Abstract

The ability to remotely monitor the estimated 103 bat species in French Guyana would lead to greatly improved estimates of bat numbers and distribution. Researchers in the South American department have spent several months in the forest recording the sound of various species in an effort to build a comprehensive database from which a sound map of the indiginous Chiroptera order can be built. We have been provided with a large number of these recordings, with which we have developed a statistical classifation algorithm which correctly identifies the species of a single bat from 16 possible species with an 82% success rate. The classification algorithm is a form of boosting with trees as weak classifiers, and the bat features used in learning relate to the shape of individual chirps in the spectrograms. The report also presents *deformable templates*, a potentially interesting tool for bat classification. While we have not had success classifying bats with this method, our implementation and analysis have shed light on how this computationally intensive method may be accelerated with the correct use of sparse matrices.

Résumé

La capacité de surveiller à distance les 103 espèces estimées de chauves-souris en Guyane française conduirait à des estimations considérablement amélioré de comptes de chauves-souris et de leur distribution. Chercheurs dans le département d'Amérique du Sud ont passé plusieurs mois dans la forêt enregistrant du son de différentes espèces dans un effort pour construire une base de données complète á partir de laquelle une carte son de l'ordre des chiroptères indiginous peut tre construit. Nous avons reçu un grand nombre de ces enregistrements, avec lequel nous avons développé un algorithme statistique qui identifie correctement les espèces d'une seule chauve-souris à partir de 16 espèces possibles, avec un taux de réussite de 82%. L'algorithme de classification est une forme de *boosting* avec des arbres pour classificateurs faibles. Le rapport prsente galement des modles dformables (*deformable templates*), un outil potentiellement intressant pour la classification des chauve-souris. Alors que nous n'avons pas eu le succs classer les chauves-souris avec cette mthode, notre mise en oeuvre et l'analyse ont mis en lumire la faon dont cette mthode de calcul intensif peut tre acclr avec l'utilisation correcte des *sparse matrices*.

Introduction and Summary

Researchers in French Guyana have provided us with 573 recordings of bat sound. Each recording is accompanied by information relating to the time and environment, and the number of bats and species present. Of these recordings, about 100 contain between two and six distinct bat species. These multi-species recordings will not be considered in this report. Of the remaining single species recordings, a further approximate 100 have not had definitive species identification, and so will be excluded during our learning process. Finally, all species which have less than 5 recordings will also be excluded from our analysis, leaving a total of 16 species and 343 recordings.

A spectrogram is taken of each recording, and sharpened using a technique called reallocation, described in Section 1. Bats leave signatures in spectrograms as temporal series of short, high energy chirps. These chirps are lines which vary in frequency over a limited time period, and they have variable shape. They can best be described by their frequency, duration, gradient and convexity. Three methods of locating chirps in spectrograms are discussed in Section 2 and implemented with varying success. Several filters are applied to found chirps to increase the purify of the final set, as discussed in Section 2.4. An average of 35 chirps of the 100 objects found per recording survive the filtering process, leaving a clean training set.

After the chirp training set has been established, two different approaches are considered in parallel. The first is a direct approach to the statistical classification of chirps (and hence recordings). The second, while also providing a route to the eventual classification of chirps, attempts to geometrically model species chirps using deformable templates. The first approach involves the fitting of weighted splines to chirps, from which parameters such as curvature and gradient are calculated and used for statistical learning, as described in Section 4. The learning algorithm chosen is a form of boosting with classification trees. Cross-validation shows that this approach results in 82% of the recordings being correctly classified.

Implementation of the deformable template approach did not reach the classification stage, and so cannot be compared to the splining method. Nonetheless in this report a detailed presentation of this method is given and an analysis of the algorithm is presented, illustrating how the correct treatment of sparse matrices can result in speed-ups proportional to image size (pixels per chirp window). This is described in Section 5 and presented in detail in Appendix F. The algorithm is applied successfully to a model example, and with mixed success to the chirps of Saccopteryx leptura.

1 Processing the signal

In this section, we will discuss the path from a sound recording to a reallocated spectrogram, providing motivation along the way. An introduction and definitions pertaining to the Short-Time Fourier Transform (STFT) and the Heisenberg uncertainty principle can be found in Appendix A. The window g used in the STFT eqrefstft plays an important role in determining the overall smoothness of a spectrogram. In a preliminary report from researchers in French Guyana [5], the authors use a Hanning[1] (Hann) window, defined as

$$w(n) = 0.5 - 0.5\cos\left(\frac{2\pi n}{M-1}\right) \qquad 0 \leq n \leq M-1.$$

In our report, both Hanning and Gaussian windows illustrated in Figure 1 are tested. As the Hanning window is what was apparently used in [5], we initially assumed that using any other window would prove fruitless. The use of the Gaussian window was implemented only because of its simplification of reallocation, as will be discussed. However reallocation aside, there was a great gain in using the Gaussian window, as illustrated in Figure 2.

We model bat sound to be of the form,

$$x(t) = \sum_{k=1}^{K} a_k(t)\cos\phi_k(t). \tag{1}$$

with frequency decomposition at time t thus being $\{\phi_k'(t)\}_{1\leq k\leq K}$. The pertinence of this model choice is discussed in Appendix B. We wish to estimate the amplitudes and frequencies in (1) from the sound file and will do so via the STFT. Reallocation, a method for making the energy distribution of time-frequency representations more concentrated, is a useful tool in this estimation. Reallocation works by moving the energy at points in the time-frequency distribution to more appropriate positions. As we assume that the underlying signal is of form (1), the appropriate target point for a given point in the time-frequency plane (t,ω) is the nearest point $(\tilde{t},\tilde{\omega})$ such that $\phi_k'(\tilde{t}) = \tilde{\omega}$ for some k. The spectrogram energy $S_x(t,\omega)$ is then moved to $(\tilde{t},\tilde{\omega})$, so that the reallocated spectrogram value at $(\tilde{t},\tilde{\omega})$ is

$$\tilde{P}_S(\tilde{t},\tilde{\omega}) = \sum_{t,\omega\to\tilde{t},\tilde{\omega}} P_S(t,\omega).$$

The method of locating the nearest ridge points, and the details of our implementation are discussed in Appendix C. Bat spectrogram reallocation is illustrated in Figure 3.

2 Chirp finding and filtering

We wish to locate individual bat chirps in a reallocated spectrogram. We have considered three ways of doing this, one of which proves to be noticeably more effective than the other two. The three methods will be referred to as: the *time-maximum method*, the *blocking method*, and the *filling method*. Both the blocking method and the filling method have the same basic structure, outlined in Algorithm 1.

[1] deduced from figure titles

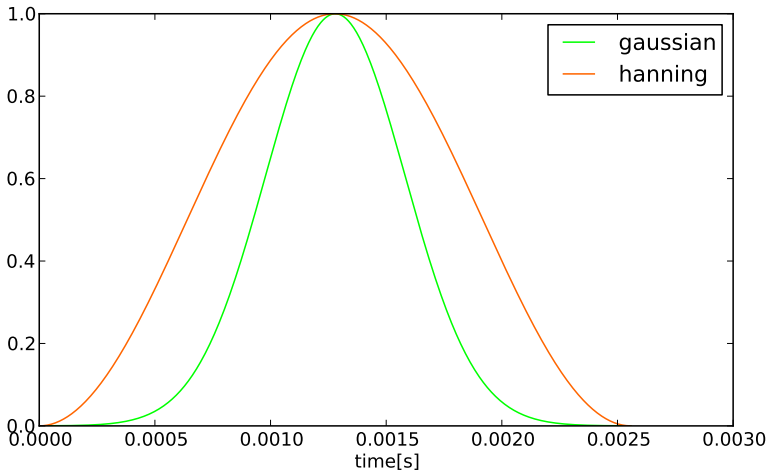


Figure 1: The two STFT windows implemented, the Gaussian window providing superior results

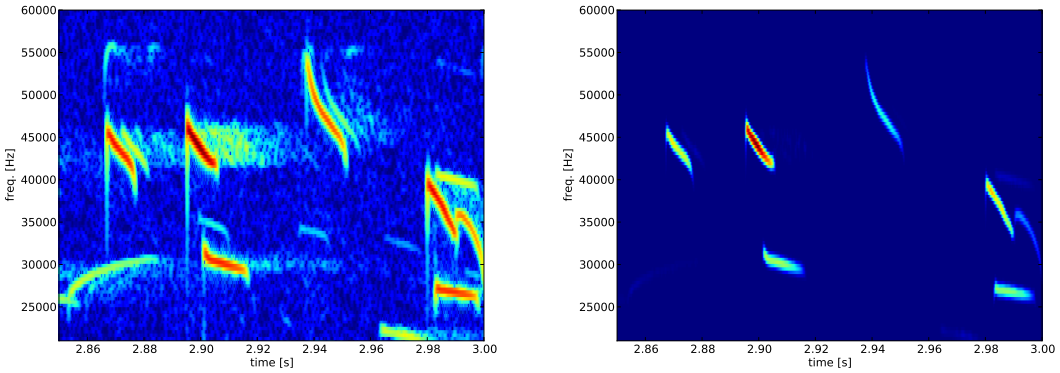


Figure 2: Spectrogram segments of a bat sound clip, using Hanning (left) and Gaussian (right) windows in the STFT

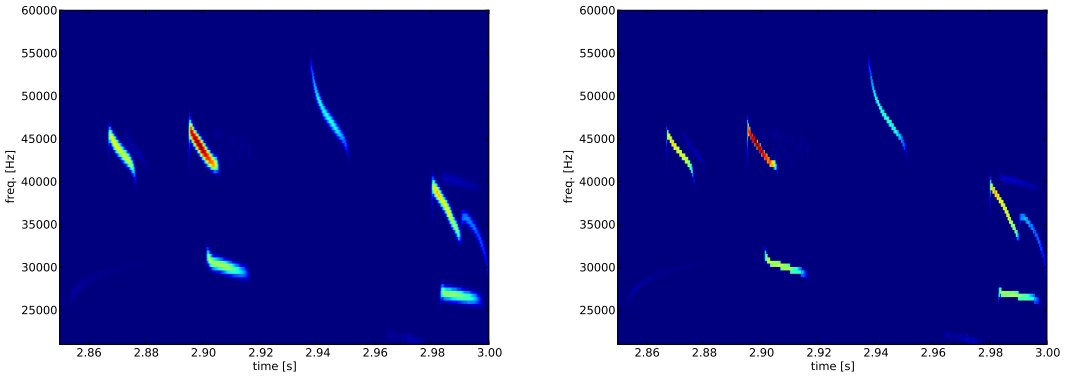


Figure 3: (Left) Spectrogram before reallocation and (Right) after reallocation

Algorithm 1 Skeleton of blocking and filling methods

1: **for** $i = 1$ to (number of chirps) **do**
2: Find spectrogram cell c_i containing the highest energy E_i
3: Locate shape S_i around c_i (using a blocking or filling step)
4: Ignore all cells in S_i in subsequent steps
5: **end for**

2.1 The time-maximum method

In this method, one finds the cell of highest energy at each spectrogram time slice, and locates contiguous segments where these cells are of equal or similar frequency. Various definitions of 'near/equal frequency' were tried, but none proved satisfactory. The main reasons for failure in comparison with the 2-dimesional methods are that chirps not overlapping in frequency can overlap in time, and a sharp sensitivity to noise.

2.2 The blocking method

The aim of the blocking step is to find the smallest block centred at c whose edge cells are all of energy less than fE, where E is the energy of cell c, and $f \in (0, 1)$. One can define 'smallest' block by edge length or area, they are here equivalent. Algorithm 2 presents the blocking step.

