# Dynamic Clustering for GNSS Positioning with Multiple Receivers

Greiff, Marcus; Di Cairano, Stefano; Berntorp, Karl

TR2022-094 August 06, 2022

**Abstract**

We consider the problem of jointly estimating the states of multiple global navigation satellite system (GNSS) receivers modeled with shared biases. In particular, we explore how to best assign these receivers to disjoint sets, so as to retain computational feasibility in the resulting filters. We propose a genetic algorithm that dynamically assigns agents to clusters subject to constraints on the maximum number of states in the clusters. Several numerical examples illustrate the flexibility of the approach, and the choice of genetic operations in the clustering algorithm is motivated by their effect on the algorithm's expected convergence rate. Numerical experiments with a GNSS-inspired problem demonstrates that the proposed clustering can yield a substantial improvements in the mean- square error compared to a random cluster assignment.

*International Conference on Information Fusion (FUSION) 2022*

# Dynamic Clustering for GNSS Positioning with Multiple Receivers

Marcus Greiff[1], Stefano Di Cairano[1], and Karl Berntorp[1]

*Abstract*— **We consider the problem of jointly estimating the states of multiple global navigation satellite system (GNSS) receivers modeled with shared biases. In particular, we explore how to best assign these receivers to disjoint sets, so as to retain computational feasibility in the resulting filters. We propose a genetic algorithm that dynamically assigns agents to clusters subject to constraints on the maximum number of states in the clusters. Several numerical examples illustrate the flexibility of the approach, and the choice of genetic operations in the clustering algorithm is motivated by their effect on the algorithm's expected convergence rate. Numerical experiments with a GNSS-inspired problem demonstrates that the proposed clustering can yield a substantial improvements in the mean-square error compared to a random cluster assignment.**

## I. INTRODUCTION

Global navigation satellite system (GNSS) positioning is a ubiquitous tool in modern society, and has been the subject of intense research since the 1950's [1]. The state of the art precise point positioning (PPP) approaches utilize sophisticated models of the underlying physics to eliminate various biases from the estimation problem [2]. These methods often involve a first-order expansion of the measurement equation, with subsequent de-correlation and integer search methods to approximately solve an NP-hard mixed-integer least-squares (MILS) problem in a non-linear Kalman filter (KF) setting (see, e.g., [3]–[7]). With simpler integer fixation schemes, such as bootstrapping [8], the computational complexity of these filters[2] scales with $O(N^3)$ in the number of estimated states, $N$. Several other approaches have been developed, including those based on multiple model (MM) Kalman filtering in [9], [10], which closely relates to the particle-filtering methods employed in [11]. Such algorithms improve the positioning performance at the cost of an increased computational complexity in $N$, and are therefore often implemented using simpler estimation models.

The above approaches all consider the problem of estimating the states of a single receiver (or agent). However, if we consider a set of agents, the inter-agent measurement noise will be correlated if the measurements are processed by difference schemes using the same base station (especially if considering double difference operations, see, e.g., [3]). In addition, the biases for the different agents are similar if the receivers are in close geographical proximity [7]. Hence, multiple agents can benefit from sharing information, but this comes at a cost of increasing the size of the estimation problem. Even with the methods in [6], [7], the joint estimation of multiple agents quickly becomes computationally infeasible as the number of agents increase, due to how the computation scales with the number of states and measurements. Therefore, the question addressed in this work is how to best form clusters of agents to minimize a desired performance objective given constraints on the computational resources.

The problem of finding an optimal clustering is closely related to the KF sensor selection (KFSS) problem [12]–[16]. In KFSS, a subset of the available measurements is to be used in minimizing a cost of the estimate mean-square error (MSE). Even in the linear Gaussian setting, this problem is NP-hard and generally not submodular [14, Examples 1 and 2]. Consequently, it is often addressed with approximations and convex relaxations [16, Section III.A], which come with no performance guarantees. An appealing alternative is to employ greedy selection algorithms [16, Section III.B], which tend to work very well in practice, and can be shown to converge to an optimal solution when the sensors are separable [14]. Indeed, recent work has shown the MSE in the KFSS to be approximately supermodular [15], which can be used to derive performance guarantees for more general problems. This motivates the study of algorithms inspired by greedy selection for the clustering problem.

Unlike the KFSS, we do not seek *a single subset* of all available measurements. Rather, the measurements are clustered into *multiple disjoint subsets*. A change in clustering implies the exclusion of measurements as in [14], [16], since the estimation of certain states cannot be done using certain measurements. In addition, a change in clustering also modifies the state spaces in the clusters, as the local states differ among the agents in GNSS positioning [7]. Here, the question is not how to pick a subset of measurements to improve estimation performance with respect to a specific state-vector, but how to choose a clustering such that subsets of measurements improves estimation performance with respect to the states in the resulting clusters.

### A. Notation

Vectors are denoted by $\boldsymbol{x} \in \mathbb{R}^n$ with $x_i$ denoting the $i^{\text{th}}$ element of $\boldsymbol{x}$. The concatenation of two vectors $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{y} \in \mathbb{R}^m$ is denoted $(\boldsymbol{x}; \boldsymbol{y}) = (\boldsymbol{x}^\top, \boldsymbol{y}^\top)^\top \in \mathbb{R}^{n+m}$. Matrices are indicated in bold as $\boldsymbol{X}$, and the element on row $i$ and column $j$ of $\boldsymbol{X}$ is written $[\boldsymbol{X}]_{ij}$. The notation $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates that $\boldsymbol{x}$ is Gaussian distributed with mean $\hat{\boldsymbol{x}}$ and covariance $\boldsymbol{P}$, and $\boldsymbol{x} \sim \mathcal{U}(I)$ indicates that the $\boldsymbol{x}$ is uniformly distributed over $I$. The notation $\hat{\boldsymbol{x}}_{k|k}$ refers to the estimate of $\boldsymbol{x}$ at time step $k$ given the set of measurements $\boldsymbol{y}_{0:k} \triangleq$

[1] Mitsubishi Electric Research Labs (MERL), 02139 Cambridge, MA, USA. Email: {greiff,dicairano,berntorp}@merl.com

[2]The exponent depends on how the nonlinearities are handled. If using $\sigma$-point methods, it is cubic in the state dimension, but it can be sub-cubic for simpler EKFs. The computational complexity is also cubic in the number of measurements, which may dominate the complexity for certain problems.

