Blind Unwrapping of Modulo Reduced Gaussian Vectors: Recovering MSBs from LSBs

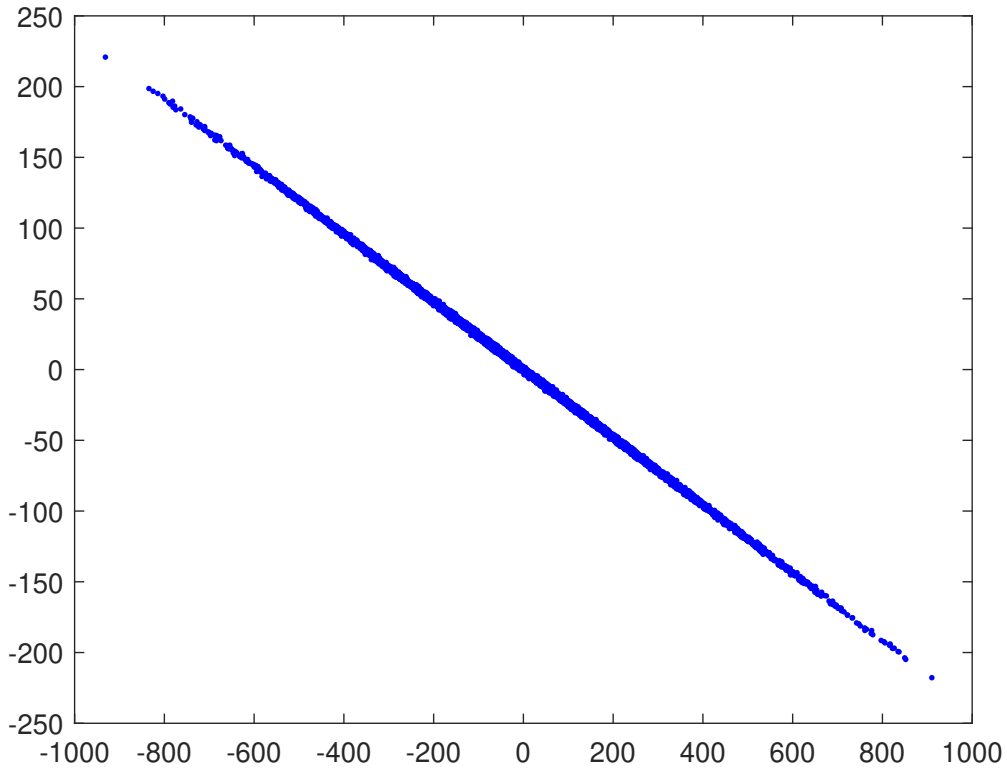
Or Ordentlich (Hebrew University of Jerusalem)
Joint work with Elad Romanov (Hebrew University of Jerusalem)

ITA, UCSD

February, 2019

Problem Setup

- Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, and $\mathbf{X}_1, \ldots, \mathbf{X}_n$ iid samples



Scatter plot

Problem Setup

- Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, and $\mathbf{X}_1, \ldots, \mathbf{X}_n$ iid samples
- For $\Delta > 0$, define

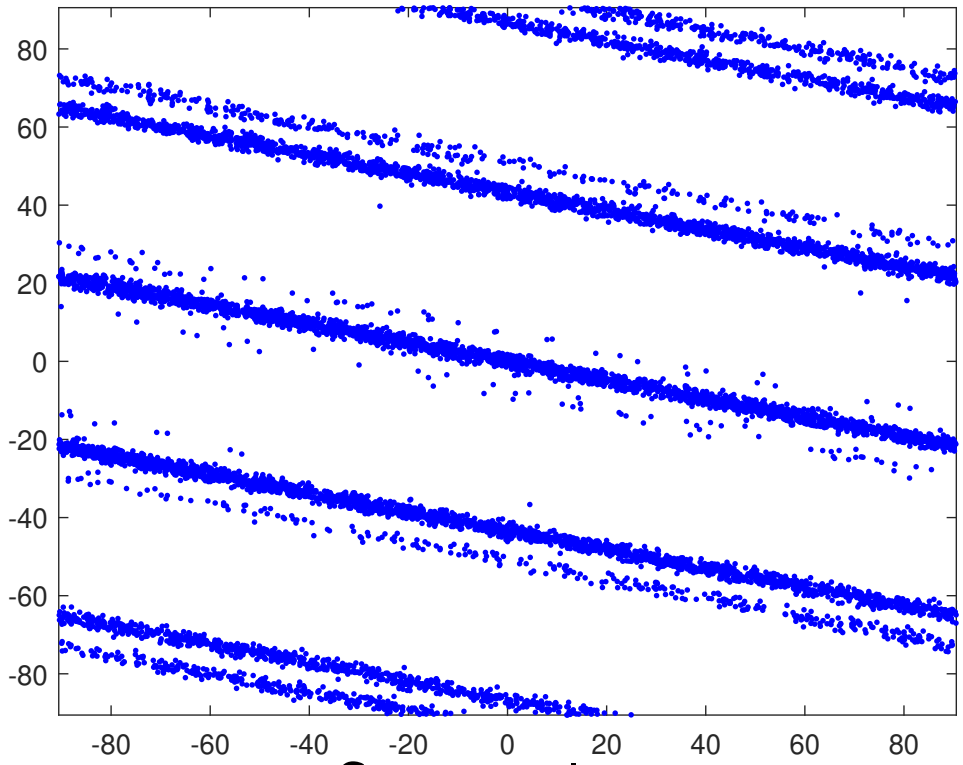
$$t^* = [t] \bmod \Delta \triangleq t - Q_{\Delta\mathbb{Z}}(t) \in \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)$$

  where $Q_{\Delta\mathbb{Z}}(t) \triangleq \arg\min_{b \in \mathbb{Z}} |t - b\Delta|$
- $\mathbf{x}^*$ is the vector obtained by applying the modulo operation on each coordinate of $\mathbf{x}$, i.e.,

$$\mathbf{x}^* \triangleq [x_1^* \ \cdots \ x_K^*]^T.$$

Problem Setup

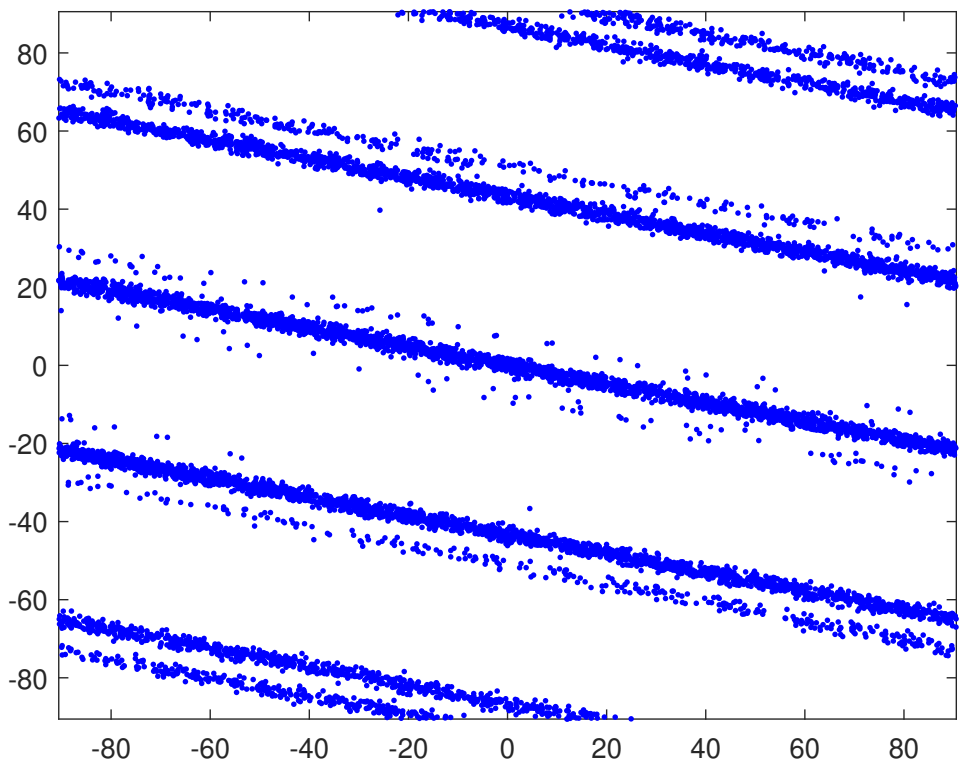
- Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, and $\mathbf{X}_1, \ldots, \mathbf{X}_n$ iid samples
- Assume we observe $\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*$



Scatter plot

Problem Setup

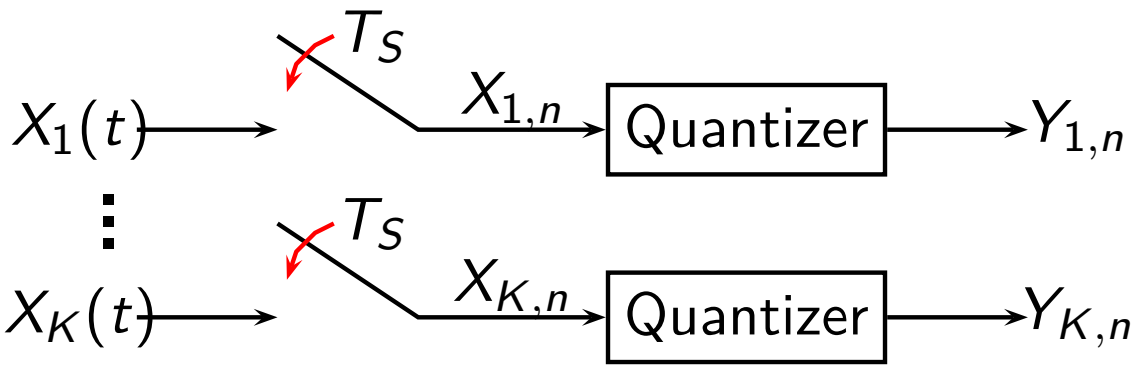
- Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, and $\mathbf{X}_1, \ldots, \mathbf{X}_n$ iid samples
- Assume we observe $\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*$



How can we recover $\mathbf{X}_1, \ldots, \mathbf{X}_n$ when Σ is not known?

