Campo local superficial

W. Luis Mochán

30 de mayo de 2021

1. Introducción

La respuesta macroscópica de un sistema no es el simple promedio de su respuesta microscópica. Esto se debe a que el campo eléctrico microscópico tiene fluctuaciones espaciales que están correlacionadas con la textura espacial del sistema. Este efecto es conocido como el *efecto de campo local*. Un ejemplo muy conocido de este efecto es el de un sólido isotrópico modelado como una red cúbica de entidades polarizables puntuales caracterizadas por su polarizabilidad α. Sin efecto de campo local, la respuesta dieléctrica macroscópica sería

$$\epsilon_M = 1 + 4\pi n\alpha, \tag{1}$$

con n la densidad de las entidades polarizables. Sin embargo, al tomar en cuenta que la polarizabilidad es la respuesta no sólo al campo externo, sino también al campo producido por todos los dipolos vecinos, la ecuación anterior se ve reemplazada por la relación de Claussius-Mossoti,

$$\frac{\epsilon_M - 1}{\epsilon_M + 2} = 4\pi n\frac{\alpha}{3}, \tag{2}$$

$$\epsilon_M = \frac{1 + 8\pi n\alpha/3}{1 - 4\pi n\alpha/3}. \tag{3}$$

Como la corrección de campo local depende de la interacción con entidades polarizables vecinas, podríamos esperar que se vea modificada en la vecindad de una superficie. El propósito de estas notas es mostrar cómo podríamos calcular el efecto de campo local superficial y explorar sus consecuencias.

2. Teoría

La interacción entre planos cristalinos está definida por el tensor

$$\phi^q(\boldsymbol{r}) = \sum_{\boldsymbol{R}} \frac{1}{||\boldsymbol{r} - \boldsymbol{R}||}. \tag{4}$$

La ecuación previa no es muy útil por la lenta convergencia en el espacio real del potencial Coulombiano. Conviene entonces recurrir a la ecuación diferencial del potencial,

$$\nabla^2 \phi^q(\boldsymbol{r}) = -4\pi \sum_{\boldsymbol{R}} \delta(\boldsymbol{r} - \boldsymbol{R}), \tag{5}$$

en el espacio recíproco $\{\boldsymbol{G}\}$ definido por $\boldsymbol{G} \cdot \boldsymbol{R} = 2\pi \times$ entero,

$$\left(\frac{d^2}{dz^2} - G^2\right) \phi^q_{\boldsymbol{G}}(z) = -4\pi\delta(z), \tag{6}$$

con $\phi^q_{\boldsymbol{G}}(z)$ es el coeficiente de Fourier 2D del potencial $\phi(\boldsymbol{r})$ evaluado a la altura z. La solución que decae al alejarnos del plano xy para $G \neq 0$ es

$$\phi^q_{\boldsymbol{G}}(z) = \frac{2\pi}{A} \frac{e^{-G|z|}}{G}. \tag{7}$$

Regresando al espacio real, el potencial queda dado por

$$\phi^q(\boldsymbol{r}) = -\frac{2\pi}{A}|z| + \frac{2\pi}{A} \sum_{\boldsymbol{G}\neq 0} \frac{e^{i\boldsymbol{g}_\pm \cdot \boldsymbol{r}}}{G}. \tag{8}$$

donde añadimos al termino $G = 0$ correspondiente al potencial producido por una película uniformemente cargada, y dónde introdujimos los vectores

$$\boldsymbol{g}_\pm = (\boldsymbol{G}, \pm iG), \tag{9}$$

y empleamos el signo $+$ cuando $z > 0$ y el signo $-$ cuando $z < 0$. Esta es una serie rápidamente convergente siempre y cuando $z \neq 0$.

Consideremos ahora la misma red de Bravais, pero ocupada por dipolos idénticos $\boldsymbol{p}$. La densidad de carga sería entonces

$$\rho(\boldsymbol{r}) = -\boldsymbol{p} \cdot \nabla \sum_{\boldsymbol{R}} \delta(\boldsymbol{r} - \boldsymbol{R}), \tag{10}$$

i.e., operamos sobre la densidad monopolar con el operador $-\boldsymbol{p}\cdot\nabla$. Luego, el potencial dipolar ϕ^p se obtiene entonces aplicando el mismo operador sobre el potencial monopolar, $\phi^p = -\boldsymbol{p}\cdot\phi^q$,

$$\phi^p(\boldsymbol{r}) = \mp p_z\frac{2\pi}{A} - \frac{2\pi}{A}\sum_{\boldsymbol{G}\neq 0} i\frac{\boldsymbol{g}_\pm}{G}\cdot\boldsymbol{p}e^{i\boldsymbol{g}_\pm\cdot\boldsymbol{r}}, \tag{11}$$

y el campo eléctrico $\boldsymbol{E} = -\nabla\phi^p$,

$$\boldsymbol{E}(\boldsymbol{r}) = -\frac{2\pi}{A}\sum_{\boldsymbol{G}}{}' \frac{\boldsymbol{g}_\pm\boldsymbol{g}_\pm}{G}\cdot\boldsymbol{p}e^{i\boldsymbol{g}_\pm\cdot\boldsymbol{r}}. \tag{12}$$

Construyamos ahora un cristal con caras planas orientadas de acuerdo a alguna dirección cristalográfica. Todos los sitios del plano son equivalentes y descritos por una red de Bravais 2D. Entonces podemos escribir la ecuación que obedece la i-ésima entidad polarizable,

$$\boldsymbol{p}_i = \alpha\left(\boldsymbol{E}_{\text{ex}} + \sum \boldsymbol{T}_{ij}\cdot\boldsymbol{p}_j\right), \tag{13}$$

donde $\boldsymbol{T}_{ij}\cdot\boldsymbol{p}_j$ representa el campo eléctrico en el sitio i producido por un dipolo que $\boldsymbol{p}_j$ en el sitio j. Suponiendo que todos los dipolos de un plano son equivalentes, podemos podemos sumar las interacciones por planos cristalinos. Definimos entonces

$$\boldsymbol{T}_{nm} = \sum_{j\in m}\boldsymbol{T}_{ij}, \tag{14}$$

donde i es un sitio cualquiera del plano n. Usando el campo dipolar previo, la interacción entre planos cristalinos está dada por el tensor

$$\boldsymbol{T}_{nm} = -2\pi\sum_{\boldsymbol{G}}\frac{\boldsymbol{g}_\pm\boldsymbol{g}_\pm}{AG}e^{i\boldsymbol{g}_\pm\cdot\boldsymbol{c}(n-m)}, \quad (n\neq m) \tag{15}$$

donde $\boldsymbol{G}$ son vectores recíprocos bidimensionales, $\boldsymbol{c}$ es un vector que va de un sitio del plano 0 a uno del plano 1, y donde usamos el signo + cuando $n > m$ y el signo − cuando $n < m$. Para el caso $n = m$ podemos usar la regla de suma

$$\sum_m \boldsymbol{T}_{nm} = \frac{4\pi n}{3}\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & -2\end{pmatrix} \tag{16}$$

en el bulto del sistema, con n la densidad de entidades polarizables, y despejar $\boldsymbol{T}_{(n)(n)}$ (sin suma), la autointeracción dipolar de un plano.

