**R**, NIPS workshop'15]
[LLPP**R**, T-IT'18]

Applications

- Distributed linear regression
- Distributed non-linear function computation
- Reducing communication in data shuffling by network coding

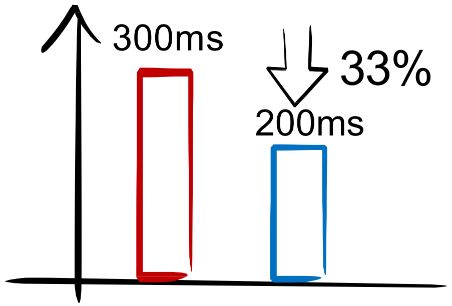
Has attracted lots of interest:

- Coded ***Matrix Multiplication in MapReduce*** setup
- Coded Computation for ***Logistic Regression***
- Coded Computation + ***Distributed Gradient Computing***
- Approximation: ***SVD + Coded Matrix Multiplication, Sketching, Second order methods…***

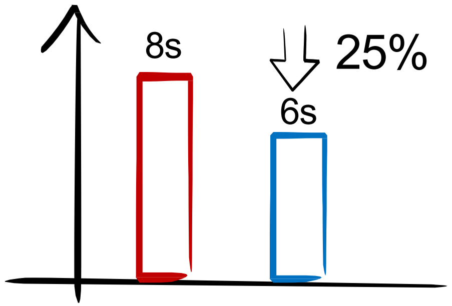
Coded Computation

[LLPPR, NIPS W'15]
[LLPPR, ToIT'18]

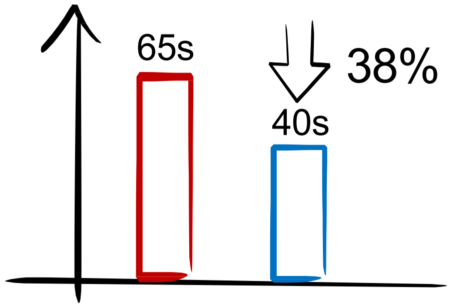
- A new interface between ML systems and information & coding theory
- Codes can be used to speed up distributed computation & distributed ML
  - Matrix-vector multiplication [LLPPR, ToIT'18],
  - Matrix-matrix multiplication [LSR, ISIT'17], [BLOR, ISIT'18], [GWCR, BG'19]
  - Gradient accumulation [LPPR, ISIT'17], [GKCMR, ICML Workshop'19]
  - Data shuffling [CLPPR, NeurIPS W'17], [CLPPR, SysML'18]
- Works in practice (Amazon EC2 experiments on real data)



Matrix multiplication



Linear regression



Matrix completion

Coded Computation

[LLPPR, NIPS W'15]
[LLPPR, ToIT'18]

- Matrix-vector multiplication [LLPPR, ToIT'18]
  - [Ferdinand and Draper, Allerton'16]
  - [Reisizadeh et al., ISIT'17]
  - [Mallick, Chaudhari, Joshi, '18]
  - [Wang, Liu, Shroff, ICML'18]
  - [Maity, Rawat, Mazumdar, SysML'18]
  - …
- Matrix-matrix multiplication [LSR, ISIT'17], [BLOR, ISIT'18] [GWCR, BG'19]
  - [Yu, Maddah-Ali, Avestimehr, NIPS'17]
  - [Dutta et al., '18]
  - …
- Gradient accumulation [LPPR, ISIT'17] [GKCMR, ICML Workshop'19]
  - [Dutta, Cadambe, Grover, NIPS'16]
  - [Tandon, Lei, Dimakis, Karampatziakis, ICML'17]
  - [Raviv, Tamo, Tandon, Dimakis, '17]
  - [Halbawi, Azizan, Salehi, Hassibi, ISIT'18]
  - [Ye and Abbe, ICML'18]
  - [Charles and Papailiopoulos, ISIT'18]
  - …
- Data shuffling [CLPPR, NeurIPS W'17], [CLPPR, SysML'18]
  - [Song et al., ISIT'17]
  - [Attia and Tandon, Globecom'16]
  - …

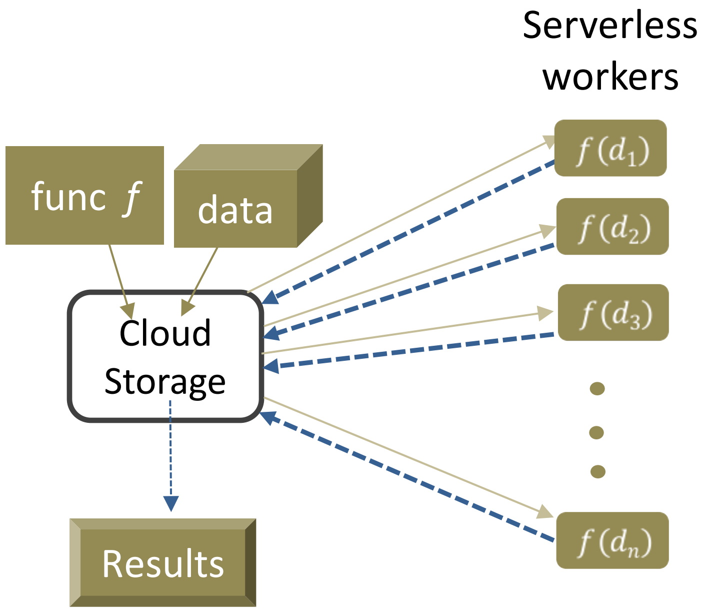
Scalable computing: Serverless platform!

- A decade ago, **cloud servers** abstracted away **physical servers**.
- Future: **"serverless"** computing will abstract away **cloud servers.**

- "Function as a Service (FaaS)"
  - Run my function "somewhere"
  - AWS, Google, IBM, Microsoft, etc.

Why Serverless computing?

