Adaptive Algorithms to Track the PARAFAC Decomposition of a Third-Order Tensor

Dimitri Nion* & Nikos Sidiropoulos*

* Technical University of Crete, Chania, Crete, Greece

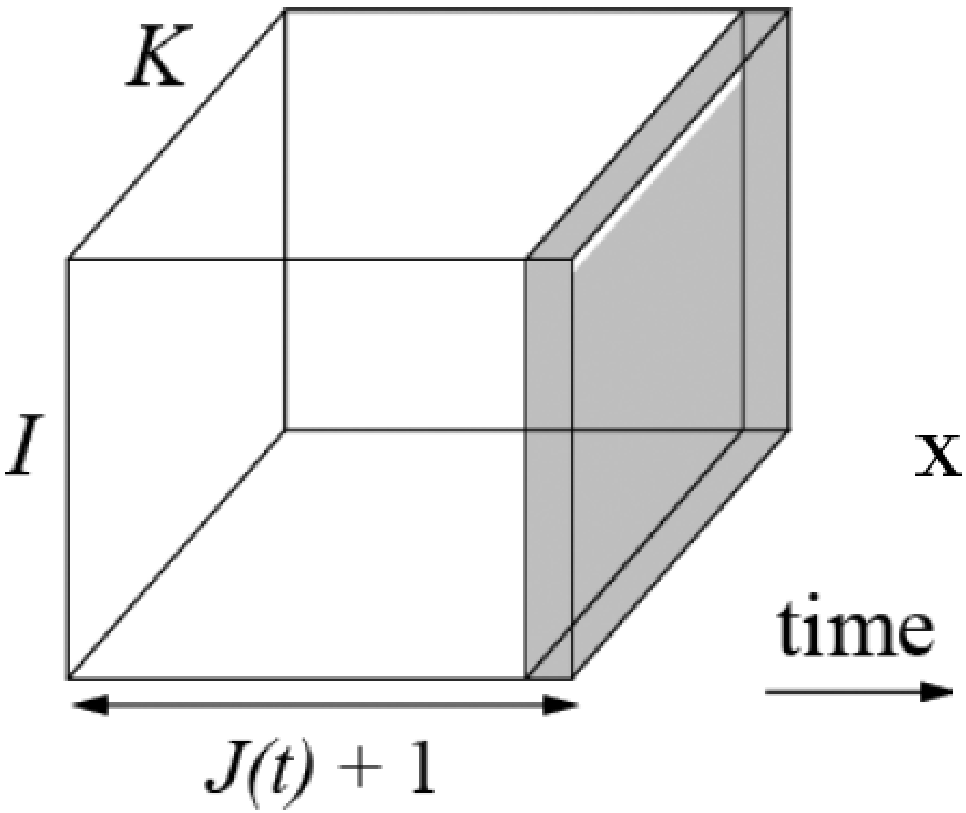
(E-mails: nikos@telecom.tuc.gr , nion@telecom.tuc.gr)

TRICAP 2009, June 14-19, 2009,
Vall de Núria - Spain

Roadmap

- Introduction (problem statement + data model)
- Sketch of basic idea
- Windowing and discounting
- Two complete algorithms:
  - PARAFAC-SDT, PARAFAC-RLST
- Complexity reduction
- Applications:
  - Target tracking in MIMO radar
  - Speech separation
- Conclusions and open issues

Setup



$$\mathbf{X}^{(1)}(t) \simeq \mathbf{H}(t)\mathbf{B}^T(t)$$

$$\mathbf{H}(t) \stackrel{\text{def}}{=} \mathbf{A}(t) \odot \mathbf{C}(t)$$

@ low cplxty?

