# Simulation of microwave beams with PROFUSION (2019 Edition)

Burkhard Plaum

February 6, 2019

# Contents

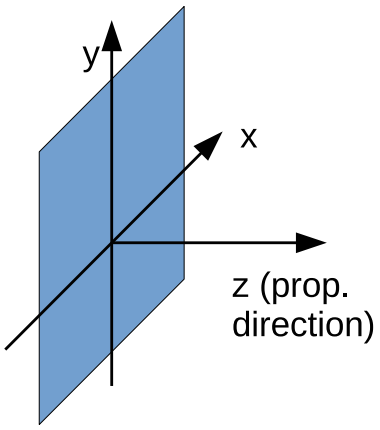


Figure 1: Geometric definitions

# 1 Introduction

Quasi-optical microwave beams are used in several applications, most notably in electron cyclotron heating systems of thermonuclear fusion experiments. Even in systems, where the actual transmission is done with oversized $HE_{11}$ waveguides, there are free space beams before and after the transmission line. At the input, the beam coming out of the gyrogron window is coupled into the waveguide via a set of beam shaping mirrors and possibly polarizers. After the transmission line, a launcher is used to inject the microwave into the plasma. These launchers are also quasioptical components.

In the simplest case, optical transmission systems can be designed using the well established Gaussian optics formalisms. In realistic scenarios, however, the higher-order modes need to be taken into account as well. They can cause high heat loads and/or arcing at possibly unexpected locations or disturb the beams, which are launched into the plasma. Especially with the ever increasing power and pulse lengths of current and future fusion experiments accurate calculation tools are mandatory.

The PROFUSION code package contains a large number of tools, which allow the calculation of different phenomena involving microwave beams. The tools are available as commandline programs for the Linux operating system, while the parameters are passed as commandline arguments. The basic data structure is the *field matrix*, which contains the sampled transversal electric field at one position on the propagation axis. Many of the tools require exactly one field matrix as input and produce another field matrix as output. Those tools can read the field matrix from the standard input and write the resulting field matrix to the standard output. This allows to connect multiple operations using Unix pipes without the need to store intermediate results in files. The implementation as commandline programs follows the old Unix approach (write one program for one task) and has numerous advantages over, e.g. a graphical user interface:

- The tools are independent executables, which are very simple since they do just one operation
- The data transfer via pipes avoids the need of temporary files, and is a well established and robust mechanism provided by the operating system
- Since all parameters can usually be passed as commandline arguments, arbitrary multidimensional parameter sweeps can be performed using the loop constructs provided by the shell
- All calculations are inherently scriptable using the Unix shell
- In addition to the Unix shell, PROFUSION commands can be used in all other enviroments, which allow the execution of external programs (e.g. Python scripts or GUI applications)

PROFUSION has been used in numerous projects and is considered a mature software package.

# 2 Field matrix structure

A field matrix descibes a field on a limited rectangular area perpendicular to the beam axis. Fig. 1 shows the coordinate system. The $z$-axis always corresponds to the main propagation direction, the transversal directions are $x$ and $y$. Typically, if we have a linear polarized field, we set the $x$-Component to zero. This way, the polarization matches the usual defintions of the waveguide modes (TE10 for rectangular, TE11 for circular waveguides) found in literature, where the dominant polarization is vertical.

For the full description of the field in the cross section of a microwave beam, we need the following parameters:

- The $x$ and/or $y$ components of the sampled field as complex matrices

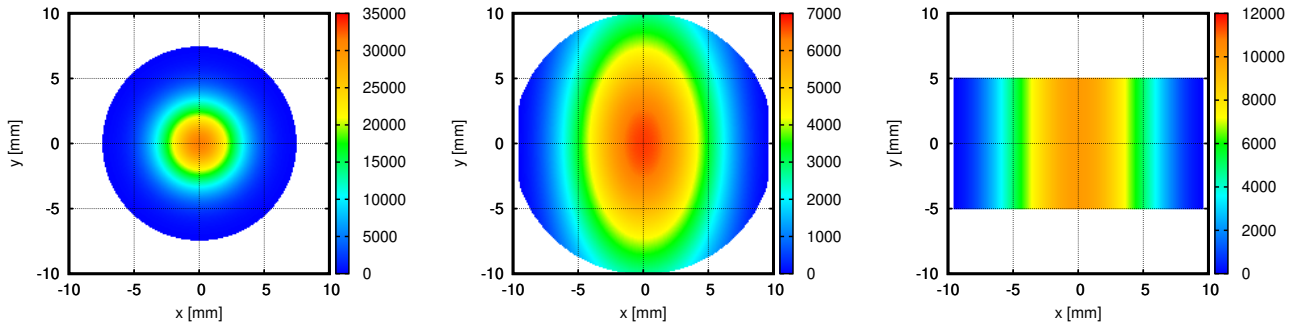


Figure 2: Absolute value of $E_y$ for a Gaussian beam (left), a $TE_{11}$ mode in a circular waveguide (center) and a $TE_{10}$ mode in a rectangular waveguide.

