# The Power-Oja Method for Decentralized Subspace Estimation/Tracking

**Sissi Xiaoxiao Wu**[†], Hoi-To Wai[†], Anna Scaglione[†], Neil A. Jacklin[⋆]

[†]Arizona State University

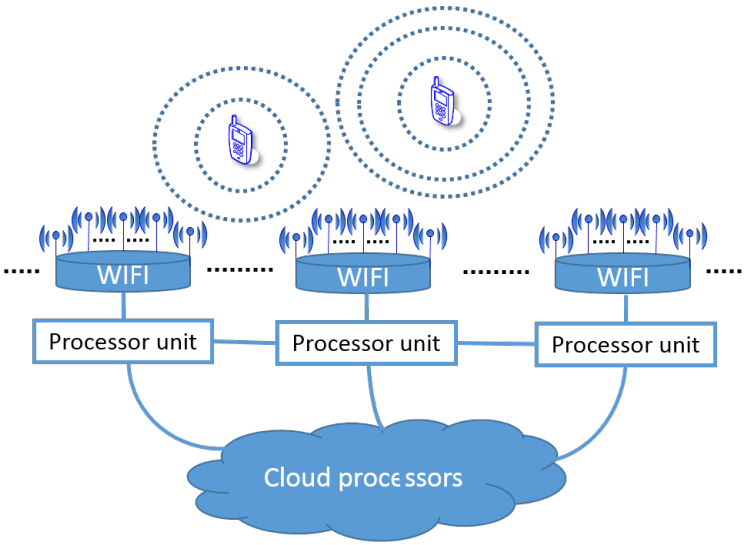
[⋆] Northrop Grumman Mission Systems


March 1, 2017

# Subspace Estimation

## New Challenges

- Tracking and estimation accuracy.
- Distributed system: separate antennas.
- Large scale: massive array.
- Complexity: distribute computations to local processors.



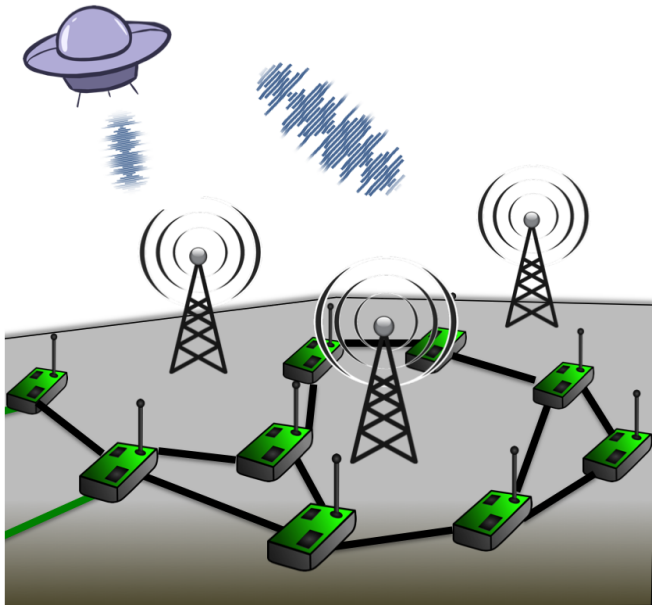



Figure: Two examples: WiFi networks and Radar networks.

# Literature Survey

- The power method [Gv96]:
  - A batch processing method with fast convergence.
  - Non-adaptive, high latency
  - Guarantee for a rank-$p$ subspace.
  - Computations can be decentralized [LSM11].
- The Oja's method [OK85]
  - A stochastic gradient decent (SGD) method adaptive for tracking varying statistics.
  - First order method: suffer from slow convergence.
  - unconstrained SGD, no guarantee for a rank-$p$ subspace.
  - Computations can be decentralized [SPK08, SPZ16].
- The key of the decentralization is average consensus [LSM11, SPK08, SPZ16, BDF13]
  - Data are often measured distributively over large networks.
  - Gossip-based consensus algorithms solve multi-agent coordination and optimization problems in a decentralized manner.
  - Their key features are
    - ✓ built-in fault tolerance to intermittent computation/communication.
    - ✓ self reorganization to automatic failure correction.

