



# High-dimensional data & scalable indexing

---

Aug 2013

@Yiming Yang, Scalable Indexing

1



## Outline

---

- Big data and high-dimensional sparse vector spaces
- Inverted indexing for scalable computation
- Review of related matrix algebra & calculus

Aug 2013

@Yiming Yang, Scalable Indexing

2



## Typical Text Mining Tasks

- Clustering
- Collaborative Filtering (CF)
- Text Categorization (TC)
- Link Analysis (HITS, PageRank)
- Information Retrieval (search)
- Dimensionality reduction (PCA, SVD, MF as sub-tasks)

Aug 2013

@Yiming Yang, Scalable Indexing

3



## An example of a document

### Full Review

I have been looking and looking for a new camera to replace our bulky, but simple and reliable (but only fair picture taker) Sony Mavica FD73. My other choice (Besides the more expensive Nikon Coolpix 3100) was the (also more expensive) Sony Cybershot P72. I recommend any of these cameras, and I was set to buy the Sony, but at the last minute I cheaped out and bought the 2100. No regrets. I bought the camera (along with 128mb memory card (the stock 16mb card will be kept in the bag as a spare) and carrying case) at the new Best Buy in Harrisburg, PA. I also bought a set of 4 Nickle-Metal Hydride rechargable batteries and charger at Walmart for less than \$20. I keep 2 in the camera and two in the charger/in the camera bag along with the original Lithium battery pack as spares.


- This format isn't useful for many algorithms
- We need to transform it into *a bag of words (or features)*

Aug 2013

@Yiming Yang, Scalable Indexing

4

## Free text → a bag of words



Useful?

Term	Freq	Term	Freq	Term	Freq
the	78	up	8	pictures	6
to	35	for	7	red	6
i	31	have	7	digital	5
and	29	image	7	eye	5
a	19	like	7	not	5
camera	17	mode	7	on	5
is	17	much	7	or	5
in	12	software	7	shutter	5
with	11	very	7	sony	5
be	9	can	6	than	5
but	9	images	6	that	5
it	9	movies	6	after	4
of	9	my	6	also	4
this	9	no	6	:	:

Aug 2013

@Yiming Yang, Scalable Indexing

5



## Typical Preprocessing

- Remove stop words (500+ in English)
- Convert words to stems
  - E.g., images → image
- Canonicalize abbreviations
  - {U. S., US, U. S. A., usa, us, ...} → USA
- Remove non-word symbols
  - 0-9, "-", "/", etc.
- Take the union of the terms in all doc's as the global vocabulary

Aug 2013

@Yiming Yang, Scalable Indexing

6



## TF-IDF Term Weighting

- TF (Term Frequency - local statistic)

$TF(t, d)$  is the count of term  $t$  in document  $d$

- IDF (Inverse of Document Frequency - global statistic)

$$IDF(t, D) = \log [N / (n(t, D) + 1)]$$

where  $D$  is a collection of  $N$  doc's and  $n(t, D)$  is the count of  $t$  in  $D$ .

- TF-IDF (a popular scheme)

$$TF\text{-}IDF(t | d, D) = TF(t, d) * IDF(t, D)$$

Aug 2013

@Yiming Yang, Scalable Indexing

7



## Viewing Data as a Matrix

	Term					
Document	$x_{11}$	$x_{12}$	$\dots$	$\dots$	$\dots$	$x_{1m}$
	$x_{21}$	$x_{22}$	$\dots$	$\dots$	$\dots$	$x_{2m}$
	$\dots$	$\dots$	$\dots$	$x_{ij}$	$\dots$	$\dots$
	$x_{n1}$	$x_{n2}$	$\dots$	$\dots$	$\dots$	$x_{nm}$

- Each row is a document (a bag of terms) in collection  $D$
- Each column is a unique term (a bag of documents) in  $D$
- Each cell is the within-document term weight (e.g.,  $TF * IDF$ )

Aug 2013

@Yiming Yang, Scalable Indexing

8



## Matrix multiplication for retrieval

$$\begin{array}{c} \text{Matrix } X \\ \text{(n doc's, m words)} \end{array} \quad \begin{array}{c} \text{Query } q \end{array} \quad \begin{array}{c} \text{Similarity} \\ \text{Scores} \end{array}$$
$$\begin{bmatrix} x_{11} & x_{12} & \cdots & \cdots & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & \cdots & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & x_{ij} & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & \cdots & \cdots & x_{nm} \end{bmatrix} \quad \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ \vdots \\ \vdots \\ q_m \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- Computing  $y := Xq$  yields the similarity scores of doc's w.r.t. the query.
- If both the doc's and the query are normalized, it yields cosine similarities.