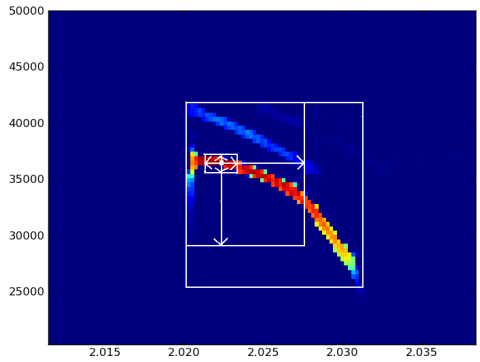
Algorithm 2 blocking step

1: grow $\leftarrow$ True
2: **while** grow = True **do**
3: grow $\leftarrow$ False
4: **for** `side` in [left, top, right, bottom] **do**
5: **if** $\max_{p \in \texttt{side}} E(p) > fE$ **then**
6: grow $\leftarrow$ True
7: move `side` one unit away from c.
8: **end if**
9: **end for**
10: **end while**

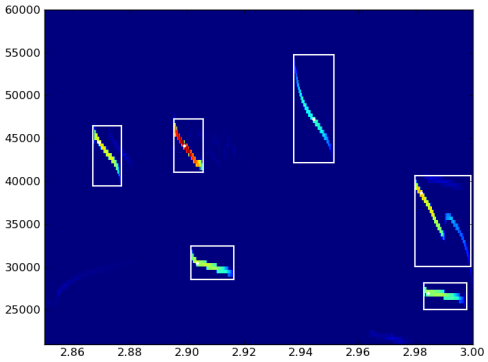
Different stages of the blocking step are illustrated in Figure 4a, and a case of it failing is shown in Figure 4b, where the blocking step has encompassed two chirps. One way to reduce the frequency of this common problem is to increase the factor f, but this results in an increased number of single chirps being dissected.

2.3 The filling method

In light of the blocking method's weakness, we introduce a method which follows chirp contours more closely than does the blocking method, reducing the occurence of multiple chirps being detected simultaneously. Again starting from point c of energy E, the final shape S consists of all cells which can be reached from c without at any stage crossing a cell of energy less than fE. Algorithm 3 presents the rough idea.



(a) The blocking method at three stages



(b) Illustration of final blocking results

Algorithm 3 the filling step

1: **while** shape S is still growing **do**
2: **for** cells on edge of shape S **do**
3: Add to shape S all direct neighbours of energy greater than fE
4: **end for**
5: **end while**

Not only is the filling method more effective at locating chirps, it also allows us to filter out surrounding noise: for the analysis of a single chirps, all cells not in shape S will be considered to have zero energy. In Figure 7 of Appendix D, the effect of performing filling on a region of spectrogram is illustrated.

2.4 Filters

While the filling method for finding chirps is better than the other methods considered, there are still many spurious objects detected. With the opinion that a small accurate set of chirps will prove most useful for our analysis, as prescribed in [5] where very few signals are used, we implement filters preventing spurious chirps from entering our training set. Initially each spectrogram is searched for the one hundred chirps of highest energy. Of these an average of 35 chirps survive the filtering process. The five filters implemented are presented in order of use in Appendix D, with the number of chirps filtered out by each also stated.

3 Chirp parameters

In this section we fit splines through chirp centres of mass. From the splines, species characterising gradient and curvature parameters are extracted. Several such parameters were considered based on their utility in classification, with the final set of parameters chosen based on the spline values at five equally spaced times along the chirp. Before curve fitting was tried, the naïve approach of using the raw center of mass values at the five points was considered. This approach performed almost as well as when splines were used to improve their location, suggesting that any additional spline modifications which could be considered beneficial, such as using the spline to calculate derivatives as opposed

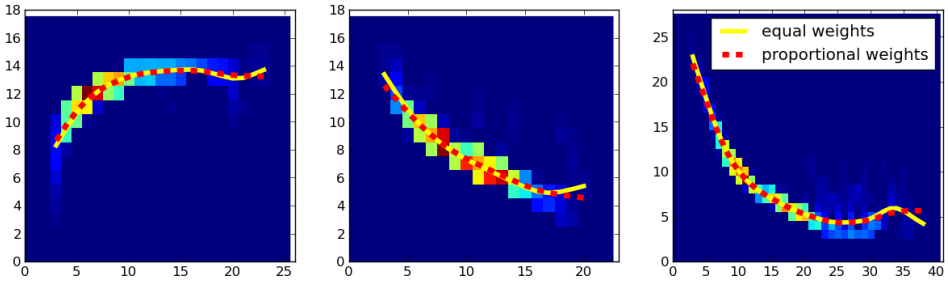


Figure 4: Spline of constant weights (yellow) and spline with weights proportional to time slice energies (red).

to the differencing approach we use, would be a waste of time. Below we describe the spline method used.

3.1 Weighted spline

High energy sections of observed chirp spectrograms are more reliable and less noisy than low energy sections, and it is in the lower energy tail that spurious energy cells commonly appear. The idea of our spline is therefore to put more weight on times of high energy. If we denote the center of mass of time slice i by y_i, the function we wish to minimize with respect to $\hat{y}$ is defined as

$$f_m(y, \hat{y}) = \sum_{i=0}^{N} w_i (y_i - \hat{y}_i)^2 + \lambda \sum_{i=1}^{N-1} (2\hat{y}_i - \hat{y}_{i+1} - \hat{y}_{i-1})^2. \tag{2}$$

Equation 2 has a unique minimum $\tilde{y}$ being the solution to the $N+1$ equations,

$$\begin{aligned}
(w_0 + \lambda)\tilde{y}_0 - 2\lambda\tilde{y}_1 + \lambda\tilde{y}_2 &= w_0\tilde{y}_0 \\
-2\lambda\tilde{y}_0 + (w_1 + 5\lambda)\tilde{y}_1 - 4\lambda\tilde{y}_2 + \lambda\tilde{y}_3 &= w_1\tilde{y}_1 \\
\lambda\tilde{y}_{i-2} - 4\lambda\tilde{y}_{i-1} + (w_i + 6\lambda)\tilde{y}_i - 4\lambda\tilde{y}_{i+1} + \lambda\tilde{y}_{i+2} &= w_i\tilde{y}_i \quad \text{for } 3 \le i \le N-2 \\
\lambda\tilde{y}_{N-3} - 4\lambda\tilde{y}_{N-2} + (w_{N-1} + 5\lambda)\tilde{y}_{N-1} - 2\lambda\tilde{y}_N &= w_1\tilde{y}_1 \\
\lambda\tilde{y}_{N-2} - 2\lambda\tilde{y}_{N-1} + (w_N + \lambda)\tilde{y}_N &= w_0\tilde{y}_0
\end{aligned}$$

The weights (w_i) are chosen to be higher for time slices of higher energy. We calculate an estimate of the spline value at time t where $t_i < t < t_{i+1}$ as a linear interpolation between $\tilde{y}_i$ and $\tilde{y}_{i+1}$. The results of using this spline with different weights are illustrated in Figure 4.

Our final choice of spline weights lies between those illustrated in Figure 4, being $w_i = 1 + E_i / \max_i(E_i)$, with $\lambda = 1$.

4 Chirp classification

Our data set consists of several hundred chirps belonging to 16 species, with the following number of chirps per species: 3401, 2023, 1282, 710, 565, 529, 492, 358, 352, 347, 320, 270, 205, 172, 164, 159. We have chosen to perform classification using a version of 'gradient boosting machine' with trees. The exact algorithm goes by the name SAMME, and is described in Appendix E.

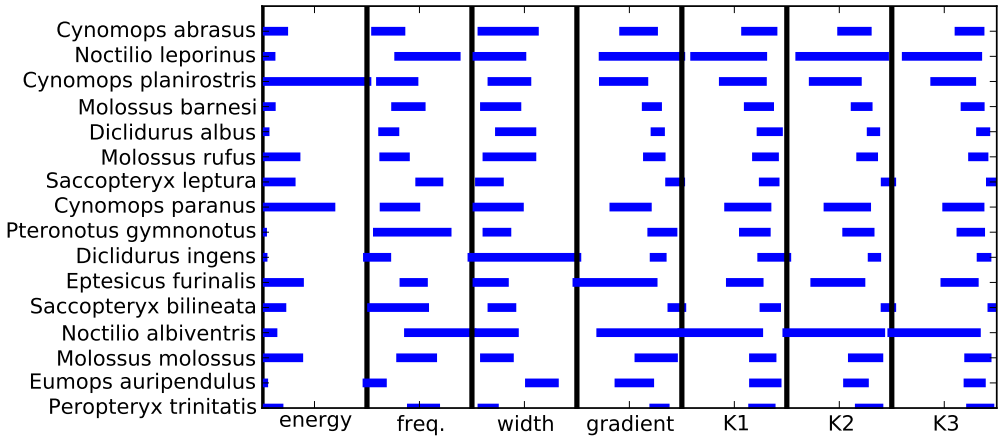


Figure 5: (20%, 80%) percentiles for normalised chirp parameters for the 16 species. Frequency proved to be the most useful single parameter, but using frequency alone only guaranteed the correct classification of 37 % of the files (125 of 343).

4.1 Classification Results

There are two sets of parameters which affect classification, those internal to SAMME and those coming from the chirp data. For the SAMME parameters, we use a learning rate of 0.1, trees of depth 3, and 100 steps of steepest descent. None of these parameters greatly affects classification accuracy in the vicinity of the chosen values. For the data parameters, the extraction of seven parameters from each chirp spline proved optimal. The first parameter is the peak energy of the chirp, the second is the mean frequency of the chirp, weighted by energy, and the third is chirp width (w). Parameters 4 to 7 are based on five equally spaced points from the start of the chirp to its end, $x_1 \cdots x_5$. They are the gradient $(x_5 - x_1)/w$ and the three curvatures, $4(2x_4 - x_5 - x_3)/w$, $4(2x_3 - x_4 - x_2)/w$ and $4(2x_2 - x_3 - x_1)/w$. In retrospect it proved unnecessary to perform this manipulation of the pure spline values, as the learning algorithm performed equally well when fed the pure spline values, however it eases the interpretation of parameters in Figure 5.

We used 4-fold validation to test performance. In cross-validation group formation, all chirps from a given recording fell into the same cross-validation group, preventing the potentially easy classification of a chirp by other chirps on the same recording. Thus grouped, single chirps from the 16 species are classified with 66 % accuracy (7504 of 11247). For file classification, the probabilities of all chirps on a file are combined to form an ensemble vote,

$$V(\text{file}) = \max_{s \in \text{species}} \sum_{c \in \text{chirps}} \max\left(\log \hat{P}(s|c), -5\right) \tag{3}$$

where $\hat{P}(s|c)$ comes from SAMME. Using this scheme, 82% of the recordings (281 of 343) were correctly classified, with species specific rates shown in Table 1 of Appendix E. The only moderate 16% increase from single chirp classification to recording classification suggests that misclassified chirps are not independently distributed. Figure 6 illustrates detected chirps on 4 recordings, with correctly classified chirps shown in red and misclassified chirps in yellow.

A bat call has 2 levels of information. At the first level are the individual chirps.

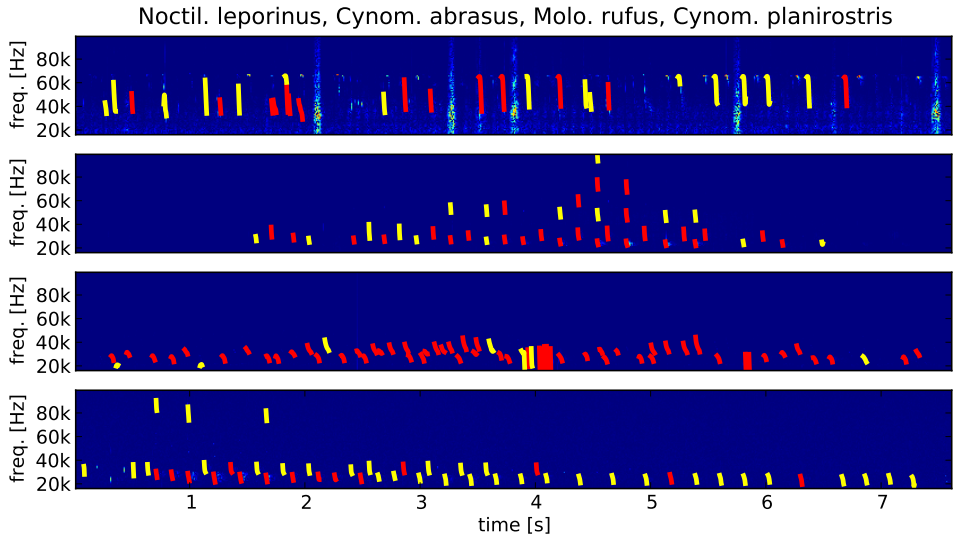


Figure 6: The spectrograms of 4 recordings, with the overlying spline fits of detected curves. Red signifies correctly classified chirps, yellow signifies incorrectly classified chirps. The four species are Noctilio Leporinus, Cynomops abrasus, Molossus rufus, Cynomops planirostris.

