LSI vs Link Analysis
(A Survey)

C. D. Meyer and A. N. Langville

Department of Mathematics
North Carolina University
Raleigh, NC

1/23/2003

Outline

- **Background & History**

Outline

- Background & History
- Vector Space Approach

Outline

- Background & History
- Vector Space Approach
- Link Analysis Approach

Outline

- Background & History
- Vector Space Approach
- Link Analysis Approach
- Hybrid Approachs

Background

Goal

- Identify documents that best match users query

Background

Goal

- Identify documents that best match users query

Measures

- Recall $= \dfrac{\#relevant\ docs\ retrieved}{\#docs\ in\ collection}$ (max # useful docs)

- Precision $= \dfrac{\#relevant\ docs\ retrieved}{\#docs\ retrieved}$ (min # useless docs)

Background

Goal

- Identify documents that best match users query

Measures

- Recall $= \dfrac{\#relevant\ docs\ retrieved}{\#docs\ in\ collection}$ (max # useful docs)

- Precision $= \dfrac{\#relevant\ docs\ retrieved}{\#docs\ retrieved}$ (min # useless docs)

Do it *FAST!*

SMART

(**S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext)

SMART

(**S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext)

Harvard 1962 – 1965

- IBM 7094 & IBM 360

SMART

(**S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext)

Harvard 1962 – 1965

- IBM 7094 & IBM 360

Gerard Salton

- Implemented at Cornell (1965 – 1970)

SMART

(System for the Mechanical Analysis and Retrieval of Text)

Harvard 1962 – 1965

- IBM 7094 & IBM 360

Gerard Salton

- Implemented at Cornell (1965 – 1970)
- Based on matrix methods

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., $landing\ gear$)

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., $landing\ gear$)

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., $landing\ gear)$

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

$$
\begin{array}{c}
\\
\textsc{Term 1} \\
\textsc{Term 2} \\
\vdots \\
\textsc{Term m}
\end{array}
\begin{array}{c}
\begin{array}{cccc} \textsc{Doc 1} & \textsc{Doc 2} & \cdots & \textsc{Doc n} \end{array} \\
\begin{pmatrix}
f_{\mathbf{11}} & f_{\mathbf{12}} & \cdots & f_{\mathbf{1}n} \\
f_{\mathbf{21}} & f_{\mathbf{22}} & \cdots & f_{\mathbf{2}n} \\
\vdots & \vdots & \ddots & \vdots \\
f_{m\mathbf{1}} & f_{m\mathbf{2}} & \cdots & f_{mn}
\end{pmatrix}
\end{array}
= \mathbf{A}_{m\times n}
$$

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., *landing gear*)

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

$$\begin{array}{c} \\ \textsc{Term}\ 1 \\ \textsc{Term}\ 2 \\ \vdots \\ \textsc{Term}\ \mathsf{m} \end{array} \begin{array}{c} \begin{array}{cccc} \textsc{Doc}\ 1 & \textsc{Doc}\ 2 & \cdots & \textsc{Doc}\ \mathsf{n} \end{array} \\ \begin{pmatrix} f_{\mathbf{11}} & f_{\mathbf{12}} & \cdots & f_{\mathbf{1}n} \\ f_{\mathbf{21}} & f_{\mathbf{22}} & \cdots & f_{\mathbf{2}n} \\ \vdots & \vdots & \ddots & \vdots \\ f_{m\mathbf{1}} & f_{m\mathbf{2}} & \cdots & f_{mn} \end{pmatrix} \end{array} = \mathbf{A}_{m\times n}$$

Features

- $\mathbf{A} \geq \mathbf{0}$

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., *landing gear*)

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

$$\begin{array}{c} \\ \textsc{Term 1} \\ \textsc{Term 2} \\ \vdots \\ \textsc{Term m} \end{array} \begin{array}{c} \begin{array}{cccc} \textsc{Doc 1} & \textsc{Doc 2} & \cdots & \textsc{Doc n} \end{array} \\ \begin{pmatrix} f_{\mathbf{11}} & f_{\mathbf{12}} & \cdots & f_{\mathbf{1}n} \\ f_{\mathbf{21}} & f_{\mathbf{22}} & \cdots & f_{\mathbf{2}n} \\ \vdots & \vdots & \ddots & \vdots \\ f_{m\mathbf{1}} & f_{m\mathbf{2}} & \cdots & f_{mn} \end{pmatrix} \end{array} = \mathbf{A}_{m \times n}$$

Features

- $\mathbf{A} \geq \mathbf{0}$
- **A** can be really big

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., $landing\ gear$)

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

$$\begin{array}{c} \\ \textsc{Term } 1 \\ \textsc{Term } 2 \\ \vdots \\ \textsc{Term } m \end{array} \begin{array}{c} \begin{array}{cccc} \textsc{Doc } 1 & \textsc{Doc } 2 & \cdots & \textsc{Doc } n \end{array} \\ \begin{pmatrix} f_{\mathbf{11}} & f_{\mathbf{12}} & \cdots & f_{\mathbf{1}n} \\ f_{\mathbf{21}} & f_{\mathbf{22}} & \cdots & f_{\mathbf{2}n} \\ \vdots & \vdots & \ddots & \vdots \\ f_{m\mathbf{1}} & f_{m\mathbf{2}} & \cdots & f_{mn} \end{pmatrix} \end{array} = \mathbf{A}_{m\times n}$$

Features

- $\mathbf{A} \geq \mathbf{0}$
- **A** can be really big
- **A** is sparse — but otherwise unstructured

Term–Document Matrix

Start With Dictionary of Terms

- Single words — or short phrases (e.g., *landing gear*)

Index Each Document (by human or by computer)

- Count f_{ij} = # times term i appears in document j

Term–Document Matrix

$$
\begin{array}{c c}
 & \begin{array}{cccc} \text{Doc 1} & \text{Doc 2} & \cdots & \text{Doc n} \end{array} \\
\begin{array}{c} \text{Term 1} \\ \text{Term 2} \\ \vdots \\ \text{Term m} \end{array} &
\begin{pmatrix}
f_{11} & f_{12} & \cdots & f_{1n} \\
f_{21} & f_{22} & \cdots & f_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
f_{m1} & f_{m2} & \cdots & f_{mn}
\end{pmatrix}
\end{array}
= \mathbf{A}_{m \times n}
$$

Features

- $\mathbf{A} \geq 0$
- **A** can be really big
- **A** is sparse — but otherwise unstructured
- **A** contains a lot of uncertainty

Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

- i.e., how close is $\mathbf{q}$ to each column $\mathbf{A}_i$?

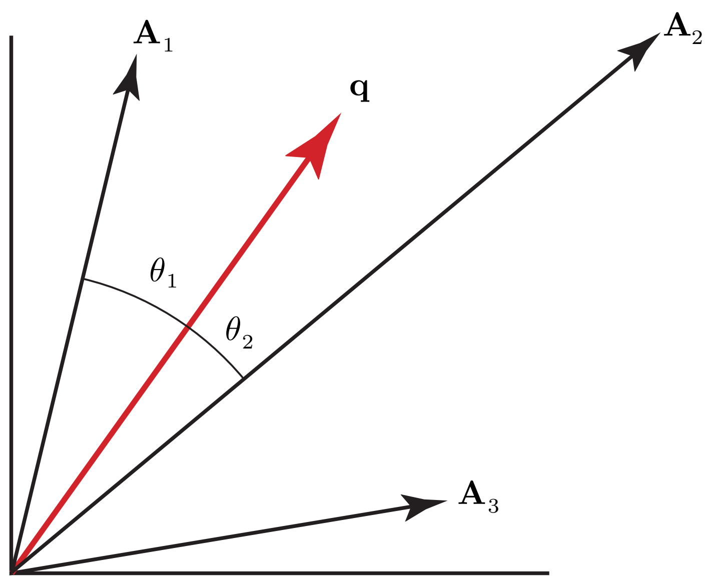
Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

- i.e., how close is $\mathbf{q}$ to each column $\mathbf{A}_i$?



$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\|$ but $\theta_2 < \theta_1$

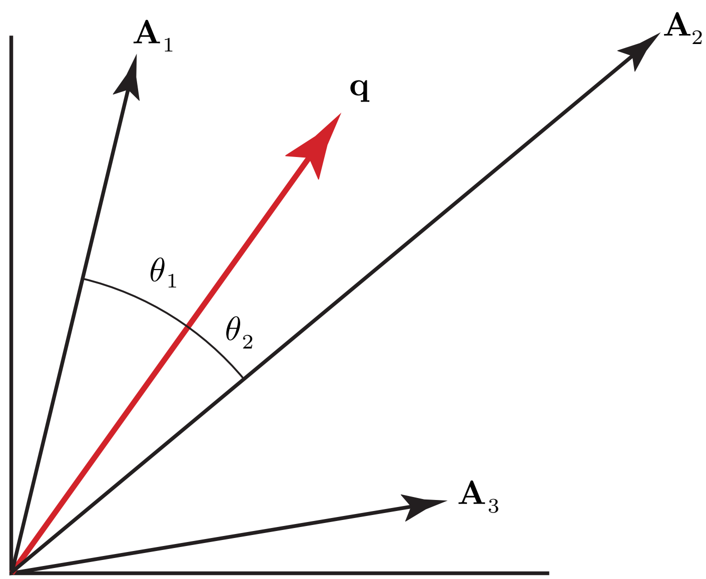
Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

- i.e., how close is **q** to each column $\mathbf{A}_i$?



$$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\| \text{ but } \theta_2 < \theta_1$$

$$\text{Use } \delta_i = \cos\theta_i = \frac{\mathbf{q}^T \mathbf{A}_i}{\|\mathbf{q}\| \, \|\mathbf{A}_i\|}$$

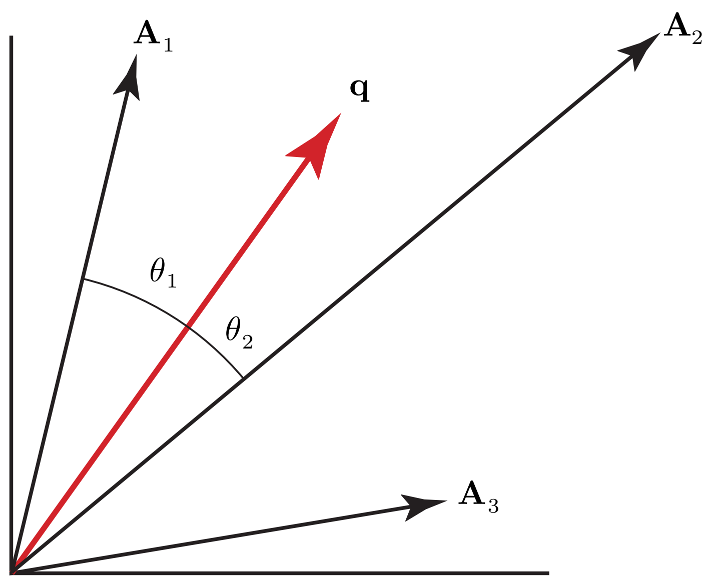
Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

- i.e., how close is **q** to each column $\mathbf{A}_i$?



$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\|$ but $\theta_2 < \theta_1$

Use $\delta_i = \cos\theta_i = \dfrac{\mathbf{q}^T \mathbf{A}_i}{\|\mathbf{q}\| \, \|\mathbf{A}_i\|}$

Rank documents by size of δ_i

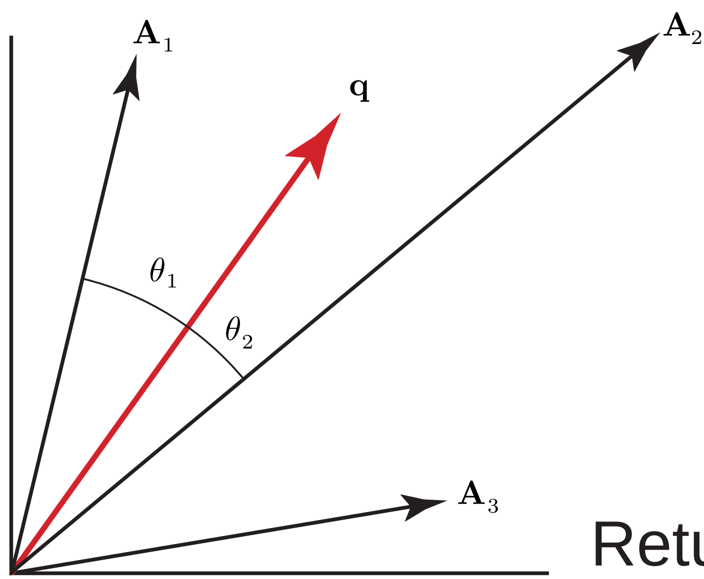
Query Matching

Query Vector

- $\mathbf{q}^T = (q_1, q_2, \ldots, q_m)$ where $q_i = \begin{cases} 1 & \text{if Term } i \text{ is requested} \\ 0 & \text{if not} \end{cases}$

How Close is the Query to Each Document?