Aug 2013

@Yiming Yang, Scalable Indexing

9



## High-dimensional Sparse Vectors

- # of unique words in English
    - 470k entries in Webster (1993)
    - 1M+ if including all abbreviations, misspellings, etc.
  - # of documents in benchmark datasets in IR
    - 870M doc's in WebClue12 (TREC), for example
  - # of unique words per document on average
    - Tens or hundreds per news article, for example
- 99.9%+ of the entries in the vectors are zero

Aug 2013

@Yiming Yang, Scalable Indexing

10



## Computing Cosine Similarity

$$x_i = (x_{i1}, \dots, x_{im}), \quad \|x_i\| = \sqrt{\sum_{j=1}^m x_{ij}^2}$$

$$q = (q_1, \dots, q_m), \quad \|q\| = \sqrt{\sum_{j=1}^m q_j^2}$$

$$\cos(x_i, q) = \frac{x_{i1}q_1 + \dots + x_{im}q_m}{\|x_i\|\|q\|} = \frac{x_i \cdot q}{\|x_i\|\|q\|} = \frac{x_i}{\|x_i\|} \cdot \frac{q}{\|q\|}$$

- **O(mn) time** and **O(mn) space** if using a dense matrix/vector to compute  $y := Xq$  -- wasting most time and space on zero entries!
- **Inverted indexing is a better alternative.**



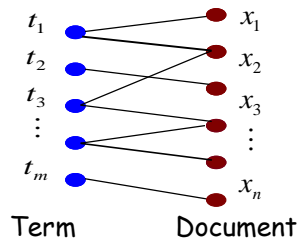
## Outline

- ✓ Big data and high-dimensional sparse matrices
- Inverted indexing for scalable computation
- Review of related matrix algebra & calculus



## Inverted Indexing

- Index documents for each unique term as  
termID: ((DID, weight), ..., (DID, weight))
- Equivalent to build a bipartite graph with sparse links



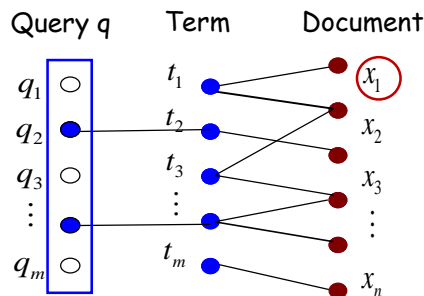
Aug 2013

@Yiming Yang, Scalable Indexing

13



## Compute the cosine similarity



$$\begin{aligned}\cos(x_i, q) &= \frac{x_{i1}q_1 + \dots + x_{im}q_m}{\|x_i\| \|q\|} \\ &= \frac{x_i \cdot q}{\|x_i\| \|q\|} = \frac{x_i}{\|x_i\|} \cdot \frac{q}{\|q\|}\end{aligned}$$

- Update the score of a doc only if it contains some query words
- For ranking doc's given  $q$ , we do not need  $\|q\|$

Aug 2013

@Yiming Yang, Scalable Indexing

14



## Time/Space Saving

- $n$  doc's,  $m$  words in the vocabulary
- $K$  unique words per doc on average
- $K_q$  unique words per query on average

Space Saving?

*DnsMtrix Space* :  $O(mn)$

*InvIndx Space* :  $O(kn)$

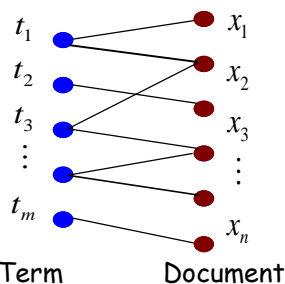
$$\frac{\text{Space}(\text{dnsMtrix})}{\text{Space}(\text{invIndx})} = \frac{m}{k}$$

Time Saving in Retrieval?

