

Slotted Waveguides

An Analysis of T-slots for Array Applications

Master Thesis

February 2000

Anders Jensen

A graduate project submitted in partial fulfilment of the requirements for the degree of Master of Science from the Technical University of Denmark. The work presented was carried out at the Department of Electrical and Computer Engineering at California State University, Northridge.

Supervisors:

Professor Sembiam R. Rengarajan
Department of Electrical and Computer Engineering
California State University, Northridge

Professor Niels Christian Albertsen
Department of Mathematical Modelling
Technical University of Denmark

Abstract

The present master thesis comprises a study of the characteristics of a single T-slot cut in the broadwall in a rectangular waveguide. The T-slot has been analyzed for radiating into one of two regions, a half space or parallel plate region.

The slot impedance is computed. The subsequent formulation involves derivation of an integral equation which is solved by the method of moments using entire basis functions for expansion of the electric field in the slot.

For verification purposes results for longitudinal and transverse slots are included. Good agreement with earlier published data has been established. Curves for various T-slot lengths are presented including resonance data. In case the T-slot radiates into a parallel plate region the effect of plate spacing on the slot impedance is observed. Radiation patterns have been computed and justify the feasibility of this slot geometry for array application having a polarization along the waveguide.



Preface

Waveguide fed slot arrays are used in many high performance radar and communication systems. Most applications require a highly polarized radiation characteristic and a significant suppression of cross-polarized fields. To obtain a polarization across the waveguide, a design with longitudinal rectangular slots has been the preferred choice, and well defined synthesis procedures for this configuration have been developed. When designing an array with a polarization along the waveguide transverse slots in either the broad wall or the narrow wall are used. However, they both have their limitations in the control of the excitation and tilting introduces degraded cross polarization performance.

To overcome these problems a T-slot in the broad wall is suggested as an alternative radiating element. It is considered as two rectilinear slots tied together - a longitudinal slot and a transverse slot. The aim of this research is to study the potential of this slot geometry and develop algorithms for the analysis of a T-slot in a rectangular waveguide. This includes computation of network parameters and radiation characteristics. Based on earlier work carried out by *Robert S. Elliot* [1] a moment method solution is used in the analysis, especially a method using global expansion and weighting functions which has been applied successfully in similar problems by *Lars Josefsson* [2][3][4].

Design and synthesis of arrays can then be done using network theory. However, since there is no mechanism to introduce 180° phase shift, the element spacing has to be one guide wavelength for broadside radiation. The grating lobes generated in the E-plane are suppressed by using a baffle structure.

Overview of the chapters

The report is divided into several parts. Chapter 1 comprises a theoretical study of a single T-slot involving derivation of an integral equation for the slot aperture field. In chapter 2 the applied moment method is outlined and the integral equation is developed further. In chapter 3 results of the analysis is presented. It includes

both network data and radiation patterns. The model and software partially verified comparing results for longitudinal and transverse slot with earlier published results. A short description of the implemented software is given in chapter 4. Derivation of waveguide Green's function and matrix elements are given in appendix B and C with a description of the equivalence principle in appendix A. Finally, appendix D lists the implemented software.

Acknowledgements

I want to thank the Department of Electrical and Computer Engineering at California State University, Northridge and especially Professor Sembiam R. Rengarajan for his support and supervision of this thesis study. Thanks also go to responsible supervisor Professor Niels Christian Albertsen, Technical University of Denmark, to Mikael Dich, Simrad Shipmate and to Hans Erik Gram, Nokia Denmark for proposing this study.

Summary in Danish

Slidset bølgeleder antenner bliver brugt i mange radar- og kommunikationssystemer. De fleste af disse anvendelser kræver antenner med en veldefinerede polarisationsskarakteristik og undertrykkelse af krydspolariserede felter. Langsgående slidser er det fortrukne valg til opnåelse af en polarisation på tværs af bølgelederen, og der er blevet udviklet veldefinerede syntesemetoder til denne antennekonfiguration. Når målet er en polarisation langs bølgelederen benyttes tværgående slidser i enten den brede eller smalle side af bølgelederen. De har imidlertid begge deres begrænsninger i form af en høj eksitation med et lille dynamikområde og skråtilning introducere et krydspolariseret felt.

For at overkomme disse problemer foreslås en T-slids i den brede side af bølgelederen. Den kan betragtes som to rektangulære slidser - en langsgående og en tværgående slids. Målet med dette arbejde er undersøgelse af potentialet for denne slids og udvikling af algoritmer til analyse af en T-slids i en rektangulær bølgeleder. Dette indebærer beregning af spredningsparametre og udstrålingskarakteristikker.

Analysemetoderne baserer sig på tidligere arbejde udført af *Robert S. Elliot* [1]. Der benyttes en moment metode med globale udviklings og testfunktioner, der har været anvendt med succes til løsning af lignende opgaver, *Lars Josefsson* [2][3][4]. Endelig vægtykkelse er inkluderet i modellen for langs- og tværgående slidser. Resultaterne fra analysen af disse to konfigurationer viser meget god overensstemmelse med tidligere publicerede resultater.

Impedansen for en T-slids, der ståler ind i enten et halvplan eller en uendelig plan parallel bølgeleder, er blevet undersøgt. Denne slids viser sig i modsætning til fx den tværgående slids at have et væsentligt bedre dynamik over mulig eksitation. Dog indeholder udstålingen en væsentlig krydspolariseret komponent. Specielt ved et lille eksitationsniveau, hvor det elektriske felt bliver dominerende i den langsgående del af T-slidsen.

Til design og syntese af arrays benyttes sædvanligvis netværksteori. Da der er i mi-

dlertid ikke findes en mekanisme til at introducere et 180° faseskift, må de enkelte elementer have en indbyrdes afstand paa én ledt bølgelængde. Dette giver anledning til en sidesløjfe i E-planet, men ved brug af en plan parallel bølgeleder uden på den rektangulære bølgeleder kan disse undertrykkes. Dette gælder dog ikke i samme grad for det krydspolariserede felt. Der er derfor et "trade off" ved den opnåede bedre dynamik i området for eksitationsniveauet.

Contents

1. Theory of the T-Slot	1
1.1. General Assumptions	2
1.2. An Equation for the T-Slot Aperture E-field	2
1.2.1. Exterior Contribution to the H-field in a Half Space Environment	4
1.2.2. Interior Contribution to the H-field	7
1.2.3. Incident Field Contribution	8
1.3. A T-slot Radiating into an Infinite PP Waveguide	8
1.4. Scattered Field	12
1.5. Radiated Field	14
1.5.1. Half Space Environment	15
1.5.2. PP Waveguide	15
2. Numerical Computation Methods	19
2.1. Method of Moments	19
2.1.1. The Integral Equation	19
2.1.2. Discretization of the Unknown Function	20
2.1.3. Solution of the Discretized Operator Equation	21
2.1.4. The Direct MoM for Solving an Operator Equation	21
2.2. Application of the MoM to Slot Problems	22
2.2.1. A T-slot Radiating into a Half Space Environment	24
2.2.2. A T-slot Radiating into an Infinite PP Waveguide	29
2.2.3. Reduction Scheme for Triple Integrals	33
2.2.4. 1/R Singularity Extraction	36
2.2.5. Deformed Contour Integral	40
2.2.6. Method of Weighted Averages	41
2.3. Scattered Field	43
2.4. Radiated Field	45
2.4.1. Half Space Environment	45
2.4.2. PP Waveguide	46
2.5. Finite Wall Thickness	49

2.6. Correction to Rounded Slot Ends	53
3. Results	55
3.1. Longitudinal Slots	55
3.2. Transverse Slots	57
3.3. T-Slots	58
3.3.1. Analysis of Convergence	59
3.3.2. Electric Field Distribution	59
3.3.3. Impedance Data	60
3.3.4. Radiation Patterns	74
4. Software	83
4.1. Organization of the Documentation	83
4.2. Software Packages	85
Conclusion	87
Bibliography	89
A. The Equivalence Principle	91
A.1. Equivalence Principle for an Open Half-Space	91
B. Green's Functions for Rectangular Waveguides	93
B.1. Green's Integral Theorems	93
B.2. Green's Theorems Applied to the Rectangular Waveguide	94
B.3. Transformation of the Integral for $H_z(\vec{R})$	95
B.4. Explicit Expression for Green's functions G_1 for a Rectangular Waveguide	97
B.5. An Integral Equation for $H_x(\vec{R})$	100
C. Matrix Elements	101
C.1. Self Terms for Longitudinal Part of T-Slot	101
C.2. Self Terms for Transverse Part of T-Slot	105
C.3. Cross Coupling Terms	109
C.4. Excitation Terms	118
D. Source Code	119

List of Figures

- 1.1. Geometry of waveguide-fed T-slot. 1
- 1.2. Waveguidefed T-slot radiating into a PP waveguide 9

- 2.1. Outline of the Method of Moments. 20
- 2.2. Integration on strips of constant $\nu, \sigma - \sigma'$ 35
- 2.3. Integration over a rectangular region in polar coordinates. 37
- 2.4. Value of the kernel I_{R_1} 39
- 2.5. Value of the kernel I_{R_2} 40
- 2.6. Integration contour in the complex k_z plane. 41
- 2.7. Rectangular cavity representing the finite wall thickness. 50

- 3.1. Longitudinal slot admittance - zero wall thickness. 56
- 3.2. Longitudinal slot resonant length versus offset. 56
- 3.3. Slot resonant length versus offset for a longitudinal radiating into a PP region. 57
- 3.4. Tranverse slot resistance - zero wall and offset. 58
- 3.5. Tranverse slot reactance - zero wall and offset. 59
- 3.6. Transverse slot resistance versus frequency for three slot lengths 60
- 3.7. Transverse slot reactance versus frequency for three slot lengths 61
- 3.8. Decay of expansion coefficients 62
- 3.9. Slot electric field distribution. $L_x = 32mm$ 63
- 3.10. Slot electric field distribution. $L_x = 44mm$ 64
- 3.11. T-slot resistance versus slot length 65
- 3.12. T-slot reactance versus slot length 66
- 3.13. T-slot resonant resistance versus slot length L_x 67
- 3.14. T-slot resonant length $Lz + 2L_x$ versus slot length L_x 68
- 3.15. T-slot resistance versus slot length 69
- 3.16. T-slot reactance versus slot length 70
- 3.17. T-slot resonant length $Lz + 2L_x$ versus slot length L_x 71
- 3.18. T-slot resonant resistance versus slot length L_x 72
- 3.19. T-slot impedance versus baffle spacing in free space wavelengths. . . . 73

3.20. E-plane patterns of a T-slot.	74
3.21. H-plane patterns of a T-slot.	75
3.22. E-plane patterns of a T-slot.	76
3.23. H-plane patterns of a T-slot.	77
3.24. E-plane patterns of a 9 element array of T-slots.	78
3.25. H-plane patterns of a 9 element array of T-slots.	79
3.26. E-plane patterns of a 9 element array of T-slots having alternating offset.	80
3.27. E-plane patterns of a 9 element array of T-slots having alternating offset. PP spacing was $a' = 19.5mm$	81
4.1. Flowchart for MoM program.	84
A.1. Equivalence for region 1 ($y < b$).	92
B.1. Definition of coordinates for rectangular waveguide.	95
C.1. Separating the weighting integral into three cases depending on the value of ζ	110

Chapter 1

Theory of the T-Slot

The objective of this chapter is to develop the theory for the radiation and scattering from a T-slot in a rectangular waveguide. This T-slot is considered as two rectangular slots tied together - a longitudinal slot and a transverse slot - which makes it possible to verify earlier work on those two types of slots. A first step in such a derivation is

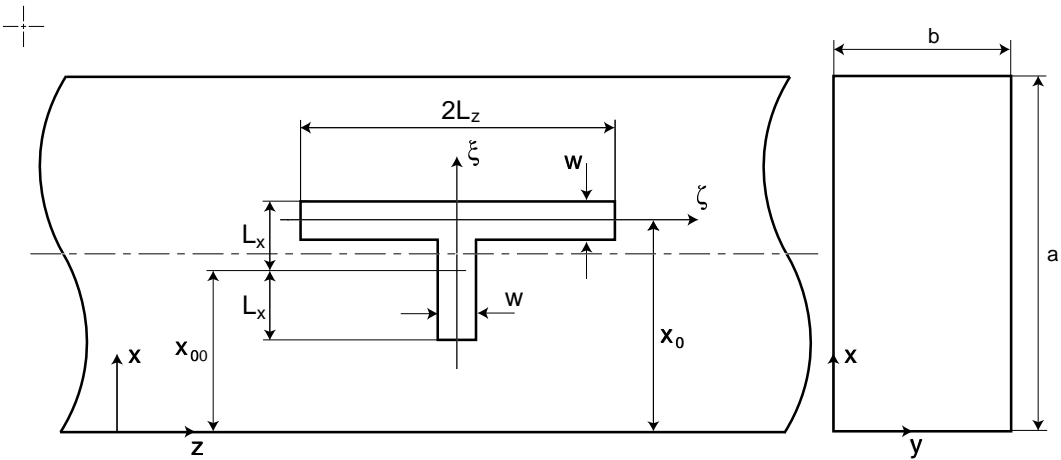


Figure 1.1.: Geometry of waveguide-fed T-slot.

to identify the structure and geometry of the problem considered. Figure 1.1 shows a single T-slot in the broadwall of a rectangular waveguide. There are two coordinate systems. The waveguide is positioned in a global (x, y, z) coordinate system while a local (ξ, η, ζ) coordinate system is used for the slot itself. We shall use primed coordinates denoting source points and unprimed coordinates denoting observation points. The length of the two slots are $2L_z$ and $2L_x$ respectively. The width of both slots is w . Furthermore, the offset of the slot is x_0 as it was defined in [2] for a

longitudinal slot. In addition x_{00} defines the offset of the transverse part of the slot in a similar manner. The waveguide is shown with zero wall thickness since the model presented here is based on that assumption. However, for verification purposes we shall subsequently extend the model to the finite thickness case, but only for longitudinal and transverse slots.

1.1. General Assumptions

In order to simplify the derivation, it is necessary to make some assumptions regarding the field components as well as the structure. Therefore, the following derivation will be based on the same assumption as those given in [2][4]

- The upper broad wall has zero thickness
- The slot ground plane has infinite extent
- The electric field component along the slot in the longitudinal and transverse part of the aperture are neglected
- The boundary condition for the magnetic field component along the slot for both the longitudinal and transverse parts of the T-slot is enforced
- The slot is excited by an incident TE_{10} wave in the waveguide
- The unknown electric fields are considered to be constant across the slots.

1.2. An Equation for the T-Slot Aperture E-field

Like all other problems dealing with electro dynamics analysis we start with Maxwell's equations. For time-harmonic fields, Maxwell's equations with $e^{j\omega t}$ are [5]

$$\nabla \times \mathbf{E} = -j\omega\mathbf{B} - \mathbf{M} \quad (1.1a)$$

$$\nabla \times \mathbf{H} = j\omega\mathbf{D} + \mathbf{J} \quad (1.1b)$$

$$\nabla \cdot \mathbf{D} = \rho \quad (1.1c)$$

$$\nabla \cdot \mathbf{B} = 0 \quad (1.1d)$$

where $j = \sqrt{-1}$, $\omega = 2\pi f$, etc., are standard notations in *SI - units*. The four field quantities of interest are the electric field \mathbf{E} , the magnetic intensity \mathbf{H} , the electric displacement field \mathbf{D} and the magnetic flux field \mathbf{B} . These fields along with the source

terms - electric current density \mathbf{J} as well as magnetic current density \mathbf{M} are the bases of the following derivations. In a free-space environment the constitutive relations are

$$\mathbf{D} = \epsilon_0 \mathbf{E} \quad (1.2a)$$

$$\mathbf{B} = \mu_0 \mathbf{H} \quad (1.2b)$$

where $\epsilon_0 = 10^{-9}/36\pi[F/m]$ is the permittivity of free space and $\mu_0 = 4\pi \times 10^{-7}[H/m]$ is the permeability of free space.

With the assumptions made above in mind we impose the continuity of the tangential components of the magnetic fields in the slot aperture

$$H_x^{ext}(\bar{R}) - H_x^{int}(\bar{R}) = H_x^{inc}(\bar{R}) \quad (1.3a)$$

$$H_z^{ext}(\bar{R}) - H_z^{int}(\bar{R}) = H_z^{inc}(\bar{R}) \quad (1.3b)$$

where *ext*, *int* and *inc* refer to external, internal and incident fields respectively and \bar{R} is a field point. Using Schelkunoff's equivalence principle described in appendix A permits the slot aperture to be covered with a perfect conducting surface and introduction of appropriate magnetic current sheets in the slot aperture as well as removal of the infinite ground plane. The electromagnetic fields in the half space $\eta > 0$ are therefore entirely due to the sheets of magnetic sources

$$\hat{z}M_z = -2\mathbf{n} \times \hat{x}E_x = 2\hat{z}E_x \quad (1.4a)$$

$$\hat{x}M_x = -2\mathbf{n} \times \hat{z}E_z = -2\hat{x}E_z \quad (1.4b)$$

where \mathbf{n} is a unit normal vector directed into the region of interest. The factor 2 is due to the application of image principle. Inside the waveguide, Schelkunoff's equivalence principle gives directly

$$\hat{z}M_z = -\mathbf{n} \times \hat{x}E_x = -\hat{z}E_x \quad (1.5a)$$

$$\hat{x}M_x = -\mathbf{n} \times \hat{z}E_z = \hat{x}E_z \quad (1.5b)$$

Finally $\mathbf{n} \times \mathbf{E} = 0$ applies to the conducting sheet at $\eta = 0$.

The objective is now to derive equations for the exterior and interior components in eq. 1.3a and 1.3b only in terms of the sources M_z and M_x which through eq. 1.4a-1.5b is given by the unknown electric field distribution in the slot. Due to the nature of this problem, we split it into two parts. One dealing with the unbounded exterior

half space and one dealing with the bounded waveguide region. The first is solved by introduction of a vector potential and the latter is solved by eigenvalue expansion.

1.2.1. Exterior Contribution to the H-field in a Half Space Environment

In this section we shall solve the exterior problem in case the T-slot radiates into a half space.

Taking the curl of eq. 1.1b, using eq. 1.1a and the constitutive relations, it is found that

$$\nabla \times \nabla \times \mathbf{H} - k^2 \mathbf{H} = -j\omega\epsilon \mathbf{M} \quad (1.6)$$

where $k^2 = \omega^2 \mu_0 \epsilon_0$ is the free space wave number. This is the equation to be solved to find the magnetic field directly in terms of the specified current source \mathbf{M} . An equation that is simpler to solve is obtained by introducing the vector potential \mathbf{F} and scalar potential Φ . Since we do not have any electric currents or charge, the divergence of the electric displacement field \mathbf{D} is identically zero. Therefore, \mathbf{D} can be expressed as [6]

$$\mathbf{D} = -\nabla \times \mathbf{F} \quad (1.7)$$

because $\nabla \cdot \nabla \times \mathbf{F} \equiv 0$. By using eq. 1.7 in eq. 1.1b, we obtain

$$\nabla \times (\mathbf{H} + j\omega \mathbf{F}) = 0 \quad (1.8)$$

Any function with zero curl can be expressed as a gradient of a scalar function. Thus

$$\mathbf{H} + j\omega \mathbf{F} = -\nabla \Phi_m \quad (1.9)$$

In order that eq. 1.1a will hold we require

$$\begin{aligned} -\nabla \times \epsilon_0 \mathbf{E} &= \nabla \times \nabla \times \mathbf{F} \\ &= j\omega \mu_0 \epsilon_0 \mathbf{H} + \epsilon_0 \mathbf{M} \\ &= j\omega \mu_0 \epsilon_0 (-j\omega \mathbf{F} - \nabla \Phi_m) + \epsilon_0 \mathbf{M} \end{aligned} \quad (1.10)$$

We use the expansion [6] $\nabla \times \nabla \times \mathbf{F} = \nabla \nabla \cdot \mathbf{F} - \nabla^2 \mathbf{F}$ to obtain after a rearrangement of terms

$$\nabla^2 \mathbf{F} + k^2 \mathbf{F} = -\epsilon_0 \mathbf{M} + \nabla(\nabla \cdot \mathbf{F} + j\omega \mu_0 \epsilon_0 \Phi_m) \quad (1.11)$$

We specify the divergence of \mathbf{F} according to the the Lorentz condition [6]

$$\nabla \cdot \mathbf{F} = -j\omega\mu_0\epsilon_0\Phi_m \quad (1.12)$$

The equation for \mathbf{F} now becomes the inhomogeneous Helmholtz equation

$$\nabla^2\mathbf{F} + k^2\mathbf{F} = -\epsilon_0\mathbf{M} \quad (1.13)$$

This equation is solved by introducing the scalar Green's function in an unbounded region, which satisfies the following inhomogeneous scalar wave equation

$$(\nabla^2 + k^2) G(\bar{R}, \bar{R}') = -\delta(\bar{R} - \bar{R}') \quad (1.14)$$

where δ is the Dirac delta function. If we make a change of variable $\bar{R} - \bar{R}' = \bar{R}_1$, the problem will have spherical symmetry with respect to the new origin $0'$. Hence the function G will be a function of R_1 only. In terms of the new spherical coordinate system, the Green function satisfies the equation

$$\frac{1}{R_1^2} \frac{d}{dR_1} \left[R_1^2 \frac{dG(\bar{R}_1, 0)}{dR_1} \right] + k^2 G(\bar{R}_1, 0) = -\frac{\delta(R_1 - 0)}{4\pi R_1^2} \quad (1.15)$$

The weighting factor $1/4\pi R_1^2$ attached to $\delta(R_1 - 0)$ is due to the fact

$$\int_V \delta(\bar{R} - \bar{R}') dV = 1 \quad (1.16)$$

where V encloses $0'$. For $R_1 \neq 0$, the homogeneous equation is the same as the spherical Bessel equation of zero'th order. The appropriate solution for $G(\bar{R}_1, 0)$ must be proportional to the spherical Hankel function of the first kind of the zero'th order or [7]

$$G(\bar{R}_1, 0) = Ah_0^{(1)}(kR_1) = A \left(\frac{e^{-jkR_1}}{jkR_1} \right) \quad (1.17)$$

To determine the constant of proportionality A , we make use of the Gauss theorem

$$\int_V \nabla^2 \phi dV = \int_S \nabla \phi \hat{n} dS \quad (1.18)$$

thus a volume of integration of Eq. 1.15 through a small sperical region with center at

0' gives

$$4\pi R_1^2 \left| \frac{dG(\bar{R}_1, 0)}{dR_1} \right|_{R_1=a} + k^2 \int_0^a G(\bar{R}_1, 0) 4\pi R_1^2 dR_1 = -1 \quad (1.19)$$

Upon substituting eq. 1.17 into eq. 1.19 and letting $a \rightarrow 0$, we obtain $A = jk/4\pi$. Therefore, the complete expression for $G(\bar{R}_1, 0)$ is given by

$$G(\bar{R}_1, 0) = \frac{e^{-jkR_1}}{4\pi R_1} \quad (1.20)$$

Transforming back into the original coordinate system with center at 0 yields

$$G(\bar{R}, \bar{R}') = \frac{e^{-jkR}}{4\pi R} \quad (1.21)$$

where $R = |\bar{R} - \bar{R}'| = \sqrt{(\xi - \xi')^2 + (\eta - \eta')^2 + (\zeta - \zeta')^2}$. However, since $\eta = \eta'$ $G(\bar{R}, \bar{R}')$ reduces to a 2 dimensional function. The solution to eq. 1.13 is obtained by a convolution with the forcing function $\epsilon_0 \mathbf{M}$ according to

$$\mathbf{F}(\bar{R}) = \int_{slot} \epsilon_0 \mathbf{M}(\bar{R}') G(\bar{R}, \bar{R}') dS' \quad (1.22)$$

The magnetic fields are now calculated in terms of the vector potential using the Lorentz condition in Eq. 1.9

$$\mathbf{H}(\bar{R}) = -j\omega \mathbf{F}(\bar{R}) + \frac{\nabla \nabla \cdot \mathbf{F}(\bar{R})}{j\omega \mu_0 \epsilon_0} \quad (1.23)$$

Separating into the two components of interest and applying eq. 1.22, the two exterior terms in eq. 1.3a and 1.3b are rewritten in the following final form

$$H_x^{ext}(\bar{R}) = \frac{-2}{j\omega \mu_0} \int_{slot} \left[E_z(\bar{R}') \left(k^2 + \frac{\partial^2}{\partial x'^2} \right) G(\bar{R}, \bar{R}') - E_x(\bar{R}') \frac{\partial^2}{\partial x' \partial z'} G(\bar{R}, \bar{R}') \right] dS' \quad (1.24a)$$

$$H_z^{ext}(\bar{R}) = \frac{2}{j\omega \mu_0} \int_{slot} \left[E_x(\bar{R}') \left(k^2 + \frac{\partial^2}{\partial z'^2} \right) G(\bar{R}, \bar{R}') - E_z(\bar{R}') \frac{\partial^2}{\partial z' \partial x'} G(\bar{R}, \bar{R}') \right] dS' \quad (1.24b)$$

1.2.2. Interior Contribution to the H-field

The derivation of the expressions for the interior contribution is quite comprehensive and most details can be seen in appendix B. It involves introduction of several concepts such as Green's Theorem, integral transformations as well as derivation of Green's functions for the waveguide region.

With our coordinate definition (x, y, z) , B.8 and B.43 becomes

$$H_x^{int}(\bar{R}) = \frac{1}{j\omega\mu_0} \int_{slot} \left[E_z(\bar{R}') \left(\frac{\partial^2}{\partial x'^2} + k^2 \right) G_2(\bar{R}, \bar{R}') - E_x(\bar{R}') \frac{\partial^2 G_2(\bar{R}, \bar{R}')}{\partial x' \partial z'} \right] dS' \quad (1.25a)$$

$$H_z^{int}(\bar{R}) = \frac{-1}{j\omega\mu_0} \int_{slot} \left[E_x(\bar{R}') \left(\frac{\partial^2}{\partial z'^2} + k^2 \right) G_1(\bar{R}, \bar{R}') + E_z(\bar{R}') \frac{\partial^2 G_1(\bar{R}, \bar{R}')}{\partial x' \partial z'} \right] dS' \quad (1.25b)$$

The two Green's functions are derived from eq. B.38 and B.40

$$G_1(\bar{R}, \bar{R}') = \frac{2}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x}{a} \cos \frac{m\pi x'}{a} e^{-\gamma_{mn}|z-z'|} \quad (1.26a)$$

$$G_2(\bar{R}, \bar{R}') = \frac{2}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \sin \frac{m\pi x}{a} \sin \frac{m\pi x'}{a} e^{-\gamma_{mn}|z-z'|} \quad (1.26b)$$

wherein $\epsilon_{00} = 1/2$, $\epsilon_{m0} = \epsilon_{0n} = 1/\sqrt{2}$ and $\epsilon_{mn} = 1$ otherwise. Finally, propagation constant γ_{mn} is given by

$$\gamma_{mn} = \sqrt{\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2 - k^2} \quad (1.27)$$

where the branches of the square root must be chosen so that real and imaginary part of γ_{mn} are positive.

1.2.3. Incident Field Contribution

The incident magnetic field H_z^{inc} due to a TE_{10} wave is given by [8]

$$H_z^{inc}(\bar{R}) = jA_{10} \cos \frac{\pi(x_0 + \xi)}{a} e^{-j\beta_{10}\zeta} \quad (1.28)$$

where β_{10} is the wave number of the TE_{10} wave [5]

$$\beta_{10} = k \sqrt{1 - \left(\frac{\lambda}{2a}\right)^2} \quad (1.29)$$

A_{10} gives the amplitude of the incident wave, but for convenience we let $A_{10} = 1$. To ensure consistency we define the H_x^{inc} component as [8]

$$h_x = \frac{1}{Z_{TE}} \frac{j\omega\mu_0}{k^2 - \beta_{10}^2} \hat{z} \times \hat{z} \times \nabla_x h_z \quad (1.30)$$

where $Z_{TE} = \omega\mu_0/\beta_{10}$. Upon evaluation, this expression becomes

$$H_x^{inc}(\bar{R}) = \frac{-\beta_{10}}{\pi/a} A_{10} \sin \frac{\pi(x_0 + \xi)}{a} e^{-j\beta_{10}\zeta} \quad (1.31)$$

1.3. A T-slot Radiating into an Infinite PP Waveguide

With the purpose of studying the applicability of a T-slot for array applications we shall extend the model of a T-slot radiating into half space to one radiating into a parallel plate (PP) waveguide. Figure 1.2(a) shows the geometry of the slot problem which is to be analyzed in this section whereas figure 1.2(b) represents a more practical configuration with a finite length PP waveguide. Compared with figure 1.1 we have introduced the parallel plate spacing a' . It is assumed that the baffle structure as well as the waveguide possess no variation in the direction of the z -axis. This ensures that the region bounded by the baffles and the waveguide can be assumed cylindrical. The object of the analysis is the determination of fields in the parallel plate region.

To evaluate the magnetic field in the PP waveguide we will use a modal expansion of the field. In this way the common double integral in the plane wave expansion of the external field from a single slot in a ground plane is now replaced by one infinite integral and one infinite summation.

Similar to previous derivations the H-fields are found from a known Green's function and the sources. In this case a thorough derivation of appropriate Green's function

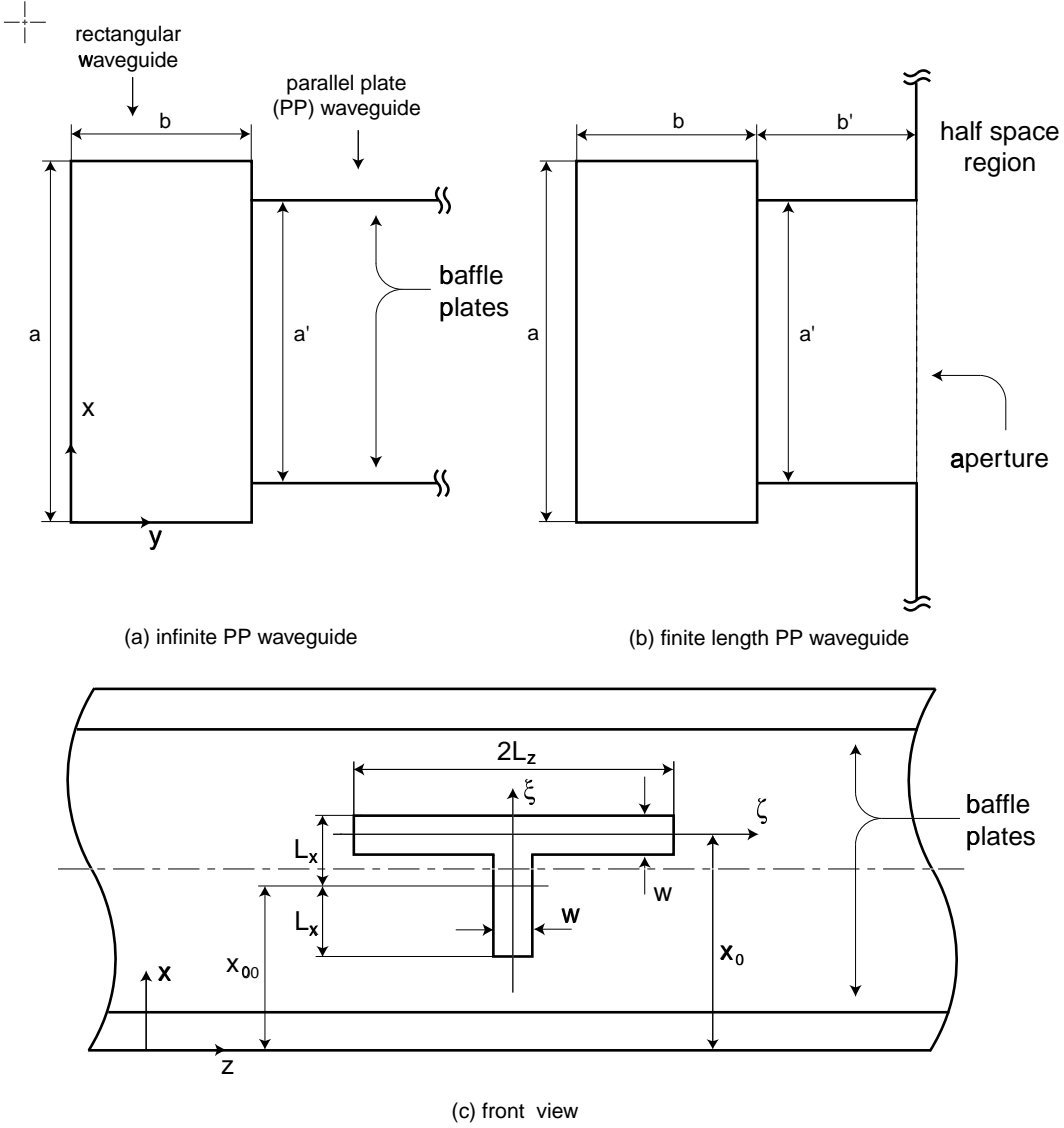


Figure 1.2.: Geometry of waveguide-fed T-slot radiating into a parallel plate waveguide consisting of two baffle plates.

can be found in [7] and for a while we shall make use of the dyadic notation used in the book by *Chen-To Tai*. However, as a beginning we shall use the two Maxwell equations. Taking the curl of eq. 1.1a and inserting eq. 1.1b gives

$$\nabla \times \nabla \times \mathbf{H} - k^2 \mathbf{H} = -j\omega\epsilon_0 \mathbf{M} \quad (1.32)$$

According to [7] the particular Green's function for this case is denoted $\overline{\overline{G}}_{e2}$ and the solution of

$$\nabla \times \nabla \times \overline{\overline{G}}_{e2} - k^2 \overline{\overline{G}}_{e2} = \overline{\overline{I}} \delta(\bar{R} - \bar{R}') \quad (1.33)$$

with the boundary conditions

$$\mathbf{n} \times \nabla \times \overline{\overline{G}}_{e2} = 0 \quad \text{on the conducting boundary} \quad (1.34)$$

Then the H-fields can be found by the convolution

$$\mathbf{H} = 2j\omega\epsilon \int \overline{\overline{G}}_{e2} \cdot [\mathbf{n} \times \mathbf{E}] dS' \quad (1.35)$$

This expression is valid if the source is located at $y' = 0$ and we have a short at $y = 0$. The latter causes a doubling of the magnetic currents $\mathbf{M} = -\mathbf{n} \times \mathbf{E}$. The dyadic Green's function is found from eq. 5.105 in [7]

$$\begin{aligned} \overline{\overline{G}}_{e2} = & \frac{1}{k^2} \hat{y} \hat{y} \delta(\bar{R}, \bar{R}') + \int_{-\infty}^{\infty} dk_z \sum_{m=0}^{\infty} \frac{-j2\epsilon_m^2}{4\pi a' k_y (k_x^2 + k_z^2)} \\ & \cdot [\bar{M}_{om}(\pm k_y, k_z) \bar{M}'_{om}(\mp k_y, -k_z) \\ & + \bar{N}_{em}(\pm k_y, k_z) \bar{N}'_{em}(\mp k_y, -k_z)] \quad y \lesseqgtr y' \end{aligned} \quad (1.36)$$

where k_x and k_z are independent spectral variables and k_y is the propagation constant in the y -direction. Due to the boundary conditions mentioned above the variable k_x can only take certain discrete values given by

$$k_x = \frac{m\pi}{a'}, \quad m = 1, 2, \dots \quad (1.37)$$

Thus the modal expansion is continuous in k_z and discrete in k_x . The propagation constant k_y is related to k_x and k_z as

$$k_y = \sqrt{k^2 - k_x^2 - k_z^2} \quad (1.38)$$

where the branches of the square root must be chosen so that its real part is positive and the imaginary part is negative. Finally the two functions \bar{M} and \bar{N} are given by

$$\bar{M}_{om}(k_y, k_z) = -e^{-j(k_z z - k_y y)} [\hat{x} j k_z \sin(k_x x) + \hat{z} k_x \cos(k_x x)] \quad (1.39)$$

$$\bar{N}_{em}(k_y, k_z) = \frac{1}{k} e^{-j(k_z z - k_y y)}. \quad (1.40)$$

$$[\hat{x}j k_x k_y \sin(k_x x) - \hat{y} [k_x^2 + k_z^2] \cos(k_x x) - \hat{z} k_y k_z \cos(k_x x)]$$

These functions have been changed in order to satisfy a change in origin of the problem defined by eq. 1.33.

Extracting terms with x and z -directed sources from eq. 1.36 gives

$$\begin{aligned} \overline{\overline{G}}_{e2} = & \int_{-\infty}^{\infty} dk_z \sum_{m=0}^{\infty} \frac{-j2\epsilon_m^2}{4\pi a' k_y (k_x^2 + k_z^2)} e^{-jk_y(y-y')} e^{-jk_z(z-z')} \\ & \cdot \left[\hat{x}\hat{x} \left[k_z^2 + \frac{k_x^2 k_y^2}{k^2} \right] \sin(k_x x) \sin(k_x x') \right. \\ & - \hat{y}\hat{x} j \frac{k_x k_y (k_x^2 + k_z^2)}{k^2} \cos(k_x x) \sin(k_x x') \\ & - \hat{z}\hat{x} j \left[k_x k_z + \frac{k_x k_z k_y^2}{k^2} \right] \cos(k_x x) \sin(k_x x') \\ & + \hat{x}\hat{z} j \left[k_x k_z - \frac{k_x k_z k_y^2}{k^2} \right] \sin(k_x x) \cos(k_x x') \\ & - \hat{y}\hat{z} \frac{k_y k_z (k_x^2 + k_z^2)}{k^2} \cos(k_x x) \cos(k_x x') \\ & \left. + \hat{z}\hat{z} \left[k_x^2 + \frac{k_y^2 k_z^2}{k^2} \right] \cos(k_x x) \cos(k_x x') \right] \end{aligned} \quad (1.41)$$

which can be rewritten as

$$\begin{aligned} \overline{\overline{G}}_{e2} = & \int_{-\infty}^{\infty} dk_z \sum_{m=0}^{\infty} \frac{-j\epsilon_m^2}{2\pi a' k_y k^2} e^{-jk_y(y-y')} e^{-jk_z(z-z')} \\ & \cdot \left[\hat{x}\hat{x} (k^2 - k_x^2) \sin(k_x x) \sin(k_x x') \right. \\ & - \hat{y}\hat{x} j k_x k_y \cos(k_x x) \sin(k_x x') \\ & - \hat{z}\hat{x} j k_x k_z \cos(k_x x) \sin(k_x x') \\ & + \hat{x}\hat{z} j k_x k_z \sin(k_x x) \cos(k_x x') \\ & \left. - \hat{y}\hat{z} j k_y k_z \cos(k_x x) \cos(k_x x') \right] \end{aligned} \quad (1.42)$$

$$+\hat{z}\hat{z}(k^2 - k_z^2) \cos(k_x x) \cos(k_x x')]$$

Separating eq. 1.35 into components of interest we obtain the following expressions similar to eq. 1.24a and 1.24a

$$H_x^{ext}(\bar{R}) = 2j\omega\epsilon_0 \int_{slot} -G_{e2,xz}(\bar{R}, \bar{R}') E_x(\bar{R}') + G_{e2,xx}(\bar{R}, \bar{R}') E_z(\bar{R}') dS' \quad (1.43a)$$

$$H_z^{ext}(\bar{R}) = 2j\omega\epsilon_0 \int_{slot} -G_{e2,zz}(\bar{R}, \bar{R}') E_x(\bar{R}') + G_{e2,zx}(\bar{R}, \bar{R}') E_z(\bar{R}') dS' \quad (1.43b)$$

Insertion of \bar{G}_{e2} gives

$$H_x^{ext}(\bar{R}) = \frac{-1}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k_y} \sin(k_x x) e^{-jk_y y} e^{-jk_z z} \cdot \left[jk_x k_z \int_{slot} E_x(\bar{R}') \cos(k_x x') e^{jk_z z'} dS' - (k^2 - k_x^2) \int_{slot} E_z(\bar{R}') \sin(k_x x') e^{jk_z z'} dS' \right] dk_z \quad (1.44a)$$

$$H_z^{ext}(\bar{R}) = \frac{-1}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k_y} \cos(k_x x) e^{-jk_y y} e^{-jk_z z} \cdot \left[(k^2 - k_z^2) \int_{slot} E_x(\bar{R}') \cos(k_x x') e^{jk_z z'} dS' + jk_x k_z \int_{slot} E_z(\bar{R}') \sin(k_x x') e^{jk_z z'} dS' \right] dk_z \quad (1.44b)$$

which is the final form. Expressions for the interior contribution as well as the incident fields are identical with those given in sect. 1.2.2 and 1.2.3.

1.4. Scattered Field

The internally scattered field in the waveguide and in particular the amplitude of the fundamental mode is obtained from one of the functions $H_x^{int}(E_z)$ or $H_z^{int}(E_x)$. The equivalent network parameters (reflection coefficient, shunt admittance etc.) can be evaluated easily. Now only looking at H_x , the incident field is given in eq. 1.31. The

fundamental mode of the backscattered field can be written similarly

$$H_x^{b,scat}(\bar{R}) = B_{10} \frac{\beta_{10}}{\pi/a} \sin \frac{\pi(x_{00} + \xi)}{a} e^{j\beta_{10}\zeta} \quad (1.45)$$

This quantity can be written in terms of eq. 1.25a

$$H_x^{b,scat}(\bar{R}) = \frac{A_{10}}{j\omega\mu_0} \int_{slot} E_z(\bar{R}') \left(\frac{\partial^2}{\partial \xi'^2} + k^2 \right) G_2(\bar{R}, \bar{R}') - E_x(\bar{R}') \frac{\partial^2 G_2(\bar{R}, \bar{R}')}{\partial \xi' \partial \zeta'} dS' \quad (1.46)$$

Insertion of the Green's function given in eq. 1.26b, this expression becomes

$$H_x^{b,scat}(\bar{R}) = \frac{-A_{10}}{\omega\mu_0 ab} \sin \left[\frac{\pi}{a}(x_{00} + \xi) \right] e^{j\beta_{10}\zeta} \cdot \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} E_z(\bar{R}') \sin \left[\frac{\pi}{a}(x_{00} + \xi') \right] e^{-j\beta_{10}\zeta'} d\xi' d\zeta' + \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} E_x(\bar{R}') \cos \left[\frac{\pi}{a}(x_0 + \xi') \right] e^{-j\beta_{10}\zeta'} d\zeta' d\xi' \right] \quad (1.47)$$

with the field point to the left of the slot ($\zeta < -L_z$) cf. [8] pp. 421. The branch of the square root in eq. 1.27 has been chosen so that its imaginary part is positive. Then equating eq. 1.45 and 1.47 gives the reflection coefficient

$$\frac{B_{10}}{A_{10}} = \frac{-\pi}{\omega\mu_0 a^2 b \beta_{10}} \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} E_z(\bar{R}') \sin \left[\frac{\pi}{a}(x_{00} + \xi') \right] e^{-j\beta_{10}\zeta'} d\xi' d\zeta' + \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} E_x(\bar{R}') \cos \left[\frac{\pi}{a}(x_0 + \xi') \right] e^{-j\beta_{10}\zeta'} d\zeta' d\xi' \right] \quad (1.48)$$

A forward scattered field can be defined similar to eq. 1.45

$$H_x^{f,scat}(\bar{R}) = -C_{10} \frac{\beta_{10}}{\pi/a} \sin \frac{\pi(x_{00} + \xi)}{a} e^{-j\beta_{10}\zeta} \quad (1.49)$$

and eq. 1.47 is rewritten as

$$H_x^{f,scat}(\bar{R}) = \frac{-A_{10}}{\omega\mu_0 ab} \sin \left[\frac{\pi}{a}(x_{00} + \xi) \right] e^{-j\beta_{10}\zeta} \quad (1.50)$$

$$\begin{aligned} & \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} E_z(\bar{R}') \sin \left[\frac{\pi}{a} (x_{00} + \xi') \right] e^{j\beta_{10}\zeta'} d\xi' d\zeta' \right. \\ & \left. - \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} E_x(\bar{R}') \cos \left[\frac{\pi}{a} (x_0 + \xi') \right] e^{j\beta_{10}\zeta'} d\xi' d\zeta' \right] \end{aligned}$$

with the field point to the right of the slot ($\zeta > L_z$). Again, the branch of the square root in eq. 1.27 is chosen so that its imaginary part is positive. Then equating eq. 1.49 and 1.50 gives the forward scattering coefficient

$$\begin{aligned} \frac{C_{10}}{A_{10}} = & \frac{\pi}{\omega\mu_0 a^2 b \beta_{10}} \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} E_z(\bar{R}') \sin \left[\frac{\pi}{a} (x_{00} + \xi') \right] e^{j\beta_{10}\zeta'} d\xi' d\zeta' \right. \\ & \left. - \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} E_x(\bar{R}') \cos \left[\frac{\pi}{a} (x_0 + \xi') \right] e^{j\beta_{10}\zeta'} d\xi' d\zeta' \right] \end{aligned} \quad (1.51)$$

1.5. Radiated Field

The radiation or distant field can easily be evaluated from sources of finite extent. In this section we shall outline a procedure for calculation of fields in the radiation zone.

We consider a distribution of magnetic currents in the vicinity of the coordinate origin, immersed in a homogeneous region of infinite extent. The complete solution can be found in terms of the eq. 1.7 and eq. 1.22. If we specialize to the radiation zone ($r \gg r'_{max}$), we have

$$|\bar{r} - \bar{r}'| = r - r' \cos \psi \quad (1.52)$$

where \bar{r} and \bar{r}' are the radius coordinate in a conventional spherical coordinate system. ψ is the angle between \bar{r} and \bar{r}' . Furthermore, the second term of eq. 1.52 can be neglected in the magnitude factors, $|\bar{r} - \bar{r}'|^{-1}$, of eq. 1.22. However, it cannot be neglected in the phase factors, $\exp(-jk|\bar{r} - \bar{r}'|)$. Thus eq. 1.22 reduces to

$$\mathbf{F} = \frac{e^{-jkr}}{4\pi r} \int \mathbf{M}(\bar{r}') e^{jkr' \cos \psi} dS' \quad (1.53)$$

in the radiation zone. The distant field of a current element is essentially outward-travelling plane waves [9]. Hence the radiation zone is characterized by

$$E_\theta = Z_0 H_\phi, \quad E_\phi = -Z_0 H_\theta \quad (1.54)$$

since it is a superposition of the fields from many current elements. Since we only have magnetic sources, the \mathbf{E} field due to \mathbf{M} can be evaluated according to $\mathbf{E} = -\nabla \times \mathbf{F}$. Retaining only the dominant terms (r^{-1} variation), we have

$$E_\theta = -(\nabla \times \mathbf{F})_\theta = -jkF_\phi \quad (1.55a)$$

$$E_\phi = -(\nabla \times \mathbf{F})_\phi = jkF_\theta \quad (1.55b)$$

with \mathbf{H} given by eq. 1.54. Thus no differentiation of the vector potential is necessary to obtain the radiation field. Finally we need to determine $r' \cos \psi$ as a function of source coordinates. With the sources located in a rectangular system, we form

$$rr' \cos \psi = \bar{r} \cdot \bar{r}' = xx' + yy' + zz \quad (1.56)$$

Substituting for x, y, z , we obtain

$$r' \cos \psi = (x' \cos \phi + y' \sin \phi) \sin \theta + z' \cos \theta \quad (1.57)$$

which is the desired form when rectangular coordinates are chosen for the source.

1.5.1. Half Space Environment

In case the slot radiates into a half space the vector potential is evaluated directly according to eq. 1.53. With a known electric field in the slot aperture the equivalent magnetic currents are found from eq. 1.4a and 1.4b and the radiated fields are subsequently found by evaluation of eq. 1.55a and 1.55b.

1.5.2. PP Waveguide

So far we have assumed the PP waveguide having infinite extent in the y -direction as well as the z -direction. However, in the following we shall outline a method to calculate the radiated field from PP waveguide with a finite height b' as shown in figure 1.2(c). For an approximate solution to this problem for $y > b'$, we assume \mathbf{E} in the aperture to be of the form of the incident field, cf. sect. 4-11 in [9]. From this field we form an equivalent magnetic aperture current $\mathbf{M}^a = -2\mathbf{n} \times \mathbf{E}$. Applying this magnetic aperture current to eq. 1.53, the vector potential \mathbf{F} is found and the radiated fields are subsequently found by evaluation of eq. 1.55a and 1.55b.

To obtain the electric field inside the PP waveguide, we turn to Maxwell's second

equation, eq. 1.1b, and eq. 1.35

$$\mathbf{E} = \frac{\nabla \times \mathbf{H}}{j\omega\epsilon} = 2 \int (\nabla \times \overline{\overline{G}}_{e2}) \cdot [\mathbf{n} \times \mathbf{E}] dS' \quad (1.58)$$

where $\overline{\overline{G}}_{e2}$ is given by eq. 1.42. Evaluating the curl operator according to eq. 1.43 [7]

$$\nabla \times \overline{\overline{G}} = \sum_i^3 \sum_j^3 (\nabla G_{ij} \times \hat{x}_i) \hat{x}_j \quad (1.59)$$

where $(x_1, x_2, x_3) = (x, y, z)$, a new dyadic Green's function can be written as

$$\begin{aligned} \overline{\overline{G}} = \nabla \times \overline{\overline{G}}_{e2} = & \left[\hat{x}\hat{x} \left[\frac{\partial}{\partial y} G_{e2,zx} - \frac{\partial}{\partial z} G_{e2,yx} \right] \right. \\ & + \hat{y}\hat{x} \left[\frac{\partial}{\partial z} G_{e2,xx} - \frac{\partial}{\partial x} G_{e2,zx} \right] \\ & + \hat{z}\hat{x} \left[\frac{\partial}{\partial x} G_{e2,yx} - \frac{\partial}{\partial y} G_{e2,xx} \right] \\ & + \hat{x}\hat{z} \left[\frac{\partial}{\partial y} G_{e2,zz} - \frac{\partial}{\partial z} G_{e2,yz} \right] \\ & + \hat{y}\hat{z} \left[\frac{\partial}{\partial z} G_{e2,xz} - \frac{\partial}{\partial x} G_{e2,zz} \right] \\ & \left. + \hat{z}\hat{z} \left[\frac{\partial}{\partial x} G_{e2,yz} - \frac{\partial}{\partial y} G_{e2,xz} \right] \right] \quad (1.60) \end{aligned}$$

which upon evaluation of the derivatives becomes

$$\begin{aligned} \overline{\overline{G}} = & \int_{-\infty}^{\infty} dk_z \sum_{m=0}^{\infty} \frac{-j\epsilon_m^2}{2\pi a' k_y k^2} e^{-jk_y(y-y')} e^{-jk_z(z-z')} \\ & \cdot \left[-\hat{y}\hat{x} j k_0^2 k_z \sin(k_x x) \sin(k_x x') \right. \\ & + \hat{z}\hat{x} j k_0^2 k_y \sin(k_x x) \sin(k_x x') \\ & + \hat{x}\hat{z} (k_z^2(1+j) - jk^2) k_y \cos(k_x x) \cos(k_x x') \\ & + \hat{y}\hat{x} k_x k_0^2 \sin(k_x x) \cos(k_x x') \\ & \left. + \hat{z}\hat{z} (j-1) k_x k_y k_z \sin(k_x x) \cos(k_x x') \right] \quad (1.61) \end{aligned}$$

Extracting terms giving x and z -directed fields from eq. 1.61 and applying this result to eq. 1.58 we obtain the following expressions

$$E_x^{ppwg}(\bar{R}) = -2 \int_{slot} G_{xz}(\bar{R}, \bar{R}') E_x(\bar{R}') dS' \quad (1.62a)$$

$$E_z^{ppwg}(\bar{R}) = 2 \int_{slot} G_{zx}(\bar{R}, \bar{R}') E_z(\bar{R}') - G_{zz}(\bar{R}, \bar{R}') E_x(\bar{R}') dS' \quad (1.62b)$$

Insertion of $\bar{\bar{G}}$ gives

$$E_x^{ppwg}(\bar{R}) = \frac{1}{\pi a'} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k^2} (k_z^2(j-1) + k^2) \cos(k_x x) e^{-jk_y y} e^{-jk_z z} \quad (1.63a)$$

$$\cdot \int_{slot} E_x(\bar{R}') \cos(k_x x') e^{jk_z z'} dS' dk_z$$

$$E_z^{ppwg}(\bar{R}) = \frac{1}{\pi a'} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \epsilon_m^2 \sin(k_x x) e^{-jk_y y} e^{-jk_z z} \quad (1.63b)$$

$$\cdot \left[\int_{slot} E_z(\bar{R}') \sin(k_x x') e^{jk_z z'} dS' \right.$$

$$\left. - \frac{(1+j)k_x k_z}{k^2} \int_{slot} E_x(\bar{R}') \cos(k_x x') e^{jk_z z'} dS' \right] dk_z$$

Assuming the electric field \mathbf{E} in the aperture to be of the form of the incident field we form an equivalent magnetic aperture current

$$\mathbf{M}^a = -2\mathbf{n} \times \mathbf{E}^{ppwg}(x=x', y=b', z=z') \quad (1.64)$$

making it possible to calculate the vector potential given in eq. 1.53. However, this three-dimensional problem having cylindrical boundaries can be reduced to a two-dimensional problem by applying a Fourier transformation with respect to z . According to sect. 5-11 [9] the radiation field is simply related to the transform of the source evaluated at $k_z = -k \cos \theta$. Equation 1.53 then becomes

$$\mathbf{F} = \frac{e^{-jk_r}}{4\pi r} \int_0^{a'} \widetilde{\mathbf{M}}^a(x', k_z = -k \cos \theta) e^{jk_x x' \cos \phi \sin \theta} dx' \quad (1.65)$$

where

$$\widetilde{\mathbf{M}}^a(x', k_z = -k \cos \theta) = -2\mathbf{n} \times \widetilde{\mathbf{E}}^{ppwg}(x = x', y = b', k_z = -k \cos \theta) \quad (1.66)$$

and the radiated fields are subsequently found by evaluation of eq. 1.55a and 1.55b.