3. Interacciones

Empezamos por cargar librerías y opciones.

```
# name: init
use strict;
use warnings;
use v5.12;

use Getopt::Long; # Read options
use Scalar::Util qw(looks_like_number);
use Exporter::Renaming;
use List::Util Renaming=>[all=>'lu_all']; # Rename all method

use PDL;  # Perl Data Language
use PDL::NiceSlice;
use PDL::Constants qw(PI);
```

Necesitamos una rutina para mandar mensajes de error y medio explicar el uso del programa.

```
# name: usage
sub usage($$) {
    my ($options, $message)=@_;
    say $message;
    say $options;
    exit 1;
}
```

También requrimos algunas rutinas de utilería. Definimos el producto escalar (no hermitiano) entre vectores posiblemente complejos. (La rutina de `PDL` no sabe aún qué hacer con complejos)

```
# name: cinner
sub cinner($$) {
    my ($a,$b)=@_;
    return ($a*$b)->sumover;
}
```

Otra rutina de utilería para acotar el número de dígitos decimales a imprimir.

```
  # name: trunc
  sub trunc($@) {
      my $size=shift; # how many decimal digits to keep
      my @a=map {s{(\.\d{$size})\d*}{$1}g; s{[ ]+}{  }g; $_}
                map {sprintf "%s", $_} @_;
      return @a;
  }
```

Define, lee y valida parámetros de la línea de comandos.

```
  # name: parameters
  my ($a,$b); # 2D basis.
  my $c;  # 3D Separation between sites in nearby planes
  my $pqmax;   # index of largest reciprocal vector
  my $dmax;   # index of largest distance to calculate
  my $N; # seminumber of planes of film
  my $digits=7; # number of decimal digits to print
  my $options=q(
      'a=s'=>\$a, # 2D basis vector x,y
      'b=s'=>\$b, # 2D basis vector x,y
      'c=s'=>\$c, # 3D separation x,y,z
      'pqmax=i'=>\$pqmax,  # index of largest reciprocal vector
      'dmax=i'=>\$dmax,  # index of largest distance to calculate
      'N=i'=>\$N, # seminumber of planes of film
      'digits=i'=>\$digits, # number of decimal digits to print
      );
  my %options=(eval $options);
  die "Bad option definition: $@" if $@;
  GetOptions(%options) or usage $options, "Bad options";
  usage $options, "Undefined parameters"
unless lu_all {defined $_} ($a, $b, $c, $pqmax, $dmax, $N);
  usage $options, "Vectors should be comma separated list of numbers"
unless lu_all {looks_like_number $_} map {split ','} ($a, $b, $c);
  #convert from strings to vectors
  ($a,$b,$c) = map {pdl(split ',', $_)} ($a, $b, $c);
  usage $options, "Basis vectors should be 2D"
unless lu_all {$_->dims==1 and $_->dim(0)==2} ($a, $b);
  usage $options, "c should be 3D" unless $c->dims==1 and $c->dim(0)==3;
  usage $options, "Third component of c should be positive" unless $c->((2)) > 0;
  usage $options, "pqmax should be positive" unless $pqmax > 0;
  usage $options, "dmax should be positive" unless $dmax > 0;
```

```
usage $options, "N should be positive" unless $N > 0;
```

Calcula el área de la celda unitaria 2D $||\boldsymbol{a} \times \boldsymbol{b}||$, el volumen de la celda unitaria 3D $|\boldsymbol{a} \times \boldsymbol{b} \cdot \boldsymbol{c}|$ y la densidad

```
# name: area
my $A=($a->((0))*$b->((1))-$a->((1))*$b->((0)))->abs;
my $V=$A*$c->((2));
my $density=1/$V;
```

Genera la red recíproca $\boldsymbol{G}_{pq} = p\boldsymbol{a}_1^* + q\boldsymbol{a}_2^*$, donde $\boldsymbol{a}_i^*$ son los elementos de la base dual, $\boldsymbol{a}_i^* \cdot \boldsymbol{a}_j = 2\pi\delta_{ij}$. En 2D, $\boldsymbol{a}_1^* = 2\pi\boldsymbol{a}_2^\perp/\boldsymbol{a}_2^\perp \cdot \boldsymbol{a}_1$ y $\boldsymbol{a}_2^* = 2\pi\boldsymbol{a}_1^\perp/\boldsymbol{a}_1^\perp \cdot \boldsymbol{a}_2$. En el programa uso `$a` y `$b` para denotar la base y `$Ga$ y ~$Gb` para la base dual.

```
# name: reciprocal
my ($a_perp, $b_perp)=map {pdl(-$_->((1)), $_->((0)))} ($a, $b);
my ($Ga, $Gb)=map {2*PI*$_->[0]/inner($_->[0], $_->[1])}
  ([$b_perp, $a], [$a_perp, $b]);
my $pq=zeroes(2*$pqmax+1,2*$pqmax+1)->ndcoords-$pqmax; #i,p,q
my $G=$pq->((0),*1)*$Ga+$pq->((1),*1)*$Gb;  #i,p,q
```

Genera los vectores 3D $\boldsymbol{g}_\pm = (\boldsymbol{G}, \pm iG)$.

```
# name: reciprocal-decay
my $G_abs=inner($G, $G)->sqrt; # p,q
my ($g_p, $g_m)=map {append($G, ($_*i()*$G_abs)->(*1))} (+1,-1); # i,p,q
```

Con ellos arma los tensores $\boldsymbol{T}_\pm = -2\pi\boldsymbol{g}_\pm\boldsymbol{g}_\pm/(AG)$.

```
# name: T+-
# i,j,gx,gy
my ($T_p, $T_m)= map {-2*PI/$A*$_->(*1,:)*$_->(:,*1)/$G_abs->(*1,*1)}
     ($g_p, $g_m); # i,j,p,q
$_->(:,:,$pqmax,$pqmax).=0 foreach ($T_p, $T_m); # Fix division by 0
```

