# Sistema per il riconoscimento emozionale in una frase parlata mediante l'utilizzo di reti neurali come classificatori.

Daniele Giardino

Lorenzo Stilo

Luigi Marino

Università degli studi di Roma – Tor Vergata
Facoltà di Ingegneria Elettronica

Laboratorio di Reti Neurali per il Controllo

Anno Accademico: 2012-2013

# Sommario

# Introduzione

Il progetto svolto ha avuto come obiettivo quello di creare un sistema che, a partire da un file audio contenente una frase parlata, fosse in grado di riconoscere l'emozione con la quale la stessa è stata pronunciata.

Il riconoscimento della frase pronunciata avviene grazie all'utilizzo di una rete neurale.

Il processo avviene mediante l'analisi del segnale audio tramite l'utilizzo di quindici diverse funzioni che hanno lo scopo di caratterizzare lo stesso e ottenere dei parametri che vengono utilizzati dalla rete neurale per discriminare il sentimento con il quale la frase sia stata pronunciata.

Il progetto si è svolto in diverse fasi ed è stato condotto dagli studenti, prima con un approccio puramente teorico e in seguito in una fase più pratica.

In ordine: ci si è approcciati al problema del riconoscimento di un'emozione in una frase parlata facendo riferimento all'articolo accademico *"Survey on speech emotion recognition: Features, classification, schemes, and databases"* nel quale venivano esposte le difficoltà nella classificazione delle emozioni e i possibili approcci alla risoluzione delle stesse. Questo ha consentito di entrare in possesso delle nozioni minime per affrontare il progetto.

Si è potuto quindi procedere all'acquisizione di un database di *parlatori* su cui poter iniziare a lavorare. Questo database è costituito da 588 tracce audio in cui attori recitano delle frasi sempre uguali dando intonazioni diverse, cioè attribuendo ad esse diversi sentimenti. La metà di questi attori è di sesso maschile e l'altra di sesso femminile. Inoltre tutte le emozioni sono state classificate in sette diverse tipologie.

Il passaggio successivo è stato quello di acquisire un codice *MATLAB*, realizzato dalla Prof.ssa. Arianna Mencattini, per l'analisi e la caratterizzazione dei file audio. Questo codice è in grado di analizzare una traccia audio mediante quindici diverse funzioni che lo caratterizzano e generare un vettore di 192 elementi che identificano la traccia audio univocamente.

Il codice è stato migliorato ed ottimizzato per poter ottenere il miglior risultato possibile per il singolo file in analisi. Questo processo è avvenuto procedendo generalmente alla parametrizzazione di quante più variabili possibili per poter regolare in maniera più fine l'analisi del *pitch* del segnale. Inoltre il *pitch* del segnale è stato calcolato in maniera duplice, sia discreta che tempo continua mediante una funzione di *smooth*.

Per implementare la semplicità dell'utilizzo del codice si è quindi realizzata una veste grafica in MATLAB che, mediante la visualizzazione a schermo, consenta all'utente di scegliere sia il segnale da analizzare che le modalità con le quali l'analisi va eseguita. L'utente, utilizzando l'interfaccia, può quindi modificare le soglie di analisi e scegliere in che modo il *pitch* del segnale sia calcolato per la caratterizzazione dello stesso. Una volta ultimata questa fase l'utente può visualizzare e controllare il vettore di caratterizzazione prima che questo venga processato dal classificatore basato su rete neurale.

In seguito è stata implementata una modalità che agisce sulla caratterizzazione dell'intero database di parlatori al fine di creare un training set per il classificatore a rete neurale. Questa funzione cicla automaticamente su tutte le tracce audio del database e le caratterizza utilizzando valori di soglie e funzioni matematiche ottimizzate.

Infine è stato scelto, realizzato e quindi addestrato, un classificatore che fosse in grado di ricevere in ingresso il vettore di caratterizzazione della traccia audio e ottenere in uscita uno dei sette sentimenti come risposta.

Il classificatore sfrutta due diversi vettori di pesi sinaptici a seconda del sesso del parlatore. Questo è stato scelto al fine di realizzare una rete il più possibile fedele agli ingressi.

## Articolo sul riconoscimento emozionale

Al fine di approcciarci al problema del riconoscimento di un'emozione in una frase parlata si è cercato di raccogliere informazioni anche da articoli e pubblicazioni scientifiche.

In particolare ci è stato proposto *"Survey on speech emotion recognition: Features, classification, schemes, and databases"* di Moataz El Ayadi, Mohamed S. Kamel, Fakhri Karray.

Nell'articolo si parla della relazione tra la macchina e l'uomo e di un efficiente metodo di interattività comunicativa.

Dalla fine degli anni Cinquanta, non c'è stata una specifica attività di ricognizione dell'attività comunicativa all'interno del contesto mondiale.

In realtà, il processo comunicativo tra il dialogo umano e la macchina è risultato necessario in quanto quest'ultima non segue il processo emozionale dell'uomo.

Ultimamente è stata posta sempre maggiore attenzione allo studio del contenuto emozionale delle parole e, allo stesso tempo, all'identificazione e catalogazione di esso.

Questo articolo vuole pertanto studiare importanti questioni riguardo allo studio del riconoscimento emozionale.

La prima sfida cui ci si trova di fronte è quella di trovare un modo per rappresentare le emozioni; una seconda sfida è trovarne un metodo di classificazione; la terza sfida è quella di creare un vero database per misurare la capacità di un sistema elettronico di valutare le emozioni.

Uno dei metodi più veloci per comunicare tra gli esseri umani è sicuramente quello attraverso i discorsi o le parole. Questo è il motivo che ha indotto i ricercatori a trovare un metodo efficiente per far parlare gli umani con i computer. Tutto ciò naturalmente richiede che i computer abbiano l'intelligenza necessaria a riconoscere la voce umana. Fin dagli anni '50, le ricerche sono continuate per cercare di capire come convertire al meglio i discorsi degli esseri umani in sequenze di parole comprensibili dal computer. Nonostante tutto, permangono delle difficoltà, di cui la principale è quella di tradurre in un linguaggio comprensibile al computer lo stato emozionale dell'uomo. Se si riuscisse a realizzare tutto questo le applicazioni sarebbero infinite.

È molto difficile il riconoscimento delle emozioni o degli strati emozionali collegati ad un discorso. Molte di queste differenze potrebbero dipendere dall'ambiente culturale, dalle tradizioni nonché dai diversi stili di colui che parla.

Lo stesso concetto di emozione è oggetto di disputa tra gli stessi ricercatori. È comunemente riconosciuto che le emozioni possono essere divise in due categorie: attivazione e valenza. Con l'attivazione intendiamo l'ammontare di energia necessaria per esprimere un'emozione. L'espressione di un'emozione di gioia o di rabbia provoca un aumento del battito cardiaco, un aumento della pressione sanguigna, una variazione della profondità del respiro, un tremore muscolare. Di conseguenza parliamo con voce più alta e più velocemente e la nostra voce diventa più acuta. Purtroppo risulta difficile classificare l'emozione espressa dal linguaggio riferendosi solamente all'attivazione. Ad esempio, alziamo la voce sia quando comunichiamo un evento piacevole che quando comunichiamo un evento triste. Diventa quindi fondamentale scoprire come attribuire significato (valenza) all'attivazione. Su questo gli scienziati sono in profondo disaccordo.

Williams and Stevens ha condotto diversi studi psicologici sul meccanismo delle emozioni osservando che gli stessi generano manifestazioni fisiche, come ad esempio l'aumento della frequenza cardiaca e di quella respiratoria.

Diverse emozioni, secondo lo studioso, portano all'attivazione di meccanismi psicologici diversi; es.: se ci altera, si tende ad alzare la voce. Queste caratteristiche hanno una valenza emozionale diversa.

Sono state pertanto elaborate delle classificazioni emozionali che sono riconducibili a diversi studiosi (tra i quali Schubiger e O'Connorand Arnol) e che contengono più di 300 stati emozionali.