*DnsMtrix Time* :  $O(mn)$

*InvIndx Time* :  $O\left(\frac{k_q kn}{m}\right)$

$$\frac{\text{Time}(\text{dnsMtrix})}{\text{Time}(\text{invIndx})} = \frac{m^2}{k_q k}$$



Aug 2013

@Yiming Yang, Scalable Indexing

15



## Time/Space Saving Factors

$k_q$	$k$	$m$	space saving ( $m/k$ )	time saving ( $m^2/k \cdot k_q$ )
3	200	1,000	5	1,667
3	200	100,000	500	16,666,667
3	200	1,000,000	5,000	1,666,666,667

Aug 2013

@Yiming Yang, Scalable Indexing

16





## When shall we use inverted indexing?

- Use it for computing  $Xv$ 
  - If data matrix ( $X$ ) is relatively stable, large and highly sparse
- Not suitable
  - If new documents arrive frequently (time series) and if updating the inverted index of matrix  $X$  for each new doc is too costly.
  - E.g., in filtering of news stories with fixed queries - re-index the entire document collection constantly (for every new doc) would be too costly

Aug 2013

@Yiming Yang, Scalable Indexing

17



## High-dimensional Sparse Data in Collaborative Filtering (CF)

$$\begin{array}{c} \text{Users (n)} \\ \begin{bmatrix} x_{11} & x_{12} & \cdots & \cdots & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & \cdots & \cdots & x_{2m} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & x_{n2} & \cdots & \cdots & \cdots & x_{nm} \end{bmatrix} \end{array} \quad \begin{array}{c} \text{Items (m)} \\ \text{Query } q \\ \begin{bmatrix} q_1 \\ q_1 \\ \vdots \\ \vdots \\ \vdots \\ q_m \end{bmatrix} \end{array} = \begin{array}{c} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \\ \text{Similarity scores} \\ \text{of the kNN users} \end{array}$$

Rating (1~5) by user  $i$  on item  $j$

Given a new user ("query"), we need to find the top- $k$  similar users (the **k-nearest neighbors**) in the matrix for predicting the "taste" of the new user.

Aug 2013

@Yiming Yang, Scalable Indexing

18

## High-dimensional Sparse Data in Link Analysis (HITS or PageRank)

Adjacency Matrix A (in HITS)

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
$y_1$	0	1	1	0	0
$y_2$	1	0	1	0	1
$y_3$	0	0	0	0	1
$y_4$	0	0	1	0	0
$y_5$	0	0	0	1	0

Probabilistic Transition Matrix T  
(in PageRank)

	$v_1$	$v_2$	$v_3$	$v_4$	$v_5$
$v_1$	0	0.5	0.5	0	0
$v_2$	0.33	0	0.33	0	0.33
$v_3$	0	0	0	0	1
$v_4$	0	0	1	0	0
$v_5$	0	0	0	1	0

$A[i,j] = 1$  iff there is a link from  $i$  to  $j$ .

$T[i,j]$ 's sum to 1 over  $j$ 's.

**Power iteration** is to compute  $x := A^T A x$  repetitively.

Aug 2013

@Yiming Yang, Scalable Indexing

19

## Analytic Tasks of Interest

- Clustering
- ✓ Collaborative Filtering (CF)
- Text Categorization (TC)
- ✓ Link Analysis (HITS, PageRank)
- ✓ Information Retrieval (IR)
- ✓ Dimensionality reduction (PCA, SVD, MF)

Aug 2013

@Yiming Yang, Scalable Indexing

20

## Outline

- ✓ Big data and high-dimensional sparse matrices
- ✓ Inverted indexing for scalable computation
- Review of related matrix algebra & calculus

Aug 2013

@Yiming Yang, Scalable Indexing

21

## Vector Norm Definitions

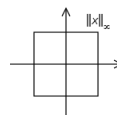
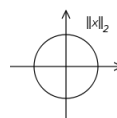
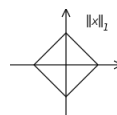
E.g.,  $x = (x_1, x_2)$

$$\|x\|_1 \equiv |x_1| + |x_2|$$

$$\|x\|_2 \equiv \sqrt{x_1^2 + x_2^2}$$

$$\|x\|_\infty \equiv \max(|x_1|, |x_2|)$$

$$\|x\|_0 \equiv \text{card}\{x_i \neq 0\}$$



$\|x\|_0 = 1$   
(Wikipedia)

