UBC MLRG (Winter 2020): Tensor Basics

Material largely follows Kolda and Bader, 2009:
"Tensor Decompositions and Applications"
Figures are taken from there
Presenter: Mark Schmidt

Recommender System Motivation: Netflix Prize

- Netflix Prize:
  - 100M ratings from 0.5M users on 18k movies.
  - Grand prize was $1M for first team to reduce squared error by 10%.
  - Started on October 2nd, 2006.
  - Netflix's system was first beat October 8th.
  - 1% error reduction achieved on October 15th.
  - Steady improvement after that.
    - ML methods soon dominated.
    - One of the reasons for explosion of interest in ML methods.

Lessons Learned from Netflix Prize

- Prize awarded in 2009:
  - Ensemble method that averaged 107 models.

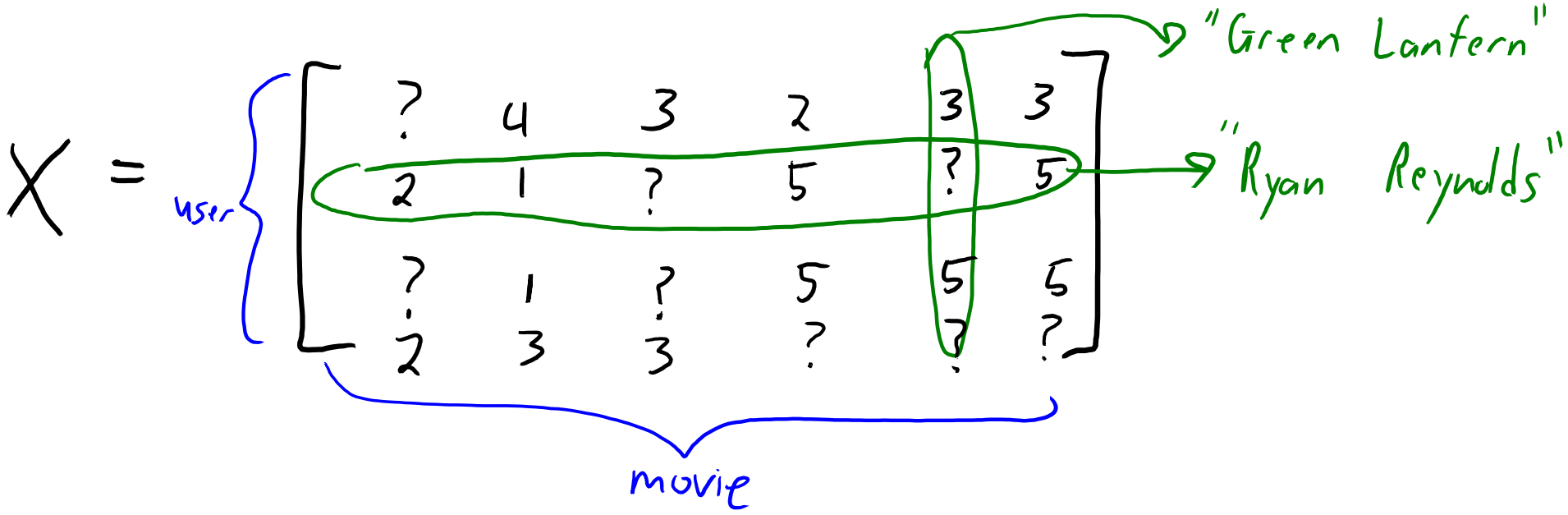
- Winning entry (and most entries) used collaborative filtering:
  - Methods that only looks at ratings, not features of movies/users.

- A simple collaborative filtering method that does really well (7%):
  - "Regularized matrix factorization". Now adopted by many companies.

http://bits.blogs.nytimes.com/2009/09/21/netflix-awards-1-million-prize-and-starts-a-new-contest/?_r=0

Collaborative Filtering Problem

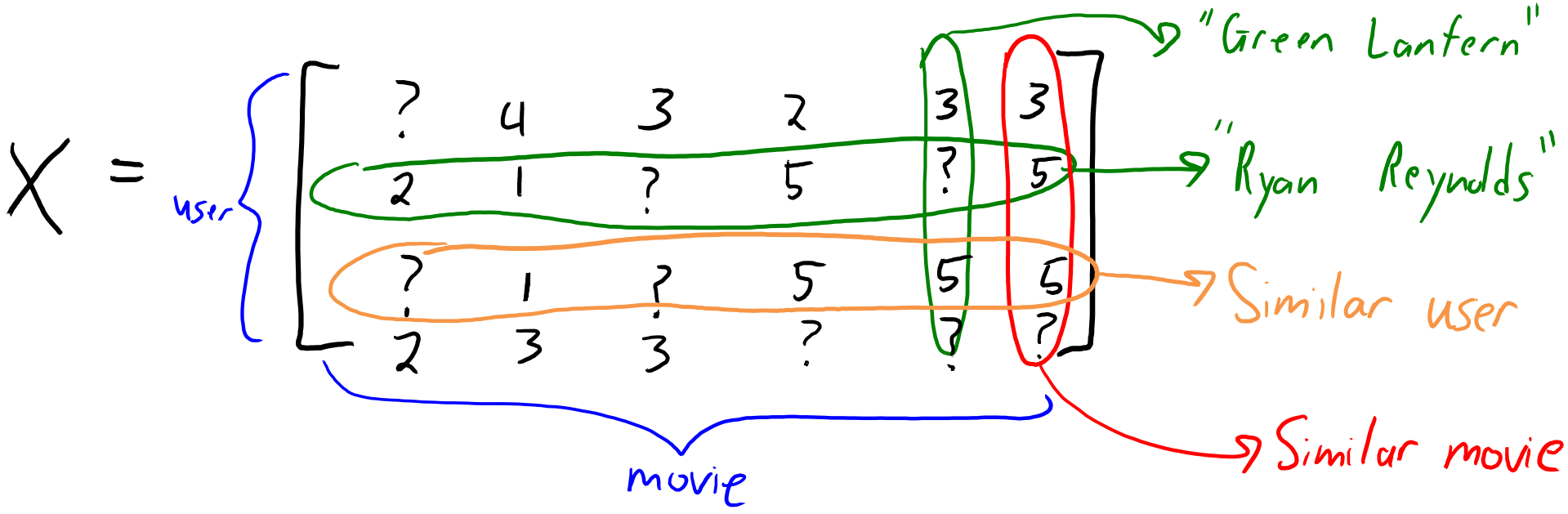
- Collaborative filtering is ‘filling in’ the user-item matrix:



- We have some ratings available with values {1,2,3,4,5}.
- We want to predict ratings “?” by looking at available ratings.