# Problem Statement

- We consider a non-stationary stochastic process $\boldsymbol{r}(t) \in \mathcal{C}^N$ and let $\mathcal{T} \subset \{1, 2, ...\}$ be a sampling set. Define the sampled covariance:

$$\hat{\boldsymbol{R}}(\mathcal{T}) := |\mathcal{T}|^{-1} \sum_{s \in \mathcal{T}} \boldsymbol{r}(s)\boldsymbol{r}^H(s) \; . \tag{1}$$

- We track top $p$-D subspace by tackling the non-convex, stochastic optimization:

$$\min_{\boldsymbol{U} \in \mathcal{C}^{N \times p}} f_t(\boldsymbol{U}) := \mathbb{E}\left[\|\boldsymbol{r}(t) - \boldsymbol{U}\boldsymbol{U}^H\boldsymbol{r}(t)\|^2\right], \; \forall \; t \geq 1 \; . \tag{2}$$

- We follow the stochastic approximation to the objective function $f(\boldsymbol{U})$:

$$\hat{f}(\boldsymbol{U}; \mathcal{T}_\tau) := \mathrm{Tr}\Big(\Big(\boldsymbol{U}\boldsymbol{U}^H\boldsymbol{U}\boldsymbol{U}^H - 2\boldsymbol{U}\boldsymbol{U}^H\Big)\hat{\boldsymbol{R}}(\mathcal{T}_\tau)\Big) \; , \tag{3}$$

  where $\mathcal{T}_\tau \subset \{1, 2, ...\}$ is the set of observations made during the $\tau$th batch.

  - If $\boldsymbol{r}(t)$ is stationary for all $t \in \mathcal{T}_\tau$, then $\mathbb{E}[\hat{f}(\boldsymbol{U}; \mathcal{T}_\tau)] = f_t(\boldsymbol{U})$.
  - When $|\mathcal{T}_\tau|$ is large, $\hat{f}(\boldsymbol{U}; \mathcal{T}_\tau)$ is a good approximation for $f_t(\boldsymbol{U})$.
  - No unitary constraint on the subspace $\boldsymbol{U}$, no guarantee for a rank-$p$ subspace.

# Review the Power Method (PM)

The PM works with a whole batch of samples in $\mathcal{T}_\tau$.

- **Step 1:** Generate a random vector as an initial point $\tilde{\boldsymbol{u}}^k(1,\tau)$
- **Step 2:** For $k = 1, ..., p$

$$\tilde{\boldsymbol{u}}^k(\ell+1,\tau) = \hat{\boldsymbol{R}}(\mathcal{T}_\tau)\ \tilde{\boldsymbol{u}}^k(\ell,\tau) - \sum_{j=1}^{k-1} (\hat{\boldsymbol{u}}^j(\tau))^H \left(\hat{\boldsymbol{R}}(\mathcal{T}_\tau)\ \tilde{\boldsymbol{u}}^k(\ell,\tau)\right) \hat{\boldsymbol{u}}^j(\tau), \forall \ell = 1, ..., L$$

$$\hat{\boldsymbol{u}}^k(\tau) := \tilde{\boldsymbol{u}}^k(L,\tau)/\|\tilde{\boldsymbol{u}}^k(L,\tau)\| \ .$$

- **Step 3:** Output the top-$p$ subspace: $\hat{\boldsymbol{U}}_{PM}(\tau) := [\hat{\boldsymbol{u}}^1(\tau)\ \hat{\boldsymbol{u}}^2(\tau)\ \ldots\ \hat{\boldsymbol{u}}^p(\tau)]$.

We use $\hat{\boldsymbol{U}}_0$ to initialize the subspace and denote the above power process by

$$\overline{\boldsymbol{U}}_{PM}(\tau) = \mathrm{PM}\big(\{\boldsymbol{r}(s)\}_{s\in\mathcal{T}_\tau}; \hat{\boldsymbol{U}}_0; L\big)\ , \tag{4}$$

# Review the Oja's Learning Rule

The Oja's learning rule works with one sample of $\boldsymbol{r}(t)$ at a time.