- Simple abstraction for user
  - Cluster management hidden
- Tremendous scale
  - 16,000 machines in 10 seconds
  - Cloud storage as infinite RAM
- Reduced Costs
  - Pay only for the time you use
- Significant interest from the cloud computing community

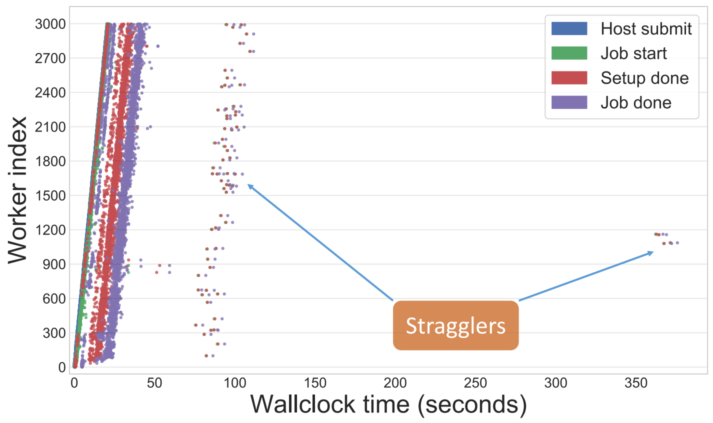


Jonas, Eric, et al. "Cloud Programming Simplified: A Berkeley View on Serverless Computing." (2019).

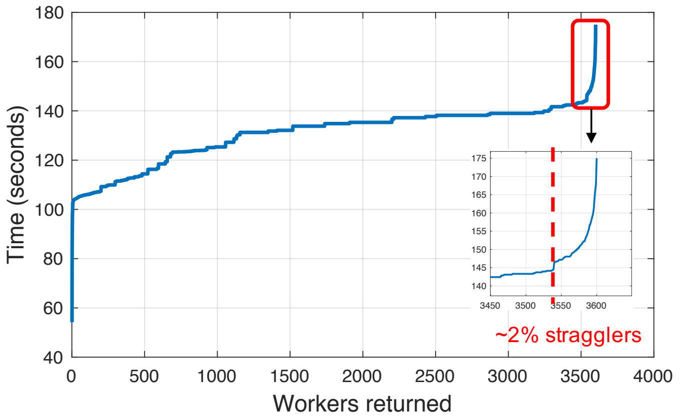
Serverless Systems: Characteristics

- Massive scale of low quality workers
- Workers do not communicate
  - Read/write data through a single data storage entity
- Workers are short-lived
- Stragglers and faults!

significant # of stragglers are observed in our experiments consistently



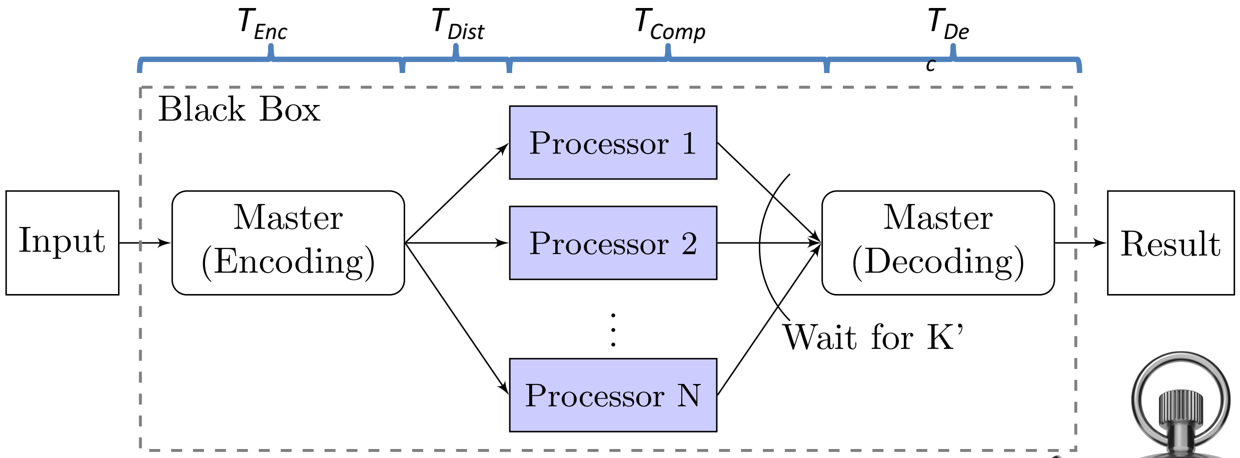
A single run snapshot

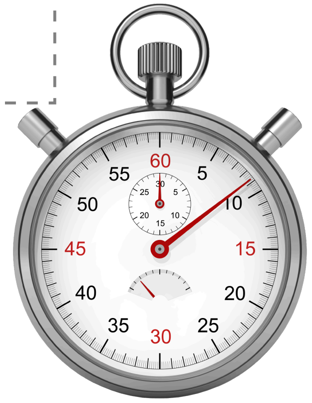


Average Runtimes over 10 trials

Can have up to 16,000 workers on AWS Lambda

What are we optimizing for?





- Matrix multiplication is a black box
  - MDS is beneficial, but target only T_{comp}
  - **End to end latency** is desired metric

Product Codes: a good tradeoff between near-MDS and local enc./dec.

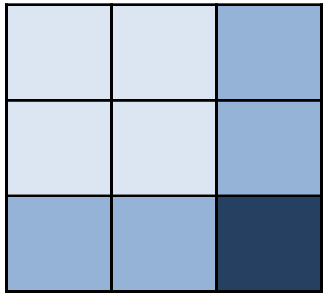
Product-coded (a.k.a. G-LDPC coded) Mat.-Mat. Mult.

