Cluster Density in crowdsourced Mobile Network Measurements

Sonja Tripkovic, Philipp Svoboda, Vaclav Raida, Markus Rupp
Institute of Telecommunications
Technische Universität Wien
Vienna, Austria
firstname.lastname@tuwien.ac.at

***Abstract*—The evaluation of mobile network performance is based on real-world measurement data. This data originates from different sources, such as drive-test, drone-measurements, and crowdsourced data. As measurements are not available at all locations, spatial interpolation is necessary to estimate spatial service coverage. Gaussian Process Regression (GPR) presents itself as a useful tool for performance metrics map reconstruction and for delivering prediction quality with it, allowing network operators to determine areas in which new measurements, e.g., drive-tests, will be useful. However, it comes at the cost of low scalability with growing data sets, expected with crowdsourced data. We aim to limit the computational effort in the GPR prediction resulting from updates in the data set, by measurement clustering and averaging while reducing the measurement and Global Positioning System (GPS) location noise. We investigate two scenarios with different measurement distributions and the influence of cluster identification, as required for real data measurements. Based on a desired error of the GPR prediction, we can determine the number of clusters required in the area of interest and the number of points needed inside each cluster for sufficient noise reduction.**

***Index Terms*—Performance map, GPR, signal strength, crowdsourcing, clustering, LTE, 5G.**

I. Introduction

Today, in 2020, users of mobile communication expect ubiquity in their internet access. Mobile network operators, therefore, have to provide and guarantee coverage at any location. The evaluation of network performance is based on the fusion of measurements from different data sources and forecasting performance metrics. As the collected data is not available at all locations, the prediction is required to estimate spatial service coverage. A rising number of smartphone subscriptions worldwide and an increase in average data volumes push the mobile traffic growth forward day by day. According to CISCO, mobile data traffic's annual run rate will grow almost seven-fold between 2017 and 2022. Additionally, the number of smart devices will grow two-fold in the same time range, reaching 9 billion in number [1]. This growing demand requires constant network performance improvement while pressuring mobile broadband operators to enhance their services continuously. Network operators can also use it to their advantage by utilizing end-user equipment for gathering crowdsourced measurements, thereby reducing the requirement of time- and cost-consuming drive-test measurements, as well as obtaining results at user locations. Fig. 1 schematically depicts possible user clusters, containing both stationary and mobile users. The benefits and challenges of crowdsourcing under the frame of mobile networks are discussed in detail in [2].

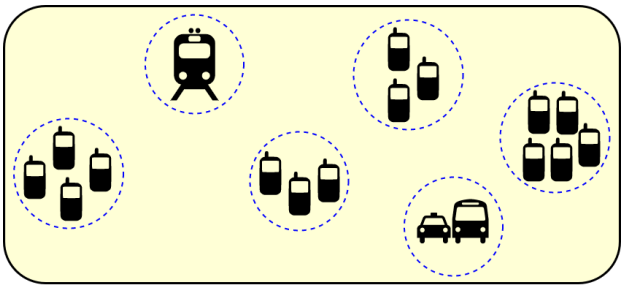

Fig. 1: Clusters of mobile users in crowdsourced scenario.

Our target is the prediction of mobile network performance on a city scale using crowdsourced measurements. We use the example of the city of Vienna, Austria. In addition to crowdsourcing, we aim to exploit the underlying spatial correlation of our data. In end-user devices, the uncertainties in location and the measurement noise level will be higher than in traditional operator-employed drive-test-collected measurements [3]. A regression model that accounts for the measurement noise and exploits the spatial properties is the GPR.

The question arises, which density of crowdsourced measurements and which density of crowdsourced measurement clusters are required to predict the network performance metrics accurately.To answer this question, we focus on Reference Signal Received Power (RSRP) as the measurement metric [4]. Nevertheless, the underlying model is extendable to other parameters, such as data-rate or service quality. Based on our simulated RSRP crowdsourced measurements, we estimate a spatial two-dimensional RSRP field using GPR.

The paper is organized as follows: Section II describes the state of the art methods for GPR complexity reduction and location uncertainty synthesis. Section III provides a theoretical introduction to the GPR method. Section IV-A and Section IV-B respectively present a simulation of equidistantly spaced and randomly spaced crowdsourced measurement clusters. In Section IV-C, we compare the GPR performance of RSRP map reconstruction using differently distributed measurement clusters, while Section IV-D introduces cluster

identification methods, which are needed when clusters are unknown. Section V compares GPR prediction performance for all three methods. Section VI summarizes our results and concludes the paper.

II. Related Work

When predicting network performance metrics, the GPR model offers the possibility of utilizing the data's underlying structure and correlation. In addition to the predictive mean, the GPR method also outputs the predictive error bars [5]. While GPR can be a powerful predictor with confidence intervals as important side information, it requires high accuracy in the training locations. Additionally, the computational complexity of GPR becomes becomes hard to grasp with growing training data sets.

