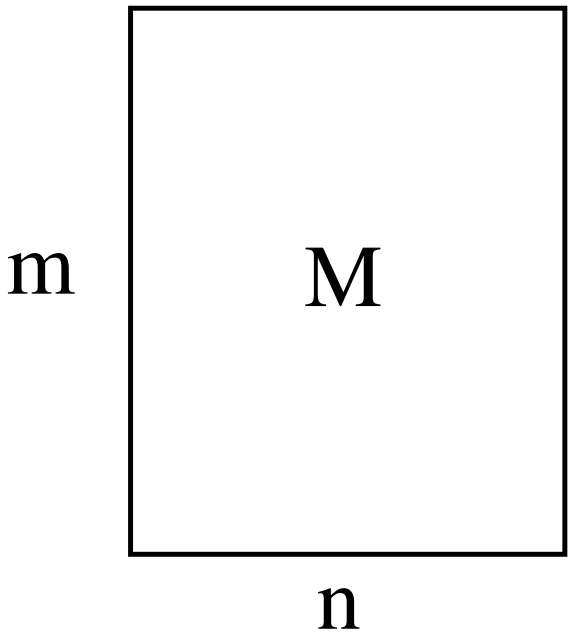
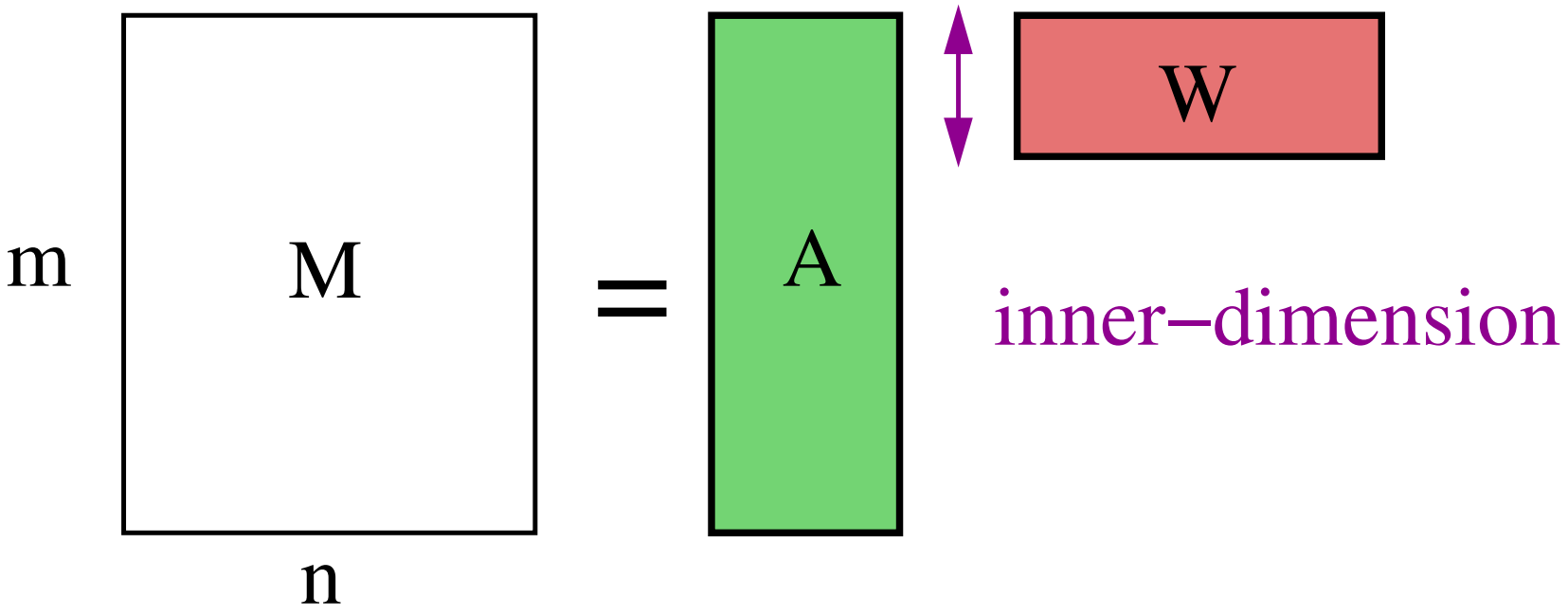
Nonnegative Matrix Factorization: Algorithms, Complexity and Applications

Ankur Moitra

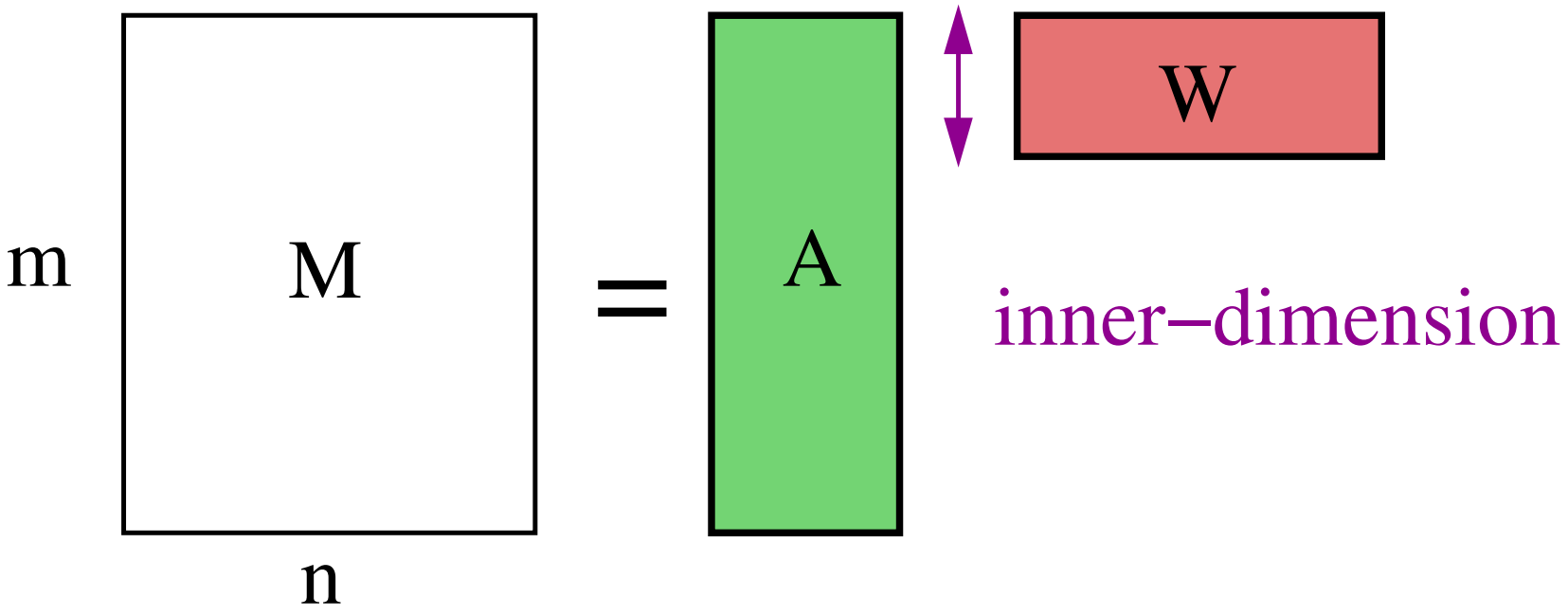
Massachusetts Institute of Technology

July 6th, 2015

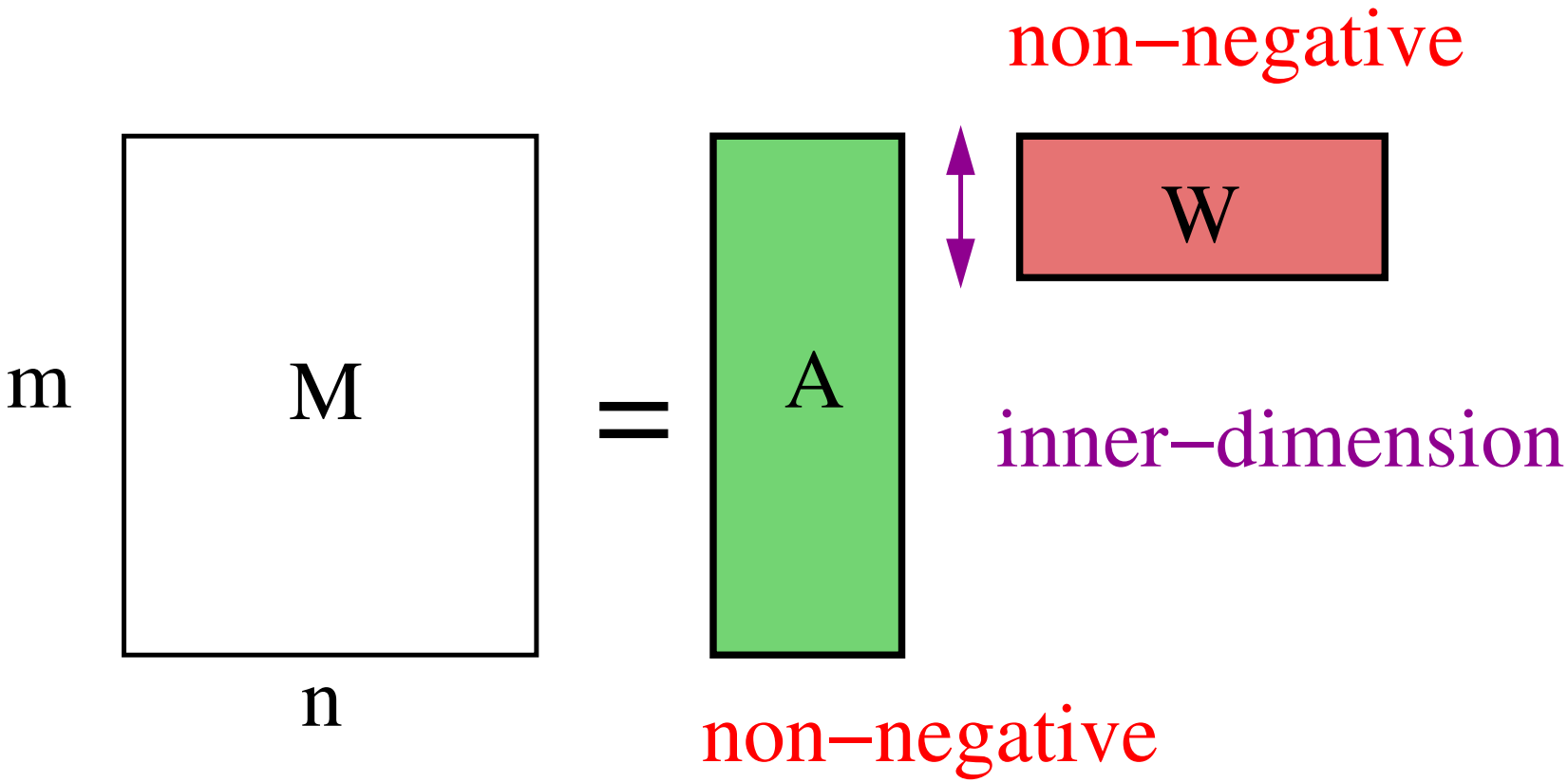

m
M
n


m
M
n
=
A
W
inner–dimension

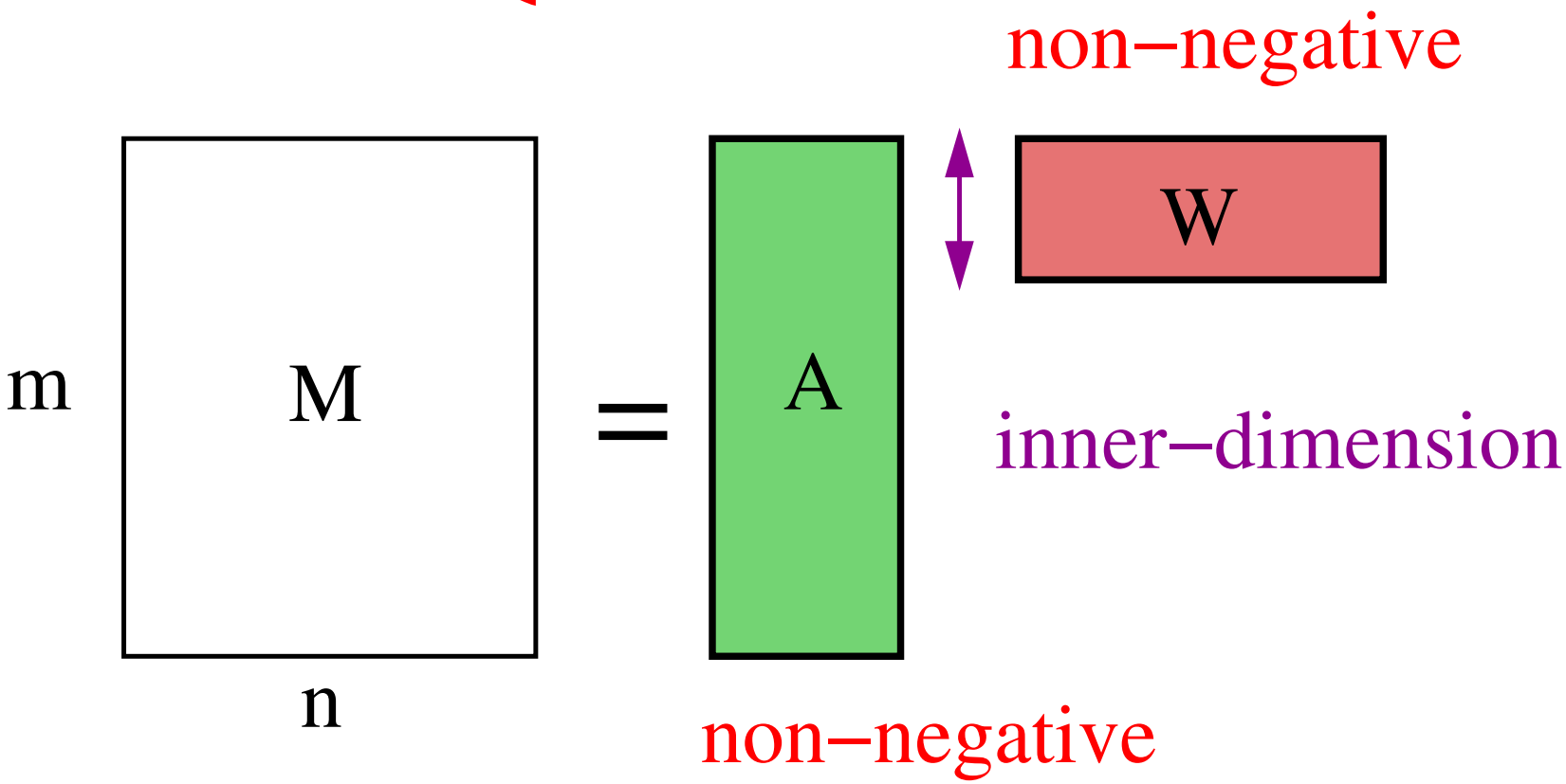
rank



rank



non−negative rank



Equivalent Definitions

The nonnegative rank $\text{rank}^+(M)$ can be defined many ways:

Equivalent Definitions

The nonnegative rank $\text{rank}^+(M)$ can be defined many ways:

- The smallest r such that there is a factorization $M = AW$ where A and W are nonnegative and have inner-dimension r

Equivalent Definitions

The nonnegative rank $\text{rank}^+(M)$ can be defined many ways:

- The smallest r such that there is a factorization $M = AW$ where A and W are nonnegative and have inner-dimension r
- The smallest r such that there are r nonnegative vectors $A_1, A_2, ...A_r$ and every column in M can be represented as a nonnegative combination of them

Equivalent Definitions

The nonnegative rank $\text{rank}^+(M)$ can be defined many ways:

- The smallest r such that there is a factorization $M = AW$ where A and W are nonnegative and have inner-dimension r
- The smallest r such that there are r nonnegative vectors $A_1, A_2, ...A_r$ and every column in M can be represented as a nonnegative combination of them
- The smallest r such that $M = \sum_{i=1}^{r} M^{(i)}$ where each $M^{(i)}$ is rank one and nonnegative

Equivalent Definitions

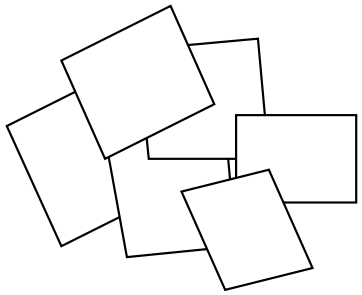
The nonnegative rank $\text{rank}^+(M)$ can be defined many ways:

- The smallest r such that there is a factorization $M = AW$ where A and W are nonnegative and have inner-dimension r
- The smallest r such that there are r nonnegative vectors $A_1, A_2, ...A_r$ and every column in M can be represented as a nonnegative combination of them
- The smallest r such that $M = \sum_{i=1}^{r} M^{(i)}$ where each $M^{(i)}$ is rank one and nonnegative

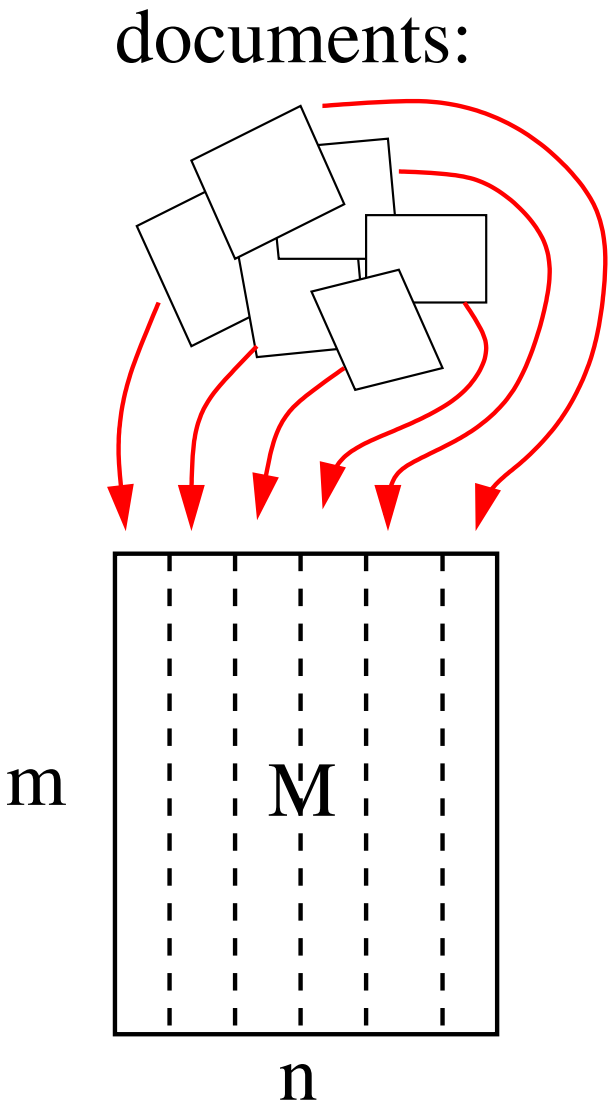
$\text{rank}(M) \leq \text{rank}^+(M)$, but it can be much larger too!

Information Retrieval

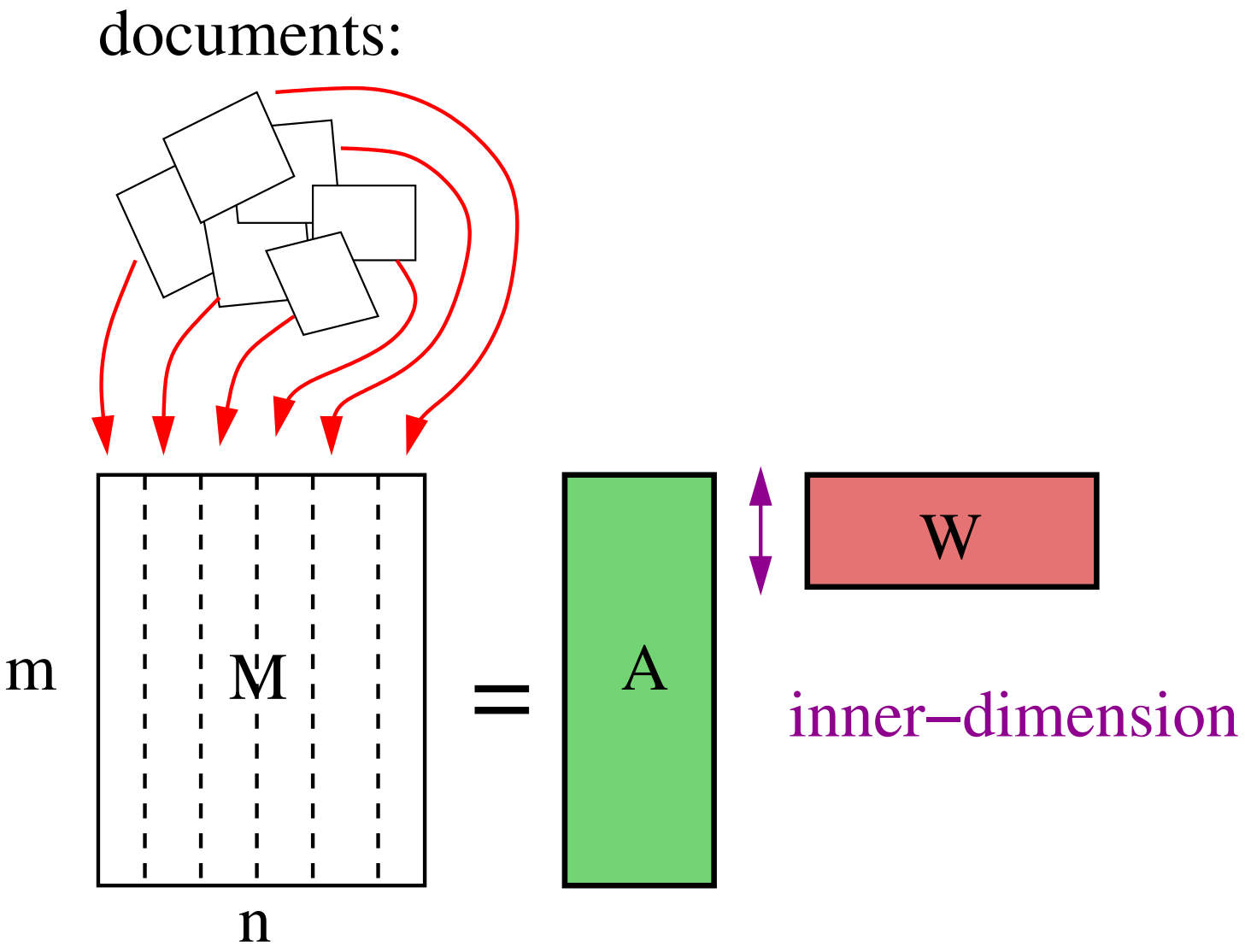
documents:



