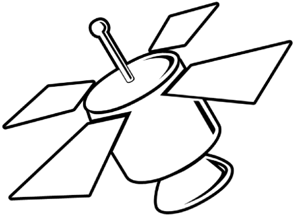
Data Reduction for Modelling Satellite Radar Cross Sections

Daniel Topa
daniel.topa@hii.com

Huntington Ingalls Industries
Mission Technologies

December 19, 2024

Models of Radar Cross Sections for Satellites



Spherical Harmonics Expansion

$$f(r,\theta,\phi) \approx a_{0,0}Y_0^0 + a_{1,-1}Y_1^{-1} + a_{1,0}Y_1^0 + a_{1,1}Y_1^1 + \ldots$$

where

$$Y_n^m(\theta,\phi) = \sqrt{\frac{(2n+1)(n-m)!}{4\pi(n+m)!}}P_n^m(\cos\theta)e^{im\phi}$$

Overview

1. Radar Cross Section Simulation
2. Preparing for Mercury MoM
3. Outputs and File Types
4. Custom Software Tools
5. Backup Slides

Input and Final Output

Input: *.obj File

```
# Created with the Wolfram Language : www.wolfram.com

mtllib sp-006.mtl

# 6 vertex positions
v  0 0 -1
v  0 -1 0
v  -1 0 0
v  1 0 0
v  0 0 1
v  0 1 0
```

Output: Amplitude Vector

$$a_{0,0} = 1.345 \pm 0.015$$
$$a_{1,-1} = 1.098 \pm 0.017$$
$$a_{1,0} = 1.210 \pm 0.017$$
$$a_{1,1} = 0.945 \pm 0.017$$
$$a_{2,-2} = 0.512 \pm 0.018$$
$$a_{2,-1} = 0.732 \pm 0.017$$
$$a_{2,0} = 1.110 \pm 0.017$$
$$a_{2,1} = 0.885 \pm 0.016$$
$$a_{2,2} = 0.658 \pm 0.017$$

Beginning to End I

Data Creation and Analysis Steps

1. Start with CAD model: `*.stl`
2. Create `*.obj`
3. Create `*.facet`
   - Create `*.geo` (geometry)
   - Create `*.lib` (EM properties)
4. Generate `*.4112.txt`
5. Harvest θ, ϕ fields
6. Create `*.rcs`

Beginning to End II

7. **Create amplitudes** a

Big Picture: CAD to RCS Table

We discuss Step 1 $\Rightarrow$ Step 2

1. **Start with CAD model:** `*.stl`
2. **Finish with table** `*.rcs`
3. Resolved to approximate $f\left(r, \theta, \phi\right)$

Biggest Challenge

Going from a CAD model to a model of different electromagnetic materials.

Software Components

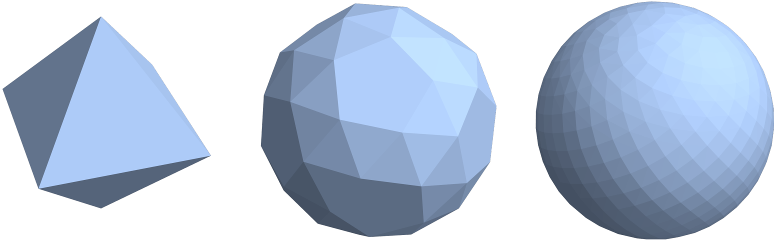
1. **converter:** `*.obj` ⇒ `*.facet`
2. **mesh analysis & repair:** `*.obj` ⇒ `*.facet`
3. **extractor: pull backscatter from** `*.4112.txt`
4. **converter: backscatter to** `*.rcs`
5. **calculator:** `*.rcs` **to spherical harmonic amplitudes**

CAD file (*.stl) to Mesh Structure File (*.obj)

