Optical Simulation of Grating Diffraction in MATLAB®

by Ken Johnson

The “Grating Diffraction Calculator” (GD-Calc®) is a MATLAB-based, electromagnetic simulation program that computes diffraction efficiencies of optical grating structures, including biperiodic gratings. The program’s capabilities include a general and flexible grating modeling facility, structure parameterization (with any number of parameters), and unrestricted control over diffraction order selection. Additionally, its implementation within the generic programming and application development framework of MATLAB provides a degree of flexibility and software interoperability that is not available with stand-alone diffraction analysis programs.

Part 1 of this article provides a conceptual overview of GD-Calc, describing in general terms how grating structure is specified and how electromagnetic computations are carried out. The presentation is primarily concept-oriented, but a few simple code examples are provided to give the reader a sense of how the GD-Calc software interface works. Part 2 provides a more in-depth introduction to the software interface, using as an example a tungsten photonic crystal structure[1] to illustrate how grating structure is specified. (The code listings from Part 2 are summarized in gdc_intro.m.)

The primary focus of this article is grating structure specification. Application examples for electromagnetic computations are provided in an accompanying document, GD-Calc_Demo.pdf. (All of the code examples in this article and in GD-Calc_Demo.pdf can be run with the free demo/tutorial code from the GD-Calc website. The photonic crystal example is based on the demo script gdc_demo11.m.) The electromagnetic theory and algorithms underlying GD-Calc are detailed in GD-Calc.pdf.

Part 1: Conceptual Overview

MATLAB development environment

An advantage of working in the MATLAB environment is that functional links into and out of GD-Calc can be created without having to rely on cumbersome data conversion and import/export procedures. For example, in a semiconductor lithography application, a photoresist grating’s thickness and refractive index might both be affected by exposure-related resist densification, so it would be natural to specify thickness and refractive index both as user-defined functions of exposure. This is especially useful with

[1] The photonic crystal structure is described in S. Y. Lin, J. G. Fleming, and I. El-Kady, “Highly efficient light emission at by a three-dimensional tungsten photonic crystal,” Optics Letters 28(18), 1683-1685 (2003).

structure parameterization, e.g., exposure could be defined as a vectorized quantity, in which case all exposure-dependent quantities, including the resist thickness, refractive index, and calculated diffraction efficiencies, will be similarly vectorized.

Typically, a grating's optical characteristics are not themselves of primary interest; what is of interest is the optical response of a complete system that includes the grating as one component. MATLAB's generic programming capability makes it easy to functionally link GD-Calc into user-defined optical system models, which can themselves be incorporated into generic optimization routines to optimize design performance. GD-Calc is simply a MATLAB function (gdc.m), which can be incorporated into other MATLAB functions or scripts, and which takes arguments that can be instantiated to user-defined functions.

Although stand-alone programs lack the generality and flexibility of the MATLAB development environment, they can have the advantage of simplicity and ease-of-use. However, GD-Calc can be used in conjunction with MATLAB to create customized user interfaces that are optimally adapted for specific applications. Many of the functions and scripts associated with GD-Calc such as its plotting facility (gdc_plot.m), its output data conversion function (gdc_eff.m), and a number of demo scripts are distributed as public-domain software so that users can freely modify and adapt the code to best suit their own or their customers' needs.

Structure specification

The primary limitation of GD-Calc's grating modeling capability is that gratings must be "block-structured" (or must be defined approximately in terms of a block-structured representation), meaning that the grating comprises optically homogeneous regions whose bounding surfaces are planes parallel to a set of primary coordinate planes. For example, a grating comprising a periodic array of pyramids would be represented using a "staircase approximation", as illustrated in Figure 1.

The grating is subdivided into a number of "strata", wherein each stratum is bounded by upper and lower planes parallel to the grating substrate. The grating has a height-independent lateral cross-section within each stratum. Figure 2 illustrates a particular stratum extracted from the pyramidal grating of Figure 1. Each stratum is partitioned into parallel "stripes", which are further partitioned into rectangular "blocks" representing optically homogeneous regions.

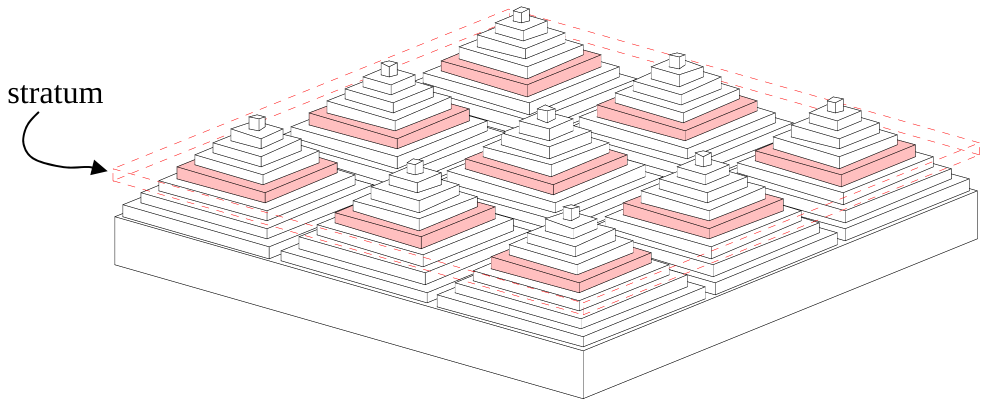

Figure 1 Pyramidal grating

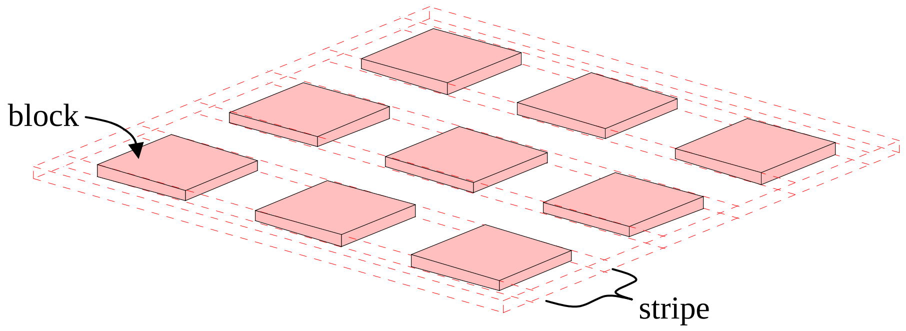

Figure 2 Grating stratum

In terms of software representation, a structure such as that of Figure 1 would be represented as a nested hierarchy of cell arrays within structs. The top-level data structure, designated here as `grating`, is a struct containing a cell-array `stratum` field whose elements, `grating.stratum{1}`, `grating.stratum{2}`, … represent the grating strata (numbered from the bottom up). The *i*-th stratum's stripes are represented as `grating.stratum{i}.stripe{1}`, `grating.stratum{i}.stripe{2}`, …, and the *j*-th such stripe's blocks are represented as `grating.stratum{i}.stripe{j}.block{1}`, `grating.stratum{i}.stripe{j}.block{2}`, … . Other fields within these structs define the bounding plane locations and the optical material within each block. The optical materials are described in terms of their complex permittivities, which are enumerated in a top-level cell array `grating.pmt`, each element of which represents a particular material. A particular grating region's optical material is specified as an integer index into this list (e.g. `grating.stratum{i}.stripe{j}.block{k}.pmt_index`); thus multiple regions can be constrained to represent the same material by giving them the same material index.