Certamente, operare una classificazione emozionale, non è un compito semplice.

Primarie emozioni sono: la rabbia, il disgusto, la paura, la gioia, la tristezza, e il sorriso.

Si parla in tal senso dell'archetipo emozionale.

Nell'articolo tre sono i punti importanti delle emozioni:

1) Importanti criteri di identificazione delle emozioni corporali
2) L'impatto delle parole nella classificazione emozionale (in italiano si parla anche di programmazione neurolinguistica per l'identificazione emozionale)
3) Classificazione dei sistemi di ricognizione delle emozioni

Una classificazione accurata si può identificare nel *Berlin corpus*. Questo ha anche operato una classificazione delle emozioni in un data base.

Inoltre le emozioni possono essere percepite attraverso l'acustica e il riconoscimento vocale determinato da un data base.

I data base sviluppati per il riconoscimento e la classificazione delle emozioni da parte della comunità scientifica non sono di pubblico uso.

Questi si basano sul riconoscimento vocale delle emozioni che devono convogliare all'interno della classificazione scelta dal relativo programmatore o dal progettista del data base.

In pratica si tratta di un'interazione tra l'uomo e la macchina attraverso però l'utilizzo del riconoscimento vocale delle relative emozioni.

Il data base può presentare determinati problemi: il riconoscimento delle emozioni simulate non è del tutto affidabile, la qualità della registrazione vocale non è buona e non consente sempre la identificazione emozionale, la fonetica trascrizione delle parole non è sufficiente per l'identificazione emozionale.

I paragrafi 3 e successivi parlano del riconoscimento vocale emozionale e approfondiscono i diversi programmi di riconoscimento, le differenziazioni del data base, il globale riconoscimento vocale e l'oggettiva associazione emozionale.

Un importante uso in questo progetto delle emozioni parlate consente un sistema di estrazione per un'efficiente organizzazione delle differenti emozioni attraverso l'uso di sistemi informatici.

Dalla ricognizione delle emozioni è possibile operare una selezione e classificazione delle performance dei programmi di riconoscimento vocale.

In alcuni casi i programmi sono in grado di operare un'analisi per estrazione mentre in altri casi riescono anche ad effettuare un'analisi statistica dei dati.

L'analisi vocale è utilizzata in diverse attività come ad esempio l'attività forense. Lo stress vocale registrato da un database consente infatti di capire se il soggetto interrogato sta mentendo oppure no. Le emozioni

registrate dal data base possono essere diverse, come ad esempio lo stress psicofisico che può essere percepito sulla base di una variazione acustica della voce dell'interlocutore, poiché, nelle attività mentali in cui è necessario un'attività mentale o psicomotoria, la modulazione della frequenza diminuisce e i muscoli si contraggono.

Sulla base di questo esempio, l'articolo in esame considera nei par. 3 e successivi i seguenti argomenti: la comparazione tra gli attuali lineamenti e quelli globali, la descrizione dei differenti tipi di modi di parlare e le emozioni (nonché il loro riconoscimento), esempi del processo descritto, discussioni sulle altre informazioni integrative che sono estraibili dall'acustica e ovviamente classificabili.

La classificazione in un data base delle emozioni comuni ha dato vita ad una identificazione del data base sulla base del Corpo emozionale, ad una verifica dell'accesso di questi programmi al pubblico (alcuni gratuiti, altri commerciali, altri con vere e proprie licenze d'uso), ad una loro differenziazione in base alla lingua di riconoscimento vocale (inglese, tedesco, francese ecc.), alla loro numerazione e nomenclatura in funzione delle emozioni riconosciute, alla tipologia di impiego degli stessi data base (Professionali, call center, madre e padre ecc.), all'identificazione delle emozioni (panico, disperazione, interesse, ecc..).

Le caratteristiche dell'espressione fonetica del parlato sono pertanto identificate nell'articolo in

1) Espressioni funzionali correlate;
2) Caratteristiche formanti la stessa espressione
3) Funzioni che caratterizzano l'energia impiegata;
4) Funzioni di temporizzazione;
5) Caratteristiche connesse all'articolazione delle parole

In ogni modo gli autori di diversi studi, giungono anche a soluzione diverse, in merito all'interpretazione della voce: per esempio, mentre Murray e Arnott indicano che la velocità di un eloquio e il timbro alto sono associati ad un'emozione di rabbia, Oster e Risberg concludono invece che possano essere associati anche alla paura e alla gioia.

La tonalità di voce pertanto, se pur appare importante per l'emozione, non prescinde dalle espressioni correlate, della foga dello stesso parlare, dal tempo e dall'articolazione delle parole.

Si può concludere affermando che dalle combinazioni acustiche e dalle informazioni linguistiche sono deducibili, attraverso l'uso dei data base indicati (di diversa origine e produzione) creati per diversi impieghi, complesse informazioni emozionali. In materia, per consentire di elaborare le informazioni ad opera di un software, sono state elaborare delle vere e proprie formule per calcolare il livello dell'alterazione delle tonalità vocali a cui associare le relative emozioni.

Non da ultimo e non meno importante è anche l'informazione dedotta dal discorso e dal contributo che la stessa semantica può dare alla progettazione di tali data base.

La difficoltà maggiore, per il software, è quella di registrare e comprendere il contesto nel quale il discorso viene pronunciato. Per tale motivo, alcuni sistemi utilizzano alcune locuzioni o espressioni, come indicatori del contesto nel quale il discorso viene pronunciato, associando, all'operato del parlato, anche le informazioni di contesto.

L'ultima parte (dal par. 4 in poi) dell'articolo tratta, anche attraverso l'espressione di un'analisi matematica, il sistema con il quale si dovrebbe progettare un software e un data base per il riconoscimento vocale emozionale. Questo deve comunque essere progettato in due parti: il primo deve comprendere una unità di elaborazione che consente di estrarre le caratteristiche appropriate dai dati vocali disponibili (questi possono essere direttamente rilevati dalla macchina o eventualmente riversati attraverso apposite

registrazioni che però devono essere di alta definizione), mentre la seconda parte deve essere costituita da un classificatore che decide la base emozione del discorso enunciato.

## Il Database EMOVO

EMOVO è un database vocale italiano che contiene frasi suddivise in stati emozionali. Consiste in 588 espressioni vocali registrate da sei diversi attori professionisti, tre uomini e tre donne.

Ogni parlatore legge quattordici frasi in lingua Italiana (due delle quali non hanno un senso compiuto) nelle quali si percepiscono sette diverse emozioni:

- Disgusto
- Gioia
- Paura
- Rabbia
- Sorpresa
- Tristezza
- Neutralità

Queste frasi sono state registrare in un ambiente silenzioso, attraverso due microfoni e un registratore digitale con frequenza di campionamento di 48KHz, ed un'ampiezza di 16 bit per ogni campione.

## L'analisi della traccia audio

Per poter analizzare un tratto di parlato e quindi caratterizzarlo in vista di una sua classificazione è stato necessario processarlo in diversi modi, ognuno dei quali genera un risultato diverso sia nella logica che nella forma.

Per far ciò ci siamo basati sul linguaggio e ambiente di calcolo numerico *MATLAB*, uno strumento potentissimo e incredibilmente versatile per applicazioni matematiche e ingegneristiche.

In questo nostro lavoro si è sfruttata la capacità di questo linguaggio nell'elaborazioni di segnali.

Il codice che è stato realizzato con il supporto e la collaborazione della Prof.ssa Arianna Mencattini è strutturato in due macro blocchi: l'acquisizione della traccia audio e la caratterizzazione della stessa.

Il primo blocco è strutturato in modo tale che, a partire da un file audio non compresso (codificato utilizzando il protocollo Microsoft lossless "WAV"), l'ambiente ottiene la frequenza con cui la traccia audio è stata campionata e un vettore riga che conterrà gli elementi del campionamento stesso.

