Born Approximation, Scattering, and Algorithm

Alex Martinez[a], Mengqi Hu[a], Haicheng Gu[a], and Zhijun Qiao[a]

[a]Department of Mathematics University of Texas Pan American, 1201 W University Dr, Edinburg 78539, USA

ABSTRACT

In the past few decades, there were many imaging algorithms designed in the case of the absence of multiple scattering. Recently, we discussed an algorithm for removing high order scattering components from collected data. This paper is a continuation of our previous work. First, we investigate the current state of multiple scattering in SAR. Then, we revise our method and test it. Given an estimate of our target reflectivity, we compute the multi scattering effects in the target region for various frequencies. Furthermore, we propagate this energy through free space towards our antenna, and remove it from the collected data.

Keywords: Born approximation, Neumann series, iterative imaging, multiple scattering, SAR imaging.

1. INTRODUCTION

Electro-magnetism theory plays a very important role in areas ranging from communication to astronomy. There is plenty of history behind the development of this theory. It is the model for all types of Electro-Magnetic(EM) radiation, from visible light to radio waves. Radar is an application of electromagnetism based on three principles: EM radiation travels at a finite constant speed while in a vacuum; radiation emitted from an antenna is contained in some localized region of space; targets scatter radiation. Initially radar was based on generated radio waves and was primarily concerned with detection and ranging as the name implies. In order to perform any task in radar we need to be able to transmit, and measure radiation. For a brief history of radar, and an introduction to this topic can be found in the reference.[1]radar is great for long distance high resolution ranging.

In this work, we aim at studying multiple scattering in SAR image reconstruction for stationary targets. In particular, we want to mitigate errors introduced by the Born(single scattering) approximation. To this end, we use a rough estimate of our targets to estimate higher order scattering effects and remove them from collected data. We will use the algorithm developed in,[4] called butterfly algorithm, for efficient evaluation of the scattering series that arises from EM wave equation.This work has also been extended to parallel processing.[7] In our paper, we provide a detailed implementation of the butterfly algorithm described in.[4] In discrete form, the integral we want to evaluate turn into a matrix, vector product. The vector represents our targets, and the matrix, the scattering operator. To compute scattered data we have to evaluate this product efficiently. For every row, if we could find a low dimensional matrix, and vector whose product approximated the row of scattering data product, then we could evaluate the matrix product very efficiently. This is not trivial to do. On the other hand, if we only perform the matrix product with only one column, then depending on the properties of the matrix, we can find a low dimensional matrix, and vector to approximate the partial product for all rows. The butterfly algorithm is about propagating these partial matrix product approximation to find a full matrix product approximation. We will review the algorithm developed in[4] and discuss a serial implementation of it. This is a continuation of.[8,9] Other recent work in multiple scattering is done in.[10] In the following sections we present the EM wave model, discuss multiple scattering, the butterfly algorithm, and how to compensate for multiple scattering.

Synthetic Aperture Radar (SAR) is an imaging scheme that involves the exploitation of scattered radiation measured along a continuous path in space. The prototype for SAR is an antenna attached to an airplane which goes along some path $\gamma(\mathrm{t})$. The antenna generates radiation directed towards some region, and listens for energy that is scattered back to the antenna(back scatter). After a sufficiently long time we assume that there is no more back scatter to wait for from any practical returns from the probe, so the antenna generates more radiation and listens again. This process is repeated several times during the flight path. Essentially a SAR system probes

Zhijun Qiao: Email: qiao@utpa.edu.

Radar Sensor Technology XIX; and Active and Passive Signatures VI, edited by G. Charmaine Gilbreath
Kenneth I. Ranney, Chadwick Todd Hawley, Armin Doerry, Proc. of SPIE Vol. 9461, 94611C
 · doi: 10.1117/12.2177864

a region of interest from several different points in space, and uses collected data to form a coherent image of the region. In order to fully appreciate this scheme we need to understand the underlying forces that govern this model.

2. BACKGROUND AND EM MODEL

In the macroscopic regime, the appropriate model for electromagnetic radiation is Maxwell's equations. A classic treatment of the full equations is given in.[11] These equations can be difficult to work with for imaging. One can show under certain conditions(no polarization)[3,12–14] that we can work with an inhomogeneous wave equation instead. Following the literature we adopt this as our model. There are down-sides to this. Working with only one component of the EM fields ignores useful effects such as polarization but captures important aspects of electromagnetism such as finite speed, and the ability to cause interference. Simplifying the model leads to more efficient algorithms at the cost of accuracy. That means,the wave model captures several important effects regarding the scattering process, and is more manageable than the full Maxwell's equations. This is a often used model for SAR imaging, as can be seen by its proliferation in the literature.[3,4,12,15–19]

As we know that SAR is tied very heavily to its particular application, and the application controls what approximations can be made. We have assumed that there are no moving targets. This corresponds to a short path if our antenna is moving fast enough. We also assume that our antenna is stationary during each transmission and measurement cycle. Obviously the antenna must either keep moving with constant velocity, or more undergo some acceleration is it is to cover some region of space, but this can be compensated for since we can just record (approximately) how fast the antenna moved during any transmit receive cycle. We work with one component of the EM field denoted by $u(x,t)$ which is a function of space $x \in \mathbb{R}^3$, and a time t. This is a scalar valued function. The source of the wave equation $j(x,t)$ is dependent on the change in current density as a function of time, and the spatial variation of the charge density. We could also work directly with the scalar potential that generates the em field, but it does not change much. For simplicity we suppose that we have complete knowledge of $j(x,t)$. This is never really the case due to background radiation, but we take this view since it may lead to some insight into handling the noisy case. Typically when one processes collected data they correlate with the transmitted signal to minimize the effects of certain types of noise. The field satisfies the inhomogeneous wave equation in $3+1$ dimensions

$$\left(\nabla^2 - \frac{\partial_t^2}{c(x)^2}\right) u = -j(x,t) \tag{1}$$

with wave speed $c(x)$.

