Data Mining the Internet

Part B: HOW TO FIND MORE

C. Faloutsos

High-level Outline

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws

Table Overview

<table>
<thead>
<tr>
<th></th>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Powerlaws, jellyfish</td>
<td>Growth pattern, Compare graphs</td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td>LRD, ON/OFF sources</td>
<td>Effect of topology and protocols</td>
<td></td>
</tr>
<tr>
<td>End-2-end</td>
<td>LRD loss and RTT</td>
<td>Troubleshoot, cluster and predict</td>
<td></td>
</tr>
<tr>
<td>Traffic</td>
<td>Skewness of location</td>
<td>Comprehensive model, troubleshoot</td>
<td></td>
</tr>
</tbody>
</table>

Table Overview

<table>
<thead>
<tr>
<th></th>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Powerlaws, jellyfish</td>
<td>Growth pattern, Compare graphs</td>
<td>SVD, fractals</td>
</tr>
<tr>
<td>Link</td>
<td>LRD, ON/OFF sources</td>
<td>Effect of topology and protocols</td>
<td>ARIMA, wavelets</td>
</tr>
<tr>
<td>End-2-end</td>
<td>LRD loss and RTT</td>
<td>Troubleshoot, cluster and predict</td>
<td>ARIMA, wavelets</td>
</tr>
<tr>
<td>Traffic</td>
<td>Skewness of location</td>
<td>Comprehensive model, troubleshoot</td>
<td>Power-laws; multifractals, clustering</td>
</tr>
</tbody>
</table>

Problem

Given: (multiple) data sources
Find: patterns (classifiers, rules, clusters, outliers...)

\[ \text{traffic(link-id, timestamp, #packets)} \]

\[ \text{Link-info(link-id, bandwidth, ...)} \]

\[ \text{??} \]
Problem 1: classification

- Eg. Given profiles of ‘good’ and ‘bad’ customers (clients, links, …)
- Classify the current customer (client, link, …)

Problem 2: clustering

- Eg. Given profiles of several customers (clients, links, …)
- Group them into ‘natural’ groups

Problem 3: Association Rules

- Given a sequence of events (eg., ‘server-A comes up’, ‘server-B goes down’, …)
- Find events that occur together too often, eg.,
  - server-A-up, server-B-down -> server-C-down

Decision trees - Problem

<table>
<thead>
<tr>
<th>Avg. packet size</th>
<th>Avg. arrival rate</th>
<th>time</th>
<th>CLASS ID</th>
<th>??</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>150</td>
<td>13:30</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Decision trees

- Pictorially, we have

  num. attr#2
  (eg., avg rate)

  num. attr#1 (eg., ‘avg size’)
**Decision trees**

- and we want to label ‘?’

```
<table>
<thead>
<tr>
<th>num. attr#2 (eg., avg rate)</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>num. attr#1 (eg., ‘avg size’)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
</tr>
</tbody>
</table>
```

- so we build a decision tree:

```
<table>
<thead>
<tr>
<th>num. attr#2 (eg., avg rate)</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
</tbody>
</table>
```

- 40 avg size

```
<table>
<thead>
<tr>
<th>avg rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>num. attr#1 (eg., ‘avg size’)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
</tr>
</tbody>
</table>
```

- 50 avg size

```
<table>
<thead>
<tr>
<th>avg size&lt;50</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 rate&lt;40</td>
</tr>
<tr>
<td>Y</td>
</tr>
<tr>
<td>N</td>
</tr>
</tbody>
</table>
```

- 50 avg size

```
<table>
<thead>
<tr>
<th>avg rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
</tbody>
</table>
```

**Conclusions -Practitioner’s guide:**

- Many available implementations
  - eg. C4.5 (freeware), C5.0
  - Also, inside larger stat. packages
- They usually hide all the details from us:
  - training / testing / tree pruning
  - ‘boosting’
  - recent, scalable methods
  - see [Han+Kamber] for details

**High-level Outline**

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws
**Problem 2: clustering**

- Eg. Given profiles of several customers (clients, links, …)
- group them into ‘natural’ groups
- (and, optionally, report misfits as ‘outliers’)

---

**Cluster generation**

- Problem:
  - given N points in D dimensions,
  - group them

Short version:
- There are *numerous* clustering algorithms, available in free / open / commercial systems (eg., Splus, ‘R’ system)
- BUT: most algorithms require # of clusters and/or don’t scale up for large datasets
  - except for recent solutions...

---

**B.I - Traditional D.M. - Outline**

- Motivating Problems
- Supervised learning: decision trees
- Unsupervised learning: clustering
  - preliminaries
  - ‘sound’ methods
  - ‘iterative’ methods
- Unsupervised learning: association rules
- Conclusions - practitioner’s guide
Cluster generation

A: *many-many* algorithms - in two groups [VanRijsbergen]:
- theoretically sound (O(N^2))
  - independent of the insertion order
- iterative (O(N), O(N log(N))

Cluster generation - ‘sound’ methods

- Approach#1: dendrograms - create a hierarchy (bottom up or top-down) - choose a cut-off (how?) and cut

```
ucb.edu
\ --- \ \ --- \ \ --- \ \ --- 
mit.edu ibm.com att.com
```

- Approach#2: min. some statistical criterion (e.g., sum of squares from cluster centers)
  - like ‘k-means’

- Approach#3: Graph theoretic [Zahn]:
  - build MST;
  - delete edges longer than 2.5* std of the local average

Cluster generation - ‘sound’ methods

- Approach#2: min. some statistical criterion (e.g., sum of squares from cluster centers)
  - like ‘k-means’
  - but how to decide ‘k’?

