Some Recent Advances in Gaussian Mixture Modeling for Speech Recognition

Ramesh Gopinath
rameshg@us.ibm.com

IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

http://www.research.ibm.com/people/r/rameshg

joint work with Scott Axelrod, Vaibhava Goel, Peder Olsen & Karthik Visweswariah
Outline

• Acoustic Modeling with Hidden Markov Models

• Advances in Gaussian Mixture Modeling
  • Maximum Likelihood Linear Transformation (MLLT) model
  • Extended MLLT (EMLLT) model
  • Precision Constrained GMM (PCGMM)
  • Subspace Constrained GMM (SCGMM)
  • GMMs with Non-linear Feature Space Transformations

• Advances in Adaptation of GMM-based systems
  • Rapid adaptation (FMAPLR)
  • Flexible framework for adaptation of acoustic front-end
How a Speech Recognizer Works

Waveform $a$:

Front-end Sig. Proc.:

$$a \mapsto x$$

$$x = (x_1, x_2, \ldots, x_T)$$

Acoustic Model:

$$p_{\theta_A}(x|w)$$

Language Model:

$$p_{\theta_L}(w)$$

Search:

$$w^* = \arg\max_w p_{\theta_L}(w)p_{\theta_A}(x|w)$$

Best Word String
Acoustic Modeling - Gaussian Mixture Models

- Consider a collection of GMMs depending on HMM state \( s \in S \) modeling an acoustic vector \( x \in \mathbb{R}^d \):

\[
p(x|s) = \sum_{g \in G(s)} \pi_g \mathcal{N}(x|\mu_g, \Sigma_g)
\]

\[
\mathcal{N}(x|\mu_g, \Sigma_g) = \frac{1}{|2\pi\Sigma_g|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_g)^T \Sigma_g^{-1} (x-\mu_g)}
\]

\[
= \left| \frac{P_g}{2\pi} \right|^{\frac{1}{2}} e^{-\frac{1}{2} \psi_g P_g^{-1} \psi_g - \frac{1}{2} x^T P_g x + x^T \psi_g}
\]

- The precision matrix \( P_g = \Sigma_g^{-1} \) and transformed mean \( \psi_g = P_g \mu_g \) appear linearly in the data-dependent terms of the Gaussian exponent.
Gaussian Mixture Models - Some Issues

- Parameters: Priors, Means and Covariances
  
  \[ x \sim \{\pi_g, \mu_g, \Sigma_g\} \Leftrightarrow \{\pi_g, \psi_g, P_g\} \]

- Estimate parameters to Maximize Likelihood (ML) of training data

- Practical considerations (computational, storage & robust estimation) imply covariance constraints in many applications. In a typical speech recognition application
  
  - Dimension of acoustic features: \( d \approx 50 \)
  
  - Number of Gaussians: \( n_{\text{Gauss}} \approx 10^5 \)
  
  - Size of a full-covariance model: \( \approx 530\text{Mb} \)
  
  - Size of a diagonal-covariance model: \( \approx 40\text{Mb} \)
Maximum Likelihood Linear Transform (MLLT)

- Basic idea: Find a linear transformation $A$ that approximately diagonalizes the covariances of all the Gaussians.

- Constrain covariance to be of the form

$$\Sigma_g = A^{-1} \Delta_g A^{-T}, \quad \Delta_g \text{ diagonal.}$$

- Equivalent precision constraint with $\Lambda_g = \Delta_g^{-1}$

$$P_g = A \Lambda_g A^T = \sum_{k=1}^{d} \lambda_g^k a_k a_k^T.$$

- Interpretation: Basis expansion of $P_g$
  - Basis: $\{a_k a_k^T\}_{k=1}^{d}$, Expansion Coefficients: $\{\lambda_g^k\}$
The EMLLT Model

• Extended ML Linear Transform (EMLLT)

\[ P_g = A \Lambda_g A^T = \sum_{k=1}^{D_P} \lambda_g^k a_k a_k^T, \quad d \leq D_P \leq \frac{d(d + 1)}{2} \]

• EMLLT is very flexible

  • \( D_P = d, \ a_k = e_k \Leftrightarrow \) Diagonal Covariance GMM
  • \( D_P = d \Leftrightarrow \) MLLT
  • \( D_P = d(d + 1)/2 \Leftrightarrow \) Full Covariance (Precision)

