Minimal Realization Problems for Hidden Markov Models

Motivation

Randomness

Observe stationary random process

$y(t) \in [d]$ = {1,2,..., d}

- The random process is characterized by the joint distribution

(Probabilities of strings of any length N)

$$\mathcal{S}^{(\infty)} = \left\{ \mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) : \quad \forall \mathbf{l}_1^N \in [d]^N, \ \forall N \in \mathbb{Z} \right\}$$

Motivation

Randomness

Structures

Observe stationary random process

$y(t) \in [d]$ = {1,2,..., d}

- The random process is characterized by the joint distribution

(Probabilities of strings of any length N)

$$\mathcal{S}^{(\infty)} = \left\{ \mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) : \quad \forall \mathbf{l}_1^N \in [d]^N, \ \forall N \in \mathbb{Z} \right\}$$

- Can we find a finite state system description of the process?

Motivation

Randomness

Observed random process

$y(t) \in [d]$ = {1,2,..., d}

- Assume that the process can be described by a finite state system

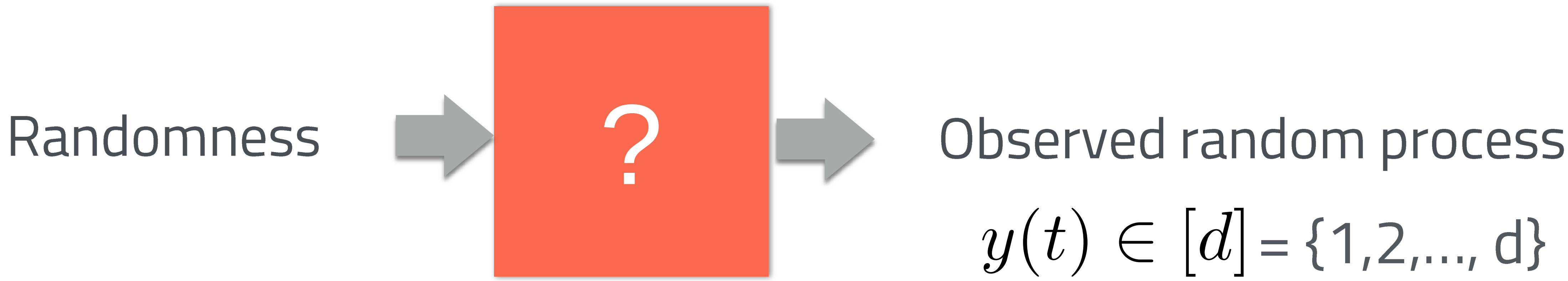
$$\theta = (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k \times k} : l \in [d])$$

$$w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v,$$
$$z_t = u' w_t,$$

$$\forall \mathbf{l}_1^N \in [d]^N, \forall N \in \mathbb{Z}$$

$$\mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) = z_N(\mathbf{l}_1^N)$$
$$= u' A^{(l_n)} \dots A^{(l_1)} A^{(l_{-1})} \dots A^{(l_{-n})} v$$

Motivation



$y(t) \in [d]$ = {1,2,..., d}

- Assume that the process can be described by a finite state system

$$\theta = (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

$$\boxed{\begin{aligned} &w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v, \\ &z_t = u' w_t, \end{aligned}}$$

State has no physical meaning
$w_t \in \mathbb{R}^k$

$$\forall \mathbf{l}_1^N \in [d]^N, \forall N \in \mathbb{Z}$$

$$\begin{aligned} \mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) &= z_N(\mathbf{l}_1^N) \\ &= u' A^{(l_n)} \ldots A^{(l_1)} A^{(l_{-1})} \ldots A^{(l_{-n})} v \end{aligned}$$

Finite state system realization

$$\theta = \ (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

- Multi-linear system $w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v,$

  $z_t = u' w_t,$

Finite state system realization

$$\theta = \ (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

- Multi-linear system $w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v,$
  $z_t = u' w_t,$

- Quasi- HMM realization

If $\mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) = z_N(\mathbf{l}_1^N)$ is a valid joint distribution

and $u' \left[\sum_{j\in[d]} A^{(j)} \right] = u', \quad \left[\sum_{j\in[d]} A^{(j)} \right] v = v.$ (stationarity)

Finite state system realization

$$\theta = \ (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

- Multi-linear system $w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v,$
  $z_t = u' w_t,$

- Quasi- HMM realization

  If $\mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) = z_N(\mathbf{l}_1^N)$ is a valid joint distribution

  and $u' \left[\sum_{j\in[d]} A^{(j)} \right] = u', \quad \left[\sum_{j\in[d]} A^{(j)} \right] v = v.$ (stationarity)

- HMM realization

  If in addition $v = \mathbf{e}_k, \quad u \in \mathbb{R}_+^k, \sum_i u_i = 1,$
  $A^{(j)} \in [0,1]^{k\times k}, \forall j \in [d],$

Finite state system realization

$$\theta = \ (d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

- Multi-linear system $w_{t+1} = A^{(l_t)} w_t, \quad w_0 = v,$

$$z_t = u' w_t,$$

- Quasi- HMM realization

If $\mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) = z_N(\mathbf{l}_1^N)$ is a valid joint distribution

and $u' \left[\sum_{j\in[d]} A^{(j)} \right] = u', \quad \left[\sum_{j\in[d]} A^{(j)} \right] v = v.$ (stationarity)

- HMM realization

If in addition $v = \mathbf{e}_k, \quad u \in \mathbb{R}^k_+, \sum_i u_i = 1,$

$$A^{(j)} \in [0,1]^{k\times k}, \forall j \in [d],$$

1-1 mapping to

Transition: $Q \in \mathbb{R}^{k\times k}$

Observation: $O \in \mathbb{R}^{d\times k}$

Realization problems

- **Input:** probabilities of length N strings of an HMM

$$\mathcal{S}^{(N)} = \{\mathbb{P}(y_1 = l_1, \cdots, y_N = l_N) : \ \forall \mathbf{l}_1^N \in [d]^N\}$$

- **Output:**

order

$$(d, k, u \in \mathbb{R}^k, v \in \mathbb{R}^k, A^{(l)} \in \mathbb{R}^{k\times k} : l \in [d])$$

✓ Minimal order quasi-HMM realization

✓ Minimal order HMM realization

$$\mathcal{S}^{(\infty)} = \left\{\mathbb{P}(\mathbf{y}_1^N = \mathbf{l}_1^N) : \quad \forall \mathbf{l}_1^N \in [d]^N, \ \forall N \in \mathbb{Z}\right\}$$

$$\mathbb{P}(\mathbf{y}_1^N = \mathbf{l}_1^N) = u' A^{(l_1)} A^{(l_1)} \dots A^{(l_N)} v, \quad \forall \mathbf{l}_1^N \in [d]^N$$

Realization problems

- **Informational** complexity
  - ✓ When is the minimal model identifiable from $\mathcal{S}^{(N)}$? window size N =?
- **Computational** complexity
  - ✓ Can we compute the minimal realization with efficient algorithms?
- **Statistical** complexity
  - ✓ Are the algorithms robust to estimation noise?

In general, learning HMMs is hard

Realization problems

- **Informational** complexity
  - ✓ When is the minimal model identifiable from $\mathcal{S}^{(N)}$? window size N =?

Main contribution 1:

- **Computational** complexity
  - ✓ Can we compute the minimal realization with efficient algorithms?

Main contribution 2

- **Statistical** complexity
  - ✓ Are the algorithms robust to estimation noise?

In general, learning HMMs is hard

Quasi-HMM realization

Quasi-HMM realization

- **Input:** probabilities of length $N = 2n + 1$ strings

$$y_{-n}, \dots, y_{-1} \;\; y_0 \;\; y_1, \dots, y_n$$

$$\mathcal{S}^{(N)} = \{\mathbb{P}_{y_{-n}, \cdots, y_n}\}$$

- **Output:** a minimal quasi-HMM realization

$$\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$$

- **Key:**

Arrange $\mathcal{S}^{(N)} = \{\mathbb{P}_{y_{-n}, \cdots, y_n}\}$ into matrices:

$$H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$$

Matrix rank decomposition of $H^{(0)}$

equivalent minimal quasi realization up to linear transformation

Complexity?

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

Enumerate all length n strings: $L(\mathbf{l}_1^n) = (l_1 - 1)d^{n-1} + (l_2 - 1)d^{n-2} + \cdots + l_n, \quad \forall \mathbf{l}_1^n = [d]^n$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(0)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\left(\mathbf{y}_0^{n-1} = \mathbf{l}_1^n,\ \mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n}\right)$$

$$= u' \boxed{A^{(l_n)} \dots A^{(l_1)}} \boxed{A^{(l_{-1})} \dots A^{(l_{-n})}} v$$

n letters now and future

n letters in the past

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

Enumerate all length n strings: $L(\mathbf{l}_1^n) = (l_1 - 1)d^{n-1} + (l_2 - 1)d^{n-2} + \dots + l_n, \quad \forall \mathbf{l}_1^n = [d]^n$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(0)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\Big(\mathbf{y}_0^{n-1} = \mathbf{l}_1^n,\ \mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n}\Big)$$

$$= u' \underbrace{A^{(l_n)} \dots A^{(l_1)}}_{\text{n letters now and future}} \underbrace{A^{(l_{-1})} \dots A^{(l_{-n})}}_{\text{n letters in the past}} v$$

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

$$H^{(0)} = \begin{bmatrix} u' A^{(1)} \dots A^{(1)} \\ u' A^{(1)} \dots A^{(2)} \\ \vdots \\ u' A^{(d)} \dots A^{(d)} \end{bmatrix} \left[A^{(1)} \dots A^{(1)} v, \cdots, A^{(d)} \dots A^{(d)} v \right]$$

Enumerate all length n strings: $L(\mathbf{l}_1^n) = (l_1 - 1)d^{n-1} + (l_2 - 1)d^{n-2} + \cdots + l_n, \quad \forall \mathbf{l}_1^n = [d]^n$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(0)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\Big(\mathbf{y}_0^{n-1} = \mathbf{l}_1^n,\ \mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n}\Big)$$
$$= u' \underbrace{\boxed{A^{(l_n)} \dots A^{(l_1)}}}_{\text{n letters now and future}} \underbrace{\boxed{A^{(l_{-1})} \dots A^{(l_{-n})}}}_{\text{n letters in the past}} v$$

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

$$H^{(0)} = \begin{bmatrix} u' & & {}^{(1)} \\ u' & U & {}^{(2)} \\ & & \\ u' & & {}^{(d)} \end{bmatrix} \begin{bmatrix} A^{(1)} \dots A & V' & {}^{l)} \dots A^{(d)} v \end{bmatrix}$$

Enumerate all length n strings: $L(\mathbf{l}_1^n) = (l_1 - 1)d^{n-1} + (l_2 - 1)d^{n-2} + \dots + l_n, \quad \forall \mathbf{l}_1^n = [d]^n$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(j)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\Big(\mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n},\ y_0 = j,\ \mathbf{y}_1^n = \mathbf{l}_1^n\Big),$$

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(j)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\Big(\mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n},\ y_0 = j,\ \mathbf{y}_1^n = \mathbf{l}_1^n\Big),$$
$$= u' \boxed{A^{(l_n)} \dots A^{(l_1)}}\ \boxed{A^{(l_0)}}\ \boxed{A^{(l_{-1})} \dots A^{(l_{-n})}} v$$

n letters in the future

Conditional on current letter

n letters in the past

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(j)}]_{L(\mathbf{l}_1^n),L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\left(\mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n},\ y_0 = j,\ \mathbf{y}_1^n = \mathbf{l}_1^n\right),$$

$$= u' \underbrace{A^{(l_n)} \dots A^{(l_1)}}_{\text{n letters in the future}} \underbrace{A^{(l_0)}}_{\text{Conditional on current letter}} \underbrace{A^{(l_{-1})} \dots A^{(l_{-n})}}_{\text{n letters in the past}} v$$

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$

$$H^{(j)} = \begin{bmatrix} u' A^{(1)} \dots A^{(1)} \\ u' A^{(1)} \dots A^{(2)} \\ \vdots \\ u' A^{(d)} \dots A^{(d)} \end{bmatrix} A^{(j)} \left[A^{(1)} \dots A^{(1)} v, \cdots, A^{(d)} \dots A^{(d)} v \right]$$

