LECTURE 6

Win+w

Signal Processing on Databases

Jeremy Kepner

Lecture 2: Group Theory

Spreadsheets, Big Tables, and the Algebra of Associative Arrays

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

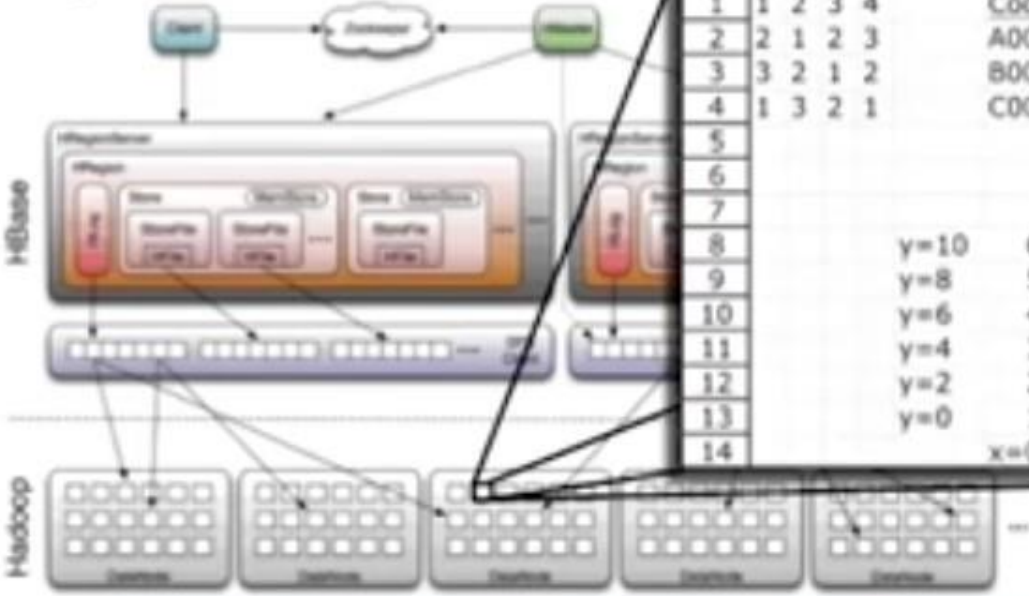
This work is sponsored by the Department of the Air Force under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

Outline

- Introduction
  - What are Spreadsheets?
  - Theoretical Goals
  - Associative Arrays
- Definitions
- Group Theory
- Vector Space
- Linear Algebra
- Summary

What are Spreadsheets and Big Tables?

Big Tables



	A	B	C	D	E	F	G	H	I	J	K
1	1	2	3	4		Code	Name	Job		Date	$ in bank
2	2	1	2	3		A0001	Alice	scientist		2000 Jan 01	$11,700
3	3	2	1	2		B0002	Bob	engineer		2001 Jan 01	$10,600
4	1	3	2	1		C0003	Charlie	mathamatician		2002 Jan 01	$10,200
5										2003 Jan 01	$8,600
6										2004 Jan 01	$10,400
7										2005 Jan 01	$10,600
8					y=10	6	12	18		2006 Jan 01	$10,900
9					y=8	5	10	15		2007 Jan 01	$12,300
10					y=6	4	8	12		2008 Jan 01	$12,600
11					y=4	3	6	9		2009 Jan 01	$9,000
12					y=2	2	4	6		2010 Jan 01	$10,600
13					y=0	1	2	3		2011 Jan 01	$11,700
14						x=0	x=5	x=10			

Spreadsheets

- *Spreadsheets* are the most commonly used analytical structure on Earth (100M users/day?)
- Big Tables (Google, Amazon, Facebook, ...) store most of the analyzed data in the world (Exabytes?)
- Simultaneous diverse data: strings, dates, integers, reals, ...
- Simultaneous diverse uses: matrices, functions, hash tables, databases, ...
- No formal mathematical basis; Zero papers in AMA or SIAM

Goal: Signal Processing on Graphs/Strings/Spreadsheets/Tables/ …

- Create a formal basis for working with these data structures based on an Algebra of Associative Arrays
- Better Algorithms
  - Can create algorithms by applying standard mathematical tools (linear algebra and detection theory)
- Faster Implementation
  - Associative array software libraries allow these algorithms to be implemented with ~50x less effort
- Good for managers, too
  - Much simpler than Microsoft Excel; formally correct

Multi-Dimensional Associative Arrays

- **Extends associative arrays to 2D and mixed data types**

```
A('alice ','bob ') = 'cited '
```

or

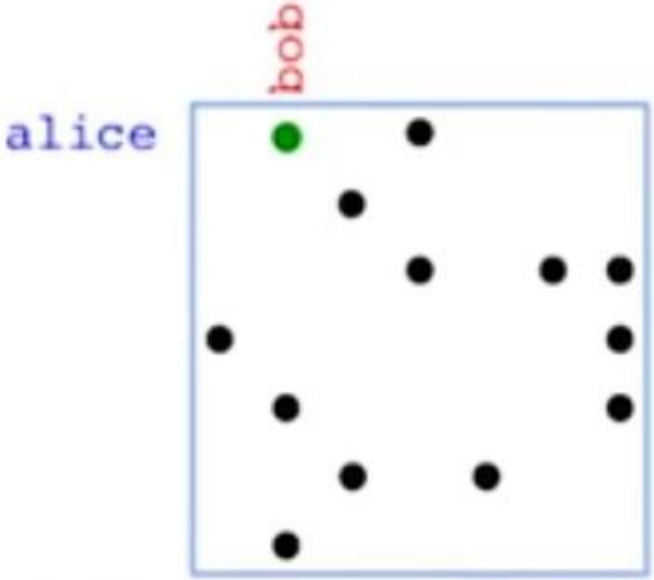
```
A('alice ','bob ') = 47.0
```

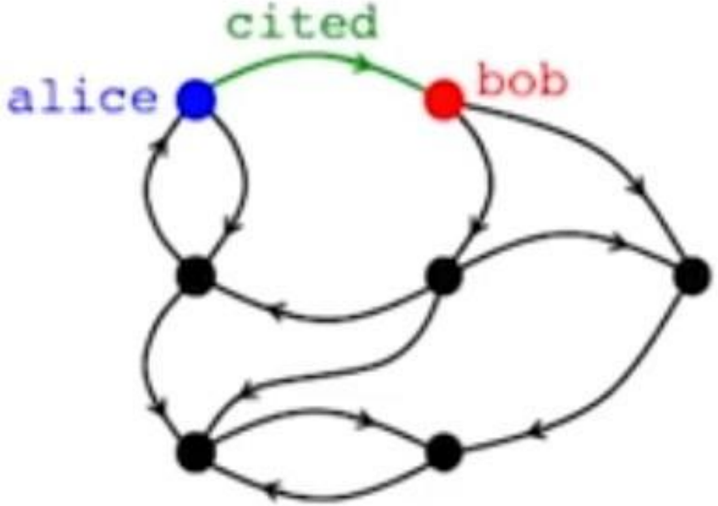
- **Key innovation: 2D is 1-to-1 with triple store**

```
('alice ','bob ','cited ')
```

or

```
('alice ','bob ',47.0)
```





Composable Associative Arrays

- **Key innovation: mathematical closure**
  - All associative array operations return associative arrays
- **Enables composable mathematical operations**

```
A + B      A - B      A & B      A|B      A*B
```