• Generalized EM algorithm for ML estimates from training data of the basis and expansion coefficients.
The EMLLT Model - A nice interpretation

• In a diagonal covariance GMM the ML variance estimate is equal to the sample variance along the co-ordinate directions.

\[ e_k^T \bar{\sum}_g e_k = e_k^T \Delta_g e_k, \quad 1 \leq k \leq d. \]

• In an EMLLT based GMM the ML variance estimate is equal to the sample variance along the directions \( a_k \).

\[ a_k^T \bar{\sum}_g a_k = a_k^T P^{-1}_g a_k = a_k^T (A \Lambda_g A^T)^{-1} a_k, \quad 1 \leq k \leq D_P. \]
Precision Constrained GMM

• Precision Constrained GMM Model

\[ P_g = \sum_{k=1}^{D_P} \lambda_g^k S_k, \quad d \leq D_P \leq \frac{d(d + 1)}{2} \]

• PCGMMs generalize EMLLT
  - \( D_P = d, S_k = e_k e_k^T \) \iff Diagonal Covariance GMM
  - \( D_P \geq d, S_k = a_k a_k^T \) \iff EMLLT
  - \( D_P = d(d + 1)/2 \) \iff Full Covariance (Precision)

• Generalized EM algorithm for ML estimates from training data of the basis and expansion coefficients.
Subspace Constrained GMM

• Gaussian as an exponential model

\[ \mathcal{N}(x|\psi_g, P_g) = e^{\theta_g^T f(x) + K(\theta_g)} \]

where

\[ f(x) = \begin{bmatrix} x \\ -0.5 \text{vec}(xx^T) \end{bmatrix} \]

\[ K(\theta) = K(\psi, P) = -0.5 \left( \psi^T P^{-1} \psi + \log |P| \right). \]

• SCGMMs constrain \( \theta_g = (\psi_g, \text{vec}(P_g)) \) to affine subspace
SCGMM Model Parameters

\[
\theta_g = \begin{bmatrix} \psi_g \\ \text{vec}(P_g) \end{bmatrix} = B\lambda_g + \theta_0 \in \mathbb{R}^{d+d(d+1)/2}
\]

\[
\theta_0 \in \mathbb{R}^{(d+d(d+1)/2)}, \quad \lambda_g \in \mathbb{R}^D, \quad B \in \mathbb{R}^{(d+d(d+1)/2) \times D}.
\]

Constraints:

\[1 \leq D \leq d + \frac{d(d+1)}{2} \cdot \]

\(P_g\) is positive definite.

Tied and Untied Parameters:

\[
\Theta_{\text{tied}} = (B, \theta_0)
\]

\[
\Theta_{\text{untied}} = \{(\pi_g, \lambda_g)\}_{g=1}^{n\text{Gauss}}.
\]
Special Cases when $B$ is Block Diagonal

$$B_{11} = \begin{bmatrix} l_1 & \ldots & l_{DM} \end{bmatrix} \quad B_{22} = \begin{bmatrix} \text{vec}(S_1) & \ldots & \text{vec}(S_{DP}) \end{bmatrix}$$

$$\psi_g = l_0 + \sum_{k=1}^{DM} \lambda_g^k l_k \quad P_g = S_0 + \sum_{k=1}^{DP} \lambda_g^{k+DM} S_k$$

<table>
<thead>
<tr>
<th>Model</th>
<th>$B_{11}$</th>
<th>${S_k}_{k=1}^{DP}$</th>
<th>$\theta_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagonal</td>
<td>1</td>
<td>${e_k e_k^T}_{k=1}^d$</td>
<td>0</td>
</tr>
<tr>
<td>MLLT</td>
<td>1</td>
<td>${a_k a_k^T}_{k=1}^d$</td>
<td>0</td>
</tr>
<tr>
<td>EMLLT</td>
<td>1</td>
<td>${a_k a_k^T}_{k=1}^{DP}$</td>
<td>non-zero for affine EMLLT</td>
</tr>
<tr>
<td>PCGMM</td>
<td>1</td>
<td>general</td>
<td>non-zero for affine PCGMM</td>
</tr>
<tr>
<td>SPAM</td>
<td>general</td>
<td>general</td>
<td>general</td>
</tr>
</tbody>
</table>
Gaussian evaluation

