SSA Analysis and Significance Tests for Periodicity in S, RS, SU, AD, BU, KK, and PR Persei

Geoff B. Chaplin
Hokkaido, Kamikawa-gun, Biei-cho, Aza-omura Okubo-kyosei, 071-0216, Japan; geoff@geoffgallery.net

Received January 4, 2019; revised March 19, 2019; accepted March 25, 2019

Abstract Visual observations made by experienced observers are adjusted for individual observer bias. We examine the time series using signal processing methods to identify periodicities and test for the significance of the results finding a reliable periods in S Per, and to a limited extent in RS, SU, BU, and KK Per. Recommendations for future visual and electronic observation are made.

1. Introduction

We look at seven semiregular and irregular variables in Perseus. All but S Per are narrow range variables. The stars studied are shown in Table 1 together with previously quoted periodicities and references to the source.

SRc variables are massive, young, Population I stars with a magnitude range of under 5 and irregular periodicity typically in the 250–1000 day range (Percy 2011). The best known such stars are alpha Orionis (Betelgeuse) and mu Cephei ("the garnet star"). Lc variables are also red supergiant stars but with irregular periods. The variables apart from S Per studied here have a narrow range of variation (less than 2 magnitudes) and as such pose a severe test for visual observers because of this, the individual's eyes' color sensitivity, the Purkinje effect (Purkinje 1825; Sigismondi 2011; AAVSO (2013) and references therein), and other observational factors such as local light pollution. We anticipate the "extrinsic" noise (related to the observational process) as opposed to the "intrinsic" noise (related to random events within the star or environment—if any) to be large, and attempt to reduce this as much as possible prior to analysis. For example, even experienced observers—defined as those with over 100 observations of the star—may differ by as much as a magnitude when observing the same star at roughly the same time. In this paper we restrict our attention to observations made by experienced observers and analyze these for consistent "bias," adjusting the data before further analysis. Adjustment of observer data is described in detail below and is atypical of standard procedures which generally reject outliers only.

A variety of analytical techniques for period identification are used in the literature: discrete Fourier transform (DFT) (Kendall 1984; Shumway and Stoffer 2017) with or without adjustment for the observational window, for example, using the CLEAN (Roberts *et al.* 1987) or CLEANEST algorithms (Foster 1995); autoregressive analysis and in particular the simple and efficient implementation by Percy and Sato (2009); wavelet analysis—see Foster (1996) or Sundararajan (2015) for theory and, for example, Percy and Kastrukoff (2001) for an application to pulsating variables, and Sabin and Zijlstra (2006) when analyzing instability in long-period variables. A general review of these techniques is given by Templeton (2004). The difficulties of using standard Fourier methods and obtaining reliable results should not be underestimated (see Thomson 1990). More recently, non-linear techniques have also been used by Kollath (1990) and Kollath *et al.* (1998), and others in the context of giant variable stars. The methods of Empirical Mode Decomposition (Huang *et al.* 1998; overview by Lambert *et al.* 2019) are also geared particularly towards non-linear series.

We apply methods from the field of singular spectrum analysis (SSA), explained in the context of astronomical data analysis by Chaplin (2018) and references therein, and derive the underlying signal (removing noise and trends). The aim here is to identify underlying patterns of behavior, summarizing them by periodicities where appropriate, although the techniques of

Table 1. Stars analyzed in this study, with previously quoted periodicities and references to the source.

Star	*GSC Designation*	*Spectral Type (Wenger 2000)*	*Class (Kiss et al. 2006)*	*Period(s)*	*Magnitude Range (BAAVSS 2019)*
S Per	03698-03073	M4.5-7Iae C	SRc	813 ± 60 (Kiss *et al.* 2006) 822 (Samus *et al.* 2017) 745, 797, 952, 2857 (Chipps *et al.* 2004)	7.9–12.8
RS Per	03694-01293	M3.5IabFe-1 C	SRc	244.5 (BAAVSS 2019) 4200 ± 1500 (Kiss *et al.* 2006)	7.8–9.0
SU Per	03694-01652	M3-M4Iab C	SRc	533 (BAAVSSS 2019) 430 ± 70 and 3050 ± 1200 (Kiss *et al.* 2006) 500 (Stothers and Leung 1971)	7.2–8.7
AD Per	03694-01613	M3Iab C	SRc	No discernable peak, rise to lowest frequencies (Kiss *et al.* 2006) 362.5 (Samus *et al.* 2017)	7.7–8.4
BU Per	03694-01247	M4Ib C	SRc	381 ± 30 and 3600 ± 1000 (Kiss *et al.* 2006) 367 (Samus *et al.* 2017)	9.0–10.0
KK Per	03693-01951	M2Iab-Ib B	Lc	No significant frequency (Kiss *et al.* 2006)	7.5–8.0
PR Per	03694-00152	M1-Iab-Ib B	Lc	No significant frequency (Kiss *et al.* 2006)	7.7–8.2

SSA do not relate any such periodicities to harmonic patterns of behavior. We then proceed to test whether these periodicities are likely to have arisen by chance from noisy data (in which case we reject such periodicity as not intrinsic to the star) or not (in which case we accept it as an intrinsic periodicity).

2. Data, observer bias, and adjustments

Data are taken from the BAA (2019) and the AAVSO (2010) databases, and from the VSOLJ (2018) database prior to 2000. The list of experienced observers for which a bias adjustment is made is given in Appendix A.

For each star other than S Per we proceed as follows. The mean magnitude of visual observations is calculated for each experienced observer separately and for all experienced observers for that star. The individual mean less the overall mean is called the observer bias and is deducted from each observation made by that observer to get the adjusted magnitude. This process generally leads to a substantial reduction in the overall variance. Results are shown for each star in Table 2 (but we intentionally do not wish to state the bias for each observer since this might lead to a change in the observer's methods—consistency is preferred to accuracy). Table 2 also gives the timespan of data but in all cases there were a number of isolated or widely separated observations at the beginning of the time series which were ignored. It should be noted that the stars for which the bias adjustment was made are all narrow range variables, so preferential observing (for example, when the variable is bright) should not be a significant source of bias. On the other hand, preferential observing is a factor for S Per so a bias adjustment is not made.

One would not expect bias for a given observer to be constant across different stars because different reference stars may be used and the group of observers being compared against is different. Nevertheless it was noted that observers' magnitude estimates tended to be consistently high or consistently low although the amount differed from star to star.

For S Per, which has a much greater range of variation, we take the data from experienced observers without further adjustment.

3. Analytical methods

3.1. Singular Spectrum Analysis (SSA)

SSA is used to extract a series from observations and is a method used widely in meteorology, medical science, economics, the sciences and industry, and appears to be becoming the method of choice for time series data analysis. In this section we very briefly outline the methods, and introduce the terminology of, singular spectrum analysis (SSA), explained more fully in the paper by Chaplin 2018, books by Golyandina and Zhigljavsky 2013, Golyandina *et al.* 2001, and Huffaker *et al.* 2017.

From an autocorrelation matrix calculated from the time series of magnitude observations the eigenvectors and eigenvalues are calculated. These eigenvectors are sorted in order from the strongest to the weakest according to the relative magnitudes of the associated eigenvalues. The related time series are then compared with each other to find correlations between them and to determine if the general patterns of behavior are similar. The original time series is "projected" along each of these eigenvectors to derive an EV-time series (which we subsequently refer to as the EV). We then group the series together into "trends" (long-term slow patterns), "cyclical" (possibly several different groups of series with different periods), and noise.

It should be noted that observations are required at equally spaced intervals in order to perform the above analysis—so we have to put data into equal time intervals (buckets), averaging values within the bucket. In the stars covered here data have been put into 20-day buckets. Also, reconstructed signals, although they may look periodic, do not necessarily have a constant period nor do they have a constant amplitude, and are not derived in any way from harmonic series—the EV time series are merely complicated averages of the original data. "Periods" indicated below represent an approximation to the actual behavior.

In this paper we use the R (2018a) statistical programming language and CRAN (2018b) libraries and in particular the function "ssa" in the R library "Rssa," and use the code as explained in detail in section 3.7 and Appendix B.

3.2. Significance tests of discovered signals

A white noise (uncorrelated random noise) is generally regarded as an insufficient test for the presence of signals in data, and Monte Carlo methods (MCSSA) have been devised to test significance (for example, Allan and Smith 1996; Ghil *et al.* 2002). We use the R implementation of MCSSA developed by Gudmundsson (2017) and in particular the functions decompSSA and MCSSA. Code is given in Appendix B5.

