OPTIMAL LEARNING OF P-LAYER ADDITIVE F0 MODELS WITH CROSS-VALIDATION

Shinsuke Sakai[1,2,3], *Tatsuya Kawahara*[3,1], *Tohru Shimizu*[1,2], *Satoshi Nakamura*[1,2]

[1] National Institute of Information and Communications Technology, Japan
[2] ATR Spoken Language Communication Research Labs, Japan
[3] School of informatics, Kyoto University, Japan

ABSTRACT

In this paper, we present the derivation of the *backfitting* training algorithms for generic p-layer additive F_0 models for arbitrary positive integer p. We have presented the special cases of the algorithms with $p = 2$ and $p = 3$ that have been successfully applied to the modelings of Japanese and English F_0 contours, whereas the derivation of the algorithm was presented only for the two-layer case. The additive F_0 model have smoothing parameters that establish a trade-off between the fit to the training data and the smoothness of the fitted curves, which have been all set to unity in the previous works. In this paper, we also present an optimal approach to set the values of these parameters using cross validation. We performed the training using the Boston University Radio News Corpus and confirmed the effectiveness of the proposed method.

Index Terms— speech synthesis, fundamental frequency, additive models, statistical learning, intonation modeling

1. INTRODUCTION

Corpus-based approach to speech synthesis has been widely explored in the research community in recent years [1, 2, 3]. Intonation modeling, or generation of fundamental frequency (F_0) contour plays a crucial role in synthesizing natural sounding speech from input text. We previously proposed a framework of F_0 modeling using *Additive Models* [4] and successfully applied it to the modeling of F_0 contour of Japanese and English speech [5, 6, 7]. The Japanese F_0 model is a two-layer additive models consisting of the intonational phrase components and the accentual phrase components, whereas the English model consists of three layers, namely, the phrase components, the word components, and the pitch accent components. The additive F_0 modeling approach has advantages over conventional techniques in that, for example, the model training from the corpus is done by a simple and straightforward procedure without any preprocessing, once the set of features to identify component functions are determined. It also has a good property that the model output is a curve, rather than a sequence of constants, which is convenient for the use in runtime synthesis requiring no significant postprocessing before use. Furthermore, since it is a nonparametric model with no specific functional form, it is natural to expect that it is applicable to a wide variety of languages in addition to Japanese and English.

Although the additive F_0 modeling approach may be applicable to a wide variety of languages where the number of additive layers p can be larger than that for Japanese, we have so far presented the derivation of the backfitting training algorithms from the error criterion only for the two-layer case [5, 6]. In this paper, therefore, we present the derivation of the general backfitting training algorithms for p-layer additive F_0 models, where p is an arbitrary natural number.

The additive F_0 models have smoothing parameters that establish a trade-off between the fit to the training data and the smoothness of the fitted curves. They have been all set to unity in our previous works on Japanese and English. Although the experimental results have so far been satisfactory, we can expect an even better results or feel more secure if these hyper-parameters are optimized using the training data. In this paper, therefore, we propose an optimal approach to set the values of these parameters using 10-fold cross-validation [4].

In the next section, we derive a training algorithm for p-layer additive F_0 models from the regularized least-squares error criterion. We then describe the experiment where we optimize the smoothing parameters with cross-validation, followed by the conclusion.

2. BACKFITTING ALGORITHM FOR P-LAYER ADDITIVE F_0 MODELS

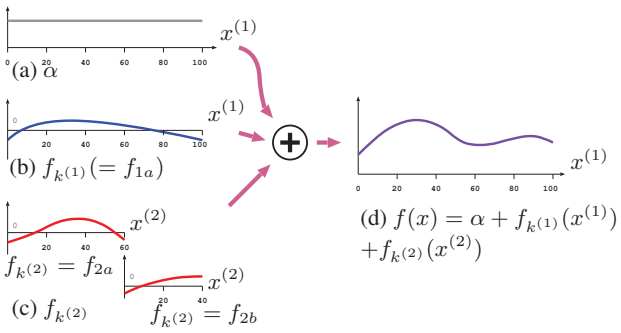

Fig. 1. Schematic diagram of an instance of a two-layered version of additive F_0 model, $f(x) = \alpha + f_{k^{(1)}}(x^{(1)}) + f_{k^{(2)}}(x^{(2)})$. In this example, $k^{(1)}$ continues to be of (hypothetical) type "1a", while $k^{(2)}$ equals "2a" during $0 \leq x^{(1)} \leq 60$, but changes to "2b" at $x^{(1)} = 60$.

