MAST30013 - Techniques in Operations Research

Power Optimisation in Wireless Sensor Networks

Authors:
Dario Latina
Michael Le
Rushaa Atawoo
Vaia Papadopoulos

May, 2021

Contents

List of Figures

List of Tables

1 Introduction

Wireless sensor networks (WSNs) are systems that are independent, self-sufficient and autonomous, used to measure aspects of an environment. Each network consists of sensors, with the number of sensors dependent on the use and scale of the network. These sensors operate by transmitting information to a central relay which aggregates the local sensed data so that it can transmit the information to a base station to be processed. Wireless sensor networks are able to monitor physical and environmental conditions to cooperatively transfer their data along the network system [10]. Examples of physical and environmental conditions include pressure, sound, pollutants, temperature, motion or vibration. In most cases a wireless sensor network consists of hundreds of thousands of sensor nodes which communicate with one another through radio signals. They are widely known as common resource constrained problems within the mathematical and engineering sciences departments, due to the limited processing speed, storage capacity and communication bandwidth that the relays and sensors consist of.

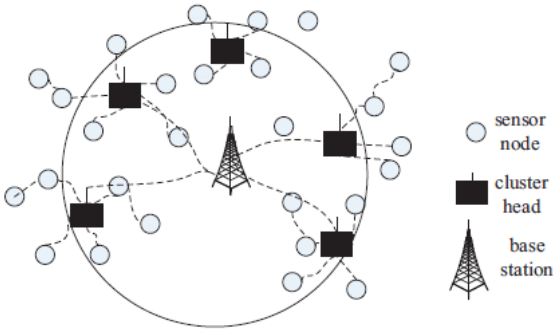


Figure 1: Example of a wireless sensor network setup

Source: https://www.researchgate.net/profile/Ashish-Patel-29/publication/321161209/figure/fig1/AS:562654695653376@1511158741435/A-hierarchical-wireless-sensor-network-architecture11.png

There are many difficulties that arise in the designing and construction process of wireless sensor networks. So that a robust and effective wireless sensor network can be created, factors such as transmission media, power consumption, fault tolerance, hardware limitations, production costs and scalability need to be thoroughly examined and considered during the design process. Having limitations in power resources is common, since for example most sensors and relays are battery operated. This is one of many reasons that the optimisation of power consumption in wireless networks is heavily researched [8]. Another reason for the emphasis on energy optimisation is so that the life of the network can be prolonged which is impacted by reduction costs in energy requirements. With efficient systems in place, costs and other challenging issues that may arise can effectively be minimised.

Wireless sensor networks can be applied within a variety of fields such as the military, transportation, healthcare, industrial monitoring, remote area monitoring and environment monitoring [1]. When accessible information is delivered to the right person at the right time, problems can be prevented. Within these various fields wireless sensor networks help deliver messages in quick and efficient ways making them effective and essential systems which can help prevent problems such as casualties from occurring.

Wireless sensor networks have been widely used within the military for some time. They have emerged as excellent tools for defence and protection mechanisms involving information gathering, various parameter monitoring, smart logistics support within unknown areas and importantly intrusion detection [3]. Within a battlefield the firing of guns and other weaponry creates sound, heat and vibrations which can be detected by the nodes of the wireless sensor network. The information processed by these signals provide an expectation of the physical location of the enemy

which can be especially useful in environments with low visibility [9].

The healthcare industry also incorporates the use of wireless sensor networks. Advanced medical sensors can be used as nodes to monitor patients within a healthcare facility, a hospital or within their home to provide continuous real time data of patient's vitals by utilizing wearable technology. The biomedical sensors provide patient's health status by monitoring blood glucose levels, blood pressure, and other medical status' [2]. Wireless sensor networks are often used amongst critically ill patients to allow for prompt action by medical staff in emergency situations. Thus, optimising these systems is integral for both the health and safety of our society, emphasising the importance of solving the optimisation problems wireless sensor networks pose within research [6].

One purpose of this study was to assess the performance of our own implemented algorithm compared to that of a pre-existing optimisation algorithm. In the pages that follow, the report will cover a bit of background on the optimisation techniques for sensor networks that been implemented so far, before moving on to the theory section, where a reformulated model and proofs of its the convexity and smoothness are shown. The report then goes on to introduce the L2-penalty method, our own implemented algorithm and MATLAB's fmincon algorithm for constrained optimisation, a benchmark algorithm. The findings of their comparison are subsequently set forth and discussed in terms of efficiency and accuracy, ending with a conclusion.

2 Background

Optimising the deployment of the relay node can minimize the energy dissipation and improve efficiency and longevity of the wireless sensor network. Many optimisation techniques look at optimising the position of relay nodes, the number of relay nodes, and modulation sizes. Many optimisation techniques have been created to optimise the positioning of relay nodes to ensure full connectivity of the network [12]. Nature inspired optimisation algorithms have been used to update the relay node positions iteratively to achieve the best possible global position. Some include Moth Flame Optimisation (MFO), Interior Search Algorithm (ISA), and Bat Algorithm (BA).