G-LDPC codes [Tanner '81, Lentmaier-Z'99, Boutros et al. '99], **Product codes** [Elias '54, Justeson '07, JEN**R** '15]

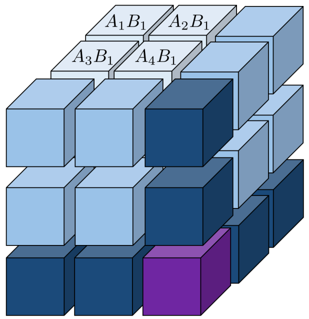
$$\begin{pmatrix} A_1 \\ A_2 \\ A_1 + A_2 \end{pmatrix} \times \begin{pmatrix} B_1 & B_2 & B_1 + B_2 \end{pmatrix}$$

$$= \begin{pmatrix} A_1B_1 & A_1B_2 & A_1(B_1+B_2) \\ A_2B_1 & A_2B_2 & A_2(B_1+B_2) \\ (A_1+A_2)B_1 & (A_1+A_2)B_2 & \end{pmatrix}$$

1. Near-MDS
2. Low ENC/DEC cost
3. DEC is parallelizable
4. N-dim product codes…

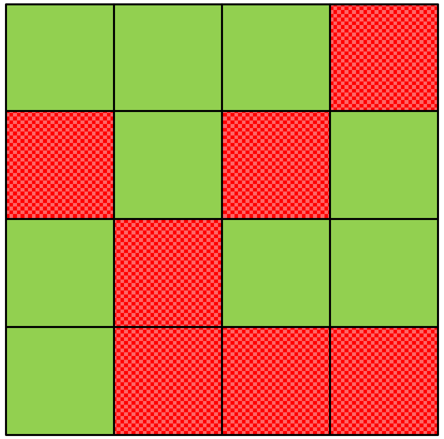


2D



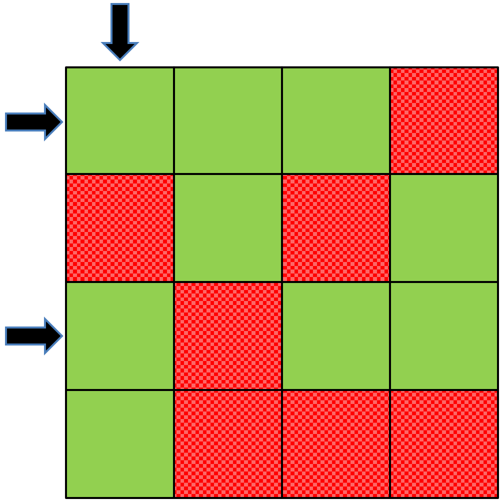
3D

Product Code Decoding



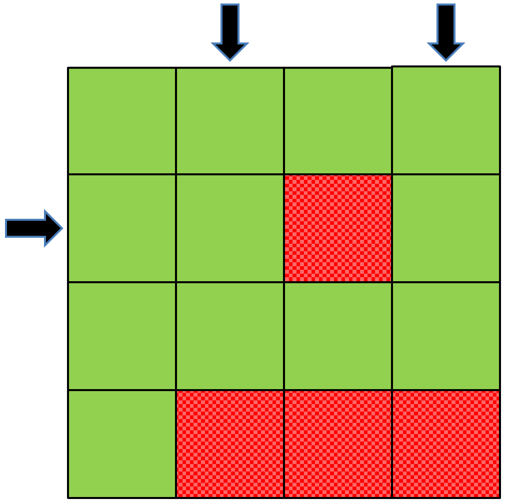
- Peeling decoder is very simple and parallelizable

Product Code Decoding



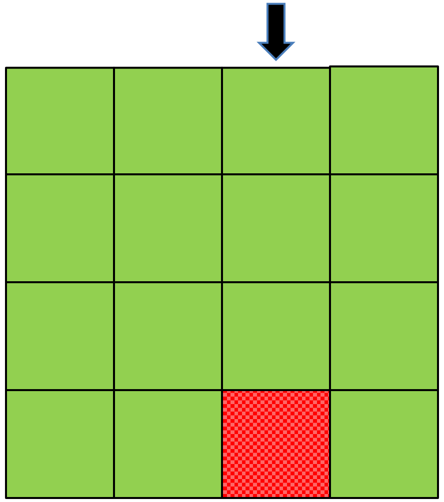
- Peeling decoder is very simple and parallelizable

Product Code Decoding



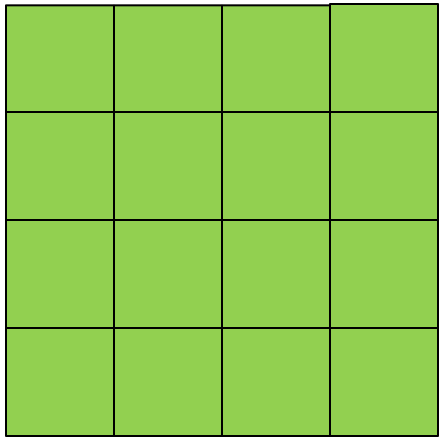
- Peeling decoder is very simple and parallelizable

Product Code Decoding



- Peeling decoder is very simple and parallelizable

Product Code Decoding



- Peeling decoder is very simple and parallelizable

Product-Coded MM: Performance

Result: *(Baharav & R'18)* In a **d**-dimensional product-coded matrix multiplication scheme with (**n, k, r+1**) component codes, the output will be decodable w.h.p. after $K' = N - \frac{N-K}{\eta(d,r)}$ nodes have completed their subtasks.

