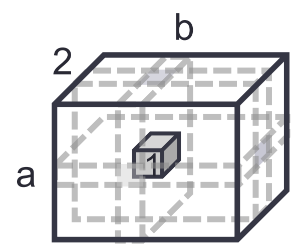
Tensor representation of data

- Most media and sensor data are
  - multi-dimensional and
  - multi-relational

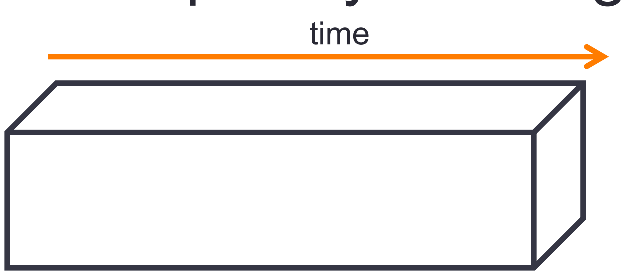
E.g.

A	B	C
:	:	:
a	b	2
:	:	:

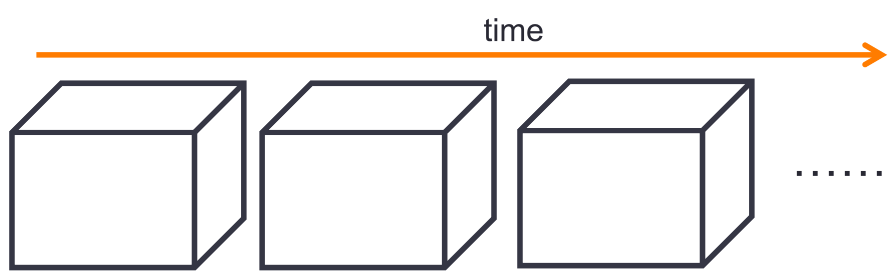
represented as



- Temporally evolving data…



Alternative #1: incrementally growing tensor



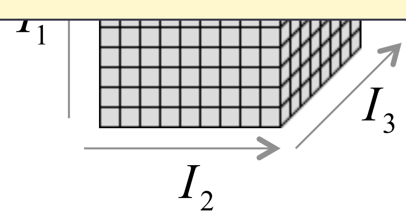
Alternative #2: sequence of tensor snapshots

Tensor Representation

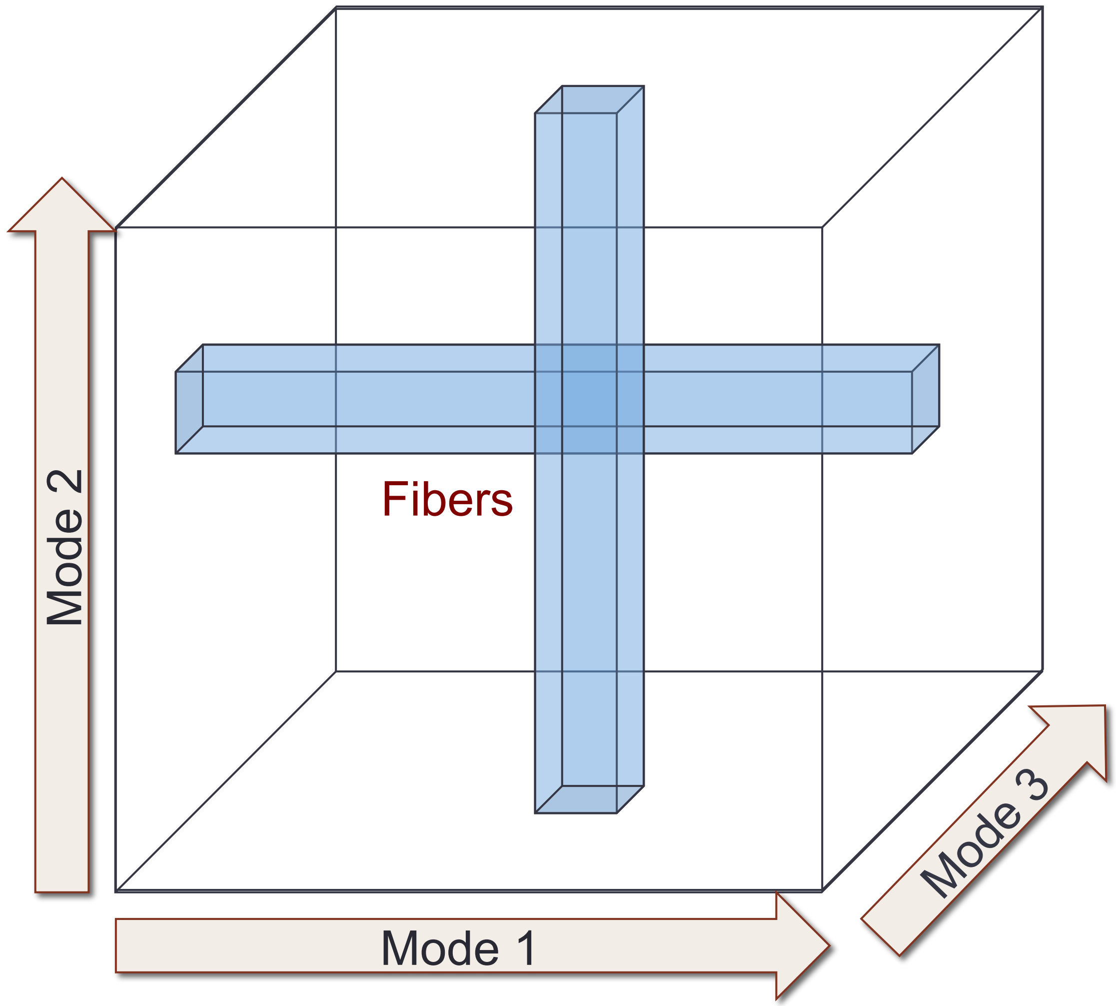
- Tensors are multidimensional arrays (generalization of

Can we decompose tensors to recover "latent" features or clusters?

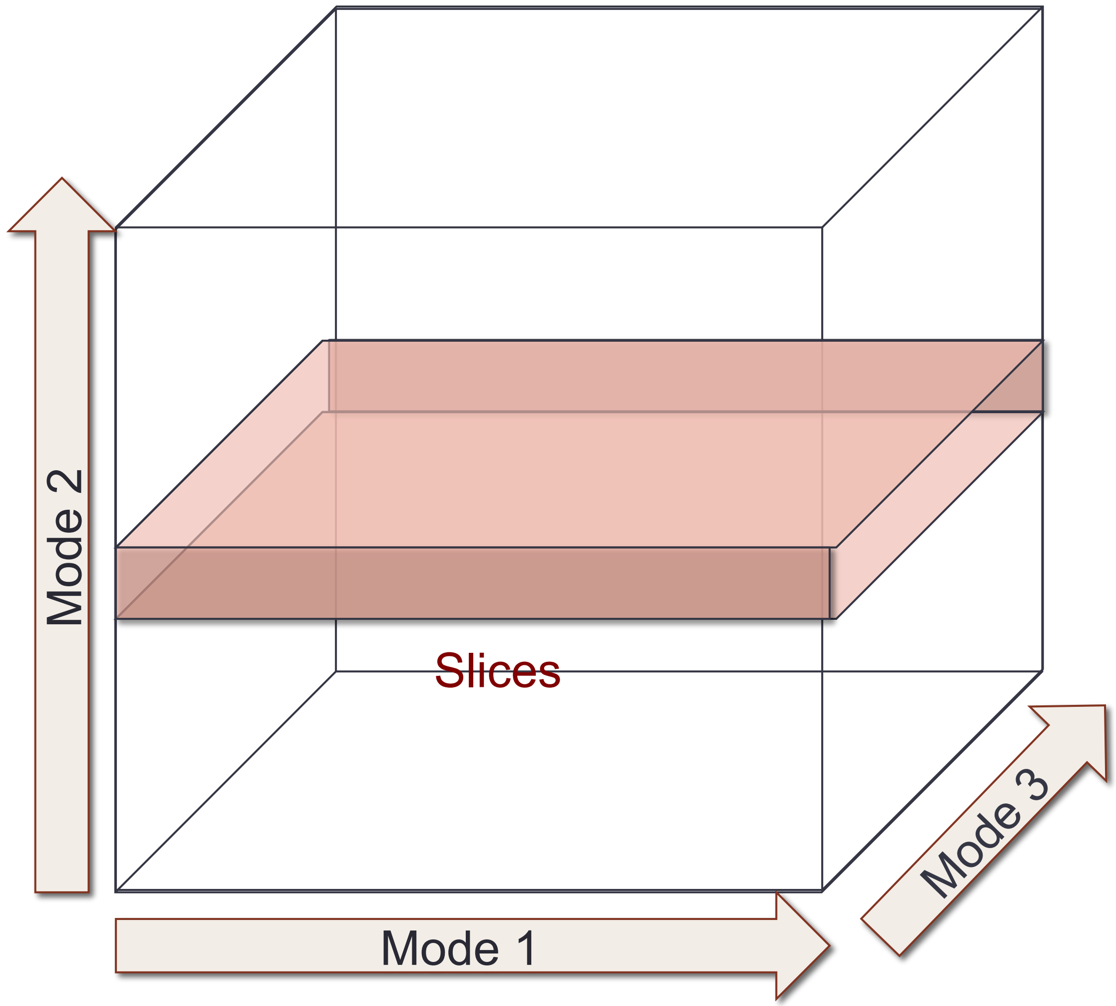
- hidden correlations among data
- clusters of data
- degree of contribution of each data element to relationships



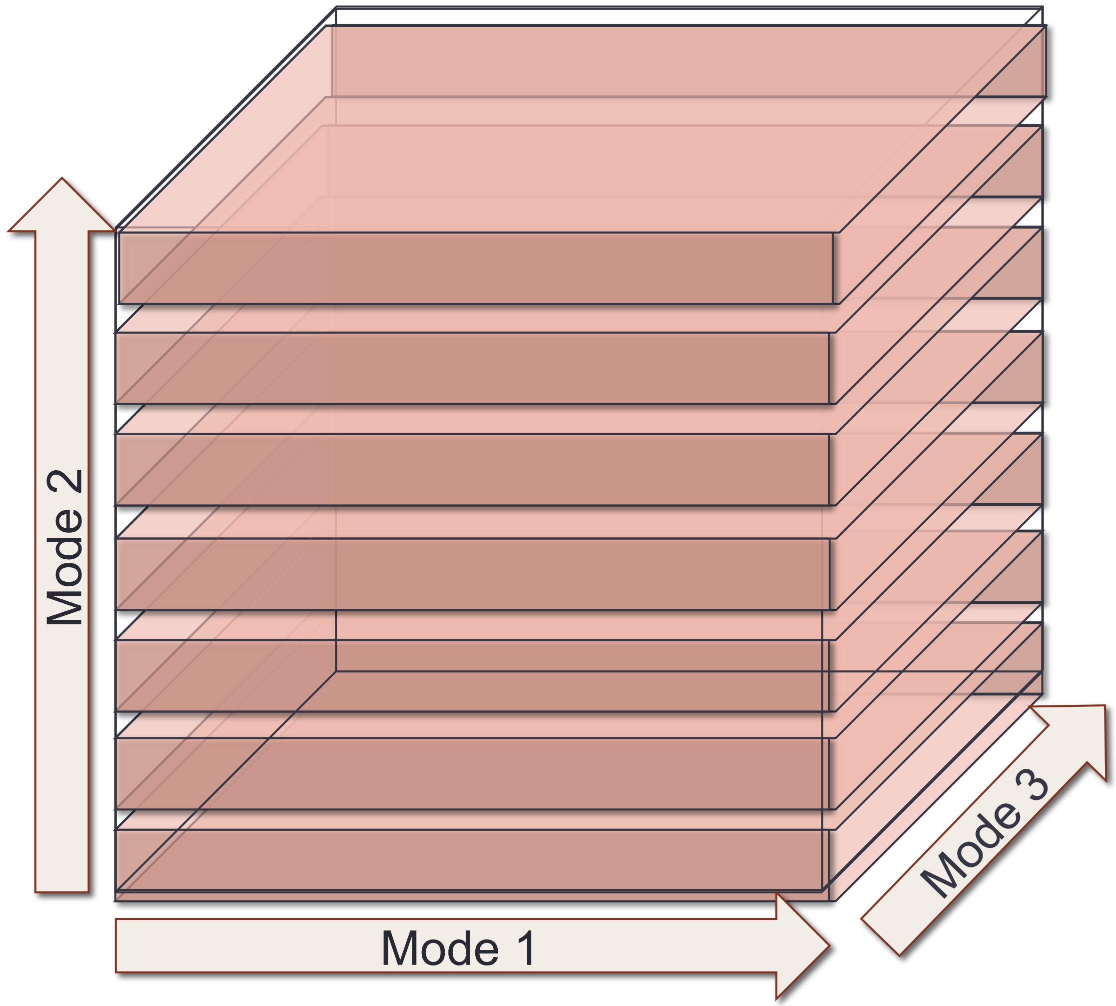
Some core concepts: Fibers



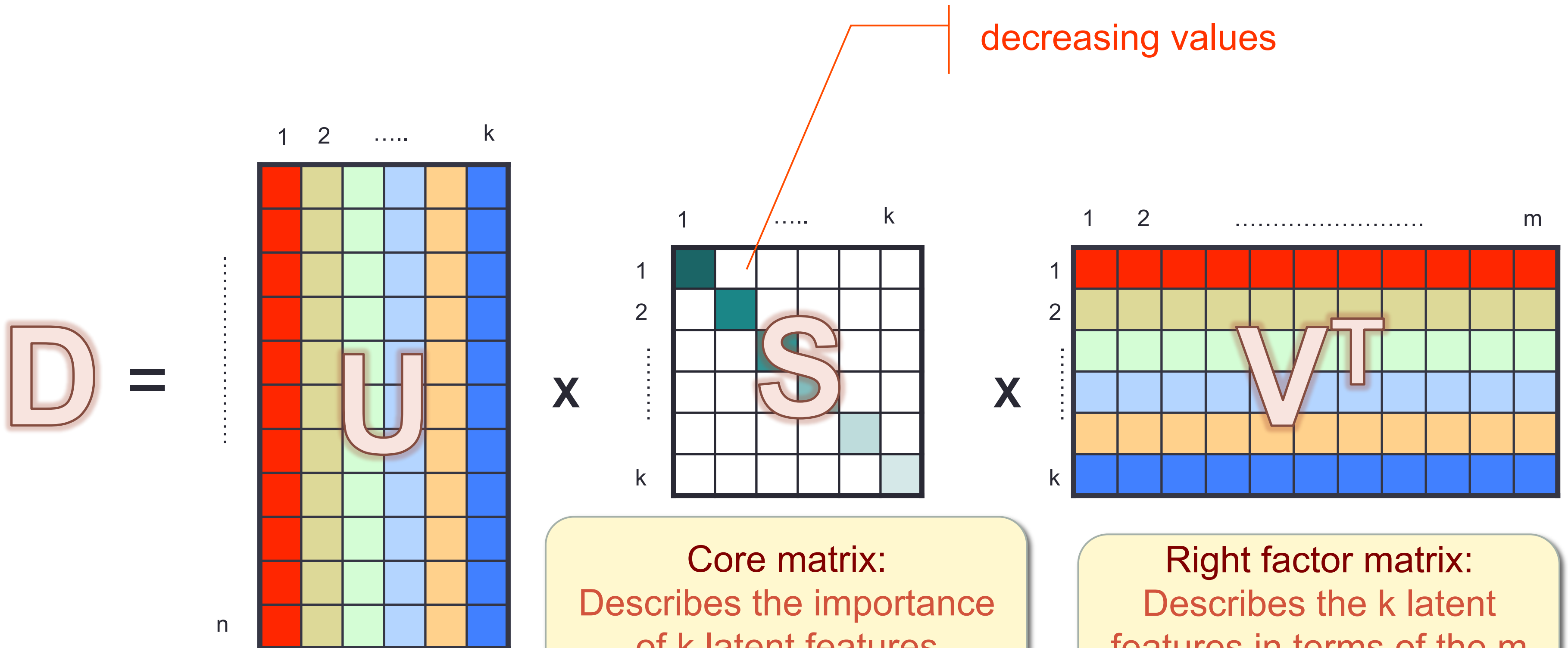
Some core concepts: slices



Some core concepts: matricization



Remember: Singular valued decomposition (SVD)

