Growth Transform of A Sum of Rational Functions and Its Application in Estimating HMM Parameters

Xiaoqiang Luo
Center for Language and Speech Processing
Department of Electrical & Computer Engineering
The Johns Hopkins University
Baltimore, MD21218, USA

ABSTRACT

Gopalakrishnan *et al* [1] described a method called "growth transform" to optimize rational functions over a domain, which has been found useful to train discriminatively Hidden Markov Models(HMM) in speech recognition [5, 6, 9]. A sum of rational functions is encountered when the contributions from other HMM states are weighted in estimating Gaussian parameters of a state, and the weights are optimized using cross-validation [8]. We will show that the growth transform of a sum of rational functions can be obtained by computing term-wise gradients and term-wise function values, as opposed to forming first a single rational function and then applying the result in [1]. This is computationally advantageous when the objective function consists of many rational terms and the dimensionality of the domain is high. We also propose a gradient directed search algorithm to find the appropriate transform constant C.

1 Introduction

Baum and Eagon [2] studied the problem of maximizing homogeneous polynomials with nonnegative coefficients over a domain, or a set of probability mass functions. Such problem arises, for instance, when maximum likelihood estimate (MLE) is sought for discrete Hidden Markov Models (HMM). Later on, Baum's result was extended to rational functions with positive denominators over a domain by Gopalakrishnan and his colleagues[1]. Maximizing rational functions over a domain occurs in the context of discriminative training of HMM [5, 6, 9, 10].

In this paper, we study the growth transform for a sum of rational functions over a domain, that is, an objective function $R(x)$ of the form

$$R(x) = \sum_{i=1}^{M} \frac{P_i(x)}{Q_i(x)} \tag{1}$$

where $P_i(x)$ and $Q_i(x)$ are polynomials. x takes value in the domain

$$D = \{y : \sum_{j=1}^{q_i} y_{ij} = 1, y_{ij} \geq 0, i = 1, 2, \cdots, p\}, \tag{2}$$

where y_{ij} is the ij^{th} component of vector y (think of y as a double-indexed vector), and $Q_i(x) > 0 (i = 1, 2, \cdots, M)$ for all $x \in D$.

A sum of rational functions is encountered when the contributions from other HMM states are weighted in estimating Gaussian parameters of a state, and the weights are optimized using cross-validation [7, 8]. To simplify the presentation, we assume that Gaussian covariance matrices $\{\Sigma_i\}$ are known (they are estimated as well in reality), and Gaussian means are estimated as follows.

$$\mu_i = \frac{\sum_j x_{ij}\gamma_j\hat{\mu}_j}{\sum_j x_{ij}\gamma_j} \tag{3}$$

where i and j are indices for HMM states, x_{ij} is the factor that weights the contribution from j, γ_j is the occupancy count of j, and $\hat{\mu}_j$ is the maximum likelihood estimate of Gaussian mean of j. For positive $\{x_{ij}\}$, we can always normalize them so that $\sum_j x_{ij} = 1$. To estimate $\{x_{ij}\}$, assume that we have a held-out set of training data $\{o_t\}$ which is independent of data used to estimate $\{\hat{\mu}_j\}$. Then the "quasi" likelihood of the held-out data measured by $\{\mu_i, \Sigma_i\}$ is

$$L(\mathbf{x}) = -\frac{1}{2}\sum_i \sum_t \eta_i(t)(o_t - \mu_i)'\Sigma_i^{-1}(o_t - \mu_i) \tag{4}$$

where $\mathbf{x}$ is the totality of $\{x_{ij}\}$, $\eta_i(t)$ is the occupancy count for i based on the held-out data $\{o_t\}$. Plug (3) into (4), and it is easy to see, after ignoring the term independent of $\mathbf{x}$, that $L(\mathbf{x})$ is of the form

$$L(\mathbf{x}) = -\frac{1}{2}\sum_i \frac{x_i' A_i x_i}{x_i' B_i x_i} \tag{5}$$

where A_i and B_i are matrices, and x_i is a vector whose j^{th} element is x_{ij}.

In the above, it is assumed that training data is split into two parts and a single Gaussian is used as the state-output distribution. If P-fold cross-validation is employed and a mixture of Gaussians is used, then since A_i and B_i depend on the partition index p and mixture label l, (5) will become [8]

$$L(\mathbf{x}) = -\frac{1}{2}\sum_i [\sum_p \sum_l \frac{x_i' A_i(p,l) x_i}{x_i' B_i(p,l) x_i}] \tag{6}$$

Therefore, optimizing each x_i boils down to optimizing a sum of rational functions (i.e, terms in the square bracket of (6)) subject to the constraints $\sum_j x_{ij} = 1, x_{ij} \geq 0$. Since (6) is a special case of (1) we will focus on optimizing (1) in this paper.