- Can tolerate $\frac{N-K}{\eta(d,r)}$ stragglers

TABLE II: Thresholds: $\eta(d, r)$

d \ r	1	2	3	4	5	6
3	1.2218	1.2880	1.3797	1.4564	1.5202	1.5741
4	1.2949	1.4998	1.6568	1.7781	1.8760	1.9575
5	1.4250	1.7275	1.9409	2.1031	2.2327	2.3406
6	1.5697	1.9577	2.2244	2.4256	2.5864	2.7199
7	1.7189	2.1869	2.5051	2.7446	2.9361	3.0953

Kernel Ridge Regression using Conjugate Gradient on AWS Lambda

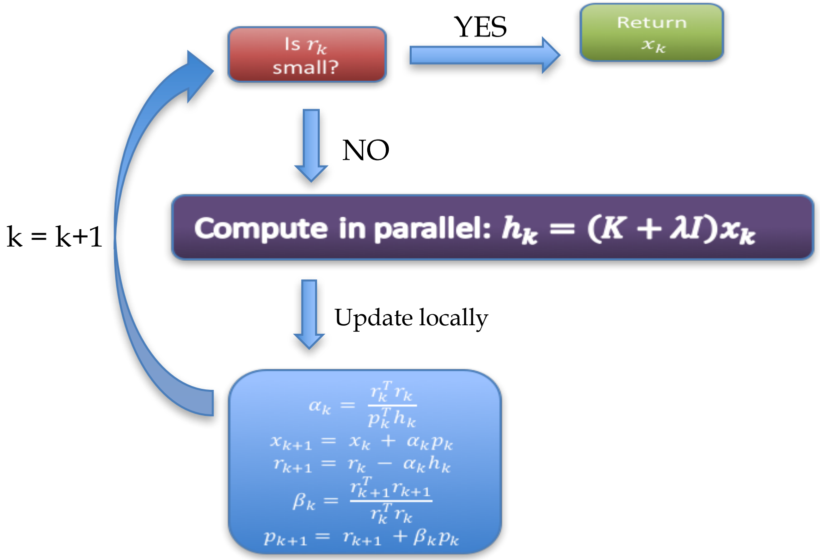
On a real-world dataset with $n = 0.4\,million$ examples and 400 workers

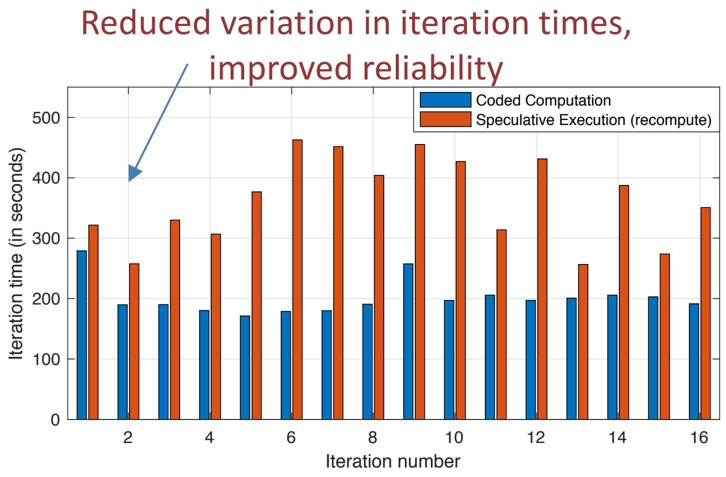
Problem:

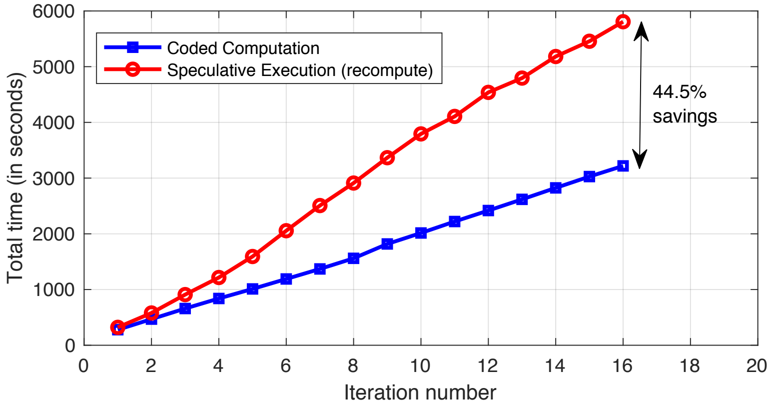
Solve for x in $(K + \lambda I)x = y$, K: Kernel matrix of dim. n

Initialization:

$$x_0 = 1^{n\times1}, r_0 = y - (K + \lambda I)x_0, p_0 = r_0, \; k = 0$$





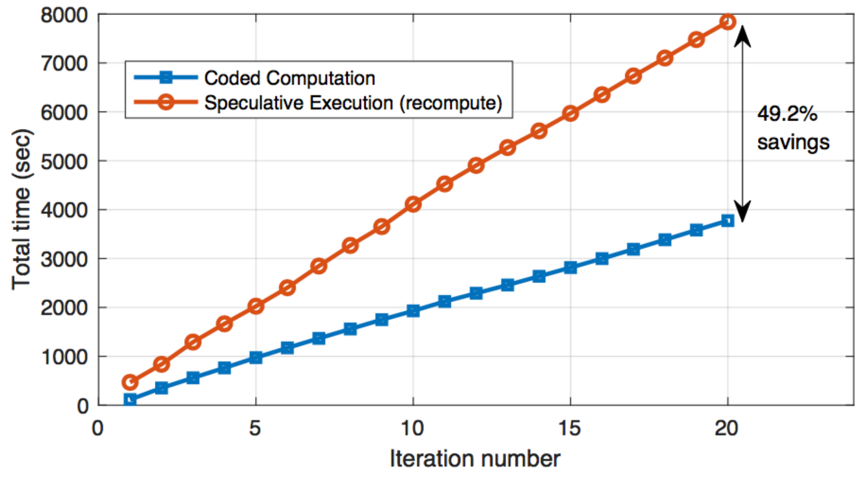


(First iteration includes the one-time encoding cost)