There are five different types of "stratum" objects that can be used to define the grating. These are enumerated below:

Stratum type	Description
homogeneous	homogeneous grating layer with no stripe boundaries
uniperiodic	stratum with homogeneous stripes, no block boundaries
biperiodic	general biperiodic stratum (as illustrated in Figure 2)
coordinate break	applies a translational shift to all strata above break plane
replication module	for defining 3-dimensionally periodic grating regions

Each stratum object has a "`type`" index indicating its type. For example, a homogeneous stratum is defined by three struct fields: the type index (zero), the stratum thickness, and the permittivity index, e.g.,

```
stratum.type = 0;  % homogeneous
stratum.thick = 0.5;
stratum.pmt_index=1;  % index into grating.pmt
grating.stratum{1} = stratum;
```

The homogeneous and uniperiodic stratum types are basically specializations of the more general biperiodic type. A "coordinate break" is a "stratum" in the abstract sense that it is associated with a lateral plane at a particular height in the grating, and it provides a simple mechanism for applying a lateral translational shift to all strata above the break plane without having to modify the individual stratum definitions. A "replication module" is a composite type of stratum object used to represent a structure pattern that repeats itself periodically in a direction transverse to the grating substrate. (The basic structure pattern is represented as a stack of strata, which can be of any type – including other replication modules.)

Figures 3 and 4 illustrate conceptually how the above structuring elements could be combined to define a photonic crystal structure. First, the three-stratum configuration illustrated in Figure 3 is defined. The first two strata are uniperiodic with orthogonal stripe orientations, and the third stratum is a coordinate break, which applies a half-period translational shift in each of the two periodicity directions. (The translational shift is represented by the arrow in Figure 3.) Next, the three strata are combined into a replication module object, which has an associated replication count indicating how many times the pattern is to be repeated. Figure 4 illustrates the resulting structure with a replication count of four. The structure comprises four stacked copies of the basic bilayer pattern, with each copy laterally shifted by a half period in both periodicity directions relative to the underlying bilayer.

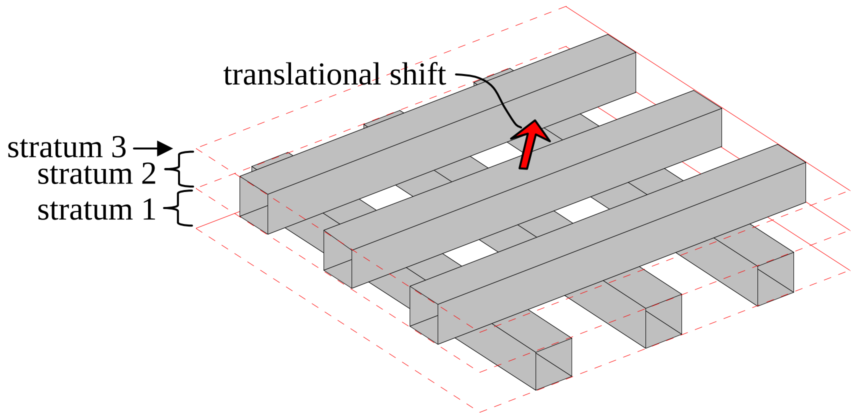

Figure 3 Bilayer pattern (with coordinate break) for photonic crystal

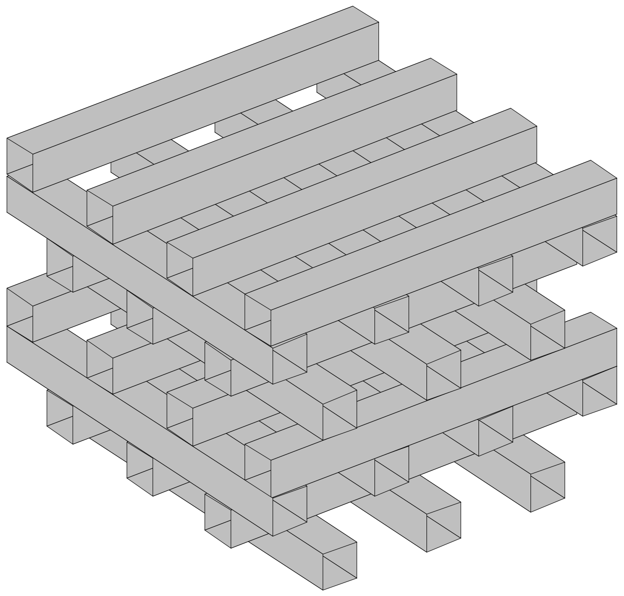

Figure 4 Photonic crystal

Rather than using a replication module in the above example, the grating could be defined by simply stacking four copies of the Figure 3 structure. However, the advantage of using a replication module is not just one of convenience. Whereas the GD-Calc computation time generally scales in proportion to the number of grating strata, the computation time for a replication module scales in proportion to the logarithm of the replication count.

Parameterization

The GD-Calc interface specification (defined in the gdc.m comment header) identifies a number of grating attributes as “parameters”. A “parameter”, in this context, is a numeric quantity that can be vectorized as a multidimensional array. A parameter can be one of a number of basis parameters, each of which is associated with a particular array dimension (and which has only one non-singleton dimension), or it can be a function of other parameters. (A parameter’s non-singleton dimensions indicate its functional dependencies.) The only fundamental restriction on parameters is that they must all be size-matched, except that singleton dimensions are implicitly repmat-extended, if necessary, to match parameter sizes.

A simple application example showing the use of parameterization is illustrated in Figure 5.[2] The figure shows a cross-section of an alignment-sensor grating comprising a substrate, a uniperiodic, surface-relief reflective grating (stratum 1), a homogeneous air space (stratum 2), a coordinate break (stratum 3), and a uniperiodic phase grating (stratum 4) on a transparent superstrate. A small, lateral translational shift between the two gratings will result in a measurable shift in the energy balance between the first- and minus-first-order diffraction efficiencies; thus the gratings can function as a sensitive positional transducer.

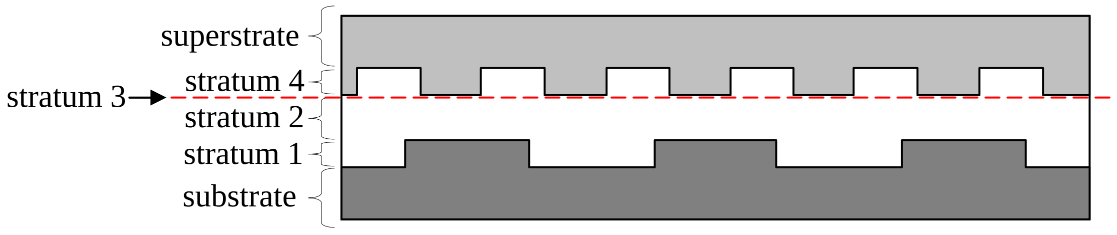

Figure 5 Alignment-sensor grating

In this example, two grating parameters are vectorized: the air space thickness, represented as `grating.stratum{2}.thick`, and the phase plate’s translational shift, represented as `grating.stratum{3}.dx2`. These could be defined, for example, as

```
grating.stratum{2}.thick = wavelength*[3;2;1];
grating.stratum{3}.dx2 = d*(0:63)/64;
```

(The `wavelength` and `d` variables are scalars representing the illumination wavelength and grating period, respectively.) The air space is size-[3,1], and is vectorized in dimension 1, while the translational shift is size-[1,64], and is vectorized in dimension 2. Any quantities that are functionally dependent on both the air space and the translational

[2] This example is based on the GD-Calc demo script, gdc_demo9.m.

shift, including computed diffraction efficiency quantities, will be size-[3,64]. Based on the repmat-extension convention described above, these arrays are all size-compatible.

Parameterization can, in some instances, dramatically improve GD-Calc's computational performance. As explained in the next section, the GD-Calc algorithms are based primarily on two operations: first, calculating an "S matrix" for each stratum, and then combining the S matrices from bottom to top by means of a "stacking" operation to determine a composite S matrix for the entire grating. In the above example, there are 192 parameter combinations (3 air space thicknesses and 64 translational displacements), so if the parameters were iterated outside of GD-Calc all of the S-matrix computations would have to be repeated 192 times. But with parameterization, the S matrices for the grating layers (strata 1 and 4) will only have to be calculated once, the air space's S matrix is calculated only three times, and the coordinate break's S matrix is calculated only 64 times. Furthermore, the stacking operation for stratum 2 will be calculated just 3 times, and only the stratum 3 and 4 stacking will have to be done for all 192 parameter combinations.