- **Enables composable query operations via array indexing**

```
A('alice bob ',:)      A('alice ',:)      A('al* ',:)
A('alice : bob ',:)    A(1:2,:)           A == 47.0
```

- **Simple to implement in a library (~2000 lines) in programming environments with: 1st class support of 2D arrays, operator overloading, sparse linear algebra**

> - Complex queries with ~50x less effort than Java/SQL
> - Naturally leads to high performance parallel implementation

Universal "Exploded" Schema

Triple Store Table: Ttranspose

	2001-01-01	2001-01-02	2001-01-03
src_ip/a	1		
src_ip/b		1	
domain/b		1	
domain/c			1
dest_ip/a	1		
dest_ip/c			1

Input Data

Time	src_ip	domain	dest_ip
2001-01-01	a		a
2001-01-02	b	b	
2001-01-03		c	c

	src_ip/a	src_ip/b	domain/b	domain/c	dest_ip/a	dest_ip/c
2001-01-01	1				1	
2001-01-02		1	1			
2001-01-03				1		1

Triple Store Table: T

Key Innovations

- Handles all data into a *single* table representation
- Transpose pairs allows quick look up of *either* row or column

Outline

- Introduction
- Definitions
  - Values
  - Keys
  - Functions
  - Matrix multiply
- Group Theory
- Vector Space
- Linear Algebra
- Summary

UNCLASSIFIED

Associative Array Definitions

- Keys and values are from the infinite strict totally ordered set $\mathbb{S}$

- Associative array $A(\mathbf{k}) : \mathbb{S}^d \rightarrow \mathbb{S}$, $\mathbf{k}=(k^1,\ldots,k^d)$, is a partial function from d keys (typically 2) to 1 value, where

  $A(\mathbf{k}_i) = v_i$ and $\emptyset$ otherwise

- Binary operations on associative arrays $A_3 = A_1 \oplus A_2$, where $\oplus = \cup_{f()}$ or $\cap_{f()}$, have the properties
  - If $A_1(\mathbf{k}_i) = v_1$ and $A_2(\mathbf{k}_i) = v_2$, then $A_3(\mathbf{k}_i)$ is

    $v_1 \cup_{f()} v_2 = f(v_1,v_2)$ or $v_1 \cap_{f()} v_2 = f(v_1,v_2)$

  - If $A_1(\mathbf{k}_i) = v$ or $\emptyset$ and $A_2(\mathbf{k}_i) = \emptyset$ or v, then $A_3(\mathbf{k}_i)$ is

    $v \cup_{f()} \emptyset = v$ or $v \cap_{f()} \emptyset = \emptyset$

- **High level usage dictated by these definitions**
- **Deeper algebraic properties set by the collision function** $f()$
- **Frequent switching between "algebras" (how spreadsheets are used)**

Associative Array Values

- Value requirements
  - Diverse types: integers, reals, strings, ...
  - Sortable
  - Set
- Let $\mathbb{S}$ be an infinite strict totally ordered set
  - Total order is an implementation (not theoretical) requirement
  - All values (and keys) will be drawn from this set

- Allowable operations for $v_1, v_2 \in \mathbb{S}$

  $v_1 < v_2$ $\quad v_1 = v_2 \quad$ $v_1 > v_2$

- Special symbols: $\varnothing$, $-\infty$, $+\infty$

  $v \leq +\infty$ is always true $(+\infty \in \mathbb{S})$

  $v \geq -\infty$ is always true $(-\infty \in \mathbb{S})$

  $\varnothing$ is the empty set $(\varnothing \subset \mathbb{S})$

- **Above properties are consistent with strict totally ordered sets**

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Collision Function $f()$

- Collision function $f(v_1,v_2)$ can have
  - two contexts ($\cup$ $\cap$)
  - three conditions ($<$ $=$ $>$)
  - $d + 5$ possible outcomes ($\mathbf{k}$ v_1 v_2 $\varnothing$ $-\infty$ $+\infty$) [or sets of these]
- Combinations result in an enormous number of functions ($\sim10^{30}$) and an even greater number of associative array algebras (function pairs)
  - Impressive level of functionality given minimal assumptions
- Focus on "nice" collision functions
  - Keys are not used inside the function; results are single valued
  - No tests on special symbols

$$
\begin{aligned}
&f(v_1,v_2)\\
&\quad v_1 < v_2 : v_1\ v_2\ \varnothing\ -\infty\ +\infty\\
&\quad v_1 = v_2 : v\quad\ \ \varnothing\ -\infty\ +\infty\\
&\quad v_1 > v_2 : v_1\ v_2\ \varnothing\ -\infty\ +\infty
\end{aligned}
$$

- **Above properties are consistent with strict totally ordered sets**
- **Note: $\varnothing$ is handled by $\cup$ $\cap$; not passed into $f()$**

What About Concatenation?

- Concatenation of values (or keys) can be represented by using $\cup$ or $\cap$ as collision function
  - Requires generalizing values to sets $v_1, v_2 \subset \mathbb{S}$
- Allowable operations for $v_1, v_2 \subset \mathbb{S}$

$$v_1 \cup v_2 \qquad\qquad v_1 \cap v_2$$

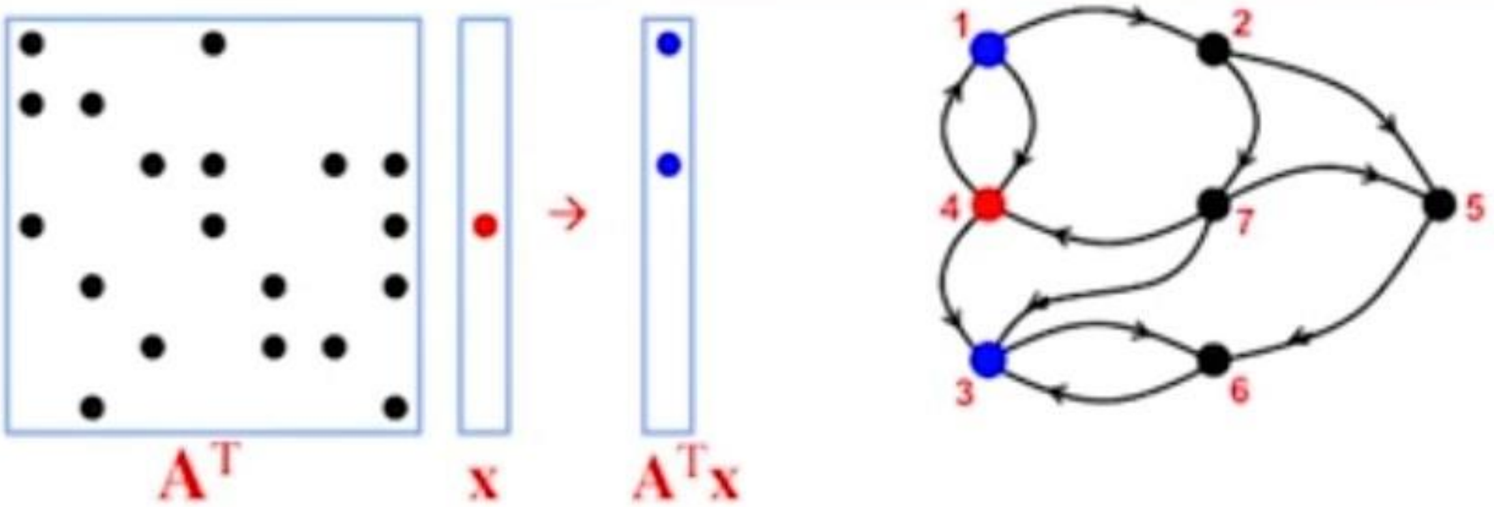
- Special symbols: $\varnothing$, $\mathbb{S}$

