2017

A Temporal Algorithm for Satellite Subset Selection in Multi-Constellation GNSS

Peter F. Swaszek
University of Rhode Island, swaszek@uri.edu

Richard J. Hartnett

See next page for additional authors

Citation/Publisher Attribution
Swaszek, P. F., Harnett, R. J., Seals, K. C., & Swaszek, R. M. A. (2017). A Temporal Algorithm for Satellite Subset Selection in Multi-Constellation GNSS. Paper from *Precise Time and Time Interval Meeting*, Monetery, CA.
Available at: https://www.ion.org/ptti/abstracts.cfm?paperID=4629

Authors
Peter F. Swaszek, Richard J. Hartnett, Kelly C. Seals, and Rebecca M. A. Swaszek

A Temporal Algorithm for Satellite Subset Selection in Multi-Constellation GNSS

Peter F. Swaszek, *University of Rhode Island*
Richard J. Hartnett, *U.S. Coast Guard Academy*
Kelly C. Seals, *U.S. Coast Guard Academy*
Rebecca M. A. Swaszek, *Boston University*

BIOGRAPHIES

Peter F. Swaszek is a Professor of Electrical Engineering at the University of Rhode Island. His research interests are in statistical signal processing with a focus on digital communications and electronic navigation systems.

Richard J. Hartnett is a Professor of Electrical Engineering at the U.S. Coast Guard Academy, having retired from the USCG as a Captain in 2009. His research interests include efficient digital filtering methods, improved receiver signal processing techniques for electronic navigation systems, and autonomous vehicle design.

Kelly C. Seals is the Chair of the Electrical Engineering program at the U.S. Coast Guard Academy in New London, Connecticut. He is a Commander on active duty in the U.S. Coast Guard and received a PhD in Electrical and Computer Engineering from Worcester Polytechnic Institute.

Rebecca M. A. Swaszek is a PhD candidate in Systems Engineering at Boston University. Her research interests include the modeling and control of discrete event systems (e.g. bike share systems) and optimization theory.

ABSTRACT

GNSS receivers convert the measured pseudoranges from the visible GNSS satellites into an estimate of the position and clock offset of the receiver. For various reasons receivers might want to process only a subset of the visible satellites; it would be desired, of course, to use the best subset. In general, selecting the best subset is a combinatorics problem; selecting m objects from a choice of n allows for $\binom{n}{m}$ potential subsets. And since the typical performance criterion (e.g. Geometric Dilution of Precision) is nonlinear and non-separable in the satellites' locations in the sky, finding the best subset is a brute force procedure; hence, a number of authors have described sub-optimal algorithms for choosing satellites.

This paper revisits this problem, especially in the context of multiple GNSS constellations. The paper begins with a review of the existing subset selection algorithms. We note that all of these algorithms are what might be called "snapshot" in nature, selecting a subset for a single, fixed skyview of satellites. Through an example with the GPS constellation, we examine the time-sequential, or temporal, characteristics of the best subset selection noting:

- That the best subset at a particular point (snapshot) in time is also the best subset for a significant time interval around that point (typically measured in minutes).
- That the changes in the best subset over time are primarily, but not always, due to the loss or gain of a satellite crossing the horizon (or, more precisely, the receiver's mask angle).

Based upon these observations this paper develops several time-sequential, or temporal, algorithms that attempt to track the optimum subset of satellites over time at low computational cost. The accuracy and complexity of the algorithms are assessed with GPS constellation data. On a larger scale, these algorithms are then tested on combined GPS, GLONASS, and Galileo constellations with the resulting performance compared to optimal solutions found via exhaustive search.

INTRODUCTION

GNSS receivers convert the measured pseudoranges from the visible GNSS satellites into an estimate of the position and the clock offset of the receiver. The typical implementation of the solution algorithm is an iterative, linearized least squares method [1]. Assuming that pseudoranges from m satellites are measured, the "direction cosines" matrix is formed. Using an East, North, and Up coordinate frame this matrix is of the form

$$\mathbf{G} = \begin{bmatrix} e_1 & n_1 & u_1 & 1 \\ e_2 & n_2 & u_2 & 1 \\ \vdots & \vdots & & \\ e_m & n_m & u_m & 1 \end{bmatrix} \tag{1}$$

in which (e_k, n_k, u_k) is the unit vector pointing toward the k^{th} satellite from the assumed receiver position. This matrix is then used to form the pseudoinverse to solve the set of overdetermined, linearized pseudorange equations. Since the pseudoranges themselves are noisy, the resulting solution is also noisy. The accuracy of the resulting solution can be described statistically by its error covariance matrix, which is proportional to the inverse of $\mathbf{G}^T\mathbf{G}$. Rather than considering all of the individual elements of this covariance matrix, it is common to reduce it to a scalar parameter. The most used reduction is the GDOP (Geometric Dilution of Precision), the square root of the trace of this matrix

$$\text{GDOP} = \sqrt{\text{trace}\left\{(\mathbf{G}^T\mathbf{G})^{-1}\right\}}$$

equivalently, proportional to the square root of the sum of the variances of the four estimates (three of position and one of time). For multiple constellations the definition of $\mathbf{G}$ can be extended by appending additional columns to account for different clock biases. Specifically, for L unsynchronized constellations, $\mathbf{G}$ is of the form

$$\mathbf{G} = \begin{bmatrix} e_{1,1} & n_{1,1} & u_{1,1} & 1 & 0 & 0 & \dots & 0 \\ & \vdots & & \vdots & & & \vdots & \\ e_{1,m_1} & n_{1,m_1} & u_{1,m_1} & 1 & 0 & 0 & \dots & 0 \\ e_{2,1} & n_{2,1} & u_{2,1} & 0 & 1 & 0 & \dots & 0 \\ e_{2,2} & n_{2,2} & u_{2,2} & 0 & 1 & 0 & \dots & 0 \\ & \vdots & & \vdots & & & \vdots & \\ e_{L,m_L} & n_{L,m_L} & u_{L,m_L} & 0 & 0 & 0 & \dots & 1 \end{bmatrix}$$

(the first of the two subscripts is the constellation number, 1 to L; the second is the satellite number within the k^{th} constellation, 1 to m_k) and the GDOP is still as written above, but now includes the variances of $L + 3$ variables, three for the receiver's position and those of the L clock biases.

In some instances a GNSS receiver cannot process all of the visible satellites. For example, the issue might be:

- That the receiver physically cannot track all of the potential signals – this might be a hardware limitation (due to a fixed number of channels) or the desire to minimize power usage [2, 3].
- That the receiver is using corrections from some augmentation system and that the bandwidth of the correction channel is insufficient to provide information for all of the visible satellites (see, for example, [4]).