Power Iteration on serverless AWS Lambda

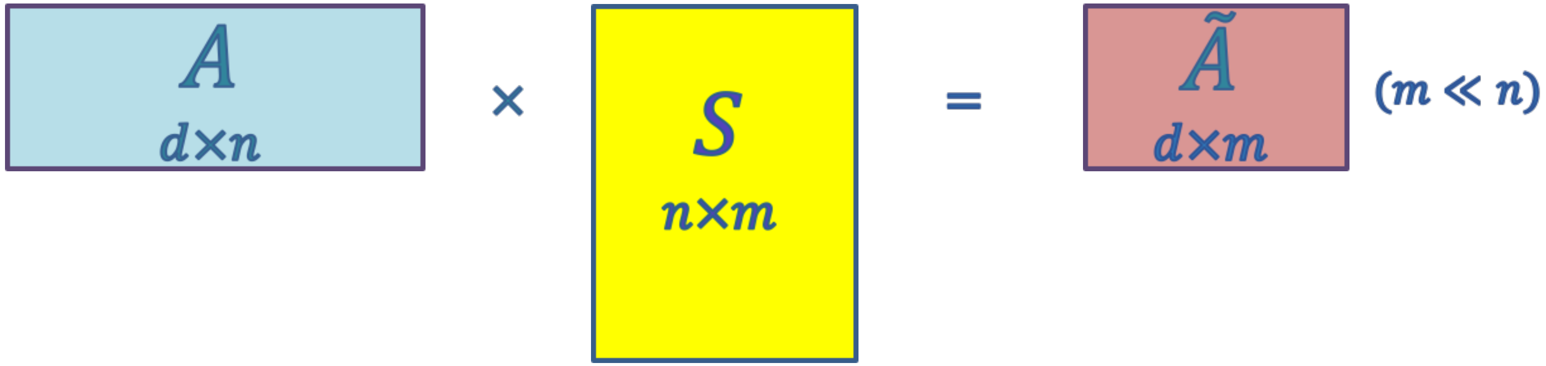
- Goal: Find the largest eigenvalue and eigenvector of a diagonalizable matrix A of dimension 0.5 $million$ with 1000 workers
- Applications: PCA, PageRank; Twitter recos. on whom to follow
- Each iteration: a matrix-vector multiplication $b_{k+1} = \frac{Ab_k}{\|Ab_k\|}$

~50% savings in total time! (1hour 6 min. less)



Matrix Multiplication: Sketching

- Exact computation is not necessary, especially if input data has redundancies
- Randomized sketching is an important technique to reduce comp. complexity
- To compute AA^T
  - Sketch the input matrix: $\tilde{A} = AS$



  - S is a random matrix such that SS^T is close to identity
  - Multiply the smaller matrices $\tilde{A}$ and $\tilde{A}^T$

Mahoney, M. W., 2011; Woodruff, D. P., 2014; Drineas et al., 2016; …….

Large-scale Convex Optimization on Serverless Systems

Recall the challenges in serverless systems:

- Slow communication
- Ephemeral workers
- Persistent stragglers

Hence, reducing the number of iterations is paramount

- Second-order methods are a natural fit for serverless systems
  - Reduce the number of iterations considerably
  - Exploit the tremendous compute power per iteration
- OverSketched Newton: Tailored to serverless systems

OverSketched Newton

Key Observation: For many common convex optimization problems

- Gradient can be written as a few large matrix-vector mults.
- Hessian can be written as a large matrix-matrix multiplication

Example problems:

- Logistic and linear regression,
- Softmax regression,
- SVMs,
- Linear program,
- Semidefinite programs,
- Lasso (in dual formulation), etc.

Example: Logistic Regression

$$\min_{w \in R^d} \left\{ f(w) = \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + e^{-y_i w^T x_i}\right) + \frac{\lambda}{2} \|w\|^2 \right\}$$

- $X = [x_1, \cdots, x_n] \in R^{d \times n}$ is the matrix containing training examples
- $y = [y_1, \cdots, y_n] \in R^n$ is vector containing training labels

- Gradient is given by

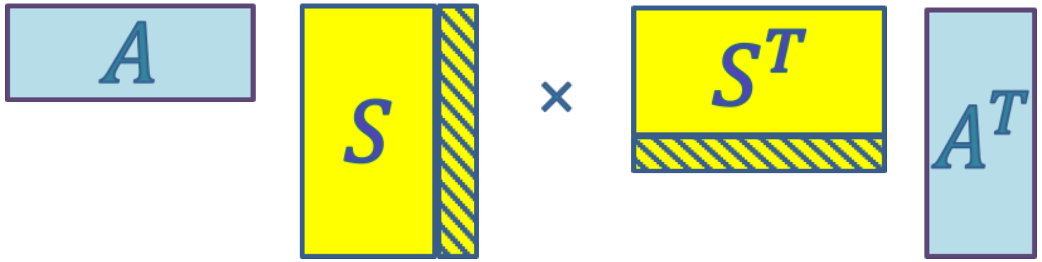
$$\nabla f(w) = \frac{1}{n} \sum_{i=1}^{n} \frac{-y_i x_i}{1 + e^{y_i w^T x_i}} + \lambda w$$

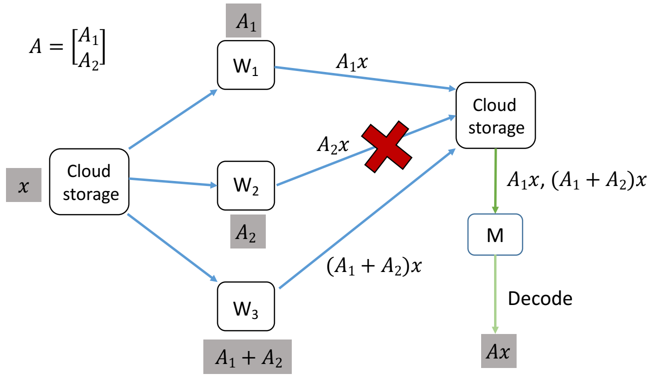
- Can be written as matrix-vector products
  $\nabla f(w) = X\beta + \lambda w$, where $\beta_i = \frac{-y_i}{1 + e^{y_i \alpha_i}}$ and $\alpha = X^T w$