Many Tools For Converting *.stl to *.obj

1. Blender
2. FreeCAD
3. OpenSCAD
4. SolidWorks
5. Tinkercad
6. MeshConvert.com
7. Online 3D Model Converter
8. others

Seeing the `*.obj` File



6 vertices | 60 vertices | 600 vertices

Decadal Improvement in Resolution:
Number of vertices increases $\times 10$

sp-006.obj

```
# Created with the Wolfram Language : www.wolfram.com

mtllib sp-006.mtl

# 6 vertex positions
v  0 0 -1
v  0 -1 0
v  -1 0 0
v  1 0 0
v  0 0 1
v  0 1 0

# 0 UV coordinates

# 0 vertex normals
```

```
# Mesh '' with 8 faces
usemtl Material_1
f  1/ 2/ 3/
f  2/ 1/ 4/
f  2/ 5/ 3/
f  5/ 2/ 4/
f  1/ 6/ 4/
f  6/ 1/ 3/
f  6/ 5/ 4/
f  5/ 6/ 3/
```

Components of the *.obj

1. **Headers and Comments (#):**
   - **Used for metadata or human-readable information.**
   - **Example:** `# Created with Wolfram Language`.
2. **Vertex Positions (`v`):**
   - **Specifies 3D coordinates for vertices.**
   - **Example:** `v 0 0 -1`.
3. **Faces (`f`):**
   - **Defines polygons by referencing vertex indices.**
   - **Example:** `f 1/2/3`.

Components of the `*.obj`

4. **Material Library Reference (`mtllib`):**
   - **External `*.mtl` file that specifies visual materials for rendering (e.g., color, shading)**
   - **Example: `sp-006.mtl`.**
   - **Important Note: This `*.mtl` file is not related to the electromagnetic materials library in CAD models, which defines physical properties like permittivity, permeability, or conductivity.**

Components of the `*.obj`

1. **Headers and Comments (#):**
   - **Used for metadata or human-readable information.**
   - **Example:** `# Created with Wolfram Language`.
2. **Vertex Positions (`v`):**
   - **Specifies 3D coordinates for vertices.**
   - **Example:** `v 0 0 -1`.
3. **Faces (`f`):**
   - **Defines polygons by referencing vertex indices.**
   - **Example:** `f 1/2/3`.

*.obj: Vertices and Plaquettes

Boo

*.geo: Material Properties

Boo

Sprawling Toolset: Languages

1. **Fortran**
2. **Python**
3. **Mathematica**
4. **Shell scripts**

Sprawling Toolset: Purposes

1. **Automation**
2. **Conversions**
3. **Data Analysis**
4. **Diagnostics**

Sprawling Toolset: Design

1. Object oriented
2. Emphasis on error tracking
3. Some crude
4. Some refined

Python Tool for *.obj to *.facet I

```python
from datetime import datetime
from Facet import Facet
from Vertex import Vertex

import io
import os
import sys

DEFAULT_ELEMENT_DESCRIPTION    = '3 {} 0 0 0 0 0'
DEFAULT_FILE_EXTENSION_OUTPUT = '.facet'
DEFAULT_PART_COUNT            = '1'
DEFAULT_PART_MIRROR           = '0'
DEFAULT_PART_NAME             = '<PTW MeshModel>'
DEFAULT_SUBPART_COUNT         = '1'
DEFAULT_SUBPART_NAME          = '<PTW MeshSheet>'

argumentCount = len(sys.argv)

# output argument-wise
if argumentCount == 2:
    objectFileName = sys.argv[1]
    outputFileName = os.path.splitext(objectFileName)[0] +
         DEFAULT_FILE_EXTENSION_OUTPUT
elif argumentCount == 3:
```

Python Tool for *.obj to *.facet II

```
    objectFileName = sys.argv[1]
    outputFileName = sys.argv[2]
else:
    sys.stderr.write('Usage:␣python␣Obj2Facet.py␣<input-obj-file-name>␣[<output-facet-file-name>]\n')
    sys.exit()

facetCount  = 0
facetLines  = ""
vertexCount = 0
vertexLines = ""
with io.open(objectFileName, 'r', encoding='utf-8') as objectFile:
    line = objectFile.readline()
    lineNumber = 1
    while line:
        tokens = line.strip().split('␣')
        if len(tokens) == 4:
            type = tokens[0]

            if type.lower() == 'f':
                facetLines += '␣'.join(tokens[1:4])
                facetLines += '␣0'
                facetLines += '\n'
                facetCount += 1
```