- Let $\hat{\boldsymbol{U}}_{Oja}(t) \in \mathcal{C}^{N \times p}$ be an estimate of $\boldsymbol{U}(t)$ at iteration $t$, we perform the updates:

$$\hat{\boldsymbol{U}}_{Oja}(t+1) = \hat{\boldsymbol{U}}_{Oja}(t) - \gamma_t \nabla \hat{f}(\hat{\boldsymbol{U}}_{Oja}(t); \{t\}) \,, \tag{5}$$

  with $\nabla \hat{f}(\hat{\boldsymbol{U}}(t), \{t\}) = -2\boldsymbol{r}(t)\boldsymbol{r}^H(t)\hat{\boldsymbol{U}}(t) + \boldsymbol{r}(t)\boldsymbol{r}^H(t)\hat{\boldsymbol{U}}(t)\hat{\boldsymbol{U}}^H(t)\hat{\boldsymbol{U}}(t) + \hat{\boldsymbol{U}}(t)\hat{\boldsymbol{U}}^H(t)\boldsymbol{r}(t)\boldsymbol{r}^H(t)\hat{\boldsymbol{U}}(t)$ .

- Convergence for stationary $\boldsymbol{r}(t)$:
  - When $p = 1$ and $\gamma_t = c/t$, at a sub-linear rate of $\mathcal{O}(1/t)$ [BDF13];
  - If $\sum_t \gamma_t = \infty, \sum_t \gamma_t^2 < \infty$, converges almost surely to the principal $p$-dimensional subspace, yet the convergence rate is not given.

- In practice, the Oja's learning rule is often used for non-stationary $\boldsymbol{r}(t)$ with $\gamma_t$.

# Motivations for the Power-Oja (P-Oja) Method

Observations for PM and Oja:

- For PM, when $\boldsymbol{r}(t)$ is non-stationary and $|\mathcal{T}_\tau| \ll \infty \rightarrow$ a poor approximation for $\hat{\boldsymbol{R}}(\mathcal{T}_\tau)$ to the true covariance $\rightarrow$ degraded performance.
- For Oja, the spectral gap,

$$\sigma_p(\hat{\boldsymbol{R}}(\mathcal{T})) - \sigma_{p+1}(\hat{\boldsymbol{R}}(\mathcal{T}))$$

  is an important factor in determining the convergence speed [BDF13].

Our motivations:

- We want both advantages of the two methods:tracking and estimation accuracy.
- Try to increase the spectral gap.

# How to Increase the Spectral Gap

Our approach:

- Modify the stochastic approximation of the objective function:

$$\hat{f}_{\text{POja}}(\boldsymbol{U};\mathcal{T}) = \text{Tr}\Big(\Big(\boldsymbol{U}\boldsymbol{U}^H\boldsymbol{U}\boldsymbol{U}^H - 2\boldsymbol{U}\boldsymbol{U}^H\Big)(\hat{\boldsymbol{R}}(\mathcal{T}))^L\Big)\,. \tag{6}$$

  Apparently, $(\hat{\boldsymbol{R}}(\mathcal{T}))^L$ has a better *spectral gap* than $\hat{\boldsymbol{R}}(\mathcal{T})$, i.e.,

$$\sigma_p((\hat{\boldsymbol{R}}(\mathcal{T}))^L) - \sigma_{p+1}((\hat{\boldsymbol{R}}(\mathcal{T}))^L) > \sigma_p(\hat{\boldsymbol{R}}(\mathcal{T})) - \sigma_{p+1}(\hat{\boldsymbol{R}}(\mathcal{T})).$$

- P-Oja tracks the subspace in a batch by batch manner:
  - For the $\tau$th batch, we have

$$\nabla\hat{f}_{\text{POja}}(\hat{\boldsymbol{U}}_{\text{POja}}(\tau);\mathcal{T}_\tau) = -2(\hat{\boldsymbol{R}}(\mathcal{T}_\tau))^L\hat{\boldsymbol{U}}_{\text{POja}}(\tau) + (\hat{\boldsymbol{R}}(\mathcal{T}_\tau))^L\hat{\boldsymbol{U}}_{\text{POja}}(\tau)\hat{\boldsymbol{U}}_{\text{POja}}^H(\tau)\hat{\boldsymbol{U}}_{\text{POja}}(\tau) + \hat{\boldsymbol{U}}_{\text{POja}}(\tau)\hat{\boldsymbol{U}}_{\text{POja}}^H(\tau)(\hat{\boldsymbol{R}}(\mathcal{T}_\tau))^L\hat{\boldsymbol{U}}_{\text{POja}}(\tau)\,.$$

