The Astrophysical Journal, 474:464–473, 1997 January 1

AN EFFICIENT ALGORITHM FOR THE DETECTION OF INFREQUENT RAPID BURSTS IN TIME SERIES DATA

A. B. Giles[1]
Laboratory for High Energy Astrophysics, Goddard Space Flight Center, Greenbelt, MD 20771; barry@pcasun3.gsfc.nasa.gov
Received 1996 February 12; accepted 1996 July 24

ABSTRACT

Searching through data for infrequent rapid bursts is a common requirement in many areas of scientific research. In this paper, we present a powerful and flexible analysis method that, in a single pass through the data, searches for statistically significant bursts on a set of specified short timescales. The input data are binned, if necessary, and then quantified in terms of probabilities rather than rates or ratios. Using a measure-like probability makes the method relatively count rate independent. The method has been made computationally efficient by the use of lookup tables and cyclic buffers, and it is therefore particularly well suited to real-time applications. The technique has been developed specifically for use in an X-ray astronomy application to search for millisecond bursts from black hole candidates such as Cyg X-1. We briefly review the few observations of these types of features reported in the literature, as well as the variety of ways in which their statistical reliability was challenged. The developed technique, termed the burst expectation search (BES) method, is illustrated using some data simulations and archived data obtained during ground testing of the proportional counter array (PCA) experiment detectors on the *Rossi X-Ray Timing Explorer* (*RXTE*). A potential application for a real-time BES method on board *RXTE* is also examined.

Subject headings: methods: data analysis — methods: statistical — stars: individual (Cygnus X-1) — stars: variables: other — X-rays: stars

1. INTRODUCTION

Searching for occasional rapid features or bursts in experiments in which photons are detected one by one can be quite challenging when the count rate is relatively low and Poisson statistics apply. A classic example of the use of this type of analysis is in searching for rapid variability from astronomical X-ray sources. This has been particularly applicable for the efforts made to date to identify the existence of a black hole as the origin of the X-rays from the source Cyg X-1. The *Rossi X-Ray Timing Explorer* (*RXTE*), launched on 1995 December 30, provides new opportunities for this type of study, and Cyg X-1 style variability forms a natural example with which to illustrate this new burst search method.

The earliest reported observations of millisecond timescale variability or bursts from Cyg X-1 were those reported by Rothschild et al. (1974, 1977). The analysis method employed revealed a total of 12 bursts. Another early observation of rapid variability was that reported by Ogawara et al. (1977). Various authors reported statistical difficulties with the analysis methods employed, particularly given the longer timescale shot noise variability seen from Cyg X-1 (Terrell 1972). Weisskopf & Sutherland (1978) modeled the shot noise using suitable parameters for the frequency, decay time, and percentage of flux within the shots, and they were able to produce bursts of similar significance to those reported by Rothschild et al. (1974, 1977). The located model bursts occurred as expected within the enhanced count rate periods, but not right at the sharp rising edge of the shots. Weisskopf & Sutherland (1978) showed that neglect of these longer timescales leads to "events" being found that have the duration of the basic sampling bins. The work of Giles (1978, 1981) indicated that such effects were only sufficient to bias the probability of acceptance of marginal cases, such as those being reported at the time, and the basic searching approach adopted by Rothschild et al. (1974) is still a useful one. A complete description of the mathematical theory for the shot noise model has been given by Sutherland, Weisskopf, & Kahn (1978).

The relationship between the duration of any enhancement, whether caused by exponential shots or not, and the averaging interval selected for the local mean was discussed by Giles (1978). Even the actual selection and binning of the basic temporal samples is important, as pointed out by Press & Schechter (1974). In Rothschild et al. (1974), groups of 320 μs bins (minimum provided by telemetry) were taken to build the basic 1.28 ms bin sizes, while in Rothschild et al. (1977) bins of 160 μs duration were available. Press & Schechter (1974) discuss the difference between these fixed bins being combined from a random start point or being specially selected by a clustering search to emphasize any possible features. They discuss some correction statistics for this second case. Questions such as "Is the phasing of the adopted binning retained throughout the entire data set?" and "Was it defined a priori?" become important if the features eventually reported have borderline significance.

The most extreme situation occurs when all the X-ray counts are individually time-tagged and complete freedom of binning becomes possible. The first data of this type were obtained in 1976 by Giles (1981) when photons from a large area detector were time-tagged to 2 μs. Unfortunately, because of an attitude control failure, only a short observation was made, but several bursts of similar significance to those found by Rothschild et al. (1974, 1977) were located. The first satellite datum obtained for Cyg X-1 with rapid sampling was an observation by Meekins et al. (1984). This single, pointed observation with the *HEAO* A-1 experiment used a special high-rate telemetry mode for a short period of time but saw no millisecond bursts. Additional

[1] Universities Space Research Association.

observations of Cyg X-1, such as those with *EXOSAT* (Belloni & Hasinger 1990) and *Ginga* (Miyamoto & Kitamoto 1989; Negoro, Miyamoto, & Kitamoto 1991) have also not reported any millisecond or submillisecond bursts.

The proportional counter array (PCA) experiment on board *RXTE* is considerably more powerful than any previous experiment and can return vast amounts of high time resolution data. In this paper, we describe a fast and efficient search method for individual rapid burst events. This is rather different from studying the low-level quasi-periodic oscillation type features, first reported by van der Klis et al. (1985) from GX 5−1 and subsequently reported in many X-ray binary systems. The search for isolated features is also quite different from that for millisecond pulsations in X-ray binaries, as reported by Wood et al. (1991) and Vaughan et al. (1994).

2. THE BASIC POISSON ANALYSIS METHOD

The basic approach followed here is developed from that described by Rothschild et al. (1974) to detect millisecond bursts from Cyg X-1. In this method, the basic bins used had a duration of 1.28 ms, and the local mean count per bin was obtained from a sample window of 409.6 ms or 320 of these bins. All bins within a sample window were then checked against the local mean for that sample. To perform the test, Rothschild et al. (1974) introduced the concept of an expectation value $N(n)$ for the number of bins containing n counts within each sample, using the mean count per bin for each sample and Poisson statistics. The Poisson probability for a single bin is given by

$$P(r;\, \mu) = \frac{e^{-\mu}\mu^{r}}{r!}\,, \qquad (1)$$

where r is the integer count in a bin and μ is the local mean per bin. The expectation value for any bin count value r is the product of the Poisson probability for a single bin $P(r;\, \mu)$ and the number of bins in the sample window. Rothschild et al. (1974) then searched all bins in each sample window to identify any that had a ≤ 0.01 chance expectation value. The value of the mean was assumed to be constant across the entire sample window.