We also use a somewhat different approach inspired by analysis of variance methods and also by the following intuitive idea. If we see a signal in a period of data, then if the signal is a permanent feature of the underlying process we expect it to continue, but if it is an artefact arising from noisy data we expect it to cease to be present in the future.

In order to perform significance analysis we compare two different time intervals of the same series (first and second halves, H1 and H2), looking for common signals, and we need to do this (for reasons which will become apparent below) in an automated way. Any series typically contains trends—long term changes—which may be quite different in two sub-intervals of the series, together with potential signals and noise. The impact of this can be that a periodic signal manifests itself as one set of eigenvectors in one subset and a different set in another.

Table 2. Timespan and variance reduction through observer bias removal.

Star	*Start (2440000+)*	*End (2440000+)*	*Length (years)*	*Variance Reduction*
S	–21576	18115	108.7	n/a
RS	2744	18096	42.0	43.5%
SU	–5347	18115	64.2	49.0%
AD	1978	18115	44.2	65.1%
BU	1636	18082	45.0	40.8%
KK	3112	18115	41.0	36.3%
PR	3112	18115	41.1	54.4%

We therefore begin by finding trend components (defined as having too long a period or no period), taking the first signal which is not a trend and whose period is not too short as defining a potential signal, and look for remaining signals whose period matches the first to within a defined amount (the "acceptance criterion"). A potential signal is required to have two or more component signals. Code "XYZgetSignals_udf" performs this analysis.

First, an analysis of H1 in the observational data is made to determine the EV groupings that correspond to a signal. We then do the same for H2. If the period in H1 is *P1* and the period in H2 is ˆ and if

$$A > abs((P1 - P2) \times 400 / (P1 + P2)) \qquad (1)$$

where A is the acceptance criterion, we accept the two periods as belonging to a signal. If we find no correspondence between signals in H1 and H2 then we conclude there is no consistent period in the data. Secondly, (assuming we have found a potential signal) we then model the original entire series (i.e. before trend removal) as a "red" noise (AR(1)) process (see section 3.4 below). Code "XYZ actual data tests.R" performs this analysis.

The AR(1) model is then used to generate simulated data ("surrogates") over the same time period as the actual data which are then analyzed as above as if they were the real data. In cases where a signal is found in both H1 and H2 of the simulated series with a difference less than the acceptance criterion, it is then counted as a (simulated) signal. Note that the simulated signal is not required to be of the same frequency as that identified in the actual data. The process is then repeated over 1,000 simulations and the proportion generating simulated real signals for the wide and the narrow acceptance criterion is calculated (together with an estimate of the accuracy of this figure). This then gives an indication of the confidence that the real signal did not arise by chance. Code "XYZ significance tests.R" performs this analysis.

Finally it is important to test variation in the parameters used to perform the analysis, in particular by changing the bucketing length, the start date by one, two, or more days (which changes the bucket contents), and the SSA window length. We test using bucket sizes such as 17, 20, 23, 30, 34, 40, and 46 days (depending on the length of data available and the suspected period—aiming to keep within about one tenth of the period) and require that the signal is found in all the decompositions. We then reduce the acceptance period subject to the signal continuing to be discovered.

The process is described more fully in the case of SU Per, which is presented first in section 4 below.

3.3. Fourier analysis

Fourier analysis is a traditional method for analyzing time-series where there is underlying periodicity and where the underlying series is stationary. For general references on traditional time series analysis including Fourier and autoregressive techniques, see Kendall (1984) and Shumway and Stoffer (2017; the latter includes R examples and code).

In this paper we use the "spectrum" function in the R stats library to perform the Fourier analysis and smoothing.

Error bars on the spectral power can be calculated from surrogate data. However, a plot of the spectrum together with the percentiles of the surrogate distribution can be misleading and can overstate the significance of peaks—underlying AR(1) noise can exaggerate the height of peaks in the spectrum (Allen and Smith 1996). Code in Appendix B4 plots the spectrum and surrogate percentiles.

3.4. Autoregressive AR(1) model

Random noise is generated from a zero mean "red" noise (AR(1)) process according to the following formula:

$$x_t = alpha \times x_{t-1} + sigma \times epsilon_t \qquad (2)$$

where *alpha* and *sigma* are constants and *epsilon* is generated from an independent random normal (zero mean, unit variance) process.

The parameters of the zero mean AR model are chosen by fitting such a model to the actual data series using the "ar" function in the R stats library.

3.5. Wavelet analysis

Where periodicity is known not to be strict or the time series non-stationary, Fourier methods are theoretically incorrect—although they may be a reasonable approximation. Instead a technique known as wavelet analysis (or more simply a moving window on the data as in Howarth and Greaves 2001) is often used. Here we use code based on the wavelet analysis code from the AAVSO (2017). For comparison with the SSA results we analyze the data using two window sizes determined by the "decay" factor—a factor of 0.0001 cycles per day (the "slow" window, roughly corresponding to a slow 10,000-day window) identifying periodicities which change slowly, and a factor of 0.003 (a "fast" 333-day window) identifying more rapid changes. In such analysis we identify the strongest period, then the next strongest, etc. It is the case that generally the second strongest period is virtually the same as the first, so when looking for a different period we require that the period is at least 20% different from its predecessor. In each case only periods significant at a certain level on an F-test (dependent on the star) are shown.

3.6. Missing data

Three methods for filling missing data were used. The first was simple linear interpolation between the last known data value and the immediately following known data value. The second followed the method of Kondrashov and Ghil (2006) by filling missing values from the first eigenseries, recentering and refitting until convergence of the eigenvalue was achieved, then potentially going on to the next eigenseries. A final method was to randomize the linearly interpolated values, the impact of which is to slightly lower the value of the autoregressive parameter in the fitted AR model.

3.7. Code

R code intended for the Rstudio environment for the analysis described in sections 3.2 and 3.3 is provided in Appendix B. Two main codes are used—one to analyze the real data ("first part" above) and another, if needed, to simulate and analyze the simulated data ("second part" above). Each part uses (directly or indirectly) some of the following helper functions (ending in _udf—user defined function) given in Appendix B1.

XYZspectrum_udf performs a spectral analysis using ar smoothing or no smoothing, producing a chart if required and returning a list of the periods discovered in declining order of strength.

XYZgetSignals_udf code performs 1d-ssa on the data, finding trends and signals meeting certain criteria.

XYZbucketData_udf takes the observational data and times of observation and collects values into the specified length of bucket, taking the average of all values in the bucket. Where gaps in the data occur linearly interpolated values are calculated, returning the bucketed data, a flag indicating whether in interpolated value is used, and other summary data.

Appendix B2 contains the **"first part"** code and loads the above functions and the data. The user sets various parameters and the code buckets data and performs a 1d-ssa analysis of the entire series and the first and second half separately, producing results for inspection. Additionally, the code fits an AR(1) model producing parameters for simulation use.

Appendix B3 contains the **"second part"** code and includes a function **matchTest2** to decide whether two signals are close (the user inputs the diffPeriodpercent figure, and other parameters, into the code where indicated), loads other helper functions, and simulates 1000 data series using user input AR(1) parameters, performing the analysis described in section 3.2 and outputting the proportion of simulations producing signals of the same period in each half of the data.

Appendix B4 contains code to plot the Fourier spectrum of the signal derived from the entire data series, together with upper and lower 2 and 10 percentiles calculated from the surrogate data and signals.

Appendix B5 contains the code to perform MCSSA on the actual data, producing a chart with error bars and identifying outlying frequencies.

4. The stars

4.1. SU Per

SU Per is covered in more detail than the following stars hence is presented first.

Prior to September 1974 data were sparse—even after bucketing into 20-day buckets more than half the buckets were empty and with long gaps prior to 1974. Attempts to fill the data using linear interpolation or the Khondrashov and Ghil method failed to give a satisfactory data series in this earlier period. Post-1974, 6,550 observations were bucketed into 803 20-day buckets. Less than 8% of the buckets were empty, with no long empty runs, and tests using linear interpolation versus the Khondrashev and Ghil method showed no material difference in the resulting signals; the following results are based on the linear interpolation gap filling method. Bucketing tests were run using 17, 20, 23, 30, 34, 40, and 46 days together with shifts in the start date by 1 or 2 days, and showed a consistent set of results with an acceptance criterion of 7% across all the following analysis. We describe the results in detail for the 20-day buckets.

An AR(1) model was fitted to the data (after removing the mean) and—after randomizing the linearly interpolated values, which reduces the alpha—showed an alpha of 0.71 and sigma of 0.145. If we assume bias adjusted observations have a standard deviation of 0.2 magnitude then the bucketed data (approximately 8 observations per bucket) should have a residual standard deviation of about 0.1. The AR model is therefore not inconsistent with observational error being by far the largest part of the noise in the data.