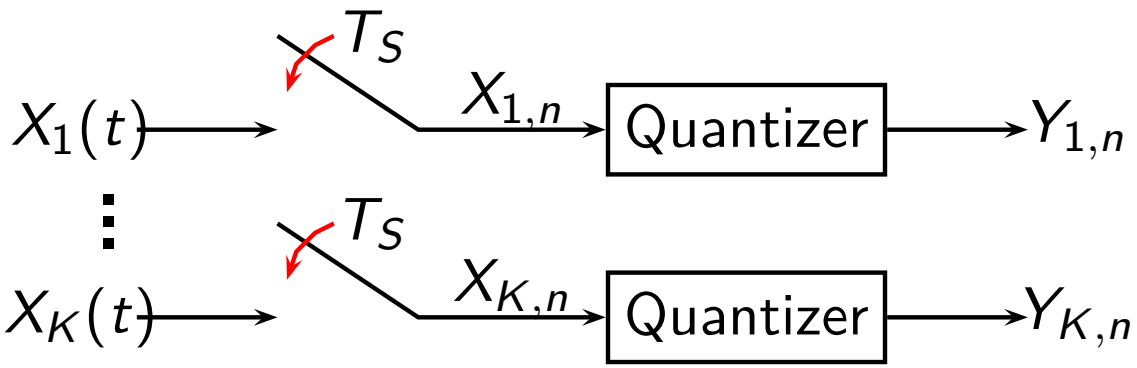
Motivation

- Analog-to-digital conversion of correlated processes



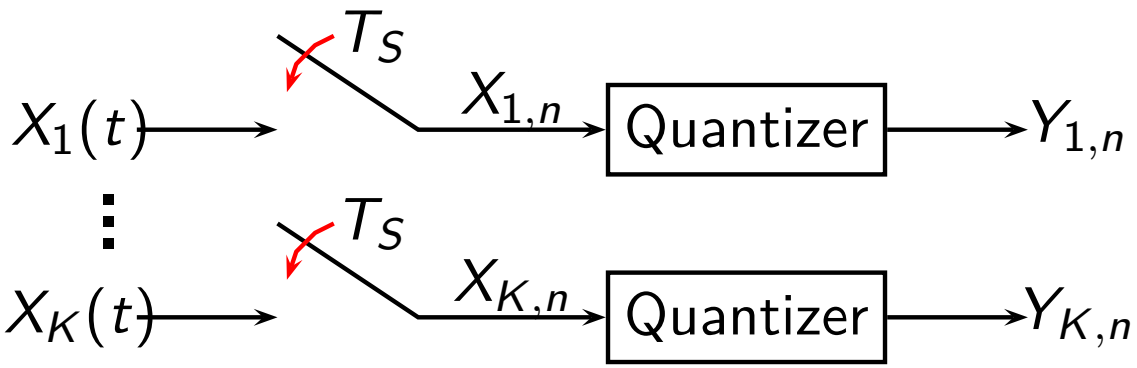
Motivation

- Analog-to-digital conversion of correlated processes



Motivation

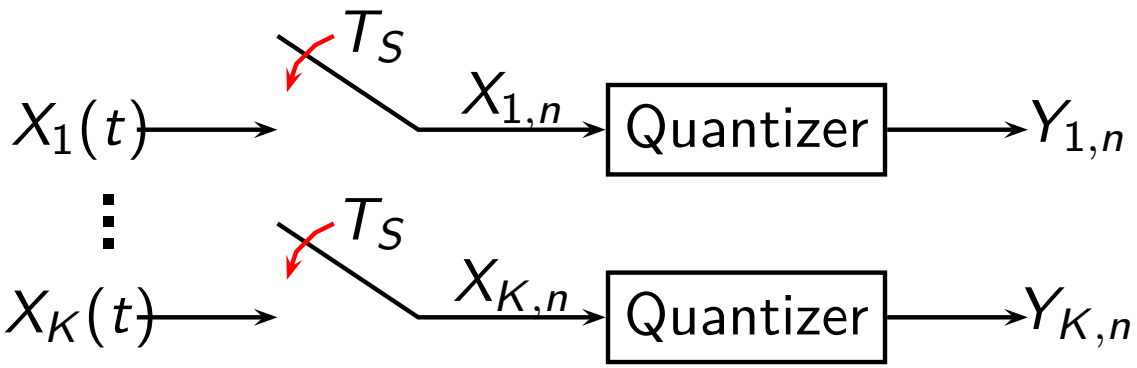
- Analog-to-digital conversion of correlated processes



- The quantizer is implemented using a mixed signal circuit
  $\Rightarrow$ must be simple, memoryless, and universal
- Default choice is uniform scalar quantizer (quantize to nearest grid point)

Motivation

- Analog-to-digital conversion of correlated processes

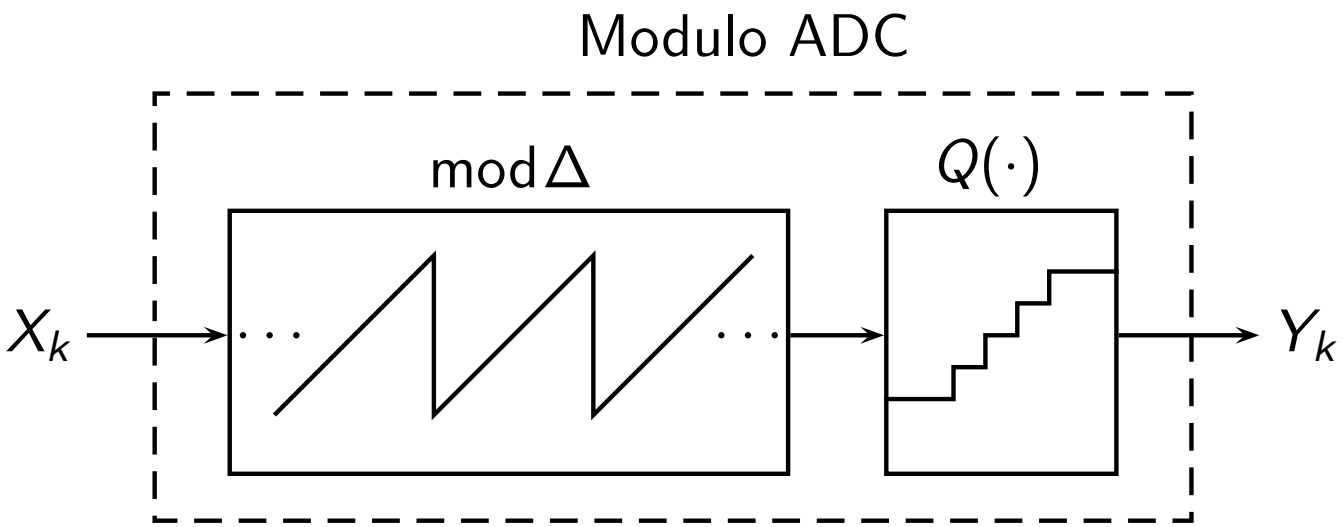


- The quantizer is implemented using a mixed signal circuit
  $\Rightarrow$ must be simple, memoryless, and universal
- Default choice is uniform scalar quantizer (quantize to nearest grid point)

How can we exploit correlation to reduce quantization rate?

Motivation

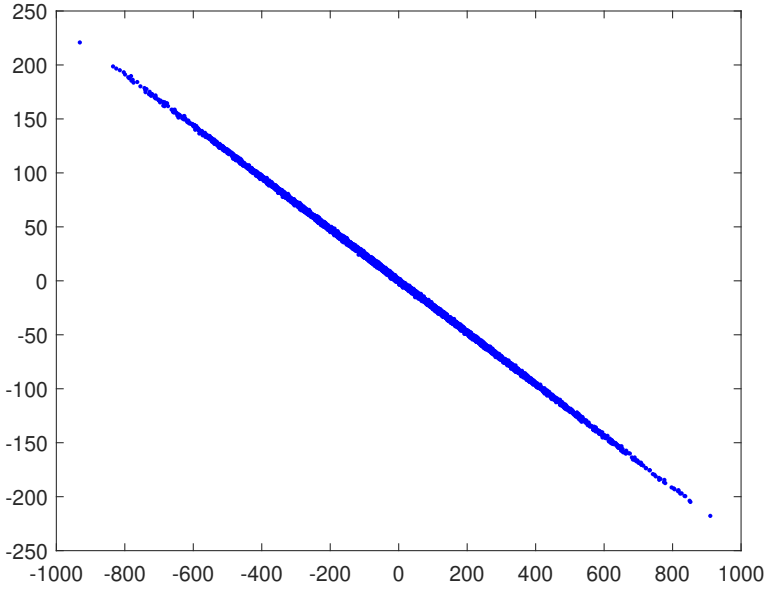
Instead of quantizing each coordinate X_k, quantize
$[X_k] \bmod \Delta$



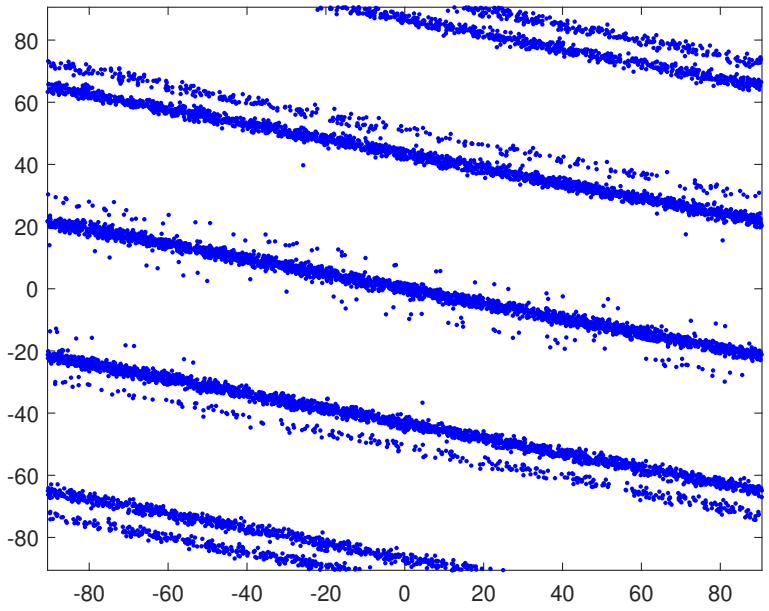
[O.-Erez IT'17], [O.-Tabak-Hanumolu-Singer-Wornell J-STSP'18]