With the high accuracy of 5G New Radio technology, continuous localization and user tracking are feasible. Authors in [6] present how accurate localization, together with signal strength, can reduce signaling overhead in mobile communications by applying a spatial regression method, e.g., GPR, for signal strength maps reconstruction from the collected measurements. In [7], authors propose exploiting spatial and temporal correlation for predicting average channel gain.

Recent publications [8] cover the reconstruction of environment maps using GPR or kriging from crowdsensed data, thereby not analyzing the positioning problem and the computational complexity of a continuously expanding data set, but rather only the reconstruction itself. While obtaining an accurate position is crucial for distance-based reconstruction methods, current measurement campaigns show certain limitations [9]. Previous works [10]–[12] in the field of data sampling show the challenges in obtaining precise measurements in reactive mobile cellular networks, as well as the fact that crowdsensed benchmarks are susceptible to time-of-day effects and change in topology [13], things we can only partially compensate [14].

GPR's main limitation is the inadequacy of the online update possibility. With each new training point, GPR needs to recompute the resource exhausting $M \times M$ [1] matrix inverses. Since its computational time depends on the size of the training data set, many authors propose different solutions that tackle GPR scalability from multiple angles. Proposed frameworks for GPR computational improvement are summarized in [15]. In [16], authors suggest GPR framework that can account for location uncertainty during learning/training and prediction/testing. Nevertheless, after performing experiments, this method did not provide us with sufficiently accurate performance. Therefore, we concentrate on different methods for clustering the training data, aiming to embrace existent location uncertainty while simultaneously reducing the GPR computational time.

III. Problem Description

We are interested in reconstructing a two-dimensional mobile network's signal strength map based on a given set of measurements. As the performance metric of the mobile network signal strength, we use the RSRP. We expect a squared exponential correlation of the RSRP values [3] in our system model in a specific distance range and choose a predictor that accounts for such a correlation between available measurements - GPR. The GPR prediction's main advantage is its property of providing the prediction quality with each of the predicted values, allowing us to determine areas in which additional measurements should be conducted. However, GPR's complexity scales cubically with the number of measurements. In addition to storing a large correlation matrix, we also have to compute the matrix inverse, making it hard to cope with massive data sets. To overcome the computational issue, we propose first clustering the measurements and then averaging the clustered measurements, thereby reducing the training data set N-fold, where N is the number of measurements available in each cluster. In addition to this, adding new measurements to existing clusters does not drive the complexity out of its boundaries, as the number of averaged training points remains the same unless we add new clusters. In Section III-A, we provide a short overview of the GPR predictor we employ, while in Section IV we introduce two simulated measurement distribution scenarios, which we use for the GPR training.

A. Intoduction to GPR-Based Reconstruction

In this section, we summarize the GPR framework and assumptions, under which we deploy it. For further reading on the subject of GPR, we refer the interested readers to [5].

Without loss of generality, we assume the underlying performance map's hyper-parameters do not vary depending on the channel conditions [3]. This assumption can easily be omitted by incorporating the GPR hyper-parameter estimation into the model. The main limitation of GPR when dealing with large data sets is the lack of an online update possibility, as its computational time depends on the size of the training data set. On the other hand, the GPR's superiority against other regression models is its ability to provide a distribution of the prediction value rather than just a single value [5]. This uncertainty can be exploited for determining the optimal location for the next measurement, thereby minimizing the prediction error of the entire map.

We define the training set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{M}$, consisting of exact training locations $\mathbf{x}_i$ and noisy observations y_i. Each location is specified with an S-dimensional real vector $\mathbf{x} \in \mathbb{R}^S$ (e.g., coordinates in space), and a collection of the train point vectors is summarized in matrix $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M)$. We denote the noisy measurements by real values y_i, consisting of a real-valued scalar function $f(\mathbf{x}_i)$ and a measurement noise n_i:

$$y_i = f(\mathbf{x}_i) + n_i. \tag{1}$$

We model $f(\mathbf{x})$ as a Gaussian Process (GP), fully parametrized by a mean function $m(\mathbf{x}) := \mathbb{E}\{f(\mathbf{x})\}$ and a covariance function $k(\mathbf{x}, \mathbf{x}') = \mathrm{cov}\{f(\mathbf{x}), f(\mathbf{x}')\}$. Random variables $f(\mathbf{x}_i)$ are jointly Gaussian. Combining them into