The rest of the paper is organized as follows. In Section 2 we briefly restate Gopalakrishnan's result [1] without proof. Then we show that the growth transform for $R(x)$ in (1) can be obtained by computing only term-wise gradients and function values. A gradient directed search for proper C (a necessary constant in the growth transform) is presented in Section 3. This is motivated by the fact that a fixed C may lead to either non-growth transform or slow convergence and therefore it is difficult to pre-determine an appropriate C. The paper ends with the conclusion remarks in Section 4.

2 Growth Transform for A Sum of Rational Functions

We first restate the main result in [1], where optimizing a single rational function is considered. That is, there is only one term in (1), $R(x) = \frac{P(x)}{Q(x)}$, where $P(x)$ and $Q(x)$ are polynomials

and $Q(x) > 0$ for all $x \in D$. It is shown [1] that a map from D to D can be constructed such that the objective function does not decrease – hence the name "growth transform". For any $\alpha \in D$, define

$$S_\alpha(x) \quad = \quad P(x) - R(\alpha)Q(x). \tag{7}$$

Let d_α be the degree of $S_\alpha(x)$ and a_α be the smallest (negative) coefficient of S_α, or 0 if all coefficients are nonnegative. Also Let

$$C_\alpha \quad = \quad -a_\alpha d_\alpha (p+1)^{d_\alpha - 1}, \tag{8}$$

$$C_0 \quad = \quad \max_{\alpha \in D} C_\alpha. \tag{9}$$

Notice that maximum in (9) is achievable since D is compact. Then for any $C_i \geq C_0 (i = 1, 2, \cdots, p)$, $\hat{\alpha} = T[\alpha]$ defined by

$$\hat{\alpha}_{ij} \quad = \quad \frac{\alpha_{ij}(\frac{\partial S_\alpha(\alpha)}{\partial x_{ij}} + C_i)}{\sum_{j=1}^{q_i} \alpha_{ij}(\frac{\partial S_\alpha(\alpha)}{\partial x_{ij}} + C_i)} \quad \forall j = 1, \cdots q_i;\ i = 1, \cdots, p. \tag{10}$$

is a *growth transform*, namely, $R(\hat{\alpha}) \geq R(\alpha)$, where $\hat{\alpha}_{ij}$ and α_{ij} are the ij^{th} element of $\hat{\alpha}$ and α respectively.

An iterative algorithm can be easily devised by repeating the transform (10) to obtain a local maximum of $R(x)$. It is worth of pointing out that C_i has to do with how fast the iterative algorithm converges. The larger C_i is, the slower the algorithm converges. Indeed, it can be seen from (10) that as $C_i \to +\infty (\forall i)$, $\hat{\alpha} \to \alpha$. Therefore it is desirable to use small C_i provided that (10) is a growth transform. It is suggested in [1] that

$$C_i = C^*(\alpha) \quad = \quad \max\left\{ \max_{ij}\{-\frac{\partial S_\alpha(\alpha)}{\partial x_{ij}}\}, 0 \right\} + \epsilon \ \ \forall i \tag{11}$$

is used, where ϵ is a preselected "small" number. However, the drawback of such $C^*(\alpha)$ is that it is no longer guaranteed that (10) is a growth transform. We will get back to this issue in Section 3 and propose a gradient directed search algorithm for a proper $C^*(\alpha)$ after presenting our main result in the following.

Since

$$R(x) \quad = \quad \sum_{i=1}^{M} \frac{P_i(x)}{Q_i(x)} = \frac{\sum_{i=1}^{M} \left(P_i(x) \prod_{j \neq i} Q_j(x)\right)}{\prod_{i=1}^{M} Q_i(x)}, \tag{12}$$

if we define

$$G(x) := \sum_{i=1}^{M} \left(P_i(x) \prod_{j \neq i} Q_j(x)\right), \quad H(x) := \prod_{i=1}^{M} Q_i(x), \tag{13}$$

then $R(x) = \frac{G(x)}{H(x)}$.

So $R(x)$ in (1) can conceptually be converted to a single rational function and then we can apply (10). Such straightforward implementation, however, may be computationally prohibitive if the dimensionality of x is high and M is large. To see this, notice that both $G(x)$ and $H(x)$ are polynomials, and that the degree of $H(x)$ is Md if each $Q_i(x)$ is of degree d. So $H(x)$ can have as many as $\binom{Md+n-1}{n-1}$ terms, where n is the dimension of x. When

M and n get large, this number becomes large rapidly. For our problem of estimating the data-sharing weights [7], $M = 72, d = 2$ and typically n is about 10, so $\binom{Md+n-1}{n-1} \approx 10^{14}$! Therefore, it is rather awkward to compute the growth transform by forming explicitly $G(x)$ and $H(x)$. Fortunately, this is avoidable, as we show now.

