Iran J Sci Technol Trans Sci (2021) 45:1043–1056
https://doi.org/10.1007/s40995-021-01079-3

RESEARCH PAPER

An Efficient Method for Fitting Gaussian Functions

Minkyu Kwak[1] · Bataa Lkhagvasuren[1] · Xiuxiu Sun[1]

Received: 23 June 2020 / Accepted: 21 January 2021 / Published online: 15 February 2021

Abstract
We propose a simple differential method and a refined Roonizi's method to fit single Gaussian function and integro-differential method to fit two Gaussian functions. The conventional method for fitting Gaussian functions is an iterative procedure. The proposed methods, being linear in estimated parameters, alleviate the problem of critical initial guess needed in iterative procedures. The experimental results confirm the methods perform better or in a competitive manner compared to Caruana's, Guo's, Roonizi's and FAS algorithm for single Gaussian case. It is found that the proposed integro-differential method to fit two Gaussian functions is stable and identifies Gaussian parameters in an efficient way.

Keywords Gaussian functions · Nonlinear fitting · Parameter estimation

1 Introduction

Due to the central limit theorem, Gaussian functions are very useful for explaining many phenomena in science and engineering problems and, thus, fitting a Gaussian function to the observed data points appears often. To mention a few among many other applications of this classical problem, Gaussian fitting can be used in spectroscopy to accurately locate the wavelength of a doppler-broadened emission or absorption line and to estimate line strength (Ireland 2005; Lenz and Ayres 1992) and Winick (1986). Gaussian fitting techniques can also find applications in sensing of object positions which are blurred by imaging processes (Bobroff 1986). Two-dimensional Gaussian fitting is even more widely used in fields such as wave front sensing, Gaussian beam characterization and bio-imaging. Obtaining the most accurate possible estimate of the parameters is the main goal of this problem.

✉ Bataa Lkhagvasuren
bataa@chonnam.ac.kr

Minkyu Kwak
mkkwak@chonnam.ac.kr

Xiuxiu Sun
sunxiuxiushuxue@163.com

[1] Mathematics Department, Chonnam National University, Gwangju, South Korea

Because determining the parameters of Gaussian curve is associated with the solution of overdetermined system of nonlinear equations, the Gaussian curve fitting is difficult problem. The conventional solution to this type of problem is to employ an iterative procedure like Newton–Raphson algorithm. These methods require sufficiently good initial guess and it is possible that the procedure does not converge to the true solution. Another methods (Guo 2011) and Caruana et al. (1986), which exploit the fact that the natural logarithm of Gaussian function produces quadratic function in unknown parameters, avoids the problem of good initial guess. The method proposed by Kheirati Roonizi (2013) performs better compared to the methods presented in Guo (2011) and Caruana et al. (1986). However, Roonizi's original method is subject to accumulated noise error from numerical integration process. The method FAS reported in Al-Nahhal et al. (2019) computes the coefficient σ (b in our notation) directly by exploiting the property of Gaussian function. That property the FAS relies on still heavily depends on the numerical integration process and accumulated noise error makes the estimation task to fail in some setting. Moreover, for the Gaussian long tail, these log-based methods (Guo 2011; Caruana et al. 1986; Al-Nahhal et al. 2019) suffer from magnified errors originated from near-zero values. Thus, in order to achieve convergence, Caruana et al. (1986) purposed iterative procedure which is also used in Al-Nahhal et al. (2019). Firstly, in this paper, we suggest refined Roonizi's method (ID-1) and simple differential method (DF) to fit single

Gaussian function. Our methods outperform the conventional ones in certain situations and are simple yet. Extensive numerical testing were performed to validate our proposed methods. Secondly, we present the integro-differential method to extract parameters from two Gaussian functions (in the sense of sum of two Gaussians). To the best of our knowledge, this is the first instance to discuss the linearization of estimation task for two Gaussians.

The rest of the paper is organized as follows. In Sect. 2, we briefly discuss the previous methods and present our proposed methods. The corresponding numerical experiments are given in Sect. 3 and complexity comparison made in Sect. 4. Finally, in Sect. 5, the fitting problem of two Gaussians is addressed.

2 Curve Fitting of Single Gaussian Function

Recall that a Gaussian function has the form

$$m = ae^{-c(t-b)^2}. \tag{1}$$

The graph of the Gaussian function is bell-shaped and centered at $t = b$ with a being the height of the peak. The parameter c controls the width of the "bell." This type of problem is usually solved by Newton method, for which a sufficiently good initial data is needed to obtain correct values. It is possible that Newton method is trapped at local solution. Thus, more reliable and efficient methods are needed. In the following subsections, we explain briefly up-to-date such methods and give our proposed methods.

2.1 Previous Methods

By exploiting the fact that Gaussian is the exponential of a quadratic function of t, Caruana et al. (1986) proposed a simple method to estimate the parameters. Caruana's method finds the natural logarithm of the data and fits the resulting values to a parabola. That is

$$\begin{aligned} \ln(m) &= -ct^2 + 2bct + \ln(a) - cb^2 \\ &= z_1 t^2 + z_2 t + z_3, \end{aligned} \tag{2}$$

where

$$z_1 = -c, z_2 = 2bc, z_3 = \ln(a) - cb^2. \tag{3}$$

Note that, for $\ln(m)$ in (2), only positive values should be considered. Caruana's method is numerically efficient, since it is based on linear least square estimation and the guess of initial data is, therefore, not needed. However, the noise could be magnified by the logarithmic operation, especially, for the points having small Gaussian values. Therefore, the accuracy of Caruana's method may be decreased dramatically in the presence of noise.

Guo addressed the problem of Caruana's method in Guo (2011), where he analyzed the effect of noise and derived a simple and improved technique. Briefly, Guo's method can be obtained by multiplying (2) by m and writing it in the form

$$\begin{aligned} m\ln(m) &= -cmt^2 + 2bcmt + m(\ln(a) - cb^2) \\ &= z_1 mt^2 + z_2 mt + z_3 m, \end{aligned} \tag{4}$$

where

$$z_1 = -c, z_2 = 2bc, z_3 = \ln(a) - cb^2. \tag{5}$$

Guo's algorithm performs better than Caruana's method; however, it still suffers from deterioration of estimation accuracy if Gaussian long tails are considered. To improve the accuracy for Gaussian long tails, he devised an iterative procedure, which can be written in a compact form as

$$m^{j-1}\ln(m) = z_1^j m^{j-1} t^2 + z_2^j m^{j-1} t + z_3^j m^{j-1}, \tag{6}$$

where

$$z_1^j = -c^j, z_2^j = 2b^j c^j, z_3^j = \ln(a^j) - c^j (b^j)^2 \tag{7}$$

and $m^j = m$ for $j = 0$ and $m^j = a^j e^{-c^j (t-b^j)^2}$ for $j > 0$ with superscripts denoting iteration indices.

Another method by Kheirati Roonizi (2013), which uses differentiation and integration, requires no iteration procedure to increase accuracy (but, iteration can be created if needed). Although Roonizi's method is designed to fit Gaussian functions riding on polynomials, let us explain the main idea for Gaussian function. The first step of Roonizi's method is to differentiate the both sides of (1) and obtain equation

$$m' = -2ctm + 2cbm. \tag{8}$$

In the next step, both sides of Eq. (8) are integrated on the interval $[t, t']$ and it yields formally equation

$$m(t') - m(t) = -2c \int_t^{t'} \tau m(\tau) d\tau + 2cb \int_t^{t'} m(\tau) d\tau. \tag{9}$$

Now, by using numerical integration, we can invoke the linear least square estimation method for (9) and find b and c. Solving another simple least square problem, a is found from (1). For Gaussian long tails, Roonizi's method is better than Caruana's and Guo's method, since the troublesome logarithm term is disappeared. The only problem Roonizi's method might have the error accumulation of noise during the numerical integration in (9). We will address this problem in subsequent sections.