In such a case the question arises: "If only m of n ($m < n$) visible satellites can be processed, which m should they be?" Consider the question for the GDOP criterion. Since the GDOP is nonlinear and non-separable in the satellites locations in the sky, finding the best subset is a combinatorics problem; selecting m objects from a choice of n allows for $\binom{n}{m}$ possible subsets. If the receiver limits its attention to the 15 or so visible GPS satellites then a brute force comparison of all subsets to find the optimal subset is possible (for $n = 15$, $\binom{15}{m} < 6,500$ for all $3 < m < 15$, well within modern computational capability). The advent of other GNSS constellations exacerbates this problem. For example, desiring to process 16 of 35 visible satellites (such as frequently occurs with GPS, GLONASS, and Galileo) brute force comparison is no longer viable (e.g. with $n = 35$, $\binom{35}{16} > 4$ billion, impossible for real time usage).

This question of selecting a subset of the possible satellites is not new in the navigation literature; multiple authors have described sub-optimal algorithms for choosing the satellite subset. (Note, however, that this question is still timely; all three of the papers referenced above, [2–4], are from 2016.) Most of these sub-optimum methods are

greedy in nature (described below) and have been developed under the assumption of a single constellation. Further all of these algorithms are what might be called "snapshot" in nature, selecting a subset for a single, fixed skyview of satellites.

This paper revisits this problem, especially in the context of multiple GNSS constellations. The next section briefly reviews the existing subset selection algorithms. This review is followed by a motivational example with the GPS constellation, examining the time-sequential, or temporal, characteristics of the best subset. Based upon these observations, this paper develops several time-sequential, or temporal, algorithms that attempt to track the optimum subset of satellites over time at low computational cost. The accuracy and complexity of the algorithms are assessed with GPS constellation data. Finally, these algorithms are tested on combined GPS, GLONASS, and Galileo constellations with the resulting performance compared to several optimal solutions found via full search (brute force) computation.

PREVIOUS SUBSET SELECTION ALGORITHMS

Multiple authors have presented sub-optimum satellite selection procedures; a number of these employ alternative performance measures beyond GDOP. These include volume of the polytope formed by the satellites [5, 6] and cosines of the angles between pairs of satellites [7–9], as well as combinations of vertical and horizontal protection limits [3].

The sub-optimum algorithms tend to be greedy algorithms, making optimum decisions on a sequence of smaller problems. For example, to generate a subset of size m the authors of [10] suggest starting with the full set of n satellites and iterating the following steps:

1. Assuming that the current subset consists of k satellites, compute k GDOPs, one for each proper subset of $k-1$ satellites.
2. Of these k values identify and remove that satellite which results in the smallest increase in GDOP; the result is a subset of size $k-1$.
3. Repeat steps 1 and 2 until $k = m$.

This algorithm is greedy in that it makes an optimum choice at each step in the iteration although the final result might not be the global optimum. Specifically, a poor (but still locally optimum) choice at one step might lead to a globally sub-optimum solution for future iterations. It is noted in [10] that the loss to the global optimum for small constellations appears to be small; however, there is no guarantee that this is true for larger numbers of satellites. In a similar way it is possible to grow the subset greedily from the best set of 4 [11].

Another sub-optimum algorithm suggests starting with a subset of size m (and one could discuss how to select this initial subset!) and then iterate in a greedy fashion – growing the subset to $m+1$ satellites by adding the most helpful (with respect to GDOP) of the unused satellites and then shrinking back down to m by removing the least helpful one, denoted a "revolving door" method [12] – until a equilibrium is reached.

Simulated annealing has also been proposed as a technique to implement a GDOP-based satellite selection algorithm [13].

Finally, an algorithm could try to mimic the minimum GDOP constellation from the known results on lower bounds to GDOP. Specifically, in [14] it is shown that for m satellites the minimum GDOP is attained by a constellation consisting of approximately 30% of the satellites at zenith and the remaining 70% at the horizon. An algorithm could, then, choose high elevation satellites to match the number desired to be at zenith and then attempt to place the remaining satellites near the horizon following "balance" [15, 16].

A MOTIVATING EXAMPLE

To motivate the time sequential, or temporal, algorithms developed in this paper, consider the case of selecting a subset of satellites from the GPS constellation. The data for the experiment consists of azimuth and elevation angles of the GPS satellites visible at our location in New England for a 24 hour period; the data was decimated to one minute snapshots (a total of 1440 snapshots). Figure 1 shows the number of satellites above the horizon over the course of the day, ranging from 9 to 14.

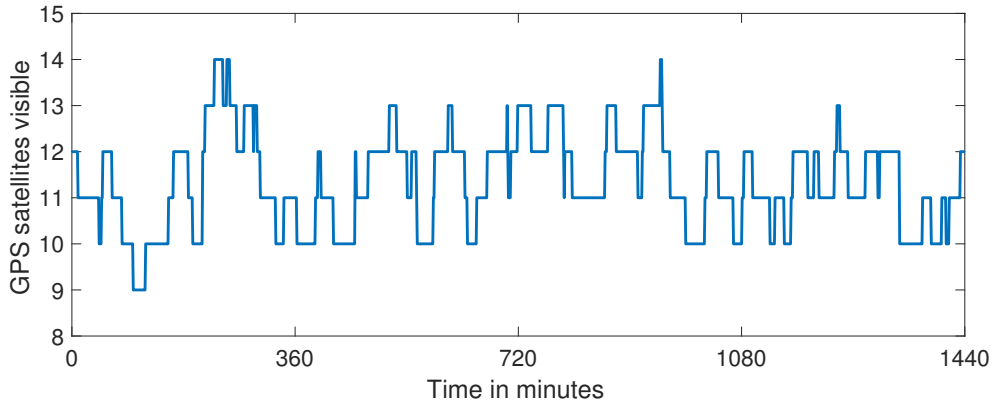

Figure 1: Number of GPS satellites visible versus time.

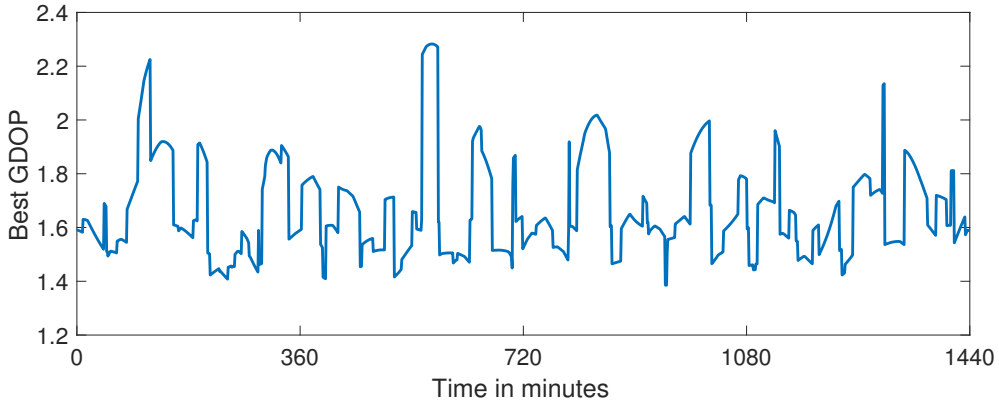

Figure 2: GDOP for the best 7 satellites versus time.