The following discussion and figures are based on a window length of 400 for the entire series. Tests with a window length of 200 show similar results but going much shorter than that starts to produce inconsistent results. Figure 1a shows the EVs and Figure 1b the correlation analysis for the entire series, with signals 5 and 6 meeting the criteria and showing a period of 475 days with the spectrum illustrated in Figure 1c. The data, trend, and signal are shown in Figure 1f.

The EVs for the first half are shown in Figure 1d, with signals 3 and 4 meeting the criteria and having a period of 464 days. Signals 5 and 6 are approximately the second harmonic.

EVs for the second half are shown in Figure 1e, with signals 6 and 7 meeting the criteria and giving a period of 475 days. EVs 9 and 10 are also approximately the second harmonic.

The AR model was then used to produce 1000 simulated sets of observational magnitudes, each of which was analyzed as described in section 3.2. If the period of the identified signal in the first half was within 7% of the period from the second half then this was counted as a "hit." It should be noted that there was no requirement that the spurious signal periods matched the signal period in the actual data—simply that there are closely similar signals in both intervals. Signals corresponding to periods of 1,000 days (50 buckets) or longer, or 100 days (5 buckets) or shorter were ignored in this test. (Very few spurious signals had periods outside this range and many signals had no identified period.)

Simulation results showed 3.7% (with a standard deviation (sd) of 2.0%) of simulations led to spurious signals of approximately the same frequency in both halves of the data.

As an independent test we use the Monte Carla SSA methods in the MCSSA algorithm from the "simsalabim" library to produce Figure 1g. Note that periods are 40 days/frequency. The figure identifies (as well as early trends) the signals at 475 days lying just below the 95% confidence level together with significant signals around the second harmonic.

We conclude that SU Per exhibits a periodicity of 475 ± 33 days with approximately 95% confidence.

4.1.1. Fourier analysis

We use the simulations generated above, together with the reconstructed signals and their spectra, to generate 10- and 2-percentile power levels. These are plotted in Figure 1h along with the (unsmoothed) spectrum of the signal in the actual data. Note the following points. The autoregressive process generates the typical "1/f" rise in power at lower frequencies widely seen in Fourier spectra of magnitude time series. Also, the figure misleadingly suggests the signal is significant at the 98% level—the noise process exaggerates the power in the actual signal, thereby overestimating its significance.

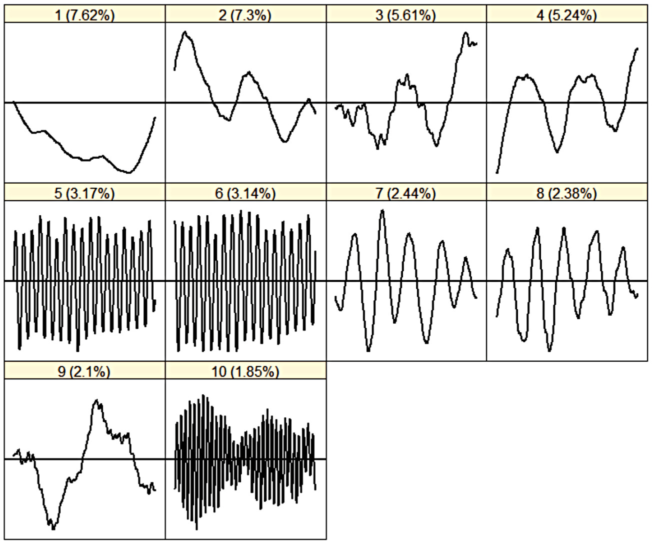

Figure 1a. SU Per entire series EVs (amplitude as a function of time).

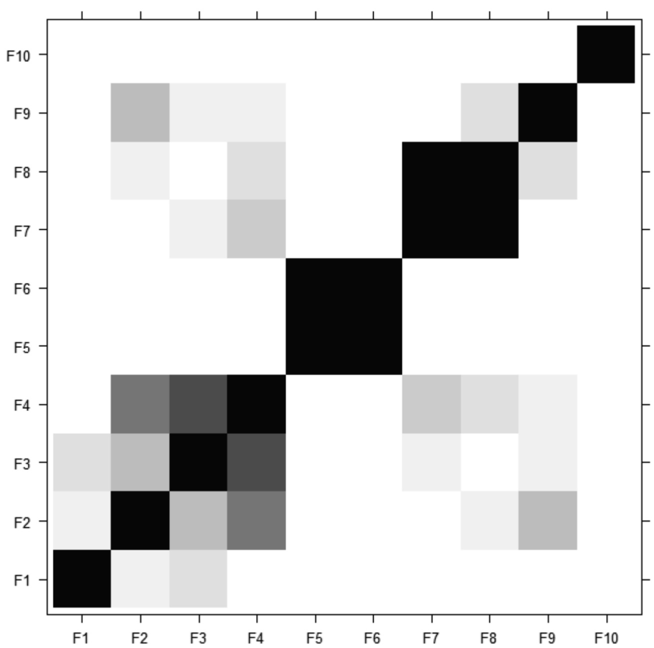

Figure 1b. SU Per entire series, correlation matrix.

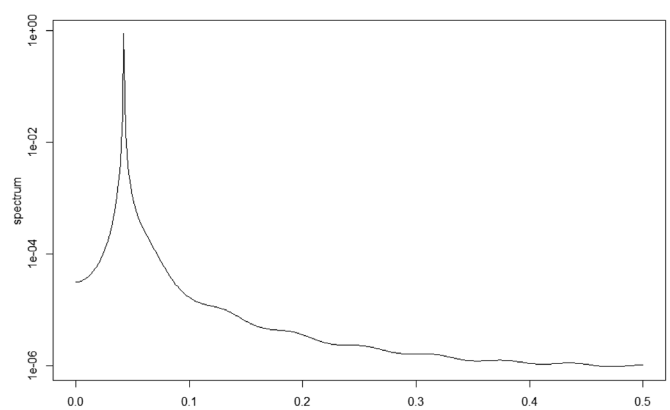

Figure 1c. SU Per spectrum derived from signals 5 and 6 in the entire series.

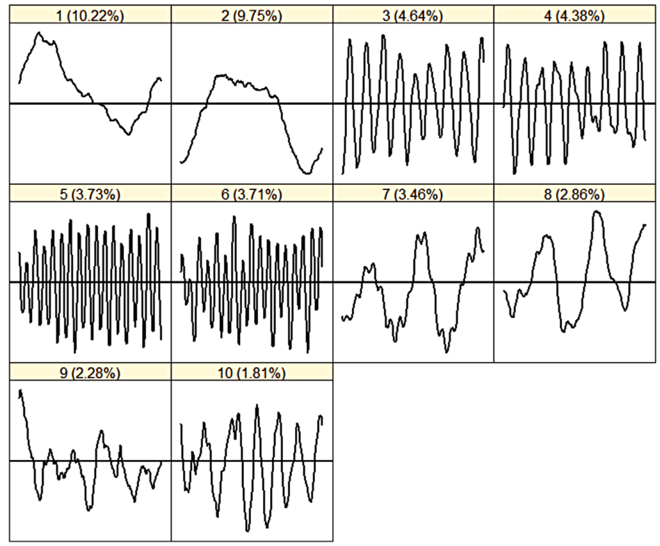

Figure1d. SU Per first half EVs (amplitude as a function of time).

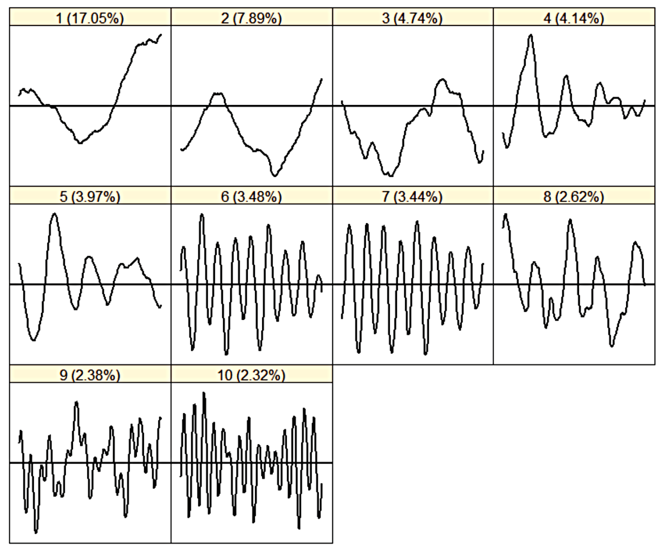

Figure 1e. SU Per second half EVs (amplitude as a function of time).

