A step-by-step implementation path for the three-dimensional acoustic scattering algorithm of Ganesh and Graham

John L. Weatherwax

September 19, 2006

This memo will discuss an incremental plan towards implementing the high-order scattering algorithm of Ganesh and Graham [4]. Specifically we will discuss and lay a plan towards an implementation of the solution of the Dirichlet sound soft scattering problem. We first present some background for this problem before presenting a detailed implementation outline.

The acoustic scattering from sound soft obsticle leads to the following problem. Find a radiating solution u to the Helmoltz equation

$$\Delta u + k^2 u = 0 \qquad \text{in} \qquad \mathbb{R}^3 \backslash \bar{D} \tag{1}$$

which satisfies the boundary condition

$$u = f \qquad \text{on} \qquad \partial D \tag{2}$$

A radiating solution is one that satisfies the Sommerfeld radiation condition expressed mathematically as

$$\lim_{r \to \infty} r\left(\frac{\partial u}{\partial r} - iku\right) = 0 \tag{3}$$

As described in [3] (Page 47) a method of characterizing this solution is as the following integral

$$u(x) = \int_{\partial D} \left\{ \frac{\partial \Phi(x,y)}{\partial \nu(y)} - i\eta \Phi(x,y) \right\} v(y) ds(y) \qquad x \in \mathbb{R}^3 \backslash \bar{D} \tag{4}$$

Where the function v is a solution to the following three dimensional integral equation

$$v + \mathcal{K}v - i\gamma \mathcal{S} v = 2f\,, \quad \text{on} \quad \partial D\,. \tag{5}$$

For the case of a fixed incident plane wave propagating in direction $\hat{d}$ and for the sound soft case considered here we have $f = -u^I(x) = -e^{ikx\cdot\hat{d}}$.

Now the single layer operator $\mathcal{S}$ and double layer operator $\mathcal{K}$ introduced above are given by

$$\mathcal{S}\psi(x) = 2\int_{\partial D} \Phi(x,y)\psi(y)ds(y) \tag{6}$$

$$\mathcal{K}\psi(x) = 2\int_{\partial D} \frac{\partial\Phi(x,y)}{\partial n(y)}\psi(y)ds(y)\,, \tag{7}$$

where the definition of the fundamental solution to Helmholtz equation in three dimensions $\Phi(x,y)$, is given by

$$\Phi(x,y) = \frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}\,. \tag{8}$$

For latter reference we derive the normal derivative of this fundamental solution

$$\begin{aligned}
\frac{\partial\Phi(x,y)}{\partial n(y)} &= \nabla_y\left(\frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}\right)\cdot\hat{n}(y) \\
&= \left(\frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}ik\nabla_y|x-y| - \frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|^2}\nabla_y|x-y|\right)\cdot\hat{n}(y) \\
&= \frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}\left(ik - \frac{1}{|x-y|}\right)\nabla_y|x-y|\cdot\hat{n}(y) \qquad (9)
\end{aligned}$$

In the above we require $\nabla_y|x-y|$ this can be computed by expanding the term $|x-y|$ in Cartesian coordinates and computing partial derivatives in a normal fashion. The result is

$$\nabla_y|x-y| = -\frac{x-y}{|x-y|} \tag{10}$$

These two subresults combine to give

$$\begin{aligned}
\frac{\partial\Phi(x,y)}{\partial n(y)} &= \frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}\left(ik - \frac{1}{|x-y|}\right)\frac{(-1)(x-y)}{|x-y|}\cdot\hat{n}(y) \\
&= \frac{-ik}{4\pi}\frac{e^{ik|x-y|}}{|x-y|^2}(x-y)\cdot\hat{n}(y) + \frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|^3}(x-y)\cdot\hat{n}(y) \qquad (11)
\end{aligned}$$

In the following, we transform the single and double layer potentials (defined above) into weakly singular and analytic kernals. To this end we will derive the real and imaginary parts of the single- and double-layer acoustic operators. Consider the following manipulations of the single layer potential

$$\begin{aligned}
\mathcal{S}\psi(x) &= 2\int_{\partial D}\frac{1}{4\pi}\frac{e^{ik|x-y|}}{|x-y|}\psi(y)ds(y) \\
&= \frac{1}{2\pi}\int_{\partial D}\left(\frac{\cos(k|x-y|)}{|x-y|} + i\frac{\sin(k|x-y|}{|x-y|}\right)\psi(y)ds(y) \\
&= \frac{1}{2\pi}\int_{\partial D}\left(\frac{S^c(x,y)}{|x-y|} + iS^s(x,y)\right)\psi(y)ds(y) \\
&= \frac{1}{2\pi}(\mathcal{S}^c\psi(x) + i\mathcal{S}^s\psi(x)) \qquad (12)
\end{aligned}$$

