Single-component privacy guarantees in Helper Data Systems and Sparse Coding with Ambiguation

Behrooz Razeghi, Slava Voloshynovskiy
Univ. of Geneva

Taras Stanko, Boris Škorić
Eindhoven Univ. of Technology

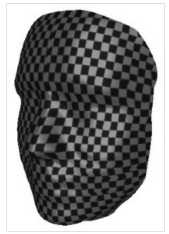
WIFS 2019
December 12

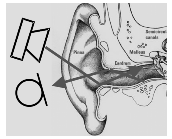
Outline

- Biometric privacy
- Attacker model & use case
- Two main approaches
  - Helper Data Systems
  - Sparse Coding with Ambiguation
- Single-component privacy
  - motivation
  - results

Biometric privacy

Not "secret". Why protect stored biometric data?





- Function creep
- Privacy
  - medical conditions
  - database crossmatching
  - tracking
- Security of biometric authentication
  - fake biometrics
  - sensor spoofing
- Framing
  - synthesized fingerprints/DNA at crime scene

Attacker model & use case

Use case: Biometric authentication

- biometric only.
  - no typed PINs
  - no prover device

Attacker model:

- no access to biometric during enrolment / verification
- full access to enrolled data
  - insider
  - hacker
- full access to encryption keys
- there is no special secure hardware

Problem: How to store biometric enrolment data?

Approach #1: Helper Data System + hash

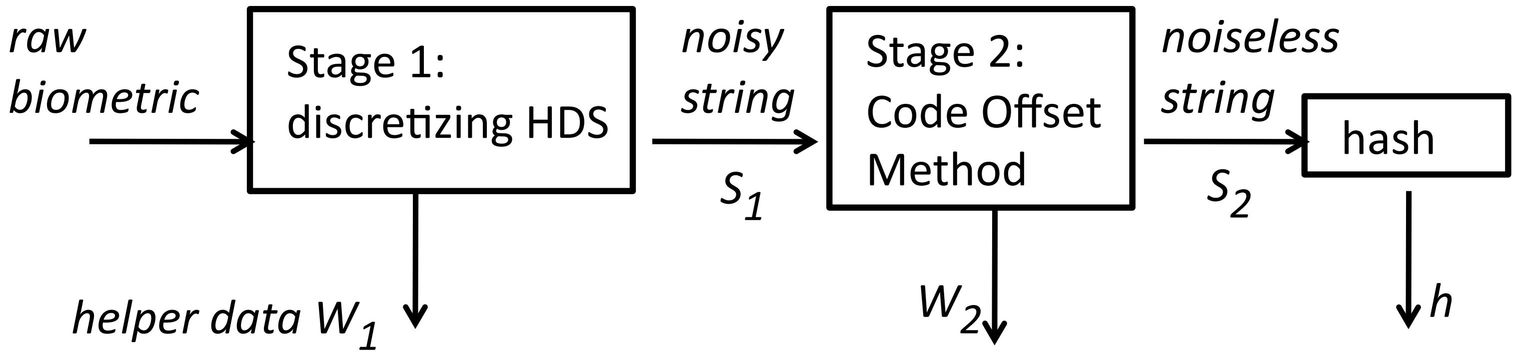
Store **hash** of biometric data ← *just like passwords!*

- needs error correction
- adversary sees redundancy data

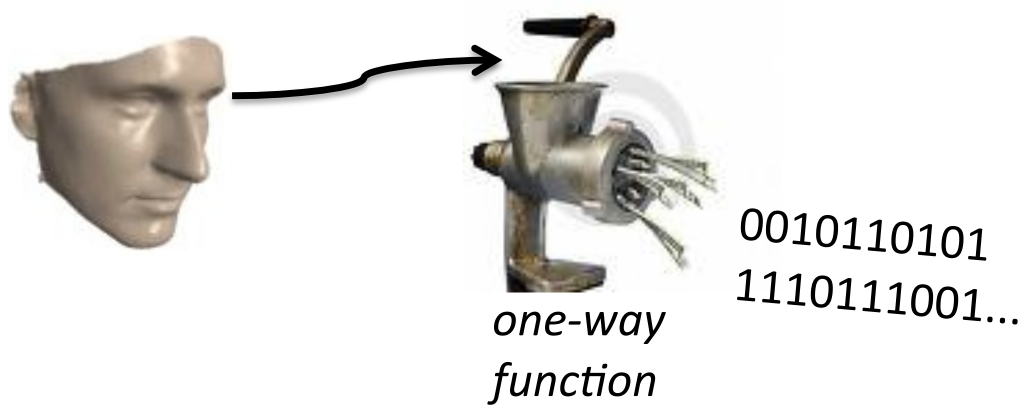
Two-stage secure error correction

1. "Zero Leakage" disretizing HDS
2. Code Offset Method

*"**Helper Data System**"*
(secure sketch, fuzzy extractor)