The authors in Al-Nahhal et al. (2019) introduced a novel, fast, accurate and separable (FAS) algorithm for estimating Gaussian function parameters. The first step of

FAS is to estimate directly the parameter c by exploiting the property of Gaussian integral:

$$\int_{-\infty}^{\infty} a e^{-c(t-b)^2} = a\sqrt{\frac{\pi}{c}}. \tag{10}$$

Suppose the observed data $m_k = m(t_k)$ is sampled over the mesh $t_0, t_1, \ldots, t_n$. Replacing the amplitude a by the maximum value of observed values $m_{\max} = \max_{k=0\ldots n} m_k$, the parameter c is estimated by

$$\hat{c} = \left[\frac{m_{max}\sqrt{\pi}}{\sum_{k=1}^{n} \Delta t_k m_k}\right]^2, \tag{11}$$

where $\Delta t_k = t_k - t_{k-1}$. In the next step, using the estimated value $\hat{c}$, the other parameters are estimated similarly with Guo's method as

$$\begin{aligned} m\ln(m) + \hat{c}mt^2 &= 2bcmt + m(\ln(a) - cb^2) \\ &= z_1 mt + z_2 m, \end{aligned} \tag{12}$$

where

$$z_1 = 2b\hat{c}, z_2 = \ln(a) - \hat{c}b^2. \tag{13}$$

The FAS method is accurate and fast to estimate the parameter c. Extensive accuracy and complexity comparisons were given in Al-Nahhal et al. (2019). However, for Gaussian long tails, the FAS method needs iterative procedure, since it still involves logarithmic term. Similarly with Guo's algorithm, the iterative version of (12) and (13) can be formulated as

$$m^{j-1}\ln(m) + \hat{c}m^{j-1}t^2 = z_1^j m^{j-1} t + z_2^j m^{j-1}, \tag{14}$$

where

$$z_1^j = 2b^j\hat{c}, z_2^j = \ln(a^j) - \hat{c}(b^j)^2. \tag{15}$$

and $m^j = m$ for $j = 0$ and $m^j = a^j e^{-\hat{c}(t-b^j)^2}$ for $j > 0$.

2.2 Proposed Methods

After performing extensive numerical testing, we suggest two methods to fit Gaussian functions. The first method, which is a refined version of Roonizi's method, takes the integration domain into account. Let us call the estimation algorithm (9) as "Integro-Differential" method and denote by ID for brevity. As mentioned before, the integration process may accumulate errors caused by noise. By choosing shorter integration interval, the estimation accuracy can be improved. In order to verify the dependence of ID method's accuracy on its integration interval, we execute two versions of it. The first proposed method, denoted by ID-1, uses $[t, t'] = [t_{k-1}, t_k]$ as integration interval for $k = 1, \ldots, n$ and has the form

$$m_k - m_{k-1} = -2c\int_{t_{k-1}}^{t_k} \tau m(\tau)d\tau + 2cb\int_{t_{k-1}}^{t_k} m(\tau)d\tau, \tag{16}$$

where the integrals are approximated by the trapezoidal rule for integral as follows

$$\begin{aligned} \int_{t_{k-1}}^{t_k} \tau m(\tau)d\tau &\approx \Delta t_k \frac{t_{k-1}m_{k-1} + t_k m_k}{2}, \\ \int_{t_{k-1}}^{t_k} m(\tau)d\tau &\approx \Delta t_k \frac{m_{k-1} + m_k}{2} \end{aligned} \tag{17}$$

for $k = 1, \ldots, n$.

The second version, denoted by ID-2, takes $[t, t'] = [t_0, t_k]$ as integration interval for $k = 1, \ldots, n$ and is written as

$$m_k - m_0 = -2c\int_{t_0}^{t_k} \tau m(\tau)d\tau + 2cb\int_{t_0}^{t_k} m(\tau)d\tau, \tag{18}$$

where the integrals are calculated by

$$\begin{aligned} \int_{t_0}^{t_k} \tau m(\tau)d\tau &\approx \sum_{i=1}^{k} \Delta t_i \frac{t_{i-1}m_{i-1} + t_i m_i}{2}, \\ \int_{t_0}^{t_k} m(\tau)d\tau &\approx \sum_{i=1}^{k} \Delta t_i \frac{m_{i-1} + m_i}{2} \end{aligned} \tag{19}$$

for $k = 1, \ldots, n$. ID-2 can be identified as ordinary Roonizi's method. Having the shortest integration domain, ID-1 is expected to have better accuracy.

We propose another simple method for single Gaussian fitting problem. Our idea is to apply the trapezoidal rule to Eq. (8). The trapezoidal rule approximates signals at points between samples:

$$\begin{aligned} m\left(\frac{t_{k-1} + t_k}{2}\right) &= \frac{m(t_k) + m(t_{k-1})}{2}, \\ m'\left(\frac{t_{k-1} + t_k}{2}\right) &= \frac{m(t_k) - m(t_{k-1})}{\Delta t_k}. \end{aligned} \tag{20}$$

In signal processing, the bilinear transform, which is a transformation from continuous time system (in the Laplace domain) to discrete time system (in the Z-domain) uses the trapezoidal rule. There are variety of other ways in which derivatives can be approximated by differences (e.g., forward or backward Euler methods); however, with the use of the bilinear transformation, stable continuous time system will always map to stable discrete time systems. We refer to Oppenheim and Young (1983) for detailed discussions.

Therefore, the trapezoidal rule can be used in the fitting problems with the same principle of system identifications. Applying the trapezoidal rule (20–8), the following discrete equations are obtained:

$$\begin{aligned} m_k - m_{k-1} &= z_1 \Delta t_k \left(\frac{t_{k-1}+t_k}{2}\right)(m_k + m_{k-1}) \\ &\quad + z_2 \Delta t_k (m_k + m_{k-1}) \end{aligned} \tag{21}$$

where

$$z_1 = -c, z_2 = cb. \tag{22}$$

Thus, after solving b and c from (21) and (22), a is found from (1). Let us call this estimation scheme as "Differential method" for Gaussian fitting and denote by DF. In order to compare the performances of ID-1, ID-2 and DF methods, we compare the noise variance in the coefficients of corresponding least square equations. Suppose the observation m_k at the point t_k contains a noise w_k, where w_k is an independent and identically distributed (i.i.d.) Gaussian random variables with the standard deviation (SD) σ and mean 0 for $k = 0, \ldots, n$. Without losing generality, let $t_k = k\Delta t$ for $k = 0, \ldots, n$. Table 1 shows the accumulated noise variance calculated for each term in ID-1, ID-2 and DF. The uncertainties for ID-1 and DF are comparable (in the same order); however, the uncertainty of integral terms for ID-2 is multiplied by factor k compared to other two. This subtle difference in accumulated variance distinguishes our proposed method ID-1 and DF from the others.

3 Numerical Testing for One Gaussian Fitting Problem

3.1 Numerical Testing Without Iteration

In this section, we compare the performances of previous methods; Caruana's (1986), Guo's (2011) and FAS (Al-Nahhal et al. 2019) with that of ID-1, ID-2 (Kheirati Roonizi 2013) and DF without iteration. Figure 1 shows the fitting results of the considered algorithms for three different sampling numbers $n = 50, n = 40$ and $n = 30$. The signal-to-noise ratio (SNR), defined by $SNR = a/\sigma$, is set to 20 and the theoretical Gaussian parameters are $a = 2, b = 10$ and $c = \frac{1}{8}$. Table 2 shows the average absolute errors (ARE) of fitted parameters $\widehat{a}, \widehat{b}, \widehat{c}$ and the root mean square error RMS of difference of fitted Gaussian $\widehat{m}$ and theoretical Gaussian $\overline{m}$ in a percentage, which are defined as:

$$\begin{aligned} ARE(a) &= \frac{|\widehat{a} - a|}{a} \times 100\%, \\ ARE(b) &= \frac{|\widehat{b} - b|}{b} \times 100\%, \\ ARE(c) &= \frac{|\widehat{c} - c|}{c} \times 100\%, \\ RMS(\widehat{m} - \overline{m}) &= \sqrt{\frac{\sum_{j=1}^{N} (\widehat{m}_j - \overline{m}_j)^2}{N}} \times 100\% \end{aligned} \tag{23}$$

with

$$\widehat{m}_j = \widehat{a} e^{-\widehat{c}(\tau_j - \widehat{b})^2}, \overline{m}_j = a e^{-c(\tau_j - b)^2} \tag{24}$$

and $\tau_1, \tau_2, ..., \tau_N$ is a mesh with length N. The number of Monte Carlo simulations is set to 5000 and RMS calculated with $N = 10n$.