[1] M is the size of the GPR training data set

a random vector $\mathbf{f} := (f(\mathbf{x}_1), \cdots, f(\mathbf{x}_M))^T$, we write their joint probability density function as:

$$p(\mathbf{f}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{f}-\mathbf{m})^T\mathbf{K}^{-1}(\mathbf{f}-\mathbf{m})\right)}{\sqrt{(2\pi)^M \det(\mathbf{K})}}, \tag{2}$$

where

$$\mathbf{m} := \mathbb{E}\{\mathbf{f}\} = \left(m(\mathbf{x}_1)\cdots m(\mathbf{x}_M)\right)^T, \tag{3}$$

$$\mathbf{K}(\mathbf{X},\mathbf{X}) := \begin{pmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_M) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_M) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_M,\mathbf{x}_1) & k(\mathbf{x}_M,\mathbf{x}_2) & \cdots & k(\mathbf{x}_M,\mathbf{x}_M) \end{pmatrix}. \tag{4}$$

Our goal is to predict the function value $f(\mathbf{x}_*)$ (e.g. RSRP) at any given test location $\mathbf{x}_*$, or learn the function behind the model. The covariance function carries our assumptions about the function we wish to infer and defines a notion of data point similarity. Therefore, training points placed close to the test point are likely to be more informative regarding prediction at that point, compared to those training points further away from it. We employ *Squared Exponential (SE)* covariance function multiplied with a *constant* kernel:

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x}-\mathbf{x}'\|^2\right). \tag{5}$$

The hyper-parameter $\ell > 0$ of the SE kernel is the characteristic length scale, also denoted as decorrelation distance (DD). A small length scale value indicates that function changes quickly, while large values portray functions that change slowly. The variance $\sigma_f^2 \geq 0$ of the constant kernel accounts for the signal variance that scales the SE kernel.

We model n_i as a zero mean GP with variance σ_n^2, statistically independent of $f(\mathbf{x}_i)$ and n_j for $j \neq i$. Therefore, y_i from Eq. (1) is also defined as a GP with the following mean and covariance:

$$\mathbb{E}\{y_i\} = \mathbb{E}\{f(\mathbf{x}_i)\} = m(\mathbf{x}_i), \tag{6}$$

$$\begin{aligned}\mathrm{cov}\{y_i, y_j\} &= \mathrm{cov}\{f(\mathbf{x}_i), f(\mathbf{x}_j)\} + \mathrm{cov}\{n_i, n_j\} \\ &= k(\mathbf{x}_i,\mathbf{x}_j) + \sigma_n^2\delta_{i,j}.\end{aligned} \tag{7}$$

Using vector notation $\mathbf{f} := \left(f(\mathbf{x}_1)\cdots f(\mathbf{x}_M)\right)^T$ and $\mathbf{y} := (y_1 \cdots y_M)^T$, we rewrite Eq. (1) as $\mathbf{y} = \mathbf{f} + \mathbf{n}$ with

$$\mathbf{m} := \mathbb{E}\{\mathbf{y}\}, \tag{8}$$

$$\mathbf{\Sigma} := \mathrm{cov}\{\mathbf{y}\} = \mathbf{K} + \sigma_n^2\mathbf{I}, \tag{9}$$

where $\mathbf{I}$ is $M \times M$ identity matrix. Eq. (8) and Eq. (9) define the prior distribution $\mathbf{y} \sim \mathcal{N}(\mathbf{m}, \mathbf{\Sigma})$. Conditioning the joint prior distribution of the training outputs $\mathbf{y}$ and the test outputs $\mathbf{f}^*$, on the observations, we can incorporate the knowledge that the training data provides about the function and derive the posterior distribution.:

$$\begin{aligned}&\mathbf{f}^*|\mathbf{y};\mathbf{X},\mathbf{X}^* \sim \mathcal{N}\left(\mathbf{m}^*, \mathrm{cov}\left(\mathbf{f}^*\right)\right), \\ &\mathbf{m}^* \triangleq \mathbb{E}\left[\mathbf{f}^*|\mathbf{y};\mathbf{X},\mathbf{X}^*\right] = \mathbf{K}^{*T}[\mathbf{K}+\sigma_n^2\mathbf{I}]^{-1}\mathbf{y}, \\ &\mathbf{\Sigma}^* \triangleq \mathrm{cov}\left(\mathbf{f}^*\right) = \mathbf{K}^{**} - \mathbf{K}^{*T}[\mathbf{K}+\sigma_n^2\mathbf{I}]^{-1}\mathbf{K}^*\end{aligned} \tag{10}$$

where $\mathbf{K} = \mathbf{K}(\mathbf{X},\mathbf{X})$, $\mathbf{K}^* = \mathbf{K}(\mathbf{X},\mathbf{X}^*)$, $\mathbf{K}^{**} = \mathbf{K}(\mathbf{X}^*,\mathbf{X}^*)$, where $\mathbf{X}$ and $\mathbf{X}^*$ represent the collections of training and test points. From Eq. (10) we can sample points for any given test location $\mathbf{x}^*$, and not only can we get the expected value of the function at that location, but also its level of uncertainty. When the hyper-parameters of the covariance function are unknown in advance, we can learn them by maximizing the log marginal likelihood [5]. To compute the posterior in Eq. (10), we must store and invert a covariance matrix of dimension $M \times M$, where M is the size of the GPR training data set, making the GPR computation cumbersome with extensive measurement data sets, which we expect in crowdsourcing. To avoid this issue, we group the simulated measurements into clusters and average measurements inside each cluster, thereby reducing the training data set and the GPR computation time.