Ahora arma las interacciones $\boldsymbol{T}_{n0}$ para n positiva y negativa.

```
# name: interactions
my $ns=sequence($dmax)+1; # n
my $exp_p=exp(i()*cinner($g_p, $c))**$ns->(*1,*1);   #p,q,n
my $exp_m=exp(-i()*cinner($g_m, $c))**$ns->(*1,*1);  #p,q,n
my $T_n0=($T_p #i,j,p,q
             *$exp_p->(*1,*1)) #i,j,p,q,n
```

```
    ->mv(-2,0)->sumover->mv(-2,0)->sumover; #i,j,n
my $T_mn0=($T_m #i,j,p,q
              *$exp_m->(*1,*1)) #i,j,p,q,n
    ->mv(-2,0)->sumover->mv(-2,0)->sumover; #i,j,n
```

Usa la regla de suma de las interacciones dipolares para obtener la autointeracción.

```
# name: selfinteraction
my $T_00 = 4*PI*$density/3*pdl([1,0,0],[0,1,0],[0,0,-2]) -
    $T_n0->mv(-1,0)->sumover - $T_mn0->mv(-1,0)->sumover;
```

Arma las interacciones $\boldsymbol{T}_{nm}$ por bloques para una película con $2 * N + 1$ planos. `T_all` tiene la interacción $n0$ en el sitio `$N2+$n`.

```
# name: interactions-film
my $N2=2*$N+1; # Number of planes
my $T_all=zeroes(cdouble, 3,3,2*$N2+1);
$T_all->(:,:,$N2+1:$N2+$dmax).=$T_n0;
$T_all->(:,:,$N2-1:$N2-$dmax).=$T_mn0;
$T_all->(:,:,($N2)).=$T_00;
my $T_nm=zeroes(cdouble,3,3,$N2,$N2);
for my $n(0..$N2-1){
    for my $m(0..$N2-1){
        $T_nm->(:,:,($n),($m)).=$T_all(:,:,($n-$m+$N2));
    }
}
```

Arma las interacciones faltantes para la misma película. Estas son $\boldsymbol{M}_n = \sum_{\text{fuera}} \boldsymbol{T}_{nm}$ para m menor a cero o m mayor al ancho de la película. Entonces, para los sitios $m < 0$, $n - m = n + 1 \ldots \infty$ y para los sitios $m > N_2$, $n - m = -\infty \ldots n - N_2$, correspondientes a los rangos `$N2+$n1+1:2*$N2` y `0:$n` de `$T_all`.

```
# name: missing
my $M_n=zeroes(cdouble,3,3,$N2);
for my $n(0..$N2-1){
    $M_n->(:,:,($n)).=
        $T_all->(:,:,$N2+$n+1:2*$N2)->mv(-1,0)->sumover
       +$T_all(:,:,0:$n)->mv(-1,0)->sumover;
}
```

Con estos pedazos podemos armar un primer programa que calcula y reporta las interacciones para una película.

```
# name: interactions.pl
<<init>>
<<usage>>
<<cinner>>
<<trunc>>
<<parameters>>
<<area>>
<<reciprocal>>
<<reciprocal-decay>>
<<T+->>
<<interactions>>
<<selfinteraction>>
<<interactions-film>>
<<missing>>
my @dir=qw(xx yy zz);
say "Diagonal components of the interaction";
say "$dir[$_]: ", trunc $digits, $T_nm->(($_),($_)) for (0..2);
say "Missing terms due to surface";
say "$dir[$_]: ", trunc $digits,$M_n->(($_),($_)) for (0..2);
```

Podemos correrlo como a continuación para la superficie $(0,0,1)$ de una red cúbica $\boldsymbol{a} = (1,0)$ $\boldsymbol{b} = (0,1)$, $\boldsymbol{c} = (0,0,1)$, truncando la red recíproca en $-2 \leq p,q \leq 2$, permitiendo interacciones con segundos vecinos y en una película de semiancho 3 (ancho 7):

```
./interactions.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2 -N 3
```

Resultados:

```
Diagonal components of the interaction
xx:
[
  [  4.5168041  -0.1637295  -0.0002774  0  0  0  0]
  [  -0.1637295  4.5168041  -0.1637295  -0.0002774  0  0  0]
  [-0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774  0  0]
  [  0  -0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774  0]
  [  0  0  -0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774]
```

```
  [  0  0  0  -0.0002774  -0.1637295  4.5168041  -0.1637295]
  [  0  0  0  0  -0.0002774  -0.1637295  4.5168041]
]

yy:
[
  [  4.5168041  -0.1637295  -0.0002774  0  0  0  0]
  [  -0.1637295  4.5168041  -0.1637295  -0.0002774  0  0  0]
  [-0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774  0  0]
  [  0  -0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774  0]
  [  0  0  -0.0002774  -0.1637295  4.5168041  -0.1637295  -0.0002774]
  [  0  0  0  -0.0002774  -0.1637295  4.5168041  -0.1637295]
  [  0  0  0  0  -0.0002774  -0.1637295  4.5168041]
]

zz:
[
  [  -9.0336083  0.3274590  0.0005549  0  0  0  0]
  [  0.3274590  -9.0336083  0.3274590  0.0005549  0  0  0]
  [0.0005549  0.3274590  -9.0336083  0.3274590  0.0005549  0  0]
  [  0  0.0005549  0.3274590  -9.0336083  0.3274590  0.0005549  0]
  [  0  0  0.0005549  0.3274590  -9.0336083  0.3274590  0.0005549]
  [  0  0  0  0.0005549  0.3274590  -9.0336083  0.3274590]
  [  0  0  0  0  0.0005549  0.3274590  -9.0336083]
]

Missing terms due to surface
xx: [-0.1640069  -0.0002774  0  0  0  -0.0002774  -0.1640069]
yy: [-0.1640069  -0.0002774  0  0  0  -0.0002774  -0.1640069]
zz: [0.3280139  0.0005549  0  0  0  0.0005549  0.3280139]
```

Ahora repito el cálculo pero truncando la red recíproca en $-3 \leq p, q \leq 3$,

```
./interactions.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 3 -dmax 3 -N 3
```

