Distributed least-squares estimation applied to GNSS networks

Khodabandeh, A.; Teunissen, P. J.G.

DOI
10.1088/1361-6501/ab034e

Publication date
2019

Document Version
Accepted author manuscript

Published in
Measurement Science and Technology

Citation (APA)
Khodabandeh, A., & Teunissen, P. J. G. (2019). Distributed least-squares estimation applied to GNSS networks. *Measurement Science and Technology*, *30*(4), Article 044005. https://doi.org/10.1088/1361-6501/ab034e

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Noname manuscript No.
(will be inserted by the editor)

Distributed least-squares estimation applied to GNSS networks

A. Khodabandeh[1,2] · **P.J.G. Teunissen**[2,3]

Received: date / Accepted: date

Abstract In view of the recent proliferation of low-cost mass-market receivers, the number of network receivers and GNSS users will be growing rapidly, demanding an efficient way of data processing in terms of computational power and capacity. One way of improving the computational capacity is to decentralize the underlying data processing and distribute the task of the computer center across individual network receivers. In this invited contribution we review the problem of distributed estimation and present an algorithm for distributed least-squares estimation using the alternating direction method of multipliers. Applying the algorithm to a network of GNSS receivers, we show how the distributed data processing of individual receivers can deliver parameter solutions comparable to their centralized network-derived counterparts. With distributed estimation techniques, GNSS single-receiver users can therefore obtain high-precision solutions without the need of having a centralized computing center.

Keywords Distributed estimation, Least-squares estimation, Alternating direction method of multipliers, Global Navigation Satellite Systems (GNSS)

1 Introduction

Parameter estimation in Global Navigation Satellite Systems (GNSS) often relies on the data processing of a network of receivers that collect measurements from GNSS satellites (Hofmann-Wellenhof et al, 2008; Teunissen and Montenbruck, 2017). The measurements are transferred to a computing center to deliver network-derived parameter solutions. In the light of the recent development of new GNSS constellations as well as the proliferation of mass-market receivers, the number of network receivers and GNSS users will be growing rapidly, demanding an efficient way of data processing in terms of computational power and capacity. One way of improving the computational capacity is to *decentralize* the underlying data processing and distribute the task of the computer center across several individual network receivers. In contrast to a centralized network processing where all receivers have to send their data to a computing (fusion) center, distributed data processing schemes enable the receivers to only share limited information with their neighboring receivers (i.e. a subset of all other receivers) and yet obtain solutions that are comparable to their centralized network-derived counterparts in a mean-squared-error sense (see Figure 1). This particular feature of the distributed data processing potentially makes the data communication between the receivers cost-effective and develop the receivers capacity to perform *parallel* computations (Molzahn et al, 2017).

Parameter estimation in decentralized and distributed form has been extensively studied in recent years, see e.g. (Olfati-Saber and Murray, 2004; Moreau, 2005; Kingston and Beard, 2006; Boyd et al, 2011; Das and Moura, 2017). For its applications to GNSS, see (Boomkamp, 2010) or the recent contributions (Khodabandeh et al, 2018) and (Wang et al, 2018). In the present contribution we review the problem of distributed estimation and present an algorithm for distributed least-squares estimation. It is intended to demonstrate how distributed

A. Khodabandeh (akhodabandeh@unimelb.edu.au)
P.J.G. Teunissen (p.teunissen@curtin.edu.au)

[1]Department of Infrastructure Engineering, The University of Melbourne, Melbourne, Australia
[2]School of Earth and Planetary Sciences, Curtin University, Perth, Australia
[3]Department of Geoscience and Remote Sensing, Delft University of Technology, Delft, The Netherlands

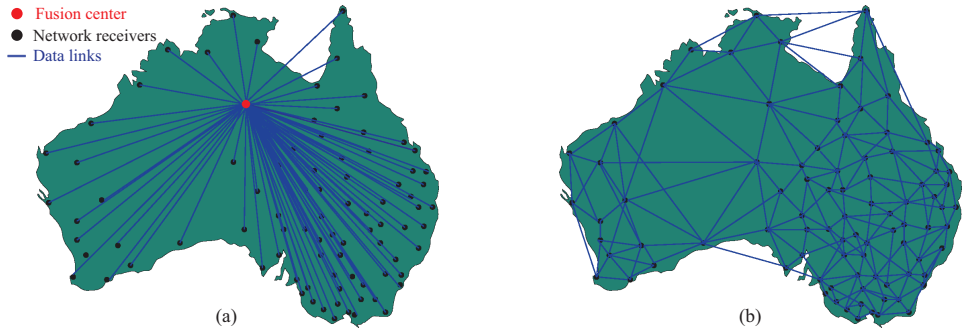

Fig. 1 Centralized versus distributed processing. *Left*: Illustration of a centralized processing scheme in which all the receivers (black dots) have to send data to a fusion center (red dot) via communication links (blue lines). *Right*: Illustration of a distributed processing scheme in which the receivers send data to their neighboring receivers.

least-squares estimation can play a pivotal role in applications for which GNSS data of a *network* of receivers are to be processed. This is the more so, since in GNSS the precision of network-derived solutions is generally much higher than its single-receiver counterpart (Khodabandeh and Teunissen, 2015; Li et al, 2017). With a distributed least-squares setup, single-receiver parameter solutions can achieve precision performances similar to that of their network-derived versions, provided that a sufficient number of iterative communications between the neighboring receivers are established. The importance of such 'collaborative' single-receiver solutions is well appreciated by multi-GNSS services that utilize a large number of low-cost receivers (Li et al, 2015; Odolinski and Teunissen, 2016; Zaminpardaz et al, 2016). With the increase in the number and types of such receivers, many more GNSS users can establish their own measurement setup to determine parameters that suit their needs. These users can therefore potentially deliver high-precision parameter solutions without the need of having a computing center.

This contribution is organized as follows. We first commence with the well-known least-squares objective function to be minimized by a computing center. The stated minimization problem is then formulated into a *constrained* form which is suited for distributed processing. Given such equivalent constrained minimization problem, an attempt to realize distributed processing is discussed by forming the corresponding augmented Lagrangian. We will make use of the 'Alternating Direction Method of Multipliers' (Glowinski and Marroco, 1975; Gabay and Mercier, 1976) and arrive at an iterative scheme for computing least-squares solutions in a distributed manner. By taking recourse to 'geometric' illustrations, the convergence performance of this iterative scheme is studied for a two-dimensional case. For a general case with n number of receivers, the data communication between the receivers is modeled by a 'connected graph' so as to avoid having a fusion center. With such communication graph, we develop an algorithm for distributed least-squares estimation and apply it to a network of 110 receivers tracking GPS triple-frequency signals L1, L2 and L5 to determine estimable satellite code biases. The local single-receiver (but collaborative) code-bias solutions are then compared with the centralized solutions and their convergence under several scenarios is studied.

2 Minimization problems in distributed form

In this section, we give a brief overview of the least-squares method and its corresponding minimization problem. We will then formulate equivalent minimization problems that are suited for distributed processing.

2.1 Least-squares estimation

Consider the sensor nodes $i = 1, \ldots, n$ each collecting observable vectors $y_i \in \mathbb{R}^{m_i}$ to estimate the common unknown parameter vector $x \in \mathbb{R}^p$. The corresponding linear(ized) observation equations read

$$y_i = A_i x + e_i, \quad \text{with} \quad \mathsf{E}(e_i) = 0 \quad \text{and} \quad \mathsf{C}(e_i, e_j) = R_i\, \delta_{ij}, \quad i, j = 1, \ldots, n \tag{1}$$

with δ_{ij} being the Kronecker delta function. The expectation and covariance operators are denoted as $\mathsf{E}(.)$ and $\mathsf{C}(.,.)$, respectively. Thus the observations y_i are assumed mutually uncorrelated, each having the positive-definite variance matrices R_i. We further assume that the system of observation equations (1) is solvable for x, that is

$$\operatorname{rank}(A) = p, \quad \text{where} \quad A = [A_1^T, \ldots, A_n^T]^T \tag{2}$$

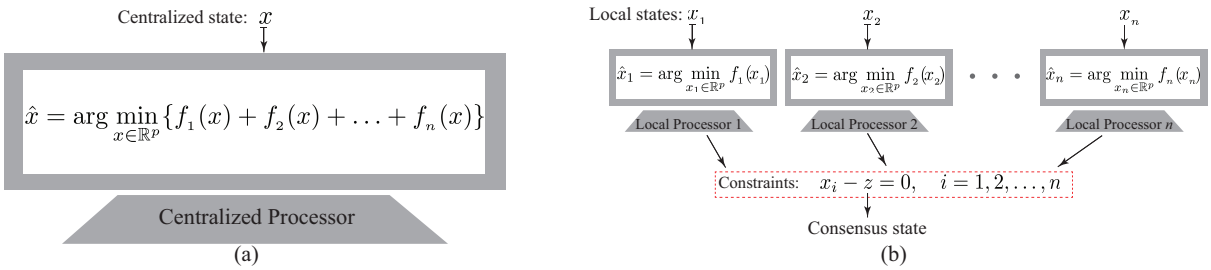

Fig. 2 The minimization problem (3) in centralized form (*left*) and its distributed form (*right*). The centralized state x is replaced by its local versions x_i $(i = 1, \ldots, n)$, while the constraints $z - x_i = 0$ result in the additional consensus state z.

Thus the $m_i \times p$ known design matrices A_i need not be of full-column rank, but their augmented version A.

We choose for the weighted least-squares criterion and take the inverse of the variance matrix of the observations y_i as weight matrix to estimate x. The resultant estimator, say $\hat{x}$, coincides with the so-called Best Linear Unbiased Estimator (BLUE) of x and follows from the minimization problem (Teunissen et al, 2005)

$$\hat{x} = \arg\min_{x\in\mathbb{R}^p} \sum_{i=1}^{n} f_i(x), \quad \text{with} \quad f_i(x) = \frac{1}{2}||y_i - A_i x||^2_{R_i^{-1}} \tag{3}$$

where $||.||^2_{R_i^{-1}} = (.)^T R_i^{-1}(.)$. As a result, the BLUE $\hat{x}$ is obtained through solving the normal equations

$$N\,\hat{x} = r, \quad \text{with} \quad N = \sum_{i=1}^{n} N_i, \quad r = \sum_{i=1}^{n} r_i \tag{4}$$

where the normal matrices and right-hand vectors are, respectively, given by $N_i = A_i^T R_i^{-1} A_i$ and $r_i = A_i^T R_i^{-1} y_i$.

