[Return to Main](#)

[Objectives](#)

Introduction:
- [Components](#)
- [Temporal Information](#)
- [Z-Transform](#)
- [Curve-Fitting](#)
- [Alternatives](#)

Finite Differences:
- [First-Order Difference](#)
- [Frequency Response](#)

Regression:
- [Mean-Square Error](#)
- [Central Difference](#)

On-Line Resources:
- [SRSTW: Derivatives](#)
- [Polynomial Fitting](#)
- [Finite Differences](#)
- [STRUT: Post-Processing](#)
- [Regression Theory](#)
- [Regression Tutorial](#)
- [Regression Applet](#)
- [Autofit](#)

LECTURE 18: DIFFERENTIATION OF FEATURES

- Objectives:
  - Introduce the concept of a derivative
  - Appreciate the computational issues
  - Derivatives based on finite differences
  - Derivatives based on linear regression

Three important references for this material are:

- F.K. Soong and A.E. Rosenberg, "On the Use of Instantaneous and Transitional Spectral Information in Speaker Recognition," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Tokyo, Japan, pp. 877-880, April 1986.
- J.G. Proakis and D.G. Manolakis, *Digital Signal Processing (Third Edition)*, Prentice-Hall, Upper Saddle River, New Jersey, USA, 1996.
- A.J. Hayter, *Probability and Statistics For Engineers and Scientists*, International Thomson Publishing, Cincinnati, Ohio, USA, 1996.

The course textbook contains references to the seminal papers in this area as well.

ECE 8463: FUNDAMENTALS OF SPEECH RECOGNITION

Professor Joseph Picone
Department of Electrical and Computer Engineering
Mississippi State University

email: picone@isip.msstate.edu
phone/fax: 601-325-3149; office: 413 Simrall
URL: http://www.isip.msstate.edu/resources/courses/ece_8463

Modern speech understanding systems merge interdisciplinary technologies from Signal Processing, Pattern Recognition, Natural Language, and Linguistics into a unified statistical framework. These systems, which have applications in a wide range of signal processing problems, represent a revolution in Digital Signal Processing (DSP). Once a field dominated by vector-oriented processors and linear algebra-based mathematics, the current generation of DSP-based systems rely on sophisticated statistical models implemented using a complex software paradigm. Such systems are now capable of understanding continuous speech input for vocabularies of hundreds of thousands of words in operational environments.

In this course, we will explore the core components of modern statistically-based speech recognition systems. We will view speech recognition problem in terms of three tasks: signal modeling, network searching, and language understanding. We will conclude our discussion with an overview of state-of-the-art systems, and a review of available resources to support further research and technology development.

Tar files containing a compilation of all the notes are available. However, these files are large and will require a substantial amount of time to download. A tar file of the html version of the notes is available here. These were generated using wget:

```
wget -np -k -m http://www.isip.msstate.edu/publications/courses/ece_8463/lectures/current
```

A pdf file containing the entire set of lecture notes is available here. These were generated using Adobe Acrobat.

Questions or comments about the material presented here can be directed to help@isip.msstate.edu.

LECTURE 18: DIFFERENTIATION OF FEATURES

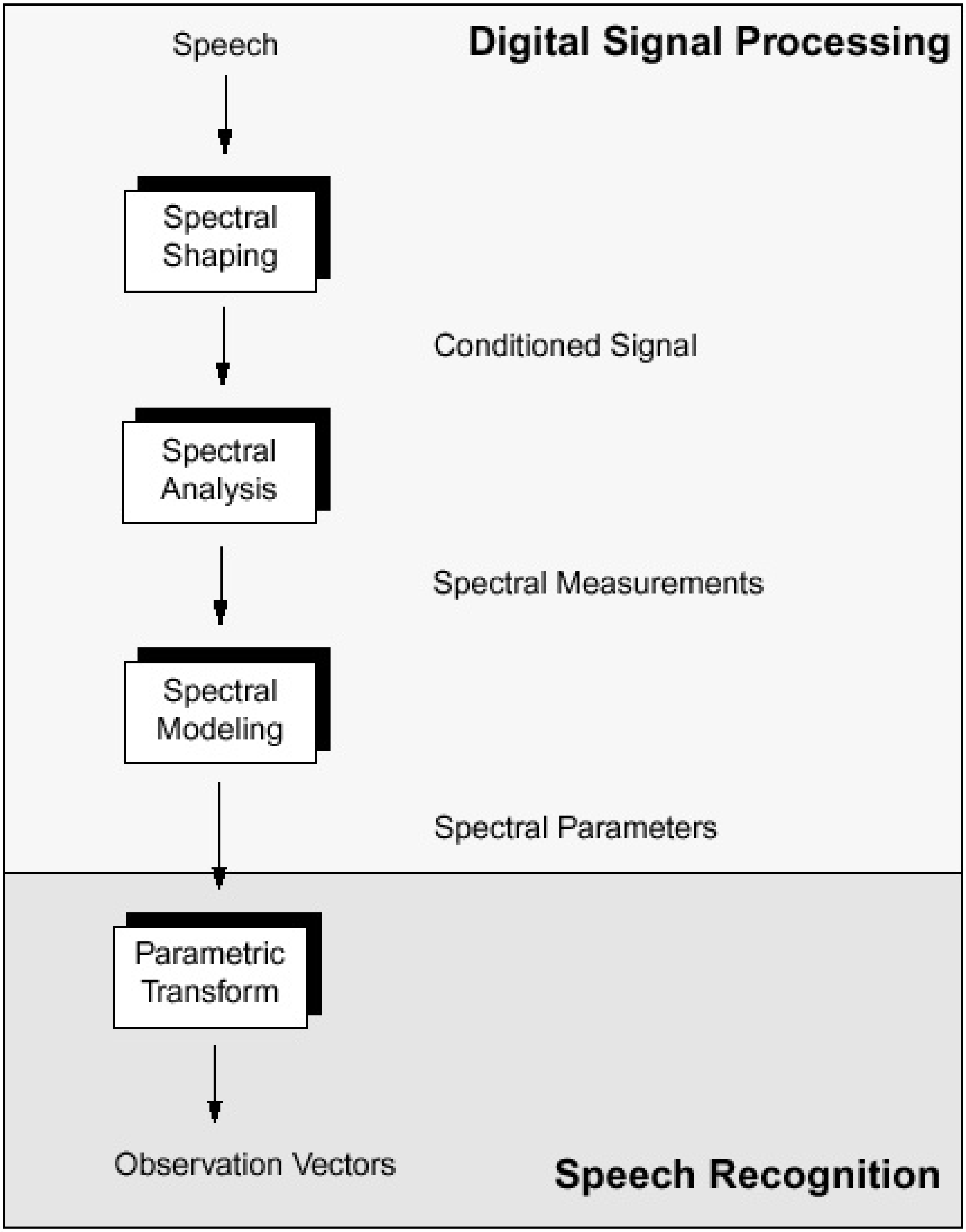
- Objectives:
    - Introduce the concept of a derivative
    - Appreciate the computational issues
    - Derivatives based on finite differences
    - Derivatives based on linear regression

Three important references for this material are:

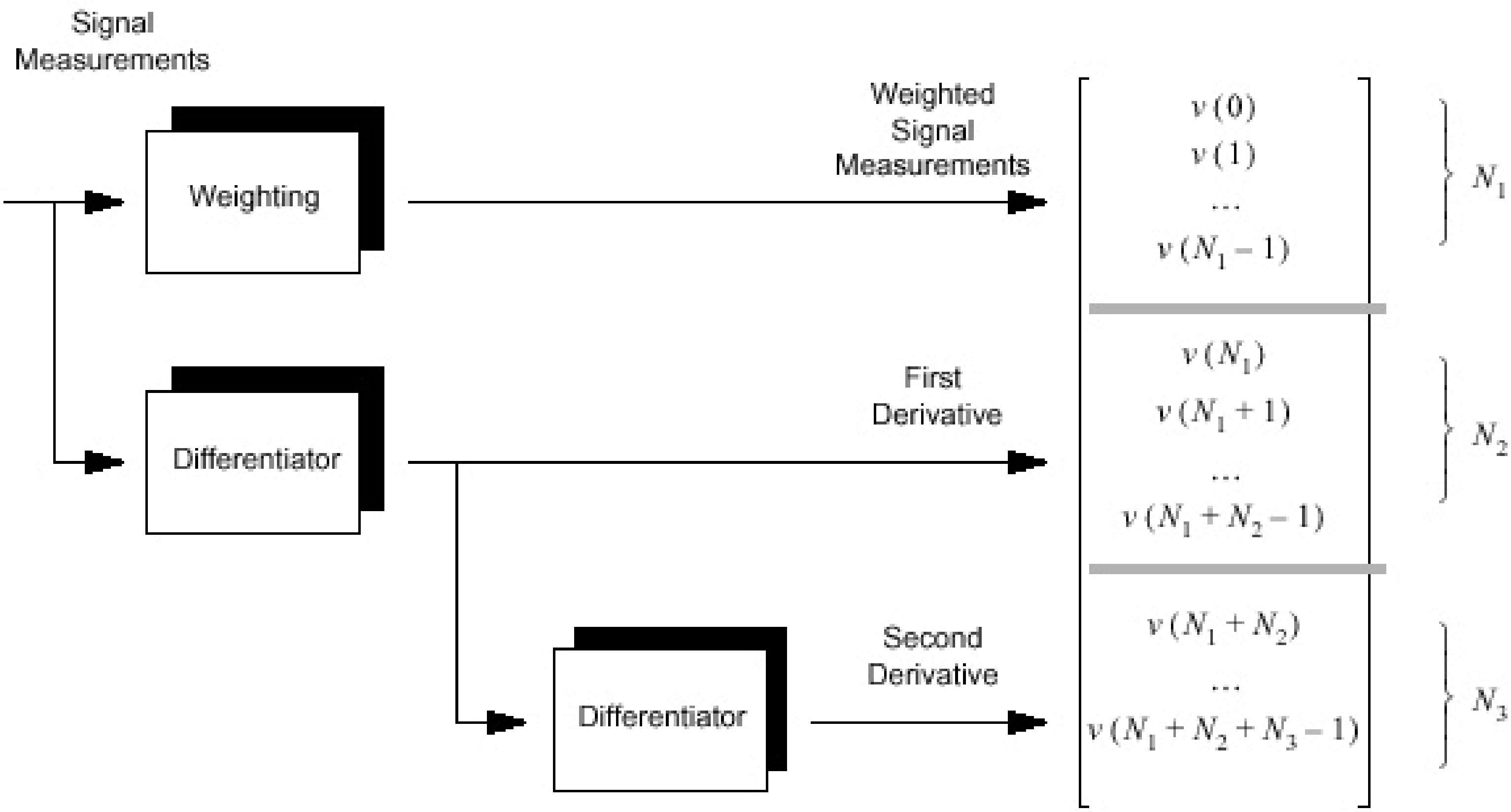
- F.K. Soong and A.E. Rosenberg, "On the Use of Instantaneous and Transitional Spectral Information in Speaker Recognition," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Tokyo, Japan, pp. 877-880, April 1986.
- J.G. Proakis and D.G. Manolakis, *Digital Signal Processing (Third Edition)*, Prentice-Hall, Upper Saddle River, New Jersey, USA, 1996.
- A.J. Hayter, *Probability and Statistics For Engineers and Scientists*, International Thomson Publishing, Cincinnati, Ohio, USA, 1996.