So far we have discussed how information at the first level can be used to classify a bat. The second level of information arises from the relationships between chirps on a recording. Chirps are not independently distributed and so combining probabilities via (3) is suboptimal. Level 2 information is defined as all dependencies between chirps which (3) fails to capture. One approach to level 2 modelling is the Hidden Markov Model, shown to be successful for bat classification in [13]. Therein bat chirps are modelled as belonging to one of several species specific subgroups, with transition probabilities between subgroups to be deduced. Our chirp data does not fall neatly into subgroups, but does show temporal patterning. As a naïve substitute to an HMM, we extended each 7 variable chirp vector to one of 9 variables, the two additional variables being the frequency of the previous chirp and the time since the previous chirp. The hope was to indirectly include whether the chirp was of high or low frequency relative to its neighboring chirps. There was no significant change in the classification rate at the chirp or the recording level, suggesting that HMMs would have little success with our chirps.

5 Deformable templates

In this section based on [1], bat chirps are modelled as deformations of species specific templates. The idea would then be to classify chirps according to how close they are (in a sense to be specified) to species templates. In the appendix F we present an implementation of the method, and provide an analysis of its performance, as well as describing an extended model which allows for several templates to exist for the same species.

Assume that the image (the window of spectrogram containing the chirp) is observed on a set of points $x_s, s \in \Lambda$, embedded in R^2. The template function is written as $I_\alpha : R^2 \to R$, where the subscript α represents the parameters defining the shape of the template, as will be discussed shortly. For each observation $y : y = \{x_s\}_{s\in\Lambda}$, one assumes

a deformation field $z_\beta : R^2 \to R$ such that

$$y(s) = I_\alpha\{x_s - z_\beta(x_s)\} + \sigma\epsilon(s), \tag{4}$$

where $\epsilon(s)$ are IID $\mathcal{N}(0,1)$. Equation 4 is written more compactly as $y = z_\beta I_\alpha + \sigma\epsilon$. The template I_α is defined as a linear combination of kernel functions, centred at a prescribed set of k_p landmark points $(x_{p,k})_{1\le k\le k_p}$. The subscript p here is used to denote a *photometric* quantity, as opposed to a *geometric* (deformation) quantity for which the subscript g will be used. The template image is represented as,

$$I_\alpha(x) = \sum_{k=1}^{k_p} \alpha(k) K_p(x, x_{p,k}).$$

In [1] the landmark points are chosen to extend over a grid of a larger size than the observation region. This is done as deformations require values which lie outside the observed domain. A discussion of this point, and our reason for not implementing it, can be found in Appendix F. We choose the photometric kernel to be the $C^{(1)}$ kernel,

$$K_p(x, x_{p,k}) = \cos^2\left(\frac{\pi|x - x_{p,k}|^2}{2\tilde{\sigma}_{p,k}^2}\right) \mathbf{1}_{[-1,1]}\left(\frac{|x - x_{p,k}|}{\tilde{\sigma}_{p,k}}\right). \tag{5}$$

Because of its compact support, kernel 5 will allow us to take computational shortcuts with sparse matrices. In addition to a photometric template described by parameters α, a deformation field is defined by parameters β. For the geometric template, the landmark points are denoted by $\{x_{g,k}\}_{1\le k\le k_g}$, and each $\beta(k)$ associated to landmark point k has an $x-$ and a $y-$ component $(\beta^1(k), \beta^2(k))$ so that

$$z_\beta(x) = \sum_{k=1}^{k_g} K_g(x, x_{g,k}) \left(\beta^{(1)}(k), \beta^{(2)}(k)\right).$$

For a given species α and σ in (4) are constant, while β is an observation dependant variable. The deformation parameter β is modelled as coming from a multivariate Gaussian of mean 0 and covariance matrix Γ. Thus the species specific parameters of interest are α, σ and Γ, which are denoted by $\theta = (\alpha, \sigma, \Gamma)$. The likelihood of the observed data has the form of an integral over the unobserved deformation parameters,

$$q(y|\theta) = \int q(y|\beta, \alpha, \sigma) q(\beta|\Gamma)\, d\beta \tag{6}$$

where $q(y|\beta, \alpha, \sigma)$ and $q(\beta|\Gamma)$ are multivariate normal distributions as previously described. In Appendix F we discuss the choice of a prior distribution over θ, and an implementation of the soft and hard EM algorithms for maximising the posterior probability of θ. We have implemented a simplified version of the two-step hard EM algorithm (a.k.a. fast approximation with modes). In [1] the β-update step in the fast approximation with modes is presented as

$$\beta^* = \arg\min_\beta \left(\frac{1}{2}\beta^T\Gamma^{-1}\beta + \frac{1}{2\sigma^2}|y - K_p^\beta\alpha|^2\right) \tag{7}$$

In our implementation we assume that the background noise is $\sigma = 0$, thus the β regularisation term in (7) falls aside. All importance is placed on matching the observation

to the deformation, irrespective of how unlikely the deformation is. In retrospect setting the background noise to be zero may not be a good idea, even if there really is no noise. In practice the noise term also takes account of the failure of the model to fit a noiseless observation. Nonetheless, we will proceed with $\sigma = 0$. The hard EM is presented in Algorithm 4,

Algorithm 4 Approximation with modes in noiseless model

1: **for** $j = 1$ to n-images **do**
2: $\quad \beta_{0j} \leftarrow 0$
3: **end for**
4: **for** $i = 0$ to n-steps **do**
5: $\quad$ Compute α_i which minimizes $\sum_{j=1}^{\text{n-images}} \|y_j - K_p^{\beta_{ij}} \alpha_i\|^2$ (α-update)
6: $\quad$ **for** $j = 1$ to n-images **do**
7: $\quad\quad$ Compute the $\beta_{(i+1)j}$ which minimises $\|y_j - K_p^{\beta_{(i+1)j}} \alpha_i\|^2$ (β-update)
8: $\quad$ **end for**
9: **end for**

In the Algorithm 4, $K_p^\beta \alpha$ are the values of the image $z_\beta I_\alpha$ at $\{x_s\}_{s\in\Lambda}$, with K_p^β being the matrix of size $|\Lambda| \times k_p$ given by

$$K_p^\beta(s, j) = K_p(x_s - z_\beta(x_s), x_{p,j}), \quad s \in \Lambda,\ 1 \le j \le k_p.$$

A detailed description of the nonlinear β-update step (line 7 of Algorithm 4) wth a description of how it can be solved using steepest descent is given in Appendix F. In general the calculation of the direction of steepest descent requires $O(|\Lambda|^3)$ operations, but with the correct use of sparse matrices this can be reduced to $O(|\Lambda|^1)$. Certain $O(|\Lambda^2|)$ calculations in the β-update step cannot be reduced to $O(|\Lambda^1|)$, and so the total gain of correctly using sparse matrices is $O(|\Lambda|^1)$. Details are provided in the Appendix. Also to be found in the Appendix F are two examples of our implementation, the first with simulated images where the true template is well reconstructed in 9 iterations. The second is with real bat chirps, with no convergence to anything particularly interesting.

Conclusion

In our report, we present the steps taken in building our classifier. These include STFT window choice, reallocation, chirp finding, chirp filtering, spline fitting and finally boosting with classification trees. The final classifier has an 82% classification rate when presented with 16 species. This should be compared to the best 'blind' classifier, which classifies all recordings as the most commonly occuring species, Molossus rufus. Such a classifier would have a 22% classification rate.

We present the method of deformable templates, and apply it succesfully to simulated data. With real bat chirps, the method has not provided satisfactory results. We address computational questions concerning the order of operations required to perform certain steps, and show how with the correct use of sparse matrices the number of steps required can be reduced by a factor proportional to the size of the image.

References

[1] S. Allassonnière, Y. Amit, and A. Trouvé. Towards a coherent statistical framework for dense deformable template estimation. J.R. Statist. Soc.B, 69:2007, 2006.

[2] D. W. Armitage and H. K. Ober. A comparison of supervised learning techniques in the classification of bat echolocation calls. Ecological Informatics, 5:465–473, 2010.

[3] F. Auger, E. Chassande-Mottin, and P. Flandrin. Réallocation de Levenberg-Marquardt. Unpublished, 1:1–4, 2011.

[4] F. Auger and P. Flandrin. Improving the readability of time-frequency and time-scale representations by the reassignment method. IEEE Trans. Signal Processing, 43:1068 – 1089, May 1995.

[5] M. Barataud, M. Delaval, T. Disca, S. Giosa, and V. Rufray. Rappor: Identification et écologie acoustique des chirpotères Guyane Française. Internal report, November 2011.

[6] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In European Conference on Computational Learning Theory, pages 23–37, 1995.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Special Invited Paper. Additive Logistic Regression: A Statistical View of Boosting. The Annals of Statistics, 28(2):pp. 337–374, 2000.

[8] J. H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. The Annals of Statistics, 29(5):pp. 1189–1232, 2001.

[9] J. H. Friedman. Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4):367 – 378, 2002.

[10] K. Kodera, R. Gendrin, and C. de Villedary. A new method for the numerical analysis of nonstationary signals. Phys. Earth and Plan. Int, 12:142–150, February 1976.

[11] J. Liu. Monte Carlo Strategies in Scientific Computing. Springer Verlag, New York, 2001. 12.

[12] S. Mallat. Traitement du signal. Département de Mathématiques Appliquées, Ecole Polytechnique, 2012.

[13] Mark D. Skowronski and John G. Harris. Acoustic detection and classification of microchiroptera using machine learning: Lessons learned from automatic speech recognition. Acoustical Society of America, 119(3):1817–1833, 2006.

[14] Ji Zhu, Saharon Rosset, Hui Zou, and Trevor Hastie. Multi-class adaboost. Technical report, 2005.

A. STFT and the Uncertainty Principle

Following the notation introduced in [12], the short-time Fourier transform of a signal $x(t)$ at time u and frequency ξ is defined as

$$S_x(u,\xi) = \int_{-\infty}^{\infty} x(t) g_{u,\xi}^* \, dt. \tag{8}$$

where $g_{u,\xi}$ consists of an even window function g, multiplied by an exponential:

$$g_{u,\xi} = g(t-u) \exp(i\xi t).$$

The window g guarantees that $S_x(u,\xi)$ depends only on times in the proximity of time u. To obtain a similar constraint on frequency, one first notes that using Parseval's formula on (8) provides

$$S_x(u,\xi) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{x}(\omega) \hat{g}_{u,\xi}^* \, d\omega,$$

and that using basic Fourier identities one can show that,

$$\hat{g}_{u,\xi}(\omega) = \exp(-iu(\omega-\xi)) \hat{g}(\omega-\xi).$$

Combining these leads to the conclusion that only frequencies ω for which $\hat{g}(\omega-\xi)$ is non-negligible will contribute to $Sf(u,\xi)$. We quantify the localisation of time and frequency dependence on $g_{u,\xi}$ and $\hat{g}_{u,\xi}$ respectively via the quantities σ_t^2 and σ_ω^2,

$$\sigma_t^2 = \int_{-\infty}^{+\infty} (t-u)^2 |g_{u,\xi}(t)|^2 \, dt = \int_{-\infty}^{+\infty} t^2 |g(t)|^2 \, dt \tag{9}$$

$$\sigma_\omega^2 = \frac{1}{2\pi} \int_{-\infty}^{+\infty} (\omega-\xi)^2 |\hat{g}_{u,\xi}(t)|^2 \, dt = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \omega^2 |\hat{g}(t)|^2 \, dt \tag{10}$$