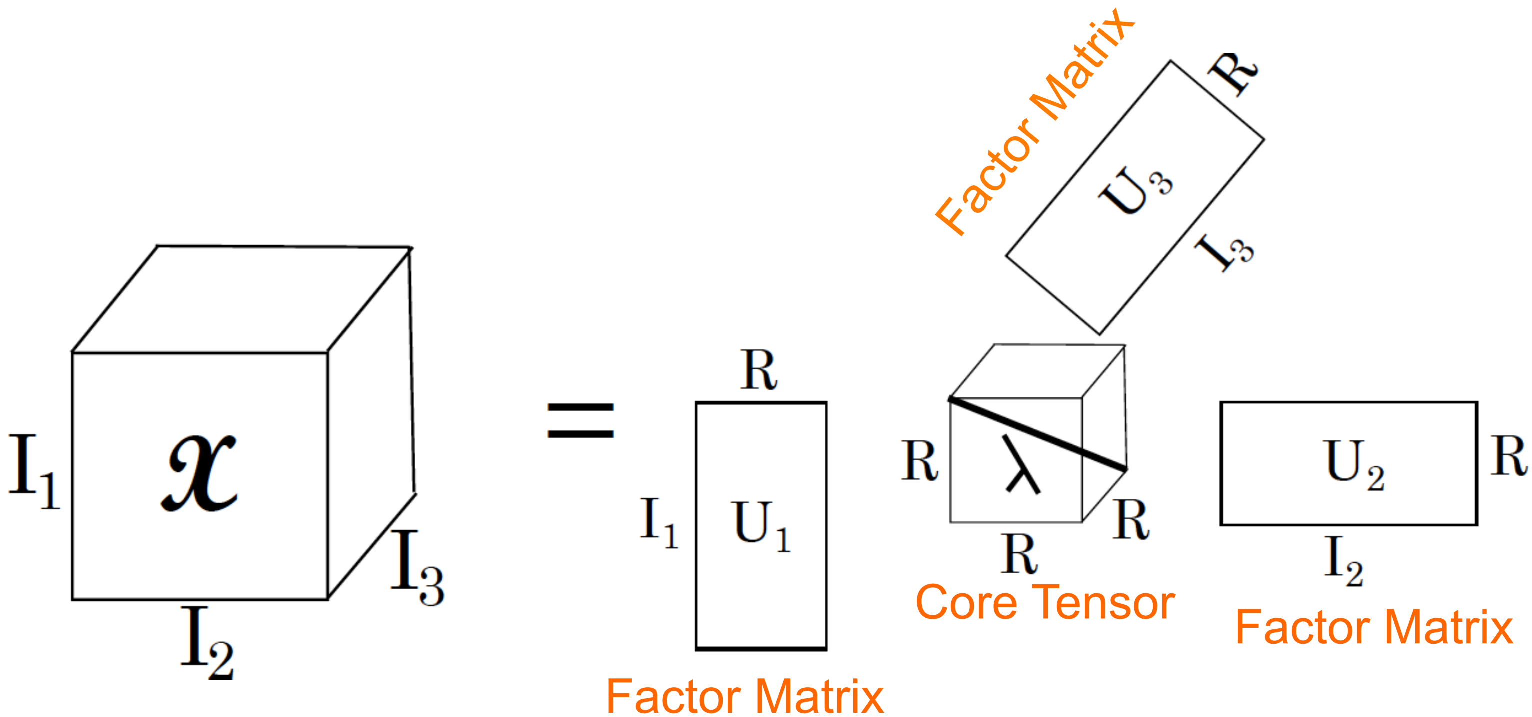


Left factor matrix:
Describes the n objects in terms of the k latent features

Core matrix:
Describes the importance of k latent features

Right factor matrix:
Describes the k latent features in terms of the m old features

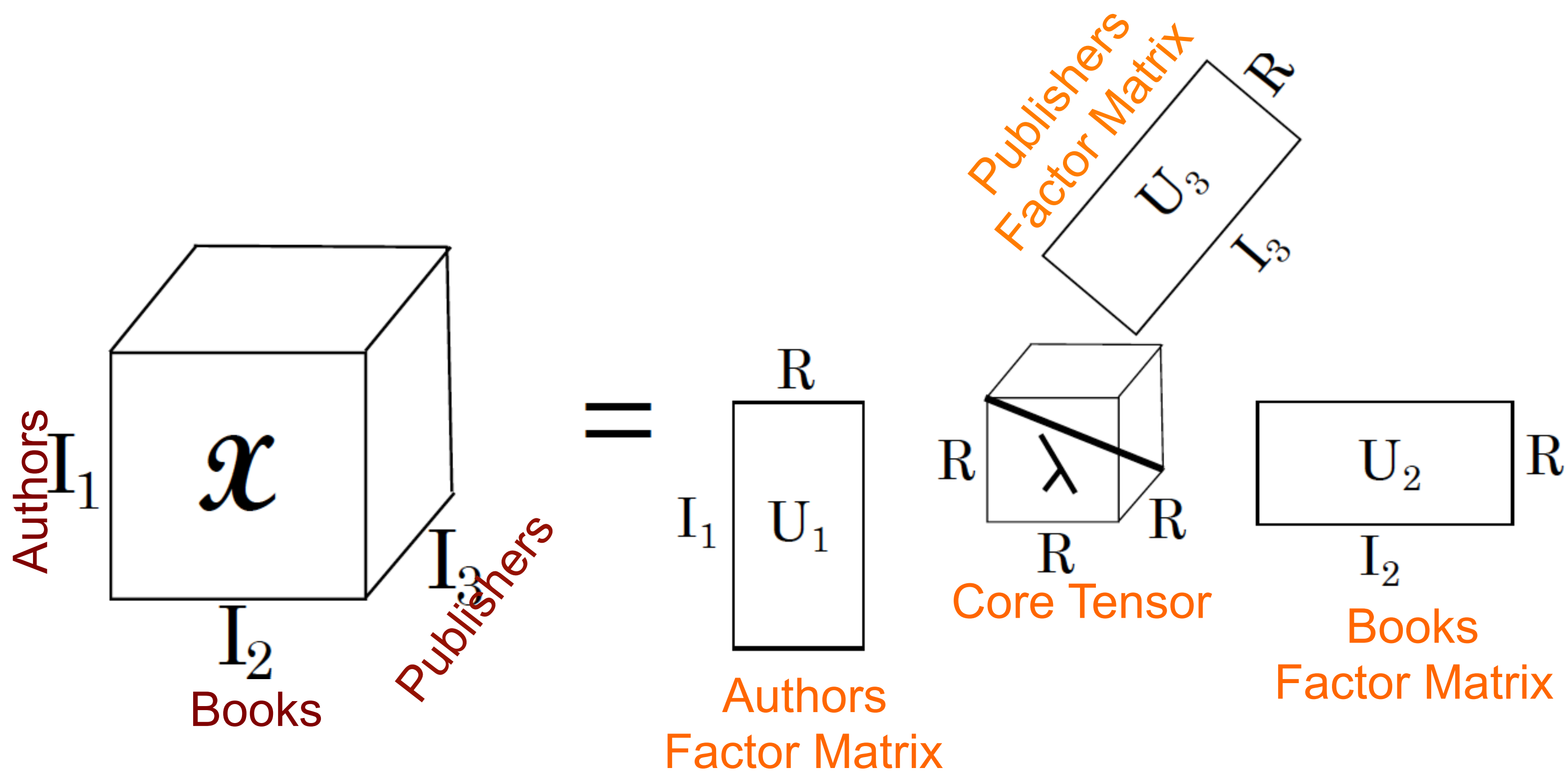
CANDECOMP / PARAllel FACtors (PARAFAC) – CP Decomposition [Carrol et al., 1970; Harshman, 1970]



Factor matrices: R latent clusters and memberships

Core tensor: strength of the R latent clusters

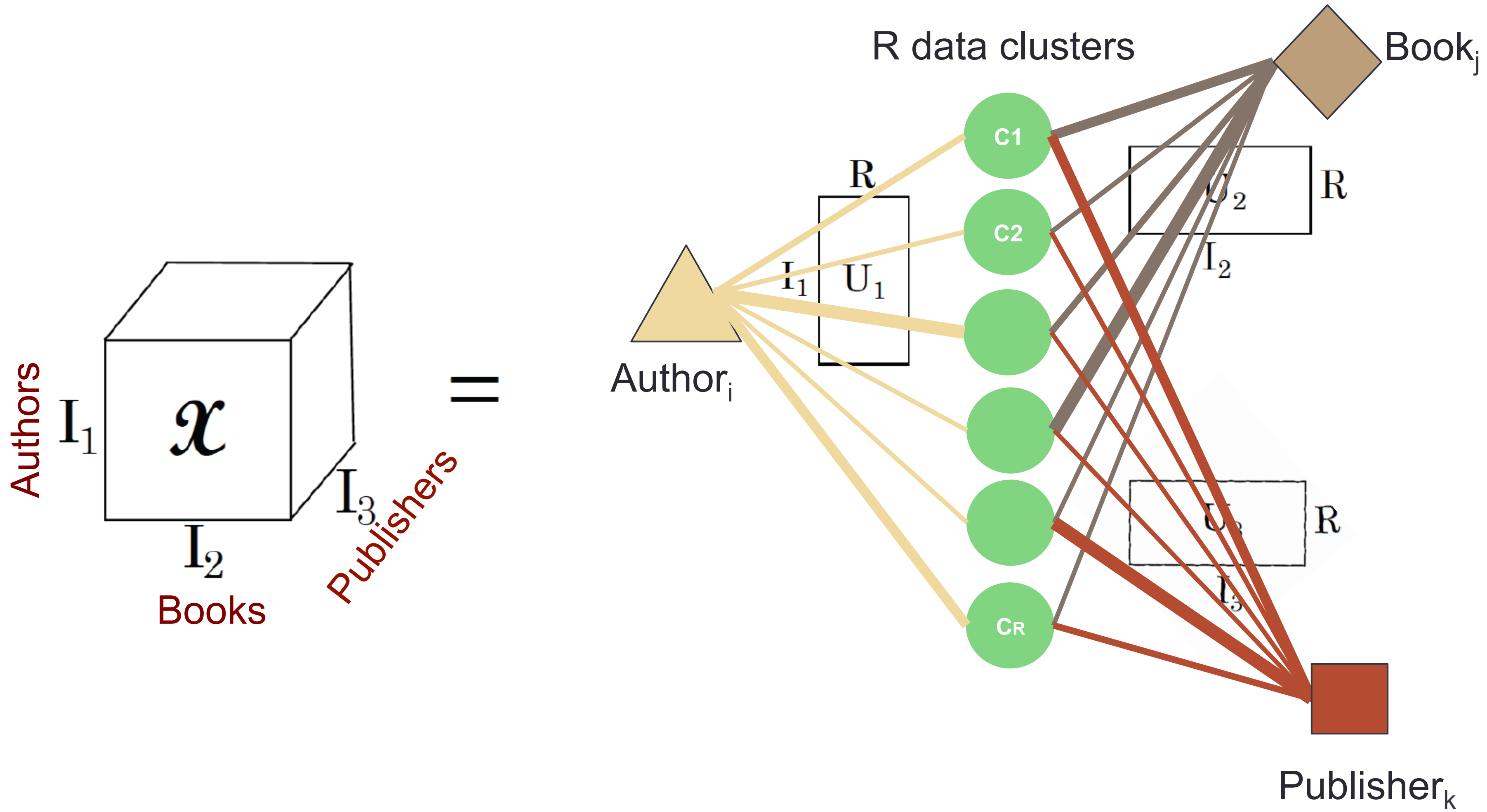
CANDECOMP / PARAllel FACtors (PARAFAC) – CP Decomposition [Carrol et al., 1970; Harshman, 1970]



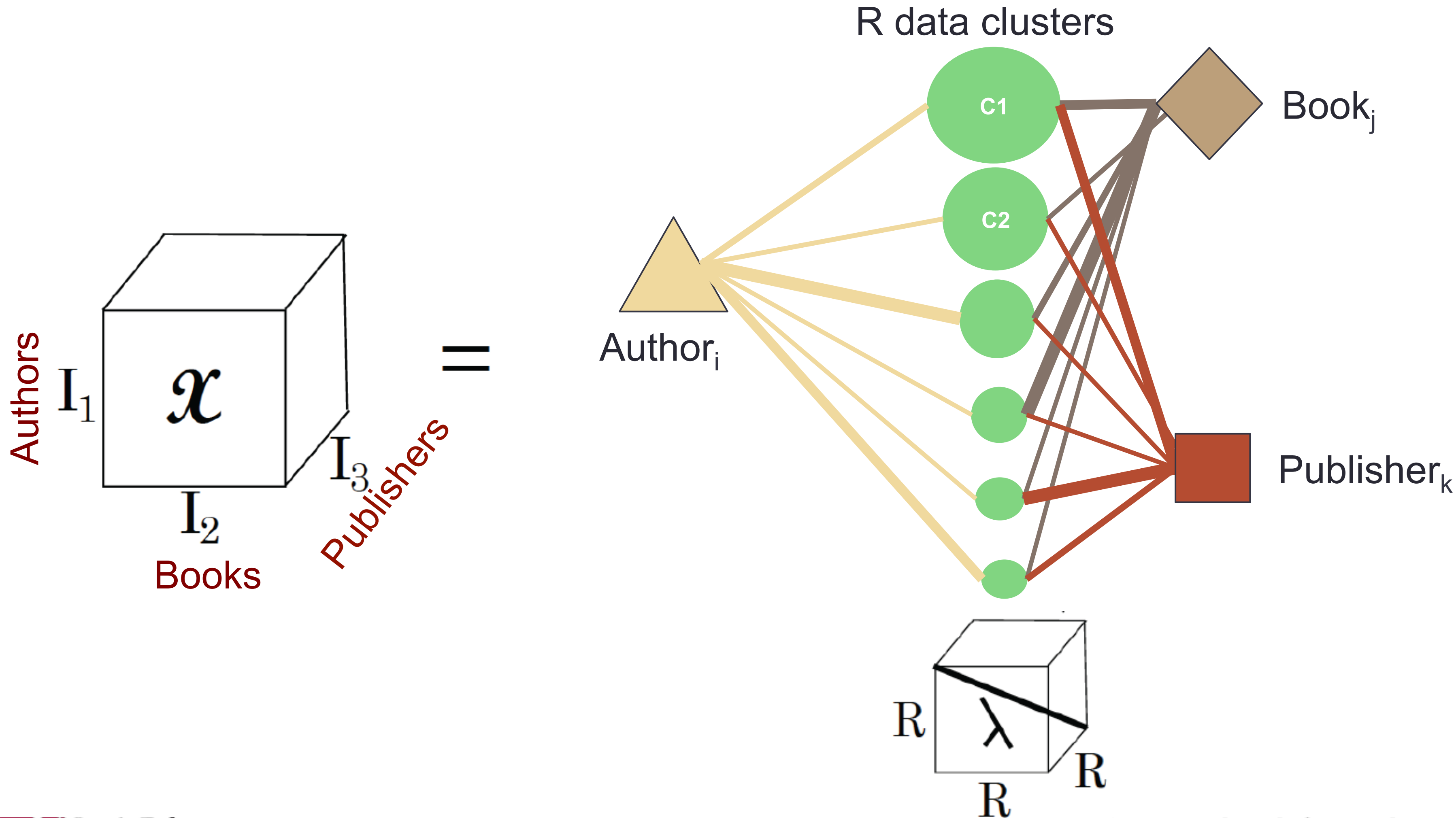
Factor matrices: R latent clusters and memberships