The course textbook contains references to the seminal papers in this area as well.

SIGNAL PROCESSING COMPONENTS
IN SPEECH RECOGNITION



ADDING TEMPORAL INFORMATION: DERIVATIVES



- Temporal derivatives of the spectrum are commonly approximated by differentiating cepstral features using a linear regression.

ADDING TEMPORAL INFORMATION: DERIVATIVES

- We would like to add information about the change in the spectrum to our feature vector to improve our ability to distinguish between stationary sounds (vowels) and nonstationary sounds (consonants).

- Recall the definition of differentiation in the time domain:

$$\frac{d}{dt}x(t) \Leftrightarrow j\omega X(\omega)$$

- Differentiation is an inherently noisy process since it amplifies high frequencies. Hence, we must be careful how we compute this. In practice, we use low-pass filtered derivatives (the derivative of a low-pass filtered version of the signal).

- What we really want to measure is the time derivative of the spectrum:

$$\frac{\partial}{\partial t}X(\omega, t) \;=\; ???$$

  But derivatives of continuous time signals are difficult to compute for discrete-time signals.

- Recall the definition of a derivative:

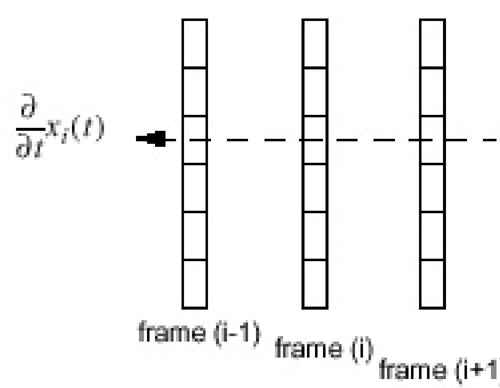
$$\frac{d}{dt}x(t) \;=\; \lim_{T \to 0} \frac{x(t) - x(t-T)}{T}$$

  This can be viewed as a digital filter:

$$y(n) = \frac{x(n) - x(n-1)}{T} \Leftrightarrow H(z) \;=\; \left(\frac{1}{T}\right)1 - z^{-1}$$

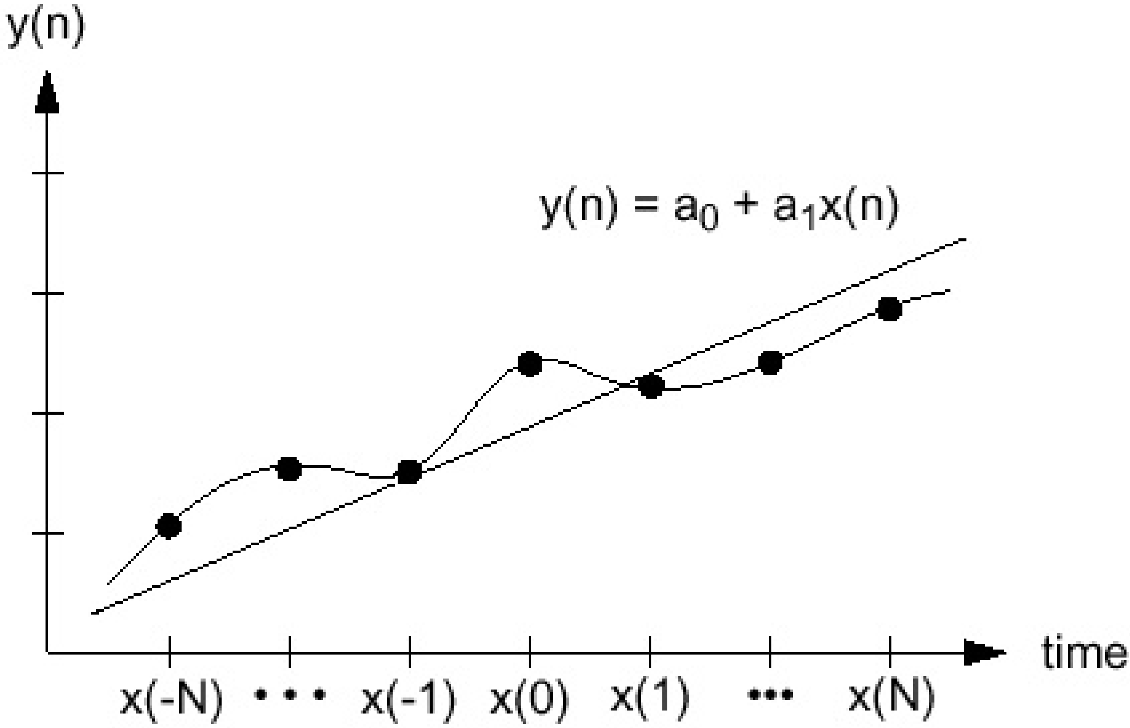
  Later we will explore the frequency response of this filter.

- In practice, we compute temporal derivatives of feature vectors by differentiating each element as a function of time. Since feature vectors measure the spectrum, this gives us a realistic measure of spectral change. These derivatives, called delta parameters, are concatenated with the absolute measurements to form an extended feature vector that contains absolute and rate of change information.



GRAPHICAL INTERPRETATION: CURVE FITTING

- What we seek is the value of the slope, not the differentiated signal. This can be directly estimated using the principle of linear regression.
- We can cast this estimation problem as a curve-fitting problem with some special constraints that result from the signal processing nature of the problem.
- Consider the estimation problem shown below:



- The slope of a signal can be estimated directly using a linear regression approach. More precisely, we are using a least mean square error parameter estimation approach to finding the equation of a line that best approximates the signal.
- Note that the slope of the line is represented by the parameter a_1 in the equation shown in the figure above. **This is the parameter of interest in this analysis**.

ALTERNATE WAYS TO COMPUTE A DERIVATIVE

In conventional digital signal processing (DSP) textbooks, derivatives, $\partial^k y(t) / \partial t$, are estimated several ways:

- simple backward difference (first-order — p=1):

$$y(nT) = \frac{x(nT) - x((n-1)T))}{T} \tag{2}$$

- central difference (first-order — p=1):

$$y(nT) = \frac{x((n+1)T) - x((n-1)T)}{2T} \tag{3}$$

- digital filters:

$$y(n) = a_1 y(n-1) + a_2 y(n-2) + \ldots + b_0 x(n) + b_1 x(n-1) + \ldots \tag{4}$$

- higher-order approximations (Taylor Series, Splines):

$$y(nT) = a_0 + a_1 \frac{\partial}{\partial t} y(nT) + \frac{\partial^2}{\partial t} y(nT) + \ldots \tag{5}$$

- What we must keep clear here is the difference between the order of the derivative (**k**), the order of the approximation (**p**), and the length of the filter or difference equation used to compute the approximation (**N**).

PROPERTIES OF A FIRST-ORDER DIFFERENCE

We can compute the frequency response of a first-order difference:

$$y(n) = \frac{1}{2}[x(n) - x(n-1)]$$

$$h(n) = \left\{-\boldsymbol{\frac{1}{2}}, \frac{1}{2}\right\}$$

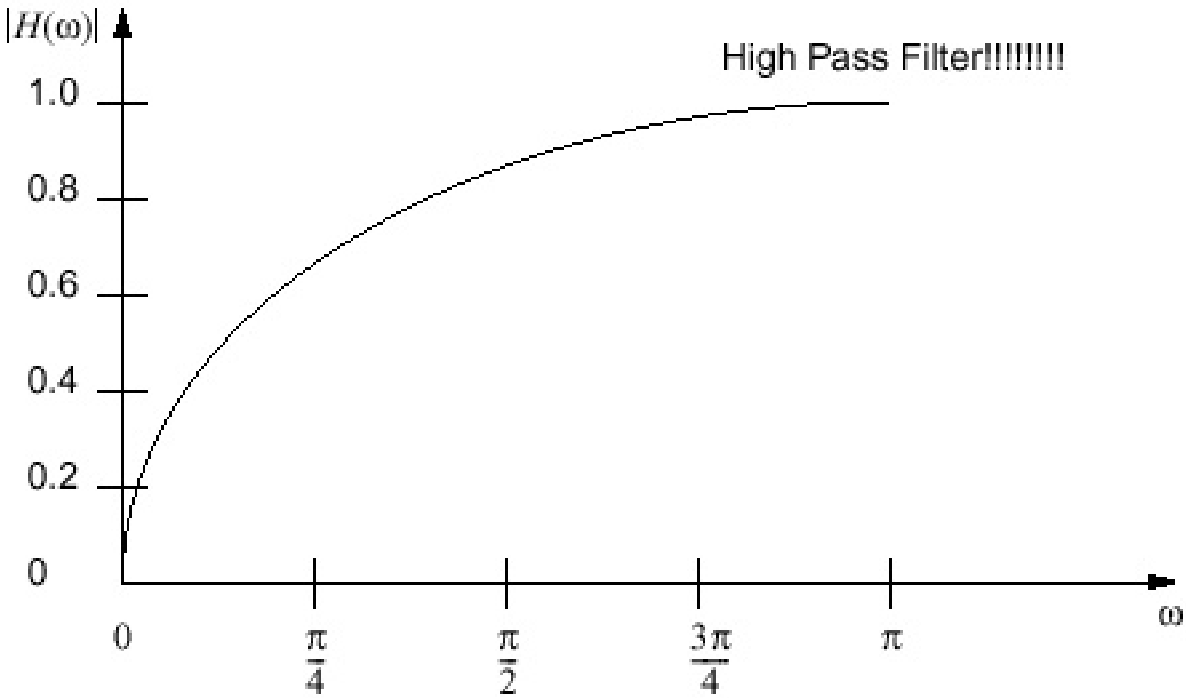
$$H(\omega) = \frac{1}{2}(1 - e^{-j\omega})$$

$$\begin{aligned}
|H(\omega)| &= \frac{1}{2}|1 - \cos\omega + j\sin\omega| \\
&= \frac{1}{2}\sqrt{(1-\cos\omega)^2 + (\sin\omega)^2} \\
&= \frac{1}{2}\sqrt{(1 - 2\cos\omega + (\cos\omega)^2) + (\sin\omega)^2} \\
&= \frac{1}{2}\sqrt{2 - 2\cos\omega} \\
&= \frac{1}{\sqrt{2}}\sqrt{1 - \cos\omega}
\end{aligned}$$

FREQUENCY RESPONSE OF A FIRST-ORDER DIFFERENCE

A plot of the frequency response for this filter is shown below:

$$y(n) = \frac{1}{2}[x(n) - x(n-1)] \qquad H(\omega) = \frac{1}{2}(1 - e^{-j\omega})$$



- Because this filter acts as a high-pass filter, it has a tendency to amplify noise.