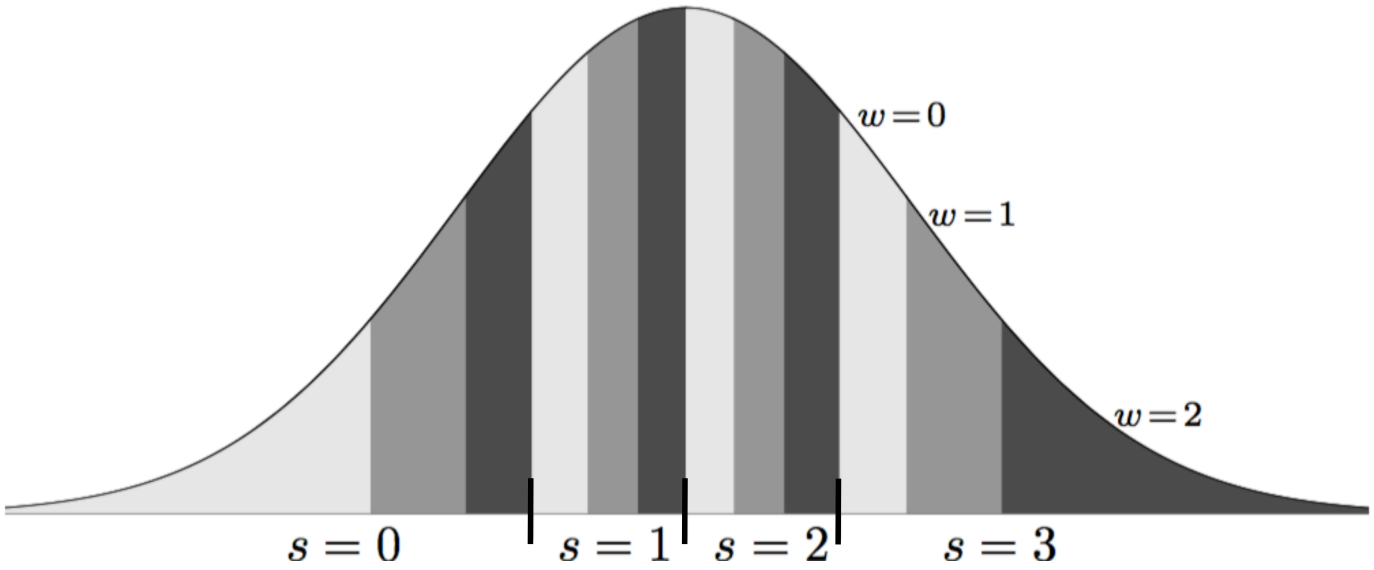
Store enrolment data: (ID, W_1, W_2, h). The W_j should not leak about S_j.


one-way
function
0010110101
1110111001...

Zero-Leakage discretizing HDS

[de Groot et al. 2012]
[Stanko et al. 2017]

- split data into 1D features (real numbers)
- apply stage1 HDS to each dimension separately



Helper Data w = "least signifcant digits"