Core tensor: strength of the R latent clusters

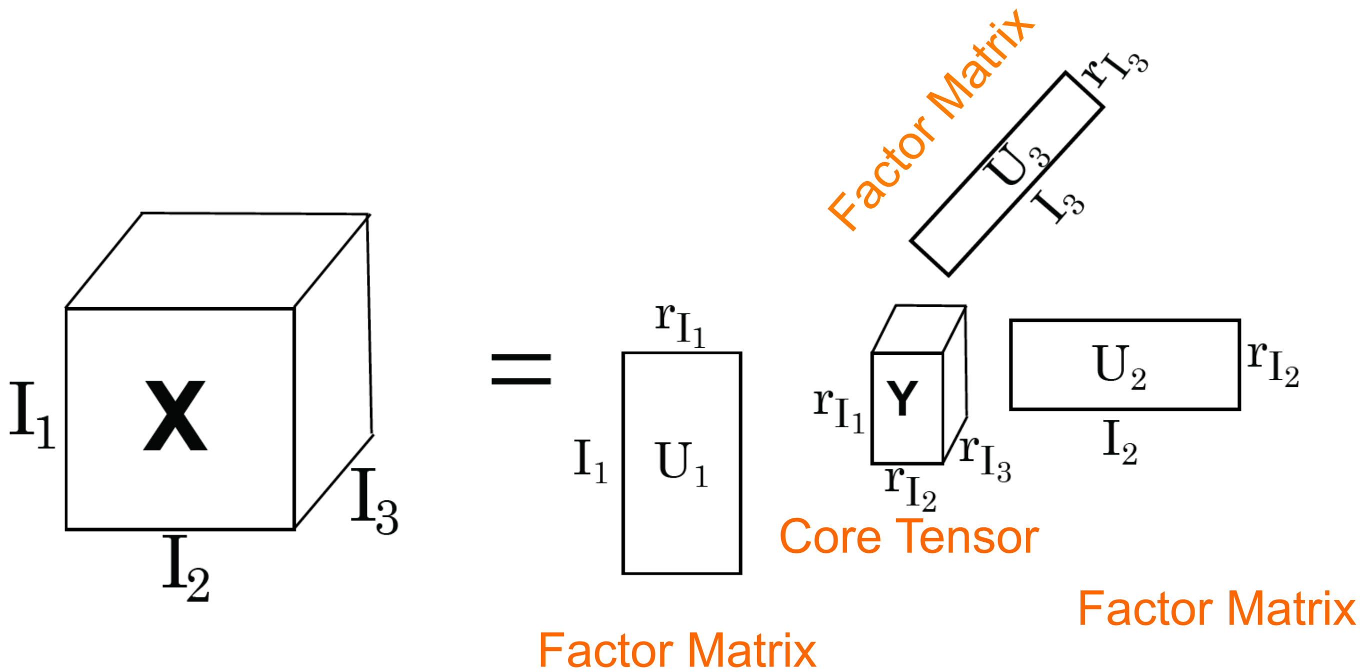
CP Decomposition: Factor matrices



CP Decomposition: Core tensor

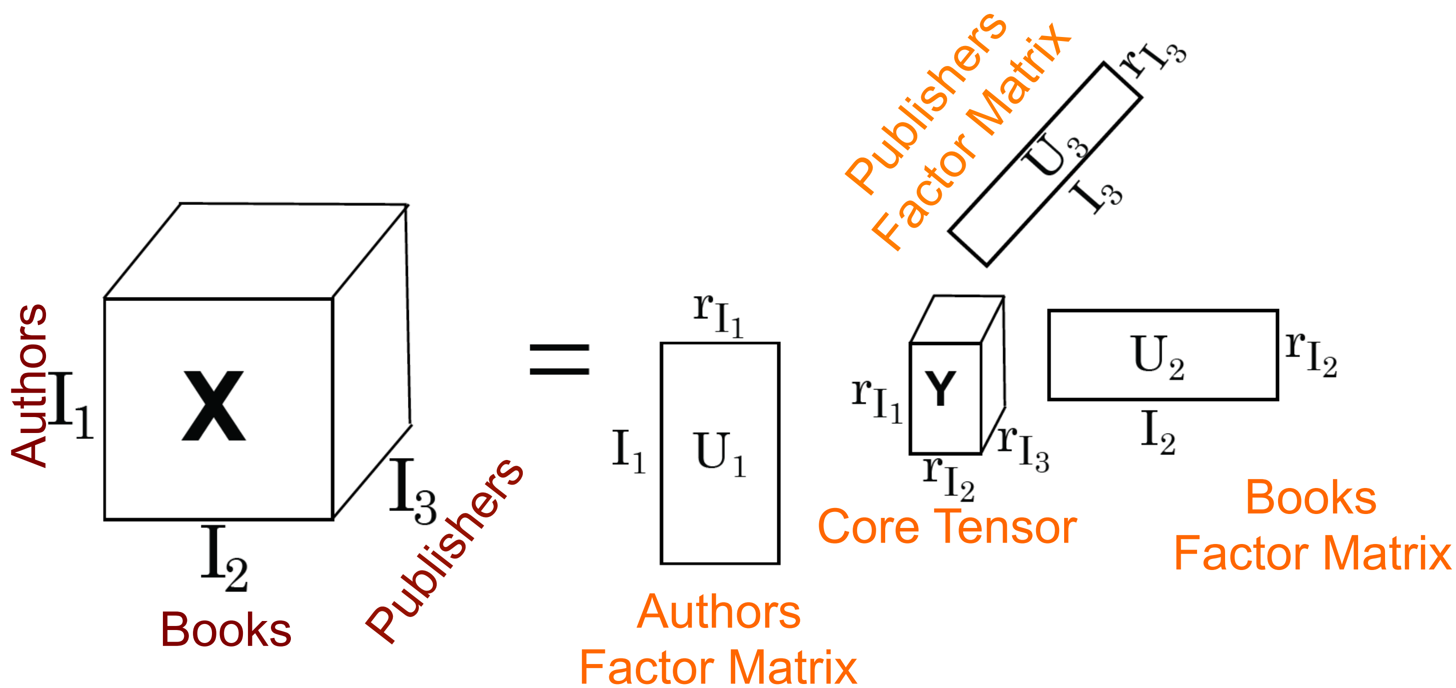


Tucker Decomposition [Tucker, 1966]



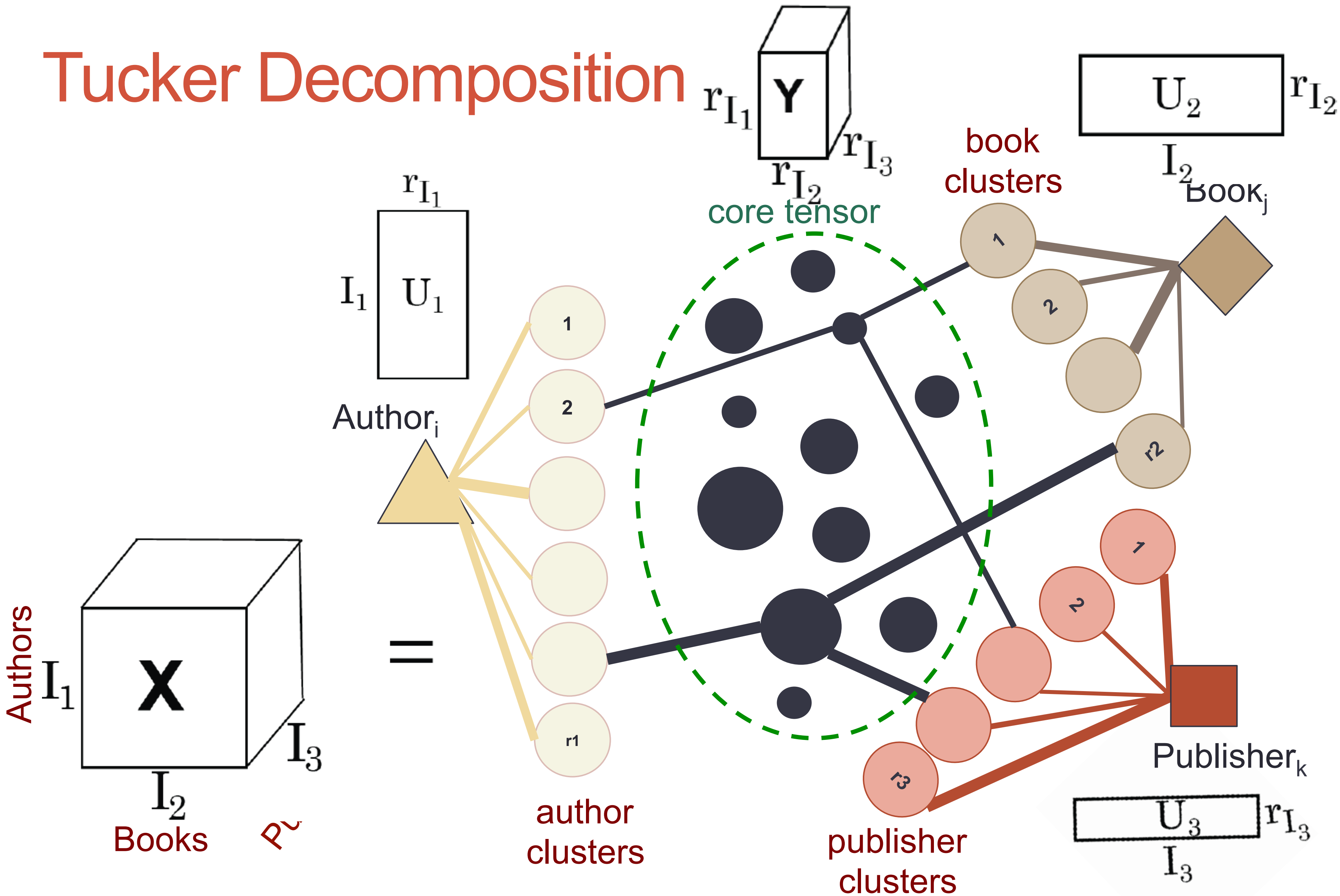
Factor matrices: r_{I1} , r_{I2}, r_{I3} modal latent clusters and memberships
[Dense] Core tensor: strength of the relationships among modal latent clusters

Tucker Decomposition [Tucker, 1966]

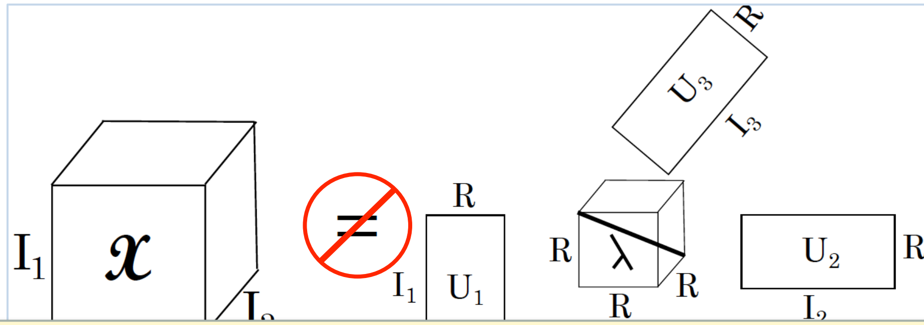


Factor matrices: r_{I1} , r_{I2}, r_{I3} modal latent clusters and memberships
[Dense] Core tensor: strength of the relationships among modal latent clusters

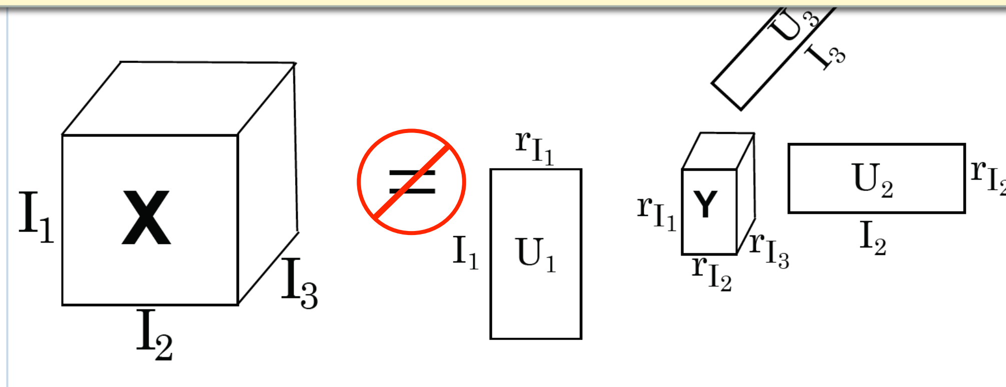
Tucker Decomposition



One problem!



Seek approximations instead!



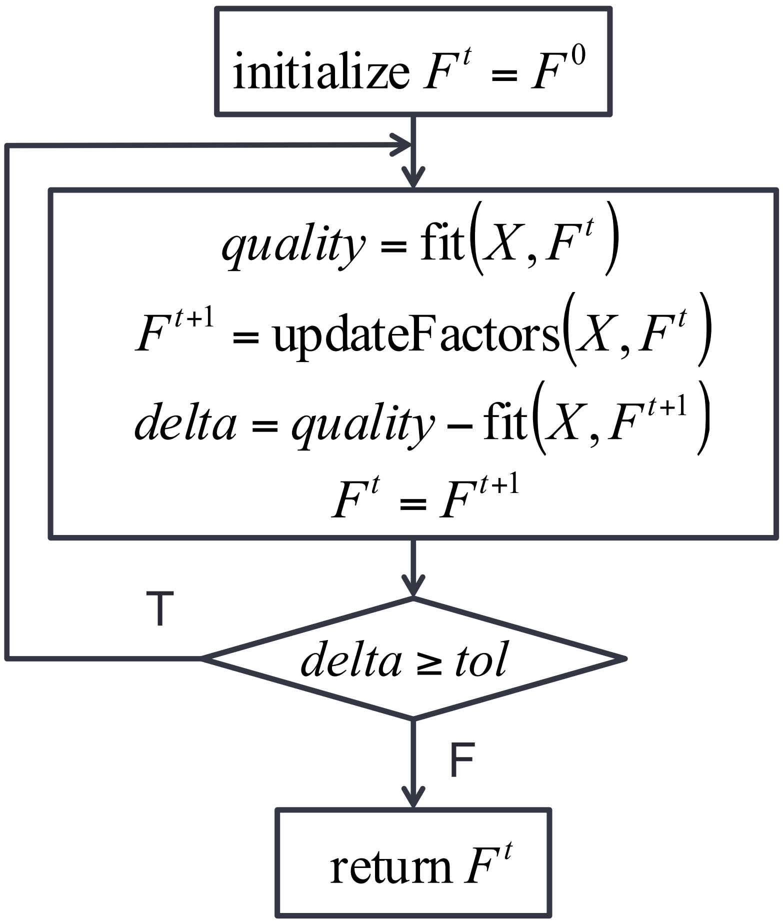
Fitting tensor decompositions

- Iterative algorithms
  - Alternating Least Squares (ALS)
  - Alternating Slice-Wise Diagonalization (ASD)
  - Self Weighted Alternating Trilinear Diagonalization (SWA-TLD)

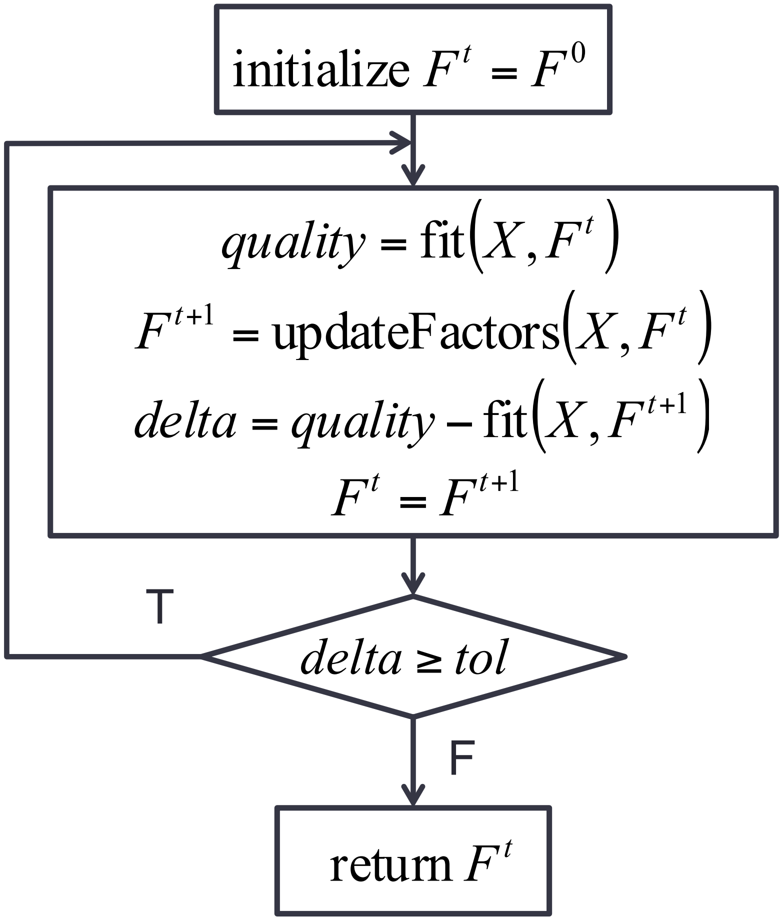
- Closed form algorithms
  - Generalized rank annihilation method (GRAM)
  - Direct trilinear decomposition (DTLD)

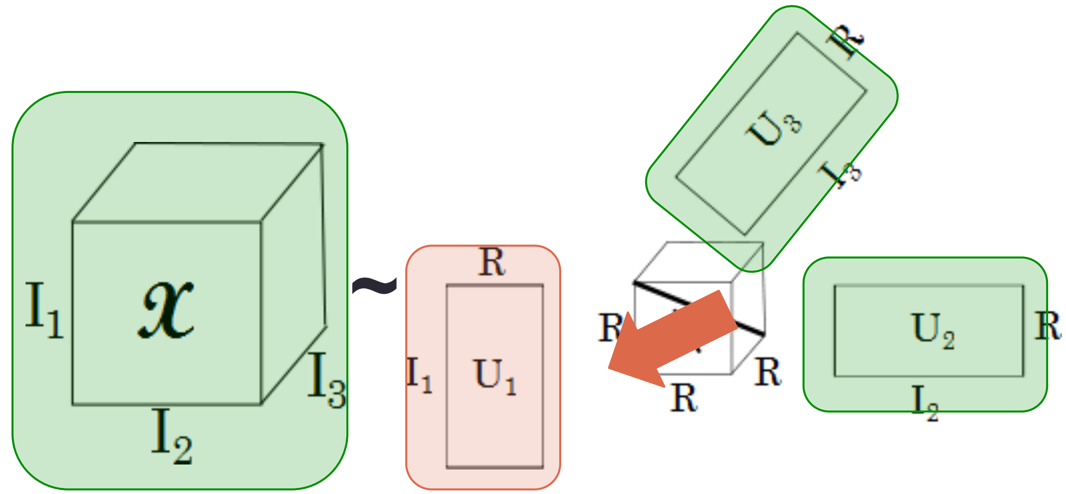
- Gradient-based methods
  - PMF3 (based on Gauss-Newton method)