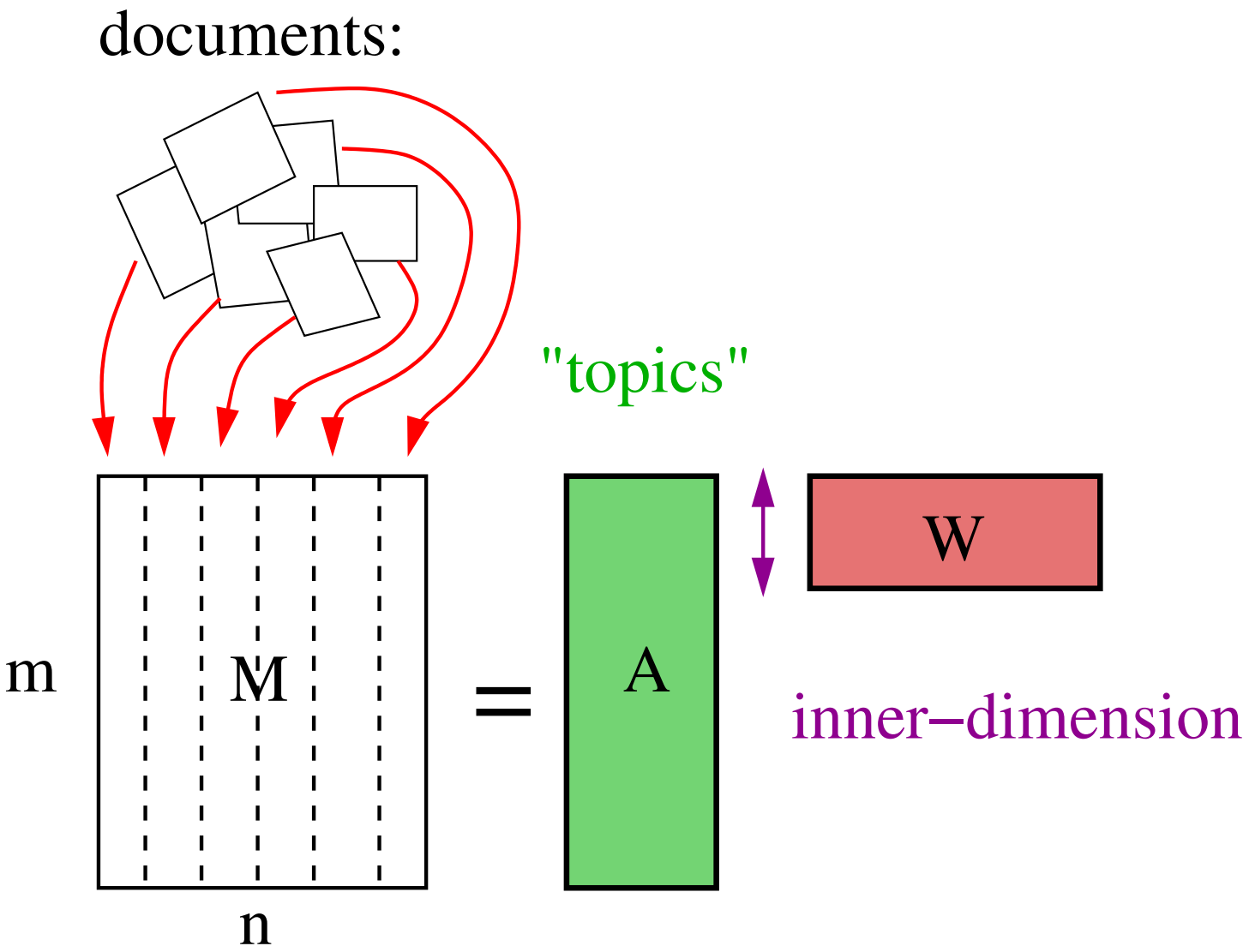
Information Retrieval



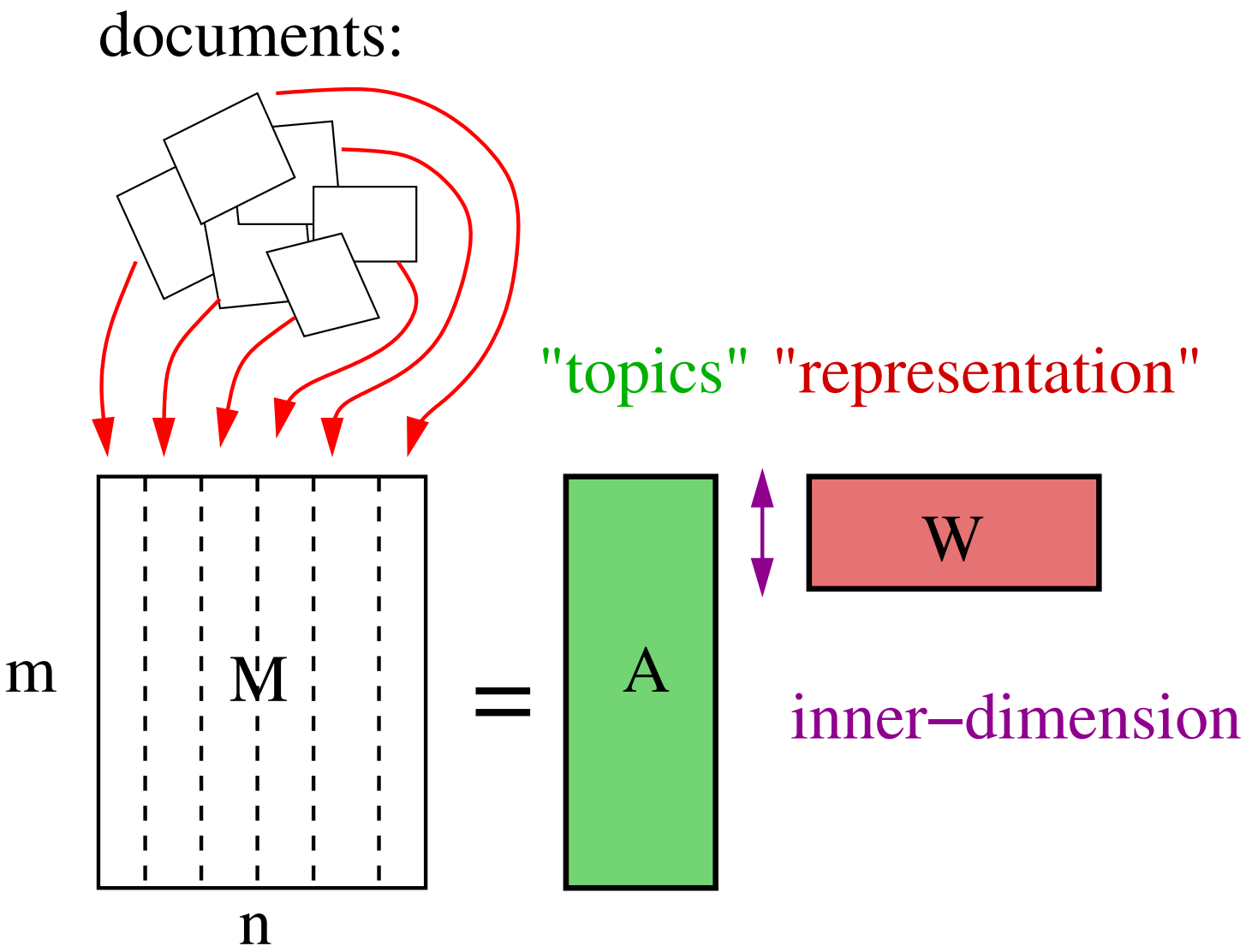
Information Retrieval



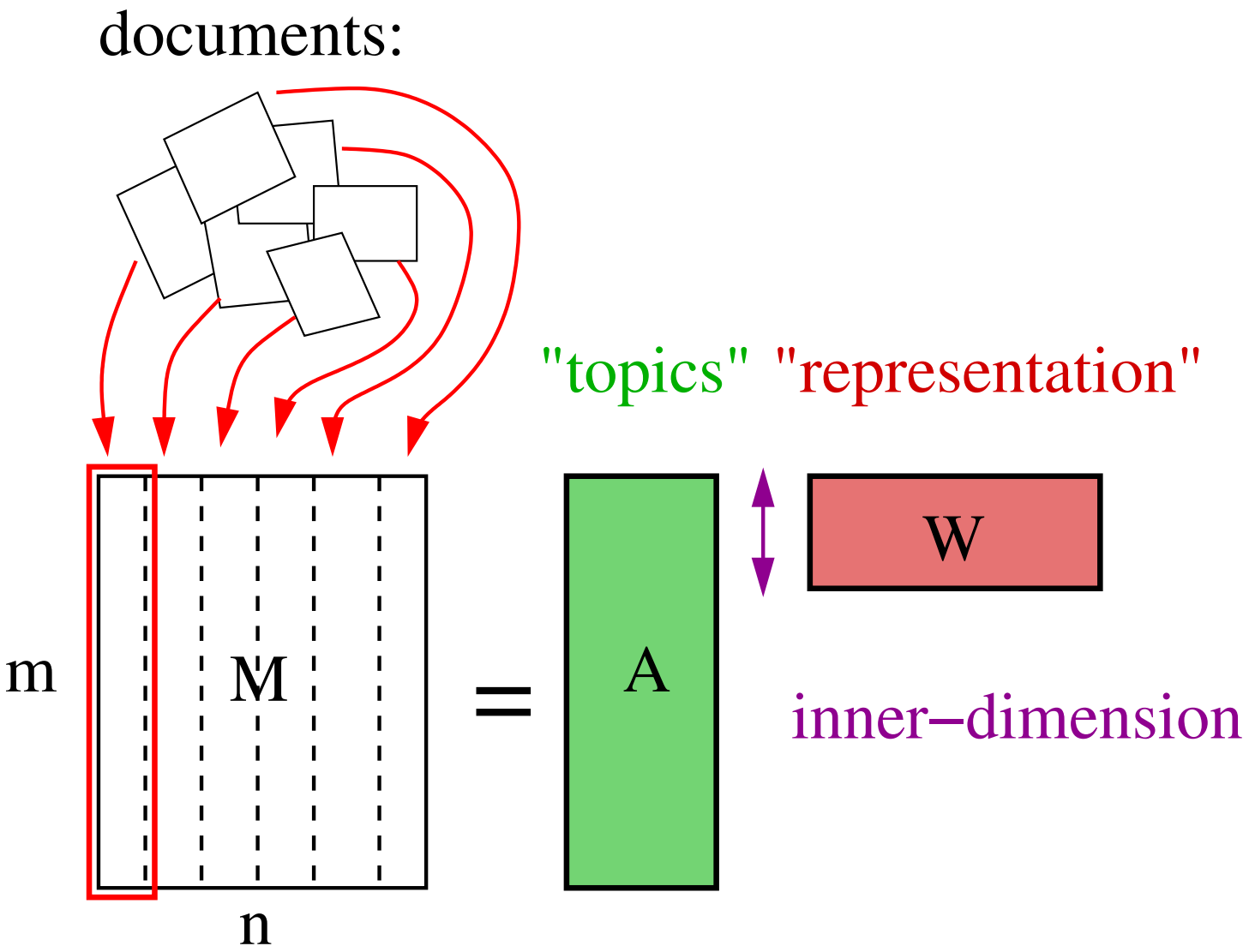
Information Retrieval



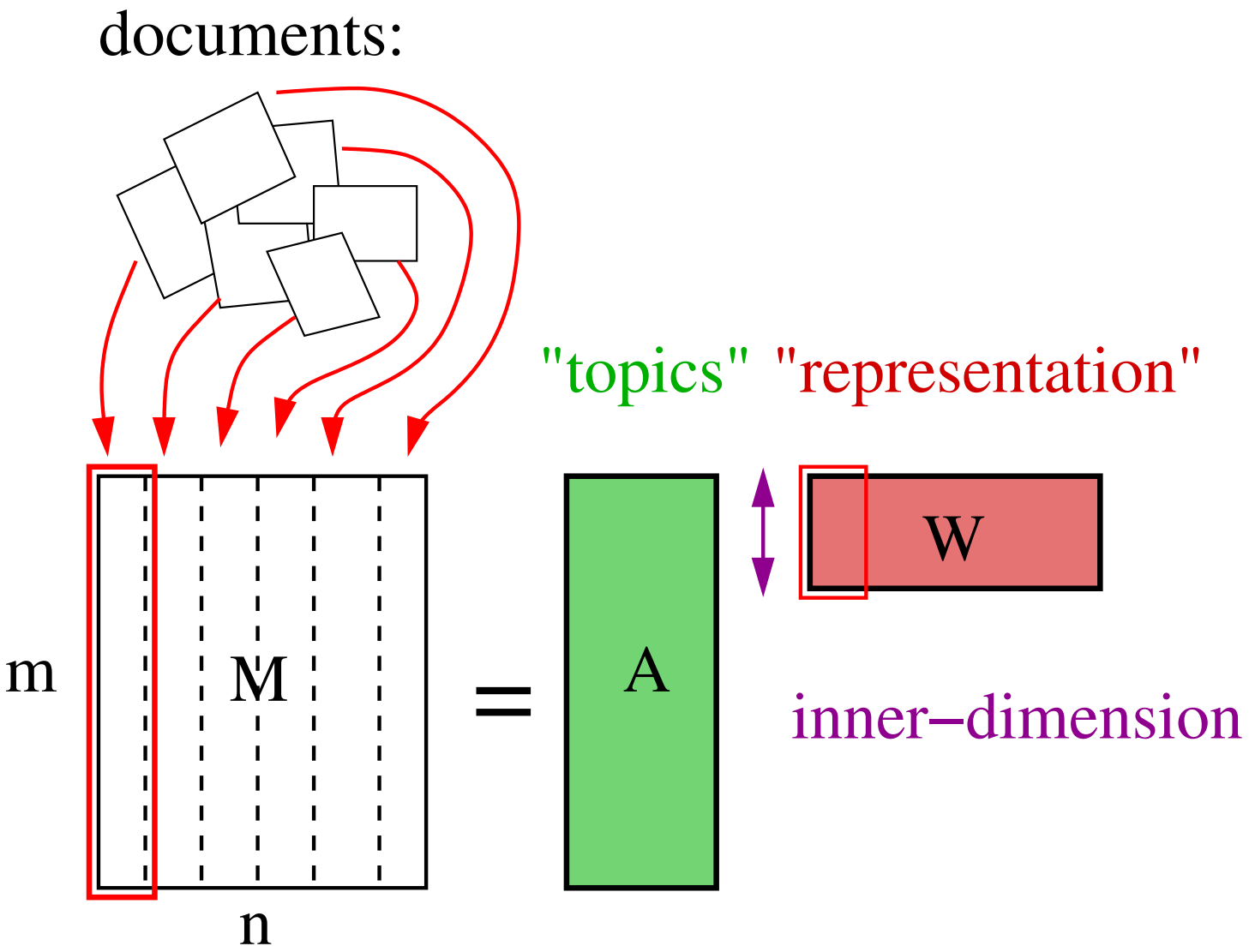
Information Retrieval



Information Retrieval



Information Retrieval



Applications

- Statistics and Machine Learning:
  - extract latent relationships in data
  - image segmentation, text classification, information retrieval, collaborative filtering, ...

    [Lee, Seung], [Xu et al], [Hofmann], [Kumar et al], [Kleinberg, Sandler]

Applications

- Statistics and Machine Learning:
  - extract latent relationships in data
  - image segmentation, text classification, information retrieval, collaborative filtering, ...

    [Lee, Seung], [Xu et al], [Hofmann], [Kumar et al], [Kleinberg, Sandler]
- Combinatorics:
  - extended formulation, log-rank conjecture

    [Yannakakis], [Lovász, Saks]

Applications

- Statistics and Machine Learning:
  - extract latent relationships in data
  - image segmentation, text classification, information retrieval, collaborative filtering, ...

    [Lee, Seung], [Xu et al], [Hofmann], [Kumar et al], [Kleinberg, Sandler]
- Combinatorics:
  - extended formulation, log-rank conjecture

    [Yannakakis], [Lovász, Saks]
- Physical Modeling:
  - interaction of components is additive
  - visual recognition, environmetrics

Local Search: Given A, compute W, compute A, ...

Local Search: Given A, compute W, compute A, ...

- Known to fail on worst-case inputs (stuck in local minima)

Local Search: Given A, compute W, compute A, ...

- Known to fail on worst-case inputs (stuck in local minima)
- Highly sensitive to cost function, regularization, update procedure

Local Search: Given A, compute W, compute A, ...

- Known to fail on worst-case inputs (stuck in local minima)
- Highly sensitive to cost function, regularization, update procedure

Question

Are there efficient **provable** *algorithms to compute the nonnegative rank?*

Local Search: Given A, compute W, compute A, ...

- Known to fail on worst-case inputs (stuck in local minima)
- Highly sensitive to cost function, regularization, update procedure

Question

Are there efficient **provable** *algorithms to compute the nonnegative rank?*

Can it even be computed in **finite** time?

Local Search: Given A, compute W, compute A, ...

- Known to fail on worst-case inputs (stuck in local minima)
- Highly sensitive to cost function, regularization, update procedure

Question

Are there efficient **provable** *algorithms to compute the nonnegative rank?*

Can it even be computed in **finite** time?

Theorem (Cohen, Rothblum)

There is an exact algorithm (based on solving systems of polynomial inequalities) that runs in time exponential in n, m and r

Hardness of NMF

Theorem (Vavasis)

NMF is NP-hard to compute

Hardness of NMF

Theorem (Vavasis)

NMF is NP-hard to compute

Hence it is unlikely that there is an exact algorithm that runs in time polynomial in n, m and r

Hardness of NMF

Theorem (Vavasis)

NMF is NP-hard to compute

Hence it is unlikely that there is an exact algorithm that runs in time polynomial in n, m and r

Question

Should we expect r to be large?

Hardness of NMF

Theorem (Vavasis)

NMF is NP-hard to compute

Hence it is unlikely that there is an exact algorithm that runs in time polynomial in n, m and r

Question

Should we expect r to be large?

What if you gave me a collection of 100 documents, and I told you there are 75 topics?

Hardness of NMF

Theorem (Vavasis)

NMF is NP-hard to compute

Hence it is unlikely that there is an exact algorithm that runs in time polynomial in n, m and r

Question

Should we expect r to be large?

What if you gave me a collection of 100 documents, and I told you there are 75 topics?

How quickly can we solve NMF if r is small?

The Worst-Case Complexity of NMF

Theorem (Arora, Ge, Kannan, Moitra)

There is an $(nm)^{O(2^r r^2)}$ *time exact algorithm for NMF*

The Worst-Case Complexity of NMF

Theorem (Arora, Ge, Kannan, Moitra)

There is an $(nm)^{O(2^r r^2)}$ *time exact algorithm for NMF*

Previously, the fastest (provable) algorithm for $r = 4$ ran in time exponential in n and m

The Worst-Case Complexity of NMF

Theorem (Arora, Ge, Kannan, Moitra)

There is an $(nm)^{O(2^r r^2)}$ time exact algorithm for NMF

Previously, the fastest (provable) algorithm for $r = 4$ ran in time exponential in n and m

Can we improve the exponential dependence on r?

The Worst-Case Complexity of NMF

Theorem (Arora, Ge, Kannan, Moitra)

There is an $(nm)^{O(2^r r^2)}$ time exact algorithm for NMF

Previously, the fastest (provable) algorithm for $r = 4$ ran in time exponential in n and m

Can we improve the exponential dependence on r?

Theorem (Arora, Ge, Kannan, Moitra)

An exact algorithm for NMF that runs in time $(nm)^{o(r)}$ would yield a sub-exponential time algorithm for 3-SAT

An Almost Optimal Algorithm

Theorem (Moitra)

There is an $(2^r nm)^{O(r^2)}$ *time exact algorithm for NMF*

An Almost Optimal Algorithm

Theorem (Moitra)

There is an $(2^r nm)^{O(r^2)}$ *time exact algorithm for NMF*

These algorithms are based on methods for **variable reduction**

An Almost Optimal Algorithm

Theorem (Moitra)

There is an $(2^r nm)^{O(r^2)}$ *time exact algorithm for NMF*

These algorithms are based on methods for **variable reduction**

Question

How many variables are needed to express a decision problem as a feasibility problem for a system of polynomial inequalities?

An Almost Optimal Algorithm

Theorem (Moitra)

There is an $(2^r nm)^{O(r^2)}$ *time exact algorithm for NMF*

These algorithms are based on methods for **variable reduction**

Question

How many variables are needed to express a decision problem as a feasibility problem for a system of polynomial inequalities?

Open Question

Is the true complexity of NMF exponential in r^2 *or* r*?*

Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Outline

- Introduction
- Algebraic Algorithms for NMF
  - **Systems of Polynomial Inequalities**
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Is NMF Computable?

Is NMF Computable?

[**Cohen and Rothblum**]: Yes

Is NMF Computable?

[**Cohen and Rothblum**]: Yes (**DETOUR**)

Semi-algebraic sets: s polynomials, k variables, Boolean function B

$$S = \{x_1, x_2 \ldots x_k | B(\text{sgn}(f_1), \text{sgn}(f_2), \ldots \text{sgn}(f_s)) = \text{"true"} \}$$

Semi-algebraic sets: s polynomials, k variables, Boolean function B

$$S = \{x_1, x_2 \ldots x_k | B(\text{sgn}(f_1), \text{sgn}(f_2), \ldots \text{sgn}(f_s)) = \text{"true"} \}$$

Question

How many sign patterns arise (as $x_1, x_2, \ldots x_k$ range over $\mathbb{R}^k$)?

Semi-algebraic sets: s polynomials, k variables, Boolean function B

$$S = \{x_1, x_2 \ldots x_k | B(\text{sgn}(f_1), \text{sgn}(f_2), \ldots \text{sgn}(f_s)) = \text{"true"} \}$$

Question

How many sign patterns arise (as $x_1, x_2, \ldots x_k$ range over $\mathbb{R}^k$)?

Naive bound: 3^s (all of $\{-1, 0, 1\}^s$),

Semi-algebraic sets: s polynomials, k variables, Boolean function B

$$S = \{x_1, x_2 \ldots x_k | B(\text{sgn}(f_1), \text{sgn}(f_2), \ldots \text{sgn}(f_s)) = \text{"true"} \}$$

Question

How many sign patterns arise (as $x_1, x_2, \ldots x_k$ range over $\mathbb{R}^k$)?

Naive bound: 3^s (all of $\{-1, 0, 1\}^s$), **[Milnor, Warren]**: at most $(ds)^k$, where d is the maximum degree

Semi-algebraic sets: s polynomials, k variables, Boolean function B

$$S = \{x_1, x_2 \ldots x_k | B(\text{sgn}(f_1), \text{sgn}(f_2), \ldots \text{sgn}(f_s)) = \text{"true"} \}$$

Question

How many sign patterns arise (as $x_1, x_2, \ldots x_k$ range over $\mathbb{R}^k$)?

Naive bound: 3^s (all of $\{-1, 0, 1\}^s$), **[Milnor, Warren]**: at most $(ds)^k$, where d is the maximum degree

In fact, best known algorithms (e.g. **[Renegar]**) for finding a point in S run in $(ds)^{O(k)}$ time

Is NMF Computable?

[**Cohen, Rothblum**]: Yes (**DETOUR**)

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)
- Constraints: $A, W \geq 0$ and $AW = M$ (degree two)

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)
- Constraints: $A, W \geq 0$ and $AW = M$ (degree two)

Running time for a solver is exponential in the number of **variables**

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)
- Constraints: $A, W \geq 0$ and $AW = M$ (degree two)

Running time for a solver is exponential in the number of **variables**

Question

What is the smallest formulation, measured in the number of variables?

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)
- Constraints: $A, W \geq 0$ and $AW = M$ (degree two)

Running time for a solver is exponential in the number of **variables**

Question

What is the smallest formulation, measured in the number of variables? Can we use only $f(r)$ variables?

Is NMF Computable?

[Cohen, Rothblum]: Yes (**DETOUR**)

- Variables: entries in A and W ($nr + mr$ total)
- Constraints: $A, W \geq 0$ and $AW = M$ (degree two)

Running time for a solver is exponential in the number of **variables**

Question

What is the smallest formulation, measured in the number of variables? Can we use only $f(r)$ variables? $O(r^2)$ variables?

