```
ClearAll["Global`*"]
SetDirectory[NotebookDirectory[]]
```

```
C:\Users\Javier\Desktop\Física\Prácticas\Mathematica\Propagación
  MB 3 niveles\MBCPRFiles\Everything in 4000 Loop Files
```

CUIDADO: GUARDAR EL PROGRAMA EN CARPETA PROPIA CON EL FICHERO LECTURA DATOS

CUIDADO: CUANDO SE ABORTE EL PROGRAMA, CORRER EL CIERRE DEL FICHERO EN LA ULTIMA LINEA O NO SE GUARDARÁN LOS DATOS EN POSTERIORES EJECUCIONES DEL PROGRAMA

```
γ1 = 0; (*Relaxation decays*)
γ2 = 0;
γ3 = 0;
τ12 = 5; (*in ns*)
τ23 = 5; (*in ns*)
ΩP = 20; (*in ns^-1*)
ΩS = 20; (*in ns^-1*)
ΔP = 4; (*in ns^-1 PROBE *)
ΔS = 9; (*in ns^-1 PUMP *)
```

```
ti = 0; (*in ns*)
tf = 200; (*in ns*)
Δt = 0; (*in ns*)
ℏ = 1 × 10^-25; (*J·ns*)
```

```
N0 = 1 × 10^4; (*Sacada de la ecuacion de los gases ideales,
igual para todos los gases*)
           (*Densidad atómica 137 (g/mol) ----BARIO---- in part/μm^3*)
μ12 = 8 * 3.33 * 10^-24;
(*momento dipolar electrico------Bario----8 Debye (Debye=3.33 10^-24 C·μm) ----*)
μ23 = 0.2 * 3.33 * 10^-24; (*momento dipolar electrico----Bario----0.2 Debye------*)
c = 3 × 10^5; (*μm/ns*)
λ12 = 553.7 × 10^-3; (*in μm*)
λ23 = 1500.4 × 10^-3; (*in μm*)
ϵ0 = 8.8541 × 10^-18; (*in F/μm*)
k12 = 2 π / λ12; (*in μm^-1*)
k23 = 2 π / λ23; (*in μm^-1*)
ω12 = k12 c; (*in ns^-1*)
ω23 = k23 c; (*in ns^-1*)
α12 = N0 k12 μ12 / ϵ0 ;
α23 = N0 k23 μ23 / ϵ0 ;
Modifα12 = α12 μ12 / ℏ ;
Modifα23 = α23 μ23 / ℏ;
```

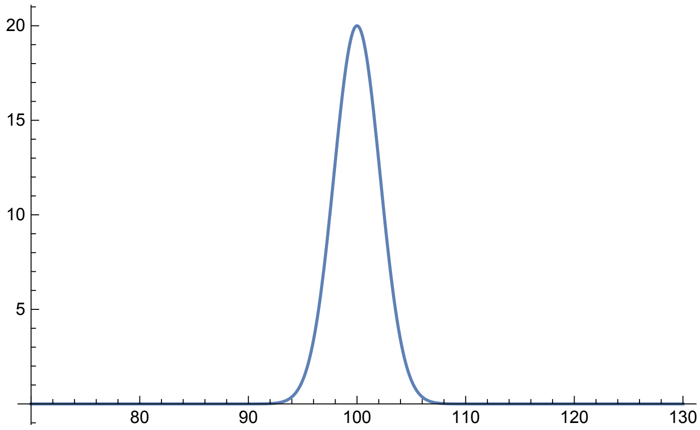
Longitude required to start trespassing a noticeable amount of energy from one side of the pulse to the other for the case in which the stokes laser is turned off ΩP = 20, ΩS = 0, ΔP = 4, ΔS = 9 is LONG=5 $\times 10^{14}$ and the case for which the pump laser ΩP = 0, ΩS = 20, ΔP = 4, ΔS = 9 is LONG=0.2

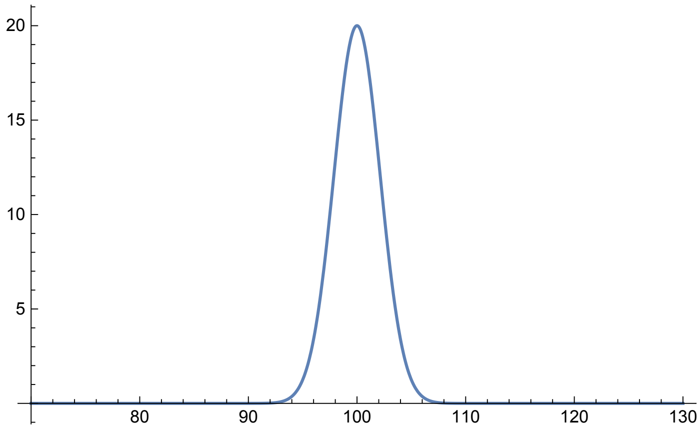
Printed by Wolfram Mathematica Student Edition

```
j = 200;
(*j is the number of loops*)
LONG = 1; (*in μm*)
ξi = LONG / (j);
paso = LONG / (j) (*in μm*)
```

$\frac{1}{200}$

```
(*Now we define the initial Rabi and the electric field shapes
 WARNING:THE Ω2 WONT CHANGE FOR NOW*)
(*f[t_,τ_]=HeavisidePi[t/20-(200)/40];*)
f[t_, τ_] = Exp[- 4 Log[2] ((t - (tf - ti) / 2) / τ)^2];
Ω1 = ΩP * f[t, τ12];
Ω2 = ΩS * f[t + Δt, τ23];
Plot[Ω1, {t, 70, 130}, Exclusions → None, PlotRange → All]
Plot[Ω2, {t, 70, 130}, Exclusions → None, PlotRange → All]
```






```
i = 1;
Doc = 1;
LoopsPerDoc = 4000;
Sol0 = {};
Listatime = {};

(*En el siguiente Do separado guardamos el tiempo para
 calcularlo sólo una vez en vez de una por cada iteración*)