Where we have implicitly defined the real and imaginary single layer *kernals* $S^c(x,y)$ and $S^s(x,y)$ given by

$$S^c(x,y) = \cos(k|x-y|) \tag{13}$$

$$S^s(x,y) = \begin{cases} \frac{\sin(k|x-y|)}{|x-y|}, & x \neq y \\ k, & x = y \end{cases} \tag{14}$$

and the real and imaginary single layer *operators* $\mathcal{S}^s$ and $\mathcal{S}^c$ as

$$\mathcal{S}^c\psi(x) = \int_{\partial D} \frac{1}{|x-y|} S^c(x,y)\psi(y)ds(y) \tag{15}$$

$$\mathcal{S}^s\psi(x) = \int_{\partial D} S^s(x,y)\psi(y)ds(y)\,. \tag{16}$$

Next consider the following manipulations of the double layer potential

$$\begin{aligned}
\mathcal{K}\psi &= 2\int_{\partial D} \frac{\partial\Phi(x,y)}{\partial n(y)}\psi(y)ds(y) \\
&= 2\frac{-ik}{4\pi}\int_{\partial D} \frac{e^{ik|x-y|}}{|x-y|^2}(x-y)\cdot\hat{n}(y)\psi(y)ds(y) \\
&+ 2\frac{1}{4\pi}\int_{\partial D} \frac{e^{ik|x-y|}}{|x-y|^3}(x-y)\cdot\hat{n}(y)\psi(y)ds(y) \\
&= \frac{-ik}{2\pi}\int_{\partial D} \frac{\cos(k|x-y|)+i\sin(k|x-y|)}{|x-y|^2}(x-y)\cdot\hat{n}(y)\psi(y)ds(y) \\
&+ \frac{1}{2\pi}\int_{\partial D} \frac{\cos(k|x-y|)+i\sin(k|x-y|)}{|x-y|^3}(x-y)\cdot\hat{n}(y)\psi(y)ds(y) \\
&= \frac{k}{2\pi}\int_{\partial D} \frac{\sin(k|x-y|)-i\cos(k|x-y|)}{|x-y|^2}(x-y)\cdot\hat{n}(y)\psi(y)ds(y) \\
&+ \frac{1}{2\pi}\int_{\partial D} \frac{\cos(k|x-y|)+i\sin(k|x-y|)}{|x-y|^3}(x-y)\cdot\hat{n}(y)\psi(y)ds(y)
\end{aligned} \tag{17}$$

Grouping real and imaginary terms of the above expression (to derive the real and imaginary double layer kernels) gives

$$\mathcal{K}\psi = \frac{1}{2\pi}\int_{\partial D} \frac{1}{|x-y|}\{\frac{k\sin(k|x-y|)}{|x-y|} + \frac{\cos(k|x-y|)}{|x-y|^2}\}\,(x-y)\cdot\hat{n}(y)\,\psi(y)ds(y) \tag{18}$$

$$- \frac{i}{2\pi}\int_{\partial D} \{\frac{k\cos(k|x-y|)}{|x-y|^2} - \frac{\sin(k|x-y|)}{|x-y|^3}\}\,(x-y)\cdot\hat{n}(y)\,\psi(y)ds(y) \tag{19}$$

These expressions motivate the following real and imaginary double layer *kernel* definitions

$$K^c(x,y) = \frac{(x-y)\cdot\hat{n}(y)}{|x-y|^2}\{\cos(k|x-y|) + k\frac{\sin(k|x-y|)}{|x-y|}\} \tag{20}$$

$$K^s(x,y) = \frac{(x-y)\cdot\hat{n}(y)}{|x-y|^2}\{\frac{\sin(k|x-y|)}{|x-y|} - k\cos(k|x-y|)\} \tag{21}$$

The naming convention follows by analagy with Eulers formula:

$$e^{i\theta} = \cos(\theta) + i\sin(\theta)$$

in that that the real part is the "cosign" kernel and the imaginary part the "sign" kernel. In a similary way to above we can define two real and imaginary double layer $\mathcal{K}$ *operators* as

$$\mathcal{K}^c\psi(x) = \int_{\partial D} \frac{1}{|x-y|} K^c(x,y)\psi(y)ds(y) \tag{22}$$

$$\mathcal{K}^s\psi(x) = \int_{\partial D} K^s(x,y)\psi(y)ds(y)\,. \tag{23}$$

Since the real and imaginary single layer kernels appear in the definition of the real and imaginary double layer kernels we can write the later in terms of the former as

$$K^c(x,y) = \frac{(x-y)^t n(y)}{|x-y|^2} S^c(x,y) + k(x-y)^t n(y) S^s(x,y)\,, \tag{24}$$

$$K^s(x,y) = \frac{(x-y)^t n(y)}{|x-y|^2}[S^s(x,y) - kS^c(x,y)]\,. \tag{25}$$

Note that this agrees with Eq. 24 and Eq. 25. All of these manipulation motivates the following splitting the single- and double-layer acoustic operators as

$$\mathcal{S}\psi(x) = \frac{1}{2\pi}[\mathcal{S}^c\psi(x) + i\mathcal{S}^s\psi(x)] \tag{26}$$

$$\mathcal{K}\psi(x) = \frac{1}{2\pi}[\mathcal{K}^c\psi(x) + i\mathcal{K}^s\psi(x)] \tag{27}$$

Using the factorizations developed up to this point equation 5 can be written in component form as

$$v + \frac{1}{2\pi}[\mathcal{K}^c v + i\mathcal{K}^s v] - \frac{i\gamma}{2\pi}[\mathcal{S}^c v + i\mathcal{S}^s v] = 2f\,. \tag{28}$$

