Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Enginyeria en Tecnologies Industrials

Títol: Creació d'un software d'anàlisi acústica d'instruments musicals

Document: Memòria i annexos

Alumne: Adrià Vila Espígol

Tutor: Daniel Trias Mansilla
Departament: Enginyeria mecànica i de la construcció industrial
Àrea: Enginyeria mecànica

Convocatòria (mes/any) Juny del 2019

ÍNDEX

ÍNDEX DE FIGURES

ÍNDEX DE TAULES

1 INTRODUCCIÓ

1.1 Antecedents

Aquest treball sorgeix de l'interès del grup de recerca AMADE en l'anàlisi acústica d'instruments musicals. La finalitat d'aquesta anàlisi és poder distingir si els instruments són de baixa o alta qualitat, a partir d'una gravació d'àudio que contingui diverses repeticions de diferents notes interpretades per l'instrument musical en qüestió. Per dur a terme aquesta anàlisi s'han d'analitzar les diferents propietats del so mitjançant espectres de freqüència i espectrogrames.

1.2 Objecte

Creació d'un software capaç de proporcionar la informació necessària perquè l'usuari pugui analitzar la qualitat d'un instrument. Se centrarà en guitarres clàssiques, però podria ser extensible a altres instruments. No donarà un resultat explícit.

1.3 Abast

L'abast del projecte seria el disseny d'un codi i una interfície gràfica que, a partir d'una gravació de repeticions de diferents notes interpretades per un instrument, primer en faciliti el post procés obtenint arxius individuals de cadascuna de les repeticions i, després, post-processi la informació continguda en aquests arxius generant espectres de potència, espectrogrames i qualsevol altre tipus de dada temporal o freqüencial que es pugui creure rellevant.

Es pretén que el software resultant sigui fàcil d'utilitzar i que no requereixi grans coneixements de programació per a ser utilitzat, per tal que es pugui a dur a terme l'anàlisi amb comoditat.

Aquest treball se centrarà únicament en el disseny del programa i no en el seu ús ni la comparació d'instruments de diferents gammes.

2 MARC TEÒRIC

2.1 El so

En aquest apartat s'exposen les idees principals sobre el so, segons el llibre *Fundamentals of Music Processing* de *Meinard Müller*.

2.1.1 Ones i formes d'ona

Un so es genera quan algun objecte vibra i, per tant, provoca el desplaçament i l'oscil·lació de les molècules de l'aire. Això fa que es generin regions on hi ha compressió o rarefacció[1]. Aquesta pressió alternant viatja a través de l'aire en forma d'ona des de l'emissor fins al receptor, on pot ser percebuda en forma de so, si el receptor és humà, o pot ser convertida en senyal elèctric, si el receptor és un micròfon. Gràficament, el canvi en la pressió de l'aire en una posició concreta es pot representar mitjançant un gràfic pressió-temps, el qual també s'utilitza per representar la forma d'ona del so. Com podem veure a la Figura 1, la forma d'ona mostra gràficament la desviació de la pressió de l'aire respecte a la pressió mitjana d'aquesta.

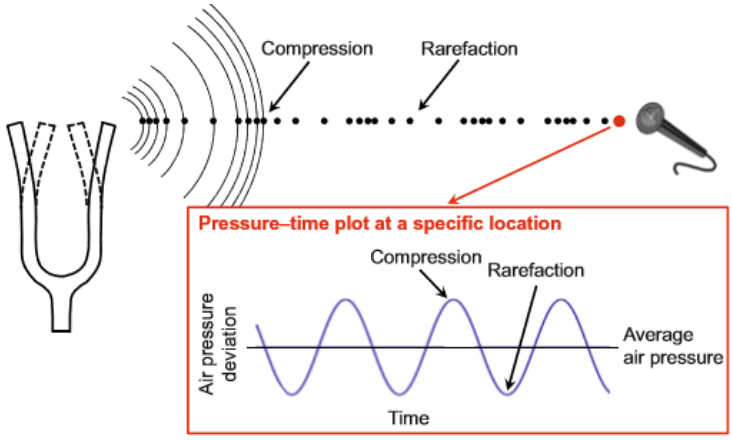

Figura 1: La forma d'ona mostra la desviació al llarg del temps de la pressió de l'aire respecte a la mitjana d'aquesta (Müller, 2015).

En termes generals, una ona mecànica es pot descriure com a una oscil·lació que viatja a través de l'espai, on l'energia és transferida d'un punt a un altre. Això fa que quan l'ona es mou a través d'un medi, la substància d'aquest es deformi temporalment. Per tant, podem dir que

[1] En una ona sonora la rarefacció es refereix a aquelles zones on la pressió és més baixa. És el fenòmen oposat a la compressió.

les ones sonores es propaguen a través de la col·lisió entre les molècules d'aire del so i les que estan al seu voltant. Després d'aquesta col·lisió s'allunyen entre elles, la qual cosa dóna lloc a una força restauradora. Això impedeix que les molècules se segueixin propagant en la mateixa direcció i, per tant, oscil·lin al voltant d'aquesta. Una ona pot ser transversal o longitudinal, depenent de la direcció de la corresponent oscil·lació. Les ones transversals es produeixen quan una pertorbació crea oscil·lacions perpendiculars a la direcció de propagació, per exemple, una vibració a una corda. Les ones longitudinals es produeixen quan les oscil·lacions són paral·leles a la direcció de propagació, per exemple, una molla o una ona sonora. Una ona transversal pot generar una ona longitudinal i viceversa. Per exemple, quan la corda d'un instrument està vibrant, és a dir, oscil·lant entre dos punts fixos, al mateix temps està emetent la seva energia a l'aire generant una ona sonora longitudinal. Si aquesta ona arriba al timpà, es torna a generar una ona transversal a partir d'una ona sonora longitudinal.

2.1.2 La freqüència i el to

Com ja hem pogut veure, una ona sonora es pot representar com una forma d'ona. Si els punts superiors i inferiors de pressió de l'aire es repeteixen de forma regular, s'obté una forma d'ona periòdica. En aquest cas, el període de la forma d'ona és el temps necessari per completar un cicle. La Figura 2 ens mostra una sinusoide, la qual és la forma d'ona periòdica més simple. Una sinusoide es defineix per la seva freqüència, la seva amplitud (desviació del pic respecte a la seva mitjana) i la seva fase (valor que té la sinusoide al principi de cada cicle, és a dir, quan el temps és 0). Aquests tres atributs són molt importants a l'hora d'analitzar una sinusoide.

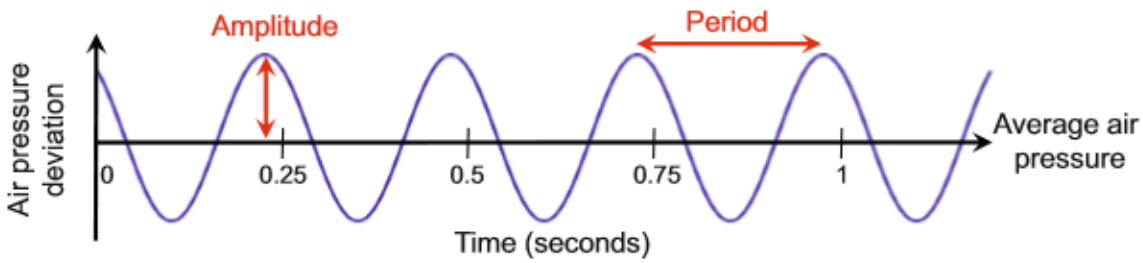

Figura 2**:** Forma d'ona d'una sinusoide de freqüència igual a 4 Hz (Müller, 2015).

Com més alta és la freqüència en una ona sinusoidal, més alta és la nota. El rang audible pels humans va de 20 Hz fins a 20 kHz.

La sinusoide es pot considerar el prototip de la interpretació acústica d'una nota musical. A vegades, el so resultant d'una sinusoide s'anomena so harmònic o to pur, per tant, podem dir que la freqüència està molt relacionada amb el to de cada so. En general, el to és un atribut subjectiu del so. En el cas dels sons complexos, la relació que tenen amb la freqüència pot ser especialment ambigua, en canvi, en els tons purs, la relació entre la freqüència i el to és molt clara. Per exemple, una sinusoide amb una freqüència de 440 Hz correspon a la nota A4. Aquest to, en concret, és conegut com a to de concert, ja que és el que s'utilitza com a referència quan un grup d'instruments musicals ha d'afinar per dur a terme una actuació. Una petita diferència a la freqüència no necessàriament comporta un canvi perceptible, ja que s'acostuma a associar un rang enter de freqüències a un to determinat.

Si dues freqüències difereixen per un valor igual a una potència de 2, aleshores la percepció de les dues notes és similar. Això passa perquè les notes són iguals, però una és una octava més alta que l'altra. Per exemple, els tons A3 (220 Hz), A4 (440 Hz) i A5 (880 Hz) tenen un so similar. A més, la distància entre A3 i A4 és la mateixa que la distància entre A4 i A5. En altres paraules, podem dir que la percepció humana del to és logarítmica.

Com ja sabem, les notes estan separades per semitons, excepte el E i el F i el B i el C. El concepte semitò incorpora una nova unitat de mesura logarítmica anomenada cent, la qual s'utilitza per intervals musicals. Per definició, una octava està dividida entre 1200 cents, per tant, cada semitò correspon a 100 cents. La diferència en cents entre dues freqüències ω_1 i ω_2 ve donada per

$$\log_2\left(\frac{\omega_1}{\omega_2}\right) \cdot 1200 \tag{1}$$

La diferència de cents mínima perceptible per una persona normal és de 25 cents i per una persona entrenada és de 10 cents. A l'hora d'identificar les notes, el codi d'aquest projecte

realitza una aproximació entre la freqüència corresponent a la nota en qüestió i la taula de freqüències. La freqüència més pròxima és la que es pren com a bona.

En el món real, els sons estan lluny de ser tons purs amb una freqüència ben definida. Quan es toca una nota amb un instrument, pot ser que el so resultant sigui complex, la qual cosa significa que el so tindrà una barreja de diferents freqüències variant al llarg del temps. Per tant, un to musical es pot descriure com una superposició de tons purs o sinusoides, cadascuna amb la seva pròpia freqüència de vibració, amplitud i fase. Aquests tons purs que componen el so complex s'anomenen sons parcials. Un harmònic és un so parcial, el qual és un múltiple enter de la freqüència fonamental. La freqüència més baixa de cada so parcial és la freqüència fonamental del so en qüestió, és a dir, la freqüència corresponent al to de la nota. Per tant, la freqüència fonamental, a part de ser el primer parcial, també és el primer harmònic d'una nota musical. El terme inharmònic s'utilitza per denotar la desviació d'un parcial respecte l'harmònic més proper.

La majoria dels instruments afinats estan dissenyats perquè els parcials siguin molt pròxims als harmònics, la qual cosa implica que tinguin molt poca inharmonia. Això fa que als parcials d'aquests instruments se'ls anomeni directament harmònics. Un exemple de so harmònic el podem trobar a la Figura 3 (b), la qual ens mostra que el so és quasi periòdic. La forma d'ona compresa en aquests 500 ms correspon a la nota D durant l'etapa de caiguda. Si entrem en detall, podem contar 37 períodes, els quals ens indiquen que la freqüència fonamental d'aquesta secció és de 74 Hz, corresponent a la nota D2.

$$f = \frac{1}{T} \cdot n\ períodes = \frac{1}{0.5} \cdot 37 = 74Hz \tag{2}$$

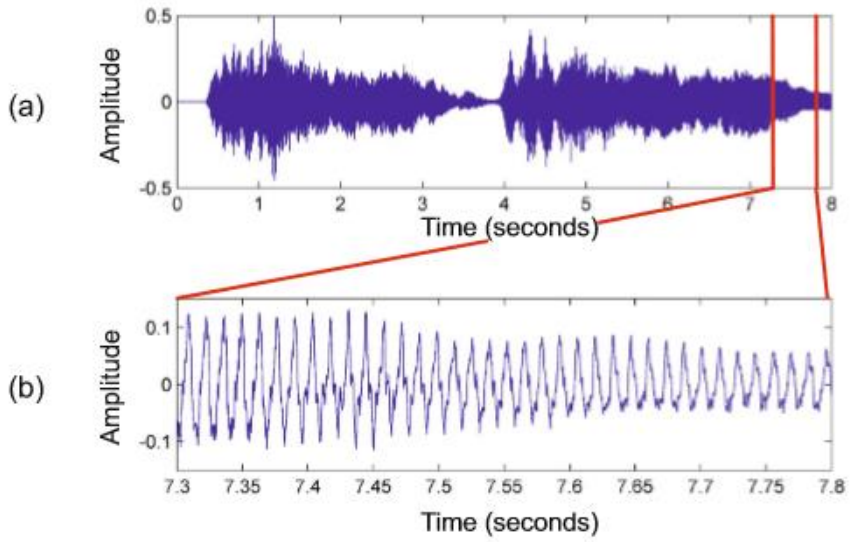

Figura 3: (a) Forma d'ona dels primers 8 segons d'una peça musical. (b) Zoom de la secció compresa entre els segons 7.3 i 7.8 (Müller, 2015).

2.1.3 Harmònics

Per explicar què són els harmònics, s'utilitzarà la teoria explicada a la pàgina web *The Physics Classroom*, la qual, per dur a terme una explicació visual, se centra en la corda d'una guitarra vibrant a la seva freqüència natural. Com que els seus extrems estan fixos a causa de l'estructura de la guitarra, no es poden moure i, per això, aquests passen a ser nodes[2]. Com que la corda està vibrant, entre aquests dos nodes hi ha d'haver almenys un antinode[3]. La freqüència fonamental de la corda d'una guitarra en vibració, com podem veure a la Figura 4, és l'harmònic associat a una ona que només conté un antinode posicionat entre dos nodes situats a l'extrem de la corda. Aquest serà l'harmònic amb la longitud d'ona més llarga i la freqüència més baixa.

La freqüència més baixa produïda per qualsevol instrument s'anomena freqüència fonamental, la qual equival al primer harmònic. Podem veure a la Figura 4 que la corda no forma una ona completa. Una ona completa comença en posició de repòs (node), puja fins a

[2] En un cos en vibració, punt que no vibra.
[3] Punt d'un cos vibrant, situat entre dos nodes adjacents i equidistants d'aquests, en què l'amplitud de la vibració és màxima.

una amplitud màxima (antinode), torna al repòs (node), baixa fins a una amplitud mínima (antinode) i torna al repòs (node) que és on comença el següent cicle.

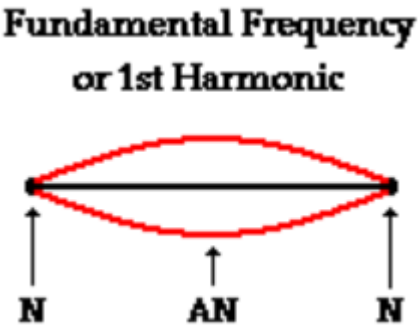

Figura 4: Freqüència fonamental o primer harmònic 2 (The Physics Classroom, 2019).

Per tant, com podem veure a la Figura 5, la freqüència fonamental només conté una meitat de l'ona completa, és a dir que la longitud d'ona és el doble de la longitud de la corda.

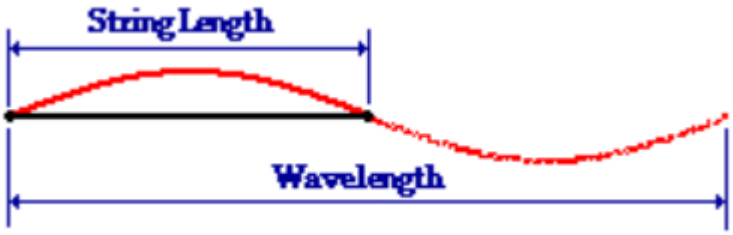

Figura 5: Pel primer harmònic de la corda d'una guitarra, la longitud d'ona és el doble de la llargada de la corda (The Physics Classroom, 2019).

El segon harmònic es produeix afegint 1 node a la meitat dels dos que estan situats als extrems de la corda, la qual cosa fa que s'hagi d'afegir un nou antinode, per tal de mantenir el patró de nodes i antinodes. Per tant, el segon harmònic conté 3 nodes i 2 antinodes, el patró del qual podem veure a la Figura 6. També es pot veure que el segon harmònic conté una ona completa, la qual cosa indica que la longitud de la corda és igual a la longitud de l'ona.

Figura 6: Segon harmònic (The Physics Classroom, 2019).

El tercer harmònic es produeix afegint 2 nodes i 2 antinodes entre els extrems, els quals, per crear un patró regular i repetitiu, han d'estar a una distància equidistant entre ells. Per tant, el tercer harmònic conté 4 nodes i 3 antinodes, tal com podem veure a la Figura 7.

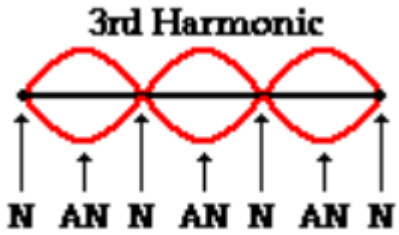

Figura 7: Tercer harmònic (The Physics Classroom, 2019).

També es pot veure que el tercer harmònic conté una ona completa i mitja més al llarg de la longitud de la corda. Per aquest motiu, i com podem veure a la , podem afirmar que la longitud de la corda és igual a una longitud d'ona i mitja.

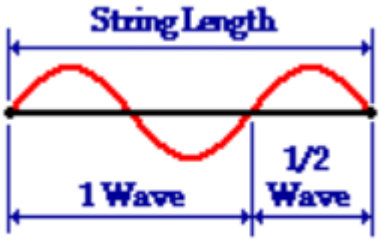

Figura 8: En el tercer harmònic, la longitud de la corda conté una ona completa i mitja més (The Physics Classroom, 2019).

Després d'haver analitzat el patró dels tres primers harmònics, podem reconèixer el patró general que se segueix. Per cada harmònic nou s'afegeixen un node, un antinode i mitja ona

més al llarg de la corda. Si es coneix el nombre d'ones d'una corda, aleshores l'equació que relaciona la longitud d'ona (λ) i la longitud de la corda (L) és la següent:

$$\lambda = \frac{2}{n} \cdot L, \text{ on n és el número d'harmònic} \tag{3}$$

2.1.4 Dinàmica, intensitat i sonoritat

Una propietat també important de la música és la dinàmica, la qual s'utilitza per referir-se al volum del so i als símbols musicals que indiquen el volum. En l'àmbit de l'àudio, però, la dinàmica està relacionada amb una propietat perceptiva anomenada sonoritat o volum, la qual ordena els sons de fluix a fort. De forma similar a la relació entre el to i la freqüència, la sonoritat és una mesura subjectiva que està correlacionada amb mesures objectives d'intensitat sonora i potència sonora. No obstant això, la sonoritat també depèn d'altres característiques del so com la durada o la freqüència.

Des d'un punt de vista físic, és complicat definir la intensitat o la potència sonora. La potència és la mesura amb la qual l'energia és transferida, utilitzada o transformada. La unitat de mesura és el watt (W), el qual es defineix com a joules per segon. De la mateixa manera, la potència sonora expressa quina quantitat d'energia per unitat de temps és emesa per una font sonora a través de l'aire per totes les direccions. El terme intensitat sonora s'utilitza per denotar la potència sonora per unitat d'àrea.

A la pràctica, la potència sonora i la intensitat sonora tenen uns valors, els quals, tot i ser molt petits, són significants pels humans. Per exemple, el llindar d'audició (TOH), és a dir, la mínima intensitat sonora d'un to pur que un humà pot sentir, és de $I_{TOH} \coloneqq 10^{-12}\ W/m^2$ i llindar de dolor (TOP), és a dir, la màxima intensitat sonora, és de $I_{TOP} \coloneqq 10\ W/m^2$. Per raons pràctiques, per expressar potència i intensitat, s'utilitzen l'escala de decibels (dB). El decibel és una unitat logarítmica que expressa la relació entre dos valors, on un dels dos valors serveix de referència, com el llindar I_{TOH} en el cas de la intensitat sonora. Aleshores, la intensitat mesurada en decibels es defineix de la següent manera:

$$dB(I) := 10 \cdot \log_{10}\left(\frac{I}{I_{TOH}}\right) \tag{4}$$

Amb aquesta definició s'obté que $dB(I_{TOH}) = 0$ i, si es dobla la intensitat, s'obté que $dB(2 \cdot I) = 10 \cdot \log_{10}(2) + dB(I) \approx 3\ dB + dB(I)$.

Quan s'especifiquen valors d'intensitat en decibels, es parla de nivells d'intensitat. La Taula 1 mostra valors típics d'intensitat sonora. Podem veure que les notes que s'han tocat dins una dinàmica *pianissimo* tenen un nivell d'intensitat de 40 dB, mentre que les que s'han tocat *fortissimo* poden arribar a 100 dB.

Taula 1: Valors típics d'intensitat en W/m² (intensitat), dB (nivell d'intensitat) i per un factor de comparació amb el TOH (Müller, 2015).

Source	Intensity	Intensity level	× TOH
Threshold of hearing (TOH)	10^{-12}	0 dB	1
Whisper	10^{-10}	20 dB	10^{2}
Pianissimo	10^{-8}	40 dB	10^{4}
Normal conversation	10^{-6}	60 dB	10^{6}
Fortissimo	10^{-2}	100 dB	10^{10}
Threshold of pain	10	130 dB	10^{13}
Jet take-off	10^{2}	140 dB	10^{14}
Instant perforation of eardrum	10^{4}	160 dB	10^{16}

Com s'ha mencionat anteriorment, el concepte de sonoritat està correlacionat amb la intensitat sonora i depèn de diferents factors. En primer lloc, cada persona pot percebre un so amb una sonoritat diferent, per exemple, dues persones de diferent edat. Un altre factor és la duració del so, la qual també afecta la seva percepció, ja que el sistema auditiu humà regula l'efecte de la intensitat sonora al llarg d'un interval de fins a 1 segon. Per tant, un humà té la sensació que un so que ha durat 200 ms és més fort que un so similar que ha durat 50 ms. A més, dos sons amb una mateixa intensitat, però diferents freqüències, no són percebuts amb la mateixa sonoritat. Els humans amb una capacitat auditiva normal, tenen més sensibilitat amb sons de 2 kHz fins a 4 kHz, la qual decreix tant per les freqüències més baixes com per les més altes.

2.1.5 Timbre

A més del to, la sonoritat i la durada, hi ha un altre aspecte del so anomenat timbre que és fonamental. El timbre permet al receptor distingir el to musical d'un violí, un oboè o una trompeta, encara que la nota tocada tingui el mateix to i la mateixa sonoritat. Com el to i la sonoritat, el timbre és una propietat perceptiva del so. No obstant això, la comprensió del concepte timbre és complicada i, degut a la falta de claredat, normalment es descriu d'una forma indirecta: el timbre és un atribut, el qual permet a l'oient diferenciar dos sons sense utilitzar cap altre criteri que el to, la sonoritat i la duració. Per exemple, la informació del timbre ens permet distingir els sons produïts per un oboè i un violí, encara que el to i la sonoritat d'aquests siguin idèntics. El so d'un instrument musical, es pot descriure amb paraules com clar, fosc, càlid, etc. Hi ha investigadors, però, que han provat d'enfocar la definició del timbre buscant correlacions amb característiques sonores més objectives, com l'evolució temporal i espectral, l'absència o presència de components tonals i sorollosos, o la distribució de l'energia al llarg dels parcials d'un to.

Quan es toca la tecla d'un piano, el so resultant és molt més que una superposició de sinusoides pures que corresponen a la freqüència fonamental i als seus sobretons. Pel sol fet de tocar una nota, es produeix una barreja de so amb unes característiques que varien constantment al llarg del temps, les quals poden contenir tant components periòdics com components no periòdics.

Totes les notes estan compostes per 4 fases: la fase d'atac, la fase de caiguda, la fase de sosteniment i la fase d'extinció, les quals formen l'envolvent ADSR[4]. A l'inici de cada nota normalment hi ha un increment sobtat d'energia. En aquesta fase, anomenada fase d'atac de la nota, és on es comença a formar el so. Conté una alta quantitat de components no periòdics repartits al llarg de tot el rang de freqüències, la qual cosa és una propietat també inherent en el soroll. En l'acústica, un so similar al soroll de curta durada i amb una amplitud alta, el qual ocórrer al principi d'una forma d'ona també s'anomena transitori. Després de la fase d'atac

[4] La sigla ADSR correspon a l'anglès *attack decay sustain release*.

entra a la fase de caiguda, on el so de la nota musical s'estabilitza i arriba a una fase, més o menys, estable amb un patró periòdic. La tercera fase s'anomena fase de sosteniment i està composta per la major part de la nota. És on l'energia es manté, més o menys, constant o decreix lleugerament, depenent de l'instrument. La fase final s'anomena fase d'extinció i és on la nota s'apaga.

Intuïtivament, l'envolvent d'una forma d'ona pot ser considerat com una corba suau que ressegueix els seus extrems d'amplitud, com es pot veure a la Figura 9 (a). Les diferents fases mencionades anteriorment estan directament relacionades amb la forma de l'envolvent de la nota analitzada. En la síntesi del so, l'envolvent que s'ha de generar del senyal tractat segueix el model ADSR, com es pot veure a la Figura 9 (b). Les duracions relatives i les amplituds de les 4 fases tenen un impacte significatiu sobre com sonarà la nota sintetitzada.

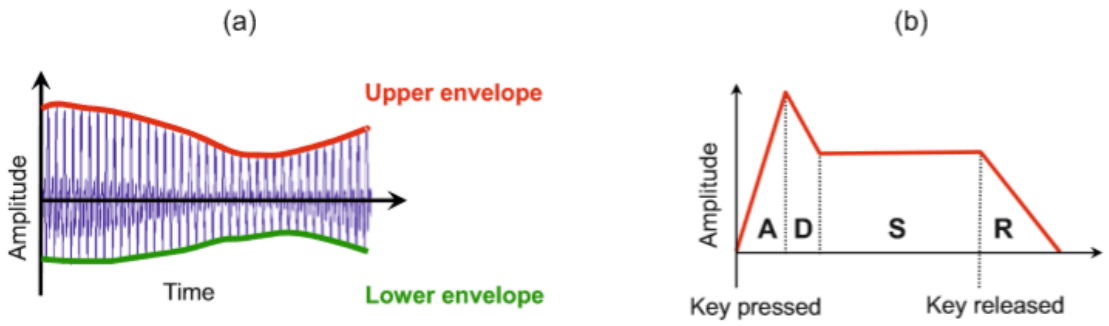

Figura 9: (a) Envolvent del senyal. (b) Vista esquemàtica de l'envolvent ADSR (Müller, 2015).

El model ADSR és un model que permet simplificar molt la forma dels senyals i només proporciona una aproximació significativa per envolvents d'amplitud de notes generades per certs instrumentes. Per exemple, a la Figura 10 (a), la nota tocada té un envolvent similar al model ADSR. Com podem veure, després d'una fase d'atac curta i intensa, moment en el qual el martell pica la corda o, en el cas de la guitarra clàssica, moment en el qual l'usuari fa vibrar la corda, i d'una fase de caiguda estabilitzant, la nota s'apaga. En el cas del piano, el decreixement de la intensitat sonora és molt lent i comença a partir del moment en el qual el martell deixa de tocar la corda. Aquesta fase es pot identificar com a fase de sosteniment. Però, per altres tipus d'instruments, com el que podem veure a la Figura 10 (b), l'amplitud evoluciona

de forma totalment diferent. En primer lloc, a la fase d'atac, la nota es toca suaument amb un increment gradual de volum. A continuació, però, no es veu cap fase de caiguda, sinó que comença la fase de sosteniment, la qual no és estable però oscil·la de forma regular. La fase d'extinció comença quan l'usuari deixa de tocar la corda amb la vara i, aleshores, el so s'apaga ràpidament.

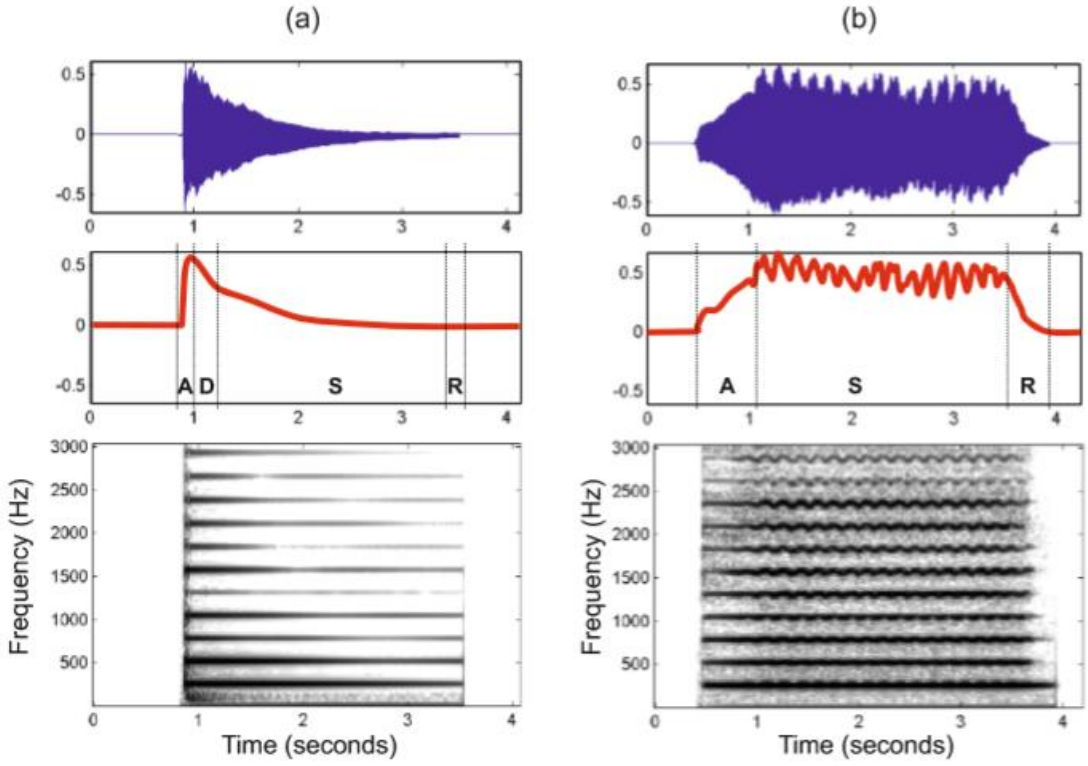

Figura 10: Forma d'ona, envolvent d'amplitud i espectrograma de diferents instruments tocant la mateixa nota C4 (261.6 Hz). (a) Piano. (b) Violí (Müller, 2015).