$v \cup \varnothing = \varnothing$	annihilator (but never reached, so identify)
$v \cup \mathbb{S} = \mathbb{S}$	annihilator
$v \cup \mathbb{S} = v$	identity
$v \cup \varnothing = v$	identity

- Possible operators: $\cup_{\cup}$, $\cap_{\cup}$, $\cup_{\cap}$, $\cap_{\cap}$

> - **Concatenating collision functions are very useful**
> - **Can be handled by extending values to be sets**

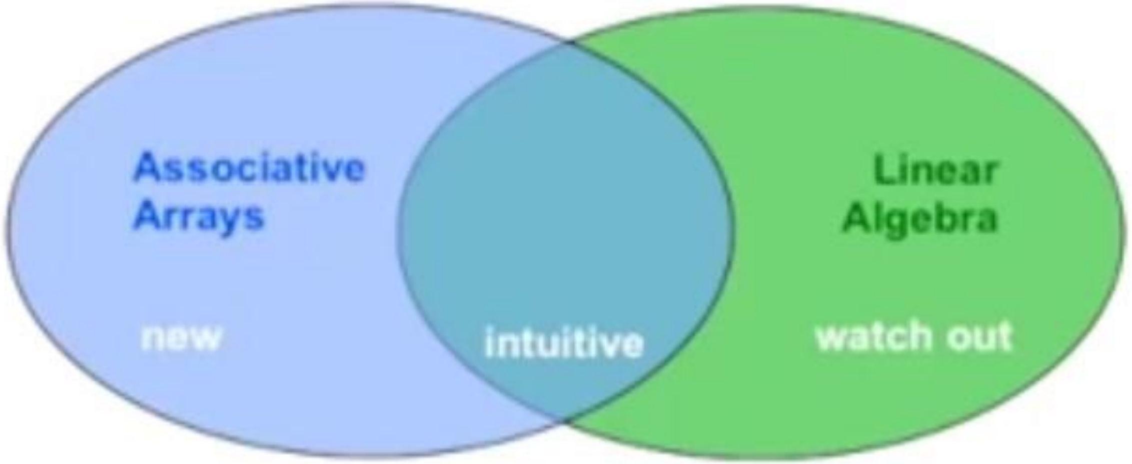
Matrix Multiply Framework



- Graphs can be represented as a sparse matrices
  - Multiply by adjacency matrix → step to neighbor vertices
  - Work-efficient implementation from sparse data structures
- Graph algorithms reduce to products on semi-rings: $A_3 = A_1 \oplus.\otimes A_2$
  - $\otimes$: associative, distributes over $\oplus$
  - $\oplus$: associative, commutative
  - Examples: +.* min.+ or.and

Theory Questions

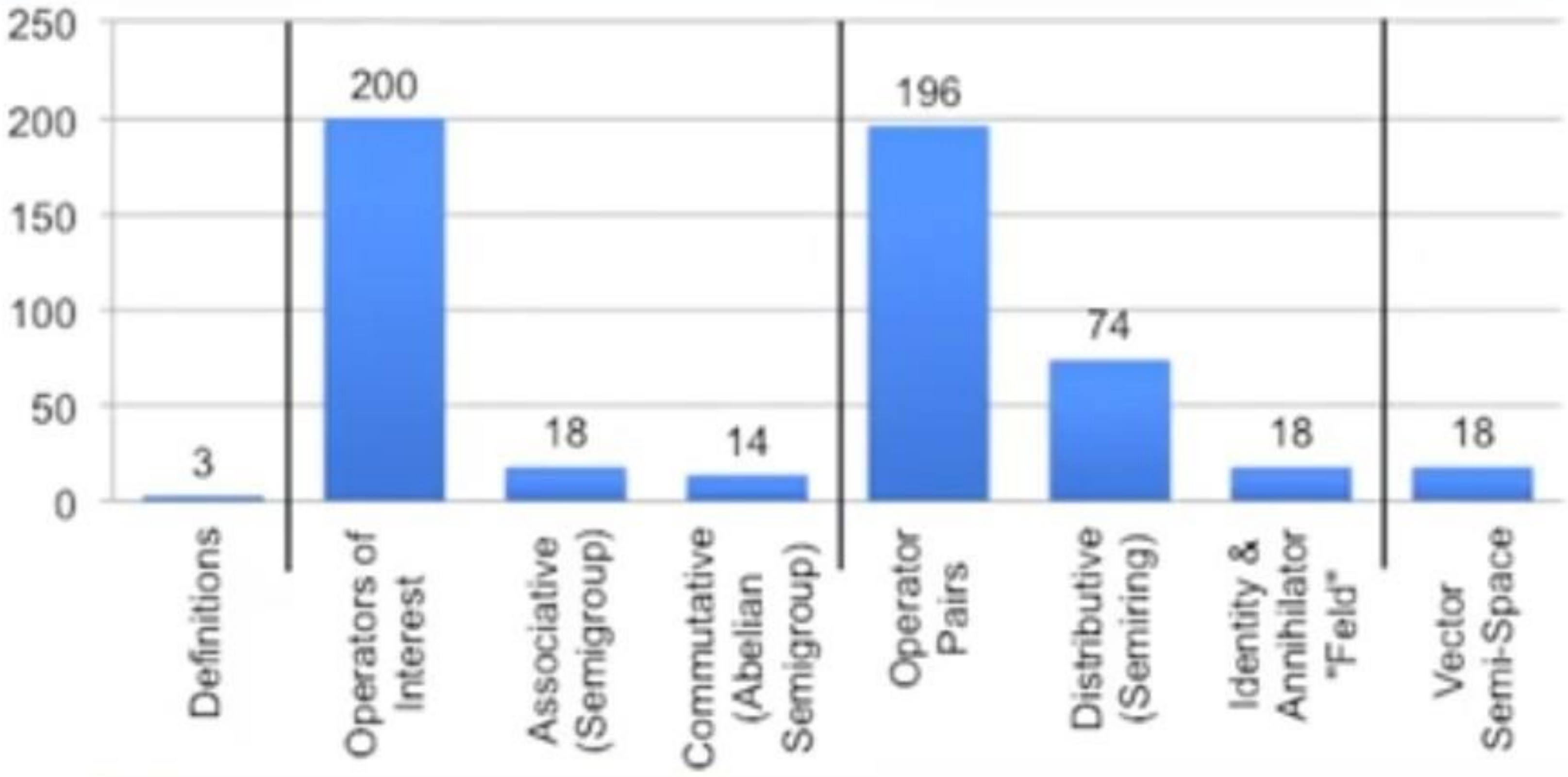
- Associative arrays can be constructed from a few definitions
- Similar to linear algebra, but applicable to a wider range of data
- Key questions
  - Which linear algebra properties do apply to associative arrays (intuitive)
  - Which linear algebra properties do not apply to associative arrays (watch out)
  - Which associative array properties do not apply to linear algebra (new)