- The horizontal and vertical dimensions of the matrix
- The $x$ and $y$ positions of the boundaries
- The frequency

In the linear polarized case the matrix for the $E_x$ or $E_y$ component is missing rather than filled with zeros. This speeds up most operations for linear polarization. All mentioned parameters are stored in a binary format, which is optimized for fast reading and writing. Internally all quantities are stored in SI-units, in particular the frequency is given in Hertz and lengths are given in Meters. If output files or plots contain different units, they are converted just before they are written. Also, complex numbers are *always* stored as real and imaginary parts. Only the output- and plot routines convert them to amplitude and phase.

# 3 Field generators

Field matrices can be generated from numerical (e.g. measured) data or from analytical functions. In the former case, we just need a preprocessor for converting the data into the field matrix format. For analytical field descriptions, there is the tool `fm_gen`. For generating e.g. a Gaussian beam at 140 GHz with a waist radius of 4.5 mm sampled at $200 \times 200$ points over a cross section of $20 \times 20$mm, one can call

```
fm_gen -xy 200,-0.01,0.01 -freq 140e9 -f gauss,4.5e-3 -o gauss.fm
```

In this case, the output is written to the file `gauss.fm`.

In addition to the simple Gaussian generator, there are numerous other field generators:

- Gaussian beam with general astigmatism
- Mixtures of Gauss-Hermite modes
- Mixtures of Gauss-Laguerre modes
- $J_0$-pattern for generating simplified $HE_{11}$ modes of corrugated cylindrical waveguides
- Radiating waveguide apertures (rectangular, cylindrical smooth, cylindrical corrugated, square corrugated) fed by arbitrary mode mixtures

Some examples for generated fields are shown in Fig. 2. Usually fields are generated with a power of 1 W. A normlized power is important for many analysis methods described in the following sections.

# 4 Propagation

Propagation is the most common operation on field matrices. The tool `fm_prop` uses an FFT transform to decompose the field into a number of plane waves, which are tilted with respect to the beam axis [1]. These plane waves can be propagated trivially by any distance. The propagated field is calculated using an inverse FFT.

One issue with this method is the implicit assumption of the Fourier transform, that the field is infinitely periodic in $x$ and $y$ directions. This means, that the propagation is done for the beam plus an infinite number of virtual neigbor beams. Depending on the extent of the field matrix, the beam size and the beam divergence, we can observe an interference with these virtual neighbor beams in the propagated field.

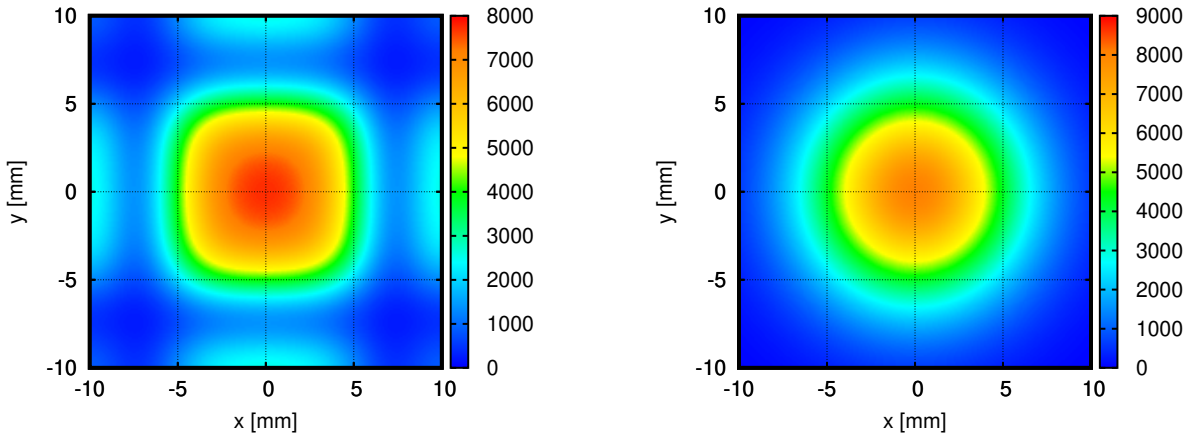


Figure 3: Gaussian beam from Section 3 propagated by 50 mm. Left: Without padding, right: With 50 mm padding.