In the time-frequency plane, a *Heisenberg box* of size $\sigma_t \times \sigma_\omega$ centred at (u,ξ) denotes the region of contribution to $g_{u,\xi}$. Ideally, to sharply localise the energy in the time-frequency plane this box should be as small as possible, rendering contributions of nearby time-frequency points negligible. But there is a theoretical minimal area of the Heisenberg box, and a trade-off thus needs to be reached between the sharpening of resolution in time (decreasing σ_t) and in frequency (decreasing σ_ω). This trade-off is summarised by the Heisenberg uncertainty principle,

$$\sigma_t^2 \sigma_\omega^2 \geq \frac{1}{4}. \tag{11}$$

For most applications only the magnitude of the STFT is of interest, and the phase is discarded. The square of the magnitude of the STFT is commonly referred to as the spectrogram, denoted henceforth by $P_x(u,\xi)$. The decomposition of the STFT into spectrogram and phase can be presented as follows,

$$S_x(u,\xi) = P_S f(u,\xi)^{\frac{1}{2}} \Phi(u,\xi).$$

B. Chirp modelling

We have introduced the STFT as a means of decomposing a signal into time and frequency components, but we have not addressed the important question as to how best an instantaneous frequency should be defined, should a congruent definition even exists. To this end, we consider a case where a natural definition of instantaneous frequency exists, that is

$$x(t) = a\cos(\phi(t)) = a\cos(\omega_0 t + \phi_0).$$

The frequency of this signal is ω_0, independent of time. We can extend this to the generalised case where the signal is

$$x(t) = a(t)\cos(\phi(t)), \tag{12}$$

for which we define the instantaneous frequency as

$$\omega(t) = \phi'(t).$$

It is clear that there is not a unique decomposition of the form (12) for a given signal $x(t)$. This problem can be solved by considering the analytical part of $x(t)$, that is $x_f(t)$, where then $x(t) = \mathrm{Re}(x_f(t))$. However there are cases where decomposing a signal using one amplitude varying cosine function is not ideal, as illustrated by the example

$$x(t) = a\cos(\omega_1 t) + a\cos(\omega_2 t),$$

whose analytic decomposition results in the representation of x as an amplitude varying cosine function of frequency $(\omega_1+\omega_2)/2$, with amplitude $a(t) = a|\cos(\frac{\omega_1-\omega_2}{2}t)|$. To avoid this unnatural representation, it is necessary to replace the idea of a single instantaneous frequency by that of a discrete set of frequencies, such that our signal becomes a sum of cosines,

$$x(t) = \sum_{k=1}^{K} a_k(t)\cos\phi_k(t). \tag{13}$$

with frequency decomposition at time t being $\{\phi_k(t)\}_{1\le k\le K}$. It is common to assume that a signal is of the form (13), and attempt to reconstruct the amplitudes and frequencies from the signal via the STFT. Reallocation is a useful tool in this process, and we will describe it in the following subsection.

C. Reallocation and Ridge detection

To locate the nearest *ridge* point $(\tilde{t}, \tilde{\omega})$, we make use of the following Theorem from [12],

Theorem 1. *Let $x(t) = a(t)\cos\phi(t)$. If the variation of $a(t)$ and $\phi'(t)$ are negligible in $[u - s/2, u + s/2]$, the support of $g(t-u)$, and $1/\phi'(t) \le s$, then for all $\xi \ge 0$ we have*

$$S_x(u,\xi) \approx \frac{1}{2}a(u)\exp(i(\phi(u) - \xi u))\,\hat{g}\left(\xi - \phi'(u)\right). \tag{14}$$

For a given time u, the approximation (14) obtains its maximum at $\xi = \phi'(u)$, which is as we would expect as this is precisely the frequency of the single cosine signal $x(t)$. We consider now the phase of the approximation (14),

$$\Phi_S(u,\xi) = \phi(u) - \xi u.$$

Consider the partial derivatives of the phase with respect to time and frequency on the ridge:

$$\frac{\partial \Phi_S(u,\xi)}{\partial u} = \phi'(u) - \xi \ \ (= 0 \text{ on ridge}) \tag{15}$$

$$\frac{\partial \Phi_S(u,\xi)}{\partial \xi} = -u. \tag{16}$$

Thus in the case of a single slowly varying cosine satisfying Theorem 1, to find the ridges it suffices to locate the points satisfying (15) and (16). In the case of several chirps, where $x(t)$ is expressed as a sum of cosines as in (13), the linearity of the STFT guarantees the approximation,

$$x(t) \approx \frac{1}{2}\sum_{k=1}^{K} a_k(u)\hat{g}\left(\xi - \phi_k'(u)\right)\exp(i(\phi_k(u) - \xi u)).$$

For which ridges corresponding to the kth cosine will satisfy equations 15 and 16 at time u provided that for all $j \neq k$

$$|\hat{g}\left(\phi_k'(u) - \phi_j'(u)\right)| << |\hat{g}(0)|.$$

We will now introduce the notation used in [3],

$$R_x^g(u,\xi) = \begin{pmatrix} u + \dfrac{\partial \Phi_S(u,\xi)}{\partial \xi}(u,\xi) \\ -\dfrac{\partial \Phi_S(u,\xi)}{\partial u}(u,\xi) \end{pmatrix} \tag{17}$$

so that $R_x^g(u,\xi) = 0$ for points on the ridges of the signal. There is no approach for numerically finding points satisfying $R_x^g(u,\xi) = 0$ which is immediately evident. Working directly with the phase of the spectrogram is seldom a good approach, as algorithms for calculating phases are delicate and generally the calculated phase is modulus 2π, which is useless in any calculation of phase derivative. A reformulation of reallocation as presented in [4] does not require the direct use of the phase. Therein it is shown that $R_x^h(u,\xi)$ can be expressed as,

$$R_x^h(u,\xi) = \begin{pmatrix} \text{Re}\left(\dfrac{S_x^{tg}(u,\xi)}{S_x^g(u,\xi)}\right) \\ -\text{Im}\left(\dfrac{S_x^{dg/dt}(u,\xi)}{S_x^g(u,\xi)}\right) \end{pmatrix}, \tag{18}$$

where the superscript of the STFT indicates the window used, S_x^{tg} and $S_x^{dg/dt}$ thus indicating the use of windows $tg(t)$ and $\frac{dg}{dt}(t)$ respectively. With the use of a Gaussian window

$$g(t) = \frac{1}{\pi^{\frac{1}{4}}\sqrt{\lambda}}\exp\left(-t^2/(2\lambda^2)\right),$$

for which $tg(t) = -\lambda^2\frac{dg}{dt}(t)$, to calculate $R_x^g(u,\xi) = 0$ it suffices to calculate the quotient S_x^{tg}/S_x^g from whose real and imaginary parts $R_x^g(u,\xi) = 0$ are found.

As described in [3], there are three main classes of methods which one can now use to solve numerically $R_x^g(u,\xi) = 0$:

1. *Fixed point methods*, where one expects the convergence of a sequence $(u,\xi)_{n+1} = (u,\xi)_n - \lambda R_x^g(u_n,\xi_n)$.

2. *Differential methods*, where the differential equation $\frac{d(u,\xi)}{dt} = \lambda R_x^g(u_n,\xi_n)$ is simulated until a steady state is reached.

3. *Newtonian methods*, in which one wishes to observe the convergence of the sequence $(u,\xi)_{n+1} = (u,\xi)_n - \left[R_x'(u_n,\xi_n)\right]^{-1} R_x^g(u_n,\xi_n)$

For the reallocation of bat sound spectrograms we follow the approach of [10], that is take a single iteration of the fixed point method with $\lambda = 1$. Thus for each discrete point (u,ξ) in the time-frequency plane, we move the spectrogram energy $P_x(u,\xi)$ to the point $(\tilde{u},\tilde{\xi})$ where

$$(\tilde{u},\tilde{\xi}) = (u,\xi) - R_x^g(u,\xi)$$

The effect of this single step can be observed in Figure 4b

D. Figure and Filters

Filter 1: Touching neighbour filter

The edge of a loud chirp, that is the set of points around the chirp which are at energy fE, may still be at a high enough energy such that at a subsequent step in the chirp search, they have the highest energy. The resulting shape will wrap around the previously detected chirp, and is not relevant to us. Occasionally good chirps will be trapped by this filter, such as the 'yellow' chirp in Figure 7, but the overall purity of the chirp set increases sufficiently using this filter to validate its use. Of the 34500 chirps in the starting set, 15416 are removed by this filter, leaving 19084 chirps.

Filter 2: overall energy distribution

Clean chirps have energy localised in a narrow frequency band. Our second filter discards detections which have their energy dispersed too widely over the surrounding rectangle. The surrounding rectangle in this case is taken to be the smallest rectangle containing the chirp, which as an aside does not in general correspond to the rectangle found using the blocking method, described in Section 2.2. We find that discarding all chirps which contain less than 90% of their energy in the 13% of pixels of highest energy is optimal. This filter removes 4732 chirps, leaving 14352.

Filter 3: time-sliced energy distribution

The third filter is a localised version of filter 2. If, for any time slice of the chirp's spectrogram window, less than 95% of the chirps lie in the 20% of pixels of highest energy, the chirp is removed. This filter removes a further 2700 chirps, leaving 11652 chirps.

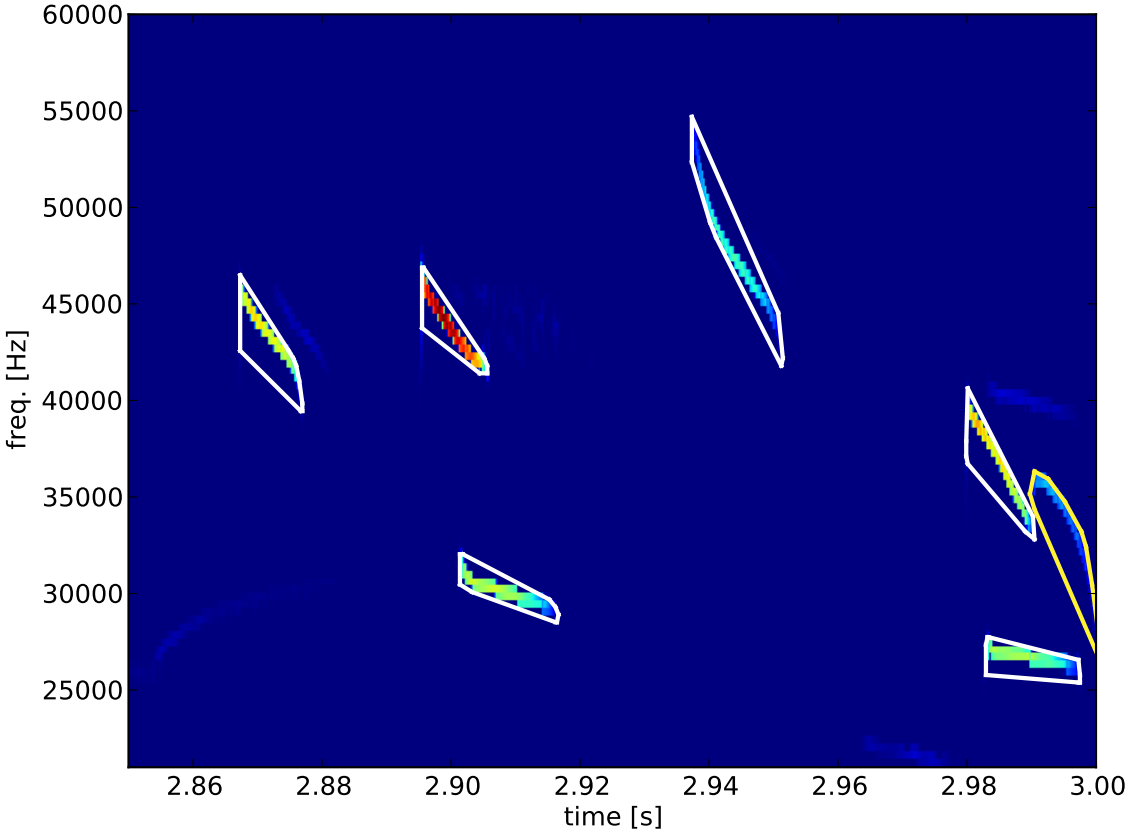


Figure 7: Result of *filling* on the reallocated spectrogram. The convex hulls surrounding each chirp are shown only for illustration, and do not feature in the analysis. The reason that the final chirp found is yellow will be described in the section on filtering, in particular the section 'Touching neighbours'.