$$\mathbf{X}^{(1)}(t+1) \simeq \mathbf{H}(t+1)\mathbf{B}^T(t+1)$$

Anyone's initial reaction

- I'll use the previous A, C as init, do a few ALS iterations
- Or I'll use the previous A, C to predict the new column of B, then do a few ALS iterations
- What's the big deal? I do have an accurate initialization from the previous time instant!
- … provided of course things change slowly …
- The big deal is that even a few ALS cycles entail complexity that is prohibitive for on-line implementation
- … and the number of cycles till convergence is "random"
- Can we do any better?
- … without giving up much on estimation accuracy?
- Context is classic in SP: adaptive algorithms

Problem statement

Problem: Given estimates $\mathbf{A}(t)$, $\mathbf{B}(t)$ and $\mathbf{C}(t)$ for the PARAFAC decomposition in R terms of the $I \times J(t) \times K$ tensor $\mathcal{X}(t)$, find recursive updates for $\mathbf{A}(t+1)$, $\mathbf{B}(t+1)$ and $\mathbf{C}(t+1)$, which stand for estimates of the PARAFAC decomposition in R terms of the $I \times J(t+1) \times K$ tensor $\mathcal{X}(t+1)$, the latter being obtained from $\mathcal{X}(t)$ after appending a new slice in the second dimension.

PARAFAC-SDT: Skeleton

STEP 1

Suppose $\mathbf{H}(t) \simeq \mathbf{H}(t+1)$ and get a first estimate of $\mathbf{b}(t+1)$:
$\mathbf{b}^T(t+1) = \mathbf{H}^\dagger(t)\mathbf{x}(t+1)$.

STEP 2

Get a first estimate of $\mathbf{B}(t+1)$: $\mathbf{B}^T(t+1) = [\mathbf{B}^T(t), \mathbf{b}^T(t+1)]$.

STEP 3

Estimate $\mathbf{H}(t+1)$: $\mathbf{H}(t+1) = \mathbf{X}^{(1)}(t+1)(\mathbf{B}^T(t+1))^\dagger$.

STEP 4

Estimate $\mathbf{A}(t+1)$ and $\mathbf{C}(t+1)$ from $\mathbf{H}(t+1)$:
For $r = 1 \ldots R$, Do
$\mathbf{H}_r(t+1) = \text{unvec}([\mathbf{H}(t+1)]_{:,r})$
$[\mathbf{c}_r, \sigma_r, \mathbf{a}_r] = \text{svd}(\mathbf{H}_r(t+1))$, —- = principal singular vectors —
$[\mathbf{C}(t+1)]_{:,r} = \sigma_r \mathbf{c}_r$ and $[\mathbf{A}(t+1)]_{:,r} = \mathbf{a}_r^*$
End

STEP 5

Re-estimate $\mathbf{B}(t+1)$ with a time-shift structure:

$$\begin{cases} \mathbf{b}^T(t+1) & = & \mathbf{H}^\dagger(t+1)\mathbf{x}(t+1), \\ \mathbf{B}^T(t+1) & = & [\mathbf{B}^T(t), \mathbf{b}^T(t+1)]. \end{cases}$$

Ignores KR structure → suboptimal; price paid for simplicity

Idea is that slow variation will imply small degradation

PARAFAC-SDT: Key issues

i) the observed matrix $\mathbf{X}^{(1)}(t)$ has to be properly windowed so as to weight past observations;

ii) pseudoinverse matrices should be recursively updated;

iii) SVDs should be replaced by SVD tracking algorithms;

iv) operations having a complexity increasing with time should be avoided.