For $\alpha \in D$, redefine $S_\alpha(x) = G(x) - R(\alpha)H(x)$. To apply (10), we need to calculate $\nabla S_\alpha(\alpha)$, the gradient of $S_\alpha(x)$ evaluated at $x = \alpha$. Since

$$\begin{aligned} \nabla S_\alpha(x) &= \nabla G(x) - R(\alpha)\nabla H(x) && (14) \\ &= \nabla R(x)H(x) + R(x)\nabla H(x) - R(\alpha)\nabla H(x) && (15) \end{aligned}$$

we have

$$\nabla S_\alpha(\alpha) = \big(\nabla R(x)\big)|_{x=\alpha} H(\alpha) = \Big(\sum_{i=1}^{M} \frac{\nabla P_i(\alpha)Q_i(\alpha) - P_i(\alpha)\nabla Q_i(\alpha)}{Q_i^2(\alpha)}\Big) \prod_{i=1}^{M} Q_i(\alpha). \qquad (16)$$

This shows that $\nabla S_\alpha(\alpha)$ can be expressed as a function of $\{P_i(\alpha), Q_i(\alpha)\}$ and their gradients $\{\nabla P_i(x), \nabla Q_i(x)\}$ evaluated at $x = \alpha$. Therefore, the growth transform can be obtained by only calculating term-wise gradients and function values. When the number of terms M is large and the dimension of x is high, this is preferable than carrying out the growth transform directly on $G(x)$ and $H(x)$.

3 Gradient Directed Search for Proper C

As mentioned in Section 2, to speed up the convergence of the algorithm, we want $\{C_i\}$ as small as possible, provided that the growing nature, $R(\hat{x}) \geq R(x)$ is maintained for all transforms. The bound C_0 defined in (9) is of little use in implementing the growth transform because, first, computing C_0 itself is nontrivial. Especially in our case where we do not want to form $G(x)$ and $H(x)$ explicitly, and therefore a_α, the smallest (negative) coefficients of $S_\alpha(x)$, is unavailable; Second, C_0 is a bound that assures (10) is a growth transform starting from any $\alpha \in D$. It is likely to be too large, and consequently, the convergence will be slow if $C_i = C_0$ is used. The heuristic (11) proposed in [1] may speed up the convergence of a growth transform. However, its drawback is that it may result in a "decrease" transform. So here we propose a gradient-directed search for $\{C_i\}$.

One observation about (10) is that scaling the gradient $\nabla S_\alpha(\alpha)$ by a positive constant does not change the transform since it can be absorbed in the constants C_i. Let

$$\begin{aligned} \beta_\alpha &= \max_{ij} \Big\{ |\frac{\partial S_\alpha(\alpha)}{\partial x_{ij}}| \Big\} && (17) \\ g_{ij}(\alpha) &= \frac{1}{\beta_\alpha} \frac{\partial S_\alpha(\alpha)}{\partial x_{ij}} && (18) \\ C_i(\alpha) &= \frac{1}{\beta_\alpha} C_i, && (19) \end{aligned}$$

In the above we have assumed $\beta_\alpha > 0$ (otherwise, $\nabla S_\alpha(\alpha) = 0$, so (10) implies $\hat{\alpha} = \alpha$, which is a trivial case). Note that $|g_{ij}(\alpha)| \leq 1$. With these notations, (10) can be written as

$$\hat{\alpha}_{ij} = \frac{\alpha_{ij}\Big(g_{ij}(\alpha) + C_i(\alpha)\Big)}{\sum_{k=1}^{q_i} \alpha_{ik}\Big(g_{ik}(\alpha) + C_i(\alpha)\Big)}. \qquad (20)$$

In other words, we can always normalize the gradient components of $\nabla S_\alpha(\alpha)$ so that the normalized values are in $[-1, 1]$.

The second observation is that, roughly speaking, small $C_i(\alpha)$ allows the transform (20) to make big moves while large $C_i(\alpha)$ will confine the transformed $\hat{\alpha}$ to a neighborhood of α. Therefore, we hope to determine an interval from which $C_i(\alpha)$ takes values. Let s and L be two positive numbers such that $s \ll 1$ and $L \gg 1$, and define

