Fast, Robust Classification with Applications to Periodic Variable Stars

Siqi Wu, Mark Rogers, and James Long

May 9, 2012

1 Introduction

2 The Data

2.1 Background on Periodic Variables

Modern astronomical surveys observe millions of light sources (stars, galaxies, asteroids) over the course of a mission lasting a few years. Periodic variables, sources which vary periodically in brightness over time, are some of the most interesting. In the 1920's, periodic variables were crucial in Edwin Hubble's discovery the existence of galaxies [**?**]. More recently, periodic variables have played an important role in determining expansion of the universe [**?**].

Periodic variables may be divided into a few dozen classes based on physical properties of light sources. Separating the sources into classes is a critical step in turning raw astronomical observations into scientific knowledge. The size of modern data sets requires that much of this work be automated by machine learning and statistical classifiers.

Figure 1 displays the light curve (i.e. time series) of a periodic variable belonging to the class Classical Cepheid. The points in the top plot represent flux measurements in magnitudes (i.e. brightness of the source) made by the telescope at particular times. The 0 point on the time axis is arbitrary. Using fourier methods, one can estimate a period using these measurements. The lower plot of Figure 1 displays the flux measurements of the same object. However here the x-axis is phase of each time measurement, computed using the estimated period of 4.51 days. Here we can observe the structure of the periodic variation. This is known as the *folded light curve*.

Figure 2 displays an example of a Mira light curve. From the y-axis we can see that this source has higher amplitude than the Classical Cepheid (this is typical of the Mira class) and more sinusoidal variation (also typical). Note that the fourier methods appear to have estimated an incorrect period for this source. The true period appears to be around 161 days, half of the estimate.

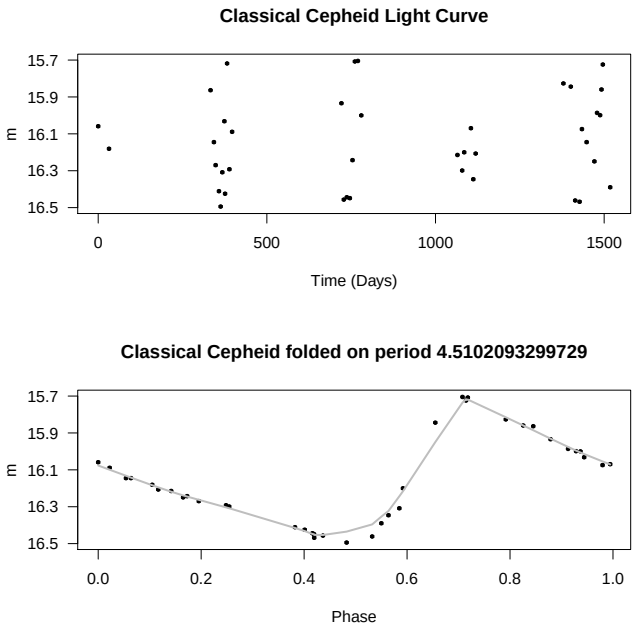

Figure 1: Light curve of a Classical Cepheid type variable star. The brightness of the star is on the y-axis. For the top plot the brightness is plotted against time. This time series is periodic. Using fourier methods we can estimate a period of 4.51 days. We can convert the times from the top plot into phase ((time modulo period) / period)). Brightness versus phase is plotted in the bottom plot. Here we can observe the structure in the time series, which is usually similar for stars of the same class but different for stars of different classes.

2.2 Classification Methodology

There has been a lot of recent progress in the astronomy literature towards developing highly accurate classifiers for periodic variables [**?**, **?**, **?**]. The standard approach works as follows. A telescope observes a source j at times $t_1, \ldots, t_l$, recording flux measurements of $m_1, \ldots, m_l$. Typically there are measurements of uncertainty on the flux measurements $e_1, \ldots, e_l$. So each source j is initially characterized by an $l \times 3$ matrix $D_j = \{(t_i, m_i, e_i)\}_{i=1}^{l}$. Note that l, the number of times the source is observed, is different for each D_j. Also the time sampling of the flux measurements is irregular and different for each source. Associated with each source is a classification, such as Mira, Classical Cepheid, RR Lyrae, etc.

In order to construct a classifier, features are *extracted* from D_j (i.e. take functions of D_j) that will separate sources into different classes. Features vary from study to study,

