Database Techniques with Motion Capture

Abstract

Motion-capture databases are now large, varied, and widely used. This course covers techniques that are useful for organizing, processing, and navigating such databases. Topics include choice of distance function, indexing for fast retrieval, and time-series prediction for stitching, segmentation, and outlier detection. Current and potential applications are discussed.

Syllabus

- Introduction / Overview (5min)
- Database techniques: Examples in Computer Animation (25min)
- Database techniques: Methods (85min)
  - Similarity Search and Database indexing
    - Why we need similarity search
    - Distance functions (Euclidean, LP norms, time-warping)
    - Fast searching (R-trees, M-trees)
  - Feature extraction and dimensionality reduction
    - DFT
    - Wavelets
    - SVD/PCA
    - FastMap
    - ICA
  - Linear Forecasting
    - Main idea behind linear forecasting
    - AR methodology
    - Multivariate regression
    - Recursive Least Squares
    - De-trending; periodicities
- Wrapup (20min)
  - The CMU motion capture database
  - Discussion of possible future applications

Speaker Short Bios

Christos Faloutsos is a Professor at Carnegie Mellon University. He has received the Presidential Young Investigator Award by the National Science Foundation (1989), 7 “best paper” awards, and four teaching awards. He is a member of the executive committee of SIGKDD; he has published over 150 refereed articles, one monograph, and holds four patents. His research interests include data mining, fractals, indexing methods for multimedia and text data bases, and data base performance.

Jessica Hodgins joined the Robotics Institute and Computer Science Department at Carnegie Mellon University as a Associate Professor in fall of 2000. Prior to moving to CMU, she was an an Associate Professor and Assistant Dean in the College of Computing at Georgia Institute of Technology. She received her Ph.D. in Computer Science from Carnegie Mellon University in 1989. She has received a NSF Young Investigator Award, a Packard Fellowship, and a Sloan Fellowship. She was editor-in-chief of ACM Transactions on Graphics from 2000-2002 and Papers Chair for ACM SIGGRAPH 2003.

Nancy Pollard is an Associate Professor in the Robotics Institute and the Computer Science Department at Carnegie Mellon University. She received her PhD from MIT EECS in 1994. She was awarded an NSF CAREER grant in 2001 for research on ’Quantifying Humanlike Enveloping Grasps,’ and the Okawa Research Grant in 2006 for her work on ’Dexterity and Natural Motion for Computer Graphics and Robotics’. Her primary research objective is to understand how to create natural motion for animated human characters and humanoid robots, with a particular focus on grasping, manipulation, and hands.

Introduction

Over the past decade, motion capture data has moved from being an expensive proposition, with motion data a rare and valuable commodity, to a point where many labs are now fortunate to have their own motion capture equipment, and databases of human motion are freely available on the web. Although there are still many open problems related to capturing human motion data reliably, rapidly, and accurately, we have also reached a point where we wish to understand how to make the best use of the data we already have available. Challenges include fast retrieval of desired motions, feature selection for measuring similarity between motions, creation of cleaner datasets, identification of outliers, and identification of fundamental characteristics of human motion that can be used to generate believable new motions or to better understand the nature of human motion itself.

The aim of this course is to present a selection of core database techniques that may be of use in computer graphics (CG). Some of these techniques are well known in the CG community, and others less so. Our goal is to review and document these techniques in such a way as to make them accessible to a broader audience in CG, as well as to present examples of their use to date and speculate on future use of these techniques.

Database Techniques: Examples in Computer Animation

Database techniques have been used in a variety of areas in computer graphics. Fast and flexible database retrieval algorithms, for example, are of interest for databases of images, video, and 3D shapes. We focus here on human motion capture databases and the use of database techniques for animation of human characters. Our aim is to present a variety of examples, not a complete survey of the field. We welcome suggestions for additions to this overview.

Motion Retrieval

We begin with the problem of fast retrieval of a desired pose or motion. Motion retrieval is an area where database techniques are of particular interest, as clever precomputation can dramatically improve retrieval times.

One subclass of retrieval problems is example-based retrieval. Here, a complete example motion is provided as a query, and the goal is to extract from the database all similar motions. This type of query can be used if a user can easily find a single example of a desired motion (e.g. a single example of a jump), and then wishes to search the database more thoroughly for a specific jump (e.g. one with more energy). This type of query has also been used to identify motion families (e.g., all jumps present in the database).

For fast example-based retrieval, Kovar and Gleicher [16] precompute locally optimal time alignments of motions for all motion in the database. However, their technique requires a great deal of preprocessing time for sizable databases. Forbes and Fiume [10] demonstrate how clustering and dimensionality reduction can improve retrieval time for techniques that rely on dynamic time warping. Chiu and his colleagues [7] employ clustering using a self-organizing map and separately index segments of the body (e.g., arms, legs, torso) to improve retrieval times for techniques based on dynamic time warping. Müller and his colleagues [24, 23, 8] demonstrate how the use of binary features to represent pose (e.g., right foot in front of the body) can produce dramatically faster motion retrieval for techniques that use dynamic time warping. Keogh et al. [15] observe that fully general dynamic time warping may not be needed in many cases, and they present a fast retrieval algorithm for motions that can be aligned well using uniform time scaling. For retrieval of motion families, Kovar and Gleicher [16] and Jenkins and Mataric [14] present an iterative approach, where query results are used as new queries to “grow” a motion family outward from a single example.

A second subclass of retrieval problems is retrieval from a small set of controls. Here, the goal is to retrieve detailed motion capture data from a sparse query that could be obtained quickly, cheaply, easily, or interactively. A number of researchers have developed algorithms for database retrieval from a small set of controls such as may be captured from a pair of cameras or inexpensive tracking devices. Hsu et al. [12] present a dynamic programming technique for assembling best motion sequences based on a reduced marker set (or other controls). Liu et al. [21, 22] preprocess the motion database into a hierarchy of local linear models for fast retrieval of a character pose from a sparse marker set. They also attempt to identify the particular subset of markers that provides the most information for accurate pose retrieval. Chai and Hodgins [6] preprocess the motion database by constructing a motion graph. They then track estimated actor state within that motion graph, and use the graph to narrow the search for poses that match the query at each point in time.

In a third subclass of problems, the user must browse through the database for the desired motion. Sakamoto et al. [29] present a visual interface for retrieving a desired motion, where the user identifies key postures within a map of poses obtained using a self-organizing map algorithm. Assa et al. [4] present a visual synopsis of motions that could aid database browsing. Ren [26] also explores representations that may help to make motions and motion classes easy to visualize in a browsing-style interface.

Dimensionality Reduction and Database Compression

Many of the motion retrieval algorithms mentioned above make use of some form of dimensionality reduction, including Principal Component Analysis (PCA) [10], local linear models [21, 6], and nonlinear dimension reduction [14].

These and related techniques have been used for other research problems in character animation. For example, PCA has been used by Safonova et al. [28] to create a reduced dimensional space within which to optimize human motion. Notably, they use this reduced dimensional space only for redundant degrees of freedom, allowing them to maintain desired contact constraints such as foot plants. Arikan [1] makes use of clustered PCA specifically for human motion database compression. Notably, they compress the feet separately for better reconstruction of foot contact. Pan et al. [25] separate motion into components using Independent Component Analysis (ICA), with the goal of identifying meaningful independent features of the motion. Shapiro et al. [30] also use ICA to separate out components of motion, with the goal of recombining them in various ways to alter motion style. Linear dynamic models have been employed by Li et al. [20] in the construction of motion textons, which allow creation of extended and variable sequences of human motion. Grochow et al. [11] use a Scaled Gaussian Process Latent Variable Model to map poses into a low dimensional space that can be used to do a form of intelligent inverse kinematics, returning poses in response to user input that are highly probable given a training dataset.

Motion Segmentation and Classification

Segmentation and classification are of interest for automatically labelling motion databases. In the area of segmentation, Fod et al. [9] divide motion into very short primitive segments by looking at zero-velocity crossings. Barbič et al. [5] explore three techniques for segmenting motions into higher level behaviors such as running, walking, and jumping. The main idea is to segment motions at points where local models of the motion created for windows before and after the segmentation point do not match each other well. Probabilistic PCA is recommended for construction of these local models.

For motion annotation, Arikan et al. [3] present an approach that makes use of Support Vector Machine classifiers to create annotations such as “carry” and “jump”. Müller and his colleagues [24, 23, 8] use vectors of binary features (e.g., right foot in front of the body) to learn templates for motion classes. These motion classes can then be used to annotate new motions with the appropriate motion class.

Distance Metrics

Distance functions determine how motions are clustered, compressed, indexed for fast search, formed into motion graphs, etc. Computing the distance between motions requires understanding how motion may be timewarped, as considered briefly in the section on Motion Retrieval. However, consider just the simple question of how to measure differences between character state at two points in time.

Weighted Euclidean distance between poses is the most commonly used distance metric. Perhaps the most straightforward choice is weighted Euclidean distance with a state vector that consists of joint angles and angular velocities (e.g., [19]). Wang and Bodenheimer [31] investigated an optimal weighting for this distance metric through human-subjects experiments. A number of researchers have argued for weighted Euclidean distance on 3D Cartesian points instead of using joint angles. These points may be located at the joint positions (e.g.,[2]), more densely sampled on the surface of the character (e.g., [17]) or formed into a 3D coordinate system placed at joint positions (e.g., [1]). Müller and his colleagues [24, 23, 8] compute distance based on vectors of binary features (e.g., right foot in front of the body). Li et al. [20] use a statistical two level Markov approach to learn basic motion textons, and use the transition likelihood between the textons as the distance metric for determining whether those textons may be successfully stitched together.

Arikan [1] suggests that for the application of database compression, weighted Euclidean distance metrics (and other related metrics) may not capture true perceptual distances well. Certainly, unexpected foot sliding may cause large perceptual error that is not captured well by such metrics. (See, for example [13, 1, 18] for further discussion of this topic.) The jury is still out on good distance functions for animation, and we believe that a better understanding of how we perceive natural human motion (e.g., following on the studies of Ren et al. [27], Ikemoto et al. [13] and many others) will be necessary to answer this question.

Database Techniques: Methods

The slides that follow give an overview of database techniques that may be useful in computer graphics, with a specific focus on potential applications in computer animation.

Wrapup

In the course itself, we will follow presentation of these methods with a wrapup, where we speculate on how these techniques may be of use in solving some of the outstanding issues in our field today, especially issues related to memory, speed, and smooth generation of motion.

References

[1] Okan Arikan. Compression of motion capture databases. *ACM Transactions on Graphics*, 25(3):890–897, July 2006.

[2] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Transactions on Graphics*, 21(3):483–490, July 2002.

[3] Okan Arikan, David A. Forsyth, and James F. O'Brien. Motion synthesis from annotations. *ACM Transactions on Graphics*, 22(3):402–408, July 2003.

[4] Jackie Assa, Yaron Caspi, and Daniel Cohen-Or. Action synopsis: pose selection and illustration. *ACM Transactions on Graphics*, 24(3):667–676, August 2005.

[5] Jernej Barbic, Alla Safonova, Jia-Yu Pan, Christos Faloutsos, Jessica K. Hodgins, and Nancy S. Pollard. Segmenting motion capture data into distinct behaviors. In *Graphics Interface 2004*, pages 185–194, May 2004.

[6] Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. *ACM Transactions on Graphics*, 24(3):686–696, August 2005.

[7] C. Y. Chiu, S. P. Chao, M. Y. Wu, and S. N. Yang. Efficient content-based retrieval for motion capture data. *Journal of Visual Communication and Image Representation*, 15:446–466, 2004.

[8] B. Demuth, M. Müller, and B. Eberhardt. An information retrieval system for motion capture data. In *Proceedings of the 28th European Conference on Information Retrieval (ECIR)*, pages 373–384, 2006.

[9] A. Fod, M. Matarić, and O. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12(1):39–54, Jan 2002.

[10] Kevin Forbes and Eugene Fiume. An efficient search algorithm for motion data using weighted pca. In *2005 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 67–76, July 2005.

[11] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popović. Style-based inverse kinematics. *ACM Transactions on Graphics*, 23(3):522–531, August 2004.

[12] Eugene Hsu, Sommer Gentry, and Jovan Popović. Example-based control of human motion. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 69–77, July 2004.

[13] L. Ikemoto, O. Arikan, and D. Forsyth. Quick transitions. In *Symposium on Interactive 3D Graphics and Games (I3D)*, 2007.

[14] Odest Chadwicke Jenkins and Maja J. Matarić. Performance-derived behavior vocabularies: Data-driven acqusition of skills from motion. *International Journal of Humanoid Robotics*, 1(2):237–288, Jun 2004.

[15] E. Keogh, T. Palpanas, V. Zordan, D. Gunopulos, and M. Cardle. Indexing large human-motion databases. In *In Proceedings of the 30th International Conference on Very Large Data Bases*, 2004.

[16] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics*, 23(3):559–568, August 2004.

[17] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, July 2002.

[18] Lucas Kovar, John Schreiner, and Michael Gleicher. Footskate cleanup for motion capture editing. In *ACM SIGGRAPH Symposium on Computer Animation*, pages 97–104, July 2002.

[19] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, July 2002.

[20] Yan Li, Tianshu Wang, and Heung-Yeung Shum. Motion texture: A two-level statistical model for character motion synthesis. *ACM Transactions on Graphics*, 21(3):465–472, July 2002.

[21] Guodong Liu, Jingdan Zhang, Wei Wang, and Leonard McMillan. A system for analyzing and indexing human-motion databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 924–926, 2005.

[22] Guodong Liu, Jingdan Zhang, Wei Wang, and Leonard McMillan. Human motion estimation from a reduced marker set. In *Proceedings of the Symposium on Interactive 3D Graphics and Games (SI3D)*, pages 35–42, 2006.

[23] Meinard Müller and Tido Röder. Motion templates for automatic classification and retrieval of motion capture data. In *2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 137–146, September 2006.

[24] Meinard Müller, Tido Röder, and Michael Clausen. Efficient content-based retrieval of motion capture data. *ACM Transactions on Graphics*, 24(3):677–685, August 2005.

[25] J.-Y. Pan, C. Faloutsos, M. Hamamoto, and H. Kitagawa. AutoSplit: Fast and scalable discover of hidden variables. In *Stream and Multimedia Databases (PAKDD)*, 2004.

[26] Liu Ren. *Statistical Analysis of Natural Human Motion for Animation*. PhD thesis, Carnegie Mellon University, 2007.

[27] Liu Ren, Alton Patrick, Alexei A. Efros, Jessica K. Hodgins, and James M. Rehg. A data-driven approach to quantifying natural human motion. *ACM Transactions on Graphics*, 24(3):1090–1097, August 2005.

[28] Alla Safonova, Jessica K. Hodgins, and Nancy S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics*, 23(3):514–521, August 2004.

[29] Yasuhiko Sakamoto, Shigeru Kuriyama, and Toyohisa Kaneko. Motion map: image-based retrieval and segmentation of motion data. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 259–266, July 2004.

[30] Ari Shapiro, Yong Cao, and Petros Faloutsos. Style components. In *GI '06: Proceedings of the 2006 conference on Graphics interface*, pages 33–39, Toronto, Ont., Canada, Canada, 2006. Canadian Information Processing Society.

[31] Jing Wang and Bobby Bodenheimer. Computing the duration of motion transitions: an empirical approach. In *2004 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 335–344, July 2004.

Database Techniques with Motion Capture

Part 2: Methods

Christos Faloutsos

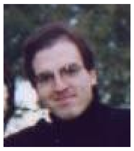
CMU

Thanks

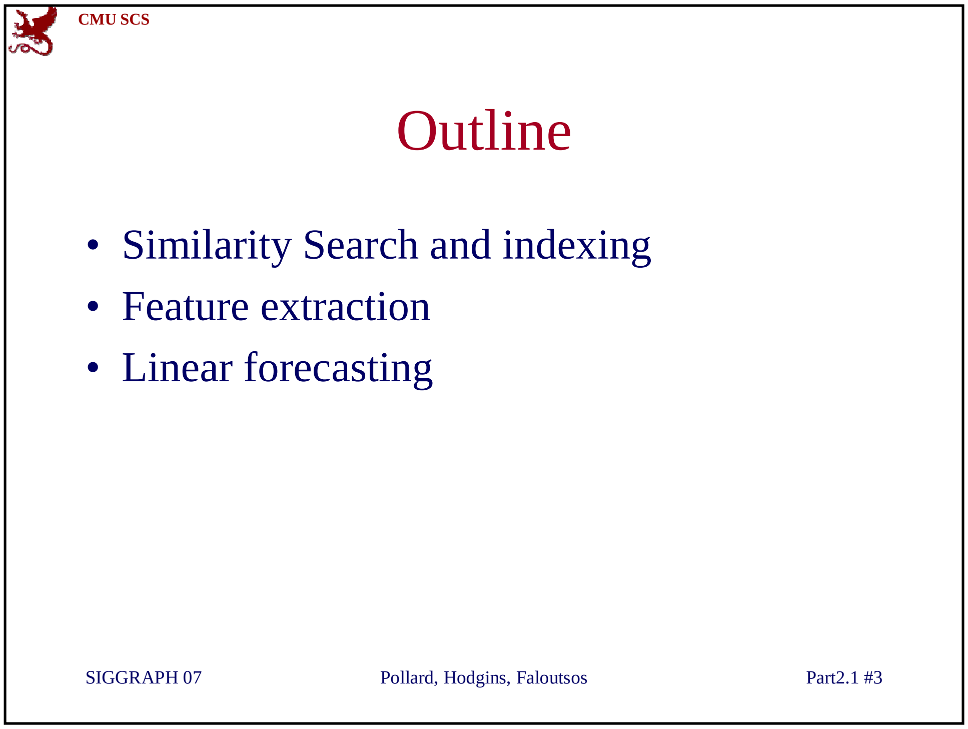
Petros Faloutsos (UCLA) →

← Eamonn Keogh (UCR)

Spiros Papadimitriou (IBM) →

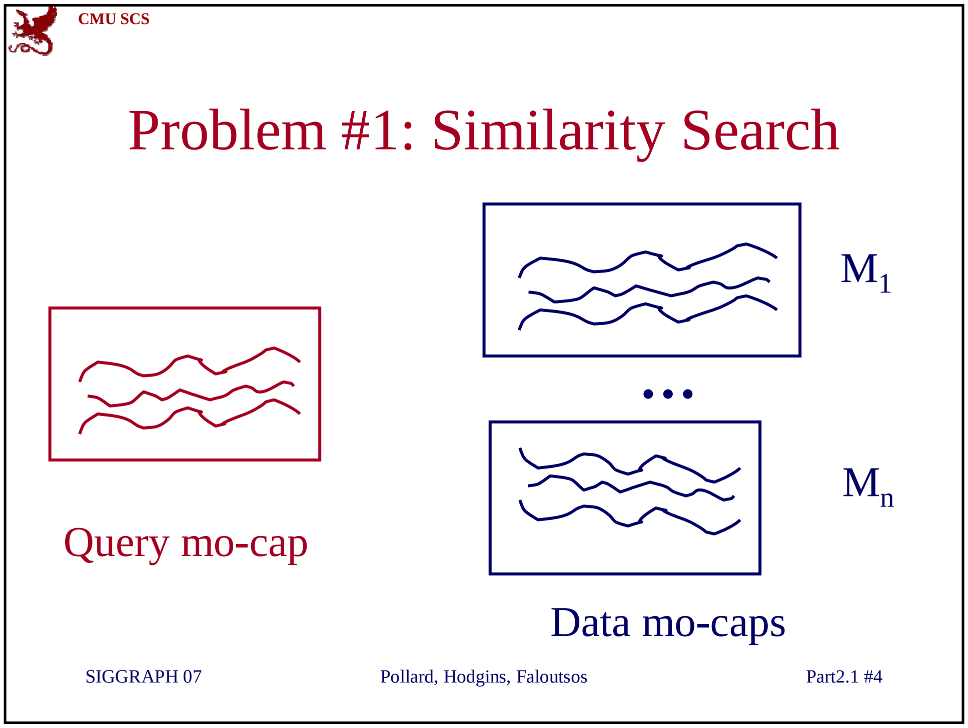


← Byoung-Kee Yi (Pohang U.)



Outline

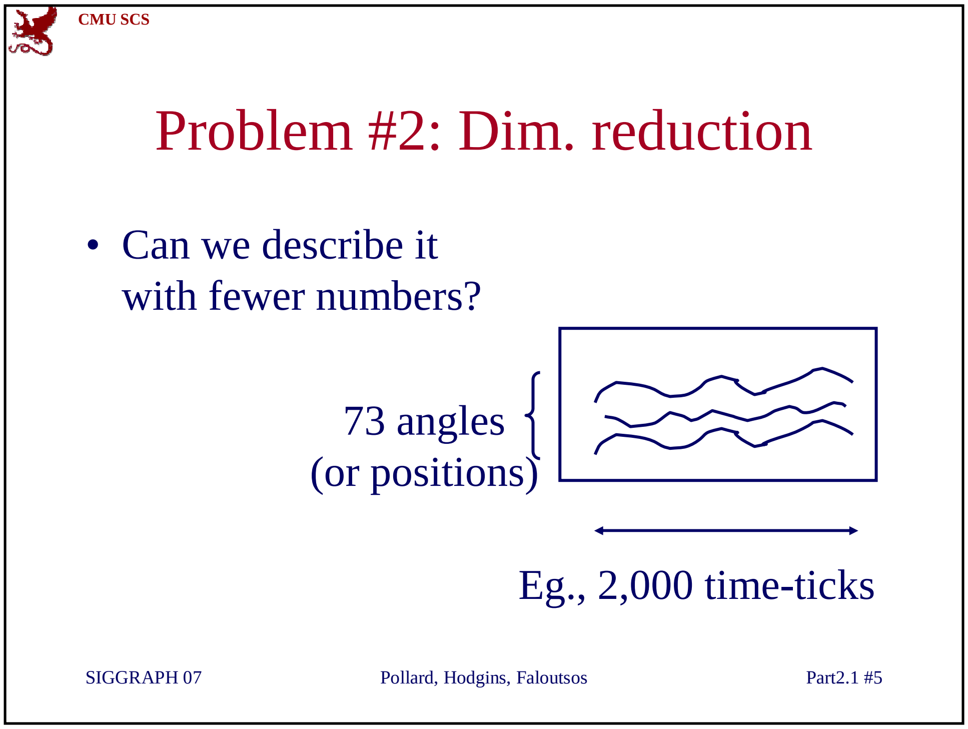
- Similarity Search and indexing
- Feature extraction
- Linear forecasting



Problem #1: Similarity Search

Query mo-cap

Data mo-caps



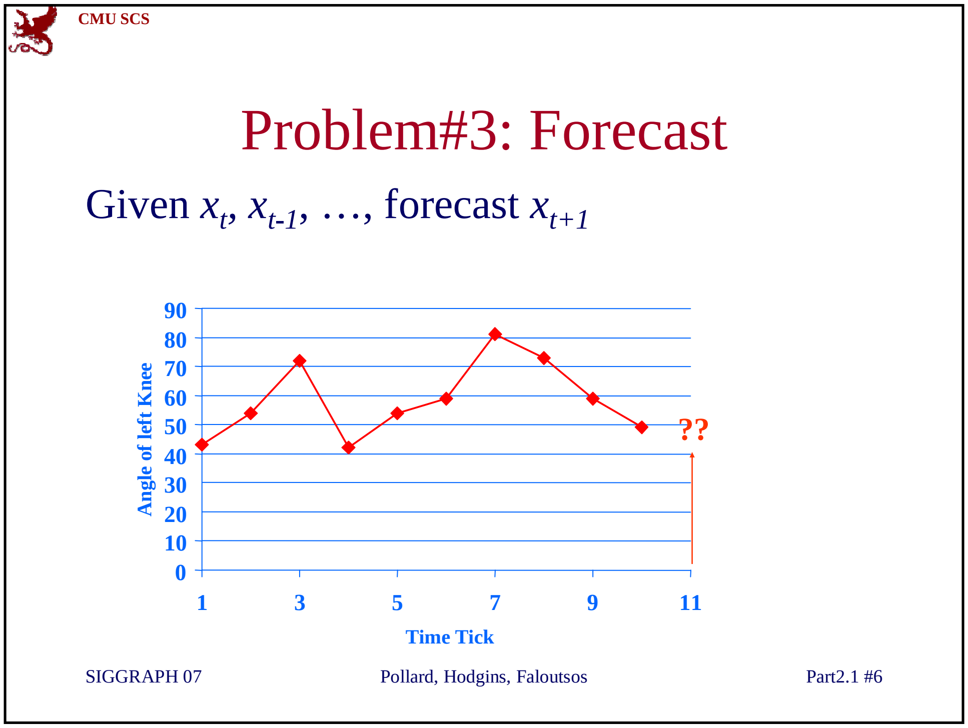
CMU SCS

Problem #2: Dim. reduction

- Can we describe it with fewer numbers?

73 angles
(or positions)

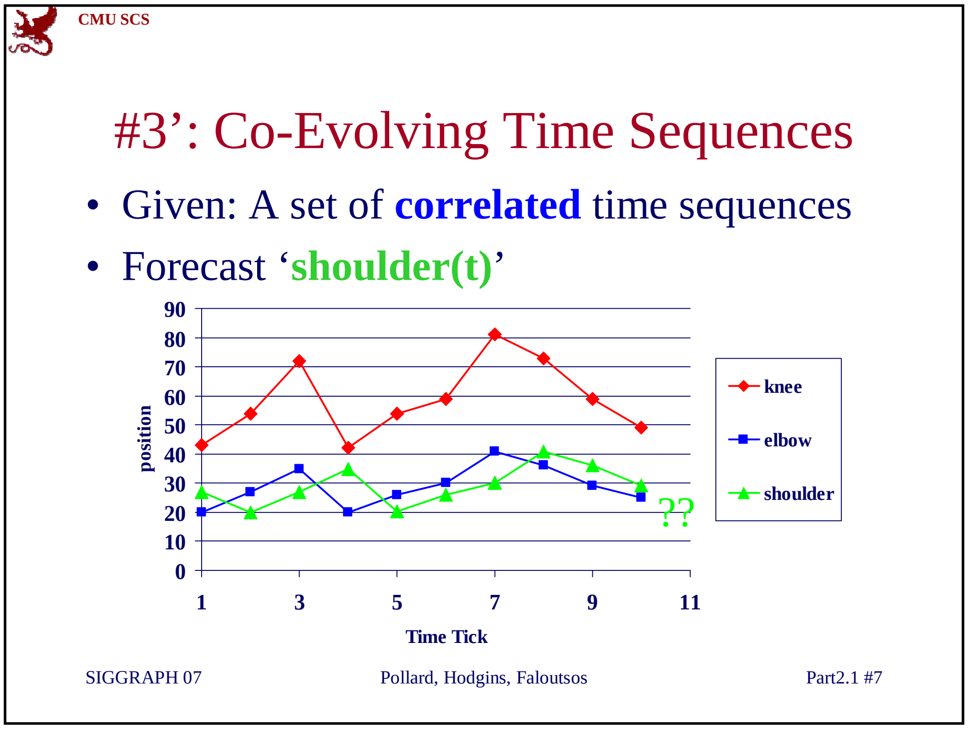
Eg., 2,000 time-ticks



CMU SCS

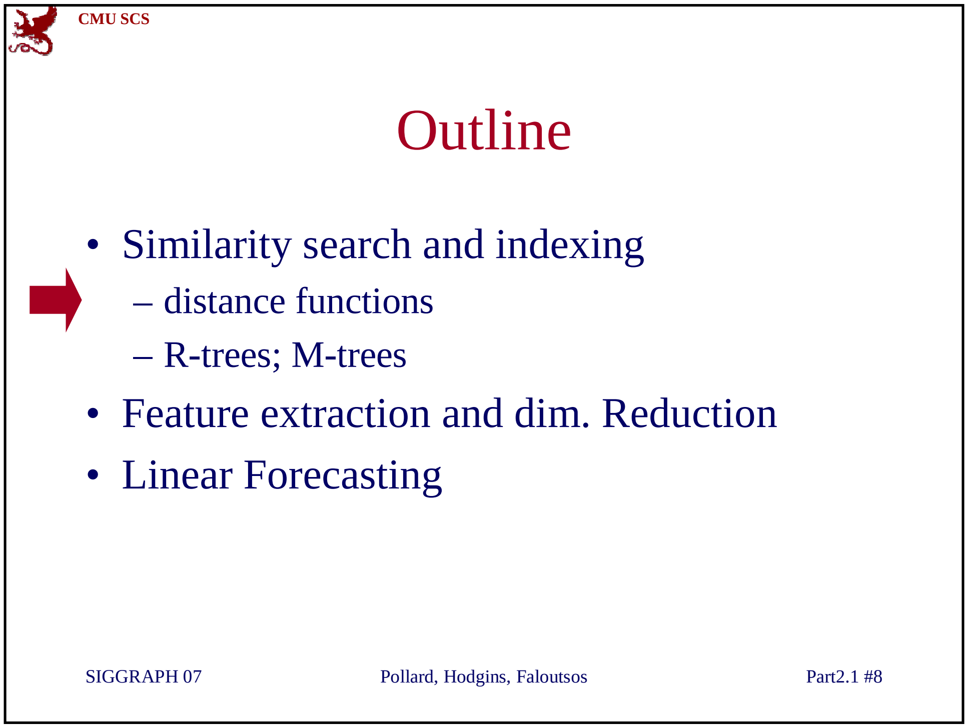
Problem#3: Forecast

Given x_t, x_{t-1}, …, forecast x_{t+1}



#3’: Co-Evolving Time Sequences

- Given: A set of **correlated** time sequences
- Forecast ‘**shoulder(t)**’



Outline

- Similarity search and indexing
  - – distance functions
  - – R-trees; M-trees
- Feature extraction and dim. Reduction
- Linear Forecasting

Importance of distance functions

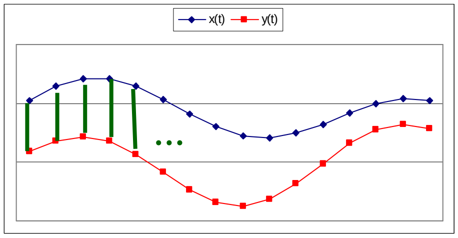
Subtle, but **absolutely necessary**:

- A ‘must’ for similarity indexing (-> forecasting)
- A ‘must’ for clustering

Two major families

- – Euclidean and Lp norms
- – Time warping and variations

Euclidean and Lp

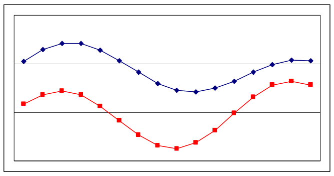


$$D(\vec{x}, \vec{y}) = \sum_{i=1}^{n} (x_i - y_i)^2$$

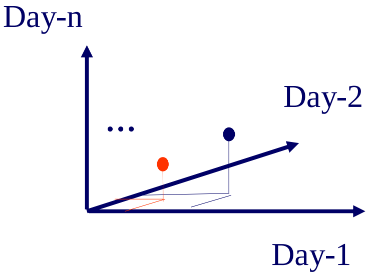
$$L_p(\vec{x}, \vec{y}) = \sum_{i=1}^{n} | x_i - y_i |^p$$

•L_1: city-block = Manhattan
•L_2 = Euclidean
•L_∞

Observation #1



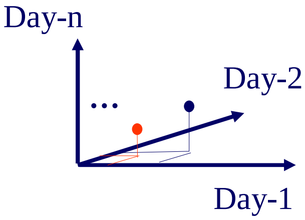
- **Time sequence -> n-d vector**

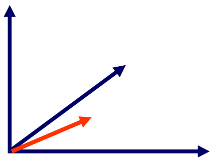


Observation #2

Euclidean distance is closely related to

- cosine similarity
- dot product
- ‘cross-correlation’ function

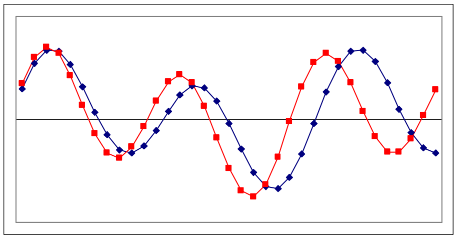




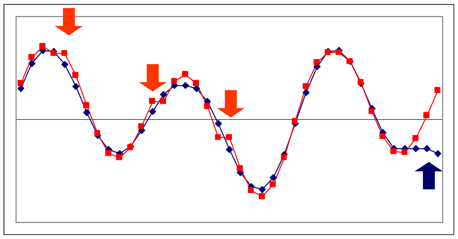
Time Warping

- A.k.a. Dynamic Time Warping (DTW)
- allow accelerations - decelerations
  - (with or w/o penalty)
- THEN compute the (Euclidean) distance (+ penalty)
- related to the string-editing distance

Time Warping



'stutters':



Skip

Time Warping

Q: how to compute it?

A: dynamic programming

D(i, j) = cost to match

prefix of length *i* of first sequence *x* with prefix of length *j* of second sequence *y*

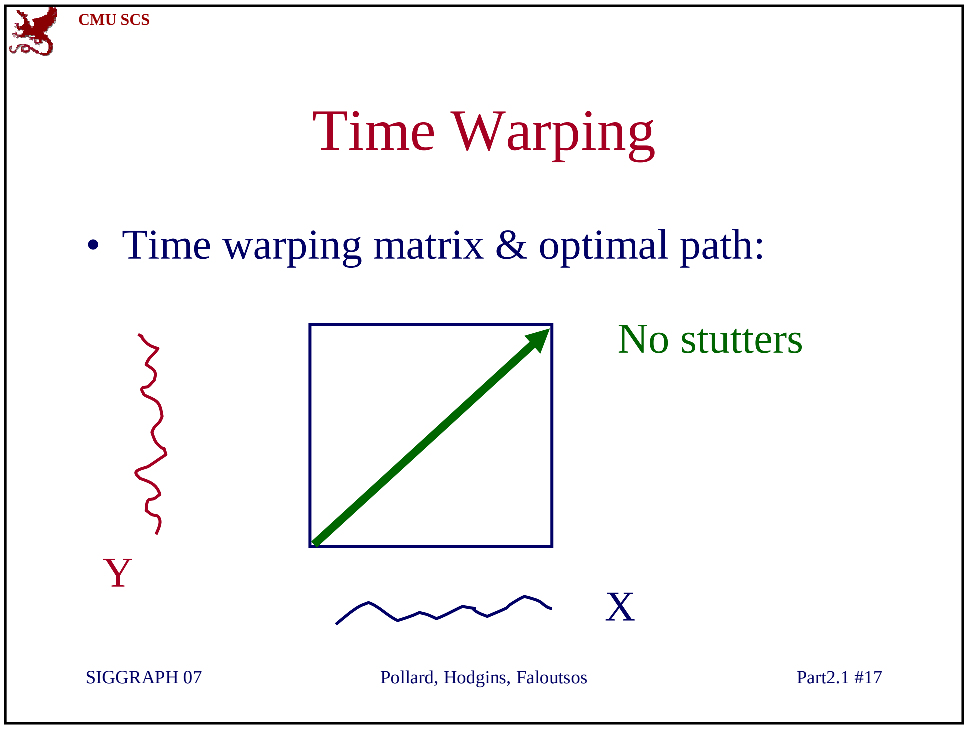
Skip

Time Warping

Thus, with no penalty for stutter, for sequences

$x_1, x_2, \ldots, x_{i,;} \qquad y_1, y_2, \ldots, y_j$

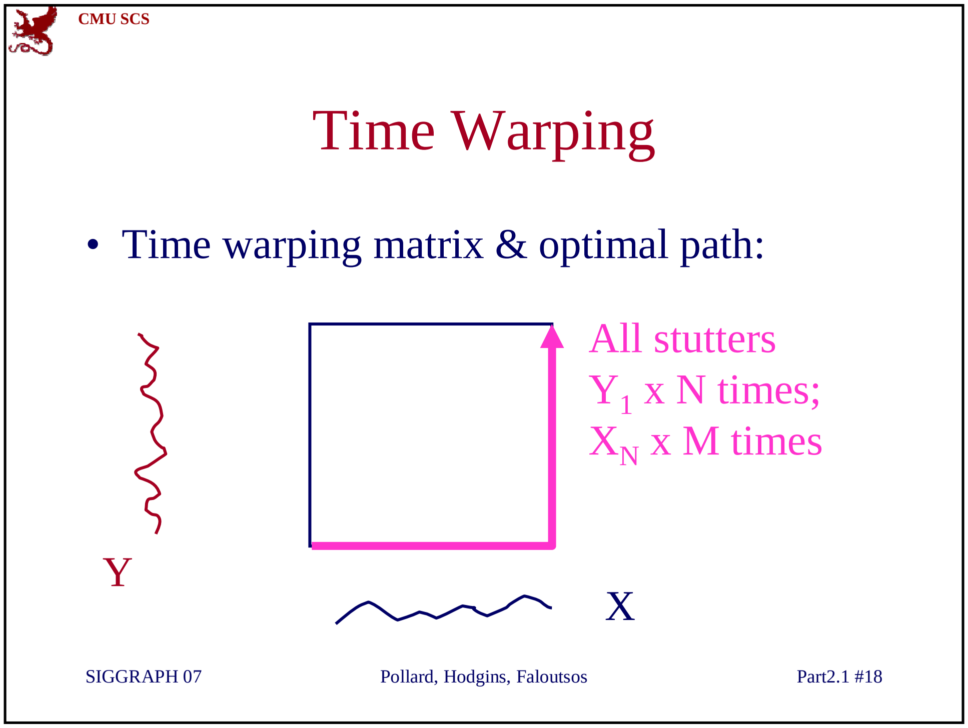
$$D(i,j) = \left\| x[i] - y[j] \right\| + \min \begin{cases} D(i-1, j-1) & \text{no stutter} \\ D(i, j-1) & \text{x-stutter} \\ D(i-1, j) & \text{y-stutter} \end{cases}$$



Time Warping

- Time warping matrix & optimal path:

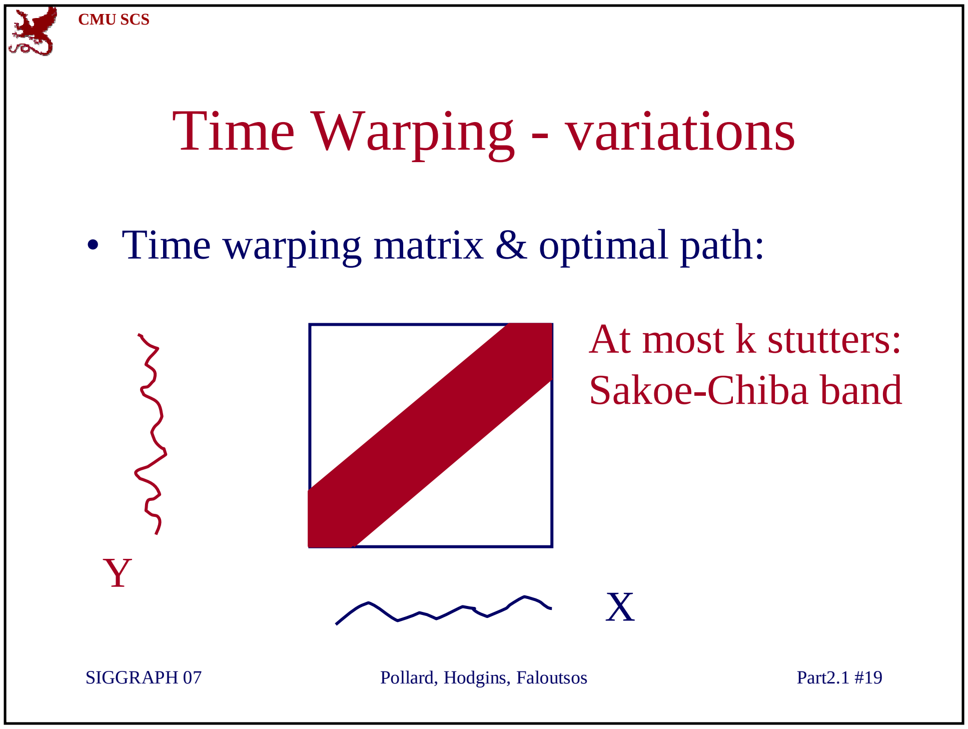
No stutters



Time Warping

- Time warping matrix & optimal path:

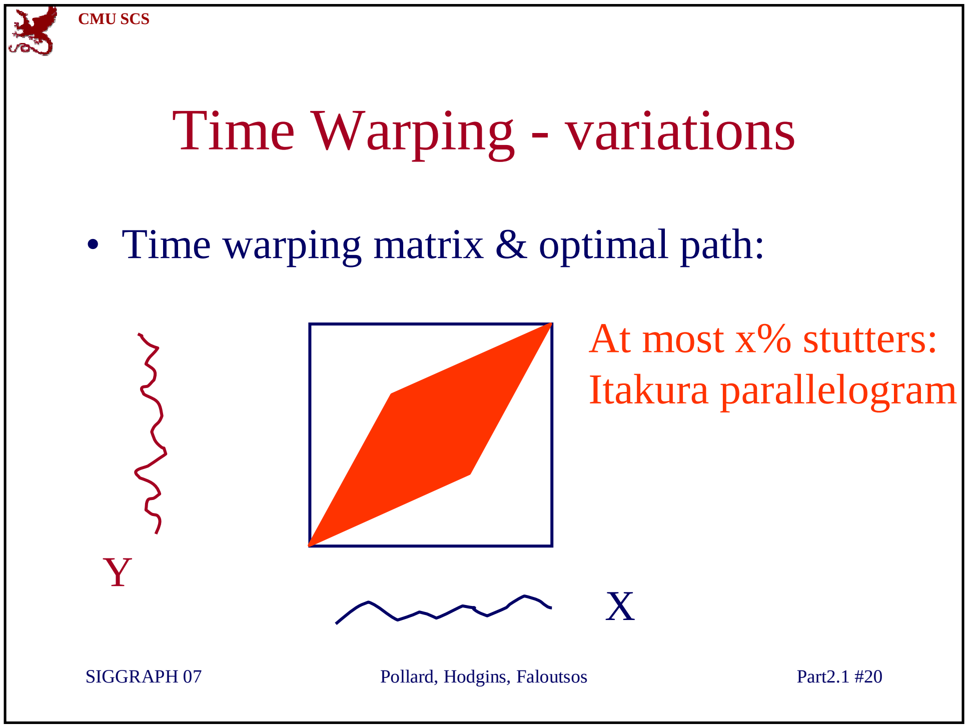
All stutters
Y_1 x N times;
X_N x M times



Time Warping - variations

- Time warping matrix & optimal path:

At most k stutters:
Sakoe-Chiba band



Time Warping - variations

- Time warping matrix & optimal path:

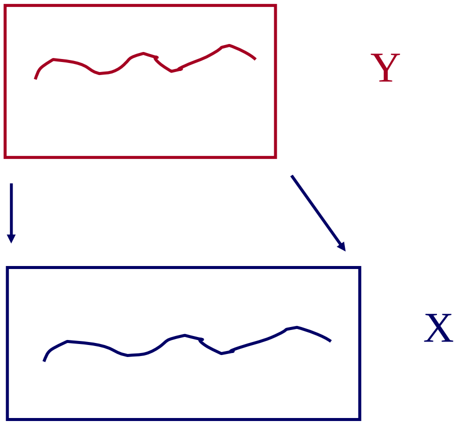
At most x% stutters:
Itakura parallelogram

Skip

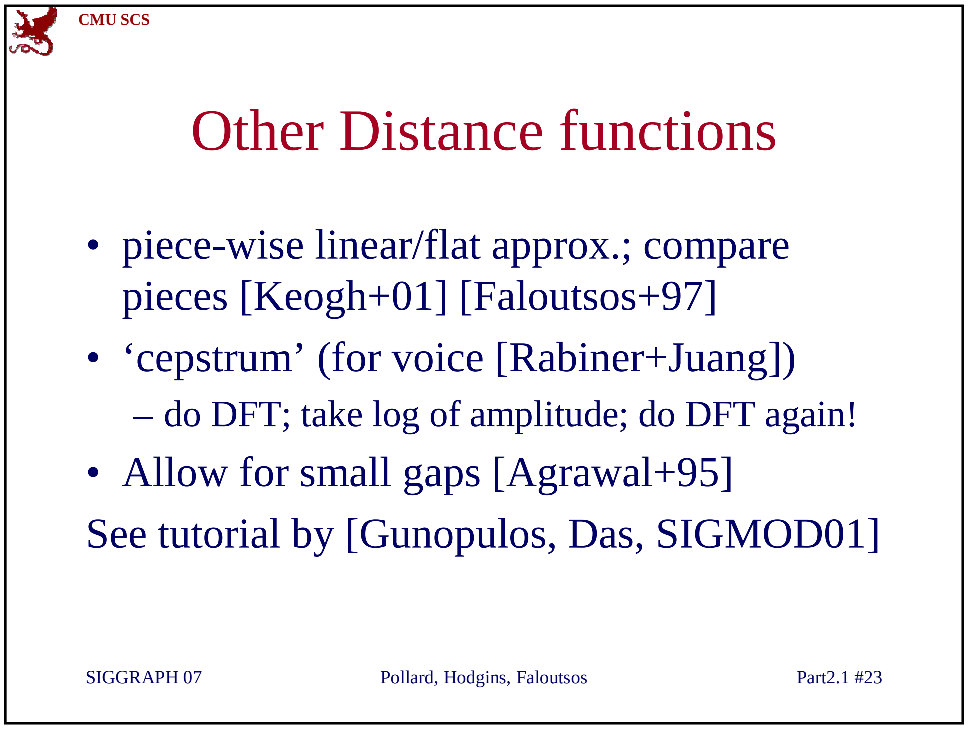
Time warping

- Complexity: O(M*N) - quadratic on the length of the strings
- **Many** variations (penalty for stutters; limit on the number/percentage of stutters; …)
- popular in voice processing [Rabiner+Juang]
- Seems suitable for mo-cap

A variation: Uniform axis scaling



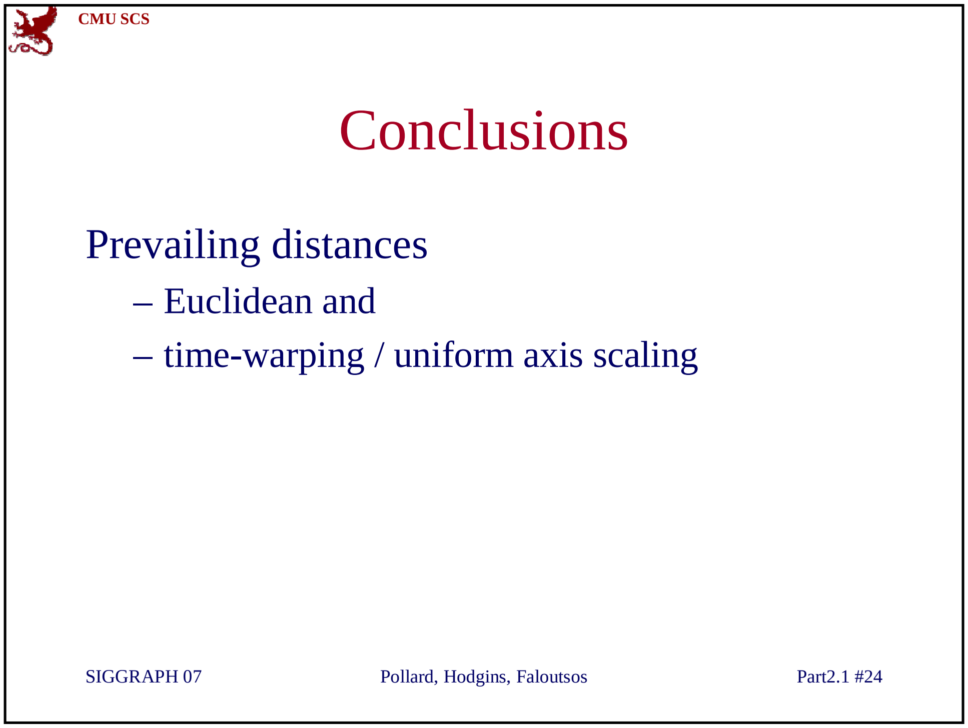
- Stretch / shrink time axis of Y, up to p%, for free
- THEN compute Euclidean distance
- [Keogh+, VLDB04]



Other Distance functions

- piece-wise linear/flat approx.; compare pieces [Keogh+01] [Faloutsos+97]
- ‘cepstrum’ (for voice [Rabiner+Juang])
  - do DFT; take log of amplitude; do DFT again!
- Allow for small gaps [Agrawal+95]

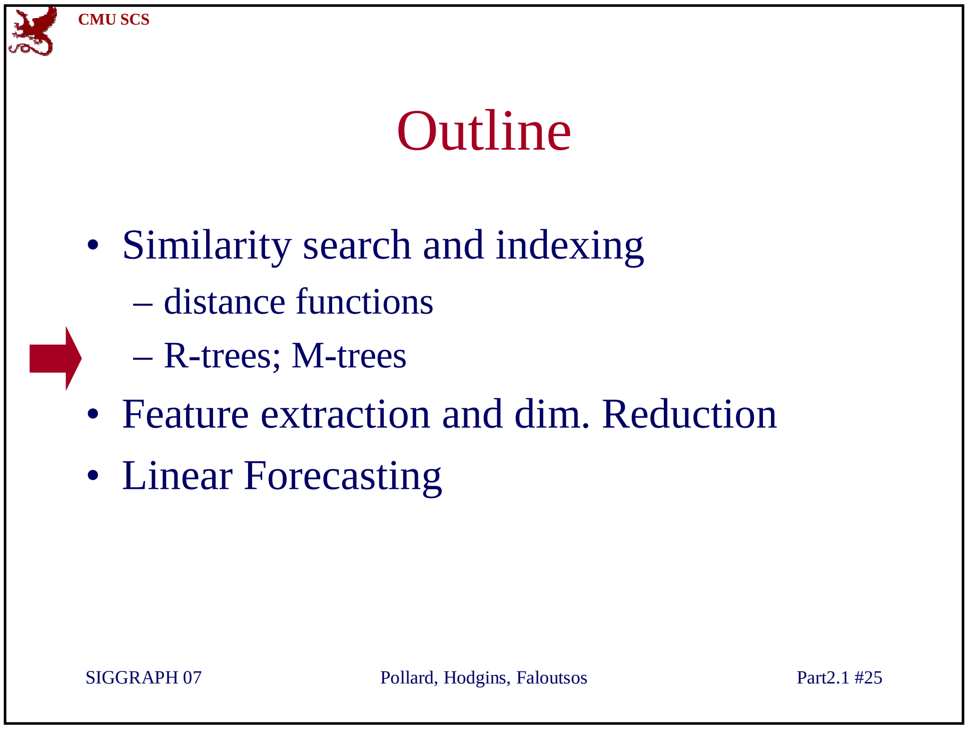
See tutorial by [Gunopulos, Das, SIGMOD01]



Conclusions

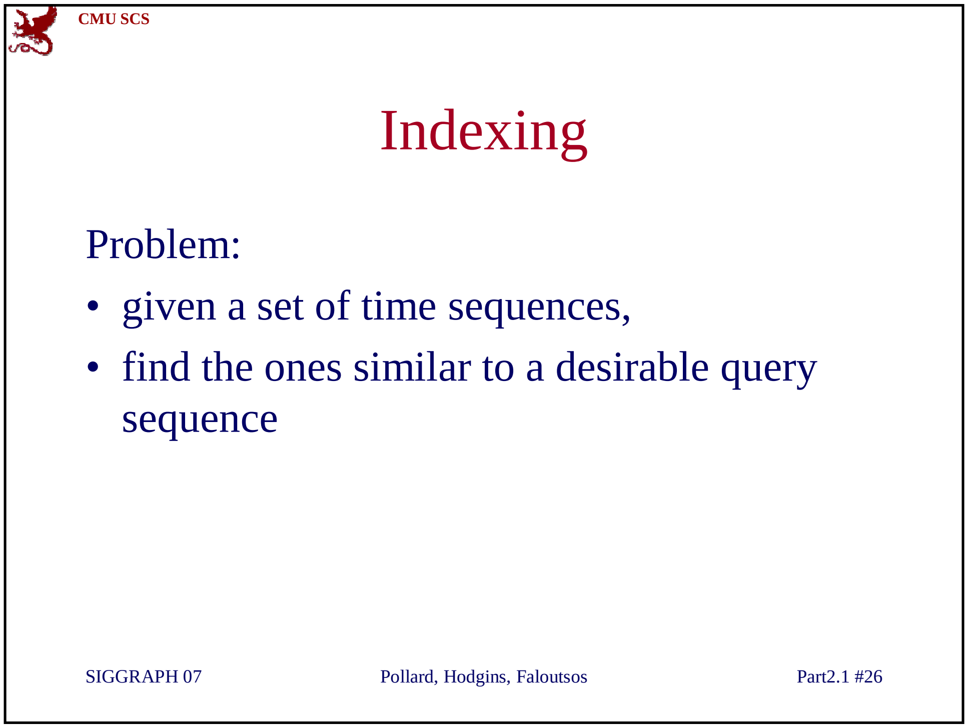
Prevailing distances

- Euclidean and
- time-warping / uniform axis scaling



Outline

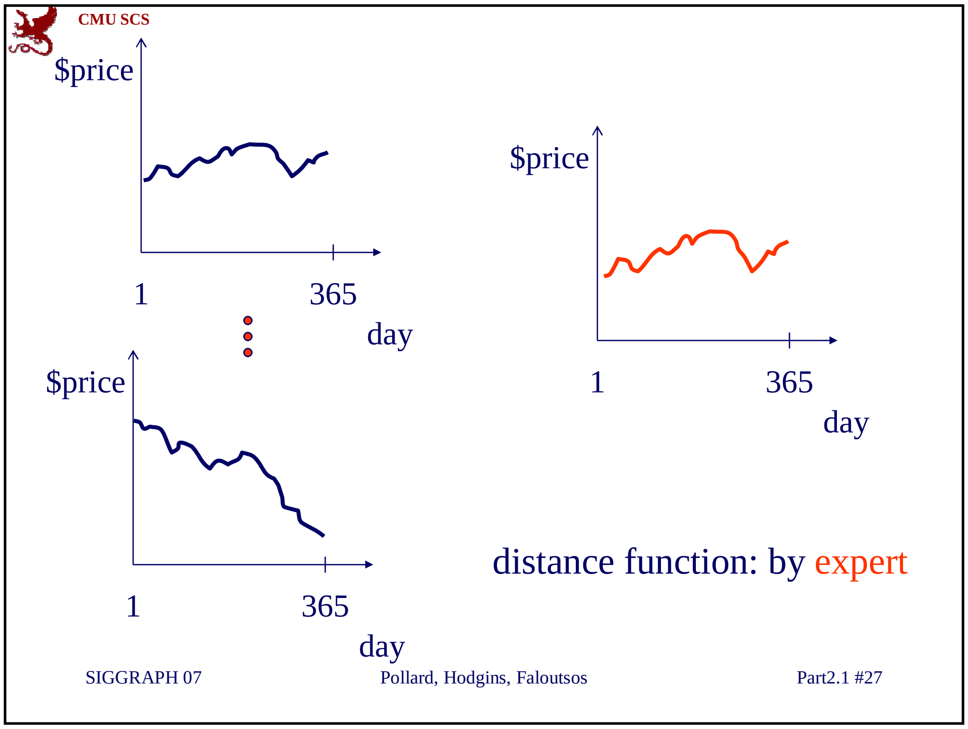
- Similarity search and indexing
  - distance functions
  - R-trees; M-trees
- Feature extraction and dim. Reduction
- Linear Forecasting



Indexing

Problem:

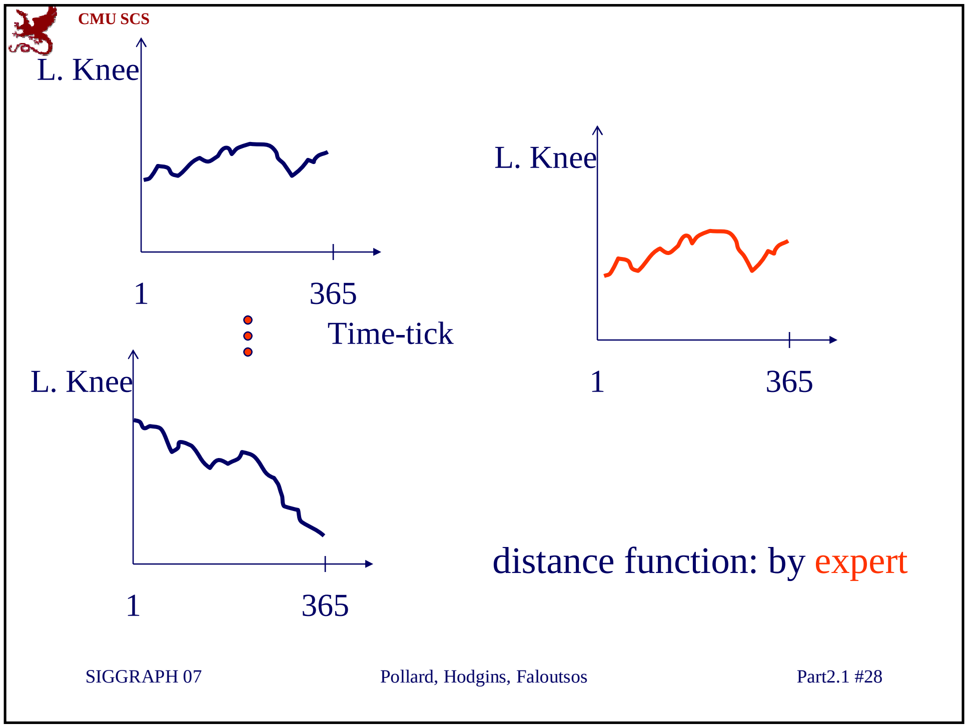
- given a set of time sequences,
- find the ones similar to a desirable query sequence



CMU SCS

distance function: by expert

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.1 #27



CMU SCS

distance function: by expert

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.1 #28

Idea: ‘GEMINI’

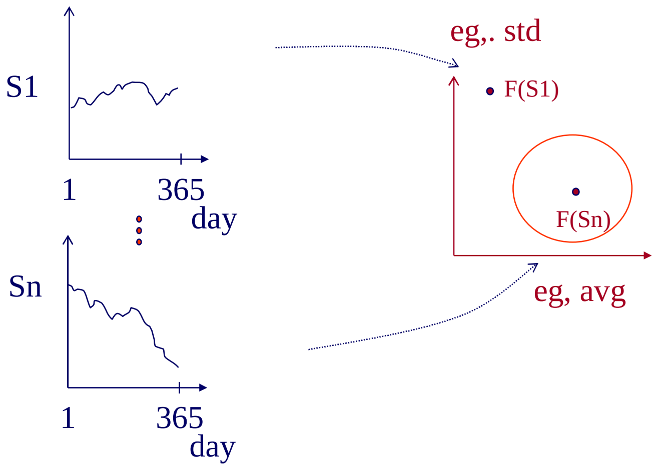
Eg., ‘*find stocks similar to MSFT*’

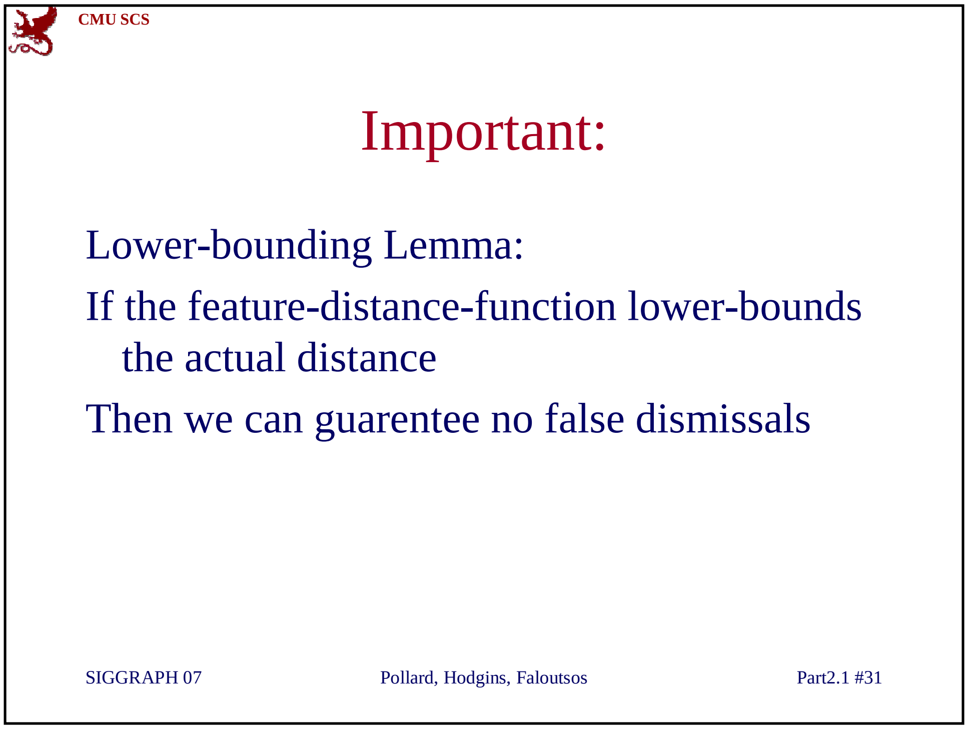
Seq. scanning: too slow

How to accelerate the search?

[Faloutsos96]

‘GEMINI’ - Pictorially



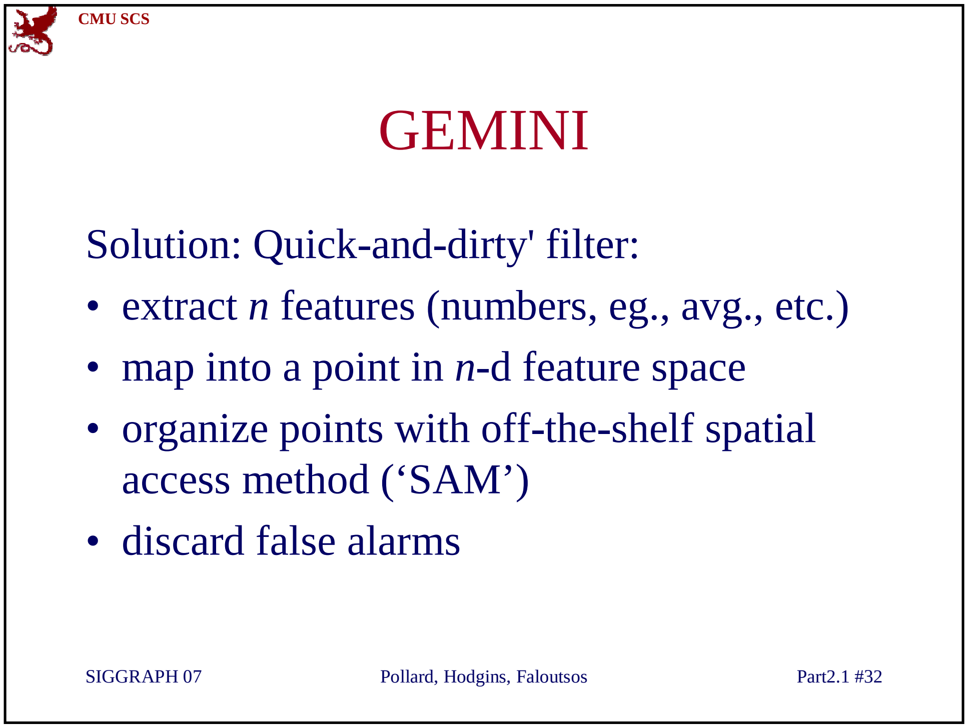


Important:

Lower-bounding Lemma:

If the feature-distance-function lower-bounds the actual distance

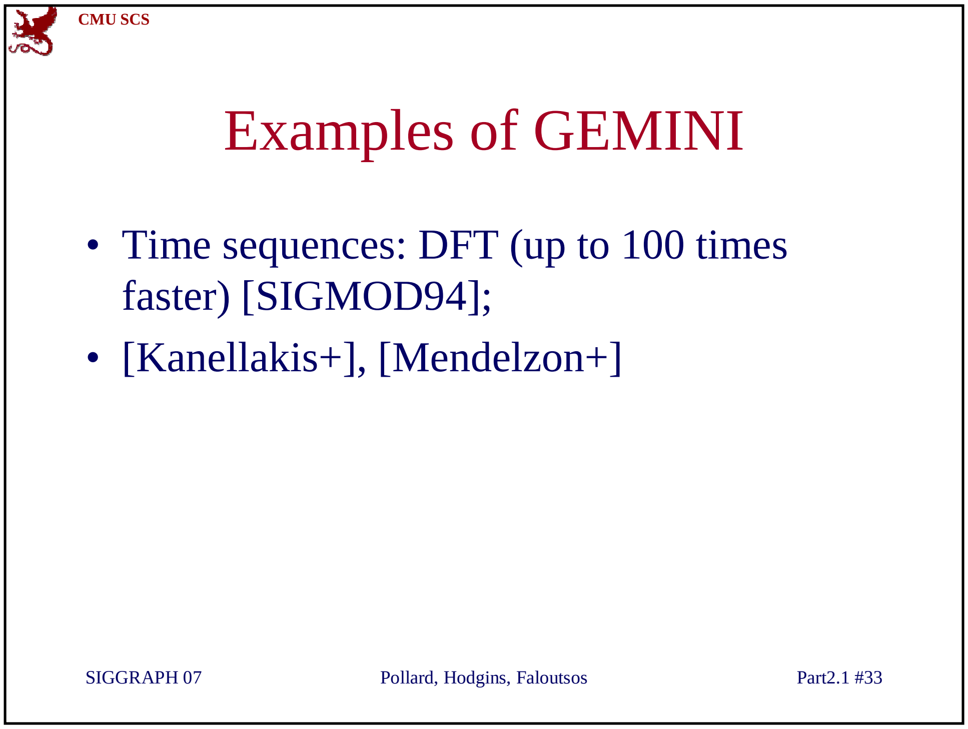
Then we can guarentee no false dismissals



GEMINI

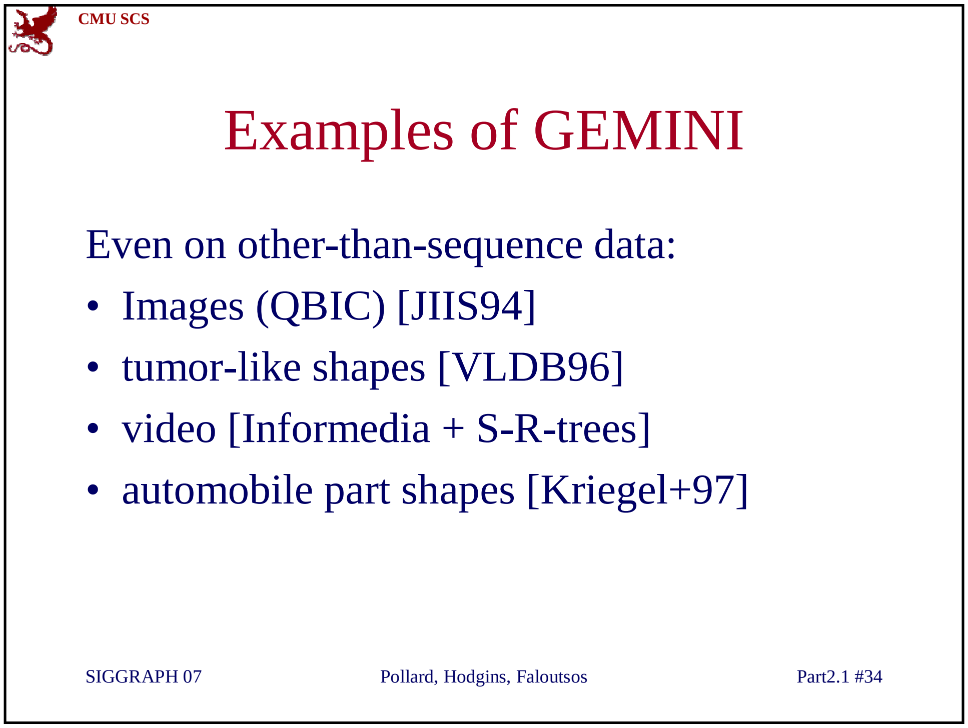
Solution: Quick-and-dirty' filter:

- extract *n* features (numbers, eg., avg., etc.)
- map into a point in *n*-d feature space
- organize points with off-the-shelf spatial access method (‘SAM’)
- discard false alarms



Examples of GEMINI

- Time sequences: DFT (up to 100 times faster) [SIGMOD94];
- [Kanellakis+], [Mendelzon+]



Examples of GEMINI

Even on other-than-sequence data:

- Images (QBIC) [JIIS94]
- tumor-like shapes [VLDB96]
- video [Informedia + S-R-trees]
- automobile part shapes [Kriegel+97]

Indexing - SAMs

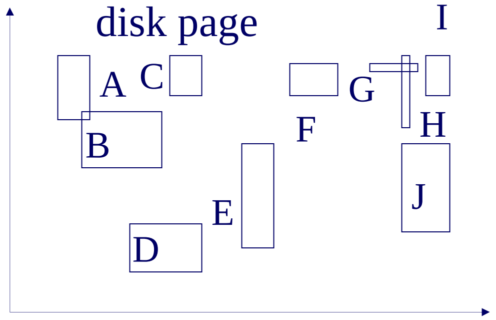
Q: How do Spatial Access Methods (SAMs) work?

A: they group nearby points (or regions) together, on nearby disk pages, and answer spatial queries quickly ('range queries', 'nearest neighbor' queries etc)

For example:

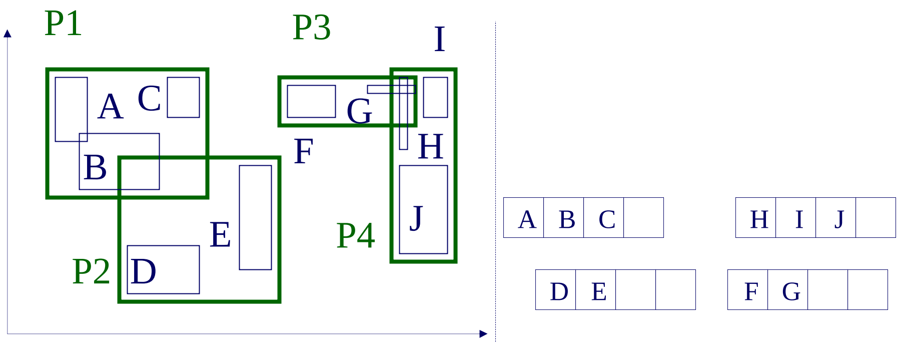
R-trees

- [Guttman84] eg., w/ fanout 4: group nearby rectangles to parent MBRs; each group -> disk page



R-trees

- eg., w/ fanout 4:



A B C

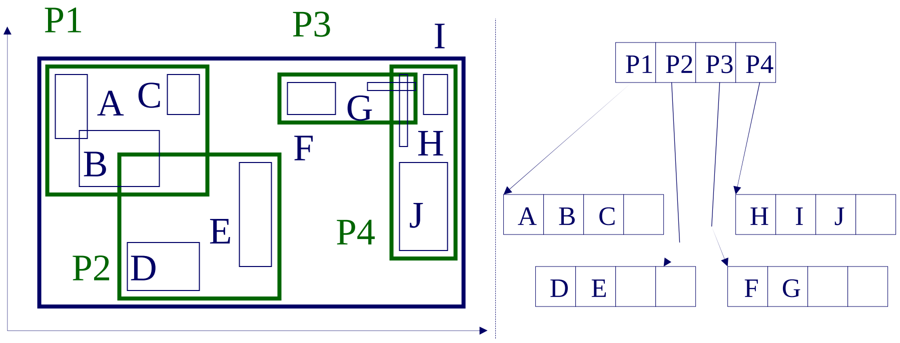
H I J

D E

F G

R-trees

- eg., w/ fanout 4:



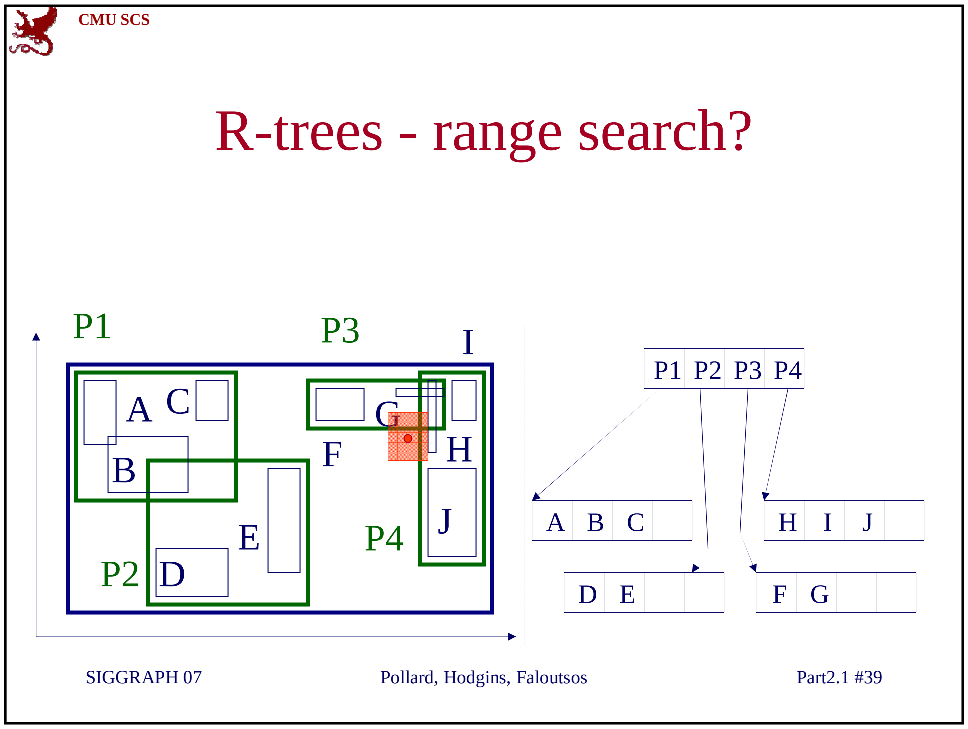
P1 P2 P3 P4

A B C

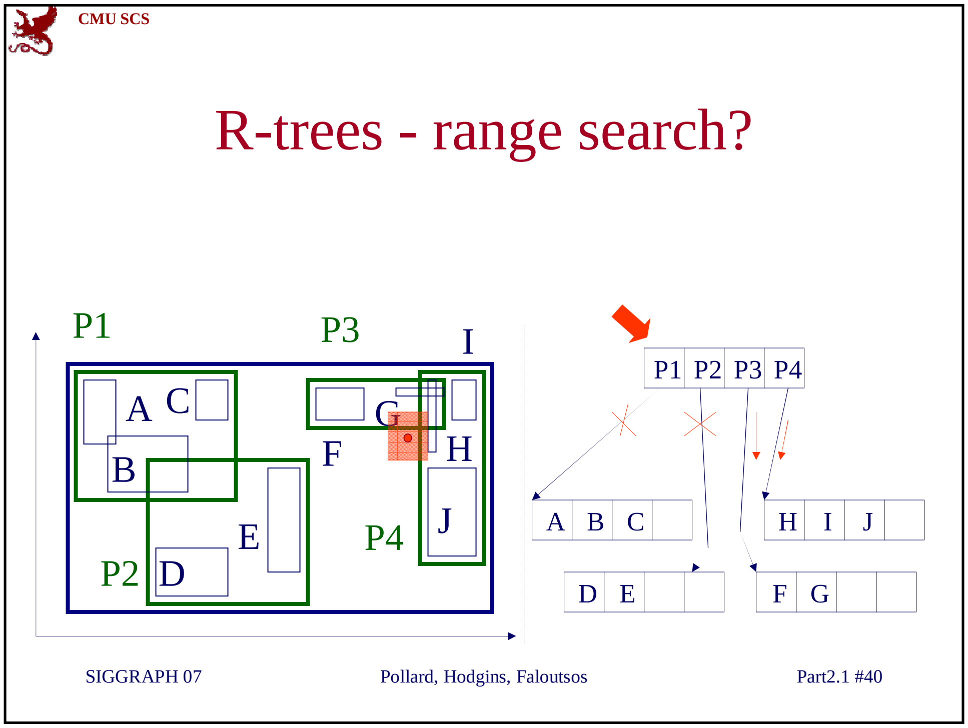
H I J

D E

F G



R-trees - range search?

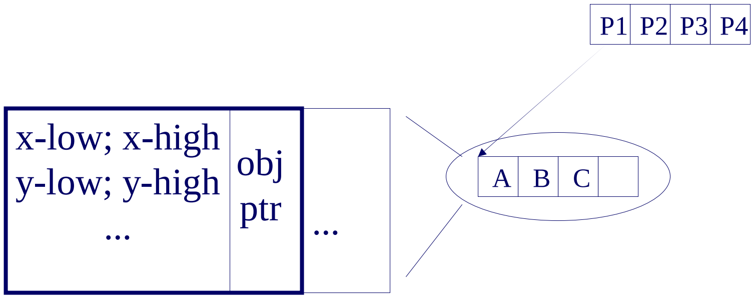


R-trees - range search?

skip

R-trees - format of nodes

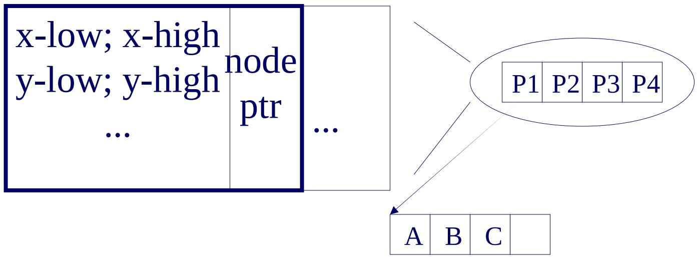
- {(MBR; obj-ptr)} for leaf nodes



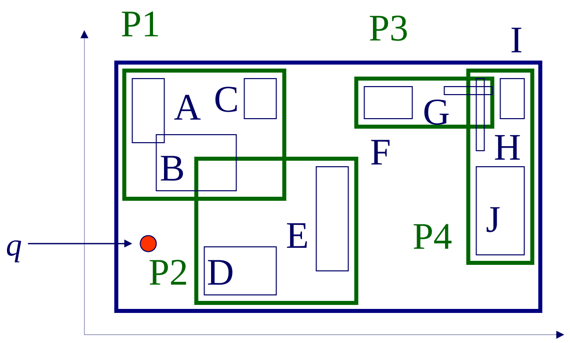
skip

R-trees - format of nodes

- {(MBR; node-ptr)} for non-leaf nodes

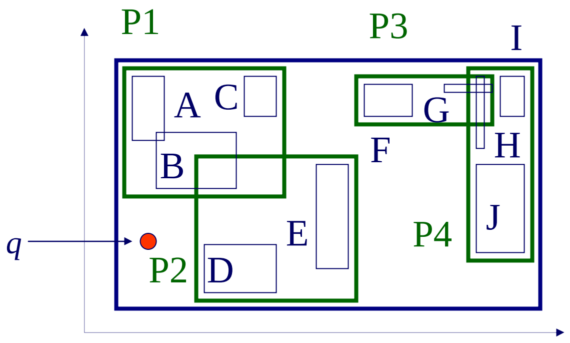


R-trees - nn search



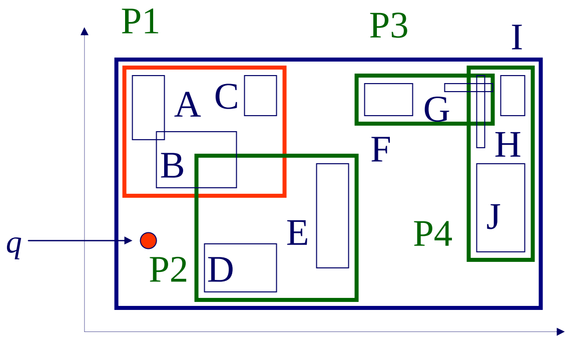
R-trees - nn search

- Q: How? (find near neighbor; refine...)



