Classification of Poorly Time Sampled Light Curves of Periodic Variable Stars

James P. Long, Josh S. Bloom, Nouredine El Karoui, John Rice, Joseph W. Richards

UC Berkeley

June 1, 2011

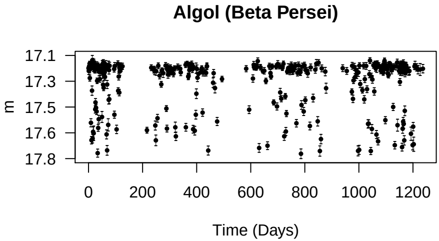
Outline

- classification of variable stars important
  - scientific knowledge discovery
  - allocation of telesopic resources
- size of data sets require statistical / machine learning methods

Classifying Time Series

1. compute real valued functions, termed *features*, of the time series
   - fourier coefficients
   - skew, standard deviation, amplitude, ect.
   - context features - where in the sky was source observed
2. each lightcurve becomes a vector in $\mathbb{R}^p$
3. apply classification methods such as Random Forests, Naive Bayes, ect.

Approach taken by [1, 3, 2].



	class	frequency	max_slope	skew	amp.
1	Algol (Beta Persei)	1.0871418	72	1.4017419	0.3195
2	Multiple Mode Cepheid	1.7341435	86	-0.8146807	0.2185
3	Algol (Beta Persei)	0.4750477	134	2.0813171	0.2090
4	Beta Lyrae	0.3799993	7	2.6630483	0.5550
5	Algol (Beta Persei)	0.7836879	130	2.5590869	0.4585

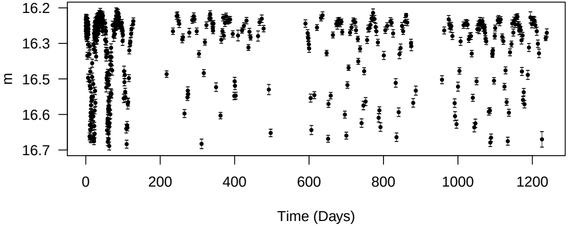
Problem

- labeled data from catalogs (the *training set*) have hundreds of flux measurements
  - OGLE
  - *Hipparcos*
- unlabeled data from ongoing / upcoming surveys (the *test set*) have many fewer flux measurements
  - GAIA
  - LSST

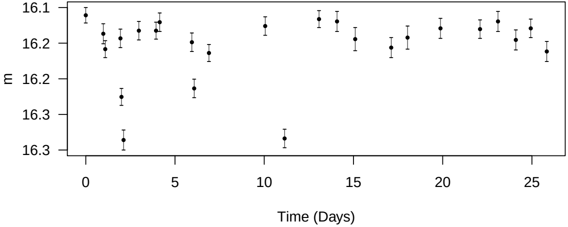
Problem

- the conditional distribution $p(class|features)$ may be different in *test* and *training* sets
  - features in test data have error
- but classifiers assume these conditional distributions are the same

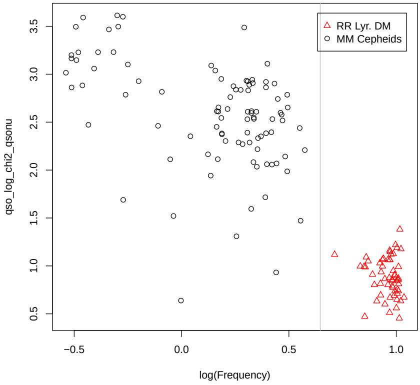
How do you use lightcurves that look like this?



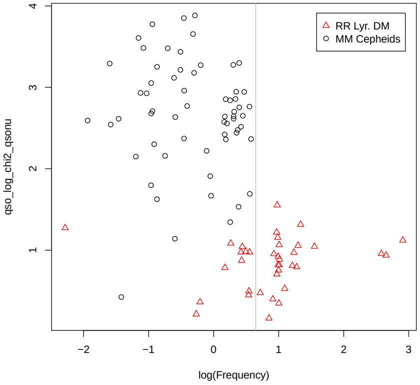
To classify lightcurves like this?



Light curves with $\sim$ 200 Flux Measurements



Light curves with 30 Flux Measurements



Outline

Problem Motivation

Frameworks

Data Experiments

Conclusions

Problem Setup and Notation

Training Data:
$\{(X_i, Z_i)\}_{i=1}^n$ $i.i.d.$ with $X_i \in \mathbb{R}^p$, Z_i is class of observation i

Test Data:
Observe Y where $Y = f(X, \delta)$. Same relationship between X and Z as in training set.