According to (4), one needs to collect normal matrices N_i and right-hand vectors r_i from *all* the sensor nodes $i = 1, \ldots, n$ in order to compute the BLUE $\hat{x}$. This is in fact done by a fusion center. From now on, the solution $\hat{x}$ in (4) is therefore referred to as the 'centralized solution'. Likewise, we call (3) the 'minimization problem in centralized form'.

2.2 Equivalent minimization problems

The centralized solution $\hat{x}$, given in (4), is optimal in a mean-squared-error sense (Teunissen et al, 2005). This is what one would expect as $\hat{x}$ enjoys the advantage of using *all* information about the nodes' data through N_i and r_i $(i = 1, \ldots, n)$. The price one has to pay for this advantage is that data-links between the fusion center and all the n nodes $i = 1, \ldots, n$ are required to be established, a situation that makes data communication and processing power very expensive (particularly for a large number of nodes). The basic idea of distributed processing is to do away with such a stringent requirement by enabling each node to run its own data processing. We therefore as a start consider the case where node i is aimed to estimate x through its own data, that is (compare with 3)

$$\hat{x}_i = \arg\min_{x\in\mathbb{R}^p} f_i(x) \quad \Longrightarrow \quad N_i\,\hat{x}_i = r_i \tag{5}$$

The minimization problem stated above is subject to two restrictions. First, the information content of the individual observation vector y_i is often *not* enough to enable one to determine the solution $\hat{x}_i$, i.e. the normal matrix N_i might be singular. In case of GNSS for instance, a single terrestrial receiver cannot simultaneously track all the satellites revolving around the Earth. As a consequence, *not* all the corresponding satellite-specific biases can be determined by measurements of a single receiver. Second, even if the normal equations (5) contain enough information for the determination of $\hat{x}_i$, the precision of such single-receiver solution is generally shown to be much lower than its network-derived counterparts (Khodabandeh and Teunissen, 2015; Li et al, 2017). In order to tackle these two restrictions, we impose constraints on the parameters, while letting the nodes minimize their own objective functions $f_i(x)$ individually. In doing so, the 'centralized' state x is replaced by its 'local'

versions x_i, but with the equality constraints $x_i - x = 0$ $(i = 1, \ldots, n)$. This yields the following equivalent minimization problems

$$\begin{aligned} \min_{x\in\mathbb{R}^p} \sum_{i=1}^{n} f_i(x) &\equiv \min_{x_i\in\mathbb{R}^p} \sum_{i=1}^{n} f_i(x_i) \quad \text{subject to} \quad x_i - x = 0, \quad i = 1, \ldots, n \\ &\equiv \min_{x_i\in\mathbb{R}^p} f_i(x_i) \qquad\quad \text{subject to} \quad x_i - x = 0, \quad i = 1, \ldots, n \end{aligned} \tag{6}$$

where the symbol '$\equiv$' means 'equivalent to'. The last expression, in (6), follows from the fact that the sum of the nonnegative functions $f_i(x_i)$ is minimized when each individual function $f_i(x_i)$ is minimized. For the sake of notational convenience, from now on we use the symbol z instead of x to express the constraints $x_i - x = 0$. Thus the least-squares solutions for the local states x_i follows from (compare with 5)

$$\hat{x}_i = \arg\min_{x_i\in\mathbb{R}^p} f_i(x_i) \quad \text{subject to} \quad x_i - z = 0, \quad i = 1, \ldots, n, \tag{7}$$

The constraints $x_i - z = 0$ gives the 'consensus' state $z \in \mathbb{R}^p$ as an extra parameter vector. The term 'consensus' is attributed to the role of z upon which the least-squares solutions $\hat{x}_i$ $(i = 1, \ldots, n)$ would *agree* to become equal, i.e. to reach consensus. Note that the minimization problem (7) is *equivalent* to that of (3), while at the same time it has multiple objective functions $f_i(x_i)$. We therefore call (7) the 'minimization problem in distributed form'. A visualization is shown in Figure 2. In fact, many more equivalent minimization problems can be formed by choosing different constraint sets, which all (similar to $x_i - z = 0$) lead to the equalities $x_i = x_j$ $(i, j = 1, \ldots, n)$. Such constraint sets are discussed in Sect. 4.

3 Distributed least-squares estimation

Now that the minimization problem (3) has been expressed in the distributed form (7), we can take the constraints $x_i - z = 0$ into account by forming an *augmented Lagrangian* of (7) which reads (Boyd et al, 2011)

$$\begin{aligned} L_W &= \sum_{i=1}^{n} f_i(x_i) + u_i^T W(x_i - z) + \tfrac{1}{2}||x_i - z||_W^2 \\ &= \sum_{i=1}^{n} f_i(x_i) + \tfrac{1}{2}||x_i - (z - u_i)||_W^2 - \tfrac{1}{2}||u_i||_W^2 \end{aligned} \tag{8}$$

with $||.||_W^2 = (.)^T W(.)$ and $u_i \in \mathbb{R}^p$ being the corresponding Lagrange multipliers. The term 'augmented' is due to the presence of the additional penalty term $(1/2)||x_i - z||_W^2$ in (8). In (ibid) the positive-definite weight matrix $W \in \mathbb{R}^{p\times p}$ takes the special form $W = \rho\, I_p$, i.e. a scaled identity matrix with positive scalar ρ.

According to the Lagrange multiplier rule, the solutions $\hat{x}_i$ in (7) follow by equating the partial derivatives of L_W (denoted as $\partial_{x_i} L_W$, $\partial_z L_W$ and $\partial_{u_i} L_W$) to zero, noting that the least-squares objective functions $f_i(x_i)$ are convex. Solving these three systems of equations for x_i, z and u_i results in the following solutions (Appendix)

$$\hat{x}_i = \hat{z} = \hat{x}, \quad \text{and} \quad W\hat{u}_i = r_i - N_i\,\hat{x}, \quad i = 1, \ldots, n \tag{9}$$

Thus if one 'simultaneously' solves the three systems of equations $\partial_{x_i} L_W = 0$, $\partial_z L_W = 0$ and $\partial_{u_i} L_W = 0$ $(i = 1, \ldots, n)$, the solutions of the local states x_i would then be identical to the centralized solution $\hat{x}$. In distributed processing however, the individual node i only has access to its own systems of equations. As a consequence, the number of unknowns exceeds the number of equations. Now the idea is to let the individual nodes i start with initial values for the unknowns and set up an *iterative* scheme with the intention to obtain improved approximations for $\hat{x}_i$ in each cycle of the iteration. To gain a better insight into the mechanism of such iterative scheme, we first consider the case where the last system of equations (i.e. $\partial_{u_i} L_W = 0$) is absent.

3.1 The method of multipliers: an attempt to realize distributed processing

In the absence of $\partial_{u_i} L_W = 0$ $(i = 1, \ldots, n)$, solving the equations $\partial_{x_i} L_W = 0$ and $\partial_z L_W = 0$ for x_i and z gives a 'class' of solutions upon which the augmented Lagrangian (8) is minimized over x_i and z, while u_i are assumed

given. We denote such a class of solutions by $\hat{x}_i(u)$ and $\hat{z}(u)$ as they are functions of the Lagrange multiplier vector $u = [u_1^T, \ldots, u_n^T]^T$. Likewise, the minimum value of L_W over x_i and z is a function of u which is given by

$$\begin{aligned} g(u) &= \min_{x_1,\ldots,x_n,z} L_W \\ &= \sum_{i=1}^{n} f_i(\hat{x}_i(u)) + \tfrac{1}{2}||\hat{x}_i(u) - (\hat{z}(u) - u_i)||_W^2 - \tfrac{1}{2}||u_i||_W^2 \end{aligned} \tag{10}$$

The function $g(u)$ is referred to as the *dual* function (Boyd and Vandenberghe, 2004). The dual function $g(u)$ provides 'lower bounds' on the optimal value $\sum_{i=1}^{n} f_i(\hat{x}_i)$ as

$$\begin{aligned} \min_{x_1,\ldots,x_n,z} L_W &\leq \min_{x_1=\ldots=x_n=z} L_W \\ &= \min_{x_1=\ldots=x_n} \sum_{i=1}^{n} f_i(x_i) \\ &= \sum_{i=1}^{n} f_i(\hat{x}_i) \end{aligned} \tag{11}$$

The inequality is true since a minimum never gets smaller when adding constraints. The first equality follows by substituting the constraints $x_i = z$ into (8), while the second equality follows from the definition of the least-squares solutions $\hat{x}_i = \hat{x}$ $(i = 1, \ldots, n)$. The lower bound $g(u)$ is, however, not guaranteed to be sharp. Its sharpness is driven by the Lagrange multiplier vector u. Clearly, the maximum value of $g(u)$ delivers the best lower bound on the optimal value $\sum_{i=1}^{n} f_i(\hat{x}_i)$. And fortunately for most 'convex' objective functions $f_i(x_i)$ (such as those of the least-squares method), the maximum value of $g(u)$ *coincides* with the optimal value $\sum_{i=1}^{n} f_i(\hat{x}_i)$ by the maximizer $\hat{u} = [\hat{u}_1^T, \ldots, \hat{u}_n^T]^T$, that is (Boyd and Vandenberghe, 2004)

$$\begin{aligned} g(\hat{u}) &= \max_u g(u) \\ &= \sum_{i=1}^{n} f_i(\hat{x}_i) \end{aligned} \tag{12}$$

Upon comparison of (12) with (10), one can see that the maximizer $\hat{u}$ delivers $\hat{x}_i = \hat{x}(\hat{u})$ and $\hat{z} = \hat{z}(\hat{u})$ which are indeed the sought for least-squares solutions. Thus if the multiplier vector u is initialized by an arbitrary value $u^{[0]}$ giving the corresponding approximate solutions $\hat{x}_i(u^{[0]})$ and $\hat{z}(u^{[0]})$, an iterative scheme can be set up with the intention to increase the dual function $g(u)$ in each cycle of the iteration. As $g(u)$ gets closer to its maximum value, the updated solution for u gets closer to the maximizer $\hat{u}$. Consequently, the approximate solutions $\hat{x}_i(u)$ and $\hat{z}(u)$ tend to the centralized solution $\hat{x}$ over a number of iterations.