MEAN SQUARE ERROR DERIVATION

- In speech recognition, we prefer to use a statistical approach to estimating the derivative. Why?
- This technique uses a statistical method known as linear regression. In this approach, we choose the regression coefficients to minimize the mean squared error:

$$E = \sum_{n=-\infty}^{\infty} [y(n) - (a_0 + a_1 x(n))]^2$$

- The solution to this equation is well-known (in DSP literature, this is known as linear prediction), and is found by differentiating the error equation with respect to the regression coefficients, setting the derivative to zero, and solving for the regression coefficients. This results in the following equations:

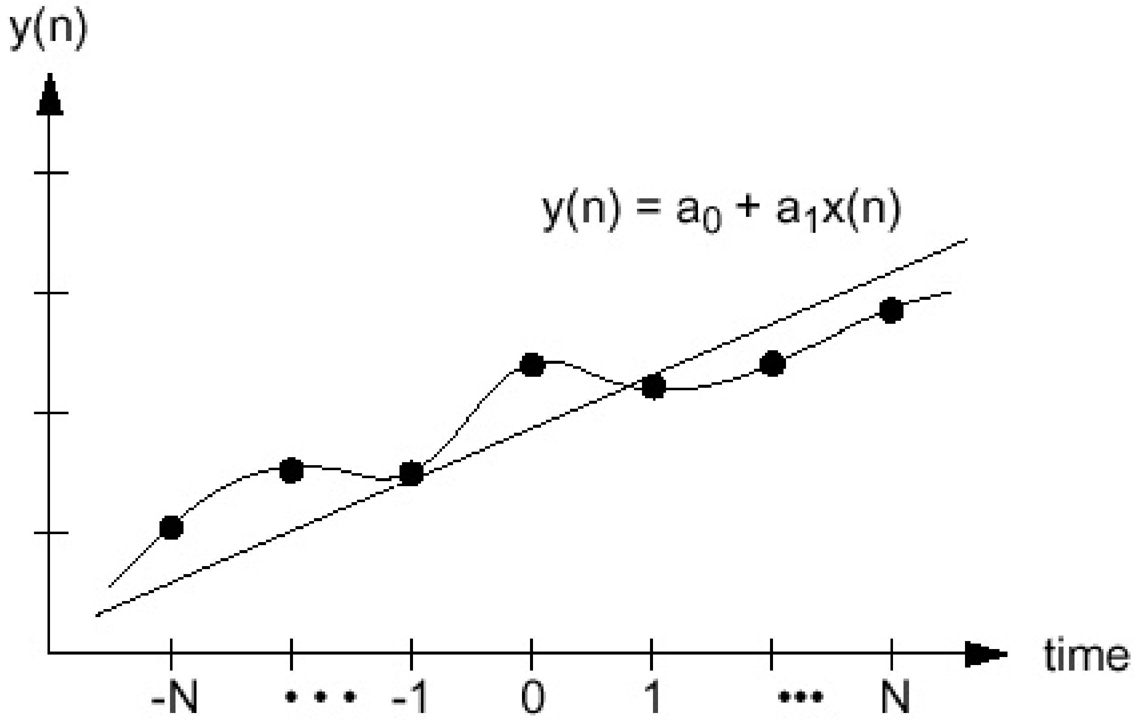
$$a_1 = \frac{n\sum x(n)y(n) - (\sum x(n))(\sum y(n))}{n\sum x^2(n) - (\sum x(n))^2}$$

$$a_0 = \left(\frac{1}{n}\right)\sum y(n) - a_1\left(\frac{1}{n}\right)\sum x(n)$$

- This equation is fairly general. Note that if the input data have an average value of zero, the resulting equations are even simpler.

LINEAR REGRESSION

- We can simpify the previous equation by imposing a central difference type formulation of the problem, as shown below:



The x-axis is relabeled in terms of equispaced sample indices, and centered about zero.

- This simplifies the calculation to:

$$a_1 = \frac{n\sum x(n)y(n) - (\sum x(n))(\sum y(n))}{n\sum x^2(n) - (\sum x(n))^2}$$

$$= \frac{n\sum x(n)y(n)}{n\sum x^2(n)}$$

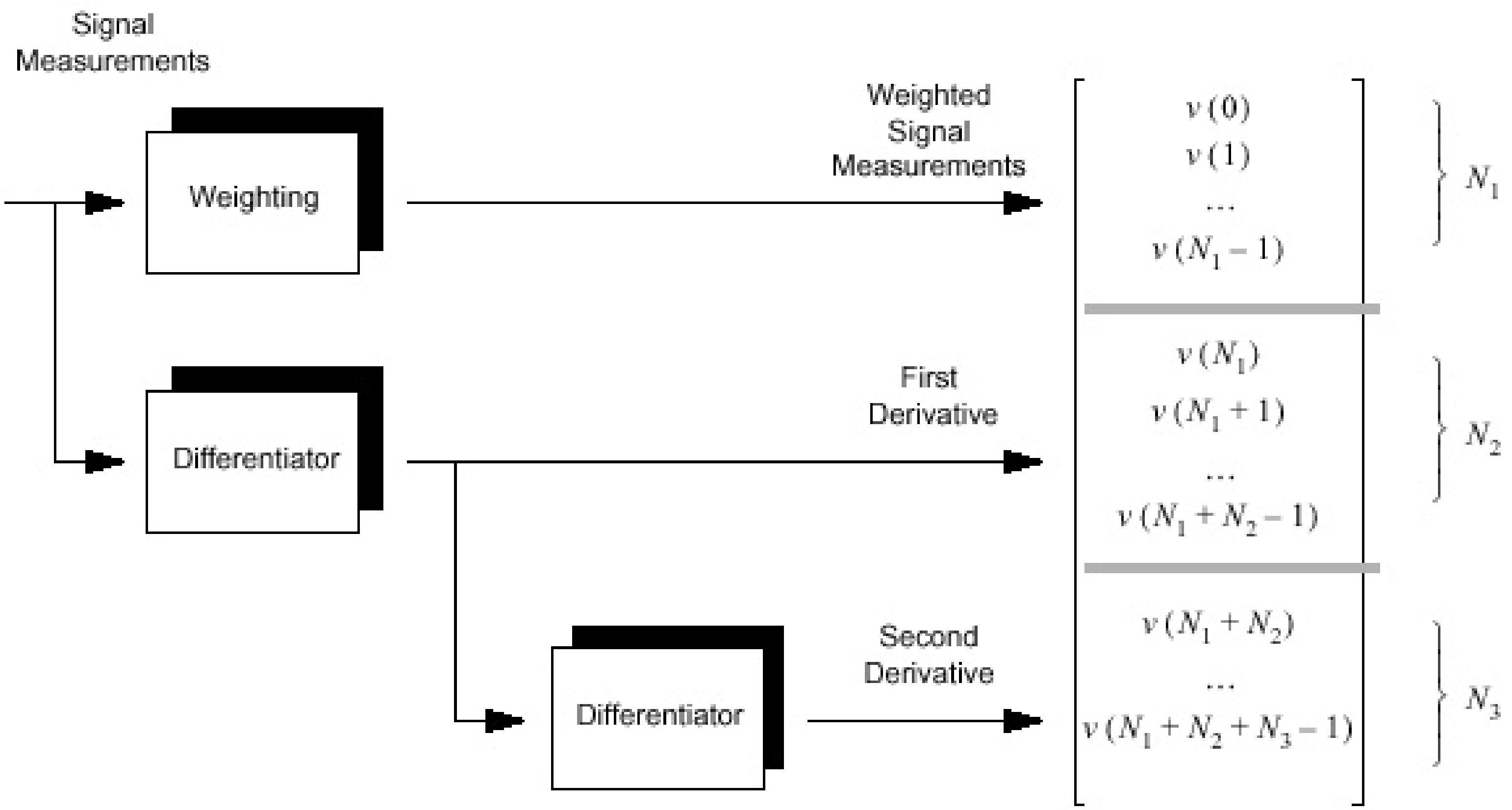
$$= \frac{\sum_{n=-N}^{N} ny(n)}{\sum_{n=-N}^{N} n^2}$$

- This equation is the form we desire, and is extremely efficient to compute. The denominator can be precomputed, and the integer multiplications are easily implemented even in fixed-point DSPs.

- Obviously, this approach can be extended to higher order derivatives. However, historically, second derivatives in speech recognition have been computed by applying two first-order derivatives in succession.

- Further, the order of regression used, N, is most commonly set to 2, which means a five-frame sequence of features is required to compute the first-order derivative.

ADDING TEMPORAL INFORMATION: DERIVATIVES



- Temporal derivatives of the spectrum are commonly approximated by differentiating cepstral features using a linear regression.

Lecture 19

Input Processing

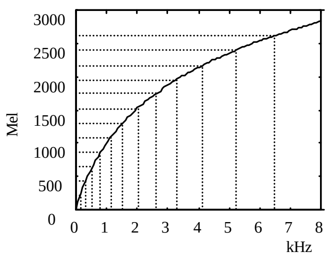
- Mel Frequency Scale
- Input Signal Preprocessing
  - Discrete Cosine Transform
  - Time Derivative estimates
  - Feature Decorrelation
  - Feature Vector length reduction
- Gaussian Mixtures
- Speech Recognition Summary

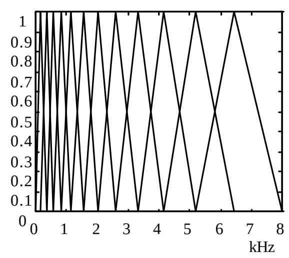
Feature Vector Requirements

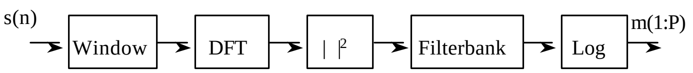
- When different people say the same phoneme, the feature vectors should have similar values.
- Different phonemes from the same or different speakers should give dissimilar values.
- For different examples of the same phoneme, the features should be independent and uncorrelated: this allows us to multiply their probabilities.
- For different examples of the same phoneme, each feature should preferably follow a probability distribution that is well described as a sum of gaussians.
- The features should not be affected by the amplitude of the speech signal otherwise recognition performance would vary with your distance from the microphone.