Motivation

- The gain: Need to quantize variables with smaller support



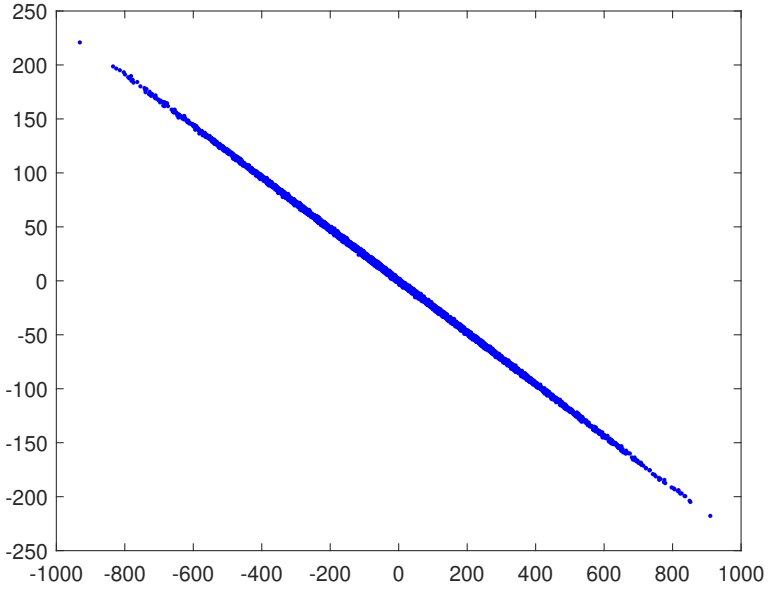
(a) Original



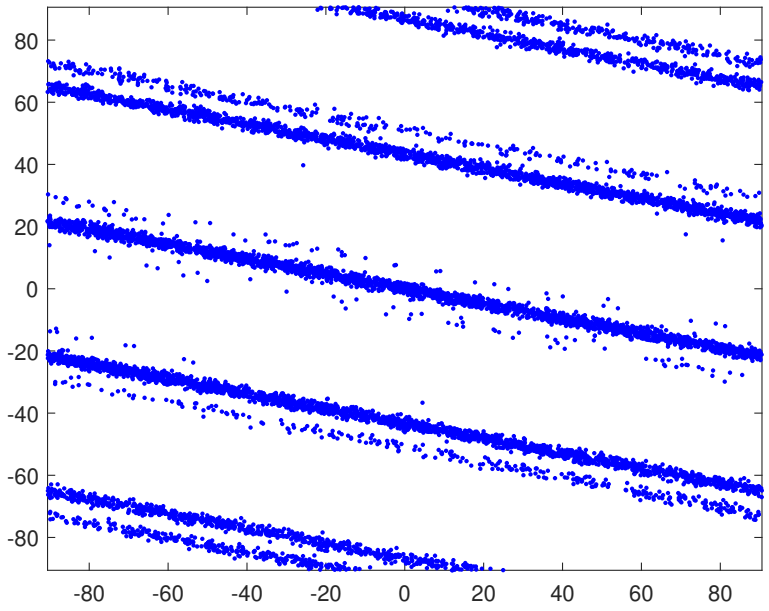
(b) Folded

Motivation

- The gain: Need to quantize variables with smaller support



(c) Original



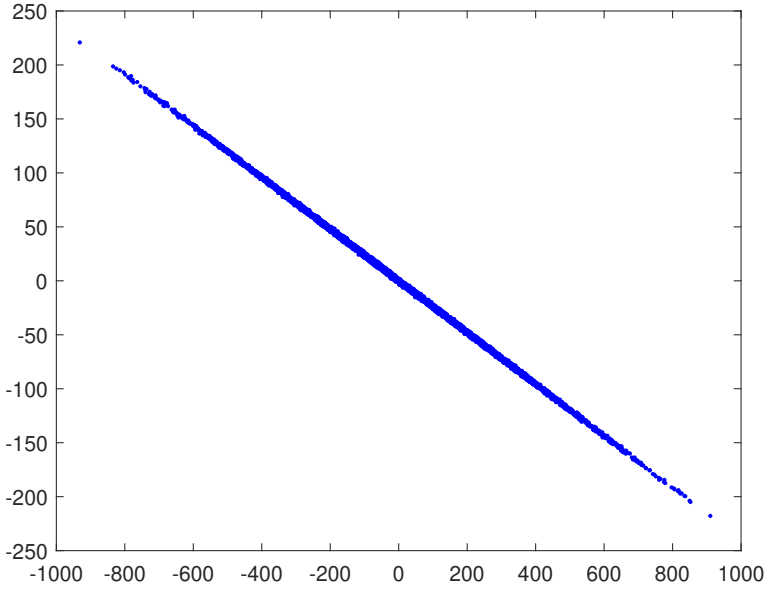
(d) Folded

$\Rightarrow$ Quantization can be done with the same accuracy (distortion) using less bits

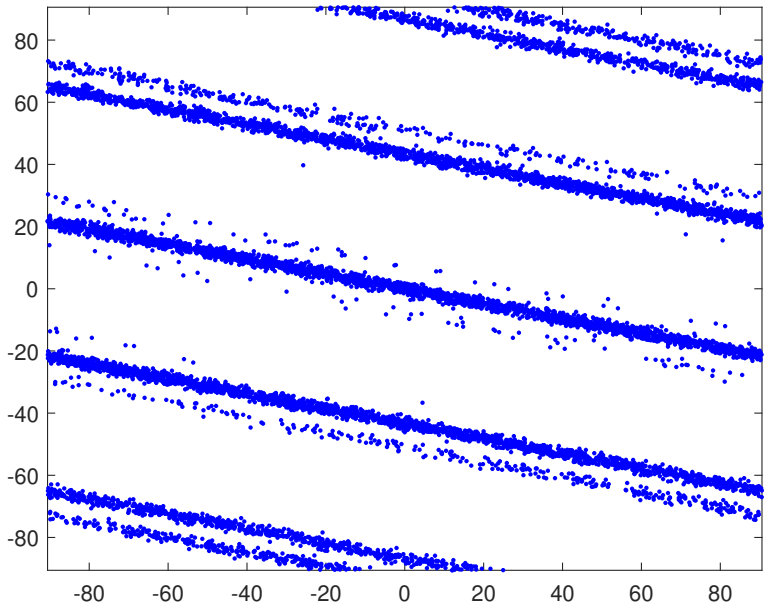
$\Rightarrow$ In ADCs Less bits = Smaller power consumption

Motivation

- The gain: Need to quantize variables with smaller support



(e) Original



(f) Folded

But... need to unwrap modulo measurements

Solving for known Σ - Suboptimal IF decoding

Fact: For any integer vector $\mathbf{a} \in \mathbb{Z}^K$ and and $\mathbf{x} \in \mathbb{R}^K$ it holds that

$$[\mathbf{a}^T[\mathbf{x}] \bmod \Delta] \bmod \Delta = [\mathbf{a}^T\mathbf{x}] \bmod \Delta.$$

In particular: $\forall \mathbf{A} \in \mathbb{Z}^{K\times K} \ : \ \mathbf{A}\mathbf{x} \in \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K \Rightarrow [\mathbf{A}\mathbf{x}^*]^* = \mathbf{A}\mathbf{x}$

Solving for known Σ - Suboptimal IF decoding

Fact: For any integer vector $\mathbf{a} \in \mathbb{Z}^K$ and and $\mathbf{x} \in \mathbb{R}^K$ it holds that

$$[\mathbf{a}^T[\mathbf{x}] \bmod \Delta] \bmod \Delta = [\mathbf{a}^T\mathbf{x}] \bmod \Delta.$$

In particular: $\forall \mathbf{A} \in \mathbb{Z}^{K\times K} \ : \ \mathbf{A}\mathbf{x} \in \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K \Rightarrow [\mathbf{A}\mathbf{x}^*]^* = \mathbf{A}\mathbf{x}$

For any invertible $\mathbf{A} \in \mathbb{Z}^{K\times K}$ we can use the sub-optimal decode

$$\hat{\mathbf{x}}_{\mathrm{IF}}(\mathbf{x}^*) = \mathbf{A}^{-1}\left([\mathbf{A}\mathbf{x}^*]^*\right)$$

which is correct iff $\mathbf{A}\mathbf{x} \in \mathrm{CUBE} \triangleq \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K$

Solving for known Σ - Suboptimal IF decoding

Fact: For any integer vector $\mathbf{a} \in \mathbb{Z}^K$ and and $\mathbf{x} \in \mathbb{R}^K$ it holds that

$$[\mathbf{a}^T[\mathbf{x}] \bmod \Delta] \bmod \Delta = [\mathbf{a}^T\mathbf{x}] \bmod \Delta.$$

In particular: $\forall \mathbf{A} \in \mathbb{Z}^{K\times K} \ : \ \mathbf{A}\mathbf{x} \in \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K \Rightarrow [\mathbf{A}\mathbf{x}^*]^* = \mathbf{A}\mathbf{x}$

For any invertible $\mathbf{A} \in \mathbb{Z}^{K\times K}$ we can use the sub-optimal decode

$$\hat{\mathbf{x}}_{\mathsf{IF}}(\mathbf{x}^*) = \mathbf{A}^{-1}\left([\mathbf{A}\mathbf{x}^*]^*\right)$$

which is correct iff $\mathbf{A}\mathbf{x} \in \mathrm{CUBE} \triangleq \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K$

Error probability: $\epsilon_{\mathsf{IF}} \triangleq \Pr\left(\hat{\mathbf{X}}_{\mathsf{IF}}(\mathbf{X}^*) \neq \mathbf{X}\right) = \Pr(\mathbf{A}\mathbf{X} \notin \mathrm{CUBE})$

Solving for known Σ - Suboptimal IF decoding

Fact: For any integer vector $\mathbf{a} \in \mathbb{Z}^K$ and and $\mathbf{x} \in \mathbb{R}^K$ it holds that

$$[\mathbf{a}^T[\mathbf{x}] \bmod \Delta] \bmod \Delta = [\mathbf{a}^T\mathbf{x}] \bmod \Delta.$$

In particular: $\forall \mathbf{A} \in \mathbb{Z}^{K\times K} \ : \ \mathbf{A}\mathbf{x} \in \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K \Rightarrow [\mathbf{A}\mathbf{x}^*]^* = \mathbf{A}\mathbf{x}$

For any invertible $\mathbf{A} \in \mathbb{Z}^{K\times K}$ we can use the sub-optimal decode

$$\hat{\mathbf{x}}_{\mathsf{IF}}(\mathbf{x}^*) = \mathbf{A}^{-1}\left([\mathbf{A}\mathbf{x}^*]^*\right)$$

which is correct iff $\mathbf{A}\mathbf{x} \in \mathrm{CUBE} \triangleq \left[-\frac{\Delta}{2}, \frac{\Delta}{2}\right)^K$

Error probability: $\epsilon_{\mathsf{IF}} \triangleq \Pr\left(\hat{\mathbf{X}}_{\mathsf{IF}}(\mathbf{X}^*) \neq \mathbf{X}\right) = \Pr(\mathbf{A}\mathbf{X} \notin \mathrm{CUBE})$

Any invertible $\mathbf{A} \in \mathbb{Z}^{K\times K}$ works
$\Rightarrow$ choose $\mathbf{A} = \arg\min \Pr(\mathbf{A}\mathbf{X} \notin \mathrm{CUBE})$

Solving for known Σ - Suboptimal IF decoding contd.

Exact solution is hard, so instead we minimize an upper bound

$$\epsilon_{\mathrm{IF}} \leq 2K \cdot Q\left(\frac{\Delta/2}{\max_{k\in[K]}\sqrt{\mathbf{a}_k^T\Sigma\mathbf{a}_k}}\right)$$

To minimize ϵ_{IF} we will choose

$$\mathbf{A} = [\mathbf{a}_1|\cdots|\mathbf{a}_K]^T = \underset{\substack{\bar{\mathbf{A}}\in\mathbb{Z}^{K\times K}\\ |\bar{\mathbf{A}}|\neq 0}}{\arg\min}\ \max_{k\in[K]}\ \bar{\mathbf{a}}_k^T\Sigma\bar{\mathbf{a}}_k,$$

Solving for known Σ - Suboptimal IF decoding contd.

Exact solution is hard, so instead we minimize an upper bound

$$\epsilon_{\mathrm{IF}} \leq 2K \cdot Q\left(\frac{\Delta/2}{\max_{k\in[K]}\sqrt{\mathbf{a}_k^T\Sigma\mathbf{a}_k}}\right)$$

To minimize ϵ_{IF} we will choose

$$\mathbf{A} = [\mathbf{a}_1|\cdots|\mathbf{a}_K]^T = \underset{\substack{\bar{\mathbf{A}}\in\mathbb{Z}^{K\times K}\\ |\bar{\mathbf{A}}|\neq 0}}{\arg\min}\ \max_{k\in[K]}\ \bar{\mathbf{a}}_k^T\Sigma\bar{\mathbf{a}}_k,$$

Can be approximated efficiently by LLL

Solving for known Σ - Suboptimal IF decoding contd.

Exact solution is hard, so instead we minimize an upper bound

$$\epsilon_{\mathrm{IF}} \leq 2K \cdot Q\left(\frac{\Delta/2}{\max_{k\in[K]}\sqrt{\mathbf{a}_k^T\Sigma\mathbf{a}_k}}\right)$$

To minimize ϵ_{IF} we will choose

$$\mathbf{A} = [\mathbf{a}_1|\cdots|\mathbf{a}_K]^T = \underset{\substack{\bar{\mathbf{A}}\in\mathbb{Z}^{K\times K}\\ |\bar{\mathbf{A}}|\neq 0}}{\arg\min}\ \max_{k\in[K]}\ \bar{\mathbf{a}}_k^T\Sigma\bar{\mathbf{a}}_k,$$

Can be approximated efficiently by LLL

Performance is typically close to that of the optimal decoder [O.-Erez IT'17], [Domanovitz-Erez ITW'17]

Solving for known Σ - Suboptimal IF decoding contd.