Cluster generation - ‘sound’ methods

- Result:
  - why ‘2.5’?
B.I - Traditional D.M. - Outline

- Motivating Problems
- Supervised learning: decision trees
- Unsupervised learning: clustering
  - preliminaries
  - 'sound' methods
  - 'iterative' methods
- Unsupervised learning: association rules
- Conclusions - practitioner’s guide

Cluster generation - ‘iterative’ methods

general outline:
- Choose ‘seeds’ (how?)
- assign each vector to its closest seed
  (possibly adjusting cluster centroid)
- possibly, re-assign some vectors to improve clusters
Fast and practical, but ‘unpredictable’

Cluster generation - ‘iterative’ methods

Many, recent, fast methods [see book by Han+Kamber]:
- BIRCH
- CURE
- CHAMELEON
- WaveCluster
- ...

Cluster generation- how many clusters?

- one way to estimate # of clusters $k$: X-means method [Moore+Pelleg]
- in general: AIC or BIC/MDL (= minimize not only error, but also model complexity, i.e.: $RMSE + C \cdot k$ )
  - BIC: Bayesian Information Criterion
  - AIC: Akaike Inf. Criterion
  - MDL: minimum description language

Conclusions - Practitioner’s guide

- Many clustering methods
- Many available implementations (BIRCH is free; all stat. packages include several versions of clustering algorithms)
- Usually need a ‘magic number’ (eg., # of clusters)

High-level Outline

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws
B.I - Traditional D.M. - Outline

- Motivating Problems
- Supervised learning: decision trees
- Unsupervised learning: clustering
- Unsupervised learning: association rules
- Conclusions - practitioner’s guide

Problem 3; Association rules

[Mannila+97]
- Given a stream of telecommunication events
- Find rules of the form
  \( A.A.B \rightarrow C \)
  (within windows of 5')

\[ \text{Eg: } A \quad B \quad C \quad \text{time} \]

Association rules - idea

[Agrawal+SIGMOD93]
- Consider ‘market basket’ case:
  (milk, bread)
  (milk, bread, chocolate)
  (milk, chocolate)
  ...
  (milk, bread)
- Find ‘interesting things’, eg., rules of the form:
  milk, bread -> chocolate

Association rules - example

INPUT:
- milk, bread
- milk, bread, chocolate
- milk, chocolate
- milk, bread

Sample rule:
- milk, bread -> chocolate
  - confidence: 33%
  - support: 25%

- confidence: how often people by chocolate, given that they have bought milk and bread
- support: how often people buy bread, milk and chocolate

Association rules - problem dfn

Problem definition:
- given
  - a set of ‘market baskets’ (=binary matrix, of N rows/baskets and M columns/products)
  - min-support ‘s’ and
  - min-confidence ‘c’
- find
  - all the rules with higher support and confidence

Association rules

Association rules:
- Do NOT need the user to give ‘hypotheses’
- because they discover automatically frequent items, pairs, triplets, ...
- They solve the problem, QUICKLY! (a few passes over the dataset)
  - ‘A priori’ algorithm of Agrawal+
  - faster algorithms (FP-trees - see [Han+Kamber])
**Association rules - Conclusions**

Association rules: a new tool to find patterns
- easy to understand its output
- fine-tuned algorithms exist
- Many available implementations
  - IBM (IntelligentMiner)
  - Stand-alone ones

**Overall Conclusions**

- Many, mature (and often, free!) tools for classification, clustering, and association rules

**Table Overview**

<table>
<thead>
<tr>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td></td>
<td></td>
</tr>
<tr>
<td>End-2-end</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Traffic Matrix</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Resources - software & urls**

- Stat. Packages: SAS, Splus, ‘R’ (freeware!)
  - www.r-project.org/
  (all have SVD, ARIMA, clustering etc)
- Data Mining ‘central’: Software, datasets, conference announcements
  - www.kdnuggets.com/

**Resources - Books**

- Data mining: Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2000.

**Additional Reading**

Additional Reading


Additional reading


High-level Outline

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws

Part B.II: Time series, Fourier, wavelets and forecasting

B.II - Time Series Analysis - Outline

- Motivating problems
- DFT
- DWT
- AR(IMA) and forecasting

Problem #1:

Goal: given a signal (eg., #packets over time)
Find: patterns, periodicities, and/or compress

count

year

lynx caught per year
(packets per day; virus infections per month)
Problem #2: Forecast
Given $x_1, x_2, \ldots, x_{n+1}$, forecast $x_{n+1}$

![Graph showing time series analysis](image)

Problem #3:
- Given a set of correlated time sequences
- Forecast ‘Sent(t)’

![Graph showing time series analysis](image)

B.II - Time Series Analysis - Outline
- DFT
  - Definition of DFT and properties
  - How to read the DFT spectrum
- DWT
- AR(IMA) and forecasting

Recall from Part A:
UCR-CMU RTTs showed periodicity!

![Graph showing RTTs](image)

Introduction - definitions
Goal: given a signal (e.g., packets over time)
Find: patterns and/or compress

![Graph showing lynx catch](image)

What does DFT do?
A: highlights the periodicities
**DFT: definition**

- **(n-point) Discrete Fourier Transform:**
  \[
  X_f = \frac{1}{\sqrt{n}} \sum_{n=0}^{N-1} x_n \cdot \exp(-j2\pi f n / N) \quad f = 0, \ldots, n-1
  \]
  \[
  (j = \sqrt{-1})
  \]
  inverse DFT

- **Good news:** Available in all symbolic math packages, e.g., in `mathematica`
  
  \[
  x = [1,2,1,2];
  X = \text{Fourier}[x];
  \text{Plot[ Abs[X] ];}
  \]

**DFT: Amplitude spectrum**

Amplitude: \[
A_f = \text{Re}^2(X_f) + \text{Im}^2(X_f)
\]

**DFT: examples**

- Low frequency sinusoid
- Sinusoid - symmetry property: \[
X_f = X^*_n f
\]
DFT: examples

- Higher freq. sinusoid

DFT: examples

B.II - Time Series Analysis - Outline

- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
- AR(IMA) and forecasting

DFT: Amplitude spectrum

\[ A_j^2 = \text{Re}^2 (X_j) + \text{Im}^2 (X_j) \]