Goal:
Construct a classifier: $\mathcal{C}_z(y) = p(Z = z | Y = y)$

Strategy 1: Noisification

Main Idea:

1. truncate light curves in training set to match length of light curves in test set
2. derive features for truncated training light curves
3. train classifier on features derived from truncated training light curves
   - use random forests

Strategy 1: Noisification

Main Idea:

1. truncate light curves in training set to match length of light curves in test set
2. derive features for truncated training light curves
3. train classifier on features derived from truncated training light curves
   - use random forests

Repeat several times, selecting different subset of flux measurements from training light curves each time. Average resulting classifiers.

Strategy 2: Denoisification

Main Idea:

1. construct classifier on unmodified training data
2. denoise features of test observation
3. combine to get classifier for test data

Strategy 2: Denoisification

Main Idea:

1. construct classifier on unmodified training data
   - $\hat{p}(z|x)$, use random forests
2. denoise features of test observation
   - $\hat{p}(x|y) = \frac{\hat{p}(x,y)}{\hat{p}(y)} = \frac{\hat{p}(y|x)\hat{p}(x)}{\hat{p}(y)}$
3. combine to get classifier for test data
   - $\hat{p}(z|y) = \frac{\frac{1}{n}\sum_{i=1}^{n}\hat{p}(z|x_i)\hat{p}(y|x_i)}{p(y)}$

Strategy 2: Denoisification

$$
\begin{aligned}
p(z|y) &= \int p(z,x|y)dx \\
&= \int p(z|x,y)p(x|y)dx \\
&= \int p(z|x)p(x|y)dx \\
&= \frac{\int p(z|x)p(y|x)p(x)dx}{p(y)}
\end{aligned}
$$

which we estimate using,

$$
\hat{p}(z|y) = \frac{\frac{1}{n}\sum_{i=1}^{n} \hat{p}(z|x_i)\hat{p}(y|x_i)}{\hat{p}(y)}
$$

Denoisification

Estimating $p(y|x)$,

- $\{x_i\}_{i=1}^n$ features for well sampled training
- $\{y_i\}_{i=1}^n$ features for poorly sampled training
- $y_i^k = g_k(x_i) + \epsilon_{k,i}$

Using data $\{(x_i, y_i^k)\}_{i=1}^n$ and random forests regression, estimate g_k

Denoisification

$$\widehat{p}(y|x) = \prod_{k=1}^{p} \widehat{p}(y^k|x)$$
$$= \prod_{k=1}^{p} \phi\left(\frac{\widehat{g}_k(x) - y^k}{\hat{\sigma}_k}\right)$$

- assumptions:
  1. conditional independence of features from poorly sampled curve y given features from well sampled version of curve x
  $$p(y|x) = \prod_{k=1}^{p} p(y^k|x)$$
  2. $\epsilon_{k,i}$ are gaussian (ϕ is standard gaussian density)

Outline

Problem Motivation

Frameworks

Data Experiments

Conclusions

Data Sets

Simulated (500 train / 500 test)

- 200 measurements / curve
- RR Lyrae, Cepheid, β Persei, β Lyrae, Mira
- equal class sizes

OGLE (358 train / 165 test)*

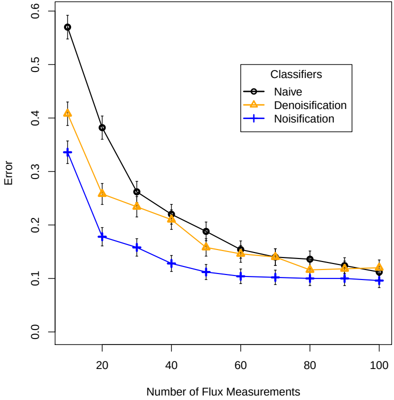
- $\sim$ 250 measurements / curve
- RR Lyrae DM, MM Cepheid, β Persei, β Lyrae, W Ursae Majoris
- smallest 50 largest 150

* from [3]

Test Set

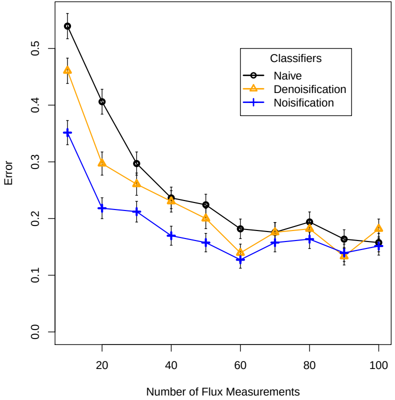
- truncate test curves at $10, 20, \ldots, 100$ measurements
- now have 10 test sets
- study how methods perform under varying noise levels

Simulated Error Rates



- noisification and denoisification improve performance
- improvement strongest at low number of flux / light curve test sets

OGLE Error Rates

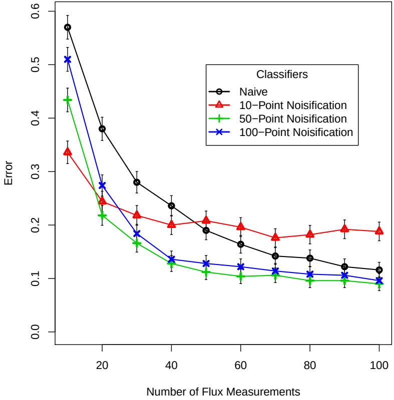


- similar story as with simulated data
- curves rougher because of cadence issues