In this section, we sketch the derivation of the *backfitting* training algorithm for p-layered additive F_0 models from the regularized least-squares error criterion, where p is an arbitrary natural number. It deals with the generic case that includes the two-layered and three-layered models applied for modeling Japanese and English F_0 [6, 7] as special cases.

The F_0 contour is modeled as a scalar random variable Y that has a functional dependence $Y = f(\boldsymbol{z}) + \epsilon$ on the vector of input variables $\boldsymbol{z} = (k^{(1)}, x^{(1)}, ..., k^{(p)}, x^{(p)})$. The error ϵ is assumed to be normally distributed with mean 0 and variance σ^2. The function $f(\boldsymbol{z})$ has the form of the sum of p component functions and a constant α,

$$f(\boldsymbol{z}) = \alpha + f_{k^{(1)}}(x^{(1)}) + \cdots + f_{k^{(p)}}(x^{(p)}), \quad (1)$$

where each of $k^{(m)}$ $(m = 1, ..., p)$ is an index variable that represents a type of F_0 component in the m-th layer, and it selects the specific F_0 component function $f_{k^{(m)}}$ for that type. Each of the continuous variables $x^{(m)}$ $(m = 1, \cdots, p)$ represents relative time that starts from zero when a new intonation component of type $k^{(m)} = \kappa$ starts at the m-th layer. We impose a simplifying assumption that syllables always have the same time length in the model. Therefore, before we start training, the raw F_0 data is time-normalized to have equal number of points per syllable to make it possible to learn the model from the speech corpus that originally have varying per-syllable durations. A schematic diagram of a two-layer additive F_0 model is shown in Fig 1. We will learn this regression model from the training data $\mathcal{D}$, a collection of input-output pairs $\mathcal{D} = \{(\boldsymbol{z}_n,\ y_n) \mid n = 1, \cdots, N\}$, where $\boldsymbol{z}_n = (k_n^{(1)},\ x_n^{(1)},\ \cdots, k_n^{(p)},\ x_n^{(p)})$. Note that N is the total number of data points in the training data, where each data point represent a (normalized) time instance, and these N points may correspond to hundreds or thousands of utterances. The boundaries of linguistic units such as utterance, intonational phrase, or word are represented through the change of the value of $k^{(m)}$ and the reset of $x^{(m)}$ to zero for linguistic units at the m-th layer. The estimate of the component functions of f are obtained from this training data as the minimizer of a regularized loss functional L, which is defined as the sum of squared errors plus *roughness penalties* on all the component functions, s.t.,

$$L(f) = \sum_{n=1}^{N}(y_n - f(\boldsymbol{z}_n))^2 + \sum_{m=1}^{p} \lambda_m \sum_{\kappa \in \mathcal{R}(k^{(m)})} \int {f_\kappa}''(\xi^{(m)})^2 \mathrm{d}\xi^{(m)}, \tag{2}$$

where "$\mathcal{R}(v)$" stands for the *range*, i.e., the set of distinct values, of the variable v. The first term of (2) measures the closeness to the training data while the second term penalizes the curvatures of the component functions, and the smoothing parameters λ_m establish a trade-off between them. Large values of λ_m yield smoother curves while smaller values result in more fluctuation.