We assume for the moment that an *ascent* direction vector of $g(u)$, say $d = [d_1^T, \ldots, d_n^T]^T$, is available in each iteration. Thus $g(u + d) > g(u)$. Such iterative scheme, with k as the iteration index, is then outlined as follows

– Initialize:	$u_1^{[0]}, \ldots, u_n^{[0]}$	
– Set the iteration $k \mapsto 1$		
– Step 1: solve $\partial_{x_i} L_W = 0,\ \partial_z L_W = 0$	for	$x_i^{[k]} = \hat{x}_i(u^{[k-1]}),\ z^{[k]} = \hat{z}(u^{[k-1]})$
– Step 2: update $u_1^{[k-1]}, \ldots, u_n^{[k-1]}$	as	$u_i^{[k]} = u_i^{[k-1]} + d_i^{[k]}$
– Set the iteration $k + 1 \mapsto k$ and go to Step 1		

(13)

To realize the iterative scheme given above, the ascent direction vectors $d_i^{[k]}$ are required to be available. Making use of the definition of the dual function $g(u)$ in (10), the differences $x_i^{[k]} - z^{[k]}$ $(i = 1, \ldots, n)$ can be shown to lie in an ascent direction of $g(u)$ (cf. Appendix). Thus

$$d_i^{[k]} = x_i^{[k]} - z^{[k]}, \quad i = 1, \ldots, n \tag{14}$$

For the least-squares case, the approximate solutions $x_i^{[k]}$ and $z^{[k]}$ can also be expressed by (Appendix)

$$\begin{aligned} x_i^{[k]} &= z^{[k]} - u_i^{[k-1]} + G_i(y_i - A_i[z^{[k]} - u_i^{[k-1]}]), \quad \text{with} \quad G_i = (N_i + W)^{-1} A_i^T R_i^{-1}, \\ z^{[k]} &= (\sum_{i=1}^{n} H_i)^{-1} (\sum_{i=1}^{n} h_i), \quad \text{with} \quad H_i = A_i^T (R_i + A_i W^{-1} A_i^T)^{-1} A_i \end{aligned} \tag{15}$$

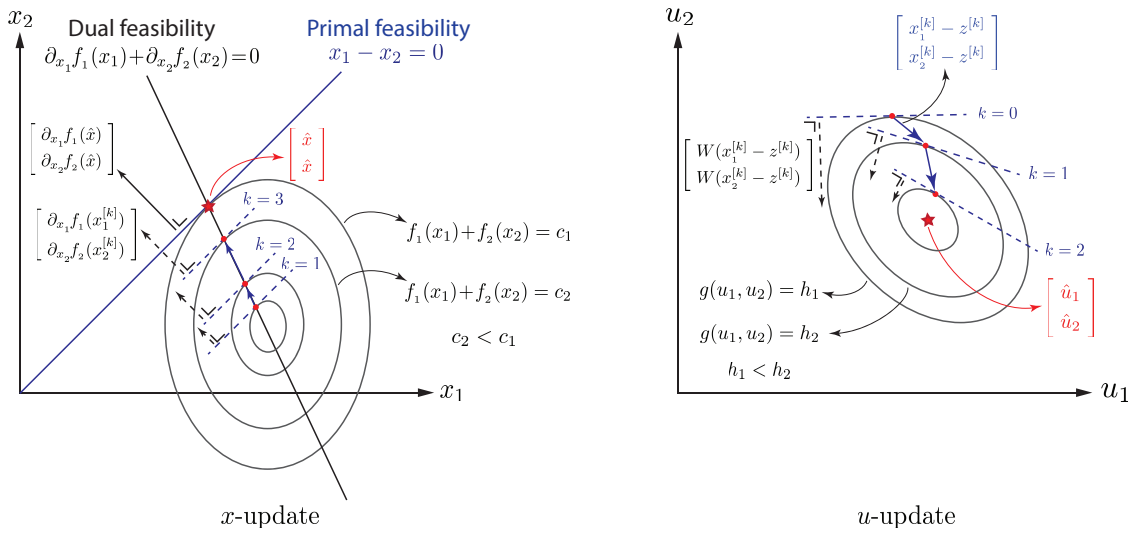

Fig. 3 Two-dimensional visualization of the iterative scheme (13). *Left*: Contour lines of the objective function $f_1(x_1) + f_2(x_2)$ (grey ellipses), the primal feasibility line (blue solid line) and the dual feasibility line (black solid line). The primal feasibility line is 'tangent' to the contour line corresponding to the optimal value $f_1(\hat{x}_1) + f_2(\hat{x}_2)$ and it crosses the dual feasibility line at the least-squares solution $[\hat{x}_1^T, \hat{x}_2^T]^T$ (red star). *Right*: Contour lines of the dual function $g(u)$ (grey ellipses) together with solution $\hat{u} = [\hat{u}_1^T, \hat{u}_2^T]^T$ at the center (red star). Given the approximate values $u^{[k-1]}$ $(k = 1, 2, 3)$ (red dots in the right panel), the approximate solutions $x_1^{[k]}$ and $x_2^{[k]}$ (red dots in the left panel) move towards the least-squares solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$ along the dual feasibility line over a number of iterations.

and $h_i = A_i^T(R_i + A_i W^{-1} A_i^T)^{-1}(y_i + A_i u_i^{[k-1]})$.

The iterative scheme, outlined in (13), follows the 'method of multipliers' (Hestenes, 1969; Powell, 1969). For the least-squares case, a two-dimensional example (i.e. $n = 2$) concerning the mechanism of the method is given in Figure 3. In the left panel of the figure, contour lines of the objective function $f_1(x_1) + f_2(x_2)$ are shown as grey ellipses. The objective function is aimed to be minimized, subject to the constraint $x_1 - x_2 = 0$ that is referred to the 'primal feasibility' (blue solid line). The contour line, that just touches the primal feasibility line, corresponds to the optimal value $f_1(\hat{x}_1) + f_2(\hat{x}_2)$, i.e. the primal feasibility line is 'tangent' to that contour line at the least-squares solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$ (red star). In other words, the gradient of the objective function (denoted by $[\partial_{x_1^T} f_1(x_1), \partial_{x_2^T} f_2(x_2)]^T$) at $[\hat{x}_1^T, \hat{x}_2^T]^T$ is *orthogonal* to the line $x_1 - x_2 = 0$, i.e. $\partial_{x_1} f_1(\hat{x}_1) + \partial_{x_2} f_2(\hat{x}_2) = 0$. For the least-squares case, the equations $\partial_{x_1} f_1(x_1) + \partial_{x_2} f_2(x_2) = 0$ represent a line of points at which the contour lines touch the dashed blue lines parallel to the primal feasibility line. The line $\partial_{x_1} f_1(x_1) + \partial_{x_2} f_2(x_2) = 0$ is referred to the 'dual feasibility' (black solid line) and it crosses the primal feasibility line at the solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$.

In the right panel of Figure 3, contour lines of the dual function $g(u)$ are shown as grey ellipses, with solution $\hat{u} = [\hat{u}_1^T, \hat{u}_2^T]^T$ at the center indicated by a red star. The approximate values $u^{[k-1]}$ $(k = 1, 2, 3)$ are indicated by red dots. At every cycle of the iteration, the approximate solutions $x_1^{[k]}$ and $x_2^{[k]}$ $(k = 1, 2, 3)$ follow from (15), i.e. the red dots in the left panel of the figure. As shown, the solutions $x_1^{[k]}$ and $x_2^{[k]}$ lie in the dual feasibility line as they fulfill the equations $\partial_{x_1} f_1(x_1) + \partial_{x_2} f_2(x_2) = 0$ through $\partial_{x_i} L_W = 0$ and $\partial_z L_W = 0$. Accordingly, the approximate solutions $x_1^{[k]}$ and $x_2^{[k]}$ move towards the least-squares solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$ along the dual feasibility line over a number of iterations. Likewise, the approximate values $u^{[k-1]}$ are updated by the ascent direction vectors $x_i^{[k]} - z^{[k]}$ (blue arrows in the right panel of the figure) so as to become closer to the solution $\hat{u}$.

Although the method of multipliers ensures the convergence of the local solutions $x_i^{[k]}$ $(i = 1, \ldots, n)$ to their centralized version $\hat{x}$, a closer look at its iterative scheme (13) will reveal that the method is *not* suitable for distributed processing. The reason being that in each iteration k, the sensor node i has to have access to all other nodes' data (i.e. y_j, A_j, R_j, $j \neq i$) in order to be able to compute $z^{[k]}$ given in (15). Apart from the computation of $z^{[k]}$ however, each node could *in parallel* initialize and update its own variables x_i and u_i individually, if $z^{[k]}$ would have been given. This suggests a slight modification to the iterative scheme (13), i.e. initialization of the consensus state z by an arbitrary value.

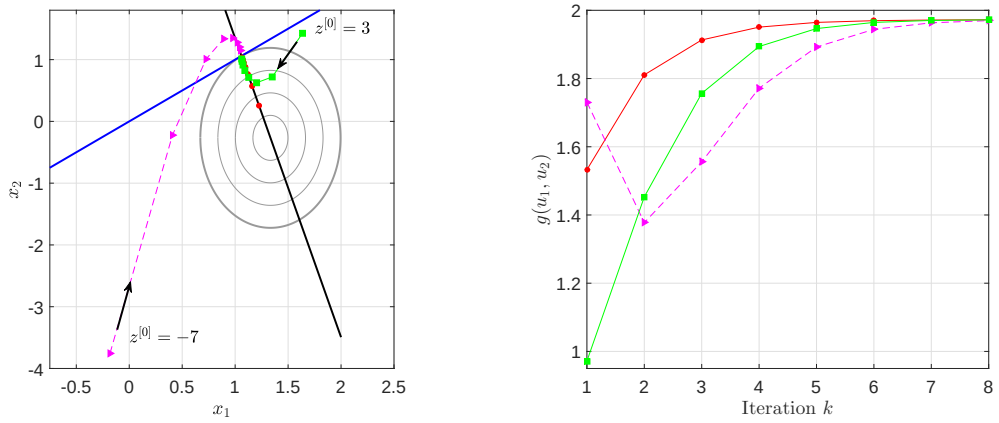

Fig. 4 Two-dimensional visualization of the iterative scheme (16). *Left*: Contour lines of the objective function $f_1(x_1) + f_2(x_2)$ (grey ellipses), the primal feasibility line (blue solid line) and the dual feasibility line (black solid line). The three sets of points represent approximate solutions $x_1^{[k]}$ and $x_2^{[k]}$ for 1) the method of multipliers (red dots), 2) ADMM initialized by $z^{[0]} = 3$ (green squares) and 3) ADMM initialized by $z^{[0]} = -7$ (magenta triangles) over eight iterations $k = 1, \ldots, 8$. *Right*: The corresponding values of the dual function $g(u)$.