For the simulation reported in Table 2, the method DF performs best for $RMS(\hat{m} - \overline{m})$ and $ARE(b)$, while ID-2 takes lead for $ARE(a)$ and FAS is best at $ARE(c)$. As expected ID-1 and DF perform almost equally. To check the dependence of the performance on SNR, the same simulation executed for SNR varying between 20 and 200 and the result is graphed in Fig. 2 (Fig. 3a for $a = 1$). It is clear that ID-1 and DF is best for most of the cases and the difference of their performance is very little. For $ARE(a)$, $ARE(c)$, ID-2 is slightly better at some lower values of SNR and FAS is also performing better for $ARE(c)$ starting from some level. This is because integral-based methods (ID-2, FAS) are expected to perform better than differential-based methods for higher noise level and fewer observation values, since accumulated noise error is not a problem. However, the situation is changed when the higher number of observations is considered. Inspired by the error analysis reported in Al-Nahhal et al. (2019), Monte Carlo simulations were performed for the varying parameters W and n, where W is the parameter controlling the length of observation interval $\left(b - \frac{W}{2\sqrt{2c}}, b + \frac{W}{2\sqrt{2c}}\right)$. The value $W = 6$ corresponds to the three-sigma-rule case, in which 99.73% of the sampled values from Gaussian lie within

Table 1 Accumulated noise variance

Method	Terms	Accumulated noise variance
ID-1	$m(t_k) - m(t_{k-1})$	$2\sigma^2$
	$\int_{t_{k-1}}^{t_k} \tau m(\tau) d\tau$	$\left(\frac{\sigma(\Delta t)^2}{2}\right)^2 \left((k-1)^2 + k^2\right)$
	$\int_{t_{k-1}}^{t_k} m(\tau) d\tau$	$\frac{(\Delta t)^2}{2}$
ID-2	$m(t_k) - m(t_0)$	$2\sigma^2$
	$\int_{t_0}^{t_k} \tau m(\tau) d\tau$	$\left(\frac{\sigma(\Delta t)^2}{2}\right)^2 \left(\frac{(k-1)k(2k-1)}{3} + k^2\right)$
	$\int_{t_0}^{t_k} m(\tau) d\tau$	$\frac{(\sigma\Delta t)^2 k}{2}$
DF	$m_k - m_{k-1}$	$2\sigma^2$
	$\Delta t_k \left(\frac{t_{k-1}+t_k}{2}\right)(m_k + m_{k-1})$	$2\left(\sigma(\Delta t)^2\right)^2 \left(k + \frac{1}{2}\right)^2$
	$\Delta t_k (m_k + m_{k-1})$	$2(\sigma\Delta t)^2$

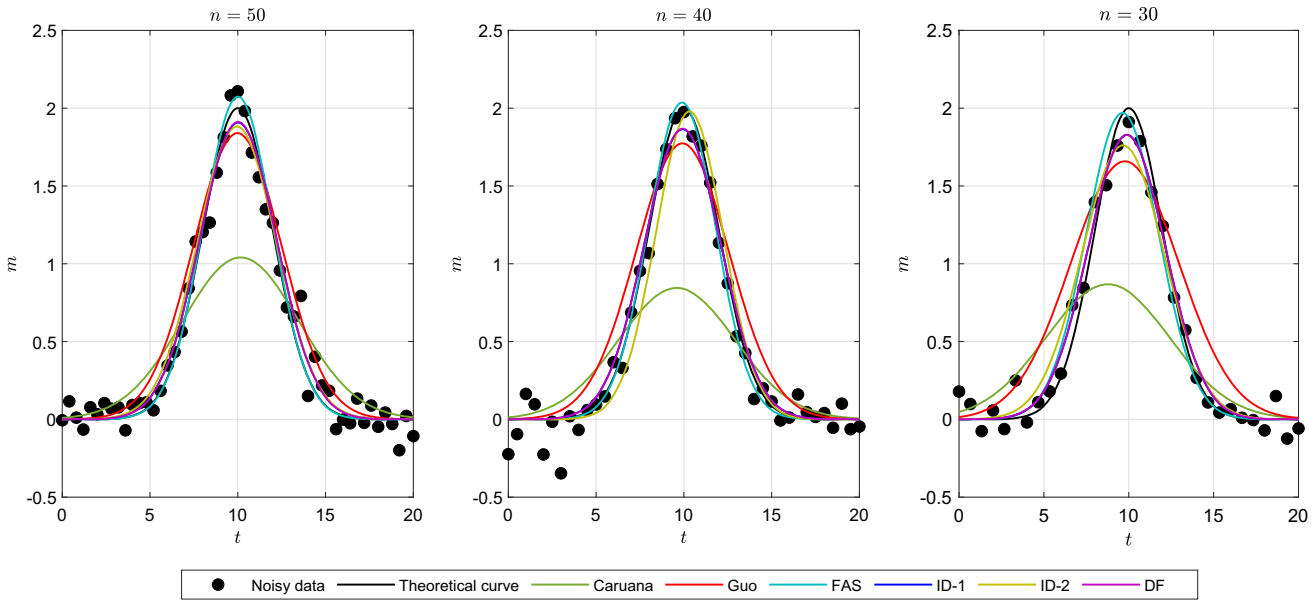

Fig. 1 Results of different fitting algorithms of the Gaussian curve with $a = 2, b = 10$ and $c = \frac{1}{8}$ in the presence of noise with $\sigma = 0.1$ ($SNR = 20$) for $n = 50, n = 40$ and $n = 30$, respectively

Table 2 Comparison of *ARE* and *RMS* for $a = 2, b = 10, c = \frac{1}{8}, \sigma = 0.1 (SNR = 20)$ and $n = 50$ (averaged over 5000 trails)

Algorithm	*ARE* (a)	*ARE* (b)	*ARE* (c)	$RMS(\hat{m} - \overline{m})$
Caruana	55.41	3.39	65.07	42.86
Guo	9.06	0.92	43.61	18.66
FAS	5.23	1.17	8.67	5.92
ID-1	3.19	0.39	11.24	4.16
ID-2	2.80	1.07	9.29	4.93
DF	3.04	0.39	10.68	4.00

$\left(b - \frac{W}{2\sqrt{2c}}, b + \frac{W}{2\sqrt{2c}}\right)$. As depicted in Fig. 3b, for shorter observation interval $W \leq 6$, the methods except for FAS are performing well. But, for longer observation interval $W > 6$, the opposite is happening; FAS is performing better. However, for *RMS* comparison, ID-1, ID-2 and DF are better. As shown in Fig. 3c, $ARE(c)$ is increasing for FAS when W is increased, because accumulated noise error influences the estimation significantly. ID-1, ID-2 and DF also produce better fitting result in this case.

3.2 Numerical Testing with Iteration for Gaussian Long Tail

In this section, the numerical testing of Gaussian long tail is given. As pointed out in Guo (2011), if the long tail of Gaussian curve is included in observed data, the noise level far away from the peak may affect largely the estimation results. It even leads to the failure in the estimation task. Therefore, the estimation task with Gaussian long tail can be a good test problem for checking the robustness of the methods. It is fair to compare between the iterative versions of the considered methods. Since Caruana's algorithm has no structure to execute iteration, it is excluded from the testing. Like Guo's algorithm and FAS method, the iterative version of DF can be formulated as follows.

$$\begin{aligned} m_k - m_{k-1} &= z_1^j \Delta t_k \left(\frac{t_{k-1} + t_k}{2}\right)(m_k^{j-1} + m_{k-1}^{j-1}) \\ &\quad + z_2^j \Delta t_k (m_k^{j-1} + m_{k-1}^{j-1}), \end{aligned} \tag{25}$$

where $z_1^j = -c^j, z_2^j = c^j b^j$ and

$$m_k = z_3^j e^{-c^j (t - b^j)^2} \tag{26}$$

where $z_3^j = a^j$ with $m^j = m$ for $j = 0$ and $m^j = a^j e^{-c^j (t-b^j)^2}$ for $j > 0$. Here Eq. (26) is solved in a least square sense for z_3^j. The iterative versions of ID-1 and ID-2 can be written with same principle; change the values of m in right-hand side of (9) by $j-1$th iterated value m^{j-1} with $m^j = a^j e^{-c^j (t-b^j)^2}$.