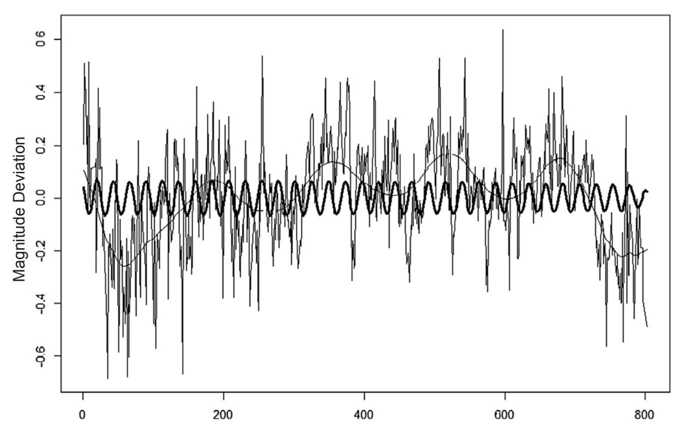

Figure 1f. SU Per entire data series with recovered trend components (EVs 1 to 4) and signal.

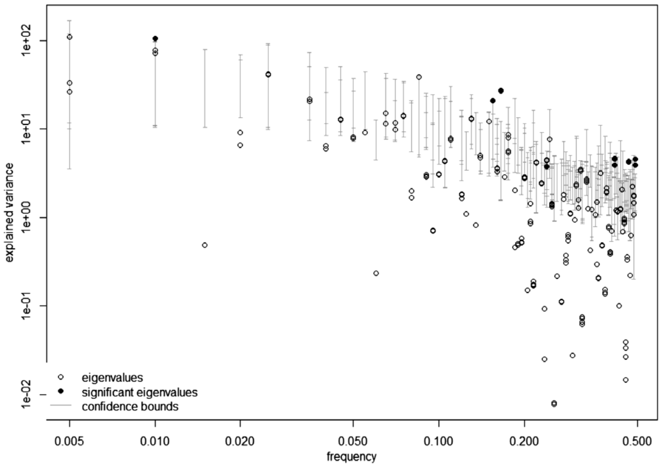

Figure 1g. SU Per significance test of EV signals with 95% error bars.

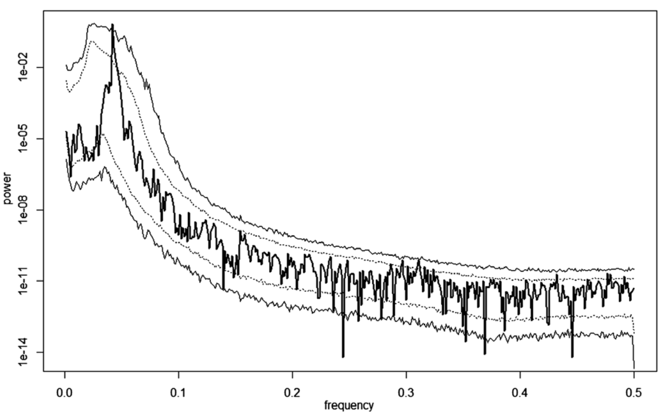

Figure 1h. Fourier spectrum of data signal together with simulation based 10 and 2 percentile envelopes.

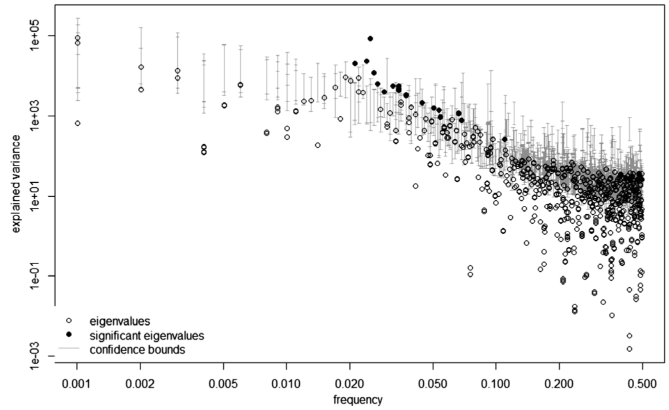

Figure 2. S Per significance test of EV signals with 99% error bars.

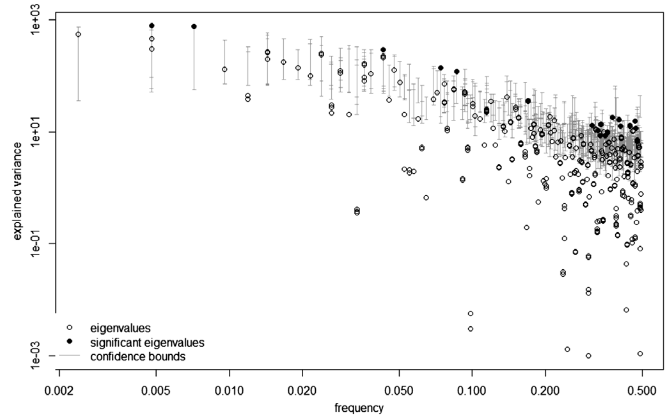

Figure 3. RS Per significance test of EV signals with 95% error bars.

4.1.2. Wavelet analysis

The slow wavelet identifies a 3,225-day period and many periods longer than 25% of the time series with 99% confidence—we reject longer periods as trends in our SSA analysis—together with a period at 476 days significant at the 97.5% level and briefly a period of approximately four times this. The fast wavelet also identifies the very long waves and identifies a period rising from about 1,600 days to 1,900 days.

We note however that simulated data regularly also show signals persisting over large fractions of the data span but, while these are significant in the context of that specific series, in the context of a series which may be generated by a random process, wavelet analysis carries little meaning and is therefore not covered further for the following stars.

4.2. S Per

S Per is analyzed in some detail in Chaplin (2018). We simply summarize the data and state the simulation results here.

The data are well populated from January 1920. From 25,860 observations 1,789 20-day buckets were constructed with less than 2% being empty. A fitted AR(1) model gave an alpha of 0.96 and sigma of 0.20, the higher sigma possibly arising because of unadjusted bias in the observations and the high alpha because of the large amplitude of variation relative to the noise.

In simulations the high alpha tends to generate very few signals with a period as short as that analyzed for S Per, and simulations resulted in only 0.1% generating signals within 5% of each other, hence we accept that S Per has a period of 815 ± 40 days with over 99% confidence.

Using the MCSSA significance testing methods produces the results shown in Figure 2. The 815-day signal lies well outside the error bars, with neighboring and many harmonics also outside the error bars consistent with amplitude and frequency modulation of the signal.

4.3. RS Per

Prior to November 1972 data were sparse. Post-1972 4,820 observations were bucketed into 838 20-day buckets. Less than 9% of the buckets were empty, with no long empty runs. A fitted AR(1) model gave an alpha of 0.78 and sigma of 0.165.

SSA consistently revealed periods in the 445–495 day range with a 5% acceptance criterion and simulations resulted in 4.6% generating signals.

Using the MCSSA significance testing methods produces the results shown in Figure 3. The signal lies outside the error bars, with neighboring and some harmonics also outside the error bars.

We conclude that S Per has a period of 475 ± 25 days with 95% confidence.

4.4. AD Per

Prior to September 1974 data were sparse. Post-1974 3,945 observations were bucketed into 805 20-day buckets.

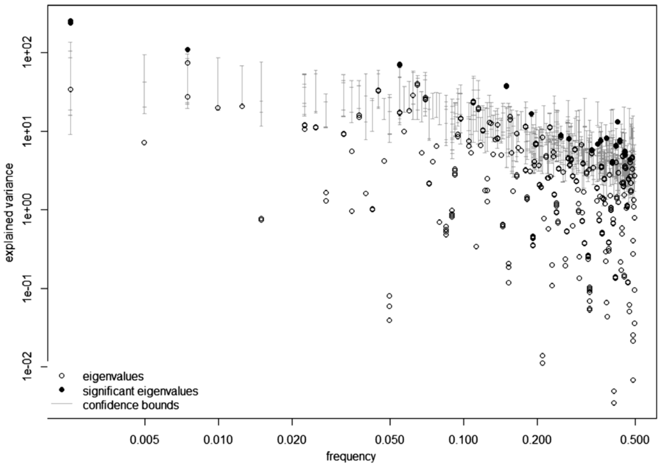

Figure 4.AD Per significance test of EV signals with 90% error bars.