The burst expectation search (BES) method differs from Rothschild's approach in a number of ways. Rather than simply taking each bin in turn and calculating its chance expectation, we acknowledge that the counts per bin are, by definition, integer numbers, and for a range of suitable values of r (zero to a few hundred) calculate the complimentary set of means that corresponds to a predefined expectation value such as 0.01. Each of these calculated values can be considered to be a threshold mean in the sense that, when evaluating real data, a measured mean will fall above or below it. The upper value of r simply depends on the application. The BES method also tests the data against a range of specified expectation values in a single pass through the data.

The initial data stream is either binned up into counts per unit time interval, or it is automatically provided in this form by a telemetry system structure. For the BES method, the basic binning process can occur on any suitable short timescale, and it is convenient to work in powers of 2 in microseconds. Numbers of 64 to 1024 μs are suitable values. Some longer time interval, perhaps 128, 256, or 512 times longer than the basic bin size, is then used to define a local mean count rate centered around the basic bin of interest. The bin of interest is included in the calculation for the local mean. The bin count and local mean are then tested to see if the bin count has a low probability of simple chance occurrence. This again differs from Rothschild's method in that a running average local mean is used, rather than assuming it is constant for the duration of the sample window.

Taking the local mean as constant over the sample window is a potential problem if step function edges, rather than sharp bursts, are just entering or leaving the specified window. A sudden large increase in count rate occurring near the end of the averaging window is an extreme example, but Cyg X-1 is very variable, and power spectra show significant energy from $\sim$10 ms to 10 s timescales. Since one cannot simply assume that the mean is constant over the whole observation, the choice of the averaging interval becomes a trade-off. There must be enough bins to measure the mean to an acceptable precision (320 bins for Rothschild, 256 bins in this paper), but not so many that longer timescale intensity increases within the averaging window make the mean too much of an underestimate. This would bias the subsequent Poisson testing and result in bursts, if present, having a higher statistical significance than they really should. The use of a running mean helps to further reduce these effects by making the test carried out on each bin use a mean that is more representative of its local environment.

In cases in which the long-term mean is relatively constant, the running mean offers no advantage, and the distribution of the bin counts over the whole data sequence can be examined to identify outlying cases. Such a distribution can also be compared directly with an expected distribution generated from the observed and stable mean using Poisson statistics.

In addition to the introduction of a running mean, the BES approach also uses a lookup-table method that minimizes computation once the lookup table itself has been computed. To construct the lookup table it is necessary to define the set of expectation values the data are to be tested against, the number of bins in the averaging window, and the maximum bin count for which we wish to test. As an example, assume that we are looking for bins that have a 0.01 or less chance expectation, that the averaging window has 256 bins, and that a count 7 bin is being considered. We now calculate what the threshold mean level would be that gives just the count 7 bin this level of significance. This can be done using an iterative algorithm that is repeated until the calculated mean is stable, within some defined small limit. When doing this calculation, the Poisson probability must be multiplied by the total number of bins within the window, which is why this parameter is an input variable. For the particular Table 1 location example above, we obtain a value of 0.9018 for the mean. To save space, not all bin count rows are shown in Tables 1 and 2. It is now a simple matter to evaluate any given bin count by testing it against each of the Poisson table row entries against that count value. If the measured window mean is lower than the tabulated threshold mean, then the bin count chance expectation is less than the column heading value, and the "burst" is significant.

In the above discussion, we have dealt with the mean value within the sample window, but deriving the actual

TABLE 1

THRESHOLD AVERAGES WITHIN A 256 SAMPLE WINDOW

	EXPECTATION VALUES					
HEIGHT	0.04	0.02	0.01	0.005	0.002	0.001
2	0.0178	0.0126	0.0089	0.0063	0.0044	0.0028
3	0.1012	0.0798	0.0630	0.0498	0.0394	0.0289
4	0.2644	0.2198	0.1832	0.1529	0.1278	0.1009
5	0.4988	0.4281	0.3683	0.3174	0.2739	0.2259
6	0.7930	0.6950	0.6105	0.5373	0.4737	0.4017
7	1.1369	1.0114	0.9018	0.8057	0.7209	0.6238
8	1.5226	1.3698	1.2351	1.1158	1.0098	0.8867
9	1.9437	1.7641	1.6046	1.4624	1.3350	1.1859
10	2.3955	2.1895	2.0057	1.8407	1.6921	1.5172
15	4.9968	4.6676	4.3690	4.0966	3.8471	3.5476
20	7.9885	7.5485	7.1461	6.7762	6.4345	6.0204
25	11.2339	10.6927	10.1956	9.7362	9.3098	8.7902
30	14.6605	14.0257	13.4407	12.8984	12.3935	11.7756
40	21.8965	21.0916	20.3469	19.6539	19.0058	18.2091
60	37.3404	36.2418	35.2214	34.2679	33.3726	32.2665
80	53.5699	52.2175	50.9591	49.7809	48.6723	47.2991
100	70.2956	68.7159	67.2445	65.8653	64.5659	62.9540

NOTE.—Threshold Poisson means for the range of expectation values 0.04, 0.02, 0.01, 0.005, 0.002, and 0.001, assuming an averaging window of 256 bins. Not all integer bin counts are shown down the page. This Poisson array table is constructed using methods described in § 2.

mean is unnecessary. We can take the sum within the window and multiply the values in Table 1 by the number of bins in the window. Then we test as before, but we compare the sum in the window against the sums in the new table. Table 2 shows the new computed table with the corresponding entries now appearing as integers because the sum is by definition a summation of integers. The equivalent value of the mean for the above count 7 example is now 230. Notice that Table 2 can be conveniently represented by 2 byte unsigned integers for real-time applications using ROM or RAM storage.

3. FAST ANALYSIS ALGORITHM

The BES method uses lookup tables and simple algorithms, which are computationally efficient, for rapidly examining large data volumes. For analyzing archived data, the processing speed is less important than for real-time applications, but the method described below produces a concise output that clearly illustrates any excessive bursting in the input data stream. Some comments on the computational efficiency are made in a later section. A particular feature of this analysis method is that the data are interpreted in terms of probability levels, rather than rates or ratios, and are therefore largely independent of the actual mean input data rate that can drift up and down through the observation. The BES method is particularly suitable for real-time analysis systems, and, in a later section, we discuss its application to the Electronic Data System (EDS) that is used by the PCA experiment on board *RXTE*.