PARAFAC-SDT: Main idea

$$\begin{cases} \mathbf{X}_{\mathrm{EW}}(t) = (\mathbf{A}(t) \odot \mathbf{C}(t))\mathbf{B}^T(t)\mathbf{\Lambda}(t) \\ \mathbf{X}_{\mathrm{EW}}(t) = \mathbf{U}_{\mathrm{EW}}(t)\Sigma_{\mathrm{EW}}(t)\mathbf{V}^H_{\mathrm{EW}}(t). \quad \text{economy-size SVD} \end{cases}$$

$$\mathbf{E}_{\mathrm{EW}}(t) \stackrel{\mathrm{def}}{=} \mathbf{U}_{\mathrm{EW}}(t)\Sigma_{\mathrm{EW}}(t)$$

$$\begin{cases} \mathbf{A}(t) \odot \mathbf{C}(t) = \mathbf{E}_{\mathrm{EW}}(t)\mathbf{W}_{\mathrm{EW}}(t) \\ \mathbf{B}^T(t)\mathbf{\Lambda}(t) = \mathbf{W}^{-1}_{\mathrm{EW}}(t)\mathbf{V}^H_{\mathrm{EW}}(t) \end{cases}$$

Idea: link by exploiting time-shift structure of $\mathbf{B}(t+1)$
Exploitation of common block between
$\mathbf{B}(t)$ and $\mathbf{B}(t+1)$
yields recursive updates we're looking for!

$$\begin{cases} \mathbf{A}(t+1) \odot \mathbf{C}(t+1) = \mathbf{E}_{\mathrm{EW}}(t+1)\mathbf{W}_{\mathrm{EW}}(t+1) \\ \mathbf{B}^T(t+1)\mathbf{\Lambda}(t+1) = \mathbf{W}^{-1}_{\mathrm{EW}}(t+1)\mathbf{V}^H_{\mathrm{EW}}(t+1) \end{cases}$$

PARAFAC-SDT: Steps and 'tricks'

- **SVD tracking:**

  For EW: Combine Bi-SVD1 with the time-updating recursion for the growing orthonormal right basis matrix in [Strobach] . This way, only one step involves complexity growing linearly with time, instead of three.

  For TW: use SWASVD [Badeau, Richard, David]

- **Recursive updates of** $\mathbf{W}$ *and* $\mathbf{W}^{-1}$

  Exploit common block and matrix inversion lemma for rank-1 updates

- **Updates of columns of** $\mathbf{A}$ *and* $\mathbf{C}$:

  Use a single Bi-SVD iteration to track the left and right principal singular vectors of $\mathbf{H}_r(t+1)$

PARAFAC-SDT: Complexity

- PARAFAC-SDT with exponential window entails per-iteration complexity that is linearly growing with time, due to first step

- Truncated window is preferable for PARAFAC-SDT, for which complexity is
$$16R^3+R^2(31IK+31N+40)+R(32IK+10K+20)$$

- One cycle of ALS: $$88R^3+R^2(64NK+64IK+64IN+24)+R(24INK+8NK+8IK+8IN)$$

- Difference significant. Also, ALS will typically do several cycles

PARAFAC-RLST

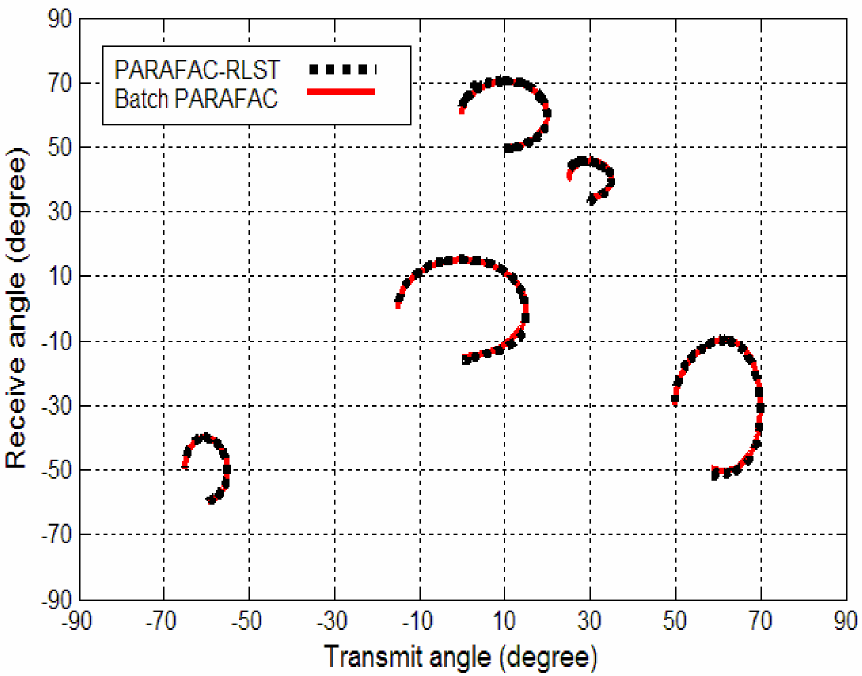
- Follows same skeleton as PARAFAC-SDT
- But draws upon basic steps of RLS (matrix inversion lemma)
- Works with exponential or truncated window
- Details in paper (*IEEE Trans. Signal Processing*, June 2009):
- Complexity much lower than one cycle of ALS:

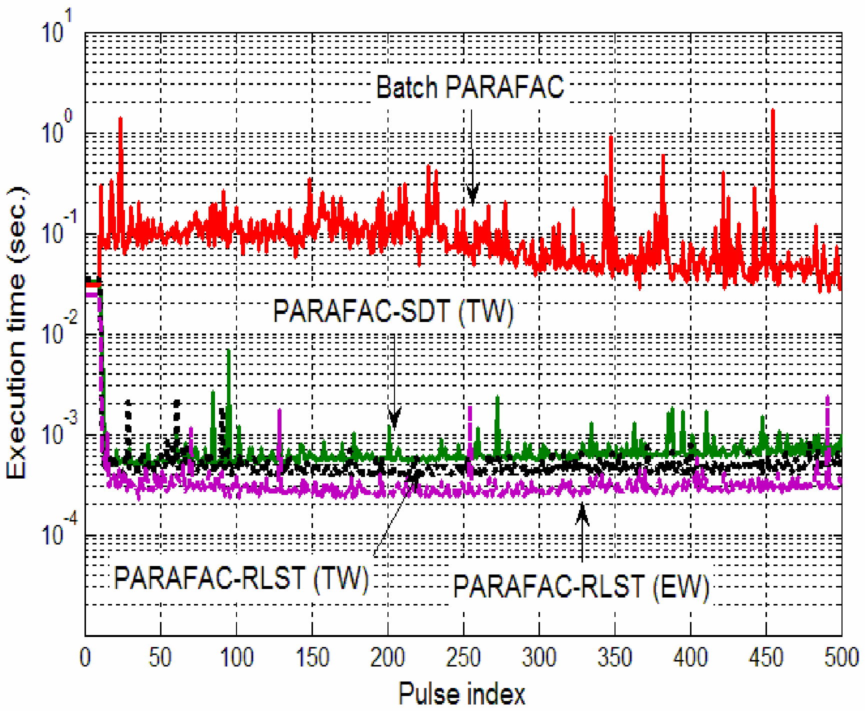
TW: $R^2(16IK + 72) + R(144IK + 10K + 20)$

