On the influence of roundoff errors on the convergence of the gradient descent method with low-precision floating-point computation∗

Lu Xia†, Stefano Massei†, Michiel Hochstenbach†, and Barry Koren†

Abstract. The employment of stochastic rounding schemes helps prevent stagnation of convergence, due to vanishing gradient effect when implementing the gradient descent method in low precision. Conventional stochastic rounding achieves zero bias by preserving small updates with probabilities proportional to their relative magnitudes. In this study, we propose a new stochastic rounding scheme that trades the zero bias property with a larger probability to preserve small gradients. Our method yields a constant rounding bias that, at each iteration, lies in a descent direction. For convex problems, we prove that the proposed rounding method has a beneficial effect on the convergence rate of gradient descent. We validate our theoretical analysis by comparing the performances of various rounding schemes when optimizing a multinomial logistic regression model and when training a simple neural network with 8-bit floating-point format.

Key words. Stochastic rounding, gradient descent method, roundoff errors analysis, floating-point arithmetic, low-precision computation, convergence analysis, regression, neural networks.

AMS subject classifications. 97N20, 62J02, 65G50, 68T01

1. Introduction. Low-precision computations attract increasing attention as they allow to drastically reduce the consumption of computational resources. Recently developed infrastructures support low-precision computations with the aim of improving the efficiency of deep learning inference; examples are Google’s TPU [19], NVIDIA’s Tensor Core [28], Intel’s Nervana [13], and Microsoft’s Project Brainwave [5]. These hardware accelerators support single precision (32-bit), bfloat16 (16-bit), and half precision (16-bit).

Adopting a lower precision generally introduces larger roundoff errors. Hence, it is crucial to analyze the error propagation in algorithmic procedures and to investigate the effect of different rounding schemes [18]. In addition to classical deterministic rounding strategies, such as round down, round up, round to the nearest (RN), stochastic rounding techniques have emerged [23]. An unbiased stochastic rounding scheme, that we call conventional stochastic rounding (CSR), was applied by Gupta et al. [12] to train neural networks (NNs) using low-precision fixed-point arithmetic. The experiments reported in [12] show that the training accuracy obtained using 16-bit fixed-point representation and CSR is very similar to that obtained in single precision with RN. On the other hand, RN with 16-bit fixed-point representation makes the numerical scheme stagnate. This has inspired further investigations of the use of CSR in training NNs with low-precision computations [25, 29, 36]. Besides in machine learning, CSR has been recently employed in climate modeling [30] and in solving PDEs with low precision [7], and its implementation in hardware is also growing [8, 34, 23].

Many tasks related to machine learning, e.g., training NNs, linear regression and logistic
regression, are carried out by means of the gradient descent method (GD). The latter is also widely employed in many other areas such as path planning [31, 33], autonomous vehicle systems [2], and optimal control [10, 22]. The convergence of GD in exact arithmetic is well understood; see, e.g., [26, 27, 1, 21, 37]. Still, a theoretical analysis that takes into account either deterministic or stochastic rounding errors is lacking.

A major difference between CSR and RN is that, when using CSR the gradient updates that are below the minimum rounding precision are partially captured by certain probability, while RN rounds all these small gradients to 0. This may make CSR preferable because it has the potential to prevent this loss of information [16, 12]. However, in order to have a zero rounding bias, the probability of preserving these small gradients, with CSR, is proportional to their distance to 0. Whether it is worth to introduce some rounding bias to increase the probability to preserve gradient information is to be discussed.

In this paper, we analyze the influence of rounding errors on the convergence of GD with floating-point representation for convex problems. Our analysis splits the rounding process of GD into two steps: the first incorporates the rounding in the evaluation of the gradient and of the multiplication with the stepsize; the second takes into account the rounding in the subtraction. We analyze these two steps for two different stages, i.e., the stage where GD does not stagnate with RN (Stage I) and the stage where GD does stagnate with RN (Stage II). For the first step, we propose a new stochastic rounding method that we call $\varepsilon$-stochastic rounding (SR$_\varepsilon$). The idea behind SR$_\varepsilon$ is to increase or decrease by $\varepsilon$ the probability of rounding down in CSR, depending on the sign of the input number. This yields a smaller worst-case roundoff error, compared to CSR, and an always nonnegative relative rounding bias. Hence, using SR$_\varepsilon$ to evaluate gradients and to perform the multiplication with the stepsize leads to a rounding bias that has always the same sign (component-wise) as the gradient. In other words, rounding biases are always oriented towards a descent direction. For the second step, we propose another rounding scheme that we call signed $\varepsilon$-stochastic rounding (signed-SR$_\varepsilon$), obtained by further modifying SR$_\varepsilon$. By means of signed-SR$_\varepsilon$, we can customize the sign of the total rounding bias and guarantee the monotone convergence of GD, under mild assumptions.

The outcomes of our theoretical analysis concern two aspects of GD: monotonicity and convergence rate. A summary of the main results with respect to different stages and steps is given in Table 1. Note that in Stage II, only monotonicity is studied for different rounding strategies. This is mainly because in this stage, GD converges to the optimum on the level of the rounding errors.

We validate the theoretical results with experiments on quadratic functions with 16-bit floating-point number representation. Finally, we show the effectiveness of our approach when training a multinomial logistic regression model (MLR) and a two-layer NN (although the latter is a non-convex problem) using 8-bit floating-point number representation. The results confirm that both SR$_\varepsilon$ and signed-SR$_\varepsilon$ provide higher classification accuracy than RN and CSR, with the same number of training epochs.

The work is organized as follows. In section 2, we recall the basic properties of floating-point arithmetic and CSR, and we introduce SR$_\varepsilon$ and signed-SR$_\varepsilon$. The source of rounding errors when implementing GD with floating-point representation is analyzed in section 3. In section 4, we study the influence of rounding bias on the convergence of GD for convex problems during two stages, i.e., the stage where GD stagnates with RN or not. Then, we
INFLUENCE OF FLOATING-POINT ROUNDOFF ERRORS ON CONVERGENCE OF GD

Table 1
Summary of the main theoretical results.

| Stage | Steps       | Result       | Rounding scheme                        | Reference |
|-------|-------------|--------------|----------------------------------------|-----------|
|       | (3.2a), (3.2b) | monotonicity | general rounding                       | Lemma 4.3 |
| Stage I | convergence rate | convergence rate | general rounding and CSR (3.2b) | Theorem 4.4 |
|       | convergence rate | convergence rate | CSR (3.2a) and CSR (3.2b) | Corollary 4.6 |
|       | (3.2a) and CSR (3.2b) | convergence rate | general rounding and CSR (4.23) | Proposition 4.11 |
| Stage II | (3.2a), (4.23) | monotonicity | general rounding                       | Lemma 4.7 |
|       | monotonicity | CSR (3.2a) and CSR (4.23) | Proposition 4.12 |
|       | monotonicity | SR \(\varepsilon\) (3.2a) and CSR (4.23) | Proposition 4.13 |

We test the proposed rounding methods in section 5; we consider the optimization of quadratic functions, the training of a MLR and the training of a simple NN. Some conclusions are drawn in section 6.

2. Number representation system and rounding schemes. We start this section by recalling some basic properties of the floating-point arithmetic and the definition of CSR. Then, we introduce the SR\(\varepsilon\) and signed-SR\(\varepsilon\) schemes.

2.1. Floating-point representation. A floating-point system \([18]\) \(\mathbb{F} \subset \mathbb{R}\) is a proper subset of real numbers. A floating-point number \(\tilde{x} \in \mathbb{F}\) can be represented by radix (base) \(b\), precision \(s\), and exponent \(e\) \([14,\text{Sec. 2.1}]\), as

\[
\tilde{x} = \pm a \cdot b^{e-s},
\]

where \(a, b, e,\) and \(s\) are integers satisfying \(a \in [0, b^s - 1]\), \(b = 2\) (binary representation), and \(e \in [e_{\min}, e_{\max}]\). A technical standard is the IEEE Standard for Floating-Point Arithmetic (IEEE 754) \([18]\). Based on IEEE 754, there are five basic formats for binary computation, i.e., binary16 (half precision), binary32 (single precision), binary64 (double precision), binary128 (quadruple precision), and binary256 (octuple precision); see \([18,\text{Sec. 3.3}]\). In this paper, we use a 8-bit (binary8) and 16-bit (binary16) number formats with customized significand precisions and exponents for the numerical studies. In the numerical experiments, we employ binary32, binary16, and binary8, of which the number formats are summarized in Table 2.

Table 2
Number formats.

| Number format | Sign | Exponent | Significand field | Total bits |
|---------------|------|----------|-------------------|------------|
| binary32      | 1    | 8        | 23                | 32         |
| binary16      | 1    | 9        | 6                 | 16         |
| binary8       | 1    | 5        | 2                 | 8          |

We call rounding any map that associates with \(x \in \mathbb{R}\) a certain \(\tilde{x} \in \mathbb{F}\). The unit roundoff \(u := \frac{1}{2}b^{1-s}\) is the maximum relative error caused by approximating the real number \(x \in \mathbb{R}\) by \(\tilde{x} \in \mathbb{F}\) using RN. This paper focuses on floating-point arithmetic, with the roundoff error indicating the relative rounding error, and with the absolute rounding error specified if necessary. The default rounding mode used in IEEE 754 floating-point operations is RN with half to even. For the detailed description of floating-point number representation see \([18,\text{Sec. 3.3}]\).
Sec. 3] and [14, Sec. 2.1]. In this work, we focus on RN and the stochastic rounding methods that we outline in the next section.

2.2. Stochastic rounding. Throughout the paper we denote by \( \text{fl}(\cdot) \) a general rounding operator that maps \( x \in \mathbb{R} \) into \( \text{fl}(x) \in \mathbb{F} \). When a specific rounding scheme is applied, \( \text{fl}(\cdot) \) will be replaced by the corresponding rounding operator.

The most natural choice, for a rounding operation is to opt for one of the two floating-point numbers that are adjacent to \( x \). More precisely, rounding schemes choose \( \text{fl}(x) \in \{ \lfloor x \rfloor, \lceil x \rceil \} \) where \( \lfloor x \rfloor := \max\{ y \in \mathbb{F} : y \leq x \} \) and \( \lceil x \rceil := \min\{ y \in \mathbb{F} : y \geq x \} \). A stochastic rounding scheme chooses whether \( \text{fl}(x) = \lfloor x \rfloor \) or \( \text{fl}(x) = \lceil x \rceil \), according to a certain \( x \)-dependent probability. We write \( \sigma(x) := \text{fl}(x) - x \) and \( \delta(x) := (\text{fl}(x) - x)/x \) the absolute and relative errors, respectively. An appropriate superscript will be added when the latter quantities will refer to a specific rounding scheme.

We are now ready to introduce the CSR scheme.