Quasi-HMM realization (algorithm)

- **Input:** $\mathcal{S}^{(N)}$ arranged into $H^{(0)}, \{H^{(j)} : j \in [d]\} \in \mathbb{R}^{d^n \times d^n}$

$$[H^{(j)}]_{L(\mathbf{l}_1^n), L(\mathbf{l}_{-1}^{-n})} = \mathbb{P}\left(\mathbf{y}_{-1}^{-n} = \mathbf{l}_{-1}^{-n},\ y_0 = j,\ \mathbf{y}_1^n = \mathbf{l}_1^n\right),$$

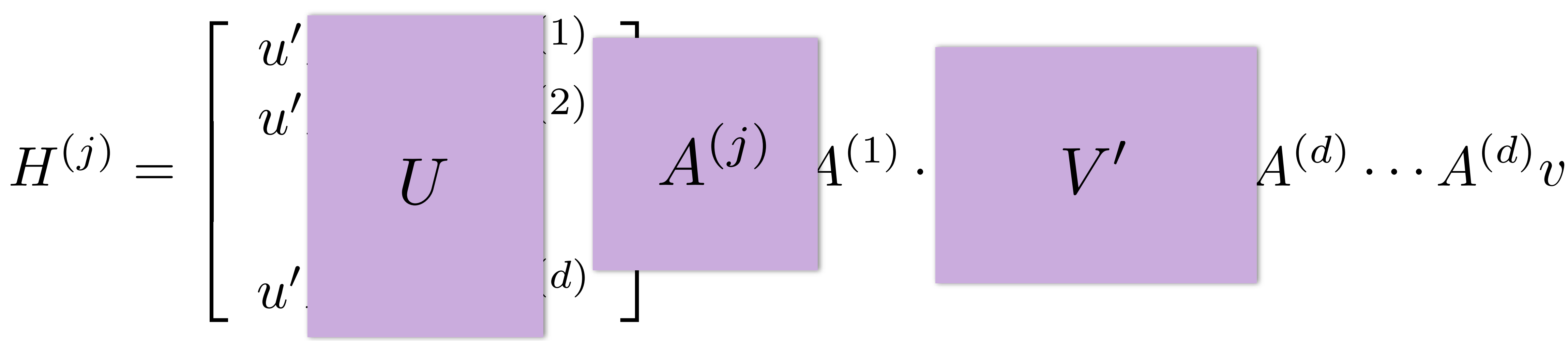
$$= u' \boxed{A^{(l_n)} \ldots A^{(l_1)}}\ \boxed{A^{(l_0)}}\ \boxed{A^{(l_{-1})} \ldots A^{(l_{-n})}} v$$

n letters in the future

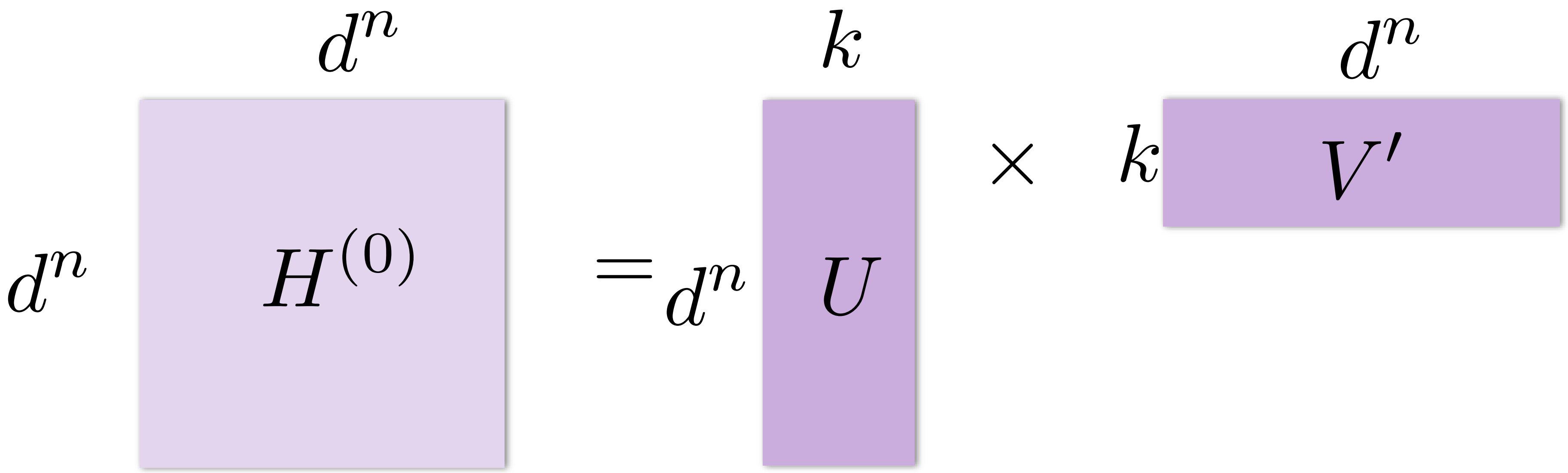
Conditional on current letter

n letters in the past

- assume minimal realization $\theta^o = (d, k, u, v, A^{(l)} : l \in [d])$



Quasi-HMM realization (algorithm)

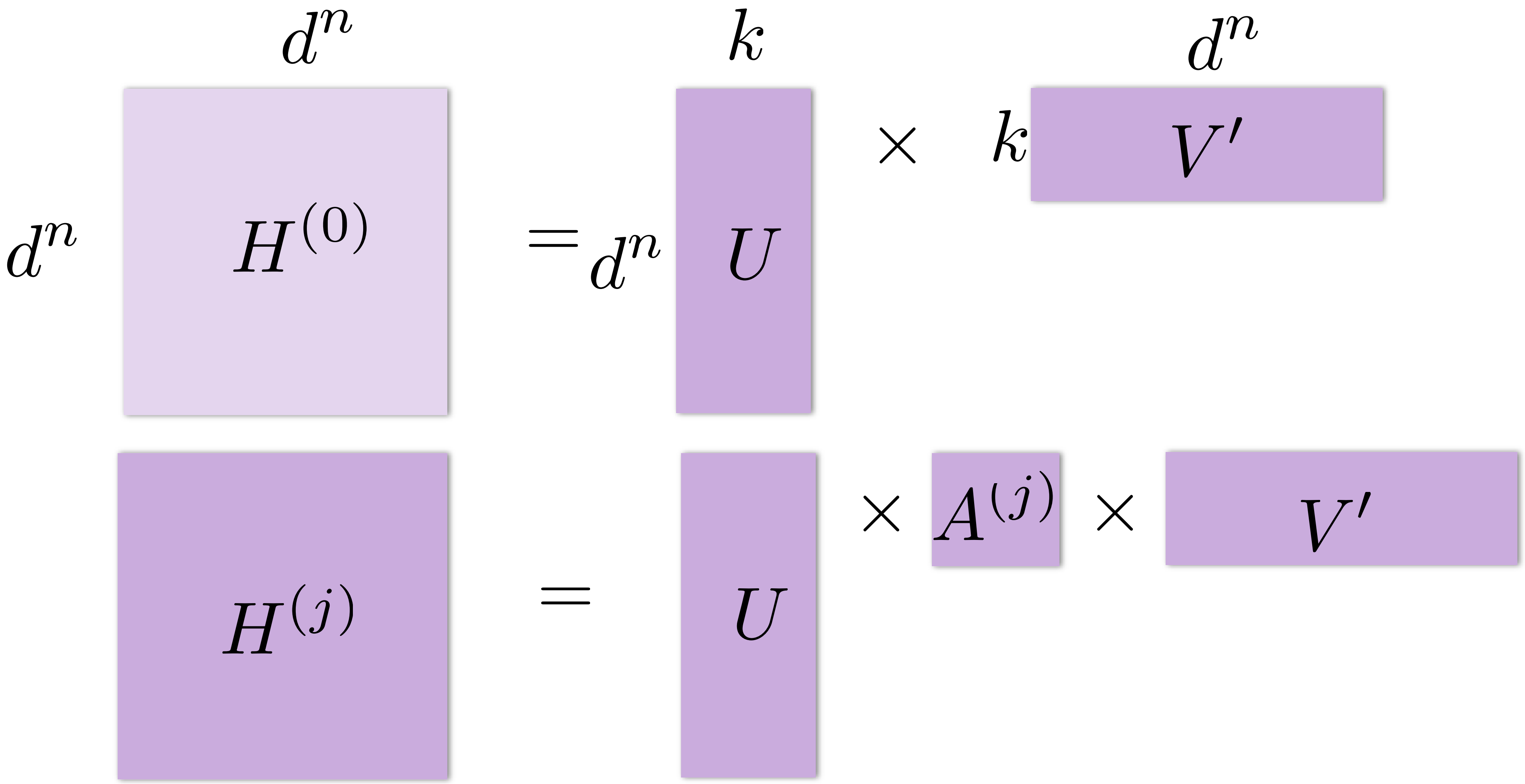


matrix rank = k
If U, V are of full column rank

Algorithm 1 (Rank decomposition)

$$H^{(0)} = UV'$$

Quasi-HMM realization (algorithm)



Algorithm 1 (Rank decomposition)

$$H^{(0)} = UV'$$

$$\widetilde{A}^{(j)} = U^{\dagger} H^{(j)} (V^{\dagger})', \quad \forall j \in [d]$$

$$\widetilde{u} = U'\mathbf{e}$$

$$\widetilde{v} = V'\mathbf{e}$$

Unique up to linear transformation

Require U V have column rank = k

Quasi-HMM realization (informational complexity)

How large N needs to be so that U, V have full column rank = k?

$$U = \begin{bmatrix} u' A^{(1)} \dots A^{(1)} \\ u' A^{(1)} \dots A^{(2)} \\ \vdots \\ u' A^{(d)} \dots A^{(d)} \end{bmatrix} T \in \mathbb{R}^{d^n \times k}$$

In practice, used as a heuristic algorithm

$$V = \begin{bmatrix} v'(A^{(1)} \dots A^{(1)})' \\ v'(A^{(1)} \dots A^{(2)})' \\ \vdots \\ v'(A^{(d)} \dots A^{(d)})' \end{bmatrix} (T^{-1})' \in \mathbb{R}^{d^n \times k}$$

Quasi-HMM realization (informational complexity)

How large N needs to be so that U, V have full column rank = k?

$$U = \begin{bmatrix} u' A^{(1)} \dots A^{(1)} \\ u' A^{(1)} \dots A^{(2)} \\ \vdots \\ u' A^{(d)} \dots A^{(d)} \end{bmatrix} T \in \mathbb{R}^{\boxed{d^n \times k}}$$

$$V = \begin{bmatrix} v' (A^{(1)} \dots A^{(1)})' \\ v' (A^{(1)} \dots A^{(2)})' \\ \vdots \\ v' (A^{(d)} \dots A^{(d)})' \end{bmatrix} (T^{-1})' \in \mathbb{R}^{\boxed{d^n \times k}}$$

Best hope:

$$n \sim \mathcal{O}\left(\log_d k\right)$$

Quasi-HMM realization (informational complexity)

How large N needs to be so that U, V have full column rank = k?

$$U = \begin{bmatrix} u' A^{(1)} \dots A^{(1)} \\ u' A^{(1)} \dots A^{(2)} \\ \vdots \\ u' A^{(d)} \dots A^{(d)} \end{bmatrix} T \in \mathbb{R}^{d^n \times k}$$

$$V = \begin{bmatrix} v'(A^{(1)} \dots A^{(1)})' \\ v'(A^{(1)} \dots A^{(2)})' \\ \vdots \\ v'(A^{(d)} \dots A^{(d)})' \end{bmatrix}$$

Theorem 1:
For **almost all** HMMs

$$\mathbf{n} > \mathbf{2}\log_{\mathbf{d}}(\mathbf{k})$$

guarantees U V full rank

Quasi-HMM realization (informational complexity)

$\Theta^o_{(d,k)} = \{\theta^o : \text{ order } k, \text{ alphabet size } d\}$ $N = 2n + 1$

Theorem 1 (Information complexity for quasi-HMM realization).

