SeRaNDiP - Leveraging Inherent Sensor Random Noise for Differential Privacy Preservation in Wearable Community Sensing Applications

AYANGA IMESHA KUMARI KALUPAHANA, National University of Singapore, Singapore
ANANTA NARAYANAN BALAJI, National University of Singapore, Singapore
XIAOKUI XIAO, National University of Singapore, Singapore
LI-SHIUAN PEH, National University of Singapore, Singapore

Personal data collected from today's wearable sensors contain a rich amount of information that can reveal a user's identity. Differential privacy (DP) is a well-known technique for protecting the privacy of the sensor data being sent to community sensing applications while preserving its statistical properties. However, differential privacy algorithms are computationally expensive, requiring user-level random noise generation which incurs high overheads on wearables with constrained hardware resources. In this paper, we propose *SeRaNDiP* - which utilizes the inherent random noise existing in wearable sensors for distributed differential privacy. We show how various hardware configuration parameters available in wearable sensors can enable different amounts of inherent sensor noise and ensure distributed differential privacy guarantee for various community sensing applications with varying sizes of populations. Our evaluations of SeRaNDiP on five wearable sensors that are widely used in today's commercial wearables - MPU-9250 accelerometer, ADXL345 accelerometer, BMP 388 barometer, MLP 3115A2 barometer, and MLX90632 body temperature sensor show a 1.4X-1.8X computation/communication speedup and 1.2X-1.5X energy savings against state-of-the-art DP implementation. To the best of our knowledge, SeRaNDiP is the first framework to leverage the inherent random sensor noise for differential privacy preservation in community sensing without any hardware modification.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; • **Security and privacy** → *Human and societal aspects of security and privacy* ; • **Human-centered computing** → Ubiquitous and mobile computing.

Additional Key Words and Phrases: Sensors, Differential Privacy, Inherent Sensor Noise, Low Power

ACM Reference Format:
Ayanga Imesha Kumari Kalupahana, Ananta Narayanan Balaji, Xiaokui Xiao, and Li-Shiuan Peh. 2023. SeRaNDiP - Leveraging Inherent Sensor Random Noise for Differential Privacy Preservation in Wearable Community Sensing Applications. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 2, Article 61 (June 2023), 38 pages. https://doi.org/10.1145/3596252

1 INTRODUCTION

Today's smartphones and wearables contain an impressive array of sensors such as accelerometer, microphone, GPS, and barometer providing information about the user's activities and surroundings. Collecting people's surrounding information and activities via sensors in smartphones and wearables is typically referred to as *participatory sensing or crowd sensing*. Users' health status (heart rate, SpO2, blood pressure, etc.) and activities (daily step counts, calories etc.) can be measured by wearable devices. This has led to governments around the world promoting the health and well-being of their citizens by introducing incentive-based community-sensing programs through monitoring user activities. 10,000 Steps Australia, National Step Challenge Singapore are

Authors' addresses: Ayanga Imesha Kumari Kalupahana, National University of Singapore, Singapore, ayangaim@comp.nus.edu.sg; Ananta Narayanan Balaji, National University of Singapore, Singapore, ananta@u.nus.edu; Xiaokui Xiao, National University of Singapore, Singapore, xkxiao@nus.edu.sg; Li-Shiuan Peh, National University of Singapore, Singapore, peh@nus.edu.sg.

2474-9567/2023/June-ART61
https://doi.org/10.1145/3596252

some examples of popular community sensing programs where movement (accelerometer) and heart rate (PPG) data are collected from users to determine daily activity levels. Google Map's Live Traffic is another popular community sensing program conducted globally using GPS data from mobile phones with more than 154.4 million monthly participants [10]. Globally, there are 31 million and 100 million active Fitbit and Apple Watch users respectively [8, 12]. The privacy of the sensor data collected from such a vast community of users needs to be safeguarded.

Though many existing community sensing programs send features processed from raw sensor data (step count, activity, location, etc.) to the community sensing server, there are also numerous community sensing applications that directly share the raw sensor values [13, 37, 52, 79, 83]. Besides, just sharing processed features with the community sensing server severely limit the advanced analytics that can be performed on the server. With open-source AI analytics tools such as GGIR [64], AirSensor [35], exploreR [57] in widespread use for studying users' activity, health, etc., there have been increasing deployments of community sensing applications that share raw sensor data to the cloud [6]. For instance, crowd-sourced raw IMU data from smartphones have been used to construct an open hazard data map for preventing bicycle accidents [52]. This helps transport authorities efficiently monitor road surfaces by detecting road potholes and bumps [83]. Feverprint [79] is a community sensing program from Apple that collects body temperature values from iPhone users to identify the normal range of temperatures for a country-wide population at different times throughout the day. Weather researchers use crowdsourced raw air temperature observations [37] and raw atmospheric pressure measurements [13] from smartphone barometer sensors to create a crowd-sourced intercontinental network for weather forecasting. We believe that crowd-sensing programs will benefit significantly from receiving privacy-preserved raw sensor data from users to learn advanced insights regarding the behavior of the community.

However, preserving the privacy of wearable sensor data being sent to the community sensing server is a challenge [26, 28, 36, 54, 62, 86]. For instance, *accelerometers* can reveal a person's height [9] and emotions [5]; *barometer* data can help identify driving patterns [42], transportation modes [75]; *temperature* sensor data can reveal information related to female infertility [40] and depression [72]. Although the data being sent to the community sensing servers are anonymized, participants can still be de-identified if biometrics are monitored continuously for a considerable period of time: Foschini et al. showed that step counts of six days suffice to identify a user from 100 million users [38]. Recent reports question the privacy guarantee provided by community sensing organizations. In 2020, it was found that Samsung had been releasing Samsung Pay data to a third party for years without the user's knowledge [85]. Therefore, it is essential to implement Privacy Enhancing Technologies (PET) on wearable sensors at the user level.

Distributed Differential Privacy (DDP) is a data distortion method that perturbs the raw sensor data (being sent from the user's wearables to the community sensing servers) by the addition of statistical noise, so that attackers cannot infer information about any specific user record. In 2017, Apple implemented local differential privacy (a variant of distributed differential privacy) in Mac OS Sierra and iOS 10 [19] - to gain insights into usage patterns. DDP is also recently adopted to the Exposure Notification Privacy-preserving Analytics (ENPA) introduced by Apple and Google to enable automated alerts to users with potential exposure to COVID-19. Here, user metrics are randomized using distributed differential privacy at the smartphone before sending them to Public Health Authorities [20]. With wearables intimately worn on users throughout the day, they provide substantially more personal sensor information than smartphones, so DDP is even more critically needed. However, wearables such as smartwatches, fitness trackers, smart glasses have very limited compute and battery resources, due to cost and weight constraints. In this paper, we thus focus on lowering the power and computation overheads involved in realizing DDP on today's wearables with constrained hardware resources.

To characterize the overheads involved in supporting DDP on wearables, we implemented a state-of-the-art baseline DDP system [33] as shown in Figure 1. In our baseline DDP implementation, each user sends their raw sensor data into a differential privacy preservation module (running on the processor of the wearable device),

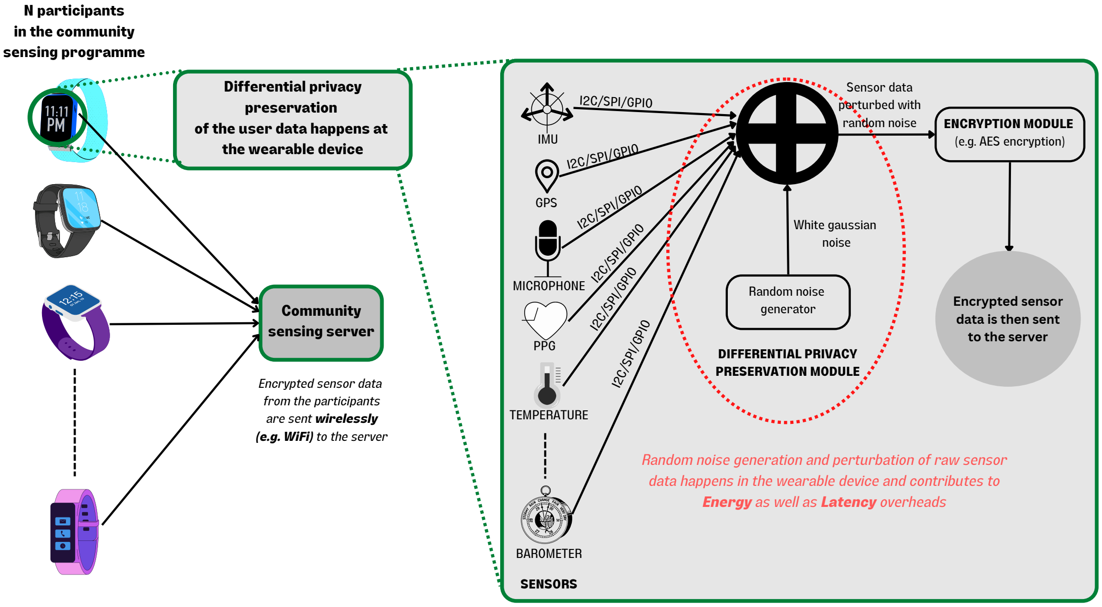

N participants in the community sensing programme
11:11 PM
12:15
Differential privacy preservation of the user data happens at the wearable device
Community sensing server
Encrypted sensor data from the participants are sent **wirelessly** **(e.g. WiFi)** to the server
IMU
GPS
MICROPHONE
PPG
TEMPERATURE
BAROMETER
SENSORS
I2C/SPI/GPIO
I2C/SPI/GPIO
I2C/SPI/GPIO
I2C/SPI/GPIO
I2C/SPI/GPIO
I2C/SPI/GPIO
Sensor data perturbed with random noise
White gaussian noise
Random noise generator
DIFFERENTIAL PRIVACY PRESERVATION MODULE
ENCRYPTION MODULE
(e.g. AES encryption)
Encrypted sensor data is then sent to the server
Random noise generation and perturbation of raw sensor data happens in the wearable device and contributes to **Energy** as well as **Latency** overheads

Fig. 1. Baseline DDP implementation for wearable community sensing applications

which generates random noise and adds it to the raw sensor data to protect the user's identity, and then the encrypted sensor data from each user's wearable device is sent to a server via WiFi. We implemented the baseline DDP architecture using the Tizen Studio environment for Samsung Galaxy smartwatch (87DB). Software based white gaussian noise generation in the Galaxy smartwatch introduces 1.3ms delay for each sampling of the accelerometer sensor, which constitutes about 33% of the total end-to-end delay illustrated in Figure 1, from the sampling of the sensor, through the noise generation and computation, transmission to the server and server computation. Further, noise generation and addition consumes 20% additional energy.

In wearables, data is continuously obtained from sensors and the acquired sensor data is inherently noisy owing to the electrical/mechanical properties of the sensor as well as the inherent variance in the biological data sensed from the person wearing the wearable (see Section 4.1 for more details). In this paper, we explore whether the inherent sensor noise observed can be used in place of the software random noise generation step, thus lowering the runtime and power overheads for differential privacy preservation in community sensing applications.

To address the above thesis, we propose ***SeRaNDiP*** (**Se**nsor **Ra**ndom **N**oise for **Di**fferential **P**rivacy) which utilizes the sensor's inherent noise to meet differential privacy requirements under distributed differential privacy, thus removing the need for noise generation. We make use of the state-of-the-art noise profiling mechanism, Allan Deviation (AD) [89] to characterize and estimate noise components required to satisfy differential privacy requirements for wearable community sensing applications. Further, *SeRaNDiP* leverages various hardware

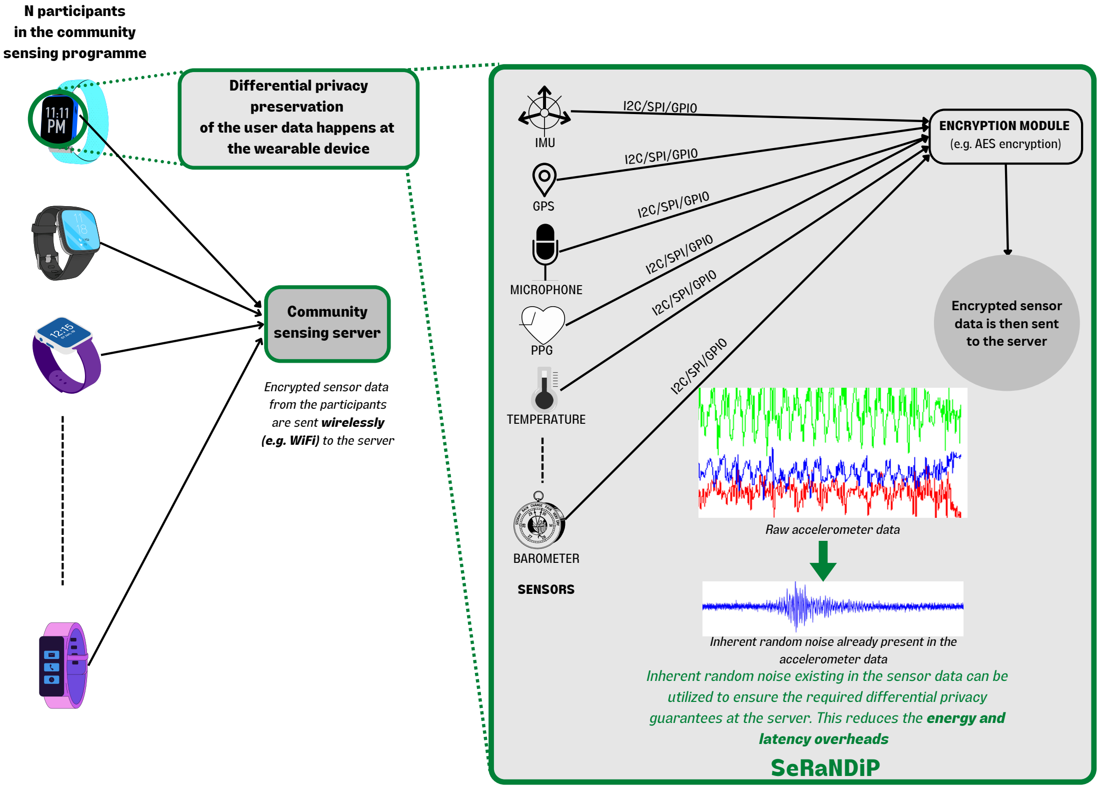


Fig. 2. Overall architecture of our *SeRaNDiP* framework for differential privacy preservation in wearable community sensing applications

configurations of the sensors to introduce different amounts of inherent noise depending on the application-specific differential privacy requirements. Thereby, *SeRaNDiP* guarantees differential privacy requirements for different population sizes whilst ensuring low power and compute requirements on wearables.

The contributions of this paper are summarized as follows:

- We study the latency and energy consumption overheads associated with realizing state-of-the-art DDP on today's wearables with accelerometers, barometers and temperature sensors.
- We propose *SeRaNDiP* which leverages a sensor's inherent noise to meet DDP requirements, thus saving on the need for noise generation for DDP. Hence *SeRaNDiP* is readily applicable to fitness trackers and smartwatches without any hardware modification.
- We analyze the inherent noise of hardware sensors common in today's wearables and demonstrate the specific sensor configurations and application scenarios where the inherent sensor noise suffices for DDP guarantees.

- We implemented *SeRaNDiP* on a low-power ESP32 microcontroller interfaced with accelerometers (MPU-9250 and ADXL-345), barometers (BMP-388 and MLP 315A2) and body temperature sensor (MLX90632) from different manufacturers and evaluated the performance of *SeRaNDiP* against three alternative differential privacy implementations - (a) the baseline state-of-the-art distributed differential privacy [33] of Figure 1, (b) Local differential privacy [77] and (c) Gamma distribution based noise generated distributed differential privacy [14]. Our experiments show that *SeRaNDiP* realizes latency as well as energy improvement of 1.4X-1.8X, 1.3X-1.7X, 1082X-3334X and 1.2X-1.5X, 1.2X-1.8X, 7.4X-11.2X respectively against these three baselines.
- We also conducted user trials with 10 participants and showed that *SeRaNDiP* realizes latency and energy savings while maintaining high accuracy at the community-sensing server.
- We studied the variation in the inherent sensor noise of the accelerometer and barometer sensors under different room temperatures and found that the inherent sensor noise remains stable at different room temperatures - demonstrating the robustness of *SeRaNDiP*.

The remaining of the paper is structured as follows. In Section 2, we review the literature on efficient differential privacy preservation techniques for community sensing applications and measured the power-performance overheads of state-of-the-art DDP implementations on wearables. Section 3 introduces background on DDP theory and sensor noise, analyzing population-scale differential privacy guarantees for community sensing applications. Section 4 explains how the inherent noise that exists in sensor data can be utilized to guarantee differential privacy. Section 5 outlines the various steps involved in our *SeRaNDiP* framework. Experimental validation of our solution against conventional DDP implementation is presented in Section 6. Section 7 discusses the limitations as well as factors that determine the scalability of *SeRaNDiP*. Finally, we conclude the paper in Section 8.

2 RELATED WORKS AND MOTIVATION

In this section, we review the related works with respect to four key characteristics - (1) DP approaches which guarantee privacy using hardware level modifications vs. software algorithms, (2) DP mechanisms that guarantee privacy without random noise generation, (3) DP techniques that consider sensor noise, and (4) Low power privacy mechanisms for IoT devices. Through experiments with an accelerometer sensor, we motivated the need for a low-power and low-compute framework for ensuring differential privacy guarantees in wearables - by characterizing the various overheads associated with the state-of-the-art baseline DDP implementation. Table 1 summarizes the literature in comparison to *SeRaNDiP* based on the above aspects. Similar to a typical DDP application, we assume that the community sensing server accepts the privacy protocol followed by the wearable device but is not trusted to hold the user's raw sensor data.

