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 formulaton involves derivation of an integral equation which is solved by the method of moments using entire basis functions for expansion of the electric filed 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 impendance 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 rectilinar 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 eguation 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 polarisationskarakteristik 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 benytten tværgående slidser i enten den bredde eller smalle side af bølgelederen. De har imidlertid begge deres begrænsniger i form af en høj eksitation med et lille dynamikområde og skråstilning introducere et krydspolariseret felt.

For at overkomme disse problemer foreslåes en T-slids i den bredde 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 udstrålingskarakteristiker.

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 success til løsning af ligende opgaver, *Lars Josefsson* [2][3][4]. Endelig vægtykkelse er inkluderet i modelen 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 bøelgelængde. Dette giver anledning til en sidesløjfe i E-planet, men ved brug af en plan paralle bølgeleder uden på den rektagulære bølgeleder kan disse undertrykkes. Dette gaelder 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

List of Figures

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 rectilinar 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

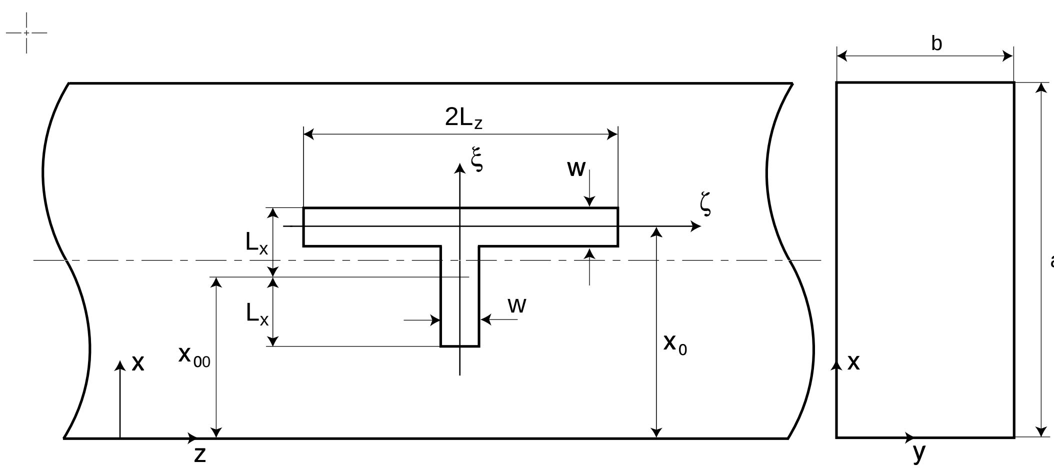

Figure 1.1.: Geometry of waveguide-fed T-slot.

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 extended 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 nessesary 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]

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

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

$$\nabla \cdot \mathbf{D} = \rho \tag{1.1c}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{1.1d}$$

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 **J** as well as magnetic current density **M** are the bases of the following derivations. In a free-space environment the constitutive relations are

$$\mathbf{D} = \epsilon_0 \mathbf{E} \tag{1.2a}$$

$$\mathbf{B} = \mu_0 \mathbf{H} \tag{1.2b}$$

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

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

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

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

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

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

where **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

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

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

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 Enviroment

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

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

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]

$$\mathbf{D} = -\nabla \times \mathbf{F} \tag{1.7}$$

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

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

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

$$\mathbf{H} + j\omega\mathbf{F} = -\nabla\Phi_m \tag{1.9}$$

In order that eq. 1.1a will hold we require

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

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

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

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

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

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

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

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

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

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

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

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

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

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]

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

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

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

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

$0'$ gives

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

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

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

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

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

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

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

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

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

Seperating 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

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

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

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

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

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

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

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

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

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

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

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]

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

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

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

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]

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

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

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

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

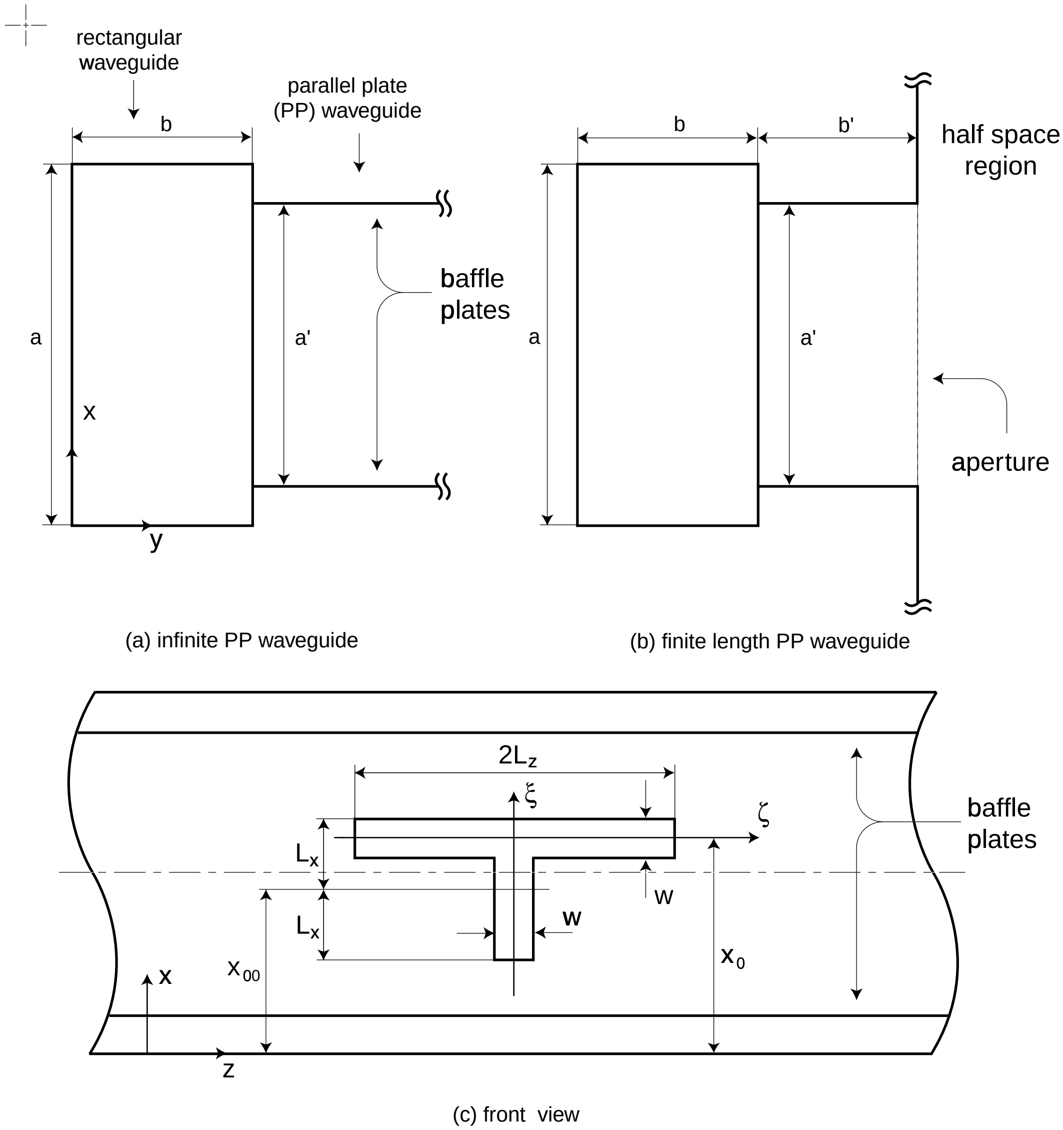

Figure 1.2.: Geometry of waveguide-fed T-slot radiating into a parallel plate waveguide consisting of two baffle plates.

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

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

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

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

with the boundary conditions

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

Then the H-fields can be found by the convolution

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

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]

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

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

$$k_x = \frac{m\pi}{a'}, \qquad m = 1, 2, \ldots \tag{1.37}$$

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

$$k_y = \sqrt{k^2 - k_x^2 - k_z^2} \tag{1.38}$$

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

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

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

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

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

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

which can be rewritten as

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

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

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 $\overline{\overline{G}}_{e2}$ gives

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

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

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

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

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

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

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

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

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

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

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

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

and eq. 1.47 is rewritten as

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

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

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

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

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

$$|\bar{r} - \bar{r}'| = r - r' \cos\psi \tag{1.52}$$

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

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

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

$$E_\theta = Z_0 H_\phi, \qquad E_\phi = -Z_0 H_\theta \tag{1.54}$$

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

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

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

with **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

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

Substituting for x, y, z, we obtain

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

which is the desired form when rectangular coordinates are choosen 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 hight b' as shown in figure 1.2(c). For an approximate solution to this problem for $y > b'$, we assume **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 **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

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

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

$$\nabla \times \overline{\overline{G}} = \sum_{i}^{3}\sum_{j}^{3} \left(\nabla G_{ij} \times \hat{x}_i\right)\hat{x}_j \tag{1.59}$$

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

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

which upon evaluation of the derivatives becomes

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

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

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

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

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

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

$$E_z^{ppwg}(\bar{R}) = \frac{1}{\pi a'}\int_{-\infty}^{\infty}\sum_{m=0}^{\infty}\epsilon_m^2\sin(k_x x)\mathrm{e}^{-jk_y y}\mathrm{e}^{-jk_z z} \cdot\left[\int_{slot} E_z(\bar{R}')\sin(k_x x')\mathrm{e}^{jk_z z'}dS' - \frac{(1+j)k_x k_z}{k^2}\int_{slot} E_x(\bar{R}')\cos(k_x x')\mathrm{e}^{jk_z z'}dS'\right]dk_z \tag{1.63b}$$

Assuming the electric field **E** in the aperture to be of the form of the incident field we form an equivalent magnetic aperture current

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

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

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

where

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

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 $\pm 45°$ 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

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

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

$$GLA = 20\log_{10} \mathrm{e}^{-jk_y b'} \tag{1.68}$$

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 decribe the concept of the method of moments and apply it to the slot problem derived in chap. 1. Furthermore, intergration concepts and schemes nessesary 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 bondary 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.

Formulate IE $\quad A(\mathbf{x}) = \mathbf{y} \quad$ on S

↓

($\mathbf{y}$ given, $\mathbf{x}$ unknown)

↓

Discretize IE $\quad$ (Choose expanding functions Ψ_n)

↓

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

↓

$\sum_{n=1}^{N} x_n^{(N)} A(\Psi_n) \simeq \mathbf{y}$ $\quad$ Discretize IE (approximated IE)

↓

Solution of discretized IE

↓

Choose weighting functions

↓

Matrix equation

↓

The unknown $x_n^{(N)}$ is determined and the solution of IE obtained

↓

Output data

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

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

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

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

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

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

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

$$\langle \mathbf{x}, \mathbf{y} \rangle = \int_S \mathbf{x} \cdot \mathbf{y} dS \tag{2.4}$$

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

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

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

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

where

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

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

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

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

where $\left[A_{mn}^{-1}\right]$ 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

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

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

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

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

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

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

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

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

and inner products defined as in eq. 2.4

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

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

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

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

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

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

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

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

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

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

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

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

$$h_q^{inc} = \left\langle H_z^{inc}, w_q\right\rangle \tag{2.17a}$$

$$h_s^{inc} = \left\langle H_x^{inc}, w_s\right\rangle \tag{2.17b}$$

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

$$[E] = \left[Y^{ext} - Y^{int}\right]^{-1} \left[h^{inc}\right] \tag{2.18}$$

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

$$\begin{aligned} Y_{pq}^{ext} \;=\; & \frac{2Z_0}{j\pi^2\omega\mu_0(q^2-p^2)}\Big[\left[(k2L_z)^2 - (q\pi)^2\right]\cdot p\cdot y_1(q) \\ & - \left[(k2L_z)^2 - (p\pi)^2\right]\cdot q\cdot y_1(p)\Big] \qquad p \neq q \end{aligned} \tag{2.19}$$

and

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

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

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

and

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

The three quantities R, R_1 and R_2 are given by

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

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

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

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

$$Y_{pq}^{int} = \frac{2j\lambda w}{ab\pi}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\epsilon_{mn}^2 W_m \left[\frac{\frac{p\pi}{2L_z}}{\gamma_{mn}}\frac{\gamma_{mn}^2 + k^2}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2}\frac{\frac{q\pi}{2L_z}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2}\right. \cdot \left(1 \pm \mathrm{e}^{-2\gamma_{mn}L_z}\right) + \left.\frac{k^2 - \left(\frac{p\pi}{2L_z}\right)^2}{\gamma_{mn}^2 + \left(\frac{p\pi}{2L_z}\right)^2} L_z\delta_{pq}\right] \tag{2.24}$$

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

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

and

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

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

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

and

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

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

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

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

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

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

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

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

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

where

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

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

and the interior coupling is

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

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

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

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

where

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

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

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

$$Y_{rq}^{int} = Y_{rq}^{int,1} + Y_{rq}^{int,23} \tag{2.35}$$

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

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

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

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

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

Excitation Terms

Finally the forcing terms are given by

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

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

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

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

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

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

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

$$\begin{aligned} Y_{pq}^{ext} &= \frac{-Z_0 w p q \pi}{a' \omega \mu_0 L_z^2} \sum_{m=0}^{\infty} \epsilon_m^2 \mathrm{sinc} \frac{k_x w}{2} \cos^2(k_x x_{0,pp}) \\ &\cdot \int_0^{\infty} \frac{\left(k^2 - k_z^2\right)}{k_y \left[\left(\frac{p\pi}{2L_z}\right)^2 - k_z^2\right] \left[\left(\frac{q\pi}{2L_z}\right)^2 - k_z^2\right]} \begin{bmatrix} 1 + \cos(2k_z L_z) \\ 1 - \cos(2k_z L_z) \end{bmatrix}_{p,q \text{ even}}^{p,q \text{ odd}} dk_z \end{aligned} \tag{2.42}$$

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

$$\begin{aligned} Y_{rs}^{ext} &= \left\langle H_x^{ext}(E_z), w_q \right\rangle \frac{Z_0}{E_r} \\ &= \frac{Z_0}{\pi a' \omega \mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 \left(k^2 - k_x^2\right)}{k_y} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \sin\left[k_x (x_{00,pp+\xi})\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{r\pi}{2L_x}(L_x + \xi')\right] \sin\left[k_x (x_{00,pp} + \xi')\right] \mathrm{e}^{jk_z\zeta'} d\xi' d\zeta' d\xi d\zeta dk_z \end{aligned} \tag{2.43}$$

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

$$\begin{aligned} Y_{rs}^{ext} &= \frac{Z_0 w r}{a' \omega \mu_0 L_x} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 \left(k^2 - k_x^2\right)}{k_y} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \sin\left[k_x (x_{00,pp} + \xi)\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \frac{\mathrm{sinc}\frac{k_z w}{2}}{\left(\frac{r\pi}{2L_x}\right)^2 - k_x^2} \begin{bmatrix} \cos(k_x L_x) \sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp}) \sin(k_x L_x) \end{bmatrix}_{r \text{ even}}^{r \text{ odd}} d\xi d\zeta dk_z \end{aligned} \tag{2.44}$$

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

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

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

$$\begin{aligned} Y_{ps}^{ext} &= \left\langle H_x^{ext}(E_x), w_s \right\rangle \frac{Z_0}{E_p} \\ &= \frac{-jZ_0}{\pi a'\omega\mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x k_z}{k_y} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right] \delta(\zeta) \sin\left[k_x(x_{00,pp}+\xi)\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right] \cos\left[k_x(x_{0,pp}+\xi')\right] \mathrm{e}^{jk_z\zeta'} d\xi' d\zeta' d\xi d\zeta dk_z \end{aligned} \tag{2.46}$$

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

$$\begin{aligned} Y_{ps}^{ext} &= \frac{-j2Z_0 p}{a'\omega\mu_0 L_z} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_z}{k_y} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right] \delta(\zeta) \sin\left[k_x(x_{00,pp}+\xi)\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \frac{\sin\frac{k_x w}{2} \cos(k_x x_{0,pp})}{\left[\left(\frac{p\pi}{2L_z}\right)^2 - k_z^2\right]} \begin{bmatrix} \cos(k_z L_z) \\ -j\sin(k_z L_z) \end{bmatrix}_{p \text{ even}}^{p \text{ odd}} d\xi d\zeta dk_z \end{aligned} \tag{2.47}$$

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

$$Y_{ps}^{ext} = \frac{-4Z_0 ps\pi}{a'\omega\mu_0 L_z L_x} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 \sin\frac{k_x w}{2} \cos(k_x x_{0,pp})}{\left[\left(\frac{s\pi}{2L_x}\right)^2 - k_x^2\right]} \tag{2.48}$$

$$\cdot \left[\begin{array}{c} \cos(k_x L_x)\sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp})\sin(k_x L_x) \end{array} \right]_{s \text{ even}}^{s \text{ odd}} \cdot \int_0^{\infty} \frac{k_z \sin(k_z L_z)}{k_y \left[\left(\frac{p\pi}{2L_z}\right)^2 - k_z^2\right]} dk_z$$

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

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

$$\begin{aligned} Y_{rq}^{ext} &= \langle H_z^{ext}(E_z), w_q \rangle \frac{Z_0}{E_r} \\ &= \frac{-jZ_0}{\pi a'\omega\mu_0} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x k_z}{k_y} \\ &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin\left[\frac{q\pi}{2L_z}(L_z + \zeta)\right] \delta(\xi) \cos\left[k_x(x_{0,pp} + \xi)\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{r\pi}{2L_x}(L_x + \xi')\right] \sin\left[k_x(x_{00,pp} + \xi')\right] \mathrm{e}^{jk_z\zeta'} d\xi' d\zeta' d\xi d\zeta dk_z \end{aligned} \tag{2.49}$$

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

$$\begin{aligned} Y_{rq}^{ext} &= \frac{-j2Z_0 r}{a'\omega\mu_0 L_x} \int_{-\infty}^{\infty} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x}{k_y} \\ &\cdot \int_{-L_z}^{L_z} \int_{-w/2}^{w/2} \sin\left[\frac{q\pi}{2L_z}(L_z + \zeta)\right] \delta(\xi) \cos\left[k_x(x_{0,pp} + \xi)\right] \mathrm{e}^{-jk_z\zeta} \\ &\cdot \frac{\sin\frac{k_z w}{2}}{\left[\left(\frac{r\pi}{2L_x}\right)^2 - k_x^2\right]} \left[\begin{array}{c} \cos(k_x L_x)\sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp})\sin(k_x L_x) \end{array} \right]_{r \text{ even}}^{r \text{ odd}} d\xi d\zeta dk_z \end{aligned} \tag{2.50}$$

Evaluating the last integral yields

$$Y_{rq}^{ext} = \frac{4Z_0 rq\pi}{a'\omega\mu_0 L_x L_z} \sum_{m=0}^{\infty} \frac{\epsilon_m^2 k_x \cos(k_x x_{0,pp})}{\left[\left(\frac{r\pi}{2L_x}\right)^2 - k_x^2\right]} \tag{2.51}$$

$$\cdot \begin{bmatrix} \cos(k_x L_x)\sin(k_x x_{00,pp}) \\ -\cos(k_x x_{00,pp})\sin(k_x L_x) \end{bmatrix}_{r\ \text{even}}^{r\ \text{odd}}$$

$$\cdot \int_0^{\infty} \frac{\sin\frac{k_z w}{2}\sin(k_z L_z)}{k_y\left[\left(\frac{q\pi}{2L_z}\right)^2 - k_z^2\right]} dk_z$$

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

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

and

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

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

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

and

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

From the above

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

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

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

where

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

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

$$\begin{aligned} i_1 &= \int_{-1}^{0} \frac{\mathrm{e}^{-jkR}}{R} \int_{-1/2}^{1/2+\nu} \sin\left[q\pi(\sigma + 1/2)\right] \sin\left[p\pi(\sigma - \nu + 1/2)\right] d\sigma d\nu \\ &+ \int_{0}^{1} \frac{\mathrm{e}^{-jkR}}{R} \int_{-1/2+\nu}^{1/2} \sin\left[q\pi(\sigma + 1/2)\right] \sin\left[p\pi(\sigma - \nu + 1/2)\right] d\sigma d\nu \end{aligned} \tag{2.60}$$

which after evaluation of inner integrals reduces to

$$i_1 = \int_{0}^{1} \frac{\mathrm{e}^{-jkR}}{R} \left[(1 - \nu)\cos(q\pi\nu) + \frac{\sin(q\pi\nu)}{q\pi}\right] d\nu \qquad p = q \tag{2.61a}$$

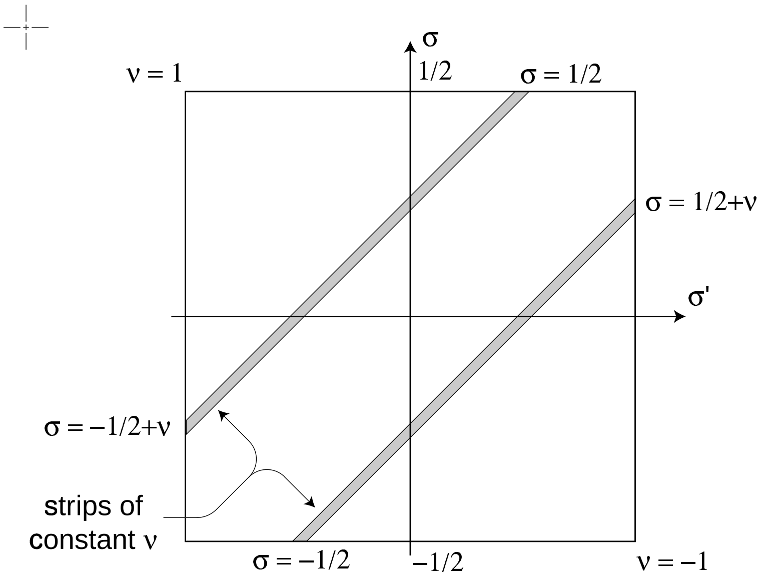

Figure 2.2.: Integration on strips of constant ν, $\sigma - \sigma'$.

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

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

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

and

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

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

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

and

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

R, R_1 and R_2 are given by

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

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

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

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

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

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

$$\int_S \frac{1}{R}\left[f(\xi,\zeta) - f(\xi_0,\zeta_0)\right] dS + f(\xi_0,\zeta_0)\int_S \frac{1}{R} dS \tag{2.68}$$

$$= \int_S \frac{1}{R} g(\xi,\zeta) dS + f(\xi_0,\zeta_0)\int_S \frac{1}{R} dS$$

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 sigular 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

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

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

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

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

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 interst 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 seperated into three regions as

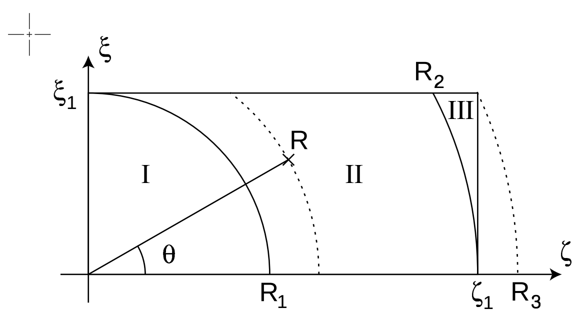

Figure 2.3.: Integration over a rectangular region in polar coordinates.

shown in figure 2.3. The tree 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

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

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

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

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

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

Similarly, the contribution from region three becomes

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

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

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

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

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

$$I_{R_1} = \frac{1}{R_1}\sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right]\cos\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right]\cos(kR_1) \tag{2.76}$$
$$I_{R_2} = \frac{1}{R_2}\sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right]\cos\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right]\cos(kR_2) \tag{2.77}$$

both being singular at $\xi = L_x$ and $\xi = L_x - w$ respectively. Applying the method outlined theses two integrands have been plottet 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

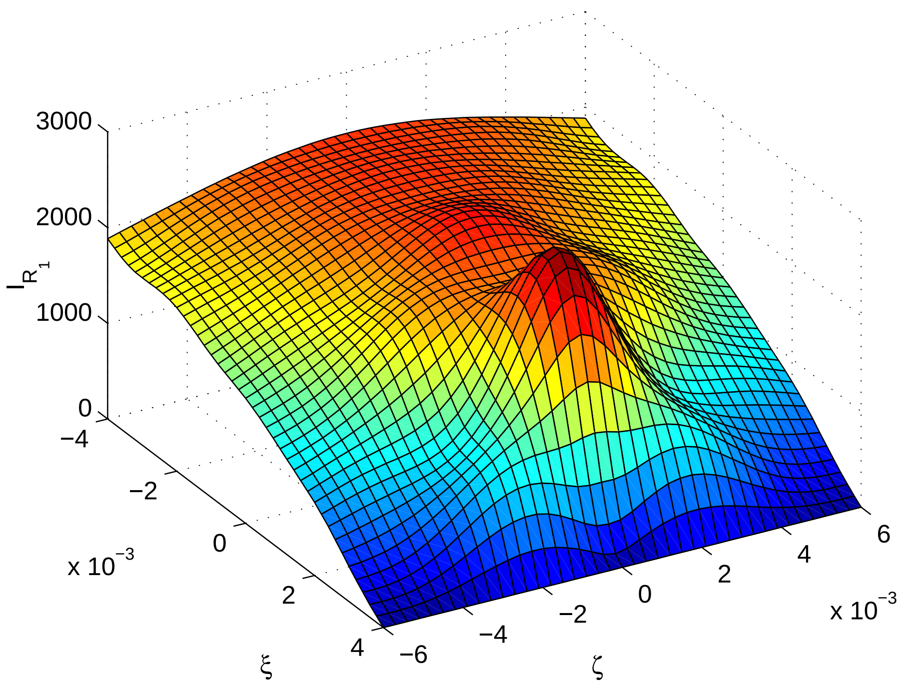

Figure 2.4.: Value of the kernel I_{R_1}.

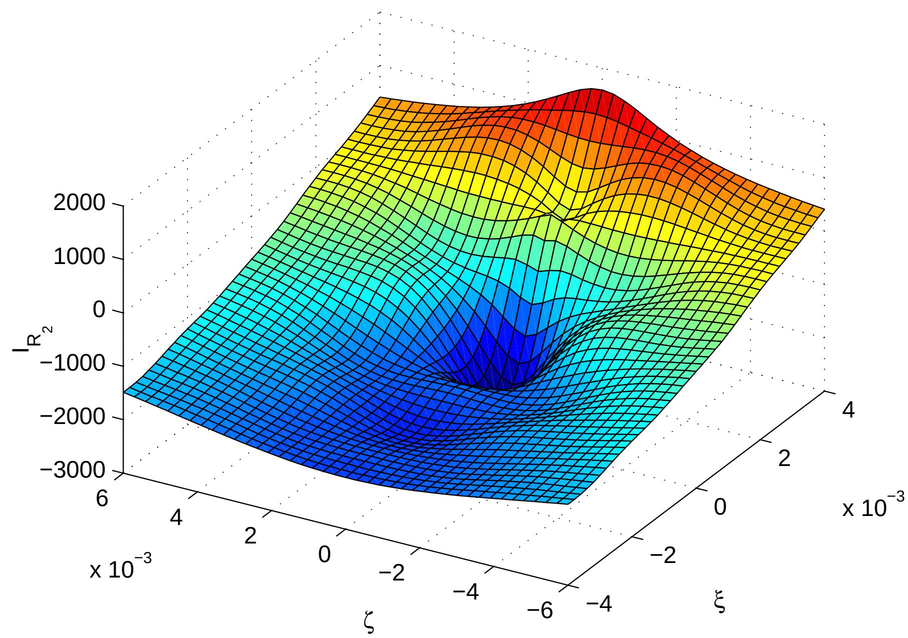

Figure 2.5.: Value of the kernel I_{R_2}.

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

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

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

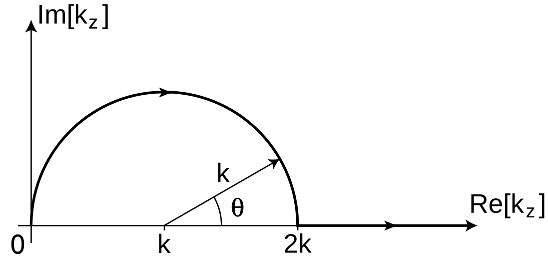

Figure 2.6.: Integration contour in the complex k_z plane.

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 $2k \leq k_z \leq \infty$

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

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

$$I(k) = \int_a^{\infty} g(k\lambda) f(\lambda) d\lambda \tag{2.80}$$

where

- $g(k\lambda)$ is a complex function whose real and imaginary parts oscillate with a strictly periodic behavior or behave asymtotically as the product of a periodic function and monotonic function.

