Computer Methods and Programs in Biomedicine 222 (2022) 106933

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

Blood flow estimation via numerical integration of temporal autocorrelation function in diffuse correlation spectroscopy

Myeongsu Seong[a,b,c,1], Yoonho Oh[d,1], Kijoon Lee[e,*], Jae G. Kim[d,*]

[a] School of Information Science and Technology, Nantong University, Nantong, Jiangsu, China
[b] Research Center for Intelligent Information Technology, Nantong University, Nantong, Jiangsu, China
[c] Nantong Research Institute for Advanced Communication Technologies, Nantong, Jiangsu, China
[d] Department of Biomedical Science and Engineering, Gwangju Institute of Science and Technology, Gwangju, Republic of Korea
[e] Department of Electrical Engineering and Computer Science, Daegu Gyeongbuk Institute of Science and Technology, Daegu, Republic of Korea

ARTICLE INFO

Article history:
Received 14 January 2022
Revised 27 May 2022
Accepted 2 June 2022

Keywords:
Blood flow
Diffuse correlation spectroscopy
Numerical integration

ABSTRACT

Background and Objective: Diffuse correlation spectroscopy (DCS) is an optical technique widely used to monitor blood flow. Recently, efforts have been made to derive new signal processing methods to minimize the systems used and shorten the signal processing time. Herein, we propose alternative approaches to obtain blood flow information via DCS by numerically integrating the temporal autocorrelation curves.
Methods: We use the following methods: the inverse of K^2 (IK2)—based on the framework of diffuse speckle contrast analysis—and the inverse of the numerical integration of squared g_1 (INISg1) which, based on the normalized electric field autocorrelation curve, is more simplified than IK2. In addition, g_1 thresholding is introduced to further reduce computational time and make the suggested methods comparable to the conventional nonlinear fitting approach. To validate the feasibility of the suggested methods, studies using simulation, liquid phantom, and *in vivo* settings were performed. In the meantime, the suggested methods were implemented and tested on three types of Arduino (Arduino Due, Arduino Nano 33 BLE Sense, and Portenta H7) to demonstrate the possibility of miniaturizing the DCS systems using microcotrollers for signal processing.
Results: The simulation and experimental results confirm that both IK2 and INISg1 are sufficiently relevant to capture the changes in blood flow information. More interestingly, when g_1 thresholding was applied, our results showed that INISg1 outperformed IK2. It was further confirmed that INISg1 with g_1 thresholding implemented on a PC and Portenta H7, an advanced Arduino board, performed faster than did the deep learning-based, state-of-the-art processing method.
Conclusion: Our findings strongly indicate that INISg1 with g_1 thresholding could be an alternative approach to derive relative blood flow information via DCS, which may contribute to the simplification of DCS methodologies.

1. Introduction

Blood flow in the human body is instrumental in transporting oxygen throughout the body and discarding biowastes. Any disturbance in the blood flow leads to a plethora of problems depending on the location at which it occurs, ranging from tissue ulceration, paralysis, to even death. To prevent such serious situations, blood flow monitoring is of utmost importance. Among various technologies, diffuse correlation spectroscopy (DCS) is an emerging technique that monitors blood flow to a tissue in a noninvasive manner. DCS, also known as diffusing-wave spectroscopy in the chemistry, biology, and materials communities, was first proposed by Boas and Yodh as a blood flow monitoring technique [1]. It utilizes the dynamic light scattering caused by the interaction between photons and red blood cells to quantify blood flow. It has been compared with multiple blood flow monitoring techniques such as xenon-enhanced computed tomography [2], laser Doppler flowmetry [3], and arterial spin labeling [4], and has been established as a widely used technique for monitoring blood flow. DCS has been used to investigate various types of tissues, including those of the muscle, breast, and brain [5]. Due to its noninvasive ability to mea-

* Corresponding authors.
E-mail addresses: kjlee@dgist.ac.kr (K. Lee), jaekim@gist.ac.kr (J.G. Kim).
[1] The authors have contributed equally to this work.

https://doi.org/10.1016/j.cmpb.2022.106933

sure deep blood flow information and its relatively low cost, it is expected to have more applications in the future.

Recently, DCS has experienced various technological advancements. Among them, two advancements that have garnered much attention are the miniaturization/simplification of DCS-based blood flow monitoring systems [6–9] and the reduction of DCS signal processing time [10–15]. Conventionally, nonlinear fitting (NF) techniques, such as the Nelder–Mead simplex method (i.e., '*fminsearch*' function in MATLAB), are used to extract αD_B and blood flow index (BFI) in DCS [12,16,17]. This NF is one of the bottlenecks limiting the signal processing speed in DCS.

To miniaturize and simplify the system, charge-coupled devices (or complementary metal-oxide-semiconductor)-sensor-based methods have been introduced [7,9,18]. The miniaturization of the system can make them more user-friendly in clinics or critical care. To reduce the signal processing time and cost of the system, the modified Beer–Lambert law for blood flow [13], software autocorrelator [12,14,15], implementation of NF on a field-programmable gate array (FPGA) [12], and the use of the inverse of the delay time at the middle g_2 (normalized electric field autocorrelation function) between the minimum and maximum values of g_2 ($\tau_{1/2}$ method) [8] were introduced. In addition, recent studies have used artificial intelligence (AI) models such as deep learning (DL) [10] and long short-term memory (LSTM) [17] as alternatives to reduce the signal processing time. Although these approaches are effective, they remain restricted by their limited probing depth [7,9,18,19], vulnerability to noise [8,13], having been tested under an insufficient number of conditions [8,17], as well as the requirement of a large amount of training data and extended training time [10,17]. Moreover, as some of these approaches are computationally heavier, they may not be used to process fast flow signals in real-time on low-cost microprocessors [7,10]. Hence, new methods that can overcome these limitations must be developed for DCS.

In this paper, we introduce two numerical-integration-based methods for DCS to obtain the relative BFI, which we termed the IK2 (the inverse of K^2) and the INISg1 (the inverse of the numerical integration of squared g_1); the latter is based on the g_1 (normalized electric field autocorrelation function) of DCS. To validate these methods, simulations, liquid phantoms, and *in vivo* arterial arm-cuff occlusion experiments were performed. Then, their feasibilities under fast-flow conditions and for enhancing signal processing speeds were tested. Finally, the methods were ported and tested on three types of Arduino boards (Arduino Due, Arduino Nano 33 BLE Sense, and Portenta H7) to demonstrate their potential for the development of low-cost, real-time blood flow measurement systems.

2. Materials and methods

2.1. Working principle of DCS

For blood flow measurement, DCS detects the dynamic light scattering caused by the flow of red blood cells, which are the primary dynamic scatterers in a living body. A long-coherence laser irradiates the tissue of interest and a detector placed away from it collects the backscattered light, which contains information on dynamic scattering. A hardware/software correlator uses these signals collected in unit time to generate a normalized intensity autocorrelation function, $g_2(\tau)$, according to (1) [3,5,14,16,17,20]:

$$g_2(\tau) = \langle I(t) \cdot I(t+\tau) \rangle \,/ \langle I(t) \rangle^2. \tag{1}$$

Here, $I(t)$ denotes the signal intensity as a function of time, $\langle \cdots \rangle$ is the time average of the signal, and τ is the delay time [3,5,14,16,17,20]. Next, $g_2(\tau)$ is converted to an unnormalized electric field autocorrelation function, $G_1(\tau) = \langle E(t) \bullet E(t-\tau) \rangle$, following the Siegert relationship [3,5,9,10,12–14,16,17,19,20] as follows:

$$g_2(\tau) = 1 + \beta \frac{|G_1(\tau)|^2}{\langle I(\tau) \rangle^2}. \tag{2}$$

Here, β is a factor determined by the detector geometry (i.e., number of speckles detected) and polarization, and $E(t)$ is the time-dependent amplitude of the electric field. $G_1(\tau)$ from the data is fitted to a Green's function solution of the correlation diffusion equation to extract BFI [3,12,14,16,19,20] as follows:

$$G_1(\tau) = \frac{3\mu_s'}{4\pi} \left(\frac{\exp(-R_D r_1)}{r_1} - \frac{\exp(-R_D r_2)}{r_2} \right). \tag{3}$$

Here, $R_D^2 = 3\mu_s'\mu_a + \alpha\mu_s'^2 k_0^2 \langle \Delta \cdot r^2(\tau) \rangle$, $r_1 = \sqrt{\rho^2 + z_0^2}$, $r_2 = \sqrt{\rho^2 + (z_0 + 2z_b)^2}$, μ_s' is the reduced scattering coefficient, μ_a is the absorption coefficient, α is the ratio between the dynamic scatterers and all scatterers, $k_0 (= {}^{2\pi}/_{\lambda})$ is the source wavenumber, λ is the wavelength of the source, $\langle \Delta \bullet r^2(\tau) \rangle$ is the mean-square displacement, ρ is the distance between the source and the detector, $z_0 = {}^{1}/_{\mu_s'}$, $z_b = {}^{2(1+R_{eff})}/_{3\mu_s'(1-R_{eff})}$, and $R_{eff} = -1.440n^{-2} + 0.710n^{-1} + 0.668 + 0.0636n$, assuming the refractive index of the tissue, $n \approx 1.37$. The mean-square displacement of Brownian motion, $\langle \Delta r^2(\tau) \rangle = 6D_B\tau$, was assumed in this work [3,5,12,14,16,20], where D_B is the Brownian diffusion coefficient. Because it is difficult to estimate α owing to the limitations of DCS, αD_B is extracted as a single parameter and used as the BFI of DCS [3,5,12,14,16,19,20].