Collaborative Filtering Problem

- Collaborative filtering is ‘filling in’ the user-item matrix:



- What rating would “Ryan Reynolds” give to “Green Lantern”?
  - Why is this not completely crazy? We may have similar users and movies.

Matrix Factorization for Collaborative Filtering

- The standard matrix factorization model for entries in matrix 'X':

$$X \approx ZW \qquad x_{ij} \approx \langle w^j, z_i \rangle$$

(X: $n \times d$, Z: $n \times k$, W: $k \times d$)

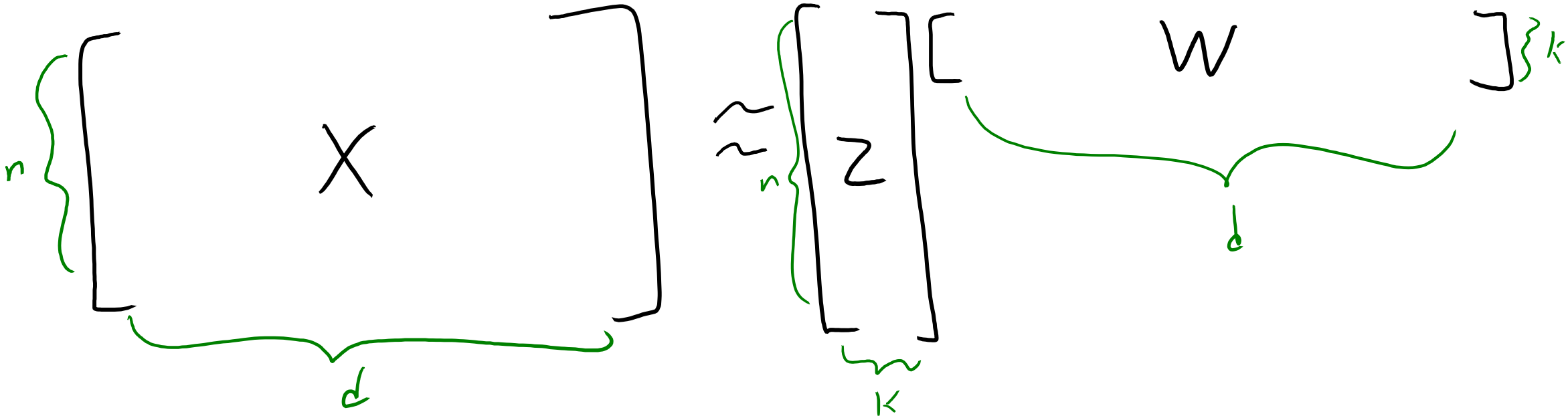
- User 'i' has latent features z_i. (z_{ic} could mean "likes Nicolas Cage")
- Movie 'j' has latent features w^j. (w_{jc} could mean "has Nicholas Cage")
- The standard loss function sums over available ratings 'R':

$$f(Z,W) = \sum_{(i,j) \in R} (\langle w^j, z_i \rangle - x_{ij})^2 + \frac{\lambda_1}{2}\|Z\|_F^2 + \frac{\lambda_2}{2}\|W\|_F^2$$

- And we add L2-regularization to both types of features.
  - Basically, this is regularized PCA on the available entries of 'X'.
  - Typically fit with SGD.
- This simple method gives you a 7% improvement on the Netflix problem.

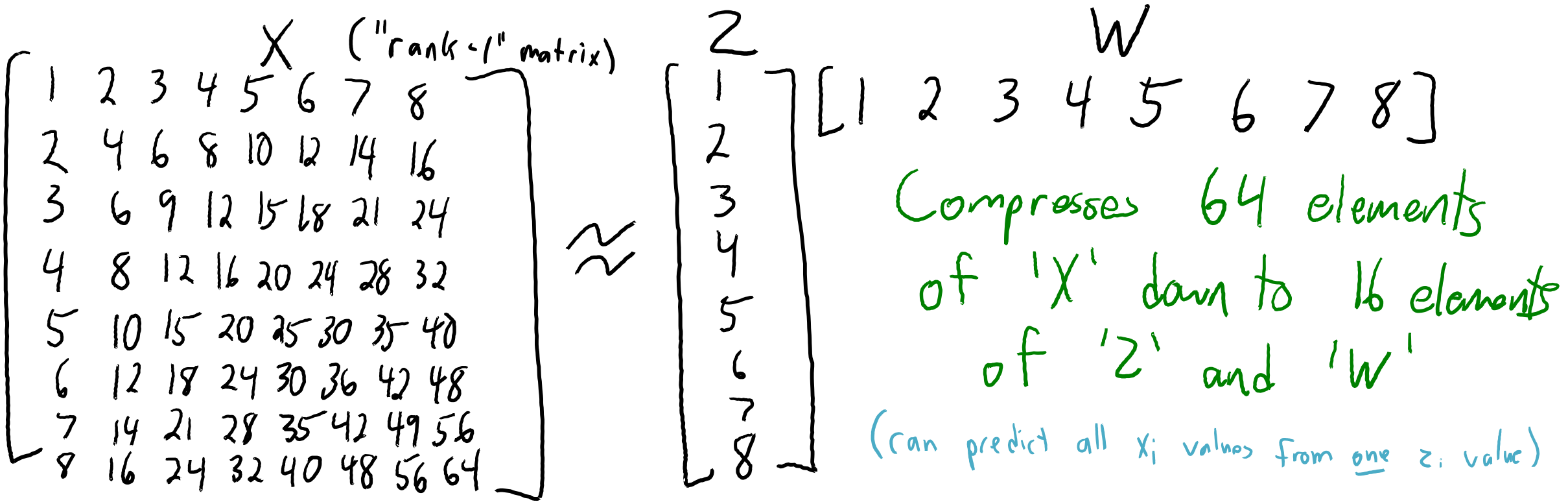
Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Dimensionality reduction: replace ‘X’ with lower-dimensional ‘Z’.
    - If k << d, then compresses data.