Questi due elementi sono gli argomenti principali di tutte le funzioni di analisi e caratterizzazione che costituiscono il secondo blocco del codice.

La restante parte del primo blocco si occupa di acquisire parametri per il miglioramento delle analisi del secondo blocco e di "preparare" la traccia audio per l'analisi. Tra le operazioni in questione la più significativa è lo scarto di tutti i canali audio aggiuntivi per ottenere quindi una traccia audio mono.

Il secondo blocco costituisce il vero cuore del codice e si compone di uno script e undici funzioni per un totale di quindici diverse analisi a cui il file audio è sottoposto.

Quando il processo è ultimato, i risultati dei quindici processi vengono giustapposti per formare un unico vettore di caratterizzazione della traccia audio. Questo ha la forma di una matrice riga con 192 elementi.

## Il vettore di caratterizzazione

Veniamo ora ad analizzare ogni singola funzione, il tipo di analisi e i risultati che fornisce.

### Operatore di *Teager*

In questa prima operazione il segnale viene processato utilizzando l'operatore di energia di *Teager*. Per completezza l'operatore è così definito per un caso continuo:

$$\psi\big(x(t)\big) = \dot{x}^2(t) - x(t)\ddot{x}(t)$$

Dove $\dot{x}$ e $\ddot{x}$ indicano rispettivamente la derivata prima e la derivata seconda.

Il risultato dell'operatore è quindi analizzato per ottenere sei diverse *features*:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

I sei risultati costituiranno il primo blocco del vettore di caratterizzazione.

### Energia del segnale

In questa operazione viene calcolata l'energia del segnale. Anche in questo caso per dare un peso vengono passati in uscita sei diverse *features* che sono:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questi sei valori numerici costituiranno il secondo blocco del vettore di caratterizzazione.

### Analisi con trasformate *wavelet* del *pitch* del segnale per le basse frequenze

In questa operazione viene in un primo stadio calcolato il *pitch* del segnale quindi processato con una trasformata *wavelet*. Questa operazione viene però limitata al segmento a bassa frequenza dello spettro del segnale.

Anche in questo caso sono calcolate le seguenti *features*:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questi sei valori costituiscono il terzo blocco del vettore di caratterizzazione.

### Analisi con trasformate *wavelet* del *pitch* del segnale per le alte frequenze

Come per l'operazione precedente, dopo aver calcolato il *pitch* del segnale, questo viene passato ad una trasformata *wavelet ,* ma in questo caso limitandosi ad un'analisi per le alte frequenze.

Le seguenti *features* sono calcolate e giustapposte al vettore di caratterizzazione:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questi sei valori costituiranno il quarto blocco.

### *Pitch* del segnale

In questo caso verrà calcolato il *pitch* del segnale in ingresso e caratterizzato dalle sei *features* che abbiamo visto precedentemente:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

L'analisi del *pitch* costituisce il quinto blocco.

### Coefficienti di variazione del *pitch*

Il *pitch* del segnale viene passato a valori discreti a questa funzione che ne calcola il coefficiente di variazione del *pitch*.

Anche in questo caso le *features* scelte sono:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

I valori d'uscita costituiscono il sesto blocco del vettore di caratterizzazione.

### Gaussian Mixture Model

In questa analisi la distribuzione viene processata utilizzando il *Gaussian Mixture Model* che si occupa di approssimare la stessa come combinazione lineare di diverse gaussiane.

Per ottenere delle *features* valide da questo tipo di analisi sono state scelte i seguenti risultati:

- Media della distribuzione
- Deviazione standard
- Coefficienti delle combinazioni lineari

Trattandosi di un modello realizzato con gaussiane, è possibile scegliere diverse forme di partenza. Nel nostro caso sono state scelte quattro diverse forme d'onda. Per ognuna di queste la distribuzione viene approssimata a combinazione lineare e analizzata per ottenere le *features* sopra elencate.

I valori in uscita costituiscono il settimo blocco del vettore di caratterizzazione.

### Rapporto silenzio-parlato

Questa analisi si occuperà del calcolo del rapporto tra silenzio nel file audio che avrà in ingresso e i tratti in cui invece è rivelata la presenza di parlato.

Questo è riassunto in un unico valore numerico che secondo la letterature consente di ottenere informazioni molto importanti per la caratterizzazione del sentimento.

Il valore costituisce l'ottavo blocco del vettore di caratterizzazione.

### Retta di regressione

Tornando all'analisi del *pitch* del segnale, questa analisi si occupa di calcolare la retta di regressione della distribuzione. L'unico parametro di questa analisi è costituito dalla pendenza della retta di regressione stessa.

Questo parametro costituisce il nono blocco del vettore di caratterizzazione

### Ampiezza massima della distribuzione

Questa analisi si limita a calcolare il massimo *range* della distribuzione, operando una differenza tra il massimo valore della stessa ed il minimo.

Questo blocco, che è costituito da un unico valore numerico, è il decimo del vettore di caratterizzazione.

### Analisi del polinomio di *Legendre*

Dalla distribuzione in ingresso vengono calcolati i polinomi di *Legendre*. Di questi, si procede all'analisi di quello del primo ordine.

Le *features* ottenute dall'analisi sono le seguenti:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questo blocco è l'undicesimo del vettore di caratterizzazione.

### Analisi del tratto vocale significativo

Dell'intero segnale audio, viene estrapolato il tratto vocale più significativo e si procede quindi all'analisi della durata di quest'ultimo.

Le *features* scelte sono le seguenti e costituiscono il dodicesimo blocco del vettore di classificazione:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

### Rapporto tra le formanti

Ottenute le formanti della traccia audio si procede ad analizzare il rapporto tra la prima e la seconda. Anche in questo caso è necessario scegliere quali analisi applicare alla distribuzione ottenuta.

La scelta è ricaduta sulle seguenti *features*:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questo è il tredicesimo blocco del vettore di caratterizzazione per il file audio in esame.

### Analisi in frequenza delle formanti

Sempre utilizzando le formanti della traccia audio in analisi, si procede in questa funzione all'analisi delle stesse in frequenza. Le *features* scelte per la loro caratterizzazione sono quelle standard per le analisi precedentemente svolte sulle distribuzioni:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

Questi sei valori costituiscono il quattordicesimo blocco per vettore di caratterizzazione.

### Analisi nel tempo delle formanti

Come ultimo procedimento si è scelto di analizzare tutte le formanti basandosi sull'ampiezza delle righe spettrali nel tempo.

Le *features* calcolate sono ancora una volta costituite da:

- Media della distribuzione
- Deviazione standard
- Indice statistico di asimmetria per la distribuzione
- Indice del picco della distribuzione
- Energia della distribuzione
- Entropia della distribuzione

I valori ottenuti costituiscono il quindicesimo è ultimo blocco del vettore di caratterizzazione.

## Il *pitch* del segnale

Uno dei procedimenti che più volte viene eseguito dal codice per la caratterizzazione della traccia audio è il calcolo del *pitch* del segnale.

Questo procedimento dal punto di vista teorico avrebbe senso solo sulle vocali in una frase pronunciata. Questo fa del *pitch* un tipo di forma che non solo è discontinua, ma per lo più discreta per il metodo con cui è calcolata.

Come però abbiamo visto il *pitch* risulta essere fondamentale per la stragrande maggioranza delle funzioni eseguite, per questo molte hanno bisogno di operare su un segnale tempo continuo. Per sopperire a questo problema si è scelto di utilizzare una funzione di *smooth* che sia in grado approssimare il comportamento della distribuzione.

### Interpolation Factor

Come è facile comprendere, una semplice interpolazione dei valori discreti è stata esclusa, in quanto fattori spuri potevano alterare il rendimento generale. Parametrizzando quello che è stato chiamato *Interpolation Factor* è stato possibile regolare quanto la funzione di *smooth* approssimasse un'interpolazione o quanto si avvicinasse ad una semplice retta di regressione.