- $f(\lambda)$ is a smooth, non-oscillating function that behaves asymtotically 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]

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

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 $I_m^1 (m = 1, 2, \ldots, M)$ as [15]

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

where the λ_m are succesive 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

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

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 $I_m^2 (m = 1, 2, \ldots, M-1)$ is then defined taking the averages of two consective values of the sequence I_m^1, following the general expression

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

Use of eq. 2.83 shows that

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

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

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

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

$$w_m^1 = (\lambda_1/\lambda_m)^\alpha \tag{2.87}$$

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

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

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

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

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

$$\cdot \cos\left[\frac{\pi}{a}(x_0+\xi')\right] \sin\left[\frac{p\pi}{2L_z}(\zeta'+L_z)\right] \mathrm{e}^{-j\beta_{10}\zeta'} d\zeta' d\xi'\Bigg]$$

Evaluation of the integrals yields

$$\frac{B_{10}}{A_{10}} = \frac{-2\pi^2}{\omega\mu_0 a^2 b\beta_{10}}\left[\sin\left(w/2\beta_{10}\right)\sum_{r=1}^{N_x} E_r \frac{\frac{r}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2}\right. \tag{2.90}$$

$$\cdot \left[\begin{array}{l} \cos\frac{\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{\pi x_{00}}{a}\sin\frac{\pi L_x}{a} \end{array}\right]_{r \text{ even}}^{r \text{ odd}} + j\sin\frac{\pi w}{2a}\cos\frac{\pi x_0}{a}$$

$$\left.\cdot \sum_{p=1}^{N_z} E_p \frac{\frac{p}{L_z}}{\left(\frac{p\pi}{2L_z}\right)^2 - \beta_{10}^2} \cdot \left[\begin{array}{l} \cos(\beta_{10}L_z) \\ j\sin(\beta_{10}L_z) \end{array}\right]_{p \text{ even}}^{p \text{ odd}}\right]$$

Similarly the forward scattering coefficient becomes

$$\frac{C_{10}}{A_{10}} = \frac{\pi}{\omega\mu_0 a^2 b\beta_{10}}\left[\int_{-w/2}^{w/2}\int_{-L_x}^{L_x} \beta_{10}\sum_{r=1}^{N_x} E_r \sin\left[\frac{r\pi}{2L_x}(\xi'+L_x)\right]\right. \tag{2.91}$$

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

$$\cdot \cos\left[\frac{\pi}{a}(x_0+\xi')\right] \sin\left[\frac{p\pi}{2L_z}(\zeta'+L_z)\right] \mathrm{e}^{j\beta_{10}\zeta'} d\zeta' d\xi'\Bigg]$$

and by evaluation of integrals we obtain

$$\frac{C_{10}}{A_{10}} = \frac{2\pi^2}{\omega\mu_0 a^2 b\beta_{10}}\left[\sin\left(w/2\beta_{10}\right)\sum_{r=1}^{N_x} E_r \frac{\frac{r}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2}\right. \tag{2.92}$$

$$\cdot \left[\begin{array}{l} \cos\frac{\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{\pi x_{00}}{a}\sin\frac{\pi L_x}{a} \end{array}\right]_{r \text{ even}}^{r \text{ odd}} - j\sin\frac{\pi w}{2a}\cos\frac{\pi x_0}{a}$$

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

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

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

Using eq. 2.9b and 1.57 we obtain

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

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

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

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

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

Inserting expressions for the electric field yields

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

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

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

In spherical coordinates, we obtain

$$F_\theta = F_x\cos\theta\cos\phi - F_z\sin\theta \tag{2.99a}$$

$$F_\phi = -F_x\sin\phi \tag{2.99b}$$

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

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

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

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

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

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

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

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

where

$$F_1(r,k_x) = \sum_{r=1}^{N_x}\frac{E_r\frac{r\pi}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2-k_x^2}\begin{bmatrix}\cos\left(k_x L_x\right)\sin\left(k_x x_{00,pp}\right) \\ -\cos\left(k_x x_{00,pp}\right)\sin\left(k_x L_x\right)\end{bmatrix}\begin{matrix}r\text{ odd}\\ \\ r\text{ even}\end{matrix} \tag{2.104}$$

$$F_2(p,k_z) = \sum_{p=1}^{N_z}\frac{E_p\frac{p\pi}{L_z}}{\left(\frac{p\pi}{2L_z}\right)^2-k_z^2}\begin{bmatrix}\cos\left(k_z L_z\right) \\ -j\sin\left(k_z L_z\right)\end{bmatrix}\begin{matrix}p\text{ odd}\\ \\ p\text{ even}\end{matrix} \tag{2.105}$$

The vector potential F_x then becomes

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

$$+\frac{(1+j)k_x k_z}{k^2}\text{sinc}\left(\frac{k_x w}{2}\right)\cos\left(k_x x_{0,pp}\right)F_2(p,k_z)\Bigg]$$

$$\cdot\int_0^{a'}\sin(k_x x')\mathrm{e}^{jkx'\cos\phi\sin\theta}dx'$$

Evaluating the last integral yields

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

Similarly the z-component of the vector potential becomes

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

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

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

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

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

The vector potential F_z then becomes

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

and upon evaluating of x' integral we obtain

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

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 heigth $c = t$ 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

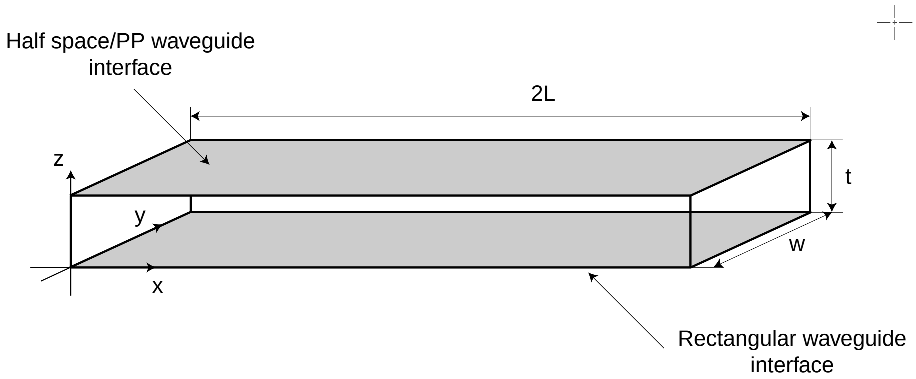

Figure 2.7.: Rectangular cavity representing the finite wall thickness.

slot of lenght $2L$ we obtain

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

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

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 priviously 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

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

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

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

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

where the magnetic currents are written in terms of the unknown electric fields at the lower and upper interfaces

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

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

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]

$$\begin{aligned} G^{cav}(P,P') \;=\; & \frac{-\omega\epsilon}{ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{2\epsilon_{mn}^2}{\beta_{mn}}\frac{k^2-\left(\frac{m\pi}{a}\right)^2}{k^2}\sin\frac{m\pi x}{a}\cos\frac{m\pi y}{b} \\ & \cdot\sin\frac{m\pi x'}{a}\cos\frac{m\pi y'}{b}\mathrm{e}^{-j\beta_{mn}|z-z'|} \end{aligned} \tag{2.117}$$

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]

$$\begin{aligned} & G^{cav}(z'{=}0, z{=}0) \\ & \quad = \frac{-2\omega\epsilon}{ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{2\epsilon_{mn}^2}{\beta_{mn}}\frac{k^2-\left(\frac{m\pi}{a}\right)^2}{k^2}f(x,x',y,y')\cot(\beta_{mn}t) \\ & \quad = \frac{2\omega\epsilon}{ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{2\epsilon_{mn}^2}{j\gamma_{mn}}\frac{k^2-\left(\frac{m\pi}{a}\right)^2}{k^2}f(x,x',y,y')\coth(\gamma_{mn}t) \end{aligned} \tag{2.118a}$$

$$\begin{aligned} & G^{cav}(z'{=}0, z{=}t) \\ & \quad = \frac{-2\omega\epsilon}{ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{2\epsilon_{mn}^2}{\beta_{mn}}\frac{k^2-\left(\frac{m\pi}{a}\right)^2}{k^2}f(x,x',y,y')\frac{1}{\sin(\beta_{mn}t)} \\ & \quad = \frac{2\omega\epsilon}{ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{2\epsilon_{mn}^2}{j\gamma_{mn}}\frac{k^2-\left(\frac{m\pi}{a}\right)^2}{k^2}f(x,x',y,y')\frac{1}{\sinh(\gamma_{mn}t)} \end{aligned} \tag{2.118b}$$

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 nomalization 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

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

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

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

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

and

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

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

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

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

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

where

$$Y_{11,pq} = Y_{pq}^{11} - Y_{pq}^{int} \tag{2.124a}$$

$$Y_{12,pq} = Y_{pq}^{12} \tag{2.124b}$$
$$Y_{21,pq} = Y_{pq}^{12} \tag{2.124c}$$
$$Y_{22,pq} = Y_{pq}^{11} + Y_{pq}^{ext} \tag{2.124d}$$

This $2N \times 2N$ matrix equatione 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 prefered. 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

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

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 succesfully 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

$$\frac{Y}{Y_0} = \frac{-2\Gamma}{1+\Gamma} \tag{3.1}$$

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 longditudinal 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

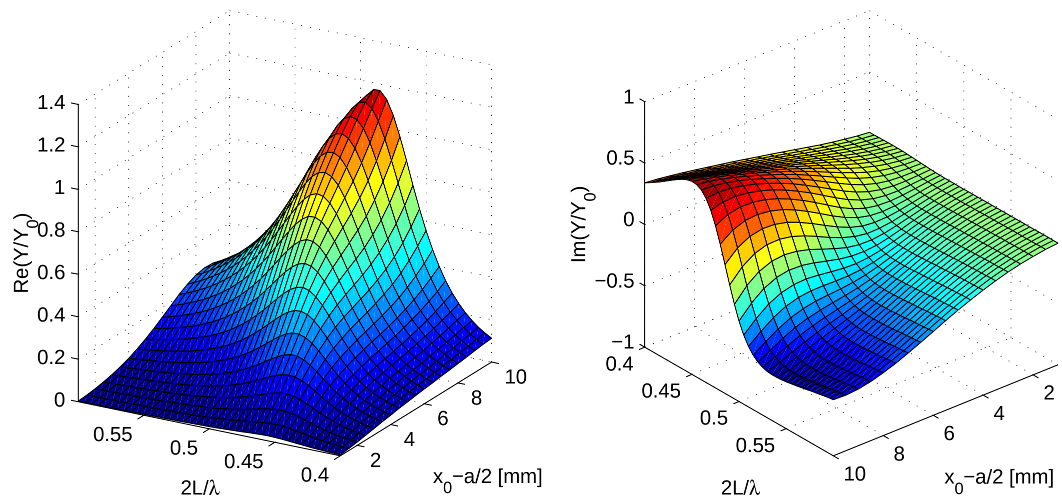

Figure 3.1.: Longitudinal slot admittance - zero wall thickness.

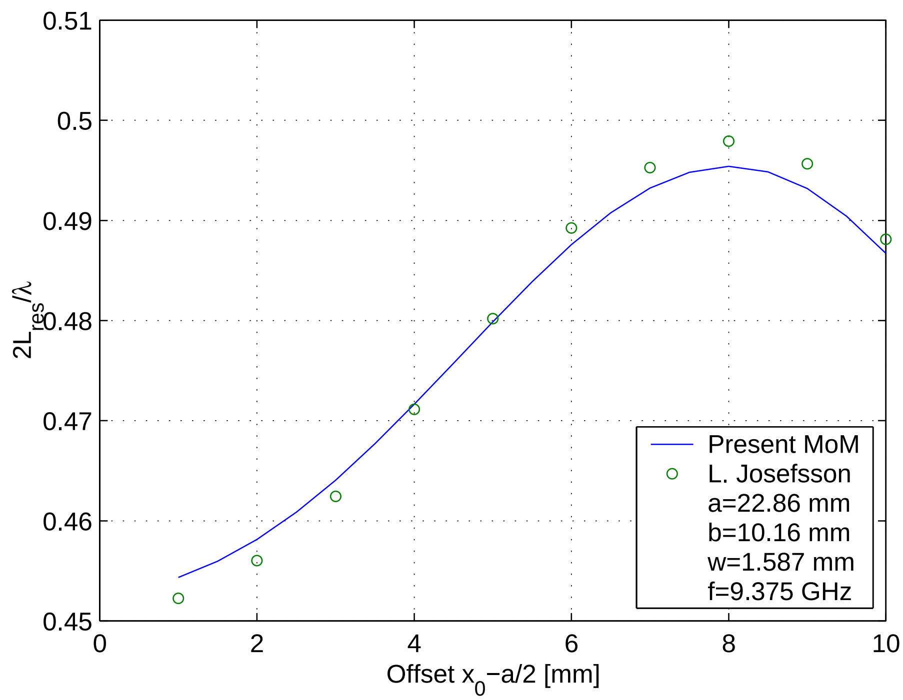

Figure 3.2.: Longitudinal slot resonant length versus offset.

are noticed. The small differences are assumed to be due to the different weighting function as mentioned before.

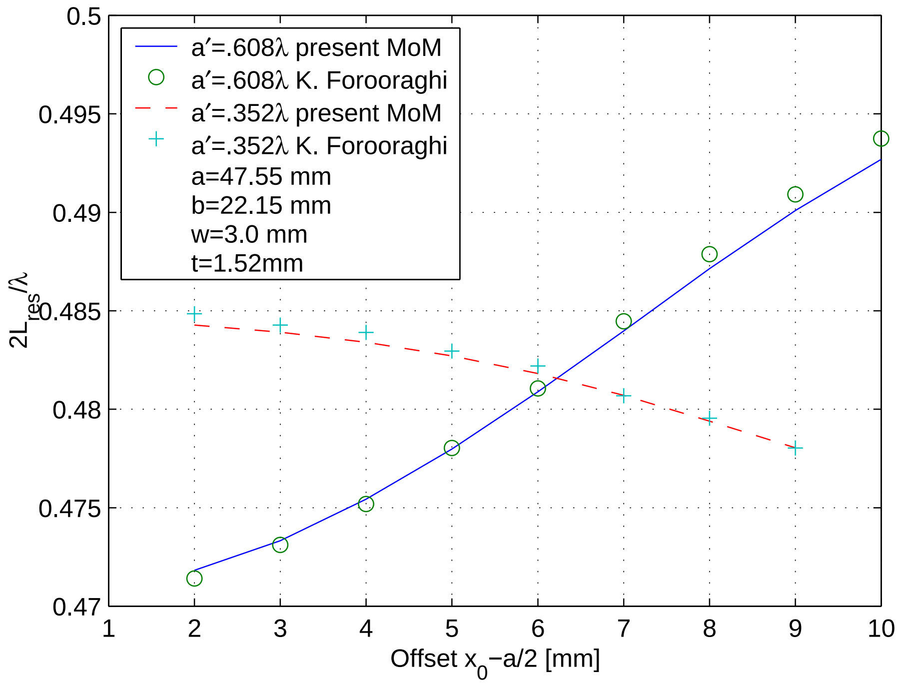

Figure 3.3.: Slot resonant length versus offset for a longitudinal radiating into a PP region.

3.2. Transverse Slots

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

$$\frac{Z}{Z_0} = \frac{2\Gamma}{1 - \Gamma} \tag{3.2}$$

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 impedancecharacteristics 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.

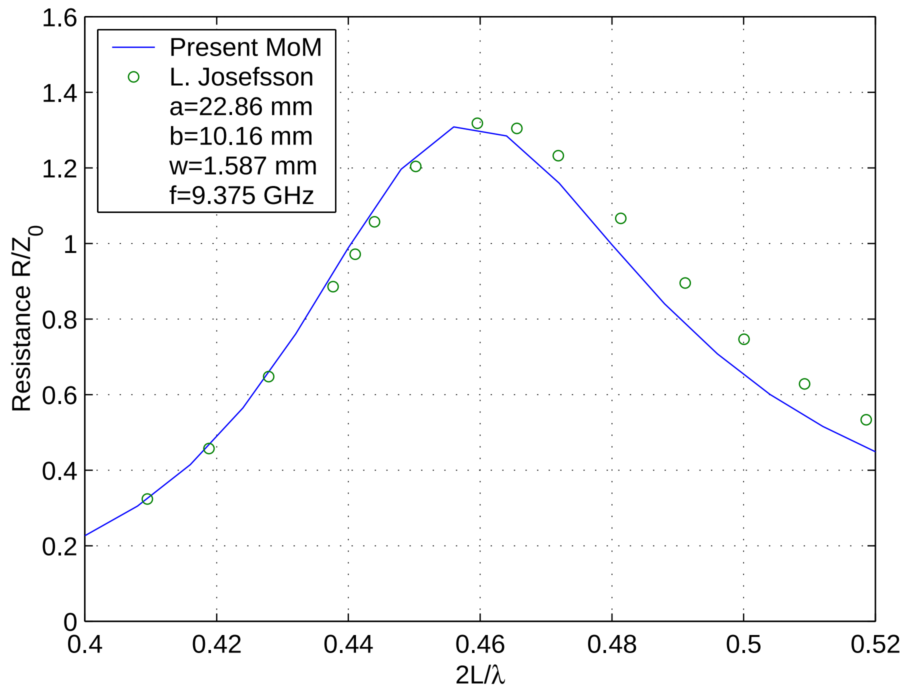

Figure 3.4.: Tranverse slot resistance - zero wall and offset.

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 lenght. 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 proportinal 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 diffrent 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 simpel 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 charateristic as as series impedance.

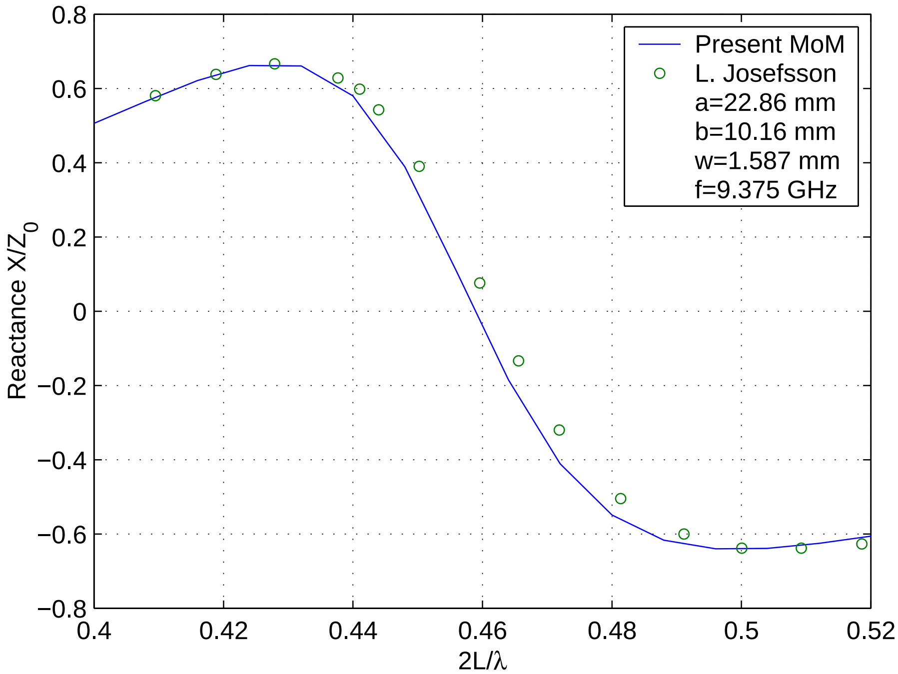

Figure 3.5.: Tranverse slot reactance - zero wall and offset.

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

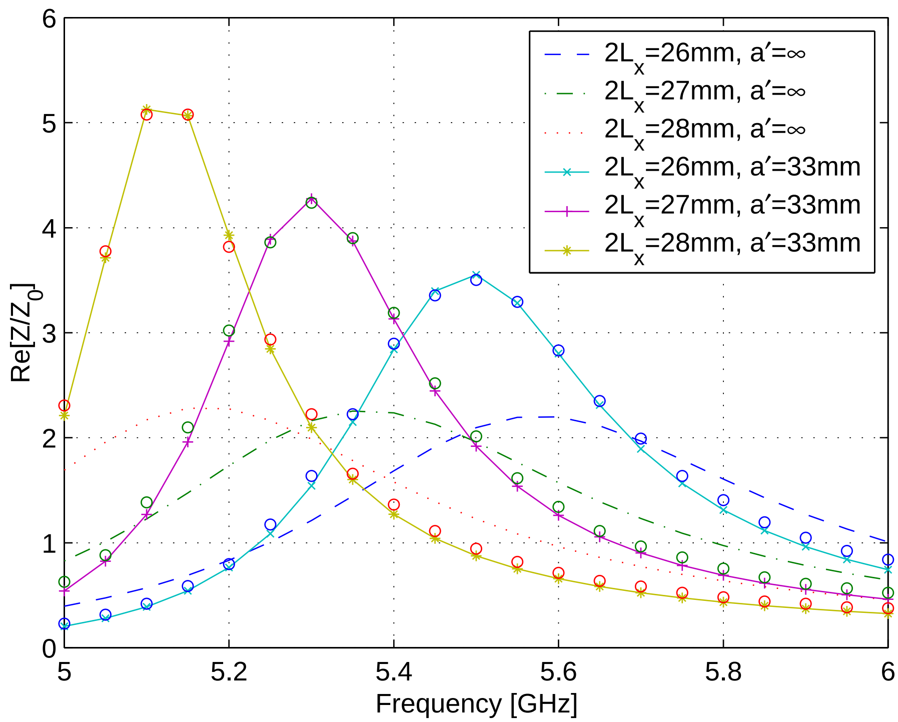

Figure 3.6.: Transverse slot resistance versus frequency for three slot lenghts. The waveguide has the dimension: $a = 39mm$ and $b = 10mm$ and the wall thickness was $t = 1.9mm$.

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 diffent slot lenghts 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 exitation 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

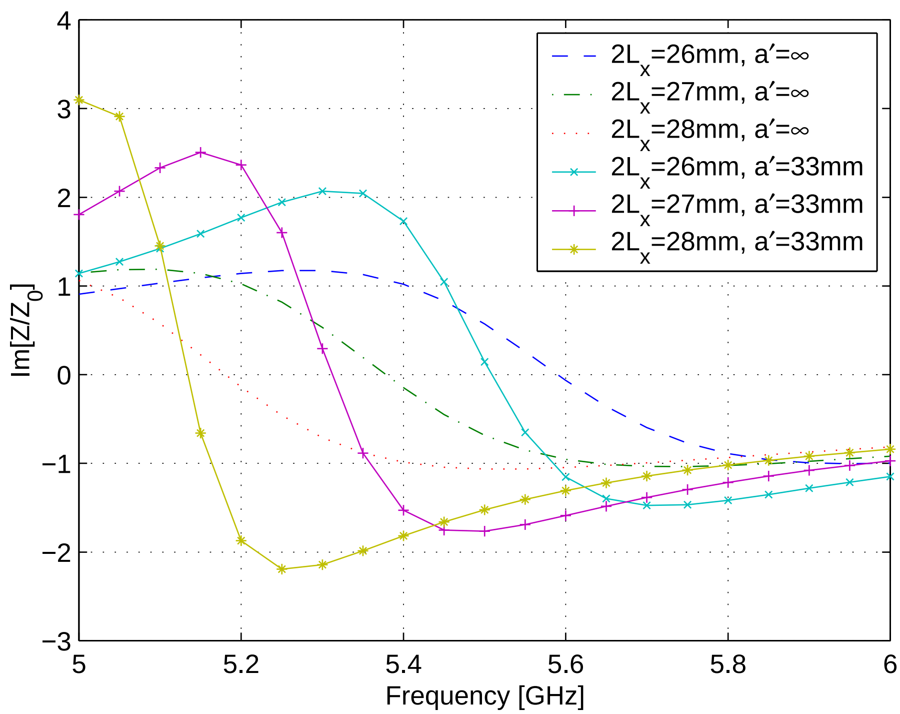

Figure 3.7.: Transverse slot reactance versus frequency for three slot lenghts. The waveguide has the dimension: $a = 39mm$ and $b = 10mm$ and the wall thickness was $t = 1.9mm$.

$L_z + 2L_x$ versus the length L_x. The length $L_z + 2L - x$ can in someway 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 diffent slot lenghts 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

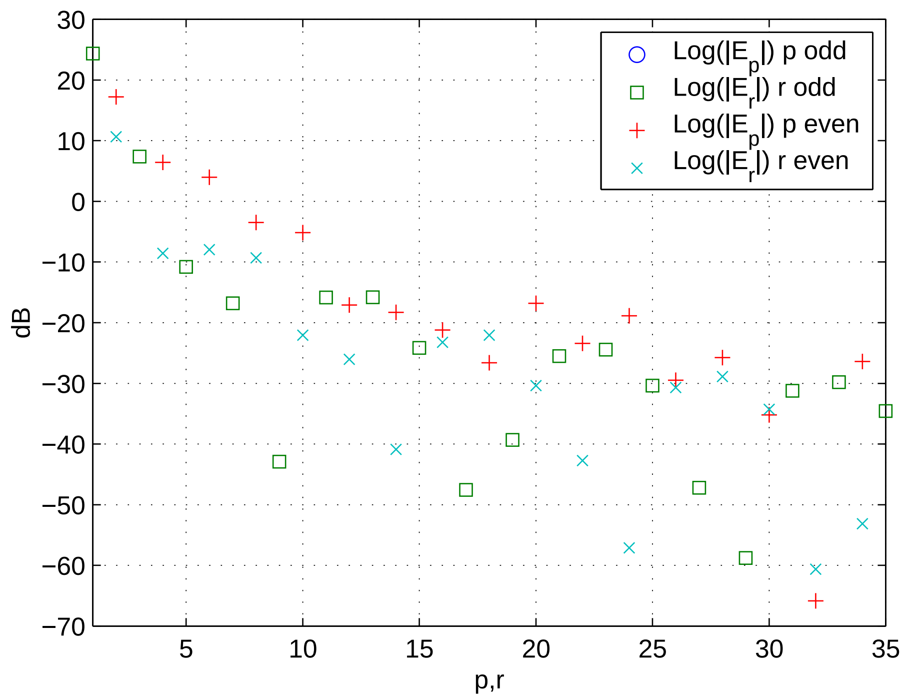

Figure 3.8.: Decay of expansion coefficients in case $x_0 = a/2$. $\log(|E_p|)$ is always below -100dB for p odd.

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.

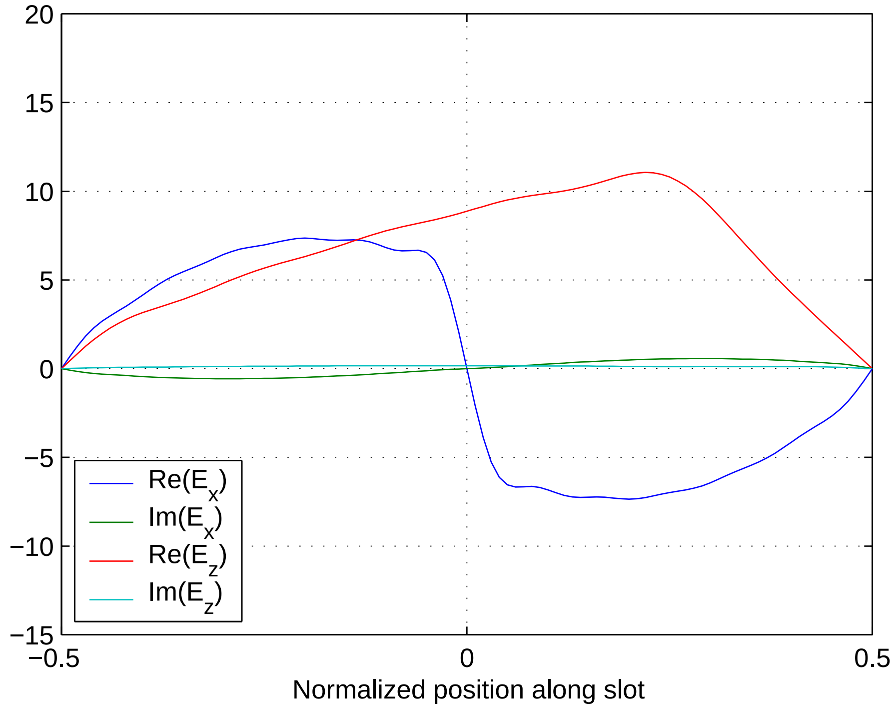

Figure 3.9.: Slot electric field distribution. $L_x = 32mm$.

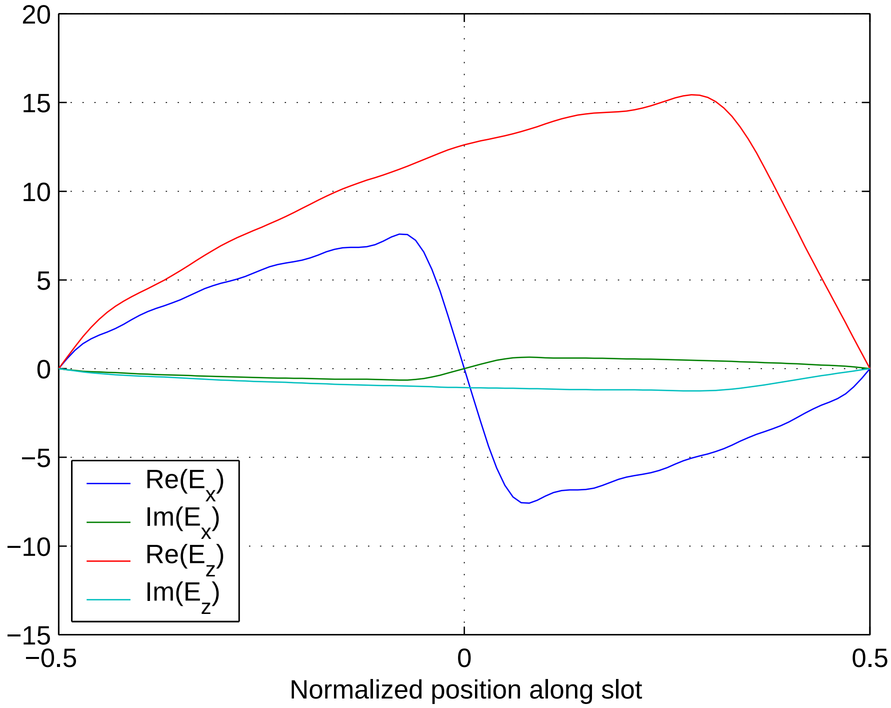

Figure 3.10.: Slot electric field distribution. $L_x = 44mm$.

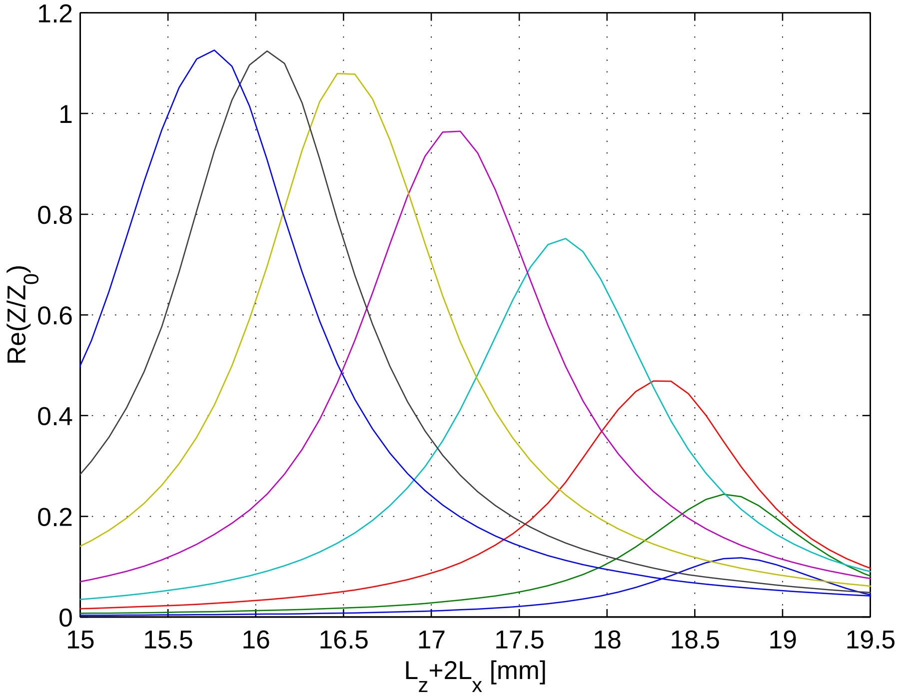

Figure 3.11.: T-slot resistance versus slot length L_z for eight slot lenghts L_x. From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

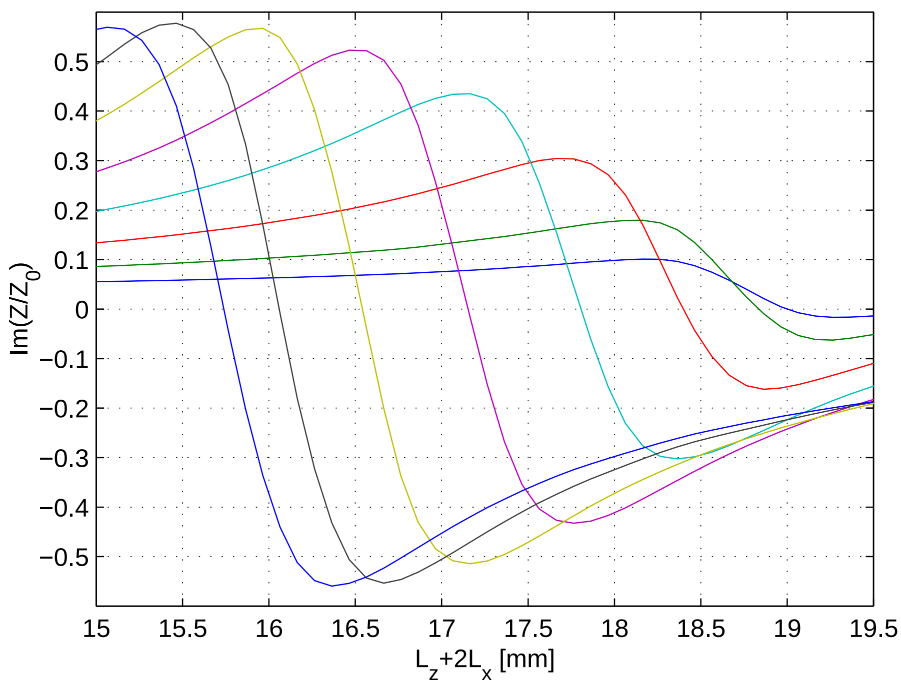

Figure 3.12.: T-slot reactance versus slot length L_z for eight slot lenghts L_x. From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

