Jing Li, Xiao-run Li, Li-jiao Wang, Liao-ying Zhao, 2016. Fast implementation of kernel simplex volume analysis based on modified Cholesky factorization for endmember extraction. *Frontiers of Information Technology & Electronic Engineering*, **17**(3):250-257. http://dx.doi.org/10.1631/FITEE.1500244

Fast implementation of kernel simplex volume analysis based on modified Cholesky factorization for endmember extraction

Key words: Endmember extraction, Modified Cholesky factorization, Spatial pixel purity index (SPPI), New simplex growing algorithm (NSGA), Kernel new simplex growing algorithm (KNSGA)

Corresponding author: Xiao-run Li
E-mail: lxr@zju.edu.cn
ORCID: http://orcid.org/0000-0001-7611-845X

Motivation

- **Disadvantages of KNSGA include:**

(1) random initialization, which creates inconsistency in the final results;

(2) excessive computational time due to the iterations of simplex volume calculation.

- **To improve KNSGA in two aspects**:

(1) use SPPI to solve the inconsistency in the result;
(2) use Cholesky factorization to reduce the computing time.

Method

Input: matrix $\boldsymbol{X}=[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]\in \mathrm{R}^{L\times N}$.

Step 1: initialization.

1a) Use VD to estimate p.

1b) For each pixel $\boldsymbol{x}_n$, calculate the spatial pixel purity index defined in Cui *et al.* (2013), $\boldsymbol{e}_1=\arg[\min_{\boldsymbol{x}_n}(P_{\mathrm{sppi}})]$.

1c) Find the pixel farthest from $\boldsymbol{e}_1$, set it as endmember $\boldsymbol{e}_2$, and set its position as id(1). For each sample pixel $\boldsymbol{x}_n$, calculate

$$d_n^1 = k(\boldsymbol{x}_n, \boldsymbol{x}_n) + k(\boldsymbol{e}_1, \boldsymbol{e}_1) - 2k(\boldsymbol{x}_n, \boldsymbol{e}_1),\ n = 1, 2, \cdots, N,$$

$$\boldsymbol{e}_2 = \arg[\max_{\boldsymbol{x}_n}(d_n^1)],\ \mathrm{id}(1) = \arg[\max_n(d_n^1)].$$

1d) Set iteration index m=2.

Step 2: start iteration.

2a) For each sample vector $\boldsymbol{x}_n$, calculate

$$l_n^m = \frac{1}{d_{\mathrm{id}(m)}^m}[\phi(\boldsymbol{x}_n) - \phi(\boldsymbol{e}_1)]^{\mathrm{T}}[\phi(\boldsymbol{e}_m) - \phi(\boldsymbol{e}_1)] - \frac{1}{d_{\mathrm{id}(m)}^m}\sum_{k=1}^{m-1} l_n^k d_{\mathrm{id}(k)}^k l_{\mathrm{id}(m)}^k .$$

2b) Calculate $d_n^{m+1} = d_n^m - l_n^m d_{\mathrm{id}(m)}^m l_n^m$. Find the pixel that satisfies the largest d_n^{m+1}, and then calculate

$$\boldsymbol{e}_{m+1} = \arg[\max_{\boldsymbol{x}_n}(d_n^{m+1})],$$

$$\mathrm{id}(m+1) = \arg[\max_n(d_n^{m+1})].$$

2c) If $m<p-1$, then $m=m+1$ and move to step 2a; otherwise, terminate.

Step 3: output the results $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_p\}$.

Major results (I)

Synthetic data results:

(1) We compared the results of KNSGA with respect to random initialization and SPPI initialization

Algorithm	Time (s)	Simplex volume	Average SAD
KNSGA (most)	1.342	1.520	0.042
KNSGA (second most)	1.351	1.508	0.043
KNSGA (SPPI)	1.340	1.749	0.036

(2) We compare the time consumed by KNSGA and the proposed FKNSGACF with respect to different endmember numbers and pixel sizes

Number of endmembers	Time (s)		Speedup
	KNSGA	FKNSGACF	
6	1.78	1.47	1.21×
9	3.65	2.01	1.82×
12	6.17	2.85	2.16×
15	9.37	3.60	2.60×
18	13.06	4.36	3.00×

Pixel size	Time (s)		Speedup
	KNSGA	FKNSGACF	
64×64	2.26	1.24	1.82×
100×100	3.50	2.01	1.74×
144×144	7.13	4.05	1.76×
196×196	12.84	6.94	1.85×
256×256	18.47	9.62	1.92×

Major results (II)

Real data results:

We compare the performance of KNSGA and FKNSGACF on real data

Number of endmembers	Simplex volume	
	KNSGA	FKNSGACF
6	4.33×10^{18}	4.33×10^{18}
10	1.67×10^{29}	1.67×10^{29}
14	1.83×10^{38}	1.83×10^{38}
18	1.29×10^{46}	1.29×10^{46}
22	8.33×10^{52}	8.33×10^{52}

Number of endmembers	Time (s)		Speedup
	KNSGA	FKNSGACF	
6	52.8	49.2	1.07×
10	135.5	83.5	1.62×
14	273.4	151.9	1.80×
18	415.4	180.2	2.31×
22	607.7	236.5	2.57×

Conclusions

- Fast implementation of KNSGA is achieved
  - SPPI is used to resolve the problem inherent to initialization
  - Modified Cholesky factorization is used to reduce the computational complexity

- Experiments indicate that the proposed algorithm achieves ideal performance in reducing computation time and acquiring larger simplex volumes