Outline

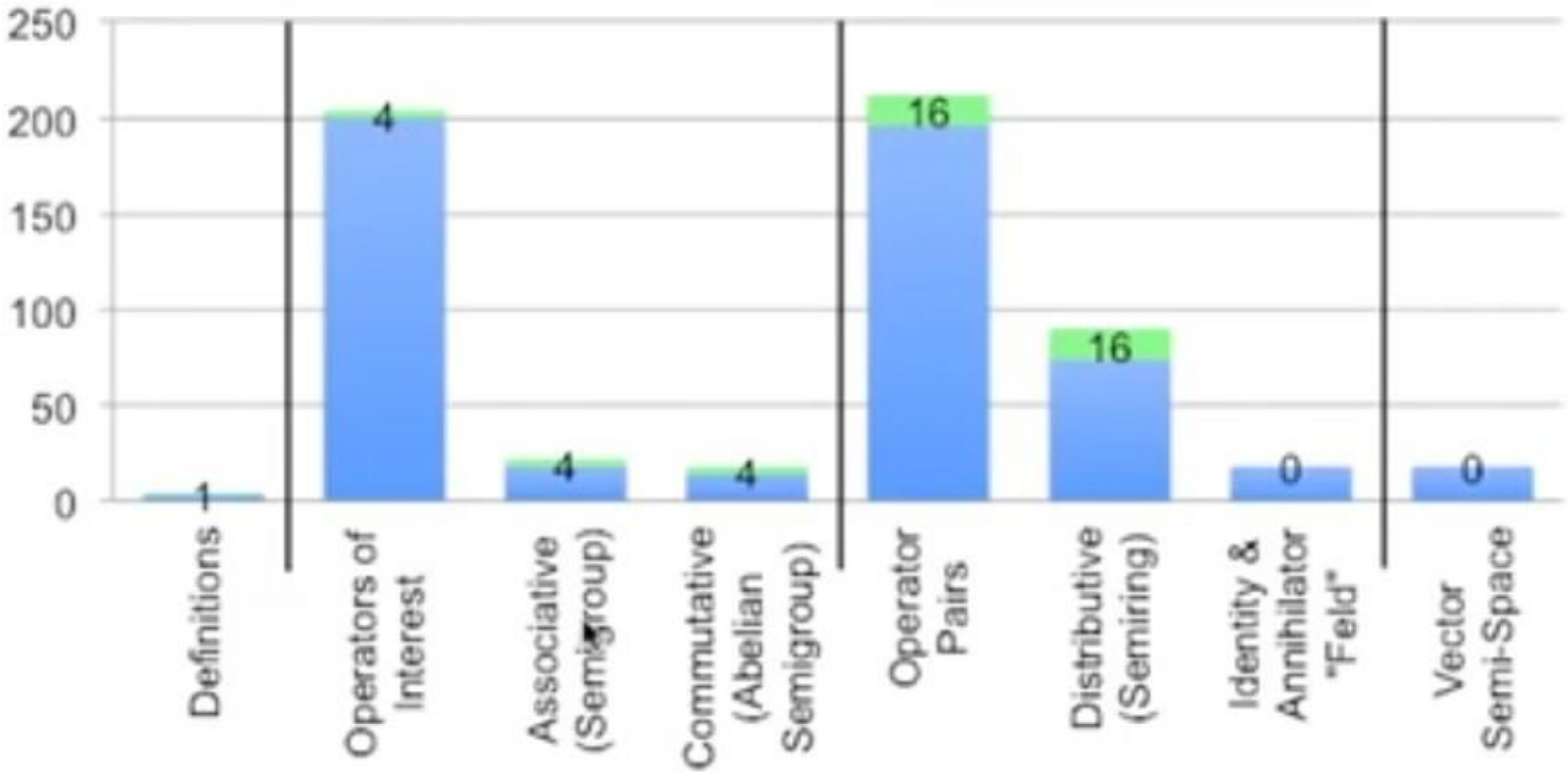
- Introduction
- Definitions
- Group Theory
  - Binary operators
  - Commutative monoids
  - Semirings
  - Feld
- Vector Space
- Linear Algebra
- Summary

Operators Roadmap



- **Begin with a few definitions**
- **Expand into many operators; reduce to well behaved**
- **Expand into many operator pairs; reduce to well behaved**

Including Concatenation



- **Including concatenation operators expands semirings**
- **Doesn't expand vector semi-space**

Associative and Commutative Operators

ID	Operator ⊕	$v_1 < v_2$	$v_1 = v_2$	$v_1 > v_2$
1	$\cup_{left}$	v_1	v	v_1
2	$\cap_{left}$	v_1	v	v_1
3	$\cup_{max}$	v_2	v	v_1
4	$\cap_{max}$	v_2	v	v_1
41	$\cup_{min}$	v_1	v	v_2
42	$\cap_{min}$	v_1	v	v_2
43	$\cup_{right}$	v_2	v	v_2
44	$\cap_{right}$	v_2	v	v_2
86	$\cap_{\delta}$	$\varnothing$	v	$\varnothing$
96	$\cap_{\varnothing}$	$\varnothing$	$\varnothing$	$\varnothing$
127	$\cup_{-\infty,\delta}$	$-\infty$	v	$-\infty$
128	$\cap_{-\infty,\delta}$	$-\infty$	v	$-\infty$
147	$\cup_{-\infty}$	$-\infty$	$-\infty$	$-\infty$
148	$\cap_{-\infty}$	$-\infty$	$-\infty$	$-\infty$
169	$\cup_{+\infty,\delta}$	$+\infty$	v	$+\infty$
170	$\cap_{+\infty,\delta}$	$+\infty$	v	$+\infty$
199	$\cup_{+\infty}$	$+\infty$	$+\infty$	$+\infty$
200	$\cap_{+\infty}$	$+\infty$	$+\infty$	$+\infty$