The parameter setting $a = 1, b = 18$ and $c = \frac{1}{8}$ generates Gaussian long tail. Figure 4 shows the fitting task of the considered algorithm for six iterations. The noise level is $\sigma = 0.1$ and the sampling number is $n = 50$. It shows our purposed algorithm ID-1 and DF fits faster than Guo's method and FAS. However, ID-2 is not converging even after six iterations. It may not converge at all. For ID-2

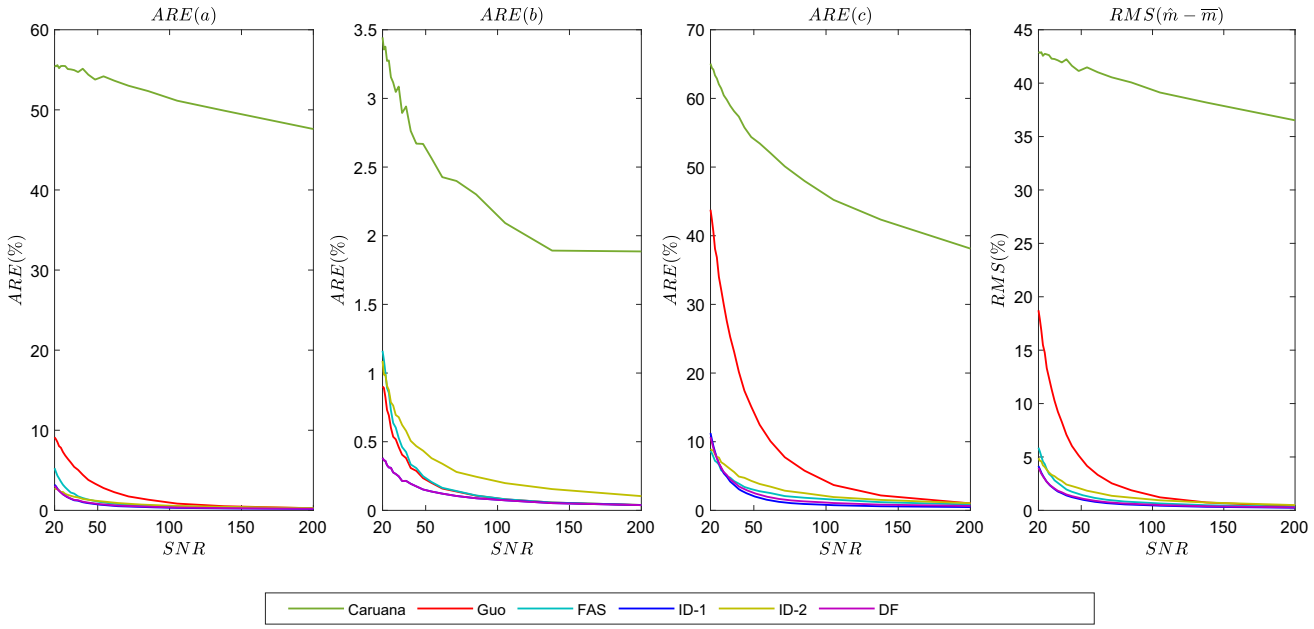

Fig. 2 Results of different fitting algorithms of the Gaussian curve with $a = 2, b = 10$ and $c = \frac{1}{8}$ with *SNR* varied from 20 to 200 ($\sigma = 0.1$ to 0.01) for $n = 50$

method, this failed estimation is due to the accumulated errors from integration as mentioned above. As FAS also relies on integration of observed values, its convergence is slower than ID-1 and DF as shown in Table 3. However, for Gaussian long tail with $a = 1, b = 19, c = \frac{1}{8}, \sigma = 0.1$ and $n = 200$, FAS is not converged even after six iterations as seen in Fig. 5, while ID-1 and DF fit nicely.

4 Complexity of the Purposed Methods

The complexity comparison of Guo's, Roonizi's and FAS methods in terms of the number of additions and multiplications required to estimate the three parameters were discussed in Al-Nahhal et al. (2019). Since ID-2 is identified as Roonizi's method for Gaussian, it is enough to consider the complexity of ID-1 and DF methods.

Assuming $t_k = k\Delta t$ for $k = 0, \ldots, n$, ID-1 has the form

$$\begin{aligned} m_k - m_{k-1} &= z_1(\Delta t)^2((k-1)m_{k-1} + km_k) \\ &\quad + z_2\Delta t(m_{k-1} + m_k), \\ z_1 &= -c, z_2 = cb \end{aligned} \tag{27}$$

for $k = 1, \ldots, n$. The least square solution of (27) is determined from the following 2×2 Gaussian system:

$$\begin{pmatrix} \sum_{k=1}^{n} p_k^1 p_k^3 \\ \sum_{k=1}^{n} p_k^2 p_k^3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{n} (p_k^1)^2 & \sum_{k=1}^{n} p_k^1 p_k^2 \\ \sum_{k=1}^{n} p_k^2 p_k^1 & \sum_{k=1}^{n} (p_k^2)^2 \end{pmatrix} \times \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \tag{28}$$

where

$$\begin{aligned} p_k^1 &= (\Delta t)^2((k-1)m_{k-1} + km_k), \\ p_k^2 &= \Delta t(m_{k-1} + m_k), \\ p_k^3 &= m_k - m_{k-1} \end{aligned} \tag{29}$$

are for shorthand notations. Using the estimated b and c, a is calculated by

$$a = \frac{\sum_{k=0}^{n} m_k e^{-c(t_k - b)^2}}{\sum_{k=0}^{n} e^{-2c(t_k - b)^2}}. \tag{30}$$

For DF method, only the term p_k^1 is different. Thus, DF takes the form

$$\begin{aligned} m_k - m_{k-1} &= z_1(\Delta t)^2(k - 1/2)(m_{k-1} + m_k) \\ &\quad + z_2\Delta t(m_{k-1} + m_k), \\ z_1 &= -c, z_2 = cb \end{aligned} \tag{31}$$

for $k = 1, \ldots, n$. The least square solution of (31) is determined from

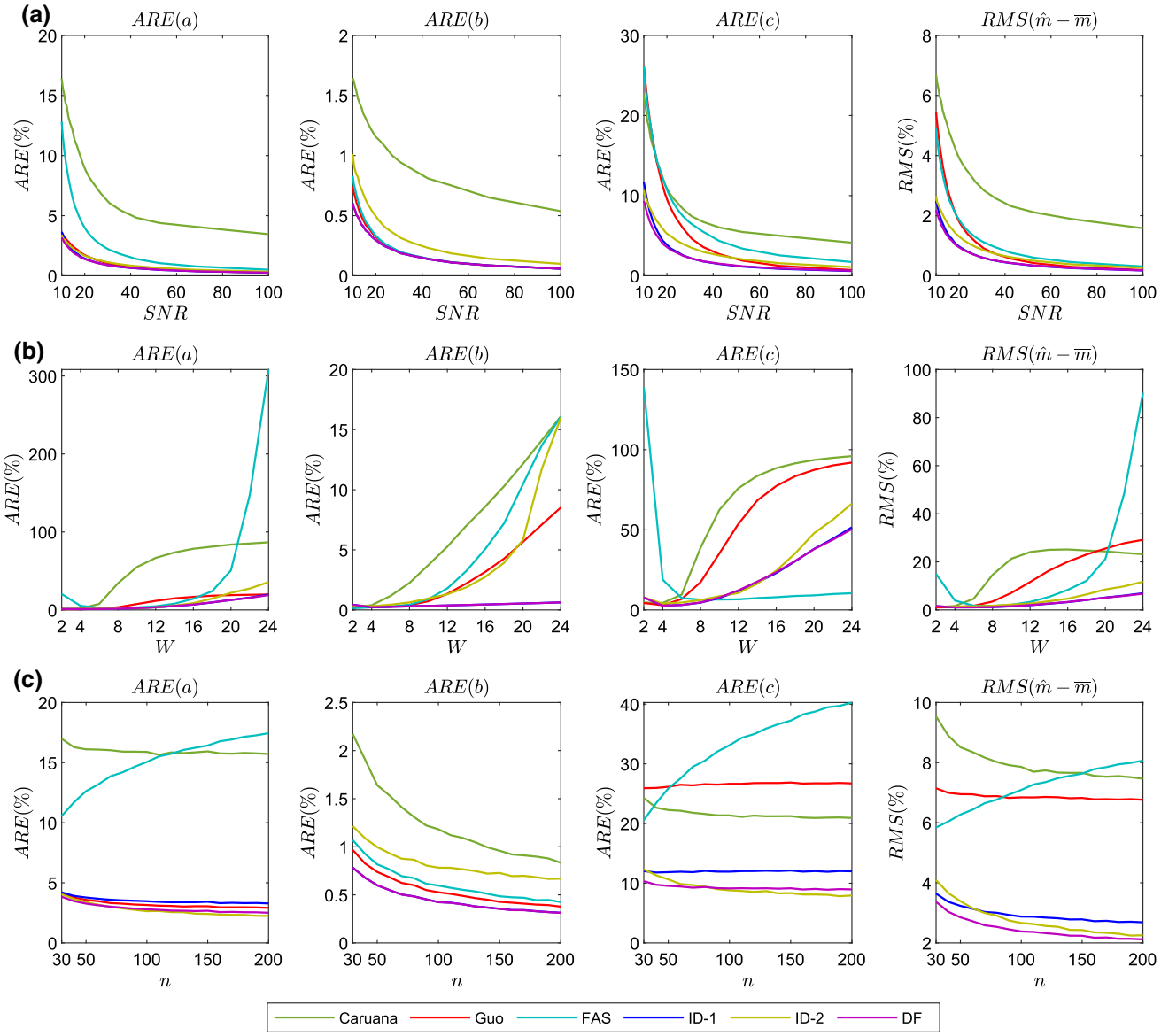

Fig. 3 Results of Monte Carlo simulation (5000 trails) for the Gaussian curve with $a = 1, b = 10$ and $c = \frac{1}{8}$ with the parameters *SNR*, *W* and *n* varied while the other two are held constant as: **a** $W = 6, n = 50$, **b** $SNR = 25, n = 40$, **c** $SNR = 10, W = 6$

$$\begin{pmatrix} \sum_{k=1}^{n} \widetilde{p}_k^1 p_k^3 \\ \sum_{k=1}^{n} p_k^2 p_k^3 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{n} (\widetilde{p}_k^1)^2 & \sum_{k=1}^{n} \widetilde{p}_k^1 p_k^2 \\ \sum_{k=1}^{n} p_k^2 \widetilde{p}_k^1 & \sum_{k=1}^{n} (p_k^2)^2 \end{pmatrix} \times \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \tag{32}$$

where

$$\widetilde{p}_k^1 = (\Delta t)^2 (k - 1/2)(m_{k-1} + m_k) \tag{33}$$

and p_k^2, p_k^3 are same as in (29).