Python Tool for `*.obj` to `*.facet` III

```python

            elif type.lower() == 'v':
                vertexLines += ' '.join(tokens[1:4])
                vertexLines += '\n'
                vertexCount += 1

        line        = objectFile.readline()
        lineNumber += 1

    objectFile.close()

with io.open(outputFileName, 'w', encoding='utf-8') as outputFile:
    outputFile.write('FACET FILE V3.4 ')
    outputFile.write(datetime.today().strftime('%d-%b-%Y %H:%M:%S'))
    outputFile.write('\n')

    outputFile.write(DEFAULT_PART_COUNT)
    outputFile.write('\n')
    outputFile.write(DEFAULT_PART_NAME)
    outputFile.write('\n')
    outputFile.write(DEFAULT_PART_MIRROR)
    outputFile.write('\n')

    outputFile.write(str(vertexCount))
```

Python Tool for *.obj to *.facet IV

```
    outputFile.write('\n')
    outputFile.write(vertexLines)

    outputFile.write(DEFAULT_SUBPART_COUNT)
    outputFile.write('\n')
    outputFile.write(DEFAULT_SUBPART_NAME)
    outputFile.write('\n')

    outputFile.write(DEFAULT_ELEMENT_DESCRIPTION.format(facetCount))
    outputFile.write('\n')
    outputFile.write(facetLines)

    outputFile.close()
```

Major Fortran Tools I

1. aeneas.f08
2. createFacetFile.f08
3. esjufjoll.08
4. facimusFacet.f08
5. facet-maker.f08
6. geo-writer.f08
7. harvestRCSfromMoM.f08
8. json-writer.f08

Major Fortran Tools II

9. `gather.f08`
10. `revised-reader.f08`
11. `shaeffer.f08`
12. `sigma.f08`

esjufjollf.f08 Execution I

Listing 1: Excerpt from esjufjoll.f08

```
! dantopa:hot/eriksjokull % ./eriksjokull
    (master)fortran-alpha
! List of 10 input files in ../elevations/list-of-files.txt:
!  1. PTW-elev-0p045.4112.txt.
!  2. PTW-elev-0p050.4112.txt.
!  3. PTW-elev-0p055.4112.txt.
!  4. PTW-elev-0p060.4112.txt.
!  5. PTW-elev-0p065.4112.txt.
!  6. PTW-elev-0p070.4112.txt.
!  7. PTW-elev-0p075.4112.txt.
!  8. PTW-elev-0p080.4112.txt.
!  9. PTW-elev-0p085.4112.txt.
! 10. PTW-elev-0p090.4112.txt.
!
! * Properties of azimuth   mesh:
! * minimum value = -180.000000, maximum value = 179.000000, length = 359.000000
! * number of samples = 360, interval size = 1.00000000
!
! #  #  Dimensions for RCS data containers  #  #
!
```

esjufjollf.f08 Execution II

```
! # Expected dimensions:
! # Number of radar frequencies scanned by MoM:   28
! # Number of azimuth   angles scanned by MoM:  360
! # Number of elevation angles scanned manually: 10
!
! # Container for each MoM 4112.txt file: rcs_table_rank_2
! # Free angle dimension = 360 indices run from 1 to 360
! # Frequency dimension  = 28 indices run from 1 to 28
!
! # Container for all MoM 4112.txt files: rcs_table_rank_3
! # Free angle dimension  = 360 indices run from 1 to 360
! # Frequency dimension   = 28 indices run from 1 to 28
! # Fixed angle dimension = 10 indices run from 1 to 10
!
! Analyzing file 001/010: 'PTW-elev-0p045.4112.txt', elevation = 45.
! Analyzing file 002/010: 'PTW-elev-0p050.4112.txt', elevation = 40.
! Analyzing file 003/010: 'PTW-elev-0p055.4112.txt', elevation = 35.
! Analyzing file 004/010: 'PTW-elev-0p060.4112.txt', elevation = 30.
! Analyzing file 005/010: 'PTW-elev-0p065.4112.txt', elevation = 25.
! Analyzing file 006/010: 'PTW-elev-0p070.4112.txt', elevation = 20.
! Analyzing file 007/010: 'PTW-elev-0p075.4112.txt', elevation = 15.
! Analyzing file 008/010: 'PTW-elev-0p080.4112.txt', elevation = 10.
! Analyzing file 009/010: 'PTW-elev-0p085.4112.txt', elevation = 5.
! Analyzing file 010/010: 'PTW-elev-0p090.4112.txt', elevation = 0.
```