Questo fattore ha un valore massimo di 1 e un valore minimo di 0;

### Energy Threshold

Trattandosi di una distribuzione di valori discreti, la prima cosa che è facile immaginare è che nel calcolo del *pitch* è possibile incontrare valori incredibilmente distanti dall'andamento generale. Per ovviare a questo problema è stata introdotta una soglia sull'energia. Questa viene passata in percentuale. Consente quindi di definire un parametro che porta all'esclusione di alcuni punti nella distribuzione.

Questo è un fattore espresso in percentuale ed ha quindi un valore massimo di 1 e un valore minimo di 0.01;

### Smooth Factor

Infine è stato introdotto un fattore che regola di quanto la funzione di approssimazione tenti di addolcire la distribuzione discreta dei valori.

Questo fattore ha un valore che oscilla in un *range* tra 1 e 0.

Occorre sottolineare che, per il calcolo di tutte quelle *features* che non richiedono una funzione tempo continua i calcoli sono eseguiti sulla distribuzione discreta di valori. Quindi nessuna informazione del *pitch* originale viene persa o scartata.

### Metodi di calcolo del *pitch*

Secondo la letteratura si può procedere in diversi modi per ottenere il *pitch* di un segnale: questi differiscono sia nel calcolo pratico che nei risultati ottenuti. Quindi ci si trova nella situazione per cui applicando metodi diversi a file uguali i risultati ottenuti sono differenti.

I possibili metodi utilizzati in questo codice sono:

- Autocorrelazione
- *Cepstrum*

Dato che, da esperienze empiriche non è emerso un metodo migliore degli altri, in un primo momento si è scelto di calcolare il *pitch* utilizzando una media tra i due risultati. Ma anche in questo caso per alcune tracce la distribuzione calcolata con altri software sembrava discostare da quella ottenuta.

Si è invece riscontrato che, per segnali diversi, metodi di calcolo differente portavano a risultati a volte ottimi. Per ottimizzare il processo si è quindi scelto di lasciare all'utente la possibilità di confrontare i tre diversi *pitch* calcolati con autocorrelazione, *Cepstrum* e media tra i due, e scegliere il migliore per il singolo file audio in esame.

## Interfaccia grafica

Al fine di rendere di più facile utilizzo il codice di analisi della traccia audio si è deciso di procedere alla creazione di una veste grafica per poter consentire all'utente di rapportarsi al problema senza la necessità di competenze specifiche con il linguaggio *MATLAB*.

Quello che si è voluto realizzare è uno strumento che consenta di ottimizzare il più possibile la fase di caratterizzazione della traccia audio e al contempo faciliti il procedimento di importazione della traccia.

Si è scelto di utilizzare il già più volte menzionato ambiente di *Mathworks*. Le motivazioni principali di questa scelta sono state due: per prima cosa era assolutamente necessario che l'integrazione tra il processo di analisi e l'interfaccia grafica per l'utente fosse totale sia per problemi di prestazioni che per puri problemi pratici di difficoltà nel *porting*. In secondo luogo è stato considerato ai fini dell'apprendimento molto più utile continuare a migliorare la conoscenza di uno strumento tanto esteso e versatile.

*MATLAB* vanta di possedere un editor completo di GUI che si occupa sia della stesura del codice operativo che del design dell'interfaccia. Questo pacchetto, chiamato *GUIDE*, è stato quello utilizzato per la creazione del nostro strumento.

Una GUI realizzata in *MATLAB* è costituita da due diversi file, che hanno il compito di definire l'uno gli oggetti e la disposizione degli stessi all'intero della GUI; l'altro il comportamento e le funzioni di *callback* che questi eseguiranno una volta invocati.

### Design dell'interfaccia

Gli approcci per realizzare una GUI in MATLAB sono molteplici, in quanto si può scegliere se scrivere il codice per entrambi i file o fare uso di un tool dedicato per sopperire alla parte di design dell'interfaccia. Per una maggiore versatilità si è scelto di adoperare quest'ultimo approccio.

Il *tool* dedicato a questo genere di processi consente di progettare il design dell'interfaccia in maniera grafica. Si è scelto di strutturare il programma in diversi blocchi logici che rappresentavano i processi dell'analisi e caratterizzazione della traccia audio.

Un primo blocco si occupa dell'acquisizione del file *WAV lossless* e della preparazione del file per l'analisi, inoltre questo blocco si occuperà anche di raccogliere tutte le informazioni aggiuntive per la caratterizzazione della traccia, quali ad esempio il sesso del parlatore.

Un secondo si occupa invece di consentire la riproduzione e il controllo della traccia che si sta procedendo ad analizzare.

Un terzo blocco è dedicato alla scelta della tipologia di processamento della traccia audio, questo consente di modificare variabili e soglie per un ottimizzazione della caratterizzazione.

Il quarto blocco è un blocco dedicato alla visualizzazione dei risultati e consente all'utente di visualizzare il vettore di caratterizzazione generato

Infine un quinto ed ultimo blocco si occupa della gestione del programma, della visualizzazione dello stato di esecuzione e guida l'utente verso i passi necessari per adoperare lo stesso.

## Comportamento dell'interfaccia

Per definire invece il modo con cui i vari oggetti si comportano e sono interconnessi tra loro è necessario un codice che descriva quelle che sono definite funzioni di *callback*.

Questo codice è realizzato in linguaggio MATLAB, con però alcune differenze rispetto ad una programmazione standard di uno script.

E' stato infatti necessario prendere domestichezza con oggetti di tipo *handles* e *hObject*. In quanto il codice che abbiamo sintetizzato dovrà lavorare in connessione al file precedentemente generato per la descrizione del design.

La struttura del sorgente è composta da due funzioni che descrivono il comportamento della GUI in apertura e chiusura e n funzioni di *callback* per definire il comportamento di ciascun oggetto.

Veniamo ora alla descrizione degli oggetti *handles* e *hObject*: è bene comprendere che questi due tipi di oggetto hanno significati e comportamenti differenti a seconda del contesto in cui si vengono a trovare: gli oggetti di tipo *hObject* fanno riferimento alla funzione nella quale sono invocati, gli oggetti *handles* invece fanno riferimento al contempo a tutte le altre funzioni dell'interfaccia e a tutte le variabili globali della GUI.

Ogni variabile che non è dichiarata in modo globale, cesserà di esistere alla chiusura della funzione che la contiene, è stato quindi necessario esportare alcuni oggetti affinché potessero essere processati da differenti funzioni in più parti dell'interfaccia.

Inoltre occorre sottolineare, che anche quando un oggetto è dichiarato in modo globale questo sarà inaccessibile fintanto che non viene lanciato un comando che si occupa di aggiornare tutte le variabili e gli oggetti di questo tipo.

Veniamo ora all'analisi del codice nelle sue parti principali.

### OpeningFnc

```
function untitled_OpeningFcn(hObject, eventdata, handles, varargin)
handles.count=0;
handles.sigma=str2double(get(handles.smooth,'String'));
handles.ener=str2double(get(handles.energy,'String'));
```

```
handles.intfa=str2double(get(handles.interpolation,'String'));
handles.output = hObject;

guidata(hObject, handles);
```

Questa funzione viene eseguita prima che l'utente possa visualizzare l'interfaccia stessa.

Quello che viene eseguito è sostanzialmente necessario per inizializzare gli oggetti che poi verranno utilizzati nel seguito della GUI.

Per ottimizzare la creazione di una matrice di risultati è stato implementato un contatore che è risultato necessario per indicizzare gli elementi nella matrice stessa e consentire di giustapporre i risultati.

Le tre seguenti inizializzazioni sono necessarie per tre parametri successivamente utilizzati dalla GUI per scegliere il metodo di analisi della traccia.

Infine il comando "guidata" si occupa di aggiornare tutti gli oggetti globali.