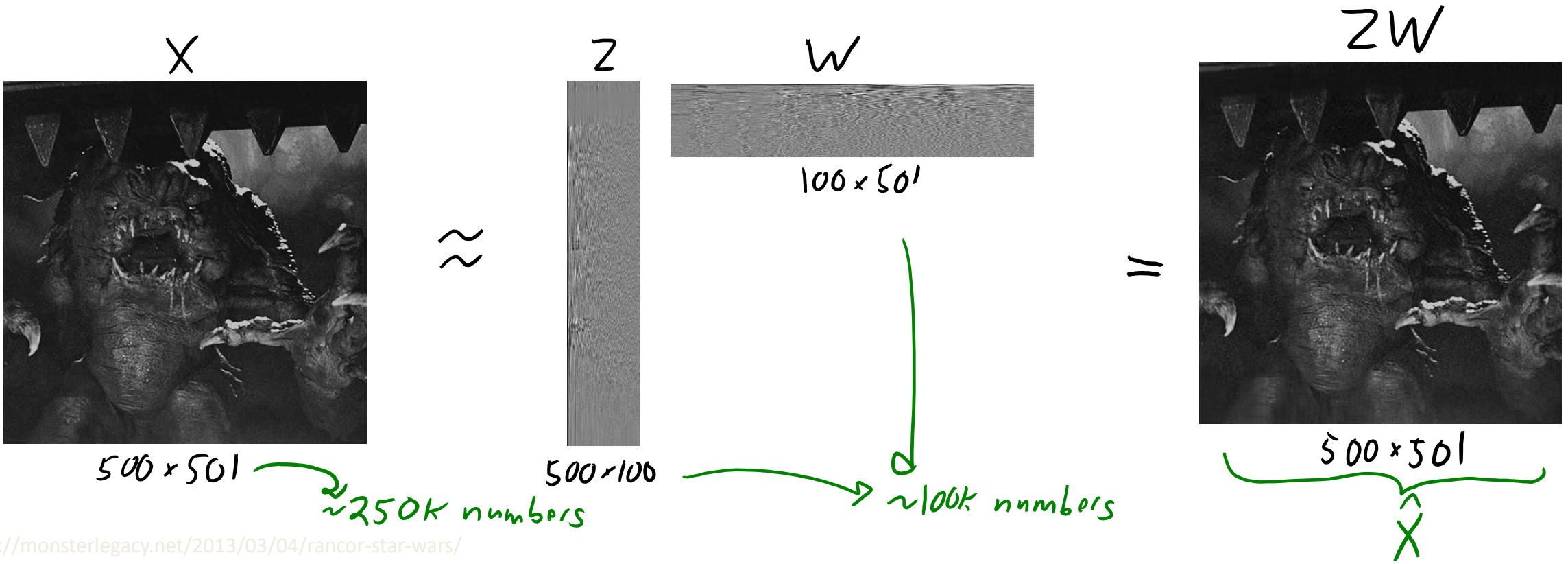
Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Dimensionality reduction: replace ‘X’ with lower-dimensional ‘Z’.
    - If k << d, then compresses data.



Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Dimensionality reduction: replace ‘X’ with lower-dimensional ‘Z’.
    - If k << d, then compresses data.



https://monsterlegacy.net/2013/03/04/rancor-star-wars/

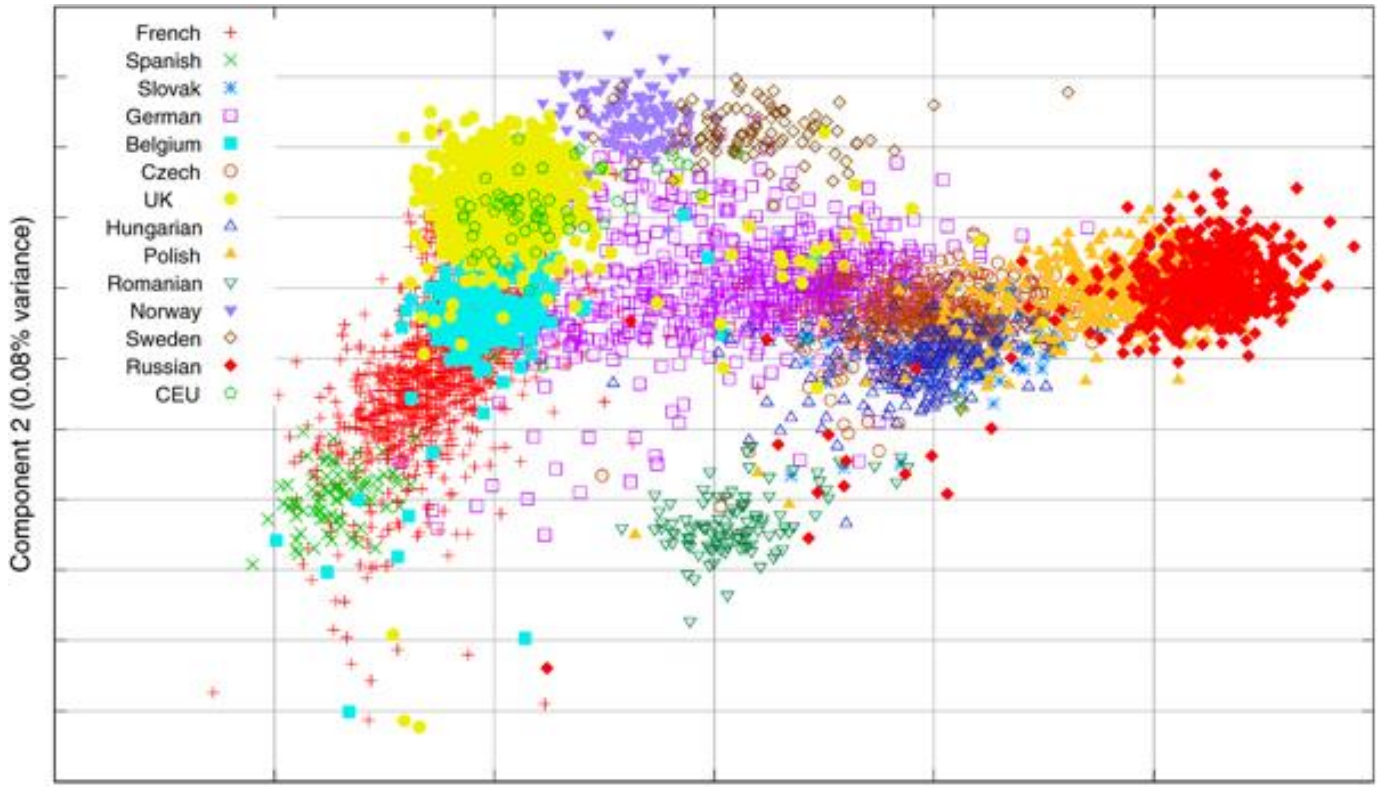
Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Dimensionality reduction: replace ‘X’ with lower-dimensional ‘Z’.
    - If k << d, then compresses data.

https://monsterlegacy.net/2013/03/04/rancor-star-wars/

Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Data visualization: plot z_i with k = 2 to visualize high-dimensional objects.



http://infoproc.blogspot.ca/2008/11/european-genetic-substructure.html

Matrix Factorization Applications

- Applications of Matrix Factorization :
  - Data interpretation: we can try to assign meaning to latent factors w_c.
    - Hidden “factors” that influence all the variables.