Electromagnetic theory

GD-Calc's algorithmic foundation is based on a generalized variant of rigorous coupled-wave (RCW) theory, which first came into use about three decades ago but has since undergone a couple of major improvements. In essence, the method represents both the electromagnetic field and the optical permittivity at any particular height in the grating in terms of their Fourier series in two lateral grating coordinates, and a set of differential equations are developed that describe the propagation of the field's Fourier coefficients through the grating.

In the original formulation of RCW theory, the propagation equations were numerically solved to determine a "transmission matrix" that defined the field amplitudes (Fourier coefficients) at the top of each stratum as a linear function of the amplitudes at the bottom, and these matrices were then multiplied to determine a composite transmission matrix for the entire grating. But numerical contamination from exponentially growing errors caused this method to be numerically unstable, especially with very deep or highly conducting gratings. The problem has been resolved by using an alternative "S-matrix" (scattering matrix) approach, which represents each stratum in terms of a linear mapping relating the outgoing field amplitudes to the incoming field amplitudes at both boundaries. The individual mappings (S matrices) for the strata are combined by means of a numerically stable "stacking" algorithm to determine the grating's composite S matrix.

The S-matrix method resolved the numerical stability problem, but convergence with respect to the number of Fourier coefficients retained in the calculations was typically very slow. The main problem had to do with terms in the differential equations representing the product of the electromagnetic field and the permittivity. Each product's Fourier series was determined from its factors' truncated Fourier series, but severe numerical convergence problems arose when the factors had concurrent discontinuities associated with optical boundaries. This problem has been resolved by using a “Fast Fourier Factorization” method that, in essence, rearranges the equations to avoid product factors with concurrent discontinuities (i.e., in each equation where a concurrent discontinuity occurs, the permittivity factor is brought to the other side of the equation before applying the Fourier decomposition).

GD-Calc uses both the S-matrix method and Fast Fourier Factorization to optimize computational performance. (The numerical algorithms are detailed in GD-Calc.pdf.) One problem that remains with RCW methods, however, is the “staircase approximation” that must be used to describe sloped or curved surfaces in terms of the “block-structured” representation. Simply partitioning the grating into very small blocks does not ensure good accuracy, because the electromagnetic field can exhibit large spikes near the block corners, and the number of retained diffracted orders must be increased in proportion to the block partitioning density in order to adequately resolve the spikes. Thus, users of GD-Calc should be aware of convergence difficulties that can arise, particularly with highly-conducting gratings that are not inherently block-structured. (The GD-Calc demo scripts and GD-Calc_Demo.pdf provide examples of the program’s convergence behavior for a variety of test cases, including comparisons with published data.)

Diffraction order selection

GD-Calc gives the user complete freedom in selecting which diffracted orders (i.e., Fourier coefficients) to retain in calculations. Generally, computational data storage requirements scale in approximate proportion to the square, and runtime scales in proportion to the cube, of the number of retained orders, so optimizing the order selection can very significantly impact computational performance.

The checkerboard grating illustrated in Figure 6 (in plan view) is one example where order selection is particularly useful. The grating is described in relation to two fundamental period vectors; for example vectors $\vec{A}$ and $\vec{B}$ would be a natural choice. However, the grating has a periodic symmetry stronger than that described by vectors $\vec{A}$ and $\vec{B}$ because, for example, the fundamental period vectors $\vec{A}$ and $\vec{C}$ define a unit cell whose area is half that of $\vec{A}$ and $\vec{B}$. Thus, if the grating’s optical permittivity is Fourier analyzed with respect to orthogonal coordinates represented by vectors $\vec{A}$ and $\vec{B}$, half of its Fourier orders will be identically zero, and similarly half of the electromagnetic field’s diffracted orders will be zero.

Each grating Fourier coefficient, and correspondingly each electromagnetic diffraction order, has two associated Fourier order indices m_1 and m_2; and the user must specify what set of (m_1, m_2) index pairs to retain in the electromagnetic field expansion. In the Figure 6 example, using basis periods $\vec{A}$ and $\vec{B}$, all orders with $m_1 - m_2$ odd would be identically zero, so only (m_1, m_2) index pairs with $m_1 - m_2$ even need be retained. The elimination of extraneous orders would reduce data storage requirements by about a factor of 4, and computation time would improve by about a factor of 8.

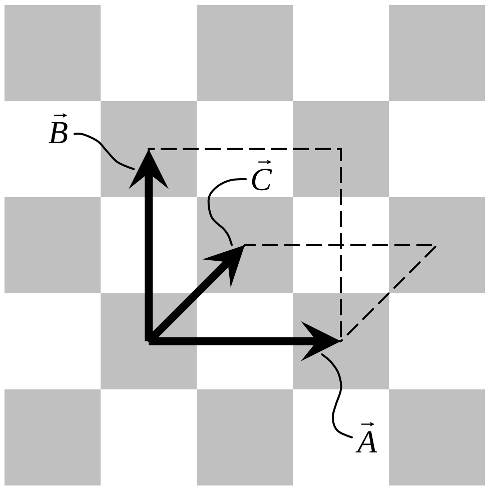

Figure 6 Checkerboard grating

The photonic crystal illustrated in Figure 4 is another example for which order selection can be used effectively. Rather than using the standard "rectangular order truncation" defined by the conditions $|m_1| \le m\_max$ and $|m_2| \le m\_max$ for some truncation limit m_max, an alternative "diagonal truncation" method, defined by the conditions $|m_1| + |m_2| \le m\_max$, may be employed. The diffraction calculations' convergence performance with respect to m_max would be similar with both methods, but diagonal truncation would be much more computationally efficient.

Part 2: GD-Calc Software Interface

GD-Calc usage overview

The first step in using GD-Calc is to construct a "`grating`" data structure defining the grating geometry and optical materials. (Most of this tutorial will focus on this step.) You can then run a data validation check as follows,

```
gdc(grating);
```
(1)

Also, you can plot a 3-D view of the grating,

```
gdc_plot(grating,param_index,pmt_display,x_limit);
```
(2)

`param_index` is a multi-dimensional parameter index, which relates to GD-Calc's parameterization capabilities. (In the following examples `param_index` is just set to 1.) `pmt_display` specifies display colors and legend strings for the grating's optical materials, and `x_limit` specifies 3-D plotting limits. `gdc_plot` internally invokes `gdc(grating)` to check data validity.

Next, you construct two additional data structures: "`inc_field`", which specifies the incident electromagnetic field's wavelength and direction; and "`order`", which specifies which diffraction orders to retain in the calculations. The GD-Calc computation engine is then invoked as follows,

```
[param_size,scat_field,inc_field]=...
      gdc(grating,inc_field,order);                    (3)
```

The "param_size" output is related to parameterization, and the "`scat_field`" and "`inc_field`" outputs (i.e., "scattered field" and "incident field") are passed to an accessory function, `gdc_eff`, which converts the results to diffraction efficiencies for reflected and transmitted diffraction orders (`R` and `T`, respectively),

```
[R,T]=gdc_eff(scat_field,inc_field);                   (4)
```

(`R` and `T` are struct arrays defining diffraction efficiencies for multiple diffraction orders.)

The diffraction efficiency calculations require the calculation engine (`gdc_engine.p`), which is not part of the free demo/tutorial package; but for the photonic crystal example (gdc_demo11.m) the demo function `gdc_demo_engine.p` can be invoked in lieu of listing (3),

```
[grating,param_size,scat_field,inc_field]=...
      gdc_demo_engine(11,inc_field,order,...
      grating_pmt,d,thick,width,rep_count);            (5)
```

gdc_demo_engine constructs the `grating` structure based on the user-specified input arguments (`grating_pmt`, `d`, `thick`, `width`, and `rep_count`), which are described below.