- i.e., how close is **q** to each column $\mathbf{A}_i$?



$\|\mathbf{q} - \mathbf{A}_1\| < \|\mathbf{q} - \mathbf{A}_2\|$ but $\theta_2 < \theta_1$

Use $\delta_i = \cos\theta_i = \dfrac{\mathbf{q}^T\mathbf{A}_i}{\|\mathbf{q}\|\,\|\mathbf{A}_i\|}$

Rank documents by size of δ_i

Return Document i to user when $\delta_i \geq tol$

Term Weighting

A Defect

- If the term $bank$ occurs once in Doc $\mathbf{1}$ but twice in Doc $\mathbf{2}$, and if $\|\mathbf{A_1}\| \approx \|\mathbf{A_2}\|$, then a query containing only $bank$ produces $\delta_\mathbf{2} \approx \mathbf{2}\delta_\mathbf{1}$ (i.e., Doc $\mathbf{2}$ is rated twice as relevant as Doc $\mathbf{1}$).

Term Weighting

A Defect

- If the term $bank$ occurs once in Doc $\mathbf{1}$ but twice in Doc $\mathbf{2}$, and if $\|\mathbf{A_1}\| \approx \|\mathbf{A_2}\|$, then a query containing only $bank$ produces $\delta_{\mathbf{2}} \approx \mathbf{2}\delta_{\mathbf{1}}$ (i.e., Doc $\mathbf{2}$ is rated twice as relevant as Doc $\mathbf{1}$).

To Compensate

- Set $a_{ij} = \log(\mathbf{1} + f_{ij})$ (other weights also possible)

Term Weighting

A Defect

- If the term $bank$ occurs once in Doc 1 but twice in Doc $\mathbf{2}$, and if $\|\mathbf{A_1}\| \approx \|\mathbf{A_2}\|$, then a query containing only $bank$ produces $\delta_{\mathbf{2}} \approx \mathbf{2}\delta_{\mathbf{1}}$ (i.e., Doc $\mathbf{2}$ is rated twice as relevant as Doc $\mathbf{1}$).

To Compensate

- Set $a_{ij} = \log(1 + f_{ij})$ (other weights also possible)

Query Weights

- Terms $Boeing$ and $airplanes$ not equally important in queries

Term Weighting

A Defect

- If the term $bank$ occurs once in Doc 1 but twice in Doc 2, and if $\|\mathbf{A_1}\| \approx \|\mathbf{A_2}\|$, then a query containing only $bank$ produces $\delta_2 \approx 2\delta_1$ (i.e., Doc 2 is rated twice as relevant as Doc 1).

To Compensate

- Set a_{ij} = log(1 + f_{ij}) (other weights also possible)

Query Weights

- Terms $Boeing$ and $airplanes$ not equally important in queries
- Importance of Term i tends to be inversely proportional to ν_i = # Docs containing Term i

Term Weighting

A Defect

- If the term $bank$ occurs once in Doc 1 but twice in Doc 2, and if $\|\mathbf{A_1}\| \approx \|\mathbf{A_2}\|$, then a query containing only $bank$ produces $\delta_2 \approx 2\delta_1$ (i.e., Doc 2 is rated twice as relevant as Doc 1).

To Compensate

- Set $a_{ij} = \log(1 + f_{ij})$ (other weights also possible)

Query Weights

- Terms $Boeing$ and $airplanes$ not equally important in queries
- Importance of Term i tends to be inversely proportional to ν_i = # Docs containing Term i

To Compensate

- Set $q_i = \begin{cases} \log(n/\nu_i) & \text{if } \nu_i \neq 0 \\ 0 & \text{if } \nu_i = 0 \end{cases}$ (other weights also possible)

Uncertainties in A

Uncertainties in A

Ambiguity in Vocabulary

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be $\cdots$

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be $\cdots$
  - A flat geometrical object

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be $\cdots$
  - A flat geometrical object
  - A woodworking tool

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be $\cdots$
  - A flat geometrical object
  - A woodworking tool
  - A Boeing product

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be $\cdots$
  — A flat geometrical object
  — A woodworking tool
  — A Boeing product

Variation in Writing Style

- No two authors write the same way

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be
  - A flat geometrical object
  - A woodworking tool
  - A Boeing product

Variation in Writing Style

- No two authors write the same way
  - One author may write car and $laptop$

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be
  - A flat geometrical object
  - A woodworking tool
  - A Boeing product

Variation in Writing Style

- No two authors write the same way
  - One author may write car and $laptop$
  - Another author may write $automobile$ and $portable$

Uncertainties in A

Ambiguity in Vocabulary

- e.g., A $plane$ could be
  - A flat geometrical object
  - A woodworking tool
  - A Boeing product

Variation in Writing Style

- No two authors write the same way
  - One author may write car and $laptop$
  - Another author may write $automobile$ and $portable$

Variation in Indexing Conventions

- No two people index documents the same way
- Computer indexing is inexact and can be unpredictable

Theory vs Practice

In Theory — it’s easy

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)
- D_2 indexed $automobile,\ fuel,\ and\ tire$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)
- D_2 indexed $automobile,\ fuel,\ and\ tire$ (missed)

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)
- D_2 indexed $automobile,\ fuel,\ and\ tire$ (missed)

Somehow Reveal Latent Connections

- Find D_2 by making the connection through $tire$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)
- D_2 indexed $automobile,\ fuel,\ and\ tire$ (missed)

Somehow Reveal Latent Connections

- Find D_2 by making the connection through $tire$
- Do it $FAST!$

Theory vs Practice

In Theory — it's easy

- Weight terms and normalize cols — Make $\|\mathbf{A}_i\| = 1$
- For each new query, weight and normalize — Make $\|\mathbf{q}\| = 1$
- Compute $\delta_i = \cos\theta_i = (\mathbf{q}^T\mathbf{A})_i$ to return the most relevant docs

In Practice — it's not so easy

- Suppose query = gas
- D_1 indexed by $gas,\ car,\ tire$ (found)
- D_2 indexed $automobile,\ fuel,\ and\ tire$ (missed)

Somehow Reveal Latent Connections

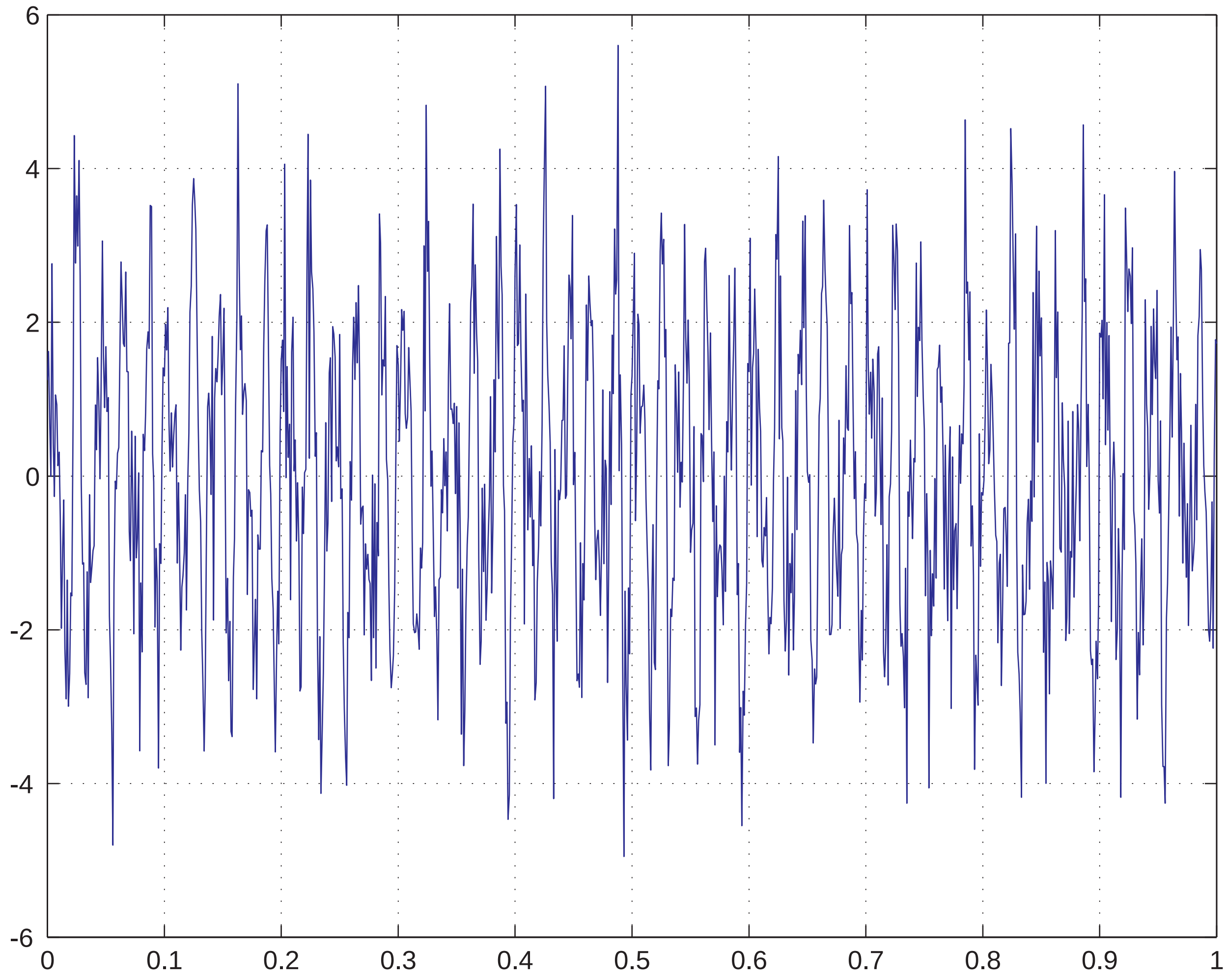
- Find D_2 by making the connection through $tire$
- Do it $FAST!$
  - Data compression

Contaminated Data (not text data)

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{510} \\ x_{511} \end{bmatrix}$$

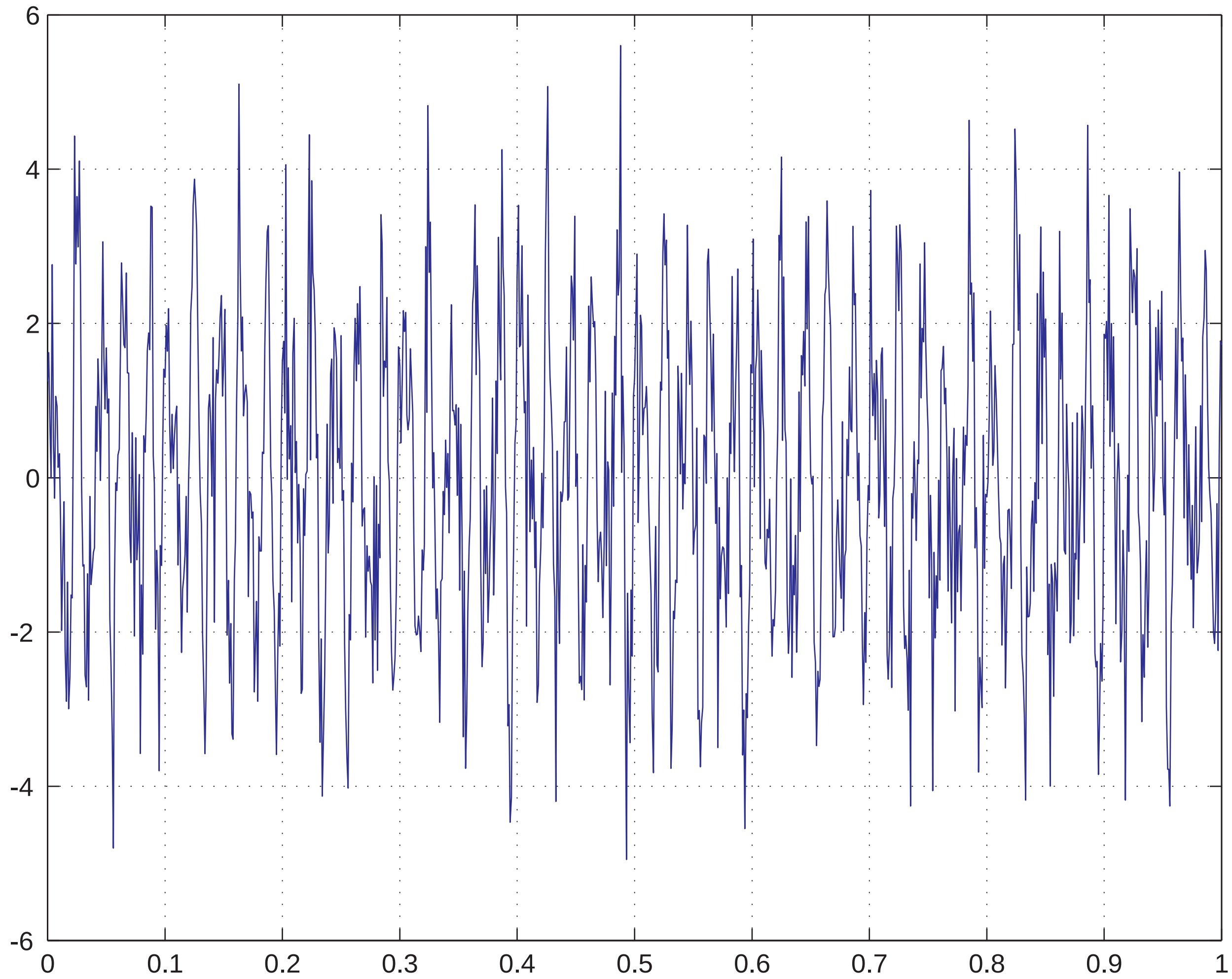
Contaminated Data (not text data)