DFT: Amplitude spectrum
DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?
- **A1**: (lossy) compression
- **A2**: pattern discovery
- **A3**: forecasting

DFT - Conclusions

- It spots periodicities (with the ‘amplitude spectrum’)
- can be quickly computed (O(n log n)), thanks to the FFT algorithm.
- **standard** tool in signal processing (speech, image etc signals)
- (closely related to DCT and JPEG)
Problem #1:
Goal: given a signal (eg., #packets over time)
Find: patterns, periodicities, and/or compress

Outlier

Wavelets - DWT

- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

Wavelets - DWT

- Similarly, DFT suffers on short-duration waves (eg., baritone, silence, soprano)
Wavelets - DWT

- Solution#1: Short window Fourier transform (SWFT)
- But: how short should the window?

freq | value
--- | ---
| | |

Time domain

freq | DFT | SWFT | DWT
--- | --- | --- | ---
| | | |

Haar Wavelets

- subtract sum of left half from right half
- repeat recursively for quarters, eight-ths, ...

Daubechies etc Wavelets

- Many more wavelets (Daubechies-4, -6 etc; Coifman; …)

B.II - Time Series Analysis - Outline

- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Motivation - definitions
  - How to read the ‘scalogram’
- AR(IMA) and forecasting
Wavelets - Drill:

- Q: baritone/silence/soprano - DWT?

Wavelets - Drill:

- Q: baritone/soprano - DWT?

Wavelets - Drill:

- Q: spike - DWT?

Wavelets - Drill:

- Q: spike - DWT?

Wavelets - Drill#2:

- Q: weekly + daily periodicity, + spike - DWT?

Wavelets - Drill#2:

- Q: weekly + daily periodicity, + spike - DWT?
Wavelets - Drill#2:

- Q: weekly + daily periodicity, + spike - DWT?

Wavelets - Drill#2:

- Q: weekly + daily periodicity, + spike - DWT?

Advantages of Wavelets

- Better compression (better RMSE with same number of coefficients - used in JPEG-2000)
- Fast to compute (usually: $O(n)$)
- Very good for ‘spikes’
- (mammalian eye and ear: Gabor wavelets)
- Suitable for self-similar/LRD signals

Advantages of Wavelets

- Suitable for self-similar/LRD signals for fractional Gaussian Noise [Riedi+99]
  - $\text{var}(W_{jk}) \sim 2^{-j(2\theta-1)}$
  - And ~ Gaussian

\[ j=2 \]
\[ W_{jk} \]
\[ j=1 \]
Advantages of Wavelets

- suitable for self-similar/LRD signals for fractional Gaussian Noise [Riedi+99]
  - \( \text{var}(W_{i,j}) \sim 2^{j(2H-1)} \)
  - and ~ Gaussian
- \( H \): Hurst exponent \( (1/2 < H < 1) \)
- Fast generation of realistic LRD traffic

Overall Conclusions

- DFT ( & DCT) spot periodicities
- DWT: multi-resolution - matches processing of mammalian ear/eye better; very suitable for self-similar traffic
- DWT: used for summarization of streams [Gilbert+01]

Overall Conclusions - cont’ed

- All three: powerful tools for compression, pattern detection in real signals
- All three: included in math packages (matlab, mathematica, ... - DFT: even in spreadsheets!)

B.II - Time Series Analysis - Outline

- Motivating problems
- DFT
- DWT
  - AR(IMA) and forecasting

Forecasting

"Prediction is very difficult, especially about the future." - Nils Bohr
http://www.hfac.uh.edu/MediaFutures/thoughts.html

ARIMA - Outline

- Auto-regression: Least Squares; recursive least squares
- Co-evolving time sequences
- Examples
- Conclusions
Problem: Forecast

- Example: give \(x_{t-1}, x_{t-2}, \ldots\), forecast \(x_t\)

![Graph showing time ticks and data points](image)

Problem: Forecast

- Solution: try to express
  \(x_t\)
as a linear function of the past: \(x_{t-2}, x_{t-3}, \ldots\)
  (up to a window of \(w\))

Formally:

\[
x_t \approx a_1 x_{t-1} + \ldots + a_w x_{t-w} + \text{noise}
\]

![Graph showing time ticks and data points](image)

(Problem: Back-cast; interpolate)

- Solution - interpolate: try to express
  \(x_t\)
as a linear function of the past AND the future:
  \(x_{t+1}, x_{t+2}, \ldots\)
  (up to windows of \(w_{\text{past}}, w_{\text{future}}\))

- EXACTLY the same algo’s

![Graph showing time ticks and data points](image)

Linear Regression: idea

- express what we don’t know (= ‘dependent variable’)
- as a linear function of what we know (= ‘indep. variable(s)’)

![Table showing linear regression data](image)

Linear Auto Regression:

- Lag \(w=1\)
- Dependent variable = # of packets sent \((S[t])\)
- Independent variable = # of packets sent \((S[t-1])\)

![Graph showing number of packets sent](image)
B.II - Time Series Analysis - Outline

- Auto-regression
- Least Squares; recursive least squares
- Co-evolving time sequences
- Examples
- Conclusions

More details:

- Q1: Can it work with window \( w > 1 \)?
- A1: YES! (we’ll fit a hyper-plane, then!)

More details:

- Q1: Can it work with window \( w > 1 \)?
- A1: YES! (we’ll fit a hyper-plane, then!)

