Fast multi-tau real-time software correlator for dynamic light scattering

Davide Magatti and Fabio Ferri

We present a PC-based multi-tau software correlator suitable for processing dynamic light-scattering data. The correlator is based on a simple algorithm that was developed with the graphical programming language LabVIEW, according to which the incoming data are processed on line without any storage on the hard disk. By use of a standard photon-counting unit, a National Instruments Model 6602-PCI timer–counter, and a 550-MHz Pentium III personal computer, correlation functions can be worked out in full real-time over time scales of \( \approx 5 \mu s \) and in batch processing down to time scales of \( \approx 300 \) ns. The latter limit is imposed by the speed of data transfer between the counter and the PC's memory and thus is prone to be progressively reduced with future technological development. Testing of the correlator and evaluation of its performances were carried out by use of dilute solutions of calibrated polystyrene spheres. Our results indicate that the correlation functions are determined with such precision that the corresponding particle diameters can be recovered to within an accuracy of a few percent rms. © 2001 Optical Society of America

1. Introduction

Dynamic light scattering (DLS) is a well-established technique that has been used for decades in many fields of basic and applied science, from physics to chemistry, biology, and medicine. DLS measures the time-averaged correlation function of the fluctuations of the intensity scattered by the investigated sample and provides information on the decay or the relaxation time (or times) that characterizes its underlying dynamics. Examples of the applications of DLS are countless, the most prominent ones probably being laser Doppler velocimetry and sizing of submicrometer particles through the measurement of the translational diffusion coefficient associated with particle Brownian motion. In particular, the latter technique has experienced continual growth over recent years and is now routinely used in many laboratories worldwide, with applications ranging from industrial production control to the fundamental study of interacting-particle systems. Reviews of the historical applications of DLS as well as of its recent developments and experimental research can be found in Refs. 1–5 and references therein.

In a DLS experiment the signal-correlation function is usually carried out by means of a digital correlator, i.e., a device capable of performing on line digital signal processing (DSP) of the stream of count pulses coming from a photodetector, usually a photomultiplier. Given the intrinsic digital nature of the input signal, a digital correlator appears, therefore, to be the ideal instrument for tackling such a task, free of noise and other distortion effects that are unavoidable in any analog-type analysis. Digital correlators were first developed in the 1970s by Pike and colleagues,6,7 and ever since they have been continually developed and improved. But it was in the late 1980s that their growth started to increase exponentially, thanks to the work of Schatzel and co-workers8–12 who invented the so-called multi-tau correlator. With the help of the multi-tau scheme it became possible to access, within the same correlation function, a huge dynamic range of time scales that extend from tens of nanoseconds to hours. Since then DLS has become the ideal tool for studying the dynamics of many interesting new systems that were not previously accessible because they were characterized by a large polydispersity of diffusion coefficients, such as colloidal or polymeric gels, complex fluids, foams, and granular materials.
Many commercial digital correlators are available nowadays on the market, the most popular ones probably being those manufactured by Laser Vertriebsgesellschaft mbH (ALV) (Langen, Germany), by Brookhaven Instruments (Holtsville, New York), and, in more recent years, by Correlator.com (Bridge- water, New Jersey). These correlators are powerful DSP devices that are based on a multi-tau scheme of the sampling times and are capable of computing correlation functions in full real time across a huge range of different lag times from \( \sim 10 \) ns to hours. They are easily installed on a personal computer (PC), are run by user-friendly software, and are provided with nontrivial software libraries for data analysis. However, they have several limitations or drawbacks because they are designed and manufactured for carrying out the whole (but only) task of measuring the correlation function of a digital signal. They are not very flexible (for example, for some of them the delay-time grid and the number of bits per channel are fixed), are rather expensive, and, above all, cannot be implemented easily with the advent of new available technologies.

In this paper, we propose an alternative approach that is based on the use of commercially available, comparatively inexpensive, general-purpose electronic devices only and on the development of a software algorithm that was written with a modern, inherently multitasking, graphical programming language. For this purpose we used LabVIEW\(^{13}\) (National Instruments), which has become one of the most popular and powerful programming tools for interactive data acquisition and DSP. At the same time the stream of count pulses from the photomultiplier were counted with a fast standard counter whose output was a 32-bit integrated count. Probably the main feature of the counter (and of its LabVIEW drivers) is the possibility of storing its 32-bit output data in a so-called double buffer from which the numbers can be transferred in blocks to the PC’s memory, asynchronously with the sampling rate. This decoupling between sampling and processing, together with the huge size of this buffer (limited by only the RAM memory available in the host PC), offers great flexibility and allows the data analysis to be optimized easily for high-speed performance.

But clearly the main limitation of a PC-based software correlator still remains speed because the time required for processing each datum is expected to be much higher than the corresponding time required by a hardware correlator (\( \sim 10 \) ns). This fact limits the minimum lag time attainable in the correlation function because full real-time operation is possible only when the (average) processing time per point is less than the sampling time of the data. If one forgoes real-time performance, higher sampling rates are possible, but batch processing is required.\(^{12}\) In other words, the data are sampled and processed in batches with a reduction of the correlator efficiency or duty cycle (the ratio of the effective measuring to the elapsed time) that is approximately equal to the ratio between the sampling and the processing times.

