University of Dublin

TRINITY COLLEGE

Analysis of Integral Equation-based Methods for Radio Coverage Prediction

Andrew Knox-Shiels
Master in Computer Science
Dissertation April 2020
Supervisor: Dr. Eamonn O'Nuallain

School of Computer Science and Statistics

O'Reilly Institute, Trinity College, Dublin 2, Ireland

Declaration

I hereby declare that this project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: [redacted] Date: 30/4/2020

Abstract

This dissertation explores using a novel means of accelerating the Electric Field Integral Equation called the Field Extrapolation Method, to predict radio coverage in a rural environment. The method reduces computation time by grouping points into flat plates, onto and from whose centres scattering occurs. The accuracy trade-off is shown to be acceptable versus the computation time reduction, and the Forward-Backward method is shown to increase accuracy in limited circumstances. Whether an accuracy gain affords actual improvements in computation speed over single forward scattering passes by permitting larger group sizes is a question left open, along with the exact implementation of the Field Extrapolation Method's calculation of K, which represents the aggregation effect of non-central points in plate centres.

Urban terrain profiles are shown to be unsuitable for every permutation of method tested, and the orientation of the plate doesn't improve accuracy, but greatly diminishes computation time; both points are in keeping with the literature that states the Electric Field Integral Equation and Forward-Backward Method (hence their combination and derivatives thereof) are unsuitable for terrains with greater-than small grazing incidence angles, where back scattering and other fast fading effects are significant.

Acknowledgements

The author would like to thank fellow student Maksim Maiberg for the assistance in calculating K, without which the discrepancy between how to calculate a K near 5, and the formulation of K as it appears in the paper, would have remained hidden.

Contents

1 Introduction

Hypothesis

To Verify the Field Extrapolation Method's Accuracy in Rural Terrain Profiles and Investigate Applicability to Urban Profiles.

1.1 Electric Field Integral Equation

The Electric Field Integral Equation is the foundation of the Field Extrapolation Method. It is this equation that the method aims to accelerate within acceptable accuracy bounds. It is the first step in going about testing the hypothesis, as it forms a ground truth for the Field Extrapolation Method. In testing it with Forward Scattering and Forward-Backward methods, as well as calculating the numerically exact solution through MATLAB's Conjugate Gradient Squared function, ample values are obtained against which to weigh the success or failure of the Field Extrapolation Method, and subsequent exploratory variations. An approximate method for interpolating the terrain profile was deployed as it was easier at the time and generated accurate results, but it would prove to be one approximation too many for the Field Extrapolation Method to bear. A visualisation of the problem can be found in Appendix A.1.

1.2 Field Extrapolation Method

The Field Extrapolation Method seeks to improve the speed of execution of the Electric Field Integral Equation by approximating a constant value K for groups of points, and using that multiplied by the group centre's scattering radiation to approximate the scattering from all points of that group. An initial sample value 5 was given for debugging the method's function. Variations of this method included testing different values for the constant value K, calculated by stricter and looser interpretations of its formula, and dropping its constant value for a per-iteration value based on the angle of incidence made by scattering group to receiving group.

1.3 K

The calculation of K was investigated as initial results differed from the sample value 5. The function was altered in several ways, with the most extreme alteration stripping away its constant nature and observing the impact of the incident angle. Angle inclusion was not expected to improve results, and it was included with an eye to testing urban environment applicability. A visualisation of K can be found in Appendix A.2.

1.4 Terrain Profiles

The methods are first tested on a gently undulating rural terrain profile, then applied to a mock urban terrain profile. The gaps between the 10 metre measurements were first interpolated across X only, but the urban profile simulating a pair of buildings demanded a more accurate interpolation method, as interpolating across x alone resulted in vertical walls being missed. It would become plain that each point along a wall needed to be accounted for. The Electric Field Integral Equation and the Field Extrapolation Method (including and excluding angle) were deployed on the both terrain profiles.

1.5 Rural with Strict Interpolation

Out of curiosity and dissatisfaction with the Forward-Backward version of the Field Extrapolation Method's performance, the more accurate terrain interpolation method was used, and the Field Extrapolation Method re-examined. A new ground truth was necessary, as comparing strict terrain Field Extrapolation Method to loose terrain Electric Field Integral Equation would be unscientific, so the Electric Field Integral Equation was also re-run.

1.6 Conclusions

The Field Extrapolation Method turned out to be a good fit for gently undulating rural terrain. The Forward-Backward showed convergence in the end, but is much more sensitive to input errors than the Electric Field Integral Equation and needs to be handled carefully. What Forward-Backward promises with the Field Extrapolation Method is the provision of further time-saving measures can be gained while maintaining good accuracy.
The incidence angle turned out to be of no import in the best case scenario, and starkly detrimental in the worst, and the Field Extrapolation Method with and without Forward-Backward is just as unsuitable for urban environments as the authors of the Electric Field Integral Equation and Generalised Forward-Backward Method both concede; the methods perform poorly where back scattering is significant and are unfit for purpose.

2 Literature Review

2.1 Cognitive Radio

Cognitive Radio devices are those which are able to sense their environment and make decisions based on their sensed data, regulations to abide by (which frequencies are strictly off-limits and which have Primary Users which, when no Primary User transmissions occur a Secondary User may operate), the device's own policies (battery power and memory capacity restrictions).

One of the reasons for this is the relatively rare periods of transmission in a frequency band by the incumbent Primary User, with the paper [1] citing an actual usage of 5.2% for frequency bands below 3 GHz in the United States. A Cognitive Radio is one which can detect when the incumbent is or is not transmitting, and will be able to exploit the fallow spectrum for its own purposes while avoiding interference with the incumbent.

The network support comes from Radio Environment Maps. These can be local (region around the user device) or global (the geographic region a user device is in such as Dublin City). Global Radio Environment Maps can be aggregated from the local Radio Environment Maps within its boundary. The Radio Environment Map is an abstraction of real-world radio scenarios, characterising the radio environment of a Cognitive Radio in several domains including geographical features, regulations and policy from which Situational Awareness is enabled at the device. It is a spatio-temporal database and supports exchanges between local devices and the network to maintain concurrency between all local Radio Environment Maps and the global Map.

The text [2] states Situational Awareness, key to several features of Cognitive Radio such as knowing the radio scenario, intent of user, regulations and the device's power supply, is obtained by direct observation, inference from network support and by combining analysis of local terrain propagation models with existing database structures.

Regarding observation, the cognitive radio device needs to have information about the radio propagation characteristics of the region the Map describes. The characteristics include those of large scale fading (path loss and shadowing). These can be effectively modelled by software

using terrain profiles of two dimensions (3 if small scale fading effects like multipath are to be included). The software models can be at least as accurate as empirical models, and can be executed remotely or in the theatre of operations. As updates between local and global Maps can be achieved, if the terrain profile of an area changes, a new propagation model can be run and its output propagated across the network as an update to the global and to particular devices' local Radio Environment Map(s).

[2] describes how a legacy radio can leverage the Radio Environment Map to achieve some characteristics of Cognitive Radio, such as by the network being aware of the device's location and the interference patterns at that location. It can then instruct the radio to use a different PHY and MAC layer where necessary. The availability of network support enables this, and it augments a purpose-built Cognitive Radio's capabilities. With a Radio Environment Map, a Cognitive Radio can predict radio performance in addition to its own awareness and learning from its own actions. It then can update the Radio Environment Map after it makes or schedules its own decisions.

5G brings challenges to the Cognitive Radio field, as it incorporates a massive and heterogeneous superset of the 4G network. Cellular telephones will be in the same network as devices like IoT sensors in smart buildings, or those supporting vehicle-to-anything networks. The paper [4] details the need for Radio Environment Maps to be kept up-to-date in a timely manner to efficiently support this vast network. It states intensive interpolation processing, transmitter localisation, change in measurement-capable devices, and changes to propagation models should pass to Radio Environment Map in the core network, with it occurring in local nodes on periodic, on-demand, or emergency bases. The utility of a fast and accurate propagation model is apparent with the emergent network.

2.2 Electric Field Integral Equation

The Electric Field Integral Equation is a means of modelling radio propagation through a terrain profile, accounting for the large scale fading effects of path loss and shadowing. [6] states that using Integral Equation-based methods for modelling radio propagation was out of favour in the 1980s and 1990s due to the effectiveness of empirical methods, and the high computational requirements exceeding what was readily available.

In [3], an integral equation is derived for use with gently undulating rural terrain, and is deliberately ignorant of back scattering and other reflections from objects. It is so because the fast Rayleigh distributed fading arising from these reflections is "impossible and undesirable to predict in detail". The integral equation is considerate of slow, large scale fading from path loss and shadowing. back scattering was tested numerically and shown to have a negligible contribution from regions beyond the receiver, while side-scattering can be ignored for the narrow band signals under experimentation.

The paper justifies the Perfect Electrical Conductor assumption by stating for vertical polarisation it corresponds to a reflection coefficient of -1, which happens to be the case in practice as the grazing angles are so small that the reflection coefficient is almost -1. It notably states that the horizontal polarisation is approximately equal to vertical in the microwave region (1 GHz to 300 GHz) where the Norton ground wave can be ignored.

The paper also states that integral equation methods have been tried before. In the 50s, Hufford derived an equation based on the Green's theorem, but variations of it resulted in numerical instabilities above 10 MHz. The authors propose their integral equation would suffer no such problems. In the experiments for path loss over a wedge at 10 MHz and 1 GHz the equation agrees well with the Uniform Theory of Diffraction method, while the low frequency experiments for their gently undulating rural terrain (143.9 MHz and 435 MHz) performed extremely well. Higher frequency experiments at 970 MHz and 1.9 GHz were less accurate, but still generated acceptable results.

2.3 Field Extrapolation Method

The Field Extrapolation Method, introduced in [5], is a development of the Electric Field Integral Equation that reduces computation time by grouping points into plates, whose centres receive and scatter radiation incident upon them. With the constant K standing in for the contributions of the other points in the group, accurate results are generated with greatly reduced computation time. In the formulation of this constant, the angle of the group is ignored due to the grazing angles of incidence, like in [3]. The phase is also forced to zero as the scatterer is electrically massive. The remaining parameters of intra-group impedance matrix elements and distances between points are constant across all groups, allowing for the assumption of K's constancy.

in 2005, when [5] was written, computational power had advanced so far that these equations could be run on a common workstation. The Field Approximation Method, was run in 0.05 seconds on a PC with a 2.2 GHz processor. During the intermittent time, computers have become even more powerful with some mid-range processors possessing several times the CPU cores and double the clock speed, along with a myriad other hardware improvements.

The method is applied to the same gently undulating rural terrain profile for Hjorringvej used in[3], with its results standing in for a ground truth. The method proved to be very accurate with just a forward scattering implementation, and ran much more quickly than the numerically exact method (tens of minutes vs hours). It is stated that further work needs to be done in evaluating the effects of including small scale Rayleigh fading such as multipath for accuracy, as that gain would permit larger group sizes and faster computation times, so far ignored by the Electric Field Integral Equation.

2.4 Forward-Backward Method

The Forward-Backward Method is an extension of the Electric Field Integral Equation method that includes back scattering effects by splitting the surface current at each point into its forward and backward components. It performs the forward scattering sweep in the same way as the Electric Field Integral Equation, then incorporates these results in a backward sweep for back scattering effects. The incident field is calculated in the forward sweep only. One iteration of the method consists of a forward pass and a backward pass, and it is said in [7] that fewer than 10 iterations are required for convergence.

It then suggests using the Method of Moments to account for Perfect Electrical Conductor obstacles encountered; ships and rogue waves in its maritime case. It uses the same Perfect Electrical Conductor assumption for the ocean surface as [3] uses for land, but introduces new impedance values for the obstacles. These new values are added to the self-terms of the impedance matrix along the main diagonal corresponding to the obstacle, but leaving the lower and upper triangular matrices untouched. The method-of-moments block accounts for interactions between obstacle and surface, as well as interactions within the obstacle which the conventional Forward-Backward method discounts. The paper asserts these are necessary inclusions for the Forward-Backward method's convergence in the presence of obstacles, as without them convergence does not occur.

The method's additional requirements are storage of MxM elements for an obstacle's impedance matrix and the associated computation cost to calculate it, for every such obstacle. The paper demonstrates the accuracy and speed, with 6-10 iterations required for the residual error to reach between 10e-2 to 10e-3. However, this is on a surface with just one obstacle of 15 metres squared, which may not be transferable to an Urban environment with many irregularly shaped obstacles. A city may have hundreds of individual buildings with heights ranging from 10 metres to over 500 metres, each of which will have a different matrix that in most cases will not be square.

3 Overall Materials and Methods

3.1 Materials

The rural terrain profile used is the first 700 metres of a 3,840 metre span of gently undulating landscape from Germany. The surface is assumed to be a two-dimensional Perfect Electrical Conductor, where side scattering contributions are ignored. It is sampled at every 10 metres along the horizontal axis. In the 700 metre segment, the highest point is 390 metres and the lowest point is 291.27 metres, with that total descent occurring over 600 metres, yielding an average slope of -0.16455. An excerpt was used over the entire profile to reduce running time during debugging.

A second terrain profile is created for Urban Environment experimentation, simulating two rectangular buildings with three sections at ground level. The first building profile is 3 metres high and 2 metres across, the second is 10 metres high and 3 metres across. The ground level spaces before, between and beyond the buildings are 5, 13 and 27 metres respectively.