- Hessian is given by
  $H^t = \frac{1}{n} X \Lambda X^T + \lambda I_d \in R^{d \times d}$
  - Λ is diagonal, $\Lambda(i,i) = \frac{-y_i}{1 + e^{y_i \alpha_i}}$
- Requires computation of AA^T, where
  $A = X\sqrt{\Lambda} \in R^{d \times n}, n \gg d$

OverSketched Newton

- Compute the gradient using classical coded computing
- Compute the Hessian approximately by “over sketching ”



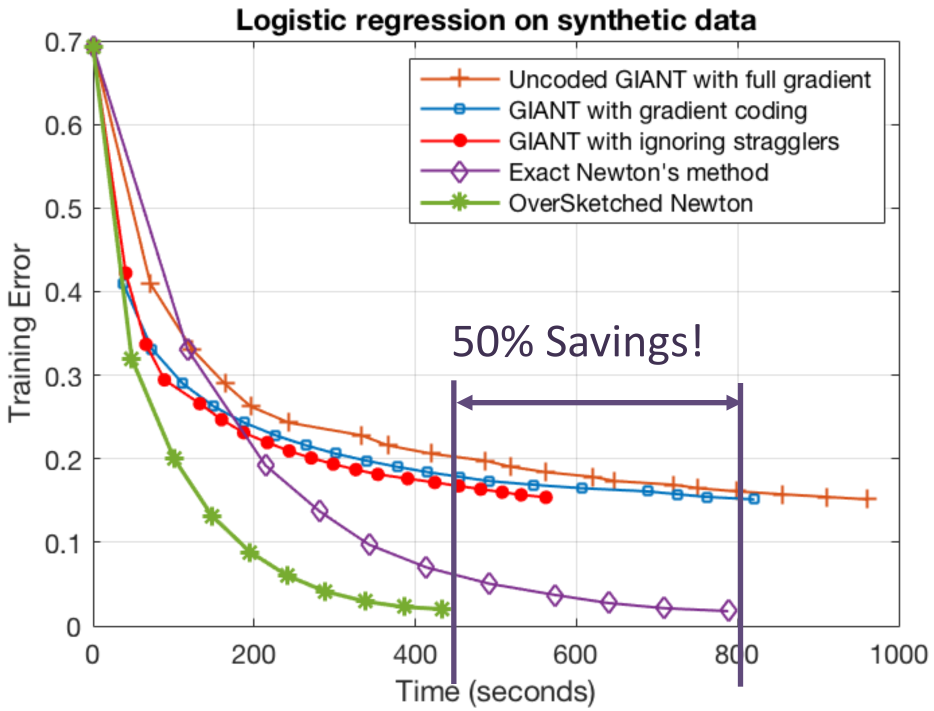


- Model update: $w^{t+1} = w^t - \hat{H}^{-1} g$
  - Can be done locally if *d* small enough

We prove convergence guarantees for OverSketched Newton when the objective is both strongly and weakly convex

Comparison with existing second-order methods

Experiments with n = 0.3 million examples and d = 3000 features on **AWS Lambda**

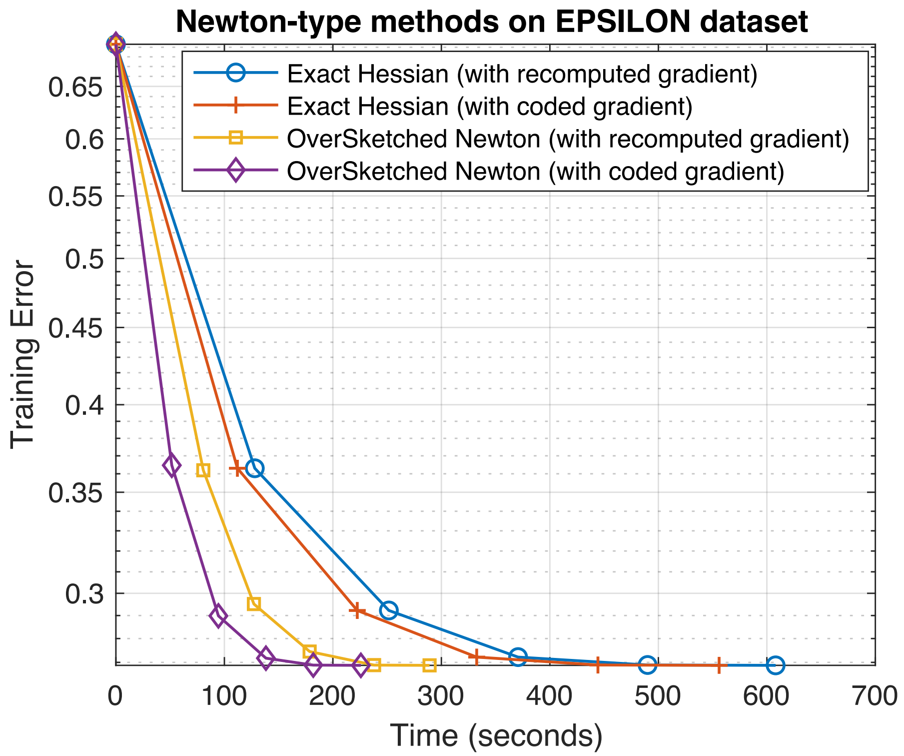


- GIANT: Linear-quadratic convergence when $n \gg d$
- 60 workers used for Gradient
- 3600 workers used to compute the exact Hessian
- 600 workers used to compute the sketched Hessian

Wang, Shusen, et al. "GIANT: Globally improved approximate newton method for distributed optimization." *NeurIPS*. 2018.

Coded computing vs Recomputing Stragglers

Experiments on logistic regression with n = 0.4 million and d = 2000



Codes used?