Definition 2.1. (cf., e.g., [6]) For \( x \in \mathbb{R} \), the rounded value CSR\((x)\) is defined as

\[
\text{CSR}(x) = \begin{cases} 
\lfloor x \rfloor, & \text{with probability } p_c(x) := 1 - \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor}, \\
\lceil x \rceil, & \text{with probability } 1 - p_c(x) = \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor}.
\end{cases}
\]

CSR has rounding probability depending on its input \( x \), in such a way that 0 rounding bias is achieved, i.e., \( \mathbb{E}[\sigma_{\text{CSR}}(x)] = 0 \), for all \( x \). In each interval of the form \( (\lfloor x \rfloor, \lceil x \rceil) \), the probability of rounding towards 0 is inversely proportional to the magnitude of the input.

To preserve more information when dealing with small gradients, we propose to set a lower bound \( \varepsilon < 1 \) to the probability of rounding away from 0. More formally, we introduce a new stochastic rounding scheme, \( \text{SR}_{\varepsilon} \), as follows.

Definition 2.2. Given \( \varepsilon \in (0, 1) \), let us consider the following functions

\[
\eta(x, \varepsilon) := 1 - \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor} - \text{sign}(x) \varepsilon, \quad \varphi(y) = \begin{cases} 
0, & y \leq 0, \\
y, & 0 \leq y \leq 1, \\
1, & y \geq 1.
\end{cases}
\]

(2.1)

We indicate by \( p_\varepsilon(x) := \varphi(\eta(x, \varepsilon)) \) and we define

\[
\text{SR}_{\varepsilon}(x) = \begin{cases} 
\lfloor x \rfloor, & \text{with probability } p_\varepsilon(x), \\
\lceil x \rceil, & \text{with probability } 1 - p_\varepsilon(x).
\end{cases}
\]

(2.2)

For a fixed \( x \in \mathbb{R} \), \( \text{SR}_{\varepsilon}(x) \) is a discrete random variable with state space \( \{ \lfloor x \rfloor, \lceil x \rceil \} \). With a direct computation we get the following expression for the expected absolute rounding error:

\[
\mathbb{E}[\sigma_{\text{SR}_{\varepsilon}}(x)] = \begin{cases} 
\lfloor x \rfloor - x, & \eta(x, \varepsilon) > 1, \\
\text{sign}(x) \varepsilon (\lfloor x \rfloor - \lfloor x \rfloor), & 0 \leq \eta(x, \varepsilon) \leq 1, \\
\lceil x \rceil - x, & \eta(x, \varepsilon) < 0.
\end{cases}
\]

(2.3)
Since \( \eta(x,\varepsilon) > 1 \) and \( \eta(x,\varepsilon) < 0 \) may happen only for \( x < 0 \) and \( x > 0 \), respectively, the expected relative error \( \delta_{\text{SR}}(x) \) is nonnegative for all \( x \in \mathbb{R} \).

Figure 1. Comparison of \( \mathbb{E}[\text{fl}(y)] \) for \( y \in ([x], \lceil x \rceil) \) (for a fixed \( x \)) using different rounding schemes for \( x > 0 \) (a) and \( x < 0 \) (b).

Figure 1 plots the value of \( \mathbb{E}[\text{fl}(y)] \) as \( y \) ranges in the interval \((\lfloor x \rfloor, \lceil x \rceil)\), for the various rounding schemes introduced so far. It can be seen that when \( x > 0 \), \( \text{SR}_\varepsilon \) behaves as the combination of stochastic rounding and ceiling, while we have the combination of stochastic rounding and flooring when \( x < 0 \). When \( \varepsilon \leq 0.5 \), the deterministic rounding part of \( \text{SR}_\varepsilon \) behaves as \( \text{RN} \).

In Definition 2.2, \( \text{SR}_\varepsilon \) leads to a rounding bias with the same direction as its input. By introducing an additional variable \( v \in \mathbb{R} \) and with a minor modification of the function \( \eta \) in Definition 2.2, we obtain a new stochastic rounding method, namely signed-\( \text{SR}_\varepsilon \), with rounding bias in the opposite sign of \( v \). In the context of GD with low-precision representation, we will use this method to get a constant rounding bias in a descent direction by substituting the corresponding entries of the gradient vector for \( v \). Later in subsection 4.2.1, we will show how signed-\( \text{SR}_\varepsilon \) is beneficial for implementing GD. The signed-\( \text{SR}_\varepsilon \) method is defined as follows.

Definition 2.3. Let sign\((v)\) be a desired sign of rounding bias, \( \varphi \) be the function introduced in Definition 2.2, and

\[
\tilde{\eta}(x,\varepsilon,v) := 1 - \frac{x - \lfloor x \rfloor}{\lfloor x \rfloor - \lceil x \rceil} + \text{sign}(v)\varepsilon.
\]

We indicate with \( \tilde{\rho}_\varepsilon(x) := \varphi(\tilde{\eta}(x,\varepsilon,v)) \) and we define (cf. (2.2))

\[
\text{signed-}\text{SR}_\varepsilon(x) = \begin{cases} 
\lfloor x \rfloor, & \text{with probability } \tilde{\rho}_\varepsilon(x), \\
\lceil x \rceil, & \text{with probability } 1 - \tilde{\rho}_\varepsilon(x).
\end{cases}
\]

By a direct computation we get the following expression for the expected absolute rounding error for signed-\( \text{SR}_\varepsilon \) (cf. (2.3)):

\[
(2.4) \quad \mathbb{E}[\sigma_{\text{signed-}\text{SR}_\varepsilon}(x)] = \begin{cases} 
[x] - x, & \tilde{\eta}(x,\varepsilon,v) > 1, \\
\text{sign}(-v)\varepsilon([x] - \lfloor x \rfloor), & 0 \leq \tilde{\eta}(x,\varepsilon,v) \leq 1, \\
[x] - x, & \tilde{\eta}(x,\varepsilon,v) < 0.
\end{cases}
\]
From (2.4) it can be seen that when \(0 \leq \tilde{\eta}(x, \varepsilon, v) \leq 1\), the expected absolute rounding error has always the opposite sign as \(v\). Therefore, we may control the sign of rounding bias by changing the sign of \(v\). When using signed-SR\(_\varepsilon\) to implement GD, one can achieve a rounding bias in a descent direction by replacing \(v\) with the gradient.

### 2.3. Arithmetic operations with SR\(_\varepsilon\) and signed-SR\(_\varepsilon\)

Standard models of floating-point operations are based on RN and satisfy [14, Sec. 2.2]

\[
\begin{align*}
\text{RN}(x) &= x(1 + \delta), \\
|\delta| &\leq u, \\
\text{RN}(x \text{ op } y) &= (x \text{ op } y)(1 + \delta), \\
|\delta| &\leq u,
\end{align*}
\]

for \(\text{op} \in \{+, -, \ast, /, \sqrt{\cdot}\}\), where \(u \ll 1\).

For CSR, (2.5) holds when replacing \(u\) by \(2u\); see [6, (2.4)]. For SR\(_\varepsilon\) we can identify the following two cases due to the role of \(\varepsilon\) in the rounding probability:

\[
\begin{align*}
\text{SR}\_\varepsilon(x) &= x(1 + \delta), \\
&\begin{cases}
|\delta| \leq 2u, \\
0 \leq \eta(x, \varepsilon) \leq 1, \\
0 \leq \delta \leq 2\varepsilon u, &\text{otherwise}
\end{cases}
\end{align*}
\]

For \(\text{SR}\_\varepsilon(x \text{ op } y)\) the same bounds as in (2.6) hold for \(\text{op} \in \{+, -, \ast, /, \sqrt{\cdot}\}\). Finally, the same bounds that hold for \(\text{SR}\_\varepsilon\) also apply to signed-SR\(_\varepsilon\).

In the next section, we briefly recall the convergence property of GD for exact computation and extend the convergence proof for low-precision floating-point arithmetic with the operation properties (2.5) and (2.6).

### 3. Gradient Descent with floating point numbers

GD is a first-order optimization method for finding a local minimum of a differentiable function \(f : \mathbb{R}^n \rightarrow \mathbb{R}\). The algorithm iteratively updates in the opposite direction of the gradient with the rule

\[
\begin{align*}
x^{(k+1)} &= x^{(k)} - t \nabla f(x^{(k)}),
\end{align*}
\]

where \(t > 0\) is the stepsize. The choice of \(t\) may affect the convergence of GD and plenty of strategies have been proposed, e.g., Cauchy stepsize, Barzilai and Borwein stepsize, alternate step, and combination of Cauchy steps with fixed stepsizes; see, e.g., [35] for a recent review.

In this paper, we study the influence of rounding errors for GD with a fixed stepsize. Further, we will show that when a very low-precision computation (e.g., 8-bit floating-point representation) is employed together with SR\(_\varepsilon\), the choice of the stepsize can hardly affect the convergence of GD.

#### 3.1. Source of rounding errors

From now on we denote by \(\tilde{x}^{(k)}\) the sequence generated by GD in finite precision. When implementing GD with floating-point numbers, there are three sources of roundoff errors in the evaluation of (3.1) that we have to take into account: the error \(\delta_1\) arising from the evaluation of the gradient, \(\delta_2\) coming from the multiplication with the stepsize \(t\) (see, e.g., (3.2a)), and \(\delta_3\) caused by the final subtraction (see, e.g., (3.2b)). Note that the magnitudes of \(\delta_1\) and \(\delta_2\) are affected by the parameter \(t\) that is user-selected. However, \(\delta_3\) does not depend on any external parameter but only on \(u\). For this reason, we
split the GD iteration in the following two steps

\begin{equation}
\begin{align}
    z^{(k+1)} &= \tilde{x}^{(k)} - t \nabla f(\tilde{x}^{(k)}) \circ (1 + \delta_1^{(k)}) \circ (1 + \delta_2^{(k)}), \\
    \tilde{x}^{(k+1)} &= z^{(k+1)} \circ (1 + \delta_3^{(k)}),
\end{align}
\end{equation}

where \(1\) is the vector of all ones, \(\circ\) indicates the component-wise product of vectors, and we denote

\[ h_m^{(k)} = 1 + \delta_m^{(k)}, \quad m = 1, 2, 3. \]

The magnitudes of the entries of \(\delta_2^{(k)}, \delta_3^{(k)}\) are bounded by either \(u\) or \(2u\); see, e.g., (2.5) and (2.6). The entries of \(\delta_1^{(k)}\) satisfy the bound \(\|\delta_1^{(k)}\|_\infty \leq \tilde{c}u\) where \(\tilde{c}\) depends on the number of basic arithmetic operations for evaluating \(\nabla f\). This implies that there exists a constant \(c > 0\), depending only on the rounding method, such that for each iteration \(k\)

\begin{equation}
\|h_1^{(k)} \circ h_2^{(k)}\|_\infty \leq 1 + cu.
\end{equation}

Specifically, when \(u \ll 1\), we have:

\[ c = \begin{cases} 
    [(1 + 2u)(1 + 2\tilde{c}u) - 1]/u < 3 + 2\tilde{c} & \text{for CSR and SR}, \\
    [(1 + u)(1 + \tilde{c}u) - 1]/u < 2 + \tilde{c} & \text{for RN}.
\end{cases} \]

Moreover, we define

\begin{equation}
\gamma_k := \min_{i=1,...,n} h_{1,i}^{(k)} h_{2,i}^{(k)},
\end{equation}

that indicates the minimum ratio between the exact entries and rounded entries of \(t \nabla f(\tilde{x}^{(k)})\) at the \(k\)th iterate. Note that \(\gamma_k\) satisfies \(1 - cu \leq \gamma_k \leq 1 + cu\). Finally, we remark that all the entries in \(h_1^{(k)}\) are positive, since all the rounding schemes mentioned in subsection 2.2 do not change the sign of the input.