The grating struct

The `grating` struct comprises the following elements: (1) `grating.pmt`, a cell array of complex permittivities associated with the grating materials; (2) `grating.pmt_sub_index` and `pmt_sup_index`, the grating substrate and superstrate permittivities (specified as indices into `grating.pmt`); (3) `grating.d21`, `d31`, `d22`, and `d32`, which specify the grating's fundamental period vectors; and (4) `grating.stratum`, a cell array of grating "strata" that define the grating's internal structure. Following is an example of a trivially simple grating structure, a bare tungsten substrate with a complex refractive index of `1.52+6.46*i`. (This is the approximate refractive index of tungsten at a wavelength of 1.825 μm. The permittivity is the square of the refractive index.)

```
d=1.5; % grating period
grating_pmt=(1.52+6.46*i)^2; % grating permittivity
clear grating
grating.pmt={1.0,grating_pmt};
grating.pmt_sub_index=2;
grating.pmt_sup_index=1;                                    (6)
grating.d21=d;
grating.d31=0;
grating.d22=0;
grating.d32=d;
grating.stratum={};
```

The grating period vectors are irrelevant for a bare substrate but they must nevertheless be specified, and they will become relevant as we add periodic strata to the structure. The grating geometry is specified in relation to orthonormal coordinate basis vectors $\hat{e}_1$, $\hat{e}_2$, and $\hat{e}_3$, wherein $\hat{e}_1$ is normal to the grating substrate, and $\hat{e}_2$ and $\hat{e}_3$ are parallel to the substrate. Relative to these coordinate bases, the grating's two period vectors $\vec{d}_1^{[g]}$ and $\vec{d}_2^{[g]}$ have the following coordinate representations,

$$\vec{d}_1^{[\mathrm{g}]} = \hat{e}_2\, d_{2,1}^{[\mathrm{g}]} + \hat{e}_3\, d_{3,1}^{[\mathrm{g}]} \tag{7}$$

$$\vec{d}_2^{[\mathrm{g}]} = \hat{e}_2\, d_{2,2}^{[\mathrm{g}]} + \hat{e}_3\, d_{3,2}^{[\mathrm{g}]} \tag{8}$$

The grating geometry is invariant under translation by $\vec{d}_1^{[g]}$ or $\vec{d}_2^{[g]}$, and the `grating` data fields, `grating.d21`, etc., correspond to $d_{2,1}^{[\mathrm{g}]}$, etc.

The following sections illustrate the variety of stratum types that can be incorporated in the grating. There are five stratum types, which are indicated by a type identifier in the range of 0 to 4. These are summarized below:

type index	stratum type
0	homogeneous
1	uniperiodic
2	biperiodic
3	coordinate break
4	replication module

Stratum type 0: homogeneous

The following code listing illustrates how the preceding grating specification (listing (6)) could be modified to represent a free-standing, homogeneous tungsten film of thickness $0.5\,\mu\text{m}$,

```
...
thick=0.5; % stratum thickness
...
grating.pmt_sub_index=1;
...
clear stratum
stratum.type=0; % homogeneous
stratum.thick=thick;
stratum.pmt_index=2;
grating.stratum{1}=stratum;
```
(9)

Note that `grating.pmt_sub_index` has been changed to 1, so the substrate now has permittivity 1.0 (`grating.pmt{1}` = 1.0), representing vacuum. The `stratum` struct has three data fields: the type index (`0` for homogeneous), a thickness, and a permittivity index (`pmt_index`), which indexes into `grating.pmt`.

Stratum type 1: uniperiodic

The following code excerpt replaces the homogeneous stratum of listing (9) with a uniperiodic stratum comprising free-standing, parallel, rectangular-section tungsten rods with a rod width of $0.5\,\mu\text{m}$,

```
...
width=0.5; % rod width
...
clear stratum
stratum.type=1; % uniperiodic
stratum.thick=thick;
stratum.h11=1;
stratum.h12=0;
clear stripe
stripe.c1=-0.5*width/d;
stripe.pmt_index=1;
stratum.stripe{1}=stripe;
stripe.c1=0.5*width/d;
stripe.pmt_index=2;
stratum.stripe{2}=stripe;
grating.stratum{1}=stratum;
```
(10)

Before discussing the above code, let’s first plot the structure:

```
clear pmt_display
pmt_display(1).name='';
pmt_display(1).color=[];
pmt_display(1).alpha=1;
pmt_display(2).name='Tungsten';                    (11)
pmt_display(2).color=[1,1,1]*0.75;
pmt_display(2).alpha=1;
x_limit=[-thick,-1.75*d,-1.75*d;...
      2*thick,1.75*d,1.75*d];
gdc_plot(grating,1,pmt_display,x_limit);
```

`pmt_display(1)` and `pmt_display(2)` define the display attributes for the two materials represented by `grating.pmt{1}` and `grating.pmt{2}`, respectively, in listing (6). (The assignment `pmt_display(1).color=[]` suppresses display of the first material, which is vacuum.) `x_limit` specifies 3-D plot limits, with columns 1, 2, and 3 corresponding to coordinates x_1, x_2, and x_3, respectively. (The x_1 axis is vertical, i.e. normal to the grating substrate; and the x_2 and x_3 axes are parallel to the substrate.) Figure 7 shows the `gdc_plot` result (with annotation added), which illustrates three tungsten rods. (The rods are shown as hollow, tubular elements, cut off at the display limits, although the grating is actually modeled as an infinite array of infinitely long, solid rods.)

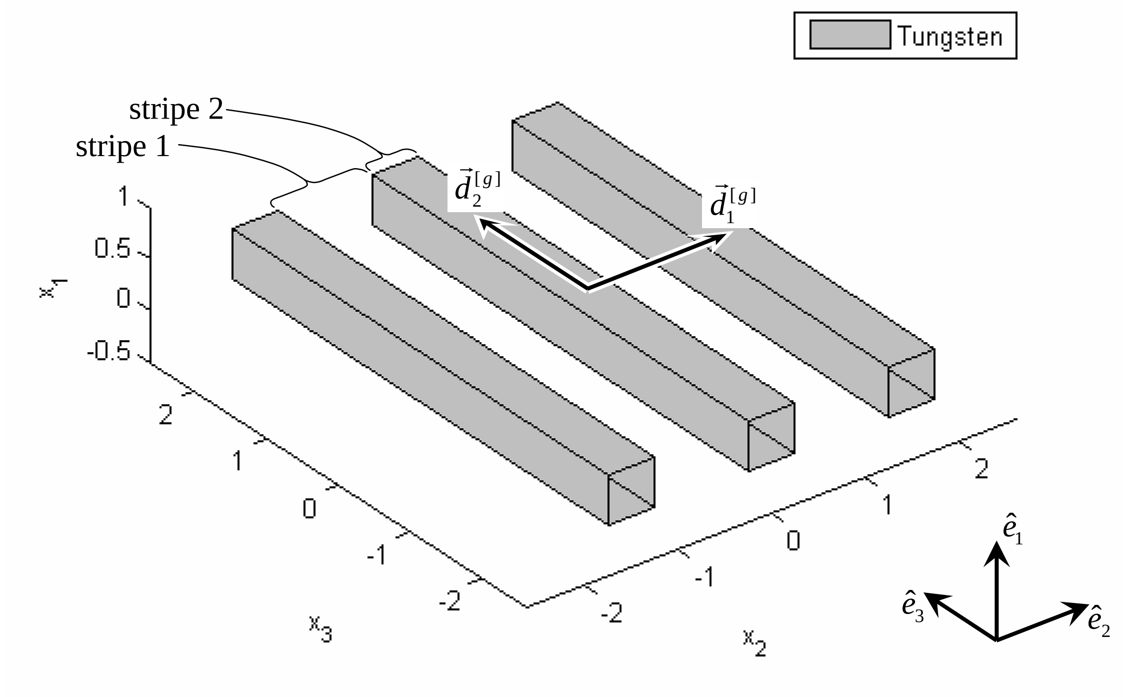

Figure 7. Uniperiodoc stratum, from listings (10) and (11).