The BES method requires the development and maintenance of five types of interrelated data arrays. The dimensions of these data arrays are consequences of defining a rigid structure to the search method—namely, a number of bins across the sampling interval, a range of expectation test values, a number of time bin sampling scales, and the range of a few hundred integer counts. These five array types are given the following names: poisson array; working array; result array; normalization array; and burst excess array. We will now consider each of these in turn.

TABLE 2

THRESHOLD TOTALS WITHIN A 256 SAMPLE WINDOW

	EXPECTATION VALUES					
HEIGHT	0.04	0.02	0.01	0.005	0.002	0.001
2	4	3	2	1	1	0
3	25	20	16	12	10	7
4	67	56	46	39	32	25
5	127	109	94	81	70	57
6	202	177	156	137	121	102
7	291	258	230	206	184	159
8	389	350	316	285	258	227
9	497	451	410	374	341	303
10	613	560	513	471	433	388
15	1279	1194	1118	1048	984	908
20	2045	1932	1829	1734	1647	1541
25	2875	2737	2610	2492	2383	2250
30	3753	3590	3440	3301	3172	3014
40	5605	5399	5208	5031	4865	4661
60	9559	9277	9016	8772	8543	8260
80	13713	13367	13045	12743	12460	12108
100	17995	17591	17214	16861	16528	16116

NOTE.—A repeat of Table 1 but with the mean values multiplied by 256 and converted to integers. This table illustrates a more useful form suitable for fast analysis and real-time hardware systems.

3.1. *Poisson Array*

The Poisson array contains a table of threshold mean values computed using the method outlined in § 2. This table is actually a constant table in the sense that, given a set of expectation values, it can be computed once and it then remains constant. Depending on the application, this table can be recomputed each time the program is started, read from a precomputed disk file, read from a ROM chip, or uplinked from the ground. Table 1 shows part of a typical Poisson array, with each vertical column representing a particular chance expectation value. The horizontal rows are bin counts in integers, and a complete table would go—in increments of 1—from 0 to the largest number of interest.

The selection of a maximum of 100 counts per bin used in constructing Table 1 is somewhat arbitrary, but it is greater than the maximum counts per bin during the data analysis and simulation results presented later. Table 1 does not contain entries for bin counts of 0 or 1, and this requires some explanation. Bins with zero counts, although potentially of interest, must by definition be deficits rather than excesses (bursts) and are therefore not of interest here. Bins with 1 count also pose a problem because they could equally be a background event contributing to the average or a " burst " event. There is no way of knowing which is the case, and, although excessive bunching of 1's in time would be of interest, this is not applicable to the present analysis method. Bin counts of 2 are also relatively insensitive to meaningful statistical interpretation. In particular, reference to Table 1 or Table 2 shows that for a bin count of 2 and expectation value of 0.002, the mean over the 256 sample window is actually less than the bin count value itself. A detailed discussion of the confidence limits for small numbers of events in astrophysical data can be found in Gehrels (1986).

To compute this array, it is only necessary to define the set of expectation values and the number of bins in the averaging window (number of columns in the working array).

3.2. *Working Array*

The working array is continuously updated with incoming binned data samples and can exist in three states: empty, loading, and operational. The empty state exists at the start of processing the incoming data stream or following a reset that may have been performed to clear out corrupted data. The loading state is illustrated in the small example of a working array shown in Table 3. This table also illustrates the basic principles of operation.

Each horizontal row of cells within the array is a ring buffer that has associated with it a head/tail pointer, a burst pointer, and a sum register. These extra registers are also shown in Table 3, where the ring buffers have a length of only 8. In practice, the ring buffer length, the number of columns in the array, will usually be 128, 256, or 512. The top row of the working array represents the fastest cycling ring buffer, and each row below represents a slower cycle time by a factor of 2. Only six rows are shown in Table 3, but the number can be increased until the averaging timescale becomes larger than required, or the counts per bin become so large that analysis for the complimentary result array is invalid. The sampling for the basic input bins and the dimensions of the working array can be adjusted to suit the application. Every time a new bin count value is available, it is placed on the top row at the location pointed at by the head/tail pointer, which is then incremented. This bin count is also added to that row's sum register, and the value overwritten is subtracted from the sum register, which therefore keeps a running value for the total sum of bin counts on the top row. The burst pointer is defined to point at a register halfway back along the ring buffer, so for each new input bin count this pointer also increments and cycles modulo the row length. In Table 3, the array is not fully loaded, but assuming it was, we would have at any time a row number, row sum, and bin count pointed at by the burst pointer with which to perform a significance test.

After the arrival of two bin counts, they can be added together and placed in the location pointed at by the head/tail register for row 2. Row 2 then updates its pointers and running sum, and it can perform a significance test as above. Similarly, after row 2 has had two updates, they can be summed and row 3 can be updated. This updating process continues indefinitely, and the cascading process occurs in sequence until, at regular intervals, all the rows in the working array will have an update and burst significance test performed at the same moment. In Table 3, only seven bin count inputs have entered the top row, and the cascade has not yet reached the first element of the last row. For the array size in Table 3, the bottom row will be filled and poised to roll round for the first time, after 256 inputs on the top row (row length $\times$ $2^{\text{no. of rows}-1}$), 128 on the second, etc., and the array is now defined as loaded. Up until this point, all attempts to test the bins for burst significance will have been inhibited, but once the loaded condition is detected, the test becomes enabled. If the working array in Table 3 were now cycled through 256 more inputs, there would have been 256 burst tests performed on bins in the top row, 256/2 performed on the next row down, etc.

To illustrate the temporal scaling for a more realistic array size, consider the case presented in Table 4. In this example, the basic bin width for input to row 1 is 64 μs, and the working array has a row length of 256 bins and a total

TABLE 3

LOADING A SMALL WORKING ARRAY

	Column										
Row	1	2	3	4	5	6	7	8	Head Pointer	Burst Pointer	Sum
1......	5	3	7	4	3	6	4	...	8	4	32
2......	8	11	9	...	...	...	...	...	4	8	28
3......	19	...	...	...	...	...	...	...	2	6	19
4......	...	...	...	...	...	...	...	...	1	5	0
5......	...	...	...	...	...	...	...	...	1	5	0
6......	...	...	...	...	...	...	...	...	1	5	0

NOTE.—This table illustrates a small working array whose loading has just been started so many locations are still empty. The relationship between the working array and the head pointer, burst pointer and sum vectors are illustrated. These relationships are explained in more detail in § 3.2.