• Gaussian evaluation involves feature precomputation and per Gaussian computation

\[ \mathcal{N}(x|\psi_g, P_g) = e^{\theta_g^T f(x) + K(\theta_g)} \]
\[ = e^{\lambda_g^T (B f(x)) + [\theta_0^T f(x) + K(\theta_g)]} \]

• Feature precomputation depends on tied parameters \( B \)
• Per Gaussian computation depends on untied parameters \( \lambda_g \)
### Flops to Evaluate Gaussians Exponents*

<table>
<thead>
<tr>
<th>type</th>
<th>precomputation</th>
<th>per Gaussian</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagonal</td>
<td>0</td>
<td>$2d$</td>
</tr>
<tr>
<td>Full Cov</td>
<td>0</td>
<td>$d(d + 1)/2$</td>
</tr>
<tr>
<td>MLLT</td>
<td>$d^2$</td>
<td>$2d$</td>
</tr>
<tr>
<td>EMLLT</td>
<td>$D_P d$</td>
<td>$d + D_P$</td>
</tr>
<tr>
<td>SPAM</td>
<td>$D_P d(d + 1)/2 + D_M d$</td>
<td>$D_M + D_P$</td>
</tr>
<tr>
<td>SCGMM</td>
<td>$D(d + d(d + 1)/2)$</td>
<td>$D$</td>
</tr>
</tbody>
</table>

| time to precompute features | = | # tied parameters |
| time to compute Gaussians given features | = | # untied parameters |

* up to constants, counting mult. + add. as one flop
Training SCGMM parameters

- Train parameters to Maximize Likelihood (ML)
- Alternate between maximizations wrt tied and untied parameters leaving the other fixed
- Optimization of untied parameters trivial to parallelize: tied parameter training bottleneck.
- Parameter optimization using quasi-newton search (limited memory BFGS) with efficient line-search that handles positive-definiteness constraint on $P_g$
- Objective function (so-called $Q$-function) is concave in tied and untied parameters separately
Experimental Setup

- Training and test database recorded in a car at various speeds.
- 500 hours training, 75k words test.
- 89 phones, 680 context dependent states.
- 117 dimensional “spliced cepstral features” projected to lower dimensional space using Linear Discriminant Analysis (LDA)
- 10k Gaussians
- Report combined WER from 4 grammar based tasks (Address, Command and Control, Digits, and Radio control) recorded in car at speeds 0, 30, and 60 mph.
PCGMMs: Near FC Performance at DC cost

- Take 52-dim case: PCGMM bridges 64% of performance gap from MLLT to FC at no additional cost!
- Gains uniform across feature-space dimension sizes
EMLLT & PCGMMs: Nature of FC approximation

<table>
<thead>
<tr>
<th>$D_P, d = 52$</th>
<th>EMLLT</th>
<th>PCGMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$</td>
<td>2.67</td>
<td>1.96</td>
</tr>
<tr>
<td>$2d$</td>
<td>2.04</td>
<td>1.75</td>
</tr>
<tr>
<td>$4d$</td>
<td>1.81</td>
<td>1.65</td>
</tr>
<tr>
<td>$8d$</td>
<td>1.65</td>
<td>1.64</td>
</tr>
<tr>
<td>$26.5d$</td>
<td>1.58</td>
<td>1.58</td>
</tr>
</tbody>
</table>

- As $D_P \xrightarrow{\frac{d(d+1)}{2}}$ EMLLT & PCGMM approach FC
- Performance gap largest for $D_P \approx d$
Comparison of MLLT, PCGMM, SPAM, and SCGMM

<table>
<thead>
<tr>
<th>Size</th>
<th>LDA+MLLT</th>
<th>PCGMM</th>
<th>SPAM</th>
<th>SCGMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>-</td>
<td>-</td>
<td>4.69</td>
<td>3.42</td>
</tr>
<tr>
<td>26</td>
<td>4.92</td>
<td>4.58</td>
<td>3.26</td>
<td>2.71</td>
</tr>
<tr>
<td>40</td>
<td>3.71</td>
<td>3.18</td>
<td>2.59</td>
<td>2.35</td>
</tr>
<tr>
<td>78</td>
<td>2.85</td>
<td>2.14</td>
<td>2.10</td>
<td>1.95</td>
</tr>
<tr>
<td>104</td>
<td>2.70</td>
<td>1.97</td>
<td>1.96</td>
<td>1.90</td>
</tr>
</tbody>
</table>