Outline

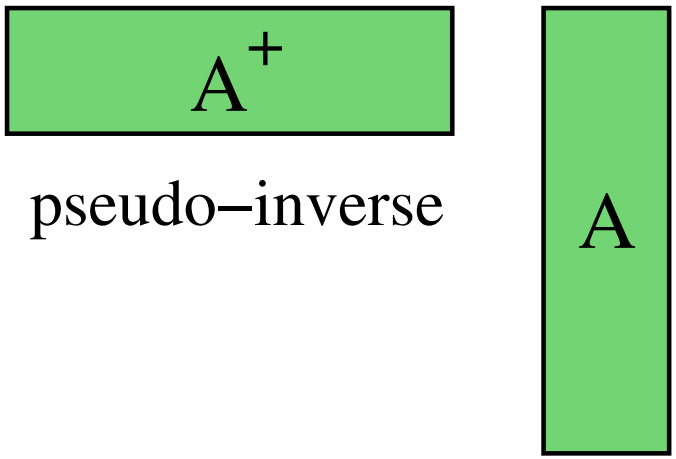
- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Outline

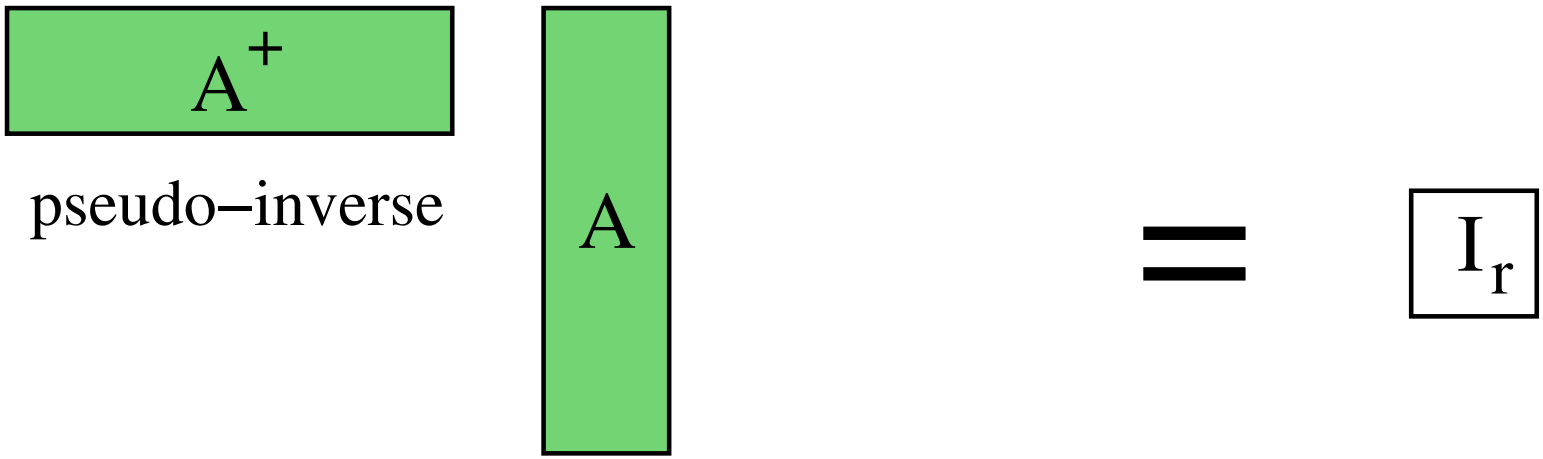
- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - **Methods for Variable Reduction**
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Easy Case: A has Full Column Rank (AGKM)

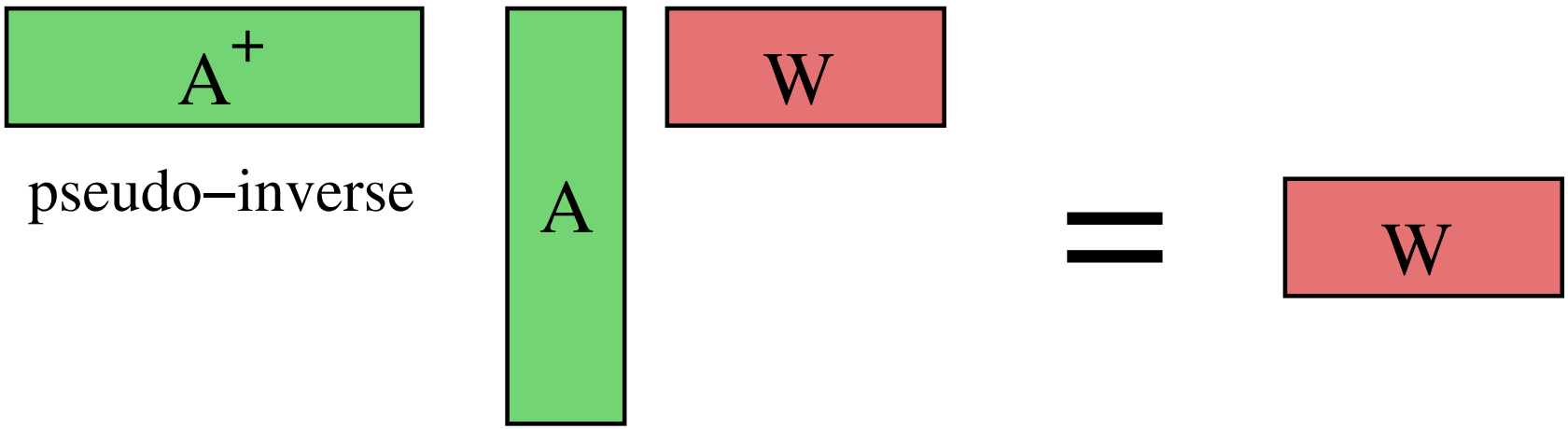
Easy Case: A has Full Column Rank (AGKM)



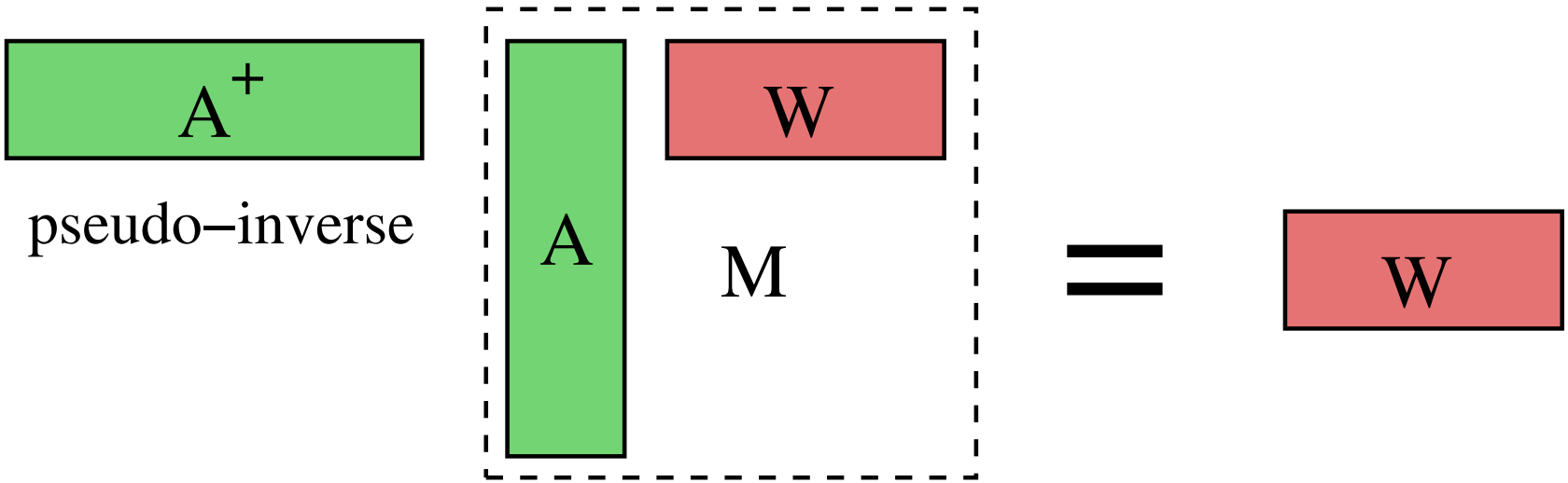
Easy Case: A has Full Column Rank (AGKM)



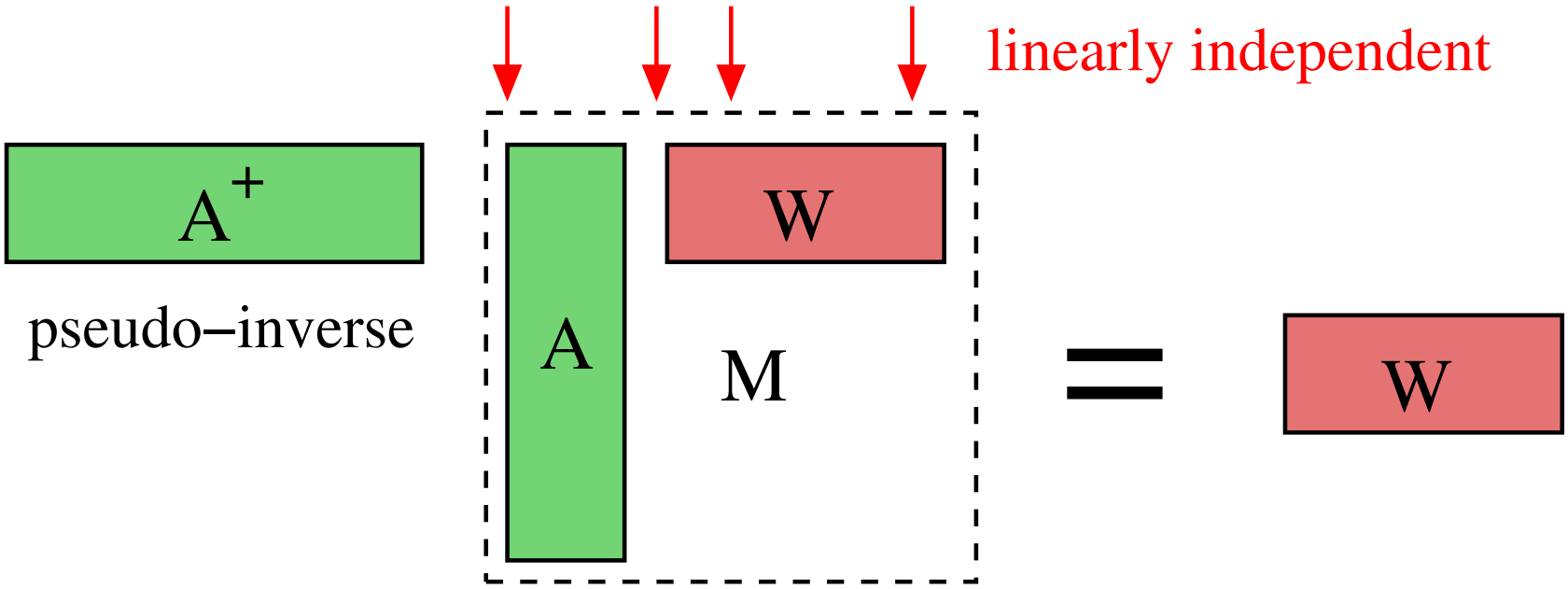
Easy Case: A has Full Column Rank (AGKM)



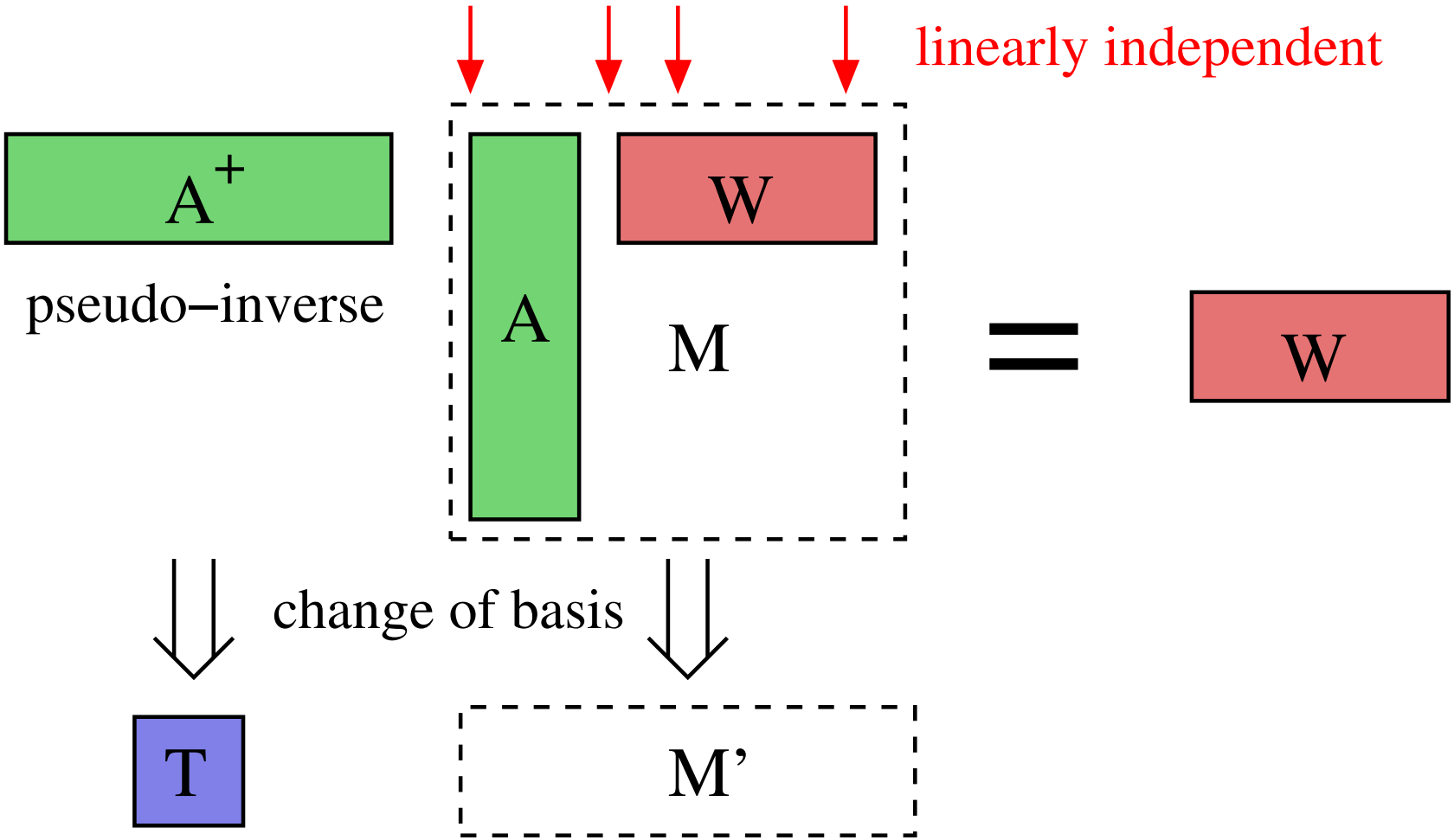
Easy Case: A has Full Column Rank (AGKM)



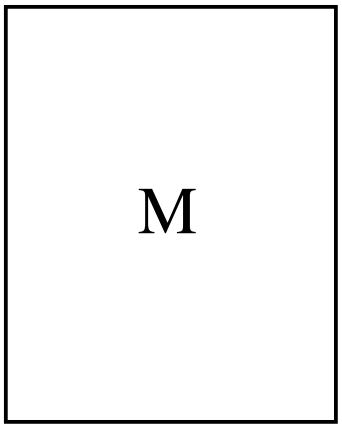
Easy Case: A has Full Column Rank (AGKM)



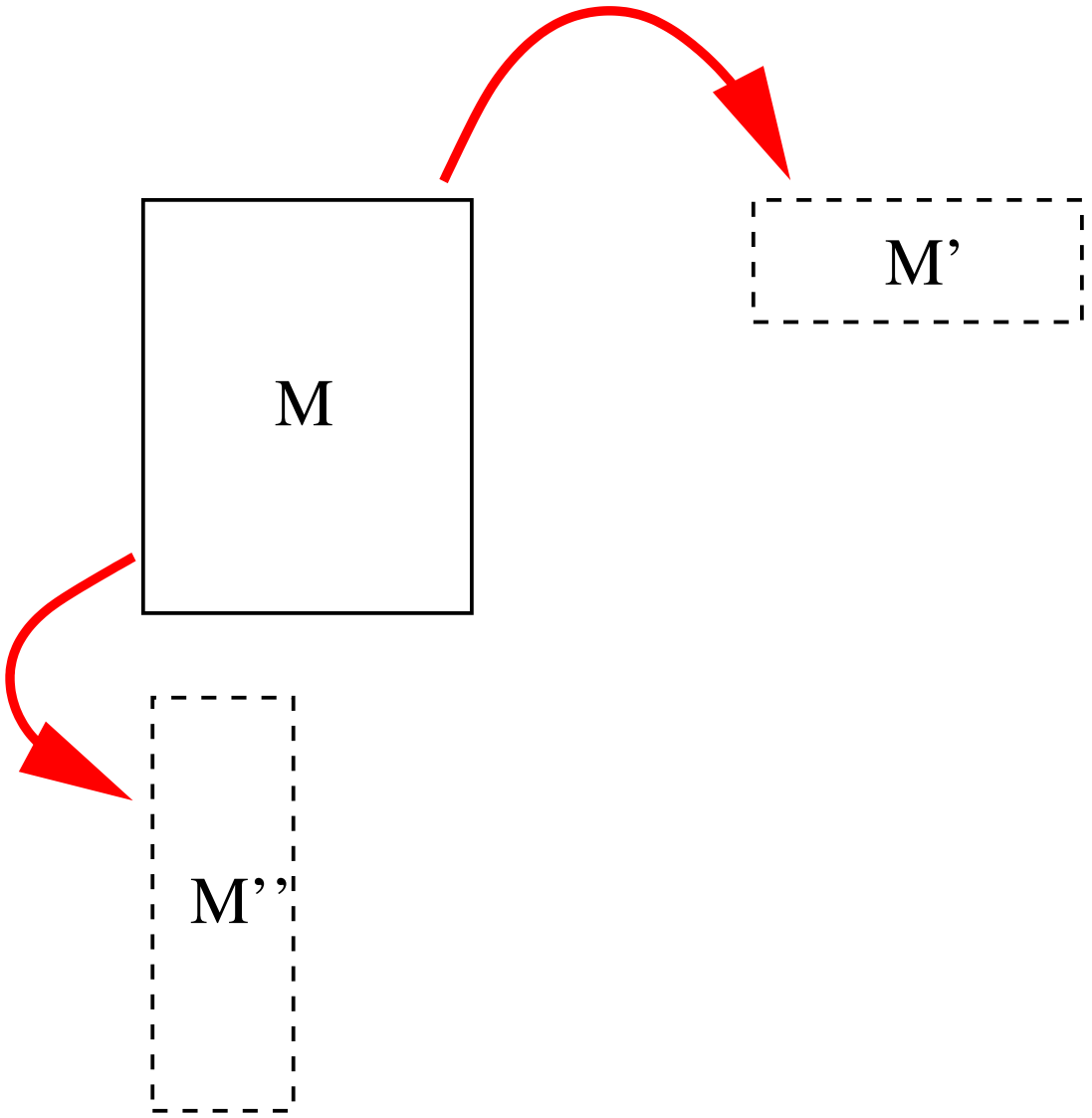
Easy Case: A has Full Column Rank (AGKM)