En el cas de l'exemple del violí, podem veure que experimenta variacions periòdiques de l'amplitud. Aquest fenomen s'anomena *tremolo* i es genera depenent de l'estil d'interpretació d'instruments tant de corda com de vent. L'efecte del *tremolo* normalment va lligat al *vibrato*, el qual és un efecte musical que consisteix a realitzar canvis de freqüència regulars i controlats. En termes tècnics, el *tremolo* correspon a una modulació de l'amplitud, mentre que el *vibrato* correspon a una modulació de la freqüència. Ambdós depenen de dos paràmetres: la quantitat de vibració i la taxa, amb la qual l'amplitud o la freqüència varien. Tot i que aquests dos efectes provoquen simples canvis concrets en la intensitat i la freqüència, no necessàriament

provoquen canvis perceptibles en la sonoritat o el to de la nota musical. En canvi, sí que són característiques que influencien el timbre d'una nota musical.

Com s'ha mencionat anteriorment, definir el timbre és molt complicat, però podríem dir que la propietat més important per caracteritzar el timbre és l'existència de certs parcials i les seves respectives amplituds. Ja sabem que els parcials són les freqüències dominants de cada nota i que el parcial més baix correspon a la freqüència fonamental, i que la inharmonia expressa la mesura en què cada parcial es desvia respecte a l'harmònic ideal més proper. Aleshores, podem dir que pels sons harmònics, per exemple una nota musical, la qual té una freqüència clarament perceptible, la majoria de parcials són molt propers als corresponents harmònics. De totes maneres, no tots els parcials apareixen amb el mateix valor d'amplitud.

La composició d'un so, centrant-nos en els seus parcials, es pot visualitzar mitjançant un espectrograma, el qual mostra la intensitat de les freqüències que conté el so al llarg del temps. A la Figura 10 (a), a la part baixa de la imatge, es pot veure l'exemple de l'espectrograma de la nota C4 interpretada per un piano, on l'evolució de la intensitat s'expressa mitjançant la tonalitat grisa, la qual com més fosca sigui, més intensa serà la intensitat en aquell punt. Les línies horitzontals ens indiquen tant la freqüència fonamental, corresponent a 261.6 Hz, com els corresponents harmònics, múltiples enters de 261.6 Hz. La caiguda de la nota queda reflectida per la caiguda de cada parcial. Es pot veure també que la majoria de l'energia es concentra als parcials més baixos, mentre que als parcials més alts l'energia tendeix a ser inferior. Aquesta distribució és típica per la majoria d'instruments.

Pel que fa a alguns instruments de corda, com el violí que podem veure a la Figura 10 (b), el so tendeix a tenir molts parcials i la majoria d'energia es concentra als més alts. En aquest cas, l'espectrograma també mostra oscil·lacions regulars dels parcials al llarg del temps, les quals indiquen el *vibrato* de l'instrument. També, per a aquesta família d'instruments, en certs casos, es poden mesurar desviacions substancials entre els parcials més alts i els harmònics corresponents. Com menys elàstica és una corda, és a dir, com més tensada està, més inharmonia exhibeix.

En resum, podem dir que el timbre és un fenomen multidimensional, el qual és difícil de mesurar. El que fa que cada instrument tingui el seu propi so, són les corresponents irregularitats i variacions.

2.2 Processament digital del senyal

La informació dels següents apartats s'ha extret de la sèrie de vídeos que la *Universidad Miguel Hernández de Elche* va publicar l'any a la plataforma *YouTube* sobre el tractament del senyal.

2.2.1 Els senyals

Els senyals són funcions que proporcionen informació sobre el comportament o atributs d'algun fenomen, per tant, podem dir que són portadors d'informació. En altres paraules, són el registre d'una quantitat física que varia respecte del temps, l'espai o alguna altra variable independent, la qual conté informació sobre el fenomen en qüestió. Els senyals, però, no només contenen informació útil, sinó que estan compostos per soroll i interferències també. El soroll i les interferències són els factors que contaminen la informació que ells mateixos contenen. Extreure únicament la informació útil és una tasca important dins el processament del senyal.

El soroll, el qual sempre està present en un senyal, pot provenir de tres fonts diferents:

- Electrònica: agitació tèrmica natural dels electrons, contactes defectuosos, soldadures mal fetes, etc.
- Ambiental: còsmic, atmosfèric, fonts externes, etc.
- Digital: a causa de la quantificació.
    - Pel que fa als senyals electrònics, els sorolls tèrmics i de quantificació sempre hi són presents.

Les interferències es poden predir, ja que estan molt relacionades amb el senyal que s'està tractant. Pot ser que en alguns casos no hi estiguin presents. Solen ser d'origen artificial i de la mateixa naturalesa que el senyal, per això, normalment s'identifiquen com a sorolls coherents. Però això també fa que sigui molt difícil distingir-les i, per tant, molt difícil separar-les. Hi ha dos camps que generen diferents tipus d'interferències:

- Comunicacions: electromagnètiques, diafonia, canal adjacent, etc.

- Propagació de les ones: com que actualment hi ha molts dispositius, pròxims entre ells, que emeten senyal, part de l'energia d'una banda passa a l'altra, aleshores sorgeix la superposició d'ones electromagnètiques, òptiques o mecàniques.

La mesura de la influència que tenen els sorolls i les interferències sobre els senyals es calcula segons quins aspectes es volen comparar. Amb aquests paràmetres de mesura es defineix la qualitat d'un sistema de transmissió, comunicació, emmagatzematge, etc. Com més gran és la relació entre el senyal i el soroll i/o interferència, millor és el sistema.

- SNR (relació entre el senyal i el soroll). És la relació més habitual. Relaciona en unitats logarítmiques la potència del senyal respecte de la potència del soroll.

$$SNR = \frac{Potència\ mitjana\ del\ senyal}{Potència\ mitjana\ del\ soroll} = 10 \cdot \log_{10} \frac{P_s}{P_n} \tag{5}$$

- SIR (relació entre el senyal i la interferència). Relaciona en unitats logarítmiques la potència del senyal respecte de la potència de la interferència.

$$SNR = \frac{Potència\ mitjana\ del\ senyal}{Potència\ mitjana\ de\ la\ interferència} = 10 \cdot \log_{10} \frac{P_s}{P_i} \tag{6}$$

- SINR (relació entre el senyal, el soroll i la interferència). Relaciona en unitats logarítmiques la potència del senyal respecte de les potències del soroll i la interferència.

$$SNR = \frac{Potència\ mitjana\ del\ senyal}{Potència\ mitjana\ del\ soroll + interferència} = 10 \cdot \log_{10} \frac{P_s}{P_n + P_i} \tag{7}$$

2.2.2 Àrees de tractament del senyal

Pel que fa a les àrees de tractament del senyal, clàssicament s'han classificat en 4: filtratge, equalització, identificació i estimació i detecció. De totes maneres, per començar a tractar el senyal és molt important dur a terme l'anàlisi d'aquest, ja que si no es coneixen les propietats del senyal ni la seva naturalesa és molt difícil dissenyar algoritmes i seleccionar tècniques de processament. En aquest projecte només s'hi aplica la part de filtratge i podria aplicar-se també la part d'equalització, si l'objectiu fos escoltar els àudios.

- Filtratge. S'utilitza per eliminar sorolls o per donar o treure èmfasi en alguna part del senyal. Per exemple, si tenim un senyal corresponent a un electrocardiograma, el qual conté molt de soroll i d'harmònics procedents de la línia d'alimentació, es pot aplicar un filtre de freqüència que suavitzarà els sorolls no desitjats i, d'aquesta manera, es podran veure millor els batecs del cor. O si tenim un senyal corresponent a una sèrie temporal de temperatures, el qual té una variabilitat elevada, però com que interessa veure l'evolució d'aquestes en un interval llarg de temps, és a dir, amb una variabilitat lenta, es pot aplicar un filtre en el temps.

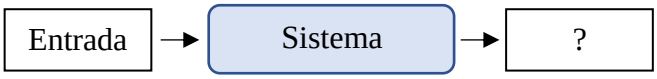

Figura 11: El filtratge ens permet variar la sortida.

- Equalització. Tot senyal emès passa per un sistema, el qual el distorsiona més o menys. Per tant, el senyal que percebem pot arribar a ser bastant diferent que l'original. L'objectiu de l'equalització és sentir aquest senyal com si estiguéssim en el lloc on s'ha generat, per exemple, a una sala de gravació. Per exemple, una equalització d'àudio té la finalitat de modificar el senyal perquè soni igual que a la sala de gravació. Si l'àudio s'escolta en una sala plena de mobles, sonarà diferent que si s'escolta en un descampat.

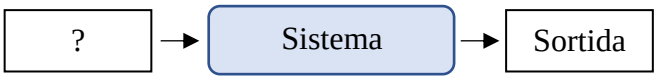

Figura 12: L'equalització permet que el senyal de sortida sigui igual al d'entrada.

- Identificació. L'objectiu és identificar les propietats del sistema pel qual es propagarà el senyal. Per exemple, volem conèixer les propietats d'un terreny. Per fer-ho, col·loquem un seguit de sensors per captar els diferents rebots generats per les explosions i, en funció de la intensitat d'aquest, es podrà fer un mapa del terreny.

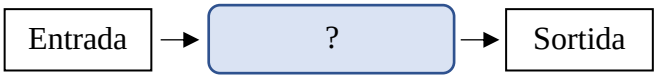

Figura 13: La identificació ens permet descriure el medi per on es propaga el senyal, a partir de l'entrada i la sortida.

- Estimació i detecció. Aquestes àrees són més avançades perquè requereixen un coneixement més profund d'estadística i senyals. La part d'estimació intenta extreure informació del senyal mitjançant l'anàlisi estadístic d'aquest. Per exemple, per dur a terme un reconeixement de patrons, un reconeixement de cares o un modelatge d'electroencefalogrames (EEG). La part de detecció, intenta detectar si ha passat quelcom al senyal. Per dur a terme això, però, s'ha de conèixer el senyal a la perfecció. Per exemple, si algú ha polsat una tecla, si s'ha emès algun so, si hi ha algun objectiu al radar.

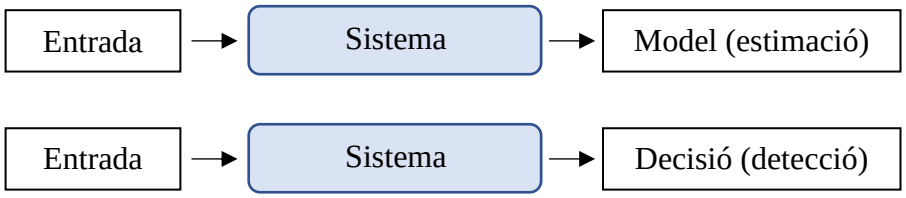

Figura 14: Estimació i detecció.

Últimament, s'està començant a utilitzar una àrea anomenada intel·ligència artificial, la qual és una combinació de l'àrea d'estimació i detecció, juntament amb estadística.

2.2.3 Classificació dels senyals

Els senyals es poden classificar a partir de la seva naturalesa o a partir de com han estat adquirits. Es representen mitjançant una variable independent, normalment el temps, situada a l'eix d'abscisses i una variable dependent, anomenada amplitud, situada a l'eix d'ordenades. Hi ha 4 formes de representar-los, les quals depenen de la seva classificació:

- Senyals continus. El temps i l'amplitud són variables contínues, és a dir que poden prendre qualsevol valor. Això fa que la quantitat d'informació sigui molt gran i, per tant, que siguin senyals poc manejables. Els sistemes analògics treballen amb aquest tipus de senyals.

- Senyals quantificats. El temps és una variable contínua i l'amplitud és una variable discreta, és a dir que l'amplitud només pot prendre valors determinats i el temps qualsevol valor.
- Senyals discrets. El temps és una variable discreta i l'amplitud és una variable contínua, és a dir que només existeix senyal en uns instants determinats de temps per un valor qualsevol d'amplitud.
- Senyals digitals. El senyal està quantificat i discretitzat. El temps i l'amplitud són variables discretes, és a dir que tant el temps com l'amplitud només poden prendre valors determinats. En aquest cas, el senyal resultant és molt més manejable que quan és digital perquè la quantitat de memòria i capacitat de processament necessàries són molt més reduïdes. Per això, en els dispositius digitals es treballa amb aquest tipus de senyals.

Els senyals també es poden classificar segons la seva predictibilitat i poden ser deterministes i aleatòries.

- Deterministes. En qualsevol instant de temps el valor de l'amplitud està determinat. El senyal està regit per una equació. Per exemple, una ona sinusoidal.
- Aleatòries. El valor de l'amplitud és aleatori en qualsevol instant de temps. Moltes vegades són la combinació d'un senyal determinista i una fracció de soroll, la qual fa que el senyal resultant sigui impredictible.

2.2.4 El processament del senyal digital

La teoria explicada en aquest subapartat sobre el processament del senyal digital s'ha extret del llibre *DAFX: Digital Audio Effects* dels autors *Arfib, Keiler, Zölzer, Verfaille, & Bonada (2011)*.

El processament del senyal consisteix a convertir un senyal analògic en un senyal digital. Aquest procés es pot dur a terme mitjançant un convertidor analògic-digital (ADC), la resposta del qual és una seqüència de nombres corresponent al senyal digital. El ADC realitza un mostreig de les amplituds del senyal analògic en una graella equidistant al llarg de l'eix

temporal (horitzontal) i una quantificació que converteix les amplituds en mostres fixes, les quals es representen mitjançant nombres al llarg de l'eix d'amplituds (vertical). Cada mostra es representa mitjançant una línia vertical amb un punt a sobre. A la Figura 15 es pot veure aquest procés.

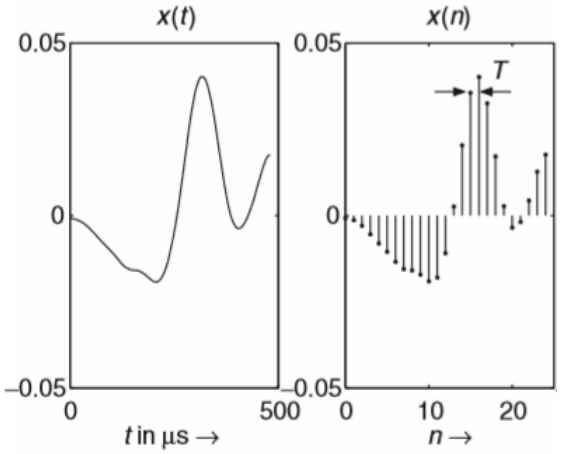

Figura 15: Senyal abans i després de passar per el ADC (Arfib et al., 2011).

El senyal analògic indica l'amplitud del senyal al llarg del temps, on tant l'amplitud com el temps són variables contínues, és a dir que poden prendre qualsevol valor. Quan aquest senyal es converteix en digital, l'amplitud del senyal passa a ser una seqüència de mostres representada per nombres al llarg del temps, on l'amplitud està quantificada i el temps és discret, és a dir que els valors que poden prendre les dues variables estan predeterminats. La distància temporal, en segons, entre dues mostres consecutives està definida pel període de mostreig. La freqüència de mostreig, la qual és la funció inversa del període, indica el nombre de mostres per segon en Hertz i, segons el teorema de mostreig de Nyquist-Shannon, el seu valor ha de ser com a mínim el doble de la freqüència màxima del senyal analògic $f_s > 2 \cdot f_{max}$. Per tant, si la freqüència de mostreig que hem d'utilitzar ha de ser fixa, abans de mostrejar el senyal, ens hem d'assegurar que l'amplada de banda màxima del senyal d'entrada sigui inferior a la meitat de la freqüència de mostreig. Si això no és possible, s'hauran de rebutjar les freqüències més altes filtrant-les amb un filtre passabaix, el qual només permeti passar les freqüències inferiors a la freqüència màxima.

La Figura 16 mostra diferents tipus representacions gràfiques que poden tenir els senyals digitals. La primera imatge mostra 8000 mostres, la segona 1000 i la tercera 100 d'un senyal d'àudio digital. Només si el nombre de mostres és suficientment baix, s'utilitzarà la representació de la línia vertical amb un punt a sobre.

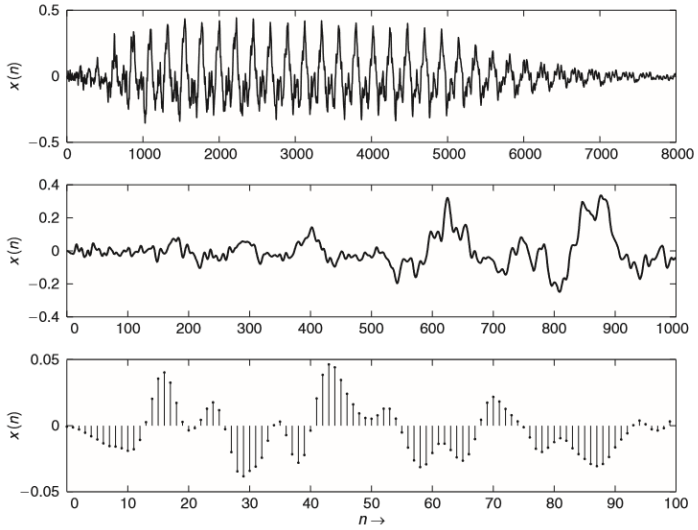

Figura 16: Diferents representacions temporals d'un mateix senyal (Arfib et al., 2011).

Existeixen dos formats d'escala per representar l'eix vertical dels senyals d'àudio digitals.

- La representació numèrica entera. Aquesta representació depèn del nombre de bits utilitzats per fer-la. En el cas de la Figura 17, podem veure que la quantificació de les amplituds va de -32768 fins a 32767. Aquesta quantificació està basada en una representació de 16 bits de les amplituds mostrejades, la qual està composta per 2^{16} valors quantificats compresos en un rang de -2^{15} fins a $2^{15}-1$. En general, per a representacions de w-bits, el rang de nombres va des de -2^{w-1} fins a $2^{w-1}-1$.
- La representació numèrica entera normalitzada. Si dividim tots els nombres enters de la representació numèrica entera pel valor absolut màxim, en el cas anterior 32768, aleshores l'escala vertical en qüestió queda normalitzada. És a dir que el rang es modifica i passa a ser de -1 fins a $1-Q$, on Q és la mida de la quantificació. Aquesta mida es pot calcular fent $Q = 2^{-(w-1)}$, la qual és de $Q = 3.0518 \cdot 10^{-5}$ per 16 bits.

Pel que fa al format de les escales horitzontals, les quals s'anomenen eixos de temps continus, també n'existeixen dos.

- Eix del temps discret.
- Eix del temps discret normalitzat. És el format que s'utilitza normalment.

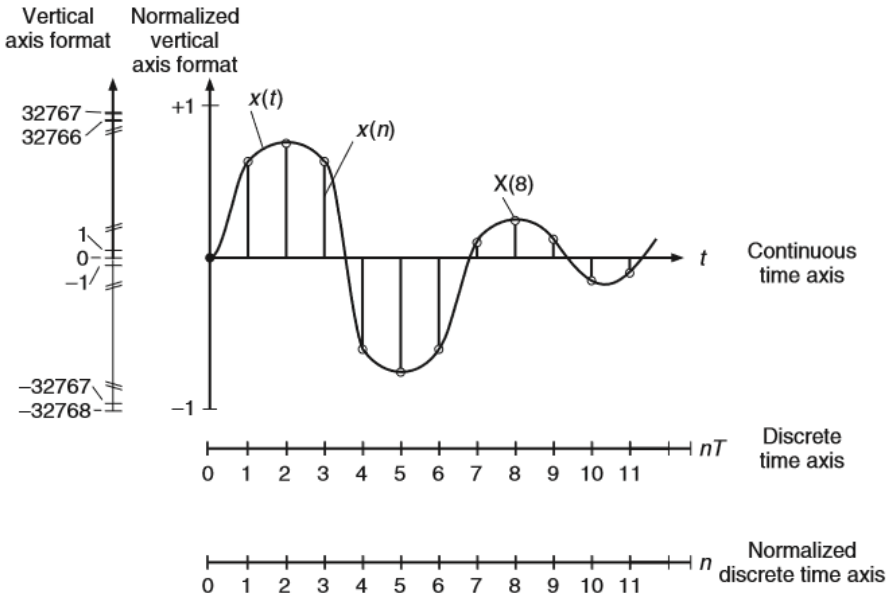

Figura 17: Formats d'escala vertical i horitzontal utilitzats per la representació de l'àudio digital (Arfib et al., 2011).

En resum, es pot definir un senyal digital com a un senyal compost per temps i amplitud discrets, els quals es formen mostrejant un senyal analògic i quantificant l'amplitud d'aquest a un nombre fix. El senyal digital es representa mitjançant una seqüència de números $x(n)$.

2.3 La transformada de Fourier

En aquest apartat s'introdueixen les idees principals de la transformada de Fourier, segons el llibre *Fundamentals of Music Processing* de *Meinard Müller (2015)*.

2.3.1 Conceptes bàsics

Per començar, analitzem el so d'una sola nota tocada amb un piano, la forma d'ona del qual està representada a la Figura 18 (a). Per saber quina és la nota que s'està tocant, hem de saber que el to d'una nota musical està molt relacionat amb la seva freqüència fonamental, és a dir, la freqüència més baixa. Per això, és necessari determinar el contingut freqüencial del senyal, és a dir, les seves oscil·lacions periòdiques principals. Si fem un zoom al senyal considerant només una secció de 10 ms i analitzem el resultat, Figura 18 (b), veurem que en aquest interval el senyal es comporta d'una forma aproximadament periòdica i que hi ha 3 crestes principals. Això significa que el senyal conté una component freqüencial d'uns 300 Hz.

La idea principal de l'anàlisi Fourier és comparar el senyal amb sinusoides de diverses freqüències $\omega \in \mathbb{R}$, mesurades en Hz, és a dir que cadascuna d'aquestes sinusoides, també anomenades tons purs, estan pensades per ser els prototips de les oscil·lacions. Com a resultat d'això, obtenim, per cada paràmetre freqüencial $\omega \in \mathbb{R}$, és a dir, per cada pic de freqüència, un coeficient de magnitud $d_\omega \in \mathbb{R} \geq 0$, el qual va acompanyat d'un coeficient de fase $\varphi_\omega \in \mathbb{R}$. En el cas que el coeficient d_ω sigui gran, hi haurà molta similitud entre el senyal i la sinusoide de freqüència ω, la qual cosa significarà que el senyal contindrà una oscil·lació periòdica en aquesta freqüència, com es pot veure a la Figura 18 (c). En el cas que d_ω sigui petita, el senyal no contindrà cap component periòdic en aquesta freqüència, com es pot veure a la Figura 18 (d).

Si plotegem els coeficients d_ω respecte dels diversos paràmetres freqüencials $\omega \in \mathbb{R}$, obtenim un gràfic. A la Figura 18 (f) es pot veure aquest gràfic, on el paràmetre freqüencial amb el valor més gran és $\omega = 262\ Hz$, el qual correspon a la nota C4 que és la que s'havia tocat amb el piano. A més, com podem veure a la Figura 18 (e), la forma del senyal és molt semblant

a la forma que té la sinusoide de freqüència $\omega = 523\,Hz$, la qual cosa indica que el senyal analitzat també conté una freqüència corresponent al segon harmònic de la nota C4.

Amb aquest exemple es pot veure de forma general com funciona la transformada de Fourier. En resum, la transformada de Fourier separa el senyal en els corresponents components freqüencials i, per cada freqüència $\omega \in \mathbb{R}$, proporciona un coeficient d_ω acompanyat d'una fase φ_ω, el qual ens indica amb quina mesura el senyal analitzat coincideix amb l'oscil·lació del prototip sinusoidal de cada freqüència.

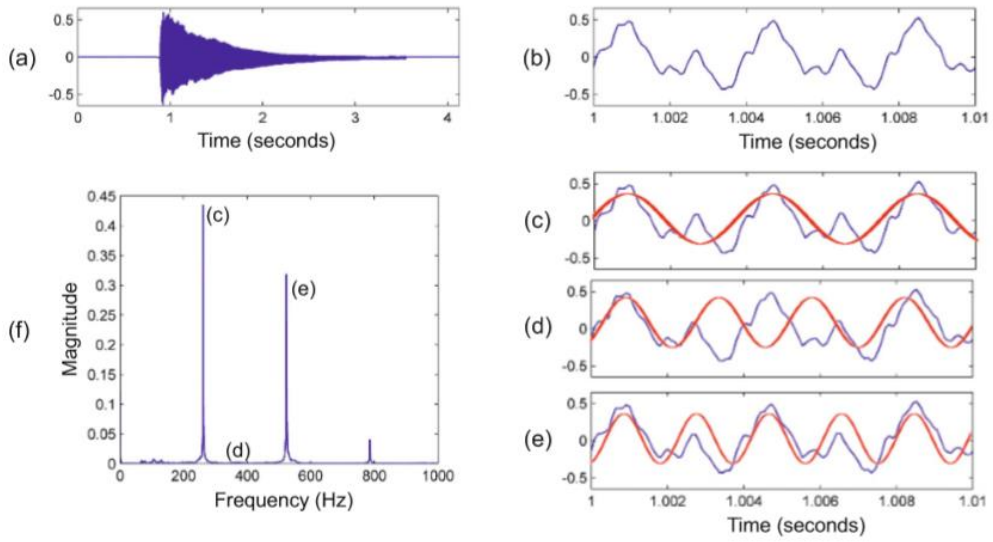

Figura 18: (a) Forma d'ona de la nota C4 (261.6 Hz) interpretada per un piano. (b) Zoom d'una secció de 10 ms que comença a l'instant de temps t = 1 s. (c-e) Comparació de la forma d'ona amb sinusoides de diferents freqüències. (f) Coeficients de magnitud d_ω dependents de la freqüència ω (Müller, 2015).

Una propietat important de la transformada de Fourier és que el senyal original pot ser reconstruït a partir dels coeficients d_ω, juntament amb les corresponents fases φ_ω. Per tant, se superposen les sinusoides de totes les possibles freqüències, el pes de les quals depèn del valor del corresponent coeficient d_ω i el desfasament de la corresponent fase φ_ω. Aquesta superposició ponderada s'anomena representació del senyal original de Fourier. El senyal original conté la mateixa quantitat d'informació que la transformada de Fourier, però representada de forma diferent. Mentre que el senyal original representa la informació al llarg

del temps, la transformada de Fourier la representa al llarg de la freqüència. Com *Hubbard* menciona al seu treball, el senyal ens indica en quin moment s'han tocat certes notes, però oculta la informació de les corresponents freqüències. En canvi, la transformada de Fourier de representacions musicals, representa quines notes, és a dir, quines freqüències s'han tocat, però oculta l'instant de temps en el qual han estat tocades.

2.4 Anàlisi espectral de senyals digitals

La teoria explicada en aquest apartat està basada en el llibre *DAFX: Digital Audio Effects* dels autors *Arfib, Keiler, Zölzer, Verfaille, & Bonada (2011).*

2.4.1 Espectre de freqüències

L'espectre d'un senyal mostra la seva distribució d'energia al llarg del rang de freqüències. La Figura 19 (a) mostra l'espectre d'una porció d'un senyal d'àudio digital. El rang de freqüències va de 0 a 20 kHz. El corresponent senyal digital s'obté a partir de mostrejar i quantificar aquest senyal analògic amb una freqüència de mostreig de $f_S = 40\ kHz$. L'espectre del senyal digital es pot veure a la Figura 19 (b). Les operacions de mostreig proporcionen una rèplica de l'espectre de banda base[5] del senyal analògic. En aquest cas, però, apareix un nou rang de freqüències simètric que va des de 20 kHz fins a 60 kHz, on la forma que té el senyal de 40 kHz fins a 60 kHz és igual a la que va de 0 kHz fins a 20 kHz. Això indica que l'eix de simetria el marca la freqüència de mostreig, en aquest cas de 40 kHz. La rèplica d'aquesta primera imatge corresponent a l'espectre de banda base, apareixerà als múltiples enters de la freqüència de mostreig $f_S = 40\ kHz$. La reconstrucció del senyal analògic a partir del digital es pot aconseguir aplicant un filtre passabaix, el qual rebutgi les freqüències superiors la meitat de la freqüència de mostreig ${f_S}/{2} = 20\ kHz$. D'aquesta manera el gràfic resultant serà igual al del senyal analògic.