In the rest of the paper, we assume the GPR hyper-parameters are precisely known and investigate the influence of different training locations distributions on the GPR prediction of network parameters, e.g., RSRP.

IV. Clustering Measurements

The distribution of training points strongly influences the performance of GPR. In reality, mobile users' distribution differs across rural and urban areas [17]. Therefore, we expect a considerable density variation of crowdsourced measurements in rural and urban regions. Since people gather in points of interest, we still face a heterogeneous distribution of measurement points in specific areas - we denote these areas as *clusters of measurements*.

Consequently, it makes sense to model measurement points using a cluster process. By finding a representative point for each simulated cluster, in location and value, and feeding it as a single GPR input, we can significantly reduce the GPR complexity and even reduce the GPS induced location errors. In the case of GPR reconstruction, a good performance is achieved in the case of neighboring data-points within the DD. Therefore, we use equally distributed cluster centers, within the DD, as a benchmark scenario in our comparisons.

In Section IV-A and Section IV-B we introduce two simulation scenarios (S1 and S2) for generating clusters (Fig. 2). We average simulated data-points inside each cluster and use their representative averaged data-points, denoted as *quasi-measurements*, as the input to the GPR algorithm for S1 and S2 comparison in Section IV-C. In Section IV-D we discuss methods for cluster identification in scenarios where measurement clusters are unknown, which would be the case with actual crowdsourced measurements.

A. Simulation Scenario 1 (S1): Regularly Spaced Clusters

In S1, we aim to predict RSRP in a city with an orthogonal grid of streets such as Barcelona or New York City. Assuming the users are distributed along those streets, we expect the highest numbers to be gathered on the street crossings, giving rise to the street crossing clusters. However, users are not standing in formations on the crossings but are rather scattered around them in a random fashion. Fig. 2 left schematically

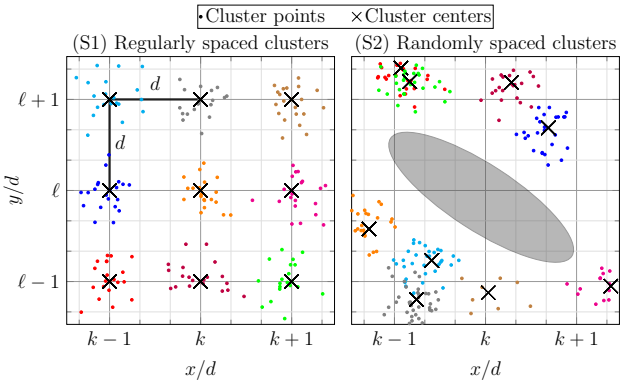

Fig. 2: Regularly spaced clusters in S1 (left); randomly spaced clusters in S2 with black hole area (right). The grid is plotted in multiples of cluster distance d of S1.

shows S1 of *regularly spaced* clusters, where each color represents different cluster. Notice that $\times$ represent street crossings placed on a regular grid and are equally spaced by distance d. In contrast, we represent the users as dots scattered around those cluster centers.

The *regularly spaced* cluster model is expected to work better in the GPR prediction framework than the *randomly spaced* cluster model shown in Fig. 2 right, which represents the second simulation scenario S2. Having random clusters across the area results in possibly large black holes with no measurements at hand, thereby reducing the GPR prediction performance in those vacant spots. As we expect *randomly spaced* clusters in actual crowdsourced data, we will use the ideal grid-like *regularly spaced* cluster model, achievable through drive-test or drone-measurements, as a benchmark in the GPR reconstruction performance comparison.

When generating *regularly spaced* clusters, we have three cluster parameters to consider:

- d ... Distance between cluster centers.
- σ ... Standard deviation of users around cluster centers.
- N ... Number of users inside each cluster (cluster points).

In S1, clusters are equidistantly spaced, meaning each cluster center is equally spaced from its nearest cluster center neighbor. To determine the influence of cluster parameters on the GPR prediction, we calculate the prediction Mean Square Error (MSE) between the simulated ground truth map and the map reconstructed using measurements simulated with a specific set of parameters d, σ and N.