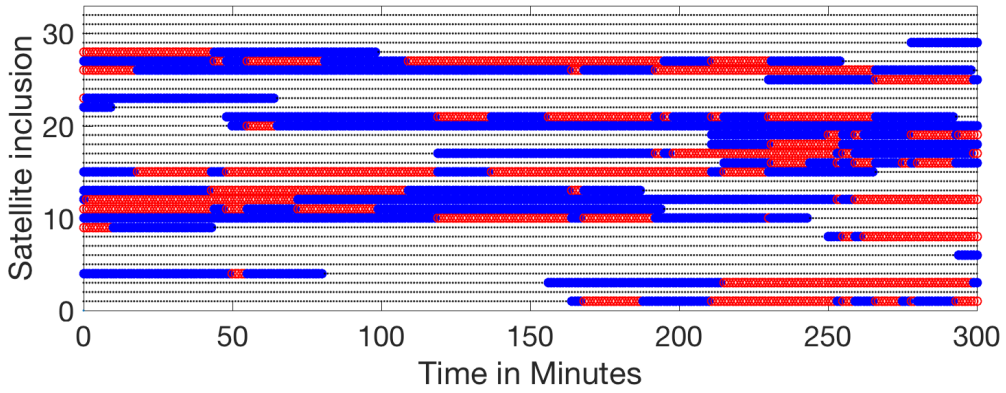

Figure 3: Satellite inclusion versus time.

Next, the best 7 satellites for the GDOP criterion were found for each minute's dataset by a brute force computation; the resulting GDOP performance is shown in Figure 2.

To analyze these results Figure 3 shows, for each satellite, its inclusion in the best subset as a function of time. Specifically, for each satellite (vertical value equal to the satellite number, 1 to 32) the satellites inclusion, exclusion, or non-visibility is shown by a blue circle, red circle, or black dot, respectively. (For ease of interpretation this plot shows only the first 300 minutes of the data; the rest is similar.) We notice that a satellite appears to be in the

best subset when rising or setting at the horizon (the colored segments appear to have blue at both ends). Further, inclusion of a satellite in the best subset is clearly correlated in time (the blue symbols are clumped together); once in the best subset, a satellite stays in the subset for some significant period of time.

A closer examination of the change in the best subsets from one minute to the next is also possible:

- Figure 4 shows what happens fully 95% of the time in this example: the next best subset either matches the current best subset exactly (left subfigure) or is different by only one satellite (right subfigure, the swapping pair is encircled in black). In these graphics the central portion is a close up of that portion of Figure 3 to show the satellite numbers in the previous and next subsets; the left and right hand portions of each subfigure show the corresponding before and after skyviews with filled red circles for satellites used in the solution and blue open circles for unused satellites. In the test data set the full match of previous and next optimum subsets occurred 88% of the time; the mismatch by one satellite was the additional 7%. This single mismatch example shows a satellite rising from the horizon and displacing one in the previous subset. Other examples with a single mismatch could show one of the satellites falling below the horizon (hence, needing to be replaced) or a simple swap of two existing satellites.
- Figure 5 shows examples of what happens the remaining 5% of the time in the experiment; 5 matches across the time interval (left subfigure, two swaps again circled in black) and only 3 matches (right subfigure). The types of changes observed include simple swaps between satellites visible at both times – due to a change in their precise spatial relationships – and gains/losses of satellites to the horizon.

These examples suggest that an algorithm that evolves the best subset over time might be simple in form; that examining those subsets that are simple modifications of the previous subset will often include the new optimum subset. We develop several such algorithms below.

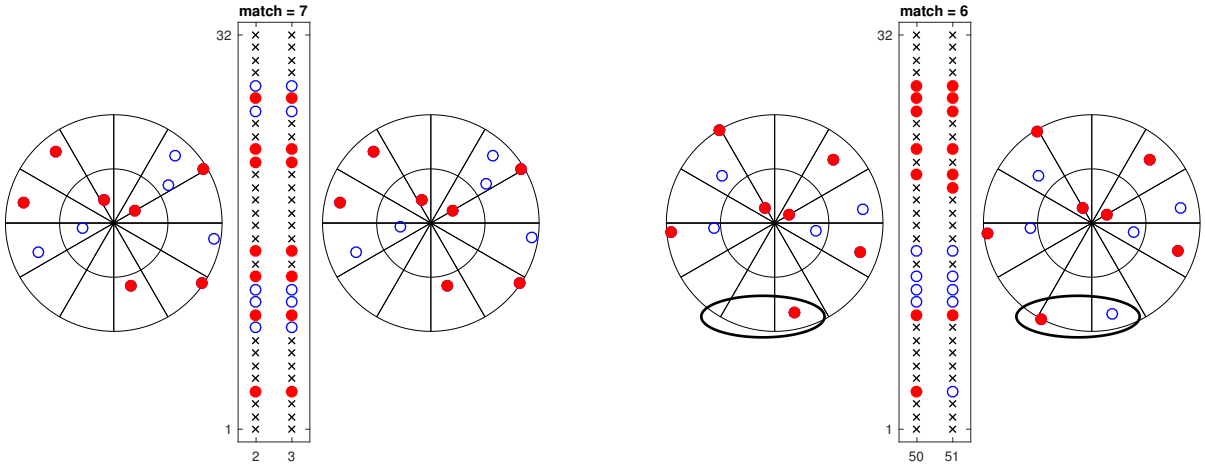

Figure 4: A match of all 7 or 6 of the 7 satellites between the previous and next optimum subsets.

PROPOSED ALGORITHMS

The example above suggests that a time sequential algorithm that makes small modifications to the previous subset selection could be an efficient way to generate the next optimum subset of satellites. In this section we describe the implementation, discuss the complexity, and examine the performance of several such approaches; all of them evolve the subset over time in a greedy fashion. In each case we assume that an initial subset of size m satellites (the so called *previous* subset) is available and that we have a set of n ($> m$) satellites to choose from for the next subset.