**3.2. Stagnation of GD with RN.** For every run of GD with limited precision and RN, after a certain number of iterations, stagnation may happen due to rounding, i.e., \(\tilde{x}^{(k+1)} = RN(\tilde{x}^{(k)} - RN(t RN(\nabla f(\tilde{x}^{(k)}) )))) = \tilde{x}^{(k)}\). Usually, this phenomenon happens earlier, with respect to the number of iterations, with low-precision computations. Let us have a closer look at this situation; we denote by \(\tilde{x}_i^{(k)} = \mu_i^{(k)} 2^{a_i^{(k)} - s}\), where \(\mu_i^{(k)} \in [2^{s-1}, 2^s)\) and \(a_i^{(k)} \in \mathbb{N}\), the floating point representation of the \(i\)th entry of \(\tilde{x}^{(k)}\). An easy consequence of [14, Thm. 2.2] is the following result.

**Proposition 3.1.** When \(RN(t RN(\nabla f(\tilde{x}^{(k)}))) \leq 2^{a_i^{(k)} - s - 1}\) and the least significant bit of \(\tilde{x}_i^{(k)}\) is 0 (even), we have \(\tilde{x}_i^{(k+1)} = RN(\tilde{x}_i^{(k)} - RN(t RN(\nabla f(\tilde{x}^{(k)})))) = \tilde{x}_i^{(k)}\).

In the spirit of Proposition 3.1, we consider the following quantity:

\[ \tau_k := \max_{i=1,...,n} \gamma_i^{(k)}, \quad \gamma_i^{(k)} = \begin{cases} 
    0 & \text{if the assumptions of Proposition 3.1 hold}, \\
    \frac{RN(t \nabla f(\tilde{x}^{(k)})))}{2^{a_i^{(k)}}} & \text{otherwise}.
\end{cases} \]
Then, the condition \( \tau_k < \frac{1}{2}u \) implies that the GD iteration with RN stagnates. As an illustrative example, in Figure 2a we show the trajectory of \( \tilde{x}(k) \) when minimizing \( f(x) = x^2 \) using GD with RN and 8-bit floating point numbers (\( u = 2^{-3} \), for the details of number format see Table 2). When \( k \geq 48 \), the method stagnates as \( \tau_k = 0 \); see Figure 2b.

In view of the phenomenon shown in Figure 2a, we divide the analysis of the convergence of GD in two stages. Stage I focuses on the initial phase of the method where \( \tau_k \geq \frac{1}{2}u \) in all iteration steps, so that RN has not caused the stagnation of GD yet. In Stage II we assume that the condition \( \tau_k < \frac{1}{2}u \) has been matched already for a certain \( k \), and it likely represents the behavior of GD close to a critical point.

In the next section, we analyze various choices of rounding schemes in steps (3.2a) and (3.2b), for both stages. In Stage I, the updating step is dominated by the gradient part; so we only analyze the use of CSR for (3.2b) as the differences with respect to the other rounding schemes are negligible. In Stage II, the update mainly depends on the scheme adopted for evaluating (3.2b). Therefore, we analyze both CSR and signed-SR\(_e\) for (3.2b). Concerning (3.2a), we study the effect of RN, CSR, and SR\(_e\) in both stages; note that when the input has the same direction of the gradient, signed-SR\(_e\) and SR\(_e\) perform the same, e.g., using signed-SR\(_e\) and SR\(_e\) for (3.2a) yields similar rounding errors, both oriented in a descent direction. A summary of the analysis is reported in Table 3.

### Table 3

| Step    | Stage I       | Stage II     |
|---------|---------------|--------------|
| (3.2a)  | RN, CSR, SR\(_e\) | RN, CSR, SR\(_e\) |
| (3.2b)  | CSR           | CSR, signed-SR\(_e\) |
4. Convergence analysis of GD for convex problems. Throughout this section we consider the unconstrained optimization problem

$$\arg \min_{x \in \mathbb{R}^n} f(x),$$

of which the objective function is assumed to satisfy the following condition.

**Assumption 4.1.** The function $f$ is convex and its gradient $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is Lipschitz continuous with constant $L > 0$.

This means that, for all $x, y \in \mathbb{R}^n$, $f$ satisfies:

\begin{align}
\tag{4.1}
f(y) & \geq f(x) + \nabla f(x)^T (y - x), \\
\tag{4.2}
f(y) & \leq f(x) + \nabla f(x)^T (y - x) + \frac{1}{2} L \|y - x\|^2,
\end{align}

where $\|\cdot\|$ indicates the Euclidean norm.

The proof of (4.2) can be found in, e.g., [26, Thm. 2.1.5]. Further, we denote by $x^* \in \mathbb{R}^n$ the minimizer of $f$. When exact arithmetic is employed, the convergence rate of GD is at least sublinear with respect to the number of iterations, as the following result shows.

**Theorem 4.2.** ([26, Thm. 2.1.14, Cor. 2.1.2]) Under Assumption 4.1, the $k$th iterate of the gradient descent method with a fixed stepsize $t \leq 1/L$ satisfies the following inequality:

$$f(x^{(k)}) - f(x^*) \leq \frac{2L}{4 + Lt} \|x^{(0)} - x^*\|^2.$$

Theorem 4.2 ensures that, in exact arithmetic, GD asymptotically converges to the optimum. However, when implementing GD in floating-point arithmetic, the method may only converge to some level of accuracy depending on the rounding precision. In the remainder of this section, we analyze the convergence of GD in two stages, i.e., where GD does not stagnate with RN and where it does.

**4.1. Stage I: RN does not cause stagnation.** Let us start to analyze the convergence of GD with roundoff errors. For the moment we do not restrict to a specific rounding scheme and we look at the conditions that guarantee the monotonicity of the method. Based on (3.2), we denote by

$$\tag{4.4} \theta^{(k)} := f(\tilde{x}^{(k+1)}) - f(z^{(k+1)}),$$

the effect of the third roundoff error $\delta_3^{(k)}$ on the objective function value. Then, we state the following result that links the monotonicity of GD to the values of $u$ and $t$.

**Lemma 4.3.** Let the objective function $f$ satisfy Assumption 4.1. Assume that $k$ iterations of GD have been carried out with a fixed stepsize $t$ such that $t \left(1 + cu\right) \leq 1/L$, where $c$ is defined in (3.3). If $u$ satisfies

$$\tag{4.5} u \leq \frac{t \gamma_{k-1} \|\nabla f(\tilde{x}^{(k-1)})\|^2}{4 \|\nabla f(\tilde{x}^{(k)})\| \|z^{(k)}\|},$$

where $\gamma_{k-1}$ is defined as in (3.4) and $z^{(k)}$ is as in (3.2a), then $f(\tilde{x}^{(k)}) \leq f(\tilde{x}^{(k-1)})$, for $k > 0$. 

Proof. Since $\nabla f$ is Lipschitz continuous with constant $L$, combining the updating rule with roundoff errors (3.2), property (4.2), and the inequality $t (1 + c u) \leq 1/L$, we have

$$f(\tilde{x}^{(k+1)}) = f(z^{(k+1)}) + \theta^{(k)}$$

$$\leq f(\tilde{x}^{(k)}) - t \nabla f(\tilde{x}^{(k)})^T(\nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}) + \frac{1}{2} L \|t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}\|^2 + \theta^{(k)}$$

$$= f(\tilde{x}^{(k)}) - t \sum_{i=1}^{n} h_{1,i}^{(k)} h_{2,i}^{(k)} (1 - \frac{1}{2} L t h_{1,i}^{(k)} h_{2,i}^{(k)}) (\nabla f(\tilde{x}^{(k)}))_i^2 + \theta^{(k)}$$

$$\leq f(\tilde{x}^{(k)}) - \frac{1}{2} t \sum_{i=1}^{n} h_{1,i}^{(k)} h_{2,i}^{(k)} (\nabla f(\tilde{x}^{(k)}))_i^2 + \theta^{(k)}$$

$$\leq f(\tilde{x}^{(k)}) - \frac{1}{2} t \gamma_k \| \nabla f(\tilde{x}^{(k)}) \|^2 + \theta^{(k)}. \tag{4.6}$$

Since $f$ is convex, based on (4.1) and (4.4), we obtain the following inequality for the $k$th iterate

$$f(\tilde{x}^{(k-1)}) - f(\tilde{x}^{(k)}) \geq \nabla f(\tilde{x}^{(k)})(z^{(k)} - \tilde{x}^{(k)}) + \frac{1}{2} t \gamma_{k-1} \| \nabla f(\tilde{x}^{(k-1)}) \|^2$$

$$\geq -2 u \| z^{(k)} \| \| \nabla f(\tilde{x}^{(k)}) \| + \frac{1}{2} t \gamma_{k-1} \| \nabla f(\tilde{x}^{(k-1)}) \|^2$$

$$\geq 0, \quad \text{in view of (4.5).} \qedhere$$

Condition (4.5) may be viewed as either an upper bound on $u$ or a lower bound on $t$, depending on the setting. When implementing GD in a given low precision, the stepsize has to satisfy both (4.5) and $t (1 + c u) \leq 1/L$. On the other hand, if $t$ is fixed, $u$ needs to be small enough to satisfy (4.5). Note that, when the strict inequality holds in (4.5), the convergence is strictly monotonic. When $\theta^{(k-1)} \leq 0$ it is sufficient to require $t (1 + c u) \leq 2/L$ to get $f(\tilde{x}^{(k)}) \leq f(\tilde{x}^{(k-1)})$; this can be shown with an analogous argument to the one used to derive (4.6).

We now address the generalization of Theorem 4.2 for the updating rule with rounding errors (3.2). The core idea is to adjust the strategy used in the proof of [26, Thm. 2.1.14, Cor. 2.1.2] to our setting. If $u$ is small enough, we ensure a similar convergence rate $O(1/k)$ and we show that a better multiplicative constant may be obtained when the accumulated absolute rounding errors are in a descent direction.

