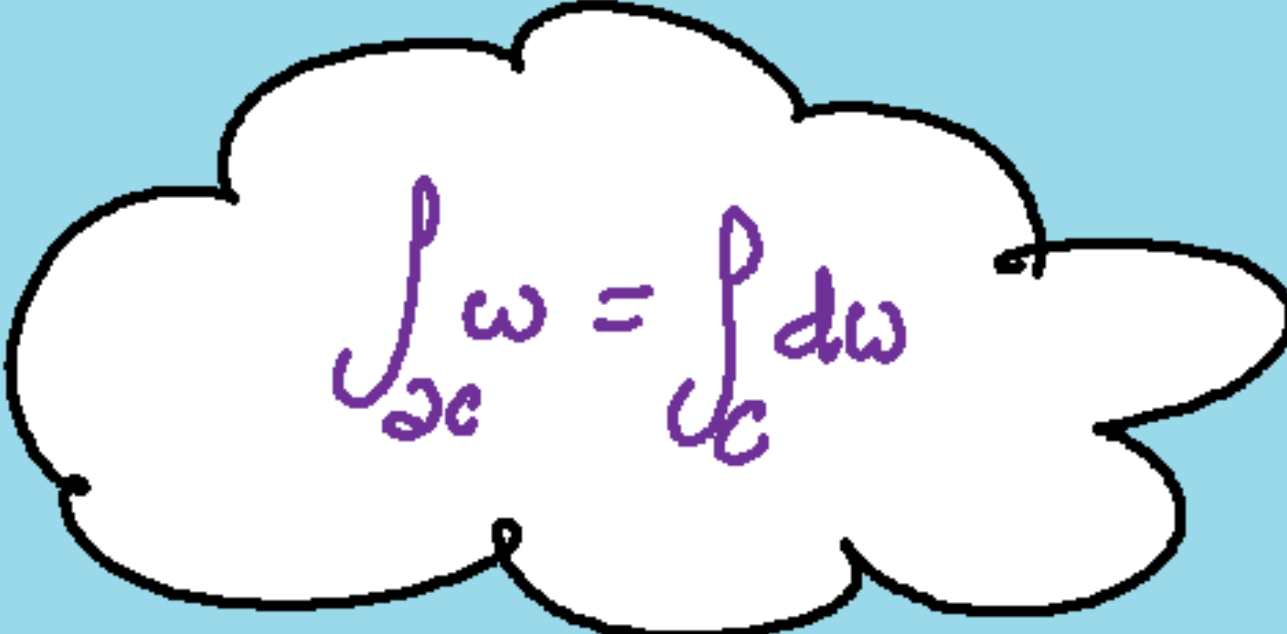


Writing on Clouds

Stephen M. Watt

Department of Computer Science
University of Western Ontario
London Ontario, Canada

A hand-drawn cloud with a black outline and a white interior, set against a light blue background. Inside the cloud, the mathematical equation $\int_{\mathcal{C}} \omega = \int_{\mathcal{C}} d\omega$ is written in purple ink. The equation consists of an integral symbol followed by the Greek letter ω , an equals sign, another integral symbol followed by $d\omega$.
$$\int_{\mathcal{C}} \omega = \int_{\mathcal{C}} d\omega$$

Outline

- A little bit about projects in our lab
- HW for Math and Math for HW

- Orthogonal series representation
- Writer-dependent HWR
- Cloud-based profile storage

The Pen as an Input Device

- Pen input for electronic devices is becoming important as an input modality.



- Pens can be used where keyboards can't, on very small or very large devices, in wet or dirty environments, by people with repetitive stress injuries.
- They also allow much easier 2-dimensional input, e.g. for drawings, music or mathematics.

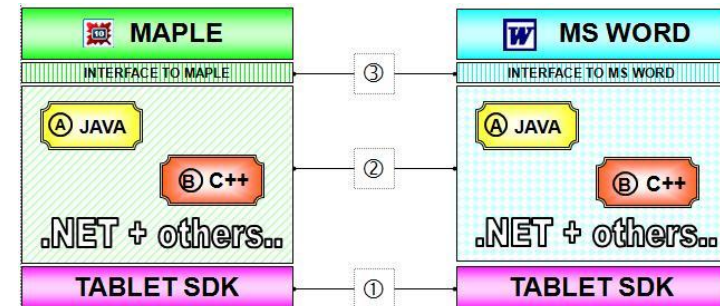
$$e^x = \int e^x dx = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$



Our Work in Pen-Based Computing

Long-term ongoing projects

- Mathematical handwriting recognition
 - for [computer algebra](#)
 - for [document processors](#)
 - Geometric and statistical methods
- Real time ink processing
- Multi-modal ink – [Skype](#) add on
- Portability of ink data – [InkML](#)
- Portability of ink software