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{510} \\ x_{511} \end{bmatrix}$$



Contaminated Data (not text data)

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{510} \\ x_{511} \end{bmatrix}$$

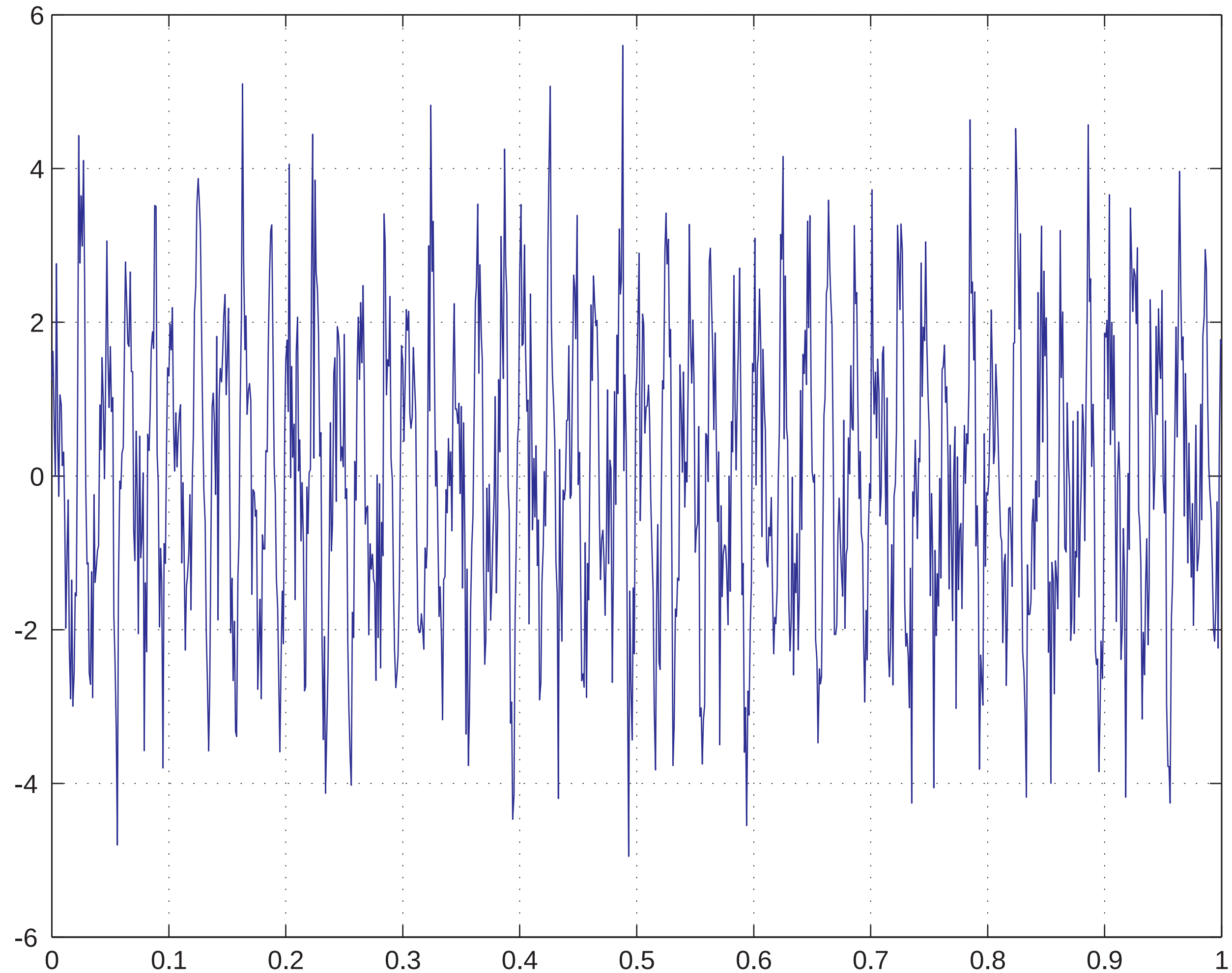


Goal

- Reveal hidden patterns

Contaminated Data (not text data)

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{510} \\ x_{511} \end{bmatrix}$$



Goal

- Reveal hidden patterns
- Compress the data

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W_0}, \mathbf{W_1}, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W_0}, \mathbf{W_1}, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_\mathbf{0} \,|\, W_\mathbf{1} \,|\, \cdots \,|\, W_{n-\mathbf{1}})$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  - $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y}=\mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  - $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y}=\mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Oscillatory

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  — Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  — $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  — $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  — $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y}=\mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Oscillatory

- $$\mathbf{W} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega \end{bmatrix}_{n\times n} \qquad \omega = e^{2\pi i/n}$$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  - $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y}=\mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Oscillatory

- $\mathbf{W} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega \end{bmatrix}_{n\times n}$ $\omega = e^{2\pi i/n}$, $W_k = \dfrac{e^{2\pi i k t}}{2}$

  $t = 0, 1/n, 2/n, \ldots$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  - $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y}=\mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Oscillatory

- $$\mathbf{W} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega \end{bmatrix}_{n\times n} \qquad \omega = e^{2\pi i/n}, \qquad W_k = \frac{e^{2\pi i k \mathbf{t}}}{2}$$

  $t = 0, 1/n, 2/n, \ldots$

- $W_k + W_{n-k} = \cos 2\pi k\mathbf{t}$

Change Of Coordinates

New Basis $\mathcal{B} = \{\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{n-1}\}$

- Find coordinates of $\mathbf{x}$ with respect to $\mathcal{B}$
  - Find y_k so that $\mathbf{x} = \sum y_k W_k$ (Fourier expansion if $\mathcal{B}$ o.n.)
  - $y_k = \langle \mathbf{W}_k | \mathbf{x} \rangle$ = amount of $\mathbf{x}$ in direction of W_k (if $\mathcal{B}$ o.n.)
  - $\mathbf{x} = \mathbf{W}\mathbf{y}$ where $\mathbf{W} = (W_0 \,|\, W_1 \,|\, \cdots \,|\, W_{n-1})$
  - $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$ ($\mathbf{y} = \mathbf{W}^*\mathbf{x}$ if $\mathcal{B}$ o.n.)

Oscillatory

- $$\mathbf{W} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega \end{bmatrix}_{n\times n} \qquad \omega = e^{2\pi i/n}, \qquad W_k = \frac{e^{2\pi i k \mathbf{t}}}{2}$$

  $t = 0, 1/n, 2/n, \ldots$

- $W_k + W_{n-k} = \cos 2\pi k \mathbf{t}$
- $W_k - W_{n-k} = i \sin 2\pi k \mathbf{t}$ $\quad (0 < k < n)$

Making The Change

Making The Change

Recall

- $\mathbf{x} = \sum y_k W_k = \mathbf{W}\mathbf{y}$

Making The Change

Recall

- $\mathbf{x} = \sum y_k W_k = \mathbf{W}\mathbf{y}$
- $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$

Making The Change

Recall

- $\mathbf{x} = \sum y_k W_k = \mathbf{W}\mathbf{y}$
- $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$

$\mathbf{W}^{-1} = (4/n)\overline{\mathbf{W}}$ = Discrete Fourier Transform

Making The Change

Recall

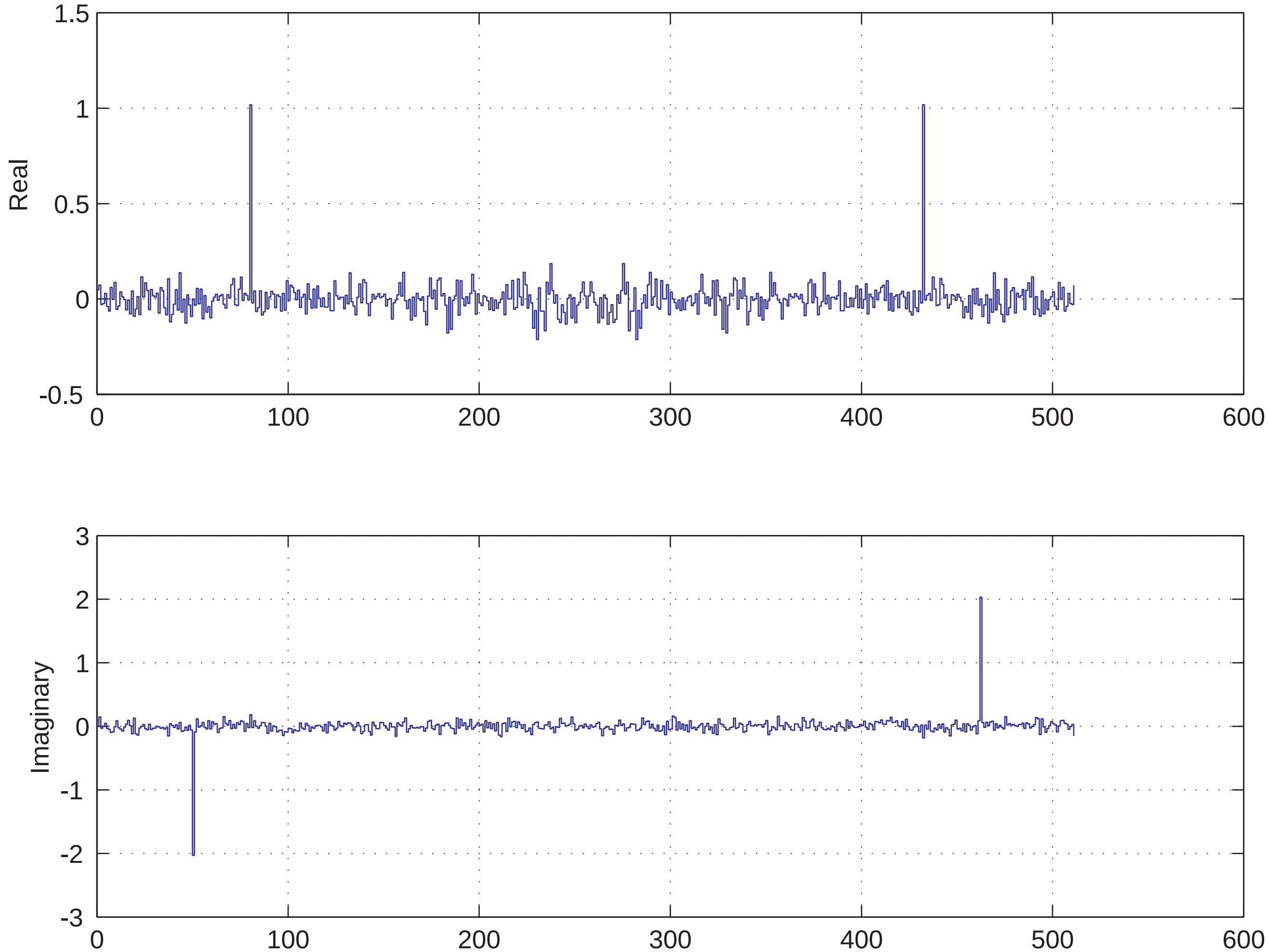
- $\mathbf{x} = \sum y_k W_k = \mathbf{Wy}$
- $\mathbf{y} = \mathbf{W}^{-1}\mathbf{x}$

