Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Grau en Enginyeria de Vehicles Aeroespacials

Title:

Optimization study of dynamic vibration absorbers parameters and distribution for its application on railway tunnels for the reduction of railway-induced vibration

Document content: ANNEXES

Delivery date: 12/06/2015

Author: Víctor Cubría Radío
Director: Robert Arcos Villamarín

Contents

LIST OF FIGURES **4**

LIST OF TABLES **5**

A. CODE EXAMPLES **6**

A.1 Fitness function 6

A.2 DVAParameters.m 8

A.3 H_UnitaryExternalForce_E1.m 10

A.4 Exemple of 'PlotFcns' 13

A.5 Hybrid algorithm (GA and PS) 15

B. INPUT EXAMPLES **17**

C. OUTPUT GRAPHS **22**

D. PLANNING AND SCHEDULING **34**

D.1 Planning and scheduling of the current study 34

D.2 Future planning and scheduling 39

List of Figures

A.1	Schema of an example of DVAParamenters.m, with a distribution of DVAs placed in the tunnel.	9
A.2	Figure A.2: Plots given by the function `'PlotFcns'` for the optimization case of three DVAs. Clockwise the graphs are: `gaplotbestf`, `gaplotbestindiv`, `gaplotgenealogy` and `gaplotdistance`.	14
C.1	Graph of the square of the acceleration against frequency, without DVAs.	23
C.2	Graph of the square of the acceleration against frequency, for one DVA.	24
C.3	Graph of the square of the acceleration against frequency, for two DVAs.	25
C.4	Graph of the square of the acceleration against frequency, for three DVAs.	26
C.5	Graph of the square of the acceleration against frequency, for four DVAs.	27
C.6	Graph of the square of the acceleration against frequency, for five DVAs.	28
C.7	Graph of the square of the acceleration against frequency, for six DVAs.	29
C.8	Graph of the square of the acceleration against frequency, for seven DVAs.	30
C.9	Graph of the square of the acceleration against frequency, for eight DVAs.	31
C.10	Graph of the square of the acceleration against frequency, for nine DVAs.	32
C.11	Graph of the square of the acceleration against frequency, for ten DVAs.	33
D.1	Gantt chart of the current study.	38
D.2	Gantt chart of the future study. It has been considered that it starts on September 18th and finishes on January 1st.	42

List of Tables

A.1	Fitness function for one DVA.	6
A.2	Fitness function for two DVAs.	7
A.3	DVAParameters.m function.	9
A.4	H_UnitaryExternalForce_E1.m function.	10
A.5	Hybrid algorithm.	15
B.1	Evaluators, first part.	18
B.2	Evaluators, second part.	19
B.3	Evaluators, third part.	20
B.4	Evaluators, fourth part.	21
D.1	Interdependences of the tasks for the current study.	36
D.2	Estimated time to develop the tasks of the current study.	37
D.3	Interdependences of the tasks for the future study.	41
D.4	Estimated time to develop the tasks of the future study.	42

A. Code examples

In this section will be presented some examples of the algorithm: the **GA** as well as the **fitness function**, **DVAParameters.m** with a schema of it, **H_Unitary ExternalForce_E1.m**, the **pattern search** formulated to implement the **hybrid algorithm** and finally the program to check the solutions, using **NOMAD**. Furthermore, there are other examples presented to complement the report,

A.1 Fitness function

Two versions of the code will be presented: first, the complete fitness function for the optimization case for one DVA (table A.1) and second, and the complete algorithm for the optimization case for two DVAs (table A.2).

Fitness function for one DVA

```
function [V] =  fitness_1DVA(x)

NodeID = x(1);
DistributionProperties = x(2);
Numberof_LD_DVA = x(3);
K = x(4);
C = 0.01*K;
M = x(5);

DVAset =
DVAParameters(NodeID,{'Periodic'},{DistributionProperties},...
    {Numberof_LD_DVA},{K},{C},{M});

fs = 1:1:100;
S = H_UnitaryExternalForce_E1('H_DVA+Eval_SpaceDomain',DVAset);

l_fs = length(fs);
l_S = length(S);
fs_1 = fs.^2;
fs_2 = reshape(fs_1,[l_fs 1]);
```

```
fs_S = repmat(fs_2,1,l_S,3);
k = 1:1:3;
j = 1:1:l_S;

A(:,j,k) = -fs_S.*S(:,j,k);
B(1,:,k) = sum(A(:,:,k),2);

t = 1:1:l_fs;
C = sqrt(B(1,t,1).^2 + B(1,t,2).^2 + B(1,t,3).^2).^2;

% ISO NORMATIVE

f1 = 10^-0.1;              % [Hz]  high pass
f2 = 100;                  % [Hz]  low pass
f3 = 1/(0.028*2*pi);       % [Hz]  pure frequency weighting
H_h = sqrt(fs.^4./(fs.^4+f1^4));
H_l = sqrt(f2^4./(fs.^4)+f2^4);
H_t = sqrt(f3^2./(f3^2+fs.^2));
H = H_h.*H_l.*H_t;

L =  (abs(C).^2.*abs(H).^2);
J = (1./l_fs^2)*L;
V = sum(J);

end
```

Table A.1: Fitness function for one DVA.

Fitness function for two DVAs

```
function [V] =  fitness_2DVAs(x)

NodeID = {32;x(1)};
DistributionProperties = {1;x(2)};
Numberof_LD_DVA = {9;x(3)};
K = {4.999844665924609e+07;x(4)};
C = {0.01*4.999844665924609e+07;0.01*x(4)};
M = {999.99471613;x(5)};

DVAset = DVAParameters(NodeID,{'Periodic';'Periodic'}, ...
DistributionProperties,Numberof_LD_DVA,K,C,M);
```

```
fs = 1:1:100;
S = H_UnitaryExternalForce_E1('H_DVA+Eval_SpaceDomain',DVAset);

l_fs = length(fs);
l_S = length(S);
fs_1 = fs.^2;
fs_2 = reshape(fs_1,[l_fs 1]);
%fs_2 = fs_1.'
fs_S = repmat(fs_2,1,l_S,3);

k = 1:1:3;
j = 1:1:l_S;
A(:,j,k) = -fs_S.*S(:,j,k);
B(1,:,k) = sum(A(:,:,k),2);

t = 1:1:l_fs;
C = sqrt(B(1,t,1).^2 + B(1,t,2).^2 + B(1,t,3).^2).^2;

% ISO NORMATIVE

f1 = 10^-0.1;              % [Hz]  high pass
f2 = 100;                  % [Hz]  low pass
f3 = 1/(0.028*2*pi);       % [Hz]  pure frequency weighting

H_h = sqrt(fs.^4./(fs.^4+f1^4));
H_l = sqrt(f2^4./(fs.^4)+f2^4);
H_t = sqrt(f3^2./(f3^2+fs.^2));
H = H_h.*H_l.*H_t;

L =  (1./l_fs^2)*(abs(C).^2.*abs(H).^2);
V = sum(L);

end
```

Table A.2: Fitness function for two DVAs.

A.2 DVAParameters.m

In this section, the DVAParameters.m function will be shown (table A.3), as well as an example with an illustrative schema to understand it.

DVAParameters.m

```
function  DVAset = DVAParameters(DVA_Positions, ...
Distribution, DistributionProperties,Numberof_LD_DVA, ...
K,C,M)

DVAset=struct('NodeID',DVA_Positions,'Distribution', ...
Distribution,'DistributionProperties', ...
DistributionProperties,'Numberof_LD_DVA', ...
Numberof_LD_DVA,'K',K,'M',M,'C',C);

DVAset=dataset(NodeID,Distribution, ...
DistributionProperties,Numberof_LD_DVA,K,C,M);
save('DVAset.mat','DVAset')
end
```

Table A.3: DVAParameters.m function.

Figure A.1 is an example of how to represent the variables saved in the DVAset, generated by the function DVAParameters.m. In this example are represented the main variables to understand how it is built the model in the tunnel.

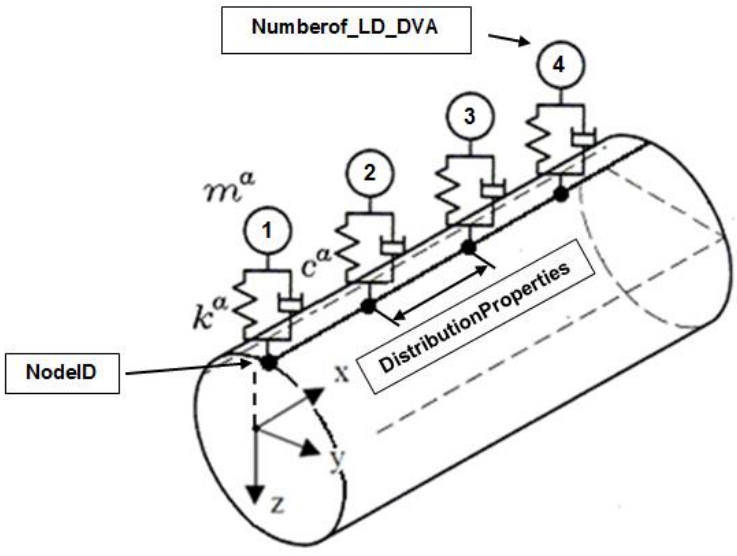

Figure A.1: Schema of an example of DVAParameters.m, with a distribution of DVAs placed in the tunnel.

A.3 H_UnitaryExternalForce_E1.m

In this section, the formulation of the H_UnitaryExternalForce_E1.m is presented, in table A.4.