Before stating the result we introduce the largest distance between the iterates of GD and the minimizer $x^*$, and the best approximation to the optimal value:

$$\chi := \max_{j=0,\ldots,k} \| \tilde{x}^{(j)} - x^* \|, \quad \zeta := \min_{j=0,\ldots,k} f(\tilde{x}^{(j)}) - f(x^*).$$

In the following theorem, we require $u$ to satisfy a bound slightly stricter than (4.5); in particular this guarantees the monotonicity of the GD iterations. Moreover, we will introduce a quantity $Q_j$ that interprets the relation between $\theta^{(j)}$, $\chi$, and $\zeta$. When $\chi$ and $\zeta$ are given, $Q_j$ has the same sign as and is proportional to $\theta^{(j)}$. This indicates that $Q_j$ has the same effect as $\theta^{(j)}$ on the convergence of GD. We will discuss more details after Theorem 4.4.
Theorem 4.4. Let the objective function \( f \) satisfy Assumption 4.1. Assume that \( k \) iterations of GD have been carried out with a fixed stepsize \( t \) such that \( t(1 + cu) \leq 1/L \). If \( u \) satisfies

\[
(4.7) \quad u \leq \frac{t \gamma_j \zeta^2}{4 \chi^2 \|\nabla f(\bar{x}^{(j)})\| \|z^{(j)}\|}, \quad j = 0, \ldots, k - 1,
\]

then it holds:

\[
(4.8) \quad f(\bar{x}^{(k)}) - f(x^*) \leq \frac{2L\chi^2}{4 + L t \sum_{j=0}^{k-1} (\gamma_j - Q_j)}, \quad Q_j := \frac{2\chi^2 \theta^{(j)}}{t \zeta^2}.
\]

Proof. We follow the main line of the proof in [26, Thm. 2.1.14, Cor. 2.1.2]. In view of (4.1), we have

\[
(4.9) \quad \zeta \leq f(\bar{x}^{(k)}) - f(x^*) \leq \nabla f(\bar{x}^{(k)})^T (\bar{x}^{(k)} - x^*) \leq \|\nabla f(\bar{x}^{(k)})\| \chi.
\]

Based on (4.9), it is easy to check that (4.7) implies (4.5). Based on (4.6) and (4.9), we have

\[
f(\bar{x}^{(k+1)}) - f(x^*) \leq f(\bar{x}^{(k)}) - f(x^*) - \frac{1}{2}t \gamma_k \|\nabla f(\bar{x}^{(k)})\|^2 + \theta^{(k)}
\]

\[
\leq f(\bar{x}^{(k)}) - f(x^*) - \frac{t \gamma_k}{2\chi^2} (f(\bar{x}^{(k)}) - f(x^*))^2 + \theta^{(k)}.
\]

Dividing both sides by \((f(\bar{x}^{(k+1)}) - f(x^*)) (f(\bar{x}^{(k)}) - f(x^*))\), we obtain

\[
\frac{1}{f(\bar{x}^{(k)}) - f(x^*)} \leq \frac{1}{f(\bar{x}^{(k+1)}) - f(x^*)} - \frac{t \gamma_k}{2\chi^2} \frac{f(\bar{x}^{(k)}) - f(x^*)}{f(\bar{x}^{(k+1)}) - f(x^*)}
\]

\[
\quad + \left( \frac{f(\bar{x}^{(k+1)}) - f(x^*)}{f(\bar{x}^{(k+1)}) - f(x^*) - \theta^{(k)}} \right)
\]

\[
\leq \frac{1}{f(\bar{x}^{(k+1)}) - f(x^*)} - \frac{t \gamma_k}{2\chi^2} + \frac{\theta^{(k)}}{\zeta^2}.
\]

Note that the convexity of \( f \) and (4.7) yield \( \frac{t \gamma_j}{2\chi^2} - \frac{\theta^{(j)}}{\zeta^2} \geq 0 \), for \( j = 0, \ldots, k - 1 \). Expanding the recursion \( k \) times, we obtain

\[
\frac{1}{f(\bar{x}^{(k)}) - f(x^*)} \geq \frac{1}{f(x^{(0)}) - f(x^*)} + \frac{t}{2\chi^2} \sum_{j=0}^{k-1} (\gamma_j - \frac{2\chi^2 \theta^{(j)}}{t \zeta^2})
\]

\[
= \frac{1}{f(x^{(0)}) - f(x^*)} + \frac{t}{2\chi^2} \sum_{j=0}^{k-1} (\gamma_j - Q_j).
\]

Property (4.2) and \( \nabla f(x^*) = 0 \) implies

\[
(4.10) \quad f(x^{(0)}) - f(x^*) \geq \frac{2}{L \|x^{(0)} - x^*\|^2}.
\]

\[
(4.11) \quad \frac{1}{f(x^{(0)}) - f(x^*)} \geq \frac{2}{L \|x^{(0)} - x^*\|^2}.
\]
Applying (4.11) to (4.10), we obtain
\[
\frac{1}{f(\tilde{x}^{(k)}) - f(x^*)} \geq \frac{2}{L\|x^{(0)} - x^*\|^2} + \frac{t}{2\chi^2} \sum_{j=0}^{k-1} (\gamma_j - Q_j).
\]

Therefore, we have
\[
f(\tilde{x}^{(k)}) - f(x^*) \leq \frac{2L\chi^2 \|x^{(0)} - x^*\|^2}{4\chi^2 + L \|x^{(0)} - x^*\|^2 \sum_{j=0}^{k-1} (\gamma_j - Q_j)}
\]
\[
\leq \frac{2L\chi^2}{4 + L \sum_{j=0}^{k-1} (\gamma_j - Q_j)}.
\]

(4.12)

We remark that, in exact arithmetic, we have $Q_j = 0$ and $\gamma_j = 1$ for all $j = 0, \ldots, k - 1$; in particular, the statement of Theorem 4.4 (cf. (4.8)) is equivalent to the same convergence rate ensured by Theorem 4.2 (cf. (4.3)). In floating-point arithmetic, the quantities $\gamma_j$ and $Q_j$ depend on the adopted rounding schemes. The condition $\gamma_j > 1$ indicates that the accumulated absolute rounding errors in (3.2a) are oriented towards a descent direction. An analogous conclusion holds for the condition $Q_j < 0$ and (3.2b). Together, $\gamma_j - Q_j > 1$ indicates that the accumulated absolute rounding error in (3.2) is in a descent direction.

The control of the quantities $\gamma_j - Q_j$ with a deterministic strategy appears difficult to realize and computationally expensive. For instance, to ensure $Q_j < 0$ one would need to switch the rounding scheme between floor and ceiling to match the condition:

$$\text{sign}(\nabla f(\tilde{x}^{(j)})) = -\text{sign}(\delta^{(j-1)} \circ x^{(j)}),$$

where the sign function is applied component-wise.

A better way to analyze and possibly control the quantities $\gamma_j - Q_j$ is to rely on stochastic rounding methods. In the next result we show that, in expectation, we have $E[\gamma_j - Q_j] > 1$, for all $j$, by employing SR$_\varepsilon$ for (3.2a) and CSR for (3.2b).

**Theorem 4.5.** Under the same assumptions as in Theorem 4.4, if (3.2a) is computed using SR$_\varepsilon$ and (3.2b) is computed using CSR, then there exists a $\Delta > 0$ such that:

\[
E[f(\tilde{x}^{(k)}) - f(x^*)] \leq \frac{2L\chi^2}{4 + L k (t + \Delta)}.
\]

(4.13)

We remark that $\Delta$ is proportional to $t \varepsilon u$ and depends on the accumulated expected roundoff error. Larger values of $t, \varepsilon$, and $u$ lead to larger $\Delta$, which in turn may lead to a faster convergence rate.

**Proof.** The absolute roundoff errors in evaluating (3.2) are given by $\sigma_1^{(k)} = t \nabla f(\tilde{x}^{(k)}) \circ \delta_1^{(k)}$, $\sigma_2^{(k)} = t \nabla f(\tilde{x}^{(k)}) \circ (1 + \delta_1^{(k)}) \circ \delta_2^{(k)}$, and $\sigma_3^{(k)} = z^{(k+1)} \circ \delta_3^{(k)}$. Based on (4.6), we obtain

\[
f(z^{(k+1)}) \leq f(\tilde{x}^{(k)}) - \frac{1}{2} \sum_{i=1}^n h_{1,i}^{(k)} h_{2,i}^{(k)} (\nabla f(\tilde{x}^{(k)}))_i^2
\]
\[
= f(\tilde{x}^{(k)}) - \frac{1}{2} t \|\nabla f(\tilde{x}^{(k)})\|^2 - \frac{1}{2} \sum_{i=1}^n (\sigma_{1,i}^{(k)} + \sigma_{2,i}^{(k)}) \nabla f(\tilde{x}^{(k)})_i.
\]

(4.14)
Subtracting \( f(x^*) \) from both sides and taking the expectation of (4.14), we obtain
\[
E \left[ f(z^{k+1}) - f(x^*) \right] \leq E \left[ f(\bar{x}^{(k)}) - f(x^*) \right] - \frac{1}{2} t E \left[ \| \nabla f(\bar{x}^{(k)}) \|^2 \right]
\]
(4.15)
\[
\quad - \frac{1}{2} \sum_{i=1}^{n} \left( E \left[ \sigma_{1,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] + E \left[ \sigma_{2,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] \right).
\]

We denote by \( S = S_1 \cup S_2 \cup S_3 \) the finite set of values that the \( i \)th component of \( \nabla f(\bar{x}^{(k)}) \) can assume; \( S_1 \) denotes the subset of \( S \) of all the possible values of \( \nabla f(\bar{x}^{(k)})_i \) that satisfy \( \varphi(\eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) = \eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) \) in Definition 2.2. Analogously, we define \( S_2 \) for the condition \( \varphi(\eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) = 0 \) and \( S_3 \) for \( \varphi(\eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) = 1 \). By applying the law of total expectation, we have
\[
E \left[ \sigma_{1,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] = \sum_{j=1}^{3} \sum_{\nabla f(\bar{x}^{(k)})_i \in S_j} E \left[ \sigma_{1,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \mid S_j \right] P(S_j)
\]
\[
\quad = t \sum_{\nabla f(\bar{x}^{(k)})_i \in S_1} \left( [\nabla f(\bar{x}^{(k)})_i] - [\nabla f(\bar{x}^{(k)})_i] \right) \text{sign}(\nabla f(\bar{x}^{(k)})_i) \varepsilon \nabla f(\bar{x}^{(k)})_i P(S_1)
\]
\[
\quad + t \sum_{\nabla f(\bar{x}^{(k)})_i \in S_2} \left( [\nabla f(\bar{x}^{(k)})_i] - \nabla f(\bar{x}^{(k)})_i \right) \nabla f(\bar{x}^{(k)})_i P(S_2)
\]
\[
\quad + t \sum_{\nabla f(\bar{x}^{(k)})_i \in S_3} \left( [\nabla f(\bar{x}^{(k)})_i] - \nabla f(\bar{x}^{(k)})_i \right) \nabla f(\bar{x}^{(k)})_i P(S_3).
\]
By Definition 2.2, \( \varphi(\eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) = 0 \) only holds for \( \nabla f(\bar{x}^{(k)})_i > 0 \) and \( \varphi(\eta(\nabla f(\bar{x}^{(k)})_i, \varepsilon) = 1 \) only holds for \( \nabla f(\bar{x}^{(k)})_i < 0 \), thus we have \( E \left[ \sigma_{1,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] > 0 \). With a similar argument we also have \( E \left[ \sigma_{2,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] > 0 \).