Our Work in Pen-Based Computing

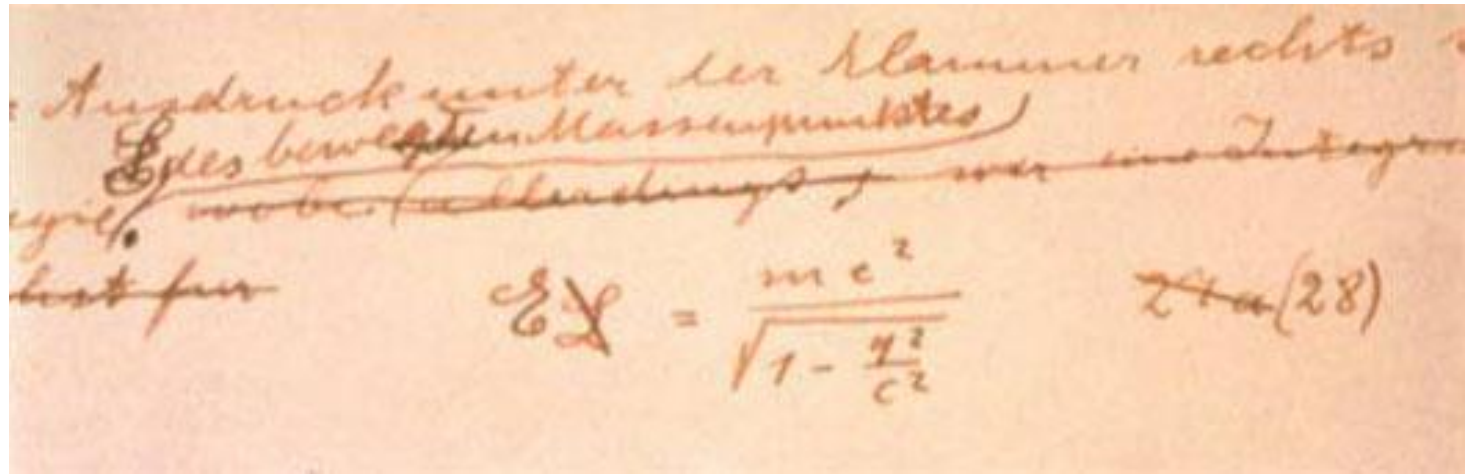
Newer projects

- Personal handwritten fonts
- Handwriting neatening
- Calligraphic rendering with InkML
- Ink compression
- Multi-language ink handling



$$A = \{x \in \mathbb{Q} \mid x \text{ را بتوان با منجر نزدیکتر از } 100 \text{ نوشت}\}$$

Pen-Based Math?



- All at this WS think it is a killer app for pen.
- 2D editing.
- Want to use it with CAS.

Pen-Based Math

- Does not require learning a special language:

$$\sum_{i=0}^r g_{r-i} x^i$$

`\sum_{i=0}^r g_{r-i} x^i`

`sum(g[r-i]*X^i, i = 0..r);`

- Different than natural language recognition:
 - 2-D layout is a combination of writing and drawing.
 - No fixed dictionary.
 - Many similar few-stroke characters. Well segmented.



Character Recognition

- Will concentrate on character recognition
- Multiple alphabets with similar characters
- Several projects ignoring this problem
- Three statisticians go hunting

Digital Ink Formats



- Collected by surface digitizer or camera
- Sequence of (x,y) points in time
sampled at some known frequency
+ possibly other info (angles, pressure, etc)
- Grouping into traces, letters, words + labelling



Ink Markup Language (InkML)

W3C Recommendation 20 September 2011

This version:

<http://www.w3.org/TR/2011/REC-InkML-20110920/>

Latest version:

<http://www.w3.org/TR/InkML>

Previous version:

<http://www.w3.org/TR/2011/PR-InkML-20110510/>

Editors:

Stephen M. Watt, University of Western Ontario and Maplesoft
Tom Underhill, Microsoft

Authors:

- Yi-Min Chee (until 2006 while at IBM)
- Katrin Franke (until 2004 while at Fraunhofer Gesellschaft)
- Max Froumentin (until 2006 while at W3C)
- Sriganesh Madhvanath (until 2009 while at HP)
- Jose-Antonio Magaña (until 2006 while at HP)
- Grégory Pakosz (until 2007 while at Vision Objects)
- Gregory Russell (until 2005 while at IBM)
- Muthuselvam Selvaraj (until 2009 while at HP)
- Giovanni Seni (until 2003 while at Motorola)
- Christopher Tremblay (until 2003 while at Corel)
- Larry Yaeger (until 2004 while at Apple)

Usual Character Reco. Methods

- Smooth and re-sample data *THEN*
- Match against N models by sequence alignment
OR
- Identify “features”, such as
 - Coordinate values of sample points, Number of loops, cusps, Writing direction at selected points, etc

Use a classification method, such as

- Nearest neighbour, Subspace projection, Cluster analysis, Support Vector Machine

THEN

- Rank choices by consulting dictionary

Difficulties

- Having many similar characters (e.g. for math) means comparison against all possible symbol models is slow.
- Determining features from points
 - Requires many *ad hoc* parameters.
 - Replaces measured points with interpolations
 - It is not clear how many points to keep, and most methods depend on number of points
 - Device dependent
- What to do since there is no dictionary?
- New ideas are needed!

Other Ways to Represent Traces

- For HWR do we need all the trace data?

- **Fundamental Thm of HWR:**

If it looks like an “A” then it can be an “A”.

Interpolation to 5 decimals is not necessary.

- Use functional approximation to represent traces more conveniently.

Orthogonal Series Representation

- **Main idea:**

Represent coordinate curves as truncated orthogonal series.

- **Advantages:**

- *Compact* – few coefficients needed
- *Geometric*
 - the truncation order is a property of the character set
 - gives a natural metric on the space of characters
- *Algebraic*
 - properties of curves can be computed algebraically (instead of numerically using heuristic parameters)
- *Device independent*
 - resolution of the device is not important

Inner Product and Basis Functions

- Choose a functional inner product, e.g.

$$\langle f, g \rangle = \int_{[a, b]} f(t) g(t) w(t) dt$$

- This determines an orthonormal basis in the subspace of polynomials of degree d .

Determine ϕ_i using GS on $\{1, t, t^2, t^3, \dots\}$.