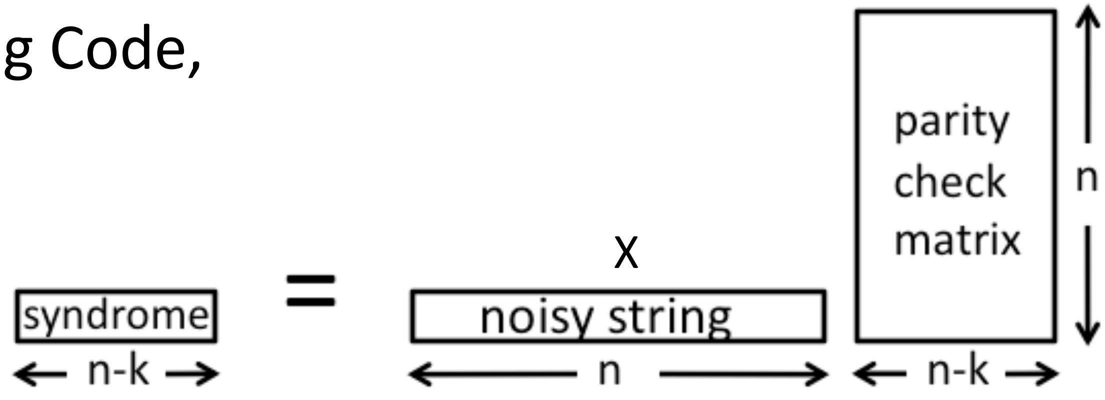
- in **quantile form**
- does not leak about Most Significant Digits (s)

Reconstruction: go to nearest interval that has correct index *w*

The Code Offset Method

[Bennett et al. 1991]
[Juels+Wattenberg 1999]
[Dodis et al. 2008]

Use linear Error-Correcting Code,
with syndrome decoder.
Message length k;
codeword length n;
syndrome length n-k.



Enrollment: **W = Syn X** *"least significant digits" !*

Reconstruction: $\hat{\mathbf{X}}$ **= X' ⊕ SynDec(W⊕Syn X')**

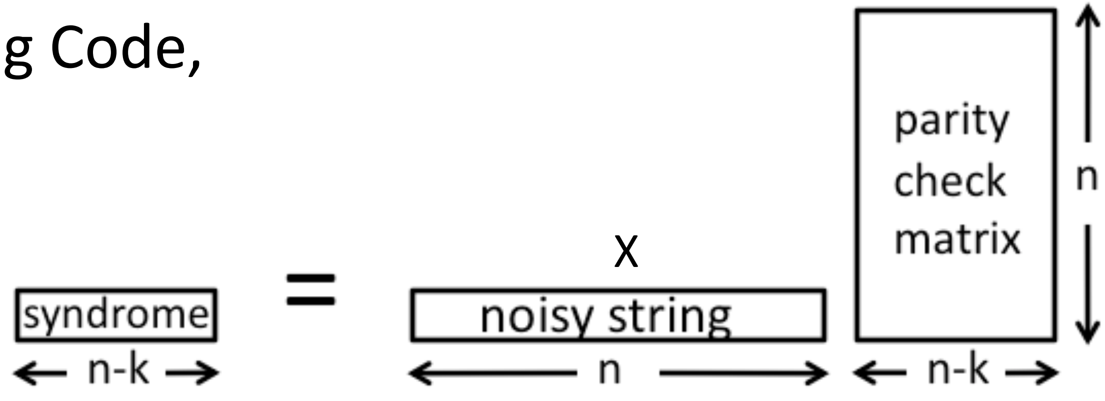
yields error pattern

Syn(x⊕x')

The Code Offset Method

[Bennett et al. 1991]
[Juels+Wattenberg 1999]
[Dodis et al. 2008]

Use linear Error-Correcting Code,
with syndrome decoder.
Message length k;
codeword length n;
syndrome length n-k.