3.2 ADMM: a decomposable version of the method of multipliers

We now revisit our earlier iterative scheme (13) and let the consensus state z be initialized by an arbitrary value, say $z^{[0]}$. The modified version of (13) reads then (Appendix)

$$
\begin{aligned}
&\text{– Initialize:} && u_1^{[0]}, \ldots, u_n^{[0]}, z^{[0]} \\
&\text{– Set the iteration } k \mapsto 1 \\
&\text{– Step 1: compute } x_i^{[k]} && \text{as } \; x_i^{[k]} = z^{[k-1]} - u_i^{[k-1]} + G_i(y_i - A_i[z^{[k-1]} - u_i^{[k-1]}]) \\
&\text{– Step 2: compute } z^{[k-1]} && \text{as } \; z^{[k]} = \tfrac{1}{n} \textstyle\sum_{i=1}^{n} x_i^{[k]} + u_i^{[k-1]} \\
&\text{– Step 3: update } u_i^{[k-1]} && \text{as } \; u_i^{[k]} = u_i^{[k-1]} + (x_i^{[k]} - z^{[k]}) \\
&\text{– Set the iteration } k+1 \mapsto k \text{ and go to Step 1}
\end{aligned}
\tag{16}
$$

Compare (16) with its counterpart (13). The expression for the approximate solutions $x_i^{[k]}$ is identical in structure to that of (15), albeit with a different value of $z^{[k-1]}$ as input. Note, in contrast to (13), that the two systems of equations $\partial_{x_i} L_W = 0$ and $\partial_z L_W = 0$ are *not* 'simultaneously' solved for x_i and z. Instead, the first system $\partial_{x_i} L_W = 0$ is solved for x_i, assuming z to be given as $z^{[k-1]}$ (Step 1 in 16). The second system $\partial_z L_W = 0$ is then solved for z, assuming x_i to be given as $x_i^{[k]}$ (Step 2 in 16). As $x_i^{[k]}$ and $z^{[k]}$ are computed in an alternating manner, the iterative scheme (16) follows the so-called 'alternating direction method of multipliers' (ADMM), see (Glowinski and Marroco, 1975; Gabay and Mercier, 1976). For a review of ADMM, see (Boyd et al, 2011).

The mechanism of the method concerning our earlier two-dimensional example is shown in Figure 4 for eight iterations $k = 1, \ldots, 8$. The three sets of points represent approximate solutions $x_1^{[k]}$ and $x_2^{[k]}$ for 1) the method of multipliers (red dots), 2) ADMM initialized by $z^{[0]} = 3$ (green squares) and 3) ADMM initialized by $z^{[0]} = -7$ (magenta triangles). We use the initial values $u_i^{[0]} = 0$ for all the three cases. In case of the method of multipliers, the approximate solutions move towards the least-squares solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$ along the dual feasibility line (black solid line). For the ADMM cases however, the approximate solutions do *not* follow the dual feasibility line, albeit approaching the solutions $[\hat{x}_1^T, \hat{x}_2^T]^T$. This is because the equations $\partial_{x_i} L_W = 0$ and $\partial_z L_W = 0$ are *not* solved simultaneously to fulfill the condition $\partial_{x_1} f_1(x_1) + \partial_{x_2} f_2(x_2) = 0$. For the sake of comparison, the corresponding values of the dual function $g(u)$ over the iterations are also given in the right panel of Figure 4. As shown, the dual function does not necessarily increase at each cycle of the iteration in case of ADMM (magenta triangles). Thus in case of ADMM, the differences $x_i^{[k]} - z^{[k]}$ do *not* necessarily point in an ascent direction of $g(u)$. Nevertheless, the convergence of ADMM over a sufficient number of iterations has been proven for most convex objective functions such as those of the least-squares method (Boyd et al, 2011).

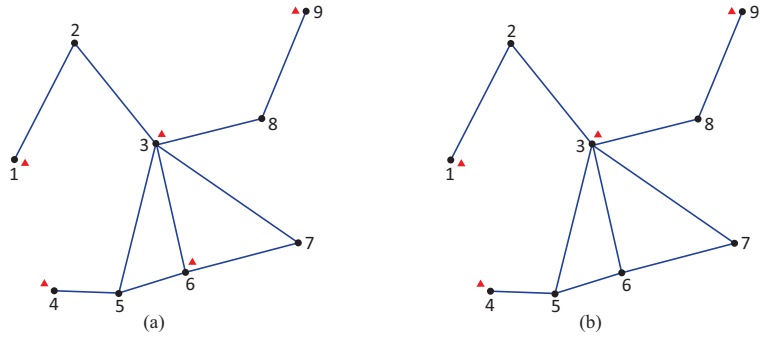

Fig. 5 A communications graph of 9 sensor nodes. The edges (blue lines) represent two-way data links between the nodes (black dots). *Left*: The set $\mathcal{C} = \{1, 3, 4, 6, 9\}$ is chosen such that every edge of the graph has at least one constraint node (red triangles). *Right*: The node 6 is excluded from the set of constraint nodes. However, the new set $\mathcal{C} = \{1, 3, 4, 9\}$ still satisfies the conditions (18).

4 Communication graphs

So far we have not addressed how the consensus state $z^{[k]}$, given in (16), is computed in a distributed manner. Upon the iterative scheme (16), each node i can in parallel compute and update its states $x_i^{[k]}$ and $u_i^{[k]}$ individually. This is what one needs for running a distributed processing setup. However, the individual states $x_i^{[k]}$ and $u_i^{[k-1]}$ are required to be *collected* somewhere in order to compute $z^{[k]}$ as the 'average of their sums'. One may therefore be inclined to conclude that the applicability of (16) is limited to the case where the nodes $i = 1, \ldots, n$, have 'direct' communication connections to one another so as to receive/send their states $x_i^{[k]}$ and $u_i^{[k]}$. So what have we gained? Instead, the nodes could have had a fusion center to compute and provide $z^{[k]}$.

Fortunately, we can do away with the requirement of having a fusion center, if the nodes are linked to each other *at least* through a 'path' so that information can flow from each node to all other nodes. Each node along the path would then play the role of an *agent* transferring information to other nodes. The way the nodes interact with each other to transfer information can be described by a 'communication graph' whose vertices and edges, respectively, represent the nodes and communication links, see e.g. (Jadbabaie et al, 2003; Olfati-Saber and Murray, 2004; Kingston and Beard, 2006).

The graph is said to be *connected* if and only if every node i is linked to all other nodes $j \neq i$ at least through one path. In order for information to flow from each node to all other nodes, we assume that the communication graph of the nodes is connected. An example of such communication graphs with 9 sensor nodes (black dots) is shown in Figure 5. The edges (blue lines) represent two-way data links between the nodes. We define the *neighbors* of node i as those to which the node i has direct data links and we denote them by the set $\mathcal{N}_i$. We also adopt the convention that each node is *its own neighbor* as each node has access to its own data. For instance for the graph in Figure 5, node 1 has only two neighboring nodes as $\mathcal{N}_1 = \{1, 2\}$, whereas node 3 has six neighboring nodes as $\mathcal{N}_3 = \{2, 3, 5, 6, 7, 8\}$. The number of the neighbors of node i (i.e. the cardinality of $\mathcal{N}_i$) is denoted by $|\mathcal{N}_i|$. Thus $|\mathcal{N}_1| = 2$, whereas $|\mathcal{N}_3| = 6$.

4.1 Equivalent constraint sets

According to (16), the consensus state $z^{[k]}$ follows as the 'average' of the sums $x_i^{[k]} + u_i^{[k-1]}$ $(i = 1, \ldots, n)$, thereby requiring a fusion center. This is because only one consensus state is commonly defined across the nodes $i = 1, \ldots, n$. The idea is now to define *multiple* consensus states across the nodes and yet the corresponding constraint set leads to the equalities $x_i = x_j$ $(i, j = 1, \ldots, n)$. As stated previously in Sect. 2, there are many (in fact infinite) constraint sets equivalent to the original constraint set $x_i - z = 0$ $(i = 1, \ldots, n)$. To form such constraint sets, let $\mathcal{C}$ be a subset of the nodes, i.e. $\mathcal{C} \subset \{1, \ldots, n\}$, for which consensus states are to be defined. We call the members of $\mathcal{C}$ the *constraint nodes* and indicate them by index s. For every constraint node $s \in \mathcal{C}$, we define the consensus state $z_s \in \mathbb{R}^p$ imposing the constraints

$$x_i - z_s = 0, \quad i \in \mathcal{N}_s \tag{17}$$

Distributed least-squares estimation

• **Initialization (every node i)**

– Compute its own G-matrix: $G_i = (A_i^T R_i^{-1} A_i + W)^{-1} A_i^T R_i^{-1}$

– Set its own x- and u-states to arbitrary values: $x_i^{[0]}$, $u_{i,s}^{[-1]}$ for $s \in \mathcal{N}_i$

• **Initialization (every constraint node s)**

– Set its own z-states to arbitrary values: $z_s^{[0]}$

– Send $z_s^{[0]}$ to its neighboring nodes $i \in \mathcal{N}_s$

Set $k \mapsto 1$

• **Step 1) u- and x-updates (every node i)**

– u-update: $u_{i,s}^{[k-1]} = u_{i,s}^{[k-2]} + (x_i^{[k-1]} - z_s^{[k-1]})$ for $s \in \mathcal{N}_i$

– compute $l_i^{[k-1]} = \dfrac{1}{|\mathcal{N}_i|} \sum\limits_{s \in \mathcal{N}_i} (z_s^{[k-1]} - u_{i,s}^{[k-1]})$

– x-update: $x_i^{[k]} = l_i^{[k-1]} + G_i (y_i - A_i\, l_i^{[k-1]})$

– send $z_{i,s}^{[k]} = x_i^{[k]} + u_{i,s}^{[k-1]}$ to its own constraint nodes $s \in \mathcal{N}_i$

• **Step 2) z-update (every constraint node i)**

– z-update: $z_s^{[k]} = \dfrac{1}{|\mathcal{N}_s|} \sum\limits_{i \in \mathcal{N}_s} z_{i,s}^{[k]}$

– send $z_s^{[k]}$ to its neighboring nodes $i \in \mathcal{N}_s$

Set $k+1 \mapsto k$

Go to Step 1

Fig. 6 Algorithmic steps of distributed least-squares estimation, given the observation vectors y_i, design matrices A_i, variance matrices R_i $(i = 1, \ldots, n)$ and weight matrix W as input.