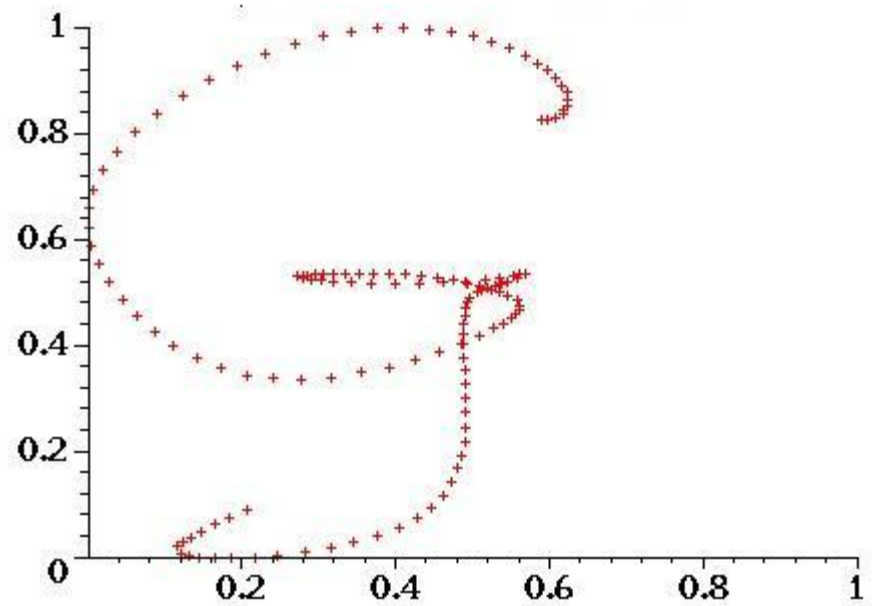
- Can then approximate functions in subspaces

$$A(t) \approx \sum_{i=0}^d \alpha_i \phi_i(t) \quad \alpha_i = \langle A(t), \phi_i(t) \rangle$$

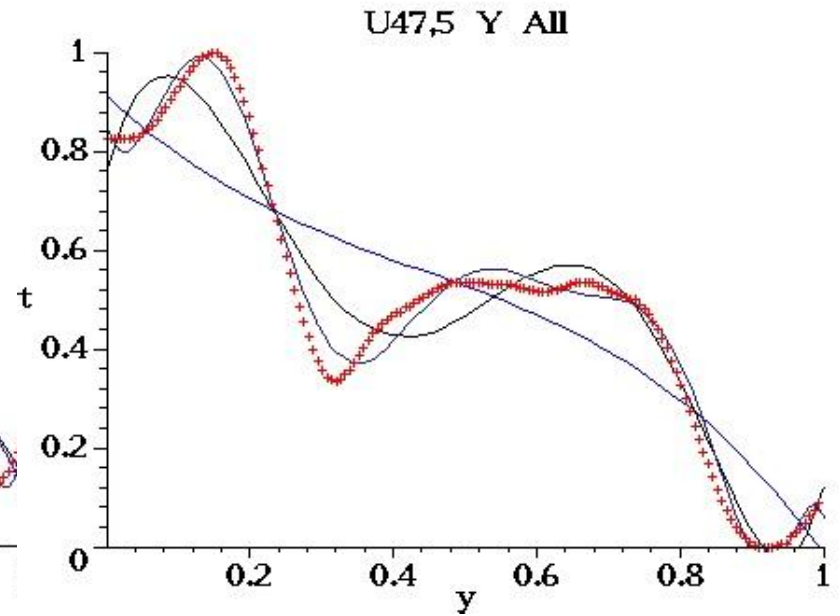
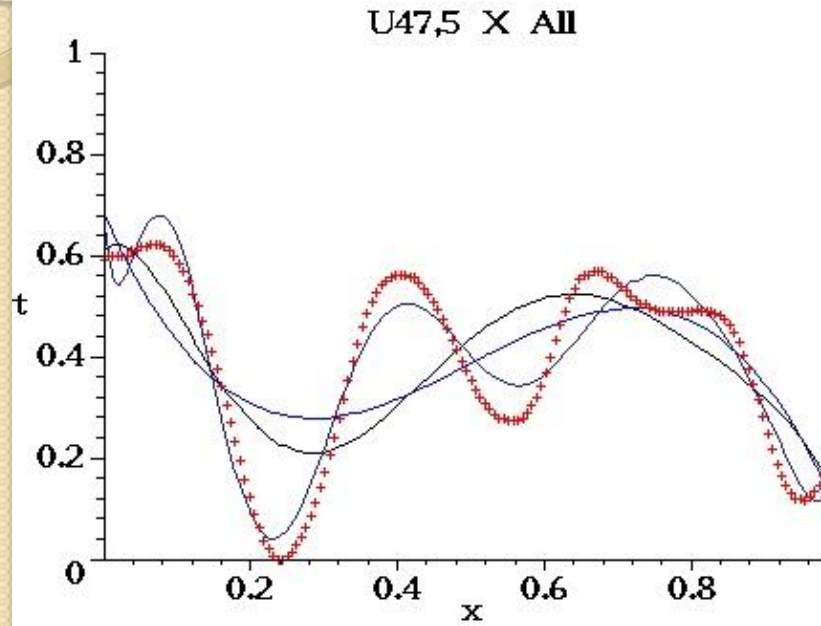
First Look: Chebyshev Series

- Initially used Chebyshev series [Char+SMW ICDAR 2007].
- Found could approximate closely (small RMS error) with series of order 10.
- Like symbols formed clusters.