Assuming that subtraction and division are equivalent to addition and multiplication, respectively, we calculated the number of additions and multiplications required to compute *a*, *b* and *c* for ID-1 and DF in Tables 4 and 5. Here, A_{exp} and M_{exp} denote the number of additions and multiplications required to calculate the exponential term $e^{-c(t_k-b)^2}$. Since it requires three additions and six multiplications to solve 2×2 linear system of equations using Gaussian elimination, a total number of additions and multiplications are

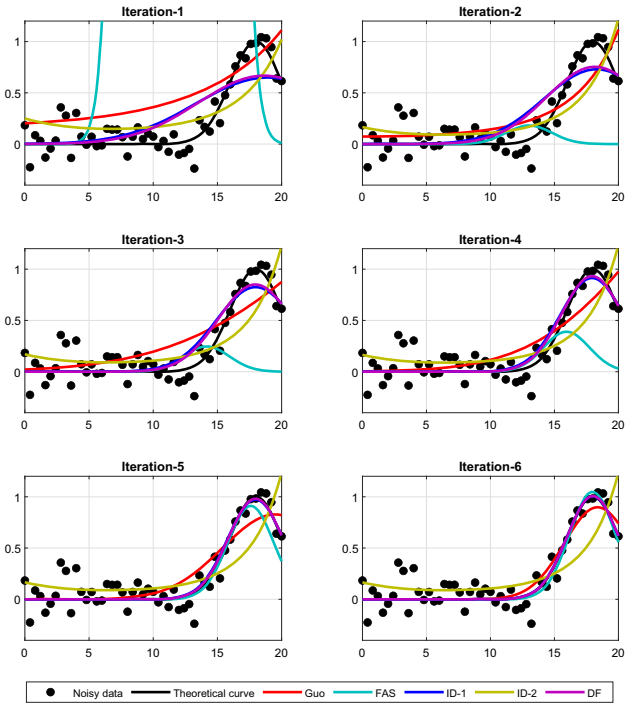

Fig. 4 Results of six iterations for $a = 1, b = 18, c = \frac{1}{8}, \sigma = 0.1$ and $n = 50$

$$\begin{aligned} \text{Add}^{(\text{ID}-1)} &= 11n + 7 + (n+1)A_{exp}, \\ \text{Mul}^{(\text{ID}-1)} &= 9n + 18 + (n+1)M_{exp}, \\ \text{Add}^{(\text{DF})} &= 10n + 7 + (n+1)A_{exp}, \\ \text{Mul}^{(\text{DF})} &= 8n + 18 + (n+1)M_{exp}. \end{aligned} \tag{34}$$

The number of operations required for Guo, Roonizi and FAS algorithms, which are calculated in Al-Nahhal et al. (2019), are

$$\begin{aligned} \text{Add}^{(\text{Guo})} &= 8n + 11 + (n+1)A_{ln}, \\ \text{Mul}^{(\text{Guo})} &= 11n + 28 + (n+1)M_{ln}, \\ \text{Add}^{(\text{Roonizi})} &= n^2 + 10n + (n+1)A_{exp} + 4, \\ \text{Mul}^{(\text{Roonizi})} &= 0.5n^2 + 10.5n + (n+1)M_{exp} + 19, \\ \text{Add}^{(\text{FAS})} &= 8n + 5 + (n+1)A_{ln}, \\ \text{Mul}^{(\text{FAS})} &= 10n + 22 + (n+1)M_{ln} \end{aligned} \tag{35}$$

where we substituted N in Al-Nahhal et al. (2019) for $n + 1$.

Roonizi's algorithm (ID-2) requires $O(n^2)$ number of operations compared to others. Thus, Guo, FAS, ID-1 and DF, which are in the order of $O(n)$, are faster than Roonizi's algorithm. However, if we compare ID-1 and DF with Guo, FAS, there is trade-off between number of additions and multiplications. In terms of multiplications, ID-1 and DF require fewer number of operations. In terms of additions, Guo and FAS are better. If ID-1 and DF are compared, DF requires less number of operations. Therefore, our purposed methods have fast property as well.

Table 3 Results of six iterations for $a = 1, b = 18, c = \frac{1}{8}, \sigma = 0.1$ and $n = 50$

Algorithm	*ARE* (a)	*ARE* (b)	*ARE* (c)	*RMS* $(\hat{m} - \overline{m})$
Iteration-1				
Guo	81.41	131.00	102.19	28.82
FAS	28669.44	33.64	24.25	11457.40
ID-1	35.10	4.89	87.13	18.72
ID-2	85.18	61.93	108.93	20.97
DF	33.24	3.47	84.59	17.66
Iteration-2				
Guo	92.65	98.93	105.55	17.02
FAS	81.88	27.68	24.25	39.71
ID-1	27.05	1.58	76.28	14.46
ID-2	90.80	64.13	111.30	19.11
DF	24.61	0.85	71.85	13.21
Iteration-3				
Guo	113.95	98.28	97.10	17.50
FAS	75.34	21.43	24.25	37.90
ID-1	17.52	0.14	57.55	9.72
ID-2	90.92	63.43	111.59	19.24
DF	14.94	0.28	51.61	8.45
Iteration-4				
Guo	114.92	71.26	94.61	14.07
FAS	61.02	11.45	24.25	30.21
ID-1	8.60	0.36	34.65	5.26
ID-2	91.12	63.80	111.59	19.18
DF	6.80	0.33	29.11	4.32
Iteration-5				
Guo	17.26	8.36	76.27	10.86
FAS	8.92	2.23	24.25	6.61
ID-1	2.67	0.19	14.86	2.10
ID-2	91.08	63.66	111.62	19.21
DF	1.82	0.14	11.67	1.62
Iteration-6				
Guo	10.22	2.05	41.42	4.81
FAS	4.72	0.09	24.25	2.40
ID-1	0.46	0.01	2.42	0.46
ID-2	91.11	63.72	111.61	19.20
DF	0.70	0.05	1.51	0.41

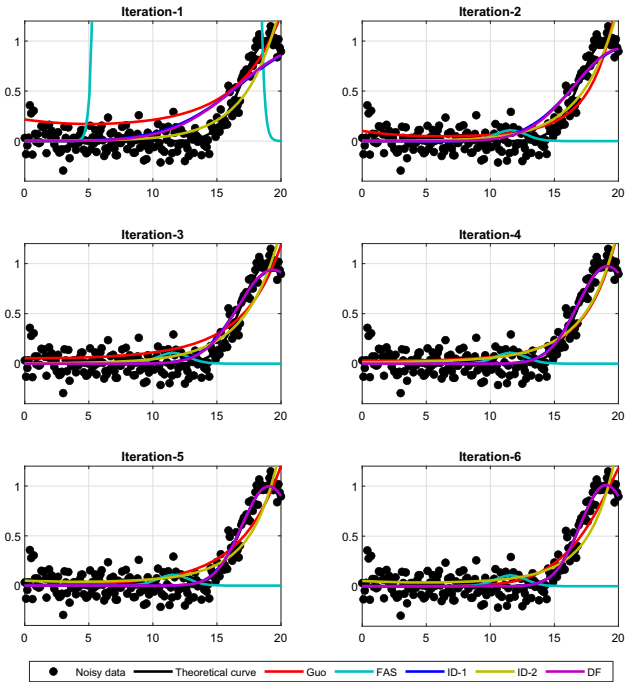

Fig. 5 Results of six iterations for $a = 1, b = 19, c = \frac{1}{8}, \sigma = 0.1$ and $n = 200$

5 Curve Fitting of Two Gaussian Functions

In this section, a linear least square estimation method based on differential and integral transformations is proposed for fitting noisy data with two Gaussian waveforms. The mean square error MSE and Pearson's correlation coefficient r between the observed noisy data and the fitted two Gaussian functions are calculated to evaluate the fitting accuracy of the proposed method. The value of r close to 1 indicates the correlation between the reconstructed two Gaussian curves and the observed noisy data is significant. Besides, the applicability of the method is tested by fitting three different types of noisy photoplethysmogram (PPG) signals.