2.4.2 Transformada discreta de Fourier

L'espectre d'un senyal digital es pot calcular mitjançant la transformada discreta de Fourier (DFT), la versió ràpida de la qual és la transformada ràpida de Fourier (FFT). La FFT pren N mostres consecutives del senyal $x(n)$ i realitza una operació matemàtica que proporciona N mostres $X(k)$ de l'espectre del senyal. A la Figura 20 es mostren els resultats d'aplicar una FFT de 16 punts sobre un senyal cosinus. El senyal resultant s'ha normalitzat respecte a N.

[5] Banda base, en telecomunicacions, és un senyal l'espectre freqüencial del qual està localitzat al voltant de l'origen o freqüència zero. Són senyals originals sense cap tipus de modulació.

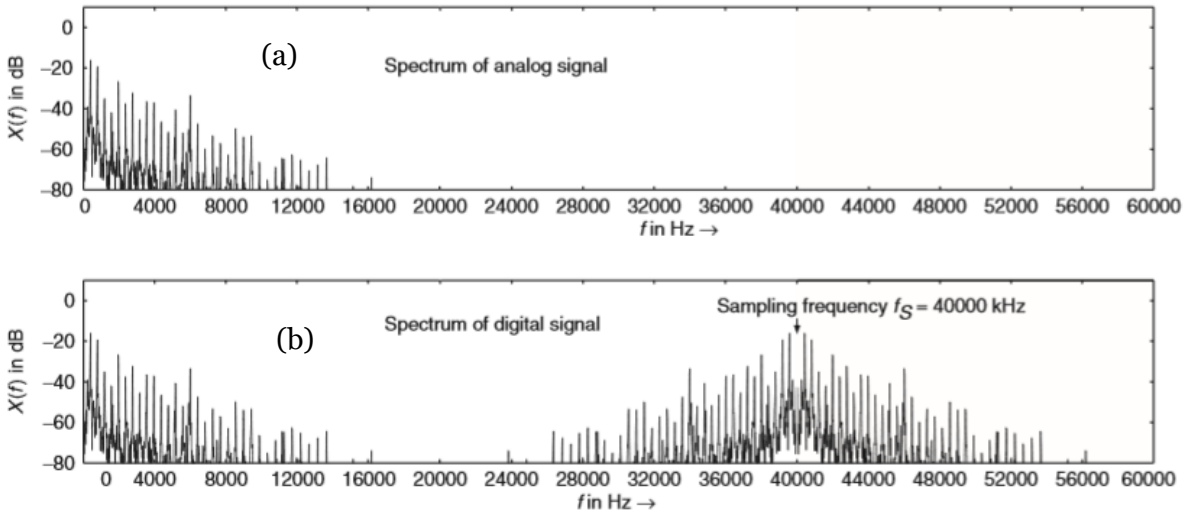

Figura 19: Espectre de freqüències d'un senyal analògic (a) i digital (b) (Arfib et al., 2011).

Les N mostres $X(k) = X_R(k) + jX_I(k)$ són valors complexos amb una part real $X_R(k)$ i una part imaginària $X_I(k)$, de les quals es pot extreure el valor absolut,

$$|X(k)| = \sqrt{X_R(k)^2 + X_I(k)^2} \; on \; k = 0, 1, \dots, N-1 \tag{8}$$

el qual serà la magnitud de l'espectre, i la fase,

$$\varphi(k) = \arctan\frac{X_I(k)}{X_R(k)} \; on \; k = 0, 1, \dots, N-1 \tag{9}$$

la qual és la fase de l'espectre. La Figura 20 també mostra que l'algoritme FFT proporciona N punts de freqüència equidistants, els quals estableixen que les N mostres de l'espectre del senyal comencen a 0 Hz i avancen en passos de f_S/N fins a $\frac{N-1}{N} \cdot f_S$. Cada punt freqüencial ve donat per $k \cdot \frac{f_S}{N}$, on el paràmetre k va avançant des de $0, 1, 2, \dots, N-1$. L'espectre de magnitud $|X(f)|$, normalment, es representa al llarg d'una escala d'amplitud logarítmica segons $20\log_{10}\left(\frac{X(f)}{0.5}\right)$, la qual fórmula retorna 0 dB per una amplitud màxima de ± 1 d'una sinusoide. Aquesta normalització és equivalent a $20\log_{10}\left(\frac{X(f)}{N/2}\right)$.

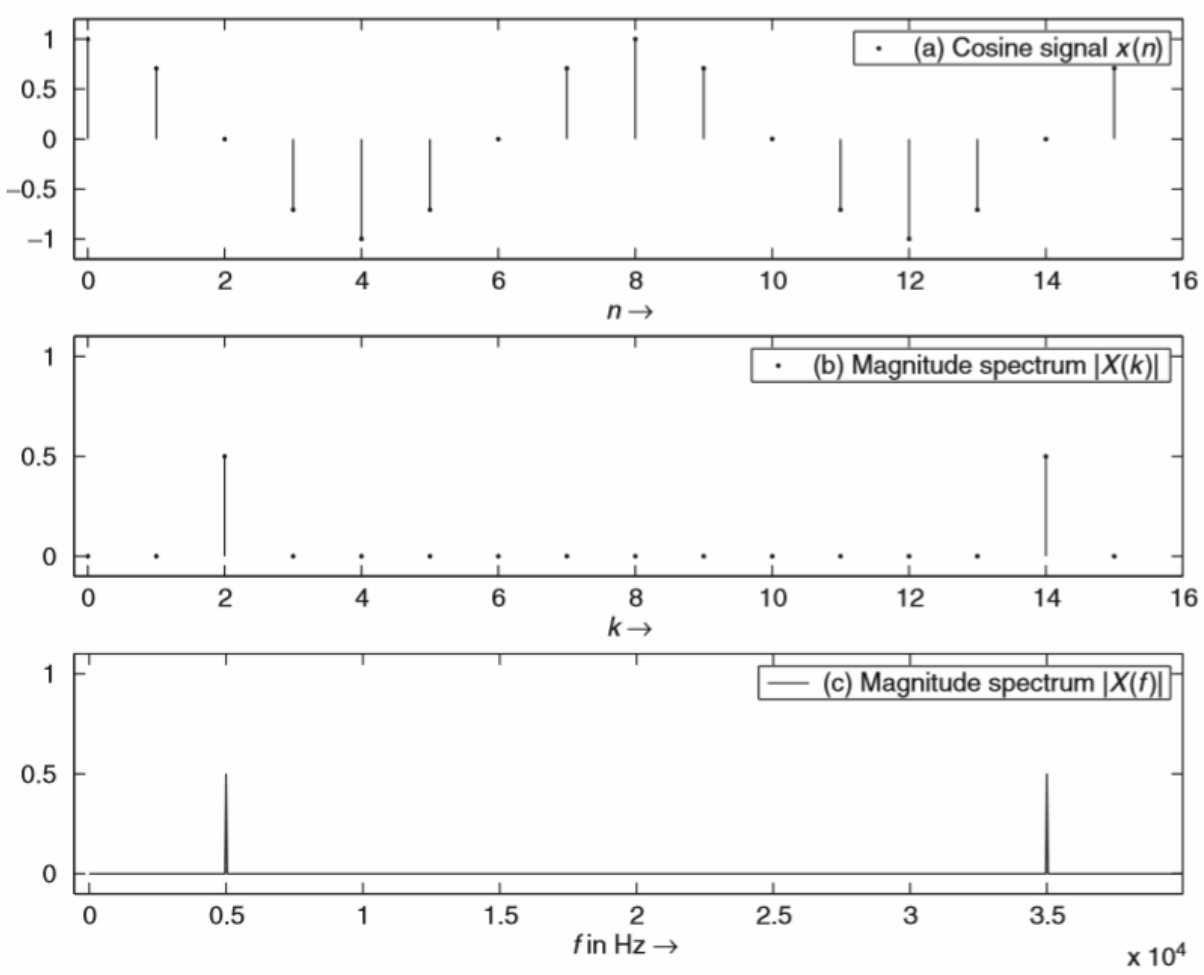

Figura 20: Anàlisi espectral mitjançant l'algoritme de la FFT. (a) Cosinus digitalitzat amb N = 16 mostres. (b) Espectre de magnitud |X(k)| amb N = 16 mostres freqüencials. (c) Espectre de magnitud |X(f)| des de 0 Hz fins a la freqüència de mostreig f_s = 40000 Hz (Arfib et al., 2011).

La Figura 21 mostra aquesta representació, a partir de l'exemple de la Figura 20. Com s'ha mencionat anteriorment, les imatges dels espectres de banda base estan situades a la freqüència de mostreig f_S i als corresponents múltiples d'aquesta. Per això, es pot veure que la freqüència original està situada a 5 kHz i el corresponent múltiple a $f_S - f_{cosinus} = 40000\ Hz - 5000\ Hz = 35000\ Hz$.

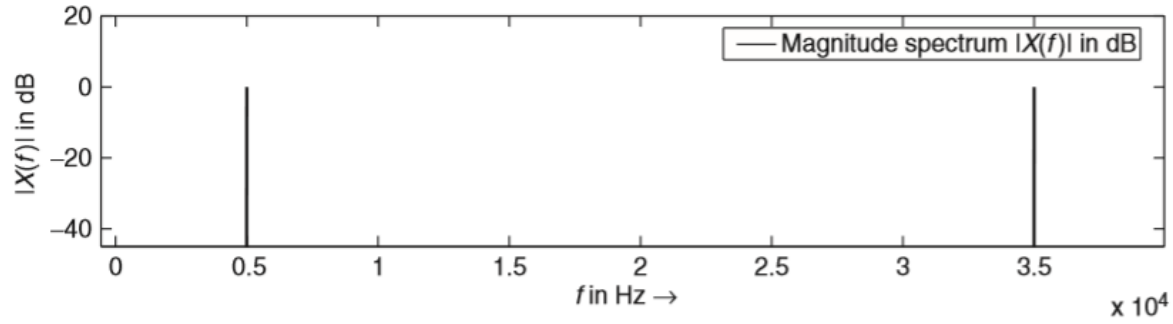

Figura 21: Espectre de magnitud |X(f)| en dB des de 0 Hz fins a la f_S = 40000 Hz (Arfib et al., 2011).

2.4.3 Transformada discreta inversa de Fourier (IDFT)

Mentre que la DFT s'utilitza per transformar del domini temporal discret al domini freqüencial discret per dur a terme anàlisis freqüencials, la IDFT permet dur a terme el procés invers, és a dir, transformar des del domini freqüencial discret al domini temporal discret. L'algoritme IDFT ve donat per

$$x(n) = IDFT[X(k)] = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi nk/N}\,, on\ n = 0, 1, \dots, N-1. \tag{10}$$

La versió ràpida de la IDFT s'anomena transformada ràpida de Fourier inversa (IFFT). A partir de N nombres amb la propietat $X(k) = X \cdot (N-k)$ en el domini freqüencial, duent a terme la IFFT s'obtenen N mostres discretes en el domini temporal, les quals tenen valors reals.

2.4.4 Resolució de freqüència: zero-padding i funcions finestra

Per millorar la resolució de freqüència en una anàlisi espectral, simplement s'han de prendre més mostres a l'hora de dur a terme la FFT. Els típics nombres utilitzats per definir la resolució de freqüència de la FFT són $N = 256, 512, 1024, 2048, 4096\ i\ 8192$. Si disposem d'un senyal de 64 mostres i volem millorar la seva resolució de freqüència de $f_S/64$ a $f_S/1024$, s'ha d'ampliar la seqüència de 64 mostres afegint zeros fins a tenir una llargada de 1024 i, aleshores, executar la FFT de 1024 punts. Aquesta tècnica s'anomena *zero-padding* i es pot veure el seu funcionament a la Figura 22. La imatge de dalt a l'esquerra mostra la seqüència original de 8 mostres i la imatge de dalt a la dreta la corresponent FFT. A baix a l'esquerra es pot veure com s'ha dut a terme el *zero-padding* afegint zeros fins a arribar a una llargada igual a 16 mostres,

i a baix a la dreta la corresponent FFT. Es pot veure clarament l'increment de la resolució de freqüència entre les dues FFT. Aquest increment es deu al fet que per cada bin de la FFT de 8 punts, la FFT de 16 punts en calcula dos. És a dir que els bins $k = 0, 2, 4, 6, 8, 10, 12, 14$ de la FFT de 16 punts corresponen als bins $k = 0, 1, 2, 3, 4, 5, 6, 7$ de la FFT de 8 punts. Aquests N bins de freqüència tenen un rang de 0 Hz fins a $\frac{N-1}{N} \cdot f_S$.

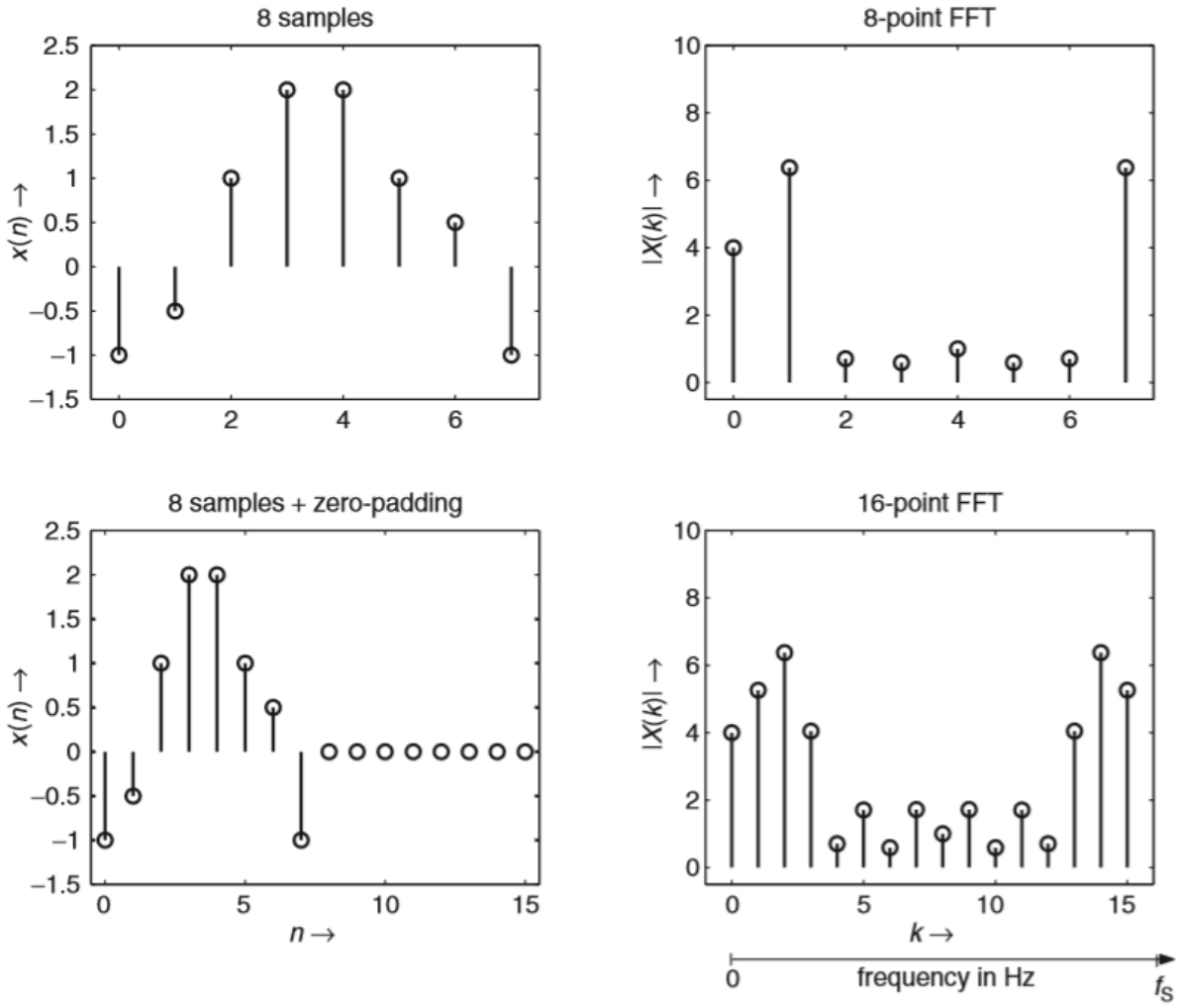

Figura 22: *Zero-padding* per millorar la resolució de freqüència (Arfib et al., 2011).

La fuga espectral sorgeix quan hi ha discontinuïtats als extrems d'una funció. Quan apareix l'efecte de la fuga espectral, sembla com si l'energia d'una freqüència es fugués cap a altres freqüències. És a dir que hi ha línies espectrals que tenen un senyal ample, però que, en principi, hauria de ser suau. Aquest efecte es mostra a la Figura 23 (a) i (b), on es pot veure que l'amplitud de les freqüències no fonamentals és molt elevada quan, en principi, hauria de ser

de nul·la. Es pot reduir l'efecte de fuga seleccionant una funció finestra com, per exemple, la finestra *Blackman* o la finestra *Hamming*

$$\omega_B(n) = 0.42 - 0.5\cos\left(\frac{2\pi n}{N}\right) + 0.08\cos(4\pi n/N) \tag{11}$$

$$\omega_H(n) = 0.54 - 0.46\cos(2\pi n/N) \tag{12}$$

$$on\ n = 0, 1, \dots, N-1$$

i ponderant les N mostres d'àudio amb la funció finestra. Aquesta ponderació es duu a terme segons $x_\omega = \omega(n) \cdot x(n) / \sum_k \omega(k)$ amb $0 \leq n \leq N-1$ i executant la FFT del senyal ponderat. La funció cosinus ponderada mitjançant una finestra *Blackman* i el corresponent espectre es poden veure a la Figura 23 (c) i (d). La Figura 23 (e) i (f) mostra un segment de senyal d'àudio ponderat mitjançant una finestra *Blackman* i el corresponent espectre dut a terme mitjançant la FFT.

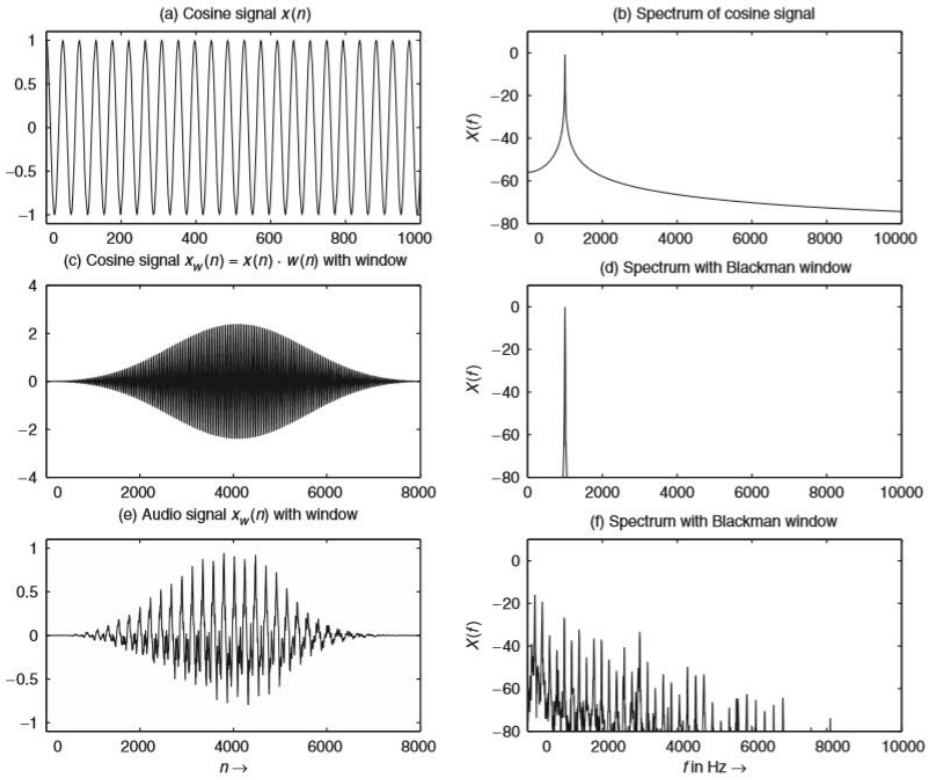

Figura 23: Anàlisi espectral de senyals digitals. S'agafen N mostres d'àudio i s'executa la DFT de N punts, la qual proporciona un espectre de freqüències amb un rang que va de 0 Hz fins a $k\frac{f_S}{N}$ on $k = 0, 1, \dots, N-1$. De la (a) fins a la (d), $x(n) = \cos(2\pi \cdot \frac{1\ kHz}{44.1\ kHz} \cdot n)$ (Arfib et al., 2011).

La Figura 24 mostra més exemples de com s'aplica una funció finestra per reduir l'efecte de la fuga espectral.

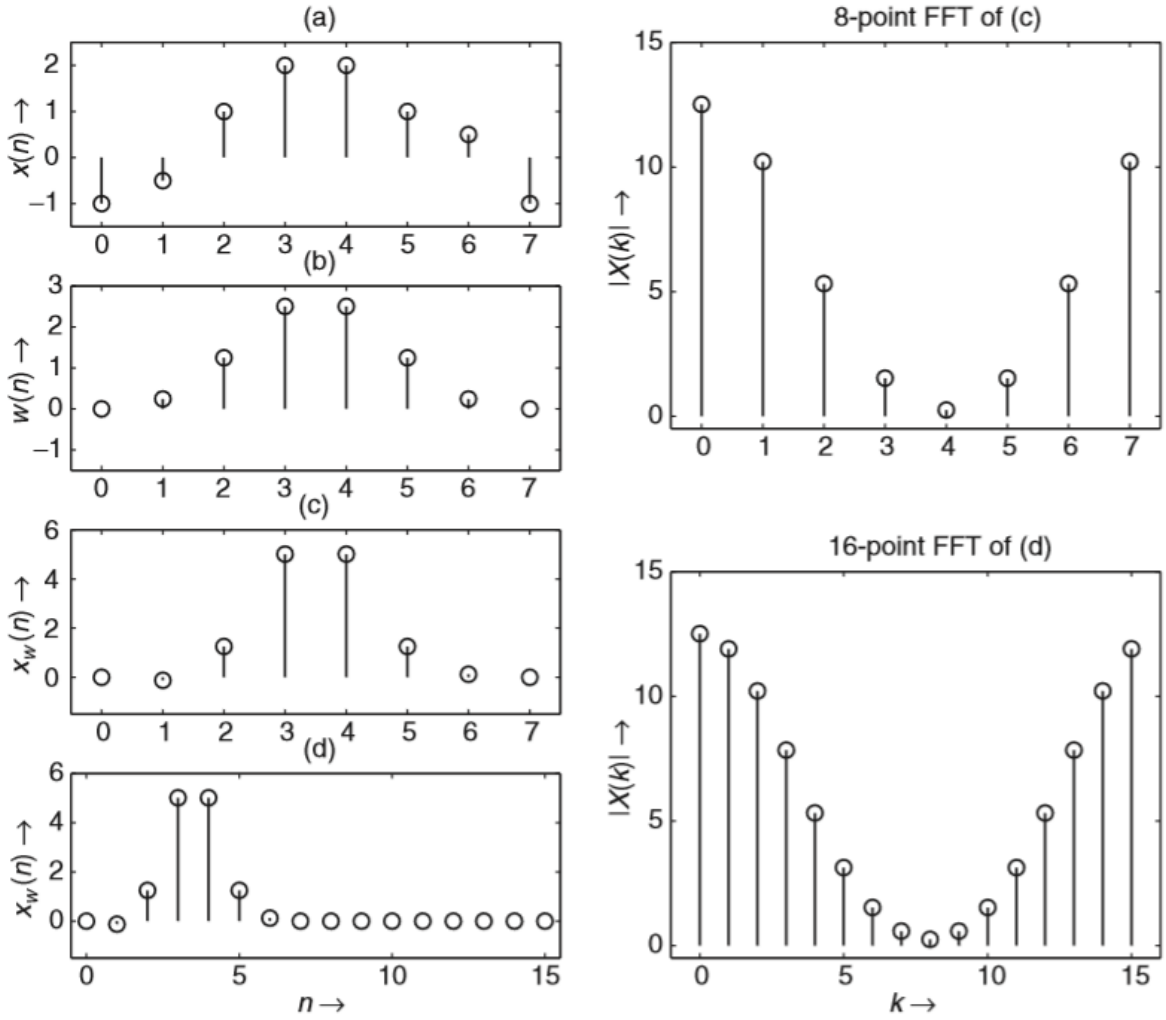

Figura 24: Reducció de l'efecte de fuga mitjançant funcions finestra. (a) El senyal original $x(n)$. (b) Finestra *Blackman* $\omega_B(n)$ de llargada N = 8. (c) Producte $x(n) \cdot \omega_B(n)$ per $0 \leq n \leq N-1$ i el corresponent espectre. (d) *Zero-padding* aplicat a $x(n) \cdot \omega_B(n)$ fins a una llargada de N = 16 i el corresponent espectre (Arfib et al., 2011).

2.4.5 Espectrograma

L'espectrograma representa les variacions de freqüència (eix vertical) i d'amplitud (intensitat del color) del senyal sonor al llarg del temps (eix horitzontal). Proporciona una estimació de l'evolució espaciotemporal del contingut freqüencial del senyal, la qual permet realitzar anàlisis de sonoritat, duració, timbre, amplitud, pauses, accents i ritme. Per obtenir un espectrograma, el senyal es divideix en segments de llargada N, els quals es multipliquen

per una funció finestra i, aleshores, s'hi aplica la FFT. Per incrementar la localització temporal de l'espectre s'utilitza una superposició de les finestres.

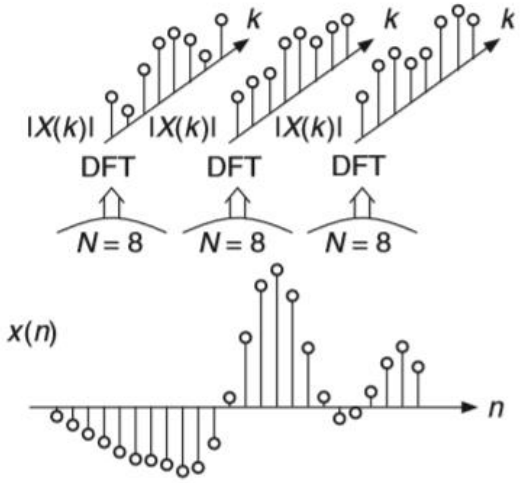

Figura 25: Anàlisi espectral mitjançant FFT (Arfib et al., 2011).

Una representació molt visual d'un espectrograma la proporciona la Figura 26. El temps incrementa linearment al llarg de l'eix horitzontal i la freqüència al llarg de l'eix vertical. Cada línia vertical representa el valor absolut $|X(f)|$ al llarg de la freqüència mitjançant una escala ponderada de grisos, com més fosc és el color, més abundant és la freqüència en qüestió. Només les freqüències més grans que la meitat de la freqüència de mostreig estan representades.

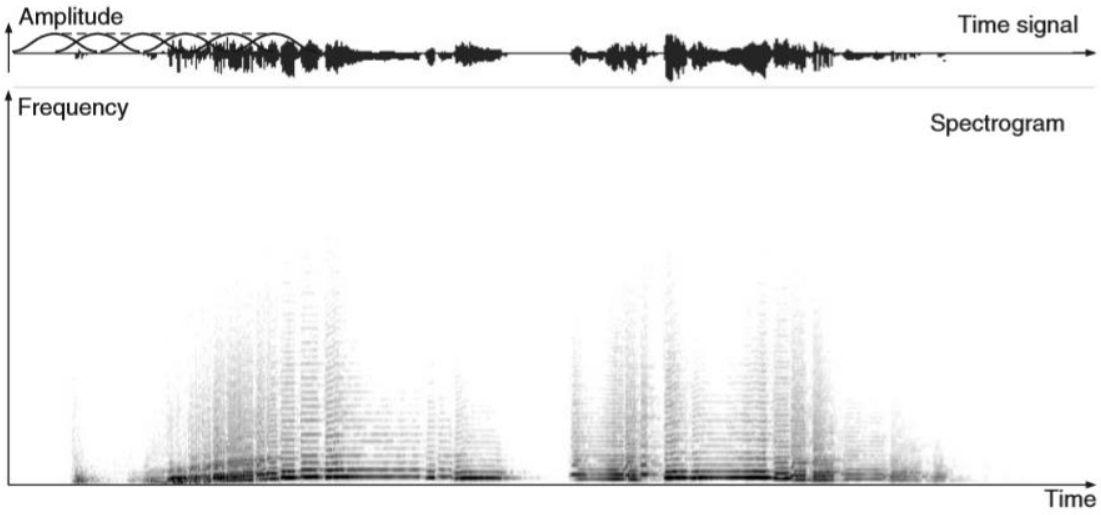

Figura 26: Espectrograma dut a terme mitjançant la FFT de segments ponderats (Arfib et al., 2011).

3 DESENVOLUPAMENT DE L'EINA D'ANÀLISI ACÚSTICA

En aquest apartat veurem com s'ha decidit que havien de funcionar les diferents parts del projecte: el codi i la interfície gràfica. El programa utilitzat pel disseny d'aquesta part és el MATLAB, el qual conté una llibreria anomenada *Signal Processing Toolbox* molt útil per dur a terme anàlisis acústiques. També s'explicarà la problemàtica que ha anat sorgint al llarg del procés.