### Acquire

```
function acquire_Callback(hObject, eventdata, handles)
set(handles.txt_directory,'String',uigetfile('*.wav','Select a file'));
[y, fs] = wavread(get(handles.txt_directory,'String'));
y=y(:,1);
handles.y=y;
handles.fs=fs;

if (get(handles.male,'Value'))
     sex='M';
elseif (get(handles.female,'Value'))
     sex='F';
end
handles.sex=sex;
handles.player=audioplayer(handles.y,handles.fs);
set(handles.gender,'String',handles.sex);
set(handles.status,'String','Sound file acquired, set the procedure');

guidata(hObject, handles);
```

*Acquire* è un oggetto di tipo *button*, il cui comportamento è definito dalla funzione sopra annessa. Questa funzione ricopre un'importanza fondamentale nel comportamento dell'interfaccia e si occupa di tutta la fase di acquisizione e processamento del file audio per prepararlo all'analisi.

Alla pressione avvia una finestra di *browsing* in cui è possibile selezionare il file audio che si vuole esaminare. Nel comando è già specificata la tipologia di file che si vuole acquisire.

Il file viene importato come un vettore in *MATLAB* e tagliato per assicurarsi di operare con una traccia mono.

Viene quindi di seguito letto il valore da due diversi oggetti dell'interfaccia, due controllori booleani che restituiscono il sesso del parlatore. Queste informazioni sono salvate in una nuova variabile globale per un successivo utilizzo da parte del classificatore.

Di seguito la traccia audio e la sua frequenza di campionamento vengono passate ad un oggetto di tipo *auidioplayer* che si occuperà di costituire un blocco assestante dell'interfaccia per la riproduzione e il controllo della traccia audio.

I successivi comandi servono per visualizzare a schermo negli oggetti appositamente creati: il nome del file, il sesso del parlatore acquisito e aggiornare un oggetto fondamentale per l'interfaccia che è lo stato in cui il la macchina a stati permane.

### Sound Player

Questo blocco di pulsanti richiama funzioni per la riproduzione e il controllo della traccia audio. A differenza di semplici funzioni di riproduzione ha la peculiarità di lavorare su un unico oggetto di tipo *audioplayer*. Questo consente, insieme alle funzioni a lui associate di riprodurre, mettere in pausa e riprendere la riproduzione della traccia salvando lo stato.

Il funzionamento avviene tramite largo utilizzo di variabili globali e quattro diversi pulsanti per controllare la riproduzione della traccia.

### Procedure Selector

Il blocco in esame è forse il più complesso dell'intera interfaccia ed è sicuramente quello che rappresenta il vero cuore del programma che è stato realizzato.

Consente infatti di scegliere in primis la metodologia per il calcolo del *pitch* del segnale. Per poter eseguire questo tipo di operazione è stato necessario implementare il calcolo parziale dello stesso. In questo modo all'utente è possibile confrontare i tre diversi risultati e scegliere quello che meglio approssima il comportamento desiderato.

Oltre a questo è possibile specificare tre diversi parametri che influenzano profondamente il calcolo del *pitch* e l'approssimazione tempo continua dello stesso. Tre variabili double sono inizializzate su valori standard sperimentali e modificabili dall'utente per questo motivo.

Ogni volta che l'utente chiede di visualizzare una tipologia d'analisi quello che viene eseguito è una funzione di questo genere:

```
function p_cor_plot_Callback(hObject, eventdata, handles)
handles.sigma=str2double(get(handles.smooth,'String'));
handles.ener=str2double(get(handles.energy,'String'));
handles.intfa=str2double(get(handles.interpolation,'String'));
[f0_corr,f0_ceps,tw,t_init,form,bwf,en,val,val2,x]=my_speech_proc(handles.
y,handles.fs,handles.ener);
sel=not(val);
[f01_corr,te11,f02_corr,te21,sel21]=smooth_pitch(tw,f0_corr,sel,handles.si
gma,handles.intfa);
figure('Name','Correlation','NumberTitle','off'),plot(te11,f01_corr,'k-');
hold on; plot(te21,f02_corr,'ro');
xlabel('Time(s)'),ylabel('Frequency(Hz)');
```

In questo caso particolare si sta eseguendo il calcolo del *pitch* del segnale mediante un'autocorrelazione. Veniamo ad analizzare come il programma si comporta: per prima cosa acquisisce dagli input i tre parametri necessari per il processamento. E si occupa che siano passati alle funzioni principali nel giusto tipo.

A questo punto viene eseguita parte del codice di analisi della traccia audio fino a che non è disponibile il *pitch* del segnale stesso e la funzione tempo continua che lo approssima.

Ottenuti questi due oggetti questi sono pronti ad essere graficati e si è scelto di farlo sulla stessa figura, in modo che l'utente potesse immediatamente visualizzare sia la distribuzione discreta del *pitch* che la funzione di *smooth* che sta tentando di approssimarla.

Una volta scelta la procedura migliore per la caratterizzazione il pulsante set si occupa di salvare e aggiornare tutte le variabili globali usate dal blocco e di spostare la macchina verso lo stato successivo aggiornando quindi la stringa di status.

### Output_table

Questo oggetto si popola dinamicamente ed è il contenitore dei vettori di caratterizzazione generati. Alla chiusura della funzione viene esportata in un file esterno che contiene tutti i risultati calcolati nella sessione di esecuzione appena terminata.

### Control

La sessione di controllo si occupa di visualizzare a schermo lo stato in cui la macchina si trova e visualizzare eventuali avvisi ed errori durante l'esecuzione.

L'unico pulsante di questa sessione è quello che avvia la caratterizzazione della traccia audio. Come precedentemente spiegato esso necessita che la procedura per la caratterizzazione sia stata precedentemente scelta, a tale fine l'utente è guidato dalla stringa di stato della macchina.

Alla pressione quello che avviene è enunciato dalla sua *callback*:

```
function start_Callback(hObject, eventdata, handles)
handles.count=handles.count+1;

set(handles.status,'String','Error in main function!')
handles.a((handles.count),:)=feat_sound(handles.y,handles.fs,handles.sel,h
andles.sigma,handles.ener,handles.intfa);

set(handles.status,'String','Process completed!');
set(handles.output_table,'Data',handles.a);


guidata(hObject, handles);
```

Come prima cosa viene incrementato il contatore per l'indicizzazione della matrice d'uscita. Di seguito viene chiamata la funzione di caratterizzazione della traccia audio, alla quale sono passati i parametri per la selezione della procedura prescelta.