Moth Flame Optimisation (MFO)

Developed by Sayedali Mirjalili in 2015, it is a population-based algorithm that looks at the movement of moths when attracted to light sources [11]. Moths generally move in a linear path when far from light sources, and then move in a spiral motion when they are more closely positioned to light. The key element is in the way the moths update their position around the light source. If a better position/solution is found, the current location of the moth is updated as every moth tried to find a better solution.

Interior Search Algorithm (ISA)

Proposed by A. Gondomi in 2014, this algorithm is inspired by artistic techniques within design [7]. The process begins from the bounds and borders of the problem and slowly moves towards the centre. The algorithm behaves in a series of steps where research is first conducted, followed by analysing results and then tailoring the knowledge into beatification process. ISA has been used repetitively to solve many optimisation problems.

3 Theory

Various methods and algorithms have been developed so far to solve unconstrained and constrained optimisation problems. A lot of them require convexity, differentiability, smoothness or numerous other imperative conditions to hold on the objective functions or even on the constraints before they can be applied. In the following section, it is proven that the problem of finding an optimal location for a relay modeled as an unconstrained problem, is not C^1, and thus not smooth. The problem is however convex and we are going to prove that as well.

For $s \in R^2$, as the location of the relay and X is a set of n points in R^2, representing the sensors:

$$P(s) = \sum_{x \in X} \|s - x\|^2 + \max_{x \in X} \|s - x\|^2 \tag{1}$$

Proving that $P(s)$ is not (C^1):

In general, the max function is not smooth and therefore not (C^1).

Below, we are going to illustrate an example, with 3 sensors in the plane. $X = x_1, x_2, x_3$ and $s = (s_1, s_2)$ The plane is divided into 3 regions A, B, and C.
Region A is the one on the bottom left without any points x, region B is the one on the right, with points x_2 and x_3, and region C if the top left region with point x_1

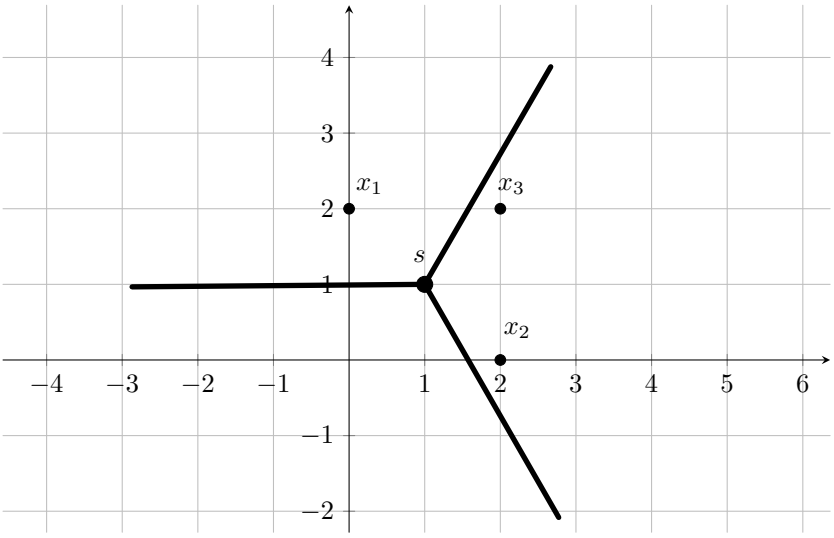


Figure 2: Illustration of Non-differentiable points

$$P(s) = \|s - x_1\|^2 + \|s - x_2\|^2 + \|s - x_3\|^2 + \max_i \|s - x_i\|^2 : x_i \in X \tag{2}$$

We are now going to demonstrate that P(s) is not continuously differentiable on the black lines for this particular example:

Assume that s is inside region A and not on the boundary, then s is furthest away from x_3 than from either x_1 or x_2 and therefore
$\max_{x \in X} \|s - x\|^2 = \|s - x_3\|^2$

$$P(s) = \|s - x_1\|^2 + \|s - x_2\|^2 + 2\|s - x_3\|^2$$
$$\nabla P(s) = 2(s - x_1) + 2(s - x_2) + 4(s - x_3)$$

As $s \to (1, 1)$ from the interior of region A,

$$\nabla P(s) \to (-4, -4)$$

In the same way, s is now assumed to be inside region B and not on the boundary. s is thus furthest away from x_1 than from either x_2 or x_3 and therefore:

$$\max_{x \in X} \|s - x\|^2 = \|s - x_1\|^2$$

$$P(s) = 2\|s - x_1\|^2 + \|s - x_2\|^2 + \|s - x_3\|^2$$
$$\nabla P(s) = 4(s - x_1) + 2(s - x_2) + 2(s - x_3)$$

As $s \to (1, 1)$ from the interior of region B,

$$\nabla P(s) \to (0, -4)$$

The different values of the gradient of $P(s)$ when approached from 2 different directions show that the limit of the gradient doesn't exist as $s \to (1, 1)$, meaning that $\nabla P(s)$ is not continuous and $P(s)$ is not differentiable at $(1, 1)$.

Thus, $P(s)$ is not continuously differentiable (C^1).