- Models in each row have same # of parameters
- Last row has complexity of 52-dim diagonal cov model
- SCGMM gives MLLT performance at one fourth the cost!
Varying the Number of Gaussians

<table>
<thead>
<tr>
<th>nGauss</th>
<th>MLLT WER</th>
<th>Full-Cov WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>3.48</td>
<td>1.83</td>
</tr>
<tr>
<td>10000</td>
<td>2.68</td>
<td>1.56</td>
</tr>
<tr>
<td>42993</td>
<td>2.00</td>
<td>1.35</td>
</tr>
<tr>
<td>142622</td>
<td>1.74</td>
<td>1.54</td>
</tr>
<tr>
<td>350286</td>
<td>1.68</td>
<td>-</td>
</tr>
<tr>
<td>609100</td>
<td>1.65</td>
<td>-</td>
</tr>
</tbody>
</table>

- **10k** FC model is one fifth the size of **609k** MLLT model with roughly the same performance.
GMMs with Nonlinear Feature Transforms

- Feature transform: $y = h(x)$ where $x, y \in \mathbb{R}^d$.

- $y$ is modeled by a GMM for each HMM state $s \in S$, 

$$ p(y|s) = \sum_{g \in s} \pi_g \mathcal{N}(y|\mu_g, \Sigma_g), \quad p(x|s) = p(y|s) |\det (J_h(x))|, $$

where $J_h(x) = \{ \frac{\partial y_i}{\partial x_j} \}_{i,j}$ is the Jacobian matrix of the transform $h$.

- ML estimation of $h$: Generalized EM approach in principle

- For $h$ volume preserving, $J_h(x) = 1$ and the $Q$ function is 

$$ Q(h) = \sum_{g \in G} n(g) \log |\det \Sigma_{gh}|, \quad n(g) = \sum_t \gamma_{tg}. $$
Constraints on $h$

- Restrict $h$ so that the Jacobian is lower triangular: $J_h(x) = 1$

  $$
  h_1(x) = x_1 \\
  h_2(x) = x_2 + u_2(x_1) \\
  \vdots \quad \vdots \\
  h_d(x) = x_d + u_d(x_1, x_2, \ldots, x_{d-1}).
  $$

- Restrict $h$ to the affine quadratic family to evaluate $\Sigma_{gh}$ efficiently

  $$
  u_j(x_1, \ldots, x_{j-1}) = \sum_{n=1}^{j-1} \sum_{m=1}^{n} a_{mnj} x_m x_n, \quad j = 2, \ldots, d.
  $$

- Allow for pre/post linear transforms: $y = Bq(Ax)$. 
Word error rates with linear and quadratic feature transforms:

<table>
<thead>
<tr>
<th>Type</th>
<th>nGauss</th>
<th>Transform</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$d = 20$</td>
</tr>
<tr>
<td>FCov 10K</td>
<td></td>
<td>$x$</td>
<td>2.54%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q(Ax)$</td>
<td>2.80%</td>
</tr>
<tr>
<td>Diag 10K</td>
<td></td>
<td>$x$</td>
<td>4.14%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q(x)$</td>
<td>4.04%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q(Ax)$</td>
<td>3.65%</td>
</tr>
<tr>
<td>MLLT 10K</td>
<td></td>
<td>$Bx$</td>
<td>3.78%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$Bq(x)$</td>
<td>3.56%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$Bq(Ax)$</td>
<td>3.64%</td>
</tr>
</tbody>
</table>
Experiments - State Dependent Quadratic Transforms

Word error rates for full covariance models with state dependent quadratic feature space transforms:

<table>
<thead>
<tr>
<th>Type</th>
<th>nGauss</th>
<th>Transform(s)</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$d = 20$</td>
</tr>
<tr>
<td>FCov</td>
<td>680</td>
<td>$x$</td>
<td>6.75%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q^j(A_j x)$</td>
<td>4.32%</td>
</tr>
<tr>
<td>FCov</td>
<td>10K</td>
<td>$x$</td>
<td>2.54%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$q^j(A_j x)$</td>
<td>2.66%</td>
</tr>
</tbody>
</table>
Adaptation

• Problem: Mismatch between training and test data

• Adaptation Approaches
  • Adapt the test acoustic features to the trained model
  • Adapt the trained acoustic model to the test acoustic features