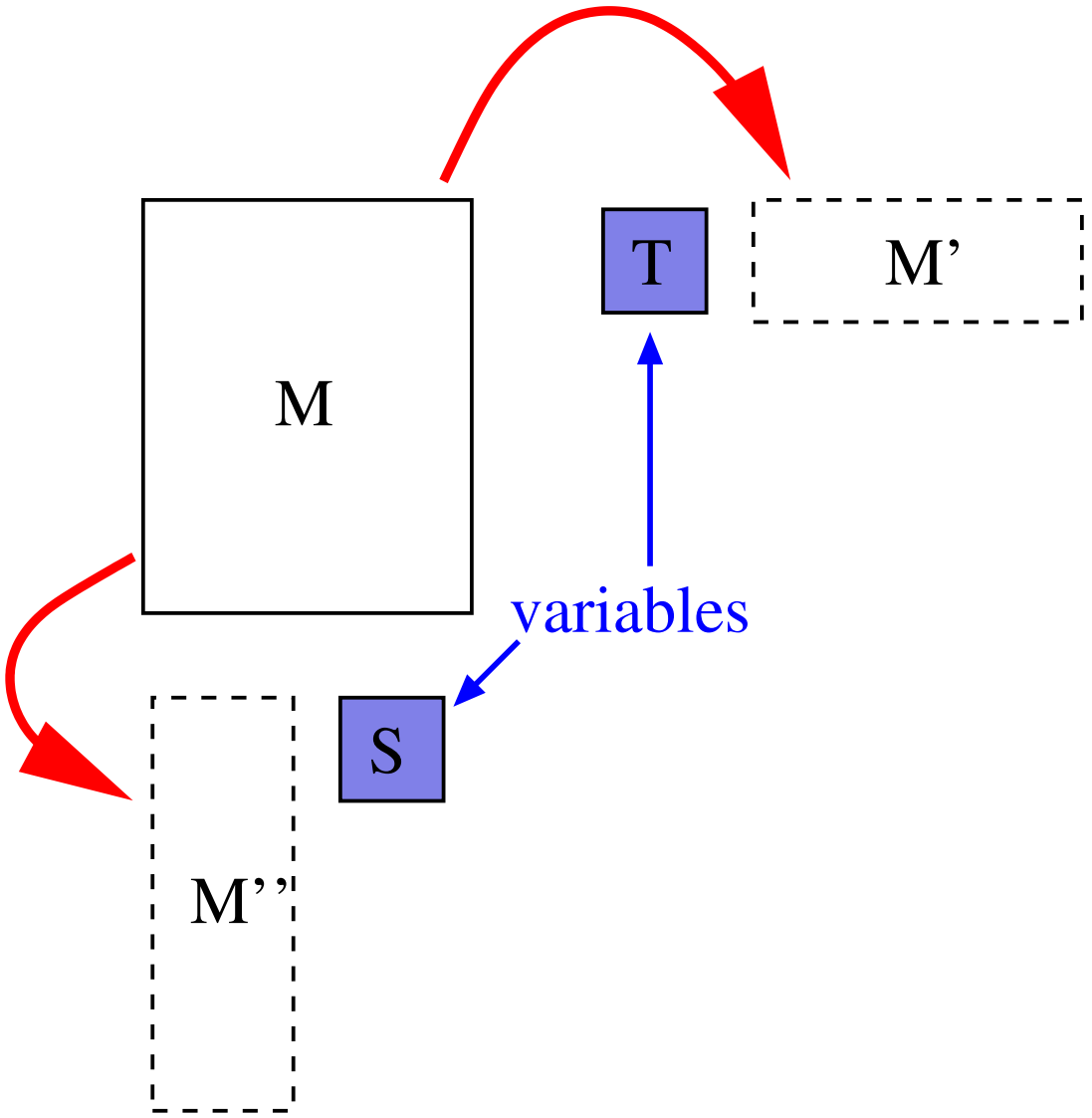
Putting it Together: Easy Case



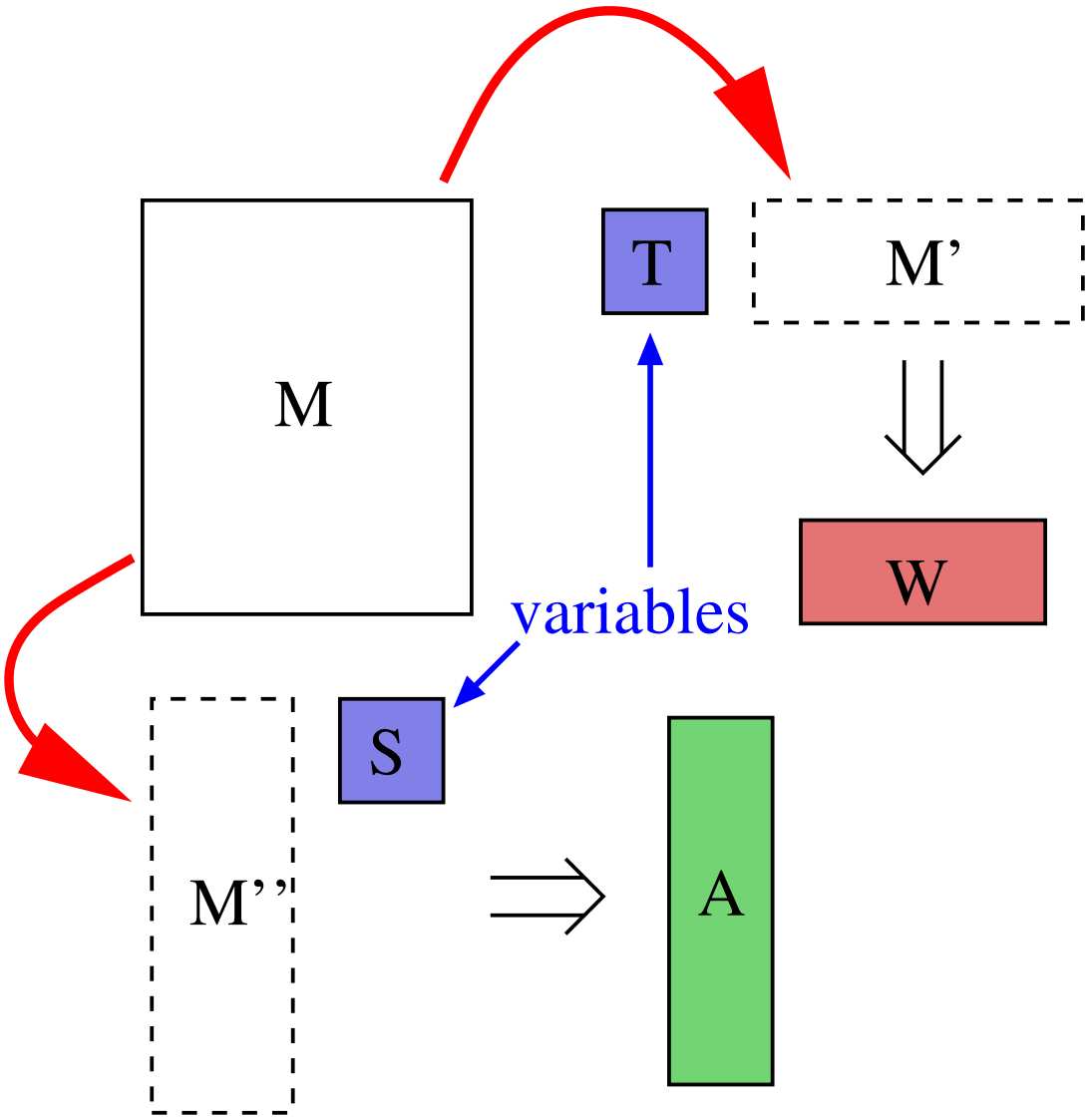
Putting it Together: Easy Case



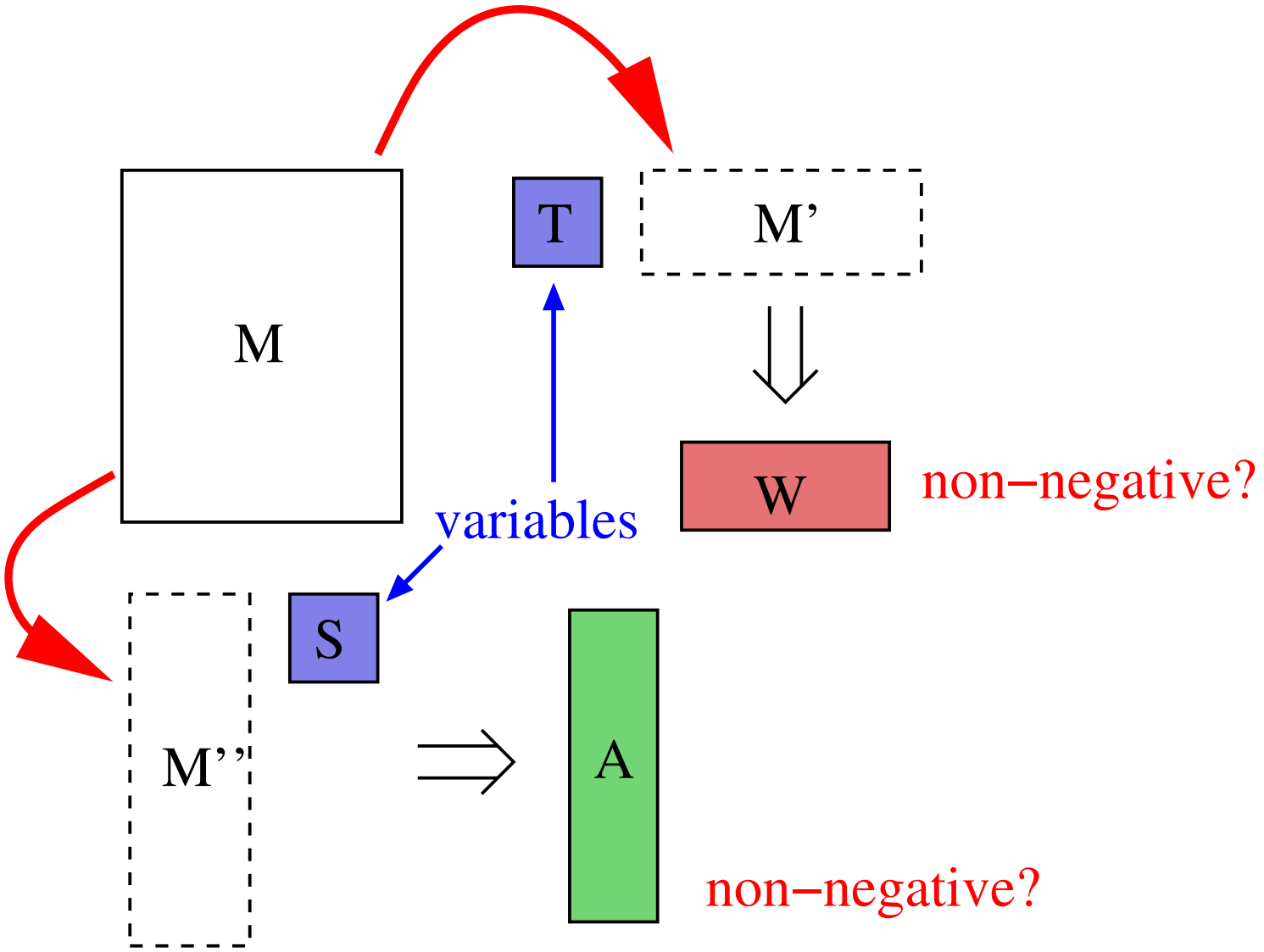
Putting it Together: Easy Case



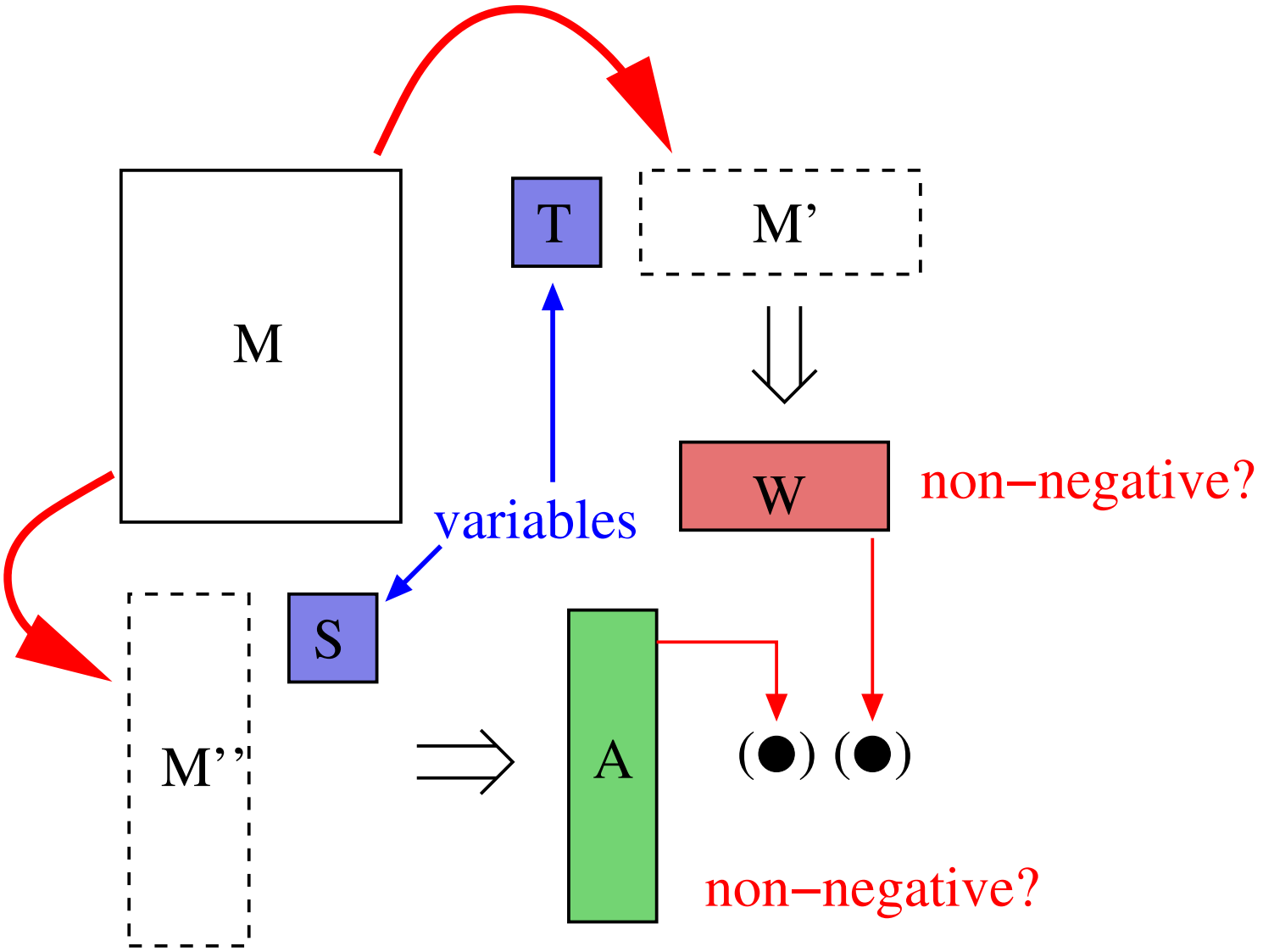
Putting it Together: Easy Case



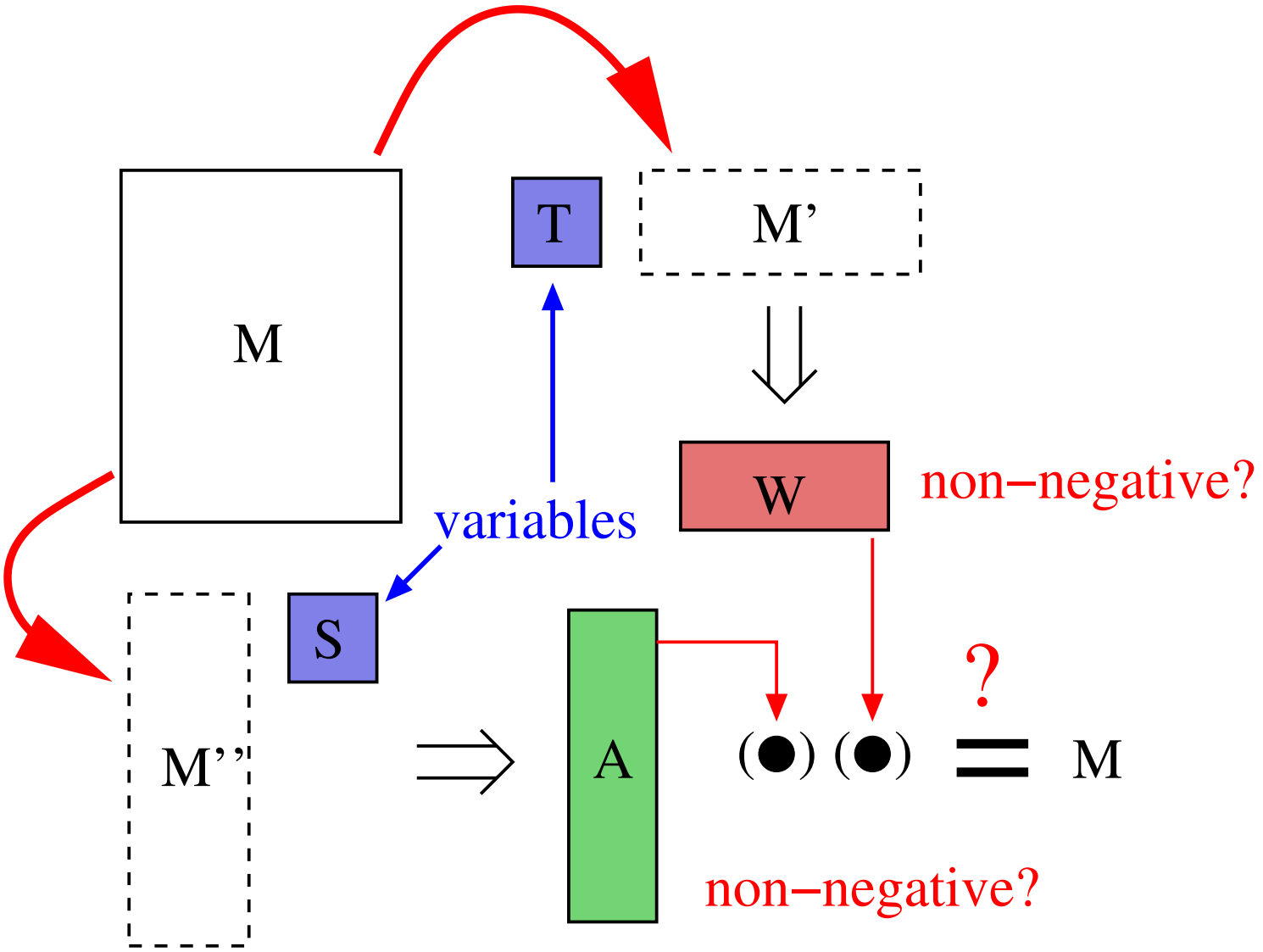
Putting it Together: Easy Case



Putting it Together: Easy Case



Putting it Together: Easy Case



General Structure Theorem

Question

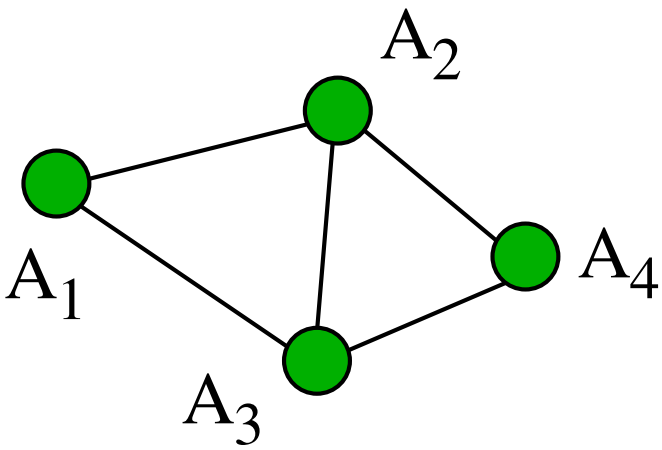
What if A does not have full column rank?

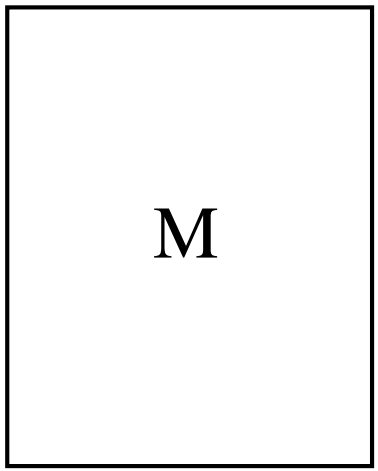
General Structure Theorem

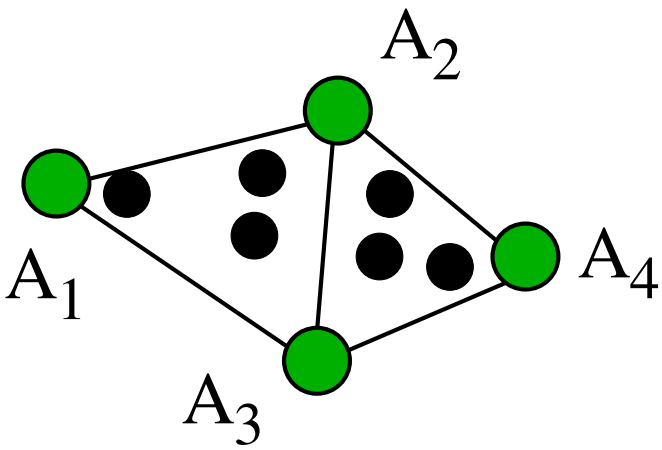
Question

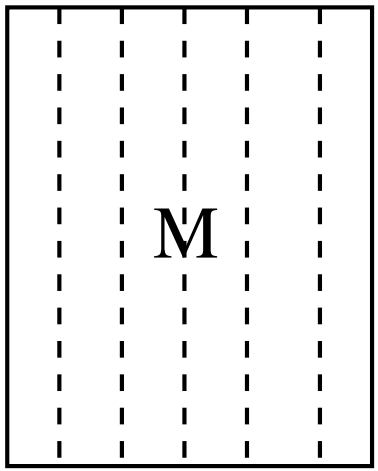
What if A does not have full column rank?

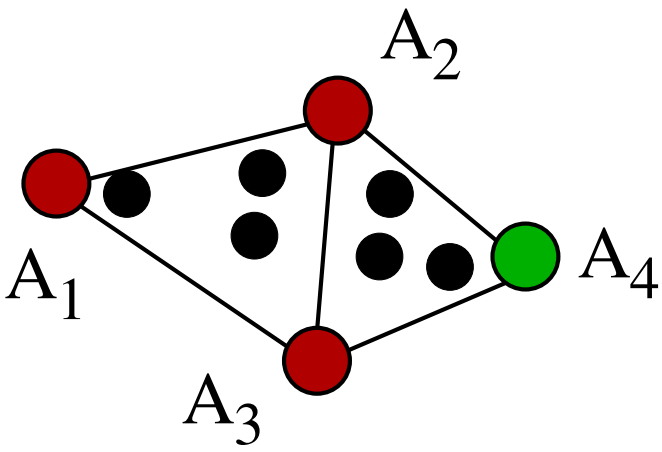
Approach: guess many linear transformations, one for each pseudo-inverse of a set of linearly independent columns

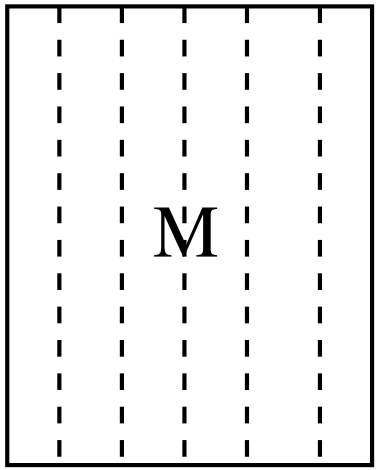

A2
A1
A4
A3


M

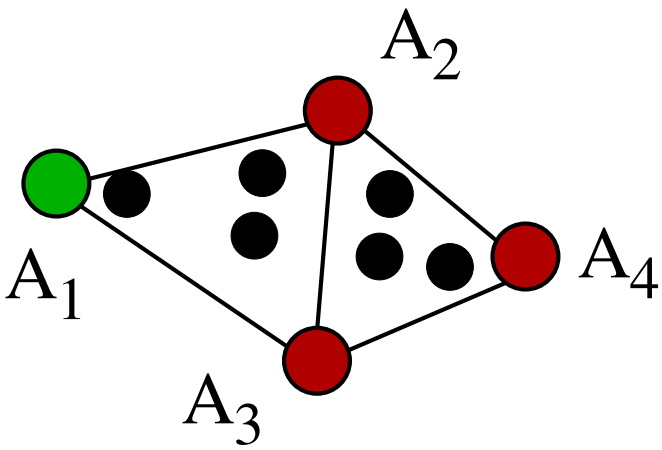

A2
A1
A4
A3

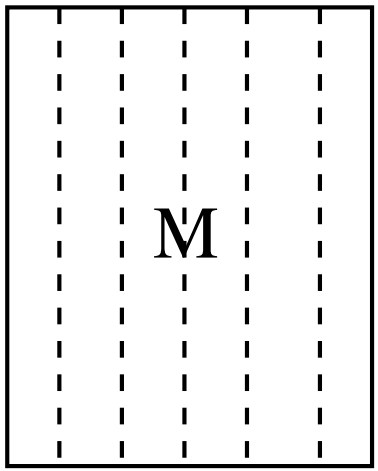

M





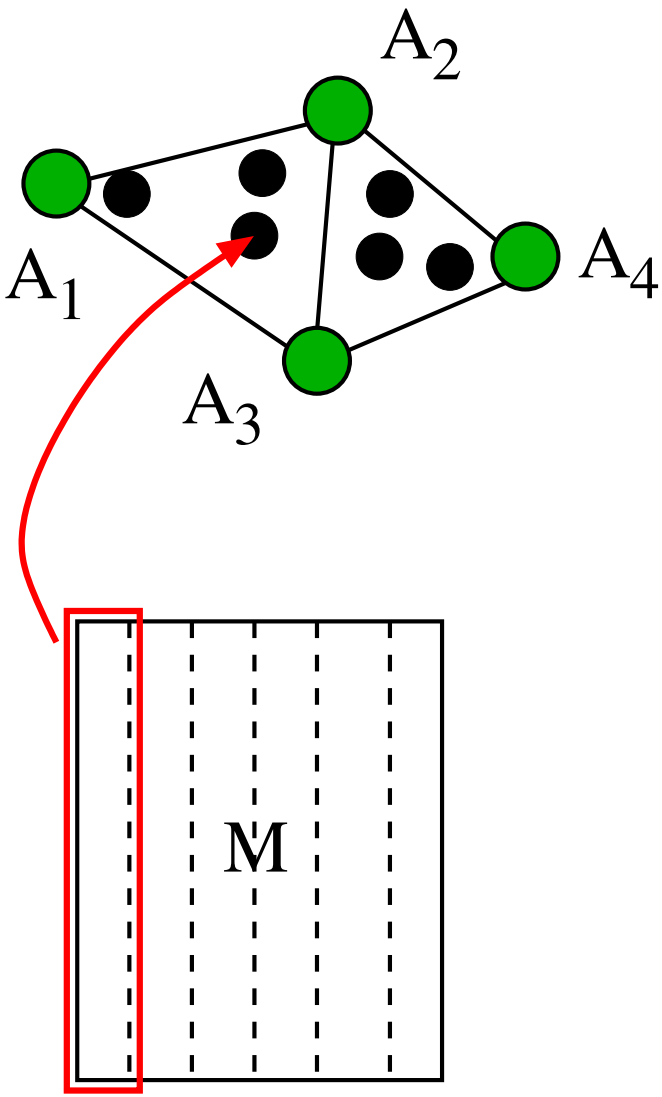
$$T_1 = (A_1 A_2 A_3)^+$$





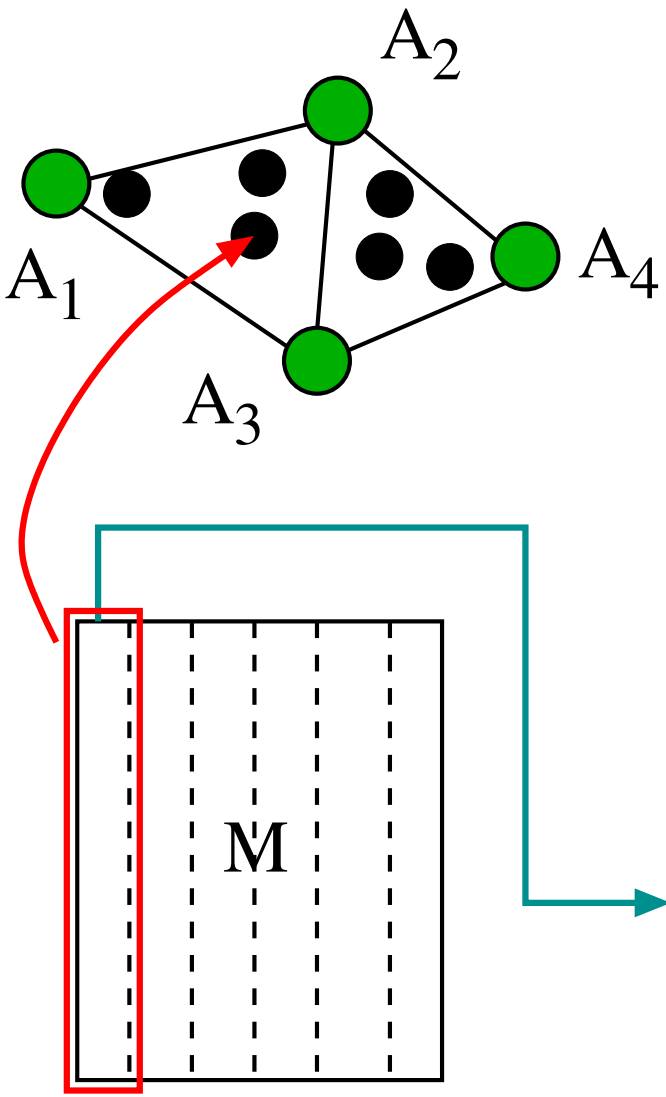
$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



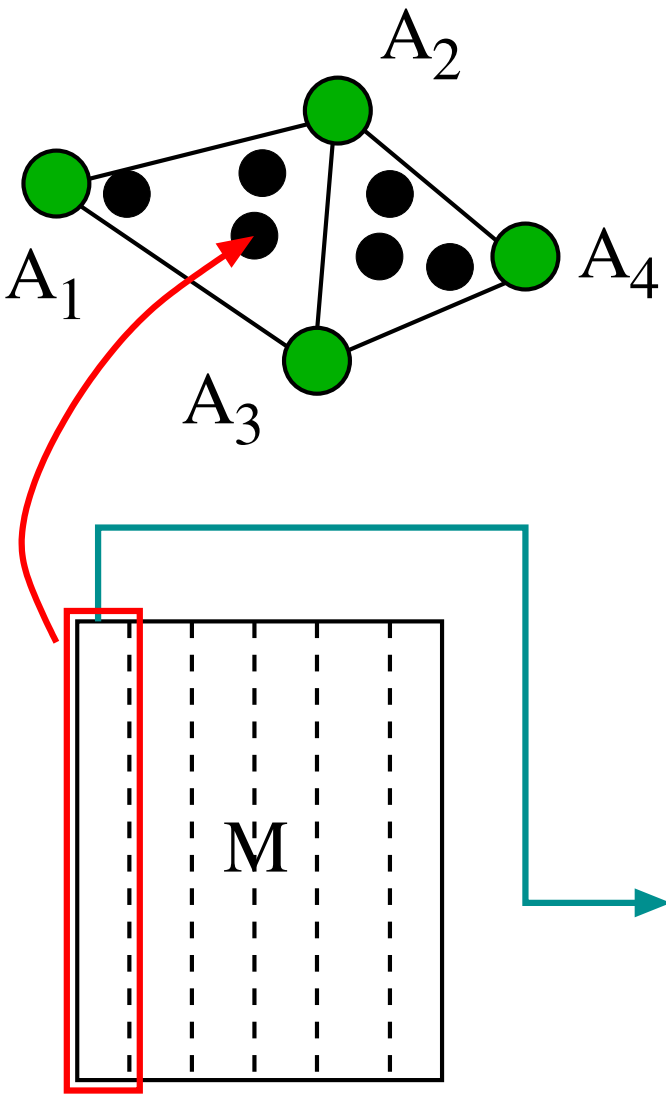
$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