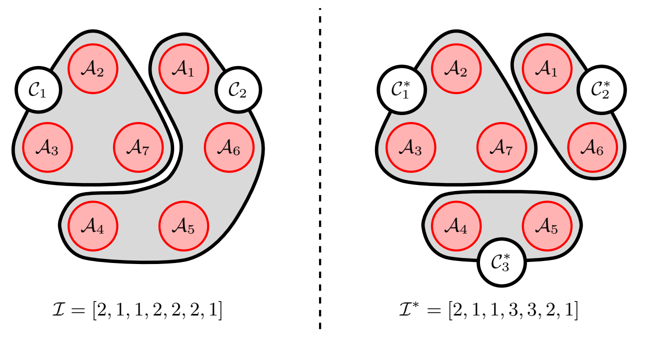

Fig. 1. Two ways of clustering a set of $n = 7$ agents into $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2\}$ where $m = 2$ (left) and $\mathcal{C}^* = \{\mathcal{C}_1^*, \mathcal{C}_2^*, \mathcal{C}_3^*\}$ with $m = 3$ (right), and a concise representation of by integer vectors, with $\mathcal{I}$ and $\mathcal{I}^*$, respectively. Agents exchange information within the clusters, and the clustering changes both the state-space in the clusters and the measurements used by each agent.

$\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_k\}$, and $\hat{\boldsymbol{x}}_{k|k-1}$ denotes the one-step prediction of $\hat{\boldsymbol{x}}_{k-1|k-1}$. Finally, $\otimes$ denotes the Kronecker product where $\boldsymbol{M} = (\boldsymbol{A} \otimes \boldsymbol{B})_{ij}$ is block structured with $\boldsymbol{M}_{ij} = [\boldsymbol{A}]_{ij}\boldsymbol{B}$.

In the following, we refer to a set of agents $\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^n$, clusters of agents $\mathcal{C}_j \subseteq \mathcal{A}$, and a set of $m$ disjoint clusters $\mathcal{C} = \{\mathcal{C}_j | \mathcal{C}_j \cap \mathcal{C}_i = \emptyset, i \neq j\}_{j=1}^m$, represented as an integer vector $\mathcal{I} \in [1, m]^n$ where value of the $i^{\text{th}}$ element indicates the cluster to which agent $\mathcal{A}_i$ belongs (see Fig. 1). Each of the agents run a local estimator, and each cluster runs an estimator based on the measurement information gathered by the agents in that cluster. The number of estimated states in $\mathcal{C}_j$ is written $N(\mathcal{C}_j)$, the number of measurements taken in $\mathcal{C}_j$ is written $N_{\boldsymbol{y}}(\mathcal{C}_j)$. Constraints are imposed on these quantities (c.f., the energy constraint in KFSS), thus bounding the computations of the estimator in each cluster.

## B. Problem Formulation

**Problem 1** *Assign a set of agents $\mathcal{A} = \{\mathcal{A}_i\}_{i=1}^n$ to a set of disjoint clusters $\mathcal{C} = \{\mathcal{C}_j\}_{j=1}^m$, where each $\mathcal{C}_j$ can fuse the information of its assigned agents, and the agents in turn can leverage this estimate. Minimize a cost $J$ of $\mathcal{I}$ subject to a constraint on the number of states and measurements permitted in $\mathcal{C}_j$, $\max\{N(\mathcal{C}_j), N_{\boldsymbol{y}}(\mathcal{C}_j)\} \leq N_{\max}\ \forall i = 1, ..., m$.*

**Remark 1** *This clustering problem is of a combinatorial nature, a single evaluation of $J(\mathcal{I})$ requires large amounts of computations if expressed in the MSE (see Sec. II).*

## C. Contributions

We propose a greedy algorithm inspired by [14], [16] that approximately solves Problem 1. The exploration is guided by a sequence of operations on the set $\mathcal{C}$, resulting in a genetic algorithm (GA) with the following properties:

(i) The method can handle partially overlapping state-spaces in the agents and correlated inter-agent noise;
(ii) The computational complexity of the clustering per time unit can be traded for slower convergence rates;
(iii) The algorithm can be parallelized and does not apply only to KFs, but can be used with any filter in [6], [9], [10], as it only operates on a filtering posterior.

We assess how the proposed genetic operations affect the expected convergence rate of the cost. This is done through several numerical experiments, including a Monte-Carlo (MC) simulation study on a simplified GNSS example.

## II. PRELIMINARIES

### A. Modeling

In GNSS applications, the filter running in $\mathcal{A}_i$ may entertain different estimates: a relaxed estimate and/or a fixed estimate (see, e.g. [5], [6]). For simplicity, we consider a clustering based on the relaxed estimate, which (after a linearization) obeys Gaussian linear time-varying dynamics [7],

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{q}_k, \qquad \boldsymbol{q}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_k), \tag{1a}$$
$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{r}_k, \qquad \boldsymbol{r}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k). \tag{1b}$$

To better illustrate the algorithm and facilitate the numerical experiments, we use a simplified model for each receiver that still captures much of the properties of a complete GNSS estimation model, where each local agent obeys the dynamics

$$\boldsymbol{x}_{k+1}^i = \boldsymbol{A}_k^i\boldsymbol{x}_k^i + \boldsymbol{q}_k^i, \qquad \boldsymbol{q}_k^i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_k^{ii}), \tag{2a}$$
$$\boldsymbol{y}_k^i = \boldsymbol{C}_k^i\boldsymbol{x}_k^i + \boldsymbol{r}_k^i, \qquad \boldsymbol{r}_k^i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_k^{ii}). \tag{2b}$$

The local state is decomposed into a position $\boldsymbol{p}_k^i \in \mathbb{R}^2$, a velocity $\boldsymbol{v}_k^i \in \mathbb{R}^2$, and a bias $\boldsymbol{\theta}_k \in \mathbb{R}^2$, with $\boldsymbol{x}_k^i = (\boldsymbol{p}_k^i\ ; \boldsymbol{v}_k^i\ ; \boldsymbol{\theta}_k)$. For receives in close proximity of each other (about 3–10km), the bias is shared among all of the models (hence omitting the super-index $i$), and enters on the positional measurements with opposite sign (c.f. [7]),

$$\boldsymbol{A}_k^i = \begin{bmatrix} \boldsymbol{I} & h_k\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}, \quad \boldsymbol{Q}_k^{ii} = \begin{bmatrix} \frac{h_k^3}{3}\boldsymbol{I} & \frac{h_k^2}{2}\boldsymbol{I} & \boldsymbol{0} \\ \frac{h_k^2}{2}\boldsymbol{I} & h_k\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & h_k\boldsymbol{I} \end{bmatrix}, \tag{3a}$$
$$\boldsymbol{C}_k^i = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} & -\boldsymbol{I} \\ \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}, \quad \boldsymbol{R}_k^{ii} = \begin{bmatrix} (\sigma_{1i}^R)^2\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & (\sigma_{2i}^R)^2\boldsymbol{I} \end{bmatrix}. \tag{3b}$$