In the derivation of the training algorithm from this error criterion, we make use of p different ways of partitioning the original training data $\mathcal{D}$ in association with the p additive layers, in order to have a focus on each of the intonation component types at each layer. The partitioning of the training data at the μ-th layer is defined as

$$\mathcal{D} = \bigcup_{\kappa \in \mathcal{R}(k^{(\mu)})} \mathcal{D}_{(\mu,\kappa)}, \tag{3}$$

where each of the mutually exclusive partitions $\mathcal{D}_{(\mu,\kappa)}$ is a collection of data items $(\boldsymbol{z}, y) = (k^{(1)}, x^{(1)}, ..., k^{(p)}, x^{(p)}, y)$ from $\mathcal{D}$, whose component type is κ at the μ-th layer, i.e.

$$\mathcal{D}_{(\mu,\kappa)} = \{(k^{(1)}, x^{(1)}, ..., k^{(p)}, x^{(p)}, y) \mid k^{(\mu)} = \kappa\} \tag{4}$$

The first term of (2), the sum of squared errors, can be re-organized according to the partitioning at the μ-th layer as

$$\begin{aligned}
&\sum_{n=1}^{N} (y_n - f(\boldsymbol{z}_n))^2 \\
&\quad = \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} \sum_{(\bar{z},\bar{y}) \in \mathcal{D}_{(\mu,\kappa)}} \{\bar{y} - f(\bar{z})\}^2 \\
&\quad = \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} \sum_{(\bar{z},\bar{y}) \in \mathcal{D}_{(\mu,\kappa)}} \{\bar{y} - \alpha - \sum_{m=1}^{p} f_{\bar{k}^{(m)}}(\bar{x}^{(m)})\}^2
\end{aligned} \tag{5}$$

where we denote a training data point in a partition as $(\bar{z}, \bar{y}) = (\bar{k}^{(1)}, \bar{x}^{(1)}, \cdots, \bar{k}^{(p)}, \bar{x}^{(p)}, \bar{y})$.

Using the property that the integral of the square of the second derivative appearing in the second term of (2) is minimized by a class of functions called *natural cubic splines*, as we did in [6], we can now express each f_κ as a linear combination of natural cubic spline bases with knots at unique values of $\bar{x}^{(\mu)}$ in $\mathcal{D}_{(\mu,\kappa)}$,

$$f_\kappa(x) = \sum_{j=1}^{J_\kappa} N_\kappa^{(j)}(x)\ \theta_\kappa^{(j)}, \tag{6}$$

for $\kappa \in \mathcal{R}(k^{(\mu)})$ and $\mu = 1, \cdots, p$, where $N_\kappa^{(j)}(x)$ $(j = 1, \cdots, J_\kappa)$ are natural cubic spline bases, $\theta_\kappa^{(j)}$ are weighting parameters, and J_κ is the number of unique values of $\bar{x}^{(\mu)}$ in $\mathcal{D}_{(\mu,\kappa)}$ [6]. Using this basis expansion form, the sum of squared errors in (5) can be further rewritten as

$$\begin{aligned}
&\sum_{n=1}^{N}(y_n - f(\boldsymbol{z}_n))^2 \\
&\quad = \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} \sum_{(\bar{z},\bar{y}) \in \mathcal{D}_{(\mu,\kappa)}} \{\bar{y} - \alpha - \sum_{m=1}^{p} f_{\bar{k}^{(m)}}(\bar{x}^{(m)})\}^2 \\
&\quad = \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} \{(\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{m=1}^{p} \boldsymbol{f}_\kappa^{(m)})^T \cdot \\
&\qquad (\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{m=1}^{p} \boldsymbol{f}_\kappa^{(m)})\} \\
&\quad = \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} \{(\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{\substack{m=1 \\ m \neq \mu}}^{p} \boldsymbol{f}_\kappa^{(m)} - \boldsymbol{N}_\kappa \boldsymbol{\theta}_\kappa)^T \cdot \\
&\qquad (\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{\substack{m=1 \\ m \neq \mu}}^{p} \boldsymbol{f}_\kappa^{(m)} - \boldsymbol{N}_\kappa \boldsymbol{\theta}_\kappa)\},
\end{aligned} \tag{7}$$