Enrollment: **$W = \text{Syn } X$** *"least significant digits" !*

Reconstruction: **$\hat{X} = X' \oplus \text{SynDec}(W \oplus \text{Syn } X')$**

yields error pattern

$\text{Syn}(x \oplus x')$

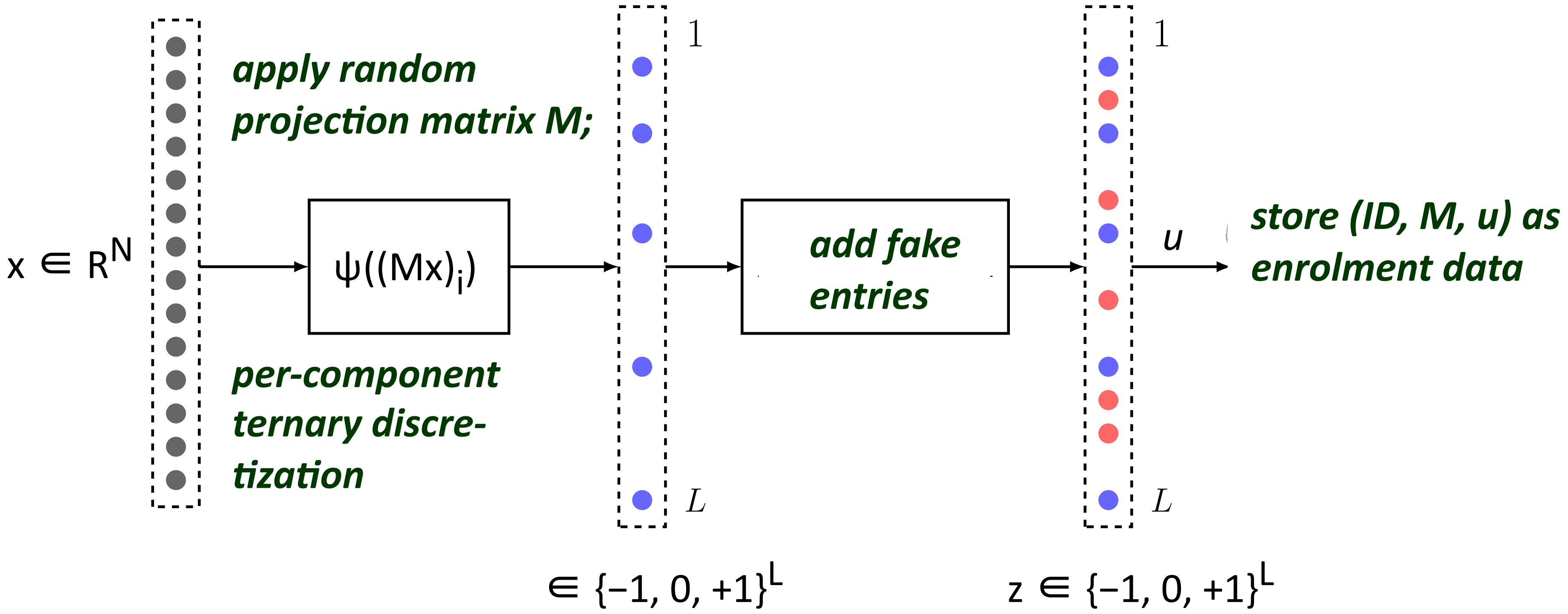
The Spammed Code Offset Method [Skoric + de Vreede 2014]

- hide w in lots of fake helper data

Approach #2: Sparse Coding [Razeghi et al. 2017]

Sparse Coding with Ambiguation

- sort of Locality Sensitive Hash, but with artificial noise
- no error-correcting code



Verification of vector y: inner product $u \cdot \psi(My)$ should be large enough

Privacy

	Helper Data Approach	***Sparse Coding approach***
Philosophy	Reveal least significant part of X • noisy anyway • does not represent X, but noise	Reveal location of reliable parts • use *polarisation* effect of random projections • add fake entries for privacy
Advantages	• compact • well controlled privacy	No ECC
Disadvantages	• input must have high entropy • error-correcting code	• reveals signs of reliable parts • enrolment data not compact (?)

Privacy

	Helper Data Approach	***Sparse Coding approach***
Philosophy	Reveal least significant part of X • noisy anyway • does not represent X, but noise	Reveal location of reliable parts • use *polarisation* effect of random projections • add fake entries for privacy
Advantages	• compact • well controlled privacy	No ECC
Disadvantages	• input must have high entropy • error-correcting code	• reveals signs of reliable parts • enrolment data not compact (?)

CAVEAT

We are ignoring other approaches!

- homomorphic crypto ← *slow; needs trusted party*
- Locality Sensitive Hashing ← *privacy unclear*
- ....

Single-component privacy guarantees

Biometric feature vector $X \in R^N$

Motivation

- What if one biometric feature X_i reveals a medical condition?

We investigate two aspects of such leakage

- sign of X_i
- $|X_i|$ > threshold?