@Yiming Yang, Scalable Indexing

Aug 2013

22



## Common Operations (cont'd)

$$Ax = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n \end{pmatrix} = \sum_{j=1}^n x_j \begin{pmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{nj} \end{pmatrix}$$

$$xA = (x_1 \quad x_2 \quad \cdots \quad x_n) \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} = \sum_{i=1}^n x_i (a_{i1} \quad a_{i2} \quad \cdots \quad a_{im})$$

@Yiming Yang, Scalable Indexing

Aug 2013

23



## Common Operations (cont'd)

$$X_{N \times M} \Lambda_{M \times M} = \left( \begin{bmatrix} \times \\ \vdots \\ \times \end{bmatrix}_1, \dots, \begin{bmatrix} \times \\ \vdots \\ \times \end{bmatrix}_M \right) \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_M \end{pmatrix} = \left( \lambda_1 \begin{bmatrix} \times \\ \vdots \\ \times \end{bmatrix}_1, \dots, \lambda_M \begin{bmatrix} \times \\ \vdots \\ \times \end{bmatrix}_M \right)$$

$$\Lambda_{N \times N} X_{N \times M} = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{pmatrix} \begin{pmatrix} \begin{bmatrix} \times & \cdots & \times \end{bmatrix}_1 \\ \vdots \\ \begin{bmatrix} \times & \cdots & \times \end{bmatrix}_N \end{pmatrix} = \begin{pmatrix} \lambda_1 \begin{bmatrix} \times & \cdots & \times \end{bmatrix}_1 \\ \vdots \\ \lambda_N \begin{bmatrix} \times & \cdots & \times \end{bmatrix}_N \end{pmatrix}$$

@Yiming Yang, Scalable Indexing

Aug 2013

24



## Common Operations (cont'd)

$$A_{M \times N} B_{N \times K} = \sum_{j=1}^N \underbrace{\begin{bmatrix} a_{1j} \\ \vdots \\ a_{Mj} \end{bmatrix}}_{A_j} \underbrace{\begin{bmatrix} b_{j1} & \cdots & b_{jK} \end{bmatrix}}_{B_j} = \sum_{j=1}^N \begin{pmatrix} a_{1j}b_{j1} & a_{1j}b_{j2} & \cdots & a_{1j}b_{jK} \\ a_{2j}b_{j1} & a_{2j}b_{j2} & \cdots & a_{2j}b_{jK} \\ \cdots & \cdots & \cdots & \cdots \\ a_{Mj}b_{j1} & a_{Mj}b_{j2} & \cdots & a_{Mj}b_{jK} \end{pmatrix}$$

$$\boxed{\phantom{A}} \boxed{\phantom{B}} = \boxed{\phantom{AB}} \quad \text{A rank-1 matrix}$$



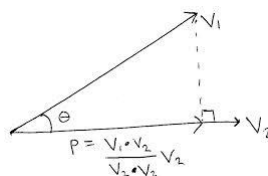
## Common Operations

Given  $x = (x_1, x_2, \dots, x_m)$  and  $y = (y_1, y_2, \dots, y_m)$ .