For the inter-agent noise, let $\mathbb{E}[\boldsymbol{r}_k^i(\boldsymbol{r}_k^j)^\top] = \boldsymbol{R}_k^{ij} = r(\boldsymbol{R}_k^{ii})^{1/2}(\boldsymbol{R}_k^{jj})^{1/2}$, with blocks of the measurement noise covariance matrix in the larger "global" model satisfying

$$\boldsymbol{R}_k^{ii} \succ \boldsymbol{0} \wedge \boldsymbol{R}_k^{jj} \succ \boldsymbol{0} \wedge r \in [0, 1) \Rightarrow \begin{bmatrix} \boldsymbol{R}_k^{ii} & \boldsymbol{R}_k^{ij} \\ \boldsymbol{R}_k^{ji} & \boldsymbol{R}_k^{jj} \end{bmatrix} \succ \boldsymbol{0}. \tag{4}$$

The state-vectors of each agent in (2) contains a unique part and a part that is shared among agents. These relate to the state and measurement vector in a larger global model by

$$\boldsymbol{x}_k \triangleq (\boldsymbol{x}_k^1;\ \cdots\ ;\ \boldsymbol{x}_k^n;\ \boldsymbol{\theta}_k) \triangleq (\boldsymbol{x}_k^u;\ \boldsymbol{\theta}_k) \in \mathbb{R}^{N(\mathcal{C})}, \tag{5a}$$
$$\boldsymbol{y}_k \triangleq (\boldsymbol{y}_k^1;\ \cdots\ ;\ \boldsymbol{y}_k^n) \in \mathbb{R}^{N_{\boldsymbol{y}}(\mathcal{C})}, \tag{5b}$$

where the super-index $(\cdot)^u$ denotes the states that only appear in a single agent. Similarly, in any given cluster $\mathcal{C}_j$, let

$$\bar{\boldsymbol{x}}_k^j \triangleq (\boldsymbol{x}_k^{l_1};\ \cdots\ ;\ \boldsymbol{x}_k^{l_m};\ \boldsymbol{\theta}_k) \triangleq (\boldsymbol{x}_k^{uj};\ \boldsymbol{\theta}_k) \in \mathbb{R}^{N(\mathcal{C}_j)}, \tag{6a}$$
$$\bar{\boldsymbol{y}}_k^j \triangleq (\boldsymbol{y}_k^{l_1};\ \cdots\ ;\ \boldsymbol{y}_k^{l_m}) \in \mathbb{R}^{N_{\boldsymbol{y}}(\mathcal{C}_j)}, \tag{6b}$$

for all $\mathcal{A}_{l_i} \in \mathcal{C}_j$, where $l_1 < l_2 < ... < l_{m_j}$ and $m_j = |\mathcal{C}_j|$. To simplify the discussion, a general linear map is defined to relate the global state-vector in (5) to the state-vector in a cluster $\mathcal{C}_j \in \mathcal{C}$ characterized by $\mathcal{I}$. To this end, let

$$\bar{\boldsymbol{x}}^j = \boldsymbol{M}_{\mathcal{I}}^j \boldsymbol{x} = (\boldsymbol{M}_{\mathcal{I}}^{uj} + \boldsymbol{M}_{\mathcal{I}}^{sj})\boldsymbol{x}, \quad \boldsymbol{M}_{\mathcal{I}}^j \in \{0,1\}^{N(\mathcal{C}_j)\times N(\mathcal{C})},$$

with $\boldsymbol{M}_{\mathcal{I}}^{uj}\boldsymbol{x} = (\boldsymbol{x}^{uj}; \boldsymbol{0}) \in \mathbb{R}^{N(\mathcal{C}_j)}$ and $\boldsymbol{M}_{\mathcal{I}}^{sj}\boldsymbol{x} = (\boldsymbol{0}; \boldsymbol{\theta}) \in \mathbb{R}^{N(\mathcal{C}_j)}$.

### B. Defining the Objective

In the GNSS literature, specifications are often given in terms of the MSE, where the positional MSE is of particular interest. However, when considering multiple receivers with partially overlapping states, other costs may be relevant. For instance, in an urban driving scenario relying on positional information, it may not matter if the estimate is excellent in all but one receiver, if the estimation errors in one receiver are large enough to violate the specification. An additional complicating factor is the shared states $\boldsymbol{\theta}_k$ in (6). To illustrate this, consider the partition of the set $\mathcal{C}$ into two clusters, $\mathcal{C}_1$ and $\mathcal{C}_2$, defined by $\mathcal{I} \in \{1,2\}^n$. In the unlikely event that the re-clustering does not change the estimates, we have that

$$\begin{bmatrix} \hat{\bar{\boldsymbol{x}}}^i \\ \bar{\boldsymbol{x}}^i \end{bmatrix} = (\boldsymbol{M}_{\mathcal{I}}^i \otimes \boldsymbol{I}_2) \begin{bmatrix} \hat{\boldsymbol{x}} \\ \boldsymbol{x} \end{bmatrix}, \tag{7}$$

then, unless $\mathrm{MSE}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) \triangleq \mathbb{E}[\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\|_2^2] = 0$,

$$\mathrm{MSE}(\boldsymbol{x}, \hat{\boldsymbol{x}}) \neq \sum_{i=1}^{2} \mathrm{MSE}(\bar{\boldsymbol{x}}^i, \hat{\bar{\boldsymbol{x}}}^i). \tag{8}$$

As such, to compare the effects of a particular clustering to another, we instead need to compare the quantities

$$\mathrm{MSE}\left(\begin{bmatrix} \boldsymbol{M}_{\mathcal{I}}^1 \\ \boldsymbol{M}_{\mathcal{I}}^2 \end{bmatrix} \boldsymbol{x}, \begin{bmatrix} \boldsymbol{M}_{\mathcal{I}}^1 \\ \boldsymbol{M}_{\mathcal{I}}^2 \end{bmatrix} \hat{\boldsymbol{x}}\right) \leq \sum_{i=1}^{2} \mathrm{MSE}(\bar{\boldsymbol{x}}^i, \hat{\bar{\boldsymbol{x}}}^i). \tag{9}$$

which, unlike (8), holds with equality in the case of (7). The inequality holds when using minimum MSE (MMSE) estimators to resolve $\hat{\boldsymbol{x}}$, $\hat{\bar{\boldsymbol{x}}}^1$, $\hat{\bar{\boldsymbol{x}}}^2$, under the assumption that additional information leads to a lower MSE. Similarly, if comparing MSE of clusters of different size, we need to account for the number of unique and shared states differently. Motivated by these considerations, we define:

- $J_G(\mathcal{I}) = (\sum_i^m N(\mathcal{C}_i))^{-1} \sum_i^m \mathrm{MSE}(\bar{\boldsymbol{x}}^i, \hat{\bar{\boldsymbol{x}}}^i)$,
- $J_C(\mathcal{I}) = \max_j (|\mathcal{C}_j| d)^{-1} \mathrm{MSE}(\bar{\boldsymbol{H}}_j \bar{\boldsymbol{x}}^j, \bar{\boldsymbol{H}}_j \hat{\bar{\boldsymbol{x}}}^j)$,
- $J_L(\mathcal{I}) = \max_j \max_i d^{-1} \mathrm{MSE}(\bar{\boldsymbol{H}}_{ij}^i \bar{\boldsymbol{x}}^j, \boldsymbol{H}_{ij}^i \hat{\bar{\boldsymbol{x}}}^j)$,

where $d = 2$, $\bar{\boldsymbol{H}}_j$ extracts the $d$-dimensional positional states of the state-vector $\hat{\bar{\boldsymbol{x}}}^j$, while $\bar{\boldsymbol{H}}_{ij}$ extracts the positional state-vector of the $i^{\text{th}}$ agent in the $j^{\text{th}}$ cluster. Here, $J_G(\mathcal{I})$ is the total MSE per state among all of the agents, $J_C(\mathcal{I})$ denotes the same measure but only considering the position in the worst performing cluster, and $J_L(\mathcal{I})$ is the worst positional MSE in the worst performing agent in any cluster.

Ideally, the clustering, $\mathcal{I}$, should minimize $J_L(\mathcal{I})$, but a more flexible cost is formed as the linear combination

$$J(\mathcal{I}) = \alpha_G J_G(\mathcal{I}) + \alpha_C J_C(\mathcal{I}) + \alpha_L J_L(\mathcal{I}), \tag{10}$$

for some positive parameters $\alpha_G, \alpha_C, \alpha_L > 0$. In the following, these costs will be evaluated in the filtering posterior.

**Remark 2** *If we disregard the constraint on the cluster state dimension in Problem 1, any clustering operation that involves filtering with MMSE estimators satisfies* (9) *when partitioning a cluster into subsets. Hence, the best clustering is to have a single cluster with all of the agents, $m = 1$, $\mathcal{C}_1 = \{\mathcal{A}_1, \cdots, \mathcal{A}_n\}$. Similarly, the worst possible clustering is to have one agent in each cluster, $m = n$, $\mathcal{C}_i = \{\mathcal{A}_i\}$.*

### C. Kalman Filtering

In the context of the dynamics defined in Sec. II-A, the MMSE estimator is the KF [17], which is initialized from a prior $\boldsymbol{x}_0 \sim \mathcal{N}(\hat{\boldsymbol{x}}_{0|0}, \boldsymbol{P}_{0|0})$ followed by a prediction step,

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{A}_{k-1}\hat{\boldsymbol{x}}_{k-1|k-1}, \tag{11a}$$
$$\boldsymbol{P}_{k|k-1} = \boldsymbol{A}_{k-1}\boldsymbol{P}_{k-1|k-1}\boldsymbol{A}_{k-1}^\top + \boldsymbol{Q}_{k-1}, \tag{11b}$$

and an update step

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{C}_k^\top(\boldsymbol{C}_k\boldsymbol{P}_{k|k-1}\boldsymbol{C}_k^\top + \boldsymbol{R}_k)^{-1}, \tag{12a}$$
$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k(\boldsymbol{y}_k - \boldsymbol{C}_k\hat{\boldsymbol{x}}_{k|k-1}), \tag{12b}$$
$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{C}_k)\boldsymbol{P}_{k|k-1}. \tag{12c}$$

Thus, if a clustering given by $\mathcal{I}_k$ is fixed in time, the dynamics in each resulting cluster are known, and the mean and covariance at time step $k$ can be computed by iterating (11b) and (12c) for each resulting system over a sufficiently large number of iterations, here denoted with $\tau$. As the considered systems are non-autonomous in general, the cost is approximated by considering the filtering posterior covariance $\boldsymbol{P}_{k|k}$ resulting from a clustering $\mathcal{I}_{k-\tau} = \cdots = \mathcal{I}_{k-1} = \mathcal{I}_k$ over the time interval $[k-\tau, k]$, and making a forward pass from a prior $\boldsymbol{P}_{k-\tau|k-\tau}$. This is illustrated in Fig. 2, and the implementation is summarized in Algorithm 1.

**Algorithm 1** Cost evaluation by a forward pass.

```
    Receive: ${\{\bar{\boldsymbol{P}}^i_{k-\tau|k-\tau}\}_{i=1}^{|\mathcal{C}|}}$, $\mathcal{I}_k$, $\mathcal{I}_k^*$, $\tau$, and let $l = k - \tau$.
    // Compute prior in the new clusters
 1: $\boldsymbol{P}_{l|l} \triangleq \sum_{i=1}^{|\mathcal{C}|}(\boldsymbol{M}^{ui}_{\mathcal{I}_k})^\top \boldsymbol{P}^i_{l|l}\boldsymbol{M}^{ui}_{\mathcal{I}_k} + \frac{1}{|\mathcal{C}|}(\boldsymbol{M}^{si}_{\mathcal{I}_k})^\top \boldsymbol{P}^i_{l|l}\boldsymbol{M}^{si}_{\mathcal{I}_k}$
 2: $\bar{\boldsymbol{P}}^i_{l|l} \triangleq \sum_{i=1}^{|\mathcal{C}^*|}\boldsymbol{M}^{ui}_{\mathcal{I}_k^*}\boldsymbol{P}^i_{l|l}(\boldsymbol{M}^{ui}_{\mathcal{I}_k^*})^\top + \frac{1}{|\mathcal{C}^*|}\boldsymbol{M}^{si}_{\mathcal{I}_k^*}\boldsymbol{P}^i_{l|l}(\boldsymbol{M}^{si}_{\mathcal{I}_k^*})^\top$
    // Recompute estimate at $k$ in modified clusters
 3: for $c \in \mathrm{unique}(\mathcal{I}_k^*)$ do
 4:     if $\mathcal{C}_c \cap \mathcal{C}_c^* = \mathcal{C}_c \cup \mathcal{C}_c^*$ then
 5:         $\bar{\boldsymbol{P}}^{i,*}_{k|k} \triangleq \bar{\boldsymbol{P}}^i_{k|k}$
 6:     else
 7:         Evaluate $\bar{\boldsymbol{P}}^{i,*}_{k|k}$ from $\bar{\boldsymbol{P}}^{i,*}_{l|l}$ using (11) and (12)
 8:     end if
 9: end for
10: Evaluate $J(\mathcal{I}_k^*)$ from ${\{\bar{\boldsymbol{P}}^{i,*}_{k|k}\}_{i=1}^{|\mathcal{C}^*|}}$ and output result
```

As such, to evaluate the cost in (10) for a single candidate cluster, we need to run the relevant filters in all modified clusters over $\tau$ time steps, where $\tau$ is sufficiently large to contain any transient induced by the prior set at $k - \tau$.