H_UnitaryExternalForce_E1.m

```
% IMPORTANT: LEAM Coordinates are   ------------>y
%                                     |
%                                     |
%                                     |
%                                     |
%                                     |
%                                    z

function H=H_UnitaryExternalForce_E1(H_DVA_Name,DVAset)

H_DVA_Extractable=matfile(H_DVA_Name);
fs=H_DVA_Extractable.fs;


fs=reshape(fs,length(fs),1);
z=H_DVA_Extractable.z;
Dz=z(2)-z(1);
AllNodes=(length(H_DVA_Extractable.Evaluators))/2;
Numberof_CSD_DVA=length(DVAset);

counter1=0;

for CSD=1:1:Numberof_CSD_DVA
    M=DVAset(CSD).M;K=DVAset(CSD).K;C=DVAset(CSD).C;
    K_Target1=(K+1i*2*pi*fs*C).*(( ...
    -(2*pi*fs).^2)*(M))./((K+1i*2*pi*fs*C)- ...
    (((2*pi*fs).^2)*(M))));
    ID_TargetDVA=(DVAset(CSD).NodeID);
    H_Ext_PiP_AFFT=H_DVA_Extractable.H_DVA(:,:,ID_TargetDVA);
    Numberof_LD_DVA1=DVAset(CSD).Numberof_LD_DVA;

    for LD=1:1:Numberof_LD_DVA1
        counter1=counter1+1;
        Z_TargetDVA=((Numberof_LD_DVA1/2) ...
```

```
        -LD+1)*(DVAset(CSD).DistributionProperties);
        Z_TargetDVA2(counter1)=Z_TargetDVA;
        H_Ext_PiP_AFFT_AIP=interp2( ...
        z,fs,H_Ext_PiP_AFFT,Z_TargetDVA,fs);
        H_Ext(counter1,1,:)=permute(H_Ext_PiP_AFFT_AIP,[2 3 1]);
        counter2=0;

        for m=1:1:Numberof_CSD_DVA
            M=DVAset(m).M;K=DVAset(m).K;C=DVAset(m).C;
            K_DVA1=(K+1i*2*pi*fs*C).*((( ...
            -(2*pi*fs).^2)*(M))./((K+1i*2*pi*fs*C)- ...
            (((2*pi*fs).^2)*(M))));
            ID_Force=(DVAset(m).NodeID);

            if ID_TargetDVA-ID_Force>0
                H_Int_PiP_AFFT=H_DVA_Extractable.H_DVA(:,:,( ...
                ID_TargetDVA-ID_Force));
            else

            H_Int_PiP_AFFT=H_DVA_Extractable.H_DVA(:,:, ...
            (AllNodes+(ID_TargetDVA-ID_Force)));

            end

            Numberof_LD_DVA=DVAset(m).Numberof_LD_DVA;
            for n=1:1:Numberof_LD_DVA
               counter2=counter2+1;
               Z_DVA_Alongm=((Numberof_LD_DVA/2)- ...
               n+1)*(DVAset(m).DistributionProperties);
               H_Int_PiP_AFFT_AIP(:,counter2)=interp2 ...
               (z,fs,H_Int_PiP_AFFT,(Z_TargetDVA- ...
               Z_DVA_Alongm),fs);
               K_DVA(:,counter2)=K_DVA1;
            end
        end

        HK(counter1,:,:)=permute(H_Int_PiP_AFFT_AIP.* ...
        K_DVA,[3 2 1]);
        K_Target(:,counter1)=K_Target1;
    end
end

I=repmat(eye(size(HK,2)),1,1,length(fs));
K_Target=permute(K_Target,[2 3 1]);
```

```
A=I-HK;
Ainv=permute(MultiSolver(A,eye(size(HK,2))),[1 3 2]);
B=H_Ext;
DVA_Deflections=mtimesx(Ainv,B);
DVA_Forces=DVA_Deflections.*K_Target;
DVA_Forces=permute(DVA_Forces,[3 2 1]);

%% Finding H
counter3=0;
Zmin=min(z)+min(Z_TargetDVA2);
Zmax=max(z)+max(Z_TargetDVA2);
NZ=(Zmax-Zmin+Dz)*(1/Dz);
ZNEW=(Zmin:Dz:Zmax);
HF_x=zeros(length(fs),NZ,size(DVA_Forces,3)+1);
HF_y=zeros(length(fs),NZ,size(DVA_Forces,3)+1);
HF_z=zeros(length(fs),NZ,size(DVA_Forces,3)+1);
HF_Ext_Surface_AFFT=H_DVA_Extractable.H_DVA(:,:, ...
AllNodes+1:AllNodes+3);
LowerLimit=(min(z)-Zmin)*(1/Dz)+1;
UpperLimit=LowerLimit+size(HF_Ext_Surface_AFFT,2)-1;
HF_x(:,LowerLimit:UpperLimit,1)=HF_Ext_Surface_AFFT(:,:,1);
HF_y(:,LowerLimit:UpperLimit,1)=HF_Ext_Surface_AFFT(:,:,2);
HF_z(:,LowerLimit:UpperLimit,1)=HF_Ext_Surface_AFFT(:,:,3);

for i=1:1:Numberof_CSD_DVA
    ID_Force2=(DVAset(i).NodeID);
    H_Int_Surface_AFFT=H_DVA_Extractable.H_DVA(:,:, ...
    AllNodes+(3*(ID_Force2+1)-2):AllNodes+(3*(ID_Force2+1)));
    Numberof_LD_DVAi=DVAset(i).Numberof_LD_DVA;

    for j=1:1:Numberof_LD_DVAi
        counter3=counter3+1;
        z_new=z+Z_TargetDVA2(counter3);
        LowerLimit=(min(z_new)-Zmin)*(1/Dz)+1;
        UpperLimit=LowerLimit+size(H_Int_Surface_AFFT,2)-1;
        HF_Int_Surface=(H_Int_Surface_AFFT).* ...
        (repmat((DVA_Forces(:,:,counter3)),1,length(z),3));
        HF_x(:,LowerLimit:UpperLimit,counter3+1)= ...
       HF_Int_Surface(:,:,1);
        HF_y(:,LowerLimit:UpperLimit,counter3+1)= ...
       HF_Int_Surface(:,:,2);
        HF_z(:,LowerLimit:UpperLimit,counter3+1)= ...
       HF_Int_Surface(:,:,3);
    end
end
```

```
H(:,:,1)=sum(HF_x,3);
H(:,:,2)=sum(HF_y,3);
H(:,:,3)=sum(HF_z,3);

end
```

Table A.4: H_UnitaryExternalForce_E1.m function.

A.4 Exemple of 'PlotFcns'

This section gives an example of the result of using the 'PlotFcns' function which has been configured in GA options (`gaoptimset`). This function has been set as:

- `'PlotFcns',{@gaplotbestf,@gaplotbestindiv,@gaplotdistance, @gaplotgenealogy}`.

As it has been explained in the report, **`@gaplotbestf`** plots the best function value versus generation. **`@gaplotbestindiv`** plots the vector entries of the individual with the best fitness function value in each generation. **`@gaplotdistance`**, plots the average distance between individuals at each generation and **`@gaplotgenealogy`**, plots the genealogy of individuals. Lines from one generation to the next are color-coded, red lines indicate mutation children, blue lines crossover children and black lines indicate elite individuals.

In figure A.2 there is an example of this function for the optimization case of three DVAs.

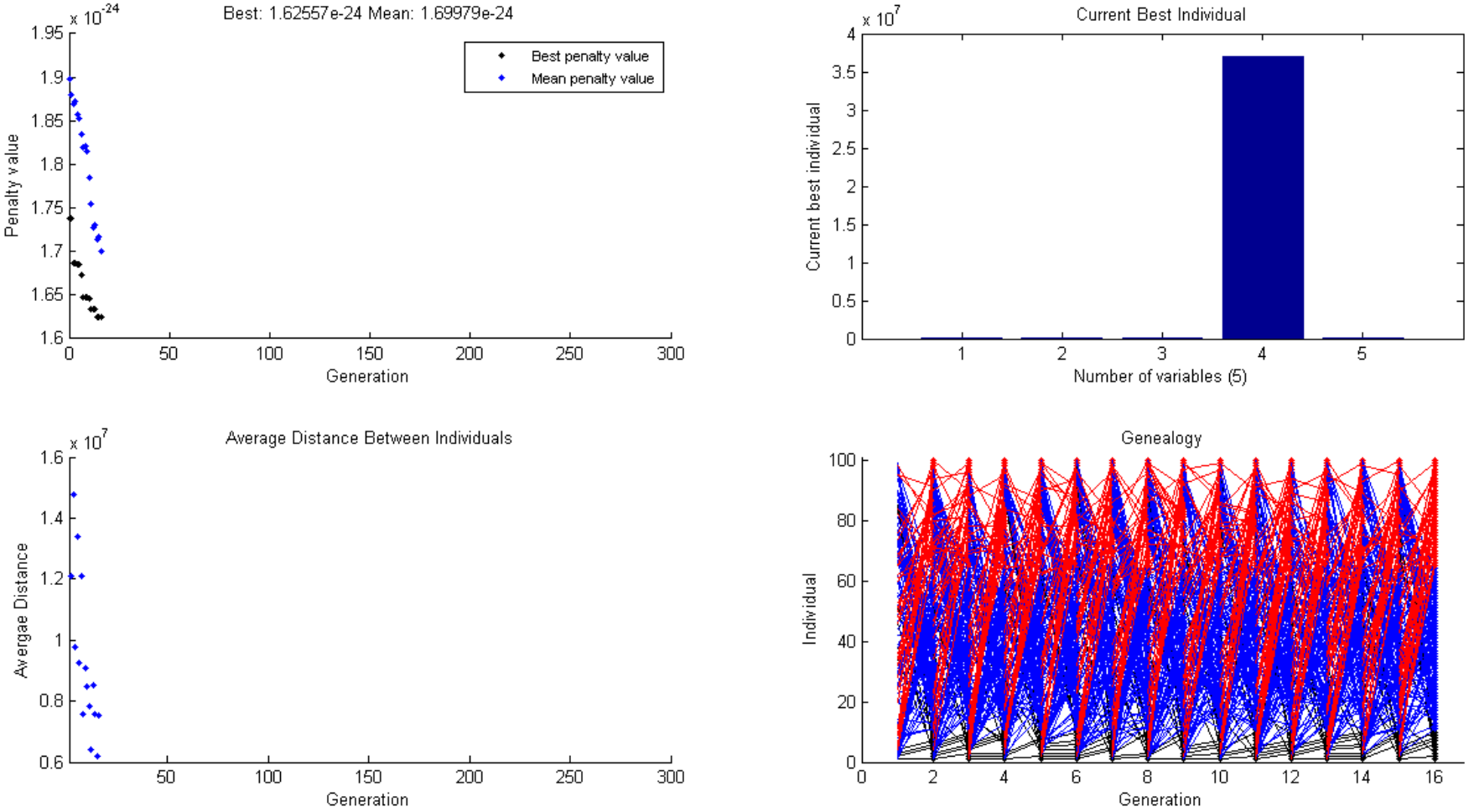

Figure A.2: Plots given by the function `'PlotFcns'` for the optimization case of three DVAs. Clockwise the graphs are: `gaplotbestf`, `gaplotbestindiv`, `gaplotgenealogy` and `gaplotdistance`.

A.5 Hybrid algorithm (GA and PS)

In this section, the **hybrid algorithm** proposed in order to improve the performance is presented, as a combination of the **GA** with a **pattern search** algorithm (PS).

Hybrid algorithm (GA and PS)

```
parpool('local', 8);

%BOUNDS
%lower bound
lb = [1 1 2 0.5*10^7 500];
%upper bound
ub = [39 10 40 0.5*10^9 1000];

%options
opts = gaoptimset(...
    'Display','iter', ...
    'CrossoverFraction', 0.6, ...
    'EliteCount', 10, ...
    'Generations', 200, ...
    'MigrationDirection', 'both', ...
    'MigrationInterval', 10, ...
    'MigrationFraction', 0.2, ...
    'PlotFcns', {@gaplotbestf,@gaplotbestindiv, ... ...
    @gaplotdistance,@gaplotgenealogy}, ...
    'PopulationSize', 100, ...
    'PopulationType', 'doubleVector', ...
    'StallGenLimit', 15, ...
    'StallTimeLimit', 10800, ...
    'TolFun', 1e-10, ...
    'UseParallel', true, ...
    'Vectorized', 'off');

rng(1, 'twister') %for reproductibility
[x_ga fval exitflag] = ga(@DVA_opt2,5,[],[],[],[], ...
lb,ub,[],[1 2 3],opts);

x0 = [x_ga(4) x_ga(5)];

%BOUNDS
%lower bound
lb = [0.5*10^7 500];
```

```
%upper bound
ub = [0.5*10^9 2000];

options = psoptimset(...
    'Tolfun', 1e-10, ...
    'TolX', 1e-10 ...
    );

x_lin = patternsearch(@fitness_lin_1DVA,x0,[],[],[],[], ...
lb,ub,[],options);

x_1DVA = [x_ga(1) x_ga(2) x_ga(3) x_lin(1) x_lin(2)];
```

Table A.5: Hybrid algorithm.