$\mathbf{W}^{-1} = (4/n)\overline{\mathbf{W}}$ = Discrete Fourier Transform

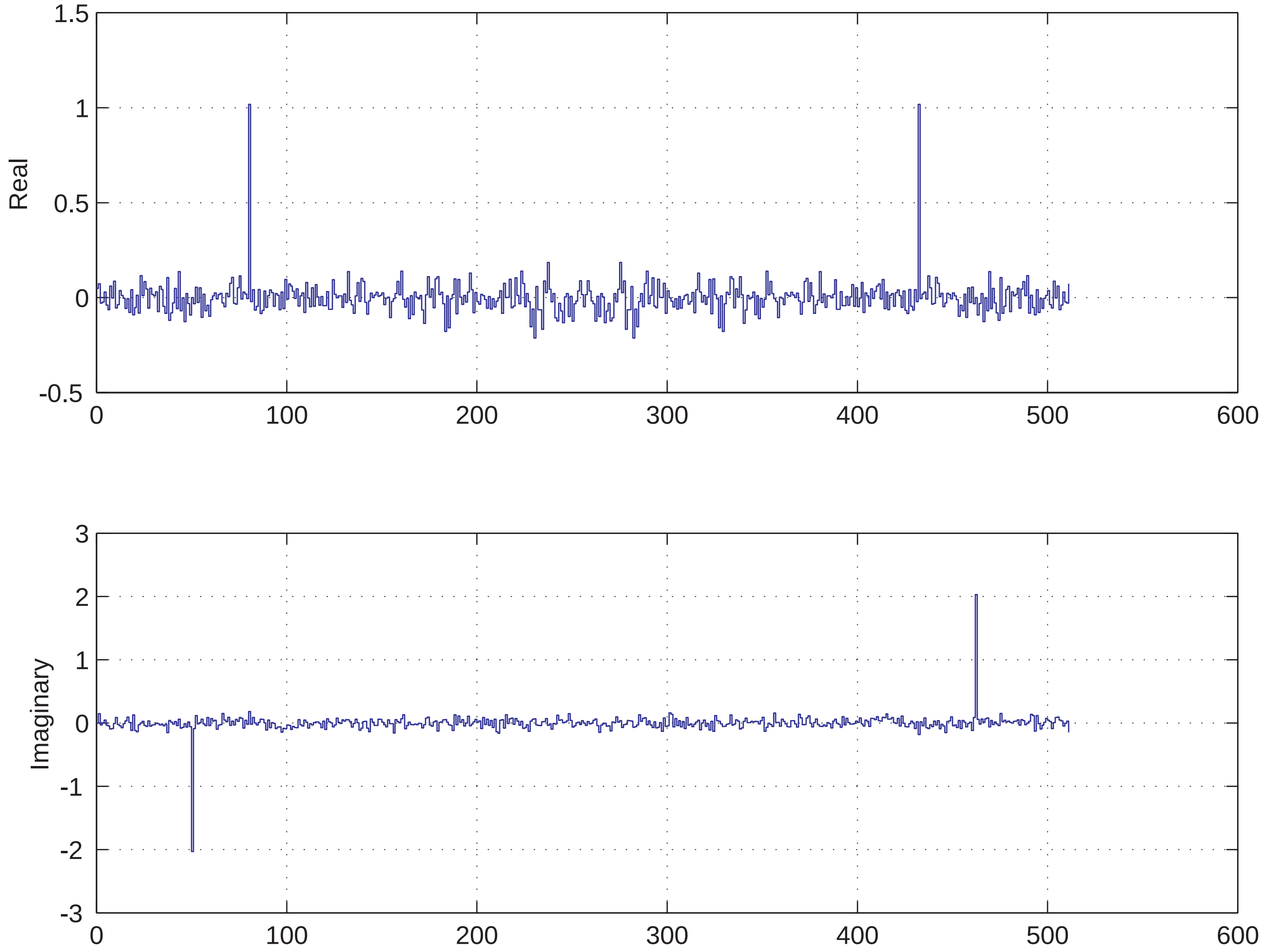
$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{bmatrix} = \frac{\mathbf{2}}{n} \begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \cdots & \mathbf{1} \\ \mathbf{1} & \xi & \xi^{\mathbf{2}} & \cdots & \xi^{n-\mathbf{1}} \\ \mathbf{1} & \xi^{\mathbf{2}} & \xi^{\mathbf{4}} & \cdots & \xi^{n-\mathbf{2}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{1} & \xi^{n-\mathbf{1}} & \xi^{n-\mathbf{2}} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_{\mathbf{0}} \\ x_{\mathbf{1}} \\ x_{\mathbf{2}} \\ \vdots \\ x_{n-\mathbf{1}} \end{bmatrix}$$

$$\xi = \mathrm{e}^{-\mathbf{2}\pi \mathrm{i}/n} = \overline{\omega}$$

The New Coordinates

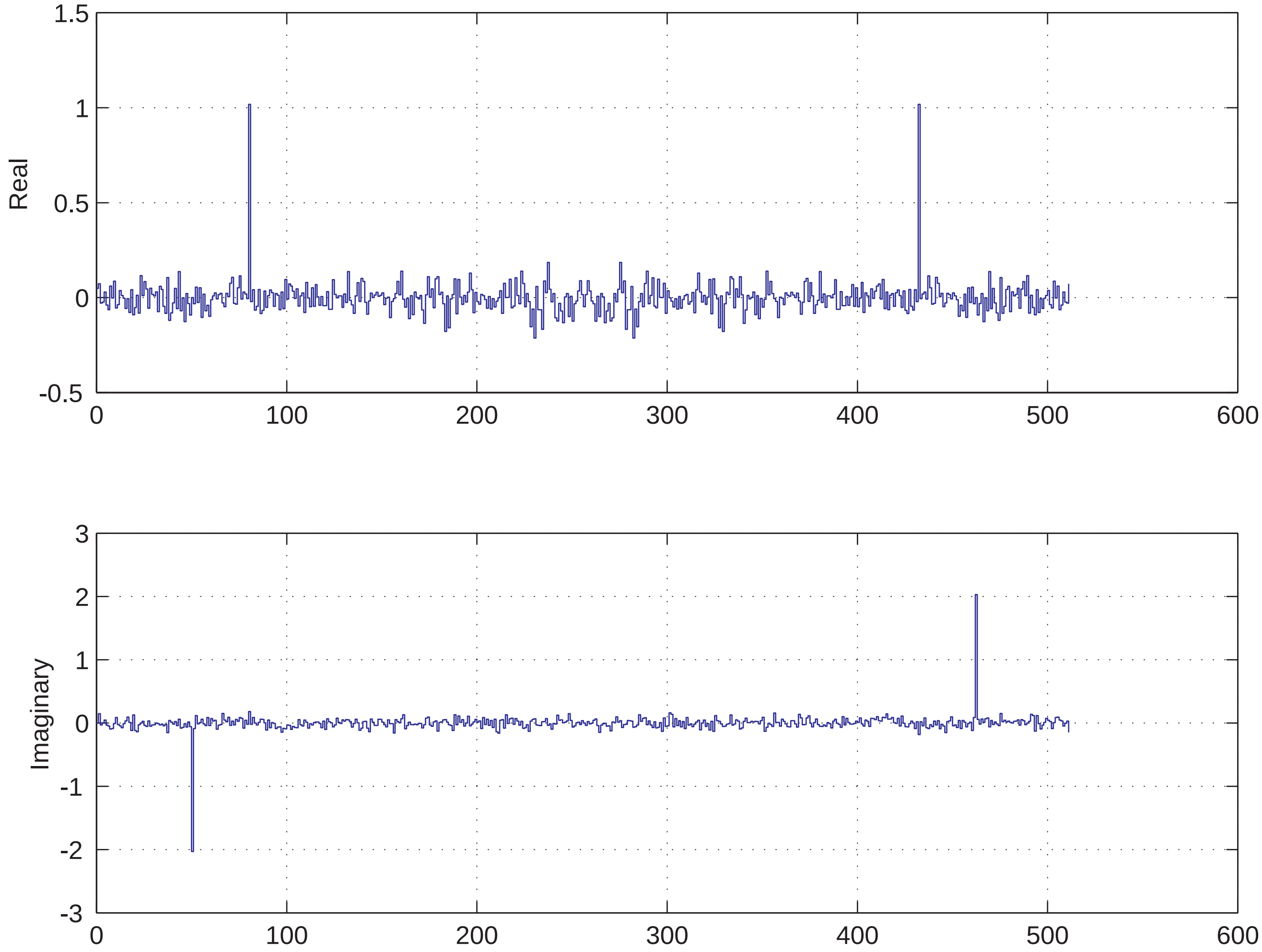


The New Coordinates



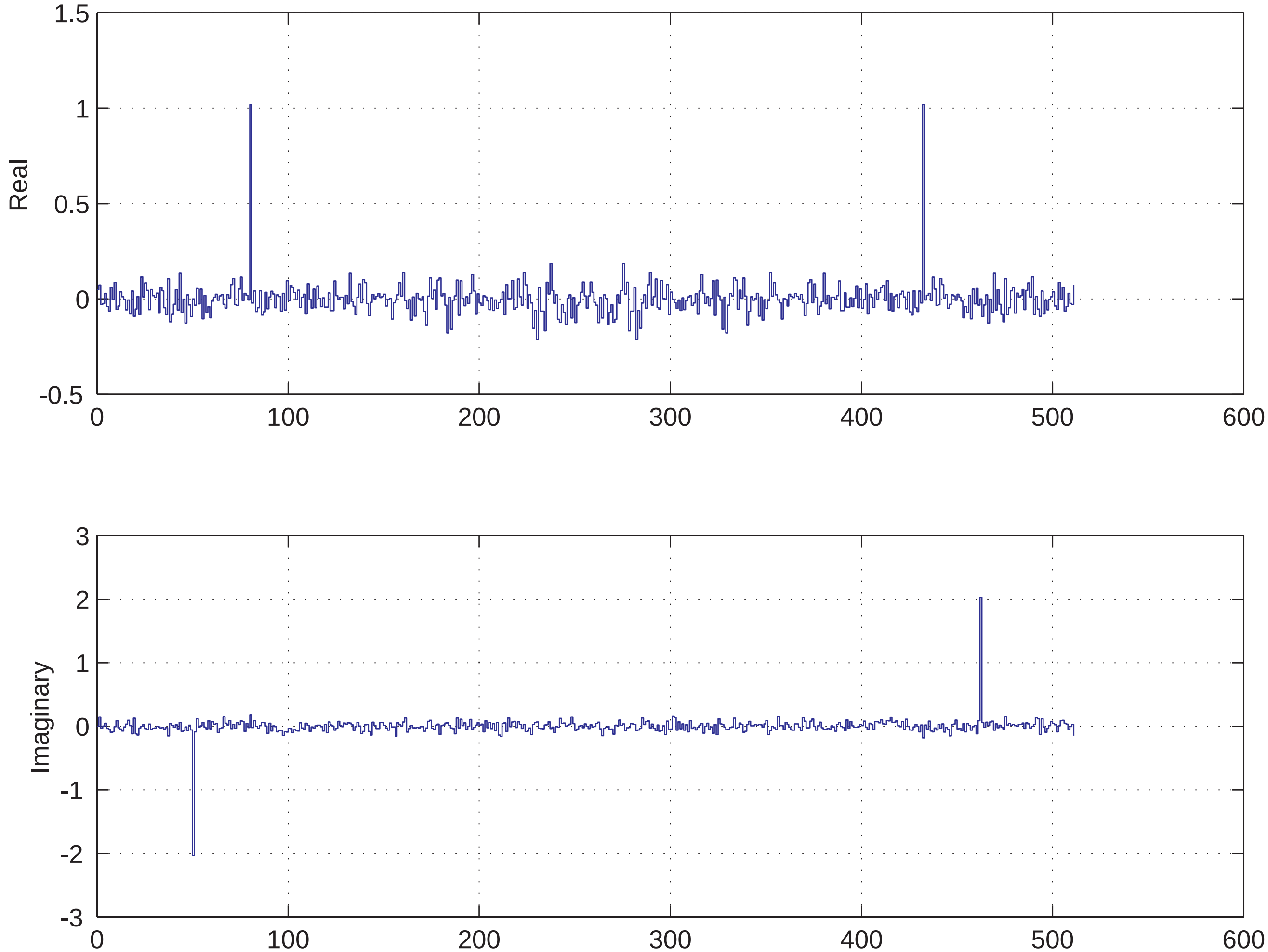
- Only 4 are significant: $y_{80} = y_{432} = 1$