Table 1. Summary of the related works with respect to *SeRaNDiP*

Related Work	Hardware Level DP	Noiseless Privacy	Sensor Noise with DP	Power Saving
Choi et al. [27]	✓	×	×	✓
Duan et al. [31]	×	✓	×	×
Bhaskar et al. [24]	×	✓	×	×
Sei et al. [77]	×	×	✓	×
Shi et al. [82]	×	×	×	✓
Ács et al. [14]	×	×	×	✓
SeRaNDiP	**✔**	**✔**	**✔**	**✔**

2.1 Software Algorithms and Hardware-level Modifications to Ensure Differential Privacy

Most current differential privacy techniques are implemented as software algorithms that run on the processor of the wearable device, where differential privacy noise is generated using the said software algorithms [14, 34, 71, 82, 87]. While it is easy to augment software-generated noise in DDP applications, it adds a considerable amount of computation overhead (timing and energy consumption) to the system compared to hardware-level modifications. The computation cost involved with random noise generation significantly affects the latency and energy consumption in wearables where battery life, as well as computation capacity, is highly limited due to form-factor restraints.

To bypass the compute overheads associated with software algorithms, Choi et al. [27] proposed a hardware-level local differential privacy implementation by introducing resampling and thresholding-based mechanisms to prevent privacy loss in fixed-point random noise generation hardware. However, it requires hardware modifications in the wearable device, and is thus not applicable to existing wearable devices. In addition, the random number generation hardware module itself [27] also incurs considerable latency and power consumption. On the other hand, *SeRaNDiP* leverages existing sensor hardware in wearables to realize differential privacy without any changes in the hardware. Our proposed approach harnesses inherent sensor noise by configuring various hardware parameters such as sampling frequency, filter settings, etc. available in the sensor to acquire sensor data with the desired amount of random noise needed to ensure a differential privacy guarantee. By doing so, *SeRaNDiP* gets rid of the computation costs (latency and energy consumption) involved in random noise generation at the wearable device.

2.2 Differential Privacy Approaches that Do Not Require Random Noise Generation

Duan et al. [31] mathematically validated that when the data is sufficiently large in a centralized server setting, inherent uncertainty associated with unknown quantities of noise available in the data can be used to guarantee privacy without adding external noise. Bhaskar et al. [24] showed that the required differential privacy guarantee could already be existing in the aggregated user data if it satisfies conditions, e.g., the adversarial attackers have very limited or no information about the data stored in the server and each user entry in the server is independent. Since the availability of auxiliary information is very much applicable in the context of wearables, this approach cannot be applied to wearables. Both [31] and [24] trust the centralized community sensing server where differential privacy is guaranteed. In our work, we consider community sensing servers as not trustworthy and ensure differential privacy guarantees at the sensor hardware on the wearable device before releasing the sensor data to the server.

2.3 DP Techniques that Consider Sensor Noise

Recently, [77] proposes a 2-stage process to estimate sensing errors in the sensor data being received at the community sensing server. Firstly, random noise is generated and added to the raw sensor data at the wearable device to ensure privacy. The server receives the perturbed data and estimates the true data distribution after accounting for the unknown sensing errors. The proposed approach still generates random noise at the wearable device and only tackles the removal of sensing errors observed in the data at the server.

2.4 Low Power Privacy Mechanisms for IoT Devices

As the battery is highly constrained in wearables, it is essential to realize DDP approaches at lower power consumption incurred from the computation and communication of the perturbed sensor data. Shi et al. [82] developed a DP approach that eliminates a complete round of sensor data communication required before decryption to make DP plausible for low-power sensor networks. Lightweight encryption [14] algorithms have also been developed for use in IoT devices. There are privacy approaches [58, 73] that filter sensitive events in the sensor data and replace it with insensitive event data. Since these algorithms are computationally expensive due to the use of machine learning algorithms, they can currently only be run on processors in smartphones or PC. While off-loading such tasks to a smartphone saves limited energy on wearable device, the communication between the smartphone and the wearable device incurs considerable latency as well as power consumption. Therefore, this will not be applicable for wearables that can already communicate directly to the Cloud over WiFi or cellular connectivity.

In our work, since we do not generate random noise for differential privacy requirements, we save a considerable amount of energy and computation whilst still providing a strong differential privacy guarantee. All the above privacy mechanisms are orthogonal to our work; *SeRaNDiP* can co-exist with these approaches to deliver further savings.

2.5 Power and Latency Bottlenecks in State-of-the-art DDP Implementation for Wearables

As detailed power and timing analysis cannot be obtained using commercial smart watches/fitness trackers due to their closed nature, we implemented the state-of-the-art gaussian noise-based DDP[33] (see Figure 1) in a widely used wearable and IoT platform, ESP32. In the rest of this paper, we will name this baseline as **DDP-BL** for brevity. We implemented the differential privacy preservation module on a low-power ESP32 microcontroller which acts as the wearable device and the Raspberry Pi 3b development board (see Figure 3) runs the server with parameters shown in Table 2. We then categorized the delays into four stages - sensing, random noise generation, encryption and BLE communication.

(1) Sensor data acquisition, I2C communication, and pre-processing by ESP32 fall under *sensing delay*.
(2) Random white gaussian noise generation (with standard deviation 0.001) and perturbation as *random noise generation delay*.
(3) AES encryption performed on sensor data is termed *encryption delay*.
(4) Buffering encrypted sensor data and BLE communication time taken to send the encrypted sensor data to the server is categorized as *BLE communication delay*.

Timing measurements were conducted using Arduino software timers and power measurements were made using a Monsoon power monitor. As shown in Figure 4, the random noise generation step in the baseline DDP implementation introduces up to 1.5x computation (SW) and communication (COMM) delays (i.e., 1.4x total delays), as well as 1.3x computation (SW) and communication (COMM) energy overheads (i.e., 1.4x total overheads) to the wearable.

In addition, we also evaluated the latency and energy overheads associated with random noise generation in a typical DDP implementation for five different wearable sensors. Random white gaussian noise was generated using the Arduino gaussian noise generation library [78]. For all the five sensors used in our characterization study - MPU-9250 accelerometer, ADXL345 accelerometer, BMP388 barometer, MPL3115A2 barometer and MLX90632 body temperature sensor, the corresponding hardware configurations can be found in Table 3.

The latency was measured using Arduino software timers and power measurements were made using a Monsoon power monitor. From Figure 5, it is evident that random noise generation adds about 30% latency and energy overheads to the wearable device for all sensors except the body temperature sensor. The reason behind such low overheads in the temperature sensor is the absence of hardware buffers and registers for sensor data access. This eliminates the overheads associated with I2C communication and sensor data acquisition. However, most sensors come with hardware buffers as they significantly reduce sensing delay and power, and thus, the overheads associated with random noise generation will be considerably higher. Our characterization study thus affirm that random noise generation adds significant overheads to wearable devices, motivating the need for a low-power and low-compute differential privacy preservation mechanism for wearable community sensing applications. By exploiting the inherent sensor noise observed in the sensor data, *SeRaNDiP* eliminates the power

Table 2. Hardware configuration details of our ESP32 prototype

Component	Model	Parameter	Setting
Sensor	MPU-9250 accelerometer sensor	Sampling rate	25Hz
		Low Pass Filter cut-off frequency	20Hz
		Range	+/- 2g
		Buffer Size	85 Samples
I2C Connectivity	-	Clock frequency	400 kHz
Processor and Wireless Communication Module	ESP32 Development Board	Version	DOIT ESP32 DEVKIT V1
		CPU Frequency	80MHz
		Voltage	5V
BLE Connectivity	-	Connection interval	6.25- 7.5 ms
	-	Buffer Size	340 Bytes
	-	MTU	345
BLE Server	Raspberry Pi Board	Version	Raspberry Pi-3b

Table 3. Sensor Parameters used in our Characterization Study

Sensor	Model	Parameters	Value
Accelerometer	MPU-9250	Low pass Filter(LPF) cut-off frequency	20Hz
		Range	+/-2g
		Sampling Rate	25 Hz
		Buffer Size	85 Samples
	ADXL345	Range	+/-2g
		Sampling Rate	25 Hz
		Buffer Size	32 Samples
Barometer	BMP388	Pressure Oversampling	x4
		IIR filter	OFF
		Temperature Oversampling	SKIP
		Sampling Rate	0.78 Hz
		Buffer Size	72 Samples
	MLP 3115A2	Oversample	x4
		Sampling Rate	1 Hz
		Buffer Size	32 Samples
Body Temperature Sensor	MLX90632	Emissivity	0.987
		Room Temperature	25^oC
		Sampling Rate	1 Hz
		Buffer Size	Buffer is Unavailable

and computation overheads caused by random noise generation and perturbation in state-of-the-art baseline DDP implementation. In Section 6, we show that SeRaNDiP offers 1.4-1.8X latency improvements and 1.2X-1.5X energy savings against DDP-BL.

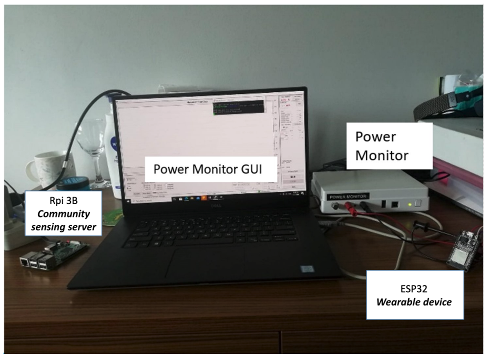


Fig. 3. Implementation of the state-of-the-art baseline DDP system (DDP-BL). The ESP32 microcontroller acts as the wearable device which performs random noise generation to meet DDP requirements. The Rpi 3B board runs the community sensing server to which the encrypted sensor data is being sent. A Monsoon power monitor is used to measure the power consumption of ESP32.

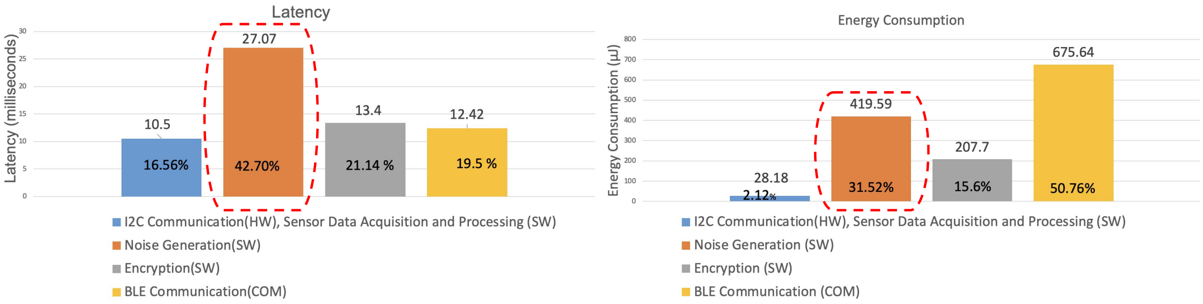


Fig. 4. Profiling state-of-the-art baseline DDP implementation (DDP-BL) w.r.t. total latency and energy consumption

3 THEORY AND BACKGROUND

In this section, we briefly review the theorems of differential privacy that form the basis of SeRaNDiP, followed by the Allan variance analysis for identifying that various wearable sensors exhibit inherent sensor noise that is gaussian, before presenting the theoretical foundations for estimating the population size needed to provide theoretical differential privacy guarantees.

3.1 Differential Privacy

Differential Privacy (DP) is a well-known privacy technique for providing the necessary privacy guarantees at the server by utilizing the statistical properties of observed user data. It ensures trust between the community sensing server and the user by making sure an adversary is unable to re-identify the user through observing his/her data stored at the server. Given a differentially private community sensing server with user data, the output of the server does not change significantly even if a user's record has been added, removed, or modified owing to the mathematical guarantees provided by the differential privacy algorithm. Hence, it is almost impossible

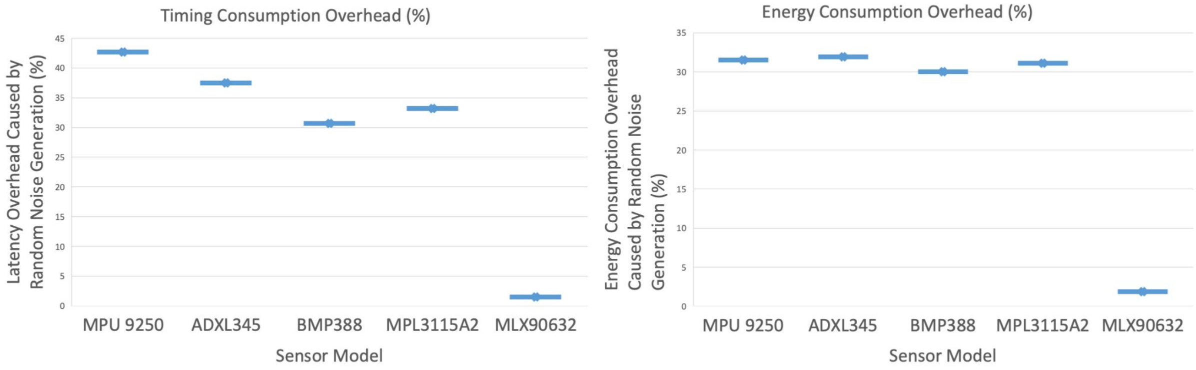


Fig. 5. Energy and latency overheads associated with random noise generation (%)

for adversaries to glean any information regarding a specific participant. We summarise the key mathematical definitions and theorems of differential privacy below.

Definition 3.1 (Differential Privacy [32]). A privacy mechanism M satisfies (ϵ, δ)-differential privacy $(\epsilon, \delta > 0)$, if we have

$$Pr[M(D_1) \in S] \leq \exp(\epsilon) \cdot Pr[M(D_2) \in S] + \delta \tag{1}$$

for all adjacent databases D_1 and D_2 which differ on at most one record, and for all sets S ⊆ Range(M). Typically, the gaussian mechanism is used to implement DP where random noise is drawn from a zero-mean gaussian distribution and added to the query output $f(D)$ with the gaussian noise scaled to l_2 sensitivity [33].

Definition 3.2 (l_2 sensitivity [33]). For any function $f : D \longrightarrow R^d$, the l_2 sensitivity of f w.r.t. D is given as

$$\Delta_2(f) = \max_{D_1, D_2 \in D} \|f(D_1) - f(D_2)\|_2 \tag{2}$$

for all D_1, D_2 differing on at most one record. $\Delta_2(f)$ measures the Euclidean distance as follows:

$$\Delta_2(f) = \sqrt{\sum_i [f_i(D_1) - f_i(D_2)]^2} \tag{3}$$

For any function $f : D \longrightarrow R^d$, the *Gaussian mechanism* with parameter σ for any dataset D is $M(D) = f(D) + N(0, \sigma^2)^d$. We have the following theorem.

Theorem 3.3 (Gaussian Mechanism [33]). *Let $\epsilon \in (0, 1]$ be arbitrary. For $c^2 > 2\ln(1.25/\delta)$, the Gaussian mechanism with parameter $\sigma \geq c\Delta_2(f)/\epsilon$ is (ϵ, δ)-differentially private.*

In addition, based on the results in [65], we have the following corollary on the overall privacy guarantee of multiple invocations of the Gaussian mechanism.

Corollary 3.4. *Let $f_1, \ldots, f_k$ be a set of functions. For $c^2 > 2\ln(1.25/\delta)$, applying the Gaussian mechanism on $f_1, \ldots, f_k$ with parameter $\sigma \geq \sqrt{k} \cdot c\Delta_2(f)/\epsilon$ is (ϵ, δ)-differentially private.*

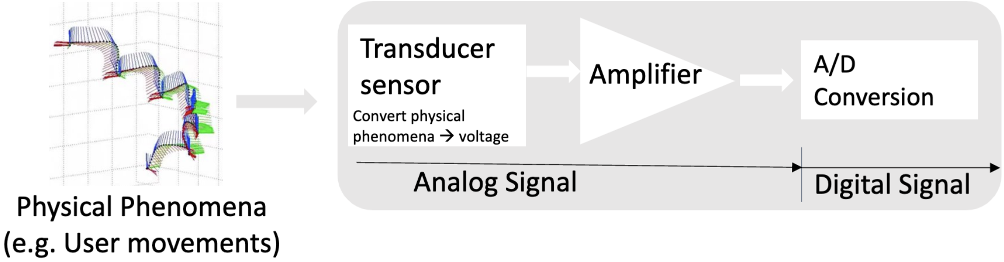

Transducer sensor
Convert physical phenomena → voltage
Amplifier
A/D Conversion
Analog Signal
Digital Signal
Physical Phenomena (e.g. User movements)

Fig. 6. Data acquisition unit of a typical wearable Sensor

3.2 Presence of Inherent Noise in Wearable Sensors

Today's wearable devices are equipped with a multitude of sensors like accelerometers, barometers, body temperature sensors, etc. The data acquisition hardware in a sensor comprises three main components - a transducer, an amplifier, and an Analog to Digital (A/D) converter (see Figure 6). The transducer captures physical phenomena such as user movements and converts them to a corresponding voltage value. The voltage is then fed to an internal amplifier for further amplification before being converted to a digital signal via an ADC chip. Electrical components such as resistors, capacitors, etc. present in amplifiers produce a considerable amount of stochastic noise along with the sensor reading [4]. The observed electrical noise distorts the signal considerably and hence, hardware developers reduce the observed electrical noise at various stages of sensor design through configurable hardware filters, more advanced filters [39, 45, 49, 60], and signal post-processing algorithms [21, 55, 84].