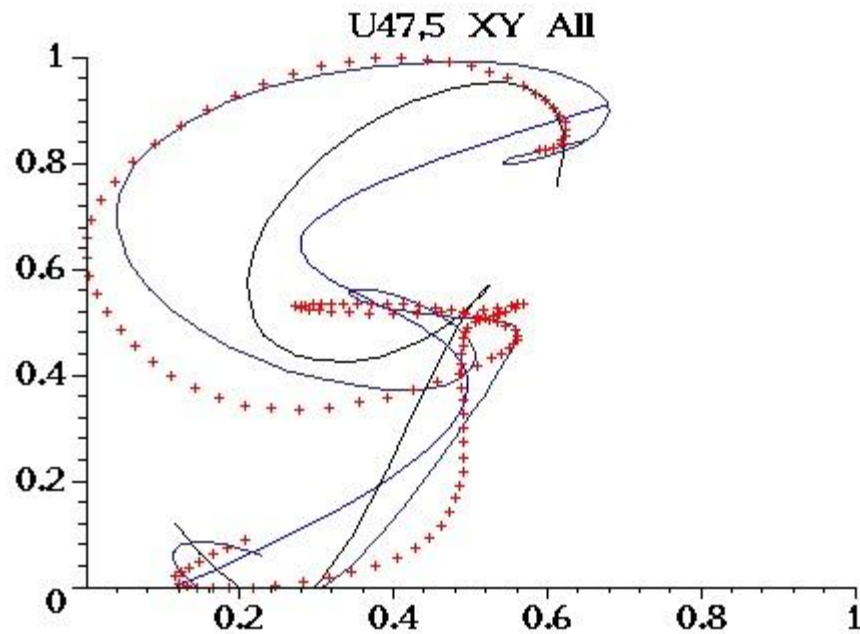
Raw Data for Symbol G



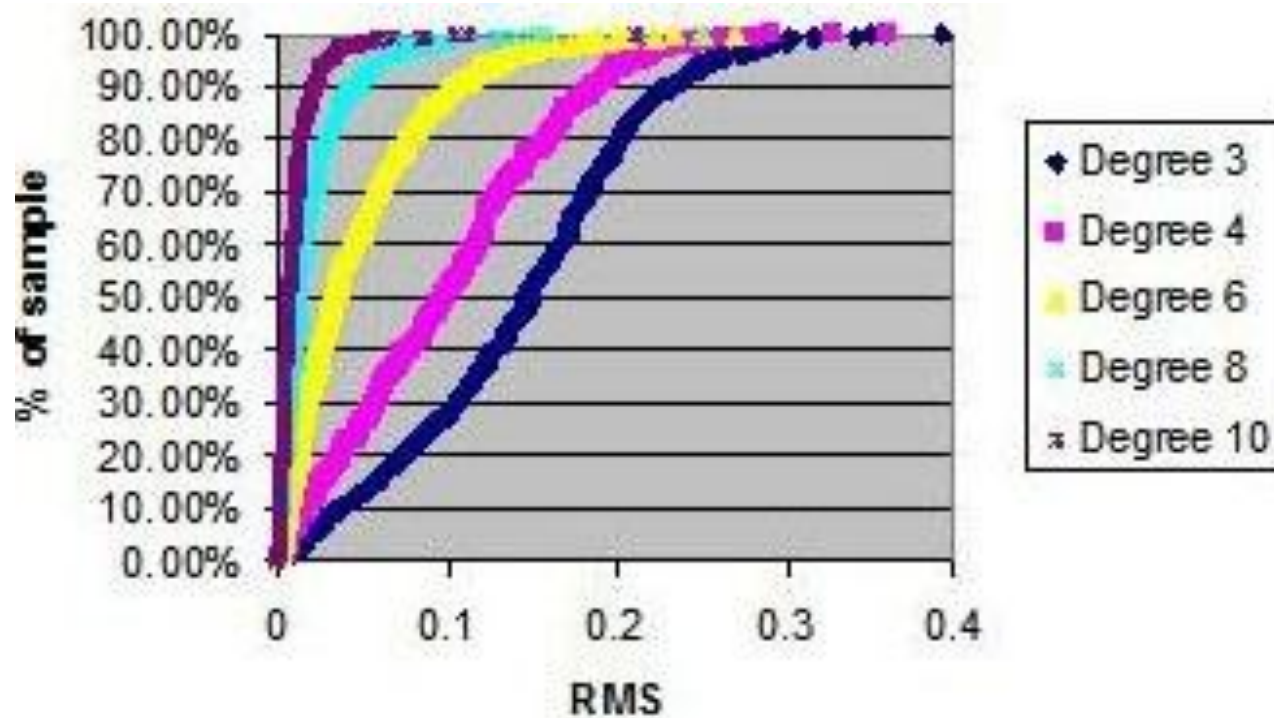
Coordinate fn approximations



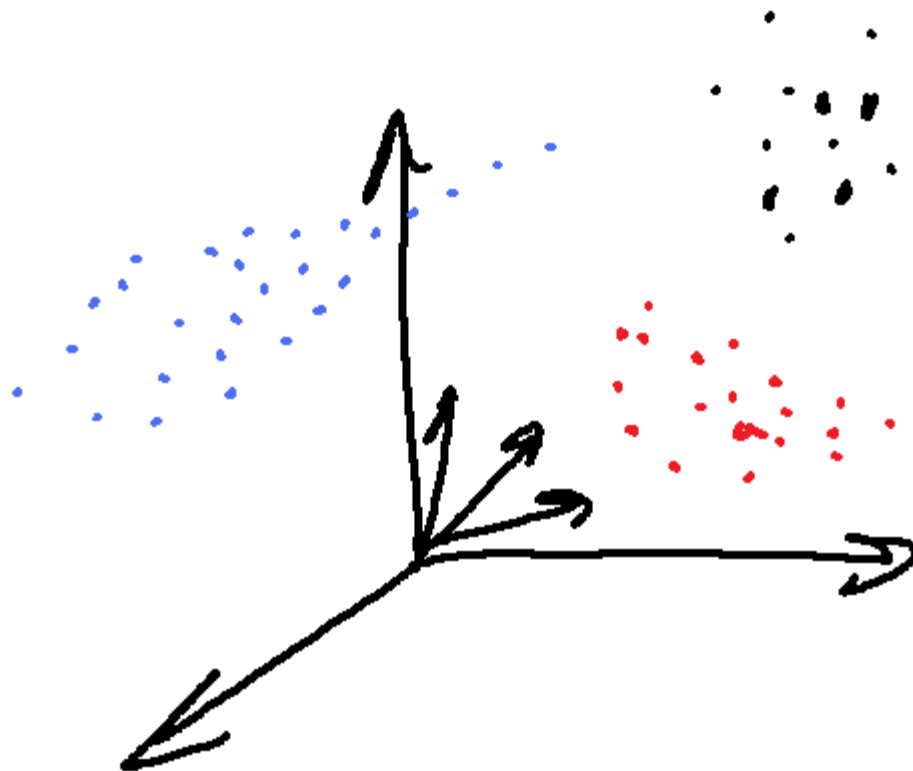
Chebyshev Approx to Character



RMS Error

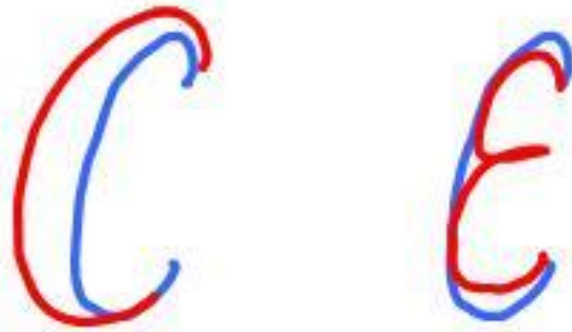


Like Symbols form Clouds



Problems

- Want fast response –
how to work while trace is being captured.
- Low RMS does not mean similar shape.



Pb 1. On-Line Ink

- The main problem:
In handwriting recognition, the human and the computer take turns thinking and sitting idle.
- We ask:
Can the computer do useful work while the user is writing and thereby get the answer faster after the user stops writing?
- We show:
The answer is “Yes”!

An On-Line Complexity Model

- Input is a sequence of N values received at a uniform rate.
- Characterize an algorithm by
 - $T_{\Delta}(n)$ no. operations as n -th input is seen
 - $T_F(n)$ no. operations after last input is seen

- Write on-line time complexity as

$$\text{OL}_n[T_{\Delta}(n), T_F(n)]$$

- E.g., linear insertion requires time

$$\text{OL}_n[O(n), 0]$$

On-Line Series Coefficients (main idea)

- If we choose the right basis functions, then the series coefficients can be computed on line.
[Golubitsky+SMW CASCON 2008, ICFHR 2008]
- The series coefficients are linear combinations of the moments, which can be computed by numerical integration as the points are received.
- This is the **Hausdorff moment problem** (1921), shown to be unstable by Talenti (1987).
- It is just fine, however, for the orders we need.

On-Line Series Coefficients (more details)

- Use Legendre polynomials P_i as basis on the interval $[-1,1]$, with weight function 1.
- Collect numerical values for $f(\lambda)$ on $[0, L]$.
 λ = arc length.
 L is not known until the pen is lifted.
- As the numerical values are collected, compute the moments $\int \lambda^i f(\lambda) d\lambda$.
- After last point, compute series coeffs for f with domain and range scaled to $[-1,1]$.
These will be linear combinations of the moments.

Complexity

- The on-line time complexity to compute coefficients for a Legendre series truncated to degree d is then