As illustrated in listing (10), a uniperiodic stratum is defined by the following data fields: the type index (`1`), the stratum thickness, two “harmonic indices” `h11` and `h12`, and a “`stripe`” data field. The physical structure represented by the stratum comprises a periodic array of parallel, vertical-wall stripes whose periodicity and orientation are determined by the harmonic indices. With the assigned values `h11=1` and `h12=0` the stripes are parallel to $\vec{d}_2^{[g]}$ and have a periodicity defined by $\vec{d}_1^{[g]}$; see Figure 7. (An explanation of how the harmonic indices are used in general is provided below.)

The stratum comprises two stripes per period (the grating line/space pairs), which are defined by the structs `stratum.stripe{1}` and `stratum.stripe{2}`. Each stripe struct has two data fields: `c1`, which defines one of the stripe's wall positions, and `pmt_index` (an index into `grating.pmt`), which defines the stripe material. In this example, the boundary between the first and second stripes is at $x_2 = \texttt{stratum.stripe\{1\}.c1*d}$, and the boundary between the second stripe and the next adjacent stripe is at $x_2 = \texttt{stratum.stripe\{2\}.c1*d}$. (An explanation of how the stripe geometry is specified more generally is provided below.) The `stratum.stripe` cell array can be extended to define any number of stripes per period.

Stratum type 2: biperiodic

Next, we consider a stratum comprising a free-standing tungsten film with a biperiodic array of square holes. Code listing (10) is modified as follows for this example,

```
...
clear stratum
stratum.type=2; % biperiodic
stratum.thick=thick;
stratum.h11=1;
stratum.h12=0;
stratum.h21=0;
stratum.h22=1;
clear stripe
stripe.type=1; % inhomogeneous
stripe.c1=-0.5*width/d;
clear block
block.c2=-0.5*width/d;
block.pmt_index=1;                                    (12)
stripe.block{1}=block;
block.c2=0.5*width/d;
block.pmt_index=2;
stripe.block{2}=block;
stratum.stripe{1}=stripe;
clear stripe
stripe.type=0; % homogeneous
stripe.c1=0.5*width/d;
stripe.pmt_index=2;
stratum.stripe{2}=stripe;
grating.stratum{1}=stratum;
```

A plot of the grating structure (cf. listing (11)) is shown in Figure 8. The basic difference between a uniperiodic stratum (listing (10)) and a biperiodic stratum (listing (12)) is that the latter has two additional harmonic indices (`h21` and `h22`), and its stripes can be either of two types: homogeneous or inhomogeneous (indicated by a `stripe.type` field equal to `0` or `1`, respectively). A homogeneous stripe such as `stratum.stripe{2}` has the same format in either a uniperiodic or biperiodic stratum, except that in the latter context the stripe has a type identifier (`stripe.type=0`).

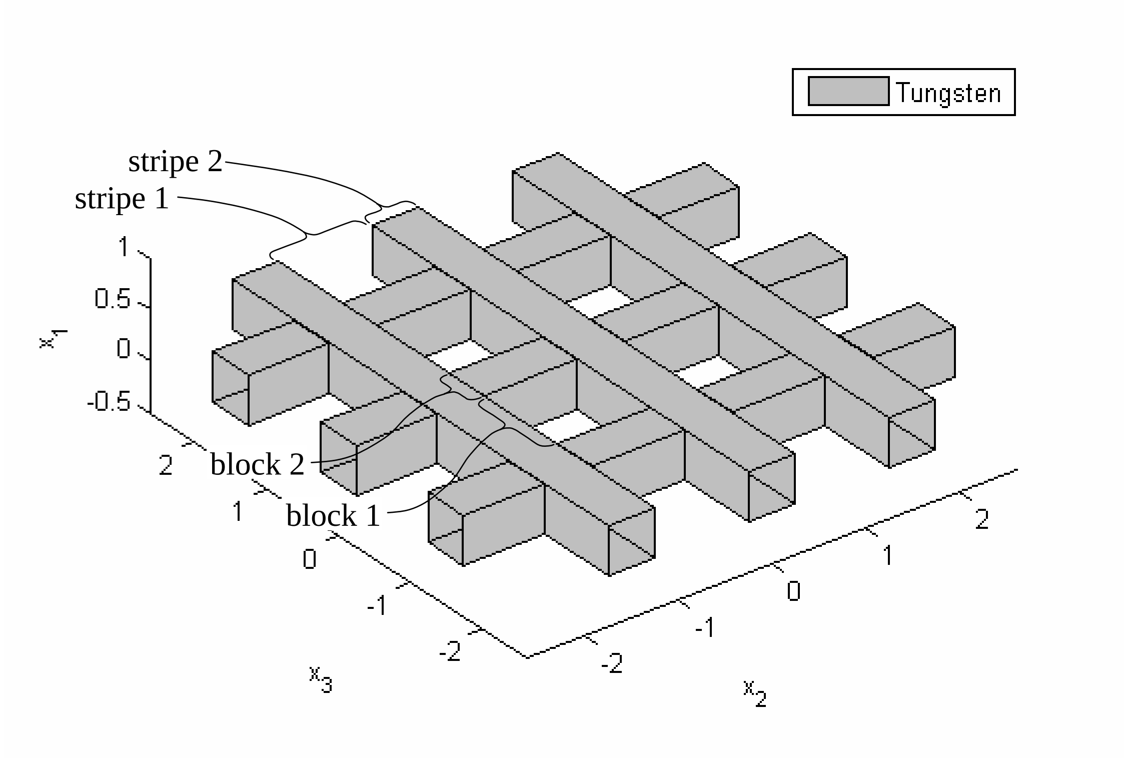

Figure 8. Biperiodoc stratum, from listing (12).

An inhomogeneous stripe (e.g. `stratum.stripe{1}` in listing (12)) comprises the type identifier (`stripe.type=1`), the `c1` data field defining the positions of boundary walls between stripes, and a "`block`" data field representing structural blocks within the stripe. As illustrated in Figure 8, the first stripe comprises two blocks per period (the square hole and the partition between holes), which are defined by `stripe.block{1}` and `stripe.block{2}`. Each block is defined by two data fields: `c2`, which defines the position of one of the walls between adjoining blocks, and `pmt_index` (an index into `grating.pmt`), which defines the block material. In this example, the boundary between the first and second blocks is at $x_3 =$ `stripe.block{1}.c2*d`, and the boundary between the second block and the next adjoining block is at $x_3 =$ `stripe.block{2}.c2*d`. (An explanation of how the block geometry is specified more generally is provided below.) The `stripe.block` cell array can be extended to define any number of blocks per period.