Fig. 3 depicts the MSE over varying measurement noise level, represented by noise standard deviation σ_n. The results are averaged over 150 different map and measurement point realizations. We use a map of dimension 300×300 pixels[2] [2] in our simulation, with a DD (length scale in Eq. (5)) of 18 pixels. By changing the distance between cluster centers d (Fig. 3 left), we investigate the influence DD has on the GPR prediction. The MSE level is low when the cluster centers are separated less than one DD one from another, with small-scale variation around the mean MSE. The standard deviation around the mean MSE increases together with the measurement noise level. However, the significant jump in the mean MSE is first observed when the distance between cluster centers exceeds the DD of the underlying map. The reason for this lies in the GPR prediction nature being able to accurately predict only the values that lie inside one DD from its training points. As we perform GPR prediction using a single averaged point from each cluster, our GPR input data set will have *quasi-measurements*[3] spaced approximately[4] one DD from each other.

[2]Pixel represents an arbitrary length metric, e.g., m (meter).

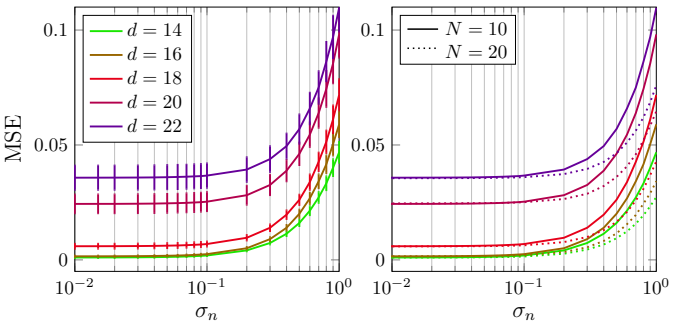

Fig. 3: MSE of the GPR prediction in S1. Bars are indicating the standard deviation around the mean MSE averaged over 150 different map and cluster points realisations (left). For a high measurement noise standard deviation σ_n the mean MSE reduction is obtained by using a larger number of points N inside each cluster (right).

The standard deviation σ around the cluster centers determines the cluster spread or cluster diameter. Such expansion of clusters[5] does not change the number of *quasi-measurements* used for GPR prediction. Therefore, its influence can be neglected on the GPR prediction performance and is left out from Fig. 3. However, increasing the number of points N inside each cluster can boost prediction performance and is particularly useful when dealing with higher noise levels. Assuming the measurement noise level is an unknown parameter in the GPR prediction, averaging over multiple cluster points reduces the noise influence on the prediction, as the noise variance reduces N times. Fig. 3 right illustrates the influence of the number N of users inside each cluster on the MSE level. However, for low measurement noise levels, the performance hits an error floor, indicating that additional measurements inside the existing clusters cannot produce an additional gain. The only further improvement is possible by reducing the cluster distance d, at the cost of increasing the number of total measurements and quasi-measurements and reduction of computation speed.

[3]Number of quasi-measurements is equal to the number of clusters N_c.

[4]The averaging is performed both in value and location inside a cluster.

[5]While keeping the number of cluster points N and the number of clusters N_c constant.

B. Simulation Scenario 2 (S2): Randomly Spaced Clusters

The previously discussed simulation scenario is only possible under a controlled measurement environment, such as drive-tests or drone measurements, which network providers can manipulate. However, we cannot ensure that the measurement clusters are omnipresent or equidistantly spaced when dealing with crowdsourced data. Additionally, we expect to have areas with highly concentrated measurements and those with only a few measurements at hand. However, in the areas where the measurement density is very high, using all available measurements as the GPR input would not improve the prediction performance but would significantly decrease its computational complexity. If we employ clustering and averaging to reduce the GPR input training data set, we would boost the GPR speed, almost without influencing the prediction quality. Under the lack of real crowdsourced data that would allow us to investigate such a scenario, we rely on the Thomas Cluster Process (TCP) to simulate *randomly spaced* clusters in a crowdsourced set-up. TCP relies on two Poisson process distributions for determining the number of clusters and the number of points inside each of those clusters [18]. We have three different parameters that influence the TCP process:

- The density of the first Poisson distribution, that determines the total number of clusters randomly placed in the area of interest.
- The density of the second Poisson distribution, which determines the number of cluster points inside each of the clusters[6].
- Cluster standard deviation σ, that determines the spread around randomly placed cluster centers.

Thereby, we can simulate both urban and rural scenarios by twitching these TCP parameters. Fig. 2 right shows one such scenario, where the clusters are randomly placed across the area of interest, leaving measurement vacant areas, we denote as *black holes*.