Grating Lobes

A broadside beam from an array of T-slots requires one guide wavelength between consecutive slots. This results in two grating lobes of considerable magnitude at $\pm 45^\circ$ from the main beam.

In order to eliminate the grating lobes we apply a PP waveguide as described in sect. 1.5.2. For baffles of finite height the grating lobes will be attenuated by an amount given by the decay rate of the evanescent field. The y -directed propagation factor corresponding to the evanescent field is

$$k_y = -j \sqrt{k_x^2 - k^2 \sin^2 \theta} \quad (1.67)$$

This gives a simple expression for the attenuation versus baffle height b'

$$GLA = 20 \log_{10} e^{-jk_y b'} \quad (1.68)$$

The grating lobe attenuation effect will also be seen in the element patterns. Compared to the omnidirectional pattern from a magnetic dipole radiating into free space, the E-plane will be quite narrow as a consequence of the spatial filtering of the baffles. The cut-off angle is $\theta_c = \arcsin(k_x/k)$, which is the limiting angle for total reflection at the free space-parallel plate interface.

Chapter 2

Numerical Computation Methods

The purpose of this chapter is to describe the concept of the method of moments and apply it to the slot problem derived in chap. 1. Furthermore, integration concepts and schemes necessary to overcome some numerical problems arising from using the method of moments are described in detail.

2.1. Method of Moments

The Method of Moments (MoM) as a numerical procedure is outlined in figure 2.1 [10]. The major steps consist of the formulation of an appropriate integral equation (IE) for the problem considered, the discretization process, and the solution of the discretized integral equation. These steps will be discussed in detail in the following sections.

2.1.1. The Integral Equation

The moment method is preferred in the numerical solution mainly due to its formulation by the integral equation. The integral equation reduces the domain of the operator equation to a finite one, with all the boundary conditions implicitly contained. We have already seen formulation of the integral equation in sect. 1.2 eq. 1.3a and 1.3a. It replaces all the equations, from eq. 1.1a to 1.2b as well as eq. 1.4a to 1.5b. Instead of having to satisfy the Maxwell equations in the entire infinite space it is now only necessary to satisfy the integral equation on the surface confined to the slot aperture.

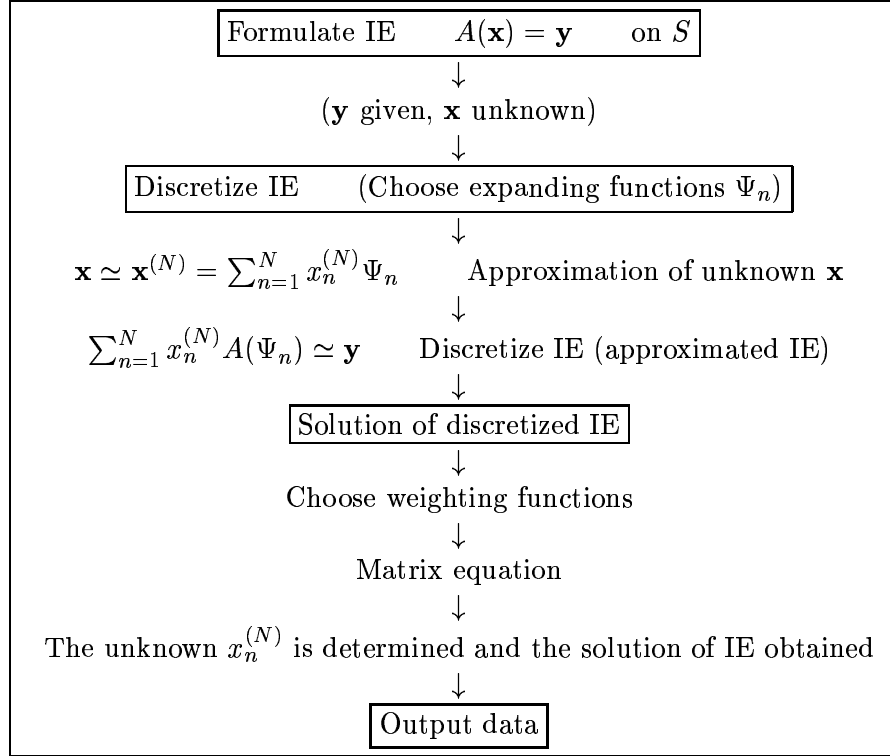


Figure 2.1.: Outline of the Method of Moments.

2.1.2. Discretization of the Unknown Function

A continuous integral equation, such as eq. 1.3a and 1.3a, can be written in an abbreviated form as follows

$$A(\mathbf{x}) = \mathbf{y} \quad \text{on } S \tag{2.1}$$

where \mathbf{x} denotes the unknown, which is \mathbf{E} in eq. 1.3a and 1.3a, and \mathbf{y} denotes the given, which is \mathbf{H}^{inc} . In terms of linear algebra, we call A an operator. \mathbf{y} and \mathbf{x} are complex vectorial functions in a linear space \mathcal{L} , on which A operates.

The first step in solving a linear operator equation such as eq. 2.1 is to project the operator onto a finite subspace, or equivalently, approximate the unknown function by a discretization process. We approximate the unknown \mathbf{x} in a finite linear space \mathcal{L}^N

of dimension N by $\mathbf{x}^{(N)}$ as

$$\mathbf{x} \simeq \mathbf{x}^{(N)} = \sum_{n=1}^N x_n^{(N)} \Psi_n \quad (2.2)$$

where $x_n^{(N)}$ are complex constants, and Ψ_n is a set of linearly independent functions called expansion functions. For an exact solution, N must in general be infinite. Therefore $\mathbf{x}^{(N)}$ is an approximation for the unknown function \mathbf{x} .

2.1.3. Solution of the Discretized Operator Equation

Substituting eq. 2.2 into eq. 2.1 and making use of the linearity of the operator A , we obtain the discretized integral equation as follows

$$\sum_{n=1}^N x_n^{(N)} A(\Psi_n) \simeq \mathbf{y} \quad \text{on } S \quad (2.3)$$

This equation is an ill-defined statement. The approximate sign is the problem, but necessary because the left and right sides cannot be exactly equal in the entire domain S . Additional weighted measures are needed to make this equation meaningful.

2.1.4. The Direct MoM for Solving an Operator Equation

The direct MoM is the method defined by *Harrington* [11], who outlined a basic principle to implement weighted measures to give precise definitions to the discretized integral equation, eq. 2.3, which is not properly defined. The method consist of choosing N weighting functions and then taking an inner product on eq. 2.3 with each of the weighting functions, resulting in N precisely defined linear equations. A set of equations, which are numerically solved by marix methods for N unknowns. The details are as follows.

We choose a inner product defined as the convolution

$$\langle \mathbf{x}, \mathbf{y} \rangle = \int_S \mathbf{x} \cdot \mathbf{y} dS \quad (2.4)$$

Next we select a set of linearly independent weighting functions w_1, w_2, \dots, w_N in the spatial domain S and take the inner product on eq. 2.3

$$\sum_{n=1}^N x_n^{(N)} \langle A(\Psi_n), w_m \rangle = \langle \mathbf{y}, w_m \rangle \quad m = 1, 2, \dots, N \quad (2.5)$$

We have thus transformed the imprecise eq. 2.3 into a set of precisely defined equations. This set of equations can be written in matrix form as

$$[A_{mn}] [x_n^{(N)}] = [y_m] \quad (2.6)$$

where

$$[A_{mn}] \begin{bmatrix} \langle A(\Psi_1), w_1 \rangle & \langle A(\Psi_2), w_1 \rangle & \cdots & \langle A(\Psi_N), w_1 \rangle \\ \langle A(\Psi_1), w_2 \rangle & \langle A(\Psi_2), w_2 \rangle & \cdots & \langle A(\Psi_N), w_2 \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle A(\Psi_1), w_N \rangle & \langle A(\Psi_2), w_N \rangle & \cdots & \langle A(\Psi_N), w_N \rangle \end{bmatrix}, \quad (2.7)$$

$$[x_n^{(N)}] = \begin{bmatrix} x_1^{(N)} \\ x_2^{(N)} \\ \vdots \\ x_N^{(N)} \end{bmatrix}, \quad [y_m] = \begin{bmatrix} \langle y, w_1 \rangle \\ \langle y, w_2 \rangle \\ \vdots \\ \langle y, w_N \rangle \end{bmatrix}$$

The unknown column matrix $[x_n^{(N)}]$ is then

$$[x_n^{(N)}] = [A_{mn}^{-1}] [y_m] \quad (2.8)$$

where $[A_{mn}^{-1}]$ is the inverse matrix. Note that by choosing various sets of weighting functions, we can have many different sets of matrix equations, and therefore, different results for the numerical solution of a particular integral equation.

2.2. Application of the MoM to Slot Problems

As mentioned in sect. 1.1 the unknown electric fields in the longitudinal and the transverse slot apertures - E_x and E_z - are assumed to be constant across the respective slot. As a part of the method of moments the electric fields are expanded using global sinusoidal functions as follows

$$E_x(\vec{R}') = \sum_{p=1}^{N_z} E_p e_p(\zeta') \quad (2.9a)$$

$$E_z(\vec{R}') = \sum_{r=1}^{N_x} E_r e_r(\xi') \quad (2.9b)$$

where E_p and E_r are the unknown coefficients and the expanding functions are

$$e_p(\zeta') = \sin \frac{p\pi}{2L_z}(\zeta' + L_z) \quad (2.10a)$$

$$e_r(\xi') = \sin \frac{r\pi}{2L_x}(\xi' + L_x) \quad (2.10b)$$

with the local coordinates centered in the slot apertures. These basis functions have been applied successfully in similar problems by *Lars Josefsson* [2][3]. Furthermore, we notice the sinusoidal basis functions automatically satisfy the boundary conditions at the ends of the slots, where $\mathbf{n} \times \mathbf{E} = 0$ applies.

The six terms in eq. 1.3a and eq. 1.3b are now defined, and we have an integral equation where the unknown variables E_x and E_z appear in the integrands. This equation is now transformed into a matrix equation according to the the method of moments described in sect. 2.1. We use the Galekin's method with the following entire domain weighting functions

$$w_q(\xi, \zeta) = \sin \left[\frac{q\pi}{2L_z}(\zeta + L_z) \right] \delta(\xi) \quad q = 1, 2, 3, \dots, N_z \quad (2.11a)$$

$$w_s(\xi, \zeta) = \sin \left[\frac{s\pi}{2L_x}(\xi + L_x) \right] \delta(\zeta) \quad s = 1, 2, 3, \dots, N_x \quad (2.11b)$$

and inner products defined as in eq. 2.4

$$\langle H, w \rangle = \int_{slot} H(P)w(P)ds \quad (2.12)$$

It gives us $(N_z + N_x) \times (N_z + N_x)$ equations which in a matrix form can be expressed as

$$\begin{bmatrix} [Y_{pq}^{ext} - Y_{pq}^{int}] & [Y_{rq}^{ext} - Y_{rq}^{int}] \\ [Y_{ps}^{ext} - Y_{ps}^{int}] & [Y_{rs}^{ext} - Y_{rs}^{int}] \end{bmatrix} \begin{bmatrix} [E_p] \\ [E_r] \end{bmatrix} = \begin{bmatrix} [h_q^{inc}] \\ [h_s^{inc}] \end{bmatrix} \quad (2.13)$$

where $[E_p]$ and $[E_r]$ are the unknown column matrixes

$$[E_p] = (E_{x,1}, E_{x,2}, \dots, E_{x,N_z})^T \quad (2.14)$$

$$[E_r] = (E_{z,1}, E_{z,2}, \dots, E_{z,N_x})^T \quad (2.15)$$

The matrix elements are defined as eq. 2.12 and normalized with Z_0 and the electric

field according to

$$Y_{pq}^{ext} - Y_{pq}^{int} = (\langle H_z^{ext}(E_x), w_q \rangle - \langle H_z^{int}(E_x), w_q \rangle) \frac{Z_0}{E_p} \quad (2.16a)$$

$$Y_{rq}^{ext} - Y_{rq}^{int} = (\langle H_z^{ext}(E_z), w_q \rangle - \langle H_z^{int}(E_z), w_q \rangle) \frac{Z_0}{E_r} \quad (2.16b)$$

$$Y_{ps}^{ext} - Y_{ps}^{int} = (\langle H_x^{ext}(E_x), w_s \rangle - \langle H_x^{int}(E_x), w_s \rangle) \frac{Z_0}{E_p} \quad (2.16c)$$

$$Y_{rs}^{ext} - Y_{rs}^{int} = (\langle H_x^{ext}(E_z), w_s \rangle - \langle H_x^{int}(E_z), w_s \rangle) \frac{Z_0}{E_r} \quad (2.16d)$$

The right hand side of eq. 2.13 is a column matrix representing the excitation from the incident TE_{10} wave, with elements

$$h_q^{inc} = \langle H_z^{inc}, w_q \rangle \quad (2.17a)$$

$$h_s^{inc} = \langle H_x^{inc}, w_s \rangle \quad (2.17b)$$

where H_z^{inc} and H_x^{inc} are given in eq. 1.28 and eq. 1.31

Once the matrix elements have been evaluated the solution is obtained after the matrix inversion

$$[E] = [Y^{ext} - Y^{int}]^{-1} [h^{inc}] \quad (2.18)$$

2.2.1. A T-slot Radiating into a Half Space Environment

A full derivation of the matrix elements given in sect. 2.2 is found in appendix C. However, for convenience the final results are given in the following sections.

Self Terms

For the longitudinal slot, the external terms Y_{pq}^{ext} becomes

$$Y_{pq}^{ext} = \frac{2Z_0}{j\pi^2\omega\mu_0(q^2 - p^2)} \left[[(k2L_z)^2 - (q\pi)^2] \cdot p \cdot y_1(q) \right. \\ \left. - [(k2L_z)^2 - (p\pi)^2] \cdot q \cdot y_1(p) \right] \quad p \neq q \quad (2.19)$$

and

$$Y_{pq}^{ext} = \frac{Z_0}{j\omega\pi\mu_0} [(k2L_z)^2 - (q\pi)^2] \cdot y_2(q) - \frac{2Z_0q}{j\omega\mu_0} \cdot y_1(q) \quad p = q \quad (2.20)$$

where the two functions $y_1(p)$ and $y_2(p)$ are given by

$$y_1(p) = \int_0^{w/2} \int_0^{1/2} \sin[p\pi(\sigma + 1/2)] \cdot \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\sigma d\xi' \quad (2.21)$$

and

$$y_2(p) = \int_0^{w/2} \int_0^1 \frac{e^{-jkR}}{R} \left[(1 - \nu) \cos(p\pi\nu) + \frac{\sin(p\pi\nu)}{p\pi} \right] d\nu d\xi' \quad (2.22)$$

The three quantities R , R_1 and R_2 are given by

$$R = \sqrt{\xi'^2 + \nu^2 4L_z^2} \quad (2.23a)$$

$$R_1 = \sqrt{\xi'^2 + (\sigma - 1/2)^2 4L_z^2} \quad (2.23b)$$

$$R_2 = \sqrt{\xi'^2 + (\sigma + 1/2)^2 4L_z^2} \quad (2.23c)$$

When p is odd and q is even or vice versa $Y_{pq}^{ext} = 0$. Furthermore, it is noticed that the matrix is symmetric, which should be utilized to speed up computations.

The internal terms for the longitudinal slot becomes

$$Y_{pq}^{int} = \frac{2j\lambda w}{ab\pi} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 W_m \left[\frac{\frac{p\pi}{2L_z}}{\gamma_{mn}} \frac{\gamma_{mn}^2 + k^2}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2} \frac{\frac{q\pi}{2L_z}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2} \right. \quad (2.24)$$

$$\left. \cdot (1 \pm e^{-2\gamma_{mn}L_z}) + \frac{k^2 - \left(\frac{p\pi}{2L_z}\right)^2}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2} L_z \delta_{pq} \right]$$

where $\delta_{pq} = 1$ for $p = q$, 0 otherwise. The upper sign is valid for both p and q odd, the lower sign for both p and q even. When p is odd and q is even or vice versa $Y_{pq}^{int} = 0$. Like the exterior terms, $Y_{pq}^{int} = 0$ when p is odd and q is even or vice versa. The symmetry properties of Y^{ext} are valid to Y^{int} too.

In case we have a transverse slot, the contribution from the exterior is given by

$$Y_{rs}^{ext} = \frac{-2Z_0}{j\pi^2\omega\mu_0(s^2 - r^2)} \left[[(k2L_x)^2 - (s\pi)^2] \cdot r \cdot y_1(s) - [(k2L_z)^2 - (r\pi)^2] \cdot s \cdot y_1(r) \right] \quad r \neq s \quad (2.25)$$

and

$$Y_{rs}^{ext} = \frac{-Z_0}{j\omega\pi\mu_0} [(k2L_x)^2 - (s\pi)^2] \cdot y_2(s) + \frac{2Z_0s}{j\omega\mu_0} y_1(s) \quad r = s \quad (2.26)$$

where the two functions $y_1(r)$ and $y_2(r)$ are given by

$$y_1(r) = \int_0^{w/2} \int_0^{1/2} \sin[\pi r(\sigma + 1/2)] \cdot \left[(-1)^r \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\sigma d\zeta' \quad (2.27)$$

and

$$y_2(r) = \int_0^{w/2} \int_0^1 \frac{e^{-jkR}}{R} \left[\cos(r\pi\nu)(1 - \nu) + \frac{\sin(r\pi\nu)}{r\pi} \right] d\nu d\zeta' \quad (2.28)$$

Similarly the three quantities R , R_1 and R_2 are now written as

$$R = \sqrt{\zeta'^2 + \nu^2 4L_x^2} \quad (2.29a)$$

$$R_1 = \sqrt{\zeta'^2 + (\sigma - 1/2)^2 4L_x^2} \quad (2.29b)$$

$$R_2 = \sqrt{\zeta'^2 + (\sigma + 1/2)^2 4L_x^2} \quad (2.29c)$$

Furthermore, $Y_{rs}^{ext} = 0$ when r is odd and s is even or vice versa. The symmetry properties applies to Y_{rs}^{ext} as well.

Finally, the interior contributions for a transverse slot Y_{rs}^{int} are given by

$$Y_{rs}^{int} = \frac{-4Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}^2} \frac{\frac{s\pi}{L_x} \frac{r\pi}{L_x} \left[\left(\frac{m\pi}{a} \right)^2 - k^2 \right] [1 - e^{-\gamma_{mn}w/2}]}{\left[\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2 \right] \left[\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2 \right]} \cdot \begin{cases} \left[\cos^2 \frac{m\pi L_x}{a} \sin^2 \frac{m\pi x_{00}}{a} \right] & r, s \text{ odd} \\ \left[\cos^2 \frac{m\pi x_{00}}{a} \sin^2 \frac{m\pi L_x}{a} \right] & r, s \text{ even} \end{cases}$$

Like Y_{pq}^{int} , Y_{rs}^{int} is symmetric and $Y_{rs}^{int} = 0$ when r is odd and s is even or vice versa.

Cross Coupling Terms

The exterior coupling from a longitudinal slot to a transverse is given by

$$Y_{ps}^{ext} = \frac{-Z_0 p}{4j\omega\mu_0 L_z} \int_{-L_x}^{L_x} \int_{-L_z}^{L_z} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \cdot \cos \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \left[\frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\zeta' d\xi \quad (2.30)$$

where

$$R_1 = \sqrt{(L_x - \xi)^2 + \zeta'^2} \quad (2.31a)$$

$$R_2 = \sqrt{(L_x - w - \xi)^2 + \zeta'^2} \quad (2.31b)$$

and the interior coupling is

$$Y_{ps}^{int} = \frac{-4Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{\frac{p\pi}{L_z} \frac{s\pi}{L_x} \sin \frac{m\pi w}{2a} \cos \frac{m\pi x_0}{a}}{\left[\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z} \right)^2 \right] \left[\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2 \right]} \cdot \left[e^{-\gamma_{mn} L_z} - \cos \frac{p\pi}{2} \right] \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} s \text{ odd} \\ s \text{ even} \end{array} \quad (2.32)$$

where $Y_{ps}^{int} = 0$ when p is odd.

The exterior coupling from a transverse slot to a longitudinal is given by

$$Y_{rq}^{ext} = \frac{Z_0 r}{4j\omega\mu_0 L_x} \int_{-L_z}^{L_z} \int_{-L_x}^{L_x} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \cdot \cos \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \left[\frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\xi' d\zeta \quad (2.33)$$

where

$$R_1 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 - \zeta)^2} \quad (2.34a)$$

$$R_2 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 + \zeta)^2} \quad (2.34b)$$

and the interior coupling is obtained as a sum of two terms

$$Y_{rq}^{int} = Y_{rq}^{int,1} + Y_{rq}^{int,23} \quad (2.35)$$

where $Y_{rq}^{int,1}$ is given by

$$\begin{aligned} Y_{rq}^{int,1} = & \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2 \right]} \\ & \cdot \frac{\frac{1}{L_z} \cos \frac{q\pi}{2}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z} \right)^2 \right]} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r \text{ odd} \\ r \text{ even} \end{array} \\ & \cdot \left[2\gamma_{mn} L_z \sin \frac{q\pi w}{4L_z} [e^{-\gamma_{mn} w} + 1] + \cos \frac{q\pi w}{4L_z} [e^{-\gamma_{mn} w} - 1] \right] \end{aligned} \quad (2.36)$$

and $Y_{rq}^{int,23}$ is given by

$$\begin{aligned} Y_{rq}^{int,23} = & \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2 \right]} \\ & \cdot \frac{\frac{1}{L_z}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z} \right)^2 \right]} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r \text{ odd} \\ r \text{ even} \end{array} \\ & \cdot \left[\cos \frac{q\pi}{2} [1 - e^{-\gamma_{mn} w}] \left[q\pi \cos \frac{q\pi w}{4L_z} + 2\gamma_{mn} L_z \sin \frac{q\pi w}{4L_z} \right] \right. \\ & \left. + q\pi [e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)}] \right] \end{aligned} \quad (2.37)$$

$Y_{rq}^{int,1} = Y_{rq}^{int,23} = 0$ when q is odd.

Excitation Terms

Finally the forcing terms are given by

$$h_q^{inc} = \frac{\frac{q\pi}{L_z}}{\left(\frac{q\pi}{2L_z} \right)^2 - \beta_{10}^2} \cos \frac{\pi x_0}{a} \cdot \begin{cases} j \cos(\beta_{10} L_z) & q \text{ odd} \\ - \sin(\beta_{10} L_z) & q \text{ even} \end{cases} \quad (2.38)$$

$$h_s^{inc} = \frac{-\beta_{10}}{\pi/a} \frac{\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2} \cdot \begin{cases} \cos \frac{\pi L_x}{a} \sin \frac{\pi x_{00}}{a} & s \text{ odd} \\ -\cos \frac{\pi x_{00}}{a} \sin \frac{\pi L_x}{a} & s \text{ even} \end{cases} \quad (2.39)$$

2.2.2. A T-slot Radiating into an Infinite PP Waveguide

In case the exterior region consists of a parallel plate (PP) waveguide, the exterior contribution to the matrix elements given in sect. 2.2 changes. However, all other terms remain unchanged.

Self Terms

For the longitudinal slot, the external term Y_{pq}^{ext} is found by weighting the magnetic field $H_z^{ext}(E_z)$ with the weight function w_q . That is by applying eq. 2.11a and 2.16a to eq. 1.44b

$$\begin{aligned} Y_{pq}^{ext} &= \langle H_z^{ext}(E_x), w_q \rangle \frac{Z_0}{E_p} \quad (2.40) \\ &= \frac{-Z_0}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 (k^2 - k_z^2)}{k_y} \\ &\quad \cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \cos [k_x(x_{0,pp} + \xi)] e^{-jk_z \zeta} \\ &\quad \cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \cos [k_x(x_{0,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta' d\xi d\zeta dk_z \end{aligned}$$

where $x_{0,pp} = x_0 - (a - a_1)/2$. Using eq. C.3 and C.65 the ξ' and ζ' integrals reduces to

$$\begin{aligned} Y_{pq}^{ext} &= \frac{-Z_0 w p}{a' \omega \mu_0 L_z} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 (k^2 - k_z^2)}{k_y} \quad (2.41) \\ &\quad \cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \cos [k_x(x_{0,pp} + \xi)] e^{-jk_z \zeta} \\ &\quad \cdot \frac{\text{sinc} \frac{k_x w}{2} \cos(k_x x_{0,pp})}{\left(\frac{p\pi}{2L_z}\right)^2 - k_z^2} \begin{bmatrix} \cos(k_z L_z) \\ -j \sin(k_z L_z) \end{bmatrix} \begin{matrix} p \text{ odd} \\ p \text{ even} \end{matrix} d\xi d\zeta dk_z \end{aligned}$$

Applying eq. C.65 in the evaluation of the ζ we obtain the following final expression

$$Y_{pq}^{ext} = \frac{-Z_0 w p q \pi}{a' \omega \mu_0 L_z^2} \sum_{m=0}^{\infty} \epsilon_m^2 \operatorname{sinc} \frac{k_x w}{2} \cos^2(k_x x_{0,pp}) \quad (2.42)$$

$$\cdot \int_0^{\infty} \frac{(k^2 - k_z^2)}{k_y \left[\left(\frac{p\pi}{2L_z} \right)^2 - k_z^2 \right] \left[\left(\frac{q\pi}{2L_z} \right)^2 - k_z^2 \right]} \begin{cases} p, q \text{ odd} \\ p, q \text{ even} \end{cases} \begin{bmatrix} 1 + \cos(2k_z L_z) \\ 1 - \cos(2k_z L_z) \end{bmatrix} dk_z$$

where we have made use of the fact that the integrand is an even function in k_z and rewritten squared trigonometric functions in terms of the double angle. The same properties regarding symmetry and zero-terms apply as we saw in the half space case.

In case of a transverse slot, the external terms Y_{rs}^{ext} are found by applying eq. 2.11b and 2.16d to eq. 1.44a

$$Y_{rs}^{ext} = \langle H_x^{ext}(E_z), w_q \rangle \frac{Z_0}{E_r} \quad (2.43)$$

$$= \frac{Z_0}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 (k^2 - k_x^2)}{k_y}$$

$$\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \delta(\zeta) \sin [k_x (x_{00,pp} + \xi)] e^{-jk_z \zeta}$$

$$\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \sin [k_x (x_{00,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta' d\xi d\zeta dk_z$$

where $x_{00,pp} = x_{00} - (a - a_1)/2$. Using eq. C.20 the ξ' and ζ' integrals reduces to

$$Y_{rs}^{ext} = \frac{Z_0 w r}{a' \omega \mu_0 L_x} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 (k^2 - k_x^2)}{k_y} \quad (2.44)$$

$$\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \delta(\zeta) \sin [k_x (x_{00,pp} + \xi)] e^{-jk_z \zeta}$$

$$\cdot \frac{\operatorname{sinc} \frac{k_z w}{2}}{\left(\frac{r\pi}{2L_x} \right)^2 - k_x^2} \begin{bmatrix} \cos(k_x L_x) \sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp}) \sin(k_x L_x) \end{bmatrix} \begin{matrix} r \text{ odd} \\ r \text{ even} \end{matrix} d\xi d\zeta dk_z$$

Applying eq. C.20 once more in the evaluation of the ξ integral we obtain the following

final expression

$$\begin{aligned}
 Y_{rs}^{ext} &= \frac{2Z_0 w r s \pi}{a' \omega \mu_0 L_x^2} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 (k^2 - k_x^2)}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - k_z^2 \right] \left[\left(\frac{s\pi}{2L_x} \right)^2 - k_z^2 \right]} \\
 &\cdot \begin{cases} \left[\cos^2(k_x L_x) \sin^2(k_x x_{00,pp}) \right]_{r,s \text{ odd}} \\ \left[\cos^2(k_x x_{00,pp}) \sin^2(k_x L_x) \right]_{r,s \text{ even}} \end{cases} \int_0^{\infty} \frac{\text{sinc} \frac{k_z w}{2}}{k_y} dk_z
 \end{aligned} \tag{2.45}$$

Again, the same symmetry and zero-term properties apply as we saw in the half space case.

Cross Coupling Terms

The exterior coupling from a longitudinal to a transverse slot Y_{ps}^{ext} is found by weighting the magnetic field $H_x^{ext}(E_x)$ with the weight function w_s

$$\begin{aligned}
 Y_{ps}^{ext} &= \langle H_x^{ext}(E_x), w_s \rangle \frac{Z_0}{E_p} \\
 &= \frac{-j Z_0}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x k_z}{k_y} \\
 &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \delta(\zeta) \sin [k_x (x_{00,pp} + \xi)] e^{-jk_z \zeta} \\
 &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \cos [k_x (x_{0,pp} + \zeta')] e^{jk_z \zeta'} d\xi' d\zeta' d\xi d\zeta dk_z
 \end{aligned} \tag{2.46}$$

The integrals in ξ' and ζ' are almost identical with the expressions we saw in eq. 2.40 and evaluates to

$$\begin{aligned}
 Y_{ps}^{ext} &= \frac{-j 2 Z_0 p}{a' \omega \mu_0 L_z} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_z}{k_y} \\
 &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \delta(\zeta) \sin [k_x (x_{00,pp} + \xi)] e^{-jk_z \zeta} \\
 &\cdot \frac{\sin \frac{k_z w}{2} \cos(k_x x_{0,pp})}{\left[\left(\frac{p\pi}{2L_z} \right)^2 - k_z^2 \right]} \begin{cases} \left[\cos(k_z L_z) \right]_{p \text{ odd}} \\ \left[-j \sin(k_z L_z) \right]_{p \text{ even}} \end{cases} d\xi d\zeta dk_z
 \end{aligned} \tag{2.47}$$

Using eq. C.20 in the evaluation of the ξ integral we obtain

$$\begin{aligned}
 Y_{ps}^{ext} &= \frac{-4Z_0 p s \pi}{a' \omega \mu_0 L_z L_x} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 \sin \frac{k_x w}{2} \cos(k_x x_{0,pp})}{\left[\left(\frac{s\pi}{2L_x} \right)^2 - k_x^2 \right]} \\
 &\cdot \left[\begin{array}{l} \cos(k_x L_x) \sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp}) \sin(k_x L_x) \end{array} \right]_{s \text{ even}}^{s \text{ odd}} \cdot \int_0^{\infty} \frac{k_z \sin(k_z L_z)}{k_y \left[\left(\frac{p\pi}{2L_z} \right)^2 - k_z^2 \right]} dk_z
 \end{aligned} \tag{2.48}$$

where $Y_{ps}^{ext} = 0$ when p is odd.

Similarly the coupling from a transverse to a longitudinal slot is found by

$$\begin{aligned}
 Y_{rq}^{ext} &= \langle H_z^{ext}(E_z), w_q \rangle \frac{Z_0}{E_r} \\
 &= \frac{-j Z_0}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x k_z}{k_y} \\
 &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \cos [k_x (x_{0,pp} + \xi)] e^{-jk_z \zeta} \\
 &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \sin [k_x (x_{00,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta' d\xi d\zeta dk_z
 \end{aligned} \tag{2.49}$$

Again, applying eq. C.20 in the evaluation of the ξ' integral we obtain

$$\begin{aligned}
 Y_{rq}^{ext} &= \frac{-j 2 Z_0 r}{a' \omega \mu_0 L_x} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x}{k_y} \\
 &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \cos [k_x (x_{0,pp} + \xi)] e^{-jk_z \zeta} \\
 &\cdot \frac{\sin \frac{k_z w}{2}}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - k_x^2 \right]} \left[\begin{array}{l} \cos(k_x L_x) \sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp}) \sin(k_x L_x) \end{array} \right]_{r \text{ even}}^{r \text{ odd}} d\xi d\zeta dk_z
 \end{aligned} \tag{2.50}$$

Evaluating the last integral yields

$$\begin{aligned}
 Y_{rq}^{ext} = & \frac{4Z_0 r q \pi}{a' \omega \mu_0 L_x L_z} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x \cos(k_x x_{0,pp})}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - k_x^2 \right]} \\
 & \cdot \left[\begin{array}{l} \cos(k_x L_x) \sin(k_x x_{0,pp}) \\ -\cos(k_x x_{0,pp}) \sin(k_x L_x) \end{array} \right]_{r \text{ even}}^{r \text{ odd}} \\
 & \cdot \int_0^{\infty} \frac{\sin \frac{k_z w}{2} \sin(k_z L_z)}{k_y \left[\left(\frac{q\pi}{2L_z} \right)^2 - k_z^2 \right]} dk_z
 \end{aligned} \tag{2.51}$$

where $Y_{rq}^{ext} = 0$ when q is odd.

2.2.3. Reduction Scheme for Triple Integrals

In the derivation of the expression for the free space contributions Y_{pq}^{ext} and Y_{rs}^{ext} we need to consider the reduction of a computationally less efficient triple integral. All details have been left out in sect. 2.2.1 but the problem is addressed in appendix C.1 and C.2. The object is to reduce these expressions to numerically more efficient double integrals. This step involves several variable substitutions and application of an integration scheme. Here we shall outline a reduction scheme described in [12] for Y_{pq}^{ext} . However, with suitable substitutions it applies directly to Y_{rs}^{ext} .

Let's consider the following integral

$$I_{pq} = \int_{-w/2}^{w/2} [I_1 + I_2] d\xi' \tag{2.52}$$

where

$$\begin{aligned}
 I_1 = & \int_{-L_z}^{L_z} \int_{-L_z}^{L_z} \left[k^2 - \left(\frac{p\pi}{2L_z} \right)^2 \right] \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \\
 & \cdot \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \frac{e^{-jkR}}{R} d\zeta' d\zeta
 \end{aligned} \tag{2.53}$$

and

$$I_2 = -\frac{p\pi}{2L_z} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\zeta \tag{2.54}$$

Making the substitutions $\sigma = \zeta/2L$ and $\sigma' = \zeta'/2L$, eq. 2.53 and 2.54 becomes

$$I_1 = [(2L_z k)^2 - (p\pi)^2] \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} \sin [q\pi(\sigma + 1/2)] \cdot \sin [p\pi(\sigma' + 1/2)] \frac{e^{-jkR}}{R} d\sigma' d\sigma \quad (2.55)$$

and

$$I_2 = -p\pi \int_{-1/2}^{1/2} \sin [q\pi(\sigma + 1/2)] \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\sigma \quad (2.56)$$

From the above

$$I_{pq} = \int_{-w/2}^{w/2} [[(2L_z k)^2 - (p\pi)^2] i_1 - p\pi i_2] d\xi' \quad (2.57)$$

We shall find simplify i_1 and i_2 to obtain numerically more efficient more expressions.

Integration Scheme for i_1

From eq. 2.55 and 2.57 we have

$$i_1 = \int_{-1/2}^{1/2} \int_{-1/2}^{1/2} \sin [q\pi(\sigma + 1/2)] \sin [p\pi(\sigma' + 1/2)] \frac{e^{-jkR}}{R} d\sigma' d\sigma \quad (2.58)$$

where

$$R = \sqrt{\xi'^2 + (\sigma - \sigma')^2 4L_z^2} \quad (2.59)$$

The integration is now carried out on strips of constant ν , $\sigma - \sigma'$ as shown in figure 2.2

$$i_1 = \int_{-1}^0 \frac{e^{-jkR}}{R} \int_{-1/2}^{1/2+\nu} \sin [q\pi(\sigma + 1/2)] \sin [p\pi(\sigma - \nu + 1/2)] d\sigma d\nu \quad (2.60)$$

$$+ \int_0^1 \frac{e^{-jkR}}{R} \int_{-1/2+\nu}^{1/2} \sin [q\pi(\sigma + 1/2)] \sin [p\pi(\sigma - \nu + 1/2)] d\sigma d\nu$$

which after evaluation of inner integrals reduces to

$$i_1 = \int_0^1 \frac{e^{-jkR}}{R} \left[(1 - \nu) \cos(q\pi\nu) + \frac{\sin(q\pi\nu)}{q\pi} \right] d\nu \quad p = q \quad (2.61a)$$

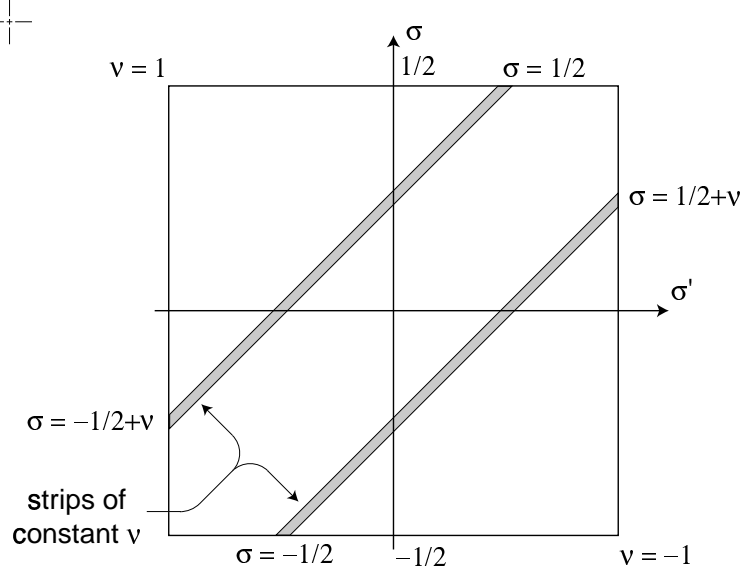


Figure 2.2.: Integration on strips of constant ν , $\sigma - \sigma'$.

$$\begin{aligned}
 &= \int_0^1 \frac{e^{-jkR}}{R} \frac{2}{(q^2 - p^2)\pi} [q \sin(p\pi\nu) - p \sin(q\pi\nu)] d\nu \quad p \neq q, p + q \text{ even} \\
 &= 0 \quad p + q \text{ odd}
 \end{aligned}$$

Thus we see from eq. 2.56, 2.57 and 2.61a that I has been reduced from a triple integral to a double integral, which is computationally more desirable.

Equation 2.57 can be further simplified by making use of the fact that the matrix elements are symmetric, $Y_{pq}^{ext} = Y_{qp}^{ext}$. Thus the simplified expression for I_{pq} when $p \neq q$ becomes

$$\begin{aligned}
 I_{pq} = \frac{4}{\pi(q^2 - p^2)} &\left[[(k2L_z)^2 - (q\pi)^2] \cdot p \cdot y_1(q) \right. \\
 &\left. - [(k2L_z)^2 - (p\pi)^2] \cdot q \cdot y_1(p) \right] \quad p \neq q
 \end{aligned} \tag{2.62}$$

and

$$I_{pq} = 2 [(k2L_z)^2 - (q\pi)^2] y_2(q) - 4q\pi y_1(q) \quad p = q \tag{2.63}$$

where the two functions $y_1(p)$ and $y_2(p)$ are given by

$$y_1(p) = \int_0^{w/2} \int_0^{1/2} \sin[q\pi(\sigma + 1/2)] \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\sigma d\xi' \quad (2.64)$$

and

$$y_2(p) = \int_0^{w/2} \int_0^1 \frac{e^{-jkR}}{R} \left[(1 - \nu) \cos(p\pi\nu) + \frac{\sin(p\pi\nu)}{p\pi} \right] d\nu d\xi' \quad (2.65)$$

R , R_1 and R_2 are given by

$$R = \sqrt{\xi'^2 + \nu^2 4L_z^2} \quad (2.66a)$$

$$R_1 = \sqrt{\xi'^2 + (\sigma - 1/2)^2 4L_z^2} \quad (2.66b)$$

$$R_2 = \sqrt{\xi'^2 + (\sigma + 1/2)^2 4L_z^2} \quad (2.66c)$$

2.2.4. 1/R Singularity Extraction

Many of the integrals arising in computation of exterior contribution to the H-field have kernels which are in some sense unpleasant. This is caused by a singularity either in the integrand or a singular-like behaviour in a low-order derivative of the integrand.

In all cases the singular behaviour arises from the $1/R$ term, which we can factor out of the integrand, in the sense that the problem may be formulated as

$$\int_S \frac{1}{R} f(\xi, \zeta) dS \quad (2.67)$$

where R becomes zero at some point (ξ_0, ζ_0) , which is within the integration limits. $f(\xi, \zeta)$ is a slowly varying function.

This singularity can be subtracted out. Suppose that the singularity occurs at the single point (ξ_0, ζ_0) , eq. 2.67 is rewritten in the form

$$\begin{aligned} & \int_S \frac{1}{R} [f(\xi, \zeta) - f(\xi_0, \zeta_0)] dS + f(\xi_0, \zeta_0) \int_S \frac{1}{R} dS \\ &= \int_S \frac{1}{R} g(\xi, \zeta) dS + f(\xi_0, \zeta_0) \int_S \frac{1}{R} dS \end{aligned} \quad (2.68)$$

Provided that the last integral can be evaluated either analytically or by a simple numerical method, eq. 2.67 can be estimated by evaluating the first integral. This

is easier than the original problem because $g(\xi_0, \zeta_0) = 0$. However, the singularity has not completely disappeared. The new integral has a singular first derivative so that we have weakened, rather than removed, the difficulty. The subtraction process can however be repeated, successively if required, to weaken the singularity further. Including the first derivative we obtain

$$\int_S \frac{1}{R} \left[f(\xi, \zeta) - f(\xi_0, \zeta_0) - (\xi - \xi_0) \frac{\partial f}{\partial \xi} \Big|_{\xi_0, \zeta_0} - (\zeta - \zeta_0) \frac{\partial f}{\partial \zeta} \Big|_{\xi_0, \zeta_0} \right] dS \quad (2.69)$$

$$+ f(\xi_0, \zeta_0) \cdot \int_S \frac{1}{R} dS + \frac{\partial f}{\partial \xi} \Big|_{\xi_0, \zeta_0} \cdot \int_S \frac{\xi - \xi_0}{R} dS + \frac{\partial f}{\partial \zeta} \Big|_{\xi_0, \zeta_0} \cdot \int_S \frac{\zeta - \zeta_0}{R} dS$$

Moving the origin of the coordinate system (ξ, ζ) to the singular point (ξ_0, ζ_0) and transforming the problem into polar coordinates, the last three integrals in eq. 2.70 become

$$\iint \left[f(\xi_0, \zeta_0) + \frac{\partial f}{\partial \xi} \Big|_{\xi_0, \zeta_0} R \sin \theta + \frac{\partial f}{\partial \zeta} \Big|_{\xi_0, \zeta_0} R \cos \theta \right] d\theta dR \quad (2.70)$$

However, since the problem is not well suited for this choice of coordinate system, we will put a little more effort in describing an integration scheme for this specific problem. First of all we will limit the region of interest to one of the four quadrants in the corresponding cartesian coordinate system. The scheme is successively extended to cover the entire region by simple summation of the contribution from the four quadrants. The area confined to $[0, \xi_1] \times [0, \zeta_1]$ is separated into three regions as

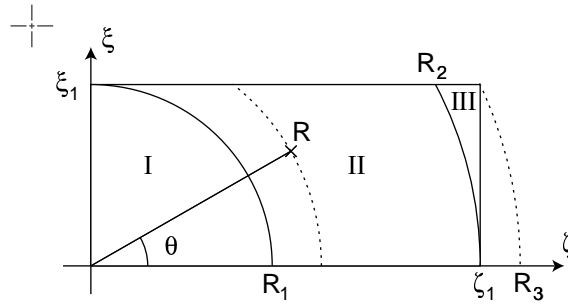


Figure 2.3.: Integration over a rectangular region in polar coordinates.

shown in figure 2.3. The three quantities R_1 , R_2 and R_3 are defined as

$$R_1 = \min(|\xi_1|, |\zeta_1|)$$

$$R_2 = \max(|\xi_1|, |\zeta_1|)$$

$$R_3 = \sqrt{\xi_1^2 + \zeta_1^2}$$

Evaluating the θ integrals for constant values of R , the contribution from region one becomes

$$I_1 = \int_0^{R_1} \frac{\pi}{2} f(\xi_0, \zeta_0) + \operatorname{sgn}(\xi_1) R \left. \frac{\partial f}{\partial \xi} \right|_{\xi_0, \zeta_0} + \operatorname{sgn}(\zeta_1) R \left. \frac{\partial f}{\partial \zeta} \right|_{\xi_0, \zeta_0} dR \quad (2.71)$$

In case $|\zeta_1| > |\xi_1|$ the contribution from region two becomes

$$\begin{aligned} I_2 = & \int_{R_1}^{R_2} f(\xi_0, \zeta_0) \arcsin \frac{R_1}{R} - \operatorname{sgn}(\xi_1) R \cos \left[\arcsin \frac{R_1}{R} - 1 \right] \left. \frac{\partial f}{\partial \xi} \right|_{\xi_0, \zeta_0} \\ & + \operatorname{sgn}(\zeta_1) R \sin \left[\arcsin \frac{R_1}{R} - 1 \right] \left. \frac{\partial f}{\partial \zeta} \right|_{\xi_0, \zeta_0} dR \end{aligned} \quad (2.72)$$

whereas if $|\zeta_1| \leq |\xi_1|$ the contribution becomes

$$\begin{aligned} I_2 = & \int_{R_1}^{R_2} \left[\frac{\pi}{2} - \arccos \frac{R_1}{R} \right] f(\xi_0, \zeta_0) + \operatorname{sgn}(\xi_1) R \cos \left[\arccos \frac{R_1}{R} \right] \left. \frac{\partial f}{\partial \xi} \right|_{\xi_0, \zeta_0} \\ & + \operatorname{sgn}(\zeta_1) R \left[1 - \sin \left[\arccos \frac{R_1}{R} \right] \right] \left. \frac{\partial f}{\partial \zeta} \right|_{\xi_0, \zeta_0} dR \end{aligned} \quad (2.73)$$

Similarly, the contribution from region three becomes

$$\begin{aligned} I_3 = & \int_{R_2}^{R_3} \left[\arcsin \frac{R_1}{R} - \arccos \frac{R_2}{R} \right] f(\xi_0, \zeta_0) \\ & - \operatorname{sgn}(\xi_1) R \left[\cos \left[\arcsin \frac{R_1}{R} \right] - \cos \left[\arccos \frac{R_2}{R} \right] \right] \left. \frac{\partial f}{\partial \xi} \right|_{\xi_0, \zeta_0} \\ & + \operatorname{sgn}(\zeta_1) R \left[\sin \left[\arcsin \frac{R_1}{R} \right] - \sin \left[\arccos \frac{R_2}{R} \right] \right] \left. \frac{\partial f}{\partial \zeta} \right|_{\xi_0, \zeta_0} dR \end{aligned} \quad (2.74)$$

in case $|\zeta_1| > |\xi_1|$ and

$$I_3 = \int_{R_2}^{R_3} \left[\arcsin \frac{R_2}{R} - \arccos \frac{R_1}{R} \right] f(\xi_0, \zeta_0) \quad (2.75)$$

$$\begin{aligned}
 & -\operatorname{sgn}(\xi_1)R \left[\cos \left[\arcsin \frac{R_2}{R} \right] - \cos \left[\arccos \frac{R_1}{R} \right] \right] \frac{\partial f}{\partial \xi} \Big|_{\xi_0, \zeta_0} \\
 & +\operatorname{sgn}(\zeta_1)R \left[\sin \left[\arcsin \frac{R_2}{R} \right] - \sin \left[\arccos \frac{R_1}{R} \right] \right] \frac{\partial f}{\partial \zeta} \Big|_{\xi_0, \zeta_0} dR
 \end{aligned}$$

if $|\zeta_1| \leq |\xi_1|$.

In order to show the strength of the method the following two kernels have been extracted from eq. C.62

$$I_{R_1} = \frac{1}{R_1} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \cos \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \cos(kR_1) \quad (2.76)$$

$$I_{R_2} = \frac{1}{R_2} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \cos \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \cos(kR_2) \quad (2.77)$$

both being singular at $\xi = L_x$ and $\xi = L_x - w$ respectively. Applying the method outlined these two integrands have been plotted in figure 2.4 and 2.5 for $p = 4$ and $s = 6$. These results shows that the integrals containing a $1/R$ kernel can be

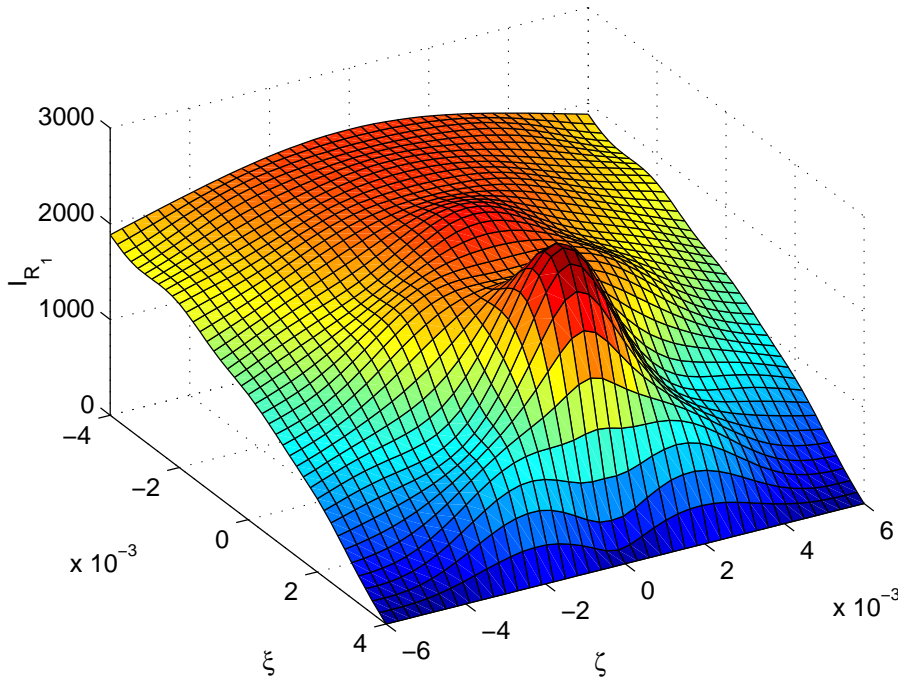


Figure 2.4.: Value of the kernel I_{R_1} .

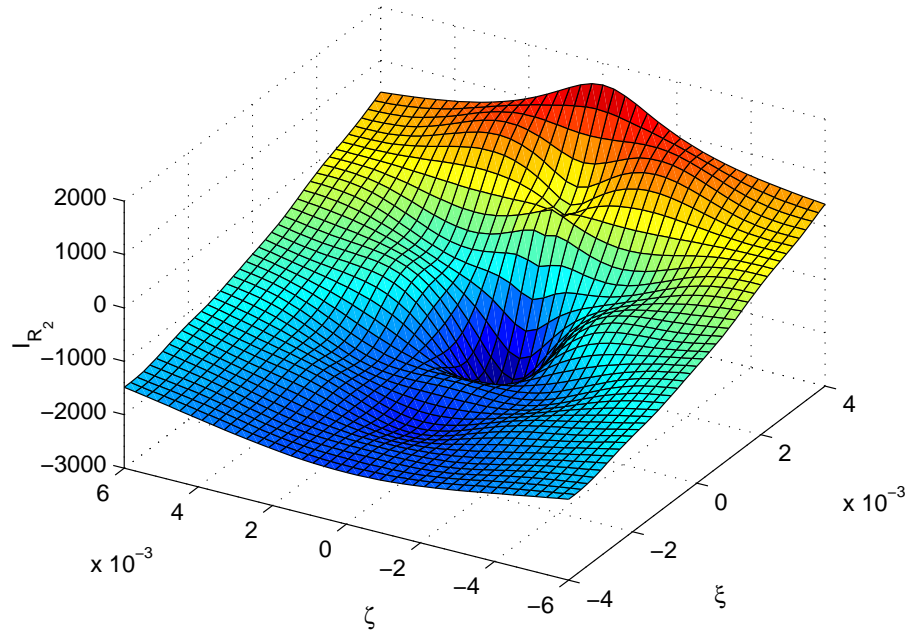


Figure 2.5.: Value of the kernel I_{R_2} .

transformed into well behaved integrals. These are much more smooth and can be evaluated to high accuracy with standard numerical integration algorithms available in mathematical libraries such as *IMSL* or *NAG*.

2.2.5. Deformed Contour Integral

To efficiently evaluate the spectral integrals of the type given in eq. 1.44a and eq. 1.44b we consider the integrals of the following type

$$I = \int_0^{\infty} f(k_z) dk_z \quad (2.78)$$

where the kernel $f(k_z)$ has a singularity at $k_y = 0$, i.e. when $k_z = \sqrt{k_0^2 - k_x^2}$. Secondly, the integral has an infinite upper limit and the kernel is an oscillating function in k_z .

There are different techniques to handle the singularity problem. One way to handle the singularity is to evaluate the integral along a revised or deformed contour [13]. In figure 2.6 the contour has been moved into the complex plane of k_z and in the vicinity of the poles a deformed contour along a half circle is used. The axes of the half circle is chosen large enough to confine the poles within the circle. The numerical

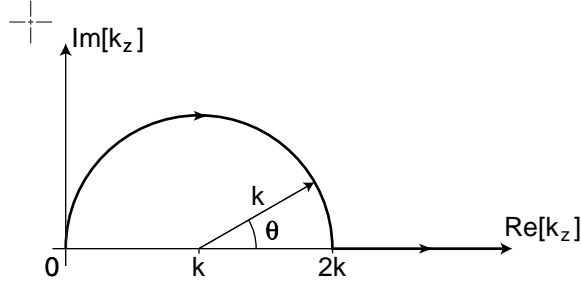


Figure 2.6.: Integration contour in the complex k_z plane.

integration of the integral given in eq. 2.78 is performed along the deformed contour over $0 \leq k_z \leq 2k$, and along the k_z axes for $2k \leq k_z \leq \infty$

$$I = \int_0^\pi f(k[(1 - \cos \theta) + j \sin \theta]) k [\sin \theta + j \cos \theta] d\theta + \int_{2k}^\infty f(k_z) dk_z \quad (2.79)$$

The advantage of using a deformed contour along a half circle is that there is no need for supplementary information concerning the type and location of the singularities. The problem solving the integral along the k_z axes for $2k \leq k_z \leq \infty$ is addressed in sect. 2.2.6.