Harmonic indices and stratum periods

In general, a grating stratum's geometry is defined relative to stratum-specific period vectors $\vec{d}_1^{[s]}$ and $\vec{d}_2^{[s]}$, which need not be identical to the grating periods $\vec{d}_1^{[g]}$ and $\vec{d}_2^{[g]}$, but which have a relationship to $\vec{d}_1^{[g]}$ and $\vec{d}_2^{[g]}$ defined by the harmonic indices. For a biperiodic stratum, this relationship is

$$\left.\begin{aligned} \vec{d}_1^{[g]} &= \vec{d}_1^{[s]}\, h_{1,1} + \vec{d}_2^{[s]}\, h_{2,1} \\ \vec{d}_2^{[g]} &= \vec{d}_1^{[s]}\, h_{1,2} + \vec{d}_2^{[s]}\, h_{2,2} \end{aligned}\right\} \tag{13}$$

wherein $h_{1,1}$, $h_{1,2}$, $h_{2,1}$ and $h_{2,2}$ are integer-valued harmonic indices. This relationship can alternatively be expressed in terms of spatial frequencies. The grating has two fundamental spatial frequency vectors $\vec{f}_1^{[g]}$ and $\vec{f}_2^{[g]}$, which are related to $\vec{d}_1^{[g]}$ and $\vec{d}_2^{[g]}$ by the reciprocal relationships

$$\left.\begin{aligned} \vec{f}_1^{[g]} \bullet \vec{d}_1^{[g]} = 1, \quad \vec{f}_1^{[g]} \bullet \vec{d}_2^{[g]} = 0 \\ \vec{f}_2^{[g]} \bullet \vec{d}_1^{[g]} = 0, \quad \vec{f}_2^{[g]} \bullet \vec{d}_2^{[g]} = 1 \end{aligned}\right\} \tag{14}$$

If the period vectors are represented as size-[2, 1] arrays, and the frequency vectors as size-[1, 2] arrays (based on their $\hat{e}_2$ and $\hat{e}_3$ projections), then the above relationship can be expressed in MATLAB syntax as $[\vec{f}_1^{[g]};\ \vec{f}_2^{[g]}] = \texttt{inv}([\vec{d}_1^{[g]},\ \vec{d}_2^{[g]}])$. The stratum is similarly characterized by spatial frequency vectors $\vec{f}_1^{[s]}$ and $\vec{f}_2^{[s]}$, which have a similar reciprocal relationship to $\vec{d}_1^{[s]}$ and $\vec{d}_2^{[s]}$, and which are harmonics of $\vec{f}_1^{[g]}$ and $\vec{f}_2^{[g]}$,

$$\left.\begin{aligned} \vec{f}_1^{[s]} &= h_{1,1}\ \vec{f}_1^{[g]} + h_{1,2}\ \vec{f}_2^{[g]} \\ \vec{f}_2^{[s]} &= h_{2,1}\ \vec{f}_1^{[g]} + h_{2,2}\ \vec{f}_2^{[g]} \end{aligned}\right\} \tag{15}$$

The above relationships apply to a biperiodic stratum. A uniperiodic stratum is characterized by a single period vector $\vec{d}_1^{[s]}$, which is orthogonal to the stratum stripes, and a single frequency vector $\vec{f}_1^{[s]}$, which is parallel to $\vec{d}_1^{[s]}$ and which satisfies the relationship $\vec{f}_1^{[s]} \bullet \vec{d}_1^{[s]} = 1$, i.e.,

$$\vec{f}_1^{[s]} = \frac{\vec{d}_1^{[s]}}{\vec{d}_1^{[s]} \bullet \vec{d}_1^{[s]}} \qquad \text{(uniperiodic)} \tag{16}$$

A uniperiodic stratum's frequency vector $\vec{f}_1^{[s]}$ is defined by the two harmonic indices $h_{1,1}$ and $h_{1,2}$,

$$\vec{f}_1^{\,[s]} = h_{1,1}\,\vec{f}_1^{\,[g]} + h_{1,2}\,\vec{f}_2^{\,[g]} \qquad \text{(uniperiodic)} \qquad (17)$$

A stratum's geometry is defined relative to its period vectors $\vec{d}_1^{\,[s]}$ and $\vec{d}_2^{\,[s]}$ (or just $\vec{d}_1^{\,[s]}$ for a uniperiodic stratum) as illustrated in Figure 9. (The coordinate origin is represented as $\vec{0}$ in the figure.) The stratum's stripe list comprises elements `stripe{`l_2`}`, $l_2 = 1 \ldots L_2$, wherein L_2 is the number of stripes per period; and the range of l_2 is implicitly extended to $l_2 = -\infty \ldots \infty$ by periodicity[3]. The boundary wall between `stripe{`l_2`}` and `stripe{`l_2`+1}` is parallel to $\vec{d}_2^{\,[s]}$ (or in the case of a uniperiodic stratum, perpendicular to $\vec{d}_1^{\,[s]}$) and intercepts the point `stripe{`l_2`}.c1*`$\vec{d}_1^{\,[s]}$.

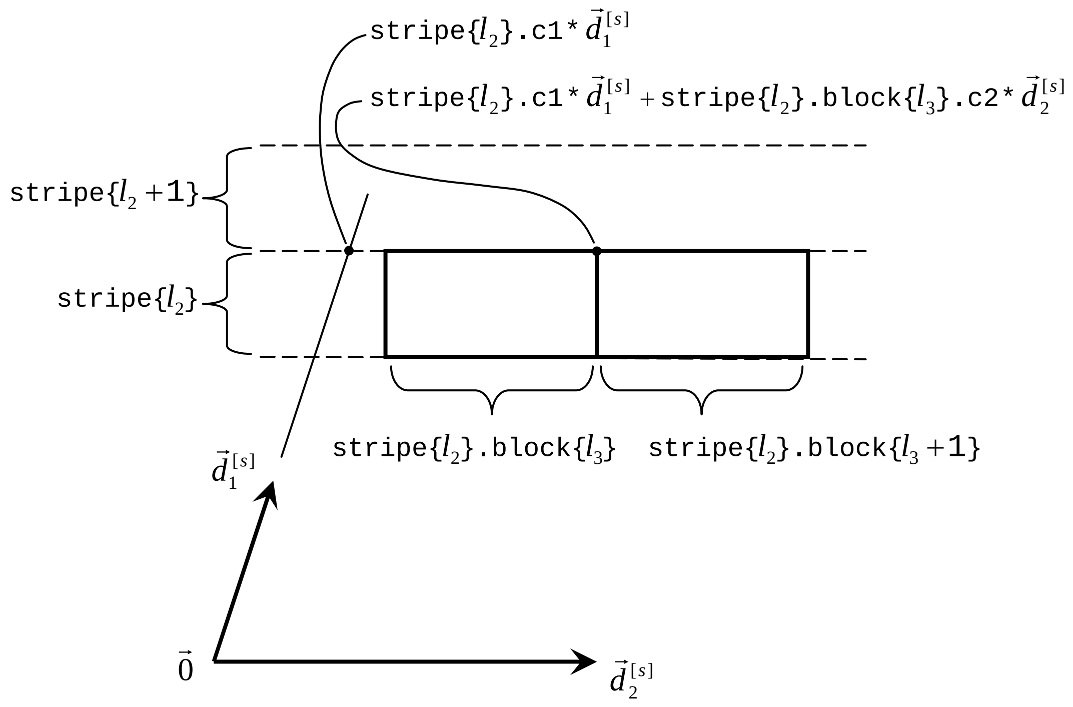
stripe{l_2}.c1*$\vec{d}_1^{\,[s]}$
stripe{l_2}.c1*$\vec{d}_1^{\,[s]}$+stripe{l_2}.block{l_3}.c2*$\vec{d}_2^{\,[s]}$
stripe{l_2+1}
stripe{l_2}
stripe{l_2}.block{l_3}
stripe{l_2}.block{l_3+1}
$\vec{d}_1^{\,[s]}$
$\vec{0}$
$\vec{d}_2^{\,[s]}$

Figure 9. Stratum geometry definition.

If `stripe{`l_2`}` is inhomogeneous, it comprises structural blocks `stripe{`l_2`}.block{`l_3`}`, $l_3 = 1 \ldots L_3$, wherein L_3 is the number of blocks per period. The range of l_3 is implicitly extended to $l_3 = -\infty \ldots \infty$ by periodicity. The blocks are rectangular, and the boundary wall between `stripe{`l_2`}.block{`l_3`}` and

[3] The GDC-Calc documentation generally uses the "l_1", "l_2" and "l_3" indices to label strata, stripes, and blocks, respectively.

`stripe{`l_2`}.block{`l_3+1`}` intercepts the point `stripe{`l_2`}.c1*`$\vec{d}_1^{[s]}$ + `stripe{`l_2`}.block{`l_3`}.c2*`$\vec{d}_2^{[s]}$.