3.1 Estructura del codi

Com s'ha mencionat anteriorment, la finalitat del programa és proporcionar espectres i espectrogrames a partir d'un arxiu d'àudio on hi ha gravades una sèrie de repeticions fetes amb una guitarra clàssica. Per dur a terme això, el codi s'ha dividit en dues parts. Per una banda hi ha el preprocés, en el qual crea un arxiu d'àudio per cada repetició, i per l'altra el postprocés, on s'identifica cada nota i es creen els corresponents espectres i espectrogrames.

3.1.1 Preprocés

El preprocés està dividit en tres parts. A la primera part es creen carpetes per guardar els fitxers necessaris per poder executar correctament el programa. A la segona part es crea un arxiu d'àudio diferent per cadascuna de les repeticions. I a la tercera part s'aplica un filtre als arxius d'àudio creats a la segona part, per tal de conservar només els que continguin notes ben tocades.

La creació de carpetes consisteix en, un cop s'ha triat el nom de la guitarra mitjançant la interfície gràfica, buscar si hi ha carpetes amb el mateix nom. Si es troben, s'esborren i es creen les noves carpetes i, si no es troben, es creen les noves carpetes. L'objectiu d'aquesta part és crear una carpeta per guardar els arxius d'àudio de cada repetició, una altra per guardar els espectres i espectrogrames i l'última per guardar una matriu que contingui el nom de totes les notes que ha tocat la guitarra clàssica.

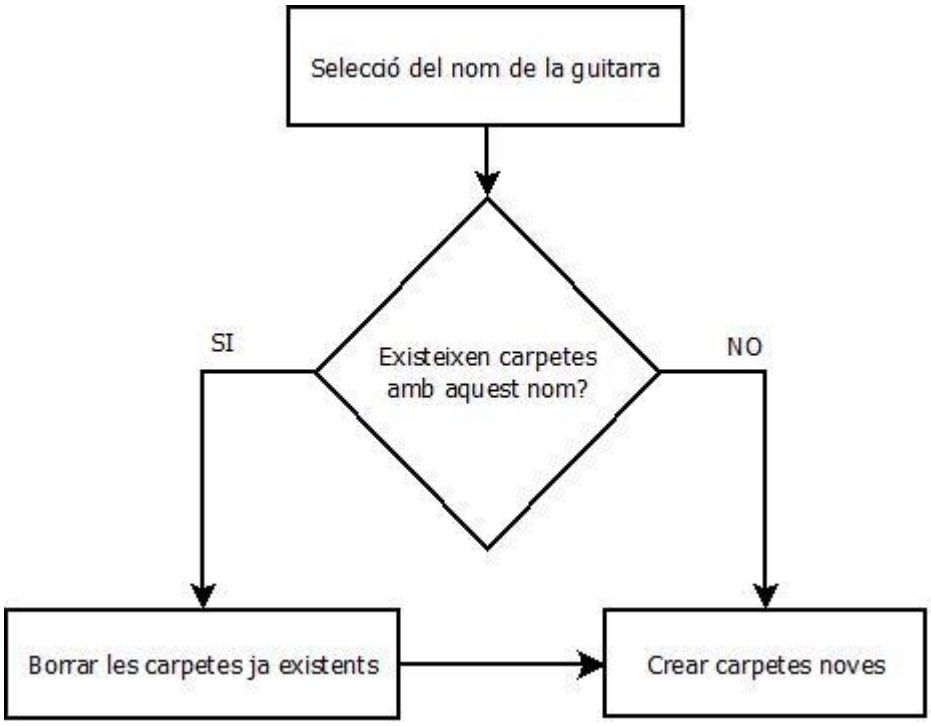

Figura 27: Diagrama de flux de la primera part del preprocés. Creació de carpetes.

La creació d'un arxiu d'àudio diferent per cada repetició consisteix a identificar l'inici i el final de cada repetició i crear un nou arxiu d'àudio de cadascuna. En un principi es va utilitzar una funció de MATLAB anomenada *imdilate*, la qual dilatava el senyal d'àudio de tal manera que es generava un envolvent d'aquest. Aleshores, se seleccionaven els punts més alts d'aquest envolvent mitjançant la funció *findpeaks* i restriccions d'amplitud i de temps, és a dir, de mínima alçada i de mínima distància lateral. Aquest mètode donava dues problemàtiques diferents: la primera era que els nous fitxers creats començaven en el punt on la nota tenia amplitud més alta i no al principi d'aquesta, la qual cosa no interessava; i la segona era que els punts de mínima amplitud i mínima distància lateral eren diferents per cada tipus de guitarra, ja que a les gravacions utilitzades qui tocava cada repetició ho feia de forma diferent, és a dir, amb una intensitat més o menys alta i una durada de cada repetició més o menys curta.

Per solucionar el problema anterior, es va decidir aplicar el filtre *Savitzky-Golay* amb un ordre polinòmic igual a 2 i una llargada de finestra igual a 2001. Aquest filtre s'implementa mitjançant la funció de MATLAB *sgolayfilt*, els paràmetres de la qual es van seleccionar després de realitzar diverses proves on es mirava si l'envolvent resultant era el desitjat. En

aquest cas, es genera un envolvent que, a diferència de l'anterior, és negatiu just abans de l'inici de cada repetició, és a dir, en els moments on hi ha canvis molt sobtats d'amplitud. Això permet localitzar l'inici i el final de cada repetició, ja que el final d'una nota és l'inici de l'altre, per tant, tant l'inici com el final són punts on l'envolvent és negatiu. D'aquesta manera s'evita posar restriccions d'amplitud i de temps mínims. Un cop seleccionats els punts inicial i final, es crea un nou fitxer per cadascuna de les repeticions detectades en el procés anterior i es guarda a la carpeta corresponent.

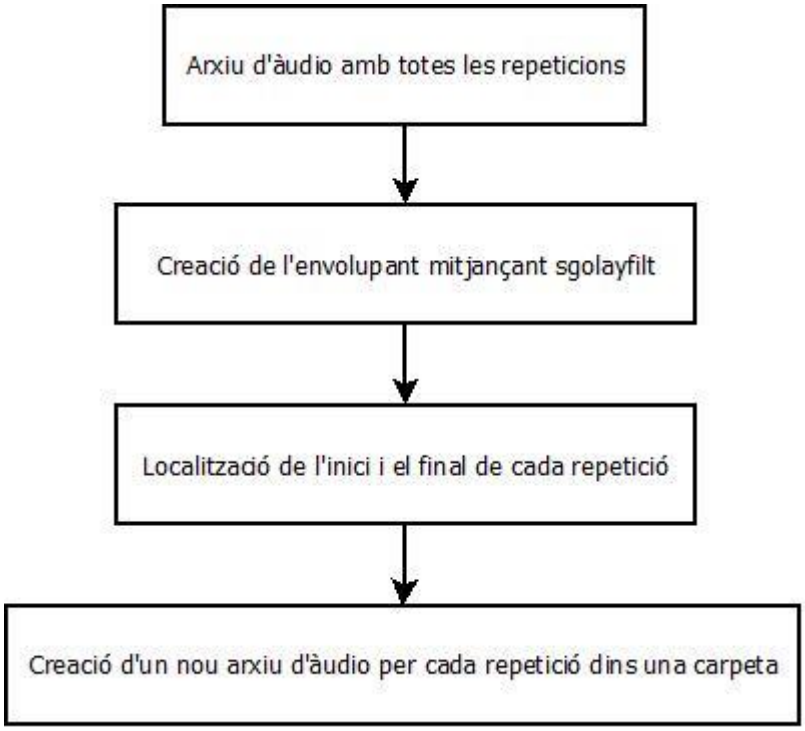

Figura 28: Diagrama de flux de la segona part del preprocés. Creació d'un arxiu d'àudio per cada repetició.

L'últim pas del preprocés, el qual consisteix a aplicar filtres per descartar repeticions que no ens interessen, és el que fa que el programa actual tardi molta estona a executar-se. Per decidir si prenem una nota com a bona o si no, es mira si segueix el model d'envolvent ADSR, el qual s'ha explicat a l'apartat 2.1.5 i si té una durada mínima. Per comprovar si la nota segueix el model d'envolvent ADSR, en primer lloc es crea l'envolvent de la nota analitzada mitjançant la funció *imdilate* i, a partir de l'envolvent i la funció *ischange*, es busca el nombre canvis

bruscos i la posició de l'últim canvi que té l'envolvent en qüestió. Un cop guardats aquests dos valors, s'aplica un filtre que es queda amb les notes que compleixen:

- El nombre de canvis detectats ha d'estar entre 4 i 10. El model d'envolvent ADSR té un nombre de canvis bruscs igual a 5. Com que s'ha de tenir en compte que no totes les notes estan tocades de forma perfecta, és a dir que poden tenir més o menys canvis bruscs, s'utilitza un rang més ampli que va de 4 fins a 10. S'han escollit aquests valors després d'observar la forma que tenia cada nota. Es podria escollir un rang més restrictiu i el programa funcionaria correctament també.
- L'últim canvi trobat no ha d'estar més lluny del 65% de la longitud de la nota. S'ha pres aquest valor a partir d'observar l'envolvent de les notes, el qual, com s'ha mencionat anteriorment, segueix el model ADSR. En les notes que segueixen aquest model, l'amplitud va disminuint fins a arribar a zero, per això aquest filtre només selecciona les notes que no tenen cap canvi brusc en els últims instants.

Si la nota compleix els requisits del filtre anterior, passa per un últim filtre que comprova si la nota té la durada mínima establerta. Per comprovar això, es busca la durada de l'arxiu d'àudio que s'està analitzant i, si la nota té més durada, es guarda, si no s'esborra.

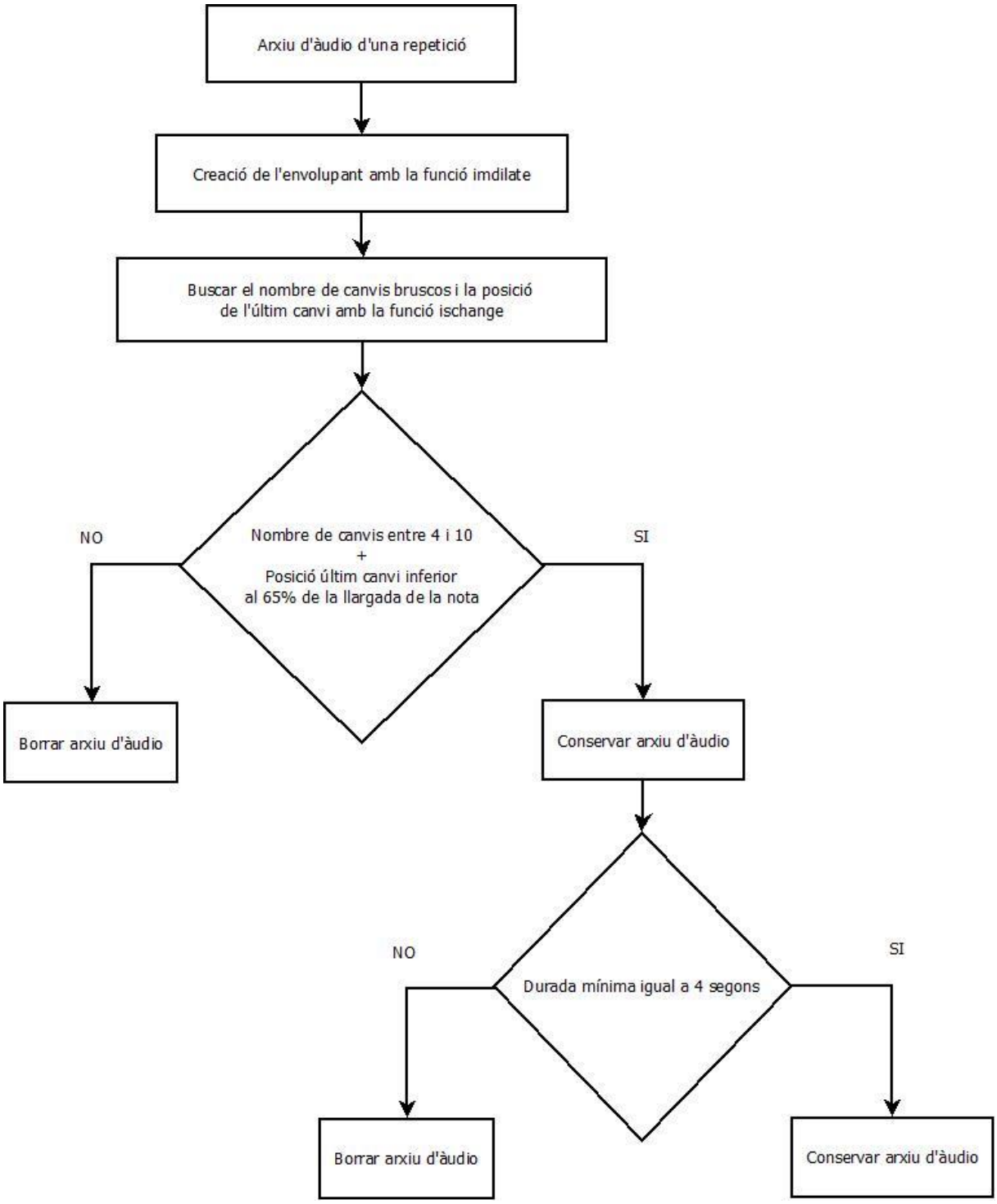

Figura 29: Diagrama de flux de la tercera part del preprocés. Filtre de selecció de les notes útils.

3.1.2 Postprocés

El postprocés també consta de tres parts. A la primera s'identifica la nota de cada repetició. A la segona s'iguala la durada de totes les repeticions de cada nota i es guarden els noms de les

notes en una matriu dins la carpeta de repeticions. A la tercera es creen els espectres i espectrogrames individuals i mitjans de cada nota.

La identificació de les notes es fa a partir dels fitxers guardats a la carpeta de les notes, els quals són processats per separat. En primer lloc, s'aplica la funció *pitch* per trobar totes les freqüències que apareixen a l'arxiu. Aquesta funció és capaç d'identificar freqüències perquè aplica la FFT a l'arxiu, la qual, com s'ha mencionat a l'apartat 2.3, és capaç d'identificar les diferents freqüències i les corresponents magnituds que apareixen a un arxiu d'àudio. Aleshores, les freqüències obtingudes s'analitzen mitjançant un histograma, del qual s'agafa la freqüència que més cops s'ha captat amb la funció anterior. Un cop s'ha trobat la nota, es canvia el nom del fitxer anterior i s'hi afegeix el nom de la nota i el número de repetició. Per cada nova nota, es reinicia el número de repetició. Una problemàtica important d'aquesta part és que, en algun cas puntual, la freqüència que més vegades ha captat la funció *pitch* no és la fonamental. Això implica que el nom de la nota identificada no sigui correcte. A la Figura 43 es pot veure l'evolució de la nota E5 de la guitarra M16, mitjançant un espectrograma, on no només apareixen harmònics, sinó que també hi ha parcials que no corresponen a la nota en qüestió.

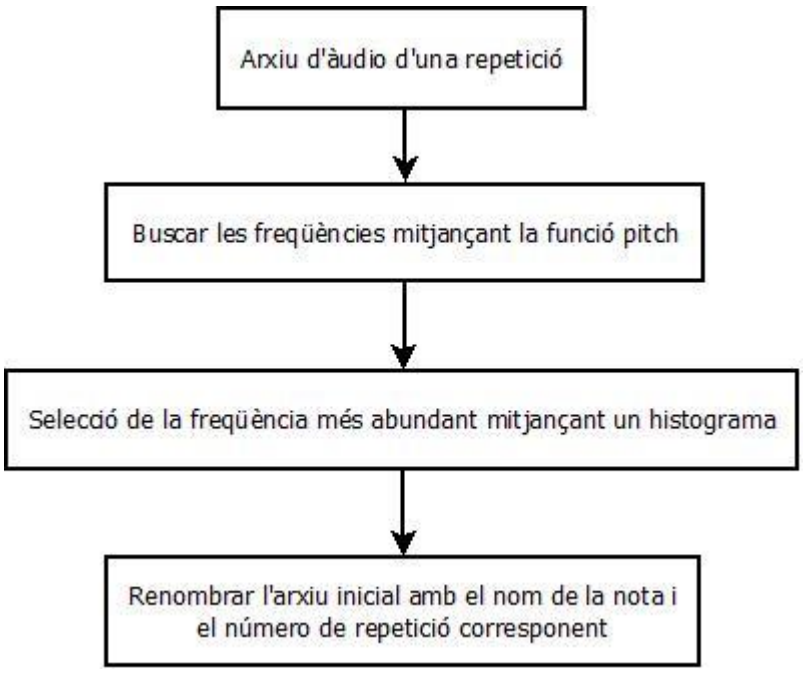

Figura 30: Diagrama de flux de la primera part del postprocés. Identificació de les notes.

Aquest error fa que l'espectrograma no sigui correcte, ja que realitza la mitjana de notes diferents. En el codi actual no hi ha cap filtre capaç de solucionar aquesta problemàtica.

En un principi, aquesta part es feia d'una manera semblant a la que s'utilitzava per seleccionar les diferents repeticions al preprocés. En aquest cas, però, es realitzava la transformada de Fourier de cadascuna de les repeticions i s'obtenia un espectre de freqüències, del qual, mitjançant la funció *findpeaks*, se seleccionaven els pics amb restriccions d'amplitud i temps. Igual que al preprocés, però, les restriccions vàlides per un tipus de guitarra, no ho eren per un altre tipus, la qual cosa impedia que el programa funcionés correctament. Això passava perquè en alguns casos el segon i tercer harmònic, l'explicació teòrica dels quals podem trobar a l'apartat 2.1.3, tenien una amplitud més gran que el primer, el qual tenia una amplitud suficientment petita perquè, amb les restriccions establertes, no se seleccionés com a pic. Quan sorgia aquest error, la nota identificada no era la correcta. El filtre que s'aplicava en aquest cas era que el segon pic havia de ser un múltiple enter del primer, per això si s'identificaven el segon i tercer harmònic, el codi donava la nota com a bona.

Per igualar la durada de cada repetició, les notes passen un altre filtre que elimina aquelles notes que només tenen una repetició, ja que, com que les notes no han estat tocades de forma perfecta, en alguns casos s'ha identificat una freqüència que no és la fonamental en el procés d'identificar notes i, per tant, el nom de la nota guardada és incorrecta. Com que són casos aïllats, perquè en el preprocés ja passen per un filtre bastant restrictiu, no hi ha més d'una repetició per cada nota mal analitzada. Un cop passat aquest filtre, per cada nota es crea una matriu on s'emmagatzemen les durades de cada repetició. A partir d'aquesta matriu se selecciona la durada més curta, es tallen la resta de repeticions a aquesta durada i es guarden els nous arxius. Les notes resultants, és a dir, les notes que no han estat eliminades en cap dels filtres, es guarden en una matriu dins la carpeta de notes.

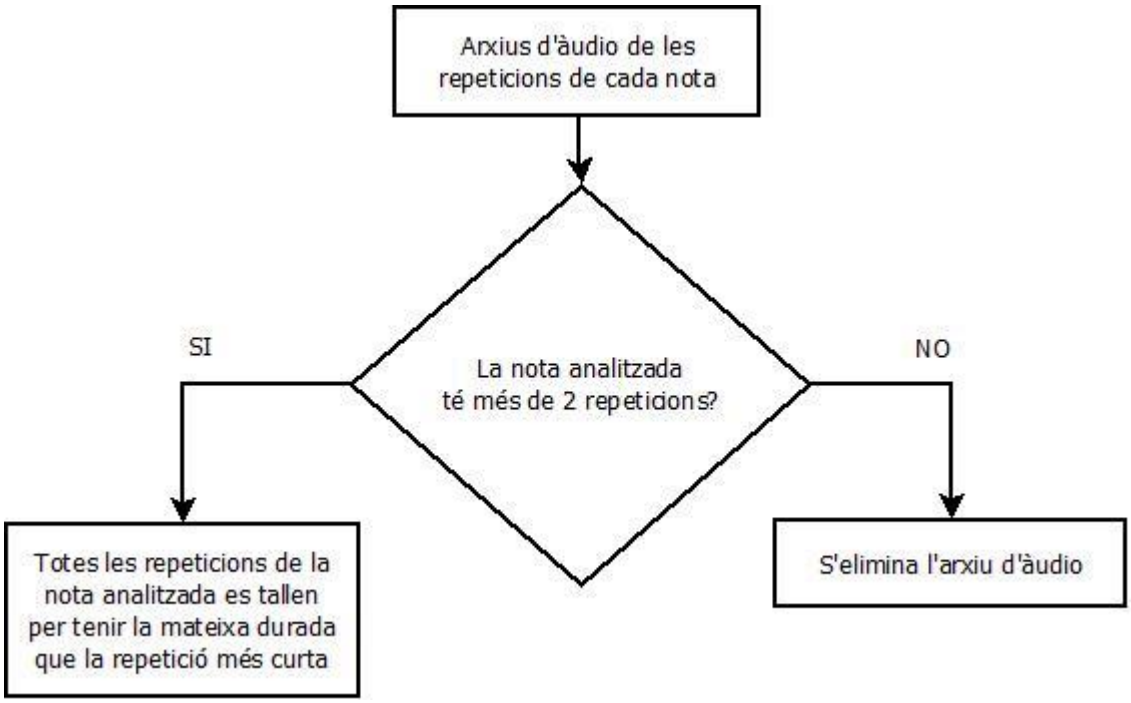

Figura 31: Diagrama de flux de la segona part del postprocés. Igualació de les durades de les repeticions de cada nota.

Finalment, l'última part del postprocés consisteix a crear espectres i espectrogrames dels fitxers situats dins la carpeta de repeticions. A l'hora d'establir els límits dels gràfics, el límit de temps està fixat perquè, en aquest punt del procés, totes les repeticions tenen la mateixa durada. El límit de freqüència, en canvi, s'estableix mitjançant el nombre d'harmònics desitjat que a la versió actual del codi és de 20, però en noves versions podria ser variable, és a dir, depenent de la freqüència de la nota analitzada utilitzar un número més gran o més petit d'harmònics. Per crear tant els espectres com els espectrogrames s'utilitza la funció *pspectrum*.

En el cas de l'espectre de freqüències, els paràmetres de la funció utilitzats són: els límits, com hem mencionat anteriorment, i la resolució de freqüència, explicada a l'apartat 2.4.4, la qual, com que el nombre de mostres utilitzat és molt elevat, el seu valor hauria de ser molt inferior a 1. Tot i això, s'ha decidit utilitzar un valor igual a 1 perquè la funció en qüestió dóna resultats correctes per aquest valor. Les dades de magnitud obtingudes es converteixen a decibels i es plotegen respecte a la freqüència.

En el cas de l'espectrograma, els paràmetres de la funció utilitzats són: els límits, com hem mencionat anteriorment, i la resolució temporal, la qual s'ha escollit provant diferents

números fins a trobar el que donava millors resultats. Mitjançant la funció *surface* i els paràmetres obtinguts amb *pspectrum*, es crea l'espectrograma desitjat. En els dos casos, primer es creen els gràfics de cada repetició i tot seguit de la corresponent mitjana. Finalment, es guarden els gràfics en els formats *.tif* i *.eps* a la carpeta de figures.

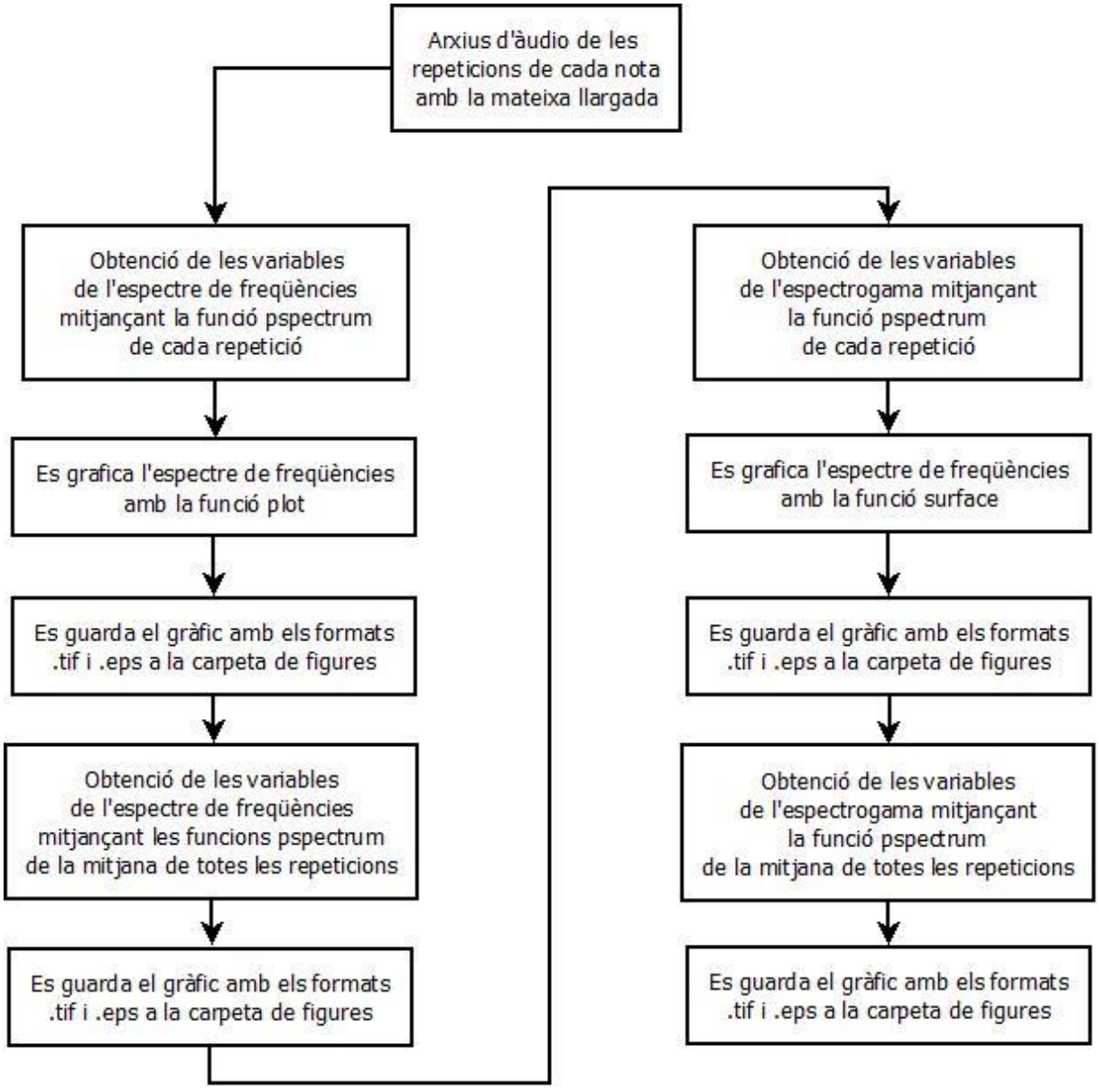

Figura 32: Diagrama de flux de la tercera part del postprocés. Creació d'espectres i espectrogrames.

3.2 Interfície gràfica

La interfície gràfica s'ha dut a terme mitjançant l'extensió de MATLAB GUIDE, la qual proporciona eines per dissenyar interfícies d'usuari per a aplicacions personalitzades. La interfície gràfica creada en aquest projecte permet executar el codi de forma senzilla. D'aquesta manera s'elimina la necessitat que l'usuari hagi d'aprendre un llenguatge de programació i hagi d'escriure comandes amb la finalitat d'executar el codi en qüestió.

El disseny de la interfície gràfica consta de 4 pestanyes, les quals separen una part de l'altra. Per tant, la interfície gràfica està composta per 4 parts: *data*, *run*, *plot* i *compare*.

3.2.1 Data

En aquesta part, tal com es pot veure a la Figura 33, l'usuari pot escollir l'arxiu d'àudio que vol analitzar polsant el botó *browse file*, el qual permet seleccionar l'arxiu d'àudio *.wav* que l'usuari desitgi. A la casella de sota, el títol de la qual és *guitar name*, s'hi ha d'introduir el nom de la guitarra. En aquest cas, el nom de la guitarra ha de ser totalment diferent que qualsevol dels altres noms que es vulguin donar a altres guitarres, ja que el codi està fet de tal manera que si dos noms tenen lletres coincidents, no és capaç d'identificar la diferència entre un nom i l'altre.