All of the information concerning a region of interest contained in measurements of $u(x,t)$ is in $c(x)$. In particular, if the field is in a vacuum, then $c(x) = c_0$. If we could measure $u(x,t)$ near a given point, then we could (mostly)reconstruct the wave speed by $c(x)^2 = \frac{\nabla^2 u(x,t)+j(x,t)}{\partial_t^2 u(x,t)}$ as long as the denominator is not zero for some time. Of course we would need local measurements on a scale comparable to what resolution we hope to attain which defeats the purpose of the system. We must consider how we treat the "functions" u, j and c. Should we treat everything as a generalized function (distribution),[20] or is a standard function powerful enough. Certainly we want our model to make sense, but distributions can be subtle. We cannot in generally multiply distributions in a meaningful way. We can multiply distributions and smooth functions though. We may also differentiate distributions, and of course tell them apart so that the equality symbol make sense. We consider u and j as distributions. Here we cannot assume $c(x)$ is continuous since many interesting objects in life have apparent discontinuities such as walls, planes, cars, and almost anything we might be interested in studying. The change from free space to target waves speed is often abrupt. We can assume that $c(x)$ is real and bounded by c_0. Sometimes $c(x)$ is considered as complex, and the imaginary part corresponds to energy lost to absorption retransmission process. As we will see soon enough, em waves do dissipate energy as they travel since they spread out in some geometric fashion. Rather than work directly with the wave speed we would like a function that accurately describes the support of our targets. That is we want a function that is zero in free space. We define the reflectivity function[3,12,15–19,21] by

$$\nu(x) = \frac{1}{c_0^2} - \frac{1}{c(x)^2}. \tag{2}$$

The reflectivity function gives us an idea of what our targets looks like. It is zero in the absence of targets, and non zero when targets are present. The goal of RADAR imaging in this wave model is to solve for (2). The reflectivity function is very closely tied to the field $u(x,t)$, and as we will see, a given reflectivity function determines u under a weak scattering assumption. The reflectivity is also related to the index of refraction of a material by

$$\nu(x) = 1 - n(x)^2. \tag{3}$$

It is very difficult to get light to completely stop moving, so unless we are imaging a very expensive science experiment, we can assume that $c(x) \neq 0$. We can also assume that are targets are not infinitely large so that $c(x) = c_0$ outside of some compact set $K \subset \mathbb{R}^3$. This means that ν is compactly supported. For this reason ν must have finite energy. That is ν is square integrable: $\nu(x) \in L^2(\mathbb{R}^3)$. We may also assume that $\nu \in L^p$ for any p since it bounded and compactly supported, but L^2 is the most useful space. Similarly we assume that any generated wave must contain a finite amount of energy, and so can also be found in $L^2\mathbb{R}^4$. There are some subtleties involved when working with these spaces. We must keep in mind that the L^p space are not sets of functions strictest sense, since two functions are equivalent if they are equal almost everywhere. The L^p spaces are sets of classes of functions that all look the same with respect to integration. Continuous functions are point wise objects so we can see what they look like at a given point in space and time. Distributions are not point wise at all, they are the linear continuous functions from a test space to the real numbers(functionals). Distributions act on other functions, so it does not make too much sense to represent evaluate a distribution at a single point. The test space we take generally affects the properties of the distributions. Most useful functions(those with finite energy) define unique distributions, so distributions are basically extended classes of functions. If the test space is smooth, and consist of fast decaying functions we can define differentiation(weak) on our distributions. For simplicity we may regard the source term j as a distribution. Distributions gives us the ability to understand point like objects in a useful way, since integrating over a single point gives the value 0.

We must consider how to represent recovered images for the user, since graphing is a point wise operation. To do this we can simply look at the action on a smooth function whose support is smaller than a pixel. In a sense this means we want an average value of our targets, and corresponds to expanding the target function into a localized basis and finding the corresponding coefficients. For now we consider c as square integrable. These are nice spaces to work with since it gives us an inner product, and a norm. Moreover, we have a well defined Fourier transform for ν, and the field u which will be useful for image formation latter. This is good news, since any kind of approximation to this needs to be made in some metric space, and vector spaces allow one to impose a finite structure on the problem in a straightforward manner. A normalised vector space is almost essential to any approximation problem, and we have that structure for reflectivity and wave functions. The solution space for the field u is not so clear. We can establish at least one solution based on a perturbation argument. We can rewrite the total field equation as

$$\left(\nabla^2 - \frac{\partial_t^2}{c_0^2}\right) u = -(j(x,t) + \nu(x)\partial_t^2 u). \tag{4}$$

The left hand side is the free space wave equation and can be solved by convolving with the Green's "function"

$$g(x,t) = \frac{\delta(t - |x|/c_0)}{4\pi|x|}. \tag{5}$$

Thus we have

$$u(x,t) = g(x,t) * (j(x,t) + \nu(x)\partial_t^2 u). \tag{6}$$

where $*$ denotes convolution over space and time. We call $g * j$ the incident field, and denote it by u^{in}. Note that if we are in free space, $\nu = 0$, and $u = u^{in}$. That is, we think of the incident field as the portion of u that satisfies the wave equation in free space. We can solve this for u by first defining the scattering operator L which takes a reflectivity function ν and an EM field u and produces a new EM field $L\{\nu, u\}$. It is evident that L that is bilinear. A bilinear operator may also be viewed as a family of linear operators indexed by one of its arguments. That is, for every fixed reflectivity our scattering operator is simply a linear operator. We define the scattering operator as