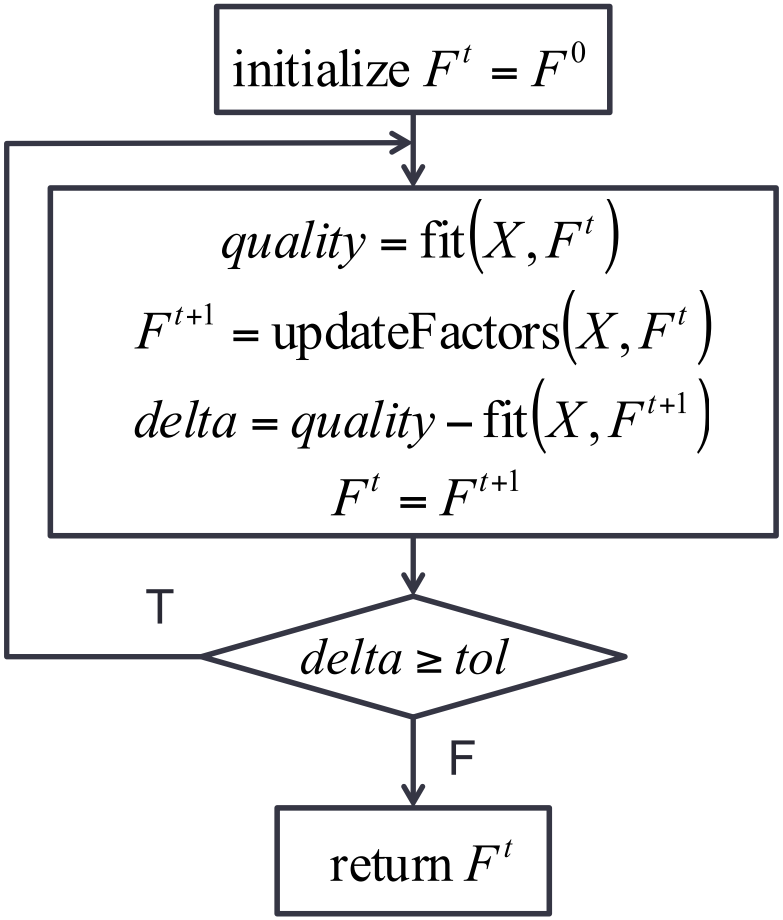
Iterative ALS algorithm

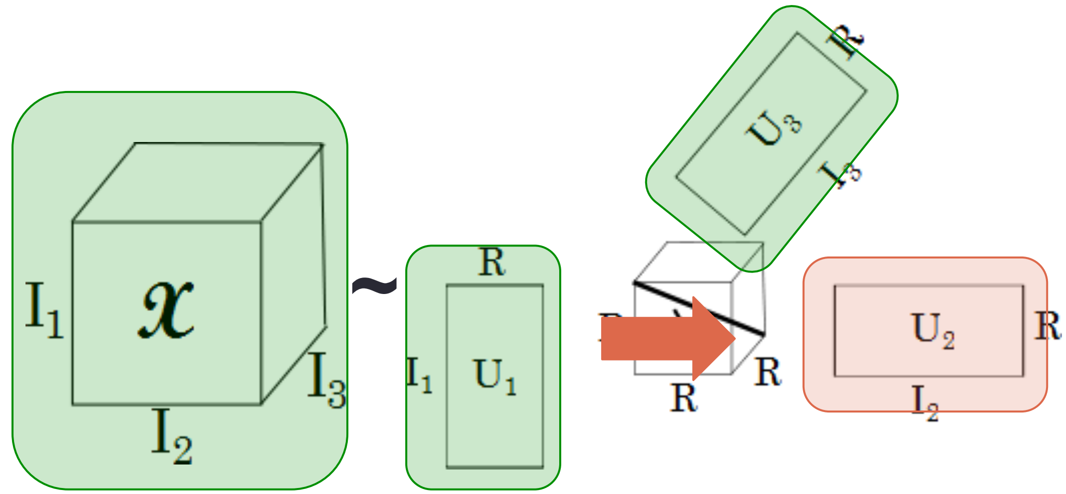


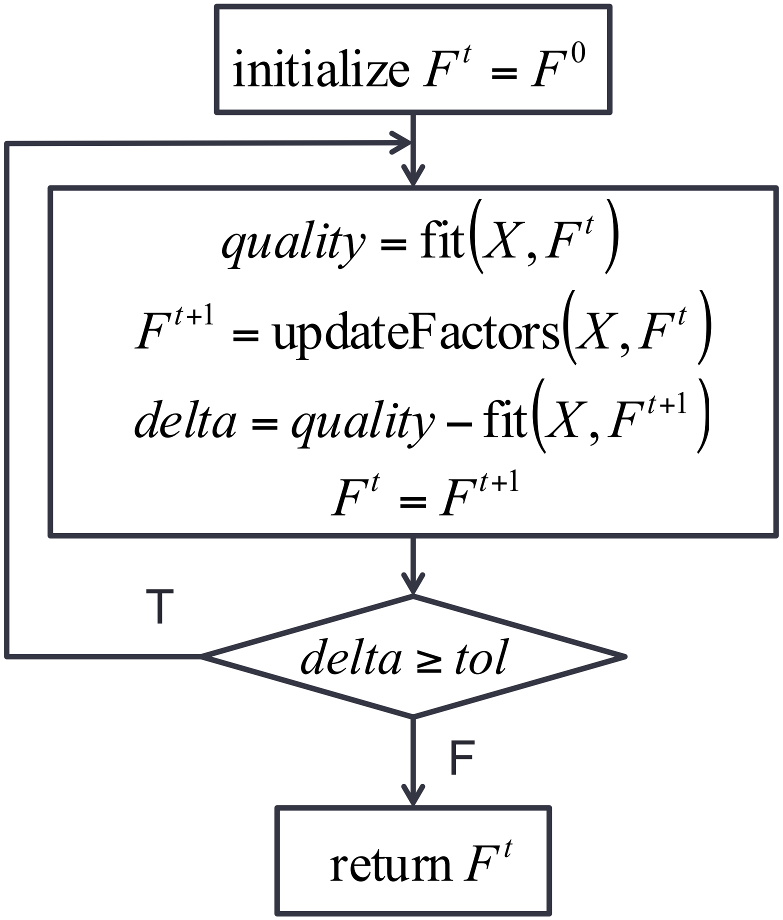
- At each step, all factor matrices are updated one at a time
- A factor matrix is estimated starting from the others

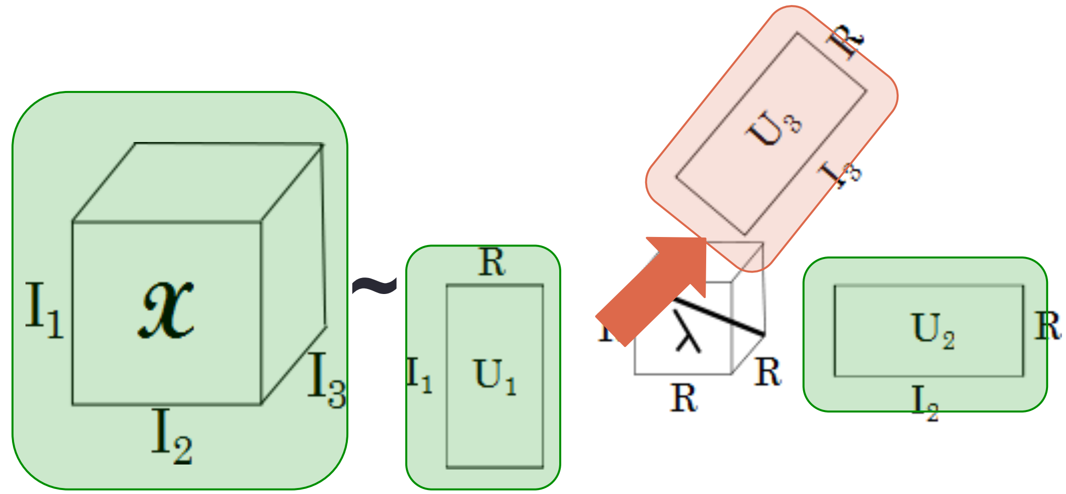

initialize $F^t = F^0$
$quality = \mathrm{fit}(X, F^t)$
$F^{t+1} = \mathrm{updateFactors}(X, F^t)$
$delta = quality - \mathrm{fit}(X, F^{t+1})$
$F^t = F^{t+1}$
T
$delta \geq tol$
F
return F^t


I_1
$\mathcal{X}$
I_2
I_3
~
R
I_1
U_1
R
R
R
U_3
I_3
R
U_2
R
I_2


initialize $F^t = F^0$
$quality = \mathrm{fit}(X, F^t)$
$F^{t+1} = \mathrm{updateFactors}(X, F^t)$
$delta = quality - \mathrm{fit}(X, F^{t+1})$
$F^t = F^{t+1}$
T
$delta \geq tol$
F
return F^t

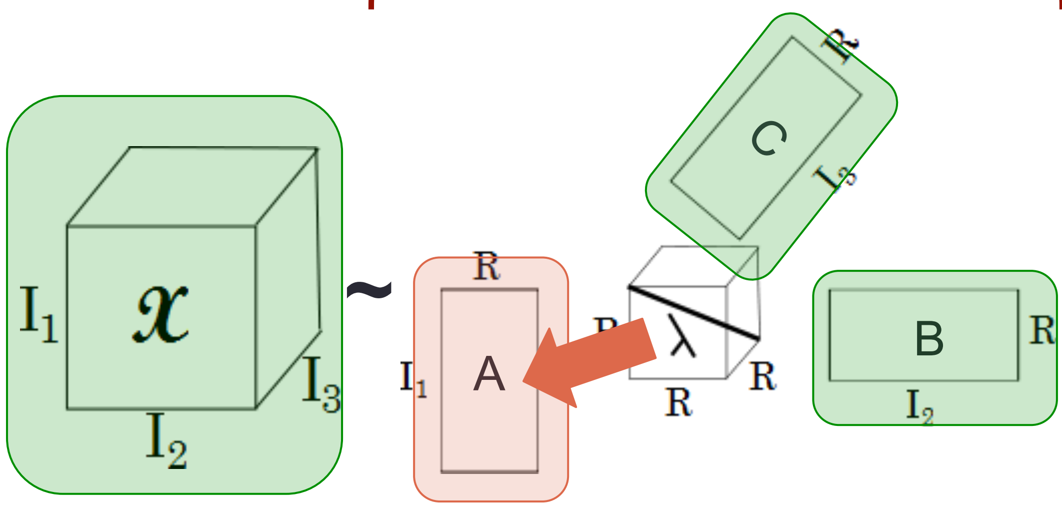

$\mathcal{X}$
I_1
I_2
I_3
~
R
I_1
U_1
U_3
I_3
R
R
R
U_2
R
I_2


initialize $F^t = F^0$
$quality = \mathrm{fit}(X, F^t)$
$F^{t+1} = \mathrm{updateFactors}(X, F^t)$
$delta = quality - \mathrm{fit}(X, F^{t+1})$
$F^t = F^{t+1}$
$delta \geq tol$
T
F
return F^t


I_1
$\mathcal{X}$
I_2
I_3
~
R
I_1
U_1
λ
R
R
R
U_3
I_3
U_2
R
I_2

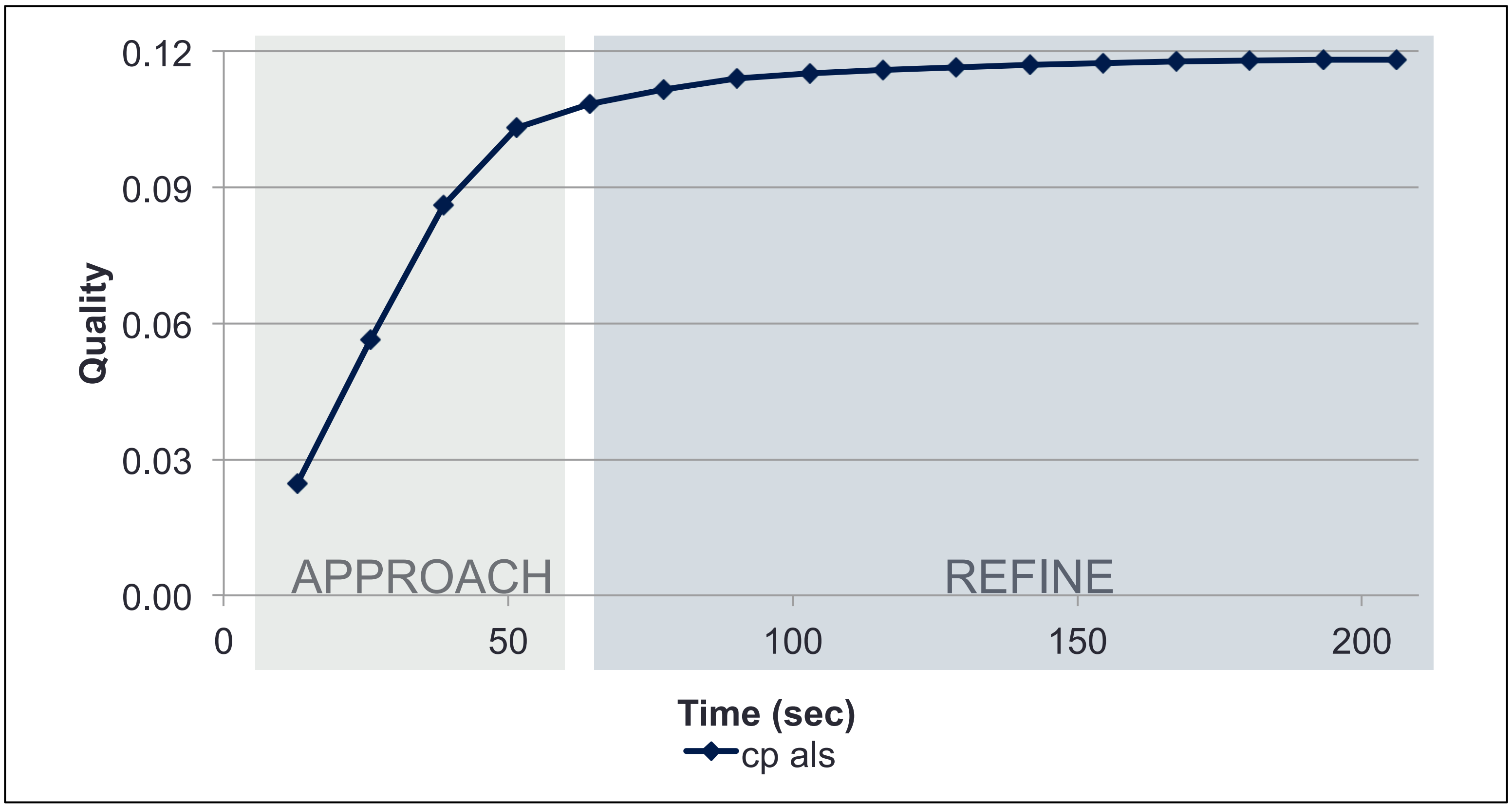
Alternating Least Squares (ALS) CP Decomposition

$$\min_{\hat{\mathcal{X}}} \overset{\text{mismatch}}{\boxed{\|\mathcal{X} - \hat{\mathcal{X}}\|}} \quad \text{with} \quad \hat{\mathcal{X}} = \sum_{r=1}^{R} \lambda_r \, \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r. = [\![\boldsymbol{\lambda}\, ; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!].$$