# Guida all’utilizzo dell’interfaccia grafica

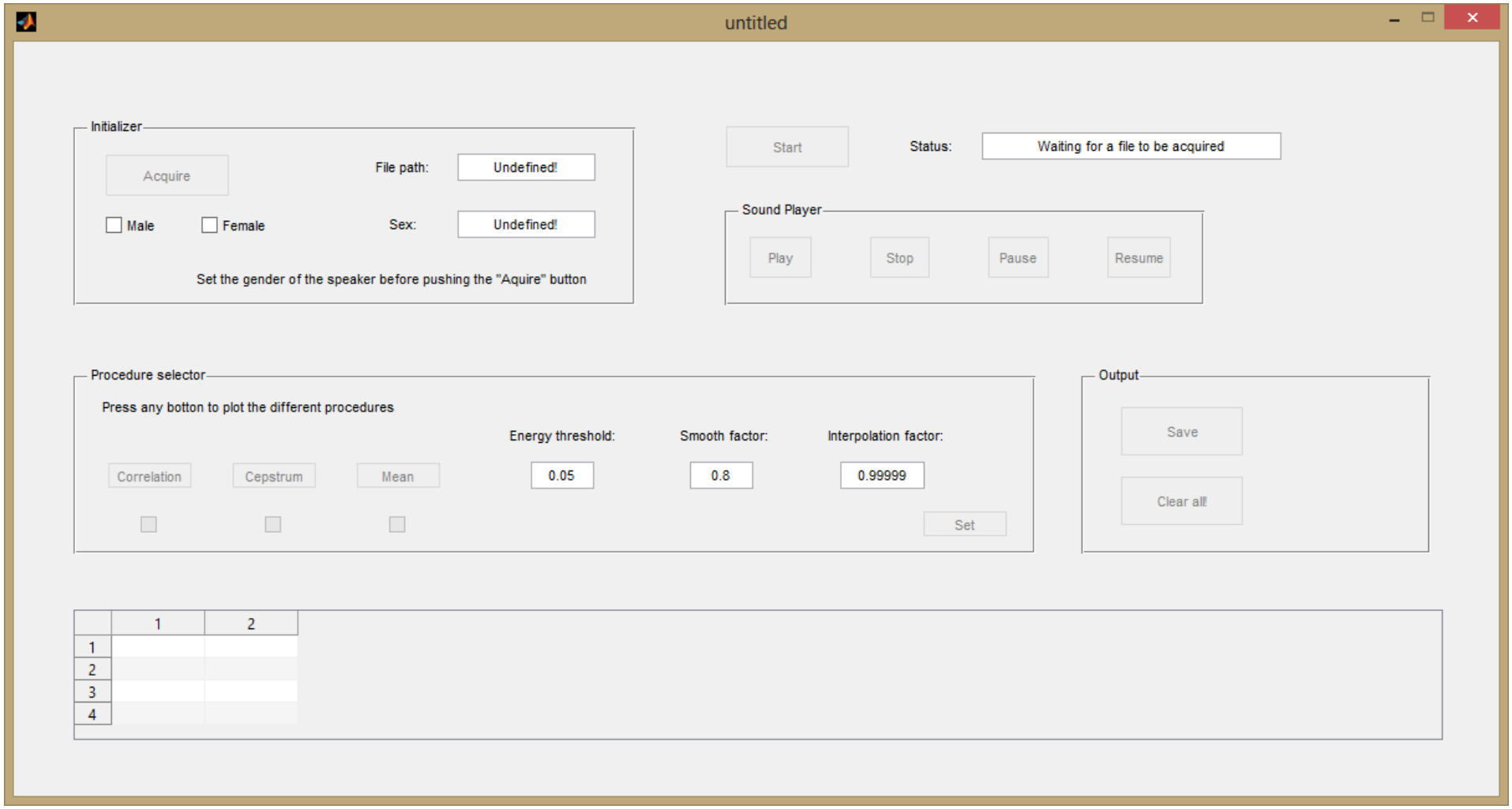
## L’analisi della traccia audio

Questa guida è destinata all'utente che vuole acquisire uno o più file audio con estensione *WAV* tramite interfaccia grafica. Per il corretto funzionamento devono essere presenti nella stessa cartella da cui viene lanciata l'interfaccia i due file che servono per l'avvio della *GUI* e le funzioni che essa richiama:

- *emovo_features.m*
- *feat_mom.m*
- *feat_sound.m*
- *filtro_ch2.m*
- *filtro_pb.m*
- *initMixture.m*
- *my_speech_proc.m*
- *pitch_feat_gmm.m*
- *pitch_feat_tr.m*
- *select_sentence.m*
- *set_emotion.m*
- *setB_sex.m*
- *smooth_pitch.m*

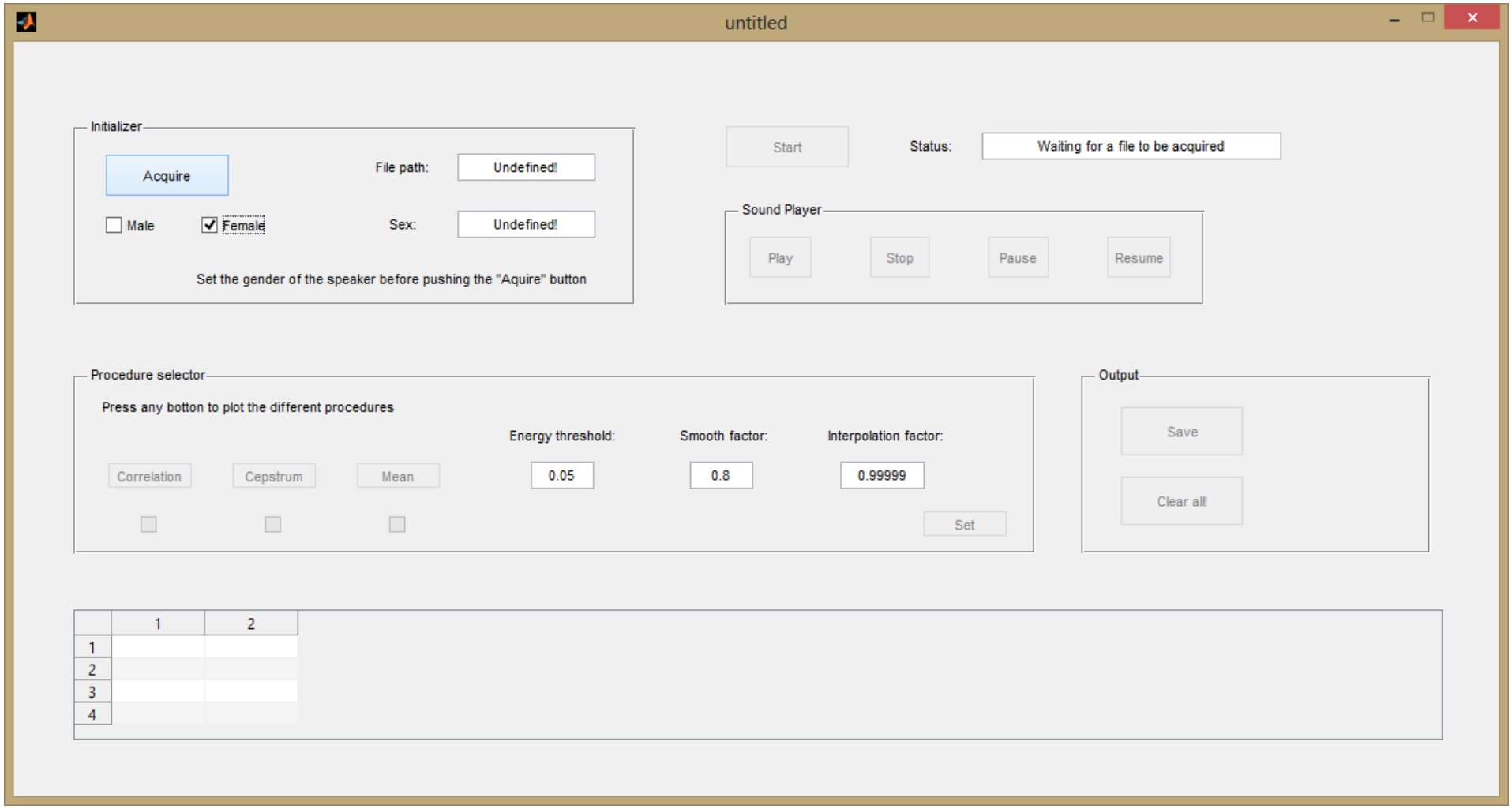
Dopo aver verificato la presenza dei file all'interno della cartella è possibile procedere con l'acquisizione seguendo passo dopo passo i punti sottostanti.