$$L\{\nu, u\} = g(x,t) * \nu(x)\partial_t^2 u. \tag{7}$$

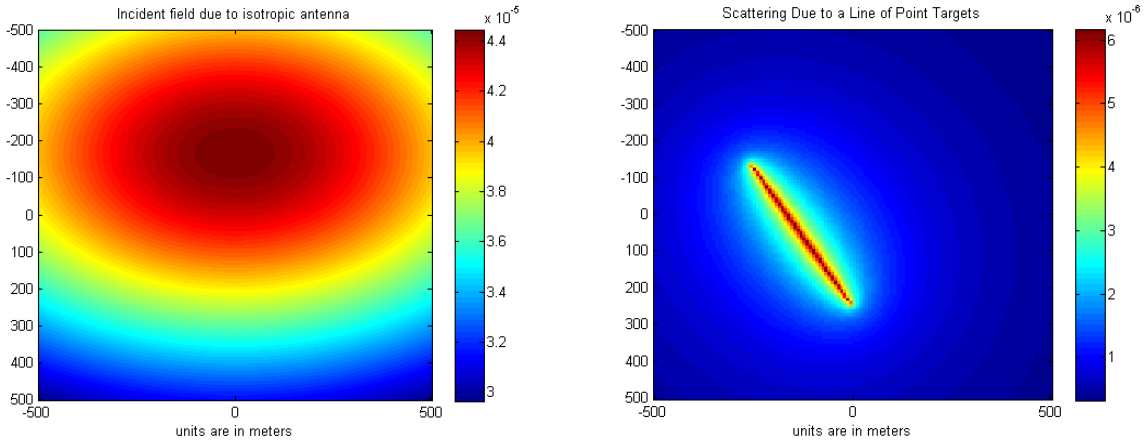

Figure 1. Incident and Born approximated scattered field on left and right, respectively.

Under this notation we have

$$u(x,t) - L\{v,u\} = u^{in}. \tag{8}$$

Adopting the convention that

$$L^n\{v,u\} = L\{v, L^{n-1}\{v,u\}\} \tag{9}$$

where $L^0 = I$ the identity map, we can solve for u as by computing the inverse of $I - L\{v,\cdot\}$ and applying to u^{in} to get

$$u(x,t) = \sum_{n=0}^{\infty} L^n\{v,u^{in}\}. \tag{10}$$

Such a series does not always converge. The series converges for any incident field if the norm of the scattering operator is less than 1, analogous to the geometric series. On the other hand, this is not necessary in order for the series to converge for a given incident field. It can be shown[22] that it is enough for $L^n\{v,u^{in}\} \to 0$. If the reflectivity scatters weakly enough, and the incident field does not vary too quickly, the terms will decay as needed. To solve for v is a very nonlinear problem. The purpose of the Born approximation is to remove this non-linearity by truncating the series to the first two terms. Physically, the first term in the series corresponds to radiation scattered away from the transmitting antenna. The second term corresponds to this first scattered field again interacting with targets and getting scattered again. In general the n-th term is the $n-1$-th term being scattered. Adding all of these together we get the scattered field. The scattered field is then the sum of these re-scattered fields. In SAR, we measure the field u at several points along a known path, and use the collected data to estimate v. If we measure far from our targets, the targets are far from each other, and our signal is not too high frequency, then the multiple scattering effects will be weak. This is not the only solution to the wave equation. There are different Green's functions for the wave equation. We have chosen one that propagates energy away from a source in a spherical manner. This is the unique solution if we assume the Sommerfeld radiation condition which is that the solution we consider decays inversely proportional to the distance from the source, and the change of energy radially at a given wave number is equal to the wave number times u times i plus a term inversely proportional to the distance from the source.[21] We will use the series to compute the multiple scattering effects a coarse estimate of the target reflectivity creates, and then subtract this from the data. Basically the model corresponds to a sphere emanating from each point on the antenna and propagating outward at the speed of light, which eventually will contact a target. Every point on the sphere that is contacting a target will start a new sphere emanating from that point with energy proportional to the second derivative of the incoming wave. We must truncate the series after some finite number of scattering events. We want to be able get a large number of terms in the series for as few applications of the scattering operator as we can. For this reason, we look at factored forms of the series. Consider the geometric series to k terms

$$s_k = 1 + x + x^2 + x^3 + \cdots + x^k. \tag{11}$$

How fast can we compute this series? For a general k degree polynomial, Horner's method may be used to evaluate it at a single point in k multiplications and additions and multiplications by writing the sum as

$$s_k = 1 + x(1 + x(1 + \cdots (1 + x) \cdots))) \tag{12}$$

and evaluating it from the inside. We can write this in an iterative fashion as

$$s_k = (1 + x)s_{k-1} \tag{13}$$

where $s_0 = 1$. These products correspond to evaluating the scattering operator. In fact, if we wanted to evaluate the geometric series for a square matrix, we could do better using a fast exponentiation method, and a different factorization. The flaw in this route is that we can only do fast multiplication for the scattering operator and not for its squares. We will use Horner's method to compute the multiply scattered data.

3. IMAGE RECONSTRUCTION

Here we briefly derive an image reconstruction method for SAR to test our algorithm against. This is basically the polar format algorithm without the interpolation. The idea in SAR is to move through space and measure the scattered field to solve for the reflectivity function. In this form the problem is still fairly difficult even if we completely know the scattered field. For this reason one typically assumes the higher order terms in the sum are negligible. This is called the Born approximation. The incident field is dependent on the source in the original model. This in turn is dependent on a known input signal. One then uses an imaging operator I on the Born approximated data and the input signal to estimate the reflectivity function. We use this estimated image to estimate and remove the higher order terms from the collected data. From experimentation it is evident that estimated reflectivity functions tend to be scaled improperly. That is to say, the positions of targets are accurate, and their relative values are reasonable, but the estimated reflectivity values are off. To resolve this we simply scale the estimated high order scattering. In addition, estimates tend to look fairly noisy, so we must perform some denoising/focusing operator $\mathcal{N}$ before using the image estimate. Formally we create a sequence of estimates $\{\hat{V}_k^N\}$ in $\mathcal{V}$ by