To suppress these artifacts, we are padding the field matrix with a border containing a zero field before propagating. This border is removed from the propagated field before it is saved, so the size of the initial and propagated field matrices is identical. The border width is adjustable (with the `-pad` option). Unfortunately it is not possible to determine the optimum border width programatically. The artifacts are, however, clearly visible and the border width can be increased experimentally until the artifacts disappear. Especially when doing parameter sweeps, the padding parameter should not be much larger than necessary because it increases the FFT size and thus the calculation time.

Fig. 3 shows the Gaussian beam of Fig 2 (left) propagated by 50 mm. In the left image we clearly see the neighboring beams. In the right image, where a padding of 50 mm was applied, these artifacts are suppressed. As mentioned before, multiple commands can be chained together by pipes. Therefore the fieldmatrix in Fig. 3 (left) can be generated by:

```
  fm_gen -xy 200,-0.01,0.01 -freq 140e9 -f gauss,4.5e-3 | \
fm_prop -d 0.05 -pad 0.05 -o propagated.fm
```

In this case, the output is written to the file `propagated.fm`.

# 5 Focusing elements

Focusing elements are typically realized as metallic mirrors with a concave surface, especially in high power applications. To simulate these in PROFUSION, there is a tool `fm_lens`. It applies a thin lens with given focal lengths $f_x$ and $f_y$ in $x$ and $y$-directions. A phase shift $\Delta\phi(x,y)$ is applied to all field points with:

$$\Delta\phi(x,y) = \frac{2\pi}{\lambda}\left(\frac{x^2}{2f_x} + \frac{y^2}{2f_y}\right) \tag{1}$$

where $x$ and $y$ are the locations on the field matrix. Special commandline options (`-dx`, `-dy`) exist for cases, where the center of the lens is not in the origin of the coordinate system.

Fig. 4 shows the Gaussian beam from Section 3 propagated by 100 mm and sent through a lens with $f_x = f_y = 50\,mm$. As predicted from theory [3], the second focal point comes 86.95 mm after the lens. The beam profiles before and after the lens were plotted with the tool `fm_propprof`, which directly generates PNG files. Special commandline options ensure, that identical color scales are used for both pictures.

# 6 Apertures

Mirrors have limited sizes, which can truncate a part of the Gaussian beam. This results in diffraction effects due to the field discontinuity at the mirror boundary. Furthermore, the power fraction situated outside of the mirror typically contributes to the stray radiation in the system.

In PROFUSION, there is a tool `fm_ap`, which applies a rectangular or elliptical aperture to a field matrix. This is done by simply setting the field values outside the aperture to zero. Fig 5 shows the example from Fig. 4 but with a limited lens diameter of 40 mm. One can see the truncation effect and the diffraction due to the field discontinuity.

To quantify the power loss due to the beam truncation, the naive approach would be to subtract the powers before and after the aperture. In the case of very small losses the result is the difference of two almost equal numbers. This means, that small errors in the powers lead to large errors in the difference. The correct way is to integrate the power, which lies outside the aperture. The tool `fm_truncloss` does the accurate calculation and supports the same aperture shapes as `fm_ap`. An example is given in Section 10.1.

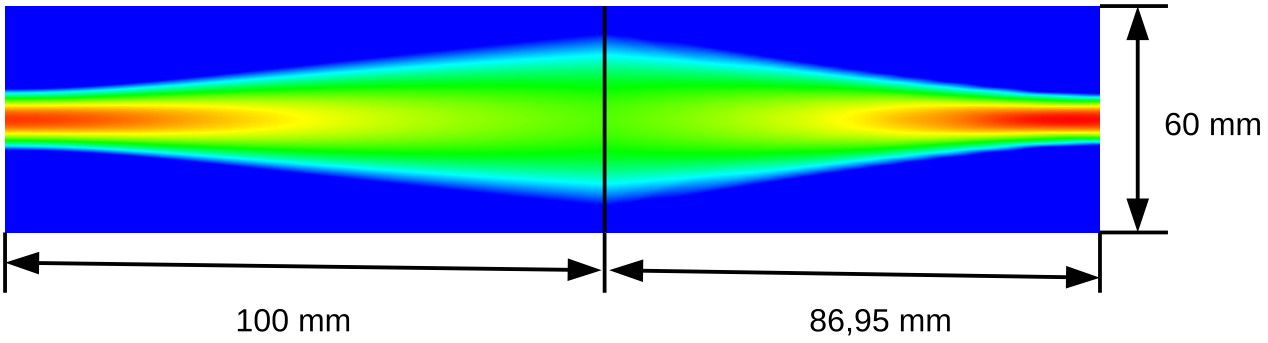


Figure 4: Gaussian beam from Section 3 propagated by 100 mm and sent through a lens with focal length 50 mm. The second focal point comes 86.95 mm after the lens

