NOTES ON DIFFERENTIABLE FMLLR

Daniel Povey

Johns Hopkins University,
3400 N Charles St, Baltimore, MD 21218
dpovey@gmail.com
(Written in 2018)

ABSTRACT

These are some notes on the math for differentiable fMLLR, intended for use in neural network training (so we can backprop through the fMLLR to train a neural network that generates features to be used with fMLLR adaptation). The reason this is not trivial is that we need to account for how the fMLLR transform itself changes when the features change.

Index Terms— fMLLR, DNNs

1. INTRODUCTION

This document is a small piece of a larger plan for neural net training integrated with a form of adaptation based on fMLLR. The idea is that the fMLLR (search the literature for "constrained MLLR" for explanation) operates on features generated by a feature-extracting neural net, and these adapted features are fed into a second neural net.

For that to work, we need to differentiate the fMLLR transform matrix w.r.t. the things used to estimate it, namely: the features; and the model means and variances in the model we are adapting the features to. This will enable us to have a more precise derivative for how the objective function depends on the features that are the input to the fMLLR, and will hopefully allow the network to learn features that are more "adaptable"– meaning more amenable to adaptation by fMLLR.

This document will cover the situation where you are estimating the fMLLR transform for a particular speaker, given class means and (spherical) variances, soft-counts expressing the class assignments, and input features. This document derives the estimation of the fMLLR transform and the computation of derivatives w.r.t. the inputs used to estimate it. For background: fMLLR, also known as "constrained MLLR", is a linear dimension-preserving transform of the feature space, estimated in a Maximum Likelihood fashion to maximize the likelihood of the features as modeled by some kind of model involving Gaussian likelihoods. See [1], although the presentation there gives it a more "model-space" interpretation than is used by most other authors.

The actual setup will be that the fMLLR transforms are estimated per speaker on each minibatch, and we will have ensured that each speaker in the minibatch has enough data in that minibatch that we can estimate a transform. We'll also estimate the model means and variances per minibatch, and since the model means and variances depend on the features used to estimate them, those derivatives will ultimately be propagated back to those features. But all that is fairly straightforward and is outside the scope of this document. We are concerned with the core estimation of the FMLLR transform matrix.

Section 2 derives the objective function that we are maximizing and Section 3 finds a closed-form solution for the transform matrix. Section 4 summarizes our solution as a "how-to" document. The reader who is not interested in the theory but just wants to know how to implement this, will likely want to skip directly to Section 4.

2. THE OBJECTIVE FUNCTION

First, to establish notation: let the input features be $\mathbf{x}_t \in \Re^D$, for $t = 1 \dots T$; the feature dimension is D. We'll be estimating fMLLR transform parameters $\mathbf{A} \in \Re^{D \times D}$ (the linear term) and $\mathbf{b} \in \Re^D$ (the offset), so the adapted features will be:

$$\mathbf{y}_t = \mathbf{A}\mathbf{x}_t + \mathbf{b}. \tag{1}$$

The setup we've chosen is that there are classes $i = 1 \dots I$ (e.g. these might correspond to some kind of context dependent phone or HMM state), and each class has a mean $\boldsymbol{\mu}_i$ and a spherical variance $s_i\mathbf{I}$ (where s_i is the scalar factor that we assume the user has provided). In actuality, these would have been estimated from the input features $\mathbf{x}_t$ somehow, and they will also end up contributing to the derivative w.r.t. $\mathbf{x}_t$, but we won't go through the details here.

We also assume that the user has provided us with posterior probabilities over the classes on each time frame, which we'll write as $\gamma_{t,i}$. These might normally be expected to satisfy $\sum_i \gamma_{t,i} = 1$ for each t, but we won't need to assume this. Note: in our application we won't be keeping track of the derivatives w.r.t. $\gamma_{t,i}$.