The basic concept for each algorithm is to select the next subset to be the same or nearly the same as the previous subset, the change in satellite selection being based on comparing the GDOP that results from simple possible changes to the subset. For example, the simplest version might compare no changes to the satellite list with those

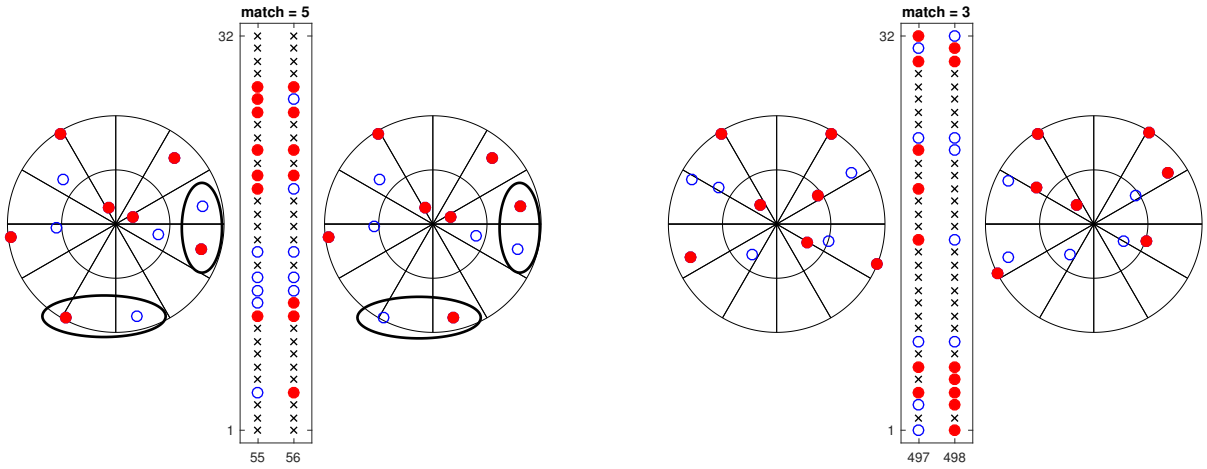

Figure 5: Fewer matches across the time step: (left) 5 of the 7 and (right) 3 of the 7.

subsets formed by swapping *one* satellite in the previous subset with one of the unused satellites. From a complexity of implementation perspective this would require computing the GDOP for $1 + m(n - m)$ subsets, one for no change in the subset (but requiring recomputation since the m satellites have moved a small amount over the time step) and $m(n - m)$ additional GDOP computations since there are m choices for the satellite being replaced and $n - m$ potential replacements.

Unfortunately, as written this algorithm could fault if one or more of the current satellites falls below the horizon; hence, we modify it as follows:

Algorithm 1:

1. If all m of the satellites in the previous subset are still available in the set of n currently visible satellites:
   (a) Recompute the GDOP for the previous subset using the satellites' new sky positions.
   (b) For each of the m satellites in the previous subset compute the GDOPs that would result if that satellite were dropped and one of the $n - m$ unused satellites replaced it.
   (c) Of these $1 + m(n - m)$ possibilities pick the subset with the smallest resulting GDOP.
2. If p $(0 < p \leq m)$ of the satellites in the previous subset are lost to the horizon:
   (a) Compute the $\binom{n-m}{p}$ GDOPs in which the p lost satellites are replaced by unused satellites.
   (b) Pick the best of these subsets with replacement, using its GDOP as the benchmark for further swaps.
   (c) For each of the m satellites in this replenished subset compute the GDOPs that would result if that satellite were dropped and one of the $n - m - p$ unused satellites replaced it.
   (d) Pick the subset with the smallest resulting GDOP.

How well does this approach work? Figure 6 replots the best GDOP from Figure 2, overlaying the performance of the subsets chosen by Algorithm 1 (the algorithm was started with the best subset found by the full search at time zero). At a cursory level the performance is indistinguishable; perhaps just a few corners of blue are visible. Figure 7 shows the percentage increase in GDOP of the subsets found by Algorithm 1 as compared to the best GDOP; we observe that the loss is often zero and at worst 6%! Figure 8 shows the number of GDOP computations per minute for Algorithm 1 (this fluctuates with the number of visible satellites, n). Our observation is that this first algorithm provides *very good* GDOP performance for *very modest* computational effort.

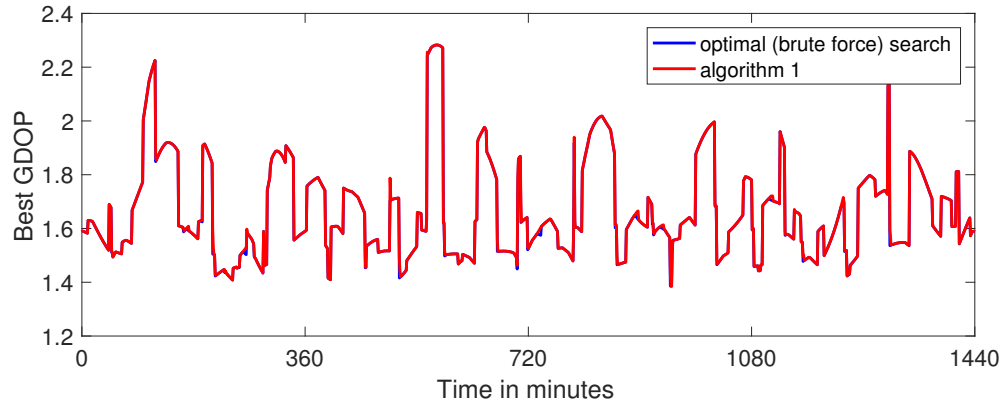

Figure 6: GDOP for the best 7 satellites vs time: brute force and the Algorithm 1.

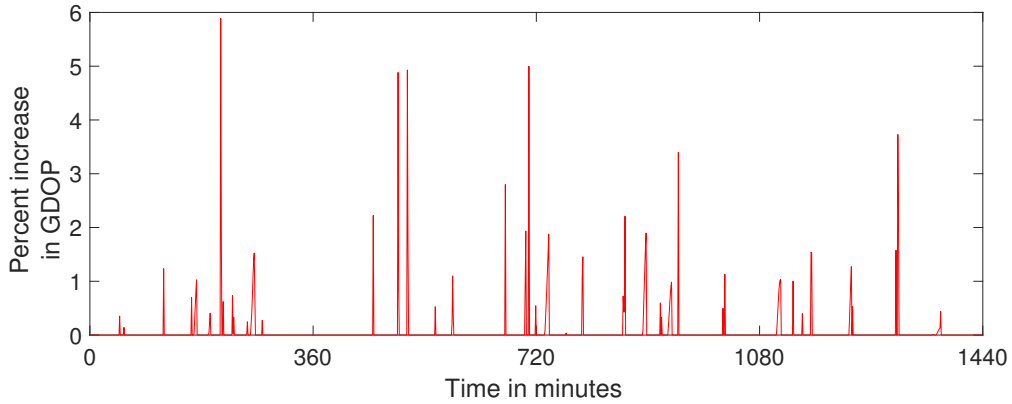

Figure 7: The percentage increase in GDOP for Algorithm 1.