$$\hat{V}_{k+1}^N = I\left\{ d - \sum_{n=2}^{N} L^n\{\lambda_k \mathcal{N}\{V_k^N\}, u^{in}\} \right\} \tag{14}$$

where λ_k is a scaling parameter. This is the iterative method mentioned in the title. The powers of the scattering operator are linear in the second argument, and homogenous of order n in the first argument, so we can scale by λ_k^n after scatting the data. The order does matter when round off error is taken into account, so we scale before scattering, but after noise reduction. We will discuss the specifics later, but first let's consider when it is appropriate to use this method. We do not establish convergence of this method, but test it computationally. Let u^{sc+} denote the higher order scattered terms in the Neumann series. That is to say, $u = u^{in} + L\{v, u^{in}\} + u^{sc+}$. Let us work in the frequency domain. If we assume a point like, isotropic antenna that is stationary between transmission and reception, the incident field can be found in the closed form

$$u^{in}(x, w) = p(w)G(x - x_0, w) \tag{15}$$

where $p(w)$ is a known input signal, and x_0 is the antenna position. Then, noting that G is even we have that

$$L\{v, u^{in}\} = -w^2 p(w) \int_{\mathbb{R}^3} G(x - z, w)v(z)G(x_0 - z, w)\, dz. \tag{16}$$

Our data d is a set of samples of the scattered field due to different antenna locations, and frequencies. Our antenna is assumed to be essentially stationary during transmissions, so that our data

$$d(x_0, w) = -w^2 p(w) \int_{\mathbb{R}^3} G(x_0 - z, w)^2 v(z)\, dz. \tag{17}$$

One may then make the far field approximation and assume that $\|x\| >> \|z\|$ to have that $\|x-z\| \approx \|x\| - \hat{x}\cdot z$, and $\frac{1}{\|x-z\|} \approx \frac{1}{\|x\|}$ which leads to

$$G(x_0 - z, w) \approx \frac{e^{-iw\frac{\|x\|}{c_0}} e^{iw\frac{\hat{x}\cdot z}{c_0}}}{4\pi\|x\|} \tag{18}$$

so that our data becomes

$$d(x_0, w) \approx -\frac{w^2 p(w) e^{-2iw\frac{\|x_0\|}{c_0}}}{(4\pi)^2\|x_0\|^2} \int_{\mathbb{R}^3} e^{2iw\frac{\hat{x_0}\cdot z}{c_0}} v(z)\, dz. \tag{19}$$

Let

$$S(w, p, x_0) = -\frac{w^2 p(w) e^{-2iw\frac{\|x_0\|}{c_0}}}{(4\pi)^2\|x_0\|^2}. \tag{20}$$

Then

$$d(x_0, w) \approx S(w, p, x_0)\mathcal{F}^{-1}\{v\}(2w\frac{\hat{x_0}}{c_0}). \tag{21}$$

Since v is square integrable, we may write

$$v(x) = \left(\frac{1}{2\pi}\right)^3 \int_{\mathbb{R}^3} e^{-ix\cdot\xi}\mathcal{F}^{-1}\{v\}(\xi)\, d\xi. \tag{22}$$

Then we can perform the Fourier transform of the data divided by S for all frequencies that the input p is supported on. The data we have is not on a uniform grid which is needed for the traditional FFT. Typically one will interpolate to such a grid to speed up computation. We use the Riemann sum directly.

4. COMPUTATION OF THE SCATTERING OPERATOR AND NEUMANN SERIES

There are several efficient methods to evaluate the oscillatory integral that defines the scattering operator.[4,5] Data is often given in the frequency domain, so we consider computation there. In the frequency domain the scattering operator becomes

$$L(v, u) = -w^2 \int_{\mathbb{R}^3} \frac{e^{-iw\frac{\|x-z\|}{c_0}} v(z) u(z, w)}{4\pi\|x - z\|}\, dz \tag{23}$$

which is in tbe form of a Fourier integral operator. We apply the butterfly algorithm to evaluate this integral efficiently. We can efficiently[23] evaluate a truncation of the scattering series by applying Horner's method to calculate the scattered field by evaluating

$$u_N^{sc} = L\{u^{in} + L\{u^{in} + \ldots L\{u^{in}\}\}\} = L\{u^{in} + u_{N-1}^{sc}\}. \tag{24}$$

bottom up ($u_1^{sc} = L\{u^{in}\}$) analogous to the geometric series we mentioned earlier. At this point we did not use a noise reduction method. We will pick a fixed scaling parameter λ for each step in the iteration.

5. SIMULATION

In this section we provide some simulation to test our method. Basically, we need to compute the multiply scattered field at the point of measurement. To do this we need to have a discrete representation of the target reflectivity. There are many ways to discretized a continuous object, but for simplicity we will consider $v(x)$ as a $2-d$ object(assuming height is negligible) that is piecewise constant on regions of size $res \times res$. We assume that the reflectivity function is only supported inside a box of size $length \times width$ with center (x_c, y_c). That is v is represented by $\frac{length\times width}{res^2}$ bins each corresponding to a constant reflectivity value. The imaging problem is then to recover the value of the reflectivity inside each bin. The scattering problem is to map the reflectivity function to scattered data measured at the antenna location. This is done by evaluating the Neumann series described above. We do this using Horner's method. That is, we take the incident field, scatter it, then add the

incident field, scatter the result, add the incident field, scatter, and repeat till we are content. The scattering operator takes the form of a Fourier integral operator, and we use this fact to apply the fast integration algorithm seen in. In order to perform the simulation using the butterfly algorithm we need to do a few things. We need to implement the butterfly algorithm for the scattering operator. We need to implement a fast reconstruction method. This can also be done using the butterfly algorithm as seen in. Then we need to combine these to compensate for multiple scattering as we have described. The main work is in implementing the butterfly algorithm.