B. Input examples

This section presents the data from the **Evaluators**. This dataset has been extracted from the archive H_DVA+Eval_SpaceDomain.mat. It is an 80x8 dataset; a table which includes important information to place DVAs and to study the vibration response. It contains the following information:

- **NodeID** values.
- Coordinates of the NodeIDs (**xNode** and **yNode**)
- Identification of the force applied, **ForceID**.
- Type of force applied, **ForceType** (can be **'Ext'** or **Int')**.
- Vector components of the forces, by **xForce** and **yForce**.
- Configuration of the force, **ThreeD_Normal**, if it is 3D or normal (not 3D).

All this information is presented in the following tables; as Evaluators are represented in a large table, it has been divided in four parts: tables B.1, B.2, B.3 and B.4.

NodeID	xNode	yNode	ForceID	ForceType	xForce	yForce	ThreeD_Normal
1	-0.4693	43	40	Ext	0	43	Normal
2	-0.9271	43	40	Ext	0	43	Normal
3	-1.3620	43	40	Ext	0	43	Normal
4	-1.7634	42	40	Ext	0	43	Normal
5	-2.1213	42	40	Ext	0	43	Normal
6	-2.4271	42	40	Ext	0	43	Normal
7	-2.6730	41	40	Ext	0	43	Normal
8	-2.8532	41	40	Ext	0	43	Normal
9	-2.9631	40	40	Ext	0	43	Normal
10	-3	40	40	Ext	0	43	Normal
11	-2.9631	39.531	40	Ext	0	43	Normal
12	-2.8532	39.073	40	Ext	0	43	Normal
13	-2.6730	38.638	40	Ext	0	43	Normal
14	-2.4271	38.237	40	Ext	0	43	Normal
15	-2.1213	37.879	40	Ext	0	43	Normal
16	-1.7634	37.573	40	Ext	0	43	Normal
17	-1.3620	37.327	40	Ext	0	43	Normal
18	-0.9271	37.147	40	Ext	0	43	Normal
19	-0.4693	37.037	40	Ext	0	43	Normal
20	0.0000	37	40	Ext	0	43	Normal
21	0.4693	37.037	40	Ext	0	43	Normal
22	0.9271	37.147	40	Ext	0	43	Normal

Table B.1: Evaluators, first part.

NodeID	xNode	yNode	ForceID	ForceType	xForce	yForce	ThreeD_Normal
23	1.3620	37.327	40	Ext	0	43	Normal
24	1.7634	37.573	40	Ext	0	43	Normal
25	2.1213	37.879	40	Ext	0	43	Normal
26	2.4271	38.237	40	Ext	0	43	Normal
27	2.6730	38.638	40	Ext	0	43	Normal
28	2.8532	39.072	40	Ext	0	43	Normal
29	2.9631	39.531	40	Ext	0	43	Normal
30	3	40	40	Ext	0	43	Normal
31	2.9631	40.469	40	Ext	0	43	Normal
32	2.8532	40.927	40	Ext	0	43	Normal
33	2.6730	41.362	40	Ext	0	43	Normal
34	2.4271	41.763	40	Ext	0	43	Normal
35	2.1213	42.121	40	Ext	0	43	Normal
36	1.7634	42.427	40	Ext	0	43	Normal
37	1.3620	42.673	40	Ext	0	43	Normal
38	0.9271	42.853	40	Ext	0	43	Normal
39	0.4693	42.963	40	Ext	0	43	Normal
40	0.0000	43	40	Ext	0	43	Normal
41	0	0	40	Ext	0	43	3D
41	0	0	1	Int	-0.4693	42.963	3D
41	0	0	2	Int	-0.9271	42.853	3D
41	0	0	3	Int	-1.3620	42.673	3D

Table B.2: Evaluators, second part.

NodeID	xNode	yNode	ForceID	ForceType	xForce	yForce	ThreeD_Normal
41	0	0	4	Int	-1.7634	42.427	3D
41	0	0	5	Int	-2.1213	42.121	3D
41	0	0	6	Int	-2.4271	41.763	3D
41	0	0	7	Int	-2.6730	41.362	3D
41	0	0	8	Int	-2.8532	40.927	3D
41	0	0	9	Int	-2.9631	40.469	3D
41	0	0	10	Int	-3	40	3D
41	0	0	11	Int	-2.9631	39.531	3D
41	0	0	12	Int	-2.8532	39.073	3D
41	0	0	13	Int	-2.6730	38.638	3D
41	0	0	14	Int	-2.4271	38.237	3D
41	0	0	15	Int	-2.1213	37.879	3D
41	0	0	16	Int	-1.7634	37.573	3D
41	0	0	17	Int	-1.3620	37.327	3D
41	0	0	18	Int	-0.9271	37.147	3D
41	0	0	19	Int	-0.4693	37.037	3D
41	0	0	20	Int	0.0000	37	3D
41	0	0	21	Int	0.4693	37.037	3D
41	0	0	22	Int	0.9271	37.147	3D
41	0	0	23	Int	1.3620	37.327	3D
41	0	0	24	Int	1.7634	37.573	3D
41	0	0	25	Int	2.1213	37.879	3D

Table B.3: Evaluators, third part.

NodeID	xNode	yNode	ForceID	ForceType	xForce	yForce	ThreeD_Normal
41	0	0	16	Int	-1.7634	37.573	3D
41	0	0	17	Int	-1.3620	37.327	3D
41	0	0	18	Int	-0.9271	37.147	3D
41	0	0	19	Int	-0.4693	37.037	3D
41	0	0	20	Int	0.0000	37	3D
41	0	0	21	Int	0.4693	37.037	3D
41	0	0	22	Int	0.9271	37.147	3D
41	0	0	23	Int	1.3620	37.327	3D
41	0	0	24	Int	1.7634	37.573	3D
41	0	0	25	Int	2.1213	37.879	3D
41	0	0	26	Int	2.4271	38.237	3D
41	0	0	27	Int	2.6730	38.638	3D
41	0	0	28	Int	2.8532	39.073	3D
41	0	0	29	Int	2.9631	39.531	3D
41	0	0	30	Int	3	40	3D
41	0	0	31	Int	2.9631	40.469	3D
41	0	0	32	Int	2.8532	40.927	3D
41	0	0	33	Int	2.6730	41.362	3D
41	0	0	34	Int	2.4271	41.763	3D
41	0	0	35	Int	2.1213	42.121	3D
41	0	0	36	Int	1.7634	42.427	3D
41	0	0	37	Int	1.3620	42.673	3D
41	0	0	38	Int	0.9271	42.853	3D
41	0	0	39	Int	0.4693	42.963	3D

Table B.4: Evaluators, fourth part.

C. Output graphs

In this section, there are the graphs obtained after running the algorithm for each number of DVAs in the cross section. These graphs represent the square of the acceleration, in absolute value, in front of the range of frequencies for each optimization case.

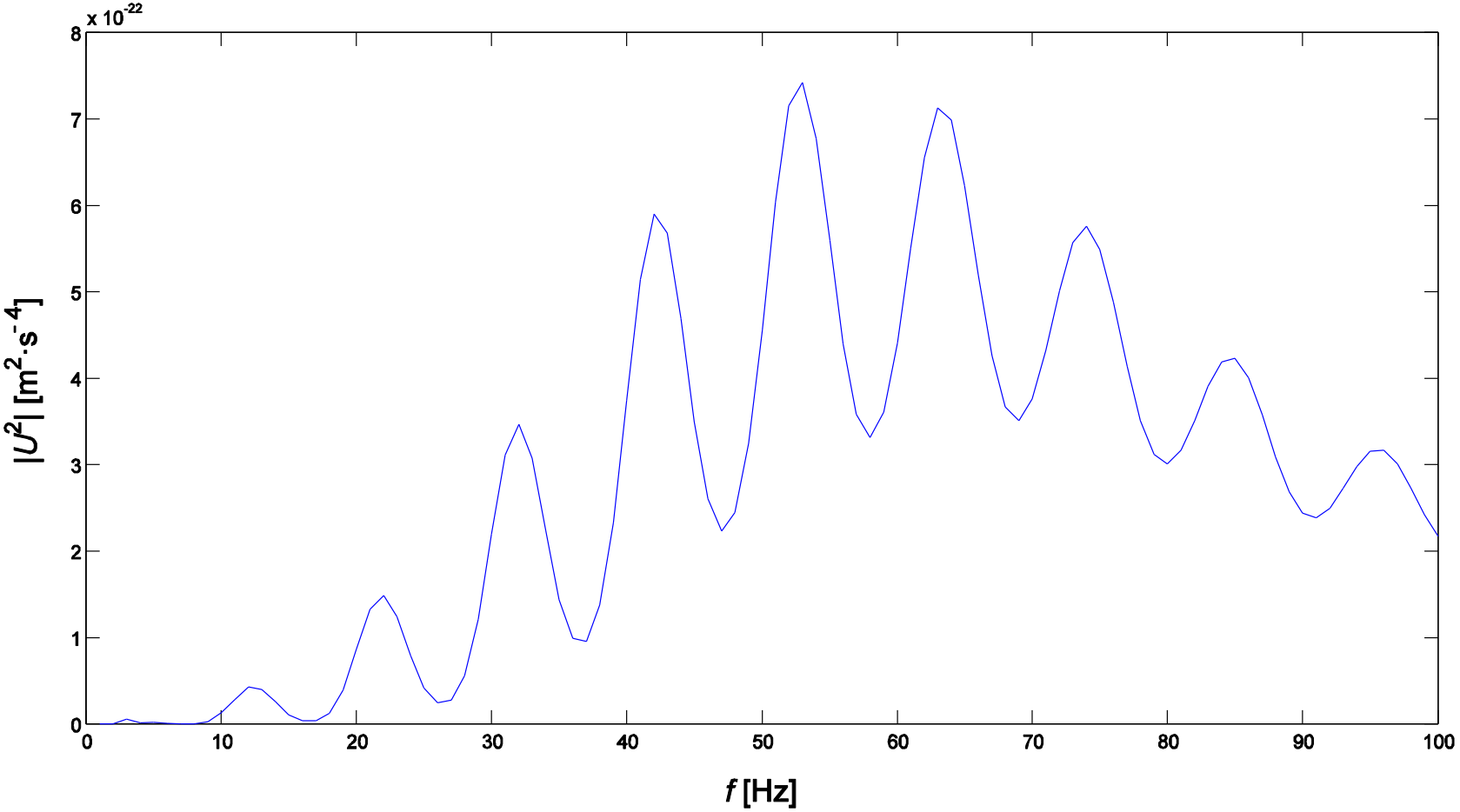

Figure C.1: Graph of the square of the acceleration against frequency, without DVAs.