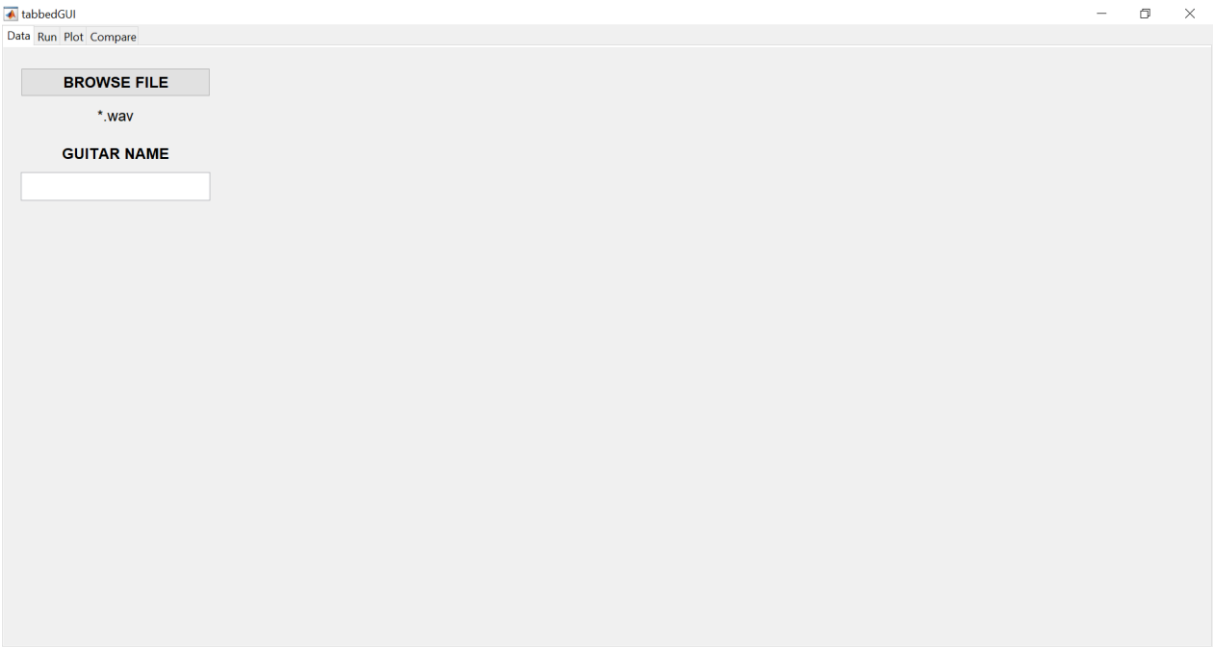

Figura 33: Part 1 (*data*) de la interfície gràfica.

3.2.2 Run

En aquesta part, l'usuari només té una opció que és polsar el botó *run*, el qual fa que s'executi el codi amb els paràmetres definits a la pestanya anterior. Un cop s'ha acabat l'execució, es torna a obrir la interfície gràfica per la pestanya 1.

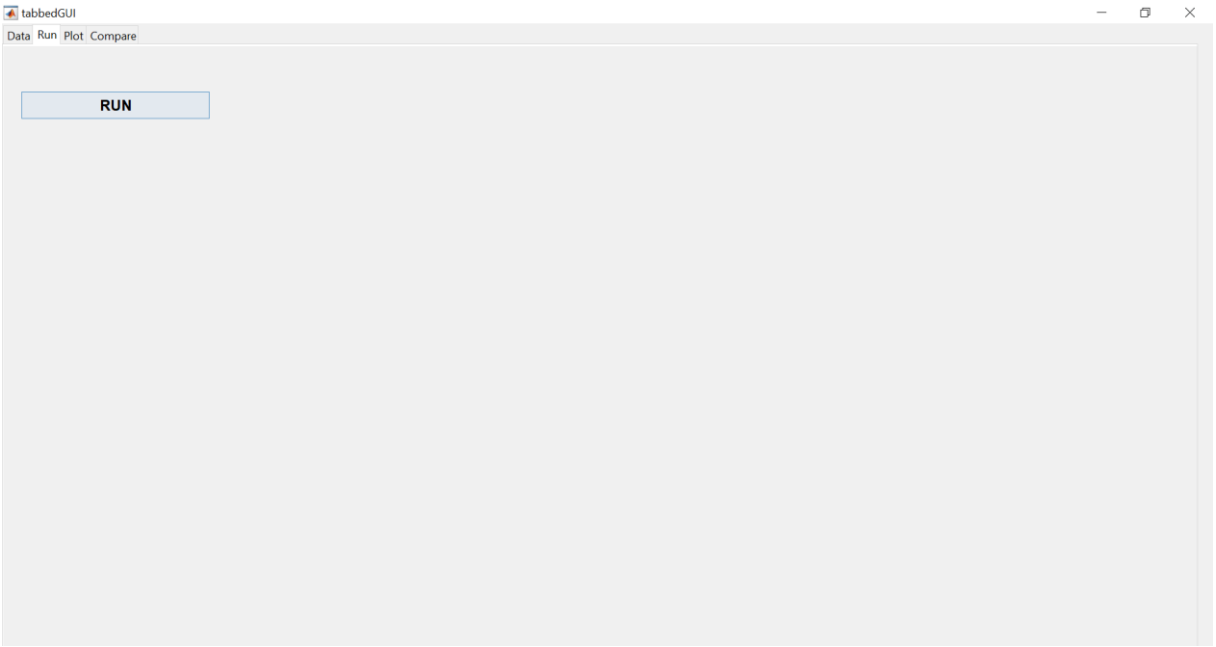

Figura 34: Part 2 (*run*) de la interfície gràfica.

3.2.3 Plot

En aquesta part, tal com diu el nom, el programa té la capacitat de graficar la nota de la guitarra que l'usuari vulgui analitzar. Un cop s'ha executat el codi que calcula els espectres i els espectrogrames, el programa ja té accés a tots els gràfics desitjats. L'usuari simplement ha de seleccionar, en primer lloc, la guitarra i després la nota que es vol analitzar. Un cop s'ha seleccionat la nota, tal com es veu a la X, apareixen tant l'espectre de freqüències com l'espectrograma d'aquesta.

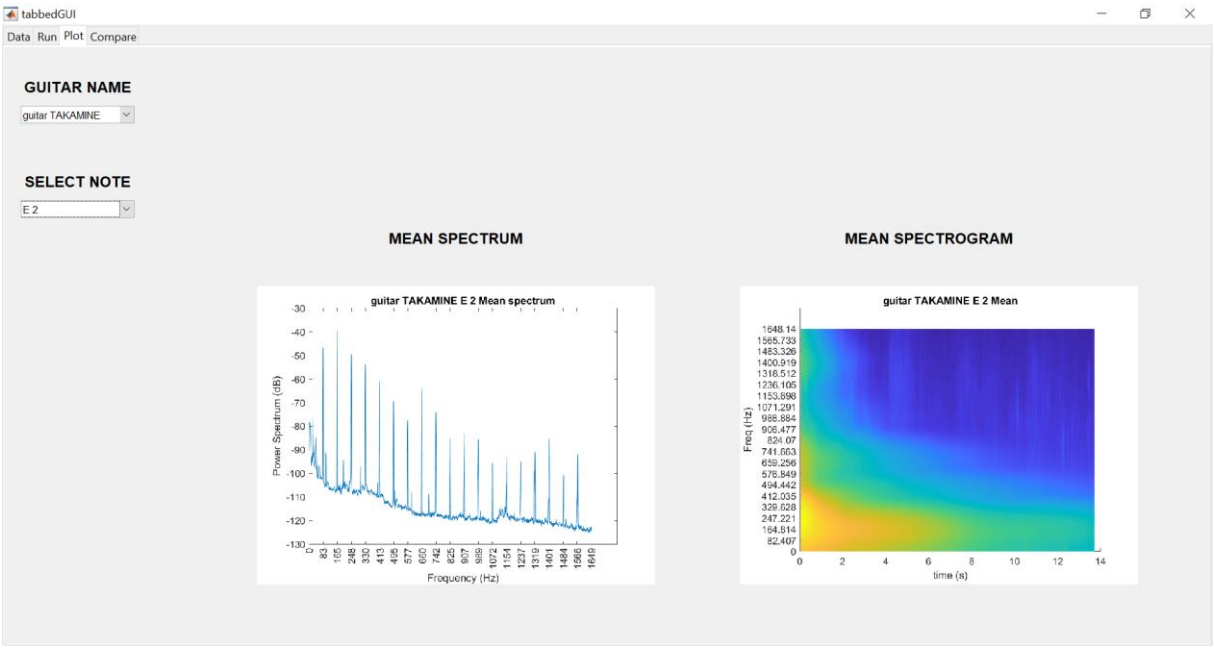

Figura 35: Part 3 (*plot*) de la interfície gràfica.

3.2.4 Compare

Finalment, aquesta última part, permet a l'usuari dur a terme una comparació d'una mateixa nota entre dues guitarres diferents. En aquest cas, només es pot comparar un tipus de gràfic, és a dir, o l'espectre de freqüències o l'espectrograma. Perquè l'usuari pugui dur a terme aquesta comparació, en primer lloc ha de seleccionar els dos tipus de guitarra que vol comparar, en segon lloc, el tipus de gràfic i, per últim, la nota que es vulgui analitzar. Les notes que apareixen són les coincidents entre una guitarra i l'altra. Un cop s'ha seleccionat la nota, tal com es veu a la Figura 36, apareixen els gràfics seleccionats de la nota en qüestió d'una guitarra i de l'altra.

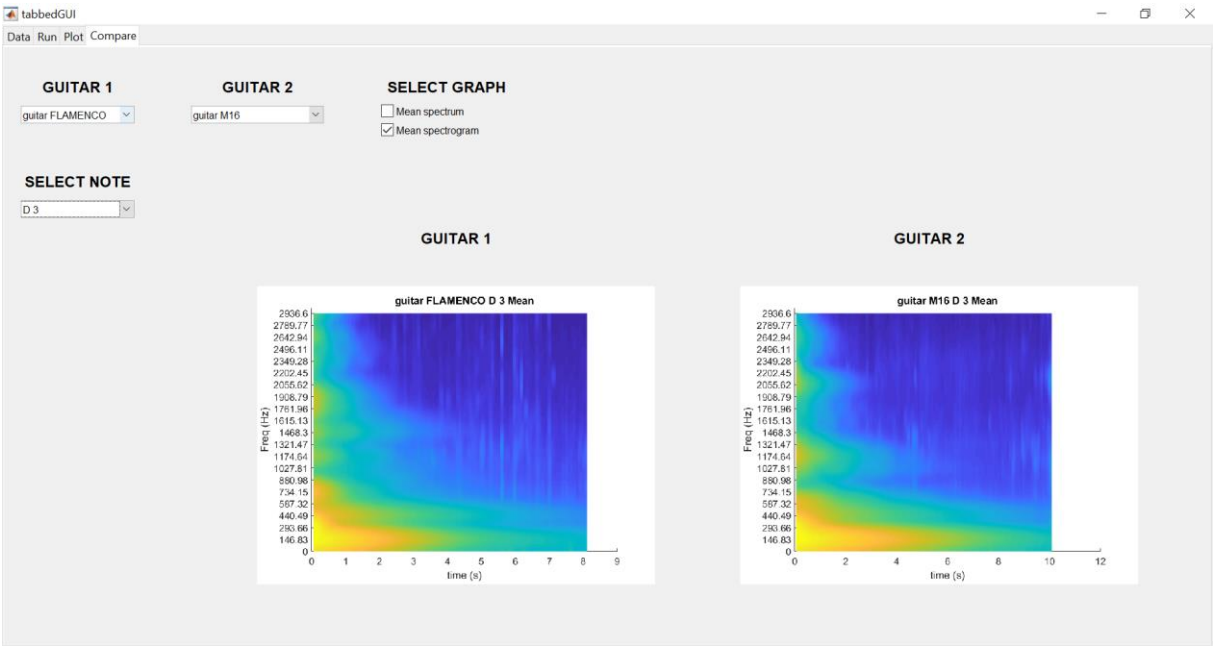

Figura 36: Part 4 (*compare*) de la interfície gràfica.

4 EXEMPLES D'APLICACIÓ

L'aplicació que s'ha obtingut com a resultat d'aquest projecte, en principi hauria de ser capaç de poder proporcionar qualsevol espectre i espectrograma de les notes interpretades per cadascuna de les guitarres clàssiques que l'usuari desitgi analitzar. Si l'objectiu fos analitzar algun altre instrument, la forma del seu envolvent hauria de ser semblant al de la guitarra clàssica, si no, el programa podria no funcionar o donar resultats erronis.

Un cop processat el codi, els gràfics es guarden automàticament a les corresponents carpetes. Com podem veure a la Figura 37, hi ha 3 carpetes:

- figures_guitar FLAMENCO: és la carpeta on es guarden les figures amb els formats *.eps* i *.tif*.
- guitar FLAMENCO: és la carpeta on es guarden els arxius d'àudio amb el format *.wav*.
- notes_guitar FLAMENCO: és la carpeta on es guarden la matriu que conté el nom de les notes.

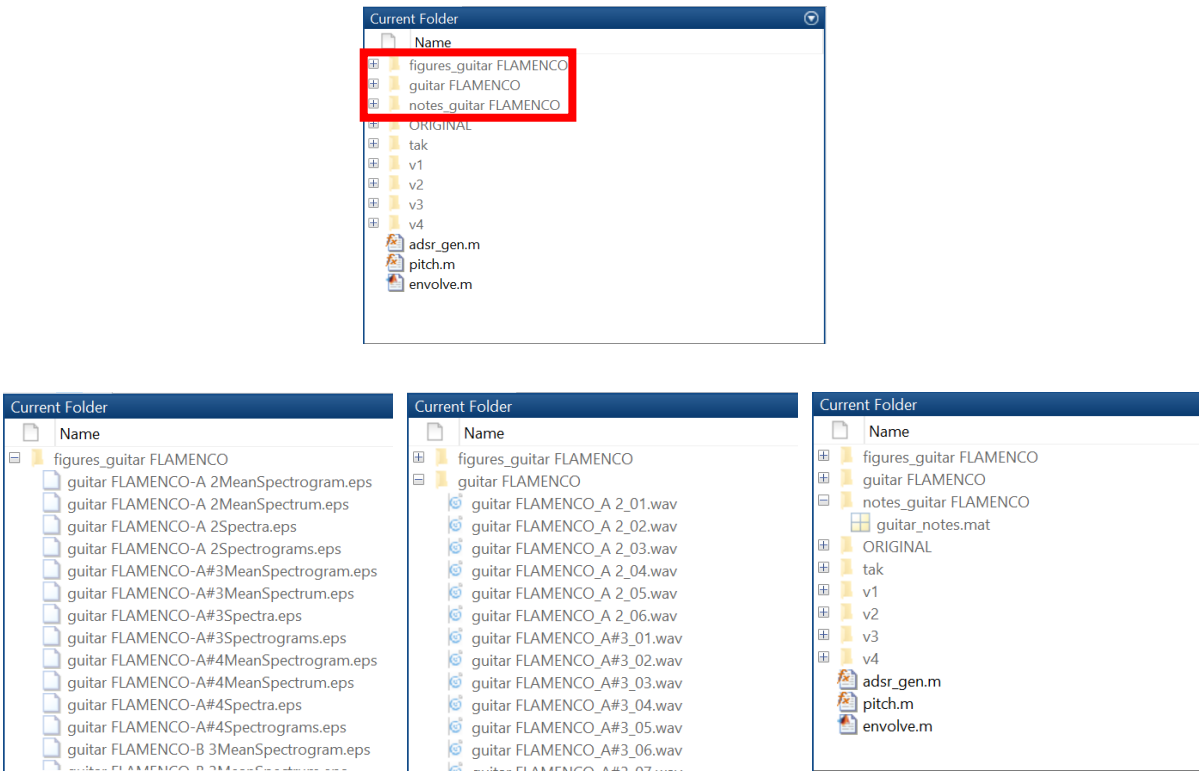

Figura 37: Nom de les carpetes creades i fitxers que contenen dins.

A partir dels arxius que contenen aquestes carpetes, la interfície pot funcionar correctament, ja que els gràfics i el nom de les notes de les parts 3 i 4 s'extreuen d'aquests arxius.

Per veure un exemple de l'aplicació del programa, en primer lloc es compararà l'evolució dels harmònics mitjançant l'espectrograma de tres notes de diferent freqüència d'una mateixa guitarra i, en segon lloc, l'evolució dels harmònics de les notes anteriors per dos models de guitarra diferents. Com s'ha explicat a l'apartat 2.4.5, com més intens és el color, més intensitat té la freqüència en qüestió.

En aquest cas, tenim un exemple de la nota E2, D3 i E5, les quals han estat interpretades pel mateix model de guitarra, anomenat ECORONDA. La nota E2 (82.407 Hz), l'espectrograma de la qual està representat a la Figura 38, podem veure que els quatre primers harmònics, sobretot el segon i el tercer, són els que tenen més presència en els primers segons. A mesura que el temps avança, el segon harmònic, corresponent a una freqüència de 164.814 Hz, és el més present. Els harmònics més aguts apareixen però amb una intensitat molt baixa. La nota D3 (146.3 Hz), l'espectrograma de la qual està representat a la Figura 39, podem veure que els harmònics que més presència tenen al llarg del temps són el primer, corresponent a una freqüència de 146.3 Hz, i el tercer, corresponent a una freqüència de 438.9 Hz. També es pot veure que alguns harmònics més elevats hi són presents però amb menys força, igual que en cas de la nota E2. Finalment, la nota E5 (659.26 Hz) l'evolució dels harmònics queda més clara i podem afirmar que tots els harmònics del rang d'harmònics representat hi són presents i que els més greus sonen durant més estona que els aguts. Els dos primers harmònics, corresponents a freqüències de 659.29 i 1318.52 Hz, respectivament, són els que tenen més presència al llarg de la nota.

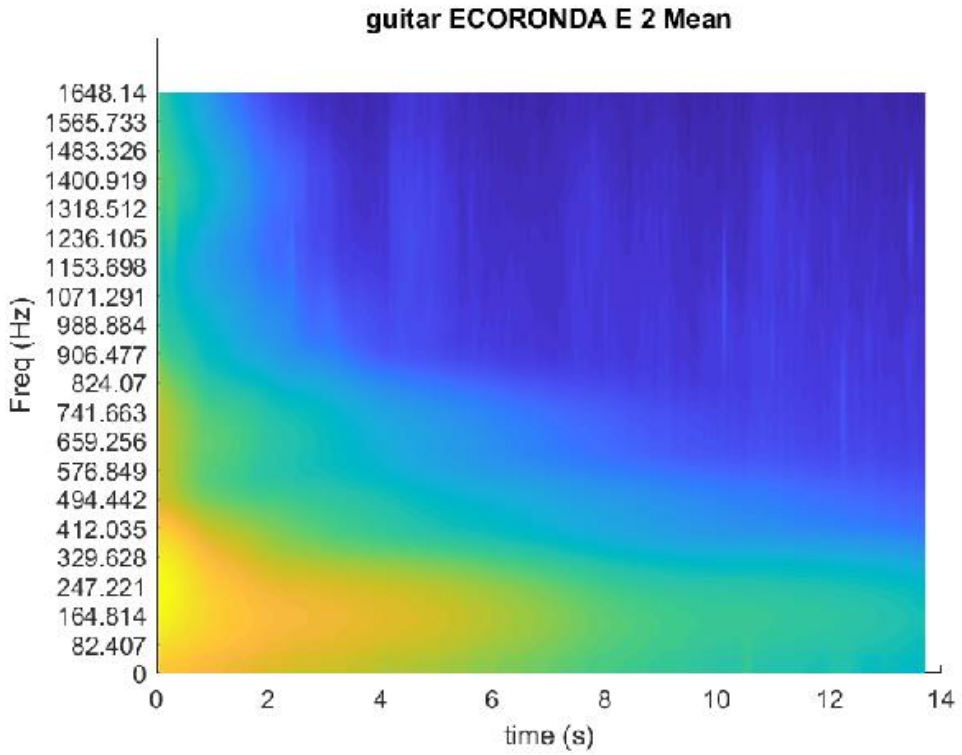

Figura 38: Espectrograma de la nota E2 interpretada per la guitarra ECORONDA.

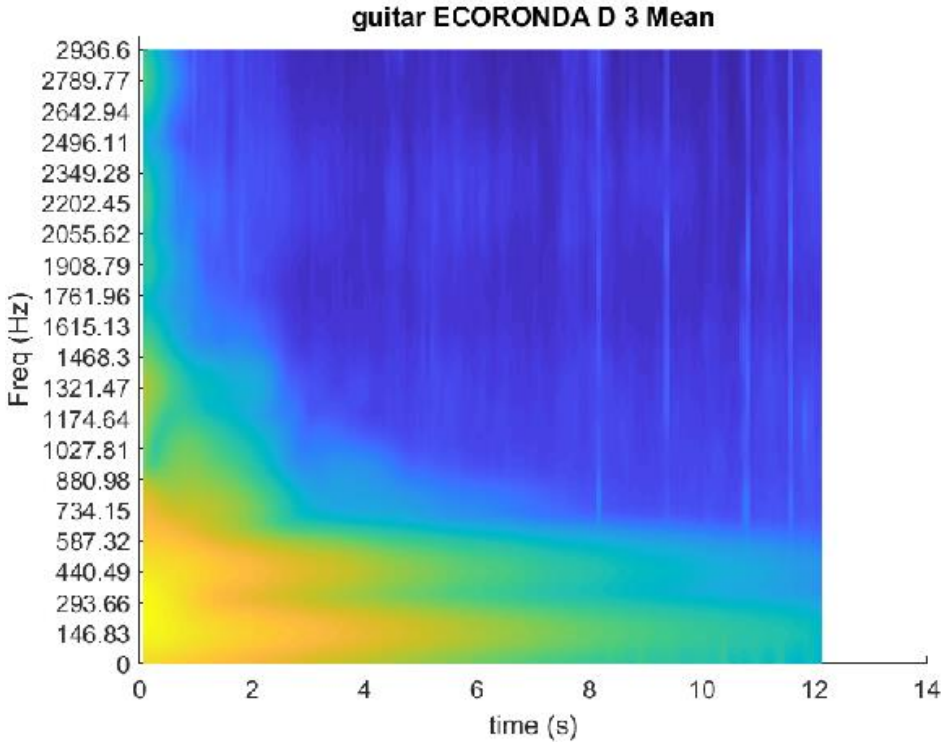

Figura 39: Espectrograma de la nota D3 interpretada per la guitarra ECORONDA.

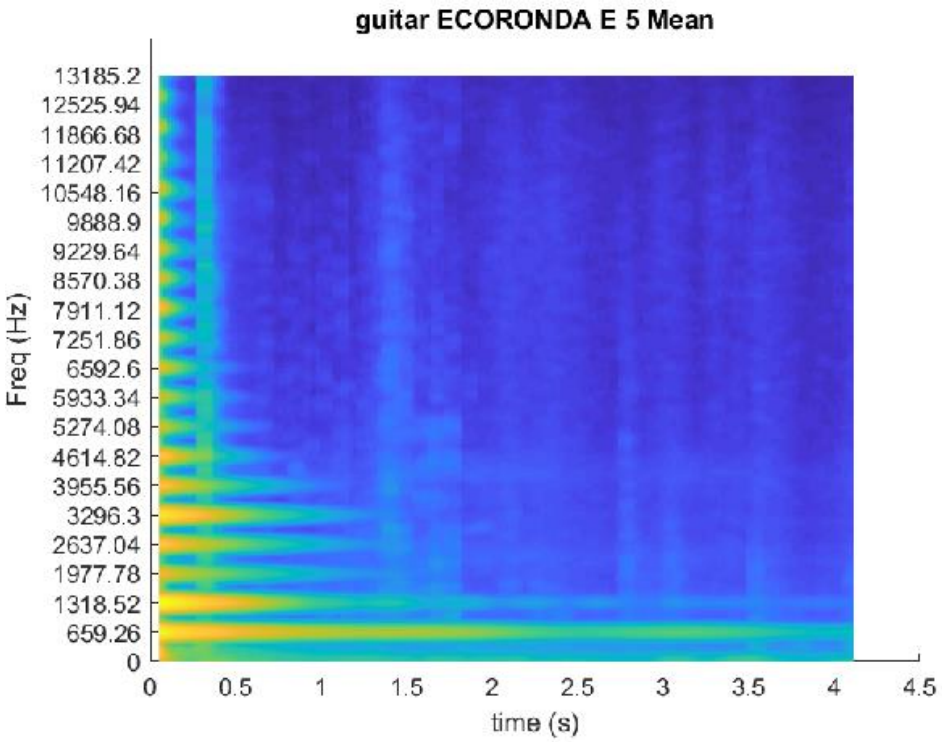

Figura 40: Espectrograma de la nota E5 interpretada per la guitarra ECORONDA.

En aquest cas, tenim un exemple de la nota E2 (82.407 Hz) interpretada per la guitarra FLAMENCO i per la guitarra M16. A la Figura 41 es pot veure que la diferència principal entre una guitarra i l'altra és que el rang d'harmònics de la guitarra FLAMENCO és molt més ampli que el de la guitarra M16. Pel que fa als harmònics greus, però, podem dir que no es pot veure cap diferència significativa. La nota D3 (146.3 Hz), en els dos casos té un rang d'harmònics bastant ampli. Si es comparen els més presents dins aquest rang, en el cas dels primers els de la guitarra FLAMENCO tenen una intensitat més gran, però l'evolució d'aquests tant d'una guitarra com de l'altra és bastant semblant. En el cas dels més alts, els quals tenen una intensitat significativament més baixa que els primers, es pot veure que hi ha certes diferències pel que fa al número d'harmònic, és a dir que a la guitarra M16 hi apareixen amb més força el vuitè i el catorzè harmònic, corresponents a freqüències de 1174.64 i 2055.62 Hz respectivament, mentre que a la guitarra FLAMENCO hi apareixen amb més força el cinquè, el vuitè i el desè, corresponents a freqüències de 734.15, 1174.64 i 1468.3 Hz respectivament. Finalment, la nota E5 (659.26 Hz) podem veure a la Figura 43 que a la guitarra M16 hi apareixen parcials que no són harmònics, a causa de la problemàtica explicada a l'apartat 3.1.2.

Això fa que la comparació entre les dues guitarres sigui impossible perquè, en el cas de la guitarra M16, s'ha realitzat la mitjana de les repeticions de dues notes de freqüències diferents.

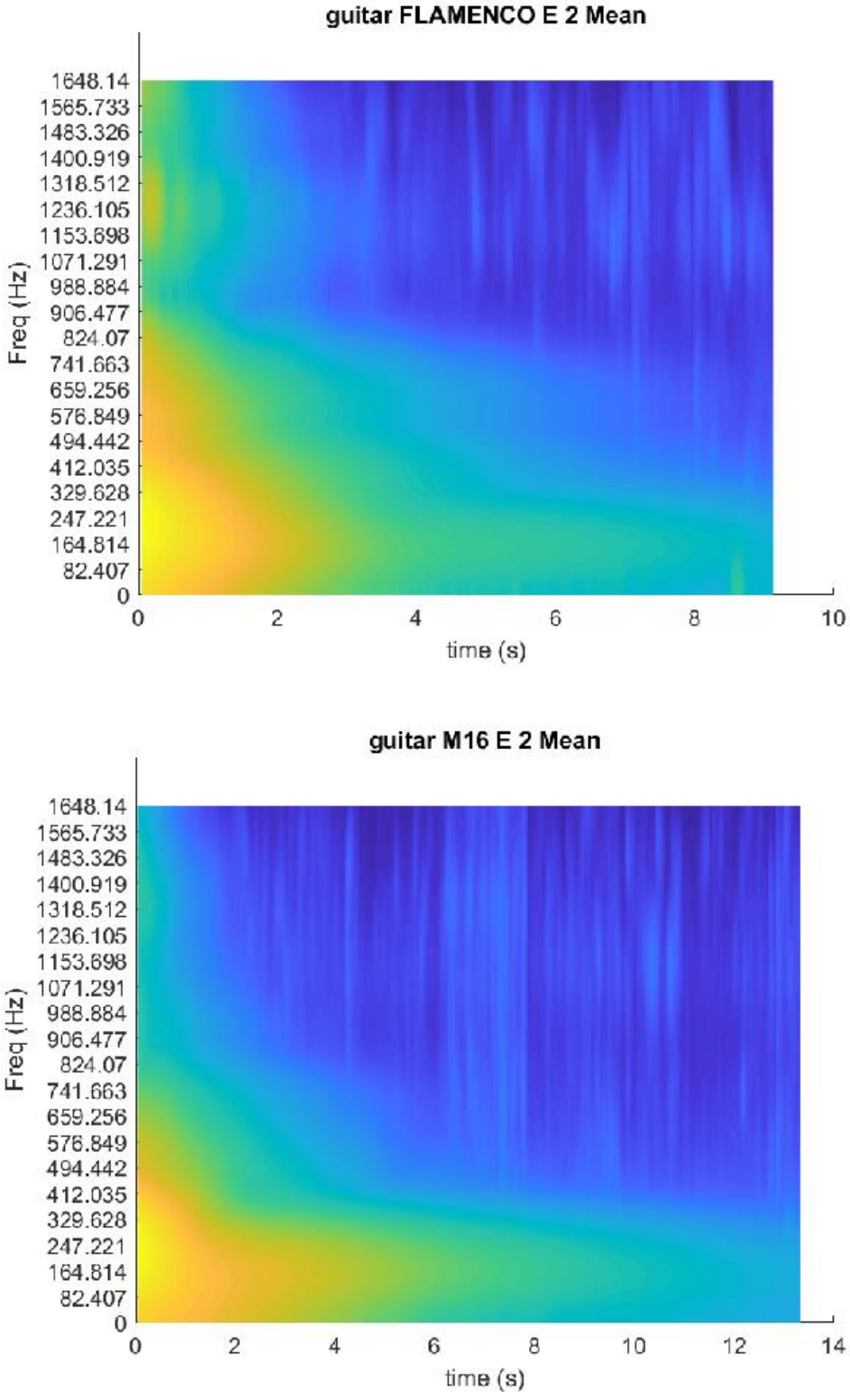

Figura 41: Espectrograma de la nota E2 de les guitarres FLAMENCO i M16.