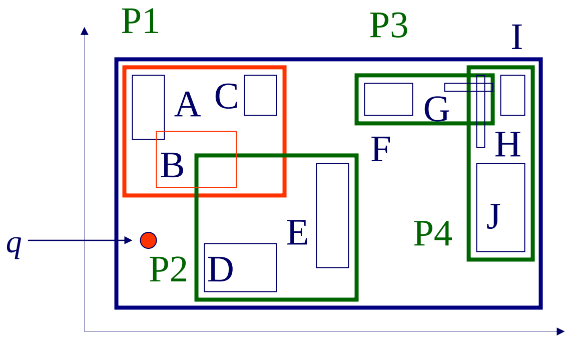
R-trees - nn search

- A1: depth-first search; then, range query



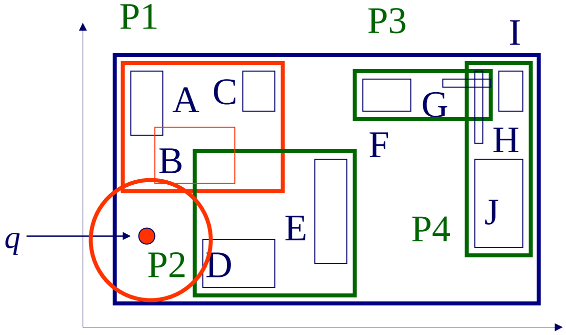
R-trees - nn search

- A1: depth-first search; then, range query



R-trees - nn search

- A1: depth-first search; then, range query



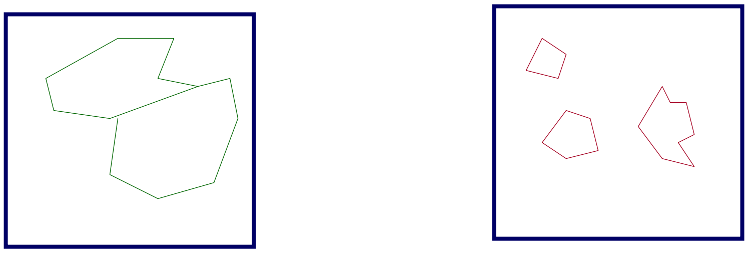
R-trees - nn search

- A2: [Roussopoulos+, sigmod95]:
  - priority queue, with promising MBRs, and their best and worst-case distance

R-trees - spatial joins

Spatial joins: find (quickly) all

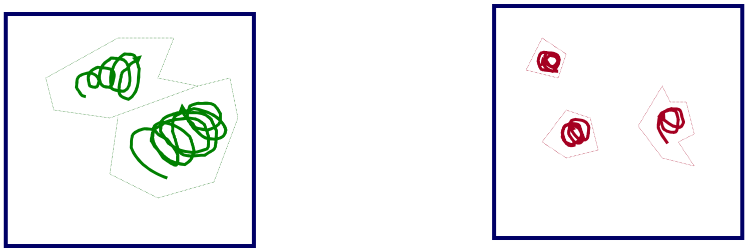
counties intersecting lakes



R-trees - spatial joins

Spatial joins: find (quickly) all

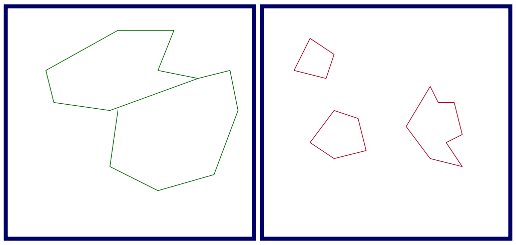
runs similar to walks



R-trees - spatial joins

Spatial joins: find (quickly) all

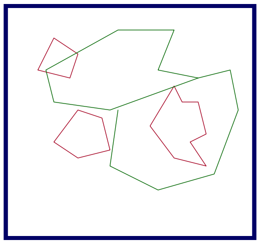
counties intersecting lakes



R-trees - spatial joins

Spatial joins: find (quickly) all

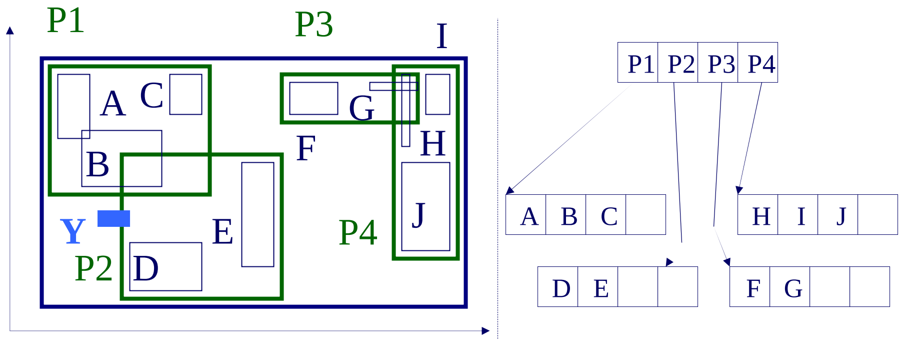
counties intersecting lakes



skip

R-trees - insertion

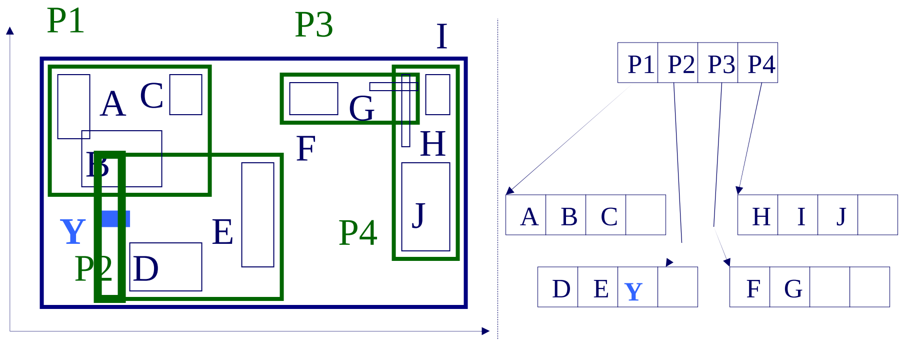
- eg., rectangle ‘Y’



skip

R-trees - insertion

- eg., rectangle ‘Y’: extend suitable parent.

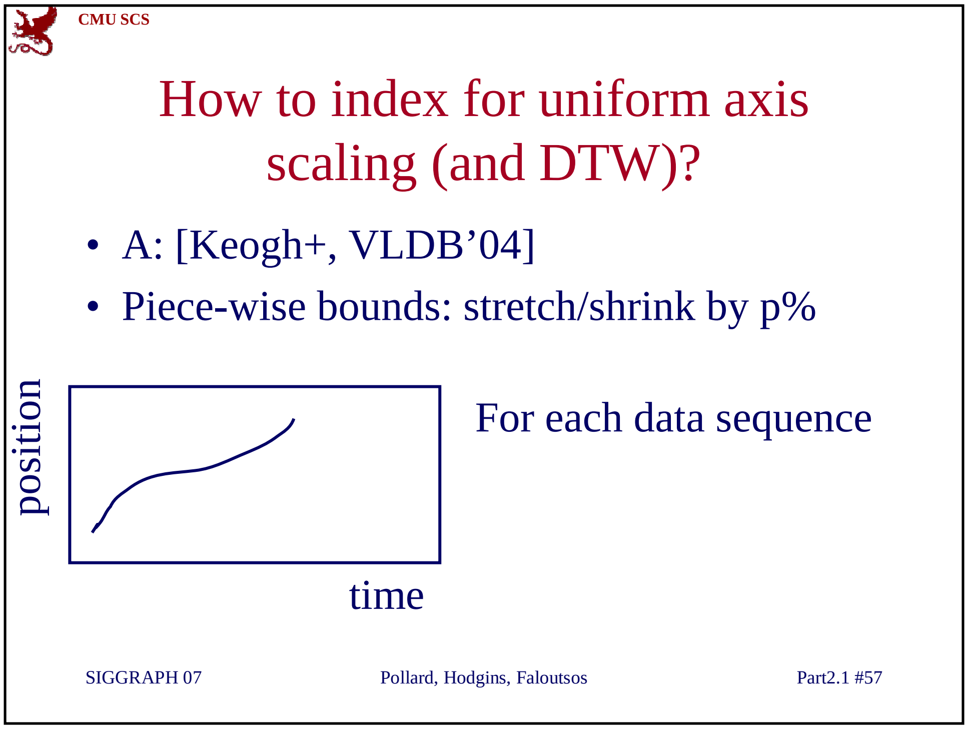


R-trees - insertion

- On leaf page over-flow: split
  - – And possibly, propagate the split upwards
  - – (~like B-trees)
- A *lot* of variations, for insertion and deletion
- Initial R-trees, and ``R*-trees'', are good choices.

Outline

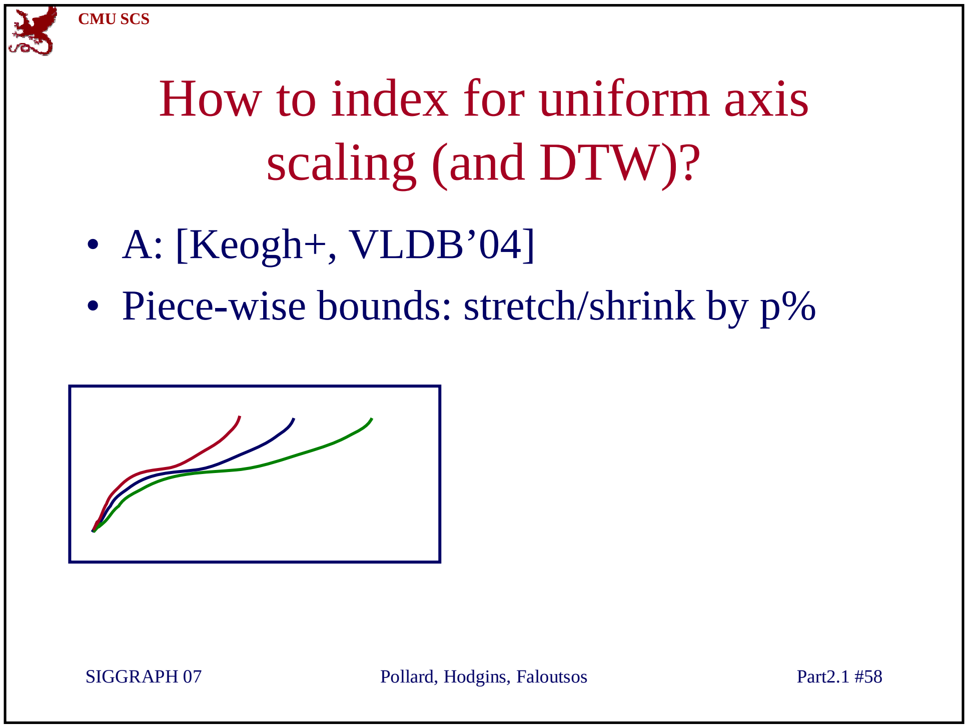
- Similarity search and indexing
  - – distance functions
  - ➡ – R-trees; **discussion**; M-trees
- Feature extraction and dim. Reduction
- Linear Forecasting



How to index for uniform axis scaling (and DTW)?

- A: [Keogh+, VLDB’04]
- Piece-wise bounds: stretch/shrink by p%

For each data sequence

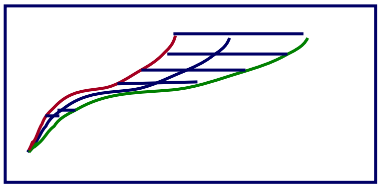


How to index for uniform axis scaling (and DTW)?

- A: [Keogh+, VLDB’04]
- Piece-wise bounds: stretch/shrink by p%

How to index for uniform axis scaling (and DTW)?

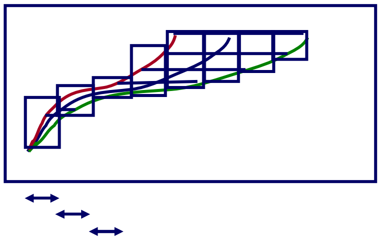
- A: [Keogh+, VLDB'04]
- Piece-wise bounds: stretch/shrink by p%



Consider the shaded area
And cover it with MBRs

How to index for uniform axis scaling (and DTW)?

- A: [Keogh+, VLDB'04]
- Piece-wise bounds: stretch/shrink by p%



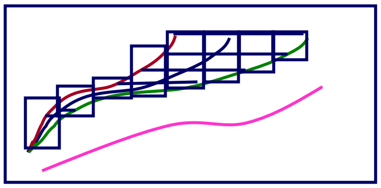
Same duration

Consider the shaded area
And cover it with MBRs
And use them for

- lower-bounding
- indexing

How to index for uniform axis scaling (and DTW)?

- A: [Keogh+, VLDB'04]
- Piece-wise bounds: stretch/shrink by p%



Just from the MBRs, the blue sequence does NOT qualify wrt the pink one

Outline

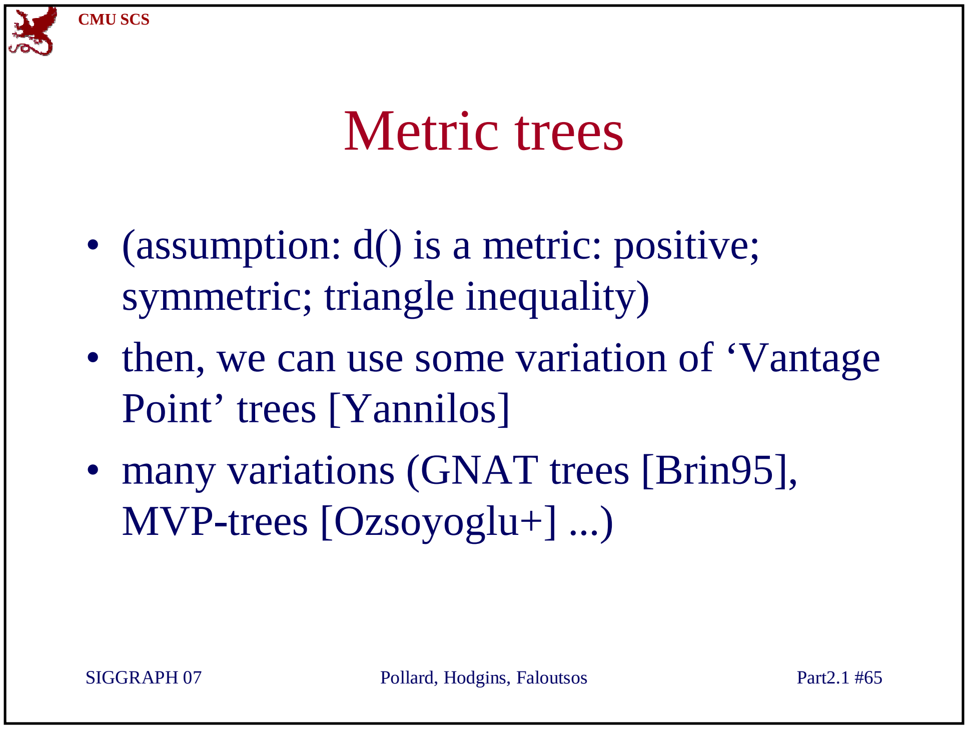
- Similarity search and indexing
  - distance functions
  - R-trees; discussion; **M-trees**
- Feature extraction and dim. Reduction
- Linear Forecasting

Metric trees

- Additional variations, when we don’t want to extract features:
- M-trees (=metric trees) [Ciaccia+]
  - Spheres within spheres
- OMNI-trees [Filho+]: choose vantage points

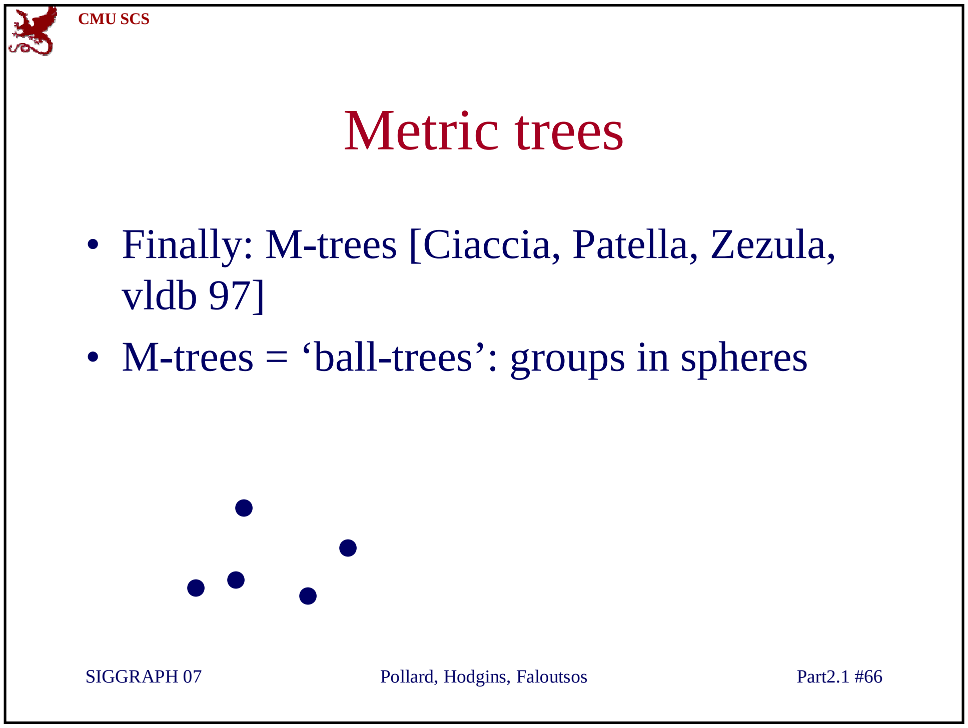
Metric trees

- What if we only have a distance function d(o1, o2)?



Metric trees

- (assumption: d() is a metric: positive; symmetric; triangle inequality)
- then, we can use some variation of ‘Vantage Point’ trees [Yannilos]
- many variations (GNAT trees [Brin95], MVP-trees [Ozsoyoglu+] ...)

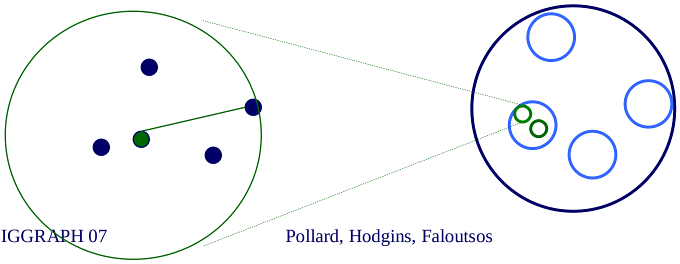


Metric trees

- Finally: M-trees [Ciaccia, Patella, Zezula, vldb 97]
- M-trees = ‘ball-trees’: groups in spheres

Metric trees

- Finally: M-trees [Ciaccia, Patella, Zezula, vldb 97]
- M-trees = ‘ball-trees’: Minimum Bounding spheres



Metric trees

- Search (range and k-nn): like R-trees
- Split? Several criteria (see paper)

Outline

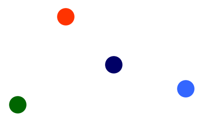
- Similarity search and indexing
  - – distance functions
  - – R-trees; discussion; M-trees, **OMNI family**
- Feature extraction and dim. Reduction
- Linear Forecasting

Metric trees - OMNI trees

- How to turn objects into vectors?
- (assume that distance computations are expensive; we need to answer range/nn queries quickly)

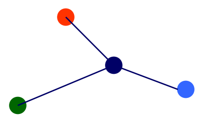
Metric trees - OMNI trees

- How to turn objects into vectors?
- A: pick *n* ‘anchor’ objects; record the distance of each object from them -> *n*-d vector



Metric trees - OMNI trees

- How to turn objects into vectors?
- A: pick *n* ‘anchor’ objects; record the distance of each object from them -> *n*-d vector



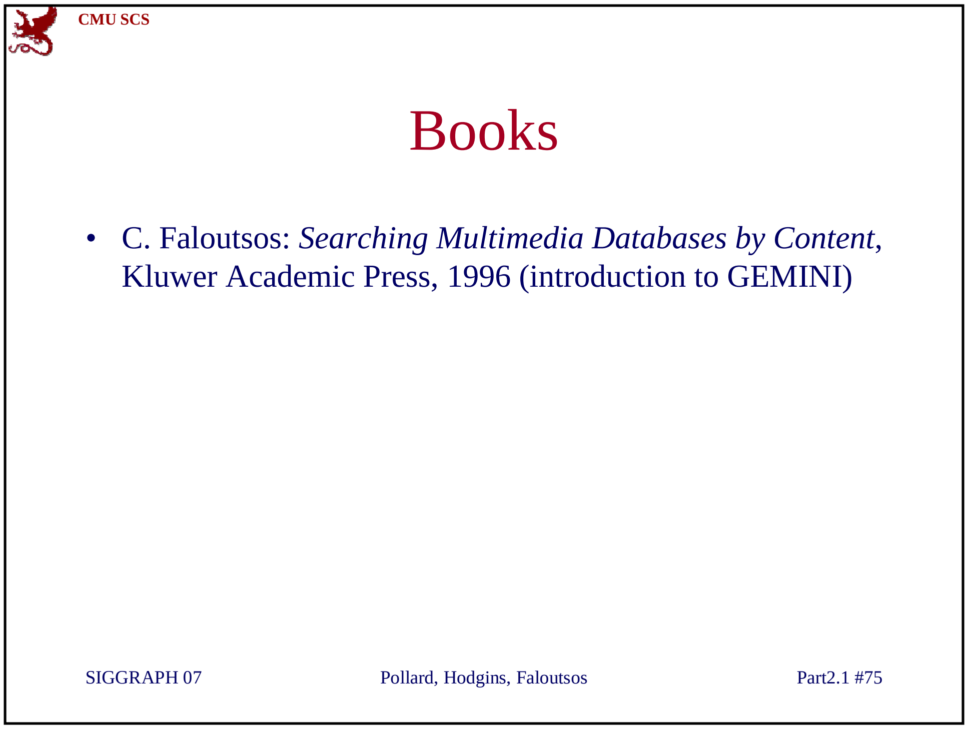
Metric trees - OMNI trees

- Result: faster than M-trees and seq. scanning (especially if distance computations are expensive)
- Can answer all the usual types of queries (range, nearest neighbors, etc)

Conclusions

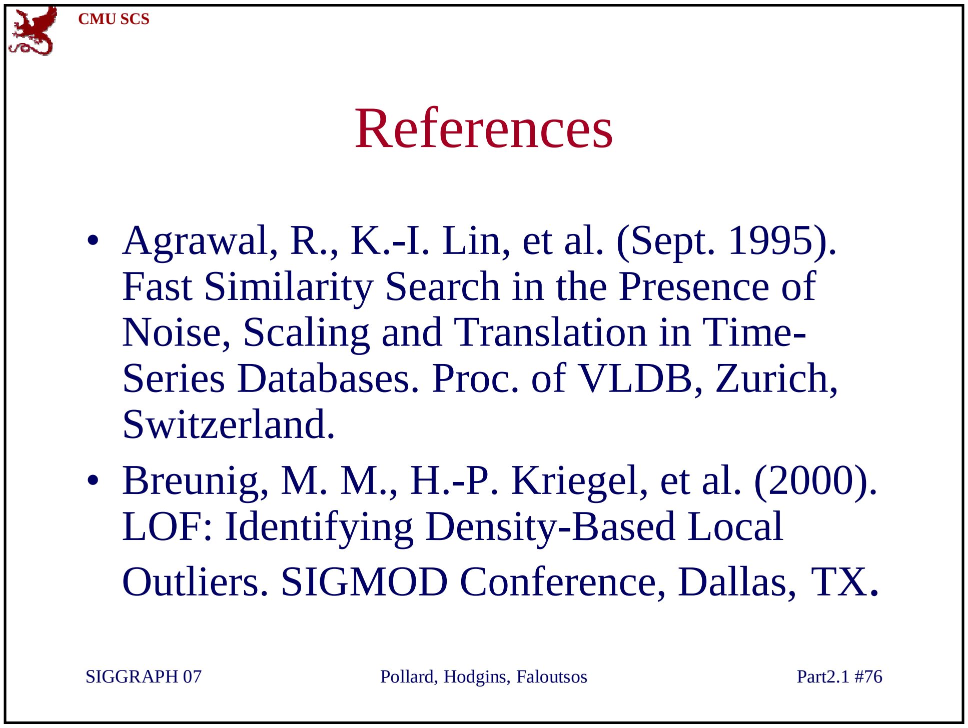
- Fast indexing: through GEMINI
  - feature extraction and
  - (off the shelf) Spatial Access Methods [Gaede+98]
- or metric trees (M-trees, OMNI)

- Subtle point: for high *intrinsic* dimensionality, use seq. scan.



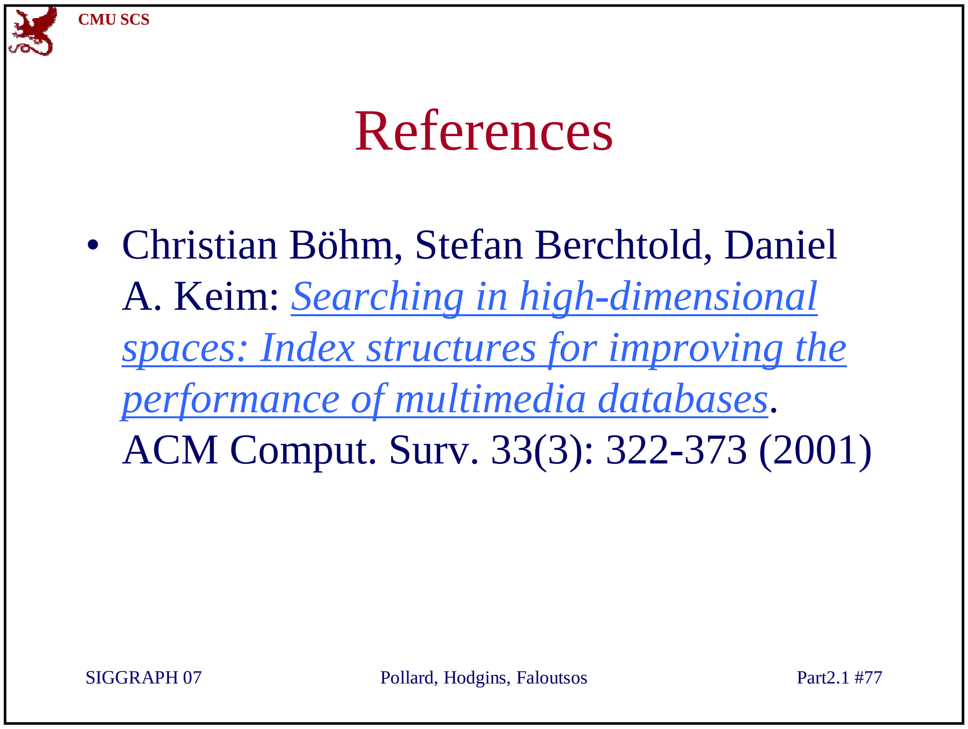
Books

- C. Faloutsos: *Searching Multimedia Databases by Content*, Kluwer Academic Press, 1996 (introduction to GEMINI)



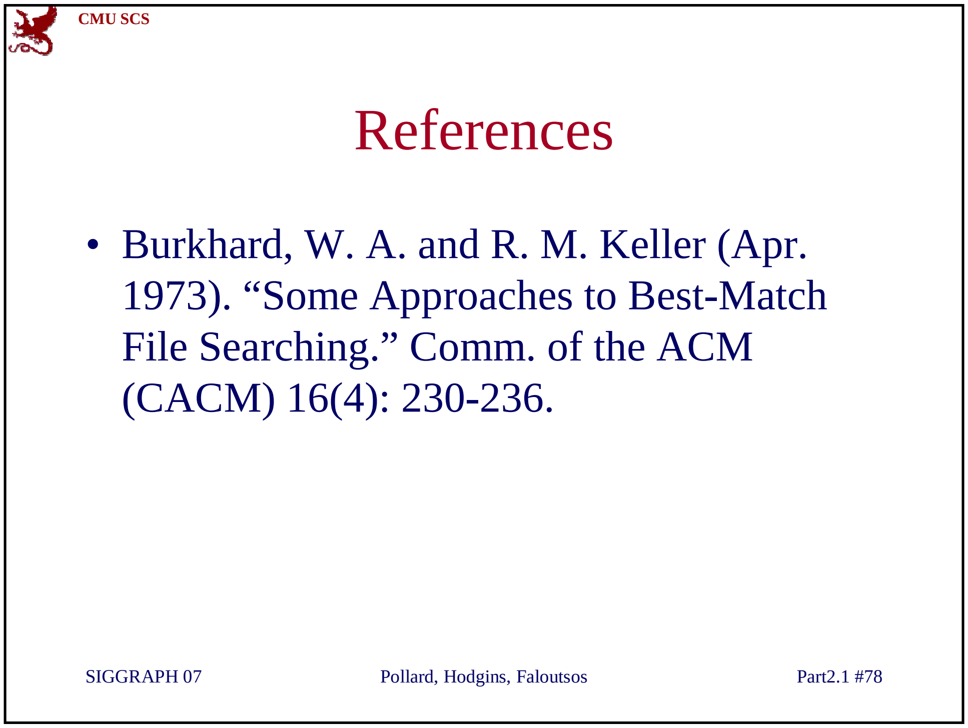
References

- Agrawal, R., K.-I. Lin, et al. (Sept. 1995). Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases. Proc. of VLDB, Zurich, Switzerland.
- Breunig, M. M., H.-P. Kriegel, et al. (2000). LOF: Identifying Density-Based Local Outliers. SIGMOD Conference, Dallas, TX.



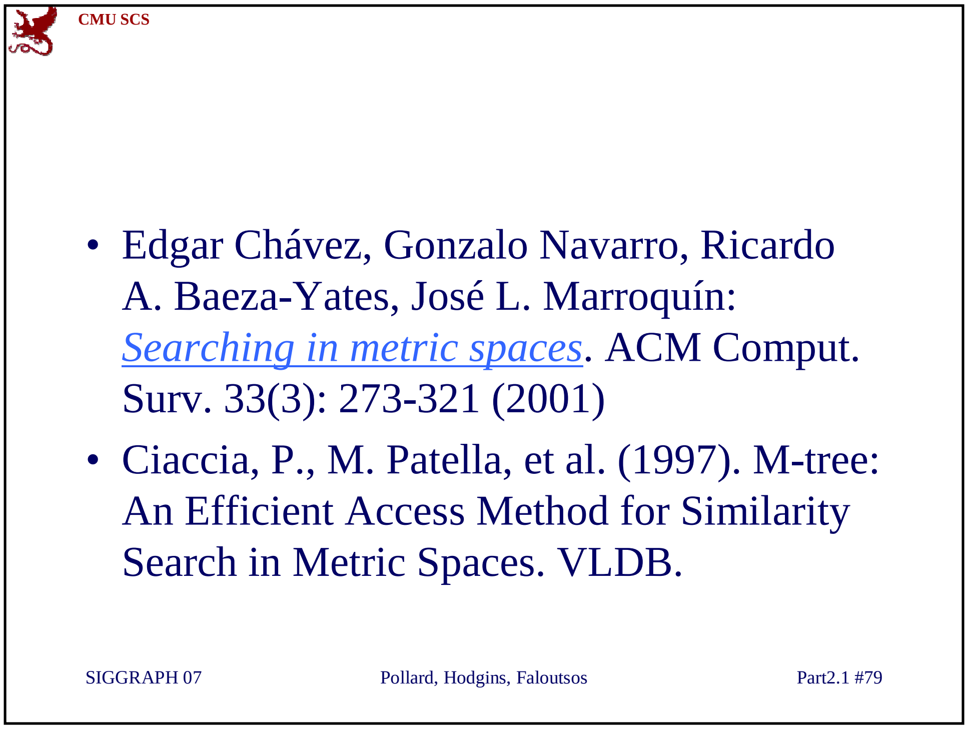
References

- Christian Böhm, Stefan Berchtold, Daniel A. Keim: *Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases*. ACM Comput. Surv. 33(3): 322-373 (2001)

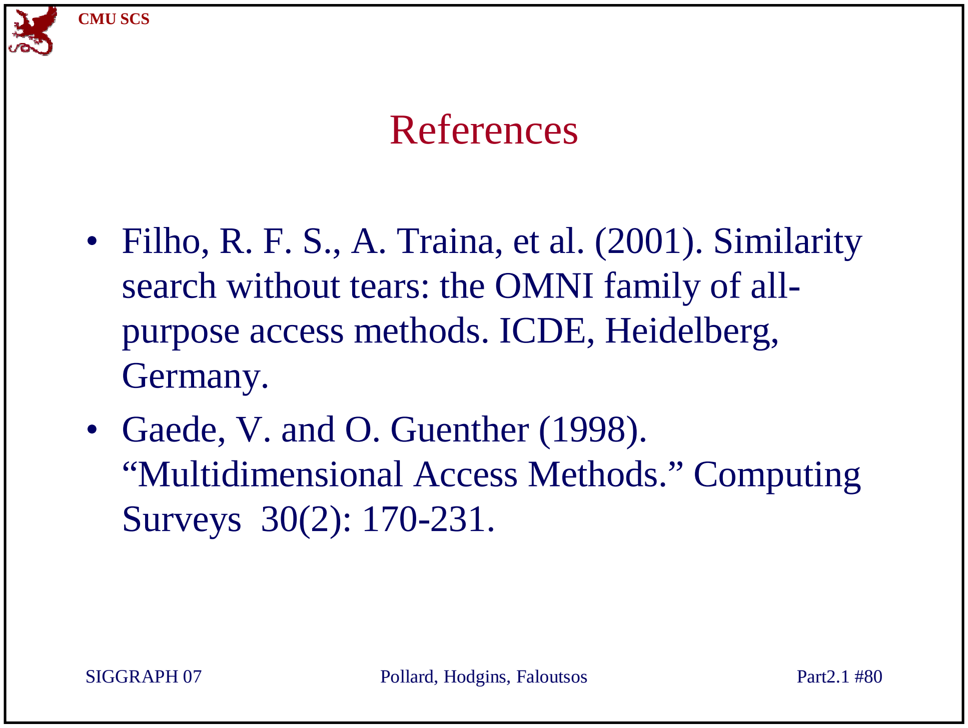


References

- Burkhard, W. A. and R. M. Keller (Apr. 1973). “Some Approaches to Best-Match File Searching.” Comm. of the ACM (CACM) 16(4): 230-236.

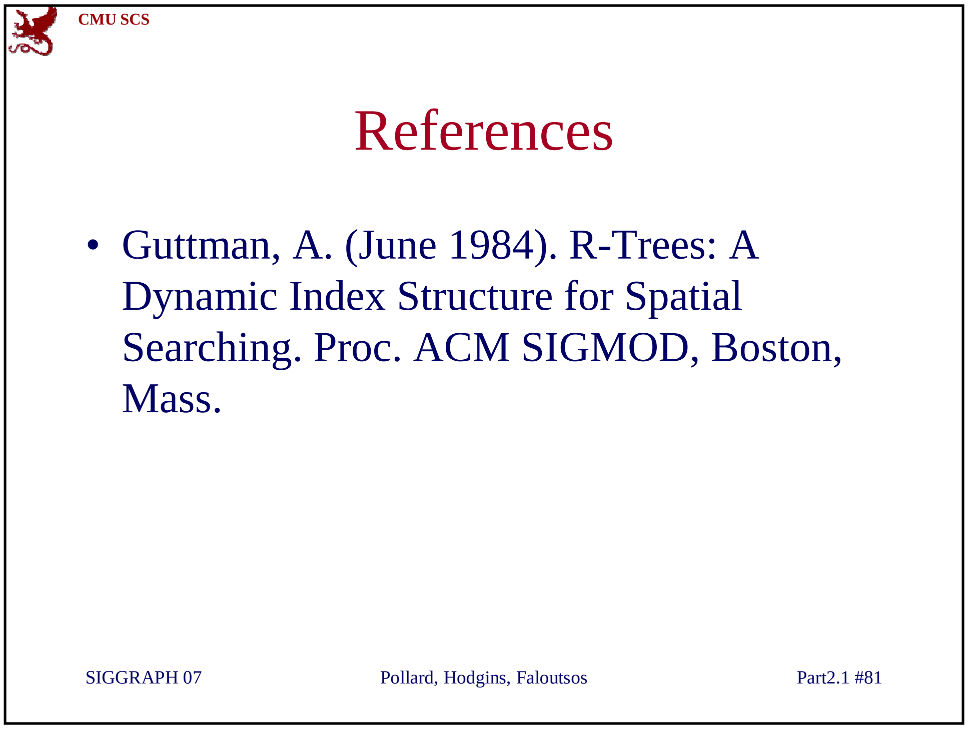


- Edgar Chávez, Gonzalo Navarro, Ricardo A. Baeza-Yates, José L. Marroquín: *Searching in metric spaces*. ACM Comput. Surv. 33(3): 273-321 (2001)
- Ciaccia, P., M. Patella, et al. (1997). M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. VLDB.



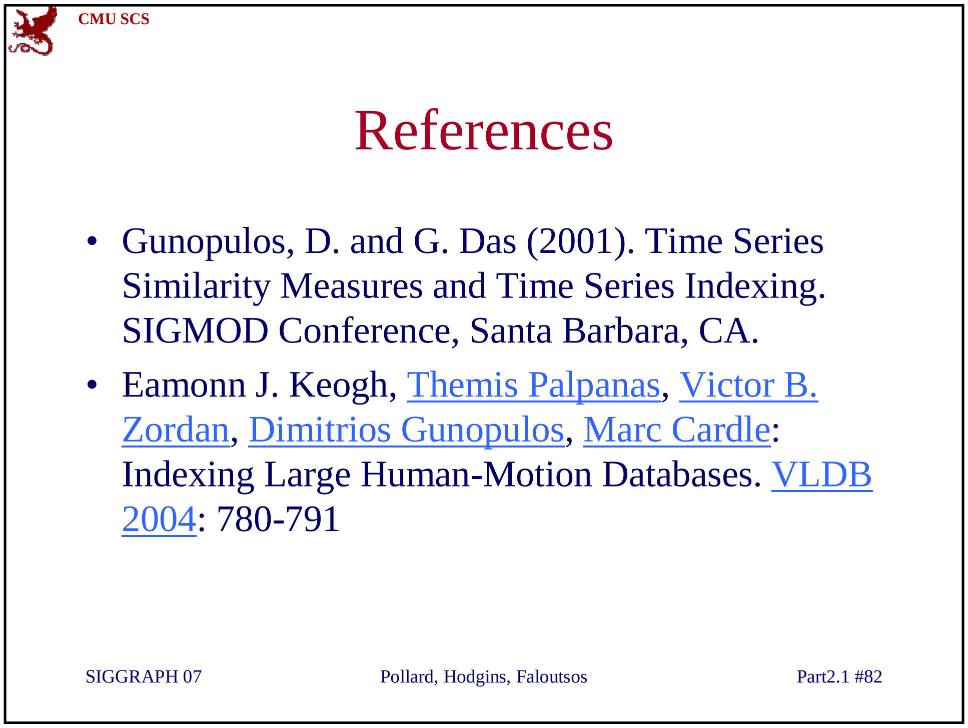
References

- Filho, R. F. S., A. Traina, et al. (2001). Similarity search without tears: the OMNI family of all-purpose access methods. ICDE, Heidelberg, Germany.
- Gaede, V. and O. Guenther (1998). “Multidimensional Access Methods.” Computing Surveys 30(2): 170-231.



References

- Guttman, A. (June 1984). R-Trees: A Dynamic Index Structure for Spatial Searching. Proc. ACM SIGMOD, Boston, Mass.