where $\boldsymbol{y}_\kappa$ is a column vector of length I_κ consisting of all $\bar{y}$'s in $\mathcal{D}_{(\mu,\kappa)}$, and I_κ is the number of data points in the partition $\mathcal{D}_{(\mu,\kappa)}$. (Vectors will be all column vectors unless otherwise noted, hereafter.) Vector $\boldsymbol{\alpha}$ simply consists of I_κ α's. $\boldsymbol{f}_\kappa^{(m)}$ is a vector of $f_{\bar{k}^{(m)}}(\bar{x}^{(m)})$'s evaluated at each data point in $\mathcal{D}_{(\mu,\kappa)}$. $\boldsymbol{N}_\kappa$ is an $I_\kappa \times J_\kappa$ matrix and its i-th row represents the values of $N_\kappa^{(1)}(\bar{x}^{(\mu)}), \cdots, N_\kappa^{(J_\kappa)}(\bar{x}^{(\mu)})$ evaluated at the i-th data point in $\mathcal{D}_{(\mu,\kappa)}$. $\boldsymbol{\theta}_\kappa$ is a vector of all $\theta_\kappa^{(j)}$ for $j = 1, \cdots, J_\kappa$.

Using the basis expansion form, the second term, or roughness penalty term of (2) can be rewritten as

$$\begin{aligned}
&\sum_{m=1}^{p} \lambda_m \sum_{\kappa \in \mathcal{R}(k^{(m)})} \int {f_\kappa}''(\xi^{(m)})^2 \mathrm{d}\xi^{(m)} \\
&\quad = \sum_{m=1}^{p} \lambda_m \sum_{\kappa \in \mathcal{R}(k^{(m)})} \int \sum_{j=1}^{J_\kappa} \{{N_\kappa^{(j)}}''(\xi)\theta_\kappa^{(j)}\}^2 \mathrm{d}\xi^{(m)} \\
&\quad = \sum_{m=1}^{p} \lambda_m \sum_{\kappa \in \mathcal{R}(k^{(m)})} \boldsymbol{\theta}_\kappa^T \boldsymbol{\Omega}_{N_\kappa} \boldsymbol{\theta}_\kappa,
\end{aligned} \tag{8}$$

where $\boldsymbol{\Omega}_{N_\kappa}$ is a $J_\kappa \times J_\kappa$ matrix whose (p, q)-element is

$$\boldsymbol{\Omega}_{N_\kappa}(p, q) = \int {N_\kappa^{(p)}}''(\xi) {N_\kappa^{(q)}}''(\xi) \mathrm{d}\xi, \ \ 1 \leq p, q \leq J_\kappa. \tag{9}$$

Now the whole loss functional $L(f)$ in (2) is given by the sum of (7) and (8). We note that it is quadratic in every $\boldsymbol{\theta}_\kappa$ at all layers and take a partial derivative of $L(f)$ and set it to zero to obtain the equation for $\boldsymbol{\theta}_\kappa$ that minimizes $L(f)$ which is

$$\hat{\boldsymbol{\theta}}_\kappa = (\boldsymbol{N}_\kappa^T \boldsymbol{N}_\kappa + \lambda_\mu \boldsymbol{\Omega}_{N_\kappa})^{-1} \boldsymbol{N}_\kappa^T \cdot (\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{\substack{m=1 \\ m \neq \mu}}^{p} \hat{\boldsymbol{f}}_\kappa^{(m)}) \tag{10}$$

for all $\kappa \in \mathcal{R}(k^{(\mu)})$, for all $\mu = 1, \cdots, p$. This system consists of $\mathcal{J}$ equations with $\mathcal{J}$ unknowns, where

$$\mathcal{J} = \sum_{\mu=1}^{p} \sum_{\kappa \in \mathcal{R}(k^{(\mu)})} J_\kappa.$$