- Associative

$$(v_1 \oplus v_2) \oplus v_3 = v_1 \oplus (v_2 \oplus v_3)$$

- 18 associative operators
  - Semigroups
  - Groups w/o inverses

- Commutative

$$v_1 \oplus v_2 = v_2 \oplus v_1$$

- 14 associative & commutative operators
  - Removes left and right
  - Abelian Semigroups
  - Abelian Groups w/o inverses

Distributive Operator Pairs

- 14 x 14 = 196 Pairs of Abelian Semigroup operators
- Distributive

$$v_1 \otimes (v_2 \oplus v_3) = (v_1 \otimes v_2) \oplus (v_1 \otimes v_3)$$

- 74 distributive operator pairs
  - Semirings
  - Rings without inverses and without identity elements

- **1/3 of possible operator pairs are semirings**

Distributive Operator Pairs with Annihilators (0) and Identities (1)

- $\oplus$ identity: $v_1 \oplus 0 = v_1$ $0 = \varnothing, -\infty, +\infty$
- $\otimes$ identity: $v_1 \otimes 1 = v_1$ $1 = \varnothing, -\infty, +\infty$
- $\otimes$ annihilator: $v_1 \otimes 0 = 0$ $0 = \varnothing, -\infty, +\infty$

- 12 Semirings with appropriate 0 1 set (4 with two)
- 16 total over six operators: $\cup_{max}$, $\cap_{max}$, $\cup_{min}$, $\cap_{min}$, $\cup_{-\infty}$, $\cup_{+\infty}$
  - Felds? (Fields w/o inverses)

- $\oplus = \cup_{f()}$ in 10/16 ($\cup$ feels more like plus)
- $\otimes = \cap_{f()}$ in 10/16 ($\cap$ feels more like multiply)
- $\oplus = \cup_{f()}$ and $\otimes = \cap_{f()}$ in 8/16
- $0 = \varnothing$ in 6/8 ($\varnothing$ feels more like zero, $0 > 1$ might be a problem)

- **1/5 of semirings are Felds (Fields w/o inverses)**

Operator Pairs

$\otimes$ (columns), $\oplus$ (rows)

0 1	$\cup_{max}$	$\cap_{max}$	$\cup_{min}$	$\cap_{min}$	$\cap_{\delta}$	$\cap_{\emptyset}$	$\cup_{-\infty,\delta}$	$\cap_{-\infty,\delta}$	$\cup_{-\infty}$	$\cap_{-\infty}$	$\cup_{+\infty,\delta}$	$\cap_{+\infty,\delta}$	$\cup_{+\infty}$	$\cap_{+\infty}$
$\cup_{max}$	D	$\emptyset$ $-\infty$		$-\infty$ $+\infty$ $\emptyset$ $+\infty$		D				D			D	D
$\cap_{max}$		D	$-\infty$ $+\infty$ $-\infty$ $\emptyset$	$-\infty$ $+\infty$		D			$-\infty$ $\emptyset$	D				D
$\cup_{min}$	$+\infty$ $-\infty$ $+\infty$ $\emptyset$	$+\infty$ $-\infty$ $\emptyset$ $-\infty$	D	$\emptyset$ $+\infty$		D			D	D				D
$\cap_{min}$		$+\infty$ $-\infty$		D		D				D			$+\infty$ $\emptyset$	D
$\cap_{\delta}$					D	D								
$\cap_{\emptyset}$		D		D	D	D		D		D		D		D
$\cup_{-\infty,\delta}$						D	D	D	D	D				D
$\cap_{-\infty,\delta}$						D		D		D				D
$\cup_{-\infty}$				$\emptyset$ $+\infty$		D		D		D				
$\cap_{-\infty}$				D		D		D		D				
$\cup_{+\infty,\delta}$						D				D	D	D	D	D
$\cap_{+\infty,\delta}$						D				D		D		D
$\cup_{+\infty}$		$\emptyset$ $-\infty$				D						D		D
$\cap_{+\infty}$		D				D						D		D

D=distributes; 0=Plus Identity/Multiply Annihilator; 1=Multiply Identity

Concatenate Operators

ID	Operator ⊕	$f(v_1,v_2)$
201	$\cup_{\cup}$	$v_1 \cup v_2$
202	$\cap_{\cup}$	$v_1 \cup v_2$
203	$\cup_{\cap}$	$v_1 \cap v_2$
204	$\cap_{\cap}$	$v_1 \cap v_2$

- Recall v_1 and v_2 are sets
- All operators are associative and commutative
  - 4 Abelian Semigroups

$\otimes$

$\oplus$

0 1	$\cup_{\cup}$	$\cap_{\cup}$	$\cup_{\cap}$	$\cap_{\cap}$
$\cup_{\cup}$	D	$\emptyset$ $-\infty$	D	$-\infty$ $+\infty$ $\emptyset$ $+\infty$
$\cap_{\cup}$	D	D	$-\infty$ $+\infty$ $-\infty$ $\emptyset$	$-\infty$ $+\infty$
$\cup_{\cap}$	$+\infty$ $-\infty$ $+\infty$ $\emptyset$	$+\infty$ $-\infty$ $\emptyset$ $-\infty$	D	$\emptyset$ $+\infty$
$\cap_{\cap}$	D	$+\infty$ $-\infty$	D	D

- All operator pairs distribute
  - 16 Semirings

Outline

- Introduction
- Definitions
- Group Theory
- Vector Space
  - Vector Semispace
  - Uniqueness
- Linear Algebra
- Summary

Vector Space over a Feld

- Associative Array Vector ⊕
  - All associative arrays are conformant (unlike matrices)
- Associative Array Scalar ⊗
  - Scalar is a value applied directly to values; similar to constant function; or a function that takes on keys of non-scalar argument
- Vector Space ⊕ requirements
  - Commutes [Yes]; Associative [Yes]; 0 Identity element [Yes]
  - Inverse [No]
- Vector Space scalar ⊗ requirements
  - Commutes [Yes]; Associative [Yes]; Distributes over addition [Yes]; 1 Identity element [Yes]

- **All associative array operator pairs that yield Felds also result in Vector Spaces wo/inverses (Vector Semispace?)**

Vector Semispace Properties

- Scalar $\oplus$ identity annihilates under $\otimes$ [Yes]
- Subspace [Yes]
  - Any linear combination of vectors taken from the subspace is in the subspace and obeys the properties of a vector space
  - Theorem: Intersection of any subspaces is a subspace?
- Span [Yes+]
  - Given a set of vectors A_j, their span is all linear combinations of those vectors (includes vectors of different lengths)

$$\oplus_j (a_j \otimes A_j)$$

- Span = Subspace [Yes?]
  - Given an arbitrary set of vectors, their span is a vector space?
- Linear dependence [No]
  - There is a non-trivial linear combination of vectors equal to the $\oplus$ identity; can't do this without additive inverse
  - Need to redefine linear independence or all vectors are linearly independent; use minimum vectors in a subspace definition?
  - Likewise need to redefine basis as it depends upon linear dependence