t = ti;
```

```
Do[
  Listatime = Append[Listatime, t];
  t2 = t;
  Clear[t];
  t = t2 + 0.5,
  {401}];

Timing[
 Do[
   SetOptions[OpenWrite[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"]],
    PageWidth → Infinity];
   WriteString[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], "{"];

   If[j - LoopsPerDoc * Doc > 0,
    Do[
      Clear[ρ11,  ρ22, ρ33,  Rρ12, Iρ12, Rρ13, Iρ13, Rρ23, Iρ23, HA, ρ, ρrelax,
       coherence, s, r11, r22, r33, SolveRρ12, SolveIρ12, SolveRρ13, SolveIρ13,
       SolveRρ23, SolveIρ23, ξf, ΩPN1, ΩPN2, Lista1Rρ12, Lista1Iρ12, Lista1Rρ13,
       Lista1Iρ13, Lista1Rρ23, Lista1Iρ23, Lista1ΩN1, Lista1ΩN2, FuncRρ12, FuncIρ12,
       FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, FuncΩN1, FuncΩN2, PopuCoheRabi, t];

      (*Matrices for the hamiltonian, coherence and decay rates are*)
      HA[t_] =
       1 / 2 * {{0, Ω1, 0}, {Conjugate[Ω1], 2 ΔP, Ω2}, {0, Conjugate[Ω2], 2 (ΔP - ΔS)}};
      ρ[t_] = {{ρ11[t], Rρ12[t] + ⅈ * Iρ12[t], Rρ13[t] + ⅈ * Iρ13[t]},
         {Rρ12[t] - ⅈ * Iρ12[t], ρ22[t], Rρ23[t] + ⅈ * Iρ23[t]},
         {Rρ13[t] - ⅈ * Iρ13[t], Rρ23[t] - ⅈ * Iρ23[t], ρ33[t]}};
      ρrelax[t_] = 1 / 2 * {{(γ1 + γ1) * ρ11[t], (γ1 + γ2) * (Rρ12[t] + ⅈ * Iρ12[t]),
           (γ1 + γ3) * (Rρ13[t] + ⅈ * Iρ13[t])},
          {(γ2 + γ1) * (Rρ12[t] - ⅈ * Iρ12[t]), (γ2 + γ2) * ρ22[t],
           (γ2 + γ3) * (Rρ23[t] + ⅈ * Iρ23[t])}, {(γ3 + γ1) * (Rρ13[t] - ⅈ * Iρ13[t]),
           (γ3 + γ2) * (Rρ23[t] - ⅈ * Iρ23[t]), (γ3 + γ3) * ρ33[t]}};

      coherence = -ⅈ * (HA[t].ρ[t] - ρ[t].HA[t]) - ρrelax[t];

      s = NDSolve[{
          ρ11'[t] == coherence[[1, 1]],
          ρ22'[t] == coherence[[2, 2]],
          ρ33'[t] == coherence[[3, 3]],
          Rρ12'[t] == Re[coherence[[1, 2]]],
          Iρ12'[t] == Im[coherence[[1, 2]]],
          Rρ13'[t] == Re[coherence[[1, 3]]],
          Iρ13'[t] == Im[coherence[[1, 3]]],
          Rρ23'[t] == Re[coherence[[2, 3]]],
          Iρ23'[t] == Im[coherence[[2, 3]]],
          ρ11[ti] == 1, ρ22[ti] == 0, ρ33[ti] == 0, Rρ12[ti] == 0,
          Iρ12[ti] == 0, Rρ13[ti] == 0, Iρ13[ti] == 0, Rρ23[ti] == 0, Iρ23[ti] == 0},
         {ρ11[t], ρ22[t], ρ33[t], Rρ12[t], Iρ12[t], Rρ13[t], Iρ13[t], Rρ23[t],
          Iρ23[t]}, {t, ti, tf}, MaxStepSize → 0.1, MaxSteps → Infinity, "Method" →
          {"EquationSimplification" → {Automatic, "TimeConstraint" → Infinity}}];
      r11 = Re[ρ11[t] /. s[[1, 1]]];
      r22 = Re[ρ22[t] /. s[[1, 2]]];
      r33 = Re[ρ33[t] /. s[[1, 3]]];
      SolveRρ12 = Re[Rρ12[t] /. s[[1, 4]]];