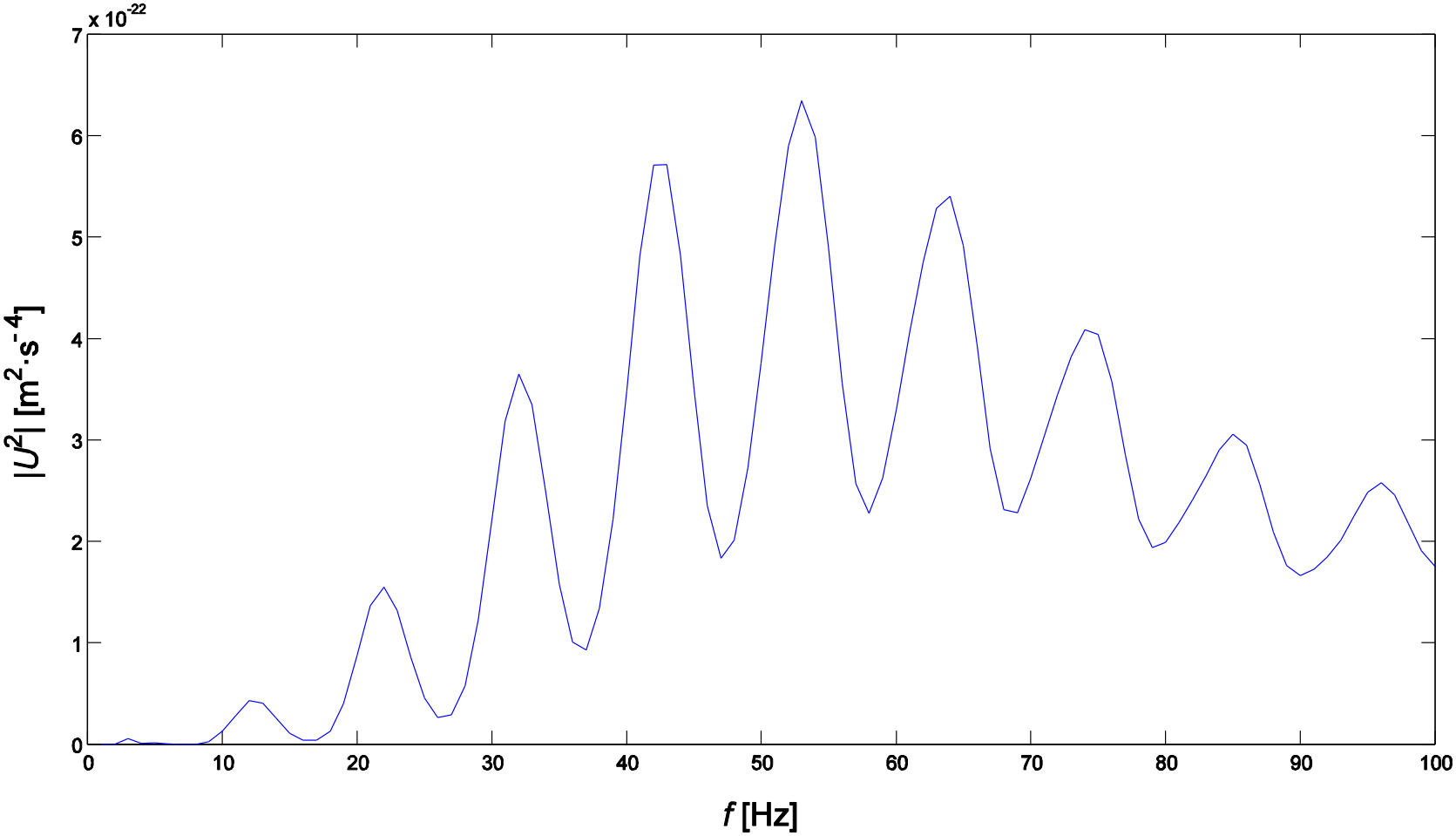

Figure C.2: Graph of the square of the acceleration against frequency, for one DVA.

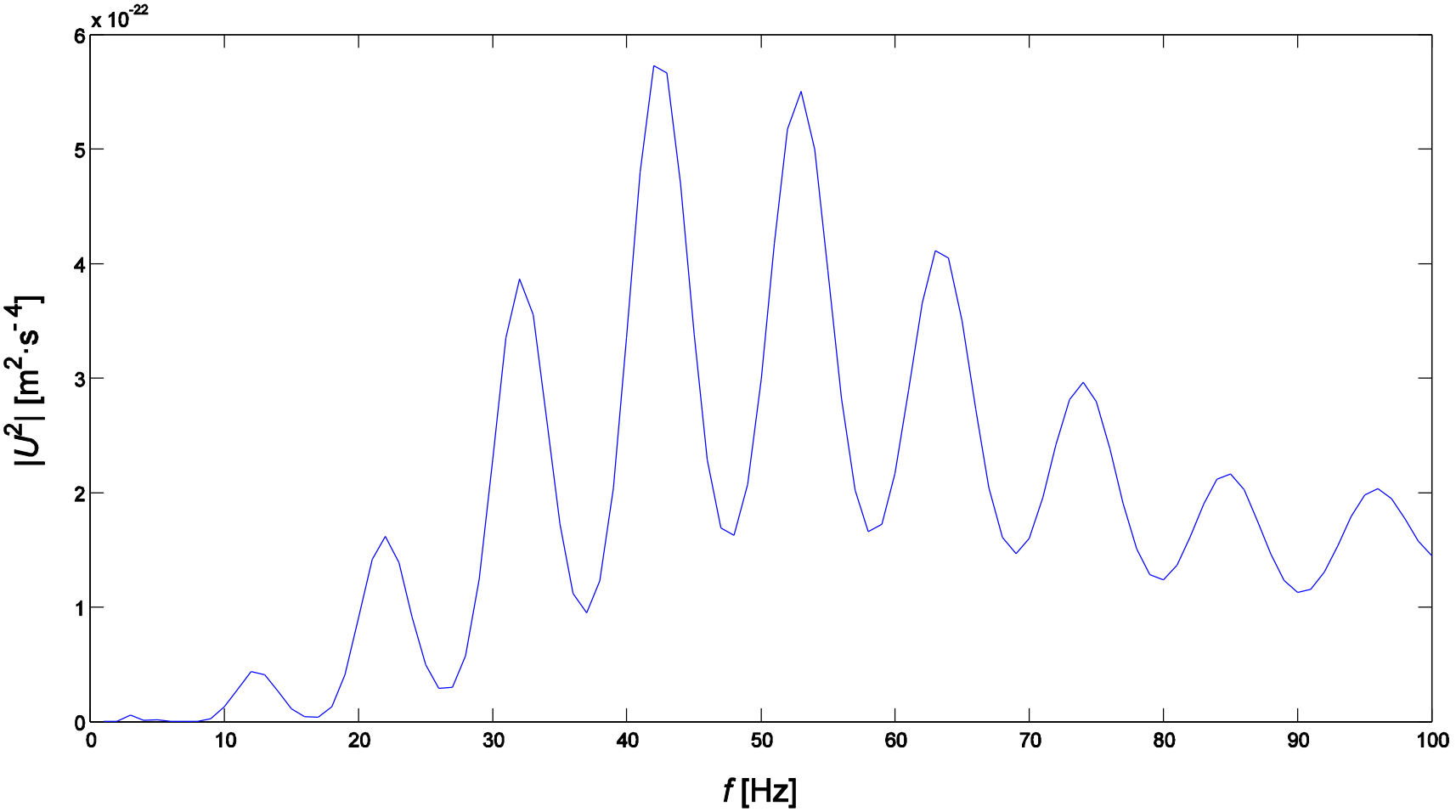

Figure C.3: Graph of the square of the acceleration against frequency, for two DVAs.

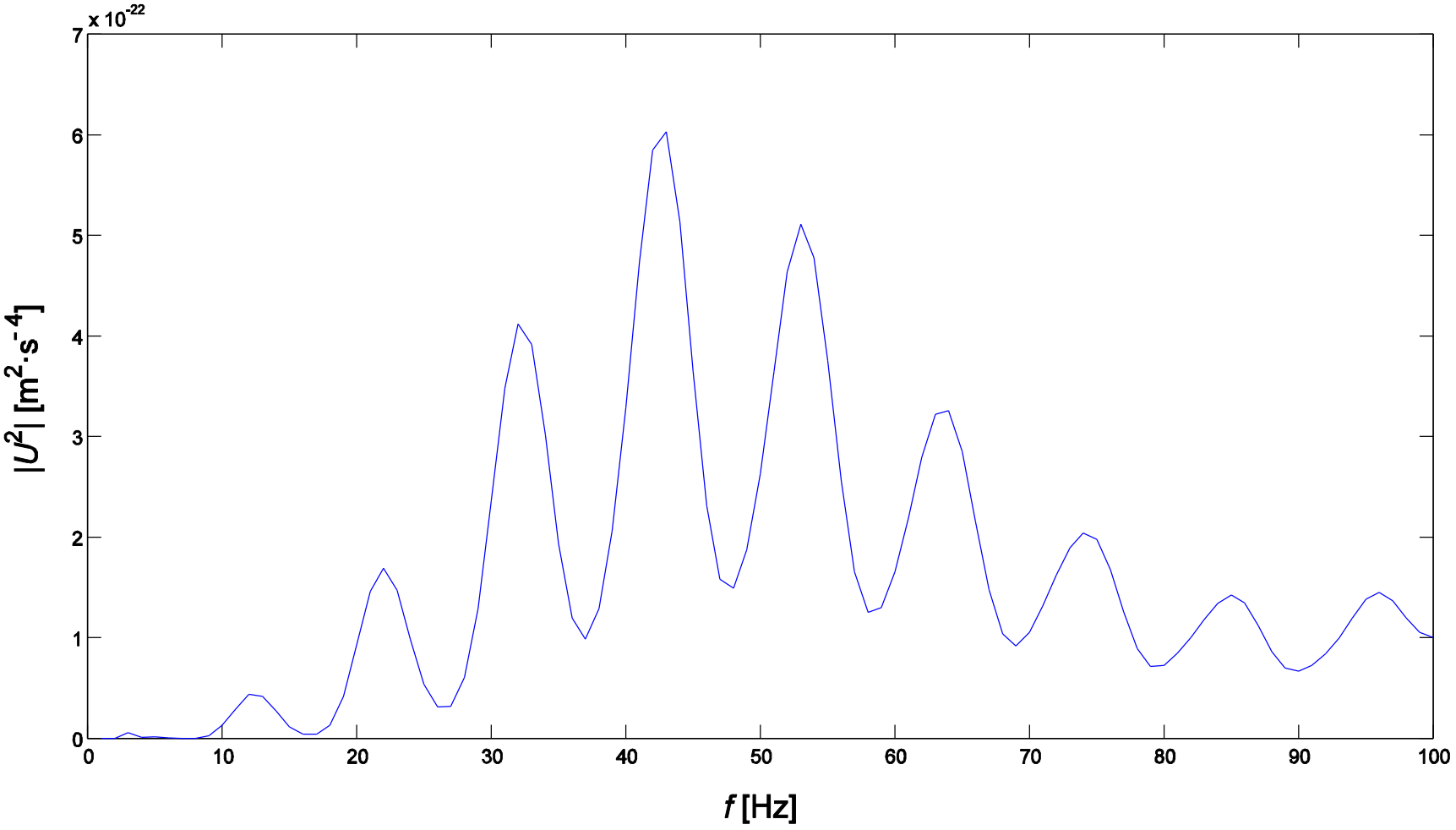

Figure C.4: Graph of the square of the acceleration against frequency, for three DVAs.