(1) Consider $\Theta^h_{(d,k)}$, the class of all HMMs with output alphabet size d and order k. There exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,k)}$, such that for all the output process generated by HMMs in the set $\Theta^h_{(d,k)} \backslash \mathcal{E}$, the information in $\mathcal{S}^{(N)}$ is sufficient for computing the minimal quasi-HMM realization, for $N = 2n + 1$ with

$$n > 2\log_d(k). \tag{16}$$

(2) For any (d, k) pair, randomly generated a HMM in $\Theta^h_{(d,k)}$. If for a given window size $N = 2n + 1$, the matrix $H^{(0)} \in \mathbb{R}^{d^n \times d^n}$ constructed with $\mathcal{S}^{(N)}$ is of the maximal rank k, then for all output processes generated by $\Theta^o_{(d,k)}$, excluding a measure zero set, $\mathcal{S}^{(N)}$ is sufficient for computing the minimal quasi-HMM realization.

Quasi-HMM realization (informational complexity)

$$\Theta^o_{(d,k)} = \{\theta^o : \text{ order } k, \text{ alphabet size } d\} \quad N = 2n+1$$

Theorem 1 (Information complexity for quasi-HMM realization).

(1) Consider $\Theta^h_{(d,k)}$, the class of all HMMs with output alphabet size d and order k. There exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,k)}$, such that for all the output process generated by HMMs in the set $\Theta^h_{(d,k)} \backslash \mathcal{E}$, the information in $\mathcal{S}^{(N)}$ is sufficient for computing the minimal quasi-HMM realization, for $N = 2n+1$ with

$$\boxed{n > 2\log_d(k).} \tag{16}$$

(2) For any (d,k) pair, randomly generated a HMM in $\Theta^h_{(d,k)}$. If for a given window size $N = 2n+1$, the matrix $H^{(0)} \in \mathbb{R}^{d^n \times d^n}$ constructed with $\mathcal{S}^{(N)}$ is of the maximal rank k, then for all output processes generated by $\Theta^o_{(d,k)}$, excluding a measure zero set, ufficient for computing the minimal quasi-HMM realization.

Polynomial computational complexity

$$\mathcal{O}(d(d^n)^3) \sim \mathcal{O}(dk^6)$$

Quasi-HMM realization (informational complexity)

$$\Theta^o_{(d,k)} = \{\theta^o : \text{ order } k, \text{ alphabet size } d\} \quad N = 2n+1$$

Theorem 1 (Information complexity for quasi-HMM realization).

(1) Consider $\Theta^h_{(d,k)}$, the class of all HMMs with output alphabet size d and order k. There exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,k)}$, such that for all the output process generated by HMMs in the set $\Theta^h_{(d,k)} \backslash \mathcal{E}$, the information in $\mathcal{S}^{(N)}$ is sufficient for computing the minimal quasi-HMM realization, for $N = 2n+1$ with

$$\boxed{n > 2\log_d(k).} \tag{16}$$

(2) For any (d,k) pair, randomly generated a HMM in $\Theta^h_{(d,k)}$. If for a given window size $N = 2n+1$, the matrix $H^{(0)} \in \mathbb{R}^{d^n \times d^n}$ constructed with $\mathcal{S}^{(N)}$ is of the maximal rank k, then for all output processes generated by $\Theta^o_{(d,k)}$, excluding a measure zero set, ufficient for computing the minimal quasi-HMM realization.

Polynomial computational complexity

$$\mathcal{O}(d(d^n)^3) \sim \mathcal{O}(dk^6)$$

- For almost all HMM, min order quasi realization is easy
- However, there exist information theoretic hard cases

Sketch of the proof

- We want to show for generic choices of Q and O, if $N = 2n+1$ satisfies $n > 2\log_d(k)$, the matrices U, V have full column rank k.
- Since the minors of U and V are nonzero polynomials in the elements of Q and O, it is enough to show for some specific choice of Q and O.
- $Q \in \mathbb{R}^{k\times k}$: the state shifting matrix

$$Q_{i-1,i} = 1, \text{ for } 2 \leq i \leq k, \text{ and } \quad Q_{k,1} = 1,$$

  $O \in \mathbb{R}^{d\times k}$: columns are independent unit-norm d-dimensional random vectors.
  The i-th column of U is given by

$$U_{[:,i]} = O_{[:,i]} \odot \ldots O_{[:,i+n-1]}.$$

- Show that with probability greater than 0 (random matrix, concentration ineq)

$$\sigma_{min}(U'U) > 0.$$

Minimal quasi-HMM realization (summary)

- Key to quasi-HMM realization:
  - ✓ conditional independence,
  - ✓ matrix decomposition, unique up to linear transformation
  - ✓ Minimal order is determined by the rank of $H^{(0)}$

- For **almost all** HMM, minimal quasi-HMM realization is easy
  - ✓ Informational : $N \sim \log_d(k)$
  - ✓ Computational : matrix factorization $\mathcal{O}(dk^6)$

Minimal quasi-HMM realization (summary)

- Key to quasi-HMM realization:
  - conditional independence,
  - matrix decomposition, unique up to linear transformation
  - Minimal order is determined by the rank of $H^{(0)}$

- For **almost all** HMM, minimal quasi-HMM realization is easy
  - Informational : $N \sim \log_d(k)$
  - Computational : matrix factorization $\mathcal{O}(dk^6)$

Minimal quasi-HMM realization (summary)

- Key to quasi-HMM realization:
  - conditional independence,
  - matrix decomposition, unique up to linear transformation
  - Minimal order is determined by the rank of $H^{(0)}$

- For **almost all** HMM, minimal quasi-HMM realization is easy
  - Informational : $N \sim \log_d(k)$
  - Computational : matrix factorization $\mathcal{O}(dk^6)$

- Smoothed analysis VS generic analysis

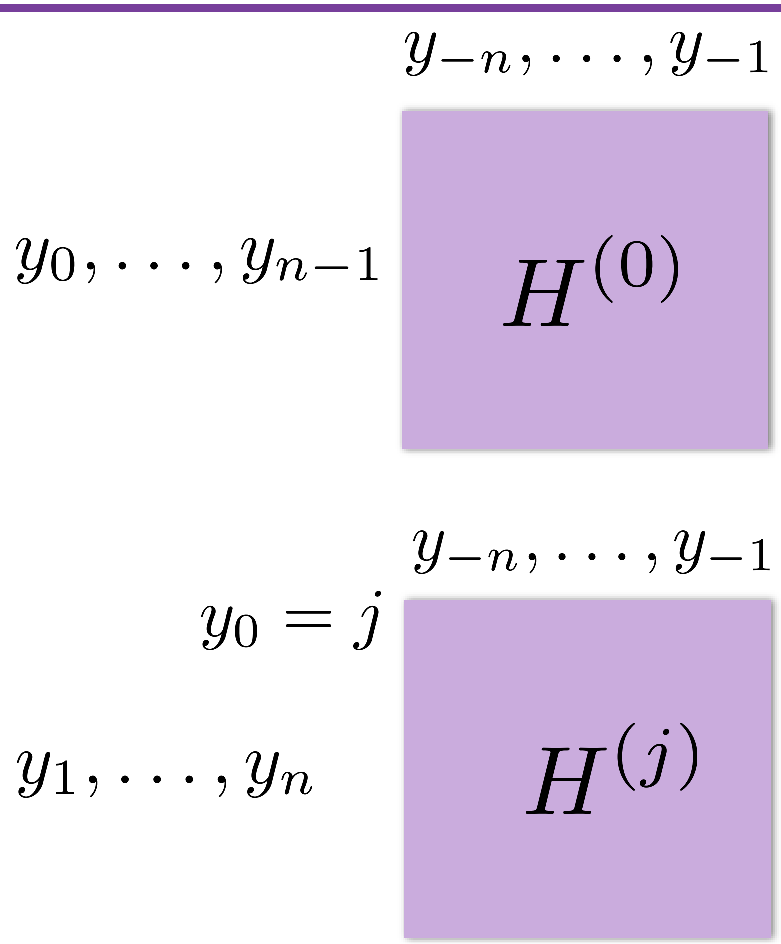
Minimal HMM realization

Minimal HMM realization

- HMM conditional independence : probabilities in product form

Minimal HMM realization

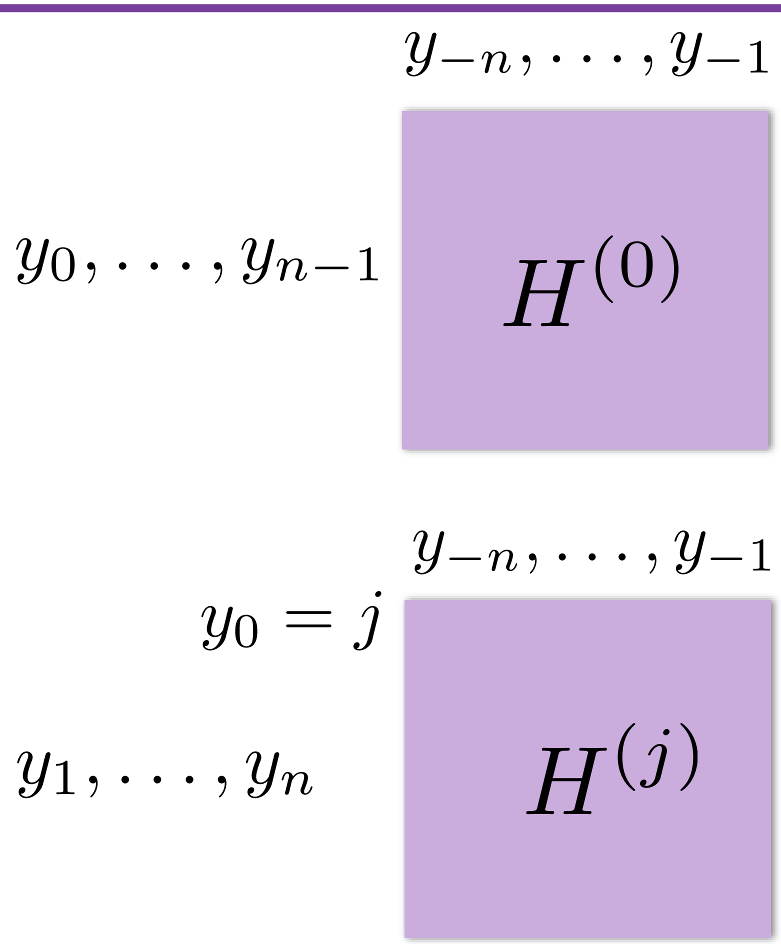
- HMM conditional independence : probabilities in product form



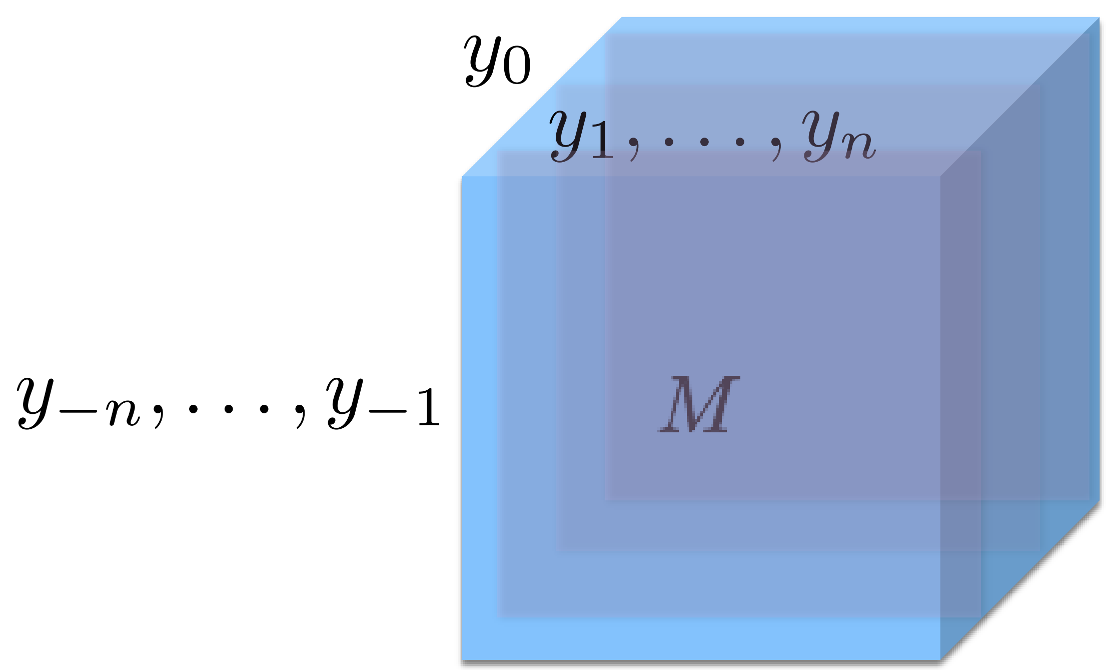
$$\{y_{-n}, \ldots, y_{-1}\} \perp \{y_0, \ldots, y_{n-1}\} \Big| x_0$$