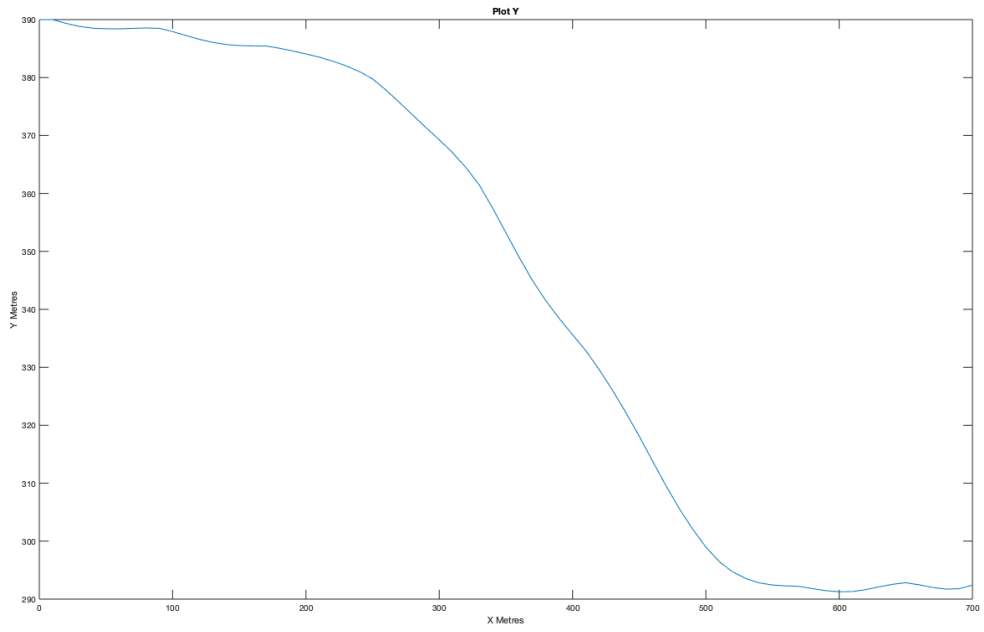


Figure 3.1: The 700 metres of the Rural Terrain Profile used in Rural Experiments

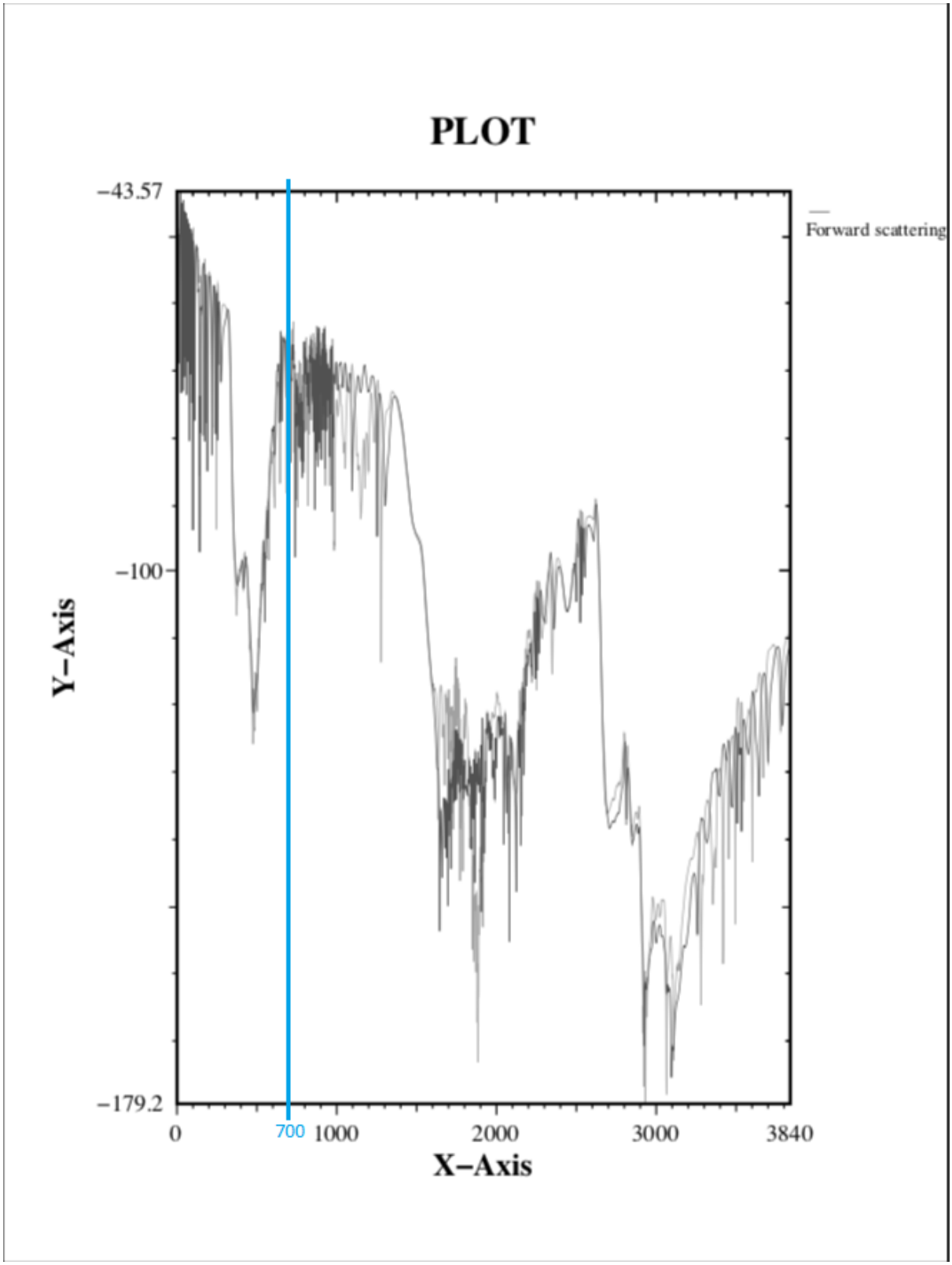


Figure 3.2: Field above the surface for the Rural Terrain Profile with 700 metre line marked

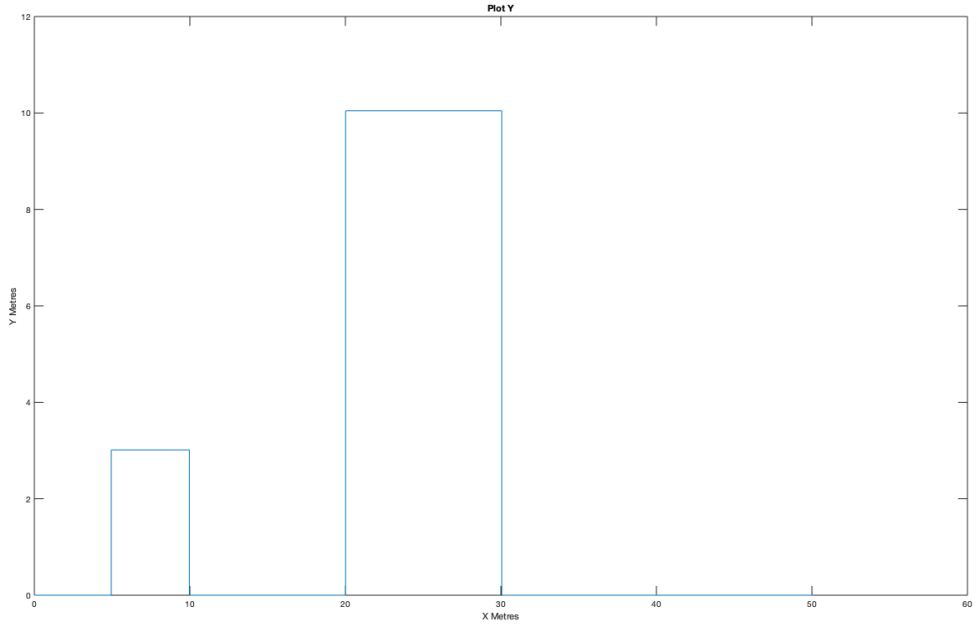


Figure 3.3: The mock Urban Terrain Profile used in Urban Experiments

3.2 Methods

The method under experiment is the Forward-Backward method, in turn applied to both the Electric Field Integral Equation and the Field Extrapolation Method. The Forward-Backward method involves calculating the total electrical field at every point along a surface by summing the incident field emanating from the transmitter with the forward scattering contributions from points between the point under consideration and the transmitter, and the back scattering contributions from points beyond the point under consideration.

Graphs and tables of Mean Absolute Error differences from the Ground Truth are used to judge efficacy. To find where Forward-Backward acceptably converges, it was run with 4, 7 and 10 iterations. The upper limit of 10 refers to [7], which states convergence should occur within 10 iterations, although in some Field Extrapolation Method cases extra accuracy at up to 20 iterations is investigated as such additional computation time is afforded.

Samples were taken at $\frac{1}{4\lambda}$ metre intervals. As the terrain profile was sampled every 10 metres along the x axis, points needed to be interpolated with $\frac{1}{4\lambda}$ metres between each point in order to generate the complete discretised surface. There are different methods for interpolation in the rural terrain profile and the urban terrain profile, both are shown in Appendix A.3.

In the rural profile, there was assumed to be a tolerance for using Δs $(\frac{\lambda}{4})$ sized segments along x, with the Y value for each point being assigned the previous point's value plus Δs times the

slope m of the current 10 metre "chunk". In other words, the x of the right-angle triangle was Δs and the y was $\Delta s \times m$. The urban profile had to be stricter, with Δs sized segments taken along the hypotenuse instead of the simpler way of the rural profile. The reason for the stricter tolerance is it considers large buildings with vertical walls, which have many points that would be missed by using x instead.

The frequency of the monochromatic signal used was 970 MHz. The Electric Field Integral Equation is

$$E(r) = \frac{\beta\eta}{4} \int_S J(r') H_0^{(2)}(\beta|r - r'|) dr' \tag{1}$$

Which through the Method of Moments is discretised into the matrix equation

$$E = ZJ \tag{2}$$

Where:

$$E = E(r_i) \tag{3}$$

$$Z_{ji} \approx \Delta s \frac{\beta\eta}{4} H_0^{(2)}(\beta|r_j - r_i|) \tag{4}$$

$$Z_{ii} \approx \Delta s \frac{\beta\eta}{4} \left(1 - j\frac{2}{\pi} \ln\left(\frac{1.781\beta\Delta s}{4e}\right)\right) \tag{5}$$

$$J_j = J(r_j) \tag{6}$$

$E(r_i)$ is the incident field at point i

Z_{ji} is the impedance between the point j and the point i

Z_{ii} is the impedance between the point i and itself

J_j is the surface current at point j

$\frac{\beta\eta}{4}$ is a constant that is ignored as it doesn't impact the shape of the graph

$H_0^{(2)}$ is the zeroth order Hankel function of the second kind

Δs is the sampling interval

β is the wavenumber

$|r_j - r_i|$ is the distance between points j and i.

The forward and backward entries for J and Z can be decomposed to

$$Z = Z_f + Z_{self} + Z_b \tag{7}$$

$$J = J_f + J_b \tag{8}$$

Which when combined with E = ZJ yields

$$Z_{self} \times J_{fwd} = E_{inc} - Z_{fwd} \times (J_{fwd} + J_{back}) \tag{9}$$

for forward scattering and

$$Z_{self} \times J_{back} = -Z_{back} \times (J_{fwd} + J_{back}) \tag{10}$$

for backward scattering.

The total number of forward and backward sweeps required to achieve convergence is typically less than 10.

The algorithm to calculate the electric field above the surface first calculates the surface current for each point. Using this current, the field above the surface for each point is calculated. The surface current is calculated in three ways during the experiments to follow:

1. Applying the Conjugate Gradient Squared method to find a numerically exact solution to the $E = ZJ$ problem
2. Performing a single pass of forward scattering
3. Performing multiple passes of the Forward-Backward method

Where the surface current is calculated by forward scattering, J is calculated by iterating over every point P and subtracting the sum of the contributions from points 1 to P-1 from the incident field at that point P. This summation is then divided by the self-impedance term to gain the value for J at point P.

For back scattering this process is done in reverse (P = N - 1 to 1), while the sum of back scattering contributions iterates from N to P + 1. In this case, the back scattering contributions are subtracted from the current value (initially 0 + 0i), without the incident field's inclusion so as to not double-count it. The resulting sum is incident field + forward scattering + back scattering for each iteration of Forward-Backward employed.

The field above the surface is calculated using one sweep of forward scattering. The methods under experimentation differ in the calculation of the surface current only, each method uses this same single forward sweep to calculate the field above the surface. When calculating the

sum of forward scattering contributions, the inner loop runs to point P, unlike the surface current's calculation running to P - 1. This is because the field is observed at 2.4 metres above the surface, so the current at surface point P does contribute to observation point P + (0, 2.4 m). The code that performs the surface current and field above the surface calculations can be found in Appendix A.4.

As the impedance matrix Z exists as a lower and upper triangular matrix (for forward and back scattering respectively), initial values can be calculated for the top-left element which has no forward scattering to it, and then propagated through the remaining rows through forward substitution. For back scattering, back substitution takes place from the opposite corner of the matrix.

For evaluating the results of the algorithms, the field above the surface in decibels is used instead of the surface current. The decibel measurement is achieved through the calculation

$$E_{tot}(dB) = 20 \log_{10} E_{tot} \tag{11}$$

1 is used implicitly in the denominator of the logarithm as the goal is to measure the power difference between the signal and the absence of a signal; no other transmitters are assumed to be present. The reason for using field above the surface instead of surface current is due to the ill-conditioned nature of the problem. A small change in the surface current, almost imperceptible to the human eye, can result in a large, plainly visible error in the field above the surface.

4 Experiments

4.1 Numerically Exact Method

The control for the future experiments is the numerically exact Conjugate Gradient Squared solution for $E = ZJ$. This control was used to judge whether a method was returning acceptable results and which method was most correct in the case of comparing multiple methods. It depends on the fact that with the $E = ZJ$ equation, E and Z are easily calculable. Using MATLAB's cgs(A,b) function, which accepts a matrix A and vector b, and attempts to solve the system of linear equations $Ax = b$, J can be calculated exactly by passing the impedance matrix M and incident field vector E as parameters A and b respectively. The resulting J is used to calculate a field E_{tot} above the surface to be used as a ground truth going forward.

The forward propagation values for Z form a lower triangular matrix, while the diagonal is composed of self-terms Z_{ii}. Using the fact that distance is always positive, and is the only variable in the formula for Z_{ij} for a given frequency, it becomes clear that the upper triangular portion for back scattering is a reflection of the lower triangular portion ($Z_{ij} = Z_{ji}$). This symmetry allows for a slight relaxing in computational requirements, as each calculation effectively renders two results.

The cgs(A,b) function was called without any preconditioners, and with MATLAB's default tolerance of 10^{-6}. The maximum iterations parameter was assigned 100 (it needed 78 iterations for convergence), as it did not converge within the default 20 iterations.

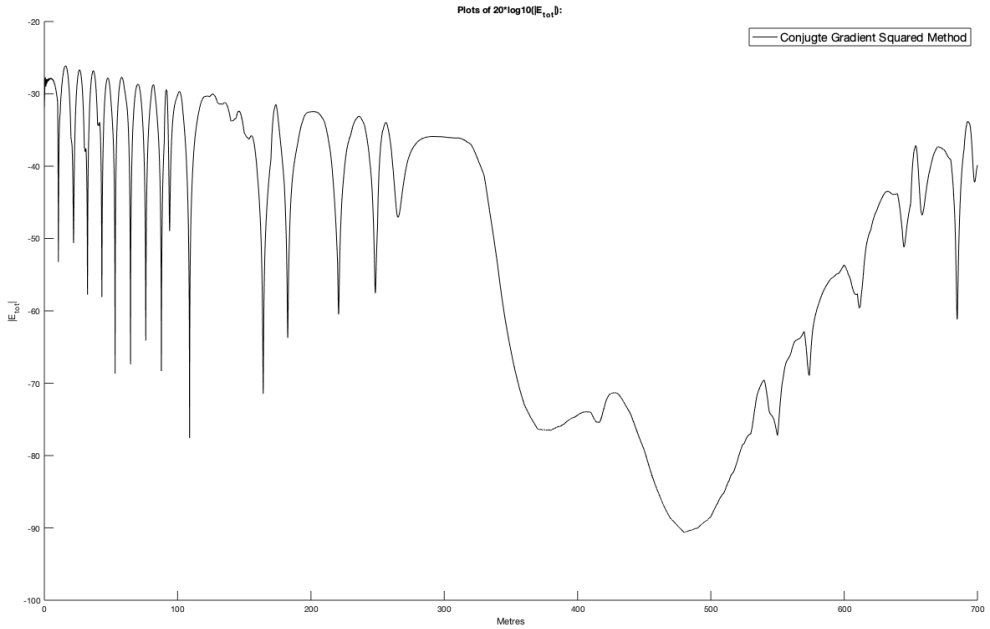


Figure 4.1: The Conjugate Gradient Squared method's result, taken to be the Ground Truth going forward

4.2 Electric Field Integral Equation: Forward Scattering

The first experiment with the Electric Field Integral Equation was a single pass of forward scattering. The aim was to ensure that some results close to the correct solution could be obtained by a single pass, which could be improved with the addition of back scattering and additional iterations. The method called for first calculating the surface current J at every point along the surface, then using those surface currents it evaluated the field above the surface.

The graph shows that the single forward scattering iteration of the Electric Field Integral Equation is a very close approximation for the exact field found using the Conjugate Gradient Squared method. In many places the two agree completely (where red occludes black), while their differences are small in the case of the trough.