The likelihood function we are maximizing is:

$$\mathcal{L} = \sum_{t,i} \gamma_{t,i} \left(\log |\mathbf{A}| + \mathcal{N}(\mathbf{y}_t; \boldsymbol{\mu}_i, s_i\mathbf{I}) \right), \tag{2}$$

and for the justification for the need for the log-determinant term, you can probably find it in [1], although, as mentioned, that document is based on a model-space rather than feature-space interpretation of fMLLR. (2) expands to:

$$\mathcal{L} = \sum_{t,i} \gamma_{t,i} \left(\log |\mathbf{A}| - \frac{1}{2s_i} (\mathbf{y}_t - \boldsymbol{\mu}_i)^T (\mathbf{y}_t - \boldsymbol{\mu}_i) \right) + \text{const}, \tag{3}$$

where by "const" we mean terms that are independent of $\mathbf{A}$. Because of our assumption of spherical covariances, it will be possible to separate the estimation of $\mathbf{b}$ from that of $\mathbf{A}$, and this will simplify the process of estimating $\mathbf{A}$. Firstly, notice that in the expression above, the log-likelihood looks a lot like a log-likelihood with unit covariance but with occupation counts given by $\frac{\gamma_{t,i}}{s_i}$. In fact, it will be convenient to define modified occupation counts:

$$\hat{\gamma}_{t,i} = \frac{\gamma_{t,i}}{s_i} \tag{4}$$

and let us define the total-count (and modified-total-count):

$$\gamma = \sum_{t,i} \gamma_{t,i} \quad (5)$$

$$\hat{\gamma} = \sum_{t,i} \hat{\gamma}_{t,i}. \quad (6)$$

Then define the weighted averages of the means, and of $\mathbf{x}$, as follows:

$$\mathbf{m} = \frac{1}{\hat{\gamma}} \sum_i \left(\sum_t \hat{\gamma}_{t,i} \right) \boldsymbol{\mu}_i \quad (7)$$

$$\mathbf{n} = \frac{1}{\hat{\gamma}} \sum_t \left(\sum_i \hat{\gamma}_{t,i} \right) \mathbf{x}_t. \quad (8)$$

and then define the following quantities, which are "mean-subtracted" versions of the features and class means:

$$\hat{\mathbf{x}}_t = \mathbf{x}_t - \mathbf{n} \quad (9)$$

$$\hat{\boldsymbol{\mu}}_i = \boldsymbol{\mu}_i - \mathbf{m} \quad (10)$$

We'll then estimate $\mathbf{A}$ with reference to these mean-subtracted quantities; and once $\mathbf{A}$ is estimated we'll set:

$$\mathbf{b} = \mathbf{m} - \mathbf{A}\mathbf{n}. \quad (11)$$

We won't go through the derivation of the parts related to the mean; it's intuitively quite easy. The basic intuition is that the derivative w.r.t. the feature-shift $\mathbf{b}$ is zero when the appropriately weighted means of the transformed features $\mathbf{y}_t$ and of the model means are the same. We can use the following objective function to optimize $\mathbf{A}$:

$$\begin{aligned} \mathcal{L} &= \sum_{t,i} \gamma_{t,i} \left(\log|\mathbf{A}| - \frac{1}{2s_i} (\mathbf{A}\hat{\mathbf{y}}_t - \hat{\boldsymbol{\mu}}_i)^T (\mathbf{A}\hat{\mathbf{y}}_t - \hat{\boldsymbol{\mu}}_i) \right) \\ &= \gamma \log|\mathbf{A}| - \sum_{t,i} \frac{\hat{\gamma}_{t,i}}{2} (\mathbf{A}\hat{\mathbf{x}}_t - \hat{\boldsymbol{\mu}}_i)^T (\mathbf{A}\hat{\mathbf{x}}_t - \hat{\boldsymbol{\mu}}_i) \\ &= \gamma \log|\mathbf{A}| + \mathrm{tr}\left(\mathbf{A}^T\mathbf{K}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{G}\mathbf{A}^T\right) + \mathrm{const} \quad (12) \end{aligned}$$

where:

$$\mathbf{G} = \sum_{t,i} \hat{\gamma}_{t,i} \hat{\mathbf{x}}_t \hat{\mathbf{x}}_t^T \quad (13)$$

$$\mathbf{K} = \sum_{t,i} \hat{\gamma}_{t,i} \hat{\boldsymbol{\mu}}_i \hat{\mathbf{x}}_t^T \quad (14)$$