  - $(\hat{\boldsymbol{R}}(\mathcal{T}_\tau))^L\hat{\boldsymbol{U}}_{\text{POja}}(\tau)$: performing $L$ rounds of the power iterations on $\hat{\boldsymbol{U}}_{\text{POja}}(\tau)$ and we can approximately calculate it by $\hat{\boldsymbol{U}}_{PM}(\tau) \approx \text{PM}\big(\{\boldsymbol{r}(s)\}_{s\in\mathcal{T}_\tau};\hat{\boldsymbol{U}}_{\text{POja}}(\tau);L\big)$.

# The Power-Oja (P-Oja) Method

Finally, the P-Oja method is given by the following iterations:

$$\hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau+1) = \hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau) - \gamma_\tau \hat{\nabla} \hat{f}_{\mathrm{POja}}(\hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau); \mathcal{T}_\tau) \,, \tag{7}$$

where $\hat{\nabla} \hat{f}_{\mathrm{POja}}$ is the approximated gradient, evaluated as:

$$\begin{aligned} \hat{\nabla} \hat{f}_{\mathrm{POja}}(\hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau); \mathcal{T}_\tau) = {} & \overline{\boldsymbol{U}}_{PM}(\tau) \hat{\boldsymbol{U}}_{\mathrm{POja}}^H(\tau) \hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau) \\ & + \hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau) \hat{\boldsymbol{U}}_{\mathrm{POja}}^H(\tau) \overline{\boldsymbol{U}}_{PM}(\tau) - 2\overline{\boldsymbol{U}}_{PM}(\tau) \,. \end{aligned}$$

where $\hat{\boldsymbol{U}}_{PM}(\tau) \approx \mathrm{PM}\big(\{\boldsymbol{r}(s)\}_{s\in\mathcal{T}_\tau}; \hat{\boldsymbol{U}}_{\mathrm{POja}}(\tau); L\big)$.

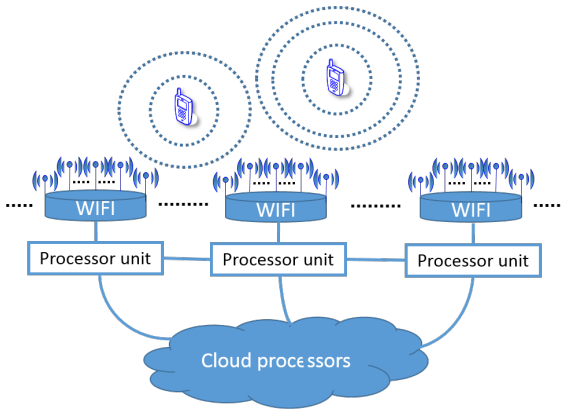
# Some Remarks for Power-Oja

- The P-Oja method is parametrized by $L$ and $T$
  - $T$: controls the variance in the sampled covariance $\hat{\boldsymbol{R}}(\mathcal{T}_\tau)$;
  - $L$: the acceleration given by the power method subroutine.
- P-Oja reduces into the Oja's learning rule when $p = 1,\ T = 1,\ L = 1$.
- If the samples in the batch is sufficient, we are very likely to obtain a rank-$p$ subspace. Recall that no guarantee for a rank-$p$ subspace for Oja.

# Decentralization

Preliminaries:

- We denote the communication network between $M$ processor units as an undirected graph $G = (V, E)$ such that $V = \{1, ..., M\}$ and $E \subseteq V \times V$.
- The graph is assumed to be sparse and connected.
- A doubly stochastic matrix $\boldsymbol{W}$ associated with $G$, s.t. $[\boldsymbol{W}]_{ij} = 0$ iff $(i, j) \notin E$.
- Each processor unit locally processes its subarray's sampling data, and meanwhile exchanges information with its neighbors in $G$.