`esjufjollf.f08` Execution III

facet-maker.f08 Execution I

Listing 2: Excerpt from `facet-maker.f08`

```
! dantopa:rcs/facet % ./facet-maker B20-standard-1m
    (master)fortran-alpha
!
! target directory: ./data/
! input  file: ./data/B20-standard-1m.obj
! output file: ./data/B20-standard-1m.facet
!
! Opening ./data/B20-standard-1m.obj to read data lists.
!
! Opening ./data/B20-standard-1m.facet for writing.
!
! completed at 2020-04-08 16:05:04
```

gather.f08 Execution I

Listing 3: Excerpt from gather.f08

```
! dantopa:3d/xylorimba % ./gather (master)fortran-alpha
! 1. file = PTW-elev-0n179.4112.txt, -179, elevation angle = 0.
! 2. file = PTW-elev-0n178.4112.txt, -178, elevation angle = 0.
! 3. file = PTW-elev-0n177.4112.txt, -177, elevation angle = 0.

! 358. file = PTW-elev-0p178.4112.txt, 178, elevation angle = 0.
! 359. file = PTW-elev-0p179.4112.txt, 179, elevation angle = 0.
! 360. file = PTW-elev-0p180.4112.txt, 180, elevation angle = 0.
!
! completed at 2020-05-12 21:49:23
```

revised-reader.f08 Excerpts I

Listing 4: Excerpt from `revised-reader.f08` (lines 39-45)

```
! dantopa:readers/cleat % ./revised-reader
     (master)fortran-alpha
! B-20A.4112.txt has 14843 lines
! B-20A.4112.txt has 28 frequencies.
! 3.00000000
! '...Finished' found at line 854
! 4.00000000
! '...Finished' found at line 1368
```

Listing 5: Excerpt from `revised-reader.f08` (lines 123-127)

```
! 26. nu = 28.0000000, start = 13344, stop = 13703, terms = 360
! 27. nu = 29.0000000, start = 13858, stop = 14217, terms = 360
! 28. nu = 30.0000000, start = 14372, stop = 14731, terms = 360
!
! completed at 2020-05-09 15:30:20
```

sigma.f08 Overview I

Listing 6: Excerpt from sigma.f08

```
! nb: /Users/dantopa/Mathematica_files/nb/ert/mercury/snake/fortran-01.nb
program rcs

! Read the Mercury Methods of Moments processed into a table of mean total RCS
    values
! Use the method of least squares to find
!   RCS ( yaw angle )           radar frequency fixed
!   RCS ( radar frequency )     yaw angle fixed

! Daniel Topa, ERT Corp
! COVID-19 Prisoner

! Class structure
!   RCStable: table of mean total RCS ( nu, alpha )
!   LinearSystem: Sytem Matrix A, data vector b
!        flavors: Fourier, monomial
!        tied to RCStable
!   LeastSquaresResults:
!        amplitudes
!        errors
!        residual error vector
```

sigma.f08 Overview II

```
!          tied to linear system
```

Mathematica Commands I

```
(* Generate a mesh for a convex hull of points on a sphere *)
mesh = ConvexHullMesh[SpherePoints[60]] // Region;

(* Extract vertex coordinates from the mesh *)
vlist = MeshCoordinates[mesh];

(* Label each point in the mesh *)
labeledPoints = MapIndexed[
    Text[Style[ToString[#2[[1]]], Bold, Black], #1] &, vlist
];
```

Facet (Face)

- Discretized as small triangular or quadrilateral elements.
- Supports surface currents ($\vec{J}$) induced by incident fields.
- Enforces boundary conditions derived from Maxwell's equations:
  - PEC: $\vec{E}_t = \vec{0}$
  - Dielectric: $\vec{E}_t^{(1)} = \vec{E}_t^{(2)}, \quad \vec{H}_t^{(1)} - \vec{H}_t^{(2)} = \vec{K}$
- Surface currents are discretized using basis functions (e.g., RWG).
- Integral equations relate $\vec{J}$ to scattered fields via Green's functions.