C. Comparison of S1 and S2

To ensure a fair comparison of *regularly spaced clusters* in S1 and *randomly spaced clusters* in S2, we first ensure that the density of available points and clusters across both scenarios is equal. We use S1 as a benchmark for S2, which we expect to encounter in real crowdsourced measurements. For the comparison setup, we use a 100×100 pixel^2 map with a DD of 15 pixels. We investigate the clustering influence in both scenarios and compare it with the case when the entire measurement data set (without averaging across the clusters) is fed into the GPR algorithm. The spread of the points around the cluster centers is the same in both scenarios ($\sigma = 2$). Using higher values of σ improves the GPR performance when all available points are used, but has negligible influence when averaging is performed. The influence of averaging is summarized in Table I, which shows the mean MSE value across different scenarios for three measurement noise levels. As we are interested in the GPR performance when averaging is applied, we show only values for $\sigma = 2$. We give the results for number of clusters $N_c = 19$ and $N_c = 26$, denoted as Case I and Case II respectively. As expected, we obtain a minimum MSE when utilizing all the points in S1, denoted as S1 all. When following the same approach in S2, we face a performance degradation as some clusters may be generated close together or even overlap, leaving many black holes in the area of interest. Table I shows that both S1 avg and S2 avg have very similar performance over different measurement noise levels. This similarity stems from the same number of quasi-measurements (clusters) as the GPR input in both cases.

[6]The number of cluster points varies across different clusters, in contrast to S1, where we had an equal number of points inside each cluster.

TABLE I: Mean MSE over 100 different map realisations

Case	σ_n	S1 avg	S2 avg	S1 all	S2 all
Case I: $N = 470$, $N_c = 19$	1	0.153	0.369	0.096	0.263
	0.1	0.121	0.315	0.005	0.075
	0.01	0.1198	0.3132	0.0002	0.0314
Case II: $N = 610$, $N_c = 26$	1	0.114	0.296	0.073	0.201
	0.1	0.081	0.231	0.003	0.047
	0.01	0.0801	0.2277	0.0001	0.0176

S1 = regularly spaced clusters, S2 = randomly spaced clusters, all = all measurements, avg = averaged cluster centers

Ultimately, we reduce the impact of the measurement noise upon the data by averaging over multiple data-points. This is most evident for the case of $\sigma_n > 0.1$. Notice from Table I, that averaging over clusters and using quasi-measurements as the GPR input results in an MSE increase from one order of magnitude at high measurement noise up to three orders of magnitude at low measurement noise. Table II shows the number of total measurements (for S1 all and S2 all) and quasi-measurements (for S1 avg and S2 avg) used as the GPR training set.

TABLE II: Number of training points M as the GPR input

	S1 avg	S2 avg	S1 all	S2 all
Case I	19	19	470	470
Case II	26	26	610	610

As the computation scales cubically $O(M^3)$ with the number of training points M, cluster averaging with much larger data sets results in a significant computational improvement. Therefore, we have to find a balance between the computational resources and the desired quality of prediction. Furthermore, the question arises: given an MSE target, what is the required density of points and clusters to achieve the set goal. We will address this question in Section V.

D. Cluster Identification

Going from a complete measurement set as the training data set, we moved on to clustering the complete measurement set to known simulated clusters, which we then averaged to find a single representative point (quasi-measurement) for each measurement group. Now, we analyze a more realistic scenario

where cluster identification is required in S2 before averaging, as the real clusters are unknown.

When dealing with clusters generated in the simulations, there is no need for cluster identification, as we already know which data point belongs to which cluster. However, in real data measurements, where we only have measurements taken in a specific area of interest, we have to determine the best way to group those measurements into clusters. We investigated four types of clustering algorithms, K-Means, GM, AC and OPTICS[7] [19]. For our TCP based scenario, K-Means clustering has given the best performance under S2 in terms of the GPR prediction. K-Means requires that the number of clusters be set beforehand. For efficient GPR map reconstruction, we must have cluster centers no further apart than one DD from their nearest neighbors. As the number of clusters corresponds to the number of quasi-measurements, we can use our pre-knowledge of the DD to determine the minimum number of clusters we need in order to perform the prediction. Therefore, in the real crowdsourced data scenario, we will have to choose the number of clusters that would still fulfill this DD condition.

V. Performance Comparison of the Complete, Clustered and Identified Training Sets

As the final comparison, we provide the performance of the GPR prediction plotted against the density of measurement points (number of total measurement points M per unit area) for the following cases of the GPR training data set, shown in Fig. 4:

- **(1)** All measurement points in S1,
- **(2)** All measurement points in S2,
- **(3)** Clustered and averaged points in S1, regularly spaced clusters,
- **(4)** Clustered and averaged points in S2, TCP clusters,
- **(5)** Clustered and averaged points in S2, TCP clusters, N_c[8] is constant
- **(6)** Clustered and averaged points in S2 - using K-Means method for cluster identification.

The increase in point density originates from increasing the number of clusters, for all cases apart from (5), where the density increase is the result of adding new data points to the existing clusters.