Assume that two Gaussian functions have the form

$$m = m_1 + m_2, \tag{36}$$

where

$$m_1 = a_1 e^{-c_1(t-b_1)^2}, \; m_2 = a_2 e^{-c_2(t-b_2)^2}. \tag{37}$$

The six parameters to be estimated are the height-controlling parameters a_1 and a_2, the peak position-controlling parameters b_1 and b_2 and the width-controlling parameters c_1 and c_2. Taking differential of Eq. (36) gives us

Table 4 Number of additions and multiplications required to calculate a, b and c for ID-1

Terms		Additions	Multiplications
–	$(\Delta t)^2$	–	1
–	$(\Delta t)^3$	–	1
–	$(\Delta t)^4$	–	1
$\sum_{k=1}^{n} p_k^1 p_k^2$	$(\Delta t)^3 \sum_{k=1}^{n}((k-1)m_{k-1} + km_k)(m_{k-1} + m_k)$	$3n + n - 1$	$3n + 1$
$\sum_{k=1}^{n} p_k^1 p_k^3$	$(\Delta t)^2 \sum_{k=1}^{n}((k-1)m_{k-1} + km_k)(m_k - m_{k-1})$	$n + n - 1$	$n + 1$
$\sum_{k=1}^{n}(p_k^1)^2$	$(\Delta t)^4 \sum_{k=1}^{n}((k-1)m_{k-1} + km_k)^2$	$n - 1$	$n + 1$
$\sum_{k=1}^{n}(p_k^2)^2$	$(\Delta t)^2 \sum_{k=1}^{n}(m_{k-1} + m_k)^2$	$n - 1$	$n + 1$
$\sum_{k=1}^{n} p_k^2 p_k^3$	$\Delta t \sum_{k=1}^{n}(m_k + m_{k-1})(m_k - m_{k-1})$	$n - 1$	$n + 1$
c	$-z_1$	1	–
b	$\frac{z_2}{c}$	–	1
–	$\sum_{k=0}^{n} m_k e^{-c(t_k-b)^2}$	$n + (n+1)A_{exp}$	$n + 1 + (n+1)M_{exp}$
–	$\sum_{k=0}^{n} e^{-2c(t_k-b)^2}$	n	$n + 1$
a	$\frac{\sum_{k=0}^{n} m_k e^{-c(t_k-b)^2}}{\sum_{k=0}^{n} e^{-2c(t_k-b)^2}}$	–	1
Total	–	$11n + 4 + (n+1)A_{exp}$	$9n + 12 + (n+1)M_{exp}$

Table 5 Number of additions and multiplications required to calculate a, b and c for DF

Terms		Additions	Multiplications
–	$(\Delta t)^2$	-	1
–	$(\Delta t)^3$	–	1
–	$(\Delta t)^4$	–	1
$\sum_{k=1}^{n} \widetilde{p}_k^1 p_k^2$	$(\Delta t)^3 \sum_{k=1}^{n} (k-1/2)(m_{k-1}+m_k)^2$	$2n+n-1$	$2n+1$
$\sum_{k=1}^{n} \widetilde{p}_k^1 p_k^3$	$(\Delta t)^2 \sum_{k=1}^{n} (k-1/2)(m_{k-1}+m_k)(m_k-m_{k-1})$	$n+n-1$	$n+1$
$\sum_{k=1}^{n} (\widetilde{p}_k^1)^2$	$(\Delta t)^4 \sum_{k=1}^{n} (k-1/2)^2 (m_{k-1}+m_k)^2$	$n-1$	$n+1$
$\sum_{k=1}^{n} (p_k^2)^2$	$(\Delta t)^2 \sum_{k=1}^{n} (m_{k-1}+m_k)^2$	$n-1$	$n+1$
$\sum_{k=1}^{n} p_k^2 p_k^3$	$\Delta t \sum_{k=1}^{n} (m_k+m_{k-1})(m_k-m_{k-1})$	$n-1$	$n+1$
c	$-z_1$	1	–
b	$\frac{z_2}{c}$	–	1
–	$\sum_{k=0}^{n} m_k e^{-c(t_k-b)^2}$	$n+(n+1)A_{exp}$	$n+1+(n+1)M_{exp}$
–	$\sum_{k=0}^{n} e^{-2c(t_k-b)^2}$	n	$n+1$
a	$\frac{\sum_{k=0}^{n} m_k e^{-c(t_k-b)^2}}{\sum_{k=0}^{n} e^{-2c(t_k-b)^2}}$	–	1
Total	–	$10n+4+(n+1)A_{exp}$	$8n+12+(n+1)M_{exp}$

$$m' = m_1' + m_2', \tag{38}$$

where

$$m_1' = 2c_1(b_1 - t)m_1, \; m_2' = 2c_2(b_2 - t)m_2. \tag{39}$$

By combining Eqs. (36 and 38), we can obtain a system of linear equations with variables m_1 and m_2

$$\begin{pmatrix} 1 & 1 \\ 2c_1(b_1-t) & 2c_2(b_2-t) \end{pmatrix} \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} = \begin{pmatrix} m \\ m' \end{pmatrix}. \tag{40}$$

The analytical solutions of the linear system (40) are given by

$$\begin{aligned} m_1 &= \frac{-m' - 2c_2 m(t-b_2)}{2c_1(t-b_1) - 2c_2(t-b_2)}, \\ m_2 &= \frac{m' + 2c_1 m(t-b_1)}{2c_1(t-b_1) - 2c_2(t-b_2)}. \end{aligned} \tag{41}$$

After taking differential from Eq. (38), we have

$$m'' = m_1'' + m_2'', \tag{42}$$

where

$$\begin{aligned} m_1'' &= \left(4c_1^2(b_1-t)^2 - 2c_1\right)m_1, \\ m_2'' &= \left(4c_2^2(b_2-t)^2 - 2c_2\right)m_2. \end{aligned} \tag{43}$$

Putting the expression (41) in (42) yields formally

$$Ax = B, \tag{44}$$

where A, x and B are

$$A = \begin{pmatrix} m' & m'' & m & m't & mt & m't^2 & mt^2 & mt^3 \end{pmatrix}, \tag{45}$$

$$x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{pmatrix} = C \cdot \frac{1}{c_2 - c_1}, \tag{46}$$

in which

$$C = \begin{pmatrix} -2b_1^2c_1^2 + 2b_2^2c_2^2 + c_1 - c_2 \\ b_1c_1 - b_2c_2 \\ 4b_1^2b_2c_1^2c_2 - 4b_1b_2^2c_1c_2^2 + 2b_1c_1c_2 - 2b_2c_1c_2 \\ 4b_1c_1^2 - 4b_2c_2^2 \\ -4b_1^2c_1^2c_2 - 8b_1b_2c_1^2c_2 + 8b_1b_2c_1c_2^2 + 4b_2^2c_1c_2^2 \\ -2c_1^2 + 2c_2^2 \\ 8b_1c_1^2c_2 - 4b_1c_1c_2^2 + 4b_2c_1^2c_2 - 8b_2c_1c_2^2 \\ -4c_1^2c_2 + 4c_1c_2^2 \end{pmatrix} \tag{47}$$

and

$$B = -m''t. \tag{48}$$

Let us suppose that the observed data $m(t_i)$ is sampled over $t_i, i = 1, 3, \cdots, n$. Since the second-order derivative is much sensitive to a noise, we integrate twice (like Roonizi's method for single Gaussian case) Eq. (44) on the intervals $[t_1, t_i], i = 2, 3, \cdots, n$. Then, we obtain the $n \times 10$ matrix $\bar{\bar{A}}$, the 10×1 vector y, and the $n \times 1$ vector $\bar{\bar{B}}$, which can be described as follows:

$$\bar{\bar{A}} = \left(\overline{m}m\overline{\overline{m}}\,\overline{tm} - \overline{\overline{m}}\,\overline{\overline{tm}}\,\overline{t^2m} - 2\overline{\overline{tm}}\,\overline{\overline{t^2m}}\,\overline{\overline{t^3m}}\,t\;1 \right), \tag{49}$$