Efficient use of computational resources is gained for the accuracy lost; the single iteration of forward scattering is much faster and requires far less memory than the Conjugate Gradient Squared method, as each entry for Z and E can be calculated quickly on the fly instead of stored in massive matrices. For this particular case, there are 9,060 points in the 700 metre profile, necessitating a 9,060 element vector and 9,060 x 9,060 element matrix for E and Z respectively, both composed of complex floats.

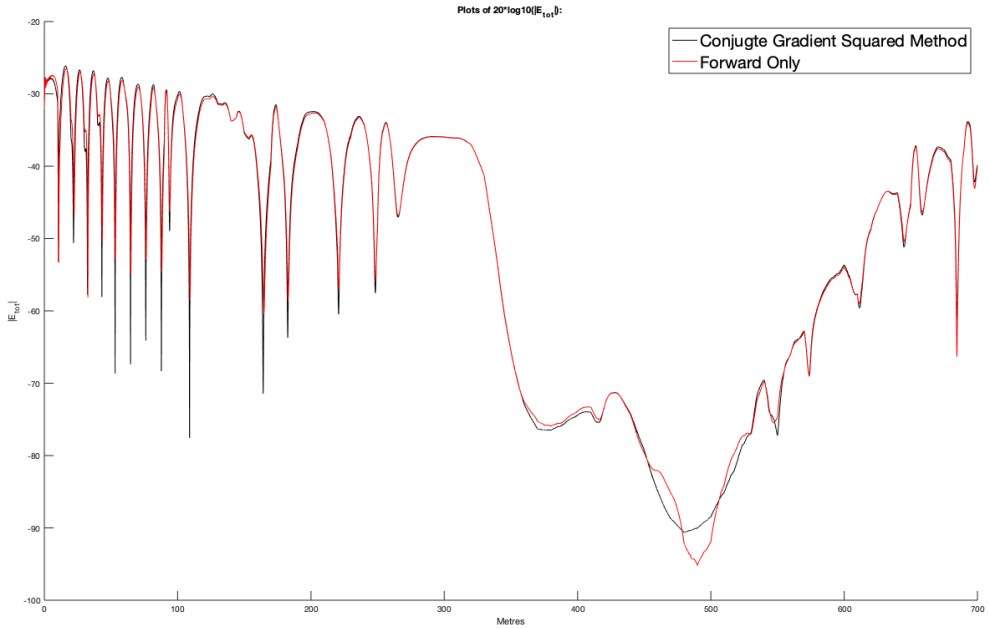


Figure 4.2: Single pass of Forward Scattering

4.3 Electric Field Integral Equation: Forward-Backward

To improve upon the results from the single forward sweep of the EFIE, the Forward-Backward method was employed. The aims of this experiment were to investigate whether or not it yielded a meaningful increase in accuracy over the single forward sweep, and to determine how many iterations it needs to converge.

The back scattering code was similar to forward scattering code, but with the incident field omitted and with iteration from final point to start point. In both cases, the current forward and back scattering contributions are considered, but are kept separate until the final iteration's completion where they are summed for the complete surface current.

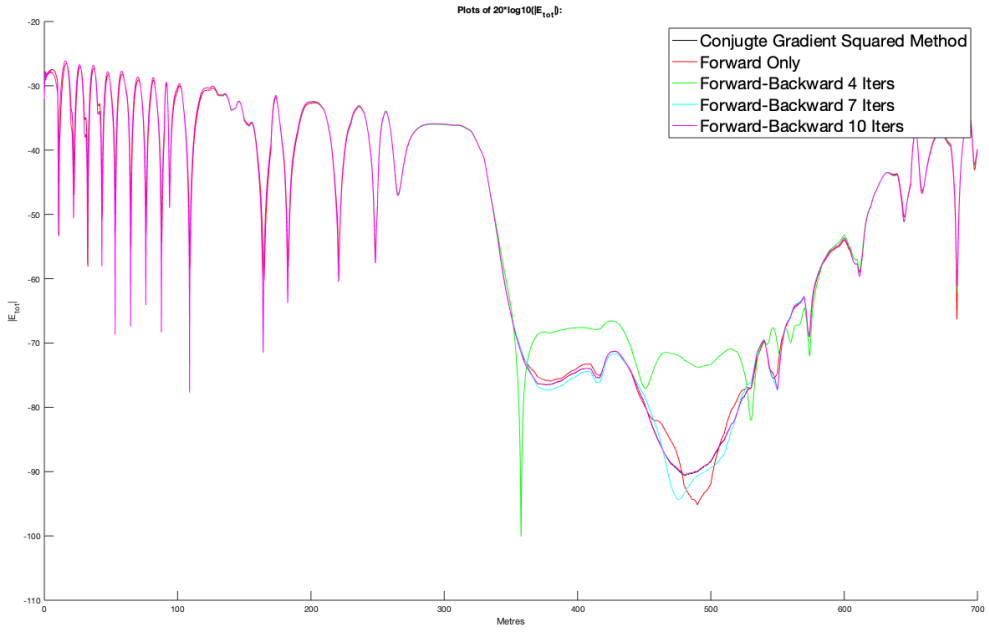


Figure 4.3: Forward Scattering and 4, 7, 10 iterations of Forward-Backward

Experiment	Mean Absolute Error(dB)
EFIE Forward Scattering	0.0400
EFIE Forward-Backward 4 Iterations	2.0874
EFIE Forward-Backward 7 Iterations	0.1532
EFIE Forward-Backward 10 Iterations	0.0081

The results here and in the earlier Forward Scattering experiment show that for the gently undulating rural terrain profiles, a single pass of Forward Scattering is a very good approximation of the total field above the surface. Forward-Backward gradually converges towards the exact solution, with 10 iterations having almost completely occluded the exact method's graph. This is due to the small effect of back scattering in such a terrain, with the only significant improvements being made at the lowest point of the trough, that corresponds to the terrain profile descending.

4.4 Field Extrapolation Method: Forward Scattering

As with the Electric Field Integral Equation, a single forward scattering sweep of the Field Extrapolation Method was first tested to ensure the method was performing correctly when applied to the rural terrain profile. A smooth region of 90 metres was introduced where forward and back scattering contributions are ignored, to avoiding contributions which may have been too great in the early stages. Beyond the smooth region, the volume of contributions was considered great enough to check an errant individual contribution.

The equations it uses are the same as in the Electric Field Integral Equation, but their application differs. Instead of visiting each point, the surface is divided into groups of 13 Δs segments. The size of the group depends on the terrain profile, and requires trial and error experimentation to calculate; 13 segments is the group size suggested by the author's supervisor. The aggregation point, upon which the incident field falls and which receives and transmits scattering from to the other groups respectively, is taken to be the group centre.

As stated in the paper [5], the groups are assumed to have uniform plane incident fields upon them. Continuity of the surface current is enforced on either side of the group and the surface current induced can be assumed to have a locally constant amplitude due to the surface's smoothness. The profile is extended, which artificially enforces the surface current's continuity across both ends of the profile. The surface current, having been induced by a uniform plane wave, can be estimated as

$$J_{jl'} = -J_{l'} \frac{Z_{l'l}}{Z_{jj}} e^{-j(\beta s_j \cos\theta_{l'} + \phi_{l'})} \tag{1}$$

with the resulting equation for the incident field being

$$E_l = \sum_{l'<l} \sum_{j \in G_l} J_{l'}|_{l'l} \times (1 - \frac{e^{-j(\beta s_j \cos\theta_{l'} + \phi_{l'})} Z_{jl}}{Z_{ll}}) + Z_{ll} J_l \tag{2}$$

where:

β is the wave number

$\theta_{l'}$ is the angle of incidence upon group l'

$\phi l'$ is the phase at l'

s_j is the distance from the start of the group to the point j

Z_{jl} is the impedance between the point j and the group centre

ϕ is forced to zero, and θ is approximated as 0. This is because the incident fields in a gently undulating terrain profile are grazing (small angles, $\cos\theta$ approximately comes to 1), and that the sum is dominated by near field interactions where θ is very small. K is then approximately constant for all groups, and needs to be evaluated only once, resulting in the below

$$K = 1 - \sum_{j \in G_l} \frac{e^{-j\beta s_j} Z_{jl}}{Z_{ll}}. \tag{3}$$

In other words, the groups only differ in their distance from the transmitter; differences between groups are negligable and it is computationally feasible to ignore them. For the purposes of getting the Field Extrapolation Method running in both the single forward scattering sweep and Forward-Backward methods, K was taken to be 5. K calculation code can be found in Appendix A.5.

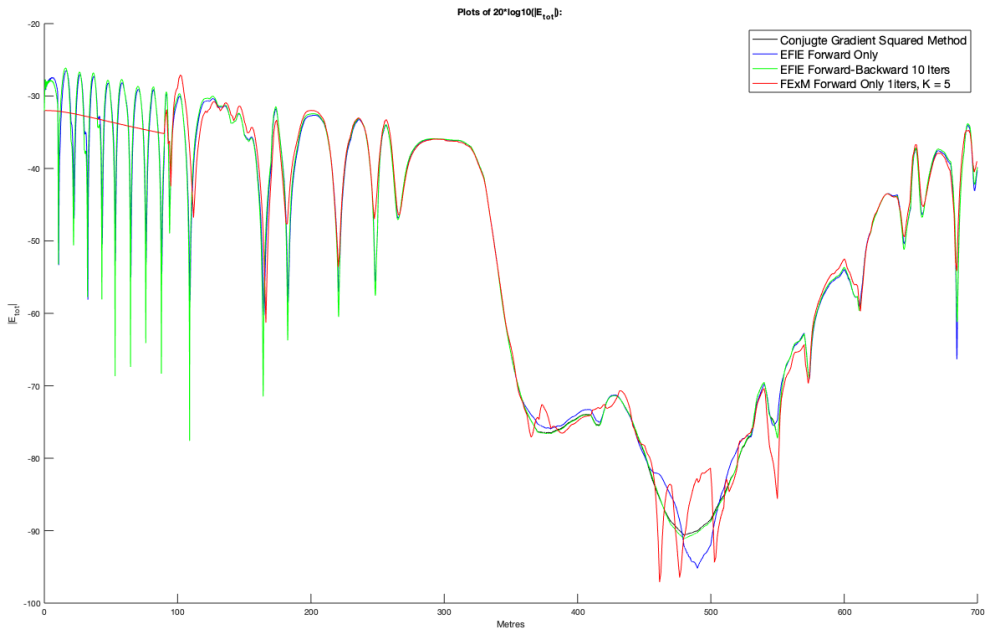


Figure 4.4: Forward Scattering with the Field Extrapolation Method (K=5) and 10 iterations of Forward-Backward with the Electric Field Integral Equation

The single Forward Scattering sweep with the Field Extrapolation Method again proves to be a good approximation for the field above the surface. The graph shows that it is appropriate to consider K a constant value, and although its true value is unlikely to be exactly 5, it is an encouraging start.

4.5 Field Extrapolation Method - Forward-Backward

The Forward-Backward method was now applied to the Field Extrapolation Method. The number of iterations used ranged from 4 to 20, as it didn't appear to converge within 10. The algorithm used was mechanically the same as for the Electric Field Integral Equation, with the forward scattering and back scattering surface currents summed at the end. K is still presumed to be 5. Field Extrapolation Method code can be found in Appendix A.6.

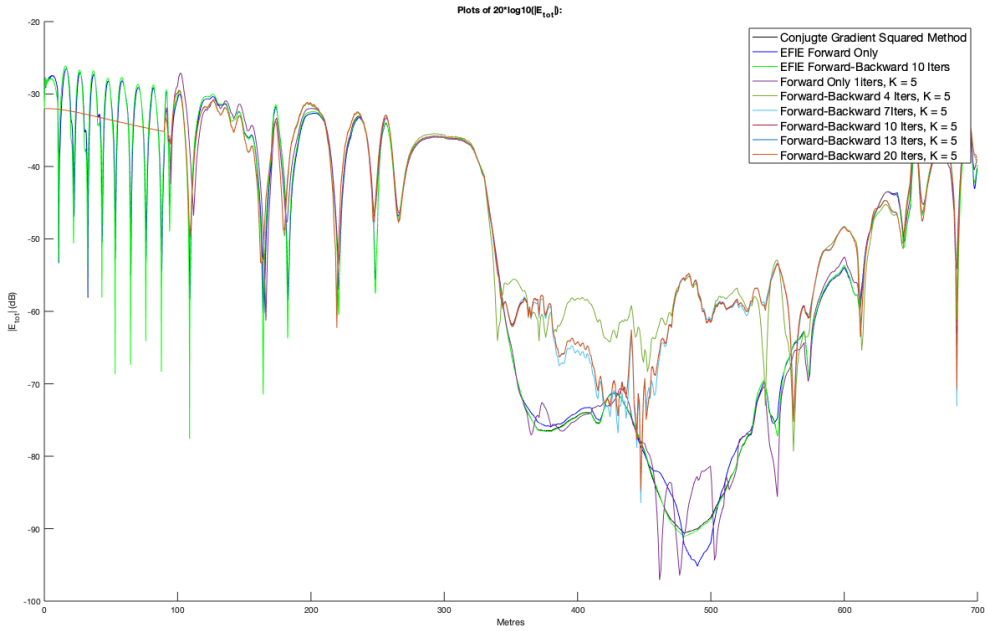


Figure 4.5: Forward-Backward with the Field Extrapolation Method (K=5)

Experiment (K = 5)	Mean Absolute Error(dB)
FExM Forward Scattering	0.3063
FExM Forward-Backward 4 Iterations	5.8104
FExM Forward-Backward 7 Iterations	5.0190
FExM Forward-Backward 10 Iterations	5.1772
FExM Forward-Backward 13 Iterations	5.1602
FExM Forward-Backward 20 Iterations	5.1612

In this case, Forward-Backward results in a loss of accuracy instead of a gain. The trough shows the single Forward Scattering loop result is the only Field Extrapolation Method result that is near the actual value, with the higher iterations of Forward-Backward converging to a higher value. A possible reason for this result is that the true value for K is not 5, as convergence to the true solution is theoretically possible following from the Electric Field Integral Equation experiment.

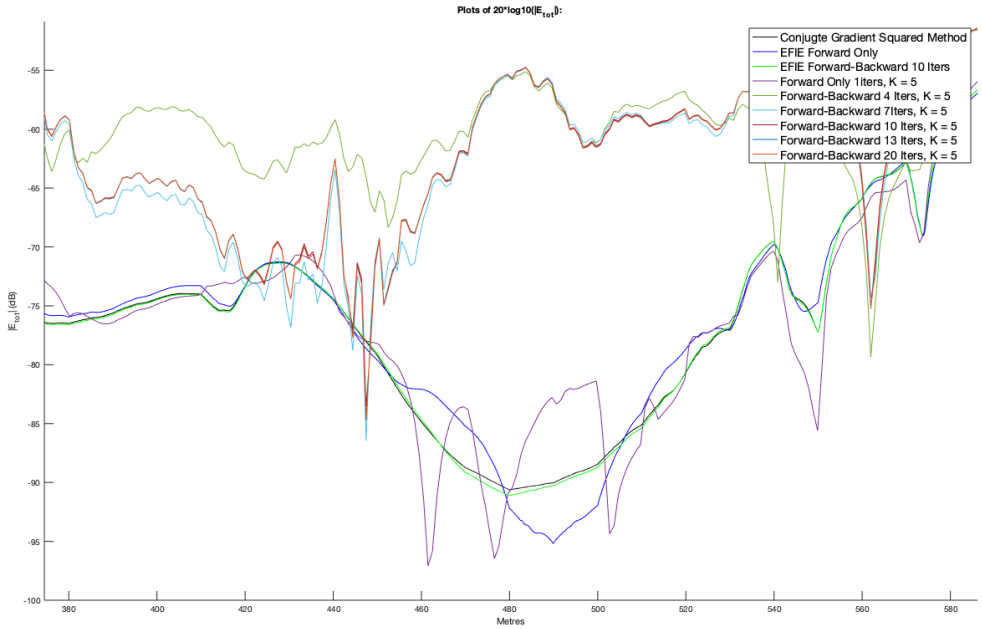


Figure 4.6: Zoomed image of Forward-Backward with the Field Extrapolation Method (K=5)

4.6 Calculating K

This experiment was aimed at calculating the K value of 5 using the formula in the paper[5]. Leaving K as a complex number and taking its modulus were both tested. θ's impact on the value for K and hence the graph overall was also to be evaluated, along with the function applied to θ. Ensuring the formula could be successfully deployed with present materials would open up its testing with other terrain profiles and frequencies, where K may differ due to the different group size and different sampling interval size $\frac{\lambda}{4}$ respectively.