Minimal HMM realization

- HMM conditional independence : probabilities in product form
- Exploit the uniqueness of tensor decomposition



$$\{y_{-n}, \ldots, y_{-1}\} \perp \{y_0, \ldots, y_{n-1}\} \Big| x_0$$



$$\{y_{-n}, \ldots, y_{-1}\} \perp \{y_0\} \perp \{y_1, \ldots, y_n\} \Big| x_0$$

Minimal HMM realization

- **Input:** $\mathcal{S}^{(N)} = \{\mathbb{P}_{y_{-n},\cdots,y_n}\}$ $\quad N = 2n+1$
- **Output:** a minimal HMM model $\theta^h = (d, k, Q, O)$
- **Key:**

Arrange $\mathcal{S}^{(N)}$ into a three way tensor

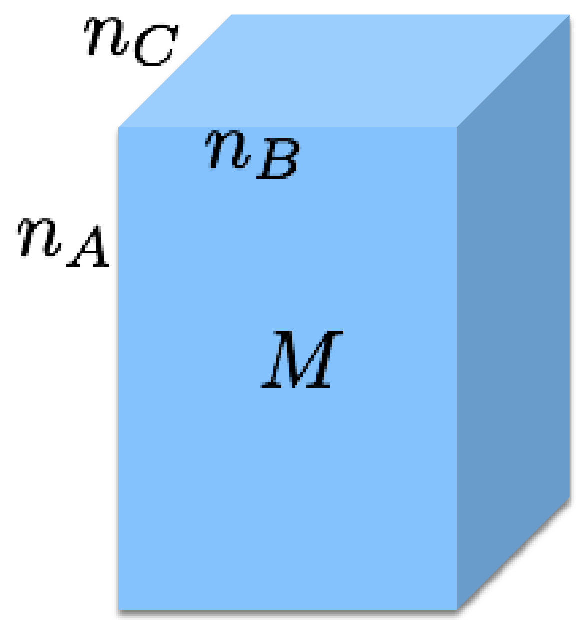
$$M \in \mathbb{R}^{d^n \times d^n \times d}$$

Tensor rank decomposition

recover minimal HMM realization up to state relabeling

Preliminaries on Tensor

- Tensor definition
  - Multi-way array
  - Multi-linear mapping



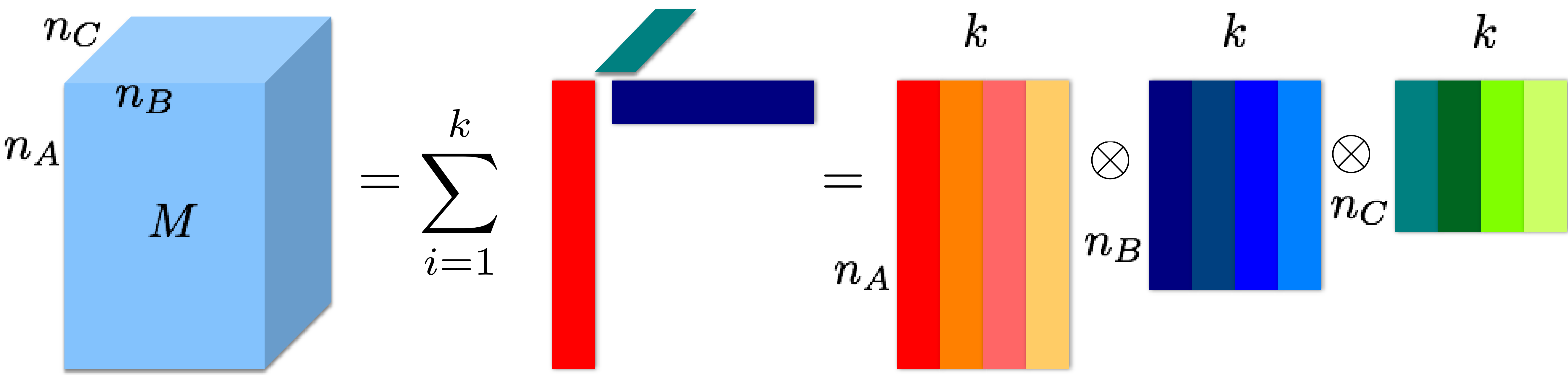
$$M_{j_1,j_2,j_3}, \quad j_1 \in [n_A], j_2 \in [n_B], j_3 \in [n_C]$$

Compare to matrix H_{j_1,j_2}

Preliminaries on Tensor

- Tensor rank decomposition (CP/PARAFAC)
  - Sum of rank one tensors $[\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c}]_{j_1,j_2,j_3} = a_{j_1} b_{j_2} c_{j_3}$

$$M = \sum_{i=1}^{k} A_{[:,i]} \otimes B_{[:,i]} \otimes C_{[:,i]} = A \quad \otimes \quad B \quad \otimes \quad C$$



Compare to matrix rank decomposition $H = U \otimes V = \sum_{i}^{k} U_{[:,i]} V_{[:,i]}^{\top}$

Minimal HMM realization (intuition)

- **Key:** $\mathcal{S}^{(N)}$ arranged into a three way tensor $M \in \mathbb{R}^{d^n \times d^n \times d}$

$$M_{L(\mathbf{l}_{-1}^{-n}),\ L(\mathbf{l}_1^n),\ l_0} = \mathbb{P}\Big(\mathbf{y}_{-n}^n = \mathbf{l}_{-n}^n\Big)$$

M relates to the model parameters via unique tensor factorization

$$M = A \otimes B \otimes C$$

$A, B \in \mathbb{R}^{d^n \times k}$ and $C \in \mathbb{R}^{d \times k}$

$$A = \mathbb{P}\Big(y_1, y_2, \ldots, y_n \Big| x_0\Big)$$

$$B = \mathbb{P}\Big(y_{-1}, y_{-2}, \ldots, y_{-n} \Big| x_0\Big)$$

$$C = \mathbb{P}\Big(y_0, x_0\Big)$$

Minimal HMM realization (intuition)

- **Key:** $\mathcal{S}^{(N)}$ arranged into a three way tensor $M \in \mathbb{R}^{d^n \times d^n \times d}$

$$M_{L(\mathbf{l}_{-1}^{-n}),\ L(\mathbf{l}_1^n),\ l_0} = \mathbb{P}\Big(\mathbf{y}_{-n}^{n} = \mathbf{l}_{-n}^{n}\Big)$$

M relates to the model parameters via unique tensor factorization

$$M = A \otimes B \otimes C$$

$A, B \in \mathbb{R}^{d^n \times k}$ and $C \in \mathbb{R}^{d \times k}$

$$A = \mathbb{P}\Big(y_1, y_2, \ldots, y_n \Big| x_0\Big)$$

$$B = \mathbb{P}\Big(y_{-1}, y_{-2}, \ldots, y_{-n} \Big| x_0\Big)$$

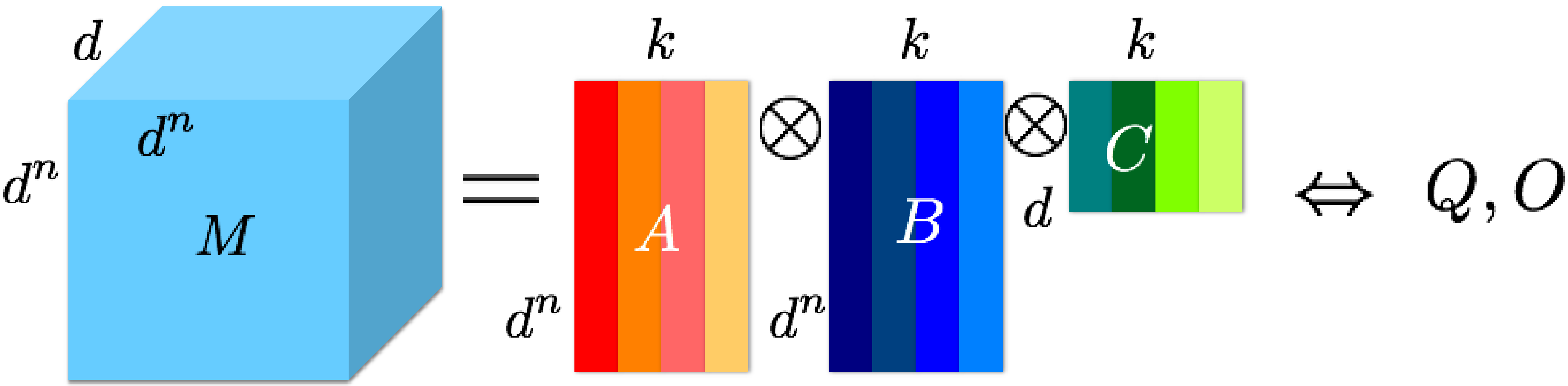
$$C = \mathbb{P}\Big(y_0, x_0\Big)$$

$$A = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{Q) \ldots)Q)Q)Q}_{n},$$

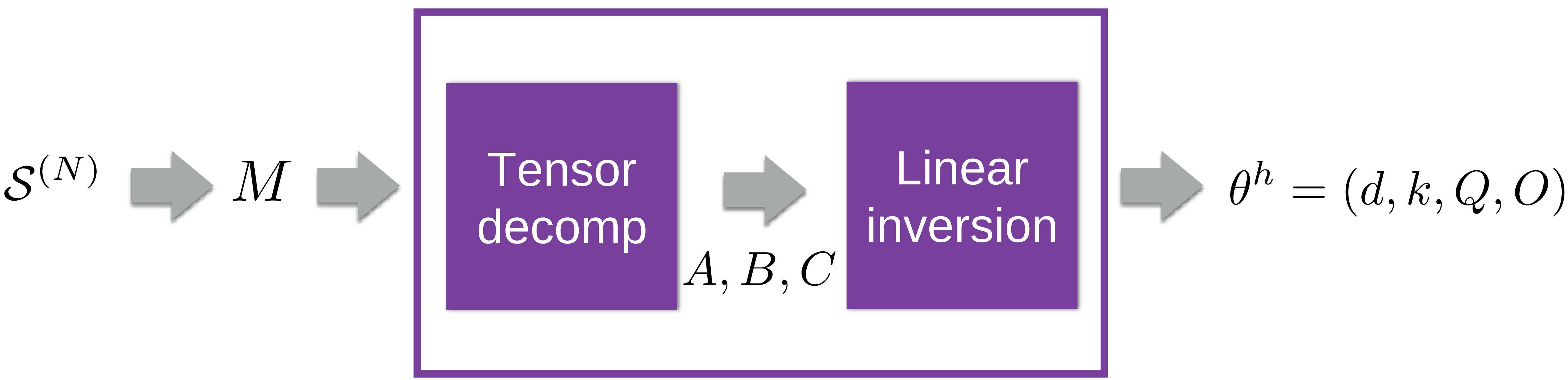
$$B = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{\widetilde{Q}) \ldots)\widetilde{Q})\widetilde{Q})\widetilde{Q}}_{n},$$

$$C = O diag(\pi)$$

Minimal HMM realization (two-step approach)



Two-step realization algorithm



Minimal HMM realization (identifiability)

$$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$$

Minimal HMM realization (identifiability)

$$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$$

Proposition 2: sufficient conditions for invertibility of (1)

$$krank(A) + krank(B) + krank(C) \geq 2k + 2,$$

Minimal HMM realization (identifiability)

$$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$$

Theorem 3: sufficient conditions for invertibility of (1)

$$rank(A) = rank(B) = k$$

Minimal HMM realization (identifiability)

N ➡ $\mathcal{S}^{(N)}$ ➡ $(1) \quad M \Leftrightarrow A, B, C$ ➡ $(2) \quad A, B, C \Leftrightarrow (Q, O)$

Theorem 3: sufficient conditions for invertibility of (1)

$$rank(A) = rank(B) = k$$

Theorem:
For almost all HMMs

$$\mathbf{n > 2\log_d(k)}$$

guarantees A,B full rank

Minimal HMM realization (identifiability)

$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$

Theorem 2: sufficient conditions for invertability of (2)

1. If A has full column rank
2. Or if O has full column rank

Minimal HMM realization (algorithm)

N ➡ $\mathcal{S}^{(N)}$ ➡ (1) $M \Leftrightarrow A, B, C$ ➡ (2) $A, B, C \Leftrightarrow (Q, O)$

Theorem 2: algorithm for inverting (2)

Obtain the observation probabilities by: $O_{[:,i]} = C_{[:,i]}/(\mathbf{e}'C_{[:,i]}), \quad \forall i \in [k]$.
Given the matrix A, i.e., $\mathbb{P}(\mathbf{y}_1^n|x_0)$, we can marginalize to obtain the matrices:

$$\widetilde{A}^{(1)} = \mathbb{P}(y_1 | x_0) \quad \widetilde{A}^{(n-1)} = \mathbb{P}(\mathbf{y}[1:n-1] | x_0),$$

1. If A has full column rank: $Q = (O \odot \widetilde{A}^{(n-1)})^{-1} A'$;
2. if O has full column rank: $Q = O^{\dagger} \widetilde{A}^{(1)}$.