Exact solution is hard, so instead we minimize an upper bound

$$\epsilon_{\text{IF}} \leq 2K \cdot Q\left(\frac{\Delta/2}{\max_{k\in[K]}\sqrt{\mathbf{a}_k^T\Sigma\mathbf{a}_k}}\right)$$

To minimize ϵ_{IF} we will choose

$$\mathbf{A} = [\mathbf{a}_1|\cdots|\mathbf{a}_K]^T = \underset{\substack{\bar{\mathbf{A}}\in\mathbb{Z}^{K\times K}\\ |\bar{\mathbf{A}}|\neq 0}}{\arg\min}\ \max_{k\in[K]}\ \bar{\mathbf{a}}_k^T\Sigma\bar{\mathbf{a}}_k,$$

Can be approximated efficiently by LLL

Performance is typically close to that of the optimal decoder [O.-Erez IT'17], [Domanovitz-Erez ITW'17]

In the blind setup Σ is not known

A can't be found via optimization

Blind Algorithm - Overview

Our algorithm starts with $\mathbf{A}^{(0)} = \mathbf{I}$ and iteratively updates the choice of $\mathbf{A}$ such that $\Pr(\mathbf{AX} \notin \mathrm{CUBE})$ decreases each iteration

Blind Algorithm - Overview

Our algorithm starts with $\mathbf{A}^{(0)} = \mathbf{I}$ and iteratively updates the choice of $\mathbf{A}$ such that $\Pr(\mathbf{AX} \notin \mathrm{CUBE})$ decreases each iteration

For the update, we estimate the covariance matrix $\check{\Sigma}$ of the RV $\mathbf{X}$ **truncated** to CUBE:

$$\mathbb{E}\left[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathrm{CUBE}\right] \approx \sum_{\mathbf{x}_i^* \,:\, \mathbf{x}_i \in \mathrm{CUBE}} \mathbf{x}_i^* \mathbf{x}_i^{*T}$$

(How do we know what samples to include in the sum?)

Blind Algorithm - Overview

Our algorithm starts with $\mathbf{A}^{(0)} = \mathbf{I}$ and iteratively updates the choice of $\mathbf{A}$ such that $\Pr(\mathbf{AX} \notin \mathrm{CUBE})$ decreases each iteration

For the update, we estimate the covariance matrix $\check{\Sigma}$ of the RV $\mathbf{X}$ **truncated** to CUBE:

$$\mathbb{E}\left[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathrm{CUBE}\right] \approx \sum_{\mathbf{x}_i^* \,:\, \mathbf{x}_i \in \mathrm{CUBE}} \mathbf{x}_i^* \mathbf{x}_i^{*T}$$

(How do we know what samples to include in the sum?)

We ignore the effect of truncation, and compute

$\tilde{\mathbf{A}} = \arg\min_{\bar{A}} \Pr(\bar{\mathbf{A}}\mathbf{Y} \notin \mathrm{CUBE})$ for $\mathbf{Y} \sim \mathcal{N}(\mathbf{0}, \check{\Sigma})$

Blind Algorithm - Overview

Our algorithm starts with $\mathbf{A}^{(0)} = \mathbf{I}$ and iteratively updates the choice of $\mathbf{A}$ such that $\Pr(\mathbf{AX} \notin \mathrm{CUBE})$ decreases each iteration

For the update, we estimate the covariance matrix $\check{\Sigma}$ of the RV $\mathbf{X}$ **truncated** to CUBE:

$$\mathbb{E}\left[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathrm{CUBE}\right] \approx \sum_{\mathbf{x}_i^* \,:\, \mathbf{x}_i \in \mathrm{CUBE}} \mathbf{x}_i^* \mathbf{x}_i^{*T}$$

(How do we know what samples to include in the sum?)

We ignore the effect of truncation, and compute

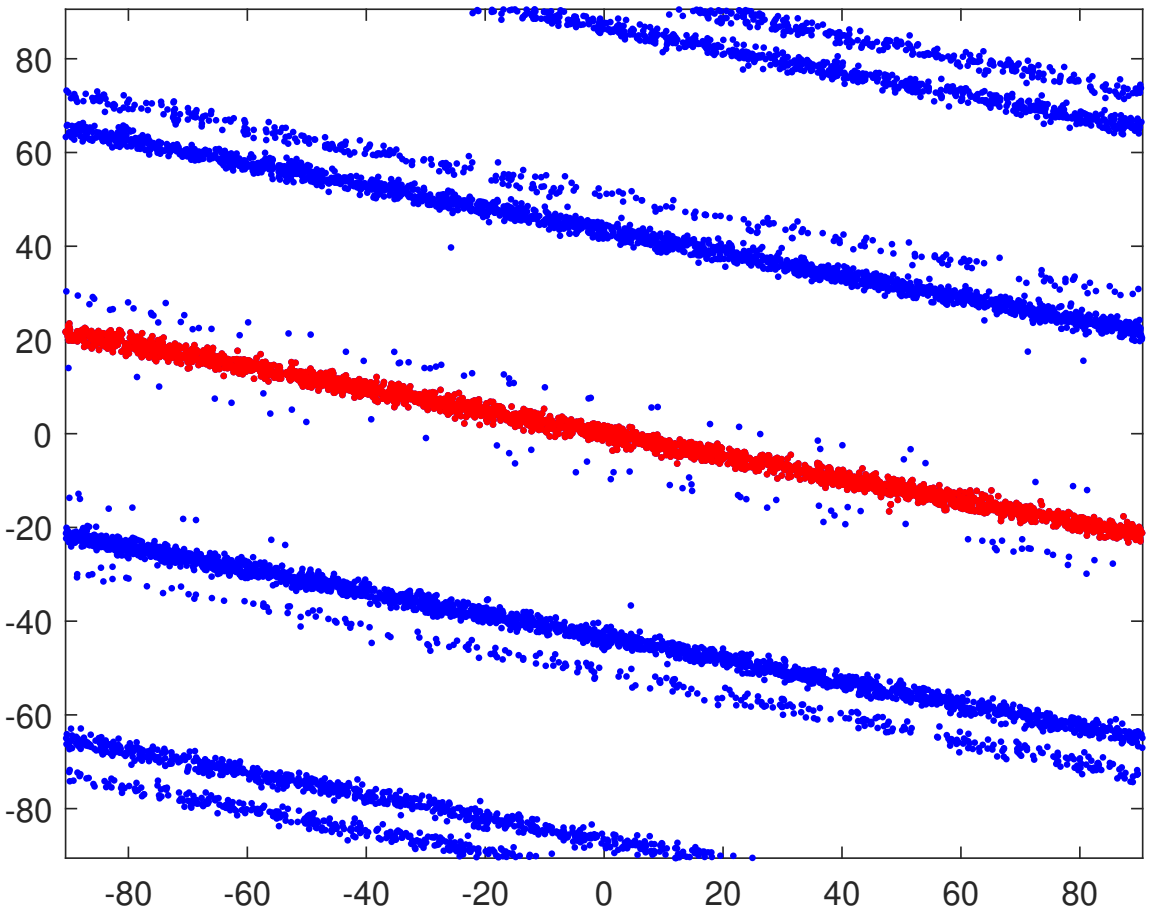
$\tilde{\mathbf{A}} = \arg\min_{\bar{A}} \Pr(\bar{\mathbf{A}}\mathbf{Y} \notin \mathrm{CUBE})$ for $\mathbf{Y} \sim \mathcal{N}(\mathbf{0}, \check{\Sigma})$

Then, we update $\mathbf{A}$ to $\tilde{\mathbf{A}}\mathbf{A}$ and $\mathbf{X}_i^*$ to $[\tilde{\mathbf{A}}\mathbf{X}_i^*]^* = [\tilde{\mathbf{A}}\mathbf{X}_i]^*$, for all i

Identifying Non-Folded Points

Let $\mathbf{S} = \{\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*\}$ and define

$$d_0(\mathbf{S}) = \min_{i,j \ s.t. \mathbf{X}_i = \mathbf{X}_i^*, \mathbf{X}_j \neq \mathbf{X}_j^*} \|\mathbf{X}_i^* - \mathbf{X}_j^*\|$$



Identifying Non-Folded Points

Let $\mathbf{S} = \{\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*\}$ and define

$$d_0(\mathbf{S}) = \min_{i,j \ s.t. \mathbf{X}_i = \mathbf{X}_i^*, \mathbf{X}_j \neq \mathbf{X}_j^*} \|\mathbf{X}_i^* - \mathbf{X}_j^*\|$$

Theorem (informal)

Assume $\min_{\mathbf{A}} \Pr(\mathbf{AX} \notin \mathrm{CUBE}) \leq \epsilon \ll 1$ and that $\lambda_{\min}(\Sigma) \geq \tau_{\min}^2$, then for any $0 < \eta < 1$ we have that

$$\Pr\left(d_0(\mathbf{S}) \leq 2\eta\tau_{\min}\sqrt{2\log\frac{1}{\epsilon}}\right) \leq n\epsilon^{(1-\eta)^2}.$$

Identifying Non-Folded Points

Algorithm for identifying non-folded point:

Inputs: $\mathbf{S} = \{\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*\}$, and a parameter $d > 0$

Algorithm:

1. Add a point $\mathbf{X}_0^* = \mathbf{0}$ to $\mathbf{S}$
2. Construct a graph where each of the $n+1$ points is a vertex, and an edge between $\mathbf{X}_i^*$ and $\mathbf{X}_j^*$ exists iff $\|\mathbf{X}_i^* - \mathbf{X}_j^*\| < d$

Output: the connected component of $\mathbf{X}_0^* = \mathbf{0}$

Identifying Non-Folded Points

Algorithm for identifying non-folded point:

Inputs: $\mathbf{S} = \{\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*\}$, and a parameter $d > 0$

Algorithm:

1. Add a point $\mathbf{X}_0^* = \mathbf{0}$ to $\mathbf{S}$
2. Construct a graph where each of the $n+1$ points is a vertex, and an edge between $\mathbf{X}_i^*$ and $\mathbf{X}_j^*$ exists iff $\|\mathbf{X}_i^* - \mathbf{X}_j^*\| < d$

Output: the connected component of $\mathbf{X}_0^* = \mathbf{0}$

If $d_0(\mathbf{S}) > d$, there are no false alarms
Probability of mis-detection depends on density of points