As was pointed out above, $P(s)$ is convex. From the book Convex Optimisation of Boyd and Vandenberghe, the following are proven [5]

- Norms are convex
- squaring a convex non-negative function gives a convex function
- point-wise max function preserves convexity
- sum of convex function is convex

Therefore $P(s)$ is convex

3.1 Reformulation of the unconstrained problem

Since the problem is not C^1 and thus not smooth, we are going to remodel it into constrained program with a smooth objective. The reformulated problem will still be convex and we are going to prove that at the end.

To convert the unconstrained problem into a constrained one, we introduced a new variable d:

For $X = \{x_1, x_2, ..., x_n\}$

$$\min \ P(s) = \sum_{x_i \in X} \|s - x_i\|^2 + \max_{x_i \in X} \|s - x_i\|^2$$

Taking the max out to the front:

$$\min \max_{x_i \in X} \ \{ \ \sum_{x_j \in X} \|s - x_j\|^2 + \|s - x_i\|^2$$

Then, introducing new variable d:

$$\min \ d \tag{3}$$

$$\text{s.t.} \ \sum_{x_j \in X} \|s - x_j\|^2 + \|s - x_i\|^2 - d \leq 0 \ \text{ for all } x_i \in X \tag{4}$$

There are now n constraints, one for each $x_i \in X$ and variable d can be seen as the upper bound for $\sum_{x_j \in X} \|s - x_j\|^2 + \|s - x_i\|^2$ for each of the constraints g_i.

Since d is linear, we can deduce that the objective is C^1 and smooth.

As for convexity, from (ref) above, we showed that $\|s - x_j\|^2$ is convex and so, $\sum_{x_j \in X} \|s - x_j\|^2$ is also convex and seeing that d is linear and thus convex, we can infer that the constraints $g_i(s, d) \leq 0$ are all a sum of convex functions and must thus be convex. The objective, d being linear is likewise convex.

Hence, the reformulated problem is still convex.

3.2 KKT Conditions

We now derive the KKT conditions for the model in task 2.

Each of the constraints are of the form

$$g_i(s, d) = \sum_{x_j \in X} \|s - x_j\|^2 + \|s - x_i\|^2 - d$$

and so the Lagrangian is

$$\begin{aligned} L(s, d, \lambda) &= d + \sum_i \lambda_i g_i(s, d) \\ &= d + \sum_{x_i \in X} \lambda_i \left(\sum_{x_j \in X} \|s - x_j\|^2 + \|s - x_i\|^2 - d \right) \end{aligned}$$

KKTa

Note that $\|s - x\|^2 = (s_1 - x_1)^2 + (s_2 - x_2)^2$.

$$\nabla L(s, d, \lambda) = \begin{bmatrix} \frac{\partial L}{\partial s_1} \\ \frac{\partial L}{\partial s_2} \\ \frac{\partial L}{\partial d} \end{bmatrix} = \begin{bmatrix} 2 \sum_{x_i \in X} \left(\sum_{x_j \in X} (s_1 - x_{j1}) + (s_1 - x_{i1}) \right) \\ 2 \sum_{x_i \in X} \left(\sum_{x_j \in X} (s_2 - x_{j2}) + (s_2 - x_{i2}) \right) \\ 1 - \sum_i \lambda_i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

KKTb

$$g_i(s^*, d^*) = \sum_{x_j \in X} \|s^* - x_j\|^2 + \|s^* - x_i\|^2 - d^* \leq 0 \quad \forall x_i \in X$$

$$\lambda_i^* \geq 0 \quad \forall i$$

$$\lambda_i^* g_i(s^*, d^*) = \lambda_i^* \left(\sum_{x_j \in X} \|s^* - x_j\|^2 + \|s^* - x_i\|^2 - d^* \right) = 0 \quad \forall x_i \in X$$

KKTc

There are no equality constraints in the model.

.

4 Algorithms

The first algorithm we created was designed to minimise the l_2 penalty function with the aid of the BFGS method. The l_2 penalty function and its gradient are below.

$$P_{\alpha_k}(s,d) = d + \frac{k}{2}\left[\sum_{x_i \in X}\left(\left(\sum_{x_j \in X}\|s-x_j\|^2 + \|s-x_i\|^2 - d\right)_+\right)^2\right]$$

$$\nabla P_{\alpha_k}(s,d) = \begin{bmatrix} 2k\sum_{x_i\in X}\left(\left(\sum_{x_j\in X}\|s-x_j\|^2+\|s-x_i\|^2-d\right)_+\right)\left(\sum_{x_i\in X}(s_1-x_{j1})+(s_1-x_{i1})\right) \\ 2k\sum_{x_i\in X}\left(\left(\sum_{x_j\in X}\|s-x_j\|^2+\|s-x_i\|^2-d\right)_+\right)\left(\sum_{x_i\in X}(s_2-x_{j2})+(s_2-x_{i2})\right) \\ 1-k\sum_{x_i\in X}\left(\left(\sum_{x_j\in X}\|s-x_j\|^2+\|s-x_i\|^2-d\right)_+\right) \end{bmatrix}$$

l_2 method pseudo-code

1. Set the tolerances for the BFGS method.
2. Set the number of sensors to be placed on the map and the range in which they will lie.
3. Generate random positions for the sensors and the initial point (s,d), which is to be optimised, using a uniform distribution.
4. Set $\alpha_k = 2k$
5. Employ the BFGS method on the l_2 penalty function to minimise (s,d).
6. Repeat steps 4 and 5 with increasing k in α_k until the norm of the difference of successive minima are less than 0.001, i.e. $\|(s^*,d^*)_{\alpha_k} - (s^*,d^*)_{\alpha_{k-1}}\| < 0.001$.
7. Repeat steps 1 to 6 for different number of sensors.