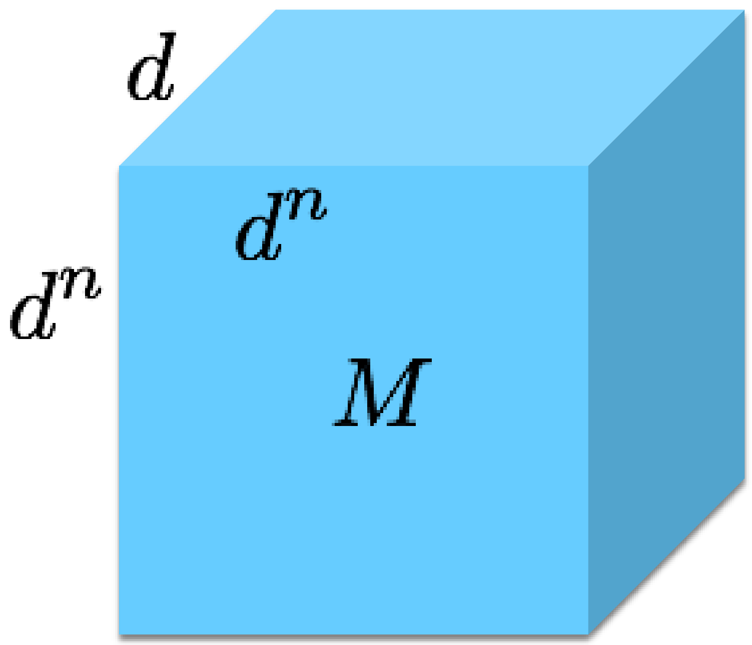
Minimal HMM realization (algorithm)

$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$

- Unique tensor decomposition to invert (1)
  - In general tensor decomposition is hard (Alternating least square)

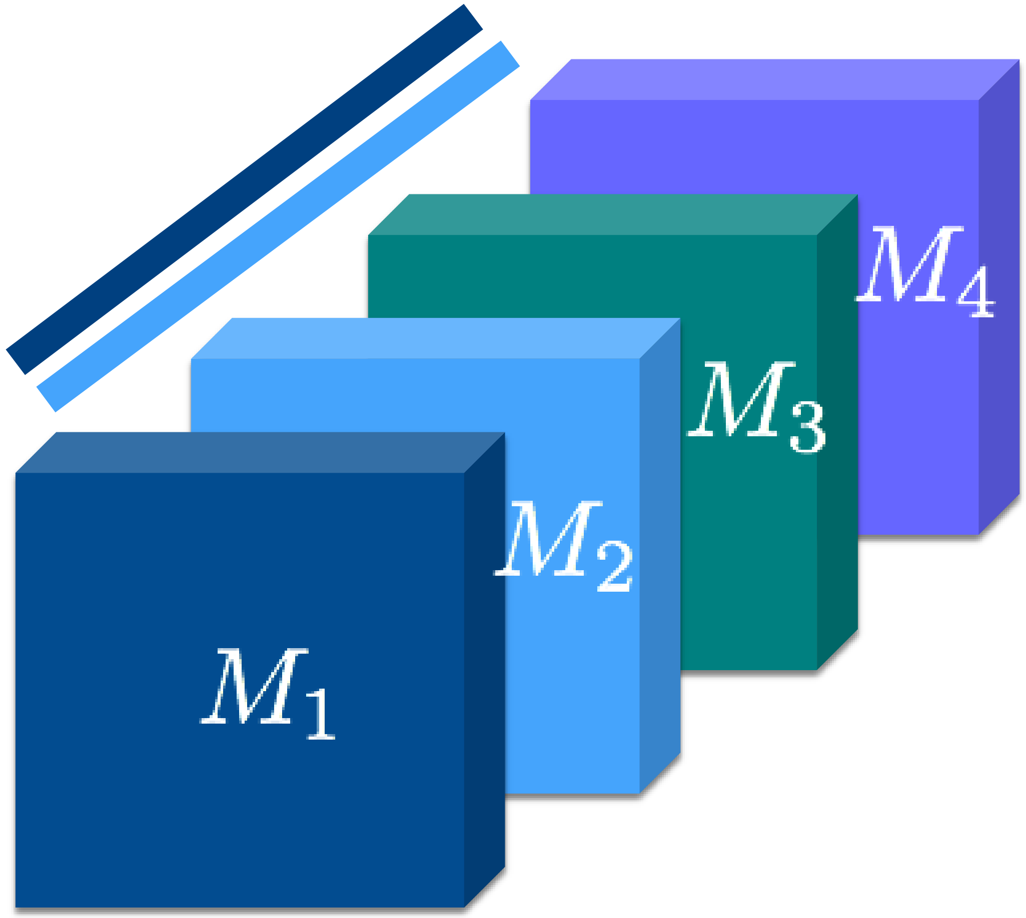
- Have efficient algorithm if
  - A and B have full column rank ← Algorithm 2
  - C is full rank, A⊙B is full rank and… ← Algorithm 3
  - The two cases apply to *almost all* HMMs for large enough N

Algorithm 2 (Simultaneous diagonalization)

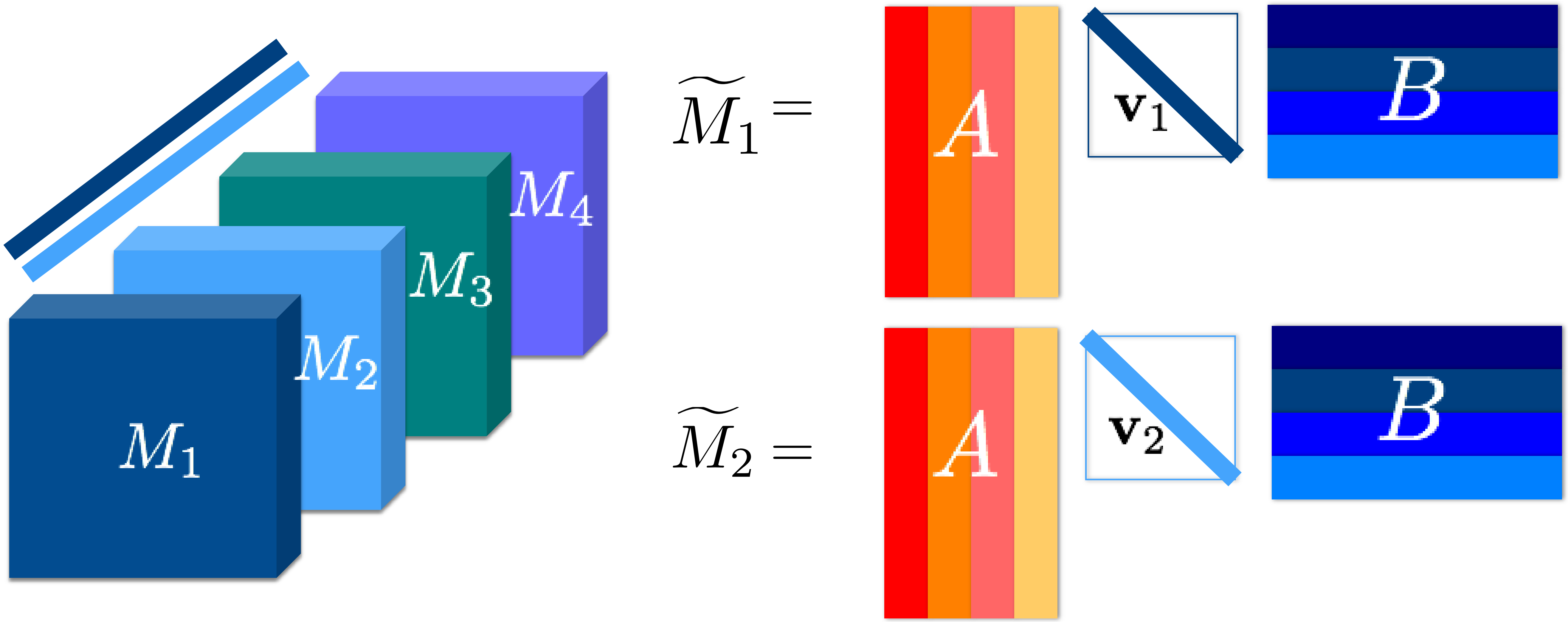


- Take 2 random projections along the 3rd dim

Algorithm 2 (Simultaneous diagonalization)

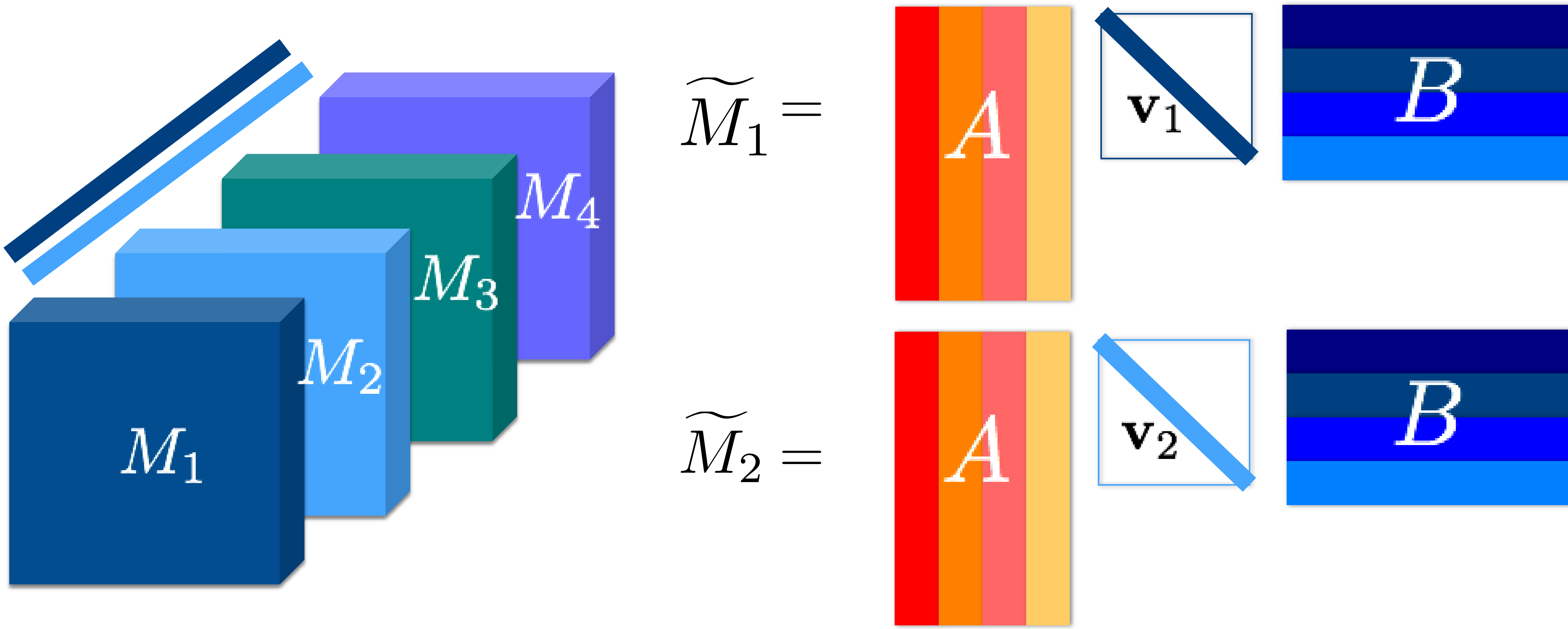


Algorithm 2 (Simultaneous diagonalization)