Results for HDS: first stage

Under the assumption of even prob.distribution of X_i

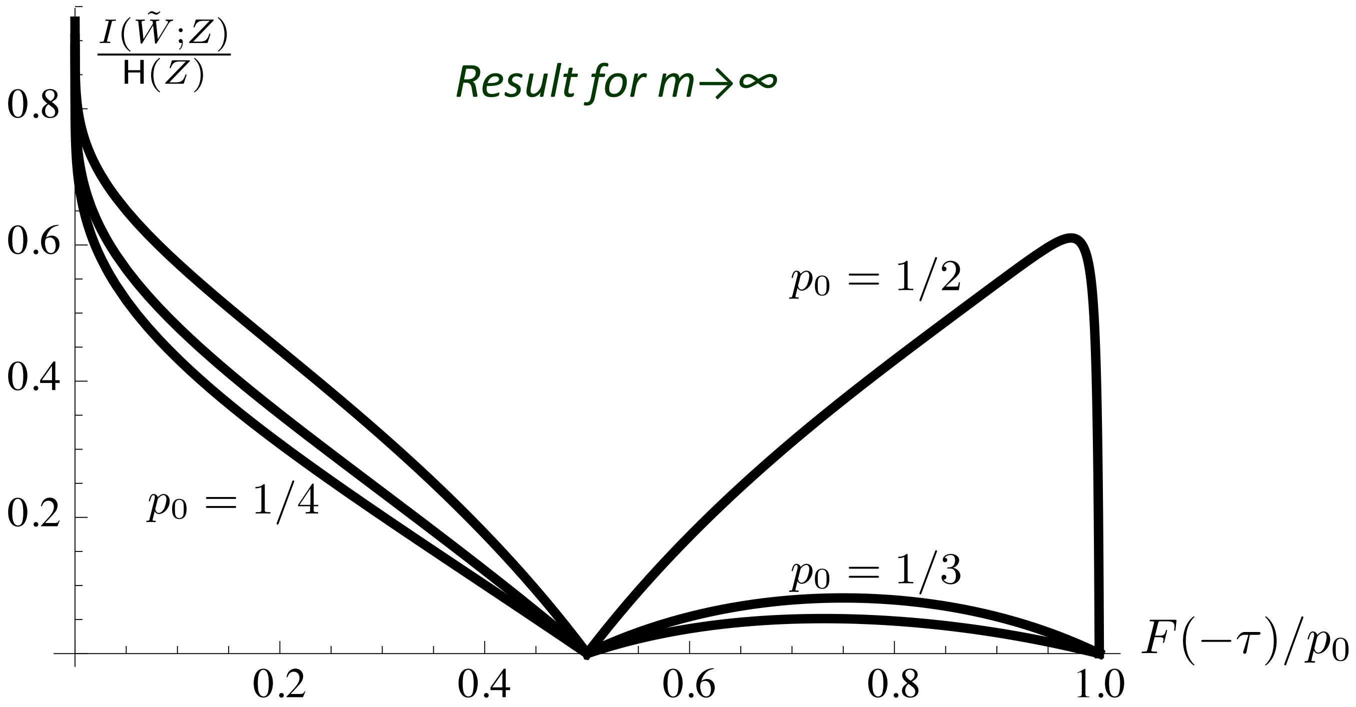
Leakage about sign(X_i)

- none, if #quant.intervals is even
- (some leakage if odd)

m = #helper data values
p_0 = Prob[S=0]

Leakage about binary variable Z = [|X_i| > τ]

- *assuming large threshold* τ: **no leakage at m=2**
- nonzero at m>2



Results for HDS: 2nd stage

Sign of X_i becomes bit value

- input for 2nd stage
- Does the Code Offset Method leak this bit?