3. SOLVING IT

We find the $\mathbf{A}$ that maximizes (12) with the help of two changes of variables. Define

$$\mathbf{B} = \mathbf{A}\mathbf{G}^{0.5} \quad (15)$$

so that

$$\mathbf{A} = \mathbf{B}\mathbf{G}^{-0.5} \quad (16)$$

and we can rewrite (12) as:

$$\mathcal{L} = \gamma \log|\mathbf{B}| + \mathrm{tr}\left(\mathbf{G}^{-0.5}\mathbf{B}^T\mathbf{K}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{B}\mathbf{G}^{-0.5}\mathbf{G}\mathbf{G}^{-0.5}\mathbf{B}^T\right) + \mathrm{const} \quad (17)$$

which can be simplified to:

$$\mathcal{L} = \gamma \log|\mathbf{B}| + \mathrm{tr}\left(\mathbf{B}^T\mathbf{L}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{B}^T\mathbf{B}\right) + \mathrm{const}. \quad (18)$$

where

$$\mathbf{L} = \mathbf{K}\mathbf{G}^{-0.5}. \quad (19)$$

Our solution is to do an SVD on $\mathbf{L}$, i.e.

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T, \quad (20)$$

and to consider solutions for $\mathbf{B}$ of the form:

$$\mathbf{B} = \mathbf{U}\mathbf{D}\mathbf{V}^T \quad (21)$$

where $\mathbf{U}$ and $\mathbf{V}$ are the same values as we obtained in the SVD of $\mathbf{L}$, and $\mathbf{D}$ is a diagonal matrix for which we need to solve for the diagonal elements, as we'll explain below. After writing out (18) with these factorizations of $\mathbf{L}$ and $\mathbf{B}$, and canceling products like $\mathbf{U}\mathbf{U}^T$ which reduce to the unit matrix, we get:

$$\mathcal{L} = \gamma \log|\mathbf{D}| + \mathrm{tr}\left(\mathbf{D}\boldsymbol{\Lambda}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{D}\mathbf{D}\right) + \mathrm{const}. \quad (22)$$

For purposes of showing that this is the optimal solution, it is convenient to treat this just like a change of variables (like when we changed $\mathbf{A}$ to $\mathbf{B}$) and to treat $\mathbf{D}$ as an arbitrary quantity that does not necessarily have to be diagonal. We can show that $\mathbf{D}$ must be diagonal at a maximum of the objective function, by noting that the derivative of (22) w.r.t. $\mathbf{D}$ is

$$\frac{\partial \mathcal{L}}{\partial \mathbf{D}} = \gamma \mathbf{D}^{-T} + \boldsymbol{\Lambda} - \mathbf{D}^T. \quad (23)$$

and to note that $\boldsymbol{\Lambda}$ is diagonal, so if this derivative is to equal zero, then the off-diagonal parts of $\gamma\mathbf{D}^{-1}$ and $-\mathbf{D}$ must cancel, which implies that $\mathbf{D}$ itself is diagonal[1]. The proof of this would take a couple of paragraphs, I believe. It relies on the fact that the function $f(x) = \gamma/x - 1$ is anti-monotone, so that initially distinct eigenvalues of $\mathbf{D}$ stay distinct... we'll try to work on the details of this [2] The objective function is now separable over the diagonal elements d_i of $\mathbf{D}$, and can be written as:

$$\mathcal{L} = \sum_{i=1}^{D} \gamma \log|d_i| + d_i\lambda_i - \frac{1}{2}d_i^2. \quad (24)$$

The derivative of this w.r.t. d_i is:

$$\frac{\partial \mathcal{L}}{\partial d_i} = \frac{\gamma}{d_i} + \lambda_i - d_i \quad (25)$$

and equating to 0 and multiplying by d_i, we get:

$$-d_i^2 + \lambda_i d_i + \gamma = 0 \quad (26)$$

which we can solve by applying the quadratic formula with $a = -1$, $b = \lambda_i$ and $c = \gamma$, to give:

$$d_i = \frac{-\lambda_i \pm \sqrt{\lambda_i^2 + 4\gamma}}{-2} \quad (27)$$

$$= \frac{\lambda_i \pm \sqrt{\lambda_i^2 + 4\gamma}}{2} \quad (28)$$

[1] The transpose in (23) is present because we are using a notation where the derivative w.r.t a matrix has the same layout as the matrix itself, i.e. the i, j'th element is the derivative w.r.t. the i, j't element of the matrix. This equation, of course, highlights why that transpose can sometimes be convenient.