In this research, we tackled the speed problem by designing a correlator that is based on a master–slave scheme in which two correlators, one slow and one fast, work in parallel to process the same stream of count pulses that are sampled with two very different gate times. The slow correlator operates in full real-time and behaves as the master and triggers the operation of the fast correlator (the slave), which consequently operates as a batch processor. Both correlators adopt a flexible multi-tau scheme of lag times in which the sampling times of the two correlators are programmable by means of software and the sequence of incoming data is grouped (or binned) in blocks of \( m \) points, with \( m \) an integer that we call the binning ratio. Then the binned data are fed to a set \( S \) of linear correlators that compute the correlation function on \( P \) evenly spaced lag times. Clearly, the correlator performance, such as the accuracy of the measured correlation function, the processing time per point, the minimum and the maximum lag times, depend on the parameters \( m, P, \) and \( S \) and on the two fast and slow sampling times as well. By tuning these parameters properly, one can easily optimize the correlator functioning, and this is a useful flexibility that is missing in most of the current hardware correlators.

This paper is organized as follows: In Section 2, we recall the basic concepts of DLS, and, in particular, we discuss triangular averaging, which determines the ultimate accuracy attainable in a correlation function. Section 3 is devoted to illustrating the multi-tau scheme adopted in the correlator and provides estimates of the processing time per point. Section 4 describes the correlator architecture and the master–slave scheme used in our design. The experimental data, taken on solutions of polystyrene spheres, are reported in Section 5 and are used to ascertain the correlator performance. Finally, in Section 6 a summary and further developments of this research are presented and discussed.

2. Theory

In a DLS experiment the intensity-correlation function of the scattered light is obtained by the measurement of the correlation function \( G_s(\tau) \) of the pulses streaming out from a photomultiplier\(^2\)

\[
G_s(\tau) = \langle n(t)n(t + \tau) \rangle = \lim_{T \to \infty} \frac{1}{T} \int_0^T n(t)n(t + \tau)dt, \tag{1}
\]

where \( \tau \) is the delay or the lag time and \( n(t) \) is the photon-counting rate. Equation (1) is more commonly represented in its normalized form of

\[
g_n(\tau) = \frac{\langle n(t)n(t + \tau) \rangle}{\langle n(t) \rangle^2}, \tag{2}
\]

for which \( g_n(\infty) = 1 \) and \( g_n(0) = 1 + \frac{\sigma_n^2}{\langle n \rangle^2} \), where \( \langle n \rangle \) is the average count rate, and \( \sigma_n \) is the standard deviation. For a stochastic Gaussian process\(^{14}\) in
which the scattered light is detected within a single coherence area, $\sigma_n = \langle n \rangle$, and, consequently, the amplitude of the correlation function is $\beta = g_n(0) - g_n(\infty) = 1$. However, in many experimental situations more than one coherence area is collected, leading to a damping of the fluctuations and a consequent reduction of that amplitude $\beta$ to values $< 1$.

Equations (1) and (2) are based on the assumption that the time resolution used for counting the pulse stream is infinite. However, in a real experiment the pulse stream is always integrated over a finite gate or sampling time $\Delta t$ (see Fig. 1). Consequently, we must replace $n(t)$ as it appears in Eqs. (1) and (2) with the average count rate $\mu(t)$, given by

$$\mu(t) = \frac{1}{\Delta t} \int_{t-\Delta t/2}^{t+\Delta t/2} n(t')dt' = n(t) * \text{rect}(t/\Delta t), \quad (3)$$

where $\text{rect}(x)$ is the rectangular function that is equal to 1 for $|x| \leq 0.5$ and zero elsewhere and the asterisk denotes a convolution product. Thus the measured correlation function

$$G_\mu(\tau) = \langle \mu(t) \mu(t + \tau) \rangle \quad (4)$$

depends on $\Delta t$, and, because $\langle \mu \rangle = \langle n \rangle$, it is straightforward to show that its normalized version can be written as

$$g_\mu(\tau) = g_n(\tau) = \frac{\Lambda(\tau/\Delta t)}{\Delta t}, \quad (5)$$

where $g_n(\tau)$ is given by Eq. (2) and $\Lambda(x)$ is the triangular function, defined as $\Lambda(x) = 1 - |x|$ on the support $|x| \leq 1$ and zero elsewhere. Equation (5) shows that $g_n(\tau)$ is a smoothed version of the true correlation function $g_n(\tau)$ and represents a good approximation of it, provided that $g_n(\tau)$ does not vary significantly over a time scale that is comparable with $\Delta t$. Note that, in the limit of $\Delta t \to 0$, the convolution function tends to the delta function and therefore $g_\mu(\tau) \to g_n(\tau)$.