To illustrate the use of harmonic indices, code listing (18) extends listing (10) to add a second stratum to the grating. The second stratum is identical to the first, except that its stripe orientation is rotated by 90° by swapping `h11` and `h12`. Figure 10 shows a plot of the grating structure. (In generating Figure 10, the assignment `x_limit(2,1)= 3*thick` is made to extend the plot limits; cf. listing (11). The limit should be similarly extended in other figures to follow.)

```
...
stratum.h11=1;
stratum.h12=0;
...                                                  (18)
grating.stratum{1}=stratum; % same as listing (10)
stratum.h11=0; % Swap h11 and h12.
stratum.h12=1;
grating.stratum{2}=stratum;
```

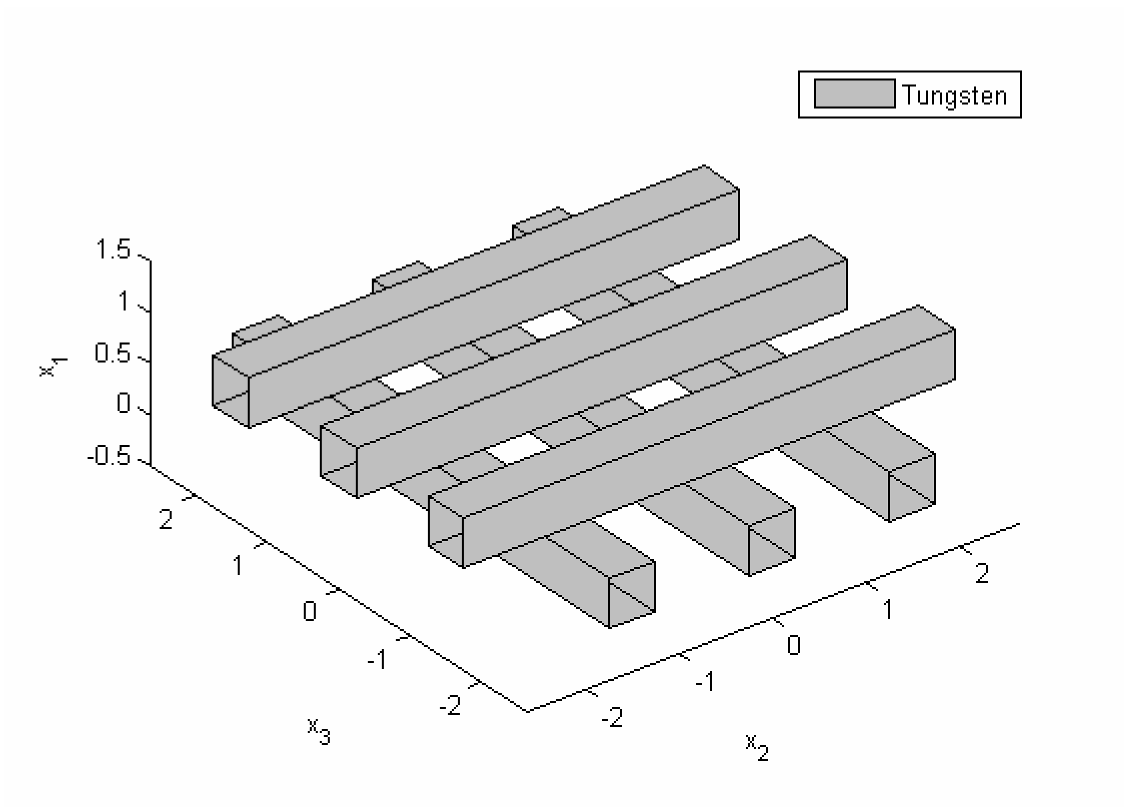

Figure 10. Grating structure from listing (18).

Stratum type 3: coordinate break

The coordinate-break stratum type does not represent a physical grating layer, but it is abstractly classified as a "stratum" of zero thickness, in the sense that it is associated with a lateral plane at a particular x_1 height in the grating. It has the effect of applying a specified lateral (x_2, x_3) translational shift to all strata above the coordinate break. Listing (19) illustrates the use of a coordinate break. This example extends listing (18) by inserting a coordinate break above the second stratum (this applies a half-period lateral shift to both the x_2 and x_3 coordinates), and then adding copies of the first two strata to the grating. As illustrated in Figure 11, the translational shift is applied to the top two strata.

```
...
grating.stratum{1}=...; % same as listing (18)
...
grating.stratum{2}=...; % same as listing (18)
clear stratum
stratum.type=3; % coordinate break
stratum.dx2=d/2; % half-period x2-shift
stratum.dx3=d/2; % half-period x3-shift
grating.stratum{3}=stratum;
grating.stratum{4}=grating.stratum{1};
grating.stratum{5}=grating.stratum{2};
```

(19)

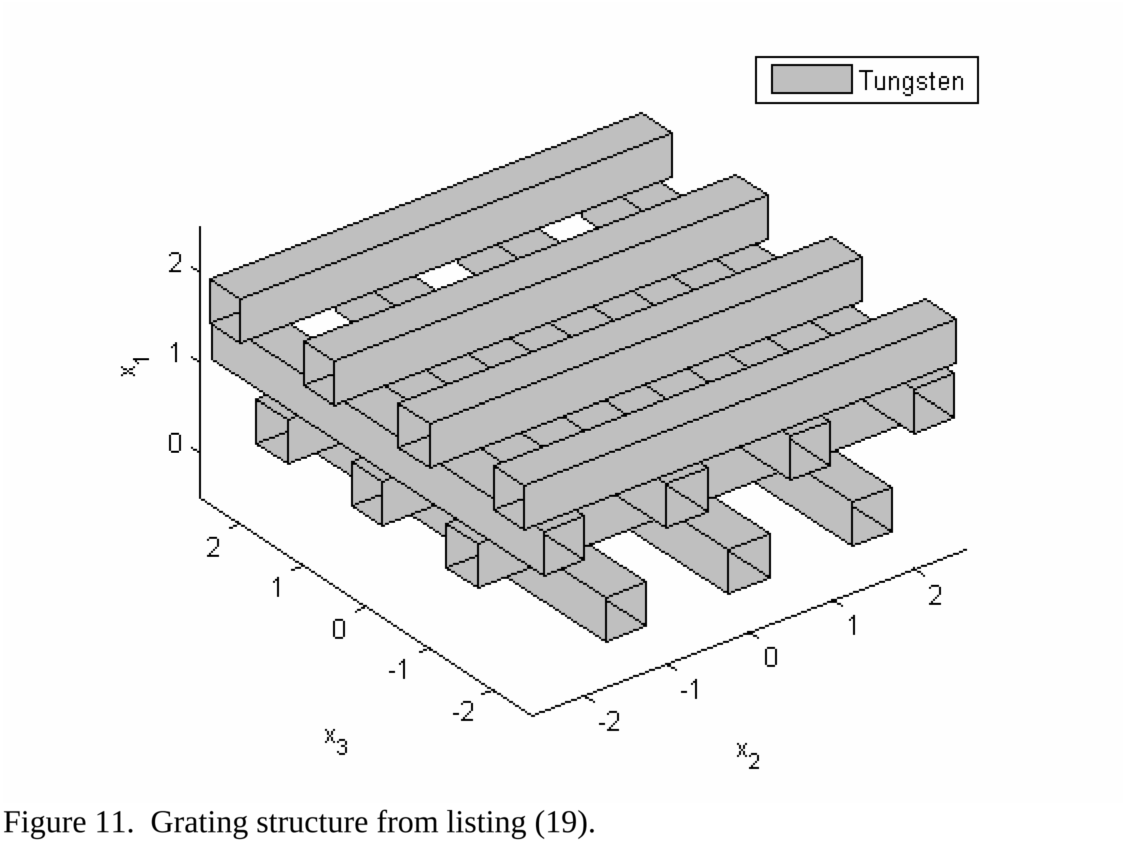

Figure 11. Grating structure from listing (19).