The second algorithm we created was designed to minimise the constrained NLP found in task 2. For this we used the MATLAB function 'fmincon', which took the inputs of our objective function, an initial starting point, (s,d), and the constraints function.

fmincon method pseudo-code

1. Set the number of sensors to be placed on the map and the range in which they will lie.
2. Generate random positions for the sensors and the initial point (s,d), which is to be optimised, using a uniform distribution.
3. Create constraint function that is consistent with the number of senors.
4. For each $i \in [1,10]$ use fmincon, with the appropriate inputs, to determine the minimum from an initial point $(s,d)_i$.
5. Repeat steps 1 to 4 for different number of sensors.

5 Experimental Set-up

Having established and explained the 2 main algorithms that we used, namely Matlab's 'fmincon' algorithm for constrained optimisation and our own algorithm, the L2-penalty method, the next part of this report will focus on the experimental set-up that was used to compare them. Previous research has shown that the way algorithms are implemented and the respective parameter settings chosen can greatly influence performance and accuracy [4].

For both the algorithms, we used the same set of sensor locations, X and initial instances s and d. The set X was generated randomly from a set of uniformly distributed random numbers, within the interval 0 to 10. For the initial starting points, they are generated randomly as well, using the set X such that the initial location s of the relay is within the same range as X, while the initial d is just large enough such that the constraints are not satisfied. This allows us to feed only the infeasible points to our L2-penalty algorithm which is an exterior-point method, and furthermore, it decreases the average number of iterations, making the algorithms less computationally expensive since the initial points being not too far away from the domain in which the optimal can be found. However, we used the interior search algorithm within MATLAB's fmincon function which did not specify the type of inputs it required. Thus, we fed it the same infeasible initial instances as the L2-penalty, to be consistent in our comparison.

Our L2-penalty method uses the BFGS algorithm within it, which in turn uses the golden search algorithm. Several values and combinations of tolerances and step sizes were tested for both of the sub-algorithms' tolerances and peak performance (least time and number of iterations) was achieved with the respective tolerances of 10^{-4} and 10^{-9} and with step size 10. As for the stopping condition, we used a tolerance of 10^{-3}. For the Matlab's fmincon algorithm, we used the 'Interior-point algorithm' with default parameter settings. All in all, we compared an exterior point method, our L2-penalty algorithm with an interior point method, MATLAB's fmincon algorithm.

For the comparison of the algorithms, the MATLAB code for this computational study was executed on a Microsoft Surface laptop running Windows 10 with the following system specifications:

- **Processor:** Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71GHz
- **RAM:** 4.00 GB
- **System type:** 64-bit Operating System, x64-based processor
- **Graphics:** Intel(R) HD Graphics 620

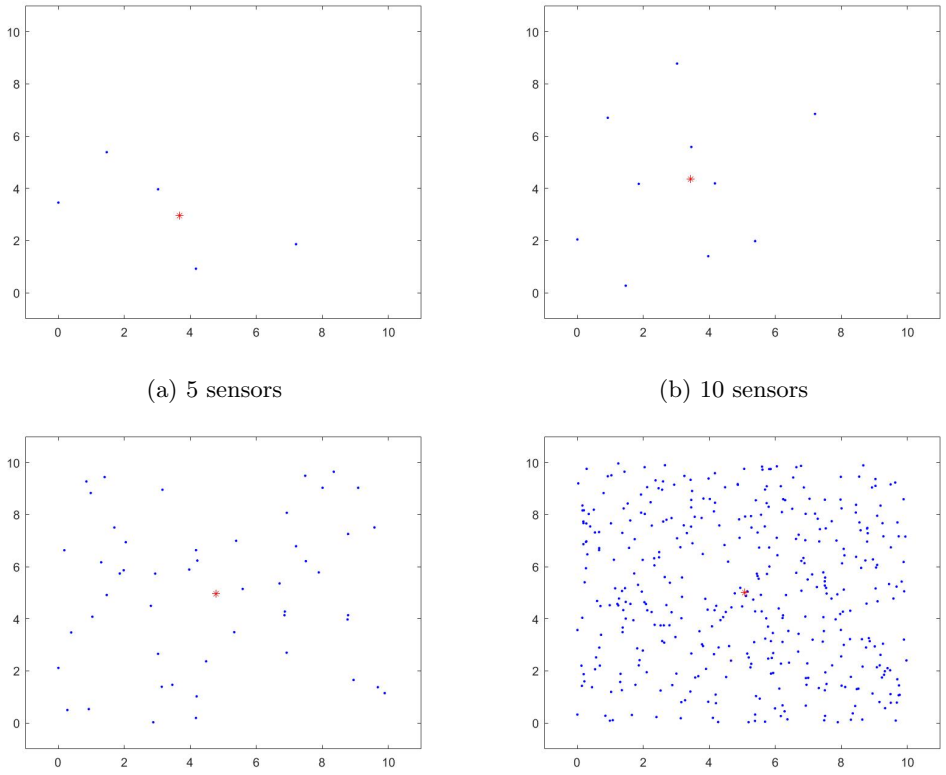
6 Experimental Results

Turning now to the experimental results, this section gives an account of the different ways in which the L2-penalty method and MATLAB's fmincon can be compared. It explores the relationships between their convergence rates, their efficiency and accuracy.