To give an estimate of the approximation introduced by Eq. (5), which is often called triangular averaging, we examine the case of a single-exponential decay

$$g_n(\tau) = 1 + \beta \exp(-2\tau/t_0), \quad (6)$$

which is characterized by an amplitude $\beta$ and a decay rate $t_0/2$. By the insertion of Eq. (6) into Eq. (5) it is easy to show that, for $\tau \approx \Delta t$, the measured correlation function results in

$$g_\mu(\tau) = 1 + \beta \exp(-2\tau/t_0) \left[ \frac{\sinh(\Delta t/t_0)}{\Delta t/t_0} \right]^2,$$

$$\tau \approx \Delta t. \quad (7)$$

It should be stressed that Eq. (7) is valid for only $\tau \approx \Delta t$. When $\tau < \Delta t$ the result is significantly different (see, for example, Ref. 15) but is not considered here because in a multi-tau correlator we always have $\tau \approx \Delta t$. A comparison of Eqs. (6) and (7) shows that the error $\delta g_\mu$ is

$$\delta g_\mu(\tau) = g_\mu(\tau) - g_n(\tau)$$

$$= \beta \exp(-2\tau/t_0) \left[ \frac{\sinh(\Delta t/t_0)}{\Delta t/t_0} \right]^2 - 1), \quad (8)$$

where $\delta g_\mu$ as a function of the ratio $\Delta t/t_0$ for different values of the ratio $\alpha = \tau/\Delta t$ is shown in Fig. 2 for the case of $\beta = 1$. All the curves exhibit a maximum for the ratio $\Delta t/t_0$ in the range of approximately 0.1–1 and decay to zero for $\Delta t/t_0 \ll 1$ and $\Delta t/t_0 \gg 1$.

As was mentioned above, the errors are maximum for the curve with $\alpha = 1$ and decrease with increasing $\alpha$. In particular, if an accuracy of $10^{-5}$ is wanted the value of $\alpha$ must be $\geq 7$, as is shown by the solid curve
that corresponds to $\alpha = 7$. This is the level of accuracy according to which our correlator was designed (see Section 3). It should be pointed out that, although the curves of Fig. 2 were worked out for the specific case of a single-exponential decay with a decay time $\tau_0$, the conclusion drawn above is quite general. Indeed, suppose that more decay times are simultaneously present in the correlation function, i.e., that

$$g_s(\tau) = 1 + \sum_i a_i \exp(-2\tau/t_i),$$

in which $\sum a_i = 1$, because $g_s(0) = 2$. Thus, by use of Eq. (5) the error $\Delta g_s$ becomes a sum of different terms, each one weighted by $a_i$ and equal to expression (8) but with $t_i$ instead of $\tau_0$. If we now plot $\Delta g_s$ against $\Delta \tau$, we obtain curves whose maxima are always lower (for the same $\alpha$) than the maximum of the curves shown in Fig. 2 (this is because each decay time gives a contribution that is centered at different values of $\Delta \tau$). Thus, in conclusion, regardless of the fact that $g_s(\tau)$ may be characterized by one or more decay times, triangular averaging introduces systematic deviations that can always be upper bounded, in accord with the curves of Fig. 2. Moreover, these curves were drawn for the case of $\beta = 1$, and thus the deviations they represent have to be rescaled by amplitude $\beta$ when this is less than 1. Finally, it is worth mentioning that a systematic error of the order of $10^{-3}$ is rather small and can easily be masked by shot noise if the measuring time, the count rate, or both are not high enough.

### 3. Multi-Tau Scheme

A multi-tau correlator is based on the simple consideration that is given at the end of the Section 2: As long as the lag time is much longer than the sampling time, there is no point in sampling the signal over time scales that are orders of magnitude less than the lag time. Thus before computing the correlation function at a lag time $\tau$ the signal is averaged over an integration time $\Delta t$ that is chosen so that the ratio $\alpha = \tau/\Delta t$ is higher than a given value. This choice ensures that the correlation function is computed with the desired accuracy (see Fig. 2) and sensibly reduces the number of operations to be carried out when long lag times are considered. Ideally, a multi-tau correlator would have the ratio $\alpha$ equal for all lag times, but, in practice, many of the hardware correlators that are commercially available are realized by the division of the channels of the correlation function into blocks (typically made up of eight channels) and by doubling the integration time that corresponds to each block. In this way lag times that span many decades (from tens of nanoseconds to hours) can be covered by use of only a few hundred channels.