[2] Thanks to Hainan Xu and Desh Raj for working out some aspects of this.

and, taking into account that $\lambda_i > 0$ (which we require to ensure derivatives exist) and $\gamma > 0$, it is not hard to show that if we take the more positive solution it always gives us the best value of $\mathcal{L}$, so

$$d_i = \frac{\lambda_i + \sqrt{\lambda_i^2 + 4\gamma}}{2}. \quad (29)$$

We can view $\mathbf{B} = F(\mathbf{L})$ as an instance of the kind of singular-value-rescaling function described in [2], with the scalar function $f(\lambda) = \frac{\lambda+\sqrt{\lambda^2+4\gamma}}{2}$ operating on the singular values. In fact we can use that same framework to backprop through the function $\mathbf{G} \rightarrow \mathbf{G}^{-\frac{1}{2}}$, since for positive definite matrices we can compute powers straightforwardly using the singular value decomposition.

3.1. Requirements on inputs

In order for the expressions above to be well defined it is necessary for $\mathbf{G}$, which mathematically is positive semidefinite, to be strictly positive definite. This amounts to saying that the features for this speaker are not linearly dependent. In practice we will likely modify (13) by adding a small fixed term like $\epsilon\mathbf{I}$. This doesn't really interact with any of our formulas here so we won't have to discuss it further.

A trickier requirement is the requirement for $\mathbf{K}$ to have no more than one zero singular values. This is required in order for the derivatives of our solution $\mathbf{A}$ w.r.t. the inputs to be well defined; in [2] you will see some discussion of this issue. However, $\mathbf{K}$ doesn't have the kind of structure that would allow us to ensure that its singular values are positive by adding a fixed matrix. We do have an expectation that the diagonal elements of $\mathbf{K}$ will generally be positive, which amounts to an expectation that the model is "reasonable" (i.e. its means vary in the same direction as the features aligned to them)... but this is not a provable thing, and might not be true early in training. Anyway, we can likely ensure that the singular values of $\mathbf{K}$ are not too tiny in a "reactive" way, by, if we notice very small or zero singular values for a particular speaker, adding different constants times $\mathbf{I}$ to $\mathbf{K}$ until the singular values are large enough. Actually, the real problems arise when *more than one* singular values of $\mathbf{K}$ are zero; the derivatives are well defined if only one singular value is zero. Assuming there is a random component to those singular values, it's actually quite unlikely for more than one of them to be so close to zero as to cause a problem, so this may not really be an issue in practice.

3.2. Diagostics

A useful diagnostic in this type of computation is the amount of log-likelihood improvement per frame that we get from the fMLLR estimation. For instance, in speech tasks with conventional features, a value between 5 and 10 is typical. Bear in mind that the "default" transform parameters (corresponding to keeping the original, un-adapted features) is: $\mathbf{A} = \mathbf{I}$, $\mathbf{b} = \mathbf{0}$. We can separately compute the contribution to the change in log-likelihood from the estimation of $\mathbf{A}$ and of $\mathbf{b}$. The log-likelihood improvement from estimating $\mathbf{A}$ can be written as:

$$\begin{aligned}\mathcal{L}(\mathbf{A}) - \mathcal{L}(\mathbf{I}) &= \left(\gamma \log|\mathbf{A}| + \mathrm{tr}\left(\mathbf{A}^T\mathbf{K}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{A}\mathbf{G}\mathbf{A}^T\right)\right) \\ &\quad - \left(\gamma \log|\mathbf{I}| + \mathrm{tr}\left(\mathbf{I}\mathbf{K}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{I}\mathbf{G}\mathbf{I}\right)\right) \quad (30) \\ &= \gamma \log||\mathbf{A}|| + \mathrm{tr}\left(\mathbf{A}^T\mathbf{K}\right) - \mathrm{tr}\left(\mathbf{K}\right) \\ &\quad + \frac{1}{2}\mathrm{tr}\left(\mathbf{G}\right) - \frac{1}{2}\mathrm{tr}\left(\mathbf{B}\mathbf{B}^T\right) \quad (31)\end{aligned}$$

where $||\cdot||$ represents the absolute value of the determinant; and notice that

$$\begin{aligned}\log||\mathbf{A}|| &= \log||\mathbf{B}\mathbf{G}^{-0.5}|| \\ &= \log|\mathbf{B}| + \log||\mathbf{G}^{-0.5}||, \quad (32)\end{aligned}$$

which is convenient because those log-determinants will be easy to obtain from the forward computation without extra work. The quantity $\mathrm{tr}\left(\mathbf{B}\mathbf{B}^T\right)$ can be computed as the sum of squared singular values of $\mathbf{B}$, which is also efficient to compute.