The New Coordinates



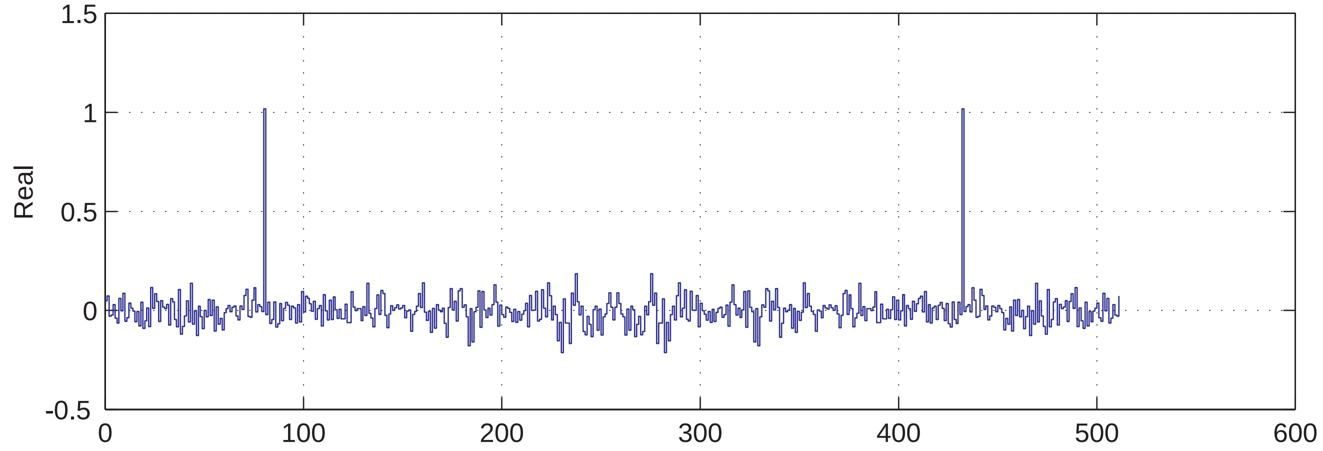
- Only 4 are significant: $y_{80} = y_{432} = 1$ and $y_{50} = -2\mathrm{i} = -y_{462}$

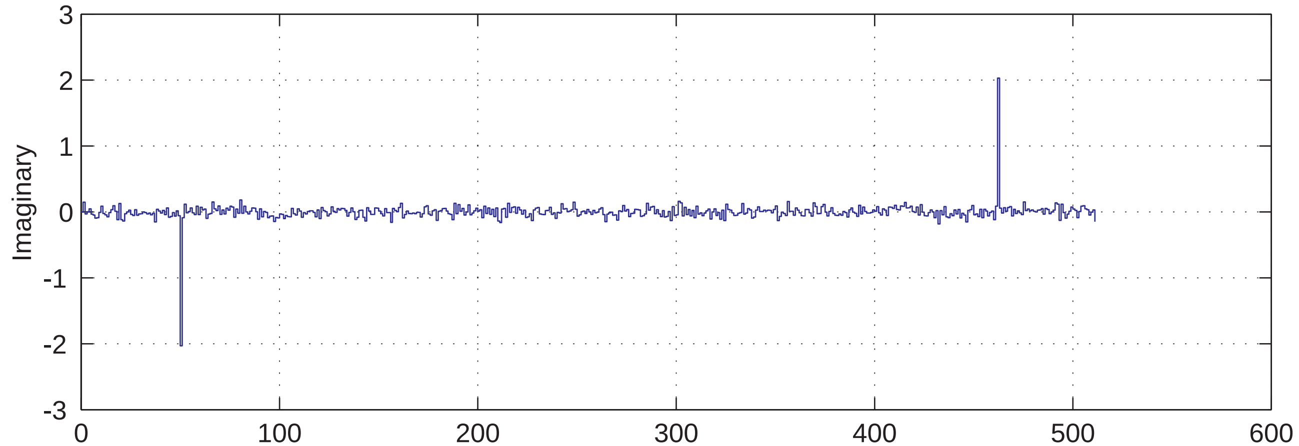
The New Coordinates



- Only 4 are significant: $y_{80} = y_{432} = 1$ and $y_{50} = -2\mathrm{i} = -y_{462}$
- $\mathbf{x} = \sum y_k W_k = 1W_{80} + 1W_{432} - 2\mathrm{i}W_{50} + 2\mathrm{i}W_{462} + \sum \varepsilon_j W_j$

The New Coordinates





- Only 4 are significant: $y_{80} = y_{432} = 1$ and $y_{50} = -\mathbf{2}\mathrm{i} = -y_{462}$
- $\mathbf{x} = \sum y_k W_k = \mathbf{1}W_{\mathbf{80}} + \mathbf{1}W_{\mathbf{432}} - \mathbf{2}\mathrm{i}W_{\mathbf{50}} + \mathbf{2}\mathrm{i}W_{\mathbf{462}} + \sum \varepsilon_j W_j$
- Small components (noise) are nondirectional

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = \mathbf{1} W_{\mathbf{80}} + \mathbf{1} W_{\mathbf{432}} - \mathbf{2}\mathrm{i} W_{\mathbf{50}} + \mathbf{2}\mathrm{i} W_{\mathbf{462}} + \sum \varepsilon_j W_j$

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = \mathbf{1} W_{\mathbf{80}} + \mathbf{1} W_{\mathbf{432}} - \mathbf{2}\mathrm{i} W_{\mathbf{50}} + \mathbf{2}\mathrm{i} W_{\mathbf{462}} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{\mathbf{80}} + W_{\mathbf{432}}) - \mathbf{2}\mathrm{i}(W_{\mathbf{50}} - W_{\mathbf{462}})$

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = \mathbf{1} W_{\mathbf{80}} + \mathbf{1} W_{\mathbf{432}} - \mathbf{2}\mathsf{i} W_{\mathbf{50}} + \mathbf{2}\mathsf{i} W_{\mathbf{462}} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{\mathbf{80}} + W_{\mathbf{432}}) - \mathbf{2}\mathsf{i}(W_{\mathbf{50}} - W_{\mathbf{462}})$
- $n = \mathbf{512}$
- $\widetilde{\mathbf{x}} = (W_{\mathbf{80}} + W_{n-\mathbf{80}}) - \mathbf{2}\mathsf{i}(W_{\mathbf{50}} - W_{n-\mathbf{50}})$

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = \mathbf{1} W_{\mathbf{80}} + \mathbf{1} W_{\mathbf{432}} - \mathbf{2}\mathrm{i} W_{\mathbf{50}} + \mathbf{2}\mathrm{i} W_{\mathbf{462}} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{\mathbf{80}} + W_{\mathbf{432}}) - \mathbf{2}\mathrm{i}(W_{\mathbf{50}} - W_{\mathbf{462}})$
- $n = \mathbf{512}$
- $\widetilde{\mathbf{x}} = (W_{\mathbf{80}} + W_{n-\mathbf{80}}) - \mathbf{2}\mathrm{i}(W_{\mathbf{50}} - W_{n-\mathbf{50}})$ Compressed $(\mathbf{512} \to \mathbf{4})$

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = 1W_{80} + 1W_{432} - 2\mathrm{i}W_{50} + 2\mathrm{i}W_{462} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{432}) - 2\mathrm{i}(W_{50} - W_{462})$
- $n = 512$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{n-80}) - 2\mathrm{i}(W_{50} - W_{n-50})$ Compressed $(512 \to 4)$
  - $W_k + W_{n-k} = \cos 2\pi k\mathbf{t}$
  - $W_k - W_{n-k} = \mathrm{i}\sin 2\pi k\mathbf{t}$

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = 1W_{80} + 1W_{432} - 2\mathrm{i}W_{50} + 2\mathrm{i}W_{462} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{432}) - 2\mathrm{i}(W_{50} - W_{462})$
- $n = 512$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{n-80}) - 2\mathrm{i}(W_{50} - W_{n-50})$ Compressed $(512 \to 4)$
  - $W_k + W_{n-k} = \cos 2\pi k\mathbf{t}$
  - $W_k - W_{n-k} = \mathrm{i}\,\sin 2\pi k\mathbf{t}$
- $\widetilde{\mathbf{x}} = \cos 2\pi 80\mathbf{t} + 2\sin 2\pi 50\mathbf{t}$

Drop Small Coordinates

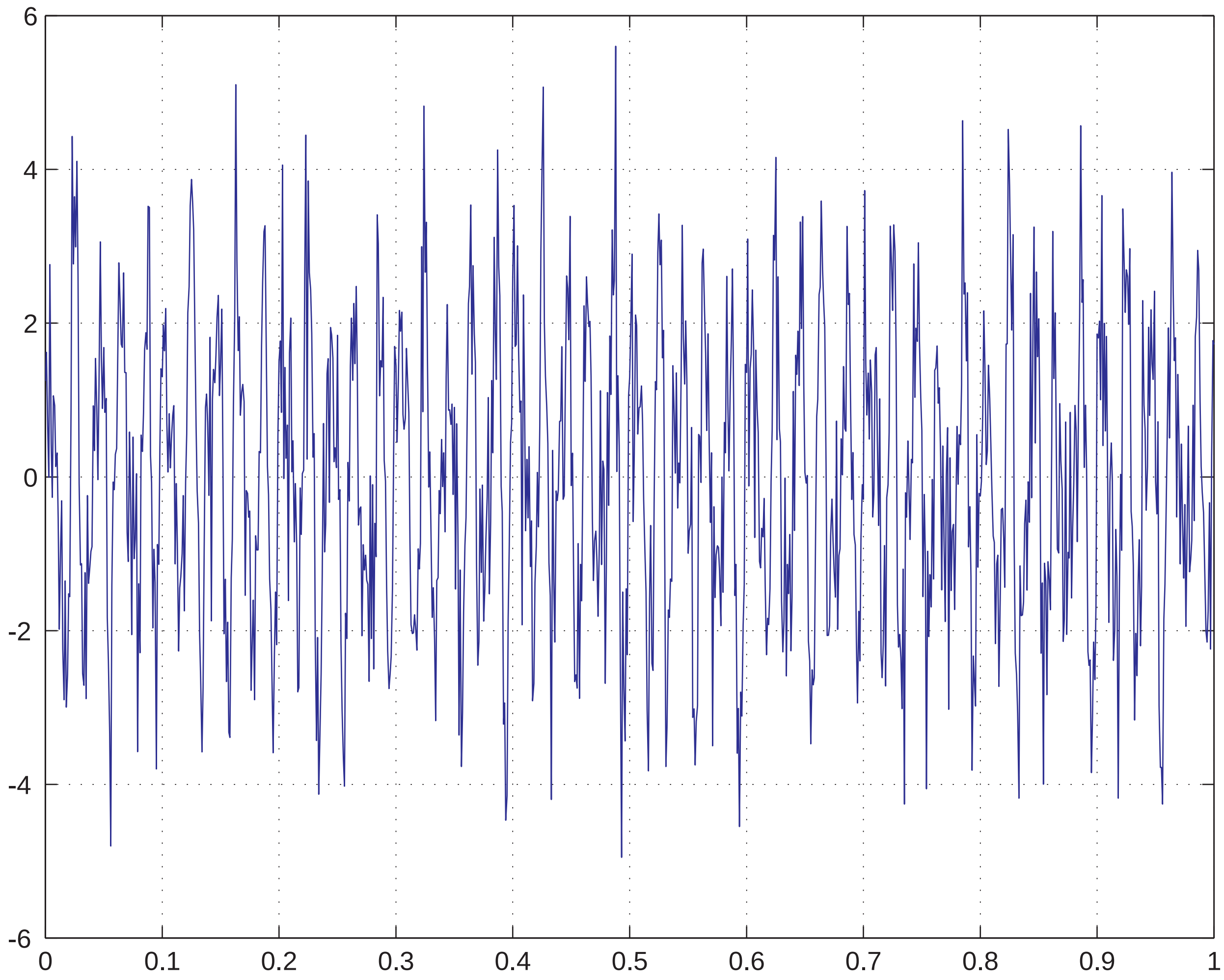
- $\mathbf{x} = \sum y_k W_k = 1W_{80} + 1W_{432} - 2\mathrm{i}W_{50} + 2\mathrm{i}W_{462} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{432}) - 2\mathrm{i}(W_{50} - W_{462})$
- $n = 512$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{n-80}) - 2\mathrm{i}(W_{50} - W_{n-50})$ Compressed $(512 \to 4)$
  - $W_k + W_{n-k} = \cos 2\pi k\mathbf{t}$
  - $W_k - W_{n-k} = \mathrm{i}\sin 2\pi k\mathbf{t}$
- $\widetilde{\mathbf{x}} = \cos 2\pi 80\mathbf{t} + 2\sin 2\pi 50\mathbf{t}$ Cleaned