More details:

- Q1: Can it work with window \( w > 1 \)?
- A1: YES! The problem becomes:

\[
X_{[N \times w]} \times a_{[w \times 1]} = y_{[N \times 1]}
\]

- OVER-CONSTRAINED
  - \( a \) is the vector of the regression coefficients
  - \( X \) has the \( N \) values of the \( w \) indep. variables
  - \( y \) has the \( N \) values of the dependent variable
More details:
- \( \mathbf{X}_{[N \times W]} \times \mathbf{a}_{[W \times 1]} = \mathbf{y}_{[N \times 1]} \)

Ind-var1 \hspace{1cm} Ind-var-w

time \[ \begin{bmatrix} x_{11}, x_{12}, \ldots, x_{1w} \\ x_{11}, x_{12}, \ldots, x_{2w} \\ \vdots \\ \vdots \\ x_{N1}, x_{N2}, \ldots, x_{Nw} \end{bmatrix} \]

\[ \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_w \end{bmatrix} \]

\[ \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \]

Even more details
- Q2: How to estimate \( a_1, a_2, \ldots, a_w = \mathbf{a} \)?
- A2: with Least Squares fit
  \[ \mathbf{a} = (\mathbf{X}^T \times \mathbf{X})^{-1} \times (\mathbf{X}^T \times \mathbf{y}) \]
- (Moore-Penrose pseudo-inverse)
- \( \mathbf{a} \) is the vector that minimizes the RMSE from \( \mathbf{y} \)

Even more details
- Q3: Can we estimate \( \mathbf{a} \) incrementally?
- A3: Yes, with the brilliant, classic method of ‘Recursive Least Squares’ (RLS) (see, e.g., [Yi+00], for details) - pictorially:

Even more details
- RLS: quickly compute new best fit
Even more details

- Q4: can we ‘forget’ the older samples?
- A4: Yes - RLS can easily handle that

Adaptability - ‘forgetting’

- (R)LS: can *trivially* handle ‘forgetting’
- Examples

B.II - Time Series Analysis

Outline

- Auto-regression
- Least Squares; recursive least squares
- Co-evolving time sequences
  - Examples
  - Conclusions

Co-Evolving Time Sequences

- Given: A set of correlated time sequences
- Forecast ‘Repeated(t)’
Solution:

Q: what should we do?

Solution:

Least Squares, with

- Dep. Variable: Repeated(t)
- Indep. Variables: Sent(t-1) … Sent(t-w); Lost(t-1) … Lost(t-w); Repeated(t-1), ...
- (named: ‘MUSCLES’ [Yi+00])

Outline

- Auto-regression
- Least Squares; recursive least squares
- Co-evolving time sequences

Examples - Experiments

- Datasets
  - Modem pool traffic (14 modems, 1500 time-ticks; #packets per time unit)
  - AT&T WorldNet internet usage (several data streams; 980 time-ticks)
- Measures of success
  - Accuracy: Root Mean Square Error (RMSE)

Accuracy - “Modem”

MUSCLES outperforms AR & “yesterday”

Accuracy - “Internet”

MUSCLES consistently outperforms AR & “yesterday”
B.II - Time Series Analysis - Outline

- Auto-regression
- Least Squares; recursive least squares
- Co-evolving time sequences
- Examples
  - Conclusions

Conclusions - Practitioner’s guide

- AR(IMA) methodology: prevailing method for linear forecasting
- Brilliant method of Recursive Least Squares for fast, incremental estimation.
- See [Box-Jenkins]

Just a moment

Q: ARIMA - how about ‘I’ and ‘MA’?
A1: ‘I’ - Integration (actually, differentiation - apply AR to $\Delta x_t = x_t - x_{t-1}$)
A2: ‘MA’: Moving Average (see book by Box-Jenkins - also: ARFIMA for ‘F’ractal integration, GARFIMA etc)

Table Overview

<table>
<thead>
<tr>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Powerlaws, jellyfish</td>
<td>Growth pattern, Compare graphs</td>
</tr>
<tr>
<td>Link</td>
<td>LRD, ON/OFF sources</td>
<td>Effect of topology and protocol</td>
</tr>
<tr>
<td>End-2-end</td>
<td>LRD loss and RTT</td>
<td>Troubleshoot, cluster and predict</td>
</tr>
<tr>
<td>Traffic Matrix</td>
<td>Sheariness of location</td>
<td>Comprehensive model, troubleshoot</td>
</tr>
</tbody>
</table>

Resources - software and urls

- [http://www.dsptutor.ece.uiuc.edu/FTSpectrumAnalyser.html](http://www.dsptutor.ece.uiuc.edu/FTSpectrumAnalyser.html) : Nice Java applets for FFT
- [http://www.relisoft.com/freeware/freq.html](http://www.relisoft.com/freeware/freq.html) : voice frequency analyzer (needs microphone)

Resources: software and urls

- [xwpl]: open source wavelet package from Yale, with excellent GUI
- [http://monet.me.ic.ac.uk/people/gavin/java/ wavedept.html](http://monet.me.ic.ac.uk/people/gavin/java/wavedept.html) : wavelets and scalograms
- MUSCLES ([christos@cs.cmu.edu](mailto:christos@cs.cmu.edu))
Books


Additional Reading

- [Yi+00] Byung-Kee Yi et al.; *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000. (Describes MUSCLES and Recursive Least Squares)

Time for a break!

Data Mining the Internet

Part B: HOW TO FIND MORE

C. Faloutsos

Part B - III and IV new tools: SVD and fractals

High-level Outline

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws
B.III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
- Conclusions

SVD - Motivation

- problem #1: find patterns in a matrix
  - (e.g., traffic patterns from several IP-sources)
  - compression; dim. reduction
- problem #2: find most 'interesting' node in a graph (google/Kleinberg-style)

Problem #1

- ~10^6 rows; ~10^3 columns; no updates;
- Compress / find patterns

Problem #2

Given a graph, find its most interesting/central node

SVD - in short:

It gives the best hyperplane to project on

SVD - in short:

It gives the best hyperplane to project on
B.III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
- Conclusions

SVD - Definition

\[ A = U \Lambda V^T \] - example:

SVD - notation

Conventions:
- bold capitals -> matrix (e.g., \( A, U, \Lambda, V \))
- bold lower-case -> column vector (e.g., \( x, v_1, u_3 \))
- regular lower-case -> scalars (e.g., \( \lambda_1, \lambda_2 \))