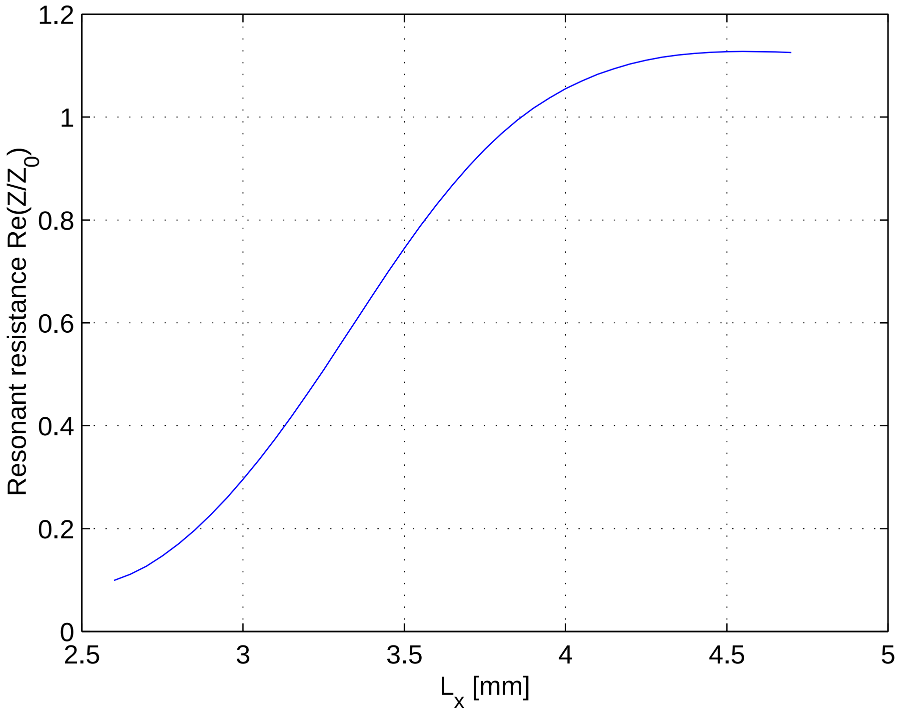

Figure 3.13.: T-slot resonant resistance versus slot length L_x

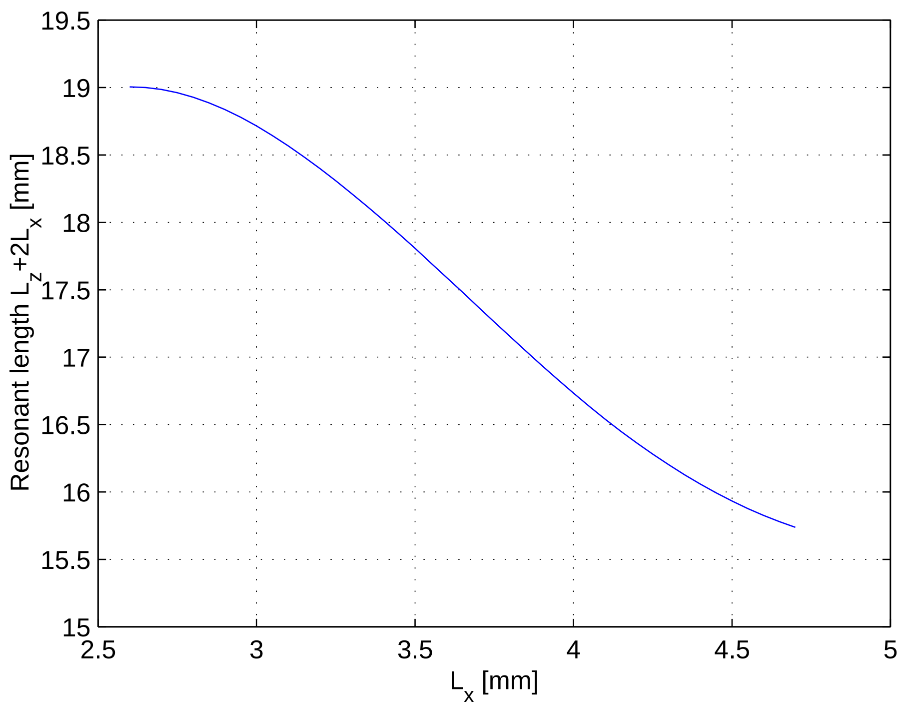

Figure 3.14.: T-slot resonant length $Lz + 2L_x$ versus slot length L_x

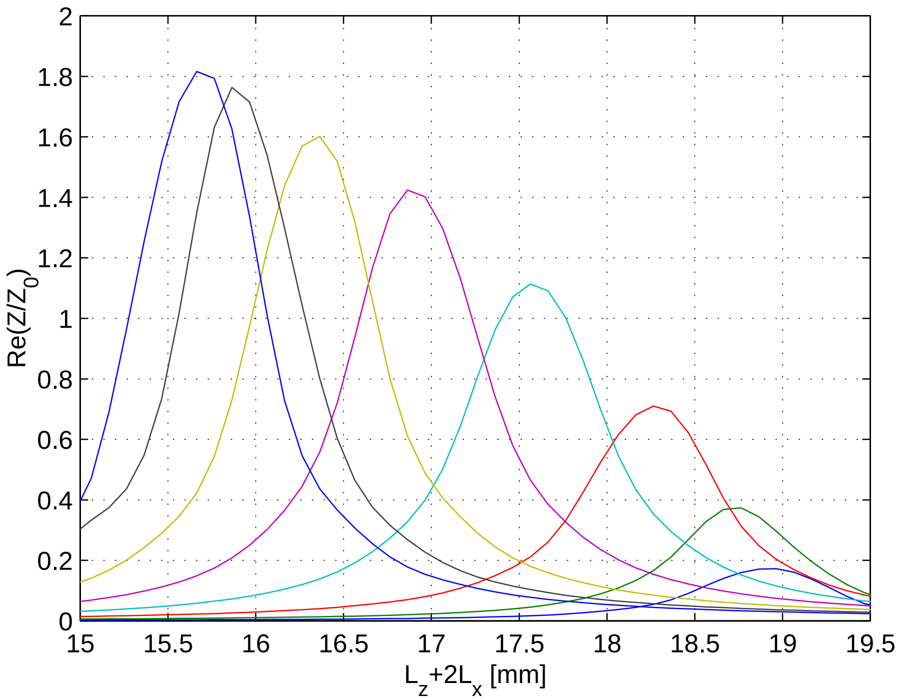

Figure 3.15.: T-slot resistance versus slot length L_z for eight slot lenghts L_x. From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

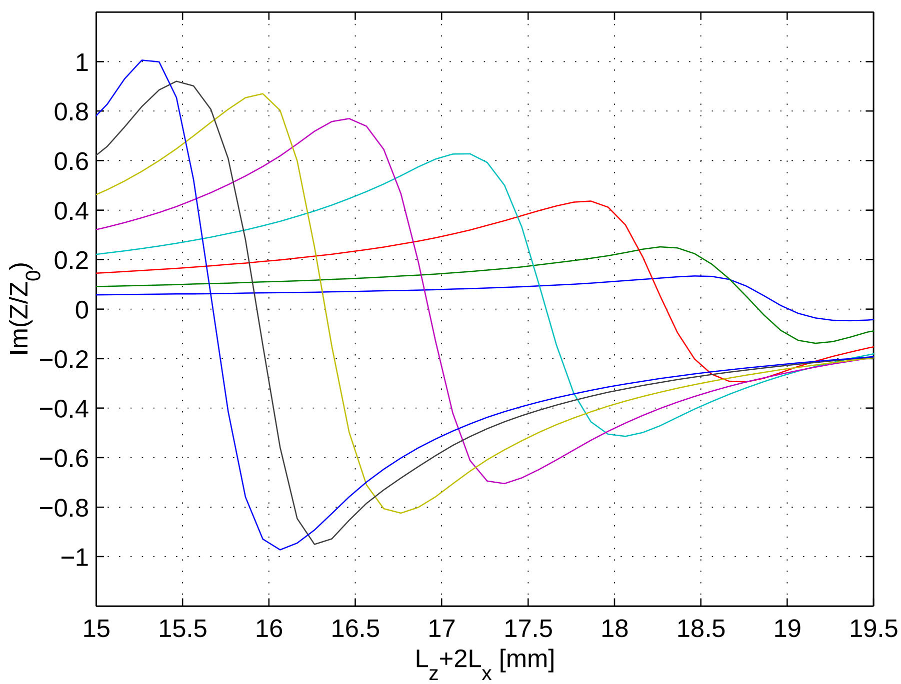

Figure 3.16.: T-slot reactance versus slot length L_z for eight slot lenghts L_x. From right to left: 2.6, 2.9, 3.2, 3.5, 3.8, 4.1, 4.4 and 4.7mm.

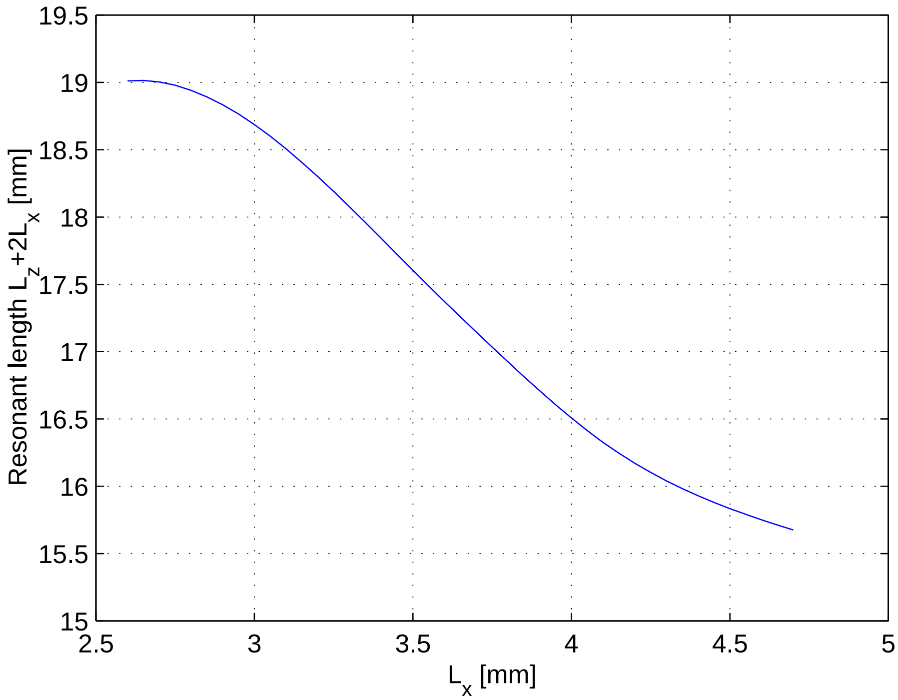

Figure 3.17.: T-slot resonant length $Lz + 2L_x$ versus slot length L_x

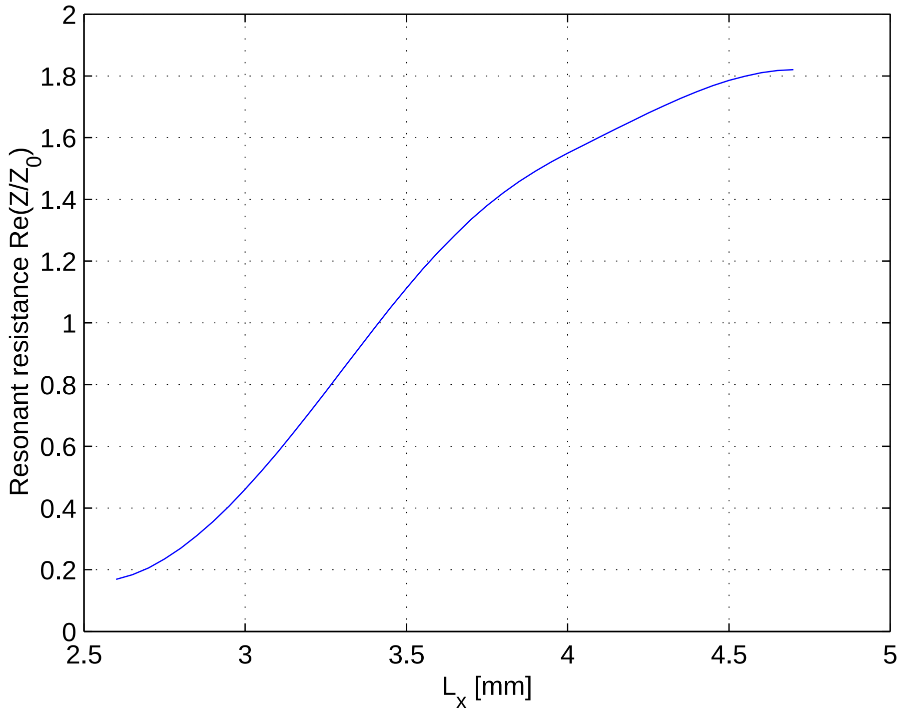

Figure 3.18.: T-slot resonant resistance versus slot length L_x

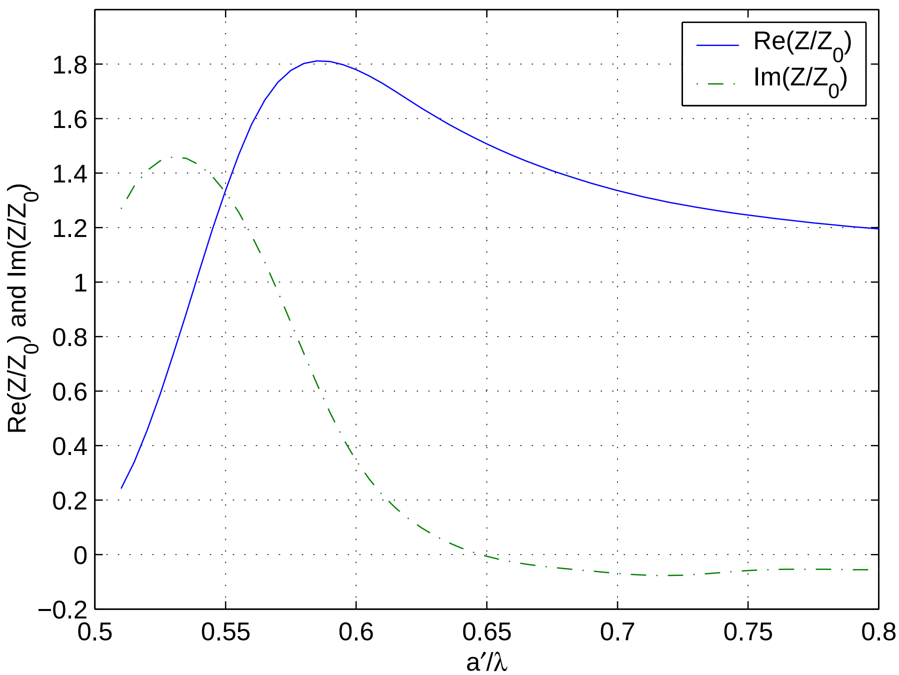

Figure 3.19.: T-slot impedance versus baffle spacing in free space wavelengths.

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.

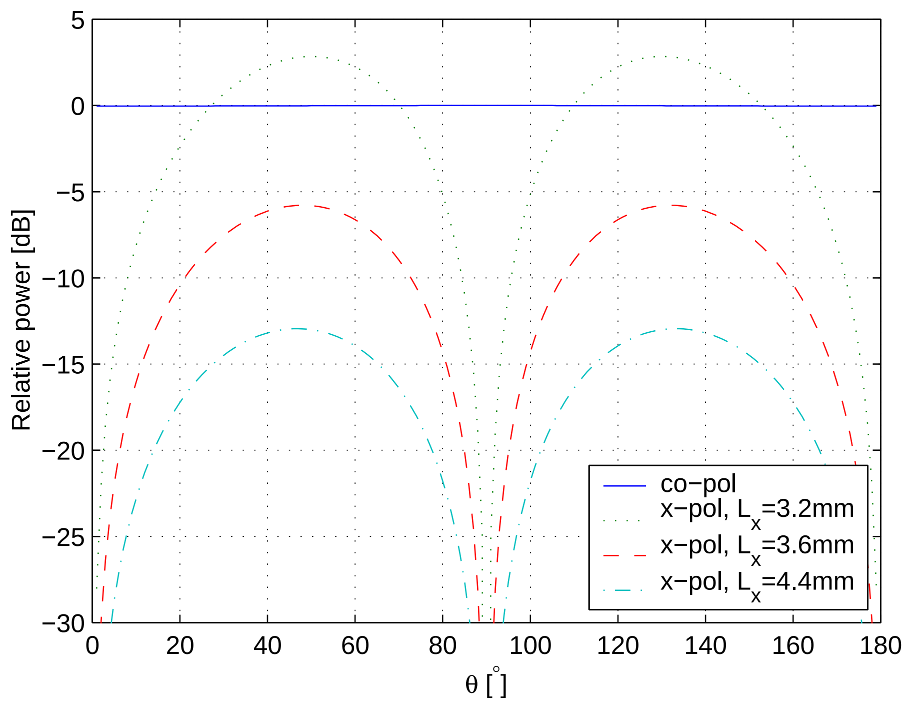

Figure 3.20.: E-plane patterns of a T-slot.

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 occuring at about

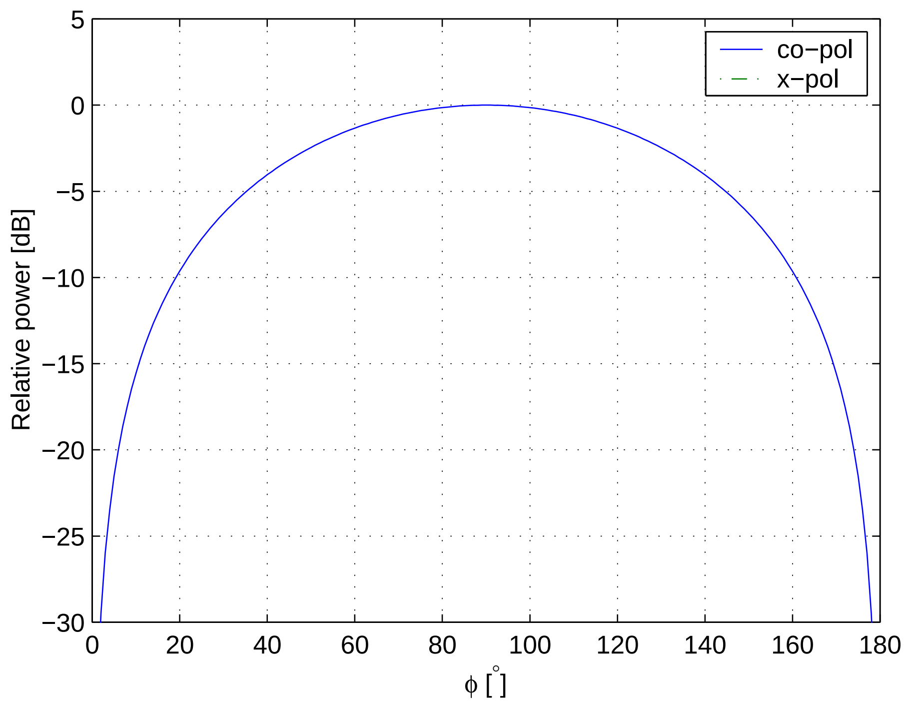

Figure 3.21.: H-plane patterns of a T-slot.

$\pm 40°$. Asumming only the first order mode is propagating in the PP waveguide, the theoretical cut-off angle becomes $\theta_c = 90 - \arcsin(\lambda_0/2a') = 40.3°$. 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]

$$D(\theta, \phi) \propto |f(\theta, \phi)|^2 \, |F(\theta, \phi)|^2 \tag{3.3}$$

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

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

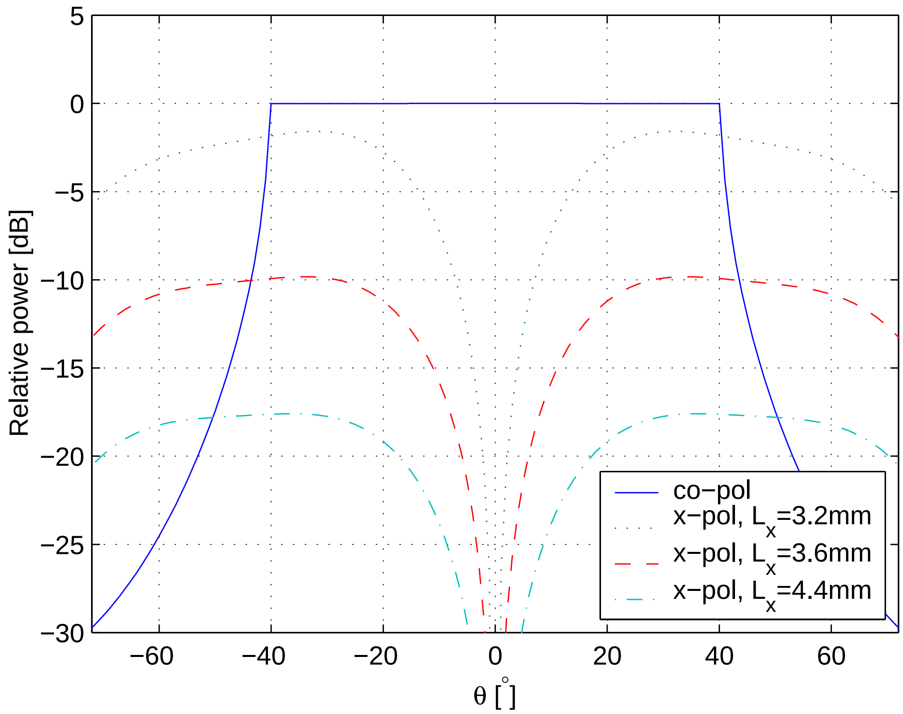

Figure 3.22.: E-plane patterns of a T-slot.

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

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

The results shows substatially suppression of grating lobes occurring at about $\pm 45°$, which indicates the high degree of spatial filtering obatined by application of the PP waveguide. Even better perfomance 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 tranverse 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 consective slots. However, it does mainly affect the position of the grating lobes in the cross polarized pattern.

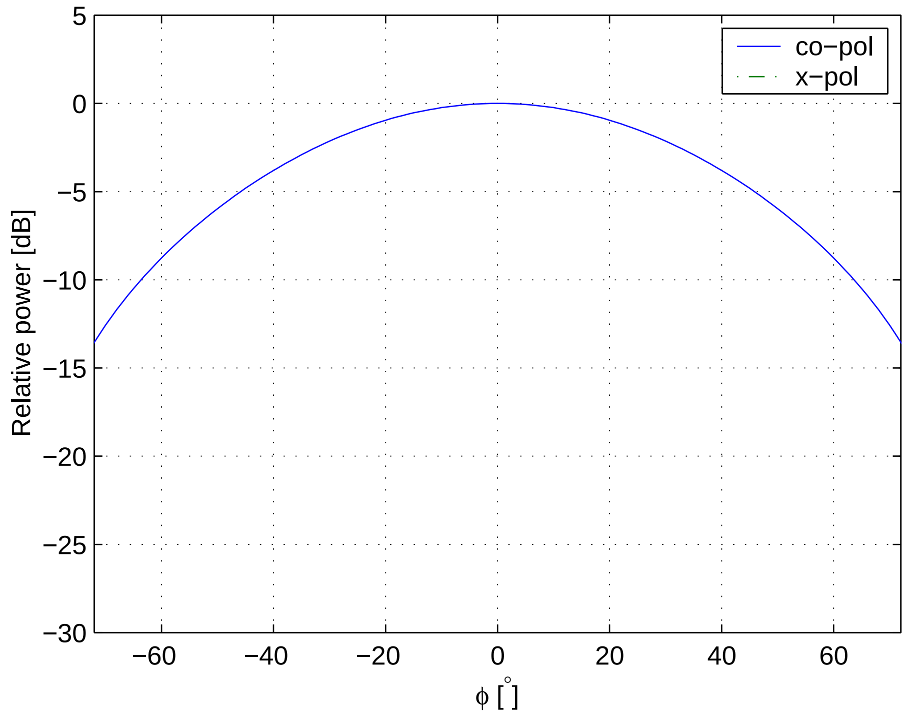

Figure 3.23.: H-plane patterns of a T-slot.

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 $\theta_c = 90 - \arcsin(\lambda_0/2a') = 34.9°$ giving better suppression of the grating lobe at $\pm 45°$.

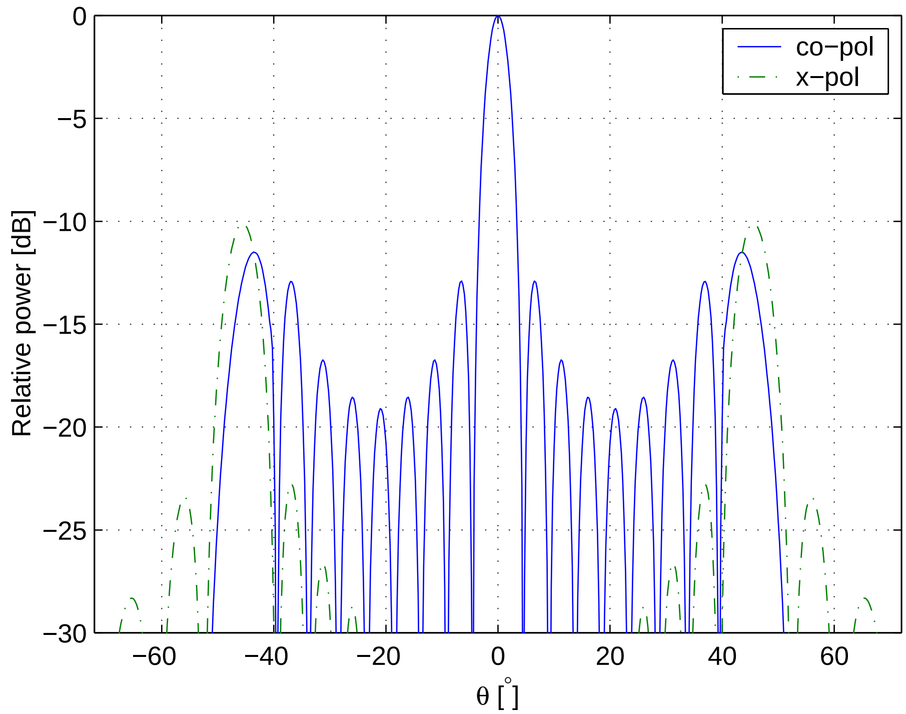

Figure 3.24.: E-plane patterns of a 9 element array of T-slots.

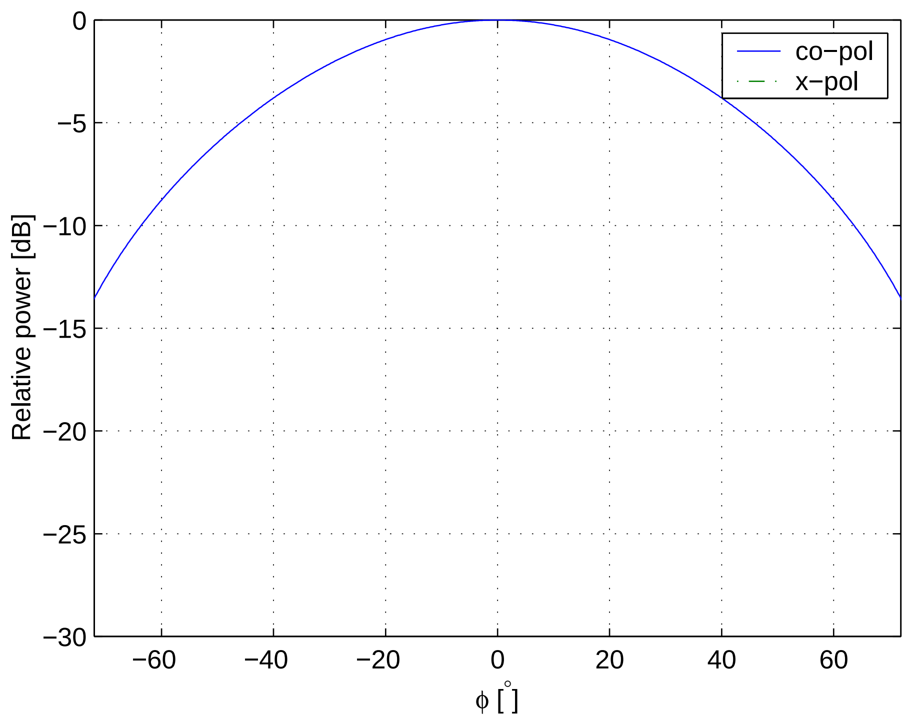

Figure 3.25.: H-plane patterns of a 9 element array of T-slots.

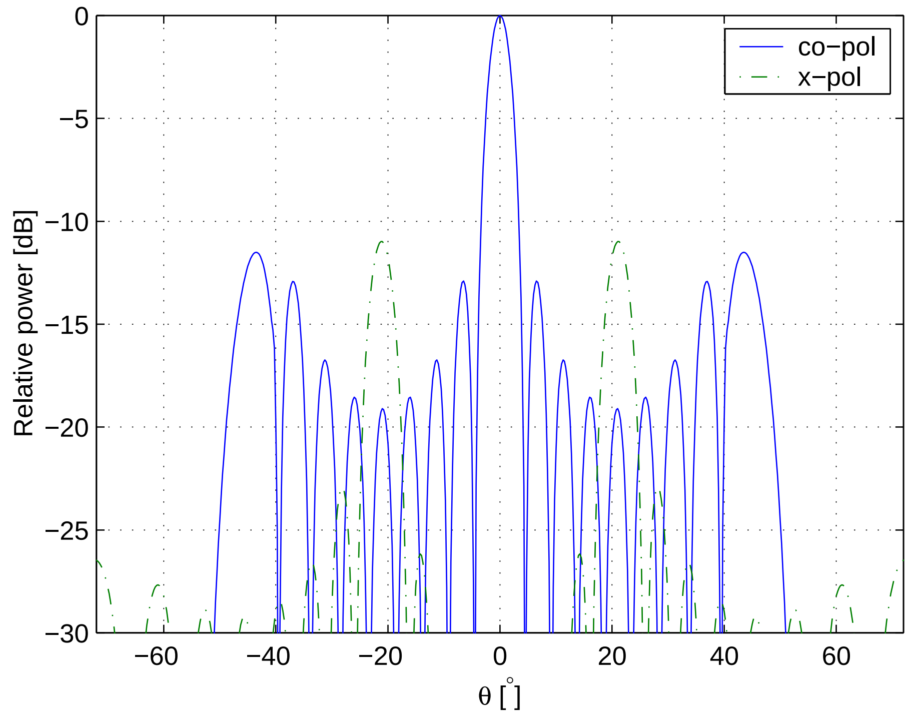

Figure 3.26.: E-plane patterns of a 9 element array of T-slots having alternating offset.

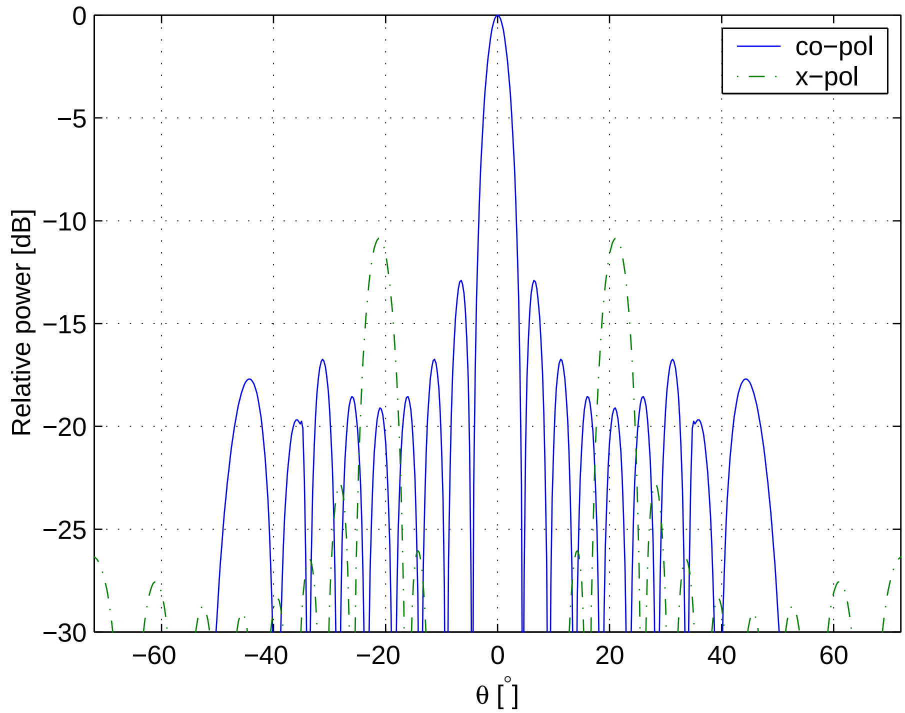

Figure 3.27.: E-plane patterns of a 9 element array of T-slots having alternating offset. PP spacing was $a' = 19.5mm$

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 mathmatical 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:

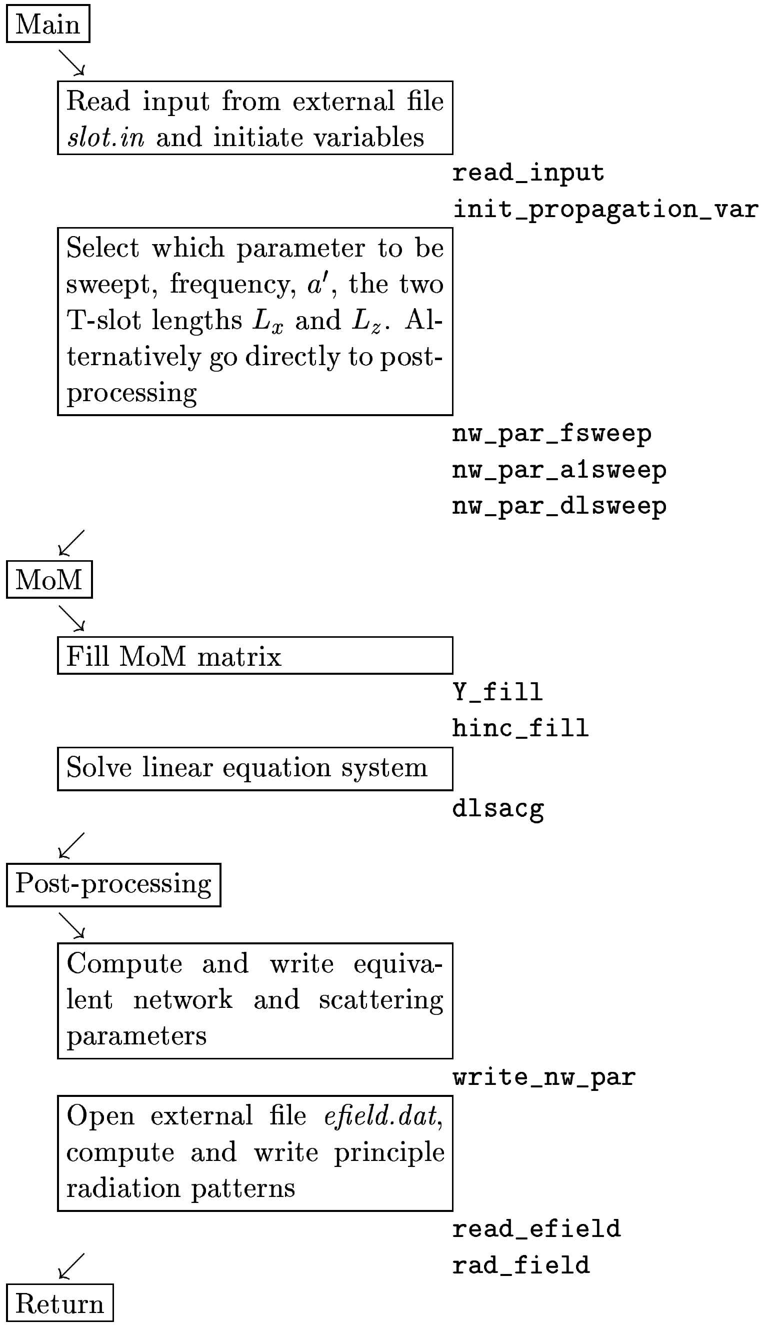

Figure 4.1.: Flowchart for MoM program.