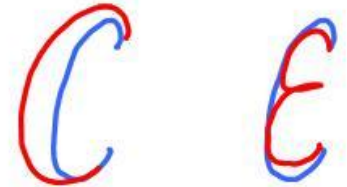
$$T_{\Delta} = 2(d + 2)$$

$$T_F = \frac{3}{2}d^2 + \frac{11}{2}d + 10.$$

- The time at pen up is *constant* with respect to the number of points.

Pb 2. Shape vs Variation

- The corners are not in the right places.



- Work in a jet space to force coords & derivatives close.
- Use a Legendre-Sobolev inner product

$$\langle f, g \rangle = \int_{-1}^1 f(t)g(t)dt + \mu_1 \int_{-1}^1 f'(t)g'(t)dt + \mu_2 \int_{-1}^1 f''(t)g''(t)dt + \dots$$

- 1st jet space \Rightarrow set $\mu_i = 0$ for $i > 1$.
Choose μ_1 experimentally to maximize reco rate.
Can be also done on-line.

[Golubitsky + SMW 2008, 2009]

Life in an Inner Product Space

- With the Legendre-Sobolev inner product we have
 - Low dimensional rep for curves ($10 + 10 + 1$)
 - Compact rep of samples ~ 160 bits [CCA WS 2009]
 - A useful notion of distance between curves
that is very fast to compute
 - $>99\%$ linear separability and convexity of classes
- Training data of 50,000 math char samples.
Use 75% for training 25% test. 10X cross validation.

Convexity of Classes

- Can separate N classes with $N(N-1)$ SVM planes.
- Each class is then (mostly) within its own convex polyhedral cell.
- Can classify either by
 - SVM majority voting + run-off elections (96%)
 - Distance to convex hull of k nearest neighbours (97.5%).
 - On-line computation.

Comparison of Candidate to Models

- Some classification methods compute the distance between the input curve and models.

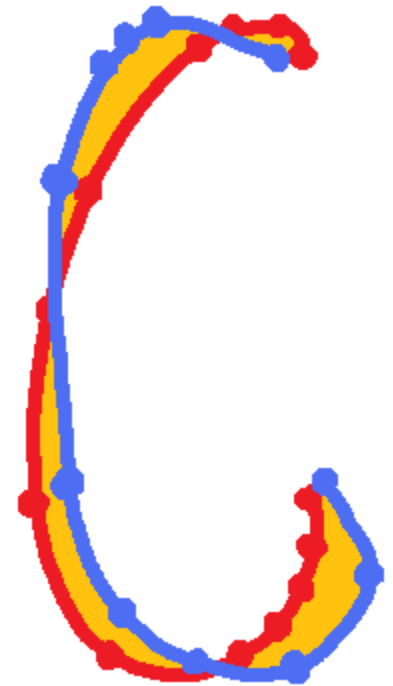
E.g. Elastic matching takes time up to quadratic in the number of sample points and linear in the number of models.

- Many tricks and **heuristics** to improve on this.

E.g. Limit amount of dynamic time warping, pre-classify based on features, ...

Distance Between Curves

- Approximate the variation between curves by some fn of distances between points.
- May be coordinate curves or curves in a jet space.
- Sequence alignment
- Interpolation (“resampling”)
- Why not just calculate the area?
- This is very fast in ortho series representation.