SVD - Definition

\[ A_{[n \times m]} = U_{[n \times r]} \Lambda_{[r \times r]} (V_{[m \times r]})^T \]
- \( \Lambda \): \( n \times m \) matrix (e.g., \( n \) customers, \( m \) days)
- \( U \): \( n \times r \) matrix (\( n \) customers, \( r \) concepts)
- \( \Lambda \): \( r \times r \) diagonal matrix (strength of each ‘concept’) \( r \) : rank of the matrix
- \( V \): \( m \times r \) matrix (\( m \) days, \( r \) concepts)

SVD - Properties

**THEOREM** [Press+92]: always possible to decompose matrix \( A \) into \( A = U \Lambda V^T \), where
- \( U, \Lambda, V \): unique (*)
- \( U, V \): column orthonormal (i.e., columns are unit vectors, orthogonal to each other)
  - \( U^T U = I \), \( V^T V = I \) (I: identity matrix)
- \( \Lambda \): eigenvalues are positive, and sorted in decreasing order

SVD - example

- Customers; days; #packets

<table>
<thead>
<tr>
<th>Comm</th>
<th>Res</th>
<th>ETF</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>XYZ</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>GHI</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>JKL</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>MNO</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
B. III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
  - #1: customers, days, concepts
  - #2: best projection - dimensionality reduction
  - #3: fixed point
- Solutions to posed problems
- Conclusions

SVD - Example

- $A = U \Lambda V^T$ - example:

$$
\begin{bmatrix}
1 & 1 & 0 & 0 \\
2 & 2 & 0 & 0 \\
1 & 1 & 0 & 0 \\
5 & 5 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
0.18 & 0 \\
0.36 & 0 \\
0.18 & 0 \\
0.90 & 0
\end{bmatrix}
\begin{bmatrix}
9.64 & 0 \\
0 & 5.29
\end{bmatrix}
\begin{bmatrix}
0.58 & 0.58 \\
0.58 & 0 \\
0.58 & 0 \\
0 & 0.71
\end{bmatrix}
$$

SVD - Interpretation #1

- 'customers', 'days' and 'concepts'
- $U$: customer-to-concept similarity matrix
- $V$: day-to-concept sim. matrix
- $A$: its diagonal elements: 'strength' of each concept

(reminder)

- Customers; days; #packets

<table>
<thead>
<tr>
<th>Time</th>
<th>Packets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon</td>
<td>1</td>
</tr>
<tr>
<td>Tue</td>
<td>2</td>
</tr>
<tr>
<td>Wed</td>
<td>3</td>
</tr>
<tr>
<td>Thu</td>
<td>4</td>
</tr>
<tr>
<td>Fri</td>
<td>5</td>
</tr>
<tr>
<td>Sat</td>
<td>6</td>
</tr>
</tbody>
</table>

(continued)
**B.III - SVD - outline**

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
  - #1: customers, days, concepts
  - #2: best projection - dimensionality reduction
  - #3: fixed point
- Solutions to posed problems
- Conclusions
SVD - Interpretation #2

• best axis to project on: (‘best’ = min sum of squares of projection errors)

SVD - Interpretation #2

• $A = U \Lambda V^T$ - example:

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\times
\begin{bmatrix}
0.18 & 0.36 & 0.18 & 0.90 & 0 & 0.53 & 0 & 0.80 & 0
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 & 5.29 & 0.58 & 0.58 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
\]

SVD - Interpretation #2

• $A = U \Lambda V^T$ - example:

\[
\begin{bmatrix}
1 & 1 & 1 & 0 & 0 \\
2 & 2 & 2 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
5 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 2 & 2 \\
0 & 0 & 0 & 3 & 3 \\
0 & 0 & 0 & 1 & 1
\end{bmatrix}
\times
\begin{bmatrix}
0.18 & 0 & 0 & 0 & 0.36 & 0.18 & 0 & 0 & 0.90 & 0 & 0.53 & 0 & 0.80 & 0
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 & 5.29 & 0.58 & 0.58 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
\]

\[
\text{variance (‘spread’) on the v1 axis}
\]

\[
\begin{bmatrix}
0.18 & 0 & 0.36 & 0 & 0.18 & 0 & 0.90 & 0 & 0.53 & 0 & 0.80 & 0
\end{bmatrix}
\times
\begin{bmatrix}
9.64 & 0 & 5.29 & 0.58 & 0.58 & 0 & 0 & 0.71 & 0.71
\end{bmatrix}
\]

SVD - Interpretation #2

• minimum RMS error
SVD - interpretation #2

SVD gives best axis to project

- minimum RMS error

SVD, PCA and the v vectors

- how to ‘read’ the v vectors (= principal components)

SVD

- Recall: $A = U \Lambda V^T$ - example:

| 1 1 1 0 0 | 0.18 0 |
| 2 2 2 0 0 | 0.36 0 |
| 1 1 1 0 0 | 0.18 0 |
| 5 5 5 0 0 | 0.90 0 |
| 0 0 0 2 2 | 0 0.53 |
| 0 0 0 3 3 | 0 0.80 |
| 0 0 0 1 1 | 0 0.27 |

$x = \begin{bmatrix} 9.64 & 0 \\ 0 & 5.29 \end{bmatrix}$

- First Principal component $v_1$ -> weekdays are correlated positively
- similarly for $v_2$
- (we’ll see negative correlations later)

$SVD, PCA$ and the $v$ vectors

B.III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
  - #1: customers, days, concepts
  - #2: best projection - dimensionality reduction
- #3: fixed point
- Solutions to posed problems
- Conclusions

SVD - Interpretation #3

If $A$ is symmetric, $x$ is an eigenvector of $A$ if

$$Ax = \lambda x$$
**SVD - Interpretation #3**

- A as vector transformation (assume A is symmetric)

\[
\begin{bmatrix}
2 \\
1 
\end{bmatrix}
= \begin{bmatrix}
2 & 1 \\
1 & 3 
\end{bmatrix}
\begin{bmatrix}
2 \\
1 
\end{bmatrix}
\]

- For a symmetric A, by defn. its eigenvectors remain parallel to themselves (fixed points)

\[
\lambda_1 = 3.62, \quad v_1 = \begin{bmatrix}
0.52 \\
0.85 
\end{bmatrix}
\]

**SVD - Complexity**

- O(n*m) or O(n*n*m) (whichever is less)
- less work, if we just want eigenvalues
- ... or if we want first k eigenvectors
- ... or if the matrix is sparse [Berry]
- Implemented: in any linear algebra package (LINPACK, matlab, Splus, mathematica ...)