Unknown Σ - Main Algorithm

Inputs: $\mathbf{X}_1^*, \ldots, \mathbf{X}_n^*$

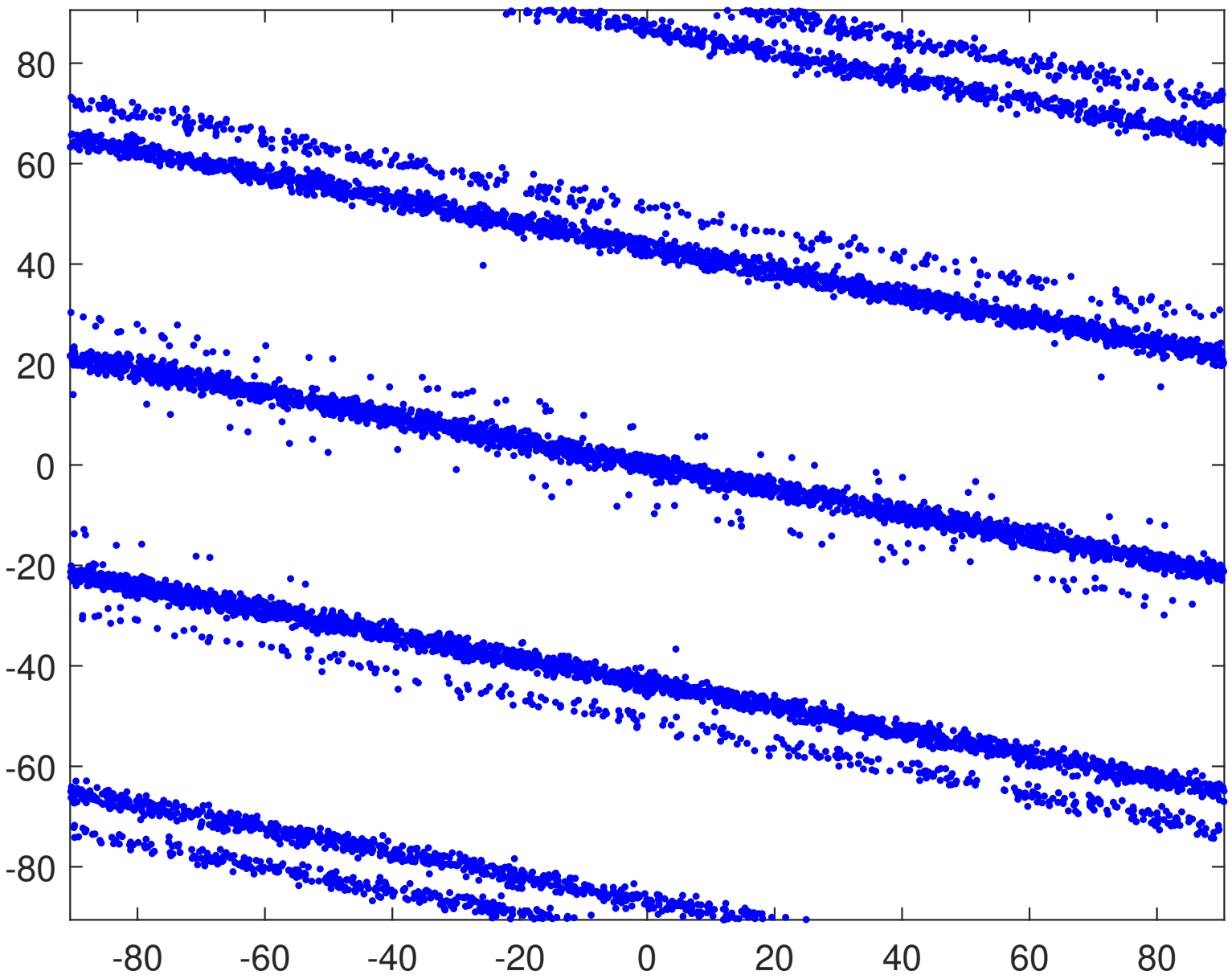
Initialization: $\mathbf{A} = \mathbf{I}$ and $\tilde{\mathbf{A}} = \mathbf{0}$

Algorithm: While $\tilde{\mathbf{A}} \neq \mathbf{I}$

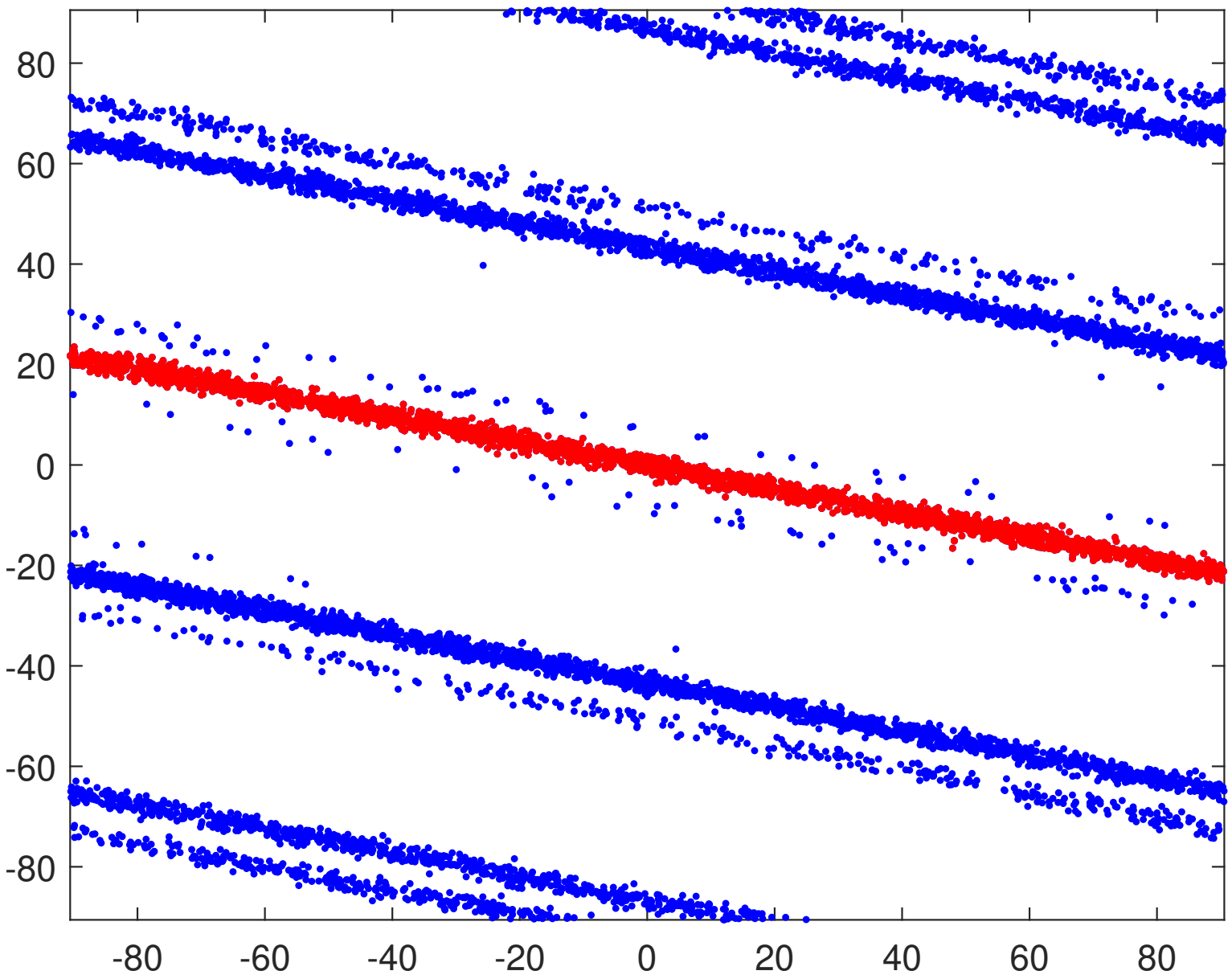
1. For each $\mathbf{X}_i^*$ decide whether or not $\mathbf{X}_i \in \mathrm{CUBE}$
2. Compute the empirical covariance matrix $\check{\Sigma}$ of the vectors that were estimated to be in CUBE
3. Set
$$\tilde{\mathbf{A}} = [\tilde{\mathbf{a}}_1 | \cdots | \tilde{\mathbf{a}}_K]^T = \underset{\mathbf{A} \in \mathrm{GL}_K(\mathbb{Z})}{\arg\min} \max_{k=1,\ldots,K} \bar{\mathbf{a}}_k^T \check{\Sigma} \bar{\mathbf{a}}_k,$$
4. Set $\mathbf{X}_i^* \leftarrow [\tilde{\mathbf{A}}\mathbf{X}_i^*]^* = [\tilde{\mathbf{A}}\mathbf{X}_i]^*$ for $i = 1, \ldots, n$, and $\mathbf{A} \leftarrow \tilde{\mathbf{A}} \cdot \mathbf{A}$