Drop Small Coordinates

- $\mathbf{x} = \sum y_k W_k = 1W_{80} + 1W_{432} - 2\mathrm{i}W_{50} + 2\mathrm{i}W_{462} + \sum \varepsilon_j W_j$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{432}) - 2\mathrm{i}(W_{50} - W_{462})$
- $n = 512$
- $\widetilde{\mathbf{x}} = (W_{80} + W_{n-80}) - 2\mathrm{i}(W_{50} - W_{n-50})$ Compressed $(512 \to 4)$
  - $W_k + W_{n-k} = \cos 2\pi k\mathbf{t}$
  - $W_k - W_{n-k} = \mathrm{i}\ \sin 2\pi k\mathbf{t}$
- $\widetilde{\mathbf{x}} = \cos 2\pi 80\mathbf{t} + 2 \sin 2\pi 50\mathbf{t}$ Cleaned
- $\mathbf{x} = \cos 2\pi 80\mathbf{t} + 2 \sin 2\pi 50\mathbf{t} + \text{noise}$

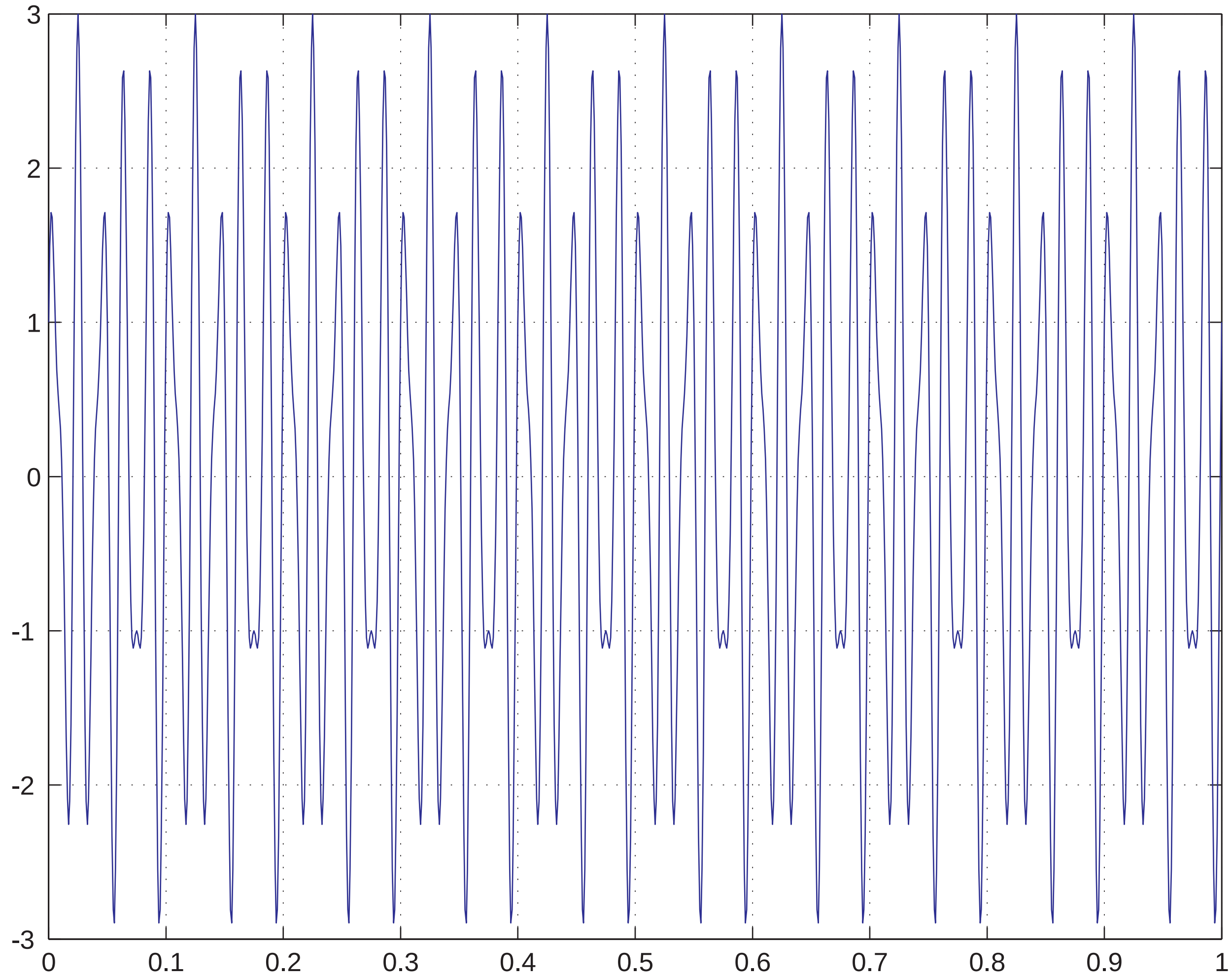
Original Data

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{510} \\ x_{511} \end{bmatrix}$$



Cleaned & Compressed Data

$$\widetilde{\mathbf{x}} = \mathbf{x} - \text{noise} = (W_{80} + W_{432}) - 2i(W_{50} - W_{462})$$



$$\cos 2\pi 80\mathbf{t} + 2\sin 2\pi 50\mathbf{t}$$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = \mathrm{e}^{-2\pi\mathrm{i}/n}$$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = \mathrm{e}^{-2\pi\mathrm{i}/n}$$

Simple in Theory, But $\cdots$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{\mathbf{2}}{n}\begin{bmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \cdots & \mathbf{1} \\ \mathbf{1} & \xi & \xi^{\mathbf{2}} & \cdots & \xi^{n-\mathbf{1}} \\ \mathbf{1} & \xi^{\mathbf{2}} & \xi^{\mathbf{4}} & \cdots & \xi^{n-\mathbf{2}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{1} & \xi^{n-\mathbf{1}} & \xi^{n-\mathbf{2}} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_{\mathbf{0}} \\ x_{\mathbf{1}} \\ x_{\mathbf{2}} \\ \vdots \\ x_{n-\mathbf{1}} \end{bmatrix} \qquad \xi = \mathrm{e}^{-\mathbf{2}\pi\mathrm{i}/n}$$

Simple in Theory, But $\cdots$

- Must do it $FAST!$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = \mathrm{e}^{-2\pi \mathrm{i}/n}$$

Simple in Theory, But $\cdots$

- Must do it *FAST!*

Need For Speed $\Longrightarrow$ Matrix Factorizations $\Longrightarrow$ FFT

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = \mathrm{e}^{-2\pi \mathrm{i}/n}$$

Simple in Theory, But $\cdots$

- Must do it *FAST!*

Need For Speed $\Longrightarrow$ Matrix Factorizations $\Longrightarrow$ FFT

- $\mathbf{F}_n = \mathbf{B}_n(\mathbf{I}_2 \otimes \mathbf{F}_{n/2})\mathbf{P}_n \qquad \mathbf{B}_n = \begin{bmatrix} \mathbf{I}_{n/2} & \mathbf{D}_{n/2} \\ \mathbf{I}_{n/2} & -\mathbf{D}_{n/2} \end{bmatrix} \qquad \mathbf{D}_{n/2} = \begin{bmatrix} 1 & & & \\ & \xi & & \\ & & \xi^2 & \\ & & & \ddots \end{bmatrix}$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = \mathrm{e}^{-2\pi \mathrm{i}/n}$$

Simple in Theory, But $\cdots$

- Must do it *FAST!*

Need For Speed $\Longrightarrow$ Matrix Factorizations $\Longrightarrow$ FFT

- $\mathbf{F}_n = \mathbf{B}_n(\mathbf{I}_2 \otimes \mathbf{F}_{n/2})\mathbf{P}_n \qquad \mathbf{B}_n = \begin{bmatrix} \mathbf{I}_{n/2} & \mathbf{D}_{n/2} \\ \mathbf{I}_{n/2} & -\mathbf{D}_{n/2} \end{bmatrix} \qquad \mathbf{D}_{n/2} = \begin{bmatrix} 1 & & & \\ & \xi & & \\ & & \xi^2 & \\ & & & \ddots \end{bmatrix}$
- FFT changes n^2 flop requirement into $(n/2)\log_2 n$

The DFT Game

Matrix–Vector Product

$$\mathbf{y} = \frac{2}{n}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \xi^{n-1} & \xi^{n-2} & \cdots & \xi \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \qquad \xi = e^{-2\pi i/n}$$

Simple in Theory, But $\cdots$

- Must do it *FAST!*

Need For Speed $\Longrightarrow$ Matrix Factorizations $\Longrightarrow$ FFT

- $\mathbf{F}_n = \mathbf{B}_n(\mathbf{I}_2 \otimes \mathbf{F}_{n/2})\mathbf{P}_n \qquad \mathbf{B}_n = \begin{bmatrix} \mathbf{I}_{n/2} & \mathbf{D}_{n/2} \\ \mathbf{I}_{n/2} & -\mathbf{D}_{n/2} \end{bmatrix} \qquad \mathbf{D}_{n/2} = \begin{bmatrix} 1 & & & \\ & \xi & & \\ & & \xi^2 & \\ & & & \ddots \end{bmatrix}$
- FFT changes n^2 flop requirement into $(n/2)\log_2 n$

"The most valuable numerical algorithm in our lifetime."
—G. Strang, Bulletin of the AMS, April, 1993.

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij} \qquad \mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij}$ $\mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Change Basis to $\mathcal{B} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots\}$ That Can Squeeze & Clean

- $\mathbf{A} = \sum \sigma_i \mathbf{Z}_i$ (Fourier Expansion)

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij} \qquad \mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Change Basis to $\mathcal{B} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots\}$ That Can Squeeze & Clean

- $\mathbf{A} = \sum \sigma_i \mathbf{Z}_i$ (Fourier Expansion)
- $\mathcal{B}$ o.n. $\Rightarrow \sigma_i = \langle \mathbf{Z}_i | \mathbf{A} \rangle$ = amount of $\mathbf{A}$ in direction of $\mathbf{Z}_i$

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij} \qquad \mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Change Basis to $\mathcal{B} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots\}$ That Can Squeeze & Clean

- $\mathbf{A} = \sum \sigma_i \mathbf{Z}_i$ (Fourier Expansion)

- $\mathcal{B}$ o.n. $\Rightarrow \sigma_i = \langle \mathbf{Z}_i | \mathbf{A} \rangle$ = amount of $\mathbf{A}$ in direction of $\mathbf{Z}_i$

Matrix Factorizations: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij}\mathbf{u}_i\mathbf{v}_j^T = \sum r_{ij}\mathbf{Z}_{ij}$

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij}$ $\quad\quad$ $\mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Change Basis to $\mathcal{B} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots\}$ That Can Squeeze & Clean

- $\mathbf{A} = \sum \sigma_i\mathbf{Z}_i$ (Fourier Expansion)
- $\mathcal{B}$ o.n. $\Rightarrow \sigma_i = \langle \mathbf{Z}_i | \mathbf{A} \rangle$ = amount of $\mathbf{A}$ in direction of $\mathbf{Z}_i$

Matrix Factorizations: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij}\mathbf{u}_i\mathbf{v}_j^T = \sum r_{ij}\mathbf{Z}_{ij}$

- Represent data with as few directions $\mathbf{Z}_i$ as possible

Back To IR

Almost the Same Problem

- Reveal hidden patterns & evaluate $\mathbf{q}^T\mathbf{A}$ fast (clean & compress)

Data is Now the Term-Doc Matrix in Standard Coordinates

- $\mathbf{A} = \sum_{i,j} a_{ij}\mathbf{E}_{ij} \qquad \mathbf{E}_{ij} = \mathbf{e}_i\mathbf{e}_j^T$

Change Basis to $\mathcal{B} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots\}$ That Can Squeeze & Clean

- $\mathbf{A} = \sum \sigma_i \mathbf{Z}_i$ (Fourier Expansion)
- $\mathcal{B}$ o.n. $\Rightarrow \sigma_i = \langle \mathbf{Z}_i | \mathbf{A} \rangle$ = amount of $\mathbf{A}$ in direction of $\mathbf{Z}_i$