- Take 2 random projections along the 3rd dimension→ 2 matrix slices
- Simultaneous diagonalization is unique almost surely

Algorithm 2 (Simultaneous diagonalization)



- Take 2 random projections along the 3rd dimension

$$\widetilde{M}_1 = M(I_{d^n \times d^n}, I_{d^n \times d^n}, \mathbf{v}_1), \quad \widetilde{M}_2 = M(I_{d^n \times d^n}, I_{d^n \times d^n}, \mathbf{v}_2).$$

$$\begin{aligned} \widetilde{M}_1 \widetilde{M}_2^{-1} &= A \Lambda A^{-1}, \\ \widetilde{M}_2 \widetilde{M}_1^{-1} &= B \Lambda^{-1} B^{-1}. \end{aligned} \Rightarrow \text{ pair up eigenvalues to get } A, B$$

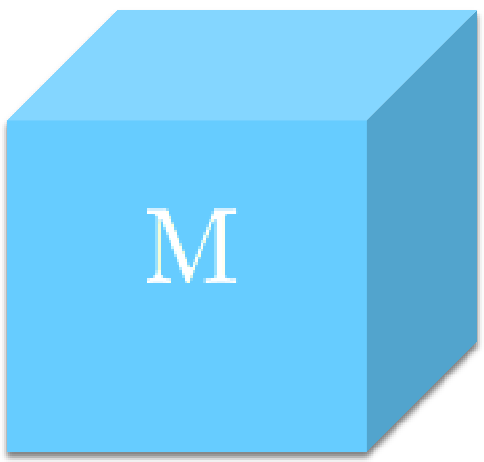
solve linear eqn to obtain C

Algorithm 3 (FOOBI)

$$(1) \quad M = A \otimes B \otimes C$$

A,B are rank degenerate
But if $A \odot B$ has full column rank

- Degenerate case: Q is not full rank, set N = 3



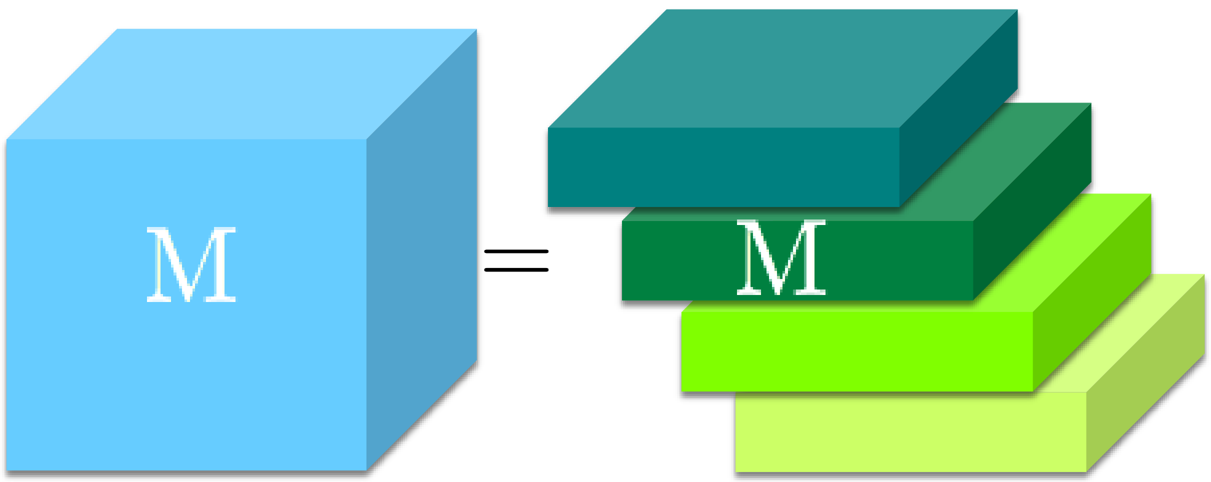
$=$

Algorithm 3 (FOOBI)

$$(1) \quad M = A \otimes B \otimes C$$

A,B are rank degenerate
But if $A \odot B$ has full column rank

- Degenerate case: Q is not full rank, set N = 3
- Matricization M along the 3rd dim: $\overline{M}$

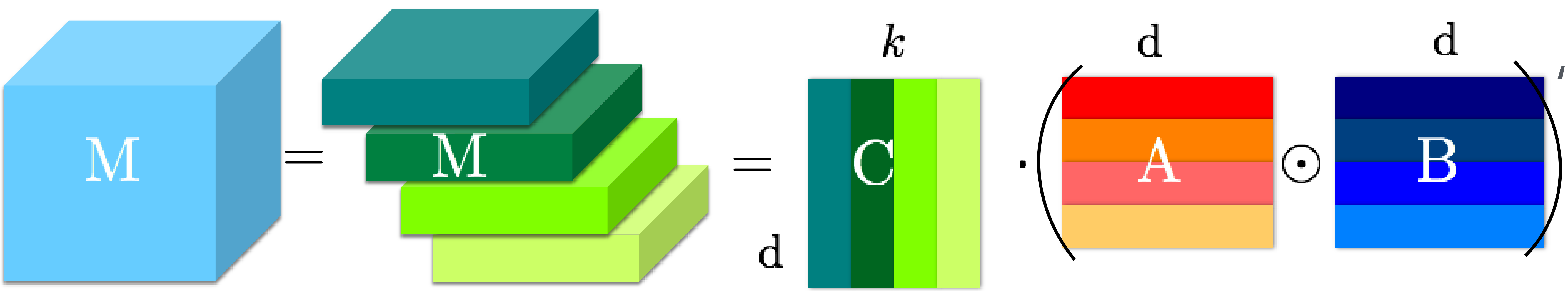


Algorithm 3 (FOOBI)

$$(1) \quad M = A \otimes B \otimes C$$

A,B are rank degenerate
But if $A \odot B$ has full column rank

- Degenerate case: Q is not full rank, set N = 3
- Matricization M along the 3rd dim: $\overline{M}$



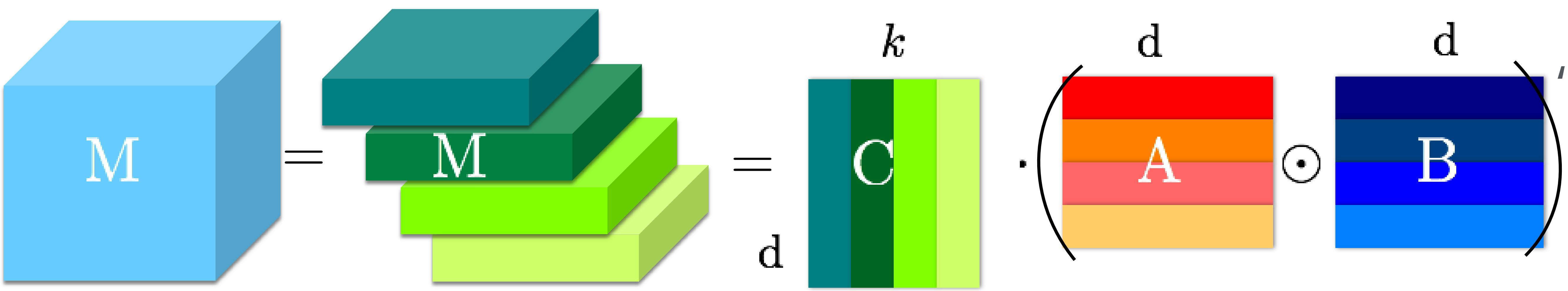
$$\overline{M} = C \otimes (A \odot B)'$$

Algorithm 3 (FOOBI)

$$(1) \quad M = A \otimes B \otimes C$$

A,B are rank degenerate
But if A $\odot$ B has full column rank

- Degenerate case: Q is not full rank, set N = 3
- Matricization M along the 3rd dim: $\overline{M}$



$$\overline{M} = C \otimes (A \odot B)'$$

rank decomposition $\overline{M} = UV'$

such that: each column of V is a rank 1 matrix

Can be converted to solving linear equations and simultaneous diagonalization

Algorithm 3 (correctness)

$$\Theta^h_{(d,k,r)} = \{\theta^h : \text{ order } k, \text{ alphabet size } d, Q \text{ is of rank } r < k\}$$

Theorem 4: Correctness of Algorithm 3

Pick a random instance from the model class $\Theta^h_{(d,k,r)}$

If Algorithm 3 works correctly to recover the minimal HMM

Then it works for almost all instances in $\Theta^h_{(d,k,r)}$

Minimal HMM realization (summary)

- Key to HMM realization:
  - conditional independence: **current, past** and **future**
  - tensor decomposition, unique up to column permutation

- For almost all HMM, minimal HMM realization is easy
  - Informational : $N \sim \log_d(k)$
  - Computational : simultaneous diagonalization $\mathcal{O}(dk^6)$
- A class of degenerate cases can also be efficiently realized

Minimal HMM realization (summary)

- Key to HMM realization:
  - conditional independence: **current, past** and **future**
  - tensor decomposition, unique up to column permutation

- For almost all HMM, minimal HMM realization is easy
  - Informational : $N \sim \log_d(k)$
  - Computational : simultaneous diagonalization $\mathcal{O}(dk^6)$
- A class of degenerate cases can also be efficiently realized

Summary & Discussions

Discussions

- Hardness of worst cases
  - ✓ Reduction to parity of noise
- Learning via realization algorithms
  - ✓ Statistical complexity
- Ongoing works
  - ✓ Smoothed analysis
  - ✓ Agnostic learning
  - ✓ HMM model reduction