The butterfly algorithm is an algorithm that exploits multi scale pairing of regions to construct low rank approximations each row of the matrix product in an efficient manner. There are many variants of these algorithms.[?] Hence in the literature there are butterfly algorithms. Since we will only be working with the particular butterfly algorithm developed in.[4]

5.1 Low Rank Approximations

In any linear problem, one must be able to perform matrix operations efficiently. Consider multiplying an $N \times N$ matrix A with an N dimensional vector x. How can we do this fast? For a general matrix, computing the action on a vector takes $O(N^2)$ steps since we must multiply each row of A with x. If A is diagonal we only need to perform $O(n)$ multiplications, but most matrices are not diagonal. If the matrix is low rank with rank r, then it can be factored into a product $Ax = BCx = B(Cx)$ where B is $n \times r$ and Cx is $r \times 1$. This means that the product Ax can be found in terms of a low rank matrix times a low rank vector. This means that we can perform a matrix multiplication in $2rn$ steps by multiplying Cx first to get a r vector, then $B(Cx)$ to get an n vector. A being rank r means that there is a $\delta \in \mathbb{R}^r$, and a $n \times r$ matrix A' such that $Ax = A'\delta$. The action of our matrix on a vector can be approximated by the action of a low rank matrix acting on a low rank vector. The elements in the vector δ are called equivalent weights. Finding the matrix and vector is not trivial but it can be done if we exploit analytic properties of the entires of the matrix.

This would be nice, but most matrices that are not full rank are not necessarily low rank. The next relaxation we can consider is that A be approximately low rank. By this we mean for any $\epsilon > 0$ there exist a matrix A' with rank r_{eps} such that $\|A - A'\| < \epsilon$ where r_ϵ is much less than N. The rank depends on the amount of error we are willing to allow in our approximation. Even this is often too much to ask for. Generally there does not exist a low rank approximation for an entire matrix. The point is that global approximation is often quite difficult. Since global does not work local is the next step which immediately requires analysis. Consider any sub matrix contained in A. If it is low rank, then we may perform block matrix multiplication for a more efficient multiplication. It is difficult to see if a general matrix large has low rank sub matrices. This requires analysis of where the matrix entries come from. In the hierarchal matrix scheme,[?] one store matrices in a hierarchy of factored low rank matrices. This method is concerned with formating a matrix so that we can do efficient matrix algebra. The matrices that are done for are typically the result of elliptic PDEs. Once the hierarchical structure is built one can perform several matrix vectors efficiently. We will not make use of hierarchical matrices, but they are closely related to the butterfly algorithm. The butterfly algorithm is concerned with evaluating the matrix on a given vector.

Consider evaluating the integral operator $\int_0^1 k(x,y)f(y)dy$ numerically for several values of x. This is a fairy difficult problem since we would need to perform this integral for every x. If we use a Riemann sum to approximate the integral this is the same as the matrix problem above. If $k(x,y) = \sin(x+y)$, then we know that $k(x,y) = \sin(x)\cos(y) + \cos(x)\sin(y)$, so that $\int_0^1 k(x,y)f(y)dy = \sin(x)\int_0^1 \cos(y)f(y)dy + \cos(x)\int_0^1 \sin(y)dy$. That is, the kernel k can be decomposed as a sum of products of single variable functions separating the integration. We say that k has a separated representation. This greatly speeds things up. Now we only need to compute two integrals, store those numbers and for different values of x evaluate sin and cos. The key feature of this kernel is that it has a low rank(i.e the number of terms in the sum is small) separated representation allowing us to evaluate the integral very efficiently. The representation corresponds to a matrix factorization low rank matrix(tensor)

factorization. In general it is not possible to represent most functions in this manner, allowing a small amount of controlled error widens the space of functions that have such approximations. Having one approximation that is valid for all x and y is still rare. Restricting the domain of the kernel to a suitably small region often guarantees the existence of a low rank separated approximation. Such a restriction is called an admissibility condition. A pair of sets whose product satisfies the admissibility condition is said to be an admissible pair. This means we can perform the partial integral over the range of the y variable. The admissibility condition is exploited by the butterfly algorithm by pairing subsets of X and Y whose product satisfies the admissibility condition evaluate the integral.

5.2 The Butterfly Algorithm

The butterfly algorithm is a method for evaluating

$$F(x_m) = \sum_{n=1}^{N} k(x_m, y_n) f(y_n) \tag{25}$$

for all $m = 1, \dots N$ where $x_m = m/N$ and $y_n = n/N$(approximate integral over $[0,1]$). Consider two partitions X_l of $[0,1]$ with cell size $\frac{1}{2^l}$ and Y_l with cell size $\frac{1}{2^{L-l}}$ for some L and $l = 0, \dots, L$. The partition is a set of equivalent classes of points in $[0,1]$ where points are identified if they are in the same box. Each class is called a cell. The scale of the partition refers to the number of cells it has. These partitions are on opposite scale since they are inversely proportional. The first partition has 2^l cells, and the second 2^{L-l} cells. The number of pairs of cells from the two partitions is 2^L which is independent of the scale l. Let $A \in X_l$ and $B \in Y_l$. Then the product of A and B has area $1/2^L$ independent of l. Moreover if L is sufficiently large, the area becomes an admissibility condition for certain kernels. For each pairing at each scale, there exist a low rank separated approximation

$$k(x, y) \approx \sum_{t=1}^{q} \alpha_t^{AB}(x) \beta_t^{AB}(y) \tag{26}$$

for all $x \in A$ and $y \in B$. This is the natural extension of separation of variables, and is closely related to tensor decompositions.[?] Denote by $F_B(x)$ the sum restricted to points that are in B. Then

$$F_B(x_m) \approx \sum_{y \in B} \sum_{t=1}^{q} \alpha_t^{AB}(x) \beta_t^{AB}(y) f(y_n) = \sum_{t=1}^{q} \alpha_t^{AB}(x) \delta_t^{AB} \tag{27}$$