Denote by
\[
\lambda_k := \min_{i=1, \ldots, n} E \left[ \sigma_{1,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] + E \left[ \sigma_{2,i}^{(k)} \nabla f(\bar{x}^{(k)})_i \right] > 0,
\]
then, we have
\[
E \left[ f(z^{k+1}) - f(x^*) \right] \leq E \left[ f(\bar{x}^{(k)}) - f(x^*) \right] - \frac{1}{2} E \left[ \| \nabla f(\bar{x}^{(k)}) \|^2 \right] - \frac{n}{2} \lambda_k.
\]
In view of (4.9), we have
\[
E \left[ \| \nabla f(\bar{x}^{(k)}) \|^2 \right] \geq \frac{E \left[ (f(\bar{x}^{(k)}) - f(x^*))^2 \right]}{\lambda^2}.
\]
When \( \sigma_3^{(k)} \) is obtained by CSR, we get zero mean independent errors [6, Lemma 5.2]; in particular, we have \( E \left[ f(\bar{x}^{(k+1)}) \right] - E \left[ f(\bar{x}^{(k+1)}) \right] = 0 \). Therefore, substituting (4.18) into (4.17), we obtain
\[
E \left[ f(z^{k+1}) - f(x^*) \right] \leq E \left[ f(\bar{x}^{(k)}) - f(x^*) \right] - \frac{n \lambda_k}{2} - t E \left[ (f(\bar{x}^{(k)}) - f(x^*))^2 \right] \frac{1}{2 \lambda^2}
\]
\[
\leq E \left[ f(\bar{x}^{(k)}) - f(x^*) \right] - \frac{n \lambda_k}{2} - t \left( E \left[ f(\bar{x}^{(k)}) - f(x^*) \right]^2 \right) \frac{1}{2 \lambda^2}.
\]
where the last relation is implied by Jensen’s inequality; see, e.g., [20, Lemma 5.3.1]. Dividing both sides of (4.19) by $E[f(\tilde{x}^{(k+1)}) - f(x^*)]$ we obtain

$$
\frac{1}{E[f(\tilde{x}^{(k)}) - f(x^*)]} \leq \frac{1}{E[f(\tilde{x}^{(k+1)}) - f(x^*)]} - \frac{t}{2} \frac{E[f(\tilde{x}^{(k+1)}) - f(x^*)]}{2E[f(\tilde{x}^{(k)}) - f(x^*)]}
$$

Denote $\alpha := \min_{j=1,\ldots,k} \frac{n\lambda_j}{E[f(\tilde{x}^{(j+1)}) - f(x^*)]E[f(\tilde{x}^{(j)}) - f(x^*)]}$, then we have

$$
(4.20) \quad \frac{1}{E[f(\tilde{x}^{(k)}) - f(x^*)]} \leq \frac{1}{E[f(\tilde{x}^{(k+1)}) - f(x^*)]} - \frac{t}{2\chi^2} - \frac{\alpha}{2}.
$$

Expanding the recursion of (4.20) and based on (4.11) until the $k$th iteration, we obtain

$$
\frac{1}{E[f(\tilde{x}^{(k)}) - f(x^*)]} \geq \frac{2}{L\|x^{(0)} - x^*\|^2} + \frac{kt}{2\chi^2} + \frac{k\alpha}{2}.
$$

Denoting $\Delta = \alpha\chi^2$, proceeding similarly as for (4.12), we find

$$
E[f(\tilde{x}^{(k)}) - f(x^*)] \leq \frac{2L\chi^2}{4 + Lk(t + \Delta)}.
$$

Based on (4.17) and the property $\lambda_j > 0$, for all $j$, we have $\alpha > 0$, which implies $\Delta > 0$. 

In the above proof, the condition $\varepsilon > 0$ is crucial to ensure $\Delta > 0$, i.e., a tighter convergence rate, in expectation, with respect to Theorem 4.2. Indeed, when CSR (SR$\varepsilon$ with $\varepsilon = 0$) is applied for computing both steps in (3.2), we get $E[\sigma_{1,i}^k \nabla f(\tilde{x}^{(k)})] = E[\sigma_{2,i}^k \nabla f(\tilde{x}^{(k)})] = 0$, which yields $\lambda_k = 0$ in (4.17). Comparing (4.3) and (4.13), this gives the same decay rate of Theorem 4.2, in expectation.

**Corollary 4.6.** Under the same assumptions of Theorem 4.4, if we use CSR in both steps of (3.2), we have that

$$
(4.21) \quad E[f(\tilde{x}^{(k)}) - f(x^*)] \leq \frac{2L\chi^2}{4 + Ltk}.
$$

### 4.2. Stage II: RN causes stagnation

As we discussed in subsection 3.1, the stagnation of GD with RN may be due to two reasons; the first one is that the rounded evaluation of the gradient is exactly 0, i.e., $t \nabla f(\tilde{z}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} = 0$ in (3.2a). Unless we have reached an exact stationary point, this is unlikely since subnormal numbers are supported in most floating-point formats. More likely, stagnation occurs when performing the rounding at (3.2b), i.e., $\tilde{x}^{(k+1)} = \text{fl}(\tilde{z}^{(k+1)}) = \tilde{x}^{(k)}$. With RN, this is unavoidable for values of the gradient that are relatively small with respect to the current iterate $\tilde{x}^{(k)}$. In this section, we show that with stochastic rounding, GD can still update with respect to the rounding errors until a certain level of accuracy.
In subsection 3.2, we identified the parameter $\tau_k$ to distinguish the area where RN stagnates. However, computing $\tau_k$ is impractical as it requires the exact (or accurate) value of the gradient update. Here, we propose to analyze the convergence of GD under a slightly weaker assumption on $u$ that is only based on floating-point numbers at the working accuracy. Proposition 3.1 implies the inequality $t |\nabla f(\tilde{x}^{(k)}), h_{1,i}^{(k)}h_{2,i}^{(k)}| < u |\tilde{x}_i^{(k)}|$ and, in turn, gives the following condition.

**Condition of Stage II** When stagnation happens at the $k_0$th iteration of GD with RN, for $k \geq k_0$, the rounding precision satisfies

$$u > t \frac{\|\nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}\|}{\|\tilde{x}^{(k)}\|}.$$  

(4.22)

Inequality (4.22) says that the rounding errors in (3.2a) are less important than those in (3.2b) because they are so small that only their direction (sign) affects the update. More precisely, under Condition of Stage II, the magnitudes of the GD updates are constrained to the level of rounding errors, and we can rewrite (3.2b) as

$$\tilde{x}^{(k+1)} = \text{fl}(z^{(k+1)}) = \tilde{x}^{(k)} - d^{(k)},$$

(4.23)

where each component of $d^{(k)}$ is either 0 or equal to the distance to a floating point number that is adjacent to the corresponding entry in $\tilde{x}^{(k)}$. For this reason, in this section we do not specify the rounding strategy for (3.2a) and we focus on the effect of various stochastic rounding methods, CSR, SR$_\varepsilon$, and signed-SR$_\varepsilon$, for evaluating (3.2b). Note that, the nonzero entries of $d^{(k)}$ point in the same direction as the corresponding entries of the gradient, so that $\nabla f(\tilde{x}^{(k)})^T d^{(k)} \geq 0$ for all $k$. In particular, (4.23) performs similarly to the sign gradient descent method [24], rather than conventional GD; see, e.g., (3.1).

Independently of the specific stochastic rounding method, we can adapt Lemma 4.3 and guarantee the monotonicity of GD when $u$ satisfies the following condition.

**Lemma 4.7.** Under Assumption 4.1 and Condition of Stage II, if $u$ satisfies

$$u \leq \frac{1}{\sqrt{2L}} \sqrt{\frac{\nabla f(\tilde{x}^{(k-1)})^T d^{(k-1)}}{\|\tilde{x}^{(k-1)}\|^2}},$$

(4.24)

then $f(\tilde{x}^{(k)}) \leq f(\tilde{x}^{(k-1)})$, for all $k > 0$.

**Proof.** Based on (4.2), (4.23), and the bound $2u |\tilde{x}_i^{(k)}| \geq |d_i^{(k)}|$, we obtain

$$f(\tilde{x}^{(k+1)}) \leq f(\tilde{x}^{(k)}) - \nabla f(\tilde{x}^{(k)})^T d^{(k)} + \frac{1}{2}L \|d^{(k)}\|^2$$
$$\leq f(\tilde{x}^{(k)}) - \nabla f(\tilde{x}^{(k)})^T d^{(k)} + 2u^2L \|\tilde{x}^{(k)}\|^2.$$  

(4.25)

Finally, inequality (4.24) implies $f(\tilde{x}^{(k)}) \leq f(\tilde{x}^{(k-1)})$ for all $k > 0.$

Looking at (4.24) one might wonder if it is possible to enforce monotonicity by shifting the objective function so that the iterates $\tilde{x}^{(k)}$ have smaller norms. More precisely, one could
stop GD at a certain iteration \( \bar{k} \) and restart GD for the objective function \( \bar{f}(x) := f(\bar{x}^{(k)} + x) \) and starting guess \( 0 \). However, this only moves the problem in the evaluation of \( f(x) \); for instance, if GD with RN stagnates then this will happen again because RN(\( \bar{x}^{(k)} + x \)) = \( \bar{x}^{(k)} \). Increasing the accuracy for the evaluation of \( f \) might cure this problem, but this approach is outside the scope of the paper.

We remark that in order to have (4.24) under Condition of Stage II, it is necessary to have \( \nabla f(\bar{x}^{(k)})^T d^{(k)} \geq 2L t^2 \| \nabla f(\bar{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} \|^2 \). Since \( d^{(k)} \) is stochastic, in the next section, we study the expectation of the updating direction for different rounding methods, i.e., CSR and \( \text{SR}_{\epsilon} \). Then, we consider signed-\( \text{SR}_{\epsilon} \) in evaluating (4.23), and we show that it always provides a rounding bias in a descent direction.

**4.2.1. Analysis of the updating direction dominated by rounding errors.** Before analyzing the updating rule (4.23), we introduce the functions that return the successor and the predecessor of a given floating point number \( \tilde{x} \in \mathbb{F} \):

\[
(4.26) \quad \text{su}(\tilde{x}) = \min \{ \tilde{y} > \tilde{x} \mid \tilde{y} \in \mathbb{F} \} \quad \text{and} \quad \text{pr}(\tilde{x}) = \max \{ \tilde{y} < \tilde{x} \mid \tilde{y} \in \mathbb{F} \}.
\]

Note that \( \text{su}(\cdot) \) and \( \text{pr}(\cdot) \) differ from the ceiling and floor operations in view of the strict inequality in (4.26).

Depending on the sign of the components of \( \tilde{x}^{(k)} \) and \( \nabla f(\tilde{x}^{(k)}) \), the entries of \( d^{(k)} \) can be written as follows,

\[
(4.27a) \quad d_i^{(k)} = \begin{cases} 
\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)}), & \text{with probability } p(z_i^{(k+1)}), \\
0, & \text{with probability } 1 - p(z_i^{(k+1)}),
\end{cases}
\]

or

\[
(4.27b) \quad d_i^{(k)} = \begin{cases} 
0, & \text{with probability } p(z_i^{(k+1)}), \\
\tilde{x}_i^{(k)} - \text{su}(\tilde{x}_i^{(k)}), & \text{with probability } 1 - p(z_i^{(k+1)}),
\end{cases}
\]

where \( p \in \{ p_c, p_\varepsilon, \tilde{p}_\varepsilon \} \) identifies the stochastic rounding scheme employed. Table 4 shows the four cases that tune the rounding errors in a descent direction for the two updating rules in (4.27). Note that we always have \( \text{sign}(\tilde{x}^{(k)}) = \text{sign}(\tilde{x}^{(k+1)}) \); this is natural in this scenario as \( t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} \) is relatively small with respect to \( \tilde{x}^{(k)} \).