2.2. System configuration

An 852-nm, single-frequency laser (DL-852–100-SO, CrystaLaser, ~10 m coherence length) connected to a fiber attenuator (BB-100–11–685–200/240-QM-35–33–3-1-SP, 200 μm core size, OZ Optics) was used as the radiation source of the system, while a single-photon counting module (SPCM, SPCM-AQRH-12FC, Excelitas Technologies) connected to a single-mode fiber (SMF, P1–780A-FC-2, Thorlabs) was used as the detector. The distance between the source and the detector was set to 1 cm. The detector outputs a transistor-transistor logic (TTL) signal when a photon is collected. The TTL pulse train generated by the SPCM was provided as the counter input to a counter/timer board (PCIe-6612, National Instruments). The sampling frequency of this board was fixed at 1 MHz, with a 1 μs bin time interval throughout the experiments. An intensity autocorrelation function of the collected TTL pulse trains was generated using a fast Fourier transform (FFT)-based software autocorrelator [14]:

$$g_2(\tau) = F^{-1}[F\{I(t)\}F\{I(-t)\}] / \langle I(t) \rangle^2 = \langle \mathrm{I(t)} \cdot \mathrm{I(t+\tau)} \rangle \,/ \langle I(t) \rangle^2. \tag{4}$$

Here, $F\{\cdots\}$ and $F^{-1}[\cdots]$ denote the FFT and inverse FFT of the given signal, respectively. Note that we use the FFT-based software autocorrelator because this approach can perform the calculation of g_2 in a fairly short time, compared to the calculation of g_2 using Eq. (1).

A customized LabVIEW program was used to collect and save data, generate an autocorrelation function, and change measurement options, including the sampling frequency of the counter/timer board (LabVIEW 2016, National Instruments).

2.3. Numerical-integration-based estimation of blood flow

As an alternative approach, we introduce two methods based on numerical integration to obtain relative BFI via DCS. They are based on (5), which relates speckle fluctuation to the normalized electric field autocorrelation function, g_1. Eq. (5) is the theoretical

basis of diffuse speckle contrast analysis (DCSA), which is another optical method used in blood-flow measurement [7,18,19].

$$K(T)^2 = \left(\frac{\sigma_t}{\langle I_t \rangle}\right)^2 = \frac{2\beta}{T}\int_n^T \left(1 - \frac{\tau}{T}\right)[g_1(\tau)]^2 d\tau. \quad (5)$$

Here, $K(T)$ is the temporal speckle contrast, n is the minimum delay time, T is the maximum delay time, and σ_t and $\langle I_t \rangle$ are the standard deviation (SD) and mean of intensity, respectively; β is a constant determined by the detector geometry and polarization, while τ is the delay time. Eq. (6), a simplified expression of the right-hand side of (5), is the first method that we explore in this work. In practice, in addition to BFI, β is extracted through NF [12,16]. However, to simplify our calculations, we removed T and 2β from the right side of (5), assuming them to be unchanging constants throughout the signal measurement. Although there are subtle differences compared to the original form of K^2, this method is referred to as IK2 throughout the manuscript and denotes the inverse of K^2:

$$IK2 = 1/\int_n^T \left(1 - \frac{\tau}{T}\right)[g_1(\tau)]^2 d\tau. \quad (6)$$

Additionally, a more simplified method, shown in (7), was explored. Henceforth, (7) is referred to as INISg1 (the inverse of the numerical integration of squared g_1). Compared to IK2, INISg1 is not weighted for the delay time.

$$INISg1 = 1/\int_n^T [g_1(\tau)]^2 d\tau. \quad (7)$$

Fig. 1(a) summarizes the three methods tested in this work: the conventional NF method for extracting BFI in DCS, and the IK2 and INISg1 methods, which are the numerical-integration-based methods proposed in this work.

From preliminary simulation experiments, we found that the flow estimated using either IK2 or INISg1 suffers from significant underestimation relative to the designated flow. Attributing this to the accumulation of noise and non-relevant g_1 values, g_1 was thresholded within a certain value of g_1 (the threshold) by first selecting the shortest delay time with g_1 closest to the threshold, and then performing numerical integration from the shortest delay time of the system to the selected delay time.

2.4. Bland–altman analysis (BAA)

For the comparison of different methods, BAA, which visualizes the discrepancy or similarity of two different methods, was used in this work [7,21]. In this analysis, a thick black line indicates the mean of the differences between the methods ($mean = (method\,A + method\,B)/2$), while the red dotted lines indicate the upper and lower limits of agreement ($mean \pm 2 \times std(method\,A - method\,B)$). The smaller the mean difference and the narrower the range of these limits, the less discrepancies there are between these two tested methods.

2.5. Simulation

To confirm the feasibility of the suggested methods and determine an appropriate value for g_1 thresholding, a set of autocorrelation functions were generated using (3) by varying αD_B from 5×10^{-10} to 5.005×10^{-8} cm^2/s (with a step size of 0.5×10^{-10} cm^2/s), assuming $\mu_s' = 11.134\,\text{cm}^{-1}$ and $\mu_a = 0.0449\,\text{cm}^{-1}$ at 852 nm, a 1 cm source-detector separation, $\beta = 0.45$ (β of (2) for converting g_2 to g_1) and a 1 MHz sampling frequency. To make the simulation data more realistic, a DCS noise model was employed [13,15,16,22], in which the bin time interval was set to 1 μs and the photon count rate was set to 100 kcps. These noise-added $g_1(\tau)$ values were directly numerically integrated via the IK2 and INISg1 methods. The integration time was set to 1 s, and the value of g_1 for thresholding was varied from 0 (no thresholding) to 0.35, in increments of 0.05. The results of normalized NF, normalized IK2 with and without g_1 thresholding, and normalized INISg1 with and without g_1 thresholding were compared with the normalized assigned flow values using BAA.

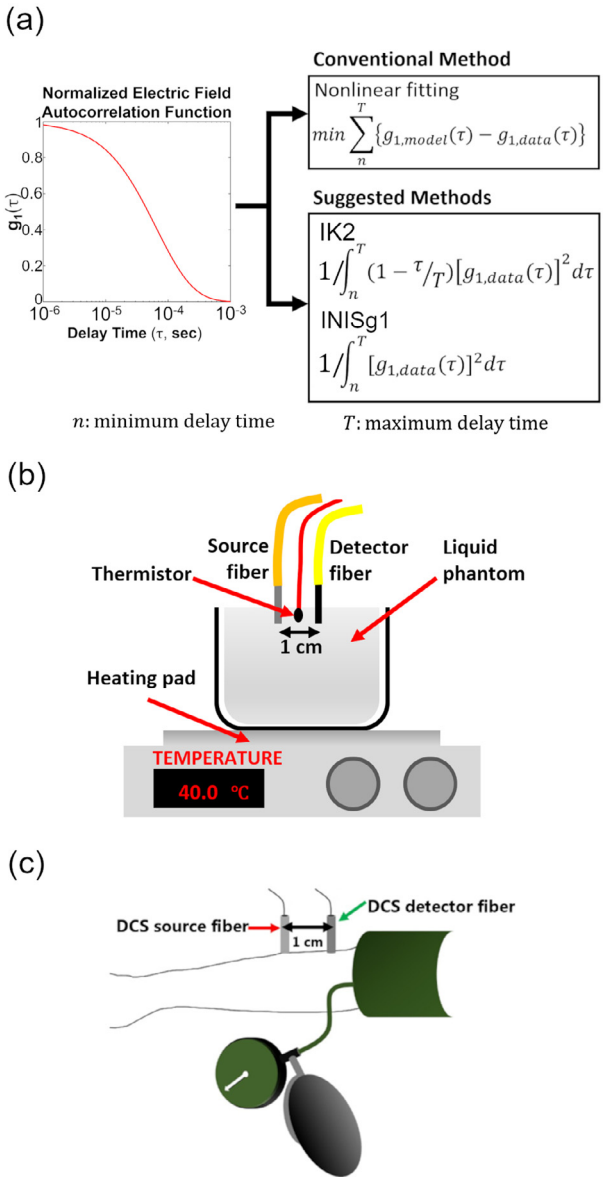

Fig. 1. (a) Comparison of conventional and suggested methods investigated for deriving BFI; in this work, n and T are fixed as 1 μs and 2 ms, respectively, except for simulation. (b) Schematic of a liquid phantom test. (c) Schematic of an arterial arm-cuff occlusion test.

2.6. Phantom experiment

A phantom experiment was performed to validate the proposed methods. A simple liquid phantom was constructed by mixing distilled water and 20% intravenous fat emulsion (Lipision 20%, JW Pharm.) that resulted in $\mu_s' = 11.134\,\text{cm}^{-1}$ and $\mu_a = 0.0449\,\text{cm}^{-1}$ at 852 nm [23,24]. Note that a similar liquid phantom has been used to evaluate other optical systems, including optoacoustic tomography, interferometric diffusing wave spectroscopy, and NIR-II fluorescence imaging system [25–27]. Its temperature was varied from 25 to 45 °C (in increments of 5 ºC) using a hot plate

Table 1
Linear regression results of (Assigned blood flow) = A^*(IK2)+B. A: slope, B: intercept, T: maximum time delay, G: the value of g_1 for thresholding, No: no g_1 thresholding, and E: error. A, B, and E of the condition (T: 2 ms and G: 0.2) used in the manuscript are presented as red text.

T	**1 s**			**100 ms**			**10 ms**			**5 ms**			**2 ms**			**1 ms**		
G	***a***	***b***	***e***	***a***	***b***	***e***	***a***	***b***	***e***	***a***	***b***	***e***	***a***	***b***	***e***	***a***	***b***	***e***
No	2348	−2454	106.5	219.8	−311.5	92.75	19.74	−71.70	10.55	10.55	−48.74	39.19	5.973	−28.27	23.29	5.169	−18.06	17.94
0.05	1.008	0.698	12.59	1.014	0.692	12.59	1.079	0.627	12.54	1.162	0.553	12.48	1.503	0.323	12.31	2.172	−0.091	12.02
0.1	0.994	0.792	11.14	1.000	0.786	11.14	1.061	0.726	11.13	1.138	0.659	11.11	1.454	0.447	11.08	2.102	0.067	11.04
0.15	1.000	0.938	10.13	1.005	0.933	10.13	1.062	0.876	10.13	1.133	0.812	10.12	1.420	0.612	10.11	2.052	0.252	10.10
0.2	0.995	0.895	9.140	1.001	0.890	9.139	1.055	0.836	9.133	1.123	0.775	9.127	1.391	0.587	9.110	2.006	0.251	9.094
0.25	0.987	0.989	10.67	0.992	0.984	10.66	1.043	0.934	10.66	1.106	0.878	10.65	1.350	0.704	10.63	1.936	0.393	10.61
0.3	1.011	1.120	10.18	1.015	1.115	10.18	1.060	1.069	10.18	1.115	1.018	10.17	1.322	0.859	10.16	1.851	0.577	10.14
0.35	0.975	1.117	10.96	0.979	1.113	10.96	1.021	1.072	10.96	1.073	1.025	10.95	1.266	0.881	10.95	1.762	0.627	10.93

and monitored using a thermistor (TH10K, Thorlabs) connected to an Arduino (Arduino Uno, Arduino). While varying temperature, 60 raw data points (5 (steps) × 60 (s) × 1000,000 data points) were acquired and saved for post-processing. Fig. 1(b) shows a schematic of the phantom measurements. The results of normalized IK2 and normalized INISg1 with and without g_1 thresholding were compared with those of normalized NF using BAA.