At known map dimensions and known DD, we can determine the minimum number of required clusters that would suffice for an accurate GPR prediction. Depending on the measurement noise level σ_n, we can choose the number of points inside each cluster, which would improve the performance when cluster averaging is applied. By fixing the parameters map dimension, DD and σ_n we can determine the point density required to achieve a mean MSE below 0.1.

For our comparison we choose a map of dimension 100×100 pixels^2, with DD $= 25$ and $\sigma_n = 0.01$. As DD is 25, we need at least one training point (quasi-measurement) per DD$\times$DD area for sufficient prediction quality, this would mean at least 16 training points/clusters in an area of 100×100 pixels^2 are required. This minimum number of clusters we define as a *DD condition* for satisfactory reconstruction performance. The prediction MSE in cases (1) and (2) actively improves with higher densities when the cluster number increases but stays relatively flat in the regions where the number of clusters is constant and the density increases. Implying that to improve prediction, we must introduce new clusters in the black hole areas, rather than increasing the density by adding new points in already existing clusters.

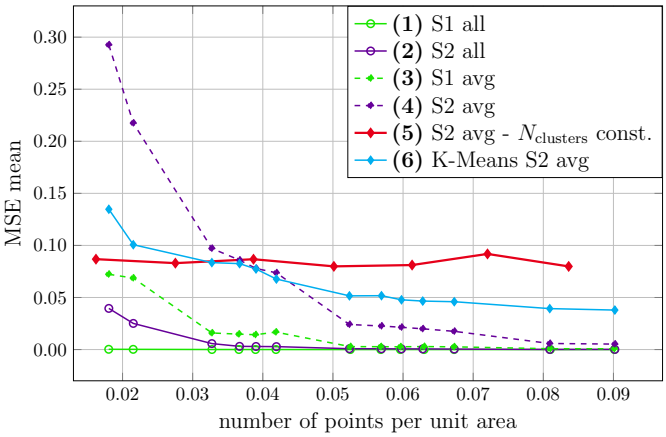

Fig. 4: Comparing the mean MSE over varying point density among six different cases.

Notice that case (1) and (2) outperform the rest, as they both use the total measurement data set, thereby containing more information about the area of interest. Clustering those measurements brings us to cases (3) and (4), respectively, where we observe a significant mean MSE increase when the density of points and correspondingly the number of clusters is low, but a comparable performance with an increasing number of clusters. Case (4) performs much worse at very low densities because black holes are more prominent at random than at regular clusters. Therefore, increasing the point density by introducing new clusters in black hole areas boosts its performance.

Case (5) shows no improvement over an increasing density, as the number of clusters is kept constant, but the number of points we average over inside each of them is increasing. Adding more measurements to the existing clusters is therefore only useful when dealing with higher measurement noise. Lastly, in case (6), we used the K-Means algorithm for identifying clusters previously generated using TCP (S2). For K-Means, we used the minimum required number of identified clusters, keeping it constant at 16 throughout different point densities. Notice that though the cases (5) and (6) both have a constant number of clusters, (6) outperforms (5) at higher densities. So even though we have the same black holes in both cases, K-Means manages to space its identified clusters in a way that boosts the GPR performance, resulting in lower

[7]Gaussian Mixture (GM), Agglomerative Clustering (AC), Ordering Points To Identify the Clustering Structure (OPTICS)

[8]Number of clusters.

MSE at higher point densities.

At point densities between 0.032 and 0.042 samples per unit area we are clustering our measurement data set into 16 clusters, which is the minimum required in this specific set up to achieve a mean MSE below 0.1. Increasing the density by introducing new clusters would further improve the prediction accuracy and result in a higher computational cost. Using fewer clusters than required per DD condition, the prediction performance becomes inadequate, with an MSE over 0.1.

The density of 0.032 indicates 320 points in an area of 100×100 pixels^2. Clustering those measurements to $N_c = 16$ results in $N = 20$ measurement points per cluster. At 0.042 point density, we have 420 data-points and $N = 26.25$ points per cluster, boosting the performance due to a higher measurement noise reduction.

Assuming the DD in Vienna is in the order of 25m, the signal variance σ_f is of the order ~ 1, as in our simulations, and we restrict our area of interest to Vienna's first district of 3km^2, we would require $4\,800$ clusters to achieve an MSE of below 0.1. If we further assume the measurement noise standard deviation σ_n lies in the range from 0.1 to 1, we would require between 10 and 20 measurements inside each cluster, resulting in $48\,000$ to $96\,000$ total crowdsourced measurements.