The attempts to follow the formula in the [5] resulted in a K value of 3.5538. Further investigation into the calculation of K was necessary, as it did not return 5, for the current group size and signal frequency. With the aid of a fellow student it became apparent that to achieve a value close to 5, an additional Δs term needed to be added to the formula's denominator, effectively replacing the Z_{ji} calculation with just the Hankel function. This resulted in the value of 5.1052. At the advice of my supervisor, sigma's subtraction from 1 was replaced with subtraction from 0 resulting in a final value of 4.1052.

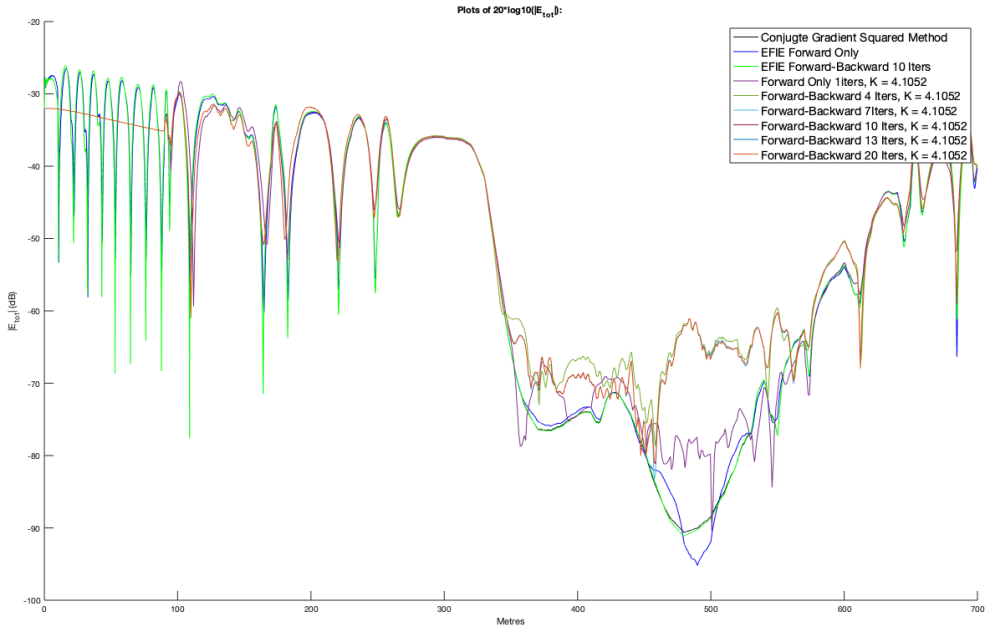


Figure 4.7: Forward-Backward with the Field Extrapolation Method (K=4.1052)

Experiment (K = 4.1052)	Mean Absolute Error(dB)
FExM Forward Scattering	1.2188
FExM Forward-Backward 4 Iterations	3.6877
FExM Forward-Backward 7 Iterations	3.3646
FExM Forward-Backward 10 Iterations	3.3787
FExM Forward-Backward 13 Iterations	3.3786
FExM Forward-Backward 20 Iterations	3.3787

With the new value for K, the Field Extrapolation Method performs better overall. The single pass of Forward Scattering approximates the trough less accurately, but the Forward-Backward graphs all approximate it better than before. As before with the value of 5, their approximation does not improve with increased iterations, suggesting that although they do not converge to an exact solution, they do so extremely rapidly. The Field Extrapolation Method with this new K also approximates the first dip in the trough much better than it did before, suggesting that there may be a value for K which allows the method to return highly accurate results quicker than the exact and the Electric Field Integral Equation methods.

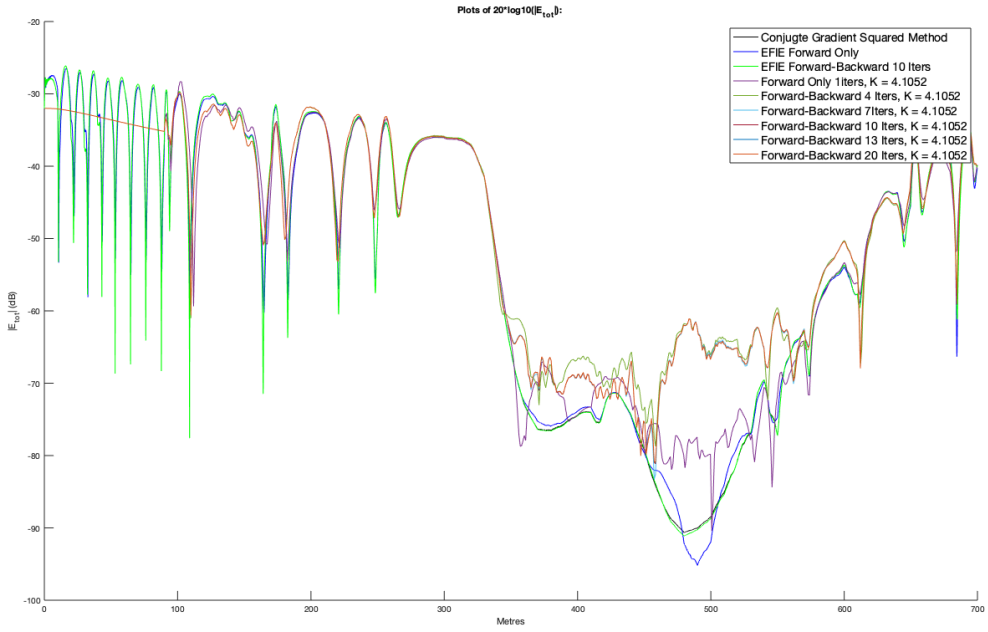


Figure 4.8: Forward-Backward with the Field Extrapolation Method, K left as complex number

Experiment (K = 4.0884 + 0.37057i)	Mean Absolute Error(dB)
FExM Forward Scattering	1.4875
FExM Forward-Backward 4 Iterations	4.0646
FExM Forward-Backward 7 Iterations	3.8876
FExM Forward-Backward 10 Iterations	3.8885
FExM Forward-Backward 13 Iterations	3.8876
FExM Forward-Backward 20 Iterations	3.8874

Including θ required a new value for K for every unique combination of group centres, as θ depended on the angle of incidence that the ray from the scattering group makes with the normal vector of the receiving group. There also needed to be values for back scattering, as the angle of incidence of the ray from the first centre to the second centre is not the same as from the second centre to the first centre. This resulted in a matrix similar in structure to the Z matrix from the exact Conjugate Gradient Squared method, but 13^2 times smaller.

First, the target centre's normal was calculated by taking the angle α the group makes with the horizontal, adding 90 degrees to it for a positive rotation (to ensure the normal points "up" to the sky and not "down" into the earth) gaining the angle β. The normal vector was then $[\cos\beta, \sin\beta]$, whose modulus is 1. Finally the angle of incidence was found by taking the dot product between the vector made by subtracting the target centre from the source centre, and the target centre's normal vector. For back scattering, the angle α is calculated for the other plate, and the centres are swapped in the function call. The code for θ's inclusion is listed in Appendix A.7.

The experiment of θ's impact was tested with $\sin\theta$ as well as $\cos\theta$ as in the original formulation. The reason for trying $\sin\theta$ is that θ was originally presumed to be the angle between plates, but the incident ray forms the angle θ with the receiving group's surface normal which is at +90 degrees to the group itself.

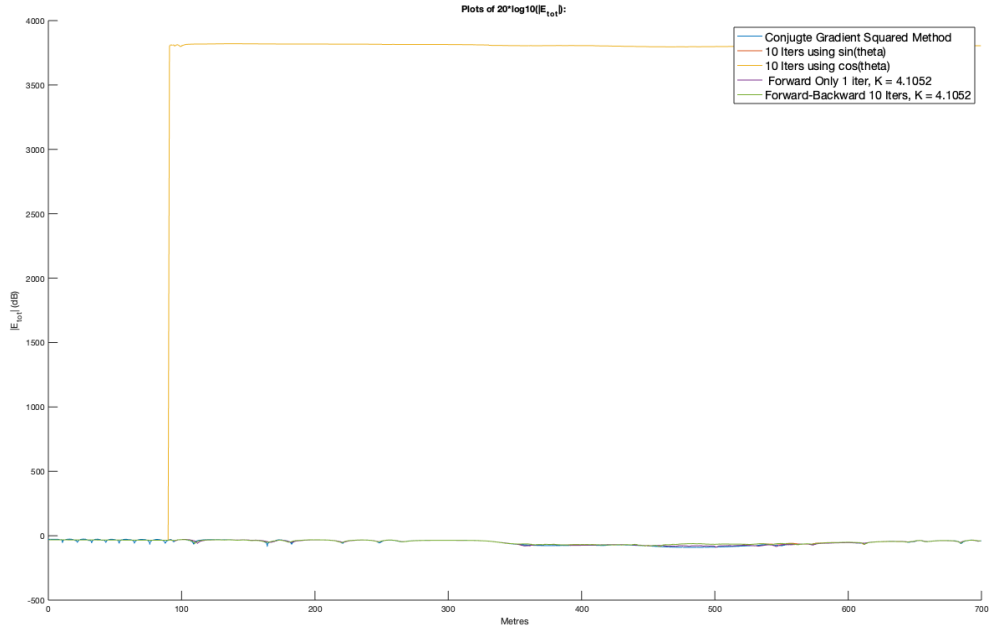


Figure 4.9: Forward-Backward with the Field Extrapolation Method, accounting for θ

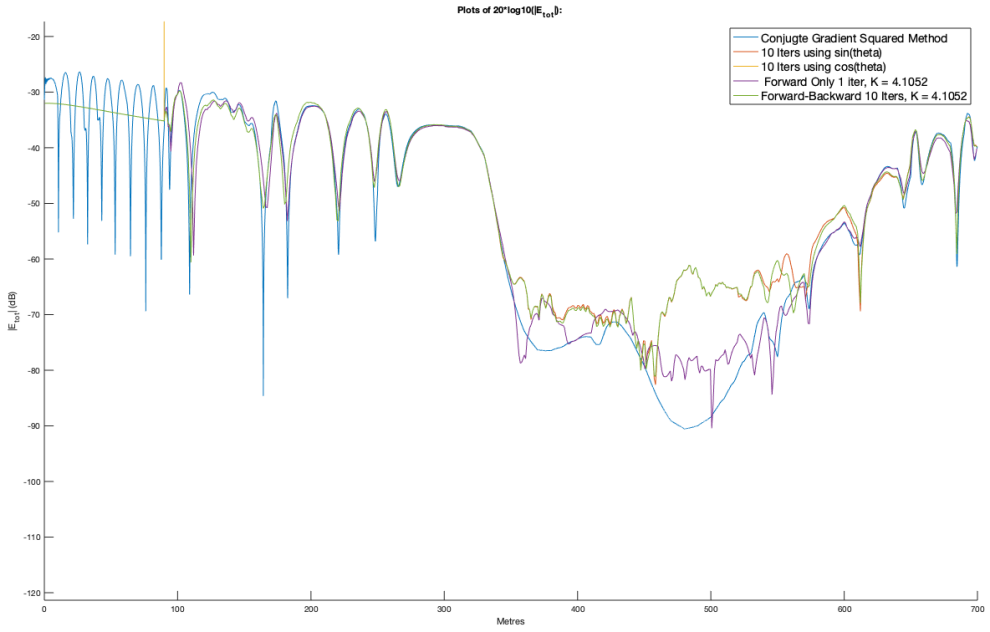


Figure 4.10: Zoomed in image of Forward-Backward with the Field Extrapolation Method, accounting for θ

Experiment K = 4.1052 Base, accounting for θ	Mean Absolute Error(dB)
FExM Forward Scattering	1.2188
FExM Forward-Backward 10 Iterations	3.3787
FExM Forward-Backward 10 Iterations $\sin\theta$	3.4580
FExM Forward-Backward 10 Iterations $\cos\theta$	3.3636×10^3

The first graph confirms the suspicions about the cosine function. When $\cos\theta$ is used for the K values, it diverges massively from the actual solution. $\sin\theta$ does better, although it does perform worse than a single forward scattering loop, 4 iterations of Forward-Backward, and all Electric Field Integral Equation results. In this scenario with gently undulating rural terrain, it is not surprising that accounting for the angles does not aid in convergence, but that it diverges more from the single Forward Scattering result than the Forward-Backward with constant K result is a surprise.

Forward Scattering FExM continues to be a decent approximation for gently undulating rural environments, while including back scattering and incidence angles hinders rather than helps, possibly because they do not play a strong role in the real-world version of this profile and too much weight is being attached to these parameters. The different K value causes an improvement in accuracy, suggesting there may be a way to calculate K in order to achieve excellent accuracy with low compute time.

4.7 Urban Profile

The urban profile was then fed through the same battery of tests:

- Conjugate Gradient Squared method
- Electric Field Integral Equation - Forward Scattering and Forward-Backward (10 iterations)
- Field Extrapolation Method - Forward Scattering and Forward-Backward (10 iterations)
- Field Extrapolation Method - Accounting for θ

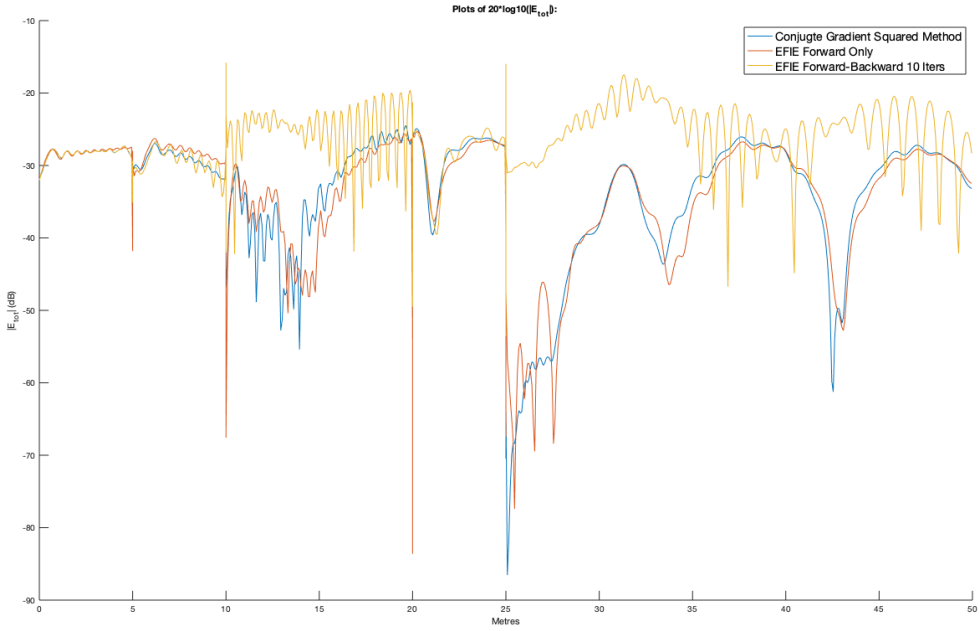


Figure 4.11: Electric Field Integral Equation in the Urban Terrain Profile

Experiment Urban EFIE	Mean Absolute Error(dB)
EFIE Forward Scattering	3.8452
EFIE Forward-Backward 10 Iterations	43.8480

Using the Forward-Backward method in the urban profile results in a divergence from the actual solution. The single Forward Scattering result in this case is a much better match for the exact Conjugate Gradient Squared method for the horizontal parts of the profile, but their degree of agreement along the walls is unclear from the graphs (the walls are at 5, 10, 15 and 20 metres along X).