We note that unknowns $\theta_\kappa^{(j)}$ are not only included in $\hat{\boldsymbol{\theta}}_\kappa$ but also in $\hat{\boldsymbol{f}}_\kappa^{(m)}$ in the equation (10). Using the relationship $\boldsymbol{f}_\kappa = \boldsymbol{N}_\kappa \boldsymbol{\theta}_\kappa$, we can rewrite (10) to obtain the set of linear equations in $\hat{\boldsymbol{f}}_\kappa$,

$$\hat{\boldsymbol{f}}_\kappa = \boldsymbol{N}_\kappa (\boldsymbol{N}_\kappa^T \boldsymbol{N}_\kappa + \lambda_\mu \boldsymbol{\Omega}_{N_\kappa})^{-1} \boldsymbol{N}_\kappa^T \cdot (\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{\substack{m=1 \\ m \neq \mu}}^{p} \hat{\boldsymbol{f}}_\kappa^{(m)}), \tag{11}$$

for $\mu = 1, ..., p$ and $\kappa \in \mathcal{R}(k^{(\mu)})$ and we have shown that the component functions are obtained as the solution of this linear system.

This linear system is efficiently solved using a *backfitting* algorithm depicted in Fig 2, a repetitive method that combines *Gauss-Seidel* and *Jacobi*'s methods [8]. In the algorithm, we define a *smoother matrix* $\boldsymbol{S}_\kappa$ which is an initial part of the equation (11), i.e.,

$$\boldsymbol{S}_\kappa = \boldsymbol{N}_\kappa (\boldsymbol{N}_\kappa^T \boldsymbol{N}_\kappa + \lambda_\mu \boldsymbol{\Omega}_{N_\kappa})^{-1} \boldsymbol{N}_\kappa^T. \tag{12}$$

The algorithm starts by setting all $\hat{\boldsymbol{f}}_\kappa$ to zero and then proceed by repetitively updating $\hat{\boldsymbol{f}}_\kappa$ by the difference between the training data $\boldsymbol{y}_\kappa$ and the current estimate of the sum of additive components except $\hat{\boldsymbol{f}}_\kappa$, smoothed with $\boldsymbol{S}_\kappa$.

(1) **(Initialization)**

$$\hat{\alpha} = \frac{1}{N} \sum_{n=1}^{N} y_n,$$

$$\hat{\boldsymbol{f}}_\kappa = \boldsymbol{0}, \quad for\ all\ \kappa \in \mathcal{R}(k^{(\mu)})\ for\ all\ \mu = 1, \cdots, p.$$

(2) **(Cycle)**
Repeat the following, until $\hat{\boldsymbol{f}}_\kappa$'s stabilize:

$$For\ \mu = 1, ..., p:$$
$$For\ all\ \kappa \in \mathcal{R}(k^{(\mu)}):$$
$$\hat{\boldsymbol{f}}_\kappa \Leftarrow \boldsymbol{S}_\kappa (\boldsymbol{y}_\kappa - \boldsymbol{\alpha} - \sum_{\substack{m=1 \\ m \neq \mu}}^{p} \hat{\boldsymbol{f}}_\kappa^{(m)})$$

Fig. 2. Backfitting algorithm for a p-layer additive F_0 model.

3. MODEL TRAINING WITH SMOOTHING PARAMETER SELECTION

3.1. Experimental Settings

The training algorithm described in the last section was applied to the training of an three-layer English F_0 model with an optimal selection of smoothing parameters described in this section. Three layers consist of the phrase layer, the word layer, and the pitch accent layer with the syllable granularity. The model was trained and tested using the Boston University Radio News Corpus [9], speaker F2B. This part of the entire corpus consists of approximately 45 minutes of radio news read aloud by a female speaker of American English. ToBI labels are assigned by hand to the corpus. We further transcribed the corpus with syllable and word labels by performing a forced alignment with acoustic models adapted to this corpus. All 122 paragraphs that had full ToBI labels associated with them were divided into 110 paragraphs containing 12,704 syllables for training and the remaining 12 paragraphs with 1,863 syllables for testing. F_0 values were extracted from the corpus every 10ms using the Snack Sound Toolkit [10]. Possible octave errors were rectified by a modified de-step filter [11], in which F_0 instances detected as too low were doubled in frequency. The mean and the standard deviation of the F_0 were 170.5 Hz and 42.4 Hz in the training set, and were 170.5 Hz and 43.9 Hz in the test set. The original pitch samples were normalized to have the same number of data points per syllable interval by linearly stretching or shrinking each syllable, before the estimation. The number of data points per syllable was set to ten in the experiments described in this paper. There were 64 distinct intonational phrase types, 59 word-level component types, and 24 pitch accent types including contextual labels and <none> in the training set [7].