**SVD - conclusions so far**

- SVD: A = U \Lambda V^T : unique (*)
- U: row-to-concept similarities
- V: column-to-concept similarities
- \Lambda: strength of each concept

(*) see [Press+92]
**B.III - SVD - outline**
- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
  - P1: patterns in a matrix; compression
  - P2: most ‘important’ node in a graph
- Conclusions

**Problem #1 - specs**
- ~10**6** rows; ~10**3** columns; no updates;
- random access to any cell(s); small error: OK
- compress; find patterns / rules

**Idea**

**SVD to the rescue**
- space savings: 2:1
- minimum RMS error

**Compression - Performance**
- 3 pass algo (-> scalability)
- random cell(s) reconstruction
- 10:1 compression with < 2% error
- [Korn+, 97]

**Performance - scaleup**
B.III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
  - P1: patterns in a matrix; compression
  - P2: most 'important' node in a graph
- Conclusions

SVD & visualization:

- Visualization for free!
  - Time-plots are not enough:

SVD & visualization:

- Visualization for free!
  - Time-plots are not enough:

SVD & visualization:

- SVD: project 365-d vectors to best 2 dimensions, and plot:
  - no Gaussian clusters;
  - Zipf-like distribution

SVD and visualization

NBA dataset
- ~500 players;
- ~30 attributes
  (#games, #points, #rebounds, ...)

SVD and visualization

could be network dataset:
- ~N IP sources
- ~K attributes
  (#http bytes, #http packets)
Moreover, PCA/rules for free!

- SVD ~ PCA = Principal component analysis
- PCA: get eigenvectors v1, v2, ...
- ignore entries with small abs. value
- try to interpret the rest

### PCA & Rules

#### NBA dataset - V matrix (term to ‘concept’ similarities)

<table>
<thead>
<tr>
<th>field</th>
<th>RR1</th>
<th>RR2</th>
<th>RR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes played</td>
<td>-.608</td>
<td>-.486</td>
<td>-.602</td>
</tr>
<tr>
<td>field goals</td>
<td>.608</td>
<td>.198</td>
<td>.612</td>
</tr>
<tr>
<td>points</td>
<td>-.486</td>
<td>-.486</td>
<td>-.107</td>
</tr>
<tr>
<td>total rebounds</td>
<td>.612</td>
<td>.612</td>
<td>-.107</td>
</tr>
<tr>
<td>assists</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>steals</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### (Ratio) Rule#1: minutes:points = 2:1

- corresponding concept?

#### RR1: minutes:points = 2:1
- corresponding concept?
- A: ‘goodness’ of player
- (in a networks setting, could be ‘volume of traffic’ generated by this IP address)

#### RR2: points: rebounds negatively correlated(!)

- ignore entries with small abs. value

### PCA & Rules

- RR2: points: rebounds negatively correlated(!) - concept?
PCA & Rules

- RR2: points: rebounds negatively correlated(!) - concept?
- A: position: offensive/defensive
- (in a network setting, could be e-mailers versus gnutella-users)

Problem#2

Given a graph, find its most interesting/central node

Problem#2

Given a graph, find its most interesting/central node

Proposed solution: Random walk; spot most ‘popular’ node (→ steady state prob.)

google/page-rank algorithm

- Let $A$ be the transition matrix (= adjacency matrix); let $A^T$ become column-normalized = then

$$A^T \begin{bmatrix} 1 & 1 & 1/2 & 1/2 \\ 1 & 1 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1 & 1 \\ 1/2 & 1/2 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

B.III - SVD - outline

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
  - P1: patterns in a matrix; compression
  - P2: most ‘important’ node in a graph
- Conclusions

- Let $A$ be the transition matrix (= adjacency matrix); let $A^T$ become column-normalized = then

$$A^T \begin{bmatrix} 1 & 1 & 1/2 & 1/2 \\ 1 & 1 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1 & 1 \\ 1/2 & 1/2 & 1/2 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$
**google/page-rank algorithm**

- $\mathbf{A}^T \mathbf{p} = 1 \times \mathbf{p}$
- thus, $\mathbf{p}$ is the eigenvector that corresponds
to the highest eigenvalue ($=1$, since the matrix is
column-normalized)

**Kleinberg’s algorithm**

- Kleinberg’s algorithm of ‘hubs’ and
‘authorities’: closely related [Kleinberg’98]
- (and still based on SVD of the adjacency
matrix)

**Kleinberg’s algorithm - results**

Eg., for the query ‘java’:
- 0.328 www.gamelan.com
- 0.251 java.sun.com
- 0.190 www.digitalfocus.com (“the java
developer”)

**B.III - SVD - outline**

- Introduction - motivating problems
- Definition - properties
- Interpretation / Intuition
- Solutions to posed problems
  - P1: patterns in a matrix; compression
  - P2: most ‘important’ node in a graph
- Conclusions

**SVD - conclusions**

SVD: a valuable tool, whenever we have a
matrix, e.g.
- many time sequences
- many feature vectors
- graph (→ adjacency matrix)
SVD - conclusions

SVD: a valuable tool, whenever we have a matrix, e.g.