3.2.1 Stochastic Noise in Sensors. Stochastic noise in the observed sensor data comprises random noise components - which can also be represented as random variables having a certain probability distribution [68, 80]. In a typical DDP implementation, white gaussian noise is generated at the wearable device and added to the raw sensor data before being sent to the community sensing server. Fortunately, white gaussian noise is one of the random noise components observed in the sensor data. To identify white gaussian noise in the stochastic sensor noise, a well-known noise analysis method can be used - Allan Deviation (AD) analysis [59].

3.2.2 Allan Deviation Analysis of Sensor Noise. Allan deviation analysis is used for identifying the various sources of noise in stationary sensor data [46, 59]. Taking an accelerometer sensor as an example, given N samples of accelerometer sensor data at a sample time of τ_0, we can form data clusters of duration $\tau_0, 2\tau_0, 3\tau_0.., m\tau_0$ where $m < (N-1)/2$. We then obtain the averages of the sum of the data points contained in each cluster over the length of the cluster. The Allan variance (in this case) is defined as the two-sample variance of the data cluster averages as a function of cluster time. However, the clusters are overlapping in the assumed Allan variance estimation. The Allan variance estimation performs much better when the clusters are non-overlapping. Therefore, the Allan deviation function divides the sampled accelerometer signals into non-overlapping cluster $\bar{\Omega}_k(\tau)$ averages over a cluster duration τ and computes the variance among each cluster group as a function of varying τ.

$$\sigma_a(\tau) = \sqrt{\frac{1}{2(N-2n)} \sum_{k=1}^{N-2n} [\bar{\Omega}_{k+1}(\tau) - \bar{\Omega}_k(\tau)]^2} \qquad (4)$$

where $n = \tau/\Delta t$, N = total number of samples in $\Omega(t)$, Δt =sampling period and

$$\bar{\Omega}_{k+1}(\tau) = \frac{1}{(\tau)} \int_{t_k}^{t_k+\tau} \Omega(t)dt, \;\; \Delta t \leq \tau \leq N\Delta t/2. \tag{5}$$

Figure 7 shows a plot of Allan deviation values against different cluster time duration (τ). Prior research [46] has identified the relationship between the slope of the Allan deviation plot and contributing noise sources in the observed sensor data. As seen in Figure 7, there are different slopes for different τ and each stochastic noise component is prominent at a unique cluster time duration (τ). Therefore, each specific stochastic noise component can be obtained by looking at a specific τ.

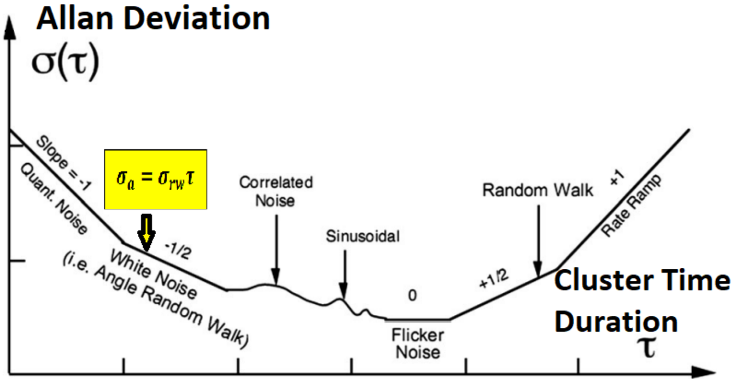


Fig. 7. Sample plot of Allan Deviation analysis results (Source : [46])

Similarly, noise resulting from barometer readings have a significant white gaussian noise component since the Allan deviation plot has a slope of -0.5 [59]. Thermopile detectors used for contactless body temperature sensing in wearables [3] also produce an inherent thermal noise (Johnson noise) which is identical to white gaussian noise [29].

3.3 Theoretical Estimation of Sensor Noise Required from Each User for Providing DP Guarantees to a Given Population Size

For a community sensing program with a given number of participants (e.g. 10 million or 100 million participants), there is a minimum amount (standard deviation) of inherent sensor noise required from each user's sensor data to provide a differential privacy guarantee. In this subsection, we explain how to estimate the minimum standard deviation of white Gaussian noise required from each user's sensor data to provide distributed differential privacy at the community sensing server with a known number of participants. We also study how the DDP guarantee is affected by machine learning-based classifiers applied to the aggregated user data at the community sensing server.

To begin with, we apply Theorem 3.3 to estimate the standard deviation of sensor noise required from each participant. δ in Theorem 3.3 is represented as 1/(number of participants in the community sensing program) and $\Delta_2(f)$ is the measurement range (difference between the maximum and minimum sensor reading). The standard deviation estimated using Theorem 3.3 is the total standard deviation of accumulated inherent sensor noise from all the participants under a typical DDP setup ($\sigma_{\text{a+b+....}}$). The total variance of the inherent sensor noise is the sum of variances from all the participants.

$$\sigma^2_{a+b+...+N} = \sigma^2_a + \sigma^2_b + ... + \sigma^2_N \tag{6}$$

Therefore, the standard deviation of the inherent sensor noise required from each user to provide a $(\epsilon, \delta)-$DDP guarantee would be obtained as,

$$\sigma_a = \sigma_{a+b+...+N}/\sqrt{n} \tag{7}$$

Community sensing servers also perform data analytics on the aggregated data from each user to evaluate the community's behavior of interest whilst still providing a privacy guarantee. Based on the data analytics required, the desired ML/DL-based classification algorithms are used. However, classifiers usually require consecutive time series sensor data to estimate a community's common behavior at a particular time. Given the classifier requires j consecutive samples of the sensor data as input, we know from Corollary 3.4 that $\sqrt{j} \cdot c\Delta_2(f)/\epsilon$ would be the required SD of inherent white gaussian noise in the sensor data to ensure (ε, δ)-differential privacy. Therefore, each user's sensor data should have an inherent white gaussian noise with a standard deviation(σ_a) of -

$$\sigma_a \geq \sqrt{j} \cdot c\Delta_2(f)/\sqrt{n}\epsilon \tag{8}$$

to provide (ϵ, δ)-distributed differential privacy guarantee at the server after applying the classifier. This means that the amount of standard deviation (SD) of inherent sensor noise required to ensure differential privacy will also increase (by $\sqrt{j}$) with the number of consecutive time series sensor data samples (j) required by the ML/DL classifier employed at the server.

4 EXPERIMENTAL VALIDATION OF INHERENT WHITE GAUSSIAN NOISE OBSERVED IN SENSORS

In this section, we experimentally validated the inherent noise observed in the sensor data and show that it would be sufficient for meeting differential privacy requirements. The inherent sensor noise can be harnessed to get rid of the runtime and energy overheads associated with random noise generation. Through our experiments, we answer the following questions:

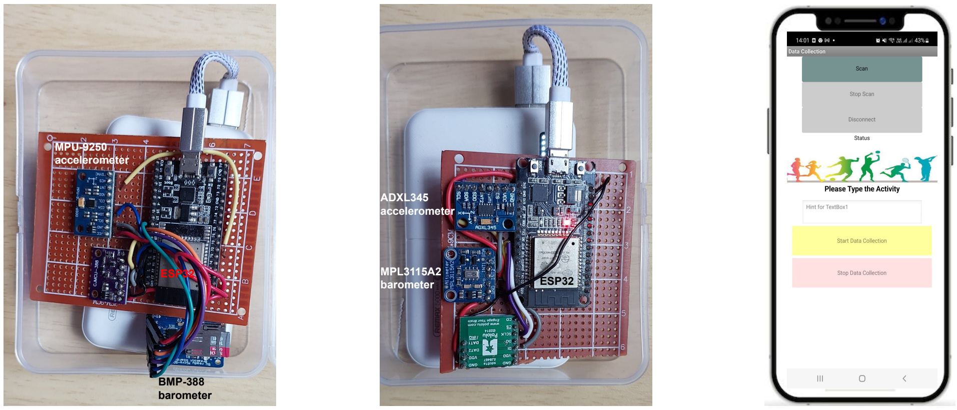
Q1. Is inherent white gaussian noise present in experimental sensor data? (see Section 4.1)
Q2. Can the amount of hardware sensor noise be controlled by software? (see Section 4.2)
Q3. Will the inherent sensor noise be enough to satisfy the differential privacy requirements for a particular community sensing application with a known participant count? (see Section 4.3)

4.1 Is Inherent Gaussian Noise Present in the Sensor Data? (Q1)

Here, we experimentally show that inherent white gaussian noise is observed even in the stationary sensor data. This validates our assumption that inherent sensor noise is caused by the electrical noise arising from the sensor hardware rather than the noise due to environmental factors and motion artifacts.

4.1.1 Prototype Details: We built four prototypes using the ESP32 development board, accelerometer, barometer sensors, and an SD card module. Two of the ESP32 prototypes (Prototype v1 in Figure 8) were interfaced with an MPU-9250 accelerometer from InvenSense[1] and a BMP388 barometer from Bosch[2]. Another two of the ESP32 prototypes (Prototype v2 in Figure 8) had an ADXL345 accelerometer from Analog Devices[3] and a MPL3115A2 barometer from NXP[4]. We used existing firmware libraries to acquire raw sensor data from the ESP32

[1] https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/
[2] https://www.adafruit.com/product/3966
[3] https://www.analog.com/en/products/adxl345.html
[4] https://www.nxp.com/docs/en/data-sheet/MPL3115A2.pdf



Prototype v1

Prototype v2

Android app for
data collection

Fig. 8. Our ESP32 sensor prototypes (Prototype v1 and Prototype v2) interfaced with sensors from different manufacturers and the Android app UI for data collection.

board [15, 61, 90, 91]. Each sensor was interfaced to ESP32 via I2C and the SD card module was interfaced via SPI connection. An Android app for data acquisition from ESP32 was developed using MIT App Inventor library(see Figure 8). The application controls sensor data logging and communicates with the ESP32 prototypes via BLE. Since the programmable amplifiers in digital sensors produce a higher amount of noise at a lower sampling rate [4], each sensor was configured to the lowest reasonable frequencies for community sensing applications: 3.13 Hz for accelerometers, 0.78 Hz for BMP 388 and 0.125 Hz for MPL3115A2. The lower sampling frequencies chosen for the sensors are practical for most existing community sensing applications. In Section 6, we also show that lower sampling frequencies of the sensors do not have a considerable effect on the accuracy of the community sensing applications.

4.1.2 Experimental Setup: *SeRaNDiP* primarily focuses on wearables and hence we collected sensor data for both stationary and mobile settings to reflect practical deployment scenarios. We recruited 10 participants (5 male, 5 female) for our experiments. For the accelerometer sensor, We considered standing as a stationary setting and jogging, walking, or any other horizontal activity as a dynamic mobile setting. We first asked the participants to wear our ESP32 prototypes on the left leg pocket (the y-axis of the accelerometer is affected by gravity). The participants were requested to stand still for 5 minutes before engaging in the following activities - jogging, walking and climbing up stairs, each for a duration of 5 minutes. For the barometer sensor, we kept the prototypes on a still table to emulate a stationary setting and escalators up/down, elevators up/down, and stairs up/down as a dynamic setting. Prototype v1 was worn by the participants with user IDs 3, 7, 8 and 10. Prototype V2 was worn by the remaining participants. Next, we obtained the sensor data from the prototypes and fed the noise traces (observed sensor data - mean (observed sensor data)) to the Allan Deviation algorithm to calculate the standard deviation of white gaussian noise present in the sensor data.

4.1.3 Results: As shown in Figure 9, the standard deviation of the white gaussian noise of the accelerometer is higher in a dynamic setting compared to the stationary setting. We also obtained similar plots for the barometer sensors (see Figure 10). *SeRaNDiP* can thus provide the minimum privacy guarantee with the available white gaussian noise in a stationary setting and much higher privacy guarantees in a dynamic setting. The evaluations affirm the existence of inherent white gaussian noise to aid in the elimination of compute-intensive random noise generation in DP.

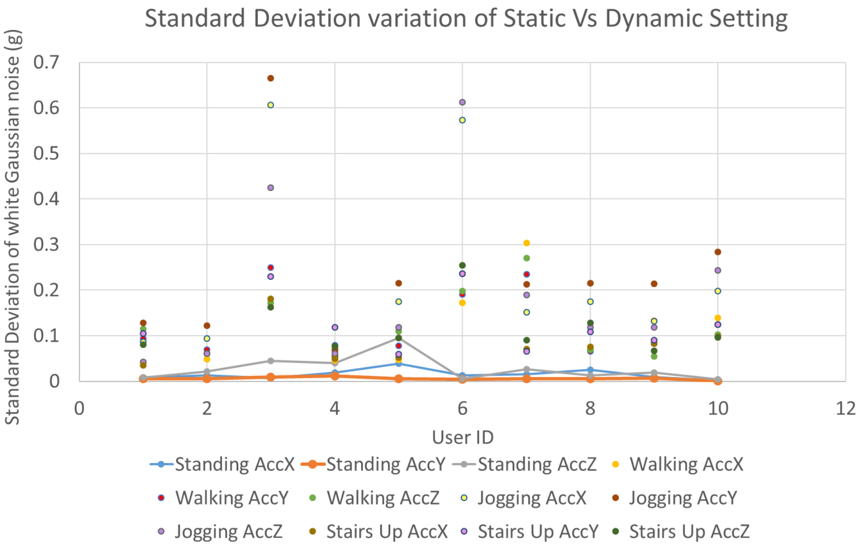


Fig. 9. Standard deviation (SD) values of the inherent white gaussian noise observed in the accelerometer sensor during stationary and dynamic settings.

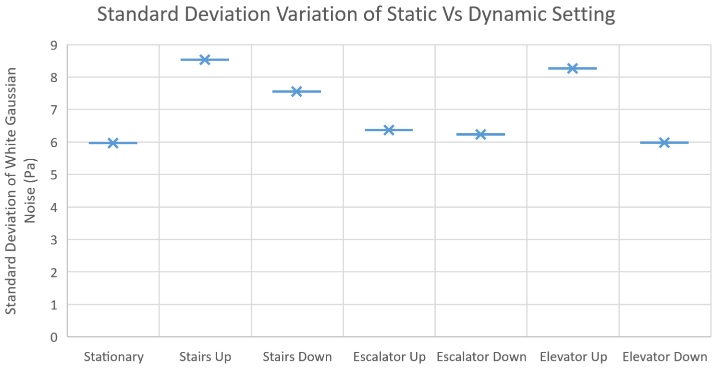


Fig. 10. Standard deviation(SD) values of the inherent white gaussian noise observed in the barometer sensor during stationary and dynamic settings

4.2 Impact of Sensor Hardware Configuration on the Amount of Inherent Sensor Noise (Q2)

Here, we show that the programmable hardware configurations of the sensor can be controlled to produce varying amounts of inherent sensor noise from the same sensor without any hardware modification.

4.2.1 Experimental Setup: We varied the sampling rate of the accelerometer and barometer sensors in our earlier prototypes and collected the noise traces from the observed sensor data in a stationary setting. The prototypes were left on a still table to emulate a stationary setting. In addition to the sensors in our earlier prototypes, we also interfaced MLX 90632 infrared contactless body temperature sensor to the ESP32 board via I2C and placed a finger on it to obtain noise traces under a stationary setting. The configured hardware filter parameters for each sensor are listed in Table 4. While we use the best-performing hardware filter configurations in this experiment, various parameters of the digital filters in the sensor hardware can be modified to produce different amounts of inherent sensor noise.

4.2.2 Observations: As observed in Figures 11 and 12, the standard deviation of white gaussian noise decreases with increasing sampling rate regardless of sensor manufacturer or the sensor type. We can thus conclude that the sampling rate can be varied to produce the desired amount of inherent noise in the observed sensor readings - with the highest sensor noise produced at the lowest sampling frequency. Since various sensor configurations can result in different amounts of inherent white gaussian noise, the community sensing app can set the user's wearable device's sensor configuration based on the amount of inherent sensor noise required from each user.