Resultados:

```
Diagonal components of the interaction
xx:
[
  [  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0  0  0]
```

```
  [  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0  0]
  [-0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0]
  [-5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.14495
  [  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774]
  [  0  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323]
  [  0  0  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108]
]

yy:
[
  [  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0  0  0]
  [  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0  0]
  [-0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.1449511e-07  0]
  [-5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774  -5.14495
  [  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323  -0.0002774]
  [  0  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108  -0.1637323]
  [  0  0  0  -5.1449511e-07  -0.0002774  -0.1637323  4.5168108]
]

zz:
[
  [  -9.0336216  0.3274646  0.0005549  1.0289902e-06  0  0  0]
  [  0.3274646  -9.0336216  0.3274646  0.0005549  1.0289902e-06  0  0]
  [0.0005549  0.3274646  -9.0336216  0.3274646  0.0005549  1.0289902e-06  0]
  [1.0289902e-06  0.0005549  0.3274646  -9.0336216  0.3274646  0.0005549  1.0289902e-0
  [  0  1.0289902e-06  0.0005549  0.3274646  -9.0336216  0.3274646  0.0005549]
  [  0  0  1.0289902e-06  0.0005549  0.3274646  -9.0336216  0.3274646]
  [  0  0  0  1.0289902e-06  0.0005549  0.3274646  -9.0336216]
]

Missing terms due to surface
xx: [-0.1640103  -0.0002780  -5.1449511e-07  0  -5.1449511e-07  -0.0002780  -0.1640103]
yy: [-0.1640103  -0.0002780  -5.1449511e-07  0  -5.1449511e-07  -0.0002780  -0.1640103]
zz: [0.3280206  0.0005560  1.0289902e-06  0  1.0289902e-06  0.0005560  0.3280206]
```

Comparando con los resultados previos, vemos que hay convergencia en la quinta cifra. Para otras orientaciones y otras redes podría requerirse el uso de más vectores recíprocos, pero siempre es un número muy modesto.

4. Polarización superficial

Con esto, ya podemos calcular las modificaciones superficiales a la polarización. Para esto, modificamos nuestra lista de parámetros para poder dar una respuesta dieléctrica y elegir una orientación.

```
# name: parameters2
my ($a,$b); # 2D basis.
my $c;  # 3D Separation between sites in nearby planes
my $pqmax;   # index of largest reciprocal vector
my $dmax;   # index of largest distance to calculate
my $N; # seminumber of planes of film
my $direction; # cartesian direction, x y or z
my $epsilon; # dielectric function
my $digits=7; # number of decimal digits to print
my $options=q(
    'a=s'=>\$a, # 2D basis vector x,y
    'b=s'=>\$b, # 2D basis vector x,y
    'c=s'=>\$c, # 3D separation x,y,z
    'pqmax=i'=>\$pqmax,  # index of largest reciprocal vector
    'dmax=i'=>\$dmax,  # index of largest distance to calculate
    'N=i'=>\$N, # seminumber of planes of film
    'dir=s'=>\$direction, # cartesian direction, x y or z
    'epsilon=s'=>\$epsilon, # dielectric function e',e''
    'digits=i'=>\$digits, # number of decimal digits to print
    );
my %options=(eval $options);
die "Bad option definition: $@" if $@;
GetOptions(%options) or usage $options, "Bad options";
usage $options, "Undefined parameters"
      unless lu_all {defined $_}
                    ($a, $b, $c, $pqmax, $dmax, $N, $direction, $epsilon);
usage $options, "Vectors should be comma separated list of numbers"
      unless lu_all {looks_like_number $_} map {split ','} ($a, $b, $c);
($a,$b,$c) = map {pdl(split ',', $_)} ($a, $b, $c); #convert from strings to vectors
usage $options, "Basis vectors should be 2D"
      unless lu_all {$_->dims==1 and $_->dim(0)==2} ($a, $b);
usage $options, "c should be 3D" unless $c->dims==1 and $c->dim(0)==3;
usage $options, "Third component of c should be positive" unless $c->((2)) > 0;
usage $options, "pqmax should be positive" unless $pqmax > 0;
```

```
usage $options, "dmax should be positive" unless $dmax > 0;
usage $options, "dir should be x, y or z" unless $direction=~m{^[xyzXYZ]$};
my %index_from_direction=(x=>0, y=>1, z=>2);
my $dir_i=$index_from_direction{lc $direction};
usage $options, "N should be positive" unless $N > 0;
usage $options, "epsilon should be two comma separated numbers eps', eps''"
    unless lu_all {looks_like_number $_} split ',', $epsilon;
$epsilon=[split ',', $epsilon];
$epsilon=$epsilon->[0]+i()*$epsilon->[1];
```

Dado un campo externo normalizado $\boldsymbol{E}^{\text{ex}}$, podemos obtener la polarización de bulto $\boldsymbol{P}_B$ y el dipolo de bulto $\boldsymbol{p}_B$.

```
# name: pB
my $PB=$dir_i==2?(1-1/$epsilon)/(4*PI):($epsilon-1)/(4*PI);
my $pB=$PB/$density; # dipole moment
```

Usando Claussius Mossoti $(\epsilon-1)/(\epsilon+2) = (4\pi/3)n\alpha$ también podemos obtener la polarizabilidad α,

```
# name: alpha
my $alpha=3/(4*PI*$density)*($epsilon-1)/($epsilon+2);
```

La ecuación a resolver es

$$\boldsymbol{p}_n = \alpha\left(\boldsymbol{E}^{\text{ex}} + \sum_{\text{dentro}} \boldsymbol{T}_{nm}\cdot\boldsymbol{p}_m\right). \tag{17}$$

La ecuación en el bulto es

$$\boldsymbol{p}_B = \alpha\left(\boldsymbol{E}^{\text{ex}} + \sum_{m=-\infty}^{\infty} \boldsymbol{T}_{nm}\cdot\boldsymbol{p}_B\right). \tag{18}$$

Restando,

$$\begin{aligned}\Delta\boldsymbol{p}_n &= \alpha\left(\sum_{m\,\text{dentro}} \boldsymbol{T}_{nm}\cdot\boldsymbol{p}_m - \sum_{m=-\infty}^{\infty} \boldsymbol{T}_{nm}\cdot\boldsymbol{p}_B\right)\\ &= \alpha\left(\sum_{m\,\text{dentro}} \boldsymbol{T}_{nm}\cdot\Delta\boldsymbol{p}_m - \sum_{m\,\text{fuera}} T_{nm}\cdot\boldsymbol{p}_B\right),\end{aligned} \tag{19}$$

que reescribimos como

$$\sum_{m\,\text{dentro}} (\delta_{nm}\mathbf{1} - \alpha\boldsymbol{T}_{nm})\cdot\Delta\boldsymbol{p}_m = -\alpha\boldsymbol{M}_n\cdot\boldsymbol{p}_B. \tag{20}$$

donde $\mathbf{1}$ es el tensor identidad y $\boldsymbol{M}_n \equiv \sum_{m\,\text{fuera}} \boldsymbol{T}_{nm}$ es la suma de las interacciones con los planos ausentes.