- many time sequences
  - SVD finds groups
  - principal components
  - dim. reduction

<table>
<thead>
<tr>
<th>IP address1</th>
<th>packets on day1</th>
<th>packets on day2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

SVD - conclusions

Has been used/re-invented many times:

- LSI (Latent Semantic Indexing) [Foltz+92]
- PCA (Principal Component Analysis) [Jolliffe86]
- KL (Karhunen-Loeve Transform)
- Mahalanobis distance

SVD - conclusions - cont’d

End-2-end

Table Overview

<table>
<thead>
<tr>
<th></th>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Powerlaws, jellyfish</td>
<td>Growth pattern, Compare graphs</td>
<td>SVD</td>
</tr>
<tr>
<td>Link</td>
<td>LRD, ON/OFF sources</td>
<td>Effect of topology and protocols</td>
<td>SVD</td>
</tr>
<tr>
<td>End-2-end</td>
<td>LRD loss and RTT</td>
<td>Troubleshoot, cluster and predict</td>
<td>SVD</td>
</tr>
<tr>
<td>Traffic Matrix</td>
<td>Skewness of location</td>
<td>Comprehensive model, troubleshoot</td>
<td>SVD</td>
</tr>
</tbody>
</table>

Resources: Software and urls

- SVD packages: in many systems (matlab, mathematica, LINPACK, LAPACK)
- stand-alone, free code: SVDPACK from Michael Berry
  http://www.cs.utk.edu/~berry/projects.html
**Books**


**Additional Reading**

- Berry, Michael: http://www.cs.utk.edu/~lsi/

**Books**

Part B - IV
fractals

High-level Outline
- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws

Problem #0: GIS - points
Road end-points of Montgomery county:
- Q1: # neighbors(?)
- Q2: distribution?
  - not uniform
  - not Gaussian
  - no rules??

Problem #0: GIS - points
(could be: geo-locations of IP addresses launching DDoS attack)

Problem #1: traffic
- disk trace (from HP - J. Wilkes); Web traffic - fit a model
  - #bytes
  - how many explosions to expect?
  - queue length distr.?
Problem #1: traffic
- Kb per unit time (requests on a web server)
  http://repository.cs.vt.edu/ lbl-conn-7.tar.Z

Problem #2 - topology
How does the Internet look like?

Problem #3 - spatial d.m.
Galaxies (Sloan Digital Sky Survey w/ B. Nichol)
- ‘spiral’ and ‘elliptical’
galaxies
- patterns?
- attraction/repulsion?
- separable?

Problem #3 - spatial d.m.
Avg packet rate
- ‘good’ and ‘bad’ IP addresses
- can we separate them?

Problem #3 - spatial d.m.
Avg packet size

Common answer:
Fractals / self-similarities / power laws
B.IV - Fractals - outline

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems
- More examples and tools
- Conclusions – practitioner’s guide

What is a fractal?

= self-similar point set, e.g., Sierpinski triangle:

Paradox: Infinite perimeter ; Zero area!

Definitions (cont’d)

- Paradox: Infinite perimeter ; Zero area!
- ‘Dimensionality’: between 1 and 2
- actually: Log(3)/Log(2) = 1.58...

Dfn of fd:

ONLY for a perfectly self-similar point set:

= log(n)/log(f) = log(3)/log(2) = 1.58

Intrinsic (‘fractal’) dimension

- Q: fractal dimension of a line?
- A: 1 (= log(2)/log(2))
Intrinsic (‘fractal’) dimension

- Q: defn for a given set of points?

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Intrinsic (‘fractal’) dimension

- Q: fractal dimension of a line?
- A: nn (<= r) \( \sim r^d \)
  
  \( \text{‘power law’: } y=rx^a \)

Intrinsic (‘fractal’) dimension

- A: nn (<= r) \( \sim r^d \)
  
  \( \text{fd= slope of (log(nn)) vs log(r) } \)

Intrinsic (‘fractal’) dimension

- Algorithm, to estimate it?

  Notice

  \( \text{avg nn}(<=r) \) is exactly

  \( \text{tot#pairs}(<=r) / (N) \)

Observations:

- Euclidean objects have integer fractal dimensions
  - point: 0
  - lines and smooth curves: 1
  - smooth surfaces: 2
- fractal dimension -> roughness of the periphery

B.IV - Fractals - outline

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems
- More examples and tools
- Conclusions – practitioner’s guide
**Fast estimation**

- Bad news: There are more than one fractal dimensions
  - Minkowski fd; Hausdorff fd; Correlation fd; Information fd
- Great news:
  - they can all be computed fast! (O(N); O(N log N))
  - Code is on the web (www.cs.cmu.edu/~christos)
  - they usually have nearby values

**B.IV - Fractals - outline**

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems: P#0 - points
- More examples and tools
- Conclusions – practitioner’s guide

**Problem #0: GIS points**

- Montgomery county:
  - F=C*r^(-2)
- How, for the (correlation) fractal dimension?
- A: Box-counting plot:

**Solution #0**

\[ \log(\text{#pairs(within } \leq r)) \]

\[ \log(r) \]

A: self-similarity ->
- \( \Leftrightarrow \) fractals
- \( \Leftrightarrow \) scale-free
- \( \Leftrightarrow \) power-laws
  \( y=x^a, F=C r^{a-2} \)

**Examples:LB county**

- Long Beach county of CA (road end-points)
Example: traffic

• Kb per unit time (requests on a web server)

Solution #1: traffic

• disk traces: self-similar; (also: [Leland+94])
• How to generate such traffic?