Output: $\hat{\mathbf{X}}_i = \mathbf{A}^{-1}\mathbf{X}_i^*$

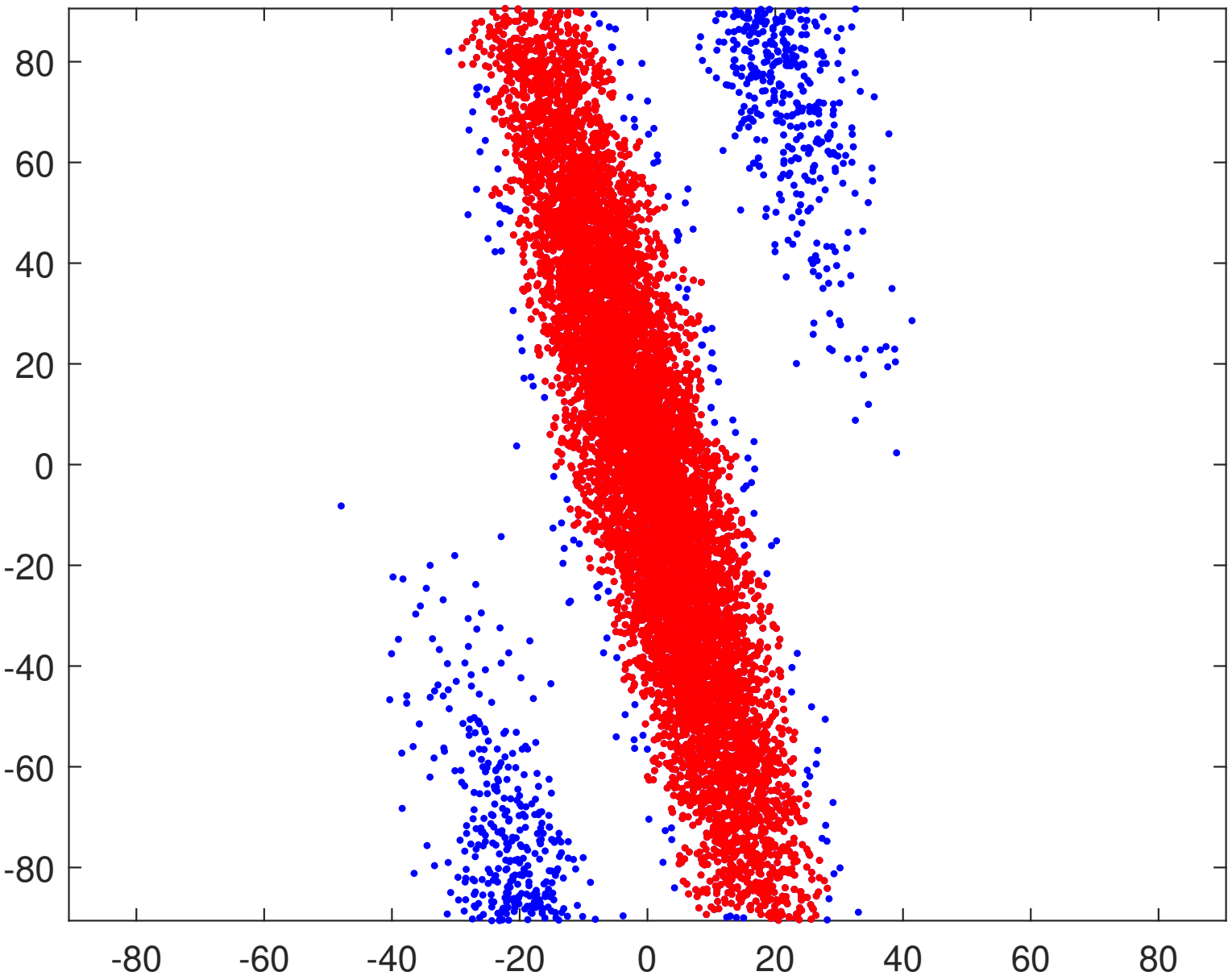
Unknown Σ - Main Algorithm



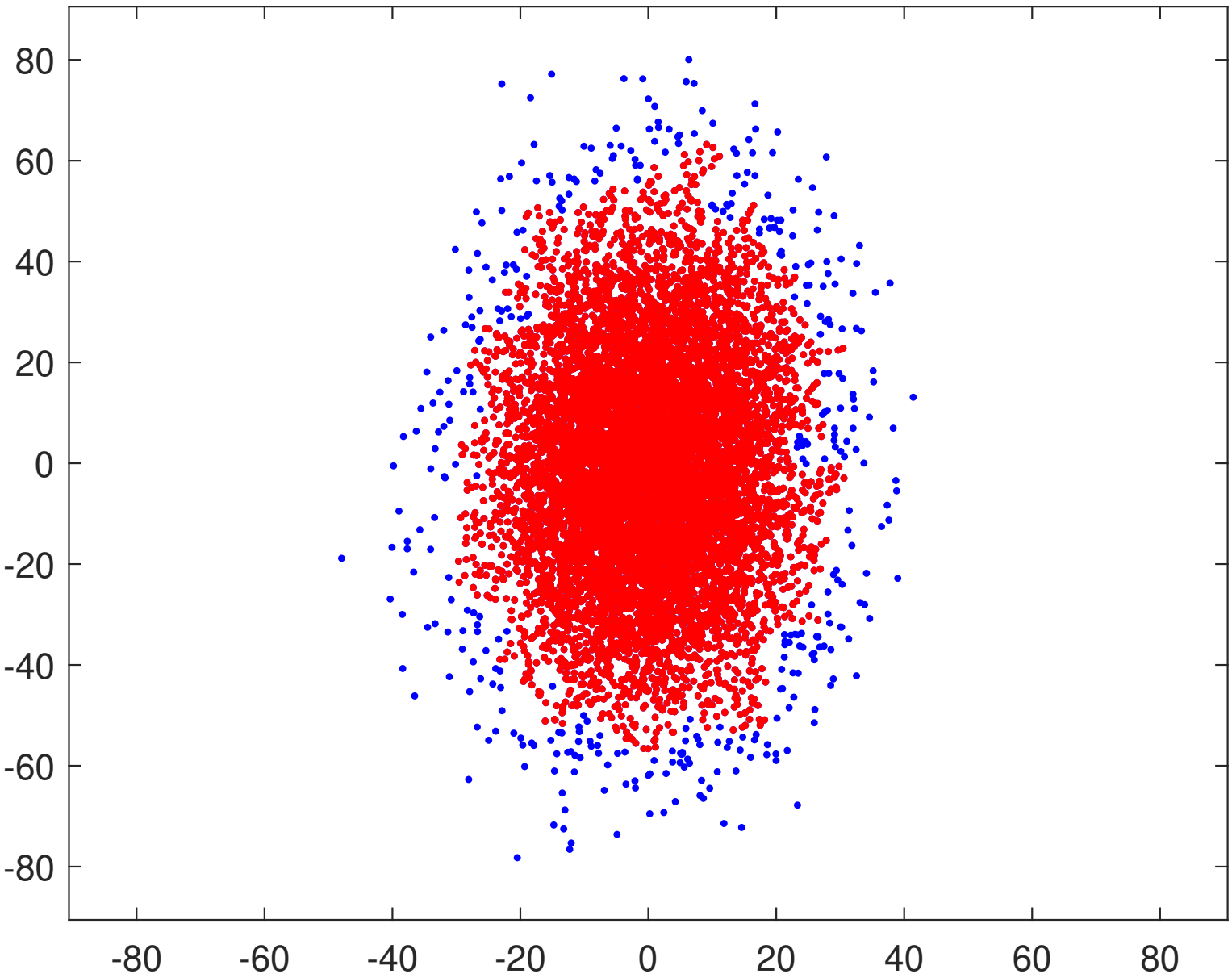
Unknown Σ - Main Algorithm



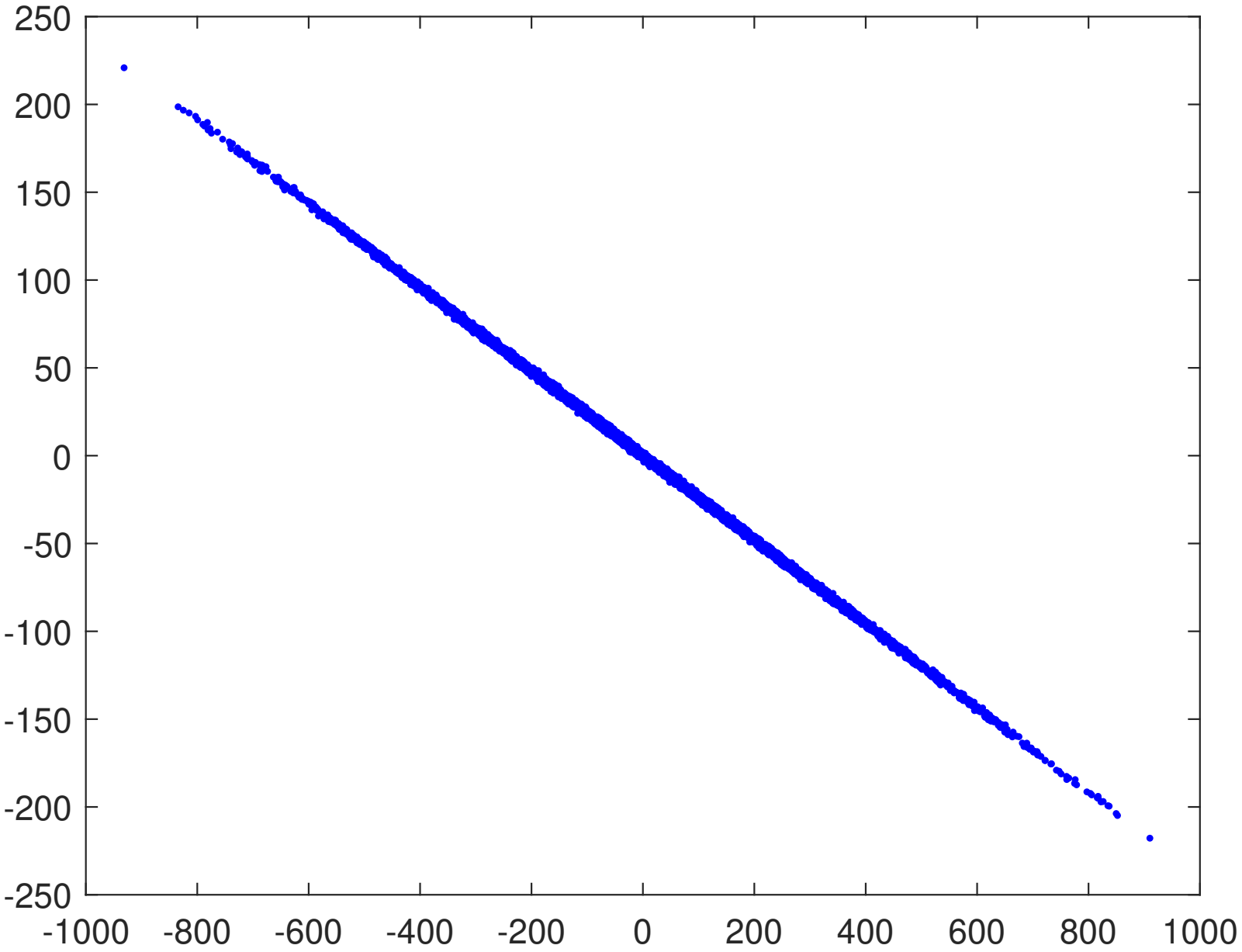
Unknown Σ - Main Algorithm



Unknown Σ - Main Algorithm



Unknown Σ - Main Algorithm



Analysis

Thm: Genie-Aided Algorithm Finds Almost Optimal $\mathbf{A}$

Assume $\min_{\mathbf{A}\in \mathrm{GL}_K(\mathbb{Z})} \Pr(\mathbf{AX} \notin \mathrm{CUBE}) \leq \epsilon \ll 1$ and let $P = \Pr(\mathbf{X} \notin \mathrm{CUBE})$. If the truncated covariance estimation at each iteration is **perfect**, then after at most $\frac{6}{\log \frac{4}{3+P}} + 2$ iterations of the main algorithm we have that

$$\Pr(\mathbf{AX} \notin \mathrm{CUBE}) \leq \epsilon^{0.98}$$

Analysis

Thm: Genie-Aided Algorithm Finds Almost Optimal $\mathbf{A}$

Assume $\min_{\mathbf{A}\in \mathrm{GL}_K(\mathbb{Z})} \Pr(\mathbf{AX} \notin \mathrm{CUBE}) \leq \epsilon \ll 1$ and let $P = \Pr(\mathbf{X} \notin \mathrm{CUBE})$. If the truncated covariance estimation at each iteration is **perfect**, then after at most $\frac{6}{\log \frac{4}{3+P}} + 2$ iterations of the main algorithm we have that

$$\Pr(\mathbf{AX} \notin \mathrm{CUBE}) \leq \epsilon^{0.98}$$

Thm: Blind Algorithm Asymptotically As Good As Informed

Under the same assumptions, if the main algorithm is run with $d = f(\epsilon, K, P)$, and $n = \epsilon^{-\delta}$ for some fixed δ, then after at most $\frac{6}{\log \frac{4}{3+P}} + 2$ iterations

$$\Pr\left(\{\hat{\mathbf{X}}_1, \ldots, \hat{\mathbf{X}}_n\} \neq \{\mathbf{X}_1, \ldots, \mathbf{X}_n\}\right) \leq n\epsilon^{0.96}.$$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

Proof based on Gaussian Correlation Inequality

We also prove a lower bound of the form $\psi(\Pr(\mathbf{X} \notin \mathcal{S}), K)\Sigma$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

Let $\mathbf{Y} = [\mathbf{X} | \mathbf{X} \in \mathrm{CUBE}]$. Since IF succeeds w.p. $\geq 1 - \epsilon$, there are linearly independent integer vectors $\mathbf{a}_1, \ldots, \mathbf{a}_K$ for which

$$\sqrt{\mathrm{Var}(\mathbf{a}_k^T \mathbf{Y})} = \sqrt{\mathbf{a}_k^T \overset{\vee}{\Sigma} \mathbf{a}_k} \leq \sqrt{\mathbf{a}_k^T \Sigma \mathbf{a}_k} \leq \frac{1}{Q^{-1}\left(\frac{\epsilon}{2}\right)} \cdot \frac{\Delta}{2} \triangleq \eta \cdot \frac{\Delta}{2}$$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

Let $\mathbf{Y} = [\mathbf{X} | \mathbf{X} \in \mathrm{CUBE}]$. Since IF succeeds w.p. $\geq 1 - \epsilon$, there are linearly independent integer vectors $\mathbf{a}_1, \ldots, \mathbf{a}_K$ for which

$$\sqrt{\mathrm{Var}(\mathbf{a}_k^T \mathbf{Y})} = \sqrt{\mathbf{a}_k^T \overset{\vee}{\Sigma} \mathbf{a}_k} \leq \sqrt{\mathbf{a}_k^T \Sigma \mathbf{a}_k} \leq \frac{1}{Q^{-1}\left(\frac{\epsilon}{2}\right)} \cdot \frac{\Delta}{2} \triangleq \eta \cdot \frac{\Delta}{2}$$

The algorithm chooses

$$\tilde{\mathbf{A}} = [\tilde{\mathbf{a}}_1 | \cdots | \tilde{\mathbf{a}}_K]^T = \arg\min_{\mathbf{A}} \max_{k=1,\ldots,K} \bar{\mathbf{a}}_k^T \overset{\vee}{\Sigma} \bar{\mathbf{a}}_k,$$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

Let $\mathbf{Y} = [\mathbf{X} | \mathbf{X} \in \mathrm{CUBE}]$. Since IF succeeds w.p. $\geq 1 - \epsilon$, there are linearly independent integer vectors $\mathbf{a}_1, \ldots, \mathbf{a}_K$ for which

$$\sqrt{\mathrm{Var}(\mathbf{a}_k^T \mathbf{Y})} = \sqrt{\mathbf{a}_k^T \overset{\vee}{\Sigma} \mathbf{a}_k} \leq \sqrt{\mathbf{a}_k^T \Sigma \mathbf{a}_k} \leq \frac{1}{Q^{-1}\left(\frac{\epsilon}{2}\right)} \cdot \frac{\Delta}{2} \triangleq \eta \cdot \frac{\Delta}{2}$$

So we must have that

$$\max_k \sqrt{\mathrm{Var}(\tilde{\mathbf{a}}_k^T \mathbf{Y})} \leq \eta \cdot \frac{\Delta}{2}$$

Why Does the Genie-Aided Algorithm Work?