```

```
SolveIρ12 = Re[Iρ12[t] /. s[[1, 5]]];
SolveRρ13 = Re[Rρ13[t] /. s[[1, 6]]];
SolveIρ13 = Re[Iρ13[t] /. s[[1, 7]]];
SolveRρ23 = Re[Rρ23[t] /. s[[1, 8]]];
SolveIρ23 = Re[Iρ23[t] /. s[[1, 9]]];


(****************************************)
Clear[Funcr11, Funcr22, Funcr33,
 FuncRρ12, FuncIρ12, FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, t];

Funcr11 = r11;
Funcr22 = r22;
Funcr33 = r33;
FuncRρ12 = SolveRρ12;
FuncIρ12 = SolveIρ12;
FuncRρ13 = SolveRρ13;
FuncIρ13 = SolveIρ13;
FuncRρ23 = SolveRρ23;
FuncIρ23 = SolveIρ23;
t = 0;

Lista1r11 = {};
Lista1r22 = {};
Lista1r33 = {};
Lista1Rρ12 = {};
Lista1Iρ12 = {};
Lista1Rρ13 = {};
Lista1Iρ13 = {};
Lista1Rρ23 = {};
Lista1Iρ23 = {};

Lista1ΩN1 = {};
Lista1ΩN2 = {};


PopuCoheRabi = {};

Do[
 Valorr11 = Funcr11;
 Valorr22 = Funcr22;
 Valorr33 = Funcr33;
 ValorRρ12 = FuncRρ12;
 ValorIρ12 = FuncIρ12;
 ValorRρ13 = FuncRρ13;
 ValorIρ13 = FuncIρ13;
 ValorRρ23 = FuncRρ23;
 ValorIρ23 = FuncIρ23;

 Lista1r11 = Append[Lista1r11, Valorr11];
 Lista1r22 = Append[Lista1r22, Valorr22];
 Lista1r33 = Append[Lista1r33, Valorr33];
 Lista1Rρ12 = Append[Lista1Rρ12, ValorRρ12];
 Lista1Iρ12 = Append[Lista1Iρ12, ValorIρ12];
 Lista1Rρ13 = Append[Lista1Rρ13, ValorRρ13];
```

```
 Lista1Iρ13 = Append[Lista1Iρ13, ValorIρ13];
 Lista1Rρ23 = Append[Lista1Rρ23, ValorRρ23];
 Lista1Iρ23 = Append[Lista1Iρ23, ValorIρ23];

 t2 = t;
 Clear[t, Funcr11, Funcr22, Funcr33, FuncRρ12, FuncIρ12,
  FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, Valorr11, Valorr22, Valorr33,
  ValorRρ12, ValorIρ12, ValorRρ13, ValorIρ13, ValorRρ23, ValorIρ23];
 Funcr11 = r11;
 Funcr22 = r22;
 Funcr33 = r33;
 FuncRρ12 = SolveRρ12;
 FuncIρ12 = SolveIρ12;
 FuncRρ13 = SolveRρ13;
 FuncIρ13 = SolveIρ13;
 FuncRρ23 = SolveRρ23;
 FuncIρ23 = SolveIρ23;
 t = t2 + 0.5,
 {401}];

PopuCoheRabi =
 Append[PopuCoheRabi, Partition[Riffle[Listatime, Lista1r11], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1r22], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1r33], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Rρ12], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Iρ12], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Rρ13], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Iρ13], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Rρ23], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Iρ23], 2, 2]];

Clear[t];

(*Calculation of the modified intensity,
electric field and rabi frequency in the chunk*)
g[Ω_, Modifα_, Iρij_] = Ω - Modifα * Iρij * paso;