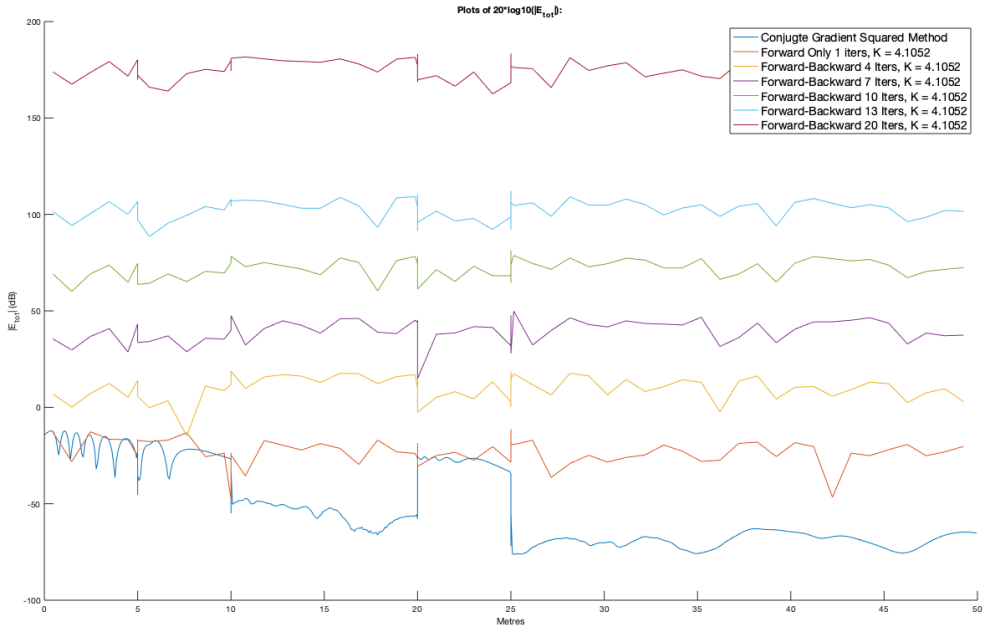


Figure 4.12: Field Extrapolation Method in the Urban Environment

Experiment K = 4.1052	Mean Absolute Error(dB)
FExM Forward Scattering	25.3653
FExM Forward-Backward 4 Iterations	58.6645
FExM Forward-Backward 7 Iterations	87.9287
FExM Forward-Backward 10 Iterations	120.5459
FExM Forward-Backward 13 Iterations	151.2804
FExM Forward-Backward 20 Iterations	223.6432

Divergence occurs with Forward-Backward in the Field Extrapolation Method, and the single forward scattering loop is much less accurate than it was with the rural profile. None of the graphs follow the exact method's shape.

θ's inclusion was then examined, as it may now play a more significant role with the sheer vertical walls. As no significant improvements in accuracy accrued from increasing iterations beyond 10, 10 iterations of Forward-Backward were used for the θ experiments. As with the rural profile, including θ greatly increased computation time.

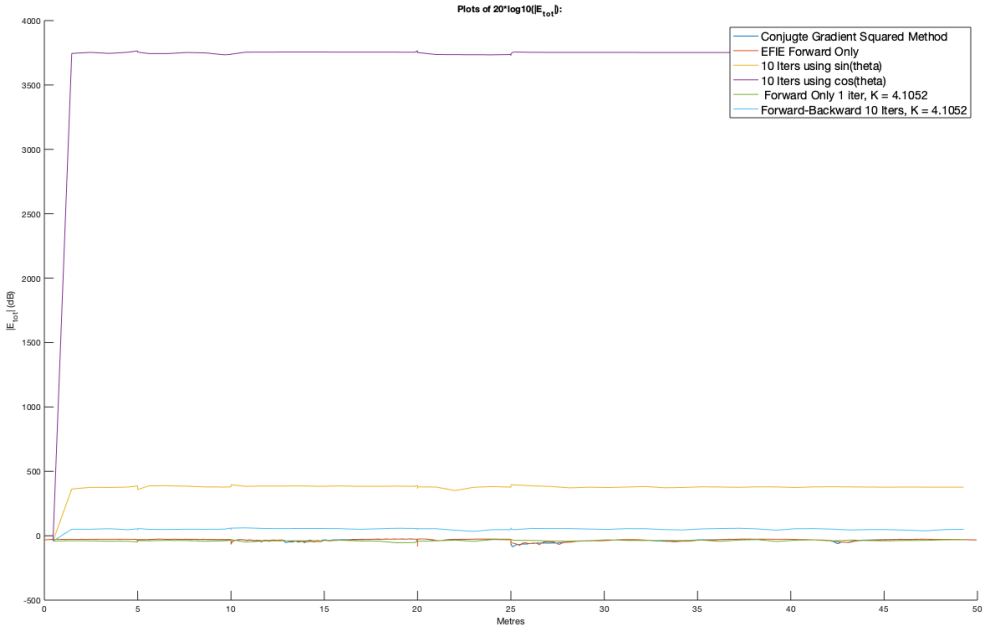


Figure 4.13: Forward-Backward with the Field Extrapolation Method, accounting for θ

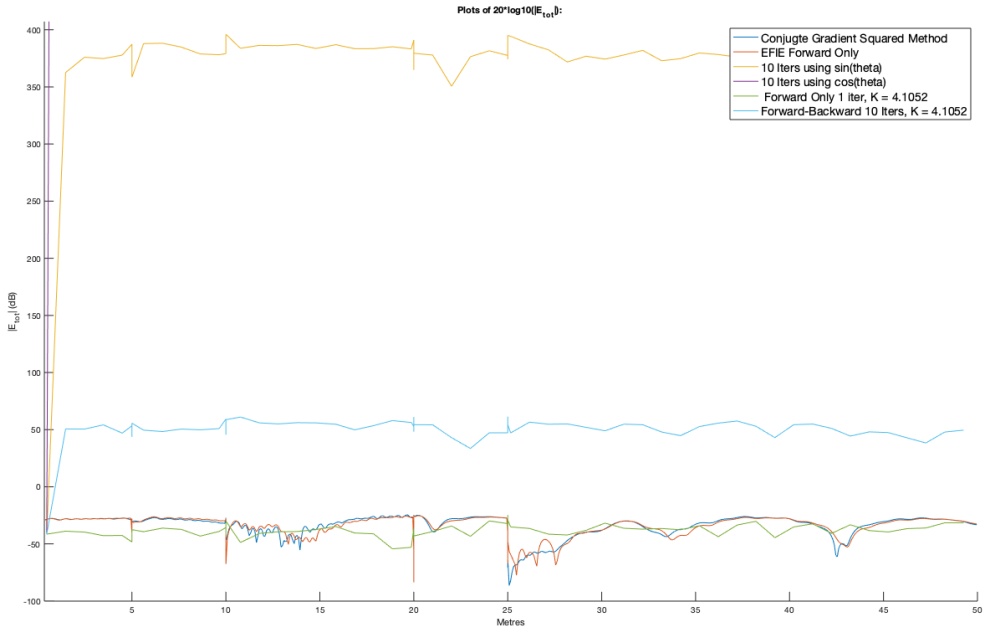


Figure 4.14: Zoomed in image of Forward-Backward with the Field Extrapolation Method, accounting for θ

Experiment K = 4.1052 Base, accounting for θ	Mean Absolute Error(dB)
FExM Forward Scattering	20.9366
FExM Forward-Backward 10 Iterations	116.0372
FExM Forward-Backward 10 Iterations $\sin\theta$	474.7822
FExM Forward-Backward 10 Iterations $\cos\theta$	3.7764×10^3

Including the angle θ for the urban profile does nothing to improve accuracy, being just as divergent as it was in the rural terrain profile. None of the methods tested show convergence in the urban profile. It is likely that the Ground Truth is inaccurate to begin with, as it is the exact solution for an equation that is unsuitable for use outside of gentle rural terrain.

4.8 Re-running Rural Profile using Stricter Terrain Interpolation

The results from running the previous Field Extrapolation Method experiments were inconsistent with those from the Electric Field Integral Equation. Forward-Backward was showing convergent behaviour but not towards the Ground Truth. This opened the question about the sensitivity of the method to errors, and so the Field Extrapolation Method was re-examined with the rural profile, but using the stricter urban terrain profile interpolation method to remove this source of error.

Each value of K was tested again; 5, 3.5538, 4.1052 and 4.0884 + 0.37057i. Re-trying 3.5538 was prudent as it was found with a strict interpretation of the K formula. The Electric Field Integral Equation was also re-tested to make sure it was not greatly affected by the previous approximate interpolation method.

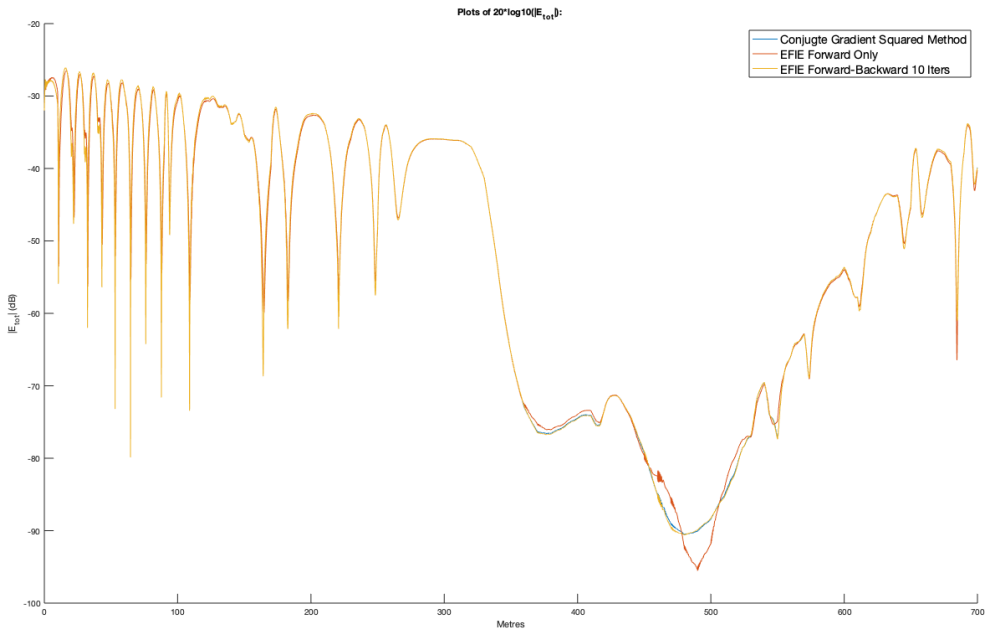


Figure 4.15: Electric Field Integral Equation Re-Run with stricter Terrain Profile

Experiment	Mean Absolute Error(dB)
EFIE Forward Scattering	0.0243
EFIE Forward-Backward 10 Iterations	0.0151

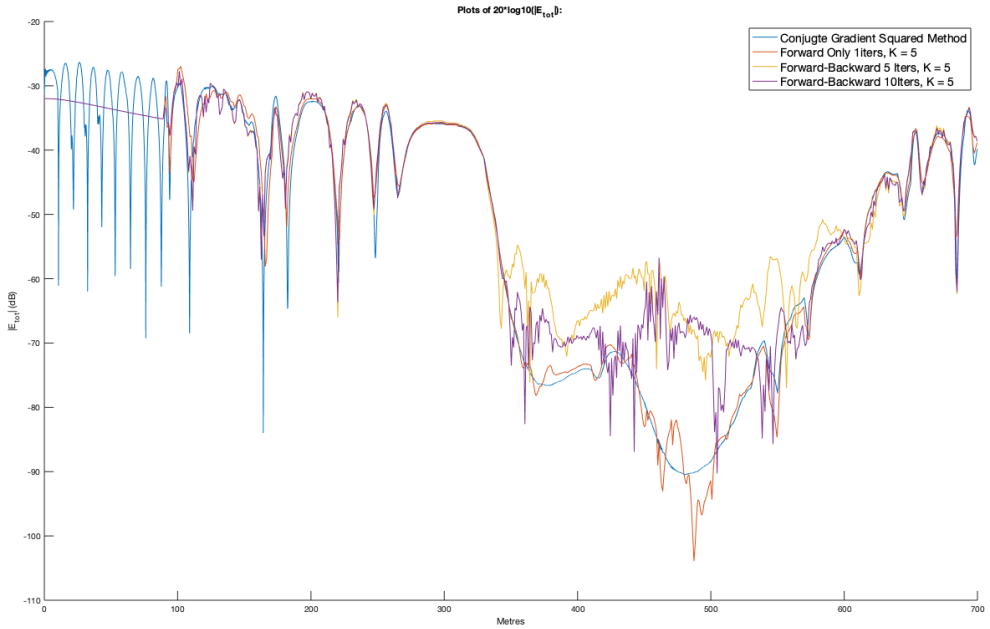


Figure 4.16: Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 5

Experiment K = 5	Mean Absolute Error(dB)
FExM Forward Scattering	0.1492
FExM Forward-Backward 5 Iterations	4.7264
FExM Forward-Backward 10 Iterations	2.7644

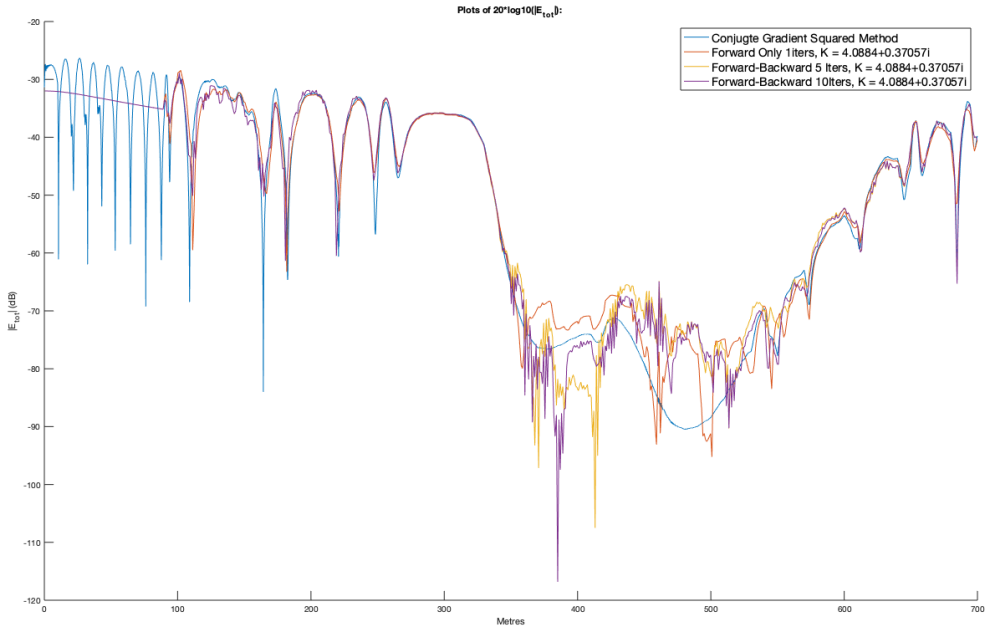


Figure 4.17: Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 4.0884 + 0.37057i