Thus the local states of all 'neighbors' of $s \in \mathcal{C}$ are constrained to be equal, i.e. $x_i = x_j$ $(i, j \in \mathcal{N}_s)$. According to (17), the set $\mathcal{C}$ must be chosen such

$$\begin{aligned} &1): \ \bigcup_{s \in \mathcal{C}} \mathcal{N}_s = \{1, \ldots, n\}, \\ &2): \ \forall\ s_1, s_2 \in \mathcal{C} \text{ with } \mathcal{N}_{s_1} \bigcap \mathcal{N}_{s_2} = \emptyset, \ \ \exists\ s_3 \in \mathcal{C}: \ \mathcal{N}_{s_1} \bigcap \mathcal{N}_{s_3} \neq \emptyset, \ \mathcal{N}_{s_2} \bigcap \mathcal{N}_{s_3} \neq \emptyset \end{aligned} \tag{18}$$

so as to fulfill the equalities $x_i = x_j$ $(i, j = 1, \ldots, n)$. The first condition states that *every* node i must have at least one constraint node s as its neighbor to ensure that its local state is equal to one of the consensus states, i.e. $x_i = z_s$. The second condition states that the neighbors of every two constraints nodes s_1 and s_2 must have either overlaps or neighbors of *another* constraint node s_3 in common. Thus on the one hand, we have $x_i = z_{s_1}$ and $z_{s_1} = z_{s_3}$ for every node i as a neighbor of s_1. On the other hand, we have $x_j = z_{s_2}$ and $z_{s_2} = z_{s_3}$ for every node j as a neighbor of s_2. The two conditions together imply that $x_i = z_{s_1} = z_{s_2} = x_j$ $(i, j = 1, \ldots, n)$. Similar conditions concerning the choice for constraint sets are originally given by Schizas et al (2008).

One rather trivial choice that can satisfy (18) is $\mathcal{C} = \{1, \ldots, n\}$, i.e. choosing all the n nodes as the constraint nodes, thus defining n consensus states z_{s_i} $(i = 1, \ldots, n)$. Alternatively, the set $\mathcal{C}$ can be chosen such that every edge of the communication graph has at least one constraint node. In the left panel of Figure 5, such a choice of constraint nodes are indicated by red triangles. As shown in the right panel of the figure, one can exclude node 6 from the set of constraint nodes and yet obtain a new set $\mathcal{C}$ satisfying the conditions (18). In the following it becomes clear why such a choice is *not* favorable and which choice would be considered 'best'.

4.2 An algorithm for distributed least-squares estimation

Given a choice for constraint nodes $\mathcal{C}$, our earlier constraints $x_i - z = 0$ $(i = 1, \ldots, n)$ are replaced by $x_i - z_s = 0$ $(i \in \mathcal{N}_s,\ s \in \mathcal{C})$. Likewise, the augmented Lagrangian, given in (8), takes the following form

$$\begin{aligned} L_W &= \sum_{i=1}^{n} f_i(x_i) + \sum_{s\in\mathcal{C}} \sum_{i\in\mathcal{N}_s} u_{i,s}^T W(x_i - z_s) + \tfrac{1}{2}||x_i - z_s||_W^2 \\ &= \sum_{i=1}^{n} f_i(x_i) + \sum_{s\in\mathcal{C}} \sum_{i\in\mathcal{N}_s} \tfrac{1}{2}||x_i - (z_s - u_{i,s})||_W^2 - \tfrac{1}{2}||u_{i,s}||_W^2 \end{aligned} \tag{19}$$

Thus the Lagrange multipliers $u_i \in \mathbb{R}^p$ $(i = 1, \ldots, n)$ are replaced by their new versions $u_{i,s} \in \mathbb{R}^p$ $(i \in \mathcal{N}_s,\ s \in \mathcal{C})$.

With the augmented Lagrangian (19), we apply the iterative scheme (16) and arrive at an algorithm for distributed least-squares estimation (see Figure 6). As shown in the figure, the node i only communicates with its own constraint nodes $s \in \mathcal{N}_i$ by providing the sums $x_i^{[k]} + u_{i,s}^{[k-1]}$. On the other hand, the constraint node s only communicates to its own neighboring nodes $i \in \mathcal{N}_s$ by providing the averages $z_s^{[k]}$. Therefore, the requirement of having a fusion center among all nodes is relaxed. We can now address why the choice $\mathcal{C} = \{1, 3, 4, 9\}$ in Figure 5 is not favorable. Upon this choice, the data links (i.e. edges) (5,6) and (6,7) become *obsolete*, since there is no constraint node defined among the nodes 5, 6 and 7 to communicate with. As this choice satisfies the conditions (18) and thus the equalities $x_i = x_j$ $(i, j = 1, \ldots, n)$, the convergence of the local solutions $x_i^{[k]}$ to the centralized solution $\hat{x}$ is guaranteed. However, the associated convergence rate would *never* be higher than the one corresponding to the choice $\mathcal{C} = \{1, 3, 4, 6, 9\}$, as use is made of *fewer* data links between the nodes. Would one choose the 'maximization' of communication links between the nodes as 'optimality criterion', the 'best' choice for the set $\mathcal{C}$ is then the one upon which every edge has at least one constraint node. With this choice, one takes full advantage of the existing data links among the sensor nodes.

4.3 On the computation of the local states

Consider the computations involved in the algorithm presented in Figure 6. Apart from matrix inversions that are required for the G-matrix computation at the initialization step, each node only needs to carry out simple mathematical operations (e.g. addition and averaging). From a computational point of view, this makes the distributed least-squares algorithm very demanding. As to the G-matrix computation, consider the following matrix identity (Teunissen et al, 2005, p. 188)

$$\begin{aligned} G_i &= (A_i^T R_i^{-1} A_i + W)^{-1} A_i^T R_i^{-1} \\ &= W^{-1} A_i^T (R_i + A_i W^{-1} A_i^T)^{-1} \end{aligned} \tag{20}$$

The first expression is suited for the case where the inverse-matrix R_i^{-1} is given as input and when the inversion of the $p \times p$ matrix $A_i^T R_i^{-1} A_i + W$ does *not* require so much computational effort. This is particularly the case when there are fewer unknowns x than the individual observations y_i, i.e. when $p < m_i$. The second expression is suitable when matrices R_i and W^{-1} are given as input, while the inversion of the $m_i \times m_i$ matrix $R_i + A_i W^{-1} A_i^T$ would be more straightforward than that of $A_i^T R_i^{-1} A_i + W$, e.g. when the individual observation vector y_i has fewer entries than the unknowns x (i.e. $m_i < p$).

Instead of computing the G-matrix explicitly, one can take an alternative approach by solving the linear system of equations concerning the x-update in Figure 6. To this end, the term $\delta = G_i(y_i - A_i\, l_i^{[k-1]})$ in the x-update $x_i^{[k]} = l_i^{[k-1]} + \delta$ can be interpreted as the unknowns of the following system of equations

$$(A_i^T R_i^{-1} A_i + W)\, \delta = A_i^T R_i^{-1} (y_i - A_i\, l_i^{[k-1]}) \tag{21}$$

Since matrix $(A_i^T R_i^{-1} A_i + W)$ is symmetric and positive-definite, it can be written as the product LL^T (i.e. its Cholesky decomposition), where L is a lower triangular matrix. The term δ is then computed relatively easy by solving the system using 'forward-backward-substitution'. Accordingly, the lower triangular system $L\, \delta_L = A_i^T R_i^{-1} (y_i - A_i\, l_i^{[k-1]})$ is solved by forward substitution, followed by solving the upper triangular system $\delta_L = L^T \delta$ through backward substitution, see e.g. (Teunissen et al, 2005). Another alterative approach is to formulate the algorithm in 'information form' so as to eliminate the burden of inversions, see e.g. (Khodabandeh et al, 2018).

Next to the G-matrix computation, the initial values $x_i^{[0]}$, $z_s^{[0]}$ and $u_{i,s}^{[-1]}$ have to be set. Although these values can be chosen in an arbitrary fashion, a good choice for these values can—as the following section will show—expedite the convergence of the local solutions $x_i^{[k]}$. The initial values are preferred to be as close as possible to the solutions $\hat{x}$ and $\hat{u}$. With regard to the second expression of (9), the solution for u_i can be expressed as

$$W\hat{u}_i = A_i^T R_i^{-1}\hat{e}_i \quad \text{with} \quad \hat{e}_i = y_i - A_i\hat{x}, \quad i = 1, \ldots, n \tag{22}$$

This shows that $\hat{u}_i$ are linear functions of the least-squares residuals $\hat{e}_i$. If one has any confidence in one's measurement model $\mathsf{E}(y_i) = A_i x$, one would like the least-squares residuals $\hat{e}_i$ to be as close as possible to zero. This suggests that the Lagrange multipliers are preferred to be initialized by $u_{i,s}^{[-1]} = 0$. In case of $x_i^{[0]}$ and $z_s^{[0]}$ however, a good choice for initial values relies to a large extent on experience from past experiments. For instance, if parameters x behave rather stable over time, solutions of previous experiments are appropriate candidates for the initial values $x_i^{[0]} = z_s^{[0]}$.