• Popular Approach: Linear Transform of features: \( y = Ax + b \)
  • ML estimation of \((A, b)\) (FMLLR)
  • Issue: Need few tens of seconds of adaptation data
FMAPLR: Rapid Adaptation with Little Adaptation Data

• Subspace Constraint: Restricting linear transform to a subspace works well \( A = \sum_i \lambda_i A_i \)

• Soft Subspace Constraint: Estimating via MAP works even better

\[
\max_A P(x|A)P(A)
\]

• Learning a Gaussian prior \( P(A) \)
  • Learn covariance on training data (mean taken as identity)
  • Storing and computing with full covariance costly: Approximate using factor analysis, \( \Sigma = D + \Lambda \Lambda^T \)
FMAPLR Results

<table>
<thead>
<tr>
<th>Method</th>
<th>SER</th>
<th>Avg. session length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>10.7</td>
<td>8sec</td>
</tr>
<tr>
<td>FMLLR</td>
<td>9.8</td>
<td>8sec</td>
</tr>
<tr>
<td>FMAPLR ( nFactors = d )</td>
<td>8.9</td>
<td>8sec</td>
</tr>
</tbody>
</table>
Adaptation of Front-End

• Raw acoustic data for an utterance (pcm): \( a \)

• Final features \( x \): Signal processing pipeline

\[
x = h_M(\theta_M, \ldots h_2(\theta_2, h_1(\theta_1, a)))
\]

• Objective: Minimize \( g(x(\Theta), \mathcal{M}) \) over \( \Theta \), \( \Theta = (\theta_1, \ldots, \theta_M) \) and \( \mathcal{M} \) is the acoustic model for speech recognition.
Choice of the Objective Function

• ML: Likelihood of data under fixed model (e.g., FMLLR)
  • Require \( h_i \) to be invertible i.e., \( J_{h_i}(\cdot) \) cannot vanish anywhere
  • Invertibility cannot be guaranteed e.g., LDA, Mel-Binning

• MMI: Posterior probability of correct sentence under fixed model
  • Need correct transcript (i.e., unsupervised adaptation not possible)
  • Harder to optimize

• MMI’: Approximate MMI
  • Single Gaussian denominator model
Choice of the objective function: Contd

• The three objective functions we consider:

\[ g_{ML}(x) = -\log \det J_h(x) - \log P(x|M, G_N) \]

\[ g_{MMI}(x) = \log P(x|M, G_D) - \log P(x|M, G_N) \]

\[ g_{MMI'}(x) = -0.5N \log \det T - \log P(x|M, G_N) \]
Optimization of objective function

- Limited Memory BFGS: requires gradient of $g(x)$.
- Chain rule: Calculate gradient of $g$ w.r.t $x$ and backpropagate
- Lends to a modular implementation of the required calculation.
- Let $y_i = h_i(y_{i-1}, \theta_i)$. Assume we have calculated the gradient of the objective function w.r.t $y_i$ then

\[
\frac{dg(x)}{d\theta_i} = \frac{dg(x)}{dy_i} \frac{dy_i}{d\theta_i},
\]

\[
\frac{dg(x)}{dy_{i-1}} = \frac{dg(x)}{dy_i} \frac{dy_i}{dy_{i-1}}.
\]
Modular Implementation

- **Gradients**: Each module computes gradient w.r.t its input & parameters

  \[
  \frac{dg}{dy_2} = \frac{dg}{dx} \frac{dx}{dy_2}
  \]

  - \(y_1 \xrightarrow{h_1(y_1, \theta_1)} y_2 \xrightarrow{h_2(y_2, \theta_2)} y_3 = x\)

- **Jacobian**: For ML objective each module computes its own log-det Jacobian exploiting

  \[
  \log \det J_h(x) = \sum_i \log \det J_{h_i}(x).
  \]
Example: Linear Transforms on a SPAM Model

- Baseline WER: **1.99%**.
- ML (FMLLR): **1.55%**, MMI: **1.74%** (overtraining), Hybrid: **1.49%**.
- Pre-LDA Linear Transform Adaptation (i.e., on Cepstra)
  - Objective Function: $g_{\text{MMI}'}(x)$

<table>
<thead>
<tr>
<th>Transform type</th>
<th>Number of parameters</th>
<th>Number of Adaptation utts.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Pre-LDA</td>
<td>169</td>
<td>1.67</td>
</tr>
<tr>
<td>Post-LDA (FMLLR)</td>
<td>2704</td>
<td>2.08</td>
</tr>
</tbody>
</table>
Flexibility: Some Additional Examples of Adaptation