Our multi-tau correlator is designed in a similar way but is more flexible than hardware correlators. It is based on a set $S$ of linear correlators whose integration times $\Delta t_s$ increase in a geometric progression given by

$$\Delta t_s = m^s \Delta t_0, \quad s = 0, 1, 2, \ldots, S - 1, \quad (9)$$

where $\Delta t_0$ is the integration time of the fastest linear correlator and is equal to the gate time of the counter. The integer $m$ represents the binning ratio as given by the ratio between the sampling times of two adjacent linear correlators. Thus, if $N_0$ denotes the number of data points fed to the zeroth correlator, the number of points $N_s$ handled by the $s$th correlator is

$$N_s = m^{-s} N_0, \quad s = 0, 1, 2, \ldots, S - 1. \quad (10)$$

Each correlator computes the correlation function of the pulse stream [averaged over the gate time $\Delta t_s$, see Eq. (3)] on a set of equally spaced lag times $\tau_s(p) = p\Delta t_s$, with $p = 0, 1, \ldots, p - 1$, where $P$ is the size of an array called shift register that is equal for all the linear correlators. The lag-time overlap between two adjacent correlators $s$ and $s + 1$ corresponds to the first $P/m$ points of the $(s + 1)$th correlator, as sketched in Fig. 3 for the particular cases of $m = 3$ and $P = 9$. Thus the first $P/m$ points of each linear correlator can be discarded (without computing the correlation function) and the lowest lag time for each correlator remains

$$\tau_s(\min) = (P/m)\Delta t_s. \quad (11)$$

Because we require that the ratio between the minimum lag time and the sampling time must be larger than the factor $\alpha$, $P/m$ is constrained to the value $P/m = \alpha$. The actual values of $P$ and $m$ can be
chosen in accord with the speed performance, which can vary depending on experimental conditions. Incidentally, we note that most of the hardware correlators work with values of $P = 16$ and $m = 2$; thus each linear correlator uses eight different lag times, and half of the points are discarded.

The software algorithm used for computing the correlation function of each linear correlator is quite simple and resembles the scheme used in hardware correlators. After an initial transient in which the shift register is loaded with the first $P$ points, the stream of incoming data is processed one by one in the following way: First, the last (most recent) datum is used for updating the shift register, which means that all the components of the shift register are moved to the right (see Fig. 3), the last component is discarded, and the first component (with a lag time of $\tau = 0$) is replaced with the last datum. Then all the components are multiplied by the last datum, and the results are summed up into an array, which represents the correlation function. Although, in principle, all the incoming data could be treated as integer numbers, all the operations that involve the shift register were carried out with single-precision accuracy. This approach was used because we checked that there was no significant gain in reducing the processing time when integers were compared with single-precision numbers. We normalized the correlation function by following either Eq. (2) or dividing for $G_0(t \to \infty)$ or using the symmetrical-normalization procedure proposed by Schatzel et al.\textsuperscript{10} In any case to save time the normalization as well as the other data reductions, such as merging all the linear correlators, discarding the overlapped data, and data plotting, were carried out only once in a while (approximately every 5–10 s) and at the end of the measurement.

A qualitative estimate of the time required for computing the overall correlation function can be given as follows: Let us start by considering that each linear correlator is fed with $N_s$ points [see Eq. (10)] and has to handle a shift register with $P$ components. The time necessary for carrying out this task can be divided into two parts. The first one is due to the operations that involve only the number of points (averaging, sums, updating the first component of the shift register, etc.) and therefore scale as approximately $N_s$. The second one is due to the number of operations carried out with the shift register (rotating the shift register, multiplications and sums of its components, etc.) and therefore scales as approximately $N_s P$. Here we neglect the $P/m$ points to be discarded, for which the correlation function does not need to be computed. This approach is valid because we checked that it is faster to compute the correlation function over the entire shift register (and to discard the $P/m$ points at the end of the process) rather than split the shift register into two subarrays and process only one of them. The time saved in carrying out the above operations on an array of reduced size [the size is reduced by a factor of $(m - 1)/m$] does not compensate for the effort of handling the array itself, at least for arrays of small sizes, as with the shift registers of a standard multi-tau correlator (approximately ≤50 components).

If we indicate with $w_1$ and $w_2$ the weights that are associated with the operations that scale as $N_s$ and $N_s P$, respectively, and sum up over the $S$ stages, with $N_s$ given by Eq. (10), we find that the overall computation time $t_{comp}$ scales as

$$t_{comp} \sim w_1 \sum_{i=0}^{s} N_s + w_2 \sum_{i=0}^{s} N_s P$$

$$= N_s (w_1 + w_2 P) \frac{1 - m^{-s}}{1 - m^{-1}}$$

$$\sim N_s (w_1 + w_2 P) \frac{m}{m - 1}, \quad m > 1,$$

(12)

in which the last approximation is valid because usually $S \gg 1$. Expression (12) shows that $t_{comp}$ is independent of $S$ and is determined mostly by the first ($s = 0$) correlator. The remaining stages contribute for only a factor of $m/(m - 1)$, which is of the order of unity for $m \gg 1$. If we now insert into expression (12) the constraint that $P = \alpha m$, we obtain

$$t_{comp} \sim N_s (w_1 + w_2 \alpha m) \frac{m}{m - 1},$$

(13)

which exhibits a minimum for

$$m_{min} = 1 + \left( \frac{1 + w_1/w_2}{\alpha} \right)^{1/2}. \quad (14)$$

It is interesting to note that, when the time required for the multiplications is predominant ($w_2 \gg w_1$) for very long shift registers ($\alpha \gg 1$), we have $m_{min} \sim 2$, which corresponds to the value used in many hardware correlators.