Please note that there are missing values for the fmincon algorithm, omitted on some of the graphs, due to the fact that we obtained multiple different minimisers and different minimum function values for the sets X of 300 and 400. Further explanation is provided in the discussion sections.

6.1 Demonstration of the L2-penalty algorithm

The first set of scatter graphs showcases the optimal relay locations unearthed by the L2-penalty algorithm for random sensor set X of size 5, 10, 50 and 400. It can be observed that as the size of the set of sensors increases, the algorithm's optimal relay location tends towards the center.



(a) 5 sensors

(b) 10 sensors

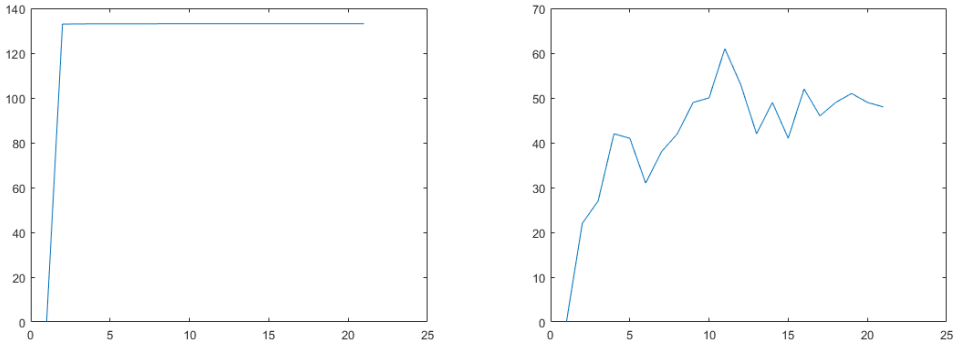
(c) 50 sensors

(d) 400 sensors

Figure 3: sensors and relay locations as found by the L2-penalty algorithm

6.2 The L2-penalty algorithm

In order to assess the performance of the L2-penalty method independently, both the f values for the penalty functions and the number of iterations K were plotted against the penalty parameter alpha, ($\alpha_k = 2k$), for a set X of size 10.



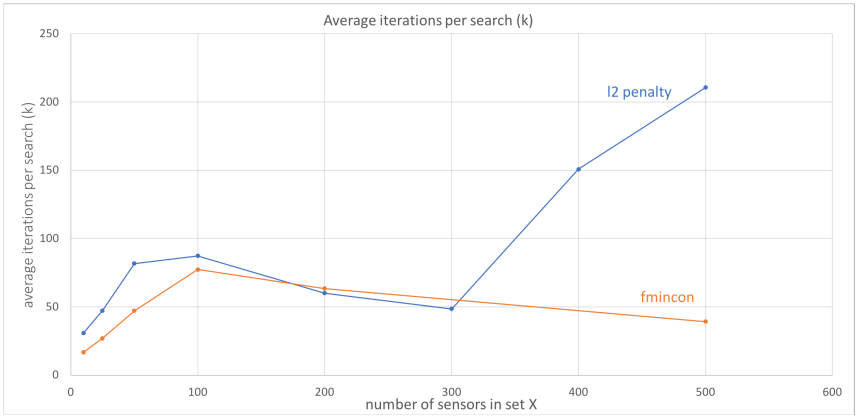
(a) Plot of F value

(b) Plot of K value

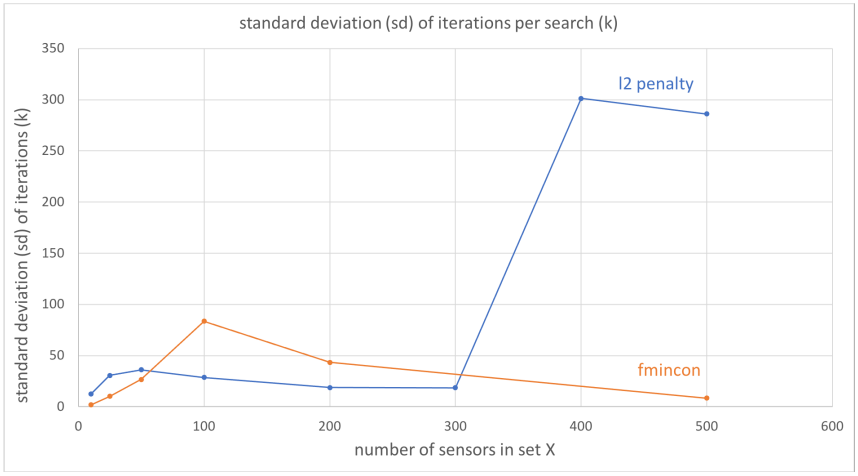
Figure 4: Plots of F values and iteration numbers for each penalty parameter alpha