**Remark 3** *Moving an agent to a new cluster not only changes the state-space of each individual cluster, but also the measurements that will be available to the cluster in the*

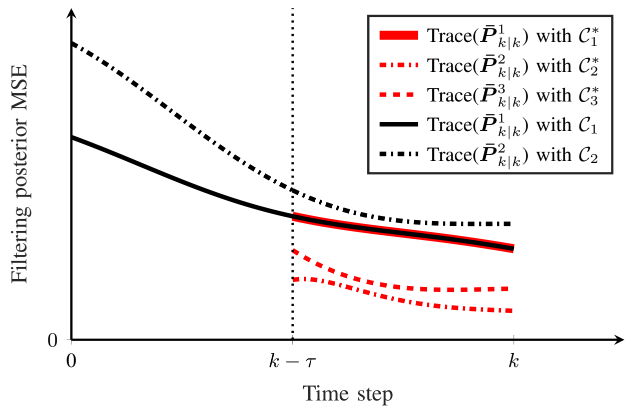

Fig. 2. Sketch depicting the estimate posterior MSE as a function of time in two differently formed clusters mirroring Fig. 1: with $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2\}$ (black), and $\mathcal{C}^* = \{\mathcal{C}^*_1, \mathcal{C}^*_2, \mathcal{C}^*_3\}$ (red). At a time $k$, the performance of cluster $\mathcal{C}^*$ is evaluated by computing the filtering posterior, when initialized from a prior computed by the estimate corresponding to the clustering $\mathcal{C}$ at $k-\tau$.

*future. By evaluating the cost in Algorithm 1, we approximately extrapolate how the clustering is likely to perform in the future based on its performance on the interval* $[k-\tau, k]$.

### D. Combinatorial Properties of the Optimization Problem

Theoretically, the best possible clustering given the energy constraints in Problem 1 is computable. To illustrate the complexity of such a computation, we note that just checking all of the possible solutions with $n$ agents into $m$ clusters of equal size results in $n!/(((n/m)!)^m(m)!)$ possible cluster combinations. For the problem of size $(n, m) = (30, 10)$ considered later in Sec. IV-A, this equates to $\approx 10^{18}$ solutions. In practice, this number can be much larger, as both $|\mathcal{C}_j|$ and $|\mathcal{C}|$ are permitted to vary when re-clustering.

Relating this back to the KFSS problem in [14], the greedy algorithms iteratively include sensors based on how the resulting measurements affect the estimate covariance. With $n$ sensors and a sensor budget of $m$ (akin to having two clusters, $\mathcal{C}_1$ with $m$ agents, and $\mathcal{C}_2$ with $m-n$ agents), this includes $m(n-m)$ evaluations of an objective expressed in the posterior of the estimate in $\mathcal{C}_1$. However, for Problem 1 and the cost in (10), a modification of $\mathcal{C}_1$ might imply a modification of $\mathcal{C}\backslash\{\mathcal{C}_1\}$ that affects the cost. Hence, checking all of the possible ways in which an agent can be introduced into the clusters quickly becomes infeasible due to the number of evaluations of (10). Consequently, we seek numerically tractable heuristics that can solve the problem sub-optimally by incrementally improving the clustering with respect to the cost in (10), in as few evaluations of the cost as possible.

## III. A GENETIC ALGORITHM

In this section, we propose a GA heuristic that approximately solves Problem 1. GAs are gradient-free heuristics that are often applied to approximately solve combinatorial optimization problems [18, Chapter 5]. In particular, GAs have been considered in computer science applications related to data clustering and feature selection problems [19]–[21]. In the context of Problem 1, the appeal of the GA is its ability to incrementally improve solutions, as highlighted in [22]. Furthermore, the structure of the algorithm allows for parallelism, and its expected convergence rate (when considered as a function of time) can be easily traded for a decrease in computational complexity per time step, which becomes particularly relevant for real-time implementations.

In classical GAs, a set of solutions (a generation) are permuted, and allowed to propagate to future generations by mechanisms inspired by genetics and natural selection. For the clustering problem, the set of candidate solutions consists of $N_S$ integer vectors $\mathcal{I}^i \in [1, m]^n$, and the fitness function is defined as the cost in (10). To clarify the presentation, let $\mathcal{S} = \{(\mathcal{I}^i, J(\mathcal{I}^i))\}_{i=1}^{N_S}$, and consider an algorithm consisting of three main operations: selection, cross-over, and mutation.

*1) Selection:* The selection step consists of randomly selecting candidate solutions from $\mathcal{S}$. Here, the solutions are ordered by fitness, and their sampling is heavily skewed toward solution candidates with lower fitness. This is done by selecting the best solution, $\mathcal{I}^1$, and the $(r+1)^{\text{th}}$ best solution with $r = \max\{\lceil|\bar{r}|\rceil, N_S - 1\}$ where $\bar{r} \sim \mathcal{N}(0, \alpha(N_S - 1))$.

*2) Cross-over:* The cross-over operations, denoted by $\mathcal{O}^C_i$, are subsequently applied to any two selected solutions to produce a new solution candidate in the next generation. Two such operations are defined in Table I.

*3) Mutation:* The solutions produced by the cross-over are subject to a mutation operation, here denoted with $\mathcal{O}^M_i$, defined in Table I. The mutations result in a new solution set with cardinality $N_S$, and the process is repeated from the selection step. The algorithm is summarized in Algorithm 2.

TABLE I
OPERATIONS CONSIDERED IN THE GENETIC ALGORITHM.

| Operation | Description |
|---|---|
| $\mathcal{O}^C_1(\mathcal{I}^A, \mathcal{I}^B)$ | Randomize $j \in \{1, ..., n\}$. Combine the first $l$ elements of $\mathcal{I}^A$ with last $j-l$ elements of $\mathcal{I}^B$. |
| $\mathcal{O}^C_2(\mathcal{I}^A, \mathcal{I}^B)$ | For each element in the solutions, choose the element in the output from $\mathcal{I}^A$ or $\mathcal{I}^B$ with equal probability. |
| $\mathcal{O}^M_1(\mathcal{I})$ | Set a random element in $\mathcal{I}$ to a number in $[1, |\mathcal{C}|]$. |
| $\mathcal{O}^M_2(\mathcal{I})$ | Swap the positions of two random elements in $\mathcal{I}$. |
| $\mathcal{O}^M_3(\mathcal{I})$ | Apply $\mathcal{O}^C_2$ to two randomly selected clusters in $\mathcal{I}$. |

### A. Implementation Considerations

The operations defined in Table I not only change the solution by reassigning agents to new clusters, but $\mathcal{O}^M_1$ and $\mathcal{O}^M_3$ also modify the cardinality of individual clusters, and possibly also the number of clusters. As such, when performing the operations, checks needs to be made to ensure that the resulting solution is feasible subject to the constraint in Problem 1. This check is implicitly done when performing the operations, which only return a solution if it is feasible.