Writing out the integral operators above we have

$$v + \frac{1}{2\pi}\int_{\partial D} \frac{K^c(x,y)v(y)ds(y)}{|x-y|} + \frac{i}{2\pi}\int_{\partial D} K^s(x,y)v(y)ds(y) \tag{29}$$

$$- \frac{i\gamma}{2\pi}\int_{\partial D} \frac{S^c(x,y)v(y)ds(y)}{|x-y|} + \frac{\gamma}{2\pi}\int_{\partial D} S^s(x,y)ds(y) = 2f(x) \tag{30}$$

Grouping the factors into terms containing the singular term $(1/|x-y|)$ we get

$$v + \int_{\partial D} \frac{1}{2\pi}\left(K^c(x,y) - i\gamma S^c(x,y)\right)\frac{v(y)}{|x-y|}ds(y) \tag{31}$$

$$+ \int_{\partial D} \frac{1}{2\pi}\left(iK^s(x,y) + \gamma S^s(x,y)\right)v(y)ds(y) = 2f(x) \tag{32}$$

From Eq. 2.4 and Eq. 2.5 in the paper we see that

$$m(x,y) = \frac{1}{2\pi}\left(K^c(x,y) - i\gamma S^c(x,y)\right)\frac{1}{|x-y|} + \frac{1}{2\pi}\left(iK^s(x,y) + \gamma S^s(x,y)\right) \tag{33}$$

so separating $m(x,y)$ into a weakly singular part $m_1(x,y)$ and a nonsingular part $m_2(x,y)$ as

$$m(x,y) = \frac{m_1(x,y)}{|x-y|} + m_2(x,y) \tag{34}$$

we obtain

$$m_1(x,y) = \frac{1}{2\pi}\left(K^c(x,y) - i\gamma S^c(x,y)\right) \tag{35}$$

$$m_2(x,y) = \frac{1}{2\pi}\left(iK^s(x,y) + \gamma S^s(x,y)\right) \tag{36}$$

To follow the paper we must transform the integral equation Eq. 5 into the form given by Eq. (2.3) in that paper

$$w + \mathcal{M}w = [\alpha I + \mathcal{N}]h\,, \quad \text{on} \quad \partial D\,. \tag{37}$$

In our case the corresponance is

$$w = v \tag{38}$$

$$\mathcal{M} = \mathcal{K} - i\gamma\mathcal{S} \tag{39}$$

$$\alpha = 2 \tag{40}$$

$$\mathcal{N} = 0 \tag{41}$$

$$h = f(x) \tag{42}$$

With the operator $\mathcal{M}$ in the paper given by the kernel $m(x,y)$ as

$$\mathcal{M}\psi(x) = \int_{\partial D} m(x,y)\psi(x)ds(y) \tag{43}$$

The numerical algorithm finds a $W_n \in \mathbb{P}$, the space of all Harmonic functions on S^2 of order less than or equal to n, (note this space is of dimension $(n+1)^2$) such that

$$W_n = \sum_{l=0}^{n}\sum_{|j|\le l}\omega_{l,j}Y_{l,j} \tag{44}$$

To perform this we will extract the coefficients $\omega_{l,j}$ from a linear system

$$[I + M]\omega = \alpha I h \tag{45}$$

where $\omega = \omega_{l,j}$, and $I_{l',j',l,j} = \delta_{l',l}\delta_{j',j} = I_{(2(n+1)^2, 2(n+1)^2)}$, and $M_{l',j,'l,j}$ is constructed below. Note that the size of M is $m \times m$ or $2(n+1)^2 \times 2(n+1)^2$. This matrix should be initialized and filled with the coefficients of $M_{l',j',l,j}$ computed below in Eq. **??** below. The double to single index mapping mentioned in item **??** below will have to be used to place the element belonging in position $((l',j'),(l,j))$.

A Note On the Matlab Implementation of $M_{l',j,'l,j}$

As mentioned above we must compute the matrix $M_{l',j,'l,j}$ given by (this is taken almost verbatim from [4], the discussion below is novel)

$$M_{l',j,'l,j} = \sum_{q=1}^{m} \zeta_q \left(\sum_{q'=1}^{m'} \left[\alpha_{q'}^{n'} M_1(T_{\hat{x}_q}^{-1}\hat{n}, T_{\hat{x}_q}^{-1}\hat{z}_{q'}) + M_2(T_{\hat{x}_q}^{-1}\hat{n}, T_{\hat{x}_q}^{-1}\hat{z}_{q'}) \right] Y_{l,j}(T_{\hat{x}_q}^{-1}\hat{z}_{q'}) \overline{Y_{l',j'}(\hat{z}_q)} \right) . \tag{46}$$

A note on the Matlab implementation. To avoid the constructing an object with the largest dimensionality possible (dimension 4), we note that the term *inside* the bracket in equation 46 (before the summation) is only of dimension 3 (i.e. it depends on values for the "indexes" of q, q', and (l, j). These three dimensional arrays can be assembled in a for loop, multiplied by the Gaussian weights $\eta_{q'}$ (a scalar) and summed over the q' index. This results in a structure that only has two indices q and (l, j). In Matlab, this structure can be repmated producing a three dimensional array with indices given by q, (l, j), and (l', j') which is then multiplied by $\overline{Y_{l',j'}(\hat{z}_q)}$ and the Gaussian weigh ζ_q and summed in another for loop. This "judicious" use of for loops may in fact allow the code to produce results at a much higher discretization before memory becomes an issue since we now never have an object of four dimensions.