6.3 Efficiency comparison

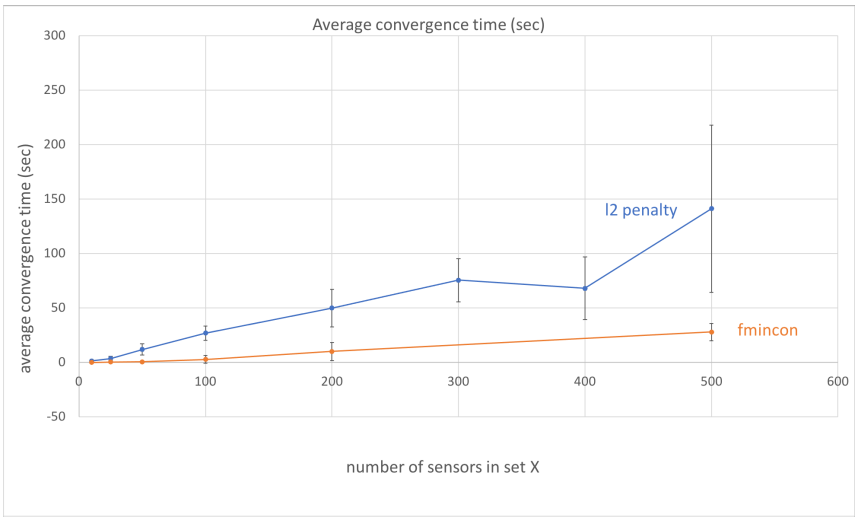
Figure 5 illustrates a comparison of the efficiency of the L2-penalty algorithm and MATLAB's fmincon for the average number of iterations and the average times of the 10 different initial instances. The standard deviation for each was also included, with the average number of iterations requiring a separate graph to display its standard deviation due to overlapping.



(a) Average iterations



(b) Standard deviation of average iterations

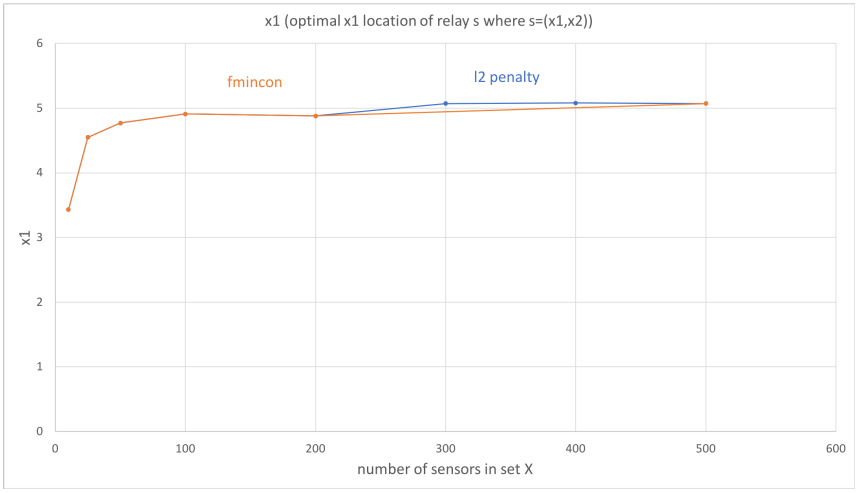


(c) Average time

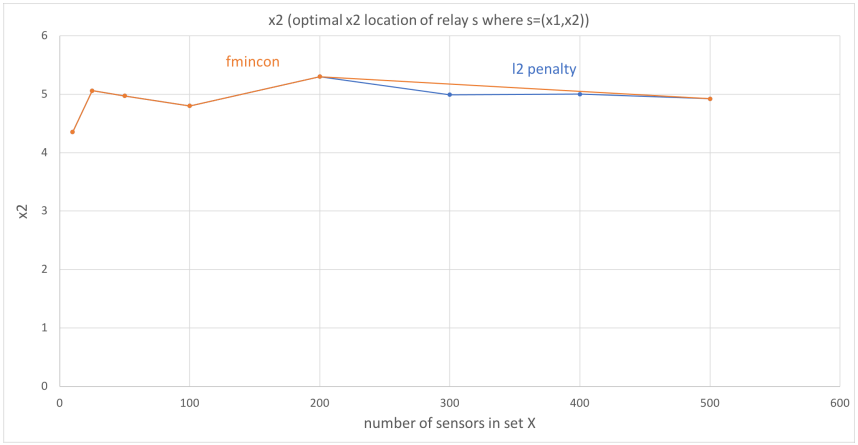
Figure 5: Efficiency plots for number of iterations and time

6.4 Accuracy comparison

Figure 7 gives an overview of the accuracy of the 2 algorithms, with each variable s_1, s_2 and the penalty function value, plotted on their own graphs.