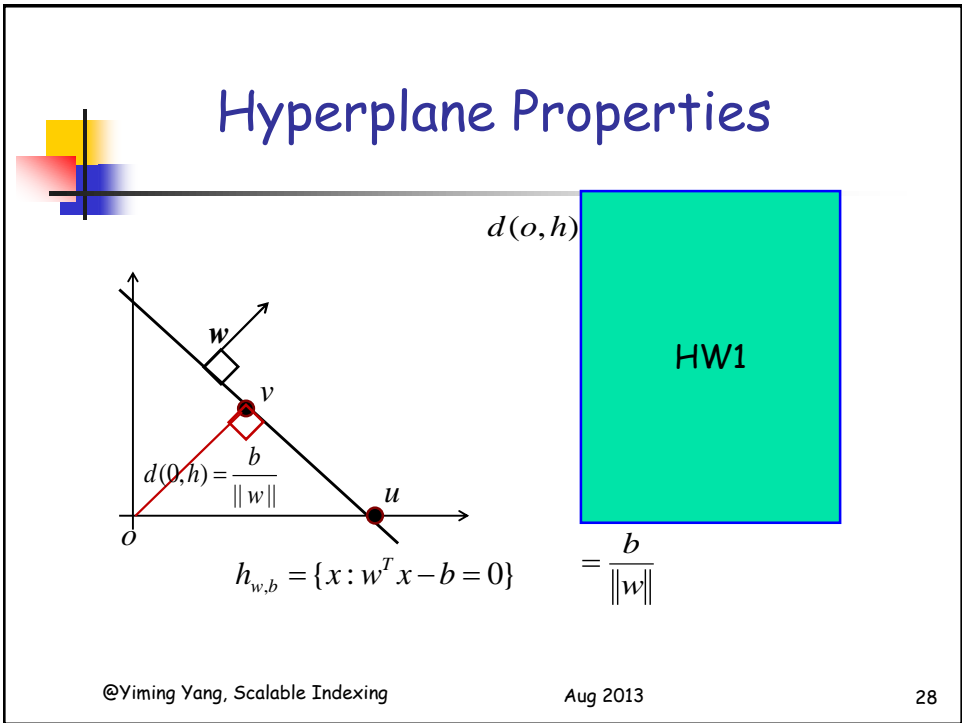
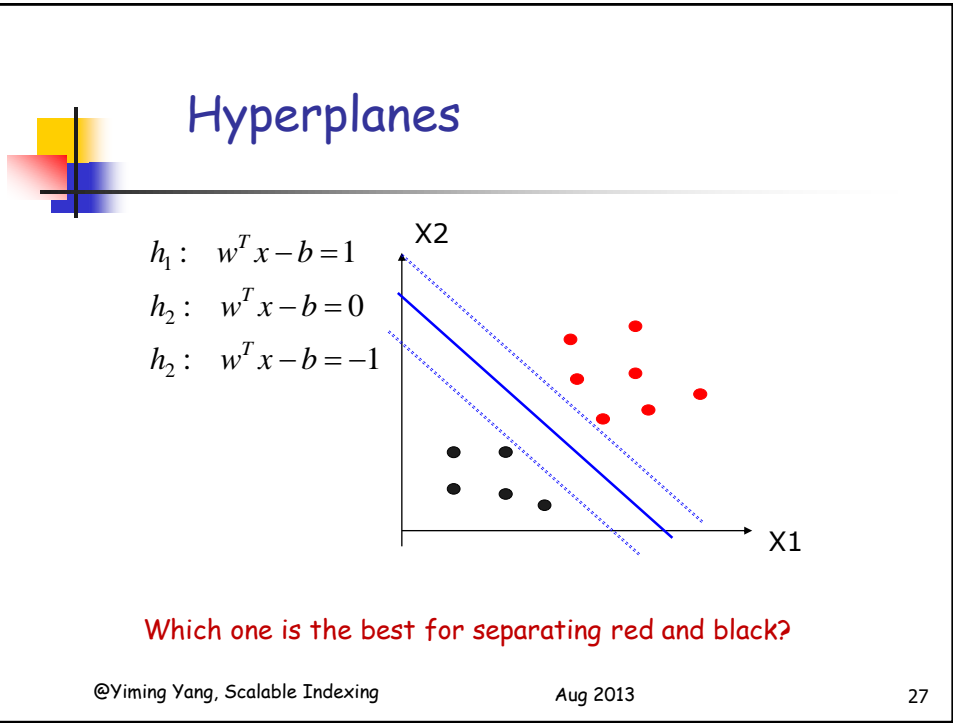
$$x \cdot y = x_1 y_1 + x_2 y_2 + \cdots + x_m y_m = \sum_{i=1}^m x_i y_i$$


$$\cos(x, y) = \frac{x \cdot y}{\|x\| \times \|y\|}$$

$$\|x\| \cos(\theta) = \|x\| \frac{x \cdot y}{\|x\| \times \|y\|} = x \cdot \frac{y}{\|y\|}$$