$$y = \begin{pmatrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} \end{pmatrix}^T, \tag{50}$$

$$\bar{\bar{B}} = -(\,tm - 2\overline{m}\,), \tag{51}$$

where

$$y_i = x_i, \;\; i = 1, 2, \ldots, 8; \tag{52}$$

$$y_9 = -e_1x_1 - e_2x_2 - e_3x_4 - e_4x_6 - e_5 + e_1, \tag{53}$$

$$y_{10} = -e_1x_2 - t_1y_9 - e_3, \tag{54}$$

$$e_1 = m(t_1), \tag{55}$$

$$e_2 = m'(t_1), \tag{56}$$

$$e_3 = t_1m(t_1), \tag{57}$$

$$e_4 = t_1^2m(t_1), \tag{58}$$

$$e_5 = t_1m'(t_1). \tag{59}$$

The notation $\overline{\phi}$, where $\phi = m, tm, t^2m$, represents the cumulative integral computed by the composite trapezoidal rule, which can be defined as

$$\overline{\phi}_i = \begin{cases} 0, & i = 1; \\ \sum_{j=2}^{i} \dfrac{m(t_{j-1}) + m(t_j)}{2}(t_j - t_{j-1}), & i = 2, 3, \ldots, n. \end{cases} \tag{60}$$

And the notation $\overline{\overline{\psi}}$, where $\overline{\psi} = \overline{m}, \overline{tm}, \overline{t^2m}, \overline{t^3m}$, denotes the cumulative integral of $\overline{\psi}$. Therefore, the problem of fitting two Gaussian functions is, finally, transformed to the linear least squares estimation problem. Note that it is not necessary to evaluate the values of $e_1, \ldots, e_5$ as follows. The parameters c_1 and c_2 are obtained from

$$\begin{cases} c_1 = \dfrac{y_6 - \sqrt{y_6^2 - 4y_8}}{4}, \\ c_2 = \dfrac{y_6 + \sqrt{y_6^2 - 4y_8}}{4}. \end{cases} \text{ or } \begin{cases} c_1 = \dfrac{y_6 + \sqrt{y_6^2 - 4y_8}}{4}, \\ c_2 = \dfrac{y_6 - \sqrt{y_6^2 - 4y_8}}{4}. \end{cases} \tag{61}$$

The peak-controlling parameters b_1 and b_2 are found from

$$\begin{cases} b_1 = \dfrac{8c_1c_2y_2 - 4c_1^2y_2 - y_7}{4c_1(c_1 + c_2)}, \\ b_2 = \dfrac{8c_1c_2y_2 - 4c_2^2y_2 - y_7}{4c_2(c_1 + c_2)}. \end{cases} \tag{62}$$

Furthermore, putting the estimated parameters b_1, b_2 and c_1, c_2 into the original two Gaussian functions gives us

$$\begin{pmatrix} e^{-c_1(t-b_1)^2} & e^{-c_2(t-b_2)^2} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = m. \tag{63}$$

Hence, based on Eq. (63), the linear least square method can be invoked again to get the estimates of height-controlling parameters a_1 and a_2. Let us denote the proposed method described through (49– 63) by ID-3.

The evaluation indicators in this section are defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(m_{noise}(i) - m_{fitted}(i))^2 \tag{64}$$

$$r = \frac{\sum_{i=1}^{n}(m_{noise}(i) - \overline{m_{noise}})(m_{fitted}(i) - \overline{m_{fitted}})}{\sqrt{\sum_{i=1}^{n}(m_{noise}(i) - \overline{m_{noise}})^2}\sqrt{\sum_{i=1}^{n}(m_{fitted}(i) - \overline{m_{fitted}})^2}} \tag{65}$$

where $m_{noise}(i)$ and $m_{fitted}(i)$ are individual points of noisy data and the fitted two Gaussian curve, $\overline{m_{noise}}$ and $\overline{m_{fitted}}$ are the average value of m_{noise} and m_{fitted}, and n is the number of noisy data points.

Figure 6 shows noisy data and the reconstructed two Gaussian curves using the parameters estimated by the ID-3 method. It is clear that the reconstructed curve agrees well with the theoretical curve and the noisy data. The indicators of the corresponding fitting results, MSE and r, are presented in Table 6. It tells the fitted two Gaussian curves have a strong correlation and low MSE with the observed noisy data.

In Fig. 6a, the red curve represents the theoretical two Gaussian curves with parameters $a_1 = 3$, $b_1 = 6.5$, $c_1 = 0.1$, $a_2 = 5$, $b_2 = 5$, $c_2 = 0.6$. The blue dots are observed noisy data that are sampled from the theoretical curve corrupted by Gaussian random noise with the

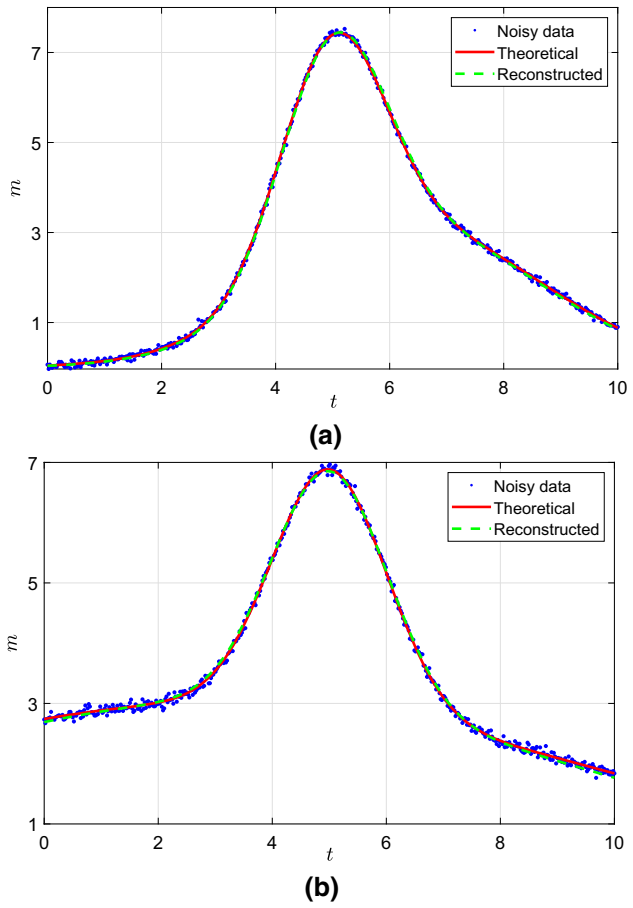

Fig. 6 Reconstructed two Gaussian curves by using the ID-3 method. **a** $a_1 = 3,\ b_1 = 6.5,\ c_1 = 0.1,\ a_2 = 5,\ b_2 = 5,\ c_2 = 0.6$, **b** $a_1 = 3,\ b_1 = 3,\ c_1 = 0.01,\ a_2 = 4,\ b_2 = 5,\ c_2 = 0.5$

standard deviation $\sigma = 0.05$. The total number of observed data points is $n = 500$. The t-range of the sampled data varies from 0 to 10. The green dashed curve denotes the reconstructed two Gaussian curves by the ID-3 method. In Fig. 6b, the theoretical two Gaussian curves have parameters of $a_1 = 3,\ b_1 = 3,\ c_1 = 0.01,\ a_2 = 4,\ b_2 = 5,\ c_2 = 0.5$.

To evaluate the performance of the ID-3 method, Monte Carlo simulations were performed. Figure 7 shows the mean and standard deviation of the MSE and Pearson's correlation coefficient over a range of noise levels and sampling numbers. The parameters of the theoretical two Gaussian, used in plotting Fig. 7, are $a_1 = 3,\ b_1 = 6.5,\ c_1 = 0.1,\ a_2 = 5,\ b_2 = 5,\ c_2 = 0.6$, the t-range varies from 0 to 10, the number of observed data points used in plotting Fig. 7(a, b) is $n = 1,000$, and the level of added noise is $\sigma = 0.05$ in plotting Fig. 7c, d). For each error bar, the mean and standard deviation are calculated by 5,000 simulations. Figure 7 indicates that ID-3 can generate better fitting results when the level of noise is low. It is observed that the fitting accuracy of ID-3 is improved when n is increased.