(a) Accuracy of s_1 point



(b) Accuracy of s_2 point

Figure 6: Accuracy plots for the location of relay, $s = (s_1, s_2)$

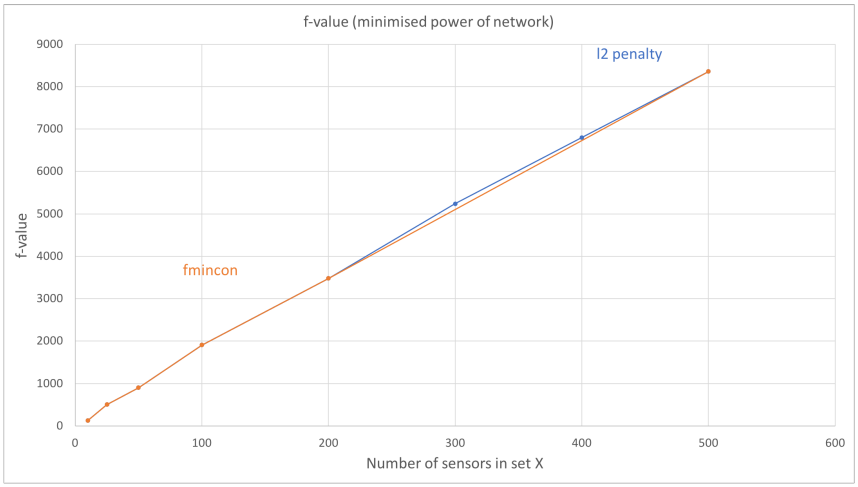


Figure 7: Accuracy plot for the f value

6.5 Tabled results for L2-penalty and Fmincon

The following tables provides a summary of the results obtained:

Results using L2 method with BFGS

No. of sensors	Optimal value of function	Minimiser	No. of times found	Average number of iterations	Average time per search (s)
10	133.16	(3.43, 4.35)	10	30.8	1.37
25	505.77	(4.55, 5.06)	10	47.3	3.63
50	904.77	(4.77, 4.97)	10	81.8	12.04
100	1910.63	(4.91, 4.8)	10	87.4	26.96
200	3479.83	(4.88, 5.3)	10	60.1	50.03
300	5240.96	(5.07, 4.99)	10	48.6	75.52
400	6798.98	(5.08, 5.00)	10	150.9	68.04
500	8356.23	(5.07, 4.92)	10	210.6	141.26

Table 1

Results using MATLAB function 'fmincon'

No. of sensors	Optimal value of function	Minimiser	No. of times found	Average number of iterations	Average time per search (s)
10	133.16	(3.43, 4.35)	10	16.8	0.089
25	505.77	(4.55, 5.06)	10	27	0.36
50	904.77	(4.77, 4.97)	10	47.1	0.65
100	1910.64	(4.91, 4.8)	10	77.4	2.79
200	3479.84	(4.88, 5.3)	10	63.4	10.11
300	5240.97	(5.07, 4.99)	6	117.83	44.79
”	8868.46	(4.07, 4.79)	1	371	142.91
”	6810.96	(4.98, 6.46)	1	336	141.38
”	7717.51	(4.43, 5.32)	1	371	139.33
”	6004.14	(4.97, 4.82)	1	367	144.48
400	6798.99	(5.08, 5.00)	8	36.5	15.61
”	10550.46	(4.24, 5.15)	1	369	251.15
”	10636.86	(3.98, 5.37)	1	370	250.40
500	8356.24	(5.07, 4.92)	10	39.3	27.93

Table 2

7 Discussions and Results

Our results showed that our algorithm, using the l2 penalty method, was able to run and converge to feasible solutions for each set X of 10, 25, 50, 100, 200, 300, 400 and 500 sensors, demonstrating its ability to handle sets of n sensor locations including large instances. However, the inbuilt MATLAB optimisation function, fmincon, produced inconsistent results for sets X of 300, and 400 sensors, giving multiple optimal values and minimisers for each set (Section 6.5, Table 2). As the constrained problem is a convex one, only one minimum objective value should have been found, which only occurred with sets X of 10, 25, 50, 100, 200 and 500 sensors by fmincon. Due to this, our algorithm proved to be more robust than fmincon. It was difficult to determine why the fmincon function delivered inconsistent results.

Efficiency:

In terms of efficiency, the fmincon function was shown to be a lot faster than our algorithm having on average a faster convergence time (sec) than the l2 penalty method. The efficiency graph (6.3, c) of average time vs the number of sensors showed that fmincon proved to be faster with much smaller average times compared to our algorithms, even with larger sets of X. It is visible from the error bars that our algorithm's convergence times deviated much more with larger sets of X whilst fmincon's convergence times were more consistent throughout. The graph of average iterations (6.3, a), showed that fmincon has a better convergence rate, than the l2 penalty method, especially with larger sets of sensors X. Both the l2 penalty function and fmincon shows an increase in average iterations for sets of X under 100, and then slightly decrease with larger sets. The l2 penalty function then increases with much larger sets of over 300 whilst the fmincon one decreases assuming that it keeps the same trend for the sensor sets X of 300 and 400, indicating our algorithm's inability to perform quickly with large data sets. It is also important to consider the graph indicating the standard deviation of iterations per search (6.3, b), as although our method produced consistent iteration results with smaller X sets, the data then largely deviated with larger X sets compared to fmincon, demonstrating our algorithms inability to be consistently efficient under large data sets. However, as fmincon produced inconsistent results for 300 and 400 sensors, it is difficult to accurately compare the two algorithms to draw conclusions for the sensor sets X of 300 and 400. Overall however, the fmincon function performs much better in terms of efficiency.