Edges

- **Shared boundaries between adjacent facets.**
- **Enforces physical continuity of surface current, $\vec{J}$.**
- **Charge conservation at the edge:**

$$\nabla_s \cdot \vec{J} = -j\omega\rho \tag{5.1}$$

where ρ is the surface charge density.

- **Used in testing (e.g., Galerkin's method) to evaluate interaction integrals.**

Boundary Conditions

- **Maxwell's boundary conditions on facets:**

$$\textbf{PEC:} \quad \vec{E}_t = \vec{0}$$

$$\textbf{Dielectric:} \quad \vec{E}_t^{(1)} = \vec{E}_t^{(2)}$$

$$\vec{H}_t^{(1)} - \vec{H}_t^{(2)} = \vec{K}$$

- **Continuity enforced on edges:**

$$\vec{J}_{\textbf{facet 1}} \cdot \hat{n}_{\textbf{edge}} = \vec{J}_{\textbf{facet 2}} \cdot \hat{n}_{\textbf{edge}}$$

- **Ensures no spurious currents or charge accumulation.**

Interplay Between Face and Edge

- **Facet**: Supports surface currents $\vec{J}$ and tangential electric field $\vec{E}_t$.
- **Edge**: Ensures:
  - Continuity of $\vec{J}$ across facets.
  - Charge conservation, (5.1):
- Maxwell's equations are satisfied numerically:

$$\nabla \times \vec{H} = \vec{J} + j\omega\epsilon\vec{E}$$

$$\nabla \cdot \vec{E} = \frac{\rho}{\epsilon}$$

RWG Basis Functions – Overview

- **Used to represent surface currents ($\vec{J}$) in MoM simulations.**
- **Defined on pairs of adjacent triangular elements sharing an edge.**
- **Ensures:**
  - **Continuity of surface current across shared edges.**
  - **Sparse and efficient numerical representation.**
- **Piecewise linear variation within triangles.**

RWG Basis Function Definition

- For two adjacent triangles T^+ and T^- sharing edge l_n:
- RWG function $\vec{f}_n(\vec{r})$:

$$\vec{f}_n(\vec{r}) = \begin{cases} \frac{l_n}{2A^+}(\vec{r} - \vec{r}_+), & \vec{r} \in T^+ \\ \frac{l_n}{2A^-}(\vec{r}_- - \vec{r}), & \vec{r} \in T^- \\ 0, & \textbf{otherwise} \end{cases} \tag{1}$$

- Parameters:
  - l_n: Length of the shared edge.
  - A^+, A^-: Areas of triangles T^+ and T^-.
  - $\vec{r}_+$, $\vec{r}_-$: Opposite vertices in T^+, T^- relative to l_n.

Surface Current Representation

- **Total surface current density $\vec{J}(\vec{r})$:**

$$\vec{J}(\vec{r}) = \sum_n I_n \vec{f}_n(\vec{r}) \tag{5.2}$$

- **I_n: Coefficients representing the current magnitude for basis function n.**
- **RWG basis functions provide local support, simplifying matrix assembly.**

Matrix Assembly in MoM

- Integral form of Maxwell's equations discretized using RWG functions.
- Resulting system of equations:

$$\mathbf{Z}\mathbf{I} = \mathbf{V} \tag{5.3}$$

- Terms:
  - $\mathbf{Z}$: Impedance matrix from basis function interactions.
  - $\mathbf{I}$: Vector of current coefficients (I_n).
  - $\mathbf{V}$: Excitation vector from incident fields.