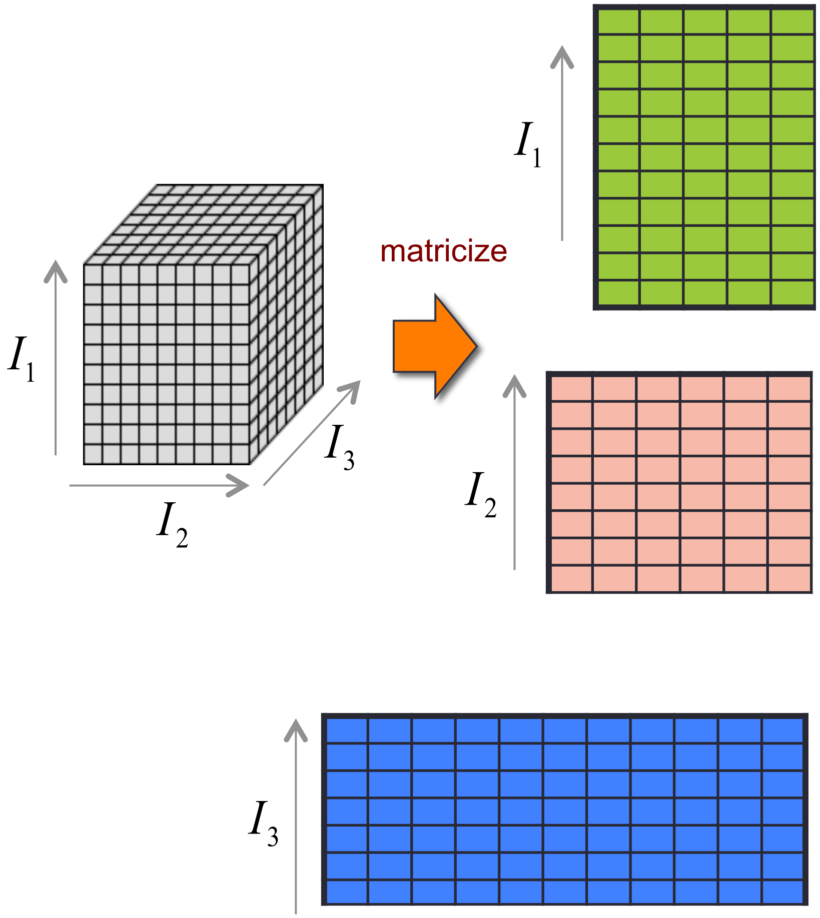


Computing the factor matrix, A, given X and the two other factor matrices B and C

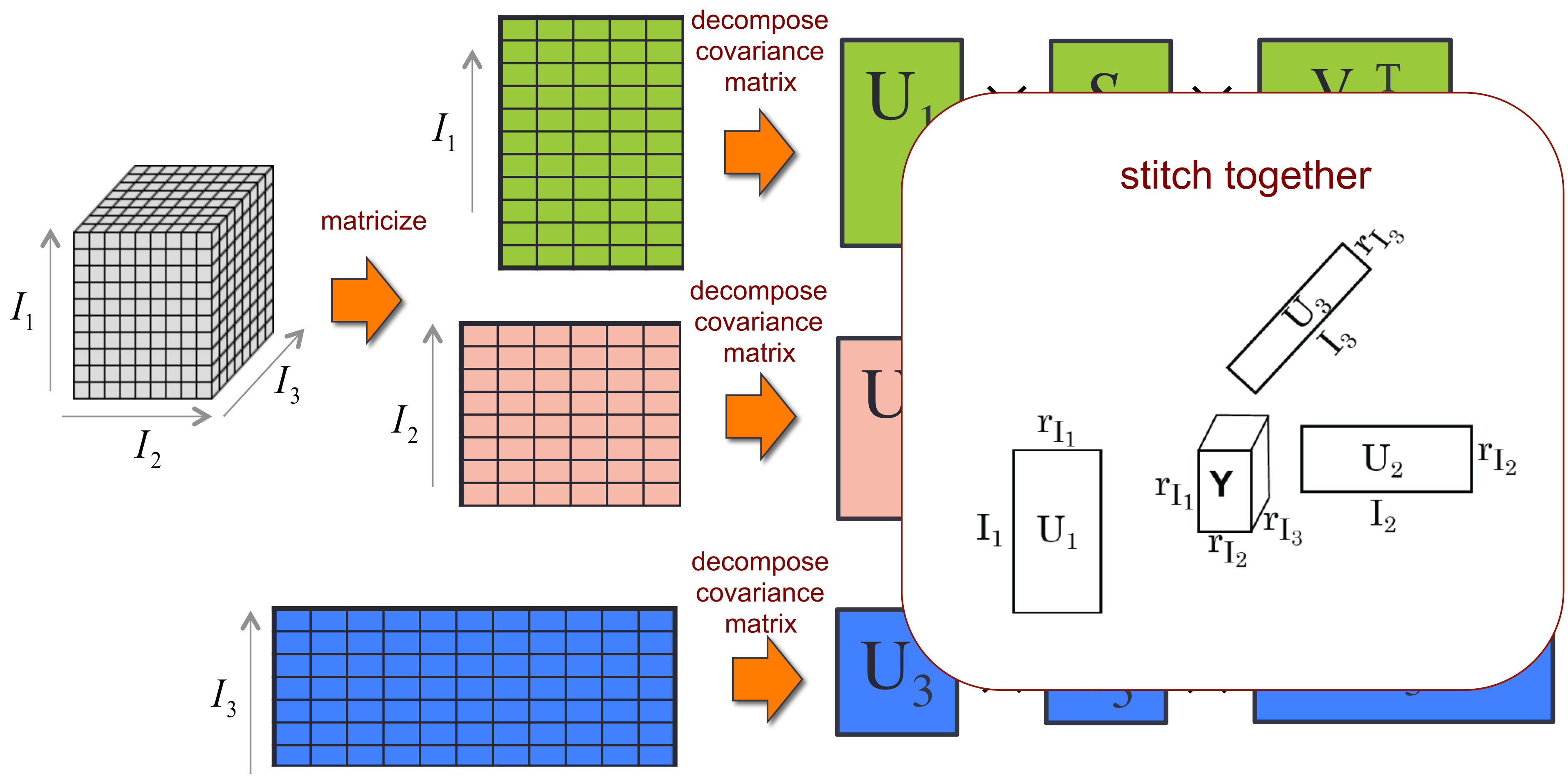
Iterative ALS



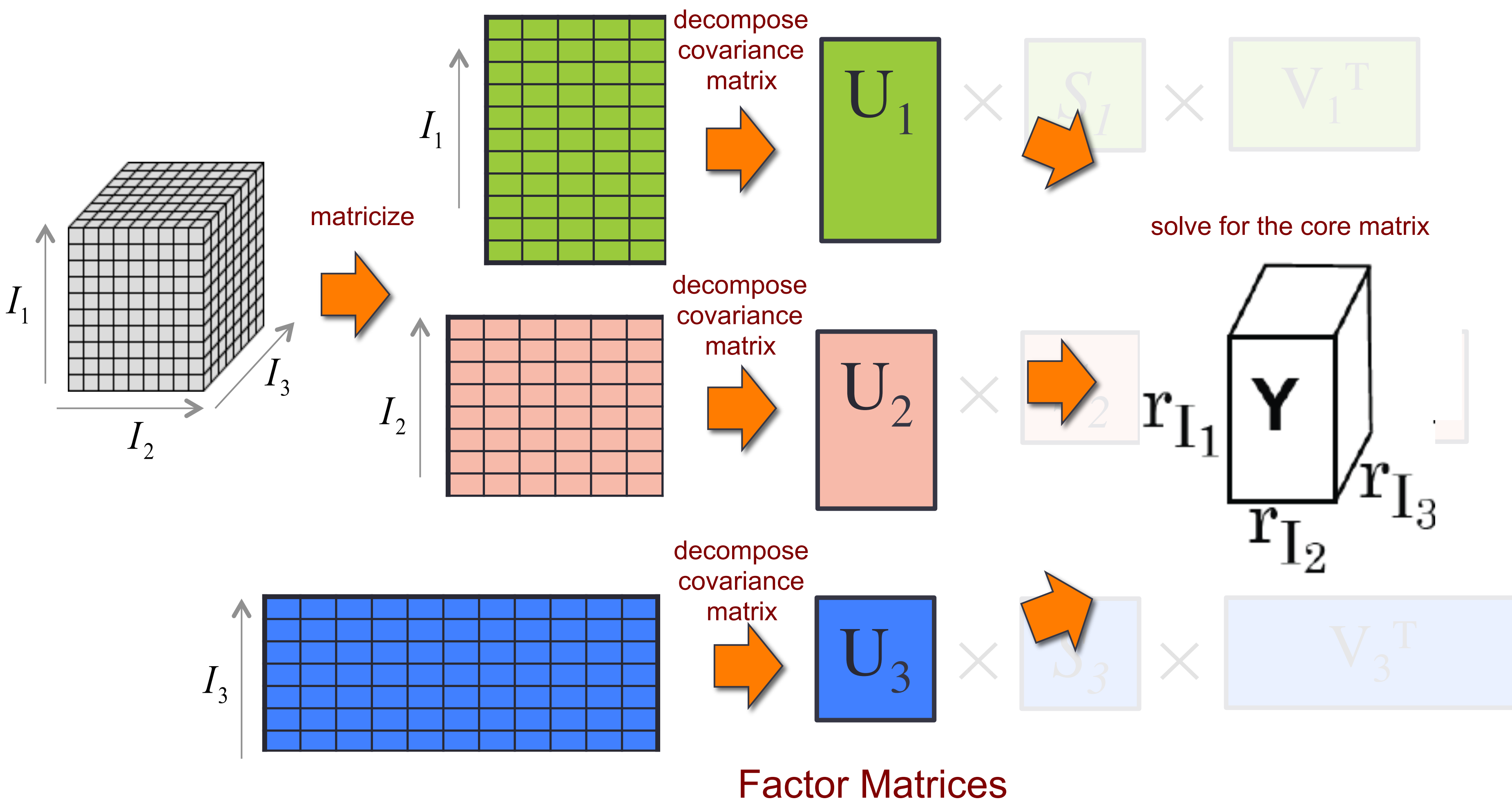
Tucker Decomposition – HOSVD



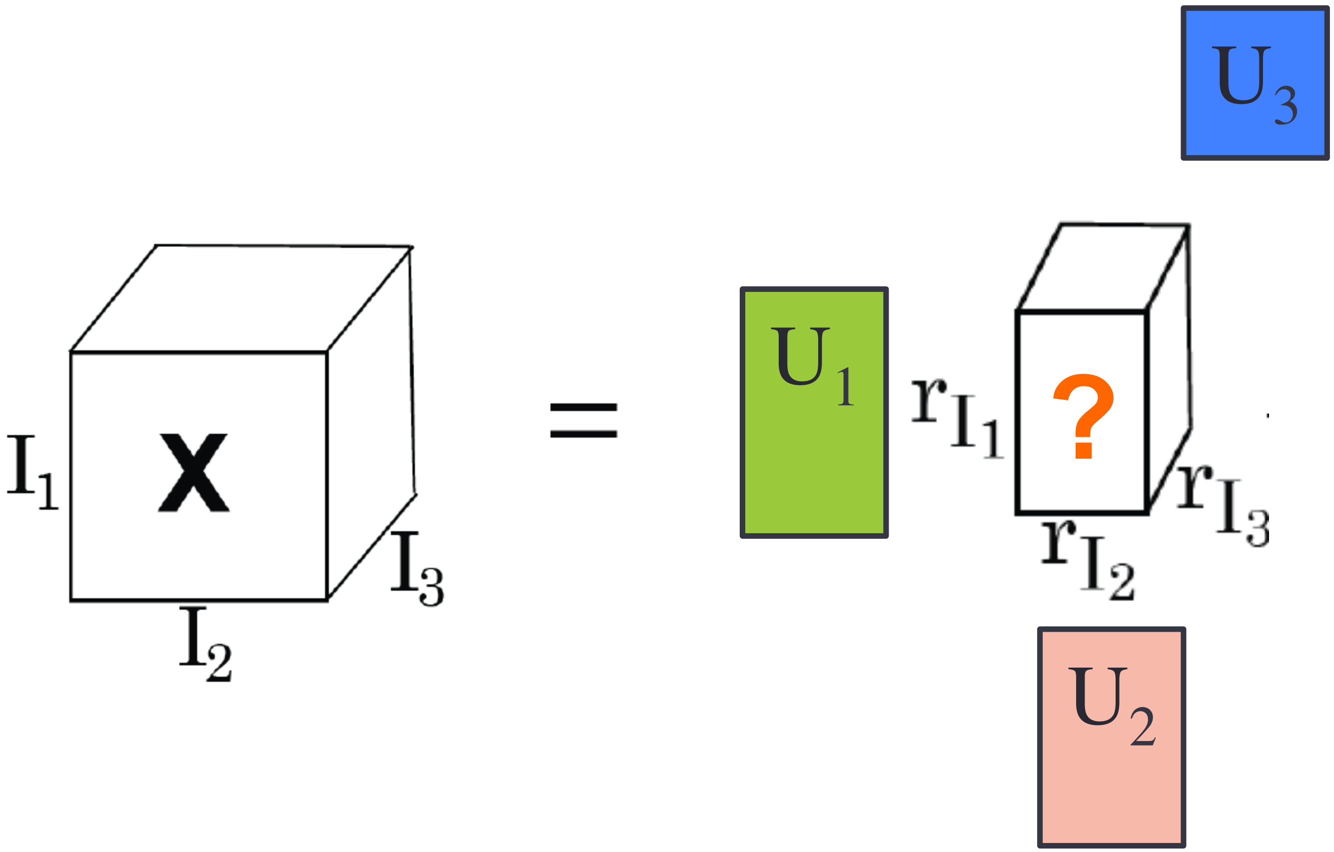
Tucker Decomposition – HOSVD



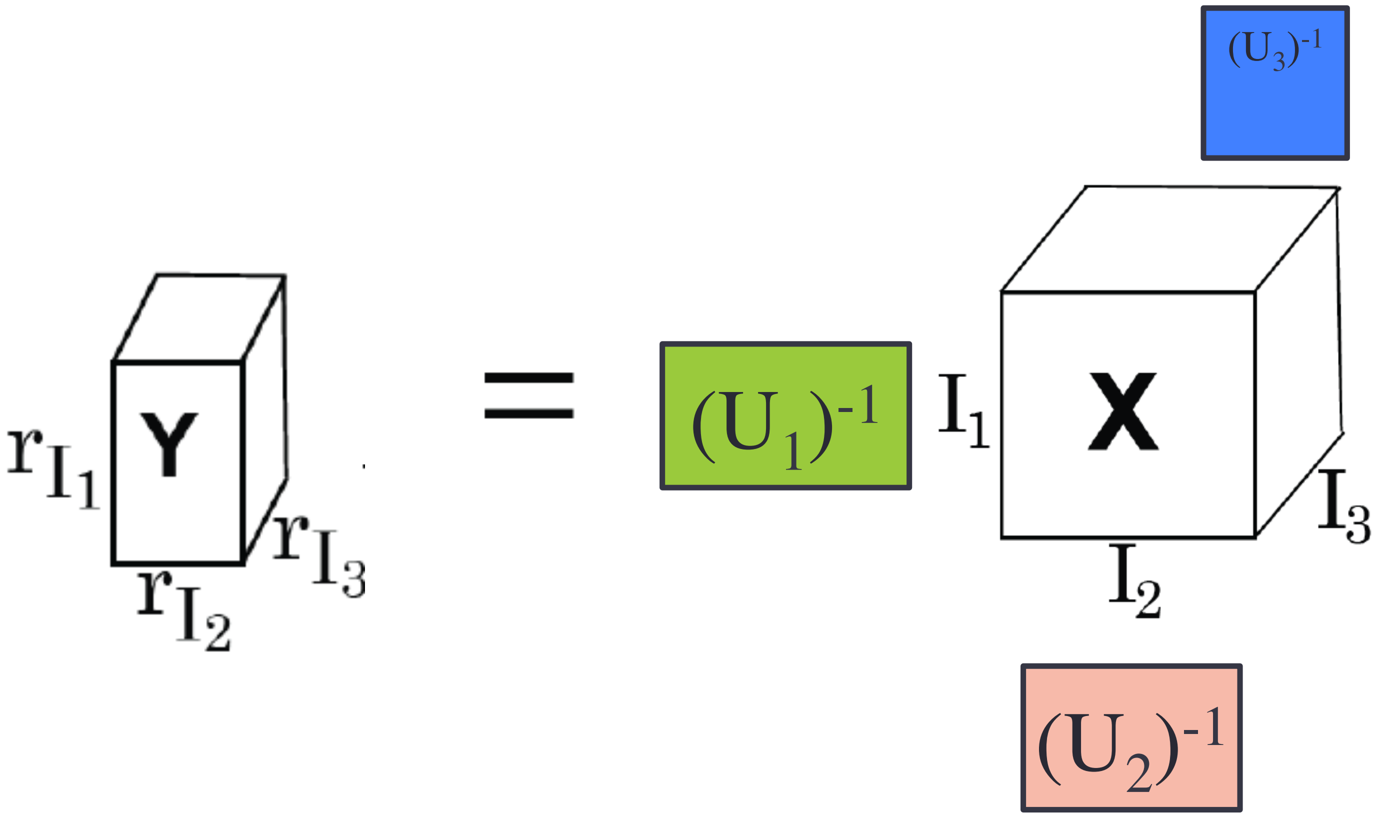
Tucker Decomposition – HOSVD



How to solve for the core matrix?