ΩPN1 = FunctionInterpolation[
   g[Ω1, Modifα12, SolveIρ12], {t, ti, tf}, InterpolationPoints → 2000][t];
ΩPN2 = FunctionInterpolation[g[Ω2, Modifα23, SolveIρ23],
   {t, ti, tf}, InterpolationPoints → 2000][t];

(*Guardado de los valores de cada Intensidad en una serie de matrices*)
t = 0;
FuncΩN1 = ΩPN1;
FuncΩN2 = ΩPN2;
Do[
 Clear[ValorΩN1, ValorΩN2];
```

```
     ValorΩN1 = FuncΩN1;
     ValorΩN2 = FuncΩN2;
     Lista1ΩN1 = Append[Lista1ΩN1, ValorΩN1];
     Lista1ΩN2 = Append[Lista1ΩN2, ValorΩN2];
     t2 = t;
     Clear[t, FuncΩN1, FuncΩN2, ValorΩN1, ValorΩN2];
     FuncΩN1 = ΩPN1;
     FuncΩN2 = ΩPN2;
     t = t2 + 0.5,
     {401}];

    Clear[t];
    FunΩN1 = Riffle[Listatime, Lista1ΩN1];
    FunΩN2 = Riffle[Listatime, Lista1ΩN2];
    MuchFunΩN1 = Partition[FunΩN1, 2, 2];
    MuchFunΩN2 = Partition[FunΩN2, 2, 2];
    PopuCoheRabi = Append[PopuCoheRabi, MuchFunΩN1];
    PopuCoheRabi = Append[PopuCoheRabi, MuchFunΩN2];
    Write[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], PopuCoheRabi];
    WriteString[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], ","];

    Sol0 = Append[Sol0, ξi];

    (*Reset of material chunk we are to study in next loop*)
    ξf = (i + 1) * LONG / (j);
    Clear[ξi, Ω1, Ω2];
    Ω1 = ΩPN1;
    Ω2 = ΩPN2;
    ξi = ξf;
    i = i + 1,
    {LoopsPerDoc}];

  , (*Coma entre condicion cierta o falsa del IF*)

  Do[
    Clear[ρ11, ρ22, ρ33, Rρ12, Iρ12, Rρ13, Iρ13, Rρ23, Iρ23, HA, ρ, ρrelax,
     coherence, s, r11, r22, r33, SolveRρ12, SolveIρ12, SolveRρ13, SolveIρ13,
     SolveRρ23, SolveIρ23, ξf, ΩPN1, ΩPN2, Lista1Rρ12, Lista1Iρ12, Lista1Rρ13,
     Lista1Iρ13, Lista1Rρ23, Lista1Iρ23, Lista1ΩN1, Lista1ΩN2, FuncRρ12, FuncIρ12,
     FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, FuncΩN1, FuncΩN2, PopuCoheRabi, t];

    (*Matrices for the hamiltonian, coherence and decay rates are*)
    HA[t_] =
     1 / 2 * {{0, Ω1, 0}, {Conjugate[Ω1], 2 ΔP, Ω2}, {0, Conjugate[Ω2], 2 (ΔP - ΔS)}};
    ρ[t_] = {{ρ11[t], Rρ12[t] + i * Iρ12[t], Rρ13[t] + i * Iρ13[t]},
       {Rρ12[t] - i * Iρ12[t], ρ22[t], Rρ23[t] + i * Iρ23[t]},
       {Rρ13[t] - i * Iρ13[t], Rρ23[t] - i * Iρ23[t], ρ33[t]}};
    ρrelax[t_] = 1 / 2 * {{(γ1 + γ1) * ρ11[t], (γ1 + γ2) * (Rρ12[t] + i * Iρ12[t]),
         (γ1 + γ3) * (Rρ13[t] + i * Iρ13[t])},
       {(γ2 + γ1) * (Rρ12[t] - i * Iρ12[t]), (γ2 + γ2) * ρ22[t],
         (γ2 + γ3) * (Rρ23[t] + i * Iρ23[t])}, {(γ3 + γ1) * (Rρ13[t] - i * Iρ13[t]),
         (γ3 + γ2) * (Rρ23[t] - i * Iρ23[t]), (γ3 + γ3) * ρ33[t]}};

    coherence = -i * (HA[t].ρ[t] - ρ[t].HA[t]) - ρrelax[t];