Construimos entonces rutinas para resolver sistemas lineales de ecuaciones complejas, basadas en rutinas estandard reales. Hacemos una descomposición LU y la usamos para resolver la ecuación.

```
# name: lu
sub solve {
    my ($Matrix, $rhs)=@_;
    #Convert complex equation to real equation
    my ($Mr, $Mi)=($Matrix->re, $Matrix->im);
    my $N=$Mr->dim(0);
    my $real_M=pdl($Mr,-$Mi, $Mi, $Mr)->reshape($N,$N,2,2)
               ->mv(2,1)->reshape(2*$N,2*$N); #assumed no extra dims.
    my $real_rhs=append($rhs->re, $rhs->im)->dummy(1); #make row vector
    my ($lu,$perm,$par) = $real_M->copy->lu_decomp;
    my $real_sol=lu_backsub($lu, $perm, $real_rhs->copy); #returns row
    my $sol=$real_sol->(0:$N-1,(0))+i()*$real_sol->($N:2*$N-1,(0));
    $sol; #ordinary complex vector
}
```

Finalmente, usamos estas rutinas para resolver el sistema lineal de ecuaciones.

```
# name: Dp
my $identity=$T_nm->(($dir_i),($dir_i))->zeroes;
$identity->diagonal(0,1)++;
my $Dp=solve($identity-$alpha*$T_nm->(($dir_i),($dir_i)),
             -$alpha*$M_n->(($dir_i),($dir_i))*$pB);
```

Armamos el programa juntando los fragmentos,

```
# name Delta p
<<init>>
<<usage>>
<<cinner>>
<<trunc>>
<<lu>>
<<parameters2>>
<<area>>
<<reciprocal>>
<<reciprocal-decay>>
```

```
  <<T+->>
  <<interactions>>
  <<selfinteraction>>
  <<interactions-film>>
  <<missing>>
  <<pB>>
  <<alpha>>
  <<Dp>>
  say trunc $digits, $Dp;
```

Probémoslo con un dieléctrico no disipativo.

```
./Delta_p.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2 -N 5 -dir x \
             -epsilon 2,0  -digits 4
```

Resultados: [1]

```
[0.0010  -1.2466e-05  1.4262e-07  -1.6256e-09  1.8520e-11  -4.2182e-13
  1.8520e-11  -1.6256e-09  1.4262e-07  -1.2466e-05  0.0010]
```

Ahora, con vacío + disipación

```
./Delta_p.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2 -N 5 -dir x \
             -epsilon 1,1 -digits 4
```

Resultados:

```
[-0.0010-2.7054e-05i  -2.4601e-06+1.3453e-05i  1.7311e-07+5.9471e-08i
  1.1283e-09-2.1714e-09i  -2.6469e-11-1.9207e-11i  -6.1268e-13+6.2393e-13i
 -2.6469e-11-1.9207e-11i  1.1283e-09-2.1714e-09i  1.7311e-07+5.9471e-08i
 -2.4601e-06+1.3453e-05i  -0.0010-2.7054e-05i]
```

5. Sistema semiinfinito.

Podemos simular un sistema semiinfinito como una película finita, suficientemente ancha y eliminando el término que fuerza al sistema por uno de los lados, i.e. redefiniendo las $\boldsymbol{M}_n$'s. Para ello, modificamos el bloque `missing`.

```
# name: missing1
  my $M_n=zeroes(cdouble,3,3,$N2);
```

[1] En versiones previas de este documento había un error.

```
for my $n(0..$N2-1){
    $M_n->(:,:,($n)).=
        $T_all->(:,:,$N2+$n+1:2*$N2)->mv(-1,0)->sumover;
}
```

Con esto armamos un nuevo programa.

```
# name Delta p1
<<init>>
<<usage>>
<<cinner>>
<<trunc>>
<<lu>>
<<parameters2>>
<<area>>
<<reciprocal>>
<<reciprocal-decay>>
<<T+->>
<<interactions>>
<<selfinteraction>>
<<interactions-film>>
<<missing1>>
<<pB>>
<<alpha>>
<<Dp>>
say trunc $digits, $Dp;
```

Le aplicamos las mismas pruebas que arriba.

```
./Delta_p1.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2 -N 5 -dir x \
              -epsilon 2,0 -digits 4
```

Resultados:

```
[0.0010  -1.2466e-05  1.4262e-07  -1.6256e-09  1.8517e-11  -2.1091e-13
  2.4021e-15  -2.7358e-17  3.1159e-19  -3.5489e-21  4.0413e-23]
```

```
./Delta_p1.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2 -N 5 -dir x \
              -epsilon 1,1 -digits 4
```

Resultados:

```
[-0.0010-2.7054e-05i  -2.4601e-06+1.3453e-05i  1.7311e-07+5.9471e-08i
  1.1283e-09-2.1714e-09i  -2.6473e-11-1.9211e-11i  -3.0634e-13+3.1196e-13i
 3.5181e-15+4.6669e-15i  6.8655e-17-3.7276e-17i  -3.5698e-19-9.8135e-19i
 -1.3679e-20+2.7779e-21i  9.6595e-24+1.8636e-22i]
```

Comparando con los resultados de la sección anterior vemos que Δp es esencialmente idéntica al caso anterior cerca de la superficie mientras que ahora se hace prácticamente cero en el otro extremo.

6. Respuesta superficial

Ahora podemos promediar el exceso de polarización sobre la celda unitaria e integrarla sobre la coordenada normal para obtener la corriente superficial

$$\boldsymbol{j}_s = -i\omega \frac{1}{A} \int d^3r \, \Delta \boldsymbol{P} = -i \frac{\omega}{A} \sum_n \Delta p_n. \tag{21}$$

De aquí podemos identificar las conductividades superficiales,

$$\langle\langle \Delta\sigma^{xx} \rangle\rangle = -i \frac{\omega}{A} \sum_n \Delta p_n^x, \tag{22}$$

$$\langle\langle \Delta\sigma^{yy} \rangle\rangle = -i \frac{\omega}{A} \sum_n \Delta p_n^y, \tag{23}$$

$$\langle\langle \Delta s^{zz} \rangle\rangle = -i \frac{\omega}{A} \sum_n \Delta p_n^z, \tag{24}$$

$$\tag{25}$$

donde A es el área de la celda unitaria 2D, supusimos que x, y, y z son direcciones principales y que usamos campos normalizados $E^i = 1$ al calcular Δp_n^i, $i = x, y$ y $D^z = 1$ al calcular Δp_n^z.

Con estas funciones respuesta puedo calcular la impedancia superficial

$$Z_s = \frac{Z_s^0}{1 + 4\pi Z_s^0 \frac{\langle\langle \Delta\sigma \rangle\rangle}{c}}, \tag{26}$$

$$Z_p = \frac{Z_p^0 + 4\pi \frac{Q^2 c^2}{\omega^2} \frac{\langle\langle \Delta s \rangle\rangle}{c}}{1 + 4\pi Z_p^0 \frac{\langle\langle \Delta\sigma \rangle\rangle}{c}}, \tag{27}$$

donde elegimos las componentes apropiadas de acuerdo a la polarización y a la orientación del plano de incidencia.