Due to the high computational cost associated with approximately evaluating the fitness of the candidate solutions with Algorithm 1, the GA heuristic is designed to always propagate the best solution in one generation to the next without applying mutations. This makes the algorithm less

**Algorithm 2** A genetic algorithm for dynamic clustering.

```
    Receive: $\mathcal{S}_0=\{(\mathcal{I}_0^i, J(\mathcal{I}_0^i))\}_{i=1}^{N_S}, \alpha_G, \alpha_C, \alpha_L, \alpha, N_S, N_{\max}$
    Let $J(\cdot) = \alpha_G J_G(\cdot) + \alpha_C J_C(\cdot) + \alpha_L J_L(\cdot)$ and $p = 1$
 1: for $k = 1$ to $K$ do
        // Filtering
 2:     Run filters in $\mathcal{C}_j$ according to (11) and (12)
        // Clustering
 3:     if $\mathrm{mod}(k, N_T) = 0$ then
            // Selection
 4:         Draw $\bar{r} \sim \mathcal{N}(0, (\alpha(N_S - 1))^2)$
 5:         Round $r$ to nearest integer in $[1, N_S - 1]$.
            // Cross-over
 6:         Let $\mathcal{S}_p = \{(\mathcal{I}_{p-1}^1, J(\mathcal{I}_{p-1}^1))\}$
 7:         while $|\mathcal{S}_k| < N_S/2$ do
 8:             Draw $i \in \{1, 2\}$
 9:             $\mathcal{S}_p \triangleq \mathcal{S}_p \cup \mathcal{O}_i^C(\mathcal{I}_{p-1}^1, \mathcal{I}_{p-1}^{r+1})$
10:         end while
            // Mutation
11:         while $|\mathcal{S}_p| < N_S$ do
12:             Draw $i \in \{1, 2, 3\}$
13:             $\mathcal{S}_p \triangleq \mathcal{S}_p \cup \mathcal{O}_i^M(\mathcal{I}_p^l)$
14:         end while
            // Acceptance
15:         Sort $\mathcal{S}_p$ by increasing fitness.
16:     end if
        // Re-initialization
17:     if $\mathcal{I}_p^1 \neq \mathcal{I}_{p-1}^1$ then
18:         Redefine $\mathcal{C} = \{\mathcal{C}_j\}_{j=1}^m$ according to $\mathcal{I}_p^1$
19:         Reinitialize the estimates in $\mathcal{C}_j$ with the
            moments used in evaluating $J(\mathcal{I}_p^1)$
20:     end if
21: end for
```

likely to escape locally optimal solutions, but has the advantage that the best solution (found so far) will always be present in future generations until a better one is found.

If the considered estimation problem is large and the sampling time $h_k$ in (3) is small, it may be infeasible to compute the clustering at each time step $k$. In Algorithm 2, the generations are therefore only propagated once every $N_T$ time steps. Consequently, the computation of new clusters can be spread out over time, resulting in a worst case computational complexity of $O(m\tau N_T^{-1} N_S N_{max}^3)$ per propagated generation and per time step. In practice, the complexity is smaller, as only the subset of $\mathcal{C}$ that changes between generations needs to be recomputed by Algorithm 1.

## IV. NUMERICAL EXAMPLES

To demonstrate the heuristic in Algorithm 2, we consider a scenario with cluster dynamics defined by (2), (3), (6). We start by studying a case where the number of agents per cluster is fixed (see Sec. IV-A). For this example, the expected convergence rate is explored as a function of the operations used in the GA (see Sec. IV-B). Finally, to show the flexibility of the approach, an example is given where the number of agents per cluster varies with time (see Sec. IV-C).

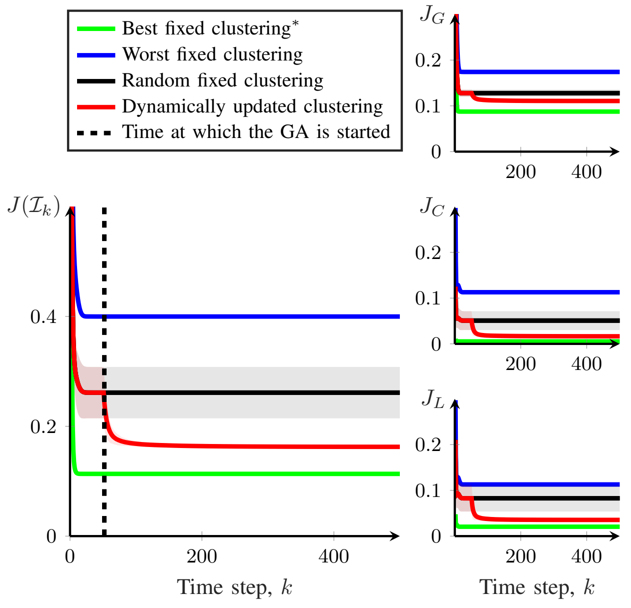

Fig. 3. *Left:* The cost in (10) as a function of time, when (i) separating all agents into individual clusters (blue); (ii) using a random clustering that satisfies the constraints, depicted with an empirical 2-$\sigma$ confidence interval (black); (iii) running the proposed heuristic from the same cluster initializations, depicted with an empirical 2-$\sigma$ confidence interval (red, here the variance becomes vanishingly small); and (iv) forming a single large cluster, thus violating the constraint which provides a lower bound on the cost when operating *without* the cluster size constraint (green). *Right, top to bottom:* Components of the cost as a function of time, with global cost, $J_G$, cluster cost, $J_C$, and local cost, $J_L$.

### A. Clustering with Constant Cluster Sizes

In this example we consider $n = 30$ agents and $m = 10$ clusters, with a constraint of $N_{\max} = 14$ states per cluster. Given the state composition in Sec. II-A, with four unique states and two shared states (biases) per agent, this constrains each cluster to operate with exactly three agents at all times. From the simulation, we compute the cost associated with:

(i) The worst possible clustering ($m = n$);
(ii) The best possible clustering *when removing the constraint on the number of states in Problem 1* ($m = 1$);
(iii) A random clustering that is fixed in time, while enforcing the constraint in Problem 1 ($m = 10$).
(iv) Dynamically re-clustering with the GA in Algorithm 2, while enforcing the constraint in Problem 1 ($m = 10$).