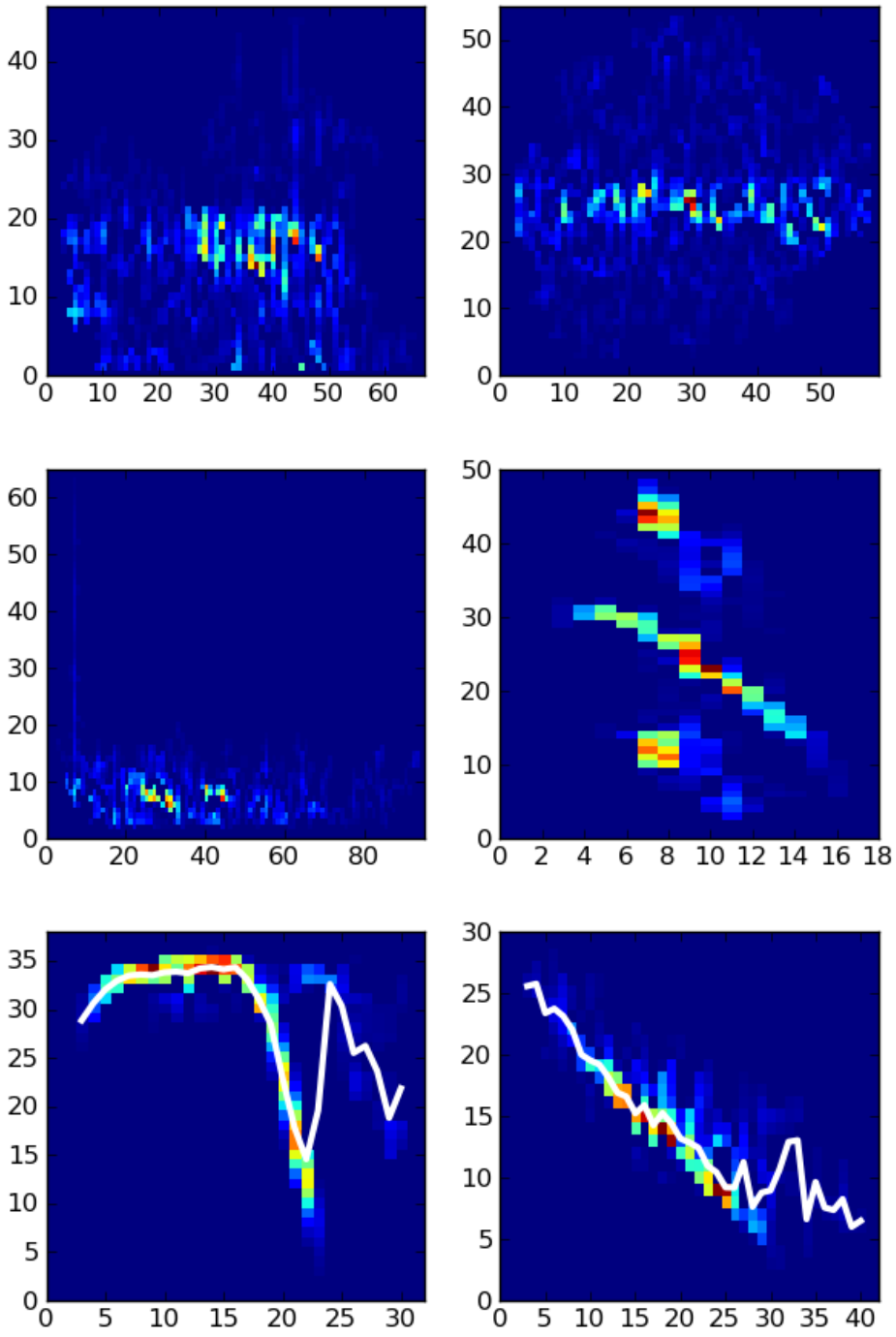


Figure 8: Examples of spurious objects removed using (Above) filter 2 and (Middle) filter 3 and (Below) filter 5.

Filter 4: Shape filter

Detections of duration less than 5 ms are likely to be artefacts. The second filter removes all detected objects of width less than 10 spectrogram frames (5 ms). 102 detections are removede leaving 11550 chirps.

Filter 5: overall movement filter

A final filter for removing chirps with high noise content discards objects whose mean frequency varies too greatly from time slice to time slice. In particular, if the inequality:

$$\sum_{1 \leq i \leq W} |\mathrm{m}(t_i) - \mathrm{m}(t_{i-1})| > fH$$

holds, where $m(t)$ is the mean frequency of the spectrogram window at time t, W and H are respectively the pixel width and height of the chirp spectrogram and f is some factor (which we take as 1.8), then the chirp is discarded. This filter removes a further 201 detections, leaving a final chirp set of size 11247.

The two chirps illustrated in Figure 8 removed by this filter do carry information, but would need to be cleaned if they were to contribute to the final classification algorithm. They are of poor quality in comparison with the chirps that made it through this final filter.

E. Boosting trees for classification

Boosting

Boosting [6] is a meta-technique used in statistical learning. It works by combining weak-classifiers into a committee, whose combined decision is potentially better than that of individual weak-classifiers. Boosting produces the committee by sequentially adding together weak-classifiers calculated by steepest descent [7]. The original algorithm has seen improvements with the inclusion of slow learning [8] and bagging [9], culminating in what is today the most commonly used version, the Gradient Boosting Machine (GBM) algorithm. The algorithm has recently been implemented in the python scikits.learn package[2], with a version for multiclass classification which is what use. A brief discussion of trees and loss functions is presented, and then the SAMME algorithm, which is the multiclass version of GBM, is explained.

Tree functions

The most commonly used weak-classifiers (a.k.a. basis functions) in boosting are trees. Trees are discontinuous functions which take discrete values in different regions of a domain. That is to say, a tree T has the form:

$$T(\vec{x}) = \begin{cases} z_1 & \text{if } \vec{x} \in R_1 \\ \vdots & \\ z_K & \text{if } \vec{x} \in R_K \end{cases}$$

where the K distinct regions $R_1 \cdots R_K$ together partition $\vec{x}$-space. The region boundaries can be described through the branchings of a tree, as illustrated in Figure 9. For

[2] http://scikit-learn.org/stable/

boosting, it is common to only use trees of a very simple form, that is only trees with branchings of the form $x^{(i)} < v$, where $x^{(i)}$ is one of the dimensions of $\vec{x}$-space and v is a real number. In the case of our bat data, $\vec{x}$ are the parameters fitted to the chirps in Section 3.

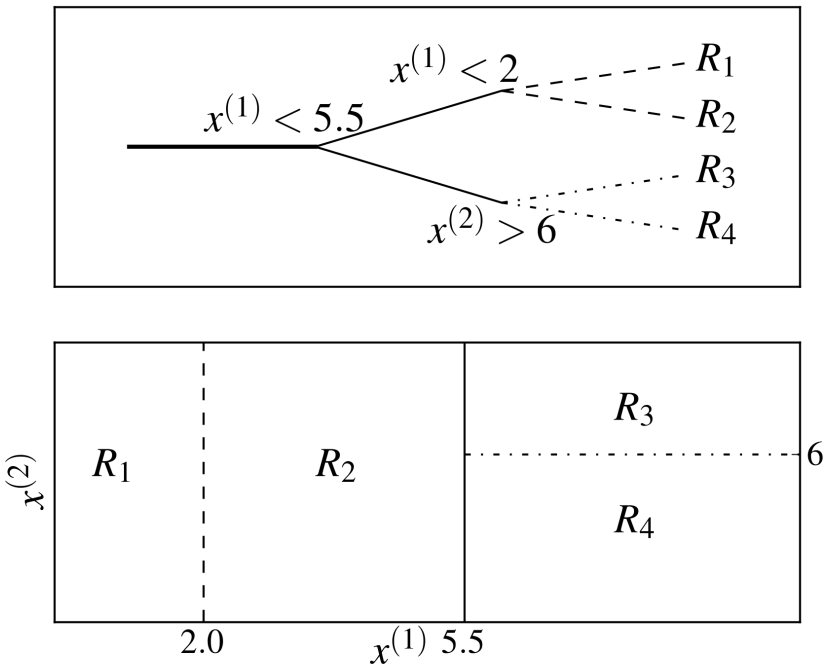


Figure 9: (*Above*) A tree of depth 2 for classifying an object into one of 2^2 regions. (*Below*) The tree domain containing 2^2 distinct regions as defined by the tree.

The multinomial loss function

Suppose we have observed n training points, each consisting of data and type: $(\vec{X}_i, \tau_i)$, where the data $\vec{X}_i$ is a d-dimensional vector, and τ_i is the label for one of K classes. For multiclass classification, it is useful to replace the label τ_i by a vector y_i, whose kth component is given by

$$y^{(k)} = \begin{cases} 1 & k = \tau \\ -\frac{1}{K-1}, & k \neq \tau. \end{cases}$$

Suppose we wish to find the function $F : R^d \to R^K$ which minimises the *multi-class exponential loss function*:

$$L(F) = \sum_{i=1}^{n} \exp\left(-\frac{1}{K} y_i^T F(X_i)\right), \tag{19}$$

where F has the constraint that it sums to zero at all $x \in R^d$. The properties of such a minimizer can be inferred on considering the minimizer of the expectation of a single observation,

$$\arg\min_{F(x)} \; E_{Y|x} \exp\left(-\frac{1}{K}(y^T F)\right) \tag{20}$$

subject to F summing to 0. The Lagrange of this constrained optimisation problem can be written as,

$$\exp\left(-\frac{F_1(x)}{K-1}\right)\mathrm{P}(c=1|x) + \cdots + \exp\left(-\frac{F_K(x)}{K-1}\right)\mathrm{P}(c=K|x) - \lambda\left(F_1(x) + \cdots + F_K(x)\right)$$

which has a global minimum at

$$F_k^*(x) = (K-1)\left(\log \mathrm{P}(c=k|x) - \frac{1}{K}\sum_{k'=1}^{K} \log \mathrm{P}(c=k'|x)\right) \;\; k = 1 \cdots K. \tag{21}$$

thus

$$\arg\max_k F_k^*(x) = \arg\max_k P(c=k|x).$$

This is exactly the Bayes optimal classifier rule for minimizing classification error. This validates our choice of loss function, as an F which minimises the loss function (19) and chosen from a sensible dictionary should provide a good approximation to the underlying probability distribution.

SAMME

Originally proposed for two class classification, the Gradient Boosting Machine [8] (GBM) with trees works by sequentially adding new trees to a function F, each addition reducing $L(F)$ and so hopefully improving the approximation of F to (21). It has been extended to the multiclass setting in the form of Stagewise Additive Modeling using a Multiclass Exponential loss function (SAMME) in [14]. The trees, which have depth D, are appended to F at each of the M iterations of the SAMME algorithm. Choosing larger M and D values results in a final lower $L(F)$. However, our end objective is not to reach the global minimum of (19) but to construct a good approximation to (21), and trees of lower depth are generally better suited to this end, being less prone to fitting noise. Algorithm 5 outlines the basics of an implementation of the SAMME algorithm.

Algorithm 5 SAMME

1: Initialize the observation weights $w_i = 1/n$, $1 \le i \le n$.
2: **for** $1 \le m \le M$ **do**
3: Fit a classifier $T^{(m)}(x)$ to the training data using weights w_i
4: Compute $err^{(m)} = \sum_{i=1}^{n} w_i \mathbf{1}\left(c_i \neq T^{(m)}(x_i)\right) / \sum_{i=1}^{n} w_i$
5: Compute $\alpha^{(m)} = \log \frac{1-err^{(m)}}{err^{(m)}} + \log(K-1)$
6: Set $w_i \leftarrow w_i \cdot \exp\left(\alpha^{(m)} \mathbf{1}(c_i \neq T^{(m)}(x_i))\right)$
7: Renormalise w_i
8: **end for**
9: Output is $C(x) = \arg\max_k \sum_{m=1}^{M} \alpha^{(m)} \cdot \mathbf{1}(T^{(m)}(x) = k)$.