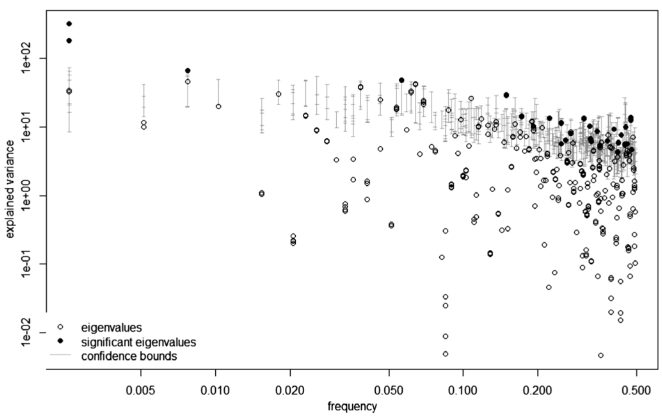

Figure 5. BU Per significance test of 20 day bucketing and signal from EVs 4 and 5 with 80% error bars.

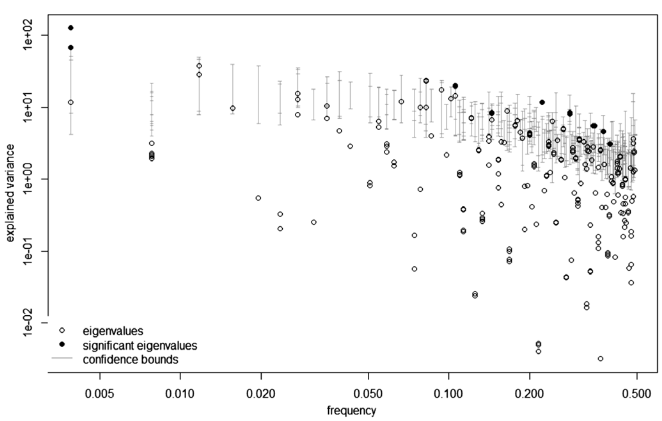

Figure 6. KK Per significance test of EV 5, 6, 9, 10 and 30 day bucketing with 90% error bars.

Approximately 8% were empty, with no long empty runs. A fitted AR(1) model gave an alpha of 0.61 and sigma of 0.13.

Testing with 17- to 35-day bucketing with a 15% acceptance criterion nevertheless revealed periods between 320 and 450 days, with many spectral peaks being very broadly defined. The entire series gave a signal period of 360 days. Simulation with a 15% acceptance criterion gave 13% generating signals. On the other hand, MCSSA using 20-day bucketing and signals 5–8 detected significance at the 90% level at periods of 350 days together with a second harmonic and an intermediate period depicted in Figure 4.

Because of the instability of the signal detected with changing bucketing we conclude AD Per has no clear intrinsic period.

4.5. BU Per

Prior to December 1975 data were sparse. Post-1975 3,562 observations were bucketed into 782 20-day buckets. Approximately 7% of the buckets were empty, with no long empty runs. A fitted AR(1) model gave an alpha of 0.56 and sigma of 0.13.

SSA gave periods in the range 300–360 days although there were exceptions with a 17-day bucketing and in one case the second harmonic gained preference. Simulations resulted in 14% of signals lying within a 15% acceptance criterion. MCSSA significance testing produces the results shown in Figure 5. The signal lies just outside the error bars with neighboring and some harmonics also outside the error bars.

We conclude BU Per has a period of 330 ± 50 days with 80% confidence.

4.6. KK Per

Prior to July 1976 data were sparse. Post-1976 3,391 observations were bucketed into 771 20-day buckets. Less than 7% of the buckets were empty, with no long empty runs. A fitted AR(1) model gave an alpha of 0.57 and sigma of 0.13 (virtually the same as BU Per).

SSA with 17- to 35-day buckets consistently gave well-defined periods in the range 330–360 days using a 7% acceptance criterion, with the entire series showing 348 days, and simulations resulted in 3% false signals. However, selection of EVs to form the signal was sensitive to whether or not linearly interpolated values were randomized. MCSSA significance testing produces the results shown in Figure 6. The signal lies just on the error bar with neighboring and some harmonics outside the error bars.

We tentatively conclude KK Per has a period of 345 ± 25 days with approximately 90% confidence.

4.7. PR Per

Prior to August 1982 data were sparse. Post-1982 2,826 observations were bucketed into 659 20-day buckets. Less than 9% of the buckets were empty, with no long empty runs. A fitted AR(1) model gave an alpha of 0.59 and sigma of 0.11.

SSA with 17- to 35-day gave no identified period in many cases and when signals were identified they tended to be 460 and 300 days.

We conclude PR Per has no clear period.

5. Conclusions and observer recommendations

SSA provides a means of exploring the signals within the data and separating trends and noise from cyclical patterns, but needs separate analysis to gain confidence that these signals are meaningful and not randomly generated by noise in the observations. It is clear from the above analysis that narrow range late spectral type stars are problematic for visual

Table 3. Results of the analysis.

Star	Period (days)	Uncertainty	Confidence
S Per	815	40	>99%
RS Per	475	25	95%
SU Per	475	33	95%
AD Per	No clear period		
BU Per	330	50	80%
KK Per	345	25	90%
PR Per	No clear period		

observation. Nevertheless a long run of data can help overcome the noise, but a better and available solution is to reduce the noise. We strongly recommend the use of CCD/DSLR equipment by amateurs as outlined further below to overcome the problem of the substantial component of extrinsic noise in future data.

We reject the use of wavelet analysis in the context of noisy data such as these.

Results of the analysis are summarized in Table 3.

S Per makes it clear that a long history and large range of magnitude variation lead to a period determination with high confidence. For the other stars, where a period is determined, the confidence is in the 80-95% region.

It is unfortunate that observations of SRc variables have reduced in recent years. These stars are not well understood and a rich long database of observations is essential for future study. Visual observation is helpful in order to relate visual and future electronic observations and in any event is likely to be more plentiful than electronic observations. Visual observers are encouraged to build up a series of over 100 observations, making observations no more frequently than once a week.

The narrow range of variability and the strong color make these objects ideal for CCD observation with a V filter, or DSLR observation. A good consumer digital camera and 200mm lens on an equatorial mount is sufficient to produce high quality data for these objects. Variable sky conditions can mean any single observation may be accurate to only 0.1 magnitude (even though the software stated reduction accuracy is much better), so electronic observations should ideally be a set of 30 to 100 observations to reduce the error in the mean to 0.01 magnitude or less. It should be noted that with short focal length instruments (500mm or less) six or more SRc variables in Perseus will fit on a 35mm frame sensor, making data collection efficient. A long history of accurate magnitudes derived from electronic data is essential to apply some of the analysis in this paper with a high level of confidence and is essential for a better understanding of pulsating variables.

6. Acknowledgements

The author is grateful to the AAVSO, the BAA, and the VSOLJ for providing the observational data used in this analysis and, in particular, to the dedicated observers listed in Appendix A who followed one or more of these stars regularly for a period of years or decades. The author would also like to thank J. Howarth and R. Pickard for helpful suggestions during an early version of this paper. The author is also grateful to the AAVSO for the code for the wavelet analysis performed in this paper. This paper benefited greatly from the comments of anonymous reviewers for which the author is very grateful.

References

AAVSO. 2010, Data access (https://www.aavso.org/data-access).

AAVSO. 2013, Purkinje and v faint estimates (https://www.aavso.org/purkinje-and-v-faint-estimates).

AAVSO. 2017, Software directory WWZ Fortran (https://www.aavso.org/software-directory).

Allen, M. R., and Smith, L. A. 1996, *J. Climate*, **9**, 3373.

British Astronomical Association. 2019, BAA Photometry Database (https://www.britastro.org/vssdb). **[baa2]**

British Astronomical Association Variable Star Section. 2019, BAA/VSS website (http://www.britastro.org/vss). **[baa1]**

Chaplin, G. B. 2018, *J. Amer. Assoc. Var. Star Obs.*, **46**, 157.

Chipps, K. A., Stencel, R. E., and Mattei, J. A. 2004, *J. Amer. Assoc. Var. Star Obs.*, **32**, 1.

Foster, G. 1995, *Astron. J.*, **109**, 1889.

Foster, G. 1996, *Astron. J.*, **112**, 1709.

Ghil, M., *et al.* 2002, *Rev. Geophys.*, **40**, 1003.

Golyandina, N., Nekrutkin, V., and Zhigljavsky, A. 2001, *Analysis of Time Series Structure*, Chapman and Hall, CRC Press, Boca Raton, FL.

Golyandina, N., and Zhigljavsky, A. 2013, *Singular Spectrum Analysis for Time Series*, Springer-Verlag, Berlin.