However, in our software correlator the situation is quite different because the number of operations necessary for handling the data, besides the multiplications carried out with the shift register, is fairly high. Because it is almost impossible to estimate the relative weight $w_1/w_2$, we directly measured the computational time required for the program to build a correlation function with given values of $P$, $m$, and $S$. The result is shown in Fig. 4 in which the computational time per point $t_{comp}/N_0$ is reported as a function of $m$ for different values of the ratio $\alpha = P/m$ (solid symbols). The test was carried out on a given set of simulated data that was generated in accordance with Ref. 16 (and stored on the hard disk) so that correlation functions with different values for parameters $P$ and $m$ were obtained by processing the exact same data. The value of $S$ was chosen to have, for each pair of $P$ and $m$, approximately the same maximum lag time (~2 s). To avoid unreliable results caused by interference from the operative system under which the host PC works (Windows '98) required that no other applications be active at the time of the test. Figure 4 shows unambiguously that the curves (solid symbols) with a higher $P/m$
require a higher $t_{\text{comp}}/N_0$ and that their minima clearly fall at values of $m$ that are larger than 2. In particular, the curve representing $P/m = 7$ (which corresponds to the value used in this study) exhibits its minimum for $m \sim 4$; the corresponding computational time per point is $\sim 5$ $\mu$s. For comparison, we also reported in Fig. 4 the computational time per point for obtaining, from the original counts at the fastest sampling time $\Delta t_{\text{fs}}$, the average counts for all the other gate times $\Delta t_s$. This is shown by the data (open symbols) that are plotted in the bottom of the graph and indicates that this time is independent of $P$ and is of the order of 0.5–1 $\mu$s, i.e., approximately 10% of the overall computational time. Its dependence on $m$ is rather mild, and after an initial decay takes place for the lower values of $m$ it levels out to a constant value of $\sim 0.5$ $\mu$s, consistent with what is predicted by expression (12) when $w_2 = 0$.

4. Correlator Architecture

The correlator design was based on a simple algorithm written in LabVIEW, a graphical programming language (National Instruments) that is particularly suited to interactive data acquisition and DSP. LabVIEW is based on a data-flow programming model in which the execution order is not determined by sequential lines of text (as in traditional text-based languages) but rather is decided by the flow of data between block diagrams. Thus many diagrams can be executed in parallel within a single program, and, as matter of fact, LabVIEW behaves as a multitasking system with a high level of modularity and hierarchy. Moreover, LabVIEW can easily address almost the entire RAM available on the PC and, consequently, operations on large arrays or matrices can be carried out at high speeds.

The only hardware used by the correlator are a detector and a counter. The light detector used in our setup was a photon-counting unit (Hamamatsu Model H6180-01) whose output is a stream of transistor–transistor logic pulses, with a pulse width of 9 ns and a maximum operation rate of 30 MHz. Its dark count at room temperature is $\sim 10$ counts/s. The pulse stream was counted by a 32-bit counter–timer board (National Instruments Model PCI-6022) equipped with eight input channels and has a maximum input rate of 80 MHz. The minimum source–pulse duration is 5 ns, and the minimum gate-time period is 10 ns. The first-in–first-out (FIFO) buffer for the direct memory access (DMA) transfer of the data to the host PC is a $16 \times 32$ bit buffer. The rate at which data are transferred depends on both the PC specifications and the operation mode of the counter, but is ultimately limited by the maximum speed of the peripheral–component interconnect (PCI) bus ($\sim 33$ MHz). In our system this was of the order of $\sim 200$ ns and was due mainly to the rather shallow FIFO buffer of the counter board.

The counter can be set to work in the so-called double-buffer (or circular-buffer) acquisition mode in which the counter counts continuously and stores the data in a data buffer whose maximum size is limited by the memory that is available on the host PC. Because each datum is a 32-bit number, for a PC with 256-Mbyte RAM this maximum size has to be somewhat smaller than $6.4 \times 10^9$. From the data buffer the data can be read in blocks and transferred to the LabVIEW program by means of a reading buffer whose size can be chosen at will but that obviously must be smaller than the data-buffer size.

The time required for transferring these data depends on the host PC and for a 550-MHz Pentium III is of the order of 0.1 $\mu$s/point, a time much shorter than typical processing time per point (see Fig. 4). The reading–writing procedure from and into the double data buffer takes places asynchronously, and as the old data are retrieved at one location in the buffer, the new data are stored in a different location so that still-unread data are not overwritten by the newer data. As long as the backlog between the written and the read data is less than the double-buffer size, data handshaking can continue. Of course, the procedure can keep going indefinitely only if the average rate at which the data are written is equal to or less than the average reading rate.

The output of the counter is an integrated count, i.e., the number of pulses counted from the beginning of the count up to the current time. This requirement offers the advantage that the average count corresponding to an integration time $\Delta t_s$ that is longer than the sampling time is carried out with a simple difference between two integrated counts. It should be noted that the integrated count is limited to 32 bits, i.e., $\sim 4.2 \times 10^9$. Thus if an average count rate of 1–10 MHz is used, the counter goes into overflow after approximately 430–4300 s, a time that is long enough for typical DLS measurements (if longer...
times are needed, the overflow can easily be taken into account.