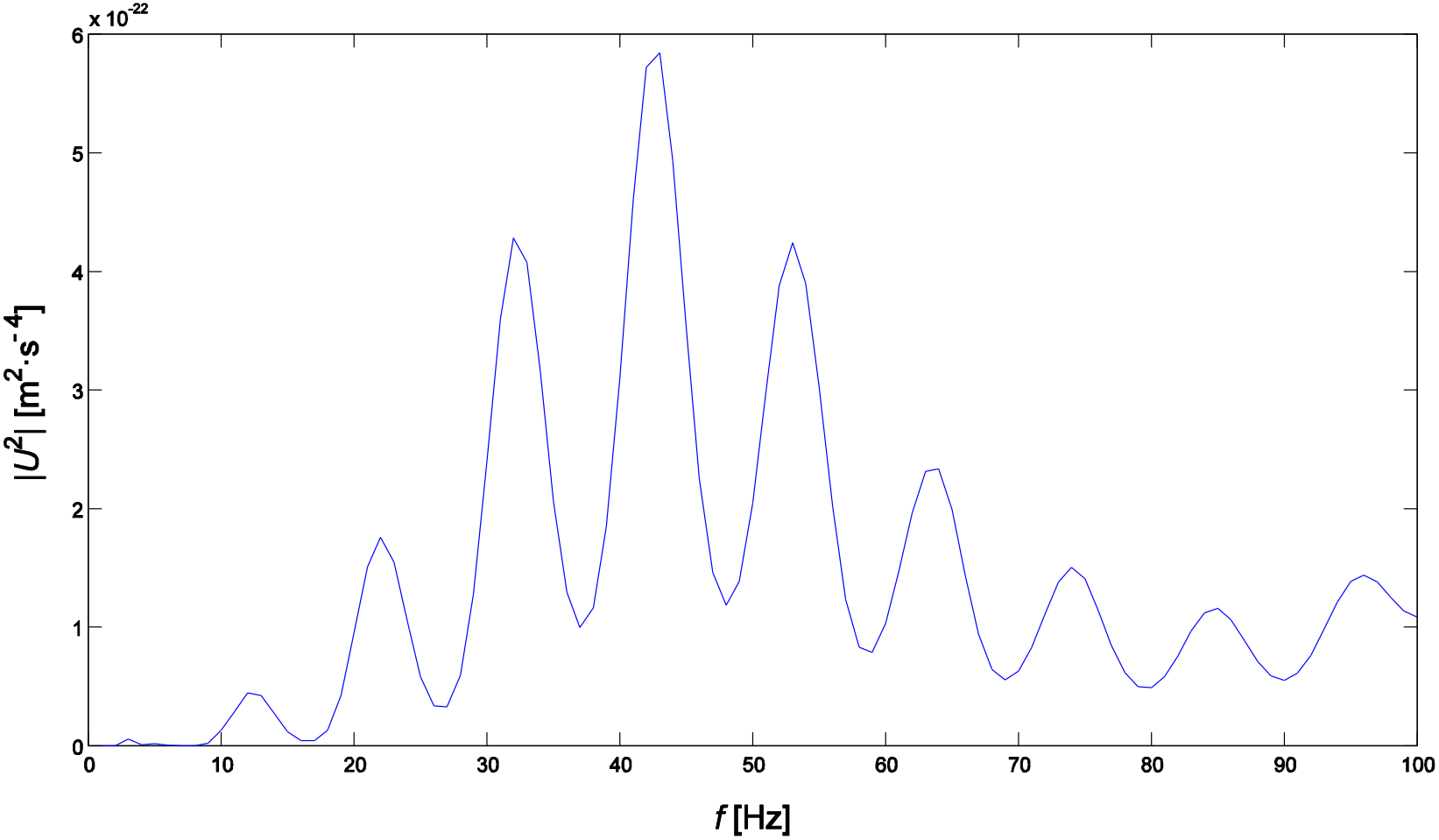

Figure C.5: Graph of the square of the acceleration against frequency, for four DVAs.

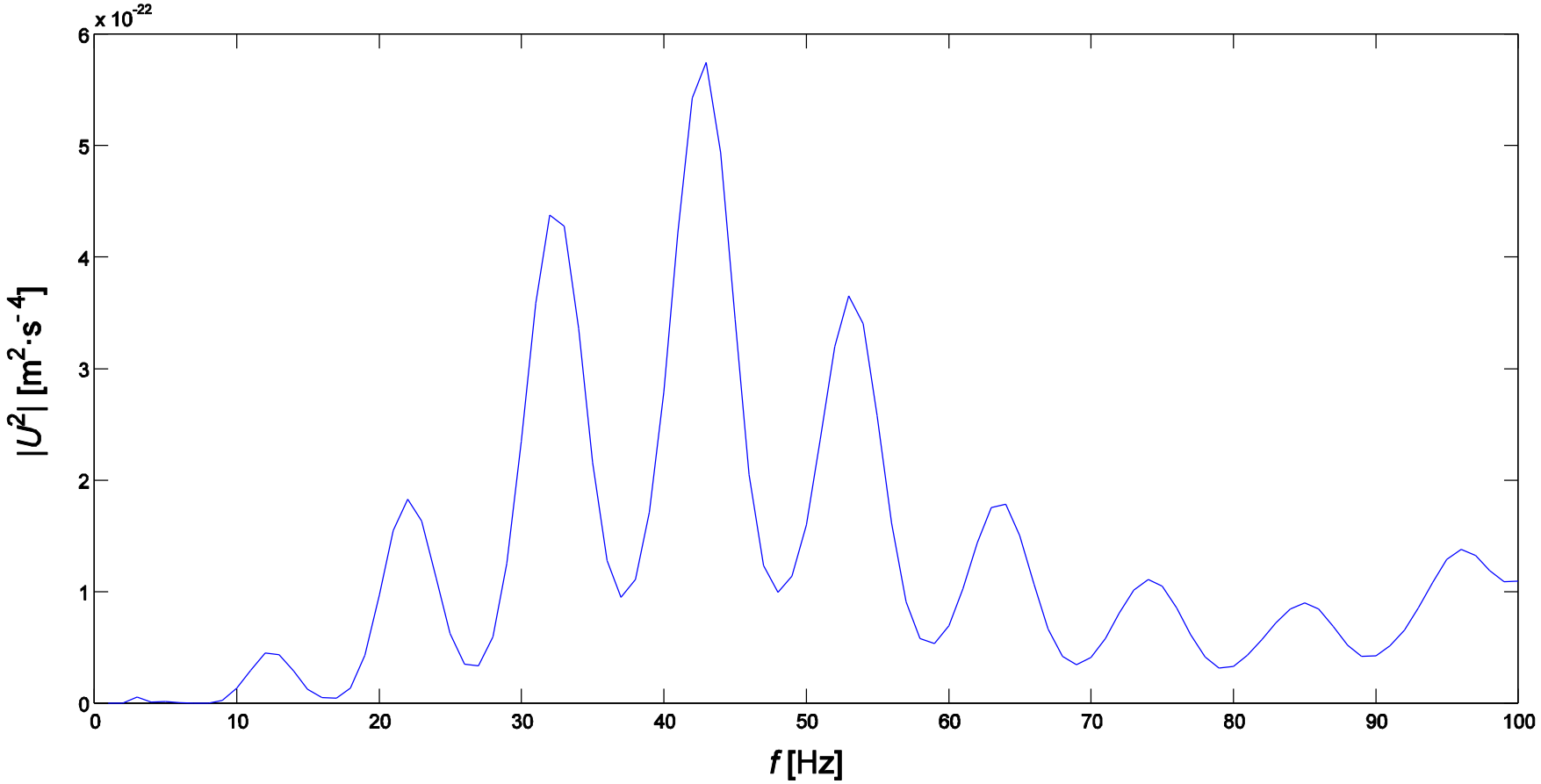

Figure C.6: Graph of the square of the acceleration against frequency, for five DVAs.

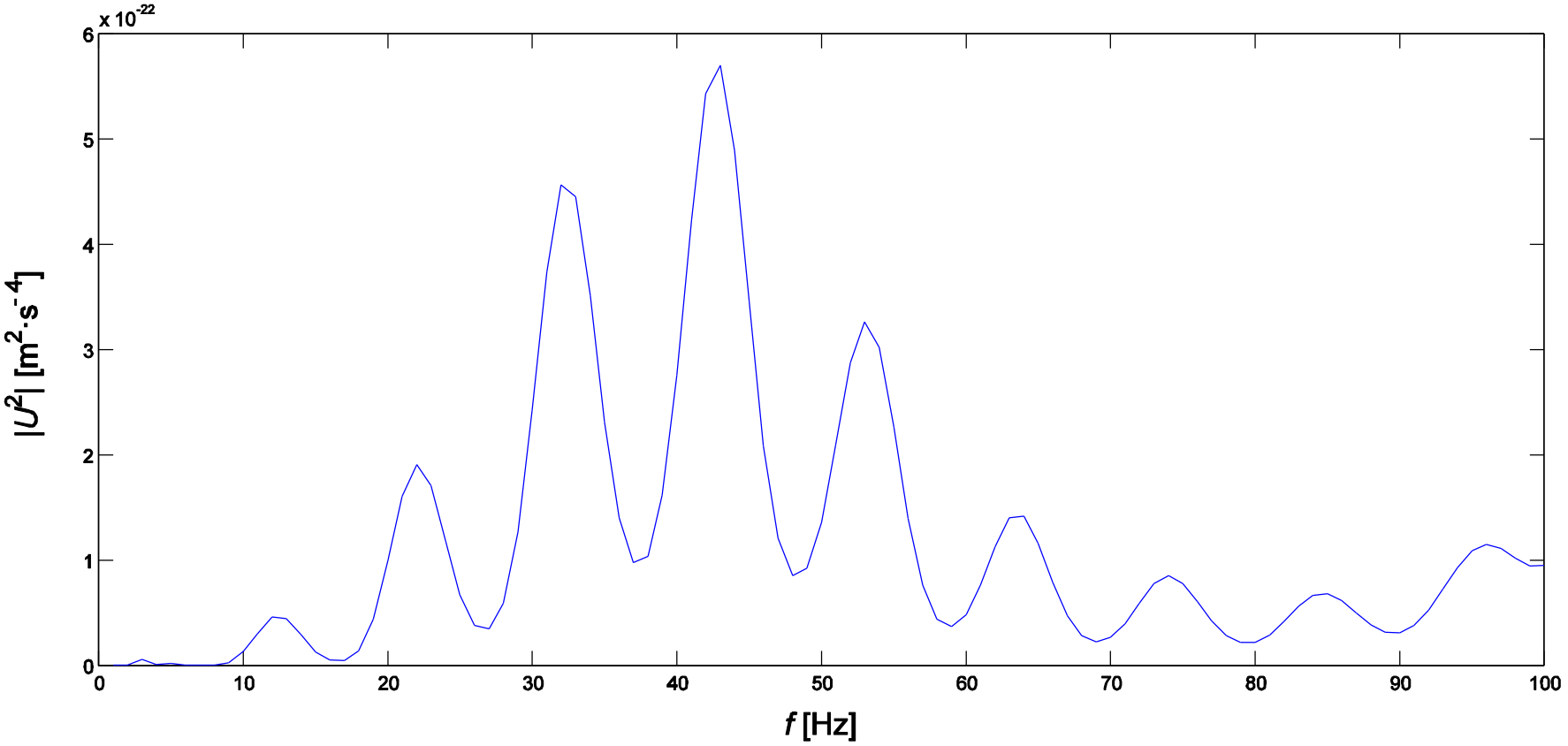

Figure C.7: Graph of the square of the acceleration against frequency, for six DVAs.