- **Key question: under what conditions does the result of a linear combination of associative arrays uniquely determine the coefficients**

Unique Coefficient Conditions

- Consider a linear combinations of two associative array vectors

$$A_3 = (a_1 \otimes A_1) \oplus (a_2 \otimes A_2)$$

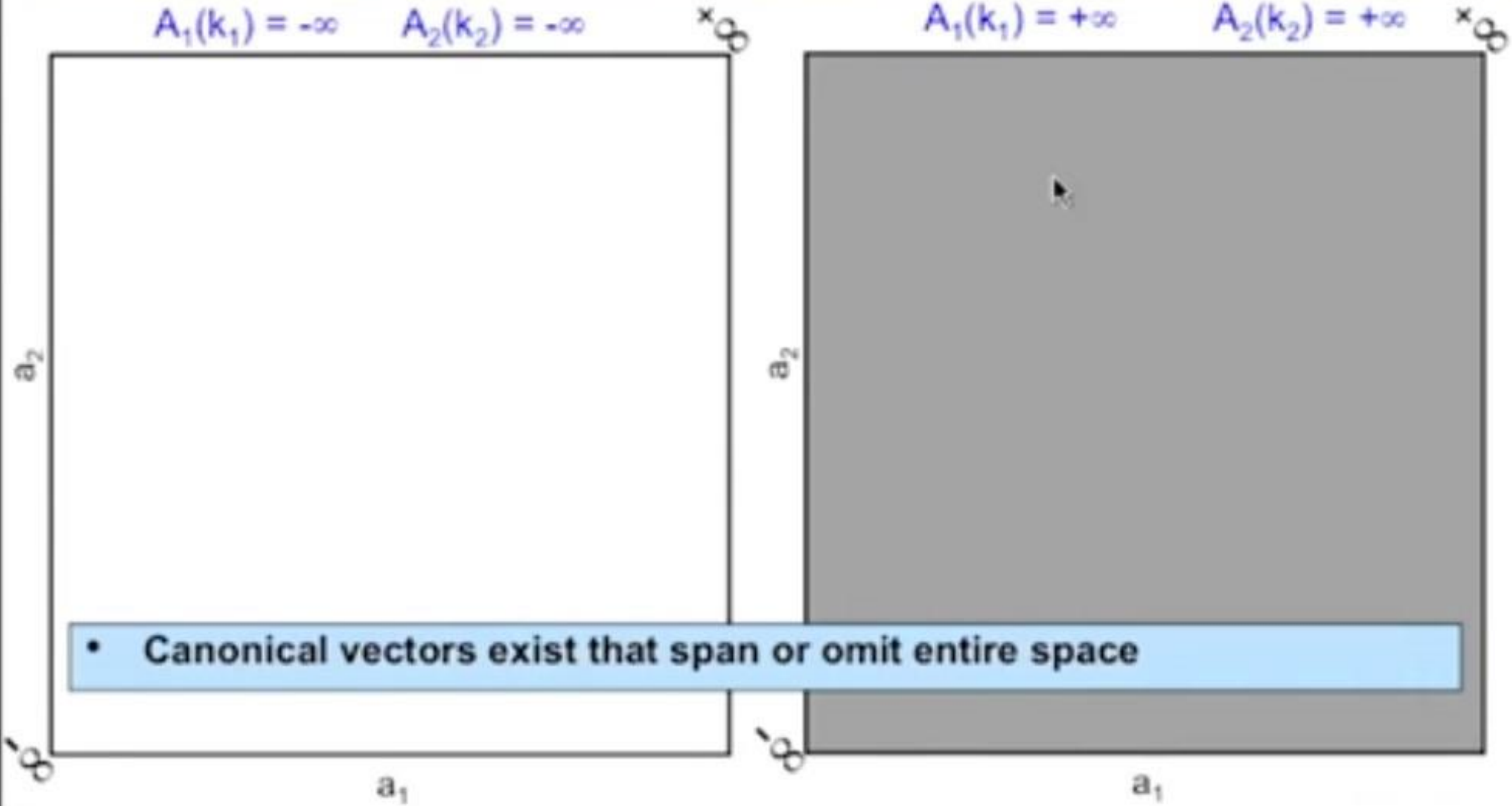
- Let $\oplus = \cup_{min}$, $\otimes = \cap_{max}$, $0 = \emptyset$, and $1 = -\infty$

- When are a_1 and a_2 uniquely determined by A_1, A_2 and A_3 ?

Canonical Vectors	Single valued	Multi-valued
$A_1(k_1) = -\infty$ $A_2(k_2) = -\infty$		$A_1(k_1\ k_2) = (v_1\ v_2)$ $A_2 = A_1$ $v_1 < v_2$
$A_1(k_1) = +\infty$ $A_2(k_2) = +\infty$	$A_1(k_1\ k_2) = (v\ v)$ $A_2 = A_1$	$A_1(k_1\ k_2) = (v_1\ v_2)$ $A_2(k_1\ k_2) = (v_2\ v_1)$ $v_1 < v_2$