TABLE 4

SUMMARY OF A MORE TYPICAL WORKING ARRAY

Row	Bin Size (ms)	Mean Counts per Bin	Total No. of Bins	Averaging Period (ms)	Mean Counts per Average Period	No. of Average Periods
1......	0.064	0.32	32768	16.384	81	128
2......	0.128	0.64	16384	32.768	163	64
3......	0.256	1.28	8192	65.536	327	32
4......	0.512	2.56	4096	131.072	655	16
5......	1.024	5.12	2048	262.144	1310	8
6......	2.048	10.24	1024	524.288	2621	4
7......	4.096	20.48	512	1048.576	5243	2
8......	8.192	40.96	256	2097.152	10486	1

NOTE.—After a full cycle of input data to a more typical working array a summary of its contents might look like that shown here. The length of a complete cycle depends on the number of rows and columns in the working array. In this example the full cycle lasts 0.008192 $\times$ 256 s.

of eight rows. Within Table 4, the input count rate or source intensity is assumed to provide 5120 counts s^{-1}. Although the processing of incoming numbers occurs continuously, this working array can be considered to have a complete cycle time of about 2.1 s. It will be apparent that the combination of (1) always working in powers of 2 and (2) the cascaded summation method, results in a simple and efficient algorithm that can also have a flexible row length (power of 2 in this paper) and number of rows (any integer). The above description may sound a little complicated but, in fact, it is very easy to implement.

3.3. *Result Array*

The result array is continuously updated once the working array is loaded, but at defined regular intervals it is read out and reset to zero. This might occur at one-fourth, one-half, or integer multiples of the complete cycle time. The number of columns in the result array is the same as the number of defined expectation values in the Poisson array. The number of rows in the result array is equal to the number of rows in the working array. When a test is made for an excess, the calling routine provides the row number, row sum, and bin count of the potential burst. The row number provides the index for the possible result array updates, and the value of the bin count being tested is used as the row index to obtain the set of threshold sums for all the column entries in the Poisson array table. Each is then compared with the row sum provided by the calling routine, which will either exceed or not exceed each of them. If the row sum is less than a sum from the table, then the accompanying bin count is considered to have an excess (a burst), since its chance expectation value, if computed, would be less than that of the expectation value in the corresponding column heading. Assuming that at least one or more columns produce a significant result, then the column with the most significance (least likely by chance) is used as the X index to increment the result array. Only one cell (or none) is incremented for each test. After a period of time, it is possible that the regular series of analysis tests may have detected no significant excesses in bin counts and all locations in the result array will remain at zero.

If the same parameters and duration used to construct Table 4 are retained, then a possible outcome for the result array might look like that shown in Table 5. Such a table can then be produced at regular intervals. Notice that, except for building the Poisson array in the first place, obtaining this table has required no multiplication or divisions, and has only required the use of unsigned 2 byte integers. This table is also source-intensity invariant, since it deals only in probabilities.

Each result array has three basic components: expectation value, bin size, and bin count. A regular series of result arrays can be accumulated over time to build up a multidimensional data cube, where the integration sample number becomes a fourth component. Various parameter combinations can then be selected from this data set. For example, a three-dimensional graph for the 0.01 chance expectation value could take the vertical axis to be the bin counts, the horizontal axis to be the time bin sizes, and the depth axis to be the integration samples, with time running from the back to the front. With a figure of this type, it is possible to follow the form and evolution of any included features.

3.4. *Normalization Array*

The values within the rows and columns of any general result array should always contain a kind of "fixed" pattern, assuming that the data source has Poissonian noise statistics. In Table 6, we show a normalization array that has been computed to suit the set of expectation test values and number of rows used in Table 5. The entries in this table are calculated by multiplying the chance expectation values by the number of averaging windows within the data sample, since the expectation value is already referenced to the averaging window width. For example, multiplying the 0.01 expectation value for row 1 of Table 5 by 128, the number of averaging windows in a complete analysis cycle for row 1 (see Table 4) gives a value of 1.28 for the chance occurrence of this result array element. This assumes that the expectation values within each averaging window are independent and that taking 128 sets of observations simply means that any particular feature is 128 times more likely to be seen. Some of the symmetry apparent in Table 6 is an artifact of the spacing of the six chosen expectation values.

Although this type of exact normalization is attractive, it turns out to be a little too simplistic due to the rapid sensitivity threshold over which bins go from being nonsignificant to highly significant. Suppose a large, fast, single-bin spike exists in the data that exceeds the expectation value for the uppermost right-hand element in a result array. As

TABLE 5

Typical Result Array for the Working Array Setup in Table 4

	Expectation Values					
Row	0.04	0.02	0.01	0.005	0.002	0.001
1......	67	21	1	0	0	0
2......	35	13	0	0	0	0
3......	17	4	0	0	0	1
4......	6	0	0	0	0	1
5......	1	0	0	0	0	1
6......	0	0	0	0	0	0
7......	0	0	0	0	0	0
8......	0	0	0	0	0	0

Note.—A possible result array state for a working array setup like that in Table 4. This imaginary set of results illustrates how "rare" events with a 0.001 expectation value might be seen superimposed on a low-expectation noise background.

TABLE 6

Typical Normalization Array for Table 5

	Expectation Values					
Row	0.04	0.02	0.01	0.005	0.002	0.001
1......	5.12	2.56	1.28	0.64	0.256	0.128
2......	2.56	1.28	0.64	0.32	0.128	0.064
3......	1.28	0.64	0.32	0.16	0.064	0.032
4......	0.64	0.32	0.16	0.08	0.032	0.016
5......	0.32	0.16	0.08	0.04	0.016	0.008
6......	0.16	0.08	0.04	0.02	0.008	0.004
7......	0.08	0.04	0.02	0.01	0.004	0.002
8......	0.04	0.02	0.01	0.005	0.002	0.001

Note.—Table 4 shows that many more samples were processed at the shorter time intervals and a normalization array can be constructed to match the result array shown in Table 5 by multiplying the expectation values by the respective number of samples tested. This results in a normalization array of the chance occurrences for each array element in Table 5.