2.7. In vivo *arm-cuff occlusion*

To test the *in vivo* applicability of the suggested methods, an arterial arm-cuff occlusion paradigm with 220 mmHg pressure was applied, as shown in Fig. 1(c). A healthy subject was recruited for measurement, and the cuff occlusion protocol was reviewed and approved by the Institutional Review Board of the Gwangju Institute of Science and Technology (IRB 20,140,319-HR-10–01–02). An arm cuff was placed on the subject's upper arm, and DCS signals were obtained from their lower arm. The measurement times included a baseline of 60 s, cuff occlusion of 180 s, and another 180 s after cuff deflation. Laser power was set to 10.6 mW to ensure that the tissue was not damaged. Raw data (420 (s) × 1000,000 data points) were acquired and saved for post-processing. For NF, $\mu_s' = 6.02\ \mathrm{cm}^{-1}$ and $\mu_a = 0.139\ \mathrm{cm}^{-1}$ were assumed [28]. The results of normalized IK2 with and without g_1 thresholding and those of normalized INISg1 with and without g_1 thresholding were compared with those of normalized NF using BAA.

2.8. Implementation of IK2 and INISg1 in arduino

To demonstrate the utility of the suggested methods in DCS simplification, we implemented IK2 and INISg1 with and without g_1 thresholding for three different Arduino boards: Arduino Due (microcontroller: AT91SAM3×8E, clock speed: 84 MHz, SRAM: 96 kB, Arduino), Arduino Nano 33 BLE Sense (microcontroller: nRF52840, clock speed: 64 MHz, SRAM: 256 kB, Arduino), and Portenta H7 (microcontroller: STM32H747XI, clock speed: 480 MHz and 240 MHz (dual-core), SRAM: 1 MB, SDRAM: 8 MB, Arduino). However, the implemented algorithms could not be uploaded to the Arduino Uno (microcontroller: ATmega328P, clock speed: 16 MHz, SRAM: 2 kB, Arduino Uno, Arduino) due to its SRAM limitation. Fig. 2 shows the pseudocodes of IK2 and INISg1 with and without g_1 thresholding, as implemented on the Arduino boards.

IK2 without g_1 thresholding

```
tauArr = 1e-6:1e-6:2e-3;
β = 0.45;
K2 = 0.0;
FOR (int i = 0; i < (size of g2); i++)
        g1[i] = Square root(abs(g2[i] - 1) / β);
        K2 = K2 + (1 - tauArr[i]/ tauArr[(size of g2)-1]) * (g1[i])²;
END
IK2 = 1/K2;
```

INISg1 without g_1 thresholding

```
NISg1 = 0.0;
FOR (int i = 0; i < (size of g2); i++)
        g1[i] = Square root(abs(g2[i] - 1) / β));
        NISg1 = NISg1 + (g1[i])²;
END
INISg1 = 1/NISg1;
```

IK2 with g_1 thresholding

```
tauArr = 1e-6:1e-6:2e-3;
β = 0.45;
threshold = 0.2;
NIK2 = 0.0;
FOR (int i = 0; i < (size of g2); i++)
        g1[i] = Square root(abs(g2[i] - 1) / β);
        K2 = K2 + (1 - tauArr[i]/ tauArr[(size of g2)-1] )* (g1[i])²;
        IF (g1[i] <= threshold)
                break;
        END
END
IK2 = 1/K2;
```

INISg1 with g_1 thresholding

```
threshold = 0.2;
NISg1 = 0.0;
FOR (int i = 0; i < (size of g2); i++)
        g1[i] = Square root(abs(g2[i] - 1) / β);
        NISg1 = NISg1 + (g1[i])²;
        IF (g1[i] <= threshold)
                break;
        END
END
INISg1 = 1/NISg1;
```

Fig. 2. Pseudocodes of the implemented algorithms of IK2 and INISg1 with and without g_1 thresholding on the Arduino boards. **FOR**: the beginning of *for* loop; **IF**: the beginning of *if* structure; **END**: end of *for* loop or *if* structure; *tauArr*: array of delay times from 1 μs to 2 ms, in increments of 1 μs; *abs*: absolute value; β: parameter of Siegert relationship (determined by detector geometry – i.e. number of speckles detected – and polarization); *K2*: K^2; *NISg1*: numerical integration of squared g_1; *IK2*: the inverse of K^2; *INISg1*: the inverse of numerical integration of squared g_1; *threshold*: the value for g_1 thresholding.

3. Results

3.1. Optimization of maximum delay time and g_1 thresholding

Table 1 shows the linear regression results between the assigned flows and the results of IK2 with and without g_1 thresholding. Table 2 shows the corresponding results between the assigned flows and the results of INISg1 with and without g_1 thresholding. Linear regression between the assigned flows and NF resulted in a (slope) = 1.014, b (intercept) = 0.147, and e (error) = 8.158. Note that, e is the maximum absolute deviation, which is calculated via $e = max(|estimated\ data - assigned\ data|)$ [29]. When g_1 thresholding was not applied, e of the linear regression between the assigned flow and IK2 decreased as T (the maximum delay time) decreased till 10 ms, suggesting an improvement in the regression results (Table 1). A similar trend was observed as T decreased till

Table 2
Linear regression results of (Assigned blood flow) = A^*(INISg1)+B. A: slope, B: intercept, T: maximum time delay, G: the value of g_1 for thresholding, No: no g_1 thresholding, and E: error. A, B, and E of the condition (T: 2 ms and G: 0.2) used in the manuscript are presented as red text.

T	1 s			100 ms			10 ms			5 ms			2 ms			1 ms		
G	a	b	e	a	b	e	a	b	e	a	b	e	a	b	e	a	b	e
No	4688	−4793	106.7	449.9	−547.0	98.16	38.48	−104.7	67.20	18.33	−70.42	53.09	7.763	−41.67	34.91	5.257	−26.93	22.67
0.05	1.007	0.699	12.59	1.007	0.699	12.59	1.007	0.699	12.59	1.007	0.699	12.59	1.045	0.699	12.59	1.340	0.689	12.59
0.1	0.994	0.793	11.14	0.994	0.793	11.14	0.994	0.793	11.14	0.994	0.793	11.14	1.011	0.793	11.14	1.296	0.785	11.14
0.15	0.999	0.939	10.13	0.999	0.939	10.13	0.999	0.939	10.13	0.999	0.939	10.13	0.999	0.939	10.13	1.265	0.933	10.14
0.2	0.995	0.895	9.140	0.995	0.895	9.140	0.995	0.895	9.140	0.995	0.895	9.140	0.995	0.895	9.140	1.237	0.891	9.143
0.25	0.987	0.990	10.67	0.987	0.990	10.67	0.987	0.990	10.67	0.987	0.990	10.67	0.987	0.990	10.67	1.194	0.987	10.67
0.3	1.010	1.120	10.18	1.010	1.120	10.18	1.010	1.120	10.18	1.010	1.120	10.18	1.010	1.120	10.18	1.141	1.119	10.18
0.35	0.974	1.118	10.96	0.974	1.118	10.96	0.974	1.118	10.96	0.974	1.118	10.96	0.974	1.118	10.96	1.086	1.117	10.96

1 ms in the linear regression results of INISg1 without g_1 thresholding (Table 2). This may be due to the reduction in the contribution of the noise signals to the numerical integration in both IK2 and INISg1. Meanwhile, when g_1 thresholding was not used, the linear regression errors of IK2 were always smaller than those of INISg1 throughout all of the T tested.

When g_1 thresholding was applied, the linear regression results of IK2 and INISg1 showed different tendencies. When it comes to $T = 1$ s, both IK2 and INISg1 exhibit very similar regression outcomes, with only minor differences. In the case of IK2, however, for every G (the value for g_1 thresholding), a increases as T decreases, which implies that the results of IK2 become less relevant to the assigned flow (Table 1). On the other hand, the linear regression results of INISg1 do not change as T varies from 1 s to 2 ms, whereas a becomes larger when $T = 1$ ms (Table 2). Overall, when g_1 thresholding was applied, INISg1 was more robust than IK2.

Considering a, e, and calculation speed (larger T leads to longer calculation time), parametric values of $T = 2$ ms and $G = 0.2$ are used in the later sections for processing simulation and experimental data. While this condition is not optimal for IK2, it is used for IK2 as well to enable a fair comparison of signal processing time.

3.2. Simulation

Fig. 3 shows the simulation results. Based on the results from Section 3.1, the maximum delay time and a g_1 thresholding value were set to 2 ms and 0.2, respectively. As shown in Fig. 3(a), IK2 and INISg1 significantly underestimated the flow information when g_1 thresholding was not applied. In addition, until the assigned flow reached 1.32×10^{-8} cm^2/s, INISg1 was closer to the assigned flow than was IK2; this situation was reversed when the assigned flow exceeded 1.32×10^{-8} cm^2/s. Based on this observation, this flow rate is defined as the transition value (TV) in this study.

Similar to the results in Section 3.1, g_1 thresholding led to more accuracy in both IK2 and INISg1, compared to the cases without g_1 thresholding. More importantly, when g_1 thresholding was applied, as shown in Fig. 3(c) and (d), INISg1, a simplified numerical integration method, showed less discrepancy than IK2 when compared with the assigned blood flow. Comparing Fig. 3(b) and (d), INISg1 performed comparably to NF, with a similar mean difference (0.09 vs. 0.51) and a similar range of limits of agreement (7.42 vs. 6.98). In the meantime, with g_1 thresholding, INISg1 always performed better than IK2, showing minor discrepancies from the assigned blood flow.