References

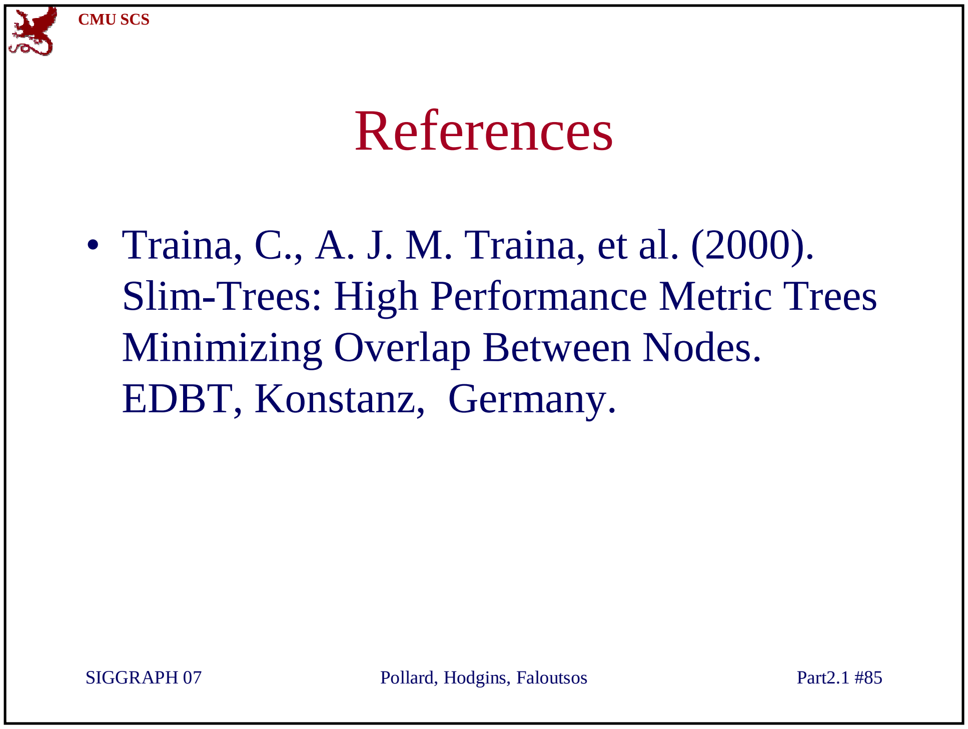
- Gunopulos, D. and G. Das (2001). Time Series Similarity Measures and Time Series Indexing. SIGMOD Conference, Santa Barbara, CA.
- Eamonn J. Keogh, Themis Palpanas, Victor B. Zordan, Dimitrios Gunopulos, Marc Cardle: Indexing Large Human-Motion Databases. VLDB 2004: 780-791

References

- Keogh, E. J., K. Chakrabarti, et al. (2001). Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. SIGMOD Conference, Santa Barbara, CA.
- Rabiner, L. and B.-H. Juang (1993). Fundamentals of Speech Recognition, Prentice Hall.

References

- Shapiro, M. (May 1977). “The Choice of Reference Points in Best-Match File Searching.” Comm. of the ACM (CACM) 20(5): 339-343.
- Shasha, D. and T.-L. Wang (Apr. 1990). “New Techniques for Best-Match Retrieval.” ACM TOIS 8(2): 140-158.



References

- Traina, C., A. J. M. Traina, et al. (2000). Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes. EDBT, Konstanz, Germany.

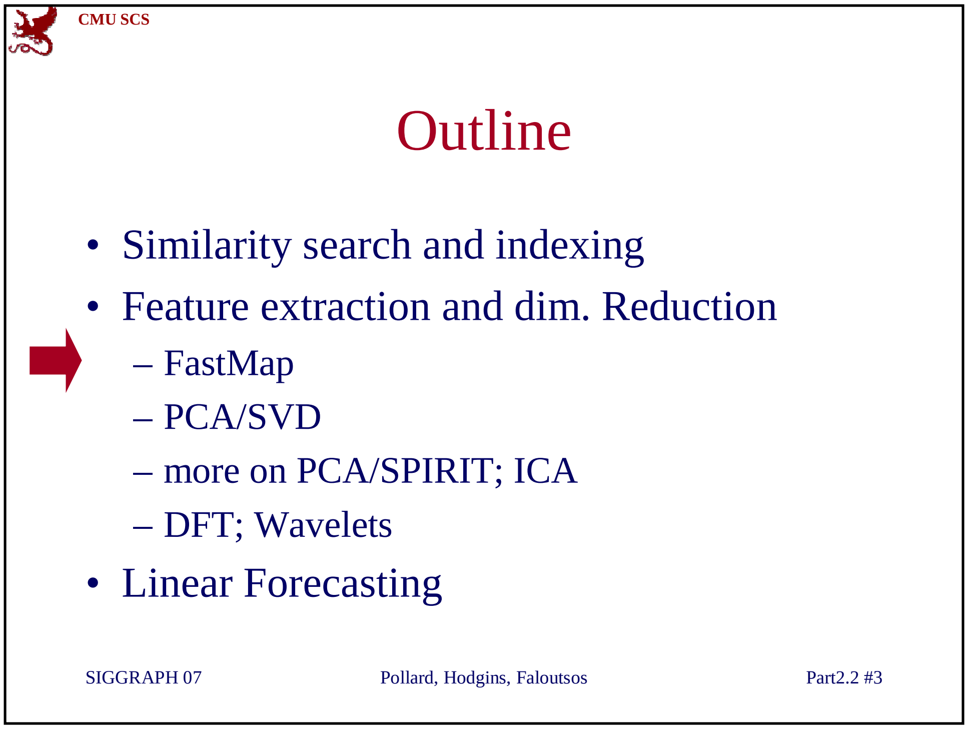
Database Techniques with Motion Capture

Part 2: Methods

Christos Faloutsos

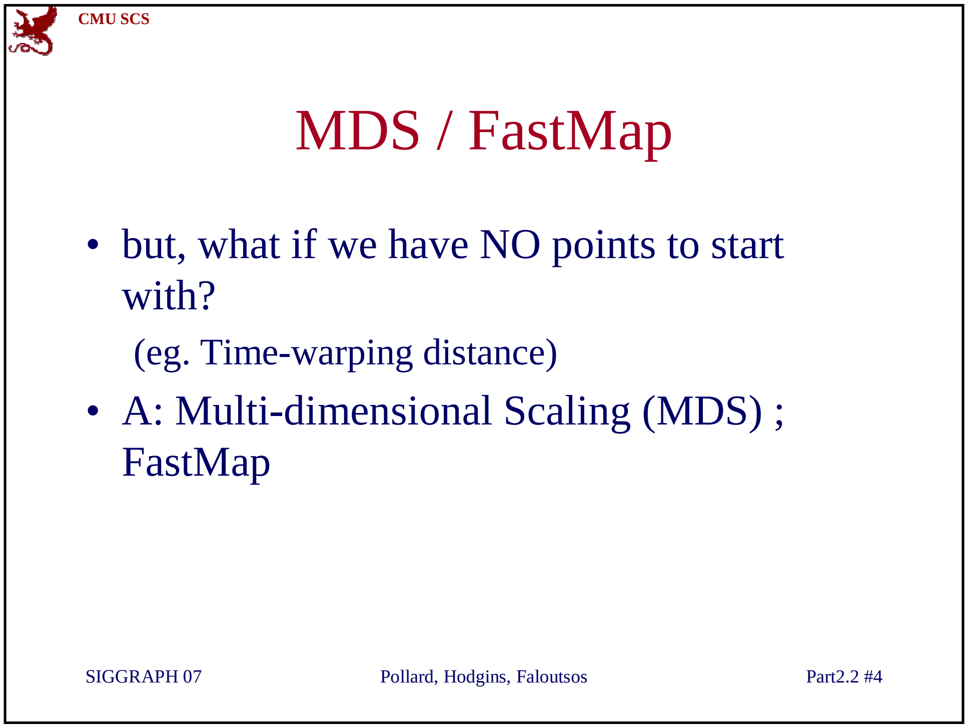
CMU

Part 2.2 Feature extraction and dim. reduction



Outline

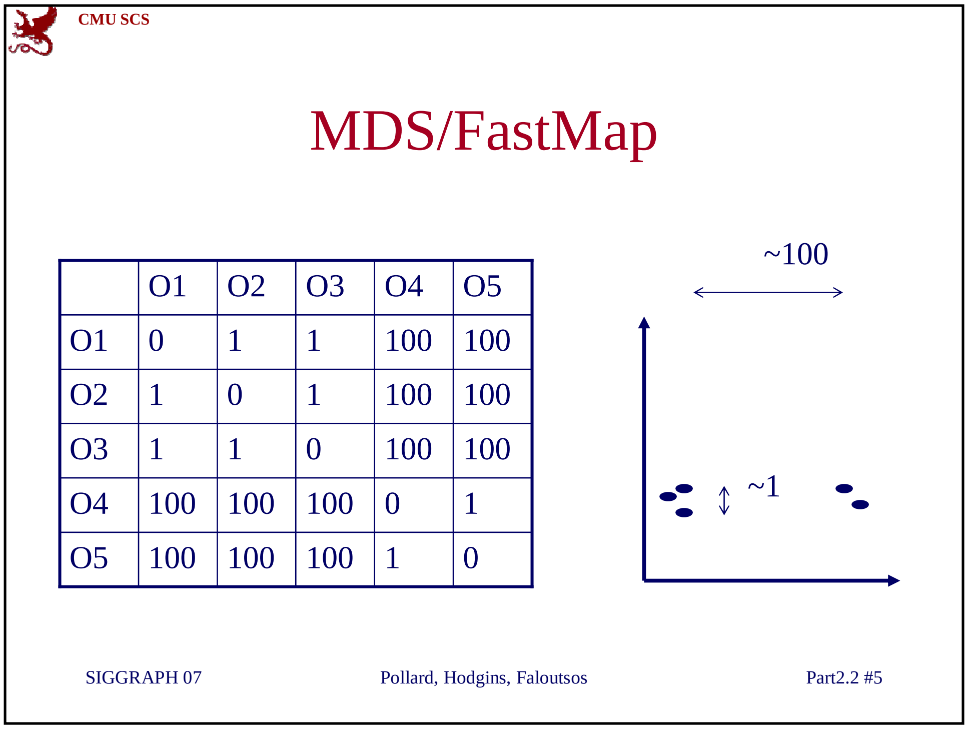
- Similarity search and indexing
- Feature extraction and dim. Reduction
  - FastMap
  - PCA/SVD
  - more on PCA/SPIRIT; ICA
  - DFT; Wavelets
- Linear Forecasting



MDS / FastMap

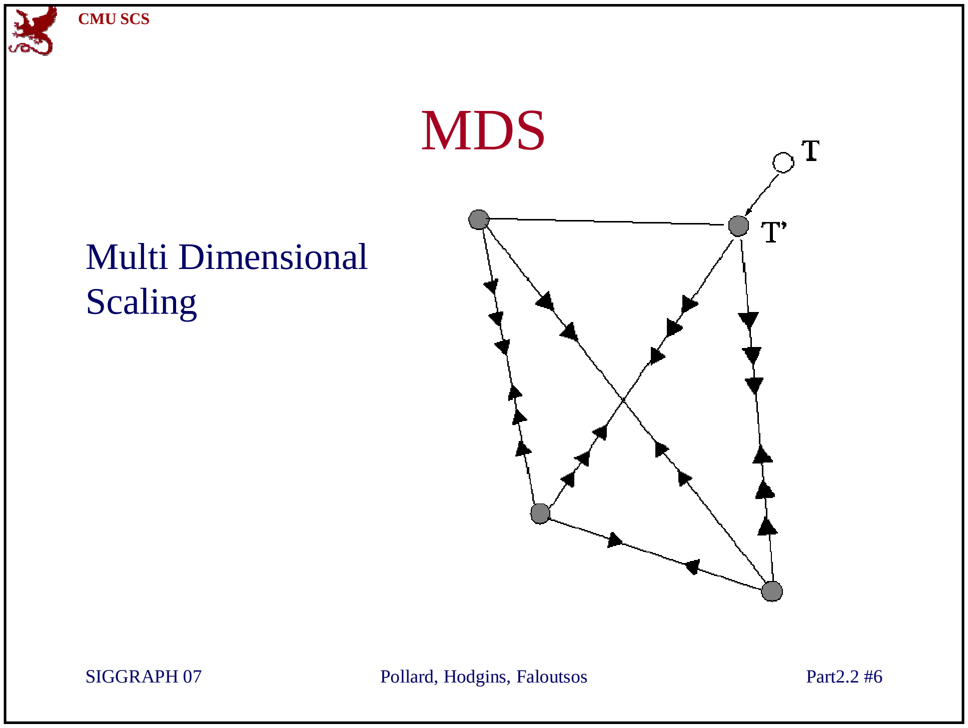
- but, what if we have NO points to start with?

  (eg. Time-warping distance)
- A: Multi-dimensional Scaling (MDS) ; FastMap



MDS/FastMap

	O1	O2	O3	O4	O5
O1	0	1	1	100	100
O2	1	0	1	100	100
O3	1	1	0	100	100
O4	100	100	100	0	1
O5	100	100	100	1	0



MDS

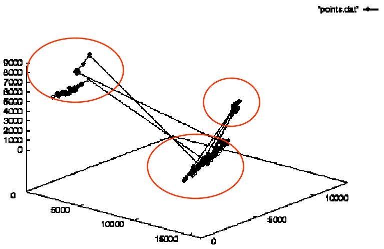
Multi Dimensional Scaling

FastMap

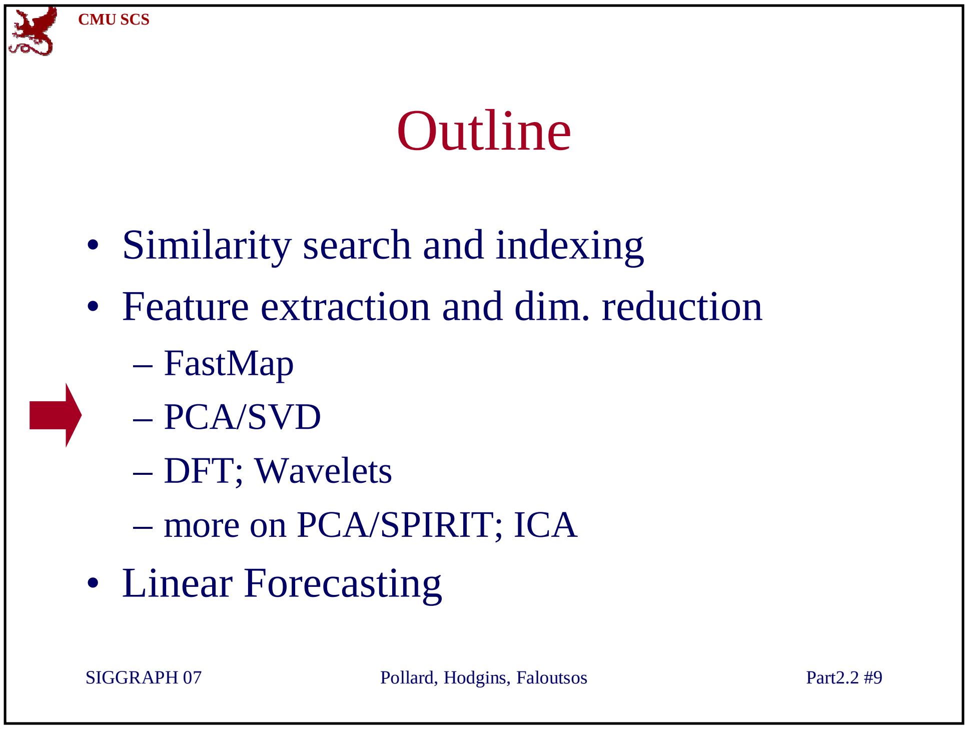
- Multi-dimensional scaling (MDS) can do that, but in O(N**2) time
- FastMap [Faloutsos+95] takes O(N) time

FastMap: Application

VideoTrails [Kobla+97]

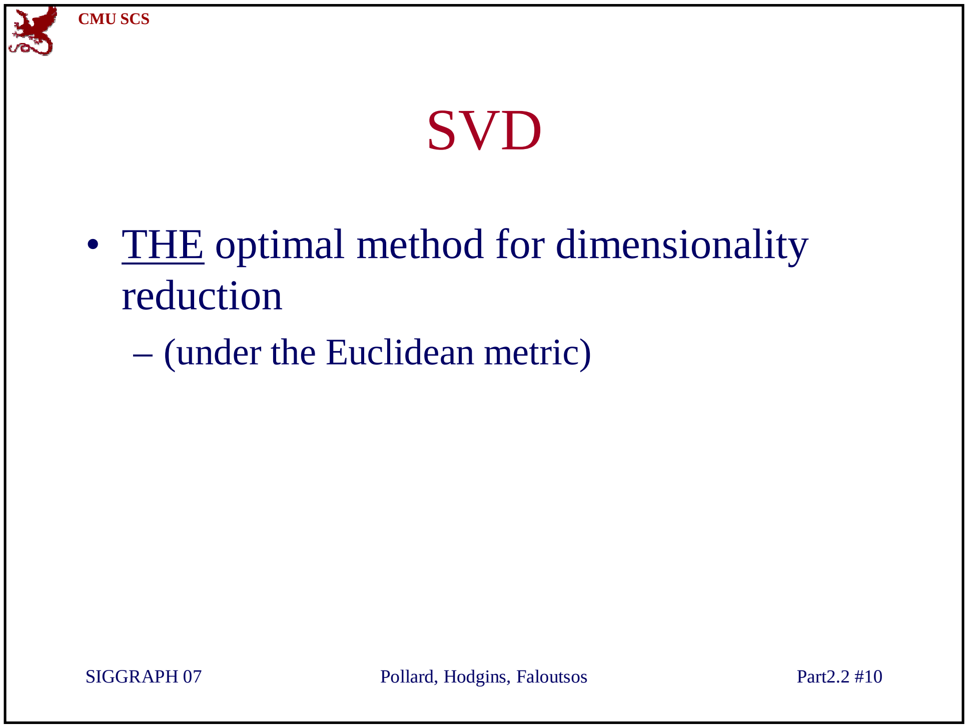


scene-cut detection (about 10% errors)



Outline

- Similarity search and indexing
- Feature extraction and dim. reduction
  - – FastMap
  - – PCA/SVD
  - – DFT; Wavelets
  - – more on PCA/SPIRIT; ICA
- Linear Forecasting



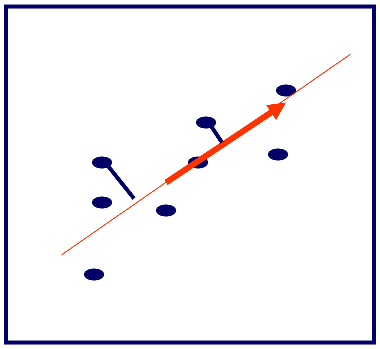
SVD

- THE optimal method for dimensionality reduction
  - – (under the Euclidean metric)

Singular Value Decomposition (SVD)

- SVD (~LSI ~ KL ~ PCA ~ spectral analysis...)

Right elbow



LSI: S. Dumais; M. Berry

KL: eg, Duda+Hart

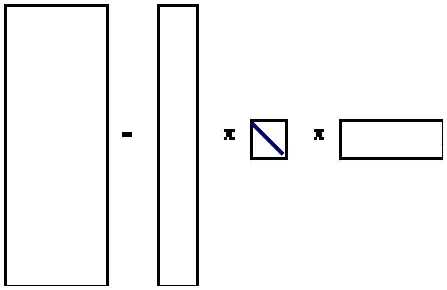
PCA: eg., Jolliffe

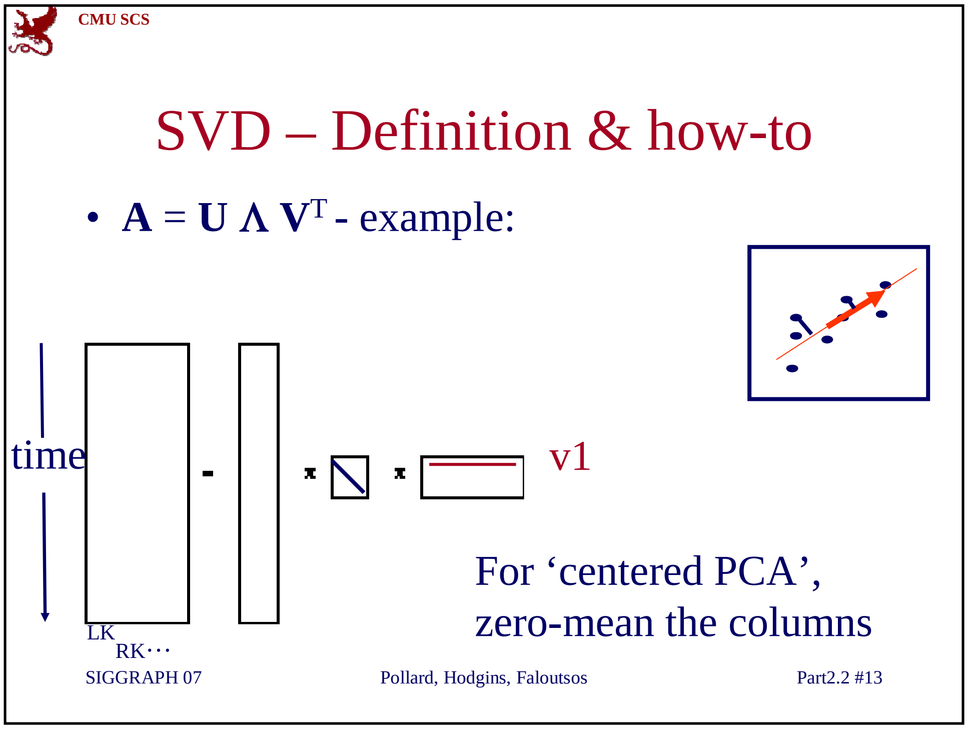
Details: [Press+], [Faloutsos96]

Left knee

SVD – Definition & how-to

- **A** = **U** **Λ** **V**^{T} - example:





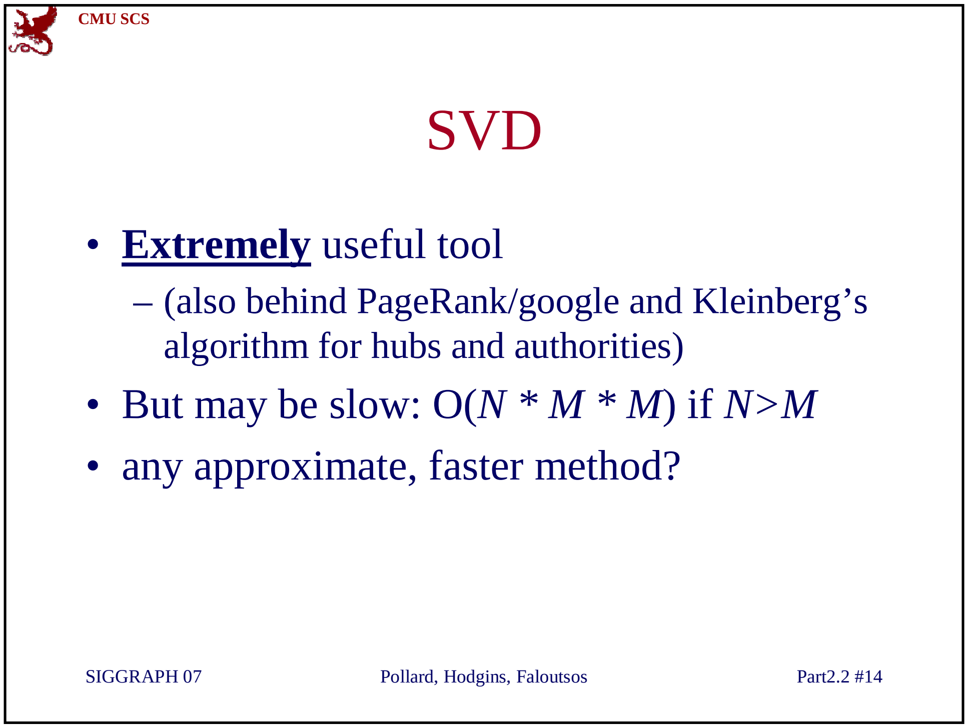
CMU SCS

SVD – Definition & how-to

- $\mathbf{A} = \mathbf{U}\,\mathbf{\Lambda}\,\mathbf{V}^T$ - example:

For 'centered PCA', zero-mean the columns

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #13



CMU SCS

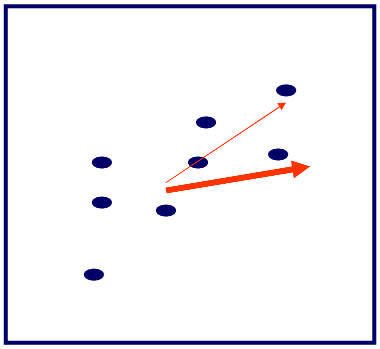
SVD

- **Extremely** useful tool
  - (also behind PageRank/google and Kleinberg's algorithm for hubs and authorities)
- But may be slow: O(N * M * M) if N>M
- any approximate, faster method?

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #14

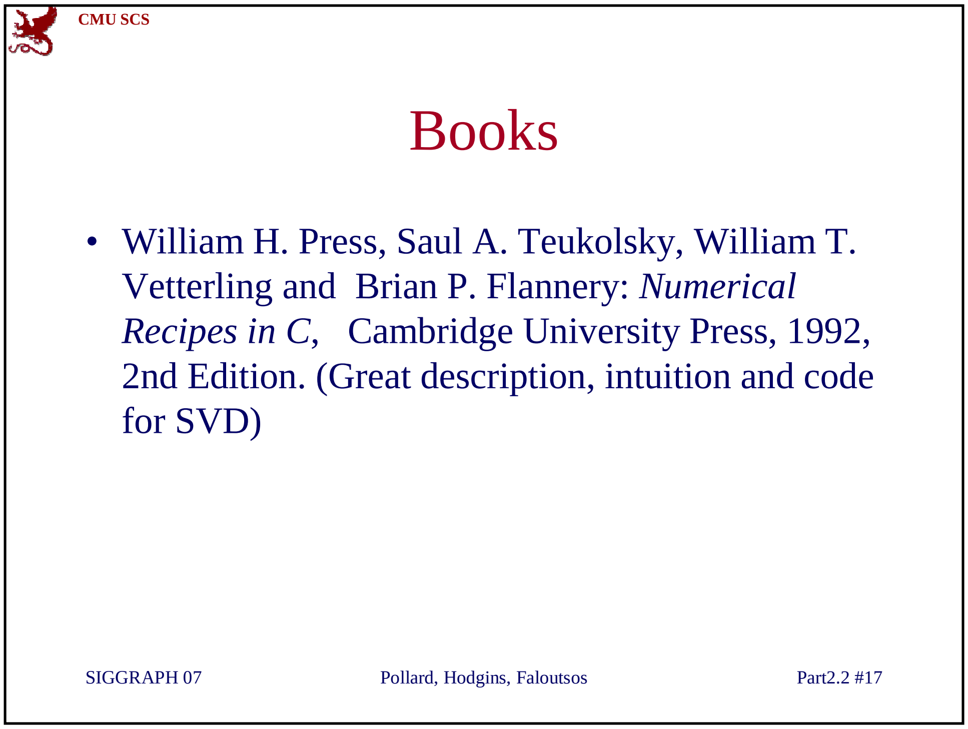
SVD shorcuts

- random projections (Johnson-Lindenstrauss THM [Papadimitriou+ pods98])



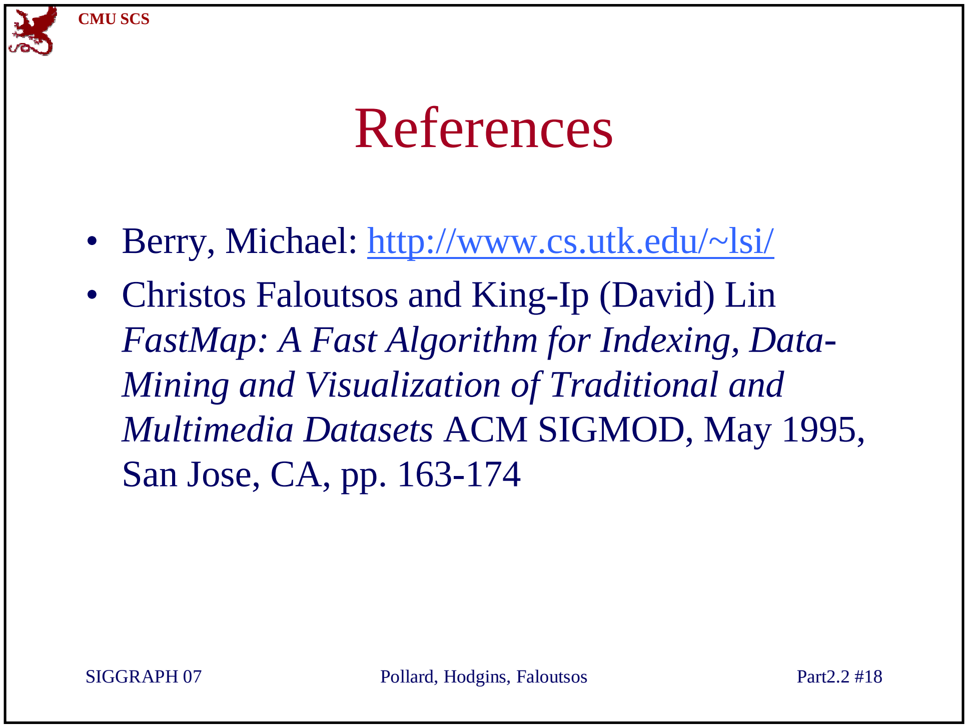
Random projections

- pick ‘enough’ random directions (will be ~orthogonal, in high-d!!)
- distances are preserved probabilistically, within epsilon
- (also, use as a pre-processing step for SVD [Papadimitriou+ PODS98])



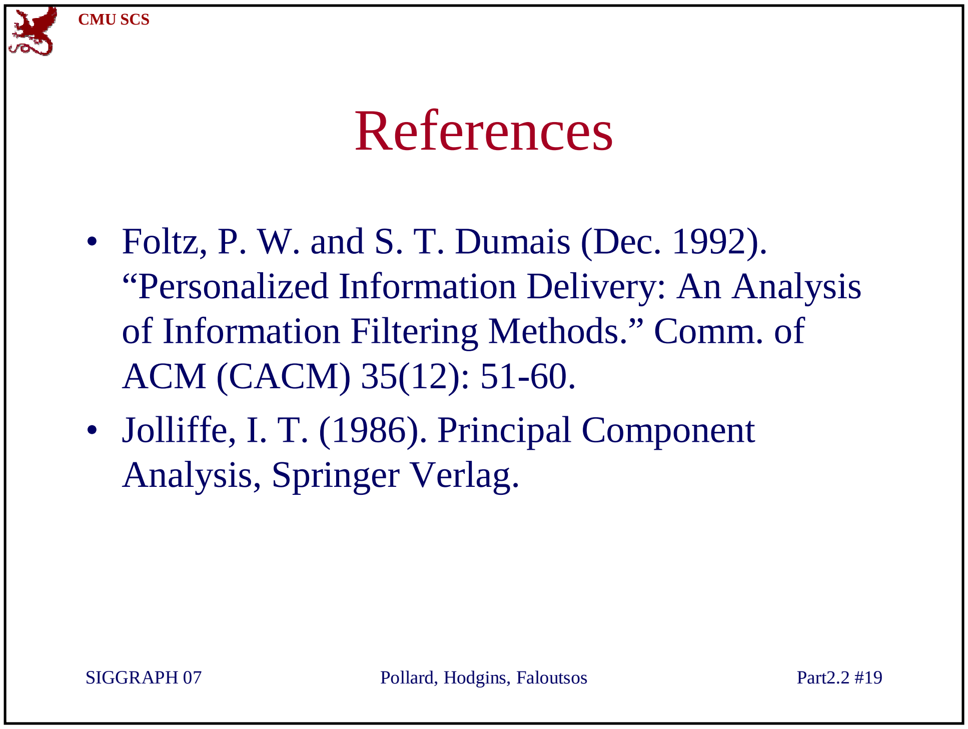
Books

- William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery: *Numerical Recipes in C*,   Cambridge University Press, 1992, 2nd Edition. (Great description, intuition and code for SVD)



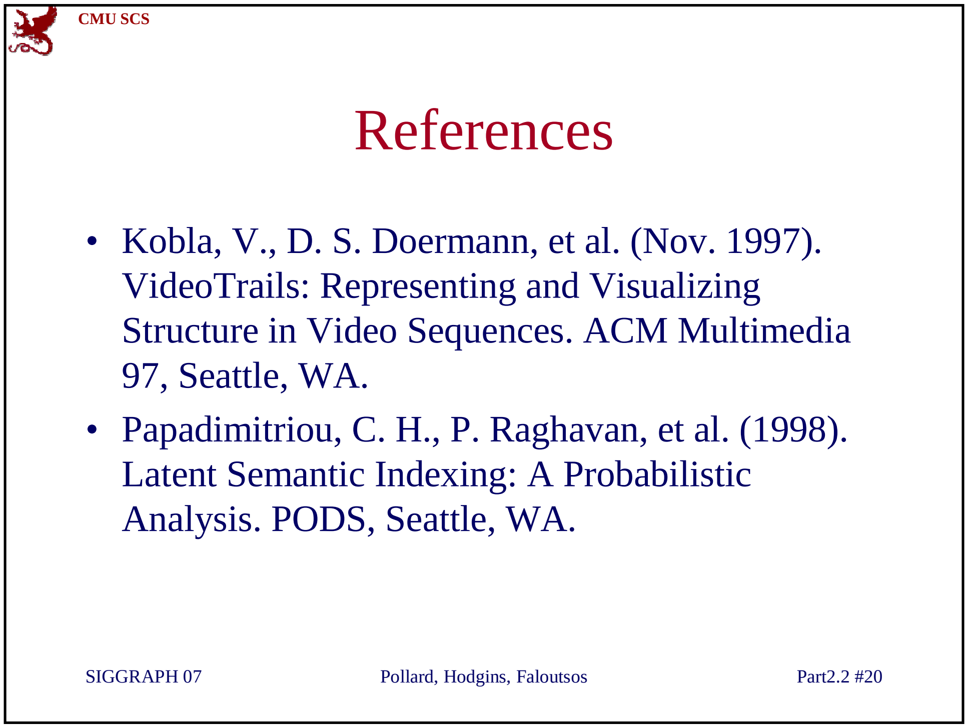
References

- Berry, Michael: http://www.cs.utk.edu/~lsi/
- Christos Faloutsos and King-Ip (David) Lin *FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets* ACM SIGMOD, May 1995, San Jose, CA, pp. 163-174



References

- Foltz, P. W. and S. T. Dumais (Dec. 1992). “Personalized Information Delivery: An Analysis of Information Filtering Methods.” Comm. of ACM (CACM) 35(12): 51-60.
- Jolliffe, I. T. (1986). Principal Component Analysis, Springer Verlag.



References

- Kobla, V., D. S. Doermann, et al. (Nov. 1997). VideoTrails: Representing and Visualizing Structure in Video Sequences. ACM Multimedia 97, Seattle, WA.
- Papadimitriou, C. H., P. Raghavan, et al. (1998). Latent Semantic Indexing: A Probabilistic Analysis. PODS, Seattle, WA.

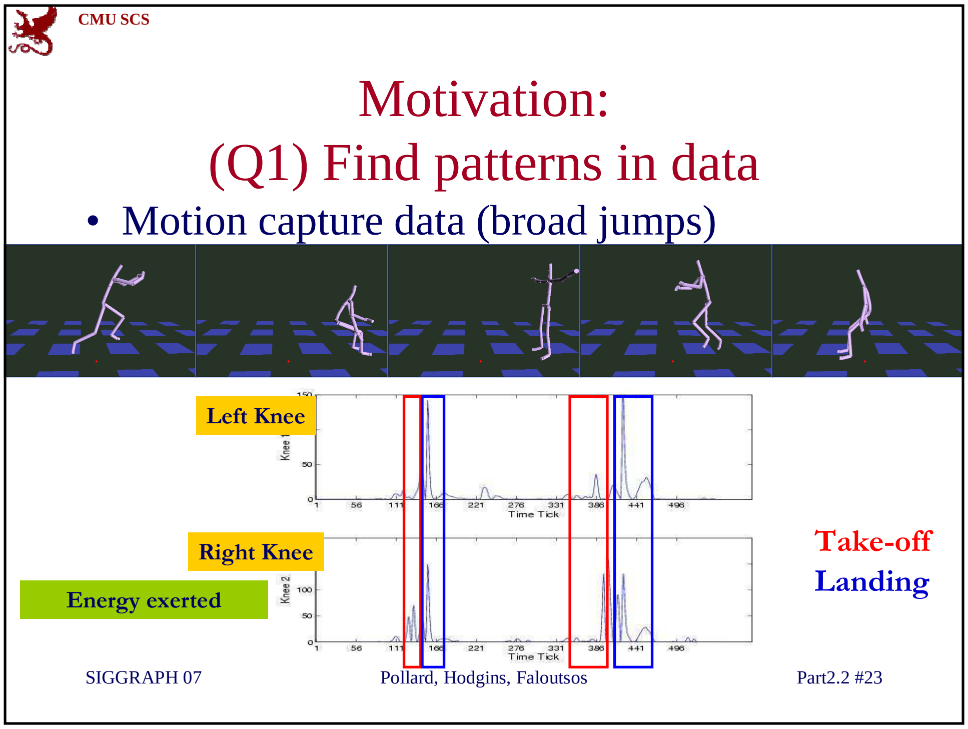
Outline

- Similarity search and indexing
- Feature extraction and dim. Reduction
  - FastMap
  - PCA/SVD
  - more on PCA/SPIRIT; ICA
  - DFT; Wavelets
- Linear Forecasting

Citation

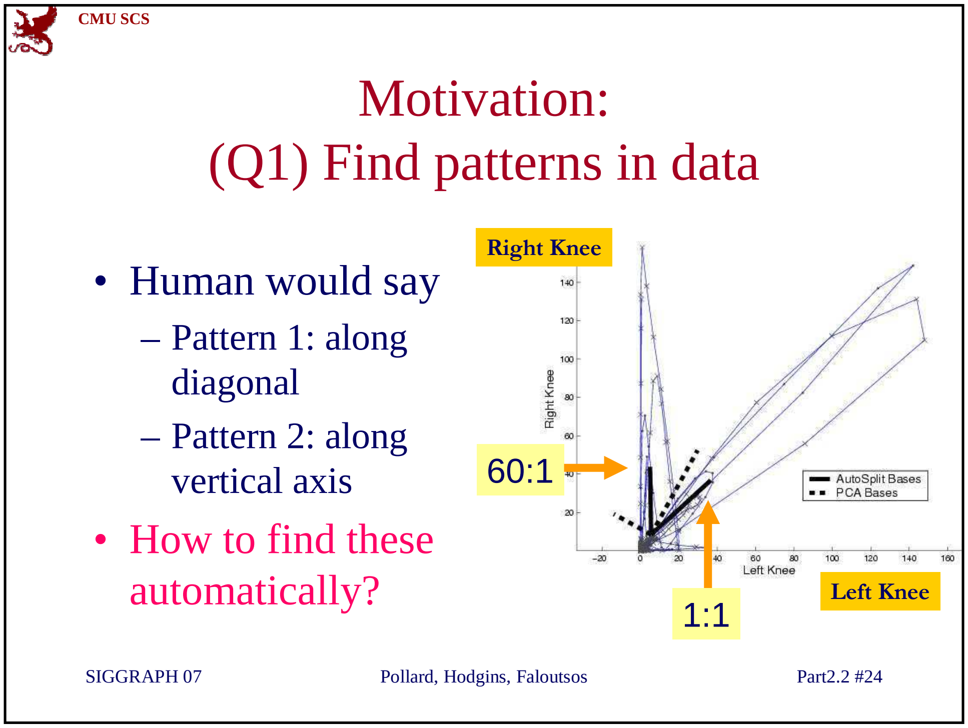
- *AutoSplit: Fast and Scalable Discovery of Hidden Variables in Stream and Multimedia Databases*, **Jia-Yu Pan**, Hiroyuki Kitagawa, Christos Faloutsos and Masafumi Hamamoto

PAKDD 2004, Sydney, Australia



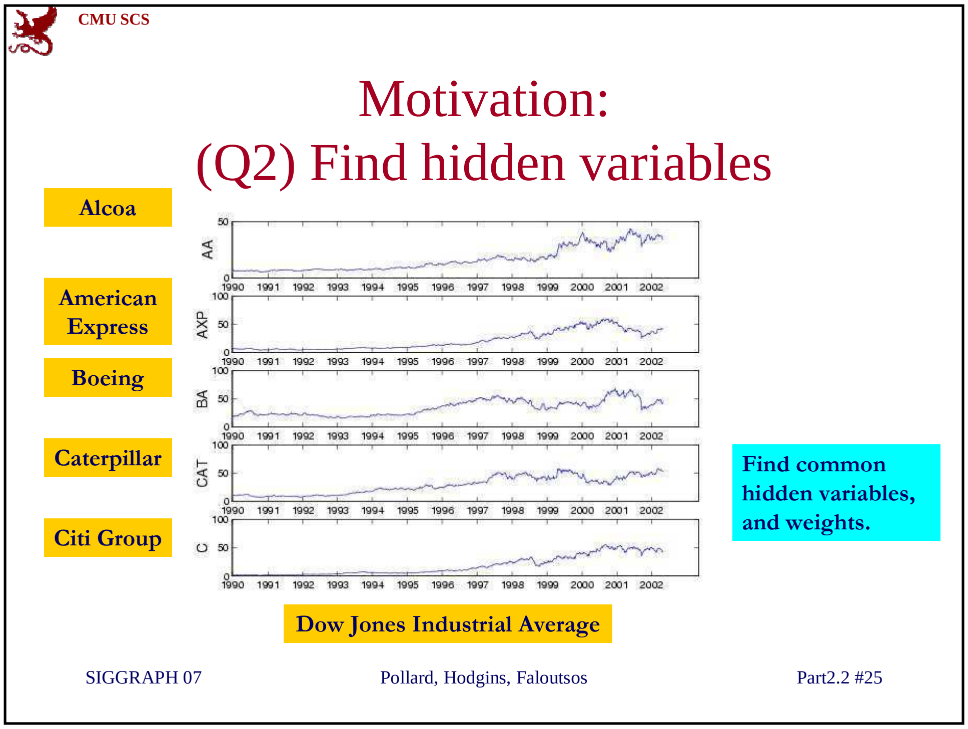
Motivation: (Q1) Find patterns in data

- Motion capture data (broad jumps)



Motivation: (Q1) Find patterns in data

- Human would say
  - – Pattern 1: along diagonal
  - – Pattern 2: along vertical axis
- How to find these automatically?