The linearized case. The algorithm, in Figure 6, holds for the *linear* measurement models $\mathsf{E}(y_i) = A_i x_i$ subject to $x_i - z_s = 0$ ($i \in \mathcal{N}_s,\ s \in \mathcal{C}$). Now consider the case where the linear maps $A_i x_i$ are replaced by the *nonlinear* maps $A_i(x_i)$ ($A_i : \mathbb{R}^p \mapsto \mathbb{R}^{m_i}$). In that case, one can still apply the algorithm to the *linearized* versions of the measurement models $\mathsf{E}(y_i) = A_i(x_i)$. Given an approximate value for x_i, say x_i^o, the role of the observation vector y_i and parameter vector x_i is taken by the 'increments' $\Delta y_i = y_i - A_i(x_i^o)$ and $\Delta x_i = x_i - x_i^o$, respectively. As a consequence, the constraints $\Delta x_i - z_s = 0$ ($i \in \mathcal{N}_s,\ s \in \mathcal{C}$) must holds. But this implies that the nodes must have 'identical' approximate values, i.e. $x_i^o = x^o$, since $x_i = x_j$ $(i, j = 1, \ldots, n)$. Under this condition, the linearized version of distributed least-squares estimation follows by *repeated* application of the algorithm in Figure 6. First, the nodes linearize their measurement models by the same approximate value x^o. Given the resultant linearized models, the algorithm in Figure 6 is then applied with the initial values $\Delta x_i^{[0]} = z_s^{[0]} = 0$. After a sufficient number of iterations k, the local solutions $\Delta x_i^{[k]}$ reach consensus, i.e. $\Delta x_i^{[k]} \approx \Delta x_1^{[k]}$ $(i = 2, \ldots, n)$. Each node would then update the approximate value as $(x^o + \Delta x_i^{[k]}) \mapsto x^o$. Given this new approximate value, the nodes again linearize their measurement models, set the initial values $\Delta x_i^{[0]} = z_s^{[0]} = 0$ and apply the algorithm. This procedure is repeated until the nodes consider their solutions converged and decide to stop the iteration, a decision which is largely driven by the nonlinearity of the measurement models and the approximate value x^o (Teunissen, 1990).

The nonlinear case. As an alternative approach to the linearized case discussed above, one might think of the case where each node *individually* solves its own nonlinear minimization problem. As shown through (31) given in Appendix, the x-update in Figure 6 follows by solving a *regularized* least-squares problem. This implies, in case the maps $A_i(x_i)$ ($A_i : \mathbb{R}^p \mapsto \mathbb{R}^{m_i}$) are nonlinear, that the distributed least-squares algorithm may still be employed, provided that the corresponding objective function $\sum_{i=1}^n f_i(x_i)$ is assumed to satisfy the ADMM and nonlinearity convergence conditions. Under this assumption, the x-update is generalized as

$$x\text{-update:} \quad x_i^{[k]} = \arg\min_{x_i \in \mathbb{R}^p} \left\{ ||y_i - A_i(x_i)||^2_{R_i^{-1}} + ||l_i^{[k-1]} - x_i||^2_W \right\} \tag{23}$$

The above equation represents a 'regularized nonlinear least-squares problem'. The presence of the regularization term $||l_i^{[k-1]} - x_i||^2_W$ is due to the vector $l_i^{[k-1]}$ that is known through $u_{i,s}^{[k-1]}$ and $z_s^{[k-1]}$ (cf. Figure 6).

5 GNSS code-bias determination

This section is intended to demonstrate how distributed least-squares estimation, discussed previously, can play a pivotal role in applications for which GNSS data of a *network* of receivers are to be processed. In a GNSS network setup, each receiver serves as a sensor node for receiving observables from visible GNSS satellites to determine a range of different parameters such as positions and velocities, atmospheric delays, timing and instrumental biases, see e.g. (Hofmann-Wellenhof et al, 2008; Teunissen and Montenbruck, 2017). As an illustrative example, here we restrict our focus to the determination of instrumental biases that are experienced as *delays* on GNSS pseudo-range measurements. These delays are referred to as *code biases* and often behave rather stable over time, see e.g. (Schaer, 1999; Zhang and Teunissen, 2015; Nadarajah et al, 2018).

5.1 Data-sets and measurement models

Let $\bar{y}_{i,j} \in \mathbb{R}^{m_i}$ contain pseudo-range measurements on frequency j $(j = 1, 2, 3)$ that are collected by receiver i from m_i visible satellites. Estimable combinations of the code biases can then be shown to be determined by the

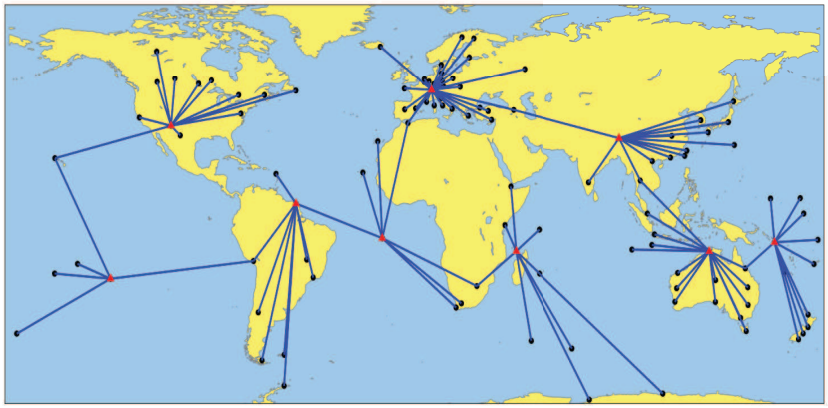

Fig. 7 A communications graph of 110 GNSS receivers distributed over the globe (IGS stations). The edges (blue lines) represent two-way data links between the receivers (black dots). The 9 red triangles indicate the set of constraint nodes.

following observation equations (Khodabandeh and Teunissen, 2015)

$$\begin{aligned} \mathsf{E}(y_1) &= A_1\, x, \quad \text{with} \quad y_1 = \bar{y}_{1,3} - \bar{y}_{1,IF} - \mu_3\, \bar{y}_{1,GF} \\ \mathsf{E}(y_i) &= A_i\, x, \quad \text{with} \quad y_i = D_{m_i}^T(\bar{y}_{i,3} - \bar{y}_{i,IF} - \mu_3\, \bar{y}_{i,GF}),\ i = 2, \ldots, n \end{aligned} \tag{24}$$

in which $x \in \mathbb{R}^p$ contains the unknown (but estimable) satellite code biases. Thus here p is the total number of satellites that are tracked by the receivers $i = 1, \ldots, n$. The 'ionosphere-free' (IF) and 'geometry-free' (GF) combinations of the first two-frequencies $j = 1, 2$ are given by

$$\begin{aligned} \bar{y}_{i,IF} &= \tfrac{1}{\mu_2 - \mu_1}(\mu_2\, \bar{y}_{i,1} - \mu_1\, \bar{y}_{i,1}) \\ \bar{y}_{i,GF} &= \tfrac{1}{\mu_2 - \mu_1}(\bar{y}_{i,2} - \bar{y}_{i,1}) \end{aligned} \tag{25}$$

The frequency-dependant coefficients are defined as $\mu_j = (f_1/f_j)^2$, with f_j being the frequency band ($j = 1, 2, 3$). The $m_i \times (m_i - 1)$ matrix D_{m_i} is the 'between-satellite' differencing operator. For instance when $m_i = 3$ or $m_i = 4$, it takes the following forms

$$D_3^T = \begin{bmatrix} -1 & +1 & 0 \\ -1 & 0 & +1 \end{bmatrix}, \quad D_4^T = \begin{bmatrix} -1 & +1 & 0 & 0 \\ -1 & 0 & +1 & 0 \\ -1 & 0 & 0 & +1 \end{bmatrix} \tag{26}$$

Due to the linear dependence that exists between the design matrices of the code biases, we choose for an specific $\mathcal{S}$-basis (Baarda, 1973; Teunissen, 1985) upon which the code biases of the first receiver are lumped with the satellite code biases. The design matrices would then read $A_1 = \bar{A}_1$ and $A_i = D_{m_i}^T \bar{A}_i$ ($i \neq 1$), where the $m_i \times p$ matrix $\bar{A}_i$ is formed by choosing those rows of the identity matrix I_p which correspond to the satellites visible to receiver i ($i = 1, \ldots, n$).

A GPS data-set of 110 IGS stations are used (cf. Figure 7) to compute the estimable satellite code biases x. We only consider the GPS block IIF satellites (currently 12 satellites) so as to collect pseudo-range measurements on the three frequencies L1 ($j = 1$), L2 ($j = 2$) and L5 ($j = 3$). These pseudo-range measurements, i.e. $\bar{y}_{i,j}$ in (24), are assumed to be mutually uncorrelated with a zenith-referenced standard-deviation of 20 cm. To model the dependency of the measurements on the satellites' elevation, we take the elevation-dependent weighting function as given in (Euler and Goad, 1991).

For the sake of comparison, we first compute centralized solutions of the estimable satellite code biases. Examples of such solutions (solid lines), together with their 3-sigma confidence intervals (dashed lines) are shown in Figure 8. It can be seen that the solutions behave rather stable over time. Kalman filtering is used to compute the solutions over time. Thus the solution corresponding to an epoch is obtained on the basis of the measurements collected up to and including that epoch. That is why the 3-sigma confidence intervals decrease as the number of epochs increases.