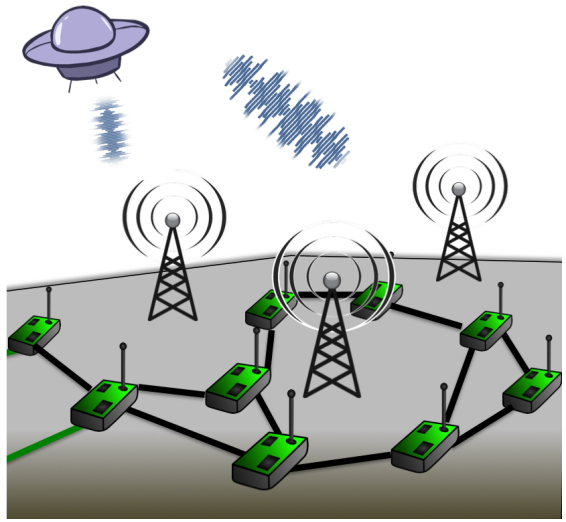



Figure: Two examples: WiFi networks and Radar networks.

# Average Consesus

- Stored and computed in processor unit $i$:
  - $\boldsymbol{r}_i(t) \in \mathbb{C}^{\frac{N}{M}} = [\boldsymbol{r}_1(t); ...; \boldsymbol{r}_M(t)]$
  - $\hat{\boldsymbol{u}}_i(\ell, \tau) \in \mathbb{C}^{\frac{N}{M}} = [\hat{\boldsymbol{u}}_1(\ell, \tau); ...; \hat{\boldsymbol{u}}_M(\ell, \tau)$
  - $z_i^0 := \boldsymbol{r}_i^H(t)\hat{\boldsymbol{u}}_i(\ell, \tau)$ and $z_i^0$

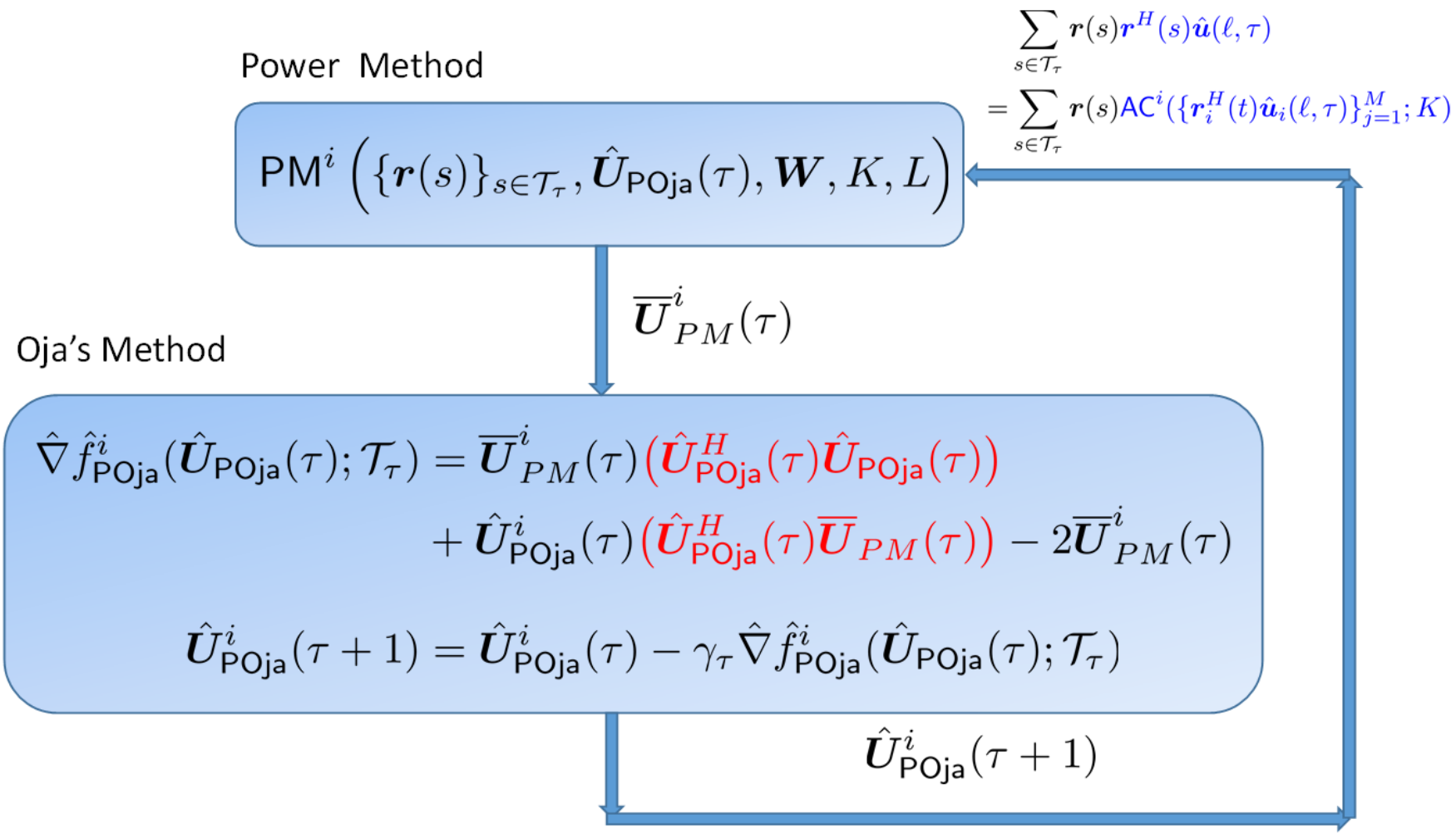
- The power iteration can be expressed as:

$$\sum_{s \in \mathcal{T}_\tau} \boldsymbol{r}(s) \underbrace{\boldsymbol{r}^H(s)\hat{\boldsymbol{u}}(\ell, \tau)}_{\substack{\text{centralized:} \\ \sum_{i=1}^M z_i^0}} = \sum_{s \in \mathcal{T}_\tau} \boldsymbol{r}(s) \underbrace{\mathrm{AC}^i(\{z_j^0\}_{j=1}^M; K)}_{\substack{\text{decentralized:} \\ z_i^{k+1} = \sum_{j=1}^M W_{ij} z_j^k}}$$

with $\|\boldsymbol{z}^K - \left(\sum_{i=1}^M z_i^0/M\right)\mathbf{1}\| \leq |\lambda_2(\boldsymbol{W})|^K \|\boldsymbol{z}_0 - \left(\sum_{i=1}^M z_i^0/M\right)\mathbf{1}\|$ ,

  - The convergence rate depends on $\lambda_2(\boldsymbol{W})$ [DKM+10].
  - $\lim_{K\to\infty} \sum_{j=1}^M z_j^0 = M \cdot \mathrm{AC}^i(\{z_j^0\}_{j=1}^M; K)$ at a geometric rate in $K$ [DKM+10].

# An Illustration of Decentralized Power-Oja




Key technique in Oja: $\boldsymbol{A}^H\boldsymbol{B} = \sum_{i=1}^M \boldsymbol{V}_i = \sum_{i=1}^M \underbrace{(\boldsymbol{A}^i)^H\boldsymbol{B}^i}_{p\times p}$.

# Some Remarks for Decentralized Power-Oja

- The message exchanged in the approximate gradient is a $p \times p$ matrix.
- It is crucial for us to choose a proper network topology.

$$\|\boldsymbol{z}^K - \bar{z}\mathbf{1}\| \leq |\lambda_2(\boldsymbol{W})|^K \|\boldsymbol{z}_0 - \bar{z}\mathbf{1}\|.$$

- It is more economical to connect the nearby units with a higher probability while the far-apart units with a lower probability.
- Example: small-world graph with optimal constant weights [XB04]:

$$\boldsymbol{W} = \boldsymbol{I} - \frac{2}{\lambda_1(\boldsymbol{L}) + \lambda_{N-1}(\boldsymbol{L})}\boldsymbol{L}\,,$$

where $\boldsymbol{L}$ is the Laplacian matrix.