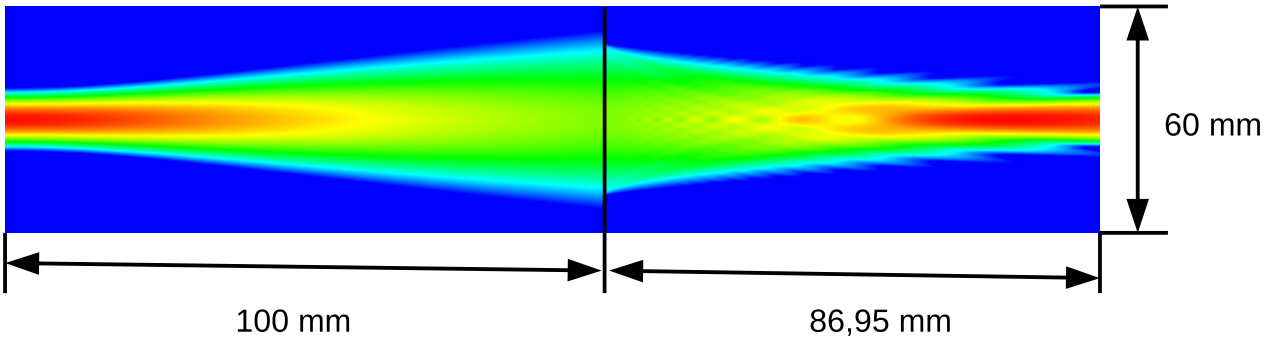


Figure 5: Gaussian beam from Fig. 4 but with a limited lens size of 40 mm

# 7 Diagnostic tools

For obtaining relevant parameters from the calculated fields, a large number of diagnostic tools is available.

## 7.1 Power calculation

One of the most important diagnostic tool is the integration of the total power of a field distribution. Details of the calculation, however, are also applicable to other diagnostic methods outlined below.

The simplest method is to calculate the sum of the power densities over all pixels. A more precise method, however, is to use a higher-order interpolation scheme to emulate a smooth function, and an integration method with adaptive stepsize and error estimation. The most accurate results are obtained with a Catmull-Rom interpolation. The integration is a 2D Simpson method as suggested in [2]. Another important detail is that the interpolation is applied to the *power density* instead of the field strength itself. Interpolating the E-field itself can distort the result because of a slight violation of the energy conservation. Table 7.1 shows the power of the Gaussian beam from Fig. 2 (left) for different sample densities and for least and most accurate algorithms. Note that even for $20 \times 20$ samples the interpolation method is sufficient in calculating the power.

For noisy (i.e. measured) field data, however, the sum method is advantageous because single noise peaks in conjunction with interpolation distort the results and have a poor convergence in the integration routine. In the case of measured data, the error bars are typically dominated by the equipment rather than the analsis algorithm. In PROFUSION the tool `fm_getpow` can calculate the total power with various algorithms and `fm_norm` can normalize a field matrix to unity power.

## 7.2 Overlap integral

Overlap integrals are used to characterize the purity of a measured or calculated field with respect to some reference field distribution. Physically, the overlap integral can be interpreted as the complex power integral, where the E-component is taken from one field distribution, the H-field is taken from the other field:

$$C_{12} = \frac{\iint_S \mathbf{E}_1 \times \mathbf{H}_2^* dS}{\sqrt{\left|\iint_S \mathbf{E}_1 \times \mathbf{H}_1^* dS\right|}\sqrt{\left|\iint_S \mathbf{E}_2 \times \mathbf{H}_2^* dS\right|}} \tag{2}$$

| Linear size | Sum | Interpolation |
|---|---|---|
| 20 | 0.90250 | 0.99999 |
| 40 | 0.95062 | 0.99998 |
| 60 | 0.96693 | 0.99999 |
| 80 | 0.97514 | 0.99997 |
| 100 | 0.98009 | 0.99996 |
| 120 | 0.98339 | 0.99998 |
| 140 | 0.98575 | 0.99998 |
| 160 | 0.98752 | 0.99998 |
| 180 | 0.98890 | 0.99998 |
| 200 | 0.99001 | 0.99998 |

Table 1: Results of the power integration for a Gaussian beam (See Fig. 2 left) using the sum and Catmull-Rom interpolation.

where the terms in the denominator ensure that both field components have a unity power. Although the calculation corresponds to a power calculation, the result is equivalent to a complex amplitude. The phase angle can be interpreted as an overall phase shift between the two fields and is not relevant in most cases. The purity, which is a power quantity, is therefore the squared absolute value ($|C_{12}|^2$) of the overlap integral. Overlap integrals are calculated with `fm_overlap`, which has similar options as `fm_getpow`.

Note that the power normalization (as in the demoninator of Eq. 2) is *not* done by `fm_overlap`. If necessary, the input matrices need to be normalized with `fm_norm` before.