TABLE 7

Burst Excess Array—Ratio of Table 5 Divided by Table 6

	Expectation Values					
Row	0.04	0.02	0.01	0.005	0.002	0.001
1......	13.08	8.2	0.78	0	0	0
2......	13.67	10.16	0	0	0	0
3......	13.28	6.25	0	0	0	31.25
4......	9.37	0	0	0	0	62.5
5......	3.1	0	0	0	0	125
6......	0	0	0	0	0	0
7......	0	0	0	0	0	0
8......	0	0	0	0	0	0

Note.—A normalized result can be obtained by dividing all the entries in Table 5 by those in Table 6. This shows the "excess" ratio of activity at any particular timescale and expectation value. In this hypothetical example, this excess ratio clearly shows activity at a low level on short timescales, which one might quantify as noise. The few 0.001 expectation value features added to Table 5 now show up quite dramatically.

this "spike" bin count is diluted in successive rows down the table, it may still exceed the threshold mean values in that column in the next row, and, perhaps, also the next. On the next row, it may be so smoothed out that it suddenly only satisfies a much higher expectation value and appears some columns to the left. Depending on the precise numbers, it may not even qualify for an entry on this row and therefore only ever appear in the right-hand column. Because of this complex sensitivity, the expectation pattern formed subtly depends on the actual input rate, but for any random data set it will, after sufficient integration, roughly resemble the idealized and symmetric normalization array described above.

3.5. *Burst Excess Array*

A burst excess array can be formed by dividing the result array by the normalization array. An example of this is given in Table 7 (Table 5 divided by Table 6). In such an array, values significantly greater than 1 indicate an excess above expected occurrence, and values significantly below 1 indicate a deficit. This is only a qualitative evaluation due to the previously described problems defining the normalization array, but it does provide a general way to search for excess activity.

4. INTERPRETATION

Table 5 is intended to give a plausible impression of the possible contents of the result array after about 1 s. There appear to be a few significant bursts and also some "noise" down at the level of chance occurrence that might be expected considering the number of trials carried out. The distribution of counts within the result array will clearly depend on the variability characteristics of the incident count rate, but the following general observations can be made:

1. Faster variability or "spikiness" tends to occur in the upper rows.
2. Slower variability tends to occur toward the bottom rows.
3. There will be more entries toward the top of the table, simply because more trials have been carried out.
4. There will be more entries in the higher chance expectation columns to the left-hand side.

The result of these combined effects is for the entries to cluster to the top left-hand side of a typical result array. Note that any features causing increments in, for example, the 0.001 column of Table 5 must also exceed the expectation values in the columns to its left-hand side, but, according to the definition employed here, only the right most or lowest expectation column satisfying the test on each row actually gets incremented. Any significant burst may also appear in successive rows, one beneath the other, until it becomes too diluted to stand out above the local mean for that particular row. Tables 5, 6, and 7 represent a rather contrived example, and it is necessary to use some real or simulated data to illustrate the contents of a more realistic result array.

5. EVALUATION EXAMPLES

Three types of data are used in the following sections to illustrate the use of the BES method.

5.1. *PCA Ground Test Data*

Some extended data sets of archived PCA ground calibration tests have been examined to illustrate the BES algorithm. A description of the PCA detectors for *RXTE* can be found in Zhang et al. (1993). The flight system consists of five almost identical detectors. The archived data sets store 34 bits for every detected PCA event, but for the BES analysis we are only concerned with the time tag that is available with 1 μs resolution. With a Fe^{55} calibration source over the detector, providing an average rate of ~ 3200 counts s^{-1}, a continuous run of 28,672 s produces the result array shown in Table 8. This table summarizes the bin count expectations of more than 9×10^7 photons, with over 5.7×10^7 basic 0.5 ms bins being tested. A short sample of the data is shown in Figure 1. This data set is from a single detector.

At first sight, Table 8 may appear to suggest a few excess event clusters, but, with the exception of a single example, this is not the case. The formal probability associated with the individual bins is a very strong function of marginal increases or decreases in the integer bin counts. Raising a bin count by 1 leads to a very much lower chance expectation value, while lowering a bin count by 1 leads to a much higher chance expectation value. For example, the expectation of 0.0002 for the count 11, mean 1.5742 burst (see Table

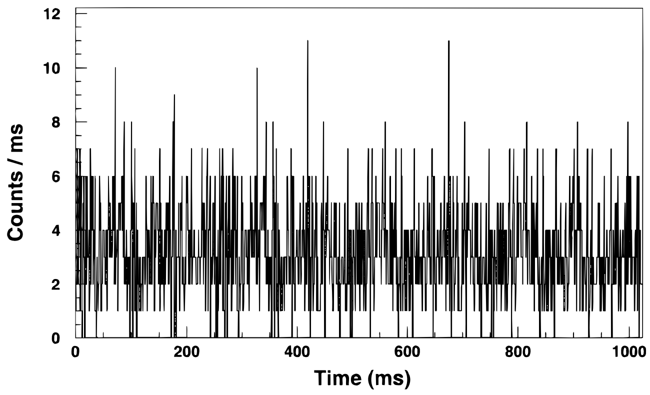

Fig. 1.—Light curve of a typical data segment for a PCA detector taken from the archives of prelaunch ground test data. An ^{55}Fe source illuminates the detector to produce a mean rate of ~ 3200 counts s^{-1}. The plot spans 1.024 s with the time axis having a resolution of 1 ms.

TABLE 8

Result Array from PCU Ground Test Data Similar to That Illustrated in Figure 1

Bin Size (ms)	Expectation Values								
	0.04	0.02	0.01	0.005	0.002	0.001	0.0005	0.0002	0.0001
0.5	524	592	320	47	79	8	8	2	3
1.0	1165	242	240	44	58	9	14	1	1
2.0	605	218	142	53	31	8	8	2	0
4.0	428	150	90	36	13	7	1	0	1
8.0	282	129	70	33	16	5	1	2	0
16.0......	200	112	50	17	14	6	1	0	0

Note.—The result array generated for real PCA detector data obtained during prelaunch ground testing. The detector was irradiated with a ^{55}Fe calibration source to produce a count rate of ~ 3200 counts s^{-1}. This table is a summation over 28,672 s of data or more than 9×10^{-7} photons. A typical short data segment is shown in Fig. 1.