The smoothing parameters λ_m, $(m = 1, 2, 3)$ were chosen by 10-fold cross-validation (CV) [4] described below. About 15 iterations were enough for the convergence of backfitting iteration for the three-layer additive model presented in this paper. The elapsed time for the training with 15 iterations was approximately 200 seconds using one CPU on a 2.0GHz Intel Xeon machine running Linux. As an objective evaluation, we measured the accuracy of F_0 contour production in terms of root mean squared error (RMSE) and correlation coefficient (Corr) between model output and corpus F_0 in the voiced portions of the data, which are widely used to measure the goodness of F_0 models [12, 13, 14, 15]. Specifically, we calculated these values by aligning the model output with the corpus F_0 by linearly adjusting the length of the model output, then using the intervals where both model output and corpus F_0 are existent.

3.2. Smoothing Parameter Selection

In the loss functional for our three-layered F_0 model,

$$L(f) = \sum_{n=1}^{N} (y_n - f(k_n^{(1)}, x_n^{(1)}, k_n^{(2)}, x_n^{(2)}, k_n^{(3)}, x_n^{(3)}))^2 + \sum_{m=1}^{3} \lambda_m \sum_{\kappa \in \mathcal{R}(k^{(m)})} \int {f_\kappa}''(\xi^{(m)})^2 \mathrm{d}\xi^{(m)}, \tag{13}$$

there are three smoothing parameters, namely, λ_1, λ_2, and λ_3, which we need to optimize for the best expected predicting power of the model. We performed a 10-fold cross-validation [4] for this purpose. We had noticed from a few preliminary experiments that the additive F_0 model is quite insensitive to linear changes in these parameters. Therefore, we chose to explore discrete values that evenly change exponentially. We chose grid points

$$(10^{i/4}, 10^{j/4}, 10^{k/4}), \quad (i, j, k = \cdots, -2, -1, 0, 1, 2, \cdots)$$

in the three dimensional space spanned by λ_1, λ_2, and λ_3 as the search space for the best combination of these parameters. The optimization procedure was done in three stages where it first makes a rough global estimate and then seeks a local optimum:

1. First we tied three parameters together i.e. $\lambda_1 = \lambda_2 = \lambda_3$, and varied the value from 0.0001 to 10000 to find the value that yield the best values of root mean squared errors (RMSE) and the correlation coefficient (Corr), which turned out to be 1000. The best cross-validation (CV) estimates [4] of RMSE and Corr were 34.91 and 0.5738, respectively.
2. Each of λ_1, λ_2, λ_3 were varied with the other two parameters fixed to the best tied value (1000) to seek one that gives the best RMSE and Corr. The set of the best values obtained from this process was $(\lambda_1, \lambda_2, \lambda_3) = (10^4, 10^{10/4}, 10^{6/4}) = (10000, 316.2, 31.62)$, where the CV estimates of RMSE and Corr were 34.861 and 0.5749, respectively.
3. A gradient search was performed from this point to iteratively pick up the adjacent grid point with best average improvement of RMSE and Corr, up to the point from which no more improvement is obtained. As it turned out, the best combination from the procedure turned out to be $(\lambda_1, \lambda_2, \lambda_3) = (10^4, 10^{11/4}, 10^{6/4}) = (10000, 562.3, 31.62)$. However, the search for the best parameters were effectively convergent before this gradient search and the CV estimates of the RMSE and Corr made almost no change, which were 34.860 and 0.5749, respectively. (The RMSE and Corr values evaluated around the optimum point is shown in Fig. 3.)

When trained with the whole training data, the parameter set (λ_1, λ_2, λ_3) = (10000,562.3,31.62), yielded the RMSE and Corr scores, 33.85 and 0.638, respectively, for the test set.