Robustness of Noisified Classifiers

- difficult to noisify all data for every new observation
- continuity in how feature distribution change with number of points per curve

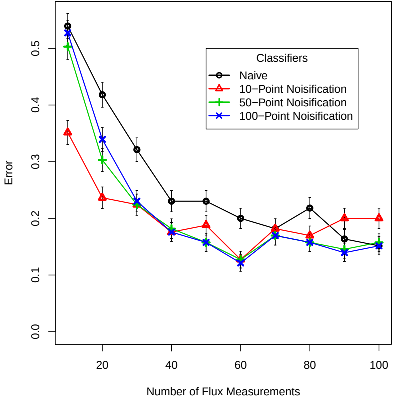
Robustness of Noisified Classifiers for Simulated Data



	error	CI
Naive	0.09	(0.06,0.11)
10-Point	0.17	(0.14,0.21)
50-Point	0.08	(0.05,0.1)
100-Point	0.08	(0.05,0.1)

Table: Error on Well Sampled Test Light Curves

Robustness of Noisified Classifiers for OGLE Data

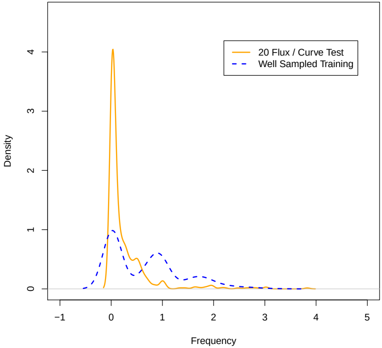


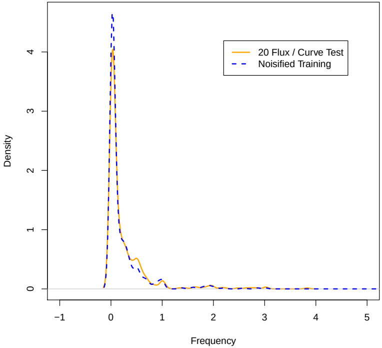
	error	CI
Naive	0.12	(0.07,0.17)
10-Point	0.18	(0.12,0.24)
50-Point	0.13	(0.08,0.18)
100-Point	0.12	(0.07,0.16)

Table: Error on Well Sampled Test Light Curves

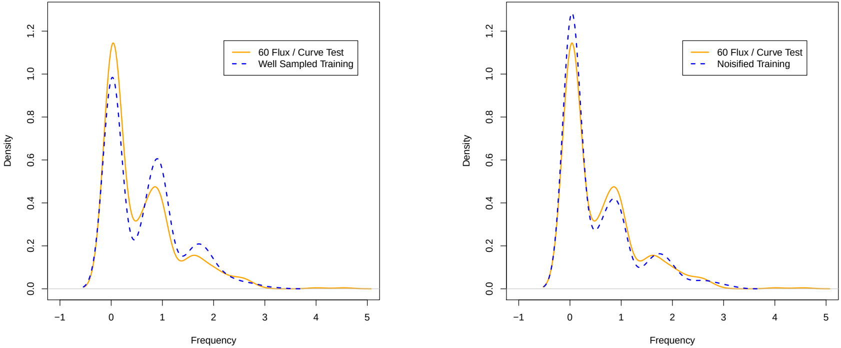
Noisifying Frequency for 20 Flux Curves

Noisification shifts distribution of training to distribution of test.





Noisifying Frequency for 60 Flux Curves



Outline

Problem Motivation

Frameworks

Data Experiments

Conclusions

Uses of Noisification and Denoisification

- here we used noisification / denoisification for differences in number of flux measurements per curve in training / test data
- could use nois. / denois. to address other systematic differences between training and test sets
    - flux noise
    - censoring
    - cadences

Conclusions

- not addressing noise results in suboptimal classifiers
- noisification and denoisification methods improve results
- noisification easier to implement in astronomy setting

Bibliography I

[1] J. Debosscher, L. Sarro, C. Aerts, J. Cuypers, B. Vandenbussche, R. Garrido, and E. Solano.
Automated supervised classification of variable stars.
Astronomy and Astrophysics, 475(3):1159–1183, 2007.

[2] L. Eyer and C. Blake.
Automated classification of variable stars for all-sky automated survey 1–2 data.
Monthly Notices of the Royal Astronomical Society, 358(1):30–38, 2005.

[3] J. Richards, D. Starr, N. Butler, J. Bloom, J. Brewer, A. Crellin-Quick, J. Higgins, R. Kennedy, and M. Rischard.
On machine-learned classification of variable stars with sparse and noisy time-series data.
The Astrophysical Journal, 733:10, 2011.