## Avvio dell'interfaccia

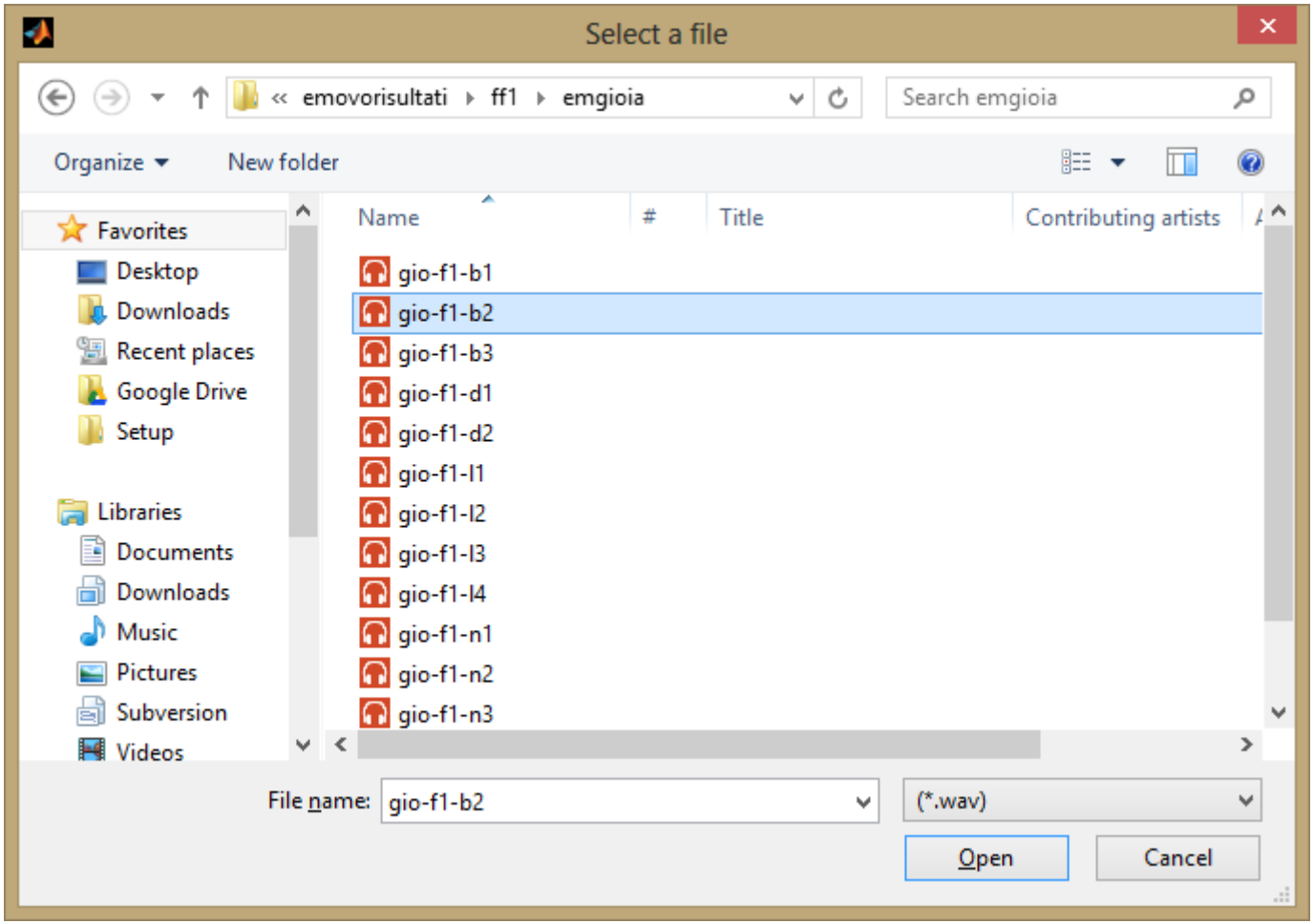



Questa è l'interfaccia della *GUI* una volta lanciata.

## Selezione del sesso del parlatore

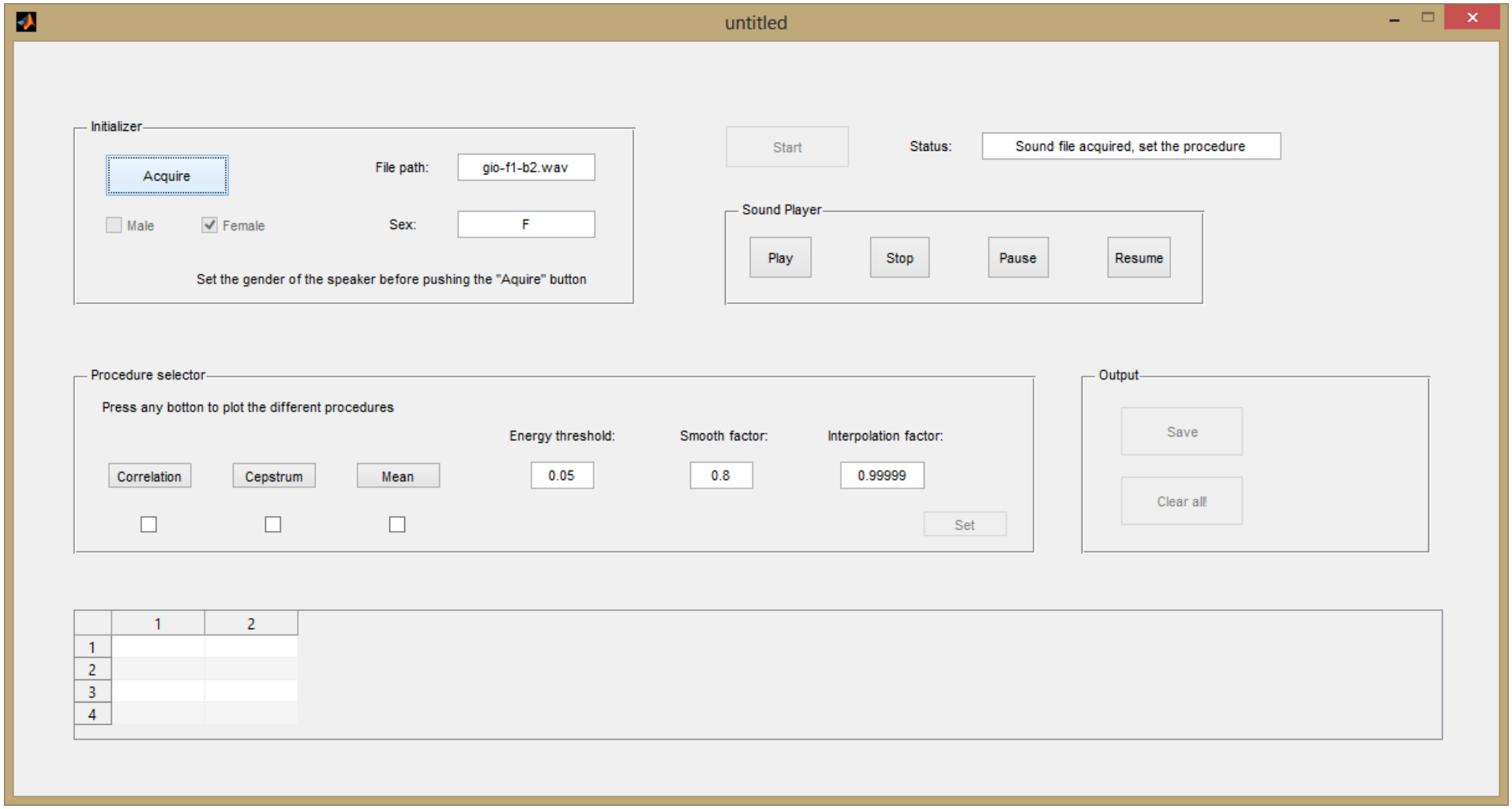



Scegliere il sesso del parlatore per poter abilitare il pulsante di acquisizione.