Key Properties of RWG

- **Continuity**:

$$\vec{J}_{\text{facet 1}} \cdot \hat{n}_{\text{edge}} = \vec{J}_{\text{facet 2}} \cdot \hat{n}_{\text{edge}} \tag{4}$$

  Ensures smooth current flow across edges.
- **Sparse Representation**:
  - Non-zero support only on two triangles sharing an edge.
- **Accuracy**:
  - Captures linear current variations.
  - Suitable for arbitrary geometries.

Summary of RWG Functions

- Represent surface currents in MoM using triangular mesh discretization.
- Defined on pairs of adjacent triangles sharing a common edge.
- Ensure:
  - Continuity of surface currents across edges.
  - Sparse, efficient representation of $\vec{J}$.
- Efficient matrix assembly in MoM simulations.

Impedance Matrix

- Each element Z_{mn} evaluates interaction between basis functions:

$$Z_{mn} = \iint \vec{f}_m(\vec{r}) \cdot \vec{G}(\vec{r}, \vec{r}') \cdot \vec{f}_n(\vec{r}')\, \mathbf{d}S\, \mathbf{d}S' \tag{2}$$

- Terms:
  - $\vec{f}_m(\vec{r})$: RWG basis functions.
  - $\vec{G}(\vec{r}, \vec{r}')$: Green's function coupling source and observation points.
- Dense matrix, costly to compute and store.

Impedance Matrix

- **Each element Z_{mn} evaluates interaction between basis functions:**

$$Z_{mn} = \iint \vec{f}_m(\vec{r}) \cdot \vec{G}(\vec{r}, \vec{r}') \cdot \vec{f}_n(\vec{r}') \, \mathbf{d}S \, \mathbf{d}S' \qquad \textbf{(2)}$$

- **Terms:**
  - $\vec{f}_m(\vec{r})$: **RWG basis functions.**
  - $\vec{G}(\vec{r}, \vec{r}')$: **Green's function coupling source and observation points.**
- **Dense matrix, costly to compute and store.**

Excitation Vector

- **Represents contribution of incident fields:**

$$V_m = \iint \vec{f}_m(\vec{r}) \cdot \vec{E}_{\text{inc}}(\vec{r})\, \mathbf{d}S \qquad \textbf{(3)}$$

- **Terms:**
  - $\vec{E}_{\text{inc}}(\vec{r})$: **Incident electric field.**
  - $\vec{f}_m(\vec{r})$: **RWG basis function.**

Physical and Numerical Behavior

- **Surfaces Reflect**:
  - Represent scattering and reflection of electromagnetic waves.
  - Surface currents ($\vec{J}$) induced by incident fields.
- **Edges Ring**:
  - Enforce continuity of surface currents across facets.
  - Numerical challenges can cause spurious oscillations.
  - Proper charge conservation ensures stable edge behavior.

Challenges in Solving the System

- $\mathbf{Z}$ is dense:
  - High memory requirement ($O(N^2)$).
  - Computationally expensive for direct solvers ($O(N^3)$).
- Ill-conditioning may require preconditioning.

Solution Techniques

- **Direct Solvers**:
  - **Gaussian elimination or LU decomposition.**
  - **Cost:** $O(N^3)$.
- **Iterative Solvers**:
  - **Conjugate Gradient (CG), GMRES.**
  - **Cost per iteration:** $O(N^2)$.
  - **Requires preconditioning for convergence.**
- **Fast Multipole Method (FMM)**:
  - **Reduces complexity to** $O(N \log N)$.
  - **Approximates far-field interactions.**

Summary of Linear System and Solutions

- **Linear system:**

$$\mathbf{ZI} = \mathbf{V} \tag{1}$$

- **Key challenges:**
  - **Dense, large-scale matrix $\mathbf{Z}$.**
  - **Computational cost of direct solvers.**
- **Efficient techniques:**
  - **Iterative solvers for large systems.**
  - **FMM for reducing complexity.**

Literature Survey I

- **Electromagnetic Scattering and MoM:**
  - Harrington (1967, 1987): Foundational work on the Method of Moments for electromagnetic problems Harrington 1967; Harrington 1987.
  - Rao (1980): Triangular patch modeling for arbitrarily shaped surfaces Rao 1980.
  - Mosig (2024): Historical insights into MoM and its applications in electrodynamics Mosig 2024.
- **Radar Cross Section (RCS):**
  - Gordon (1975): Far-field approximations for scattered fields Gordon 1975.