- **Consider specific cases to show existence of uniqueness**

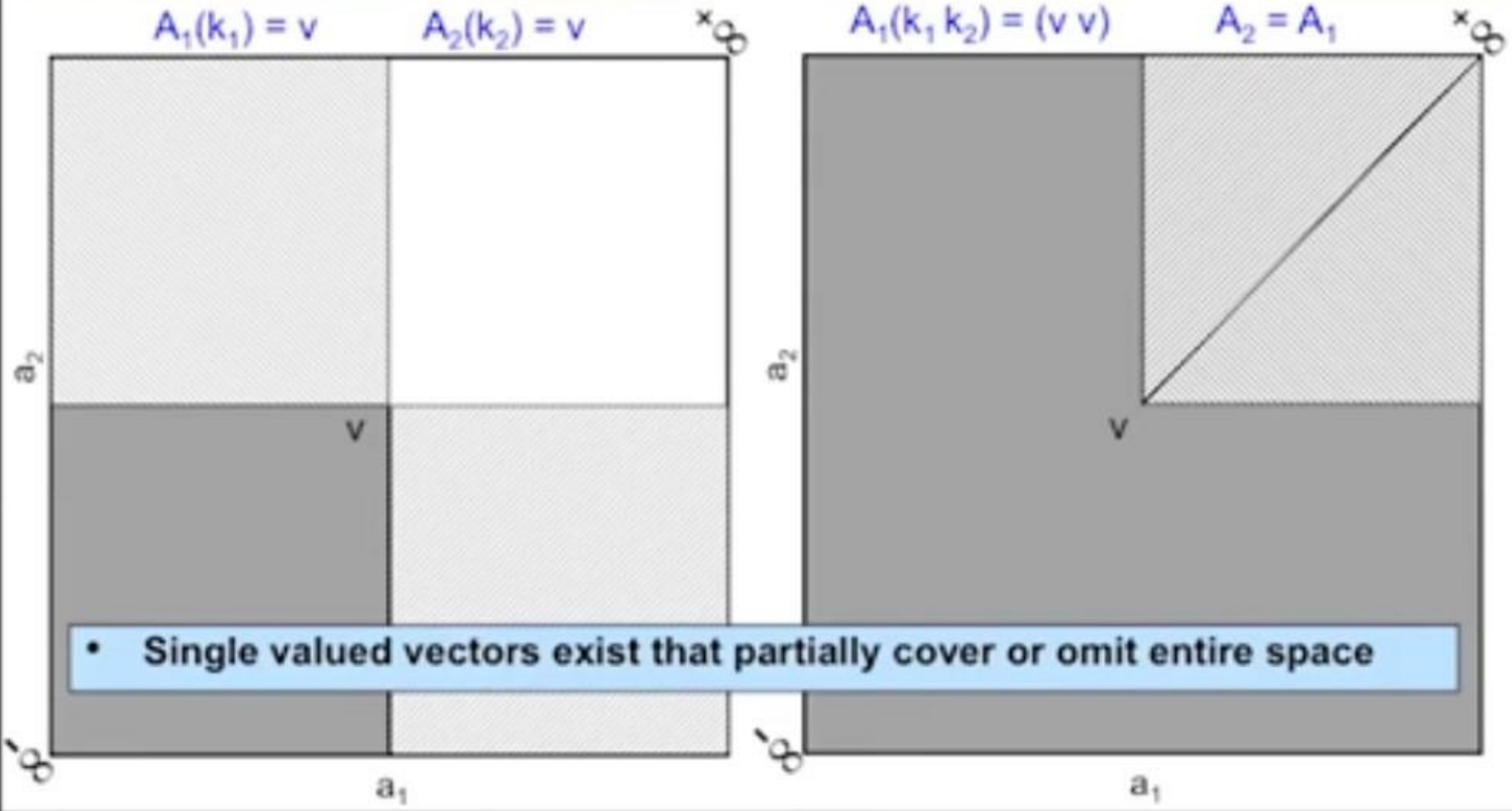
Canonical Vectors



- **Canonical vectors exist that span or omit entire space**

a_1, a_2 unique
a_1 unique
a_2 unique
a_1, a_2 not unique

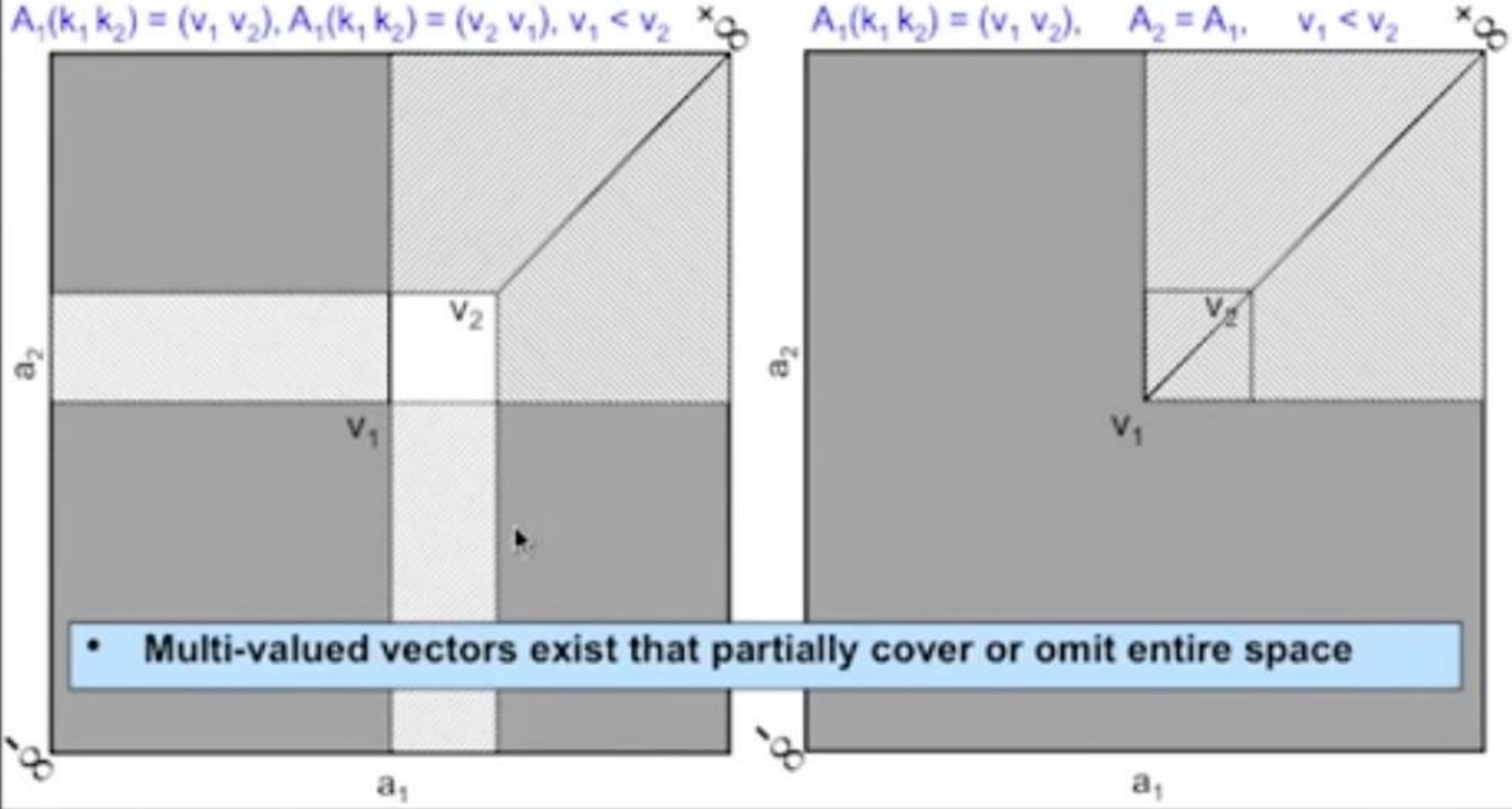
Single Valued Vectors



- **Single valued vectors exist that partially cover or omit entire space**

a_1, a_2 unique | a_1 unique | a_2 unique | a_1, a_2 not unique

Multi-Valued Vectors



- **Multi-valued vectors exist that partially cover or omit entire space**

a_1, a_2 unique · a_1 unique · a_2 unique · a_1, a_2 not unique

Outline

- Introduction
- Definitions
- Group Theory
- Vector Space
- Linear Algebra
  - Transpose
  - Special Matrices
  - Matrix Multiply
  - Identity
  - Inverses
  - Eigenvectors
- Summary

Matrix Transpose

- Swap keys (rows and columns)

$$A(r,c)^T = A(c,r)$$

- No change with even number of transposes

- Transpose distributes across $\oplus$ and scalar $\otimes$

$$((a_1 \otimes A_1) \oplus (a_2 \otimes A_1))^T = (a_1 \otimes A_1^T) \oplus (a_2 \otimes A_1^T)$$