2.2.6. Method of Weighted Averages

Sommerfeld integrals, as given in e.g. eq. 2.79 can be grouped in a more general class of integrals defined by

$$I(k) = \int_a^\infty g(k\lambda) f(\lambda) d\lambda \quad (2.80)$$

where

- $g(k\lambda)$ is a complex function whose real and imaginary parts oscillate with a strictly periodic behavior or behave asymptotically as the product of a periodic function and monotonic function.
- $f(\lambda)$ is a smooth, non-oscillating function that behaves asymptotically as $C\lambda^\alpha$. Above a certain value of the argument λ , the function $f(\lambda)$ and all its derivatives have a constant sign. For the sake of simplicity, $f(\lambda)$ is assumed to be real. Complex functions can be handled by working successively with their real and imaginary parts.

- The lower integration bound a has been chosen conveniently to ensure that the interval $[a, \infty]$ is far from possible singularities of $f(\lambda)$.

A technique used to evaluate eq. 2.80 is defined by [14]

$$I(k) = \sum_{m=0}^{\infty} \int_{a+mp/2}^{a+(m+1)p/2} g(k\lambda)f(\lambda)d\lambda \quad (2.81)$$

The integration over each half-cycle is performed prior to the summation. In practice, the infinite integration must obviously be bounded and we apply a simple technique based on the concept of weighted average between the half cycle integrals. This accelerating device has proven to be faster and very accurate for Sommerfeld integrals.

For clarity we shall show the algorithm for the kernel $g(k\lambda) = \cos(k\lambda)$. Partial values are calculated numerically, defining $I_m^1(m = 1, 2, \dots, M)$ as [15]

$$I_m^1 = \int_a^{\lambda_m} \cos(k\lambda)f(\lambda)d\lambda, \quad m = 1, 2, \dots, M \quad (2.82)$$

where the λ_m are successive zeros of the oscillatory function $\cos(k\lambda)$ beyond the integration boundary a . The variation between the real value of I and the approximations I_m^1 is given by the value of the integral over $[\lambda_m, \infty]$. This value can be estimated, dividing the interval into an infinite number of subintervals, each having as its width one period of $\cos(k\lambda)$. Using the conditions described at the beginning of this section, the function f can be replaced by the first terms of its Taylor series with respect to the center of the subinterval. The following result is obtained in this way by simultaneously integrating by parts

$$I - I_m^1 \simeq -f(\lambda_m) \sin(k\lambda_m)/k \quad (2.83)$$

where the term $\sin(k\lambda_m)$ takes alternative values $+1$ and -1 . The sequence of M values I_m^1 thus oscillates back and forth around the real value I , its convergence being determined by the function $f(\lambda)$. A new sequence $I_m^2(m = 1, 2, \dots, M - 1)$ is then defined taking the averages of two consecutive values of the sequence I_m^1 , following the general expression

$$I_m^{l+1} = 1/2(I_m^l + I_{m+1}^l), \quad l = 1, \dots, M - 1, \quad m = 1, \dots, M - l \quad (2.84)$$

Use of eq. 2.83 shows that

$$I - I_m^2 \simeq (\pi/2k^2) f'(\lambda_m) \sin(k\lambda_m) \quad (2.85)$$

where the I_m^2 sequence also has an oscillatory behavior around I , its convergence depending on the first derivative. Subsequent use of the average relation eq. 2.83 produces new sequences I_m^l . Taking into account the asymptotic behavior of $f(\lambda)$, the sequence I_m^l with $l > \alpha + 1$ is the first one that will converge toward the real value of I . Successive sequences converge faster each time. The last sequence reduces to a single value I_1^M which will be closer to the true value than I_M^1 in spite of the fact no additional evaluations of the integrand have been required.

Noticing that the variation between every term of the I_m^1 sequence and the exact value I depends on m in eq. 2.83, it is apparent that the averaging algorithm can be improved. The arithmetical mean eq. 2.84 can be replaced by a weighted average, in which more weight is given to the value of I_m^1 closest to I . The general expression is

$$I_m^{l+1} = (w_m^l I_m^l + w_{m+1}^l I_{m+1}^l) / (w_m^l + w_{m+1}^l), \quad \begin{cases} l = 1, \dots, M-1, \\ m = 1, \dots, M-l \end{cases} \quad (2.86)$$

For $l = 1$, the optimal value of weights is

$$w_m^1 = (\lambda_1/\lambda_m)^\alpha \quad (2.87)$$

where, as mentioned previously, α is the asymptotic exponent of the function $f(\lambda)$. Since, at each successive averaging iteration, the order of the function controlling the convergence decreases by one unit, eq. 2.87 can be generalized to

$$w_m^l = (\lambda_1/\lambda_m)^{\alpha+1-l} \quad (2.88)$$

2.3. Scattered Field

Knowing the field distribution in the slot aperture, the scattering coefficients given in eq. 1.48 and eq. 1.51 can be evaluated. Inserting the expansion terms introduced in sect. 2.2 we obtain the backward scattering coefficient

$$\frac{B_{10}}{A_{10}} = \frac{-\pi}{\omega\mu_0 a^2 b \beta_{10}} \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} \sum_{r=1}^{N_x} E_r \sin \left[\frac{r\pi}{2L_x} (\xi' + L_x) \right] \right] \quad (2.89)$$

$$\begin{aligned} & \cdot \sin \left[\frac{\pi}{a}(x_{00} + \xi') \right] e^{-j\beta_{10}\zeta'} d\xi' d\zeta' + \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} \sum_{p=1}^{N_z} E_p \\ & \cdot \cos \left[\frac{\pi}{a}(x_0 + \xi') \right] \sin \left[\frac{p\pi}{2L_z}(\zeta' + L_z) \right] e^{-j\beta_{10}\zeta'} d\zeta' d\xi' \end{aligned}$$

Evaluation of the integrals yields

$$\begin{aligned} \frac{B_{10}}{A_{10}} = & \frac{-2\pi^2}{\omega\mu_0 a^2 b \beta_{10}} \left[\sin(w/2\beta_{10}) \sum_{r=1}^{N_x} E_r \frac{\frac{r}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2} \right. \\ & \cdot \left[\begin{array}{l} \cos \frac{\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{\pi x_{00}}{a} \sin \frac{\pi L_x}{a} \end{array} \right]_{r \text{ even}}^{r \text{ odd}} + j \sin \frac{\pi w}{2a} \cos \frac{\pi x_0}{a} \\ & \cdot \left. \sum_{p=1}^{N_z} E_p \frac{\frac{p}{L_z}}{\left(\frac{p\pi}{2L_z}\right)^2 - \beta_{10}^2} \cdot \left[\begin{array}{l} \cos(\beta_{10}L_z) \\ j \sin(\beta_{10}L_z) \end{array} \right]_{p \text{ even}}^{p \text{ odd}} \right] \end{aligned} \quad (2.90)$$

Similarly the forward scattering coefficient becomes

$$\begin{aligned} \frac{C_{10}}{A_{10}} = & \frac{\pi}{\omega\mu_0 a^2 b \beta_{10}} \left[\int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \beta_{10} \sum_{r=1}^{N_x} E_r \sin \left[\frac{r\pi}{2L_x}(\xi' + L_x) \right] \right. \\ & \cdot \sin \left[\frac{\pi}{a}(x_{00} + \xi') \right] e^{j\beta_{10}\zeta'} d\xi' d\zeta' - \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \frac{j\pi}{a} \sum_{p=1}^{N_z} E_p \\ & \cdot \cos \left[\frac{\pi}{a}(x_0 + \xi') \right] \sin \left[\frac{p\pi}{2L_z}(\zeta' + L_z) \right] e^{j\beta_{10}\zeta'} d\zeta' d\xi' \end{aligned} \quad (2.91)$$

and by evaluation of integrals we obtain

$$\begin{aligned} \frac{C_{10}}{A_{10}} = & \frac{2\pi^2}{\omega\mu_0 a^2 b \beta_{10}} \left[\sin(w/2\beta_{10}) \sum_{r=1}^{N_x} E_r \frac{\frac{r}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2} \right. \\ & \cdot \left[\begin{array}{l} \cos \frac{\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{\pi x_{00}}{a} \sin \frac{\pi L_x}{a} \end{array} \right]_{r \text{ even}}^{r \text{ odd}} - j \sin \frac{\pi w}{2a} \cos \frac{\pi x_0}{a} \\ & \cdot \left. \sum_{p=1}^{N_z} E_p \frac{\frac{p}{L_z}}{\left(\frac{p\pi}{2L_z}\right)^2 - \beta_{10}^2} \cdot \left[\begin{array}{l} \cos(\beta_{10}L_z) \\ j \sin(\beta_{10}L_z) \end{array} \right]_{p \text{ even}}^{p \text{ odd}} \right] \end{aligned} \quad (2.92)$$

$$\cdot \sum_{p=1}^{N_z} E_p \frac{\frac{p}{L_z}}{\left(\frac{p\pi}{2L_z}\right)^2 - \beta_{10}^2} \cdot \begin{bmatrix} \cos(\beta_{10}L_z) \\ -j \sin(\beta_{10}L_z) \end{bmatrix} \begin{matrix} p \text{ odd} \\ p \text{ even} \end{matrix}$$

2.4. Radiated Field

In the following we shall derive expressions for the electric fields in the radiation zone. Again we need to distinguish between a slot radiating into half space or a parallel plate waveguide. However, with a known aperture current distribution, the radiation fields can be evaluated according to eq. 1.55a and 1.55b which require knowledge of the vector potentials.

2.4.1. Half Space Environment

The vector potential is obtained from eq. 1.53. With the magnetic current defined by eq. 1.4b the x -component becomes

$$F_x = -\frac{e^{-jkr}}{2\pi r} \int E_z(\bar{r}') e^{jk r' \cos \psi} dS' \quad (2.93)$$

Using eq. 2.9b and 1.57 we obtain

$$F_x = -\frac{e^{-jkr}}{2\pi r} \sum_{r=1}^{N_x} E_r \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (\xi' + L_x) \right] e^{jk\xi' \cos \phi \sin \theta + jk\zeta' \cos \theta} d\xi' d\zeta' \quad (2.94)$$

Applying the integral formula eq. C.65 this is reduces to the following closed form expression

$$F_x = -\frac{e^{-jkr}}{r} \frac{w e^{jk(w/2-L_x) \cos \phi \sin \theta}}{2L_x} \text{sinc} \left[\frac{kw}{2} \cos \theta \right] \cdot \sum_{r=1}^{N_x} \frac{E_r r}{\left(\frac{r\pi}{2L_x}\right)^2 - k^2 \cos^2 \phi \sin^2 \theta} \cdot \begin{bmatrix} \cos(kL_x \cos \phi \sin \theta) \\ -j \sin(kL_x \cos \phi \sin \theta) \end{bmatrix} \begin{matrix} r \text{ odd} \\ r \text{ even} \end{matrix} \quad (2.95)$$

where the extra phase term $\exp(jk(w/2 - L_x) \cos \phi \sin \theta)$ has been included to account for a coordinate transformation. Similarly the z -component becomes

$$F_z = \frac{e^{-jkr}}{2\pi r} \int E_x(\bar{r}') e^{jkr' \cos \psi} dS' \quad (2.96)$$

Inserting expressions for the electric field yields

$$F_z = \frac{e^{-jkr}}{2\pi r} \sum_{p=1}^{N_z} E_p \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z} (\zeta' + L_z) \right] e^{jk\xi' \cos \phi \sin \theta + jk\zeta' \cos \theta} d\zeta' d\xi' \quad (2.97)$$

Again, using the integral formula eq. C.65 this expression reduces to

$$F_z = \frac{e^{-jkr}}{r} \frac{w}{2L_z} \text{sinc} \left[\frac{kw}{2} \cos \phi \sin \theta \right] \cdot \sum_{p=1}^{N_z} \frac{E_p p}{\left(\frac{p\pi}{2L_z} \right)^2 - k^2 \cos^2 \theta} \cdot \begin{bmatrix} \cos(kL_z \cos \theta) \\ -j \sin(kL_z \cos \theta) \end{bmatrix} \begin{matrix} p \text{ odd} \\ p \text{ even} \end{matrix} \quad (2.98)$$

In spherical coordinates, we obtain

$$F_\theta = F_x \cos \theta \cos \phi - F_z \sin \theta \quad (2.99a)$$

$$F_\phi = -F_x \sin \phi \quad (2.99b)$$

The electric fields in the radiation zone are then found from eq. 1.55a and 1.55b.

2.4.2. PP Waveguide

In this case the vector potential is obtained from eq. 1.65. With the magnetic current defined by eq. 1.66 the x -component becomes

$$F_x = \frac{e^{-jkr}}{4\pi r} \int_0^{a'} \widetilde{M}_x^a(x', k_z = -k \cos \theta) e^{jkx' \cos \phi \sin \theta} dx' \quad (2.100)$$

where the magnetic current \widetilde{M}_x^a is given by

$$\widetilde{M}_x^a(x', k_z = -k \cos \theta) = -2\widetilde{E}_z^{ppwg}(x = x', y = b', k_z = -k \cos \theta) \quad (2.101)$$

The electric field inside the PP waveguide, \tilde{E}_z^{ppwg} , is found from eq. 1.63b and with the expansion fields given in eq. 2.9a and 2.9b we obtain

$$\begin{aligned} \tilde{E}_z^{ppwg}(x, y, kz) &= \frac{1}{\pi a'} \sum_{m=0}^{\infty} \epsilon_m^2 \sin(k_x x) e^{-jk_y y} & (2.102) \\ &\cdot \left[\sum_{r=1}^{N_x} E_r \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \right. \\ &\cdot \sin [k_x (x_{00,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta' \\ &- \frac{(1+j)k_x k_z}{k^2} \sum_{p=1}^{N_z} E_p \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \\ &\left. \cdot \cos [k_x (x_{0,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta' \right] \end{aligned}$$

Using eq. C.3, C.20 and C.65 this expression reduces to

$$\begin{aligned} \tilde{E}_z^{ppwg}(x, y, kz) &= \frac{w}{\pi a'} \sum_{m=0}^{\infty} \epsilon_m^2 \sin(k_x x) e^{-jk_y y} \left[\operatorname{sinc} \left(\frac{k_z w}{2} \right) F_1(r, k_x) \right. & (2.103) \\ &\left. - \frac{(1+j)k_x k_z}{k^2} \operatorname{sinc} \left(\frac{k_x w}{2} \right) \cos(k_x x_{0,pp}) F_2(p, k_z) \right] \end{aligned}$$

where

$$F_1(r, k_x) = \sum_{r=1}^{N_x} \frac{E_r \frac{r\pi}{L_x}}{\left(\frac{r\pi}{2L_x} \right)^2 - k_x^2} \left[\begin{array}{l} \cos(k_x L_x) \sin(k_x x_{00,pp}) \\ - \cos(k_x x_{00,pp}) \sin(k_x L_x) \end{array} \right]_{r \text{ even}}^{r \text{ odd}} \quad (2.104)$$

$$F_2(p, k_z) = \sum_{p=1}^{N_z} \frac{E_p \frac{p\pi}{L_z}}{\left(\frac{p\pi}{2L_z} \right)^2 - k_z^2} \left[\begin{array}{l} \cos(k_z L_z) \\ -j \sin(k_z L_z) \end{array} \right]_{p \text{ even}}^{p \text{ odd}} \quad (2.105)$$

The vector potential F_x then becomes

$$F_x = \frac{e^{-jkr}}{4\pi r} \frac{2w}{\pi a'} \sum_{m=0}^{\infty} \epsilon_m^2 e^{-jk_y b'} \left[- \operatorname{sinc} \left(\frac{k_z w}{2} \right) F_1(r, k_x) \right] \quad (2.106)$$

$$\begin{aligned}
 & + \frac{(1+j)k_x k_z}{k^2} \operatorname{sinc}\left(\frac{k_x w}{2}\right) \cos(k_x x_{0,pp}) F_2(p, k_z) \Big] \\
 & \cdot \int_0^{a'} \sin(k_x x') e^{jk_x' \cos \phi \sin \theta} dx'
 \end{aligned}$$

Evaluating the last integral yields

$$\begin{aligned}
 F_x = & \frac{e^{-jkr}}{4\pi r} \frac{2w}{\pi a'} \sum_{m=0}^{\infty} \epsilon_m^2 e^{-jk_y b'} \left[-\operatorname{sinc}\left(\frac{k_z w}{2}\right) F_1(r, k_x) \right. \\
 & \left. + \frac{(1+j)k_x k_z}{k^2} \operatorname{sinc}\left(\frac{k_x w}{2}\right) \cos(k_x x_{0,pp}) F_2(p, k_z) \right] \\
 & \cdot \frac{k_x}{k^2 \cos^2 \phi \sin^2 \theta - k_x^2} \left[(-1)^m e^{jk_x a' \cos \phi \sin \theta} - 1 \right]
 \end{aligned} \tag{2.107}$$

Similarly the z -component of the vector potential becomes

$$F_z = \frac{e^{-jkr}}{4\pi r} \int_0^{a'} \widetilde{M}_z^a(x', k_z = -k \cos \theta) e^{jk_x' \cos \phi \sin \theta} dx' \tag{2.108}$$

where the magnetic current \widetilde{M}_z^a is given by

$$\widetilde{M}_z^a(x', k_z = -k \cos \theta) = 2\widetilde{E}_x^{ppwg}(x=x', y=b', k_z = -k \cos \theta) \tag{2.109}$$

The electric field \widetilde{E}_z^{ppwg} is found from eq. 1.63a and with the expansion fields given in eq. 2.9a we obtain

$$\begin{aligned}
 \widetilde{E}_x^{ppwg}(x, y, k_z) = & \frac{1}{\pi a'} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k^2} (jk_z^2 + (k^2 - k_z^2)) \cos(k_x x) e^{-jk_y y} \\
 & \cdot \sum_{p=1}^{N_z} E_p \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \\
 & \cdot \cos[k_x(x_{0,pp} + \xi')] e^{jk_z \zeta'} d\xi' d\zeta'
 \end{aligned} \tag{2.110}$$

Using eq. C.3 and C.65 this expression reduces to

$$\begin{aligned} \tilde{E}_x^{ppwg}(x, y, kz) &= \frac{w}{\pi a'} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k^2} (jk_z^2 + (k^2 - k_z^2)) \cos(k_x x) e^{-jk_y y} \\ &\quad \cdot \text{sinc}\left(\frac{k_x w}{2}\right) \cos(k_x x_{0,pp}) F_2(p, k_z) \end{aligned} \quad (2.111)$$

The vector potential F_z then becomes

$$\begin{aligned} F_z &= \frac{e^{-jkr}}{4\pi r} \frac{2w}{\pi a'} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k^2} (jk_z^2 + (k^2 - k_z^2)) \text{sinc}\left(\frac{k_x w}{2}\right) \cos(k_x x_{0,pp}) \\ &\quad \cdot e^{-jk_y y} F_2(p, k_z) \int_0^{a'} \cos(k_x x') e^{jk_x' \cos \phi \sin \theta} dx' \end{aligned} \quad (2.112)$$

and upon evaluating of x' integral we obtain

$$\begin{aligned} F_z &= \frac{e^{-jkr}}{4\pi r} \frac{2jw}{\pi a'} \sum_{m=0}^{\infty} \frac{\epsilon_m^2}{k^2} (jk_z^2 + (k^2 - k_z^2)) \text{sinc}\left(\frac{k_x w}{2}\right) \cos(k_x x_{0,pp}) \\ &\quad \cdot e^{-jk_y y} F_2(p, k_z) \frac{k \cos \phi \sin \theta}{k^2 \cos^2 \phi \sin^2 \theta - k_x^2} \left[1 - (-1)^m e^{jka' \cos \phi \sin \theta}\right] \end{aligned} \quad (2.113)$$

In spherical coordinates, the vector potentials are found according to eq. 2.99a and 2.99b and the electric fields in the radiation zone are then found from eq. 1.55a and 1.55b.

2.5. Finite Wall Thickness

A rectangular slot in a waveguide of thickness t can be seen as a cavity of length $a = 2L$, width $b = w$ and height $c = t$ connecting the rectangular waveguide and the exterior region. This geometry is shown in figure 2.7 where the previous defined coordinate system remains unchanged. As we shall see, this model is easily incorporated into the previous model for the zero wall case. Though the model outlined is not valid for a T-slot, we shall later use it for verification of results for longitudinal and transverse slots.

With the same assumption made in sect. 1.2 we impose the continuity of the tangential component of the magnetic field at the two interfaces. For a longitudinal

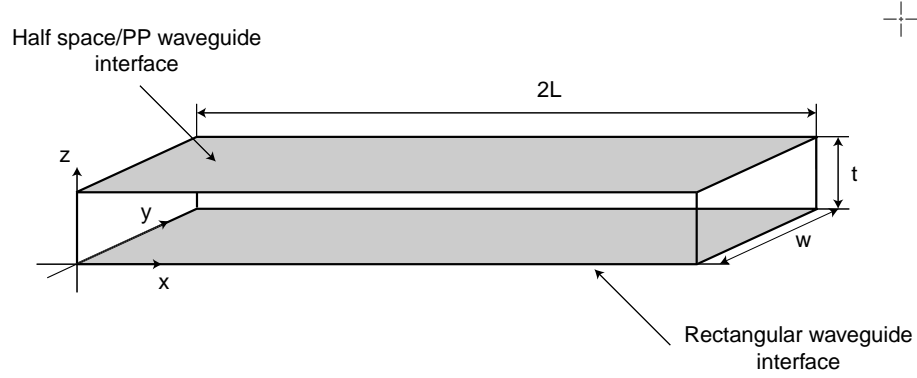


Figure 2.7.: Rectangular cavity representing the finite wall thickness.

slot of length $2L$ we obtain

$$H_z^{ext}(P) - H_z^{cav,upper}(P) = 0 \quad (2.114a)$$

$$H_z^{cav,lower}(P) - H_z^{int}(P) = H_z^{inc}(P) \quad (2.114b)$$

where *cav, lower* and *cav, upper* refer to fields inside the cavity at the lower and upper interface respectively. Other quantities are identical with those previously introduced.

The two terms $H_z^{cav,upper}$ and $H_z^{cav,lower}$ in the integral equation above are written in terms of a convolution of a cavity Green's function and the magnetic currents at the interfaces. Since we have two interfaces, the magnetic fields are obtained by superposition of such two convolutions

$$H_z^{cav,lower}(P) = \int_{slot} M_z^{lower}(P') G^{cav}(z'=0, z=0) ds' - \quad (2.115a)$$

$$\int_{slot} M_z^{upper}(P') G^{cav}(z'=t, z=0) ds' \quad (2.115b)$$

$$H_z^{cav,upper}(P) = \int_{slot} M_z^{upper}(P') G^{cav}(z'=t, z=t) ds' + \quad (2.115c)$$

$$\int_{slot} M_z^{lower}(P') G^{cav}(z'=0, z=t) ds' \quad (2.115d)$$

where the magnetic currents are written in terms of the unknown electric fields at the

lower and upper interfaces

$$M_z^{lower}(P') = \sum_{p=1}^N E_p^{lower} \sin \left[\frac{p\pi}{2L} (\zeta' + L) \right] \quad (2.116a)$$

$$M_z^{upper}(P') = \sum_{p=1}^N E_p^{upper} \sin \left[\frac{p\pi}{2L} (\zeta' + L) \right] \quad (2.116b)$$

The cavity problem has been solved by *Chen-To Tai* and the Green's function can be found in [7]. For the specific component of interest, the solution is given by [12]

$$G^{cav}(P, P') = \frac{-\omega\epsilon}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{2\epsilon_{mn}^2 k^2 - \left(\frac{m\pi}{a}\right)^2}{\beta_{mn} k^2} \sin \frac{m\pi x}{a} \cos \frac{m\pi y}{b} \quad (2.117)$$

$$\cdot \sin \frac{m\pi x'}{a} \cos \frac{m\pi y'}{b} e^{-j\beta_{mn}|z-z'|}$$

wherein $\epsilon_{00} = 1/2$, $\epsilon_{m0} = \epsilon_{0n} = 1/\sqrt{2}$ and $\epsilon_{mn} = 1$ otherwise. β_{mn} is the wave number of a TE_{mn} wave as given in eq. 1.27. It is noticed that this quantity can be real as well as complex depending on whether the mode is propagating or evanescent. In the latter case we replace it with $-j\gamma_{mn}$. Now scattering superposition is used to obtain $G^{cav}(P, P')$ for the two situations mentioned above [12]

$$G^{cav}(z'=0, z=0) \quad (2.118a)$$

$$= \frac{-2\omega\epsilon}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{2\epsilon_{mn}^2 k^2 - \left(\frac{m\pi}{a}\right)^2}{\beta_{mn} k^2} f(x, x', y, y') \cot(\beta_{mn}t)$$

$$= \frac{2\omega\epsilon}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{2\epsilon_{mn}^2 k^2 - \left(\frac{m\pi}{a}\right)^2}{j\gamma_{mn} k^2} f(x, x', y, y') \coth(\gamma_{mn}t)$$

$$G^{cav}(z'=0, z=t) \quad (2.118b)$$

$$= \frac{-2\omega\epsilon}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{2\epsilon_{mn}^2 k^2 - \left(\frac{m\pi}{a}\right)^2}{\beta_{mn} k^2} f(x, x', y, y') \frac{1}{\sin(\beta_{mn}t)}$$

$$= \frac{2\omega\epsilon}{ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{2\epsilon_{mn}^2 k^2 - \left(\frac{m\pi}{a}\right)^2}{j\gamma_{mn} k^2} f(x, x', y, y') \frac{1}{\sinh(\gamma_{mn}t)}$$

where the function $f(x, x', y, y')$ is the sinusoidal terms in eq. 2.117. The Green's

functions $G^{cav}(z = t, z' = t)$ and $G^{cav}(z = t, z' = 0)$ are due to symmetry identical to the expressions given in eq. 2.118a and 2.118b respectively. Next we apply eq. 2.12 and the normalization given in 2.16a to the two equations 2.115b and 2.115d. This gives the following contribution to the fields at the two interfaces

$$Y_{pq}^{11} = Y_{pq}^{22} = Z_0 \int_{slot} \sin \frac{q\pi x}{2L} \int_{slot} G^{cav}(0, 0) \sin \frac{p\pi x'}{2L} ds' ds \quad (2.119a)$$

$$Y_{pq}^{12} = Y_{pq}^{21} = Z_0 \int_{slot} \sin \frac{q\pi x}{2L} \int_{slot} G^{cav}(0, t) \sin \frac{p\pi x'}{2L} ds' ds \quad (2.119b)$$

where the expansion and weighting functions have been changed in accordance with the new coordinate definition given in figure 2.7. The result of the integrations in eq. 2.119a and 2.119b becomes very simple because of the orthogonality of the sinusoidal functions. Thus all four terms can be written as diagonal matrices with the following terms

$$\begin{aligned} Y_{pp}^{11} = Y_{pp}^{22} &= \frac{jZ_0 L \beta_{p0}}{\mu\omega} \cot(\beta_{p0}t) && \text{propagating modes} \\ &= \frac{jZ_0 L \gamma_{p0}}{\mu\omega} \coth(\gamma_{p0}t) && \text{evanecent modes} \end{aligned} \quad (2.120)$$

and

$$\begin{aligned} Y_{pp}^{12} = Y_{pp}^{21} &= \frac{jZ_0 L \beta_{p0}}{\mu\omega} \frac{1}{\sin(\beta_{p0}t)} && \text{propagating modes} \\ &= \frac{jZ_0 L \gamma_{p0}}{\mu\omega} \frac{1}{\sinh(\gamma_{p0}t)} && \text{evanecent modes} \end{aligned} \quad (2.121)$$

where γ_{p0} and β_{p0} are given by the two branches

$$\gamma_{p0} = \sqrt{\left(\frac{p\pi}{a}\right)^2 - k^2}, \quad \beta_{p0} = \sqrt{k^2 - \left(\frac{p\pi}{a}\right)^2} \quad (2.122)$$

Collecting all terms, eq. 2.115b and 2.115d are written in the following matrix form

$$\begin{bmatrix} [Y_{11}] & [Y_{12}] \\ [Y_{21}] & [Y_{22}] \end{bmatrix} \begin{bmatrix} [E^{lower}] \\ [E^{upper}] \end{bmatrix} = \begin{bmatrix} [h_q^{inc}] \\ [0] \end{bmatrix} \quad (2.123)$$

where

$$Y_{11,pq} = Y_{pq}^{11} - Y_{pq}^{int} \quad (2.124a)$$

$$Y_{12,pq} = Y_{pq}^{12} \quad (2.124b)$$

$$Y_{21,pq} = Y_{pq}^{12} \quad (2.124c)$$

$$Y_{22,pq} = Y_{pq}^{11} + Y_{pq}^{ext} \quad (2.124d)$$

This $2N \times 2N$ matrix equation is easily solved with only slightly increased computation time. The most time-consuming part is still computation of the Y_{pq}^{ext} and Y_{pq}^{int} terms.

This method is applicable to longitudinal slots and transverse slots only. For T-slots in a finite wall a special treatment is required and hence it is not considered in this thesis.

2.6. Correction to Rounded Slot Ends

In the manufacturing of waveguide slot antennas, slots with rounded ends are often preferred. Especially at lower frequencies where the waveguide dimensions are large, a milling process is obvious and thereby rounded slot ends.

In order to correct the rounded slot ends the following correction formula is assumed to be valid

$$L_0 - L_{\square} = \frac{w}{2} \left(1 - \frac{\pi}{4}\right) \quad (2.125)$$

where L_0 and L_{\square} are the lengths of rounded and rectangular slot ends respectively, w is the slot width. eq. 2.125 ensures equal slot area, which was successfully applied in [2].

Chapter 3

Results

The moment method described in chap. 2 has been implemented and used to investigate longitudinal, transverse and T-slots. In the following sections design data for these three cases are presented.

3.1. Longitudinal Slots

From the reflection coefficient Γ given in sect. 1.4 for the fundamental TE_{10} mode the equivalent normalized slot admittance is obtained as

$$\frac{Y}{Y_0} = \frac{-2\Gamma}{1 + \Gamma} \quad (3.1)$$

This simple shunt model implies that the scattering of the slot is symmetrical, i.e., the backscattering equals the forward scattering [2]. Figure 3.1 shows the basic characteristic of the admittance for a longitudinal slot cut in a full-height (standard) X -band waveguide as a function of offsets and slot lengths. The frequency was kept constant at $9.375GHz$ and the number of expansion functions for representing the slot fields was nine. Fewer terms can be used when less accurate results are acceptable to save computer time. It is seen that the slot acts like a resonant circuit with a certain Q , where a resonant length of a slot is defined by its equivalent admittance being purely real. The resonant length versus offset from the waveguide centerline is shown in figure 3.2. This result shows to be in good agreement with computed data given in [2] though the MoM data is obtained with a different weighting function. In [2] pulse weighting is applied across the slot aperture while the present MoM relies on point-matching. A longitudinal slot radiating into an infinite PP waveguide has also been investigated and the results for two different PP spacings are shown in figure 3.3. For verification computed data given in [16] has been included and good agreements

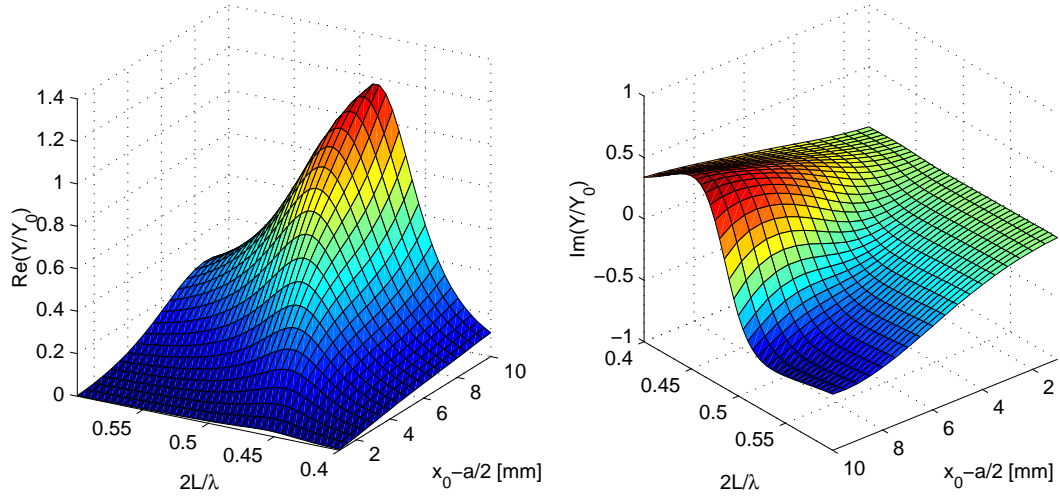


Figure 3.1.: Longitudinal slot admittance - zero wall thickness.

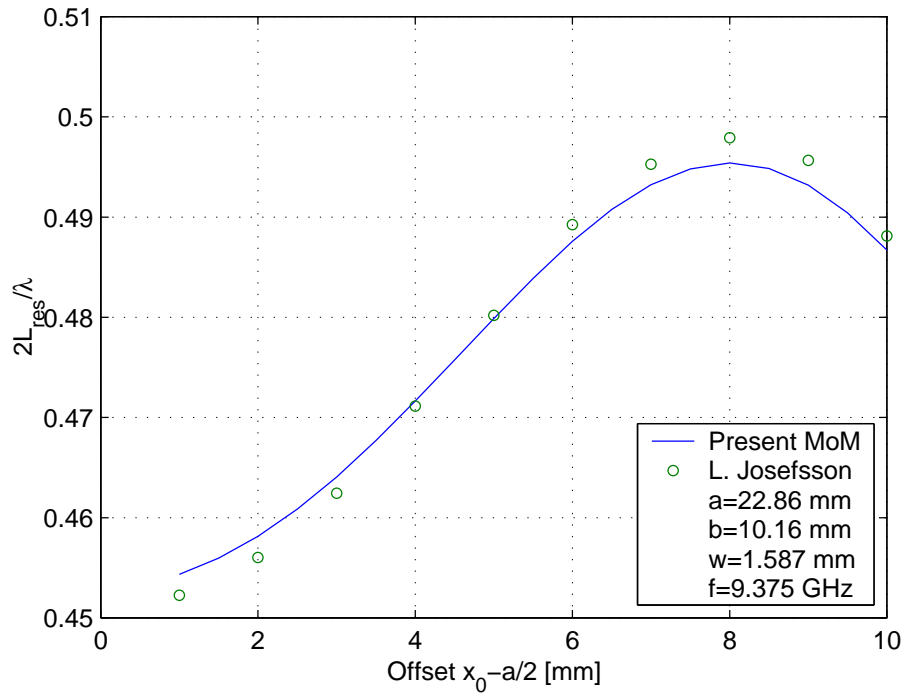


Figure 3.2.: Longitudinal slot resonant length versus offset.

are noticed. The small differences are assumed to be due to the different weighting

function as mentioned before.

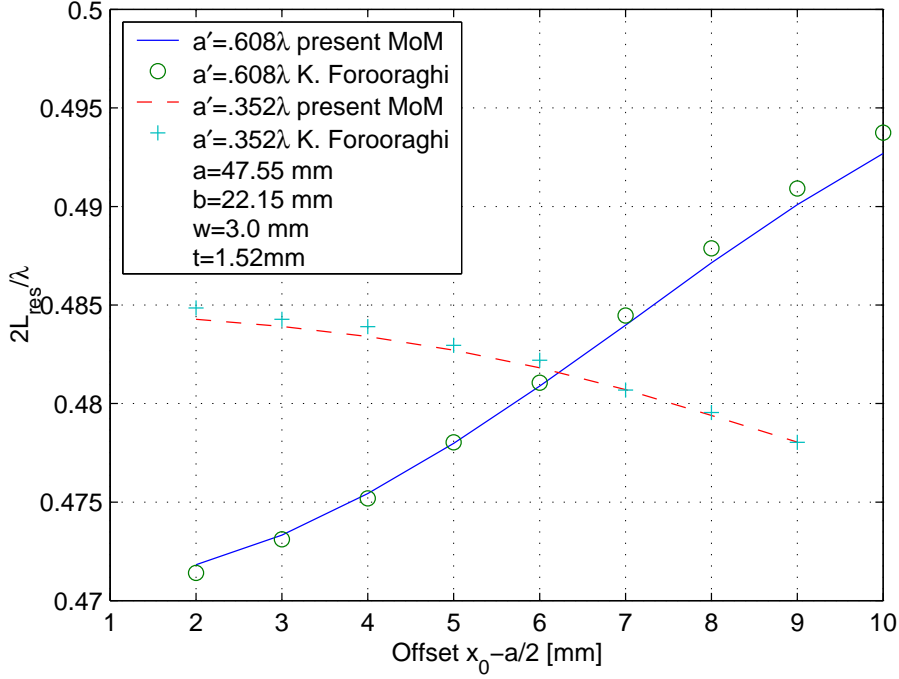


Figure 3.3.: Slot resonant length versus offset for a longitudinal radiating into a PP region.

3.2. Transverse Slots

The equivalent normalized slot impedance for a transverse slot is obtained as

$$\frac{Z}{Z_0} = \frac{2\Gamma}{1 - \Gamma} \quad (3.2)$$

This series model implies that the forward and backward scattering are equal in magnitude but 180° out of phase.

Figure 3.4 and 3.5 show the basic impedance characteristics for a transverse slot as a function of frequency. This also acts like a resonant circuit with a certain Q and we define the resonance frequency when the equivalent impedance is purely real. The number of expansion functions for representing the slot fields was nine.

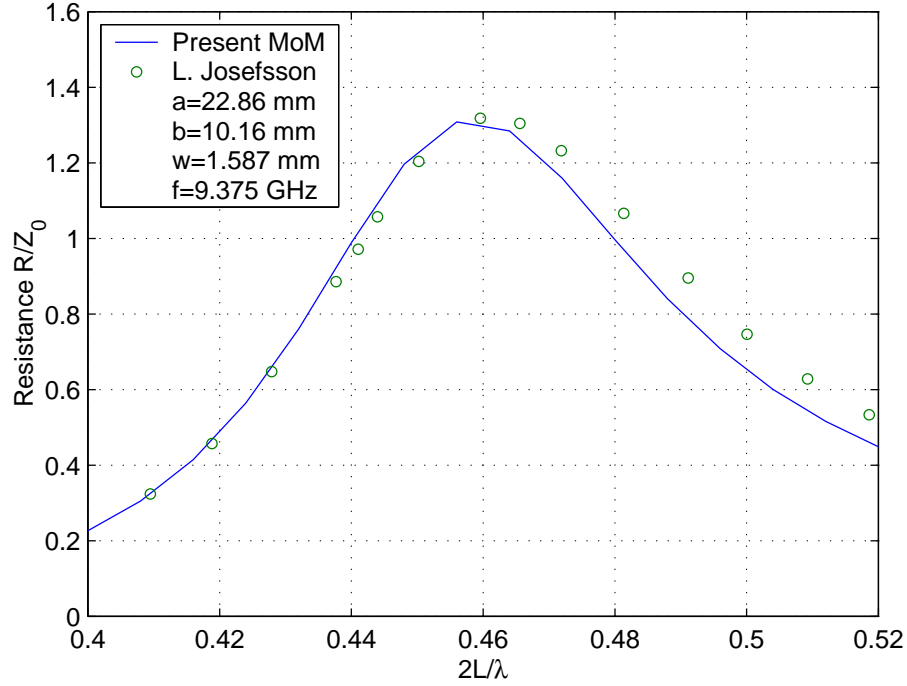


Figure 3.4.: Transverse slot resistance - zero wall and offset.

Figure 3.6 and 3.7 shows the impedance of a centered transverse slot radiating into an infinite PP waveguide. It demonstrates the effect of changing the slot length. We notice that the characteristic is different from that of transverse slots in an infinite ground plane. For the latter a change in slot length corresponds to a proportional frequency shift, the impedance level largely remaining unchanged. Data from *Lars Josefsson's* paper [3] has been included in figure 3.6 (shown as circles) and a comparison shows excellent agreement. This is despite use of different weighting functions in the method of moments.

3.3. T-Slots

Finally the T-slot has been investigated. Since this type of slots is more complicated than the previous two, a simple network model might not be adequate. However, the results shown here are for zero offset ($x_0 = a/2$) which is assumed to be a sufficient requirement in order to model the slot characteristic as a series impedance.

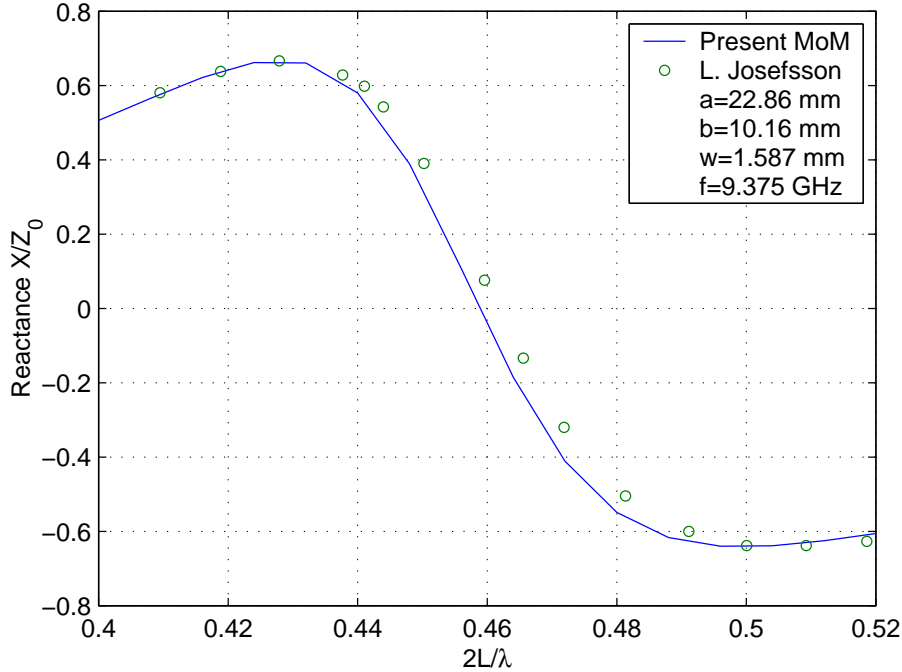


Figure 3.5.: Transverse slot reactance - zero wall and offset.

3.3.1. Analysis of Convergence

Before obtaining valid results, we investigated the convergence requirements of the series representations for the slot fields. The decay of expansion coefficients has been investigated for one particular case and the result is seen in figure 3.8. When the T-slot is centered the expansion terms E_p for p odd are very low. This confirms our expectations regarding the electric fields in the longitudinal part being odd-symmetric. In the following analysis the number of expansion functions used in the moment method was 24 for the longitudinal current M_z and 13 for the transverse current M_x . As seen in figure 3.8 this should ensure all higher order terms being 50dB below the expansion term with the highest amplitude.

3.3.2. Electric Field Distribution

Two examples of computed electric field distribution along the slots are shown in figure 3.9 and 3.10. As expected the electric field E_z along the longitudinal part is odd-symmetric. However, for small excitation levels the fields becomes more confined to the longitudinal part, which might give rise to a higher cross polarized field in the

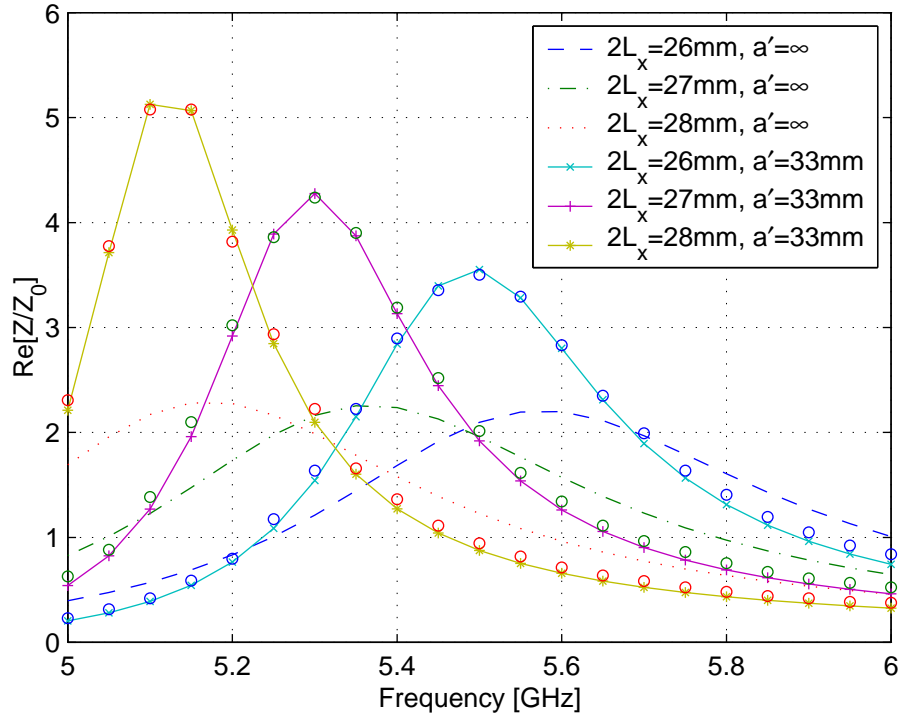


Figure 3.6.: Transverse slot resistance versus frequency for three slot lengths. The waveguide has the dimension: $a = 39\text{mm}$ and $b = 10\text{mm}$ and the wall thickness was $t = 1.9\text{mm}$.

radiation patterns.

3.3.3. Impedance Data

T-slot Radiating into Half Space

Figure 3.11 and 3.12 show computed resistance and reactance for eight different slot lengths L_x as a function of the length $L_z + 2L_x$. The slots were cut in a full-height X-band waveguide and the frequency was kept constant at 9.375GHz . The slot width w was 1.5875mm . It demonstrates how the level of excitation is changed by changing the length L_x .

Figure 3.13 shows the resonant resistance as a function of slot length L_x . Compared to the transverse slot we have significantly better dynamic range. The return loss varies approximately from 10 to 25dB . Figure 3.14, finally, shows the resonant length

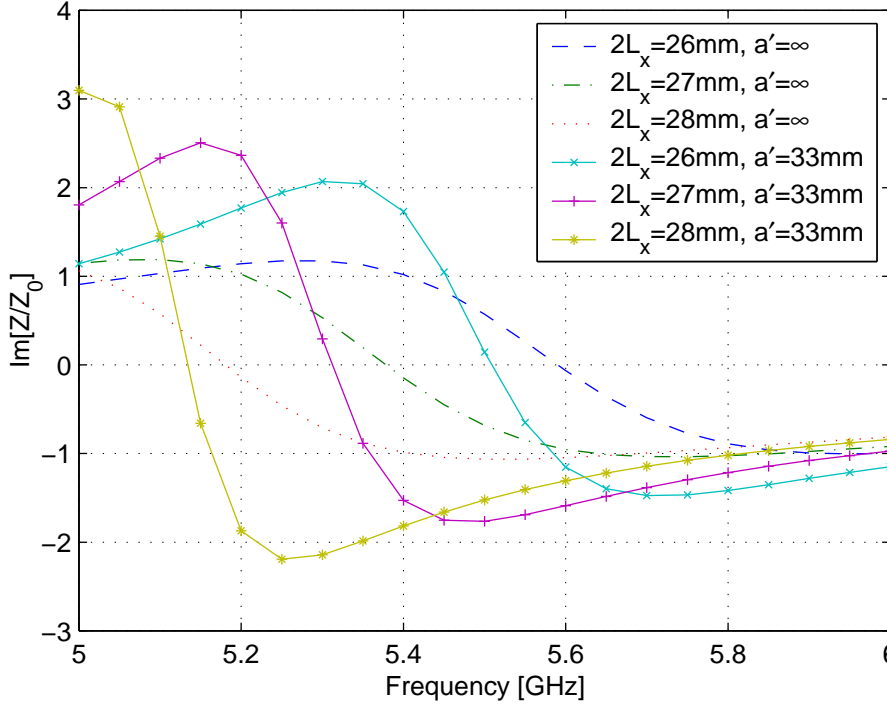


Figure 3.7.: Transverse slot reactance versus frequency for three slot lengths. The waveguide has the dimension: $a = 39\text{mm}$ and $b = 10\text{mm}$ and the wall thickness was $t = 1.9\text{mm}$.

$L_z + 2L_x$ versus the length L_x . The length $L_z + 2L - x$ can in some way be compared with the resonant length for longitudinal or transverse slots where a resonant length of a slot is defined by its equivalent admittance or impedance being purely real. Measured in wavelength the resonant length changes from $.49\lambda$ to $.59\lambda$ which is significantly more than any of the former slots.

T-slot Radiating into a PP waveguide

Figure 3.15 and 3.16 show computed resistance and reactance for eight different slot lengths L_x as a function of the length $L_z + 2L_x$. The waveguide dimensions, frequency etc. as before. Compared to the T-slot radiating into half space it demonstrates how the impedance level is increased. This is expected as we have seen before in figure 3.6 and 3.7.

Figure 3.18 shows the resonant resistance as a function of slot length L_x . The dynamic range largely remains unchanged and return loss varies approximately from

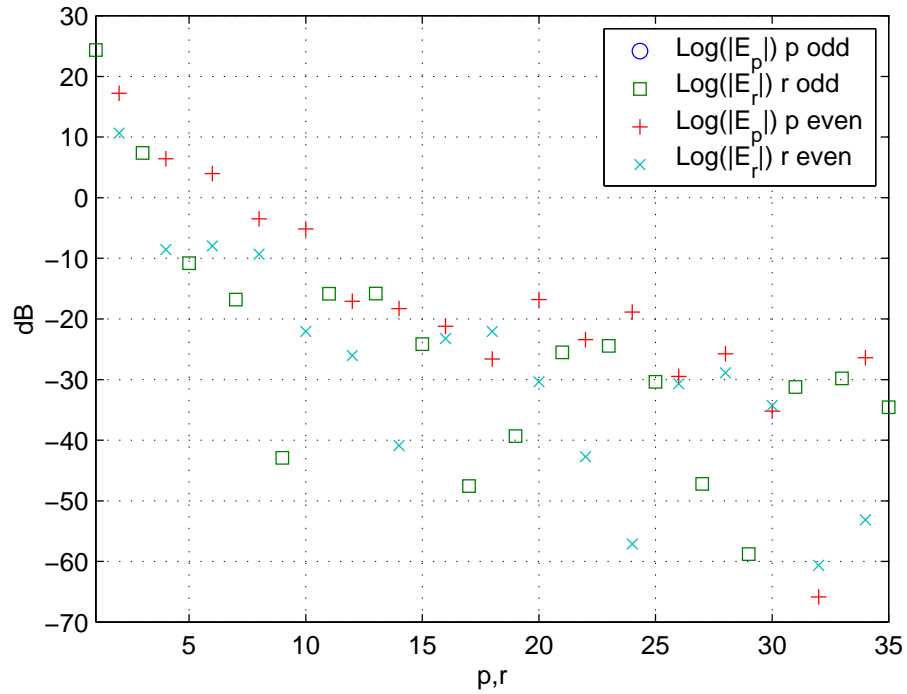


Figure 3.8.: Decay of expansion coefficients in case $x_0 = a/2$. $\log(|E_p|)$ is always below -100dB for p odd.

6 to 21dB . Figure 3.17 shows the resonant length $L_z + 2L - x$ versus the length L_x . Again, there is only little difference between the half space and PP waveguide.

Figure 3.19 shows for one particular frequency how the impedance changes with baffle spacing a' . This slot was resonant for $a' = 0.65\lambda$. It remains approximately resonant for higher spacings but the resistance changes more rapidly. Near cutoff the impedance decreases rapidly.

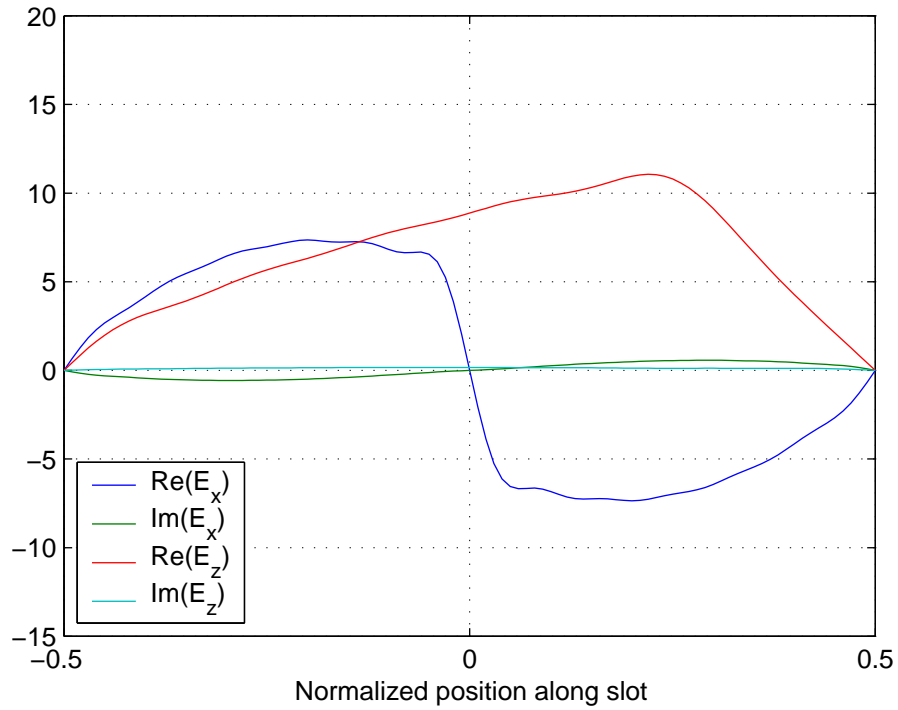


Figure 3.9.: Slot electric field distribution. $L_x = 32mm$.