Gudmundsson , L. 2017, Singular System/Spectrum Analysis (SSA) (https://r-forge.r-project.org/projects/simsalabim).

Howarth, J., and Greaves, J. 2001, *Mon. Not. Roy. Astron. Soc.*, **325**, 1383.

Huang, N. E., *et al.* 1998, *Roy. Soc. London Proc, Ser. A*, **454**, 903.

Huffaker, R., Bittelli, M., and Rosa, R. 2017, *Non-linear Time Series Analysis with R*, Oxford Univ. Press, Oxford.

Kendall, M. 1984, *Time Series*, 2nd ed., Charles Griffin and Co., Ltd., London and New York, ch. 8.

Kiss, L. L., Szabo, Gy. M., and Bedding, T. R. 2006 *Mon. Not. Roy. Astron. Soc.*, **372**, 1721.

Kollath, Z. 1990, *Mon. Not. Roy. Astron. Soc.*, **247**, 377.

Kollath, Z., Buchler, J. R., Serre, T., and Mattei, J. A. 1998, *Astron. Astrophys.*, **329**, 147.

Kondrashov, D., and Ghil, M. 2006, *Nonlinear Process Geophys.*, **13**, 151.

Lambert, M., Engroff, A., Dyer, M. and Byer, B. 2019, Empirical Mode Decomposition (https://www.clear.rice.edu/elec301/Projects02/empiricalMode).

Percy, J. R. 2011, *Understanding Variable Stars*, Cambridge Univ. Press, Cambridge, 217.

Percy, J. R., and Kastrukoff, R. 2001, *J. Amer. Assoc. Var. Star Obs.*, **30**, 16.

Percy, J. H., and Sato, H. 2009, *J. Roy. Astron. Soc. Canada*, **103**, 11.

Purkinje, J. E. 1825, *Neue Beiträge zur Kenntniss des Sehens in Subjectiver Hinsicht*, Reimer, Berlin, 109.

The R Foundation for Statistical Computing. 2018a, R: A language and environment for statistical computing (https://www.R-project.org).

The R Foundation for Statistical Computing. 2018b, CRAN: The Comprehensive R Archive Network (https://cran.r-project.org).
Roberts, D. H., Lehar, J., and Dreher, J. W. 1987, *Astron. J.*, **93**, 968.
RStudio. 2018, RStudio software (https://www.rstudio.com).
Sabin, L., and Zijlstra, A. A. 2006, *Mem. Soc. Astron. Ital.*, **77**, 933.
Samus N. N., Kazarovets E. V., Durlevich O. V., Kireeva N. N., and Pastukhova E. N. 2017, *General Catalogue of Variable Stars*, version GCVS 5.1 (http://www.sai.msu.su/gcvs/gcvs/index.htm).
Shumway, R. H., and Stoffer, D. S. 2017, *Time Series Analysis and its Applications*, 4th ed., Springer, New York, ch. 4.
Sigismondi, C. 2011, arXiv:1106.6356v1.
Stothers R., and Leung K. C. 1971, *Astron. Astrophys.*, **10**, 290.
Sundararajan, D. 2015, *Discrete Wavelet Transform*, Wiley, Hoboken, NJ.
Templeton, M. 2004, *J. Amer. Assoc. Var. Star Obs.*, **32**, 4.
Thompson, D. J. 1990, *Philos. Trans. Roy. Soc. London, Ser. A*, **330**, 601.
Variable Star Observers League in Japan. 2018, VSOLJ variable star observation database (http://vsolj.cetus-net.org/database.html).
Wegner, M., *et al.* 2000, *Astron. Astrophys., Suppl. Ser.*, **143**, 9.

Appendix A

List of the observers for which a bias adjustment is made, and the number of visual observation for the stars analyzed.

D. Stott	1153
G. Poyner	238
G. Ramsay	104
C. Hadhazi	1498
S. Hoeydalsvik	224
I. A. Middleton	1391
J. D. Shanklin	100
T. Kato	750
J. Krticka	221
A. Kosa-Kiss	1139
R. S. Kolman	352
L. K. Brundle	2239
W. Lowder	112
M. J. Nicholson	313
O. J. Knox	337
E. Oravec	588
P. J. Wheeler	467
S. Papp	202
R. C. Dryden	629
S. W. Albrighton	4211
S. Sharpe	1048
A. Sajtz	746
T. Markham	1781
P. Vedrenne	2926
W. J. Worraker	203
Y. Watanabe	503

Appendix B: R code

Notes:

1. We recommend the use of RStudio (2018) which provides a simple and highly efficient way of handling R code and results including the production of graphics.

2. The user needs to set the path according to where the R system has been installed—see the code comments below—and also define certain input parameters.

3. Comments are in italics, code in bold, headings in larger type italics.

B.1. Helper functions

```
# function to get periods corresponding to peak intensities
XYZspectrum_udf <-
function(x, drawPlot, graphText, smoothing) {
   if (drawPlot) spec.out←spectrum(x, main=graphText,
      method=smoothing)
   else spec.out←spectrum(x, plot=FALSE, method=smoothing)
   #Power Spectrum Plots
   power<-spec.out$spec # vertical axis values in spectral plot
   frequency<-spec.out$freq # all the frequencies on the x-axis
   cycle<-1/frequency # corresponding wavelengths
   #Sort cycles in order of magnitude of power spikes
   hold<-matrix(0,(length(power)-2),1)
   for(i in 1:(length(power)-2)){
      max1<-if(power[i+1]>power[i]&&power[i+1]>power[i+2])1 else (0)
        hold[i,]<-max1
   }
   max<-which(hold==1)+1
   if (length(max) == 0) {
      max = 1
   } else {
      if (power[1] == Inf) {
         max = 1
   } else {
      if(power[1]>power[max]) max = 1
      }
   }
   power.max<-power[max]
   cycle.max<-cycle[max]
   o←order(power.max, decreasing=TRUE)
   cycle.max.o<-cycle.max[o]
   peakFrequencies<-1/cycle.max[o]
   results<-list(cycle.max.o)
   return(results)
}

# function to identify trends and primary periodic signal
XYZgetSignals_udf <-
function(y, s, longestPeriod, shortestPeriod, bucketSize,
periodDiffpercent,outputVecCount){
   # find trends
   trendSignals = seq(0, 0, length.out=outputVecCount)
   EVPeaks = seq(0, 0, length.out=outputVecCount)
   for (i in 1:outputVecCount){
      r <- reconstruct(s, groups = list(EV = c(i:i)))
      recon = unlist(r[1])
      spec.out = XYZspectrum_udf(recon, drawPlot=FALSE, "",
         smoothing="ar")
      specPeaks = unlist(spec.out[1])*bucketSize
      if (length(specPeaks) == 0) specPeaks = 0
      EVPeaks[i] = specPeaks[1]
      if (EVPeaks[i]>longestPeriod) trendSignals[i] = i
   }
   trendSignals = trendSignals[trendSignals != 0]
   #determine first periodic signal neither too long nor too short a period
   periodSignals = seq(0, 0, length.out=outputVecCount)
   pStart = 0
   for (i in 1:(outputVecCount-1)) {
```

```
        if (pStart == 0) {
            if (EVPeaks[i] >= shortestPeriod & EVPeaks[i] <= longestPeriod) {
                itmp = 1
                periodSignals[itmp] = i
                pStart = i
            }
        }
    }
#determine subsequent periodic signals matching first
    if (pStart>0){
        for (i in (pStart+1):outputVecCount) {
            if (EVPeaks[i] >= shortestPeriod & EVPeaks[i] <= longestPeriod &
                abs(EVPeaks[i]-EVPeaks[pStart]) < periodDiffpercent*EVPe
                    aks[pStart]/100) {
                itmp = itmp + 1
                periodSignals[itmp] = i
            }
        }
    }
    periodSignals = periodSignals[periodSignals != 0]
    if (length(periodSignals) > 1) { # NB single signal not allowed
        r2 <- reconstruct(s, groups = list(EV = periodSignals))
        signal = unlist(r2[1])
        spec.out2 = XYZspectrum_udf(signal, drawPlot=FALSE, "",
            smoothing="ar")
        signalPeaks = unlist(spec.out2[1])*bucketSize
    } else signalPeaks = NULL
    return(list(trendSignals, EVPeaks, periodSignals, signalPeaks))
}

# function to collect irregularly timed data into constant size buckets
XYZbucketData_udf <-
function(bucketSize, data, time, ndata){
    n = 1
    bucketSum = data[1] # sum within a bucket
    count = 1 # number of obs within bucket
    sumCount = 0 # calculates the average number of data points in non-
      empty buckets
    nbucket = 1 # number of buckets
    Tstart = time[1]
    maxBuckets = floor((time[ndata] - Tstart) / bucketSize) + 1
    bucketData = seq(0, 0, length.out=maxBuckets)
    EMPTYBUCKETFLAG = seq(0, 0, length.out=maxBuckets)
    while (n < ndata) {
        if (time[n+1]>=Tstart+nbucket*bucketSize){
            bucketData[nbucket] = bucketSum / count
            sumCount = sumCount + count
            count = 0
            bucketSum = 0
            while (time[n+1]>=Tstart+(nbucket+1)*bucketSize) {
                nbucket = nbucket + 1
                EMPTYBUCKETFLAG[nbucket] = 1
            }
            nbucket = nbucket + 1
            count = 1
            n = n + 1
            bucketSum = data[n]
        } else {
            n = n + 1
            count = count + 1
            bucketSum = bucketSum + data[n]
        }
    } #end while
    if (count > 0) { # final bucket (incomplete)
        bucketData[maxBuckets] = bucketSum / count
    } else emptBucketCount = emptBucketCount + 1
    totalEmpty = sum(EMPTYBUCKETFLAG)
    avgNoInNonemptyBuckets = ndata / (maxBuckets - totalEmpty)
    #now fill empty buckets by by linear interpolation
    LIbucketData = seq(0, 0, length.out=maxBuckets)
    iLast = 1
    LIbucketData[1] = bucketData[1]
    for (i in 2:maxBuckets){
        if (EMPTYBUCKETFLAG[i] == 0 & EMPTYBUCKETFLAG[i-1]
            == 1 ) {
            LIbucketData[iLast] = bucketData[iLast]
            LIbucketData[i] = bucketData[i]
            for (j in iLast+1:i-1) LIbucketData[j] =
                bucketData[iLast] + (bucketData[i] - bucketData[iLast])*
                    (j-iLast)/(i-iLast)
            iLast = i
        }
        else if (EMPTYBUCKETFLAG[i] == 0) {
            iLast = i
            LIbucketData[i] = bucketData[i]
        }
    }
    ntmp = length(LIbucketData)
    bucketData = LibucketData[-(maxBuckets+1:ntmp)]
    result = list(bucketData, maxBuckets,totalEmpty,
        avgNoInNonemptyBuckets,
    EMPTYBUCKETFLAG, ntmp)
    return(result)
}
```