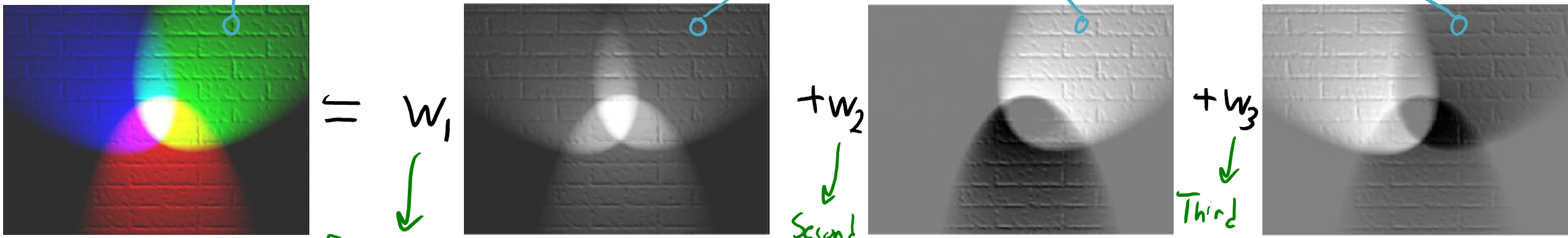
Trait	Description
Openness	Being curious, original, intellectual, creative, and open to new ideas.
Conscientiousness	Being organized, systematic, punctual, achievement-oriented, and dependable.
Extraversion	Being outgoing, talkative, sociable, and enjoying social situations.
Agreeableness	Being affable, tolerant, sensitive, trusting, kind, and warm.
Neuroticism	Being anxious, irritable, temperamental, and moody.

https://new.edu/resources/big-5-personality-traits

"Most Personality Quizzes Are Junk Science. I Found One That Isn't."

Matrix Factorization Applications

- Applications of Matrix Factorization : seeing colours.

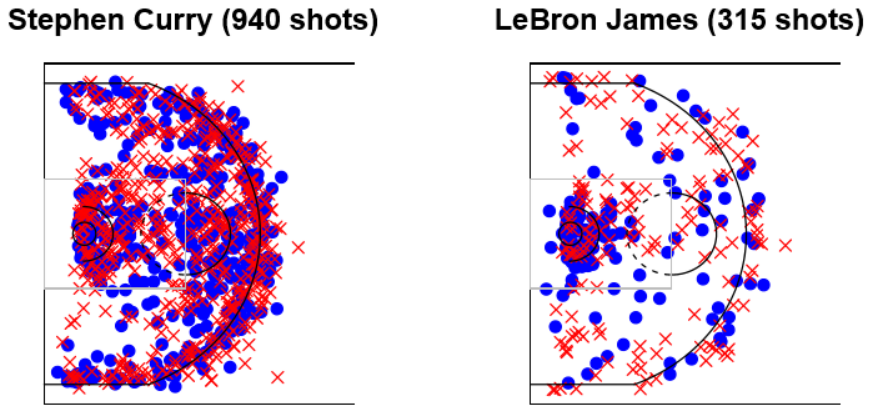


Analogous to means in k-means.

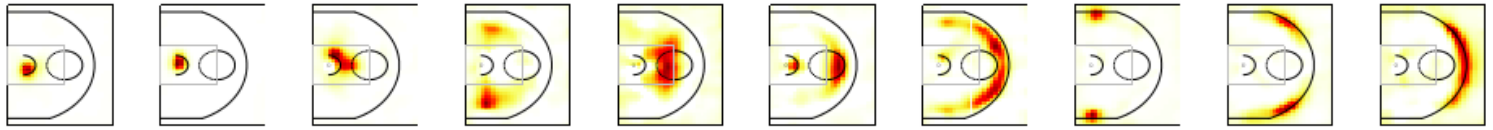
https://en.wikipedia.org/wiki/RGB_color_model

Matrix Factorization Applications

- NBA shot charts:



- MF (non-negative w/ “KL divergence” with k=10 + smoothed data):