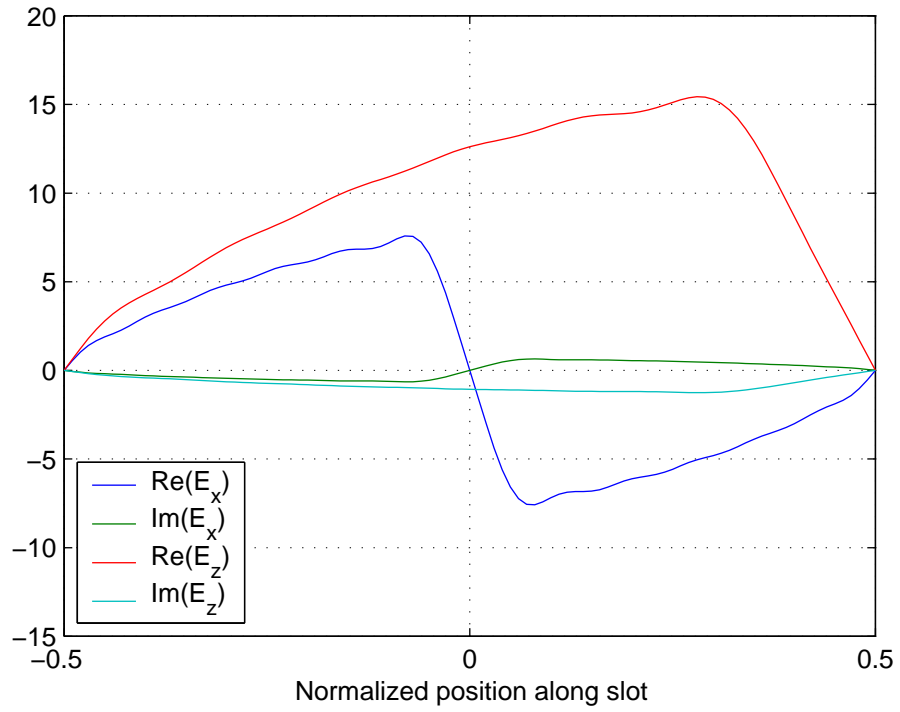


Figure 3.10.: Slot electric field distribution. $L_x = 44mm$.

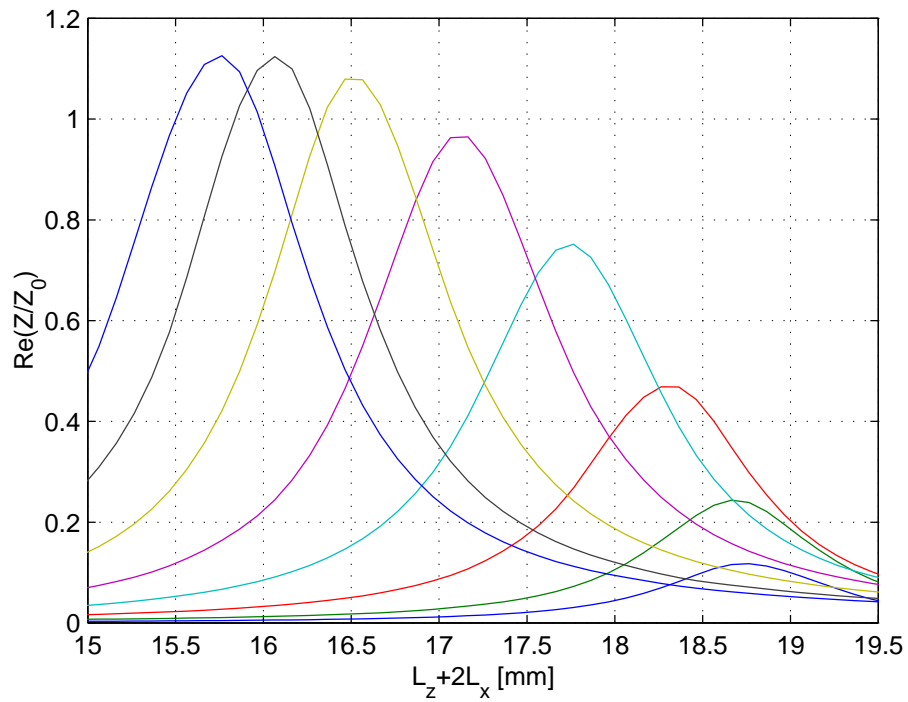


Figure 3.11.: T-slot resistance versus slot length L_z for eight slot lengths L_x . From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

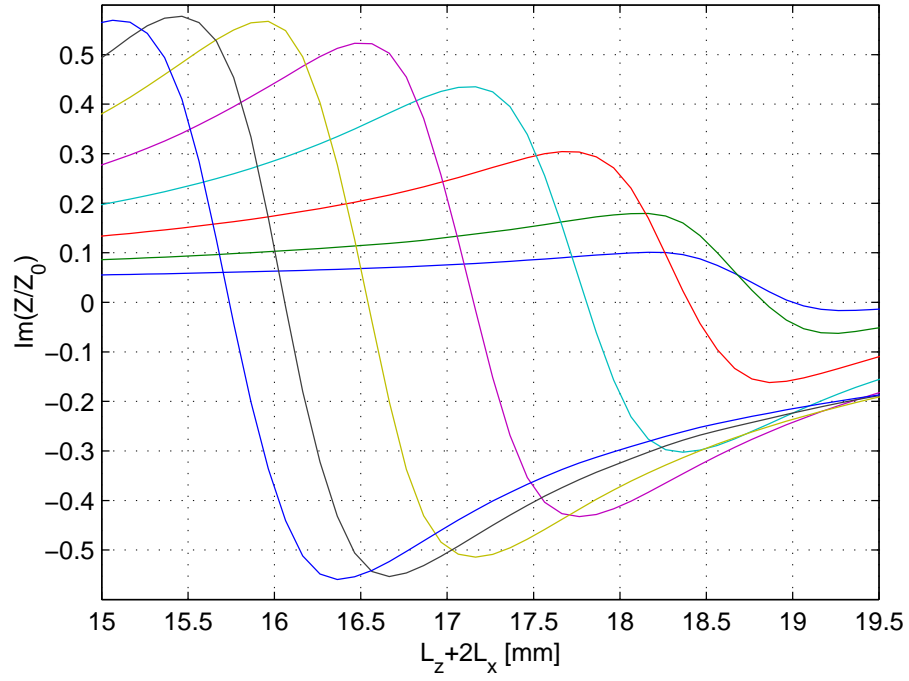


Figure 3.12.: T-slot reactance versus slot length L_z for eight slot lengths L_x . From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

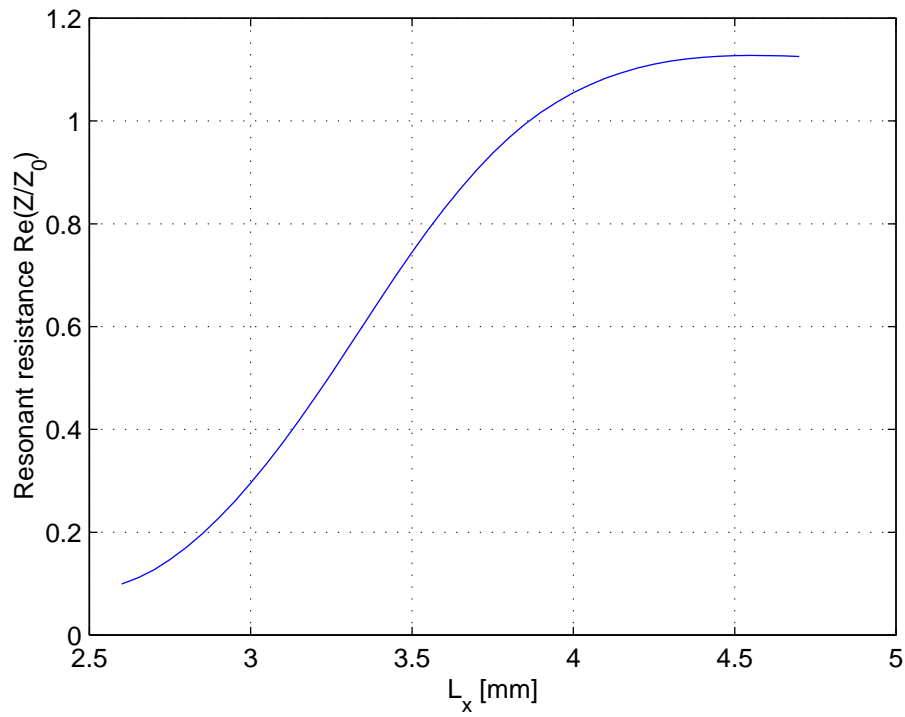


Figure 3.13.: T-slot resonant resistance versus slot length L_x

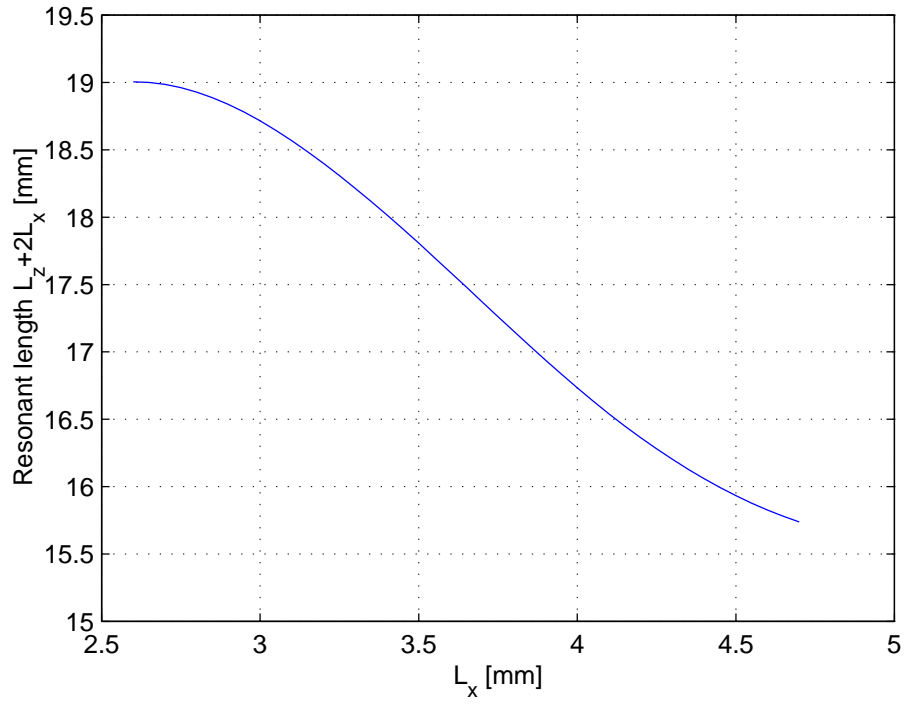


Figure 3.14.: T-slot resonant length $L_z + 2L_x$ versus slot length L_x

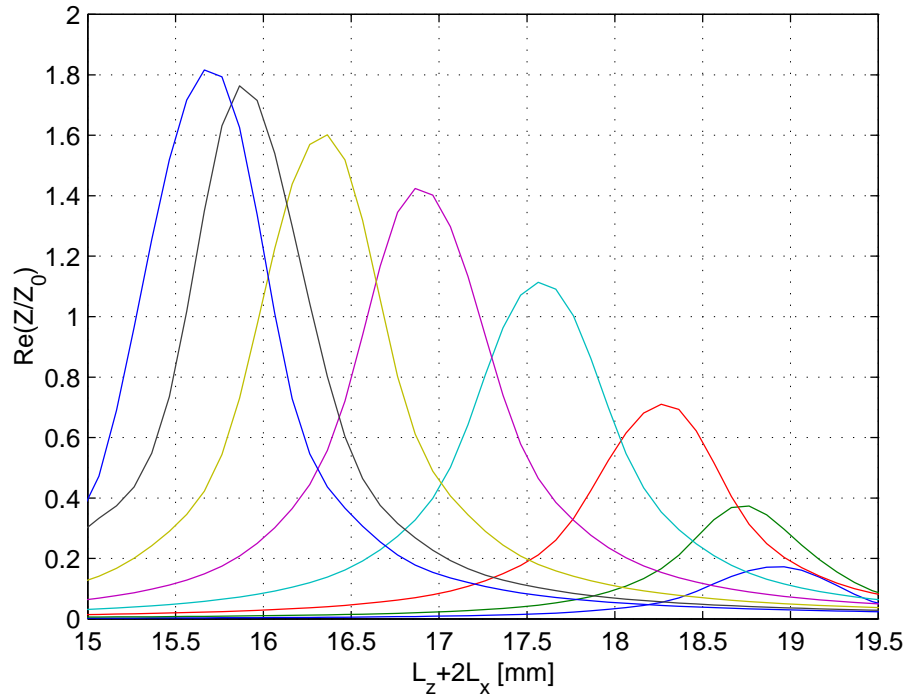


Figure 3.15.: T-slot resistance versus slot length L_z for eight slot lengths L_x . From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

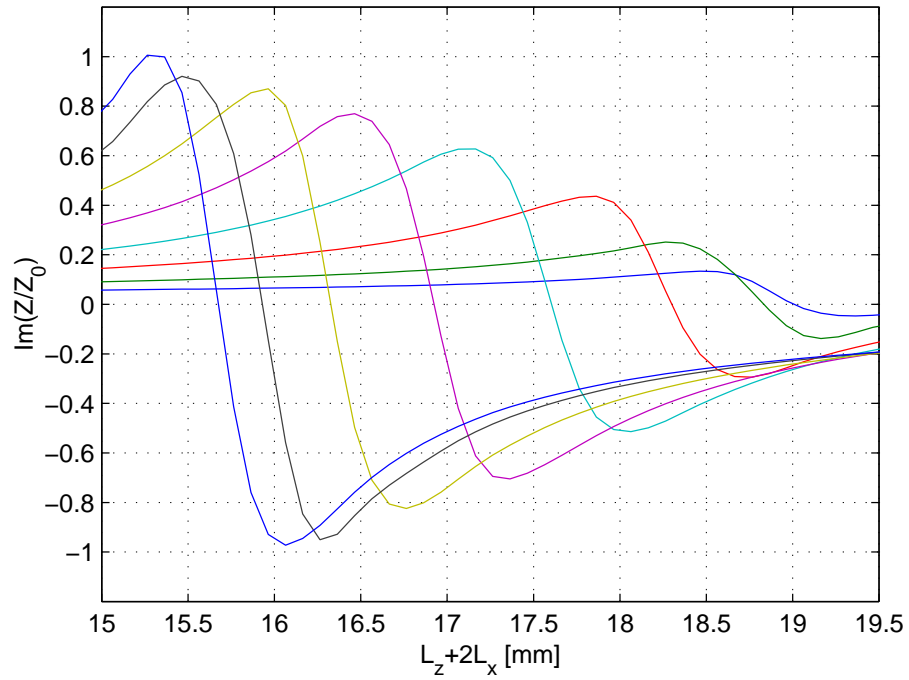


Figure 3.16.: T-slot reactance versus slot length L_z for eight slot lengths L_x . From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

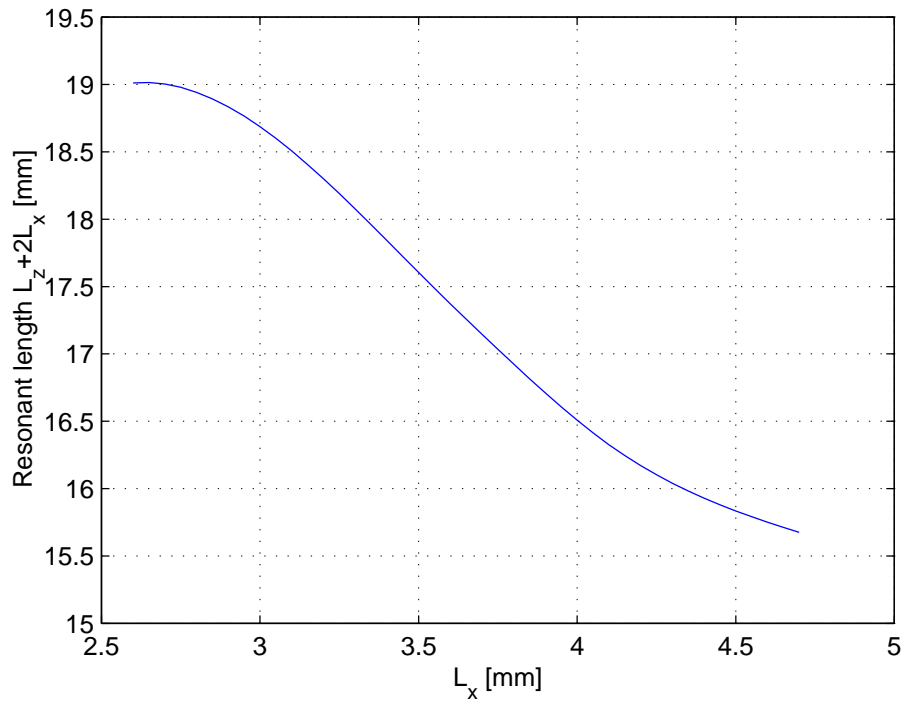


Figure 3.17.: T-slot resonant length $L_z + 2L_x$ versus slot length L_x

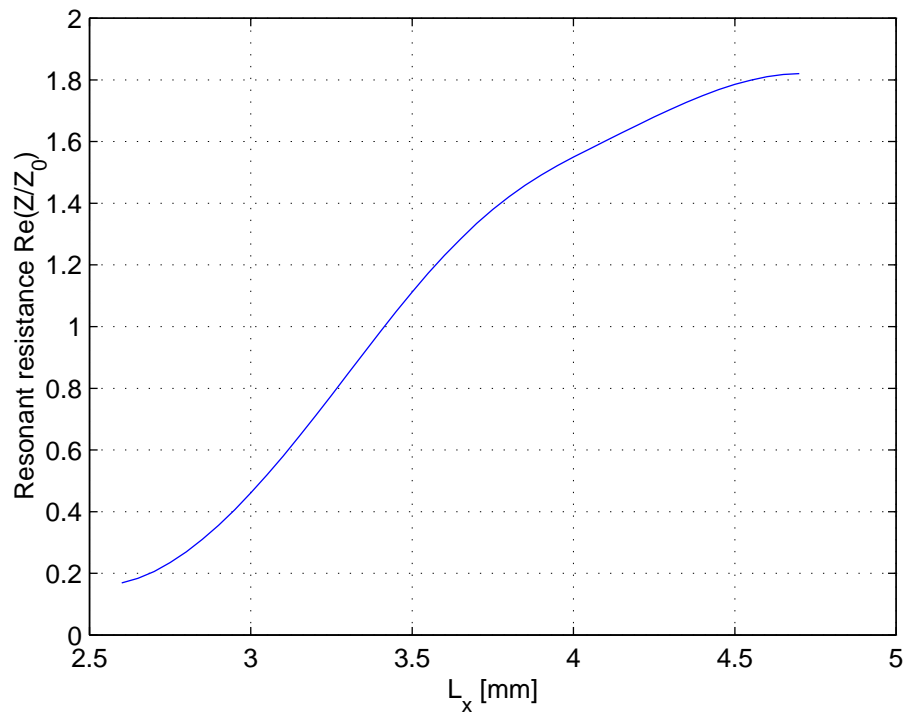


Figure 3.18.: T-slot resonant resistance versus slot length L_x

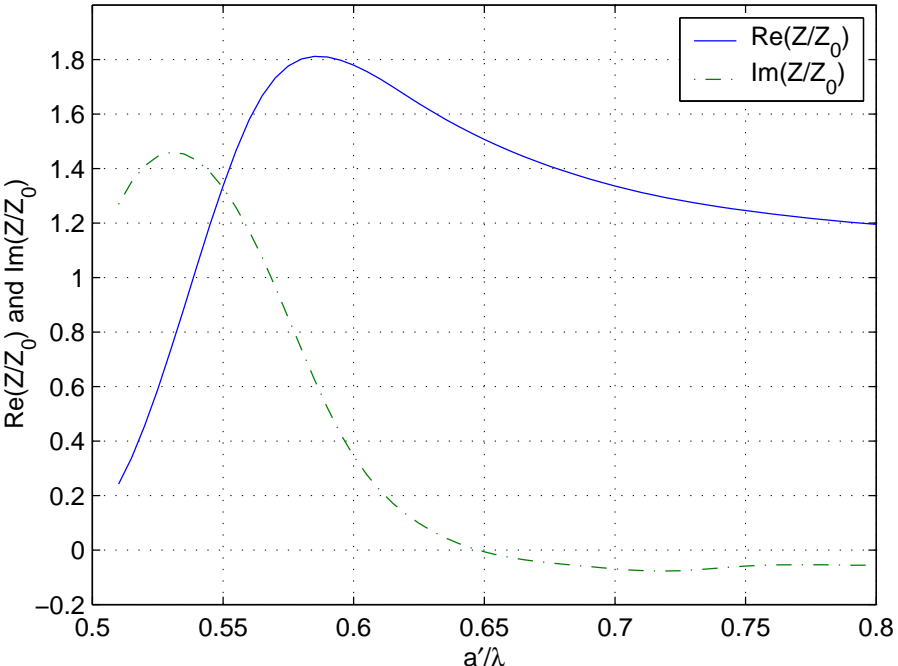


Figure 3.19.: T-slot impedance versus baffle spacing in free space wavelengths.

3.3.4. Radiation Patterns

T-slot Radiating into Half Space

Figure 3.20 and 3.21 show computed principle planes for three resonant T-slots radiating into half space. Again, the slot was cut in a full-height X -band waveguide, the frequency was kept constant at $9.375GHz$ and the slot width was $w = 1.5875mm$. The E-plane is seen to be omnidirectional which indicates that the slot has the same characteristics as a magnetic dipole. As mentioned earlier, the cross-polarized field becomes high for low levels of excitation. However, this cross-polarized pattern has a null at broadside which should be utilized in array applications. The H-plane is seen to be wide and the level of the cross polarized field is far below the axis.

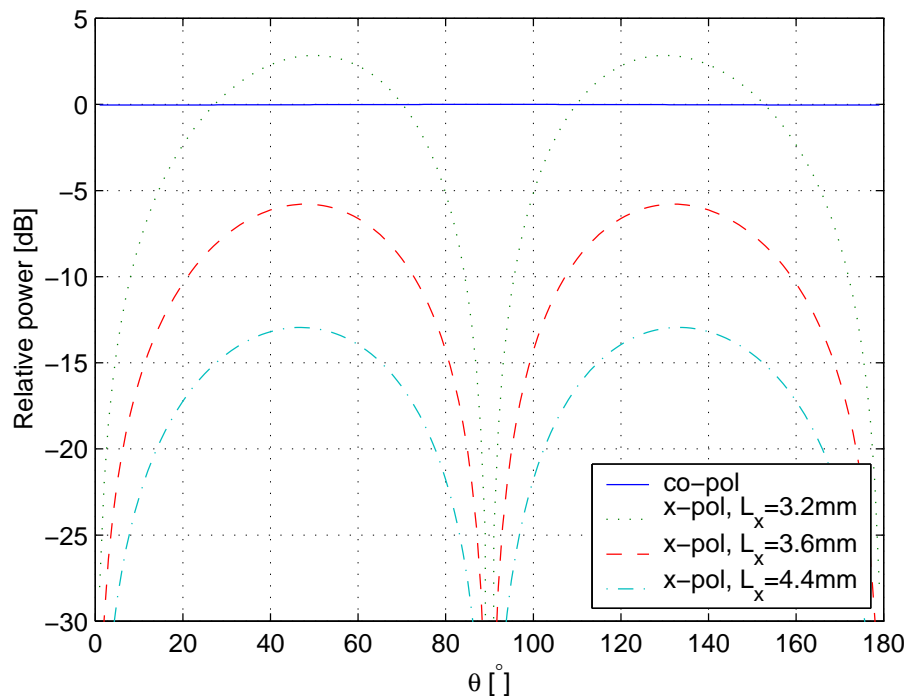


Figure 3.20.: E-plane patterns of a T-slot.

T-slot Radiating into a PP waveguide

Figure 3.22 and 3.23 show computed principle planes for three resonant T-slots radiating into a PP waveguide with the dimensions $a' \times b' = 21.0 \times 25.0mm^2$. Other dimensions remain unchanged. Most conspicuous is the cut-off angle occurring at about

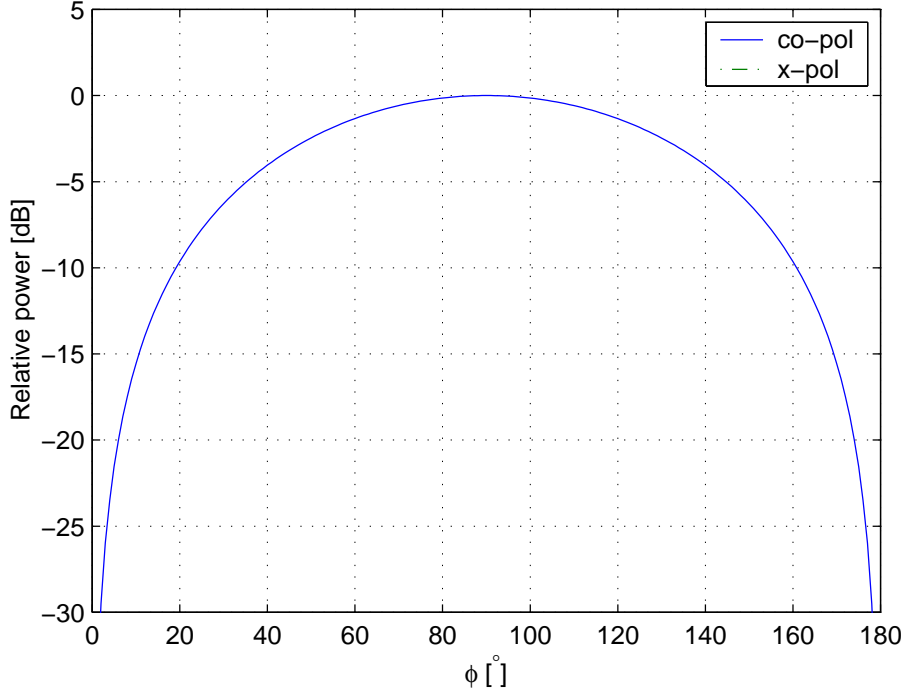


Figure 3.21.: H-plane patterns of a T-slot.

$\pm 40^\circ$. Assuming only the first order mode is propagating in the PP waveguide, the theoretical cut-off angle becomes $\theta_c = 90 - \arcsin(\lambda_0/2a') = 40.3^\circ$. Therefore, the omnidirectional E-plane pattern seen for the T-slot radiating into half space has been shaped substantially introducing the PP waveguide. Again, the cross polarized field becomes high for low levels of excitation, but the null at broadside remains unchanged.

Figure 3.24 and 3.25 show computed principle planes for a 9-element array of uniform excited T-slots ($L_x = 3.8mm$). Computation of these array patterns is based on the principle of pattern multiplication [6]

$$D(\theta, \phi) \propto |f(\theta, \phi)|^2 |F(\theta, \phi)|^2 \quad (3.3)$$

where D is the directivity. The array factor $F(\theta, \phi)$ is given by [6]

$$F(\theta, \phi) = \sum_{i=n}^N C_n e^{jnd(k\cos\theta - \beta_{10})} \quad (3.4)$$

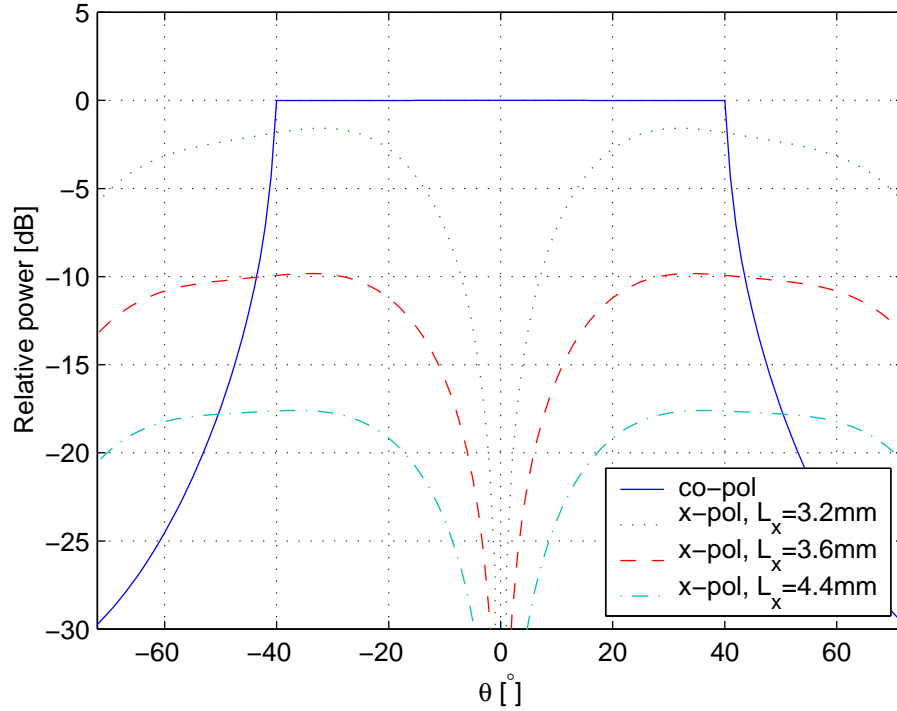


Figure 3.22.: E-plane patterns of a T-slot.

where C_n is the excitation coefficient for the n 'th element. d is the element spacing, which is set to be one guide wavelength λ_g and the element pattern $\mathbf{f}(\theta, \phi)$ is found from

$$\mathbf{E} = \mathbf{f}(\theta, \phi) \frac{e^{-jkr}}{4\pi r} \quad (3.5)$$

The results shows substantially suppression of grating lobes occurring at about $\pm 45^\circ$, which indicates the high degree of spatial filtering obtained by application of the PP waveguide. Even better performance can be achieved by decreasing the parallel plate spacing or applying amplitude tapering.

Figure 3.26 shows computed E-plane pattern for a 9-element array of uniform excited T-slots, but with the transverse part of the slot alternating around the centerline. This will cause the phase of the electric field in the longitudinal part to change 180° for consecutive slots. However, it does mainly affect the position of the grating lobes in the cross polarized pattern.

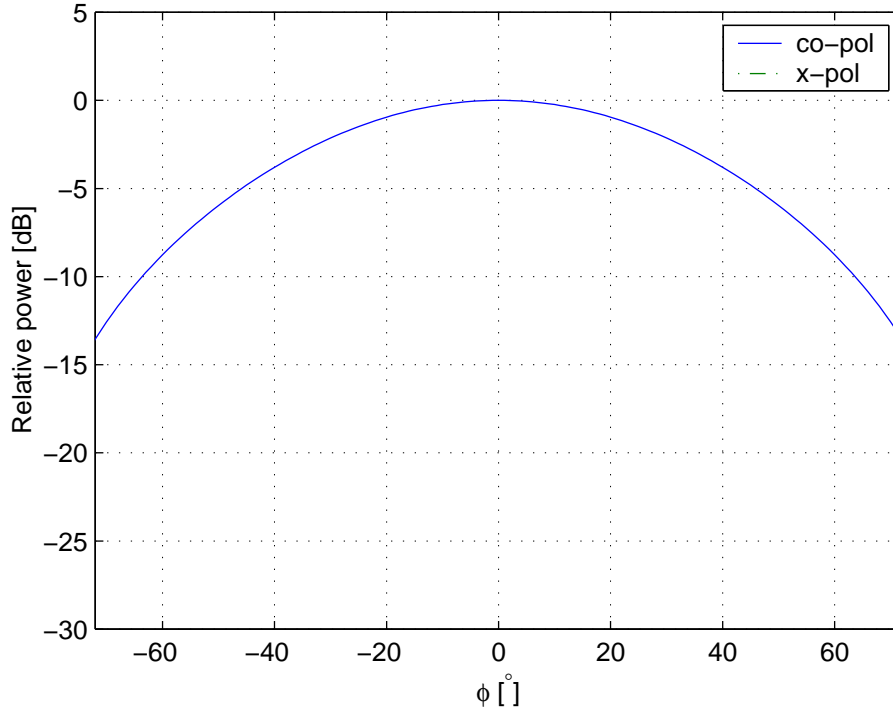


Figure 3.23.: H-plane patterns of a T-slot.

Figure 3.27 shows computed E-plane pattern for the same set up except for a different parallel plate spacing reduced to $a' = 19.5mm$. Then the theoretical cut-off angle becomes $\theta_c = 90 - \arcsin(\lambda_0/2a') = 34.9^\circ$ giving better suppression of the grating lobe at $\pm 45^\circ$.

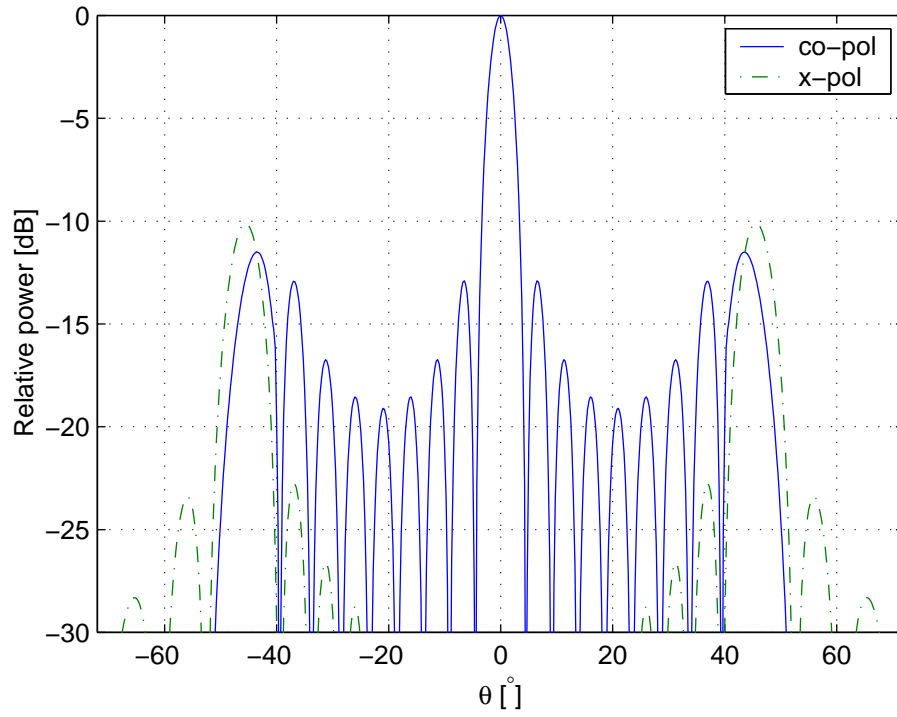


Figure 3.24.: E-plane patterns of a 9 element array of T-slots.

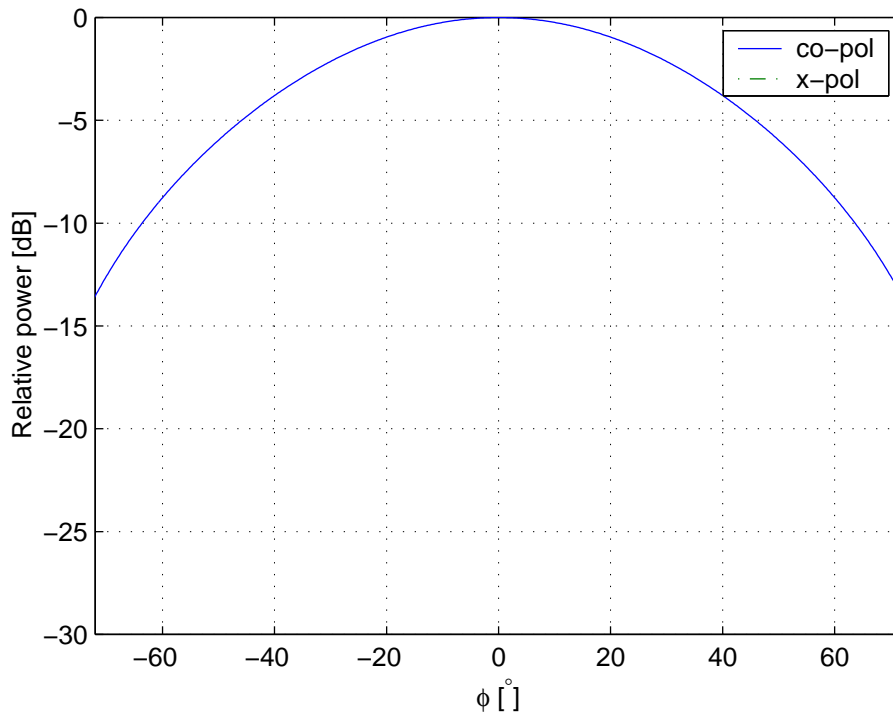


Figure 3.25.: H-plane patterns of a 9 element array of T-slots.

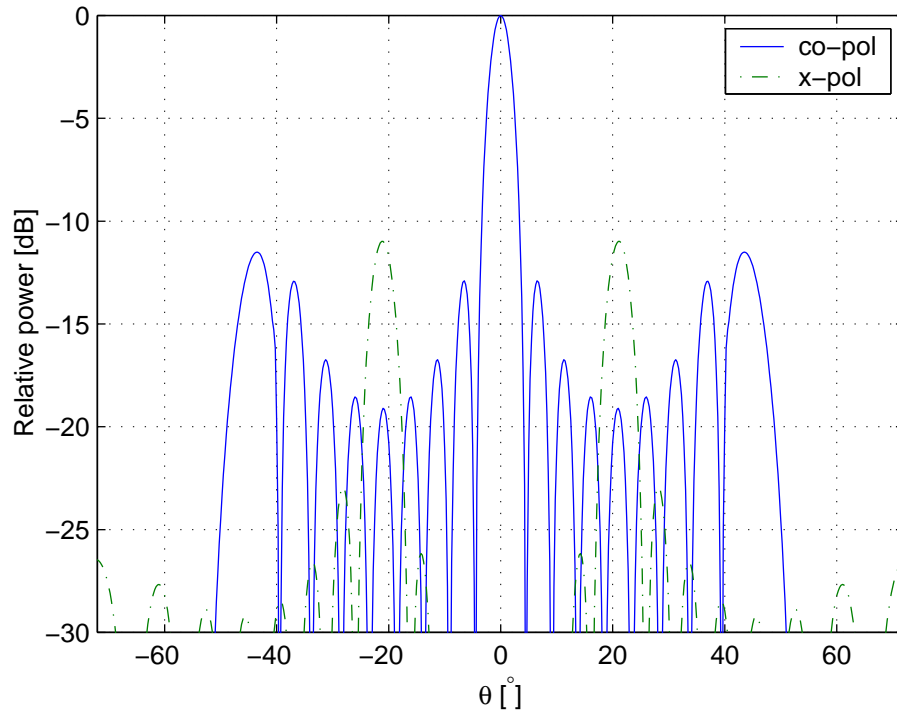


Figure 3.26.: E-plane patterns of a 9 element array of T-slots having alternating offset.

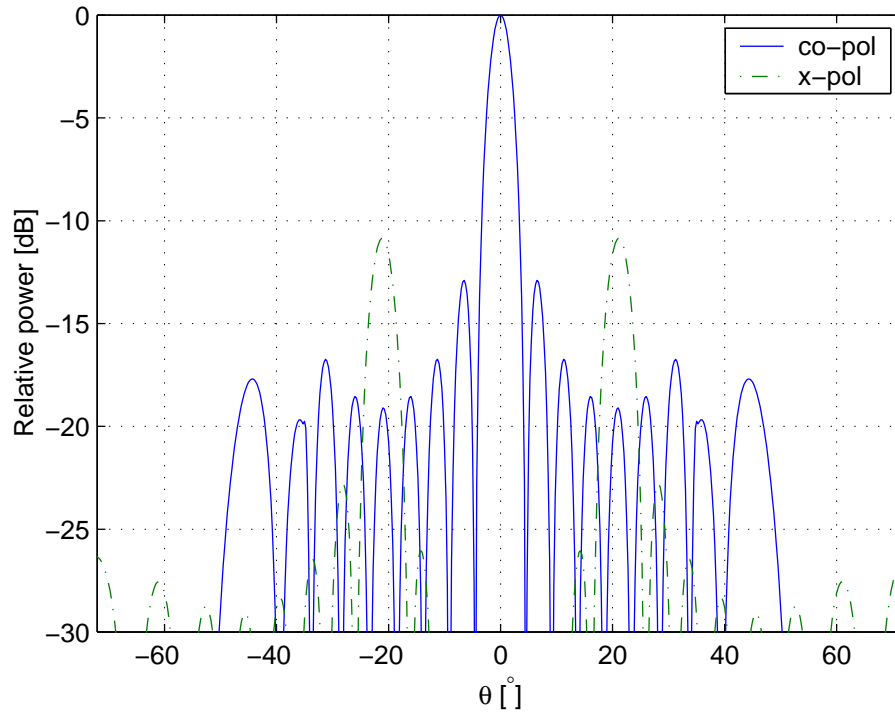


Figure 3.27.: E-plane patterns of a 9 element array of T-slots having alternating offset. PP spacing was $a' = 19.5mm$

Chapter 4

Software

The implemented MoM program used to investigate the properties of a T-slot is listed in appendix D. Furthermore, some postprocessing programs are listed.

The MoM code was written in Fortran90 because it's inherently suitable for complex variables. Also, Fortran is by far the language of choice because of its many computational advantages and because of a large existing library of mathematical routines.

A flowchart for the MoM program is given in figure 4.1 showing the main parts of the program and the related routine names. However, these routines depends on several subprograms and library routines. The program uses an input file *slot.in* containing all the necessary data for set up of a new simulation or post-processing of data from a previous run. To fulfill this flexibility requirement the program has several branches in the control statements and subprograms are not always executed sequentially as indicated in figure 4.1.

4.1. Organization of the Documentation

The MoM program listed in appendix D contains a concise description of each routine and all information pertaining to the program is found in the header of each routine. This header consists of the following information

- Routine Name
- Purpose: a statement of the purpose of the routine
- Usage: the form for referencing the subprogram with arguments listed. There are two usage forms:

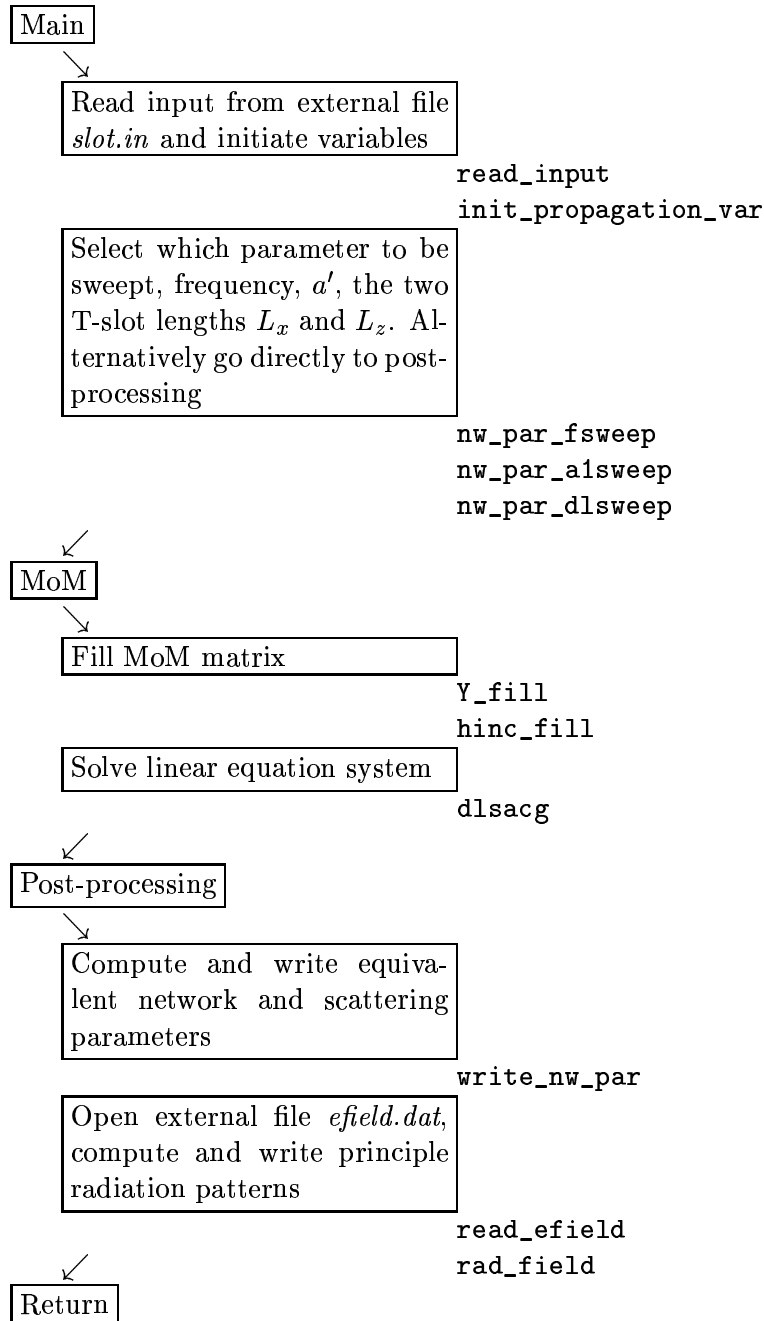


Figure 4.1.: Flowchart for MoM program.

- `call sub(argument-list)` for subroutines
- `fun(argument-list)` for functions
- Arguments: a description of the arguments in the order of their occurrence. Input arguments usually first, followed by output arguments. For functions, the symbolic name is described after the argument descriptions
- Modules: a list of modules used by the routine
- Subroutines: a list of other routines used by the routine
- IMSL subroutines: a list of routine from the IMSL library used by the routine
- Remarks: details pertaining to code usage.

4.2. Software Packages

Some of the subroutines and functions given in appendix D involving evaluation of integrals and solving linear equation systems relies on the *IMSL* library [17]. This library applies fast and accurate algorithms and has reduced the workload significantly and made it possible to reduce the effort put into implementing and optimizing numerical algorithms. The following routines have been applied

- `dlsacg` Solves a complex general system of linear equations with iterative refinement. It computes an *LU* factorization of the coefficient matrix and obtain the solution using iterative refinement.
- `dqdag` Integrates a function using a globally adaptive scheme based on Gauss-Kronrod rules. It is a general-purpose integrator that uses a globally adaptive scheme in order to reduce the absolute error. It subdivides the interval $[A, B]$ and uses a $(2k + 1)$ -point Gauss-Kronrod rule to estimate the integral over each subinterval. The error for each subinterval is estimated by comparison with the k -point Gauss quadrature rule. The subinterval with the largest error is then bisected and the same procedure is applied to both halves. The bisection process is continued until the error criterium is satisfied.
- `dqdagi` Integrates a function over an infinite or semi-infinite interval. The routine uses a globally adaptive scheme in an attempt to reduce the absolute error. It initially transforms an infinite or semi-infinite interval into the finite interval $[0, 1]$. Then, `dqdagi` uses a 21-point Gauss-Kronrod rule to estimate the integral and error. It bisects any interval with an unacceptable error estimate and continues the process until termination.

- `dq3nd` Integrates an function on a hyper-rectangle. The routine approximates the n -dimensional iterated integral

$$\int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} f(x_1, \dots, x_n) dx_n \cdots dx_1$$

The approximation is achieved by iterated application of product Gauss formulas. The integral is first estimated by a two-point tensor product formula in each direction. Then for $i = 1, \dots, n$ the routine calculates a new estimate by doubling the number of points in the i -th direction, but halving the number immediately afterwards if the new estimate does not change appreciably. This process is repeated until either one complete sweep results in no increase in the number of sample points in any dimension, or the number of Gauss points in one direction exceeds 256.

Visualization of results was made in *Matlab v5.3*. The graphical solutions shown in eg. figure 3.11 and the contour plots figure 3.14 and 3.13 are based on the `interp` command [18]. It is a one-dimensional data interpolation (table lookup) algorithm based on splines [19]. The routine forms a small set of functions for working with piecewise polynomials and perform cubic spline interpolation.

Conclusion

The method of moments using entire expansion and weight functions was applied for solving the field distribution in the T-slot aperture. Based on expressions derived for the zero wall case, finite wall thickness was included in the formalism for longitudinal and transverse slots. Results from the analysis of these two configurations show good agreement with earlier published work.

The impedance of a T-slot radiating into half space or an infinite parallel plate waveguide was analyzed. It shows that the T-slot gives good control on the excitation, a property the transverse slot lacks. The radiation from a T-slot does have a considerable cross polarized component. Especially for a low level of excitation where the electric fields become more confined to the longitudinal part of the T-slot.

In array applications use of a baffle structure shows to be a very efficient way to suppress grating lobes in the co-polarized E-plane pattern. They arise due to an element spacing of one guide wavelength. Application of an amplitude tapering function could be used to improve the sidelobe level. However, due to the high cross-polarized field at low levels of excitation, tapering is not likely to significantly improve the cross-polarization performance.

For larger arrays, the position around the centerline of the transverse part of the T-slot can be randomized giving some performance improvement, though it is unlikely to suppress the cross-polarized field below eg. -25dB. Probably the problem can be solved by designing larger arrays consisting of a number of small sub-arrays. All single elements will then be higher excited giving lower cross-polarized fields. However, this will not utilize the better dynamic range in excitation levels gained by the T-slot. Therefore, compared with the transverse slot there is a trade off between the degree of excitation freedom and the cross-polarized field.

If the problem with the cross-polarized field is solved, a possible application of T-slots is as elements in single and dual polarized array antennas.

Conclusion

California State University, Northridge
February 29th, 2000

Anders Jensen

Bibliography

- [1] G. J. Stern and R. S. Elliot, "Resonant length of longitudinal slots and validation of circuit representation: Theory and experiment," *IEEE Transactions on Antennas and Propagation*, vol. AP-33, no. 11, pp. 1264–1271, 1985.
- [2] L. G. Josefsson, "Analysis of longitudinal slots in rectangular waveguides," *IEEE Transactions on Antennas and Propagation*, vol. AP-35, no. 12, pp. 1351–1357, 1987.
- [3] L. G. Josefsson, "A waveguide transverse slot for array applications," *IEEE Transactions on Antennas and Propagation*, vol. AP-41, no. 7, pp. 845–850, 1993.
- [4] H. Y. Yee and P. Stelitano, "I-slot characteristics," *IEEE Transactions on Antennas and Propagation*, vol. AP-40, no. 2, pp. 224–228, 1992.
- [5] S. Ramo, J. R. Whinnery, and T. V. Duzer, *Fields and Waves in Communication Electronics*. John Wiley and Sons, Inc., 1993.
- [6] R. E. Collin, *Antennas and Radiowave Propagation*. McGraw-Hill, 1992.
- [7] C.-T. Tai, *Dyadic Green Functions in Electromagnetic Theory*. IEEE Press, 1993.
- [8] R. S. Elliot, *An Introduction to Guided Waves and Microwave Circuits*. Prentice Hall, 1993.
- [9] R. F. Harrington, *Time-Harmonic Electromagnetic Fields*. McGraw-Hill, 1961.
- [10] J. H. W. Johnson, *Generalized Moment Methods in Electromagnetics*. John Wiley and Sons, Inc., 1991.
- [11] R. F. Harrington, *Field Computations by Moments Methods*. MacMillan, 1968.
- [12] S. R. Rengarajan, "Analysis of compound radiating slots," Technical Report 886-001, University of California, Los Angeles, June 1988.

- [13] T. J. Cui and W. C. Chew, "Fast evaluation of sommerfeld integrals for em scattering and radiation by three-dimensional buried objects," *IEEE Transactions on Antennas and Propagation*, vol. AP-37, no. 2, pp. 887–900, 1999.
- [14] T. Itoh, *Numerical Techniques for Microwave and Millimeter-Wave Passive Structures*. John Wiley and Sons, Inc., 1989.
- [15] J. R. Mosig and F. E. Gardiol, "A dynamic radiation model for microstrip structures," *Advances in Electronics and Electron Physics*, vol. 59, pp. 139–237, 1982.
- [16] K. Forooghi, "Theoretical determination of self admittance for a longitudinal shunt slot radiating into a parallel plate region," Technical Report 57L, Chalmers University of Technology, Gothenburg, Sweden, June 1988.
- [17] IMSL, *User's Manual - FORTRAN subroutines for mathematical application*. IMSL Inc., 1991.
- [18] MathWorks, *MATLAB Function Reference, version 5*. The MathWorks Inc., 1999.
- [19] C. de Boor, *A Practical Guide to Splines*. Springer-Verlag, 1978.

Appendix A

The Equivalence Principle

A short description of the equivalence and image principle applied in the derivation of eq. 1.4a- 1.5b.

A.1. Equivalence Principle for an Open Half-Space

With the assumptions made in sect. 1.1, fields in the open half-space are represented by using a surface equivalence principle in conjunction with a free-space Green's function. Figure A.1(a) shows the original problem, and (b) and (c) each represent an equivalent problem as far as region 1 is concerned [10]. In the equivalent problem (b), we have the same source $(\mathbf{J}_i, \mathbf{M}_i)$ as in problem (a). However, in (b), the plane at $y = b$ is a perfect conducting surface including the aperture area A . The magnetic current \mathbf{M} over aperture A (now replaced by the conducting surface) is given by

$$\mathbf{M} = -\hat{\mathbf{n}} \times \mathbf{E}_1 = -\hat{\mathbf{n}} \times \mathbf{E}^a \quad (\text{A.1})$$

where $\hat{\mathbf{n}}$ is a unit normal vector directed into region 1, and \mathbf{E}^a is the electric field in the aperture area A .