Table 4. Hardware parameters of the sensors used in our study

Sensor	Model	Parameters	Value
Accelerometer	MPU-9250	Low pass Filter(LPF) cut-off frequency	460Hz
		Range	+/-2g
	ADXL345	Range	+/-2g
Barometer	BMP388	Pressure Oversampling	x4
		IIR filter	OFF
		Temperature Oversampling	SKIP
	MLP 3115A2	Oversample	x4
Body Temperature Sensor	MLX90632	Emissivity	0.987
		Room Temperature	25^oC

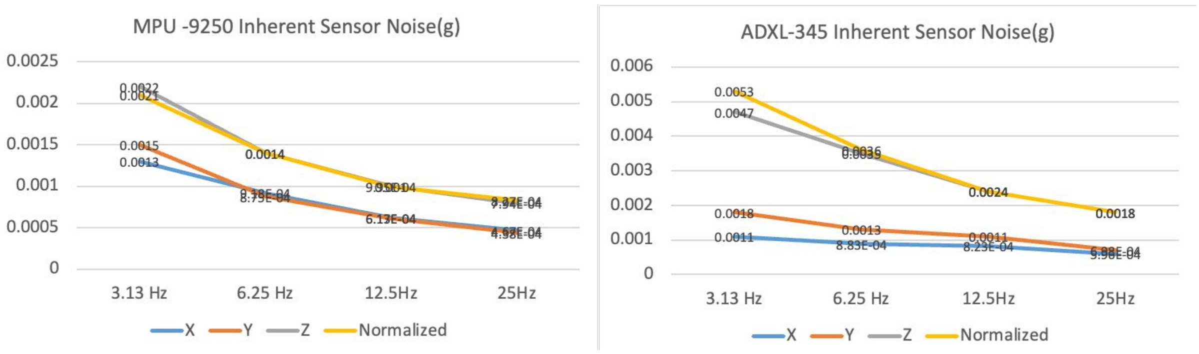


Fig. 11. Variations in the SD of White gaussian noise in the accelerometer sensor w.r.t. sampling rate

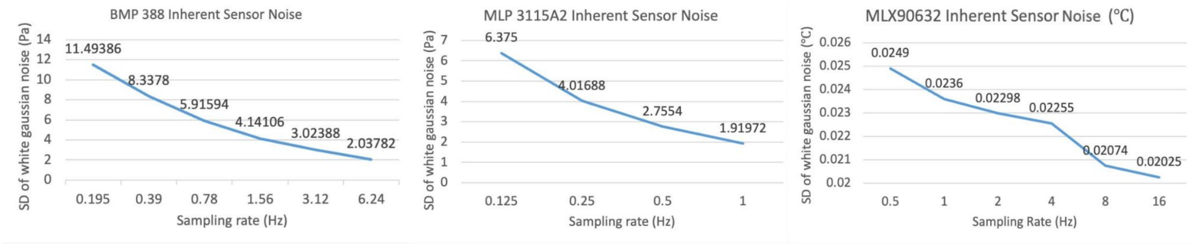


Fig. 12. Variations in the SD of white gaussian noise in the barometer and temperature sensor w.r.t. sampling rate

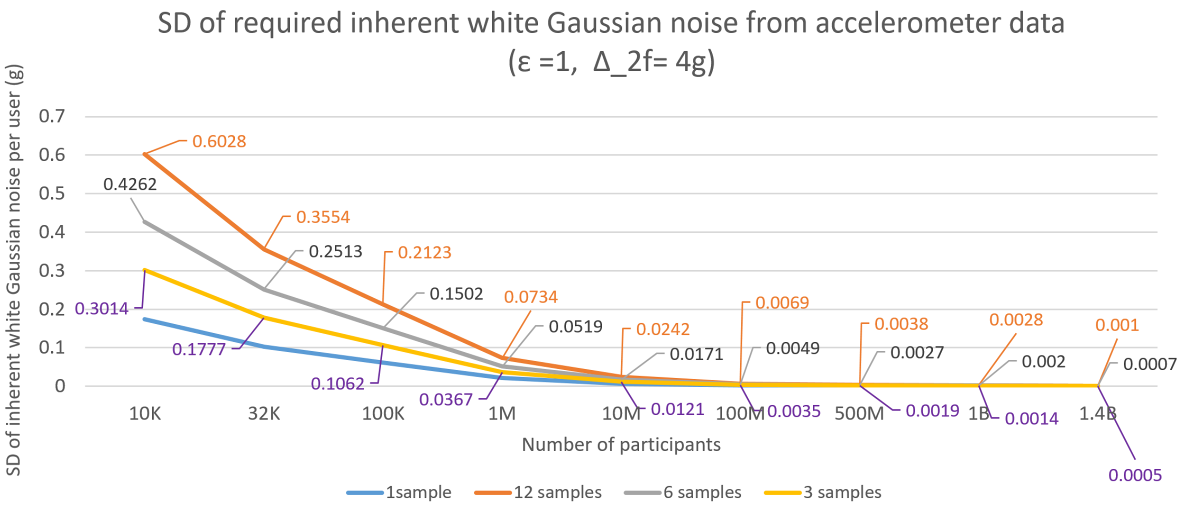


Fig. 13. SD of the inherent sensor noise required per user for the accelerometer sensor based on the number of participants and classifier's input size

4.3 Inherent Sensor Noise Meets Differential Privacy Requirement (Q3)

Finally, we empirically show that the inherent sensor random noise is sufficient for satisfying differential privacy guarantees required for a community sensing program. Based on Equations 7 and 8, Figure 13 shows the minimum SD of inherent white gaussian noise (when $\epsilon = 1$) from the accelerometer sensor of each user for different population sizes. The SD of inherent white gaussian noise required from each user varies inversely with the number of participants. As explained in Section 3.3, the number of consecutive sensor data samples required by the community sensing server is a critical bottleneck in providing the desired amount of inherent sensor noise from each user. From Figure 13, the SD of inherent sensor noise required from each user is directly proportional to the number of consecutive data samples.

Prior works [53] have shown that accelerometer-based activity classification can be done with >90% accuracy with just 4 seconds of data sampled at 5Hz. From Figure 11, we find that both MPU-9250 and ADXL345 accelerometers' sensor data have the highest amount of inherent sensor noise when the sampling rate was 3.13 Hz. Therefore, we configure the accelerometer to the lowest sampling rate of 3.13 Hz, a sensing range of ±2g, and a 460Hz low pass filter cut-off frequency. If fewer number of consecutive sensor data samples are used as input to the classifier at the community sensing server, it reduces the amount of inherent sensor noise required from each user. Hence, we input 1 second of consecutive data samples (3 samples for a 3.13Hz sampling rate) to the classifier at the community sensing server. As explained earlier in Section 4.1, we considered standing as a stationary setting and obtained the best inherent noise-producing sensor hardware configuration empirically from the user's stationary accelerometer data. From the data collected from 10 participants while standing, Acc X, Acc Y, and Acc Z values had a white gaussian noise with average standard deviation values of 0.017 g, 0.007g, and 0.03g respectively. From Section 3.3, *SeRaNDiP* can provide differential privacy guarantees for community sensing programs with a minimum of 4.7 million, 27.5 million, and 1.5 million participants for Acc X, Acc Y, and AccZ data respectively. Thus, we can conclude that the community sensing program (which needs data from all three accelerometer axes) must have at least 27.5 million participants to ensure a differential privacy guarantee with *SeRaNDiP*. Similar to standing, we also observed the inherent white gaussian noise available in the accelerometer sensor data for stationary activities like sitting, lying, etc. and observed that the SD of white gaussian noise was around 0.002g. Therefore, when all the participants are completely stationary, *SeRaNDiP* will require at least 337 million participants to ensure differential privacy guarantees. Fortunately, as the accelerometer sensor's intent is for monitoring motion, stationary data is not as useful. In a practical activity detection application with *SeRaNDiP*, phases with motion can be locally detected at the wearable, and inherent sensor noise is only leveraged during these phases, while stationary phases send a predefined white gaussian noise.

Though a differential privacy guarantee of ϵ=1 is the theoretical goal, most practical deployments of community sensing programs utilize a lower differential privacy guarantee (ϵ) ranging between 2 to 34.9 as shown in Table 5. From Corollary 3.4, we find that the SD of inherent sensor noise required from the user decreases with the differential privacy guarantee (ϵ). For the MPU-9250 accelerometer sensor data sampled at 3.13Hz, the SD of inherent sensor noise required from each user against differential privacy guarantee (ϵ) is shown in Figure 14. For instance, if a community sensing program requires a less stringent ϵ=16, we can guarantee DP for a minimum population size of just 21.06 million participants as opposed to the minimum population size of 337 million participants with the stricter privacy guarantee of $\epsilon = 1$ (when all the participants are stationary). Similarly, when all the participants are in standing poses (for instance, applications that focus on tracking exercise-related health), the minimum population size for $\epsilon = 16$ would be just 1.72 million participants as opposed to the minimum population size of 27.5 million participants with $\epsilon = 1$.

Table 5. Differential Privacy Guarantee(ϵ) Requirement in Existing Community Sensing Programs [7]

Data collector	Real-world DP application	ϵ value
Apple and Google	Exposure Notification framework	8
Apple	QuickType suggestions	16
	Emoji suggestions	4
	Lookup hints	8
	Health Type Usage	2
	Safari Energy Draining Domains	8
	Safari Crashing Domains	8
	Safari Autoplay Intent Detection	16
LinkedIn	Labor Market Insights	28.8
	Audience Engagements API	34.9
OhmConnect	Energy Differential Privacy	4.72
United States Census Bureau	Post-Secondary Employment Outcomes	3
	2020 Census Redistricting Data	19.61

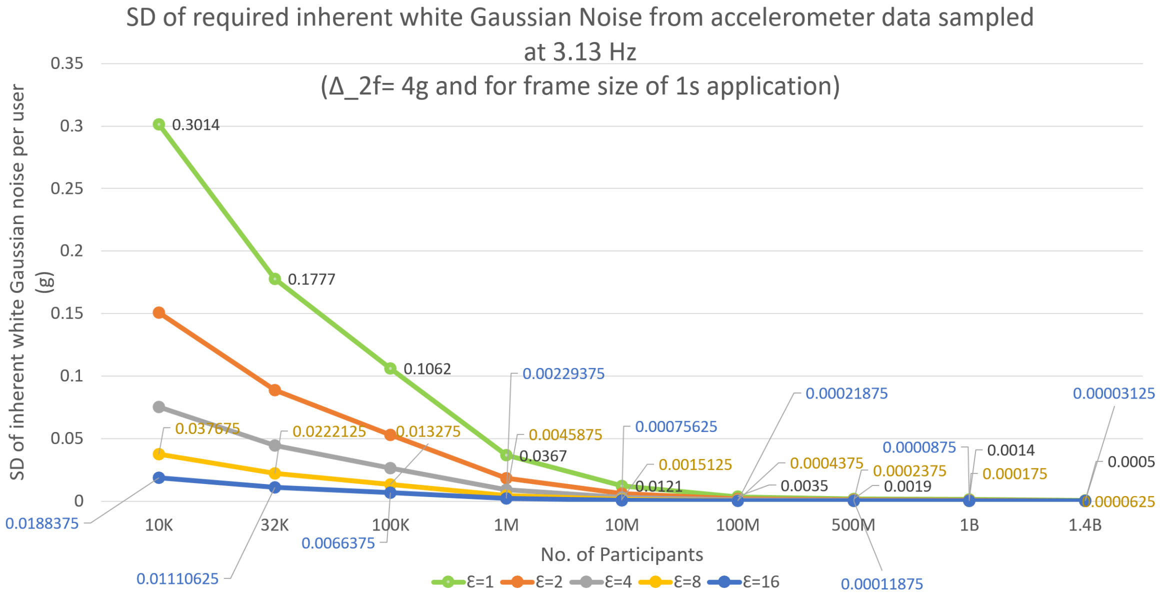


Fig. 14. SD of the inherent sensor noise required per user for the MPU-9250 accelerometer sensor for providing different DP guarantees(ϵ) w.r.t. a given population size.

Figure 15 shows the required SD of inherent sensor noise from each user's barometer to ensure differential privacy guarantee for different population sizes. We only need two consecutive samples to determine the pressure difference in a barometer data. From Figure 12, the SD of inherent sensor noise (from BMP388 barometer with 0.78Hz sampling rate, pressure oversampling x4, temperature oversampling OFF and from MLP3115A2 barometer with 0.125 Hz sampling rate, oversampling x4 settings) can satisfy differential privacy requirements for a community size of at least 0.8 million participants. For a lower differential privacy guarantee (ϵ = 16), *SeRaNDiP* can offer differential privacy guarantee for a population size of just 50,000 participants with the barometer sensor.

Figure 15 plots the SD of inherent sensor noise required from a body temperature sensor (MLX90632) for ensuring differential privacy guarantees in community fever sensing applications with different population sizes. Since a body temperature greater than 37.5^oC is fever, we only require a single temperature value for the classification. From Figure 12, we find that the highest amount of white gaussian noise (SD = 0.025^oC) was observed when the temperature sensor was sampled at 0.5Hz. Thus, *SeRaNDiP* can guarantee differential privacy guarantee for such a community fever sensing program with at least 3.64 million participants (see Figure 15). With a higher ϵ =16, only 227.5K participants are required for the body temperature sensor.

For all three sensors studied, differential privacy requirements can be satisfied at a higher sampling frequency when more participants take part in community sensing. We acknowledge that higher sampling rates provide fine granularities in the sensor data. However, sensors operated at such high sampling rates also contribute to the power consumption of the wearable device. Our experiments indicate that the choice of sampling rate needs to appropriately consider privacy in the face of data utility and battery life.

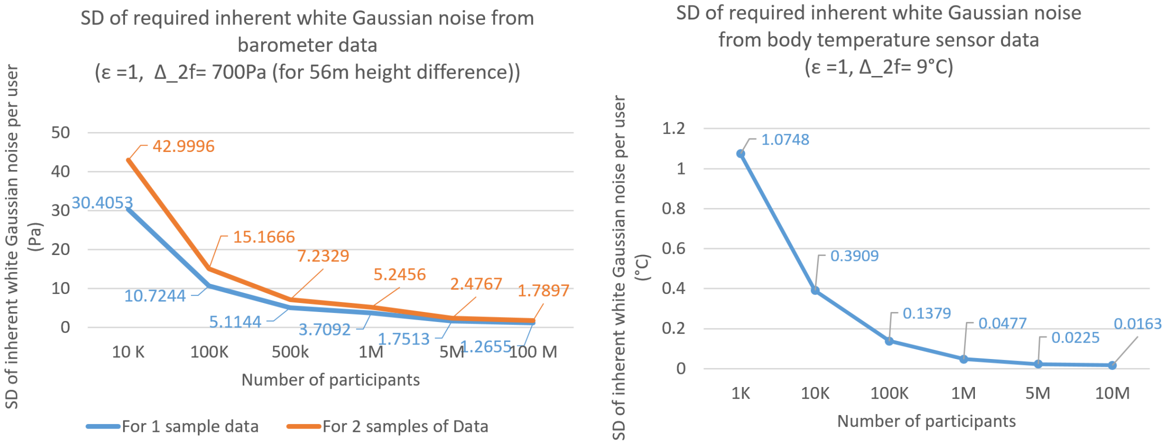


Fig. 15. SD of the inherent sensor noise required per user for the barometer and body temperature sensor based on the number of participants and classifier's input size

5 SYSTEM DESIGN

Based on our observations from Section 3.3, we propose a low-power and low-compute *SeRaNDiP* framework for providing differential privacy guarantees in wearable community sensing applications. *SeRaNDiP* has three main stages - *Hardware configuration*, *Real-time data streaming* and *Inherent sensor noise update*.

Hardware configuration parameters of the sensor and the corresponding SD of the inherent sensor noise for each sensor are stored as tables in the external flash memory of wearables. For sensors like accelerometers, the sensor output can be configured for multiple ranges (e.g. accelerometers can be configured to produce sensor outputs in the range of ±g, ±2g, ±4g, ±8g, etc.). For sensors whose output range can be configured, the inherent sensor noise also varies with the sensor range. For sensors with a configurable range, the hardware configuration parameters of the sensor and the corresponding SD of inherent sensor noise for each range configuration will be stored as sub-tables within the table allocated to the sensor. Hardware configuration parameters include (but not limited to) - Low pass filter cut-off frequency, oversampling ratio, sampling rate, etc. The records in the hardware configuration table for each sensor are sorted in ascending order of the SD of the inherent sensor noise. During each session, the community sensing server estimates the minimum required SD of the inherent noise from each user based on the number of participants, range of the sensor readings, etc. This minimum required SD is sent to each user's wearable device, which next runs the configuration step of *SeRaNDiP* to choose the appropriate hardware configuration to produce the desired amount of inherent sensor noise.

5.1 Hardware Configuration Step

After receiving the differential privacy requirements from the community sensing server, the wearable device chooses the appropriate configuration sub-table from flash memory w.r.t the sensor range requested by the community sensing server. The exact hardware configuration that can produce sensor data with the requested standard deviation of white gaussian noise is obtained from the configuration sub-table using Binary Search (BS) algorithm. The sensors' hardware registers are then configured for the chosen configuration and the chosen sampling Rate (SR) is sent back to the community server as shown in Figure 16.

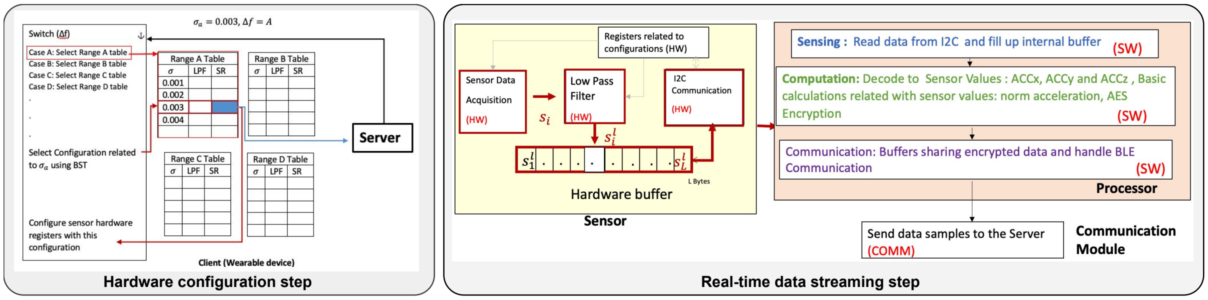

$\sigma_a = 0.003, \Delta f = A$
Switch (Δf)
Case A: Select Range A table
Case B: Select Range B table
Case C: Select Range C table
Case D: Select Range D table
Select Configuration related to σ_a using BST
Configure sensor hardware registers with this configuration
Range A Table
σ
LPF
SR
0.001
0.002
0.003
0.004
Range B Table
Range C Table
Range D Table
Server
Client (Wearable device)
Hardware configuration step
Registers related to configurations (HW)
Sensor Data Acquisition (HW)
s_i
Low Pass Filter (HW)
I2C Communication (HW)
s_i^l
s_1^l
s_L^l
L Bytes
Hardware buffer
Sensor
Sensing : Read data from I2C and fill up internal buffer (SW)
Computation: Decode to Sensor Values : ACCx, ACCy and ACCz , Basic calculations related with sensor values: norm acceleration, AES Encryption (SW)
Communication: Buffers sharing encrypted data and handle BLE Communication (SW)
Processor
Send data samples to the Server (COMM)
Communication Module
Real-time data streaming step

Fig. 16. Hardware configuration and real-time data streaming steps in *SeRaNDiP*

5.2 Real-time Data Streaming Step

After configuring the sensor hardware to the chosen hardware configuration, wireless data streaming is done from the wearable device to the community sensing server as shown in Figure 16. First, low pass filtering via the in-built hardware digital filters present in the sensor is applied to the acquired sensor data and the filtered data is stored in the hardware buffer of the sensor. As the sensor and the wearable processor are interfaced over I2C/SPI, the sensor data from the hardware buffer are read by the processor once the hardware buffers are full. At the processor, there are three main tasks - sensing, computation(encryption) and communication of the sensor data happens. The sensing task reads the hardware buffer of the sensor and fills up an internal buffer in the wearable processor, the computation task performs calculations like computing the normalized 3-axis accelerometer values, AES encryption, etc., and the communication task buffers the AES encrypted processed sensor data to send wirelessly to the server.