B.2. "first part" analysis in section 3.2

```
rm(list=ls(all=TRUE))
#Load User-Defined Functions
setwd("C:/Users/Geoff/Documents/R/GBC Defined Functions")
dump("XYZgetSignals_udf", file="XYZgetSignals_udf.R")
source("XYZgetSignals_udf.R")
dump("XYZspectrum_udf", file="XYZspectrum_udf.R")
source("XYZspectrum_udf.R")
dump("XYZbucketData_udf", file="XYZbucketData_udf.R")
source("XYZbucketData_udf.R")
#load Rssa R library from Install Packages
library(Rssa)
# end user defined functions

# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT
longestPeriod = 1000 # maximum acceptable period in days
shortestPeriod = 100 # shortest
periodDiffpercent = 10.0 # % of frequency or supposed period, used as
acceptance criterion
randomiseLinterp = TRUE
#NB user can set up a loop over the following variables and write output if
desired
Xfactor = 1 # change to adjust bucket size
dataStart = 1
baseBucketSize = 20
# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT

# STEP 1: Read in and select data
fileIn = "SU Per" # data is 3 col CSV file headers JD, mag and adjMag
setwd(paste0("C:/Users/Geoff/Documents/ASTRO/data analysis/", fileIn,
"/raw data"))
#D:/ or your own path here
tsIn←read.csv("biasAdjusted.csv") # data is 3 col CSV file headers JD,
mag and adjMag
plot(tsIn$adjMag,xlim=c(1,length(tsIn$adjMag)), xlab="", ylab="",
type="l", col="black",
lwd=2, main="complete series actual data")
if (dataStart > 1) ts = tsIn[-c(1:dataStart-1),] else ts = tsIn
ndata = nrow(ts)
mag<-ts$adjMag
timeJD = ts$JD

# STEP 2a: bucket data
bucketSize = baseBucketSize * Xfactor
tmp = XYZbucketData_udf(bucketSize,mag,timeJD,ndata)
bucketDates = seq(timeJD[1]+bucketSize/2,timeJD[ndata],by=bucketSize)
maxBuckets = unlist(tmp[2])
emptyBuckets = unlist(tmp[3])
avgFilledBucketCount = unlist(tmp[4])
bucketMag = unlist(tmp[1])
emptyFlag = unlist(tmp[5])
```

```
L = floor(maxBuckets/4)*2
magMean = mean(bucketMag)
bucketMag = bucketMag - magMean
outputVecCount = 10

# STEP 2b:
# calculate mean average change from bucket to bucket and randomise linterp values
if (randomiseLinterp) {
    bucketMagLagged = bucketMag[2:maxBuckets]
    delta = sum(abs(bucketMag-bucketMagLagged))/(maxBuckets-1)
    set.seed(0)
    randomNormal <- rnorm(maxBuckets)
    bucketMagRand = bucketMag
    for (i in 2:maxBuckets) {
        if (emptyFlag[i] == 1) bucketMagRand[i] = bucketMagRand[i] +
            delta*randomNormal[i]
        }
        bucketMag = bucketMagRand - mean(bucketMagRand)
    }

# STEP 3: automated SSA of actual data
x = bucketMag
x1 = x[1:L]
x2 = x[-(1:L)]
for (kk in 1:3) {
    if (kk == 1) { x = x - mean(x); Lx = L
    } else if (kk == 2) { x = x1 - mean(x1); Lx = L/2
    } else if (kk == 3) { x = x2 - mean(x2); Lx = L/2 }
    s←ssa(x, Lx, kind=”1d-ssa”)
    plot(s, type=”vectors”, idx=1:outputVecCount, xlim=c(1,Lx),
col=”black”, lwd=2)
    w←wcor(s, groups=c(1:outputVecCount))
    plot(w, title=”correlation matrix”)
    results = XYZgetSignals_udf(x, s, longestPeriod, shortestPeriod,
        bucketSize, periodDiffpercent, outputVecCount)
    actualTrendSignals = results[1]
    actualEVPeaks = results[2]
    actualPeriodSignals = results[3]
    actualSignalPeaks = results[4]
    count = length(actualPeriodSignals[[1]])
    signal = reconstruct(s, groups = list(EV = unlist(actualPeriodSignals)))
    XYZspectrum_udf(unlist(signal[1]), drawPlot=TRUE, “signal
        spectrum”, smoothing=”ar”)
    }
# STEP 4: fit AR(1) model for later simulation use
autoAR1 = ar(bucketMag, aic=FALSE, order.max=1)
alphaLI = autoAR1$ar
errors = autoAR1$resid
sigmaLI = sqrt(var(errors[2:maxBuckets], y=NULL, na.rm=TRUE))
write(“alphaAR, sigmaAR”, file = “actualDataAnalysis.csv”, ncolumns =
1, append = TRUE,
sep = “,”)
write(paste(alphaLI, sigmaLI, sep=”,”), file = “actualDataAnalysis.csv”,
ncolumns = 2,
append = TRUE, sep = “,”)
write(“ “, file = “actualDataAnalysis.csv”, ncolumns = 1, append = TRUE,
sep = “,”)
```