Literature Survey II

- Knott et al. (2004): Comprehensive guide on RCS prediction and measurement Knott, Schaeffer, and Tulley 2004.
- Crocker (2020): Dynamic RCS data handling and analysis Crocker 2020.

- **Müntz-Szász Theorem and Approximation:**
  - Siegel (1972), Sedletskii (2008): Extensions of approximation theorems in weighted spaces Siegel 1972; Sedletskii 2008.
  - Szasz (1916): Approximation by aggregates of powers Szász 1916.
- **Numerical Integration and Harmonics:**

Literature Survey III

- Colombo (1981): Harmonic analysis on spheres for numerical applications Colombo 1981.
- Bellet et al. (2022): Quadrature techniques on the cubed sphere Bellet, Brachet, and Croisille 2022.

- **Computational Methods and Advancements:**
  - Newman (1991): Introduction to MoM for computational physics Newman and Kingsley 1991.
  - Taddei et al. (2014): Fast MoM algorithms for phased arrays Taddei et al. 2014.

Bibliography I

[1] Jean-Baptiste Bellet, Matthieu Brachet, and Jean-Pierre Croisille. “Quadrature and symmetry on the Cubed Sphere”. In: Journal of Computational and Applied Mathematics 409 (2022), p. 114142.

[2] Oscar L Colombo. Numerical methods for harmonic analysis on the sphere. Vol. 310. Department of Geodetic Science, The Ohio State University, 1981.

[3] Dylan Andrew Crocker. “A File Format and API for Dynamic Radar Cross Section Data”. In: (Aug. 2020). DOI: 10.2172/1664641. URL: https://www.osti.gov/biblio/1664641.

Bibliography II

[4] W Gordon. "Far-field approximations to the Kirchoff-Helmholtz representations of scattered fields". In: IEEE Transactions on antennas and propagation 23.4 (1975), pp. 590–592.

[5] Roger F Harrington. "Matrix methods for field problems". In: Proceedings of the IEEE 55.2 (1967), pp. 136–149.

[6] Roger F Harrington. "The method of moments in electromagnetics". In: Journal of Electromagnetic waves and Applications 1.3 (1987), pp. 181–200.

Bibliography III

[7] Eugene F Knott, John F Schaeffer, and Michael T Tulley. Radar cross section. SciTech Publishing, 2004.

[8] Juan R. Mosig. “Roger F. Harrington and the Method of Moments: Part 2: Electrodynamics”. In: IEEE Antennas and Propagation Magazine 66.2 (2024), pp. 24–34. DOI: 10.1109/MAP.2024.3362251.

[9] EH Newman and K Kingsley. “An introduction to the method of moments”. In: Computer physics communications 68.1-3 (1991), pp. 1–18.

Bibliography IV

[10] Sadasiva Madiraju Rao. Electromagnetic scattering and radiation of arbitrarily-shaped surfaces by triangular patch modeling. The University of Mississippi, 1980.

[11] AM Sedletskii. "Approximation of the Müntz-Szász type in weight spaces L p and zeroes of functions of Bergman classes in a half-plane". In: Russian Mathematics 52 (2008), pp. 80–87.

[12] Alan R Siegel. "On the Müntz-Szász theorem for $C[0,1]$". In: Proceedings of the American Mathematical Society 36.1 (1972), pp. 161–166.

Bibliography V

[13] Otto Szász. “Über die Approximation stetiger Funktionen durch lineare Aggregate von Potenzen”. In: Mathematische Annalen 77.4 (1916), pp. 482–496.

[14] Ruggero Taddei et al. “A fast MoM code for finite arrays”. In: 2014 44th European Microwave Conference. 2014, pp. 552–555. DOI: 10.1109/EuMC.2014.6986493.

Data Reduction for Modelling Satellite Radar Cross Sections

Daniel Topa
daniel.topa@hii.com

Huntington Ingalls Industries
Mission Technologies

December 19, 2024