Since *SeRaNDiP* offers distributed differential privacy, encryption still has to be performed on the raw sensor data at the wearable device to preserve user privacy [71, 82]. However, resource-constrained devices such as smartphones and wearables cannot run intensive cryptographic algorithms continuously. Therefore, wearables have built-in low-power AES hardware accelerators to perform on-device encryption of the sensor data before sharing it with the community sensing server. At the server, the encrypted data cannot be used directly for data analytics. Homomorphic encryption is a well-known encryption scheme that allows servers to perform computations on the user data in its encrypted format [88]. Thus, AES to homomorphic encryption conversion [17] happens at the server. To perform homomorphic encryption, AES-ciphertexts (AES encrypted sensor data) are received from each participant and converted to the corresponding homomorphic ciphertexts (homomorphic encrypted sensor data) through an open-source AES-to-homomorphic conversion library [41]. Then, the homomorphic ciphertexts from all the participants are added together to obtain an aggregated homomorphic ciphertext. The server distributes the AES encryption keys to all the participants when they sign up for the community sensing program through a secure channel. The server also stores the corresponding decryption key for decrypting the aggregated homomorphic ciphertext. Using the decryption key, the server decrypts the aggregated homomorphic ciphertext to obtain aggregated(summed) sensor data from all the participants. As the number of participants per session varies and classifiers are trained on different population sizes, the average of aggregated sensor data is fed to ML/DL classifiers in the server. Since the aggregated(summed) sensor data from all the participants preserve privacy and the average of the aggregated sensor data is a statistic derived from the aggregated sensor data, the same level of differential privacy is guaranteed [93].

5.3 Inherent Sensor Noise Update Step

Hardware configuration tables on the wearables are updated locally and the Algorithm 1 updates the hardware configuration table for a specific hardware configuration **X**. We first configure the sensor hardware to **X** configuration and collect sensor data with the specified sample size of **L** samples, while the wearable device remains stationary. The collected sensor data[1:**L**], **X** configuration and the sample size of the sub-data (**a** samples) are input to the classifier algorithm running on the community sensing server. From our experiments, **L**=16,000 and **a**=800 values work very well with our algorithm. Standard deviation of inherent sensor noise (σ) is calculated from the sensor data using AD algorithm for each sub-data and the average standard deviation σ_{average} is obtained for hardware configuration X. The main intuition behind dividing the collected sensor data to a few sub-data items of equal length (**a**) is to apply AD algorithm multiple times and obtain a fair SD value for the inherent sensor noise (σ) for the hardware configuration X. If the existing standard deviation (σ_{current}) in the hardware configuration table significantly differs (for example, $\Delta > 0.0001$ for accelerometer) from newly calculated σ_{average}, *SeRaNDiP* updates the related hardware configuration sub-table with σ_{average}. Otherwise, σ_{current} remains unchanged in the hardware configuration sub-table.

Algorithm 1: Pseudo Code for Updating **X** Configuration in Configuration Table

```
Data: X configuration, Dataset[1:L] , a (size of Sub-dataset)
Result: Update $\sigma$ of X configuration ($\sigma$_X)
N= L/a ;
$\sigma_{\text{total}}$=0; Initialize Buffer[1:N] to Empty
for (i=1; i < N; i++) do
    Sub-dataset [:] = Dataset [(i-1)*a+1: i*a ]
    Apply AV algorithm on Sub-dataset and obtain $\sigma_i$ of white gaussian noise available
    Store at Buffer[i]
for (i=1; i < N; i++) do
    $\sigma_{\text{total}}$= $\sigma_{\text{total}}$ + Buffer[i]
$\sigma_{\text{average}}$ = $\sigma_{\text{total}}$/(N-1)
if abs($\sigma_{average}$ - $\sigma_{current}$) > then
    $\sigma$_X = $\sigma_{\text{average}}$
    $\sigma_{\text{average}}$ =0
    Buffer[1:N]=null
else
    $\sigma$_X = $\sigma_{\text{current}}$
    $\sigma_{\text{average}}$ =0
    Buffer[1:N]=null
```

6 RESULTS AND EVALUATION

In this section, we evaluate results in terms of accuracy, energy consumption, and latency of *SeRaNDiP* in comparison with several baselines. We collected in-field sensor data for this evaluation using the same four prototypes and Android app mentioned in Section 4.1. We implemented baselines with sensor parameters mentioned in Table 3 and *SeRaNDiP* with sensor parameters listed in Table 4 (3.13Hz, 0.78Hz and 0.125 Hz were the sampling rates of the accelerometer(both), BMP 388 barometer and MLP 3115A2 barometer respectively).

The following baselines were implemented on the ESP32 microcontroller in our prototypes - (a) State-of-the-art Distributed Differential Privacy (DDP) [33] implementation (For details, please refer to section 1), (b) Local Differential Privacy (LDP) implementation [77] and (c) Gamma-distributed noise generation based Distributed Differential Privacy (DDP) implementation [14]. For the rest of the paper, we will refer to the baselines (a), (b), and (c) as *DDP-BL, LDP-BL, and G-DDP-BL* respectively. The details of baselines (b) and (c) are explained below -
(b) Local Differential Privacy implementation [77] (LDP-BL) - Unlike a DDP setting, LDP is more relaxed and the users only send completely anonymized noisy sensor data to the community sensing server. Therefore, each wearable device involved in LDP has to generate enormous amount of noise to anonymize the sensor data before sending it to the community sensing server. This in turn leads to higher runtime and energy overheads at the user's wearable device. The overview of our baseline LDP implementation is shown in Figure 17(a). In LDP, Laplace distributed noise is added since it provides a stronger privacy guarantee (ϵ-DP) than the gaussian distributed noise ((ϵ,δ)-DP) [33]. The wearable device generates a Laplace-distributed noise and adds it to the raw sensor data. Since a relatively higher amount of noise is added to the raw sensor data for complete anonymization, the anonymized sensor data does not need to be encrypted prior to sending to the community sensing server. The configuration parameters of the LDP implementation are summarized in Table 6.
(c) Gamma-distributed noise generation based DDP implementation [14] (G-DDP-BL) - In [14], the authors used a gamma-distributed random noise generation technique to perturb the raw sensor data followed by a modulo addition based encryption technique. Unlike DDP-BL where gaussian random noise generation was employed, gamma noise generation requires relatively higher computation. Table 7 outlines the various parameters used in our implementation. The higher privacy guarantee provided by Laplace noise generated using gamma distribution [51] and the computationally-efficient XOR operation-based homomorphic encryption [23] makes it an attractive alternative to DDP-BL.

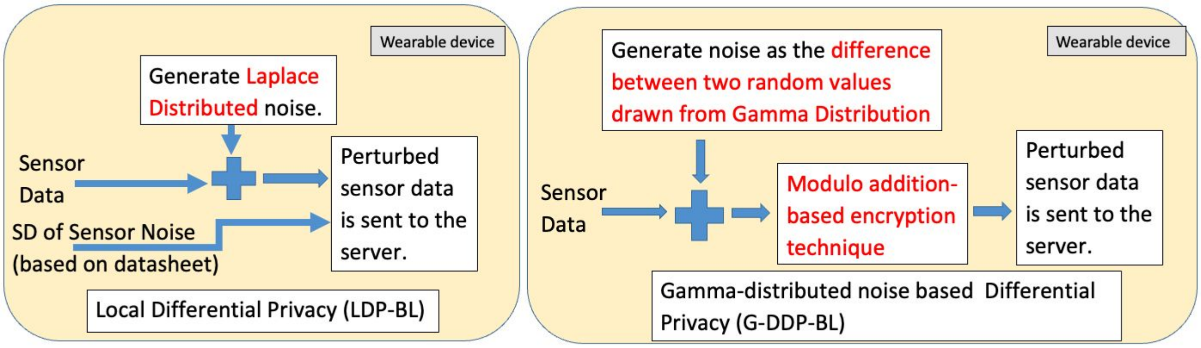

Wearable device
Generate Laplace Distributed noise.
Sensor Data
SD of Sensor Noise (based on datasheet)
Perturbed sensor data is sent to the server.
Local Differential Privacy (LDP-BL)
Wearable device
Generate noise as the difference between two random values drawn from Gamma Distribution
Sensor Data
Modulo addition-based encryption technique
Perturbed sensor data is sent to the server.
Gamma-distributed noise based Differential Privacy (G-DDP-BL)

Fig. 17. Overview of - (a) Local differential Privacy implementation and (b) Gamma-distributed noise generation based DDP implementation

6.1 Accuracy Evaluation

In this subsection, we discuss our user study (conducted in accordance with our university's IRB) in detail, along with the accuracy metrics on collected data from 10 college students.

6.1.1 Data Collection Protocol: Each participant had a prototype in their left leg pocket with the accelerometers' Y axis pointed downwards while standing. The four prototypes' available sensors are configured to *SeRaNDiP* settings and an Android phone installed with data logging controlling app was given to each participant to control data logging activity for the duration of data collection. Hence, this user trial generates 5 user data sets for each

Table 6. Configuration Parameters of the Local Differential Privacy (LDP-BL) implementation [77]

Parameter	Value
Noise Generation Mechanism	Laplace Distribution (L(μ,b))
Mean(μ)	0
Scale (b)	17888.5mg (for accelerometer) 990Pa (for barometer)

Table 7. Configuration parameters of the Gamma-distributed noise based Distributed Differential Privacy (G-DDP-BL) implementation [14]

Phase	Parameter	Value
Data Sanitization	Noise Generation Mechanism	Difference of two random values independently drawn from same Gamma Distribution (G_1(N,λ)-G_2(N,λ))
	Number of participants in a cluster(N)	27, 500, 000 (for accelerometer) 800,000 (for barometer)
	Number of clusters	1
	Scale of Gamma Distribution (λ)	17888.5 mg (for accelerometer) 990Pa (for barometer)
Data Encryption	Pseudo-Random Function(PRF)	XOR
	Key Stream(K_i')	Random number between 0 and 10,000,000
	Pairwise Key($K_{i,j}$)	6-digit number generated at initialization
	Slot key(r_k)	6-digit number generated by community sensing server for each slot

sensor model. Data were collected while the participants performing the following activities: sitting, standing, walking, jogging and climbing up stairs for 5 minutes each at different locations of the university premises for accelerometer-based physical activity classification. Stairs up (both indoors and outdoors), stairs down (both indoors and outdoors), elevator up, elevator down, escalator up and escalator down data for 5 minutes each were also collected at different locations in university for barometer-based vertical transport mode detection application.

6.1.2 Impact of SeRaNDiP on User-level Accuracy. Since *SeRaNDiP* changes the sensor configurations in wearables to generate the required noise for differential privacy, there is a possibility that application accuracy may be adversely affected. We thus investigated this with the physical activity classification application for the accelerometer sensor and vertical activity mode classification application for the barometer sensor. The physical activity classification application uses the 2D CNN model based open source single accelerometer based activity classifier [47] where input is raw accelerometer data along three axes. For the vertical transport mode detection, we used the decision tree based classifier which classifies escalator, elevator and stair modes using the XgBoost decision tree algorithm [66] where input is pressure difference. Table 8 details the settings used for both classifiers, based on their published configurations.

First, classification accuracy based on 10-user *SeRaNDiP* data is compared with previously published accuracy using 5-fold cross-validation. 86.4% accuracy was obtained for activity classification based on data sampled at a 3.13 Hz sampling rate from MPU-9250 and ADXL-345 accelerometers compared to 89.74% accuracy reported by publishers for WISDM dataset based on cell-phone accelerometer data sampled at 20 Hz [47]. This 3.34% accuracy reduction is due to the 6X lower sampling rate. Accuracy of pressure difference based vertical activity classification was 67.2% and 81.25% for barometers: BMP388 sampled at 0.78Hz and MLP 3115A2 sampled at

Table 8. Details of the neural network classifiers

Classifier	Type	Configurable Hyperparameter	Value
Physical Activity Classifier	2D CNN Model[47]	Size (width) of kernels used in the Conv Layers	2x2
		Activation function for Conv Layers	relu
		Number of kernels in 1st Conv layer	16
		Drop out rate after 1st Conv layer	0.1
		Number of kernels in 2nd Conv layer	32
		Units of first Dense Layer	64
		Activation of first Dense Layer	relu
		Drop out rate after 2nd Conv layer	0.2
		Units of second Dense Layer	6
		Activation of second Dense Layer	softmax
		Optimizer	adam
		Learning Rate	0.001
		Frame Size	6.4 seconds
		Hop Size	4.8 seconds
Vertical Activity Classifier	Xg Boost	Max Depth	12
		Sub Sample	0.33
		Objective	multi:softprob
		n_estimators	1000
		learning rate	0.001
		num_classes	6
		Frame Size	7.7 seconds
		Hop Size	5.1 seconds

0.125Hz respectively. This is on par with the 69% accuracy obtained in prior work [66] for data sampled from smartphone barometers at 15Hz and 25Hz rates in both indoors and outdoors spaces. The small 1.8% drop in accuracy for BMP388 can be acceptable at the system level.

Since pre-trained classifiers installed in mobile phones were mostly trained using data not belonging to a particular user, data from one participant was selected as test data, and data from the other 9 participants were used as training data in the next evaluation. Both classifiers were applied 10 times while changing the test data set with another participant's data until all participant's data were used as test data. We obtained 78% accuracy on average for physical activity classification and 64.42% and 52% accuracy on average for BMP388 and MLP 3115A2 barometers respectively for vertical activity classification. As the sensors used for *SeRaNDiP*, *DDP-BL, LDP-BL, and G-DDP-BL* implementations are configured to the same sampling rates, there is no difference in their accuracy (see Table 9). Besides, these accuracies are close to the accuracy numbers of the WISDM user trials which track the same activities (at a higher sensor sampling rate) [56].

6.1.3 *Impact on Server-level Accuracy due to the Configuration Changes.*

Next, we evaluate the accuracy at the server, with classifiers applied on aggregated data averages to predict the most popular activity for the community in DDP community sensing applications.

We evaluated the accuracy of the community sensing server for four different implementations - (1) *SeRaNDiP*, (2) DDP-BL, (3) LDP-BL and (4) G-DDP-BL using both accelerometers sampled at 3.13 Hz, BMP 388 barometer sampled at 0.78Hz and MLP 311512A2 barometer sampled at 0.125Hz. The aggregated data is averaged across 10 users, with the most popular activity among users selected. Given our limited user sample, these results just serve to illustrate server-side functionality. For each user, the first data point was selected randomly, and the average is computed across consecutive 85 samples (i.e. the number of floats that fit within the BLE buffer size (see Table 2)). The process was repeated until the creation of an average aggregated data set with 0.85 million data points (The number of data points is set sufficiently high for higher instances). In LDP-BL, the standard

Table 9. Classification Accuracy for Prototypes - Baselines Vs *SeRaNDiP*

Classifier	Model	Accuracy
Physical Activity Classification	MPU 9250 Accelerometer (***SeRaNDiP***)	84%
	MPU 9250 Accelerometer (**DDP-BL** [33])	84%
	MPU 9250 Accelerometer (**LDP-BL** [77])	84%
	MPU 9250 Accelerometer (**G-DDP-BL** [14])	84%
	ADXL 345 Accelerometer (***SeRaNDiP***)	72%
	ADXL 345 Accelerometer (**DDP-BL** [33])	72%
	ADXL 345 Accelerometer (**LDP-BL** [77])	72%
	ADXL 345 Accelerometer (**G-DDP-BL** [14])	72%
Vertical Activity Classification	BMP 388 Barometer (***SeRaNDiP***)	64.42%
	BMP 388 Barometer (**DDP-BL** [33])	64.42%
	BMP 388 Barometer (**LDP-BL** [77])	64.42%
	BMP 388 Barometer (**G-DDP-BL** [14])	64.42%
	MLP3115A2 Barometer (***SeRaNDiP***)	52%
	MLP3115A2 Barometer (**DDP-BL** [33])	52%
	MLP3115A2 Barometer (**LDP-BL** [77])	52%
	MLP3115A2 Barometer (**G-DDP-BL** [14])	52%

deviation value of the inherent sensor noise is required by the community sensing server to effectively denoise the perturbed/anonymized sensor data received. For each sensor, we obtain the standard deviation values through the noise estimates available from their datasheets. Using the noise estimates, the community sensing server denoises the anonymized sensor data obtained from 10 users and builds up a denoised aggregated data set that serves as the best approximation of the data from 10 users. *SeRaNDiP* and the three baselines were evaluated using the sensor data obtained from the 10 users in our study.