for all $x_m \in A$ where $\delta_t^{AB} = \sum_{y \in B} \beta_t^{AB}(y) f(y_n)$. That is, for any pairing, the restricted integral becomes a low rank matrix multiplying a low rank vector. So choosing L large enough guarantees that existence of an equivalent weight and matrix. When $l = L$, $Y_L = [0,1]$, so for each pair $F_B(x) = F(x)$ for all $x \in A$(i.e the sum is over all of Y). There exist a low rank separated for the integral for each of these pairs at this scale but the weights are not easy to construct. On the other hand, when $l = 0$ we have for all $x \in [0,1] \in X_0$ that $\delta_t^{AB} = \sum_{y \in B} \beta_t^{AB}(y) f(y_n)$. Since B is very fine, the sum will be very cheap to compute. At scale $l = 1, 2, \dots L-1$, each $A \in X_l$ has a parent $A_p \in X_{l-1}$ and each $B \in Y_l$ has children $B_c \in Y_{l-1}$. Each parent child pair satisfies the admissibility condition so

$$F_{B_c}(x_m) \approx \sum_{t_c}^{q} \alpha_{t_c}^{A_p B_c}(x_m) \delta_{t_c}^{A_p B_c} \tag{28}$$

for all $x_m \in A \subset A_p$. We also know that the partial sum over a cell B is equal to the sum of the partial sums over its children

$$F_B(x_m) = \sum_{c=1}^{2} F_{B_c}(x_m) \approx \sum_{c=1}^{2} \sum_{t_c}^{q} \alpha_{t_c}^{A_p B_c}(x_m) \delta_{t_c}^{A_p B_c}. \tag{29}$$

Then we have the approximate equation

$$\sum_{t=1}^{q} \alpha_t^{AB}(x_m) \delta_t^{AB} \approx \sum_{c=1}^{2} \sum_{t_c}^{q} \alpha_{t_c}^{A_p B_c}(x_m) \delta_{t_c}^{A_p B_c}. \tag{30}$$

This is an over determined linear system for δ_t^{AB} in terms of $\delta_{t_c}^{A_p B_c}$. The butterfly algorithm is the process of computing the weights for X_L and Y_L bottom up starting from X_0, Y_0. The butterfly algorithm requires that the separation rank is constant for different pairs. It also requires a depth L. The algorithm is then to compute the weights, and matrices $\alpha_t^{AB}(x_m)$ for all $A \in X_L$ and $B \in Y_L$ bottom up by finding approximate solutions to 30 and using the definition for $l = 0$. For the FIO case the matrices $\alpha_t^{AB}(x_m)$ will be constructed via Chebyshev interpolation in either x or y depending on if $l < L/2$, $l = L/2$, or $l > L/2$so will be known. This is because the low rank separation can be shown to be in monomials of the respective variable, and the Chebyshev interpolant is almost proportional to the best approximating monomial. The admissibility condition for the FIO case is that $dim(A)dim(B) < \frac{1}{N}$ This is for integration over one dimension. The same idea works in higher dimensions. The work we are following focuses on 2-d due to its ease of visualization and practicality. This is exactly the method we need.

An admissibility condition on the size of the support can be used with a sequence of partitions X_l, Y_l such that the product of any pair is admissible, to find equivalent weights bottom up for the entire sum starting by computing the equivalent weights by definition for $l = 0$. These final pairs correspond to evaluating the entire sum and we can now perform this very quickly since we have the weights. We have described the algorithm for integrating over one dimension. This is easily extendible to any dimension d. In this case each cell has 2^d children. We will work with 2 dimensions for the FIO algorithm.

5.3 Butterfly Algorithm For Fourier Integral Operators

In this section we give the butterfly algorithm from.[4] This algorithm was developed in[5] as well. We define a discrete Fourier integral Operator(DFIO) to be

$$F(x_{kl}) = \sum_{m=1}^{N} \sum_{n=1}^{N} a(x_{kl}, y_{mn}) e^{i\phi(x_{kl}, y_{mn})} f(y_{mn}) \tag{31}$$

where $x_{kl} = (k/N - 1/2N, j/N - 1/2N)$ and $y_{mn} = (m/N - 1/2N, n/N - 1/2N)$ where $k, l = 1, \ldots N$. This is an integral over and evaluated on the unit square. As we saw in the previous section the goal is to reconstruct equivalent weights to approximate the integral. To do this we need the partitions X_l and Y_l as before. The unit square is naturally partitioned. For example, with $[0, 1]$ partitioned into $[0, 1/2)$ and $[1/2, 1]$, then we know which points are in either set by looking at the first bit after the decimal point in the binary representation of the number. The number $1/4$ is given by $(.01)_2$ in binary, so it belongs to the first class. since its first bit after the decimal is 0. On the other hand $1/2 = (.1)_2$ in binary so it belongs to the second class since its bit is 1. This is also true for the unit box since we can look at each numbers bits after the decimal. At level l, the point x belongs to the box $A \in X_l$ if the first l bits of its first and second component correspond to A. In the algorithm itself we will need to know the center and dimensions of each cell in a given partition. This can be found analytically. This means we do not actually have construct any partition since we can figure out the parameters of the cells directly, and know membership. In the paper by Demanet the first step in the algorithm is to construct two quad trees(the sets of partitions). This is a booking tool and is not necessary since points in $[0, 1]^2$ are naturally partitioned. We will use this natural partitioning.