3.3. Phantom experiment

Fig. 4 shows the results of the phantom experiments. As shown in Fig. 4(a), without g_1 thresholding, both IK2 and INISg1 led to more underestimated flow properties than did conventional NF, as

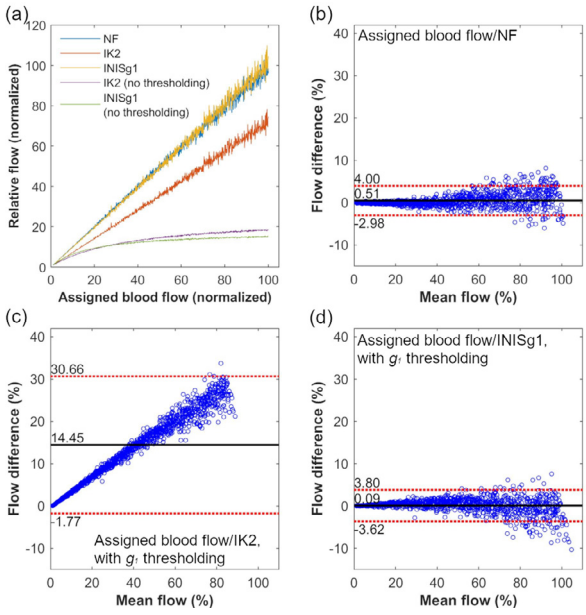

Fig. 3. (a) Comparison of assigned blood flow vs. estimated flow of NF, IK2 and INISg1 with (not explicitly indicated) and without g_1 thresholding (indicated with '(no thresholding)'). BAA of: (b) assigned blood flow and NF, (c) assigned blood flow and IK2 with g_1 thresholding, and (d) assigned blood flow and INISg1 with g_1 thresholding.

the temperature increased. In addition, the higher the phantom temperature, the more significant was this underestimation. The BAA of NF vs. IK2 (Fig. 4(c)) and that of NF vs. INISg1 (Fig. 4(e)) reveal quite large mean differences and a wide range of limits of agreement. As shown in Fig. 4(b), the discrepancy between the NF and the suggested methods decreased when g_1 thresholding was applied. In the case of the BAA of NF and IK2, the introduction of g_1 thresholding (Fig. 4(d)) changed the mean difference and the range of limit of agreements from 9.13 and 28.09 (i.e., non-thresholded values) to 1.35 and 9.01 (thresholded values), respectively. By contrast, the corresponding values of the BAA of NF and INISg1 (Fig. 4(f)) changed from 13.58 and 42 (non-thresholded values) to 0.67 and 8.84 (thresholded values), respectively, showing more significant improvement than IK2 with g_1 thresholding.

3.4. In vivo *arm-cuff occlusion*

Fig. 5 shows the results of the arm-cuff occlusion experiment. As shown in Fig. 5(a), without g_1 thresholding, both IK2 and INISg1 underestimated the flow compared to the NF methods. The BAA of NF vs. IK2 (Fig. 5(c)) and that of NF vs. INISg1 (Fig. 5(e)) show markedly large mean differences and a wide range of limits of agreements. In addition, the underestimation becomes more

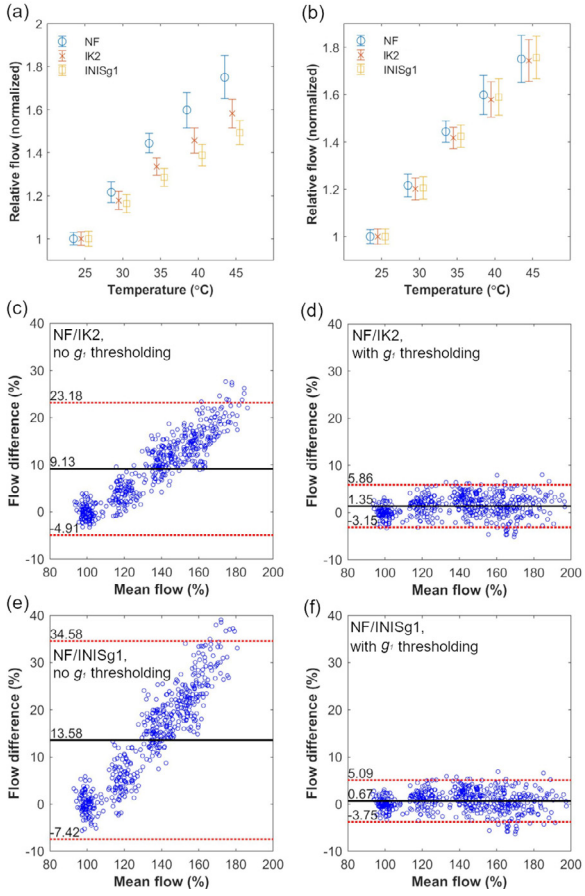

Fig. 4. Relative flow changes of NF, IK2, and INISg1 without (a) and with (b) g_1 thresholding as the temperature rises in the liquid phantom experiment. BAA of NF vs. IK2 without (c) and with (e) g_1 thresholding, and BAA of NF vs. INISg1 without (d) and with (f) g_1 thresholding.

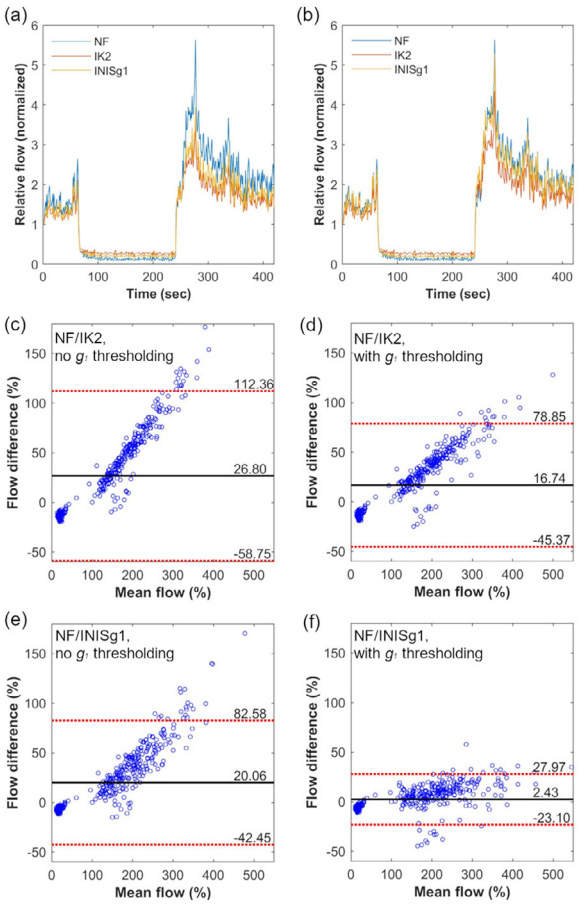

Fig. 5. Relative flow changes (1 Hz) of NF, IK2, and INISg1 without (a) and with (b) g_1 thresholding in the arm-cuff occlusion experiment. BAA of NF vs. IK2 without (c) and with (e) g_1 thresholding, and BAA of NF vs. INISg1 without (d) and with (f) g_1 thresholding.

significant at higher flows. As shown in Fig. 5(b), the discrepancy between the NF and the suggested methods became smaller when g_1 thresholding was applied. In the case of BAA of NF and IK2, the introduction of g_1 thresholding (Fig. 5(d)) changed the mean difference and the range of limit of agreements from 26.80 and 171.11 (i.e., the non-thresholded values) to 16.74 and 124.22 (thresholded values), respectively. On the other hand, for the BAA of NF and INISg1, the introduction of g_1 thresholding (Fig. 5(f)) changed the mean difference and the range of limit of agreements from 20.06 and 125.03 (non-thresholded values) to 2.43 and 51.07 (thresholded values), respectively, showing a more pronounced improvement than the case of IK2 with g_1 thresholding.

3.5. Comparison of signal processing speed

As previously described, one of the purposes of the suggested approach is to reduce the signal processing time. Therefore, their processing speeds were compared with NF using the *timeit* function in MATLAB (MATLAB R2018b, Mathworks) on a PC (i7–10,700, 8-core, Intel CPU) with 32 GB RAM. As described in Section 3.1, $T = 2$ ms and a threshold of 0.2 for g_1 thresholding were used. Since the processing times of IK2 and INISg1 with g_1 thresholding decrease as the flow of g_2 increases, noise-added g_2 curves were tested for ten different assigned flows. For each condition, the mean and SD values of the processing times were calculated from 50 trials performed. Tables 3–5 show the mean and SD of the processing times of NF, IK2 with and without g_1 thresholding, and INISg1 with and without g_1 thresholding when processing one, ten, and 1000 g_2 curves (including a procedure for converting g_2 to g_1 using the Siegert relationship). Processing a single g_2 took 4.96 ms (grand mean of the processing time of g_2 with the different assigned flows) for NF. Among the assigned flows, NF took the shortest time (3.31 ms) when processing g_2 with 1.05×10^{-8} cm^2/s, since the initial value of the flow for NF was 1.00×10^{-8} cm^2/s. This trend continued during the processing of 10 and 1000 g_2 curves.

The grand means of the processing times of IK2 and INISg1 without g_1 thresholding were 29.21 and 26.43 μs, respectively. Compared to NF, IK2 and INISg1 were approximately 170 and 188 times faster, respectively. As expected, due to reduced calculations, INISg1 was approximately 1.1 times faster than IK2. On the other hand, with g_1 thresholding, IK2 (12.58 μs) and INISg1 (12.20 μs) were approximately 394 and 407 times faster than NF, respectively. INISg1 with g_1 thresholding was approximately 1.03 times faster than IK2 with g_1 thresholding. Based on the grand mean of processing time, IK2 with g_1 thresholding and INISg1 with g_1 thresholding perform approximately 2.3 and 2.2 times faster than those without g_1 thresholding, owing to the reduced calculations. The time reduction becomes more significant as the flow increases.