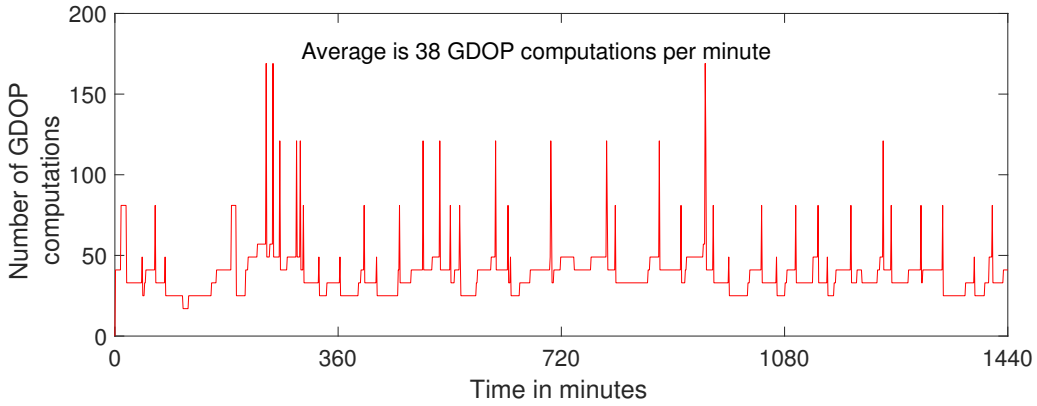

Figure 8: The number of GDOP computations per time interval for Algorithm 1.

From the examples in the section above we expect that this algorithm should be able to track the best subset 95% of the time for our data set (specifically, 88% of the time the previous and next subsets were seen to be identical and in an additional 7% the subsets only differed by one satellite); however, the fraction of time that the GDOP difference is non-zero is greater than the remaining 5% of the time. The difference is that when the next best subset is significantly different than the previous one (as in Figure 5), Algorithm 1 takes several time intervals to get back on track. Figures 9 attempts to document one such instance:

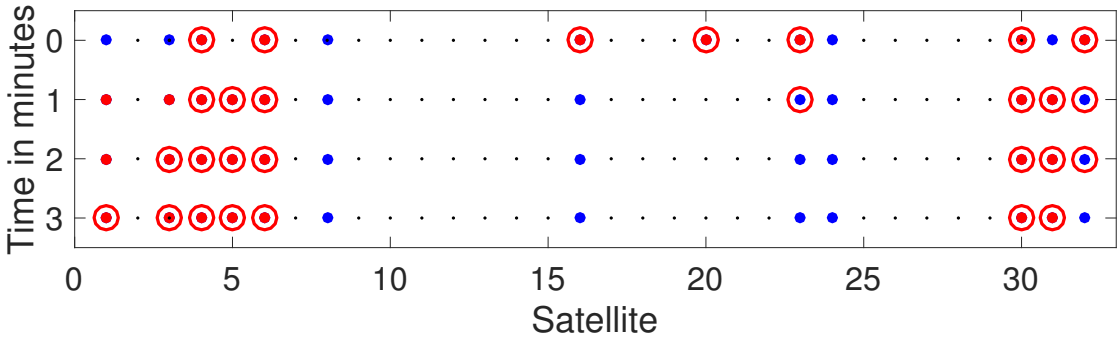

Figure 9: Algorithm 1's response to a significant subset change.

- The vertical scale is time in minutes, progressing downward; the horizontal scale is satellite number.
- Black dots and filled blue circles represent invisible and visible satellites, respectively.
- Filled red circles show the best subset of size 7 as computed via brute force (these overlay the blue circles). Note that the best subset jumps from (4,6,16,20,23,30,32) at time 0 to (1,3,4,5,6,30,31) at time 1, a loss of one to the horizon (SV 20), a gain of one from the horizon (SV 5), and overall a change 4 satellites.
- Algorithm 1's subset is shown by open red circles; it has the best configuration at time 0. Algorithm 1 adapts to this hiccup at time 1 over multiple time steps:
  1. For time 1 it replaces SV 20 with SV 5 and then swaps SV 16 for SV 31.
  2. For time 2 it swaps SV 23 for SV 3.
  3. For time 3 it swaps SV 32 for SV 1, and is again tracking the optimum subset.

This delay in catching up to the optimum subset suggests two modifications of Algorithm 1:

Algorithm 2 – extending Algorithm 1 to pairs of satellite swaps (modify steps 1(b) and 2(c)):

1. If all m of the satellites in the previous subset are still available:
   (a) Recompute the GDOP for the previous subset using the satellites' new sky positions.
   (b) For each of the $\binom{m}{2}$ pairs of satellites in the previous subset compute the GDOPs that would result if that satellite were dropped and any of the $\binom{n-m}{2}$ pairs of the $n-m$ unused satellites replaced them.
   (c) Pick the subset with the smallest resulting GDOP.
2. If p $(0 < p \leq m)$ of the satellites in the previous subset are lost to the horizon:
   (a) Compute the $\binom{n-m}{p}$ GDOPs in which the p lost satellites are replaced by unused satellites.
   (b) Pick the best of these subsets with replacement, using its GDOP as the benchmark for further swaps.
   (c) For each of the $\binom{m}{2}$ pairs of satellites in the subset with replacements compute the GDOPs that would result if those two satellites were dropped and any of the $\binom{n-m-p}{2}$ pairs of the $n-m$ unused satellites replaced them.
   (d) Pick the subset with the smallest resulting GDOP.

and

Algorithm 3 – an iterative version of Algorithm 1 (add step 3):

1. If all m of the satellites in the previous subset are still available:
   (a) Recompute the GDOP for the previous subset using the satellites' new sky positions.

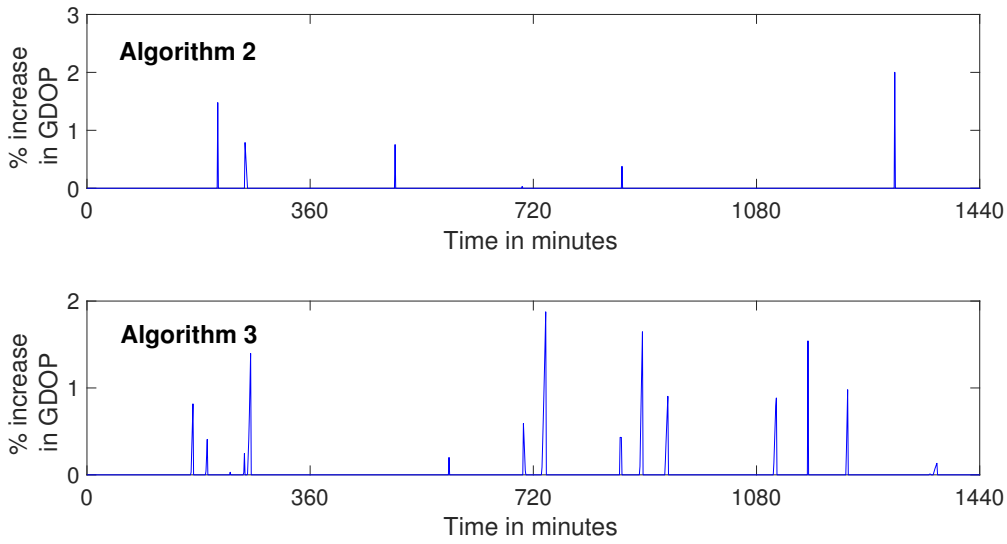

Figure 10: The percentage increase in GDOP for Algorithms 2 and 3.