```

```
s = NDSolve[{
   ρ11'[t] == coherence[[1, 1]],
   ρ22'[t] == coherence[[2, 2]],
   ρ33'[t] == coherence[[3, 3]],
   Rρ12'[t] == Re[coherence[[1, 2]]],
   Iρ12'[t] == Im[coherence[[1, 2]]],
   Rρ13'[t] == Re[coherence[[1, 3]]],
   Iρ13'[t] == Im[coherence[[1, 3]]],
   Rρ23'[t] == Re[coherence[[2, 3]]],
   Iρ23'[t] == Im[coherence[[2, 3]]],
   ρ11[ti] == 1, ρ22[ti] == 0, ρ33[ti] == 0, Rρ12[ti] == 0,
   Iρ12[ti] == 0, Rρ13[ti] == 0, Iρ13[ti] == 0, Rρ23[ti] == 0, Iρ23[ti] == 0},
  {ρ11[t], ρ22[t], ρ33[t], Rρ12[t], Iρ12[t], Rρ13[t], Iρ13[t], Rρ23[t],
   Iρ23[t]}, {t, ti, tf}, MaxStepSize → 0.1, MaxSteps → Infinity, "Method" →
   {"EquationSimplification" → {Automatic, "TimeConstraint" → Infinity}}];
r11 = Re[ρ11[t] /. s[[1, 1]]];
r22 = Re[ρ22[t] /. s[[1, 2]]];
r33 = Re[ρ33[t] /. s[[1, 3]]];
SolveRρ12 = Re[Rρ12[t] /. s[[1, 4]]];
SolveIρ12 = Re[Iρ12[t] /. s[[1, 5]]];
SolveRρ13 = Re[Rρ13[t] /. s[[1, 6]]];
SolveIρ13 = Re[Iρ13[t] /. s[[1, 7]]];
SolveRρ23 = Re[Rρ23[t] /. s[[1, 8]]];
SolveIρ23 = Re[Iρ23[t] /. s[[1, 9]]];


(****************************************)
Clear[Funcr11, Funcr22, Funcr33,
 FuncRρ12, FuncIρ12, FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, t];

Funcr11 = r11;
Funcr22 = r22;
Funcr33 = r33;
FuncRρ12 = SolveRρ12;
FuncIρ12 = SolveIρ12;
FuncRρ13 = SolveRρ13;
FuncIρ13 = SolveIρ13;
FuncRρ23 = SolveRρ23;
FuncIρ23 = SolveIρ23;
t = 0;

Lista1r11 = {};
Lista1r22 = {};
Lista1r33 = {};
Lista1Rρ12 = {};
Lista1Iρ12 = {};
Lista1Rρ13 = {};
Lista1Iρ13 = {};
Lista1Rρ23 = {};
Lista1Iρ23 = {};

Lista1ΩN1 = {};
Lista1ΩN2 = {};
```

```
PopuCoheRabi = {};

Do[
 Valorr11 = Funcr11;
 Valorr22 = Funcr22;
 Valorr33 = Funcr33;
 ValorRρ12 = FuncRρ12;
 ValorIρ12 = FuncIρ12;
 ValorRρ13 = FuncRρ13;
 ValorIρ13 = FuncIρ13;
 ValorRρ23 = FuncRρ23;
 ValorIρ23 = FuncIρ23;

 Lista1r11 = Append[Lista1r11, Valorr11];
 Lista1r22 = Append[Lista1r22, Valorr22];
 Lista1r33 = Append[Lista1r33, Valorr33];
 Lista1Rρ12 = Append[Lista1Rρ12, ValorRρ12];
 Lista1Iρ12 = Append[Lista1Iρ12, ValorIρ12];
 Lista1Rρ13 = Append[Lista1Rρ13, ValorRρ13];
 Lista1Iρ13 = Append[Lista1Iρ13, ValorIρ13];
 Lista1Rρ23 = Append[Lista1Rρ23, ValorRρ23];
 Lista1Iρ23 = Append[Lista1Iρ23, ValorIρ23];

 t2 = t;
 Clear[t, Funcr11, Funcr22, Funcr33, FuncRρ12, FuncIρ12,
  FuncRρ13, FuncIρ13, FuncRρ23, FuncIρ23, Valorr11, Valorr22, Valorr33,
  ValorRρ12, ValorIρ12, ValorRρ13, ValorIρ13, ValorRρ23, ValorIρ23];
 Funcr11 = r11;
 Funcr22 = r22;
 Funcr33 = r33;
 FuncRρ12 = SolveRρ12;
 FuncIρ12 = SolveIρ12;