Vector projection

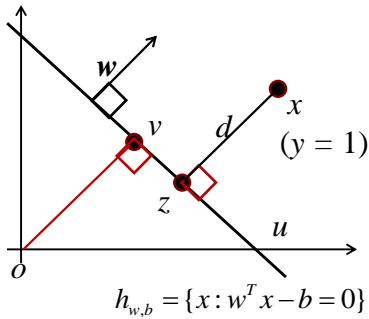




## Proof of

$$d \equiv d(x, h) = \frac{y(w^T x - b)}{\|w\|}$$

---




$h_{w,b} = \{x : w^T x - b = 0\}$

Case 1:  $y = 1, x = z + d \frac{w}{\|w\|}$

HW1

Case 2:  $y = -1, x = z - d \frac{w}{\|w\|}, \dots$

@Yiming Yang, Scalable Indexing
Aug 2013
29



## Matrix Calculus

---

- Useful in optimizing prediction models
- Different types of derivatives
  - Scalar-by-scalar
  - Scalar-by-vector
  - Vector-by-vector
  - ...

@Yiming Yang, Scalable Indexing
Aug 2013
30



## Matrix Derivatives

- **Scalar-by-scalar**

$$x \in R, \quad f(x) \in R$$

$$\text{Ex 1. } f(x) = ax^2 + b, \quad \frac{\partial f}{\partial x} = 2ax, \quad \frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}(2ax) = 2a$$

$$\text{Ex 2. } y = \underbrace{x^2}_u \underbrace{\log x}_v, \quad \frac{dy}{dx} = u \frac{dv}{dx} + \frac{du}{dx} v = x^2 \cdot \frac{1}{x} + 2x \log x$$



## Matrix Derivatives (cont'd)

- **Scalar-by-vector**


$$\mathbf{x} = (x_1, x_2, \dots, x_d)^T \in R^d, \quad f(x) \in R$$

- The gradient  $\nabla f \equiv \frac{\partial f}{\partial \mathbf{x}} = \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_d} \right] \in R^d$

- The Hessian

$$H \equiv \nabla \nabla f \equiv \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_d \partial x_1} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_d \partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_d} & \frac{\partial^2 f}{\partial x_2 \partial x_d} & \dots & \frac{\partial^2 f}{\partial x_d \partial x_d} \end{bmatrix} \in R^{d \times d}$$





## Example 1.

- **Scalar-by-vector**

*e.g.*,  $\mathbf{x} = (x_1, x_2)^T$ ,  $f(x) = ax_1^2 + bx_1x_2$

- The gradient  $\nabla f \equiv \left[ \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \right] = [2ax_1 + bx_2, bx_1]$

- The Hessian  $H \equiv \nabla \nabla f = \begin{bmatrix} 2a & b \\ b & 0 \end{bmatrix}$



## Matrix Derivatives (cont'd)

- **Vector-by-vector**

*e.g.*,  $\mathbf{x} \in R^d$ ,  $A \in R^{n \times d}$ ,  $f(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

- The gradient

$$\nabla f \equiv \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \dots & \frac{\partial y_1}{\partial x_d} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \dots & \frac{\partial y_2}{\partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_d}{\partial x_1} & \frac{\partial y_d}{\partial x_2} & \dots & \frac{\partial y_d}{\partial x_d} \end{bmatrix} \in R^{n \times d}, \quad \nabla f = A$$



## Example 2

$$\mathbf{x} \in R^d, \quad A \in R^{d \times d}, \quad f = \mathbf{x}^T A \mathbf{x} \quad \leftarrow \text{Scalar}$$

$$\nabla f = ? \quad \leftarrow \text{Vector (1-by-d)}$$

$$\text{Let } \mathbf{u} = \mathbf{x}, \quad \mathbf{v} = A\mathbf{x}, \quad f = \mathbf{u}^T \mathbf{v}.$$

$$\frac{\partial \mathbf{u}}{\partial \mathbf{x}} = I_{d \times d}, \quad \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = A, \quad \mathbf{u}^T \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \mathbf{x}^T A, \quad \mathbf{v}^T \frac{\partial \mathbf{u}}{\partial \mathbf{x}} = \underbrace{\mathbf{x}^T A^T}_{\mathbf{v}^T} I$$

$$\mathbf{u}^T \frac{\partial \mathbf{v}}{\partial \mathbf{x}} + \mathbf{v}^T \frac{\partial \mathbf{u}}{\partial \mathbf{x}} = \mathbf{x}^T A + \mathbf{x}^T A^T = \mathbf{x}^T (A + A^T) \quad \leftarrow \text{Vector (1-by-d)}$$