It is important to compute the far field solution once the linear system above has been solved. The far field soltuion to the exterior Dirchelet problem is given by the following integral expression

$$u_\infty(\hat{x}) = \frac{1}{4\pi} \int_{\partial D} \left(\frac{\partial e^{-ik\hat{x}\cdot y}}{\partial \hat{n}(y)} - i\gamma e^{-ik\hat{n}\cdot y} \right) v(y) ds(y) \tag{47}$$

as before we evaluate the partial derivative as follows

$$\begin{aligned} \frac{\partial e^{-ik\hat{x}\cdot y}}{\partial \hat{n}(y)} &= \nabla_y e^{-ik\hat{x}\cdot y} \cdot \hat{n}(y) && (48) \\ &= -ik\hat{x} e^{-ik\hat{x}\cdot y} \cdot \hat{n}(y) && (49) \\ &= -ike^{-ik\hat{x}\cdot y} \hat{x} \cdot \hat{n}(y) && (50) \end{aligned}$$

Thus the integral expression for $u_\infty(\hat{x})$ becomes

$$u_\infty(\hat{x}) = \frac{-i}{4\pi} \int_{\partial D} (k\hat{x} \cdot \hat{n}(y) + \gamma) e^{-ik\hat{x}\cdot y} v(y) ds(y) \tag{51}$$

When this is written in kernal form as suggested in the paper we have the explicit representation of the kernel as

$$m^f(\hat{x}, y) = \frac{-i}{4\pi}(k\hat{x} \cdot \hat{n}(y) + \gamma) e^{-ik\hat{x}\cdot y} \tag{52}$$

so

$$M_{\hat{x}}^f(\hat{y}) = m^f(\hat{x}, q(\hat{y}))J(\hat{y}) = \frac{-i}{4\pi}\{k\hat{x} \cdot \hat{n}(q(\hat{y})) + \gamma\} e^{-ik\hat{x}\cdot q(\hat{y})} J(\hat{y}) \tag{53}$$

The *numerical* calculation of the far field is given by (for the exterior Dirchlet problem) (with $N = 0$)

$$w_n(x) = \sum_{l=0}^{n} \sum_{|j|\le l} \omega_{lj} \tilde{M}_{lj}^m(x) \qquad x \in \tag{54}$$

with

$$\tilde{M}_{lj}^{m}(x) = \sum_{q=1}^{m} \xi_q \tilde{M}_x(\hat{x}_q) Y_{lj}(\hat{x}_q) \tag{55}$$

$$= \sum_{q=1}^{m} \xi_q \tilde{m}(x, q(\hat{x}_q)) J(\hat{x}_q) Y_{lj}(\hat{x}_q) \tag{56}$$

Giving in total

$$\omega_n(x) = \sum_{l=0}^{n} \sum_{|j|\le l} \omega_{lj} \sum_{q=1}^{m} \xi_q \tilde{m}(x, q(x_q)) J(\hat{x}_q) Y_{lj}(\hat{x}_q) \tag{57}$$

Once the solution to Eq. 45 is found for ω the assembly procedure for obtaining the scattered field $w_n(x)$ is as follows. For $w_n(x)$ we evaluate Eq. 3.40 in the Ganesh Graham paper with $\tilde{N} = 0$. This is

$$w_n(x) = \sum_{l=0}^{n} \sum_{|j|\le l} \omega_{lj} \tilde{M}_{lj}^{m}(x) \tag{58}$$

with

$$\tilde{M}_{lj}^{m}(x) = \sum_{q=1}^{m} \xi_q \tilde{M}_x(\hat{x}_q) Y_{lj}(\hat{x}_q) \tag{59}$$

$$= \sum_{q=1}^{m} \xi_q \tilde{m}(x, q(\hat{x}_q)) J(\hat{x}_q) Y_{lj}(\hat{x}_q) \tag{60}$$

Here we simply note that $\tilde{m}(x, y)$ in terms of $m(x, y)$ is given by

$$\tilde{m}(x, y) = \frac{1}{2} m(x, y)$$

At this time the manner in which I implementation the code makes it difficult to evaluate the near field pattern, since I expect every argument to be a *unit* vector. This will obviously not work to evaluate the near field using the above method. This can be seen by looking at $\tilde{m}(x, q(\hat{x}_q))$ which depends on a non unit vector x. One can however evaluate the far field using only unit vectors and this is explained next.

The far field is given by Eq. 3.42 of the Ganesh and Graham paper. Written here it is

$$w_{n,\infty}(x) = \sum_{l=0}^{n} \sum_{|j|\le l} \omega_{lj} \tilde{M}_{lj}^{m}(x) \tag{61}$$

The following is a road map of pieces of code that needs to be implemented to produce a working version of Ganesh and Grahams code. It is assume that if these are implemented in the order given that a fully working version of the code will be possible in which at a given level each component calls others that have been implemented above it. To follow this persciption the a specific set of pieces of code that need to be implemented are described in the following sections.

The far field from a sphere

$$u_\infty(\hat{x}) = \frac{i}{k}\sum_{n=0}^{\infty}(2n+1)\frac{j_n(kR)}{h_n^{(1)}(kR)}P_n(\cos(\theta)) \tag{62}$$

This is discussed on page 52 in [3]. This is implemented in the matlab routine

```
gen_ss_ball_0_ff_3d.m
```

and the results validated against the forward solver of Bruno and Kunyansky in [2].