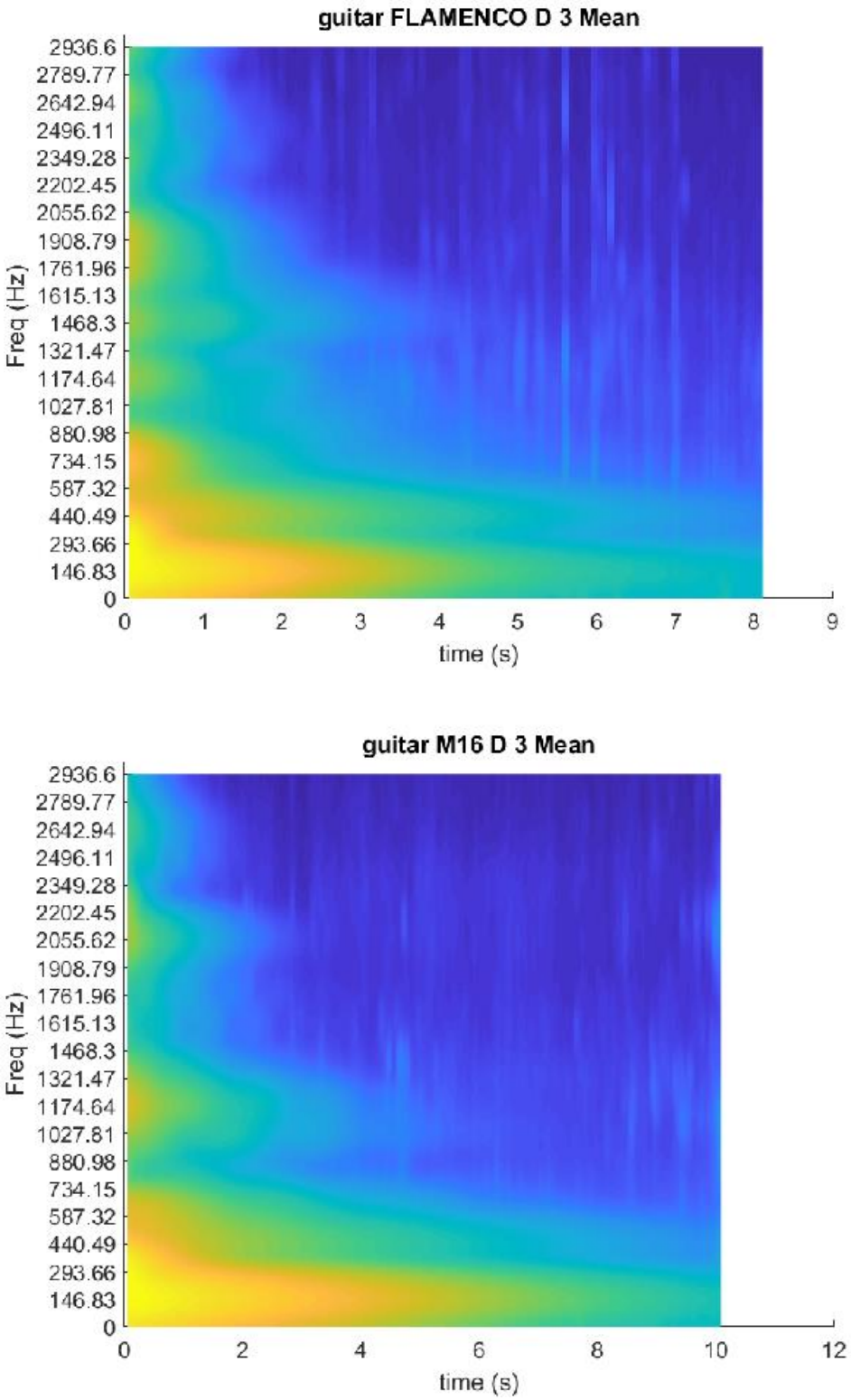

Figura 42: Espectrograma de la nota D3 de les guitarres FLAMENCO i M16.

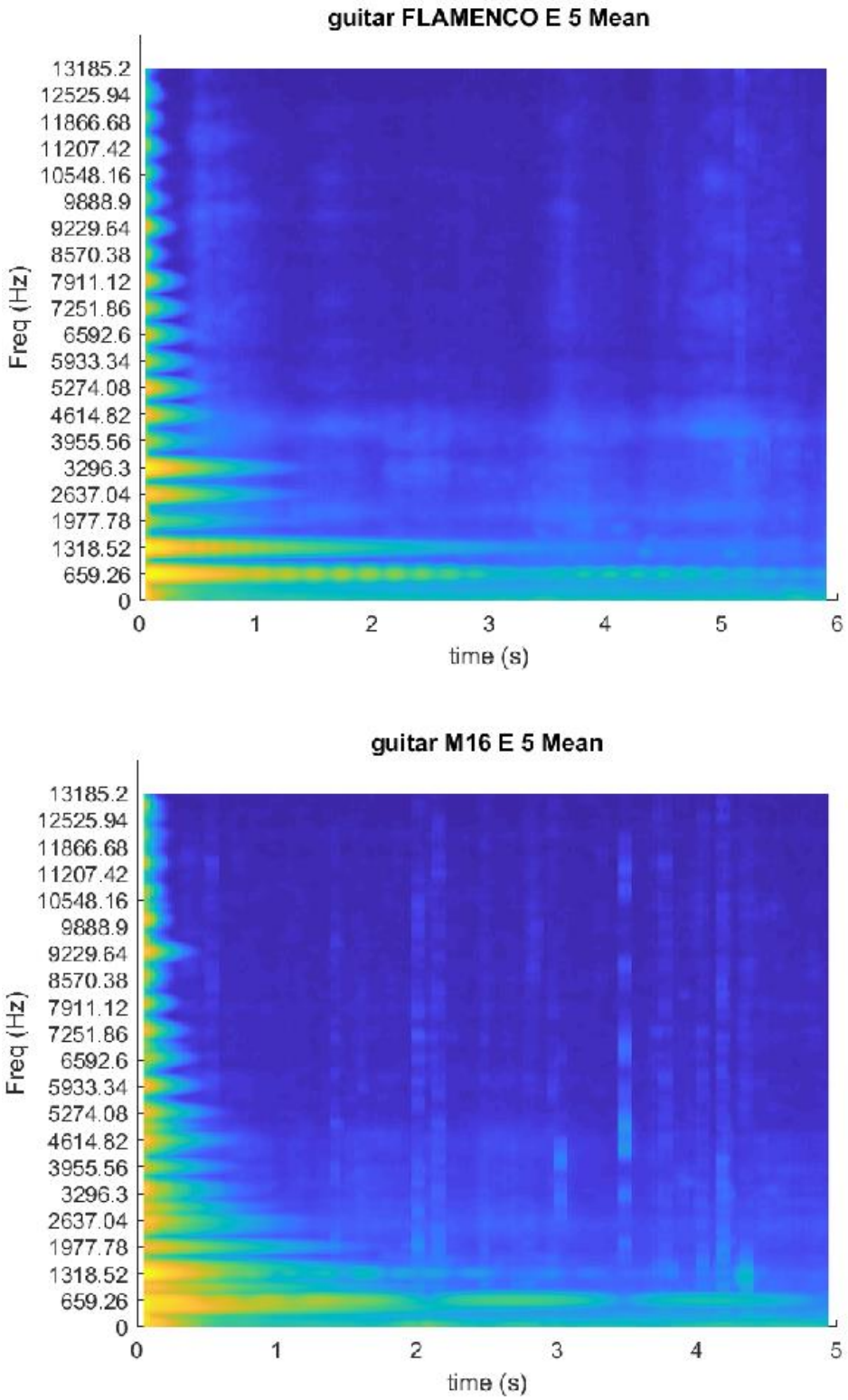

Figura 43: Espectrograma de la nota E5 de les guitarres FLAMENCO i M16.

5 RESUM DEL PRESSUPOST

En aquest apartat es realitzarà el pressupost per conèixer el cost econòmic del projecte. A la Taula 2 es pot veure la inversió de temps per cadascuna de les tasques realitzades, el cost material i el cost del personal implicat per, finalment, calcular el cost total.

Taula 2: Temps empleat per dur a terme cada tasca i temps total.

TASCA	TEMPS EMPLEAT
1. Definició dels objectes i organització del projecte.	10 h
2. Recerca de la documentació necessària.	40 h
3. Gravació dels arxius d'àudio.	8 h
4. Disseny del codi.	250 h
5. Disseny de la interfície gràfica.	150 h
6. Anàlisi i avaluació de resultats.	100 h
7. Redacció de la memòria.	150 h
TOTAL	**708 h**

Els costos personals són tots aquells aspectes relacionats amb l'execució del projecte i la gravació dels arxius d'àudio utilitzats per dur-lo a terme.

Taula 3: Costos personals.

NOM	CATEGORIA	TEMPS EMPLEAT	PREU/HORA	COST
Adrià Vila Espígol	Enginyer	700 h	30 €/h	21000 €
Lluís Costa	Músic	8 h	30 €/h	240 €
Daniel Trias Mansilla	Director projecte	50 h	40 €/h	2000 €
			TOTAL	**23.240 €**

Els costos materials són tots aquells aspectes relacionats amb la compra o lloguer d'equipaments, llicències, etc. A la Taula 4 es detallen tots els costos dels equips i les llicències utilitzats i el lloguer de la sala de gravació i les guitarres.

Taula 4: Costos materials.

MATERIAL	PREU	COST
1. Soundclub Studio	120 €/dia	120 €
2. Amortització llicència MATLAB	67 €	67 €
3. Amortització equipament informàtic	60 €	60 €
4. Impressió	16 €	16 €
5. Lloguer de les guitarres	3 x 20 €/dia	60 €
	TOTAL	**323 €**

Tant amb la llicència de MATLAB com amb l'equipament informàtic s'ha estimat el temps d'amortització. El temps d'ús de la llicència ha estat de dos mesos i mig i, el temps d'ús de

l'equipament informàtic de quatre mesos. Tenint en compte que la llicència, el preu de la qual va ser de 320 €, s'ha de renovar anualment i que l'ordinador, el preu del qual va ser de 650€, s'haurà de canviar al cap de quatre anys, els seus costos són els que hi ha reflectits a la Taula 4.

Finalment, a la Taula 5 hi ha detallat el cost total al qual s'hi ha aplicat l'impost del 21% d'IVA.

Taula 5: Cost total.

CONCEPTE	**COST**
Costos personals	23.240 €
Costos materials	323 €
Base imposable	23.563 €
21% IVA	4.948,23 €
TOTAL	**28.511,23 €**

El cost total del projecte puja a un valor de VINT-I-VUIT MIL CINC-CENTS ONZE COMA VINT-I-TRES EUROS (28.511,23 €).

6 CONCLUSIONS I POSSIBLES MILLORES

En aquest capítol es presenten les conclusions i les possibles millores que han sorgit després de la realització del projecte.

L'objectiu del projecte era poder proporcionar diferents espectres de freqüència i espectrogrames mitjançant una interfície gràfica per poder comparar diferents tipus de guitarres clàssiques. Durant el procés s'ha vist que una de les propietats del so que, principalment, defineix la qualitat d'aquest és el timbre, el qual és una propietat perceptiva. Això significa que és complicat dur-ne a terme una anàlisi. Una possible alternativa per analitzar aquesta propietat és a partir espectrogrames, els quals mostren l'evolució dels harmònics de cada nota. D'aquesta manera, un expert en aquest àmbit podria ser capaç d'analitzar si el so de la guitarra en qüestió és millor o pitjor que el d'una altra a partir d'un espectrograma i, a partir d'això, definir si és de més bona o menys bona qualitat. Els espectres de freqüència, però, no mostren una diferència tan significativa com la del cas anterior, per tant, deduïm que no són tan útils a l'hora de dur a terme aquesta anàlisi.

Hi ha quatre aspectes més que s'han de tenir en compte, els quals estan molt lligats a possibles millores.

- El primer és que la funció *regions*, la qual podem veure a l'annex B.1.4, fa que l'execució del codi sigui molt lenta. Això ens fa pensar que pot haver-hi algun altre mètode més eficient capaç de dur a terme aquesta funció.
- El segon és que hem pogut veure que per les notes més greus, el nombre d'harmònics representat no és suficient, és a dir que hauria de ser més gran, en canvi per les notes agudes el nombre d'harmònics establert, el qual és de 20, és massa alt, és a dir que hauria de ser inferior.
- El tercer és que, com que les notes han estat tocades de forma diferent, el programa processa de forma diferent les notes d'una guitarra i les d'una altra. Això provoca que, a l'hora de dur a terme comparacions d'una mateixa nota entre dues guitarres diferents, els gràfics d'una tinguin una llargada significativament diferent dels de

l'altra i dificulti la comparació. Pel que fa a la selecció de les diferents repeticions que contenen els arxius, podem dir que ha estat la part del projecte que ha portat més problemes, a causa del fet que les notes han estat tocades amb diferent intensitat i llargada i, per tant, la identificació correcta de cadascuna d'aquestes és complicada.

- I el quart és que, en algun cas puntual, la freqüència que més vegades capta la funció *pitch* no és la fonamental. Això implica que el nom de la nota identificada no sigui correcte i quan es realitzi la mitjana per representar l'espectrograma, apareixin parcials que no són harmònics, la qual cosa fa que l'espectrograma mitjà no sigui vàlid. Si el nom d'aquesta nota coincideix amb el nom d'una de les notes que si ha tocat la guitarra, és a dir, que té més d'una repetició, no hi haurà cap filtre que solucioni aquest error.

6.1 Possibles millores

Les possibles millores, les quals suposarien un canvi tan pel que fa al codi com a pel que fa a la interfície gràfica, són les següents:

- Ampliar l'anàlisi a qualsevol classe d'instrument. La dificultat principal és la de seleccionar només aquelles notes que siguin vàlides per dur a terme la corresponent anàlisi. Per poder-ho fer de forma correcta, s'haurien d'analitzar tots els tipus d'envolvents possibles i afegir un apartat a la interfície gràfica que permetés a l'usuari triar el tipus d'envolvent en funció de l'instrument que es volgués analitzar.
- Reducció del temps de càlcul del programa. Aquesta millora facilitaria molt, per exemple, el procés que ha de dur a terme el programador de buscar quins paràmetres poden proporcionar millors resultats que els actuals.
- El nombre d'harmònics hauria de variar segons la freqüència de la nota analitzada, és a dir, com més greu és la nota, més harmònics hauria de representar l'espectrograma i, com més aguda, menys.

- Quan es proporcionen els gràfics a la pestanya de comparació, la llargada de la nota comparada hauria de ser la mateixa per una guitarra que per l'altra. D'aquesta manera es podria dur a terme la comparació d'una forma més eficient.
- Aplicar un filtre que sigui capaç de solucionar la problemàtica mencionada anteriorment, la qual assigna el nom d'una nota equivocat i provoca que l'espectrograma mitjà sigui incorrecte. L'altra solució podria ser canviar la manera d'identificar la freqüència fonamental.

7 BIBLIOGRAFIA

Arfib, D., Keiler, F., Zölzer, U., Verfaille, V., & Bonada, J. (2011). DAFX: Digital Audio Effects. En *Helmut Schmidt University - University of the Federal Armed Forces, Hambur, Germany*. doi:10.1002/9781119991298.ch7

Müller, M. (2015). *Fundamentals of Music Processing*. doi:10.1007/978-3-319-21945-5

Schoeps Mikrofone. (2015). *Colette Modular System - User Guide*.

The Physics Classroom. (2019). *Fundamental Frequency and Harmonics*. Recuperat 4 juny 2019 de https://www.physicsclassroom.com/class/sound/Lesson-4/Fundamental-Frequency-and-Harmonics

Universidad Miguel Hernández de Elche. (22 març 2019). Lec001 Fundamentos de señales y sistemas (umh2277) [Vídeo]. Recuperat 4 juny 2019 de : https://www.youtube.com/watch?v=_N1CcODmvRk&list=PLClKgnzRFYe51LFskLy4Vfo6vdpdazO4r&index=1

Stack Overflow. *Respostes diverses*. Recuperat 1 de juny de 2019 de https://stackoverflow.com/

MATLAB. *Respostes diverses*. Recuperat 1 de juny de 2019 de https://es.mathworks.com/products/matlab.html

A ANNEX: MANUAL D'USUARI

A.1 Pestanya data

La primera pestanya que s'obre és la que es mostra a la Figura 44.

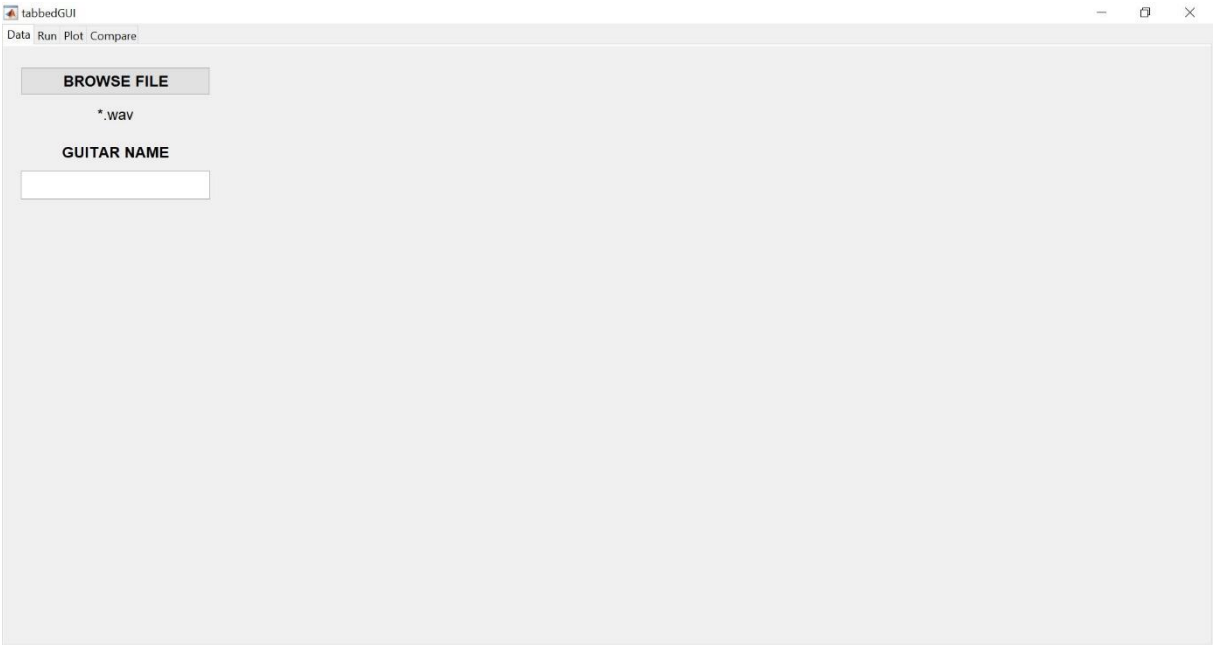

Figura 44: Pestanya *data* sense paràmetres.

Per triar l'arxiu d'àudio que es vol analitzar s'ha de polsar el botó *browse*, el qual ens permet escollit l'arxiu d'àudio en format *.wav* desitjat.

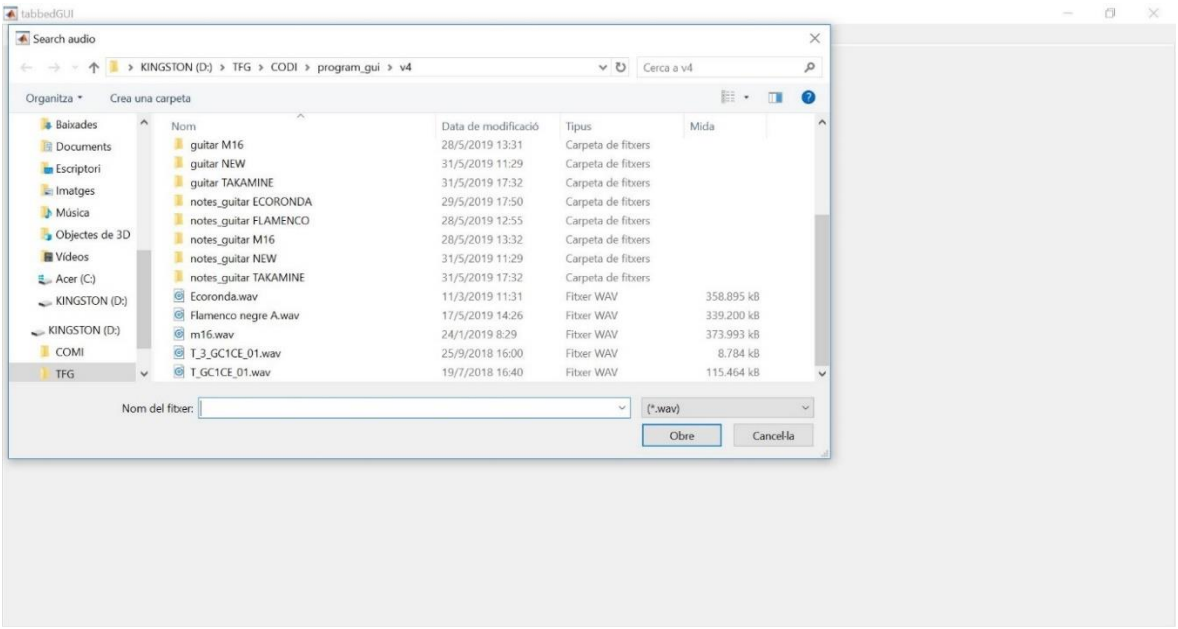

Figura 45: Pestanya *data* després de polsar el botó *browse*.

Un cop s'ha triat l'arxiu d'àudio, l'usuari pot escriure el nom desitjat per la guitarra.

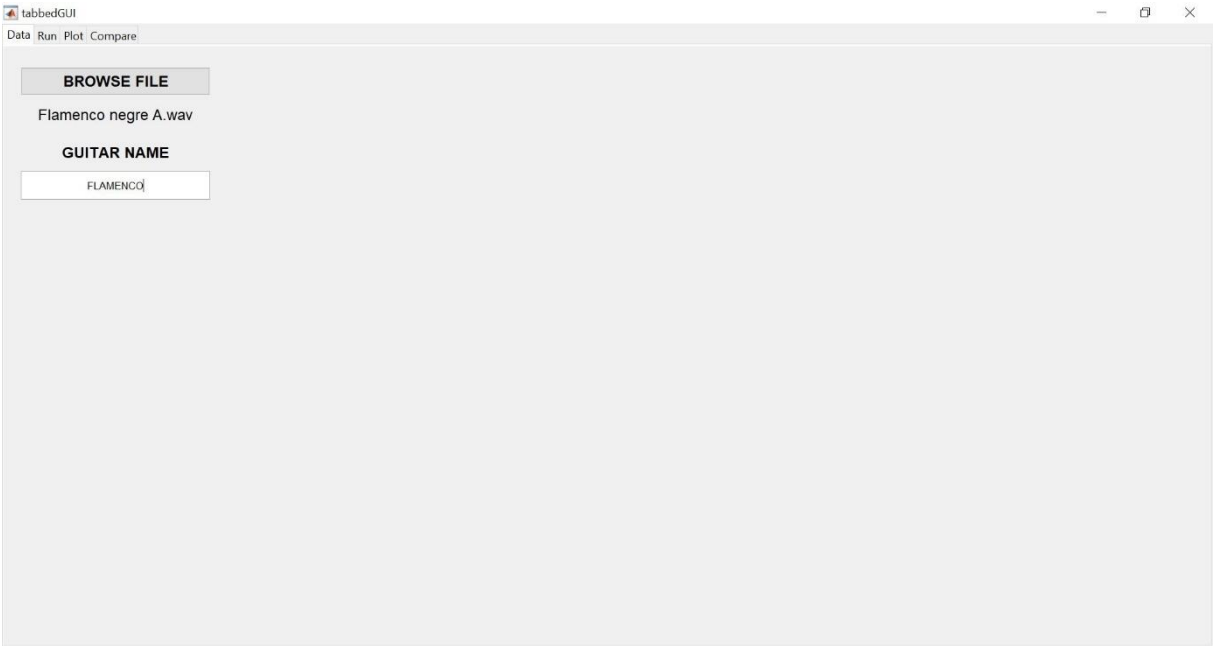

Figura 46: Pestanya *data* després d'escriure el nom de la guitarra.

A.2 Pestanya run

Un cop l'usuari ha seleccionat l'arxiu i ha escrit el nom de la guitarra, aleshores pot entrar a la pestanya *run*, la qual, únicament, disposa d'un botó. Per executar el programa que proporcionarà els gràfics desitjats s'ha de polsar el botó *run*.

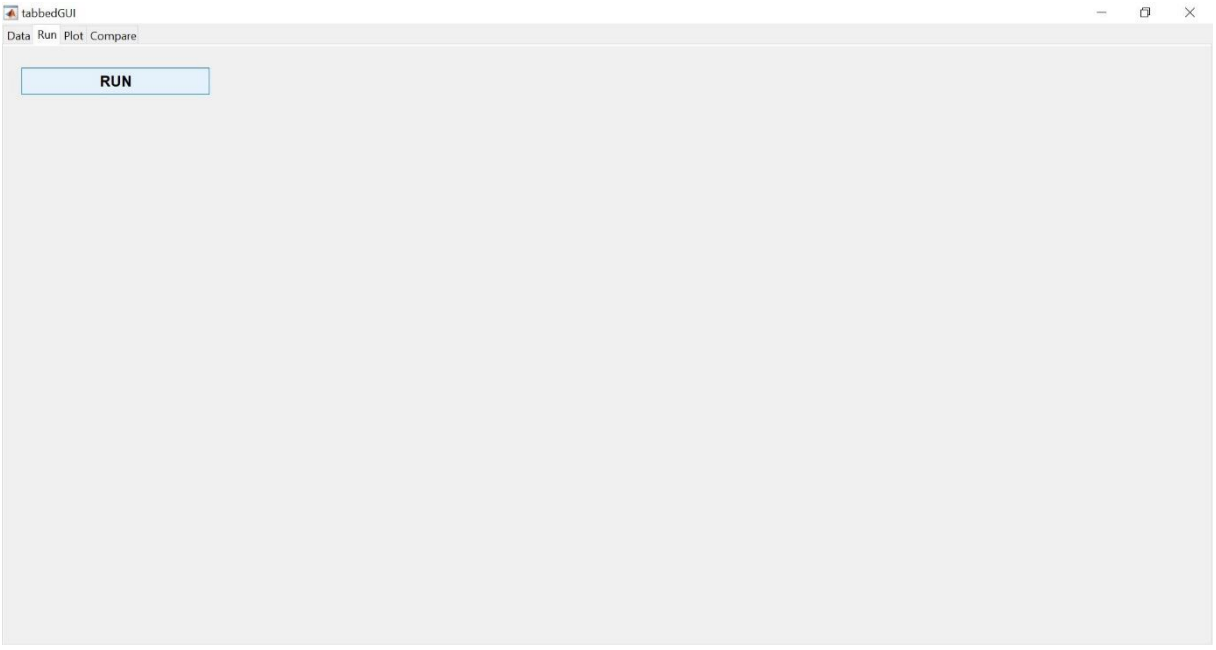

Figura 47: Pestanya *run*.

A.3 Pestanya plot

Un cop el programa s'ha executat, la interfície gràfica es reobre a la pestanya *data*. Per veure els espectres de freqüència i espectrogrames que ha calculat el programa executat al pas anterior, l'usuari ha d'entrar a la pestanya *plot*, la qual podem veure a la Figura 48.

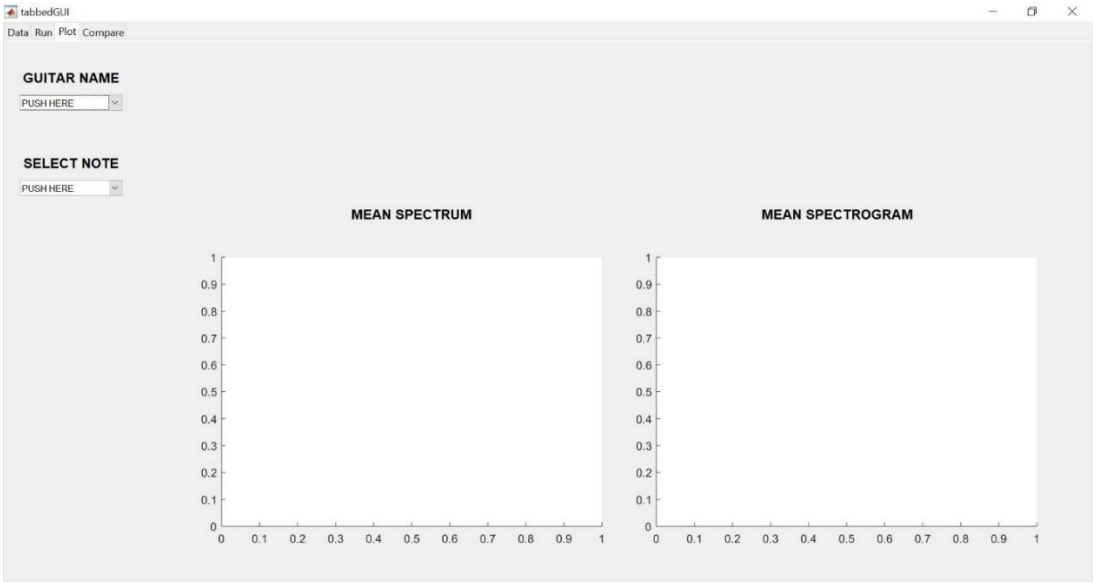

Figura 48: Pestanya *plot* sense cap paràmetre escollit.