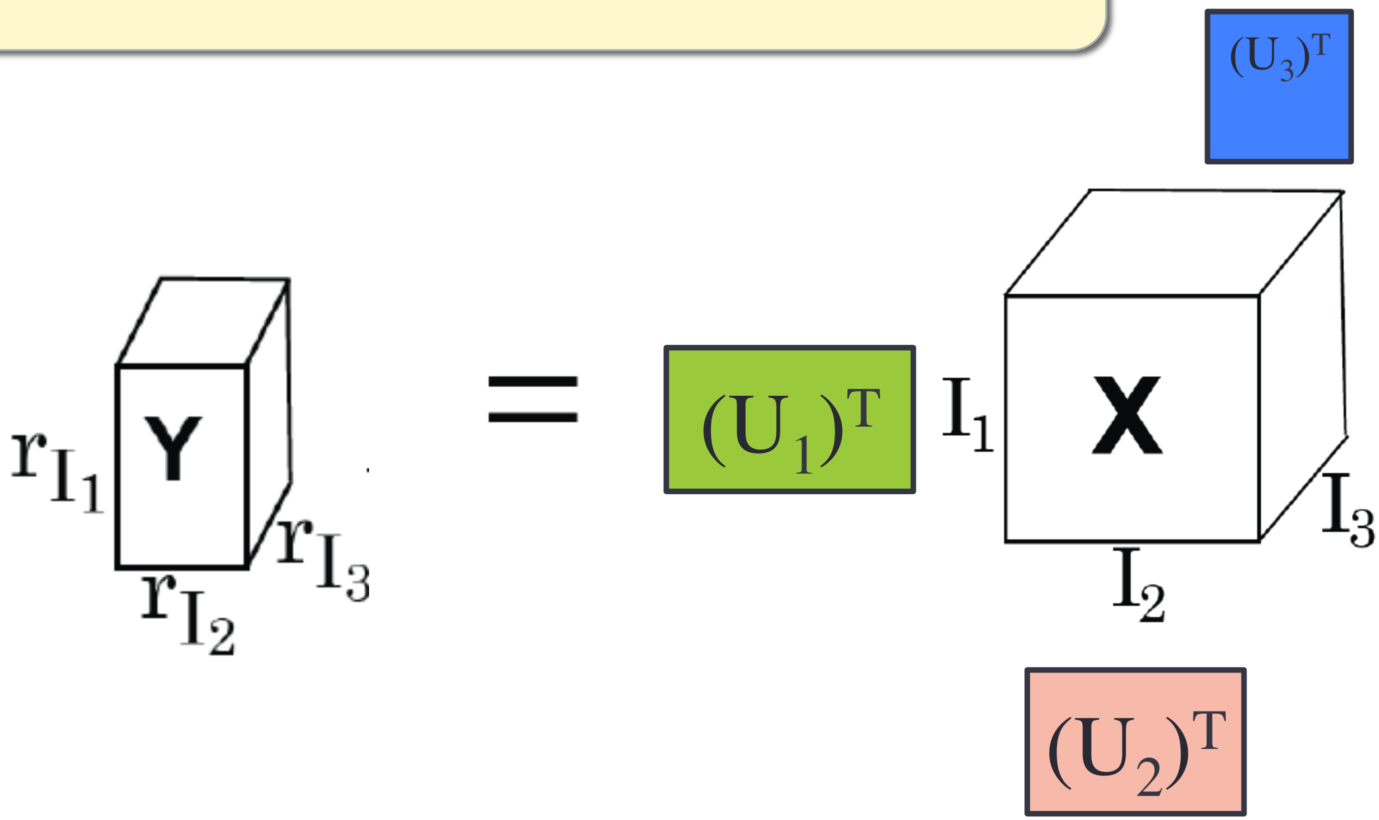


How to solve for the core matrix?



How to solve for the core matrix?

Reminder: U's are left eigenvectors of the symmetric, square covariance matrix. Therefore $U^{-1} = U^T$

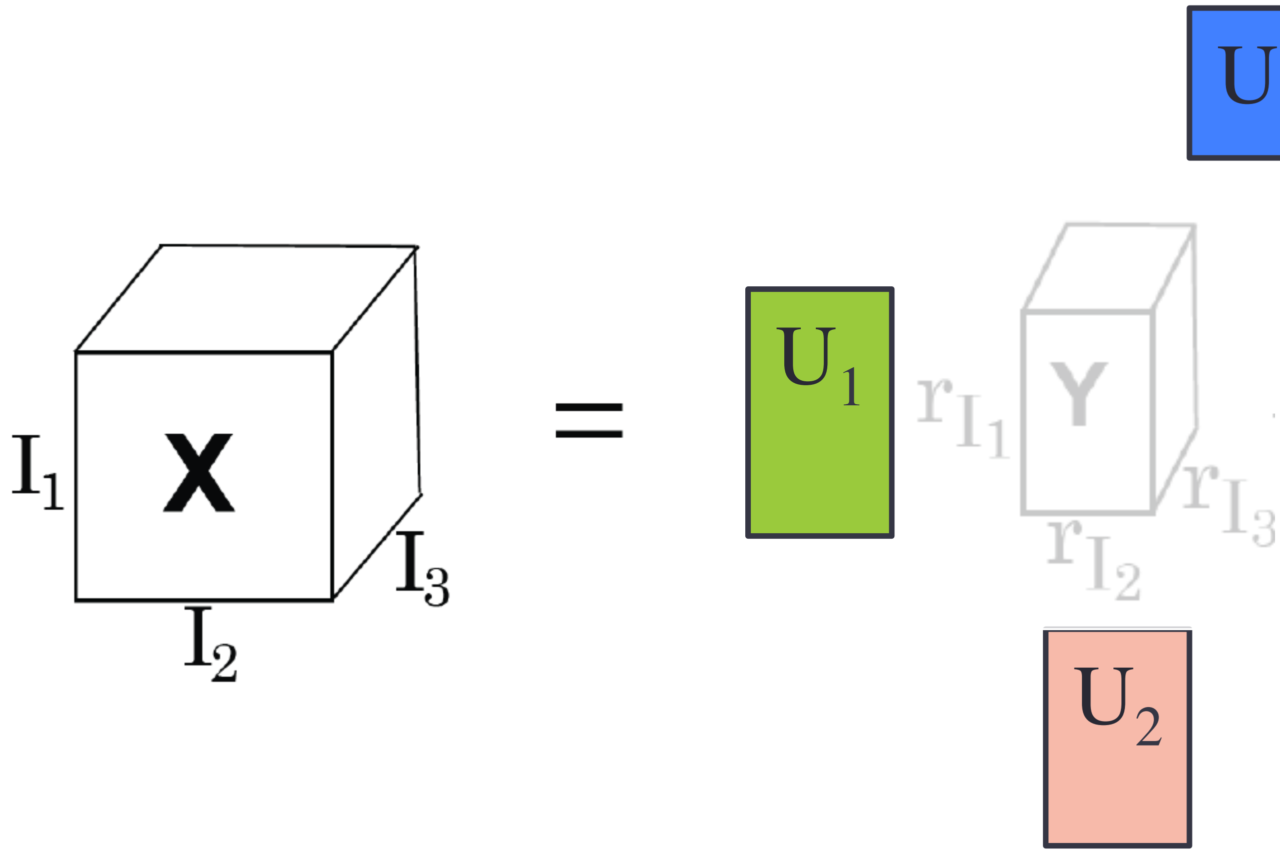


Tucker - ALS

- How about alternating least squares?