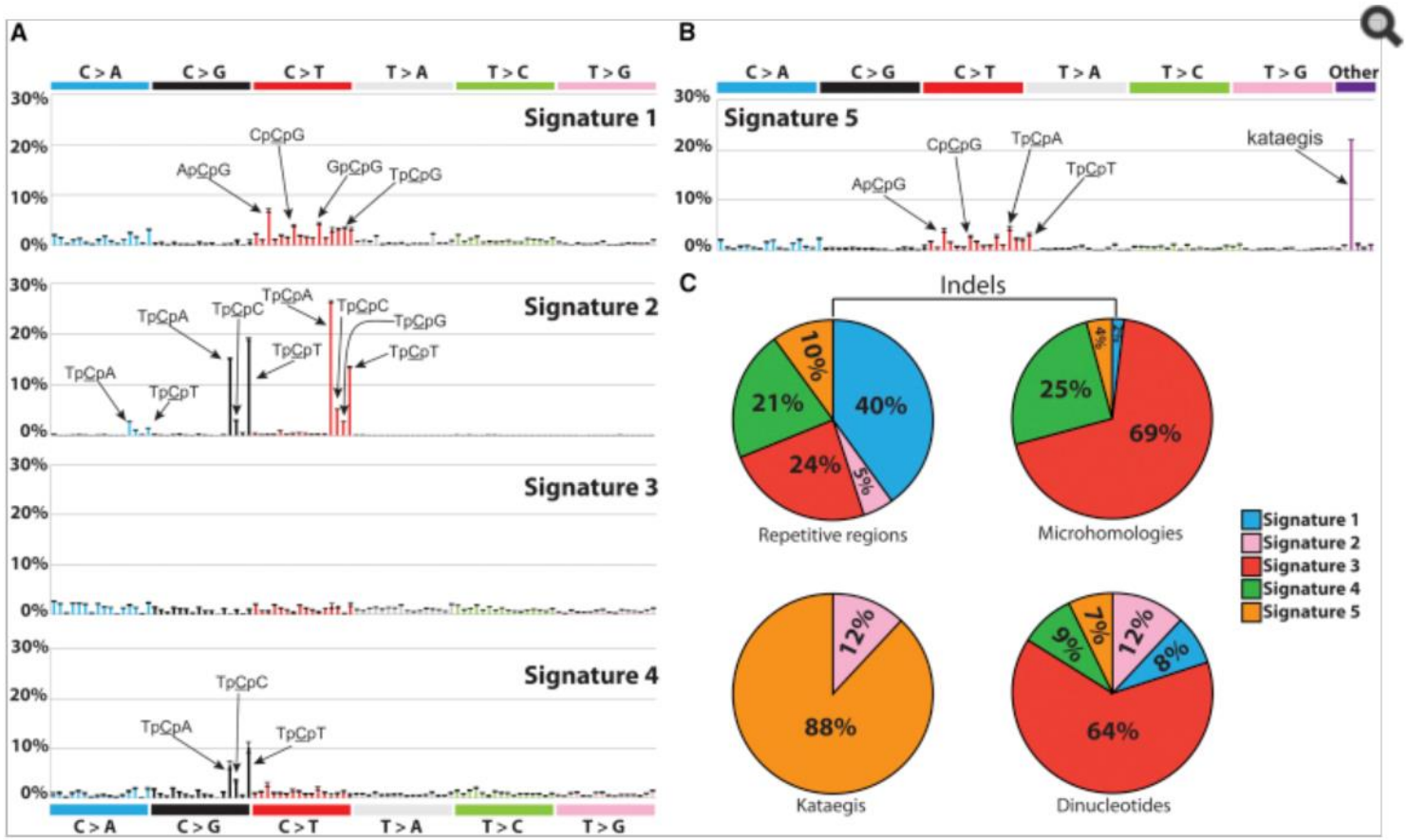


LeBron James	0.21	0.16	0.12	0.09	0.04	0.07	0.00	0.07	0.08	0.17
Brook Lopez	0.06	0.27	0.43	0.09	0.01	0.03	0.08	0.03	0.00	0.01
Tyson Chandler	0.26	0.65	0.03	0.00	0.01	0.02	0.01	0.01	0.02	0.01
Marc Gasol	0.19	0.02	0.17	0.01	0.33	0.25	0.00	0.01	0.00	0.03
Tony Parker	0.12	0.22	0.17	0.07	0.21	0.07	0.08	0.06	0.00	0.00
Kyrie Irving	0.13	0.10	0.09	0.13	0.16	0.02	0.13	0.00	0.10	0.14
Stephen Curry	0.08	0.03	0.07	0.01	0.10	0.08	0.22	0.05	0.10	0.24
James Harden	0.34	0.00	0.11	0.00	0.03	0.02	0.13	0.00	0.11	0.26
Steve Novak	0.00	0.01	0.00	0.02	0.00	0.00	0.01	0.27	0.35	0.34

http://jmlr.org/proceedings/papers/v32/miller14.pdf

Matrix Factorization Applications

- What are common sets of mutations in different cancers?
  - May lead to new treatment options.

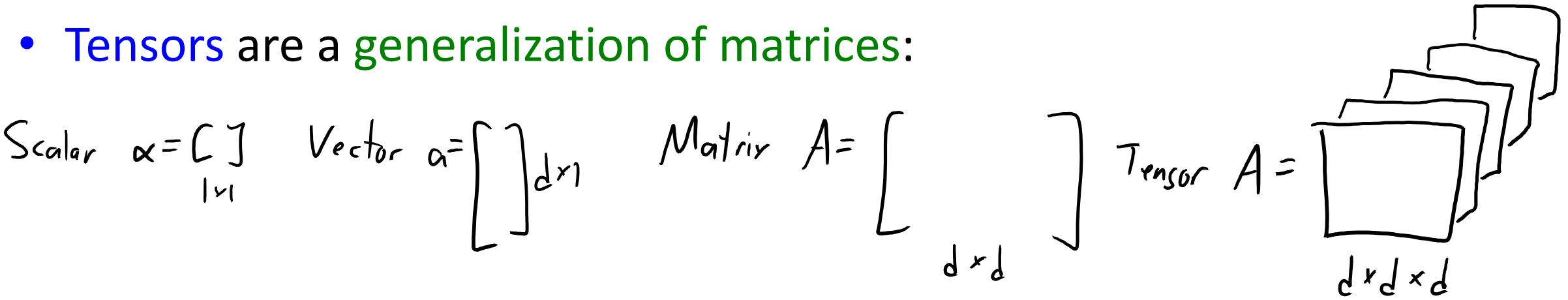


https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3588146/

(pause)

Tensor Factorization

- Tensors are a generalization of matrices:



- Generalization of matrix factorization is tensor factorization:

$$x_{ijk} \approx \sum_{c=1}^{r} w_{jc} z_{ic} v_{kc}$$

- Useful if there are other relevant variables:
  - Instead of ratings based on {user,movie}, ratings based on {user,movie,country}.
  - Useful if you have groups of users, or if ratings change over time.

Tensor Order

- The order of a tensor is "how many indices" you have:
  - A 1st-order tensor is a vector, T[i].
  - A 2nd-order tensor is a matrix, T[i,j].
  - A 3rd-order tensor has 3 indices, T[i,j,k].
  - A 4th-order tensor has 4 indices, T[i,j,k,m].
- Other names for order:
  - "Way" (as in 3-way tensor) and "modes" (as in mode-3 tensor).
  - Linear algebra people generally don't use "dimension".
    - Probably to avoid confusion with "dimension of subspace".
- Applications:
  - Psychometrics, chemometrics, signal processing, numerical analysis, computer vision, machine learning, neuroscience, graph theory, and so on.
- To save space I use "tensor" in place of "3rd-order tensors" in examples.
  - But higher-order tensor properties tend to be analogous.