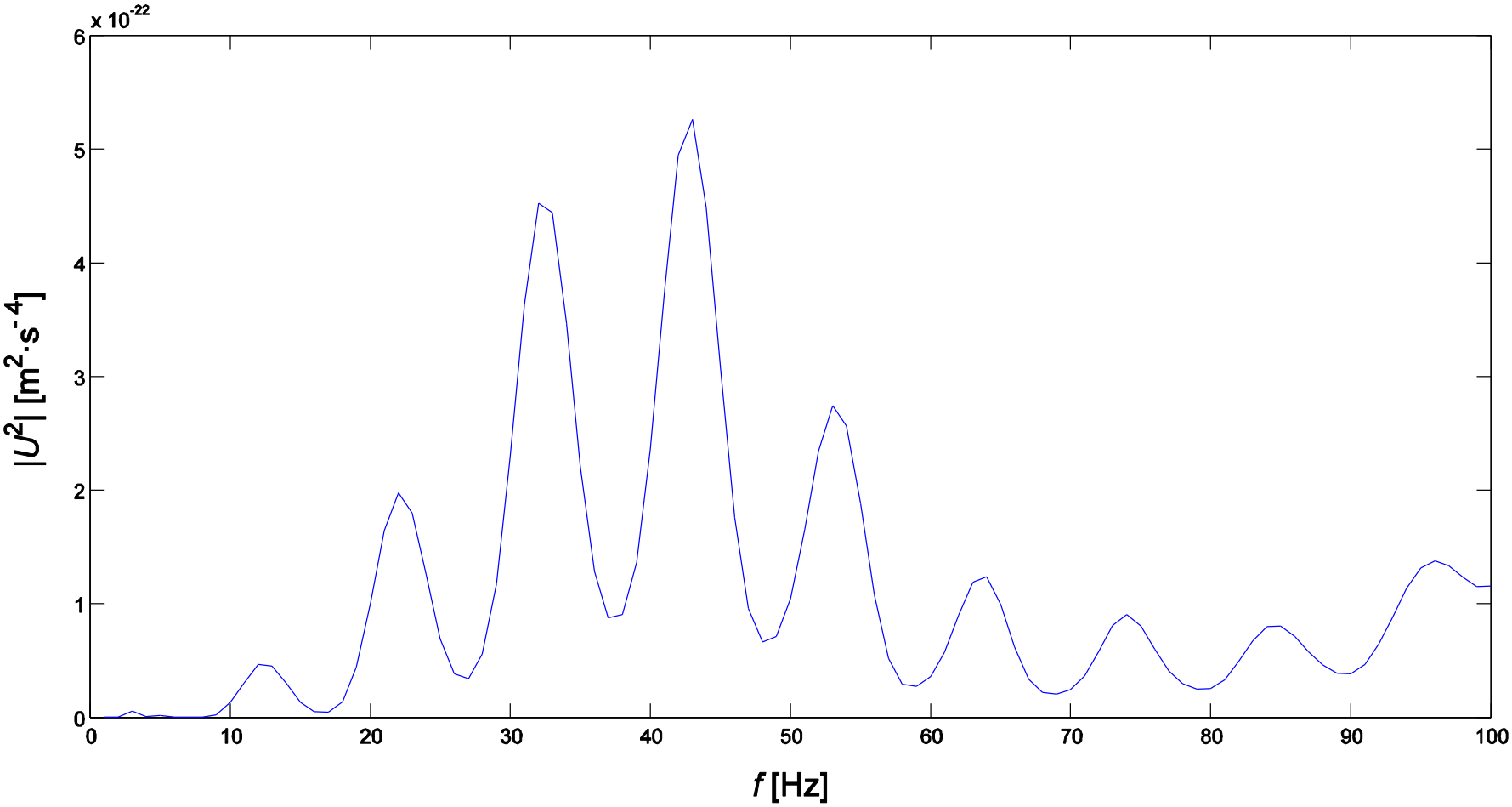

Figure C.8: Graph of the square of the acceleration against frequency, for seven DVAs.

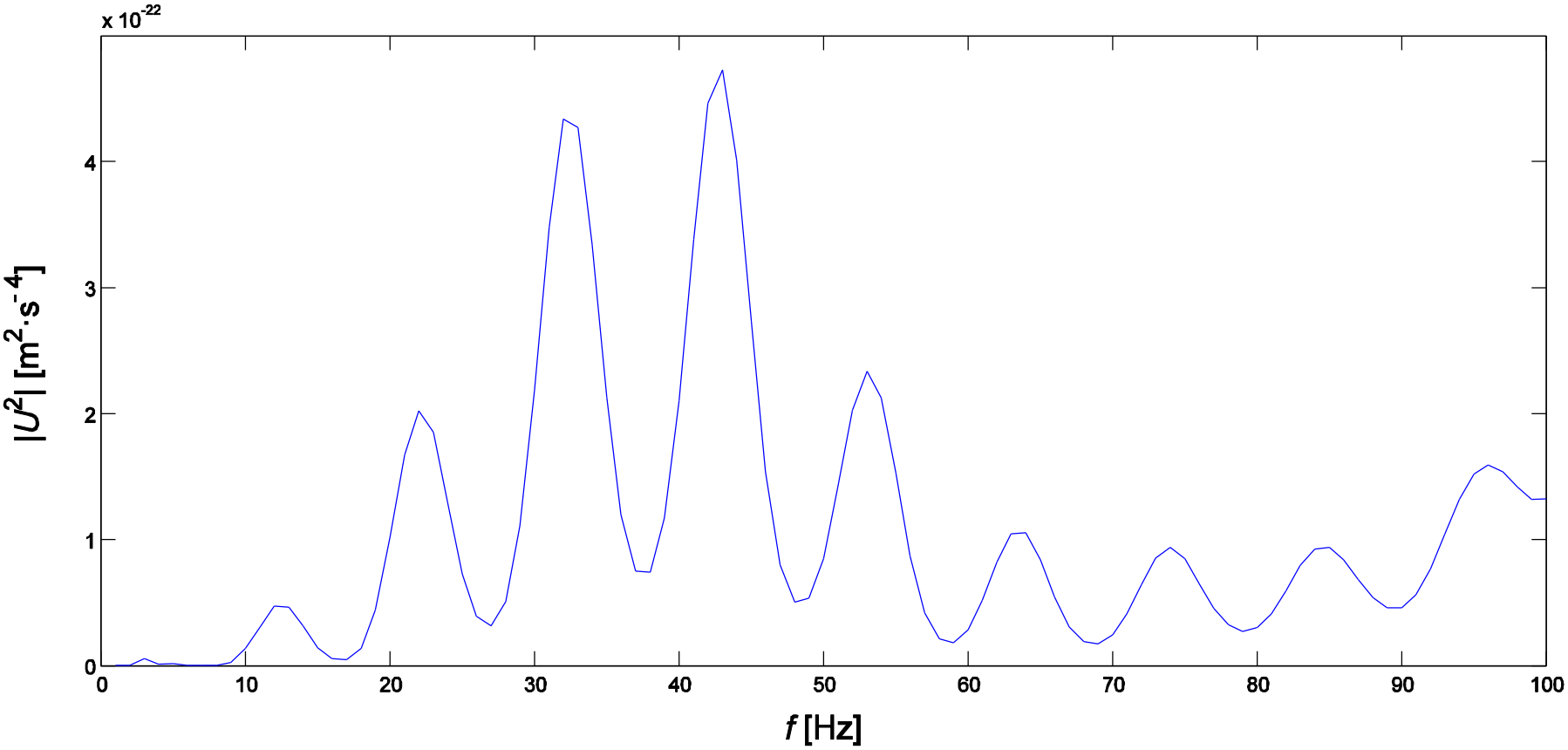

Figure C.9: Graph of the square of the acceleration against frequency, for eight DVAs.