The cost is defined as in (10), and evaluated in the filtering posterior with $\alpha_G = \alpha_C = \alpha_L = 1$. This weighting is arbitrary, and can be changed to reflect the specifications of the application. The GA operates with $\alpha = 0.1$, $N_S = 20$, $\tau = 20$, where the last parameter is set long enough to contain the typical transients induced by the prior set in Alg. 1. Furthermore, we let $N_T = 1$, and each simulation runs for $K = 500$ time steps. The dynamics are defined with $h_k = 0.1$ for all $k$, and the measurement noise is realized with $(\sigma_{1i}^R; \sigma_{2i}^R) \sim \mathcal{U}((0, 1]^2)$ for all $i = 1, ..., n$ and $r = 0.9$. A set of $10^3$ MC-simulations are performed with (ii) and (iii) to compute the empirical mean and variance of the cost evolution as time progresses. The exact same problem realization is used in all simulations, with the only variation

TABLE II

CONVERGENCE RATES IN EXPECTATION AS A FUNCTION OF THE USED GA OPERATIONS ($\mathrm{Var}[J(\mathcal{I}_K)] \approx 10^{-4}$ IN ALL TESTED CASES).

| Operations | $k^*_{\epsilon=0.1}$ | $k^*_{\epsilon=0.05}$ | $k^*_{\epsilon=0.01}$ | $\mathbb{E}[J(\mathcal{I}_K)]$ |
|---|---|---|---|---|
| $O = (0, 1, 1, 1, 1)$ | 71 | 94 | 203 | 0.1775 |
| $O = (1, 0, 1, 1, 1)$ | 73 | 99 | 201 | 0.1775 |
| $O = (1, 1, 0, 1, 1)$ | 77 | 107 | 248 | 0.1780 |
| $O = (1, 1, 1, 0, 1)$ | 70 | 91 | 194 | 0.1774 |
| $O = (1, 1, 1, 1, 0)$ | 71 | 93 | 182 | 0.1776 |
| $O = (1, 1, 1, 1, 1)$ | 72 | 94 | 203 | 0.1773 |

being in the seed of the initial clustering. The GA is switched on at $k = 50$, and the resulting cost is depicted in Fig. 3.

Despite keeping the number of agents per cluster constant over time, just exchanging agents across the clusters using the heuristic in Algorithm 2 provides a significant improvement in the mean of the cost. Notably, there is relatively large variance in the cost associated with the random clustering (black), where $\mathrm{Var}[J(\mathcal{I}_K)] \approx 5 \cdot 10^{-4}$. This is reduced by the GA heuristic (red), to $\mathrm{Var}[J(\mathcal{I}_K)] \approx 10^{-6}$. The heuristic does not converge to the same solution across all realizations, but the variance of the cost at $K = 500$ is small, and it is clear that significant gains in estimation performance can be made by using the GA for dynamic re-clustering.

## B. Expected Convergence Rate vs. Included Operations

To demonstrate that the considered operations achieve the desired effect, a total of $10^3$ MC-simulations are performed on the same problem as in Sec. IV-A, now varying the problem realization and only permitting subsets of the defined GA operations. To study this, a vector $\boldsymbol{O} \in \{0, 1\}^5$ is used to indicate the included operations, with the $i^{\text{th}}$ index set to 1 indicating that the operation on the $i^{\text{th}}$ row in Table I is used. For each tested combination of the operations, measure of of cost convergence, here weak convergence, is evaluated empirically. We also compute the time step when this measure is below a threshold $\epsilon$ in expectation, as

$$k^*_\epsilon = \min\left\{ k \in [1, K] \;\middle|\; \mathbb{E}\left[\frac{J(\mathcal{I}_k) - J(\mathcal{I}_K)}{J(\mathcal{I}_K)}\right] \le \epsilon \right\}.$$

The resulting convergence rates are characterized in Tab. II, and we emphasize that the elimination of an operation in Algorithm 2 does not imply fewer number of evaluations of the cost in (10). The cost is evaluated exactly the same amount of times in all tested combinations of the operations.

In this experiment, the proposed combination of operations in Sec. III yields the lowest expected cost at time step $k = 500$ for the problem defined Sec. II, but it is clear from the convergence rates that while the mutation operation $\mathcal{O}^M_1$ is necessary, either $\mathcal{O}^M_2$ and $\mathcal{O}^M_3$ could be removed. Thus, a simpler GA using a subset of the defined operations is still expected to significantly outperform a random clustering (compare the right-most column in Table II with Fig. 3).

## C. Clustering with Time-varying Cluster Sizes

To demonstrate the flexibility of the proposed GA heuristic, the simulation in Sec. IV-C is repeated with a relaxed constraint characterized by $N_{\max} = 42$, thus permitting a maximum of 10 agents per cluster. In light of Remark 2, we expect the heuristic to remove some of the clusters entirely, and maximize the number of agents in each cluster. Indeed, given the state composition in Sec. II-A, the algorithm should find a solution with $M = 3$ clusters containing 10 agents each, which is precisely the behavior observed in Fig. 4.

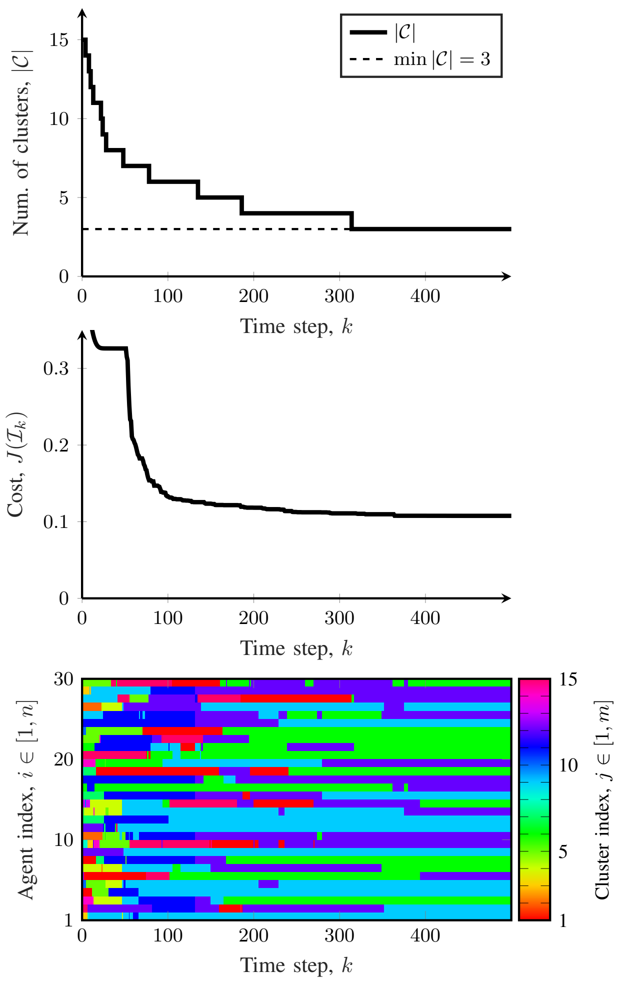

Fig. 4. *Top:* The number of clusters used as a function of the time-step, $k$. *Center:* The cost in (10) as a function of the time step, $k$. *Bottom:* Time evolution of the best solution found by the GA. Each column depicts the index vector $\mathcal{I}^1_k$ at a time $k$: the color indicates the cluster to which the agent $\mathcal{A}_i$ belongs to at a time $k$. At $k = 500$, there are three clusters.