	Cynomops abrasus	Cynomops paranus	Cynomops planirostris	Didelphurus albus	Didelphurus ingens	Eptesicus furinalis	Eumops auripendulus	Molossus barnesi	Molossus molossus	Molossus rufus	Noctilio albiventris	Noctilio leporinus	Peropteryx trinitatis	Pteronotus gymnonotus	Saccopteryx bilineata	Saccopteryx leptura
C.a.	15	0	0	0	0	0	1	0	0	5	0	0	0	0	0	0
C.pa.	3	0	1	0	0	0	0	0	0	8	0	1	0	0	0	0
C.pl.	0	0	16	0	0	1	0	0	0	1	1	1	0	0	0	0
D.a.	0	0	0	5	0	0	0	0	1	2	0	0	0	0	0	0
D.i.	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0
E.f.	0	0	0	0	0	12	0	0	0	0	0	0	0	0	1	0
E.a.	0	0	1	0	2	0	9	1	0	0	0	0	0	1	0	0
M.b.	0	0	0	0	0	0	0	36	5	4	0	0	0	0	0	0
M.m.	0	0	0	0	0	0	0	0	44	9	0	0	0	0	1	0
M.r.	0	0	1	0	0	0	0	2	1	75	1	0	0	0	1	0
N.a.	0	0	0	0	0	0	0	0	1	3	11	2	0	0	0	0
N.l.	0	0	1	0	0	0	0	0	0	1	0	18	0	0	0	0
P.t.	0	0	0	0	0	1	0	0	1	0	1	0	3	0	1	0
P.g.	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0
S.b.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0
S.l.	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	3

Table 1: Summary of recording classifications, with true species (row) vs classification (column). Cynomops paranus is the clear 'problem' species, with 0 of 13 recordings being correctly classified. We put this down to an inferior prior probability and a lack of distinct features. Figure 5 indicates a strong similarity between Cynomops paranus and Cynomops abrasus, and the sheer number of Molossus rufus makes gives any classification a prior 'bias' towards this species.

In passing, we mention that there are several methods for fitting the classifying tree $T^{(m)}(x)$ at line 3 of Algorithm 5, which will not be discussed here. The leaves of the trees are K-dimensional vectors of the form in eq. 5. The algorithm above can be easily adapted to include a learning rate of less than 1 (partial steepest descent) by manipulating the α-update step, and to include random sampling in training, commonly referred to in the literature as 'bagging'.

F. Deformable templates

Extending the domain

The factor by which the photometric landmark points extend beyond the observation region depends on the expected magnitude of deformation: the template necessarily goes to zero beyond the region of photometric landmark points, and so unless this decrease is an accurate model of the image, it is necessary that deformations do not reach the photometric boundary. Therefore larger deformations require a larger extension of

the photometric landmark region. If however the photometric landmark region extends unnecessarily beyond any expected deformation, parameter update calculations are unnecessarily enlarged. There is an undesirable effect caused by this extension. In the process of updating estimates of α, only the discrepancy between the image and model at deformation locations $\{x_s - z_\beta(x_s)\}_{s\in\Lambda}$ is considered. This leads to the template exploding in regions beyond the deformed template region. This is not a problem unless an observation has an unexpectedly large deformation. Constraining α to be positive does not prevent the boundary effect. In our bat implementation no extension is performed as chirp spectrograms are zero on and beyond the image boundary.

Prior distributions and Parameter Estimation with the EM algorithm

It is desirable to present the model in a Bayesian framework, and hence a prior distribution is introduced on the model parameters Γ, α and σ. Specifically $\Gamma \sim \nu_g$ and $\alpha, \sigma^2 \sim \nu_p$. In [1], ν_g is an inverse Wishart with covariance Σ_g, and ν_p consists of $\alpha \sim N(\mu_p, \Sigma_p)$, and on σ^2 an inverse Wishart. With these priors, the model for generating N observations from a new species is given by Algorithm 6,

Algorithm 6 Model for generating a species template and observations

1: Draw $\alpha, \sigma \sim \nu_p$ and $\Gamma \sim \nu_g$
2: **for** $i = 1$ to N **do**
3: Draw β_i from $\mathcal{N}(0, \Gamma)$
4: Construct observations y_i as the sum of $z_{\beta_i} I_\alpha$ and noise. Specifically, $y_{i,s} = z_{\beta_i} I_\alpha(x_s) + \sigma\epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$ are independent of observation location $s \in \Gamma$.
5: **end for**

Parameter values for prior distributions need to be chosen, in particular Σ_p and Σ_g. On page 8 of [1], it is suggested that these square matrices of dimensions k_p and $2k_g$ are chosen with respect to the matrices induced by the kernels,

$$M_p(k,k') = K_p(x_{p,k}, x_{p,k'})$$
$$M_g(k,k') = K_g(x_{g,k}, x_{g,k'}).$$

In particular, it is suggested that $\Sigma_p = M_p^{-1}$ and Σ_g be the matrix with blocks $\begin{bmatrix} M_g^{-1} & 0; & 0 & M_g^{-1} \end{bmatrix}$. To illustrate the reasonableness of this choice, consider the covariance of x-deformations at two deformation landmark points of $x_{g,k}$ and $x_{g,k'}$,

$$\mathbf{E}(D(k)D(k')) = \mathbf{E}\left(\sum_{1\le j\le k_g} K_g(x_{g,j}, x_{g,k})\beta^{(1)}(j) \sum_{1\le j'\le k_g} K_g(x_{g,j'}, x_{g,k'})\beta^{(1)}(j') \right)$$

Letting $M_{k,:}$ denote the kth row of M_g, and $\beta = (\beta^{(1)}(1) \cdots \beta^{(1)}(k_g))$ this becomes

$$\begin{aligned} &= \mathbf{E}\left(M_{k,:}\beta\, M_{k',:}\beta\right) \\ &= \mathbf{E}\left(M_{k,:}\beta\beta^T M_{:,k'}\right) \\ &= M_{k,:}M^{-1}M_{:,k'} \\ &= M_g(k,k'). \end{aligned}$$

It is therefore natural to choose Σ_g if deformations are expected to have covariance M_g. In terms of correlations between deformations this seems reasonable, points nearer to each other being similarly deformed. It is possible however that Σ_g should be multiplied by a constant depending on the relation between expected deformation and kernel magnitude.

Parameter estimation

For a set of observations $\mathbf{y}$ coming from a single species, the objective is to find parameters maximizing the posterior likelihood, $q(\theta|\mathbf{y})$. To find these it is necessary to marginalise over the *latent* deformation variables β associated with observations $\mathbf{y}$, as observed in the following decomposition of $q(\theta|\mathbf{y})$,

$$\begin{aligned} q(\theta|\mathbf{y}) &\propto q(\mathbf{y}|\theta)q(\theta) \\ &= \int q(\mathbf{y}, \beta|\theta)q(\theta)\, d\beta. \end{aligned} \tag{22}$$

It is common to solve optimisation problems containing latent variables such as this using an Expectation-Maximisation (EM) algorithm, as described in the following subsection.

The EM algorithm

In a general setting [11], the EM algorithm is used for finding the maximum (mode) of

$$F(\theta) = \int f(\beta, \theta) d\beta. \tag{23}$$

There are two main versions of the EM algorithm, referred to as the *hard* EM and the *soft* EM. The hard EM algorithm, also called the 'Fast approximation with modes' in [1] proceeds as follows:

Algorithm 7 The *hard EM*, or *Fast approximation with modes*

1: Initialize the parameters θ to some starting estimate
2: **for** $i = 1$ to N **do**
3: Compute the β which maximises f with the current θ estimate
4: Compute the θ which maximises f with the current β estimate
5: **end for**

The hard EM finds an approximation of the optimal solution. There are two potential pitfalls with this method. The first is getting stuck at locally maximal solutions, more likely in cases of high noise. The second risk is that the optimal solution may have a relatively narrow support in θ at the optimal β making the integral (23) sub maximal. The soft EM algorithm replaces the two steps on lines 3 and 4 of Algorithm 7 by what are referred to as the E- and M- steps:

- **E-step.** Compute

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E_t\left[\log f(\beta, \theta)\right] \\ &= \int \log f(\beta, \theta) f(\beta, \theta^{(t)}) / F(\theta^{(t)})\, d\beta. \end{aligned} \tag{24}$$

- **M-step.** Find $\theta^{(t+1)}$ to maximize $Q(\theta|\theta^{(t)})$.

In the context of the original problem (22) where $f(\beta, \theta) = q(\mathbf{y}, \beta|\theta)q(\theta)$, the function Q of the E-step is,

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E_t \left[\log \left(q(\mathbf{y}, \beta|\theta)q(\theta)\right)\right] \\ &= \int \left(\log \left[q(\mathbf{y}, \beta|\theta)\right] + \log \left[q(\theta)\right]\right) q(\beta|\theta^{(t)}, \mathbf{y})\, d\beta. \end{aligned} \tag{25}$$

The parameters $\theta = (\alpha, \Gamma, \sigma)$ are expanded out of (25) resulting in terms separating,

$$\begin{aligned} Q(\alpha, \sigma, \Gamma|\theta^{(t)}) = \int (&\log \left[q(y|\beta, \alpha, \sigma)\right] + \log \left[q(\beta|\Gamma)\right] + \log \left[q(\Gamma)\right] + \\ &\log \left[q(\sigma, \alpha)\right])q(\beta|\theta^{(t)}, \mathbf{y})\, d\beta. \end{aligned} \tag{26}$$

where

$$q(\beta|\theta^{(t)}, \mathbf{y}) = \prod_{i=1}^{N} q(\beta_i|\theta^{(t)}, y_i)$$

and for the ith observation,

$$q(\beta_i|\theta^{(t)}, y_i) = \frac{q(y_i|\beta_i, \alpha^{(t)}, \sigma^{(t)})q(\beta_i|\Gamma^{(t)})}{\int q(y_i|\beta', \alpha^{(t)}, \sigma^{(t)})q(\beta'|\Gamma^{(t)})d\beta'}.$$

The separation of variables in equation 26 enables us to decompose the M-step into two parallel parts. For the geometric parameter Γ we have:

$$\Gamma^{(t+1)} = \arg\max_{\Gamma} \int \log \left[q(\beta|\Gamma)\right] q(\beta|\theta^{(t)}, \mathbf{y})\, d\beta + \log \left[q(\Gamma)\right] \tag{27}$$

and for the photometric parameters α and σ we have

$$\alpha^{(t+1)}, \sigma^{(t+1)} = \arg\max_{\alpha,\sigma} \int \log \left[q(y|\beta, \sigma, \alpha)\right] q(\beta|\theta^{(t)}, \mathbf{y})\, d\beta + \log \left[q(\sigma, \alpha)\right] \tag{28}$$

On page 12 of [1] it is noted with example that the soft EM provides better estimates than the hard EM. However, the hard EM method can be used successfully, as shown in an example of digit classification. As described in the text, we have implemented a version of the hard EM algorithm.