Stratum type 4: replication module

A three-dimensionally periodic ("tri-periodic") grating structure could be defined by first constructing the strata for one period in the vertical direction, and then iteratively replicating the first period. For example, listing (19) could be modified to define a photonic crystal structure by replacing the last two lines with the following loop,

```
for l1=4:11
    grating.stratum{l1}=grating.stratum{l1-3};
end
```
(20)

However, a more efficient way to define this type of structure would be to use a "replication module", which is a composite type of stratum comprising an encapsulated list of strata (for one period) and a replication count.

The following code excerpt illustrates the use of a replication module. The three strata encapsulated by the replication module (`stratum.stratum{1}`, `stratum.stratum{2}`, `stratum.stratum{3}`) are defined the same way as the first three grating strata in listings (18) and (19). Figure 12 illustrates the grating structure defined by listing (21).

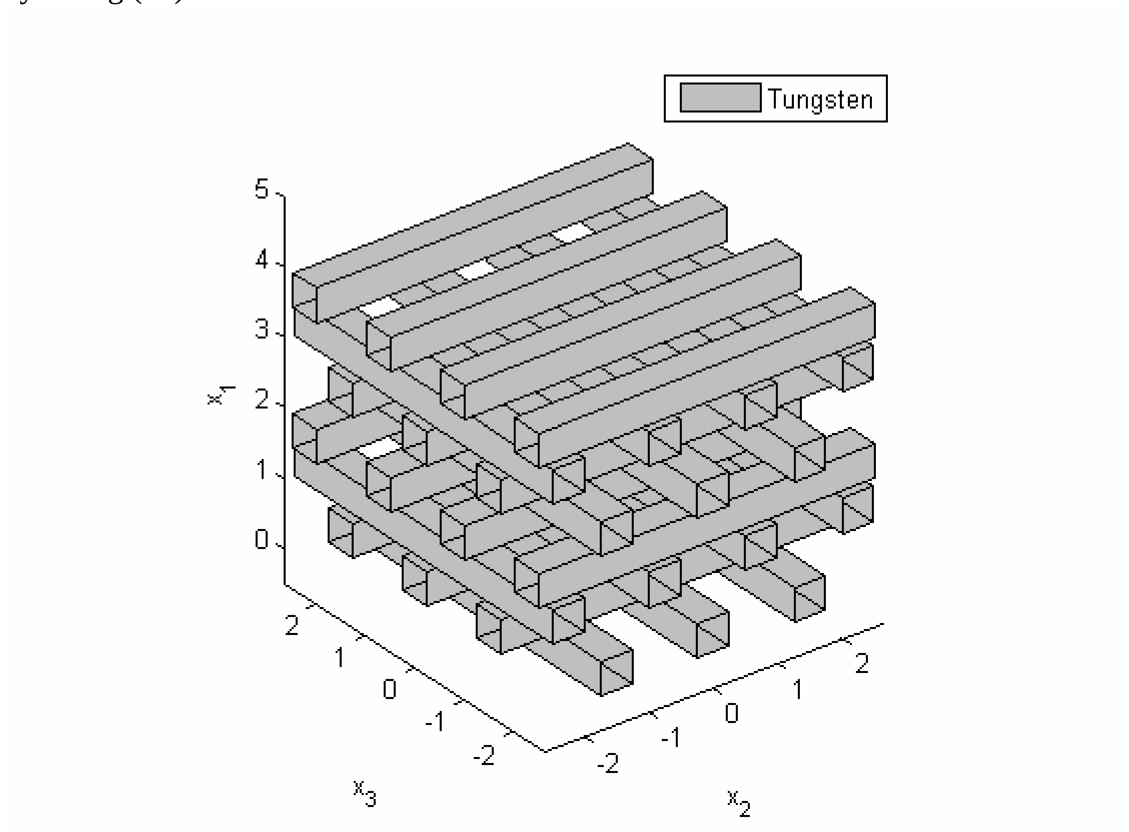

Figure 12. Grating structure from listing 21.

```
...
clear stratum
stratum.type=4; % replication module
stratum.stratum{1}=...; % same as previous grating.stratum{1}
stratum.stratum{2}=...; % same as previous grating.stratum{2}
stratum.stratum{3}=...; % same as previous grating.stratum{3}
stratum.rep_count=4; % replication count
grating.stratum={stratum};
```

(21)

Incident field and diffraction order selection

The diffraction calculations assume a plane-wave incident electromagnetic field characterized by its spatial frequency vector $\vec{f}^{\,[i]}$. Using polar coordinates, this vector can be defined as

$$\vec{f}^{\,[i]} = \tfrac{1}{\lambda}\left(-\hat{e}_1 \cos\theta + \hat{e}_2 \sin\theta\cos\phi + \hat{e}_3 \sin\theta\sin\phi\right) \tag{22}$$

wherein λ is wavelength, and θ and ϕ are polar and azimuth angles of incidence. The `inc_field` struct specifies the wavelength and the $\hat{e}_2$ and $\hat{e}_3$ projections of $\vec{f}^{\,[i]}$ (denoted as `f2` and `f3`), e.g.,

```
wavelength=1.825;
theta=0.0;
phi=0.0;
inc_field.wavelength= wavelength;
inc_field.f2=sin(theta)*cos(phi)/wavelength;
inc_field.f3=sin(theta)*sin(phi)/wavelength;
```

(23)

(The incident field's polarization state is not specified because the GD-Calc output can be used to determine diffraction efficiencies for any incident polarization state.)

The diffracted electromagnetic field comprises diffraction orders whose grating-tangential spatial frequencies differ from that of the incident field by increments $m_1\,\vec{f}_1^{\,[g]} + m_2\,\vec{f}_2^{\,[g]}$, wherein m_1 and m_2 are integer-valued order indices. Only a finite number of orders are retained in diffraction calculations, and GD-Calc provides the user full control over which orders are retained. The orders are specified by the "`order`" struct array, each element of which corresponds to a specific m_2 index and a list of m_1 indices associated with that m_2 value. Typically, the order selection is defined by "rectangular" order truncation: $|m_1| \le$ `m_max` and $|m_2| \le$ `m_max` for some truncation limit `m_max`, as defined by the following code excerpt,

```
m_max=10;
order=[];
m1=-m_max:m_max;
for m2=-m_max:m_max
    order(end+1).m2=m2;
    order(end).m1=m1;
end
```
(24)

However, alternative truncation conditions can be used. For example, the photonic crystal grating described above can be analyzed using a "diagonal" truncation method defined by the conditions $|m_1| + |m_2| \le$ `m_max`. (The two methods exhibit similar numerical convergence with respect to `m_max`, but the computation time is reduced by about a factor of 8 by using diagonal truncation.) The following code excerpt illustrates diffraction order selection using diagonal truncation,

```
m_max=10;
order=[];
m1=-m_max:m_max;
for m2=-m_max:m_max
    order(end+1).m2=m2;
    order(end).m1=m1(abs(m1)+abs(m2)<=m_max);
end
```
(25)

In the previous checkerboard grating example, using basis periods $\vec{A}$ and $\vec{B}$ (Figure 6), all orders with $m_1 - m_2$ odd would be identically zero, These can be eliminated from the order selection by modifying the above `order(end).m1` assignment to select only m_1 indices for which $m_1 - m_2$ is even,

```
order(end).m1 = m1(mod(m1-m2,2)==0);
```
(26)

Going further

The m-file comment headers (especially gdc.m, gdc_eff.m, and gdc_plot.m) define the GD-Calc software interface more completely. GD-Calc_Demo.pdf provides additional technical background tutorial examples (including diffraction calculations) and computational performance data for a variety of grating types. (All of the code examples can be run with the demo/tutorial package, which can be freely downloaded from the GD-Calc website.) For detailed technical background on the theoretical basis of GD-Calc, see GD-Calc.pdf.