  - `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 etimate 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

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

  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 $i = 1, \ldots, n$ the routine calculates a new estimat by doubling the number of points in the i-th direction, but halving the number immediately afterwards if the new etsimate 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 countour plots figure 3.14 and 3.13 mare 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 earliere 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 wavelenght. 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 ramdomized giving some perfomance 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.

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. Stellitano, "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. Forooraghi, "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

$$\mathbf{M} = -\hat{\mathbf{n}} \times \mathbf{E}_1 = -\hat{\mathbf{n}} \times \mathbf{E}^a \tag{A.1}$$

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

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

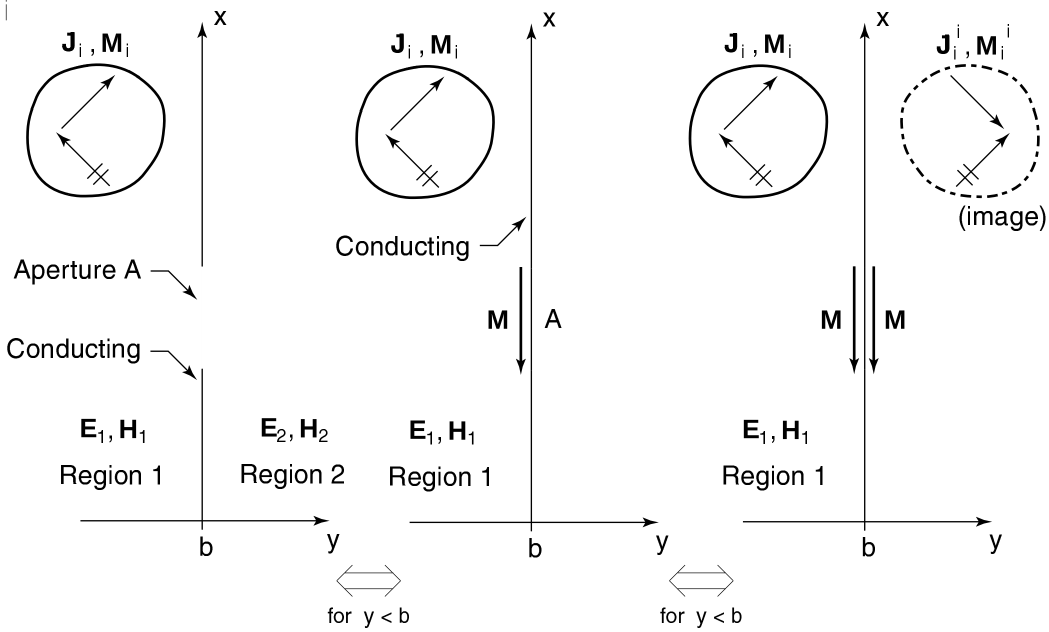

Figure A.1.: Equivalence for region 1 ($y < b$).

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 fileds 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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

in which the Green's function has the properties

- $G_1(\bar{R}, \bar{R}') = G_{c1}(\bar{R}, \bar{R}') + e^{-jkr(\bar{R},\bar{R}')}/4\pi r(\bar{R}, \bar{R}')$
- $\partial G_1(\bar{R}, \bar{R}')/\partial n' \equiv 0$ 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

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

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

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

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

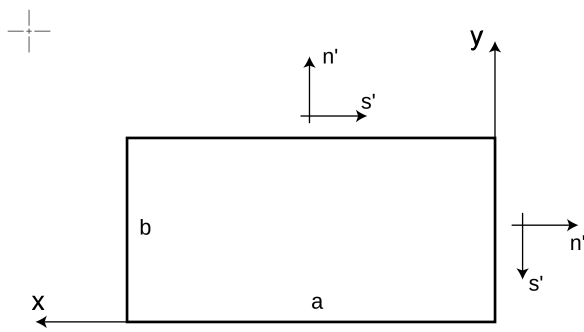

Figure B.1.: Definition of coordinates for rectangular waveguide.

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

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

Differentiation of eq. B.10 yields

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

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

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

By a similar procedure we can show that

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

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

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

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

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

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

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

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

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

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

$$\int_S \frac{\partial^2 E_z}{\partial s' \partial z'} G_1 dS' = \int_S \frac{\partial^2 G_1}{\partial s' \partial z'} E_z dS' \tag{B.18}$$

In a like manner, we can show that

$$\int_S \frac{\partial^2 E_s}{\partial z'^2} G_1 dS' = \int_S E_s \frac{\partial^2 G_1}{\partial z'^2} dS' \tag{B.19}$$

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 represenattion 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]

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

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

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

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

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

with γ_{rs}^2 defined as

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

Thus

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

where

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

Now using the method of undertimed coefficients [8]

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

Then

$$\frac{\partial f_{rs}}{\partial z'} = \frac{\partial v_1}{\partial z'}\mathrm{e}^{\gamma_{rs}z'} + \frac{\partial v_2}{\partial z'}\mathrm{e}^{-\gamma_{rs}z'} + \gamma_{rs}v_1\mathrm{e}^{\gamma_{rs}z'} - \gamma_{rs}v_2\mathrm{e}^{-\gamma_{rs}z'} \tag{B.27}$$

Letting

$$\frac{\partial v_1}{\partial z'}\mathrm{e}^{\gamma_{rs}z'} + \frac{\partial v_2}{\partial z'}\mathrm{e}^{-\gamma_{rs}z'} = 0 \tag{B.28}$$

we obtain

$$\frac{\partial^2 f_{rs}}{\partial z'^2} = \gamma_{rs}\frac{\partial v_1}{\partial z'}\mathrm{e}^{\gamma_{rs}z'} - \gamma_{rs}\frac{\partial v_2}{\partial z'}\mathrm{e}^{-\gamma_{rs}z'} + \gamma_{rs}^2 v_1\mathrm{e}^{\gamma_{rs}z'} + \gamma_{rs}^2 v_2\mathrm{e}^{-\gamma_{rs}z'} \tag{B.29}$$

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

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

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

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

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

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

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

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

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

Similarly

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

Thus

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

or

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

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

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

From this it follows that

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

with

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

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 than 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

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

in which

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

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

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

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

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

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)

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

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

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

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

where we have used the integral formula

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

and the quantity W_m is given by

$$W_m = \mathrm{sinc}\frac{m\pi w}{2a}\cos^2\frac{m\pi x_0}{a} \tag{C.4}$$

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

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

Thereby eq. C.2 reduces to

$$\begin{aligned} Y_{pq}^{int} \;=\; & \frac{-\lambda w}{jab\pi}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\epsilon_{mn}^2 W_m \int_{-L_z}^{L_z}\sin\left[\frac{q\pi}{2L_z}(L_z+\zeta)\right]\frac{k^2+\gamma_{mn}^2}{\gamma_{mn}} \\ & \cdot\int_{-L_z}^{L_z}\sin\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right]\mathrm{e}^{-\gamma_{mn}|\zeta-\zeta'|}d\zeta' - 2\sin\left[\frac{p\pi}{2L_z}(L_z+\zeta)\right]d\zeta \end{aligned} \tag{C.6}$$

Evaluating the inner integral we obtain the following expression

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

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)

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

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 funcions. Beyond the diagonal terms it is seen from symmetry, that it is only nessesary 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)

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

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

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

where

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

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

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

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

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

which is written as

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

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

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

and

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

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 extrated.

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)

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

Differentiation with respect to ξ' gives

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

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

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

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

where we have used the following integration scheme

$$\int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right] \sin\left[\frac{m\pi}{a}(x_{00}+\xi)\right] d\xi \tag{C.19}$$

$$= \frac{\frac{s\pi}{4L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2} \left[\sin\left[\frac{m\pi}{a}\left(\frac{sa}{m} - L_x - x_{00}\right)\right] \right.$$

$$\left. - \sin\left[\frac{m\pi}{a}\left(\frac{sa}{m} + L_x + x_{00}\right)\right] - 2\sin\left[\frac{m\pi}{a}(L_x - x_{00})\right]\right]$$

which by use of trigonometric identities reduces to

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

The last two ζ integrals evaluates to

$$\int_{-w/2}^{w/2} \delta(\zeta) \int_{-w/2}^{w/2} \mathrm{e}^{-\gamma_{mn}|\zeta-\zeta'|} d\zeta' d\zeta \tag{C.21}$$

$$= \int_{-w/2}^{w/2} \delta(\zeta) \left[\int_{-w/2}^{\zeta} \mathrm{e}^{-\gamma_{mn}(\zeta-\zeta')} d\zeta' + \int_{\zeta}^{w/2} \mathrm{e}^{\gamma_{mn}(\zeta-\zeta')} d\zeta' \right] d\zeta$$

$$= \int_{-w/2}^{w/2} \delta(\zeta) \frac{1}{\gamma_{mn}} \left[2 - \mathrm{e}^{-\gamma_{mn}w/2} \left(\mathrm{e}^{\gamma_{mn}\zeta} + \mathrm{e}^{-\gamma_{mn}\zeta} \right) \right] d\zeta$$

$$= \frac{2}{\gamma_{mn}} \left[1 - \mathrm{e}^{-\gamma_{mn}w/2} \right]$$

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

$$Y_{rs}^{int} = \frac{-4Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}^2} \frac{\frac{s\pi}{L_x}\frac{r\pi}{L_x} \left[\left(\frac{m\pi}{a}\right)^2 - k^2 \right] \left[1 - \mathrm{e}^{-\gamma_{mn}w/2} \right]}{\left[\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2 \right] \left[\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2 \right]}$$

$$\cdot \left[\begin{array}{c} \cos^2\frac{m\pi L_x}{a} \sin^2\frac{m\pi x_{00}}{a} \\ \cos^2\frac{m\pi x_{00}}{a} \sin^2\frac{m\pi L_x}{a} \end{array} \right]_{r,s \text{ even}}^{r,s \text{ odd}}$$

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)

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

$$= \frac{-Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x+\xi)\right] \delta(\zeta) \tag{C.23}$$

$$\cdot \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{r\pi}{2L_x}(L_x+\xi')\right] \left[k^2 + \frac{\partial^2}{\partial \xi'^2}\right] \frac{\mathrm{e}^{-jkR}}{R} d\xi' d\zeta' d\xi d\zeta$$

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

$$\int_{-L_x}^{L_x} \sin\left[\frac{r\pi}{2L_x}(L_x+\xi')\right] \left[k^2 + \frac{\partial^2}{\partial \xi'^2}\right] \frac{\mathrm{e}^{-jkR}}{R} d\xi'$$

$$= \int_{-L_x}^{L_x} \left[k^2 - \left(\frac{\pi r}{2L_x}\right)^2\right] \sin\left[\frac{r\pi}{2L_x}(L_x+\xi')\right] \frac{\mathrm{e}^{-jkR}}{R} d\xi'$$

$$- \frac{\pi r}{2L_x} \left[(-1)^r \frac{\mathrm{e}^{-jkR_1}}{R_1} - \frac{\mathrm{e}^{-jkR_2}}{R_2}\right] \tag{C.24}$$

where

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

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

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

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

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

$$\cdot \sin\left[\frac{r\pi}{2L_x}(L_x+\xi')\right]\frac{\mathrm{e}^{-jkR}}{R}d\xi'$$
$$-\frac{r\pi}{2L_x}\left[(-1)^r\frac{\mathrm{e}^{-jkR_1}}{R_1}-\frac{\mathrm{e}^{-jkR_2}}{R_2}\right]d\zeta' d\xi \tag{C.26}$$

which is written as

$$Y_{rs}^{ext} = \frac{-Z_0}{2\pi j\omega\mu_0}\int_{-w/2}^{w/2}\left[I_1+I_2\right]d\zeta' \tag{C.27}$$

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

$$Y_{rs}^{ext} = \frac{-2Z_0}{j\pi^2\omega\mu_0(s^2-r^2)}\Big[\left[(k2L_x)^2-(s\pi)^2\right]\cdot r\cdot y_1(s)$$
$$-\left[(k2L_z)^2-(r\pi)^2\right]\cdot s\cdot y_1(r)\Big] \qquad r\neq s \tag{C.28}$$

and

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

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

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

and

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

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

$$R = \sqrt{\zeta'^2+\nu^2 4L_x^2} \tag{C.32a}$$
$$R_1 = \sqrt{\zeta'^2+(\sigma-1/2)^2 4L_x^2} \tag{C.32b}$$
$$R_2 = \sqrt{\zeta'^2+(\sigma+1/2)^2 4L_x^2} \tag{C.32c}$$

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

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

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

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

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

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

$$\cdot \frac{\frac{m\pi}{a}\frac{r\pi}{L_x}}{\left(\frac{r\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2} \begin{bmatrix} \cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a} \end{bmatrix}^{r \text{ odd}}_{r \text{ even}}$$

$$\cdot \int_{-L_z}^{L_z}\int_{-w/2}^{w/2} \sin\left[\frac{q\pi}{2L_z}(L_z+\zeta)\right]\left[\pm \mathrm{e}^{-\gamma_{mn}|\zeta-\zeta'|}\right] d\zeta' d\zeta$$

The problem is now seperated 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

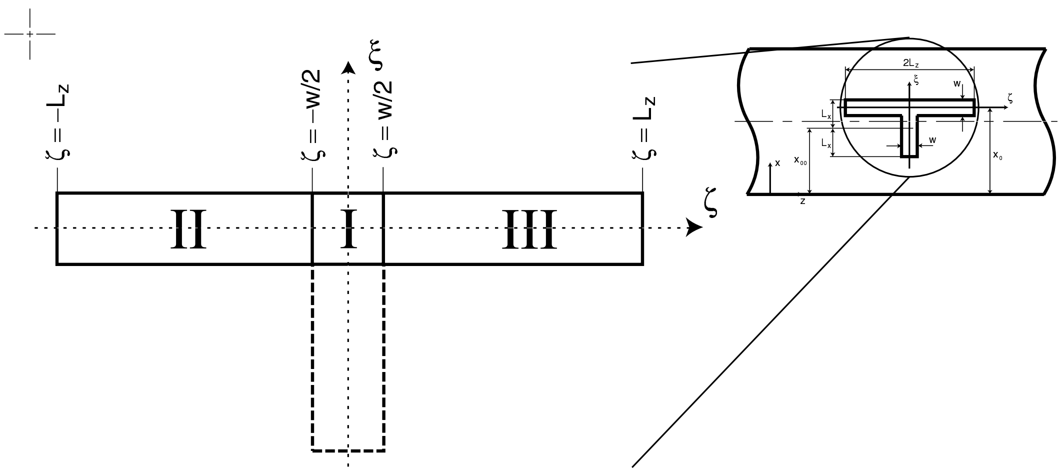

Figure C.1.: Seperating the weighting integral into three cases depending on the value of ζ.

$$\int_{-w/2}^{\zeta} \mathrm{e}^{-\gamma_{mn}(\zeta-\zeta')} d\zeta' - \int_{\zeta}^{w/2} \mathrm{e}^{\gamma_{mn}(\zeta-\zeta')} d\zeta' \tag{C.36}$$

$$= \frac{1}{\gamma_{mn}}\left[\mathrm{e}^{\gamma_{mn}(\zeta-w/2)} - \mathrm{e}^{-\gamma_{mn}(\zeta+w/2)}\right]$$

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

$$\int_{-w/2}^{w/2} \frac{1}{\gamma_{mn}} \sin\left[\frac{q\pi}{2L_z}(L_z+\zeta)\right]\left[\mathrm{e}^{\gamma_{mn}(\zeta-w/2)} - \mathrm{e}^{-\gamma_{mn}(\zeta+w/2)}\right] d\zeta \tag{C.37}$$

$$= \frac{1}{\gamma_{mn}} \frac{\frac{1}{L_z}\cos\frac{q\pi}{2}}{\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2}\left[2\gamma_{mn}L_z \sin\frac{q\pi w}{4L_z}\left[\mathrm{e}^{-\gamma_{mn}w}+1\right]\right.$$

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

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

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

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

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

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

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

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

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

$$\begin{aligned} Y_{rq}^{int,2} = & \frac{-2Z_0}{j\omega\mu_0 ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{\epsilon_{mn}^2}{\gamma_{mn}}\cos\frac{m\pi x_0}{a}\frac{\frac{m\pi}{a}\frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2-\left(\frac{m\pi}{a}\right)^2\right]} \\ & \cdot\frac{\frac{1}{2L_z}}{\left[\gamma_{mn}^2+\left(\frac{q\pi}{2L_z}\right)^2\right]}\left[\begin{array}{c}\cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a}\end{array}\right]_{r \text{ even}}^{r \text{ odd}} \\ & \cdot\left[\left[1-\mathrm{e}^{-\gamma_{mn}w}\right]\left[q\pi\cos\left[\frac{q\pi}{4L_z}(2L_z-w)\right]-2\gamma_{mn}L_z\right.\right. \\ & \left.\left.\cdot\sin\left[\frac{q\pi}{4L_z}(2L_z-w)\right]\right]+q\pi\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)}-\mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]\right] \end{aligned} \quad (C.41)$$

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

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

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

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

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

$$Y_{rq}^{int,3} = \frac{-2Z_0}{j\omega\mu_0 ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{\epsilon_{mn}^2}{\gamma_{mn}}\cos\frac{m\pi x_0}{a}\frac{\frac{m\pi}{a}\frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2-\left(\frac{m\pi}{a}\right)^2\right]} \tag{C.44}$$

$$\cdot\frac{\frac{1}{2L_z}}{\left[\gamma_{mn}^2+\left(\frac{q\pi}{2L_z}\right)^2\right]}\left[\begin{array}{c}\cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a}\\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a}\end{array}\right]_{r\text{ even}}^{r\text{ odd}}$$

$$\cdot\left[\left[1-\mathrm{e}^{-\gamma_{mn}w}\right]\left[q\pi\cos\left[\frac{q\pi}{4L_z}(2L_z+w)\right]+2\gamma_{mn}L_z\right.\right.$$

$$\left.\left.\cdot\sin\left[\frac{q\pi}{4L_z}(2L_z+w)\right]\right]+q\pi(-1)^q\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)}-\mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]\right]$$

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

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

$$\left.-2\gamma_{mn}L_z\sin\left[\frac{q\pi}{4L_z}(2L_z-w)\right]\right]+q\pi\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)}-\mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]$$

$$+\left[1-\mathrm{e}^{-\gamma_{mn}w}\right]\left[q\pi\cos\left[\frac{q\pi}{4L_z}(2L_z+w)\right]\right.$$

$$\left.+2\gamma_{mn}L_z\sin\left[\frac{q\pi}{4L_z}(2L_z+w)\right]\right]+q\pi(-1)^q\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)}-\mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]$$

which by use of trigonometric identities is rewritten as

$$2\cos\frac{q\pi}{2}\left[1-\mathrm{e}^{-\gamma_{mn}w}\right]\left[q\pi\cos\frac{q\pi w}{4L_z}+2\gamma_{mn}L_z\sin\frac{q\pi w}{4L_z}\right]$$

$$+q\pi\left[1+(-1)^q\right]\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)}-\mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]$$

this joint expression becomes

$$Y_{rq}^{int,23} = \frac{-2Z_0}{j\omega\mu_0 ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\frac{\epsilon_{mn}^2}{\gamma_{mn}}\cos\frac{m\pi x_0}{a}\frac{\frac{m\pi}{a}\frac{r\pi}{L_x}}{\left[\left(\frac{r\pi}{2L_x}\right)^2-\left(\frac{m\pi}{a}\right)^2\right]} \tag{C.46}$$

$$\cdot \frac{\frac{1}{L_z}}{\left[\gamma_{mn}^2 + \left(\frac{q\pi}{2L_z}\right)^2\right]} \left[\begin{array}{c} \cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a} \end{array} \right]^{r \text{ odd}}_{r \text{ even}}$$

$$\cdot \left[\cos\frac{q\pi}{2}\left[1-\mathrm{e}^{-\gamma_{mn}w}\right]\left[q\pi\cos\frac{q\pi w}{4L_z} + 2\gamma_{mn}L_z\sin\frac{q\pi w}{4L_z}\right]\right.$$
$$\left. +q\pi\left[\mathrm{e}^{-\gamma_{mn}(L_z+w/2)} - \mathrm{e}^{-\gamma_{mn}(L_z-w/2)}\right]\right]$$

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

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

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

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

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

Evaluating the ξ integral gives the following computational efficient double integral

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

where

$$R_1 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 - \zeta)^2} \tag{C.51a}$$

$$R_2 = \sqrt{(\xi' + w/2 - L_x)^2 + (w/2 + \zeta)^2} \tag{C.51b}$$

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

$$\begin{aligned} Y_{ps}^{int} &= \langle H_x^{int}(E_x), w_s \rangle \frac{Z_0}{E_p} \qquad \text{(C.52)} \\ &= \frac{-2Z_0}{j\omega\mu_0 ab} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \frac{\epsilon_{mn}^2}{\gamma_{mn}} \\ &\quad \cdot \sin\left[\frac{m\pi}{a}(x_{00} + \xi)\right] \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \\ &\quad \cdot \frac{\partial^2}{\partial\xi'\partial\zeta'} \sin\left[\frac{m\pi}{a}(x_0 + \xi')\right] \mathrm{e}^{-\gamma_{mn}|\zeta - \zeta'|} d\zeta' d\xi' d\xi d\zeta \end{aligned}$$

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

$$\begin{aligned} Y_{ps}^{int} &= \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{m\pi}{a} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \qquad \text{(C.53)} \\ &\quad \cdot \sin\left[\frac{m\pi}{a}(x_{00} + \xi)\right] \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \\ &\quad \cdot \cos\left[\frac{m\pi}{a}(x_0 + \xi')\right] \left[\pm \mathrm{e}^{-\gamma_{mn}|\zeta - \zeta'|}\right] d\zeta' d\xi' d\xi d\zeta \end{aligned}$$

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

$$Y_{ps}^{int} = \frac{-2Z_0}{j\omega\mu_0 ab} \sum_{m=0}^{\infty} \sum_{n=0}^{\infty} \epsilon_{mn}^2 \frac{\frac{m\pi}{a}\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{m\pi}{a}\right)^2} \tag{C.54}$$

$$\cdot \begin{bmatrix} \cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a} \end{bmatrix}_{s\text{ even}}^{s\text{ odd}} \int_{-w/2}^{w/2} \cos\left[\frac{m\pi}{a}(x_0+\xi')\right]$$

$$\cdot \int_{-w/2}^{w/2} \delta(\zeta) \int_{-L_z}^{L_z} \sin\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right] \left[\pm \mathrm{e}^{-\gamma_{mn}|\zeta-\zeta'|}\right] d\zeta' d\zeta d\xi'$$

Evaluating the ξ' integral and expanding the inner most intergral gives

$$Y_{ps}^{int} = \frac{-4Z_0}{j\omega\mu_0 ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\epsilon_{mn}^2 \frac{\frac{s\pi}{L_x}\sin\frac{m\pi w}{2a}\cos\frac{m\pi x_0}{a}}{\left(\frac{s\pi}{2L_x}\right)^2-\left(\frac{m\pi}{a}\right)^2} \tag{C.55}$$

$$\cdot \begin{bmatrix} \cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a} \end{bmatrix}_{s\text{ even}}^{s\text{ odd}} \int_{-w/2}^{w/2} \delta(\zeta)$$

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

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

where we have used the integral formulation given in eq. C.3. Interchanging the ζ' and ζ integrals and directly evaluation of the latter, these integrals reduces to the following expression

$$\int_{-L_z}^{\zeta} \sin\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right] \mathrm{e}^{\gamma_{mn}\zeta'} d\zeta' - \int_{\zeta}^{L_z} \sin\left[\frac{p\pi}{2L_z}(L_z+\zeta')\right] \mathrm{e}^{-\gamma_{mn}\zeta'} d\zeta' \tag{C.56}$$

which evaluates to

$$\frac{\frac{p\pi}{2L_z}}{\gamma_{mn}^2+\left(\frac{p\pi}{2L_z}\right)^2}\left[\mathrm{e}^{-\gamma_{mn}L_z}\left(1+(-1)^p\right)-2\cos\frac{p\pi}{2}\right] \tag{C.57}$$

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

$$Y_{ps}^{int} = \frac{-4Z_0}{j\omega\mu_0 ab}\sum_{m=0}^{\infty}\sum_{n=0}^{\infty}\epsilon_{mn}^2 \frac{\frac{p\pi}{L_z}\frac{s\pi}{L_x}\sin\frac{m\pi w}{2a}\cos\frac{m\pi x_0}{a}}{\left[\gamma_{mn}^2+\left(\frac{p\pi}{2L_z}\right)^2\right]\left[\left(\frac{s\pi}{2L_x}\right)^2-\left(\frac{m\pi}{a}\right)^2\right]} \tag{C.58}$$

$$\cdot \left[\mathrm{e}^{-\gamma_{mn}L_z}-\cos\frac{p\pi}{2}\right] \begin{bmatrix} \cos\frac{m\pi L_x}{a}\sin\frac{m\pi x_{00}}{a} \\ -\cos\frac{m\pi x_{00}}{a}\sin\frac{m\pi L_x}{a} \end{bmatrix}_{s\text{ even}}^{s\text{ odd}}$$

$Y_{rs}^{int} = 0$ 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

$$Y_{ps}^{ext} = \left\langle H_x^{ext}(E_x), w_s \right\rangle \frac{Z_0}{E_p} \tag{C.59}$$

$$= \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \tag{C.60}$$

$$\cdot \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \frac{\partial^2}{\partial\xi'\partial\zeta'} \frac{\mathrm{e}^{-jkR}}{R} d\zeta' d\xi' d\xi d\zeta$$

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

$$Y_{ps}^{ext} = \frac{Z_0}{2\pi j\omega\mu_0} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \tag{C.61}$$

$$\cdot \frac{-p\pi}{2L_z} \int_{-L_z}^{L_z} \cos\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \left. \frac{\mathrm{e}^{-jkR}}{R} \right|_{\xi'=-w/2}^{w/2} d\zeta' d\xi d\zeta$$

Evaluating the ζ integral gives a computational efficient double integral

$$Y_{ps}^{ext} = \frac{-Z_0 p}{4j\omega\mu_0 L_z} \int_{-L_x}^{L_x} \int_{-L_z}^{L_z} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \tag{C.62}$$

$$\cdot \cos\left[\frac{p\pi}{2L_z}(L_z + \zeta')\right] \left[\frac{\mathrm{e}^{-jkR_1}}{R_1} - \frac{\mathrm{e}^{-jkR_2}}{R_2}\right] d\zeta' d\xi$$

where

$$R_1 = \sqrt{(L_x - \xi)^2 + \zeta'^2} \tag{C.63a}$$

$$R_2 = \sqrt{(L_x - w - \xi)^2 + \zeta'^2} \tag{C.63b}$$

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

$$\begin{aligned} h_q^{inc} &= \langle H_z^{inc}, w_q \rangle \qquad \text{(C.64)} \\ &= j \int_{-w/2}^{w/2} \int_{-L_z}^{L_z} \sin\left[\frac{q\pi}{2L_z}(\zeta + L_z)\right] \delta(\xi) \cos\left[\frac{\pi}{a}(x_0 + \xi)\right] \mathrm{e}^{-j\beta_{10}\zeta} d\zeta d\xi \end{aligned}$$

Using the more general integral formula

$$\int_{-L}^{L} \sin\left[\frac{p\pi}{2L}(x + L)\right] \mathrm{e}^{-jkx} dx = \frac{\frac{p\pi}{L}}{\left(\frac{p\pi}{2L}\right)^2 - k^2} \cdot \begin{cases} \cos(kL) & p \text{ odd} \\ j\sin(kL) & p \text{ even} \end{cases} \qquad \text{(C.65)}$$

eq. C.65 becomes

$$h_q^{inc} = \frac{\frac{q\pi}{L_z}}{\left(\frac{q\pi}{2L_z}\right)^2 - \beta_{10}^2} \cos\frac{\pi x_0}{a} \cdot \begin{cases} j\cos(\beta_{10}L_z) & q \text{ odd} \\ -\sin(\beta_{10}L_z) & q \text{ even} \end{cases} \qquad \text{(C.66)}$$

Similarly we get the following driving term for the x-component

$$\begin{aligned} h_s^{inc} &= \langle H_x^{inc}, w_s \rangle \qquad \text{(C.67)} \\ &= \frac{-\beta_{10}}{\pi/a} \int_{-w/2}^{w/2} \int_{-L_x}^{L_x} \sin\left[\frac{s\pi}{2L_x}(L_x + \xi)\right] \delta(\zeta) \sin\left[\frac{\pi}{a}(x_{00} + \xi)\right] \mathrm{e}^{-j\beta_{10}\zeta} d\xi d\zeta \end{aligned}$$

Evaluating this integral and applying various trigometric identities it reduces to the following final equation

$$h_s^{inc} = \frac{-\beta_{10}}{\pi/a} \frac{\frac{s\pi}{L_x}}{\left(\frac{s\pi}{2L_x}\right)^2 - \left(\frac{\pi}{a}\right)^2} \cdot \begin{cases} \cos\frac{\pi L_x}{a} \sin\frac{\pi x_{00}}{a} & s \text{ odd} \\ -\cos\frac{\pi x_{00}}{a} \sin\frac{\pi L_x}{a} & s \text{ even} \end{cases} \qquad \text{(C.68)}$$