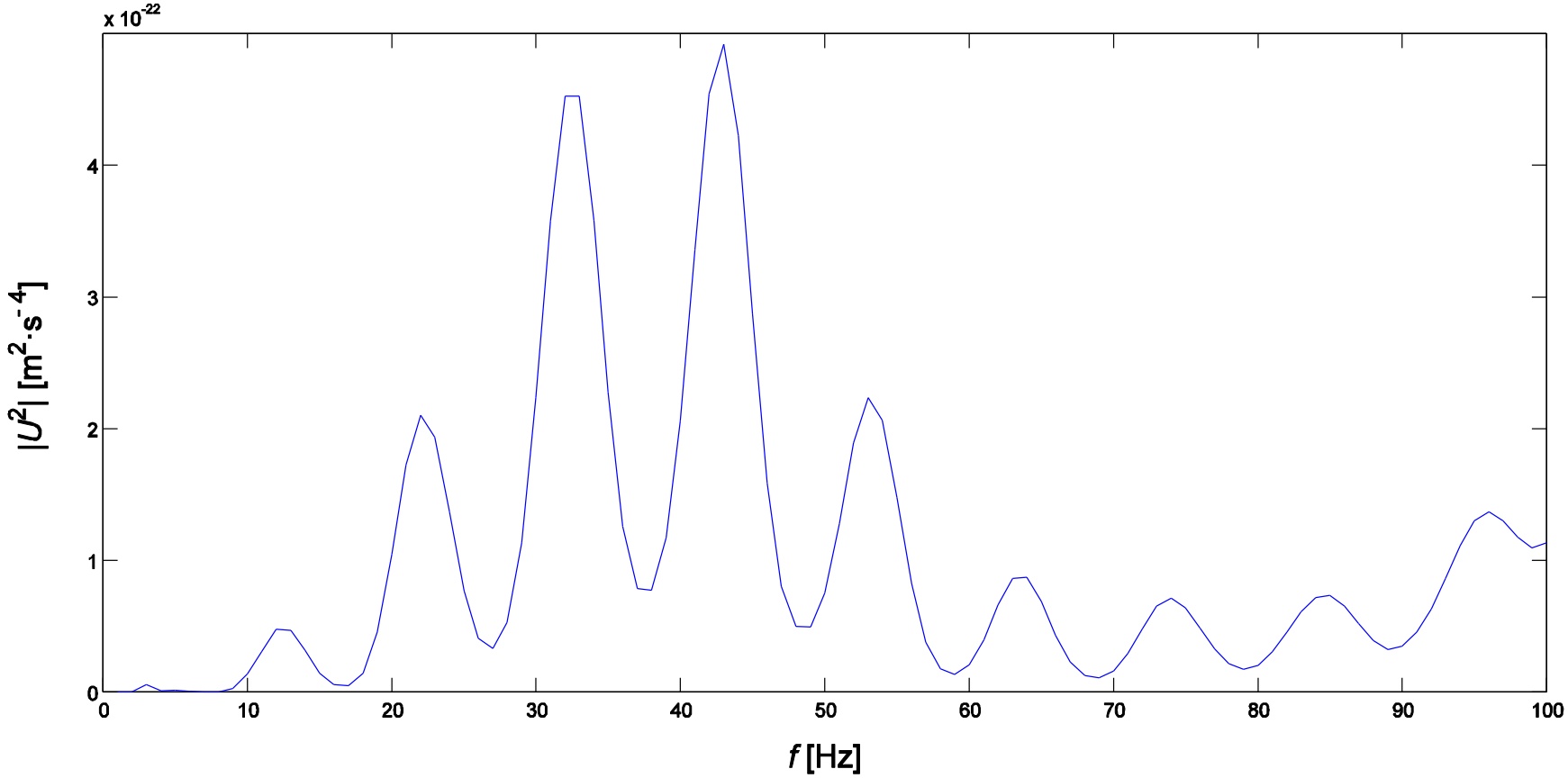

Figure C.10: Graph of the square of the acceleration against frequency, for nine DVAs.

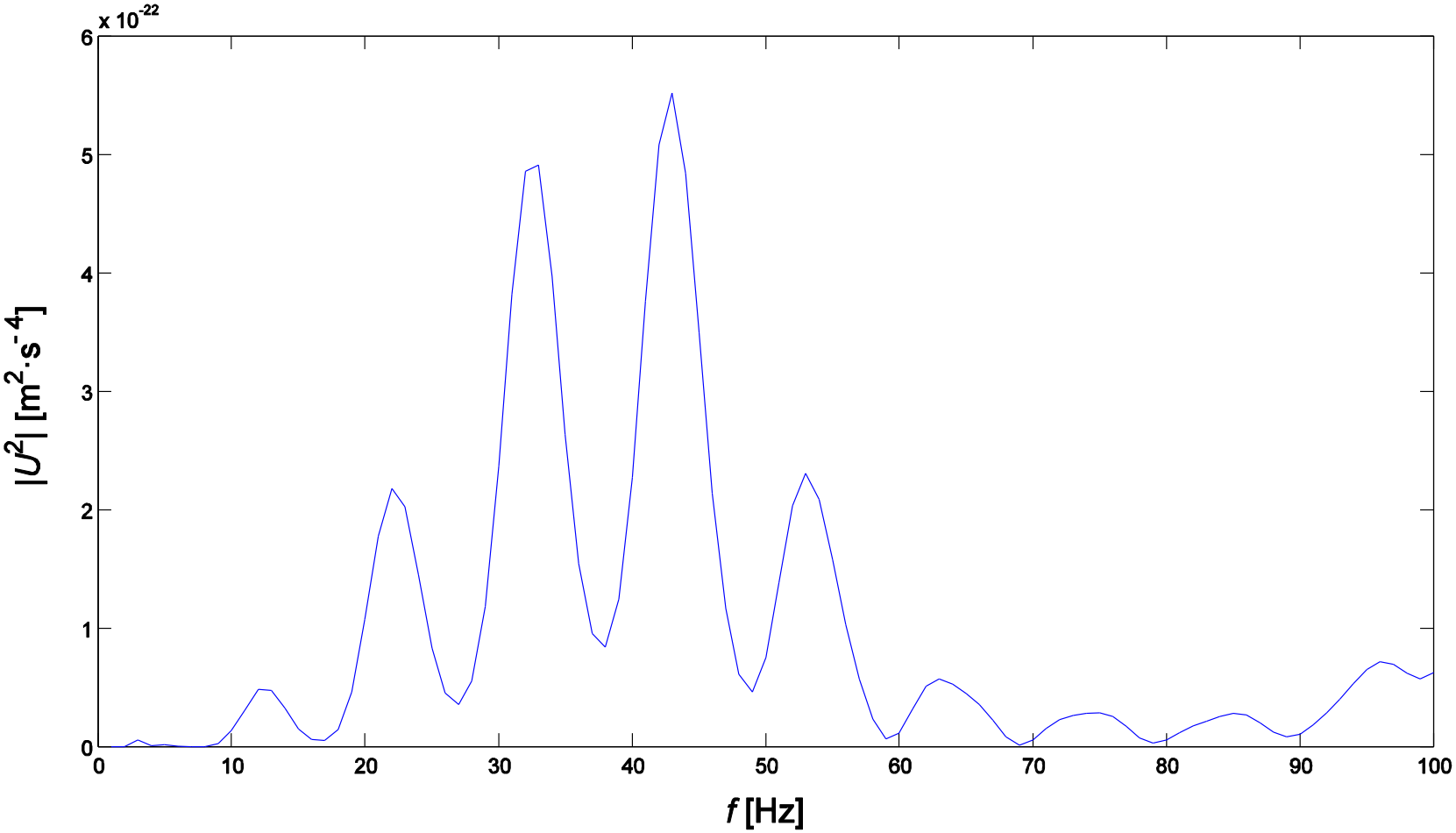

Figure C.11: Graph of the square of the acceleration against frequency, for ten DVAs.

D. Planning and scheduling

In this section the complete explanations of the tasks and interdependencies of timelines proposed in the report, are presented (chapter 12). Both cases are presented: the planning and scheduling of the current study as well as the proposed to continue the study through the future lines of investigation.

D.1 Planning and scheduling of the current study

First of all, the tasks to develop will be explained, then the identification of the interdependencies between the tasks, a planning with the amount of estimated time dedicated and finally the Gantt chart to clarify the timeline.

1. Brief description of the tasks and work packages that must be developed to achieve the goal of the project study:

A. Information review and research: first approach of the project reviewing and researching information related with the study.

A.1 Review railway-induced vibration problem: review the basics of railway-induced vibration problem applied to the project.

A.2 Review how to couple DVAs to tunnel walls: search for the procedure of how to couple DVAs to tunnel walls.

A.3 Genetic algorithm programming research: research of how to implement genetic algorithm code and also learn how to use *Global Optimization Toolbox* from MATLAB.

B. Identification of the problem: evaluation of the problem (case study) to solve and the functions/equations to be optimized.

C. Background: all the theoretical concepts needed to evaluate the problem and first approach to the genetic algorithm programming.

C.1 Green's functions evaluation of the superstructure/tunnel/ /ground/building system: theoretical background needed to be able to set the problem. The final equations will be provided by TFG's director and his research group.

C.2 Description of the dynamic response of the train: theoretical background needed to be able to set the problem. The final equations will be provided by TFG's director and his research group.

D. Optimization process: development of optimization procedure for each study case.

D.1 Formulation of the problem: formulation of the optimization problem for the different study cases and for the global problem.

D.2 Specific case studies: study of the particular cases.

D.2.1 Implementation of the algorithm: programming the complete algorithm for the study case, included the genetic algorithm.

D.2.2 Refinement of the genetic algorithm: refinement of the genetic algorithm in order to allow more accurate treatment.

D.2.3 Validation of the algorithm: correlation of the results with data provided by the TFG's director and his research group.

D.3 Global case:

D.3.1 Implementation of the algorithm: programming the complete algorithm for the global case included the genetic algorithm.

D.3.2 Refinement of the genetic algorithm: refinement of the genetic algorithm in order to allow more accurate treatment.

D.3.3 Validation of the algorithm: correlation of the results with data provided by the TFG's director and his research group.

2. Identification of the dependences amongst tasks:

Code of task	Task identification	Preceding task(s)
TFG		
A	Information review and research	-
A.1	Review railway-induced vibration problem	-
A.2	Review how to couple DVAs to tunnel walls	-
A.3	Genetic algorithm programming research	-
B	Identification of the problem	A.1, A.2
C	Background	A, B
C.1	Green's functions evaluation of the superstructure/tunnel/ /ground/building system	A.1, A.2, A.3, B
C.2	Description of the dynamic response of the train	A.1, A.2, A.3, B
D	Optimization process	A, B, C
D.1	Formulation of the problem	A.1, A.2, B, C
D.2	Specific case studies	A.1, A.2, B, C, D.1
D.2.1	Implementation of the algorithm	A.3, D.1, D.2
D.2.2	Refinement of the genetic algorithm	A.3, D.1, D2, D.2.1
D.2.3	Validation of the algorithm	A.3, D.1, D.2, D.2.1, D.2.2
D.3	Global case	A.3, B, C, D.1, D.2
D.3.1	Implementation of the algorithm	A.3, B, C, D.1, D.3
D.3.2	Refinement of the genetic algorithm	A.3, B, C, D.1, D.3, D.3.1
D.3.3	Validation of the algorithm	A.3, B, C, D.1, D.3, D.3.1, D.3.2

Table D.1: Interdependences of the tasks for the current study.

3. Estimation of the necessary time to allocate the tasks:

Code of task	Task identification	Level of efforts (hours)
-	TFG	540
A	Information review and research	54
A.1	Review railway-induced vibration problem	18
A.2	Review how to couple DVAs to tunnel walls	18
A.3	Genetic algorithm programming research	18
B	Identification of the problem	54
C	Background	135
C.1	Green's functions evaluation of the superstructure/tunnel/ /ground/building system	67
C.2	Description of the dynamic response of the train	68
D	Optimization process	297
D.1	Formulation of the problem	65
D.2	Specific case studies	100
D.2.1	Implementation of the algorithm	47
D.2.2	Refinement of the genetic algorithm	43
D.2.3	Validation of the algorithm	10
D.3	Global case	132
D.3.1	Implementation of the algorithm	62
D.3.2	Refinement of the genetic algorithm	55
D.3.3	Validation of the algorithm	15

Table D.2: Estimated time to develop the tasks of the current study.