Accuracy:

In terms of accuracy, the l2 penalty function proved to be quite accurate as it produced the same results as fmincon in finding the optimal relay location and minimised power of the network. Despite the sets X of 300 and 400 sensors, it can be seen from the graphs (6.4, a and b) that the minimisers found for each algorithm were around the point (5,5). Since we used a range of 10 and a uniform distribution to randomly allocate the sensor locations our results makes intuitive sense that the relay locations tend towards the point (5,5), the midpoint of this range, as the number and span of sensors go up. This can be seen in the graphs in Figure 3. The graph of f values (6.4, figure 7) which depicts the minimised power of the network, increases with larger sets of X in both the l2 penalty function and fmincon, which again follows intuition, as a network with more sensors would require more power to operate. Overall, using the fmincon function as benchmark, our algorithm produced accurate optimal results.

Limitations:

Additionally, a potential downside in our experimental setup was that set X was generated randomly from a uniform distribution on the interval 0 to 10. Our plane was then restricted to a square of length 10 for each test which can be considered a small scale. However, we can ultimately scale or project these points to fit larger planes in order to suit larger networks. Another limitation is that we used only 10 initial starting points for s and d for each set X of n sensors and that can be considered a relatively small number. By increasing the number of initial starting points, this could

have resulted in smaller standard deviations in our l2 penalty results, since standard deviation is inversely proportional to n. However anything larger than 10 would have taken much longer to run. We used sensor sets of up to 500, and this may not have been a large enough set, depending on the application of the wireless sensor network. Lastly, due to the inconsistent results of fmincon with sets 300 and 400 sensors, this caused our fmincon lines on the graphs to have two less points plotted than the l2 penalty lines. This may have caused inconsistency when making comparisons, as our findings were based on more information on the trends of the l2 penalty method.

8 Conclusion

In conclusion, it appears as though both algorithms employed tend to give the same minimums. The fmincon algorithm gave better performance in efficiency relative to the l2 penalty one. However, it also produced inconsistent results for the sensor sets X of 300 and 400, leading to multiple minimiser and minimum f-value pairs. Further investigations should be conducted to determine the cause of this inconsistency. It could be due to the infeasibility of the initial points, or possibly the sets of sensor positions. In future computational studies, more research should be done to understand the distribution and magnitude of the sensor networks, since in practice, it may not be uniform and it may not be as big as 500, or it may even be bigger. A better understanding may have led to the construction of a more efficient algorithm.

Bibliography

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002. ISSN 1389-1286. doi: https://doi.org/10.1016/S1389-1286(01)00302-4. URL https://www.sciencedirect.com/science/article/pii/S1389128601003024.

[2] H. Ali. Energy efficient hierarchical clustering mechanism for wireless sensor network fields. *International Journal of Computer Applications*, 153:42–46, 11 2016. doi: 10.5120/ijca2016912130.

[3] T. Arampatzis, J. Lygeros, and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. pages 719 – 724, 07 2005. ISBN 0-7803-8936-0. doi: 10.1109/.2005.1467103.

[4] A. Blelly, M. Felipe-Gomes, A. Auger, and D. Brockhoff. Stopping Criteria, Initialization, and Implementations of BFGS and their Effect on the BBOB Test Suite. In *GECCO '18 Companion*, Kyoto, Japan, July 2018. doi: 10.1145/3205651.3208303. URL https://hal.inria.fr/hal-01811588.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. doi: 10.1017/CBO9780511804441. URL https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf.

[6] A. Djedouboum, A. ARI, A. Gueroui, A. Mohamadou, and Z. Aliouat. Big data collection in large-scale wireless sensor networks. *Sensors*, 18, 12 2018. doi: 10.3390/s18124474. URL https://www.researchgate.net/figure/General-architecture-of-a-wireless-sensor-network-WSN_fig1_329012374.

[7] A. Gandomi. Interior search algorithm (isa), 01 2014. URL https://www.researchgate.net/profile/Amir-Gandomi-2/publication/292747832_Interior_search_algorithm_ISA/data/56b0eef408ae5ec4ed484363/ISA.m.

[8] U. Jain and M. Hussain. Securing wireless sensors in military applications through resilient authentication mechanism. *Procedia Computer Science*, 171:719–728, 2020. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2020.04.078. URL https://www.sciencedirect.com/science/article/pii/S1877050920310462. Third International Conference on Computing and Network Communications (CoCoNet'19).

[9] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras. Applications of wireless sensor networks: An up-to-date survey. *Applied System Innovation*, 3(1), 2020. ISSN 2571-5577. doi: 10.3390/asi3010014. URL https://www.mdpi.com/2571-5577/3/1/14.

[10] M. Matin and M. Islam. Overview of wireless sensor network. In M. A. Matin, editor, *Wireless Sensor Networks*, chapter 1. IntechOpen, Rijeka, 2012. doi: 10.5772/49376. URL https://doi.org/10.5772/49376.

[11] S. Mirjalili. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems*, 89:228–249, 2015. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2015.07.006. URL https://www.sciencedirect.com/science/article/pii/S0950705115002580.

[12] S. Sapre and M. S. Optimized relay nodes positioning to achieve full connectivity in wireless sensor networks. *Wireless Personal Communications*, 99, 04 2018. doi: 10.1007/s11277-018-5290-8.