Appendix D

Source Code

A list of the implemented FORTRAN90 program including all subroutines and functions.

```
!##############################################################################
!
!  Name:
!           main
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Reads input file slot.in and initiates simulation.
!
!  Usage:
!           main()
!
!  Modules:
!           gen_var
!
!  Subroutines:
!           read_input,init_propagation_var,nw_par_fsweep,nw_par_lsweep,
!           nw_par_dlsweep,nw_par_a1sweep,rad_field,rad_field_ppwg
!
program main
    !modules
    use gen_var, only : freq,freq_0,sw_type
    !implicit statement
    implicit none

    call read_input()
    freq=freq_0
    call init_propagation_var()

    if(sw_type.eq.1) then
        call nw_par_fsweep()
    elseif(sw_type.eq.2) then
        call nw_par_lsweep()
    elseif(sw_type.eq.3) then
```

```
            call nw_par_dlsweep()
        elseif(sw_type.eq.4) then
            call nw_par_a1sweep()
        elseif(sw_type.eq.5) then
            call  rad_field(1)
            call  rad_field(2)
        else
            write(*,*) 'Wrong type of sweep selected!'
        endif
    end

    !##############################################################################
    !
    !  Name:
    !           nw_par_a1sweep
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           This subroutine performs a sweep over the PP waveguide a'
    !           dimension computing the equivalent network parameters for a
    !           longitudinal, transverse or T-slot.
    !
    !  Usage:
    !           call nw_par_a1sweep()
    !
    !  Arguments:
    !
    !  Modules:
    !           slot_var,mom_var,gen_var,ppwg_var
    !
    !  Subroutines:
    !           read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
    !
    !  IMSL subroutines:
    !           dlsacg
    !
    !  Remarks:
    !           The output is printed to standard output.
    !
    subroutine nw_par_a1sweep()
        !modules
        use slot_var, only  : w,Lx,Lx0,Lz,Lz0,x0,x00,wp
        use mom_var, only   : Nz,Nx,Y,h_inc,E
        use gen_var, only   : freq_0,freq
        use ppwg_var, only  : a1
        !implicit statement
        implicit none
        !list of local variables
        integer     ::  i

        Lx=Lx0
        Lz=Lz0
        x00=x0+w/2.0-Lx
        do i= 0,10
            a1=16.0001E-3_wp+i*1.0E-3_wp
            call Y_fill()
            call hinc_fill()
```

```
        call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
        call write_nw_par()
    enddo
return
end

!###############################################################################
!
!  Name:
!           nw_par_fsweep
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine performs a frequency sweep computing the
!           equivalent network parameters for a longitudinal, transverse or
!           T-slot.
!
!  Usage:
!           call nw_par_fsweep()
!
!  Arguments:
!
!  Modules:
!           slot_var,mom_var,gen_var,wg_var
!
!  Subroutines:
!           read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
!
!  IMSL subroutines:
!           dlsacg
!
!  Remarks:
!           The output is printed to standard output.
!
subroutine nw_par_fsweep()
    !modules
    use slot_var, only  : w,Lx,Lx0,Lz,Lz0,x0,x00,wp
    use mom_var, only   : Nz,Nx,Y,h_inc,E
    use gen_var, only   : freq_0,freq,nfreq,dfreq
    use wg_var, only    : a
    !implicit statement
    implicit none
    !list of local variables
    integer     ::  i

    !round end correction
    !Lx=Lx0-0.25*0.215*w    !for t-slot
    !Lx=Lx0-0.1075_wp*w     !for transverse slot
    !Lz=Lz0-0.1075_wp*w     !for longitudinal and t-slot
    !no round end correction
    Lx=Lx0
    Lz=Lz0

    x00=x0+w/2.0-Lx
    do i= 0,nfreq
        freq=freq_0+(i-nfreq/2.0_wp)*dfreq
        call init_propagation_var()
```

```
        call Y_fill()
        call hinc_fill()
        call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
        call write_nw_par()
    enddo
return
end

!############################################################################
!
!  Name:
!           nw_par_dlsweep
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine performs a sweep over both T-slot dimensions
!           Lx and Lz computing the equivalent network parameters.
!
!  Usage:
!           call nw_par_dlsweep()
!
!  Arguments:
!
!  Modules:
!           slot_var,mom_var,gen_var
!
!  Subroutines:
!           read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
!
!  IMSL subroutines:
!           dlsacg
!
!  Remarks:
!           The output is printed to standard output.
!
subroutine nw_par_dlsweep()
    !modules
    use slot_var, only  : w,Lz,Lz0,Lx,Lx0,x0,x00,wp
    use mom_var, only   : Nz,Nx,Y,h_inc,E
    use gen_var, only   : freq_0,freq,lambda_0,nll,dll,ntl,dtl
    !implicit statement
    implicit none
    !list of local variables
    integer     ::  i,j

    write(*,1076) (i,i=1,Nz),(j,j=1,Nx)
    do i= 3,3 !0,ntl
        Lx=Lx0+(i-ntl/2.0_wp)*dtl
        Lz0=0.48*lambda_0-2.0_wp*Lx+w/2.0_wp
        x00=x0+w/2.0_wp-Lx
        do j=3,3 !0,nll
            Lz=Lz0+(3+j-nll/2)*dll*lambda_0
            call Y_fill()
            call hinc_fill()
            call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
            call write_nw_par()
        enddo
```

```
    enddo

1076 format(9x,'freq[Hz]',12x,'x0[m]',12x,'a1[m]',&
            12x,'Lx[m]',12x,'Lz[m]',13x,'z_re',13x,'z_im',&
            13x,'R_re',13x,'R_im',13x,'T_re',13x,'T_im',&
             24(16x,'Ex(',I2.2,')',12x),13(16x,'Ez(',I2.2,')',12x))
return
end

!##########################################################################
!
!  Name:
!           nw_par_lsweep
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine is used to investigate a longitudinal slot
!           radiating into either a half space or a PP waveguide. Performs a
!           sweep over both slot length and offset from centerline computing
!           the equivalent network parameters.
!
!  Usage:
!           call nw_par_lsweep()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,mom_var,gen_var,ppwg_var,wg_var
!
!  Subroutines:
!           read_input,init_propagation_var,Y_fill,hinc_fill,write_nw_par
!
!  IMSL subroutines:
!           dlsacg
!
!  Remarks:
!           The output is printed to standard output.
!
subroutine nw_par_lsweep()
    !modules
    use const, only     : mu_0,pi,cj,Z0,wp
    use slot_var, only  : w,Lz,Lz0,Lx,Lx0,x0,x00
    use mom_var, only   : Nz,Nx,Y,h_inc,E
    use gen_var, only   : freq_0,freq,lambda_0,nll,dll,ntl,dtl
    use wg_var, only    : a
    use ppwg_var, only  : a1
    !implicit statement
    implicit none
    !list of local variables
    integer     ::  i,j

    Lz0=lambda_0/4.0_wp
    do i=0,8  !offset sweep
        x0=a/2.0_wp+2.0E-3_wp+i*1.0E-3_wp
        do j= 0,nll !length sweep
            Lz=Lz0*(0.94_wp+dll*j/nll)
            call Y_fill()
```

```
            call hinc_fill()
            call dlsacg(Nz+Nx,Y,Nz+Nx,h_inc,1,E)
            call write_y_norm()
        enddo
    enddo
return
end
```

```
!##############################################################################
!
!  Name:
!           const
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           module containing all general constants
!
!  Usage:
!           use  const
!
module const
    !list of parameters
    !kind of double
    integer,parameter       :: wp    = kind(1.0D0)
    !pi
    real(wp),   parameter   :: pi    = 3.14159265359_wp
    !degrees2radians
    real(wp),   parameter   :: dtr   = 1.74532925199E-2_wp
    !radians2degrees
    real(wp),   parameter   :: rtd   = 57.2957795132_wp
    !inch2meter
    real(wp),   parameter   :: itm   = 25.4000000E-3_wp
    !meter2inch
    real(wp),   parameter   :: mti   = 39.3700787_wp
    !speed of light in vaccum
    real(wp),   parameter   :: c     = 2.997925E8_wp
    !free space permittivity
    real(wp),   parameter   :: eps_0 = 8.8540000E-12_wp
    !free space permeability
    real(wp),   parameter   :: mu_0  = 1.25663706E-6_wp
    !free space intrinsic impedance
    real(wp),   parameter   :: Z0    = 376.734309_wp
    !sqrt(-1)
    complex(wp),parameter   :: cj    = (0.0_wp,1.0_wp)
end module const
!##############################################################################
!
!  Name:
!           mom_var
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           module containing variables and parameters related to the MoM.
!
!  Usage:
!           use  mom_var
```

```
!
!  Modules:
!          const
!
module mom_var
    !modules
    use     const, only : wp
    !list of parameters
    integer,parameter   ::  Nx    = 13      !number of transverse exp. func
    integer,parameter   ::  Nz    = 24      !number of longitudinal exp func
    integer,parameter   ::  mmax  = 150     !maximum number of terms in
                                            !interior green's function.
    integer,parameter   ::  nmax  = 75      !nmax=mmax/2
    !list of variables
    integer             ::  r,s,p,q         !mode index for exp./weight func.
    integer             ::  y1_mode,y2_mode !mode index for aux. func. y1/y2
    real(wp)            ::  kp,kq,kr,ks     !wavenumber for exp./weight func.
    real(wp)            ::  kp_sq,kq_sq     !do squared
    real(wp)            ::  kr_sq,ks_sq     !do squared
    complex(wp),dimension(1:Nx+Nz,1:Nx+Nz)  ::  Y       !moment matrix
    complex(wp),dimension(1:Nx+Nz)          ::  E       !Exp. coeff.
    complex(wp),dimension(1:Nx+Nz)          ::  h_inc   !excitation coeff.
end module mom_var
!############################################################################
!
!  Name:
!          slot_var
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Module containing variables describing a single slot such as
!          lengths, offsets, width etc.
!
!  Usage:
!          use  slot_var
!
!  Modules:
!          const
!
module slot_var
    !modules
    use     const, only : wp
    !variables
    real(wp)    :: w            !slot width
    real(wp)    :: Lx           !length of transverse slot
    real(wp)    :: Lz           !lenght of longitudinal slot
    real(wp)    :: Lx0          !initial length of transverse slot
    real(wp)    :: Lz0          !initial length of longitudinal slot
    real(wp)    :: Ls           !aux. variable for any slot length
    real(wp)    :: x0           !offset of longitudinal slot
    real(wp)    :: x00          !offset of transverse slot
    real(wp)    :: x0_pp        !offset of longitudinal slot in PP wg.
    real(wp)    :: x00_pp       !offset of transverse slot in PP wg.
    real(wp)    :: xi,eta,zeta  !local coordinates
end module slot_var
!############################################################################
!
```

```
!  Name:
!          gen_var
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Module containing general variables such as frequency, propagation
!          constants etc.
!
!  Usage:
!          use  gen_var
!
!  Modules:
!          const
!
module gen_var
    !modules
    use const, only : wp
    !list of variables
    integer     ::  ext_type    !flag indicating halv-space/PP-wg
    integer     ::  sw_type     !flag indicating kind of sweep to be performed
    real(wp)    ::  freq_0      !design frequency
    real(wp)    ::  freq        !frequency
    integer     ::  nfreq       !number of steps in frequency sweep
    real(wp)    ::  dfreq       !size of frequency step in sweep
    integer     ::  nll         !number of steps in length sweep for
                                !longitudinal slot
    real(wp)    ::  dll         !size of lenght step
    integer     ::  ntl         !number of steps in length sweep for
                                !transverse slot
    real(wp)    ::  dtl         !size of lenght step
    real(wp)    ::  k0,k0_sq    !free space wavenumber and do squared
    real(wp)    ::  lambda_0    !free space wavelength
    real(wp)    ::  omega       !angular frequency

end module gen_var
!##############################################################################
!
!  Name:
!          wg_var
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Module containing parameters and variables related to the main
!          waveguide.
!
!  Usage:
!          use  wg_var
!
!  Modules:
!          const,mom_var
!
module wg_var
    !modules
    use     const, only : wp
    use     mom_var, only : mmax,nmax
```

```
    !list of variables
    !wg a dimension
    real(wp)    :: a
    !wg b dimension
    real(wp)    :: b
    !wall thickness
    real(wp)    :: t
    !guided wavelenght
    real(wp)    :: lambda_g
    !TE_10 propagation constant
    real(wp)    :: beta_10
    !propagation constants in waveguide
    complex(wp),dimension(0:mmax,0:nmax) :: gamma

end module wg_var
!##############################################################################
!
!  Name:
!           ppwg_var
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Module containing parameters and variables related to the PP
!           waveguide.
!
!  Usage:
!           use  ppwg_var
!
!  Modules:
!           const
!
module ppwg_var
    !modules
    use     const, only : wp
    !list of variables
    !pp waveguide a' dimension
    real(wp)        :: a1
    !pp waveguide b' dimension
    real(wp)        :: b1
    !kx mode
    integer         :: m
    !kx propagation constant
    real(wp)        :: kx,kx_sq
    !ky propagation constant
    complex(wp)     :: ky,ky_sq
    !kz propagation constant
    complex(wp)     :: kz,kz_sq
end module ppwg_var
!##############################################################################
!
!  Name:
!           quad_var
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
```

```
!           This module contains some variables used in the integration of the
!           function 1/R*f(xi,zeta) over the rectangle: 2Lz*2Lx and in
!           evaluation of sommerfeld integrals by the method of weighted
!           averages. Also parameters specifying the accuracy etc.
!           are included.
!
!  Usage:
!           use  quad_var
!
!  Modules:
!           const
!
module quad_var
    !modules
    use     const, only : wp
    !list of parameters
    !maximum func. evaluations in quadr.
    integer, parameter  ::  maxfcn  = 500000
    !relative accuracy desired in quadr.
    real(wp),parameter  ::  errrel  = 0.0001_wp
    !absolute accuracy desired in quadr.
    real(wp),parameter  ::  errabs  = 0.0_wp
    !number of partial values to be calculated
    integer, parameter  ::  mm  = 6

    !list of variables
    integer     ::  c_flag      !flag indicating real or imag part
    real(wp)    ::  xi_0,zeta_0 !coordinate of singular point
    real(wp)    ::  xi_1,zeta_1 !upper limits in subintegral (aux. var)
    real(wp)    ::  f_0         !walue of f(xi,zeta) at singular point
    real(wp)    ::  df_dxi      !value of xi derivative at sing. point
    real(wp)    ::  df_dzeta    !value of zeta derivative at sing. point
    real(wp)    ::  R1          !min(abs(xi_1),abs(zeta_1)) (aux. var)
    real(wp)    ::  R2          !max(abs(xi_1),abs(zeta_1)) (aux. var)
    real(wp)    ::  R3          !sqrt(xi_1**2+zeta_1**2) (aux. var)
    real(wp)    ::  kw,kw1,kw2  !wavenumber in osc. part of sommerfelt int.
    real(wp)    ::  kz_max      !max kz in deformed contour integral
    real(wp)    ::  dkw         !wavenumber in osc. part of sommerfelt int.
    real(wp)    ::  dkz_max     !max kz in deformed contour integral
end module quad_var
!################################################################################
!
!  Name:
!           radfield_var
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Module containing variables related to far zone fields.
!
!  Usage:
!           use  radfield_var
!
!  Modules:
!           const
!
module radfield_var
    !modules
```

```
    use const, only : wp
    !list of variables
    !vector holding spherical comp. of the E-field
    complex(wp),dimension(1:3)  ::  E_sph
    !vector holding spherical comp. of the H-field
    complex(wp),dimension(1:3)  ::  H_sph
    !vector holding spherical comp. of the elec. vector potential
    complex(wp),dimension(1:3)  ::  F_sph
    !vector holding cartisian comp. of the elec. vector potential
    complex(wp),dimension(1:3)  ::  F_xyz
    !vector holding spherical comp. of the magn. vector potential
    complex(wp),dimension(1:3)  ::  A_sph
    !vector holding cartesian comp. of the magn. vector potential
    complex(wp),dimension(1:3)  ::  A_xyz
    !coordinate vectors
    real(wp),dimension(1:3)     ::  sph,xyz
end module radfield_var
```

```
!###########################################################################
!
!  Name:
!           read_input
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine reads information from the slot.in file into
!           a namelist
!
!  Usage:
!           call read_input()
!
!  Arguments:
!
!  Modules:
!           ppwg_var,slot_var,mom_var,gen_var,wg_var
!
!  Subroutines:
!
!  Remarks:
!           The input files should not contain entries not in namelist.
!
subroutine read_input()
    !modules
    use gen_var, only : freq_0,nfreq,dfreq,nll,dll,ntl,dtl,ext_type,sw_type
    use wg_var, only : a,b,t
    use ppwg_var, only : a1,b1
    use slot_var, only : w,Lz0,Lx0,x0
    !implicit statement
    implicit none
    !declaration of namelist
    namelist /var_list/ freq_0,nfreq,dfreq,Lz0,nll,dll,Lx0,ntl,dtl,&
                        w,x0,a,b,t,ext_type,a1,b1,sw_type
    !list of local variables
    integer :: ioerr

    OPEN (7, FILE='slot.in', STATUS='OLD', ACTION='READ',IOSTAT=ioerr)
    READ (UNIT=7,NML=var_list)
    CLOSE (7)
```

```
return
end

!###########################################################################
!
!  Name:
!           read_efield
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine reads expansion coefficients from a previous
!           simulation stored in the efield.dat file
!
!  Usage:
!           call read_efield()
!
!  Arguments:
!
!  Modules:
!           const,ppwg_var,slot_var,mom_var,gen_var
!
!  Subroutines:
!
subroutine read_efield()
    !modules
    use const, only : cj,wp
    use gen_var, only : freq
    use slot_var, only : Lz,Lx,x0
    use ppwg_var, only : a1
    use mom_var, only : Nz,Nx,E
    !implicit statement
    implicit none
    !list of local variables
    integer :: ioerr,i,j
    real(wp), dimension(1:2)    :: z,R,T
    real(wp), dimension(1:2*Nz) :: Ex_in
    real(wp), dimension(1:2*Nx) :: Ez_in

    OPEN (8, FILE='efield.dat', STATUS='OLD',ACTION='READ',IOSTAT=ioerr)
    READ(8,1075) freq,x0,a1,Lx,Lz,z(1),z(2),R(1),R(2),T(1),T(2),Ex_in,Ez_in
    CLOSE (8)

    do i=1,Nz
        E(i)=Ex_in(2*i-1)+cj*Ex_in(2*i)
    enddo
    do i=1,Nx
        E(i+Nz)=Ez_in(2*i-1)+cj*Ez_in(2*i)
    enddo

1075 format(2x,EN15.6,4(2x,EN15.6),6(2x,E15.6),48(2x,E15.6),26(2x,E15.6))
return
end

!###########################################################################
!
!  Name:
```

```
!           init_propagation_var
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine initialize all propagation variables on basis of
!           frequency, waveguide dimensions etc.
!
!  Usage:
!           call init_propagation_var()
!
!  Arguments:
!
!  Modules:
!           const,mom_var,gen_var,wg_var
!
!  Subroutines:
!
subroutine init_propagation_var()
    !modules
    use const, only : eps_0,mu_0,c,pi,cj,wp
    use gen_var, only : k0,k0_sq,freq_0,freq,lambda_0,omega
    use wg_var, only : a,b,gamma,beta_10,lambda_g
    use mom_var, only : mmax,nmax
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,cdsqrt,dcmplx
    !list of local variables
    integer ::  m,n

    omega=2.0_wp*pi*freq
    k0=omega*dsqrt(eps_0*mu_0)
    k0_sq=k0**2
    lambda_0=c/freq
    beta_10=k0*dsqrt(1-(lambda_0/(2.0_wp*a))**2)
    lambda_g=2.0_wp*pi/beta_10

    do m=0,mmax
        do n=0,nmax
            gamma(m,n)=cdsqrt(dcmplx((m*pi/a)**2+(n*pi/b)**2-k0_sq))
        enddo
    enddo

return
end

!###############################################################################
!
!  Name:
!           dsinc
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Calculate the sinc function sin(x)/x
!
```

```
!  Usage:
!           dsin(x)
!
!  Arguments:
!   x       - Independent variable (Input)
!
!  Modules:
!           const
!
function dsinc(x)
    !modules
    use const, only : wp
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dabs
    !list of calling arguments
    real(wp)    ::  dsinc
    real(wp), intent(in)    ::  x
    !list of local parameters
    real(wp),parameter :: epsilon = 1.0E-30_wp

    if(dabs(x).gt.epsilon) then
        dsinc=dsin(x)/x
    else
        dsinc=1.0_wp
    endif

end function dsinc

!############################################################################
!
!  Name:
!           xyz2sph
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Transform cartesian vector components to spherical components
!
!  Usage:
!           xyz2sph(F_xyz,sph,F_sph)
!
!  Arguments:
!   F_xyz   - vector holding cartesian components (Input)
!
!   sph     - spherical coordinates (Input)
!
!   F_sph   - vector holding spherical components (output)
!
!  Modules:
!           const
!
subroutine xyz2sph(F_xyz,sph,F_sph)
    !modules
    use const, only : wp
    !implicit statement
    implicit none
```

```
    !declaration of intrinsic functions
    intrinsic dsin,dcos
    !list of local variables
    real(wp), dimension(1:3),intent(in) :: sph
    complex(wp), dimension(1:3),intent(out) :: F_sph
    complex(wp), dimension(1:3),intent(in) :: F_xyz

    F_sph(1)=F_xyz(1)*dsin(sph(2))*dcos(sph(3))+&
             F_xyz(2)*dsin(sph(2))*dsin(sph(3))+&
             F_xyz(3)*dcos(sph(2))
    F_sph(2)=F_xyz(1)*dcos(sph(2))*dcos(sph(3))+&
             F_xyz(2)*dcos(sph(2))*dsin(sph(3))-&
             F_xyz(3)*dsin(sph(2))
    F_sph(3)=-F_xyz(1)*dsin(sph(3))+&
             F_xyz(2)*dcos(sph(3))

return
end
```

```
!###############################################################################
!
!  Name:
!          yzz_int_pq
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Computes the interior self contribution to the fields in a
!          longitudinal slot.
!
!  Usage:
!          yzz_int_pq()
!
!  Arguments:
!
!  Modules:
!          const,slot_var,mom_var,gen_var,wg_var
!
function yzz_int_pq()
    !modules
    use const, only : pi,cj,wp
    use slot_var, only : w,Lz,x0
    use mom_var, only : p,q,mmax,nmax
    use gen_var, only : k0_sq,freq,lambda_0
    use wg_var, only : a,b,gamma
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic mod,dcos,dsin,cdexp
    !declaration of functions
    real(wp)    ::  dsinc
    !list of calling arguments
    complex(wp) ::  yzz_int_pq
    !list of local variables
    integer     ::  m,n
    real(wp)    ::  p_term,q_term
    real(wp)    ::  w_m,sign
    complex(wp) ::  sum1,sum2, add_term1,add_term2,gam

```

```
        sum1=(0.0_wp,0.0_wp)
        sum2=(0.0_wp,0.0_wp)
        if(mod((p+q),2).eq.1) then
            yzz_int_pq=(0.0_wp,0.0_wp)
        else
            p_term=(p*pi/(2.0_wp*Lz))**2
            q_term=(q*pi/(2.0_wp*Lz))**2
            if((mod(p,2).eq.1).and.(mod(q,2).eq.1)) then
                sign=1.0_wp
            else if ((mod(p,2).eq.0).and.(mod(q,2).eq.0)) then
                sign=-1.0_wp
            end if

            do m=0,mmax
                if (m.eq.0) then
                    w_m=2.0_wp
                else
                    w_m=4.0_wp*dsinc((m*pi*w)/(2.0_wp*a))*&
                        (dcos(m*pi*x0/a))**2
                end if
                do n=0,nmax
                    add_term1=(0.0_wp,0.0_wp)
                    add_term2=(0.0_wp,0.0_wp)
                    gam=gamma(m,n)
                    add_term1=(1.0_wp+sign*cdexp(-2.0_wp*gam*Lz))*(gam**2+k0_sq)/&
                        (gam*(gam**2+p_term)*(gam**2+q_term))

                    add_term2=w_m/(gam**2+p_term)
                    !adjust addterms in case n==0
                    if (n.eq.0) then
                        add_term1=add_term1*0.5_wp
                        add_term2=add_term2*0.5_wp
                    end if

                    sum1=sum1+w_m*add_term1
                    if(p.eq.q) then
                        sum2=sum2+add_term2
                    end if
                enddo
            enddo
            yzz_int_pq=cj*lambda_0*w/(8.0_wp*a*b*pi)*&
                        (sum1*p*pi/Lz*q*pi/Lz+sum2*4.0_wp*(k0_sq-p_term)*Lz)
        end if

    end function yzz_int_pq

    !############################################################################
    !
    !  Name:
    !           yxx_int_rs
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Computes the interior self contribution to the fields in a
    !           transverse slot.
    !
    !  Usage:
```

```
!           yxx_int_rs()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,mom_var,gen_var,wg_var
!
function yxx_int_rs()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lx,x00,w
    use mom_var, only : r,s,mmax,nmax
    use gen_var, only : k0,k0_sq,lambda_0,omega
    use wg_var, only : a,b,beta_10,gamma
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,cdexp,mod
    !list of calling arguments
    complex(wp) ::  yxx_int_rs
    !list of local variables
    integer     ::  m,n
    real(wp)    ::  w_m,w_half,m_term,r_term,s_term
    complex(wp) ::  sum, add_term, gam

    if(mod((r+s),2).eq.1) then
        yxx_int_rs=(0.0_wp,0.0_wp)
    else
        sum=(0.0_wp,0.0_wp)
        w_half=0.5_wp*w
        r_term=(r*pi/(2.0_wp*Lx))**2
        s_term=(s*pi/(2.0_wp*Lx))**2
        do m=0,mmax
            m_term=m*pi/a
            w_m=(m_term**2-k0_sq)/&
                ((r_term-m_term**2)*(s_term-m_term**2))
            if(mod(r,2).eq.1) then !r odd
                w_m=w_m*(dcos(m_term*Lx)*dsin(m_term*x00))**2
            else    !r even
                w_m=w_m*(dcos(m_term*x00)*dsin(m_term*Lx))**2
            end if
            !adjust w_m in case m==0
            if(m.eq.0) then
                w_m=w_m*0.5_wp
            end if
            do n=0,nmax
                gam=gamma(m,n)
                add_term=w_m*(1.0_wp-cdexp(-gam*w_half))/gam**2
                !adjust addterm in case n==0
                if (n.eq.0) then
                    add_term=add_term*0.5_wp
                end if
                sum=sum+add_term
            enddo !n loop
        enddo !m loop

        yxx_int_rs=-4.0_wp*Z0/(cj*omega*mu_0*a*b)*r*pi/Lx*s*pi/Lx*sum

    end if
```

```
end function yxx_int_rs

!############################################################################
!
!  Name:
!           yxz_int_rq
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the interior cross coupling contribution to the fields
!           in a longitudinal slot due to currents in a transverse slot.
!
!  Usage:
!           yxz_int_rq()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,mom_var,gen_var,wg_var
!
function yxz_int_rq()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lx,Lz,x0,x00,w
    use mom_var, only : r,q,mmax,nmax
    use gen_var, only : k0,omega
    use wg_var, only : a,b,gamma
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,cdexp,mod
    !list of calling arguments
    complex(wp) ::  yxz_int_rq
    !list of local variables
    integer     ::  m,n
    real(wp)    ::  cos_term1,cos_term2,sin_term,w_m,q_term
    complex(wp) ::  w_mn,gam
    complex(wp) ::  sum, add_term1,add_term23

    if(mod(q,2).eq.1) then !q odd
        yxz_int_rq=(0.0_wp,0.0_wp)
    else
        sum=(0.0,0.0)
        sin_term=2.0_wp*Lz*dsin(w*q*pi/(4.0_wp*Lz))
        cos_term1=q*pi*dcos(w*q*pi/(4.0_wp*Lz))
        cos_term2=dcos(q*pi/2.0_wp)
        q_term=(q*pi/(2.0_wp*Lz))**2
        do m=1,mmax
            w_m=2.0_wp*m*pi/a*dcos(m*pi*x0/a)
            w_m=w_m/((r*pi/(2.0_wp*Lx))**2-(m*pi/a)**2)
            if(mod(r,2).eq.1) then !r odd
                w_m=w_m*dcos(m*pi*Lx/a)*dsin(m*pi*x00/a)
            else    !r even
                w_m=-w_m*dcos(m*pi*x00/a)*dsin(m*pi*Lx/a)
            end if
            !adjust w_m in case m==0
            if (m.eq.0) then
```

```
                w_m=w_m*0.5_wp
            end if
            do n=0,nmax
                gam=gamma(m,n)
                w_mn=2.0_wp/(q_term+gam**2)
                !adjust w_mn in case n==0
                if (n.eq.0) then
                    w_mn=w_mn*0.5_wp
                end if
                !case 1
                add_term1=cos_term2*(&
                        sin_term*gam*(cdexp(-gam*w)+1.0_wp)+&
                        cos_term1*(cdexp(-gam*w)-1.0_wp))
                !case 2+3
                add_term23=cos_term2*(1.0_wp-cdexp(-gam*w))*&
                           (cos_term1+gam*sin_term)&
                            +q*pi*(cdexp(-gam*(Lz+w/2.0_wp))-&
                                    cdexp(-gam*(Lz-w/2.0_wp)))
                sum=sum+(add_term1+add_term23)*w_m*w_mn/gam
            enddo !n loop
        enddo !m loop

        yxz_int_rq=-Z0/(cj*omega*mu_0*a*b)*r*pi/Lx/Lz/2.0_wp*sum

    end if
end function yxz_int_rq

!##############################################################################
!
!  Name:
!           yzx_int_ps
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the interior cross coupling contribution to the fields
!           in a transverse slot due to currents in a longitudinal slot.
!
!  Usage:
!           yzx_int_ps()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,mom_var,gen_var,wg_var
!
function yzx_int_ps()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lx,Lz,x0,x00,w
    use mom_var, only : p,s,mmax,nmax
    use gen_var, only : k0,omega
    use wg_var, only : a,b,gamma
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod,cdexp
    !list of calling arguments
```

```
    complex(wp) ::  yzx_int_ps
    !list of local variables
    integer     ::  m,n
    real(wp)    ::  p_term,s_term,cos_term,w_m
    complex(wp) ::  sum,w_mn

    if(mod(p,2).eq.1) then !p odd
        yzx_int_ps=(0.0_wp,0.0_wp)
    else
        p_term=(p*pi/(2.0_wp*Lz))**2
        s_term=(s*pi/(2.0_wp*Lx))**2
        cos_term=dcos(p*pi/2.0_wp)
        sum=(0.0,0.0)
        do m=0,mmax
            w_m=2.0_wp*dsin(m*pi*w/(2.0_wp*a))*dcos(m*pi*x0/a)
            w_m=w_m/(s_term-(m*pi/a)**2)
            if(mod(s,2).eq.1) then !s odd
                w_m=w_m*dcos(m*pi*Lx/a)*dsin(m*pi*x00/a)
            else    !s even
                w_m=-w_m*dcos(m*pi*x00/a)*dsin(m*pi*Lx/a)
            end if
            !adjust w_m in case m==0
            if (m.eq.0) then
                w_m=w_m*0.5_wp
            end if
            do n=0,nmax
                w_mn=2.0_wp/(p_term+gamma(m,n)**2)
                !adjust w_mn in case n==0
                if (n.eq.0) then
                    w_mn=w_mn*0.5_wp
                end if
                sum=sum+(cdexp(-gamma(m,n)*Lz)-cos_term)*w_m*w_mn
            enddo !n loop
        enddo !m loop

        yzx_int_ps=-Z0/(cj*omega*mu_0*a*b)*p*pi/Lz*s*pi/Lx*sum

    end if
end function yzx_int_ps
```

```
!#############################################################################
!
!  Name:
!           yzx_ext_ps
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exterior cross coupling contribution to the fields
!           in a transverse slot due to currents in a longitudinal slot. Used
!           in case the slot radiates into half space.
!
!  Usage:
!           yzx_ext_ps()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
```

```
!
!  Subroutines:
!          yzx_ext_re_ps_p,yzx_ext_im_ps_p,inv_r_int
!
!  IMSL subroutines:
!          dq3nd
!
function yzx_ext_ps()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lx,Lz,w
    use mom_var, only : p,s,ks,kp
    use gen_var, only : k0,omega
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,maxfcn,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,sum,mod
    !declaration of functions
    real(wp), external          ::  yzx_ext_re_ps_p,yzx_ext_im_ps_p
    real(wp)                    ::  inv_r_int
    !list of calling arguments
    complex(wp)                 ::  yzx_ext_ps
    !list of local variables
    real(wp), dimension(1:2)    ::  upper
    real(wp), dimension(1:2)    ::  lower
    real(wp), dimension(1:2)    ::  re_val_r1
    real(wp), dimension(1:2)    ::  re_val_r2
    real(wp)                    ::  errest,cos_term
    real(wp)                    ::  im_val
    integer                     ::  ier

    if(mod(p,2).eq.1) then !p odd
        yzx_ext_ps=(0.0_wp,0.0_wp)
    else
        upper=(/Lx,Lz/)
        lower=(/-Lx,-Lz/)
        zeta_0=0.0_wp
        cos_term=dcos(p*pi/2.0_wp)
        kp=p*pi/(2.0_wp*Lz)
        ks=s*pi/(2.0_wp*Lx)

        !real part of 1/R1 term
        xi_0=Lx
        f_0=0.0_wp
        df_dxi=(-1.0_wp)**s*ks*cos_term
        df_dzeta=0.0_wp
        call dq3nd (yzx_ext_re_ps_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,re_val_r1(1),errest)
        re_val_r1(2)=inv_r_int(-Lx,Lx,-Lz,Lz)

        !real part of 1/R2 term
        xi_0=Lx-w
        f_0=-(-1.0_wp)**s*dsin(ks*w)*cos_term
        df_dxi=(-1.0_wp)**s*ks*dcos(ks*w)*cos_term
        df_dzeta=0.0_wp
        call dq3nd (yzx_ext_re_ps_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,re_val_r2(1),errest)
        re_val_r2(2)=inv_r_int(-Lx,Lx,-Lz,Lz)
```

```

        !imaginary part
        call dq3nd (yzx_ext_im_ps_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,im_val,errest)

        yzx_ext_ps=-Z0*p/(4.0_wp*cj*omega*mu_0*Lz)*&
            (sum(re_val_r1)-sum(re_val_r2)+k0*cj*im_val)
    endif

end function yzx_ext_ps

!#############################################################################
!
!  Name:
!           yzx_ext_re_ps_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yzx_ext_ps. Evaluates
!           the real part of kernel used to compute the exterior cross
!           coupling contribution to the fields in a transverse slot due to
!           currents in a longitudinal slot.
!
!  Usage:
!           yzx_ext_re_ps_p(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
function yzx_ext_re_ps_p(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Lx,Lz
    use mom_var, only : s,p,ks,kp
    use gen_var, only : k0
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,dsin,dcos
    !list of calling arguments
    real(wp)                :: yzx_ext_re_ps_p
    real(wp), dimension(1:2), intent(in) :: x
    integer , intent(in)    :: dim
    !list of local variables
    real(wp)                :: RR

    !xi=x(1)    zeta=x(2)
    RR=dsqrt((x(1)-xi_0)**2+(x(2)-zeta_0)**2)

    if(RR.lt.1E-8_wp) then
        yzx_ext_re_ps_p=0.0_wp
    else
```

```
            yzx_ext_re_ps_p=(dsin(ks*(Lx+x(1)))&
                *dcos(kp*(Lz+x(2)))*dcos(k0*RR)&
                -f_0-(x(1)-xi_0)*df_dxi-(x(2)-zeta_0)*df_dzeta)/RR
        endif

    end function yzx_ext_re_ps_p

    !############################################################################
    !
    !  Name:
    !           yzx_ext_im_ps_p
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Auxiliary function used by the function yzx_ext_ps. Evaluates
    !           the imaginary part of kernel used to compute the exterior cross
    !           coupling contribution to the fields in a transverse slot due to
    !           currents in a longitudinal slot.
    !
    !  Usage:
    !           yzx_ext_im_ps_p(dim,x)
    !
    !  Arguments:
    !   dim     - Obsolete!
    !   x       - Independent variable (Input)
    !
    !  Modules:
    !           const,quad_var,slot_var,mom_var,gen_var
    !
    !  Subroutines:
    !           dsinc
    !
    function yzx_ext_im_ps_p(dim,x)
        !modules
        use const, only : pi,wp
        use slot_var, only : Lx,Lz,x0,x00,w
        use mom_var, only : ks,kp
        use gen_var, only : k0
        !implicit statement
        implicit none
        !declaration of intrinsic functions
        intrinsic dsqrt,dsin,dcos
        !declaration of functions
        real(wp)            ::  dsinc
        !list of calling arguments
        real(wp)                ::  yzx_ext_im_ps_p
        real(wp), dimension(1:2),intent(in)::   x
        integer,intent(in)      ::  dim
        !list of local variables
        real(wp)                ::  R1,R2,xii

        !xi=x(1)    zeta=x(2)
        xii=(x0-x00)-x(1)

        R1=dsqrt((xii+w/2.0_wp)**2+x(2)**2)
        R2=dsqrt((xii-w/2.0_wp)**2+x(2)**2)