Implemenation, and analysis of order of operations

We repeat here the scheme of our implemenation via approximation with modes as Algorithm 8,

Algorithm 8 Approximation with modes in noiseless model

for $j = 1$ to n-images **do**
  $\beta_{0j} \leftarrow 0$
end for
for $i = 0$ to n-steps **do**
  Compute α_i which minimizes $\sum_{j=1}^{\text{n-images}} \|y_j - K_p^{\beta_{ij}}\alpha_i\|^2$ (α-update)
  for $j = 1$ to n-images **do**
    Compute the $\beta_{(i+1)j}$ which minimises $\|y_j - K_p^{\beta_{(i+1)j}}\alpha_i\|^2$ (β-update)
  end for
end for

In Algorithm 8, K_p^β will be a sparse matrix with number of non-zero elements of order $O(k_p \tilde{\sigma_p}^2)$. The α-update has solution,

$$\alpha = \left(\sum_{j=1}^{\text{n-images}} K_p^{\beta_{ij}\,T} K_p^{\beta_{ij}} \right)^{-1} \left(\sum_{j=1}^{\text{n-images}} K_p^{\beta_{ij}\,T} y_j \right). \tag{29}$$

This assumes that the inverted matrix in (29) is indeed invertible. As we do not extend the photometric landmark range as is done in [1] this is a fair assumption, but in the case where a kernel around a photometric landmark point is zero for all deformations of $x_{s(s\in\Lambda)}$, $K_p^{\beta_{ij}}$ will have a column of zeros and the matrix in (29) will not be invertible. The α-update algorithm is given by,

Algorithm 9 α-update

1: $[K^T K] \leftarrow 0$
2: $[K^T Y] \leftarrow 0$
3: **for** $j = 1$ to n-images **do**
4: Calculate shifted coordinates $x'_s = x_s - \sum_{l=1}^{k_g} \beta(l) K(x_s, x_{g,l})$. $O(|\Lambda| k_g)$
5: Calculate $K_p^{\beta(j)}$. $O(|\Lambda| k_p)$
6: $[K^T K] \leftarrow [K^T K] + K_p^{\beta(j)\,T} K_p^{\beta(j)}$
7: $[K^T Y] \leftarrow [K^T Y] + K_p^{\beta(j)\,T} y_j$
8: **end for**
9: $\alpha = [K^T K]^{-1} [K^T Y]$. $O(k_p^{2.8})$

The order of the number of operations is given where relevant. The factor of 2.8 on line 9 is required for matrix inversion using the Strassen algorithm. In total, the requirement is $O\left(k_p^{2.8} + \text{n-images}(|\Lambda| k_p + |\Lambda| k_g)\right)$. The β-update algorithm which we will now present is to be more time consuming for the image sizes we consider than the α-update step. The β-update will be performed with repeated steps of steepest descent, that is with repeated steps of $\beta^{(new)} = \beta^{(old)} + \lambda D$ where the ith component of D is $D_i \propto \frac{\partial}{\partial \beta_i} \| y_j - K_p^\beta \alpha \|^2_{\beta=\beta^{(old)}}$, where β_i the ith element of the $2k_g$ element β-vector, the first k_g elements being those related to x-direction deformation, and the last k_g being those related to the y-deformation. Dropping all unnecessary sub- and super- scripts, we have

$$D_i = (y - K_p^\beta \alpha)^T \left[\frac{\partial K_p^\beta}{\partial \beta_i} \right] \alpha.$$

where we calculate

$$\left[\frac{\partial K_p^\beta}{\partial \beta_i} \right]_{st} = \frac{\partial \left(K_p^\beta \right)_{st}}{\partial r^2} \cdot \frac{\partial r_{st}^2}{\partial \beta_i}. \tag{30}$$

Letting $r_{st}^2 = dx_{st}^2 + dy_{st}^2$, the second term in the product above becomes,

$$\begin{aligned} \frac{\partial r_{st}^2}{\partial \beta_i} &= -2\, dx_{st} \cdot K_g\left(x_s, x_{g,i}\right) && \text{for } 1 \le i \le k_g, \\ \frac{\partial r_{st}^2}{\partial \beta_i} &= -2\, dy_{st} \cdot K_g\left(x_s, x_{g,i}\right) && \text{for } k_g + 1 \le i \le 2k_g. \end{aligned} \tag{31}$$

Combining (30) and (31) we have (without loss of generality we consider from now on only $1 \leq i \leq k_g$, that is x-direction deformations),

$$\left[\frac{\partial K_p^{\beta}}{\partial \beta_i}\right]_{st} = -2\left(\frac{\partial\left(K_p^{\beta}\right)_{st}}{\partial r^2} \cdot dx_{st}\right) \cdot K_g\left(x_s, x_{g,i}\right). \tag{32}$$

Equation 32 naively indicates to us that the calculation of D will require $O(|\Lambda|\, k_p\, k_g)$ operations: A matrix of size $|\Lambda| \times k_p$ for each of the $2k_g$ dimensions of D. However the number of operations can be reduced by taking advantage of certain sparsities which emerge. The bracketed term in eqn. 32 is a matrix which is constant for all i, and has $O(k_p\tilde{\sigma}_p^2)$ non-zero elements, due to the compactness of the photometric kernel. For each of the $2k_g$ dimensions of D, each row of the bracketed matrix corresponding to a given $s \in \Lambda$ gets multiplied by the constant $K_g\left(x_s, x_{g,i}\right)$. Only $O(\tilde{\sigma}_g^2)$ of the $|\Lambda|$ elements of the vector $K_g\left(x_s, x_{g,i}\right)$ are non-zero, so the number of operations per dimension of D which need to be performed is $O(k_p\tilde{\sigma}_p^2\tilde{\sigma}_g^2/|\Lambda|)$, and hence a total number of operations of order $O(k_p k_g\tilde{\sigma}_p^2\tilde{\sigma}_g^2/|\Lambda|)$ are needed to calculate D. In practice one chooses $k_g \propto |\Lambda|$ and $k_p \propto |\Lambda|$, and so the number of operations is $O(\tilde{\sigma}_p^2\tilde{\sigma}_g^2|\Lambda|)$. This is a marked improvement over the case where sparsity is ignored, where the number of operations is $O(|\Lambda|^3)$. In Algorithm 10 the method for finding the direction of steepest descent is presented, with indications of where sparsity should be used. In Algorithm 11 the full algorithm for updating β is given.

Algorithm 10 Find direction of steepest descent for β-update

1: Input α, β and image y
2: Calculate the $|\Lambda| \times k_g$ matrix $K_g(\mathcal{I}, \mathcal{G})$ (actually as this calculation is β-independent, it is a one-off calculation which is done outside)
3: Calculate the $|\Lambda|$ deformation locations $z_\beta\mathcal{I} = \mathcal{I} - K_g(\mathcal{I}, \mathcal{G})\beta_j$
4: Calculate the $|\Lambda| \times k_p$ vector differences, $\vec{\mathcal{R}}[s, l] = \mathcal{P}[s] - z\mathcal{I}[l]$
5: Define $\mathcal{R}_x \leftarrow$ (x-component of $\vec{\mathcal{R}}$) and $\mathcal{R}_y \leftarrow$ (y-component of $\vec{\mathcal{R}}$)
6: Define $\mathcal{R}^2 \leftarrow \mathcal{R}_x^2 + \mathcal{R}_y^2$
7: Calculate the $|\Lambda| \times k_p$ matrix of kernel values, $K_p(\mathcal{R}^2)$.
8: Calculate the $|\Lambda| \times k_p$ matrix $\dfrac{\partial K_p}{\partial R^2}$ where $\dfrac{\partial K_p}{\partial R^2}[s, l] = \dfrac{\partial K_p}{\partial r^2}|_{r=\mathcal{R}[s,l]}$
9: Calculate $\mathcal{Q}_x = \dfrac{\partial K_p}{\partial R^2} \cdot \mathcal{R}_x$ and $\mathcal{Q}_y = \dfrac{\partial K_p}{\partial R^2} \cdot \mathcal{R}_y$ and store as sparse matrices.
10: Calculate $v = y - K_p(\mathcal{R}^2)\alpha$, where y is the observed image.
11: **for** $1 \leq j \leq k_g$ **do**
12: Calculate the (extremely) sparse matrix $\mathcal{S}_j$ of size $|\Lambda| \times k_p$ whose sth row is $\mathcal{Q}_x[s, :] \cdot K_g[s, j]$
13: $D_j = (S_j\alpha)^T v$
14: **end for**
15: **for** $k_g + 1 \leq j \leq 2k_g$ **do**
16: Calculate $\mathcal{S}_j$ where now the sth row is $\mathcal{Q}_y[s, :] \cdot K_g[s, j]$
17: $D_j = (S_j\alpha)^T v$
18: **end for**
19: Return direction D.

For our implementation, the kernel used (5) in spherical coordinates is,

$$K_p(r) = \cos^2\left(\frac{\pi r^2}{2\tilde{\sigma_p}^2}\right)\mathbf{1}_{[-\pi/2,\pi/2]}(|r|).$$

The only kernel specific quantity which is needed in the calculation of the direction is $\frac{\partial K_p}{\partial r^2}$ which in our case is given by,

$$\frac{\partial K_p(r^2)}{\partial r^2} = \frac{\pi}{\tilde{\sigma}_p^2}\sin\left(\frac{\pi r^2}{2\tilde{\sigma_p}^2}\right)\mathbf{1}_{[-\pi/2,\pi/2]}(|r|).$$

Algorithm 11 Steepest descent for β-update

1: Input $\alpha, \beta^{(0)}$ and image y.
2: **for** $0 \leq w \leq$n_descents - 1 **do**
3: Calculate direction of steepest descent $D^{(w)}$ at $\beta^{(w)}$ using Algorithm 10.
4: Find $\lambda^{(w)} \approx \arg\min_{\lambda} |y - K_p^{\beta^{(w)}+\lambda D^{(w)}}\alpha|^2$ using a golden section search of 10 steps.
5: Set $\beta^{(w+1)} \leftarrow \beta^{(w)} + \lambda^{(w)} D$
6: **end for**
7: Return $\beta^{(\text{steps})}$.

In Figure 10 the performance time of a single direction finding step treating matrices as sparse and non-sparse (dense) is compared. The number of photometric and geometric landmark points were chosen to scale with Λ, specifically $k_p \sim 0.5\Lambda$ and $k_g \sim 0.3\Lambda$. The bandwidth of the photometric and geometric kernels were chosen to be 2.1 and 1.5 times the distance between photometric and geometric gridpoints respectively. A quick calculation shows that these bandwidths imply that each (internal) photometric kernel covers 13 photometric landmark points and each (internal) geometric kernel covers 9 geometric landmark points. In the speed comparison $|\Lambda|$ was taken over 25 values between $10^2 \cdots 85^2$. Our preceding analysis suggested that the sparse method should require only $O(|\Lambda|^1)$ operations *for lines 11 to 17*. In effect, the number of loops on these lines is $O(|\Lambda|^1)$, with an $O(1)$ operation for each loop. However, there are several $O(|\Lambda|^2)$ operations which are performed before entering the loops. These $O(|\Lambda|^2)$ lines (3 to 9) are the same for the sparse and dense methods. Figure 10 illustrates the time requirements, decomposed by a) lines 3 to 9 (common for dense and sparse methods) and b) the loops (distinct for dense and sparse methods). Finding the gradients of the log-log plots in the right panel of Figure 10 provides the expected values:

- Pre-loop: $O(2.016 \pm 0.014)$
- Loop (Sparse): $O(1.019 \pm 0.009)$
- Loop (Dense): $O(3.021 \pm 0.061)$

The $O(|\Lambda|^3)$ loop operation has been reduced to a $O(|\Lambda|)$ operation, is it not possible to reduce the $O(|\Lambda|^2)$ pre-loop operation? It may be possible, but it is not clear to me how it can be done. The problem is that the sparsity structure of $K_p(\mathcal{R}^2)$, the sparse matrix on line 7, depends on β: it is theoretically possible for any observation point x_s to be deformed to a position close to any photometric landmark point x_p. The same matrix

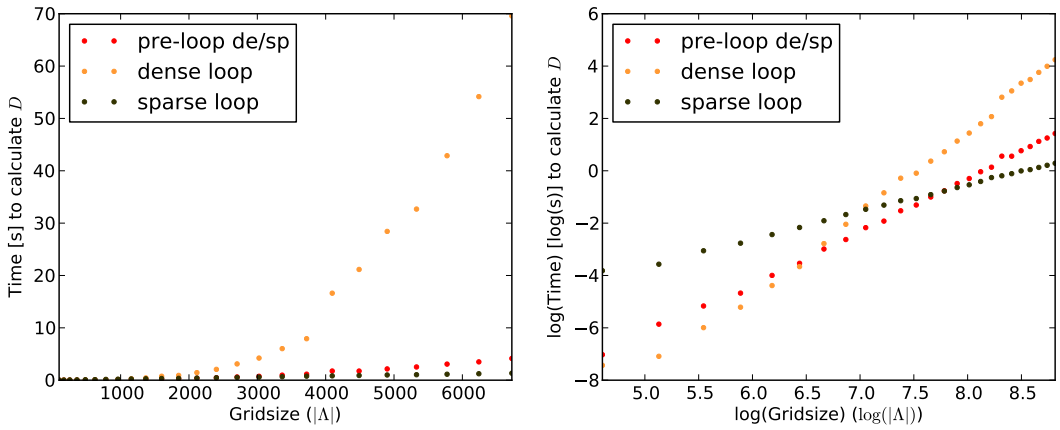


Figure 10: Time for calculating direction of steepest descent, decomposed by $O(|\Lambda|^2)$ and $O(|\Lambda|^3)/O(|\Lambda|^1)$ calculations.

appears on line 4 of Algorithm 11, and this time it needs to be calculated several times to esetimate the optimal distance to along D. In summary, the sparse β-update step is $O(\text{n\_descents} \times \text{n\_images}(|\Lambda|k_g + |\Lambda|k_p))$.