Per veure els gràfics, l'usuari ha de seleccionar el nom de la guitarra polsant al desplegable que diu *push here*. Aleshores apareixen els noms de totes les guitarres processades, tal com es veu a la Figura 49, i l'usuari pot escollir el desitjat.

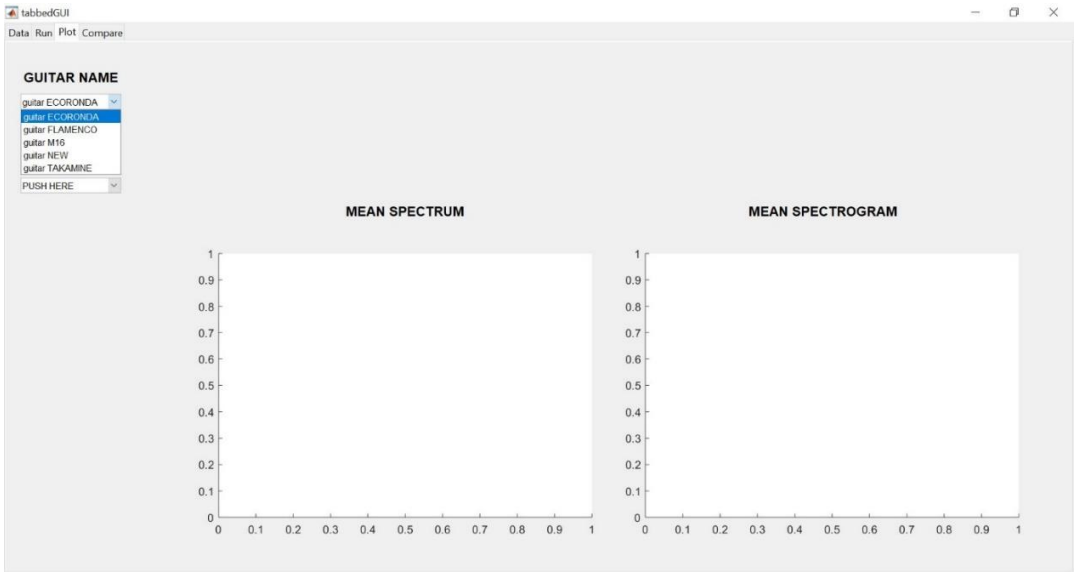

Figura 49: Pestanya *plot* després d'escollir el nom de la guitarra.

Després s'ha de seleccionar la nota que es vulgui veure seguint el mateix procediment anterior. A la Figura 50 es pot veure com apareixen els noms de les notes després de polsar el desplegable on diu *push here*. Un cop s'ha seleccionat la nota, la interfície gràfica mostra l'espectre de freqüències mitjà al gràfic de l'esquerra i l'espectrograma mitjà a la dreta.

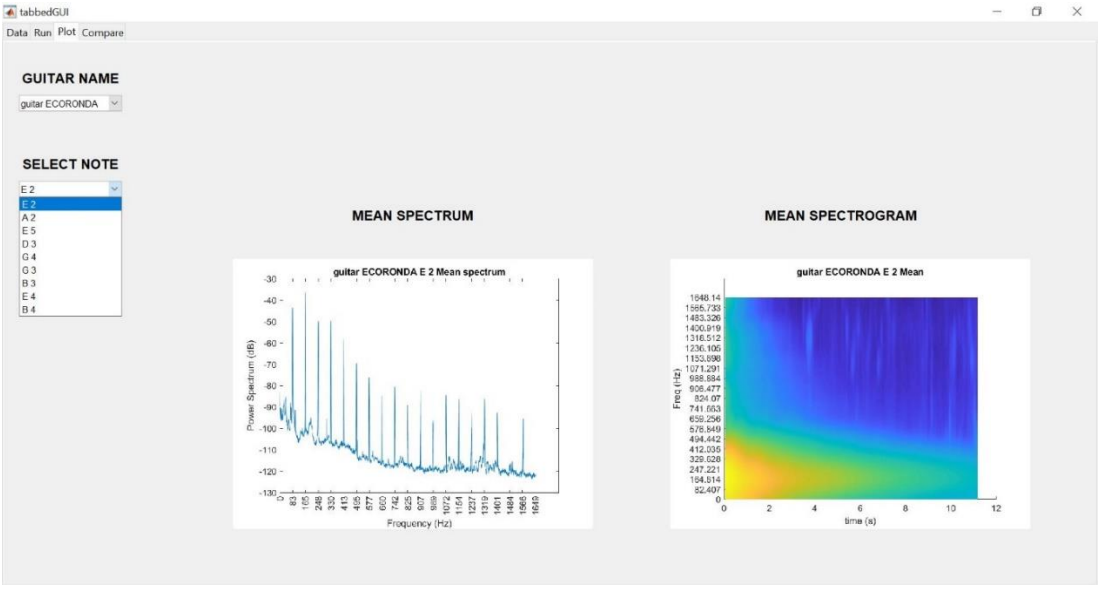

Figura 50: Pestanya *plot* després de seleccionar la nota.

A.4 Pestanya compare

Finalment, si l'usuari vol comparar una mateixa nota per dos tipus de guitarres diferents, ha d'entrar a la pestanya *compare*.

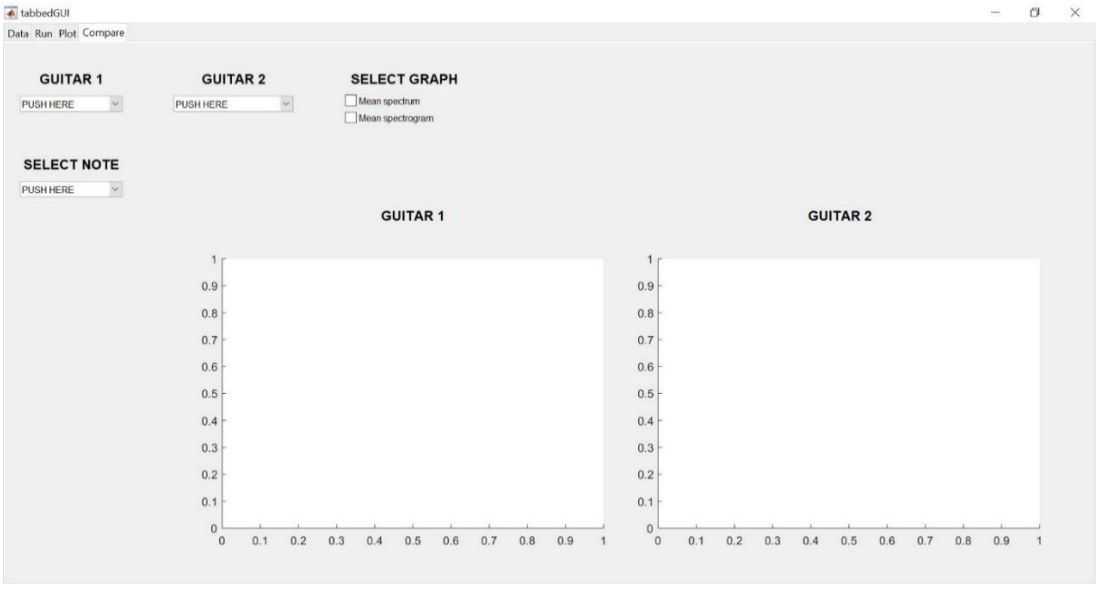

Figura 51: Pestanya *compare* sense cap paràmetre escollit.

En aquest cas, el primer que ha de fer l'usuari és triar els noms de les dues guitarres, de la mateixa manera que ho ha fet a la pestanya *plot*.

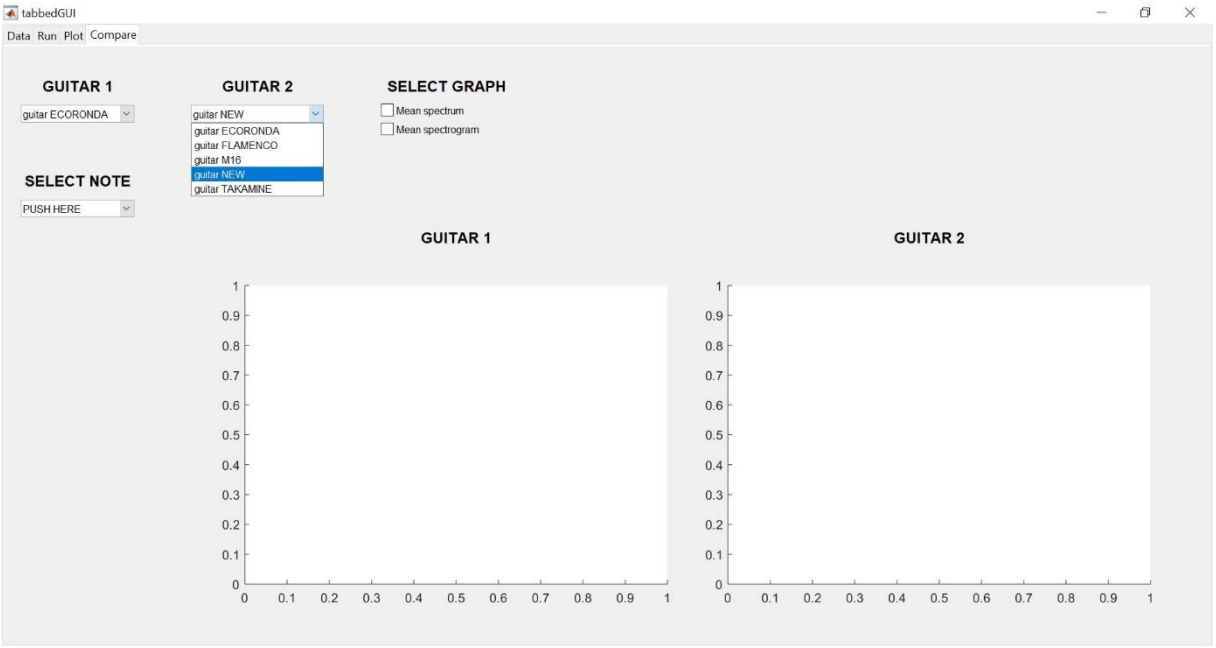

Figura 52: Pestanya *compare* després d'escollir els noms de les guitarres.

Un cop s'han escollit les guitarres que es volen comparar, s'ha d'escollir el tipus de gràfic que es vol veure. A la Figura 54 s'ha seleccionat l'espectrograma i a la Figura 55 l'espectre de freqüències.

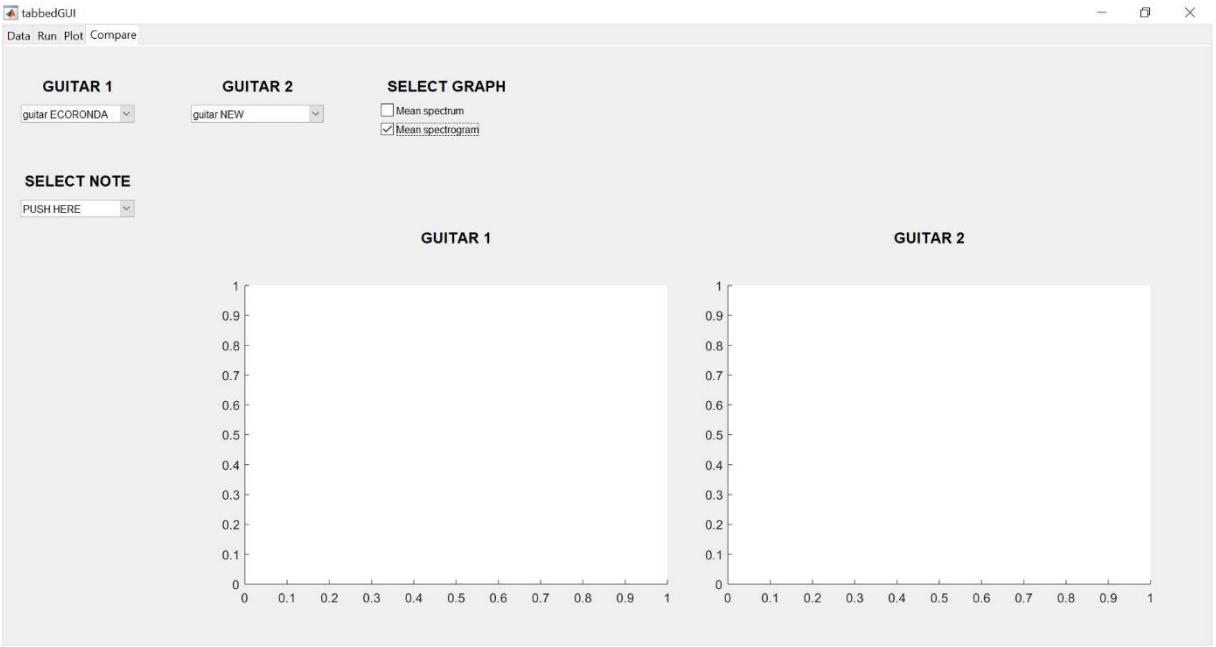

Figura 53: Pestanya *compare* després d'escollir el tipus de gràfic.

Finalment, s'ha de seleccionar la nota que es vol comparar. Per seleccionar-la també s'ha de seguir el mateix procediment que s'ha seguit a la pestanya *plot*. Un cop seleccionada la nota apareixen els gràfics desitjats d'aquesta per cadascuna de les guitarres seleccionades.

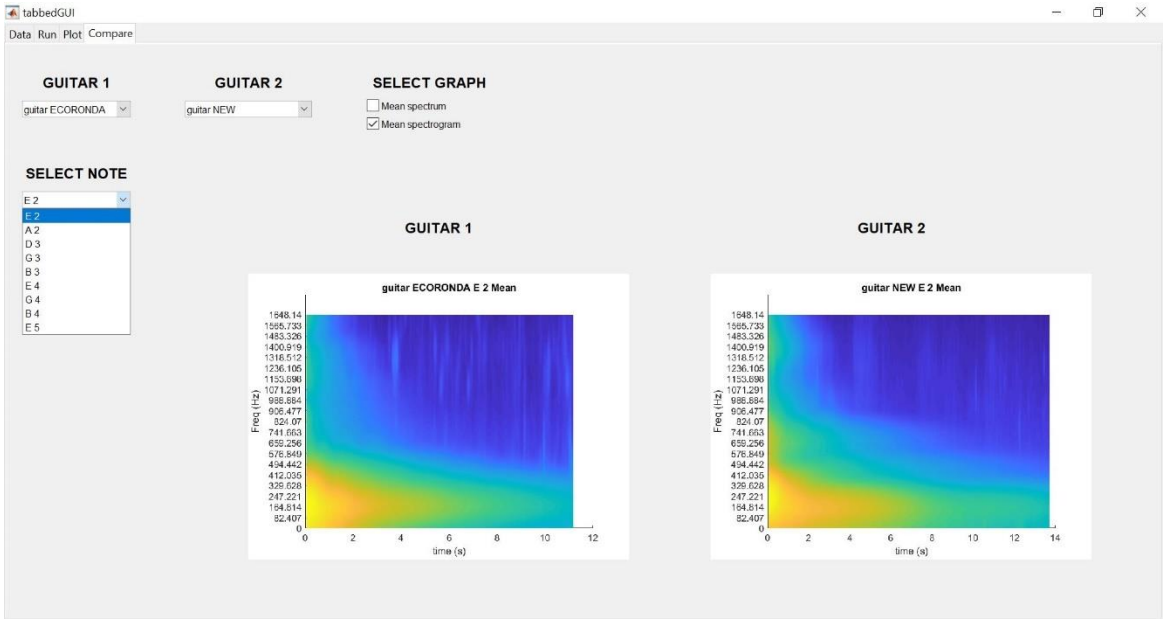

Figura 54: Pestanya *compare* després d'escollir la nota i espectrograma.

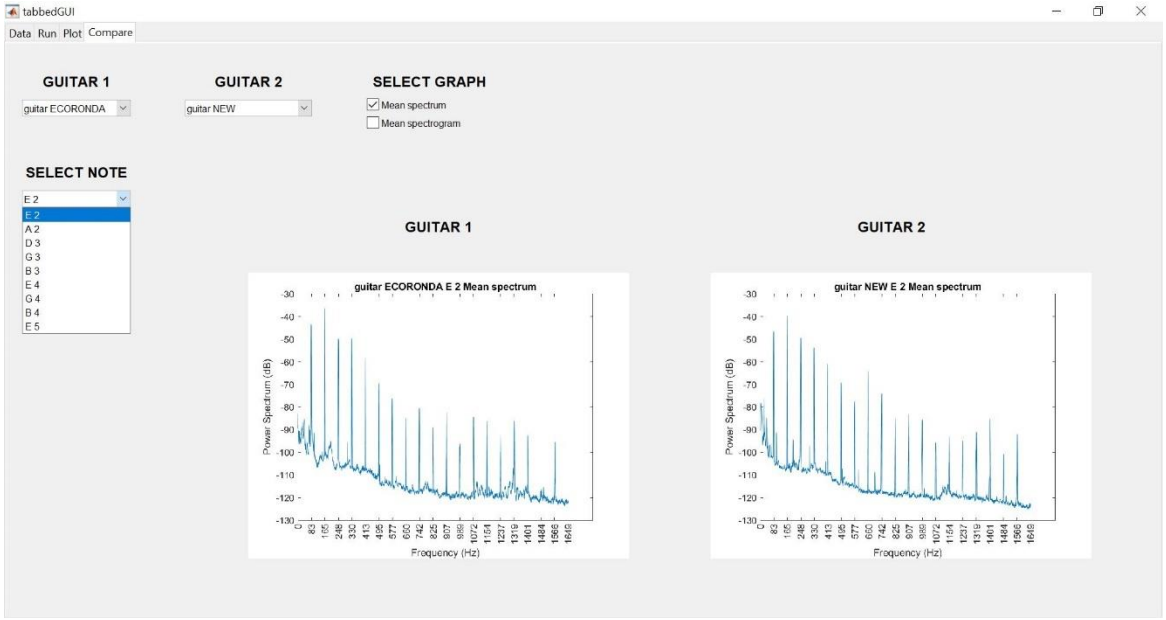

Figura 55: Pestanya *compare* després d'escollir la nota i espectre.

B ANNEX: CODI DEL PROGRAMA I DE LA INTERFÍCIE GRÀFICA

En aquest annex trobarem les diferents parts del codi de què s'ha anat parlant al llarg de la secció de MARC PRÀCTIC, així com el codi que permet que la interfície gràfica funcioni. S'ha de tenir en compte que les llibreries i funcions utilitzades estan incorporades al programa MATLAB.

Per tal de facilitat la lectura del codi, s'ha decidit que la tipologia i el color del text siguin els mateixos que apareixen al programa. D'aquesta manera el text quedarà més compacte.

B.1 Codi del programa

```
function batch_analysis_v4(audio_file,guitar_name)

% Definició tipus de guitarra.
guitars = guitar_name;
figs = fullfile(pwd,'/',sprintf('figures_%s',guitars));
notes_folder = fullfile(pwd,'/',sprintf('notes_%s',guitars));

% Definim la carpeta inicial.
oldFolder = pwd;

% Definim les notes amb les corresponents freqüències fonamentals.
[notes,fund_freqs] = database_freqs;

% Llegir el fitxer d'àudio
[y,Fs]=audioread(audio_file);

% Es seleccionen les notes tocades i es classifiquen.
[M, guitar_notes] = cut_audio(guitars, fund_freqs, notes, y, Fs,
oldFolder);

% Creeem la matriu de notes i freqüències
notes_freqs(guitar_notes, notes_folder, oldFolder, guitars);

% Definim el temps, la resolucio de frequencia i el nombre d'harmonics.
tmaxs = M/Fs;
freq_resols=[1;1;1;1;1;1;1;1;1];
n_harm = 20;

% Espectres i espectrogrames
specs(tmaxs,freq_resols,n_harm,guitar_notes,guitars,fund_freqs,notes,figs,o
ldFolder)

end % Ja s'han analitzat totes les repeticions de totes les notes.
```

B.1.1 database_freqs

```
function [notes,fund_freqs] = database_freqs
```

```
notes = ["E 2";"F 2";"F#2";"G 2";"G#2";"A 2";"A#2";"B 2";"C 3";"C#3";"D
3";"D#3";"E 3";"F 3";"F#3";"G 3";"G#3";"A 3";"A#3";"B 3";"C 4";"C#4";"D
4";"D#4";"E 4";"F 4";"F#4";"G 4";"G#4";"A 4";"A#4";"B 4";"C 5";"C#5";"D
5";"D#5";"E 5"];
fund_freqs =
[82.407,87.307,92.499,97.999,103.83,110,116.54,123.47,130.81,138.59,146.83,
155.56,164.81,174.61,185,196,207.65,220,233.08,246.94,261.63,277.18,293.66,
311.13,329.63,349.23,369.99,392,415.30,440,466.16,493.88,523.25,554.37,587.
33,622.25,659.26];

end
```

B.1.2 cut_audio

```
function [M, guitar_notes] = cut_audio(guitars, fund_freqs, notes, y, Fs,
oldFolder)

    %% PREPROCÉS
    % Volem separar l'audio en tantes parts com repeticions hi hagi.
    removeNotes(guitars)
    % Definició dels paràmetres mínima durada dels àudios.
    duration = 4;
    [num_notes] = regions(y, Fs, oldFolder, guitars, duration);

    %% POSTPROCÉS
    % Calculem spectrum de cadascuna de les notes per trobar quina nota és
    [guitar_notes]=identifyNotes(num_notes,fund_freqs,notes,guitars,
    oldFolder);
    % Mateixa llargada per poder fer mitjanes
    [M, guitar_notes] = sameLength(guitar_notes, guitars);

    pwd
end
```

B.1.3 removeNotes

```
function removeNotes(guitars)

    % ELIMINA LA CARPETA NOTES I EN CREA UNA DE NOVA.
    % Llista de tots els arxius de la carpeta.
    currentFolder = pwd;
    guitar_figures = sprintf('figures_%s',guitars);
    notes = sprintf('notes_%s',guitars);
    files = dir;
    % Vector que ens indica quins dels arxius són carpetes.
    dirFlags = [files.isdir];
    % Seleccionem només les carpetes.
    subFolders = files(dirFlags);
    % Busquem la carpeta notes i l'eliminem.
    for i=1:length(subFolders)
       s1 = subFolders(i).name;
       s2 = guitars;
       tf = strcmp(s1,s2);
       if tf == 1
           rmdir(s2, 's')
       end
    end
    mkdir(s2)
```

```
    % Eliminem carpeta figures (si existeix)
    for i=1:length(subFolders)
       s1 = subFolders(i).name;
       s2 = guitar_figures;
       tf = strcmp(s1,s2);
       if tf == 1
           rmdir(s2, 's')
       end
    end
    mkdir(s2)

    % Eliminem carpeta notes (si existeix)
    for i=1:length(subFolders)
       s1 = subFolders(i).name;
       s2 = notes;
       tf = strcmp(s1,s2);
       if tf == 1
           rmdir(s2, 's')
       end
    end
    mkdir(s2)

end
```

B.1.4 regions

```
function [num_notes] = regions(y, Fs, oldFolder, guitars, duration)

guitar_1 = guitars;
signal = y;
% Apliquem el filtre al senyal d'àudio
filteredSignal = sgolayfilt(abs(signal), 2, 2001);
nota={};
a = 1;
L = length(filteredSignal);
i = 1;

% Busca les notes de nombre negatiu a nombre negatiu.
while i < L
    A = find(filteredSignal(i:L) < 0, 1,'first');
    if i == 1
        A = A;
    else
        A = i + A;
    end
    B_1 = find(filteredSignal(A:L) > 0, 1, 'first');
    B_1 = A + B_1;
    B_2 = find(filteredSignal(B_1:L) < 0, 1, 'first');
    if isempty(B_2) == 0
        B_2 = B_1 + B_2;
        nota{a} = y(A:B_2);
        a = a + 1;
        i = B_2;
    else
        B_2 = find(filteredSignal > 0, 1, 'last');
        nota{a} = y(A:B_2);
        i = L;
    end
end
```

```
% Calcula el nombre de canvis bruscs i la posició de l'últim canvi.
L = length(nota);
for i=1:L
    envelope = imdilate(abs(nota{i}), true(1501, 1));
    TF = ischange(envelope,'linear','Threshold',1);
    percent = find(TF==1, 1, 'last')/length(TF)*100;
    if isempty(percent) == 0
        P(i) = find(TF==1, 1, 'last')/length(TF)*100;
    else
        P(i) = 0;
    end
    T(i) = numel(find(TF==1));
end

nota_1 = {};
a = 1;

% Filtre selecció de notes.
for i=1:L
    if (3 < T(i)) && (T(i) < 11) && (P(i) < 65)
        nota_1{a} = nota{i};
        a = a + 1;
    else
        a = a;
    end
end

L = numel(nota_1);
% Creem un fitxer d'audio (.wav) per cada repetició a una nova carpeta.
% Filtre selecció notes curtes.
a = 1;
for i=1:L
    if a<10
        fnm = sprintf('%s_00%d.wav',guitar_1,a);
    elseif a<100
        fnm = sprintf('%s_0%d.wav',guitar_1,a);
    else
        fnm = sprintf('%s_%d.wav',guitar_1,a);
    end
    fnm_ = strcat(pwd, '\', guitar_1, '\', fnm);
    audiowrite(fnm_,nota_1{i},Fs);
    info = audioinfo(fnm_);
    if info.Duration < duration
        delete(fnm_)
        a = a;
    elseif info.Duration > duration
        a = a + 1;
    end
end

cd(oldFolder)
folder = dir(strcat(pwd, '\', guitar_1, '\', '*.wav'));
num_notes = numel(folder);

end
```

B.1.5 identifyNotes

```
function [guitar_notes] = identifyNotes(num_notes, fund_freqs, notes,
guitars, oldFolder)
    % IDENTIFICA LA NOTA TOCADA MITJANÇANT LA FFT
    b = 1;
    % Creem una matriu que ens proporcioni les notes tocades.
    guitar_notes = string;
    str = string;

    for i=1:num_notes
        files = dir;
        dirFlags = [files.isdir];
        % Seleccionem només les carpetes.
        subFolders = files(dirFlags);
        guitar_1 = deblank(guitars);
        for j=1:length(subFolders)
            s1 = guitar_1;
            s2 = subFolders(j).name;
            n = min(numel(s1),numel(s2));
            result = sum(cumprod(s1(1:n)==s2(1:n)));
            if result == length(guitar_1)
                cd(fullfile(pwd, guitar_1));
            end
        end

        if i<10
            fnm = sprintf('%s_00%d.wav',guitar_1,i);
        elseif i<100
            fnm = sprintf('%s_0%d.wav',guitar_1,i);
        else
            fnm = sprintf('%s_%d.wav',guitar_1,i);
        end

        currentFolder = pwd;
        [x, Fs] = audioread(fnm);
        cd(oldFolder)
        [~,pos] = note(x,Fs,fund_freqs);

        cd(currentFolder)

        % Reinicialitza la numeració quan canvia la nota.
        if str == notes(pos)
            a = a;
        else
            a = 1;
        end

        mypath = strcat(pwd, '\');
        names = dir(mypath);
        names([names.isdir]) = [];
        fileNames = {names.name};
        str = notes(pos);
        g = fullfile(mypath, fileNames{1});

        % Les notes repetides no les tornem a guardar.
        reps = find(str == guitar_notes);
        if reps > 0
            b = b;
        else
            guitar_notes(b) = str;
```

```
            b = b + 1;
        end

        % Guardar el fitxer anterior afegint la nota que és.
        if a<10
            newName = sprintf('%s_%s_0%d.wav', guitar_1, str, a);
        else
            newName = sprintf('%s_%s_%d.wav', guitar_1, str, a);
        end

        a = a + 1;

        f = fullfile(mypath, newName);
        movefile(g,f);
    end


 cd(oldFolder)
```

B.1.6 note

```
function [nota,pos] = note(x,Fs,fund_freqs)

% Paràmetres funció pitch
windowLength = round(0.052*Fs);
overlapLength = round(0.042*Fs);
range = [50,700];
[f0,idx] =
pitch(x,Fs,'WindowLength',windowLength,'OverlapLength',overlapLength,'Metho
d','NCF','Range',range);

% Histograma
[h,data] = histcounts(f0, unique(f0));
for i=1:length(data)
    [minim,pos] = min(abs(fund_freqs-data(i)));
    max_prox = fund_freqs(pos);
    nota = max_prox;
    pos_1(i) = nota;
end
data = pos_1;
[~,idx] = sort(-h);
h(idx);
p = unique(data(idx),'stable');
nota = p(1);
[~,pos] = min(abs(fund_freqs-p(1)));

End
```