**Table 4**

The \( i \)th entry of the updating step vector at the \( k \)th iteration \( d_i^{(k)} \), under different conditions and its corresponding rounding method that tunes it in a descent direction.

| Sign \( \text{sign}(\tilde{x}_i^{(k)}) \text{sign}(\nabla f(\tilde{x}^{(k)})) \) | Rounding method | Case |
| --- | --- | --- |
| \( \geq 0 \) | round down | I |
| \( < 0 \) | round up | II |
| \( \geq 0 \) | round up | III |
| \( < 0 \) | round down | IV |

Now let us study (4.23) with stochastic rounding. When CSR is employed in (4.23), the unbiased property implies that \( E [d_i^{(k)}] = t \nabla f(\tilde{x}^{(k)}) h_{1,i}^{(k)} h_{2,i}^{(k)} = t \nabla f(\tilde{x}^{(k)}) h_{1,i}^{(k)} h_{2,i}^{(k)} \). Based on this, we obtain the following expectation for the quantity \( \nabla f(\tilde{x}^{(k)})^T d^{(k)} \).
Lemma 4.8. When CSR is applied for computing (4.23), we have

\[(4.28) \quad E \left[ \nabla f(\tilde{x}^{(k)})^T d^{(k)} \right] = E \left[ t \nabla f(\tilde{x}^{(k)})^T (\nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}) \right]. \]

Proof. We are going to use an analogous argument to the one employed in the proof of Theorem 4.5. We denote by \( \mathcal{S} \) the finite set of values for the \( i \)th component of \( t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} \). By means of the law of total expectation, we write

\[(4.29) \quad E \left[ \nabla f(\tilde{x}^{(k)})^T d^{(k)} \right] = \sum_{i=1}^{n} \sum_{\nabla f(\tilde{x}^{(k)})_i h_1^{(k)} h_2^{(k)} \in \mathcal{S}} E \left[ d_i^{(k)} \nabla f(\tilde{x}^{(k)})_i | \mathcal{S} \right] P(\mathcal{S})
\[
= \sum_{i=1}^{n} \sum_{\nabla f(\tilde{x}^{(k)})_i h_1^{(k)} h_2^{(k)} \in \mathcal{S}} t \nabla f(\tilde{x}^{(k)})_i^2 h_1^{(k)} h_2^{(k)} P(\mathcal{S})
\[
= \sum_{i=1}^{n} E \left[ t \nabla f(\tilde{x}^{(k)})_i^2 h_1^{(k)} h_2^{(k)} \right] P(\mathcal{S})
\]

We now shed some light on why the use of SR\( \varepsilon \) in this context might be problematic and why it is a good idea to consider signed-SR\( \varepsilon \). Let us assume that SR\( \varepsilon \) is employed for evaluating the updating rule for Case I in Table 4; then, by means of the relation \( p_\varepsilon(x) = \varphi(p_\varepsilon(x) - \text{sign}(x)\varepsilon) \), we have

\[
E \left[ d_i^{(k)} | (t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)})_i \right] = (\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)})) p_\varepsilon(z_i^{(k+1)})
\[
= \begin{cases} 
\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)}), & \text{if } p_\varepsilon = 1 \\
(\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)})) (p_\varepsilon(z_i^{(k+1)}) - \text{sign}(z_i^{(k+1)}) \varepsilon), & \text{otherwise}
\end{cases}
\]

\[(4.30) \quad = \begin{cases} 
\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)}), & \text{if } p_\varepsilon = 1 \\
t \nabla f(\tilde{x}^{(k)})_i h_1^{(k)} h_2^{(k)} - (\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)})) \text{sign}(\tilde{x}_i^{(k)}) \varepsilon, & \text{otherwise}
\end{cases}
\]

From (4.30), it can be seen that when \( p_\varepsilon \neq 1 \), it is hard to control the updating direction of GD, unless \( \tilde{x}_i^{(k)} \) has always the opposite sign as \( \nabla f(\tilde{x}^{(k)})_i \). Clearly, this cannot be guaranteed by SR\( \varepsilon \) but can be easily achieved by signed-SR\( \varepsilon \) by substituting \( v \) with \( \nabla f(\tilde{x}^{(k)})_i \) for the corresponding input \( x_i^{(k)} \) in (2.4). With this choice, signed-SR\( \varepsilon \) may achieve a higher average convergence rate than CSR.

Lemma 4.9. When signed-SR\( \varepsilon \) is used in (4.23), we have

\[(4.31) \quad E \left[ \nabla f(\tilde{x}^{(k)})^T d^{(k)} \right] > E \left[ t \nabla f(\tilde{x}^{(k)})^T (\nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}) \right]. \]
Proof. First we consider Case I; in particular we have $\nabla f(\tilde{x}^{(k)})_i > 0$. Taking the conditional expectation of (4.27a), analogously to (4.30), we obtain

$$E[d_i^{(k)} | (t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)})_i]$$

$$= (\tilde{x}_i^{(k)} - pr(\tilde{x}_i^{(k)})) \varphi(p_c(z_i^{(k+1)}) + \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon)$$

$$= \begin{cases} (\tilde{x}_i^{(k)} - pr(\tilde{x}_i^{(k)})) \varphi(p_c(z_i^{(k+1)}) + \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon), & \tilde{p}_c = 1. \\ (t \nabla f(\tilde{x}^{(k)})_i h_{1,i}^{(k)} h_{2,i}^{(k)} + (\tilde{x}_i^{(k)} - pr(\tilde{x}_i^{(k)})) \varphi(p_c(z_i^{(k+1)}) + \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon), & \text{otherwise.} \end{cases}$$

(4.32)

In Case II, we have $\nabla f(\tilde{x}^{(k)})_i < 0$. Applying the similar steps as for Case I, we obtain

$$E[d_i^{(k)} | (t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)})_i]$$

$$= (\tilde{x}_i^{(k)} - \text{su}(\tilde{x}_i^{(k)}))(1 - \varphi(p_c(z_i^{(k+1)}) + \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon))$$

$$= \begin{cases} (\tilde{x}_i^{(k)} - \text{su}(\tilde{x}_i^{(k)})) \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon), & \tilde{p}_c = 0. \\ (t \nabla f(\tilde{x}^{(k)})_i h_{1,i}^{(k)} h_{2,i}^{(k)} + |\tilde{x}_i^{(k)} - \text{su}(\tilde{x}_i^{(k)})| \varphi(p_c(z_i^{(k+1)}) + \text{sign}(\nabla f(\tilde{x}^{(k)})_i) \varepsilon), & \text{otherwise.} \end{cases}$$

(4.33)

Following analogous arguments to those applied for Case I and Case II, we obtain that (4.32) and (4.33) also hold for Cases IV and III, respectively.

We denote by $S_j$ the finite set of values that can be assumed by the $i$th component of $t \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)}$ and satisfy Case I in Table 4. Analogously we define $S_j$, $j = 2, 3, 4$, for Cases II, III, and IV. Moreover, we rewrite the (conditional) expectation of $d_i^{(k)}$ as

$$E[d_i^{(k)} | S_j] = (1 + q_{i,j}) t \nabla f(\tilde{x}^{(k)})_i h_{1,i}^{(k)} h_{2,i}^{(k)},$$

where

$$q_{i,j} = \frac{\varepsilon |\tilde{x}_i^{(k)} - \text{su}(\tilde{x}_i^{(k)})| \text{sign}(\nabla f(\tilde{x}^{(k)})_i)}{t \nabla f(\tilde{x}^{(k)})_i h_{1,i}^{(k)} h_{2,i}^{(k)}} \text{ or } \frac{\varepsilon |\tilde{x}_i^{(k)} - \text{pr}(\tilde{x}_i^{(k)})| \text{sign}(\nabla f(\tilde{x}^{(k)})_i)}{t \nabla f(\tilde{x}^{(k)})_i h_{1,i}^{(k)} h_{2,i}^{(k)}}.$$
Finally, we recall that, based on the proof of Theorem 4.5 ((4.15) and (4.16)), we have the following property for (3.2a).

**Remark 4.10.** The employment of SR in (3.2a) leads to $E \left[ \nabla f(\tilde{x}^{(k)})^T \nabla f(\tilde{x}^{(k)}) \circ h_1 \circ h_2 \right] > E \left[ \|\nabla f(\tilde{x}^{(k)})\|^2 \right]$, while equality holds for CSR.

In the remainder of this section, we are going to use this property together with Lemma 4.8 and Lemma 4.9 for studying the influence of rounding bias of CSR and signed-SR on the convergence of GD.

## 4.2.2. Employment of CSR

We start by proposing an upper bound for $u$ that guarantees the average monotonicity of GD when CSR is employed for both (3.2a) and (4.23).

**Proposition 4.11.** Under Assumption 4.1 and Condition of Stage II, suppose that both (3.2a) and (4.23) are computed using CSR. If $u$ satisfies

$$
(4.35) \quad u \leq \sqrt{\frac{t}{2L} \frac{E \left[ \|\nabla f(\tilde{x}^{(k-1)})\|^2 \right]}{E \left[ \|\nabla f(\tilde{x}^{(k-1)})\|^2 \right]}},
$$

then $E \left[ f(\tilde{x}^{(k)}) \right] \leq E \left[ f(\tilde{x}^{(k-1)}) \right]$, for all $k > 0$.

**Proof.** Taking the expectation of (4.25) and using Lemma 4.8, we have

$$
E \left[ f(\tilde{x}^{(k+1)}) \right] \leq E \left[ f(\tilde{x}^{(k)}) \right] - E \left[ \nabla f(\tilde{x}^{(k)})^T d^{(k)} \right] + 2u^2L E \left[ \|\tilde{x}^{(k)}\|^2 \right] + tE \left[ \nabla f(\tilde{x}^{(k)})^T \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} \right] + 2u^2L E \left[ \|\tilde{x}^{(k)}\|^2 \right].
$$

Following similar steps as in (4.15) and (4.16), we obtain

$$
E \left[ f(\tilde{x}^{(k+1)}) \right] \leq E \left[ f(\tilde{x}^{(k)}) \right] - tE \left[ \|\nabla f(\tilde{x}^{(k)})\|^2 \right] + 2u^2L E \left[ \|\tilde{x}^{(k)}\|^2 \right].
$$

Then, (4.35) implies $E \left[ f(\tilde{x}^{(k+1)}) \right] \leq E \left[ f(\tilde{x}^{(k)}) \right]$. □

The proof of Proposition 4.11 can be easily adapted, by means of the same argument used in the proof of Theorem 4.5, for the following mixed rounding strategy.