This tendency continues as more curves are processed. For processing 1000 g_1, NF took 5172.90 ms, while non-thresholded IK2 and INISg1 took 35.54 and 33.72 ms, performing approximately 146 and 153 times faster than NF, respectively. When g_1 thresholding was applied, IK2 and INISg1 took 10.09 and 9.91 ms, performing nearly 513 and 522 times faster than NF, respectively. Overall,

Table 3
Processing time and estimated flow of one g_2 curve of assigned flows of NF, and IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding. Time: (mean) ± (SD). Grand mean of processing time of NF: 4.96 ms; grand mean of processing time of IK2 without g_1 thresholding: 29.21 µs; grand mean of processing time of INISg1 without g_1 thresholding: 26.43 µs; grand mean of processing time of IK2 with g_1 thresholding: 12.58 µs; grand mean of processing time of INISg1 with g_1 thresholding: 12.20 µs.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
NF	1	10.69	20.38	30.69	39.98	50.30	61.17	66.67	76.79	90.18
Time	6.60	4.67	3.31	4.32	4.87	5.08	5.52	5.09	5.03	5.11
(ms)	±0.05	±0.31	±0.03	±0.04	±0.04	±0.05	±0.11	±0.06	±0.03	±0.04
IK2	1	5.89	9.27	11.65	12.96	14.28	15.64	16.42	17.01	18.01
Time	30.00	29.15	29.20	29.14	29.15	29.24	29.12	28.97	29.03	29.08
(µs)	±2.04	±0.23	±0.31	±0.30	±0.25	±0.60	±0.24	±0.37	±0.39	±0.35
INISg1	1	6.65	9.64	11.24	12.07	12.67	13.73	14.12	14.13	15.04
Time	27.68	26.36	26.47	26.30	26.29	26.32	26.30	26.18	26.20	26.22
(µs)	±4.16	±0.18	±0.78	±0.30	±0.31	±0.30	±0.30	±0.34	±0.37	±0.26
IK2 with g_1 thresholding	1	7.72	14.59	21.08	28.76	34.97	44.65	50.10	59.05	65.55
Time	29.81	11.88	11.02	10.79	10.48	10.48	10.39	10.30	10.33	10.30
(µs)	±4.75	±0.63	±0.11	±0.14	±0.10	±0.09	±0.15	±0.14	±0.13	±0.12
INISg1 with g_1 thresholding	1	10.48	20.08	29.14	39.92	48.56	62.12	69.73	82.22	91.30
Time	27.24	11.46	10.81	10.62	10.32	10.47	10.38	10.22	10.22	10.22
(µs)	±4.20	±0.08	±0.10	±0.12	±0.11	±0.67	±0.68	±0.14	±0.14	±0.12

Table 4
Processing time and estimated flow of ten g_2 curves of assigned flows of NF, and IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding. Time: (mean) ± (SD). Grand mean of processing time of NF: 51.44 ms; grand mean of processing time of IK2 without g_1 thresholding: 279.36 µs; grand mean of processing time of INISg1 without g_1 thresholding: 261.49 µs; grand mean of processing time of IK2 with g_1 thresholding: 106.20 µs; grand mean of processing time of INISg1 with g_1 thresholding: 103.37 µs.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
NF	1	10.69	20.38	30.69	39.98	50.30	61.17	66.67	76.79	90.18
Time	68.77	47.75	33.67	44.56	50.60	52.82	57.51	52.92	52.47	53.30
(ms)	±1.36	±0.71	±0.63	±0.69	±0.68	±0.62	±0.58	±0.60	±0.61	±0.62
IK2	1	5.89	9.27	11.65	12.96	14.28	15.64	16.42	17.01	18.01
Time	285.35	280.10	278.62	279.49	278.14	279.00	278.36	278.82	278.25	277.51
(µs)	±35.45	±5.66	±3.13	±2.48	±2.13	±2.49	±3.43	±2.68	±3.31	±3.15
INISg1	1	6.65	9.64	11.24	12.07	12.67	13.73	14.12	14.13	15.04
Time	264.96	261.89	261.61	260.95	261.55	261.01	260.95	261.30	260.43	260.21
(µs)	±36.15	±2.15	±5.72	±3.17	±2.64	±3.10	±3.03	±2.95	±3.22	±3.38
IK2 with g_1 thresholding	1	7.72	14.59	21.08	28.76	34.97	44.65	50.10	59.05	65.55
Time	275.88	99.40	89.69	89.01	85.65	85.83	84.56	84.28	83.99	83.71
(µs)	±50.95	±4.64	±1.05	±2.02	±0.74	±0.83	±1.08	±1.01	±1.14	±1.12
INISg1 with g_1 thresholding	1	10.48	20.08	29.14	39.92	48.56	62.12	69.73	82.22	91.30
Time	250.28	97.82	89.76	88.51	86.06	85.45	84.24	84.15	83.80	83.60
(µs)	±13.61	±4.81	±5.41	±3.21	±4.78	±0.78	±1.10	±0.87	±1.12	±1.08

these results prove IK2 and INISg1 to be preferable for the offline processing of DCS signals.

3.6. Fast flow measurement during arm-cuff occlusion

To further test the suitability of the suggested methods in quantifying fast flow, raw data from the *in vivo* arm-cuff occlusion experiment were used again. To balance signal quality with measurement speed, each 1 s pulse train was divided into 25 chunks (i.e., 25 Hz), resulting in 40 ms as the maximum delay time of g_2. Pulsatile fluctuation due to heart rate, which is ∼1 Hz, can be resolved by monitoring fast flow [9,15,17,30]. A one-dimensional scalogram (MATLAB R2018b, Mathworks), based on continuous wavelet transform (CWT) for which the Morse wavelet was used as the mother wavelet, was implemented.

Fig. 6 shows the flow changes and the scalogram of NF, IK2 with and without g_1 thresholding, and INISg1 with and without g_1 thresholding. The gray region outside the dashed white line in Fig. 6(c)–(g) indicates the region excluded from analysis owing to the edge effect. As shown in Fig. 6(a) and (b), flow underestimation by IK2 and INISg1 is reduced as g_1 thresholding is applied, as discussed in Sections 3.3 and 3.4. Fig. 6(c) shows a CWT-based scalogram of the NF. In the scalogram, before and after cuff occlusion (before 60 s and after 240 s), ∼1 Hz and its harmonics exist, while frequency components mostly disappear during cuff occlusion (between 60 s and 240 s). Meanwhile, compared to the NF shown in Fig. 6(c), (d)–(g) represent almost the same frequency components including ∼1 Hz heart rate, its harmonics, and the disappearance of amplitude during cuff occlusion but with lower intensity (amplitude becomes stronger on moving from (d) to (g)). In short, regardless of using g_1 thresholding, both IK2 and INISg1 can be used to monitor heart rate as well.

3.7. IK2 and INISg1 in arduino

Tables 6–8 summarize the results of IK2 with and without g_1 thresholding, and INISg1 with and without g_1 thresholding on the Arduino Due, Arduino Nano 33 BLE Sense, and the Portenta H7 boards, respectively. In Arduino, INISg1 with g_1 thresholding still performs the fastest, followed by IK2 with g_1 thresholding, INISg1

Table 5
Processing time and estimated flow of 1000 g_2 curves of assigned flows of NF, and IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding. Time: (mean) ± (SD). Grand mean of processing time of NF: 5172.90 ms; grand mean of processing time of IK2 without g_1 thresholding: 35.54 ms; grand mean of processing time of INISg1 without g_1 thresholding: 33.72 ms; grand mean of processing time of IK2 with g_1 thresholding: 10.09 ms; grand mean of processing time of INISg1 with g_1 thresholding: 9.91 ms.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
NF	1	10.69	20.38	30.69	39.98	50.30	61.17	66.67	76.79	90.18
Time	6949.90	4809.68	3399.75	4481.50	5086.35	5307.78	5771.65	5312.16	5258.89	5351.34
(ms)	±6.30	±6.48	±4.70	±12.44	±61.54	±14.30	±12.04	±10.34	±4.96	±6.83
IK2	1	5.89	9.27	11.65	12.96	14.28	15.64	16.42	17.01	18.01
Time	35.56	35.49	35.54	35.48	35.56	35.57	35.56	35.55	35.54	35.57
(ms)	±0.14	±0.16	±0.11	±0.17	±0.14	±0.16	±0.17	±0.20	±0.17	±0.19
INISg1	1	6.65	9.64	11.24	12.07	12.67	13.73	14.12	14.13	15.04
Time	33.76	33.67	33.73	33.62	33.81	33.71	33.72	33.73	33.72	33.70
(ms)	±0.11	±0.11	±0.12	±0.14	±0.87	±0.13	±0.14	±0.16	±0.13	±0.15
IK2 with g_1 thresholding	1	7.72	14.59	21.08	28.76	34.97	44.65	50.10	59.05	65.55
Time	26.01	9.38	8.66	8.42	8.16	8.17	8.04	8.02	7.99	8.01
(ms)	±0.09	±0.10	±0.06	±0.09	±0.05	±0.06	±0.06	±0.06	±0.06	±0.05
INISg1 with g_1 thresholding	1	10.48	20.08	29.14	39.92	48.56	62.12	69.73	82.22	91.30
Time	24.70	9.27	8.57	8.36	8.20	8.12	8.00	7.99	7.96	7.97
(ms)	±0.09	±0.31	±0.06	±0.10	±0.45	±0.05	±0.06	±0.06	±0.07	±0.07