## 7.3 Fitting of Gaussian beam parameters

The overlap integral as described in section 7.2 is a good measure for a beam purity in cases where a fixed set of beam parameters is defined as reference. This is, however, the most pessimistic characherizaiton, because no distinction is made between a variation of the beam parameters and the generation of higher-order modes. This is due to the fact, that in Gaussian Optics, each set of beam parameters (waist sizes and -locations) defines an individual orthogonal system of modes.

When designing optical transmission systems, however, the parameters of the fundamental $\mathrm{TEM}_{00}$ mode as well as the higher-order modes are of interest and need to be available as separate quantities. The beam parameters can be used to design mirror surfaces. The higher order modes should be supressed as much as possible because - due to their larger divergence angles - they are likely to contribute to the overall stray radiation, especially for long transmission lines.

The established method of characterizing (e.g. measured) Gaussian-like beams consists of a fit procedure, which determines the beam parameters of in order to maximize the overlap integral (according to Eq. 2) for the fundamental $\mathrm{TEM}_{00}$ mode.

The fractional power $P_{HOM}$ of higher-order modes then becomes:

$$P_{HOM} = 1 - |C_{\mathrm{TEM}_{00}}|^2 \tag{3}$$

where $C_{\mathrm{TEM}_{00}}$ is calculated from (2) with the fitted parameters. In PROFUSION, there is a tool `fm_fit`, which performs the described procedure. The initial values are obtained from the field by statistical evaluation. When run in full mode, the follwing paramers are obtained:

- The complex beam parameter in horizontal and vertical directions
- The position $(x,y)$ of the beam axis in the field matrix
- Tilt angles in $x-$ and $y-$directions

Furthermore there are commandline options, which limit the number of fit parameters. When a field distribution is e.g. known to be centered with a beam axis perpendicular to the field matrix (e.g. because it is synthetically generated), the position and tilt angles can be ommitted for the fit (`-nopos`, `-notilt`). Another option (`-circ`) can be used for field patterns, which are known to be rotationally symmetric. In this case, the beam parameters in horizontal and vertical directions are assumed to be identical. Omitting fit parameters reduces the dimensions of the parameter space and improves convergence.

As an example, Fig. 6 shows the $\mathrm{TE}_{11}$-field of an open-ended circular waveguide from Fig. 2. The fit procedure obtained an elliptical Gaussian beam which contains 88.86 % of the total power (center). The right image shows the residual field, which can be interpreted as the sum of spurious modes.

Calculating the residual field allows to investigate the behavior (e.g. divergence angles, dominant directions) of the spurious modes.

From equation 3 we can see, that it is essential, that the field passed to `fm_fit` has a unity power, especially if the value for the higher-order modes is important. This needs to be ensured before by calling `fm_norm`.

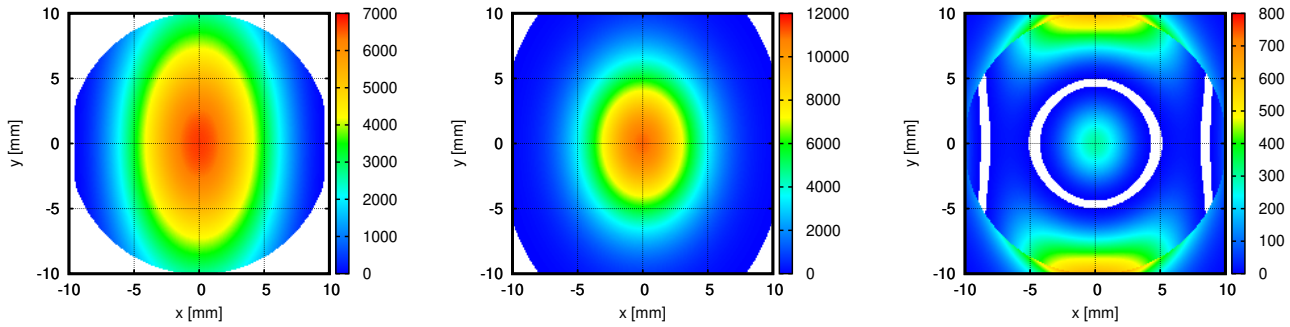


Figure 6: Absolute value of $E_y$ for a $TE_{11}$ field in a cylindrical waveguide aperture (left), the fitted Gaussian beam (center) and the residual field (right).

# 8 Further utilities

A large number of simple but useful tools for operating on field matrices has been developed over the years. They all support the commandline option `-help` to show the supported options.