```

```
    yzx_ext_im_ps_p=dsin(ks*(Lx+x(1)))*&
            dcos(kp*(Lz+x(2)))*&
            (dsinc(k0*R2)-dsinc(k0*R1))

end function yzx_ext_im_ps_p

!#############################################################################
!
!  Name:
!           yxz_ext_rq
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exterior cross coupling contribution to the fields
!           in a longitudinal slot due to currents in a transverse slot. Used
!           in case the slot radiates into half space.
!
!  Usage:
!           yxz_ext_rq()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
!  Subroutines:
!           yxz_ext_re_rq_p,yxz_ext_im_rq_p,inv_r_int
!
!  IMSL subroutines:
!           dq3nd
!
function yxz_ext_rq()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lx,Lz,w
    use mom_var, only : q,r,kq,kr
    use gen_var, only : k0,omega
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,maxfcn,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,sum,mod
    !declaration of functions
    real(wp), external          ::  yxz_ext_re_rq_p,yxz_ext_im_rq_p
    real(wp)                    ::  inv_r_int
    !list of calling arguments
    complex(wp)                 ::  yxz_ext_rq
    !list of local variables
    real(wp), dimension(1:2)    ::  upper
    real(wp), dimension(1:2)    ::  lower
    real(wp), dimension(1:2)    ::  re_val_r1
    real(wp), dimension(1:2)    ::  re_val_r2
    real(wp), dimension(1:3)    ::  arg
    real(wp)                    ::  errest
    real(wp)                    ::  im_val
    integer                     ::  ier

```

```
    if(mod(q,2).eq.1) then !q odd
        yxz_ext_rq=(0.0_wp,0.0_wp)
    else
        upper=(/Lx,Lz/)
        lower=(/-Lx,-Lz/)
        kq=q*pi/(2.0_wp*Lz)
        kr=r*pi/(2.0_wp*Lx)
        arg=(/kq*(Lz+w/2.0_wp),kr*(2.0_wp*Lx-w/2.0_wp),kq*(Lz-w/2.0_wp)/)
        xi_0=Lx-w/2.0_wp

        !real part of 1/R1 term
        zeta_0=w/2.0_wp
        f_0=dsin(arg(1))*dcos(arg(2))
        df_dxi=-kr**dsin(arg(1))*dsin(arg(2))
        df_dzeta=kq*dcos(arg(1))*dcos(arg(2))
        call dq3nd(yxz_ext_re_rq_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,re_val_r1(1),errest)
        re_val_r1(2)=inv_r_int(-Lx,Lx,-Lz,Lz)

        !real part of 1/R2 term
        zeta_0=-w/2.0_wp
        f_0=dsin(arg(3))*dcos(arg(2))
        df_dxi=-kr*dsin(arg(3))*dsin(arg(2))
        df_dzeta=kq*dcos(arg(2))*dcos(arg(3))
        call dq3nd(yxz_ext_re_rq_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,re_val_r2(1),errest)
        re_val_r2(2)=inv_r_int(-Lx,Lx,-Lz,Lz)

        !imaginary part
        call dq3nd (yxz_ext_im_rq_p,2,lower,upper,&
            maxfcn,errabs,errrel,ier,im_val,errest)

        yxz_ext_rq=Z0*r/(4.0_wp*cj*omega*mu_0*Lx)*&
            (sum(re_val_r1)-sum(re_val_r2)+cj*k0*im_val)

    end if

end function yxz_ext_rq

!##############################################################################
!
!  Name:
!           yxz_ext_re_rq_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yxz_ext_rq. Evaluates
!           the real part of kernel used to compute the exterior cross
!           coupling contribution to the fields in a longitudinal slot due to
!           currents in a transverse slot.
!
!  Usage:
!           yxz_ext_re_rq_p(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
```

```
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
function yxz_ext_re_rq_p(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Lx,Lz
    use mom_var, only : kr,kq
    use gen_var, only : k0
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,dsin,dcos
    !list of calling arguments
    real(wp)                    ::  yxz_ext_re_rq_p
    real(wp), dimension(1:2),intent(in) ::  x
    integer,intent(in)          ::  dim
    !list of local variables
    real(wp)                    ::  RR

    !xi=x(1)    zeta=x(2)
    RR=sqrt((x(1)-xi_0)**2+(x(2)-zeta_0)**2)

    if(RR.lt.1E-8_wp) then
        yxz_ext_re_rq_p=0.0_wp
    else
        yxz_ext_re_rq_p=(dsin(kq*(Lz+x(2)))&
            *dcos(kr*(Lx+x(1)))*dcos(k0*RR)&
            -f_0-(x(1)-xi_0)*df_dxi-(x(2)-zeta_0)*df_dzeta)/RR
    endif

end function yxz_ext_re_rq_p

!############################################################################
!
!  Name:
!           yxz_ext_im_rq_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yxz_ext_rq. Evaluates
!           the imaginary part of kernel used to compute the exterior cross
!           coupling contribution to the fields in a longitudinal slot due to
!           currents in a transverse slot.
!
!  Usage:
!           yxz_ext_im_rq_p(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
```

```
!  Subroutines:
!           dsinc
!
function yxz_ext_im_rq_p(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Lx,Lz,x0,x00,w
    use mom_var, only : kr,kq
    use gen_var, only : k0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,dsin,dcos
    !declaration of functions
    real(wp)                :: dsinc
    !list of calling arguments
    real(wp)                :: yxz_ext_im_rq_p
    real(wp), dimension(1:2),intent(in)::   x
    integer ,intent(in)     :: dim
    !list of local variables
    real(wp)                :: R1,R2,xii

    !xi=x(1)    zeta=x(2)
    xii=(x0-x00)-x(1)
    R1=dsqrt(xii**2+(w/2.0_wp-x(2))**2)
    R2=dsqrt(xii**2+(w/2.0_wp+x(2))**2)

    yxz_ext_im_rq_p=dsin(kq*(Lz+x(2)))*&
            dcos(kr*(Lx+x(1)))*&
            (dsinc(k0*R2)-dsinc(k0*R1))

end function yxz_ext_im_rq_p

!###############################################################################
!
!  Name:
!           y_ext
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exterior self contribution to the MoM matrix for any
!           slot of length L. Used in case the slot radiates into half space.
!
!  Usage:
!           y_ext(p,q,L)
!
!  Arguments:
!   p       - order of expansion function (Input)
!   q       - order of weighting function (Input)
!   L       - length of slot (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
!  Subroutines:
!           y2,y1
!
```

```
function y_ext(p,q,L)
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use mom_var, only : y1_mode,y2_mode
    use gen_var, only : k0,omega
    use slot_var, only : Ls
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic mod
    !declaration of functions
    complex(wp)     ::  y2,y1
    !list of calling arguments
    complex(wp)         ::  y_ext
    real(wp),intent(in) ::  L
    integer,intent(in)  ::  p,q

    Ls=L
    if(mod((p+q),2).eq.1) then
        y_ext=(0.0_wp,0.0_wp)
    else if ((p.ne.q).and.(mod((p+q),2).ne.1)) then
        y1_mode=q   !y1(q)
        y_ext=((k0*2.0_wp*Ls)**2-(q*pi)**2)*p*y1()
        y1_mode=p   !y1(p)
        y_ext=(y_ext-((k0*2.0_wp*Ls)**2-(p*pi)**2)*q*y1())/&
                (cj*pi**2*omega*(q**2-p**2))
    else if (p.eq.q) then
        y1_mode=q   !y1(q)
        y2_mode=q   !y2(q)
        y_ext=1.0_wp/(cj*2.0_wp*pi*omega)*((k0*2.0_wp*Ls)**2-(q*pi)**2)*&
                y2()-q/(cj*omega)*y1()
    end if

    y_ext=2.0_wp*Z0*y_ext/mu_0

end function y_ext

!############################################################################
!
!  Name:
!           y1
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y_ext. Computes the value
!           of the function y1.
!
!  Usage:
!           y1()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
!  Subroutines:
!           y1_re_r1_pp,y1_re_r2_pp,y1_im_pp,inv_r_quad_int
```

```
!
!  IMSL subroutines:
!          dq3nd
!
function y1()
    !modules
    use const, only : cj,pi,wp
    use gen_var, only : k0
    use slot_var, only : w,Ls
    use mom_var, only : y1_mode
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,xi_1,zeta_1,&
                        maxfcn,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic sum
    !declaration of functions
    real(wp),external   ::  y1_re_r1_pp,y1_re_r2_pp,y1_im_pp
    real(wp)            ::  inv_r_quad_int
    !list of calling arguments
    complex(wp)                 ::  y1
    !list of local variables
    real(wp), dimension(1:2)    ::  upper
    real(wp), dimension(1:2)    ::  lower
    real(wp), dimension(1:3)    ::  re_val
    real(wp)                    ::  im_val
    real(wp)                    ::  errest
    integer                     ::  ier

    upper=(/0.5_wp*w,0.5_wp/)
    lower=(/0.0_wp,0.0_wp/)

    !real part of 1/R1 term
    f_0=0.0_wp
    df_dxi=0.0_wp
    df_dzeta=y1_mode*pi
    xi_0=0.0_wp
    zeta_0=0.5_wp
    call dq3nd (y1_re_r1_pp,2,lower,upper,&
        maxfcn,errabs,errrel,ier,re_val(1),errest)

    !change of variable: zeta -> zeta/(2*Lz)
    df_dzeta=y1_mode*pi/(2.0_wp*Ls)
    xi_1=upper(1)-xi_0
    zeta_1=2.0_wp*Ls*(lower(2)-zeta_0)
    re_val(2)=inv_r_quad_int()/(2.0_wp*Ls)

    !real part of 1/R2 term
    call dq3nd (y1_re_r2_pp,2,lower,upper,&
        maxfcn,errabs,errrel,ier,re_val(3),errest)

    !imaginary part
    call dq3nd (y1_im_pp,2,lower,upper,&
        maxfcn,errabs,errrel,ier,im_val,errest)

    y1=sum(re_val)+cj*k0*im_val

end function y1
```

```
!##############################################################################
!
!  Name:
!           y2
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y_ext. Computes the value
!           of the function y2.
!
!  Usage:
!           y2()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var
!
!  Subroutines:
!           y2_re_pp,y2_im_pp,inv_r_quad_int
!
!  IMSL subroutines:
!           dq3nd
!
function y2()
    !modules
    use const, only : cj,wp
    use gen_var, only : k0
    use slot_var, only : w,Ls
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_0,zeta_0,xi_1,zeta_1,&
                         maxfcn,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic sum
    !declaration of functions
    real(wp), external          ::  y2_re_pp,y2_im_pp
    real(wp)                    ::  inv_r_quad_int
    !list of calling arguments
    complex(wp)                 ::  y2
    !list of local variables
    real(wp), dimension(1:2)    ::  re_val
    real(wp), dimension(1:2)    ::  upper
    real(wp), dimension(1:2)    ::  lower
    real(wp)                    ::  errest
    real(wp)                    ::  im_val
    integer                     ::  ier

    upper=(/w/2.0_wp,1.0_wp/)
    lower=(/0.0_wp,0.0_wp/)

    !real part of 1/R term
    xi_0=0.0_wp
    zeta_0=0.0_wp
    f_0=1.0_wp
    df_dxi=0.0_wp
    df_dzeta=0.0_wp
```

```
    call dq3nd (y2_re_pp,2,lower,upper,&
        maxfcn,errabs,errrel,ier,re_val(1),errest)

    !change of variable: zeta -> zeta/(2*Lz)
    xi_1=upper(1)-xi_0
    zeta_1=2.0_wp*Ls*(upper(2)-zeta_0)
    re_val(2)=inv_r_quad_int()/(2.0_wp*Ls) !sum(reg_val)

    !imaginary part
    call dq3nd (y2_im_pp,2,lower,upper,&
        maxfcn,errabs,errrel,ier,im_val,errest)

    y2=sum(re_val)-cj*k0*im_val

end function y2

!##############################################################################
!
!  Name:
!           y1_im_pp
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y1. Evaluates the
!           imaginary part of the kernel integrated in y1.
!
!  Usage:
!           y1_im_pp(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
!  Subroutines:
!           dsinc
!
function y1_im_pp(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Ls
    use mom_var, only : y1_mode
    use gen_var, only : k0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dsqrt
    !declaration of functions
    real(wp)        ::  dsinc
    !list of calling arguments
    real(wp)                 ::  y1_im_pp
    real(wp), dimension(1:2),intent(in) ::  x
    integer,intent(in)       ::  dim
    !list of local variables
```

```
    real(wp)        ::  R1,R2

    R1=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)-0.5_wp)**2)
    R2=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)+0.5_wp)**2)

    y1_im_pp=dsin(y1_mode*pi*(x(2)+0.5_wp))&
            *(dsinc(k0*R2)-(-1.0_wp)**y1_mode*dsinc(k0*R1))

end function y1_im_pp

!###########################################################################
!
!  Name:
!           y1_re_r1_pp
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y1. Evaluates partially
!           the real part of the kernel integrated in y1.
!
!  Usage:
!           y1_re_r1_pp(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
function y1_re_r1_pp(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Ls
    use mom_var, only : y1_mode
    use gen_var, only : k0
    use quad_var, only : df_dzeta,zeta_0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,dsqrt
    !list of calling arguments
    real(wp)                  ::  y1_re_r1_pp
    real(wp), dimension(1:2),intent(in) ::  x
    integer,intent(in)        ::  dim
    !list of local variables
    real(wp)    ::      R1

    !xi=x(1)    sigma=zeta=x(2)
    R1=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)-0.5_wp)**2)

    if(R1.lt.1E-8_wp) then
        y1_re_r1_pp=0.0_wp
    else
        y1_re_r1_pp=(dsin(y1_mode*pi*(x(2)+0.5_wp))&
            *(-1.0_wp)**y1_mode*dcos(k0*R1)-(x(2)-zeta_0)*df_dzeta)/R1
    endif
```

```

end function y1_re_r1_pp

!#############################################################################
!
!  Name:
!           y1_re_r2_pp
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y1. Evaluates partially
!           the real part of the kernel integrated in y1.
!
!  Usage:
!           y1_re_r2_pp(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
function y1_re_r2_pp(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Ls
    use mom_var, only : y1_mode
    use gen_var, only : k0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,dsqrt
    !list of calling arguments
    real(wp)                ::  y1_re_r2_pp
    real(wp), dimension(1:2),intent(in) ::  x
    integer,intent(in)      ::  dim
    !list of local variables
    real(wp)    ::      R2

    !xi=x(1)    sigma=x(2)
    R2=dsqrt(x(1)**2+4.0_wp*Ls**2*(x(2)+0.5_wp)**2)

    y1_re_r2_pp=-dsin(y1_mode*pi*(x(2)+0.5_wp))*dcos(k0*R2)/R2

end function y1_re_r2_pp

!#############################################################################
!
!  Name:
!           y2_im_pp
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y2. Evaluates the
```

```
!           imaginary part of the kernel integrated in y2.
!
!  Usage:
!           y2_im_pp(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
!  Subroutines:
!           dsinc
!
function y2_im_pp(dim,x)
    !modules
    use const, only : pi,wp
    use slot_var, only : Ls
    use mom_var, only : y2_mode
    use gen_var, only : k0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,dsqrt
    !declaration of functions
    real(wp)        ::  dsinc
    !list of calling arguments
    real(wp)                ::  y2_im_pp
    real(wp), dimension(1:2),intent(in) ::  x
    integer,intent(in)      ::  dim
    !list of local variables
    real(wp)        ::  R

    R=dsqrt(x(1)**2+(2.0_wp*Ls)**2*x(2)**2)

    y2_im_pp=dsinc(k0*R)*(dcos(y2_mode*pi*x(2))*(1.0_wp-x(2))&
        +dsin(y2_mode*pi*x(2))/(y2_mode*pi))

end function y2_im_pp

!###############################################################################
!
!  Name:
!           y2_re_pp
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function y2. Evaluates the
!           real part of the kernel integrated in y2.
!
!  Usage:
!           y2_re_pp(dim,x)
!
!  Arguments:
!   dim     - Obsolete!
!   x       - Independent variable (Input)
```

```
!
!  Modules:
!          const,slot_var,mom_var,gen_var
!
function y2_re_pp(dim,x)
    !modules
    use const, only : pi,cj,wp
    use slot_var, only : Ls
    use mom_var, only : y2_mode
    use gen_var, only : k0
    use quad_var, only : f_0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,dsqrt
    !list of calling arguments
    real(wp)                :: y2_re_pp
    real(wp), dimension(1:2),intent(in) :: x
    integer,intent(in)      :: dim
    !list of local variables
    real(wp)        :: R

    R=dsqrt(x(1)**2+(2.0_wp*Ls)**2*x(2)**2)

    if(R.lt.1E-8_wp) then
        y2_re_pp=0.0_wp
    else
        y2_re_pp=(dcos(k0*R)*(&
        (1.0_wp-x(2))*dcos(y2_mode*pi*x(2))&
            +dsin(y2_mode*pi*x(2))/(pi*y2_mode))-f_0)/R
    endif

end function y2_re_pp
!###############################################################################
!
!  Name:
!          yzx_ppwg_ps
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Computes the exterior cross coupling contribution to the fields
!          in a transverse slot due to currents in a longitudinal slot. Used
!          in case the slot radiates into a PP waveguide.
!
!  Usage:
!          yzx_ppwg_ps()
!
!  Arguments:
!
!  Modules:
!          const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
!
!  Subroutines:
!          yzx_ppwg_ps_p_revised_c,yzx_ppwg_ps_p,int_wavg
!
!  IMSL subroutines:
!          dqdag
```

```
!
function yzx_ppwg_ps()
    !modules
    use const, only          : pi,cj,Z0,mu_0,wp
    use slot_var, only       : Lx,Lz,w,x00,x00_pp,x0,x0_pp
    use mom_var, only        : p,s,kp,ks,kp_sq,ks_sq
    use gen_var, only        : k0,omega
    use ppwg_var, only       : m,a1,kx,kx_sq
    use wg_var, only         : a
    use quad_var, only       : kw,kz_max,errrel,errabs,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod
    !declaration of functions
    real(wp), external       ::  yzx_ppwg_ps_p_revised_c
    real(wp), external       ::  yzx_ppwg_ps_p
    !list of calling arguments
    complex(wp)              ::  yzx_ppwg_ps
    !list of local parameters
    integer,parameter        ::  irule=2
    real(wp),parameter       ::  upper=pi
    real(wp),parameter       ::  lower=0.0
    !list of local variables
    complex(wp)              ::  sum
    real(wp)                 ::  re_val
    real(wp),dimension(1:2)  ::  im_val
    real(wp)                 ::  errest,w_m

    if(mod(p,2).eq.1) then !p odd
        yzx_ppwg_ps=(0.0_wp,0.0_wp)
    else
        !init. of aux. var
        x0_pp=x0-(a-a1)/2.0_wp
        x00_pp=x00-(a-a1)/2.0_wp
        kp=p*pi/(2.0_wp*Lz)
        ks=s*pi/(2.0_wp*Lx)
        kp_sq=kp**2
        ks_sq=ks**2
        kw=Lz
        kz_max=max(k0,kp)
        sum=(0.0_wp,0.0_wp)
        do m=0,100 !100 gives good conv. unless you test a "free space" case
            kx=(m*pi/a1)
            kx_sq=kx**2
            w_m=dsin(kx*w/2.0_wp)*dcos(kx*x0_pp)/(ks_sq-kx_sq)
            if(mod(s,2).eq.1) then !s odd
                w_m=w_m*dcos(kx*Lx)*dsin(kx*x00_pp)
            else    !s even
                w_m=-w_m*dcos(kx*x00_pp)*dsin(kx*Lx)
            end if
            !correct w_m in case m<1
            if (m.eq.0) then
                w_m=w_m*0.5_wp
            end if

            !real part of revised contour
            if(kx.lt.k0) then
                c_flag=1
```

```
                call dqdag(yzx_ppwg_ps_p_revised_c, lower, upper,&
                            errabs, errrel, irule, re_val,errest)
            else
                re_val=0.0_wp
            end if
            !imaginary part of revised contour
            c_flag=2
            call dqdag(yzx_ppwg_ps_p_revised_c, lower, upper,&
                        errabs, errrel, irule, im_val(1),errest)
            !integration by weighted averages from kz=2kz_max->infty
            call int_wavg(yzx_ppwg_ps_p,2.0_wp*kz_max,-2,2,im_val(2))

            sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
        enddo !m loop

        yzx_ppwg_ps=-4.0_wp*Z0*p*s*pi/(a1*mu_0*omega*Lx*Lz)*sum

    endif
end function yzx_ppwg_ps

!#############################################################################
!
!  Name:
!          yzx_ppwg_ps_p_revised_c
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Auxiliary function used by the function yzx_ppwg_ps. Evaluates
!          the kernel used to compute the exterior cross coupling
!          contribution to the fields in a transverse slot due to currents
!          in a longitudinal slot. The evaluation is perfomed along a
!          deformed contour in the complex kz plane.
!
!  Usage:
!          yzx_ppwg_ps_p_revised_c(theta)
!
!  Arguments:
!   theta   - Independent variable (Input)
!
!  Modules:
!          const,quad_var,mom_var,gen_var,ppwg_var
!
function yzx_ppwg_ps_p_revised_c(theta)
    !modules
    use const, only : cj,wp
    use gen_var, only : k0,k0_sq
    use mom_var, only : kp_sq
    use ppwg_var, only : kx_sq,ky,kz,kz_sq
    use quad_var, only : kw,kz_max,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,cdsin,dcos,cdsqrt,dcmplx
    !list of calling arguments
    real(wp)    ::  yzx_ppwg_ps_p_revised_c
    !list of local variables
    complex(wp) ::  tmp
```

```
    real(wp)    ::  theta,csth,snth

    csth=dcos(theta)
    snth=dsin(theta)

    !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
    kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
    kz_sq=kz**2
    ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
    tmp=kz*cdsin(kw*kz)/ky/(kp_sq-kz_sq)*(kz_max*snth+0.05_wp*k0*cj*csth)
    if(c_flag.eq.1) then
        yzx_ppwg_ps_p_revised_c=dreal(tmp)
    else
        yzx_ppwg_ps_p_revised_c=dimag(tmp)
    endif

end function yzx_ppwg_ps_p_revised_c

!##############################################################################
!
!  Name:
!           yzx_ppwg_ps_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yzx_ppwg_ps. Evaluates
!           the kernel used to compute the exterior cross coupling
!           contribution to the fields in a transverse slot due to currents
!           in a longitudinal slot. Used in the interval from kz=kz_max->infty
!           where kz_max is chosen so all singular points are avoided.
!
!  Usage:
!           yzx_ppwg_ps_p(kz)
!
!  Arguments:
!   kz      - Independent variable (Input)
!
!  Modules:
!           const,quad_var,mom_var,gen_var,ppwg_var
!
function yzx_ppwg_ps_p(kz)
    !modules
    use const, only : wp
    use gen_var, only : k0_sq
    use mom_var, only : kp_sq
    use ppwg_var, only : kx_sq
    use quad_var, only : kw
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt
    !declaration of functions
    real(wp)        ::  dsinc
    !list of calling arguments
    real(wp)    ::  yzx_ppwg_ps_p
    real(wp),intent(in) ::  kz
    !list of local variables
```

```
    real(wp)    ::  kz_sq

    kz_sq=kz**2
    yzx_ppwg_ps_p=kz*dsinc(kw*kz)/&
                dsqrt(kz_sq+kx_sq-k0_sq)/(kp_sq-kz_sq)

end function yzx_ppwg_ps_p

!#############################################################################
!
!  Name:
!           yxz_ppwg_rq
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exterior cross coupling contribution to the fields
!           in a longitudinal slot due to currents in a transverse slot. Used
!           in case the slot radiates into a PP waveguide.
!
!  Usage:
!           yxz_ppwg_rq()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
!
!  Subroutines:
!           yxz_ppwg_rq_p_revised_c,yxz_ppwg_rq_p,int_wavg
!
!  IMSL subroutines:
!           dqdag
!
function yxz_ppwg_rq()
    !modules
    use const,      only    :   pi,cj,Z0,mu_0,wp
    use slot_var,   only    :   Lx,Lz,w,x00,x00_pp,x0,x0_pp
    use mom_var,    only    :   r,q,kq,kr,kq_sq,kr_sq
    use gen_var,    only    :   k0,omega
    use ppwg_var,   only    :   m,a1,kx,kx_sq
    use wg_var,     only    :   a
    use quad_var,   only    :   kw,kz_max,errrel,errabs,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod
    !declaration of functions
    real(wp), external      ::  yxz_ppwg_rq_p_revised_c
    real(wp), external      ::  yxz_ppwg_rq_p
    !list of calling arguments
    complex(wp)             ::  yxz_ppwg_rq
    !list of local parameters
    integer,parameter       ::  irule=2
    real(wp),parameter      ::  upper=pi
    real(wp),parameter      ::  lower=0.0
    !list of local variables
    complex(wp)             ::  sum
```

```
    real(wp)                :: re_val
    real(wp),dimension(1:3) :: im_val
    real(wp)                :: errest,w_m

    if(mod(q,2).eq.1) then !q odd
        yxz_ppwg_rq=(0.0_wp,0.0_wp)
    else
        !init. of aux. var
        x0_pp=x0-(a-a1)/2.0_wp
        x00_pp=x00-(a-a1)/2.0_wp
        kq=q*pi/(2.0_wp*Lz)
        kr=r*pi/(2.0_wp*Lx)
        kq_sq=kq**2
        kr_sq=kr**2
        kz_max=0.95_wp*max(k0,kq)

        sum=(0.0_wp,0.0_wp)
        do m=0,100 !100 gives good conv. unless you test a "free space" case
            kx=(m*pi/a1)
            kx_sq=kx**2
            w_m=kx*dcos(kx*x0_pp)/(kr_sq-kx_sq)
            if(mod(r,2).eq.1) then !r odd
                w_m=w_m*dcos(kx*Lx)*dsin(kx*x00_pp)
            else    !r even
                w_m=-w_m*dcos(kx*x00_pp)*dsin(kx*Lx)
            end if
            !correct w_m in case m<1
            if (m.eq.0) then
                w_m=w_m*0.5_wp
            end if

            !real part of revised contour
            if(kx.lt.k0) then
                c_flag=1
                call dqdag(yxz_ppwg_rq_p_revised_c, lower, upper,&
                            errabs, errrel, irule, re_val,errest)
            else
                re_val=0.0_wp
            end if
            !imaginary part of revised contour
            c_flag=2
            call dqdag(yxz_ppwg_rq_p_revised_c, lower, upper,&
                        errabs, errrel, irule, im_val(1),errest)
            !integration by weighted averages from kz=2kz_max->infty
            kw=Lz-w/2.0_wp
            call int_wavg(yxz_ppwg_rq_p,2.0_wp*kz_max,-3,1,im_val(2))
            kw=Lz+w/2.0_wp
            call int_wavg(yxz_ppwg_rq_p,2.0_wp*kz_max,-3,1,im_val(3))

            sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)-im_val(3)))
        enddo !m loop

        yxz_ppwg_rq=2.0_wp*Z0*r*q*pi/(a1*mu_0*omega*Lx*Lz)*sum

    endif
end function yxz_ppwg_rq

!###############################################################################
!
```

```
!  Name:
!          yxz_ppwg_rq_p_revised_c
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Auxiliary function used by the function yxz_ppwg_rq. Evaluates
!          the kernel used to compute the exterior cross coupling
!          contribution to the fields in a longitudinal slot due to currents
!          in a transverse slot. The evaluation is perfomed along a
!          deformed contour in the complex kz plane.
!
!  Usage:
!          yxz_ppwg_rq_p_revised_c(theta)
!
!  Arguments:
!   theta   - Independent variable (Input)
!
!  Modules:
!          const,quad_var,mom_var,gen_var,ppwg_var
!
function yxz_ppwg_rq_p_revised_c(theta)
    !modules
    use const, only : cj,wp
    use gen_var, only : k0,k0_sq
    use mom_var, only : kq_sq
    use slot_var, only      : Lz,w
    use ppwg_var, only : kx_sq,ky,kz,kz_sq
    use quad_var, only : kz_max,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,cdsin,dcos,cdsqrt,dcmplx
    !list of calling arguments
    real(wp)    ::  yxz_ppwg_rq_p_revised_c
    !list of local variables
    complex(wp) ::  tmp
    real(wp)    ::  theta,csth,snth,kw1,kw2

    csth=dcos(theta)
    snth=dsin(theta)
    kw1=w/2.0_wp-Lz
    kw2=w/2.0_wp+Lz

    !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
    kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
    kz_sq=kz**2
    ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))

    tmp=(cdcos(kz*kw1)-cdcos(kz*kw2))/&
            ky/(kq_sq-kz_sq)*(kz_max*snth+0.05_wp*k0*cj*csth)

    if(c_flag.eq.1) then
        yxz_ppwg_rq_p_revised_c=dreal(tmp)
    else
        yxz_ppwg_rq_p_revised_c=dimag(tmp)
    endif