In all the baseline implementations, noise perturbation occurs at the user's wearable device. During the noise perturbation procedure, the sensor data was perturbed with white gaussian noise traces in DDP-BL, with Laplace noise traces in LDP-BL and with gamma-distributed noise traces in G-DDP-BL. The perturbed data is encrypted (excluding LDP-BL) and sent to the community sensing server. At the community sensing server, the received data from 10 users are aggregated and averaged. In SeRaNDiP's implementation, the white gaussian noise traces are added to the raw sensor data while the user is in a stationary position(e.g. sitting) - since *SeRaNDiP* is applied only on non-sedentary activities. For each of the implementations, noise traces were generated from the ESP32 device based on the differential privacy requirements.

At the community sensing server, *SeRaNDiP* utilizes an input frame size of 1 second (3 samples) and a hop size of 0.63 seconds for the accelerometer sensor data sampled at 3.13Hz. Based on Table 10 which summarizes accuracies on the accelerometer data, *SeRaNDiP* provides on par accuracy compared to the other baselines (1.66% higher than DDP-BL, 24.35% higher than LDP-BL and 0.67% higher than G-DDP-BL). The slight increase in accuracy against baselines reflects that the inherent sensor noise being used in *SeRaNDiP* results in less noisy sensor data than the baselines which perturb the data with externally generated noise. With the barometer sensor data, *SeRaNDiP* achieves 8.34% higher accuracy than DDP-BL, 9.03% higher accuracy than LDP-BL, and 8.56% higher accuracy than G-DDP-BL [14] (See Table 11). In a practical deployment of *SeRaNDiP*, we will require millions of participants at the server to deliver DDP guarantees - so that the inherent privacy noise can be averaged out from the high number of participants and thereby help the server-side system's accuracy to improve further.

In short, *SeRaNDiP* does not compromise the accuracy of the server application compared to the state-of-the-art.

Table 10. Server side Classification Accuracy for Accelerometer Data - *SeRaNDiP* Vs Baselines (Sample Rate = 3.13 Hz, LPF cut-off = 460Hz, Range =+/-2g)

Frame Size,	**SeRaNDiP**	**DDP-BL**	**LDP-BL**	**G-DDP-BL**
1s, 0.63s	49.96% (SD of perturbed noise) for sitting data:0.007g	48.39% (SD of perturbed noise: 0.007g)	25.61% Perturbed noise from Laplace(0,6.928g)	49.29% (Perturbed noise from G_1(27.5M,6.928g)) - G_2(27.5M,6.928g)))

Table 11. Server side Classification Accuracy for Barometer Data - *SeRaNDiP* Vs Baselines (Sensor Sampling Rate = 0.78 Hz and 0.125 Hz,Oversampling = x4, IIR filter =OFF, COMM Rate =1Hz)

Frame Size,	***SeRaNDiP***	**DDP-BL**	**LDP-BL**	**G-DDP-BL**
1s,0s	**25.59%**	**17.22%** **(SD of perturbed noise: 5.916Pa)**	**16.56%** **Perturbed noise from** **Laplace(0,990Pa)**	**17.03%** **(Perturbed noise from G_1(0.8M** **,990Pa)-G_2(0.8M,990Pa))**

6.2 Latency Evaluation

In this subsection, we extend the latency evaluations that we conducted in section 2.5 to dynamic settings across five users for *SeRaNDiP*. Figure 18 summarizes the average latency per data sample by accelerometer sensor-based data acquisition systems. Since the type of activity does not affect to internal hardware and software task pipeline of the sensor-based wearable system, the latency does not change significantly across the different dynamic activities as shown in Figure 18 when the same hardware sensor is used. Since hardware latency is caused by sensing changes with the sensor hardware change, different sensor systems provide different latency values even for the same activities done by the same person. As *SeRaNDiP* sends 85 data samples once from the prototypes and it takes around 1 minute and 25 seconds to fill up buffers with that data with a 1Hz sampling rate, we cannot explicitly measure latency and energy taken by vertical activities: elevator up/down and escalator up/down which normally take less than 1 minute to complete in one turn. So, we limit latency and energy measurements to the static setting for barometers.

From Figure 19, we see that the latency per sensor data sample achieved by *SeRaNDiP* is consistently lower when compared to baselines. In summary, our experiments with both the accelerometer sensors show that *SeRaNDiP* is 1.4 X, 1.3X and 1401X-1596X faster than DDP-BL, LDP-BL and G-DDP-BL respectively. Evaluations with both the barometer sensors also shows that *SeRaNDiP* is 1.4X-1.8X, 1.4X-1.7X and 1082X-3334X times faster than DDP-BL, LDP-BL and G-DDP-BL respectively. Since encryption is not required in LDP-BL, LDP-BL results in the lowest latency compared to other baselines. However, the latency from noise generation is much higher compared to encryption (see Figure 4) and thus, *SeRaNDiP* still achieves the fastest latency. Though G-DDP-BL [14] utilizes a modulo addition-based encryption scheme as an alternative to a compute-intensive homomorphic encryption scheme, the use of complex operations such as pseudo-random function (PRF) in homomorphic encryption results in higher compute latency. As the baselines and *SeRaNDiP* implementation need to access the hardware buffer thrice in the MLP3115A2 sensor, this led to the highest latency. *SeRaNDiP*'s latency savings are critical for future wearables since most wearable applications are real-time and interactive.

6.3 Energy Evaluation

In this subsection, we used FNIRSI FNB48 USB Meter Tester to measure energy consumption while doing physical and vertical activities (dynamic setting). Accuracy was cross-checked with the Monsoon power monitor and the absolute difference in measurement was 0.009 mAh per 1 mAh energy consumption.

As our prototypes are powered by a power bank with 5V output voltage, we placed FNB48 in between the power bank and prototype. To measure ESP32's hardware energy requirement without *SeRaNDiP*, we first configured

ESP32's CPU frequency to 80MHz and ran a script of idle loop for 5 minutes. Based on our experiment, the base platform ESP32 took around 4mAh for 5 minutes of active mode non-functional operation. Hence we deducted this amount from our measurements for 5 minutes of operation of *SeRaNDiP* and baseline implementations on ESP32-based wearable configured to 80Hz CPU frequency.

One participant wore the prototype in the left leg pocket and did physical and vertical activities for 5 minutes. Figure 18 summarizes the average energy consumption per data sample by accelerometer sensor-based data acquisition systems while one participant is doing physical activities per each sensor. Energy consumption per data sample varies around 0.96 uAh and it does not change significantly across different dynamic activities with the same hardware sensor model. This is expected because each reading of the sensor activates the sensor hardware which incurs a fixed energy consumption.

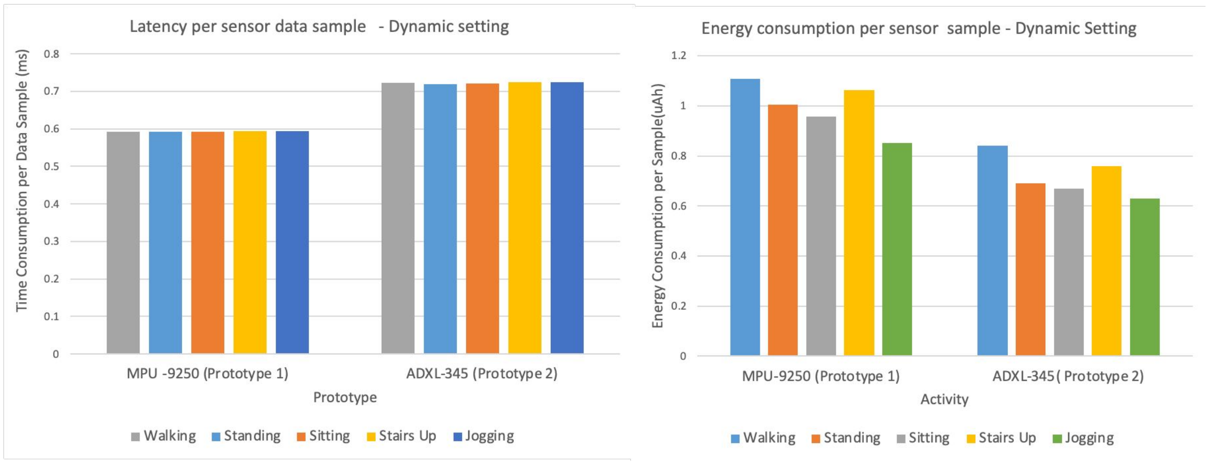


Fig. 18. Latency and energy consumption per accelerometer sensor data sample by *SeRaNDiP* at dynamic setting

Based on Figure 19, energy consumption per sensor data sample by *SeRaNDiP* is 1.4X-1.5X, 1.2X and 10.2X-11.2X lower for both accelerometer sensors compared to DDP-BL, LDP-BL, and G-DDP-BL respectively. Similarly, *SeRaNDiP* is 1.2X, 1.2X-1.8X and 7.4X-10.7X lower for both barometer sensors compared to DDP-BL, LDP-BL, and G-DDP-BL respectively. *SeRaNDiP*'s low energy consumption can be attributed to the elimination of the compute-intensive random noise generation procedure.

6.4 Is the Inherent Sensor Noise affected by Environmental Temperature Variations?

Since people live under different weather conditions, wearable devices also operate in varied environmental temperatures. The thermal noise in sensors is white gaussian [29] and arises from the vibration of charge carriers within the sensor. The thermal noise's power density is directly proportional to the temperature [1] and thus, the standard deviation of thermal noise is proportional to $\sqrt{temperature_{Kelvin}}$ [11]. Depending on the environmental conditions, the SD of the noise produced in sensors would be 0.072X lower at -12^oC compared to 30^oC. Thus, variations in environmental temperature have only a negligible impact on the total noise generated by the sensors. In *SeRANDiP*, we can also build configuration tables taking into account the SD of the inherent sensor noise under the lowest environment temperature in which humans can survive (when the amount of thermal noise generated is the lowest). By doing so, *SeRaNDiP* can provide privacy guarantees for higher temperatures.

To validate our hypothesis, we conducted controlled experiments to study the effect of temperature on the SD of white gaussian noise produced by both accelerometer and barometer sensors. For this experiment, we used the

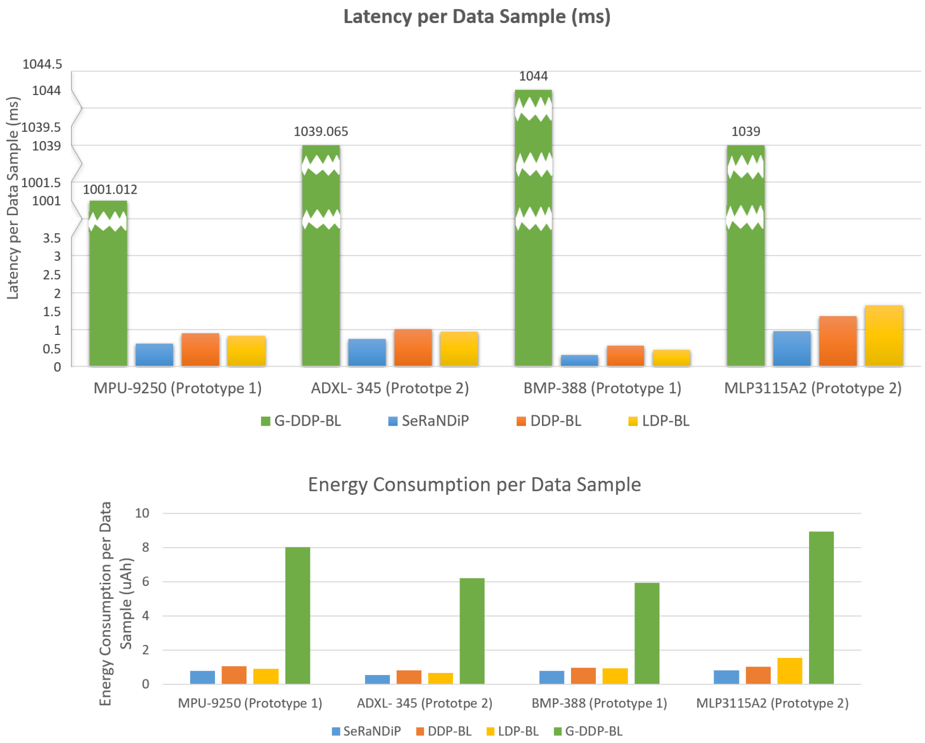


Fig. 19. Latency and energy consumption per sensor data sample by *SeRaNDiP* vs baselines

same 4 device prototypes used in our earlier experiments (see Figure 8). The configuration parameters of the sensors are outlined in Table 3. In addition to the prototypes, we also collected accelerometer sensor data at 25Hz from a Fitbit Sense smartwatch to study the effect of environmental temperature on commercial wearables where the sensors are securely packaged to remain unaffected by variations in environmental temperature.

6.4.1 Experimental Setup: We left our prototypes and the Fitbit Sense stationary on a flat table in an air-conditioned room with adjustable temperatures(between 16^oC to 30^oC) and collected data from each sensor for 2 minutes under four different temperatures: 16^oC, 21^oC, 25^oC and 30^oC. After configuring the air conditioner to a given temperature, we waited 15 minutes for the room to be cooled to the set temperature value. To study the effect of much lower temperatures on the sensor noise, a refrigerator was used. Since the motor in the refrigerator induces motion signals in the accelerometer sensor, the refrigerator was switched on for 30 minutes and switched off during the 2 minutes of data logging. With the refrigerator, we collected sensor readings under two different temperatures of 3^oC in the normal compartment and -12^oC in the freezer compartment. Before each measurement, the temperature was verified using a Xiaomi Mi Temperature and Humidity monitor.

6.4.2 Results: As shown in Figure 20, the SD of the gaussian noise produced by both the accelerometers do not change significantly with respect to temperature variations. Similarly, from Figure 21, we see that both the barometers' gaussian noise only change negligibly with respect to temperature variations. In addition, the same trend is also observed in the sensor data collected from the FitBit Sense smartwatch (see Figure 22). The results validate the robustness of *SeRaNDiP* to temperature variations in the environment.

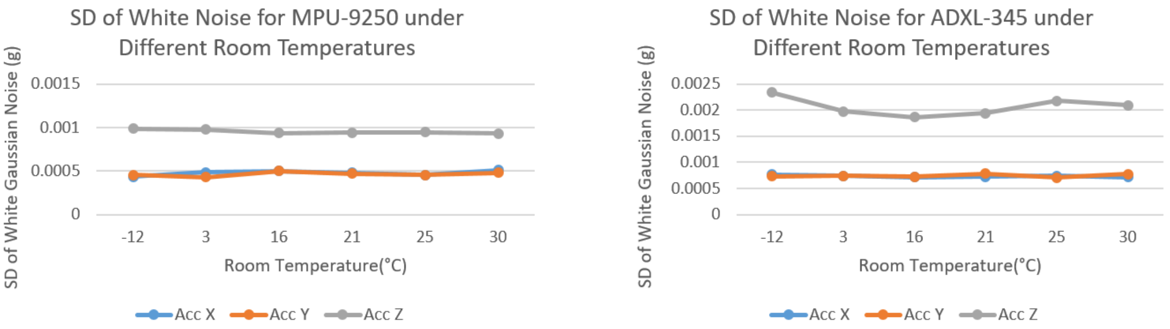


Fig. 20. SD of white noise of accelerometer sensors (sampled at 25 Hz) under different environment temperature conditions

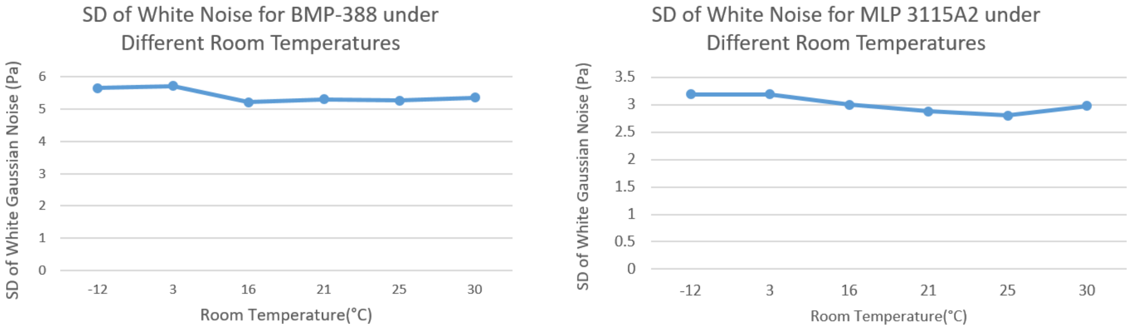


Fig. 21. SD of white noise of barometer sensors (BMP388 sampled at 0.78Hz and MLP 3115A2 at 1Hz) under different environment temperature conditions

7 DISCUSSION

In this section, we discuss how *SeRaNDiP* works with other wearable sensors, platforms, settings and differential privacy techniques.

7.1 Other Wearable Sensors

Here, we explore the possibility of expanding our solution to other wearable sensors like PhotoPlethysmoGram (PPG) sensor, microphone, ambient light sensor, Global Positioning System (GPS) sensor, magnetometer and gyroscope.