- `fm_add`, `fm_diff` Adds/subtracts two field matrices
- `fm_xpol` Calculates the power in both polarizations
- `fm_peak` Gets the maximum field value
- `fm_tilt` Compensates small phase errors due to misalignments from measured data
- `fm_profile` Calculates the horizontal and vertical profiles from a field matrix
- `fm_dump` Prints information about the fieldmatrix
- `fm_cat` Converts a fieldmatrix to ASCII format
- `fm_extract` Extract single field values at given coordinates
- `fm_antennadiagram` Calculate an antenna diagram from an aperture field
- `fm_norm` Normalize the field matrix for unity power or other criteria
- `fm_plot` Plot the field components in gnuplot or png formats
- `fm_tweak` Change some internal parameters (e.g. frequency) of a field matrix

# 9 Interface with waveguides

In transmission systems for megawatt power at millimeter wave frequencies, oversized $HE_{11}$-waveguides are typically used. Since the wave impedance in these waveguide is very similar to the free space impedance, transitions are very simple because no impedance matching is necessary. At the input of a transmission line, the beam from the Gyrotron is fed via beam shaping mirrors into the waveguide. The oversized waveguides are, however, sensitive to misalignments of the coupling optics.

The calculation of the mode spectra, which are excited by the free space field at the waveguide entrance, is done with eq. 2, where one field is the free space field, the other is the transversal field of the waveguide mode. PROFUSION has tools for doing mode analyses for all supported waveguide types: `cyl_analysis` and `rect_analysis` are for smooth wall waveguides with circular or rectangular cross section. The commands `cyl_corr_analysis` and `sqc_analysis` are for corrugated waveguides with circular and square cross-section respectively. The modes, which should be taken into account, are given in the waveguide description (see section A.1). The field distribution is the same as for the `fm_gen` command (see section 3).

The transformation of a spectrum of waveguide modes to a free-space field is done by summing the modal fields (multiplied with the amplidudes). Here we also assume that the size of the waveguide is large enough such that the impedance mismatch can be neglected. An aperture field can be generated by using the the field generators `cyl`, `corr`, `rect` and `sqc` with the `fm_gen` command.

# 10 Examples

## 10.1 Losses of a waveguide gap

This example calculates the losses due to a gap in an oversized corrugated waveguide carrying an $HE_{11}$-Mode. The $HE_{11}$-Mode is known to have a small divergence when radiating from an open-ended waveguide, provided that the waveguide diameter is much larger than the wavelength, which is usually fulfilled in $HE_{11}$-Waveguides.

Gaps can be used as simple and efficient filters for higher-order modes or as a DC-Break to isolate the waveguide electrially. For the calculation of the losses, we distinguish 2 quantities: The first is the power, which is actually lost because it will not be inside the waveguide aperture due to the divergence of the free-space beam in the gap. The second part will be converted into some higher-order modes, like the $HE_{12}$ or $HE_{13}$, which continue to propagate in the waveguide as spurious modes.

The following skript does the whole calculation:

```
#!/bin/sh
# Force decimal dot even for non-English systems
export LC_NUMERIC=C

# Generate HE11 field
fm_gen -f corr,corr.wg,,31.75e-3 -xy 1000,-0.08,0.08 \
  -freq 170e9 -o gap_start.fm

# Loop over gap widths
for i in `seq 0.001 0.001 0.500`; do

  # Propagate (gap_start.fm -> gap_end.fm)
  fm_prop -i gap_start.fm -d $i -pad 0.1 -o gap_end.fm

  # Calculate the truncation loss
  TRUNCLOSS=`fm_truncloss -s 63.5e-3 -i gap_end.fm | \
    grep -v '^#' | awk '{ print $3 }'`

  # Calculate the excited HE11 mode after the gap
  cyl_corr_analysis -wg corr.wg -r 31.75e-3 -freq 170e9 \
    -o ampl.dat -f fm,gap_end.fm

  # Format the amplitude for Writing to the file
  HE11AMPL=`dump_ampl -i ampl.dat | \
    grep -v '^#' | \
    head -n 1 | \
    awk '{ print 1.0-($1/100) }'`

  # Write output
  echo "$i $TRUNCLOSS $HE11AMPL"

# End of the loop
done
```

The waveguide diameter is 63.5 mm, the frequeny is 170 GHz. The field in the waveguide cross-section is generated with the `fm_gen` command. We choose an area of $160 \times 160$ mm$^2$, which should be large enough to also contain the power, which leaves the waveguide through the gap. This is necessary because to obtain a high accuracy, the truncation losses are calculated by integrating the field *outside* of the aperture.

The loop over the gap is done using the `seq`-command, which is available on Linux systems and generates a sequence of equidistant numbers. Within the loop, the field is first propagated with `fm_prop`. Then, the truncation losses are calculated by integrating the power, which lies outside of the waveguide cross section after the propagation.

Finally, we call `cyl_corr_analysis`, which expands the free space field at the end of the gap into waveguide modes. We are only interested in the $HE_{11}$-Model, but the amplitudes of the higher-order modes could be calculated as well by adding them to the waveguide definition file.