$$p_\alpha = \max\left\{ \max_{ij}\{-g_{ij}(\alpha)\}, 0 \right\} \tag{21}$$

$$m_\alpha = \min_{ij}\left\{ |g_{ij}(\alpha)| \right\} \tag{22}$$

$$M_\alpha = \max_{ij}\left\{ |g_{ij}(\alpha)| \right\} \tag{23}$$

$$I_\alpha = [sm_\alpha, LM_\alpha]. \tag{24}$$

We propose to use

$$C_i(\alpha) = p_\alpha + \epsilon_\alpha \tag{25}$$

for $C_i(\alpha)$ in (20) and $\epsilon_\alpha \in I_\alpha$, and we will search proper ϵ_α in I_α. p_α is necessary since it assures that (20) is always admissible, or $\hat{\alpha} \in D$. Since $s \ll 1$ and $m_\alpha \leq |g_{ij}(\alpha)|$, when $\epsilon_\alpha = sm_\alpha \ll |g_{ij}(\alpha)|$, ϵ_α has little impact on the transform (20) so it allows the algorithm to converge fast if it indeed yields a growth transform; On the other hand, when $\epsilon_\alpha = LM_\alpha$,

$$|g_{ij}(\alpha) + p_\alpha + LM_\alpha| = LM_\alpha \left| \frac{g_{ij}(\alpha)}{LM_\alpha} + \frac{p_\alpha}{LM_\alpha} + 1 \right| \approx LM_\alpha \tag{26}$$

since $L \gg 1$, $|g_{ij}(\alpha)| \leq M_\alpha$ and $0 \leq p_\alpha \leq M_\alpha$. Note that (26) implies $\hat{\alpha} \approx \alpha$.

To test the merit of an ϵ_α, we have to carry out (20) and then compute $R(\hat{\alpha})$, the objective at $\hat{\alpha}$. This should be avoided whenever possible if evaluating $R(\hat{\alpha})$ itself is costly. So a reasonable strategy would be that we first test whether a small ϵ_α leads to a growth transform. If yes, we will accept it; Otherwise, the search for $\epsilon_\alpha \in I_\alpha$ is triggered and we will pick the best ϵ_α to carry out the transform (20). Should the search fail to find an ϵ_α that generates a growth transform, we will declare α as a local maximum and terminate the algorithm.

What is appealing is that it is easy to determine I_α when $\nabla S_\alpha(\alpha)$ is available, and s and L are constants independent of the current x. If L is large enough, it is likely we can find an ϵ_α that leads to a growth transform unless α is a local maximum. We will show some examples when the paper is presented.

4 Conclusions

In this paper we have studied two practical problems in carrying out the growth transform for a sum of rational functions. we have shown that the growth transform for a sum of rational functions can be obtained by calculating the term-wise gradients and function values. When the number of terms is large and the dimension of the domain in question is high, computing the term-wise quantities is much more efficient than first forming a single numerator polynomial and a single denominator polynomial and then applying the result in [1].

Motivated by the fact that the transform constant C has great effect on the speed of convergence and it is difficult to choose an appropriate value in practice, we propose a gradient directed search algorithm to find proper C. The proposed algorithm works better, in terms

of both finding better optimal points and computation time, than the heuristic 11. Examples will be shown when the paper is presented.

For the application of the growth transform to estimating HMM parameters and the results, readers are referred to [7].

5 Acknowledgment

The author would like to thank Mr. Vaibhava Goel for many useful discussions and suggestions for improving the paper.

References

[1] P. S. Gopalakrishnan, D. Kanevsky, Arthur Nadas, and David Nahamoo, "An inequality for rational functions with applications to some statistical estimation problems," *IEEE Trans. on Information Theory*, vol. 37, no. 1, pp. 107–113, 1991.

[2] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology," *Bull. Amer. Math. Soc.*, vol. 73, pp. 360–363, 1967.

[3] L.R Bahl, F.Jelinek, and R.L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 2, pp. 179–190, 1983.

[4] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[5] V. Valtchev, P.C Woodland, and S.J Young, "Discriminative optimization of large vocabulary recognition systems," in *Proc. Inter. Conf. of Spoken Language Processing*, 1996, pp. I18–21.

[6] L. R. Bahl, M. Padmananhan, D. Nahamoo, and P.S Gopalakrishnan, "Discriminative training of Gaussian mixture models for large vocabulary speech recognition systems," in *Proc. of ICASSP*, 1996, pp. 613–617.

[7] Xiaoqiang Luo, and Frederick Jelinek, "Nonreciprocal Data Sharing in Estimating HMM Parameters," in *Proc. ICSLP*, 1998

[8] Xiaoqiang Luo, and Frederick Jelinek, "Nonreciprocal Data Sharing in Estimating HMM Parameters," in *CLSP Research Notes No. 32, The Johns Hopkins University*, 1998

[9] Y. Normandin, *Hidden Markov Models, Maximum Mutual Information Estimation, and the Speech Recognition Problem*, Ph.D thesis, Department of Electrical Engineering, McGill University, 1991.

[10] Yves Normandin, "Optimal splitting of HMM Gaussian mixture components with MMIE training," in *Proc. ICASSP*, 1995, pp. I–449–452.