Mel Frequency Scale

- The feature vector must discriminate between speech sounds using as few components as possible to reduce computation.
- The human ear has better frequency resolution at low frequencies. The *mel* scale relates perceived pitch to frequency: linear at low f, logarithmic at high f:
  - $\text{mel}(f) = 2595 \log_{10}(1 + f / 700)$ where f is in Hz
  - Form a mel-spaced filterbank by setting the centre frequencies to equally spaced mel values.

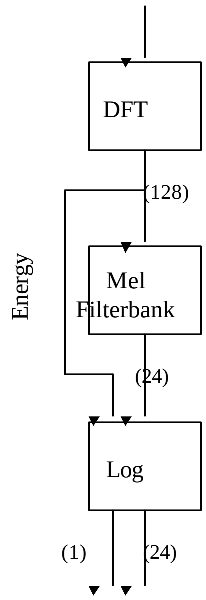






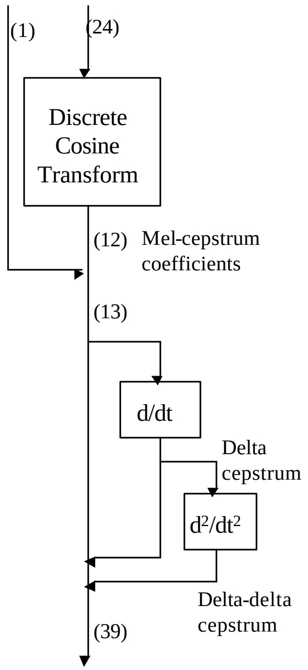
Preprocessing: Stage 1

- Divide signal into overlapping 25 ms segments at 10 ms intervals
- Apply Hamming window and take FFT
- Smooth the spectrum with a mel filterbank
  - mel filterbank concentrates data values in the more significant part of the spectrum
- Take the log of the mel spectrum
  - variations in signal level just cause a DC shift in the log spectrum
  - gaussian approximation is more nearly true for log spectrum than for the power spectrum directly



Preprocessing: Stage 2

- Discrete Cosine Transform (DCT)
  - reduces correlation between coefficients
  - compresses information into fewer low-order coefficients
  - output is the *mel-cepstrum*
  - DC component is ignored to make it independent of signal level
- First and Second time derivatives
  - provide additional information about how the spectrum is changing with time
- Result is a 39 element feature vector (or 38 if you drop the log energy).

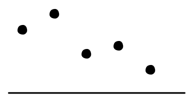


Discrete Cosine Transform

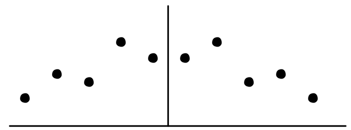
- The discrete cosine transform (DCT) of $m_1,\ldots,m_P$ is defined by

$$c_k = \sum_{p=1}^{P} m_p \cos(\pi k(p-\tfrac{1}{2})/P)$$

- The DCT of these points



is equal to the DFT of these points



with a phase shift to centre the time origin.

- Taking the DCT of the +ve frequency spectrum is essentially the same as taking the DFT of the symmetrical ±ve frequency spectrum.
- There are efficient algorithms for calculating the DCT

Polynomial Fitting

- To fit a polynomial to a set of points x_i, y_i for i=1, 2, …, N: $y_i = \sum_{k=0}^{P} a_k x_i^k$
- Error $E = \sum_{i=1}^{N} e_i^2$ where $e_i = y_i - \sum_{k=0}^{P} a_k x_i^k$
- Minimize E by differentiating w.r.t. a_m, $m=0{:}P$

$$\frac{\partial E}{\partial a_m} = \sum_{i=1}^{N} 2e_i \frac{\partial e_i}{\partial a_m} = -2\sum_{i=1}^{N} \left(y_i - \sum_{k=0}^{P} a_k x_i^k\right) x_i^m$$

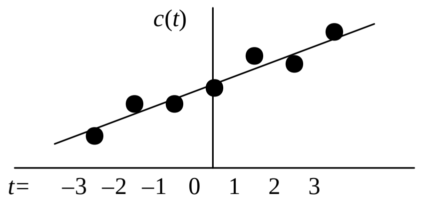
- Hence we get $P+1$ equations (same as LPC)

$$\sum_{k=0}^{P} a_k \left(\sum_{i=1}^{N} x_i^{k+m}\right) = \sum_{i=1}^{N} y_i x_i^m \quad \textit{for} \quad m = 0,1,\ldots,P$$

- In matrix form (each value of m gives one row):

$$\begin{pmatrix} \sum x^0 & \sum x^1 & \sum x^2 & \cdots \\ \sum x^1 & \sum x^2 & \sum x^3 & \\ \sum x^2 & \sum x^3 & \sum x^4 & \\ \vdots & & & \ddots \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \end{pmatrix} = \begin{pmatrix} \sum yx^0 \\ \sum yx^1 \\ \sum yx^2 \\ \vdots \end{pmatrix}$$

Cepstral Time-Derivatives



- Want to estimate dc/dt by fitting a line
  - Few points ⇒ noisy estimate
  - Many points ⇒ can't follow time variations
- Fit a 1st-order polynomial to $2T+1$ points:

$$\begin{pmatrix} \sum_{t=-T}^{T} t^0 & \sum_{t=-T}^{T} t^1 \\ \sum_{t=-T}^{T} t^1 & \sum_{t=-T}^{T} t^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \sum_{t=-T}^{T} c(t)t^0 \\ \sum_{t=-T}^{T} c(t)t^1 \end{pmatrix}$$

- this simplifies to

$$\begin{pmatrix} 2T+1 & 0 \\ 0 & \sum_{t=-T}^{T} t^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \sum_{t=-T}^{T} c(t) \\ \sum_{t=-T}^{T} tc(t) \end{pmatrix} \Rightarrow \quad a_1 = \frac{\sum_{t=-T}^{T} tc(t)}{\sum_{t=-T}^{T} t^2}$$

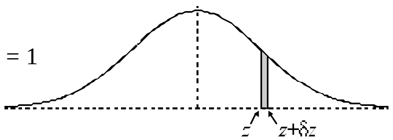
- Typically $T=5$ for 1st derivative and $T=1$ for 2nd

Multivariate Gaussian Distributions

- Z is a random variable with a standard Gaussian (or Normal) probability density func:

$$\mathrm{pr}(Z \in [z, z+\delta z]) = (2\pi)^{-½} \exp\left(-½ z^2\right)\delta z$$

- Mean: $E(Z) = 0$
  Variance: $E(Z^2) = 1$



- A linear sum of multiples of Gaussian random variables gives another Gaussian random variable. This property is unique to Gaussians.
- If we have a column random vector **Z** with P elements each of which is an *independent* standard Gaussian random variable then

$$\mathrm{pr}(\mathbf{Z} \in [\mathbf{z}, \mathbf{z}+d\mathbf{z}]) = \prod_{i=1}^{P} (2\pi)^{-½} \exp\left(-½ z_i^2\right) dz_i$$

$$= (2\pi)^{-P/2} \exp\left(-½ \sum_{i=1}^{P} z_i^2\right) d\mathbf{z} = (2\pi)^{-P/2} \exp\left(-½ \mathbf{z}^T\mathbf{z}\right) d\mathbf{z}$$

- Note too that because the z_i are independent.

$E(z_i z_j) = 0$ whenever $i \neq j \Rightarrow E(\mathbf{z}\mathbf{z}^T) = \mathbf{I}$

Correlated Gaussian Distributions

- Now suppose that $\mathbf{x} = \mathbf{A}\,\mathbf{z}$ where $\mathbf{A}$ is an non-singular matrix, then $d\mathbf{x} = |\mathbf{A}|\,d\mathbf{z}$ and $\mathbf{z} = \mathbf{A}^{-1}\mathbf{x}$. Note that **X** is gaussian and $E(\mathbf{x})$ is 0.
- The covariance matrix of $\mathbf{x}$ is $\mathbf{C}=E(\mathbf{x}\mathbf{x}^T)$ and is symmetric and positive definite

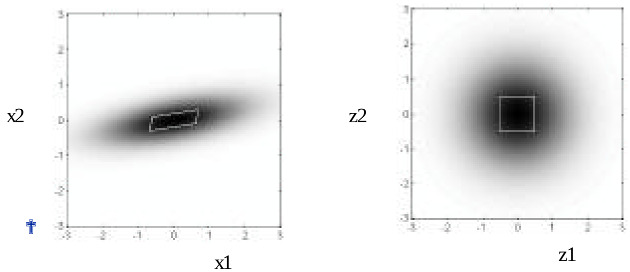
$$\mathbf{C} = E\left(\mathbf{x}\mathbf{x}^T\right) = E\left(\mathbf{A}\mathbf{z}\mathbf{z}^T\mathbf{A}^T\right) = \mathbf{A}E(\mathbf{z}\mathbf{z}^T)\mathbf{A}^T = \mathbf{A}\mathbf{A}^T$$

- We can work out the pdf of $\mathbf{x}$

$$\mathrm{pr}(\mathbf{X} \in [\mathbf{x}, \mathbf{x}+d\mathbf{x}])/d\mathbf{x} = (2\pi)^{-P/2} \exp\left(-½\left(\mathbf{A}^{-1}\mathbf{x}\right)^T\left(\mathbf{A}^{-1}\mathbf{x}\right)\right)|\mathbf{A}|^{-1}$$

$$= (2\pi)^{-P/2}\,|\mathbf{A}|^{-1} \exp\left(-½\mathbf{x}^T\mathbf{A}^{-T}\mathbf{A}^{-1}\mathbf{x}\right)$$

$$= (2\pi)^{-P/2}\,|\mathbf{C}|^{-½} \exp\left(-½\mathbf{x}^T\mathbf{C}^{-1}\mathbf{x}\right)$$



$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1.3 & 0.1 \\ 0.2 & 0.4 \end{pmatrix}\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \Rightarrow \mathbf{C} = \begin{pmatrix} 1.7 & 0.3 \\ 0.3 & 0.2 \end{pmatrix}$$ and $|\mathbf{C}| = |\mathbf{A}|^2 = 0.25$

Computational Costs

- The log prob density of correlated gaussians:

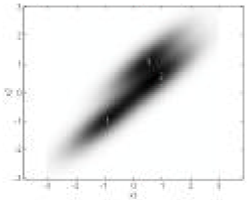
$$\log(\mathrm{pd}(x)) = \log\left((2\pi)^{-\frac{1}{2}P}|\mathbf{C}|^{-\frac{1}{2}}\exp\left(-\tfrac{1}{2}\mathbf{x}^T\mathbf{C}^{-1}\mathbf{x}\right)\right)$$
$$= -\tfrac{1}{2}\left(P\log(2\pi) + \log(|C|) + \mathbf{x}^T\mathbf{C}^{-1}\mathbf{x}\right)$$

- The first two terms are independent of $\mathbf{x}$ and can be precalculated for each state.
- For F features, the final term involves F^2+F multiplications and F^2-1 additions: $39^2 = 1521$
- If the features are (or are assumed to be) independent, $\mathbf{C}$ is diagonal and we need $2F$ multiplications and $F-1$ additions
- Probability calculations consume most of the computation in a recogniser: almost all recognisers assume feature independence
  - DCT on log spectrum improves independence
  - We can do even better by applying a linear transformation to the feature vector.

