

Rank

Stephen Boyd and Sanjay Lall
Revision by Amirhossein Afsharrad

EE263
Stanford University

Rank of a matrix

we define the *rank* of $A \in \mathbb{R}^{m \times n}$ as

$$\text{rank}(A) = \dim \text{range}(A)$$

(nontrivial) **facts:**

- ▶ $\text{rank}(A) = \text{rank}(A^T)$
- ▶ $\text{rank}(A)$ is maximum number of independent columns (or rows) of A
hence $\text{rank}(A) \leq \min(m, n)$

Rank of a matrix

- ▶ **rank**(A) is maximum number of independent columns of A
 - ▶ to see this, notice that if the columns of A are independent, then the number of columns r is the rank, since the columns are a basis for the range
 - ▶ and if not, then there must be one column in the span of the others, so remove it, and repeat if necessary
 - ▶ all other independent sets of columns must have no more than r elements.
- ▶ proof of **rank**(A) = **rank**(A^T) uses QR, to come

Conservation of dimension

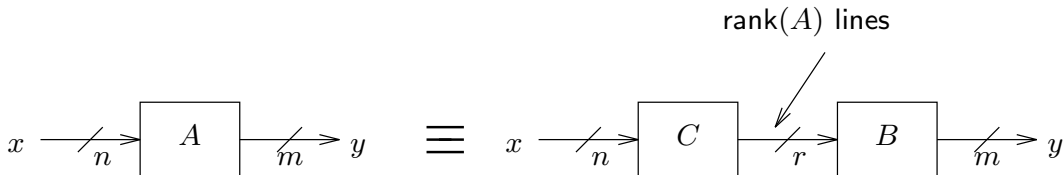
$$\mathbf{dim\ range}(A) + \mathbf{dim\ null}(A) = n$$

- ▶ **rank**(A) is dimension of set 'hit' by the mapping $y = Ax$
- ▶ **dim null**(A) is dimension of set of x 'crushed' to zero by $y = Ax$
- ▶ 'conservation of dimension': each dimension of input is either crushed to zero or ends up in output
- ▶ roughly speaking:
 - ▶ n is number of degrees of freedom in input x
 - ▶ **dim null**(A) is number of degrees of freedom lost in the mapping from x to $y = Ax$
 - ▶ **rank**(A) is number of degrees of freedom in output y
- ▶ proof using QR

Coding interpretation of rank

$$\text{rank}(BC) \leq \min\{\text{rank}(B), \text{rank}(C)\}$$

- ▶ hence if $A = BC$ with $B \in \mathbb{R}^{m \times r}$, $C \in \mathbb{R}^{r \times n}$, then $\text{rank}(A) \leq r$
- ▶ converse: if $\text{rank}(A) = r$ then $A \in \mathbb{R}^{m \times n}$ factors as $A = BC$ with $B \in \mathbb{R}^{m \times r}$, $C \in \mathbb{R}^{r \times n}$
- ▶ $\text{rank}(A) = r$ is minimum size vector needed to faithfully reconstruct y from x



Coding interpretation of rank

- ▶ $\text{rank}(BC) \leq \text{rank}(B)$ because $\text{range}(BC) \subset \text{range}(B)$
- ▶ transpose implies $\text{rank}(BC) \leq \text{rank}(C)$
- ▶ factorization converse comes from QR

Low-rank factorization: latent factors

if $A = BC$ with inner dimension r , then r numbers (the *latent factors*) mediate every input-output pair

recommender systems: a ratings matrix factors as $R \approx UV^T$

- ▶ a row of U is a user's taste, a row of V a movie's genre mix
- ▶ r hidden factors explain all ratings, and fill in the missing ones

the same idea represents users, items, and words as short *embedding* vectors in $\mathbb{R}^r$; store $(m+n)r$ numbers instead of mn

$$\begin{array}{c} R \\ \begin{array}{|c|c|c|c|c|} \hline 5 & ? & 3 & ? & 2 \\ \hline ? & 1 & 1 & ? & 5 \\ \hline ? & ? & 3 & 4 & 5 \\ \hline 4 & 4 & 3 & 3 & 5 \\ \hline \end{array} \\ \text{users} \times \text{movies} \end{array} \approx \begin{array}{|c|} \hline U \\ \hline \end{array} \cdot \begin{array}{|c|} \hline V^T \\ \hline \end{array}$$

$m \times r$ $r \times n$

r latent factors
 $r \ll m, n$

Application: fast matrix-vector multiplication

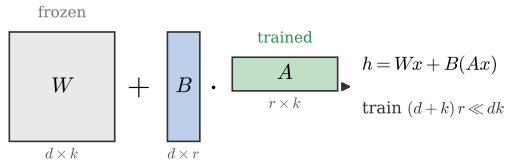
- ▶ need to compute matrix-vector product $y = Ax$, $A \in \mathbb{R}^{m \times n}$
- ▶ A has known factorization $A = BC$, $B \in \mathbb{R}^{m \times r}$
- ▶ computing $y = Ax$ directly: mn operations
- ▶ computing $y = Ax$ as $y = B(Cx)$ (compute $z = Cx$ first, then $y = Bz$): $rn + mr = (m + n)r$ operations
- ▶ savings can be considerable if $r \ll \min\{m, n\}$

Application: low-rank adaptation (LoRA)

a pretrained weight matrix $W \in \mathbb{R}^{d \times k}$ is huge, and fine-tuning all of it is expensive

LoRA freezes W and learns a *low-rank* update $\Delta W = BA$, with $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times k}$, $r \ll d, k$

- ▶ forward pass $h = Wx + B(Ax)$ (the same fast-product trick)
- ▶ trains $(d + k)r$ parameters instead of dk , often well under 1%
- ▶ many small adapters can share one frozen base model



Low-rank approximation: compression

real matrices (images, data tables) are often close to a *low-rank* matrix: a rank- r approximation needs only $r(m + n)$ numbers instead of mn

rank 5



2% of the data

rank 20



7% of the data

rank 50



18% of the data

full (rank 512)



100%

rank measures *redundancy*: a few independent patterns already capture most of the data (the best rank- r approximation comes from the SVD, later)

Full rank matrices

for $A \in \mathbb{R}^{m \times n}$ we always have $\text{rank}(A) \leq \min(m, n)$

we say A is *full rank* if $\text{rank}(A) = \min(m, n)$

- ▶ for **square** matrices, full rank means nonsingular
- ▶ for **skinny** matrices ($m \geq n$), full rank means columns are independent
- ▶ for **fat** matrices ($m \leq n$), full rank means rows are independent