Solution #1: traffic

• disk traces (80-20 ‘law’ = ‘multifractal’)
  [Riedl+99], [Wang+02]

B.IV - Fractals - outline

• Motivation – 3 problems / case studies
• Definition of fractals and power laws
• Fast Estimation of fractal dimension

  Solutions to posed problems: P#1 - traffic
  • More examples and tools
  • Conclusions – practitioner’s guide

B.IV - Fractals - outline

• Motivation – 3 problems / case studies
• Definition of fractals and power laws
• Fast Estimation of fractal dimension

  Solutions to posed problems: P#2 - topology
  • More examples and tools
  • Conclusions – practitioner’s guide

Problem#2: Internet topology

• How does the internet look like?
Problem#2: Internet topology

- How does the internet look like?
- Internet routers: how many neighbors within h hops?

Reachability function: number of neighbors within r hops, vs r (log-log).
Mbone routers, 1995

Problem#2: Internet topology

- Internet routers: how many neighbors within h hops? (= correlation integral)

Solution#3: spatial d.m.

Galaxies (‘BOPS’ plot - [sigmod2000])

B.IV - Fractals - outline

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems: P#3: spatial d.m.
- More examples and tools
- Conclusions – practitioner’s guide

Solution#3: spatial d.m.
A famous power law: Zipf’s law

Recall that they are related concepts:
- fractals <=>
- self-similarity <=>
- scale-free <=>
- power laws ( y = x^a )

B.IV - Fractals - outline

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems
  - More examples and tools
- Conclusions – practitioner’s guide
Power laws, cont’ed

- In- and out-degree distribution of web sites [Barabasi, IBM-CLEVER]
- length of file transfers [Bestavros+]
- Click-stream data [Montgomery+01]
- web hit counts [Huberman]

More power laws

- duration of UNIX jobs; of UNIX file sizes
- Energy of earthquakes (Gutenberg-Richter law) [simscience.org]

Even more power laws:

- Income distribution (Pareto’s law)
- publication counts (Lotka’s law)

Olympic medals (Sidney):

<table>
<thead>
<tr>
<th>log(rank)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Fractals

Let’s see some fractals, in real settings:

Fractals: Brain scans

- Oct-trees; brain-scans

<table>
<thead>
<tr>
<th>Log(#octants)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.63 = fd</td>
</tr>
</tbody>
</table>
Fractals: Medical images

[Burgett et al, SPIE ‘93]:
- benign tumors: \( fd \approx 2.37 \)
- malignant: \( fd \approx 2.56 \)

More fractals:

- cardiovascular system: 3 (!)
- stock prices (LYCOS) - random walks: 1.5
  - 1 year
  - 2 years
- Coastlines: 1.2-1.58 (Norway!)

B.IV - Fractals - outline

- Motivation – 3 problems / case studies
- Definition of fractals and power laws
- Fast Estimation of fractal dimension
- Solutions to posed problems
- More examples and tools
- Conclusions – practitioner’s guide

Conclusions

- Real data often disobey textbook assumptions
  (Gaussian, Poisson, uniformity, independence)
  - avoid ‘mean’ - use median, or even better, use:
  - fractals, self-similarity, and power laws, to find patterns

Practitioner’s guide:

- Fractals: help characterize a (non-uniform) set of points
- Detect non-homogeneous regions (e.g., legal login time-stamps may have different \( fd \) than intruders’)

More examples and tools
- (Gaussian, Poisson, uniformity, independence)
Practitioner’s guide

• **tool#1**: for points ‘correlation integral’: (#pairs within \( \leq r \)) vs (distance \( r \))
• **tool#2**: for categorical values rank-frequency plot (a’la Zipf)

---

Practitioner’s guide:

• **tool#1**: correlation integral, for a set of objects, with a distance function (slope = intrinsic dimensionality)

- \( \log(\text{#pairs(within} \leq r)) \)

---

Practitioner’s guide:

- **tool#2**: rank-frequency plot (for categorical attributes)

- \( \log(\text{degree}) \)
- \( \log(\text{rank}) \)
- \( \log(\text{freq}) \)

---

High-level Outline

- Part A - what we know about the Internet
- Part B - how to find more
  - B.I - Traditional Data Mining tools
  - B.II - Time series: analysis and forecasting
  - B.III - New Tools: SVD
  - B.IV - New Tools: Fractals & power laws

- ‘Take-home’ messages:

---

Table Overview

<table>
<thead>
<tr>
<th>Know</th>
<th>Don’t Know</th>
<th>How to learn more</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Powerlaws, jellyfish, Growth pattern, Compare graphs</td>
<td>SVD, fractals</td>
</tr>
<tr>
<td>Link</td>
<td>LRD, ON/OFF sources</td>
<td>ARIMA, wavelets, 80-20</td>
</tr>
<tr>
<td>End-2-end</td>
<td>LRD loss and RTT</td>
<td>ARIMA, wavelets, 80-20</td>
</tr>
<tr>
<td>Traffic Matrix</td>
<td>Skewness of location, Comprehensive model, troubleshooting</td>
<td>Power-laws, multifractals, clustering</td>
</tr>
</tbody>
</table>

---

Table Overview

<table>
<thead>
<tr>
<th>Problems</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topology</td>
<td>Classif, clustering</td>
</tr>
<tr>
<td>Link</td>
<td>ARIMA, wavelets</td>
</tr>
<tr>
<td>End-2-end</td>
<td>SVD</td>
</tr>
<tr>
<td>Traffic Matrix</td>
<td>Fractals, 80-20</td>
</tr>
<tr>
<td>Problems</td>
<td>Power-laws</td>
</tr>
</tbody>
</table>
OVERALL CONCLUSIONS

• WEALTH of powerful, scalable tools in data mining (classification, clustering, SVD, fractals)
• traditional assumptions (uniformity, iid, Gaussian, Poisson) are often violated, when fractals/self-similarity/power-laws deliver.

Resources: Software & urls

• Fractal dimensions: Software
  - www.cs.cmu.edu/~christos

Books


Further reading:


Further reading:


Further reading:

Further reading


michalis@cs.ucr.edu
www.cs.ucr.edu/~michalis

christos@cs.cmu.edu
www.cs.cmu.edu/~christos