Por motivos prácticos tenemos que enfrentar ahora el problema de las unidades. En general, $[P] = [E]$, $[j_s] = [\omega][dz][E] = [c][E]$, por lo que $[\langle\langle\Delta\sigma\rangle\rangle = [\langle\langle\Delta s\rangle\rangle = [c]$ y los cocientes $\langle\langle\Delta\sigma\rangle\rangle / c$ y $\langle\langle\Delta s\rangle\rangle / c$ son adimensionales. Al normalizar los campos a 1 y desproveerlos de unidades, los dipolos calculados arriba tendrían unidades de volumen en lugar de carga por distancia, puesto que la polarizabilidad tiene unidades de volumen. Al dividirlos entre A adquirirían unidades de distancia y al multiplicarlos por ω llegaríamos a las esperadas unidades de velocidad. Es común expresar la frecuencia en términos de la energía $\hbar\omega$ en unidades de eV. Por otro lado, es cómodo expresar los vectores de la base cristalina $\boldsymbol{a}$, $\boldsymbol{b}$ y $\boldsymbol{c}$ en unidades del parámetro de red del sistema. Entonces, la velocidad de la luz arriba c deberíamos expresarla en unidades consistentes, por ejemplo, eV $\times$ parámetro de red. Para ello usamos el factor de conversión $\hbar c = 197{,}3\text{eV nm}$ e leemos el parámetro de red en nanómetros de la línea de comandos.

Modificamos entonces una vez más nuestra lista de parámetros, añadiendo el factor de corrección, el parámetro de red y la frecuencia, quitando una dirección específica para iterar sobre todas.

```
# name: parameters3
use constant hbar_c=>197.3; # eV nm
my ($a,$b); # 2D basis.
my $c;  # 3D Separation between sites in nearby planes
my $pqmax;   # index of largest reciprocal vector
my $dmax;   # index of largest distance to calculate
my $N; # seminumber of planes of film
my $epsilon; # dielectric function
my $lattice; #lattice parameter in nm
my $hbar_w; # frequency in eV
my $digits=7; # number of decimal digits to print
my $options=q(
    'a=s'=>\$a, # 2D basis vector x,y
    'b=s'=>\$b, # 2D basis vector x,y
    'c=s'=>\$c, # 3D separation x,y,z
    'pqmax=i'=>\$pqmax,  # index of largest reciprocal vector
    'dmax=i'=>\$dmax,  # index of largest distance to calculate
    'N=i'=>\$N, # seminumber of planes of film
    'epsilon=s'=>\$epsilon, # dielectric function e',e''
    'lattice=f'=>\$lattice, #lattice parameter in nm
    'frequency=f'=>\$hbar_w, # frequency in eV
    'digits=i'=>\$digits, # number of decimal digits to print
```

```
    );
  my %options=(eval $options);
  die "Bad option definition: $@" if $@;
  GetOptions(%options) or usage $options, "Bad options";
  usage $options, "Undefined parameters"
        unless lu_all {defined $_}
                      ($a, $b, $c, $pqmax, $dmax, $N, $epsilon,
                       $lattice, $hbar_w);
  usage $options, "Vectors should be comma separated list of numbers"
        unless lu_all {looks_like_number $_} map {split ','} ($a, $b, $c);
  #convert strings->vectors
  ($a,$b,$c) = map {pdl(split ',', $_)} ($a, $b, $c);
  usage $options, "Basis vectors should be 2D"
        unless lu_all {$_->dims==1 and $_->dim(0)==2} ($a, $b);
  usage $options, "c should be 3D" unless $c->dims==1 and $c->dim(0)==3;
  usage $options, "Third component of c should be positive"
      unless $c->((2)) > 0;
  usage $options, "pqmax should be positive" unless $pqmax > 0;
  usage $options, "dmax should be positive" unless $dmax > 0;
  usage $options, "N should be positive" unless $N > 0;
  usage $options,
      "epsilon should be two comma separated numbers eps', eps''"
      unless lu_all {looks_like_number $_} split ',', $epsilon;
  $epsilon=[split ',', $epsilon];
  $epsilon=$epsilon->[0]+i()*$epsilon->[1];
```

Ahora calculamos las tres conductividades normalizadas,

```
# name: sigmas
    my @sigma= map {
    my $PB=$_==2?(1-1/$epsilon)/(4*PI):($epsilon-1)/(4*PI);
    my $pB=$PB/$density; # dipole moment
    my $identity=$T_nm->(($_),($_))->zeroes;
    $identity->diagonal(0,1)++;
    my $Dp=solve($identity-$alpha*$T_nm->(($_),($_)),
 -$alpha*$M_n->(($_),($_))*$pB);
    -i()*$hbar_w*$lattice*$Dp->sumover/(hbar_c*$A);
} (0..2);
```

Con esto armamos un nuevo programa.

```
  # name sigma
```

```
  <<init>>
  <<usage>>
  <<cinner>>
  <<trunc>>
  <<lu>>
  <<parameters3>>
  <<area>>
  <<reciprocal>>
  <<reciprocal-decay>>
  <<T+->>
  <<interactions>>
  <<selfinteraction>>
  <<interactions-film>>
  <<missing1>>
  <<alpha>>
  <<sigmas>>
  my @dirs_from_index=qw(xx yy zz);
  say "$dirs_from_index[$_]: $sigma[$_]" for (0..2);
```

Ahora lo corremos para un aislante en una red cúbica simple.

```
./sigma.pl -a 1,0 -b 0,1 -c 0,0,1 -pqmax 2 -dmax 2\
            -N 5 -epsilon 2,0 -lattice .4 -frequency 10 -digits 4
```

Resultados:

```
xx: -2.13738209040293e-05i
yy: -2.13738209040293e-05i
zz: 1.04118689039735e-05i
```