**Proposition 4.12.** Under Assumption 4.1 and Condition of Stage II, suppose that (3.2a) and (4.23) are computed using SR and CSR, respectively. If $u$ satisfies (4.35), then we have $E \left[ f(\tilde{x}^{(k)}) \right] < E \left[ f(\tilde{x}^{(k-1)}) \right]$, for all $k > 0$.

## 4.2.3. Employment of signed-SR

Under the same conditions as the analysis with CSR, we show that signed-SR guarantees the strict monotonicity of the GD iteration.

**Proposition 4.13.** Under Assumption 4.1 and Condition of Stage II, suppose that (3.2a) is computed by either CSR or SR and that (4.23) is computed using signed-SR. If $u$ satisfies (4.35), then $E \left[ f(\tilde{x}^{(k)}) \right] < E \left[ f(\tilde{x}^{(k-1)}) \right]$, for all $k > 0$.

**Proof.** Taking the expectation of (4.25) and using (4.34), we have

$$
E \left[ f(\tilde{x}^{(k+1)}) \right] \leq E \left[ f(\tilde{x}^{(k)}) \right] - E \left[ \nabla f(\tilde{x}^{(k)})^T d^{(k)} \right] + 2Lu^2 E \left[ \|\tilde{x}^{(k)}\|^2 \right]
$$

$$
\leq E \left[ f(\tilde{x}^{(k)}) \right] - tE \left[ \nabla f(\tilde{x}^{(k)})^T \nabla f(\tilde{x}^{(k)}) \circ h_1^{(k)} \circ h_2^{(k)} \right] + 2Lu^2 E \left[ \|\tilde{x}^{(k)}\|^2 \right]
$$

$$
+ 2Lu^2 E \left[ \|\tilde{x}^{(k)}\|^2 \right] - \sum_{i=1}^{n} Q_i^{(k)}.
$$

(4.36)
Based on Remark 4.10, we have
\[ E \left[ \nabla f(\tilde{x}(k)^T \nabla f(\tilde{x}(k)) \circ h_1^{(k)} \circ h_2^{(k)} \right] \geq E \left[ \| \nabla f(\tilde{x}(k)) \|^2 \right] \]
when CSR or SR$_\varepsilon$ is used for evaluating (3.2a). Therefore, in view of (4.35) and (4.36), we obtain
\[ E \left[ f(\tilde{x}(k+1)) \right] \leq E \left[ f(\tilde{x}(k)) \right] - \sum_{i=1}^{n} Q_i^{(k)}, \tag{4.37} \]
where $Q_i^{(k)} > 0$. This gives the claim.

Despite the fact that Proposition 4.13 does not provide a significant advantage of signed-SR$_\varepsilon$ with respect to CSR, inequality (4.37) shows that signed-SR$_\varepsilon$ may lead to a faster convergence depending on the accumulated rounding bias that is determined by the value of $\varepsilon$. In the next section, we demonstrate, by means of numerical simulations, that this advantage is indeed tangible.

5. Simulation study. In this section, we validate the theoretical analysis by testing the performances of GD with various choices for the rounding schemes used in the evaluation of steps (3.2a) and (3.2b). As case studies, we consider minimizing quadratic functions, the training of a multinomial logistic regression model (MLR), and the training of a two-layer NN, with low-precision floating-point computations. All three stochastic rounding schemes, CSR, SR$_\varepsilon$ and signed-SR$_\varepsilon$, are implemented by slightly modifying the MATLAB function chop [15]. As representatives of low-precision number formats, we consider binary16 for the quadratic optimization and binary8 for the training of MLR and NN, that employ 16 and 8 bits with customized significand field and exponent, respectively. See also Table 2 for the complete descriptions of the number formats. The baselines are obtained by single-precision floating-point computation (binary32) with the default rounding mode in IEEE, i.e., RN with ties to even. Note that the machine precision of single precision is very small ($2^{-24}$) compared to the limited precision employed by binary16 ($2^{-7}$) and binary8 ($2^{-3}$), i.e., the roundoff errors caused by single precision are almost negligible compared to those of binary16 and binary8. To some extent, we look at the comparison with the baseline as a comparison with GD in exact arithmetic. Further, all the expectations and variances obtained when using CSR, SR$_\varepsilon$, and signed-SR$_\varepsilon$ are estimated over 20 simulations.\(^1\)

Note that all the plots in this section employ a logarithmic scale along the vertical axis.

5.1. Quadratic optimization. In this first experiment we apply the GD method to the quadratic optimization problem
\[ \min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} x^T A x, \tag{5.1} \]
for two choices of the matrix $A$, the starting vector $x_0$, and the stepsize $t$. Our first choice (Setting I) is $A = \text{diag}(10^{-3}, \ldots, 10^{-3}, 1) \in \mathbb{R}^{1000 \times 1000}$, $x^{(0)} = [10^{-3}, \ldots, 10^{-3}, 1]^T$, and $t = 10^{-5}$. In particular, the stepsize is relatively small compared to $L^{-1}$ and all the entries of the initial point are close to the minimizer apart from the last entry. In the second choice (Setting II), we take $A = \text{diag}(1, 2, \ldots, 1000) \in \mathbb{R}^{1000}$, $x^{(0)} = [1000, 999, \ldots, 1]^T$, and $t = L^{-1} = 10^{-5}$. We remark that in Setting II, we select the largest possible stepsize among those that guarantee convergence, and a starting point that is quite far from the minimizer.

\(^1\)The MATLAB code is available on request to the corresponding author.
Figure 3. Comparison of the bound from Theorem 4.2 and the expectation of (5.1) while using CSR to implement (3.2a) and different rounding schemes to implement (3.2b), i.e., CSR and signed-SR$_{\varepsilon}$ with $\varepsilon = 0.4$ for Setting I (a) and Setting II (b).

Figure 3 reports the convergence history of the implementations of GD with various choices of number formats and rounding schemes, together with the bound $\frac{2L}{1+L^T L} \| x^{(0)} - x^* \|^2$ from Theorem 4.2. More precisely, we compare the objective function values obtained by GD in single-precision computation and RN with the average of the objective function values obtained using binary16 with CSR for (3.2a) and different stochastic rounding schemes for (3.2b); the results concerning Setting I are shown in Figure 3a while those regarding Setting II are reported in Figure 3b. We did not include the convergence history of GD with RN, in the binary16 format, as it stagnates from the first iteration. From Figure 3a, it can be seen that the bound in Theorem 4.2 is very close to the objective function obtained by single-precision computation and the one by binary16 with CSR. However, using signed-SR$_{\varepsilon}$ to implement (3.2b), we achieve almost linear convergence for GD, which is consistent with the discussion after Proposition 4.13. From Figure 3b, it can be seen that, in Setting II, the bound in Theorem 4.2 is not strict anymore although the objective function obtained with binary32 has a similar convergence rate (slope). Again, the employment of CSR with binary16 leads to a similar expectation of the objective function values. The use of signed-SR$_{\varepsilon}$ yields a much faster convergence rate than both single-precision computation and CSR with binary16. We can conclude that, for both settings, the employment of signed-SR$_{\varepsilon}$ for (3.2b) accelerates the convergence rate significantly with respect to both RN and CSR.

5.2. Multinomial logistic regression (MLR). MLR is an optimization problem that models multi-label classification tasks. The objective function of MLR is convex [3]; for a detailed description; see e.g., [17, pp. 269–272]. We consider the solution of MLR for classifying the MNIST database [9], that is a large database of 10 handwritten digits (from 0 to 9), containing 60000 training images and 10000 test images.

In our first experiment, we employ CSR to evaluate (3.2b), and we test different stochastic rounding methods for (3.2a). Figure 4a shows the expectation of testing errors of the MLR model when classifying 0 to 9 with the 10000 test images. After 10 epochs, binary8 with RN
stagnates due to the loss of gradient information. With the same number of training epochs, the testing errors of the regression model obtained by CSR are very similar to the baseline, as predicted by Corollary 4.6. The regression models computed using SR_ε have lower testing errors than those obtained by CSR. With 150 training epochs, the testing error of the baseline is 0.096; a similar accuracy is obtained by SR_ε with ε = 0.2, 0.4 (a) and different combinations of rounding schemes to implement (3.2). Further, a faster convergence is achieved with larger ε when using SR_ε, which is consistent with the conclusions after Theorem 4.5.

In the second experiment, we apply different rounding schemes to both (3.2a) and (3.2b). We use CSR and SR_ε to implement (3.2a); for (3.2b) we use CSR and signed-SR_ε with the same settings of Lemma 4.9. Figure 4b shows the comparison of the expectation of testing errors of the MLR model when implementing GD with different combinations of rounding schemes. We can see that the convergence is significantly faster when using signed-SR_ε for (4.23). Further increasing the parameter ε used in signed-SR_ε, leads GD to “jump over” the optimum, which can be seen as employing a very large learning stepsize with exact computations. We also measure the population variance [32] over 20 simulations for all the experiments in Figure 4; after 50 training epochs, all the population variances are less than 10^{-5}. This indicates small deviations from the average cases.

To further investigate the performances of the various rounding schemes, we analyze the effect of varying the parameter t. In Figure 5a we report the expectation of testing errors of the MLR model with different learning rate t while using CSR to implement both steps of (3.2). It can be seen that the convergence rate increases with the learning rate t although it never beats the baseline that consists of single-precision computations and t = 1.25. We remark that further increasing t, with single-precision, leads to large oscillations.

The experiment is repeated by employing SR_ε with ε = 0.1 for (3.2a) and signed-SR_ε with ε = 0.1 for (3.2b). We remark that the employment of SR_ε in evaluating (3.2a) yields similar performances as signed-SR_ε. The results reported in Figure 5b show that the convergence
obtained with $t = 0.5$ is already faster than the baseline. Increasing $t$ until 1 leads to higher convergence rates. However, when $t = 1.25$, the testing error starts to increase after 125 training epochs, which indicates that $t = 1.25$ is too large for this rounding strategy. With 150 training epochs, the baseline obtains a testing error of 0.086, while a similar value is obtained by signed-SR$_\varepsilon$ with $t = 1$ after only 84 training epochs (see Figure 5b).

5.3. A two-layer NN for binary classification. Although the training of a two-layer NN is not a convex problem, GD with SR$_\varepsilon$ still shows a convergence behavior similar to the one described when dealing with an MLR model. The training is performed on the set of data comprised of digits 3 and 8, resulting in 11982 training images and 1984 testing images. As in [12], the pixel values are normalized to [0, 1]. A two-layer NN is built with the ReLU activation function in the hidden layer and the sigmoid activation function in the output layer. The hidden layer contains 100 units. In the backward propagation, a binary cross-entropy loss function is optimized using GD. The weights matrix is initialized based on Xavier initialization [11] and the bias is initialized as a zero vector. Additionally, the default decision threshold is set for interpreting probabilities to class labels, that is 0.5, since the sample class sizes are almost equal [4]. Specifically, class 1 is defined for those predicted scores larger than or equal to 0.5.