Problem (b) is still difficult to solve because of the presence of the conducting plane at $y = b$. By using the image theory, we can transform problem (b) into problem (c), for which the fields due to the sources can be expressed and calculated by using the Green's function for an unbounded region. In (c) the total equivalent magnetic current \mathbf{M}_t equals $2\mathbf{M}$, and

$$\mathbf{M}_t = 2\mathbf{M} = -2\hat{\mathbf{n}} \times \mathbf{E}^a = 2\hat{\mathbf{y}} \times \mathbf{E}^a \quad \text{for fields in region 1} \quad (\text{A.2})$$

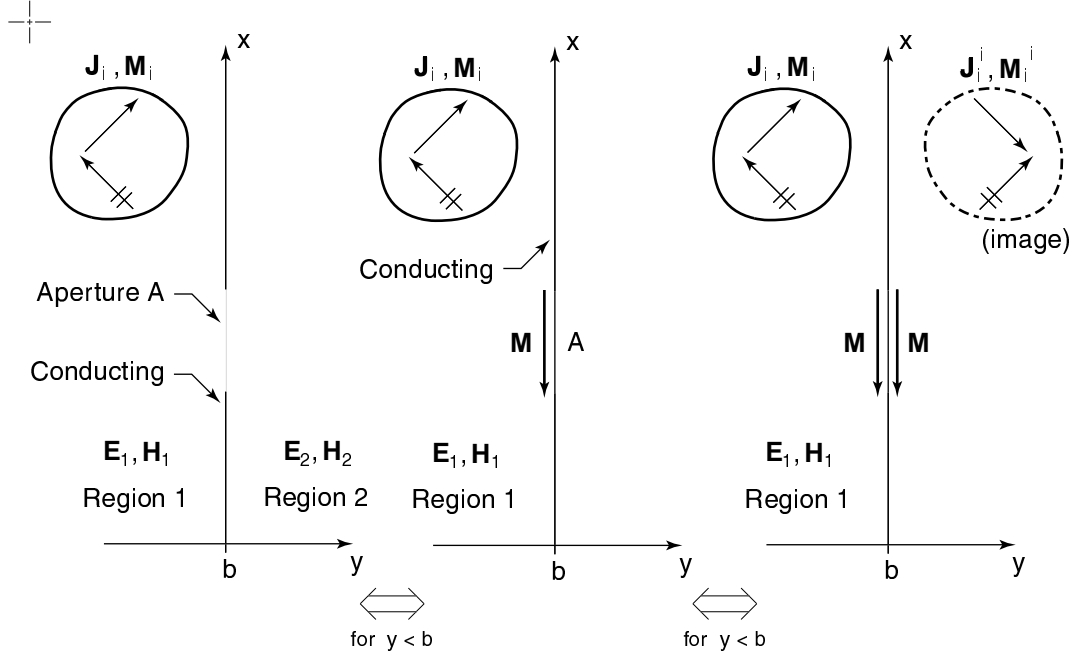


Figure A.1.: Equivalence for region 1 ($y < b$).

where the factor 2 is due to application of the image principle. The equivalence between (a), (b) and (c), as illustrated in figure A.1, is valid only for fields in region 1 ($y < b$). In other words, we have the same $(\mathbf{E}_1, \mathbf{H}_1)$ for ($y < b$) in (a), (b) and (c). However, because of symmetry in geometry, this equivalence can also be applied to the fields in region 2, in which case the total equivalent magnetic current \mathbf{M}_t on aperture A used to compute fields in region 2 is given by

$$\mathbf{M}_t = 2\mathbf{M} = -2\hat{\mathbf{n}} \times \mathbf{E}^a = -2\hat{\mathbf{y}} \times \mathbf{E}^a \quad \text{for fields in region 2} \quad (\text{A.3})$$

In addition, $(\mathbf{J}_i, \mathbf{M}_i)$, and its image, must be that in region 2.

Appendix B

Green's Functions for Rectangular Waveguides

In chap. 1 we encountered the problem finding the field distribution in the aperture of a radiating slot cut in the broad wall of a waveguide, when excitation is by an incident TE_{10} mode.

In general and specific for rectangular waveguides the transverse fields of a TE mode can be determined by taking prescribed spatial derivatives of its H_z component. For a rectangular waveguide filled with air, this longitudinal field component satisfy the homogeneous wave equation

$$(\nabla^2 + k^2)H_z(x, y, z) \equiv 0 \quad (\text{B.1})$$

wherein time variation of the form $e^{j\omega t}$ are implied and $k^2 = \omega^2\epsilon_0\mu_0$.

B.1. Green's Integral Theorems

Several integral theorems due to Green facilitate the solution of eq. B.1. However, we use the following important result

$$v(\bar{R}) = \oint_S \left(G \frac{\partial v}{\partial n'} - v \frac{\partial G}{\partial n'} \right) dS' \quad (\text{B.2})$$

where v is any scalar function and G is the Green's function.

If we impose the Dirichlet condition that $G(\bar{R}, \bar{R}') \equiv 0$ on S , then

$$v(\bar{R}) = - \oint_S v(\bar{R}') \frac{\partial G(\bar{R}, \bar{R}')}{\partial n'} dS' \quad (\text{B.3})$$

whereas if we impose the Neumann condition $\partial G/\partial n' \equiv 0$ on S , then

$$v(\bar{R}) = - \oint_S \frac{\partial v(\bar{R}')}{\partial n'} G(\bar{R}, \bar{R}') dS' \quad (\text{B.4})$$

B.2. Green's Theorems Applied to the Rectangular Waveguide

Now we can turn our attention specifically to the waveguide problem. If $H_z(P')$ is the longitudinal component of any electromagnetic field that can exist in R , then $H_z(\bar{R}')$ satisfies

$$(\nabla_s^2 + k^2)H_z \equiv 0 \quad (\text{B.5})$$

in R and thus is a candidate for $v(\bar{R}')$. Thus

$$H_z(\bar{R}) = \oint_S \frac{\partial H_z(\bar{R}')}{\partial n'} G_1(\bar{R}, \bar{R}') dS' \quad (\text{B.6})$$

in which the Green's function has the properties

- $G_1(\bar{R}, \bar{R}') = G_{c1}(\bar{R}, \bar{R}') + e^{-jkz(\bar{R}, \bar{R}')}/4\pi r(\bar{R}, \bar{R}')$
- $\partial G_1(\bar{R}, \bar{R}')/\partial n' \equiv 0$ on S
- $G_{c1}(\bar{R}, \bar{R}')$ is regular in R and satisfies $(\nabla_s^2 + k^2)G_1(\bar{R}, \bar{R}') = 0$ for all \bar{R}' and any \bar{R}
- $G_1(\bar{R}, \bar{R}')$ satisfies $(\nabla_s^2 + k^2)G_1(\bar{R}, \bar{R}') = -\delta(\bar{R} - \bar{R}')$

Since the dielectric is assumed to be slightly lossy, if the sources creating $H_z(\bar{R}')$ are all in a finite region, the contribution to eq. B.6 from end caps at $z = \pm\infty$ are nil and eq. B.6 becomes

$$H_z(\bar{R}) = \int_S \frac{\partial H_z(\bar{R}')}{\partial n'} G_1(\bar{R}, \bar{R}') dS' \quad (\text{B.7})$$

where by S we shall mean the skintight surface that lies up against the waveguide walls and is infinity long. The Neumann form eq. B.7 is the obvious choice for H_z since $\partial H_z(\bar{R}')/\partial n'$ will be zero at all points in S except the slot aperture region in the waveguide.

B.3. Transformation of the Integral for $H_z(\bar{R})$

With the perimeter coordinates s' defined as shown in figure B.1, the integral eq. B.7 can be converted to the more useful form

$$H_z(\bar{R}) = \frac{1}{j\omega\mu_0} \int_S \left[E_s(\bar{R}') \left(\frac{\partial^2}{\partial z'^2} + k^2 \right) G_1(\bar{R}, \bar{R}') - E_z(\bar{R}') \frac{\partial^2 G_1(\bar{R}, \bar{R}')}{\partial s' \partial z'} \right] dS' \quad (\text{B.8})$$

by the following procedure. The two components of Maxwell's equations are

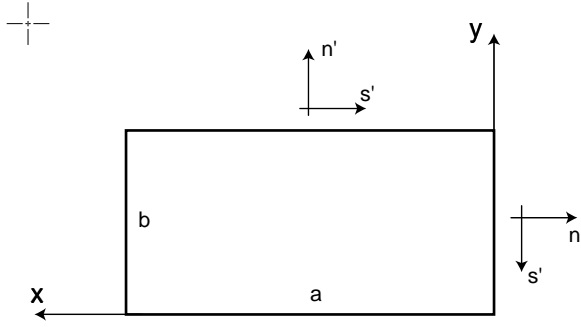


Figure B.1.: Definition of coordinates for rectangular waveguide.

$$\frac{\partial H_z}{\partial y'} - \frac{\partial H_y}{\partial z'} = (\sigma + j\omega\epsilon) E_x \quad (\text{B.9})$$

$$\frac{\partial E_x}{\partial z'} - \frac{\partial E_z}{\partial x'} = -j\omega\mu_0 H_y \quad (\text{B.10})$$

Differentiation of eq. B.10 yields

$$\frac{\partial H_y}{\partial z'} = -\frac{1}{j\omega\mu_0} \left(\frac{\partial^2 E_x}{\partial z'^2} - \frac{\partial^2 E_z}{\partial x' \partial z'} \right) \quad (\text{B.11})$$

When eq. B.9 and B.11 are combined, we obtain

$$\frac{\partial H_z}{\partial y'} = -\frac{1}{j\omega\mu_0} \left[\left(\frac{\partial^2}{\partial z'^2} + k^2 \right) E_x - \frac{\partial^2 E_z}{\partial x' \partial z'} \right] \quad (\text{B.12})$$

By a similar procedure we can show that

$$\frac{\partial H_z}{\partial x'} = \frac{1}{j\omega\mu_0} \left[\left(\frac{\partial^2}{\partial z'^2} + k^2 \right) E_y - \frac{\partial^2 E_z}{\partial y' \partial z'} \right] \quad (\text{B.13})$$

Now consider the normal and perimeter coordinates n' and s' shown in figure B.1. Along the upper broad wall, $E_x = -E_s$, $dx' = -ds'$, $dy' = dn'$, and eq. B.12 becomes

$$\frac{\partial H_z}{\partial n'} = \frac{1}{j\omega\mu_0} \left[\left(\frac{\partial^2}{\partial z'^2} + k^2 \right) E_s - \frac{\partial^2 E_z}{\partial s' \partial z'} \right] \quad (\text{B.14})$$

Along the lower broad wall, $E_x = E_s$, $dx' = ds'$, $dy' = -dn'$, and eq. B.12 once again becomes B.14. Along the right-side wall, $E_y = -E_s$, $dx' = -dn'$, $dy' = dn'$, and eq. B.13 becomes B.14. Finally, along the left-side wall, $E_y = E_s$, $dx' = dn'$, $dy' = ds'$, and eq. B.13 once again becomes B.14. Thus eq. B.14 is a valid representation for the normal derivative of H_z no matter which wall is being considered. This means that eq. B.8 can be rewritten as

$$H_z(\bar{R}) = \frac{1}{j\omega\mu_0} \int_S \left[\left(\frac{\partial^2}{\partial z'^2} + k^2 \right) E_s - \frac{\partial^2 E_z}{\partial s' \partial z'} \right] G_1(\bar{R}, \bar{R}') dS' \quad (\text{B.15})$$

This result can be converted to a desirable form through several integrations by parts. Consider

$$\begin{aligned} \int_S \frac{\partial^2 E_z}{\partial s' \partial z'} G_1 dS' &= \oint_C ds' \left(\int_{-\infty}^{\infty} G_1 \frac{\partial^2 E_z}{\partial s' \partial z'} dz' \right) \\ &= \oint_C ds' \left[\left(G_1 \frac{\partial E_z}{\partial s'} \Big|_{-\infty}^{\infty} - \int_{-\infty}^{\infty} \frac{\partial E_z}{\partial s'} \frac{\partial G_1}{\partial z'} dz' \right) \right] \end{aligned} \quad (\text{B.16})$$

where C is the transverse perimeter of the waveguide.

If the dielectric is even slightly lossy, both G_1 and $\partial E_z / \partial s'$ vanish at $z = \pm\infty$ and eq. B.16 becomes

$$\begin{aligned} - \int_{-\infty}^{\infty} dz' \left(\oint_C \frac{\partial G_1}{\partial z'} \frac{\partial E_z}{\partial s'} ds' \right) &= \\ &= - \int_{-\infty}^{\infty} dz' \left[\left(\frac{\partial G_1}{\partial z'} E_z \Big|_{-P_1}^{P_2} - \oint_C E_z \frac{\partial^2 G_1}{\partial s' \partial z'} \right) \right] \end{aligned} \quad (\text{B.17})$$

where P_1 and P_2 are the starting and ending points of the perimeter C . Since these are the same point, the term $\frac{\partial G_1}{\partial z'} E_z \Big|_{-P_1}^{P_2}$ makes no contribution and thus we obtain

the transformation

$$\int_S \frac{\partial^2 E_z}{\partial s' \partial z'} G_1 dS' = \int_S \frac{\partial^2 G_1}{\partial s' \partial z'} E_z dS' \quad (\text{B.18})$$

In a like manner, we can show that

$$\int_S \frac{\partial^2 E_s}{\partial z'^2} G_1 dS' = \int_S E_s \frac{\partial^2 G_1}{\partial z'^2} dS' \quad (\text{B.19})$$

and thus eq. B.15 transforms to eq. B.8. Since eq. B.15 is itself a transformation of eq. B.7, we have succeeded in demonstrating that eq. B.8 is a valid representation for the longitudinal component of magnetic field within the waveguide. The reason for choosing Neumann formulation for H_z is now clear, since the integrands in eq. B.8 will be zero except where there are apertures.

B.4. Explicit Expression for Green's functions G_1 for a Rectangular Waveguide

As a next step, we need to find specific expressions for $G_1(\bar{R}, \bar{R}')$. We select the generic form [8]

$$G_1(\bar{R}, \bar{R}') = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn} \frac{2}{\sqrt{ab}} \cos \frac{m\pi x'}{a} \cos \frac{n\pi y'}{a} F_{mn}(\bar{R}, z') \quad (\text{B.20})$$

wherein $\epsilon_{00} = 1/2$, $\epsilon_{m0} = \epsilon_{0n} = 1/\sqrt{2}$, and $\epsilon_{mn} = 1$ otherwise. This is a reasonable starting assumption because the individual terms satisfy the boundary conditions that $\partial G_1 / \partial n' \equiv 0$ on S .

Substituting of eq. B.20 in the inhomogeneous wave equation gives

$$\begin{aligned} & - \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \left[\left(\frac{m\pi}{a} \right)^2 + \left(\frac{n\pi}{b} \right)^2 \right] \epsilon_{mn} \frac{2}{\sqrt{ab}} \cos \frac{m\pi x'}{a} \cos \frac{n\pi y'}{b} F_{mn} \\ & + \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn} \frac{2}{\sqrt{ab}} \cos \frac{m\pi x'}{a} \cos \frac{n\pi y'}{b} \left(\frac{\partial^2 F_{mn}}{\partial z'^2} + k^2 F_{mn} \right) = -\delta(\bar{R} - \bar{R}') \end{aligned} \quad (\text{B.21})$$

If we multiply through by $\epsilon_{rs}(2/\sqrt{ab} \cos(r\pi x'/a) \cos(s\pi y'/b))$ and integrate over $[a, b]$ the result is

$$\frac{\partial^2 F_{rs}}{\partial z'^2} - \gamma_{rs}^2 F_{rs} = -\epsilon_{rs} \frac{2}{\sqrt{ab}} \cos \frac{r\pi x}{a} \cos \frac{s\pi y}{b} \delta(z - z') \quad (\text{B.22})$$

with γ_{rs}^2 defined as

$$\gamma_{rs}^2 = \left(\frac{r\pi}{a}\right)^2 + \left(\frac{s\pi}{b}\right)^2 - k^2 \quad (\text{B.23})$$

Thus

$$F_{rs}(z', \bar{R}) = -\epsilon_{rs} \frac{2}{\sqrt{ab}} \cos \frac{r\pi x}{a} \cos \frac{s\pi y}{b} f_{rs}(z, z') \quad (\text{B.24})$$

where

$$\frac{\partial^2 f_{rs}}{\partial z'^2} - \gamma_{rs}^2 f_{rs} = \delta(z - z') \quad (\text{B.25})$$

Now using the method of undertimed coefficients [8]

$$f_{rs}(z, z') = v_1(z, z') e^{\gamma_{rs} z'} + v_2(z, z') e^{-\gamma_{rs} z'} \quad (\text{B.26})$$

Then

$$\frac{\partial f_{rs}}{\partial z'} = \frac{\partial v_1}{\partial z'} e^{\gamma_{rs} z'} + \frac{\partial v_2}{\partial z'} e^{-\gamma_{rs} z'} + \gamma_{rs} v_1 e^{\gamma_{rs} z'} - \gamma_{rs} v_2 e^{-\gamma_{rs} z'} \quad (\text{B.27})$$

Letting

$$\frac{\partial v_1}{\partial z'} e^{\gamma_{rs} z'} + \frac{\partial v_2}{\partial z'} e^{-\gamma_{rs} z'} = 0 \quad (\text{B.28})$$

we obtain

$$\frac{\partial^2 f_{rs}}{\partial z'^2} = \gamma_{rs} \frac{\partial v_1}{\partial z'} e^{\gamma_{rs} z'} - \gamma_{rs} \frac{\partial v_2}{\partial z'} e^{-\gamma_{rs} z'} + \gamma_{rs}^2 v_1 e^{\gamma_{rs} z'} + \gamma_{rs}^2 v_2 e^{-\gamma_{rs} z'} \quad (\text{B.29})$$

Substitution of eq. B.26 and B.29 in eq. B.25 gives

$$\gamma_{rs} \frac{\partial v_1}{\partial z'} e^{\gamma_{rs} z'} - \gamma_{rs} \frac{\partial v_2}{\partial z'} e^{-\gamma_{rs} z'} = \delta(z - z') \quad (\text{B.30})$$

Equation B.28 and B.30 must be satisfied simultaneously by $\partial v_1/\partial z'$ and $\partial v_2/\partial z'$.

Thus

$$\frac{\partial v_1}{\partial z'} = \frac{e^{-\gamma_{rs} z'} \delta(z - z')}{2\gamma_{rs}} \quad (\text{B.31})$$

$$\frac{\partial v_2}{\partial z'} = -\frac{e^{\gamma_{rs} z'} \delta(z - z')}{2\gamma_{rs}} \quad (\text{B.32})$$

For a given z , v_1 and v_2 as a function of z' have zero slope except at $z' = z$ Therefore

$$v_1(z, z') = \begin{cases} c_1(z) & z' < z \\ 0 & z' > z \end{cases}$$

$$v_2(z, z') = \begin{cases} 0 & z' < z \\ c_2(z) & z' > z \end{cases}$$

which fulfill the requirement that f_{rs} as it appears in eq. B.26 should be well behaved at $z' = \pm\infty$. We can find $c_1(z)$ and $c_2(z)$ as follows

$$\int_{z'}^{\infty} \frac{\partial v_1}{\partial z'} dz' = \int_{z'}^{\infty} \frac{e^{-\gamma_{rs} z'}}{2\gamma_{rs}} \delta(z - z') dz' \quad z' < z \quad (\text{B.33})$$

$$v_1(z, \infty) - v_1(z, z') = \frac{e^{-\gamma_{rs} z'}}{2\gamma_{rs}}$$

$$v_1(z, z') = c_1(z) = -\frac{e^{-\gamma_{rs} z'}}{2\gamma_{rs}} \quad z' < z$$

Similarly

$$v_2(z, z') = c_2(z) = -\frac{e^{\gamma_{rs} z'}}{2\gamma_{rs}} \quad z' > z \quad (\text{B.34})$$

Thus

$$f_{rs}(z, z') = \begin{cases} -\frac{e^{-\gamma_{rs} z'}}{2\gamma_{rs}} & z' < z \\ -\frac{e^{\gamma_{rs} z'}}{2\gamma_{rs}} & z' > z \end{cases} \quad (\text{B.35})$$

or

$$f_{rs}(z, z') = -\frac{e^{-\gamma_{rs}|z-z'|}}{2\gamma_{rs}} \quad \text{All } z, z' \quad (\text{B.36})$$

Substitution of eq. B.36 in B.24 gives

$$F_{rs}(z', \bar{R}) = \epsilon_{rs} \frac{2}{\sqrt{ab}} \cos \frac{r\pi x}{a} \cos \frac{s\pi y}{b} \frac{e^{-\gamma_{rs}|z-z'|}}{2\gamma_{rs}} \quad (\text{B.37})$$

From this it follows that

$$G_1(\bar{R}, \bar{R}') = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \Psi(x', y') \Psi(x, y) \frac{e^{-\gamma_{mn}|z-z'|}}{2\gamma_{mn}} \quad (\text{B.38})$$

with

$$\Psi(x, y) = \epsilon_{mn} \frac{2}{\sqrt{ab}} \cos \frac{m\pi x}{a} \cos \frac{n\pi y}{b} \quad (\text{B.39})$$

Equation B.38 is Stevenson's H -type Green's function for a rectangular waveguide [8].

B.5. An Integral Equation for $H_x(\bar{R})$

Analysis of a transverse slot in the broad wall of a waveguide requires knowledge of the x -component of the magnetic field. Since $H_x(\bar{R})$ satisfies the homogeneous wave equation it replaces $v(\bar{R})$ in eq. B.2. We can then choose a Green's function $G_2(\bar{R}, \bar{R}')$ to satisfy a hybrid of Dirichlet and Neumann conditions by requiring that G_2 be zero at $x = 0, a$ and that $\partial G_2 / \partial n'$ be zero at $y = 0, b$. This results in the expression

$$G_2(\bar{R}, \bar{R}') = \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \phi_{mn}(x, y) \phi_{mn}(x', y') \frac{e^{-\gamma_{mn}|\zeta - \zeta'|}}{2\gamma_{mn}} \quad (\text{B.40})$$

in which

$$\phi_{mn}(x, y) = \epsilon_{mn} \frac{2}{\sqrt{ab}} \sin \frac{m\pi x}{a} \cos \frac{n\pi y}{b} \quad (\text{B.41})$$

For a slot in the broad wall eq. B.2 reduces in this case to

$$H_x(\bar{R}) = \int_S G_2(\bar{R}, \bar{R}') \frac{\partial H_x(\bar{R}')}{\partial n'} dS' \quad (\text{B.42})$$

Maxwell's curl equations and two integrations by parts convert eq. B.42 to

$$H_x(\bar{R}) = \pm \frac{1}{j\omega\mu_0} \int_S \left[E_z(\bar{R}') \left(\frac{\partial^2}{\partial x'^2} + k^2 \right) G_2(\bar{R}, \bar{R}') \right. \\ \left. - E_x(\bar{R}') \frac{\partial^2 G_2(\bar{R}, \bar{R}')}{\partial x' \partial z'} \right] dS' \quad (\text{B.43})$$

In eq. B.43 the upper sign is used if the point \bar{R} is in the upper wall of the waveguide and the lower sign is used if it is in the lower wall.

Appendix C

Matrix Elements

In what follows, all the matrix terms used to solve for the electric fields in a slot aperture are simplified for ease of computation. These are made as general as possible and applies to both a longitudinal, transverse and a T-slot problems.

C.1. Self Terms for Longitudinal Part of T-Slot

Applying eq. 1.25b and 2.11a to eq. 2.16a gives the following expression for the interior field contribution to the matrix element number (p, q)

$$\begin{aligned}
 Y_{pq}^{int} &= \langle H_z^{int}(E_x), w_q \rangle \frac{Z_0}{E_p} \tag{C.1} \\
 &= \frac{-2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \\
 &\quad \cdot \cos \frac{m\pi(x_0 + \xi)}{a} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \cos \frac{m\pi(x_0 + \xi')}{a} \\
 &\quad \cdot \left[k^2 + \frac{\partial^2}{\partial \zeta'^2} \right] e^{-\gamma_{mn}|\zeta - \zeta'|} d\zeta' d\xi' d\zeta d\xi
 \end{aligned}$$

Interchanging integration and summation and evaluation of the integrals with respect to ξ and ξ' yields

$$Y_{pq}^{int} = \frac{-\lambda w}{j ab \pi} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} W_m \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \tag{C.2}$$

$$\cdot \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] \left[k^2 + \frac{\partial^2}{\partial \zeta'^2} \right] e^{-\gamma_{mn}|\zeta - \zeta'|} d\zeta' d\zeta$$

where we have used the integral formula

$$\int_{-w/2}^{w/2} \cos \frac{m\pi(x_0 + x)}{a} dx = \frac{2a}{m\pi} \sin \frac{m\pi w}{2a} \cos \frac{m\pi x_0}{a} \quad (\text{C.3})$$

and the quantity W_m is given by

$$W_m = \text{sinc} \frac{m\pi w}{2a} \cos^2 \frac{m\pi x_0}{a} \quad (\text{C.4})$$

The derivative is evaluated using the delta function concept and we write it in the following form

$$\frac{\partial^2}{\partial \zeta'^2} e^{-\gamma_{mn}|\zeta - \zeta'|} = \gamma_{mn}^2 e^{-\gamma_{mn}|\zeta - \zeta'|} - 2\gamma_{mn} \delta(\zeta - \zeta') \quad (\text{C.5})$$

Thereby eq. C.2 reduces to

$$Y_{pq}^{int} = \frac{-\lambda w}{j a b \pi} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 W_m \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \frac{k^2 + \gamma_{mn}^2}{\gamma_{mn}} \cdot \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta') \right] e^{-\gamma_{mn}|\zeta - \zeta'|} d\zeta' - 2 \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta) \right] d\zeta \quad (\text{C.6})$$

Evaluating the inner integral we obtain the following expression

$$Y_{pq}^{int} = \frac{j\lambda w}{ab\pi} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 W_m \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \left[\frac{k^2 + \gamma_{mn}^2}{\gamma_{mn}} \cdot \frac{1}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z} \right)^2} \left[\frac{p\pi}{2L_z} \left(e^{-\gamma_{mn}(L_z + \zeta)} \pm e^{-\gamma_{mn}(L_z - \zeta)} \right) + 2\gamma_{mn} \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta) \right] \right] - 2 \sin \left[\frac{p\pi}{2L_z} (L_z + \zeta) \right] \right] d\zeta \quad (\text{C.7})$$

where the upper sign is valid for p odd and the lower sign for p even. Evaluating the last integral leads to the final expression for the interior field contribution to the

matrix element number (p, q)

$$Y_{pq}^{int} = \frac{2j\lambda w}{ab\pi} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 W_m \left[\frac{\frac{p\pi}{2L_z} \gamma_{mn}^2 + k^2}{\gamma_{mn} \gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2} \frac{\frac{\pi q}{2L_z}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2} \right. \\ \left. \cdot (1 \pm e^{-2\gamma_{mn}L_z}) + \frac{k^2 - \left(\frac{p\pi}{2L_z}\right)^2}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2} L_z \delta_{pq} \right] \quad (C.8)$$

where $\delta_{pq} = 1$ for $p = q$, 0 otherwise. The upper sign is valid for both p and q odd, the lower sign for both p and q even. When p is odd and q is even or vice versa $Y_{pq}^{int} = 0$ because of the orthogonality of the sinusoidal functions. Beyond the diagonal terms it is seen from symmetry, that it is only necessary to compute the terms on the lower half of the diagonal.

In a similar way, applying eq. 1.24c and 2.11a to eq. 2.16a gives the following expression for the exterior field contribution to the matrix element number (p, q)

$$Y_{pq}^{ext} = \langle H_z^{ext}(E_x), w_q \rangle \frac{Z_0}{E_p} \quad (C.9) \\ = \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \delta(\xi) \\ \cdot \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \left[k^2 + \frac{\partial^2}{\partial \zeta'^2} \right] \frac{e^{-jkR}}{R} d\zeta' d\xi' d\zeta d\xi$$

The integration with respect to ξ is done immediately. Next let us consider the ζ' integration by parts

$$\int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \left[k^2 + \frac{\partial^2}{\partial \zeta'^2} \right] \frac{e^{-jkR}}{R} d\zeta' \\ = \int_{-L_z}^{L_z} \left[k^2 - \left(\frac{p\pi}{2L_z} \right)^2 \right] \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \frac{e^{-jkR}}{R} d\zeta' \\ - \frac{\pi p}{2L_z} \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] \quad (C.10)$$

where

$$R = \sqrt{\xi'^2 + (\zeta - \zeta')^2} \quad (\text{C.11a})$$

$$R_1 = \sqrt{\xi'^2 + (\zeta - L_z)^2} \quad (\text{C.11b})$$

$$R_2 = \sqrt{\xi'^2 + (\zeta + L_z)^2} \quad (\text{C.11c})$$

Using Eq. C.10, Eq. C.9 becomes

$$\begin{aligned} Y_{pq}^{ext} = & \frac{Z_0}{2\pi j\omega\mu_0} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \left[k^2 - \left(\frac{p\pi}{2L_z} \right)^2 \right] \\ & \cdot \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \frac{e^{-jkR}}{R} d\zeta' \\ & - \frac{p\pi}{2L_z} \left[(-1)^p \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\xi' d\zeta \end{aligned} \quad (\text{C.12})$$

which is written as

$$Y_{pq}^{ext} = \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} [I_1 + I_2] d\xi' \quad (\text{C.13})$$

The object is to reduce Y_{pq}^{ext} from a triple integral to numerically more efficient double integrals. An integration scheme solving this problem is addressed in sect. 2.2.3, and only the result is given here

$$\begin{aligned} Y_{pq}^{ext} = & \frac{2Z_0}{j\pi^2\omega\mu_0(q^2 - p^2)} \left[[(k2L_z)^2 - (q\pi)^2] \cdot p \cdot y_1(q) \right. \\ & \left. - [(k2L_z)^2 - (p\pi)^2] \cdot q \cdot y_1(p) \right] \quad p \neq q \end{aligned} \quad (\text{C.14})$$

and

$$Y_{pq}^{ext} = \frac{Z_0}{j\pi\omega\mu_0} [(k2L_z)^2 - (q\pi)^2] \cdot y_2(q) - \frac{2Z_0q}{j\omega\mu_0} y_1(q) \quad p = q \quad (\text{C.15})$$

where the two functions $y_1(p)$ and $y_2(p)$ are given in eq. 2.64 and 2.65. Like the interior terms, $Y_{pq}^{ext} = 0$ when p is odd and q is even or vice versa. Similar, the symmetry properties of Y_{pq}^{int} applies Y_{pq}^{ext} .

When implementing a numerical integration routine it should be noticed that the integrands have singularities at $R_1 = 0$ which occurs when $\sigma = 1/2$ and at $R = 0$ which occurs when $\xi' = 0$, $\nu = 0$. In order to achieve a correct result and to make sure that the program runs stable, these points must be extracted.

C.2. Self Terms for Transverse Part of T-Slot

Applying eq. 1.25a and 2.11b to eq. 2.16d gives the following expression for the interior field contribution to the matrix element number (r, s)

$$\begin{aligned}
 Y_{rs}^{int} &= \langle H_x^{int}(E_z), w_s \rangle \frac{Z_0}{E_r} & (C.16) \\
 &= \frac{2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \\
 &\quad \cdot \sin \left[\frac{m\pi}{a}(x_{00} + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] e^{-\gamma_{mn}|\zeta - \zeta'|} \\
 &\quad \cdot \left[k^2 + \frac{\partial^2}{\partial \xi'^2} \right] \sin \left[\frac{m\pi}{a}(x_{00} + \xi') \right] d\xi' d\zeta' d\xi d\zeta
 \end{aligned}$$

Differentiation with respect to ξ' gives

$$\begin{aligned}
 Y_{rs}^{int} &= \frac{-2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} & (C.17) \\
 &\quad \cdot \sin \left[\frac{m\pi}{a}(x_{00} + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \sin \left[\frac{m\pi}{a}(x_{00} + \xi') \right] \\
 &\quad \cdot \left[\left(\frac{m\pi}{a} \right)^2 - k^2 \right] e^{-\gamma_{mn}|\zeta - \zeta'|} d\xi' d\zeta' d\xi d\zeta
 \end{aligned}$$

and interchanging summation and integration and evaluating the two ξ integrals yields

$$\begin{aligned}
 Y_{rs}^{int} &= \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \frac{\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2} \frac{\frac{r\pi}{L_x}}{\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2} & (C.18) \\
 &\quad \cdot \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right]_{s \text{ odd}} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right]_{r \text{ odd}} \\
 &\quad \cdot \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right]_{s \text{ even}} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right]_{r \text{ even}}
 \end{aligned}$$

$$\cdot \left[\left(\frac{m\pi}{a} \right)^2 - k^2 \right] \int_{-w/2}^{w/2} \delta(\zeta) \int_{-w/2}^{w/2} e^{-\gamma_{mn}|\zeta-\zeta'|} d\zeta' d\zeta$$

where we have used the following integration scheme

$$\begin{aligned} & \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \sin \left[\frac{m\pi}{a} (x_{00} + \xi) \right] d\xi \\ &= \frac{\frac{s\pi}{4L_x}}{\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2} \left[\sin \left[\frac{m\pi}{a} \left(\frac{sa}{m} - L_x - x_{00} \right) \right] \right. \\ & \quad \left. - \sin \left[\frac{m\pi}{a} \left(\frac{sa}{m} + L_x + x_{00} \right) \right] - 2 \sin \left[\frac{m\pi}{a} (L_x - x_{00}) \right] \right] \end{aligned} \quad (C.19)$$

which by use of trigonometric identities reduces to

$$\frac{\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} s \text{ odd} \\ s \text{ even} \end{array} \quad (C.20)$$

The last two ζ integrals evaluates to

$$\begin{aligned} & \int_{-w/2}^{w/2} \delta(\zeta) \int_{-w/2}^{w/2} e^{-\gamma_{mn}|\zeta-\zeta'|} d\zeta' d\zeta \\ &= \int_{-w/2}^{w/2} \delta(\zeta) \left[\int_{-w/2}^{\zeta} e^{-\gamma_{mn}(\zeta-\zeta')} d\zeta' + \int_{\zeta}^{w/2} e^{\gamma_{mn}(\zeta-\zeta')} d\zeta' \right] d\zeta \\ &= \int_{-w/2}^{w/2} \delta(\zeta) \frac{1}{\gamma_{mn}} \left[2 - e^{-\gamma_{mn}w/2} \left(e^{\gamma_{mn}\zeta} + e^{-\gamma_{mn}\zeta} \right) \right] d\zeta \\ &= \frac{2}{\gamma_{mn}} \left[1 - e^{-\gamma_{mn}w/2} \right] \end{aligned} \quad (C.21)$$

Inserting this into Y_{rs}^{int} gives the final expression

$$\begin{aligned} Y_{rs}^{int} &= \frac{-4Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}^2} \frac{\frac{s\pi}{L_x} \frac{r\pi}{L_x} \left[\left(\frac{m\pi}{a} \right)^2 - k^2 \right] \left[1 - e^{-\gamma_{mn}w/2} \right]}{\left[\left(\frac{s\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2 \right] \left[\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{\pi}{a} \right)^2 \right]} \\ & \cdot \left[\begin{array}{l} \cos^2 \frac{m\pi L_x}{a} \sin^2 \frac{m\pi x_{00}}{a} \\ \cos^2 \frac{m\pi x_{00}}{a} \sin^2 \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r, s \text{ odd} \\ r, s \text{ even} \end{array} \end{aligned}$$

Like Y_{pq}^{int} , Y_{rs}^{int} is symmetric and due to the orthogonality of the sinusoidal functions, $Y_{rs}^{int} = 0$ when p is odd and q is even or vice versa.

Derivation of the exterior contribution follows very much the procedure outlined in the section C.1. However, the following will give all details with the right variables substituted. Weighting the magnetic field $H_x^{ext}(E_r)$ with the function w_s gives directly the exterior field contribution to the matrix element number (r, s)

$$Y_{rs}^{ext} = \langle H_x^{ext}(E_z), w_s \rangle \frac{Z_0}{E_r} \quad (C.22)$$

$$\begin{aligned} &= \frac{-Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \\ &\quad \cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \left[k^2 + \frac{\partial^2}{\partial \xi'^2} \right] \frac{e^{-jkR}}{R} d\xi' d\zeta' d\xi d\zeta \end{aligned} \quad (C.23)$$

The integration with respect to ζ is done immediately and the ξ' integration is done by parts

$$\begin{aligned} &\int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \left[k^2 + \frac{\partial^2}{\partial \xi'^2} \right] \frac{e^{-jkR}}{R} d\xi' \\ &= \int_{-L_x}^{L_x} \left[k^2 - \left(\frac{\pi r}{2L_x} \right)^2 \right] \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \frac{e^{-jkR}}{R} d\xi' \\ &\quad - \frac{\pi r}{2L_x} \left[(-1)^r \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] \end{aligned} \quad (C.24)$$

where

$$R = \sqrt{\zeta'^2 + (\xi - \xi')^2} \quad (C.25a)$$

$$R_1 = \sqrt{\zeta'^2 + (\xi - L_x)^2} \quad (C.25b)$$

$$R_2 = \sqrt{\zeta'^2 + (\xi + L_x)^2} \quad (C.25c)$$

Using Eq. C.24, Eq. C.24 becomes

$$Y_{rs}^{ext} = \frac{-Z_0}{2\pi j\omega\mu_0} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \left[k^2 - \left(\frac{r\pi}{2L_x} \right)^2 \right]$$

$$\begin{aligned} & \cdot \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \frac{e^{-jkR}}{R} d\xi' \\ & - \frac{r\pi}{2L_x} \left[(-1)^r \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\xi' d\xi \end{aligned} \quad (\text{C.26})$$

which is written as

$$Y_{rs}^{ext} = \frac{-Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} [I_1 + I_2] d\xi' \quad (\text{C.27})$$

Once more, the object is to reduce Y_{rs}^{ext} from a triple integral to numerically more efficient double integrals. Applying the same integration scheme, as we did in section C.1, we end up with the following result

$$\begin{aligned} Y_{rs}^{ext} = \frac{-2Z_0}{j\pi^2\omega\mu_0(s^2 - r^2)} & \left[[(k2L_x)^2 - (s\pi)^2] \cdot r \cdot y_1(s) \right. \\ & \left. - [(k2L_x)^2 - (r\pi)^2] \cdot s \cdot y_1(r) \right] \quad r \neq s \end{aligned} \quad (\text{C.28})$$

and

$$Y_{rs}^{ext} = \frac{-Z_0}{j\pi\omega\mu_0} [(k2L_x)^2 - (s\pi)^2] \cdot y_2(s) - \frac{2Z_0s}{j\omega\mu_0} y_1(s) \quad r = s \quad (\text{C.29})$$

where the two functions $y_1(r)$ and $y_2(r)$ are given by

$$y_1(r) = \int_0^{w/2} \int_0^{1/2} \sin[\pi r(\sigma + 1/2)] \cdot \left[(-1)^r \frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\sigma d\xi' \quad (\text{C.30})$$

and

$$y_2(r) = \int_0^{w/2} \int_0^1 \frac{e^{-jkR}}{R} \left[\cos(r\pi\nu)(1 - \nu) + \frac{\sin(r\pi\nu)}{r\pi} \right] d\nu d\xi' \quad (\text{C.31})$$

On the basis of the variable substitutions the three quantities R , R_1 and R_2 given in Eq. C.25a, C.25b and C.25c are now written as

$$R = \sqrt{\zeta'^2 + \nu^2 4L_x^2} \quad (\text{C.32a})$$

$$R_1 = \sqrt{\zeta'^2 + (\sigma - 1/2)^2 4L_x^2} \quad (\text{C.32b})$$

$$R_2 = \sqrt{\zeta'^2 + (\sigma + 1/2)^2 4L_x^2} \quad (\text{C.32c})$$

Again, $Y_{rs}^{ext} = 0$ when r is odd and s is even or vice versa. The symmetry properties applies to Y_{rs}^{ext} as well.

It is noticed, the integrands have singularities at $R_1 = 0$ which occurs when $\sigma = 1/2$ and at $R = 0$ which occurs when $\zeta' = 0$, $\nu = 0$.

C.3. Cross Coupling Terms

The interior field contribution to the matrix element number (p, s) is found by applying eq. 1.25b and 2.11a to eq. 2.16b

$$\begin{aligned}
 Y_{rq}^{int} &= \langle H_z^{int}(E_z), w_q \rangle \frac{Z_0}{E_r} & (C.33) \\
 &= \frac{2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \\
 &\quad \cdot \cos \left[\frac{m\pi}{a} (x_0 + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \\
 &\quad \cdot \frac{\partial^2}{\partial \xi' \partial \zeta'} \cos \left[\frac{m\pi}{a} (x_{00} + \xi') \right] e^{-\gamma_{mn} |\zeta - \zeta'|} d\xi' d\zeta' d\zeta d\xi
 \end{aligned}$$

Differentiation with respect to ξ' and ζ' and interchanging summation and integration gives

$$\begin{aligned}
 Y_{rq}^{int} &= \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{m\pi}{a} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \delta(\xi) & (C.34) \\
 &\quad \cdot \cos \left[\frac{m\pi}{a} (x_0 + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x} (L_x + \xi') \right] \\
 &\quad \cdot \sin \left[\frac{m\pi}{a} (x_{00} + \xi') \right] \left[\pm e^{-\gamma_{mn} |\zeta - \zeta'|} \right] d\xi' d\zeta' d\zeta d\xi
 \end{aligned}$$

where the upper sign is valid when $\zeta > \zeta'$ and lower sign when $\zeta < \zeta'$. Using eq. C.20, the ξ' and ξ integrals evaluates to

$$Y_{rq}^{int} = \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \cos \frac{m\pi x_0}{a} \quad (C.35)$$

$$\begin{aligned} & \cdot \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2} \begin{bmatrix} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{bmatrix} \begin{matrix} r \text{ odd} \\ r \text{ even} \end{matrix} \\ & \cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \left[\pm e^{-\gamma_{mn} |\zeta - \zeta'|} \right] d\zeta' d\zeta \end{aligned}$$

The problem is now separated into three cases depending on the value of ζ as indicated in figure C.1. In case one, the ζ' integral in eq. C.35 is rewritten as

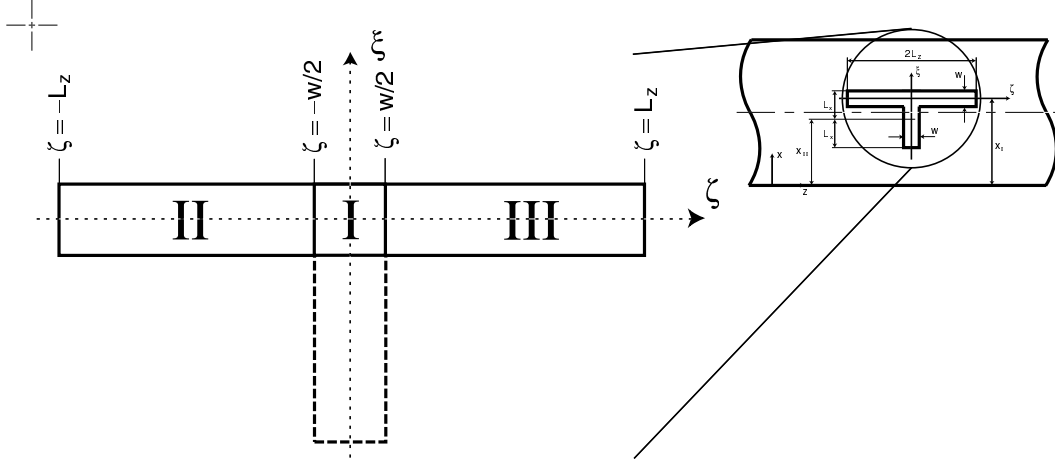


Figure C.1.: Separating the weighting integral into three cases depending on the value of ζ .

$$\begin{aligned} & \int_{-w/2}^{\zeta} e^{-\gamma_{mn}(\zeta - \zeta')} d\zeta' - \int_{\zeta}^{w/2} e^{\gamma_{mn}(\zeta - \zeta')} d\zeta' \quad (C.36) \\ & = \frac{1}{\gamma_{mn}} \left[e^{\gamma_{mn}(\zeta - w/2)} - e^{-\gamma_{mn}(\zeta + w/2)} \right] \end{aligned}$$

Noticing the limits for ζ have changed, the last integral reduces to

$$\begin{aligned} & \int_{-w/2}^{w/2} \frac{1}{\gamma_{mn}} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \left[e^{\gamma_{mn}(\zeta - w/2)} - e^{-\gamma_{mn}(\zeta + w/2)} \right] d\zeta \quad (C.37) \\ & = \frac{1}{\gamma_{mn}} \frac{\frac{1}{L_z} \cos \frac{q\pi}{2}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2} \left[2\gamma_{mn} L_z \sin \frac{q\pi w}{4L_z} \left[e^{-\gamma_{mn} w} + 1 \right] \right] \end{aligned}$$

$$+ \cos \frac{q\pi w}{4L_z} \left[e^{-\gamma_{mn}w} - 1 \right]$$

Inserting this into eq. C.35 gives the final expression for case one

$$\begin{aligned}
 Y_{rq}^{int,1} &= \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x} \right)^2 - \left(\frac{m\pi}{a} \right)^2 \right]} \\
 &\quad \cdot \frac{\frac{1}{L_z} \cos \frac{q\pi}{2}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z} \right)^2 \right]} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ - \cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r \text{ odd} \\ r \text{ even} \end{array} \\
 &\quad \cdot \left[2\gamma_{mn} L_z \sin \frac{q\pi w}{4L_z} \left[e^{-\gamma_{mn}w} + 1 \right] + \cos \frac{q\pi w}{4L_z} \left[e^{-\gamma_{mn}w} - 1 \right] \right]
 \end{aligned} \tag{C.38}$$

where $Y_{rq}^{int,1} = 0$ when q is odd.

In case two, the ζ' integral in eq. C.35 is rewritten as

$$\begin{aligned}
 &\int_{-w/2}^{w/2} -e^{\gamma_{mn}(\zeta-\zeta')} d\zeta' \\
 &= \frac{1}{\gamma_{mn}} \left[e^{\gamma_{mn}(\zeta-w/2)} - e^{\gamma_{mn}(\zeta+w/2)} \right]
 \end{aligned} \tag{C.39}$$

Changing the limits for ζ corresponding to the values given in figure C.1, the last integral reduces to

$$\begin{aligned}
 &\int_{-L_z}^{-w/2} \frac{1}{\gamma_{mn}} \sin \left[\frac{q\pi}{2L_z} (L_z + \zeta) \right] \left[e^{\gamma_{mn}(\zeta-w/2)} - e^{\gamma_{mn}(\zeta+w/2)} \right] d\zeta \\
 &= \frac{1}{\gamma_{mn}} \frac{\frac{1}{2L_z}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z} \right)^2} \left[\left[1 - e^{-\gamma_{mn}w} \right] \left[q\pi \cos \left[\frac{q\pi}{4L_z} (2L_z - w) \right] \right. \right. \\
 &\quad \left. \left. - 2\gamma_{mn} L_z \sin \left[\frac{q\pi}{4L_z} (2L_z - w) \right] \right] + q\pi \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right] \right]
 \end{aligned} \tag{C.40}$$

Inserting this into eq. C.35 gives the final expression for case two

$$\begin{aligned}
 Y_{rq}^{int,2} = & \frac{-2Z_0}{j\omega\mu_0ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2\right]} \quad (C.41) \\
 & \cdot \frac{\frac{1}{2L_z}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2\right]} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \end{array} \right]_{r \text{ odd}} \\
 & \left[-\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \right]_{r \text{ even}} \\
 & \cdot \left[[1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \left[\frac{q\pi}{4L_z}(2L_z - w) \right] - 2\gamma_{mn}L_z \right. \right. \\
 & \left. \left. \cdot \sin \left[\frac{q\pi}{4L_z}(2L_z - w) \right] \right] + q\pi \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right] \right]
 \end{aligned}$$

Case three is very much like case two except for some sign changes. The ζ' integral in eq. C.35 is rewritten as

$$\begin{aligned}
 & \int_{-w/2}^{w/2} e^{-\gamma_{mn}(\zeta-\zeta')} d\zeta' \quad (C.42) \\
 = & \frac{1}{\gamma_{mn}} \left[e^{-\gamma_{mn}(\zeta-w/2)} - e^{-\gamma_{mn}(\zeta+w/2)} \right]
 \end{aligned}$$

Again, changing the limits for ζ corresponding to the values given in figure C.1, the last ζ integral reduces to

$$\begin{aligned}
 & \int_{w/2}^{L_z} \frac{1}{\gamma_{mn}} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \left[e^{-\gamma_{mn}(\zeta-w/2)} - e^{-\gamma_{mn}(\zeta+w/2)} \right] d\zeta \quad (C.43) \\
 = & \frac{1}{\gamma_{mn}} \frac{\frac{1}{2L_z}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2} \left[[1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \left[\frac{q\pi}{4L_z}(2L_z + w) \right] + 2\gamma_{mn}L_z \right. \right. \\
 & \left. \left. \cdot \sin \left[\frac{q\pi}{4L_z}(2L_z + w) \right] \right] + q\pi(-1)^q \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right] \right]
 \end{aligned}$$

Inserting this into eq. C.35 gives the final expression for case three

$$\begin{aligned}
 Y_{rq}^{int,3} = & \frac{-2Z_0}{j\omega\mu_0ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2\right]} \\
 & \cdot \frac{\frac{1}{2L_z}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2\right]} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r \text{ odd} \\ r \text{ even} \end{array} \\
 & \cdot \left[[1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \left[\frac{q\pi}{4L_z}(2L_z + w) \right] + 2\gamma_{mn}L_z \right. \right. \\
 & \left. \left. \cdot \sin \left[\frac{q\pi}{4L_z}(2L_z + w) \right] \right] + q\pi(-1)^q \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right] \right]
 \end{aligned} \tag{C.44}$$

Now, since $Y_{rq}^{int,2}$ and $Y_{rq}^{int,3}$ are very similar, the two expressions are collected into one. Identifying the difference as

$$\begin{aligned}
 & [1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \left[\frac{q\pi}{4L_z}(2L_z - w) \right] \right. \\
 & \left. - 2\gamma_{mn}L_z \sin \left[\frac{q\pi}{4L_z}(2L_z - w) \right] \right] + q\pi \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right] \\
 & + [1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \left[\frac{q\pi}{4L_z}(2L_z + w) \right] \right. \\
 & \left. + 2\gamma_{mn}L_z \sin \left[\frac{q\pi}{4L_z}(2L_z + w) \right] \right] + q\pi(-1)^q \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right]
 \end{aligned} \tag{C.45}$$

which by use of trigonometric identities is rewritten as

$$\begin{aligned}
 & 2 \cos \frac{q\pi}{2} [1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \frac{q\pi w}{4L_z} + 2\gamma_{mn}L_z \sin \frac{q\pi w}{4L_z} \right] \\
 & + q\pi [1 + (-1)^q] \left[e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)} \right]
 \end{aligned}$$

this joint expression becomes

$$Y_{rq}^{int,23} = \frac{-2Z_0}{j\omega\mu_0ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \cos \frac{m\pi x_0}{a} \frac{\frac{m\pi}{a} \frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2\right]} \tag{C.46}$$

$$\begin{aligned} & \cdot \frac{1}{L_z} \left[\begin{array}{l} \cos \frac{m\pi L_x}{a} \sin \frac{m\pi x_{00}}{a} \\ -\cos \frac{m\pi x_{00}}{a} \sin \frac{m\pi L_x}{a} \end{array} \right] \begin{array}{l} r \text{ odd} \\ r \text{ even} \end{array} \\ & \cdot \left[\cos \frac{q\pi}{2} [1 - e^{-\gamma_{mn}w}] \left[q\pi \cos \frac{q\pi w}{4L_z} + 2\gamma_{mn}L_z \sin \frac{q\pi w}{4L_z} \right] \right. \\ & \left. + q\pi [e^{-\gamma_{mn}(L_z+w/2)} - e^{-\gamma_{mn}(L_z-w/2)}] \right] \end{aligned}$$

where $Y_{rq}^{int,23} = 0$ when q is odd.

The exterior contribution to the matrix element number (r, q) is found by weighting the magnetic field $H_z^{ext}(E_z)$ with the function w_q

$$Y_{rq}^{ext} = \langle H_z^{ext}(E_z), w_q \rangle \frac{Z_0}{E_r} \quad (C.47)$$

$$\begin{aligned} & = \frac{-Z_0}{2\pi j \omega \mu_0} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \delta(\xi) \\ & \cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \frac{\partial^2}{\partial \xi' \partial \zeta'} \frac{e^{-jkR}}{R} d\xi' d\zeta' d\zeta d\xi \end{aligned} \quad (C.48)$$

Two times integrations by parts, first with respect to ζ' and then with respect to ξ' gives

$$\begin{aligned} Y_{rq}^{ext} & = \frac{-Z_0}{2\pi j \omega \mu_0} \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \delta(\xi) \\ & \cdot \frac{-r\pi}{2L_x} \int_{-L_x}^{L_x} \cos \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \frac{e^{-jkR}}{R} \Big|_{\zeta'=-w/2}^{w/2} d\xi' d\zeta d\xi \end{aligned} \quad (C.49)$$

Evaluating the ξ integral gives the following computational efficient double integral

$$\begin{aligned} Y_{rq}^{ext} & = \frac{Z_0 r}{4j \omega \mu_0 L_x} \int_{-L_z}^{L_z} \int_{-L_x}^{L_x} \sin \left[\frac{q\pi}{2L_z}(L_z + \zeta) \right] \\ & \cdot \cos \left[\frac{r\pi}{2L_x}(L_x + \xi') \right] \left[\frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\xi' d\zeta \end{aligned} \quad (C.50)$$

where

$$R_1 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 - \zeta)^2} \quad (\text{C.51a})$$

$$R_2 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 + \zeta)^2} \quad (\text{C.51b})$$

The interior field contribution to the other cross term - matrix element number (p, s) is found by a weighting the magnetic field $H_x^{int}(E_x)$ with the function w_s

$$\begin{aligned} Y_{ps}^{int} &= \langle H_x^{int}(E_x), w_s \rangle \frac{Z_0}{E_p} \quad (\text{C.52}) \\ &= \frac{-2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \\ &\quad \cdot \sin \left[\frac{m\pi}{a}(x_{00} + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \\ &\quad \cdot \frac{\partial^2}{\partial \xi' \partial \zeta'} \sin \left[\frac{m\pi}{a}(x_0 + \xi') \right] e^{-\gamma_{mn}|\zeta - \zeta'|} d\zeta' d\xi' d\xi d\zeta \end{aligned}$$

Differentiation with respect to ξ' and ζ' and interchanging summation and integration gives

$$\begin{aligned} Y_{ps}^{int} &= \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{m\pi}{a} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \quad (\text{C.53}) \\ &\quad \cdot \sin \left[\frac{m\pi}{a}(x_{00} + \xi) \right] \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \\ &\quad \cdot \cos \left[\frac{m\pi}{a}(x_0 + \xi') \right] \left[\pm e^{-\gamma_{mn}|\zeta - \zeta'|} \right] d\zeta' d\xi' d\xi d\zeta \end{aligned}$$

where the upper sign is valid when $\zeta > \zeta'$ and lower sign when $\zeta < \zeta'$. Using eq. C.20, the integral with respect to ξ evaluates to

$$Y_{ps}^{int} = \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{\frac{m\pi}{a} \frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2} \quad (\text{C.54})$$

$Y_{rs}^{int} = 0$ when p is odd.