1 Parameterized Mappings

Implement the mapping $q(\hat{x}; c, p)$ from $\hat{x} \in S^2$ to the parameterized objects, their Jacobian's $J(\hat{x}; c, p)$, and their normals $n(\hat{x}; c, p)$. Objects to consider include: ellipse, ogive, bean, NASA almond, pea, cone sphere, and a generalized radial spherical harmonic perturbation. The Parameters of these function calls are, a switch (called `type` below), a string representing the scattered object and a set of parameters specific to that object This is a vector called `param` (for "parameters") the length is determined by the specific scattering object.

In evaluating the normals for each object we note many of our scatters will have a parametric representation as (u, v) with $u = \theta$ and $v = \phi$. Thus we have the following

$$N(u, v) = \pm\frac{\partial r(u, v)}{\partial u} \times \frac{\partial r(u, v)}{\partial v} \tag{63}$$

Care must be exercised that the resulting normal is *outward* facing (CHECK ME!!!). For the evaluation of the Jacobian's we have the following identity [1] when the surface is parameterized by θ and ϕ

$$\int_{\partial D}\psi(x)ds(x) = \int_0^{2\pi}\int_0^{\pi}\psi(q(\theta,\phi))||\frac{\partial q(\theta,\phi)}{\partial\theta} \times \frac{\partial q(\theta,\phi)}{\partial\phi}||d\theta d\phi \tag{64}$$

The cross product explicitly contains the $\sin(\theta)$ term needed for the surface integral and so we rewrite this as follows

$$\int_{\partial D}\psi(x)ds(x) = \int_0^{2\pi}\int_0^{\pi}\psi(q(\theta,\phi))\left(\frac{1}{\sin(\theta)}||\frac{\partial q(\theta,\phi)}{\partial\theta} \times \frac{\partial q(\theta,\phi)}{\partial\phi}||\right)\sin(\theta)d\theta d\phi \tag{65}$$

From this we see that we need to define the Jacobian $J(\hat{y})$ to be

$$J(\theta,\phi) = \frac{1}{\sin(\theta)}||\frac{\partial q(\theta,\phi)}{\partial\theta} \times \frac{\partial q(\theta,\phi)}{\partial\phi}|| \tag{66}$$

Each object is called from a master driver called `q_scat`, `n_scat`, and `j_scat`. Each with a call sequence like the following (shown for `q_scat` as an example):

```
[ q ] = q_scat( theta, phi, type, param );
```

Here, `type` represents the object scattering happens from for example it could be the string "ellipse", and then `param` would represent the three parameters an ellipse depends on (a, b, and c), see below. These matlab functions are implemented such that they input vectors (column or row) of `theta` and `phi` and return their resulting x,y,z vectors stacked as rows). In the following subsections we record the mathematical representation of the objects considered.

1.1 Ellipse

An ellipse with axis *diameters* of a, b, c has the following representation:

$$x(\theta,\phi) = \frac{a}{2}\sin(\theta)\cos(\phi) \tag{67}$$

$$y(\theta,\phi) = \frac{b}{2}\sin(\theta)\sin(\phi) \tag{68}$$

$$z(\theta,\phi) = \frac{c}{2}\cos(\theta) \tag{69}$$

$$q(\theta,\phi) = (x(\theta,\phi), y(\theta,\phi), z(\theta,\phi)) \tag{70}$$

$$J(\theta,\phi) = \frac{1}{4}\left(b^2c^2\sin(\theta)^2\cos(\phi)^2 + a^2c^2\sin(\theta)^2\sin(\phi)^2 + a^2b^2\cos(\theta)^2\right)^{1/2} \tag{71}$$

$$\hat{n}(\theta,\phi) = \frac{(bc\sin(\theta)\cos(\phi), ac\sin(\theta)\sin(\phi), ab\cos(\theta))}{\sqrt{(b^2c^2\sin(\theta)^2\cos(\phi)^2 + a^2c^2\sin(\theta)^2\sin(\phi)^2 + a^2b^2\cos(\theta)^2)}} \tag{72}$$

The specific function calls for the ellipse are:

- `q_ellipse(theta,phi,param)`
- `n_ellipse(theta,phi,param)`
- `j_ellipse(theta,phi,param)`

2 The Fundamental Scattering Kernels & Derivatives

Implement the fundamental scattering kernels $S^c(x,y)$, $S^s(x,y)$, $K^c(x,y)$, $K^s(x,y)$ defined by Eqs. 13, 14, 24 and 25 with switches to evaluate these expressions accurately when $x \approx y$. These functions depend on the wave number k, the object normal. The specific function calls are given by:

- `Sc = cmpt_S_sup_c( theta_x, phi_x, theta_y, phi_y, k, choice, param )`
- `Ss = cmpt_S_sup_s( theta_x, phi_x, theta_y, phi_y, k, choice, param )`
- `Kc = cmpt_K_sup_c( theta_x, phi_x, theta_y, phi_y, k, choice, param )`
- `Ks = cmpt_K_sup_s( theta_x, phi_x, theta_y, phi_y, k, choice, param )`

These routines are vectorized so that they input column vectors for `theta` and `phi` and return a matrix as an output. For example the i, j element in the matrix output `Sc` from `cmpt_S_sup_c` represents the scalar S^c evaluated at $\hat{x}_i = p(\theta_i, \phi_i)$ and $\hat{y}_j = p(\theta_j, \phi_j)$. Here $p(\theta, \phi)$ is defined as a point on S^2 (given in Cartesian by)

$$p(\theta, \phi) = (\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta)) \tag{73}$$

The specific Taylor expansions used to evaluate S^c, S^s, K^c, and K^s accurately when $x \approx y$ are defined in Appendix A.