# Numerical Simulations: parameter settings

- A massive array with $N = 256$ antennas grouped to $M = 64$ subarrays, each equipped with four antennas.
- $T = 1500$, SNR$= 20$dB, the power iteration is $L = 20$.
- Degree-6 small-world graph with rewiring probability 0.2.
- $\gamma_t = 5 \times 10^{-4}$ for the Oja's learning rule, $\gamma_t = 0.01t$ with P-Oja for stationary signals and $\gamma_t = 0.04t$ for non-stationary signals.

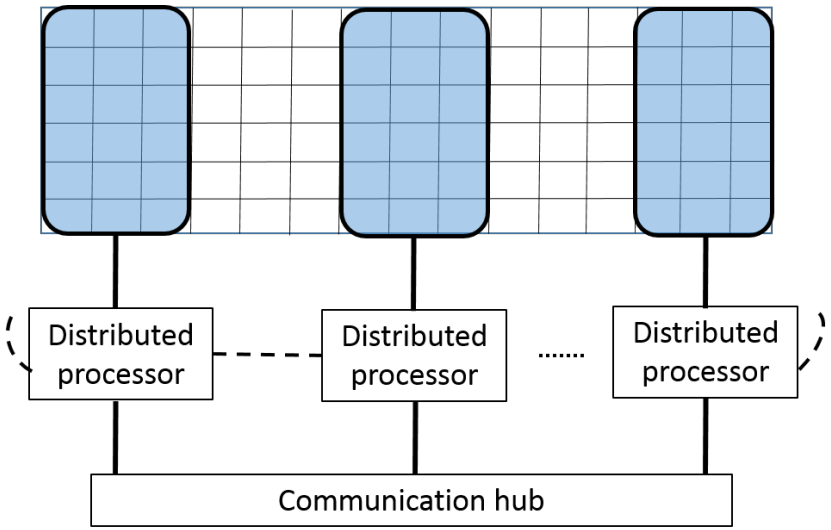



Figure: Grouping antennas into subarrays with distributed processors for spectrum sensing.

# Numerical Simulations: for stationary signals

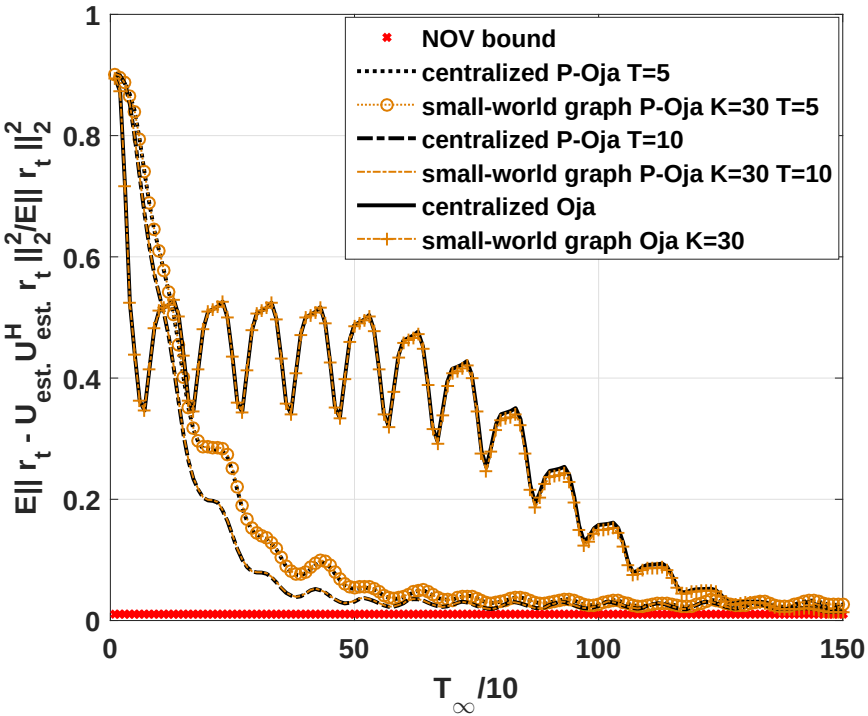



Figure: The normalized objective value for a constant 2-D signal space.

- The convergence rate increases as $T$ increases.
- The decentralized performance will approach the centralized one as $K$ increases.
- The P-Oja method converges much faster than the Oja's method, and the decentralized algorithms work well under the chosen graph.