VI. Conclusions

Based on simulations, we investigated how different measurement distributions with and without using clustering and averaging influence the GPR prediction quality. In two simulated scenarios, S1 and S2, we have concluded that random cluster distribution in S2 results in a higher MSE of the GPR prediction due to the presence of larger measurement vacant areas. Through measurement clustering, we reduce the training set size and keep the GPR complexity constant even when adding new measurements to the existing clusters in both S1 and S2, while adding new clusters increases the prediction quality and GPR computational complexity simultaneously. By determining a target MSE, the number of clusters required in an area of interest for efficient GPR prediction is defined. Based on the present measurement noise level, the required number of points inside each cluster that would reduce both the measurement and the GPS location noise can be derived. Number of clusters N_c in the area of interest together with the number of points inside each of the clusters N, determines the required point density $N \times N_c$ per area of interest.

Through investigating different cluster identifications methods required for real-world crowdsourced measurements, we found K-Means to be best suited with regards to the GPR prediction quality in our simulations. Possible further work would be employing clustering and averaging with real-world crowdsourced data as a quality check of our simulation results.

References

[1] CISCO. (Mar. 2020). Cisco Annual Internet Report (2018–2023), [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf.

[2] C. Midoglu and P. Svoboda, "Opportunities and challenges of using crowdsourced measurements for mobile network benchmarking a case study on RTR open data," in *2016 SAI Computing Conference (SAI)*, 2016, pp. 996–1005.

[3] R. Di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson, and H. Wymeersch, "Location-aware communications for 5G networks: How location information can improve scalability, latency, and robustness of 5G," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 102–112, 2014.

[4] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 36.331, Apr. 2017, Version 14.2.2. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2440.

[5] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, 3. MIT press Cambridge, MA, 2006, vol. 2.

[6] R. Di Taranto, S. Muppirisetty, R. Raulefs, D. Slock, T. Svensson, and H. Wymeersch, "Location-aware communications for 5G networks: How location information can improve scalability, latency, and robustness of 5G," *IEEE Signal Processing Magazine*, vol. 31, pp. 102–112, 2014.

[7] Q. Liao, S. Valentin, and S. Stanczak, "Channel gain prediction in wireless networks based on spatial-Temporal correlation," *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, vol. 2015-August, pp. 400–404, 2015.

[8] Z. Han, J. Liao, Q. Qi, H. Sun, and J. Wang, "Radio Environment Map Construction by Kriging Algorithm Based on Mobile Crowd Sensing," *Wireless Commun. Mobile Comput.*, vol. 2019, pp. 1–12, Feb. 2019.

[9] V. Platzgummer, V. Raida, G. Krainz, P. Svoboda, M. Lerch, and M. Rupp, "UAV-Based Coverage Measurement Method for 5G," in *IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, IEEE, Ed., 2019, ISBN: 978-1-7281-1220-6. [Online]. Available: https://publik.tuwien.ac.at/files/publik_282558.pdf.

[10] V. Raida, P. Svoboda, M. Kruschke, and M. Rupp, "Constant Rate Ultra Short Probing (CRUSP): Measurements in Live LTE Networks," in *IEEE International Conference on Communications (ICC 2019)*, IEEE, Ed., 2019.

[11] W. Hofer, P. Svoboda, W. Kastner, V. Raida, and M. Rupp, "Open Monitoring Platform for Mobile Broadband," in *IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, IEEE, Ed., 2020.

[12] M. Laner, P. Svoboda, and M. Rupp, "Parsimonious Fitting of Long-Range Dependent Network Traffic Using ARMA Models," *IEEE Communications Letters*, vol. 17, no. 12, pp. 2368–2371, 2013. DOI: 10.1109/LCOMM.2013.102613.131853.

[13] V. Raida, P. Svoboda, M. Lerch, and M. Rupp, "Crowdsensed Performance Benchmarking of Mobile Networks," *IEEE Access*, vol. 7, pp. 154 899–154 911, 2019.

[14] L. Eller, P. Svoboda, and M. Rupp, "Semi-Supervised Detection of Tariff Limits in LTE Network Benchmarks," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, IEEE, Ed., 2020, ISBN: 978-1-7281-5207-3. [Online]. Available: https://publik.tuwien.ac.at/files/publik_289619.pdf.

[15] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When Gaussian Process Meets Big Data: A Review of Scalable GPs," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–19, 2020.

[16] L. S. Muppirisetty, T. Svensson, and H. Wymeersch, "Spatial wireless channel prediction under location uncertainty," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1031–1044, 2015.

[17] M. W. Fong, "Digital divide between urban and rural regions in China," *The Electronic Journal of Information Systems in Developing Countries*, vol. 36, no. 1, pp. 1–12, 2009.

[18] A. Baddeley, E. Rubak, and R. Turner, *Spatial Point Patterns: Methodology and Applications with R*. London: Chapman and Hall/CRC Press, 2015, p. 461. [Online]. Available: http://www.crcpress.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200/.

[19] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*, 2013.