Implement $m_1(x, y)$ and $m_2(x, y)$, defined by Eqs. 35 and 36 above. These are implemented in the following Matlab routine

- `[m1, m2] =`
  `cmpt_litMs( theta_x, phi_x, theta_y, phi_y, k, gamma, choice, param)`

Implement the following functions

$$\begin{aligned} R(\hat{x}, \hat{y}) &= \frac{|\hat{x} - \hat{y}|}{|q(\hat{x}) - q(\hat{y})|} && (74) \\ M_1(\hat{x}, \hat{y}) &= R(\hat{x}, \hat{y}) m_1(q(\hat{x}), q(\hat{y})) J(\hat{y}) && (75) \\ M_2(\hat{x}, \hat{y}) &= m_2(q(\hat{x}), q(\hat{y})) J(\hat{y}) && (76) \end{aligned}$$

These are implemented in the following matlab functions

- `[ R ] = bigR( theta_x, phi_x, theta_y, phi_y, choice, param )`
- `[M1, M2] =`
  `cmpt_bigMs( theta_x, phi_x, theta_y, phi_y, k, gamma, choice, param)`

3 The Right Hand Side

Implement the spherical fourier transform of the Dirchlet boundary conditons $f(x) = -2e^{ikx\cdot\hat{d}}$ This is done in the routine `cmpt_RHS.m` with the following call

- `[ rhs ] = cmpt_RHS( n, k, dhat, choice, param );`

This routine has been checked and found correct using the following identity (from Page 62 in [3]) which gives an analytic representation of the spherical Fourier coefficients of the incoming plane wave.

$$e^{ikx\cdot d} = 4\pi \sum_{n=0}^{\infty} \sum_{m=-n}^{n} i^n j_n(k|x|) Y_n^m(d)^* Y_n^m(\hat{x}) \tag{77}$$

This routine is tested and compared in the routine `test_cmpt_RHS.m`.

4 Utilities

Implement some utilities functions from going from spherical coordinates to Cartesian

$$\hat{z} = p(\theta, \phi) = (\sin(\theta)\cos(\phi), \sin(\theta)\cos(\phi), \cos(\theta)) \tag{78}$$

This is implemented in the Matlab function `lit_p`.

4.1 Quadrature Points for S^2

Implement the discretization of the unit sphere S^2 with two indices. This requires the following sub-elements

- Numerical discretization points for the θ (elevation) integration over the unit sphere (abscissa and weights): θ_s and ν_s for $s = 1, 2, \ldots n+1$. Here ν_s are the so called Gauss-Legendre weights and $\theta_s = \cos^{-1}(z_s)$ with z_s the zeros of the Legendre polynomials of degree $n+1$.

  These can be comuted individually in the Matlab function `[theta_s, nu_s] = gauss(n+1)` which returns the $n+1$ sorted abscissa and weights in the vectors `theta_s` and `nu_s` respectively.

- Numerical discretization points for the ϕ (azimuth) integration (abscissa and weights): ϕ_r and μ_r. Defined as follows

$$\phi_r = \frac{(r-1)\pi}{n+1} \quad \text{for} \quad r = 1, 2, \ldots, 2n+2 \tag{79}$$

$$\mu_r = \frac{\pi}{n+1} \quad \text{for} \quad r = 1, 2, \ldots, 2n+2 \tag{80}$$

- The combined weights from both the azimuthal and the elevation discretization (there will be $m = 2(n+1)^2$ elements in each vector)
  - $\zeta_q = \xi_q = \mu_{r(q)}\nu_{s(q)}$ for $q = 1, 2, \ldots, m$

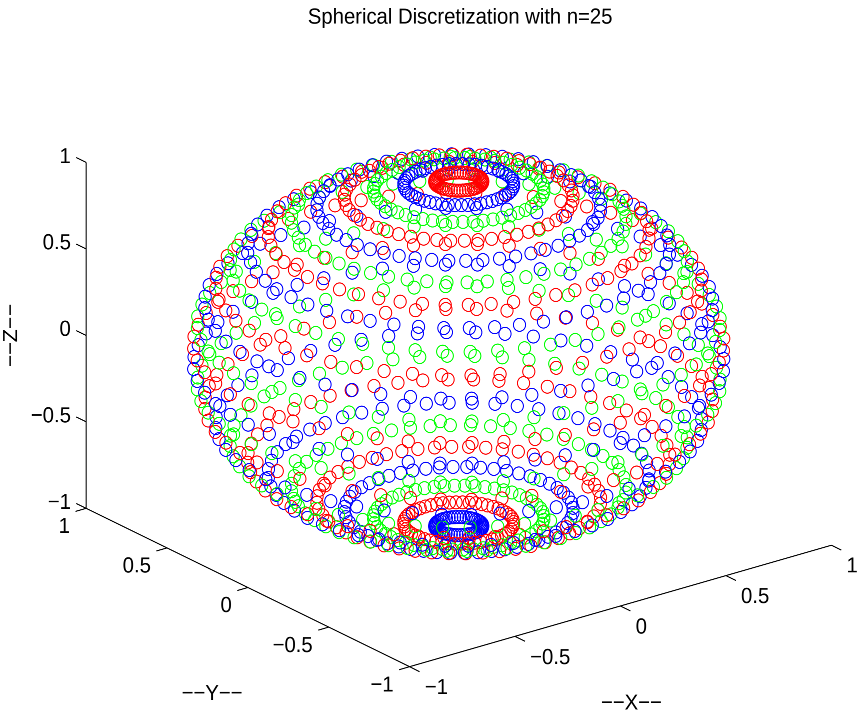

Figure 1: An example spherical discretization with $n = 25$.