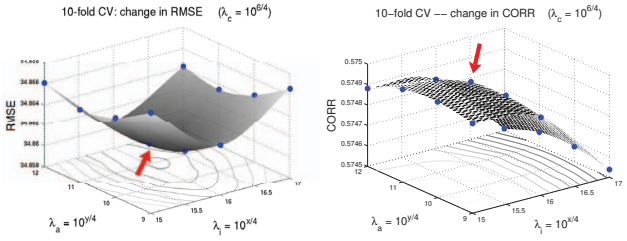

Fig. 3. RMSE and Corr plotted against the changes in λ_1 and λ_2 when $\lambda_3 = 10^{6/4}$. Red arrows point to the optimum combinations where $\lambda_1 = 10^4$ and $\lambda_2 = 10^{11/4}$.

4. DISCUSSION

From the experiment, we confirmed that the determination of the smoothing parameter values by 10-fold cross validation, in fact, improves the training results as shown in the Table 1. We, however, found out that the goodness-of-fit measures we have been using, i.e. RMSE and Corr, are not very sensitive to the changes in smoothing parameter values and that they were already close to the optimum with the baseline settings.

It is also interesting to see that the smoothing parameters after optimization show the tendency of having larger values to incur more smoothness for longer-span components (e.g. λ_1) and smaller values that compells less smoothness for shorter-span components (e.g. λ_3), which matches our intuition.

Table 1. RMSE and Corr results for the test set with baseline ($\lambda_1 = \lambda_2 = \lambda_3 = 1$) and 10-fold CV-trained models.

method	RMSE	Corr
baseline	33.98	0.6340
10-fold CV	33.85	0.6380

5. CONCLUSION

In this paper, we presented the derivation of the *backfitting* training algorithms for the general multi-layer additive F_0 models and presented an optimal method to set the values of smoothing parameters using cross-validation. We performed the training using an English speech corpus and confirmed the effectiveness of the proposed method.

6. REFERENCES

[1] A. Hunt and A. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. ICASSP '96*, 1996, pp. 373–376.

[2] E. Eide et al., "Recent improvements to the ibm trainable speech synthesis system," in *Proc. ICASSP 2003*, pp. I–708–I–711.

[3] M. Chu et al., "Microsoft Mulan – a bilingual TTS system," in *Proc. ICASSP 2003*, pp. I–264–I–267.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2001.

[5] S. Sakai and J. Glass, "Fundamental frequency modeling for corpus-based speech synthesis based on a statistical learning technique," in *Proc. ASRU 2003*, pp. 712–717.

[6] S. Sakai, "Fundamental frequency modeling for speech synthesis based on a statistical learning technique," *IEICE Transactions on Information and Systems*, vol. E88-D, no. 3, pp. 489–495, 2005.

[7] S. Sakai, "Additive modeling of english f0 contour for speech synthesis," in *Proc. ICASSP 2005*, 2005, pp. 277–280.

[8] G. Strang, *Introduction to Linear Algebra*, Wellesley Cambridge Press, 1998.

[9] M. Ostendorf et al., "The boston university radio news corpus," Tech. Rep. ECS-95-001, Boston University, Mar. 1995.

[10] `http://www.speech.kth.se/snack/`.

[11] P. Bagshaw, *Automatic Prosodic Analysis for Computer-Aided Pronunciation Teaching*, Ph.D. thesis, Univ. of Edinburgh, 1994.

[12] K. Ross and M. Ostendorf, "A dynamical system model for generating fundamental frequency for speech synthesis," *IEEE Trans. SAP*, vol. 7, pp. 295–309, 1999.

[13] Alan W. Black and Andrew J. Hunt, "Generating f0 contours from tobi labels using linear regression," in *Proc. ICSLP '96*, pp. 1385–1388.

[14] K. Dusterhoff, A. Black, and P. Taylor, "Using decision trees within the tilt intonation model to predict F0 contours," in *Proc. EUROSPEECH'99*, pp. 1627–1630.

[15] X. Sun, "F0 generation for speech synthesis using a multi-tier approach," in *Proc. ICSLP'02*, pp. 2077–2080.