EW: $R^2(16KI + 40) + R(88KI + 10K + 10)$

Application: MIMO Radar target tracking

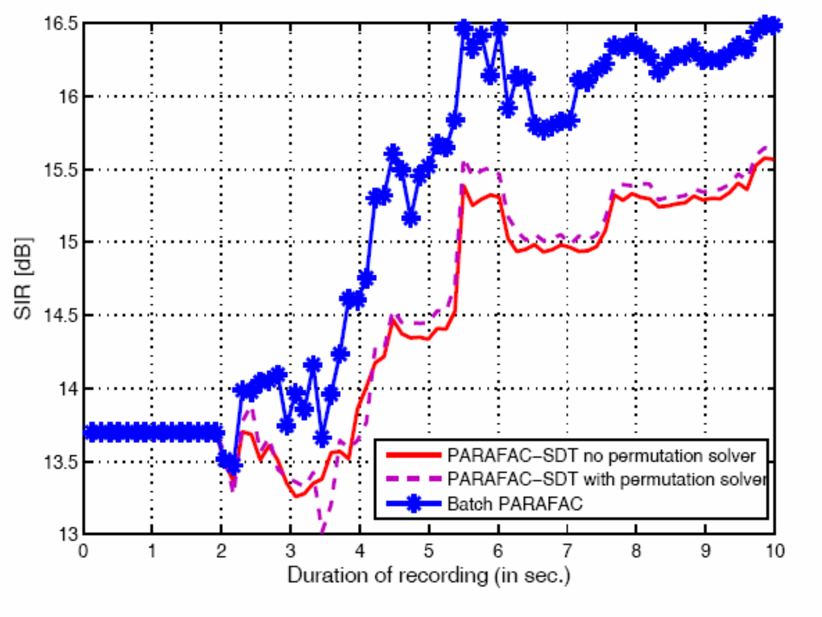
5 moving targets. Estimated trajectories. Comparison between Batch PARAFAC (init w/ est from previous ‘tick’) and PARAFAC-RLST (« Recursive Least Squares Tracking »)



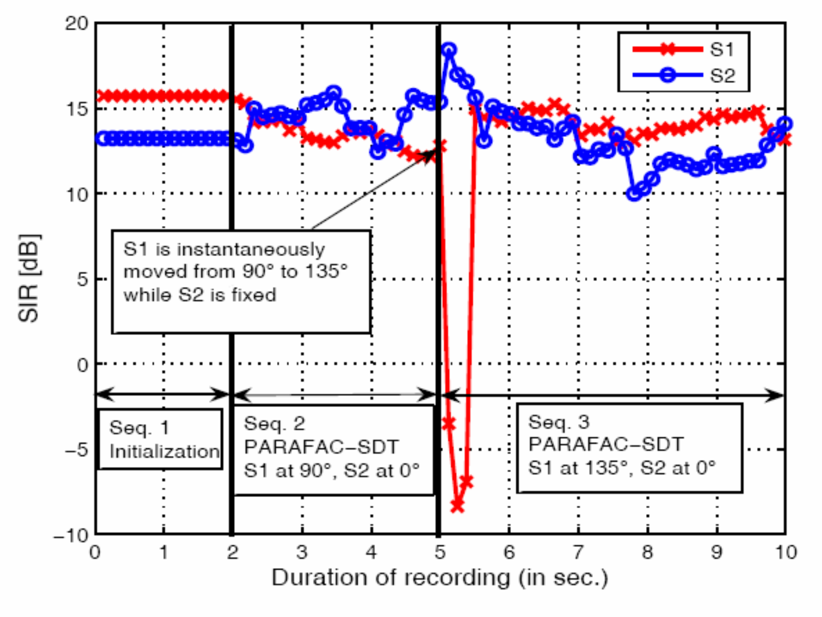


Application: Blind Speech Separation

PARAFAC-SDT, 2 speakers, 1024 freq. bins: (a) Static, speakers at 0° and 90°. (b) TV: Seq. 1: init, speakers @ 0° and 90°. Seq. 2: tracking mode. Seq. 3: speaker 2 fixed, speaker 1 moved instantaneously to 135°



(a)



(b)

Conclusions

- Proposed two adaptive algorithms to track the PARAFAC decomposition of a third-order tensor: PARAFAC-SDT, PARAFAC-RLST
- Both can be used with EW or TW;
- … but PARAFAC-SDT w/ EW has per-iteration complexity that increases linearly with time
- PARAFAC-RLST can be used with both windows.
- Excellent tracking performance, orders-of-magnitude lower complexity than `adaptively initialized’ ALS
- Generalization to higher orders where only one mode is growing: straightforward

Open issues

- Generalization to the case where two or more dimensions are growing is open
- Tracking the rank is wide-open!

Thank you ☺