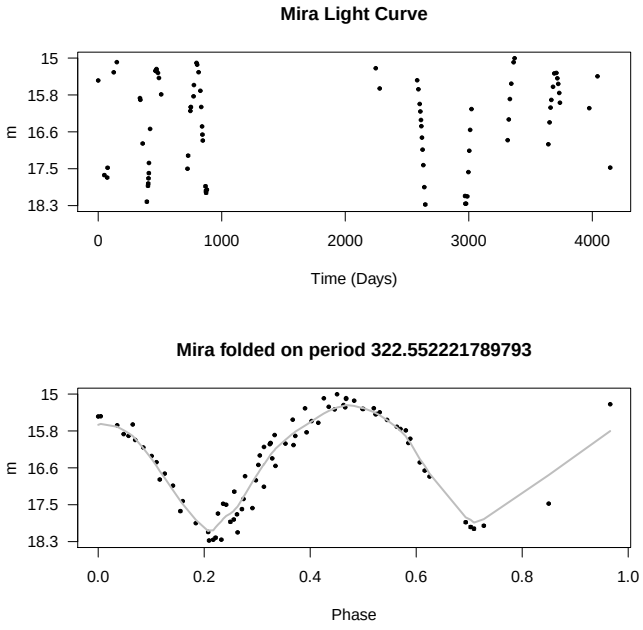

Figure 2: Light curve of a Mira type variable star.

but typical ones include period (inverse of strongest fourier frequency), amplitude, skew, and estimates of derivatives.

If we compute p features and have a total of n training stars of known class $(D_1, \ldots D_n)$, then we can use standard classification techniques on this $n \times p$ data matrix to construct a classifier. Given this classifier, we can then assign a class to a new source by extracting features and running the features through the classifier.

2.3 Uncertain Features

A major problem with the standard approach is the high levels and heteroskedastic nature of the uncertainty in the features due to having poorly sampled time series (l is small), high error in the flux measurements (e_i are systematically large), or irregular nature of sampling (t_i's are highly concentrated, giving a poor representation of variation). This leads to situations where features meant to represent physical quantities of the time series, such as amplitude or period, are incorrect. As an example in Figure 3 we plot the period and amplitude features for two classes of stars with a box around known physical limits for these features for these two classes. Some of the observations have values outside these physical limits, clearly indicating that the features are incorrectly estimated. This suggests

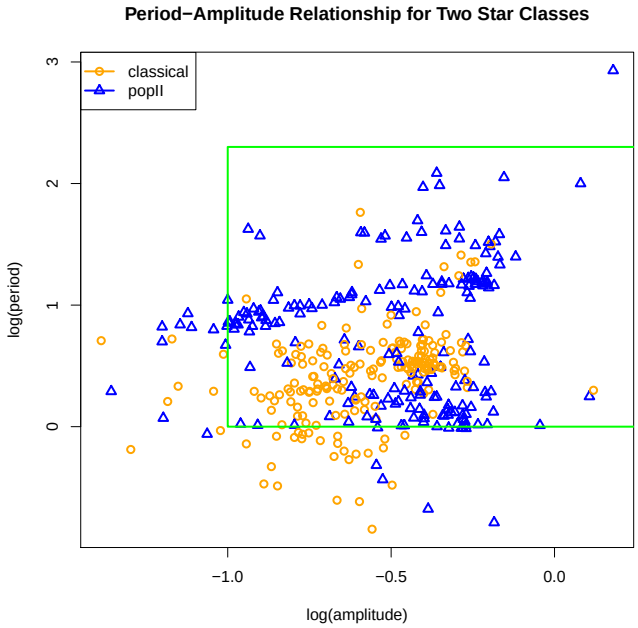

Figure 3: Period-Amplitude relationship for two classes: *Classical Cepheid* stars and *Population II Cepheid* stars. The green box represents (roughly) the physically possible range of periods and amplitudes. Clearly some of these features have been estimated incorrectly.

that classifiers which incorporate feature uncertainty may be able to acheive improved performance.

3 SVMs for Interval Data

Support Vector Machines (SVMs) are a popular method for constructing classifiers. Let $x_i \in \mathbb{R}^p$ be the vector of features for observation i and y_i its class (either -1 or $+1$), we can determine the SVM classifier by solving the optimization problem,

$$\min_{\beta,\beta_0} \sum_{i=1}^{n} (1 - y_i(\beta^T x + \beta_0))_+ + \frac{\lambda}{2}\beta^T\beta \tag{1}$$

The optimal β^* and β_0^* are used to classify a new observation x using $sign(\beta^* x + \beta_0)$. λ is a tuning parameter that balances a tradeoff between maximizing the margin and separating observations in different classes. An optimal λ may be chosen using 0-1 loss on a test set or through cross-validation. See [**?**] for an extensive overview of SVMs.

Various extensions to the standard SVM have been proposed that seek to incorporate uncertainty in the features into the optimization problem. These formulations often take (or can be interpreted) as minimizing the objective function for a worst case allocation of features in some fixed set, i.e.

$$\min_{\beta,\beta_0} \max_{z \in \mathcal{X}} \sum_{i=1}^{n} (1 - y_i(\beta^T z_i + \beta_0))_+ + \frac{\lambda}{2}\beta^T\beta \tag{2}$$

See [**?**, **?**, **?**] for discussions of some of these models.