The losses have been discussed in literature ([4], [5]), where analytical formulas were derived for typical waveguide modes. In the case of the $HE_{11}$-mode, the loss can be approximated with:

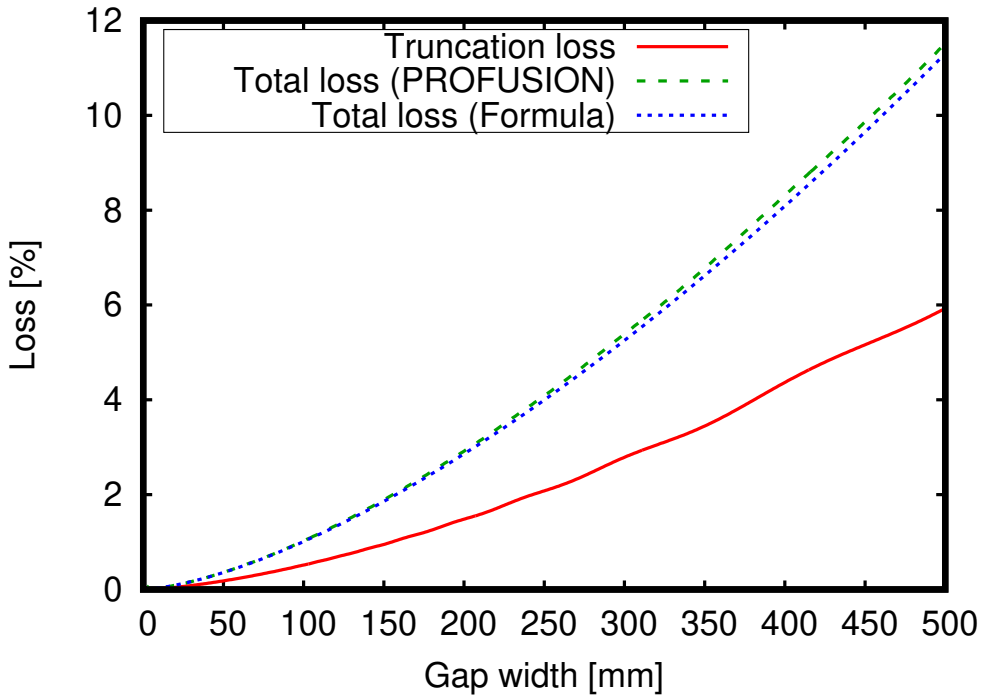


Figure 7: Losses of a gap in am $HE_{11}$-Waveguide as a function of the gap width

$$1 - |A_{\mathrm{HE}_{11}}|^2 = \frac{2}{\sqrt{\pi}} \frac{X_{\mathrm{HE}_{11}}^2}{3} \left( \frac{L}{k_0 a^2} \right)^{\frac{3}{2}} \tag{4}$$

Here, $X_{\mathrm{HE}_{11}}$ is the Eigenvalue (2.405), $a$ is the waveguide radius, $k_0$ is the free space wavenumber and $L$ is the width of the gap. Figure 7 shows the results. We can see, that the numerically calculated total loss agrees very well with the prediction from the formula.

Note that the power is not re-normalized before coupling into the waveguide. This means all losses are relative to the total power of the $HE_{11}$-mode *before* the gap.

# 11 Conclusions

The PROFUSION package contains a versatile set of tools for the calculation of free space microwave beams. The realization, which consists of a number of commandline programs, is easy to learn for users with experience on the Unix commandline. The strengths in comparison with GUI tools are the versatility because the programs can be combined in any way, and the inherent scritability. PROFUSION is used in many ongoing projects and the code is under permanent maintenance. Also, new features are permanently added.

Numerous comparisons were made to compare the results with measurements, analytical formulas and other code packages, and the agreement was always very good.

This work is supported by the Max Planck Institute for Plasma Physics, Garching and Greifswald/Germany.

# A File formats

Configuration files are in a simple ASCII format, which can be edited with any text editor. All file formats have a signature as the first line, which specifies the files type. It prevents accidental usage errors and confusions. Most formats contain a simple set of name/variable pairs. Sectioning or other hierarchical structures are not supported. Below is an example for a hypothetical format:

```
SIGNATURE
Var1: val1
Var2: 1.0
Var3: 1
Var4: 100,0.2,0.5
```

The variable names are followed by a colon, after the colon we can have an arbitrary amount of whitespace. The first non-space character up to the end of the line is the value. The hash-character ('#') starts a comment, which extends to the end of the line.

## A.1 Waveguide descriptions

Waveguide description follow the format described above, except that the "Mode" variable can be used multiple times to add multiple waveguide modes. The first mode is considered the "main mode". For commands, which need a spectrum of mode amplitudes in a separate file (see section A.2), the default is to assume 100% of the power and zero phase for the main mode and zero power for all others.

