## **Fast dictionary learning based on rank-1 matrix pursuit with sparse constraint**

In the proposed rank-1 pursuit dictionary learning (R1DL) model, we aim to minimize the following energy function, identifying rank-1 basis pair ***u*** (*T*×1 vector with unit length) and ***v*** (*P*×1 vector) to decompose the input signal matrix ***S*** of dimension *T*×*P*:

|  |  |
| --- | --- |
|  | (1) |

Eq. 1 indicates that the product of ***u*** and ***v*** are supposed to well-fit the input ***S*** while the total number of non-zero element in ***v*** should be smaller than the given parameter *r*. The minimization could be solved by iteratively updating ***u*** and ***v*** until convergence (i.e. ||***u****k*+1-***u****k*||<*ε*):

|  |  |
| --- | --- |
|  | (2) |
|  | (3) |

Eq. 2 involves calculating the product between ***S*** and vector ***u*** (which would be initialized at the first iteration), followed setting all elements in the resulting ***v*** smaller than its *r*-th largest value. After each run, the pair of the rank-1 basis [***u***, ***v***] would be found, the input matrix ***S*** would be then deflated:

|  |  |
| --- | --- |
|  | (4) |

where *K* is the total number of expected basis (i.e. dictionary atoms) to fully describe the input data. The totally *K* number of ***u*** vectors would constitute the learned dictionary while the corresponding sparse vectors ***v*** contain the loading coefficients of each dictionary atom. The algorithmic pipeline is illustrated in Fig. 2.

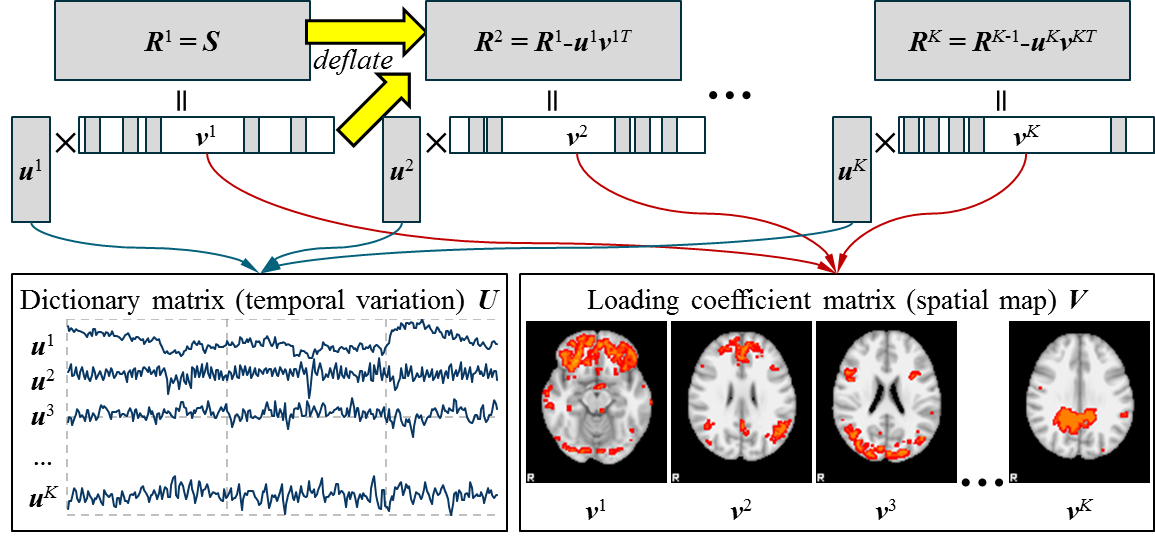


Figure 2. Illustration of the rank1 pursuit dictionary learning applied on fMRI dataset with results (dictionary matrix ***U*** and the corresponding loading coefficient matrix ***V*** mapped on brain cortical space) from running examples.

Comparing with the widely-applied whole-matrix gradient-based dictionary learning algorithm, including online dictionary learning (using stochastic gradient descent algorithm) and the K-SVD (using gradient descent algorithm), the proposed rank-1 pursuit dictionary learning algorithm is advantages at:

1. The method is not based on gradient calculation, thus needs no tuning for the learning rate/step size, while also avoid the slower convergence speed around the solution problem of the gradient-based algorithm.
2. The proposed algorithm is generally faster and more scalable, especially on very large datasets as it avoids maintaining the large coefficient matrix ***V*** (of dimension *P*\**K*, where *P* is usually on the magnitude of millions) in the memory.
3. From our experiment results it could be observed that atoms learned in the first several runs would be more functionally meaningful while latter-learned atoms tend to be noises. This observation is consistent with the fact that the algorithm is essentially the sparse-constraint principle component analysis, thus the cardinality of the atoms learned is related to their importance in explaining the input data. In comparison, atoms learned by previous dictionary learning methods are unrelated to their cardinality as all atoms would be learned simultaneously. This characteristic helps determining the dictionary size for the learning, which has been a difficult problem for applying dictionary learning in practice, as we could safely set a sufficiently large dictionary size (i.e. *K*) for the iteratively learning and truncate the latter noise atoms.
4. In addition, learning dictionary atoms in the iterative fashion enables accelerated user feedback, as the model could generate the results from the first few learned atoms in near real-time.