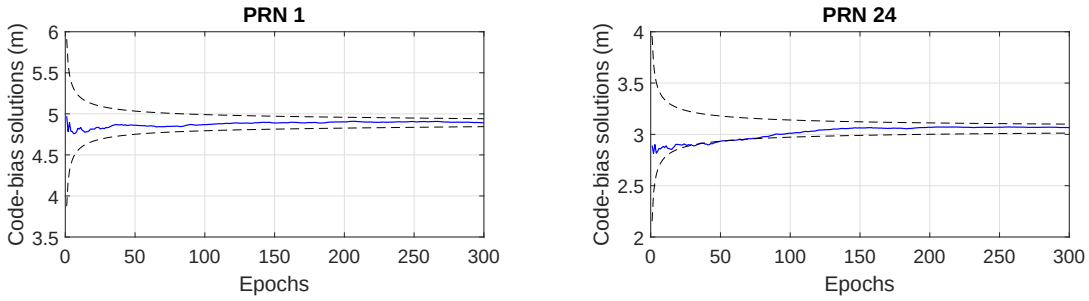

Fig. 8 Centralized solutions of the estimable code biases (solid lines) of PRN 1 (left) and PRN 24 (right), together with their 3-sigma confidence intervals (dashed lines) over time. The results correspond with a GPS L1/L2/L5 data-set of the receivers shown in Figure 7, 10 January 2018, interval 00:00:00 – 02:30:00 UTC.

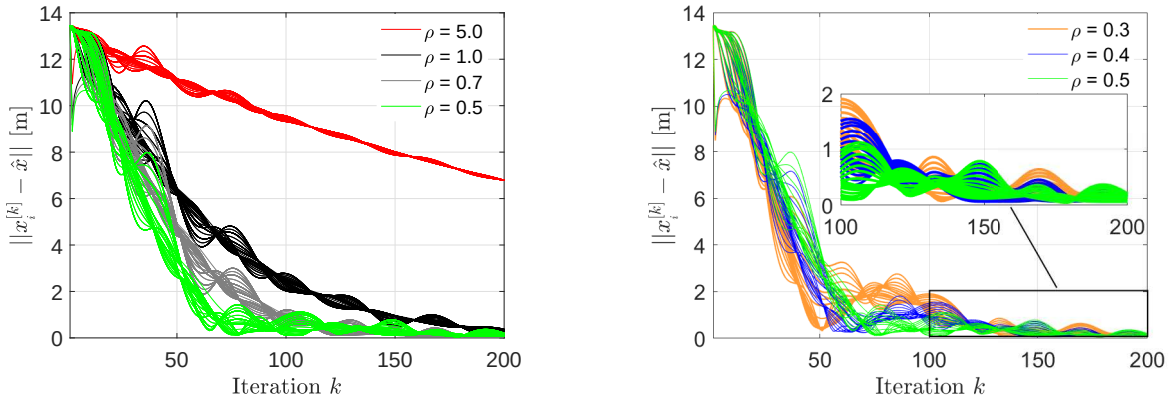

Fig. 9 The norm-differences $||x_i^{[k]} - \hat{x}||$ $(i = 1, \ldots, 110)$ as a function of the iteration number k for different choices of the weight matrix $W = \rho\, I_p$, given the initial values $x_i^{[0]} = z_s^{[0]} = 0$. Each colour indicates 110 curves corresponding to a specific value for ρ.

5.2 Convergence of distributed least-squares estimation

To show the role played by the weight matrix W and the initial values $x_i^{[0]}$ and $z_s^{[0]}$ in the convergence of the local solutions $x_i^{[k]}$ $(i = 1, \ldots, n)$, a *weakly connected* communication graph for the 110 receivers (i.e. $n = 110$) is established. As shown in Figure 7, only 118 data links (edges) between the receivers are considered, whereas the maximum number of potential data links is $n(n-1)/2 = 5995$. Accordingly, only 9 constraint nodes (red triangles) are required so that every edge has at least one constraint node. Similar to $u_i^{[-1]}$, we first set the initial values $x_i^{[0]}$ and $z_s^{[0]}$ to zero and compute the 'magnitude' of the differences $x_i^{[k]} - \hat{x}$ over the number of iteration k. Figure 9 shows the norm-differences $||x_i^{[k]} - \hat{x}||$ $(i = 1, \ldots, 110)$ as a function of k for different choices of the weight matrix $W = \rho\, I_p$. When we give more weight to the initial values $x_i^{[0]} = z_s^{[0]} = 0$ (e.g. $\rho = 5.0$), the norm-differences (red lines) very slowly converge to zero (left panel of the figure). This is the case as the initial values $x_i^{[0]} = z_s^{[0]} = 0$ are far different from the actual solutions $\hat{x}$ (consider the magenta triangles in Figure 4 as their two-dimensional analogue). Decreasing the weight to $\rho = 0.5$, the local solutions rely more on the actual measurements, thus converging faster to the centralized solution. As shown in the right panel of the figure however, a further decrease of ρ does *not* necessarily expedite the convergence (blue and orange curves in the left panel). This is the due to the fact that measurements of the receivers do *not* contain all the information required to obtain the centralized solution $\hat{x}$ (i.e. their normal matrices N_i are singular.) Therefore, the regularization parameter ρ is needed so as to allow the receivers to communicate their information to one another. For a discussion on the optimal values of ρ to achieve fast convergence, see e.g. (Ghadimi et al, 2015).

Next to different choices of the weight matrix W, we also compute the norm-differences $||x_i^{[k]} - \hat{x}||$ for different initial values $x_i^{[0]} = z_s^{[0]}$, given the weight matrix $W = 0.5\, I_p$ (see Figure 10). As illustrated, the solutions $x_i^{[k]}$

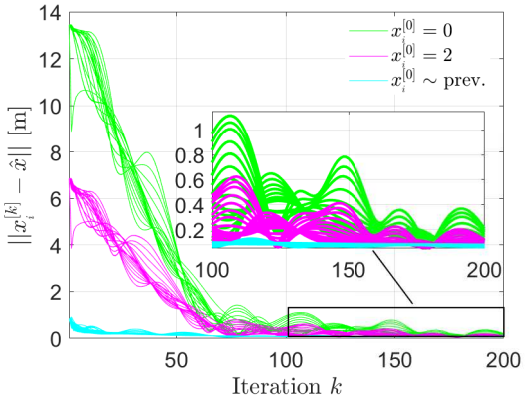

Fig. 10 The norm-differences $||x_i^{[k]} - \hat{x}||$ $(i = 1, \ldots, 110)$ as a function of the iteration number k for different initial values $x_i^{[0]} = z_s^{[0]}$, given the weight matrix $W = 0.5\, I_p$. Each colour indicates 110 curves corresponding to a specific value for $x_i^{[0]}$. The symbol '$x_i^{[0]} \sim$ prev.' refers to the case where solutions of a previous experiment are taken for the initial values $x_i^{[0]}$.

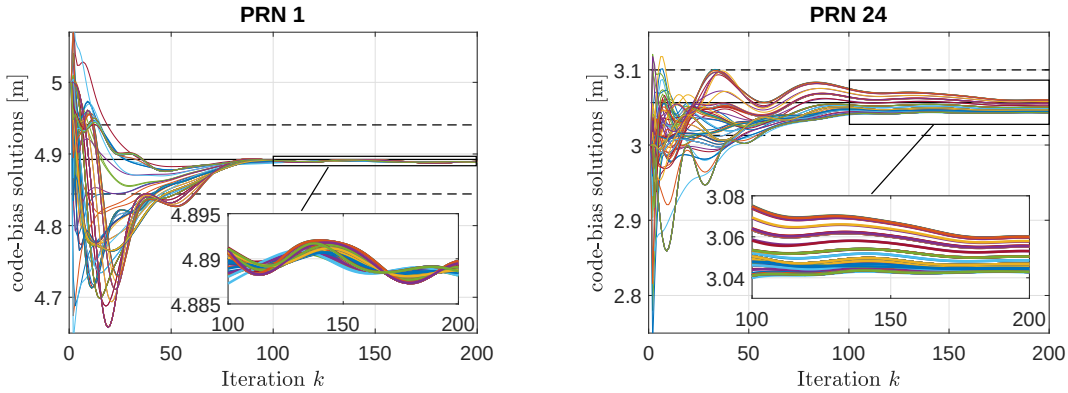

Fig. 11 The individual differences $x_i^{[k]} - \hat{x}$ $(i = 1, \ldots, 110)$ as a function of the iteration number k (colored lines), given the initial values '$z^{[0]} = x_i^{[0]} \sim$ prev.' and the weight matrix $W = 0.5\, I_p$. The 3-sigma confidence intervals of the centralized solutions $\hat{x}$ are indicated by the dashed lines. Each colour corresponds to a different receiver.

converge faster to their centralized counterpart $\hat{x}$, when the initial values $x_i^{[0]} = z_s^{[0]}$ are close to $\hat{x}$ (consider the green squares in Figure 4 as their two-dimensional analogue).

To conclude this section, the convergence dependency on the 'strength' underlying the observation equations $\mathsf{E}(y_i) = A_i\, x$ $(i = 1, \ldots, n)$ is briefly considered. To that end, the local code-bias solutions of a satellite that is tracked by most receivers (e.g. PRN 1) are compared with that of a satellite that is tracked by fewer receivers (e.g. PRN 24) during the observational time-span. The corresponding individual differences $x_i^{[k]} - \hat{x}$ (colored lines) are shown in Figure 11. Due to a better visibility, the measurement models $\mathsf{E}(y_i) = A_i\, x$ contain more information concerning the code-bias parameter of PRN 1. As a result, the corresponding local code-bias solutions converge faster to their centralized counterpart than those of PRN 24. This shows, next to W, $x_i^{[0]}$ and $z_s^{[0]}$, that measurement model's strength does also play a role in the convergence of the local solutions.

6 Conclusions and future outlook

In this contribution we developed an algorithm for distributed least-squares estimation. In this regard, we commenced with the well-known least-squares minimization problem and formulated it into a form that is well-suited for distributed processing (cf. Figure 2). Accordingly, the unknown parameter vector x, i.e. the centralized state, was replaced by multiple parameter vectors x_i $(i = 1, \ldots, n)$, i.e. local states. Equality constraints $x_i - z$ were then imposed on the problem at the cost of having the consensus state z as an extra unknowns. By forming the corresponding augmented Lagrangian, it was shown why the method of multipliers is not suited for distributed

processing (cf. 13 and Figure 3). The reason being that in each iteration, each receiver has to have access to all other receivers' data to be able to compute solutions for the consensus state z . With a slight modification to the method of multipliers, i.e. initialization of the consensus state z by an arbitrary value, the ADMM iterative scheme follows with which each receiver can *in parallel* initialize and update its own states x_i and u_i individually (cf. 16 and Figure 4).