As was reported in Section 3, the minimum computational time for our system under the scheme \( P = 28, m = 4 \), and \( S = 8 \) is approximately \( t_{\text{comp}} \approx 5 \mu s/\text{point} \). Thus the correlator can work in real time to gate times as high as \( \Delta t_0 \approx 5 \mu s \). To achieve shorter times, we implemented a second fast correlator, which works in parallel with the first slow correlator, as shown schematically in Fig. 5. The slow correlator processes the data that arrive from a counter operating with a long gate time, whereas the fast correlator processes the same data acquired by use of a counter operating at a much higher sampling rate. The slow correlator works continually, whereas the fast one works only when there is some spare time left over from the slow correlator. In this scheme the slow correlator is the master correlator that works in real time and triggers the operation of the slave correlator, which works as a batch processor, i.e., intermittently, asynchronously with the master, and, obviously, not in real time. The slave correlator is set to work in a way similar to that of the master, i.e., with a multi-tau scheme of sampling times and an adjustable reading buffer, but with a single buffer rather than a double buffer.

On a software trigger (called occurrence in LabVIEW language) from the master correlator, the slave correlator acquires data, stores them in its data buffer, and processes the first reading buffer. Then on the next occurrence it processes another reading buffer, and so on for each occurrence, until all the data present in its data buffer are processed. At the next occurrence it starts again from the beginning, acquiring new data. The occurrence from the master correlator arrives whenever, at the end of the processing of each reading buffer, the backlog between writing and reading is less than a given fraction (typically \( \sim 50\% \)) of the double-buffer size. Clearly, the duty cycle of the slave correlator, i.e., the ratio between the effective measuring time and the elapsed time, depends on the settings of the two correlators and can be tuned with a certain freedom. The program keeps track of all these times so that it can take them into account for estimating the error bars that are associated with the channels of the correlation function (not implemented here).

In Table 1, we report two typical configurations under which our correlator can work. In both cases the parameters \( P \) and \( m \) were chosen so that a ratio of \( \alpha = \tau/\Delta t \geq P/m = 7 \) was obtained for all the lag times except the shortest ones that belong to the first stage (for which \( \tau \geq \Delta t_0 \)). Thus aside from these first lag times (in which shot noise is usually predominant) the systematic error that is due to triangular averaging is always \(< 10^{-3} \). The slow configuration, in which only the master correlator is active, works in full real time (with a duty cycle of 100\%) and has a minimum lag time of 5 \( \mu s \). Its maximum lag time is \( \sim 2 \) s but can be made much longer without affecting the correlator performance. Note that the data- and the reading-buffer sizes are quite large. This size is important to keep the computing time per point short because in LabVIEW many operations are carried out on a vectorial basis; consequently, the longer the array to be processed, the shorter the elaboration time per point. In the fast configuration, in which the slave correlator is also active, the master correlator works with longer gate times (\( \sim 12 \mu s \)) and has a much smaller reading buffer. These differences exist for two reasons: On the one hand, there should be some time left at the end of each reading buffer for triggering the fast correlator; on the other hand, to keep the duty cycle of the slave from being small means that this triggering has to occur frequently, and thus the reading buffer of the master has to be reasonably small. From the table one can see that, because the data acquisition of the slave occurs approximately every 24 ms (12 \( \mu s \times 2000 \)) and lasts for \( \sim 300 \mu s \) (300 \( \text{ns} \times 1000 \)), the duty cycle is \( \sim 1\% \), as reported. This value is rather small, but, in principle, at these short lag times a not very long measur-

---

**Fig. 5.** Schematic diagram of the correlator architecture. The pulse stream exiting the photon-counting unit is sampled at two different times and simultaneously fed to the slow and the fast correlators. The slow correlator operates in full real time and acts as the master correlator that triggers the operation of the slave correlator, the fast correlator, that thus operates as a batch processor.
ing time is necessary for accumulating good intensity statistics. However, this assumption is true for only high count rates, when the photon noise (the noise associated with the detection process) can be neglected with respect to the signal noise (the noise associated with the finite measuring time). Unfortunately, in many experimental situations, this relation is not true, and higher duty cycles should be used.

To increase the duty cycle, one could operate in two ways: (a) Further reduce the size of the reading buffer of the master and so increase the rate of occurrence sent to the slave. However, this approach would increase heavily the elaboration time per point of the master, soon reaching a value that would be too close to its gate time and leaving no spare time. (b) Increase the size of the reading buffer of the slave; however, because of the small FIFO buffer of the counter, this approach would cause DMA transfer failures with a subsequent loss of data. Obviously, the higher the rate at which the data are to be transferred to the PC memory, the smaller the buffer has to be, and, as a matter of fact, a data buffer of ~1000 is the maximum size allowable with a gate time of 300 ns.