Feature Decorrelation

- We can apply a linear transformation to our feature vectors, $\mathbf{x}$, to reduce correlations.

$\mathbf{x}$:



$\mathbf{W}_s$ is the covariance matrix of state s.

$\mathbf{W}$ is the average of the $\mathbf{W}_s$: the average within-state covariance matrix.

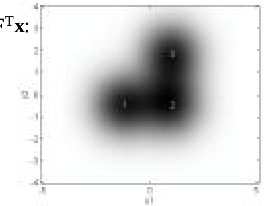
- If we multiply the feature vectors by a matrix $\mathbf{F}^T$, $\mathbf{y}=\mathbf{F}^T\mathbf{x}$, then the covariance matrix of $\mathbf{y}$ within state s is given by:

$$E\left((\mathbf{y}-\bar{\mathbf{y}}_s)(\mathbf{y}-\bar{\mathbf{y}}_s)^T\right) = E\left(\mathbf{F}^T(\mathbf{x}-\bar{\mathbf{x}}_s)(\mathbf{x}-\bar{\mathbf{x}}_s)^T\mathbf{F}\right) = \mathbf{F}^T\mathbf{W}_s\mathbf{F}$$

where $\bar{\mathbf{x}}_s$ is the mean value of $\mathbf{x}$ in state s.

- We transform our data with an $\mathbf{F}^T$ satisfying $\mathbf{F}^T\mathbf{W}\mathbf{F} = \mathbf{I}$.

$\mathbf{y}=\mathbf{F}^T\mathbf{x}$:



Eigenvectors

- *d* is an eigenvalue of **W** and **y** is an associated eigenvector if

$$\mathbf{Wy}=\mathbf{y}d$$

- Since **W** is symmetric and positive definite, we can find *F* orthonormal eigenvectors and make them the columns of a matrix:

$$\mathbf{WY}=\mathbf{YD}$$

where **D** is a diagonal matrix of eigenvalues

- The orthonormality of the eigenvectors means that

$$\mathbf{Y}^T\mathbf{Y}=\mathbf{I}$$

- Now we define

$$\mathbf{F}=\mathbf{YD}^{-½}$$

- This gives

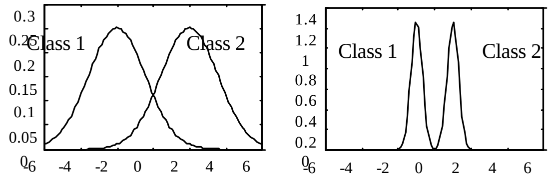
$$\mathbf{F}^T\mathbf{WF}=\mathbf{D}^{-½}\mathbf{Y}^T\mathbf{WYD}^{-½}=\mathbf{D}^{-½}\mathbf{Y}^T\mathbf{YDD}^{-½}=\mathbf{I}$$

- In MATLAB:

```
[Y,D] = eig[W];
F = Y * sqrt(inv(D));
```


Class Discrimination

- We would like to make our feature vector a short as possible while preserving its ability to discriminate.



- The graphs show two possible distributions of a parameter for two different speech sounds (or *classes*).

- For a single parameter, Fisher's *F* Ratio is a measure of discriminability (the bigger the better): $F\ ?\ \dfrac{\text{Variance of the class means}}{\text{Average variance within a class}}$

- For a parameter vector, this generalises to:

$$F\ ?\ \mathrm{trace}(\mathbf{W}^{?1}\mathbf{B})$$

where **W** and **B** are "average within-class" and "between-class" covariance matrices

Dimensionality Reduction

- We define **B** to be the between-state covariance matrix:

$$\mathbf{B} = \frac{1}{S}\sum_{s=1}^{S}\left(\bar{\mathbf{y}}_s - \bar{\bar{\mathbf{y}}}\right)\left(\bar{\mathbf{y}}_s - \bar{\bar{\mathbf{y}}}\right)^T$$

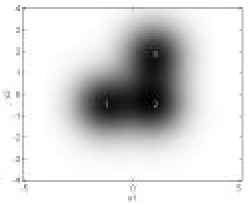
- As before we can find the eigenvalues of B

$$\mathbf{BG}=\mathbf{GL}$$

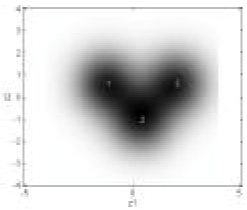
- where **G** is orthogonal and **L** diagonal.
- Set $\mathbf{z}=\mathbf{G}^T\mathbf{y}=\mathbf{G}^T\mathbf{F}^T\mathbf{x}$
- The between-state covariance matrix is now

$$\mathbf{G}^T\mathbf{BG}=\mathbf{L}$$

- We can discard any elements of **z** for which the corresponding element of **L** is very small. Gives reduced feature set with equal (or even better) discrimination.



$\mathbf{y}=\mathbf{F}^T\mathbf{x}$:



$\mathbf{z}=\mathbf{G}^T\mathbf{y} = \mathbf{G}^T\mathbf{F}^T\mathbf{x}$:

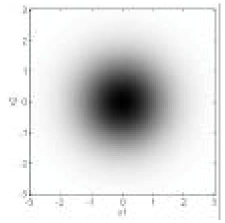
Gaussian Mixtures

- For large vocabularies, independent gaussian model is too simple. Use instead a mixture of gaussians:

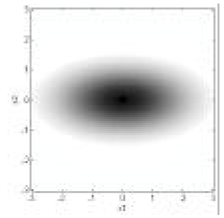
$$\mathrm{pd}(\mathbf{x}) = \sum_{i=1}^{K} w_i N(\mathbf{m}_i,\mathbf{C}_i) \quad \text{with} \quad \sum_{i=1}^{K} w_i = 1$$

$$\text{where} \quad N(\mathbf{m}_i,\mathbf{C}_i) = (2\pi)^{-\frac{1}{2}P}\,|\mathbf{C}|^{-\frac{1}{2}}\exp\left(-\tfrac{1}{2}\mathbf{x}^T\mathbf{C}^{-1}\mathbf{x}\right)$$

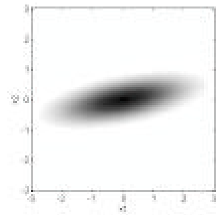
For simplicity we restrict **C** to be diagonal.



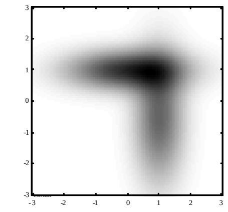
Symmetric: **W**=**I**



Independent: **W** diagonal

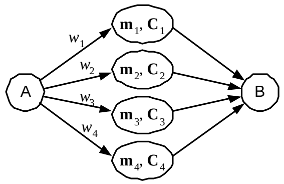


Correlated



Diagonal Gaussian Mixture

Mixtures = Alternate HMM states



- The total probability of all paths from A to B is the sum of the individual path probabilities

$$\text{pd}(\mathbf{x}) = \sum_{i=1}^{K} w_i N(\mathbf{m}_i, \mathbf{C}_i)$$

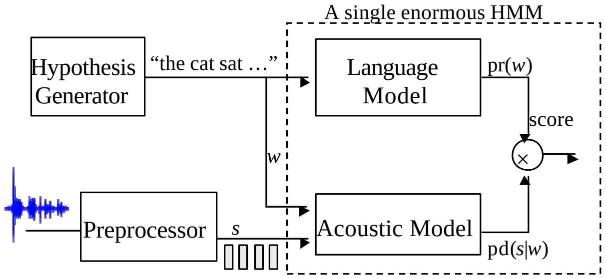
this is identical to the gaussian mixture expression.

- Once we have initial values for the model parameters we can use *Viterbi* and *Baum-Welch* procedures to train them.
- We can view gaussian mixtures as describing alternative pronunciations of a particular speech sound

K-means Algorithm

- We need to form an initial estimate for the K mixture means, $\mathbf{m}_i$, and covariances, $\mathbf{C}_i$.
- First create and train models with only one mixture using Viterbi training.
- Use Viterbi alignment to determine which training frames correspond to each state.
- For each state
  - Set the $\mathbf{m}_i$ to K randomly choosen training frames
  - Repeat until convergence occurs:
    - Allocate each training frame to whichever $\mathbf{m}_i$ it is nearest to.
    - Update each $\mathbf{m}_i$ to the mean of all the frames that were allocated to it
    - If no frames were allocated to $\mathbf{m}_i$, set it to a randomly chosen point from one of the other distributions.
  - Set $\mathbf{C}_i$ to the covariance of the frames allocated to $\mathbf{m}_i$.

Speech Recognition



- Preprocessor
  - Mel Cepstrum + Velocity + Acceleration
  - Linear Transform to decorrelate & reduce F
- Acoustic Model
  - 60,000 triphones × 3 states × 20 features × 10 mixtures = 72,000,000 parameters to train.
- Language Model
  - Phonetic description of each word in vocabulary + trigram or quadram transition probabilities
- Dynamic model creation
  - Create storage only for models when needed: use pruning to delete models with a hopelessly low probability.
  - Trade-off memory/computation versus accuracy

Finite Differences vs. the Bilinear Transform

Recall that the finite difference approximation (FDA) defines the elementary differentiator by $y(n) = x(n) - x(n-1)$ (ignoring the scale factor T for now) which approximates the ideal transfer function $H(s)=s$ by $H_d(z)=1-z^{-1}$. The bilinear transform calls instead for the transfer function $H'_d(z)=(1-z^{-1})/(1+z^{-1})$ (again dropping scale factors) which introduces a pole at $z=-1$ and gives us the recursion $y(n) = x(n) - x(n-1) - y(n-1)$. Note that this new pole is right on the unit circle and is therefore undamped. Any signal energy at half the sampling rate will circulate forever in the recursion, and due to round-off error, it will tend to grow. This is therefore not a very useful improvement of the differentiator. To get something really practical, we need to specify that the filter frequency response approximate $H(j\omega)=j\omega$ over a *finite range* of frequencies $[-\omega_c,\omega_c]$, where $\omega_c\ll\pi$, above which we allow the response to ``roll off'' to zero. This is how we pose the differentiator problem in terms of general purpose filter design, and we will return to this topic later.