(b) For each of the m satellites in the previous subset compute the GDOPs that would result if that satellite were dropped and one of the $n-m$ unused satellites replaced it.

(c) Pick the subset with the smallest resulting GDOP.

2. If p $(0 < p \leq m)$ of the satellites in the previous subset are lost to the horizon:

   (a) Compute the $\binom{n-m}{p}$ GDOPs in which the p lost satellites are replaced by unused satellites.

   (b) Pick the best of these subsets with replacement, using its GDOP as the benchmark for further swaps.

   (c) For each of the m satellites in this replenished subset compute the GDOPs that would result if that satellite were dropped and one of the $n-m-p$ unused satellites replaced it.

   (d) Pick the subset with the smallest resulting GDOP.

3. Iterate the following steps until an equilibrium is reached:

   (a) For each of the m satellites in the subset compute the GDOPs that would result if that satellite were dropped and one of the $n-m$ unused satellites replaced it.

   (b) Pick the subset with the smallest resulting GDOP.

Clearly we could also extend Algorithm 2 to larger swap sets (although at a cost of a significant increase in complexity) and could iterate Algorithm 2.

Both Algorithms 2 and 3 are expected to have higher performance, but also higher complexity. With respect to complexity, recall that Algorithm 1 required $1 + m(n-m)$ computations of GDOP per time step. For the two modifications:

- Algorithm 2 requires $1 + \binom{m}{2}\binom{n-m}{2}$ GDOP calculations per time step.
- The complexity of Algorithm 3 depends upon how many cycles of iteration are required until two swapping converges; we can think of this as $s(1 + m(n-m))$ in which s is that number of repetitions.

Figure 10 shows the resulting improvement in GDOP for these two extensions; both reduced the maximum percentage increase in GDOP to about 2%. Most notable when comparing these results back to Figure 7 is now much sparser the increases are; more of the points match exactly. Figure 11 shows the required computation in multiple of GDOPs; not surprisingly, both Algorithms 2 and 3 require more work than the simplicity of Algorithm 1. Algorithm 2 is worse since it requires additional computation at *every* time step; Algorithm 3 is more parsimonious at spending extra

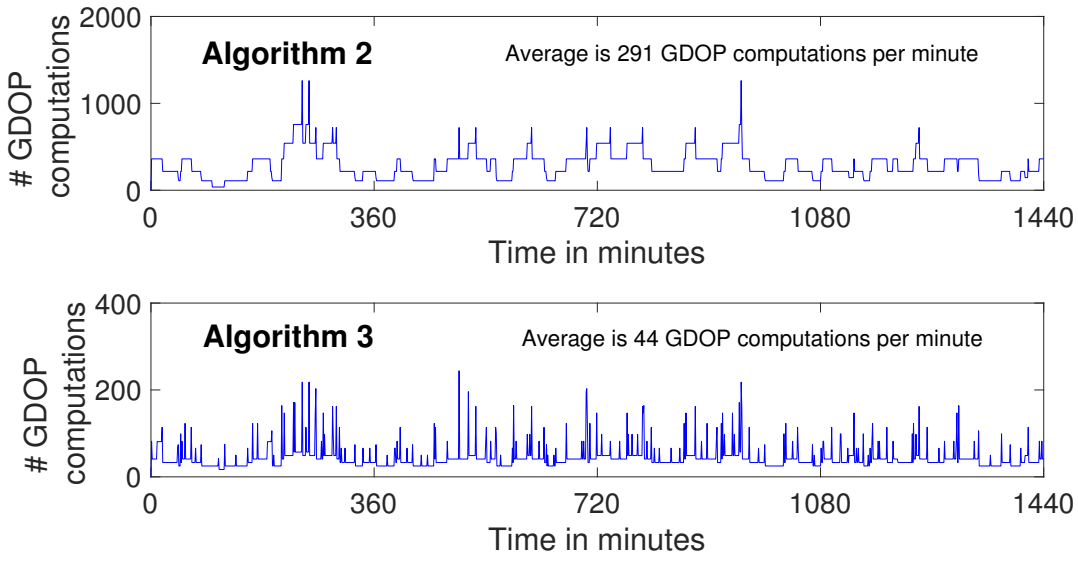

Figure 11: The number of GDOP computations per time step for Algorithms 2 and 3.

computation effort only when needed (the average number of iterations per time step is only 1.15 for this data set). One can, of course, debate the value of the increase in performance versus the penalty in computational load.

As a final comment, we note that the above example started Algorithm 1 (and actually the others) with the best subset at time 0. To investigate the convergence of this algorithm to the best subset we reran it starting in all possible subsets (since at time 0 there were 12 satellites visible, there are $\binom{12}{7} = 792$ initial subsets). Figure 12 shows the GDOP for all of these trials (in red) versus the number of time steps; the best performance is shown in blue. We note that Algorithm 1 converges quite quickly to the best subset.

EXAMPLES WITH MULTI-CONSTELLATIONS

To test the temporal algorithms developed in this paper, consider the case of selecting a subset of satellites from the combination of the GPS, GLONASS, and Galileo constellations. As above, the data consists of azimuth and elevation angles every minute at our location in New England for a 24 hour period. Figure 13 shows the number of satellites above the horizon over the course of the day, ranging from 23 to 35.

Referring to the Introduction, the inclusion of the variances of the L clock biases makes GDOP a less than ideal

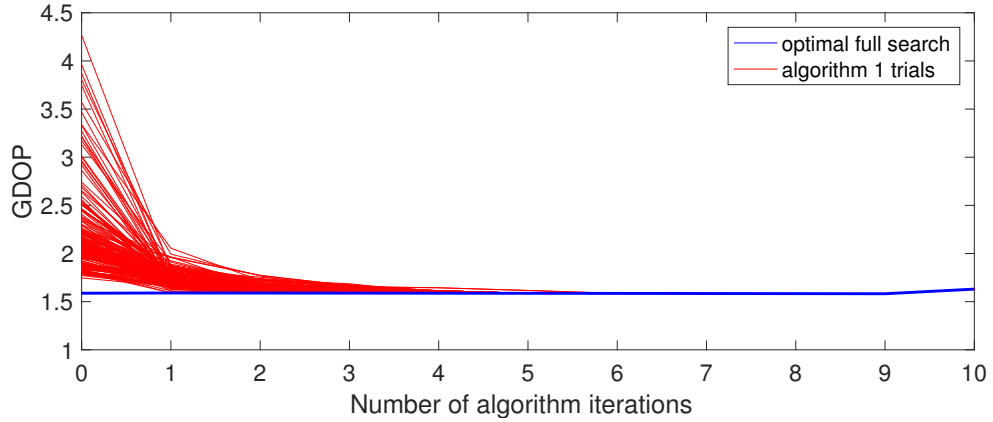

Figure 12: Convergence of Algorithm 1 from a random starting subset.