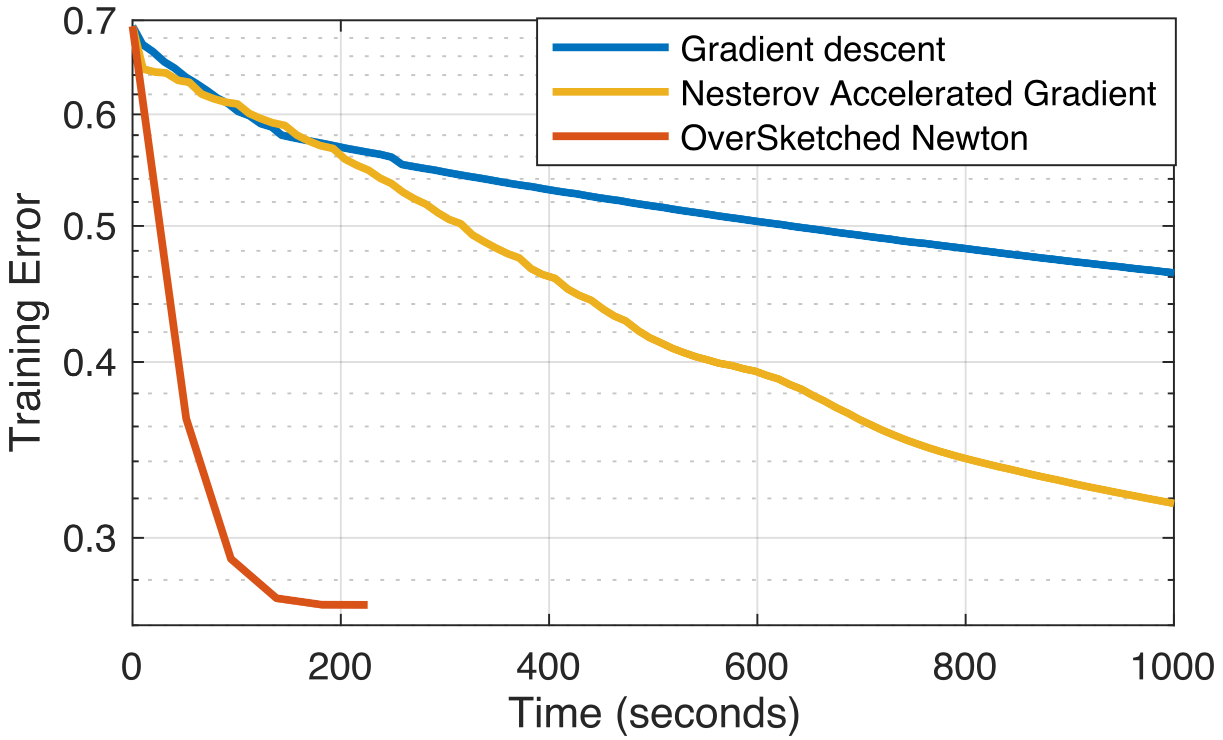
Gradient	Hessian
✗	✗
✓	✗
✗	✓
✓	✓

Running time

First order vs Second order on AWS Lambda

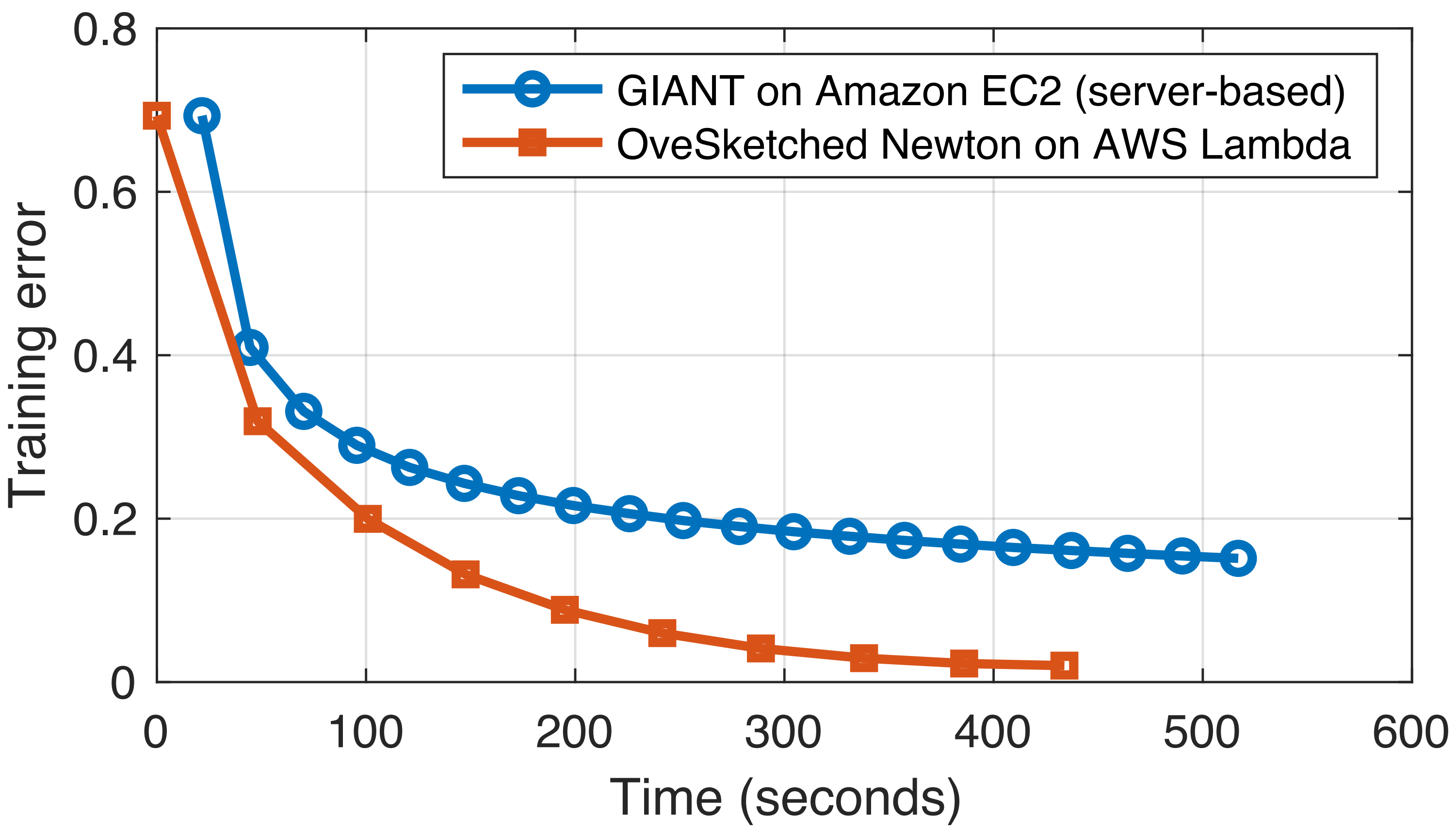
Experiments on a EPSILON dataset with **n = 0.4 million** ex. and **d = 2000 features**