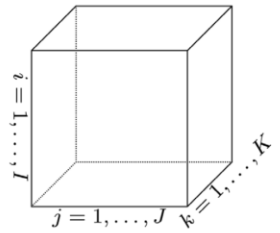
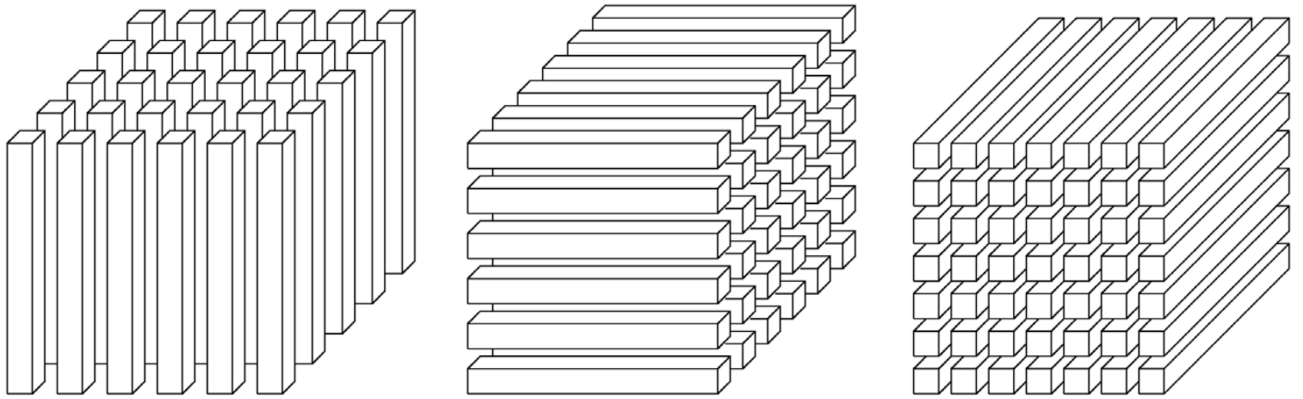


Fig. 1.1 *A third-order tensor:* $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$.

Fibers and Slices

- Fiber: vector formed by fixing every index but one to a constant.
  - Matrix has “columns” (mode-1 fibers) and “rows” (mode-2 fibers).
    - Columns correspond to M[:,j] and rows correspond to M[i,:].
  - Tensor has “columns” (mode-1), “rows” (mode-2), and “tubes” (mode-3).
    - Columns are M[:,j,k], rows are M[i,:,k], and tubes are M[i,j,:].
    - Notice you need to fix 2 indices for mode-3, three indices for mode-4, and so on.

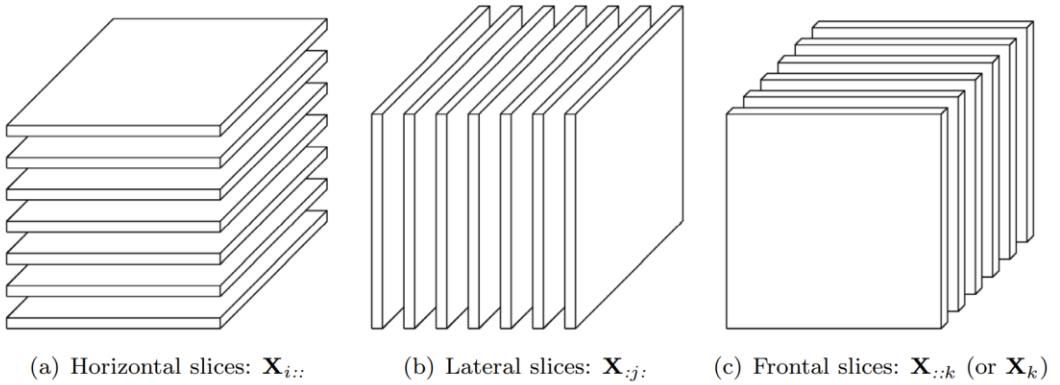


(a) Mode-1 (column) fibers: $\mathbf{x}_{:jk}$ (b) Mode-2 (row) fibers: $\mathbf{x}_{i:k}$ (c) Mode-3 (tube) fibers: $\mathbf{x}_{ij:}$

Fig. 2.1 *Fibers of a 3rd-order tensor.*

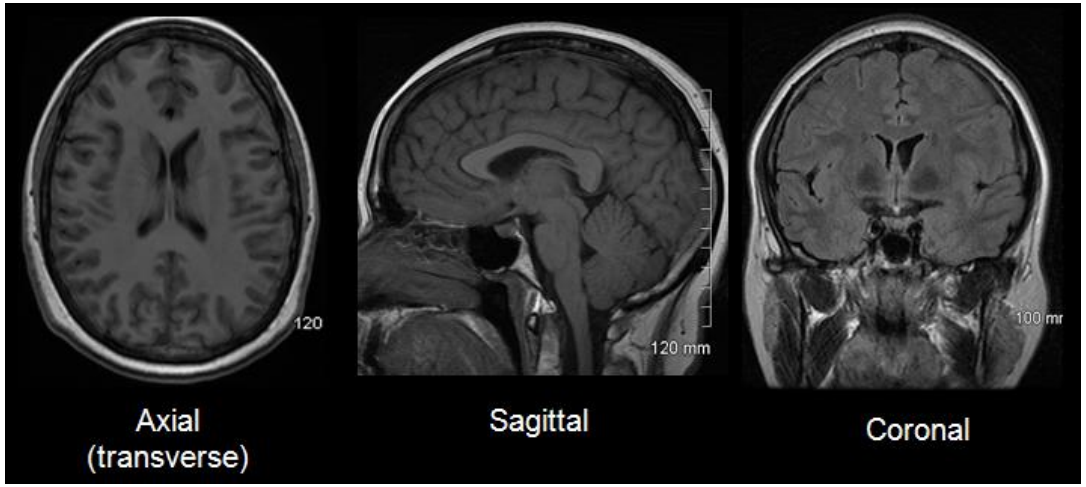
Fibers and Slices

- Slice: matrix formed by fixing every index but two to a constant.
  - Matrix only has itself as a slice, M[:,:].
  - Tensor has “horizontal slices”, “vertical slices”, and “frontal slices”.



(a) Horizontal slices: $\mathbf{X}_{i::}$ (b) Lateral slices: $\mathbf{X}_{:j:}$ (c) Frontal slices: $\mathbf{X}_{::k}$ (or $\mathbf{X}_k$)

Fig. 2.2 *Slices of a 3rd-order tensor.*



  - In medicine these are called “axial”, “saggital”, and “coronal”.

https://sites.google.com/site/postgraduatetraining/image-acquisition/the-basics

Inner Product and Norm

- Euclidean-norm of a vector: $\|x\|_2 = \sqrt{\sum_i x_i^2}$
- Frobenius-norm of a matrix: $\|X\|_F = \sqrt{\sum_i \sum_j x_{ij}^2}$
- 3rd-order tensor norm: $\|\mathcal{X}\| = \sqrt{\sum_i \sum_j \sum_k x_{ijk}^2}$
  - I wish these were all called "Euclidean"

Notice that $\langle \mathcal{X}, \mathcal{X} \rangle = \|\mathcal{X}\|^2$

- Inner-product between vectors: $\langle x, y \rangle = \sum_i x_i y_i = x^T y$
- Inner-product between matrices: $\langle X, Y \rangle = \sum_i \sum_j x_{ij} y_{ij} = Tr(X^T Y)$
- Inner-product between tensors: $\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_i \sum_j \sum_k x_{ijk} y_{ijk}$

Symmetry and Super-Symmetry

- A matrix is square if #rows = #columns.
- A tensor is cubical if #rows = #columns = #tubes.
  - Same size along each index.