Recall that the one dimensional Chebyshev grid adapted to $[-1/2, 1/2]$ is given by

$$\{z_j = \frac{1}{2}\cos(j\pi/(q-1)) | 0 \leqslant j \leqslant q-1\}. \tag{32}$$

This can be adapted to any interval by appropriate scaling and shifting. The Lagrange basis polynomial for a given grid are given by

$$L_j(z) = \prod_{0 \leqslant k \leqslant q-1, k \neq j} \frac{z - z_k}{z_j - z_k}. \tag{33}$$

The 2-d Chebyshev grid is defined by taking tensor products of the 1-d grids $\{(z_{t_1}, z_{t_2})\}$. The Chebyshev basis functions are

$$L_t(z) = L_{t_1}(z_1) L_{t_2}(z_2) \tag{34}$$

where $z = (z_1, z_2)$ and $t = (t_1, t_2)$. The Chebyshev grid adapted for box B is denoted $\{y_t^B\}$. The basis polynomials for these grids are denoted $\{L_t^B\}$. Let $x_0(A)$ be the center of A, and $y_0(B)$ be the center of B. We need to choose a number of partitions L to use so that the admissibility condition is satisfied. Choose $L \geqslant 2\log_2(N)$ since we want the size of the finest boxes to be at most $\frac{1}{N} \times \frac{1}{N}$. Finally we need to choose the number q of Chebyshev points to use. This is not so trivial on what q should be. One thing we can do is run the algorithm several times with q being squared each time to see if we get noticeable improvements. We will just choose q to be some reasonably large number. The algorithm is then:

1. For $l = 0$, initialize
$$\delta_t^{AB} = e^{-i\phi(x_0(A), y_t^B)} \sum_{y \in B} L_t^B(y) e^{i\phi(x_0(A), y)} f(y) \tag{35}$$
for all $A \in X_0$, $B \in Y_0$, and $t = 1, \ldots q$. There are qN^2 weights to compute.

2. For $l = 1, \ldots, L/2$ compute the weights for all $A \in X_l$ and $B \in Y_l$ by
$$\delta_t^{AB} = e^{-i\phi(x_0(A), y_t^B)} \sum_c \sum_{t_c} L_t^B(y_{t_c}^{B_c}) e^{i\phi(x_0(A), y_{t_c}^{B_c})} \delta_{t_c}^{A_p B_c} \tag{36}$$

3. Switch representation. Compute weights by
$$\delta_t^{AB} = \sum_s a(x_t^A, y_s^B) e^{i\phi(x_t^A, y_s^B)} \delta_s^{AB}. \tag{37}$$
Note that $l = L/2$ is seen twice in this algorithm.

4. For $l = L/2 + 1, \ldots L$, compute weights by
$$\delta_t^{AB} = \sum_c e^{i\phi(x_t^A, y_0(B_c))} \sum_{t_p} L_{t_p}^{A_p}(x_t^A) e^{-i\phi(x_{t_p}^{A_p}, y_0(B_c))} \delta_{t_p}^{A_p B_c} \tag{38}$$

5. Compute the sum using the equivalent weights from last step
$$F(x_{kl}) = e^{i\phi(x_{kl}, y_0(B))} \sum_t L_t^A(x_{kl}) e^{-i\phi(x_t^A, y_0(B))} \delta_t^{AB} \tag{39}$$

5.4 Imaging With The Butterfly Algorithm

Suppose that $N_s = N_w = N$. The imaging algorithm given data $d(i, j)$ on $\gamma(s_j)$ is

$$\nu(x_{mn}) = -\frac{3200B\pi}{N^2 c_0^2} \sum_{l=1}^{N} \sum_{j=1}^{N} \frac{e^{-i(\omega_0 - \frac{B}{2} + q_j B)(\frac{2(x_{mn} \cdot \hat{\gamma}(s_l) - 10)}{c_0})} \overline{\hat{p}}(\omega_0 - \frac{B}{2} + q_j B) d(s_l, \omega_0 - \frac{B}{2} + q_j B)}{(\omega_0 - \frac{B}{2} + q_j B)|\hat{p}(\omega_0 - \frac{B}{2} + q_j B)|^2} \tag{40}$$

for all $x_{mn} = (m/N + 1/2N, n/N + 1/2N)$, $n, m = 0, 1, \ldots N - 1$. Setting $\phi(x_{mn}, s_l, q_j) = -(\omega_0 - \frac{B}{2} + q_j B)(\frac{2(x_{mn} \cdot \hat{\gamma}(s_l) - 10)}{c_0})$, $a(x_{mn}, s_l, q_j) = 1$, and $f(s_l, q_j) = \frac{\overline{\hat{p}}(\omega_0 - \frac{B}{2} + q_j B) d(s_l, \omega_0 - \frac{B}{2} + q_j B)}{(\omega_0 - \frac{B}{2} + q_j B)|\hat{p}(\omega_0 - \frac{B}{2} + q_j B)|^2}$ then the problem is exactly in the form of the butterfly algorithm. We will ignore the amplitude for now since it is constant. The algorithm is then

1. For $l = 0$, initialize
$$\delta_t^{AB} = e^{i\phi(x_0(A), y_t^B)} \sum_{y \in B} L_t^B(y) e^{i\phi(x_0(A), y)} f(y) \tag{41}$$
for all $A \in X_0$, $B \in Y_0$, and $t = 1, \ldots q$. There are $q^2 N^2$ weights to compute.

2. For $l = 1, \ldots, L/2$ compute the weights for all $A \in X_l$ and $B \in Y_l$ by

$$\delta_t^{AB} = e^{i\phi(x_0(A), y_t^B)} \sum_c \sum_{t_c} L_t^B(y_{t_c}^{B_c}) e^{i\phi(x_0(A), y_{t_c}^{B_c})} \delta_{t_c}^{A_p B_c} \tag{42}$$

3. Switch representation. Compute weights by

$$\delta_t^{AB} = \sum_s e^{i\phi(x_t^A, y_s^B)} \delta_s^{AB}. \tag{43}$$

Note that $l = L/2$ is seen twice in this algorithm.