```

```
end function yxz_ppwg_rq_p_revised_c

!###########################################################################
!
!  Name:
!           yxz_ppwg_rq_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yxz_ppwg_rq. Evaluates
!           the kernel used to compute the exterior cross coupling
!           contribution to the fields in a longitudinal slot due to currents
!           in a transverse slot. Used in the interval from kz=kz_max->infty
!           where kz_max is chosen so all singular points are avoided.
!
!  Usage:
!           yxz_ppwg_rq_p(kz)
!
!  Arguments:
!   kz      - Independent variable (Input)
!
!  Modules:
!           const,quad_var,mom_var,gen_var,ppwg_var
!
function yxz_ppwg_rq_p(kz)
    !modules
    use const, only : wp
    use gen_var, only : k0_sq,k0
    use mom_var, only : kq_sq
    use ppwg_var, only : kx,kx_sq
    use quad_var, only : kw
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,dcos
    !list of calling arguments
    real(wp)            ::  yxz_ppwg_rq_p
    real(wp),intent(in) ::  kz
    !list of local variables
    real(wp)            ::  kz_sq

    kz_sq=kz**2
    yxz_ppwg_rq_p=dcos(kz*kw)/&
                    dsqrt(kz_sq+kx_sq-k0_sq)/(kq_sq-kz_sq)

end function yxz_ppwg_rq_p

!###########################################################################
!
!  Name:
!           yxx_ppwg_rs
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exterior self contribution to the fields in a
```

```
!           transverse slot. Used in case the slot radiates into a PP
!           waveguide.
!
!  Usage:
!           yxx_ppwg_rs()
!
!  Arguments:
!
!  Modules:
!           const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
!
!  Subroutines:
!           yxx_ppwg_rs_p_revised_c,yxx_ppwg_rs_p,int_wavg
!
!  IMSL subroutines:
!           dqdag
!
function yxx_ppwg_rs()
    !modules
    use const, only         : pi,cj,Z0,mu_0,wp
    use slot_var, only      : Lx,w,x00,x00_pp
    use mom_var, only       : r,s,kr_sq,ks_sq
    use gen_var, only       : k0,k0_sq,omega
    use ppwg_var, only      : m,a1,kx,kx_sq
    use wg_var, only        : a
    use quad_var, only      : kw,kz_max,errrel,errabs,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod
    !declaration of functions
    real(wp), external          :: yxx_ppwg_rs_p_revised_c
    real(wp), external          :: yxx_ppwg_rs_p
    !list of calling arguments
    complex(wp)             :: yxx_ppwg_rs
    !list of local parameters
    integer,parameter       :: irule=2
    real(wp),parameter      :: upper=pi
    real(wp),parameter      :: lower=0.0
    !list of local variables
    complex(wp)             :: sum
    real(wp),dimension(1:2) :: im_val
    real(wp)                :: re_val
    real(wp)                :: errest,w_m

    if(mod((r+s),2).eq.1) then
            yxx_ppwg_rs=(0.0_wp,0.0_wp)
    else
        !init. of aux. var
        kz_max=k0
        kw=w/2.0_wp
        x00_pp=x00-(a-a1)/2.0_wp
        kr_sq=(r*pi/(2.0_wp*Lx))**2
        ks_sq=(s*pi/(2.0_wp*Lx))**2
        sum=(0.0_wp,0.0_wp)
        do m=0,100 !100 gives good conv. unless you test a "free space" case
            kx=(m*pi/a1)
            kx_sq=kx**2
            w_m=(k0_sq-kx_sq)/((kr_sq-kx_sq)*(ks_sq-kx_sq))
```

```
            if(mod(r,2).eq.1) then !r and s odd
                w_m=w_m*(dcos(kx*Lx)*dsin(kx*x00_pp))**2
            else    !r and s even
                w_m=w_m*(dcos(kx*x00_pp)*dsin(kx*Lx))**2
            end if
            !correct w_m in case m<1
            if (m.eq.0) then
                w_m=w_m*0.5_wp
            end if
            !real part of revised contour
            if(kx.lt.k0) then
                c_flag=1
                call dqdag(yxx_ppwg_rs_p_revised_c, lower, upper,&
                            errabs, errrel, irule, re_val,errest)
            else
                re_val=0.0_wp
            end if
            !imaginary part of revised contour
            c_flag=2
            call dqdag(yxx_ppwg_rs_p_revised_c, lower, upper,&
                        errabs, errrel, irule, im_val(1),errest)
            !integration by weighted averages from kz=2kz_max->infty
            call int_wavg(yxx_ppwg_rs_p,2.0_wp*kz_max,-2,2,im_val(2))

            sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
        enddo !m loop

        yxx_ppwg_rs=2.0_wp*Z0*w*r*s*pi/(Lx**2*a1*mu_0*omega)*sum

    endif
end function yxx_ppwg_rs

!###############################################################################
!
!  Name:
!           yxx_ppwg_rs_p_revised_c
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yxx_ppwg_rs. Evaluates
!           the kernel used to compute the exterior self contribution to the
!           fields in a transverse slot. The evaluation is perfomed along a
!           deformed contour in the complex kz plane.
!
!  Usage:
!           yxx_ppwg_rs_p_revised_c(theta)
!
!  Arguments:
!   theta   - Independent variable (Input)
!
!  Modules:
!           const,quad_var,mom_var,gen_var,ppwg_var
!
function yxx_ppwg_rs_p_revised_c(theta)
    !modules
    use const, only : cj,wp
    use gen_var, only : k0,k0_sq
```

```
        use slot_var, only : w
        use ppwg_var, only : kx_sq,ky,kz,kz_sq
        use quad_var, only : kw,kz_max,c_flag
        !implicit statement
        implicit none
        !declaration of intrinsic functions
        intrinsic dsin,cdsin,dcos,cdsqrt,dcmplx
        !list of calling arguments
        real(wp)    ::  yxx_ppwg_rs_p_revised_c
        real(wp),intent(in) ::  theta
        !list of local variables
        complex(wp) ::  tmp
        real(wp)    ::  csth,snth

        csth=dcos(theta)
        snth=dsin(theta)

        !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
        kz=kz_max*(1.0_wp-csth)+0.05_wp*k0*cj*snth
        kz_sq=kz**2
        ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
        tmp=cdsin(kw*kz)/(kw*kz)/ky*(kz_max*snth+0.05_wp*k0*cj*csth)

        if(c_flag.eq.1) then
            yxx_ppwg_rs_p_revised_c=dreal(tmp)
        else
            yxx_ppwg_rs_p_revised_c=dimag(tmp)
        endif

    end function yxx_ppwg_rs_p_revised_c

    !##############################################################################
    !
    !  Name:
    !           yxx_ppwg_rs_p
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Auxiliary function used by the function yxx_ppwg_rs. Evaluates
    !           the kernel used to compute the exterior self contribution to the
    !           fields in a longitudinal slot. Used in the interval from
    !           kz=kz_max->infty where kz_max is chosen so all singular points
    !           are avoided.
    !
    !  Usage:
    !           yxx_ppwg_rs_p(kz)
    !
    !  Arguments:
    !   kz      - Independent variable (Input)
    !
    !  Modules:
    !           const,quad_var,gen_var,ppwg_var
    !
    function yxx_ppwg_rs_p(kz)
        !modules
        use const, only : wp
        use gen_var, only : k0_sq
```

```
    use ppwg_var, only : kx_sq
    use quad_var, only : kw
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt
    !declaration of functions
    real(wp)            ::  dsinc
    !list of calling arguments
    real(wp)            ::  yxx_ppwg_rs_p
    real(wp),intent(in) ::  kz

    yxx_ppwg_rs_p=dsinc(kw*kz)/dsqrt(kz**2+kx_sq-k0_sq)

end function yxx_ppwg_rs_p

!###############################################################################
!
!  Name:
!          yzz_ppwg_pq
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Computes the exterior self contribution to the fields in a
!          longitudinal slot. Used in case the slot radiates into a PP
!          waveguide.
!
!  Usage:
!          yzz_ppwg_pq()
!
!  Arguments:
!
!  Modules:
!          const,quad_var,slot_var,mom_var,gen_var,ppwg_var,wg_var
!
!  Subroutines:
!          yzz_ppwg_pq_p_revised_c,int_yzz_ppwg_pq_inf,int_wavg,dsinc
!
!  IMSL subroutines:
!          dqdag
!
function yzz_ppwg_pq()
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use slot_var, only : Lz,w,x0,x0_pp
    use mom_var, only : p,q,kp,kq,kp_sq,kq_sq
    use gen_var, only : k0,omega
    use ppwg_var, only : m,a1,kx,kx_sq
    use wg_var, only : a
    use quad_var, only : kw,kz_max,errrel,errabs,c_flag
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,mod,max
    !declaration of functions
    real(wp), external          ::  yzz_ppwg_pq_p_revised_c
    real(wp)                    ::  dsinc,int_yzz_ppwg_pq_inf
```

```
    !list of calling arguments
    complex(wp)              ::  yzz_ppwg_pq
    !list of local parameters
    integer,parameter        ::  irule=2
    real(wp),parameter       ::  upper=pi
    real(wp),parameter       ::  lower=0.0_wp
    !list of local variables
    complex(wp)              ::  sum
    real(wp)                 ::  re_val
    real(wp),dimension(1:2)  ::  im_val
    real(wp)                 ::  errest,w_m

    if(mod((p+q),2).eq.1) then
            yzz_ppwg_pq=(0.0_wp,0.0_wp)
    else
        !init. of aux. var
        kp=p*pi/(2.0_wp*Lz)
        kq=q*pi/(2.0_wp*Lz)
        kp_sq=kp**2
        kq_sq=kq**2
        kz_max=0.95_wp*max(k0,kp,kq)
        kw=2.0_wp*Lz
        x0_pp=x0-(a-a1)/2.0_wp
        sum=(0.0_wp,0.0_wp)
        do m=0,100
            kx=(m*pi/a1)
            kx_sq=kx**2
            w_m=dsinc(kx*w/2.0_wp)*(dcos(kx*x0_pp))**2
            !correct w_m in case m<1
            if (m.eq.0) then
                w_m=w_m*0.5
            end if
            !real part of revised contour
            if(kx.lt.k0) then
            c_flag=1
            call dqdag(yzz_ppwg_pq_p_revised_c, lower, upper,&
                        errabs, errrel, irule, re_val,errest)
            else
            re_val=0.0_wp
            end if
            !imaginary part of revised contour
            c_flag=2
            call dqdag(yzz_ppwg_pq_p_revised_c, lower, upper,&
                        errabs, errrel, irule, im_val(1),errest)
            !integration of sommerfeld int. by weighted averages
            !from kz=2kz_max->infty
            im_val(2)=int_yzz_ppwg_pq_inf(2.0*kz_max)

            sum=sum+w_m*(re_val+cj*(im_val(1)+im_val(2)))
        enddo !m loop

        yzz_ppwg_pq=-Z0*w*p*q*pi/(a1*mu_0*omega*Lz**2)*sum

    endif
end function yzz_ppwg_pq

!###############################################################################
!
!  Name:
```

```
!           yzz_ppwg_pq_p_revised_c
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yzz_ppwg_pq. Evaluates
!           the kernel used to compute the exterior self contribution to the
!           fields in a longitudinal slot. The evaluation is perfomed along a
!           deformed contour in the complex kz plane.
!
!  Usage:
!           yzz_ppwg_pq_p_revised_c(theta)
!
!  Arguments:
!   theta   - Independent variable (Input)
!
!  Modules:
!           const,quad_var,mom_var,gen_var,ppwg_var
!
function yzz_ppwg_pq_p_revised_c(theta)
    !modules
    use const, only : wp,cj
    use gen_var, only : k0,k0_sq
    use ppwg_var, only : kx_sq
    use quad_var, only : kw,kz_max,c_flag
    use mom_var, only : p,q,kp_sq,kq_sq
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,cdcos,cdsqrt,dcmplx
    !list of calling arguments
    real(wp)            ::  yzz_ppwg_pq_p_revised_c
    real(wp),intent(in) ::  theta
    !list of local variables
    complex(wp)         ::  ky,kz,kz_sq,tmp
    real(wp)            ::  cos_term,sin_term

    cos_term=dcos(theta)
    sin_term=dsin(theta)

    !change of var: kz -> kzmax*(1.0-cos(theta))+cj*0.05*k0*sin(theta)
    kz=kz_max*(1.0_wp-cos_term)+0.05_wp*k0*cj*sin_term
    kz_sq=kz**2
    ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz_sq))
    tmp=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)/ky*&
            (kz_max*sin_term+0.05_wp*k0*cj*cos_term)

    if(mod(p,2).eq.1) then !p and q odd
        tmp=tmp*(cdcos(1.0_wp*kz*kw)+1.0_wp)
    else    !p and q even
        tmp=tmp*(1.0_wp-cdcos(1.0_wp*kz*kw))
    end if

    if(c_flag.eq.1) then
        yzz_ppwg_pq_p_revised_c=dreal(tmp)
    else
        yzz_ppwg_pq_p_revised_c=dimag(tmp)
    endif
```

```
end function yzz_ppwg_pq_p_revised_c

!##############################################################################
!
!  Name:
!           int_yzz_ppwg_pq_inf
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function yzz_ppwg_pq. Integrates
!           the monotomic and oscilating part of the kernel used to compute
!           the exterior self contribution to the fields in a longitudinal
!           slot. Used in the interval from kz=kz_low->infty where kz_low is
!           chosen so all singular points are avoided.
!
!  Usage:
!           int_yzz_ppwg_pq_inf(kz_low)
!
!  Arguments:
!   kz_low  - Lower integration limit (Input)
!
!  Modules:
!           const,mom_var,quad_var
!
!  Subroutines:
!           yzz_ppwg_pq_p_osc,yzz_ppwg_pq_p_mono,int_wavg
!
!  IMSL subroutines:
!           dqdagi
!
function int_yzz_ppwg_pq_inf(kz_low)
    !modules
    use const, only : wp
    use mom_var, only : p,q
    use quad_var, only : errrel,errabs
    !implicit statement
    implicit none
    !declaration of functions
    real(wp), external      ::  yzz_ppwg_pq_p_osc,yzz_ppwg_pq_p_mono
    !list of calling arguments
    real(wp)                ::  int_yzz_ppwg_pq_inf
    real(wp),intent(in)     ::  kz_low
    !list of local variables
    real(wp),dimension(1:2) ::  subint
    real(wp)                ::  errest

    ! integraton of monotonic part
    call dqdagi(yzz_ppwg_pq_p_mono,kz_low,1,&
                errabs,errrel,subint(1),errest)
    !integraton of oscillating part
    call int_wavg(yzz_ppwg_pq_p_osc,kz_low,-3,1,subint(2))

    if(mod(p,2).eq.1) then !p and q odd
        int_yzz_ppwg_pq_inf=subint(1)+subint(2)
    else    !r and s even
        int_yzz_ppwg_pq_inf=subint(1)-subint(2)
```

```
    end if

end function int_yzz_ppwg_pq_inf

!##############################################################################
!
!  Name:
!           yzz_ppwg_pq_p_osc
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function int_yzz_ppwg_pq_inf.
!           Evaluates the oscilating part of the kernel used to compute the
!           exterior self contribution to the fields in a longitudinal slot.
!
!  Usage:
!           yzz_ppwg_pq_p_osc(kz)
!
!  Arguments:
!   kz      - Independent variable (Input)
!
!  Modules:
!           const,quad_var,gen_var,ppwg_var,mom_var
!
function yzz_ppwg_pq_p_osc(kz)
    !modules
    use const, only : wp
    use gen_var, only : k0_sq
    use ppwg_var, only : kx_sq
    use quad_var, only : kw
    use mom_var, only : kp_sq,kq_sq
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt,dcos
    !list of calling arguments
    real(wp)            ::  yzz_ppwg_pq_p_osc
    real(wp),intent(in) ::  kz
    !list of local variables
    real(wp)            ::  kz_sq

    kz_sq=kz**2
    yzz_ppwg_pq_p_osc=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)&
                        /dsqrt(kz_sq+kx_sq-k0_sq)*dcos(kz*kw)

end function yzz_ppwg_pq_p_osc

!##############################################################################
!
!  Name:
!           yzz_ppwg_pq_p_mono
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function int_yzz_ppwg_pq_inf.
```

```
!           Evaluates the monotonic part of the kernel used to compute the
!           exterior self contribution to the fields in a longitudinal slot.
!
!  Usage:
!           yzz_ppwg_pq_p_mono(kz)
!
!  Arguments:
!   kz      - Independent variable (Input)
!
!  Modules:
!           const,quad_var,gen_var,ppwg_var,mom_var
!
function yzz_ppwg_pq_p_mono(kz)
    !modules
    use const, only : wp
    use gen_var, only : k0_sq
    use ppwg_var, only : kx_sq
    use quad_var, only : kw
    use mom_var, only : kp_sq,kq_sq
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt
    !list of calling arguments
    real(wp)            ::  yzz_ppwg_pq_p_mono
    real(wp),intent(in) ::  kz
    !list of local variables
    real(wp)            ::  kz_sq

    kz_sq=kz**2
    yzz_ppwg_pq_p_mono=(k0_sq-kz_sq)/(kp_sq-kz_sq)/(kq_sq-kz_sq)&
                        /dsqrt(kz_sq+kx_sq-k0_sq)

end function yzz_ppwg_pq_p_mono


!################################################################################
!
!  Name:
!           inv_r_int
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Sets up the integration of the function 1/R*(f(xi_0,zeta_0)+
!           xi*df_dxi(xi_0,zeta_0)+zeta*df_dzeta(xi_0,zeta_0)). The
!           integration is performed in polar coordinates over a rectangular
!           region. The intergration  is seperated into the four quadrants
!           around the singular point specified.
!
!  Usage:
!           inv_r_int(x0,x1,z0,z1)
!
!  Arguments:
!   x0,z0   - lower bounds for the rectangular region (Input)
!
!   x1,z1   - upper bounds for the rectangular region (Input)
!
```

```
!  Modules:
!          const,quad_var
!
!  Subroutines:
!          inv_r_quad_int
!
function inv_r_int(x0,x1,z0,z1)
    !modules
    use const, only : wp
    use quad_var, only : xi_0,zeta_0,xi_1,zeta_1
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic sum
    !declaration of functions
    real(wp)                    ::  inv_r_quad_int
    !list of calling arguments
    real(wp)                    ::  inv_r_int
    real(wp)                    ::  x0,x1,z0,z1
    !list of local variables
    real(wp), dimension(1:4)    ::  value

    !1st. quadrant
    xi_1=x1-xi_0
    zeta_1=z1-zeta_0
    value(1)=inv_r_quad_int()
    !2nd. quadrant
    xi_1=x1-xi_0
    zeta_1=z0-zeta_0
    value(2)=inv_r_quad_int()
    !3rd. quadrant
    xi_1=x0-xi_0
    zeta_1=z0-zeta_0
    value(3)=inv_r_quad_int()
    !4th. quadrant
    xi_1=x0-xi_0
    zeta_1=z1-zeta_0
    value(4)=inv_r_quad_int()

    !sum contribution from all four quadrants
    inv_r_int=sum(value)

end function inv_r_int

!###############################################################################
!
!  Name:
!          inv_r_quad_int
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Integrates the function 1/R*(f(xi_0,zeta_0)+
!          xi*df_dxi(xi_0,zeta_0)+zeta*df_dzeta(xi_0,zeta_0)) over one
!          quadrant with the singular point located at the origin. The
!          integration is seperated into 3 regions.
!
!  Usage:
```

```
!          inv_r_quad_int()
!
!  Arguments:
!
!  Modules:
!          const,quad_var
!
!  IMSL subroutines:
!          dqdag
!
function inv_r_quad_int()
    !modules
    use const, only : wp
    use quad_var, only : R1,R2,R3,xi_1,zeta_1,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic min,max,dabs,dsqrt,sum
    !declaration of functions
    real(wp),external           ::  reg1_p,reg2_p,reg3_p
    !list of calling arguments
    real(wp)        ::  inv_r_quad_int
    !list of local parameters
    integer, parameter          ::  irule=2
    !list of local variables
    real(wp), dimension(1:3)    ::  value
    real(wp), dimension(1:3)    ::  upper
    real(wp), dimension(1:3)    ::  lower
    real(wp)                    ::  errest

    R1=min(dabs(xi_1),dabs(zeta_1))
    R2=max(dabs(xi_1),dabs(zeta_1))
    R3=dsqrt(xi_1**2+zeta_1**2)
    upper=(/R1,R2,R3/)
    lower=(/0.0_wp,R1,R2/)

    !region 1
    CALL DQDAG (reg1_p, lower(1), upper(1),&
        ERRABS, ERRREL, IRULE, value(1),ERREST)
!   write(*,*) 'reg1 =',value(1),' reg1 errest = ',errest
    !region 2
    CALL DQDAG (reg2_p, lower(2), upper(2),&
        ERRABS, ERRREL, IRULE, value(2),ERREST)
!   write(*,*) 'reg2 =',value(2),' reg2 errest = ',errest
    !region 3
    CALL DQDAG (reg3_p, lower(3), upper(3),&
        ERRABS, ERRREL, IRULE, value(3),ERREST)
!   write(*,*) 'reg3 =',value(3),' reg3 errest = ',errest

    !summation of contribution from all three regions
    inv_r_quad_int=sum(value)

end function inv_r_quad_int

!###############################################################################
!
!  Name:
!          reg1_p
!
```

```
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function inv_r_quad_int.
!
!  Usage:
!           reg1_p(R)
!
!  Arguments:
!   R       - Independent variable, distance from the origin (Input)
!
!  Modules:
!           const,quad_var
!
function reg1_p(R)
    !modules
    use const, only : pi,wp
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsign
    !list of calling arguments
    real(wp)            ::  reg1_p
    real(wp),intent(in) ::  R

    reg1_p=f_0*pi/2.0_wp+df_dzeta*R*dsign(1.0_wp,zeta_1)&
            +df_dxi*R*dsign(1.0_wp,xi_1)

end function reg1_p

!#############################################################################
!
!  Name:
!           reg2_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function inv_r_quad_int.
!
!  Usage:
!           reg2_p(R)
!
!  Arguments:
!   R       - Distance from the origin (Input)
!
!  Modules:
!           const,quad_var
!
function reg2_p(R)
    !modules
    use const, only : pi,wp
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1,R1
    !implicit statement
    implicit none
    !declaration of intrinsic functions
```

```
    intrinsic dsin,dcos,dasin,dacos,dsign,dabs
    !list of calling arguments
    real(wp)            ::  reg2_p
    real(wp),intent(in) ::  R

    if(dabs(zeta_1).gt.dabs(xi_1)) then
        reg2_p=f_0*dasin(R1/R)&
            +df_dzeta*R*dsign(1.0_wp,zeta_1)*dsin(dasin(R1/R))&
            -df_dxi*R*dsign(1.0_wp,xi_1)*(dcos(dasin(R1/R))-1.0_wp)
    else
        reg2_p=f_0*(pi/2.0_wp-dacos(R1/R))&
            +df_dzeta*R*dsign(1.0_wp,zeta_1)*(1.0-dsin(dacos(R1/R)))&
            +df_dxi*R*dsign(1.0_wp,xi_1)*dcos(dacos(R1/R))
    endif

end function reg2_p

!##############################################################################
!
!  Name:
!           reg3_p
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Auxiliary function used by the function inv_r_quad_int.
!
!  Usage:
!           reg3_p(R)
!
!  Arguments:
!   R       - Distance from the origin (Input)
!
!  Modules:
!           const,quad_var
!
function reg3_p(R)
    !modules
    use const, only : pi,wp
    use quad_var, only : f_0,df_dxi,df_dzeta,xi_1,zeta_1,R2,R1
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,dasin,dacos,dsign,dabs
    !list of calling arguments
    real(wp)            ::  reg3_p
    real(wp),intent(in) ::  R

    if(abs(zeta_1).gt.abs(xi_1)) then
        reg3_p=f_0*(dasin(R1/R)-dacos(R2/R))+&
            df_dzeta*R*dsign(1.0_wp,zeta_1)*&
            (dsin(dasin(R1/R))-dsin(dacos(R2/R)))-&
            df_dxi*R*dsign(1.0_wp,xi_1)*&
            (dcos(dasin(R1/R))-dcos(dacos(R2/R)))
    else
        reg3_p=f_0*(dasin(R2/R)-dacos(R1/R))+&
            df_dzeta*R*dsign(1.0_wp,zeta_1)*&
            (dsin(dasin(R2/R))-dsin(dacos(R1/R)))-&
```

```
            df_dxi*R*dsign(1.0_wp,xi_1)*&
            (dcos(dasin(R2/R))-dcos(dacos(R1/R)))
    endif

end function reg3_p
```

```
!##############################################################################
!
!  Name:
!          int_wavg
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          Integrate a function f(x)=g(x)*w(kw*x) using method of
!          weighted averages.
!
!  Usage:
!          call int_w_avg (f, a, alfa,w_type,result)
!
!  Arguments:
!   f       - Asymtotic part user-supplied function to be integrated. The
!             form is:
!             f(x), where
!             x      - Independent variable.  (Input)
!             f      - The function value.  (Output)
!             f must be declared external in the calling program.
!   a       - Lower limit of integration.  (Input)
!   alfa    - asymtotic exponet of function f(x).  (Input)
!   w_type  - Type of weight function w used.  (Input)
!                 w_type        weight
!                   1         cos(kw*x)
!                   2         sin(kw*x)
!   result  - Estimate of the integral from a to infinity of g*w.  (Output)
!
!  Modules:
!          const,quad_var
!
!  IMSL subroutines:
!          dqdag
!
subroutine int_wavg(f,a,alfa,w_type,result)
    !modules
    use const, only : pi,cj,wp
    use quad_var, only : kw,mm,errrel,errabs
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic sin,cos,sqrt
    !declaration of functions
    real, external      :: f
    !list of calling arguments
    integer,intent(in)  :: w_type,alfa
    real(wp),intent(in) :: a
    real(wp),intent(out):: result
    !list of local parameters
    real(wp),allocatable,dimension(:,:) ::  I_ml,w_ml
    real(wp),allocatable,dimension(:)   ::  lambda_m
    integer, parameter  ::  irule=2
```

```
        integer            :: m,m0,l
        integer            :: status
        real(wp)           :: errest

        allocate(I_ml(mm,mm),w_ml(mm,mm),lambda_m(mm), STAT=status)

        !compute succesive zeroes of osc. function superior to lower
        !limit a, and I_ml(m,1)
        if(w_type.eq.1) then !w=cos
            m0=int((a+pi/(2.0_wp*kw))*kw/pi)
        else !w=sin
            m0=int(a*kw/pi)
        endif
        do m=1,mm
            if(w_type.eq.1) then !w=cos
                lambda_m(m)=(m0+m-0.5_wp)*pi/kw
            else !w=sin
                lambda_m(m)=(m0+m)*pi/kw
            endif
            call dqdag(f,a,lambda_m(m),errabs,errrel,irule,I_ml(m,1),errest)
        enddo
        !compute weights
        do l=1,mm-1
            do m=1,mm-l+1
            w_ml(m,l)=(lambda_m(1)/lambda_m(m))**(alfa+1-l)
            enddo
        enddo
        !compute partial values
        do l=1,mm-1
            do m=1,mm-l
                I_ml(m,l+1)=(w_ml(m,l)*I_ml(m,l)+w_ml(m+1,l)*I_ml(m+1,l))/&
                            (w_ml(m,l)+w_ml(m+1,l))
            enddo
        enddo

        result=I_ml(1,mm)

        deallocate(I_ml,w_ml,lambda_m, STAT=status)
    return
    end
```

```
    !############################################################################
    !
    !  Name:
    !           Y_fill
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Computes the matrix terms filling the lefthand side of the
    !           MoM matrix system.
    !
    !  Usage:
    !           Y_fill()
    !
    !  Arguments:
    !
    !  Modules:
    !           const,gen_var,slot_var,mom_var
```

```
!
!  Subroutines:
!            y_ext,yzx_ext_ps,yxz_ext_rq,yzz_int_pq,yxx_int_rs,
!            yzx_int_ps,yxz_int_rq,yxx_ppwg_rs,yzz_ppwg_pq,
!            yzx_ppwg_ps,yxz_ppwg_rq
!
subroutine Y_fill()
    !modules
    use const, only : wp
    use mom_var, only : r,s,p,q,Nz,Nx,Y
    use slot_var, only : Lz,Lx
    use gen_var, only : ext_type
    !implicit statement
    implicit none
    !declaration of functions
    complex(wp) :: y_ext,yzx_ext_ps,yxz_ext_rq
    complex(wp) :: yzz_int_pq,yxx_int_rs,yzx_int_ps,yxz_int_rq
    complex(wp) :: yxx_ppwg_rs,yzz_ppwg_pq,yzx_ppwg_ps,yxz_ppwg_rq

    !self terms for longitudinal part
    !write(*,*) 'Y(q,p) - self term for longi. slot'
    do q=1,Nz
        do p=1,q-1
        Y(q,p)=Y(p,q)
        enddo
        do p=q,Nz
            if(ext_type.eq.1) then
                Y(q,p)=y_ext(p,q,Lz)-yzz_int_pq()
            else
                Y(q,p)=yzz_ppwg_pq()-yzz_int_pq()
            endif
            !write(*,*) q,p,Y(q,p)
        enddo
    enddo

    !self terms for transverse part
    !write(*,*) 'Y(s,r) - self term for transv. slot'
    do s=1,Nx
        do r=1,s-1
        Y(s+Nz,r+Nz)=Y(r+Nz,s+Nz)
        enddo
        do r=s,Nx
            if(ext_type.eq.1) then
                Y(s+Nz,r+Nz)=-y_ext(r,s,Lx)-yxx_int_rs()
            else
                Y(s+Nz,r+Nz)=yxx_ppwg_rs()-yxx_int_rs()
            endif
            !write(*,*) s,r,Y(s+Nz,r+Nz)
        enddo
    enddo

    !cross terms in longitudinal part
    !write(*,*) 'Y(q,r) - cross term for longi. slot'
    do q=1,Nz
        do r=1,Nx
            if(ext_type.eq.1) then
                Y(q,r+Nz)=yxz_ext_rq()-yxz_int_rq()
            else
                Y(q,r+Nz)=yxz_ppwg_rq()-yxz_int_rq()
```

```
                endif
                !write(*,*) q,r,Y(q,r+Nz)
            enddo
        enddo

        !cross terms in transverse part
        !write(*,*) 'Y(s,p) - cross term for transv. slot'
        do s=1,Nx
            do p=1,Nz
                if(ext_type.eq.1) then
                    Y(s+Nz,p)=yzx_ext_ps()-yzx_int_ps()
                else
                    Y(s+Nz,p)=yzx_ppwg_ps()-yzx_int_ps()
                endif
                !write(*,*) s,p,Y(s+Nz,p)
            enddo
        enddo
    return
    end

    !############################################################################
    !
    !  Name:
    !           Y_fill_fnt_thk_long
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Computes the matrix terms filling the lefthand side of the
    !           MoM matrix system. To be used for investigation of a longitudinal
    !           slot having finite wall thickness.
    !
    !  Usage:
    !           Y_fill_fnt_thk_long()
    !
    !  Arguments:
    !
    !  Modules:
    !           const,gen_var,slot_var,mom_var
    !
    !  Subroutines:
    !           y_ext,yzz_int_pq,yzz_ppwg_pq,y11,y12
    !
    subroutine Y_fill_fnt_thk_long()
        !modules
        use const, only : wp
        use mom_var, only : p,q,Nz,Y
        use slot_var, only : Lz
        use gen_var, only : ext_type
        !implicit statement
        implicit none
        !declaration of functions
        complex(wp) :: y_ext
        complex(wp) :: yzz_int_pq
        complex(wp) :: y11,y12
        complex(wp) :: yzz_ppwg_pq

        do p=1,Nz
```

```
            do q=1,p-1
                Y(p,q)=Y(q,p)
            enddo
            do q=p,Nz
                Y(p,q)=y11(p,q,Lz)-yzz_int_pq()
            enddo
        enddo

        do p=1,Nz
            do q=1,Nz
                Y(p,q+Nz)=-y12(p,q,Lz)
            enddo
        enddo

        do p=1,Nz
            do q=1,Nz
                Y(p+Nz,q)=-y12(p,q,Lz)
            enddo
        enddo

        do p=1,Nz
            do q=1,p-1
                Y(p+Nz,q+Nz)=Y(q+Nz,p+Nz)
            enddo
            do q=p,Nz
                if(ext_type.eq.1) then
                    Y(p+Nz,q+Nz)=y_ext(p,q,Lz)+y11(p,q,Lz)
                else
                    Y(p+Nz,q+Nz)=yzz_ppwg_pq()+y11(p,q,Lz)
                endif
            enddo
        enddo

    return
    end

    !##############################################################################
    !
    !  Name:
    !           Y_fill_fnt_thk_trans
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Computes the matrix terms filling the lefthand side of the
    !           MoM matrix system. To be used for investigation of a transverse
    !           slot having finite wall thickness.
    !
    !  Usage:
    !           Y_fill_fnt_thk_trans()
    !
    !  Arguments:
    !
    !  Modules:
    !           const,gen_var,slot_var,mom_var
    !
    !  Subroutines:
    !           y_ext,yxx_int_rs,yxx_ppwg_rs,y11,y12
```

```
!
subroutine Y_fill_fnt_thk_trans()
    !modules
    use const, only : wp
    use mom_var, only : r,s,Nx,Y
    use slot_var, only : Lx
    use gen_var, only : ext_type
    !implicit statement
    implicit none
    !declaration of functions
    complex(wp) :: y_ext
    complex(wp) :: yxx_int_rs
    complex(wp) :: y11,y12
    complex(wp) :: yxx_ppwg_rs

    do r=1,Nx
        do s=1,r-1
            Y(r,s)=Y(s,r)
        enddo
        do s=r,Nx
            Y(r,s)=-y11(r,s,Lx)-yxx_int_rs()
        enddo
    enddo
    do r=1,Nx
        do s=1,Nx
        Y(r,s+Nx)=y12(r,s,Lx)
        enddo
    enddo
    do r=1,Nx
        do s=1,Nx
        Y(r+Nx,s)=y12(r,s,Lx)
        enddo
    enddo
    do r=1,Nx
        do s=1,r-1
        Y(r+Nx,s+Nx)=Y(s+Nx,r+Nx)
        enddo
        do s=r,Nx
            if(ext_type.eq.1) then
                Y(r+Nx,s+Nx)=-y_ext(r,s,Lx)-y11(r,s,Lx)
            else
                Y(r+Nx,s+Nx)=yxx_ppwg_rs()-y11(r,s,Lx)
            endif
        enddo
    enddo

return
end

!##############################################################################
!
!  Name:
!           y11
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Used in the investigation of a transverse or longitudinal slot
```

```
!           of any length having finite wall thickness. Computes the self
!           contribution to the fields at the two interfaces.
!
!  Usage:
!           y11(p,q,L)
!
!  Arguments:
!   p       - order of expansion function (Input)
!   q       - order of weighting function (Input)
!   L       - length of slot (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
function y11(p,q,L)
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use gen_var, only : k0_sq,omega
    use wg_var, only : t
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt, dtan, dtanh
    !list of calling arguments
    complex(wp)         :: y11
    integer ,intent(in) :: p,q
    real(wp),intent(in) :: L
    !list of local variables
    real(wp)    :: beta,beta_sq

    if (p.eq.q) then
        beta_sq=k0_sq-(p*pi/(2.0_wp*L))**2
        if (beta_sq.gt.0.0_wp) then
            beta=dsqrt(beta_sq)
            y11=cj*beta*L/(mu_0*omega)/dtan(beta*t)*Z0
        else
            beta=dsqrt(-beta_sq)
            y11=cj*beta*L/(mu_0*omega)/dtanh(beta*t)*Z0
        end if
    else
        y11=(0.0_wp,0.0_wp)
    end if

end function y11

!############################################################################
!
!  Name:
!           y12
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Used in the investigation of a transverse or longitudinal slot
!           of any length having finite wall thickness. Computes the cross
!           contribution to the fields at the two interfaces.
!
!  Usage:
```

```
!           y12(p,q,L)
!
!  Arguments:
!   p       - order of expansion function (Input)
!   q       - order of weighting function (Input)
!   L       - length of slot (Input)
!
!  Modules:
!           const,slot_var,mom_var,gen_var
!
function y12(p,q,L)
    !modules
    use const, only : pi,cj,Z0,mu_0,wp
    use gen_var, only : k0_sq,omega
    use wg_var, only : t
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsqrt, dsin, dsinh
    !list of calling arguments
    complex(wp)         :: y12
    integer ,intent(in) :: p,q
    real(wp),intent(in) :: L
    !list of local variables
    real(wp)    :: beta,beta_sq

    if (p.eq.q) then
        beta_sq=k0_sq-(p*pi/(2.0_wp*L))**2
        if (beta_sq.gt.0.0_wp) then
            beta=dsqrt(beta_sq)
            y12=cj*beta*L/(mu_0*omega)/dsin(beta*t)*Z0
        else
            beta=dsqrt(-beta_sq)
            y12=cj*beta*L/(mu_0*omega)/dsinh(beta*t)*Z0
        end if
    else
        y12=(0.0_wp,0.0_wp)
    end if

end function y12
```

```
!############################################################################
!
!  Name:
!           hinc_fill_fnt_thk_long
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exitations terms filling the righthand side of the
!           MoM matrix system.
!
!  Usage:
!           hinc_fill_fnt_thk_long()
!
!  Arguments:
!
!  Modules:
!           const,mom_var
```

```
!
!  Subroutines:
!           hz_inc_q
!
!  Remarks:
!           This routine is used invetigating a longitudinal slot including
!           finite wall thickness.
!
subroutine hinc_fill_fnt_thk_long()
    !modules
    use const, only : wp
    use mom_var, only : q,Nz,h_inc
    !implicit statement
    implicit none
    !declaration of functions
    complex(wp) :: hz_inc_q

    do q=1,Nz
        h_inc(q)=hz_inc_q()
    enddo
    do q=Nz+1,2*Nz
        h_inc(q)=(0.0_wp,0.0_wp)
    enddo

return
end

!###############################################################################
!
!  Name:
!           hinc_fill_fnt_thk_trans
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           Computes the exitations terms filling the righthand side of the
!           MoM matrix system.
!
!  Usage:
!           hinc_fill_fnt_thk_trans()
!
!  Arguments:
!
!  Modules:
!           const,mom_var
!
!  Subroutines:
!           hx_inc_s
!
!  Remarks:
!           This routine is used investigating a transverse slot including
!           finite wall thickness.
!
subroutine hinc_fill_fnt_thk_trans()
    !modules
    use const, only : wp
    use mom_var, only : s,Nx,h_inc
    !implicit statement
```

```
        implicit none
        !declaration of functions
        complex(wp) :: hx_inc_s

        do s=1,Nx
            h_inc(s)=hx_inc_s()
        enddo
        do s=Nx+1,2*Nx
            h_inc(s)=(0.0_wp,0.0_wp)
        enddo

    return
    end

    !##############################################################################
    !
    !  Name:
    !           hinc_fill
    !
    !  Revised:
    !           February 29, 2000
    !
    !  Purpose:
    !           Computes the exitations terms filling the righthand side of the
    !           MoM matrix system.
    !
    !  Usage:
    !           hinc_fill()
    !
    !  Arguments:
    !
    !  Modules:
    !           const,mom_var
    !
    !  Subroutines:
    !           hz_inc_q,hx_inc_s
    !
    subroutine hinc_fill()
        !modules
        use const, only : wp
        use mom_var, only : q,s,Nz,Nx,h_inc
        !implicit statement
        implicit none
        !declaration of functions
        complex(wp) :: hz_inc_q, hx_inc_s

        do q=1,Nz
            h_inc(q)=hz_inc_q()
        enddo

        do s=1,Nx
            h_inc(Nz+s)=hx_inc_s()
        enddo

    return
    end

    !##############################################################################
    !
```

```
! Name:
!           hz_inc_q
!
! Revised:
!           February 29, 2000
!
! Purpose:
!           Computes the q'th exitations terms for a longitudinal slot.
!
! Usage:
!           hz_inc_q()
!
! Arguments:
!
! Modules:
!           const,mom_var,slot_var,gen_var,wg_var
!
function hz_inc_q()
    !modules
    use const, only : pi,cj,wp
    use slot_var, only : Lz,x0
    use mom_var, only : q
    use gen_var, only : k0
    use wg_var, only : a,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,dsin
    !list of calling arguments
    complex(wp) ::  hz_inc_q

    hz_inc_q=pi*q/Lz*dcos(pi*x0/a)/(beta_10**2-(q*pi/(2.0_wp*Lz))**2)

    if(mod(q,2).eq.1) then
        hz_inc_q=-cj*hz_inc_q*dcos(beta_10*Lz)
    else
        hz_inc_q=hz_inc_q*dsin(beta_10*Lz)
    end if

end function hz_inc_q

!##############################################################################
!
! Name:
!           hx_inc_s
!
! Revised:
!           February 29, 2000
!
! Purpose:
!           Computes the s'th exitations terms for a transverse slot.
!
! Usage:
!           hx_inc_s()
!
! Arguments:
!
! Modules:
!           const,mom_var,slot_var,gen_var,wg_var
```

```
!
function hx_inc_s()
    !modules
    use const, only : pi,wp
    use mom_var, only : s
    use slot_var, only : Lx,x00
    use wg_var, only : a,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,dsin
    !list of calling arguments
    complex(wp) ::  hx_inc_s

    hx_inc_s=-beta_10*a*s/Lx/&
                ((s*pi/(2.0_wp*Lx))**2-(pi/a)**2)

    if(mod(s,2).eq.1) then !s odd
        hx_inc_s=hx_inc_s*dcos(pi*Lx/a)*dsin(pi*x00/a)
    else    !s even
        hx_inc_s=-hx_inc_s*dcos(pi*x00/a)*dsin(pi*Lx/a)
    end if

end function hx_inc_s
!################################################################################
!
!  Name:
!         backscat
!
!  Revised:
!         February 29, 2000
!
!  Purpose:
!         Computes the complex refelction coefficient B_10/A_10=s11 for
!         both longitudinal, transvers and t-slots
!
!  Usage:
!         backscat()
!
!  Arguments:
!
!  Modules:
!         const
!
!  Subroutines:
!         backscat_trans,backscat_long
!
function backscat()
    !modules
    use const, only : wp
    !implicit statement
    implicit none
    !declaration of functions
    complex(wp) ::  backscat_trans,backscat_long
    !list of calling arguments
    complex(wp) ::  backscat

    backscat=backscat_trans()+backscat_long()