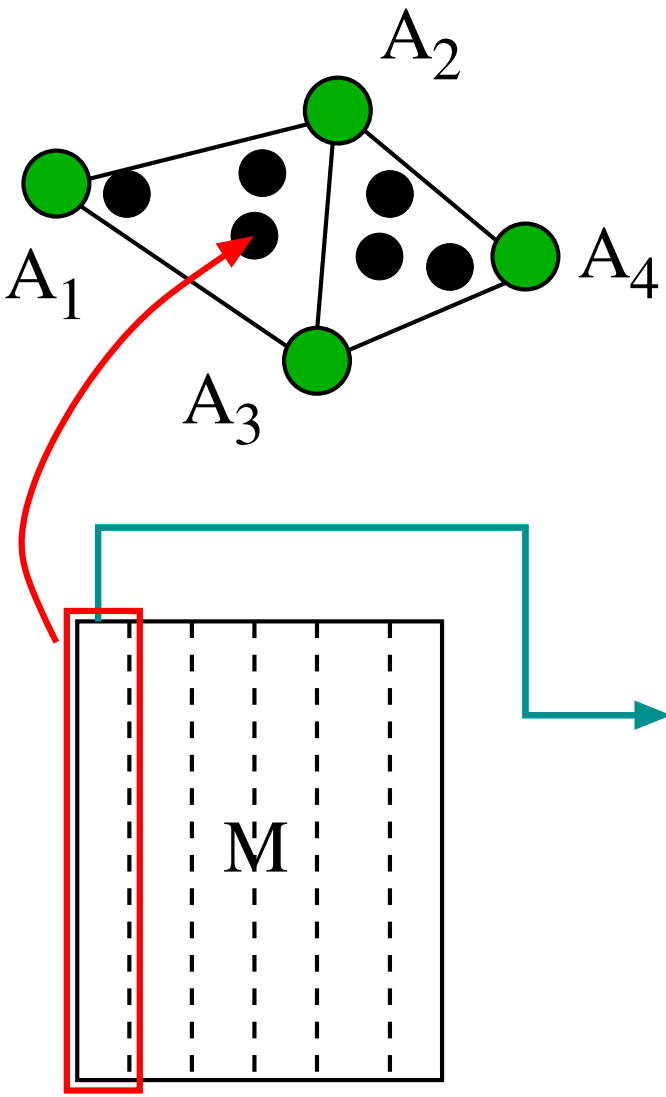
$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



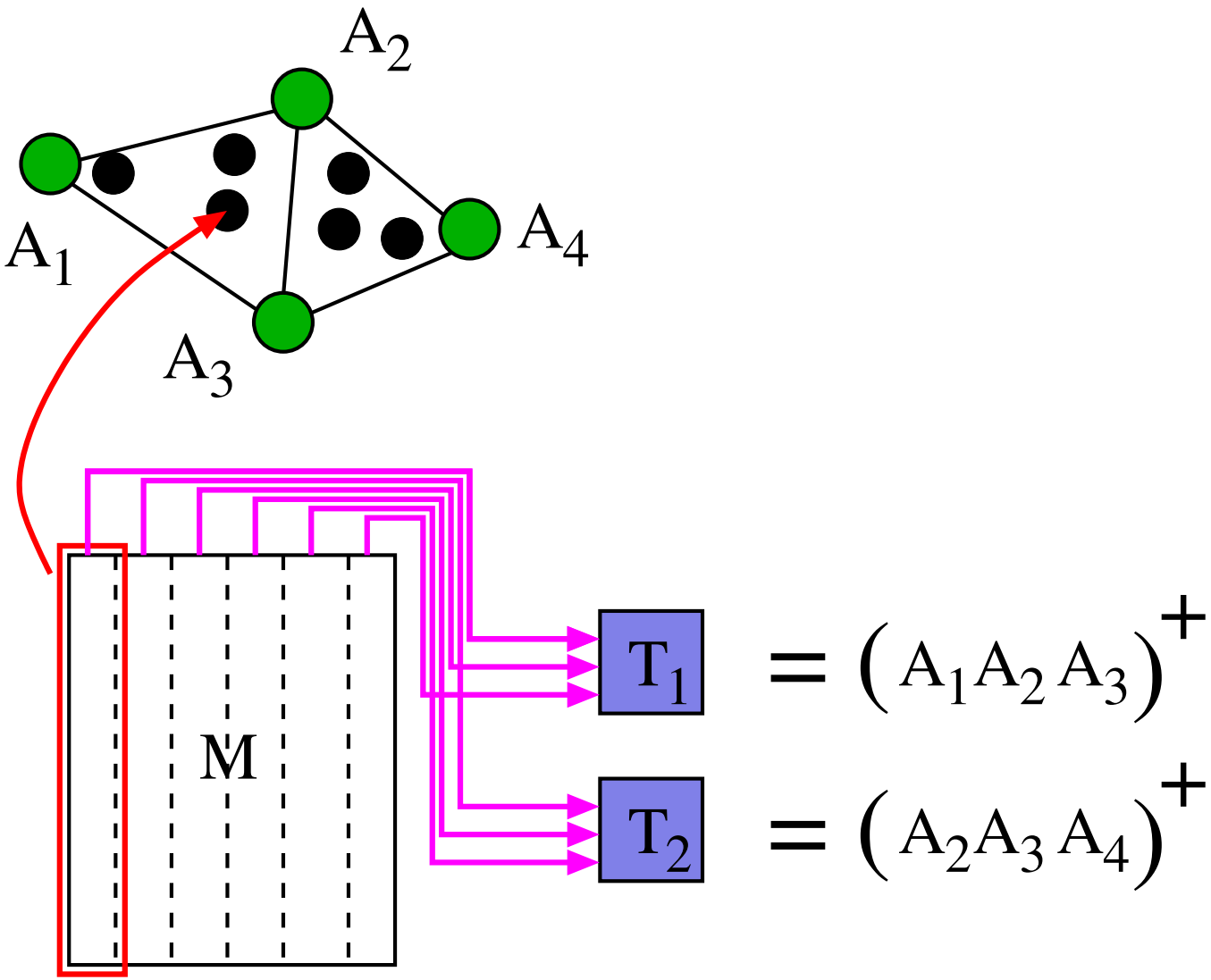
$T_1 = (A_1 A_2 A_3)^+$

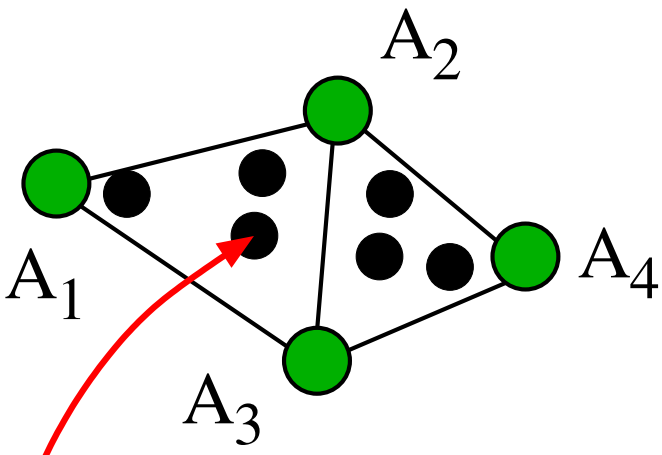
$T_2 = (A_2 A_3 A_4)^+$ ✗



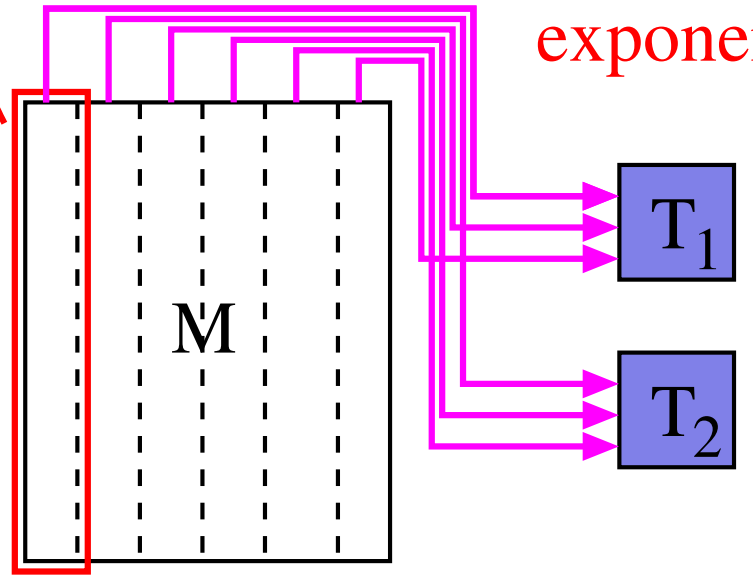
$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$


A2
A1
A4
A3
M
T1
T2
= (A1A2A3)+
= (A2A3A4)+



exponentially many choices?



$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$

General Structure Theorem

Question

What if A does not have full column rank?

Approach: guess many linear transformations, one for each pseudo-inverse of a set of linearly independent columns

General Structure Theorem

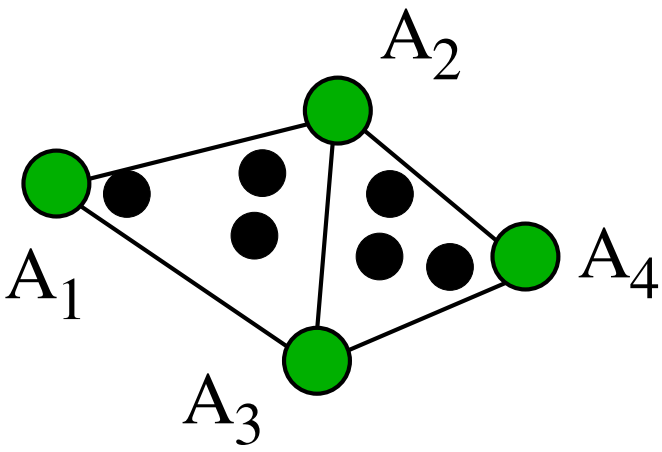
Question

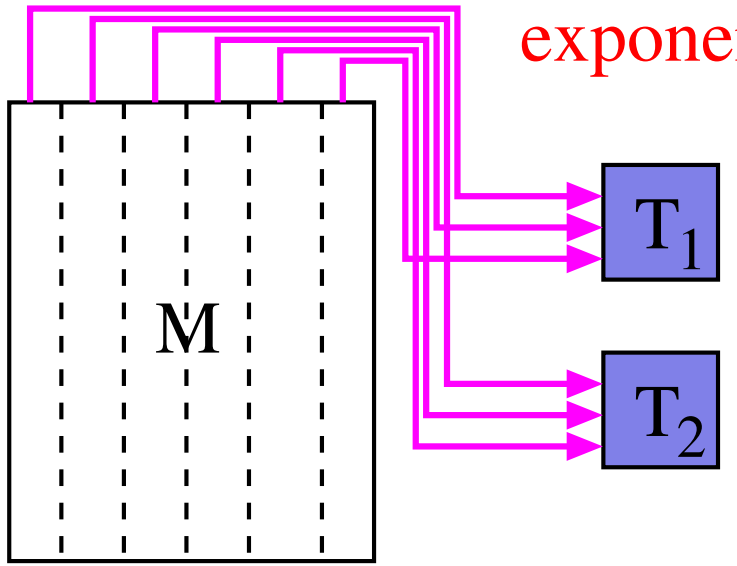
What if A does not have full column rank?

Approach: guess many linear transformations, one for each pseudo-inverse of a set of linearly independent columns

Problem

Which linear transformation should we use (for a column of M)?

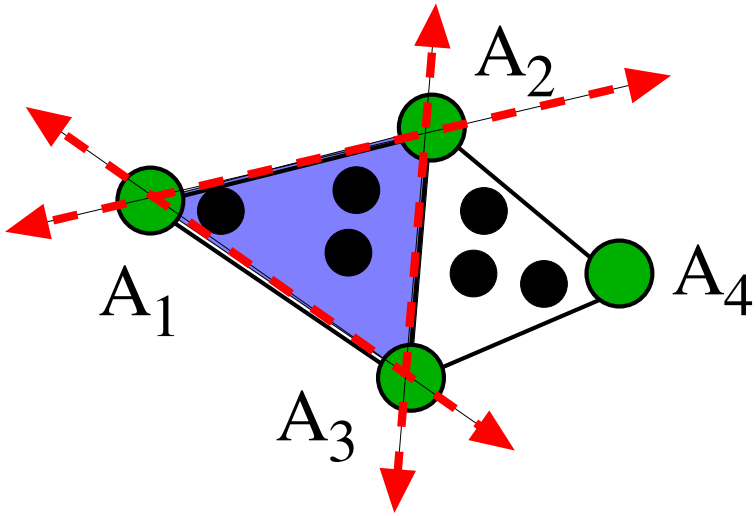


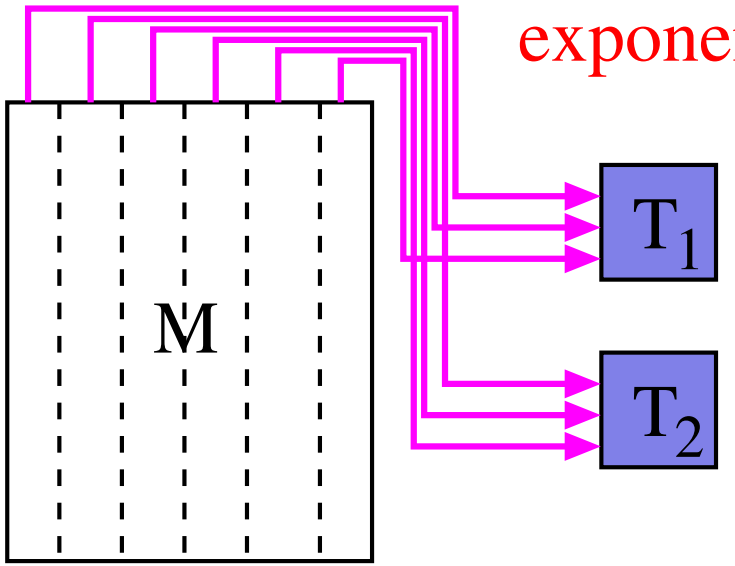


exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$

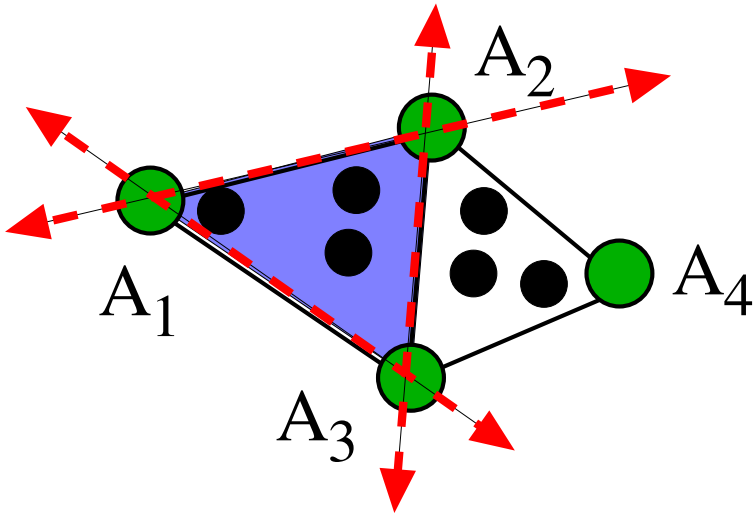


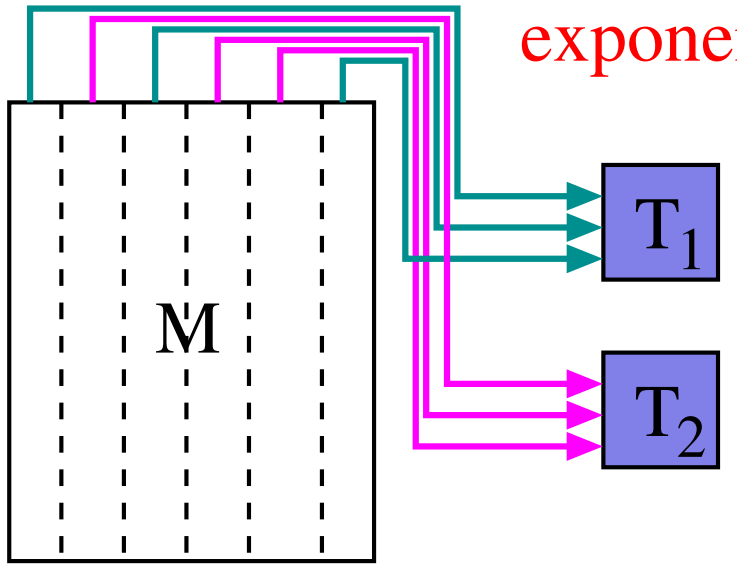


exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$

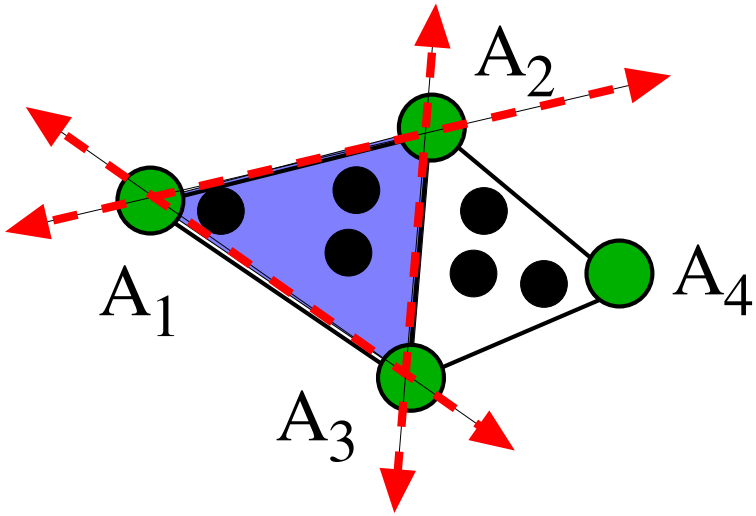


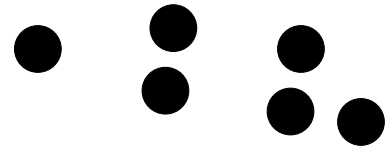


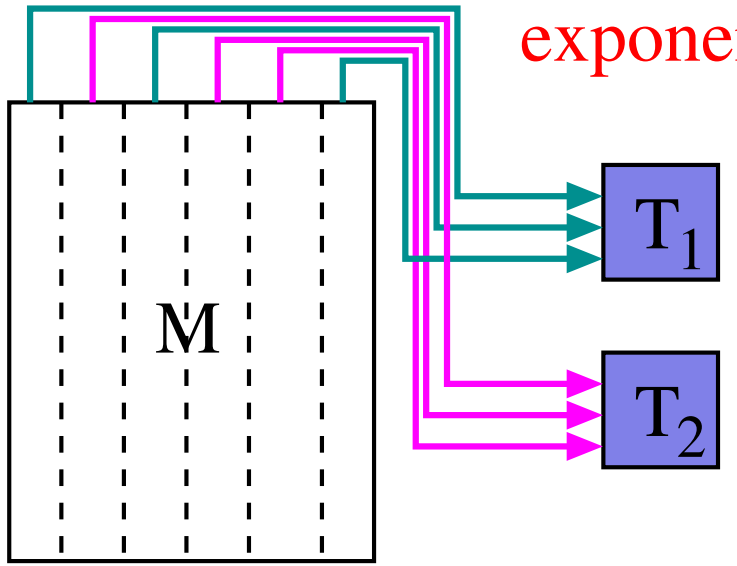
exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



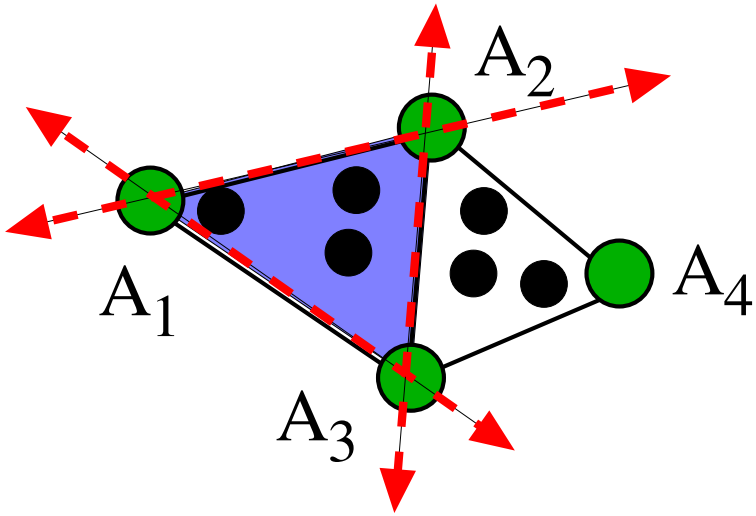


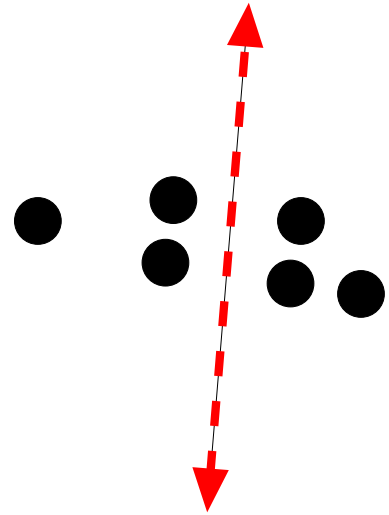


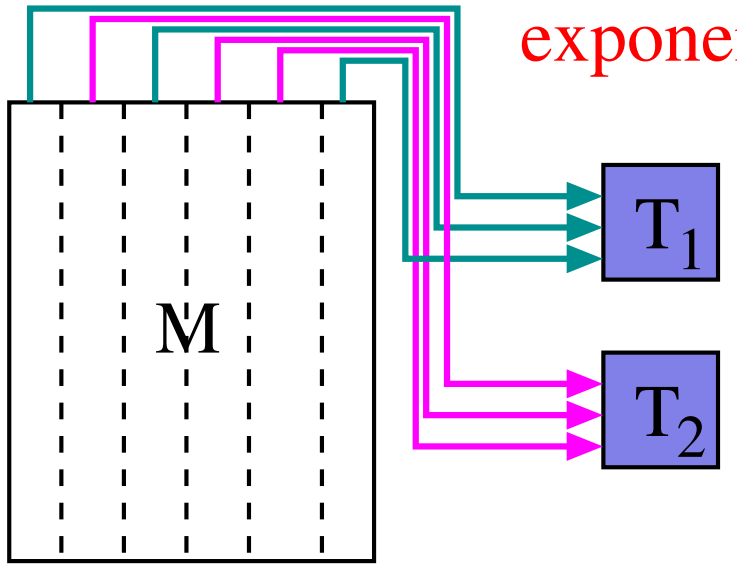
exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