7. Reflectancia diferencial

Tenemos ya prácticamente todas las herramientas para calcular los cambios en la reflectancia. Para ello, en lugar de leer un valor de ϵ desde la línea de comandos, leeremos el nombre de un archivo con una tabla de valores de frecuencia, ϵ' y ϵ'' para, y para cada valor obtendremos la impedancia superficial, la conductividad superficial y la reflectancia. A la lista de parámetros hay que añadir el nombre del archivo, el ángulo de incidencia y el nombre del archivo de salida.

```
# name: parameters4
```

```
use constant hbar_c=>197.3; # eV nm
my ($a,$b); # 2D basis.
my $c;  # 3D Separation between sites in nearby planes
my $pqmax;   # index of largest reciprocal vector
my $dmax;   # index of largest distance to calculate
my $N; # seminumber of planes of film
my $lattice; #lattice parameter in nm
my $angle; #incidence angle (degrees)
my $ifilename;  # name of input file: hnu eps' eps''
my $title;  # title of the output plot
my $ofilename;  # name of output plot (png)
my $digits=7; # number of decimal digits to print
my $options=q(
'a=s'=>\$a, # 2D basis vector x,y
       'b=s'=>\$b, # 2D basis vector x,y
       'c=s'=>\$c, # 3D separation x,y,z
       'pqmax=i'=>\$pqmax,  # index of largest reciprocal vector
       'dmax=i'=>\$dmax,  # index of largest distance to calculate
       'N=i'=>\$N, # seminumber of planes of film
       'lattice=f'=>\$lattice, #lattice parameter in nm
'angle=f'=>\$angle, #incidence angle (degrees)
       'ifilename=s'=>\$ifilename, # name of input file: hnu eps' eps''
'ofilename=s'=>\$ofilename, # name of output plot (png)
        'title=s'=>\$title,  # title of the output plot
       'digits=i'=>\$digits, # number of decimal digits to print
      );
my %options=(eval $options);
die "Bad option definition: $@" if $@;
GetOptions(%options) or usage $options, "Bad options";
usage $options, "Undefined parameters"
    unless lu_all {defined $_}
($a, $b, $c, $pqmax, $dmax, $N, $lattice, $angle, $ifilename,
     $ofilename, $title);
usage $options, "Vectors should be comma separated list of numbers"
    unless lu_all {looks_like_number $_} map {split ','} ($a, $b, $c);
#convert strings->vectors
($a,$b,$c) = map {pdl(split ',', $_)} ($a, $b, $c);
usage $options, "Basis vectors should be 2D"
    unless lu_all {$_->dims==1 and $_->dim(0)==2} ($a, $b);
usage $options, "c should be 3D" unless $c->dims==1 and $c->dim(0)==3;
```

```
usage $options, "Third component of c should be positive"
    unless $c->((2)) > 0;
usage $options, "pqmax should be positive" unless $pqmax > 0;
usage $options, "dmax should be positive" unless $dmax > 0;
usage $options, "N should be positive" unless $N > 0;
usage $options, "Angle should be between 0 and 90"
    unless $angle >=0 and $angle < 90;
my $angler=$angle*PI/180; # angle in radians
usage $options, "ifilename should be readable" unless -r $ifilename;
# frequency in eV, real, imag and full dielectric function
my ($hbar_ws, $epsilons1, $epsilons2)=rcols $ifilename;
usage $options,
    "File should have three columns of space separated numbers"
    unless $hbar_ws->dim(0)==$epsilons1->dim(0) and
    $hbar_ws->dim(0)==$epsilons2->dim(0) and  $hbar_ws->dim(0) > 0;
my $epsilons=$epsilons1+i()*$epsilons2;
```

Aprovechando que en `PDL`, rutinas diseñadas para actuar sobre un escalar se pueden aplicar a arreglos multidimensionales sin modificación, entonces el fragmento `alpha` puede ser usado sin modificación. Sin embargo, los fragmentos `lu` y `sigma` si deben modificarse para acomodar la nueva dimensión (la frecuencia). Aunque no es lo más elegante, podemos evitarlo y reciclar los fragmentos previos si hacemos una iteración. Una solución más elegante hubiera sido preveer el reciclar cada fragmento construyéndolo como módulos a usar o como subrutinas con argumentos en lugar de ser fragmentos a incorporar en un *código* lineal (de *espageti*) y comunicarnos mediante el nombre de las variables. Es el precio a pagar por armar el código incrementalmente. Dentro cada iteración podemos también calculamos la impedancia superficial y la reflectancia.

Primero calculo la impedancia superficial no perturbada, la impedancia perturbada, la amplitud de reflexión y la reflectancia.

```
# name DR/R
my $q=1+0*i(); # normalize to free wavevector
my $Q=$q*sin($angler);
my ($kv, $km)= map {mysqrt($_*$q**2-$Q**2)} (1, $epsilon);
my ($Zsv, $Zpv)= ($q/$kv, $kv/$q);
my ($Zsm0, $Zpm0)= ($q/$km, $km/($q*$epsilon));
my @Zsm=map {$Zsm0/(1+4*PI*$Zsm0*$_)} @sigma[0,1];
my @Zpm=map {($Zpm0+4*PI*$Q**2/$q**2*$sigma[2])/(1+4*PI*$Zpm0*$_)}
            @sigma[0,1];
```

```
#perturbed and non perturbed
my ($rs0,@rs)=map {($_-$Zsv)/($_+$Zsv)} $Zsm0, @Zsm;
my ($rp0,@rp)=map {($Zpv-$_)/($Zpv+$_)} $Zpm0, @Zpm;
my ($Rs0,@Rs)=map {$_->abs**2} ($rs0,@rs);
my ($Rp0,@Rp)=map {$_->abs**2} ($rp0,@rp);
my @DRRs=map {($Rs[$_]-$Rs0)/$Rs0} (0,1); #differential
my @DRRp=map {($Rp[$_]-$Rp0)/$Rp0} (0,1);
```

El cálculo de la impedancia requiere una nueva rutina de utilería `mysqrt`; como la raiz cuadrada tiene dos ramas, debo escoger aquella con parte imaginaria no negativa, i.e., colocando el corte ramal justo abajo del eje positivo.

```
# name mysqrt
sub mysqrt {
    my $x2=shift @_;
    my $x=sqrt($x2);
    $x=-$x if $x->im <0;
    return $x;
}
```