To understand the properties of the finite difference approximation in the frequency domain, we may look at the properties of its s-plane to z-plane mapping

$$\displaystyle s = \frac{1 - z^{-1}}{T}$$

We see the FDA is actually a portion of the bilinear transform, since following the FDA mapping by the mapping $s = (c/T)/(1+z^{-1})$ would convert it to the bilinear transform. Like the bilinear transform, the FDA does not alias, since the mapping $s = 1 - z^{-1}$ is one-to-one.

Setting T to 1 for simplicity and solving the FDA mapping for z gives

$$\displaystyle z = \frac{1}{1-s}$$

We see that dc ($s=0$) maps to dc ($z=1$) as desired, but higher frequencies unfortunately map inside the unit circle rather than onto the unit circle in the z plane. Solving for the image in the z plane of the $j\omega$ axis in the s plane gives

$$\displaystyle z = \frac{1}{1-j\omega } = \frac{1 - j \omega }{1+\omega^2}$$

From this it can be checked that the FDA maps the $j\omega$ axis in the s plane to the circle of radius $1/2$ centered at the point $z = 1/2$ in the z plane, as shown in Fig. 1.16

\resizebox{3in}{!}{\includegraphics{eps/lfdacirc.eps}}

Figure 1.16: Image of the $j\omega$ axis in the z plane: a circle of radius $1/2$ centered at the point $z = 1/2$. Note that the analog and digital frequency axes coincide well enough at very low frequencies (high sampling rates).

Under the FDA, analog and digital frequency axes coincide well enough at very low frequencies (high sampling rates), but at high frequencies relative to the sampling rate, *artificial damping* is introduced as the image of the $j\omega$ axis diverges away from the unit circle.

While the bilinear transform ``warps'' the frequency axis, we can say the FDA ``doubly warps'' the frequency axis: It has a progressive, compressive warping in the direction of increasing frequency, like the bilinear transform, but unlike the bilinear transform, it also warps *normal* to the frequency axis.

Consider a point traversing the upper half of the unit circle in the z plane, starting at $z=1$ and ending at $z=-1$. At dc, the FDA is perfect, but as we proceed out along the unit circle, we diverge from the $j\omega$ axis image and carve an arc somewhere out in the image of the right-half s plane. This has the effect of introducing an artificial damping.

Consider, for example, an undamped mass-spring system. There will be a complex conjugate pair of poles on the $j\omega$ axis in the s plane. After the FDA, those poles will be inside the unit circle, and therefore damped in the digital counterpart. The higher the resonance frequency, the larger the damping. It is even possible for unstable s-plane poles to be mapped to stable z-plane poles.

In summary, both the bilinear transform and the FDA preserve order, stability, and positive realness. They are both free of aliasing, high frequencies are compressively warped, and both become ideal at dc, or as f_s approaches ∞. However, at frequencies significantly above zero relative to the sampling rate, only the FDA introduces artificial damping. The bilinear transform maps the continuous-time frequency axis in the s (the $j\omega$ axis) plane precisely to the discrete-time frequency axis in the z plane (the unit circle).

``Discrete-Time Modeling of Lumped Elements (IN PREPARATION)'' by Julius O. Smith III, (From Course Reader, Music 421).

Post-Processing of the feature coefficients

Before describing how to train speech recognition systems, we need to introduce some parameters relative to the features and that will be used during the training and recognition phase. Most of the current speech recognition systems perform post-processing of the feature vectors in order to increase their performance. The most popular post-processing techniques are :

lifting
: Weighting of the different coefficients of the feature vector enhancing the coefficients that are known to be less sensitive to the transmission channel and to the speaker. Liftering leads to great improvement in the case of discrete HMMs.

derivatives
: Introduction of some dynamic parameters in the recognizer is often achieved by adding the first and/or the second derivatives of the coefficients. This has been shown to improve greatly the performance of all recognizers.

STRUT allows the user to perform such post-processing techniques on the feature vector. However, due to the very low computational load required by these techniques, they are never performed by the feature extraction programs. It is actually unnecessary to store liftered parameters or derivatives on the disk. The liftering and derivatives are rather computed by programs processing feature files.

The parameters allowing post-processing of the feature data are :

- `liftering`
  The liftering performed in STRUT is the classical sinusoidal liftering described by the equation :

  $$W_f(m) = 1 + \frac{Q}{2} sin\left(\frac{\pi.m}{Q}\right), 1 \leq m \leq Q$$

  where Q is the number of coefficients (excluding the energy).

- `feature-selection`
  This array of floats allows the user to select the coefficients to be used for training and recognition. For example,

  ```
  feature-selection=[0.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0]
  ```

  supposes that each feature vector stored on the disk is composed of 12 coefficients from which the second to the sixth coefficient will be selected for training and recognition.

  The `feature-selection` parameter also allows the user to define a ``home-made'' liftering. For example,

  ```
  feature-selection=[1.0 1.5 2.0 2.5 3.0 2.5 2.0 1.5 1.0]
  ```

  defines a triangular liftering window.

- `delta-selection`
  This parameter is very similar to `feature-selection` but stands for the computation of the first derivatives of the feature vector components. How the derivaves are computed is defined by the parameter `delta-expression`. Note that the derivatives are computed before the user-defined liftering so that diffrent liftering windows can be applied on the features and their derivatives. This is useful when the user wants normalize the feature components by their variance.

- `delta-delta-selection`
  This parameter selects the second derivatives of the feature components that will be used for training and recognition.

The previous paramters allows selection of feature components and/or their first and second derivatives. STRUT computes these derivatives from regression formula :

$$\Delta f_t(m) = \sum_{k=-K}^{K} g_k f_{t+k}(m), 1 \leq m \leq Q$$

where $f_t(m)$ is the **m** coefficients of the feature vector at time index **t** and g_k are the coefficients of the linear regression specified through the parameters `delta-expression` and `delta-delta-expression`.

- `delta-expression`
  Array of floats containing the coefficients of the linear regression used to compute the first derivatives.
- `delta-delta-expression`
  Array of floats containing the coefficients of the linear regression used to compute the second derivatives. Note that the linear regression is expressed in terms of the feature components and not in terms of their first derivatives.

For example,

```
delta-expression=[-2,-1,0,1,2]
delta-delta-expression=[2,1,-2,-2,-2,1,2]
```

will compute the first derivative in terms of the two preceding and the two following vectors and the second derivative as the difference between the first derivatives of the following and the preceding feature vector.

Now we have completed all the pre-processing of the speech data, and have the feature information for the training and test utterances. Hence, we are in a position to train the STRUT system for speech recognition.

Special speech User
Thu Aug 22 16:29:47 MET DST 1996

Regression Analysis

The most commonly performed statistical procedure in SST is multiple regression analysis. The REG command provides a simple yet flexible way compute ordinary least squares regression estimates. Options to the REG command permit the computation of regression diagnostics and two-stage least squares (instrumental variables) estimates.

The linear regression model

In the linear regression model, the dependent variable is assumed to be a linear function of one or more independent variables plus an error introduced to account for all other factors:

In the above regression equation, y_i is the *dependent variable*, x_1i, ..., x_Ki are the *independent* or *explanatory variables*, and u_i is the *disturbance* or *error term*. The goal of regression analysis is to obtain estimates of the unknown parameters Beta_1, ..., Beta_K which indicate how a change in one of the independent variables affects the values taken by the dependent variable.

Applications of regression analysis exist in almost every field. In economics, the dependent variable might be a family's consumption expenditure and the independent variables might be the family's income, number of children in the family, and other factors that would affect the family's consumption patterns. In political science, the dependent variable might be a state's level of welfare spending and the independent variables measures of public opinion and institutional variables that would cause the state to have higher or lower levels of welfare spending. In sociology, the dependent variable might be a measure of the social status of various occupations and the independent variables characteristics of the occupations (pay, qualifications, etc.). In psychology, the dependent variable might be individual's racial tolerance as measured on a standard scale and with indicators of social background as independent variables. In education, the dependent variable might be a student's score on an achievement test and the independent variables characteristics of the student's family, teachers, or school.

The common aspect of the applications described above is that the dependent variable is a quantitative measure of some condition or behavior. When the dependent variable is qualitative or categorical, then other methods (such as logit or probit analysis, described in Chapter 7) might be more appropriate.

Ordinary least squares estimation

The usual method of estimation for the regression model is ordinary least squares (OLS). Let b_1, ..., b_K denote the OLS estimates of Beta_1, ..., Beta_K. The predicted value of y_i is:

The error in the OLS prediction of y_i, called the *residual*, is:

The basic idea of ordinary least squares estimation is to choose estimates Beta_1, ..., Beta_K to minimize the sum of squared residuals:

It can be shown that:

where X is an n * K matrix with (i,k)th element x_ki, y is an n * K vector with typical element y_i, and b is a K * 1 vector with typical element b_k.

Assumptions for regression analysis

The least squares fitting procedure described below can be used for data analysis as a purely descriptive technique. However, the procedure has strong theoretical justification if a few assumptions are made about how the data are generated. The starting point in the regression equation presented above which describes some causal or behavioral process. The independent variables take play the role of experimental or treatment variables, though in few social science applications will the investigator actually have control over the values of the independent variables. The error term captures the effects of all omitted variables. In an experiment, randomization of the treatments (independent variables) ensures that the omitted factors (the disturbance) are uncorrelated with the treatments. This greatly simplifies inference. Non-experimental researchers, however, must substitute assumptions for experimental controls. The validity of non-experimental results therefore depends critically upon the accuracy of the assumptions. We present one set of assumptions, known as the Gauss-Markov assumptions, that are sufficient to guarantee that ordinary regression estimates will have good properties.

First, we assume that the errors u_i have an expected value of zero: E(u_i) = 0 This means that on average the errors balance out.

Second, we assume that the independent variables are non-random. In an experiment, the values of the independent variable would be fixed by the experimenter and repeated samples could be drawn with the independent variables fixed at the same values in each sample. As a consequence of this assumption, the independent variables will in fact be independent of the disturbance. For non-experimental work, this will need to be assumed directly along with the assumption that the independent variables have finite variances.

Third, we assume that the independent variables are linearly independent. That is, no independent variable can be expressed as a (non-zero) linear combination of the remaining independent variables. The failure of this assumption, known as multicollinearity, clearly makes it infeasible to disentangle the effects of the supposedly independent variables. If the independent variables are linearly dependent, SST will produce an error message (`singularity in independent variables`) and abort the REG command.

Fourth, we assume that the disturbances u_i are homoscedastic:

This means that the variance of the disturbance is the same for each observation.

Fifth, we assume that the disturbances are not autocorrelated:

This means disturbances associated with different observations are uncorrelated.