• LDA Adaptation
  • Objective Function: $g_{\text{MMI}'}(x)$
  • Baseline WER: 2.08%, FMLLR: 1.41%, LDA-Adapt.: 1.20%

• Spectral Subtraction (SS) Adaptation
  • Parametrized analytical approximation of SS used for $h_i$
  • Objective Function: $g_{\text{MMI}'}(x)$
  • Baseline WER: 1.98%, SS-adaptation ($d$ parameters): 1.52%

• Other front-end components experimented with: Power-law compression of Spectrum, Mel-Binning, DCT etc.
Summary

• Modeling
  • SCGMM: Gaussian parameters \((\psi_g, P_g)\) constrained to shared affine subspace
  • SCGMM & specializations (SPAM, PCGMM, EMLLT & MLLT) give a flexible way to build robust, compact & efficient GMMs

• Adaptation
  • FMAPLR: a robust technique for rapid adaptation
  • Introduced flexible framework for adaptation of front-end parameters in a speech recognizer
Backup
EM Update Formulas

\( J(x) \) is independent of priors, means and covariances. Optimization wrt \( \pi_g, \mu_{gh} \) and \( \Sigma_{gh} \) achieved with EM update formulas (\( \gamma_{tg} \) are the occupation counts):

\[
\begin{align*}
\pi_g & = \frac{n(g)}{\sum_{g^* \in s} n(g^*)}, \quad n(g) = \sum_t \gamma_{tg} \\
\mu_{gh} & = \frac{1}{n(g)} \sum_t \gamma_{tg} y_t \quad \text{and} \\
\Sigma_{gh} & = \frac{1}{n(g)} \sum_t \gamma_{tg} (y_t - \mu_{gh})(y_t - \mu_{gh})^T.
\end{align*}
\]
Criteria for Choosing Nonlinear Feature Transform

The auxiliary function at the optimal values for $\pi_g$, $\mu_{gh}$ and $\Sigma_{gh}$ becomes

$$Q(h) = -\sum_g n(g)\ell_g(h),$$

where

$$\ell_g(h) = \log |\det \Sigma_{gh}| + 2n(g)\sum_t \gamma_{tg} \log |\det (J(x_t))|.$$

If $h$ is constrained to a small family of functions, then $Q(h)$ can be used to choose $h$. 
Computational Considerations

• Cost of evaluating \( \det(J_h(x_t)) \) for \( t = 1, \ldots, T \) involves at least \( \mathcal{O}(Td^3) \) flops.

• Eliminate cost of Jacobian by restricting to lower triangular, volume preserving transforms of the form

\[
\begin{align*}
    h_1(x) &= x_1 \\
    h_2(x) &= x_2 + u_2(x_1) \\
    h_3(x) &= x_3 + u_3(x_1, x_2) \\
    &\quad \vdots \quad \vdots \\
    h_d(x) &= x_d + u_d(x_1, x_2, \ldots, x_{d-1}).
\end{align*}
\]
Sufficient Statistics

Can evaluate $\Sigma_{gh}$ efficiently for any $h$ in the affine family

$$h(x; \phi) = h^0(x) + \sum_{j=1}^{n} \phi_j h^j(x) = \sum_{j=0}^{n} \phi_j h^j(x).$$

Sufficient statistics are

$$\mu^j_g = \frac{1}{n(g)} \sum_t \gamma_{tg} f^j(x_t)$$

and

$$\Sigma_{g}^{ij} = \frac{1}{n(g)} \sum_t \gamma_{tg} f^i(x_t) f^j(x_t)^T - \mu^i(\mu^j)^T.$$
Quadratic Feature Transforms

For experiments we restricted ourselves to the affine quadratic family:

\[ h_j(x_1, \ldots, x_{j-1}) = \sum_{n=1}^{j-1} \sum_{m=1}^{n} a_{mnj} x_m x_n, \quad j = 2, \ldots, d. \]

The free parameters \( \phi \) here are \( a_{mnj} \) and their cardinality is \( n = \binom{d-1}{3} \).