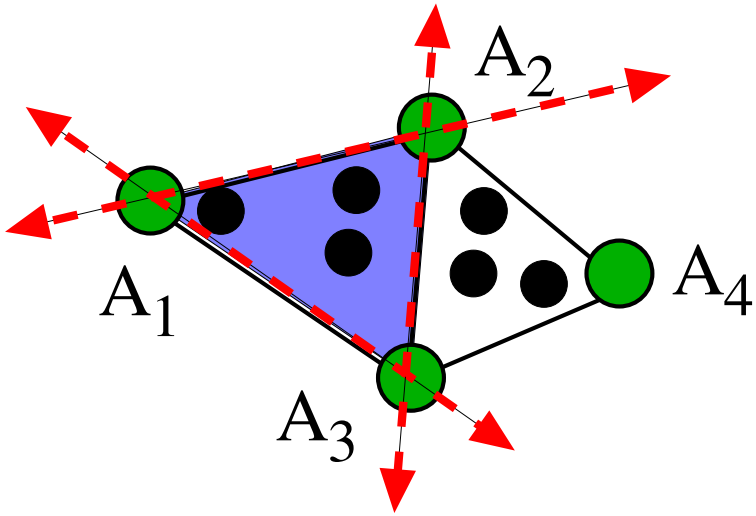


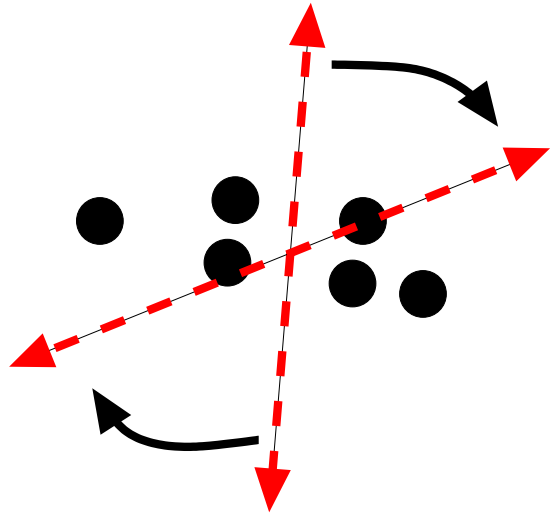


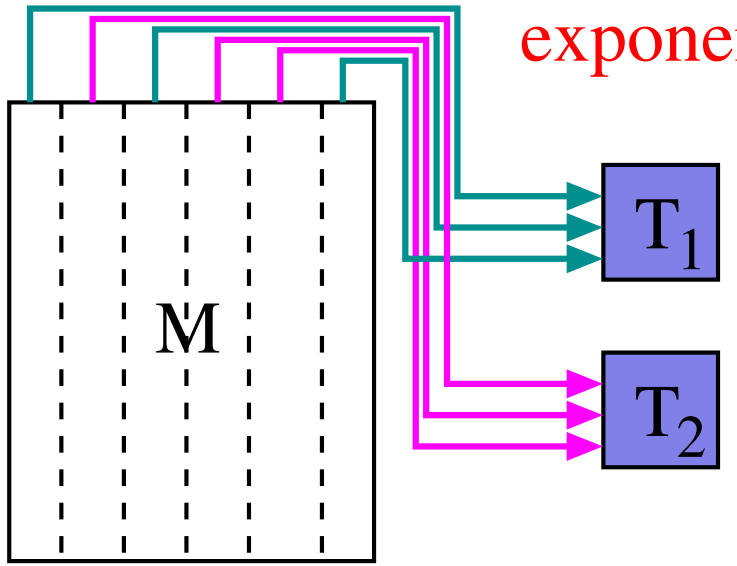
exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$



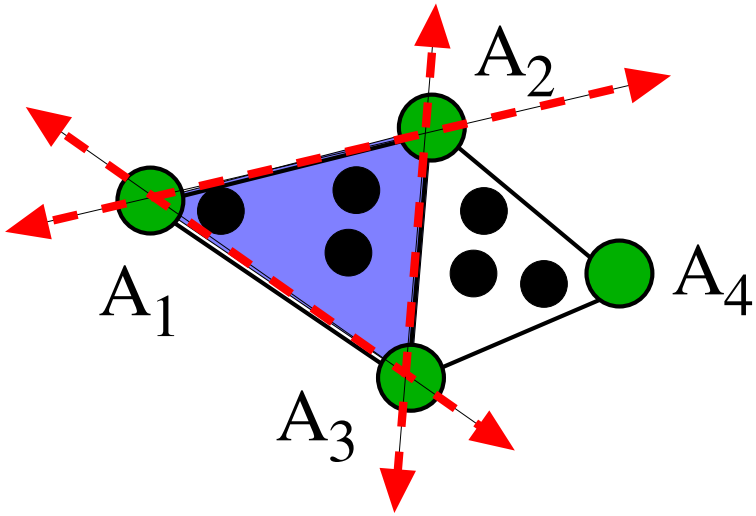


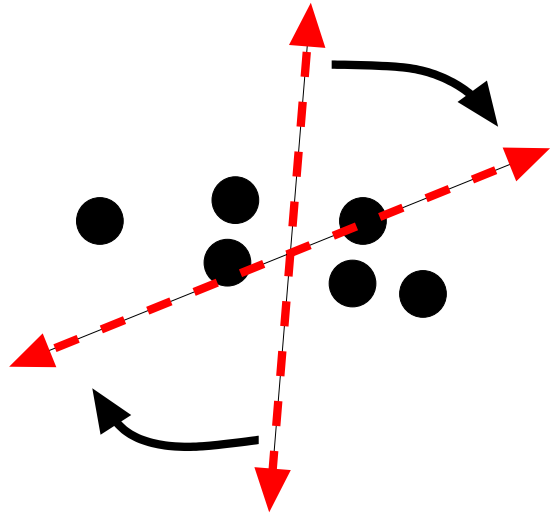


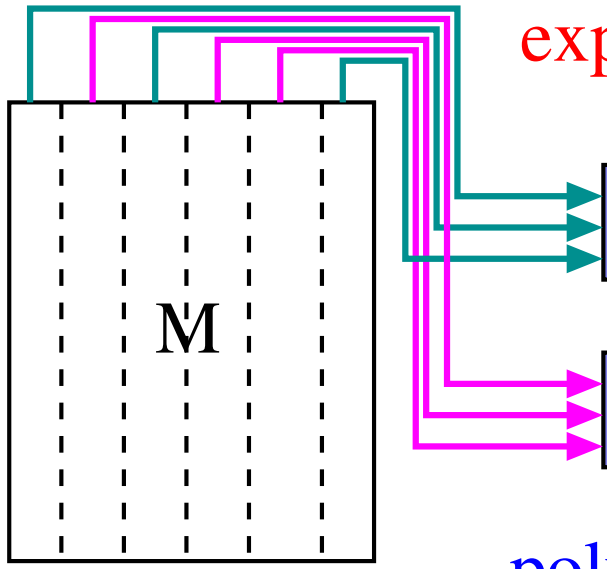
exponentially many choices?

$$T_1 = (A_1 A_2 A_3)^+$$

$$T_2 = (A_2 A_3 A_4)^+$$







exponentially many choices?

$T_1 = (A_1 A_2 A_3)^+$

$T_2 = (A_2 A_3 A_4)^+$

polynomially many choices

General Structure Theorem

Question

What if A does not have full column rank?

Approach: guess many linear transformations, one for each pseudo-inverse of a set of linearly independent columns

Problem

Which linear transformation should we use (for a column of M)?

General Structure Theorem

Question

What if A does not have full column rank?

Approach: guess many linear transformations, one for each pseudo-inverse of a set of linearly independent columns

Problem

Which linear transformation should we use (for a column of M)?

Observation

Which linear transformation depends on a partition of space defined by at most $2^r r^2$ half spaces in r dimensional space

Putting it Together: The General Case

We can brute force search over all such decision rules, and solve a system of polynomial equations for each

Putting it Together: The General Case

We can brute force search over all such decision rules, and solve a system of polynomial equations for each

Claim (Soundness)

If any resulting system has a solution, it yields a nonnegative matrix factorization

Putting it Together: The General Case

We can brute force search over all such decision rules, and solve a system of polynomial equations for each

Claim (Soundness)

If any resulting system has a solution, it yields a nonnegative matrix factorization

Claim (Completeness)

If there is a nonnegative matrix factorization, at least one of the systems will have a solution

Algebraic Interpretation

Theorem (Arora, Ge, Kannan, Moitra)

Given a nonnegative matrix M and a target r there is a one-to-many reduction to a set of $(nm)^{2^r r^2}$ systems of polynomial inequalities each in $2^r r^2$ variables

Algebraic Interpretation

Theorem (Arora, Ge, Kannan, Moitra)

Given a nonnegative matrix M and a target r there is a one-to-many reduction to a set of $(nm)^{2^r r^2}$ systems of polynomial inequalities each in $2^r r^2$ variables

This reduction is (essentially) based on covering the convex hull of the columns of A by 2^r simplices

Algebraic Interpretation

Theorem (Arora, Ge, Kannan, Moitra)

Given a nonnegative matrix M and a target r there is a one-to-many reduction to a set of $(nm)^{2^r r^2}$ systems of polynomial inequalities each in $2^r r^2$ variables

This reduction is (essentially) based on covering the convex hull of the columns of A by 2^r simplices

Is there a more efficient reduction that uses fewer **variables**?

Algebraic Interpretation

Theorem (Arora, Ge, Kannan, Moitra)

Given a nonnegative matrix M and a target r there is a one-to-many reduction to a set of $(nm)^{2^r r^2}$ systems of polynomial inequalities each in $2^r r^2$ variables

This reduction is (essentially) based on covering the convex hull of the columns of A by 2^r simplices

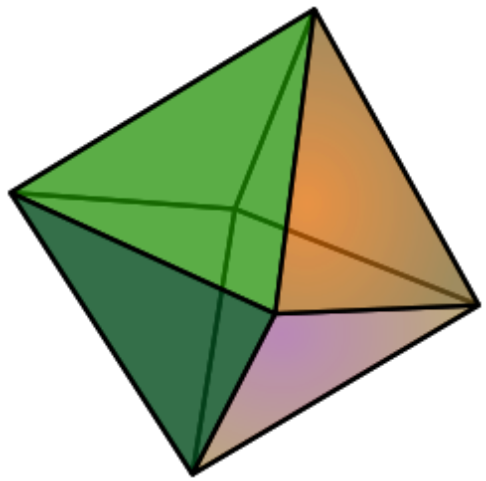
Is there a more efficient reduction that uses fewer **variables**?

Problem

If A does not have full column rank, we may need as many as $\frac{2^r}{r+1}$ simplices to cover it (and only it)

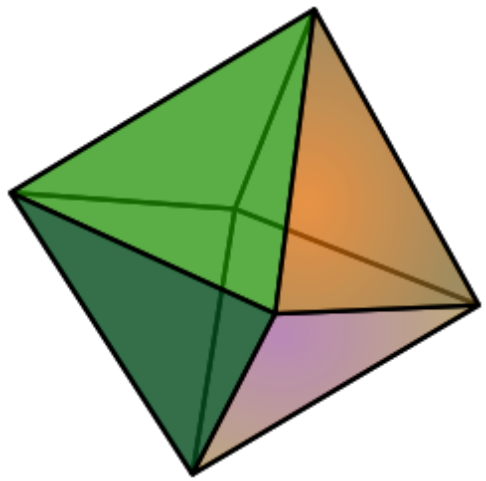
The Cross Polytope

Let $K_r = \{x \text{ s.t. } \sum_{i=1}^{r} |x_i| \leq 1\} = \text{conv}\{\pm e_i\}$ – i.e.



The Cross Polytope

Let $K_r = \{x \text{ s.t. } \sum_{i=1}^{r} |x_i| \leq 1\} = \text{conv}\{\pm e_i\}$ – i.e.



Fact

K_r has $2r$ vertices and 2^r facets (it is dual to the hypercube)

Question

If we want to cover K_r with simplices (but cover nothing else) how many simplices do we need?

Question

If we want to cover K_r with simplices (but cover nothing else) how many simplices do we need?

Each simplex covers at most $r+1$ facets, so we need at least $\frac{2^r}{r+1}$ simples

Question

If we want to cover K_r with simplices (but cover nothing else) how many simplices do we need?

Each simplex covers at most $r+1$ facets, so we need at least $\frac{2^r}{r+1}$ simples

In our setting, this means we really do need **exponentially** many linear transformations to recover the factorization from our input

Question

If we want to cover K_r with simplices (but cover nothing else) how many simplices do we need?

Each simplex covers at most $r+1$ facets, so we need at least $\frac{2^r}{r+1}$ simples

In our setting, this means we really do need **exponentially** many linear transformations to recover the factorization from our input

However these linear transformations are **algebraically dependent**!

Fact (Cramer's Rule)

If R is invertible, the entries of R^{-1} are ratios of polynomials of entries in R

Fact (Cramer's Rule)

If R is invertible, the entries of R^{-1} are ratios of polynomials of entries in R

More precisely, $(R^{-1})_{i,j} = \dfrac{\det(R_{i,j})}{\det(R)}$

Fact (Cramer's Rule)

If R is invertible, the entries of R^{-1} are ratios of polynomials of entries in R

More precisely, $(R^{-1})_{i,j} = \frac{\det(R_{i,j})}{\det(R)}$

Claim

We can express the entries of the 2^r unknown linear transformations as ratios of polynomials in the unknown entries of any invertible submatrix of A and W.

Fact (Cramer's Rule)

If R is invertible, the entries of R^{-1} are ratios of polynomials of entries in R

More precisely, $(R^{-1})_{i,j} = \frac{\det(R_{i,j})}{\det(R)}$

Claim

We can express the entries of the 2^r unknown linear transformations as ratios of polynomials in the unknown entries of any invertible submatrix of A and W.

Thus we only need $2r^2$ variables to express the unknown linear transformations

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

- **[Cohen, Rothblum]**: $mr + nr$

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

- **[Cohen, Rothblum]**: $mr + nr$
- **[Arora, Ge, Kannan, Moitra]**: $2r^2 2^r$

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

- **[Cohen, Rothblum]**: $mr + nr$
- **[Arora, Ge, Kannan, Moitra]**: $2r^2 2^r$
- **[Moitra]**: $2r^2$

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

- **[Cohen, Rothblum]**: $mr + nr$
- **[Arora, Ge, Kannan, Moitra]**: $2r^2 2^r$
- **[Moitra]**: $2r^2$

Corollary

There is an $(2^r nm)^{O(r^2)}$ *time algorithm for NMF*

Summary of Variable Reduction

How many variables do we need to express the decision problem $\text{rank}^+(M) \leq r$?

- [**Cohen, Rothblum**]: $mr + nr$
- [**Arora, Ge, Kannan, Moitra**]: $2r^2 2^r$
- [**Moitra**]: $2r^2$

Corollary

There is an $(2^r nm)^{O(r^2)}$ *time algorithm for NMF*

Are there other applications of variable reduction in geometric problems?

Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - **Intermediate Simplex Problem**
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Other Algorithms?

Question

Are there other algorithms (perhaps non-algebraic) that can solve NMF much faster?

Other Algorithms?

Question

Are there other algorithms (perhaps non-algebraic) that can solve NMF much faster?

Probably not, under standard complexity assumptions

Other Algorithms?

Question

Are there other algorithms (perhaps non-algebraic) that can solve NMF much faster?

Probably not, under standard complexity assumptions

Theorem (Vavasis)

NMF is NP-hard to compute

Other Algorithms?

Question

Are there other algorithms (perhaps non-algebraic) that can solve NMF much faster?

Probably not, under standard complexity assumptions

Theorem (Vavasis)

NMF is NP-hard to compute

Open Question

Is NMF NP-hard when restricted to Boolean matrices?

Intermediate Simplex Problem:

Intermediate Simplex Problem:

- Input: polytopes $P \subseteq Q$ where P is encoded by its vertices and Q is encoded by its facets.

Intermediate Simplex Problem:

- Input: polytopes $P \subseteq Q$ where P is encoded by its vertices and Q is encoded by its facets.
- Is there a simplex K such that $P \subseteq K \subseteq Q$?

Intermediate Simplex Problem:

- Input: polytopes $P \subseteq Q$ where P is encoded by its vertices and Q is encoded by its facets.
- Is there a simplex K such that $P \subseteq K \subseteq Q$?

Theorem (Vavasis)

The intermediate simplex problem and a special case of nonnegative matrix factorization are polynomial time inter-reducible

Intermediate Simplex Problem:

- Input: polytopes $P \subseteq Q$ where P is encoded by its vertices and Q is encoded by its facets.
- Is there a simplex K such that $P \subseteq K \subseteq Q$?

Theorem (Vavasis)

The intermediate simplex problem and a special case of nonnegative matrix factorization are polynomial time inter-reducible

How can we construct geometric gadgets that encode SAT?

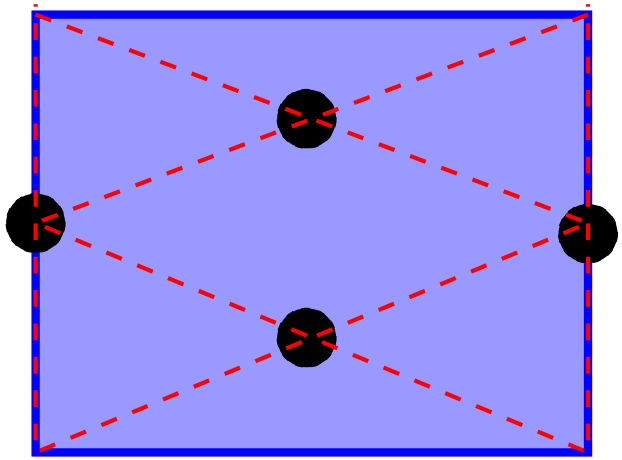
Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

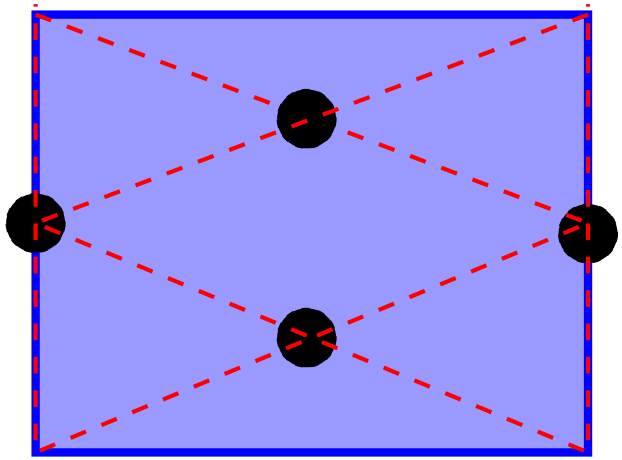
Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - **Geometric Gadgets**
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Vavasis:



Vavasis:



place three points!

Each gadget represents a variable and forces it to be **true** or **false**

Each gadget represents a variable and forces it to be **true** or **false**

But this approach requires r to be large (at least the number of variables in a SAT formula).

Each gadget represents a variable and forces it to be **true** or **false**

But this approach requires r to be large (at least the number of variables in a *SAT* formula).

Question

Are there lower dimensional gadgets that can be used to encode hard problems?

Each gadget represents a variable and forces it to be **true** or **false**

But this approach requires r to be large (at least the number of variables in a *SAT* formula).

Question

Are there lower dimensional gadgets that can be used to encode hard problems?

d-SUM Problem:

Each gadget represents a variable and forces it to be **true** or **false**

But this approach requires r to be large (at least the number of variables in a *SAT* formula).

Question

Are there lower dimensional gadgets that can be used to encode hard problems?

d-SUM Problem:

- Input: n numbers $s_1, s_2, \ldots s_n \in \mathbb{R}$

Each gadget represents a variable and forces it to be **true** or **false**

But this approach requires r to be large (at least the number of variables in a SAT formula).

Question

Are there lower dimensional gadgets that can be used to encode hard problems?

d-SUM Problem:

- Input: n numbers $s_1, s_2, \ldots s_n \in \mathbb{R}$
- Is there a set of d distinct numbers $U \subset [n]$ where $\sum_{i \in U} s_i = 0$?

Hardness for d-SUM

The best known algorithms for d-SUM run in time $n^{\lceil d/2 \rceil}$

Hardness for d-SUM

The best known algorithms for d-SUM run in time $n^{\lceil d/2 \rceil}$

Theorem (Patrascu, Williams)

If d-SUM can be solved in time $n^{o(d)}$ then there are subexponential time algorithms for 3-SAT.

Hardness for d-SUM

The best known algorithms for d-SUM run in time $n^{\lceil d/2 \rceil}$

Theorem (Patrascu, Williams)

If d-SUM can be solved in time $n^{o(d)}$ then there are subexponential time algorithms for 3-SAT.

Conjecture (Impagliazzo, Paturi)

3-SAT on n variables cannot be solved in time $2^{o(n)}$

Hardness for d-SUM

The best known algorithms for d-SUM run in time $n^{\lceil d/2 \rceil}$

Theorem (Patrascu, Williams)

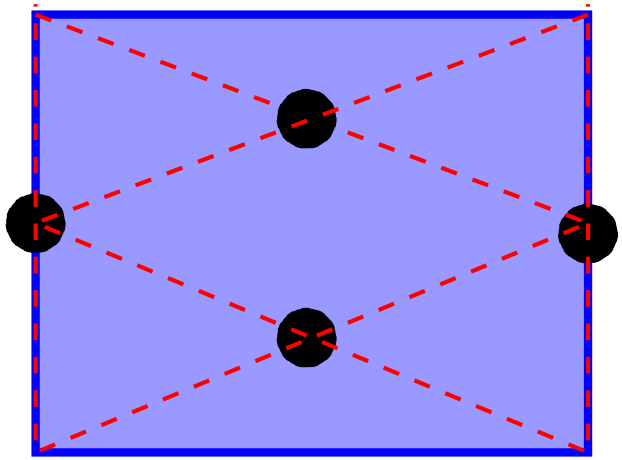
If d-SUM can be solved in time $n^{o(d)}$ then there are subexponential time algorithms for 3-SAT.

Conjecture (Impagliazzo, Paturi)

3-SAT on n variables cannot be solved in time $2^{o(n)}$

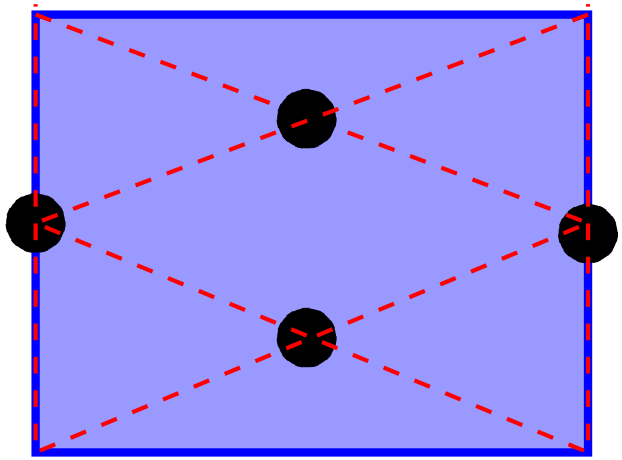
Can we use d-SUM as our source of hardness?

Vavasis:



place three points!

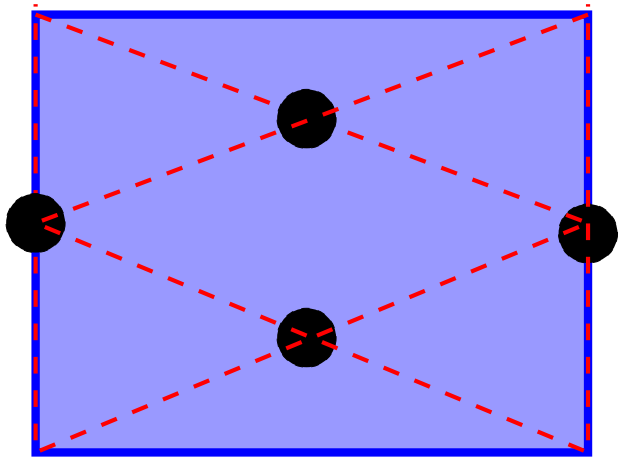
Vavasis:



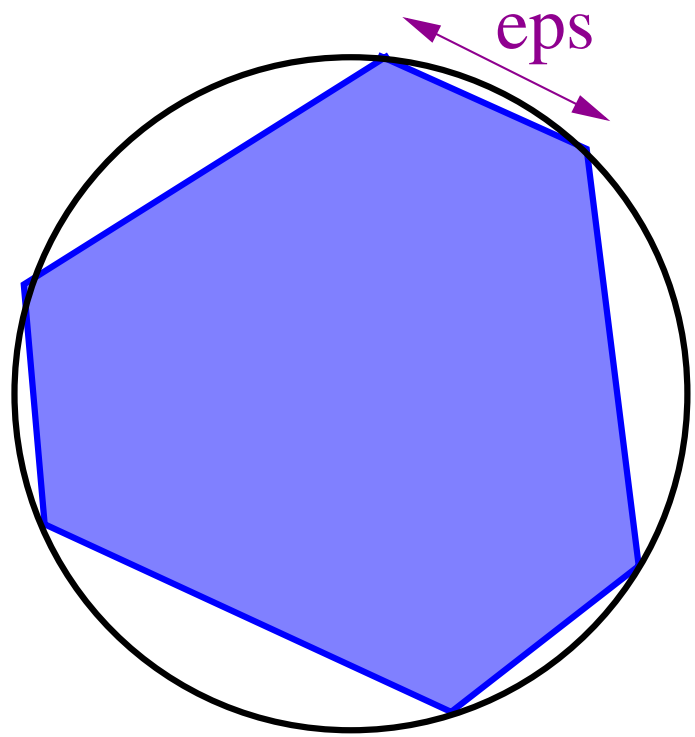
place three points!

d–SUM gadget:

Vavasis:

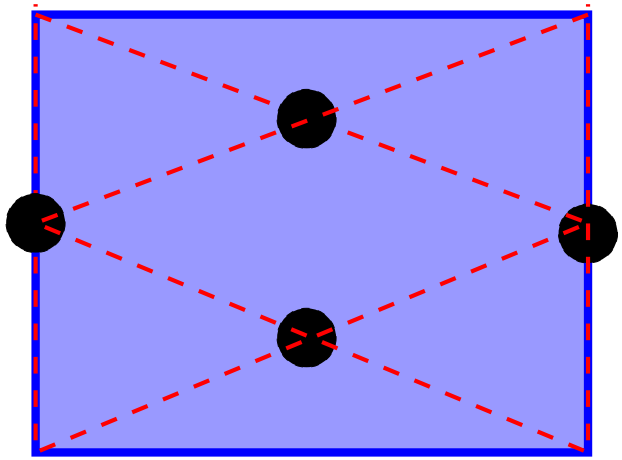


d–SUM gadget:

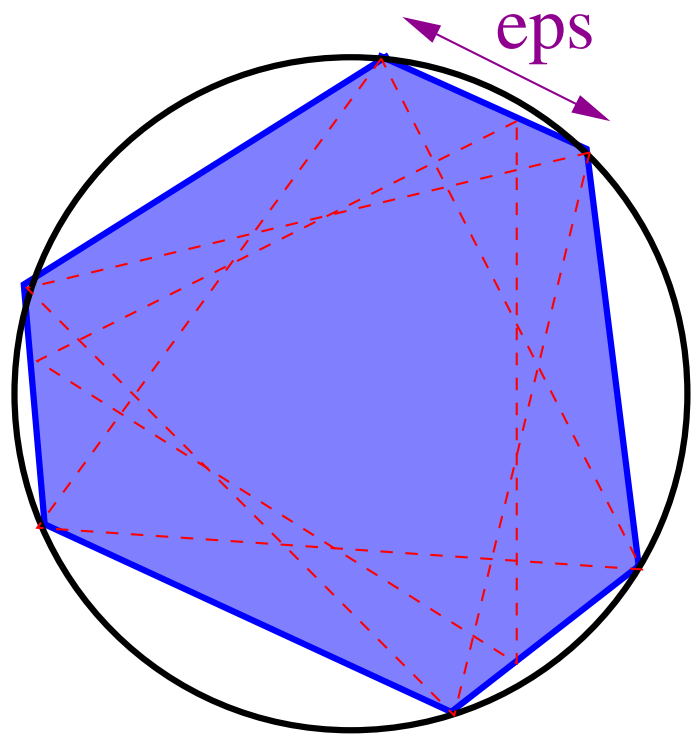


place three points!