The objective function improvement from our estimation of $\mathbf{b}$ equals $0.5\hat{\gamma}\mathbf{b}\mathbf{b}^T$.

4. SUMMARY

We will now summarize what we have derived above, using a format that will be convenient for both explaining the computation and for computing derivatives for manual backprop.

4.1. Inputs

The inputs to the computation are:

- The unadapted features $\mathbf{x}_t \in \Re^D$, for $t = 1 \ldots T$
- The model means $\boldsymbol{\mu}_i \in \Re^D$, for $i = 1 \ldots I$
- The scalar variance factors $s_i > 0$ (the actual variances are $s_i\mathbf{I}$).
- The class posteriors $\gamma_{t,i}$.

The output of the forward computation will be the adapted features $\mathbf{y}_t$.

When we do the backprop we'll propagate the derivatives back to the first three inputs, but we'll treat the posteriors $\gamma_{t,i}$ as constants (the justification for this gets complicated and is beyond the scope of this document).

4.2. Computation

We define "modified posteriors" $\hat{\gamma}_{t,i}$, and total-counts as follows. We define various sums over the $\gamma_{t,i}$ and $\hat{\gamma}_{t,i}$, as follows:

$$\gamma_i = \sum_t \gamma_{t,i} \quad (33)$$

$$\gamma = \sum_i \gamma_i \quad (34)$$

$$\hat{\gamma}_{t,i} = \frac{\gamma_{t,i}}{s_i} \quad (35)$$

$$\hat{\gamma}_t = \sum_i \hat{\gamma}_{t,i} \quad (36)$$

$$\hat{\gamma} = \sum_i \frac{\gamma_i}{s_i} \quad (37)$$

then we define weighted sums of input features per class as follows (these are part of a more efficient method to compute $\mathbf{K}$):

$$\mathbf{z}_i = \sum_t \gamma_{t,i}\mathbf{x}_t \quad (38)$$

then we can compute the mean, and input, weighted averages, as follows:

$$\mathbf{m} = \frac{1}{\hat{\gamma}}\sum_i \frac{\gamma_i}{s_i}\boldsymbol{\mu}_i \quad (39)$$

$$\mathbf{n} = \frac{1}{\hat{\gamma}}\sum_i \frac{1}{s_i}\mathbf{z}_i. \quad (40)$$

then we compute the sufficient statistics for estimating $\mathbf{A}$ as follows (these have been rearranged slightly for efficiency and convenience):

$$\mathbf{G} = \left(\sum_t \hat{\gamma}_t \mathbf{x}_t \mathbf{x}_t^T\right) - \hat{\gamma}\mathbf{n}\mathbf{n}^T \quad (41)$$

$$\mathbf{K} = \left(\sum_i \frac{1}{s_i}\boldsymbol{\mu}_i \mathbf{z}_i^T\right) - \hat{\gamma}\mathbf{m}\mathbf{n}^T \quad (42)$$

(and we make it a precondition that $\mathbf{G}$ be positive definite and $\mathbf{K}$ have nonzero singular values; we may have to modify these values somehow in order to ensure this); then we compute:

$$\mathbf{H} = \mathbf{G}^{-0.5}. \quad (43)$$

We can use the same code framework for backpropping through SVD, as described in [2], to compute, and to backprop through, this matrix power. Next, compute the preconditioned linear statistic:

$$\mathbf{L} = \mathbf{K}\mathbf{H} \quad (44)$$

and then compute:

$$\mathbf{B} = F(\mathbf{L}) \quad (45)$$

where $F(\cdot)$ is the function that does the SVD on $\mathbf{L}$, puts the singular values through the function $f(\lambda) = \frac{\lambda+\sqrt{\lambda^2+4\gamma}}{2}$, and reconstructs. We described in [2] how to backprop through this type of function.