Motivation: (Q2) Find hidden variables

Alcoa

American Express

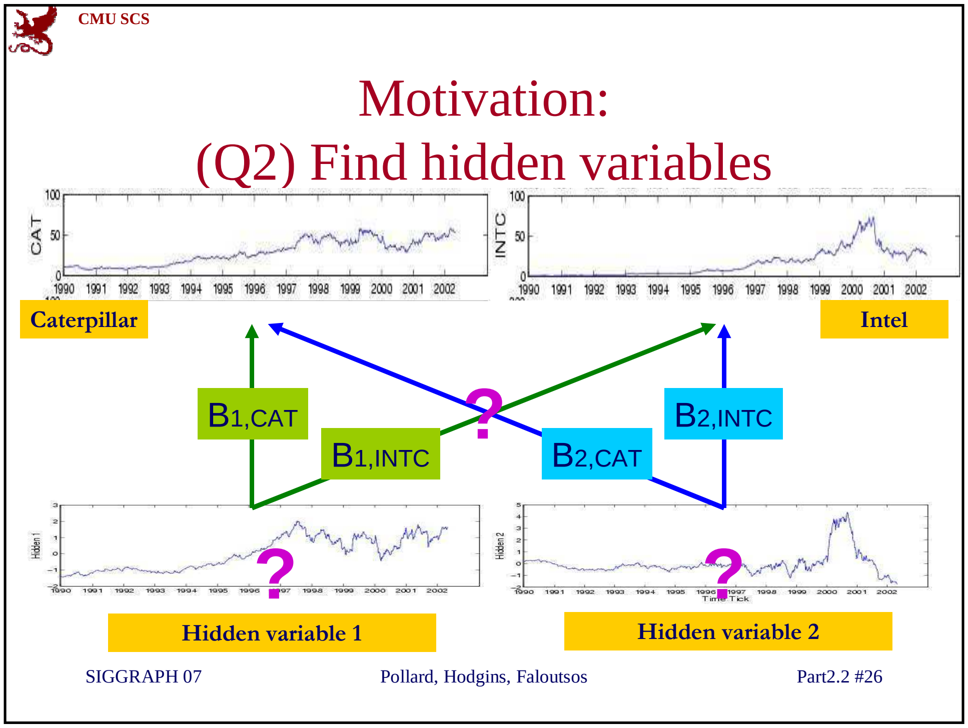
Boeing

Caterpillar

Citi Group

Find common hidden variables, and weights.

Dow Jones Industrial Average



Motivation: (Q2) Find hidden variables

Caterpillar

Intel

$B_{1,CAT}$

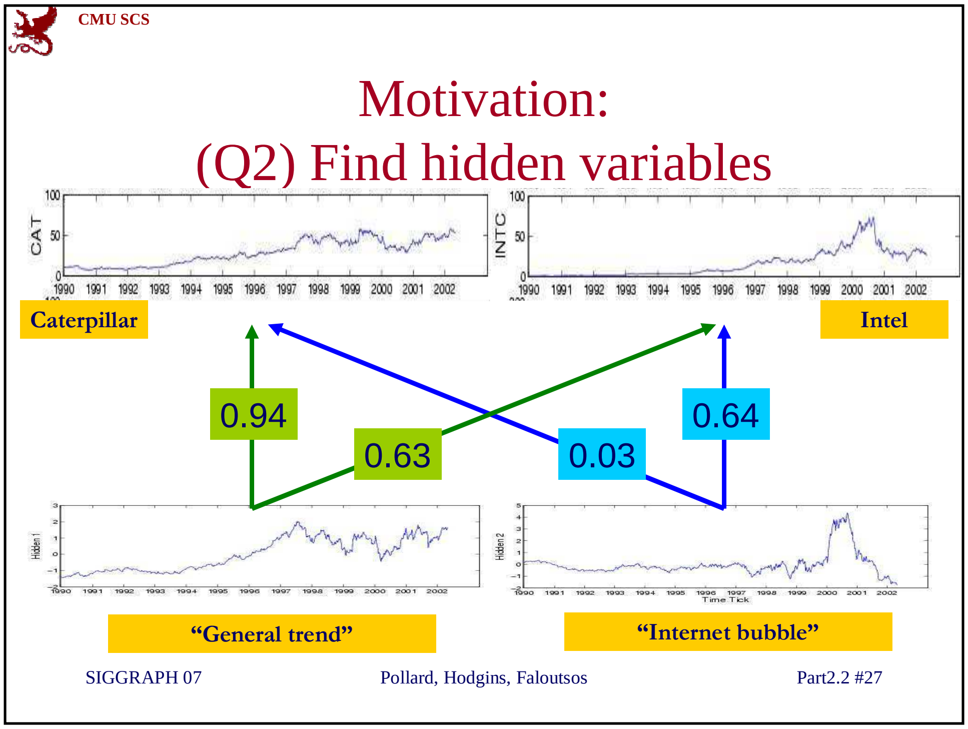
$B_{1,INTC}$

$B_{2,CAT}$

$B_{2,INTC}$

Hidden variable 1

Hidden variable 2



Motivation:
(Q2) Find hidden variables

Caterpillar

Intel

0.94

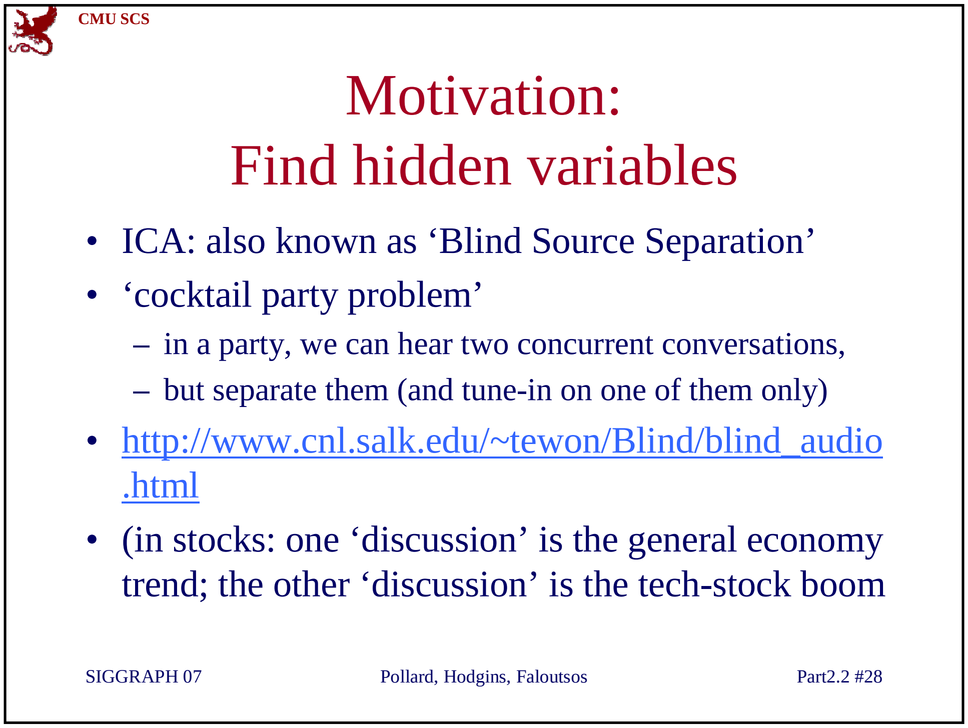
0.63

0.03

0.64

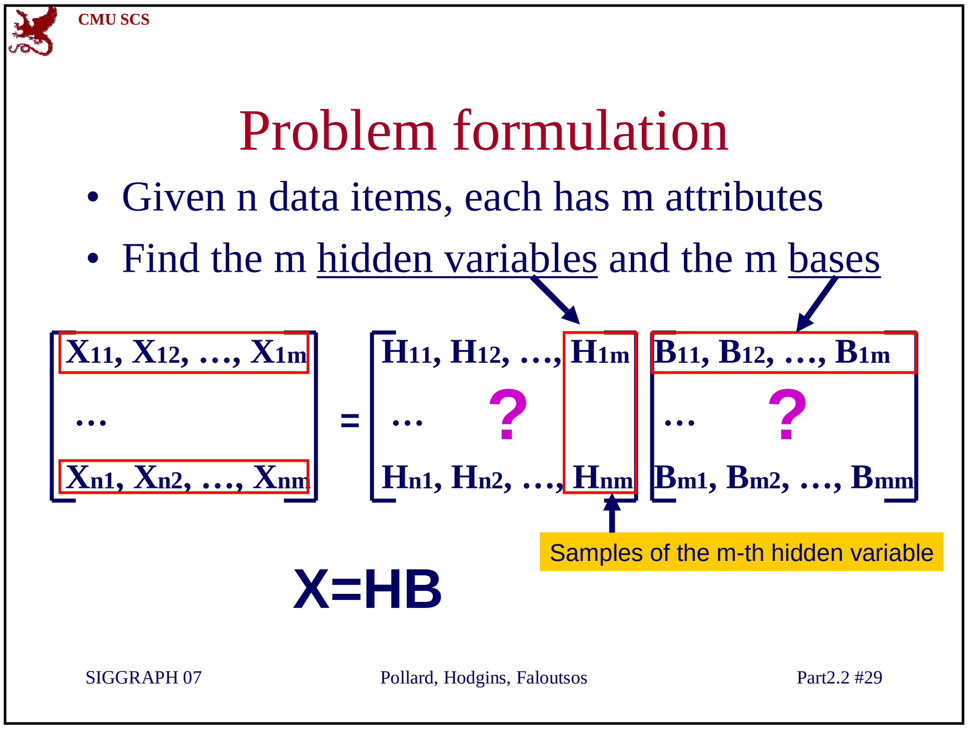
“General trend”

“Internet bubble”



Motivation:
Find hidden variables

- ICA: also known as ‘Blind Source Separation’
- ‘cocktail party problem’
  - in a party, we can hear two concurrent conversations,
  - but separate them (and tune-in on one of them only)
- http://www.cnl.salk.edu/~tewon/Blind/blind_audio.html
- (in stocks: one ‘discussion’ is the general economy trend; the other ‘discussion’ is the tech-stock boom



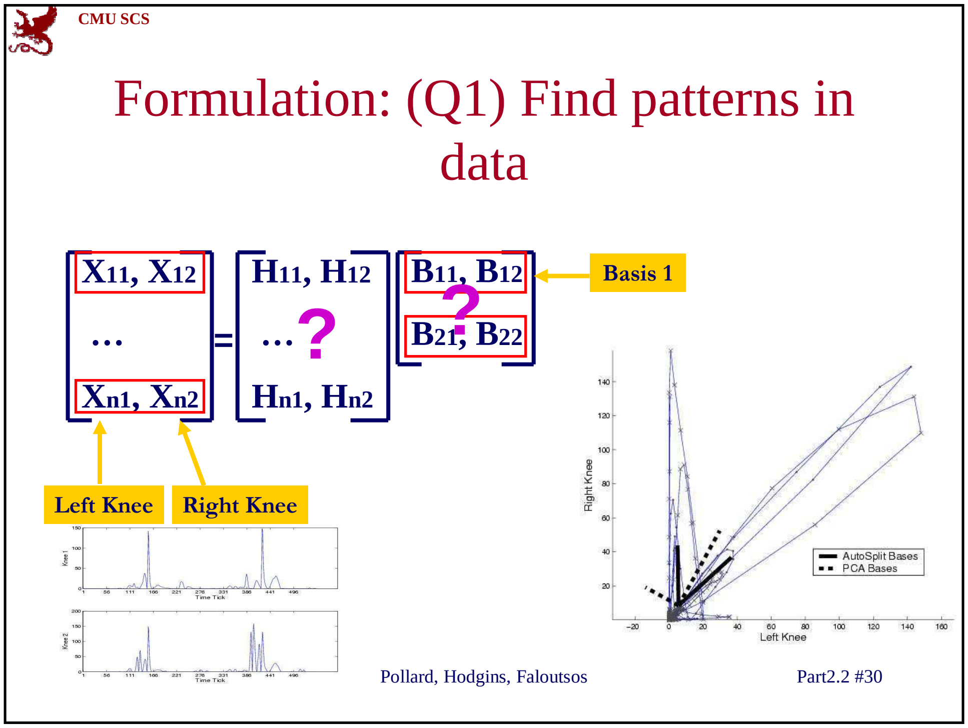
Problem formulation

- Given n data items, each has m attributes
- Find the m hidden variables and the m bases

$$\begin{bmatrix} X_{11}, X_{12}, \ldots, X_{1m} \\ \ldots \\ X_{n1}, X_{n2}, \ldots, X_{nm} \end{bmatrix} = \begin{bmatrix} H_{11}, H_{12}, \ldots, H_{1m} \\ \ldots \quad ? \\ H_{n1}, H_{n2}, \ldots, H_{nm} \end{bmatrix} \begin{bmatrix} B_{11}, B_{12}, \ldots, B_{1m} \\ \ldots \quad ? \\ B_{m1}, B_{m2}, \ldots, B_{mm} \end{bmatrix}$$

Samples of the m-th hidden variable

X=HB



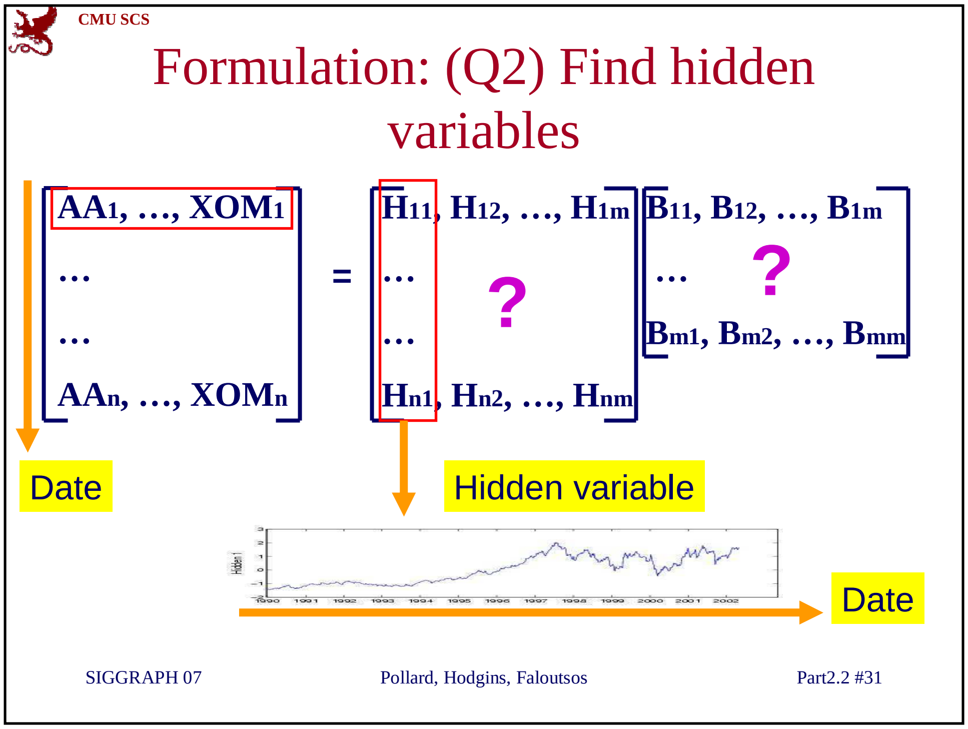
Formulation: (Q1) Find patterns in data

$$\begin{bmatrix} X_{11}, X_{12} \\ \ldots \\ X_{n1}, X_{n2} \end{bmatrix} = \begin{bmatrix} H_{11}, H_{12} \\ \ldots ? \\ H_{n1}, H_{n2} \end{bmatrix} \begin{bmatrix} B_{11}, B_{12} \\ ? \\ B_{21}, B_{22} \end{bmatrix}$$

Basis 1

Left Knee

Right Knee



CMU SCS

Formulation: (Q2) Find hidden variables

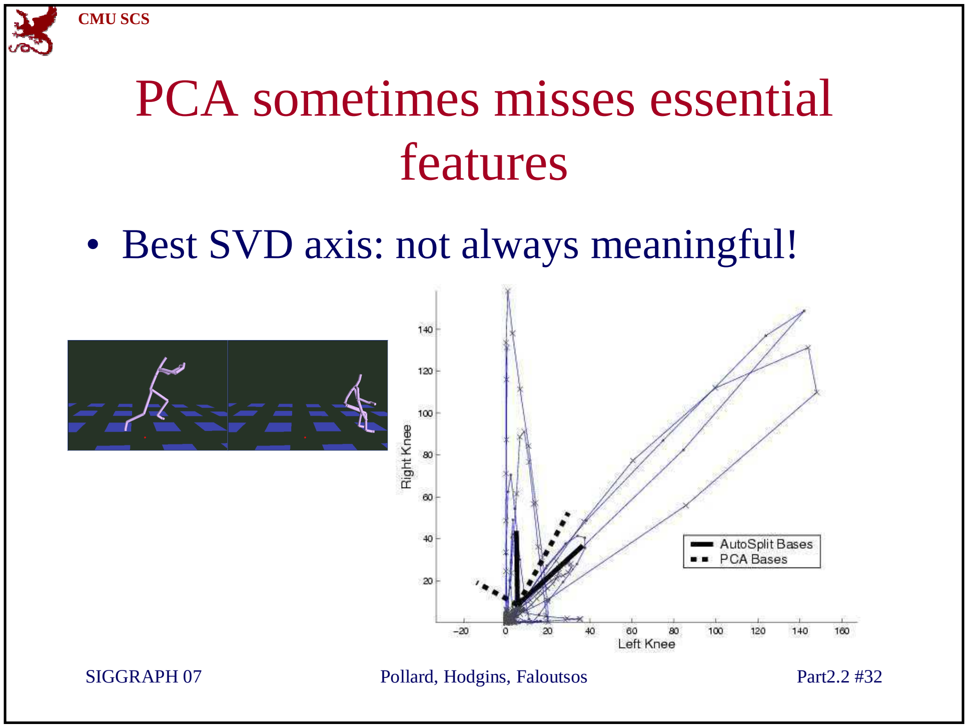
$$\begin{bmatrix} AA_1, \ldots, XOM_1 \\ \ldots \\ \ldots \\ AA_n, \ldots, XOM_n \end{bmatrix} = \begin{bmatrix} H_{11}, H_{12}, \ldots, H_{1m} \\ \ldots \quad ? \\ \ldots \\ H_{n1}, H_{n2}, \ldots, H_{nm} \end{bmatrix} \begin{bmatrix} B_{11}, B_{12}, \ldots, B_{1m} \\ \ldots \quad ? \\ B_{m1}, B_{m2}, \ldots, B_{mm} \end{bmatrix}$$

Date

Hidden variable

Date

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #31



CMU SCS

PCA sometimes misses essential features

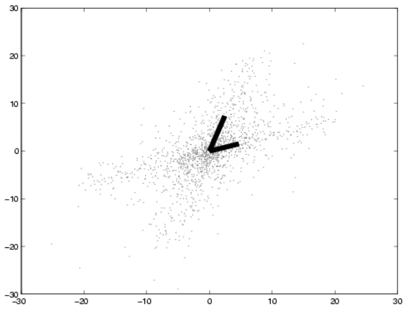
- Best SVD axis: not always meaningful!

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #32

CMU SCS

Batch-AutoSplit at work

‘Sparse coding’:
most points have ~zero
for one or both coefficients



Converge

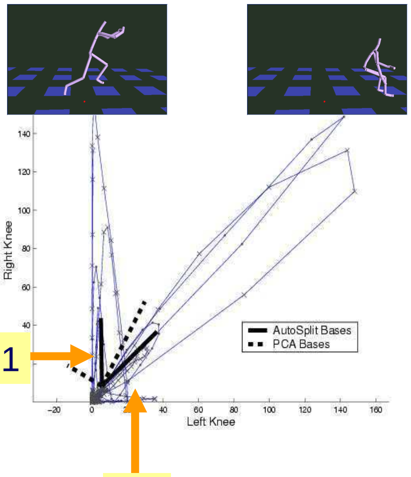
SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #33

CMU SCS

Q1: Find patterns

Take-off Landing

- Patterns found
  - Landing: both knees
  - Take-off: right knee
  - Right-handed actor



60:1

1:1

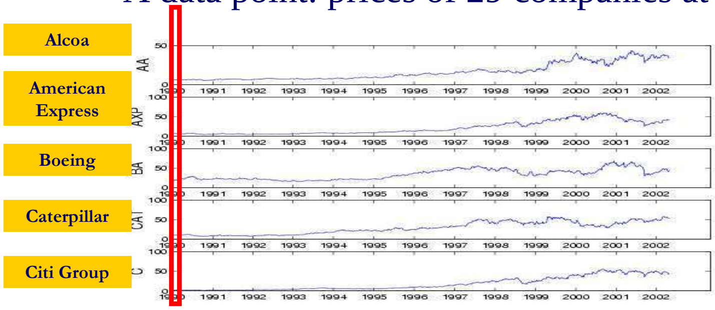
m=2, n=550

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #34

CMU SCS

Q2: Find hidden variables (DJIA stocks)

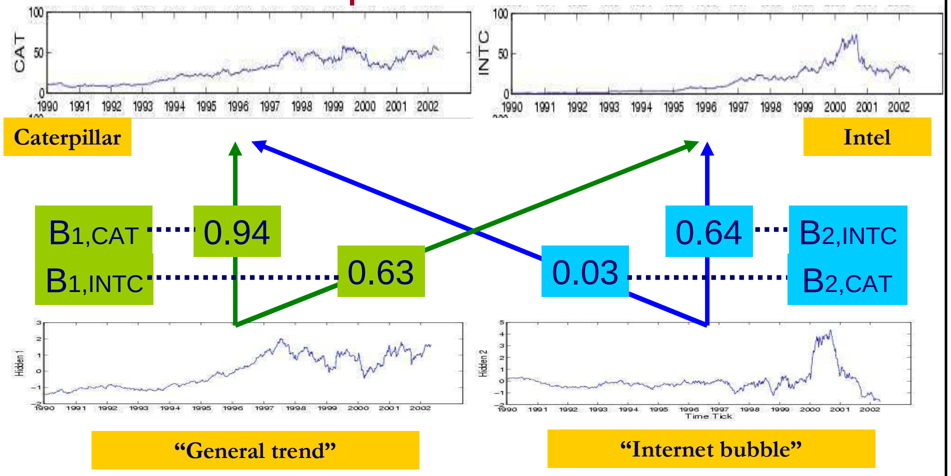
- Weekly DJIA closing prices
  - 01/02/1990-08/05/2002, n=660 data points
  - A data point: prices of 29 companies at the time



SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #35

CMU SCS

(Q2) Characterize hidden variable by the companies it influences



Caterpillar

Intel

$B_{1,CAT}$ 0.94

$B_{1,INTC}$ 0.63

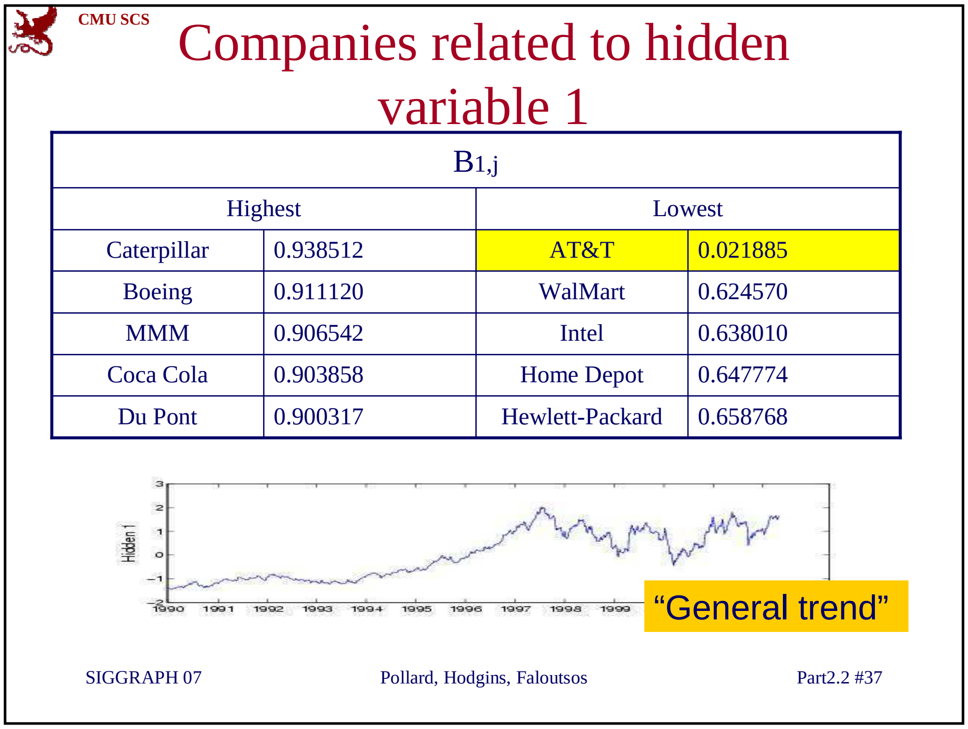
0.64 $B_{2,INTC}$

0.03 $B_{2,CAT}$

“General trend”

“Internet bubble”

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #36

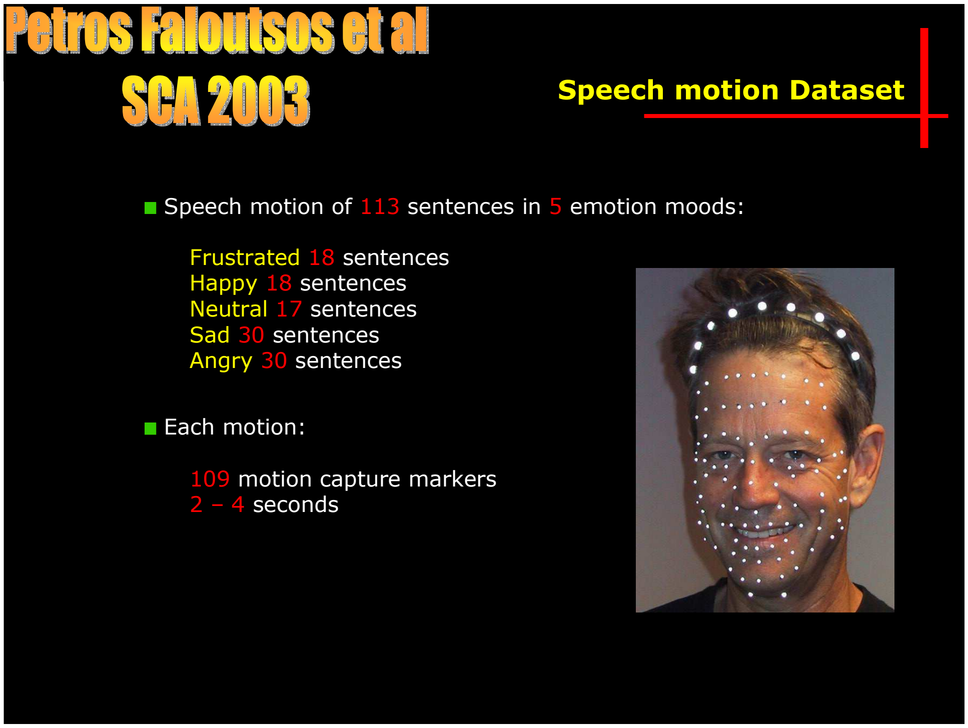

CMU SCS

Companies related to hidden variable 1

B1,j			
Highest		Lowest	
Caterpillar	0.938512	AT&T	0.021885
Boeing	0.911120	WalMart	0.624570
MMM	0.906542	Intel	0.638010
Coca Cola	0.903858	Home Depot	0.647774
Du Pont	0.900317	Hewlett-Packard	0.658768

“General trend”

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #37


Petros Faloutsos et al SCA 2003

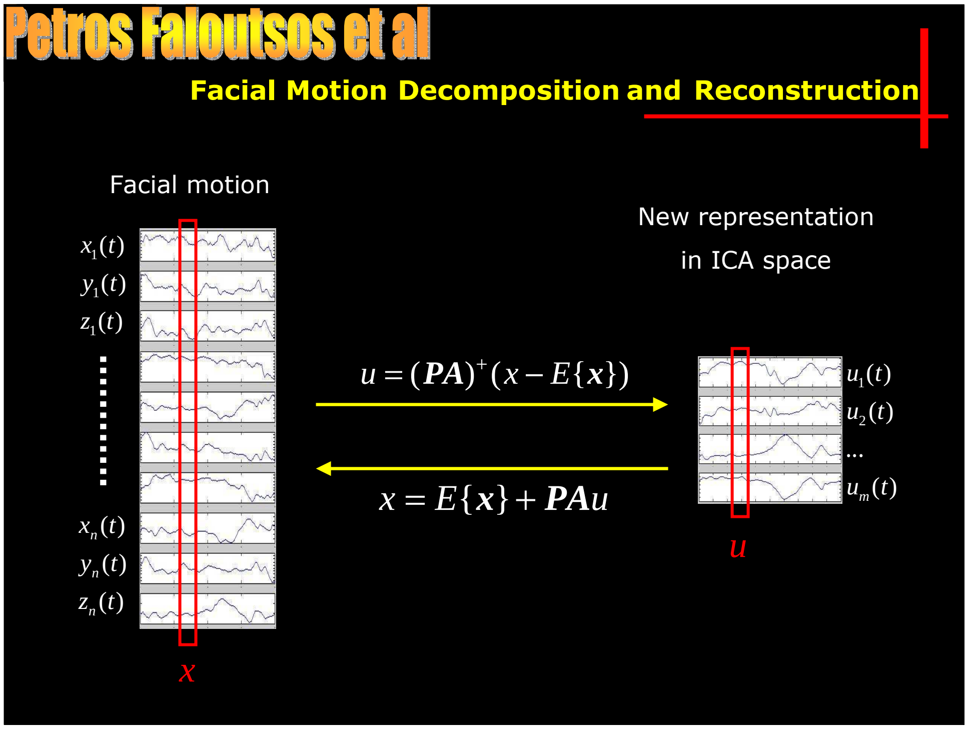
Speech motion Dataset

- Speech motion of 113 sentences in 5 emotion moods:

  Frustrated 18 sentences
  Happy 18 sentences
  Neutral 17 sentences
  Sad 30 sentences
  Angry 30 sentences

- Each motion:

  109 motion capture markers
  2 – 4 seconds



Petros Faloutsos et al

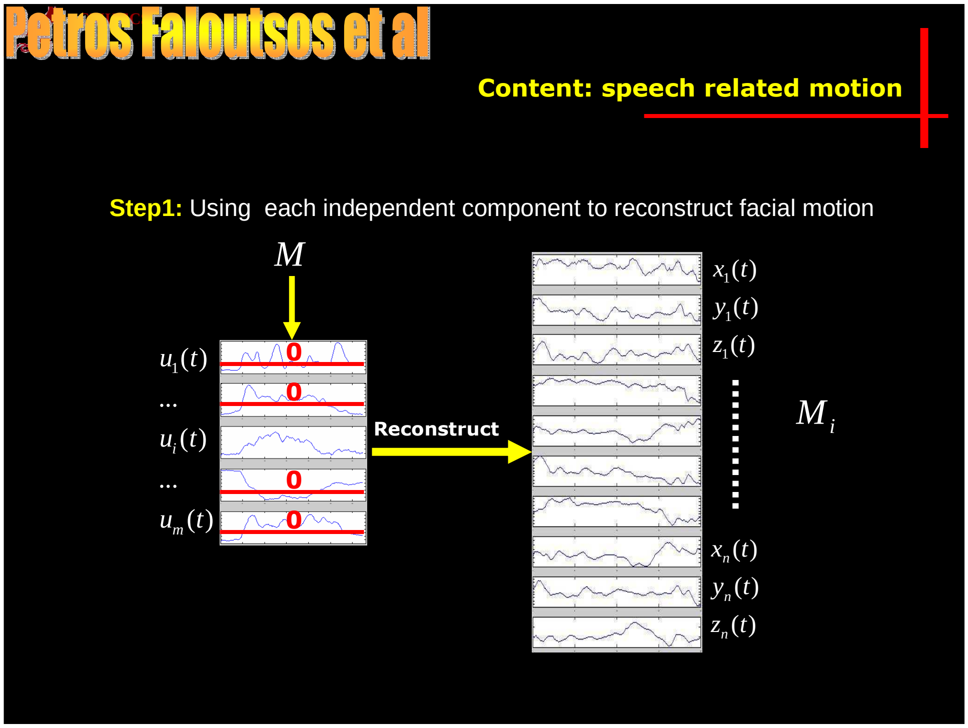
Facial Motion Decomposition and Reconstruction

Facial motion

New representation in ICA space

$$u = (\boldsymbol{PA})^{+}(x - E\{\boldsymbol{x}\})$$

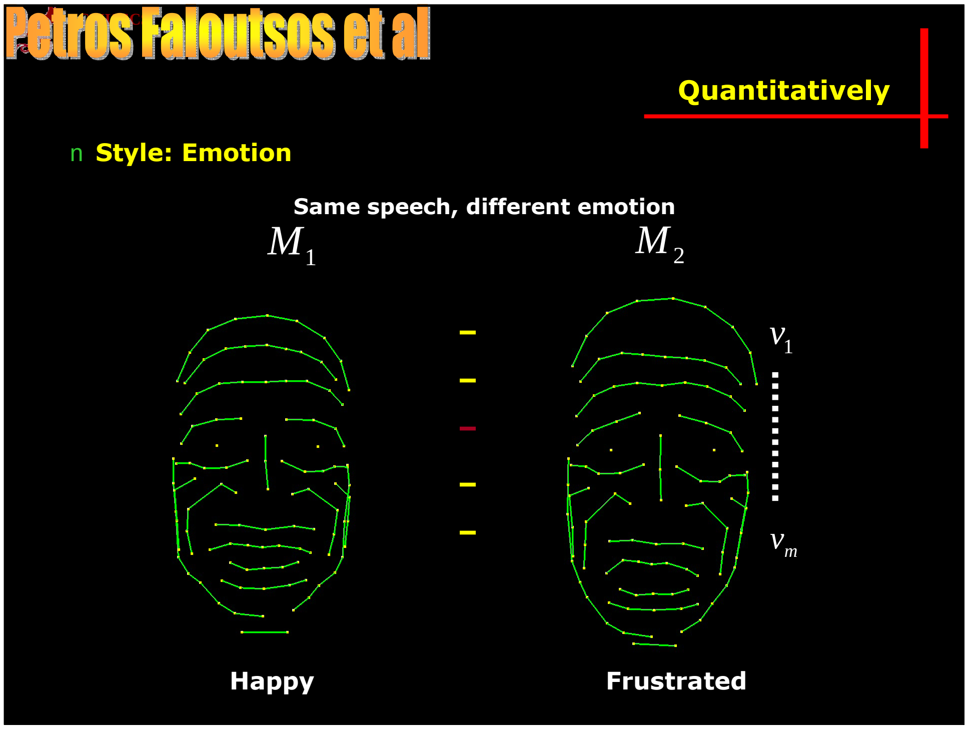
$$x = E\{\boldsymbol{x}\} + \boldsymbol{PA}u$$



Petros Faloutsos et al

Content: speech related motion

Step1: Using each independent component to reconstruct facial motion



Petros Faloutsos et al

Quantitatively

- Style: Emotion

Same speech, different emotion

M_1 − M_2

v_1 ⋮ v_m

Happy

Frustrated

Running & Walking (Copy & Replace)

Walk style + Run = Jog?

PETROS FALOUTSOS ET AL, 2006

References

- Yong Cao, Petros Faloutsos, Frederic Pighin, *Unsupervised Learning for Speech Motion Editing*, ACM SIGGRAPH / Eurographics Symposium on Computer Animation, 225-231 (2003)

References

- A. Hyvärinen, J. Karhunen, E. Oja, *Independent Component Analysis*, John Wiley & Sons, 2001 (excellent book on ICA; also check the FastICA software at http://www.cis.hut.fi/projects/ica/fastica/

References

- Jia-Yu Pan, Hiroyuki Kitagawa, Christos Faloutsos and Masafumi Hamamoto *AutoSplit: Fast and Scalable Discovery of Hidden Variables in Stream and Multimedia Databases*, PAKDD 2004, Sydney, Australia

References

- A. Shapiro, Y. Cao, P. Faloutsos, *Interactive Motion Decomposition*, ACM SIGGRAPH 2004 Technical Sketches, Los Angeles, CA, August 2004.
- Ari Shapiro, Yong Cao, Petros Faloutsos, *Style Components*, In Proceedings of Graphics Interface 2006, pp. 33-40.

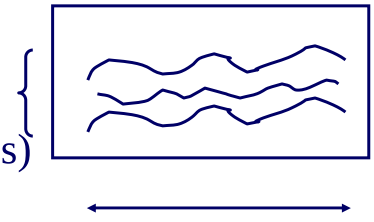
Outline

- Similarity search and indexing
- Feature extraction and dim. Reduction
  - FastMap
  - PCA/SVD
  - more on PCA/SPIRIT; ICA
  - DFT; Wavelets
- Linear Forecasting

Problem #2: Dim. reduction

- Can we describe it with fewer numbers?

73 angles (or positions)



Eg., 2,000 time-ticks

What does DFT do?