Finally, it should be pointed out that a gate time of 300 ns for the slave represents almost an ultimate limit for our system. If we relax this requirement and set, for example, the gate time of the slave to 500 ns, the reading buffer of the slave can be increased to ~10^4, and its duty cycle grows immediately to ~10%.

5. Experimental Results

We tested the correlator on real samples by studying aqueous suspensions of calibrated polystyrene spheres. The particle samples were from Duke Scientific Corporation, Palo Alto, California, with diameters \( d \) between 19 and 107 nm. According to the manufacturer, the particle diameters were certified by means of DLS (\( d \leq 40 \) nm) and transmission electron microscopy (\( d \geq 50 \) nm), and their accuracy was of the order of 5% or less. All the samples were diluted in distilled water and filtered through a 0.22-\( \mu \)m Millipore membrane. Their concentrations were chosen to produce a count rate of \( \sim 10^5 \) counts/s at 90°. The scattering cell was a glass tube with a 15-mm inner diameter that was clamped onto the axis of a homemade goniometer (the angular resolution was \( \sim 0.1° \)). No index-matching vat was used, and the measurements were taken at room temperature (20 ± 1°C). The light source was a frequency-doubled 100-mW Nd:YAG laser emitting at 532 nm whose TEM\(_{00}\) beam was mildly focused into the cell with a waist of 2\( \nu_0 \) = 200 \( \mu \)m. The scattered light was detected by use of a standard optical fiber receiver (ALV, Langen, Germany) that was designed to be monomode for a wavelength of 633 nm. As a consequence, approximately two modes could propagate through the fiber, and the amplitude of the correlation function was \( \beta \sim 0.4 \).

Figure 6 shows a typical result obtained with our correlator working in the slow configuration, i.e., by setting the correlator parameters equal to the ones reported in the second column of Table 1. Thus the lag-time span was from 5 \( \mu \)s to \( \sim 2 \) s. The particle diameter was \( d_{\text{cent}} = 107 ± 7 \) nm; the scattering angle, 90°; and the measuring time, \( \sim 300 \) s. The data were normalized by use of the so-called symmetrical-normalization procedure proposed by Schatzel et al.\(^a\) and were fitted to the function

\[
g(\tau) = B + \beta \exp(-\Gamma \tau),
\]

where the baseline \( B \), the amplitude \( \beta \), and the decay rate \( \Gamma \) were the fitting parameters. The solid curve plotted in Fig. 6(a) shows the fitted curve, whereas the relative residuals between the data and the fit are shown in Fig. 6(b). The match is fairly good with almost nonsystematic deviations of the order of \( \sim 10^{-3} \) peak to peak.

As is known, the decay rate \( \Gamma \) depends on the scattering angle \( \theta \) and the particle-diffusion coefficient \( D \) through the relation\(^2\)

\[
\Gamma = 2Dq^2,
\]

where \( q \) is the magnitude of the scattering wave vector and \( q = (4\pi/\lambda)n \sin(\theta/2) \), with \( n \) as the refractive index of the medium and \( \lambda \) as the laser wavelength.

---

\(^a\)In the slow configuration, only the master is active and the correlator works in full real time (duty cycle of 100%) with a minimum lag time of 5 \( \mu \)s. In the fast configuration, both the master and the slave are active. The master works in full real time with a gate time of 12 \( \mu \)s, whereas the slave behaves as a batch processor with a duty cycle of \( \sim 1% \) and has a gate time of 300 ns.
In the case depicted in Fig. 7 the particle diameter and the measuring time results shown in Fig. 6, the scattering angle was 90°, and the diameter recovered from the fitting was \( d = 106.73 \pm 0.1 \) nm.

Thus by using Eq. (16) together with the classical Einstein–Stokes relation, we have

\[
D = \frac{kT}{6\pi\eta R},
\]

and it is possible to determine the particle radius \( R \).

\( k \), \( T \), and \( \eta \) are the Boltzmann constant, the absolute temperature, and the solvent viscosity, respectively. From the data of Fig. 6, we recovered a particle diameter of \( d = 106.7 \pm 0.1 \) nm, which is in excellent agreement with the expected certified value.

A typical result that was obtained with the correlator working in the fast configuration (see Table 1) is illustrated in Fig. 7. As for the results shown in Fig. 6, the scattering angle was 90°, and the measuring time \( t \) was approximately 300 s, but in the case depicted in Fig. 7 the particle diameter was \( d_{\text{cert}} = 30 \pm 1.3 \) nm. The data processed by the master correlator are represented by the circles, whereas those processed by the slave correlator are indicated by the squares. As is shown in the residual plot of Fig. 7(b), the data are fitted excellently to the single-exponential decay function of Eq. (15) with deviations that are quite different depending on whether the master or the slave correlator is considered. For the master, the situation is similar to the one shown in Fig. 6(b), and the amplitude of the deviations is of the order of \( 10^{-3} \). Conversely, the deviations of the slave are almost an order of magnitude higher, and this is clearly due to the shot noise's not being sufficiently averaged out. We recall that, in this configuration, the duty cycle of the slave is \( \sim 1\% \); thus the effective measuring time of the slave in this measurement was \( \sim 3 \) s. The bump at very short lag times (\( \leq 1 \) μs) is probably due to afterpulse effects of our phototube (we observed it also on still samples), and the corresponding data were not considered in the fit. The final result was an estimated diameter of \( d = 30.2 \pm 0.2 \) nm, in excellent agreement with the expected certified value.