Distance Between Curves

$$\bar{x}(t) = x(t) + \xi(t) \quad \xi(t) = \sum_{i=0}^{\infty} \xi_i B_i(t)$$

$$\bar{y}(t) = y(t) + \eta(t) \quad \eta(t) = \sum_{i=0}^{\infty} \eta_i B_i(t)$$

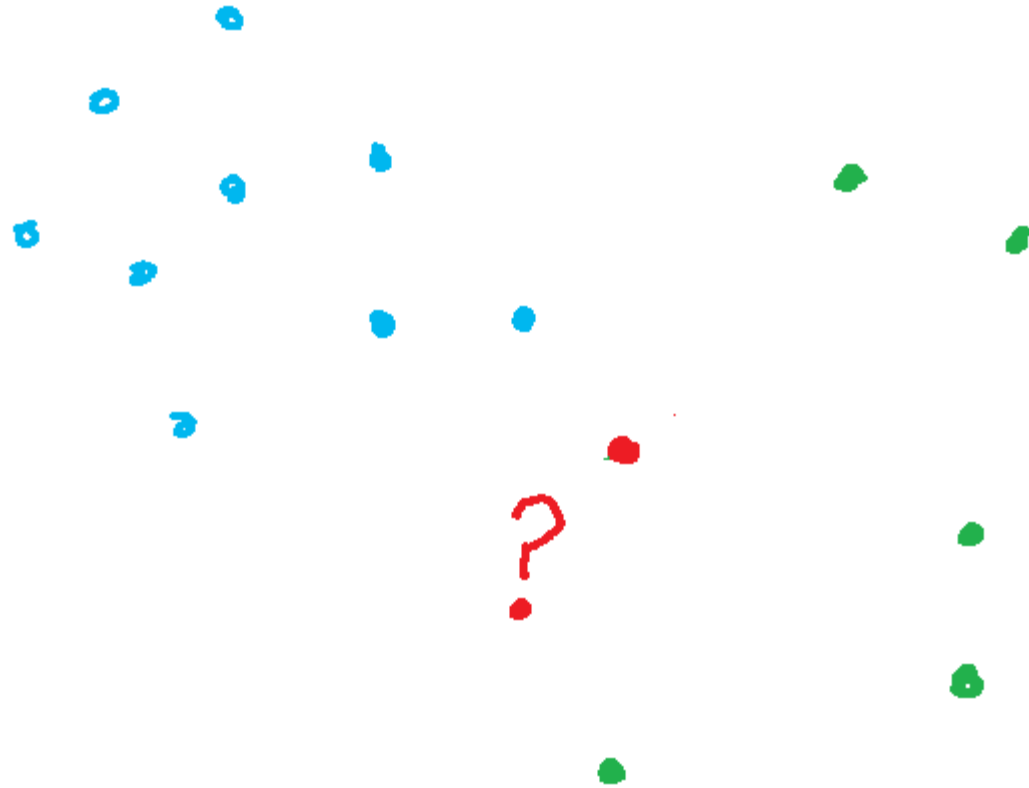
B_i orthonormal on $[a, b]$ with $w(t) = 1$.

$$\begin{aligned} \rho^2(C, \bar{C}) &= \int_a^b \left[(x(t) - \bar{x}(t))^2 + (y(t) - \bar{y}(t))^2 \right] dt \\ &= \int_a^b \left[\xi(t)^2 + \eta(t)^2 \right] dt \\ &\approx \int_a^b \left[\sum_{i=0}^d \xi_i^2 B_i^2(t) + \text{cross terms} + \sum_{i=0}^d \eta_i^2 B_i^2(t) + \text{cross terms} \right] dt \\ &= \sum_{i=0}^d \xi_i^2 + \sum_{i=0}^d \eta_i^2 \end{aligned}$$