Figure 6a shows the comparison of the expectation of testing errors of the two-layer NN trained using binary8 with RN for (3.2) and using CSR for (3.2b) and different stochastic rounding methods for (3.2a). Again, the NN trained using RN fails to converge due to the loss of gradient information. CSR leads to similar testing errors as the baseline, while SR$_\varepsilon$ results in a slightly higher convergence rate than CSR. Based on Definition 2.1, a larger $\varepsilon$ leads to a larger rounding bias, which also leads to slightly faster convergence in Figure 6a.

To study the influence of rounding bias in each step of (3.2), we employ signed-SR$_\varepsilon$ with the same settings of Lemma 4.9 for evaluating (3.2b). Figure 6b shows the expectation of testing
errors when implementing GD with different combinations of rounding schemes. Again, the use of signed-SR$_e$ for (3.2b), yields lower testing errors with less training epochs. For instance, the testing error after 50 training epochs with single-precision computations is 0.042, while a similar testing error is obtained after only 25 training epochs when using the combination of SR$_e$ and signed-SR$_e$ (see Figure 6b). Also here, a large rounding bias in evaluating the second step (4.23) leads GD to “jump over” the optimum (see the results of the $\varepsilon = 0.2$ case in Figure 6b).

As observed in these numerical experiments, the magnitude of the parameter $\varepsilon$ has a crucial role when implementing SR$_e$ or signed-SR$_e$, as it controls the rounding bias in the descent direction. In particular, $\varepsilon > 0$ may accelerate the convergence of GD, but a too large value may also make GD “jump over” the optimum. In general, the choice of $\varepsilon$ needs to take into account the machine precision $u$. In the case of binary8, we suggest to choose an $\varepsilon \in (0, 0.1]$.

6. Conclusion. In this paper, we have studied the influence of rounding bias on the convergence of the gradient descent method (GD) with low-precision floating-point computation for convex problems. Based on the source of rounding errors, we have distributed the rounding process of GD in two steps. We have proposed two new stochastic rounding methods, called SR$_e$ and signed-SR$_e$, to be applied in the two steps. Further, we have analyzed the role of rounding on the convergence of GD in two stages; the stage where the rounding to the nearest method (RN) has no stagnation and where RN stagnates. We have showed that the employment of the proposed rounding methods can eliminate the vanishing gradient problems. More precisely, we have proved, both theoretically and practically, that faster convergence rates may be achieved in both stages, by the use of SR$_e$ and signed-SR$_e$. The proposed rounding methods are especially beneficial for machine learning, e.g., training neural networks and regression models, where low-precision computations and the GD are widely employed.
REFERENCES

[1] B. Afsari, R. Tron, and R. Vidal, On the convergence of gradient descent for finding the Riemannian center of mass, SIAM J. Control Optim., 51 (2013), pp. 2230–2260, https://doi.org/10.1137/12086282X.

[2] R. Bachmayer and N. E. Leonard, Vehicle networks for gradient descent in a sampled environment, in Proc. of the 41st IEEE Conf. Decis. Control, vol. 1, IEEE, 2002, pp. 112–117, https://doi.org/10.1109/CDC.2002.1184477.

[3] D. Böhning, Multinomial logistic regression algorithm, Ann. Inst. Stat. Math., 44 (1992), pp. 197–200, https://doi.org/10.1007/BF00048682.

[4] J. Chen, C.-A. Tsai, H. Moon, H. Ahn, J. Young, and C.-H. Chen, Decision threshold adjustment in class prediction, SAR QSAR Environ. Res., 17 (2006), pp. 337–352, https://doi.org/10.1080/10659360600787700.

[5] E. Chung, J. Fowers, K. Ovtcharov, M. Papamichael, A. Caulfield, T. Massengill, M. Liu, D. Lo, S. Alkalay, M. Haselman, et al., Serving DNNs in real time at datacenter scale with project brainwave, IEEE Micro, 38 (2018), pp. 8–20, https://doi.org/10.1109/MM.2018.022071131.

[6] M. P. Connolly, N. J. Higham, and T. Mary, Stochastic rounding and its probabilistic backward error analysis, SIAM J. Sci. Comput., 43 (2021), pp. A566–A585, https://doi.org/10.1137/20M1334796.

[7] M. Croci and M. B. Giles, Effects of round-to-nearest and stochastic rounding in the numerical solution of the heat equation in low precision, 2020, https://arxiv.org/abs/2010.16225.

[8] M. Davies, N. Srinivasa, T. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C. Lin, A. Lines, R. Liu, D. Mathaiakudity, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y. Weng, A. Wild, Y. Yang, and H. Wang, Loihi: A neuromorphic manycore processor with on-chip learning, IEEE Micro, 38 (2018), pp. 82–99, https://doi.org/10.1109/MM.2018.112130359.

[9] L. Deng, The MNIST database of handwritten digit images for machine learning research, IEEE Signal Process. Mag., 29 (2012), pp. 141–142, https://doi.org/10.1109/MSP.2012.2211477.

[10] M. Egerstedt, Y. Wardi, and F. Delmotte, Optimal control of switching times in switched dynamical systems, in Proc. of the 42nd IEEE Conf. Decis. Control, vol. 3, IEEE, 2003, pp. 2138–2143, https://doi.org/10.1109/CDC.2003.1272934.

[11] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in Proc. of the 13th Int. Conf. Artif. Intell. Stat. (AISTATS), 2010, pp. 249–256, http://proceedings.ml.org/papers/v9/glorot10a.html.

[12] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, Deep learning with limited numerical precision, in Proc. of the 32nd Int. Conf. Mach. Learn. (ICML), 2015, pp. 1737–1746, http://www.jmlr.org/proceedings/papers/v37/gupta15a.html.

[13] B. Hickmann, J. Chen, M. Rotzin, A. Yang, M. Urbanski, and S. Avancha, Intel Nervana neural network processor-t (NNP-T) fused floating point many-term dot product, in Proc. of the 27th IEEE Symp. Comput. (ARITH), IEEE, 2020, pp. 133–136, https://doi.org/10.1109/ARITH48897.2020.00029.

[14] N. J. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, 2002, https://doi.org/10.1137/1.9780898781807.

[15] N. J. Higham and S. Pranesh, Simulating low precision floating-point arithmetic, SIAM J. Sci. Comput., 41 (2019), pp. C585–C602, https://doi.org/10.1137/19M1251308.

[16] M. Höhfeld and S. E. Fahlman, Probabilistic rounding in neural network learning with limited precision, Neurocomputing, 4 (1992), pp. 291–299, https://doi.org/10.1016/0925-2312(92)90014-G.

[17] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, Applied Logistic Regression, vol. 398, John Wiley & Sons, 2013, https://doi.org/10.1002/9781118548387.

[18] IEEE, IEEE standard for floating-point arithmetic, IEEE Std 754-2019 (Revision of IEEE 754-2008), (2019), pp. 1–84, https://doi.org/10.1109/IEEESTD.2019.8766229.

[19] N. P. Jouppi, D. H. Yoon, G. Kurian, S. Li, N. Patil, J. Laudon, C. Young, and D. Patterson, A domain-specific supercomputer for training deep neural networks, Commun. ACM, 63 (2020), pp. 67–78, https://doi.org/10.1145/3360307.

[20] M. Kuczma, An Introduction to the Theory of Functional Equations and Inequalities: Cauchy’s Equa-
tion and Jensen’s Inequality, Springer Science & Business Media, 2009, https://doi.org/10.1007/978-3-7643-8749-5.

[21] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht, Gradient descent only converges to minimizers, in Proc. of the 29th Annual Conference on Learning Theory (COLT), PMLR, 2016, pp. 1246–1257, https://proceedings.mlr.press/v49/lee16.html.

[22] Y.-J. Liu, Y. Gao, S. Tong, and Y. Li, Fuzzy approximation-based adaptive backstepping optimal control for a class of nonlinear discrete-time systems with dead-zone, IEEE Trans. Fuzzy Syst., 24 (2015), pp. 16–28, https://doi.org/10.1109/TFUZZ.2015.2418000.

[23] M. Mikaitis, Stochastic rounding: Algorithms and hardware accelerator, in Proc. of 2021 Int. Jt. Conf. Neural Netw., IEEE, 2021, pp. 1–6, https://doi.org/10.1109/IJCNN52387.2021.9533756.

[24] E. Moulay, V. Léchappé, and F. Plestan, Properties of the sign gradient descent algorithms, Inf. Sci., 492 (2019), pp. 29–39, https://doi.org/10.1016/j.ins.2019.04.012.

[25] T. Na, J. H. Ko, J. Kung, and S. Mukhopadhyay, On-chip training of recurrent neural networks with limited numerical precision, in Proc. of the 2017 Int. Jt. Conf. Neural Netw., IEEE, 2017, pp. 3716–3723, https://doi.org/10.1109/IJCNN.2017.7966324.

[26] Y. Nesterov, Introductory Lectures on Convex Optimization: A Basic Course, vol. 87, Springer, 2003, https://doi.org/10.1007/978-1-4419-8853-9.

[27] Y. Nesterov, Gradient methods for minimizing composite functions, Math. Program., 140 (2013), pp. 125–161, https://doi.org/10.1007/s10107-012-0629-5.

[28] Nvidia, V100 GPU architecture, Online available, https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf.

[29] H. Singh, L. Upadhyaya, and U. Namjoshi, Estimation of finite population variance, Curr. Sci, (1988), pp. 1331–1334, https://www.jstor.org/stable/2409446.

[30] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane, Path planning for autonomous underwater vehicles, IEEE Trans. Robot., 23 (2007), pp. 331–341, https://doi.org/10.1109/TRO.2007.895057.

[31] H. Singh, L. Upadhyaya, and U. Namjoshi, Estimation of finite population variance, Curr. Sci, (1988), pp. 1331–1334, https://www.jstor.org/stable/24094466.

[32] D. E. Soltero, M. Schwager, and D. Rus, Decentralized path planning for coverage tasks using gradient descent adaptive control, Int. J. Rob. Res., 33 (2014), pp. 401–425, https://doi.org/10.1177/0278364913497241.

[33] C. Su, S. Zhou, L. Feng, and W. Zhang, Towards high performance low bitwidth training for deep neural networks, J. Semicond., 41 (2020), p. 022404, https://doi.org/10.1088/1674-4926/41/2/022404.

[34] C. Sun and J.-P. Liu, New stepsizes for the gradient method, Optim. Lett., 14 (2020), pp. 1943–1955, https://doi.org/10.1007/s11590-019-01512-y.

[35] N. Wang, J. Choi, D. Brand, C.-Y. Chen, and K. Gopalakrishnan, Training deep neural networks with 8-bit floating point numbers, in Proc. of the 31st Neural Inf. Process. Syst. Conf. (NIPS), 2018, pp. 7675–7684. https://proceedings.neurips.cc/paper/2018/hash/335d31cd7e05ec77714a215134914c-Abstract.html.

[36] D. Zou, Y. Cao, D. Zhou, and Q. Gu, Gradient descent optimizes over-parameterized deep ReLU networks, Mach. Learn., 109 (2020), pp. 467–492, https://doi.org/10.1007/s10994-019-05839-6.