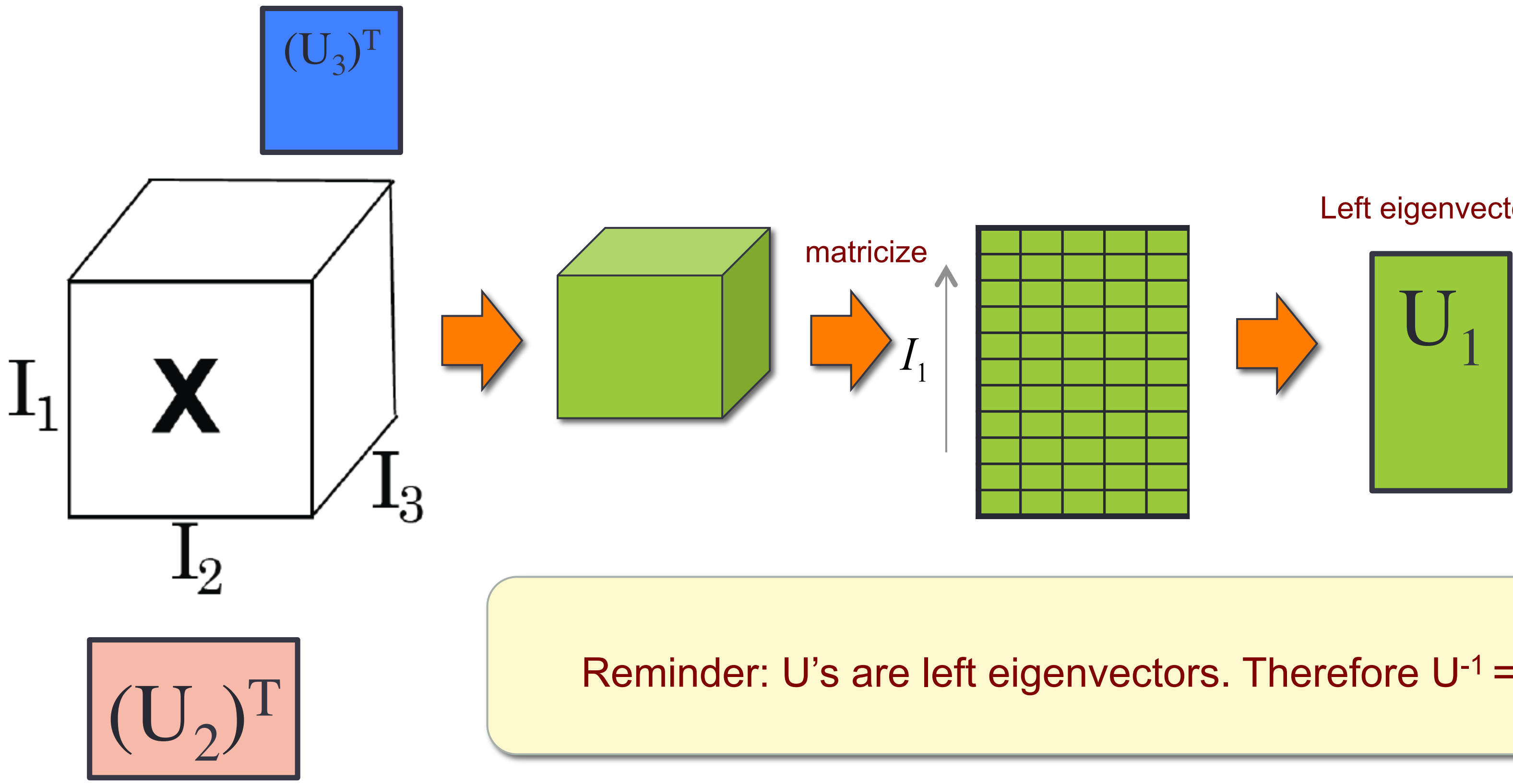
Tucker - ALS

- Step 1: Start with random factor matrices



Tucker - ALS

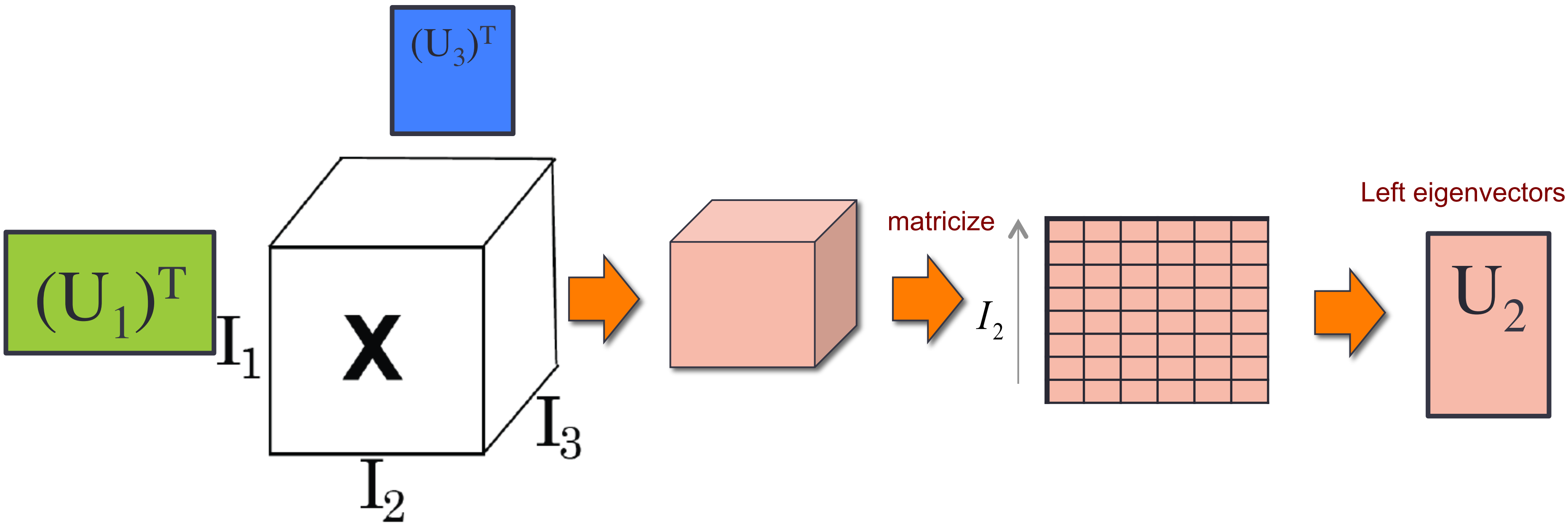
- Step 2: Solve for revised U_1



Reminder: U's are left eigenvectors. Therefore $U^{-1} = U^T$

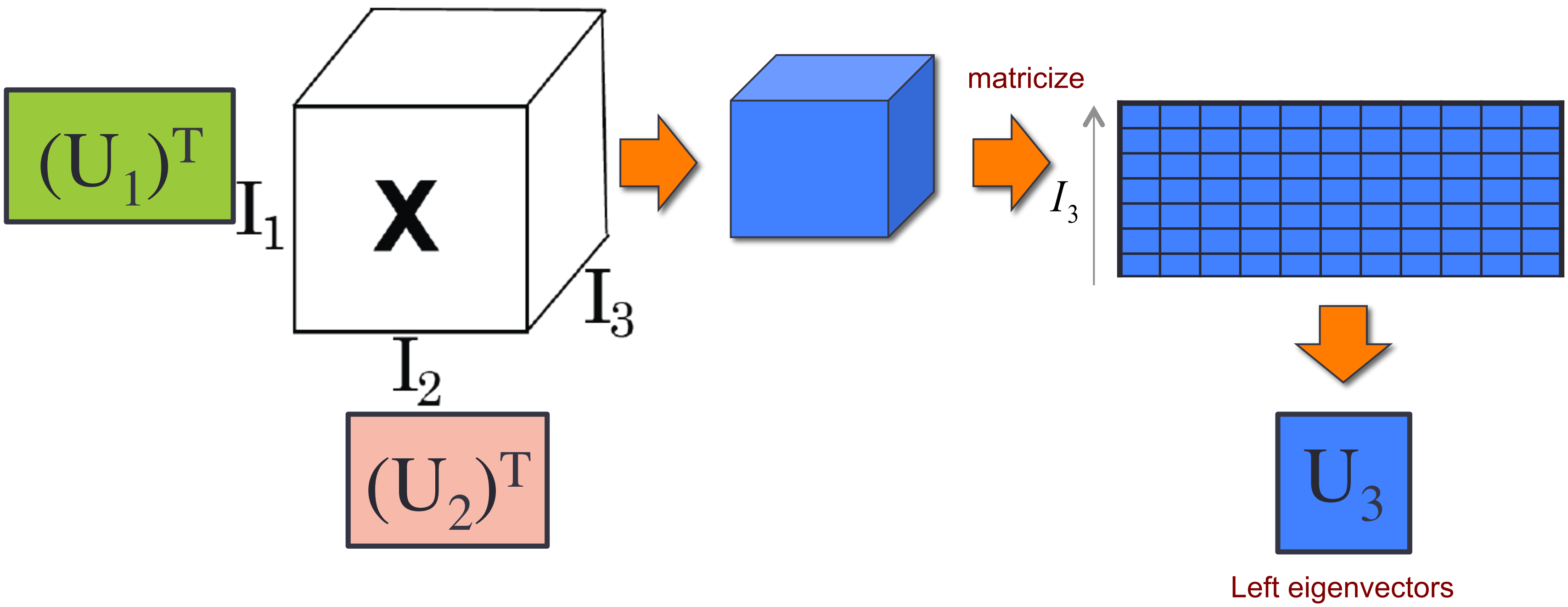
Tucker - ALS

- Step 3: Solve for revised U_2



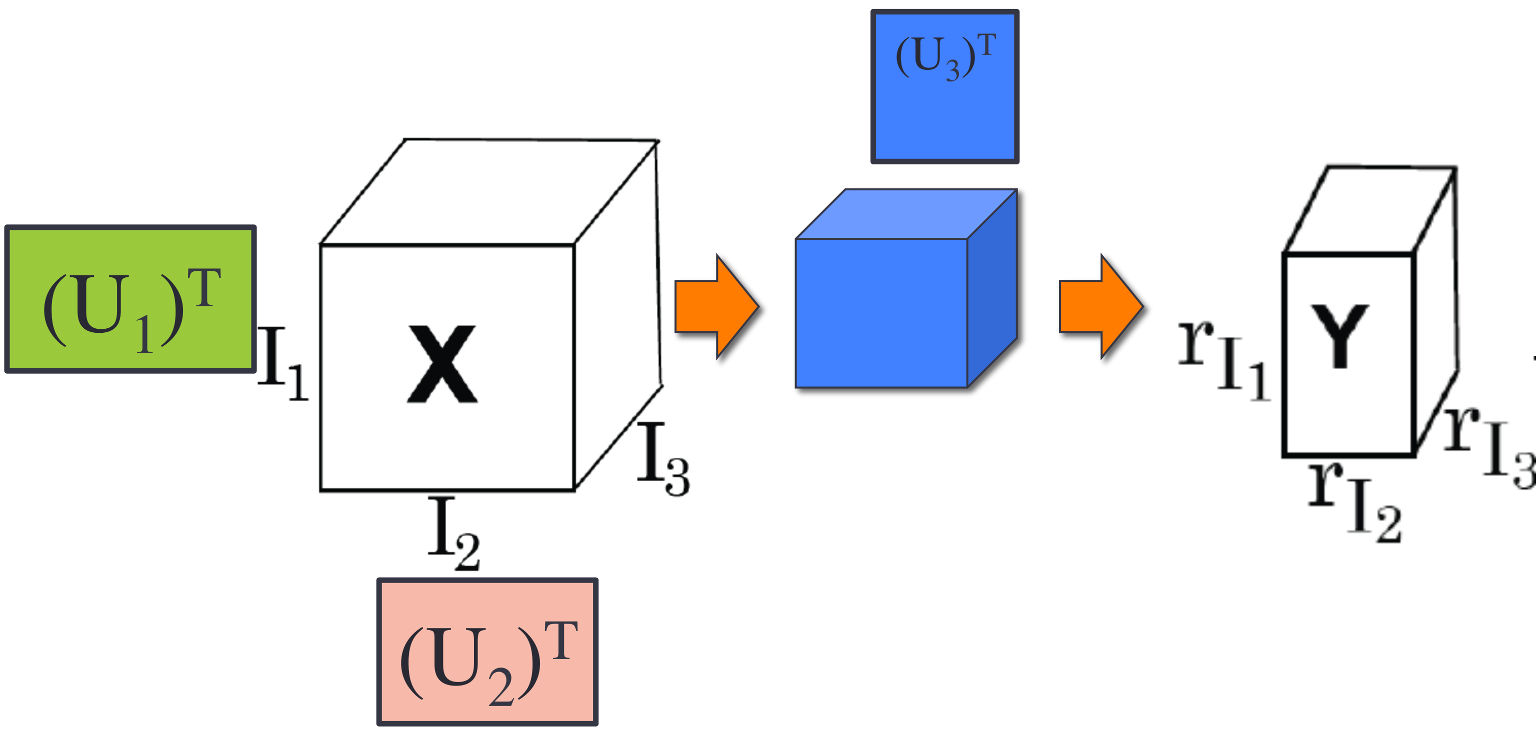
Tucker - ALS

- Step 4: Solve for revised U_3



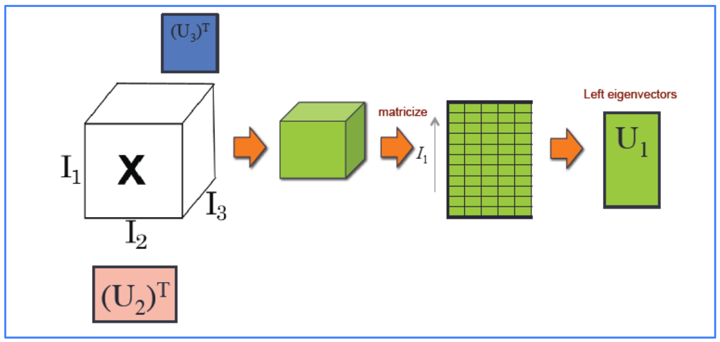
Tucker - ALS

- Step 5: Compute the revised core matrix

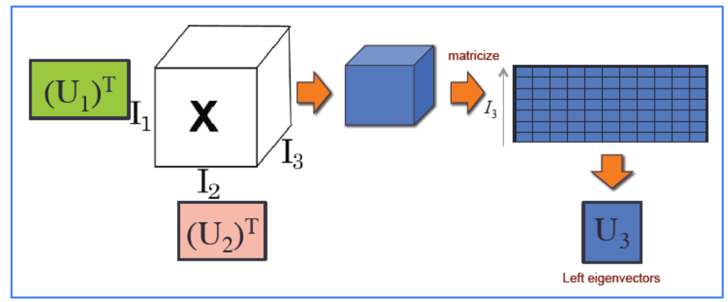


Tucker - ALS

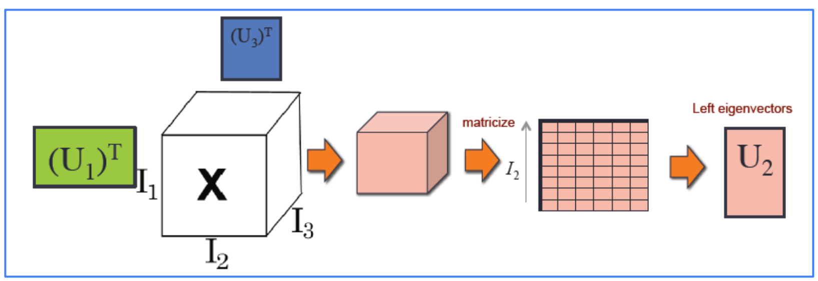
- Repeat the process until the norm of the core matrix stops increasing



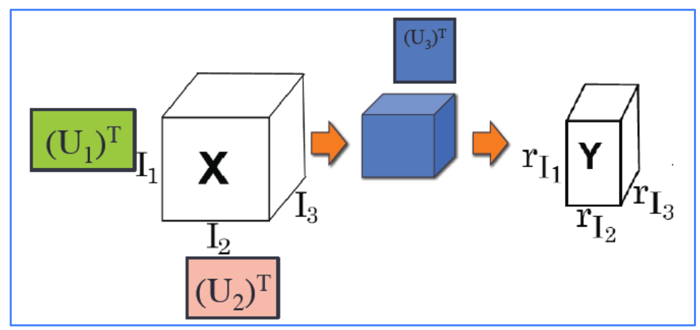
Solve for revised U_1



Solve for revised U_3



Solve for revised U_2



Compute the revised core

Observations

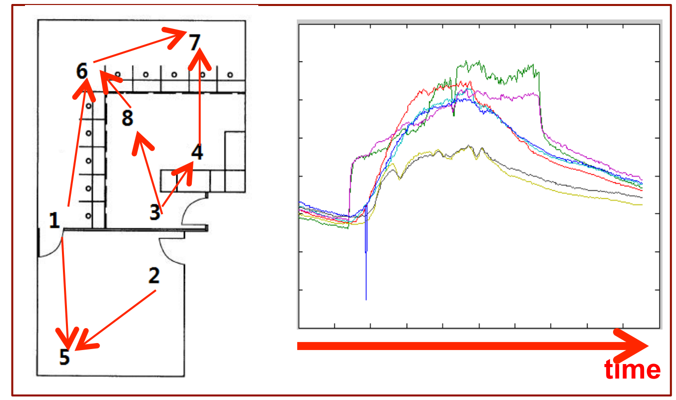
- Tensor decomposition algorithms are, especially for dense tensors, time consuming:

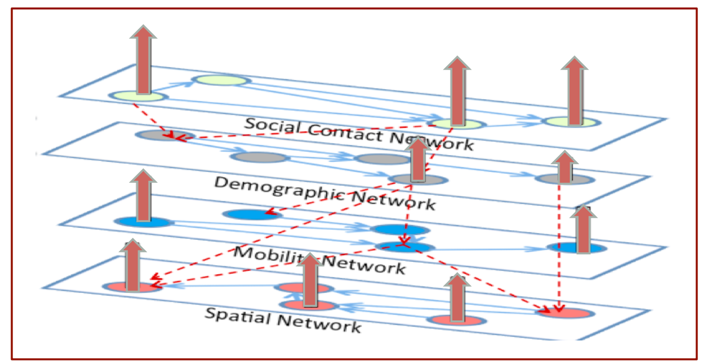
Problems:

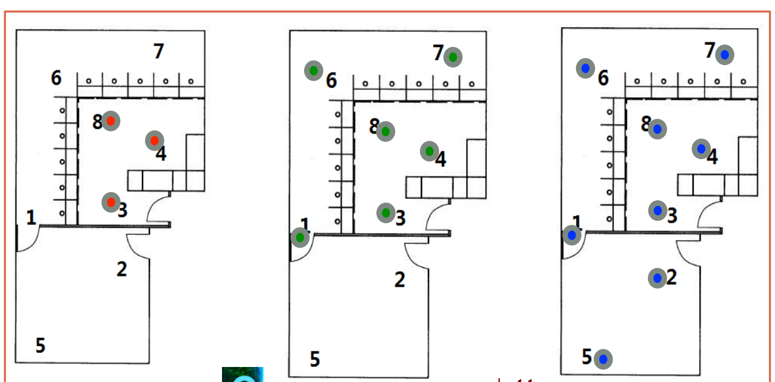
- these are very computationally expensive operations,
- they are also memory intensive:
  - Intermediary data blow-up!!!!!!

Common data characteristics…

- The key characteristics of the real worlddata sets include the following:
  - multi-variate
  - multi-modal
    - temporal,
    - spatial,
    - hierarchical,
    - graphical
  - multi-layer
  - multi-resolution
  - inter-dependent
    - observations of interest depend on and impact each other

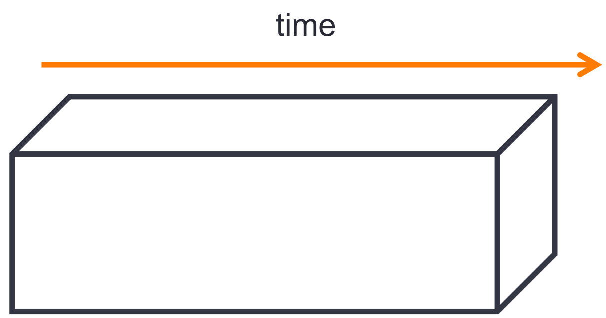






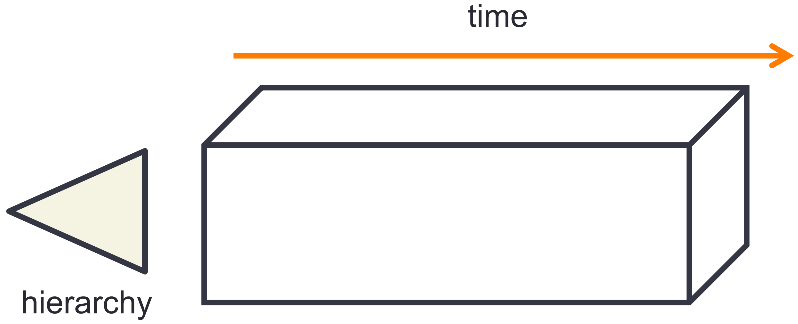
..and the metadata……

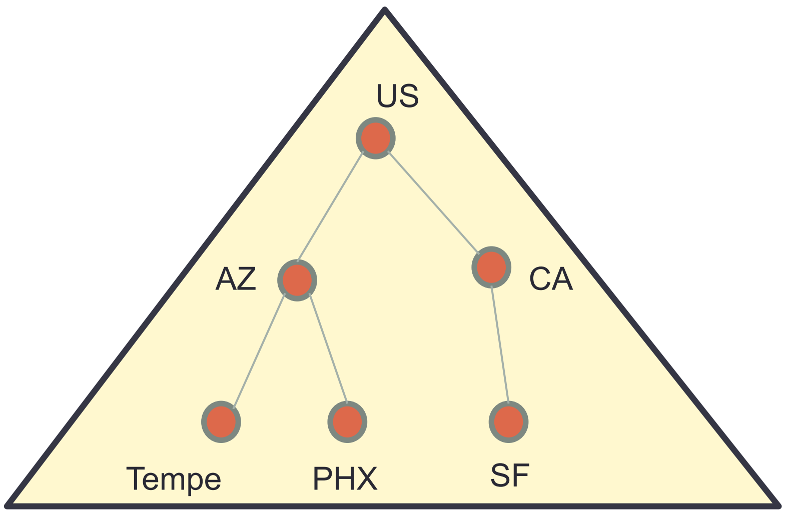
- Different modes of the tensor can have different types of metadata..



..and the metadata……

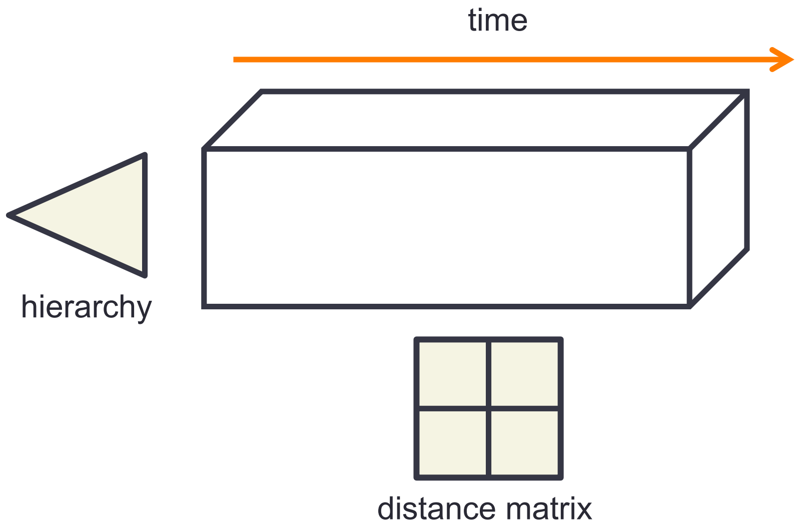
- Different modes of the tensor can have different types of metadata..

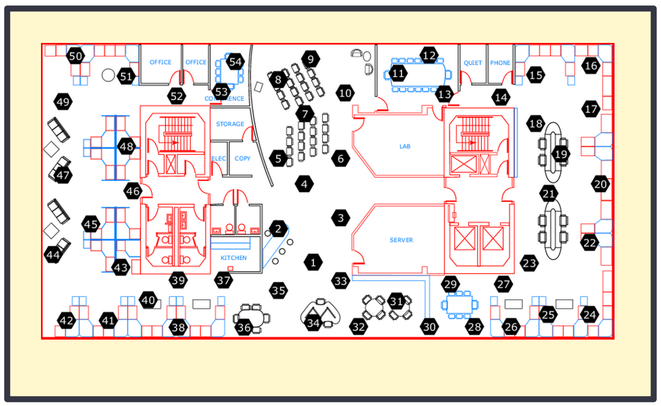




..and the metadata……

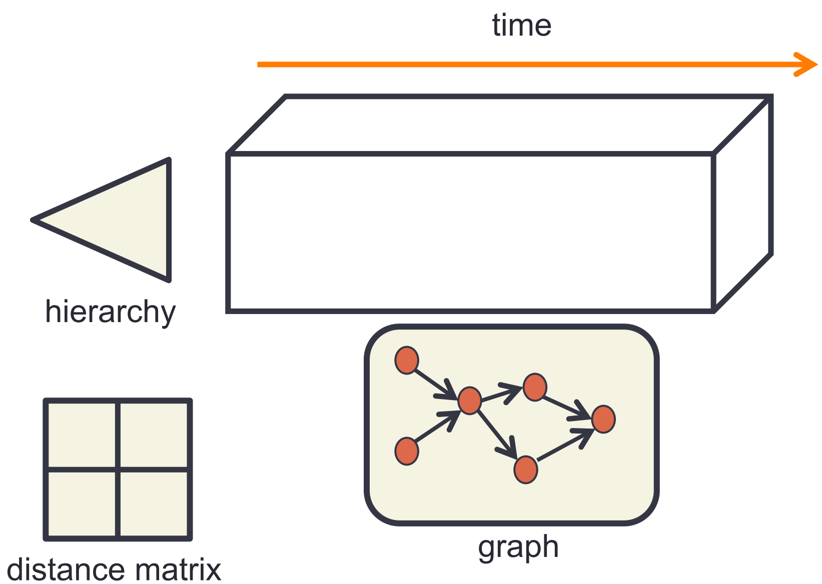
- Different modes of the tensor can have different types of metadata..