TO DISCUSS:

1. similarities to elastic net of LASSO / SVM e.g. Wang [**?**]
2. our algorithm is not contained (?) in those considered by Russet [**?**]

4 Path Algorithm for Interval SVM

5 Application to Variable Star Data Sets

5.1 Constructing the Intervals from the Time Series

In certain applications, methods for constructing hyper-rectangles (or hyper-spheres) in feature space may be fairly straitforward. For example in El Ghaoui [**?**], the authors discuss applications to micro-array data where several replicates of some experiment are available, and a hyper-rectangle can be chosen to enclose all replicates.

With astronomy data there is no clearly correct way to construct intervals around features. A few possibilities include:

1. Make width of feature interval proportional to number of measurements in time series. Long time series will have small intervals around features, short time series will have wide intervals. For simple features, such as the standard deviation of the brightness measurements, making the interval width go down at rate root-n has a probabilistic interpretation via the central limit theorem.
2. Subsample the time series and derive features for each subsample. Represent observations as hyper-rectangles that contain features derived from every subsample (or a certain fraction of the subsamples).
3. Heuristically put intervals around observations with features that are wrong. For example, the amplitude of a star cannot be greater than 6 magnitudes, so any star with amplitude greater than 6 mags could be given an interval that contains values less than 6.

In this work we take the second approach. While the first approach is attractive for simple features, many features (such as frequency) do not have a sqrt-n convergence to the true value. Further the irregular time sampling means that even simple features may not have a root-n convergence. The third approach involves a lot of domain knowledge and will change from application to application. Further, there is no guarantee that features within the range of what is physically possible are actually correct.

Figure 4 describes precisely how we subsample the time series and determine interval widths. The details of this procedure could be changed. For example one could bootstrap sample the time measurements, instead of subsampling contiguous sections. Certain computational considerations enter here. It takes around 1 second to derive features for an average length light curve. So sampling, say 20 times, could become prohibitively expansive if the data set is initially at the limits of what is computationally feasible.

Constructing Hyper-rectangles from Time Series

1. Order the light curve measurements in time i.e. $\{(t_1, m_1, e_1), (t_2, m_2, e_2), \ldots, (t_l, m_l, e_l)\}$ where $t_i < t_j$ for $i < j$.
2. Slice each light curve into 5 contiguous sections, producing 5 time series. Slice 1 is (t_i, m_i, e_i) for $\frac{0}{10}l \leq i \leq \frac{5}{10}l$, slice 2 is $\frac{1}{10}l \leq i \leq \frac{6}{10}l$, ect.
3. Derive features for each of these slices.
4. Put the interval minimum at the lowest value of the feature for the 5 slices. Put the interval max at the maximum feature value across the 5 slices.

Figure 4: Description of interval construction algorithm.

5.2 The Data Sets

The first dataset analyzed contained two classes of Cepheids (CEP): classical and [popII]. Each class consisted of $n = 200$ observations and $p = 58$ features so that each observation is hyperrectangle in $\mathbb{R}^{58}$. The second dataset consisted of two classes of RR, RR Lyrae AB and RR Lyrae D, with the same values of n and p.

5.2.1 Preprocessing the data with quantile ranking

A quantile ranking technique was used to normalize values in each feature dimension by reducing the effects of outliers. Viewing the interval data as a three-dimensional array, the end-slices corresponding to the lower- and upper-bounds of the confidence intervals were isolated and then concatenated together to yield a $2n \times p$ array whose values were then mapped to the standard normal distribution. Features containing all zeroes or some NaNs were removed from the dataset.

5.3 Results

5.3.1 Performance

The performance of the combined SVM model was compared to those of its degenerates by computing expected losses for multiple values of λ and ρ. For a given (λ, ρ), the expected loss was computed by performing cross-validation in 90-10 splits so that the union of testing points across splits is the entire dataset and the intersection is disjoint. See surface plots of loss versus (λ, ρ): for the CEP dataset, $\lambda \in \{0\} \cup [0.375, 0.750]$ and $\rho \in \{0\} \cup [0.090, 0.032]$ while for the RR dataset, $\lambda \in \{0\} \cup [54, 284]$ and $\rho \in \{0\} \cup [0.375, 0.750]$. Observe that $(\lambda^*, \rho^*) \neq (0, 0)$ for both datasets, which implies that the combined model outperforms both of its degenerates. For the CEP dataset, the optimal $(\lambda^*, \rho^*) = (9, 0.320)$ yielded 13.25% classification error while for the RR dataset, $(\lambda^*, \rho^*) = (284, 0.48)$ yielded 21.50% classification error. THESE NUMBERS DON'T LOOK RIGHT, MUST RERUN analyze$_f eature.m$.

The curves of loss versus λ and loss versus ρ were also analyzed. The loss versus λ plot corresponds to the standard point SVM model in Hastie while the loss versus ρ plot corresponds to the interval SVM model in El Ghaoui. Observe the convex-like curvatures, which suggest global minima and hence optimal λ and ρ.

5.3.2 Feature selection as suggested by $||\beta||_0$ values

5.4 Conclusions

1. constructing intervals is non-trivial in many high dimensional problems
2. not clear that this robust framework is ideal when new observation might have low error on certain features
3. kernels, are they possible to use
4. links to code / git repo