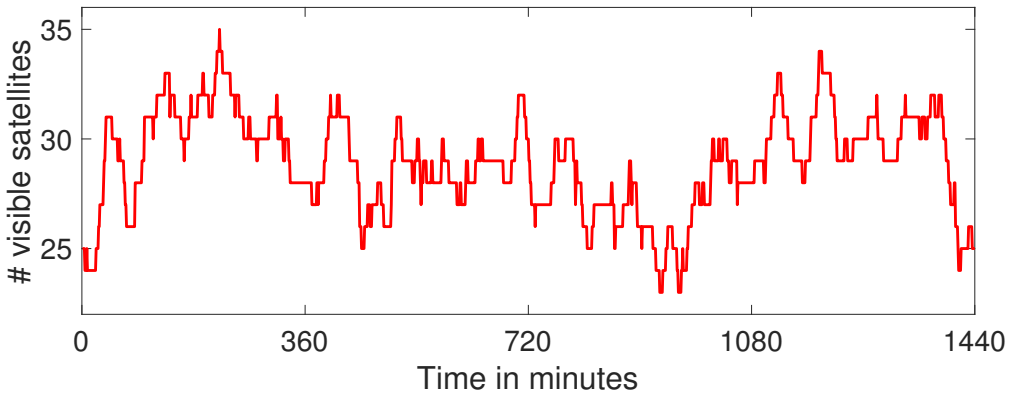

Figure 13: Visible satellites for 3 constellations versus time of day.

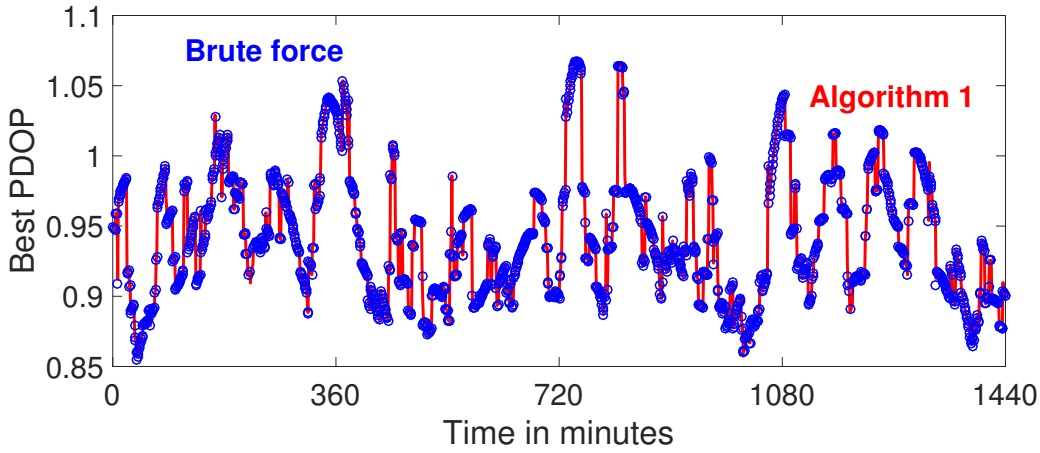

Figure 14: PDOP for 3 constellations versus time of day.

choice when comparing subsets of satellites from multiple constellations; for example, the best subset for GDOP might limit itself to a poorer geometric choice of satellite locations just to eliminate the estimation of an extra clock bias. A similar performance metric that resolves this problem is Position DOP (PDOP), dependent only on the position variances. Defining

$$\mathbf{H} = \left(\mathbf{G}^T\mathbf{G}\right)^{-1}$$

then PDOP is a function of the first three diagonal terms

$$\text{PDOP} \equiv \sqrt{\text{trace}\left\{\mathbf{H}_{[1,1]} + \mathbf{H}_{[2,2]} + \mathbf{H}_{[3,3]}\right\}}$$

a combination of the variances of the three position variables of the estimate.

Algorithm 1 was updated to optimize the PDOP for multiple constellations and run on this data set; the resulting performance is shown in Figure 14 as the red line. For comparison, brute force computation was implemented for this data set as well; these minimum PDOP values are shown as blue dots. As in the GPS example above, Algorithm 1 seems to produce excellent results. Figure 15 examines the percentage increase in PDOP; at worst 2% across this 24-hour data set.

The computational load for Algorithm 1 is shown in Figure 16. There is considerable computational load difference between Algorithm 1 and brute force computation. Specifically, on a current day laptop running the computation in MatLab, the CPU time required to generate all 1440 subsets by Algorithm 1 was less than 10 seconds; the brute force computation required approximately 1.5 CPU years (!!!).

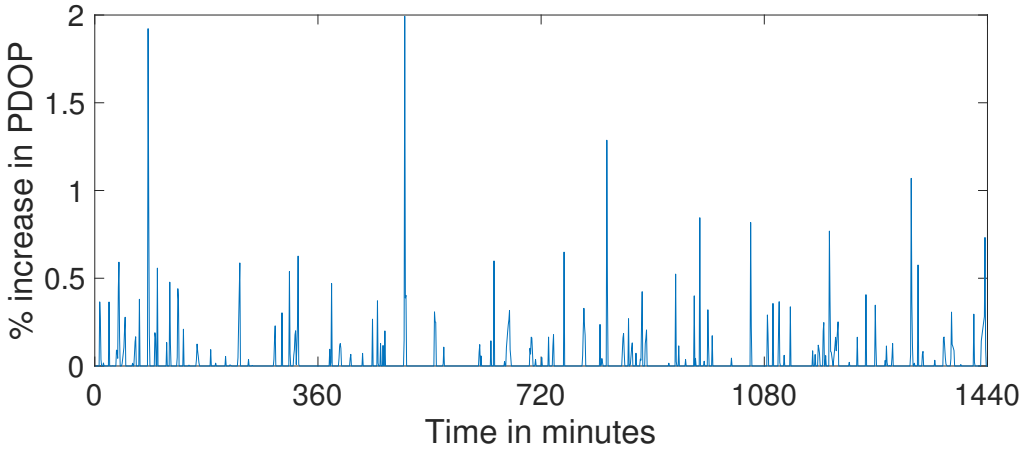

Figure 15: Close ups of the PDOP comparison.