The exterior contribution to the matrix element number (p, s) is found by weighting the magnetic field $H_x^{ext}(E_x)$ with the function w_s

$$Y_{ps}^{ext} = \langle H_x^{ext}(E_x), w_s \rangle \frac{Z_0}{E_p} \quad (C.59)$$

$$\begin{aligned} &= \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \\ &\quad \cdot \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \frac{\partial^2}{\partial \xi' \partial \zeta'} \frac{e^{-jkR}}{R} d\zeta' d\xi' d\xi d\zeta \end{aligned} \quad (C.60)$$

Two times integrations by parts, first with respect to ξ' and then with respect to ζ' gives

$$\begin{aligned} Y_{ps}^{ext} &= \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \delta(\zeta) \\ &\quad \cdot \frac{-p\pi}{2L_z} \int_{-L_z}^{L_z} \cos \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \frac{e^{-jkR}}{R} \Big|_{\xi'=-w/2}^{w/2} d\zeta' d\xi d\zeta \end{aligned} \quad (C.61)$$

Evaluating the ζ integral gives a computational efficient double integral

$$\begin{aligned} Y_{ps}^{ext} &= \frac{-Z_0 p}{4j\omega\mu_0 L_z} \int_{-L_x}^{L_x} \int_{-L_z}^{L_z} \sin \left[\frac{s\pi}{2L_x}(L_x + \xi) \right] \\ &\quad \cdot \cos \left[\frac{p\pi}{2L_z}(L_z + \zeta') \right] \left[\frac{e^{-jkR_1}}{R_1} - \frac{e^{-jkR_2}}{R_2} \right] d\zeta' d\xi \end{aligned} \quad (C.62)$$

where

$$R_1 = \sqrt{(L_x - \xi)^2 + \zeta'^2} \quad (C.63a)$$

$$R_2 = \sqrt{(L_x - w - \xi)^2 + \zeta'^2} \quad (C.63b)$$

C.4. Excitation Terms

From the incident TE_{10} mode we have the magnetic fields eq. 1.28 and 1.31. Applying eq. 2.11a and 2.11b gives directly

$$\begin{aligned} h_q^{inc} &= \langle H_z^{inc}, w_q \rangle \\ &= j \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin \left[\frac{q\pi}{2L_z} (\zeta + L_z) \right] \delta(\xi) \cos \left[\frac{\pi}{a} (x_0 + \xi) \right] e^{-j\beta_{10}\zeta} d\zeta d\xi \end{aligned} \quad (C.64)$$

Using the more general integral formula

$$\int_{-L}^L \sin \left[\frac{p\pi}{2L} (x + L) \right] e^{-jkx} dx = \frac{\frac{p\pi}{L}}{\left(\frac{p\pi}{2L}\right)^2 - k^2} \cdot \begin{cases} \cos(kL) & p \text{ odd} \\ j \sin(kL) & p \text{ even} \end{cases} \quad (C.65)$$

eq. C.65 becomes

$$h_q^{inc} = \frac{\frac{q\pi}{L_z}}{\left(\frac{q\pi}{2L_z}\right)^2 - \beta_{10}^2} \cos \frac{\pi x_0}{a} \cdot \begin{cases} j \cos(\beta_{10}L_z) & q \text{ odd} \\ -\sin(\beta_{10}L_z) & q \text{ even} \end{cases} \quad (C.66)$$

Similarly we get the following driving term for the x -component

$$\begin{aligned} h_s^{inc} &= \langle H_x^{inc}, w_s \rangle \\ &= \frac{-\beta_{10}}{\pi/a} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin \left[\frac{s\pi}{2L_x} (L_x + \xi) \right] \delta(\zeta) \sin \left[\frac{\pi}{a} (x_{00} + \xi) \right] e^{-j\beta_{10}\zeta} d\xi d\zeta \end{aligned} \quad (C.67)$$

Evaluating this integral and applying various trigometric identities it reduces to the following final equation

$$h_s^{inc} = \frac{-\beta_{10}}{\pi/a} \frac{\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2} \cdot \begin{cases} \cos \frac{\pi L_x}{a} \sin \frac{\pi x_{00}}{a} & s \text{ odd} \\ -\cos \frac{\pi x_{00}}{a} \sin \frac{\pi L_x}{a} & s \text{ even} \end{cases} \quad (C.68)$$

Appendix D

Source Code

A list of the implemented FORTRAN90 program including all subroutines and functions.

```
1      !#####
2      !
3      !   Name:
4      !         main
5      !
6      !   Revised:
7      !         February 29, 2000
8      !
9      !   Purpose:
10     !         Reads input file slot.in and initiates simulation.
11     !
12     !   Usage:
13     !         main()
14     !
15     !   Modules:
16     !         gen_var
17     !
18     !   Subroutines:
19     !         read_input,init_propagation_var,nw_par_fsweep,nw_par_lsweep,
20     !         nw_par_dlsweep,nw_par_a1sweep,rad_field,rad_field_ppwg
21     !
22     program main
23     !modules
24     use gen_var, only : freq,freq_0,sw_type
25     !implicit statement
26     implicit none
27
28     call read_input()
29     freq=freq_0
30     call init_propagation_var()
31
32     if(sw_type.eq.1) then
33     call nw_par_fsweep()
34     elseif(sw_type.eq.2) then
35     call nw_par_lsweep()
36     elseif(sw_type.eq.3) then
```

```

37         call nw_par_dlsweep()
38     elseif(sw_type.eq.4) then
39         call nw_par_aisweep()
40     elseif(sw_type.eq.5) then
41         call rad_field(1)
42         call rad_field(2)
43     else
44         write(*,*) 'Wrong type of sweep selected!'
45     endif
46 end
47
48 !#####
49 !
50 !   Name:
51 !       nw_par_aisweep
52 !
53 !   Revised:
54 !       February 29, 2000
55 !
56 !   Purpose:
57 !       This subroutine performs a sweep over the PP waveguide a'
58 !       dimension computing the equivalent network parameters for a
59 !       longitudinal, transverse or T-slot.
60 !
61 !   Usage:
62 !       call nw_par_aisweep()
63 !
64 !   Arguments:
65 !
66 !   Modules:
67 !       slot_var,mom_var,gen_var,ppwg_var
68 !
69 !   Subroutines:
70 !       read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
71 !
72 !   IMSL subroutines:
73 !       dlsacg
74 !
75 !   Remarks:
76 !       The output is printed to standard output.
77 !
78 subroutine nw_par_aisweep()
79     !modules
80     use slot_var, only : w,Lx,Lx0,Lz,Lz0,x0,x00,wp
81     use mom_var, only : Nz,Nx,Y,h_inc,E
82     use gen_var, only : freq_0,freq
83     use ppwg_var, only : a1
84     !implicit statement
85     implicit none
86     !list of local variables
87     integer      :: i
88
89     Lx=Lx0
90     Lz=Lz0
91     x00=x0+w/2.0-Lx
92     do i= 0,10
93         a1=16.0001E-3_wp+i*1.0E-3_wp
94         call Y_fill()
95         call hinc_fill()

```

```

96         call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
97         call write_nw_par()
98     enddo
99 return
100 end
101
102 !#####
103 !
104 ! Name:
105 !     nw_par_fsweep
106 !
107 ! Revised:
108 !     February 29, 2000
109 !
110 ! Purpose:
111 !     This subroutine performs a frequency sweep computing the
112 !     equivalent network parameters for a longitudinal, transverse or
113 !     T-slot.
114 !
115 ! Usage:
116 !     call nw_par_fsweep()
117 !
118 ! Arguments:
119 !
120 ! Modules:
121 !     slot_var,mom_var,gen_var,wg_var
122 !
123 ! Subroutines:
124 !     read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
125 !
126 ! IMSL subroutines:
127 !     dlsacg
128 !
129 ! Remarks:
130 !     The output is printed to standard output.
131 !
132 subroutine nw_par_fsweep()
133     !modules
134     use slot_var, only : w,Lx,Lx0,Lz,Lz0,x0,x00,wp
135     use mom_var, only : Nz,Nx,Y,h_inc,E
136     use gen_var, only : freq_0,freq,nfreq,dfreq
137     use wg_var, only : a
138     !implicit statement
139     implicit none
140     !list of local variables
141     integer      :: i
142
143     !round end correction
144     !Lx=Lx0-0.25*0.215*w      !for t-slot
145     !Lx=Lx0-0.1075_wp*w      !for transverse slot
146     !Lz=Lz0-0.1075_wp*w      !for longitudinal and t-slot
147     !no round end correction
148     Lx=Lx0
149     Lz=Lz0
150
151     x00=x0+w/2.0-Lx
152     do i= 0,nfreq
153         freq=freq_0+(i-nfreq/2.0_wp)*dfreq
154         call init_propagation_var()

```

```

155         call Y_fill()
156         call hinc_fill()
157         call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
158         call write_nw_par()
159     enddo
160 return
161 end
162
163 !#####
164 !
165 ! Name:
166 !     nw_par_dlsweep
167 !
168 ! Revised:
169 !     February 29, 2000
170 !
171 ! Purpose:
172 !     This subroutine performs a sweep over both T-slot dimensions
173 !     Lx and Lz computing the equivalent network parameters.
174 !
175 ! Usage:
176 !     call nw_par_dlsweep()
177 !
178 ! Arguments:
179 !
180 ! Modules:
181 !     slot_var,mom_var,gen_var
182 !
183 ! Subroutines:
184 !     read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
185 !
186 ! IMSL subroutines:
187 !     dlsacg
188 !
189 ! Remarks:
190 !     The output is printed to standard output.
191 !
192 subroutine nw_par_dlsweep()
193     !modules
194     use slot_var, only : w,Lz,Lz0,Lx,Lx0,x0,x00,wp
195     use mom_var, only : Nz,Nx,Y,h_inc,E
196     use gen_var, only : freq_0,freq,lambda_0,nll,dll,ntl,dtl
197     !implicit statement
198     implicit none
199     !list of local variables
200     integer      :: i,j
201
202     write(*,1076) (i,i=1,Nz),(j,j=1,Nx)
203     do i= 3,3 !0,ntl
204         Lx=Lx0+(i-ntl/2.0_wp)*dtl
205         Lz0=0.48*lambda_0-2.0_wp*Lx+w/2.0_wp
206         x00=x0+w/2.0_wp-Lx
207         do j=3,3 !0,nll
208             Lz=Lz0+(3+j-nll/2)*dll*lambda_0
209             call Y_fill()
210             call hinc_fill()
211             call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
212             call write_nw_par()
213         enddo

```

```

214     enddo
215
216 1076 format(9x,'freq[Hz] ',12x,'x0[m] ',12x,'a1[m] ',&
217           12x,'Lx[m] ',12x,'Lz[m] ',13x,'z_re',13x,'z_im',&
218           13x,'R_re',13x,'R_im',13x,'T_re',13x,'T_im',&
219           24(16x,'Ex(',I2.2,') ',12x),13(16x,'Ez(',I2.2,') ',12x))
220 return
221 end
222
223 !#####
224 !
225 ! Name:
226 !     nw_par_lsweep
227 !
228 ! Revised:
229 !     February 29, 2000
230 !
231 ! Purpose:
232 !     This subroutine is used to investigate a longitudinal slot
233 !     radiating into either a half space or a PP waveguide. Performs a
234 !     sweep over both slot length and offset from centerline computing
235 !     the equivalent network parameters.
236 !
237 ! Usage:
238 !     call nw_par_lsweep()
239 !
240 ! Arguments:
241 !
242 ! Modules:
243 !     const,slot_var,mom_var,gen_var,ppwg_var,wg_var
244 !
245 ! Subroutines:
246 !     read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
247 !
248 ! IMSL subroutines:
249 !     dlsacg
250 !
251 ! Remarks:
252 !     The output is printed to standard output.
253 !
254 subroutine nw_par_lsweep()
255     !modules
256     use const, only : mu_0,pi,cj,Z0,wp
257     use slot_var, only : w,Lz,Lz0,Lx,Lx0,x0,x00
258     use mom_var, only : Nz,Nx,Y,h_inc,E
259     use gen_var, only : freq_0,freq,lambda_0,nll,dll,ntl,dtl
260     use wg_var, only : a
261     use ppwg_var, only : a1
262     !implicit statement
263     implicit none
264     !list of local variables
265     integer :: i,j
266
267     Lz0=lambda_0/4.0_wp
268     do i=0,8 !offset sweep
269         x0=a/2.0_wp+2.0E-3_wp+i*1.0E-3_wp
270         do j= 0,nll !length sweep
271             Lz=Lz0*(0.94_wp+dll*j/nll)
272             call Y_fill()

```

```
273         call hinc_fill()
274         call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
275         call write_y_norm()
276     enddo
277 enddo
278 return
279 end

1  !#####
2  !
3  ! Name:
4  !     const
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     module containing all general constants
11 !
12 ! Usage:
13 !     use const
14 !
15 module const
16     !list of parameters
17     !kind of double
18     integer,parameter      :: wp    = kind(1.0D0)
19     !pi
20     real(wp),  parameter  :: pi    = 3.14159265359_wp
21     !degrees2radians
22     real(wp),  parameter  :: dtr   = 1.74532925199E-2_wp
23     !radians2degrees
24     real(wp),  parameter  :: rtd   = 57.2957795132_wp
25     !inch2meter
26     real(wp),  parameter  :: itm   = 25.4000000E-3_wp
27     !meter2inch
28     real(wp),  parameter  :: mti   = 39.3700787_wp
29     !speed of light in vaccum
30     real(wp),  parameter  :: c     = 2.997925E8_wp
31     !free space permittivity
32     real(wp),  parameter  :: eps_0 = 8.8540000E-12_wp
33     !free space permeability
34     real(wp),  parameter  :: mu_0  = 1.25663706E-6_wp
35     !free space intrinsic impedance
36     real(wp),  parameter  :: Z0    = 376.734309_wp
37     !sqrt(-1)
38     complex(wp),parameter  :: cj   = (0.0_wp,1.0_wp)
39 end module const
40 !#####
41 !
42 ! Name:
43 !     mom_var
44 !
45 ! Revised:
46 !     February 29, 2000
47 !
48 ! Purpose:
49 !     module containing variables and parameters related to the MoM.
50 !
51 ! Usage:
52 !     use mom_var
```



```

53  !
54  !  Modules:
55  !      const
56  !
57  module mom_var
58      !modules
59      use      const, only : wp
60      !list of parameters
61      integer,parameter  :: Nx   = 13      !number of transverse exp. func
62      integer,parameter  :: Nz   = 24      !number of longitudinal exp func
63      integer,parameter  :: mmax  = 150     !maximum number of terms in
64      !interior green's function.
65      integer,parameter  :: nmax  = 75     !nmax=mmax/2
66      !list of variables
67      integer            :: r,s,p,q      !mode index for exp./weight func.
68      integer            :: y1_mode,y2_mode !mode index for aux. func. y1/y2
69      real(wp)           :: kp,kq,kr,ks   !wavenumber for exp./weight func.
70      real(wp)           :: kp_sq,kq_sq   !do squared
71      real(wp)           :: kr_sq,ks_sq   !do squared
72      complex(wp),dimension(1:Nx+Nz,1:Nx+Nz) :: Y      !moment matrix
73      complex(wp),dimension(1:Nx+Nz)         :: E      !Exp. coeff.
74      complex(wp),dimension(1:Nx+Nz)         :: h_inc   !excitation coeff.
75  end module mom_var
76  !#####
77  !
78  !  Name:
79  !      slot_var
80  !
81  !  Revised:
82  !      February 29, 2000
83  !
84  !  Purpose:
85  !      Module containing variables describing a single slot such as
86  !      lengths, offsets, width etc.
87  !
88  !  Usage:
89  !      use slot_var
90  !
91  !  Modules:
92  !      const
93  !
94  module slot_var
95      !modules
96      use      const, only : wp
97      !variables
98      real(wp)  :: w          !slot width
99      real(wp)  :: Lx         !length of transverse slot
100     real(wp)  :: Lz         !length of longitudinal slot
101     real(wp)  :: Lx0        !initial length of transverse slot
102     real(wp)  :: Lz0        !initial length of longitudinal slot
103     real(wp)  :: Ls         !aux. variable for any slot length
104     real(wp)  :: x0         !offset of longitudinal slot
105     real(wp)  :: x00        !offset of transverse slot
106     real(wp)  :: x0_pp      !offset of longitudinal slot in PP wg.
107     real(wp)  :: x00_pp     !offset of transverse slot in PP wg.
108     real(wp)  :: xi,eta,zeta !local coordinates
109  end module slot_var
110  !#####
111  !

```

```
112 ! Name:
113 !     gen_var
114 !
115 ! Revised:
116 !     February 29, 2000
117 !
118 ! Purpose:
119 !     Module containing general variables such as frequency, propagation
120 !     constants etc.
121 !
122 ! Usage:
123 !     use gen_var
124 !
125 ! Modules:
126 !     const
127 !
128 module gen_var
129     !modules
130     use const, only : wp
131     !list of variables
132     integer    :: ext_type    !flag indicating halv-space/PP-wg
133     integer    :: sw_type    !flag indicating kind of sweep to be performed
134     real(wp)   :: freq_0     !design frequency
135     real(wp)   :: freq       !frequency
136     integer    :: nfreq      !number of steps in frequency sweep
137     real(wp)   :: dfreq      !size of frequency step in sweep
138     integer    :: nll        !number of steps in length sweep for
139     !longitudinal slot
140     real(wp)   :: dll        !size of lenght step
141     integer    :: ntl        !number of steps in length sweep for
142     !transverse slot
143     real(wp)   :: dtl        !size of lenght step
144     real(wp)   :: k0,k0_sq   !free space wavenumber and do squared
145     real(wp)   :: lambda_0   !free space wavelength
146     real(wp)   :: omega      !angular frequency
147
148 end module gen_var
149 !#####
150 !
151 ! Name:
152 !     wg_var
153 !
154 ! Revised:
155 !     February 29, 2000
156 !
157 ! Purpose:
158 !     Module containing parameters and variables related to the main
159 !     waveguide.
160 !
161 ! Usage:
162 !     use wg_var
163 !
164 ! Modules:
165 !     const,mom_var
166 !
167 module wg_var
168     !modules
169     use const, only : wp
170     use mom_var, only : mmax,nmax
```

```

171     !list of variables
172     !wg a dimension
173     real(wp)    :: a
174     !wg b dimension
175     real(wp)    :: b
176     !wall thickness
177     real(wp)    :: t
178     !guided wavelenght
179     real(wp)    :: lambda_g
180     !TE_10 propagation constant
181     real(wp)    :: beta_10
182     !propagation constants in waveguide
183     complex(wp),dimension(0:mmax,0:nmax) :: gamma
184
185 end module wg_var
186 !#####
187 !
188 !   Name:
189 !       ppwg_var
190 !
191 !   Revised:
192 !       February 29, 2000
193 !
194 !   Purpose:
195 !       Module containing parameters and variables related to the PP
196 !       waveguide.
197 !
198 !   Usage:
199 !       use ppwg_var
200 !
201 !   Modules:
202 !       const
203 !
204 module ppwg_var
205     !modules
206     use const, only : wp
207     !list of variables
208     !pp waveguide a' dimension
209     real(wp)    :: a1
210     !pp waveguide b' dimension
211     real(wp)    :: b1
212     !kx mode
213     integer     :: m
214     !kx propagation constant
215     real(wp)    :: kx,kx_sq
216     !ky propagation constant
217     complex(wp) :: ky,ky_sq
218     !kz propagation constant
219     complex(wp) :: kz,kz_sq
220 end module ppwg_var
221 !#####
222 !
223 !   Name:
224 !       quad_var
225 !
226 !   Revised:
227 !       February 29, 2000
228 !
229 !   Purpose:

```

```

230 !           This module contains some variables used in the integration of the
231 !           function 1/R*f(xi,zeta) over the rectangle: 2Lz*2Lx and in
232 !           evaluation of sommerfeld integrals by the method of weighted
233 !           averages. Also parameters specifying the accuracy etc.
234 !           are included.
235 !
236 ! Usage:
237 !       use quad_var
238 !
239 ! Modules:
240 !       const
241 !
242 module quad_var
243 !modules
244     use    const, only : wp
245     !list of parameters
246     !maximum func. evaluations in quadr.
247     integer, parameter :: maxfcn = 500000
248     !relative accuracy desired in quadr.
249     real(wp),parameter :: errrel = 0.0001_wp
250     !absolute accuracy desired in quadr.
251     real(wp),parameter :: errabs = 0.0_wp
252     !number of partial values to be calculated
253     integer, parameter :: mm = 6
254
255     !list of variables
256     integer      :: c_flag      !flag indicating real or imag part
257     real(wp)     :: xi_0,zeta_0 !coordinate of singular point
258     real(wp)     :: xi_1,zeta_1 !upper limits in subintegral (aux. var)
259     real(wp)     :: f_0         !value of f(xi,zeta) at singular point
260     real(wp)     :: df_dxi     !value of xi derivative at sing. point
261     real(wp)     :: df_dzeta   !value of zeta derivative at sing. point
262     real(wp)     :: R1         !min(abs(xi_1),abs(zeta_1)) (aux. var)
263     real(wp)     :: R2         !max(abs(xi_1),abs(zeta_1)) (aux. var)
264     real(wp)     :: R3         !sqrt(xi_1**2+zeta_1**2) (aux. var)
265     real(wp)     :: kw,kw1,kw2 !wavenumber in osc. part of sommerfeld int.
266     real(wp)     :: kz_max     !max kz in deformed contour integral
267     real(wp)     :: dkw        !wavenumber in osc. part of sommerfeld int.
268     real(wp)     :: dkz_max    !max kz in deformed contour integral
269 end module quad_var
270 !#####
271 !
272 ! Name:
273 !       radfield_var
274 !
275 ! Revised:
276 !       February 29, 2000
277 !
278 ! Purpose:
279 !       Module containing variables related to far zone fields.
280 !
281 ! Usage:
282 !       use radfield_var
283 !
284 ! Modules:
285 !       const
286 !
287 module radfield_var
288     !modules

```

```

289     use const, only : wp
290     !list of variables
291     !vector holding spherical comp. of the E-field
292     complex(wp),dimension(1:3) :: E_sph
293     !vector holding spherical comp. of the H-field
294     complex(wp),dimension(1:3) :: H_sph
295     !vector holding spherical comp. of the elec. vector potential
296     complex(wp),dimension(1:3) :: F_sph
297     !vector holding cartesian comp. of the elec. vector potential
298     complex(wp),dimension(1:3) :: F_xyz
299     !vector holding spherical comp. of the magn. vector potential
300     complex(wp),dimension(1:3) :: A_sph
301     !vector holding cartesian comp. of the magn. vector potential
302     complex(wp),dimension(1:3) :: A_xyz
303     !coordinate vectors
304     real(wp),dimension(1:3) :: sph,xyz
305 end module radfield_var

1  !#####
2  !
3  ! Name:
4  !     read_input
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     This subroutine reads information from the slot.in file into
11 !     a namelist
12 !
13 ! Usage:
14 !     call read_input()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 !     ppwg_var,slot_var,mom_var,gen_var,wg_var
20 !
21 ! Subroutines:
22 !
23 ! Remarks:
24 !     The input files should not contain entries not in namelist.
25 !
26 subroutine read_input()
27     !modules
28     use gen_var, only : freq_0,nfreq,dfreq,nll,dll,ntl,dtl,ext_type,sw_type
29     use wg_var, only : a,b,t
30     use ppwg_var, only : a1,b1
31     use slot_var, only : w,Lz0,Lx0,x0
32     !implicit statement
33     implicit none
34     !declaration of namelist
35     namelist /var_list/ freq_0,nfreq,dfreq,Lz0,nll,dll,Lx0,ntl,dtl,&
36         w,x0,a,b,t,ext_type,a1,b1,sw_type
37     !list of local variables
38     integer :: ioerr
39
40     OPEN (7, FILE='slot.in', STATUS='OLD', ACTION='READ', IOSTAT=ioerr)
41     READ (UNIT=7,NML=var_list)
42     CLOSE (7)

```

```

43
44 return
45 end
46
47 !#####
48 !
49 ! Name:
50 !     read_efield
51 !
52 ! Revised:
53 !     February 29, 2000
54 !
55 ! Purpose:
56 !     This subroutine reads expansion coefficients from a previous
57 !     simulation stored in the efield.dat file
58 !
59 ! Usage:
60 !     call read_efield()
61 !
62 ! Arguments:
63 !
64 ! Modules:
65 !     const,ppwg_var,slot_var,mom_var,gen_var
66 !
67 ! Subroutines:
68 !
69 subroutine read_efield()
70 !modules
71 use const, only : cj,wp
72 use gen_var, only : freq
73 use slot_var, only : Lz,Lx,x0
74 use ppwg_var, only : a1
75 use mom_var, only : Nz,Nx,E
76 !implicit statement
77 implicit none
78 !list of local variables
79 integer :: ioerr,i,j
80 real(wp), dimension(1:2) :: z,R,T
81 real(wp), dimension(1:2*Nz) :: Ex_in
82 real(wp), dimension(1:2*Nx) :: Ez_in
83
84 OPEN (8, FILE='efield.dat', STATUS='OLD',ACTION='READ',IOSTAT=ioerr)
85 READ(8,1075) freq,x0,a1,Lx,Lz,z(1),z(2),R(1),R(2),T(1),T(2),Ex_in,Ez_in
86 CLOSE (8)
87
88 do i=1,Nz
89     E(i)=Ex_in(2*i-1)+cj*Ex_in(2*i)
90 enddo
91 do i=1,Nx
92     E(i+Nz)=Ez_in(2*i-1)+cj*Ez_in(2*i)
93 enddo
94
95 1075 format(2x,EN15.6,4(2x,EN15.6),6(2x,E15.6),48(2x,E15.6),26(2x,E15.6))
96 return
97 end
98
99 !#####
100 !
101 ! Name:

```

```

102 !           init_propagation_var
103 !
104 !   Revised:
105 !           February 29, 2000
106 !
107 !   Purpose:
108 !           This subroutine initialize all propagation variables on basis of
109 !           frequency, waveguide dimensions etc.
110 !
111 !   Usage:
112 !           call init_propagation_var()
113 !
114 !   Arguments:
115 !
116 !   Modules:
117 !           const,mom_var,gen_var,wg_var
118 !
119 !   Subroutines:
120 !
121 subroutine init_propagation_var()
122 !modules
123 use const, only : eps_0,mu_0,c,pi,cj,wp
124 use gen_var, only : k0,k0_sq,freq_0,freq,lambda_0,omega
125 use wg_var, only : a,b,gamma,beta_10,lambda_g
126 use mom_var, only : mmax,nmax
127 !implicit statement
128 implicit none
129 !declaration of intrinsic functions
130 intrinsic dsqrt,cdsqrt,dcmplx
131 !list of local variables
132 integer :: m,n
133
134 omega=2.0_wp*pi*freq
135 k0=omega*dsqrt(eps_0*mu_0)
136 k0_sq=k0**2
137 lambda_0=c/freq
138 beta_10=k0*dsqrt(1-(lambda_0/(2.0_wp*a))**2)
139 lambda_g=2.0_wp*pi/beta_10
140
141 do m=0,mmax
142     do n=0,nmax
143         gamma(m,n)=cdsqrt(dcplx((m*pi/a)**2+(n*pi/b)**2-k0_sq))
144     enddo
145 enddo
146
147 return
148 end
149
150 !#####
151 !
152 !   Name:
153 !           dsinc
154 !
155 !   Revised:
156 !           February 29, 2000
157 !
158 !   Purpose:
159 !           Calculate the sinc function sin(x)/x
160 !

```

```
161 ! Usage:
162 !       dsin(x)
163 !
164 ! Arguments:
165 !   x       - Independent variable (Input)
166 !
167 ! Modules:
168 !       const
169 !
170 function dsinc(x)
171 !modules
172 use const, only : wp
173 !implicit statement
174 implicit none
175 !declaration of intrinsic functions
176 intrinsic dsin,dabs
177 !list of calling arguments
178 real(wp)      :: dsinc
179 real(wp), intent(in)  :: x
180 !list of local parameters
181 real(wp),parameter :: epsilon = 1.0E-30_wp
182
183 if(dabs(x).gt.epsilon) then
184   dsinc=dsin(x)/x
185 else
186   dsinc=1.0_wp
187 endif
188
189 end function dsinc
190
191 !#####
192 !
193 ! Name:
194 !       xyz2sph
195 !
196 ! Revised:
197 !       February 29, 2000
198 !
199 ! Purpose:
200 !       Transform cartesian vector components to spherical components
201 !
202 ! Usage:
203 !       xyz2sph(F_xyz,sph,F_sph)
204 !
205 ! Arguments:
206 !   F_xyz - vector holding cartesian components (Input)
207 !
208 !   sph   - spherical coordinates (Input)
209 !
210 !   F_sph - vector holding spherical components (output)
211 !
212 ! Modules:
213 !       const
214 !
215 subroutine xyz2sph(F_xyz,sph,F_sph)
216 !modules
217 use const, only : wp
218 !implicit statement
219 implicit none
```



```

220     !declaration of intrinsic functions
221     intrinsic dsin,dcos
222     !list of local variables
223     real(wp), dimension(1:3),intent(in) :: sph
224     complex(wp), dimension(1:3),intent(out) :: F_sph
225     complex(wp), dimension(1:3),intent(in) :: F_xyz
226
227     F_sph(1)=F_xyz(1)*dsin(sph(2))*dcos(sph(3))+&
228             F_xyz(2)*dsin(sph(2))*dsin(sph(3))+&
229             F_xyz(3)*dcos(sph(2))
230     F_sph(2)=F_xyz(1)*dcos(sph(2))*dcos(sph(3))+&
231             F_xyz(2)*dcos(sph(2))*dsin(sph(3))-&
232             F_xyz(3)*dsin(sph(2))
233     F_sph(3)=-F_xyz(1)*dsin(sph(3))+&
234             F_xyz(2)*dcos(sph(3))
235
236     return
237 end

1  !#####
2  !
3  ! Name:
4  !     yzz_int_pq
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Computes the interior self contribution to the fields in a
11 !     longitudinal slot.
12 !
13 ! Usage:
14 !     yzz_int_pq()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 !     const,slot_var,mom_var,gen_var,wg_var
20 !
21 function yzz_int_pq()
22     !modules
23     use const, only : pi,cj,wp
24     use slot_var, only : w,Lz,x0
25     use mom_var, only : p,q,mmax,nmax
26     use gen_var, only : k0_sq,freq,lambda_0
27     use wg_var, only : a,b,gamma
28     !implicit statement
29     implicit none
30     !declaration of intrinsic functions
31     intrinsic mod,dcos,dsin,cdexp
32     !declaration of functions
33     real(wp)    :: dsinc
34     !list of calling arguments
35     complex(wp) :: yzz_int_pq
36     !list of local variables
37     integer     :: m,n
38     real(wp)    :: p_term,q_term
39     real(wp)    :: w_m,sign
40     complex(wp) :: sum1,sum2, add_term1,add_term2,gam
41

```

```

42     sum1=(0.0_wp,0.0_wp)
43     sum2=(0.0_wp,0.0_wp)
44     if(mod((p+q),2).eq.1) then
45         yzz_int_pq=(0.0_wp,0.0_wp)
46     else
47         p_term=(p*pi/(2.0_wp*Lz))**2
48         q_term=(q*pi/(2.0_wp*Lz))**2
49         if((mod(p,2).eq.1).and.(mod(q,2).eq.1)) then
50             sign=1.0_wp
51         else if ((mod(p,2).eq.0).and.(mod(q,2).eq.0)) then
52             sign=-1.0_wp
53         end if
54
55         do m=0,mmax
56             if (m.eq.0) then
57                 w_m=2.0_wp
58             else
59                 w_m=4.0_wp*dsinc((m*pi*w)/(2.0_wp*a))*&
60                     (dcos(m*pi*x0/a))**2
61             end if
62             do n=0,nmax
63                 add_term1=(0.0_wp,0.0_wp)
64                 add_term2=(0.0_wp,0.0_wp)
65                 gam=gamma(m,n)
66                 add_term1=(1.0_wp+sign*cdexp(-2.0_wp*gam*Lz))*(gam**2+k0_sq)/&
67                     (gam*(gam**2+p_term)*(gam**2+q_term))
68
69                 add_term2=w_m/(gam**2+p_term)
70                 !adjust addterms in case n==0
71                 if (n.eq.0) then
72                     add_term1=add_term1*0.5_wp
73                     add_term2=add_term2*0.5_wp
74                 end if
75
76                 sum1=sum1+w_m*add_term1
77                 if(p.eq.q) then
78                     sum2=sum2+add_term2
79                 end if
80             enddo
81         enddo
82         yzz_int_pq=cj*lambda_0*w/(8.0_wp*a*b*pi)*&
83             (sum1*p*pi/Lz*q*pi/Lz+sum2*4.0_wp*(k0_sq-p_term)*Lz)
84     end if
85
86 end function yzz_int_pq
87
88 !#####
89 !
90 !   Name:
91 !       yxx_int_rs
92 !
93 !   Revised:
94 !       February 29, 2000
95 !
96 !   Purpose:
97 !       Computes the interior self contribution to the fields in a
98 !       transverse slot.
99 !
100 !   Usage:

```

```

101 !           yxx_int_rs()
102 !
103 ! Arguments:
104 !
105 ! Modules:
106 !           const,slot_var,mom_var,gen_var,wg_var
107 !
108 function yxx_int_rs()
109     !modules
110     use const, only : pi,cj,Z0,mu_0,wp
111     use slot_var, only : Lx,x00,w
112     use mom_var, only : r,s,mmax,nmax
113     use gen_var, only : k0,k0_sq,lambda_0,omega
114     use wg_var, only : a,b,beta_10,gamma
115     !implicit statement
116     implicit none
117     !declaration of intrinsic functions
118     intrinsic dsin,dcos,cdexp,mod
119     !list of calling arguments
120     complex(wp) :: yxx_int_rs
121     !list of local variables
122     integer      :: m,n
123     real(wp)     :: w_m,w_half,m_term,r_term,s_term
124     complex(wp) :: sum, add_term, gam
125
126     if(mod((r+s),2).eq.1) then
127         yxx_int_rs=(0.0_wp,0.0_wp)
128     else
129         sum=(0.0_wp,0.0_wp)
130         w_half=0.5_wp*w
131         r_term=(r*pi/(2.0_wp*Lx))**2
132         s_term=(s*pi/(2.0_wp*Lx))**2
133         do m=0,mmax
134             m_term=m*pi/a
135             w_m=(m_term**2-k0_sq)/&
136                 ((r_term-m_term**2)*(s_term-m_term**2))
137             if(mod(r,2).eq.1) then !r odd
138                 w_m=w_m*(dcos(m_term*Lx)*dsin(m_term*x00))**2
139             else !r even
140                 w_m=w_m*(dcos(m_term*x00)*dsin(m_term*Lx))**2
141             end if
142             !adjust w_m in case m==0
143             if(m.eq.0) then
144                 w_m=w_m*0.5_wp
145             end if
146             do n=0,nmax
147                 gam=gamma(m,n)
148                 add_term=w_m*(1.0_wp-cdexp(-gam*w_half))/gam**2
149                 !adjust addterm in case n==0
150                 if (n.eq.0) then
151                     add_term=add_term*0.5_wp
152                 end if
153                 sum=sum+add_term
154             enddo !n loop
155         enddo !m loop
156
157         yxx_int_rs=-4.0_wp*Z0/(cj*omega*mu_0*a*b)*r*pi/Lx*s*pi/Lx*sum
158
159     end if

```

```

160 end function yxz_int_rs
161
162 !#####
163 !
164 ! Name:
165 !     yxz_int_rq
166 !
167 ! Revised:
168 !     February 29, 2000
169 !
170 ! Purpose:
171 !     Computes the interior cross coupling contribution to the fields
172 !     in a longitudinal slot due to currents in a transverse slot.
173 !
174 ! Usage:
175 !     yxz_int_rq()
176 !
177 ! Arguments:
178 !
179 ! Modules:
180 !     const,slot_var,mom_var,gen_var,wg_var
181 !
182 function yxz_int_rq()
183     !modules
184     use const, only : pi,cj,Z0,mu_0,wp
185     use slot_var, only : Lx,Lz,x0,x00,w
186     use mom_var, only : r,q,mmax,nmax
187     use gen_var, only : k0,omega
188     use wg_var, only : a,b,gamma
189     !implicit statement
190     implicit none
191     !declaration of intrinsic functions
192     intrinsic dsin,dcos,cdexp,mod
193     !list of calling arguments
194     complex(wp) :: yxz_int_rq
195     !list of local variables
196     integer      :: m,n
197     real(wp)     :: cos_term1,cos_term2,sin_term,w_m,q_term
198     complex(wp) :: w_mn,gam
199     complex(wp) :: sum, add_term1,add_term23
200
201     if(mod(q,2).eq.1) then !q odd
202         yxz_int_rq=(0.0_wp,0.0_wp)
203     else
204         sum=(0.0,0.0)
205         sin_term=2.0_wp*Lz*dsin(w*q*pi/(4.0_wp*Lz))
206         cos_term1=q*pi*dcos(w*q*pi/(4.0_wp*Lz))
207         cos_term2=dcos(q*pi/2.0_wp)
208         q_term=(q*pi/(2.0_wp*Lz))**2
209         do m=1,mmax
210             w_m=2.0_wp*m*pi/a*dcos(m*pi*x0/a)
211             w_m=w_m/((r*pi/(2.0_wp*Lx))**2-(m*pi/a)**2)
212             if(mod(r,2).eq.1) then !r odd
213                 w_m=w_m*dcos(m*pi*Lx/a)*dsin(m*pi*x00/a)
214             else !r even
215                 w_m=-w_m*dcos(m*pi*x00/a)*dsin(m*pi*Lx/a)
216             end if
217             !adjust w_m in case m==0
218             if (m.eq.0) then

```

```

219         w_m=w_m*0.5_wp
220     end if
221     do n=0,nmax
222         gam=gamma(m,n)
223         w_mn=2.0_wp/(q_term+gam**2)
224         !adjust w_mn in case n==0
225         if (n.eq.0) then
226             w_mn=w_mn*0.5_wp
227         end if
228         !case 1
229         add_term1=cos_term2*(&
230             sin_term*gam*(cdexp(-gam*w)+1.0_wp)+&
231             cos_term1*(cdexp(-gam*w)-1.0_wp))
232         !case 2+3
233         add_term23=cos_term2*(1.0_wp-cdexp(-gam*w))*&
234             (cos_term1+gam*sin_term)&
235             +q*pi*(cdexp(-gam*(Lz+w/2.0_wp))-&
236                 cdexp(-gam*(Lz-w/2.0_wp)))
237         sum=sum+(add_term1+add_term23)*w_m*w_mn/gam
238     enddo !n loop
239 enddo !m loop
240
241     yxz_int_rq=-Z0/(cj*omega*mu_0*a*b)*r*pi/Lx/Lz/2.0_wp*sum
242
243     end if
244 end function yxz_int_rq
245
246 !#####
247 !
248 ! Name:
249 !     yzx_int_ps
250 !
251 ! Revised:
252 !     February 29, 2000
253 !
254 ! Purpose:
255 !     Computes the interior cross coupling contribution to the fields
256 !     in a transverse slot due to currents in a longitudinal slot.
257 !
258 ! Usage:
259 !     yzx_int_ps()
260 !
261 ! Arguments:
262 !
263 ! Modules:
264 !     const,slot_var,mom_var,gen_var,wg_var
265 !
266 function yzx_int_ps()
267     !modules
268     use const, only : pi,cj,Z0,mu_0,wp
269     use slot_var, only : Lx,Lz,x0,x00,w
270     use mom_var, only : p,s,mmax,nmax
271     use gen_var, only : k0,omega
272     use wg_var, only : a,b,gamma
273     !implicit statement
274     implicit none
275     !declaration of intrinsic functions
276     intrinsic dsin,dcos,mod,cdexp
277     !list of calling arguments

```

```

278     complex(wp) :: yzx_int_ps
279     !list of local variables
280     integer      :: m,n
281     real(wp)     :: p_term,s_term,cos_term,w_m
282     complex(wp) :: sum,w_mn
283
284     if(mod(p,2).eq.1) then !p odd
285         yzx_int_ps=(0.0_wp,0.0_wp)
286     else
287         p_term=(p*pi/(2.0_wp*Lz))**2
288         s_term=(s*pi/(2.0_wp*Lx))**2
289         cos_term=dcos(p*pi/2.0_wp)
290         sum=(0.0,0.0)
291         do m=0,mmax
292             w_m=2.0_wp*dsin(m*pi*w/(2.0_wp*a))*dcos(m*pi*x0/a)
293             w_m=w_m/(s_term-(m*pi/a)**2)
294             if(mod(s,2).eq.1) then !s odd
295                 w_m=w_m*dcos(m*pi*Lx/a)*dsin(m*pi*x00/a)
296             else !s even
297                 w_m=-w_m*dcos(m*pi*x00/a)*dsin(m*pi*Lx/a)
298             end if
299             !adjust w_m in case m==0
300             if (m.eq.0) then
301                 w_m=w_m*0.5_wp
302             end if
303             do n=0,nmax
304                 w_mn=2.0_wp/(p_term+gamma(m,n)**2)
305                 !adjust w_mn in case n==0
306                 if (n.eq.0) then
307                     w_mn=w_mn*0.5_wp
308                 end if
309                 sum=sum+(cdexp(-gamma(m,n)*Lz)-cos_term)*w_m*w_mn
310             enddo !n loop
311         enddo !m loop
312
313         yzx_int_ps=-Z0/(c*j*omega*mu_0*a*b)*p*pi/Lz*s*pi/Lx*sum
314
315     end if
316 end function yzx_int_ps

```

```

1  !#####
2  !
3  ! Name:
4  !     yzx_ext_ps
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Computes the exterior cross coupling contribution to the fields
11 !     in a transverse slot due to currents in a longitudinal slot. Used
12 !     in case the slot radiates into half space.
13 !
14 ! Usage:
15 !     yzx_ext_ps()
16 !
17 ! Arguments:
18 !
19 ! Modules:
20 !     const,quad_var,slot_var,mom_var,gen_var

```

```

21  !
22  !  Subroutines:
23  !      yzx_ext_re_ps_p, yzx_ext_im_ps_p, inv_r_int
24  !
25  !  IMSL subroutines:
26  !      dq3nd
27  !
28  function yzx_ext_ps()
29  !modules
30  use const, only : pi,cj,Z0,mu_0,wp
31  use slot_var, only : Lx,Lz,w
32  use mom_var, only : p,s,ks,kp
33  use gen_var, only : k0,omega
34  use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,maxfcn,errrel,errabs
35  !implicit statement
36  implicit none
37  !declaration of intrinsic functions
38  intrinsic dsin,dcos,sum,mod
39  !declaration of functions
40  real(wp), external      :: yzx_ext_re_ps_p, yzx_ext_im_ps_p
41  real(wp)                :: inv_r_int
42  !list of calling arguments
43  complex(wp)            :: yzx_ext_ps
44  !list of local variables
45  real(wp), dimension(1:2) :: upper
46  real(wp), dimension(1:2) :: lower
47  real(wp), dimension(1:2) :: re_val_r1
48  real(wp), dimension(1:2) :: re_val_r2
49  real(wp)                :: errest,cos_term
50  real(wp)                :: im_val
51  integer                  :: ier
52
53  if(mod(p,2).eq.1) then !p odd
54  yzx_ext_ps=(0.0_wp,0.0_wp)
55  else
56  upper=(/Lx,Lz/)
57  lower=(/-Lx,-Lz/)
58  zeta_0=0.0_wp
59  cos_term=dcos(p*pi/2.0_wp)
60  kp=p*pi/(2.0_wp*Lz)
61  ks=s*pi/(2.0_wp*Lx)
62
63  !real part of 1/R1 term
64  xi_0=Lx
65  f_0=0.0_wp
66  df_dxi=(-1.0_wp)**s*ks*cos_term
67  df_dzeta=0.0_wp
68  call dq3nd (yzx_ext_re_ps_p,2,lower,upper,&
69  maxfcn,errabs,errrel,ier,re_val_r1(1),errest)
70  re_val_r1(2)=inv_r_int(-Lx,Lx,-Lz,Lz)
71
72  !real part of 1/R2 term
73  xi_0=Lx-w
74  f_0=(-1.0_wp)**s*dsin(ks*w)*cos_term
75  df_dxi=(-1.0_wp)**s*ks*dcos(ks*w)*cos_term
76  df_dzeta=0.0_wp
77  call dq3nd (yzx_ext_re_ps_p,2,lower,upper,&
78  maxfcn,errabs,errrel,ier,re_val_r2(1),errest)
79  re_val_r2(2)=inv_r_int(-Lx,Lx,-Lz,Lz)

```

```

80
81      !imaginary part
82      call dq3nd (yzx_ext_im_ps_p,2,lower,upper,&
83                maxfcn,errabs,errrel,ier,im_val,errest)
84
85      yzx_ext_ps=-Z0*p/(4.0_wp*cj*omega*mu_0*Lz)*&
86                (sum(re_val_r1)-sum(re_val_r2)+k0*cj*im_val)
87      endif
88
89  end function yzx_ext_ps
90
91  !#####
92  !
93  ! Name:
94  !     yzx_ext_re_ps_p
95  !
96  ! Revised:
97  !     February 29, 2000
98  !
99  ! Purpose:
100 !     Auxiliary function used by the function yzx_ext_ps. Evaluates
101 !     the real part of kernel used to compute the exterior cross
102 !     coupling contribution to the fields in a transverse slot due to
103 !     currents in a longitudinal slot.
104 !
105 ! Usage:
106 !     yzx_ext_re_ps_p(dim,x)
107 !
108 ! Arguments:
109 !     dim      - Obsolete!
110 !     x        - Independent variable (Input)
111 !
112 ! Modules:
113 !     const,quad_var,slot_var,mom_var,gen_var
114 !
115 function yzx_ext_re_ps_p(dim,x)
116   !modules
117   use const, only : pi,wp
118   use slot_var, only : Lx,Lz
119   use mom_var, only : s,p,ks,kp
120   use gen_var, only : k0
121   use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0
122   !implicit statement
123   implicit none
124   !declaration of intrinsic functions
125   intrinsic dsqrt,dsin,dcos
126   !list of calling arguments
127   real(wp)                :: yzx_ext_re_ps_p
128   real(wp), dimension(1:2), intent(in) :: x
129   integer , intent(in)    :: dim
130   !list of local variables
131   real(wp)                :: RR
132
133   !xi=x(1)    zeta=x(2)
134   RR=dsqrt((x(1)-xi_0)**2+(x(2)-zeta_0)**2)
135
136   if(RR.lt.1E-8_wp) then
137     yzx_ext_re_ps_p=0.0_wp
138   else

```



```

139         yzx_ext_re_ps_p=(dsin(ks*(Lx+x(1)))&
140             *dcos(kp*(Lz+x(2)))*dcos(k0*RR)&
141             -f_0-(x(1)-xi_0)*df_dxi-(x(2)-zeta_0)*df_dzeta)/RR
142     endif
143
144 end function yzx_ext_re_ps_p
145
146 !#####
147 !
148 ! Name:
149 !     yzx_ext_im_ps_p
150 !
151 ! Revised:
152 !     February 29, 2000
153 !
154 ! Purpose:
155 !     Auxiliary function used by the function yzx_ext_ps. Evaluates
156 !     the imaginary part of kernel used to compute the exterior cross
157 !     coupling contribution to the fields in a transverse slot due to
158 !     currents in a longitudinal slot.
159 !
160 ! Usage:
161 !     yzx_ext_im_ps_p(dim,x)
162 !
163 ! Arguments:
164 !     dim - Obsolete!
165 !     x   - Independent variable (Input)
166 !
167 ! Modules:
168 !     const,quad_var,slot_var,mom_var,gen_var
169 !
170 ! Subroutines:
171 !     dsinc
172 !
173 function yzx_ext_im_ps_p(dim,x)
174     !modules
175     use const, only : pi,wp
176     use slot_var, only : Lx,Lz,x0,x00,w
177     use mom_var, only : ks,kp
178     use gen_var, only : k0
179     !implicit statement
180     implicit none
181     !declaration of intrinsic functions
182     intrinsic dsqrt,dsin,dcos
183     !declaration of functions
184     real(wp)          :: dsinc
185     !list of calling arguments
186     real(wp)          :: yzx_ext_im_ps_p
187     real(wp), dimension(1:2),intent(in):: x
188     integer,intent(in) :: dim
189     !list of local variables
190     real(wp)          :: R1,R2,xii
191
192     !xi=x(1)    zeta=x(2)
193     xii=(x0-x00)-x(1)
194
195     R1=dsqrt((xii+w/2.0_wp)**2+x(2)**2)
196     R2=dsqrt((xii-w/2.0_wp)**2+x(2)**2)
197

```

```

198     yzx_ext_im_ps_p=dsin(ks*(Lx+x(1)))*&
199         dcos(kp*(Lz+x(2)))*&
200         (dsinc(k0*R2)-dsinc(k0*R1))
201
202 end function yzx_ext_im_ps_p
203
204 !#####
205 !
206 ! Name:
207 !     yxz_ext_rq
208 !
209 ! Revised:
210 !     February 29, 2000
211 !
212 ! Purpose:
213 !     Computes the exterior cross coupling contribution to the fields
214 !     in a longitudinal slot due to currents in a transverse slot. Used
215 !     in case the slot radiates into half space.
216 !
217 ! Usage:
218 !     yxz_ext_rq()
219 !
220 ! Arguments:
221 !
222 ! Modules:
223 !     const,quad_var,slot_var,mom_var,gen_var
224 !
225 ! Subroutines:
226 !     yxz_ext_re_rq_p,yxz_ext_im_rq_p,inv_r_int
227 !
228 ! IMSL subroutines:
229 !     dq3nd
230 !
231 function yxz_ext_rq()
232 !modules
233     use const, only : pi,cj,Z0,mu_0,wp
234     use slot_var, only : Lx,Lz,w
235     use mom_var, only : q,r,kq,kr
236     use gen_var, only : k0,omega
237     use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,maxfcn,errrel,errabs
238     !implicit statement
239     implicit none
240     !declaration of intrinsic functions
241     intrinsic dsin,dcos,sum,mod
242     !declaration of functions
243     real(wp), external      :: yxz_ext_re_rq_p,yxz_ext_im_rq_p
244     real(wp)                :: inv_r_int
245     !list of calling arguments
246     complex(wp)             :: yxz_ext_rq
247     !list of local variables
248     real(wp), dimension(1:2) :: upper
249     real(wp), dimension(1:2) :: lower
250     real(wp), dimension(1:2) :: re_val_r1
251     real(wp), dimension(1:2) :: re_val_r2
252     real(wp), dimension(1:3) :: arg
253     real(wp)                 :: errest
254     real(wp)                 :: im_val
255     integer                  :: ier
256

```

```

257     if(mod(q,2).eq.1) then !q odd
258         yxz_ext_rq=(0.0_wp,0.0_wp)
259     else
260         upper=(/Lx,Lz/)
261         lower=(/-Lx,-Lz/)
262         kq=q*pi/(2.0_wp*Lz)
263         kr=r*pi/(2.0_wp*Lx)
264         arg=(/kq*(Lz+w/2.0_wp),kr*(2.0_wp*Lx-w/2.0_wp),kq*(Lz-w/2.0_wp)/)
265         xi_0=Lx-w/2.0_wp
266
267         !real part of 1/R1 term
268         zeta_0=w/2.0_wp
269         f_0=dsin(arg(1))*dcos(arg(2))
270         df_dxi=-kr**dsin(arg(1))*dsin(arg(2))
271         df_dzeta=kq*dcos(arg(1))*dcos(arg(2))
272         call dq3nd(yxz_ext_re_rq_p,2,lower,upper,&
273             maxfcn,errabs,errrel,ier,re_val_r1(1),errest)
274         re_val_r1(2)=inv_r_int(-Lx,Lx,-Lz,Lz)
275
276         !real part of 1/R2 term
277         zeta_0=-w/2.0_wp
278         f_0=dsin(arg(3))*dcos(arg(2))
279         df_dxi=-kr*dsin(arg(3))*dsin(arg(2))
280         df_dzeta=kq*dcos(arg(2))*dcos(arg(3))
281         call dq3nd(yxz_ext_re_rq_p,2,lower,upper,&
282             maxfcn,errabs,errrel,ier,re_val_r2(1),errest)
283         re_val_r2(2)=inv_r_int(-Lx,Lx,-Lz,Lz)
284
285         !imaginary part
286         call dq3nd (yzx_ext_im_rq_p,2,lower,upper,&
287             maxfcn,errabs,errrel,ier,im_val,errest)
288
289         yxz_ext_rq=Z0*r/(4.0_wp*cj*omega*mu_0*Lx)*&
290             (sum(re_val_r1)-sum(re_val_r2)+cj*k0*im_val)
291
292     end if
293
294 end function yxz_ext_rq
295
296 !#####
297 !
298 ! Name:
299 !     yxz_ext_re_rq_p
300 !
301 ! Revised:
302 !     February 29, 2000
303 !
304 ! Purpose:
305 !     Auxiliary function used by the function yxz_ext_rq. Evaluates
306 !     the real part of kernel used to compute the exterior cross
307 !     coupling contribution to the fields in a longitudinal slot due to
308 !     currents in a transverse slot.
309 !
310 ! Usage:
311 !     yxz_ext_re_rq_p(dim,x)
312 !
313 ! Arguments:
314 !     dim - Obsolete!
315 !     x   - Independent variable (Input)

```

```

316      !
317      ! Modules:
318      !           const,quad_var,slot_var,mom_var,gen_var
319      !
320      function yxz_ext_re_rq_p(dim,x)
321      !modules
322      use const, only : pi,wp
323      use slot_var, only : Lx,Lz
324      use mom_var, only : kr,kq
325      use gen_var, only : k0
326      use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0
327      !implicit statement
328      implicit none
329      !declaration of intrinsic functions
330      intrinsic dsqrt,dsin,dcos
331      !list of calling arguments
332      real(wp)                :: yxz_ext_re_rq_p
333      real(wp), dimension(1:2),intent(in) :: x
334      integer,intent(in)      :: dim
335      !list of local variables
336      real(wp)                :: RR
337
338      !xi=x(1)    zeta=x(2)
339      RR=sqrt((x(1)-xi_0)**2+(x(2)-zeta_0)**2)
340
341      if(RR.lt.1E-8_wp) then
342          yxz_ext_re_rq_p=0.0_wp
343      else
344          yxz_ext_re_rq_p=(dsin(kq*(Lz+x(2)))&
345              *dcos(kr*(Lx+x(1)))&dcos(k0*RR)&
346              -f_0-(x(1)-xi_0)*df_dxi-(x(2)-zeta_0)*df_dzeta)/RR
347      endif
348
349      end function yxz_ext_re_rq_p
350
351      !#####
352      !
353      ! Name:
354      !           yxz_ext_im_rq_p
355      !
356      ! Revised:
357      !           February 29, 2000
358      !
359      ! Purpose:
360      !           Auxiliary function used by the function yxz_ext_rq. Evaluates
361      !           the imaginary part of kernel used to compute the exterior cross
362      !           coupling contribution to the fields in a longitudinal slot due to
363      !           currents in a transverse slot.
364      !
365      ! Usage:
366      !           yxz_ext_im_rq_p(dim,x)
367      !
368      ! Arguments:
369      !           dim      - Obsolete!
370      !           x        - Independent variable (Input)
371      !
372      ! Modules:
373      !           const,quad_var,slot_var,mom_var,gen_var
374      !