A: highlights the periodicities

Skip

DFT: definition

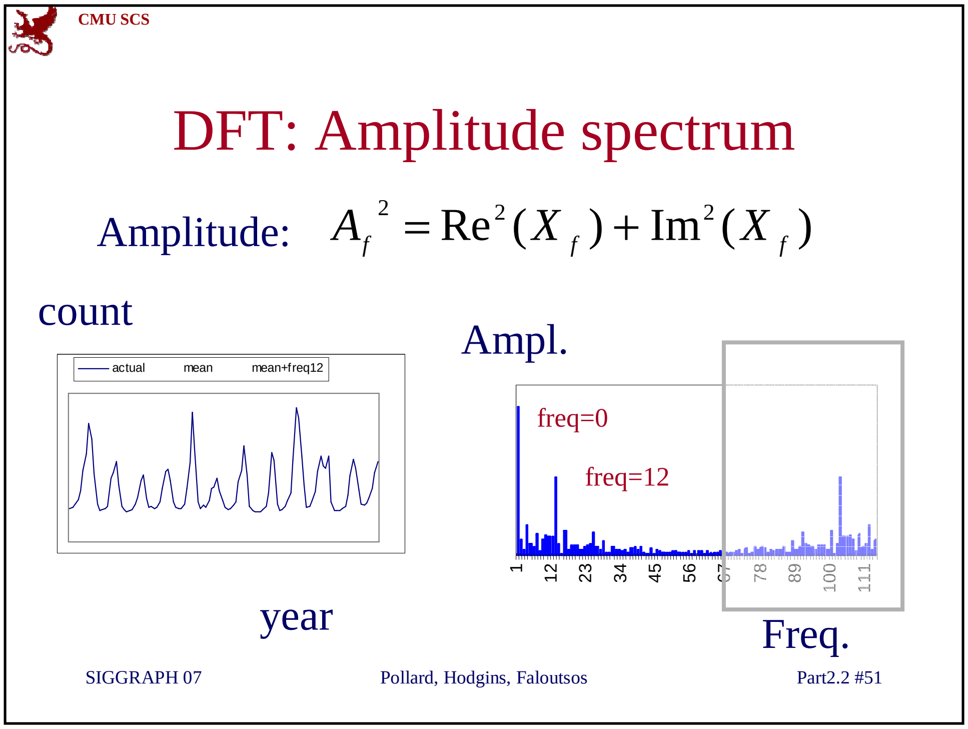
- For a sequence $x_0, x_1, \ldots x_{n-1}$
- the (**n-point**) Discrete Fourier Transform is
- $X_0, X_1, \ldots X_{n-1}$:

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j 2\pi \, tf / n) \qquad f = 0, \ldots, n-1$$

$$(j = \sqrt{-1})$$

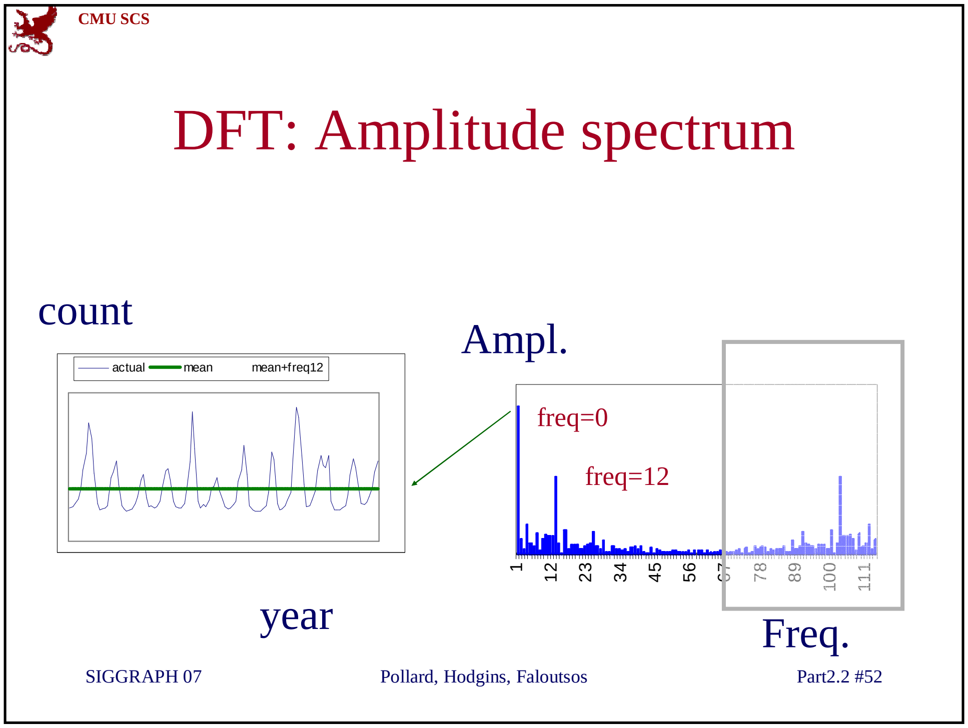
inverse DFT

$$x_t = 1/\sqrt{n} \sum_{t=0}^{n-1} X_f * \exp(+j 2\pi \, tf / n)$$

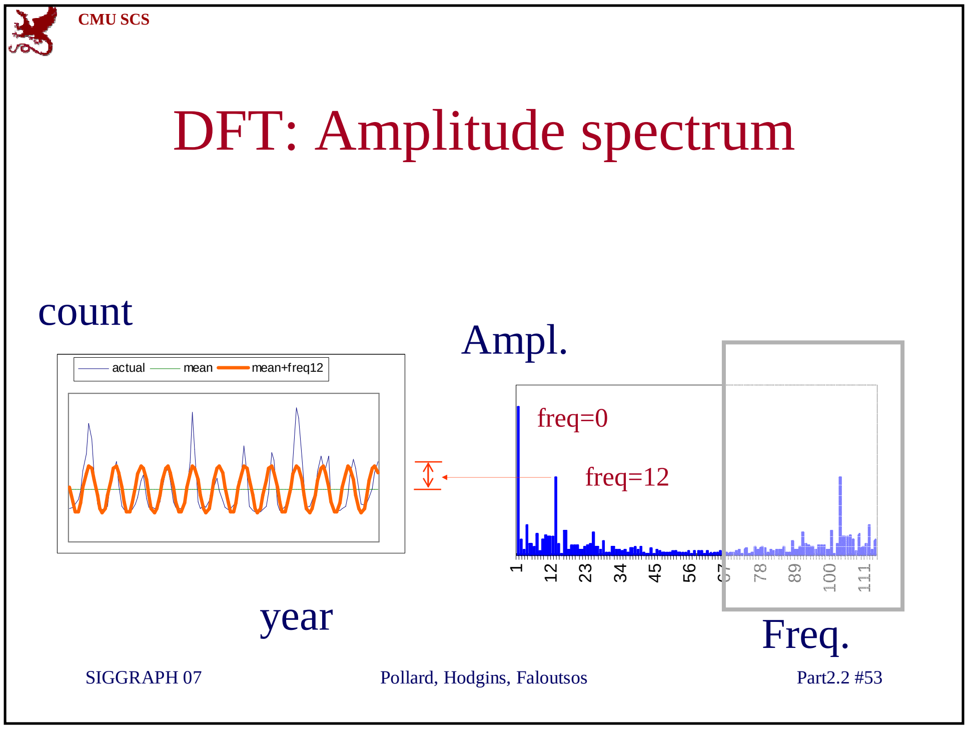


DFT: Amplitude spectrum

Amplitude: $A_f^2 = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$



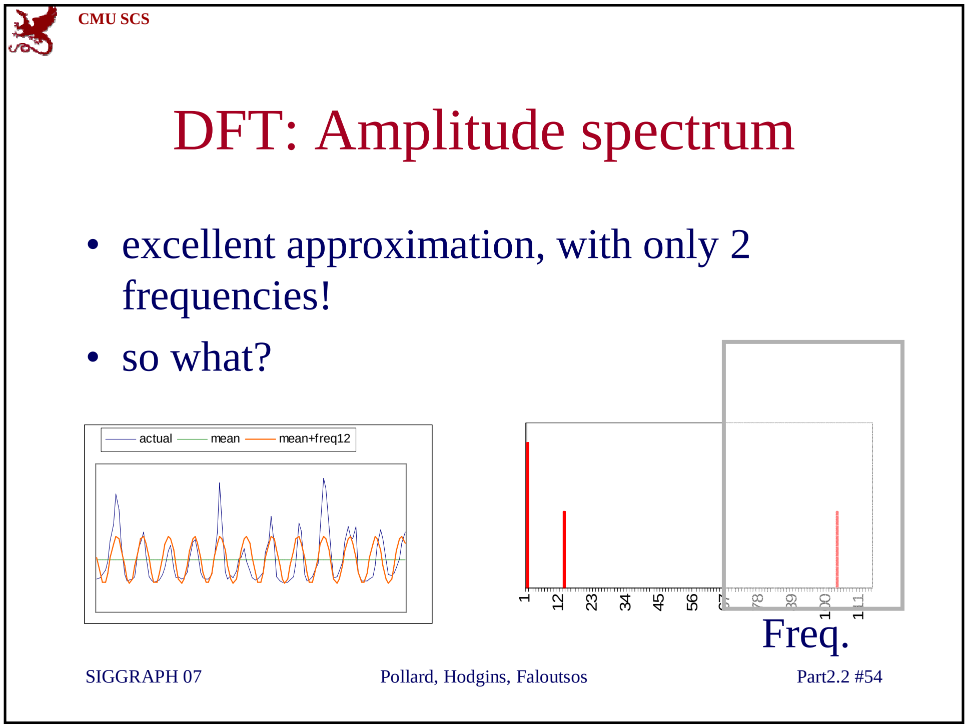
DFT: Amplitude spectrum



CMU SCS

DFT: Amplitude spectrum

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #53



CMU SCS

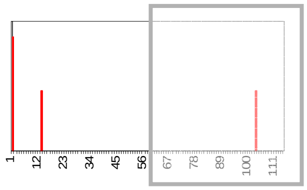
DFT: Amplitude spectrum

- excellent approximation, with only 2 frequencies!
- so what?

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #54

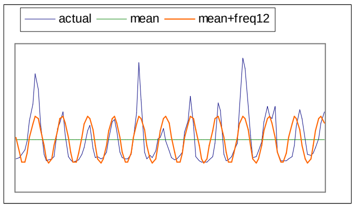
DFT: Amplitude spectrum

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



DFT: Amplitude spectrum

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

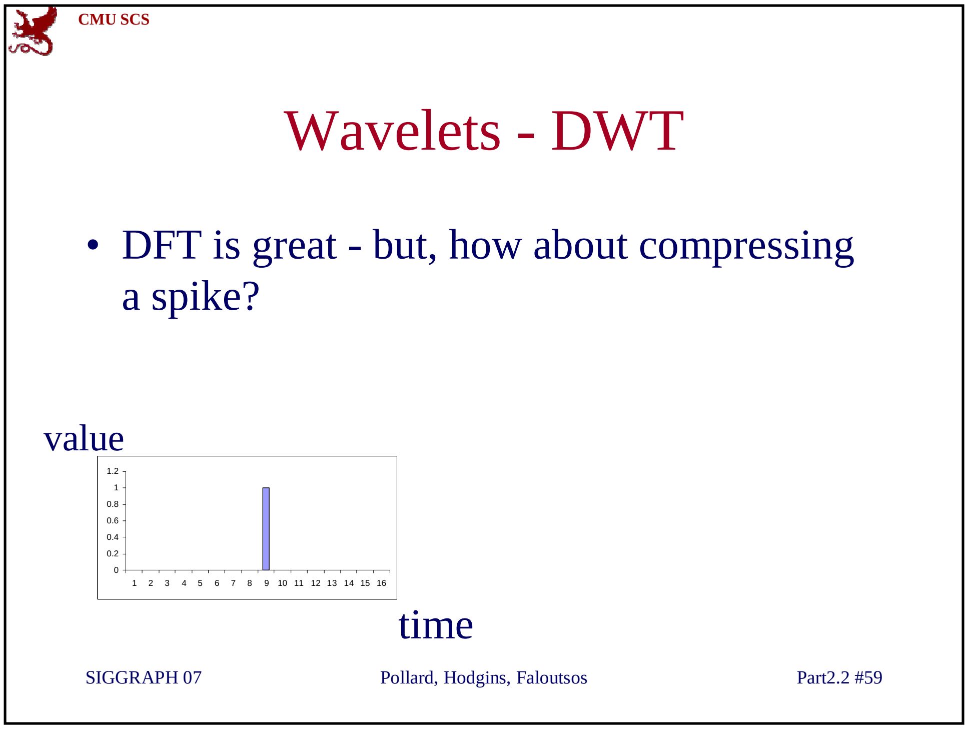


DFT - Conclusions

- 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)

Outline

- Similarity search and indexing
- Feature extraction and dim. Reduction
  - FastMap
  - PCA/SVD
  - more on PCA/SPIRIT; ICA
  - DFT; **Wavelets**
- Linear Forecasting

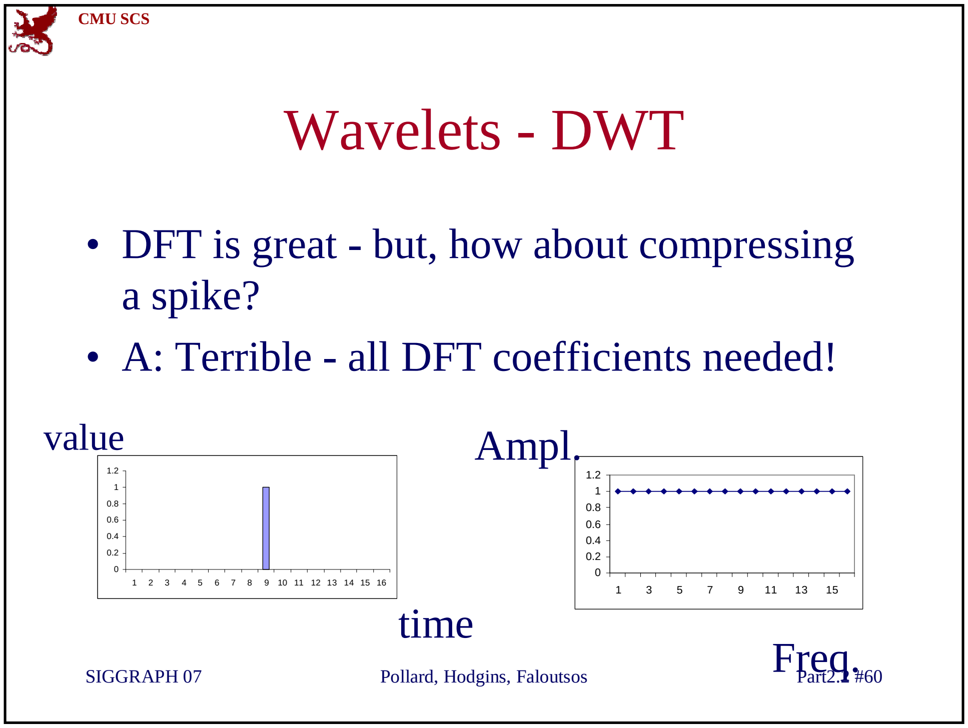


CMU SCS

Wavelets - DWT

- DFT is great - but, how about compressing a spike?

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #59



CMU SCS

Wavelets - DWT

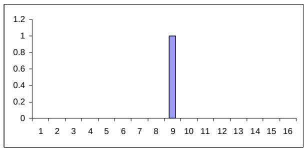
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

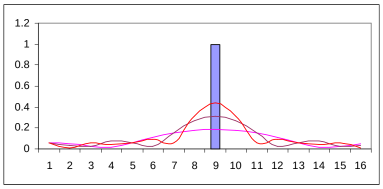
SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #60

Wavelets - DWT

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

value



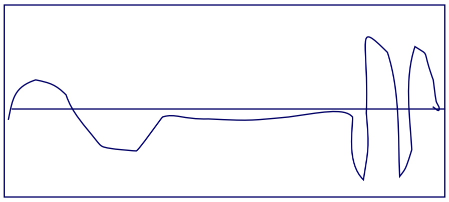


time

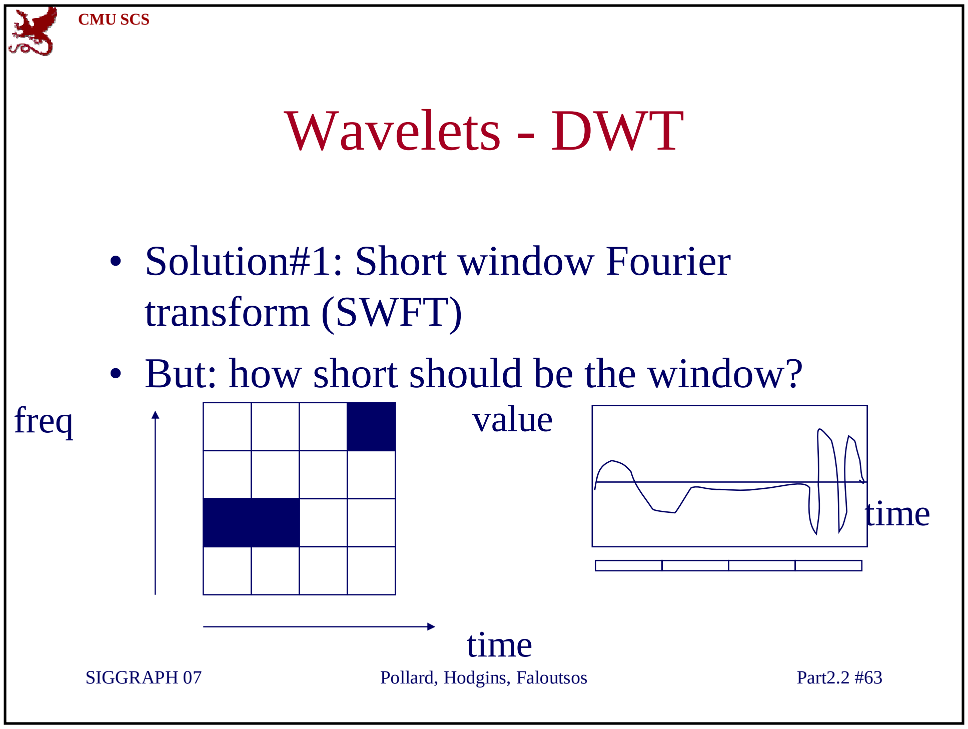
Wavelets - DWT

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

value



time



Wavelets - DWT

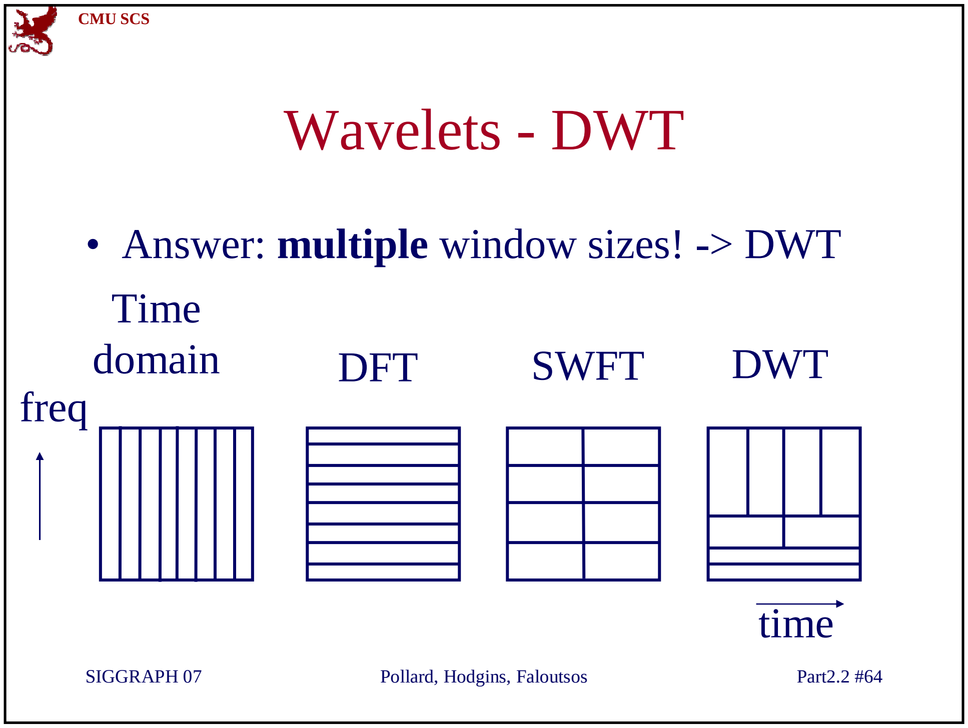
- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?

freq

time

value

time



Wavelets - DWT

- Answer: **multiple** window sizes! -> DWT

Time domain

DFT

SWFT

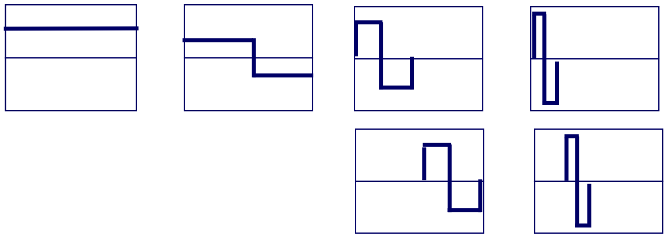
DWT

freq

time

Haar Wavelets

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



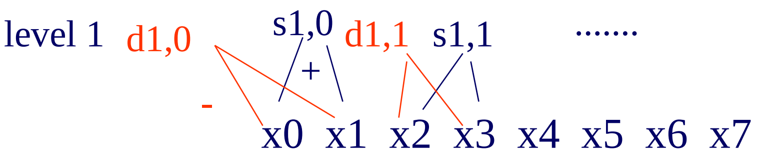
Skip

Wavelets - construction

x0 x1 x2 x3 x4 x5 x6 x7

Skip

Wavelets - construction



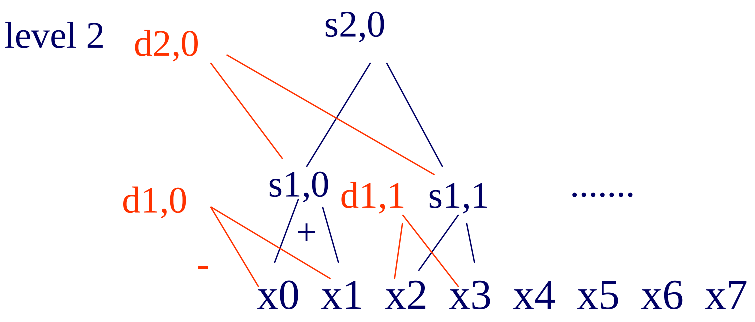
level 1 d1,0 s1,0 d1,1 s1,1 ....

- +

x0 x1 x2 x3 x4 x5 x6 x7

Skip

Wavelets - construction



level 2 d2,0 s2,0

d1,0 s1,0 d1,1 s1,1 ....

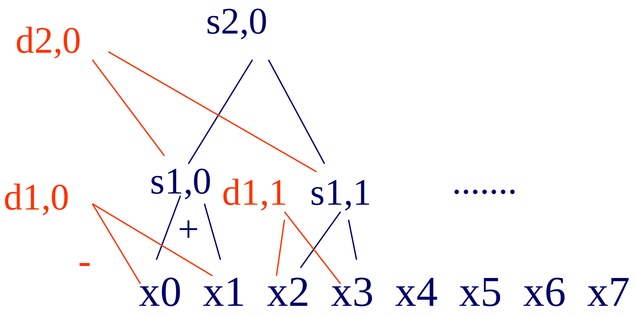
- +

x0 x1 x2 x3 x4 x5 x6 x7

Skip

Wavelets - construction

etc ...

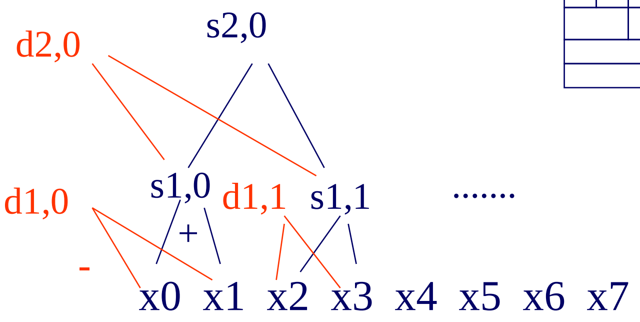


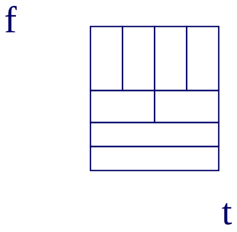
Skip

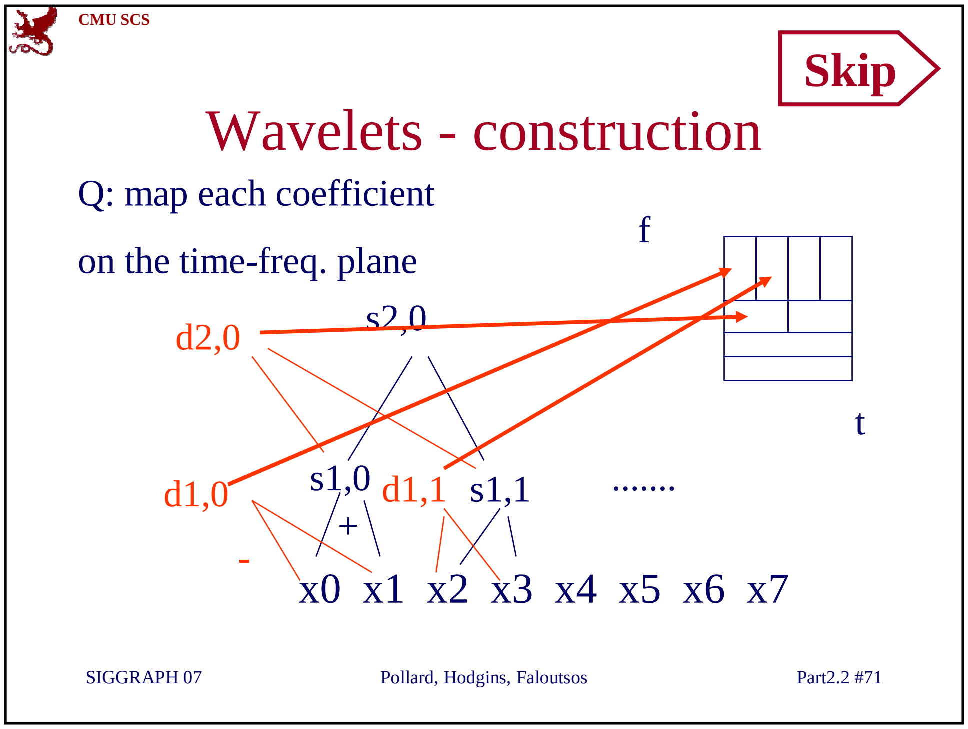
Wavelets - construction

Q: map each coefficient

on the time-freq. plane







CMU SCS

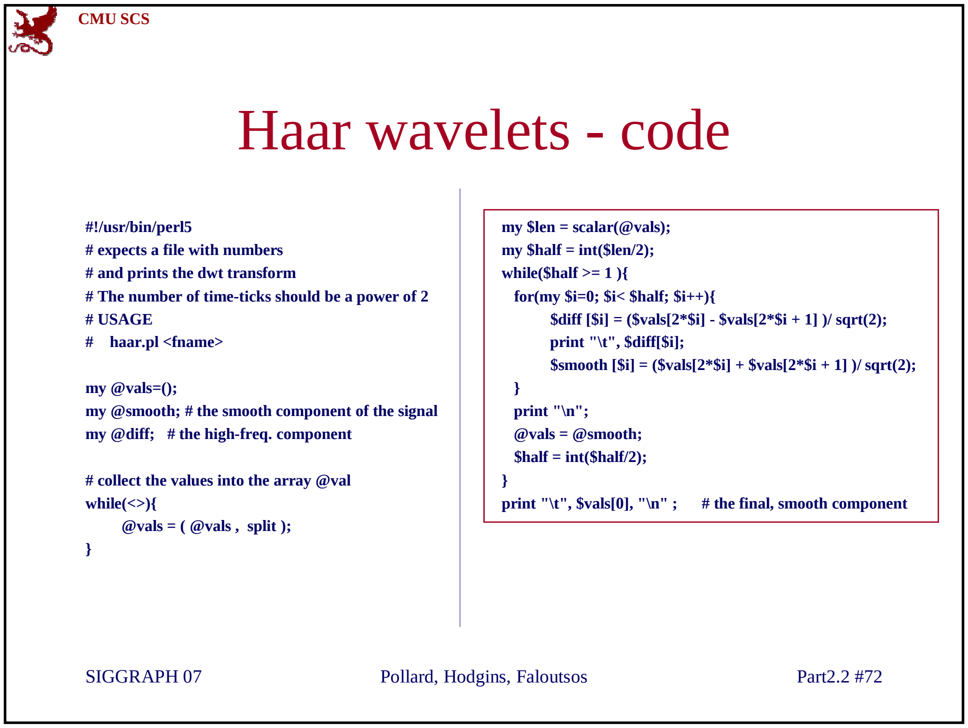
Skip

Wavelets - construction

Q: map each coefficient

on the time-freq. plane

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #71



CMU SCS

Haar wavelets - code

```
#!/usr/bin/perl5
# expects a file with numbers
# and prints the dwt transform
# The number of time-ticks should be a power of 2
# USAGE
#    haar.pl <fname>

my @vals=();
my @smooth; # the smooth component of the signal
my @diff;   # the high-freq. component

# collect the values into the array @val
while(<>){
      @vals = ( @vals ,  split );
}
```

```
my $len = scalar(@vals);
my $half = int($len/2);
while($half >= 1 ){
  for(my $i=0; $i< $half; $i++){
        $diff [$i] = ($vals[2*$i] - $vals[2*$i + 1] )/ sqrt(2);
        print "\t", $diff[$i];
        $smooth [$i] = ($vals[2*$i] + $vals[2*$i + 1] )/ sqrt(2);
  }
  print "\n";
  @vals = @smooth;
  $half = int($half/2);
}
print "\t", $vals[0], "\n" ;      # the final, smooth component
```

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.2 #72

Wavelets - construction

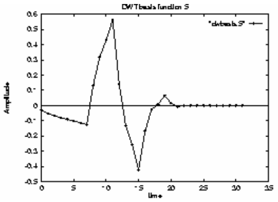
Observation1:

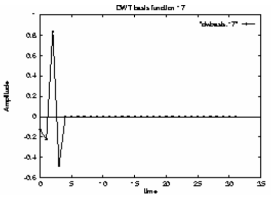
- '+' can be some weighted addition
- '-' is the corresponding weighted difference ('Quadrature mirror filters')

Observation2: unlike DFT/DCT,

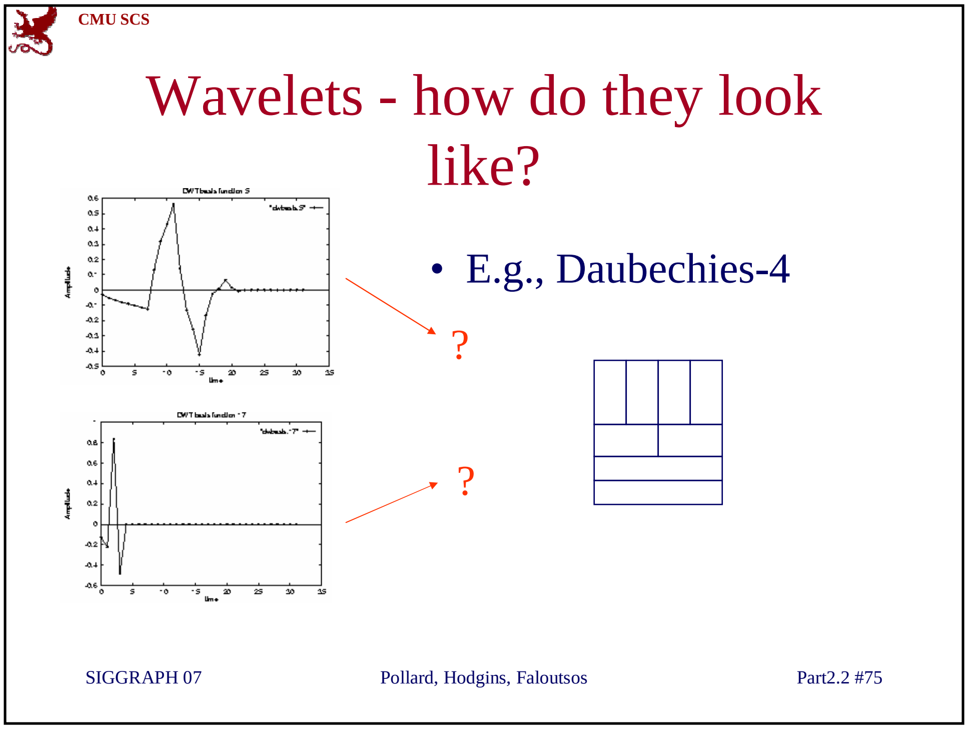
- there are *many* wavelet bases: Haar, Daubechies-4, Daubechies-6, Coifman, Morlet, Gabor, ...

Wavelets - how do they look like?



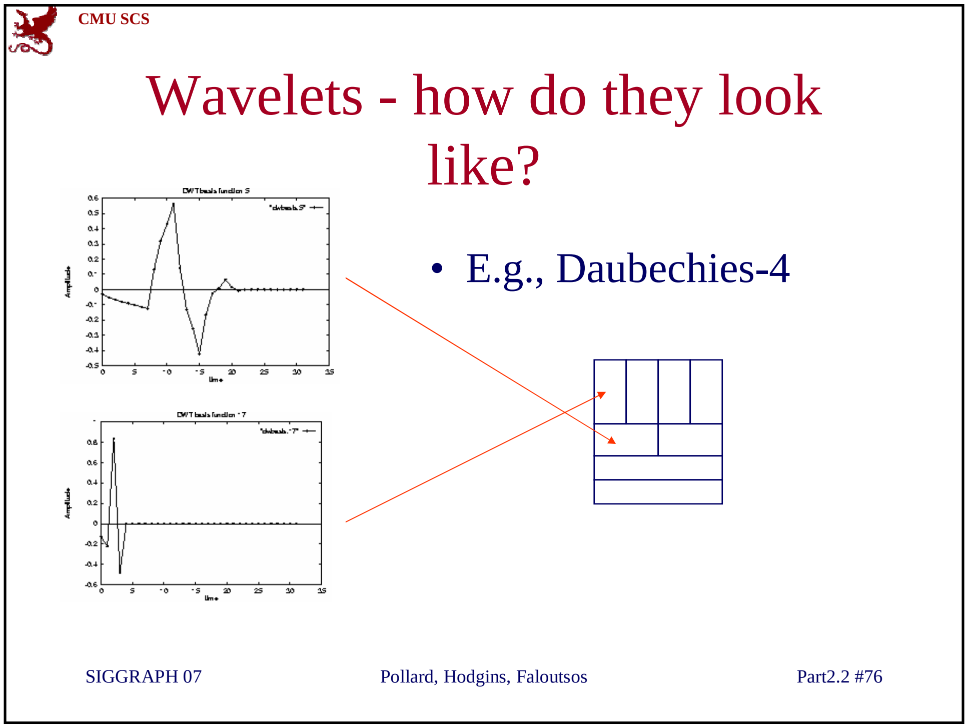


- E.g., Daubechies-4



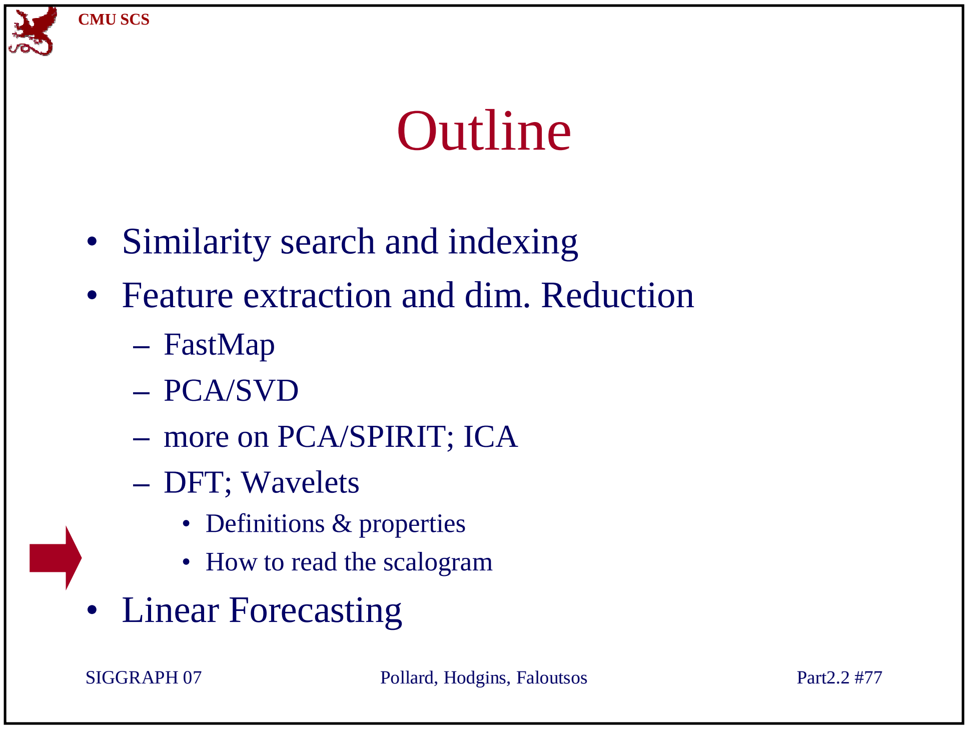
Wavelets - how do they look like?

- E.g., Daubechies-4



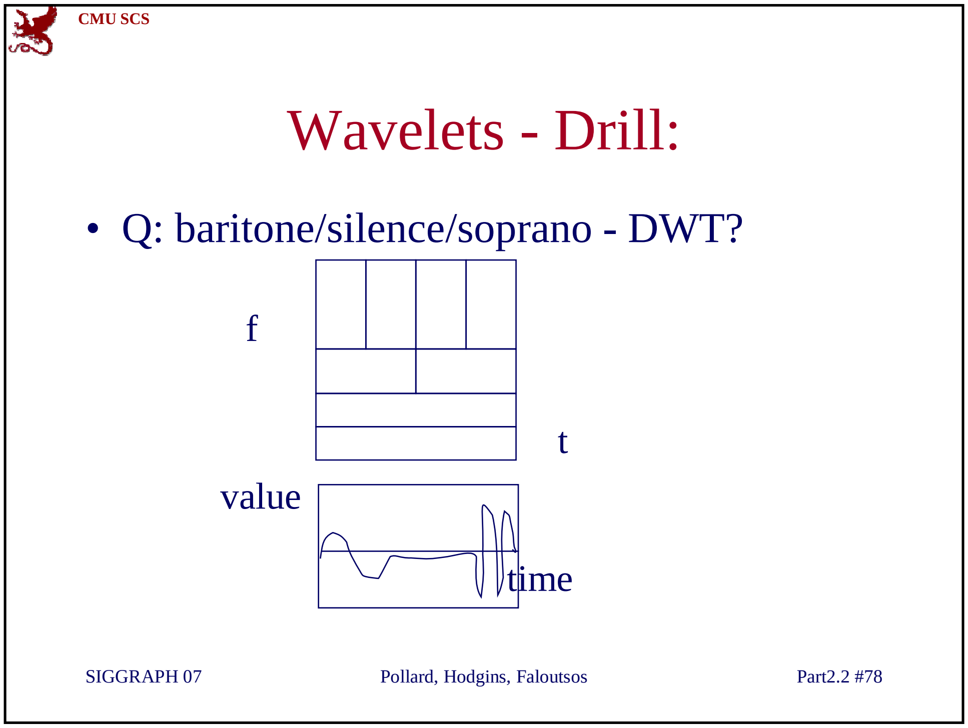
Wavelets - how do they look like?

- E.g., Daubechies-4



Outline

- Similarity search and indexing
- Feature extraction and dim. Reduction
  - FastMap
  - PCA/SVD
  - more on PCA/SPIRIT; ICA
  - DFT; Wavelets
    - Definitions & properties
    - How to read the scalogram
- Linear Forecasting

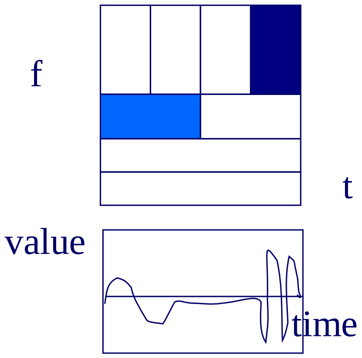


Wavelets - Drill:

- Q: baritone/silence/soprano - DWT?

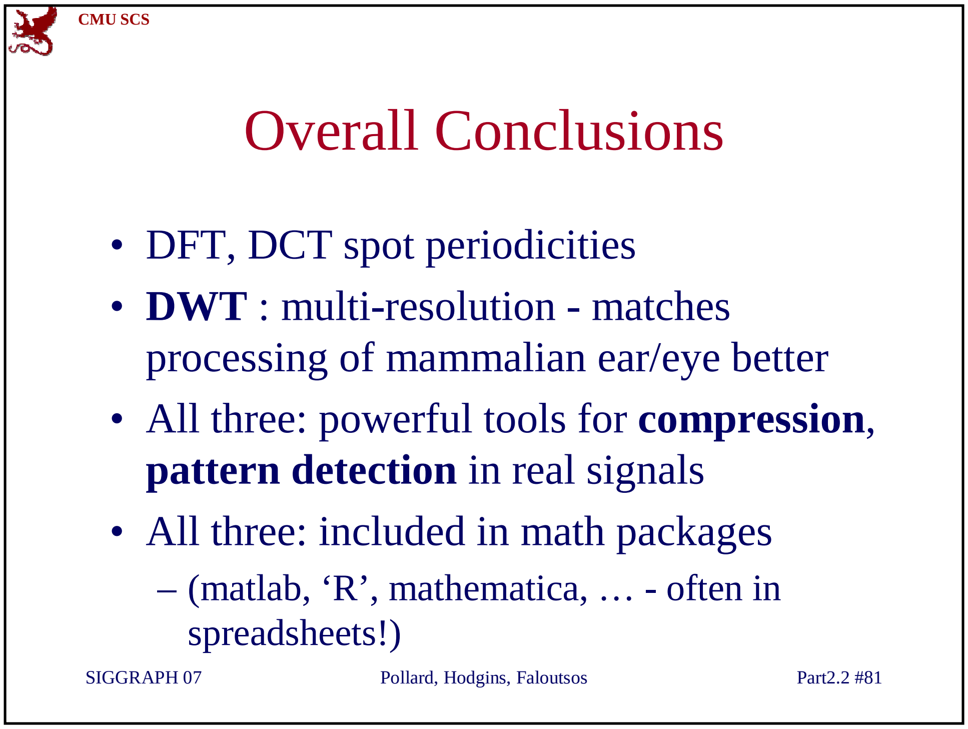
Wavelets - Drill:

- Q: baritone/soprano - DWT?