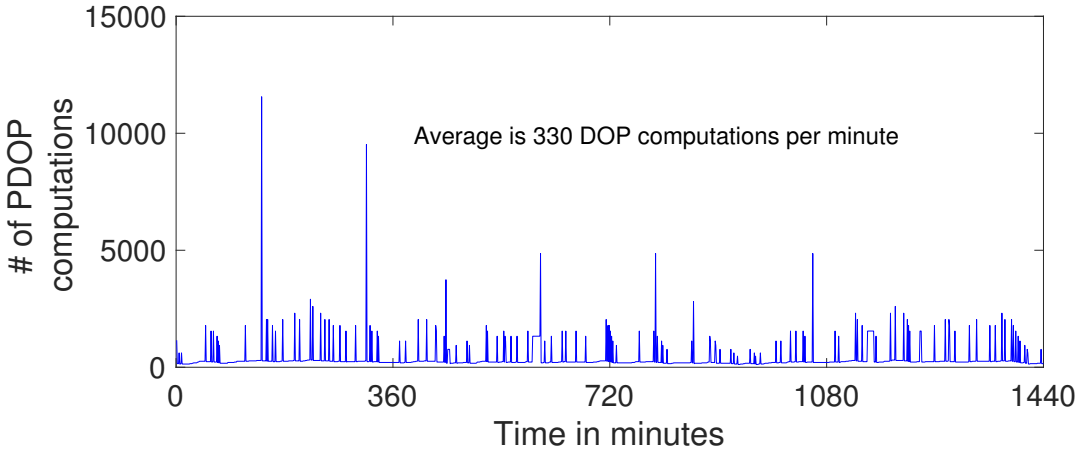

Figure 16: The number of PDOP computations per time step for Algorithm 1 with 3 constellations.

CONCLUSIONS/FUTURE WORK

The sections above make several points:

- That the problem of selecting a subset of the visible GNSS satellites is still an important problem.
- That with multiple constellations and the GDOP/PDOP performance criterion, a brute force approach to selecting the best subset is infeasible from a complexity perspective.
- The proposed time sequential, or temporal, algorithms shows great promise; contrary to other greedy algorithms, they appear to be able to pull out of local minima. Our conjecture is that the changing locations of the satellites act like methods in simulated annealing.

Our future work is to combine the understanding of optimum constellations to improve suboptimum subset selection algorithms for multiple constellations. Specifically, the algorithms herein searched overall all available satellites for the swaps. Clearly we could improve on this by identifying selected candidate satellites.

For example Figure 17 shows the elevation angles of the visible GPS satellites versus time (limited to the first 300 minutes to allow for interpretation; the rest of the data is similar) for the motivational example above; the blue and red colors identify those satellites in the best 7 and not, respectively. Several observations are apparent:

- The highest and lowest elevation satellites appear to be always within the best 7 (at least to the resolution of this graphic); mid-elevation satellites are usually *not* chosen, although that varies with the total set of visible

satellites.

- A satellite near the horizon appears to always be in the subset.

One goal would be to exploit these observations to further reduce the computational load of the subset selection algorithms.

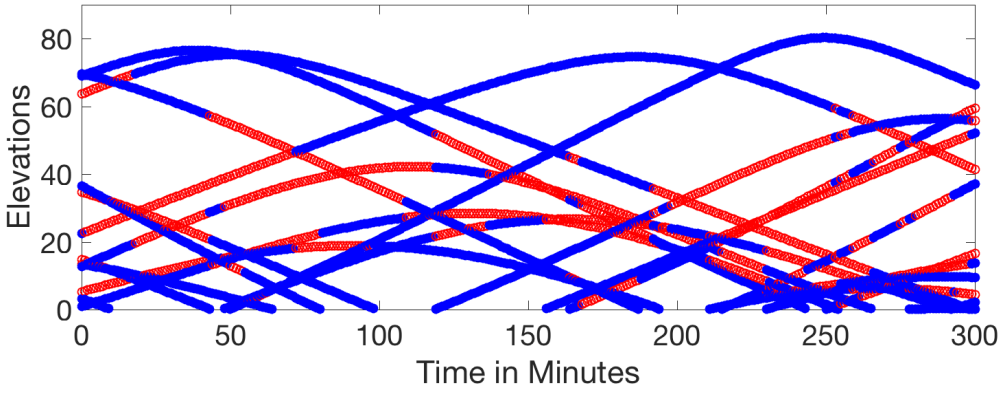

Figure 17: Elevation angles versus time of the GPS motivating example.

REFERENCES

[1] P. Misra and P. Enge, **Global Positioning System: Signals, Measurement, and Performance**, Ganga-Jamuna Press, 2001.

[2] D. A. Divis, "Handling an embarrassment of riches – too many satellites," *Inside GNSS*, Oct. 31, 2016.

[3] T. Walter, J. Blanch, and V. Kropp, "Satellite selection for multi-constellation SBAS," *Proc. ION GNSS+ 2016*, Portland OR, Sept. 2016.

[4] D. Gerbeth, M. Felux, M-S. Circiu, and M. Caamano, "Optimized selection of satellites subsets for a multi-constellation GBAS," *Proc. ION Int'l. Tech. Mtg.*, Monterey CA, Jan. 2016.

[5] M. Kihara and T. Okada, "A satellite selection method and accuracy for the Global Positioning System," *Jour. Navigation*, vol. 31, no. 1, Spring 1984, pp. 8-20.

[6] N. Blanco-Delgado and F. D. Nunes, "Satellite selection method for multi-constellation GNSS using convex geometry," *IEEE Trans. Veh. Tech.*, vol. 59. no. 9, Nov. 2010, pp. 4289-4297.

[7] J. Li, A. Ndili, L. Ward, and S. Buchman, "GPS receiver satellite/antenna selection algorithm for the Stanford gravity probe B relativity mission," *Proc. ION NTM*, San Diego CA, Jan. 1999.

[8] C-W. Park and J. P. How, "Quasi-optimal satellite selection algorithm for real-time applications," *Proc. ION GPS 2001*, Salt Lake City UT, Sept. 2001.

[9] M. Wei, J. Wang, and J. Li, "A new satellite selection algorithm for real-time application," *Proc. IEEE Intl. Cong. Sys. & Informatics*, 2012, pp. 2567-2570.

[10] M. Liu, M-A. Fortin, and R. Landry, "A recursive quasi-optimal fast satellite selection method for GNSS receivers," *Proc. ION GNSS 2009*, Savannah GA, Sept. 2009.

[11] A. Peng, G. Ou, and G. Li, "Fast satellite selection method for multi-constellation Global Navigation Satellite System under obstacle environments," *IET Radar Sonar Navig.*, vol. 8, no. 9, 2014, pp. 10511058.

[12] M. S. Phatak, "Recursive method for optimum GPS satellite selection," *IEEE Trans. Aero. Elect. Sys.*, vol. 37, no. 2, Apr. 2001, pp. 751-754.

[13] M. Ranjbar and M. R. Mosavi, "Simulated annealing clustering for optimum GPS satellite selection," *Int'l. Jour. Comp. Sci.*, vol. 9, no. 3, May 2012.

[14] P. F. Swaszek, R. J. Hartnett, and K. C. Seals, "Lower bounds to DOP," *Jour. Navigation*, in review.

[15] M. Zhang and J. Zhang, "A fast satellite selection algorithm: beyond four satellites," *IEEE Jour. Selected Topics in Signal Processing*, vol. 3, no. 5, Oct. 2009, pp. 740-747.

[16] P. F. Swaszek, R. J. Hartnett, and K. C. Seals, "Multi-constellation GNSS: new bounds on DOP and a related satellite selection process," *Proc. ION GNSS+ 2016*, Portland OR, Sept. 2016.