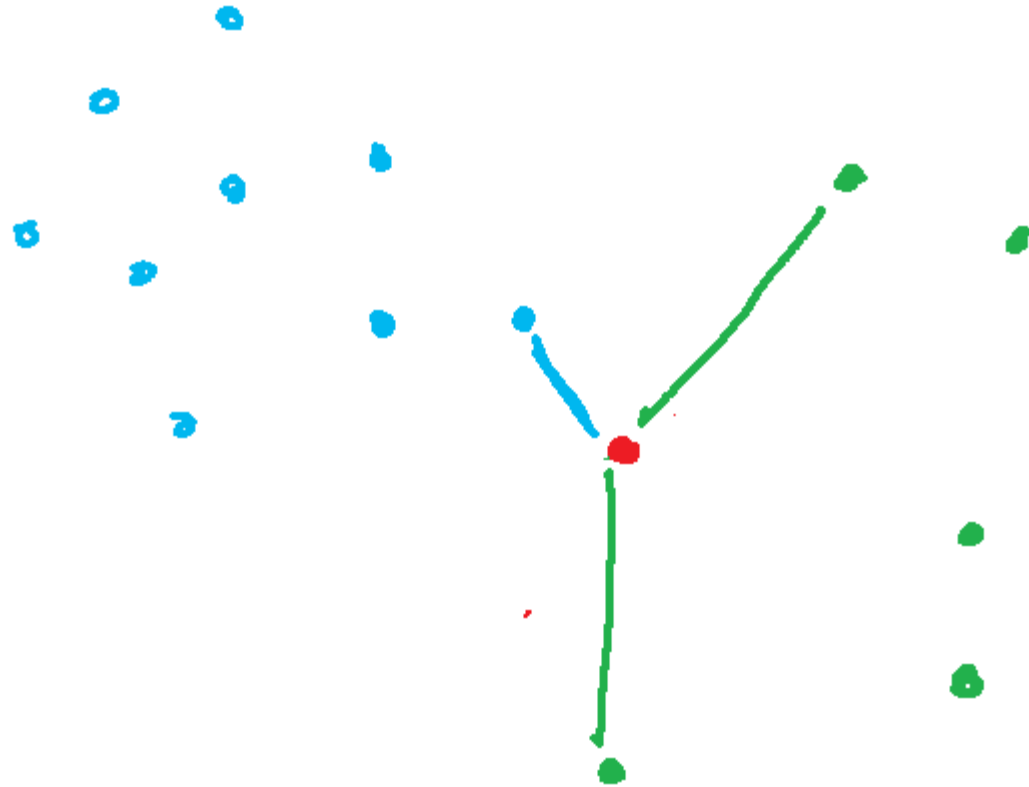
Comparison of Candidate to Models

- Use Euclidean distance in the coefficient space.
- *Just as accurate* as elastic matching.
- *Much less expensive.*
- Linear in d , the degree of the approximation.
< $3d$ machine instructions (30ns) vs several thousand!
- Can trace through SVM-induced cells incrementally.
- Normed space for characters gives other advantages.