Here $r(q)$ and $s(q)$ are the mappings developed in **??** I should mention that the mapping from double to single indices is done such that the "slow" index is in θ and the "fast" index is in ϕ.

- All of these are output from the following Matlab function:
  - `[theta_s, nu_s, phi_r, mu_r, xi_q] = sphere_disc(n+1)`
- The q ordered points themselves (for a given n) are output from the following Matlab function
  - `[ xyzPts, tpPts ] = all_zq_hat( n );`

  With `xyzPts` a matrix of Cartesian points and `tpPts` a matrix of the corresponding (θ, ϕ) spherical points, both row stacked matrices. We note that in this output format the points are ordered such that the discretization of ϕ_r is the *fast* index (with $2(n+1)$ points) and the discretization of θ_s is the *slow* (with $n+1$ points). When ploted with the ϕ discretzation points all in one color one obtains Figure 1.

4.2 Transformation Matrices

Implement the transformation matrices $T_{\hat{x}} = P(\phi)Q(-\theta)P(-\phi)$ and its inverse in terms of spherical inputs (θ, ϕ) i.e. $T_{\theta,\phi}$. Do the multiplications and inversion exactly. These are implemented in the following matlab functions

- `[ T ] = bigT( theta, phi );`
- `[ TInv ] = bigTInv( theta, phi );`

These are vectorized in the sense that multiple thetas and phis will produce block column sets of matrices. See Subsection 4.3 for an example of how the outputs above are formulated.

4.3 The Rotated Coordinate System

Implement $y_{r,s}^{r',s'}$ where

$$y_{r,s}^{r',s'} = T^{-1}_{p(\theta_s,\phi_r)} p(\theta_{s'}, \phi_{r'}) \tag{81}$$

The Matlab implementation can be visuallized with the following block matrix expressions

$$\begin{aligned} T^{-1}_{p(\theta_s,\phi_r)} p(\theta_{s'}, \phi_{r'}) &= \begin{bmatrix} T_1^{-1} \\ T_2^{-1} \\ T_3^{-1} \\ \vdots \\ T_q^{-1} \end{bmatrix} [p_{1'}\, p_{2'}\, p_{3'}\, \cdots\, p_{q'}] \\ &= \begin{bmatrix} T_1^{-1}p_{1'} & T_1^{-1}p_{2'} & T_1^{-1}p_{3'} & \cdots & T_1^{-1}p_{q'} \\ T_2^{-1}p_{1'} & T_2^{-1}p_{2'} & T_2^{-1}p_{3'} & \cdots & T_2^{-1}p_{q'} \\ \vdots & \vdots & \vdots & & \vdots \\ T_q^{-1}p_{1'} & T_q^{-1}p_{2'} & T_q^{-1}p_{3'} & \cdots & T_q^{-1}p_{q'} \end{bmatrix} \end{aligned} \tag{82}$$

The first block matrix (with the T^{-1}) is the output from `bigTInv` and expressions like p_3 is short hand for $p(\theta_{q=3}, \phi_{q=3})$. In general there are more primed variables than unprimed variables. In the Ganash and Gram paper $n' = 2n$ and that will be observed here too. This is implemented in the following function

- `[ yqqp_xyz_col, yqqp_tp_col ] = all_yqqp_hat( n, nprime );`

Where `yqqp_xyz_col` is a column matrix of the Cartesian components representing the object $\hat{y}_{r,s}^{r',s'}$. As such the indices run from fastest to slowest in the order r (with $2(n+1)$ elements), s (with $n+1$ elements), r' with $(2(n'+1)$ elements), and s' (with $n'+1$ elements). In addtion, `yqqp_tp_col` is the corresponding polar (θ, ϕ) representation.

1. Implement the $\alpha_{q'}^{n'}$ coefficients

   $$\alpha_{q'}^{n'} = \sum_{l=0}^{n'} P_l(\hat{n}.\hat{z}_{q'}) \tag{83}$$

   where $P_l(.)$ is the Legendre polynomial of degree l, $\hat{n} = (0, 0, 1)$, and $\hat{z}_{q'} = p(\theta_{s(q')}, \phi_{r(q')})$

2. Implement $E^1_{s,r,s',\tilde{j}}$ and $E^2_{s,r,s',\tilde{j}}$

$$E^1_{s,r,s',\tilde{j}} = \sum_{r=0}^{2n'+1} \xi_{r'} M_1(\hat{x}_{r,s}, y^{r',s'}_{r,s}) e^{i\tilde{j}\phi_{r'}} \quad (84)$$
$$E^2_{s,r,s',\tilde{j}} = \sum_{r=0}^{2n'+1} \xi_{r'} M_2(\hat{x}_{r,s}, y^{r',s'}_{r,s}) e^{i\tilde{j}\phi_{r'}} \quad (85)$$