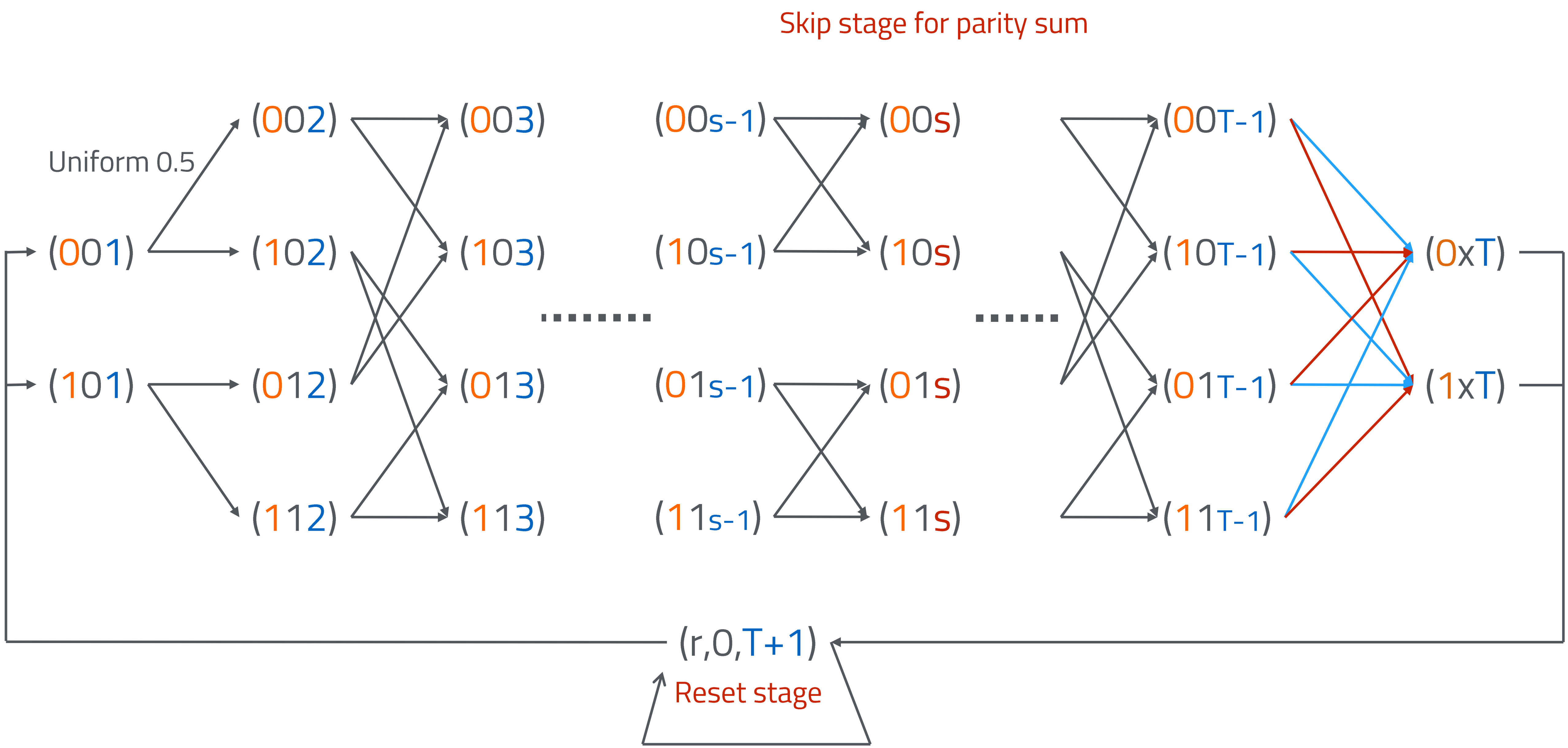
Hardness results

- We showed for almost all HMM, operator model and HMM model both are easy to learn (poly time algorithm, poly sample complexity)

- There exist information theoretic hard cases [Abe,Warmuth] [Kearns]
  - ✓ Given order k, cannot learn $\theta^h = (d, k, Q, O)$ in poly time, unless RP=NP
  - ✓ Unknown k, HMM is not efficiently PAC learnable, under noisy parity assumption

- Example: reduction to parity of noise
  - ✓ Number of states = 4T-1, observation alphabet = 2
  - ✓ required window size = T
  - ✓ There exists an η such that there is no efficient algorithm for learning parity under uniform distribution in the PAC framework with classification noise η

State = (Emission, Parity sum, Stage)



HMM learning problems

- Observe: sample sequences of the output process of an HMM
- Goal: an operator / HMM model to fit the process

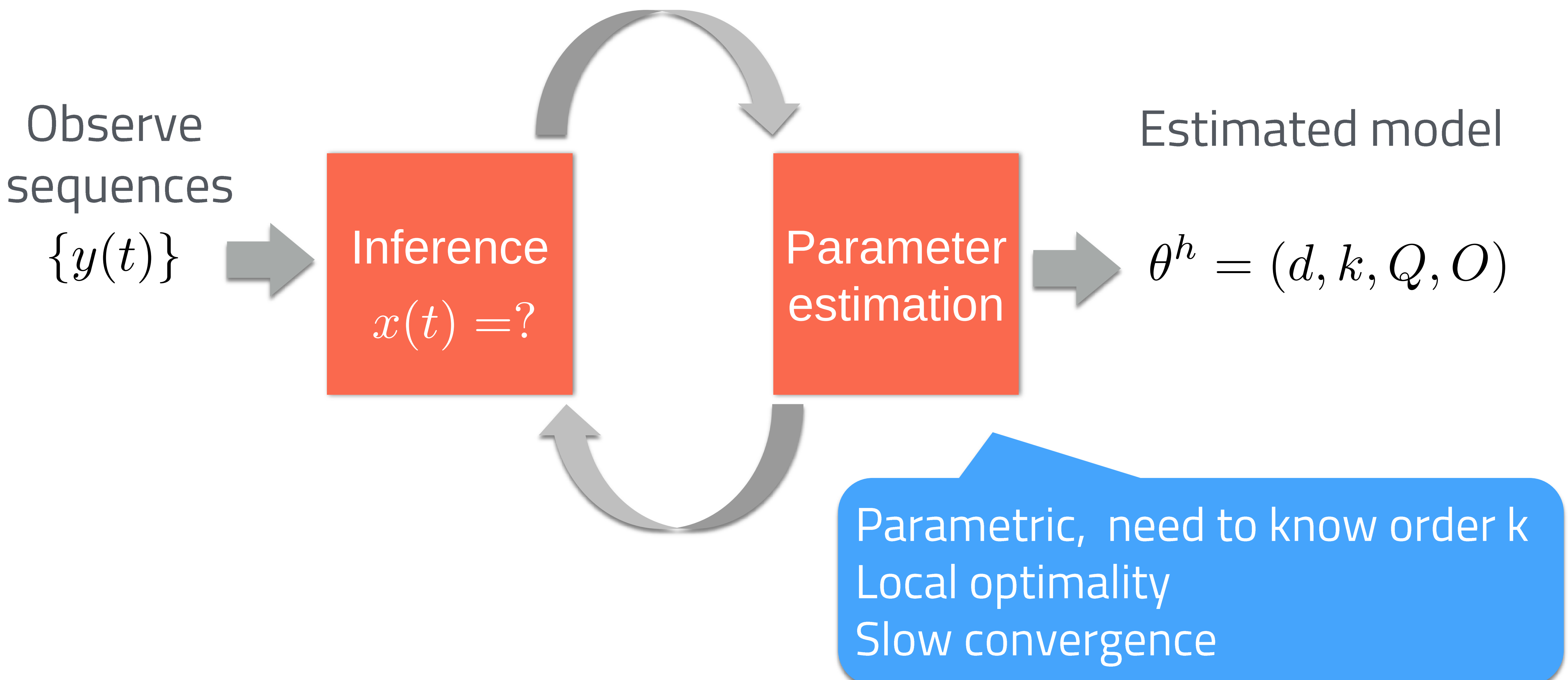
Maximal likelihood estimator

- Old art: EM (Expectation-Maximization)
- New art: Spectral method (subspace id/ automata / tensor)

Statistics matching

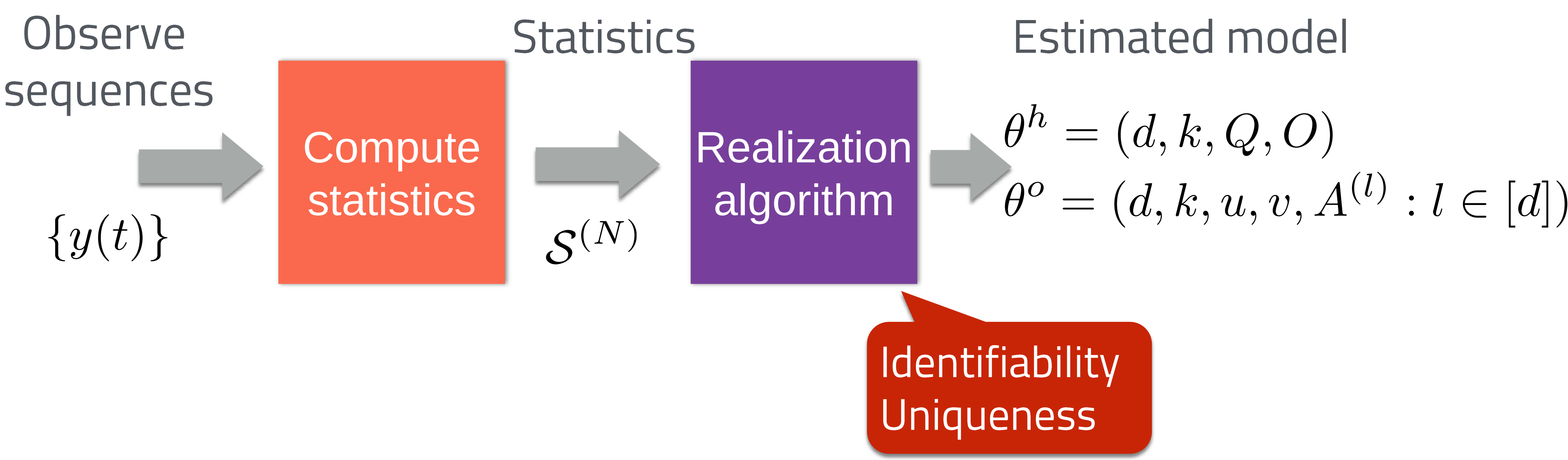
HMM learning problems

- Observe: sample sequences of the output process of an HMM
- Goal: an operator / HMM model to fit the process
- **Old art:** EM



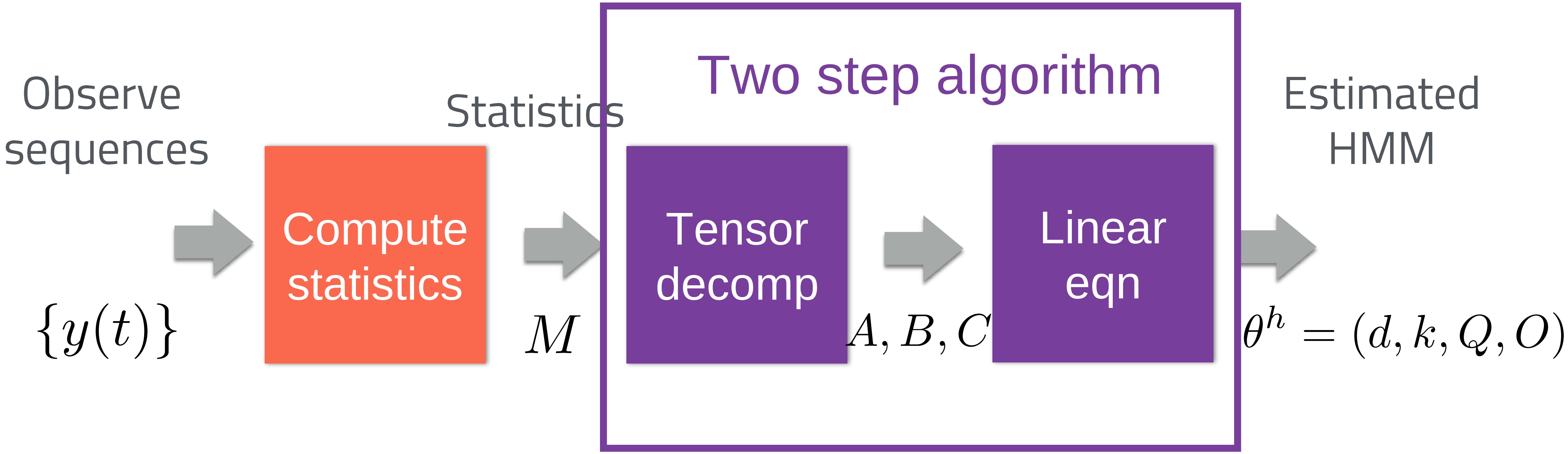
HMM learning problems

- Observe: sample sequences of the output process of an HMM
- Goal: an operator / HMM model to fit the process
- **New art:** spectral method



Minimal HMM learning

- Observe: sample sequences of the output process of an HMM
- Goal: an operator / HMM model to fit the process
- **New art:** spectral method



Sample complexity

Ongoing works

- Smoothed analysis – perturbation based

  Understand the polynomial time algorithm in practice

- Agnostic HMM learning

  If the underlying model is not a HMM…

  which class of models have better fitting power?

- HMM model reduction

Appendix

Quasi-HMM realization (correctness of the algorithm)

Proposition 1. *Let $\theta^o = (d, k, u, v, A^{(j)} : j \in [d])$ be a minimal order quasi realization for the output process. If the matrix $H^{(0)}$ has rank k, then the Algorithm returns an operator model $\widetilde{\theta}^o$ that is equivalent to θ^o up to linear transformation.*

Assume the output process can be realized by an HMM model of order k^h. The order k^h is always lower bounded by the order of the minimal order quasi realization k.