9) would reduce to an expectation of only 0.0025 for a count 10 bin with the same mean background.

The 12 events that contributed to the two right-hand columns in Table 8 represent the most significant features found in many millions of bins of data, and their details are given in Table 9. Only a single case seems to stand out, and the data for the count 16, 0.5 ms bin example has been carefully examined. This is possible for the PCA test data since all the bits generated by a single event in the detectors are retained by the ground logging system. This is in contrast to the flight situation, where on-board telemetry constraints dictate that, for all but the weakest observed sources, the additional diagnostic flag bits describing the various detector layer and anticoincidence information are not retained. This anomalous count 16 bin contains two short strings of what are known as "no flag" events, which, under flight conditions, the processing system would have dropped. They were retained during this particular test, since the processing system was running in what is known as "transparent mode." Further discussion is outside the scope of this paper and requires detailed knowledge of the technical aspects of the PCA and its event processing hardware and software.

TABLE 9

Features in the Two Right-Hand Columns of Table 8

Number of Tests	Bin Width (ms)	Expectation Values			
		0.0002		0.0001	
		Height	Mean	Height	Mean
57,335,808......	0.5	11	1.5742	12	1.6367
57,335,808......	0.5	11	1.5625	12	1.6367
57,335,808......	0.5	...	...	16	1.6523
28,667,904......	1.0	16	3.4414	18	3.2539
14,333,952......	2.0	23	6.6875	...	...
14,333,952......	2.0	22	6.2656	...	...
7,166,976	4.0	...	...	34	12.7617
3,583,488	8.0	54	25.8945	...	...
3,583,488	8.0	54	26.0195	...	...
1,791,744	16.0	...	...	...	...

Note.—Features in the PCA ground test data set that are represented in the two right-hand columns of Table 8. These have the expectation values of 0.0002 and 0.0001. The columns show the bin width (ms), the bin count (height), the mean rate of the surrounding 256 bin data window (mean) and the number of bins tested to find the specified example.

5.2. *Simulated Random Data*

In order to examine more critically the results produced by the BES method, an experiment was performed in which the input bin count stream was simulated using algorithms from Press et al. (1992). This is a straightforward process when using the algorithm called poidev. This routine uses the ran2 random number generator and the gammln function. The simulation was run for the same equivalent duration and mean count rate as that for the PCA ground test data discussed in the previous section. The final contents of the result array are shown in Table 10, which can be directly compared with Table 8.

To first order, there is reasonable agreement considering the large number of bins being tested, but the simulation consistently has more features than the real PCA data. This is encouraging in the sense that it is not the real detector that has excess features. But why is this so, considering that the simulation was purposely set up to produce equivalent data? The answer is that the PCA detectors and electronics have a number of different types of dead time, whereas the simulation does not. The simulation has more time (no dead time) in which to randomly cram in more events, and there-

TABLE 10

Result Array from Simulated Constant Rate Data

Bin Size (ms)	Expectation Values								
	0.04	0.02	0.01	0.005	0.002	0.001	0.0005	0.0002	0.0001
0.5	1097	1577	717	94	285	29	28	12	11
1.0	2482	440	533	122	123	25	21	2	13
2.0	1131	354	325	115	56	31	11	1	2
4.0	706	301	132	78	66	24	11	2	1
8.0	483	175	104	54	30	8	3	2	1
16.0......	265	118	80	17	14	0	2	1	1

Note.—The result array generated from simulated data designed to match the mean count rate of the bench test data used to generate Table 8. This table is a summation over 28,672 s of data.

TABLE 11

Burst Excess Array for Result Array in Table 10

Bin Size (ms)	Expectation Values								
	0.04	0.02	0.01	0.005	0.002	0.001	0.0005	0.0002	0.0001
0.5	0.12	0.35	0.32	0.08	0.51	0.13	0.25	0.27	0.49
1.0	0.55	0.20	0.48	0.22	0.44	0.22	0.38	0.09	1.16
2.0	0.51	0.32	0.58	0.41	0.40	0.55	0.39	0.09	0.36
4.0	0.63	0.54	0.47	0.56	0.94	0.86	0.79	0.38	0.36
8.0	0.86	0.63	0.74	0.77	0.86	0.57	0.43	0.71	0.71
16.0	0.95	0.84	1.14	0.49	0.80	0.00	0.57	0.71	1.43

Note.—The burst excess array generated from simulated data. This is simply the ratio of the result array (Table 10) divided by the corresponding normalization array (not shown). Values should typically be centered about 1.0 with greater values indicating an excess of features above those expected.

fore at a low level it can produce both more features at any particular expectation value and some features that have a lower chance expectation, than for a real detector. Dead-time effects are particularly severe for the shorter time bins, which accounts for the deficit in the first row of Table 8. The burst excess array is presented in Table 11 and clearly shows the "loss" along the 0.5 ms row. Since each X-ray event seen by a single real detector and not vetoed by its anticoincidence system produces at least 10 μs of dead time, a burst of more than 12 photons in 0.5 ms leaves a detector dead for more than 25% of the time. A range of other factors contribute to dead time, the most significant of which is the occurrence of very large event (VLE) pulses. When high-energy charged particles pass through the detector, they saturate the analog electronics, requiring that all events be inhibited for a period of time while it recovers. In the case of the test data used here, the VLE inhibit window was set at 550 μs. The VLE events occur at random and their rate is much higher in orbit than on the ground. The dead time of the PCA is quite a complicated topic, but it has already been well studied in terms of its effect on power density spectra when integrating on long data sets (Zhang et al. 1995). The effect of the dead time on intensity, or on bin count distributions, has been less well quantified. Extensive simulations are underway to reconcile a detailed model of the gas proportional counter and analog/digital electronics with mission data and laboratory tests on the flight spare detector. We do not propose to discuss dead-time corrections in detail in this paper.