- **Similar to linear algebra**

Special Matrices

- Submatrices [Yes]
- Zero matrix [Yes?] (empty set)
- Square matrix [Yes]
- Diagonal matrix [Yes]
- Upper/lower triangular [Yes]
- Skew symmetric [No] (no $\oplus$ inverse)
- Hermitian [No] (no $\oplus$ inverse)
- Elementary row/column operations [Yes?]
  - Swap both keys or values? No $\otimes$ inverse.
  - If both key and value swap, then equivalent to matrix multiply
- Row/column equivalence [Yes?]
  - If limit to swaps

- **Similar and different from linear algebra**
- **Possible to construct these forms, but may not be applicable to associative arrays that have fixed keys (i.e., functions over a keys)**

Matrix Multiply

- Matrix multiply

$$A_3 = A_1\,A_2 = A_1 \oplus.\otimes A_2$$

- Always conformant (can multiply any sizes)

- Inner product formulation (computation)

$$A_3(r_i,c_j) = \oplus_k\ (\ A_1(r_i,k) \otimes A_2(k,c_j)\ )$$

- Outer product formulation (theory)

$$A_k(r_i,c_j) = A_1(r_i,k) \otimes A_2(k,c_j)$$
$$A_3 = \oplus_k\ A_k$$

- **Different from linear algebra**
- **Associative arrays have no conformance requirements**

Matrix Multiply Examples

- 1x2 Row matrix: $A_1(r, k_1 k_2) = v_1$
- 2x1 Column matrix: $A_1(k_2 k_3, c) = v_2$

- Example 1: 1x1 Matrix: $A_3(r,c) = A_1 A_2 =$ [See Table]
- Example 2: 2x2 Matrix ($r \neq c$): $A_3(k_1 k_2, k_2 k_3) = A_2 A_1 =$ [See Table]
- Example 3: 2x2 Matrix ($r = c$): $A_3(k_1 k_2, k_2 k_3) = A_2 A_1 = f(v_1, v_2)$

- Value of A_3 depends upon specifics of $\oplus$ and $\otimes$

Example 1	$\otimes = \cup_{f()}$	$\otimes = \cap_{f()}$
$\oplus = \cup_{g()}$	$g(g(v_1, f(v_1, v_2), v_2))$	$f(v_1, v_2)$
$\oplus = \cap_{g()}$	$g(g(v_1, f(v_1, v_2), v_2))$	$\emptyset$

Example 2	$\otimes = \cup_{f()}$	$\otimes = \cap_{f()}$
$\oplus = \cup_{g()}$	$g(v_1, v_2)$	$\emptyset$
$\oplus = \cap_{g()}$	$g(v_1, v_2)$	$\emptyset$

- **Wide range of behaviors possible given specific operator choices**

Identity

- Left Identity: $I_{left} = diag(Row(A)) = 1$
- When does? $I_{left}\ A = A$

- Right Identity: $I_{right} = diag(Col(A)) = 1$
- When does? $A\ I_{right} = A$

- Generally possible when

$$\oplus = \cup_{g()} \qquad \otimes = \cap_{f()}$$

- In some circumstances

$$I = I_{left} \oplus I_{right} \quad \text{and} \quad A\ I = A = I\ A$$

- **Similar to linear algebra for a limited set of $\oplus$ and $\otimes$**

Inverses

- Left Inverse: $A\,A^{-1} = I_{left}$
- Right Inverse: $A^{-1}\,A = I_{right}$
- Is it possible to construct matrix inverses with no $\oplus$ inverse and no $\otimes$ inverse
- Generally, no. Exception
  - A is a column/row vector
  - $\oplus = \cup_{g()}$, $\otimes = \cap_{f()}$
  - $I_{right/left}$ is 1x1 equal to "local" 1 (i.e., 1 wrt to A)

> - **Different from linear algebra**
> - **Inverses generally do not appear in associative arrays**

Eigenvectors (simple case)

- Let $\oplus = \cup_g$, $\otimes = \cap_f$
- Let A, A_e, A_λ be NxN and have 1 element per row and column

$$A(r_i,r_i) = v_i \qquad A_e(r_i,c_i) = e_i \qquad A_\lambda(c_i,c_i) = v_i$$

- Eigenvector equation

$$A\,A_e = A_e\,A_\lambda \qquad = A_{e\lambda}$$

- where: $A_{e\lambda}(r_i,c_i) = f(v_i,e_i)$

> - **Eigenvector equation satisfied in a simple case**
> - **Row and column keys must match**

Pseudoinverse (simple case)

- Let $\oplus = \cup_g$, $\otimes = \cap_f$

- Let A, A^+ be NxN (or N_rxN_c?) and have 1 element per row and column

$$A(r_i,c_i) = v_i \qquad A^+(c_i,r_i) = v_i^+$$

- Pseudoinverse requires

$$A = A\,A^+\,A$$

$$A = A^+\,A\;A^+$$

$$(A\,A^+)^T = A\,A^+$$

$$(A\,A^+)^T = A\,A^+$$

- where: $f(v_i,v_i^+) = v_i$

- **Pseudoinverse equation satisfied in a simple case**
- **Row and column keys can be different**

Future Work: Got Theorems?

- Spanning theorems: when is a span a vector space?
- Linear dependence: adding a vector doesn't change span?
- Identity Array: when do left/right identity exist?
- Inverse: why doesn't it exist?
- Determinant: existance?
- Pseudoinverse: existence? How to compute?
- Linear transforms: existance?
- Norms or inner product space
- Compressive sensing requirements
- Eigenvectors
- Convolution (with next operator)
- Complementary matrices

- For which $\oplus$, $\otimes$, 0/1 do these apply

Summary

- Algebra of Associative Arrays provides the mathematics for representing and operating on Spreadsheets and Big Tables
- Small number of assumptions yields a rich mathematical environment
- Much of linear algebra is available without $\oplus$ inverse and $\otimes$ inverse

Relational Model High Level Comparison

	Relational Database	Associative Arrays
Fill	Dense	Sparse
Columns	Static	Dynamic
Data	Typed	Untyped
#Rows	Unlimited	Unlimited
#Columns	Small	Unlimited
Dimensions	2 different	N same
Main Operation	Join	Linear Algebra

- Relational algebra (Codd 1970) is the de facto theory of databases
- The design goal of relational algebra and associative arrays algebra are fundamentally different
- Result in a fundamental differences in the theory

Relational Model High Level Comparison

	Relational Database	Associative Arrays
Fill	Dense	Sparse
Columns	Static	Dynamic
Data	Typed	Untyped
#Rows	Unlimited	Unlimited
#Columns	Small	Unlimited
Dimensions	2 different	N same
Main Operation	Join	Linear Algebra

- Relational algebra (Codd 1970) is the de facto theory of databases
- The design goal of relational algebra and associative arrays algebra are fundamentally different
- Result in a fundamental differences in the theory