B.1.7 sameLength

```
function [M, guitar_notes] = sameLength(guitar_notes, guitars)

guitar_type=deblank(guitars);
len=[];
M = [];
inote = 1;
% Creem una matriu (len) amb la llargada mínima de cada repetició.
% Creem una matriu (M) amb la llargada mínima de totes les repeticions.
```

```
while inote < length(guitar_notes) + 1
    note=guitar_notes(inote); % note to analyse
    str=sprintf('./%s/%s_%s_*.wav',guitar_type,guitar_type,note);
    list=dir(str);
    n_files=length(list);
    if n_files < 2
        guitar_notes(inote) = [];
        str=sprintf('./%s/%s_%s_0%i.wav',guitar_type,guitar_type,note,
        n_files);
        delete(str)
    else
        for i=1:n_files
            if i<10
                file=sprintf('./%s/%s_%s_0%i.wav',guitar_type,guitar_type,
                note,i);
            else
                file=sprintf('./%s/%s_%s_%i.wav',guitar_type,guitar_type,
                note,i);
            end
            [y, ~] = audioread(file);
            len(i,1) = length(y);
        end
        M(inote) = min(len);

        % Cadascuna de les repeticions la reescrivim amb la nova mida.
        for i=1:n_files
            if i<10
                file=sprintf('./%s/%s_%s_0%i.wav',guitar_type,guitar_type,
                note,i);
            else
                file=sprintf('./%s/%s_%s_%i.wav',guitar_type,guitar_type,
                note,i);
            end
            [y, Fs] = audioread(file);
            y1 = y(1:M(inote));
            audiowrite(file,y1,Fs);
        end
        inote = inote + 1;
    end
end
```

B.1.8 specs

```
function specs(tmaxs,freq_resols,n_harm,guitar_notes,guitars,fund_freqs,
notes,figs,oldFolder)

% Per cada nota es creen les corresponents característiques (temps, f...)
for inote=1:length(guitar_notes)
    % Nota que es vol analitzar.
    note=guitar_notes(inote);
    freq_resol=freq_resols(1);
    % Definició del temps límit (màxim).
    t_max=tmaxs(inote);
    % Definició de la freqüència límit.
    fund_freq=fund_freqs(find(notes == guitar_notes(inote)));
    limits=[0 n_harm*fund_freq];

    % Nombre d'arxius disponibles per cada nota (repeticions).
    guitar_type=deblank(guitars);
    str=sprintf('./%s/%s_%s_*.wav',guitar_type,guitar_type,note);
```

```
list=dir(str);
n_files=length(list);
% Creació de matrius per guardar les dades dels espectres.
pxx_=[];
f_=[];

% Per cadascuna de les repeticions de cada nota.
for i=1:n_files
    if i<10
        file=sprintf('./%s/%s_%s_0%i.wav',guitar_type,guitar_type,
        note,i);
    else
        file=sprintf('./%s/%s_%s_%i.wav',guitar_type,guitar_type,
        note,i);
    end

    [y, Fs] = audioread(file);
    % Divisió dels samples en instants de temps. Cada sample és un
    instant de temps.
    t = (1:length(y))/Fs;
    % Agafem només els valors que no superin el límit de temps
    establert(tots mateixa mida).
    y_tmax=y(t<t_max);

    % Analitzem el senyal en l'àmbit freqüencial i obtenim l'espectre
    (pxx) amb les corresponents freqüències (f).
    [pxx,f] = pspectrum(y_tmax,Fs,'FrequencyResolution',freq_resol,
    'FrequencyLimits',limits);

    % Guardem les dades de l'anàlisi fet amb el pspectrum.
    pxx_(i,:) = pxx;
    f_(i,:) = f;

    % Calculem l'espectrograma mitjançant el pspectrum i obtenim
    l'espectre (p) amb les corresponents freqüències (f1) i temps (t1)
    [p,f1,t1] = pspectrum(y_tmax,Fs,'spectrogram','FrequencyLimits',
    limits,'TimeResolution',0.01);
    % Guardem, per cada repetició, les variables p, f1 i t1.
    Spectdata_(i).f = f1;
    Spectdata_(i).t = t1;
    Spectdata_(i).p = p;

end

%% Plots for Spectrum

% Creem una matriu per guardar els valors mitjans dels espectres.
pxx_mean = zeros(1,size(pxx(1,:),2));
% Creem una matriu per saber el nombre de divisions que hem de fer a
l'hora de plotejar els espectres.
nsubplot = ceil(sqrt(n_files));

% Per cada repetició plotegem els espectres.
for i=1:n_files

    subplot(nsubplot,nsubplot,i)


    % Agafem les dades emmagatzemades anteriorment i comencem a
    construir la matriu mitjana.
    f = f_(i,:);
```

```
        pxx = pxx_(i,:);
        pxx_mean = pxx_mean + pxx;

        % Plotegem (freqüència vs dB -> espectre) de cadascuna de les
        repeticions.
        plot(f,pow2db(pxx))
        % Figure Title
        tit=sprintf('%s %s %i',guitar_type,note,i);
        title(tit)
        % Calculem el nombre d'harmònics desitjat i fem que l'eix x prengui
        aquests valors.
        xt=[0:fund_freq:fund_freq*n_harm];
        xticks(ceil(xt));
        % Create ylabel
        ylabel('Power Spectrum (dB)');
        % Create xlabel
        xlabel('Frequency (Hz)');

    end

    % Create a new folder
    cd(figs)

    % Es guarda el resultat amb els formats tif i eps.
    figname = sprintf('%s-%sSpectra',guitar_type,note);
    print(figname,'-dtiff')
    print(figname,'-depsc')
    close all

    % Calculem la mitjana de les repeticions plotejades anteriorment (mean
    spectrum)
    pxx_mean = pxx_mean/n_files;
    figure
    plot(f,pow2db(pxx_mean))
    tit=sprintf('%s %s Mean spectrum',guitar_type,note);
    title(tit)
    % Create ylabel
    ylabel('Power Spectrum (dB)');
    % Create xlabel
    xlabel('Frequency (Hz)');
    xticks(ceil(xt));
    xtickangle(90)

    % Es guarda el resultat amb els formats tif i eps.
    figname = sprintf('%s-%sMeanSpectrum',guitar_type,note);
    print(figname,'-dtiff')
    print(figname,'-depsc')
    close all

    % Calculem la mida de l'espectre per crear la matriu mitjana.
    ps = Spectdata_(1).p;
    nn = size(ps);
    ps_mean = zeros(nn(1),nn(2));

    % Creem una matriu per saber el nombre de divisions que hem de fer a
    l'hora de plotejar els espectres.
    nsubplot = ceil(sqrt(n_files));

    % Valors del nombre d'harmònics desitjat.
    yt = [0:fund_freq:fund_freq*n_harm];
```

```
    figure
    % Per cadascuna de les repeticions es crea un espectrograma amb les
    dades emmagatzemades anteriorment mitjançant surface.
    for i=1:n_files
        % Els valors de freqüència i temps són els mateixos per cada
        repetició.
        F = Spectdata_(i).f;
        T = Spectdata_(i).t;
        % Valor que marca la diferència entre una repetició i l'altra.
        ps = Spectdata_(i).p;
        subplot(nsubplot,nsubplot,i)
        s = surface(T',F,pow2db(ps));
        s.EdgeColor = 'none';
        tit = sprintf('%s %s %i',guitar_type,note,i);
        title(tit)
        yticks(yt);
        % Cada iteració suma el valor per després, quan estiguin totes
        sumades, poder fer la mitjana.
        ps_mean = ps_mean+ps;
    end

    % Es guarda el resultat amb els formats tif i eps.
    figname = sprintf('%s-%sSpectrograms',guitar_type,note);
    print(figname,'-dtiff')
    print(figname,'-depsc')
    close all

    % Es calcula i ploteja la mitjana.
    ps_mean = ps_mean/n_files;
    figure
    s = surface(T',F,pow2db(ps_mean));
    s.EdgeColor = 'none';
    % Create ylabel
    ylabel('Freq (Hz)');
    % Create xlabel
    xlabel('time (s)');
    % Create title
    tit = sprintf('%s %s Mean',guitar_type,note);
    title(tit)

    % Harmònics.
    yt = [0:fund_freq:fund_freq*n_harm];
    yticks(yt);

    % Es guarda el resultat amb els formats tif i eps.
    figname = sprintf('%s-%sMeanSpectrogram',guitar_type,note);
    print(figname,'-dtiff')
    print(figname,'-depsc')

    close all
    cd(oldFolder)

end % Es repeteix el mateix per una altra nota.

end
```

B.2 Codi de la interfície gràfica

B.2.1 Creació de les diferents parts de la interfície gràfica

```
function varargout = tabbedGUI(varargin)
% TABBEDGUI MATLAB code for tabbedGUI.fig
%      TABBEDGUI, by itself, creates a new TABBEDGUI or raises the existing
%      singleton*.
%
%      H = TABBEDGUI returns the handle to a new TABBEDGUI or the handle to
%      the existing singleton*.
%
%      TABBEDGUI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TABBEDGUI.M with the given input
arguments.
%
%      TABBEDGUI('Property','Value',...) creates a new TABBEDGUI or raises
the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before tabbedGUI_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to tabbedGUI_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help tabbedGUI

% Last Modified by GUIDE v2.5 18-Dec-2018 10:13:40

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @tabbedGUI_OpeningFcn, ...
                   'gui_OutputFcn',  @tabbedGUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before tabbedGUI is made visible.
function tabbedGUI_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to tabbedGUI (see VARARGIN)
```

```
% Choose default command line output for tabbedGUI
handles.output = hObject;
set(handles.output,'Units','Pixels','Position',get(0,'ScreenSize'))

% Figure
s = warning('off', 'MATLAB:uitabgroup:OldVersion');
handles.TabGroup = uitabgroup('Parent',handles.output);
warning(s);

%% Tabs definition
handles.Tabs(1) = uitab('Parent',handles.TabGroup, 'Title','Data');
handles.Tabs(2) = uitab('Parent',handles.TabGroup, 'Title','Run');
handles.Tabs(3) = uitab('Parent',handles.TabGroup, 'Title','Plot');
handles.Tabs(4) = uitab('Parent',handles.TabGroup, 'Title','Compare');
set(handles.TabGroup,'SelectedTab',handles.Tabs(1));

%% Tabs
% Tab 1 (data)
% Plot push button
handles.browse = uicontrol('Style','pushbutton','String','BROWSE
FILE','FontWeight','bold','FontSize',12,'Parent',handles.Tabs(1),'Position'
,[20 580 200 30],'Callback',@browse_Callback);
handles.audio_file =
uicontrol('Style','text','Parent',handles.Tabs(1),'Position',[20 540 200
30],'String','*.wav','FontSize',12);
% Plot edit text for guitar name
uicontrol('Style','text','Parent',handles.Tabs(1),'Position',[20 500 200
30],'String','GUITAR NAME','FontWeight','bold','FontSize',12);
handles.guitar_name =
uicontrol('Style','edit','Parent',handles.Tabs(1),'Position',[20 470 200
30],'Callback',@guitar_name_Callback);

% Tab 2 (run)
handles.run =
uicontrol('Style','pushbutton','String','RUN','FontWeight','bold','FontSize
',12,'Parent',handles.Tabs(2),'Position',[20 580 200
30],'Callback',@run_Callback);

% Tab 3 (plot)
% Guitar name
uicontrol('Style','text','Parent',handles.Tabs(3),'Position',[20 580 120
20],'String','GUITAR NAME','FontWeight','bold','FontSize',12);
handles.guitar_names = uicontrol('Style','popupmenu','String',{"PUSH
HERE"},'Parent',handles.Tabs(3),'Position',[20 550 120
20],'Callback',@guitar_names_Callback);
% Plot axes
handles.mean_spectrum_axes = axes('Parent',handles.Tabs(3),'Position',[.2
.1 .35 .5]);
handles.mean_spectrogram_axes =
axes('Parent',handles.Tabs(3),'Position',[.6 .1 .35 .5]);
uicontrol('Style','text','Parent',handles.Tabs(3),'Position',[280 420 400
20],'String','MEAN SPECTRUM','FontWeight','bold','FontSize',12);
uicontrol('Style','text','Parent',handles.Tabs(3),'Position',[780 420 400
20],'String','MEAN SPECTROGRAM','FontWeight','bold','FontSize',12);
% Plot notes
uicontrol('Style','text','Parent',handles.Tabs(3),'Position',[20 480 120
20],'String','SELECT NOTE','FontWeight','bold','FontSize',12);
handles.notes = uicontrol('Style','popupmenu','String',{"PUSH
HERE"},'Parent',handles.Tabs(3),'Position',[20 450 120
20],'Callback',@notes_Callback);
```

```
% Tab 4 (compare)
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[20 580 120
20],'String','GUITAR 1','FontWeight','bold','FontSize',12);
handles.names = uicontrol('Style','popupmenu','String',{"PUSH
HERE"},'Parent',handles.Tabs(4),'Position',[20 550 120
20],'Callback',@names_Callback);
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[200 580 140
20],'String','GUITAR 2','FontWeight','bold','FontSize',12);
handles.names_1 = uicontrol('Style','popupmenu','String',{"PUSH
HERE"},'Parent',handles.Tabs(4),'Position',[200 550 140
20],'Callback',@names_1_Callback);
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[20 480 120
20],'String','SELECT NOTE','FontWeight','bold','FontSize',12);
handles.notes_1 = uicontrol('Style','popupmenu','String',{"PUSH
HERE"},'Parent',handles.Tabs(4),'Position',[20 450 120
20],'Callback',@notes_1_Callback);
% Create radio buttons.
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[400 580 140
20],'String','SELECT GRAPH','FontWeight','bold','FontSize',12);
handles.mean_spectrum_button = uicontrol('Style','checkbox','String','Mean
spectrum','Parent',handles.Tabs(4),'Position',[400 555 120
20],'HandleVisibility','off','Callback',@mean_spectrum_button_Callback);
handles.mean_spectrogram_button =
uicontrol('Style','checkbox','String','Mean
spectrogram','Parent',handles.Tabs(4),'Position',[400 535 120
20],'HandleVisibility','off','Callback',@mean_spectrogram_button_Callback);
% Create axes
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[280 420 400
20],'String','GUITAR 1','FontWeight','bold','FontSize',12);
uicontrol('Style','text','Parent',handles.Tabs(4),'Position',[780 420 400
20],'String','GUITAR 2','FontWeight','bold','FontSize',12);
handles.mean_axes_1 = axes('Parent',handles.Tabs(4),'Position',[.2 .1 .35
.5]);
handles.mean_axes_2 = axes('Parent',handles.Tabs(4),'Position',[.6 .1 .35
.5]);

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes tabbedGUI wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = tabbedGUI_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
```

B.2.2 Pestanya data

```
%% Tab 1 (DATA)
% --- Executes on button press in browse.
function browse_Callback(hObject, eventdata, handles)
% hObject    handle to browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
handles = guidata(hObject);

[filename,filepath] = uigetfile('*.wav', 'Search audio');
fullname = [filepath filename];
set(handles.audio_file,'String',filename);
drawnow;

guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function audio_file_CreateFcn(hObject, eventdata, handles)
% hObject    handle to audio_file (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function guitar_name_Callback(hObject, eventdata, handles)
% hObject    handle to guitar_name (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of guitar_name as text
%        str2double(get(hObject,'String')) returns contents of guitar_name
as a double

handles = guidata(hObject);

guitar = get(hObject,'String'); % Guarda el valor escrit
% guitar = str2double(guitar);
handles.guitar_name = sprintf('guitar %s',guitar); % Guardar a
l'identificador
guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes during object creation, after setting all properties.
function guitar_name_CreateFcn(hObject, eventdata, handles)
% hObject    handle to guitar_name (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

B.2.3 Pestanya run

```
%% Tab 2 (RUN)
% --- Executes on button press in run.
function run_Callback(hObject, eventdata, handles)
% hObject    handle to run (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

handles = guidata(hObject);

f = waitbar(0,'Please wait...');
pause(.5)

waitbar(100,f,'Loading your data');
pause(1)

waitbar(100,f,'Processing your data');
pause(1)

waitbar(100,f,'Finishing');
pause(1)

close(f)

audio_file = get(handles.audio_file, 'String');
guitar_name = handles.guitar_name;
batch_analysis_v4(audio_file,guitar_name);
tabbedGUI
```

B.2.4 Pestanya results

```
%% Tab 3 (RESULTS)
% --- Executes on selection change in guitar_names.
function guitar_names_Callback(hObject, eventdata, handles)
% hObject    handle to guitar_names (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns names contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from names

% signalGenerators=importdata('availableSignalGenerators.txt').';
% handles.popup1.String = signalGenerators; % If using R2014b or later
% % OR
% set(handles.popup1.String, 'String', signalGenerators); % R2014a or
earlier
handles = guidata(hObject);

files = dir('guitar *');
for k = 1 : length(files)
  listboxItems{k} = files(k).name;
end
set(handles.guitar_names, 'String', listboxItems);

allItems = get(handles.guitar_names,'string');
selectedIndex = get(handles.guitar_names,'Value');
selected_guitar = allItems{selectedIndex};
handles.selected_guitar_0 = selected_guitar;

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes on selection change in notes.
function notes_Callback(hObject, eventdata, handles)
% hObject    handle to notes (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns names contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from names

% signalGenerators=importdata('availableSignalGenerators.txt').';
% handles.popup1.String = signalGenerators; % If using R2014b or later
% % OR
% set(handles.popup1.String, 'String', signalGenerators); % R2014a or
earlier
handles = guidata(hObject);

oldFolder = pwd;
cd(sprintf('notes_%s',handles.selected_guitar_0))
load('guitar_notes.mat');
listboxItems = notes;
set(handles.notes, 'String', listboxItems);

cd(oldFolder)

allItems = get(handles.notes,'string');
selectedIndex = get(handles.notes,'Value');
selected_notes = allItems{selectedIndex};
handles.selected_note = selected_notes;

cd(sprintf('figures_%s',handles.selected_guitar_0))
spectrogram = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrogram.tif',handles.selected_guitar_0,handles.selected_note)));
axes(handles.mean_spectrogram_axes);
imshow(spectrogram)
spectrum = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrum.tif',handles.selected_guitar_0,handles.selected_note)));
axes(handles.mean_spectrum_axes);
imshow(spectrum);

cd(oldFolder);

guidata(hObject,handles); % Guardar les dades de l'app
```

B.2.5 Pestanya compare

```
%% Tab 4 (COMPARE)
% --- Executes on selection change in names.
function names_Callback(hObject, eventdata, handles)
% hObject    handle to names (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns names contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from names

% signalGenerators=importdata('availableSignalGenerators.txt').';
% handles.popup1.String = signalGenerators; % If using R2014b or later
% % OR
% set(handles.popup1.String, 'String', signalGenerators); % R2014a or
earlier
handles = guidata(hObject);
```

```
files = dir('guitar *');
for k = 1 : length(files)
  listboxItems{k} = files(k).name;
end
set(handles.names, 'String', listboxItems);

allItems = get(handles.names,'string')
selectedIndex = get(handles.names,'Value')
selected_guitar = allItems{selectedIndex};
handles.selected_guitar = selected_guitar;

% notes de la guitarra
oldFolder = pwd;
cd(sprintf('notes_%s',handles.selected_guitar))
load('guitar_notes.mat');
handles.A = notes;
cd(oldFolder)

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes on selection change in names_1.
function names_1_Callback(hObject, eventdata, handles)
% hObject    handle to names_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns names_1 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from names_1
handles = guidata(hObject);

oldFolder = pwd;
cd(oldFolder)

files = dir('guitar *');
for k = 1 : length(files)
  listboxItems{k} = files(k).name;
end
set(handles.names_1, 'String', listboxItems);

allItems_1 = get(handles.names_1,'string')
selectedIndex_1 = get(handles.names_1,'Value')
selected_guitar_1 = allItems_1{selectedIndex_1};
handles.selected_guitar_1 = selected_guitar_1;

% notes de la guitarra
cd(sprintf('notes_%s',handles.selected_guitar_1))
load('guitar_notes.mat');
handles.B = notes;
cd(oldFolder)

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes on button press in mean_spectrum_button.
function mean_spectrum_button_Callback(hObject, eventdata, handles)
% hObject    handle to mean_spectrum_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles = guidata(hObject);

allItems_1 = get(handles.mean_spectrum_button,'string')
```

```
% selectedIndex = get(handles.mean_spectrum_button,'Value')
% selected_graph = allItems{selectedIndex};
handles.selected_graph = allItems_1;

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes on button press in mean_spectrogram_button.
function mean_spectrogram_button_Callback(hObject, eventdata, handles)
% hObject    handle to mean_spectrogram_button (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles = guidata(hObject);

allItems = get(handles.mean_spectrogram_button,'string')
% selectedIndex = get(handles.mean_spectrogram_button,'Value')
% selected_graph = allItems{selectedIndex};
handles.selected_graph = allItems;

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes when selected object is changed in uibuttongroup2.
function uibuttongroup_SelectionChangedFcn(hObject, eventdata, handles)
% hObject    handle to the selected object in uibuttongroup2
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
handles = guidata(hObject);
get(eventdata.NewValue, 'Tag')

% --- Executes on selection change in select_graph.
function select_graph_Callback(hObject, eventdata, handles)
% hObject    handle to select_graph (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns select_graph
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
select_graph
handles = guidata(hObject);

allItems = get(handles.select_graph,'string')
selectedIndex = get(handles.select_graph,'Value')
selected_graph = allItems{selectedIndex};
handles.selected_graph = selected_graph;

guidata(hObject,handles); % Guardar les dades de l'app

% --- Executes during object creation, after setting all properties.
function select_graph_CreateFcn(hObject, eventdata, handles)
% hObject    handle to select_graph (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in notes_1.
```

```
function notes_1_Callback(hObject, eventdata, handles)
% hObject    handle to notes_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: contents = cellstr(get(hObject,'String')) returns names contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from names
% signalGenerators=importdata('availableSignalGenerators.txt').';
% handles.popup1.String = signalGenerators; % If using R2014b or later
% % OR
% set(handles.popup1.String, 'String', signalGenerators); % R2014a or
earlier
handles = guidata(hObject);

oldFolder = pwd;
A = handles.A;
B = handles.B;

if length(A) > length(B)
    C = ismember(A,B);
    listboxItems = A(C);
else
    C = ismember(B,A);
    listboxItems = B(C);
end

set(handles.notes_1, 'String', listboxItems);

cd(oldFolder)
allItems = get(handles.notes_1,'string')
selectedIndex = get(handles.notes_1,'Value')
selected_notes = allItems{selectedIndex};
handles.selected_note = selected_notes;

cd(sprintf('figures_%s',handles.selected_guitar))
if handles.mean_spectrogram_button.Value == 1
    spectrogram = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrogram.tif',handles.selected_guitar,handles.selected_note)));
    axes(handles.mean_axes_1);
    imshow(spectrogram);
elseif handles.mean_spectrum_button.Value == 1
    spectrum = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrum.tif',handles.selected_guitar,handles.selected_note)));
    axes(handles.mean_axes_1);
    imshow(spectrum);
end
cd(oldFolder)
cd(sprintf('figures_%s',handles.selected_guitar_1))
if handles.mean_spectrogram_button.Value == 1
    spectrogram_1 = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrogram.tif',handles.selected_guitar_1,handles.selected_note)));
    axes(handles.mean_axes_2);
    imshow(spectrogram_1);
elseif handles.mean_spectrum_button.Value == 1
    spectrum_1 = imread(fullfile(pwd,'/',sprintf('%s-
%sMeanSpectrum.tif',handles.selected_guitar_1,handles.selected_note)));
    axes(handles.mean_axes_2);
    imshow(spectrum_1);
end
cd(oldFolder);
guidata(hObject,handles); % Guardar les dades de l'app
```

C ANNEX: DADES TÈCNIQUES DE LES GRAVACIONS USADES

Les gravacions d'àudio utilitzades per dur a terme el treball van ser gravades per Lluís Costa a l'estudi Soundclub Studio amb un micròfon *Schoeps CMC 5* amb la càpsula *MK4*. Els models de guitarra utilitzats van ser M16, Takamine, Flamenco Negre A i Ecoronda.

A la Figura 56 es mostren les dades tècniques de l'amplificador *CMC 5* utilitzat pel micròfon *Schoeps*. I a la Figura 57 el diagrama de blocs del micròfon.

Amplifier type	Powering	Current consumption	Impedance	Low-cut frequency (-3 dB)
CMC 6U / 6Uxt:	12 V phantom	8 mA	25 Ohms	20 Hz
	48 V phantom (automatic switchover)	4 mA	35 Ohms	20 Hz
CMC 5U:	48 V phantom	4 mA	35 Ohms	30 Hz
CMC 3U:	12 V phantom	11 mA	20 Ohms	30 Hz

Polarity: Increasing sound pressure on the microphone's 0° axis produces a positive-going voltage at pin 2.
Maximum output voltage: 1 V (at 1 kHz and 1 kOhm load resistance)
Minimum recommended load resistance: 600 Ohms (A load resistance below this value will particularly reduce the maximum output level.)
The other technical specifications depend on the choice of capsule – see page 12 ff.
Length: 116 mm (incl. 3 mm capsule thread)
Diameter: 20 mm
Weight: 65 – 68 g, depending on type
Surface finish: matte gray (g) or nickel (ni)

Figura 56: Dades tècniques de l'amplificador CMC 5 del micròfon Schoeps (Schoeps Mikrofone, 2015).

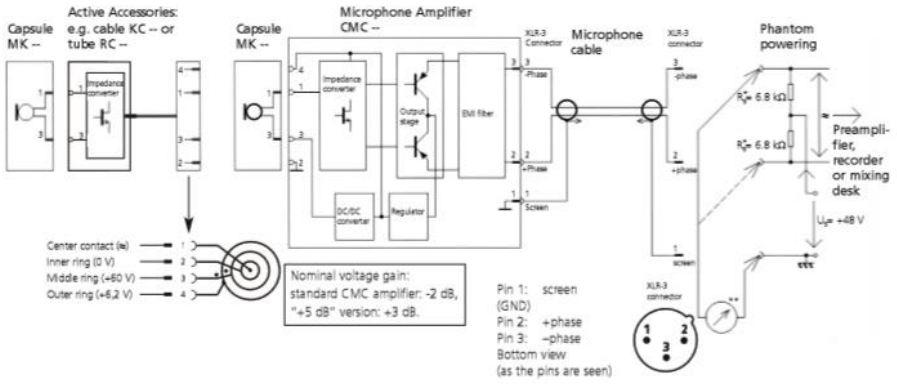

Figura 57: Diagrama de blocs del micròfon Schoeps (Schoeps Mikrofone, 2015).

Finalment, a la Taula 6 es mostren les especificacions dels micròfons complets, és a dir, compostos per l'amplificador *CMC 6* i els diferents tipus de càpsules. S'ha de tenir en compte, però que l'amplificador *CMC 6* té millors prestacions que *CMC 5,* el qual s'ha utilitzat per dur a terme les gravacions.

Taula 6: Especificacions del micròfon compost per l'amplificador CMC 6 i els diferents tipus de càpsules (Schoeps Mikrofone, 2015).

Capsule Type	Polar Pattern	Frequency Range	Sensitivity	Equivalent Noise Level CCIR	Equivalent Noise Level A-weighted	Signal-to-Noise Ratio A-weighted	Max. SPL (0.5% THD)
MK 2	omni	20 Hz – 20 kHz*	15 mV/Pa	23 dB	11 dB	83 dB	130 dB
MK 2H	omni	20 Hz – 20 kHz*	15 mV/Pa	23 dB	12 dB	82 dB	130 dB
MK 2S	omni	20 Hz – 20 kHz*	12 mV/Pa	24 dB	12 dB	82 dB	132 dB
MK 2XS	omni	20 Hz – 20 kHz*	10 mV/Pa	26 dB	14 dB	80 dB	134 dB
BLM 3g	hemisphere	20 Hz – 20 kHz	19 mV/Pa	23 dB	12 dB	82 dB	128 dB
BLM 03 Cg	hemisphere	20 Hz – 20 kHz	19 mV/Pa	23 dB	12 dB	82 dB	128 dB
MK 21	wide cardioid	30 Hz – 20 kHz*	13 mV/Pa	25 dB	15 dB	79 dB	132 dB
MK 21H	wide cardioid	30 Hz – 20 kHz*	10 mV/Pa	26 dB	16 dB	78 dB	134 dB
MK 22	Open Cardioid	40 Hz – 20 kHz*	14mV/Pa	23 dB	14 dB	80 dB	131 dB
MK 4	cardioid	40 Hz – 20 kHz*	13 mV/Pa	24 dB	15 dB	79 dB	132 dB
MK 4V	cardioid	40 Hz – 20 kHz	13 mV/Pa	24 dB	14 dB	80 dB	132 dB
MK 41	supercardioid	40 Hz – 20 kHz*	13 mV/Pa	24 dB	16 dB	78 dB	132 dB
MK 41V	supercardioid	40 Hz – 20 kHz	13 mV/Pa	24 dB	15 dB	79 dB	132 dB
MK 8	figure-8	40 Hz – 16 kHz	10 mV/Pa	26 dB	18 dB	76 dB	134 dB
MK 5	omni	20 Hz – 20 kHz*	11 mV/Pa	26 dB	14 dB	80 dB	133 dB
	cardioid	40 Hz – 20 kHz	13 mV/Pa	25 dB	16 dB	78 dB	132 dB
MK 4P	cardioid	close pickup	13 mV/Pa	24 dB	15 dB	79 dB	132 dB
MK 4VP	cardioid	close pickup	13 mV/Pa	24 dB	15 dB	79 dB	132 dB
MK 4XP	cardioid	close pickup	12 mV/Pa	25 dB	15 dB	79 B	132 dB
MK 4VXP	cardioid	close pickup	10 mV/Pa	25 dB	14 dB	80 dB	134 dB