- Sketch of the proof

Quasi-HMM realization (correctness of the algorithm)

Proposition 1. *Let $\theta^o = (d, k, u, v, A^{(j)} : j \in [d])$ be a minimal order quasi realization for the output process. If the matrix $H^{(0)}$ has rank k, then the Algorithm returns an operator model $\widetilde{\theta}^o$ that is equivalent to θ^o up to linear transformation.*

Assume the output process can be realized by an HMM model of order k^h. The order k^h is always lower bounded by the order of the minimal order quasi realization k.

- Sketch of the proof

$H^{(0)} = UV'$, $\quad U, V$ both of full rank k

$$\boxed{\begin{aligned} \widetilde{A}^{(j)} &= T^{-1}AT \\ \widetilde{u} &= T'u \\ \widetilde{v} &= T^{-1}v \end{aligned}}$$

$$U = \begin{bmatrix} u'A^{(1)} \cdots A^{(1)} \\ u'A^{(1)} \cdots A^{(2)} \\ \vdots \\ u'A^{(d)} \cdots A^{(d)} \end{bmatrix} T = \begin{bmatrix} \widetilde{u}'\widetilde{A}^{(1)} \cdots \widetilde{A}^{(1)} \\ \widetilde{u}'\widetilde{A}^{(1)} \cdots \widetilde{A}^{(2)} \\ \vdots \\ \widetilde{u}'\widetilde{A}^{(d)} \cdots \widetilde{A}^{(d)} \end{bmatrix}$$

$$V' = T^{-1}\left[A^{(1)} \cdots A^{(1)}v, \cdots, A^{(d)} \cdots A^{(d)}v\right] = \left[\widetilde{A}^{(1)} \cdots \widetilde{A}^{(1)}v, \cdots, \widetilde{A}^{(d)} \cdots \widetilde{A}^{(d)}\widetilde{v}\right]$$

Quasi-HMM realization (correctness of the algorithm)

Proposition 1. *Let $\theta^o = (d, k, u, v, A^{(j)} : j \in [d])$ be a minimal order quasi realization for the output process. If the matrix $H^{(0)}$ has rank k, then the Algorithm returns an operator model $\widetilde{\theta}^o$ that is equivalent to θ^o up to linear transformation.*

Assume the output process can be realized by an HMM model of order k^h. The order k^h is always lower bounded by the order of the minimal order quasi realization k.

- Sketch of the proof

Unique up to linear transformation

$$H^{(0)} = UV', \quad U, V \text{ both of full rank } k$$

$$\boxed{\begin{aligned} \widetilde{A}^{(j)} &= T^{-1}AT \\ \widetilde{u} &= T'u \\ \widetilde{v} &= T^{-1}v \end{aligned}}$$

$$U = \begin{bmatrix} u'A^{(1)} \dots A^{(1)} \\ u'A^{(1)} \dots A^{(2)} \\ \vdots \\ u'A^{(d)} \dots A^{(d)} \end{bmatrix} T = \begin{bmatrix} \widetilde{u}'\widetilde{A}^{(1)} \dots \widetilde{A}^{(1)} \\ \widetilde{u}'\widetilde{A}^{(1)} \dots \widetilde{A}^{(2)} \\ \vdots \\ \widetilde{u}'\widetilde{A}^{(d)} \dots \widetilde{A}^{(d)} \end{bmatrix}$$

$$V' = T^{-1}\left[A^{(1)} \dots A^{(1)}v, \cdots, A^{(d)} \dots A^{(d)}v\right] = \left[\widetilde{A}^{(1)} \dots \widetilde{A}^{(1)}v, \cdots, \widetilde{A}^{(d)} \dots \widetilde{A}^{(d)}\widetilde{v}\right]$$

Quasi-HMM realization (correctness of the algorithm)

Proposition 1. *Let $\theta^o = (d, k, u, v, A^{(j)} : j \in [d])$ be a minimal order quasi realization for the output process. If the matrix $H^{(0)}$ has rank k, then the Algorithm returns an operator model $\widetilde{\theta}^o$ that is equivalent to θ^o up to linear transformation.*

Assume the output process can be realized by an HMM model of order k^h. The order k^h is always lower bounded by the order of the minimal order quasi realization k.

- Sketch of the proof

$H^{(0)} = UV'$, $\boxed{U, V \text{ both of full rank } k}$

$$\widetilde{A}^{(j)} = T^{-1}AT \qquad \widetilde{u} = T'u \qquad \widetilde{v} = T^{-1}v$$

$$U = \begin{bmatrix} u'A^{(1)} \dots A^{(1)} \\ u'A^{(1)} \dots A^{(2)} \\ \vdots \\ u'A^{(d)} \dots A^{(d)} \end{bmatrix} \quad T =$$

Computation complexity $\mathcal{O}\left(d(d^n)^3\right)$

Want poly time algorithm $n \sim \mathcal{O}\left(\log_d k\right)$

$$V' = T^{-1}\left[A^{(1)} \dots A^{(1)}v, \cdots, A^{(d)} \dots A^{(d)}v\right] = \left[\widetilde{A}^{(1)} \dots \widetilde{A}^{(1)}v, \cdots, \widetilde{A}^{(d)} \dots \widetilde{A}^{(d)}\widetilde{v}\right]$$

Algorithm 2 (correctness)

Theorem 3:

$$(1) \quad M = A \otimes B \otimes C$$

If both A and B have full column rank, then with probability one (over random projections), Algorithm 1 uniquely recovers the factors A, B, C up to state permutation. Moreover, Theorem 2.1 can be applied to recover (Q, O).

Algorithm 2 (correctness)

Theorem 3: $$(1) \quad M = A \otimes B \otimes C$$

If both A and B have full column rank, then with probability one (over random projections), Algorithm 1 uniquely recovers the factors A, B, C up to state permutation. Moreover, Theorem 2.1 can be applied to recover (Q, O).

$$A = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{Q) \ldots)Q)Q)Q}_{n},$$

$$B = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{\widetilde{Q}) \ldots)\widetilde{Q})\widetilde{Q})\widetilde{Q}}_{n},$$

$$C = Odiag(\pi)$$

Algorithm 2 (correctness)

Theorem 3: $$M = A \otimes B \otimes C \tag{1}$$

If both A and B have full column rank, then with probability one (over random projections), Algorithm 1 uniquely recovers the factors A, B, C up to state permutation. Moreover, Theorem 2.1 can be applied to recover (Q, O).

$$A = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{Q) \ldots)Q)Q)Q}_{n},$$

$$B = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{\widetilde{Q}) \ldots)\widetilde{Q})\widetilde{Q})\widetilde{Q}}_{n},$$

$$C = Odiag(\pi)$$

Q needs to be of full rank k

Increasing n can boost the rank of A,B

How large N=2n+1 needs to be?

Algorithm 2 (correctness)

Theorem 3: (1) $M = A \otimes B \otimes C$

If both A and B have full column rank, then with probability one (over random projections), Algorithm 1 uniquely recovers the factors A, B, C up to state permutation. Moreover, Theorem 2.1 can be applied to recover (Q, O).

$$\Theta^h_{(d,k)} = \{\theta^h : \text{ order } k, \text{ alphabet size } d\}$$

There exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,k)}$ such that for all HMMs in the set $\Theta^h_{(d,k)} \backslash \mathcal{E}$, for $N = 2n + 1$, with

$$n > 2 \log_d(k),$$

the matrices A and B have full column rank, thus $\mathcal{S}^{(N)}$ is sufficient for learning the minimal order HMM of the output process.

Algorithm 2 (correctness)

Theorem 3: (1) $M = A \otimes B \otimes C$

If both A and B have full column rank, then with probability one (over random projections), Algorithm 1 uniquely recovers the factors A, B, C up to state permutation. Moreover, Theorem 2.1 can be applied to recover (Q, O).

Polynomial computational complexity

$$\mathcal{O}(d(d^n)^3) \sim \mathcal{O}(dk^6)$$

$\Theta^h_{(d,k)} = \{\theta^h :$ order k, alphabet

There exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,k)}$ that for all HMMs in the set $\Theta^h_{(d,k)} \backslash \mathcal{E}$, for $N = 2n+1$, with

$$\boxed{n > 2\log_d(k),}$$

the matrices A and B have full column rank, thus $\mathcal{S}^{(N)}$ is sufficient for learning the minimal order HMM of the output process.

Minimal HMM realization (identifiability)

N ➡ $\mathcal{S}^{(N)}$ ➡ $(1) \quad M \Leftrightarrow A, B, C$ ➡ $(2) \quad A, B, C \Leftrightarrow (Q, O)$

Proposition 2: sufficient conditions for invertability of (1)

Consider the matrices A, B, C parameterized by a minimal realization (Q, O). If the following deterministic conditions on the Kruskal rank are satisfied:

$$krank(A) + krank(B) + krank(C) \geq 2k + 2, \tag{1}$$

then M can be uniquely (up to common column permutation) decomposed into A, B, C, and k can be determined thereby.

Minimal HMM realization (identifiability)

$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$

Proposition 2: sufficient conditions for invertability of (1)

Consider the matrices A, B, C parameterized by a minimal realization (Q, O). If the following deterministic conditions on the Kruskal rank are satisfied:

$$krank(A) + krank(B) + krank(C) \geq 2k + 2, \tag{1}$$

then M can be uniquely (up to common column permutation) decomposed into A, B, C, and k can be determined thereby.

$$A = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{Q) \ldots)Q)Q)Q}_{n},$$

$$B = \underbrace{(O \odot (O \odot (O \odot \ldots (O \odot O}_{n} \underbrace{\widetilde{Q}) \ldots)\widetilde{Q})\widetilde{Q})\widetilde{Q}}_{n},$$

$$C = O diag(\pi)$$

Algorithm 4 (random instance check)

$\Theta^h_{(d,k,r)} = \{\theta^h : \text{ order } k, \text{ alphabet size } d, Q \text{ is of rank } r < k\}$

1. Randomly pick an HMM model $(d, k, Q, O) \in \Theta^h_{(d,k,r)}$.
2. Construct matrices $A = OQ, \quad B = O\widetilde{Q}, \quad C = ODiag(\pi)$, where π is the stationary distribution, and $\widetilde{Q} = Diag(\pi)Q'Diag(\pi)^{-1}$.
3. Run Algorithm 2.a with input $M = A \otimes B \otimes C$.
4. Return "yes" if the algorithm recovers A, B, C up to state permutation, and "no" otherwise.

Algorithm 4 (random instance check)

$\Theta^h_{(d,k,r)} = \{\theta^h :\ \text{order } k,\ \text{alphabet size } d, Q \text{ is of rank } r < k\}$

1. Randomly pick an HMM model $(d, k, Q, O) \in \Theta^h_{(d,k,r)}$.
2. Construct matrices $A = OQ, \quad B = O\widetilde{Q}, \quad C = ODiag(\pi)$, where π is the stationary distribution, and $\widetilde{Q} = Diag(\pi)Q'Diag(\pi)^{-1}$.
3. Run Algorithm 2.a with input $M = A \otimes B \otimes C$.
4. Return "yes" if the algorithm recovers A, B, C up to state permutation, and "no" otherwise.

Theorem 4: Correctness of Algorithm 3

For given $d \geq k$ and $n = 1$, if Algorithm 2.b returns "yes", then there exists a measure zero set $\mathcal{E} \in \Theta^h_{(d,r,k)}$ such that Algorithm 2.a can identify all minimal HMM realizations in the set $\Theta^h_{(d,r,k)} \backslash \mathcal{E}$.

Moreover, if the latter is true, Algorithm 2.b returns "yes" with probability 1. Theorem 1 (2) can be applied to recover (Q, O) for this class of HMMs.

Minimal HMM realization (summary)

$$N \Rightarrow \mathcal{S}^{(N)} \Rightarrow (1) \quad M \Leftrightarrow A, B, C \Rightarrow (2) \quad A, B, C \Leftrightarrow (Q, O)$$

- **Identifiability:** if (1) (2) are bijective mappings
  - ✓ How large N is so that the minimal HMM is identifiable from $\mathcal{S}^{(N)}$?
- **Algorithms:** efficiently invert (1) (2)
  - ✓ For (1) : Algorithm 2 (Simultaneous diagnolization)

    If Q is full rank

    Algorithm 3 (Foobi)

    If O has full column rank, and check a random instance
  - ✓ For (2) : Theorem 2 (Linear inversion)

    If Q or O has full column rank