Thm: Truncated vs. Non-Truncated Covariance

Let $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and $\mathcal{S} \subset \mathbb{R}^K$ be a convex symmetric set. Then

$$\mathbb{E}[\mathbf{X}\mathbf{X}^T | \mathbf{X} \in \mathcal{S}] \preceq \Sigma$$

So we must have that

$$\max_k \sqrt{\mathrm{Var}(\tilde{\mathbf{a}}_k^T \mathbf{Y})} \leq \eta \cdot \frac{\Delta}{2}$$

By Chebishev

$$\Pr\left(|\tilde{\mathbf{a}}_k^T \mathbf{Y}| \geq \frac{1}{2} \cdot \frac{\Delta}{2}\right) \leq 4\eta^2$$

$$\Pr(\tilde{\mathbf{A}}\mathbf{Y} \in \frac{1}{2}\mathrm{CUBE}) \geq 1 - \underbrace{4K\eta^2}_{\delta}$$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\begin{aligned}
\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) &= \Pr\left(\tilde{\mathbf{A}}\mathbf{X} \in \frac{1}{2}\mathrm{CUBE}\right) \\
&\geq \Pr\left(\tilde{\mathbf{A}}\mathbf{X} \in \frac{1}{2}\mathrm{CUBE} \middle| \mathbf{X} \in \mathrm{CUBE}\right) \Pr\left(\mathbf{X} \in \mathrm{CUBE}\right) \\
&= \Pr\left(\tilde{\mathbf{A}}\mathbf{Y} \in \frac{1}{2}\mathrm{CUBE}\right) \Pr\left(\mathbf{X} \in \mathrm{CUBE}\right) \\
&\geq (1-\delta) \Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).
\end{aligned}$$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

What does this say about $\Pr\left(\tilde{\mathbf{X}} \in \mathrm{CUBE}\right)$?

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

Theorem [Latala-Oleszkiewicz'99]

For $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and convex symmetric $\mathcal{S} \subset \mathbb{R}^K$ it holds that

$$\Pr\left(\mathbf{X} \in \mathcal{S}\right) \geq 1 - 2Q\left(2Q^{-1}\left(\frac{1-\Pr\left(\mathbf{X} \in \frac{1}{2}\mathcal{S}\right)}{2}\right)\right)$$
$$\triangleq F\left(\Pr\left(\mathbf{X} \in \frac{1}{2}\mathcal{S}\right)\right)$$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

$$\Pr(\tilde{\mathbf{X}} \in \mathrm{CUBE}) \geq F\left((1-\delta)\Pr(\mathbf{X} \in \mathrm{CUBE})\right)$$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

$$\Pr(\tilde{\mathbf{X}} \in \mathrm{CUBE}) \geq F\left((1-\delta)\Pr(\mathbf{X} \in \mathrm{CUBE})\right)$$

Let $p_i = \Pr(\mathbf{AX} \in \mathrm{CUBE})$ after i iterations

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

$$\Pr(\tilde{\mathbf{X}} \in \mathrm{CUBE}) \geq F\left((1-\delta)\Pr(\mathbf{X} \in \mathrm{CUBE})\right)$$

Let $p_i = \Pr(\mathbf{AX} \in \mathrm{CUBE})$ after i iterations

We have obtained $p_i \geq F\left((1-\delta)p_{i-1}\right)$

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

$$\Pr(\tilde{\mathbf{X}} \in \mathrm{CUBE}) \geq F\left((1-\delta)\Pr(\mathbf{X} \in \mathrm{CUBE})\right)$$

Let $p_i = \Pr(\mathbf{AX} \in \mathrm{CUBE})$ after i iterations

We have obtained $p_i \geq F\left((1-\delta)p_{i-1}\right)$

Converges to some fixed point p^* close to 1

Why Does the Genie-Aided Algorithm Work?

Let $\tilde{\mathbf{X}} \triangleq \tilde{\mathbf{A}}\mathbf{X}$. We have

$$\Pr\left(\tilde{\mathbf{X}} \in \frac{1}{2}\mathrm{CUBE}\right) \geq (1-\delta)\Pr\left(\mathbf{X} \in \mathrm{CUBE}\right).$$

$$\Pr(\tilde{\mathbf{X}} \in \mathrm{CUBE}) \geq F\left((1-\delta)\Pr(\mathbf{X} \in \mathrm{CUBE})\right)$$

Let $p_i = \Pr(\mathbf{AX} \in \mathrm{CUBE})$ after i iterations

We have obtained $p_i \geq F\left((1-\delta)p_{i-1}\right)$

Converges to some fixed point p^* close to 1

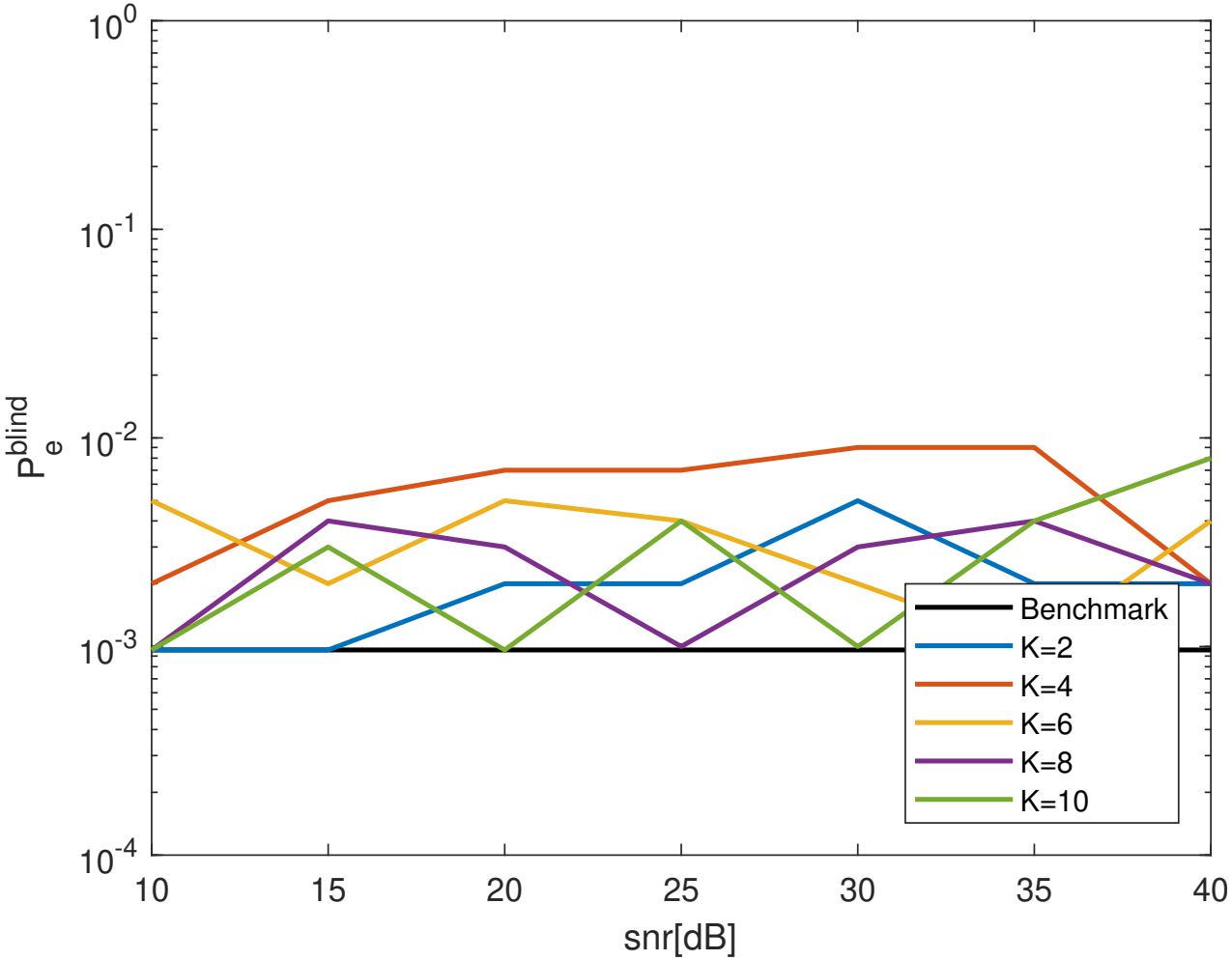
At this point $\mathbb{E}[\mathbf{AXX}^T\mathbf{A}^T|\mathbf{AX} \in \mathrm{CUBE}] \approx \mathbf{A}\Sigma\mathbf{A}^T$
the next iteration will find a "good" $\mathbf{A}$

Numerical Results

$$\Sigma = \mathrm{snr}\mathbf{H}\mathbf{H}^T + \mathbf{I},\ \mathbf{H} \in \mathbb{R}^{K \times \mathrm{rank}},\ H_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1),\ n = 1000$$