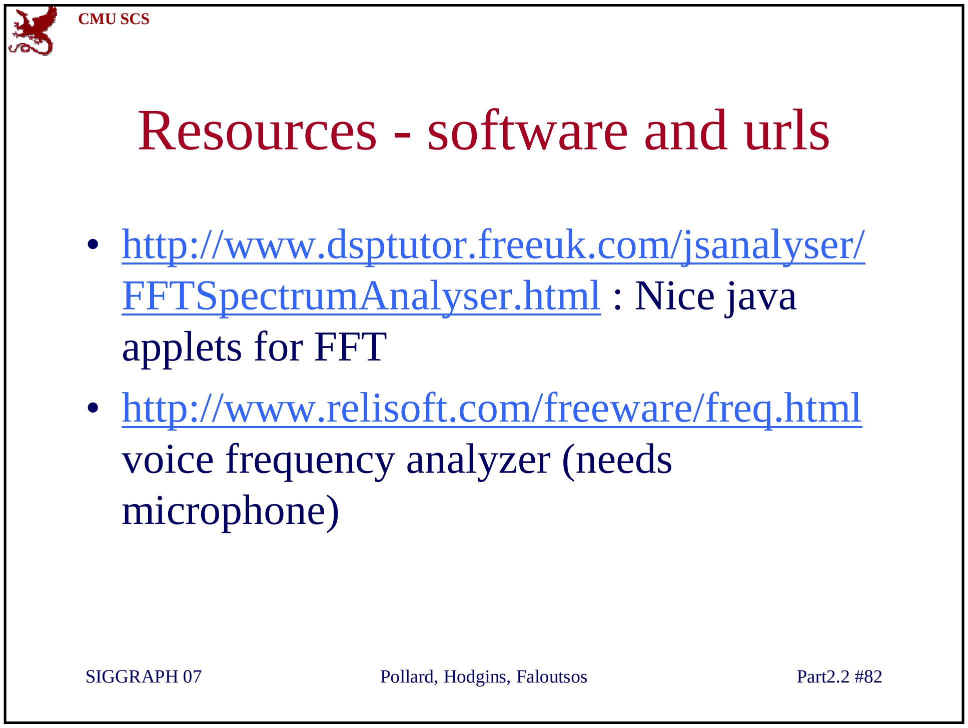
Advantages of Wavelets

- 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



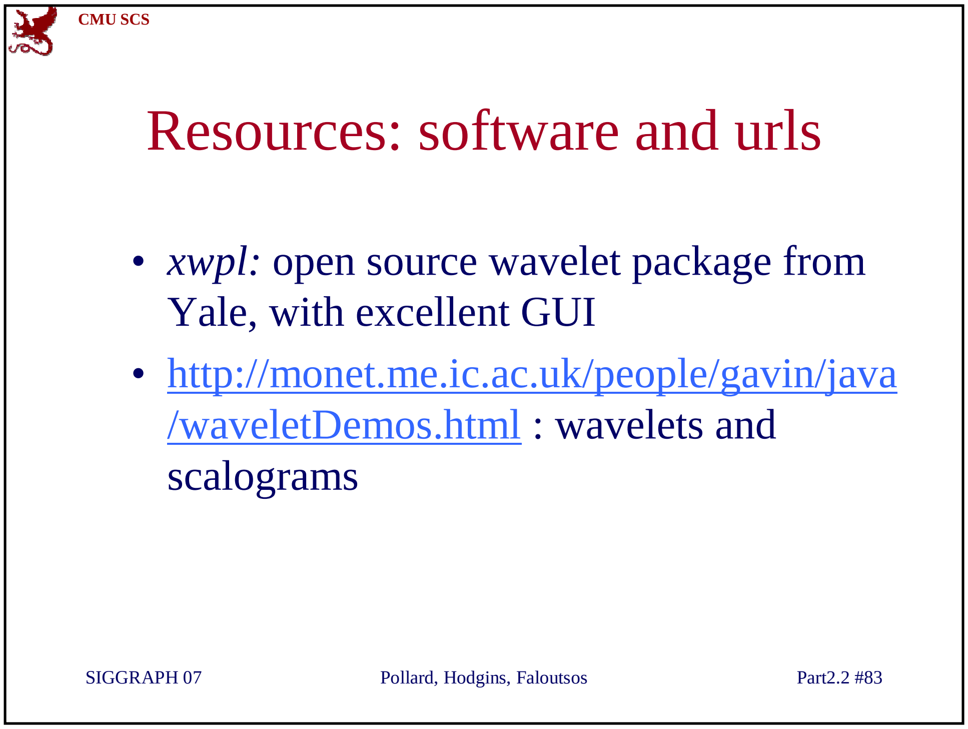
Overall Conclusions

- DFT, DCT spot periodicities
- **DWT** : multi-resolution - matches processing of mammalian ear/eye better
- All three: powerful tools for **compression**, **pattern detection** in real signals
- All three: included in math packages
  - (matlab, ‘R’, mathematica, … - often in spreadsheets!)



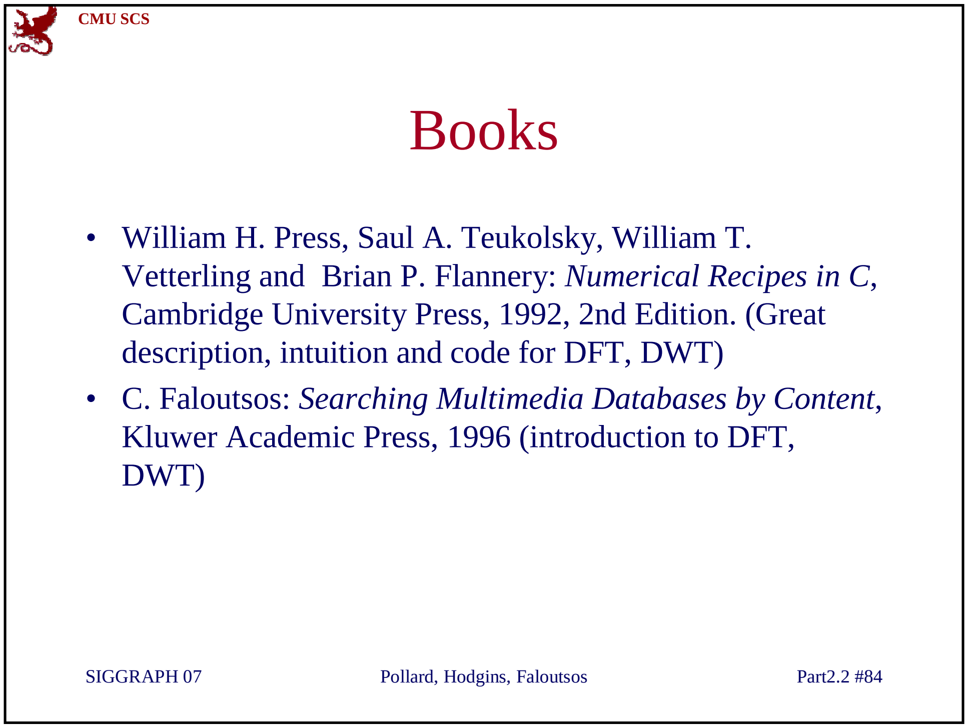
Resources - software and urls

- http://www.dsptutor.freeuk.com/jsanalyser/FFTSpectrumAnalyser.html : Nice java applets for FFT
- 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/waveletDemos.html : wavelets and scalograms



Books

- William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery: *Numerical Recipes in C*, Cambridge University Press, 1992, 2nd Edition. (Great description, intuition and code for DFT, DWT)
- C. Faloutsos: *Searching Multimedia Databases by Content*, Kluwer Academic Press, 1996 (introduction to DFT, DWT)

Database Techniques with Motion Capture

Part 2: Methods

Christos Faloutsos

CMU

Part 2.3:

Linear Forecasting

Outline

- Similarity Search and indexing
- Feature extraction
- ➡ Linear forecasting
  - – Auto-regression: Least Squares;
  - – RLS
  - – Co-evolving time sequences

Forecasting

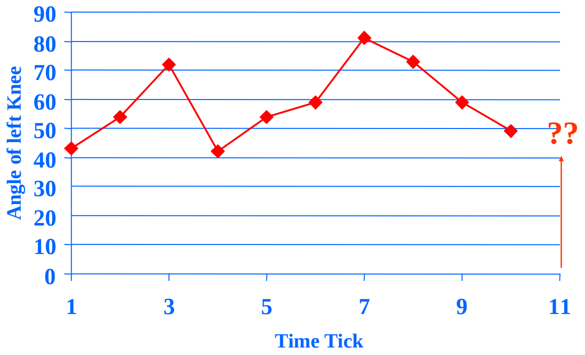
"Prediction is very difficult, especially about the future." - Nils Bohr

`http://www.hfac.uh.edu/MediaFutures/thoughts.html`

CMU SCS

Problem#3: Forecast

Given x_t, x_{t-1}, …, forecast x_{t+1}



SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #5

CMU SCS

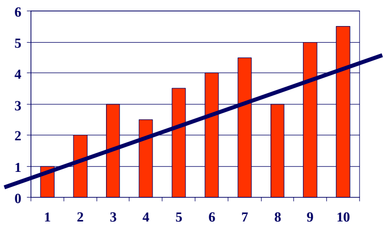
Forecasting: Preprocessing

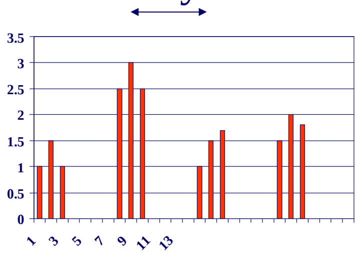
MANUALLY:

remove trends

spot periodicities

7 days

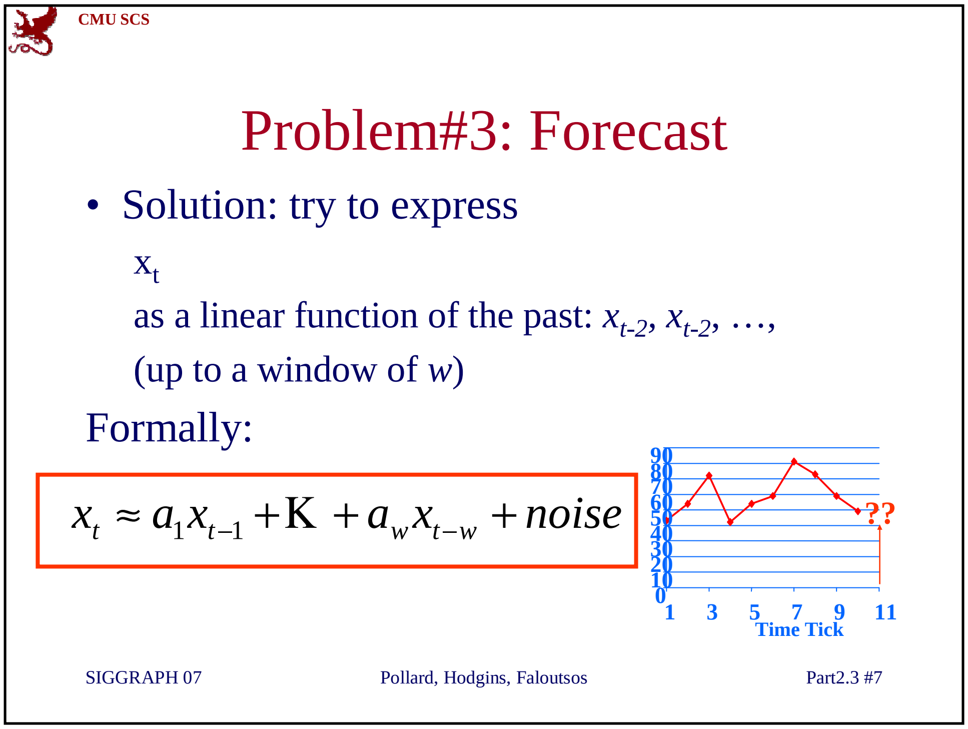




time

time

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #6



Problem#3: Forecast

- Solution: try to express

  x_t

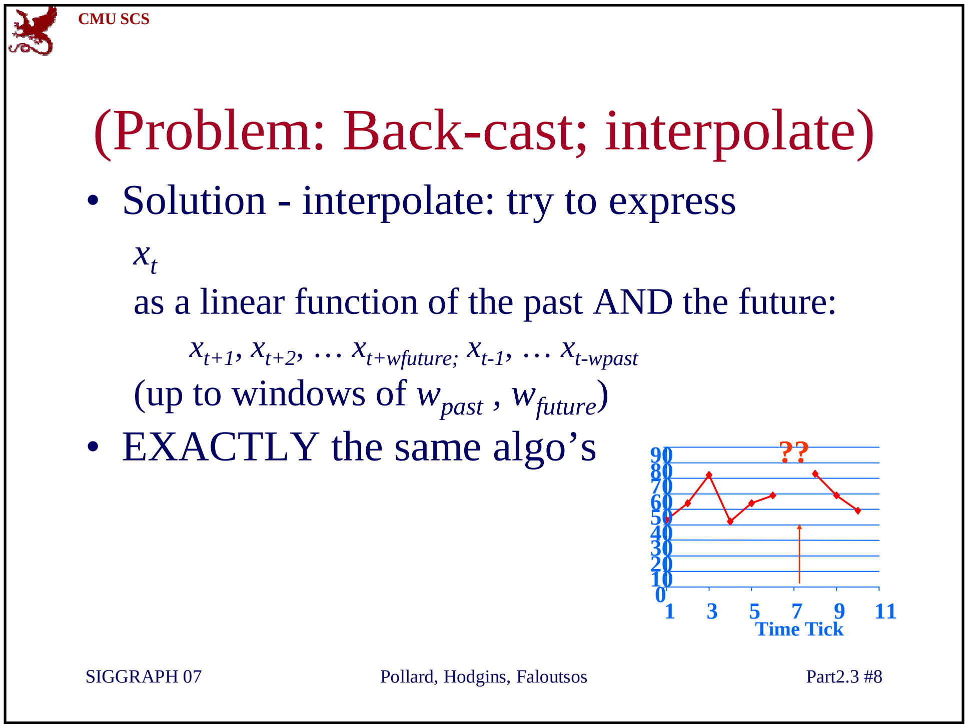
  as a linear function of the past: $x_{t\text{-}2}$, $x_{t\text{-}2}$, …,

  (up to a window of *w*)

Formally:

$$\boxed{x_t \approx a_1 x_{t-1} + \ldots + a_w x_{t-w} + noise}$$

90
80
70
60
50
40
30
20
10
0
1
3
5
7
9
11
Time Tick
??



(Problem: Back-cast; interpolate)

- Solution - interpolate: try to express

  x_t

  as a linear function of the past AND the future:

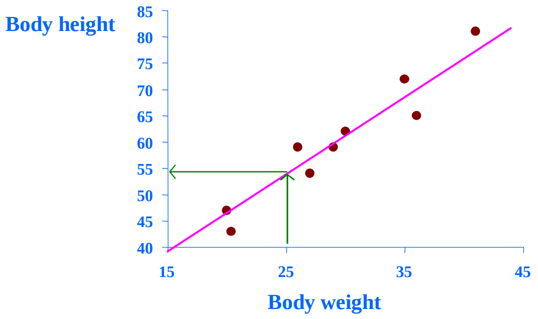
  x_{t+1}, x_{t+2}, … $x_{t+wfuture;}$ $x_{t\text{-}1}$, … $x_{t\text{-}wpast}$

  (up to windows of w_{past} , w_{future})
- EXACTLY the same algo’s

??
90
80
70
60
50
40
30
20
10
0
1
3
5
7
9
11
Time Tick

Linear Regression: idea

patient	*weight*	*height*
1	27	43
2	43	54
3	54	72
…	…	…
N	25	??



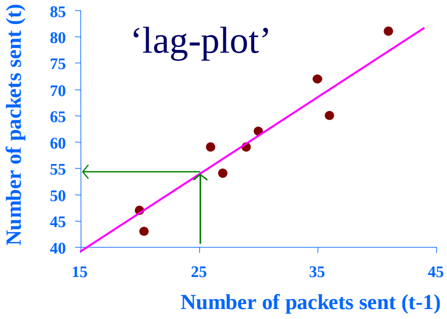
- express what we don’t know (= ‘dependent variable’)
- as a linear function of what we know (= ‘indep. variable(s)’)

Linear Auto Regression:

Time	*Packets Sent(t)*
1	43
2	54
3	72
…	…
N	??

Linear Auto Regression:

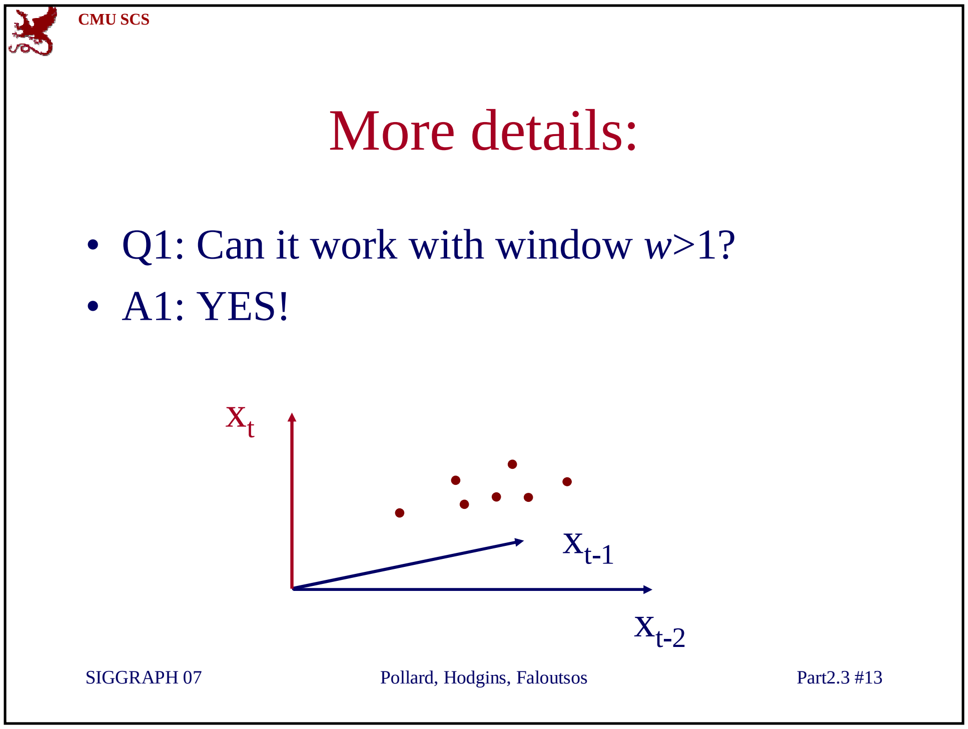
Time	*Packets Sent (t-1)*	*Packets Sent(t)*
1	-	43
2	43	54
3	54	72
	…	
…		…
N	25	??



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

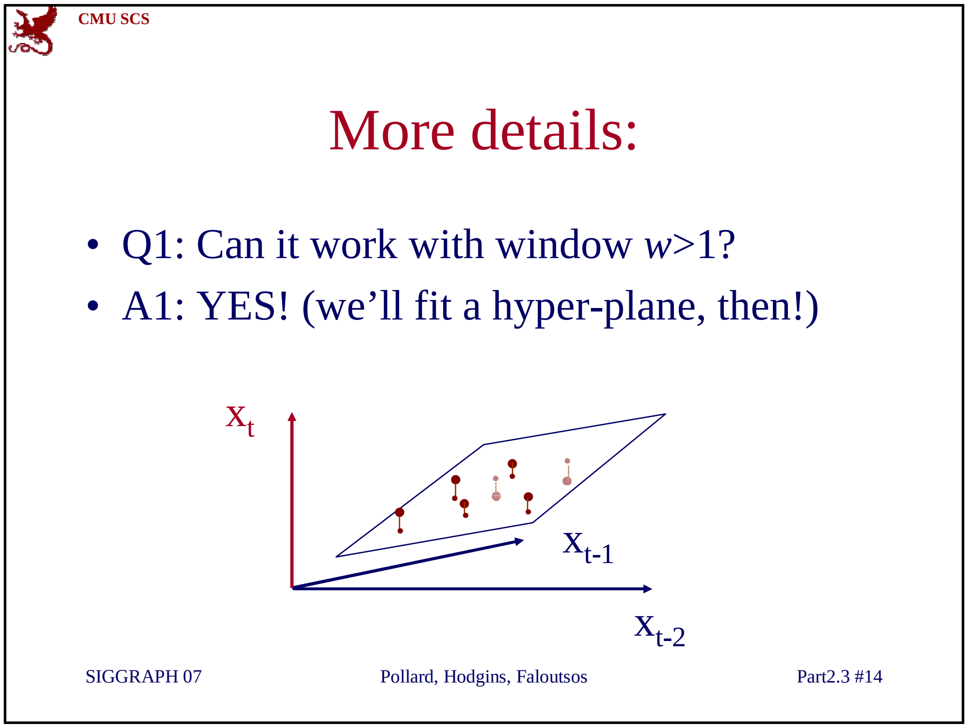
Outline

- Similarity Search and indexing
- Feature extraction
- Linear forecasting
  - – Auto-regression: **Least Squares**;
  - – RLS
  - – Co-evolving time sequences



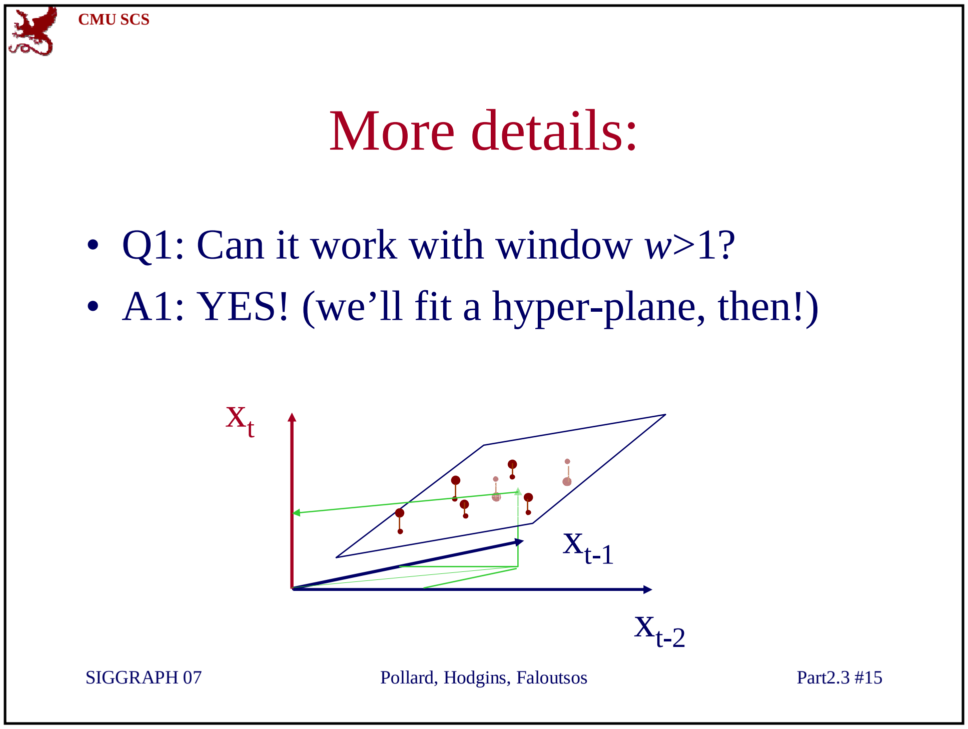
More details:

- Q1: Can it work with window *w*>1?
- A1: YES!



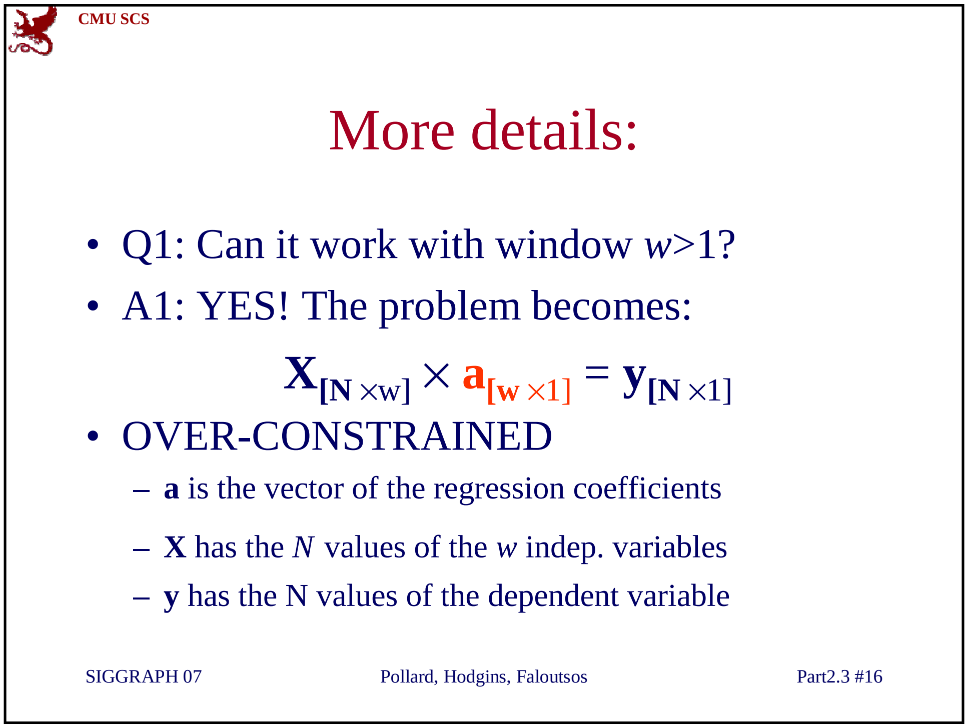
More details:

- 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!)

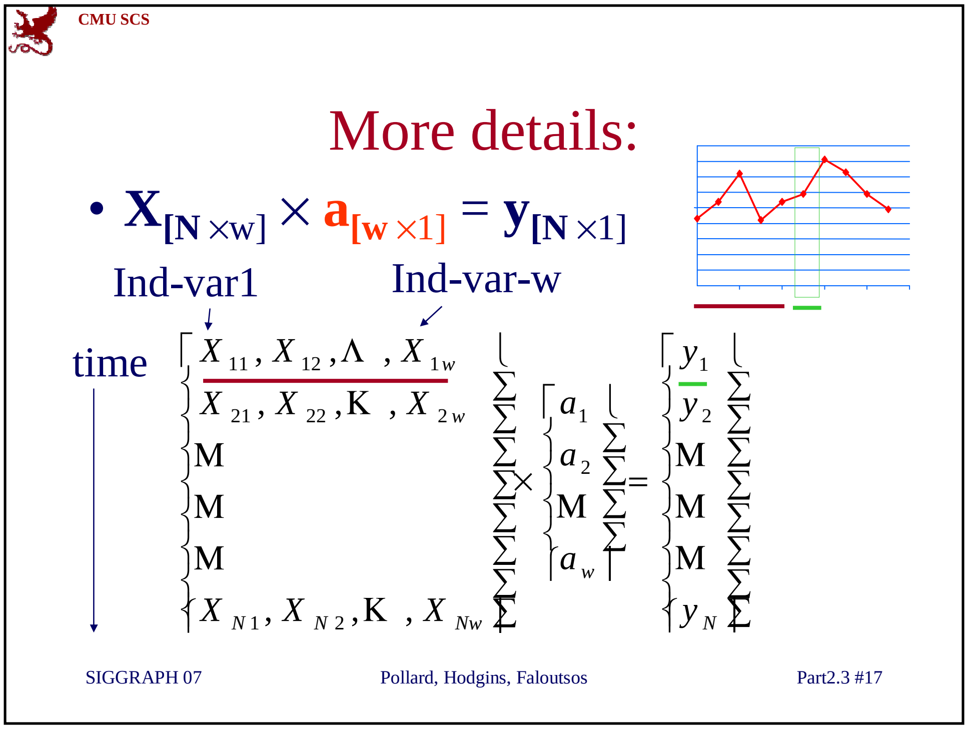


More details:

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

$$\mathbf{X}_{[\mathbf{N} \times \mathrm{w}]} \times \mathbf{a}_{[\mathbf{w} \times 1]} = \mathbf{y}_{[\mathbf{N} \times 1]}$$

- 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



CMU SCS

More details:

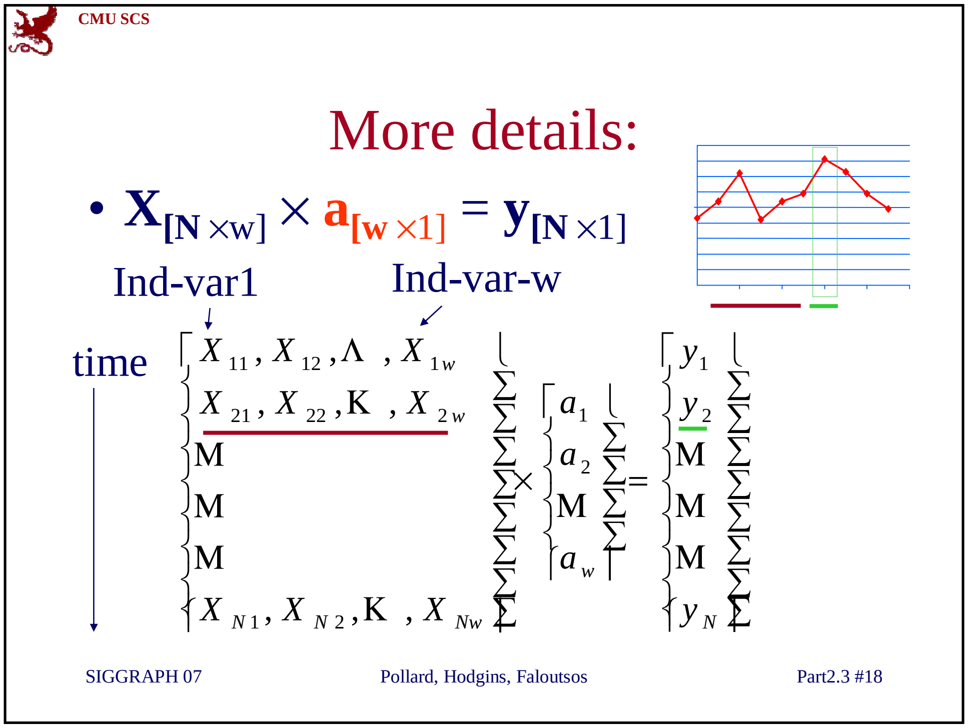
- $\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$

Ind-var1 Ind-var-w

time

$$\begin{bmatrix} \underline{X_{11}, X_{12}, \Lambda\ , X_{1w}} \\ X_{21}, X_{22}, K\ , X_{2w} \\ M \\ M \\ M \\ X_{N1}, X_{N2}, K\ , X_{Nw} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ M \\ a_w \end{bmatrix} = \begin{bmatrix} \underline{y_1} \\ y_2 \\ M \\ M \\ M \\ y_N \end{bmatrix}$$

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #17



CMU SCS

More details:

- $\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$

Ind-var1 Ind-var-w

time

$$\begin{bmatrix} X_{11}, X_{12}, \Lambda\ , X_{1w} \\ \underline{X_{21}, X_{22}, K\ , X_{2w}} \\ M \\ M \\ M \\ X_{N1}, X_{N2}, K\ , X_{Nw} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ M \\ a_w \end{bmatrix} = \begin{bmatrix} y_1 \\ \underline{y_2} \\ M \\ M \\ M \\ y_N \end{bmatrix}$$

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #18

More details

- Q2: How to estimate a_1, a_2, … a_w = **a**?
- A2: with Least Squares fit

$$\boxed{\mathbf{a} = (\mathbf{X}^{T} \times \mathbf{X})^{-1} \times (\mathbf{X}^{T} \times \mathbf{y})}$$

- (Moore-Penrose pseudo-inverse)
- **a** is the vector that minimizes the RMSE from **y**

Outline

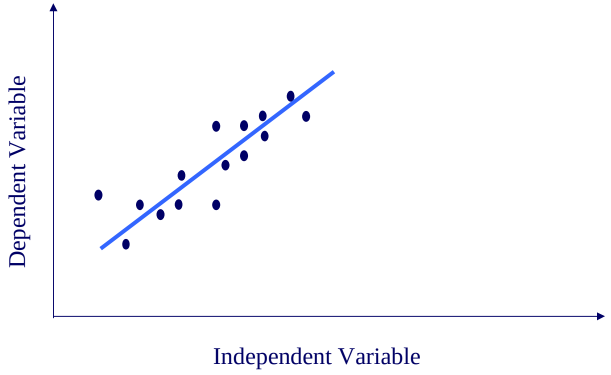
- Similarity Search and indexing
- Feature extraction
- Linear forecasting
  - Auto-regression: Least Squares;
  - RLS
  - Co-evolving time sequences

Even more details

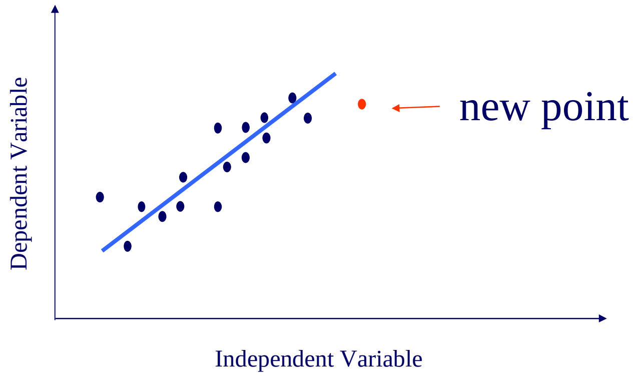
- Q3: Can we estimate **a** incrementally?
- A3: Yes, with the brilliant, classic method of ‘Recursive Least Squares’ (RLS) (see, e.g., [Yi+00], for details) - pictorially:

Even more details

- Given:

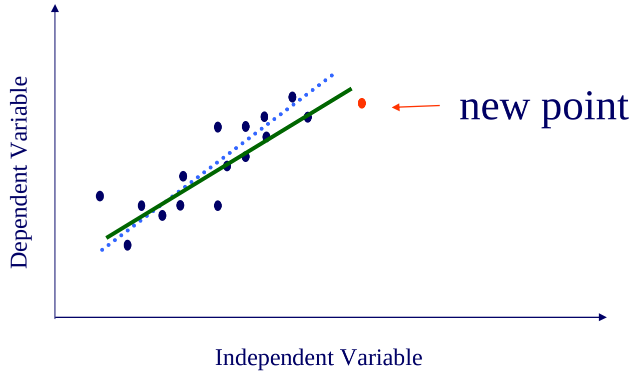


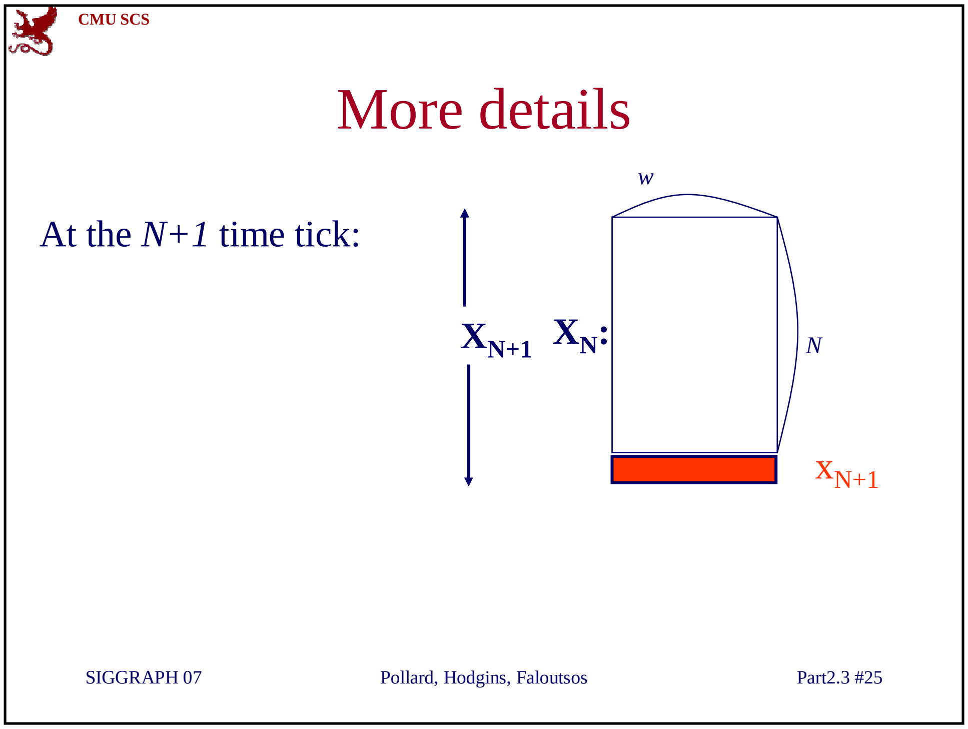
Even more details



Even more details

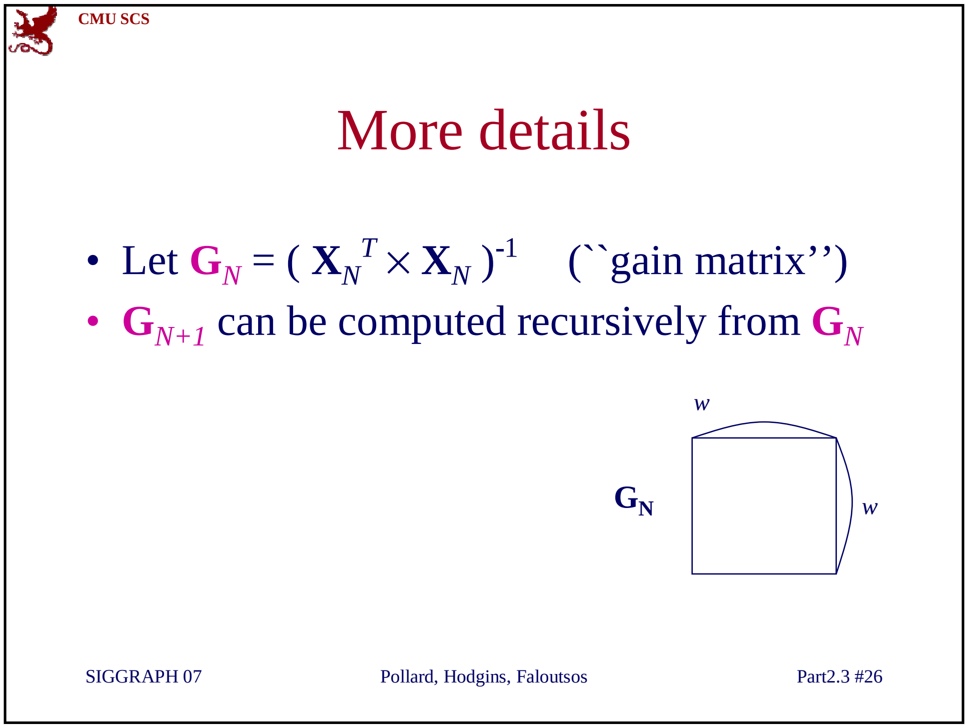
RLS: quickly compute new best fit





More details

At the *N+1* time tick:



More details

- Let $\mathbf{G}_N = (\mathbf{X}_N^T \times \mathbf{X}_N)^{-1}$ (``gain matrix'')
- $\mathbf{G}_{N+1}$ can be computed recursively from $\mathbf{G}_N$

Formula:

$$a = \boxed{[X_{N+1}^{T} \times X_{N+1}]^{-1}} \times [X_{N+1}^{T} \times y_{N+1}]$$

$$G_{N+1} \equiv [X_{N+1}^{T} \times X_{N+1}]^{-1}$$

$$G_{N+1} = G_N - [c]^{-1} \times [G_N \times x_{N+1}^{T}] \times x_{N+1} \times G_N$$

$$c = [1 + x_{N+1} \times G_N \times x_{N+1}^{T}]$$

Even more details

- Straightforward Least Squares
  - Needs huge matrix (**growing** in size) $O(N \times w)$
  - Costly matrix operation $O(N \times w^2)$

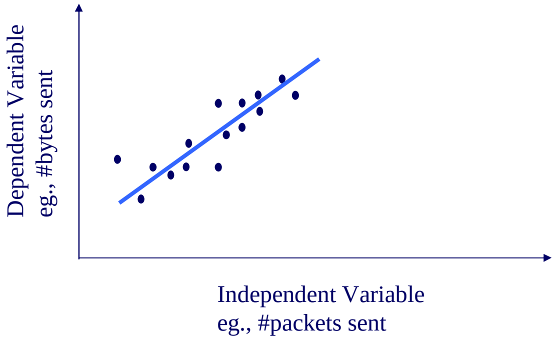
- Recursive LS
  - Need much smaller, fixed size matrix $O(w \times w)$
  - Fast, incremental computation $O(1 \times w^2)$

$\boxed{N = 10^6, \quad w = 1\text{-}100}$

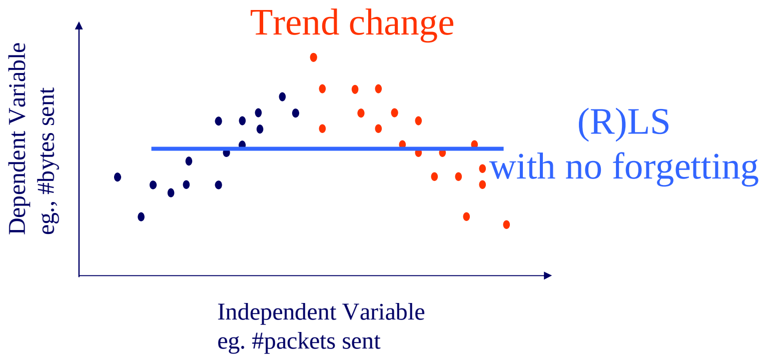
Even more details

- Q4: can we ‘forget’ the older samples?
- A4: Yes - RLS can easily handle that [Yi+00]:

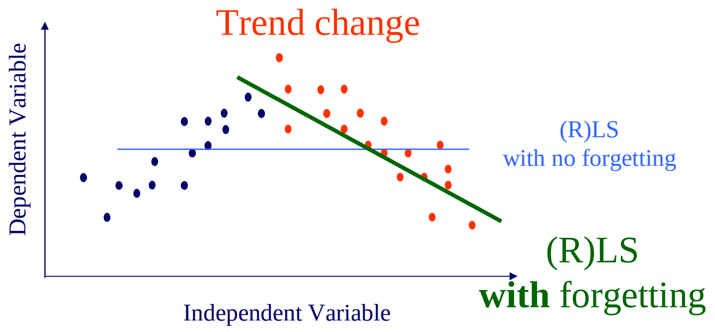
Adaptability - ‘forgetting’



Adaptability - ‘forgetting’



Adaptability - ‘forgetting’



- RLS: can *trivially* handle ‘forgetting’

skip

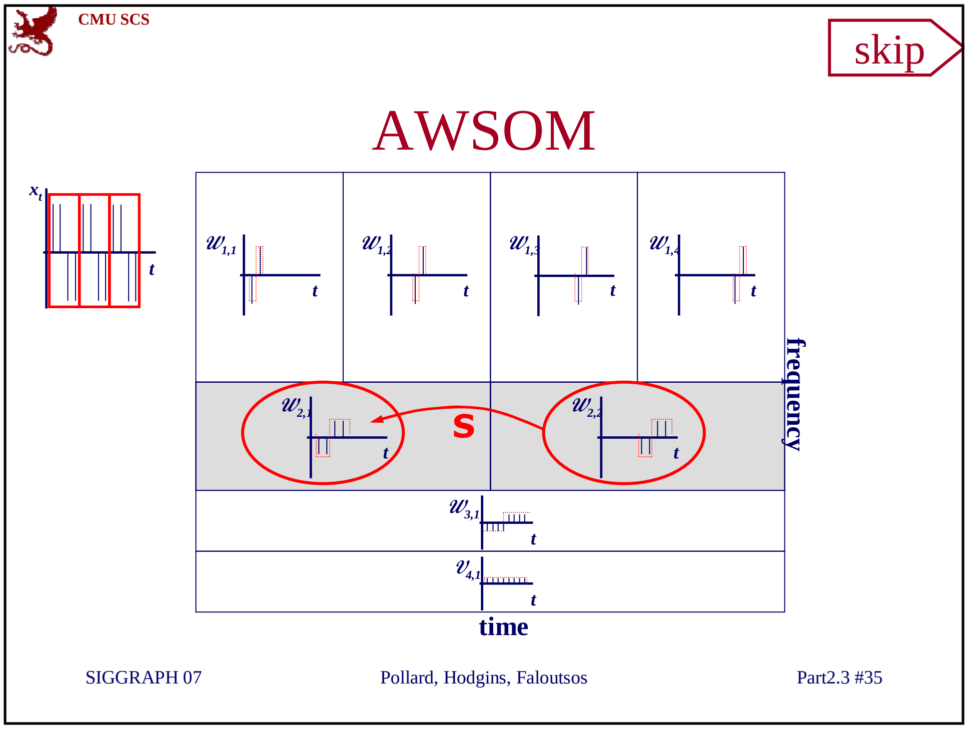
How to choose ‘*w*’?