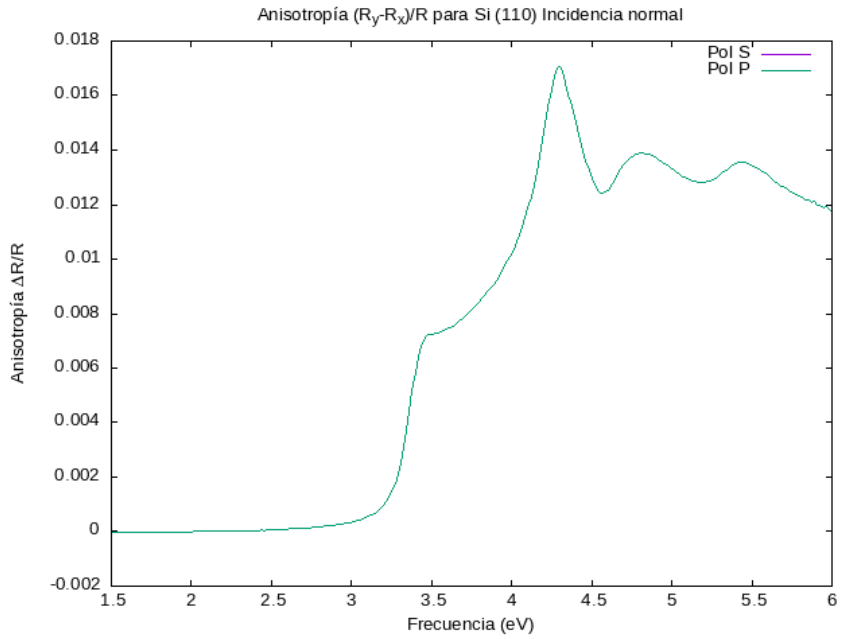
```
# name DeltaR
<<init>>
<<usage>>
<<cinner>>
<<trunc>>
<<mysqrt>>
<<lu>>
<<parameters4>>
<<area>>
<<reciprocal>>
<<reciprocal-decay>>
<<T+->>
<<interactions>>
<<selfinteraction>>
<<interactions-film>>
<<missing1>>
use PDL::Graphics::Gnuplot;
my @results; # accumulate results
for my $r(0..$hbar_ws->dim(0)-1){ # iterate over rows
    my $hbar_w=$hbar_ws->(($r));
    my $epsilon=$epsilons->(($r));
```

```
    <<alpha>>
    <<sigmas>>
    <<DR/R>>
    push @results, [$hbar_w, @DRRs, @DRRp];
}
my ($hbar_w, $DRRsx, $DRRsy, $DRRpx, $DRRpy)=
    map {my $i=$_; pdl(map {$_->[$i]} @results)} (0..4);
my $win=gpwin("png", output=>$ofilename);
$win->plot({title=>$title, xlabel=>'Frecuencia (eV)',
            ylabel=>'Anisotropía {/Symbol D}R/R'},
       {with=>'lines', legend=>'Pol S'}, $hbar_w, $DRRsx-$DRRsy,
       {with=>'lines', legend=>'Pol P'}, $hbar_w, $DRRpx-$DRRpy);
```

Ahora corremos el cálculo para la cara (110) del Si.

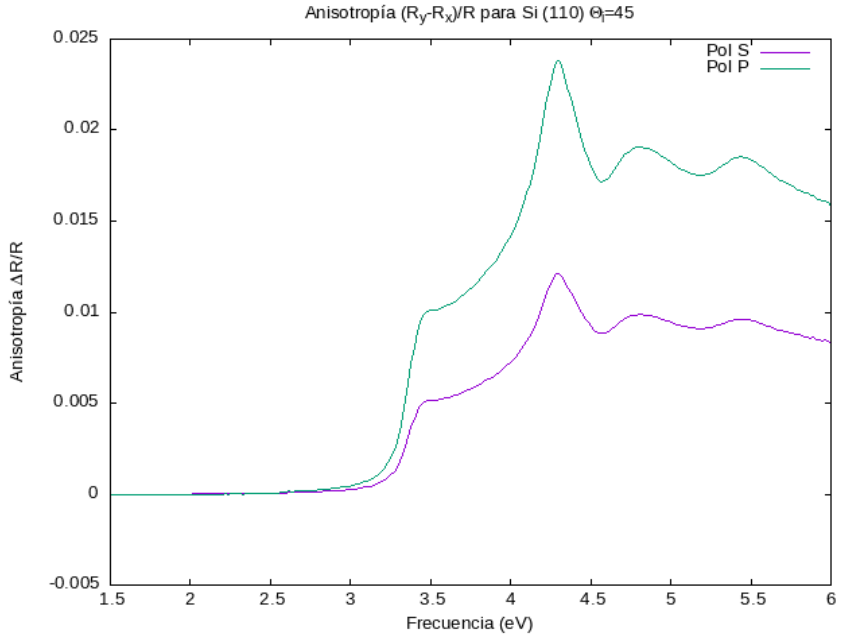
```
./DeltaR.pl -a .7071,0 -b 0,1 -c .3536,.5,.3536 \
     -pqmax 3 -dmax 5 -N 5 -lattice .54 -angle 0 \
     -ifilename epsSiAsp.dat -ofilename si110.png \
     -title 'Anisotropía (R_y-R_x)/R para Si (110) Incidencia normal'
```

Los resultados quedan en la siguiente gráfica



Lo volvemos a correr pero ahora a un ángulo de incidencia $\theta_i = 45°$.

```
./DeltaR.pl -a .7071,0 -b 0,1 -c .3536,.5,.3536 \
     -pqmax 3 -dmax 5 -N 5 -lattice .54 -angle 45 \
     -ifilename epsSiAsp.dat -ofilename si110-45.png \
     -title 'Anisotropía (R_y-R_x)/R para Si (110) {/Symbol Q}_i=45'
```



Curiosamente, aunque la respuesta normal es la misma para ambas polarizaciones, como R_p es menor, la anisotropía relativa es mayor.

8. Conclusiones

Partimos de una idea muy simple, el efecto de campo local no tiene por qué ser idéntico en la vecindad de una superficie que en su interior. Para explorar las consecuencias desarrollamos una técnica para calcular la suma de interacciones dipolares por planos, discutimos cómo calcular la autointeracción de cada plano, y luego hallamos y resolvimos las ecuaciones que cumple el exceso de momento dipolar de cada sitio por hallarse cerca de la superficie y con el mismo, obtuvimos la conductividad superficial, la impedancia superficial y finalmente la reflectancia. Esta resulta tener una pequeña dependencia con la cara cristalina iluminada, y en el caso de caras anisotrópicas, con la polarización de la luz incidente. En este último caso, hay una anisotropía

óptica inducida por la superficie en sistemas nominalmente isotrópicos. Como esta anisotropía se produce en las primeras camadas atómicas, su valor y la estructura de sus espectros dependen fuertemente de la condición de la superficie, y es fuertemente modificada por la presencia de adsorbatos, por la relajación y la reconstrucción superficial, por los cambios a la estructura electrónica, por la presencia de estados de superficie, por su modulación mediante campos externos y por los cambios de composición química. Esto permite que la espectroscopía de anisotropía en la reflectancia, conocida como RAS, sea empleada como una herramienta para *observar* superficies tanto dentro como fuera de cámaras de ultra-alto vacío, e incluso en ambientes químicamente hostiles. RAS se ha convertido en una espectroscopía óptica empleada rutinariamente en la industria semiconductora para monitorear la cinemática y para entender la dinámica del crecimiento epitaxial, así como para observar un sinnúmero de procesos superficiales.

9. Notas

Para preparar estas notas usé el editor extensible *emacs* y su modo *Org mode*, el cual permite escribir con facilidad un texto estructurado (artículo, libro, página web), incluyendo en él fragmentos de código computacional que pueden ensamblarse automáticamente, correrse y desplegar sus resultados en el mismo archivo. Los programas fueron escritos en el lenguaje `Perl` y su extensión numérica `Perl Data Language`.