Distance-Based Classification

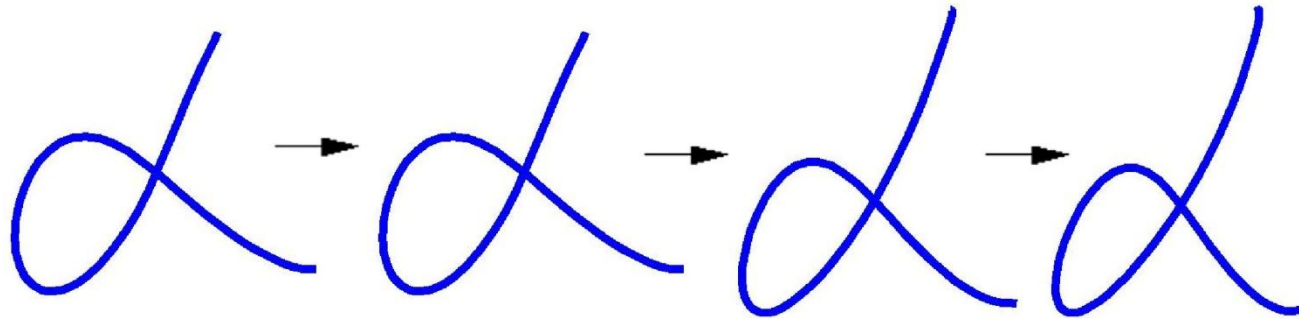


Distance-Based Classification



Geometry

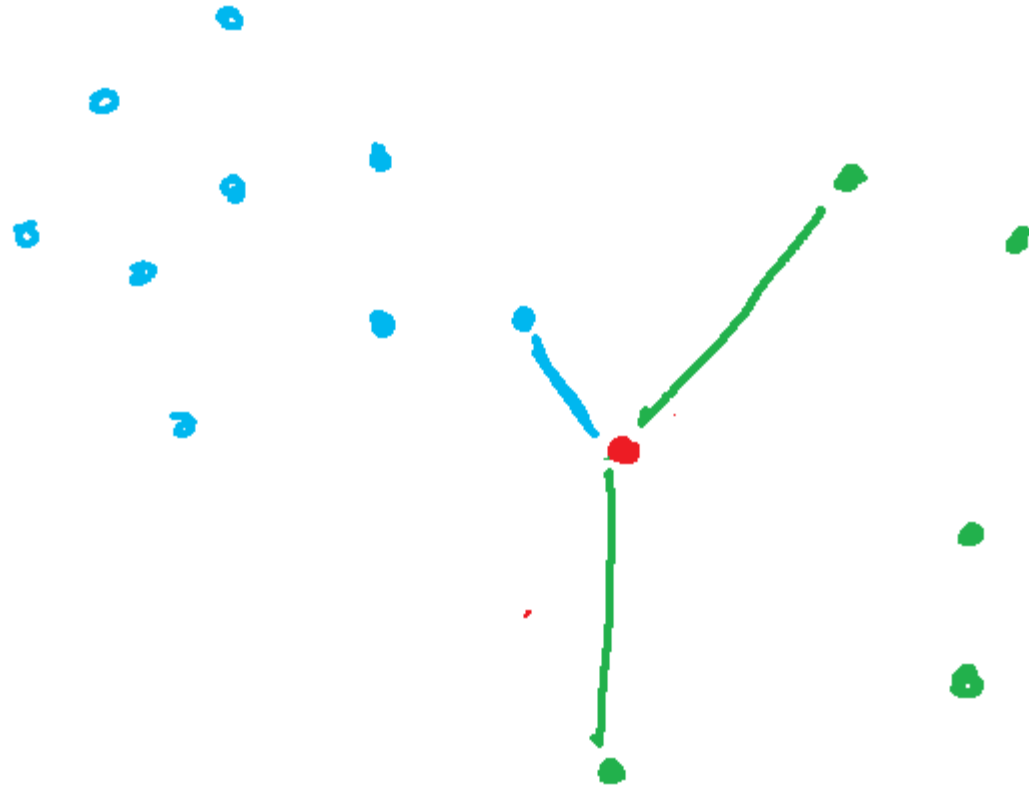
- Linear homotopies within a class



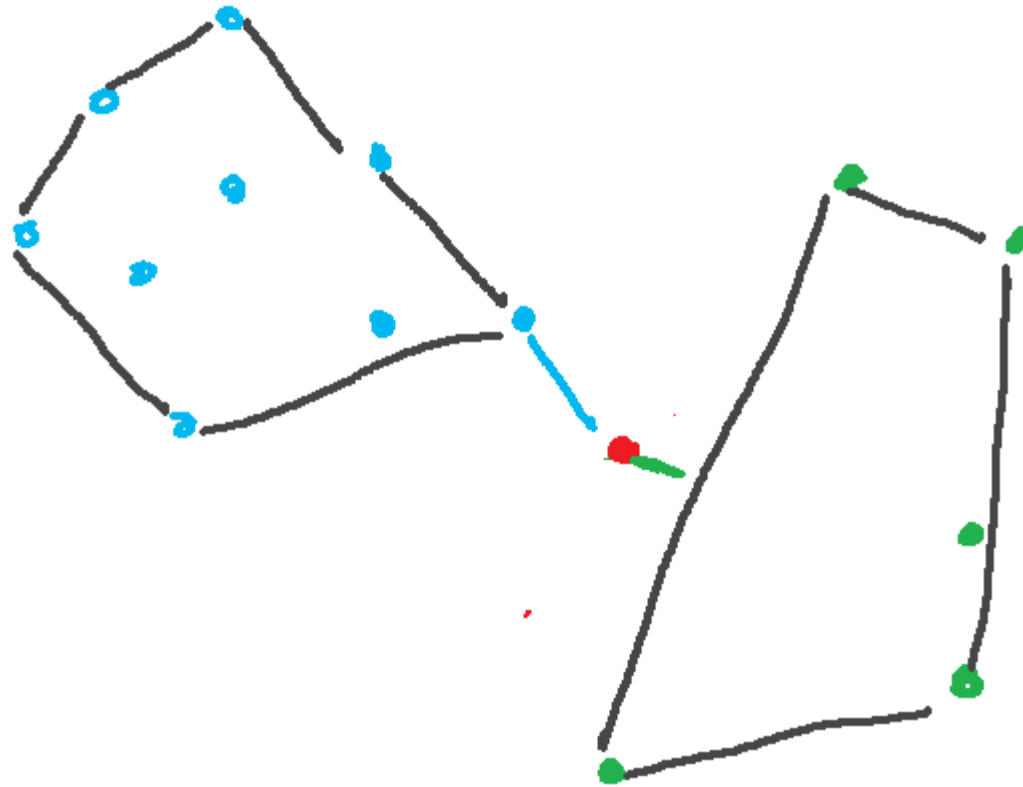
$$C = (1-t)A + tB$$

- Can compute distance of a sample to this line
- Convex hull of a set of models
- SVM separating planes

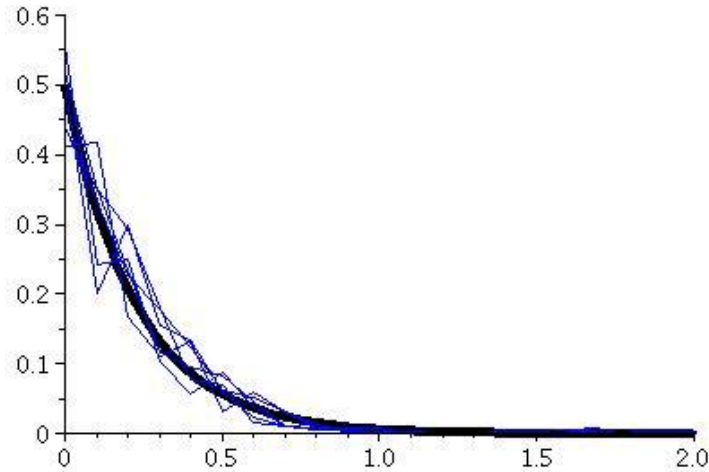
Distance-Based Classification



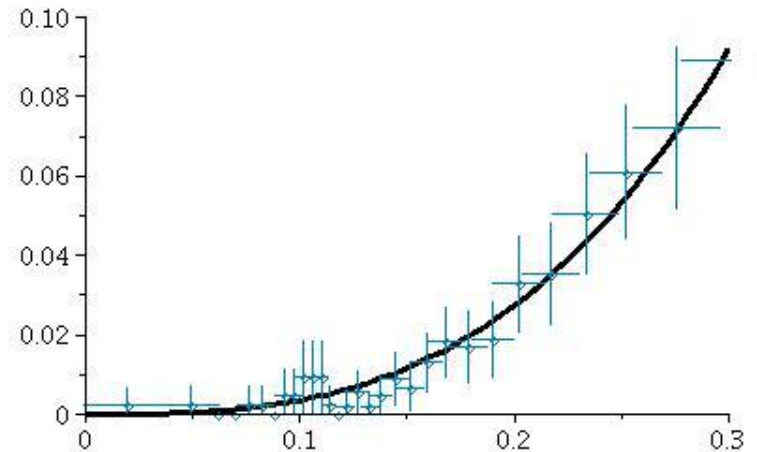
Distance-Based Classification



Error Rates as Fn of Distance



SVM



Convex Hull

- Error rate as fn of distance gives confidence measure for classifiers [MKM – Golubitsky + SMW]

Recognition Summary

- Database of samples \Rightarrow set of LS points
- Character to recognize \Rightarrow
 - Integrate moments as being written
 - Lin. trans. to obtain one point in LS space
 - Classify by distance to convex hull of k-NN.

Writer-Dependency

- Users do have writing styles
- Some symbols will be written idiosyncratically (*i.e.* wrong)
- => Multi-writer point set has some classes that are too big (in LS space) some classes that are too small.
- => Allow users to add + remove points.

Where to keep the user profile?

- Today people use multiple devices.
 - Laptop, office computer, home computer, telephone, smart white board
- Cloud-based storage

www.inkml.org

- User accounts
- Individual ink profile server
- Ink-based apps (e.g. our Skype ink chat) will retrieve profile **and save updates.**

The Quid Pro Quo

- *For the user:*
Applications will get very good at recognizing individual's handwriting.
- *For us:*
We get lots of data.

Next Directions

- More applications using the user profiles.
- Use new data:
 - Identify common writing styles
 - Identify correlations among symbol variants
 - Better writer *independent* math recognition.

Conclusions

- Two senses of “writing on clouds”:
 - Clouds of data points; reco by finding nearest.
 - Cloud storage of user profiles
- LS clouds give compact classifiers
- Suitable for storage, personalization, transmission
- Painless personalization
- A mutually useful proposition