```

```

375 ! Subroutines:
376 !         dsinc
377 !
378 function yxz_ext_im_rq_p(dim,x)
379 !modules
380     use const, only : pi,wp
381     use slot_var, only : Lx,Lz,x0,x00,w
382     use mom_var, only : kr,kq
383     use gen_var, only : k0
384     !implicit statement
385     implicit none
386     !declaration of intrinsic functions
387     intrinsic dsqrt,dsin,dcos
388     !declaration of functions
389     real(wp)          :: dsinc
390     !list of calling arguments
391     real(wp)          :: yxz_ext_im_rq_p
392     real(wp), dimension(1:2),intent(in):: x
393     integer ,intent(in) :: dim
394     !list of local variables
395     real(wp)          :: R1,R2,xii
396
397     !xi=x(1)    zeta=x(2)
398     xii=(x0-x00)-x(1)
399     R1=dsqrt(xii**2+(w/2.0_wp-x(2))**2)
400     R2=dsqrt(xii**2+(w/2.0_wp+x(2))**2)
401
402     yxz_ext_im_rq_p=dsin(kq*(Lz+x(2)))*&
403         dcos(kr*(Lx+x(1)))*&
404         (dsinc(k0*R2)-dsinc(k0*R1))
405
406 end function yxz_ext_im_rq_p
407
408 !#####
409 !
410 ! Name:
411 !     y_ext
412 !
413 ! Revised:
414 !     February 29, 2000
415 !
416 ! Purpose:
417 !     Computes the exterior self contribution to the MoM matrix for any
418 !     slot of length L. Used in case the slot radiates into half space.
419 !
420 ! Usage:
421 !     y_ext(p,q,L)
422 !
423 ! Arguments:
424 !     p     - order of expansion function (Input)
425 !     q     - order of weighting function (Input)
426 !     L     - length of slot (Input)
427 !
428 ! Modules:
429 !     const,slot_var,mom_var,gen_var
430 !
431 ! Subroutines:
432 !     y2,y1
433 !

```

```

434 function y_ext(p,q,L)
435     !modules
436     use const, only : pi,cj,Z0,mu_0,wp
437     use mom_var, only : y1_mode,y2_mode
438     use gen_var, only : k0,omega
439     use slot_var, only : Ls
440     !implicit statement
441     implicit none
442     !declaration of intrinsic functions
443     intrinsic mod
444     !declaration of functions
445     complex(wp)      :: y2,y1
446     !list of calling arguments
447     complex(wp)      :: y_ext
448     real(wp),intent(in) :: L
449     integer,intent(in) :: p,q
450
451     Ls=L
452     if(mod((p+q),2).eq.1) then
453         y_ext=(0.0_wp,0.0_wp)
454     else if ((p.ne.q).and.(mod((p+q),2).ne.1)) then
455         y1_mode=q      !y1(q)
456         y_ext=((k0*2.0_wp*Ls)**2-(q*pi)**2)*p*y1()
457         y1_mode=p      !y1(p)
458         y_ext=(y_ext-((k0*2.0_wp*Ls)**2-(p*pi)**2)*q*y1())/ &
459             (cj*pi**2*omega*(q**2-p**2))
460     else if (p.eq.q) then
461         y1_mode=q      !y1(q)
462         y2_mode=q      !y2(q)
463         y_ext=1.0_wp/(cj*2.0_wp*pi*omega)*((k0*2.0_wp*Ls)**2-(q*pi)**2)* &
464             y2()-q/(cj*omega)*y1()
465     end if
466
467     y_ext=2.0_wp*Z0*y_ext/mu_0
468
469 end function y_ext
470
471 !#####
472 !
473 ! Name:
474 !     y1
475 !
476 ! Revised:
477 !     February 29, 2000
478 !
479 ! Purpose:
480 !     Auxiliary function used by the function y_ext. Computes the value
481 !     of the function y1.
482 !
483 ! Usage:
484 !     y1()
485 !
486 ! Arguments:
487 !
488 ! Modules:
489 !     const,quad_var,slot_var,mom_var,gen_var
490 !
491 ! Subroutines:
492 !     y1_re_r1_pp,y1_re_r2_pp,y1_im_pp,inv_r_quad_int

```

```

493 !
494 ! IMSL subroutines:
495 !     dq3nd
496 !
497 function y1()
498     !modules
499     use const, only : cj,pi,wp
500     use gen_var, only : k0
501     use slot_var, only : w,Ls
502     use mom_var, only : y1_mode
503     use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,xi_1,zeta_1,&
504         maxfcn,errrel,errabs
505     !implicit statement
506     implicit none
507     !declaration of intrinsic functions
508     intrinsic sum
509     !declaration of functions
510     real(wp),external :: y1_re_r1_pp,y1_re_r2_pp,y1_im_pp
511     real(wp) :: inv_r_quad_int
512     !list of calling arguments
513     complex(wp) :: y1
514     !list of local variables
515     real(wp), dimension(1:2) :: upper
516     real(wp), dimension(1:2) :: lower
517     real(wp), dimension(1:3) :: re_val
518     real(wp) :: im_val
519     real(wp) :: errest
520     integer :: ier
521
522     upper=(/0.5_wp*w,0.5_wp/)
523     lower=(/0.0_wp,0.0_wp/)
524
525     !real part of 1/R1 term
526     f_0=0.0_wp
527     df_dxi=0.0_wp
528     df_dzeta=y1_mode*pi
529     xi_0=0.0_wp
530     zeta_0=0.5_wp
531     call dq3nd (y1_re_r1_pp,2,lower,upper,&
532         maxfcn,errabs,errrel,ier,re_val(1),errest)
533
534     !change of variable: zeta -> zeta/(2*Lz)
535     df_dzeta=y1_mode*pi/(2.0_wp*Ls)
536     xi_1=upper(1)-xi_0
537     zeta_1=2.0_wp*Ls*(lower(2)-zeta_0)
538     re_val(2)=inv_r_quad_int()/(2.0_wp*Ls)
539
540     !real part of 1/R2 term
541     call dq3nd (y1_re_r2_pp,2,lower,upper,&
542         maxfcn,errabs,errrel,ier,re_val(3),errest)
543
544     !imaginary part
545     call dq3nd (y1_im_pp,2,lower,upper,&
546         maxfcn,errabs,errrel,ier,im_val,errest)
547
548     y1=sum(re_val)+cj*k0*im_val
549
550 end function y1
551

```

```

552 !#####
553 !
554 ! Name:
555 !     y2
556 !
557 ! Revised:
558 !     February 29, 2000
559 !
560 ! Purpose:
561 !     Auxiliary function used by the function y_ext. Computes the value
562 !     of the function y2.
563 !
564 ! Usage:
565 !     y2()
566 !
567 ! Arguments:
568 !
569 ! Modules:
570 !     const,quad_var,slot_var,mom_var,gen_var
571 !
572 ! Subroutines:
573 !     y2_re_pp,y2_im_pp,inv_r_quad_int
574 !
575 ! IMSL subroutines:
576 !     dq3nd
577 !
578 function y2()
579 !modules
580 use const, only : cj,wp
581 use gen_var, only : k0
582 use slot_var, only : w,ls
583 use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,xi_1,zeta_1,&
584     maxfcn,errrel,errabs
585 !implicit statement
586 implicit none
587 !declaration of intrinsic functions
588 intrinsic sum
589 !declaration of functions
590 real(wp), external      :: y2_re_pp,y2_im_pp
591 real(wp)                :: inv_r_quad_int
592 !list of calling arguments
593 complex(wp)            :: y2
594 !list of local variables
595 real(wp), dimension(1:2) :: re_val
596 real(wp), dimension(1:2) :: upper
597 real(wp), dimension(1:2) :: lower
598 real(wp)                :: errest
599 real(wp)                :: im_val
600 integer                 :: ier
601
602 upper=(/w/2.0_wp,1.0_wp/)
603 lower=(/0.0_wp,0.0_wp/)
604
605 !real part of 1/R term
606 xi_0=0.0_wp
607 zeta_0=0.0_wp
608 f_0=1.0_wp
609 df_dxi=0.0_wp
610 df_dzeta=0.0_wp

```



```

611
612     call dq3nd (y2_re_pp,2,lower,upper,&
613               maxfcn,errabs,errrel,ier,re_val(1),errest)
614
615     !change of variable: zeta -> zeta/(2*Lz)
616     xi_1=upper(1)-xi_0
617     zeta_1=2.0_wp*Ls*(upper(2)-zeta_0)
618     re_val(2)=inv_r_quad_int()/(2.0_wp*Ls) !sum(reg_val)
619
620     !imaginary part
621     call dq3nd (y2_im_pp,2,lower,upper,&
622               maxfcn,errabs,errrel,ier,im_val,errest)
623
624     y2=sum(re_val)-cj*k0*im_val
625
626 end function y2
627
628 !#####
629 !
630 ! Name:
631 !     y1_im_pp
632 !
633 ! Revised:
634 !     February 29, 2000
635 !
636 ! Purpose:
637 !     Auxiliary function used by the function y1. Evaluates the
638 !     imaginary part of the kernel integrated in y1.
639 !
640 ! Usage:
641 !     y1_im_pp(dim,x)
642 !
643 ! Arguments:
644 !     dim - Obsolete!
645 !     x   - Independent variable (Input)
646 !
647 ! Modules:
648 !     const,slot_var,mom_var,gen_var
649 !
650 ! Subroutines:
651 !     dsinc
652 !
653 function y1_im_pp(dim,x)
654     !modules
655     use const, only : pi,wp
656     use slot_var, only : Ls
657     use mom_var, only : y1_mode
658     use gen_var, only : k0
659     !implicit statement
660     implicit none
661     !declaration of intrinsic functions
662     intrinsic dsin,dsqrt
663     !declaration of functions
664     real(wp)      :: dsinc
665     !list of calling arguments
666     real(wp)      :: y1_im_pp
667     real(wp), dimension(1:2),intent(in) :: x
668     integer,intent(in)      :: dim
669     !list of local variables

```

```

670     real(wp)          :: R1,R2
671
672     R1=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)-0.5_wp)**2)
673     R2=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)+0.5_wp)**2)
674
675     y1_im_pp=dsin(y1_mode*pi*(x(2)+0.5_wp))&
676             *(dsinc(k0*R2)-(-1.0_wp)**y1_mode*dsinc(k0*R1))
677
678 end function y1_im_pp
679
680 !#####
681 !
682 ! Name:
683 !     y1_re_r1_pp
684 !
685 ! Revised:
686 !     February 29, 2000
687 !
688 ! Purpose:
689 !     Auxiliary function used by the function y1. Evaluates partially
690 !     the real part of the kernel integrated in y1.
691 !
692 ! Usage:
693 !     y1_re_r1_pp(dim,x)
694 !
695 ! Arguments:
696 !     dim - Obsolete!
697 !     x   - Independent variable (Input)
698 !
699 ! Modules:
700 !     const,slot_var,mom_var,gen_var
701 !
702 function y1_re_r1_pp(dim,x)
703     !modules
704     use const, only : pi,wp
705     use slot_var, only : Ls
706     use mom_var, only : y1_mode
707     use gen_var, only : k0
708     use quad_var, only : df_dzeta,zeta_0
709     !implicit statement
710     implicit none
711     !declaration of intrinsic functions
712     intrinsic dsin,dcos,dsqrt
713     !list of calling arguments
714     real(wp)          :: y1_re_r1_pp
715     real(wp), dimension(1:2),intent(in) :: x
716     integer,intent(in) :: dim
717     !list of local variables
718     real(wp)          :: R1
719
720     !xi=x(1)    sigma=zeta=x(2)
721     R1=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)-0.5_wp)**2)
722
723     if(R1.lt.1E-8_wp) then
724         y1_re_r1_pp=0.0_wp
725     else
726         y1_re_r1_pp=(dsin(y1_mode*pi*(x(2)+0.5_wp))&
727                 *(-1.0_wp)**y1_mode*dcos(k0*R1)-(x(2)-zeta_0)*df_dzeta)/R1
728     endif

```

```

729
730 end function y1_re_r1_pp
731
732 !#####
733 !
734 ! Name:
735 !     y1_re_r2_pp
736 !
737 ! Revised:
738 !     February 29, 2000
739 !
740 ! Purpose:
741 !     Auxiliary function used by the function y1. Evaluates partially
742 !     the real part of the kernel integrated in y1.
743 !
744 ! Usage:
745 !     y1_re_r2_pp(dim,x)
746 !
747 ! Arguments:
748 !     dim - Obsolete!
749 !     x   - Independent variable (Input)
750 !
751 ! Modules:
752 !     const,slot_var,mom_var,gen_var
753 !
754 function y1_re_r2_pp(dim,x)
755 !modules
756 use const, only : pi,wp
757 use slot_var, only : Ls
758 use mom_var, only : y1_mode
759 use gen_var, only : k0
760 !implicit statement
761 implicit none
762 !declaration of intrinsic functions
763 intrinsic dsin,dcos,dsqrt
764 !list of calling arguments
765 real(wp) :: y1_re_r2_pp
766 real(wp), dimension(1:2),intent(in) :: x
767 integer,intent(in) :: dim
768 !list of local variables
769 real(wp) :: R2
770
771 !xi=x(1) sigma=x(2)
772 R2=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)+0.5_wp)**2)
773
774 y1_re_r2_pp=-dsin(y1_mode*pi*(x(2)+0.5_wp))*dcos(k0*R2)/R2
775
776 end function y1_re_r2_pp
777
778 !#####
779 !
780 ! Name:
781 !     y2_im_pp
782 !
783 ! Revised:
784 !     February 29, 2000
785 !
786 ! Purpose:
787 !     Auxiliary function used by the function y2. Evaluates the

```

```

788      !           imaginary part of the kernel integrated in y2.
789      !
790      ! Usage:
791      !       y2_im_pp(dim,x)
792      !
793      ! Arguments:
794      !   dim   - Obsolete!
795      !   x     - Independent variable (Input)
796      !
797      ! Modules:
798      !       const,slot_var,mom_var,gen_var
799      !
800      ! Subroutines:
801      !       dsinc
802      !
803      function y2_im_pp(dim,x)
804      !modules
805      use const, only : pi,wp
806      use slot_var, only : Ls
807      use mom_var, only : y2_mode
808      use gen_var, only : k0
809      !implicit statement
810      implicit none
811      !declaration of intrinsic functions
812      intrinsic dsin,dcos,dsqrt
813      !declaration of functions
814      real(wp)      :: dsinc
815      !list of calling arguments
816      real(wp)      :: y2_im_pp
817      real(wp), dimension(1:2),intent(in) :: x
818      integer,intent(in)      :: dim
819      !list of local variables
820      real(wp)      :: R
821
822      R=dsqrt(x(1)**2+(2.0_wp*Ls)**2*x(2)**2)
823
824      y2_im_pp=dsinc(k0*R)*(dcos(y2_mode*pi*x(2))*(1.0_wp-x(2))&
825      +dsin(y2_mode*pi*x(2))/(y2_mode*pi))
826
827      end function y2_im_pp
828
829      !#####
830      !
831      ! Name:
832      !       y2_re_pp
833      !
834      ! Revised:
835      !       February 29, 2000
836      !
837      ! Purpose:
838      !       Auxiliary function used by the function y2. Evaluates the
839      !       real part of the kernel integrated in y2.
840      !
841      ! Usage:
842      !       y2_re_pp(dim,x)
843      !
844      ! Arguments:
845      !   dim   - Obsolete!
846      !   x     - Independent variable (Input)

```

```

847 !
848 ! Modules:
849 !         const,slot_var,mom_var,gen_var
850 !
851 function y2_re_pp(dim,x)
852     !modules
853     use const, only : pi,cj,wp
854     use slot_var, only : Ls
855     use mom_var, only : y2_mode
856     use gen_var, only : k0
857     use quad_var, only : f_0
858     !implicit statement
859     implicit none
860     !declaration of intrinsic functions
861     intrinsic dsin,dcos,dsqrt
862     !list of calling arguments
863     real(wp)          :: y2_re_pp
864     real(wp), dimension(1:2),intent(in) :: x
865     integer,intent(in) :: dim
866     !list of local variables
867     real(wp)          :: R
868
869     R=dsqrt(x(1)**2+(2.0_wp*Ls)**2*x(2)**2)
870
871     if(R.lt.1E-8_wp) then
872         y2_re_pp=0.0_wp
873     else
874         y2_re_pp=(dcos(k0*R)*(&
875             (1.0_wp-x(2))*dcos(y2_mode*pi*x(2))&
876             +dsin(y2_mode*pi*x(2))/(pi*y2_mode))-f_0/R
877     endif
878
879 end function y2_re_pp

```

```

1  !#####
2  !
3  ! Name:
4  !     yzx_ppwg_ps
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Computes the exterior cross coupling contribution to the fields
11 !     in a transverse slot due to currents in a longitudinal slot. Used
12 !     in case the slot radiates into a PP waveguide.
13 !
14 ! Usage:
15 !     yzx_ppwg_ps()
16 !
17 ! Arguments:
18 !
19 ! Modules:
20 !     const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
21 !
22 ! Subroutines:
23 !     yzx_ppwg_ps_p_revised_c,yzx_ppwg_ps_p,int_wavg
24 !
25 ! IMSL subroutines:
26 !     dqdag

```

```

27      !
28      function yzx_ppwg_ps()
29          !modules
30          use const, only      : pi,cj,Z0,mu_0,wp
31          use slot_var, only   : Lx,Lz,w,x00,x00_pp,x0,x0_pp
32          use mom_var, only    : p,s,kp,ks,kp_sq,ks_sq
33          use gen_var, only    : k0,omega
34          use ppwg_var, only   : m,a1,kx,kx_sq
35          use wg_var, only     : a
36          use quad_var, only   : kw,kz_max,errrel,errabs,c_flag
37          !implicit statement
38          implicit none
39          !declaration of intrinsic functions
40          intrinsic dsin,dcos,mod
41          !declaration of functions
42          real(wp), external    :: yzx_ppwg_ps_p_revised_c
43          real(wp), external    :: yzx_ppwg_ps_p
44          !list of calling arguments
45          complex(wp)          :: yzx_ppwg_ps
46          !list of local parameters
47          integer,parameter    :: irule=2
48          real(wp),parameter    :: upper=pi
49          real(wp),parameter    :: lower=0.0
50          !list of local variables
51          complex(wp)          :: sum
52          real(wp)              :: re_val
53          real(wp),dimension(1:2) :: im_val
54          real(wp)              :: errest,w_m
55
56          if(mod(p,2).eq.1) then !p odd
57              yzx_ppwg_ps=(0.0_wp,0.0_wp)
58          else
59              !init. of aux. var
60              x0_pp=x0-(a-a1)/2.0_wp
61              x00_pp=x00-(a-a1)/2.0_wp
62              kp=p*pi/(2.0_wp*Lz)
63              ks=s*pi/(2.0_wp*Lx)
64              kp_sq=kp**2
65              ks_sq=ks**2
66              kw=Lz
67              kz_max=max(k0,kp)
68              sum=(0.0_wp,0.0_wp)
69              do m=0,100 !100 gives good conv. unless you test a "free space" case
70                  kx=(m*pi/a1)
71                  kx_sq=kx**2
72                  w_m=dsin(kx*w/2.0_wp)*dcos(kx*x0_pp)/(ks_sq-kx_sq)
73                  if(mod(s,2).eq.1) then !s odd
74                      w_m=w_m*dcos(kx*Lx)*dsin(kx*x00_pp)
75                  else !s even
76                      w_m=-w_m*dcos(kx*x00_pp)*dsin(kx*Lx)
77                  end if
78                  !correct w_m in case m<1
79                  if (m.eq.0) then
80                      w_m=w_m*0.5_wp
81                  end if
82
83                  !real part of revised contour
84                  if(kx.lt.k0) then
85                      c_flag=1

```

```

86         call dqdag(yzx_ppwg_ps_p_revised_c, lower, upper,&
87                 errabs, errrel, irule, re_val,errest)
88     else
89         re_val=0.0_wp
90     end if
91     !imaginary part of revised contour
92     c_flag=2
93     call dqdag(yzx_ppwg_ps_p_revised_c, lower, upper,&
94             errabs, errrel, irule, im_val(1),errest)
95     !integration by weighted averages from kz=2kz_max->infty
96     call int_wavg(yzx_ppwg_ps_p,2.0_wp*kz_max,-2,2,im_val(2))
97
98     sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
99 enddo !m loop
100
101     yzx_ppwg_ps=-4.0_wp*Z0*p*s*pi/(a1*mu_0*omega*Lx*Lz)*sum
102
103     endif
104 end function yzx_ppwg_ps
105
106 !#####
107 !
108 ! Name:
109 !     yzx_ppwg_ps_p_revised_c
110 !
111 ! Revised:
112 !     February 29, 2000
113 !
114 ! Purpose:
115 !     Auxiliary function used by the function yzx_ppwg_ps. Evaluates
116 !     the kernel used to compute the exterior cross coupling
117 !     contribution to the fields in a transverse slot due to currents
118 !     in a longitudinal slot. The evaluation is performed along a
119 !     deformed contour in the complex kz plane.
120 !
121 ! Usage:
122 !     yzx_ppwg_ps_p_revised_c(theta)
123 !
124 ! Arguments:
125 !     theta - Independent variable (Input)
126 !
127 ! Modules:
128 !     const,quad_var,mom_var,gen_var,ppwg_var
129 !
130 function yzx_ppwg_ps_p_revised_c(theta)
131     !modules
132     use const, only : cj,wp
133     use gen_var, only : k0,k0_sq
134     use mom_var, only : kp_sq
135     use ppwg_var, only : kx_sq,ky,kz,kz_sq
136     use quad_var, only : kw,kz_max,c_flag
137     !implicit statement
138     implicit none
139     !declaration of intrinsic functions
140     intrinsic dsin,cdsin,dcos,cdsqrt,dcmplx
141     !list of calling arguments
142     real(wp)      :: yzx_ppwg_ps_p_revised_c
143     !list of local variables
144     complex(wp)  :: tmp

```

```

145     real(wp)    :: theta,csth,snth
146
147     csth=dcos(theta)
148     snth=dsin(theta)
149
150     !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
151     kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
152     kz_sq=kz**2
153     ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
154     tmp=kz*cdsin(kw*kz)/ky/(kp_sq-kz_sq)*(kz_max*snth+0.05_wp*k0*cj*csth)
155     if(c_flag.eq.1) then
156         yzx_ppwg_ps_p_revised_c=dreal(tmp)
157     else
158         yzx_ppwg_ps_p_revised_c=dimag(tmp)
159     endif
160
161 end function yzx_ppwg_ps_p_revised_c
162
163 !#####
164 !
165 ! Name:
166 !     yzx_ppwg_ps_p
167 !
168 ! Revised:
169 !     February 29, 2000
170 !
171 ! Purpose:
172 !     Auxiliary function used by the function yzx_ppwg_ps. Evaluates
173 !     the kernel used to compute the exterior cross coupling
174 !     contribution to the fields in a transverse slot due to currents
175 !     in a longitudinal slot. Used in the interval from kz=kz_max->infty
176 !     where kz_max is chosen so all singular points are avoided.
177 !
178 ! Usage:
179 !     yzx_ppwg_ps_p(kz)
180 !
181 ! Arguments:
182 !     kz     - Independent variable (Input)
183 !
184 ! Modules:
185 !     const,quad_var,mom_var,gen_var,ppwg_var
186 !
187 function yzx_ppwg_ps_p(kz)
188     !modules
189     use const, only : wp
190     use gen_var, only : k0_sq
191     use mom_var, only : kp_sq
192     use ppwg_var, only : kx_sq
193     use quad_var, only : kw
194     !implicit statement
195     implicit none
196     !declaration of intrinsic functions
197     intrinsic dsqrt
198     !declaration of functions
199     real(wp)    :: dsinc
200     !list of calling arguments
201     real(wp)    :: yzx_ppwg_ps_p
202     real(wp),intent(in) :: kz
203     !list of local variables

```



```

204     real(wp)    :: kz_sq
205
206     kz_sq=kz**2
207     yzx_ppwg_ps_p=kz*dsinc(kw*kz)/&
208                 dsqrt(kz_sq+kx_sq-k0_sq)/(kp_sq-kz_sq)
209
210 end function yzx_ppwg_ps_p
211
212 !#####
213 !
214 ! Name:
215 !     yxz_ppwg_rq
216 !
217 ! Revised:
218 !     February 29, 2000
219 !
220 ! Purpose:
221 !     Computes the exterior cross coupling contribution to the fields
222 !     in a longitudinal slot due to currents in a transverse slot. Used
223 !     in case the slot radiates into a PP waveguide.
224 !
225 ! Usage:
226 !     yxz_ppwg_rq()
227 !
228 ! Arguments:
229 !
230 ! Modules:
231 !     const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
232 !
233 ! Subroutines:
234 !     yxz_ppwg_rq_p_revised_c,yxz_ppwg_rq_p,int_wavg
235 !
236 ! IMSL subroutines:
237 !     dqdag
238 !
239 function yxz_ppwg_rq()
240     !modules
241     use const,      only    :   pi,cj,Z0,mu_0,wp
242     use slot_var,   only    :   Lx,Lz,w,x00,x00_pp,x0,x0_pp
243     use mom_var,    only    :   r,q,kq,kr,kq_sq,kr_sq
244     use gen_var,    only    :   k0,omega
245     use ppwg_var,   only    :   m,a1,kx,kx_sq
246     use wg_var,     only    :   a
247     use quad_var,   only    :   kw,kz_max,errrel,errabs,c_flag
248     !implicit statement
249     implicit none
250     !declaration of intrinsic functions
251     intrinsic dsin,dcos,mod
252     !declaration of functions
253     real(wp), external    :: yxz_ppwg_rq_p_revised_c
254     real(wp), external    :: yxz_ppwg_rq_p
255     !list of calling arguments
256     complex(wp)          :: yxz_ppwg_rq
257     !list of local parameters
258     integer,parameter    :: irule=2
259     real(wp),parameter   :: upper=pi
260     real(wp),parameter   :: lower=0.0
261     !list of local variables
262     complex(wp)          :: sum

```

```

263     real(wp)                :: re_val
264     real(wp),dimension(1:3) :: im_val
265     real(wp)                :: errest,w_m
266
267     if(mod(q,2).eq.1) then !q odd
268         yxz_ppwg_rq=(0.0_wp,0.0_wp)
269     else
270         !init. of aux. var
271         x0_pp=x0-(a-a1)/2.0_wp
272         x00_pp=x00-(a-a1)/2.0_wp
273         kq=q*pi/(2.0_wp*Lz)
274         kr=r*pi/(2.0_wp*Lx)
275         kq_sq=kq**2
276         kr_sq=kr**2
277         kz_max=0.95_wp*max(k0,kq)
278
279         sum=(0.0_wp,0.0_wp)
280         do m=0,100 !100 gives good conv. unless you test a "free space" case
281             kx=(m*pi/a1)
282             kx_sq=kx**2
283             w_m=kx*dcos(kx*x0_pp)/(kr_sq-kx_sq)
284             if(mod(r,2).eq.1) then !r odd
285                 w_m=w_m*dcos(kx*Lx)*dsin(kx*x00_pp)
286             else !r even
287                 w_m=-w_m*dcos(kx*x00_pp)*dsin(kx*Lx)
288             end if
289             !correct w_m in case m<1
290             if (m.eq.0) then
291                 w_m=w_m*0.5_wp
292             end if
293
294             !real part of revised contour
295             if(kx.lt.k0) then
296                 c_flag=1
297                 call dxdag(yxz_ppwg_rq_p_revised_c, lower, upper,&
298                     errabs, errrel, irule, re_val,errest)
299             else
300                 re_val=0.0_wp
301             end if
302             !imaginary part of revised contour
303             c_flag=2
304             call dxdag(yxz_ppwg_rq_p_revised_c, lower, upper,&
305                 errabs, errrel, irule, im_val(1),errest)
306             !integration by weighted averages from kz=2kz_max->infy
307             kw=Lz-w/2.0_wp
308             call int_wavg(yxz_ppwg_rq_p,2.0_wp*kz_max,-3,1,im_val(2))
309             kw=Lz+w/2.0_wp
310             call int_wavg(yxz_ppwg_rq_p,2.0_wp*kz_max,-3,1,im_val(3))
311
312             sum=sum+w_m*(re_val+c_j*(im_val(1)+im_val(2)-im_val(3)))
313         enddo !m loop
314
315         yxz_ppwg_rq=2.0_wp*Z0*r*q*pi/(a1*mu_0*omega*Lx*Lz)*sum
316
317     endif
318 end function yxz_ppwg_rq
319
320 !#####
321 !

```

```

322 ! Name:
323 !       yxz_ppwg_rq_p_revised_c
324 !
325 ! Revised:
326 !       February 29, 2000
327 !
328 ! Purpose:
329 !       Auxiliary function used by the function yxz_ppwg_rq. Evaluates
330 !       the kernel used to compute the exterior cross coupling
331 !       contribution to the fields in a longitudinal slot due to currents
332 !       in a transverse slot. The evaluation is performed along a
333 !       deformed contour in the complex kz plane.
334 !
335 ! Usage:
336 !       yxz_ppwg_rq_p_revised_c(theta)
337 !
338 ! Arguments:
339 !       theta - Independent variable (Input)
340 !
341 ! Modules:
342 !       const,quad_var,mom_var,gen_var,ppwg_var
343 !
344 function yxz_ppwg_rq_p_revised_c(theta)
345 !modules
346 use const, only : cj,wp
347 use gen_var, only : k0,k0_sq
348 use mom_var, only : kq_sq
349 use slot_var, only : Lz,w
350 use ppwg_var, only : kx_sq,ky,kz,kz_sq
351 use quad_var, only : kz_max,c_flag
352 !implicit statement
353 implicit none
354 !declaration of intrinsic functions
355 intrinsic dsin,cdsin,dcos,cdsqrt,dcmplx
356 !list of calling arguments
357 real(wp)      :: yxz_ppwg_rq_p_revised_c
358 !list of local variables
359 complex(wp)  :: tmp
360 real(wp)     :: theta,csth,snth,kw1,kw2
361
362 csth=dcos(theta)
363 snth=dsin(theta)
364 kw1=w/2.0_wp-Lz
365 kw2=w/2.0_wp+Lz
366
367 !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
368 kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
369 kz_sq=kz**2
370 ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
371
372 tmp=(cdcos(kz*kw1)-cdcos(kz*kw2))/&
373      ky/(kq_sq-kz_sq)*(kz_max*snth+0.05_wp*k0*cj*csth)
374
375 if(c_flag.eq.1) then
376     yxz_ppwg_rq_p_revised_c=dreal(tmp)
377 else
378     yxz_ppwg_rq_p_revised_c=dimag(tmp)
379 endif
380

```

```

381 end function yxz_ppwg_rq_p_revised_c
382
383 !#####
384 !
385 ! Name:
386 !     yxz_ppwg_rq_p
387 !
388 ! Revised:
389 !     February 29, 2000
390 !
391 ! Purpose:
392 !     Auxiliary function used by the function yxz_ppwg_rq. Evaluates
393 !     the kernel used to compute the exterior cross coupling
394 !     contribution to the fields in a longitudinal slot due to currents
395 !     in a transverse slot. Used in the interval from kz=kz_max->infty
396 !     where kz_max is chosen so all singular points are avoided.
397 !
398 ! Usage:
399 !     yxz_ppwg_rq_p(kz)
400 !
401 ! Arguments:
402 !     kz     - Independent variable (Input)
403 !
404 ! Modules:
405 !     const,quad_var,mom_var,gen_var,ppwg_var
406 !
407 function yxz_ppwg_rq_p(kz)
408     !modules
409     use const, only : wp
410     use gen_var, only : k0_sq,k0
411     use mom_var, only : kq_sq
412     use ppwg_var, only : kx,kx_sq
413     use quad_var, only : kw
414     !implicit statement
415     implicit none
416     !declaration of intrinsic functions
417     intrinsic dsqrt,dcos
418     !list of calling arguments
419     real(wp)          :: yxz_ppwg_rq_p
420     real(wp),intent(in) :: kz
421     !list of local variables
422     real(wp)          :: kz_sq
423
424     kz_sq=kz**2
425     yxz_ppwg_rq_p=dcos(kz*kw)/&
426                 dsqrt(kz_sq+kx_sq-k0_sq)/(kq_sq-kz_sq)
427
428 end function yxz_ppwg_rq_p
429
430 !#####
431 !
432 ! Name:
433 !     yxx_ppwg_rs
434 !
435 ! Revised:
436 !     February 29, 2000
437 !
438 ! Purpose:
439 !     Computes the exterior self contribution to the fields in a

```

```

440 !           transverse slot. Used in case the slot radiates into a PP
441 !           waveguide.
442 !
443 ! Usage:
444 !           yxx_ppwg_rs()
445 !
446 ! Arguments:
447 !
448 ! Modules:
449 !           const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
450 !
451 ! Subroutines:
452 !           yxx_ppwg_rs_p_revised_c,yxx_ppwg_rs_p,int_wavg
453 !
454 ! IMSL subroutines:
455 !           dqdag
456 !
457 function yxx_ppwg_rs()
458     !modules
459     use const, only      : pi,cj,Z0,mu_0,wp
460     use slot_var, only   : Lx,w,x00,x00_pp
461     use mom_var, only    : r,s,kr_sq,ks_sq
462     use gen_var, only    : k0,k0_sq,omega
463     use ppwg_var, only   : m,a1,kx,kx_sq
464     use wg_var, only     : a
465     use quad_var, only   : kw,kz_max,errrel,errabs,c_flag
466     !implicit statement
467     implicit none
468     !declaration of intrinsic functions
469     intrinsic dsin,dcos,mod
470     !declaration of functions
471     real(wp), external   :: yxx_ppwg_rs_p_revised_c
472     real(wp), external   :: yxx_ppwg_rs_p
473     !list of calling arguments
474     complex(wp)          :: yxx_ppwg_rs
475     !list of local parameters
476     integer,parameter    :: irule=2
477     real(wp),parameter   :: upper=pi
478     real(wp),parameter   :: lower=0.0
479     !list of local variables
480     complex(wp)          :: sum
481     real(wp),dimension(1:2) :: im_val
482     real(wp)              :: re_val
483     real(wp)              :: errest,w_m
484
485     if(mod((r+s),2).eq.1) then
486         yxx_ppwg_rs=(0.0_wp,0.0_wp)
487     else
488         !init. of aux. var
489         kz_max=k0
490         kw=w/2.0_wp
491         x00_pp=x00-(a-a1)/2.0_wp
492         kr_sq=(r*pi/(2.0_wp*Lx))**2
493         ks_sq=(s*pi/(2.0_wp*Lx))**2
494         sum=(0.0_wp,0.0_wp)
495         do m=0,100 !100 gives good conv. unless you test a "free space" case
496             kx=(m*pi/a1)
497             kx_sq=kx**2
498             w_m=(k0_sq-kx_sq)/((kr_sq-kx_sq)*(ks_sq-kx_sq))

```

```

499         if(mod(r,2).eq.1) then !r and s odd
500             w_m=w_m*(dcos(kx*Lx)*dsin(kx*x00_pp))**2
501         else !r and s even
502             w_m=w_m*(dcos(kx*x00_pp)*dsin(kx*Lx))**2
503         end if
504         !correct w_m in case m<1
505         if (m.eq.0) then
506             w_m=w_m*0.5_wp
507         end if
508         !real part of revised contour
509         if(kx.lt.k0) then
510             c_flag=1
511             call dqdag(yxx_ppwg_rs_p_revised_c, lower, upper,&
512                 errabs, errrel, irule, re_val,errest)
513         else
514             re_val=0.0_wp
515         end if
516         !imaginary part of revised contour
517         c_flag=2
518         call dqdag(yxx_ppwg_rs_p_revised_c, lower, upper,&
519             errabs, errrel, irule, im_val(1),errest)
520         !integration by weighted averages from kz=2kz_max->infty
521         call int_wavg(yxx_ppwg_rs_p,2.0_wp*kz_max,-2,2,im_val(2))
522
523         sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
524     enddo !m loop
525
526     yxx_ppwg_rs=2.0_wp*Z0*w*r*s*pi/(Lx**2*a1*mu_0*omega)*sum
527
528     endif
529 end function yxx_ppwg_rs
530
531 !#####
532 !
533 ! Name:
534 !     yxx_ppwg_rs_p_revised_c
535 !
536 ! Revised:
537 !     February 29, 2000
538 !
539 ! Purpose:
540 !     Auxiliary function used by the function yxx_ppwg_rs. Evaluates
541 !     the kernel used to compute the exterior self contribution to the
542 !     fields in a transverse slot. The evaluation is perfomed along a
543 !     deformed contour in the complex kz plane.
544 !
545 ! Usage:
546 !     yxx_ppwg_rs_p_revised_c(theta)
547 !
548 ! Arguments:
549 !     theta - Independent variable (Input)
550 !
551 ! Modules:
552 !     const,quad_var,mom_var,gen_var,ppwg_var
553 !
554 function yxx_ppwg_rs_p_revised_c(theta)
555     !modules
556     use const, only : cj,wp
557     use gen_var, only : k0,k0_sq

```

```

558     use slot_var, only : w
559     use ppwg_var, only : kx_sq,ky,kz,kz_sq
560     use quad_var, only : kw,kz_max,c_flag
561     !implicit statement
562     implicit none
563     !declaration of intrinsic functions
564     intrinsic dsin,cdsin,dcos,cdsqrtd,dcmplx
565     !list of calling arguments
566     real(wp)      :: yxx_ppwg_rs_p_revised_c
567     real(wp),intent(in) :: theta
568     !list of local variables
569     complex(wp) :: tmp
570     real(wp)    :: csth,snth
571
572     csth=dcos(theta)
573     snth=dsin(theta)
574
575     !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
576     kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
577     kz_sq=kz**2
578     ky=cdsqrtd(dcmplx(k0_sq-kx_sq-kz_sq))
579     tmp=cdsin(kw*kz)/(kw*kz)/ky*(kz_max*snth+0.05_wp*k0*cj*csth)
580
581     if(c_flag.eq.1) then
582         yxx_ppwg_rs_p_revised_c=dreal(tmp)
583     else
584         yxx_ppwg_rs_p_revised_c=dimag(tmp)
585     endif
586
587 end function yxx_ppwg_rs_p_revised_c
588
589 !#####
590 !
591 ! Name:
592 !     yxx_ppwg_rs_p
593 !
594 ! Revised:
595 !     February 29, 2000
596 !
597 ! Purpose:
598 !     Auxiliary function used by the function yxx_ppwg_rs. Evaluates
599 !     the kernel used to compute the exterior self contribution to the
600 !     fields in a longitudinal slot. Used in the interval from
601 !     kz=kz_max->infy where kz_max is chosen so all singular points
602 !     are avoided.
603 !
604 ! Usage:
605 !     yxx_ppwg_rs_p(kz)
606 !
607 ! Arguments:
608 !     kz      - Independent variable (Input)
609 !
610 ! Modules:
611 !     const,quad_var,gen_var,ppwg_var
612 !
613 function yxx_ppwg_rs_p(kz)
614     !modules
615     use const, only : wp
616     use gen_var, only : k0_sq

```

```

617     use ppwg_var, only : kx_sq
618     use quad_var, only : kw
619     !implicit statement
620     implicit none
621     !declaration of intrinsic functions
622     intrinsic dsqrt
623     !declaration of functions
624     real(wp)      :: dsinc
625     !list of calling arguments
626     real(wp)      :: yxx_ppwg_rs_p
627     real(wp),intent(in) :: kz
628
629     yxx_ppwg_rs_p=dsinc(kw*kz)/dsqrt(kz**2+kx_sq-k0_sq)
630
631 end function yxx_ppwg_rs_p
632
633 !#####
634 !
635 ! Name:
636 !     yzz_ppwg_pq
637 !
638 ! Revised:
639 !     February 29, 2000
640 !
641 ! Purpose:
642 !     Computes the exterior self contribution to the fields in a
643 !     longitudinal slot. Used in case the slot radiates into a PP
644 !     waveguide.
645 !
646 ! Usage:
647 !     yzz_ppwg_pq()
648 !
649 ! Arguments:
650 !
651 ! Modules:
652 !     const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
653 !
654 ! Subroutines:
655 !     yzz_ppwg_pq_p_revised_c,int_yzz_ppwg_pq_inf,int_wavg,dsinc
656 !
657 ! IMSL subroutines:
658 !     dqdag
659 !
660 function yzz_ppwg_pq()
661     !modules
662     use const, only : pi,cj,Z0,mu_0,wp
663     use slot_var, only : Lz,w,x0,x0_pp
664     use mom_var, only : p,q,kp,kq,kp_sq,kq_sq
665     use gen_var, only : k0,omega
666     use ppwg_var, only : m,a1,kx,kx_sq
667     use wg_var, only : a
668     use quad_var, only : kw,kz_max,errrel,errabs,c_flag
669     !implicit statement
670     implicit none
671     !declaration of intrinsic functions
672     intrinsic dcos,mod,max
673     !declaration of functions
674     real(wp), external      :: yzz_ppwg_pq_p_revised_c
675     real(wp)                :: dsinc,int_yzz_ppwg_pq_inf

```



```

676     !list of calling arguments
677     complex(wp)      :: yzz_ppwg_pq
678     !list of local parameters
679     integer,parameter :: irule=2
680     real(wp),parameter :: upper=pi
681     real(wp),parameter :: lower=0.0_wp
682     !list of local variables
683     complex(wp)      :: sum
684     real(wp)         :: re_val
685     real(wp),dimension(1:2) :: im_val
686     real(wp)         :: errest,w_m
687
688     if(mod((p+q),2).eq.1) then
689         yzz_ppwg_pq=(0.0_wp,0.0_wp)
690     else
691         !init. of aux. var
692         kp=p*pi/(2.0_wp*Lz)
693         kq=q*pi/(2.0_wp*Lz)
694         kp_sq=kp**2
695         kq_sq=kq**2
696         kz_max=0.95_wp*max(k0,kp,kq)
697         kw=2.0_wp*Lz
698         x0_pp=x0-(a-a1)/2.0_wp
699         sum=(0.0_wp,0.0_wp)
700         do m=0,100
701             kx=(m*pi/a1)
702             kx_sq=kx**2
703             w_m=dsinc(kx*w/2.0_wp)*(dcos(kx*x0_pp))**2
704             !correct w_m in case m<1
705             if (m.eq.0) then
706                 w_m=w_m*0.5
707             end if
708             !real part of revised contour
709             if(kx.lt.k0) then
710                 c_flag=1
711                 call dquadag(yzz_ppwg_pq_p_revised_c, lower, upper,&
712                     errabs, errrel, irule, re_val,errest)
713             else
714                 re_val=0.0_wp
715             end if
716             !imaginary part of revised contour
717             c_flag=2
718             call dquadag(yzz_ppwg_pq_p_revised_c, lower, upper,&
719                 errabs, errrel, irule, im_val(1),errest)
720             !integration of sommerfeld int. by weighted averages
721             !from kz=2kz_max->infy
722             im_val(2)=int_yzz_ppwg_pq_inf(2.0*kz_max)
723
724             sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
725         enddo !m loop
726
727         yzz_ppwg_pq=-Z0*w*p*q*pi/(a1*mu_0*omega*Lz**2)*sum
728
729     endif
730 end function yzz_ppwg_pq
731
732 !#####
733 !
734 ! Name:

```

```

735      !           yzz_ppwg_pq_p_revised_c
736      !
737      ! Revised:
738      !       February 29, 2000
739      !
740      ! Purpose:
741      !       Auxiliary function used by the function yzz_ppwg_pq. Evaluates
742      !       the kernel used to compute the exterior self contribution to the
743      !       fields in a longitudinal slot. The evaluation is performed along a
744      !       deformed contour in the complex kz plane.
745      !
746      ! Usage:
747      !       yzz_ppwg_pq_p_revised_c(theta)
748      !
749      ! Arguments:
750      !       theta - Independent variable (Input)
751      !
752      ! Modules:
753      !       const,quad_var,mom_var,gen_var,ppwg_var
754      !
755      function yzz_ppwg_pq_p_revised_c(theta)
756      !modules
757      use const, only : wp,cj
758      use gen_var, only : k0,k0_sq
759      use ppwg_var, only : kx_sq
760      use quad_var, only : kw,kz_max,c_flag
761      use mom_var, only : p,q,kp_sq,kq_sq
762      !implicit statement
763      implicit none
764      !declaration of intrinsic functions
765      intrinsic dsin,dcos,cdcos,cdsqrt,dcmplx
766      !list of calling arguments
767      real(wp)           :: yzz_ppwg_pq_p_revised_c
768      real(wp),intent(in) :: theta
769      !list of local variables
770      complex(wp)       :: ky,kz,kz_sq,tmp
771      real(wp)          :: cos_term,sin_term
772
773      cos_term=dcos(theta)
774      sin_term=dsin(theta)
775
776      !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
777      kz=kz_max*(1.0_wp-cos_term)+0.05_wp*k0*cj*sin_term
778      kz_sq=kz**2
779      ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
780      tmp=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)/ky*&
781          (kz_max*sin_term+0.05_wp*k0*cj*cos_term)
782
783      if(mod(p,2).eq.1) then !p and q odd
784          tmp=tmp*(cdcos(1.0_wp*kz*kw)+1.0_wp)
785      else !p and q even
786          tmp=tmp*(1.0_wp-cdcos(1.0_wp*kz*kw))
787      end if
788
789      if(c_flag.eq.1) then
790          yzz_ppwg_pq_p_revised_c=dreal(tmp)
791      else
792          yzz_ppwg_pq_p_revised_c=dimag(tmp)
793      endif

```

```

794
795 end function yzz_ppwg_pq_p_revised_c
796
797 !#####
798 !
799 ! Name:
800 !     int_yzz_ppwg_pq_inf
801 !
802 ! Revised:
803 !     February 29, 2000
804 !
805 ! Purpose:
806 !     Auxiliary function used by the function yzz_ppwg_pq. Integrates
807 !     the monotonic and oscilating part of the kernel used to compute
808 !     the exterior self contribution to the fields in a longitudinal
809 !     slot. Used in the interval from kz=kz_low->infy where kz_low is
810 !     chosen so all singular points are avoided.
811 !
812 ! Usage:
813 !     int_yzz_ppwg_pq_inf(kz_low)
814 !
815 ! Arguments:
816 !     kz_low - Lower integration limit (Input)
817 !
818 ! Modules:
819 !     const,mom_var,quad_var
820 !
821 ! Subroutines:
822 !     yzz_ppwg_pq_p_osc,yzz_ppwg_pq_p_mono,int_wavg
823 !
824 ! IMSL subroutines:
825 !     dqdagi
826 !
827 function int_yzz_ppwg_pq_inf(kz_low)
828     !modules
829     use const, only : wp
830     use mom_var, only : p,q
831     use quad_var, only : errrel,errabs
832     !implicit statement
833     implicit none
834     !declaration of functions
835     real(wp), external      :: yzz_ppwg_pq_p_osc,yzz_ppwg_pq_p_mono
836     !list of calling arguments
837     real(wp)                :: int_yzz_ppwg_pq_inf
838     real(wp),intent(in)    :: kz_low
839     !list of local variables
840     real(wp),dimension(1:2) :: subint
841     real(wp)                :: errest
842
843     ! integraton of monotonic part
844     call dqdagi(yzz_ppwg_pq_p_mono,kz_low,1,&
845               errabs,errrel,subint(1),errest)
846     !integraton of oscillating part
847     call int_wavg(yzz_ppwg_pq_p_osc,kz_low,-3,1,subint(2))
848
849     if(mod(p,2).eq.1) then !p and q odd
850         int_yzz_ppwg_pq_inf=subint(1)+subint(2)
851     else !r and s even
852         int_yzz_ppwg_pq_inf=subint(1)-subint(2)

```

```
853     end if
854
855 end function int_yzz_ppwg_pq_inf
856
857 !#####
858 !
859 ! Name:
860 !     yzz_ppwg_pq_p_osc
861 !
862 ! Revised:
863 !     February 29, 2000
864 !
865 ! Purpose:
866 !     Auxiliary function used by the function int_yzz_ppwg_pq_inf.
867 !     Evaluates the oscilating part of the kernel used to compute the
868 !     exterior self contribution to the fields in a longitudinal slot.
869 !
870 ! Usage:
871 !     yzz_ppwg_pq_p_osc(kz)
872 !
873 ! Arguments:
874 !     kz     - Independent variable (Input)
875 !
876 ! Modules:
877 !     const,quad_var,gen_var,ppwg_var,mom_var
878 !
879 function yzz_ppwg_pq_p_osc(kz)
880     !modules
881     use const, only : wp
882     use gen_var, only : k0_sq
883     use ppwg_var, only : kx_sq
884     use quad_var, only : kw
885     use mom_var, only : kp_sq,kq_sq
886     !implicit statement
887     implicit none
888     !declaration of intrinsic functions
889     intrinsic dsqrt,dcos
890     !list of calling arguments
891     real(wp)          :: yzz_ppwg_pq_p_osc
892     real(wp),intent(in) :: kz
893     !list of local variables
894     real(wp)          :: kz_sq
895
896     kz_sq=kz**2
897     yzz_ppwg_pq_p_osc=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)&
898         /dsqrt(kz_sq+kx_sq-k0_sq)*dcos(kz*kw)
899
900 end function yzz_ppwg_pq_p_osc
901
902 !#####
903 !
904 ! Name:
905 !     yzz_ppwg_pq_p_mono
906 !
907 ! Revised:
908 !     February 29, 2000
909 !
910 ! Purpose:
911 !     Auxiliary function used by the function int_yzz_ppwg_pq_inf.
```

```

912 !           Evaluates the monotonic part of the kernel used to compute the
913 !           exterior self contribution to the fields in a longitudinal slot.
914 !
915 ! Usage:
916 !           yzz_ppwg_pq_p_mono(kz)
917 !
918 ! Arguments:
919 !   kz      - Independent variable (Input)
920 !
921 ! Modules:
922 !           const,quad_var,gen_var,ppwg_var,mom_var
923 !
924 function yzz_ppwg_pq_p_mono(kz)
925   !modules
926   use const, only : wp
927   use gen_var, only : k0_sq
928   use ppwg_var, only : kx_sq
929   use quad_var, only : kw
930   use mom_var, only : kp_sq,kq_sq
931   !implicit statement
932   implicit none
933   !declaration of intrinsic functions
934   intrinsic dsqrt
935   !list of calling arguments
936   real(wp)          :: yzz_ppwg_pq_p_mono
937   real(wp),intent(in) :: kz
938   !list of local variables
939   real(wp)          :: kz_sq
940
941   kz_sq=kz**2
942   yzz_ppwg_pq_p_mono=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)&
943     /dsqrt(kz_sq+kx_sq-k0_sq)
944
945 end function yzz_ppwg_pq_p_mono
946
947
1  !#####
2  !
3  ! Name:
4  !       inv_r_int
5  !
6  ! Revised:
7  !       February 29, 2000
8  !
9  ! Purpose:
10 !       Sets up the integration of the function 1/R*(f(xi_0,zeta_0)+
11 !       xi*df_dxi(xi_0,zeta_0)+zeta*df_dzeta(xi_0,zeta_0)). The
12 !       integration is performed in polar coordinates over a rectangular
13 !       region. The intergration is seperated into the four quadrants
14 !       around the singular point specified.
15 !
16 ! Usage:
17 !       inv_r_int(x0,x1,z0,z1)
18 !
19 ! Arguments:
20 !   x0,z0  - lower bounds for the rectangular region (Input)
21 !
22 !   x1,z1  - upper bounds for the rectangular region (Input)
23 !