Properties of the OLS estimator

If the first three assumptions above are satisfied, then the ordinary least squares estimator b will be unbiased: E(b) = beta Unbiasedness means that if we draw many different samples, the average value of the OLS estimator based on each sample will be the true parameter value beta. Usually, however, we have only one sample, so the variance of the sampling distribution of b is an important indicator of the quality of estimates obtained.

If all five of the assumptions above hold, then it can be shown that the variance of the OLS estimator is given by: If the independent variables are highly intercorrelated, then the matrix X' X will be nearly singular and the element of (X' X)^-1 will be large, indicating that the estimates of beta may be imprecise.

To estimate Var(b) we require an estimate of sigma^2. It can be shown that:

is an unbiased estimator of sigma^2. The square root of (sigma hat)^2 is called the *standard error of the regression*. It is just the standard deviation of the residuals e_i.

There are two important theorems about the properties of the OLS estimator. The Gauss-Markov theorem states that under the five assumptions above, the OLS estimator b is best linear unbiased. That is, the OLS estimator has smaller variance than any other linear unbiased estimator. (One covariance matrix is said to be larger than another if their difference is positive semi-definite.) If we add the assumption that the disturbances u_i have a joint normal distribution, then the OLS estimator has minimum variance among all unbiased estimators (not just linear unbiased estimators).

Although the preceding theorems provide strong justification for using the OLS estimator, it should be realized that OLS is rather sensitive to departures from the assumptions. A few outliers (stray observations generated by a different process) can strongly influence the OLS estimates. SST provides useful diagnostic tools for detecting data problems that we discuss below.

Use of the REG command

To estimate a regression in SST, you need to specify one or more dependent variables (in the DEP subop) and one or more independent variables (in the IND subop). Unlike some other programs, SST does not automatically add a constant to your independent variables. If you want one, you should create a constant and add it to the list of your independent variables. For example, to regress the variable y on x with an intercept:

```
set one=1
reg dep[y] ind[one x]
```

SST will produce two coefficients: an intercept and a slope parameter. The corresponding regression line passes through the point (0,b_0) and has slope equal to b_1:

where b_0 is the coefficient of one and b_1 is the coefficient of the variable x. If, on the other hand, you had omitted the variable one from the IND subop:

```
reg dep[y] ind[x]
```

SST would produce a "regression through the origin". That is, the regression line would pass through the point (0,0) with slope equal to the coefficient of *x* (*b*):

For most purposes you will want to include a constant, but SST allows you the flexibility to decide otherwise.

The IF and OBS subops can be used to restrict the range of observations used in the regression. Only the subset of observations activated by the current RANGE statement that meet the criteria set in the IF and OBS subops will be used. If any of the variables specified in the IND or DEP subops have missing data for an observation, the entire observation is deleted from the estimation range for that regression. For example, to run a regression of y on x (and a constant) including only observations one through ten:

```
reg dep[y] ind[one x] obs[1-10]
```

Using the OBS subop does not affect the observation range for subsequent commands.

SST also allows you to specify multiple dependent variables in the DEP subop. If one variable is specified in the DEP subop, it will be regressed on the variables specified in the IND subop. If more than one variable is specified in the DEP subop, separate regressions will be run for each of these variables on the variables listed in the IND subop. The *same* observation range will be used for all regressions.

An example

We illustrate use of the REG command using an example taken from David A. Belsey, Edwin Kuh, and Roy E. Welsch, *Regression Diagnostics* (Wiley, 1980). The variables in the data set are macroeconomic and demographic indicators for fifty countries for the decade of the 1960's and are used to test a simply life cycle savings model.

A text file containing the data is supplied with your SST program disk (bkw.dat). The following variables are in the file:

```
SR        Personal savings rate
POP15     Percentage of population under age 15
POP75     Percentage of population over age 75
PDI       Personal disposable income per capita (constant dollars)
DELPDI    Percentage growth rate of PDI from 1960 to 1970
```

According to the life cycle savings model, savings rates will be highest among middle-aged individuals. Younger individuals, anticipating higher incomes as they become older, will have low savings rates. On the other hand, older individuals will tend to consume whatever savings they accumulated during middle age. The following regression equation is proposed to test this theory:

To replicate the Belsey, Kuh, and Welsch analysis, first we READ the data file bkw.dat as described in Chapter 2. Then we LABEL it, SET a variable one equal to a vector of ones, and SAVE the entire data set. The commands are the following:

```
range obs[1-100]
read to[sr pop15 pop75 dpi deldpi] file[bkw.dat]
label var[sr] lab[average personal savings rate]
label var[pop15] lab[percentage population under 15]
label var[pop75] lab[percentage population over 75]
label var[dpi] lab[real disposable income per capita]
label var[deldpi] lab[real disposable income growth rate]
save file[bkw]
set one=1
```

Now we are ready to regress sr on one, pop15, pop75, dpi, and deldpi:

```
reg dep[sr] ind[one pop15 pop75 dpi deldpi]
```

The output produced is:

```
********** ORDINARY LEAST SQUARES **********
Dependent Variable:        sr

Independent            Estimated            Standard            t-
 Variable             Coefficient             Error          Statistic

      one              28.5662941          7.3544917          3.8841969
    pop15              -0.4611972          0.1446415         -3.1885533
    pop75              -1.6914257          1.0835935         -1.5609412
      dpi              -0.0003371          0.0009311         -0.3620211
   deldpi               0.4096868          0.1961958          2.0881534

Number of Observations               50
R-squared                             0.338458
Corrected R-squared                   0.279655
Sum of Squared Residuals             6.51e+002
Standard Error of the Regression     3.8026627
Mean of Dependent Variable           9.6710000
```

The OLS estimates provide some support for the life cycle model. The *t*-statistic for the coefficient of pop15 is -3.18, which would enable us to reject the null hypothesis beta_2 = 0 at conventional significance levels. The coefficient of pop75, however, does not achieve significance at 0.05 level. Notice, also, that the income effect is small (a thousand dollars of income is associated with only a 0.3 percent rise in the savings rate) and statistically insignificant, while the income change variable has a positive and statistically significant impact on the savings rate.

Regression diagnostics

The REG procedure also allows you to produce diagnostic statistics to evaluate the regression estimates including:

- predicted values (PRED)
- residuals (RSD)
- studentized residuals (SRSD)
- diagonal elements of the "hat" matrix (HAT)
- estimated coefficients (COEF)
- covariance matrix (COVMAT)

Most of these will be familiar, but we discuss in some detail some of the less well known diagnostics: studentized residuals and the hat matrix. These two diagnostics are discussed in detail in *Regression Diagnostics*.

Studentized residuals and the hat matrix

Studentized residuals are helpful in identify *outliers* which do not appear to be consistent with the rest of the data. The *hat matrix* is used to identify "high leverage" points which are outliers among the independent variables. The two concepts are related. In the case of studentized residuals, large deviations from the regression line are identified. Since the residuals from a regression will generally not be independently or identically distributed (even if the disturbances in the regression model are), it is advisable to weight the residuals by their standard deviations (this is what is meant by *studentization*). A similar idea motivates the calculation of the hat matrix (see *Regression Diagnostics*, p. 17).

The hat matrix *H* is given by: H = X(X' X)^-1 X' Note that since: b = (X' X)^-1 X' y and by definition: y hat = Xb it follows that: y hat = Hy Since the hat matrix is of dimension n * n, the number of elements in it can become quite large. Usually it suffices to work with only the diagonal elements h_1, ..., h_n:
where x_i is the *i*th row of the matrix *X*. Note that:

so that:

since I-H is idempotent. It follows therefore:

so that the diagonal elements of the hat matrix are closely related to the variances of the residuals. To compute the studentized residuals, we divide e_i by an estimate of its variance. Rather than using

, we recompute the regression deleting the *i*th observation. Denote the corresponding estimate of sigma^2 with the *i*th observation deleted by s^2 (i) and the corresponding diagonal element of the hat matrix from the regression with the *i*th observation deleted by h_i tilde. The formula for the studentized residual for the *i*th observation is:

Use of the hat matrix diagonal elements

Since y hat = Hy, the diagonal elements of *H*, the h_i, indicate the effect of a given observation. There are a few useful facts about the diagonal elements of the hat matrix:

where *K* is the number of independent variables, including the constant if there is one. Belsley, Kuh, and Welch suggest 2p/n as a rough cutoff for determing high leverage points, terming the ith observation a *leverage point* when h_i exceeds 2p/n.

Use of studentized residuals

Belsley, Kuh, and Welch point out that the studentized residuals have an approximate *t*-distribution with n-p-1 degrees of freedom. This means we can assess the significance of any single studentized residual using a *t*-table (or, equivalently, a table of the standard normal distribution if n is moderately large). For example, we might rerun the above regression and save the studentized residuals by specifying a variable name in the SRSD subop:

```
reg dep[sr] ind[one pop15 pop75 dpi deldpi] srsd[studrsd]
```

Next, we might investigate which observations have large studentized residuals:

```
print var[studrsd] if[abs(studrsd) > 1.96]

   OBS        VARIABLES
                studrsd
    7:          -2.3134
   46:           2.8536
```

Next, we could rerun the regression omitting those observations with large studentized residuals:

```
reg dep[sr] ind[one pop15 pop75 dpi deldpi] if[abs(studrsd) <= 1.96]

********** ORDINARY LEAST SQUARES **********
Dependent Variable:        sr

Independent            Estimated            Standard            t-
 Variable             Coefficient             Error          Statistic

      one              28.8187711          6.5324171          4.4116551
    pop15              -0.4683572          0.1280318         -3.6581323
    pop75              -1.5778925          0.9686178         -1.6290146
      dpi              -0.0003989          0.0008229         -0.4846829
   deldpi               0.3480148          0.1740605          1.9993897

Number of Observations               48
R-squared                             0.410031
Corrected R-squared                   0.355150
Sum of Squared Residuals             4.85e+002
Standard Error of the Regression     3.3589505
Mean of Dependent Variable           9.6747917
```

In this case the coefficient estimates seem relatively stable with the outliers removed.

Instrumental variables estimation

SST also allows easy estimation of simulataneous equations models using two stage least squares. Models where some of the independent variables are correlated with the disturbances will be inconsistently estimated by OLS. If, however, a set of *instrumental variables* (which are correlated with the independent variables, but uncorrelated with the disturbances) is available, it is possible to "purge" the independent variables of their correlation with the disturbance. The instrumental variables are specified in the IV subop. Variables in the IV subop can overlap with those in the IND subop if their are included exogenous variables in the equation. The number of instrumental variables (including included exogenous variables) must be at least as large as the number of independent variables (or else the order condition for identification will not be met).