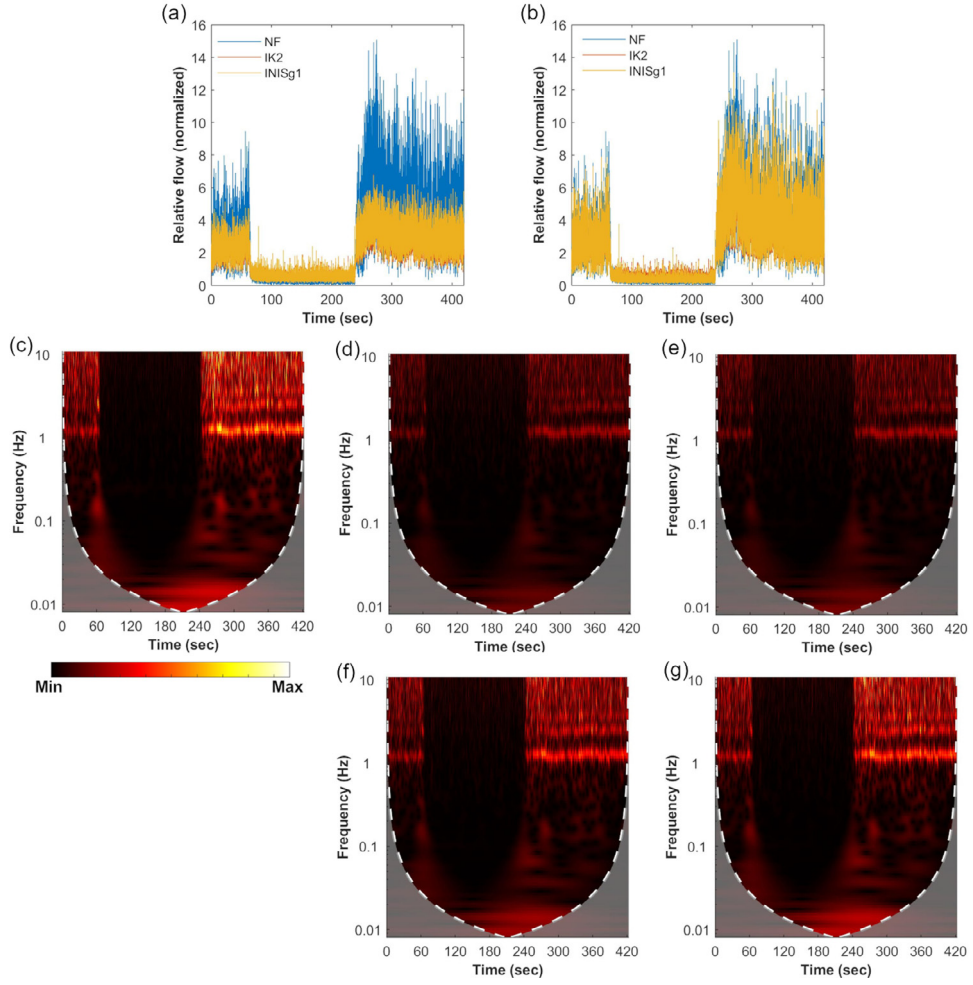

Fig. 6. Relative fast flow index changes (25 Hz) of NF, IK2, and INISg1 without (a) and with (b) g_1 thresholding in the arm cuff occlusion experiment, continuous wavelet transform-based scalogram of NF (c), IK2 (d), and INISg1 (e) without g_1 thresholding (e), and IK2 (f) and INISg1 (g) with g_1 thresholding. (c)-(g) share the same color bar. The region indicated by the white dashed line and gray color in (c)–(g) presents regions with significant edge effects, which is not considered in the analysis.

Table 6
Summary of assigned flows for g_2 curves, and estimated flow and calculation time of IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding performed on Arduino Due. Except for assigned flows, only relative values are shown for estimated flows through IK2 and INISg1. Time: (mean) ± (SD). Grand mean of processing time of IK2 without g_1 thresholding: 71.97 ms; grand mean of processing time of INISg1 without g_1 thresholding: 63.90 ms; grand mean of processing time of IK2 with g_1 thresholding: 8973.40 µs; grand mean of processing time of INISg1 with g_1 thresholding: 7547.82 µs.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
IK2	1.00	2.11	2.42	2.56	2.63	2.68	2.71	2.74	2.76	2.77
Time	72.01	71.97	72.02	71.97	71.94	71.98	71.90	71.94	72.01	71.94
(ms)	±0.01	±0.00	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.00
INISg1	1	1.72	1.86	1.92	1.94	1.96	1.98	1.99	1.99	2
Time	63.94	63.90	63.95	63.90	63.87	63.91	63.83	63.87	63.94	63.86
(ms)	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01	±0.01
IK2 with g_1 thresholding	1	7.72	14.57	21.06	28.7	34.91	44.49	49.89	58.80	65.51
Time	67,154.76	7376.24	3962.36	3143.68	1808.96	1803.04	1245.44	1129.36	1061.68	1048.48
(µs)	±9.76	±3.80	±2.38	±2.57	±1.91	±2.22	±1.94	±1.91	±1.99	±1.31
INISg1 with g_1 thresholding	1.00	10.48	20.06	29.11	39.84	48.48	61.90	69.43	81.88	91.25
Time	56,442.88	6221.76	3336.48	2662.80	1524.88	1516.96	1041.44	948.40	901.52	881.12
(µs)	±9.93	±3.80	±2.87	±2.02	±2.03	±2.07	±1.94	±2.18	±1.96	±1.81

Table 7
Summary of assigned flows for g_2 curves, and estimated flow and calculation time of IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding performed on Arduino Nano 33 BLE Sense. Except for assigned flows, only relative values are shown for estimated flows through IK2 and INISg1. Time: (mean) ± (SD). Grand mean of processing time of IK2 without g_1 thresholding: 3503.09 µs; grand mean of processing time of INISg1 without g_1 thresholding: 2855.16 µs; grand mean of processing time of IK2 with g_1 thresholding: 701.03 µs; grand mean of processing time of INISg1 with g_1 thresholding: 366.74 µs.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
IK2	1.00	2.11	2.42	2.56	2.63	2.68	2.71	2.74	2.76	2.77
Time	3502.08	3504.32	3504.00	3502.24	3502.24	3503.68	3504.64	3501.12	3502.56	3504.00
(µs)	±15.72	±22.51	±19.12	±21.46	±21.76	±18.98	±21.00	±20.17	±21.91	±17.93
INISg1	1.00	1.72	1.86	1.92	1.94	1.96	1.98	1.99	1.99	2.00
Time	2853.00	2856.16	2853.44	2855.44	2854.68	2856.76	2852.76	2856.24	2857.56	2855.52
(µs)	±14.58	±21.77	±15.00	±18.57	±18.99	±19.56	±14.92	±17.65	±20.14	±17.60
IK2 with g_1 thresholding	1	7.72	14.57	21.06	28.70	34.91	44.49	49.89	58.80	65.51
Time	5177.72	577.48	315.36	251.40	150.28	149.56	107.00	97.88	92.12	91.52
(µs)	±11.76	±11.83	±11.82	±11.89	±11.04	±10.57	±9.94	±9.37	±9.29	±9.09
INISg1 with g_1 thresholding	1.00	10.48	20.06	29.11	39.84	48.48	61.90	69.43	81.88	91.25
Time	5089.84	566.80	308.32	246.00	146.48	146.64	103.04	95.60	89.12	89.44
(µs)	±21.17	±15.25	±13.03	±12.10	±10.53	±10.86	±9.06	±8.82	±8.78	±9.35

Table 8
Summary of assigned flows for g_2 curves, and estimated flow and calculation time of IK2 and INISg1 without (not explicitly indicated in the table) and with g_1 thresholding performed on Portenta H7. Except for assigned flows, only relative values are shown for estimated flows through IK2 and INISg1. Time: (mean) ± (SD). Grand mean of processing time of IK2 without g_1 thresholding: 605.67 µs; grand mean of processing time of INISg1 without g_1 thresholding: 494.95 µs; grand mean of processing time of IK2 with g_1 thresholding: 80.77 µs; grand mean of processing time of INISg1 with g_1 thresholding: 67.29 µs.

Assigned flow (cm^2/s)	**5.00×10^{-10}**	**5.45×10^{-9}**	**1.05×10^{-8}**	**1.55×10^{-8}**	**2.05×10^{-8}**	**2.55×10^{-8}**	**3.05×10^{-8}**	**3.54×10^{-8}**	**4.04×10^{-8}**	**4.54×10^{-8}**
Relative	1	10.90	20.90	30.90	40.90	50.90	60.90	70.90	80.90	90.90
IK2	1.00	2.11	2.42	2.56	2.63	2.68	2.71	2.74	2.76	2.77
Time	605.80	605.60	605.72	605.72	605.64	605.64	605.84	605.56	605.40	605.76
(µs)	±3.53	±3.38	±3.61	±3.75	±3.67	±3.25	±3.45	±3.41	±3.86	±3.44
INISg1	1.00	1.72	1.86	1.92	1.94	1.96	1.98	1.99	1.99	2.00
Time	494.86	495.00	494.90	494.94	495.04	494.88	494.96	494.86	495.06	495.04
(µs)	±3.06	±3.66	±3.56	±3.62	±3.59	±3.65	±2.57	±3.63	±2.25	±3.49
IK2 with g_1 thresholding	1	7.72	14.57	21.06	28.70	34.91	44.49	49.89	58.80	65.51
Time	597.92	65.44	35.84	28.48	17.12	17.44	12.32	11.36	10.56	11.20
(µs)	±3.90	±3.10	±4.35	±4.01	±2.80	±3.10	±4.03	±3.99	±3.77	±3.96
INISg1 with g_1 thresholding	1.00	10.48	20.06	29.11	39.84	48.48	61.90	69.43	81.88	91.25
Time	494.64	55.68	30.32	24.24	14.72	14.88	10.36	9.64	9.08	9.32
(µs)	±3.27	±1.78	±2.11	±1.60	±1.84	±1.62	±1.44	±1.60	±1.29	±1.70

without g_1 thresholding, and IK2 without g_1 thresholding, as discussed in Section 3.5. Among the three Arduino boards, Arduino Due showed the slowest performance, resulting in grand mean processing times of 7547.82 µs, 8973.40 µs, 63.90 ms, and 71.97 ms for INISg1 with g_1 thresholding, IK2 with g_1 thresholding, INISg1 without g_1 thresholding, and IK2 without g_1 thresholding, respectively. The Arduino Nano 33 BLE Sense board performed moderately, resulting in grand mean processing times of 366.74, 701.03, 2855.16, and 3503.09 µs for INISg1 with g_1 thresholding, IK2 with g_1 thresholding, INISg1 without g_1 thresholding, and IK2 without g_1 thresholding, respectively. Note that, despite a lower clock speed (64 MHz vs. 84 MHz), Arduino Nano 33 BLE Sense performed better than Arduino Due because the microprocessor of the former has a floating-point unit (FPU), which is absent from the latter. In the meantime, due to high clock speed (480 MHz) and the existence of FPU, Portenta H7 showed the best performance, resulting in grand mean processing times of 67.29, 80.77, 494.95, and 605.67 µs for INISg1 with g_1 thresholding, IK2 with g_1 thresholding, INISg1 without g_1 thresholding, and IK2 without g_1 thresholding, respectively. Overall, the results show that the suggested methods can be used to process even high-speed flow data on low-cost microcontrollers.

4. Discussion

Among the suggested approaches, INISg1 with g_1 thresholding may offer various advantages, such as:

(i) reduced signal processing time on a PC (Section 3.5),
(ii) outcomes comparable to those of NF (Sections 3.1 to 3.4, and 3.6),
(iii) fast signal processing on low-cost microprocessors (Section 3.7), and
(iv) system cost reduction.