Matrix Factorizations: $\mathbf{A} = \mathbf{URV}^T = \sum r_{ij}\mathbf{u}_i\mathbf{v}_j^T = \sum r_{ij}\mathbf{Z}_{ij}$

- Represent data with as few directions $\mathbf{Z}_i$ as possible
- SVD $\Rightarrow \mathbf{R} = \begin{bmatrix} \sigma_1 & & & & \\ & \ddots & & & \\ & & \sigma_r & & \\ & & & 0 & \\ & & & & \ddots \\ & & & & & 0 \end{bmatrix} \Rightarrow \quad \mathbf{A} = \sum_{i=1}^r \sigma_i \mathbf{Z}_i, \qquad \langle \mathbf{Z}_i | \mathbf{Z}_j \rangle = \begin{cases} 1 & i=j \\ 0 & i \neq j \end{cases}$

Same As Before

Assume Nondirectional Uncertainty

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace $\mathbf{A}$ with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace **A** with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace **A** with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance
- Lose larger proportion of uncertainty

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace $\mathbf{A}$ with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance
- Lose larger proportion of uncertainty

New Query Matching Strategy

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace $\mathbf{A}$ with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance
- Lose larger proportion of uncertainty

New Query Matching Strategy

- Normalize
  - $\mathbf{q} \leftarrow \mathbf{q} / \|\mathbf{q}\|$

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace $\mathbf{A}$ with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance
- Lose larger proportion of uncertainty

New Query Matching Strategy

- Normalize
  - $\mathbf{q} \leftarrow \mathbf{q}/\|\mathbf{q}\|$
  - $\widetilde{\mathbf{A}} \leftarrow \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \mathbf{D} = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \widetilde{\mathbf{v}}_i^T$

Same As Before

Assume Nondirectional Uncertainty

- Drop small σ_i's — replace $\mathbf{A}$ with $\widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i \mathbf{Z}_i$
- Lose only small part of relevance
- Lose larger proportion of uncertainty

New Query Matching Strategy

- Normalize
  - $\mathbf{q} \leftarrow \mathbf{q}/\|\mathbf{q}\|$
  - $\widetilde{\mathbf{A}} \leftarrow \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \mathbf{D} = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \widetilde{\mathbf{v}}_i^T$
- Compare query to each document
  - $\mathbf{q}^T \widetilde{\mathbf{A}} = \sum_{i=1}^{k} \sigma_i (\mathbf{q}^T \mathbf{u}_i) \widetilde{\mathbf{v}}_i^T = (\delta_1, \delta_2, \ldots, \delta_n)$

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*

Pros & Cons

Advantages

- Compression
  - $\mathbf{A}$ replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T\widetilde{\mathbf{A}}_{m\times n} = \sum_{i=1}^{k} \sigma_i(\mathbf{q}^T\mathbf{u}_i)\widetilde{\mathbf{v}}_i^T$$

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T\widetilde{\mathbf{A}}_{m\times n} = \sum_{i=1}^{k} \sigma_i(\mathbf{q}^T\mathbf{u}_i)\widetilde{\mathbf{v}}_i^T$$

- Latent semantic associations are made
  - Relevant docs not found by direct matching show up

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T \widetilde{\mathbf{A}}_{m \times n} = \sum_{i=1}^{k} \sigma_i (\mathbf{q}^T \mathbf{u}_i) \widetilde{\mathbf{v}}_i^T$$

- Latent semantic associations are made
  - Relevant docs not found by direct matching show up
  - *Latent Semantic Indexing* (LSI)

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T\widetilde{\mathbf{A}}_{m\times n} = \sum_{i=1}^{k} \sigma_i(\mathbf{q}^T\mathbf{u}_i)\widetilde{\mathbf{v}}_i^T$$

- Latent semantic associations are made
  - Relevant docs not found by direct matching show up
  - *Latent Semantic Indexing* (LSI)

Disadvantages

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T \widetilde{\mathbf{A}}_{m \times n} = \sum_{i=1}^{k} \sigma_i (\mathbf{q}^T \mathbf{u}_i) \widetilde{\mathbf{v}}_i^T$$

- Latent semantic associations are made
  - Relevant docs not found by direct matching show up
  - *Latent Semantic Indexing* (LSI)

Disadvantages

- Adding & deleting docs requires updating & downdating SVD

Pros & Cons

Advantages

- Compression
  - **A** replaced with a few sing values & vectors (but dense)
  - They are determined & normalized only once
- *SPEED!*
  - Each query requires only a few inner products

$$\mathbf{q}^T\widetilde{\mathbf{A}}_{m\times n} = \sum_{i=1}^{k} \sigma_i(\mathbf{q}^T\mathbf{u}_i)\widetilde{\mathbf{v}}_i^T$$

- Latent semantic associations are made
  - Relevant docs not found by direct matching show up
  - *Latent Semantic Indexing* (LSI)

Disadvantages

- Adding & deleting docs requires updating & downdating SVD
- Determining optimal k is not easy (empirical tuning required)

Other Fourier Expansions ??

Other Fourier Expansions ??

Truncated URV Factorizations

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\qquad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\quad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \quad \Rightarrow \quad \mathbf{H}_n\mathbf{x}$ is *Fast!* (if $n = 2^p$)

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\quad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \Rightarrow \mathbf{H}_n\mathbf{x}$ is $\mathit{Fast!}$ (if n=2^p)
- Factor $\mathbf{A} = \mathbf{H}_m\mathbf{B}\mathbf{H}_n^T = \sum_{i,j} \beta_{ij}\mathbf{h}_i\mathbf{h}_j^T$ (**h**’s only use -1, 0, 1)

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\qquad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \quad \Rightarrow \quad \mathbf{H}_n\mathbf{x}$ is *Fast!* (if $n=2^p$)

- Factor $\mathbf{A} = \mathbf{H}_m \mathbf{B} \mathbf{H}_n^T = \sum_{i,j} \beta_{ij} \mathbf{h}_i \mathbf{h}_j^T$ (**h**'s only use -1, 0, 1)

  — More than a few β_{ij}'s may be needed

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\quad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \Rightarrow \mathbf{H}_n\mathbf{x}$ is *Fast!* (if $n=2^p$)

- Factor $\mathbf{A} = \mathbf{H}_m\mathbf{B}\mathbf{H}_n^T = \sum_{i,j} \beta_{ij}\mathbf{h}_i\mathbf{h}_j^T$ (**h**'s only use -1, 0, 1)

  — More than a few β_{ij}'s may be needed

  — Needs padding if m or n not a power of 2

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\quad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \Rightarrow \mathbf{H}_n\mathbf{x}$ is *Fast!* (if $n=2^p$)
- Factor $\mathbf{A} = \mathbf{H}_m\mathbf{B}\mathbf{H}_n^T = \sum_{i,j} \beta_{ij}\mathbf{h}_i\mathbf{h}_j^T$ (**h**'s only use -1, 0, 1)

— More than a few β_{ij}'s may be needed

— Needs padding if m or n not a power of 2

Semidiscrete Decomposition

(T. Kolda and D. O'Leary, 1998)

- Approximate $\mathbf{A} \approx \sum_{i=1}^{k} \alpha_i \mathbf{x}_i \mathbf{y}_j$ $\quad \mathbf{x}_i$ and $\mathbf{y}_j$ only use -1, 0, or 1

Other Fourier Expansions ??

Truncated URV Factorizations

DFT — FFT

- No compression — no oscillatory components

Haar Transform $\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad \mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$

- $\mathbf{H}_n = (\mathbf{I}_2 \otimes \mathbf{H}_{n/2})\mathbf{P}_n \begin{bmatrix} \mathbf{H}_{n/2} & \\ & \mathbf{I}_{n/2} \end{bmatrix} \quad \Rightarrow \quad \mathbf{H}_n\mathbf{x}$ is *Fast!* (if $n=2^p$)
- Factor $\mathbf{A} = \mathbf{H}_m\mathbf{B}\mathbf{H}_n^T = \sum_{i,j} \beta_{ij}\mathbf{h}_i\mathbf{h}_j^T$ (**h**'s only use -1, 0, 1)

  — More than a few β_{ij}'s may be needed

  — Needs padding if m or n not a power of 2

Semidiscrete Decomposition

(T. Kolda and D. O'Leary, 1998)

- Approximate $\mathbf{A} \approx \sum_{i=1}^{k} \alpha_i\mathbf{x}_i\mathbf{y}_j$ $\quad \mathbf{x}_i$ and $\mathbf{y}_j$ only use -1, 0, or 1

Other Wavelet Transforms?

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing

- Inverted file structure
  - $Term_{\mathbf{1}} \rightarrow P_i, P_j, \ldots$

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing
- Inverted file structure
  - $Term_{\mathbf{1}} \rightarrow P_i, P_j, \ldots$
  - $Term_{\mathbf{2}} \rightarrow P_k, P_l, \ldots$

$$\vdots$$

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing

- Inverted file structure
  - $Term_{\mathbf{1}} \rightarrow P_i, P_j, \ldots$
  - $Term_{\mathbf{2}} \rightarrow P_k, P_l, \ldots$

$$\vdots$$

- Attach an importance rating to $P_i, P_j, P_k, P_l, \ldots$

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing
- Inverted file structure
  - $Term_{\mathbf{1}} \rightarrow P_i, P_j, \ldots$
  - $Term_{\mathbf{2}} \rightarrow P_k, P_l, \ldots$

    $\vdots$
- Attach an importance rating to $P_i, P_j, P_k, P_l, \ldots$
- Direct query matching
  - Q = $Term_{\mathbf{1}}, Term_{\mathbf{2}}, \ldots$ produces $P_i, P_j, P_k, P_l, \ldots$

Link Analysis (Think Web)

How To Take Advantage of Link Structure ?

Indexing and Ranking

- Still must index key terms on each page
  - Robots crawl the web — software does indexing

- Inverted file structure
  - $Term_{\mathbf{1}} \rightarrow P_i, P_j, \ldots$
  - $Term_{\mathbf{2}} \rightarrow P_k, P_l, \ldots$

    $\vdots$

- Attach an importance rating to $P_i, P_j, P_k, P_l, \ldots$

- Direct query matching
  - $Q = Term_{\mathbf{1}}, Term_{\mathbf{2}}, \ldots$ produces $P_i, P_j, P_k, P_l, \ldots$

- Return $P_i, P_j, P_k, P_l, \ldots$ to user in order of importance

How To Measure “Importance”

How To Measure "Importance"

Hubs & Authorities

(Jon Kleinberg 1998)

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs

How To Measure “Importance”

Hubs & Authorities

(Jon Kleinberg 1998)

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs

HITS Algorithm

- For each query a “neighborhood graph” N is built

How To Measure "Importance"

Hubs & Authorities

(Jon Kleinberg 1998)

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs

HITS Algorithm

- For each query a "neighborhood graph" N is built
- Hub and authority scores for nodes in N are computed
  - Eigenvector computation

How To Measure "Importance"

Hubs & Authorities

(Jon Kleinberg 1998)

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs

HITS Algorithm

- For each query a "neighborhood graph" N is built
- Hub and authority scores for nodes in N are computed
  - Eigenvector computation
- Works, but requires new graph for each query

How To Measure "Importance"

Hubs & Authorities

(Jon Kleinberg 1998)

- Good hub pages point to good authority pages
- Good authorities are pointed to by good hubs

HITS Algorithm

- For each query a "neighborhood graph" N is built
- Hub and authority scores for nodes in N are computed
  - Eigenvector computation
- Works, but requires new graph for each query
- Similar ideas in TEOMA.com

Google’s Idea

PageRank (Sergey Brin & Lawrence Page 1998)

Google's Idea

PageRank (Sergey Brin & Lawrence Page 1998)

- Your page P has some rank $r(P)$

Google's Idea

PageRank (Sergey Brin & Lawrence Page 1998)

- Your page P has some rank $r(P)$
- Adjust $r(P)$ higher or lower depending on ranks of pages that point to P

Google's Idea

PageRank

(Sergey Brin & Lawrence Page 1998)

- Your page P has some rank $r(P)$
- Adjust $r(P)$ higher or lower depending on ranks of pages that point to P
- Importance is not number of in-links or out-links

Google's Idea

PageRank (Sergey Brin & Lawrence Page 1998)

- Your page P has some rank $r(P)$
- Adjust $r(P)$ higher or lower depending on ranks of pages that point to P
- Importance is not number of in-links or out-links
  - One link to P from Yahoo! is important
  - Many links to P from me is not