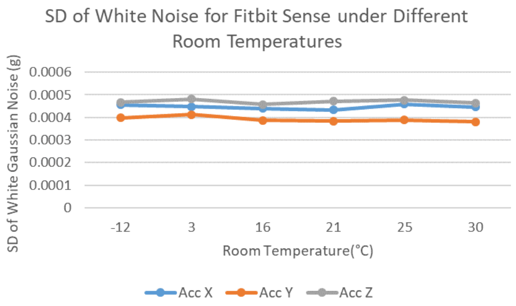


Fig. 22. SD of white noise of Fitbit accelerometer sensor (sampled at 25Hz) under different environment temperature conditions

7.1.1 PPG sensor: Though we sent raw sensor data to a server in our work, some DDP-based community sensing programs need features derived from raw sensors instead. For example, PPG-based derived features of heart rate, SpO2, and blood pressure are typically fed to community sensing programs from users [50]. Since PPG signals are substantially affected by human movements, PPG de-noising frameworks like TROIKA uses users' accelerometer signal to de-noise PPG signal to compute features [92]. Hence, controlling an accelerometer's parameters can introduce different amounts of noise to the heart rate, SpO2, and other derived features sent to the server with DDP. In relation to *SeRaNDiP*, it is required to build up a configuration table for the accelerometer concerning noise resulting in PPG-based features. Since motion artifacts are deducted from PPG signal, accelerometer noise and PPG noise shares a reciprocal relationship. Hence selecting an accelerometer setting that adds less amount of noise to the accelerometer signal increases the chance of a higher level of privacy for PPG-based features.

7.1.2 Microphone: According to Shamsabadi et al., though speaker anonymization is done on audio to remove voice print, it still needs to add DP to linguistic and prosodic attributes, since they still contain speaker information [81]. Though the Gaussian inherent noise is not so prominent in microphones, ambient noise in dB scale follows Gaussian distribution [70]. Hence we need to explore whether the white Gaussian noise component in ambient noise in air is sufficient to satisfy the DP noise requirement of linguistic and prosodic attributes for use in community sensing programs. Depending on the level of complexity, the microphone offers a variety of configurable options. For example, Microsemi ZL38063 audio processor [63] in Microsemi Development Kit for Amazon AVS [18] has settings such as gain, level tuning, and sampling frequency and Tizen OS used in Samsung smart watches for controlling microphone settings such as sampling rate (8kHz -48kHz), channel type (mono or stereo) and type of sample (8 or 16 bit) [74]. Further, simple off-the-shelf microphone modules [16] allow the configuration of just the clock signal. Hence, we see *SeRaNDiP* as being readily applicable to microphone-based community sensing applications like crowd sensing.

7.1.3 Ambient light sensor: According to our knowledge, there has not been literature regarding the inherent noise profile of the ambient light sensor. According to Hua et al., there is a significant white gaussian noise component associated with Visual Light Communication(VLC) [43] and hardware components used in VLC and ambient light sensor system are similar. So there is a possibility of having gaussian noise associated with ambient light sensor systems. Ambient light sensors also have a set of configurable settings such as amplifier gain and conversion time. Hence, there is a potential of applying our solution on ambient light sensor for community sensing program like determining UV light exposure.

7.1.4 GPS: . According to Niu et al., inherent noise in GPS sensors has a significant white Gaussian noise component [69]. Further, since GPS sensor has settings like sampling rate which can be configurable, *SeRaNDiP* is applicable for GPS sensors. But given the high power and timing overheads of the GPS sensor [67], the potential compute savings offered by *SeRaNDiP* may not be significant.

7.1.5 Magnetometer and gyroscope: Since the magnetometer and gyroscope have significant white Gaussian noise components in their inherent noise [30, 68] and the sensors have settings that are configurable [2], *SeRaNDiP* can be directly applied.

7.2 Applicability to Commercial Smartwatches

Since ESP32 achieves ultra-low power consumption through power-saving features including fine-resolution clock gating, multiple power modes and dynamic power scaling, it is widely used as a wearable development platform [25]. Hence we conducted our experiments on the ESP-32 development board. We also explored the possibility of expanding our solution to other wearable platforms like Fitbit and Samsung smartwatch. We obtained readings while placing the device on table. SD produced by accelerometer sensor in Fitbit Sense fitness tracker and Samsung Galaxy smartwatch decreases when sampling rate is increased, just like ESP32, as shown in Figure 23. This shows that *SeRaNDiP* can work readily on other wearable platforms.

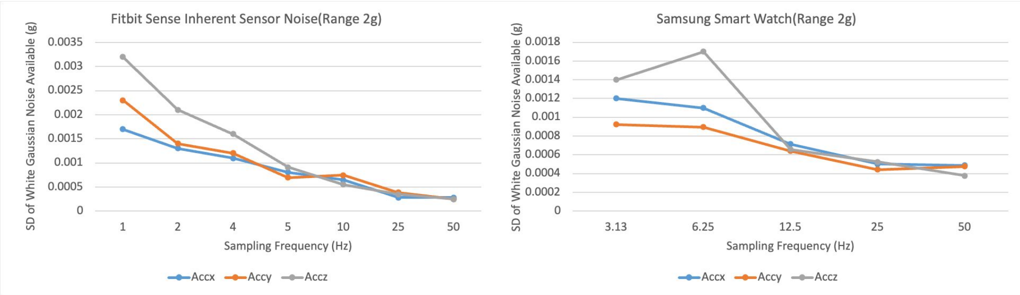


Fig. 23. Variations in the SD of white gaussian noise in the accelerometer sensor of Fitbit Sense and Samsung Galaxy smartwatch w.r.t. sampling rate

7.3 Applicability of Findings from SeRaNDiP's Limited User Trials to Large Population Sizes

While we showed earlier (Section 4.3) that differential privacy theory enables the derivation of the number of users necessary for ensuring differential privacy guarantees based on our experimentally measured sensor noise, our actual user trials are clearly limited in scale. Unfortunately, there are no large datasets of raw sensor data that we can leverage: The WISDM Lab data set involves 36 users [56], the MHEALTH Dataset data set tracked 10 users [22], the PhysioNet's labeled raw accelerometry data captured during walking, stair climbing and driving monitored 32 users [48], the PhysioNet's in-hospital physical activity has 58 users [76] and the LTMM database has 71 users [44].

We thus attempt to explore the practicality of *SeRaNDiP*'s scalability to a large population size through simulations with Matlab Simulink's MPU-9250 and ADXL345 accelerometer models. The accelerometer models produce x, y, z axis readings based on the configured parameter values obtain from their respective datasheets at 25^oC operating temperature. We simulated a population size of up to 100 million users, generating 16000 samples

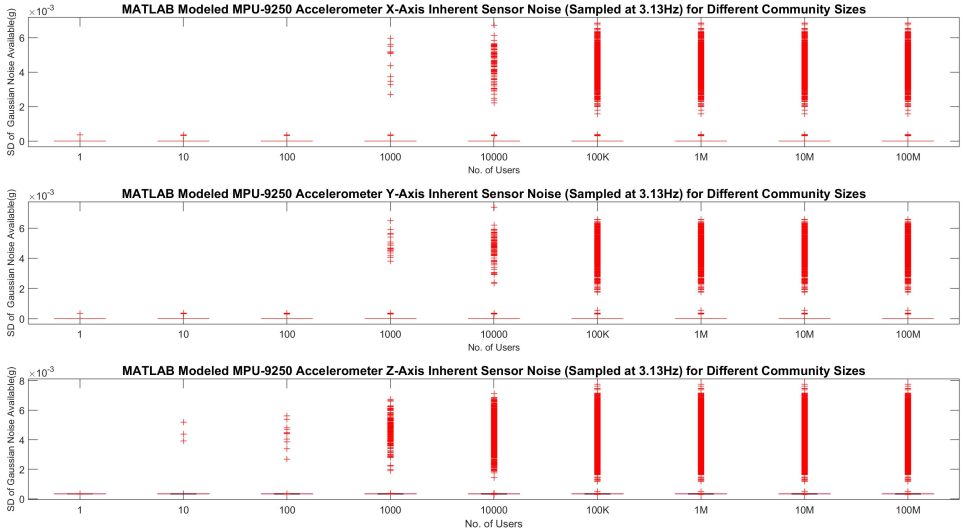


Fig. 24. Variations in the SD of white Gaussian noise in the Matlab modeled MPU-9250 accelerometer (sampled at 3.13 Hz, +/- 2g range) along sensor axes w.r.t. no. of users at environment temperature 25^oC

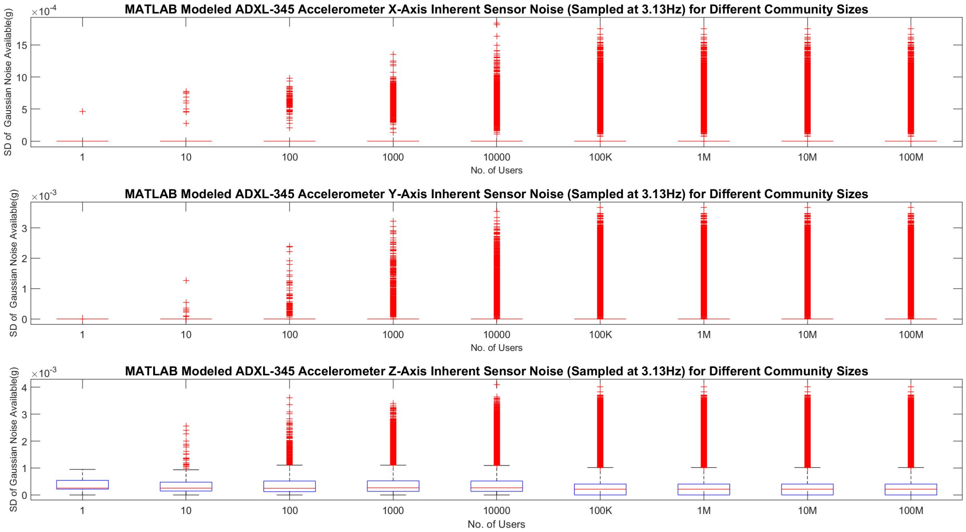


Fig. 25. Variations in the SD of white Gaussian noise in the Matlab modeled ADXL-345 accelerometer (sampled at 3.13 Hz, +/- 2g range) along sensor axes w.r.t. no. of users at environment temperature 25^oC

of data for both accelerometers using sensor configuration choices such as measurement range, resolution, temperature bias, bias instability, noise density etc. for each user.

The simulations show that the average SD of the inherent noise produced from randomly chosen population sizes of both accelerometers does not change significantly under different population sizes (see Figures 24 and 25). This implies that *SeRaNDiP*'s experimental characterization of the inherent noise of accelerometers on a small subset of 10 users can be used to populate the configuration tables apriori, before launching the app for large scale deployment.

8 CONCLUSION

In this paper, we presented *SeRaNDiP*, a framework that considers inherent sensor random noise for differential privacy preservation in wearable community sensing applications. It leverages sensors' inherent noise by changing sensor configurations at the software level without any hardware modifications. Extensive experimental results demonstrate *SeRaNDiP*'s ability to provide differential privacy to a variety of wearable sensors in different wearable platforms while delivering energy and latency savings. Hence *SeRaNDiP* can be readily applied to today's wearables, smartwatches and smartphones. We plan to release an open-source *SeRaNDiP* framework for ESP32 as a development framework for privacy-preserving community sensing applications.

ACKNOWLEDGMENTS

The authors would like to thank all the anonymous reviewers for their insightful reviews. This research was partially funded by Singapore National Research Foundation under NRF-RSS2016-005 and NRF2018NCR-NCR002-0001.

REFERENCES

[1] 2015. Chapter 8 - Noise. In *FinFET Modeling for IC Simulation and Design*, Yogesh Singh Chauhan, Darsen D. Lu, Sriramkumar Vanugopalan, Sourabh Khandelwal, Juan Pablo Duarte, Navid Paydavosi, Ai Niknejad, and Chenming Hu (Eds.). Academic Press, Oxford, 195–202. https://doi.org/10.1016/B978-0-12-420031-9.00008-7

[2] 2015. MPU-9250 Register Map and Descriptions Revision 1.6. https://invensense.tdk.com/wp-content/uploads/2015/02/RM-MPU-9250A-00-v1.6.pdf

[3] 2020. Accurate skin ear temperature sensing for wearable applications. https://www.melexis.com/en/documents/documentation/application-flyers/application-flyer-wearables

[4] 2020. Understanding the Accelerometer Noise Specification. *Understanding the accelerometer noise specification* (May 2020). https://wilcoxon.com/wp-content/uploads/2020/05/Understanding-the-accelerometer-noise-specification_TN33.pdf

[5] 2021. AI classifies people's emotions from the way they walk. https://venturebeat.com/2019/07/01/ai-classifies-peoples-emotions-from-the-way-they-walk/

[6] 2021. Blog 19 - Raw Data vs Processed Data: What It Means for Digital Health. https://verisense.net/blog/raw-sensor-data

[7] 2021. A list of real-world uses of differential privacy. https://desfontain.es/privacy/real-world-differential-privacy.html

[8] 2021. Number of active users of Apple watch in 2021. https://www.macrumors.com/2021/08/26/apple-watch-active-user-base-100-million/

[9] 2021. Stepping Science: Estimating Someone's Height from Their Walk. https://www.scientificamerican.com/article/bring-science-home-estimating-height-walk

[10] 2022. 17+ Google Maps Statistics to Survey in 2022. https://webtribunal.net/blog/google-map-statistics/#gref

[11] 2022. Electronic Instrumentation. https://ocw.tudelft.nl/wp-content/uploads/Slides_Electrical_Instrumentation_Lecture_3.pdf

[12] 2022. Number of active users of Fitbit from 2012 to 2020. https://www.statista.com/statistics/472600/fitbit-active-users/

[13] 2023. About Us- PressureNet. https://pressurenet.io/about/

[14] Gergely Ács and Claude Castelluccia. 2011. I Have a DREAM! (DiffeRentially privatE smArt Metering). In *Information Hiding*, Tomáš Filler, Tomáš Pevný, Scott Craver, and Andrew Ker (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 118–132. https://link.springer.com/chapter/10.1007/978-3-642-24178-9_9

[15] Adafruit. 2020. Adafruit/Adafruit_MPL3115A2_Library: Arduino Library for the MPL3115A2 sensors in the Adafruit Shop. https://github.com/adafruit/Adafruit_MPL3115A2_Library.git

[16] Adafruit. 2021. Adafruit I2S MEMS Microphone Breakout-SPH0645LM4H. https://www.adafruit.com/product/3421

[17] Yasmin Alkady, Fifi Farouk, and Rawya Rizk. 2019. Fully Homomorphic Encryption with AES in Cloud Computing Security. In *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2018*, Aboul Ella Hassanien, Mohamed F. Tolba, Khaled Shaalan, and Ahmad Taher Azar (Eds.). Springer International Publishing, Cham, 370–382. https://doi.org/10.1007/978-3-319-99010-1_34

[18] Amazon. 2021. Microsemi Development Kit for Amazon AVS. https://www.microsemi.com/product-directory/connected-home/4628-zlk38avs

[19] Apple. 2015. Differential Privacy. https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf

[20] Apple and Google. 2021. Exposure Notification Privacy-preserving Analytics (ENPA) White Paper. https://covid19-static.cdn-apple.com/applications/covid19/current/static/contact-tracing/pdf/ENPA_White_Paper.pdf

[21] Corneliu T.C. Arsene, Richard Hankins, and Hujun Yin. 2019. Deep Learning Models for Denoising ECG Signals. In *2019 27th European Signal Processing Conference (EUSIPCO)*. 1–5. https://doi.org/10.23919/EUSIPCO.2019.8902833

[22] Oresti Banos, Rafael García, Juan Holgado-Terriza, Miguel Damas, Hector Pomares, Ignacio Rojas, Alejandro Saez, and Claudia Villalonga. 2014. mHealthDroid: A Novel Framework for Agile Development of Mobile Health Applications, Vol. 8868. 91–98. https://doi.org/10.1007/978-3-319-13105-4_14

[23] Amina Bel Korchi and Nadia El Mrabet. 2019. A Practical Use Case of Homomorphic Encryption. In *2019 International Conference on Cyberworlds (CW)*. 328–335. https://doi.org/10.1109/CW.2019.00060

[24] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. 2011. Noiseless Database Privacy. In *Advances in Cryptology – ASIACRYPT 2011*, Dong Hoon Lee and Xiaoyun Wang (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 215–232. https://www.microsoft.com/en-us/research/wp-content/uploads/2011/12/noiseless-asiacrypt11_lncs.pdf

[25] blogofchem. 2021. The Internet of Things with ESP32. https://blogchem.com/2021/12/26/the-internet-of-things-with-esp32/

[26] Matthias Budde, Andrea Schankin, Julien Hoffmann, Marcel Danz, Till Riedel, and Michael Beigl. 2017. Participatory Sensing or Participatory Nonsense? Mitigating the Effect of Human Error on Data Quality in Citizen Science. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 39 (sep 2017), 23 pages. https://doi.org/10.1145/3131900

[27] Woo-Seok Choi, Matthew Tomei, Jose Rodrigo Sanchez Vicarte, Pavan Kumar Hanumolu, and Rakesh Kumar. 2018. Guaranteeing Local Differential Privacy on Ultra-Low-Power Systems. In *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. 561–574. https://doi.org/10.1109/ISCA.2018.00053

[28] Jiska Classen, Daniel Wegemer, Paul Patras, Tom Spink, and Matthias Hollick. 2018. Anatomy of a Vulnerable Fitness Tracking System: Dissecting the Fitbit Cloud, App, and Firmware. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 5 (mar 2018), 24 pages. https://doi.org/10.1145/3191737

[29] D.I. Crecraft and S. Gergely. 2002. 2 - Signals and signal processing. In *Analog Electronics*. Butterworth-Heinemann, Oxford, 13–71. https://doi.org/10.1016/B978-075065095-3/50002-7

[30] Katarína Draganová, F. Kmec, Josef Blazek, Dusan Praslicka, Jozef Hudak, and M. Laššák. 2014. Noise Analysis of Magnetic Sensors Using Allan Variance. *Acta Physica Polonica A* 126 (07 2014), 394–395. https://doi.org/10.12693/APhysPolA.126.394