A single pulse PPG signal consists of one systolic wave and one diastolic wave, which can be synthesized by the two Gaussian function (Tang et al. 2020). Thus, three synthetic PPG signals of different types of morphologies with specified additive noise and sampling numbers are used to verify the applicability of the ID-3 method. The number of observed data is $n = 5,000$, and the observed noisy data points are sampled from synthetic PPG signals that are corrupted by Gaussian random noise having zero-mean and the standard deviation σ of 0.05, where the two Gaussian parameters of these synthetic PPG signals can be found in Tang et al. (2020). Figure 8 depicts the reconstructed two Gaussian curves generated by ID-3 method, and table shows the corresponding fitting results. The MSEs between the observed noisy data and the reconstructed curve are 0.0086, 0.0032 and 0.0030, respectively, the corresponding Pearson's correlation coefficients are 0.9631, 0.9873 and 0.9774, which indicate that the simulated two Gaussian curves agree well with the observed

Table 6 Fitted results of parameters, the mean square errors and Pearson's correlation coefficients

Fitted results	a_1	b_1	c_1	a_2	b_2	c_2	MSE	r
Figure 6a	2.9661	6.5072	0.1027	5.0406	5.0194	0.5871	0.0037	0.9997
Figure 6b	3.0258	3.1850	0.0116	3.9397	5.0037	0.4860	0.0036	0.9992

Table 7 Estimated two Gaussian parameters, the mean square errors and the correlation coefficients corresponding to the reconstructed curves in Figure 8

Fitted results	a_1	b_1	c_1	a_2	b_2	c_2	MSE	r
Figure 8a	1.6537	−0.5119	0.3574	−1.4005	−0.1708	0.9108	0.0086	0.9631
Figure 8b	2.2069	−0.6335	0.4053	−1.4193	−0.3787	0.6523	0.0032	0.9873
Figure 8c	10.8834	0.4632	0.2096	−10.2735	0.5114	0.2329	0.0030	0.9774

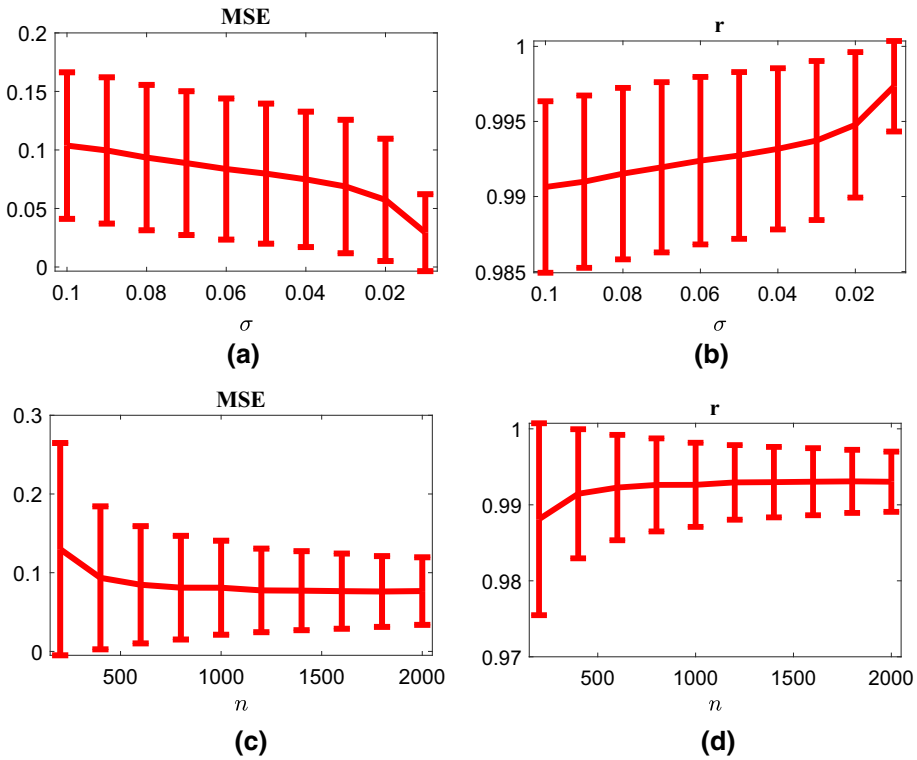

Fig. 7 Mean and standard deviation of the MSE and Pearson's correlation coefficient, respectively, over a range of noise levels and sampling numbers

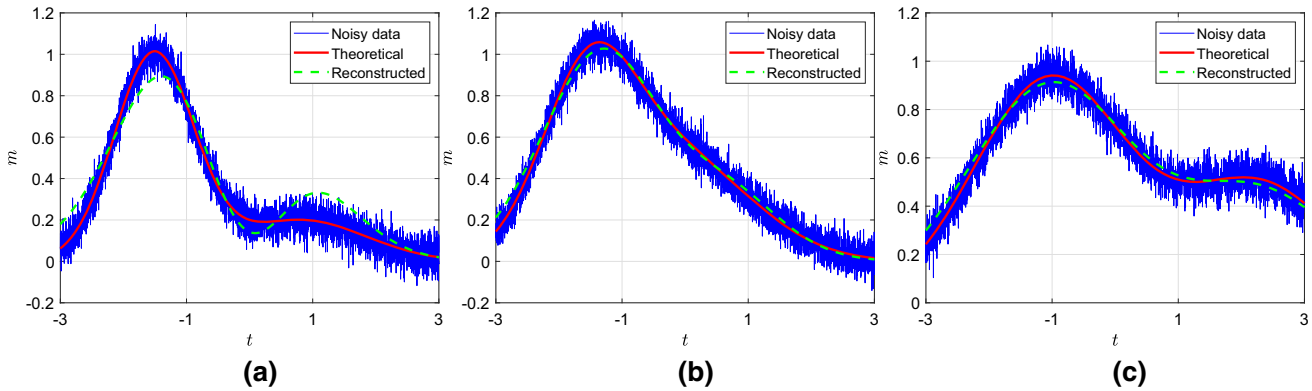

Fig. 8 Reconstructed two Gaussian curves by using the ID-3 method to fit data sets of three noisy synthetic PPG signals

noisy data points. Thus, the ID-3 method can be applied effectively to noisy PPG signals.

6 Conclusion

In this paper, we proposed simple and effective schemes to fit Gaussian functions to the observed noisy data. For one Gaussian fitting problem, the method DF, which discretizes the differential equation by trapezoidal rule, can yield good fitting result. Another method, ID-1, which is a refined Roonizi's method, performs equally well. Extensive numerical tests and comparisons are conducted to validate the proposed methods. For two Gaussian fitting problem, we transformed the nonlinear fitting problem to linear least square estimation problem by some algebraic manipulations. The experimental results show the method ID-3 gives accurate results and can be applied to real data.

Acknowledgements The authors are expressing their sincere gratitude to the anonymous reviewers for their comments. This research was funded by NRF with 2017R1E1A1A03070061.

References

Al-Nahhal I, Dobre OA, Basar E, Moloney C, Ikki S (2019) A fast, accurate, and separable method for fitting a gaussian function [tips tricks]. IEEE Signal Process Mag 36(6):157–163. https://doi.org/10.1109/MSP.2019.2927685

Bobroff N (1986) Position measurement with a resolution and noise-limited instrument. Rev Scientific Instrum 57(6):1152–1157. https://doi.org/10.1063/1.1138619

Caruana RA, Searle RB, Heller T, Shupack SI (1986) Fast algorithm for the resolution of spectra. Anal Chem 58(6):1162–1167. https://doi.org/10.1021/ac00297a041

Guo H (2011) A simple algorithm for fitting a gaussian function [dsp tips and tricks]. IEEE Signal Process Mag 28(5):134–137. https://doi.org/10.1109/MSP.2011.941846

Ireland J (2005) Precision limits to emission-line profile measuring experiments. Astrophys J 620(2):1132–1139. https://doi.org/10.1086/427230

Kheirati Roonizi E (2013) A new algorithm for fitting a gaussian function riding on the polynomial background. IEEE Signal Process Lett 20(11):1062–1065. https://doi.org/10.1109/LSP.2013.2280577

Lenz DD, Ayres TR (1992) Errors associated with fitting gaussian profiles to noisy emission-line spectra. Publ Astron Soc Pac 104:1104. https://doi.org/10.1086/133096

Oppenheim AV, Wilsky AS, Young I (1983) Signals and systems. Prentice-Hall, New Jersey

Tang Q, Chen Z, Ward R, Elgendi M (2020) Synthetic photoplethysmogram generation using two gaussian functions. Scientific Reports 10(1):13883. https://doi.org/10.1038/s41598-020-69076-x

Winick KA (1986) Cramér-rao lower bounds on the performance of charge-coupled-device optical position estimators. J Opt Soc Am A 3(11):1809–1815. https://doi.org/10.1364/JOSAA.3.001809