- goal: capture arbitrary periodicities
- with NO human intervention
- on a semi-infinite stream

skip

Answer:

- ‘AWSOM’ (Arbitrary Window Stream fOrecasting Method) [Papadimitriou+, vldb2003]
- idea: do AR on each wavelet level
- in detail:

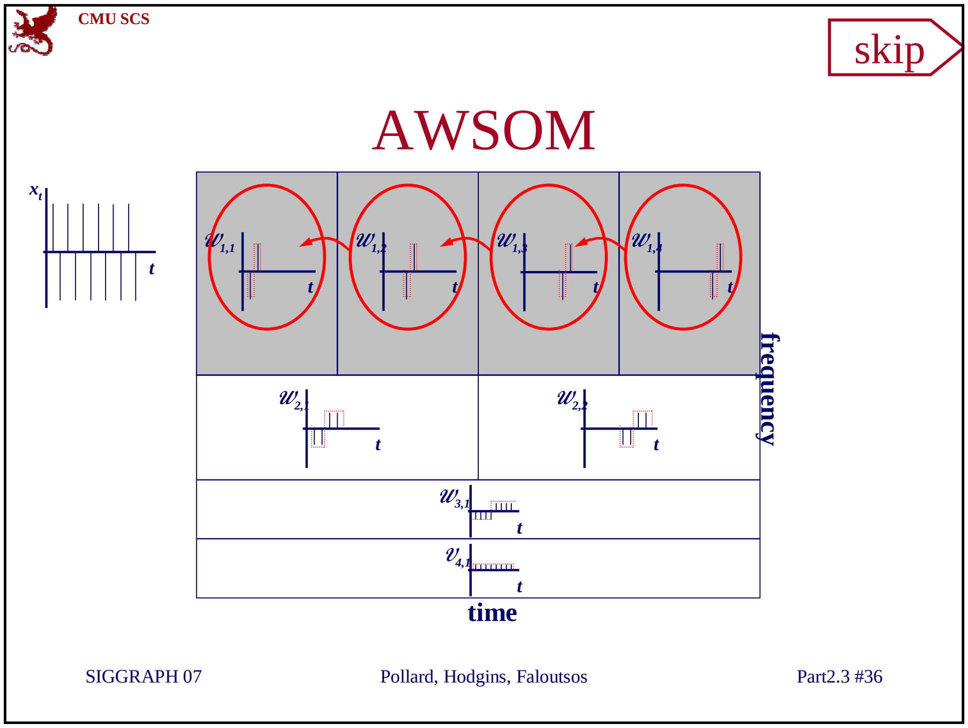


CMU SCS

skip

AWSOM

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #35



CMU SCS

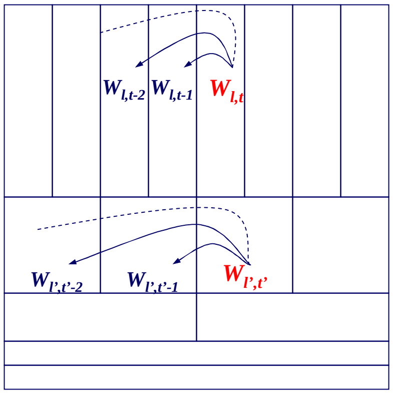
skip

AWSOM

SIGGRAPH 07 Pollard, Hodgins, Faloutsos Part2.3 #36

skip

AWSOM - idea



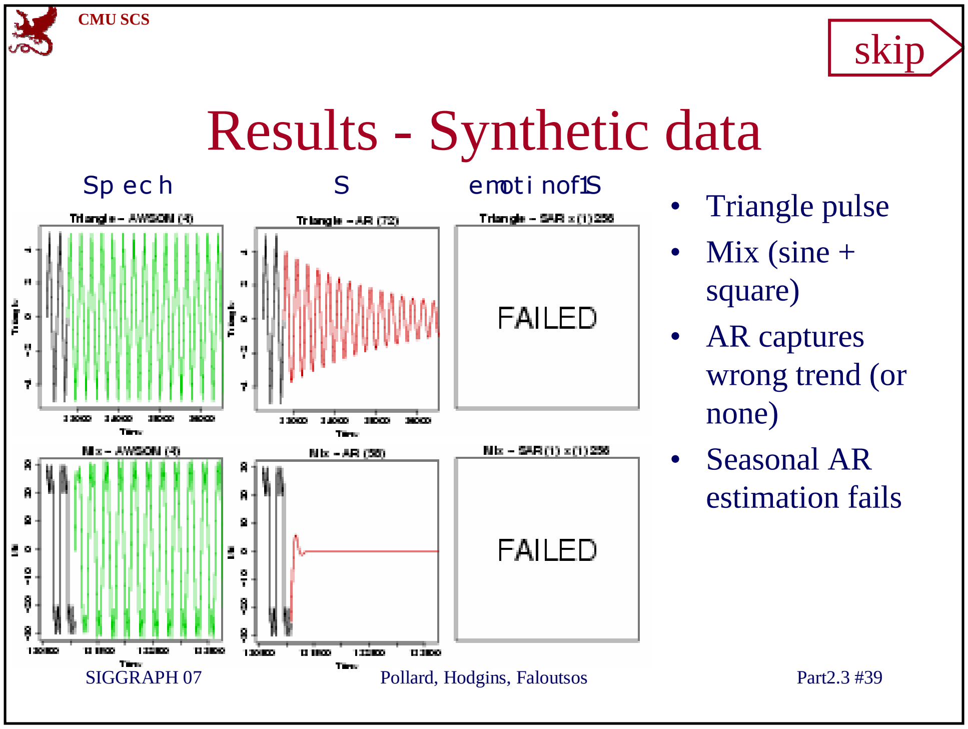
$W_{l,t} = \beta_{l,1} W_{l,t-1} + \beta_{l,2} W_{l,t-2} + \ldots$

$W_{l',t'} = \beta_{l',1} W_{l',t'-1} + \beta_{l',2} W_{l',t'-2} + \ldots$

skip

More details…

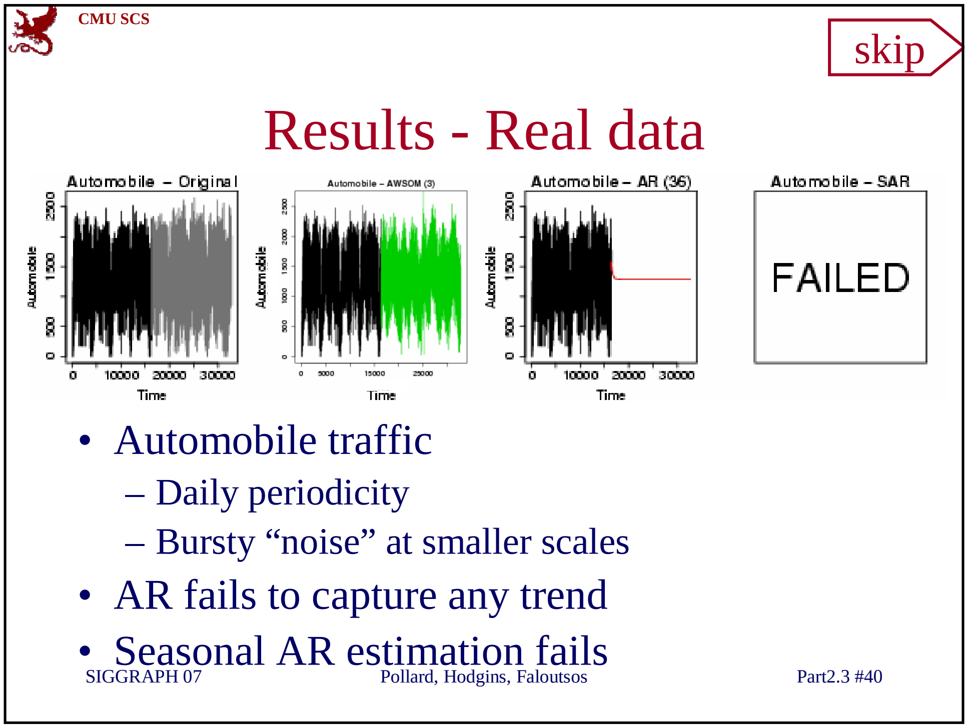
- Update of wavelet coefficients (incremental)
- Update of linear models (incremental; RLS)
- Feature selection (single-pass)
  - – Not all correlations are significant
  - – Throw away the insignificant ones (“noise”)



skip

Results - Synthetic data

- Triangle pulse
- Mix (sine + square)
- AR captures wrong trend (or none)
- Seasonal AR estimation fails



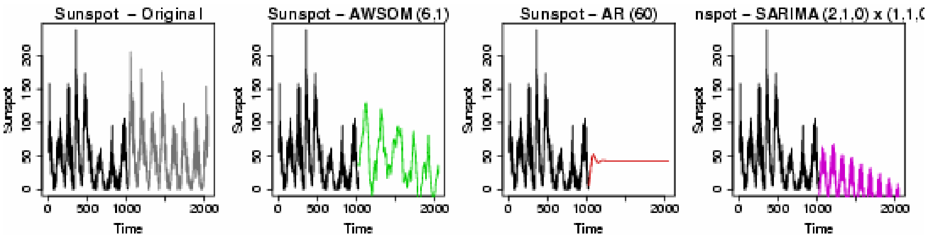
skip

Results - Real data

- Automobile traffic
  - Daily periodicity
  - Bursty “noise” at smaller scales
- AR fails to capture any trend
- Seasonal AR estimation fails

skip

Results - real data



- Sunspot intensity
  - Slightly time-varying “period”
- AR captures wrong trend
- Seasonal ARIMA
  - wrong downward trend, despite help by human!

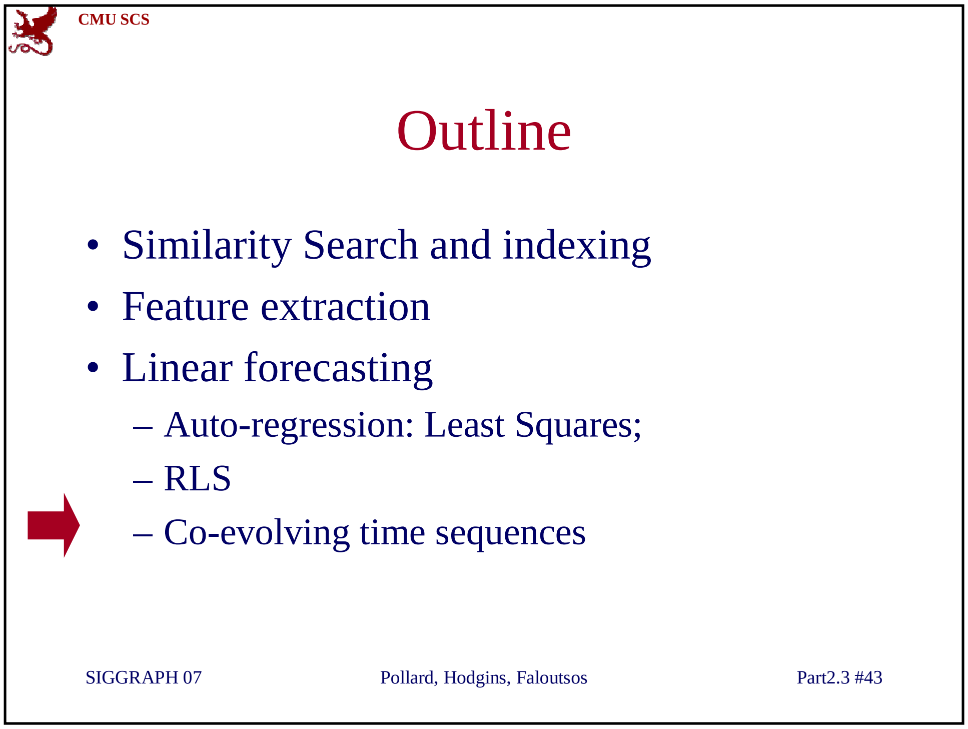
Skip

Complexity

- Model update

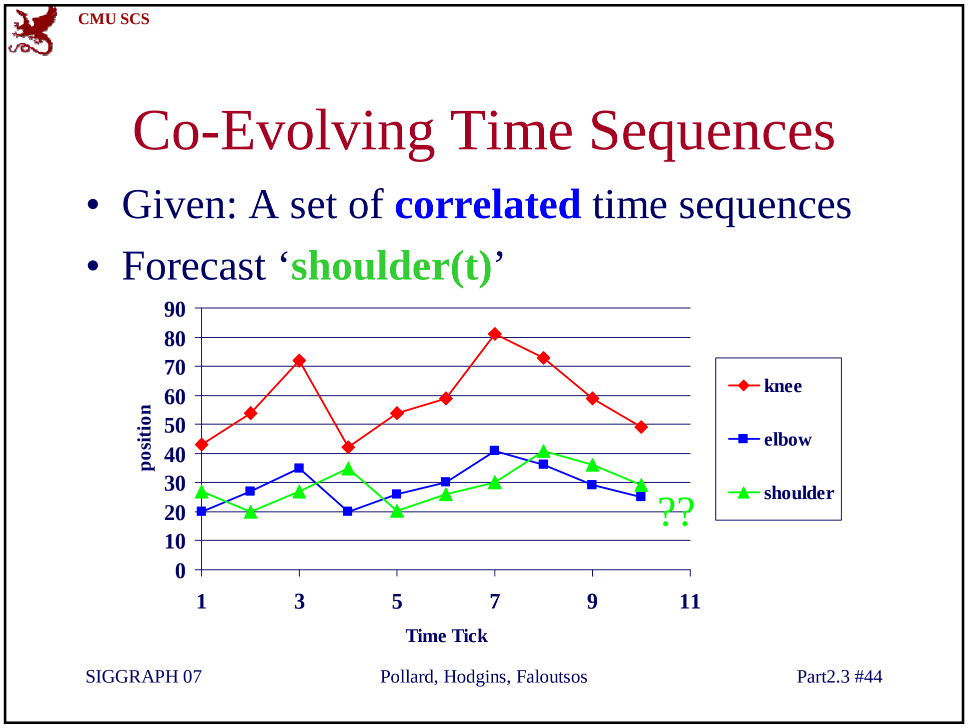
  Space: $O(lgN + mk^2) \approx O(lgN)$

  Time: $O(k^2) \approx O(1)$
- Where
  - ***N:*** number of points (so far)
  - ***k:*** number of regression coefficients; fixed
  - ***m:***number of linear models; $O(lgN)$



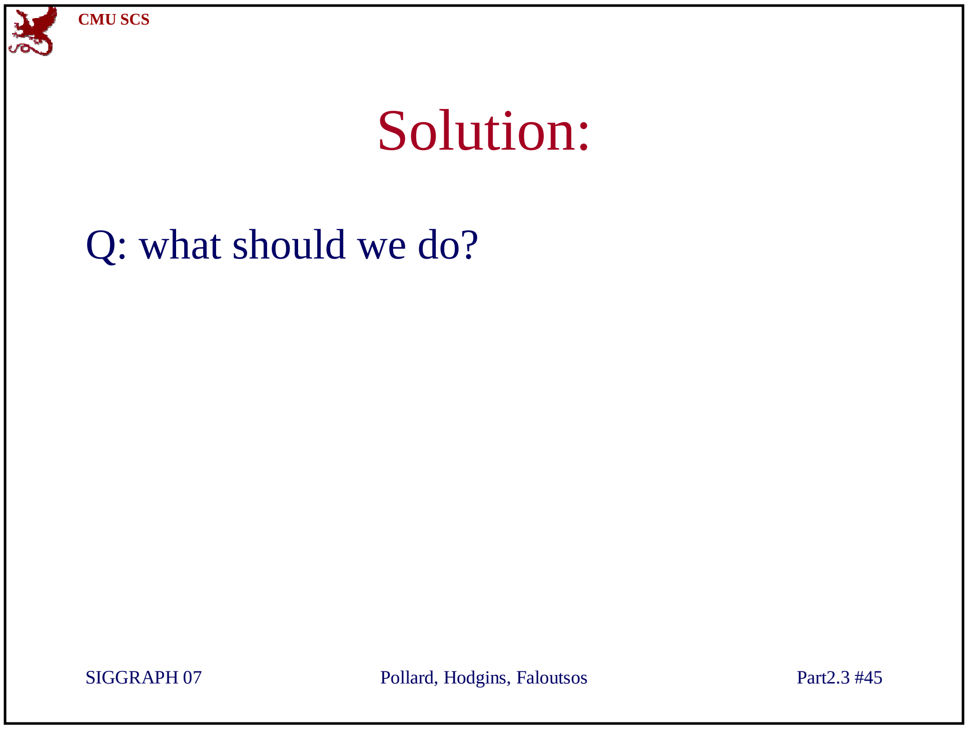
Outline

- Similarity Search and indexing
- Feature extraction
- Linear forecasting
  - – Auto-regression: Least Squares;
  - – RLS
  - ➡ – Co-evolving time sequences



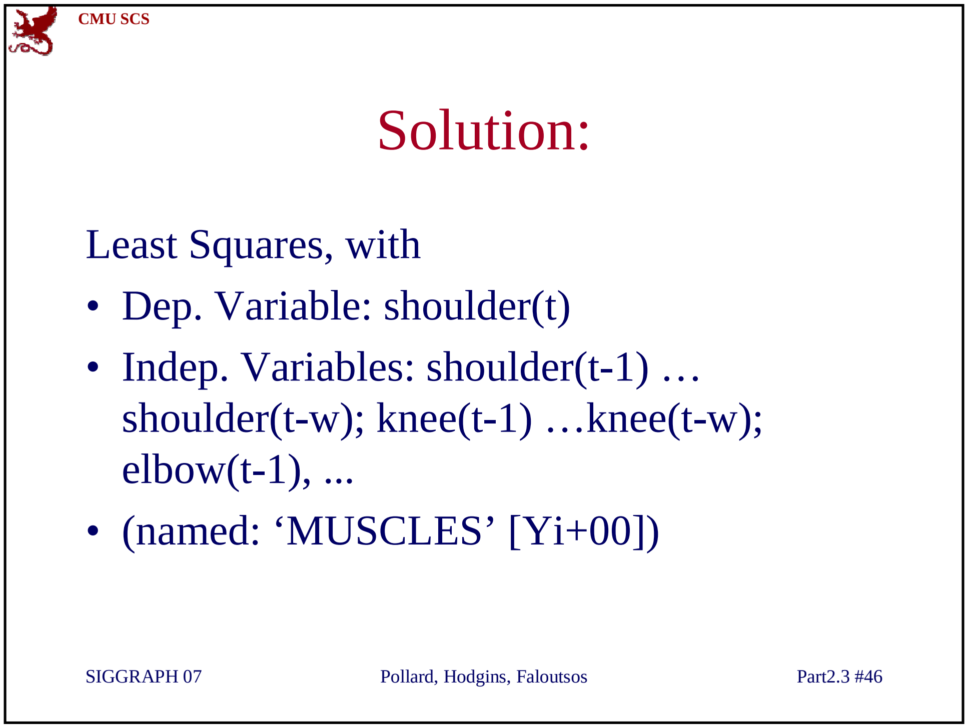
Co-Evolving Time Sequences

- Given: A set of **correlated** time sequences
- Forecast '**shoulder(t)**'



Solution:

Q: what should we do?



Solution:

Least Squares, with

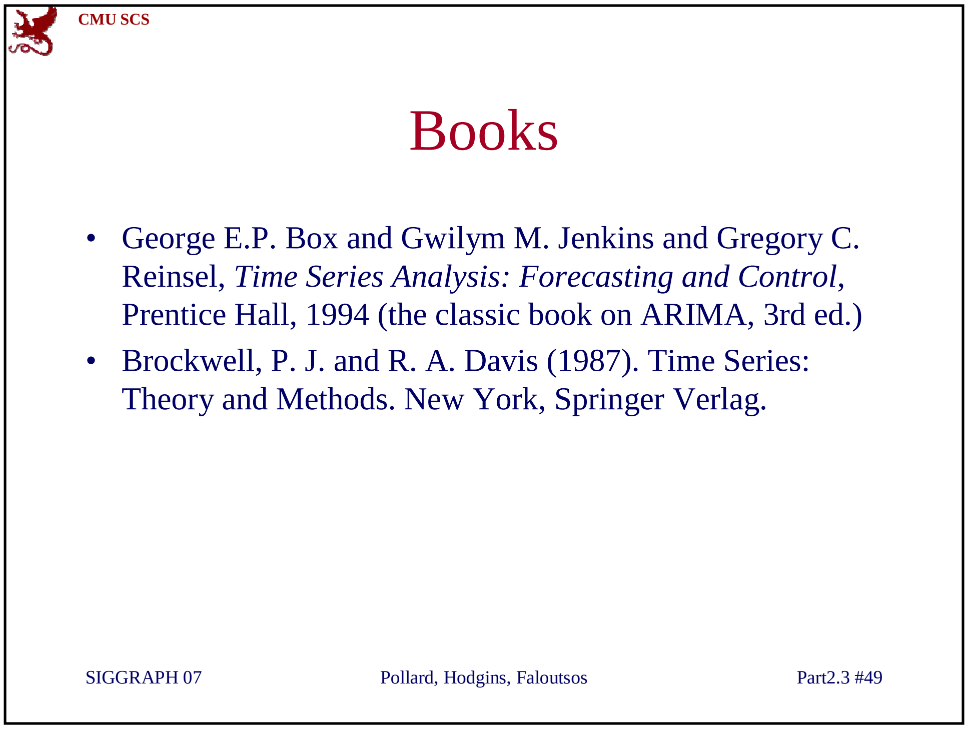
- Dep. Variable: shoulder(t)
- Indep. Variables: shoulder(t-1) … shoulder(t-w); knee(t-1) …knee(t-w); elbow(t-1), ...
- (named: ‘MUSCLES’ [Yi+00])

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]
- recently: AWSOM (no human intervention)

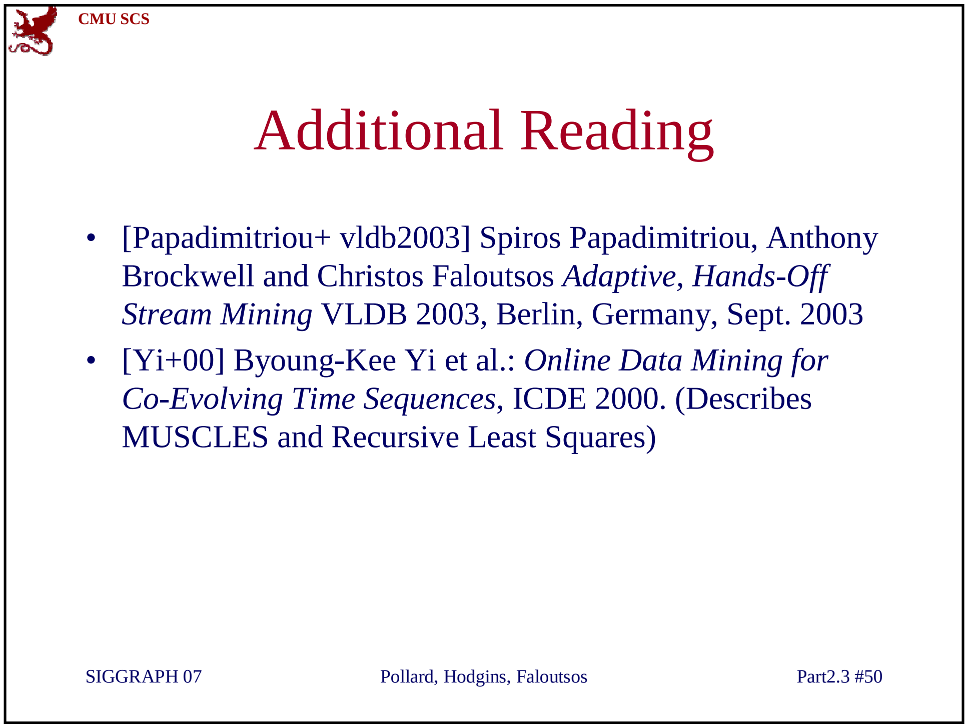
Resources: software and urls

- MUSCLES: Prof. Byoung-Kee Yi:
  `http://www.postech.ac.kr/~bkyi/`
  or `christos@cs.cmu.edu`
- free-ware: 'R' for stat. analysis
  (clone of Splus)
  `http://cran.r-project.org/`



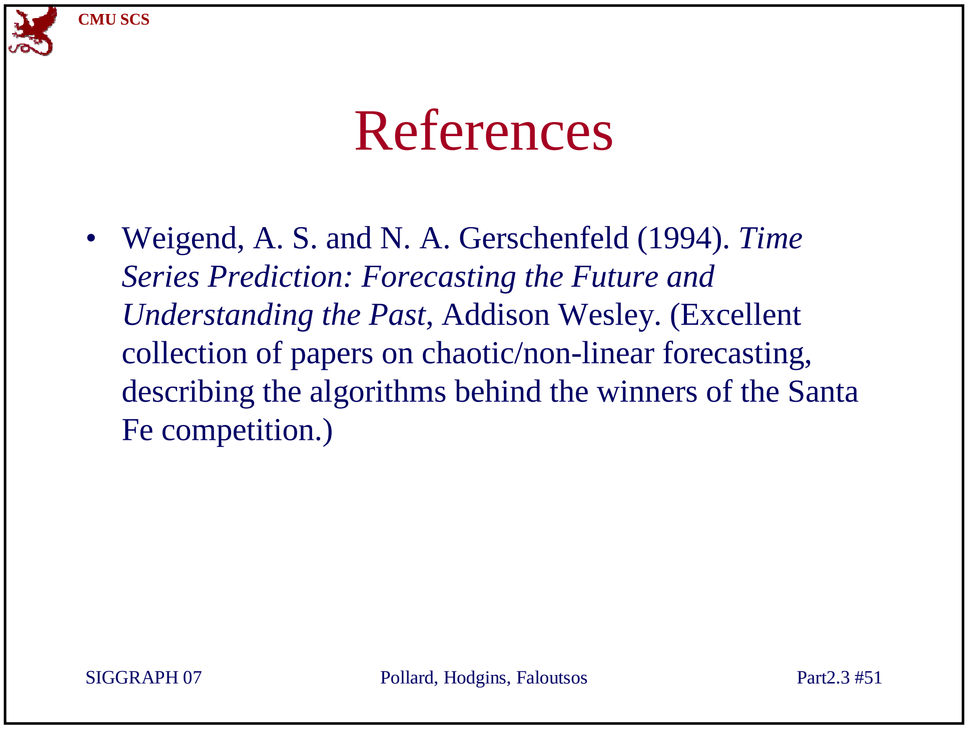
Books

- George E.P. Box and Gwilym M. Jenkins and Gregory C. Reinsel, *Time Series Analysis: Forecasting and Control*, Prentice Hall, 1994 (the classic book on ARIMA, 3rd ed.)
- Brockwell, P. J. and R. A. Davis (1987). Time Series: Theory and Methods. New York, Springer Verlag.



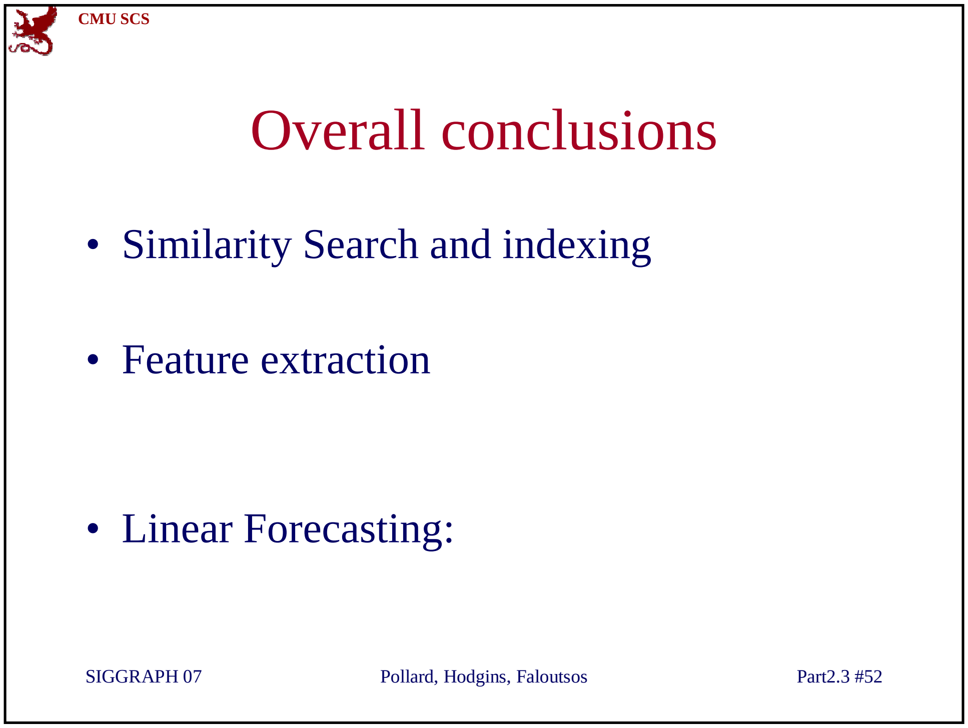
Additional Reading

- [Papadimitriou+ vldb2003] Spiros Papadimitriou, Anthony Brockwell and Christos Faloutsos *Adaptive, Hands-Off Stream Mining* VLDB 2003, Berlin, Germany, Sept. 2003
- [Yi+00] Byoung-Kee Yi et al.: *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000. (Describes MUSCLES and Recursive Least Squares)



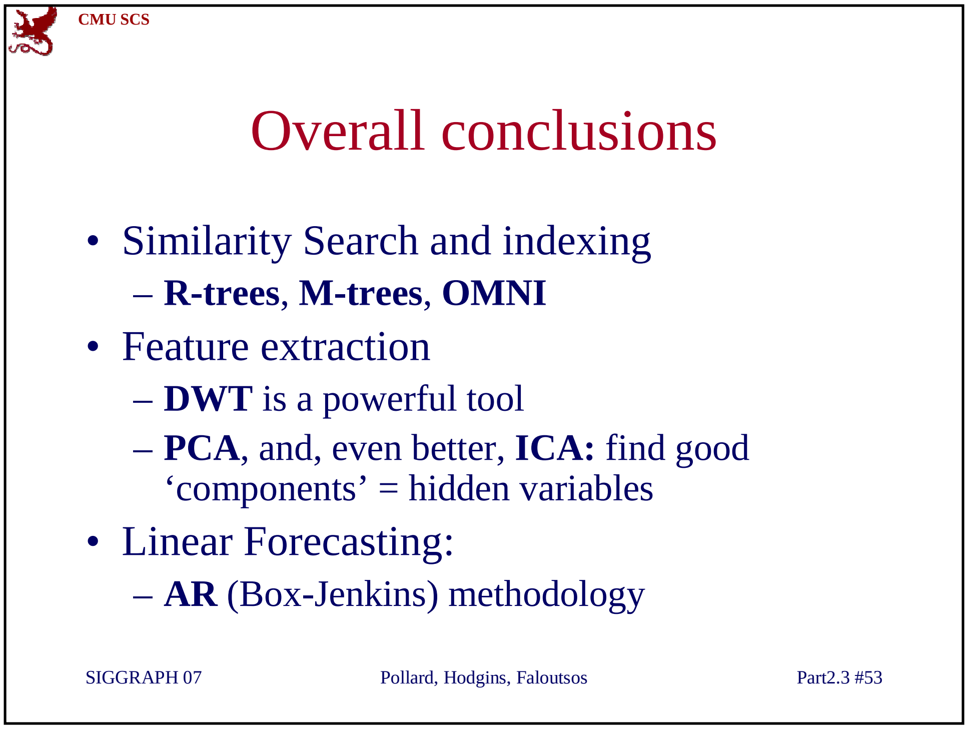
References

- Weigend, A. S. and N. A. Gerschenfeld (1994). *Time Series Prediction: Forecasting the Future and Understanding the Past*, Addison Wesley. (Excellent collection of papers on chaotic/non-linear forecasting, describing the algorithms behind the winners of the Santa Fe competition.)



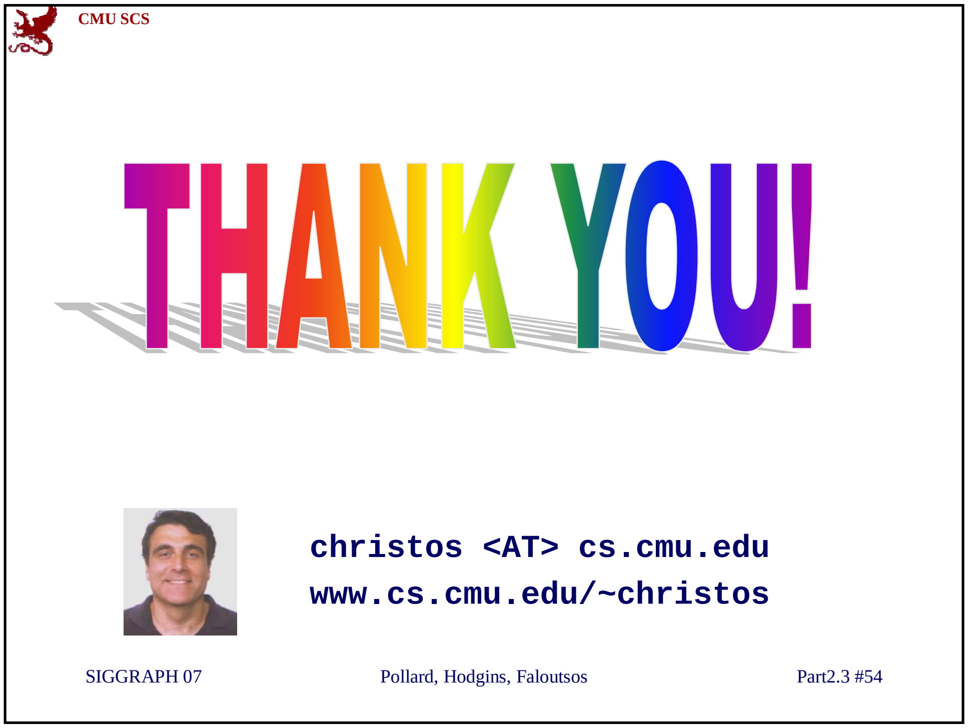
Overall conclusions

- Similarity Search and indexing
- Feature extraction
- Linear Forecasting:



Overall conclusions

- Similarity Search and indexing
  - **R-trees**, **M-trees**, **OMNI**
- Feature extraction
  - **DWT** is a powerful tool
  - **PCA**, and, even better, **ICA:** find good 'components' = hidden variables
- Linear Forecasting:
  - **AR** (Box-Jenkins) methodology



THANK YOU!

```
christos <AT> cs.cmu.edu
www.cs.cmu.edu/~christos
```