We then compute the transform matrix

$$\mathbf{A} = \mathbf{B}\mathbf{H} \quad (46)$$

and the offset term

$$\mathbf{b} = \mathbf{m} - \mathbf{A}\mathbf{n} \quad (47)$$

and we can compute the adapted features as follows:

$$\mathbf{y}_t = \mathbf{A}\mathbf{x}_t + \mathbf{b}. \quad (48)$$

In an implementation it may make sense to, at this point, compute the diagnostics described in Section 3.2.

4.3. Backprop

We'll now write down the equations for the backprop through this computation. Of course, in an automatic differentiation framework this text would be redundant.

We'll use a notation where for, say, a matrix $\mathbf{X}$, the derivative of our scalar function w.r.t. $\mathbf{X}$ is written as $\bar{\mathbf{X}}$, with each element being the derivative w.r.t. the corresponding element of $\mathbf{X}$ (i.e. there is no transpose); and the same notation for vectors and scalars.

We'll suppose that the user has supplied the derivatives $\bar{\mathbf{y}}_t$, and our task is to compute the derivatives w.r.t. the inputs $\mathbf{x}_t$, $\boldsymbol{\mu}_i$ and s_i.

First we backprop through (48):

$$\bar{\mathbf{x}}_t \leftarrow \mathbf{A}^T\bar{\mathbf{y}}_t \quad (49)$$

$$\bar{\mathbf{A}} \leftarrow \sum_t \bar{\mathbf{y}}_t \mathbf{x}_t^T \quad (50)$$

$$\bar{\mathbf{b}} = \sum_t \bar{\mathbf{y}}_t \quad (51)$$

$$(52)$$

(where we use $\leftarrow$ here in some cases instead of $=$ to indicate that we're treating something as a variable because it is at some point modified by adding extra terms). We then backprop through (47):

$$\bar{\mathbf{m}} \leftarrow \bar{\mathbf{b}} \quad (53)$$

$$\bar{\mathbf{A}} \leftarrow \bar{\mathbf{A}} - \bar{\mathbf{b}}\mathbf{n}^T \quad (54)$$

$$\bar{\mathbf{n}} \leftarrow -\mathbf{A}^T\bar{\mathbf{b}} \quad (55)$$

then through (46):

$$\bar{\mathbf{B}} = \bar{\mathbf{A}}\mathbf{H}^T \quad (56)$$

$$\bar{\mathbf{H}} \leftarrow \mathbf{B}^T\bar{\mathbf{A}} \quad (57)$$

Then we use the methods described in the separate writeup titled "Notes on derivatives of SVD" to backprop through (45) and compute $\bar{\mathbf{L}}$ from $\bar{\mathbf{B}}$ (that procedure also requires the value of $\mathbf{B}$ as input). We now backprop through (44):

$$\bar{\mathbf{K}} = \bar{\mathbf{L}}\mathbf{H}^T \quad (58)$$

$$\bar{\mathbf{H}} \leftarrow \bar{\mathbf{H}} + \mathbf{K}^T\bar{\mathbf{L}} \quad (59)$$

We next backprop through (43) (which sets $\mathbf{H}$ to $\mathbf{G}^{-0.5}$), using the same approach described in that separate writeup. We can view that matrix power as an SVD-rescaling operation with the function $f(\lambda) = \lambda^{-0.5}$; this only works for strictly positive definite $\mathbf{G}$, but we require $\mathbf{G}$ to be positive definite anyway. This backprop gives us the derivative $\bar{\mathbf{G}}$, which we will make sure to symmetrize $\bar{\mathbf{G}}$ (i.e., set $\bar{\mathbf{G}} \leftarrow \frac{1}{2}(\bar{\mathbf{G}} + \bar{\mathbf{G}}^T)$), just so that we can make the assumption that it is symmetric while manipulating quantities involving it; this simplifies some expressions.