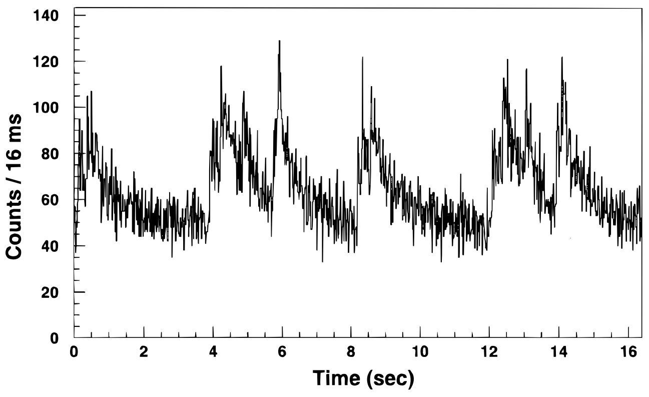

Fig. 2.—Typical segment of data showing the shot noise form of the intensity variations expected from Cyg X-1. The plot spans 16.4 s with the time axis having a resolution of 16 ms.

A comparison of the bin count distributions between the real ground data and the simulated data, e.g., for 0.5 ms bins, shows good agreement for the lower count bins. The rarely occurring largest bin counts cannot be compared well since there are so few of them. It is this part of the distribution that the BES method is intended to locate. A simple comparison is also only possible for constant mean rates, but the BES method is independent of source intensity.

5.3. *Simulated Cyg X-1 Data*

The intensity variations of Cyg X-1 have been characterized for many years in terms of a shot noise model that was first proposed by Terrell (1972). Suitable shot noise parameters have been used in a computer program, written in C, that simulates typical Cyg X-1 intensity fluctuations. A sample of the simulated Cyg X-1 light curve is given in Figure 2, and this appears somewhat similar in form to that of the earlier simulation work of Weisskopf & Sutherland (1978).

TABLE 12

Result Array from Cyg X-1 Simulation Data Similar to that Illustrated in Figure 2

Bin Size (ms)	Expectation Values								
	0.04	0.02	0.01	0.005	0.002	0.001	0.0005	0.0002	0.0001
0.5	9	1	1	1	1	0	0	0	0
1.0	1	6	0	1	1	0	0	0	0
2.0	5	4	0	3	0	0	0	0	0
4.0	7	7	8	3	1	1	1	1	0
8.0	34	24	11	8	8	4	5	1	8
16.0	71	51	15	4	3	14	23	0	0

Note.—The result array for a simulated observation of Cyg X-1 with background rates and shot noise characteristics similar to those expected during an *RXTE* PCA observation of the source. This table is a summation over 65.5 s of data. A typical short data segment is shown in Fig. 2.

The output of this simulation program can be used to generate the bin count stream for input to the new BES analysis method. Intensity variations like those shown in Figure 2 produce the result array output shown in Table 12 after a summation of just over 1 minute of data. There are more features than for a constant random source, but not as many as might be expected given the intensity variations evident in Figure 2 or the earlier comments of Weisskopf & Sutherland (1978). The numbers that appear in the result array are very sensitive to variations in the shot parameters used to generate the simulated light curve. Small changes can cause quite dramatic effects due to the various fast-changing threshold effects associated with the Poisson probability process. The simulation program will allow detailed modeling of future PCA data on Cyg X-1 once an actual observation has been made and the shot parameters have been characterized. The bias of the shot pulses on the BES method can then be estimated and removed in order to identify any millisecond bursts or similar short timescale features that may be present.

6. *RXTE* ELECTRONIC DATA SYSTEM APPLICATION

The NASA *RXTE* has several experiments on board (Bradt, Rothschild, & Swank 1993; Swank et al. 1995; Giles et al. 1995). The PCA experiment from the Goddard Space Flight Center is designed to surpass the performance of earlier experiments in all key parameters, namely detector area, time resolution, telemetry capacity, and also in respect of operational flexibility. On a second-by-second basis, the PCA will arguably be the first experiment to exceed the performance of a rocket flight carried out in 1976 (Giles 1981). That large area experiment (4000 cm^2) required sophisticated data management (Giles & Whitford 1980), and the PCA, with its much greater area of 7500 cm^2, also requires powerful processing capability. This is provided by an Electronic Data System (EDS) built at the Massachusetts Institute of Technology (MIT), which, although it contains many hard-coded operating modes, can have additional capabilities programmed from the ground after launch. Some details on the EDS can be found in Swank et al. (1995). A much more detailed description can be found in Chapter 7 of the Technical Appendix to the First *RXTE* NASA Research Announcement.

The EDS contains six Event Analyzers (EAs), two of which are committed to running standard mode 1 (1.22 kbits s^{-1}) and standard mode 2 (3.25 kbits s^{-1}). The other four EAs are available for use by guest observers, and there are a wide range of built-in data modes from which to choose to suit their scientific requirements. A BES mode could be assigned to one of these four EAs for observations of bright sources, where it would provide continuous burst monitoring statistics. Alternatively, it could operate as a sophisticated "burst trigger" option within an EA, providing a signal pulse to an additional EA that then automatically catches data for a time window encompassing the triggering event.

This paper has demonstrated that the BES method can be performed using an efficient algorithm and lookup-table approach that minimizes real-time calculation requirements, and that an implementation within the EDS on *RXTE* is very feasible. If the Poisson table has been computed and stored or uplinked from the ground, and the input to the analysis consists of "prebinned" data, then the processing only requires the control of address pointers and the use of addition, subtraction, and AND operations. Prebinning the data would also only require addition and subtraction if this were performed with software, as would be the case for *RXTE*.

Each basic binned input would result in approximately the following number of operations per second:

$$N_0 = \frac{2.0}{T_b}(N_e T_e + 6) , \qquad (2)$$

where T_b is the basic bin duration, N_e is the number of expectation values in the Poisson table, and T_e is the time required for a single comparison of the running sum with a Poisson table entry. The number 6 from equation (2) results from the operations on the running row sum, head pointer, and burst pointer.

Although the BES algorithms are not presently installed on the EDS, it is possible to load new EA programs from the ground, and a subset of the C routines developed here would provide most of the needed code. A BES system may be installed at a later stage in the *RXTE* mission, perhaps initially as a "piggyback" test exercise. The result array presented in Table 5 would require only 96 bytes of storage, so a telemetry rate of ~ 3 kbits s^{-1} would allow four result array integrations per second to be returned to the ground.

7. DISCUSSION

As noted by Giles (1981), any serious attempt to study millisecond or submillisecond rare and isolated bursts requires the following three criteria to be satisfied simultaneously: (1) a large detecting area, (2) fast low–dead-time electronics, and (3) high data-rate telemetry. It is clearly an advantage to have a long observing time and a low background, but the main requirement when searching for very rapid intensity variability is to detect the maximum possible number of photons per millisecond from the target source.