B.3. “second part” analysis in section 3.2

```
rm(list=ls(all=TRUE))
#Load User-Defined Functions
setwd(“C:/Users/Geoff/Documents/R/GBC Defined Functions”)
dump(“XYZgetSignals_udf”, file=”XYZgetSignals_udf.R”)
source(“XYZgetSignals_udf.R”)
dump(“XYZspectrum_udf”, file=”XYZspectrum_udf.R”)
source(“XYZspectrum_udf.R”)

# this helper function tests H1 signal frequency against H2
matchTest2 <- function(peakS1H1, peakS1H2, periodDiffpercent){
    hit = 0
        if ((peakS1H1>shortestPeriod) & (peakS1H1<longestPeriod)){
            if ((peakS1H2>shortestPeriod) & (peakS1H2<longestPeriod)){
                if (abs(peakS1H1-peakS1H2)<periodDiffpercent*(peakS1H1+
                    peakS1H2)/200){
                    hit = 1
                }
            }
        }
        return(hit)
    }
    # end user defined functions
    library(Rssa)

# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT
periodDiffpercent = 10.0 # % of frequency or supposed period, used as acceptance criterion
longestPeriod = 1000 # maximum acceptable period in days
shortestPeriod = 100 # shortest
bucketSize = 20 # used to calculate spectral peak in days
maxBuckets = 803
alpha = 0.71
sigma = 0.145
# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT

# simulate data, and perform analysis looking for a periodic signal
L = floor(maxBuckets/4)*2
LH = L/2
outputVecCount = 10
nsims = 1000
hitSimsCount = 1
set.seed(0)
hits = seq(0, 0, length.out=nsims) # number of hits within periofDiffPercent
for (j in 1:nsims){
    simulatedSeries <- arima.sim(list(ar=c(alpha,0,0)), sd=sigma,
    n=maxBuckets)
    y = simulatedSeries[1:L]
    y = y - mean(y)
    s<-ssa(y,LH,kind=”1d-ssa”)
    results = XYZgetSignals_udf(y, s, longestPeriod, shortestPeriod,
        bucketSize, periodDiffpercent, outputVecCount)
    signalPeaks = unlist(results[4])
    if(length(signalPeaks)==0)peakS1H1=0elsepeakS1H1=signalPeaks[[1]]
    y = simulatedSeries[-(1:L)]
    y = y - mean(y)
    s<-ssa(y,LH,kind=”1d-ssa”)
    results = XYZgetSignals_udf(y, s, longestPeriod, shortestPeriod,
        bucketSize, periodDiffpercent, outputVecCount)
    signalPeaks = unlist(results[4])
    if (length(signalPeaks) == 0) peakS1H2 = 0 else peakS1H2 =
        signalPeaks[[1]]
    # compare the strongest signal in H1 with first or second strongest in H2
    hits[j] = matchTest2(peakS1H1, peakS1H2, periodDiffpercent)
}
cat(sum(hits)*100/nsims, sum(hitsHalf)*100/nsims, “\n”)
```

B.4. Fourier spectrum and percentiles analysis in section 3.3

```
# plots spectrum of signal in the actual data together with envelopes derived from the spectra of
# signals in surrogate series

rm(list=ls(all=TRUE))
#Load User-Defined Functions
setwd(“C:/Users/Geoff/Documents/R/GBC Defined Functions”)
dump(“XYZgetSignals_udf”, file=”XYZgetSignals_udf.R”)
source(“XYZgetSignals_udf.R”)
dump(“XYZspectrum_udf”, file=”XYZspectrum_udf.R”)
source(“XYZspectrum_udf.R”)
dump(“XYZbucketData_udf”, file=”XYZbucketData_udf.R”)
source(“XYZbucketData_udf.R”)
library(Rssa)

# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT
periodDiffpercent = 10.0 # % of frequency or supposed period, used as acceptance criterion
```

```
longestPeriod = 1000 # maximum acceptable period in days
shortestPeriod = 100 # shortest
bucketSize = 20 # used to calculate spectral peak in days
fileIn = "SU Per"
setwd(paste0("C:/Users/Geoff/Documents/ASTRO/data analysis/", fileIn,
"/raw data"))
alpha = 0.71
sigma = 0.145
# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT

# STEP 1: Read in data, bucket, find signal and perform spectral analysis for
the chart
ts<-read.csv("biasAdjusted.csv")
ndata = nrow(ts)
mag<-ts$adjMag
timeJD = ts$JD
tmp = XYZbucketData_udf(bucketSize, mag, timeJD, ndata)
bucketDates = seq(timeJD[1]+bucketSize/2, timeJD[ndata], by=bucketSize)
bucketMag = unlist(tmp[1])
maxBuckets = unlist(tmp[2])
bucketMag = bucketMag - mean(bucketMag)
outputVecCount = 10
L = floor(maxBuckets/4)*2
s←ssa(bucketMag, L, kind="1d-ssa")
results = XYZgetSignals_udf(x, s, longestPeriod, shortestPeriod, bucketSize,
periodDiffpercent, outputVecCount)
actualPeriodSignals = results[3]
signal = reconstruct(s, groups = list(EV = unlist(actualPeriodSignals)))
spec.out←spectrum(unlist(signal[1]), plot=FALSE, method="pgram")
x<-spec.out$freq # all the frequencies on the x-axis
actual = spec.out$spec
nnn = length(x)

# STEP 2: simulate data, and perform analysis looking for a periodic signal
nsims = 1000
L = floor(maxBuckets/4)*2
outputVecCount = 10
set.seed(0)
power2 = matrix(0, nsims, length(x))
for (j in 1:nsims){
    y <- arima.sim(list(ar=c(alpha,0,0)), sd=sigma, n=maxBuckets)
    y = y - mean(y)
    s<-ssa(y,L,kind="1d-ssa")
    results = XYZgetSignals_udf(y, s, longestPeriod, shortestPeriod,
        bucketSize, periodDiffpercent, outputVecCount)
    actualPeriodSignals = results[3]
    signal = reconstruct(s, groups = list(EV = unlist(actualPeriodSignals)))
    spec.out = spectrum(unlist(signal[1]), plot=FALSE, method="pgram")
    if (j==1) frequency<-spec.out$freq # all the frequencies on the x-axis;
        standard intervals
    power2[j,] = spec.out$spec
    }
# find 10% and 2% envelopes
lower10 = c(nnn)
upper10 = c(nnn)
lower2 = c(nnn)
upper2 = c(nnn)
for (ifreq in 1:nnn) {
    datax = power2[,ifreq]
    lower10[ifreq] = quantile(datax,0.1)
    upper10[ifreq] = quantile(datax,0.9)
    lower2[ifreq] = quantile(datax,0.02)
    upper2[ifreq] = quantile(datax,0.98)
}
plot(x,actual, log="y", xlab="frequency", ylab="power", type="l",
    col="black", lwd=2, main=paste0(fileIn, ": signal and 10 and 2
    percentiles"))
lines(x, y=upper10, col="black", lty=3, lwd=1)
lines(x, y=lower10, col="black", lty=3, lwd=1)
lines(x, y=upper2, col="black", lty=1, lwd=1)
lines(x, y=lower2, col="black", lty=1, lwd=1)
```

B.5. MCSSA code

```
rm(list=ls(all=TRUE))
#Load User-Defined Functions
setwd("C:/Users/Geoff/Documents/R/GBC Defined Functions")
dump("XYZspectrum_udf", file="XYZspectrum_udf.R")
source("XYZspectrum_udf.R")
dump("XYZbucketData_udf", file="XYZbucketData_udf.R")
source("XYZbucketData_udf.R")
library(Rssa)
#install.packages("simsalabim", repos="http://R-Forge.R-project.org")
library(simsalabim)

# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT
longestPeriod = 1000 # maximum acceptable period in days
shortestPeriod = 100 # shortest
periodDiffpercent = 10.0 # % of frequency or supposed period, used as
acceptance criterion
bucketSize = 20
fileIn = "SU Per"
setwd(paste0("C:/Users/Geoff/Documents/ASTRO/data analysis/", fileIn,
"/raw data"))
# USER INPUT# USER INPUT# USER INPUT# USER INPUT# USER INPUT

# STEP 1: Read in data and bucket
ts<-read.csv("biasAdjusted.csv")
ndata = nrow(ts)
mag<-ts$adjMag
timeJD = ts$JD
tmp = XYZbucketData_udf(bucketSize, mag, timeJD, ndata)
bucketDates = seq(timeJD[1]+bucketSize/2, timeJD[ndata], by=bucketSize)
maxBuckets = unlist(tmp[2])
bucketMag = unlist(tmp[1])
L = floor(maxBuckets/4)*2
x = bucketMag - mean(bucketMag)
outputVecCount = 10

# STEP 3: MCSSA analysis
s←decompSSA(x, L, toeplitz = FALSE, getFreq = TRUE)
x.rc1 <- reconSSA(s, x, list(5:6)) # the signal
signalFreq = XYZspectrum_udf(unlist(x.rc1), drawPlot=TRUE,
"signalspectrum",
smoothing="ar")
x.rc2 <- reconSSA(s, x, list(1:4)) # trend
plot(x,type="l")
lines(x.rc1,col="red",lwd=2)
points(x.rc2,col="blue")
y = MCSSA(s, x, n=1000, conf = 0.9, keepSurr = FALSE, ar.method="mle")
plot(y, by = "freq", normalize = FALSE, asFreq = TRUE,
lam.pch = 1, lam.col = "black", lam.cex = 1, sig.col = "black",
sig.pch = 19, sig.cex = 1, conf.col = "darkgray", log = "xy",
ann = TRUE, legend = TRUE, axes = TRUE)
```