To test the reproducibility and the reliability of the results shown in Figs. 6 and 7, we carried out several other measurements on different samples and at different scattering angles around 90° (60° \( \leq \theta \leq 120° \)). Except for the 107-nm particles, the correlator was operated in the fast configuration, and all other parameters were similar to those of Fig. 7. All the measured correlation functions were fitted accurately with Eq. (15) with relative residuals similar to those of Fig. 7(b). The overall results are summarized in Fig. 8 in which the ratio of the recovered \( d \) to the certified \( d_{\text{cert}} \) diameter is reported as a function of \( qd_{\text{cert}} \) for four different particle diameters. As shown, the data are spread around the expected value of 1 with deviations of the order of a few percent rms. This trend is an indication of the overall reliability of our measurements, which, however, were carried out with a homemade setup that must still be...
optimized for optical misalignments and stray-light rejection. Thus some of the deviations exhibited by the data of Fig. 8 are likely to be attributed to the apparatus, and the actual accuracy attainable with our software correlator is probably even better than that shown in the figure.

6. Conclusions

In this paper, we have shown that a PC-based multi-tau software correlator can be implemented successfully by use of commercially available electronic equipment, such as a standard photon-counting unit, a fast counter working on the PCI bus, and a PC. The correlator operates online without any storage on a hard disk and is capable of measuring correlation functions over time scales of \(-5\) μs in full real time and of \(-300\) ns with batch processing. The software algorithm was developed by use of LabVIEW, a modern graphical programming language that is a popular and powerful tool for interactive data acquisition and DSP. The proper functioning of the correlator was ascertained by use of dilute solutions of calibrated polystyrene spheres whose diameters were determined to within an accuracy of a few percent.

Compared with the hardware correlators that are available on the market (which can reach time scales of the order of 10 ns), our software correlator appears to be rather slow and not competitive. However, its speed limitations are set by the current performance of today’s electronic and PC technology. For example, the limit of 300 ns is imposed by the speed of DMA data transfer over the PCI bus between the PC’s memory and the (small) FIFO buffer on the board of the counter. Considering the incredible rate at which both the PC and data-acquisition technology are growing, we believe that this limit will be reduced in the near future and that the gap with hardware correlators will be progressively shortened. Moreover, hardware correlators are in many cases rather expensive and offer little flexibility because they return only the correlation function of the incoming data. Conversely, a software correlator is much more flexible and gives access to the entire detected photon sequence, allowing a more sophisticated data analysis to be carried out. For example, higher-order correlation functions can be computed, or the distribution of the time intervals between successive photon events can be retrieved. The latter type of analysis is particularly convenient when the photon-counting rate is less than the sampling rate, as was pointed out in Ref. 18. This approach could open up a way to overcome the speed limitations of our correlator and push its minimum time scale toward values at which the threshold is determined by only the time resolution of the photodetector or the counter.

Finally, it should be noted that, in this paper, we did not address the issue of the statistical noise associated with the different channels of the measured correlation function, also called correlation estimators. The estimate of the variances and the covariances associated with the correlation estimators is a rather difficult problem that becomes even more difficult for a multi-tau correlator in which two estimators at different lag times may be derived from data sampled with two different integration times. Thus models that include both random photon noise and correlated intensity noise are to be utilized, and the variance–covariance matrix is estimated by use of only the correlation estimators and the information on the average count rate and the measuring time.\(^{14,19,20}\) However, and also by consideration of the fact that sometimes the predictions of such models do not agree (as in the cases of Refs. 14 and 19), it would be nice to provide estimates of the covariant matrix independently of the adopted model and obtained directly from the analysis of the incoming photon sequence. In this respect a software correlator offers a great opportunity because it allows the measurements of higher-order correlation functions that can be related directly to the covariance matrix of the correlation estimators. This approach is obviously demanding in terms of the computational load if the entire covariance matrix is wanted. However, only the high lag-time estimators are expected to be seriously correlated, and, at least for them, this task should not be too overwhelming. It is along these lines that we are currently focusing our scientific efforts.

We thank G. Arcovito, M. De Spirito, M. Rocco, and U. Perini for the loan of some of the equipment used in this study and A. Bailey, E. Paganini, and F. Trespidi for helpful discussions. We also acknowledge the help and the suggestions received from A. Smart and D. S. Cannell for the generation of simulated DLS data. E. Paganini and F. Trespidi are also thanked for having introduced us to the world of LabVIEW, without which we would probably never have undertaken this project. This research was partially supported by funds from the ASI (the Italian Space Agency) and from the INFM (the Istituto Na-
zionale di Fisica della Materia) with a fellowship to D. Magatti.

References