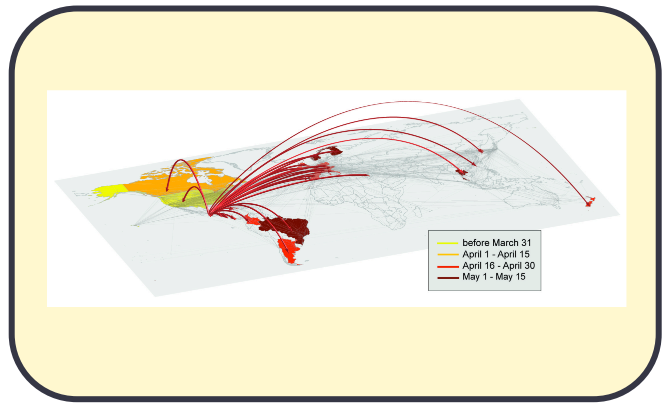




..and the metadata……

Differently-Modal Tensors (DMT)





Open research challenges…

Questions:

- how to best account for the different modalities of the data?
- can we leverage metadata to support multi-resolution and incremental tensor analysis operations?
- can we implement a memory hierarchy supported tensor analysis?
- can we co-optimize tensor analysis and other data manipulation operations?

What about other approaches?

- There are several technical approaches.
  - factorization, matrix/tensor decomposition
  - probabilistic (Bayesian/graphical model) learning
  - deep structured learning and neural networks.

(Probabilistic) Aspect Model

- Given a database,
  - $D = \{o_1, . . . , o_n\}$, of n objects
  - a feature set, $F = \{f_1, . . . , f_m\}$

  creates an object-feature matrix, P, with entries $p(o, f)$ denoting the joint probability of o and f in the corpus.

- The aspect model assumes that there is an unobserved class variable, $z \in Z = \{z_1, . . . , z_k\}$, underlying the data

Underlying generative model…

- Intuitively..
  - an object o ∈ D is selected with probability p(o),
  - a latent class z ∈ Z is selected with probability p(z|o), and
  - a feature f ∈ F is generated with probability p(f|z)

- Note that o and f can be observed in the database, but the latent semantic z is not directly observable

Probabilistic LSA (PLSA)

- Creates a object-feature matrix, P, with entries p(o, f) denoting the joint probability of o and f in the corpus.
- Key idea: p(o, f) can also be expressed in terms of the unobserved class variables

$$
\begin{aligned}
p(o,f) &= p(o)p(f|o) \\
&= p(o)\sum_{z\in Z} p(f|z)p(z|o) \\
&= p(o)\sum_{z\in Z} p(f|z)\frac{p(o|z)p(z)}{p(o)} \\
&= \sum_{z\in Z} p(z)p(f|z)p(o|z).
\end{aligned}
$$

$$
p(A|B) = \frac{p(B|A)p(A)}{p(B)}
$$

PLSA

- Decomposes the nxm matrix, P, as

$$P = U\Sigma V^T$$

where,

  - U is the nxk matrix of $p(o_i|z_l)$ entries,
  - V is the mxk matrix of $p(f_j\,|z_l)$ entries, and
  - Σ is the kxk matrix of $p(z_l)$ entries

Using expectation maximization (EM).

$$p(o,f) = \sum_{z \in Z} p(z)p(f|z)p(o|z).$$

Brief overview of EM

- Given a mapping function, **y = F(x)**, which relates
  - the observed data values **y**
  - to the values of the hidden data, **x**

  EM algorithms maximize **P(y, λ)**, where
  - **λ** are the estimates of the parameters that contribute to hidden data **x**.
    - how are the classes/topics distributed? Are they uniform in the data? Are some topics more frequent then the others?

- Intuitively EM searches for maximally likely parameter estimates for models with variables hidden from the observer.

EM overview…

- Iterative procedure, with two phases:
  - Phase 1: Expectation phase (E) formulates a function **Q**(**λ, λ'**) which links the current estimates, **λ**, of the hidden parameters to their revised estimates, **λ'**.

Given
- observed variables **y**
- current parameters **λ**

Q describes the (log) likelihood of the
- observed (**y**) and
- unobserved (**x**) variables

as a function of **λ'**

  - Phase 2: Maximization step (M) maximizes over possible values of **λ'**.

Expectation Step

- Iterative procedure, with two phases:
  - Phase 1: Expectation phase (E) formulates a function **Q(λ, λ')** which links the current estimates, **λ**, of the hidden parameters to their revised estimates, **λ'**.

Q(λ, λ') is the expected value of log(p(x,y|λ')) given the current estimate λ

$$Q(\lambda, \lambda') = \sum_{x \ s.t. \ F(x)=y} \boxed{p(x|y,\lambda)} \boxed{log(p(x,y|\lambda'))}.$$

Likelihood of hidden data **x**, given the current estimate λ and observed data **y**	(log) Likelihood of hidden data **x** and observed data **y**, given the revised estimate λ'

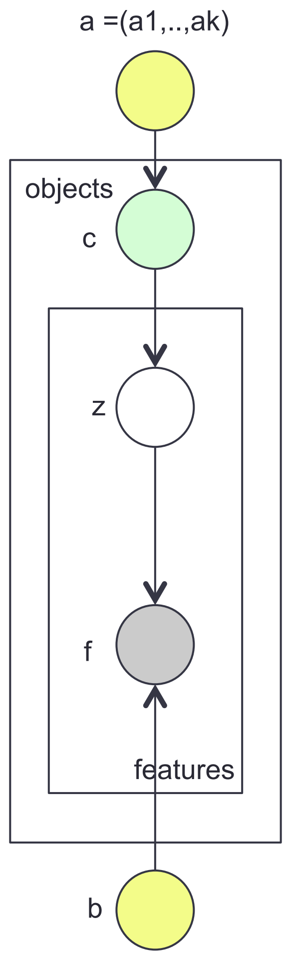
Maximization Step

- The maximization step
  - maximizes Q
  - by varying over λ'
    - often using some form of hill-climbing

LDA (graphical visualization of the generative process)

LDA assumes that each object is drawn using a generative process:

Given k hidden topics
- hidden aspect (topic) proportions are selected from a Dirichlet distribution,
- for each word
  - a hidden topic is assigned through a multinomial process, and
  - given the topic, a word is selected assuming a multinomial process.



Latent Dirichlet Allocation (LDA)

- The database,
  - observed: $D = \{o1, . . . , on\}$, of n objects,
  - observed: a feature set, $F = \{f1, . . . , fm\}$,
  - observed: each object oi is a set of Ni features
  - unobserved: topic variable, $z \in Z = \{z1, . . . , zk\}$,

$$f(x_1, \ldots, x_{k-1}; a_1, \ldots, a_k) = \frac{1}{B(a_1, \ldots, a_k)} \prod_{i=1}^{k} x_i^{a_i - 1} \quad ; 0 \le x_i \le 1 \; and \; x_k = 1 - \sum_{i=1}^{k-1} x_i$$

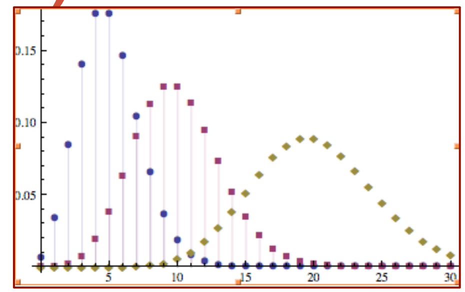
Latent Dirichlet Allocation (LDA)

- The database,
  - observed: D = {o1, . . . , on}, of n objects,
  - observed: a feature set, F = {f1, . . . , fm},
  - observed: each object oi is a set of Ni features
  - unobserved: topic variable, z ∈ Z = {z1, . . . , zk},
- Topic distribution, T, is a k dimensional Dirichlet distribution with parameters a1,..,ak

- Dirichlet probability density function models the belief that probabilities of *k* rival events are p1,..,pk given that the event ei has been observed ai-1 times.

$$f(r,N) = \begin{cases} \frac{e^{-r} r^N}{N!} & N \geq 0 \\ 0 & \text{True} \end{cases}$$

Latent Dirichlet Allocation (LDA)



- The database,
  - observed: D = {o1, . . . , on}, of n objects,
  - observed: a feature set, F = {f1, . . . , fm},
  - observed: each object oi is a set of Ni features
  - unobserved: topic variable, z ∈ Z = {z1, . . . , zk},
  - Topic distribution, T, is a k dimensional Dirichlet dist. with parameters a1,..,ak
- LDA assumes documents are <u>generated</u> as follows:
  - Document length, N is chosen with a Poisson distribution with parameter r

- Poisson distribution is often used when modeling the number of events in a fixed interval, when the average rate, r, of events is known

$$f(c_1, \ldots, c_k; N, p_1, \ldots, p_k) = \frac{N!}{c_1! \cdots c_k!} p_1^{c_1} \cdots p_k^{c_k} \quad \text{when} \sum_{i=1}^{k} c_i = N$$

$$f(c_1, \ldots, c_k; N, p_1, \ldots, p_k) = 0 \quad \text{otherwise}$$

Latent Dirichlet Allocation (LDA)

- The database,
  - observed: D = {o1, . . . , on}, of n objects,
  - observed: a feature set, F = {f1, . . . , fm},
  - observed: each object oi is a set of Ni features
  - unobserved: topic variable, z ∈ Z = {z1, . . . , zk},
  - Topic distribution, T, is a k dimensional Dirichlet dist. with parameters a1,..,ak
- LDA assumes documents are generated as follows:
  - Document length, N is chosen with a Poisson distribution with parameter r
  - Each of the N features are generated as follows:
    - A topic z is chosen with multinomial(T,N)

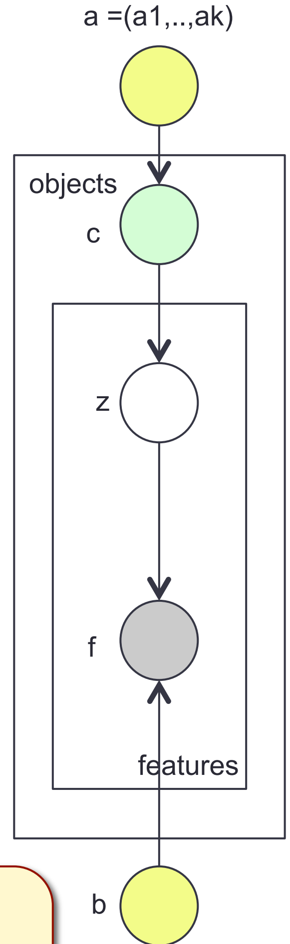
- This gives c1,.., ck, where cj indicates the number of times topic *zi* is observed over the *N* trials, where topics are selected with probability $(p_1, \ldots, p_k)$ where $(p_1, \ldots, p_k)$ are themselves selected with Dirichlet distribution T

Latent Dirichlet Allocation (LDA)

- The database,
  - observed: $D = \{o1, \ldots, on\}$, of n objects,
  - observed: a feature set, $F = \{f1, \ldots, fm\}$,
  - observed: each object oi is a set of Ni features
  - unobserved: topic variable, $z \in Z = \{z1, \ldots, zk\}$,
  - Topic distribution, T, is a k dimensional Dirichlet dist. with parameters a1,..,ak
- LDA assumes documents are generated as follows:
  - Document length, N is chosen with a Poisson distribution with parameter r
  - Each of the N features are generated as follows:
    - A topic z is chosen with multinomial(T,N)
    - A feature f is selected with probability p(f | z,b) where **b**ij = p(fj | zi),

- Given a topic, keyword distribution is assumed to be Dirichlet
- and, the specific keyword is selected by multinomial probability (conditioned on the topic z)

LDA (graphical visualization of the generative process)



LDA assumes that each object is drawn using a generative process:

Given k hidden topics

- hidden aspect (topic) proportions are selected from a Dirichlet distribution,
- for each word
  - a hidden topic is assigned through a multinomial process, and
  - given the topic, a word is selected assuming a multinomial process.

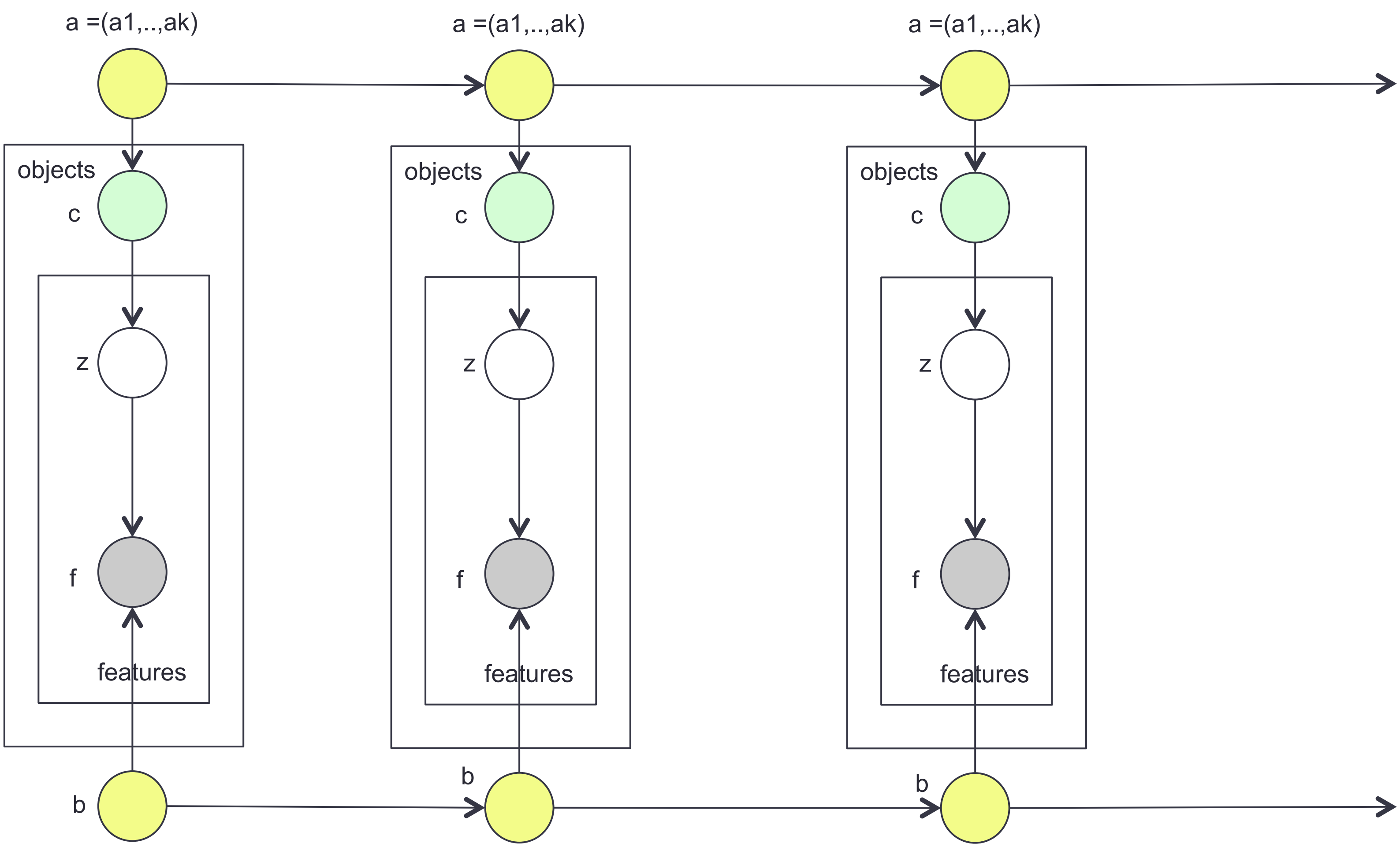
The function that ties observations to hidden parameters!:

$$p(f \mid a,b) = \int p(c \mid a)\left(\prod_{i=1}^{N} \sum_{j=1}^{k} p(f_i \mid z_j, b)\, p(z_j \mid c)\right) dc$$

Very costly!

- Gibbs sampling and variational inference are two well known techniques in solving topic model problems efficiently.
- Gibbs sampling, for example, enables obtaining samples that are approximated from a given joint probability distribution over more than one random variable.

Dynamic Topic Models (DTM)



What about other approaches?

- There are several technical approaches.
  - factorization, matrix/tensor decomposition
  - probabilistic (Bayesian/graphical model) learning
  - deep structured learning and neural networks.

….many of the algorithms are based on iterative processes, such as alternating least squares (ALS) or stochastic gradient descent (SGD), which approximate the best solution until a convergence condition is reached

Question: Can we develop metadata-supported and multi-scale techniques that can leverage the volume/cost trade-offs provided by storage hierarchies to provide high accuracy at minimum cost?