A simple example of simultaneous equations estimation occurs in estimating a market supply or demand equation. Consider, for example, the supply function for an agricultural commodity. Let price be the market price of the commodity in each period and quantity the quantity supplied of the commodity. In equilibrium, quantity demanded and quantity supplied are equal. Moreover, price and quantity are simultaneously determined in the market, so it does not make sense to regress quantity on price to obtain an estimate of the price elasticity of supply. Suppose, we believe that the supply function also depends on the weather (measured, say, by rainfall) while the demand function also depends on population (populat) and aggregate personal disposable income (pdi), but that these variables are exogenous to the market for this particular commodity. To estimate the supply equation by two stage least squares, give the command:

```
reg dep[quantity] ind[one price weather] iv[one weather populat pdi]
```

Remember to include the constant one in the IV subop, since it is certainly exogenous. For details of simultaneous equations estimation, consult any econometrics text, e.g. H. Theil, *Principles of Econometrics* (Wiley, 1971), chapters 9-10.

LINEAR REGRESSION

LEARNING OBJECTIVE

Upon completion of this station you will be able to:

- Understand the **definition** of linear regression.
- Understand the meaning of **correlation.**
- Use **scatter plots.**
- Recognize and calculate **errors** in linear regression.
- Use **simple linear regression analysis.**
- Solve the **regression equation.**
- Use **residual analysis** of the regression equation.
- Understand the significance of the **correlation coefficient and the regression coefficient** in linear regression.
- Solve **exercise problems** using linear regression.

Regression Applet

The applet below is designed to teach students the effect of leverage points on a regression line. Students may add points to the plot by clicking the mouse button. Students should note that adding points close to the existing line barely changes the line. By adding points far from the existing line, the regression line changes considerably. This is particularly true for points added outside the range of the data. This should help students understand the effect of outliers on regression analysis.

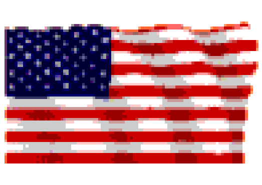
---

by R. Webster West, Dept. of Statistics, Univ. of South Carolina
`west@stat.sc.edu`

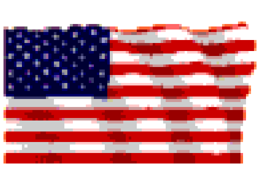
Page was last updated on Monday September 9, 1996 at 17:44:09

The Web's Multiple Regression Home Page

AUTOFIT



AutoFit is a Multiple Regression program that automatically builds a model or regression equation for you. You merely supply the dependent and independent variables and it does the rest. It will find which variables are important enough to include in the model, determine the proper transformation of each of those variables, then look for 2-way and 3-way interaction terms important enough to include in the model, and transform them appropriately.



You can run **AutoFit** below these instructions. Merely supply the URL to your input data file (or enter your data on this page), set any desired options, click **Run Regression**, and the program will develop the model for you. The input data file can have the dependent variable listed anywhere in the file, but the first column is assumed if not indicated. Enter each variable of data as a column of data (not as a row of data). Separate each number of the data file by a comma or a space. The first record of the input data file can optionally be a list of variable names followed by a comma or space. Variable names can be up to 10 characters long, only letters and numbers. If variable names are omitted, then y, x1, x2, etc. will be used. Missing values are not permitted, each data record must be complete.

The program will attempt to transform the independent variable(s) (but not the dependent variable) so as to insure a linear relationship between the independent variable(s) and the dependent variable. Currently possible transformations are:

- sine in radians of x
- e to the minus x power
- e to the x power
- log of x, base e
- x to a power (-4.0 to +4.0)

AutoFit does a stepwise solution in finding which variables to enter into the model, but provides an option to find a simultaneous system solution (SSS) as well. The SSS will use the variables found with the stepwise procedure, but will try all combinations of transformations using a simultaneous system approach, resulting in a very lengthy process. If your sole purpose is to derive a prediction equation then this option is not necessary. However, if the relationships between the independent variables and the dependent variable is important, then SSS the equation. Due to constraints of resources at Lava Net, SSS is limited to models of two independent variables only. Future plans call for SSS models with three or more independent variables when computer resources become available.

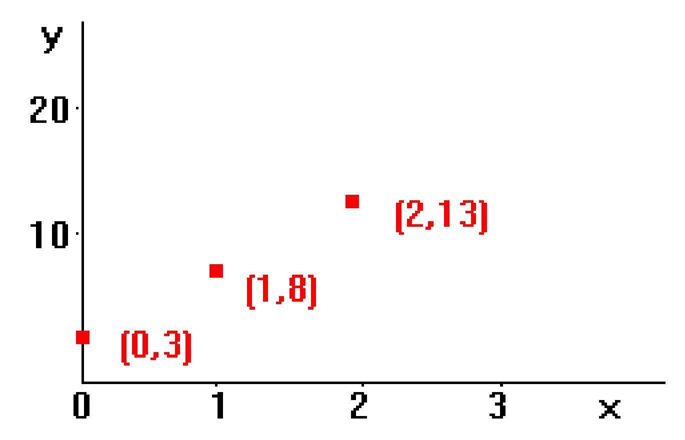
If you choose to use SSS and your model has one or more sine waves, then you must specify the wavelength(s) of those independent variables. You only need to specify the wavelength(s) if using SSS.

When you run the regression, instead of giving you the output on your screen, since most regressions take several minutes to run, the program will submit the regression for processing, emailing you the results when complete, usually within 24 hours.

Your input data does not need to have an intercept term since **AutoFit** supplies one for you. As well, your input data must have more records than independent variables, otherwise the equation cannot be determined. A sample input data file might look like the following:

percentile, ranking, noiselevel,
0.486578286340026, 1, 21,
0.246711000721213, 2, 23,
0.13629187548097, 3, 33,
0.092626329983846, 4, 41,
0.0836961246335616, 5, 39,
0.0823257870395134, 6, 37,
0.0839260785153895, 7, 35,
0.0907101513194246, 8, 31,
0.0948171785407239, 9, 29,
0.0995963399120694, 10, 27,
0.105077813876941, 11, 25,

$$Y_i = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i$$



One final caveat, each set of data is unique, and therefore may not behave as other data sets you have worked with. If you have the time, you should try different regression options such as...

1) scale "y" and scale all independent variables
2) scale "y" but do not scale any independent variables
3) do not scale "y" but scale all independent variables
4) do not scale "y" and do not scale any independent variables

URL of input file:

examples: http://www.lava.net/~seekjc/test.data
http://www.someurl.com/somedata

...OR...

Enter inputdata here:

If your data is not stored on the World Wide Web, you must enter your data above. To enter your data, follow each number by a space or comma, being sure to press the "**Enter**" key at the end of each line.

List your data-file:

Enter a "y" to list or print input data.

Your name:

Your email address:

Dependent variable:

Enter the column number of the dependent variable, eg. 1, 2, 3 etc. (1 is assumed if left blank). This is the only time column number is used to indicate which vector of data is being referenced. All other references to vectors of data are via independent variable number, not column number.

Linear columns:

Enter the independent variable number(s) of the variables you do not want transformed, separated by commas, eg. 2,4,5. Use the letter "a" if you want all independent variables of the data file to be linear. Use the letter "i" if you want all interaction terms to be linear. The dependent variable is always linear by default.

Exclude columns:

Enter the independent variable number(s) of the variables you do not want included in the model, separated by commas, eg. 1,3. Use the letter "i" if you want all interaction terms excluded.

Force columns:

Enter the independent variable number(s) of the variables you want forced into the model, separated by commas, eg. 1,3. Use the letter "a" if you want all independent variables of the data file forced into the equation. Use the letter "i" if you want all interaction terms forced into the equation.

Force transformation:

Independent variables can have a particular transformation forced. This should only be used where the transformation of the variable is known. Enter the independent variable number followed by an equals sign (=), followed by the transformation type where transformation types are
1 x to a power
2 log of x
3 e to the x power
4 e to the minus x
5 sine of x
If the trasformation type is type 1, then follow the 1 with an equals sign (=) and then the power, eg. 2=1=1.8 which means independent variable 2 is to have a power transformation to power 1.8. If more than one forced transformation, separate each with a comma or space. Forced transformations do not apply to simultaneous system solutions.

Re-scaled columns:

Enter the independent variable number(s) of the variables you want to be re-scaled, separated by commas, eg. 1,3. Use the letter "a" if you want all independent variables of the data file to be re-scaled. Use the letter "i" if you want all interaction terms to be re-scaled. Re-scaling should be used when an independent variable's scale does not lend itself to a particular transformation, e.g. e to the minus x, where x is very large for all values of the independent variable, rendering the transformed values close to zero. Re-scaling is generally recommended since transformations like logs and powers will achieve higher predictive capability when re-scaled before transformation.

Dummy var columns:

Enter the independent variable number(s) of the variables you want to be dummy variables, separated by commas, eg. 1,3. Use the letter "a" if you want all independent variables of the data file to be dummy variables. Use the letter "i" if you want all interaction terms to be dummy variables. Dummy variables will not be re-scaled nor transformed.

Orthogonalized IT's:

Enter the independent variable number(s) of the interaction terms you want made orthogonal to the transformed components that are in the model, seperated by commas, eg. 4,5. Use the letter "i" if you want all interaction terms to be made orthogonal.

Residuals to print:

Enter the number of residuals you would like printed in your output. Leave blank to print all residuals.

Scale dependent var:

Enter a "y" for scaling of the dependent variable to a scale of 1 to 2. Dependent variable scaling can be used to eliminate negative values in the dependent variable so that certain independent variable transformations are possible.

Correlation matrix:

Enter a "y" to print correlation matrix of variables. A word of caution here: listing the correlation matrix can be lengthy due to the interaction terms, e.g. a regression of 15 independent variables takes 83519 lines to print out.

Simultaneous system:

Enter a "y" for simultaneous system solution. A simultaneous system solution is when you have two independent variables in the model and every possible permutation of transformations of those two independent variables is evaluated to see which set of transformations is best. Three variables or more in the model are not permitted due to the excessive amount of time required to run the simultaneous system option.

Sine wavelength(s):

Enter the independent variable number(s) of the sine variables followed by an equals sign, then their respective wavelengths, separated by commas, e.g. 1=365.2564, 2=24.0. Use the letter "a" if you want all independent variables of the data file to have the entered wavelength, eg. a=24.0. Use the letter "i" if you want all interaction terms to have the entered wavelength, eg. i=365.2465. Must be entered if simultaneous system solution.

Run Regression Start Over

Other places to do Regression on the World Wide Web
An Introduction to Regression

Email Webmaster

Credits

The twirling letters spelling AUTOFIT came from:
Young Kim at ij@dxnet.com
The waving flags I copied from:
http://www.whitehouse.gov/
The Y = expression I copied from:
http://www.math.uncc.edu/~droyster/maed3103/...
The animated graph I copied from:
http://www.ee.unb.ca/tervo/ee2791/graph0.gif
The animated email I copied from:
http://www.aussie.net/~thunder/