Google's Idea

PageRank (Sergey Brin & Lawrence Page 1998)

- Your page P has some rank $r(P)$
- Adjust $r(P)$ higher or lower depending on ranks of pages that point to P
- Importance is not number of in-links or out-links
  - One link to P from Yahoo! is important
  - Many links to P from me is not
- But if Yahoo! points to many places, the value of the link to P is diluted

PageRank

The Definition

- $r(P) = \sum_{P \in \mathcal{B}_P} \frac{r(P)}{|P|}$
  - $\mathcal{B}_P$ = $\{$all pages pointing to $P\}$
  - $|P|$ = number of out links $from$ P

PageRank

The Definition

- $r(P) = \sum_{P \in \mathcal{B}_P} \frac{r(P)}{|P|}$

— $\mathcal{B}_P = \{$all pages pointing to $P\}$

— $|P|$ = number of out links *from* P

Successive Refinement

- Start with $r_0(P_i) = 1/n$ for all pages $P_1, P_2, \ldots, P_n$
- Iteratively refine rankings for each page

PageRank

The Definition

- $r(P) = \sum_{P \in \mathcal{B}_P} \frac{r(P)}{|P|}$
  - $\mathcal{B}_P = \{$all pages pointing to $P\}$
  - $|P|$ = number of out links *from* P

Successive Refinement

- Start with $r_0(P_i) = 1/n$ for all pages $P_1, P_2, \ldots, P_n$
- Iteratively refine rankings for each page
  - $r_1(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_0(P)}{|P|}$

PageRank

The Definition

- $r(P) = \sum_{P \in \mathcal{B}_P} \dfrac{r(P)}{|P|}$
  - $\mathcal{B}_P$ = $\{$all pages pointing to $P\}$
  - $|P|$ = number of out links *from* P

Successive Refinement

- Start with $r_0(P_i) = 1/n$ for all pages $P_1, P_2, \ldots, P_n$
- Iteratively refine rankings for each page
  - $r_1(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \dfrac{r_0(P)}{|P|}$
  - $r_2(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \dfrac{r_1(P)}{|P|}$

PageRank

The Definition

- $r(P) = \sum_{P \in \mathcal{B}_P} \frac{r(P)}{|P|}$
  - $\mathcal{B}_P = \{\text{all pages pointing to } P\}$
  - $|P| = \text{number of out links } \mathit{from} \ P$

Successive Refinement

- Start with $r_0(P_i) = 1/n$ for all pages $P_1, P_2, \ldots, P_n$
- Iteratively refine rankings for each page
  - $r_1(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_0(P)}{|P|}$
  - $r_2(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_1(P)}{|P|}$

$$\ddots$$

  - $r_{j+1}(P_i) = \sum_{P \in \mathcal{B}_{P_i}} \frac{r_j(P)}{|P|}$

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n)\right]$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ where $p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\, r_j(P_2),\, \cdots,\, r_j(P_n) \right]$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ where $p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$
- PageRank $= \lim_{j \to \infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$ (provided limit exists)

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ where $p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$
- PageRank $= \lim_{j \to \infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$ (provided limit exists)

It's A Markov Chain

- $\mathbf{P} = \left[p_{ij} \right]$ is a stochastic matrix (row sums = 1)

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n) \right]$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ where $p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \to j \\ 0 & \text{otherwise} \end{cases}$
- PageRank = $\lim_{j \to \infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$ (provided limit exists)

It's A Markov Chain

- $\mathbf{P} = \left[p_{ij} \right]$ is a stochastic matrix (row sums = 1)
- Each $\boldsymbol{\pi}_j^T$ (and $\boldsymbol{\pi}^T$) is a probability vector $\left(\sum_i r_j(P_i) = 1 \right)$

In Matrix Notation

After Step j

- $\boldsymbol{\pi}_j^T = \left[r_j(P_1),\ r_j(P_2),\ \cdots,\ r_j(P_n)\right]$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ where $p_{ij} = \begin{cases} 1/|P_i| & \text{if } i \rightarrow j \\ 0 & \text{otherwise} \end{cases}$
- PageRank $= \lim_{j\rightarrow\infty} \boldsymbol{\pi}_j^T = \boldsymbol{\pi}^T$ (provided limit exists)

It's A Markov Chain

- $\mathbf{P} = \left[p_{ij}\right]$ is a stochastic matrix (row sums = 1)
- Each $\boldsymbol{\pi}_j^T$ (and $\boldsymbol{\pi}^T$) is a probability vector $\left(\sum_i r_j(P_i) = 1\right)$
- $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ is random walk on the graph defined by links

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - No convergence!

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - No convergence!
- Could get trapped into a cycle $(P_i \rightarrow P_j \rightarrow P_i)$

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - No convergence!
- Could get trapped into a cycle $(P_i \rightarrow P_j \rightarrow P_i)$
  - No convergence!

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - No convergence!
- Could get trapped into a cycle $(P_i \to P_j \to P_i)$
  - No convergence!

Convergence

- Markov chain must be irreducible and aperiodic

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  — No convergence!
- Could get trapped into a cycle $(P_i \to P_j \to P_i)$
  — No convergence!

Convergence

- Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - — No convergence!
- Could get trapped into a cycle $(P_i \rightarrow P_j \rightarrow P_i)$
  - — No convergence!

Convergence

- Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

- Replace $\mathbf{P}$ by $\widetilde{\mathbf{P}} = \alpha\mathbf{P} + (1-\alpha)\mathbf{E}$ where $e_{ij} = 1/n$ $\quad \alpha \approx .85$

Random Surfer

Web Surfer Randomly Clicks On Links

(Back button not a link)

- Long-run proportion of time on page P_i is $\boldsymbol{\pi}_i$

Problems

- Dead end page (nothing to click on)
  - No convergence!
- Could get trapped into a cycle $(P_i \to P_j \to P_i)$
  - No convergence!

Convergence

- Markov chain must be irreducible and aperiodic

Bored Surfer Enters Random URL

- Replace $\mathbf{P}$ by $\widetilde{\mathbf{P}} = \alpha\mathbf{P} + (1-\alpha)\mathbf{E}$ where $e_{ij} = 1/n$ $\alpha \approx .85$
  - Different $\mathbf{E}$'s and α's allow customization & speedup

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T \mathbf{P}$ (stationary distribution vector)

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T \mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = 0$ (too big for direct solves)

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = \mathbf{0}$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = 0$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Updating Is A Big Problem

- Link structure of web is extremely dynamic

Computing $\boldsymbol{\pi}^T$

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T \mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = \mathbf{0}$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T \mathbf{P}$ (power method)

Updating Is A Big Problem

- Link structure of web is extremely dynamic
  - Links on CNN point to different pages every day (hour)

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = \mathbf{0}$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Updating Is A Big Problem

- Link structure of web is extremely dynamic
  - Links on CNN point to different pages every day (hour)
  - Links are added and deleted every sec (milli-sec?)

Computing π^T

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = \mathbf{0}$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Updating Is A Big Problem

- Link structure of web is extremely dynamic
  - Links on CNN point to different pages every day (hour)
  - Links are added and deleted every sec (milli-sec?)
- Google says every 3 to 4 weeks just start from scratch

Computing $\boldsymbol{\pi}^T$

World's Largest Eigenvector Problem (C. Moler)

- Solve $\boldsymbol{\pi}^T = \boldsymbol{\pi}^T\mathbf{P}$ (stationary distribution vector)
- $\boldsymbol{\pi}^T(\mathbf{I} - \mathbf{P}) = \mathbf{0}$ (too big for direct solves)
- Start with $\boldsymbol{\pi}_0^T = \mathbf{e}/n$ and iterate $\boldsymbol{\pi}_{j+1}^T = \boldsymbol{\pi}_j^T\mathbf{P}$ (power method)

Updating Is A Big Problem

- Link structure of web is extremely dynamic
  - Links on CNN point to different pages every day (hour)
  - Links are added and deleted every sec (milli-sec?)
- Google says every 3 to 4 weeks just start from scratch
- Old results don't help to restart (even if size doesn't change)
  - Cutoff phenomenon in random walks (P. Diaconis, 1996)

Report Card

FEATURES	LSI	LINK ANALYSIS

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns		

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed		

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update		

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)
Scales Up		

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)
Scales Up	D^{-}	

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^-	A^+
Easy To Update	D	F (?↑?)
Scales Up	D^-	A

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)
Scales Up	D^{-}	A
Takes Advantage of Link Structure		

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)
Scales Up	D^{-}	A
Takes Advantage of Link Structure	F	

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^{-}	A^{+}
Easy To Update	D	F (?↑?)
Scales Up	D^{-}	A
Takes Advantage of Link Structure	F	A^{+}

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^-	A^+
Easy To Update	D	F (?↑?)
Scales Up	D^-	A
Takes Advantage of Link Structure	F	A^+

Goals

- Do better job using link structure to reveal hidden connections

Report Card

FEATURES	LSI	LINK ANALYSIS
Reveals Hidden Patterns	A	C
Speed	B^-	A^+
Easy To Update	D	F (?↑?)
Scales Up	D^-	A
Takes Advantage of Link Structure	F	A^+

Goals

- Do better job using link structure to reveal hidden connections
- Improve updating

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the $``distance"$ from P_i to P_j ?

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the $``distance"$ from P_i to P_j ?
    - Link structure $\Longrightarrow \delta_{ij} \neq \delta_{ji}$

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the *"distance"* from P_i to P_j ?
    - Link structure $\Longrightarrow \delta_{ij} \neq \delta_{ji}$

1. Compute the distance δ_{ij} from P_i to P_j for all i, j
   - Keep only those for which $\delta_{ij} < \tau$ (provides sparsity)

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the *"distance"* from P_i to P_j ?
    - Link structure $\Longrightarrow \delta_{ij} \neq \delta_{ji}$

1. Compute the distance δ_{ij} from P_i to P_j for all i, j
   - Keep only those for which $\delta_{ij} < \tau$ (provides sparsity)
   - File structure: $\begin{cases} P_1 \rightarrow P_i, P_j, \ldots \\ P_2 \rightarrow P_k, P_l, \ldots \\ \quad\vdots \end{cases}$

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the *"distance"* from P_i to P_j ?
    - Link structure $\Longrightarrow \delta_{ij} \neq \delta_{ji}$

1. Compute the distance δ_{ij} from P_i to P_j for all i, j
   - Keep only those for which $\delta_{ij} < \tau$ (provides sparsity)
   - File structure: $\begin{cases} P_1 \to P_i, P_j, \ldots \\ P_2 \to P_k, P_l, \ldots \\ \qquad \vdots \end{cases}$
2. Match query most relevant page(s) $\mathcal{P}$
   - LSI — Link analysis — You pick

Hybrid Approach

The Idea

- Use link structure to define measure of page (doc) contiguity
  - What's the *"distance"* from P_i to P_j ?
    - Link structure $\Longrightarrow \delta_{ij} \neq \delta_{ji}$

1. Compute the distance δ_{ij} from P_i to P_j for all i, j
   - Keep only those for which $\delta_{ij} < \tau$ (provides sparsity)
   - File structure: $\begin{cases} P_1 \rightarrow P_i, P_j, \ldots \\ P_2 \rightarrow P_k, P_l, \ldots \\ \quad\quad \vdots \end{cases}$
2. Match query most relevant page(s) $\mathcal{P}$
   - LSI — Link analysis — You pick
3. Return $\mathcal{P}$ together with those $\mathcal{P} \rightarrow P_i, P_j, P_k, P_l, \ldots$

Distance

What's the $\textit{``distance''}$ **from** D_i **to** D_j **?**

Distance

What's the $\textit{"distance"}$ **from** D_i **to** D_j **?**

- LSI uses $\delta_{ij} = \cos\theta_{ij} = \delta_{ji}$

Distance

What's the $\textit{“distance”}$ **from** D_i **to** D_j **?**

- LSI uses $\delta_{ij} = \cos\theta_{ij} = \delta_{ji}$

$\left\{ \begin{array}{l} \text{Based only on term frequencies} \\ \text{No link structure} \end{array} \right.$

Distance

What's the $\text{"}\mathit{distance}\text{"}$ **from** D_i **to** D_j **?**

- LSI uses $\delta_{ij} = \cos\theta_{ij} = \delta_{ji}$ $\left\{\begin{array}{l}\text{Based only on term frequencies}\\ \text{No link structure}\end{array}\right.$

Directed Link Structure $\Longrightarrow$ **Nonsymmetric Metric**