 FuncRρ13 = SolveRρ13;
 FuncIρ13 = SolveIρ13;
 FuncRρ23 = SolveRρ23;
 FuncIρ23 = SolveIρ23;
 t = t2 + 0.5,
 {401}];

PopuCoheRabi =
 Append[PopuCoheRabi, Partition[Riffle[Listatime, Lista1r11], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1r22], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1r33], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Rρ12], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Iρ12], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Rρ13], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
   Riffle[Listatime, Lista1Iρ13], 2, 2]];
PopuCoheRabi = Append[PopuCoheRabi, Partition[
```

```
      Riffle[Listatime, Lista1Rρ23], 2, 2]];
   PopuCoheRabi = Append[PopuCoheRabi, Partition[
      Riffle[Listatime, Lista1Iρ23], 2, 2]];

   Clear[t];

   (*Calculation of the modified intensity,
   electric field and rabi frequency in the chunk*)
   g[Ω_, Modifα_, Iρij_] = Ω - Modifα * Iρij * paso;
   ΩPN1 = FunctionInterpolation[
      g[Ω1, Modifα12, SolveIρ12], {t, ti, tf}, InterpolationPoints → 2000][t];
   ΩPN2 = FunctionInterpolation[g[Ω2, Modifα23, SolveIρ23],
      {t, ti, tf}, InterpolationPoints → 2000][t];

   (*Guardado de los valores de cada Intensidad en una serie de matrices*)
   t = 0;
   FuncΩN1 = ΩPN1;
   FuncΩN2 = ΩPN2;
   Do[
    Clear[ValorΩN1, ValorΩN2];
    ValorΩN1 = FuncΩN1;
    ValorΩN2 = FuncΩN2;
    Lista1ΩN1 = Append[Lista1ΩN1, ValorΩN1];
    Lista1ΩN2 = Append[Lista1ΩN2, ValorΩN2];
    t2 = t;
    Clear[t, FuncΩN1, FuncΩN2, ValorΩN1, ValorΩN2];
    FuncΩN1 = ΩPN1;
    FuncΩN2 = ΩPN2;
    t = t2 + 0.5,
    {401}];

   Clear[t];
   FunΩN1 = Riffle[Listatime, Lista1ΩN1];
   FunΩN2 = Riffle[Listatime, Lista1ΩN2];
   MuchFunΩN1 = Partition[FunΩN1, 2, 2];
   MuchFunΩN2 = Partition[FunΩN2, 2, 2];
   PopuCoheRabi = Append[PopuCoheRabi, MuchFunΩN1];
   PopuCoheRabi = Append[PopuCoheRabi, MuchFunΩN2];
   Write[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], PopuCoheRabi];
   WriteString[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], ","];

   Sol0 = Append[Sol0, ξi];

   (*Reset of material chunk we are to study in next loop*)
   ξf = (i + 1) * LONG / (j);
   Clear[ξi, Ω1, Ω2];
   Ω1 = ΩPN1;
   Ω2 = ΩPN2;
   ξi = ξf;
   i = i + 1,
   {j - LoopsPerDoc * (Doc - 1)}];
];
WriteString[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"], "0 }"];
Close[StringJoin["Popu_Cohe_Rabi_", ToString[Doc], ".txt"]];
Doc++,
```

```
    {IntegerPart[j / LoopsPerDoc] + 1}];]
```

```
{196.281250, Null}
```

```
i
```

```
201
```