Experiment K = 4.0884 + 0.37057i	Mean Absolute Error(dB)
FExM Forward Scattering	1.1978
FExM Forward-Backward 5 Iterations	1.2211
FExM Forward-Backward 10 Iterations	0.8443

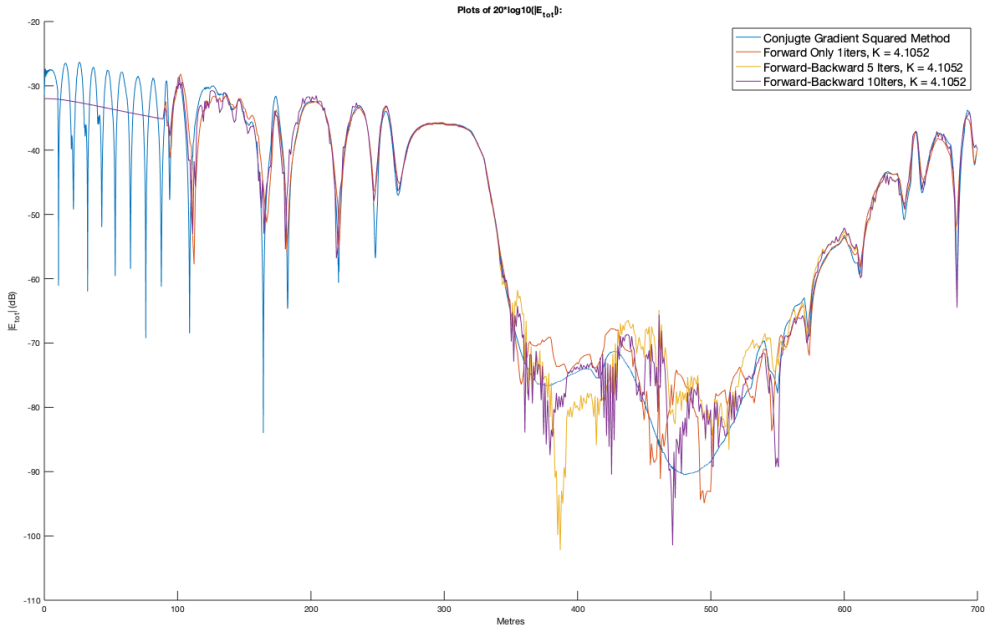


Figure 4.18: Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 4.1052

Experiment K = 4.1052	Mean Absolute Error(dB)
FExM Forward Scattering	0.9300
FExM Forward-Backward 5 Iterations	1.0658
FExM Forward-Backward 10 Iterations	0.5509

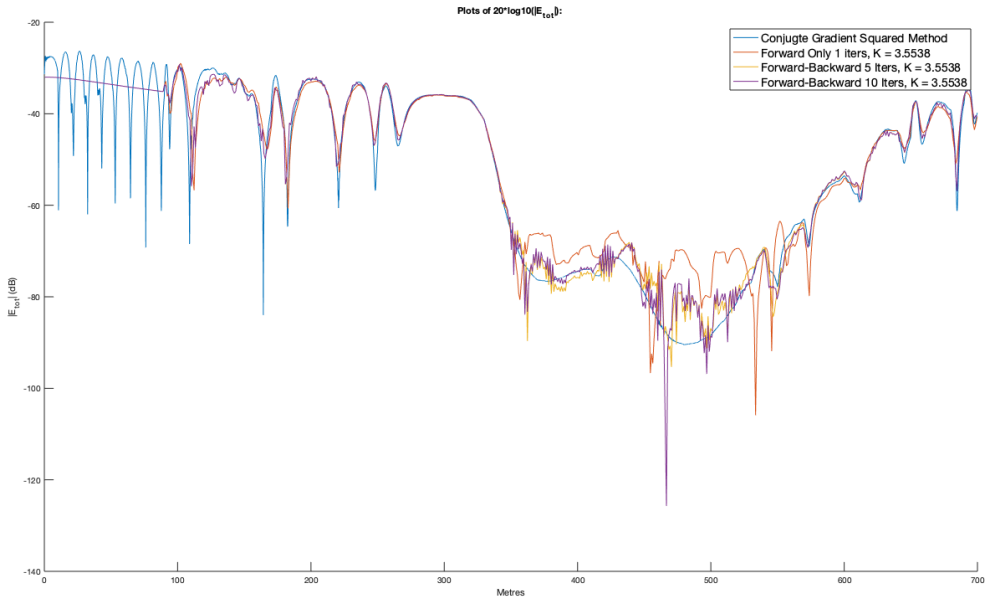


Figure 4.19: Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 3.5508

Experiment K = 3.5508	Mean Absolute Error(dB)
FExM Forward Scattering	1.7743
FExM Forward-Backward 5 Iterations	0.5581
FExM Forward-Backward 10 Iterations	0.4953

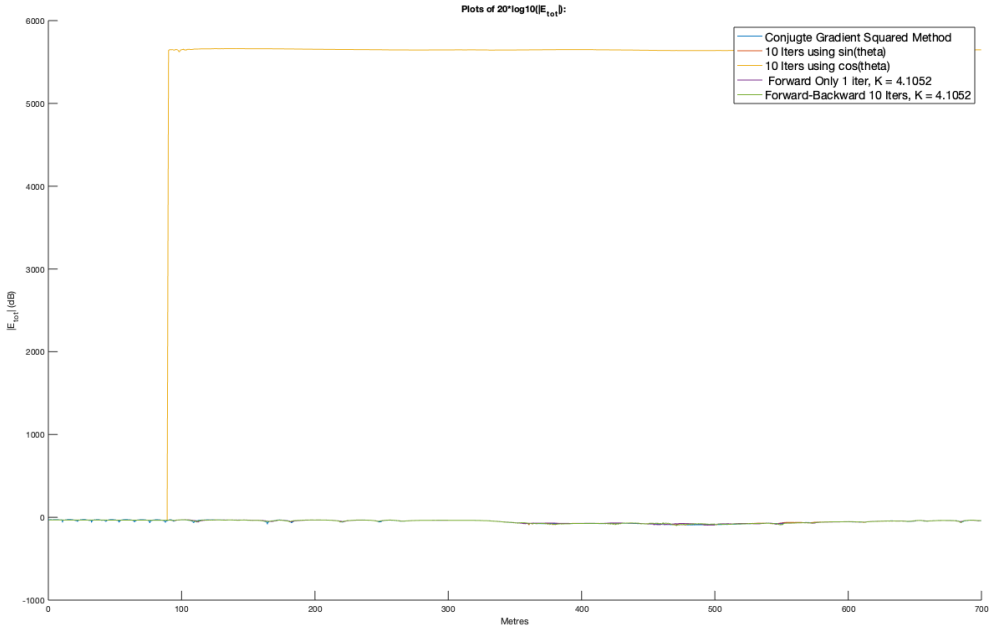


Figure 4.20: Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 4.1052, accounting for θ

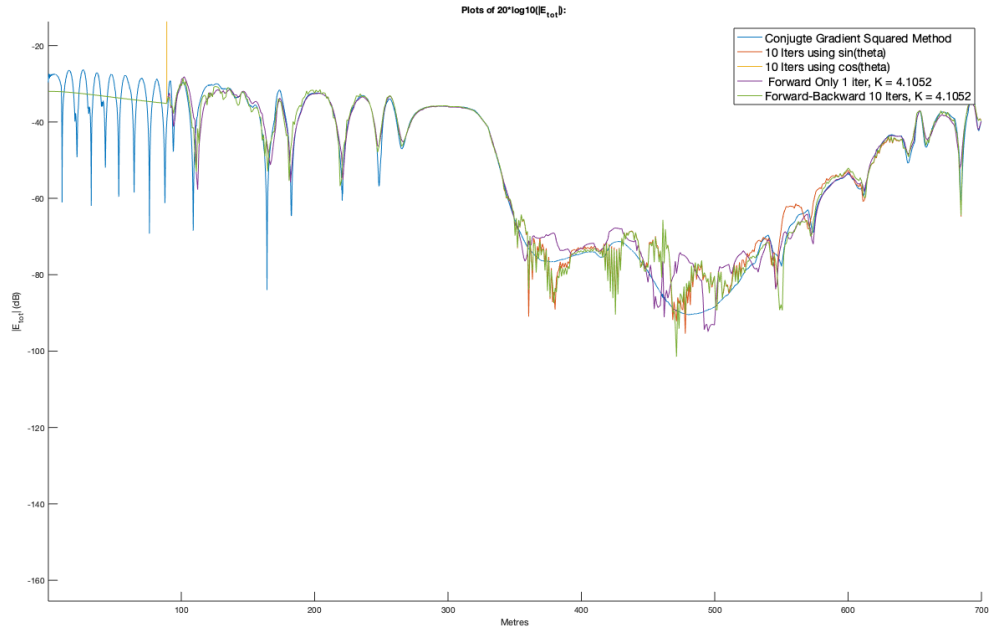


Figure 4.21: Zoomed in Field Extrapolation Method Re-Run with stricter Terrain Profile, K = 4.1052, accounting for θ

Experiment K = 4.1052, accounting for θ	Mean Absolute Error(dB)
FExM Forward Scattering	0.9300
FExM Forward-Backward 10 Iterations	0.5509
FExM Forward-Backward 10 Iterations $\cos\theta$	4.9866×10^3
FExM Forward-Backward 10 Iterations $\sin\theta$	0.9320

In the case of the Electric Field Integral Equation, the results were the same; forward scattering is a good approximation for the gently undulating profile and Forward-Backward improves the accuracy further.

Major changes do occur in the adoption of the new interpolation method for the Field Extrapolation Method. Its sensitivity to error is apparent, while the Electric Field Integral Equation is more robust. This property is demonstrated in the difference between the now good approximation and the one before it. The forward scattering approach is as good as it was, but the Forward-Backward method now converges to a more accurate solution than forward scattering.

Interestingly, forward scattering accuracy increases with increasing K values, while Forward-Backward converges towards a more correct solution, more rapidly, with smaller K values. Additionally, the 5 iteration loop provides an accuracy gain over forward scattering with small K, and an accuracy loss with large K, but the most accurate Forward-Backward solution does not outperform the most accurate forward scattering solution.

The inclusion of θ does not make the method more accurate. $\cos\theta$ is again three orders of magnitude off, but $\sin\theta$ is 0.002 dB off the forward scattering loop. This is a good result on its own, however considering the vastly greater run time and memory requirements involved, it cannot be taken as a net positive. It is also outperformed by the Forward-Backward method with a constant K value and the same number of iterations, rendering the result topically interesting but ultimately inconsequential.

5 Discussion and Further Work

Overall, the single forward scattering Field Extrapolation Method works well for Rural environments, relative to the exact Conjugate Gradient Squared method. On balance, the accuracy lost vs single forward scattering Field Extrapolation Method and Forward-Backward Electric Field Integral Equation is made up for in the speed of calculation, and is suitable where close approximations are appropriate. In the Field Extrapolation Method, while Forward-Backward does not reach the same accuracy as forward scattering, it may be accurate enough that the group size can be enlarged for greater calculation speed.

A bizarre effect observed is that of the change in accuracy of forward scattering and Forward-Backward with respect to K. When K is 5, the forward scattering pass yields a very accurate result on its own, and the Forward-Backward method does not converge. When K is 3.5538, forward scattering is very inaccurate, and Forward-Backward converges to its most accurate solution, without the characteristic accuracy loss at 5 iterations that is suffered in other runs and for 4 iterations in the Electric Field Integral Equation. The author observed that forward scattering did not become more accurate for K larger than 5 (5.1052 when $1 - \Sigma$ was still a part of the K calculation), nor did Forward-Backward with K = 3.0 (arbitrary value less than 3.5538). It would seem a large K can better approximate forward scattering, while a small K does not, but better approximates back scattering, likely because the large K multiplies the back scattering effects beyond their actual contribution.

A more comprehensive idea of the efficacy of the algorithms could be found by using multiple terrain profiles with their measured results available to compare to. This, however, was not possible due to the untimely COVID-19 pandemic and subsequent lockdown. These data were not accessible remotely, and so could not be used for more complete verification. An actual urban terrain profile would have been invaluable but as it stood, a mock urban profile had to be created, and the Conjugate Gradient Squared method trusted completely. Different signals could also have been used, as in [3], but 970 MHz was kept to due to the time used in the work done on K.

The reduction in memory requirements is encouraging for Cognitive Radio. A Cognitive Radio device is likely to have memory precluding the affordability of storing an impedance matrix. 9,060 x 9,060 elements is the size of the matrix for a 700 metre two-dimensional profile with a

0.0773 metre sampling interval, nearly on the order of 10^{10} complex floats. In the case of considering Rayleigh scattering effects by the addition of a Z axis (for side-scattering contributions which an urban environment is certain to have), enlarging the area, and shortening the wavelength will all be means of increasing the size of this matrix quickly beyond a limited device's capacity.

This can theoretically be alleviated by paging, using storage to hold entries not immediately to be used, but this exacerbates a secondary problem that holding it in memory brings. The access time of memory is extremely long, with the cost of major page faults being longer still. Computing these entries on-the-fly offers a reduction in memory capacity requirements. Further, where capacity may not be a concern, it removes the mental burden of arranging the data structure optimally for avoiding costly cache misses.

The Field Extrapolation Method will need more work done regarding the calculation of K. Its exact formulation needs to be developed, with the roles played by the angle of incidence and the phase investigated, along with the vanished δs term and $1 - \Sigma$. It's clear that K does provide an accuracy increase for both forward scattering and Forward-Backward implementations of the Field Extrapolation Method; calculating it correctly and gaining the best possible accuracy with the very low computation time is something The Author could not complete.

A means of determining the size of the group for the Field Extrapolation Method based on accuracy vs speed should be developed, as 13 segments will not fit all profiles. [5] states with additional accuracy an already large 10 metre group can be enlarged for extra speed (keeping in mind a different profile used within); accurately determining the degree of enlargement would be beneficial.

6 Conclusions

The Field Extrapolation Method takes the savings in compute time made by the Electric Field Integral Equation further by calculating K once at the beginning of the algorithm, allowing for points to be grouped and their collective contributions to be calculated as a function of the incident field and the factor K multiplied by scattering contributions from neighbouring groups where applicable. The increase in speed and decrease in memory consumption compared to the Electric Field Integral Equation method is proportional to the group size eg. 13 segments per group $\rightarrow$ one 13^{th} the memory required for J and E_{tot} vectors and one 13^{th} the number of iterations per forward and back scattering loop.

The accuracy of the single pass of forward scattering both in the Electric Field Integral Equation method and the Field Extrapolation Method and small improvement towards the Conjugate Gradient Squared method in the case of the former under Forward-Backward demonstrates that the single pass of forward scattering is a very strong approximation of the field above the surface for gently undulating rural terrain.