The resulting clustering yields a lower cost of approximately $J(\mathcal{I}_K)|_{N_{\max}=42} \approx 0.108$, which can be compared to the expected cost of $\mathbb{E}[J(\mathcal{I}_K)|_{N_{\max}=14}] \approx 0.177$ in Fig. 3. Importantly, this simulation also shows that the total number of clusters used is time varying, from the initial 15 clusters to the expected 3 clusters at $k = K = 500$. Indeed, a solution with 3 clusters is found already at approximately $k = 320$, but it is refined slightly even after this point (see Fig. 4).

## V. CONCLUSIONS

In this paper, we propose a heuristic method for clustering agents to minimize a cost expressed in the filtering posterior. The method is targeted at multi-receiver GNSS positioning, for the case where $\max\{N(\mathcal{C}), N_{\boldsymbol{y}}(\mathcal{C})\}^3$ is large. Specifically, the method forms a set of clusters that requires $O(m \max\{N(\mathcal{C}_j), N_{\boldsymbol{y}}(\mathcal{C}_j)\}^3) \leq O(m N_{\max}^3)$ operations per time step, which may be paralellized. To find a clustering, the proposed GA requires less than $O(m\tau N_T^{-1} N_S N_{max}^3)$ operations per time step, and similar to the filters, these computations can also be distributed among multiple computers. An important feature of the method is the parameter $N_T$ (how frequently re-clustering is done), which permits a trade-off between computational cost per time step and convergence rate of the chosen MSE cost. To facilitate an empirical study, an example was given where the GA heuristic yielded a substantial improvement in the estimation performance when compared to a strategy of random clustering in which the number of agents per cluster was maximized. Here, the GA reduced both the variance and and mean of the cost significantly. Similarly, the algorithm performed well when run on a problem where the cardinality of the clusters changed with time. As such, the heuristic is flexible, guarantees that the performance is no worse than the initial clustering (in the linear time-invariant setting), and can be parallelized, thereby facilitating collaborative estimation schemes with a large numbers of agents in a GNSS setting.

Future work will involve the evaluation of the algorithm in real-time GNSS applications involving a large number of moving receivers, and a theoretical examination of worst-case convergence rate of the algorithm leveraging the approximate supermodularity properties of the KFSS [15].

## References

[1] L. Wijand, "Long-range navigation system," 1959, US Patent, Last accessed 02/01/2022 at patents.google.com/patent/US2980907.

[2] T. Takasu, "RTKLIB ver. 2.4.2 manual," *RTKLIB: An Open Source Program Package for GNSS Positioning*, pp. 29–49, 2013.

[3] M. Sahmoudi and R. Landry, "A nonlinear filtering approach for robust multi-GNSS RTK positioning in presence of multipath and ionospheric delays," *IEEE Journal of Selected Topics in Signal Processing*, vol. 3, no. 5, pp. 764–776, 2009.

[4] R. Kroes, O. Montenbruck, W. Bertiger, and P. Visser, "Precise grace baseline determination using GPS," *GPS Solutions*, vol. 9, no. 1, pp. 21–31, 2005.

[5] T. Takasu and A. Yasuda, "Kalman-filter-based integer ambiguity resolution strategy for long-baseline RTK with ionosphere and troposphere estimation," in *Proceedings of the ION GNSS*, 2010, pp. 161–171.

[6] S. Zhao, X. Cui, F. Guan, and M. Lu, "A Kalman filter-based short baseline RTK algorithm for single-frequency combination of GPS and BDS," *Sensors*, vol. 14, no. 8, pp. 15 415–15 433, 2014.

[7] M. Greiff, K. Berntorp, S. Di Cairano, and K. J. Kim, "Mixed-integer linear regression Kalman filters for GNSS positioning," in *2021 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2021, pp. 980–985.

[8] P. Teunissen, "A canonical theory for short GPS baselines. Part III: the geometry of the ambiguity search space," *Journal of Geodesy*, vol. 71, no. 8, pp. 486–501, 1997.

[9] M. Greiff and K. Berntorp, "Optimal measurement projections with adaptive mixture Kalman filtering for GNSS positioning," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 4435–4441.

[10] K. Berntorp, A. Weiss, and S. Di Cairano, "Integer ambiguity resolution by mixture Kalman filter for improved GNSS precision," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 4, pp. 3170–3181, 2020.

[11] S. S. Hwang and J. L. Speyer, "Particle filters with adaptive resampling technique applied to relative positioning using GPS carrier-phase measurements," *IEEE transactions on control systems technology*, vol. 19, no. 6, pp. 1384–1396, 2010.

[12] A. S. Chhetri, D. Morrell, and A. Papandreou-Suppappola, "Scheduling multiple sensors using particle filters in target tracking," in *IEEE Workshop on Statistical Signal Processing, 2003*. IEEE, 2003, pp. 549–552.

[13] L. Zuo, R. Niu, and P. K. Varshney, "Posterior CRLB based sensor selection for target tracking in sensor networks," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 2. IEEE, 2007, pp. II–1041.

[14] H. Zhang, R. Ayoub, and S. Sundaram, "Sensor selection for Kalman filtering of linear dynamical systems: Complexity, limitations and greedy algorithms," *Automatica*, vol. 78, pp. 202–210, 2017.

[15] L. F. Chamon, G. J. Pappas, and A. Ribeiro, "The mean square error in Kalman filtering sensor selection is approximately supermodular," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 343–350.

[16] S. Liu, S. P. Chepuri, M. Fardad, E. Macsazade, G. Leus, and P. K. Varshney, "Sensor selection for estimation with correlated measurement noise," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3509–3522, 2016.

[17] S. Särkkä, *Bayesian filtering and smoothing*. Cambridge University Press, 2013, vol. 3.

[18] X.-S. Yang, *Nature-inspired optimization algorithms*, 1st ed. Elsevier, 2014.

[19] E. K. Burke, E. K. Burke, G. Kendall, and G. Kendall, *Search methodologies: introductory tutorials in optimization and decision support techniques*. Springer, 2014.

[20] U. Maulik and S. Bandyopadhyay, "Genetic algorithm-based clustering technique," *Pattern recognition*, vol. 33, no. 9, pp. 1455–1465, 2000.

[21] X.-Y. Liu, Y. Liang, S. Wang, Z.-Y. Yang, and H.-S. Ye, "A hybrid genetic algorithm with wrapper-embedded approaches for feature selection," *IEEE Access*, vol. 6, pp. 22 863–22 874, 2018.

[22] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," *SIAM journal on computing*, vol. 2, no. 2, pp. 88–105, 1973.