Numerical Results

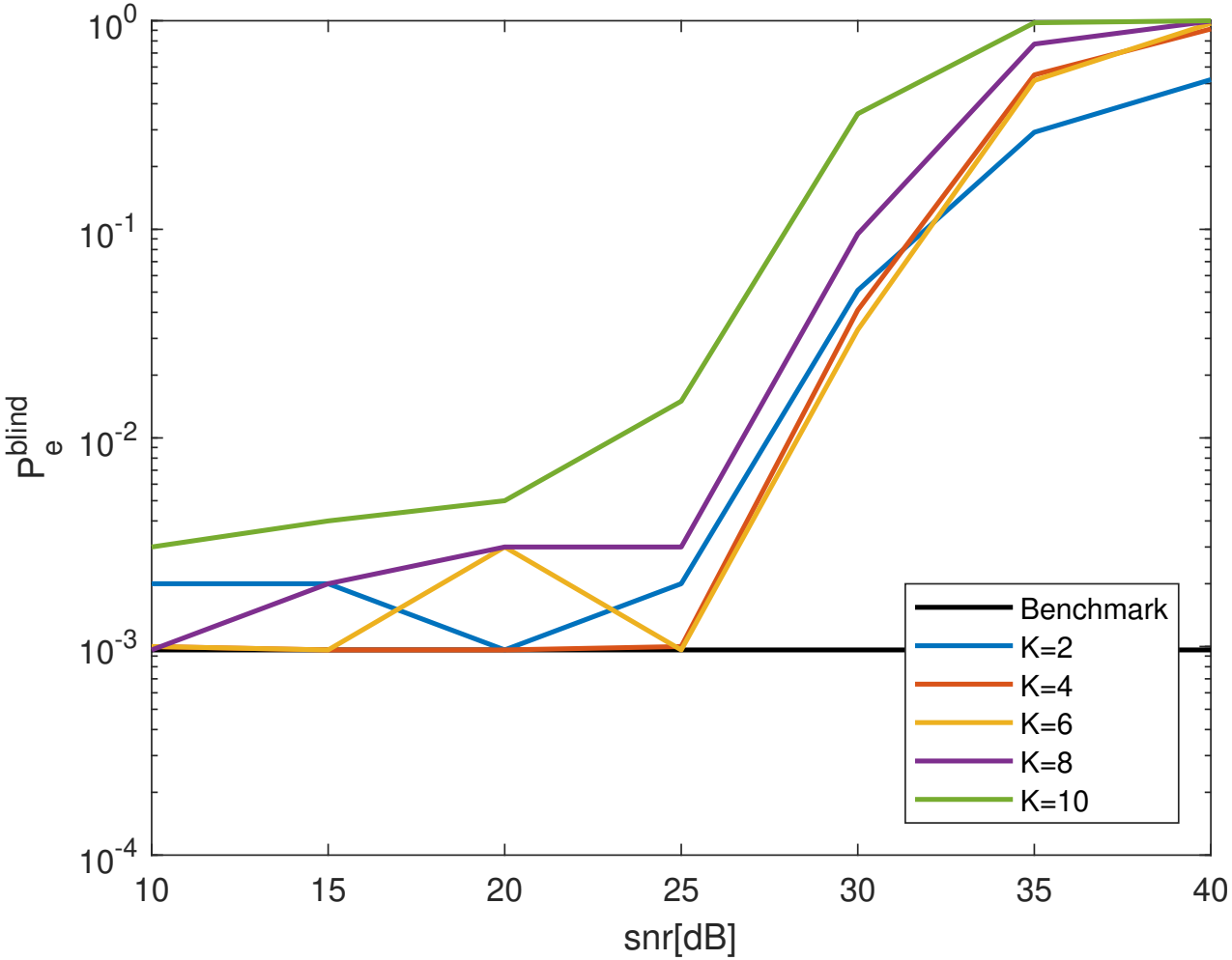
$$\Sigma = \mathrm{snr}\mathbf{H}\mathbf{H}^T + \mathbf{I},\ \mathbf{H} \in \mathbb{R}^{K\times \mathrm{rank}},\ H_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1),\ n = 1000$$



$$\mathrm{rank} = 1$$

Numerical Results

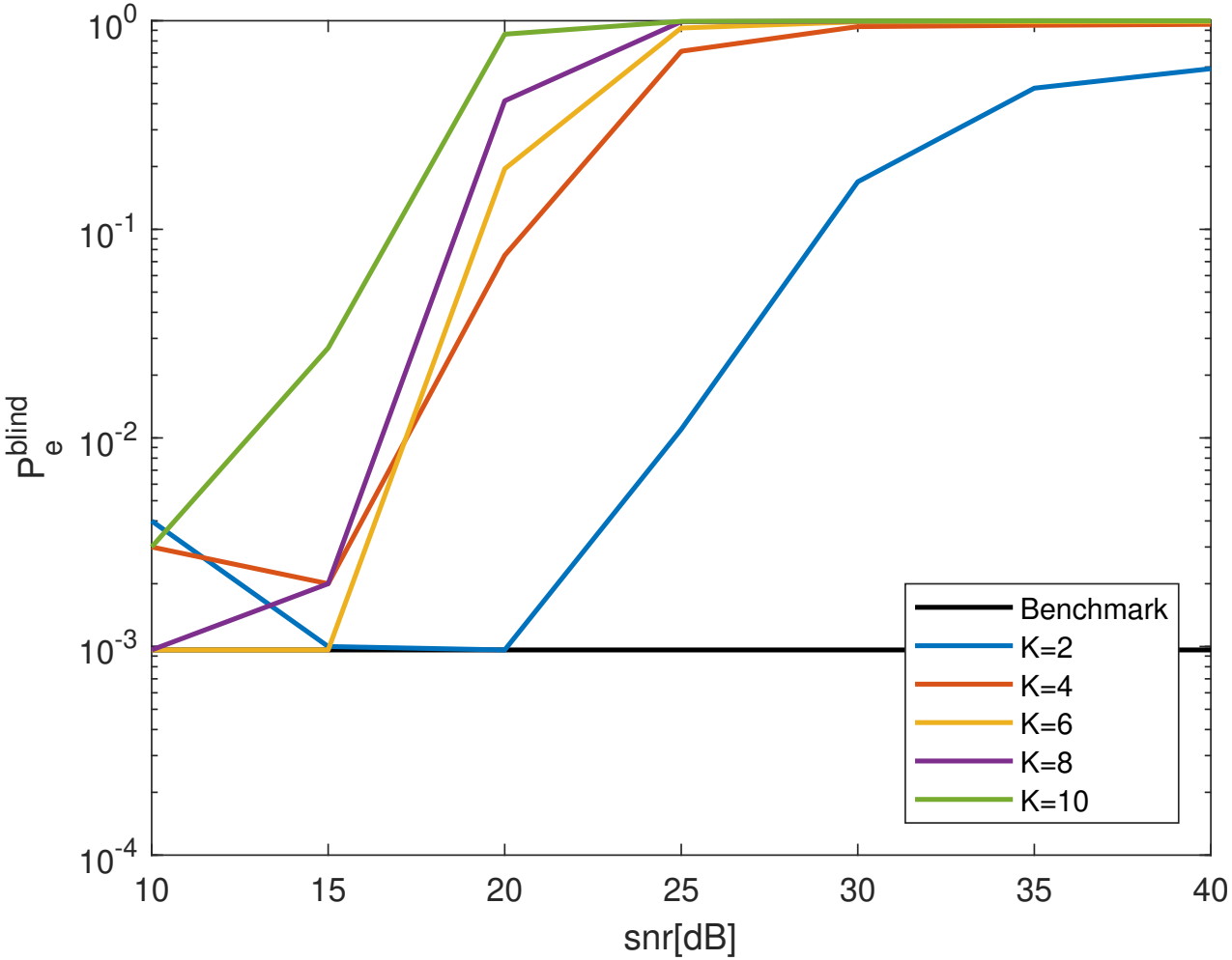
$$\Sigma = \mathrm{snr}\mathbf{H}\mathbf{H}^T + \mathbf{I},\ \mathbf{H} \in \mathbb{R}^{K\times \mathrm{rank}},\ H_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1),\ n = 1000$$



$$\mathrm{rank} = 2$$

Numerical Results

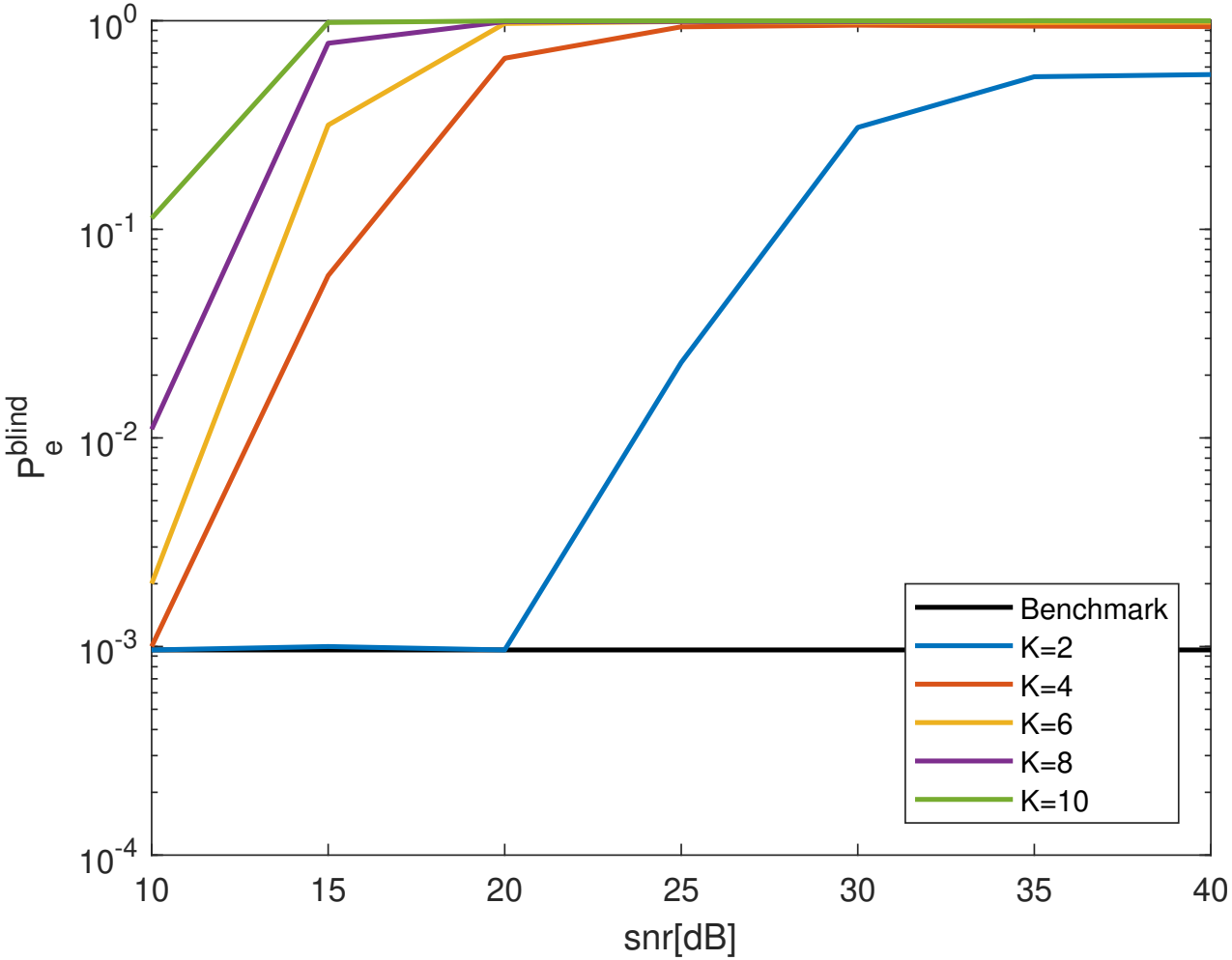
$$\Sigma = \mathrm{snr}\mathbf{H}\mathbf{H}^T + \mathbf{I},\ \mathbf{H} \in \mathbb{R}^{K\times \mathrm{rank}},\ H_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1),\ n = 1000$$



$$\mathrm{rank} = 3$$

Numerical Results

$$\Sigma = \mathrm{snr}\mathbf{H}\mathbf{H}^T + \mathbf{I},\ \mathbf{H} \in \mathbb{R}^{K\times \mathrm{rank}},\ H_{ij} \overset{i.i.d.}{\sim} \mathcal{N}(0,1),\ n = 1000$$



$$\mathrm{rank} = 4$$

Summary

- Modulo reduction of correlated variables is an effective technique for quantization/ADC
- When the correlation structure is known, efficient algorithms for unwrapping are available
- We developed a novel low-complexity algorithm for blind modulo unwrapping, with provable performance guarantees
- Algorithm iterates between two steps:
  - Extracting measurements of truncated Gaussians from the wrapped measurements
  - Balancing variances of components based on the correlations of the truncated vector
- Work can be found on arxiv