Multiclass Paramater estimation

Certain species have several distinct intraspecies classes of chirps, and therefore a model where all chirps of a single species are a deformation of a single deformable template is not realistic. In this section the single template model is extended to allow for several templates within one species. In particular, suppose that there are T distinct classes of chirps for a given species, and that the probability of chirps belonging to class τ is ρ_τ. The parameters defining the full diversity of species chirps is therefore,

$$\theta = (\theta)_{1\leq\tau\leq T} \quad \text{and} \quad \rho = (\rho(\tau))_{1\leq\tau\leq T}$$

where $\theta_\tau = (\alpha_\tau, \sigma_\tau, \Gamma_\tau)$ as in the single template case. The choice of prior distributions for the parameters in θ has been discussed. For the prior on ρ, the Dirichlet distribution is chosen,

$$\nu_\rho(\rho) \propto \left[\prod_{\tau=1}^{T} \rho(\tau)\right]^{a_\rho},$$

where larger values of a_ρ indicate higher prior belief in equal class probabilities. The model for generating species data now has one additional level:

Algorithm 12 Model for generating a species template and observations

1: Draw $(\rho)_{1\leq\tau\leq T} \sim \nu_\rho$
2: Generate $\theta_{1\leq\tau\leq T}$ as before
3: **for** $i = 1$ to N **do**
4: To generate observation y_i, draw τ_i from class τ with probability ρ_i and then β_i using Γ_{τ_i}, and finally y_i as a deformed template with added noise.
5: **end for**

The objective is to recover θ and ρ for each class of bat chirp. The latent variables are now the β_i and τ_i associated with each observation. The E-step thus becomes the estimation of the function,

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E_t\left[\log\left(q(\mathbf{y},\beta,\tau|\theta,\rho)q(\theta,\rho)\right)\right] \\ &= \sum_{\tau}\int\left(\log\left[q(\mathbf{y},\beta,\tau|\theta,\rho)\right]+\log\left[q(\theta,\rho)\right]\right)q(\beta,\tau|\theta^{(t)},\rho^{(t)},\mathbf{y})\,d\beta. \end{aligned} \tag{33}$$

where $q(\beta,\tau|\theta^{(t)},\rho^{(t)},\mathbf{y})$ is now the product over all observations i of

$$q(\beta_i,\tau_i|\theta^{(t)},y_i) = \frac{q(y_i|\beta_i,\alpha^{(t)}_{\tau_i},\sigma^{(t)}_{\tau_i})q(\beta_i,|\Gamma^{(t)}_{\tau_i})\rho_i}{\sum_{\tau'}\int_{\beta'} q(y_i|\beta',\alpha^{(t)}_{\tau'},\sigma^{(t)}_{\tau'})q(\beta',|\Gamma^{(t)}_{\tau'})\rho_i} \tag{34}$$

On expanding out θ in (33) as was done previously, expressions of the form (27) and (28) can be found.

Results illustrated

Figure 11 illustrates selected iterations of Algorithm 7. We will refer to subfigures as (row, column). In (1,1) is the unobserved true template, where $|\Lambda| = 25^2, k_p = 15^2, g_p = 5^2$. Illustrated in (1,2) are generated observations from this template, where Γ is chosen as $0.004\,\Sigma_g$. The mean image is illustrated in (2,1). As the mean image corresponds to the optimal α where all βs are zero, it is the approximation of the template at the first iteration. Then in (2,2) the optimal βs for the mean image are presented. Thereafter α and the βs are updated several times, and in (4,1) and (4,2) the 10th iterates are illustrated.

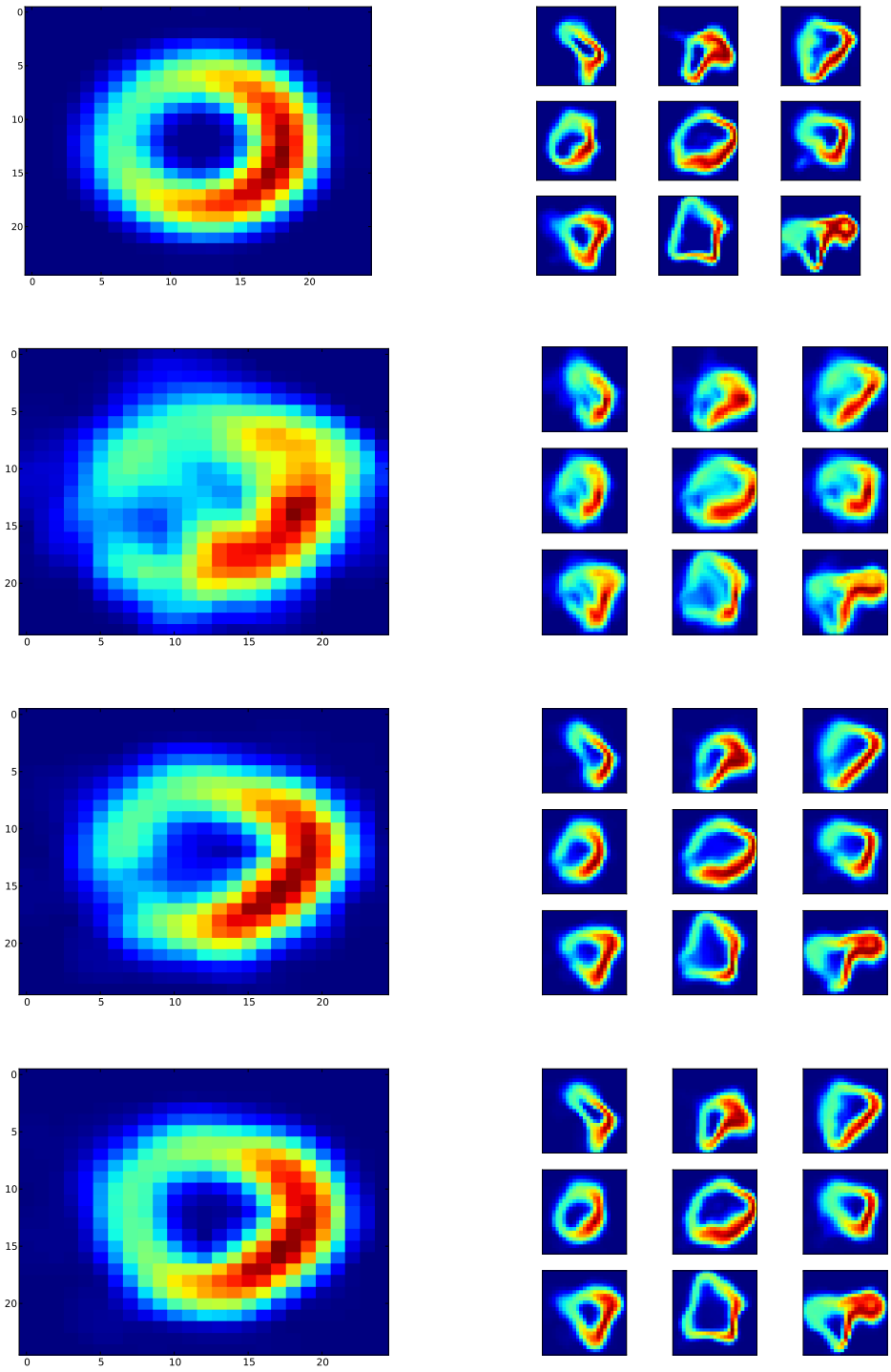


Figure 11: Successful β-matching. The problem is, given the observations (top right) to reconstruct the template (top left) and find the 9 corresponding β vectors which deform the template into the observations. Details explained in text.

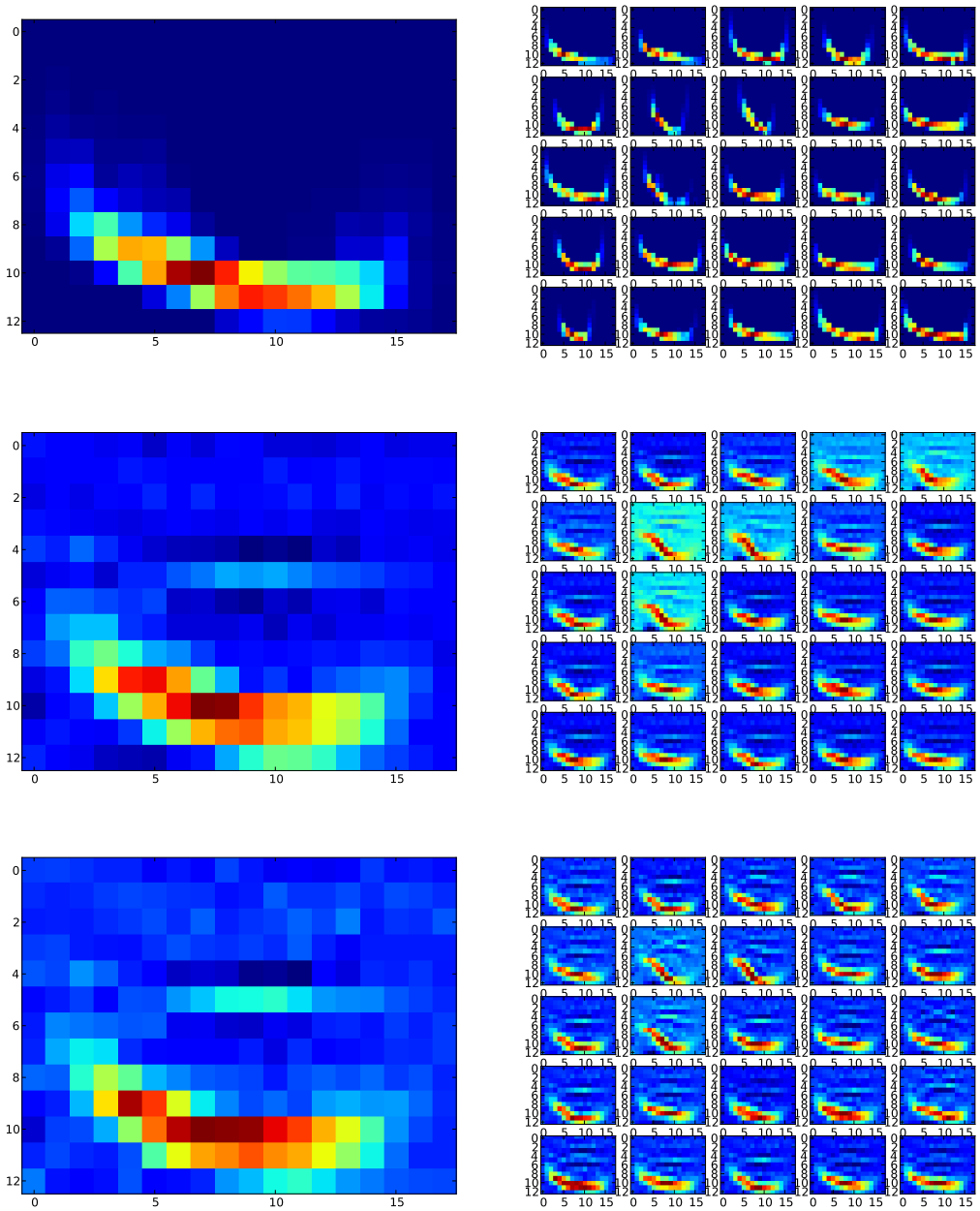


Figure 12: β-matching with 25 observed Saccopteryx leptura chirps. Figure (1,1) is the mean of the observations, and Figure (2,1) are the observations. In Figure (2,1) is the best fitting kernel function to the mean observation, and Figure (2,2) after the first *beta*-iteration. The third row is after 6 iterations, a local minimum has been found.