During the 1980s, *EXOSAT* and *Ginga* provided the type of instrumentation required, but both suffered from severe telemetry limitations that greatly reduced their potential for this type of study. Compared to the 1976 rocket flight of Giles (1981), the PCA experiment on *RXTE* offers nearly 2 times the detector area, much improved high-energy response, more spectral resolution, simultaneous bandwidth to 200 keV, and a much longer observing time. A typical dedicated *RXTE* high time resolution observation of a bright source such as Cyg X-1 could run continuously for 30 minutes at a greater than 256 kbits s^{-1} data rate. A single observation like this would log ~ 13 times the photons that Giles (1981) might have observed, or ~ 185 times those obtained by Rothschild et al. (1974) in their first flight. The crucial point is that the photons detected per millisecond would be significantly greater. The chance expectation of the millisecond bursts reported by Rothschild et al. (1974, 1977) and Giles (1981), which were of order 0.01, would be extremely small ($< 10^{-9}$) when scaled up to the PCA detector area. The BES method has been devised to enable a large quantity of data to be quickly quantified and searched for millisecond burst type features. Experiments with simulated data suggest that the BES method can provide a useful characterization of rapid variability from celestial X-ray sources. Efforts in this challenging observational field declined as the emphasis in X-ray astronomy shifted from mechanically collimated detectors to imaging telescope systems. The PCA experiment on

RXTE provides for the first time an opportunity to study these timescales in a significant and detailed way.

The PCA experiment has five separate detectors, and the high telemetry rate available with *RXTE* allows detector identification information to be retained. This is particularly important in distinguishing real millisecond burst type events from any spurious features, such as the short spike noted in the long sequence of ground test data. Any event seen in only one PCU would have to be rejected. Following the successful launch of *RXTE*, many observations of Cyg X-1 have been made, including several specifically intended to confirm and study the millisecond burst type features discussed in this paper. Preliminary analysis of part of these data suggests that isolated individual bursts of the size expected from the earlier results are not present in the data (Giles, Jahoda, & Strohmayer 1996). Further studies to quantify the dead time and shot noise effects are in progress.

8. SUMMARY

This paper has reviewed some of the statistical problems associated with the detection of rapid intensity changes, such as millisecond bursts, from black hole candidates. A new and efficient BES method is used to search for evidence of the existence of rapid infrequent bursts on a variety of short timescales. This method may be applicable to the analysis of many types of scientific time series data but is illustrated here using simulated Cyg X-1 data and test data from the PCA experiment ground calibration activities.

The BES method defines and uses a group of interrelated arrays: Poisson, working, result, normalization, and burst excess. It uses a lookup table and algorithms that are both flexible and computationally efficient for examining large data volumes. Once regions containing potentially interesting features are identified, these sections of data can then be examined in greater detail.

The BES method is suitable for real-time, on-board analysis, having only a low bit rate telemetry requirement. It can be used to trigger the capture of detailed data encompassing an identified feature or to indicate that a source should be specifically reobserved with a suitable fast-sampling, high-capacity data mode.

The BES method appears to offer a fast and efficient way of characterizing infrequent rapid intensity variability in X-ray sources, and its usefulness will be investigated using data from the PCA experiment on board *RXTE*.

We thank Will Zhang of the Goddard Space Flight Center for providing some suitable archival PCA ground test data and Ed Morgan of the MIT Center for Space Research for information on the possibilities of adding new modes to the EDS system on *RXTE*. We also thank the referee for numerous comments that helped clarify the text.

REFERENCES

Belloni, T., & Hasinger., G. 1990, A&A, 227, L36
Bradt, H. V., Rothschild., R. E., & Swank, J. H. 1993, A&AS, 97, 355
Gehrels, N. 1986, ApJ, 303, 336
Giles, A. B. 1978, Ph.D. thesis, Leicester Univ.
———. 1981, MNRAS, 195, 721
Giles, A. B., Jahoda. K., & Strohmayer, T. 1996, Adv. Space Res., submitted
Giles, A. B., Jahoda, K., Swank, J. H., & Zhang, W. 1995, Publ. Astron. Soc. Aust., 12, 219
Giles, A. B., & Whitford, C. H. 1980, IEEE Aerosp. & Electron. Sys., 16, 288
Meekins, J. F., Wood, K. S., Hedler, R. L., Byram, E. T., Yentis, D. J., Chubb, T. A., & Friedman, H. 1984, ApJ, 278, 288
Miyamoto, S., & Kitamoto, S. 1989, Nature, 342, 773
Negoro, H., Miyamoto, S., & Kitamoto, S. 1991, Frontiers of X-Ray Astronomy (Tokyo: Universal Acad. Press), 313
Ogawara, Y., Doi, K., Matsuoka, M., Miyamoto, S., & Oda., M. 1977, Institute of Space and Astronautical Science, RN, 26
Press, W. H., & Schechter, P. 1974, ApJ, 168, 437
Press, W. H., Teukolosky, S. A., Vetterling, W. H., & Flannery, B. P. 1992, Numerical Recipes (2d ed.; Cambridge: Cambridge Univ. Press)
Rothschild, R. E., Boldt, E. A., Holt, S. S., & Serlemitsos, P. J. 1974, ApJ, 189, L13
———. 1977, ApJ, 213, 818
Sutherland, P. G., Weisskopf, M. C., & Kahn, S. M. 1978, ApJ, 219, 1029
Swank, J. H., et al. 1995, in NATO ASI Ser., Lives of Neutron Stars, ed. M. A. Alpar, Ü. Kiziloŭglu, & J. van Paradijs (Boston: Kluwer), 525
Terrell, N. J. 1972, ApJ, 174, L35
van der Klis, M., Jansen, F., van Paradijs, J., Lewin, W. H. G., van den Heuval, E. P. J., Trumper, J. E., & Sztajno, M. 1985, Nature, 316, 225
Vaughan, B. A., et al. 1994, ApJ, 435, 362
Weisskopf, M. C., & Sutherland, P. G. 1978, ApJ, 221, 228
Wood, K. S., et al. 1991, ApJ, 379, 295
Zhang, W., Giles, A. B., Jahoda, K., Soong, Y., Swank, J. H., & Morgan, E. H. 1993, in Proc. SPIE, 2006, 324
Zhang, W., Jahoda, K., Swank, J. H., Morgan, E. H., & Giles, A. B. 1995, ApJ, 449, 930