To avoid having a fusion center, the receivers (nodes) are assumed to be linked to each other *at least* through a 'path' so that information can flow from each receiver to all other receivers. Accordingly, the data links between the receivers are modeled by a 'connected graph'. A subset of the receivers were then chosen as the constraints nodes s of the graph, each representing its own consensus state z_s. With multiple consensus states among the receivers, the algorithmic steps of distributed least-squares estimation (Figure 6) follow as a generalization of the iterative scheme (16). The algorithm was applied to a network of 110 receivers tacking GPS triple-frequency signals L1, L2 and L5 to determine estimable satellite code biases. It was shown that the local single-receiver (but collaborative) code-bias solutions converge to their centralized network-derived counterparts, after a sufficient number of iterative communications between the receivers. The role taken by 1) the weight matrix, 2) the initial values and 3) the strength of the underlying measurement model in the local solutions' convergence was also highlighted.

In this contribution, attention was focused on the distributed computation of the solution only. Addressing open research questions such as 'how to compute the *precision* of the solution in a distributed manner' or 'how a distributed *quality control* procedure for validating the solution looks like' are topics of future works.

Acknowledgements The second author is the recipient of an Australian Research Council (ARC) Federation Fellowship (project number FF0883188). This support is gratefully acknowledged. We also wish to thank the IGS for providing GNSS data of the stations.

Appendix

In this appendix, the differentiation rules $\partial_x\, x^T W y = W y$ and $\partial_x\, ||x||^2_W = 2\, W x$ are frequently applied.

Proof of (9). Equating the partial derivatives of the Lagrangian (8) to zero gives

$$\begin{aligned} \partial_{x_i} L_W &= (N_i \hat{x}_i - r_i) + W(\hat{x}_i - \hat{z} + \hat{u}_i) = 0, \quad i = 1, \ldots, n \\ \partial_{z_i} L_W &= -\sum_{i=1}^{n} W(\hat{x}_i - \hat{z} + \hat{u}_i) = 0, \\ \partial_{z_i} L_W &= W(\hat{x}_i - \hat{z}) = 0, \quad i = 1, \ldots, n \end{aligned} \tag{27}$$

The first set of equations follows from the equality $\partial_x f_i(x) = N_i\, x - r_i$ for $f_i(x) = (1/2)||y_i - A_i\, x||^2_{R_i^{-1}}$. Summation of the first set of equations over $i = 1, \ldots, n$, together with the second set of equations, gives

$$\sum_{i=1}^{n} N_i \hat{x}_i - r_i = 0 \tag{28}$$

From the third set of equations (27), it follows that $\hat{x}_i = \hat{z}$ $(i = 1, \ldots, n)$. Substitution into (28), together with (4), gives the equalities $\hat{x}_i = \hat{z} = \hat{x}$. The equalities $W \hat{u}_i = r_i - N_i\, \hat{x}$ follow by substituting $\hat{x}_i = \hat{z} = \hat{x}$ into the first set of equations (27). □

Proof of (14). An ascent direction of $g(u)$ can be characterized by $d = Q\, \partial_u g(u)$, with Q being an arbitrary positive-definite matrix and $\partial_u g(u)$ being the gradient vector (Teunissen, 1990). To have short-hand expressions for $g(u)$ and $\partial_u g(u)$, we use the auxiliary notations $v = [x_1^T, \ldots, x_n^T, z^T]^T$ and $\hat{v} = [\hat{x}_1^T(u), \ldots, \hat{x}_n^T(u), \hat{z}^T(u)]^T$. Thus (10) can be expressed as $g(u) = L_W(\hat{v}, u)$. Application of the chain rule of differentiation to (10) gives then

$$\partial_{u_i} g(u) = \partial_{u_i} \hat{v}\, \underbrace{\partial_v L_W(\hat{v}, u)}_{0} + \underbrace{\partial_{u_i} L_W(\hat{v}, u)}_{W(\hat{x}_i(u) - \hat{z}(u))} \tag{29}$$

The equality $\partial_v L_W(\hat{v}, u) = 0$ follows as $\hat{v}$ represents the minimizer of L_W over x_i and z by definition. Thus

$$\partial_{u_i} g(u^{[k-1]}) = W(x_i^{[k]} - z^{[k]}) \tag{30}$$

Setting $Q = W^{-1}$ gives the ascent direction (14). □

Proof of (15). With known multipliers $u_i = u_i^{[k-1]}$, the minimization of L_W over x_i and z is reduced to the following *regularized* least-squares problem

$$\min_{x_1, \ldots, x_n, z} \left\{ \sum_{i=1}^{n} ||y_i - A_i\, x_i||^2_{R_i^{-1}} + ||u_i^{[k-1]} - (z - x_i)||^2_W \right\} \tag{31}$$

Expressions (15) follow then as the corresponding least-squares solutions. Thus W can be interpreted as the weight matrix of the regularization term $||u_i^{[k-1]} - (z - x_i)||_W^2$ with the weighted constraint $u_i^{[k-1]} \approx z - x_i$. □

References

Baarda W (1973) S-transformations and Criterion Matrices. Tech. rep., Netherlands Geodetic Commission, Publ. on Geodesy, New Series, Vol. 5(1), Delft

Boomkamp H (2010) Global GPS reference frame solutions of unlimited size. Advances in Space Research 46(2):136–143

Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge university press

Boyd S, Parikh N, Chu E, Peleato B, Eckstein J, et al (2011) Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends® in Machine learning 3(1):1–122

Das S, Moura JM (2017) Consensus+Innovations Distributed Kalman Filter with Optimized Gains. IEEE Transactions on Signal Processing 65(2):467–481

Euler HJ, Goad CC (1991) On optimal filtering of GPS dual frequency observations without using orbit information. Bulletin Geodesique 65(2):130–143

Gabay D, Mercier B (1976) A dual algorithm for the solution of non linear variational problems via finite element approximation. Computers & Mathematics with Applications 2(1):17–40

Ghadimi E, Teixeira A, Shames I, Johansson M (2015) Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems. IEEE Transactions on Automatic Control 60(3):644–658

Glowinski R, Marroco A (1975) Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires. Revue française d'automatique, informatique, recherche opérationnelle Analyse numérique 9(R2):41–76

Hestenes MR (1969) Multiplier and gradient methods. Journal of optimization theory and applications 4(5):303–320

Hofmann-Wellenhof B, Lichtenegger H, Wasle E (2008) GNSS: Global Navigation Satellite Systems: GPS, Glonass, Galileo, and More. Springer, New York

Jadbabaie A, Lin J, Morse AS (2003) Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on automatic control 48(6):988–1001

Khodabandeh A, Teunissen PJG (2015) An analytical study of PPP-RTK corrections: precision, correlation and user-impact. J Geod 89(11):1109–1132

Khodabandeh A, Teunissen PJG, Zaminpardaz S (2018) Consensus-Based Distributed Filtering for GNSS. In: Kalman Filters-Theory for Advanced Applications, InTech, pp 273–304, DOI 10.5772/intechopen.71138

Kingston DB, Beard RW (2006) Discrete-time average-consensus under switching network topologies. In: American Control Conference, 2006, IEEE, pp 3551–3556

Li W, Nadarajah N, Teunissen PJ, Khodabandeh A, Chai Y (2017) Array-aided single-frequency state-space RTK with combined GPS, Galileo, IRNSS, and QZSS L5/E5a observations. Journal of Surveying Engineering 143(4):04017,006

Li X, Ge M, Dai X, Ren X, Fritsche M, Wickert J, Schuh H (2015) Accuracy and reliability of multi-GNSS real-time precise positioning: GPS, GLONASS, BeiDou, and Galileo. J Geod 89(6):607–635

Molzahn DK, Dörfler F, Sandberg H, Low SH, Chakrabarti S, Baldick R, Lavaei J (2017) A survey of distributed optimization and control algorithms for electric power systems. IEEE Transactions on Smart Grid 8(6):2941–2962

Moreau L (2005) Stability of multiagent systems with time-dependent communication links. IEEE Transactions on automatic control 50(2):169–182

Nadarajah N, Khodabandeh A, Wang K, Choudhury M, Teunissen PJG (2018) Multi-GNSS PPP-RTK: From Large- to Small-Scale Networks. Sensors 18(4):1078

Odolinski R, Teunissen PJG (2016) Single-frequency, dual-GNSS versus dual-frequency, single-GNSS: a low-cost and high-grade receivers GPS-BDS RTK analysis. J Geod 90(11):1255–1278, DOI 10.1007/s00190-016-0921-x

Olfati-Saber R, Murray RM (2004) Consensus problems in networks of agents with switching topology and time-delays. IEEE Transactions on automatic control 49(9):1520–1533

Powell MJ (1969) A method for nonlinear constraints in minimization problems. Optimization pp 283–298

Schaer S (1999) Mapping and predicting the Earth's ionosphere using the Global Positioning System. PhD thesis, University of Bern, Bern, Switzerland

Schizas ID, Ribeiro A, Giannakis GB (2008) Consensus in ad hoc WSNs with noisy links–Part I: Distributed estimation of deterministic signals. IEEE Transactions on Signal Processing 56(1):350–364

Teunissen PJG (1985) Generalized inverses, adjustment, the datum problem and S-transformations. In: Optimization and Design of Geodetic Networks, EW Grafarend and F Sanso (Eds), Springer

Teunissen PJG (1990) Nonlinear least squares. Manuscripta geodaetica 15:137–150

Teunissen PJG, Montenbruck O (eds) (2017) Springer Handbook of Global Navigation Satellite Systems. Springer

Teunissen PJG, Simons DG, Tiberius CCJM (2005) Probability and observation theory. Delft University, Faculty of Aerospace Engineering, Delft University of Technology, lecture notes AE2-E01

Wang Y, Hespanha J, Chakrabortty A (2018) Distributed estimation of power system oscillation modes under attacks on GPS clocks. IEEE Transactions on Instrumentation and Measurement 67(7):1626–1637

Zaminpardaz S, Teunissen PJ, Nadarajah N (2016) GLONASS CDMA L3 ambiguity resolution and positioning. GPS Solutions pp 1–15

Zhang B, Teunissen PJG (2015) Characterization of multi-GNSS between-receiver differential code biases using zero and short baselines. Science Bulletin 60(21):1840–1849