# Numerical Simulations: for non-stationary signals

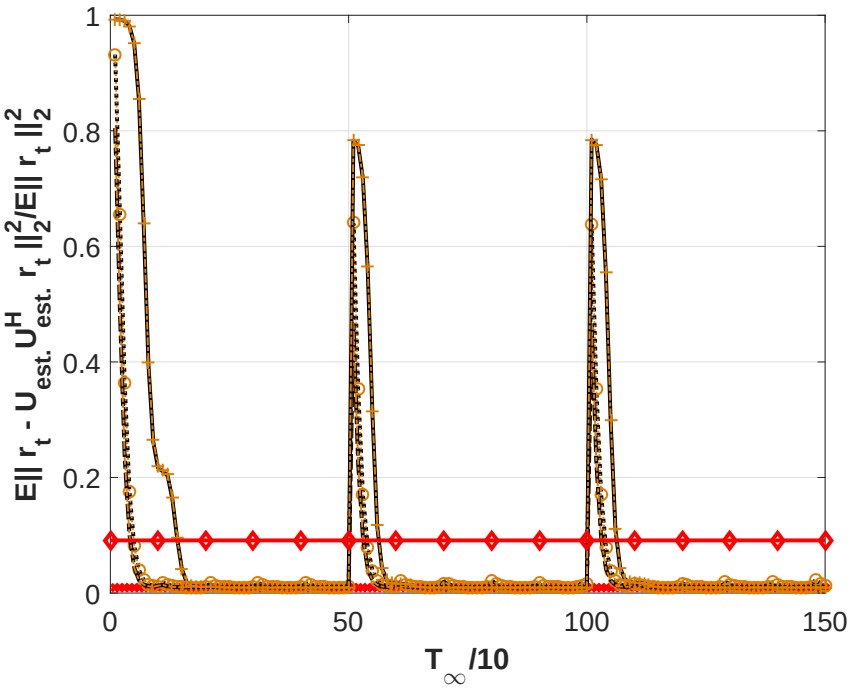



Figure: The normalized objective value for a variant 1-D signal space. The legend is the same as that in Fig. 1, except for the number of gossip iterations is now $K = 10$. The diamond-marked curve is the NOV for the conventional power method.

- As $T$ increases, the convergence rate increases.
- The decentralized and centralized methods coincide with each other when $K = 10$.
- The power method cannot track the change of the covariance.

# Conclusions

- We propose P-Oja method to integrate the Oja's learning rule and power method.
- It exhibits both tracking ability and estimation accuracy.
- All the computations are distributed into individual processor units.
- Our simulation results demonstrate that the proposed P-Oja can both track the change of statistic, but converges much faster than the conventional Oja method.

# References


[BDF13] Akshay Balsubramani, Sanjoy Dasgupta, and Yoav Freund.
The fast convergence of incremental PCA.
In *NIPS*, 2013.

[DKM+10] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione.
Gossip algorithms for distributed signal processing.
*Proc. IEEE*, 98(11):1847–1864, November 2010.

[Gv96] G. H. Golub and C. F. van Loan.
*Matrix computations*.
Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[LSM11] Lin Li, Anna Scaglione, and Johnathan H Manton.
Distributed principal subspace estimation in wireless sensor networks.
*IEEE Journal of Sel. Topics in Signal Process.*, 5(4):725–738, Aug 2011.

[OK85] E. Oja and J. Karhunen.
On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix.
*Journal of Math. Analysis and Applications*, (106):69–84, 1985.

[SPK08] Anna Scaglione, R. Pagliari, and H. Krim.
The decentralized estimation of the sample covariance.
In *Proc. Asilomar*, pages 1722–1726, November 2008.

[SPZ16] W. Suleiman, Marius Pesavento, and A. M. Zoubir.
Performance analysis of the decentralized eigendecomposition and ESPRIT algorithm.
*IEEE Trans. Signal Process.*, 64(9):2375–2386, May 2016.

[XB04] Lin Xiao and Stephen Boyd.
Fast linear iterations for distributed averaging.
*Systems & Control Letters*, (53):65–78, Feb 2004.

Thank You

&&

Question Welcomed!