The Field Extrapolation Method is very sensitive to error, highlighted in applying the Forward-Backward method. While the condition number for the impedance matrix in the Electric Field Integral Equation is only 59.7445, and for the same (though much smaller) matrix in the Field Extrapolation Method it is 4.5652, the difference made by a seemingly insignificant change in the terrain profile's interpolation determined the Forward -Backward method's convergence. The condition number of the impedance matrix does not tell the full story of the problem's ill-conditioned nature.

The Field Extrapolation Method requires a short smooth region at the beginning of the terrain profile when applying the Forward-Backward method, along with an accurate value for K and a strict interpolation of the points. Without the combination of all of these, the Forward-Backward method does not converge to an accurate solution for the Field Extrapolation Method, but will converge to an inaccurate solution. The Electric Field Integral Equation is more robust to inaccurate terrain interpolations and does not require a smooth region, as the greater volume of points to consider scattering effects to and from overpowers the errors which the Field Extrapolation Method is vulnerable to.

Forward-Backward applied to the Field Extrapolation Method did not converge to the most accurate solution. The single forward scattering run of the Field Extrapolation Method with K = 5 yielded the result with the smallest mean absolute error, with the most accurate 10 iterations of Forward -Backward with K = 3.5538 having an error 3.5x greater. It's an acceptable result as it's still half a dB off the Conjugate Gradient Squared method, but is less accurate than forward scattering, and a great deal slower.

Including the angle of incidence the scattering plate's ray makes with the receiving plate does nothing to improve accuracy in any case, amplifying them in most cases. It greatly increases the computation time and memory requirements, as an NxN matrix needs to be calculated and stored in memory. Its best performance was equalling the single forward scattering Field Extrapolation Method run with an enormous increase in computational complexity and memory storage.

As expected following reading of the [3] and [7] papers, the Forward-Backward method with both the Electric Field Integral Equation and Field Extrapolation Method fared poorly in the urban profile, with only two buildings and two dimensions. The urban profile is one where back scattering effects are very strong, as the buildings make 90 degree angles with the ground. There are interactions between building and surface, as well as within building, that the Forward-Backward method doesn't account for. [7] states a generalisation can be made to the impedance matrix, but in an urban setting with many buildings and other large, arbitrarily shaped and possibly moving obstacles it is not suitable to consider.

Bibliography

[1] C. Cordeiro et al. “IEEE 802.22: the first worldwide wireless standard based on cognitive radios”. In: *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.* 2005, pp. 328–337.

[2] Bruce A. Fette. *Cognitive Radio Technology.* 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA: Newnes, 2006.

[3] J. T. Hviid et al. “Terrain-based propagation model for rural area-an integral equation approach”. In: *IEEE Transactions on Antennas and Propagation* 43.1 (1995), pp. 41–46.

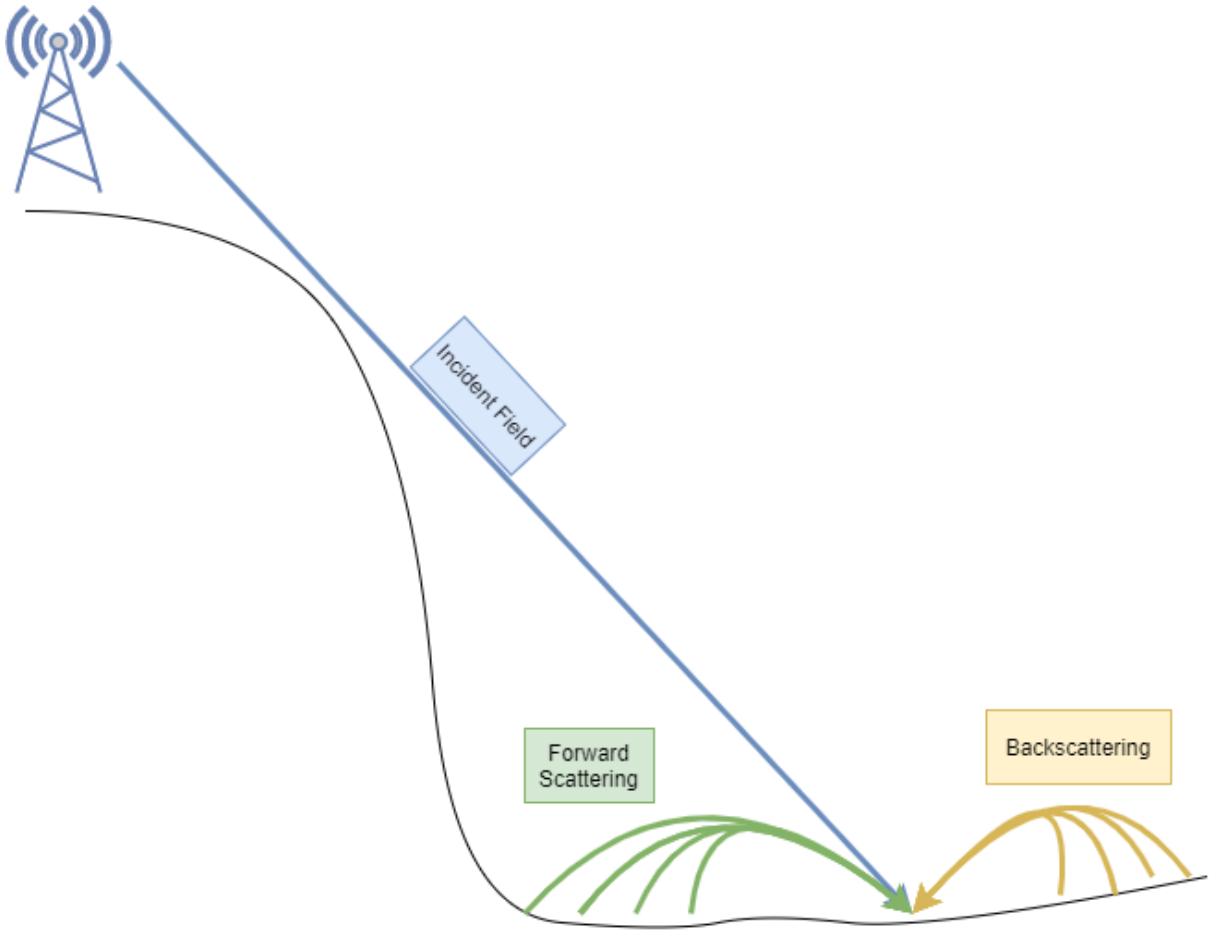
[4] I. Kakalou et al. “Radio Environment Maps for 5G Cognitive Radio Network”. In: *2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST).* 2019, pp. 1–4.

[5] E. O. Nuallain. “An efficient Integral equation-based electromagnetic propagation model for terrain”. In: *IEEE Transactions on Antennas and Propagation* 53.5 (2005), pp. 1836–1841.

[6] E. O Nuallain. “A Proposed Propagation-Based Methodology with Which to Address the Hidden Node Problem and Security/Reliability Issues in Cognitive Radio”. In: *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing.* 2008, pp. 1–5.

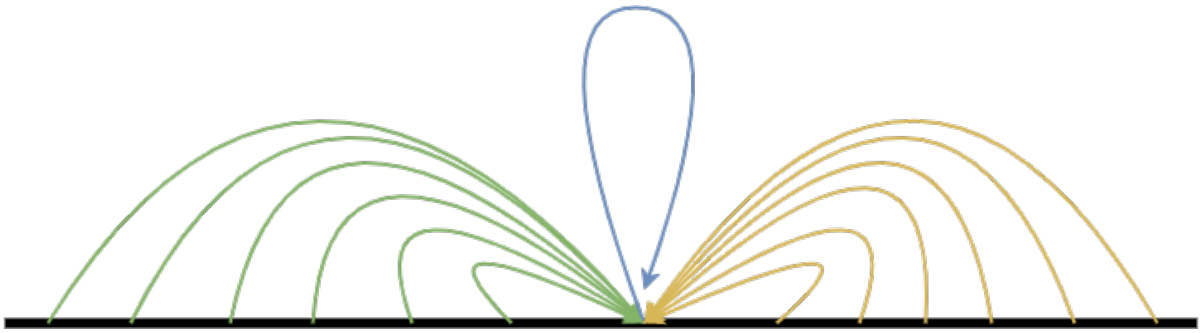
[7] M. Rodriguez Pino et al. “The generalized forward-backward method for analyzing the scattering from targets on ocean-like rough surfaces”. In: *IEEE Transactions on Antennas and Propagation* 47.6 (1999), pp. 961–969.

A1 Problem Geometry



Visualisation of the problem geometry. The surface point is irradiated by an incident field, forward scattering contributions from points between it and the transmitter, and back scattering contributions from points beyond. Side-scattering contributions are ignored; as such, there is no Z axis to consider.

A2 K Visualisation



Visualisation of the calculation of K. K represents the aggregation of an entire group's scattering contributions and is approximately constant for all groups, allowing us to forego their direct calculation and to consider scattering from the centre.

A3 Terrain Interpolation

```
function TP = import_terrain_profile(d_s, max_x, N)
   % function to read the terrain profile file with points every 10m along
   % x, and interpolate the missing points at lambda/4 metres along the x
   % axis. Rather than try to fit X samples within each 10 metres
   % (lambda/4 doesn't divide 10 evenly), it creates a series of
   % (i-1)*lambda/4, where i is the index in the series beginning at 1
   % because MATLAB. For the y points, having seeded the first y value
   % with the first given in the terrain profile, it uses the slope
   % between the current point and the next point, multiplies that slope
   % by lambda/4, and then adds that to the current y in order to find
   % the next y (slope will be either positive or negative).

   terrain_file = fopen("X.04");
   data = textscan(terrain_file, "%s");
   fclose(terrain_file);

   % rtp = rough terrain profile
   rtp = [str2double(data{1}(1:2:end)), str2double(data{1}(2:2:end))];
   clear data
   clear terrain_file

   % frtp = filtered rough terrain profile (discard entries beyond max_x - for
   % testing purposes only, remove when processing full profile)
   frtp = rtp(rtp ~= inf & rtp(:,1) <= max_x);

   % reshaped frtp to array of N (x,y) points
   xy_rtp = reshape(frtp, length(frtp) / 2, 2);

   % Every X value is an integer multiple of d_s (lambda / 4)
   % go as far as N-1, falls just short of the maximum X value but yields
   % an array of points the same length as the arrays J, E_tot
   TP = [d_s*(0:N-1); zeros(1,N)]';

   % seed the first Y value
   TP(1,2) = xy_rtp(1,2);

   % instead of finding angle theta with tan^-1 and then performing
   % x*tan(theta) to get y, can instead multiply x by the slope
   % y in long form is x*tan(tan^-1(dy/dx)) which is equivalent to
   % x * (dy/dx), trig functions being redundant
   slopes = get_slopes(xy_rtp);
```

```
42
43      % iterating from 2 to the end instead of 1 to the end can be thought of
44      % as populating a list of "next y" values
45      for i = 2:N
46          % x increments in steps of 10, get index for slope array by
47          % dividing the current x by 10 and rounding up - the rounding up is
48          % due to MATLAB array indices starting at 1
49
50          ind = ceil(TP(i,1) / 10);
51
52          % we will inevitably run to the end of the terrain profile, where
53          % there is no possible slope beyond the final point but the array
54          % size *must* be consistent with those of E and J - continue with
55          % previous slope values for now, and later determine what to do
56          % with the terminal point
57
58          if (ind > length(slopes))
59              ind = ind - 1;
60          end
61          m = slopes(ind);
62  TP(i,2) = TP(i-1,2) + d_s * m;
63      end
64  end
65
66  function slopes = get_slopes(rtp)
67      rtp_y1s = rtp(1:length(rtp)-1,2);
68      rtp_y2s = rtp(2:length(rtp), 2);
69      slopes = (rtp_y2s - rtp_y1s) / 10; % m = y2-y1 / x2-x1, always 10 in this
        case
70  end
71
72  function d = dist(x_1, y_1, x_2, y_2)
73      d = sqrt((x_2 - x_1)^2 + (y_2 - y_1)^2);
74  end
```

This code block is the erroneous terrain profile interpolation code. It is included as despite the flaws, it did not hinder the Electric Field Integral Equation's generation of an acceptable output. Only the Field Extrapolation Method suffered accuracy losses due to the interpolation.

```
% this script has no clearvars function call, as it is called by another script instead of being run on its
% own, and so should have access to the caller's globals
EFIE_group_points = import_terrain_profile();

function d = dist(pt1, pt2)
% changed prototype to points instead of point ordinates as it reads easier when called
   d = sqrt((pt2(1) - pt1(1))^2 + (pt2(2) - pt1(2))^2);
end

function group_points = import_terrain_profile()

   global N_for_TP d_s max_x

   % terrain_file = fopen("X.04");
   terrain_file = fopen("andrew_urban_simple.txt");
   data = textscan(terrain_file, "%s");
   fclose(terrain_file);

   % rtp = rough terrain profile
   rtp = [str2double(data{1}(1:2:end)), str2double(data{1}(2:2:end))];
   clear data terrain_file

   % frtp = filtered rough terrain profile (discard entries beyond max_x - for
   % testing purposes only, remove when processing full profile)
   frtp = rtp(rtp ~= inf & rtp(:,1) <= max_x);

   % reshaped frtp to array of n_grps (x,y) points
   xy_rtp = reshape(frtp, length(frtp) / 2, 2);
   TP = [zeros(1,N_for_TP); zeros(1,N_for_TP)]';

   % seed the first Y value
   TP(1,2) = xy_rtp(1,2);
   j = 2;
   p1 = TP(1,:);

   for i = 1 : length(xy_rtp) - 1
       p2 = xy_rtp(i+1,:);        % "next" point
       p2_vec = p2(:) - p1(:);    % vector to next point
       p2_vec_len = dist(p1, p2);

       m = p2_vec(2) / p2_vec(1); % slope of the vector
       alpha = atand(m);
       num_points = floor(p2_vec_len / d_s); % total points to add this iteration (how many d_s
% segments fit between current point and next point)

       points_xs = p1(1) + [1:num_points] * d_s * cosd(alpha); % calculate these points
       points_ys = p1(2) + [1:num_points] * d_s * sind(alpha);
       points = [points_xs ; points_ys]'; % assign to vector
```

```
49
50          TP(j:j+num_points-1, :) = points; % overwrite blank at position j and
        advance index j
51          j = j + num_points;
52          p1 = p2; % next point
53      end
54
55      net_points = [1; find(TP(:,1) > 0)]; % ignore the zeros beyond the last
        point for neatness
56      group_points = TP(net_points,:);
57  end
```

The more accurate interpolation code, which when combined with an appropriate K value allowed the forward-backward method to converge with the Field Extrapolation Method (3.5538 worked well for K).

A4 Electric Field Integral Equation: Forward-Backward

J - Surface Current

```
J_BS = zeros(N, 1);
J_FS = zeros(N,1);


for loop_iter = 1 : iters
   for p = 1 : N
   % FORWARD SCATTERING - J_BS ZERO WHEN EXCLUDING back scattering SWEEPS
       sigma = 0;

       for q = 1 : p - 1
           r_j_i = dist(EFIE_group_points(q,1), EFIE_group_points(q,2),
    EFIE_group_points(p,1), EFIE_group_points(p,2));
           % inclusion of the back scattering term
           sigma = sigma + (J_FS(q) + J_BS(q)) * Z_ji(Kzc, beta * r_j_i);
       end

       r = dist(tx_x, tx_y, EFIE_group_points(p,1), EFIE_group_points(p,2));
       J_FS(p) = (Ei(beta * r) - sigma) / Z_ii;
   end

   for p = N - 1 : -1 : 1
   % BACK SCATTERING
       sigma = 0;

       for q = N : -1 : p + 1
       % iterating from the end towards the point immediately in front of p
           r_j_i = dist(EFIE_group_points(q,1), EFIE_group_points(q,2),
    EFIE_group_points(p,1), EFIE_group_points(p,2));
           sigma = sigma + (J_BS(q) + J_FS(q)) * Z_ji(Kzc, beta * r_j_i);
       end
       % absence of incident field, avoid double-counting it
       J_BS(p) = -sigma / Z_ii;
   end
end
J_fsbs = J_FS + J_BS;
```

Additional iterations occur, back scattering is deployed as the reverse of forward scattering.