### A.1.1 Circular waveguide (smooth wall)

An example for a waveguide configuration is shown below. The file signature is "`WAVEGUIDE_CYL`". The supported variables are:

- `Radius`: Waveguide radius in meters. Can be overridden at the commandline for many programs
- `Material`: Wall material defined by 3 numbers separated by spaces: The specific resistance (in Ωm), the relative magnetic permeability $\mu_r$ and a factor $R_c$ which takes the surface roughness into account. The default values (`0.0 1.0 1.0`) are for a PEC waveguide wall.
- `Frequency`: 3 Values (number of points, minimum, maximum). Alternatively, for single frequency setups, the Frequency can be given as a single number

Finally, a number of modes can be given with the "Mode" keyword. It contains a string specifying the mode type, followed by two indices for corresponding the transversal coordinates.

```
WAVEGUIDE_CYL
# Radius (in m)
Radius: 19.0e-03
# Material: rho mu_r R_c
Material: 0.000000000e+00 1.000000000e+00 1.000000000e+00
# Frequency: num_points min max
Frequency: 5 70.0e+9 110.0e9
Mode: TE 1 1
Mode: TE 1 2
Mode: TE 1 3
Mode: TM 1 1
Mode: TM 1 2
Mode: TM 1 3
```

### A.1.2 Rectangular waveguide (smooth wall)

The format for rectangular waveguide is identical to the cylindrical waveguide with the exception, that the file signature is "`WAVEGUIDE_RECT`" and the "`Radius`" variable is replaced by "`Width`" and "`Height`", which specify the inner dimensions of the waveguide.

Furthermore, the variable "`Pairs`" specifies, that $TE_{mn}$ and $TM_{mn}$ ($m$, $m > 0$) modes are combined into pairs, which are linearly polarized. It is a boolean value, which can be 0 or 1. These mode pairs can be specified by either TM or TE. One application for mode pairs are E-plane bends, which are fed by a $TE_{10}$-Mode.

### A.1.3 Circular waveguide (corrugated wall)

The format for rectangular waveguide is identical to the cylindrical waveguide except that the file signature is "`WAVEGUIDE_CYLCORR`". An additional variable "`WP`" specifies the ratio of the groove width and the groove peroid of the corrugation. It can in most cases be set to 0.5.

### A.1.4 Square waveguide (corrugated wall)

The square corrugated waveguide module handles only the simple case of linearly polarized balanced $HE_{mn}$-Modes (corrugation depth: $\lambda/4$) in the square waveguide. Therefore, the dimensions are given just by "`Width`". The file signature is "`WAVEGUIDE_SQC`".

## A.2 Mode amplitudes

Mode amplitudes are saved in separate files. It has the signature "AMPLITUDES", followed by a single variable "Num", which specifies the number of mode amplitudes to follow. The remaining lines contain amplitudes in the complex format (real and imaginary part, separated by a space). Mode amplitudes are always defined such that an amplitude of 1 corresponds to unity power. For most applications, the power sum of all modes should be 1.

You can generate a PROFUSION compliant amplitude file from a simpler format, which contains no header and the mode powers (in percent) and the phase (in degrees) with the `make_ampl` command. For the example of a 95%/5% mode mixture, with a phase shift of 90°, one can generate a file:

```
95.0 0.0
5.0 90.0
```

and with `make_ampl` the resulting amplitude file will be:

```
AMPLITUDES
Num: 2
9.746794345e-01 0.000000000e+00
1.369196746e-17 2.236067977e-01
```

The real part of result for the second mode is nonzero due to some rounding errors of the square root and cosine and sine functions. To display the contents of the mode amplitudes, use the tool `dump_ampl`

# References


[1] Goodman, J.W.: Introduction to Fourier Optics. Physical and Quantum Electronics Series. McGraw-Hill (1968)

[2] Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C, 2 edn. Cambridge University Press (1992)

[3] Self, S.A.: Focusing of spherical gaussian beams. Appl. Opt. **22**(5), 658–661 (1983). DOI 10.1364/AO.22.000658. URL `http://ao.osa.org/abstract.cfm?URI=ao-22-5-658`

[4] Vaganov, R.B.: Diffraction of asymmetric waves by a wide slit in a circular waveguide. Radiophysics and Quantum Electronics **12**, 504–506 (1969). DOI 10.1007/BF01037026

[5] Wagner, D., Thumm, M., Kasparek, W., Müller, G.A., Braz, O.: Prediction of TE-, TM-, and hybridmode transmission losses in gaps of oversized waveguides using a scattering matrix code. International Journal of Infrared and Millimeter Waves **17**(6), 1071–1081 (1996). DOI 10.1007/BF02101439. URL `https://doi.org/10.1007/BF02101439`