[31] Yitao Duan. 2009. Privacy without Noise. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management* (Hong Kong, China) *(CIKM '09)*. Association for Computing Machinery, New York, NY, USA, 1517–1520. https://doi.org/10.1145/1645953.1646160

[32] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*, Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–12. https://doi.org/10.1007/11787006_1

[33] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407. https://doi.org/10.1561/0400000042

[34] Liyue Fan and Li Xiong. 2014. An Adaptive Approach to Real-Time Aggregate Monitoring With Differential Privacy. *IEEE Transactions on Knowledge and Data Engineering* 26, 9 (2014), 2094–2106. https://doi.org/10.1109/TKDE.2013.96

[35] Brandon Feenstra, Ashley Collier-Oxandale, Vasileios Papapostolou, David Cocker, and Andrea Polidori. 2020. The AirSensor open-source R-package and DataViewer web application for interpreting community data collected by low-cost sensor networks. *Environmental Modelling Software* 134 (2020), 104832. https://doi.org/10.1016/j.envsoft.2020.104832

[36] Jie Feng, Can Rong, Funing Sun, Diansheng Guo, and Yong Li. 2020. PMF: A Privacy-Preserving Human Mobility Prediction Framework via Federated Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 1, Article 10 (mar 2020), 21 pages. https://doi.org/10.1145/3381006

[37] Daniel Fenner, Benjamin Bechtel, Matthias Demuzere, Jonas Kittner, and Fred Meier. 2021. CROWDQC+—a quality-control for crowdsourced air-temperature observations enabling world-wide urban climate applications. *Frontiers in Environmental Science* 9 (2021). https://doi.org/10.3389/fenvs.2021.720747

[38] Luca Foschini, Jennifer Goldsack, Andrea Continella, and Wang Yu-Xiang. 2021. Contributed: When fitness data becomes research data, your privacy may be at risk. https://www.mobihealthnews.com/news/contributed-when-fitness-data-becomes-research-data-your-privacy-may-be-risk

[39] S. Gaamouri, Mounir Bousbia Salah, and Rachid Hamdi. 2019. Denoising ECG Signals by Using Extended Kalman Filter to Train Multi-Layer Perceptron Neural Network. *Automatic Control and Computer Sciences* 52 (2019), 528–538. https://link.springer.com/article/

10.3103/S0146411618060044

[40] Maren Goeckenjan, Esther Schiwek, and Pauline Wimberger. 2020. Continuous Body Temperature Monitoring to Improve the Diagnosis of Female Infertility. *Geburtshilfe und Frauenheilkunde* 80 (2020), 702 – 712. https://doi.org/10.1055/a-1191-7888

[41] Shai Halevi and Victor Shoup. 2020. Design and implementation of HElib: a homomorphic encryption library. *IACR Cryptol. ePrint Arch.* 2020 (2020), 1481. https://eprint.iacr.org/2020/1481

[42] Bo-Jhang Ho, Paul Martin, Prashanth Swaminathan, and Mani Srivastava. 2015. From pressure to path: Barometer-based vehicle tracking. *BuildSys'15 : proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Buildings : November 4-5, 2015, Seoul, South Korea. ACM Conference on Embedded Systems for Energy-Efficient Buildings (2nd : 2015)* (Nov 2015). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5831566/

[43] Luchi Hua, Yuan Zhuang, Longning Qi, Jun Yang, and Longxing Shi. 2018. Noise Analysis and Modeling in Visible Light Communication Using Allan Variance. *IEEE Access* PP (12 2018), 1–1. https://doi.org/10.1109/ACCESS.2018.2883737

[44] Espen Ihlen, Aner Weis, Jorunn Helbostad, and Jeffrey Hausdorff. 2015. The Discriminant Value of Phase-Dependent Local Dynamic Stability of Daily Life Walking in Older Adult Community-Dwelling Fallers and Nonfallers. *BioMed Research International* 2015 (09 2015). https://doi.org/10.1155/2015/402596

[45] Samira Jaafari. 2014. Adaptive Filtering for Heart Rate Signals. https://scholarworks.sjsu.edu/etd_theses/4420/

[46] Juan Jurado, Christine M Schubert. Kabban, and John Raquet. 2019. A regression-based methodology to improve estimation of inertial sensor errors using Allan variance data. *NAVIGATION* 66, 1 (2019), 251–263. https://doi.org/10.1002/navi.278 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/navi.278

[47] Laxmi Kant. 2019. Human Activity Recognition Using Accelerometer Data. https://github.com/laxmimerit/Human-Activity-Recognition-Using-Accelerometer-Data-and-CNN.git

[48] Marta Karas, Jiawei Bai, Marcin Strączkiewicz, Jaroslaw Harezlak, Nancy Glynn, Tamara Harris, Vadim Zipunnikov, Ciprian Crainiceanu, and Jacek Urbanek. 2019. Accelerometry Data in Health Research: Challenges and Opportunities: Review and Examples. *Statistics in Biosciences* 11 (01 2019). https://doi.org/10.1007/s12561-018-9227-2

[49] Harjeet Kaur and Rajni. 2016. ECG Signal Denoising with Savitzky-Golay Filter and Discrete Wavelet Transform (DWT). *international journal of engineering trends and technology* 36 (2016), 266–269. https://doi.org/10.14445/22315381/IJETT-V36P249

[50] Jong Wook Kim, Beakcheol Jang, and Hoon Yoo. 2018. Privacy-preserving aggregation of personal health data streams. *PLOS ONE* 13, 11 (11 2018), 1–15. https://doi.org/10.1371/journal.pone.0207639

[51] Samuel Kotz, Tomasz Kozubowski, and Krzysztof Podgorski. 2001. *The Laplace Distribution and Generalizations*. https://doi.org/10.1007/978-1-4612-0173-1_5

[52] Ryohei Kozu, Takahiro Kawamura, Shusaku Egami, Yuichi Sei, Yasuyuki Tahara, and Akihiko Ohsuga. 2017. User Participatory Construction of Open Hazard Data for Preventing Bicycle Accidents. In *Semantic Technology*, Zhe Wang, Anni-Yasmin Turhan, Kewen Wang, and Xiaowang Zhang (Eds.). Springer International Publishing, Cham, 289–303.

[53] Sian Lun Lau and Klaus David. 2010. Movement recognition using the accelerometer in smartphones. In *2010 Future Network Mobile Summit*. 1–9. https://ieeexplore.ieee.org/document/5722356

[54] Hyunsoo Lee, Soowon Kang, and Uichin Lee. 2022. Understanding Privacy Risks and Perceived Benefits in Open Dataset Collection for Mobile Affective Computing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 2, Article 61 (jul 2022), 26 pages. https://doi.org/10.1145/3534623

[55] Baozhong Liu and Jianbin Liu. 2019. Overview of Image Denoising Based on Deep Learning. *Journal of Physics: Conference Series* 1176 (mar 2019), 022010. https://doi.org/10.1088/1742-6596/1176/2/022010

[56] Jeffrey W. Lockhart, Gary M. Weiss, Jack C. Xue, Shaun T. Gallagher, Andrew B. Grosner, and Tony T. Pulickal. 2011. Design Considerations for the WISDM Smart Phone-Based Sensor Mining Architecture. *Proceedings of the Fifth International Workshop on Knowledge Discovery from Sensor Data (at KDD-11)*.

[57] Sachit Mahajan. 2022. Design and development of an open-source framework for citizen-centric environmental monitoring and data analysis. https://www.nature.com/articles/s41598-022-18700-z

[58] Mohammad Malekzadeh, Richard G. Clegg, and Hamed Haddadi. 2018. Replacement AutoEncoder: A Privacy-Preserving Algorithm for Sensory Data Analysis. *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)* (Apr 2018). https://doi.org/10.1109/iotdi.2018.00025

[59] Marin B. Marinov, Borislav Ganev, Nina Djermanova, and Tasho D. Tashev. 2019. Analysis of Sensors Noise Performance Using Allan Deviation. In *2019 IEEE XXVIII International Scientific Conference Electronics (ET)*. 1–4. https://doi.org/10.1109/ET.2019.8878552

[60] C. Marselli, D. Daudet, H. Amann, and F. Pellandini. 1998. Application of Kalman filtering to noise reduction on microsensor signal. *Proceedings of the Colloque interdisciplinaire en instrumentation* C2l, 443–450. https://core.ac.uk/download/pdf/20641186.pdf

[61] MartinL1. 2020. MARTINL1/BMP388_DEV: An Arduino compatible, non-blocking, I2C/SPI Library for the bosch BMP388 barometer. includes both interrupt and FIFO operation. https://github.com/MartinL1/BMP388_DEV.git

[62] Frank D. McSherry. 2009. Privacy Integrated Queries: An Extensible Platform for Privacy-Preserving Data Analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data* (Providence, Rhode Island, USA) *(SIGMOD '09)*. Association for

Computing Machinery, New York, NY, USA, 19–30. https://doi.org/10.1145/1559845.1559850

[63] Microsemi. 2021. Microphone Array ASR-Assist Audio Processor. https://www.microsemi.com/product-directory/connected-home/4432-zl38063

[64] Jairo H. Migueles, Alex V. Rowlands, Florian Huber, Séverine Sabia, and Vincent T. van Hees. 2019. GGIR: A Research Community–Driven Open Source R Package for Generating Physical Activity and Sleep Outcomes From Multi-Day Raw Accelerometer Data. *Journal for the Measurement of Physical Behaviour* 2, 3 (2019), 188 – 196. https://doi.org/10.1123/jmpb.2018-0063

[65] Ilya Mironov. 2017. Rényi Differential Privacy. In *CSF*. 263–275.

[66] Kartik Muralidharan, Azeem Khan, Archan Misra, Rajesh Balan, and Sharad Agarwal. 2014. Barometric phone sensors - More hype than hope! *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications, HotMobile 2014.* https://doi.org/10.1145/2565585.2565596

[67] Kalan Nawarathne, Rudi Ball, Francisco Pereira, Fang Zhao, Atika Rahardjo, Chris Zegras, Moshe Ben-Akiva, and Change Pereira. 2014. Trade-off Between Smartphone Battery Life and Stop Detection Accuracy. In *10th International Conference on Transport Survey Methods*. https://www.researchgate.net/publication/280134964_Trade-off_Between_Smartphone_Battery_Life_and_Stop_Detection_Accuracy

[68] K. Nirmal, A. G. Sreejith, Joice Mathew, Mayuresh Sarpotdar, Ambily Suresh, Ajin Prakash, Margarita Safonova, and Jayant Murthy. 2016. Noise modeling and analysis of an IMU-based attitude sensor: improvement of performance by filtering and sensor fusion. In *Advances in Optical and Mechanical Technologies for Telescopes and Instrumentation II*, Ramón Navarro and James H. Burge (Eds.), Vol. 9912. International Society for Optics and Photonics, SPIE, 2138 – 2147. https://doi.org/10.1117/12.2234255

[69] Xiaoji Niu, Qijin Chen, Quan Zhang, Hongping Zhang, J. Niu, Kejie Chen, C. Shi, and Jing nan Liu. 2013. Using Allan variance to analyze the error characteristics of GNSS positioning. *GPS Solutions* 18 (2013), 231–242. https://doi.org/10.1007/s10291-013-0324-x

[70] Sten Odenwald. 2020. Smartphone Sensors for Citizen Science Applications: Light and Sound. *Citizen Science: Theory and Practice* 13 (2020), 1–16. https://doi.org/10.5334/cstp.254

[71] Vibhor Rastogi and Suman Nath. 2010. Differentially Private Aggregation of Distributed Time-Series with Transformation and Encryption. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data* (Indianapolis, Indiana, USA) *(SIGMOD '10)*. Association for Computing Machinery, New York, NY, USA, 735–746. https://doi.org/10.1145/1807167.1807247

[72] JL Rausch, V Ganapathy, Y Fei, N Shendarkar, HM Hobby, KM Corley, and ME Johnson. 2003. Depressed patients have higher body temperature: 5-HT transporter long promoter region effects. *Neuropsychobiology* 47, 120–127. https://doi.org/10.1159/000070579

[73] Nazir Saleheen, Supriyo Chakraborty, Nasir Ali, Md Mahbubur Rahman, Syed Monowar Hossain, Rummana Bari, Eugene Buder, Mani Srivastava, and Santosh Kumar. 2016. mSieve: differential behavioral privacy in time series of mobile sensor data. *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (2016). https://doi.org/10.1145/2971648.2971753

[74] Samsung. 2021. Raw Audio Playback and Recording. https://docs.tizen.org/application/native/guides/multimedia/raw-audio/

[75] Kartik Sankaran, Minhui Zhu, Xiang Fa Guo, Akkihebbal L. Ananda, Mun Choon Chan, and Li-Shiuan Peh. 2014. Using Mobile Phone Barometer for Low-Power Transportation Context Detection. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems* (Memphis, Tennessee) *(SenSys '14)*. Association for Computing Machinery, New York, NY, USA, 191–205. https://doi.org/10.1145/2668332.2668343

[76] Severin Schricker, Nico Schmid, Moritz Schanz, Martin Kimmel, and Mark Dominik Alscher. 2020. In-hospital physical activity measured with a new Bosch Accelerometer Sensor System. https://physionet.org/content/hospital-activity-bosch/1.0/

[77] Yuichi Sei and Akihiko Ohsuga. 2020. Differentially Private Mobile Crowd Sensing Considering Sensing Errors. *Sensors* 20, 10 (2020). https://doi.org/10.3390/s20102785

[78] Ivan Seidel. 2015. Gaussian. https://github.com/ivanseidel/Gaussian

[79] Timothy J. Seppala. 2016. Use Feverprints to better understand your body temperature. https://www.engadget.com/2016-03-29-use-feverprints-to-better-understand-your-body-temperature.html

[80] M.R. Emami Shaker, A. Ghaffari, A. Maghsoodpour, and A. Khodayari. 2017. GPS/INS Integration for Vehicle Navigation based on INS Error Analysis in Kalman Filtering. *Automotive Science and Engineering* 7, 4 (2017). https://doi.org/10.22068/ijae.7.4.2562 arXiv:http://www.iust.ac.ir/ijae/article-1-437-en.pdf

[81] Ali Shahin Shamsabadi, Brij Mohan Lal Srivastava, Aurélien Bellet, Nathalie Vauquier, Emmanuel Vincent, Mohamed Maouche, Marc Tommasi, and Nicolas Papernot. 2023. Differentially private speaker anonymization. *Proceedings on Privacy Enhancing Technologies* 2023, 1 (Jan. 2023). https://hal.inria.fr/hal-03588932

[82] Elaine Shi, T.-H. Hubert Chan, Eleanor Gilbert Rieffel, Richard Chow, and Dawn Song. 2011. Privacy-Preserving Aggregation of Time-Series Data. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011.* The Internet Society. https://www.ndss-symposium.org/ndss2011/privacy-preserving-aggregation-of-time-series-data

[83] Gurdit Singh, Divya Bansal, Sanjeev Sofat, and Naveen Aggarwal. 2017. Smart patrolling: An efficient road surface monitoring using smartphone sensors and crowdsourcing. *Pervasive and Mobile Computing* 40 (2017), 71–88. https://doi.org/10.1016/j.pmcj.2017.06.002

[84] Linjuan Sun and Yuehui Jia. 2020. An improved PPG denoising methodology based on EEMD and wavelet threshold. In *2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Vol. 9. 467–471. https://doi.org/10.1109/

ITAIC49862.2020.9339069

[85] Ewdison Then. 2020. Samsung apps and services now at the center of a privacy controversy. https://www.slashgear.com/samsung-apps-and-services-now-at-the-center-of-a-privacy-controversy-09606491/

[86] Lev Velykoivanenko, Kavous Salehzadeh Niksirat, Noé Zufferey, Mathias Humbert, Kévin Huguenin, and Mauro Cherubini. 2022. Are Those Steps Worth Your Privacy? Fitness-Tracker Users' Perceptions of Privacy and Utility. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 4, Article 181 (dec 2022), 41 pages. https://doi.org/10.1145/3494960

[87] Israel Vidal, Franck Rousseau, and Javam Machado. 2019. Achieving Differential Privacy in Smart Home Scenarios. In *Anais do XXXIV Simpósio Brasileiro de Banco de Dados* (Fortaleza). SBC, Porto Alegre, RS, Brasil, 211–216. https://doi.org/10.5753/sbbd.2019.8825

[88] wikipedia. 2022. Homomorphic encryption. https://en.wikipedia.org/wiki/Homomorphic_encryption

[89] Wikipedia contributors. 2022. Allan variance — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Allan_variance&oldid=1115110583 [Online; accessed 7-November-2022].

[90] Wollewald. 2020. Wollewald/ADXL345_WE: Arduino Library for the ADXL345 accelerometer. I2C and SPI are implemented. https://github.com/wollewald/ADXL345_WE.git

[91] Yelvlab. 2018. Yelvlab/ESP32-arduino-MPU9250: BPI-bit MPU9250 Library. https://github.com/yelvlab/ESP32-Arduino-MPU9250.git

[92] Zhilin Zhang, Zhouyue Pi, and Benyuan Liu. 2015. TROIKA: A General Framework for Heart Rate Monitoring Using Wrist-Type Photoplethysmographic Signals During Intensive Physical Exercise. *IEEE Transactions on Biomedical Engineering* 62, 2 (2015), 522–531. https://doi.org/10.1109/TBME.2014.2359372

[93] Keyu Zhu, Pascal Van Hentenryck, and Ferdinando Fioretto. 2020. Bias and Variance of Post-processing in Differential Privacy. https://doi.org/10.48550/ARXIV.2010.04327