```

```

24  ! Modules:
25  !           const,quad_var
26  !
27  ! Subroutines:
28  !           inv_r_quad_int
29  !
30  function inv_r_int(x0,x1,z0,z1)
31  !modules
32  use const, only : wp
33  use quad_var, only : xi_0,zeta_0,xi_1,zeta_1
34  !implicit statement
35  implicit none
36  !declaration of intrinsic functions
37  intrinsic sum
38  !declaration of functions
39  real(wp)                :: inv_r_quad_int
40  !list of calling arguments
41  real(wp)                :: inv_r_int
42  real(wp)                :: x0,x1,z0,z1
43  !list of local variables
44  real(wp), dimension(1:4) :: value
45
46  !1st. quadrant
47  xi_1=x1-xi_0
48  zeta_1=z1-zeta_0
49  value(1)=inv_r_quad_int()
50  !2nd. quadrant
51  xi_1=x1-xi_0
52  zeta_1=z0-zeta_0
53  value(2)=inv_r_quad_int()
54  !3rd. quadrant
55  xi_1=x0-xi_0
56  zeta_1=z0-zeta_0
57  value(3)=inv_r_quad_int()
58  !4th. quadrant
59  xi_1=x0-xi_0
60  zeta_1=z1-zeta_0
61  value(4)=inv_r_quad_int()
62
63  !sum contribution from all four quadrants
64  inv_r_int=sum(value)
65
66  end function inv_r_int
67
68  !#####
69  !
70  ! Name:
71  !           inv_r_quad_int
72  !
73  ! Revised:
74  !           February 29, 2000
75  !
76  ! Purpose:
77  !           Integrates the function  $1/R*(f(xi_0,zeta_0)+$ 
78  !            $xi*df_dxi(xi_0,zeta_0)+zeta*df_dzeta(xi_0,zeta_0))$  over one
79  !           quadrant with the singular point located at the origin. The
80  !           integration is seperated into 3 regions.
81  !
82  ! Usage:

```

```

83      !           inv_r_quad_int()
84      !
85      ! Arguments:
86      !
87      ! Modules:
88      !           const,quad_var
89      !
90      ! IMSL subroutines:
91      !           dqdag
92      !
93  function inv_r_quad_int()
94      !modules
95      use const, only : wp
96      use quad_var, only : R1,R2,R3,xi_1,zeta_1,errrel,errabs
97      !implicit statement
98      implicit none
99      !declaration of intrinsic functions
100     intrinsic min,max,dabs,dsqrt,sum
101     !declaration of functions
102     real(wp),external      :: reg1_p,reg2_p,reg3_p
103     !list of calling arguments
104     real(wp)               :: inv_r_quad_int
105     !list of local parameters
106     integer, parameter    :: irule=2
107     !list of local variables
108     real(wp), dimension(1:3) :: value
109     real(wp), dimension(1:3) :: upper
110     real(wp), dimension(1:3) :: lower
111     real(wp)               :: errest
112
113     R1=min(dabs(xi_1),dabs(zeta_1))
114     R2=max(dabs(xi_1),dabs(zeta_1))
115     R3=dsqrt(xi_1**2+zeta_1**2)
116     upper=(/R1,R2,R3/)
117     lower=(/0.0_wp,R1,R2/)
118
119     !region 1
120     CALL DQDAG (reg1_p, lower(1), upper(1),&
121              ERRABS, ERRREL, IRULE, value(1),ERREST)
122     ! write(*,*) 'reg1 =',value(1),' reg1 errest = ',errest
123     !region 2
124     CALL DQDAG (reg2_p, lower(2), upper(2),&
125              ERRABS, ERRREL, IRULE, value(2),ERREST)
126     ! write(*,*) 'reg2 =',value(2),' reg2 errest = ',errest
127     !region 3
128     CALL DQDAG (reg3_p, lower(3), upper(3),&
129              ERRABS, ERRREL, IRULE, value(3),ERREST)
130     ! write(*,*) 'reg3 =',value(3),' reg3 errest = ',errest
131
132     !summation of contribution from all three regions
133     inv_r_quad_int=sum(value)
134
135  end function inv_r_quad_int
136
137  !#####
138  !
139  ! Name:
140  !           reg1_p
141  !

```

```
142 ! Revised:
143 !       February 29, 2000
144 !
145 ! Purpose:
146 !       Auxiliary function used by the function inv_r_quad_int.
147 !
148 ! Usage:
149 !       reg1_p(R)
150 !
151 ! Arguments:
152 !       R       - Independent variable, distance from the origin (Input)
153 !
154 ! Modules:
155 !       const,quad_var
156 !
157 function reg1_p(R)
158     !modules
159     use const, only : pi,wp
160     use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1
161     !implicit statement
162     implicit none
163     !declaration of intrinsic functions
164     intrinsic dsign
165     !list of calling arguments
166     real(wp)          :: reg1_p
167     real(wp),intent(in) :: R
168
169     reg1_p=f_0*pi/2.0_wp+df_dzeta*R*dsign(1.0_wp,zeta_1)&
170         +df_dxi*R*dsign(1.0_wp,xi_1)
171
172 end function reg1_p
173
174 !#####
175 !
176 ! Name:
177 !       reg2_p
178 !
179 ! Revised:
180 !       February 29, 2000
181 !
182 ! Purpose:
183 !       Auxiliary function used by the function inv_r_quad_int.
184 !
185 ! Usage:
186 !       reg2_p(R)
187 !
188 ! Arguments:
189 !       R       - Distance from the origin (Input)
190 !
191 ! Modules:
192 !       const,quad_var
193 !
194 function reg2_p(R)
195     !modules
196     use const, only : pi,wp
197     use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1,R1
198     !implicit statement
199     implicit none
200     !declaration of intrinsic functions
```



```

201     intrinsic dsin,dcos,dasin,dacos,dsign,dabs
202     !list of calling arguments
203     real(wp)          :: reg2_p
204     real(wp),intent(in) :: R
205
206     if(dabs(zeta_1).gt.dabs(xi_1)) then
207         reg2_p=f_0*dasin(R1/R)&
208             +df_dzeta*R*dsign(1.0_wp,zeta_1)*dsin(dasin(R1/R))&
209             -df_dxi*R*dsign(1.0_wp,xi_1)*(dcos(dasin(R1/R))-1.0_wp)
210     else
211         reg2_p=f_0*(pi/2.0_wp-dacos(R1/R))&
212             +df_dzeta*R*dsign(1.0_wp,zeta_1)*(1.0-dsin(dacos(R1/R)))&
213             +df_dxi*R*dsign(1.0_wp,xi_1)*dcos(dacos(R1/R))
214     endif
215
216 end function reg2_p
217
218 !#####
219 !
220 ! Name:
221 !     reg3_p
222 !
223 ! Revised:
224 !     February 29, 2000
225 !
226 ! Purpose:
227 !     Auxiliary function used by the function inv_r_quad_int.
228 !
229 ! Usage:
230 !     reg3_p(R)
231 !
232 ! Arguments:
233 !     R     - Distance from the origin (Input)
234 !
235 ! Modules:
236 !     const,quad_var
237 !
238 function reg3_p(R)
239     !modules
240     use const, only : pi,wp
241     use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1,R2,R1
242     !implicit statement
243     implicit none
244     !declaration of intrinsic functions
245     intrinsic dsin,dcos,dasin,dacos,dsign,dabs
246     !list of calling arguments
247     real(wp)          :: reg3_p
248     real(wp),intent(in) :: R
249
250     if(abs(zeta_1).gt.abs(xi_1)) then
251         reg3_p=f_0*(dasin(R1/R)-dacos(R2/R))+&
252             df_dzeta*R*dsign(1.0_wp,zeta_1)*&
253             (dsin(dasin(R1/R))-dsin(dacos(R2/R)))-&
254             df_dxi*R*dsign(1.0_wp,xi_1)*&
255             (dcos(dasin(R1/R))-dcos(dacos(R2/R)))
256     else
257         reg3_p=f_0*(dasin(R2/R)-dacos(R1/R))+&
258             df_dzeta*R*dsign(1.0_wp,zeta_1)*&
259             (dsin(dasin(R2/R))-dsin(dacos(R1/R)))-&

```

```

260         df_dxi*R*dsign(1.0_wp,xi_1)*&
261         (dcos(dasin(R2/R))-dcos(dacos(R1/R)))
262     endif
263
264 end function reg3_p
1  #####
2  !
3  ! Name:
4  !     int_wavg
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Integrate a function f(x)=g(x)*w(kw*x) using method of
11 !     weighted averages.
12 !
13 ! Usage:
14 !     call int_w_avg (f, a, alfa,w_type,result)
15 !
16 ! Arguments:
17 !     f - Asymtotic part user-supplied function to be integrated. The
18 !     form is:
19 !     f(x), where
20 !     x - Independent variable. (Input)
21 !     f - The function value. (Output)
22 !     f must be declared external in the calling program.
23 !     a - Lower limit of integration. (Input)
24 !     alfa - asymtotic exponet of function f(x). (Input)
25 !     w_type - Type of weight function w used. (Input)
26 !
27 !         w_type      weight
28 !         1           cos(kw*x)
29 !         2           sin(kw*x)
30 !
31 !     result - Estimate of the integral from a to infinity of g*w. (Output)
32 !
33 ! Modules:
34 !     const,quad_var
35 !
36 ! IMSL subroutines:
37 !     dqdag
38 !
39 subroutine int_wavg(f,a,alfa,w_type,result)
40 !modules
41 use const, only : pi,cj,wp
42 use quad_var, only : kw,mm,errrel,errabs
43 !implicit statement
44 implicit none
45 !declaration of intrinsic functions
46 intrinsic sin,cos,sqrt
47 !declaration of functions
48 real, external :: f
49 !list of calling arguments
50 integer,intent(in) :: w_type,alfa
51 real(wp),intent(in) :: a
52 real(wp),intent(out):: result
53 !list of local parameters
54 real(wp),allocatable,dimension(:,:) :: I_ml,w_ml
55 real(wp),allocatable,dimension(:) :: lambda_m
56 integer, parameter :: irule=2

```

```

55     integer           :: m,m0,l
56     integer           :: status
57     real(wp)          :: errest
58
59     allocate(I_ml(mm,mm),w_ml(mm,mm),lambda_m(mm), STAT=status)
60
61     !compute successive zeroes of osc. function superior to lower
62     !limit a, and I_ml(m,1)
63     if(w_type.eq.1) then !w=cos
64         m0=int((a+pi/(2.0_wp*kw))*kw/pi)
65     else !w=sin
66         m0=int(a*kw/pi)
67     endif
68     do m=1,mm
69         if(w_type.eq.1) then !w=cos
70             lambda_m(m)=(m0+m-0.5_wp)*pi/kw
71         else !w=sin
72             lambda_m(m)=(m0+m)*pi/kw
73         endif
74         call dqdag(f,a,lambda_m(m),errabs,errrel,irule,I_ml(m,1),errest)
75     enddo
76     !compute weights
77     do l=1,mm-1
78         do m=1,mm-1+l
79             w_ml(m,l)=(lambda_m(1)/lambda_m(m))**(alfa+1-l)
80         enddo
81     enddo
82     !compute partial values
83     do l=1,mm-1
84         do m=1,mm-1
85             I_ml(m,l+1)=(w_ml(m,l)*I_ml(m,l)+w_ml(m+1,l)*I_ml(m+1,l))/&
86                 (w_ml(m,l)+w_ml(m+1,l))
87         enddo
88     enddo
89
90     result=I_ml(1,mm)
91
92     deallocate(I_ml,w_ml,lambda_m, STAT=status)
93     return
94     end

```

```

1  !#####
2  !
3  ! Name:
4  !     Y_fill
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Computes the matrix terms filling the lefthand side of the
11 !     MoM matrix system.
12 !
13 ! Usage:
14 !     Y_fill()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 !     const,gen_var,slot_var,mom_var

```

```

20      !
21      ! Subroutines:
22      !       y_ext,yzx_ext_ps,yxz_ext_rq,yzz_int_pq,yxx_int_rs,
23      !       yzx_int_ps,yxz_int_rq,yxx_ppwg_rs,yzz_ppwg_pq,
24      !       yzx_ppwg_ps,yxz_ppwg_rq
25      !
26      subroutine Y_fill()
27      !modules
28      use const, only : wp
29      use mom_var, only : r,s,p,q,Nz,Nx,Y
30      use slot_var, only : Lz,Lx
31      use gen_var, only : ext_type
32      !implicit statement
33      implicit none
34      !declaration of functions
35      complex(wp) :: y_ext,yzx_ext_ps,yxz_ext_rq
36      complex(wp) :: yzz_int_pq,yxx_int_rs,yzx_int_ps,yxz_int_rq
37      complex(wp) :: yxx_ppwg_rs,yzz_ppwg_pq,yzx_ppwg_ps,yxz_ppwg_rq
38
39      !self terms for longitudinal part
40      !write(*,*) 'Y(q,p) - self term for longi. slot'
41      do q=1,Nz
42         do p=1,q-1
43            Y(q,p)=Y(p,q)
44         enddo
45         do p=q,Nz
46            if(ext_type.eq.1) then
47               Y(q,p)=y_ext(p,q,Lz)-yzz_int_pq()
48            else
49               Y(q,p)=yzz_ppwg_pq()-yzz_int_pq()
50            endif
51            !write(*,*) q,p,Y(q,p)
52         enddo
53      enddo
54
55      !self terms for transverse part
56      !write(*,*) 'Y(s,r) - self term for transv. slot'
57      do s=1,Nx
58         do r=1,s-1
59            Y(s+Nz,r+Nz)=Y(r+Nz,s+Nz)
60         enddo
61         do r=s,Nx
62            if(ext_type.eq.1) then
63               Y(s+Nz,r+Nz)=-y_ext(r,s,Lx)-yxx_int_rs()
64            else
65               Y(s+Nz,r+Nz)=yxx_ppwg_rs()-yxx_int_rs()
66            endif
67            !write(*,*) s,r,Y(s+Nz,r+Nz)
68         enddo
69      enddo
70
71      !cross terms in longitudinal part
72      !write(*,*) 'Y(q,r) - cross term for longi. slot'
73      do q=1,Nz
74         do r=1,Nx
75            if(ext_type.eq.1) then
76               Y(q,r+Nz)=yxz_ext_rq()-yxz_int_rq()
77            else
78               Y(q,r+Nz)=yxz_ppwg_rq()-yxz_int_rq()

```

```

79         endif
80         !write(*,*) q,r,Y(q,r+Nz)
81     enddo
82 enddo
83
84 !cross terms in transverse part
85 !write(*,*) 'Y(s,p) - cross term for transv. slot'
86 do s=1,Nx
87     do p=1,Nz
88         if(ext_type.eq.1) then
89             Y(s+Nz,p)=yzx_ext_ps()-yzx_int_ps()
90         else
91             Y(s+Nz,p)=yzx_ppwg_ps()-yzx_int_ps()
92         endif
93         !write(*,*) s,p,Y(s+Nz,p)
94     enddo
95 enddo
96 return
97 end
98
99 !#####
100 !
101 ! Name:
102 !     Y_fill_fnt_thk_long
103 !
104 ! Revised:
105 !     February 29, 2000
106 !
107 ! Purpose:
108 !     Computes the matrix terms filling the lefthand side of the
109 !     MoM matrix system. To be used for investigation of a longitudinal
110 !     slot having finite wall thickness.
111 !
112 ! Usage:
113 !     Y_fill_fnt_thk_long()
114 !
115 ! Arguments:
116 !
117 ! Modules:
118 !     const,gen_var,slot_var,mom_var
119 !
120 ! Subroutines:
121 !     y_ext,yzz_int_pq,yzz_ppwg_pq,y11,y12
122 !
123 subroutine Y_fill_fnt_thk_long()
124 !modules
125 use const, only : wp
126 use mom_var, only : p,q,Nz,Y
127 use slot_var, only : Lz
128 use gen_var, only : ext_type
129 !implicit statement
130 implicit none
131 !declaration of functions
132 complex(wp) :: y_ext
133 complex(wp) :: yzz_int_pq
134 complex(wp) :: y11,y12
135 complex(wp) :: yzz_ppwg_pq
136
137 do p=1,Nz

```

```

138         do q=1,p-1
139             Y(p,q)=Y(q,p)
140         enddo
141         do q=p,Nz
142             Y(p,q)=y11(p,q,Lz)-yzz_int_pq()
143         enddo
144     enddo
145
146     do p=1,Nz
147         do q=1,Nz
148             Y(p,q+Nz)=-y12(p,q,Lz)
149         enddo
150     enddo
151
152     do p=1,Nz
153         do q=1,Nz
154             Y(p+Nz,q)=-y12(p,q,Lz)
155         enddo
156     enddo
157
158     do p=1,Nz
159         do q=1,p-1
160             Y(p+Nz,q+Nz)=Y(q+Nz,p+Nz)
161         enddo
162         do q=p,Nz
163             if(ext_type.eq.1) then
164                 Y(p+Nz,q+Nz)=y_ext(p,q,Lz)+y11(p,q,Lz)
165             else
166                 Y(p+Nz,q+Nz)=yzz_ppwg_pq()+y11(p,q,Lz)
167             endif
168         enddo
169     enddo
170
171     return
172     end
173
174     !#####
175     !
176     ! Name:
177     !       Y_fill_fnt_thk_trans
178     !
179     ! Revised:
180     !       February 29, 2000
181     !
182     ! Purpose:
183     !       Computes the matrix terms filling the lefthand side of the
184     !       MoM matrix system. To be used for investigation of a transverse
185     !       slot having finite wall thickness.
186     !
187     ! Usage:
188     !       Y_fill_fnt_thk_trans()
189     !
190     ! Arguments:
191     !
192     ! Modules:
193     !       const,gen_var,slot_var,mom_var
194     !
195     ! Subroutines:
196     !       y_ext,yxx_int_rs,yxx_ppwg_rs,y11,y12

```

```

197  !
198  subroutine Y_fill_fnt_thk_trans()
199      !modules
200      use const, only : wp
201      use mom_var, only : r,s,Nx,Y
202      use slot_var, only : Lx
203      use gen_var, only : ext_type
204      !implicit statement
205      implicit none
206      !declaration of functions
207      complex(wp) :: y_ext
208      complex(wp) :: yxx_int_rs
209      complex(wp) :: y11,y12
210      complex(wp) :: yxx_ppwg_rs
211
212      do r=1,Nx
213          do s=1,r-1
214              Y(r,s)=Y(s,r)
215          enddo
216          do s=r,Nx
217              Y(r,s)=-y11(r,s,Lx)-yxx_int_rs()
218          enddo
219      enddo
220      do r=1,Nx
221          do s=1,Nx
222              Y(r,s+Nx)=y12(r,s,Lx)
223          enddo
224      enddo
225      do r=1,Nx
226          do s=1,Nx
227              Y(r+Nx,s)=y12(r,s,Lx)
228          enddo
229      enddo
230      do r=1,Nx
231          do s=1,r-1
232              Y(r+Nx,s+Nx)=Y(s+Nx,r+Nx)
233          enddo
234          do s=r,Nx
235              if(ext_type.eq.1) then
236                  Y(r+Nx,s+Nx)=-y_ext(r,s,Lx)-y11(r,s,Lx)
237              else
238                  Y(r+Nx,s+Nx)=yxx_ppwg_rs()-y11(r,s,Lx)
239              endif
240          enddo
241      enddo
242
243      return
244      end
245
246      !#####
247      !
248      ! Name:
249      !     y11
250      !
251      ! Revised:
252      !     February 29, 2000
253      !
254      ! Purpose:
255      !     Used in the investigation of a transverse or longitudinal slot

```

```

256 !           of any length having finite wall thickness. Computes the self
257 !           contribution to the fields at the two interfaces.
258 !
259 ! Usage:
260 !       y11(p,q,L)
261 !
262 ! Arguments:
263 !     p   - order of expansion function (Input)
264 !     q   - order of weighting function (Input)
265 !     L   - length of slot (Input)
266 !
267 ! Modules:
268 !       const,slot_var,mom_var,gen_var
269 !
270 function y11(p,q,L)
271 !modules
272 use const, only : pi,cj,Z0,mu_0,wp
273 use gen_var, only : k0_sq,omega
274 use wg_var, only : t
275 !implicit statement
276 implicit none
277 !declaration of intrinsic functions
278 intrinsic dsqrt, dtan, dtanh
279 !list of calling arguments
280 complex(wp)      :: y11
281 integer ,intent(in) :: p,q
282 real(wp),intent(in) :: L
283 !list of local variables
284 real(wp)      :: beta,beta_sq
285
286 if (p.eq.q) then
287     beta_sq=k0_sq-(p*pi/(2.0_wp*L))**2
288     if (beta_sq.gt.0.0_wp) then
289         beta=dsqrt(beta_sq)
290         y11=cj*beta*L/(mu_0*omega)/dtan(beta*t)*Z0
291     else
292         beta=dsqrt(-beta_sq)
293         y11=cj*beta*L/(mu_0*omega)/dtanh(beta*t)*Z0
294     end if
295 else
296     y11=(0.0_wp,0.0_wp)
297 end if
298
299 end function y11
300
301 !#####
302 !
303 ! Name:
304 !     y12
305 !
306 ! Revised:
307 !     February 29, 2000
308 !
309 ! Purpose:
310 !     Used in the investigation of a transverse or longitudinal slot
311 !     of any length having finite wall thickness. Computes the cross
312 !     contribution to the fields at the two interfaces.
313 !
314 ! Usage:

```



```

315 !           y12(p,q,L)
316 !
317 ! Arguments:
318 !   p       - order of expansion function (Input)
319 !   q       - order of weighting function (Input)
320 !   L       - length of slot (Input)
321 !
322 ! Modules:
323 !           const,slot_var,mom_var,gen_var
324 !
325 function y12(p,q,L)
326     !modules
327     use const, only : pi,cj,Z0,mu_0,wp
328     use gen_var, only : k0_sq,omega
329     use wg_var, only : t
330     !implicit statement
331     implicit none
332     !declaration of intrinsic functions
333     intrinsic dsqrt, dsin, dsinh
334     !list of calling arguments
335     complex(wp)      :: y12
336     integer ,intent(in) :: p,q
337     real(wp),intent(in) :: L
338     !list of local variables
339     real(wp)      :: beta,beta_sq
340
341     if (p.eq.q) then
342         beta_sq=k0_sq-(p*pi/(2.0_wp*L))**2
343         if (beta_sq.gt.0.0_wp) then
344             beta=dsqrt(beta_sq)
345             y12=cj*beta*L/(mu_0*omega)/dsin(beta*t)*Z0
346         else
347             beta=dsqrt(-beta_sq)
348             y12=cj*beta*L/(mu_0*omega)/dsinh(beta*t)*Z0
349         end if
350     else
351         y12=(0.0_wp,0.0_wp)
352     end if
353
354 end function y12

```

```

1  !#####
2  !
3  ! Name:
4  !     hinc_fill_fnt_thk_long
5  !
6  ! Revised:
7  !     February 29, 2000
8  !
9  ! Purpose:
10 !     Computes the excitations terms filling the righthand side of the
11 !     MoM matrix system.
12 !
13 ! Usage:
14 !     hinc_fill_fnt_thk_long()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 !     const,mom_var

```

```
20      !
21      ! Subroutines:
22      !       hz_inc_q
23      !
24      ! Remarks:
25      !       This routine is used investigating a longitudinal slot including
26      !       finite wall thickness.
27      !
28      subroutine hinc_fill_fnt_thk_long()
29      !modules
30      use const, only : wp
31      use mom_var, only : q,Nz,h_inc
32      !implicit statement
33      implicit none
34      !declaration of functions
35      complex(wp) :: hz_inc_q
36
37      do q=1,Nz
38         h_inc(q)=hz_inc_q()
39      enddo
40      do q=Nz+1,2*Nz
41         h_inc(q)=(0.0_wp,0.0_wp)
42      enddo
43
44      return
45      end
46
47      !#####
48      !
49      ! Name:
50      !       hinc_fill_fnt_thk_trans
51      !
52      ! Revised:
53      !       February 29, 2000
54      !
55      ! Purpose:
56      !       Computes the excitations terms filling the righthand side of the
57      !       MoM matrix system.
58      !
59      ! Usage:
60      !       hinc_fill_fnt_thk_trans()
61      !
62      ! Arguments:
63      !
64      ! Modules:
65      !       const,mom_var
66      !
67      ! Subroutines:
68      !       hx_inc_s
69      !
70      ! Remarks:
71      !       This routine is used investigating a transverse slot including
72      !       finite wall thickness.
73      !
74      subroutine hinc_fill_fnt_thk_trans()
75      !modules
76      use const, only : wp
77      use mom_var, only : s,Nx,h_inc
78      !implicit statement
```

```

79     implicit none
80     !declaration of functions
81     complex(wp) :: hx_inc_s
82
83     do s=1,Nx
84         h_inc(s)=hx_inc_s()
85     enddo
86     do s=Nx+1,2*Nx
87         h_inc(s)=(0.0_wp,0.0_wp)
88     enddo
89
90     return
91     end
92
93     !#####
94     !
95     ! Name:
96     !       hinc_fill
97     !
98     ! Revised:
99     !       February 29, 2000
100    !
101    ! Purpose:
102    !       Computes the excitations terms filling the righthand side of the
103    !       MoM matrix system.
104    !
105    ! Usage:
106    !       hinc_fill()
107    !
108    ! Arguments:
109    !
110    ! Modules:
111    !       const,mom_var
112    !
113    ! Subroutines:
114    !       hz_inc_q,hx_inc_s
115    !
116    subroutine hinc_fill()
117        !modules
118        use const, only : wp
119        use mom_var, only : q,s,Nz,Nx,h_inc
120        !implicit statement
121        implicit none
122        !declaration of functions
123        complex(wp) :: hz_inc_q, hx_inc_s
124
125        do q=1,Nz
126            h_inc(q)=hz_inc_q()
127        enddo
128
129        do s=1,Nx
130            h_inc(Nz+s)=hx_inc_s()
131        enddo
132
133    return
134    end
135
136    !#####
137    !

```

```

138  ! Name:
139  !       hz_inc_q
140  !
141  ! Revised:
142  !       February 29, 2000
143  !
144  ! Purpose:
145  !       Computes the q'th excitations terms for a longitudinal slot.
146  !
147  ! Usage:
148  !       hz_inc_q()
149  !
150  ! Arguments:
151  !
152  ! Modules:
153  !       const,mom_var,slot_var,gen_var,wg_var
154  !
155  function hz_inc_q()
156  !modules
157  use const, only : pi,cj,wp
158  use slot_var, only : Lz,x0
159  use mom_var, only : q
160  use gen_var, only : k0
161  use wg_var, only : a,beta_10
162  !implicit statement
163  implicit none
164  !declaration of intrinsic functions
165  intrinsic dcos,dsin
166  !list of calling arguments
167  complex(wp) :: hz_inc_q
168
169  hz_inc_q=pi*q/Lz*dcos(pi*x0/a)/(beta_10**2-(q*pi/(2.0_wp*Lz))**2)
170
171  if(mod(q,2).eq.1) then
172  hz_inc_q=-cj*hz_inc_q*dcos(beta_10*Lz)
173  else
174  hz_inc_q=hz_inc_q*dsin(beta_10*Lz)
175  end if
176
177  end function hz_inc_q
178
179  !#####
180  !
181  ! Name:
182  !       hx_inc_s
183  !
184  ! Revised:
185  !       February 29, 2000
186  !
187  ! Purpose:
188  !       Computes the s'th excitations terms for a transverse slot.
189  !
190  ! Usage:
191  !       hx_inc_s()
192  !
193  ! Arguments:
194  !
195  ! Modules:
196  !       const,mom_var,slot_var,gen_var,wg_var

```

```

197 !
198 function hx_inc_s()
199     !modules
200     use const, only : pi,wp
201     use mom_var, only : s
202     use slot_var, only : Lx,x00
203     use wg_var, only : a,beta_10
204     !implicit statement
205     implicit none
206     !declaration of intrinsic functions
207     intrinsic dcos,dsin
208     !list of calling arguments
209     complex(wp) :: hx_inc_s
210
211     hx_inc_s=-beta_10*a*s/Lx/&
212             ((s*pi/(2.0_wp*Lx))**2-(pi/a)**2)
213
214     if(mod(s,2).eq.1) then !s odd
215         hx_inc_s=hx_inc_s*dcos(pi*Lx/a)*dsin(pi*x00/a)
216     else !s even
217         hx_inc_s=-hx_inc_s*dcos(pi*x00/a)*dsin(pi*Lx/a)
218     end if
219
220 end function hx_inc_s

```

1 !#####
2 !
3 ! Name:
4 ! backscat
5 !
6 ! Revised:
7 ! February 29, 2000
8 !
9 ! Purpose:
10 ! Computes the complex refelction coefficient B_10/A_10=s11 for
11 ! both longitudinal, transvers and t-slots
12 !
13 ! Usage:
14 ! backscat()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 ! const
20 !
21 ! Subroutines:
22 ! backscat_trans,backscat_long
23 !
24 function backscat()
25 !modules
26 use const, only : wp
27 !implicit statement
28 implicit none
29 !declaration of functions
30 complex(wp) :: backscat_trans,backscat_long
31 !list of calling arguments
32 complex(wp) :: backscat
33
34 backscat=backscat_trans()+backscat_long()
35

```

36 end function backscat
37
38 !#####
39 !
40 ! Name:
41 !     backscat_long
42 !
43 ! Revised:
44 !     February 29, 2000
45 !
46 ! Purpose:
47 !     computes the reflection coefficient B_10/A_10=s11 for a
48 !     longitudinal slot
49 !
50 ! Usage:
51 !     backscat_long()
52 !
53 ! Arguments:
54 !
55 ! Modules:
56 !     const,slot_var,mom_var,gen_var,wg_var
57 !
58 ! Subroutines:
59 !
60 function backscat_long()
61 !modules
62 use const, only :mu_0,pi,cj,Z0,wp
63 use slot_var, only :w,Lz,x0
64 use mom_var, only :p,Nz,E
65 use gen_var, only : k0,omega
66 use wg_var, only : a,b,beta_10
67 !implicit statement
68 implicit none
69 !declaration of intrinsic functions
70 intrinsic dsin,dcos
71 !list of calling arguments
72 complex(wp) :: backscat_long
73 !list of local variables
74 complex(wp) :: sum, add_term
75
76 if(Nz.gt.0) then
77     sum=(0.0_wp,0.0_wp)
78     do p=1,Nz
79         add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-beta_10**2)
80         if(mod(p,2).eq.1) then !p odd
81             add_term=add_term*dcos(beta_10*Lz)
82         else !p even
83             add_term=add_term*cj*dsin(beta_10*Lz)
84         end if
85         sum=sum+add_term
86     enddo
87     backscat_long=-cj*2.0_wp*pi**2*Z0/(omega*mu_0*a**2*b*Lz*beta_10)
88     backscat_long=backscat_long*dsin(w*pi/(2.0*a))*dcos(x0*pi/a)*sum
89 else
90     backscat_long=(0.0_wp,0.0_wp)
91 endif
92
93 end function backscat_long
94

```

```

95  !#####
96  !
97  ! Name:
98  !     backscat_trans
99  !
100 ! Revised:
101 !     February 29, 2000
102 !
103 ! Purpose:
104 !     computes the reflection coefficient B_10/A_10=s11 for a
105 !     transverse slot
106 !
107 ! Usage:
108 !     backscat_trans()
109 !
110 ! Arguments:
111 !
112 ! Modules:
113 !     const,slot_var,mom_var,gen_var,wg_var
114 !
115 ! Subroutines:
116 !
117 function backscat_trans()
118     !modules
119     use const, only : mu_0,pi,cj,Z0,wp
120     use slot_var, only : w,Lx,x0,x00
121     use mom_var, only : r,Nx,Nz,E
122     use gen_var, only : k0,omega
123     use wg_var, only : a,b,beta_10
124     !implicit statement
125     implicit none
126     !declaration of intrinsic functions
127     intrinsic dsin,dcos
128     !list of calling arguments
129     complex(wp) :: backscat_trans
130     !list of local variables
131     complex(wp) :: sum, add_term
132
133     if(Nx.gt.0) then
134         sum=(0.0_wp,0.0_wp)
135         do r=1,Nx
136             add_term=E(r+Nz)*r/((r*pi/(2.0_wp*Lx))**2-(pi/a)**2)
137             if(mod(r,2).eq.1) then !r odd
138                 add_term=add_term*dcos(pi*Lx/a)*dsin(pi*x00/a)
139             else !r even
140                 add_term=-add_term*dcos(pi*x00/a)*dsin(pi*Lx/a)
141             end if
142             sum=sum+add_term
143         enddo
144         backscat_trans=-2.0_wp*pi**2/(omega*mu_0*a**2*b*Lx*beta_10)
145         backscat_trans=backscat_trans*Z0*dsin(w/2.0_wp*beta_10)*sum
146     else
147         backscat_trans=(0.0_wp,0.0_wp)
148     end if
149
150 end function backscat_trans

```

```

1
2  !#####
3  !

```

```
4      ! Name:
5      !       forwardscat
6      !
7      ! Revised:
8      !       February 29, 2000
9      !
10     ! Purpose:
11     !       Computes the complex transmission coefficient C_10/A_10 for both
12     !       longitudinal, transvers and t-slots
13     !
14     ! Usage:
15     !       forwardscat()
16     !
17     ! Arguments:
18     !       forwardscat - Function value. (Output)
19     !
20     ! Modules:
21     !       const
22     !
23     ! Subroutines:
24     !       forwardscat_trans,forwardscat_long
25     !
26     function forwardscat()
27     !modules
28     use const, only : wp
29     !implicit statement
30     implicit none
31     !declaration of functions
32     complex(wp) :: forwardscat_trans,forwardscat_long
33     !list of calling arguments
34     complex(wp) :: forwardscat
35
36     forwardscat=forwardscat_trans()+forwardscat_long()
37
38     end function forwardscat
39
40     !#####
41     !
42     ! Name:
43     !       forwardscat_long
44     !
45     ! Revised:
46     !       February 29, 2000
47     !
48     ! Purpose:
49     !       computes the complex transmission coefficient C_10/A_10 for a
50     !       longitudinal slot
51     !
52     ! Usage:
53     !       forwardscat_long()
54     !
55     ! Arguments:
56     !       forwardscat_long - Function value. (Output)
57     !
58     ! Modules:
59     !       const,slot_var,mom_var,gen_var,wg_var
60     !
61     ! Subroutines:
62     !
```



```

63  function forwardscat_long()
64      !modules
65      use const, only :mu_0,pi,cj,Z0,wp
66      use slot_var, only :w,Lz,x0
67      use mom_var, only :p,Nz,E
68      use gen_var, only : k0,omega
69      use wg_var, only : a,b,beta_10
70      !implicit statement
71      implicit none
72      !declaration of intrinsic functions
73      intrinsic dsin,dcos,mod
74      !list of calling arguments
75      complex(wp) :: forwardscat_long
76      !list of local variables
77      complex(wp)      :: sum, add_term
78
79      if(Nz.gt.0) then
80          sum=(0.0_wp,0.0_wp)
81          do p=1,Nz
82              add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-beta_10**2)
83              if(mod(p,2).eq.1) then !p odd
84                  add_term=add_term*dcos(beta_10*Lz)
85              else !p even
86                  add_term=-add_term*cj*dsin(beta_10*Lz)
87              end if
88              sum=sum+add_term
89          enddo
90          forwardscat_long=2.0_wp*pi**2*Z0/&
91              (cj*omega*mu_0*a**2*b*Lz*beta_10)
92          forwardscat_long=forwardscat_long*dsin(w*pi/(2.0_wp*a))*&
93              dcos(x0*pi/a)*sum
94      else
95          forwardscat_long=(0.0_wp,0.0_wp)
96      endif
97
98  end function forwardscat_long
99
100  !#####
101  !
102  ! Name:
103  !     forwardscat_trans
104  !
105  ! Revised:
106  !     February 29, 2000
107  !
108  ! Purpose:
109  !     computes the complex transmission coefficient C_10/A_10 for a
110  !     transverse slot
111  !
112  ! Usage:
113  !     forwardscat_trans()
114  !
115  ! Arguments:
116  !     forwardscat_trans - Function value. (Output)
117  !
118  ! Modules:
119  !     const,slot_var,mom_var,gen_var,wg_var
120  !
121  ! Subroutines:

```

```

122  !
123  function forwardscat_trans()
124      !modules
125      use const, only : mu_0,pi,cj,Z0,wp
126      use slot_var, only : w,Lx,x0,x00
127      use mom_var, only : r,Nx,Nz,E
128      use gen_var, only : k0,omega
129      use wg_var, only : a,b,beta_10
130      !implicit statement
131      implicit none
132      !declaration of intrinsic functions
133      intrinsic dsin,dcos,mod
134      !list of calling arguments
135      complex(wp) :: forwardscat_trans
136      !list of local variables
137      complex(wp) :: sum, add_term
138
139      if(Nx.gt.0) then
140          sum=(0.0_wp,0.0_wp)
141          do r=1,Nx
142              add_term=E(r+Nz)*r/((r*pi/(2.0_wp*Lx))**2-(pi/a)**2)
143              if(mod(r,2).eq.1) then !r odd
144                  add_term=add_term*dcos(pi*Lx/a)*dsin(pi*x00/a)
145              else !r even
146                  add_term=-add_term*dcos(pi*x00/a)*dsin(pi*Lx/a)
147              end if
148              sum=sum+add_term
149          enddo
150          forwardscat_trans=2.0_wp*pi**2/(omega*mu_0*a**2*b*Lx*beta_10)
151          forwardscat_trans=forwardscat_trans*Z0*dsin(w/2.0_wp*beta_10)*sum
152      else
153          forwardscat_trans=(0.0_wp,0.0_wp)
154      end if
155
156  end function forwardscat_trans

```

```

1
2  !#####
3  !
4  ! Name:
5  !     write_exp_coeff
6  !
7  ! Revised:
8  !     February 29, 2000
9  !
10 ! Purpose:
11 !     This subroutine writes the expansion coefficients in the matrix E
12 !
13 ! Usage:
14 !     call write_exp_coeff()
15 !
16 ! Arguments:
17 !
18 ! Modules:
19 !     const,mom_var
20 !
21 ! Subroutines:
22 !
23 ! Remarks:
24 !     The output is printed to standard output.

```

```

25      !
26      subroutine write_exp_coeff()
27          !modules
28          use const, only : wp
29          use mom_var, only : Nz,Nx,E
30          !implicit statement
31          implicit none
32          !declaration of intrinsic functions
33          intrinsic dreal,dimag,cdabs,dlog10
34          !list of local variables
35          integer i
36
37          do i=1,Nz
38              write(*,1077) i,dreal(E(i)),dimag(E(i)),&
39                  20.0_wp*dlog10(cdabs(E(i)))
40          enddo
41          do i=1,Nx
42              write(*,1077) i, dreal(E(Nz+i)), dimag(E(Nz+i)),&
43                  20.0*dlog10(cdabs(E(Nz+i)))
44          enddo
45
46          1077 format(2x,I15,3(2x,F15.6))
47          return
48      end
49
50      !#####
51      !
52      ! Name:
53      !       F_sph_fs
54      !
55      ! Revised:
56      !       February 29, 2000
57      !
58      ! Purpose:
59      !       This subroutine computes the electric vector potential F in
60      !       spherical coordinates for a slot radiating into half space
61      !
62      ! Usage:
63      !       call F_sph_fs()
64      !
65      ! Arguments:
66      !
67      ! Modules:
68      !       const,gen_var,radfield_var,slot_var,mom_var
69      !
70      ! Subroutines:
71      !       xyz2sph,dsinc
72      !
73      subroutine F_sph_fs()
74          !modules
75          use const, only : pi,cj,wp
76          use gen_var, only : k0,k0_sq
77          use slot_var, only : w,Lx,Lz
78          use mom_var, only : p,r,Nz,Nx,E
79          use radfield_var, only : F_sph,F_xyz,sph
80          !implicit statement
81          implicit none
82          !declaration of intrinsic functions
83          intrinsic dcos,dsin,cdexp

```

```

84      !declaration of functions
85      real(wp)      :: dsinc
86      !list of local variables
87      real(wp)      :: csph,csth,snth
88      complex(wp)  :: sum, add_term
89
90      !init. of aux. variables
91      csph=dcos(sph(3))
92      csth=dcos(sph(2))
93      snth=dsin(sph(2))
94      F_xyz=(/(0.0_wp,0.0_wp),(0.0_wp,0.0_wp),(0.0_wp,0.0_wp)/)
95
96      !computation of F_x
97      if(Nx.gt.0) then
98          sum=(0.0_wp,0.0_wp)
99          do r=1,Nx
100             add_term=E(r+Nz)*r/&
101                 ((r*pi/(2.0_wp*Lx))**2-k0_sq*csph**2*snth**2)
102             if(mod(r,2).eq.1) then !r odd
103                 add_term=add_term*dcos(k0*Lx*csph*snth)
104             else !r even
105                 add_term=-add_term*cj*dsin(k0*Lx*csph*snth)
106             end if
107             sum=sum+add_term
108          enddo
109          F_xyz(1)=-w/2.0_wp*cdexp(-cj*k0*sph(1))/(Lx*sph(1))*&
110              cdexp(cj*k0*(w/2.0_wp-Lx)*csph*snth)*&
111              dsinc(w*k0/2.0_wp*csth)*sum
112      end if
113
114      !computation of F_z
115      if(Nz.gt.0) then
116          sum=(0.0_wp,0.0_wp)
117          do p=1,Nz
118             add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-k0_sq*csth**2)
119             if(mod(p,2).eq.1) then !p odd
120                 add_term=add_term*dcos(k0*Lz*csth)
121             else !p even
122                 add_term=-add_term*cj*dsin(k0*Lz*csth)
123             end if
124             sum=sum+add_term
125          enddo
126          F_xyz(3)=w/2.0_wp*cdexp(-cj*k0*sph(1))/(Lz*sph(1))*&
127              dsinc(w*k0/2.0_wp*csph*snth)*sum
128
129      endif
130
131      !transform into spherical coor.
132      call xyz2sph(F_xyz,sph,F_sph)
133
134      return
135      end
136
137      !#####
138      !
139      ! Name:
140      !           F_sph_ppwg
141      !
142      ! Revised:

```

```

143 !           February 29, 2000
144 !
145 ! Purpose:
146 !           This subroutine computes the electric vector potential F in
147 !           spherical coordinates for a slot radiating into half space
148 !
149 ! Usage:
150 !           call F_sph_fs()
151 !
152 ! Arguments:
153 !
154 ! Modules:
155 !           const,gen_var,radfield_var,slot_var,mom_var
156 !
157 ! Subroutines:
158 !           xyz2sph,dsinc
159 !
160 subroutine F_sph_ppwg()
161 !modules
162 use const, only : pi,cj,wp
163 use gen_var, only : k0,k0_sq
164 use slot_var, only : w,Lx,Lz,x00_pp,x0_pp
165 use mom_var, only : p,r,Nz,Nx,E
166 use ppwg_var, only : m,a1,b1,kx,kx_sq,ky
167 use radfield_var, only : F_sph,F_xyz,sph
168 !implicit statement
169 implicit none
170 !declaration of intrinsic functions
171 intrinsic dcos,dsin,cdexp
172 !declaration of functions
173 real(wp)      :: dsinc
174 !list of local variables
175 real(wp)      :: csph,csth,snth,kz,kz_sq,epsilon_m
176 complex(wp)  :: sum, add_term
177
178 !init. of aux. variables
179 csph=dcos(sph(3))
180 csth=dcos(sph(2))
181 snth=dsin(sph(2))
182 F_xyz=(/(0.0_wp,0.0_wp),(0.0_wp,0.0_wp),(0.0_wp,0.0_wp)/)
183
184 m=1
185 epsilon_m=1.0_wp
186 kx=m*pi/a1
187 kx_sq=kx**2
188 kz=-k0*csth
189 kz_sq=kz**2
190 ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
191 if (dimag(ky).gt.0.0_wp) then
192   ky=-ky
193 endif
194
195 !computation of F_x
196 if(Nx.gt.0) then
197   sum=(0.0_wp,0.0_wp)
198   do r=1,Nx
199     add_term=E(r+Nz)*r/&
200               ((r*pi/(2.0_wp*Lx))**2-kx_sq)
201     if(mod(r,2).eq.1) then !r odd

```

```

202         add_term=add_term*dcos(kx*Lx)*dsin(kx*x00_pp)
203     else      !r even
204         add_term=-add_term*dcos(kx*x00_pp)*dsin(kx*Lx)
205     end if
206     sum=sum+add_term
207 enddo
208 F_xyz(1)=(-w*2.0_wp/(Lx*a1))*&
209     epsilon_m*dsinc(w*kz/2.0_wp)*&
210     cdexp(-cj*ky*b1)*&
211     kx/(k0_sq*cspth**2*snth**2-kx_sq)*&
212     (cdexp(cj*k0*a1*cspth*snth)*&
213     (-1.0_wp)**m-1.0_wp)*sum
214
215     !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&
216 end if
217
218 !computation of F_x
219 if(Nz.gt.0) then
220     sum=(0.0_wp,0.0_wp)
221     do p=1,Nz
222         add_term=E(p)*p/&
223             ((p*pi/(2.0_wp*Lz))**2-kz_sq)
224         if(mod(p,2).eq.1) then !p odd
225             add_term=add_term*dcos(kz*Lz)
226         else      !p even
227             add_term=-add_term*cj*dsin(kz*Lz)
228         end if
229         sum=sum+add_term
230     enddo
231     F_xyz(1)=F_xyz(1)+2.0_wp*(1.0_wp+cj)/(Lz*a1)*&
232         epsilon_m*kz*kx_sq/k0_sq/(k0_sq*cspth**2*snth**2-kx_sq)*&
233         dsinc(w*kx/2.0_wp)*dcos(kx*x0_pp)*cdexp(-cj*ky*b1)*&
234         (cdexp(cj*k0*a1*cspth*snth)*(-1.0_wp)**m-1.0_wp)*sum
235         !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&
236
237 end if
238
239 m=0
240 epsilon_m=0.5_wp
241 kx=m*pi/a1
242 kx_sq=kx**2
243 ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz**2))
244 if (dimag(ky).gt.0.0_wp) then
245     ky=-ky
246 endif
247
248 !computation of F_z
249 if(Nz.gt.0) then
250     sum=(0.0_wp,0.0_wp)
251     do p=1,Nz
252         add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-k0_sq*csth**2)
253         if(mod(p,2).eq.1) then !p odd
254             add_term=add_term*dcos(kz*Lz)
255         else      !p even
256             add_term=-add_term*cj*dsin(kz*Lz)
257         end if
258         sum=sum+add_term
259     enddo
260

```

```

261         F_xyz(3)=2.0_wp*cj*w/(Lz*a1)*&
262             epsilon_m*((k0_sq-kz_sq)+cj*kz_sq)/k0_sq*&
263             dsinc(w*kx/2.0_wp)*dcos(kx*x0_pp)*&
264             cdexp(-cj*ky*b1)*&
265             cj*a1*dsinc(a1*k0*csp*snth)*sum
266             !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&
267
268     endif
269
270     !transform into spherical coor.
271     call xyz2sph(F_xyz,sph,F_sph)
272
273 return
274 end
275
276 !#####
277 !
278 ! Name:
279 !     rad_field
280 !
281 ! Revised:
282 !     February 29, 2000
283 !
284 ! Purpose:
285 !     This subroutine reads expansion coefficients from external file
286 !     "efield.dat" and computes the principle patterns from a slot
287 !     radiating into either half space or a PP waveguide.
288 !
289 ! Usage:
290 !     call rad_field()
291 !
292 ! Arguments:
293 !     plane - Type of principle plane to be computed. (Input)
294 !           plane    principle plane
295 !           1        E-plane
296 !           2        H-plane
297 !
298 ! Modules:
299 !     const,gen_var,slot_var,radfield_var,ppwg_var,wg_var
300 !
301 ! Subroutines:
302 !     read_efield,F_sph_fs,F_sph_ppwg
303 !
304 ! Remarks:
305 !     The output is printed to standard output.
306 !
307 subroutine rad_field(plane)
308 !modules
309 use const, only : dtr,rtd,cj,wp,Z0,pi
310 use gen_var, only : lambda_0,k0,ext_type
311 use slot_var, only : Lx,w,x00,x00_pp,x0,x0_pp
312 use radfield_var, only : E_sph,F_sph,sph,H_sph
313 use ppwg_var, only : a1
314 use wg_var, only : a,lambda_g,beta_10
315 !implicit statement
316 implicit none
317 !declaration of intrinsic functions
318 intrinsic dcos,dsin,cdexp,dcmplx,cdsqrt
319 !declaration of functions

```

```

320     real(wp)      :: dsinc
321     !list of calling arguments
322     integer,intent(in) :: plane
323     !list of local variables
324     complex(wp) :: arrayfactor_copol,arrayfactor_xpol
325     integer      :: i,n,nth
326     real(wp)     :: d
327
328     call read_efield()
329     x00=x0+w/2.0-Lx
330     x0_pp=x0-(a-a1)/2.0_wp
331     x00_pp=x00-(a-a1)/2.0_wp
332
333     sph(1)=100.0_wp*lambda_0
334     if(plane.eq.1) then
335         sph(3)=90.0_wp*dtr
336     else
337         sph(2)=90.0_wp*dtr
338     endif
339
340     nth=179*4
341     do i=1,nth,1
342         if(plane.eq.1) then
343             !sph(2)=dtr*i
344             sph(2)=dtr*179.0_wp*dfloat(i)/dfloat(nth)
345         else
346             sph(3)=dtr*179.0_wp*dfloat(i)/dfloat(nth)
347         endif
348         if(ext_type.eq.1) then
349             call F_sph_fs()
350         else
351             call F_sph_ppwg()
352         endif
353         E_sph(2)=-c*j*k0*F_sph(3)
354         E_sph(3)=c*j*k0*F_sph(2)
355         H_sph(2)=-E_sph(3)/Z0
356         H_sph(3)=E_sph(2)/Z0
357
358         d=lambda_g
359         arrayfactor_copol=(0.0_wp,0.0_wp)
360         arrayfactor_xpol=(0.0_wp,0.0_wp)
361         do n=1,9
362             arrayfactor_copol=arrayfactor_copol+&
363                 cdexp(cj*n*(k0*d*dcos(sph(2))-d*beta_10))
364             arrayfactor_xpol=arrayfactor_xpol+&
365                 cdexp(cj*n*(k0*d*dcos(sph(2))-d*beta_10+1.0_wp*pi))
366         enddo
367
368         write(*,1078) rtd*sph(2),rtd*sph(3),&
369             dreal(E_sph(2)*dconjg(H_sph(3))),&
370             dreal(-E_sph(3)*dconjg(H_sph(2))),&
371             dreal(arrayfactor_copol*dconjg(arrayfactor_copol)),&
372             dreal(arrayfactor_xpol*dconjg(arrayfactor_xpol))
373     enddo
374
375     1078 format(2(2x,F7.2),4(2x,E15.6))
376     return
377     end
378

```



```

1      !#####
2      !
3      ! Name:
4      !       write_y_norm
5      !
6      ! Revised:
7      !       February 29, 2000
8      !
9      ! Purpose:
10     !       This subroutine computes the equivalent normalized shunt
11     !       admittance. Only recommended for longitudinal slots.
12     !
13     ! Usage:
14     !       call write_y_norm()
15     !
16     ! Arguments:
17     !
18     ! Modules:
19     !       const,slot_var,gen_var
20     !
21     ! Subroutines:
22     !       backscat
23     !
24     ! Remarks:
25     !       The output is printed to standard output.
26     !
27     subroutine write_y_norm()
28     !modules
29     use const, only : wp
30     use slot_var, only :Lz,x0
31     use gen_var, only : lambda_0
32     !implicit statement
33     implicit none
34     !declaration of intrinsic functions
35     intrinsic dreal,dimag
36     !declaration of functions
37     complex(wp)      :: backscat
38     !list of local variables
39     complex(wp)      :: R,y
40
41     R=backscat()
42     y=-2.0_wp*R/(1.0_wp+R)
43     write(*,*) x0, 2.0_wp*Lz/lambda_0, dreal(y), dimag(y)
44
45     return
46     end
47
48     !#####
49     !
50     ! Name:
51     !       write_nw_par
52     !
53     ! Revised:
54     !       February 29, 2000
55     !
56     ! Purpose:
57     !       This subroutine computes the equivalent normalized
58     !       series impedance. It writes frequency, slot length, offset,
59     !       series impedance, backward and forward scattering coefficients,

```

```
60      !           and expansion coefficients.
61      !
62      ! Usage:
63      !       call write_nw_par()
64      !
65      ! Arguments:
66      !
67      ! Modules:
68      !       const,slot_var,gen_var,mom_var,ppwg_var
69      !
70      ! Subroutines:
71      !       backscat,forwardscat
72      !
73      ! Remarks:
74      !       The output is printed to standard output.
75      !
76      subroutine write_nw_par()
77      !modules
78      use const, only : rtd,cj,wp
79      use slot_var, only : Lx,Lz,x0
80      use ppwg_var, only : a1
81      use gen_var, only : freq
82      use mom_var, only : E
83      !implicit statement
84      implicit none
85      !declaration of intrinsic functions
86      complex backscat, forwardscat
87      !list of local variables
88      complex(wp) R,T,z
89
90      R=backscat()
91      T=forwardscat()
92      z=2.0_wp*R/(1.0_wp-R)
93      write(*,1075) freq,x0,a1,Lx,Lz,dreal(z),dimag(z),&
94      dreal(R),dimag(R),dreal(T),dimag(T),E
95
96      1075 format(2x,EN15.6,4(2x,EN15.6),6(2x,E15.6),48(2x,E15.6),26(2x,E15.6))
97      return
98      end
99
```

Example of input file *slot.in*

```
&var_list
freq_0 = 9.375000E9,
nfreq  = 10,
dfreq  = 100.000E6,
Lz0    = 9.324199E-03,
nll    = 15,
dll    = 0.01000,
Lx0    = 3.80000E-3,
ntl    = 10,
dtl    = 0.30000E-3,
w      = 1.587500E-3,
x0     = 11.43000E-3,
a      = 22.86000E-3,
b      = 10.16000E-3,
t      = 1.27000E-3
```

```
ext_type = 2,  
a1      = 21.0000E-3,  
b1      = 25.0000E-3,  
sw_type = 5  
/
```