```

```
end function backscat

!############################################################################
!
!  Name:
!           backscat_long
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           computes the reflection coefficient B_10/A_10=s11 for a
!           longitudinal slot
!
!  Usage:
!           backscat_long()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,mom_var,gen_var,wg_var
!
!  Subroutines:
!
function backscat_long()
    !modules
    use const, only :mu_0,pi,cj,Z0,wp
    use slot_var, only :w,Lz,x0
    use mom_var, only :p,Nz,E
    use gen_var, only : k0,omega
    use wg_var, only : a,b,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos
    !list of calling arguments
    complex(wp) ::  backscat_long
    !list of local variables
    complex(wp) :: sum, add_term

    if(Nz.gt.0) then
        sum=(0.0_wp,0.0_wp)
        do p=1,Nz
            add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-beta_10**2)
            if(mod(p,2).eq.1) then !p odd
                add_term=add_term*dcos(beta_10*Lz)
            else    !p even
                add_term=add_term*cj*dsin(beta_10*Lz)
            end if
        sum=sum+add_term
        enddo
        backscat_long=-cj*2.0_wp*pi**2*Z0/(omega*mu_0*a**2*b*Lz*beta_10)
        backscat_long=backscat_long*dsin(w*pi/(2.0*a))*dcos(x0*pi/a)*sum
    else
        backscat_long=(0.0_wp,0.0_wp)
    endif

end function backscat_long
```

```
!##############################################################################
!
!  Name:
!          backscat_trans
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          computes the reflection coefficient B_10/A_10=s11 for a
!          transverse slot
!
!  Usage:
!          backscat_trans()
!
!  Arguments:
!
!  Modules:
!          const,slot_var,mom_var,gen_var,wg_var
!
!  Subroutines:
!
function backscat_trans()
    !modules
    use const, only : mu_0,pi,cj,Z0,wp
    use slot_var, only : w,Lx,x0,x00
    use mom_var, only : r,Nx,Nz,E
    use gen_var, only : k0,omega
    use wg_var, only : a,b,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos
    !list of calling arguments
    complex(wp) ::  backscat_trans
    !list of local variables
    complex(wp) :: sum, add_term

    if(Nx.gt.0) then
        sum=(0.0_wp,0.0_wp)
        do r=1,Nx
            add_term=E(r+Nz)*r/((r*pi/(2.0_wp*Lx))**2-(pi/a)**2)
            if(mod(r,2).eq.1) then !r odd
                add_term=add_term*dcos(pi*Lx/a)*dsin(pi*x00/a)
            else    !r even
                add_term=-add_term*dcos(pi*x00/a)*dsin(pi*Lx/a)
            end if
            sum=sum+add_term
        enddo
        backscat_trans=-2.0_wp*pi**2/(omega*mu_0*a**2*b*Lx*beta_10)
        backscat_trans=backscat_trans*Z0*dsin(w/2.0_wp*beta_10)*sum
    else
        backscat_trans=(0.0_wp,0.0_wp)
    end if

end function backscat_trans

!##############################################################################
!
```

```
! Name:
!          forwardscat
!
! Revised:
!          February 29, 2000
!
! Purpose:
!          Computes the complex transmission coefficient C_10/A_10 for both
!          longitudinal, transvers and t-slots
!
! Usage:
!          forwardscat()
!
! Arguments:
!  forwardscat - Function value.  (Output)
!
! Modules:
!          const
!
! Subroutines:
!          forwardscat_trans,forwardscat_long
!
function forwardscat()
    !modules
    use const, only : wp
    !implicit statement
    implicit none
    !declaration of functions
    complex(wp) ::  forwardscat_trans,forwardscat_long
    !list of calling arguments
    complex(wp) ::  forwardscat

    forwardscat=forwardscat_trans()+forwardscat_long()

end function forwardscat

!##############################################################################
!
! Name:
!          forwardscat_long
!
! Revised:
!          February 29, 2000
!
! Purpose:
!          computes the complex transmission coefficient C_10/A_10 for a
!          longitudinal slot
!
! Usage:
!          forwardscat_long()
!
! Arguments:
!  forwardscat_long    - Function value.  (Output)
!
! Modules:
!          const,slot_var,mom_var,gen_var,wg_var
!
! Subroutines:
!
```

```
function forwardscat_long()
    !modules
    use const, only :mu_0,pi,cj,Z0,wp
    use slot_var, only :w,Lz,x0
    use mom_var, only :p,Nz,E
    use gen_var, only : k0,omega
    use wg_var, only : a,b,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod
    !list of calling arguments
    complex(wp) ::  forwardscat_long
    !list of local variables
    complex(wp)     :: sum, add_term

    if(Nz.gt.0) then
        sum=(0.0_wp,0.0_wp)
        do p=1,Nz
            add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-beta_10**2)
            if(mod(p,2).eq.1) then !p odd
                add_term=add_term*dcos(beta_10*Lz)
            else    !p even
                add_term=-add_term*cj*dsin(beta_10*Lz)
            end if
        sum=sum+add_term
        enddo
        forwardscat_long=2.0_wp*pi**2*Z0/&
                     (cj*omega*mu_0*a**2*b*Lz*beta_10)
        forwardscat_long=forwardscat_long*dsin(w*pi/(2.0_wp*a))*&
                                         dcos(x0*pi/a)*sum
    else
        forwardscat_long=(0.0_wp,0.0_wp)
    endif

end function forwardscat_long

!##############################################################################
!
!  Name:
!           forwardscat_trans
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           computes the complex transmission coefficient C_10/A_10 for a
!           transverse slot
!
!  Usage:
!           forwardscat_trans()
!
!  Arguments:
!   forwardscat_trans   - Function value.  (Output)
!
!  Modules:
!           const,slot_var,mom_var,gen_var,wg_var
!
!  Subroutines:
```

```
!
function forwardscat_trans()
    !modules
    use const, only : mu_0,pi,cj,Z0,wp
    use slot_var, only : w,Lx,x0,x00
    use mom_var, only : r,Nx,Nz,E
    use gen_var, only : k0,omega
    use wg_var, only : a,b,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dsin,dcos,mod
    !list of calling arguments
    complex(wp) ::  forwardscat_trans
    !list of local variables
    complex(wp)     :: sum, add_term

    if(Nx.gt.0) then
        sum=(0.0_wp,0.0_wp)
        do r=1,Nx
            add_term=E(r+Nz)*r/((r*pi/(2.0_wp*Lx))**2-(pi/a)**2)
            if(mod(r,2).eq.1) then !r odd
                add_term=add_term*dcos(pi*Lx/a)*dsin(pi*x00/a)
            else    !r even
                add_term=-add_term*dcos(pi*x00/a)*dsin(pi*Lx/a)
            end if
            sum=sum+add_term
        enddo
        forwardscat_trans=2.0_wp*pi**2/(omega*mu_0*a**2*b*Lx*beta_10)
        forwardscat_trans=forwardscat_trans*Z0*dsin(w/2.0_wp*beta_10)*sum
    else
        forwardscat_trans=(0.0_wp,0.0_wp)
    end if

end function forwardscat_trans
```

```

!##############################################################################
!
!  Name:
!           write_exp_coeff
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine writes the expansion coefficients in the matrix E
!
!  Usage:
!           call write_exp_coeff()
!
!  Arguments:
!
!  Modules:
!           const,mom_var
!
!  Subroutines:
!
!  Remarks:
!           The output is printed to standard output.
```

```
!
subroutine write_exp_coeff()
    !modules
    use const, only : wp
    use mom_var, only : Nz,Nx,E
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dreal,dimag,cdabs,dlog10
    !list of local variables
    integer i

    do i=1,Nz
        write(*,1077) i,dreal(E(i)),dimag(E(i)),&
            20.0_wp*dlog10(cdabs(E(i)))
    enddo
    do i=1,Nx
        write(*,1077) i, dreal(E(Nz+i)), dimag(E(Nz+i)),&
            20.0*dlog10(cdabs(E(Nz+i)))
    enddo

1077 format(2x,I15,3(2x,F15.6))
return
end

!###############################################################################
!
!  Name:
!           F_sph_fs
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine computes the electric vector potential F in
!           spherical coordinates for a slot radiating into half space
!
!  Usage:
!           call F_sph_fs()
!
!  Arguments:
!
!  Modules:
!           const,gen_var,radfield_var,slot_var,mom_var
!
!  Subroutines:
!           xyz2sph,dsinc
!
subroutine F_sph_fs()
    !modules
    use const, only : pi,cj,wp
    use gen_var, only : k0,k0_sq
    use slot_var, only  : w,Lx,Lz
    use mom_var, only   : p,r,Nz,Nx,E
    use radfield_var, only  : F_sph,F_xyz,sph
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,dsin,cdexp
```

```
        !declaration of functions
        real(wp)        ::  dsinc
        !list of local variables
        real(wp)    ::  csph,csth,snth
        complex(wp) ::  sum, add_term

        !init. of aux. variables
        csph=dcos(sph(3))
        csth=dcos(sph(2))
        snth=dsin(sph(2))
        F_xyz=(/(0.0_wp,0.0_wp),(0.0_wp,0.0_wp),(0.0_wp,0.0_wp)/)

        !computation of F_x
        if(Nx.gt.0) then
            sum=(0.0_wp,0.0_wp)
            do r=1,Nx
                add_term=E(r+Nz)*r/&
                         ((r*pi/(2.0_wp*Lx))**2-k0_sq*csph**2*snth**2)
                if(mod(r,2).eq.1) then !r odd
                    add_term=add_term*dcos(k0*Lx*csph*snth)
                else    !r even
                    add_term=-add_term*cj*dsin(k0*Lx*csph*snth)
                end if
                sum=sum+add_term
            enddo
            F_xyz(1)=-w/2.0_wp*cdexp(-cj*k0*sph(1))/(Lx*sph(1))*&
                     cdexp(cj*k0*(w/2.0_wp-Lx)*csph*snth)*&
                     dsinc(w*k0/2.0_wp*csth)*sum
        end if

        !computation of F_z
        if(Nz.gt.0) then
            sum=(0.0_wp,0.0_wp)
            do p=1,Nz
                add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-k0_sq*csth**2)
                if(mod(p,2).eq.1) then !p odd
                    add_term=add_term*dcos(k0*Lz*csth)
                else    !p even
                    add_term=-add_term*cj*dsin(k0*Lz*csth)
                end if
            sum=sum+add_term
            enddo
            F_xyz(3)=w/2.0_wp*cdexp(-cj*k0*sph(1))/(Lz*sph(1))*&
                     dsinc(w*k0/2.0_wp*csph*snth)*sum

        endif

        !transform into spherical coor.
        call xyz2sph(F_xyz,sph,F_sph)

    return
    end

    !############################################################################
    !
    !  Name:
    !           F_sph_ppwg
    !
    !  Revised:
```

```
!           February 29, 2000
!
!  Purpose:
!           This subroutine computes the electric vector potential F in
!           spherical coordinates for a slot radiating into half space
!
!  Usage:
!           call F_sph_fs()
!
!  Arguments:
!
!  Modules:
!           const,gen_var,radfield_var,slot_var,mom_var
!
!  Subroutines:
!           xyz2sph,dsinc
!
subroutine F_sph_ppwg()
    !modules
    use const, only : pi,cj,wp
    use gen_var, only : k0,k0_sq
    use slot_var, only  : w,Lx,Lz,x00_pp,x0_pp
    use mom_var, only   : p,r,Nz,Nx,E
    use ppwg_var, only      : m,a1,b1,kx,kx_sq,ky
    use radfield_var, only  : F_sph,F_xyz,sph
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,dsin,cdexp
    !declaration of functions
    real(wp)        ::  dsinc
    !list of local variables
    real(wp)    ::  csph,csth,snth,kz,kz_sq,epsilon_m
    complex(wp) ::  sum, add_term

    !init. of aux. variables
    csph=dcos(sph(3))
    csth=dcos(sph(2))
    snth=dsin(sph(2))
    F_xyz=(/(0.0_wp,0.0_wp),(0.0_wp,0.0_wp),(0.0_wp,0.0_wp)/)

    m=1
    epsilon_m=1.0_wp
    kx=m*pi/a1
    kx_sq=kx**2
    kz=-k0*csth
    kz_sq=kz**2
    ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz**2))
    if (dimag(ky).gt.0.0_wp) then
        ky=-ky
    endif

    !computation of F_x
    if(Nx.gt.0) then
        sum=(0.0_wp,0.0_wp)
        do r=1,Nx
            add_term=E(r+Nz)*r/&
                     ((r*pi/(2.0_wp*Lx))**2-kx_sq)
            if(mod(r,2).eq.1) then !r odd
```

```
                    add_term=add_term*dcos(kx*Lx)*dsin(kx*x00_pp)
                else    !r even
                    add_term=-add_term*dcos(kx*x00_pp)*dsin(kx*Lx)
                end if
                sum=sum+add_term
            enddo
            F_xyz(1)=(-w*2.0_wp/(Lx*a1))*&
                     epsilon_m*dsinc(w*kz/2.0_wp)*&
                     cdexp(-cj*ky*b1)*&
                     kx/(k0_sq*csph**2*snth**2-kx_sq)*&
                     (cdexp(cj*k0*a1*csph*snth)*&
                     (-1.0_wp)**m-1.0_wp)*sum

                     !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&
        end if

        !computation of F_x
        if(Nz.gt.0) then
            sum=(0.0_wp,0.0_wp)
            do p=1,Nz
                add_term=E(p)*p/&
                         ((p*pi/(2.0_wp*Lz))**2-kz_sq)
                if(mod(p,2).eq.1) then !p odd
                    add_term=add_term*dcos(kz*Lz)
                else    !p even
                    add_term=-add_term*cj*dsin(kz*Lz)
                end if
                sum=sum+add_term
            enddo
            F_xyz(1)=F_xyz(1)+2.0_wp*(1.0_wp+cj)/(Lz*a1)*&
                    epsilon_m*kz*kx_sq/k0_sq/(k0_sq*csph**2*snth**2-kx_sq)*&
                    dsinc(w*kx/2.0_wp)*dcos(kx*x0_pp)*cdexp(-cj*ky*b1)*&
                    (cdexp(cj*k0*a1*csph*snth)*(-1.0_wp)**m-1.0_wp)*sum
                    !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&

        end if

        m=0
        epsilon_m=0.5_wp
        kx=m*pi/a1
        kx_sq=kx**2
        ky=cdsqrt(dcmplx(k0_sq-kx_sq-kz**2))
        if (dimag(ky).gt.0.0_wp) then
            ky=-ky
        endif

        !computation of F_z
        if(Nz.gt.0) then
            sum=(0.0_wp,0.0_wp)
            do p=1,Nz
                add_term=E(p)*p/((p*pi/(2.0_wp*Lz))**2-k0_sq*csth**2)
                if(mod(p,2).eq.1) then !p odd
                    add_term=add_term*dcos(kz*Lz)
                else    !p even
                    add_term=-add_term*cj*dsin(kz*Lz)
                end if
            sum=sum+add_term
            enddo
```

```
        F_xyz(3)=2.0_wp*cj*w/(Lz*a1)*&
                epsilon_m*((k0_sq-kz_sq)+cj*kz_sq)/k0_sq*&
                dsinc(w*kx/2.0_wp)*dcos(kx*x0_pp)*&
                                cdexp(-cj*ky*b1)*&
                cj*a1*dsinc(a1*k0*csph*snth)*sum
                !cdexp(-cj*k0*sph(1))/(4.0_wp*pi*sph(1))*&

    endif

    !transform into spherical coor.
    call xyz2sph(F_xyz,sph,F_sph)

return
end

!##############################################################################
!
!  Name:
!           rad_field
!
!  Revised:
!           February 29, 2000
!
!  Purpose:
!           This subroutine reads expansion coefficients from external file
!           "efield.dat" and computes the principle patterns from a slot
!           radiating into either half space or a PP waveguide.
!
!  Usage:
!           call rad_field()
!
!  Arguments:
!   plane   - Type of principle plane to be computed.  (Input)
!                  plane    principle plane
!                    1         E-plane
!                    2         H-plane
!
!  Modules:
!           const,gen_var,slot_var,radfield_var,ppwg_var,wg_var
!
!  Subroutines:
!           read_efield,F_sph_fs,F_sph_ppwg
!
!  Remarks:
!           The output is printed to standard output.
!
subroutine rad_field(plane)
    !modules
    use const, only : dtr,rtd,cj,wp,Z0,pi
    use gen_var, only   : lambda_0,k0,ext_type
    use slot_var, only      : Lx,w,x00,x00_pp,x0,x0_pp
    use radfield_var, only  : E_sph,F_sph,sph,H_sph
    use ppwg_var, only      : a1
    use wg_var, only        : a,lambda_g,beta_10
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dcos,dsin,cdexp,dcmplx,cdsqrt
    !declaration of functions
```

```
        real(wp)    ::  dsinc
        !list of calling arguments
        integer,intent(in)  ::  plane
        !list of local variables
        complex(wp) ::  arrayfactor_copol,arrayfactor_xpol
        integer     ::  i,n,nth
        real(wp)    ::  d

        call read_efield()
        x00=x0+w/2.0-Lx
        x0_pp=x0-(a-a1)/2.0_wp
        x00_pp=x00-(a-a1)/2.0_wp

        sph(1)=100.0_wp*lambda_0
        if(plane.eq.1) then
            sph(3)=90.0_wp*dtr
        else
            sph(2)=90.0_wp*dtr
        endif

        nth=179*4
        do i=1,nth,1
            if(plane.eq.1) then
                !sph(2)=dtr*i
                sph(2)=dtr*179.0_wp*dfloat(i)/dfloat(nth)
            else
                sph(3)=dtr*179.0_wp*dfloat(i)/dfloat(nth)
            endif
            if(ext_type.eq.1) then
                call F_sph_fs()
            else
                call F_sph_ppwg()
            endif
            E_sph(2)=-cj*k0*F_sph(3)
            E_sph(3)=cj*k0*F_sph(2)
            H_sph(2)=-E_sph(3)/Z0
            H_sph(3)=E_sph(2)/Z0

            d=lambda_g
            arrayfactor_copol=(0.0_wp,0.0_wp)
            arrayfactor_xpol=(0.0_wp,0.0_wp)
            do n=1,9
                 arrayfactor_copol=arrayfactor_copol+&
                            cdexp(cj*n*(k0*d*dcos(sph(2))-d*beta_10))
                 arrayfactor_xpol=arrayfactor_xpol+&
                            cdexp(cj*n*(k0*d*dcos(sph(2))-d*beta_10+1.0_wp*pi))
            enddo

            write(*,1078) rtd*sph(2),rtd*sph(3),&
                dreal(E_sph(2)*dconjg(H_sph(3))),&
                dreal(-E_sph(3)*dconjg(H_sph(2))),&
                dreal(arrayfactor_copol*dconjg(arrayfactor_copol)),&
                dreal(arrayfactor_xpol*dconjg(arrayfactor_xpol))
        enddo

    1078 format(2(2x,F7.2),4(2x,E15.6))
    return
    end
```

```
!##############################################################################
!
!  Name:
!          write_y_norm
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          This subroutine computes the equivalent normalized shunt
!          admittance. Only recommended for longitudinal slots.
!
!  Usage:
!          call write_y_norm()
!
!  Arguments:
!
!  Modules:
!          const,slot_var,gen_var
!
!  Subroutines:
!          backscat
!
!  Remarks:
!          The output is printed to standard output.
!
subroutine write_y_norm()
    !modules
    use const, only : wp
    use slot_var, only :Lz,x0
    use gen_var, only : lambda_0
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    intrinsic dreal,dimag
    !declaration of functions
    complex(wp)     ::  backscat
    !list of local variables
    complex(wp)     ::  R,y

    R=backscat()
    y=-2.0_wp*R/(1.0_wp+R)
    write(*,*) x0, 2.0_wp*Lz/lambda_0, dreal(y), dimag(y)

return
end

!##############################################################################
!
!  Name:
!          write_nw_par
!
!  Revised:
!          February 29, 2000
!
!  Purpose:
!          This subroutine computes the equivalent normalized
!          series impedance. It writes frequency, slot length, offset,
!          series impedance, backward and forward scattering coefficients,
```

```
!           and expansion coefficients.
!
!  Usage:
!           call write_nw_par()
!
!  Arguments:
!
!  Modules:
!           const,slot_var,gen_var,mom_var,ppwg_var
!
!  Subroutines:
!           backscat,forwardscat
!
!  Remarks:
!           The output is printed to standard output.
!
subroutine write_nw_par()
    !modules
    use const, only : rtd,cj,wp
    use slot_var, only : Lx,Lz,x0
    use ppwg_var, only : a1
    use gen_var, only : freq
    use mom_var, only : E
    !implicit statement
    implicit none
    !declaration of intrinsic functions
    complex backscat, forwardscat
    !list of local variables
    complex(wp) R,T,z

    R=backscat()
    T=forwardscat()
    z=2.0_wp*R/(1.0_wp-R)
    write(*,1075) freq,x0,a1,Lx,Lz,dreal(z),dimag(z),&
             dreal(R),dimag(R),dreal(T),dimag(T),E

1075 format(2x,EN15.6,4(2x,EN15.6),6(2x,E15.6),48(2x,E15.6),26(2x,E15.6))
return
end

```

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      =  25.00000E-3,
sw_type =  5
/
```