We next backprop through (41), giving us:

$$\bar{\mathbf{x}}_t \leftarrow \bar{\mathbf{x}}_t + 2\hat{\gamma}_t\bar{\mathbf{G}}\mathbf{x}_t \quad (60)$$

$$\bar{\hat{\gamma}}_t = \mathbf{x}_t^T\bar{\mathbf{G}}\mathbf{x}_t \quad (61)$$

$$\bar{\mathbf{n}} \leftarrow \bar{\mathbf{n}} - 2\hat{\gamma}\bar{\mathbf{G}}\mathbf{n} \quad (62)$$

$$\bar{\hat{\gamma}} \leftarrow -\mathbf{n}^T\bar{\mathbf{G}}\mathbf{n} \quad (63)$$

and through (42):

$$\bar{\boldsymbol{\mu}}_i \leftarrow \frac{1}{s_i}\bar{\mathbf{K}}\mathbf{z}_i \quad (64)$$

$$\bar{\mathbf{z}}_i = \frac{1}{s_i}\bar{\mathbf{K}}^T\boldsymbol{\mu}_i \quad (65)$$

$$\bar{\hat{\gamma}} \leftarrow \bar{\hat{\gamma}} - \mathbf{m}^T\bar{\mathbf{K}}\mathbf{n} \quad (66)$$

$$\bar{s}_i \leftarrow \frac{-1}{s_i^2}\boldsymbol{\mu}_i^T\bar{\mathbf{K}}\mathbf{z}_i \quad (67)$$

$$\bar{\mathbf{n}} \leftarrow \bar{\mathbf{n}} - \hat{\gamma}\bar{\mathbf{K}}^T\mathbf{m} \quad (68)$$

$$\bar{\mathbf{m}} \leftarrow \bar{\mathbf{m}} - \hat{\gamma}\bar{\mathbf{K}}\mathbf{n} \quad (69)$$

through (40):

$$\bar{\mathbf{z}}_i \leftarrow \bar{\mathbf{z}}_i + \frac{1}{s_i\hat{\gamma}}\bar{\mathbf{n}} \quad (70)$$

$$\bar{s}_i \leftarrow \bar{s}_i - \frac{1}{s_i^2\hat{\gamma}}\mathbf{z}_i^T\bar{\mathbf{n}} \quad (71)$$

$$\bar{\hat{\gamma}} \leftarrow \sum_i \frac{-\hat{\gamma}_t}{\hat{\gamma}^2}\frac{1}{s_i}\mathbf{z}_i^T\bar{\mathbf{n}} = \frac{-1}{\hat{\gamma}}\mathbf{n}^T\bar{\mathbf{n}} \quad (72)$$

through (39):

$$\bar{\boldsymbol{\mu}}_i \leftarrow \bar{\boldsymbol{\mu}}_i + \frac{\gamma_i}{s_i\hat{\gamma}}\bar{\mathbf{m}} \quad (73)$$

$$\bar{s}_i \leftarrow \bar{s}_i - \frac{\gamma_i}{s_i^2\hat{\gamma}}\boldsymbol{\mu}_i^T\bar{\mathbf{m}} \quad (74)$$

$$\bar{\hat{\gamma}} \leftarrow \bar{\hat{\gamma}} - \sum_i \frac{\gamma_i}{s_i\hat{\gamma}^2}\boldsymbol{\mu}_i^T\bar{\mathbf{m}} = \bar{\hat{\gamma}} - \frac{1}{\hat{\gamma}}\mathbf{m}^T\bar{\mathbf{m}} \quad (75)$$

through (38):

$$\bar{\mathbf{x}}_t \leftarrow \bar{\mathbf{x}}_t + \sum_i \gamma_{t,i} \bar{\mathbf{z}}_i \tag{76}$$

through (37), (36) and (35):

$$\bar{s}_i \leftarrow \bar{s}_i - \frac{\gamma_i}{s_i^2} \bar{\tilde{\gamma}} \tag{77}$$

$$\bar{\tilde{\gamma}}_{t,i} = \bar{\tilde{\gamma}}_t \tag{78}$$

$$\bar{s}_i \leftarrow \bar{s}_i - \frac{1}{s_i^2} \sum_t \bar{\tilde{\gamma}}_{t,i}. \tag{79}$$

The outputs of this backprop phase of the computation are $\bar{\mathbf{x}}_t$, $\bar{\boldsymbol{\mu}}_i$ and $\bar{s}_i$.

5. REFERENCES

[1] M. Gales, "Maximum Likelihood Linear Transformations for HMM-based Speech Recognition," *Computer Speech and Language*, vol. 12, 1998.

[2] D. Povey, "Notes on derivatives of SVD." [Online]. Available: http://www.danielpovey.com/files/2018_svd_derivative.pdf