- 100 workers used for Gradient computation
- 1500 workers used to compute the sketched Hessian



MPI (server-based) vs Serverless computing

Experiments on logistic regression with **n = 0.3 million** and **d = 3000**



Concluding Remarks

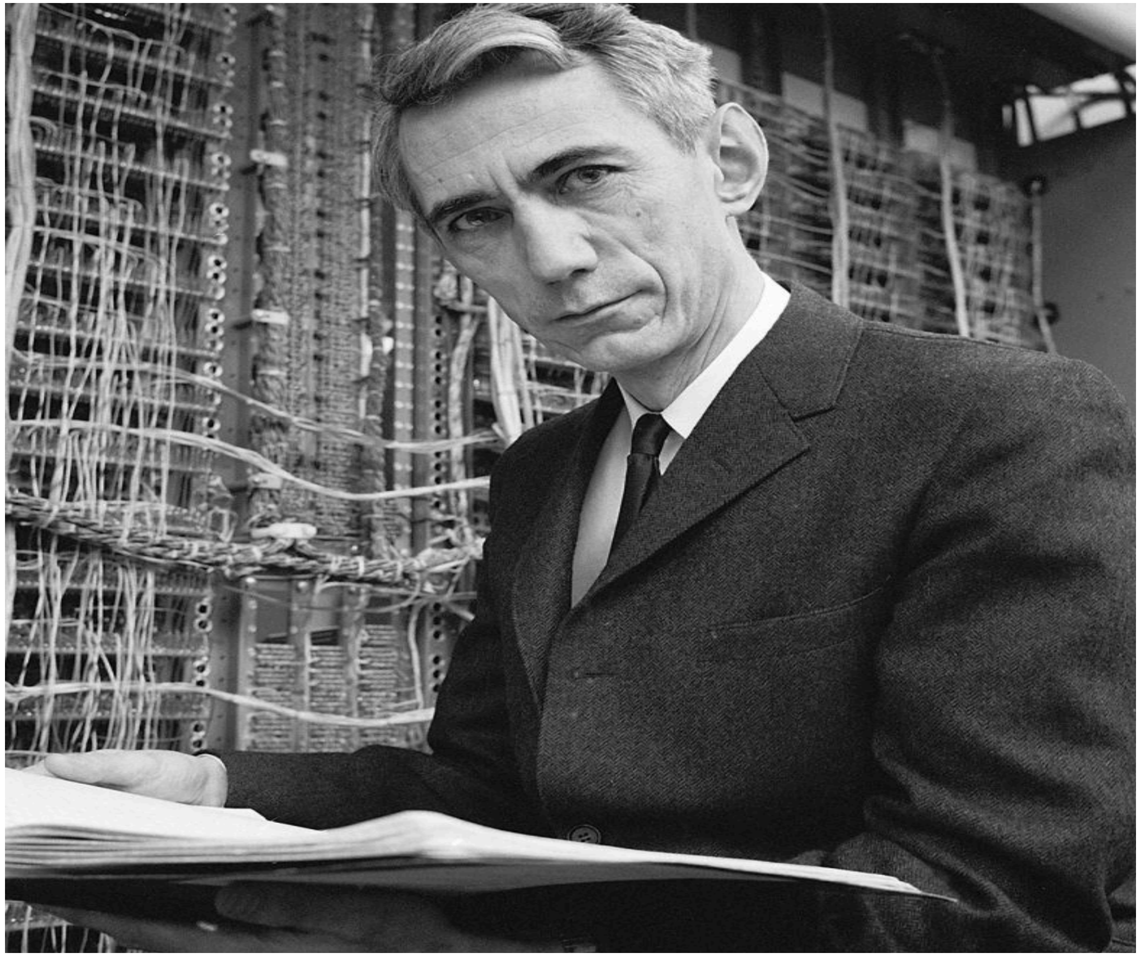
Shannon-inspired research threads on the power of codes in:

- ***Duality:***
  - “exchangability” of enc. and dec. functions in source/channel coding
- ***Encryption:***
  - “exchangability” of encryption & compression modules w/o perf. loss
- ***Sampling:***
  - unexplored connections between sampling theory and coding theory
- ***Learning:***
  - sparse-graph code based “peeling” core powerful in many sparse learning settings with sub-linear time complexity
- ***Distributed computing:***
  - straggler-proofing with codes speeds up distributed machine learning

Conclusion: Shannon's incredible legacy

- A mathematical theory of communication
- Channel capacity
- Source coding
- Channel coding
- Cryptography
- Sampling theory
- …

His legacy will last many more centuries!



(1916-2001)

Thank you!