- A matrix is symmetric if $x_{ij} = x_{ji}$ for all 'i' and 'j'.
- A tensor is symmetric in modes 1 and 2 if $x_{ijk} = x_{jik}$ for all 'i', 'j', and 'k'.
  - Slices $X(:,:,k) = X(:,:,k)^T$ for all 'k'.
  - Can be symmetric in any two are more modes.

- A tensor is super-symmetric if it's symmetric in all modes:
  - Permuting any indices does not change value. $x_{ijk} = x_{jik} = x_{kij} = x_{ikj} = x_{jki} = x_{kji}$

Diagonal Tensors

- A matrix is diagonal if $x_{ij} \neq 0$ only when i=j.
  - "All non-zeroes are along diagonal".
- A tensor is diagonal if $x_{ijk} \neq 0$ only when i=j=k.
  - "All non-zeros are along super-diagional".

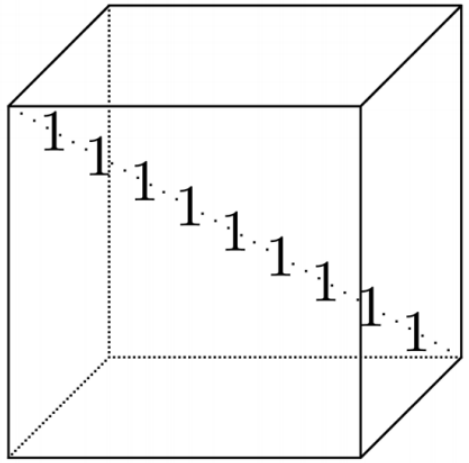


Fig. 2.4 *Three-way tensor of size $I \times I \times I$ with ones along the superdiagonal.*

Matricization

- Vectorization:
  - Convert matrix to vector by stacking columns. $X = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \rightarrow \text{vec}(X) = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$
  - Can also vectorize tensors by putting all elements in vector.
- Matricization (also known as "flattening" or "unfolding"):
  - Convert tensor to matrix by arranging mode-n fibers to be columns.

The concept is easier to understand using an example. Let the frontal slices of $\mathcal{X} \in \mathbb{R}^{3\times4\times2}$ be

$$\mathbf{X}_1 = \begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}, \quad \mathbf{X}_2 = \begin{bmatrix} 13 & 16 & 19 & 22 \\ 14 & 17 & 20 & 23 \\ 15 & 18 & 21 & 24 \end{bmatrix}. \tag{2.1}$$

  - Isn't a standard "ordering" to do this.

Then the three mode-n unfoldings are

$$\mathbf{X}_{(1)} = \begin{bmatrix} 1 & 4 & 7 & 10 & 13 & 16 & 19 & 22 \\ 2 & 5 & 8 & 11 & 14 & 17 & 20 & 23 \\ 3 & 6 & 9 & 12 & 15 & 18 & 21 & 24 \end{bmatrix},$$

$$\mathbf{X}_{(2)} = \begin{bmatrix} 1 & 2 & 3 & 13 & 14 & 15 \\ 4 & 5 & 6 & 16 & 17 & 18 \\ 7 & 8 & 9 & 19 & 20 & 21 \\ 10 & 11 & 12 & 22 & 23 & 24 \end{bmatrix},$$

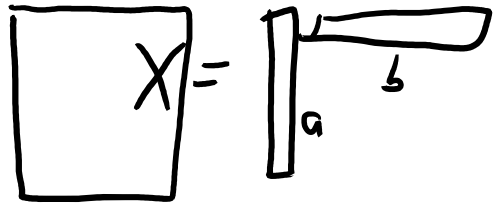
$$\mathbf{X}_{(3)} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & \cdots & 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 & 17 & \cdots & 21 & 22 & 23 & 24 \end{bmatrix}.$$

(we’re now going to get back to the tensor factorizations used in recommender systems)

Outer Product and Rank-1

- Outer product between two vectors gives ("rank-1") matrix:

$X_{ij} = a_i b_j$ or $X = ab^T = a \circ b$



- Outer product between three vectors gives ("rank-1") tensor:

$x_{ijk} = a_i b_j c_k$ or $\mathcal{X} = a \circ b \circ c$

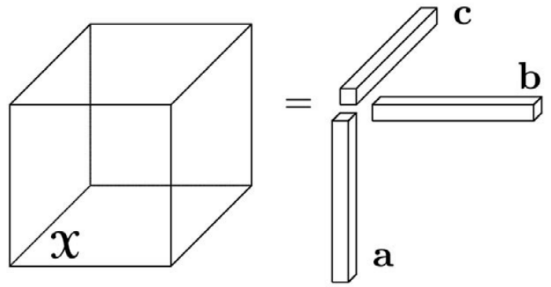


Fig. 2.3 *Rank-one third-order tensor,* $\mathcal{X} = \mathbf{a} \circ \mathbf{b} \circ \mathbf{c}$. *The* (i, j, k) *element of* $\mathcal{X}$ *is given by* $x_{ijk} = a_i b_j c_k$.

- These "rank-1" tensors are the ingredients behind CP factorizations.