Vavasis:

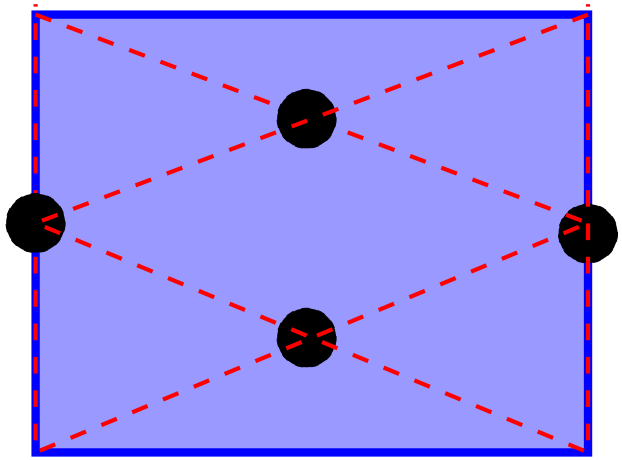


d–SUM gadget:

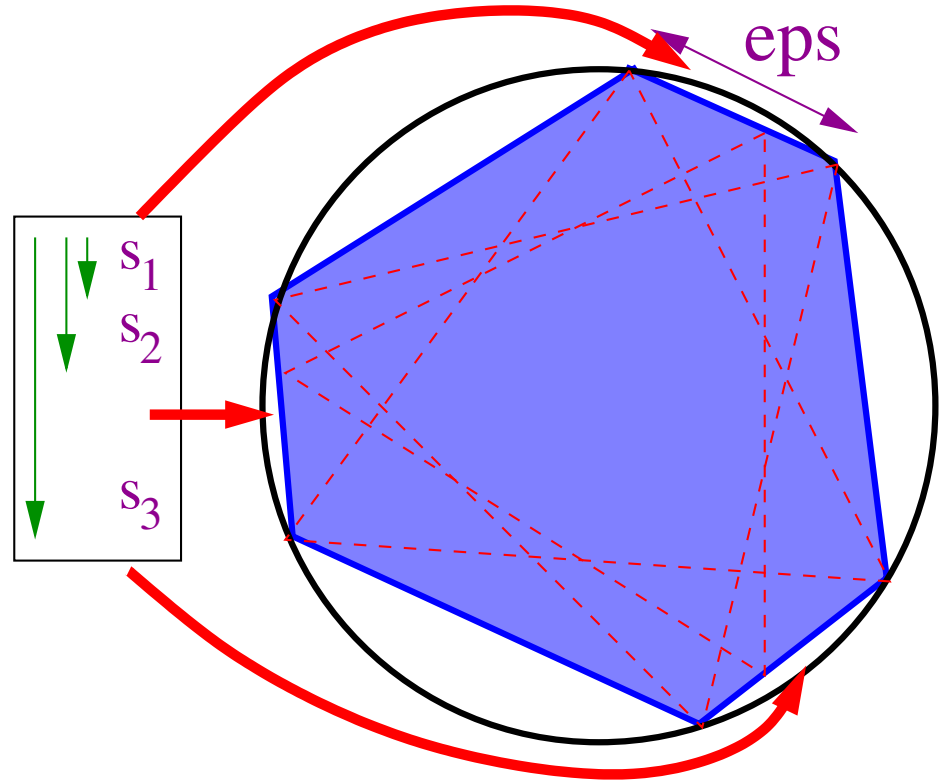


place three points!

Vavasis:

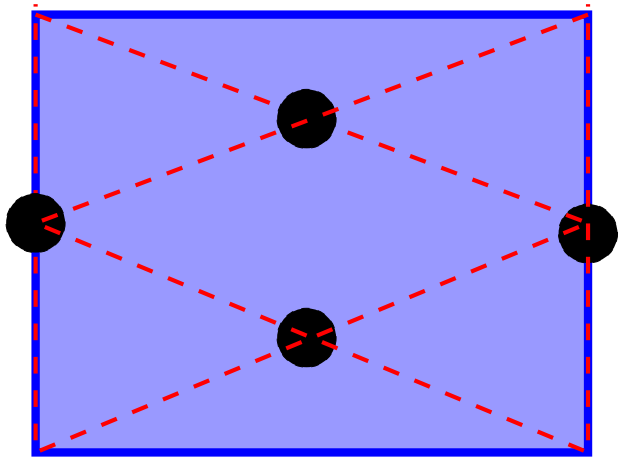


d–SUM gadget:

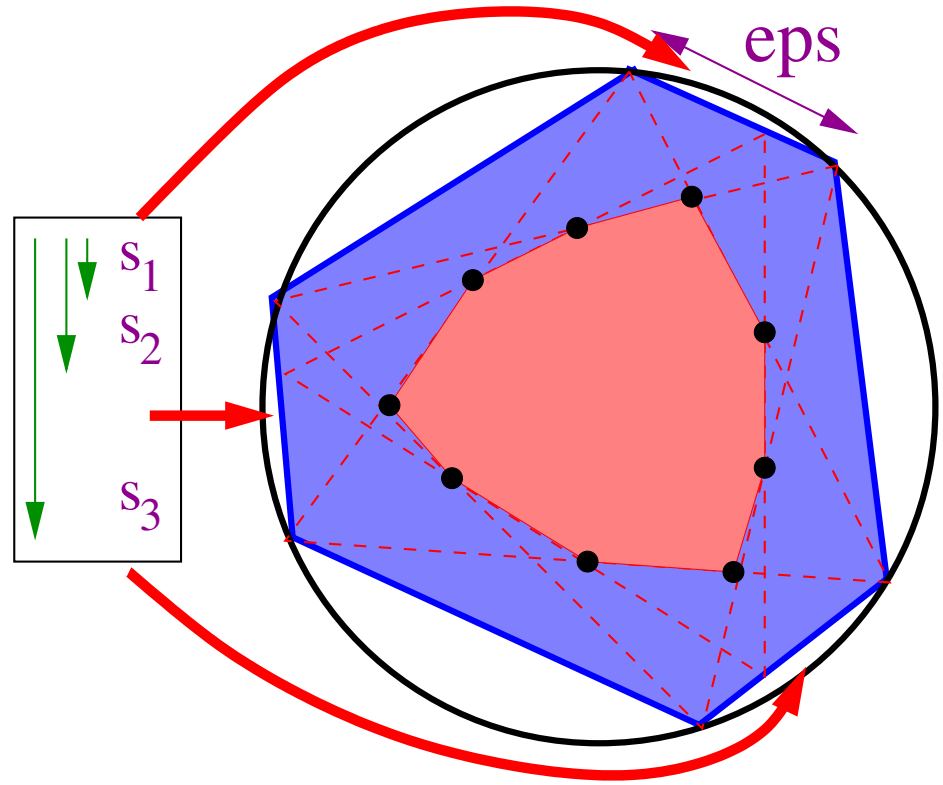


place three points!

Vavasis:



d–SUM gadget:



place three points!

Summary of Hardness Results

Theorem (Vavasis)

NMF is NP-hard

Summary of Hardness Results

Theorem (Vavasis)

NMF is NP-hard

Theorem (Arora, Ge, Kannan, Moitra)

An algorithm for NMF that runs in time $(nm)^{o(r)}$ *would yield a sub-exponential time algorithm for 3-SAT*

Summary of Hardness Results

Theorem (Vavasis)

NMF is NP-hard

Theorem (Arora, Ge, Kannan, Moitra)

An algorithm for NMF that runs in time $(nm)^{o(r)}$ *would yield a sub-exponential time algorithm for 3-SAT*

Open Question

Is it hard to solve NMF in time $(nm)^{o(r^2)}$*?*

Beyond Worst-Case Analysis

We (essentially) resolved the worst-case complexity of NMF, but there are still paths to obtaining better algorithms

Beyond Worst-Case Analysis

We (essentially) resolved the worst-case complexity of NMF, but there are still paths to obtaining better algorithms

Question

What distinguishes a realistic instance of NMF from an artificial one?

Beyond Worst-Case Analysis

We (essentially) resolved the worst-case complexity of NMF, but there are still paths to obtaining better algorithms

Question

What distinguishes a realistic instance of NMF from an artificial one?

Question

Are there practical algorithms with provable guarantees?

Beyond Worst-Case Analysis

We (essentially) resolved the worst-case complexity of NMF, but there are still paths to obtaining better algorithms

Question

What distinguishes a realistic instance of NMF from an artificial one?

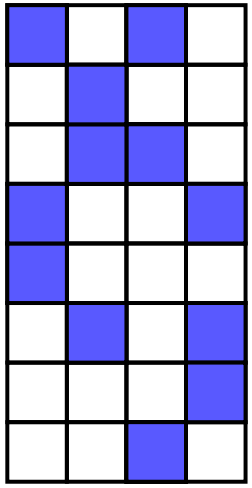
Question

Are there practical algorithms with provable guarantees?

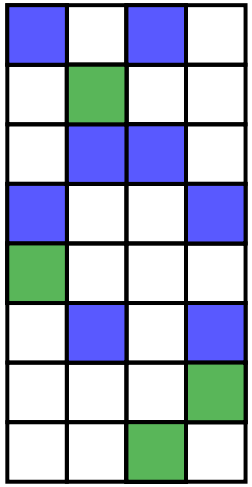
Recall, in some applications columns in A represent distributions on words (topics) and columns in W represent distributions on topics

[Donoho, Stodden]: A is separable if each column i has an unknown row $\pi(i)$ whose only non-zero is in column i

A



A



[Donoho, Stodden]: A is separable if each column i has an unknown row $\pi(i)$ whose only non-zero is in column i

[**Donoho, Stodden**]: A is separable if each column i has an unknown row $\pi(i)$ whose only non-zero is in column i

Interpretation: We call this row an **anchor word**, and any document that contains this word is very likely to be (at least partially) about the corresponding topic

[**Donoho, Stodden**]: A is separable if each column i has an unknown row $\pi(i)$ whose only non-zero is in column i

Interpretation: We call this row an **anchor word**, and any document that contains this word is very likely to be (at least partially) about the corresponding topic

e.g. personal finance $\leftarrow 401k$, baseball $\leftarrow$ bunt, ...

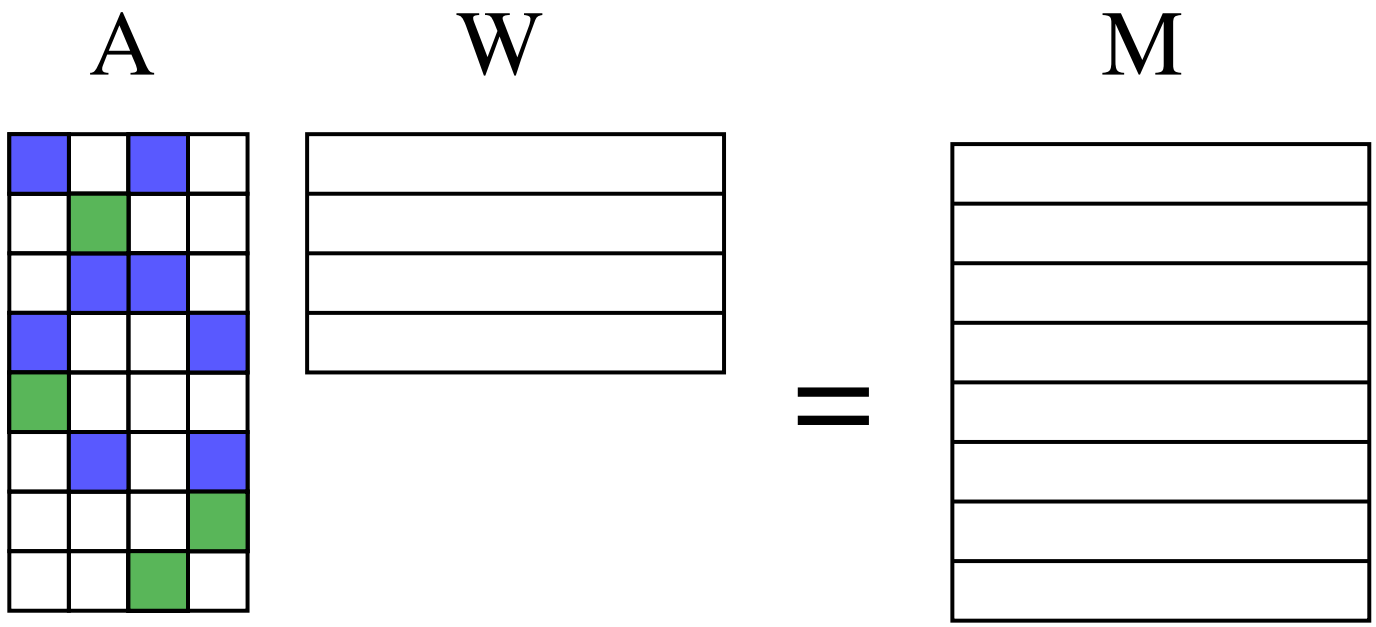
[**Donoho, Stodden**]: A is separable if each column i has an unknown row $\pi(i)$ whose only non-zero is in column i

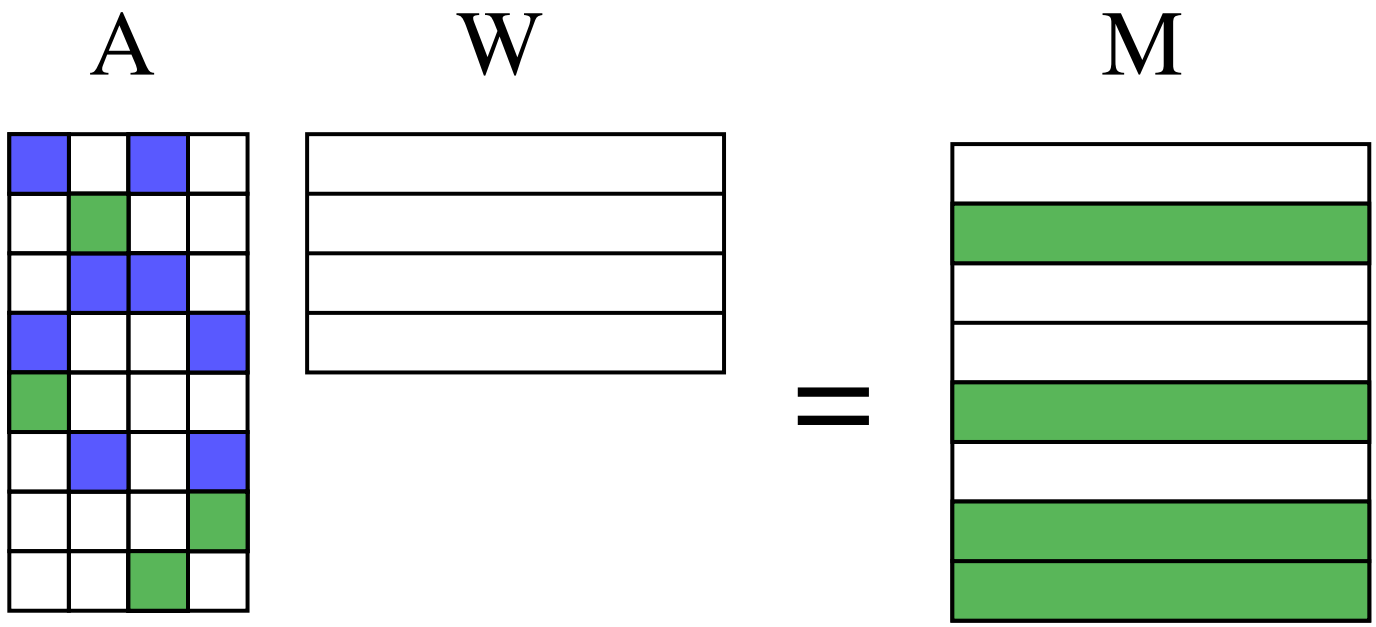
Interpretation: We call this row an **anchor word**, and any document that contains this word is very likely to be (at least partially) about the corresponding topic

e.g. personal finance $\leftarrow 401k$, baseball $\leftarrow$ bunt, ...

Question

Separability was introduced to understand when NMF is unique – Is it enough to make NMF easy?


A
W
M
=


A
W
M
=

Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Observation

Rows of W appear as (scaled) rows of M

Separable Instances

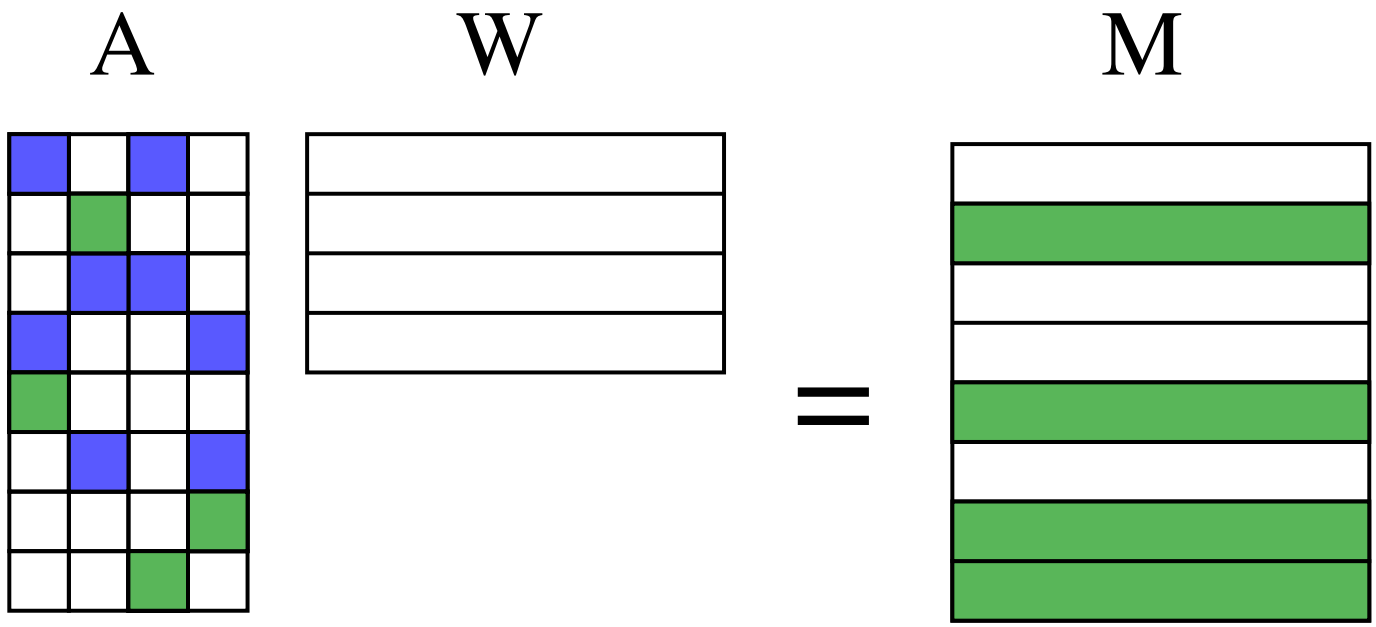
Recall: For each topic, there is some (anchor) word that only appears in this topic

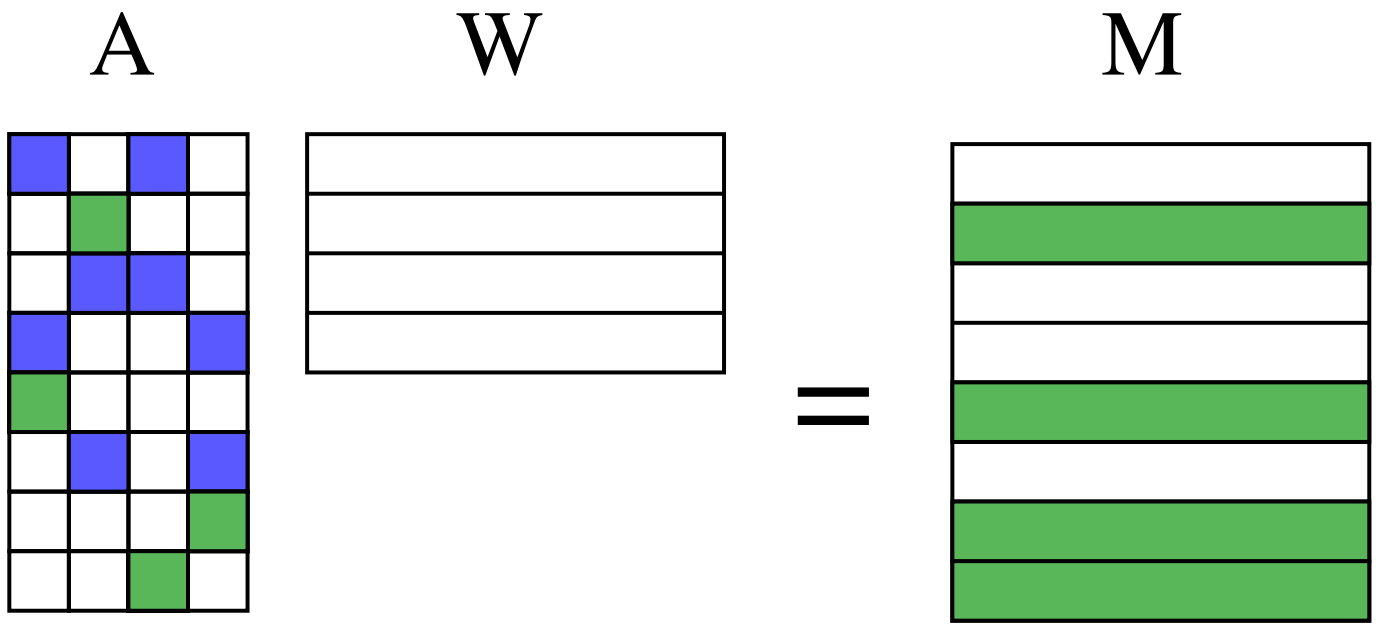
Observation

Rows of W appear as (scaled) rows of M

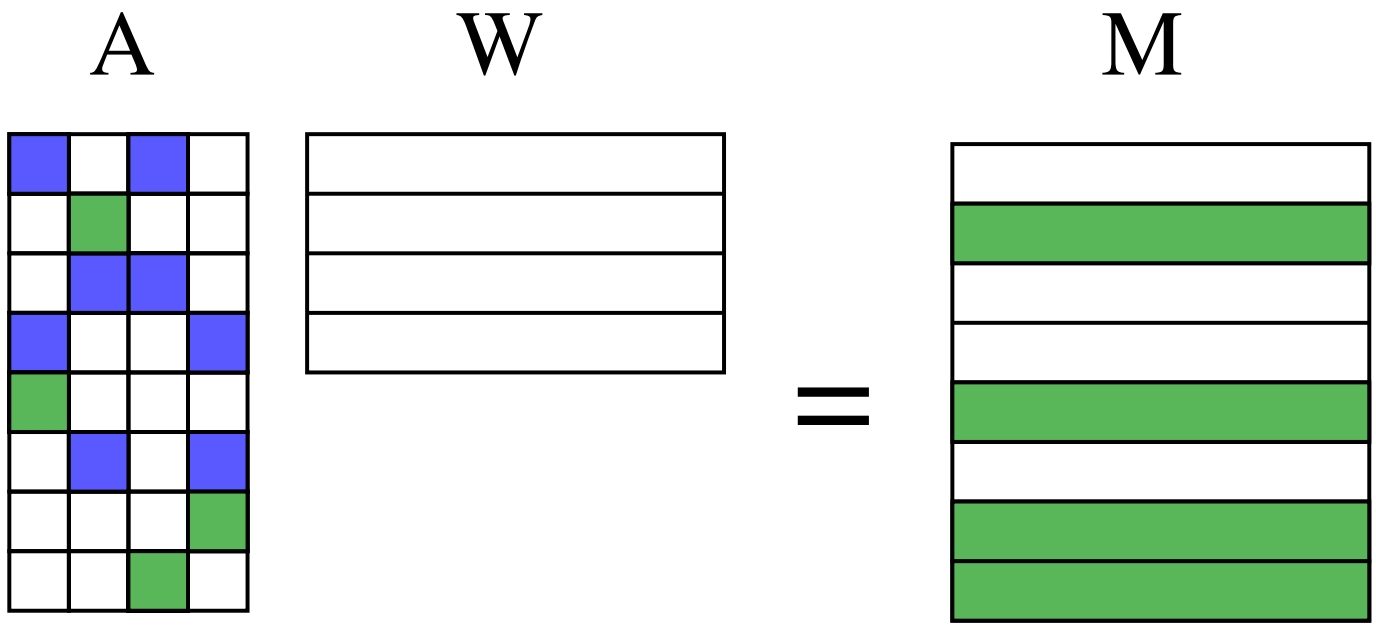
Question

How can we identify anchor words?