Will denote the full quadratic transform by \( q(x) \).
Linear Feature Transforms

The data can be linearly transformed both before and after the quadratic transform at no increase in the order of computational complexity:

\[ x \rightarrow y = Ax \]
\[ y \rightarrow z = q(y) \]
\[ z \rightarrow Bz. \]
Gradient Computation

Optimized using a numerical package. The gradient of $\ell_g(\phi)$ with respect to $\phi$ is:

$$
\frac{\partial \ell_g}{\partial \phi_j} = \text{trace} \left( \Sigma_{gh}^{-1} \frac{\partial \Sigma_{gh}}{\partial \phi_j} \right) = 2 \sum_{i=1}^{n} \phi_i \text{trace} \left( \Sigma_{gh}^{-1} \Sigma_{ij} \right).
$$

Adding a linear transform $y = Ax$ requires the additional gradient

$$
\frac{\partial \ell_g}{\partial A_{ij}} = \frac{\partial}{\partial A_{ij}} \log |\det \Sigma_{gh}| - 2 \log |\det A| = 2(A^{-1})_{ji}.
$$
Experiments - test database

- Data collected in a car at 0, 30 and 60mph.
- 22 speakers, 73743 words.
- Four grammars: A - Addresses, C - Commands, D - Digits, R - Radio control.

A: NEW YORK CITY NINETY SIXTH STREET WEST
C: SET TRACK NUMBER TO SEVEN
D: NINE THREE TWO THREE THREE ZERO ZERO
R: TUNE TO F.M. NINETY THREE POINT NINE
Acoustic Models

• Front End: 13 dimensional Mel Frequency Cepstral Coefficients.

• Input Features: Adjoined 9 consecutive MFCC vectors and applied $117 \times 20$ or $117 \times 39$ dimensional LDA matrix.

• HMM states: 680 word internal triphones.

• Training data: 462388 utterances, car data.

• Acoustic model: 10K gaussians, trained using BIC.
Calculation of the gradient: Contd

• To calculate gradient of our objective functions we need to calculate gradient of $\log P(x|M,G)$ w.r.t $x$

$$
\frac{d \log P(x|M,G)}{dx_t} = \sum_{g \in G_t} \gamma_g(t) \Sigma_g^{-1}(\mu_g - x_t)
$$

• Forward backward algorithm allows calculation of this efficiently.

• Gradient of $\log \det T$ w.r.t $x_t$ is $2T^{-1}(x_t - \mu)$
Calculation of the gradient: Contd

- One final term \( \log \det \) of the Jacobian

- Assuming all front end modules of interest are invertible the \( \log \det \) of the Jacobian can be broken down as the \( \log \det \) of the Jacobian’s of individual modules and they can be calculated by the module
Linear Transforms before Lda

<table>
<thead>
<tr>
<th>Trns. type</th>
<th>num. parms</th>
<th>Adaptation sents.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Pre-Lda</td>
<td>169</td>
<td>1.67</td>
</tr>
<tr>
<td>Post-Lda</td>
<td>2704</td>
<td>2.08</td>
</tr>
<tr>
<td>Post-Lda bd-13</td>
<td>676</td>
<td>1.87</td>
</tr>
<tr>
<td>Post-Lda bd-4</td>
<td>208</td>
<td>1.88</td>
</tr>
</tbody>
</table>

Table 1:Comparisons of linear transforms before and after the LDA with varying amounts of data
Adapting other parameters

<table>
<thead>
<tr>
<th>Transform type</th>
<th>objective function</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>-</td>
<td>2.61</td>
</tr>
<tr>
<td>LinTrans post-Lda</td>
<td>$\mathcal{g}_{ML}$</td>
<td>1.79</td>
</tr>
<tr>
<td>DCT</td>
<td>$\mathcal{g}_{MMI}'$</td>
<td>1.71</td>
</tr>
<tr>
<td>DCT + root compression</td>
<td>$\mathcal{g}_{MMI}'$</td>
<td>1.68</td>
</tr>
<tr>
<td>LinTrans pre-LDA</td>
<td>$\mathcal{g}_{MMI}'$</td>
<td>1.69</td>
</tr>
<tr>
<td>Mel bins</td>
<td>$\mathcal{g}_{MMI}'$</td>
<td>2.65</td>
</tr>
<tr>
<td>Mel bins</td>
<td>$\mathcal{g}_{MMI}$</td>
<td>2.65</td>
</tr>
</tbody>
</table>

Table 2: Adaptation at various stages in the front end