Such advantages will expand the utilization of DCS techniques by allowing for real-time, normal, and fast flow measurement, and the implementation of more miniaturized, low-cost DCS systems by replacing PC or more bulky hardware with small, low-cost microcontrollers such as Arduino or Raspberry Pi. A previous research has utilized an electric-circuit-based integrator to directly integrate analog amplitude signals detected using a photodiode [11]. However, the flow information acquired in that work was not robust and may require additional work to make the concept more mature.

When g_1 thresholding was not applied, INISg1 performed better than IK2 in the *in vivo* experiment (Section 3.4), while it was the opposite in the phantom experiment (Section 3.3). This is because of the difference in the range of flows in the two experiments. In particular, the flow in the phantom experiment ranged from 1.17×10^{-8} cm^2/s to 2.04×10^{-8} cm^2/s, which was around or higher than the TV. Conversely, the range of flow in the *in vivo* experiment was from 8.10×10^{-11} cm^2/s to 6.87×10^{-9} cm^2/s, which was below the TV. As shown in the simulation, without g_1 thresholding, INISg1 performed better than IK2 until TV, and IK2 performed better than INISg1 after TV. Thus, this discrepancy is as expected. For simplicity, the TV from the simulation was used to analyze the discrepancy between the IK2 and INISg1 results before applying g_1 thresholding. However, TV may vary based on the optical properties of the medium of interest.

Before thoroughly testing IK2 and INISg1, several preliminary tests were performed. First, the same calculation methods of IK2 and INISg1 were tested on g_2 curves. However, the g_2 curve-based estimation of blood flow was not as robust as that of IK2 and INISg1 (data not shown). Second, β other than 0.45 were tested (data not shown). A few-mode fiber (FMF) is used in some cases to increase the light collection for deep tissue measurements, such as brain measurement [31]. Because β is inversely proportional to the number of modes propagated through an optical fiber, using FMF results in a smaller β than using SMF by allowing more modes to propagate through an optical fiber. Moreover, because IK2 and INISg1 perform well with g_1 with different β as well, the wide applicability of IK2 and INISg1 can be guaranteed. Third, in addition to IK2 and INISg1, the inverse of the numerical integration of g_1 (INIg1) was also tested, which resulted in significant underestimation of flow that could not be recovered even after applying g_1 thresholding (data not shown). Thus, INIg1 was not tested in this study. However, INIg1 may still be useful in applications that require simplified calculation and monitoring of qualitative flow changes. Finally, different source-detector separations were also tested, and it was confirmed that IK2 and INISg1 still work with source-detector separations other than 1 cm (shown in appendix). Overall, it was confirmed that IK2 and INISg1 work under various conditions, guaranteeing their versatility.

For simplicity, except for NF, β was assumed to be constant in flow estimation. While this assumption worked seamlessly in this study, unexpected situations, including the change of ambient light or probe contact to the tissue, may be able to cause significant change of β so that the assumption of constant β may not be able to work properly. In such cases, β can be extracted using a single step β fitting. If an application still requires relatively fast signal processing, subtraction of $g_2(\tau_{\text{end}})$ from g_2 (τ_1) can be performed as an alternative shortcut for calculating β.

In the Arduino calculation, compared with the results from the PC, without g_1 thresholding, IK2 and INISg1 had a more significant underestimation of the flow. This is mainly due to the limited precision of floating-point ('*single*' data type in MATLAB: 32-bit precision floating-point vs. '*float*' data type in Arduino: 6–7 decimal digits precision floating point). In advanced Arduino products like the ones used in this study, '*double*' data type offers more precision in floating-point, but it slows down signal processing time. When g_1 thresholding was applied to IK2 and INISg1, the relative flows from the three types of Arduino became more comparable to the ones from the PC (e.g., When assigned flow is 5.45×10^{-9} cm^2/s, INISg1 without g_1 thresholding→INISg1 with g_1 thresholding: 6.65→10.48 from the PC and 1.72→10.48 from Arduino). Thus, g_1 thresholding combined with INISg1 allows the use of such microprocessors for high-speed DCS signal processing with satisfactory accuracy without using complicated, expensive signal processing hardware such as a graphics processing unit (GPU).

While direct comparison of processing with other literature is not straightforward due to the use of different processing units and experimental conditions, IK2 with g_1 thresholding and INISg1 with g_1 thresholding on the PC (12.58 µs and 12.20 µs) performed approximately 37 and 38 times faster than a DL-based approach (0.46 ms) [10], respectively, which was one of the fastest among the previous works. Meanwhile, on Portenta H7 IK2 with g_1 thresholding (80.77 µs) and INISg1 with g_1 thresholding (67.29 µs) performed approximately 6 and 7 times faster, respectively, than the DL-based approach. Moreover, based on our results, the higher the flow of g_2 curves being processed, the faster the suggested approaches process the data. It is noteworthy that the DL-based approach uses a GPU that allows parallel computing. If our methods were to be implemented on a GPU, we would expect a much faster processing speed than that of the CPU or Arduino by virtue of parallel processing. Ultimately, with g_1 thresholding, IK2 and INISg1 can be useful in reducing the signal processing time in DCS. Although the suggested technique was not tested for applications that require faster signal processing time, g_2 with a shorter maximum delay time (e.g., $T = 1$ ms) can also be considered.

When processing a g_2 with a flow higher than 3.05×10^{-8} cm^2/s by using INISg1 with g_1 thresholding, Portenta H7 (dual-core with 480 MHz and 240 MHz clock speed) and the PC (octa-core

with 2.90 GHz base clock speed and 4.80 GHz turbo clock speed when the octa-core is used as a single-core) showed a large difference in their performances, with Portenta H7 showing superior results, even though the PC is equipped with a better processing unit. This is primarily because Portenta H7 is solely dedicated to numerical integration, while the CPU of the PC is not [12]. When software that can utilize the turbo clock speed of the CPU is used, the performance of the PC may be significantly improved. Still, if the size, cost, and energy usage are the most important factors considered, Portenta H7 will be more advantageous than the PC. Depending on the application, Arduino Nano 33 BLE Sense (or Arduino Nano 33 BLE) can be used considering the cost of the system (the cheapest among Arduino products tested in this work) and its moderate signal processing speed.

Recently, a group introduced a mathematical method based on the Volterra integral equation (VIE) to recover g_1 from multi-exposure diffuse speckle contrast signals measured using camera-based low-cost deep and shallow blood flow measurement systems [32,33]. In the meantime, our integration-based approach is proposed to reduce the computational burden of quantifying blood flow from g_1 of DCS. Although our method and VIE-based method aim different goals, both methods are helpful in reducing the cost of DCS-derived systems. The recovered g_1 from the VIE-method is known to have ripples in long delay times, and can be used to extract blood flow using various approaches. Thus, we expect that our approach can not only be used in conventional DCS but also in DCS-derived systems, including systems that utilize the VIE-based method to recover g_1 data.

Cerebral autoregulation (CA) is a widely known mechanism that maintains the cerebral blood flow during the variation of the blood pressure within a certain range [34–38]. Impairment of CA happens in various types of brain diseases, including traumatic brain injury, stroke, and subarachnoid hemorrhage, and may be able to cause secondary brain damage [35,36,38], thus monitoring of CA is of highly importance. Due to multiple advantages, including noninvasiveness and relatively high temporal resolution, DCS combined with either invasive (e.g., thermal diffusion invasive probe) [35] or noninvasive (e.g., finger plethysmography) [36,38] blood pressure measurement device is frequently used to monitor dynamic CA. While the process of CA itself is not a fast mechanism, to monitor dynamic CA, fast CBF and blood pressure measurements are used to see the change of a phase relationship between the pulsatile CBF and blood pressure. However, the relatively high cost of DCS system and the requirement of postprocessing may be able to limit the usage of DCS in CA monitoring. As demonstrated in Section 3.6, the suggested algorithms can be used to process fast blood flow data. The capability of processing fast flow can allow wide applications of the suggested algorithms, including real-time data processing and display during monitoring of CA. By reducing the size and cost of DCS system, and allowing real-time flow data display, we believe that our algorithms will be able to reduce the hindrance of DCS in dynamic CA monitoring, especially when used in regions with relatively limited resources.

While we thoroughly evaluated the suggested methods, there are some limitations in this study. First, we did not compare the outcomes of the suggested approach when using different numerical integration methods such as Riemann sum and trapezoid rule. Comparison of different numerical integration methods for IK2 and INISg1 will be rigorously performed in a follow-up study. Second, even though we showed that with g_1 thresholding, INISg1 performs better than IK2, we could not identify the underlying reasons. In practice, as the flow varies, the values of g_1 with longer delay times vary more dynamically than those with shorter delay times. Therefore, the existence of the delay time-weighted calculation using $(1-\frac{\tau}{T})$ in IK2 could weaken the effect of g_1 values of long delay times. In future, a follow-up study will be performed to comprehensively investigate the reasons for INISg1 with g_1 thresholding performing better than IK2 with g_1 thresholding.

5. Conclusion

In conclusion, IK2 and INISg1, numerical integration-based approaches, were proposed for estimating the flow information in DCS. In preliminary experiments (data not shown), both IK2 and INISg1 suffered from a significant underestimation of the flows. To overcome the underestimation found in the preliminary experiments, we further introduced g_1 thresholding, which dramatically recovered the flow information estimated by IK2 and INISg1. Interestingly, INISg1, a more simplified method than IK2, performed much better than IK2 when g_1 thresholding was applied. In addition, INISg1 with g_1 thresholding showed the fastest performance in processing a g_2 curve by effectively reducing the calculation burden. Moreover, INISg1 with g_1 thing was even faster than the DL-based signal processing approach, which is one of the fastest methods, and is also applicable to quantification of the flow on a PC and Arduino (Sections 3.5 and 3.7). Thus, we believe that INISg1 with g_1 thresholding, one of the newly suggested approaches, can contribute to the further development of DCS technologies in terms of reducing signal processing time and system miniaturization because it is computationally light, offering comparable flow estimation as NF.

Declaration of Competing Interest

The authors have no conflicts to disclose.