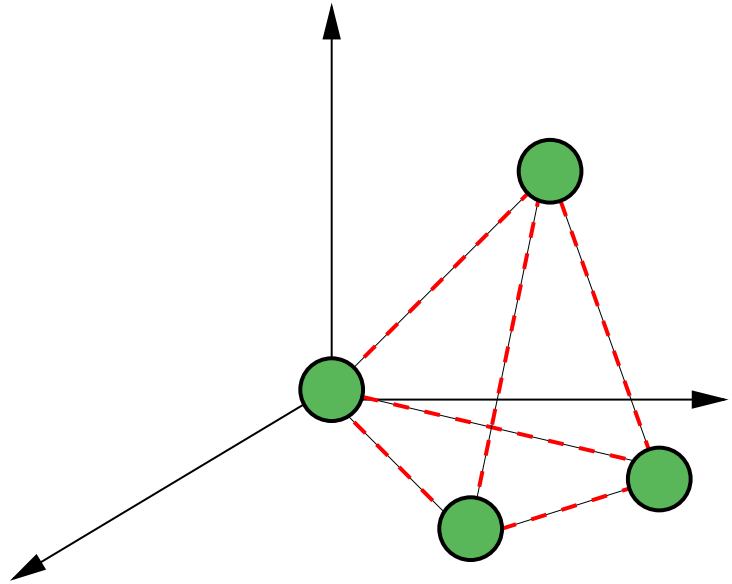

A
W
M
=

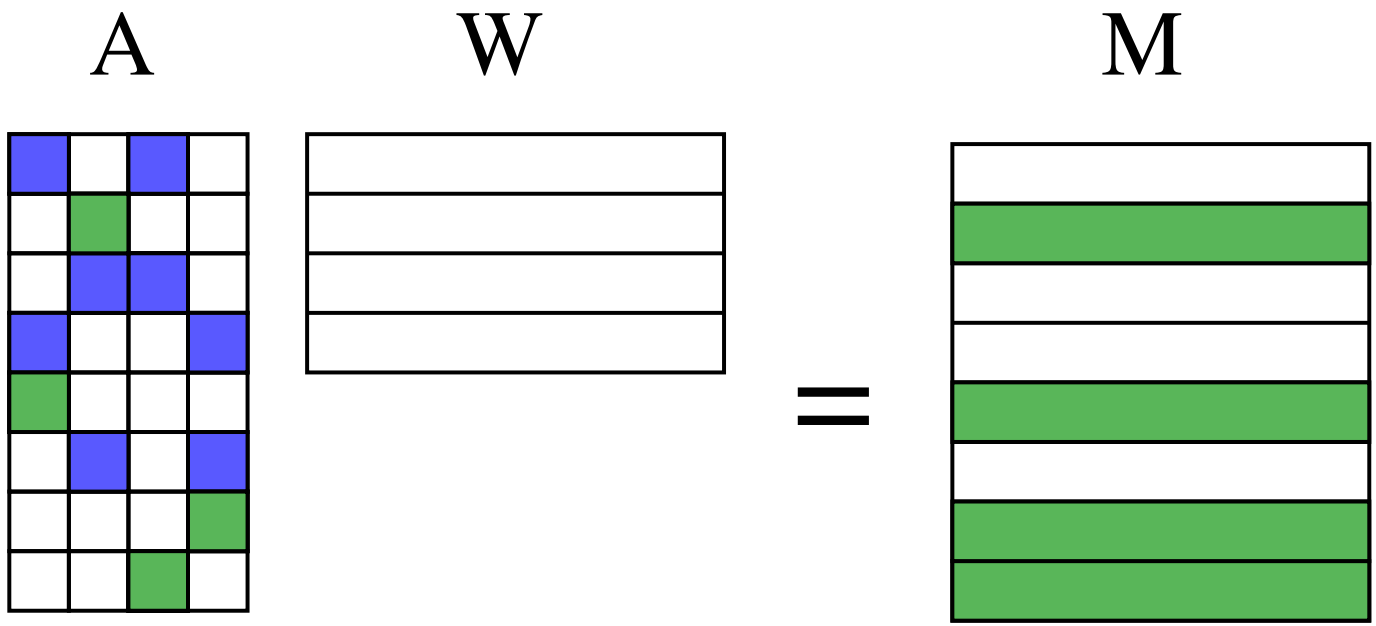


brute force: n^r

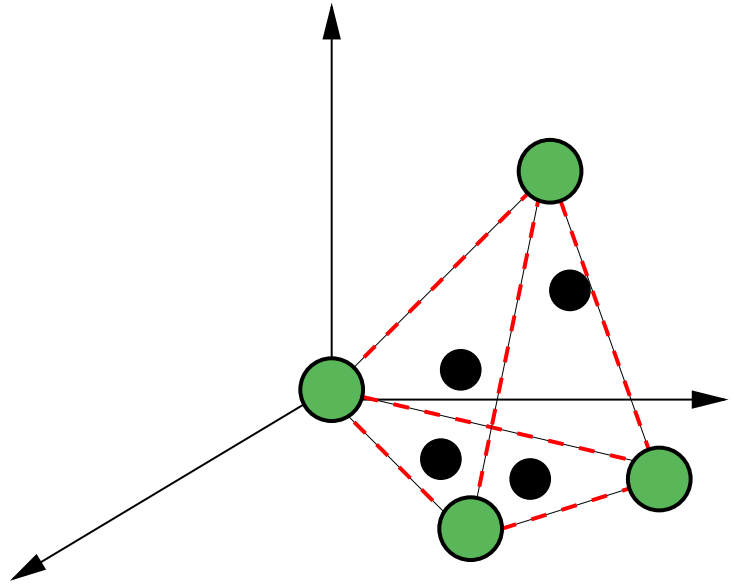


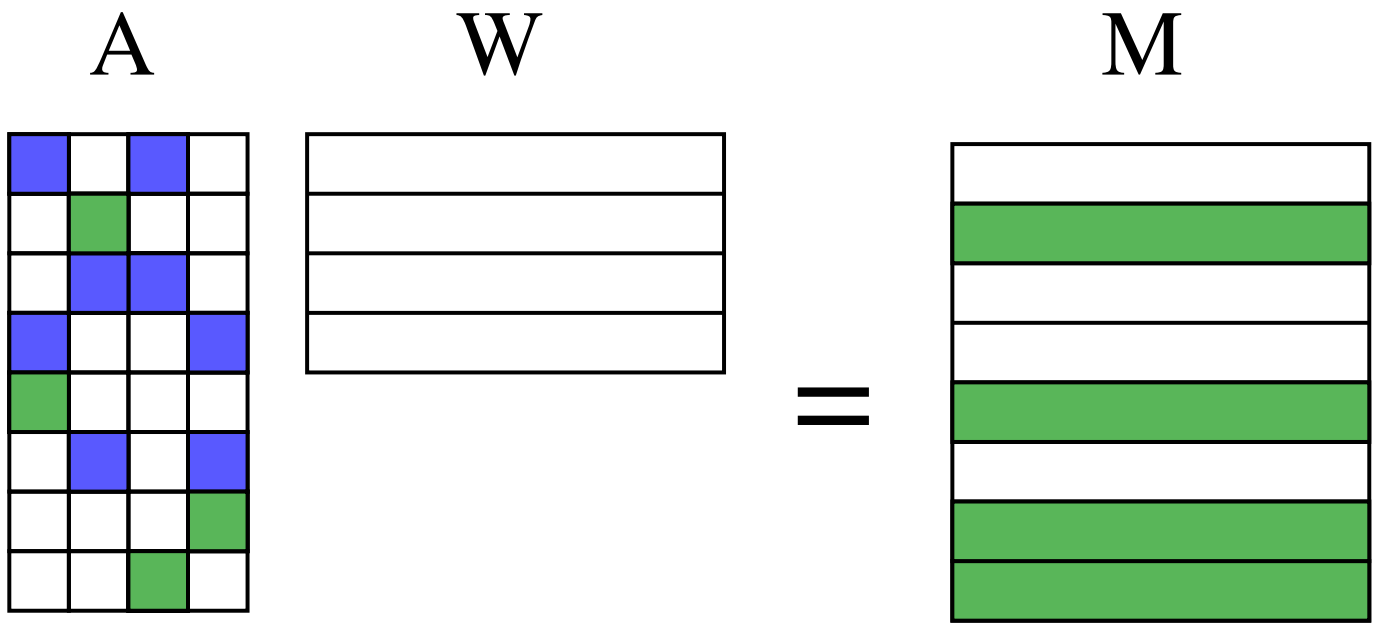
brute force: n^r



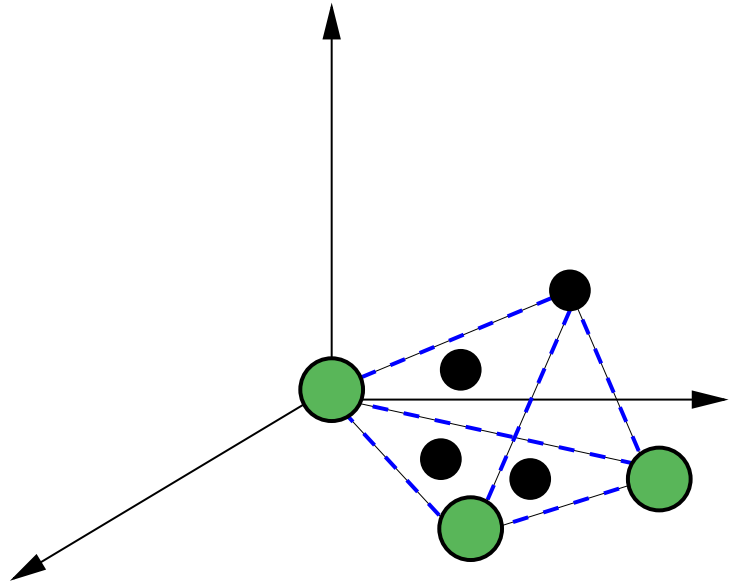


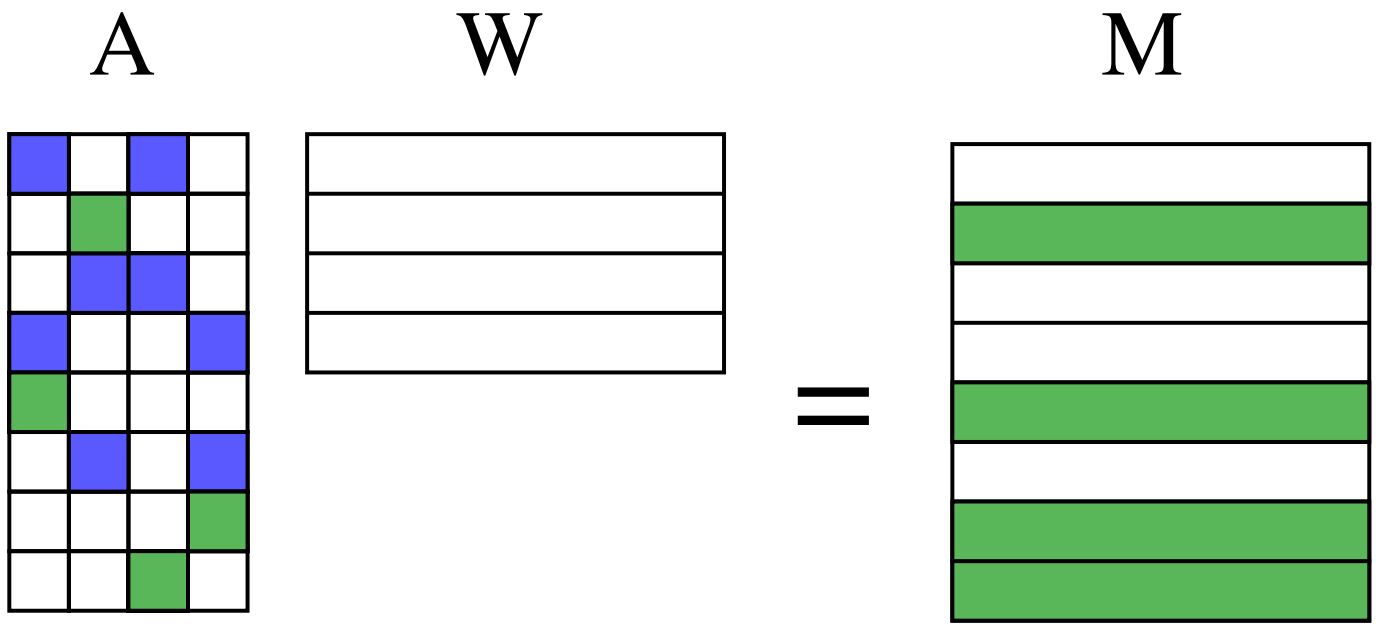
brute force: n^r



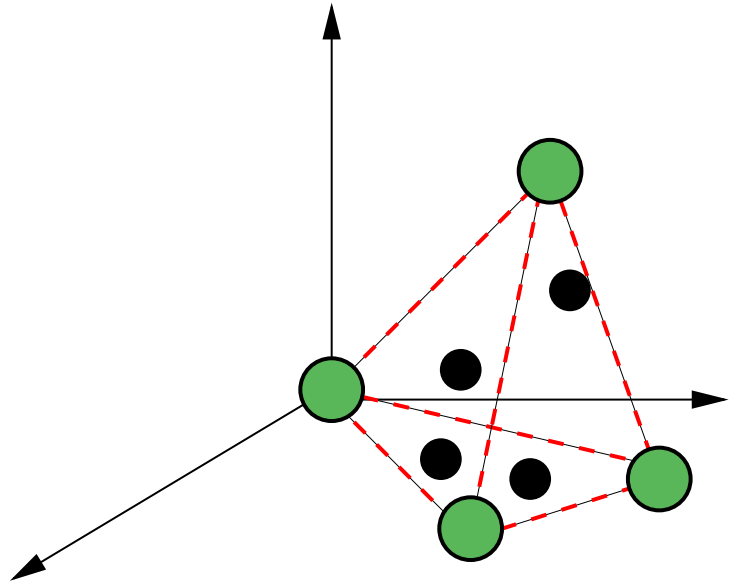


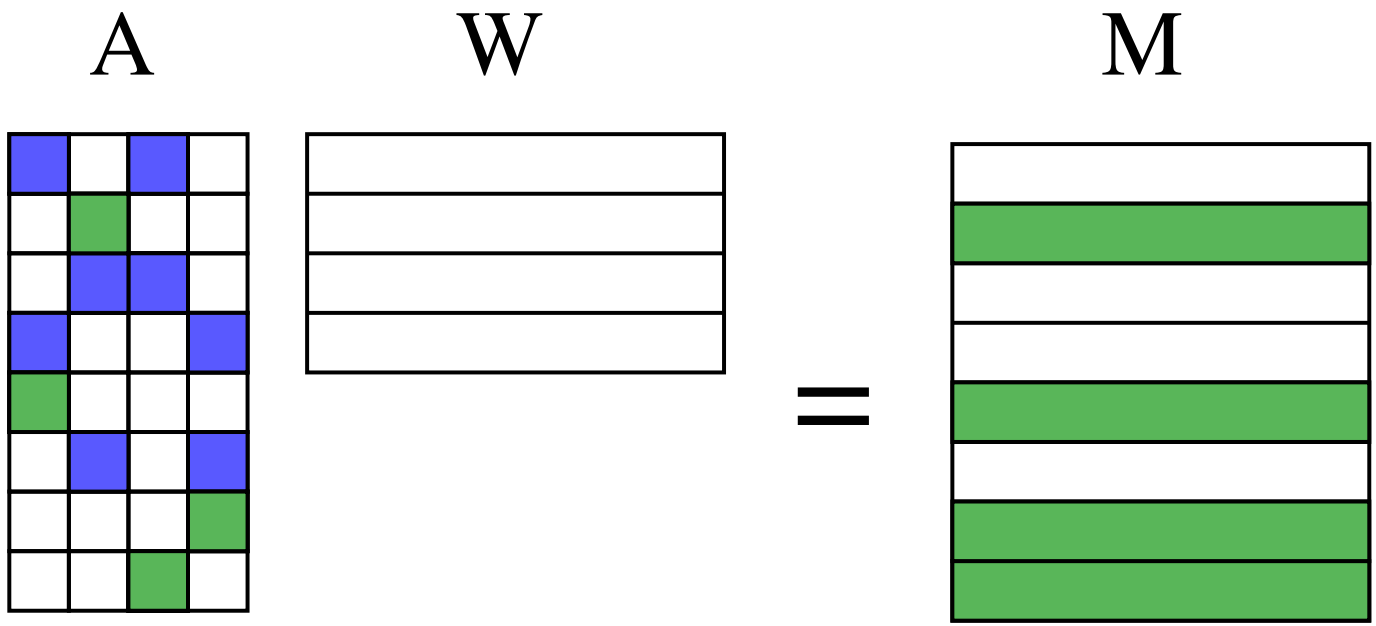
brute force: n^r



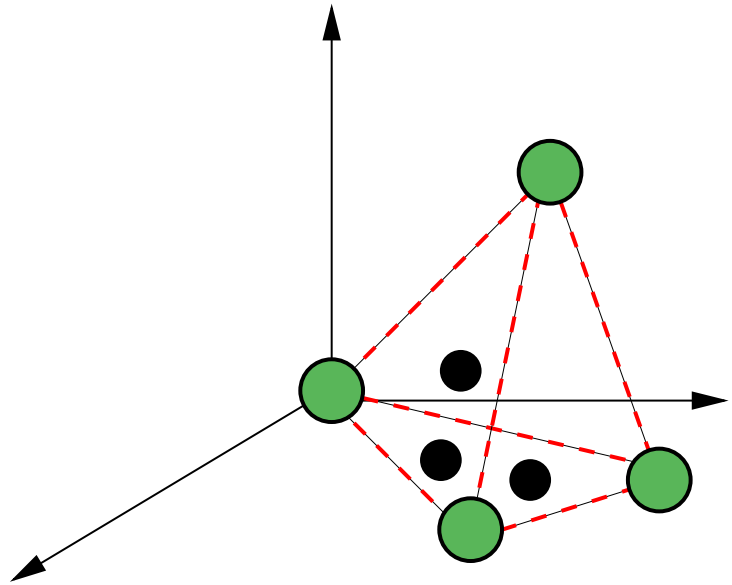


brute force: n^r





brute force: n^r



Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Observation

Rows of W appear as (scaled) rows of M

Question

How can we identify anchor words?

Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Observation

Rows of W appear as (scaled) rows of M

Question

How can we identify anchor words?

Removing a row from M strictly changes the convex hull iff it is an anchor word

Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Observation

Rows of W appear as (scaled) rows of M

Question

How can we identify anchor words?

Removing a row from M strictly changes the convex hull iff it is an anchor word

Hence we can identify all the anchor words via linear programming

Separable Instances

Recall: For each topic, there is some (anchor) word that only appears in this topic

Observation

Rows of W appear as (scaled) rows of M

Question

How can we identify anchor words?

Removing a row from M strictly changes the convex hull iff it is an anchor word

Hence we can identify all the anchor words via linear programming (can be made robust to **noise**)

Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - Applications to Topic Modeling

Outline

- Introduction
- Algebraic Algorithms for NMF
  - Systems of Polynomial Inequalities
  - Methods for Variable Reduction
- Hardness Results
  - Intermediate Simplex Problem
  - Geometric Gadgets
- Separable NMF
  - The Anchor Words Algorithm
  - **Applications to Topic Modeling**

Topic Models

- Topic matrix A, generate W stochastically

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words
- e.g. Latent Dirichlet Allocation (LDA), Correlated Topic Model (CTM), Pachinko Allocation Model (PAM), ...

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words
- e.g. Latent Dirichlet Allocation (LDA), Correlated Topic Model (CTM), Pachinko Allocation Model (PAM), ...

Question

Can we estimate A, given random samples from M?

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words
- e.g. Latent Dirichlet Allocation (LDA), Correlated Topic Model (CTM), Pachinko Allocation Model (PAM), ...

Question

Can we estimate A, given random samples from M?

Yes! **[Arora, Ge, Moitra]** we give a provable algorithm based on (noise-tolerant) NMF;

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words
- e.g. Latent Dirichlet Allocation (LDA), Correlated Topic Model (CTM), Pachinko Allocation Model (PAM), ...

Question

Can we estimate A, given random samples from M?

Yes! **[Arora, Ge, Moitra]** we give a provable algorithm based on (noise-tolerant) NMF; see also **[Anandkumar et al]**

Topic Models

- Topic matrix A, generate W stochastically
- For each document (column in $M = AW$) sample N words
- e.g. Latent Dirichlet Allocation (LDA), Correlated Topic Model (CTM), Pachinko Allocation Model (PAM), ...

Question

Can we estimate A, given random samples from M?

Yes! **[Arora, Ge, Moitra]** we give a provable algorithm based on (noise-tolerant) NMF; see also **[Anandkumar et al]**

Danger: I am about to show you experimental results

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

- Run time: 12 minutes

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

- Run time: 12 minutes (compared to 10 hours for MALLET and other state-of-the-art topic modeling tools)

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

- Run time: 12 minutes (compared to 10 hours for MALLET and other state-of-the-art topic modeling tools)
- Topics are **high quality**

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

- Run time: 12 minutes (compared to 10 hours for MALLET and other state-of-the-art topic modeling tools)
- Topics are **high quality** (because we can handle correlations in the topic model?)

Running on Real Data

joint work with Arora, Ge, Halpern, Mimno, Sontag, Wu and Zhu

We ran our algorithm on a database of 300,000 New York Times articles (from the UCI database) with 30,000 distinct words

- Run time: 12 minutes (compared to 10 hours for MALLET and other state-of-the-art topic modeling tools)
- Topics are **high quality** (because we can handle correlations in the topic model?)

Let's check out the results!

Concluding Remarks

Our algorithms are based on answering a purely algebraic question: How many variables do we need in a semi-algebraic set to encode nonnegative rank?

Concluding Remarks

Our algorithms are based on answering a purely algebraic question: How many variables do we need in a semi-algebraic set to encode nonnegative rank?

Question

Are there other examples of a better understanding of the expressive power of semi-algebraic sets can lead to new algorithms?

Concluding Remarks

Our algorithms are based on answering a purely algebraic question: How many variables do we need in a semi-algebraic set to encode nonnegative rank?

Question

Are there other examples of a better understanding of the expressive power of semi-algebraic sets can lead to new algorithms?

Observation

The number of variables plays an analogous role to VC-dimension

Concluding Remarks

Our algorithms are based on answering a purely algebraic question: How many variables do we need in a semi-algebraic set to encode nonnegative rank?

Question

Are there other examples of a better understanding of the expressive power of semi-algebraic sets can lead to new algorithms?

Observation

The number of variables plays an analogous role to VC-dimension

Is there an elementary proof of the Milnor-Warren bound?

Any Questions?

Thanks!