Summation occurs after the completion of forward and back scattering calculations, and not each time a forward and back scattering vector is calculated.

E_{tot} - Field above the surface

```
1
2  for p = 1 : N
3        sigma = 0;
4
5        for q = 1 : p
6            % include the height of the observer in the distance calculations
7            r_j_i = dist(EFIE_group_points(q,1), EFIE_group_points(q,2),
    EFIE_group_points(p,1), EFIE_group_points(p,2) + obs_y_offset);
8
9            % while the J loops in Forward-Backward will iterate over points
    behind for forward
10           % scattering and points beyond for back scattering, the field above
    the surface
11           % calculation only considers forward scattering
12           sigma = sigma + J(q) * Z_ji(Kzc, beta * r_j_i);
13       end
14
15       r = dist(tx_x, tx_y, EFIE_group_points(p,1), EFIE_group_points(p,2) +
    obs_y_offset);
16       E_tot (p) = Ei(beta * r) - sigma;
17    end
18
19
20 function Ei = Ei(x)
21     Ei = H02(x);
22 end
23
24 function Z_ji = Z_ji(Kzc, x)
25     Z_ji = Kzc * H02(x);
26     % including the bottom line for sanity check, but this library function
27     % runs quite slowly
28     % besselh(0, 2, beta * r_ji);
29 end
30
31 % happily, euclidean distance is always positive
32 function d = dist(x_1, y_1, x_2, y_2)
33     d = sqrt((x_2 - x_1)^2 + (y_2 - y_1)^2);
34 end
35
36 % zeroth order hankel function of the second kind
37 function hankel_out = H02(x)
38     hankel_out = sqrt(2 / (pi * x )) * exp(-1j * (x - (pi / 4)));
39 end
```

A5 K Calculation Code

```
function K_out = calculate_k(L)
   % function to calculate an approximate value for K generally using
   % either the "l in the group centre" approach or the
   % "l at the group end" approach
   sigma_k = 0;
   global beta Z_ii sfc_pts grp_segs


   for p = 1 : grp_segs
       % s_p is the distance between start point and point p. exclude when p
       % and l are the same point, as the hankel function when given 0 for
       % the distance parameter returns (inf, inf)
       s_p = dist(sfc_pts(1, :), sfc_pts(p, :));

       if p ~= L
           % exp maybe distance from start of group to j instead of j to l
           s_p_l = dist(sfc_pts(L, :), sfc_pts(p, :));
           loop_val = exp(-1j * beta * s_p) * Z_ji(beta * s_p_l); % -1j == iota
           sigma_k = sigma_k + loop_val;
       else
           loop_val = exp(-1j * beta * s_p) * Z_ii;
           sigma_k = sigma_k + loop_val;
       end
   end

   % Z_ii denominator for entire summation, can be taken out of those loops for
     speed
   K_out = 1 - (sigma_k / Z_ii);
end
```

Calculation for K strictly following the [5].

```
function K_out = k_with_theta(p, alpha, f_theta)
% This function is a generalised K calculator that can be used for either calculating K once at the % start or at every iteration with a new angle
% Where theta is being ignored, pass 0 to alpha and 1 to f_theta
% alpha parameter is the plate's orientation to the horizontal, used to interpolate group points
% f_theta parameter instead of cos(alpha) later on as sin and cos were both used
% p is the index of the group under consideration
   sigma_k = 0;

   global beta Z_ii  num_group_points group_points group_centres d_s

   group_start = group_points(p, :);
   centre = group_centres(p, :);

   pts = [
       group_start(1) + d_s * cosd(alpha) * [0 : num_group_points] ;
       group_start(2) + d_s * sind(alpha) * [0 : num_group_points]
   ]';

   for i = 1 : num_group_points
       % points interpolation here is a hangover from earlier interpolation of points. with the
       % updated approach developed during urban profile testing abs(7-i) * d_s would be correct as
       % the points become interpolated along the hypotenuse instead of along x
       sj = dist(pts(i, :), centre);

       if i == 7
           % oddly, no additional d_s term is necessary for this term
           k_temp = ( exp(-1j * beta * sj * f_theta) * Z_ii ) / Z_ii;
       else
           % below is the d_s term in the denominator that was missing
           k_temp = ( exp(-1j * beta * sj * f_theta) * Z_ji(sj) ) / (d_s * Z_ii );
       end

       sigma_k = sigma_k + k_temp;
   end
   % to be strict, K should be 1 - sigma to yield approx. 5, but absolute value needs to be
   % taken instead to preserve the positivity that the function otherwise returns. sigma_k at
   % this stage is -4.1052
   K_out = abs(sigma_k);
   end
```

The K calculation code that resulted in K = 4.1052. Different from the strict interpretation in the extra Δs term and what distances it calculates (paper defines s_j for the exponential function as distance from start of group to point j, and the s_j that's passed to Z_{ji} as being the

distance from point j to the centre of the group).

A6 Field Extrapolation Method: Forward-Backward

J - Surface Current

```
J = zeros(n_grps, 1);
% BACK AND FORWARD SCATTERING
J_FS = zeros(n_grps, 1);


J_BS = zeros(n_grps, 1);

for loop_iter = 1 : 10
   %  FORWARD SCATTERING
   for p = 1 : n_grps
       sigma = 0;

       for q = smooth_region_index : p - 1
           r_j_i = dist(group_centres(q, :), group_centres(p, :));
           sigma = sigma + ((J_FS(q) + J_BS(q)) * Z_ji(r_j_i));
           end
       end

       r = dist(tx_pt, group_centres(p, :));
       J_FS(p) = (Ei(beta * r) - (K * sigma)) / Z_ii;
       end
   end

   for p = n_grps - 1 : -1 : smooth_region_index
       %   BACK SCATTERING
       sigma = 0;

       for q = n_grps : -1 : p + 1
           r_j_i = dist(group_centres(q, :), group_centres(p, :));
           sigma = sigma + ((J_BS(q) + J_FS(q)) * (Z_ji(r_j_i)));
       end

       J_BS(p) = -(K * sigma) / Z_ii;
   end
end
J = J_BS + J_FS;
```

E_{tot} - Field above the surface

```
E_tot = zeros(n_grps, 1);
for p = 1 : n_grps
   sigma = 0;

   for q = smooth_region_index : p
       r_j_i = dist(group_centres(q, :), (group_centres(p, :) + [0, obs_y]));
       sigma = sigma + (J(q) * Z_ji(r_j_i));
   end

   r = dist(tx_pt, group_centres(p, :)  + [0, obs_y]);
   E_tot(p) = Ei(beta * r) - (K * sigma);
end



function Ei = Ei(x)
   % x here is beta * r_ij
   Ei = H02(x);
end

function Z_ji = Z_ji(r_ij)
   % beta * r_ij not passed as single x param as beta is used more than once
   global d_s beta eta
   Z_ji = d_s * H02(beta * r_ij); % * ((beta * eta) / 4)

end

% happily, euclidean distance is always positive
function d = dist(pt1, pt2)
   d = sqrt((pt2(1) - pt1(1))^2 + (pt2(2) - pt1(2))^2);
end

% zeroth order hankel function of the second kind
function hankel_out = H02(x)
   % avoiding besselh(0,2,x) as it adds computation time to get same result
   hankel_out = sqrt(2 / (pi * x )) * exp(-1j * (x - (pi / 4)));
end
```

A7 Including θ

```

K_sine_array = zeros(n_grps,n_grps);
K_cosine_array = zeros(n_grps,n_grps);

for p = 1 : n_grps
   p2c2 = group_centres(p, :) - group_points(p,:);
   % p2c2(2) is y2-y1, p2c2(1) is x2-x1
   m2 = p2c2(2) / p2c2(1);
   % orientation of group obtained by slope of vector from start point to
    centre
   alpha_FS = atand(m2);
   % The self-term shouldn't be 0, but should be the standard K
   K_sine_array(p,p) = K;
   K_cosine_array(p,p) = K;

   for q = 1 : p - 1
       theta_forward = calculate_theta(group_centres(q, :), group_centres(p, :)
    , alpha_FS);
       K_sine_array(p,q) = k_with_theta(p, alpha_FS, sind(theta_forward));
       K_cosine_array(p,q) = k_with_theta(p, alpha_FS, cosd(theta_forward));

       % slope of qth point for points behind the pth point, used for
    calculating
       % the angle from p to q for back scattering ks

       % same process as ps with indices swapped for direction change
       p1c1 = group_centres(q, :) - group_points(q,:);
       m1 = p1c1(2) / p1c1(1);
       alpha_BS = atand(m1);

       theta_backward = calculate_theta(group_centres(p, :), group_centres(q,
    :), alpha_BS);

       %  upper triangular matrix for Ks is back scattering, like how
       %  it was for impedance matrix
       K_sine_array(q,p) = k_with_theta(q, alpha_BS, sind(theta_backward));
       K_cosine_array(q,p) = k_with_theta(q, alpha_BS, cosd(theta_backward));
   end
end

J_FS_ONLY = zeros(n_grps, 1); % enforcing strict separation of this quantity
```

```
    from paranoia

for test = 1 : 4
    J = zeros(n_grps, 1);
    % BACK AND FORWARD SCATTERING
    J_FS = zeros(n_grps, 1);
    J_BS = zeros(n_grps, 1);

    for loop_iter = 1 : 10
        %  FORWARD SCATTERING
        for p = 1 : n_grps
            sigma = 0;

            for q = smooth_region_index : p - 1
                r_j_i = dist(group_centres(q, :), group_centres(p, :));

                % test == 1 and 2 are the sine and cosine inclusion loops, in
    them the K for the
                % current combination of points is used here, not at the end
                if test == 1
                    sigma = sigma + (K_sine_array(p,q) * ((J_FS(q) + J_BS(q)) *
    Z_ji(r_j_i)));
                elseif test == 2
                    sigma = sigma + (K_cosine_array(p,q) * ((J_FS(q) + J_BS(q))
    * Z_ji(r_j_i)));
                elseif test == 3 && loop_iter == 1
                    sigma = sigma + ((J_FS_ONLY(q)) * Z_ji(r_j_i));
                elseif test == 4
                    sigma = sigma + ((J_FS(q) + J_BS(q)) * Z_ji(r_j_i));
                end
            end

            r = dist(tx_pt, group_centres(p, :));

            if test < 3
                % avoiding use of K here
                J_FS(p) = (Ei(beta * r) - (sigma)) / Z_ii;
            elseif test == 3 && loop_iter == 1
                J_FS_ONLY(p) = (Ei(beta * r) - (K * sigma)) / Z_ii;
            elseif test == 4
                J_FS(p) = (Ei(beta * r) - (K * sigma)) / Z_ii;
            end
        end

        if test == 3 && loop_iter == 1
            % break for the forward scattering comparison
            break
        end

        for p = n_grps - 1 : -1 : smooth_region_index
            %   BACK SCATTERING
            sigma = 0;
```

```
86
87          for q = n_grps : -1 : p + 1
88              r_j_i = dist(group_centres(q, :), group_centres(p, :));
89
90              % same general rules apply, but the indices for K arrays
    reversed
91              if test == 1
92                  sigma = sigma + K_sine_array(q,p) *  ((J_BS(q) + J_FS(q)) *
    (Z_ji(r_j_i)));
93              elseif test == 2
94                  sigma = sigma + K_cosine_array(q,p) *  ((J_BS(q) + J_FS(q))
    * (Z_ji(r_j_i)));
95              elseif test == 4
96                  sigma = sigma + ((J_BS(q) + J_FS(q)) * (Z_ji(r_j_i)));
97              end
98          end
99
100         if test < 3
101             J_BS(p) = -sigma / Z_ii;
102         elseif test == 4
103             J_BS(p) = -(K * sigma) / Z_ii;
104         end
105     end
106 end
107 J = J_BS + J_FS;
108
109 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
110 %              Second loop to find E_tot up to (700,1)                  %
111 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
112
113 for p = 1 : n_grps
114     sigma = 0;
115
116     for q = smooth_region_index : p
117         r_j_i = dist(group_centres(q, :), (group_centres(p, :) + [0, obs_y])
    );
118
119         if test == 1
120             sigma = sigma + K_sine_array(p,q) * (J(q) * Z_ji(r_j_i));
121         elseif test == 2
122             sigma = sigma + K_cosine_array(p,q) * (J(q) * Z_ji(r_j_i));
123         elseif test == 3
124             sigma = sigma + (J_FS_ONLY(q) * Z_ji(r_j_i));
125         elseif test == 4
126             sigma = sigma + (J(q) * Z_ji(r_j_i));
127         end
128     end
129
130     r = dist(tx_pt, group_centres(p, :)  + [0, obs_y]);
131
132     if test < 3
133         E_tot(p) = Ei(beta * r) - (sigma);
```

```
        else
            E_tot(p) = Ei(beta * r) - (K * sigma);
        end
    end
    % plotting logic included, to serve as demonstration of how the multi-graphs
      were generated
    figure(2);
    hold on

    plot(group_centres(:,1), 20 * (log10(abs(E_tot))));
    hold off

end
plot(EFIE_group_points(:,1), 20 * (log10(abs(EFIE_E_tot_cgs))));
plot(EFIE_group_points(:,1), 20 * (log10(abs(EFIE_E_tot_fs))));
plot(EFIE_group_points(:,1), 20 * (log10(abs(EFIE_E_tot_fsbs))));









hold off
legend(test_labels, 'FontSize', 14);
title("Plots of 20*log10(|E_t_o_t|): ");
ylabel("|E_t_o_t| (dB)");
xlabel("Metres");


function Ei = Ei(x)
    % x here is beta * r_ij
    Ei = H02(x);
end

function Z_ji = Z_ji(r_ij)
    % beta * r_ij not passed as single x param as beta is used more than once
    global d_s beta eta
    Z_ji = d_s * H02(beta * r_ij); % * ((beta * eta) / 4)

end

% happily, euclidean distance is always positive
function d = dist(pt1, pt2)
    d = sqrt((pt2(1) - pt1(1))^2 + (pt2(2) - pt1(2))^2);
end

% zeroth order hankel function of the second kind
function hankel_out = H02(x)
    % avoiding besselh(0,2,x) as it adds computation time to get same result
    hankel_out = sqrt(2 / (pi * x )) * exp(-1j * (x - (pi / 4)));
end


```

```
185
186 function theta = calculate_theta(c1, c2, alpha)
187    % normal is +90 degrees to group orientation, -90 would send it into the
       earth
188    N_dir = alpha + (90);
189    % just cos and sin as the normal vector has unit length
190    N_vec = [cosd(N_dir), sind(N_dir)];
191    ray_vec = c2 - c1;
192    % formula cos(c) = v1.v2 / |v1|.|v2| for angle between vectors v1, v2
193    c = dot(N_vec, ray_vec) / (norm(N_vec) * norm(ray_vec));
194    theta = acosd(c);
195 end
```

Code attempting to account for the changing thetas at every iteration of forward and backward scattering. It never proved more accurate than the most accurate forward scattering, and had a much greater computational and memory capacity overhead.