4. For $l = L/2 + 1, \ldots L$, compute weights by

$$\delta_t^{AB} = \sum_c e^{i\phi(x_t^A, y_0(B_c))} \sum_{t_p} L_{t_p}^{A_p}(x_t^A) e^{-i\phi(x_{t_p}^{A_p}, y_0(B_c))} \delta_{t_p}^{A_p B_c} \tag{44}$$

5. Compute the sum using the equivalent weights from last step

$$\nu(x_{mn}) = e^{i\phi(x_{kl}, y_0(B))} \sum_t L_t^A(x_{kl}) e^{-i\phi(x_t^A, y_0(B))} \delta_t^{AB} \tag{45}$$

6. CONCLUSION

Simulation is still being run. To continue we must employ more accurate image reconstruction methods. We were able to produce high order scattering data, but image formation was a bottle neck. For this reason we are implementing the butterfly algorithm to image as well as scatter data. There is still much work to be done in this direction. We did not have to use the wave model at all for this approach. We could have used more exact computational electromagnetic schemes such as method of moments.[24] Moreover, one might also use a time-frequency method for data generation. This work mostly constituted a numerical experiment. The point of this work was to examine the use of high order terms to clear up images. Convergence of this method still needs to be proven/ disproved. In[25] the authors mention some work which tries to minimize the error created between the data and multiply scattered field. This should be compared to that work. We were successful in producing high order scattering in the frequency domain, and have some idea of when it is appropriate to use a truncated Neumann series to approximate the scattering function. At least one author[26] seems to have found a way around the Born approximation in a MIMO compressive sensing setting.

ACKNOWLEDGMENTS

The first author would like to thank our SAR imaging group and advisor Dr. Zhijun Qiao for insightful discussion during our seminar meetings. This work was partially supported by the U.S. Department of Education UTPA-MATH GAANN program (Grant Number P200A120256) and partially by the National Natural Science Foundation of China (Grant Numbers 61301187 and 61328103).

REFERENCES

[1] M. I. Skolnik, "Radar handbook," 1970.

[2] M. Soumekh, *Synthetic Aperture Radar Signal Processing with MATLAB Algorithms*, Wiley-Interscience publication, Wiley, 1999.

[3] M. Cheney and B. Borden, *Fundamentals of radar imaging*, CBMS-NSF regional conference series in applied mathematics, Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2009.

[4] L. Demanet, M. Ferrara, N. Maxwell, J. Poulson, and L. Ying, "A butterfly algorithm for synthetic aperture radar imaging," *SIAM Journal on Imaging Sciences* **5**(1), pp. 203–243, 2012.

[5] E. J. Candès, L. Demanet, and L. Ying, "A fast butterfly algorithm for the computation of fourier integral operators," *Multiscale Modeling & Simulation* **7**(4), pp. 1727–1750, 2009.
[6] B. D. Rigling, "Raider tracer: a matlab-based electromagnetic scattering simulator," in *Defense and Security Symposium*, pp. 65680E–65680E, International Society for Optics and Photonics, 2007.
[7] J. Poulson, L. Demanet, N. Maxwell, and L. Ying, "A parallel butterfly algorithm," *arXiv preprint arXiv:1305.4650* , 2013.
[8] A. Martinez and Z. Qiao, "Iteratively compensating for multiple scattering in sar imaging," in *SPIE Defense, Security, and Sensing*, pp. 874603–874603, International Society for Optics and Photonics, 2013.
[9] A. Martinez and Z. Qiao, "Born approximation, multiple scattering, and butterfly algorithm," in *SPIE Defense, Security, and Sensing*, pp. 90930B–90930B.
[10] A. Chai, M. Moscoso, and G. Papanicolaou, "Imaging strong localized scatterers with sparsity promoting optimization,"
[11] J. A. Stratton, *Electromagnetic theory*, vol. 33, John Wiley & Sons, 2007.
[12] J. X. Lopez and Z. Qiao, "Filtered back projection inversion of turntable isar data," pp. 805109–805109–9, 2011.
[13] D. J. Griffiths and R. College, *Introduction to electrodynamics*, vol. 3, prentice Hall Upper Saddle River, NJ, 1999.
[14] M. Born and E. Wolf, *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*, CUP Archive, 1999.
[15] M. Cheney, "A mathematical tutorial on synthetic aperture radar," *SIAM review* **43**(2), pp. 301–312, 2001.
[16] Y. Cao, J. F. Lopez, A. Martinez, and Z. Qiao, "A mathematical model for mimo imaging," pp. 839308–839308–11, 2012.
[17] G. Garza and Z. Qiao, "Resolution analysis of bistatic sar," pp. 80211V–80211V–6, 2011.
[18] T. Ray, Y. Cao, Z. Qiao, and G. Chen, "2d and 3d isar image reconstruction through filtered back projection," pp. 836107–836107–11, 2012.
[19] N. Pena, G. Garza, and Z. Qiao, "Filtered back projection type direct edge detection of real synthetic aperture radar images," pp. 83940N–83940N–7, 2012.
[20] A. N. K. an and S. V. Fomin, *Introductory Real Analysis*, Dover Publications, Mineola, N. Y., April 1975.
[21] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, Applied Mathematical Sciences, Springer, 1998.
[22] N. Suzuki, "On the convergence of neumann series in banach space," *Mathematische Annalen* **220**(2), pp. 143–146, 1976.
[23] G. Kaiser, *A friendly guide to wavelets*, Modern Birkhèauser classics, Birkhäuser Boston, 2011.
[24] D. Poljak and C. Y. Tham, *Integral equation techniques in transient electromagnetics*, Computational Mechanics, 2003.
[25] F. Cakoni, D. Colton, and P. Monk, *The linear sampling method in inverse electromagnetic scattering*, vol. 80, SIAM, 2011.
[26] A. C. Fannjiang, "Compressive inverse scattering I. High frequency SIMO/MISO and MIMO measurements."