Acknowledgment

This research was supported by the Healthcare AI Convergence Research & Development Program through the National IT Industry Promotion Agency of Korea (NIPA), funded by the Ministry of Science and ICT [grant number s1610–20–1016]; the Korea Health Technology R&D Project through the Korea Health Industry Development Institute (KHIDI), funded by the Ministry of Health & Welfare, Republic of Korea [grant number HI18C2383]; "GIST Research Institute (GRI) IIBR" grant funded by the GIST in 2022; the DGIST Creative Challenging Research Grant [grant number 21-BRP-11]; and the Nantong University Scientific Research Foundation for Introduced Talents [grant number 135421629029]. We would like to thank Editage for English language editing.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cmpb.2022.106933.

References

[1] D.A. Boas, A.G. Yodh, Spatially varying dynamical properties of turbid media probed with diffusing temporal light correlation, J. Opt. Soc. Am. A 14 (1997) 192–215.
[2] M.N. Kim, T. Durduran, S. Frangos, B.L. Edlow, E.M. Buckley, H.E. Moss, et al., Noninvasive measurement of cerebral blood flow and blood oxygenation using near-infrared and diffuse correlation spectroscopies in critically brain-injured adults, Neurointensive Care 12 (2010) 173–180.
[3] C. Cheung, J.P. Culver, K. Takahashi, J.H. Greenberg, A. Yodh, *In vivo* cerebrovascular measurement combining diffuse near-infrared absorption and correlation spectroscopies, Phys. Med. Biol. 46 (2001) 2053.
[4] G. Yu, T.F. Floyd, T. Durduran, C. Zhou, J. Wang, J.A. Detre, et al., Validation of diffuse correlation spectroscopy for muscle blood flow with concurrent arterial spin labeled perfusion MRI, Opt. Express 15 (2007) 1064–1075.
[5] Y. Shang, T. Li, G. Yu, Clinical applications of near-infrared diffuse correlation spectroscopy and tomography for tissue blood flow monitoring and imaging, Physiol. Meas. 38 (2017) R1.
[6] Y. Shang, Y. Zhao, R. Cheng, L. Dong, D. Irwin, G. Yu, Portable optical tissue flow oximeter based on diffuse correlation spectroscopy, Opt. Lett. 34 (2009) 3556–3558.

[7] C. Huang, M. Seong, J.P. Morgan, S. Mazdeyasna, J.G. Kim, J.T. Hastings, et al., Low-cost compact diffuse speckle contrast flowmeter using small laser diode and bare charge-coupled-device, J. Biomed. Opt. 21 (2016) 080501.
[8] S.Y. Lee, J.M. Pakela, M.C. Helton, K. Vishwanath, Y.G. Chung, N.J. Kolodziejski, et al., Compact dual-mode diffuse optical system for blood perfusion monitoring in a porcine model of microvascular tissue flaps, J. Biomed. Opt. 22 (2017) 121609.
[9] W. Liu, R. Qian, S. Xu, P. Chandra Konda, J. Jönsson, M. Harfouche, et al., Fast and sensitive diffuse correlation spectroscopy with highly parallelized single photon detection, APL Photonics 6 (2021) 026106.
[10] C.S. Poon, F. Long, U. Sunar, Deep learning model for ultrafast quantification of blood flow in diffuse correlation spectroscopy, Biomed. Opt. Express 11 (2020) 5557–5564.
[11] A. Biswas, A.B. Parthasarathy, An integrated detection scheme for fast, embedded measurement of deep tissue blood flow with diffuse correlation spectroscopy, Opt. Tomogr. Spectrosc. Opt. Soc. Am. (2020) SM3D–SM35.
[12] W. Lin, D.R. Busch, C.C. Goh, J. Barsi, T.F. Floyd, Diffuse correlation spectroscopy analysis implemented on a field programmable gate array, IEEE Access. 7 (2019) 122503–122512.
[13] W.B. Baker, A.B. Parthasarathy, D.R. Busch, R.C. Mesquita, J.H. Greenberg, A. Yodh, Modified beer-lambert law for blood flow, Biomed. Opt. Express 5 (2014) 4053–4075.
[14] J. Dong, R. Bi, J.H. Ho, K. Lee, P.S. Thong, K.C. Soo, Diffuse correlation spectroscopy with a fast Fourier transform-based software autocorrelator, J. Biomed. Opt. 17 (2012) 097004.
[15] D. Wang, A.B. Parthasarathy, W.B. Baker, K. Gannon, V. Kavuri, T. Ko, et al., Fast blood flow monitoring in deep tissues with real-time software correlators, Biomed. Opt. Express 7 (2016) 776–797.
[16] L. Dong, L. He, Y. Lin, Y. Shang, G. Yu, Simultaneously extracting multiple parameters via fitting one single autocorrelation function curve in diffuse correlation spectroscopy, IEEE Trans. Biomed. Eng. 60 (2013) 361–368.
[17] Z. Li, Q. Ge, J. Feng, K. Jia, J. Zhao, Quantification of blood flow index in diffuse correlation spectroscopy using long short-term memory architecture, Biomed. Opt. Express 12 (2021) 4131–4146.
[18] M. Seong, Z. Phillips, P.M. Mai, C. Yeo, C. Song, K. Lee, et al., Simultaneous blood flow and blood oxygenation measurements using a combination of diffuse speckle contrast analysis and near-infrared spectroscopy, J. Biomed. Opt. 21 (2016) 027001.
[19] R. Bi, J. Dong, K. Lee, Deep tissue flowmetry based on diffuse speckle contrast analysis, Opt. Lett. 38 (2013) 1401–1403.
[20] L. He, Y. Lin, Y. Shang, B.J. Shelton, G. Yu, Using optical fibers with different modes to improve the signal-to-noise ratio of diffuse correlation spectroscopy flow-oximeter measurements, J. Biomed. Opt. 18 (2013) 037001.
[21] D. Giavarina, Understanding bland altman analysis, Biochem. Med. 25 (2015) 141–151 (Zagreb).
[22] Y. Shang, T. Li, L. Chen, Y. Lin, M. Toborek, G. Yu, Extraction of diffuse correlation spectroscopy flow index by integration of N th-order linear model with Monte Carlo simulation, Appl. Phys. Lett. 104 (2014) 193703.
[23] H.J. van Staveren, C.J.M. Moes, J. van Marie, S.A. Prahl, M.J.C. van Gemert, Light scattering in Intralipid-10% in the wavelength range of 400-1100 nm, Appl. Opt. 30 (1991) 4507–4514.
[24] G.M. Hale, M.R. Querry, Optical constants of water in the 200-nm to 200-μm wavelength region, Appl. Opt. 12 (1973) 555–563.
[25] M.A. Kimm, S. Tzoumas, S. Glasl, M. Omar, P. Symvoulidis, I. Olefir, et al., Longitudinal imaging of T cell-based immunotherapy with multi-spectral, multi-scale optoacoustic tomography, Sci. Rep. 10 (2020) 4903.
[26] W. Zhou, O. Kholiqov, S.P. Chong, V.J. Srinivasan, Highly parallel, interferometric diffusing wave spectroscopy for monitoring cerebral blood flow dynamics, Optica 5 (2018) 518–527.
[27] G. Hong, S. Diao, J. Chang, A.L. Antaris, C. Chen, B. Zhang, et al., Through-skull fluorescence imaging of the brain in a new near-infrared window, Nat. Photon 8 (2014) 723–730, doi:10.1038/nphoton.2014.166.
[28] S.E. Boebinger, R.O. Brothers, S. Bong, B. Sanders, C. McCracken, L.H. Ting, et al., Diffuse optical spectroscopy assessment of resting oxygen metabolism in the leg musculature, Metabolites 11 (2021) 496.
[29] V.A. Sposito, Minimizing the maximum absolute deviation, SIGMAP Bull. (1976) 51–53.
[30] R. Bi, Y. Du, G. Singh, J.H. Ho, S. Zhang, A.B. Ebrahim Attia, et al., Fast pulsatile blood flow measurement in deep tissue through a multimode detection fiber, J. Biomed. Opt. 25 (2020) 055003.
[31] K. Verdecchia, M. Diop, T.Y. Lee, K.S. Lawrence, Quantifying the cerebral metabolic rate of oxygen by combining diffuse correlation spectroscopy and time-resolved near-infrared spectroscopy, J. Biomed. Opt. 18 (2013) 027007.
[32] K. Murali, A.K. Nandakumaran, T. Durduran, H.M. Varma, Recovery of the diffuse correlation spectroscopy data-type from speckle contrast measurements: towards low-cost, deep-tissue blood flow measurements, Biomed. Opt. Express 10 (2019) 5395–5413.
[33] K. Murali, A.K. Nandakumaran, H.M. Varma, On the equivalence of speckle contrast-based and diffuse correlation spectroscopy methods in measuring *in vivo* blood flow, Opt. Lett. 45 (2020) 3993–3996.
[34] A.H. van Beek, J.A. Claassen, M.G.O. Rikkert, R.W. Jansen, Cerebral autoregulation: an overview of current concepts and methodology with special focus on the elderly, J. Cereb. Blood Flow Metab. 28 (2008) 1071–1085.
[35] (Ivy) J. Selb, K.C. Wu, J. Sutin, P.Y. Lin, P. Farzam, S. Bechek, et al., Prolonged monitoring of cerebral blood flow and autoregulation with diffuse correlation spectroscopy in neurocritical care patients, Neurophotonics 5 (2018) 045005.
[36] A.B. Parthasarathy, K.P. Gannon, W.B. Baker, C.G. Favilla, R. Balu, S.E. Kasner, et al., Dynamic autoregulation of cerebral blood flow measured non-invasively with fast diffuse correlation spectroscopy, J. Cereb. Blood Flow Metab. 38 (2018) 230–240.
[37] K.C. Wu, J. Sunwoo, F. Sheriff, P. Farzam, P.Y. Farzam, F. Orihuela-Espina, et al., Validation of diffuse correlation spectroscopy measures of critical closing pressure against transcranial Doppler ultrasound in stroke patients, J. Biomed. Opt. 26 (2021) 036008.
[38] A.A. Bahrani, W. Kong, Y. Shang, C. Huang, C.D. Smith, D.K. Powell, et al., Diffuse optical assessment of cerebral-autoregulation in older adults stratified by cerebrovascular risk, J. Biophotonics 13 (2020) e202000073.