CP Factorization

- SVD factorization approximates matrices as sum of rank-1 matrices.
- CP factorization approximates tensor as sum of rank-1 tensors:

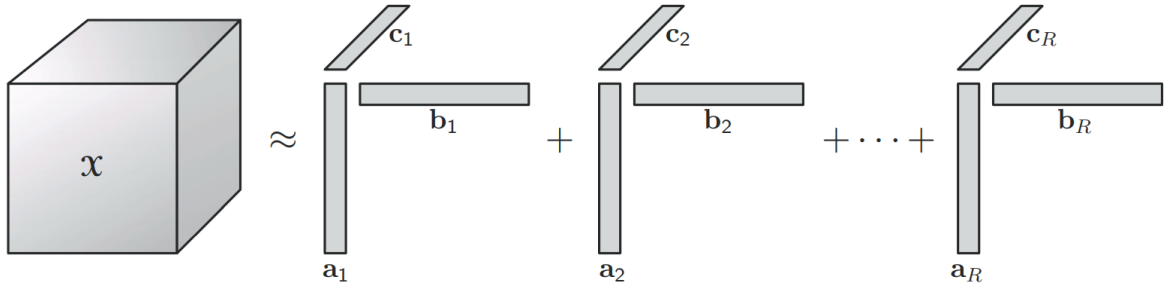


Fig. 3.1 *CP decomposition of a three-way array.*

- Mathematically, we are using the approximation:

$$x_{ijk} = \sum_r a_{ir} b_{jr} c_{kr} \quad \text{or} \quad \mathcal{X} = \sum_r a_r \circ b_r \circ c_r$$

- Has been re-invented under several names:
  - CANDECOMP (C), PARAFAC (P), polyadic form, topographic components.

Tensor Rank

- Matrix rank:
  - Minimum number of rank-1 matrices needed to decompose matrix.
    - It can be at most min{nRows,nCols}.
- Tensor rank:
  - Minimum number of rank-1 tensors needed to decompose tensor.
    - It can be at most min{nRows*nCols, nRows*nTubes, nCols*nTubes}.

- Notable differences with matrices: hard to determine tensor rank!
  - There are 9 x 9 x 9 tensors whose rank is unknown.
  - Even ranks of random tensors are weird (many basic results unknown).
  - Practice: upper-bound by trying to fit CR factorization with different ranks.

CP Uniqueness + Low-Rank Approximation

- SVD of matrix is non-unique.
  - In addition to permutation/scaling, can always rotate factors.
    - If $X = ZW$, then $X = (ZR^T)(RW)$ for any orthogonal matrix 'R' (since $R^TR = I$).

- Many not-ridiculous assumptions exist under which CP is unique.
  - Only one way to write tensor as sum of rank 1, up to permutation/scaling.

- Matrices: best rank-r approximation includes best rank-(r-1) approx.
  - So you can find the rank-1 matrices sequentially.
- Tensors: not true!
  - E.g., best rank-1 approximation may not be part of best rank-2 approximation.

Border Rank

- Even weirder:
  - Some tensors can be approximated arbitrarily by a lower-rank tensor.
    - Weights go to ∞ with opposite signs.
    - Called a “degenerate” tensor.

- “Border” rank:
  - Minimum number of rank-1 tensors to arbitrarily approximate tensor.
  - For matrices, rank == border rank.
    - For any type of tensor (border rank) ≤ rank.

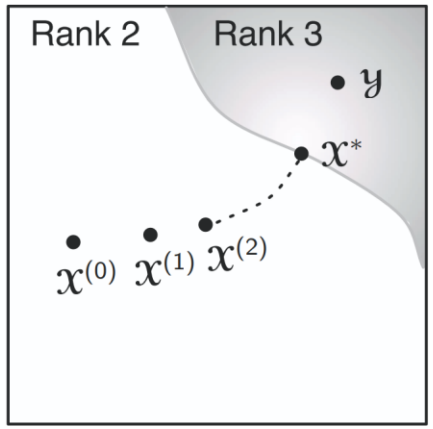


Fig. 3.2 *Illustration of a sequence of tensors converging to one of higher rank* [144].

- Matrix multiplication:
  - 2x2 case: 4x4x4 tensor w/ rank and border rank = 7 (Strassen).
  - 3x3 case: 9x9x9 tensor w/ rank in [19-23] and border rank in [14,21].

CP Factorization: Computation

- Pick a rank ‘r’, initialize randomly (to avoid starting at saddle point):
  - Alternating minimization:
    - Fix two of the components (e.g., all a_r and b_r) and solve for the third (all c_r).
      - Equivalent to a least squares problem.
      - Repeat, cycling through the components.
  - Stochastic gradient descent:
    - Sample a random ‘i’ and ‘j’ and ‘k’, then update a_i and b_j and c_k based on x_{ijk}.
    - O(k*nRows + k*nCols + k*nTubes) storage and O(k)-time updates.
- Not guaranteed to find optimal rank-r approximation.
  - Some global methods exist for special cases of rank-3 tensors.
    - E.g., can formulate as generalized eigenvalue problem if assume first two factors are full-rank.
- For ML problems, probably also want to add regularization.

Preview: Tucker Factorization

- CP factorization is one generalization of SVD to tensors:

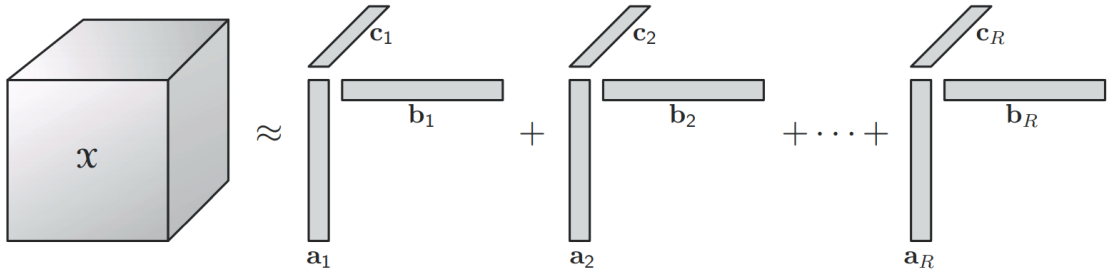


Fig. 3.1 *CP decomposition of a three-way array.*

- Tucker factorization is another generalization of SVD to tensors:

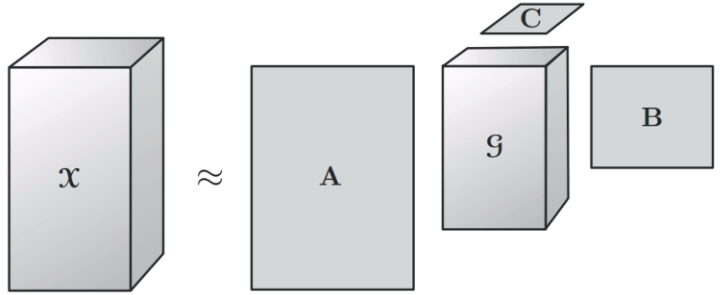


Fig. 4.1 *Tucker decomposition of a three-way array.*

- Other generalizations also exist!