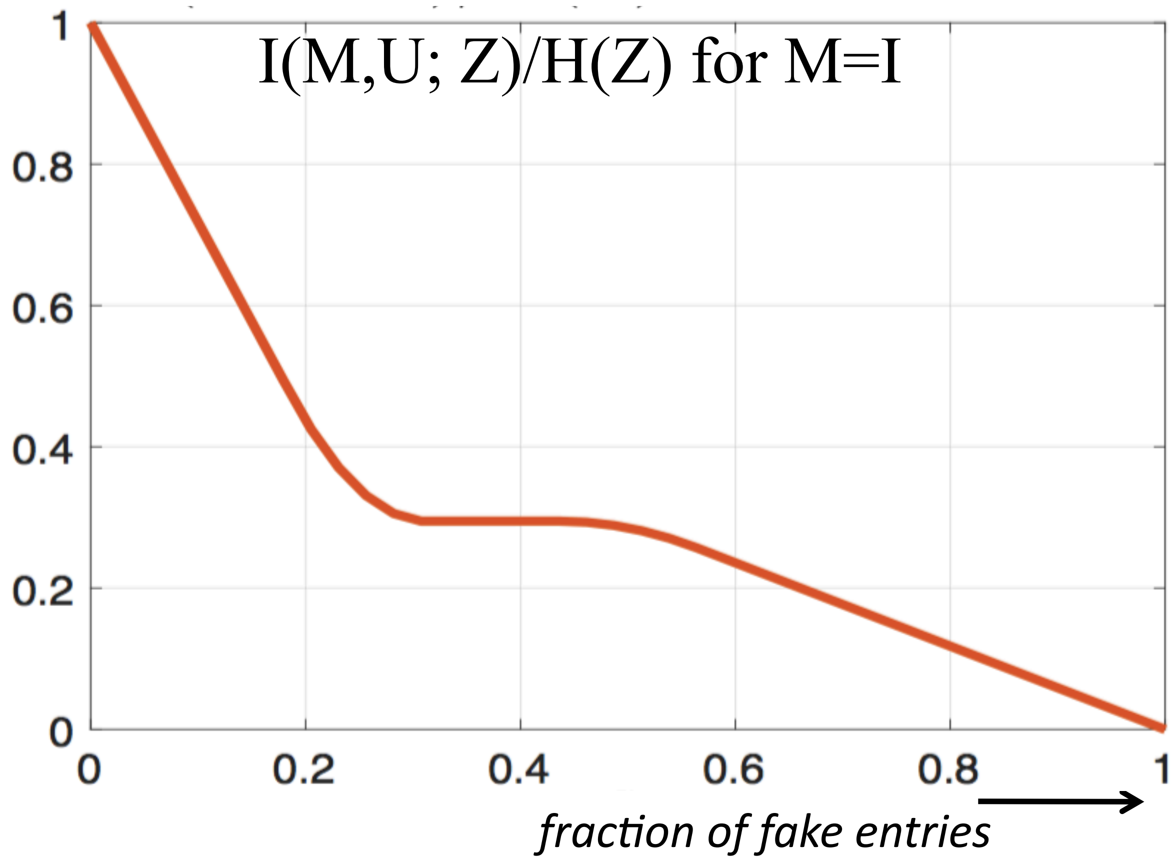
Answer: **the leakage is exponentially small**.

Total leakage about COM input $\approx (N-k)[1-h\left(\frac{1}{2}-\frac{1}{2}(1-2\varepsilon)^r\right)]$

ε = bit error rate
r = row weight of the code

Results for Sparse Coding with Ambiguation

- Very little leakage about **magnitude** of X_i



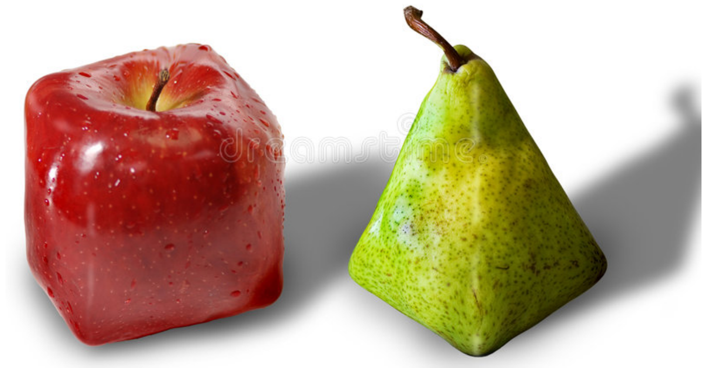
- **Sign** of X_i:
  Work in progress.
  Adversary's reconstruction prob. of whole X is small.

Summary

- Biometrics
- Single-component privacy guarantees
- Comparison of two template protection approaches (apples vs. pears)

Summary

- Biometrics
- Single-component privacy guarantees
- Comparison of two template protection approaches (apples vs. pears)



Apples and pears are different, but both taste good!

- Helper Data approach (1st stage):
  - choose even #quant.intervals
  - one-bit helper data works best
- Sparse Coding approach:
  - minimal leakage about single-component magnitude
  - low overall reconstruction probability