Here $\xi_{r'}$ is the ϕ discretization weights and equals $\frac{\pi}{n'+1}$ and $\Phi_{r'}$ are the discretization points and equals $\frac{(r'-1)\pi}{n'+1}$

3. Implement the following chain of expressions each of which depends on the previous one

$$C^j_l = (-1)^{(j+|j|)/2}\sqrt{\frac{2l+1}{4\pi}\frac{(l-|j|)!}{(l+|j|)!}} \quad (86)$$
$$P^{(a,b)}_q(0) = 2^{-q}\sum_{t=0}^{q}(-1)^t\binom{q+a}{q-t}\binom{q+b}{t} \quad (87)$$
$$d^l_{\tilde{j},j}(\pi/2) = 2^{\tilde{j}}\left[\frac{(l+\tilde{j})!(l-\tilde{j})!}{(l+j)!(l-j)!}\right]^{1/2} P^{(j-\tilde{j},-j-\tilde{j})}_{l+\tilde{j}}(0) \quad (88)$$
$$F_{s,l,\tilde{j},j} = e^{i(j-\tilde{j})\frac{\pi}{2}}\sum_{|m|\le l} d^l_{\tilde{j},m}(\pi/2)d^l_{j,m}(\pi/2)e^{im\theta_s} \quad (89)$$
$$D_{s,r,l,\tilde{j}} = \sum_{s'=1}^{n'+1}\eta_{s'}[\alpha^{n'}_{s'}E^1_{s,r,s',\tilde{j}} + E^2_{s,r,s',\tilde{j}}]c^{\tilde{j}}_l P^{|\tilde{j}|}_l(\cos(\theta_j)) \quad (90)$$
$$C_{s,r,l,j} = \sum_{|\tilde{j}|\le l} D_{s,r,l,\tilde{j}}e^{i(j-\tilde{j})\phi_r}F_{s,l,\tilde{j},j} \quad (91)$$
$$B_{s,j',l,j} = \sum_{r=0}^{2n+1} C_{s,r,l,j}\mu_r e^{-ij'\phi_r} \quad (92)$$
$$M_{l',j',l,j} = \sum_{s=1}^{n+1} B_{s,j',l,j}\nu_s c^{j'}_{l'}P^{|j'|}_{l'}(\cos(\theta_s)) \quad (93)$$

These elements of $M_{l',j',l,j}$ will need to be placed in the entire matrix M.

A Evaluations of the scattering kernels when $|x-y|\sim 0$

A numerical issue may arise in evaluating the scattering kernels in Eq. 74 if the points x and y are too close together. In fact is is impossible for them to be *equal* in the algorithm above (see below for a proof) but in the case that they are very close together we present analytic expansions valid when $|x-y|\sim 0$ below.

A.1 Proof that $x \neq y$

In Eqs. 84 a singularity problem exists if

$$y_{r,s}^{r',s'} = \hat{x}_{r,s}\,. \tag{94}$$

Using the definitions of the above expression we see that this is equivalent to

$$T^{-1}_{p(\theta_s,\phi_r)} p(\theta_{s'},\phi_{r'}) = p(\theta_s,\phi_r) \tag{95}$$

or

$$p(\theta_{s'},\phi_{r'}) = T_{p(\theta_s,\phi_r)} p(\theta_s,\phi_r) = \hat{n} \tag{96}$$

Since in fact $p(\theta_{s'},\phi_{r'}) = \hat{n}$ *only* if $\theta_{s'} = 0$, or equivalently $\cos^{-1}(z_{s'}) = 0$, and $z_{s'} = \frac{\pm\pi}{2}$, we see that in fact we never encounter a zero of this expression. For accurate computational purposes however we express the Taylor series expansions of the scattering kernels below.

A.2 Small Argument Evaluation of S^c

As this expression is analytic for all x and y no specific checks need to be performed

A.3 Small Argument Evaluation of S^s

To accurately evaluate the expressions $S^s(x,y;k,c,p)$, $K^c(x,y;k,c,p)$, and $K^s(x,y;k,c,p)$ we Taylor expand in terms of $|x-y|$. For $S^s(x,y)$ this is

$$\frac{\sin(k|x-y|)}{|x-y|} = \frac{1}{|x-y|}\sum_{j\geq 0}\frac{(-1)^j k^{2j+1}|x-y|^{2j+1}}{(2j+1)!} = \sum_{j\geq 0}\frac{(-1)^j k^{2j+1}|x-y|^{2j}}{(2j+1)!} \tag{97}$$

$$= k - \frac{k^3|x-y|^2}{6} + \frac{k^5|x-y|^4}{5!} + O(|x-y|^6) \tag{98}$$

A.4 Small Argument Evaluation of K^c

A.5 Small Argument Evaluation of K^s

References

[1] T. Apostol. *Calculus, Vol. II.* Wiley, 1969.

[2] O. P. Bruno and L. A. Kunyansky. A fast, high-order algorithm for the solution of surface scattering problems: basic implementation, tests, and applications. *J. Comput. Phys.*, 169(1):80–110, 2001.

[3] D. Colton and R. Kress. *Inverse Acoustic and Electromagnetic Scattering Theory*. Springer-Verlag, 1998.

[4] M. Ganesh and I. G. Graham. A high-order algorithm for obstacle scattering in three dimensions. *J. Comput. Phys.*, 198(1):211–242, 2004.