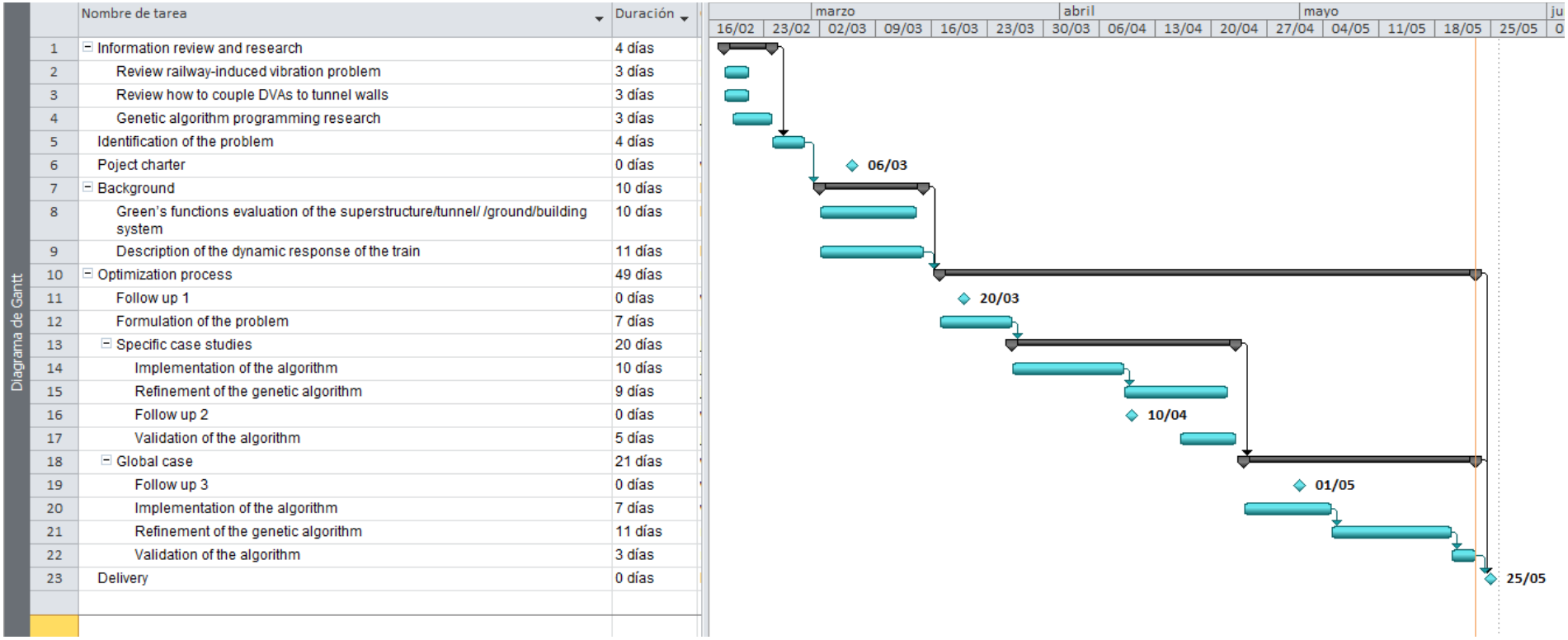

Figure D.1: Gantt chart of the current study.

D.2 Future planning and scheduling

First of all, the tasks to develop will be explained, then the identification of the interdependencies between the tasks, a planning with the amount of estimated time dedicated and finally the Gantt chart to clarify the timeline.

1. Brief description of the tasks and work packages that must be developed to study the future lines of investigation:

A. Information review and research: first approach of the moving load problems as well as to understand how to include the building in the system. Furthermore, to investigate new ways to speed up the code.

A.1 Review railway-induced vibration problem with moving load: review the basics of moving load applied to railway-induced vibration problem.

A.2 Review how to introduce the building in the system: search for the procedure of how to couple DVAs to tunnel walls.

A.3 Genetic algorithm programming research: research of how to implement a more complex and accurate refinement of the GA and look for alternative ways to improve the performance.

B. Identification of the problem: evaluation of the problem (case study) to solve and the functions/equations to be optimized.

C. Theoretical background: all the theoretical concepts needed to evaluate the problem.

C.1 Moving load theoretical description: theoretical background needed to be able to set the problem.

C.2 Inclusion of the building as a part of the system: develop the theoretical concepts to be able to introduce this part to the global system.

C.3 Description of the dynamic response of the train: theoretical background needed to be able to set the problem.

D. **Optimization process**: development of the optimization procedure.

D.1 Formulation of the problem: formulation of the optimization problem for the different study cases and for the global problem.

D.2 Study of the performance of each part of the algorithm: study of each component of the algorithm and how to speed it up.

D.2.1 Implementation of the algorithm: programming the complete algorithm for the study case included the genetic algorithm.

D.2.2 Study the effects of mutation strategy: study of the effects generated by changing the mutation ratio.

D.2.3 Study the effects of crossover strategy: study of the effects generated by changing the crossover ratio.

D.2.4 Study the effects of elitism strategy: study of the effects generated by changing the elitism ratio.

D.2.5 Study of the improvement of H_UnitaryExternal Force_E1.m: study of how to improve this function in order to speed up the code.

D.3 Optimization case:

D.3.1 Implementation of the algorithm: programming the complete algorithm included the genetic algorithm.

D.3.2 Final refinement of the genetic algorithm: after all the improving methods, at least with the final code, it is needed to evaluate the GA parameters in order to allow more accurate treatment.

D.3.3 Validation of the algorithm: correlation of the results with data provided by the research group.

2. Identification of the dependences amongst tasks:

Code of task	Task identification	Preceding task(s)
TFG		
A	Information review and research	-
A.1	Review railway-induced vibration problem with moving load	-
A.2	Review how to introduce the building in the system	-
A.3	Genetic algorithm programming research	-
B	Identification of the problem	A.1, A.2
C	Theoretical background	A, B
C.1	Moving load theoretical description	A.1, A.2, B
C.2	Inclusion of the building as a part of the system	A.1, A.2, B
C.3	Description of the dynamic response of the train	A.1, A.2, B, C.1, C.2
D	Optimization process	A, B, C
D.1	Formulation of the problem	A, B, C
D.2	Study of the performance of each part of the algorithm	A, B, C, D.1
D.2.1	Implementation of the algorithm	A, B, C, D.1
D.2.2	Study the effects of mutation strategy	A, B, C, D.1, D.2.1
D.2.3	Study the effects of crossover strategy	A, B, C, D.1, D.2.1
D.2.4	Study the effects of elitism strategy	A, B, C, D.1, D.2.1
D.2.5	Study of the improvement of H_UnitaryExternal Force_E1.m	A, B, C, D.1, D.2.1
D.3	Global case	A.3, B, C, D.1, D.2
D.3.1	Implementation of the algorithm	A.3, B, C, D.1, D.2
D.3.2	Final refinement of the genetic algorithm	A.3, B, C, D.1, D.2, D.3.1
D.3.3	Validation of the algorithm	A.3, B, C, D.1, D.3, D.3.1, D.3.2

Table D.3: Interdependences of the tasks for the future study.

3. Estimation of the necessary time to allocate the tasks:

Code of task	Task identification	Level of efforts (hours)
-	TFG	540
A	Information review and research	78
A.1	Review railway-induced vibration problem with moving load	35
A.2	Review how to introduce the building in the system	25
A.3	Genetic algorithm programming research	18
B	Identification of the problem	30
C	Theoretical background	135
C.1	Moving load theoretical description	35
C.2	Inclusion of the building as a part of the system	35
C.3	Description of the dynamic response of the train	65
D	Optimization process	297
D.1	Formulation of the problem	65
D.2	Study of the performance of each part of the algorithm	122
D.2.1	Implementation of the algorithm	42
D.2.2	Study the effects of mutation strategy	20
D.2.3	Study the effects of crossover strategy	20
D.2.4	Study the effects of elitism strategy	20
D.2.5	Study of the improvement of H_UnitaryExternal Force_E1.m	20
D.3	Global case	110
D.3.1	Implementation of the algorithm	50
D.3.2	Final refinement of the genetic algorithm	45
D.3.3	Validation of the algorithm	15

Table D.4: Estimated time to develop the tasks of the future study.

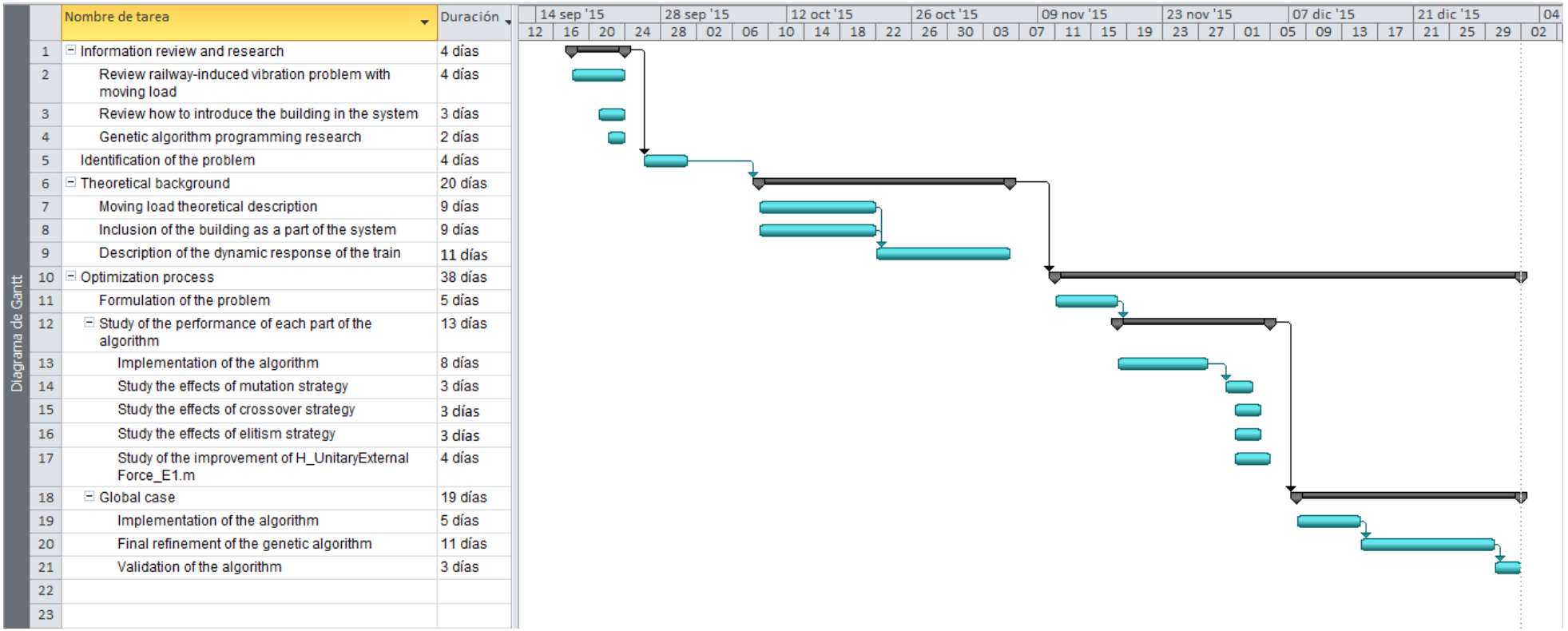

Figure D.2: Gantt chart of the future study. It has been considered that it starts on September 18th and finishes on January 1st.