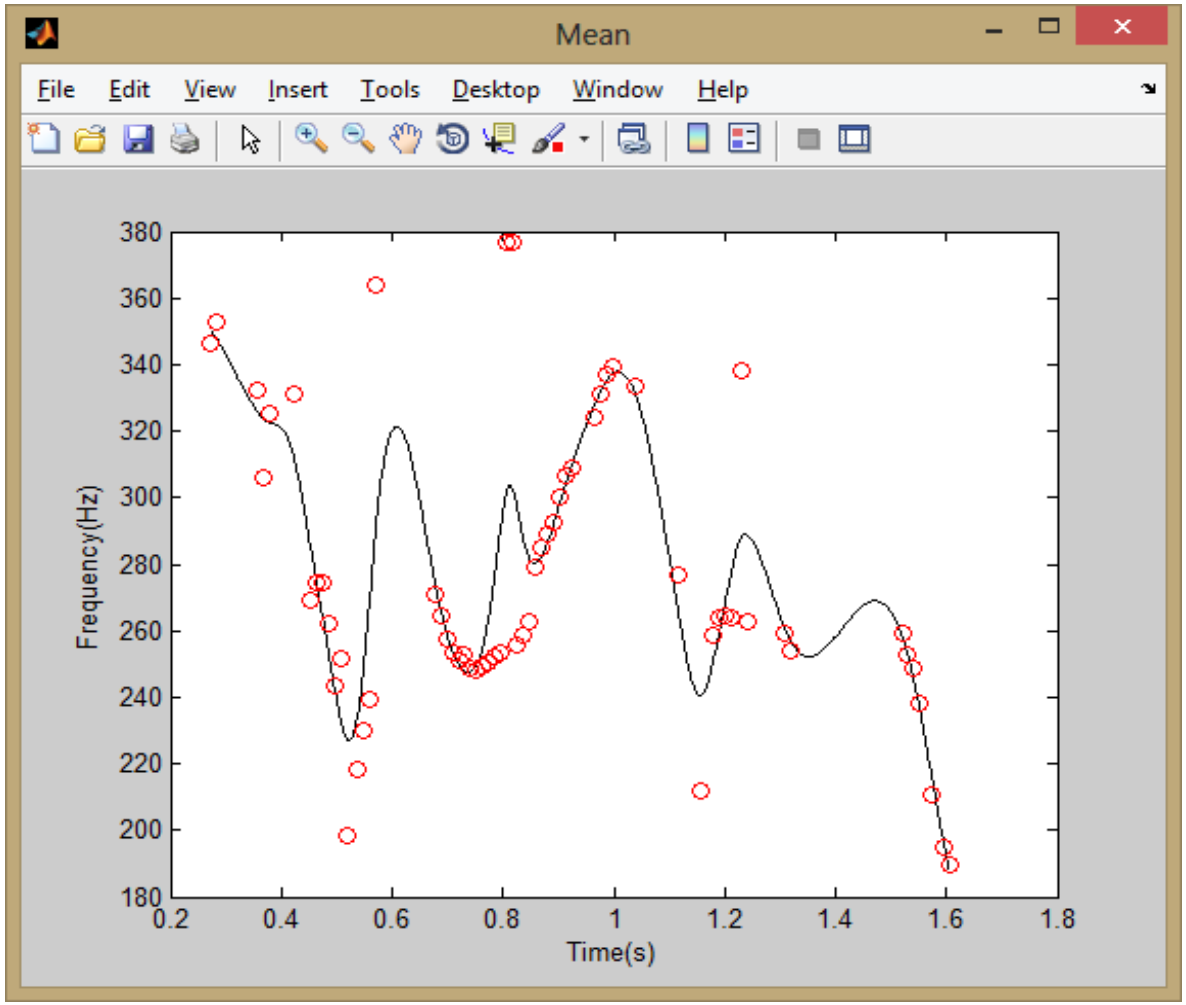

Dopo aver cliccato sul tasto di *Acquire* si aprirà la finestra soprastante che permetterà di sfogliare tra le cartelle e selezionare il file audio desiderato. Per facilitare l'utente la finestra mostra solo file con estensione *wav*.

## Acquisizione avvenuta e riproduzione del file audio



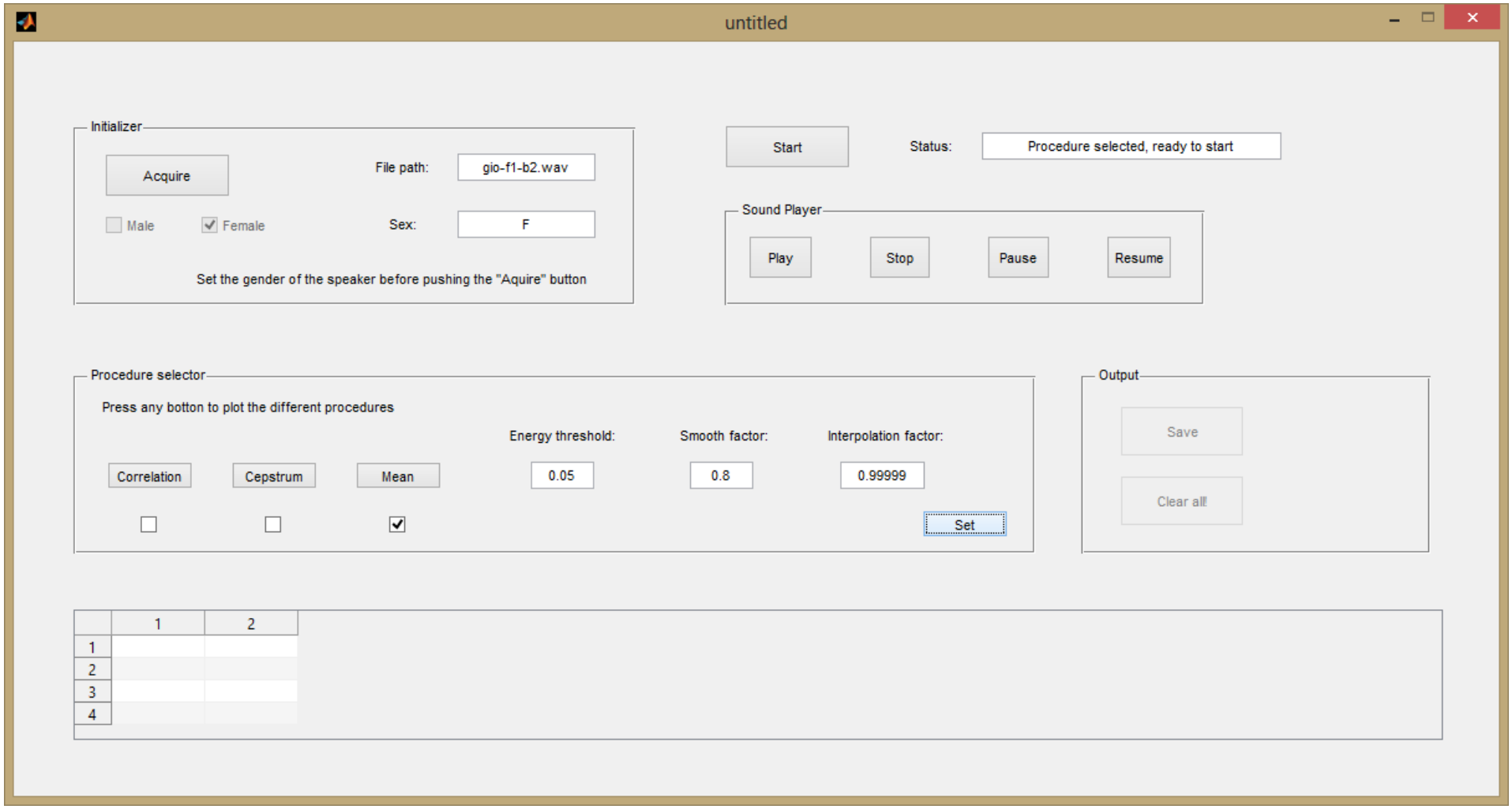

Il *File path* e *Sex* sono stati aggiornati col nome del file audio ed il sesso scelto in precedenza. Come si può vedere tramite lo *Status* il programma ha acquisito il file e inoltre si può procedere con la riproduzione del file acquisito tramite il *Sound Player*.

## Grafico del metodo che si desidera



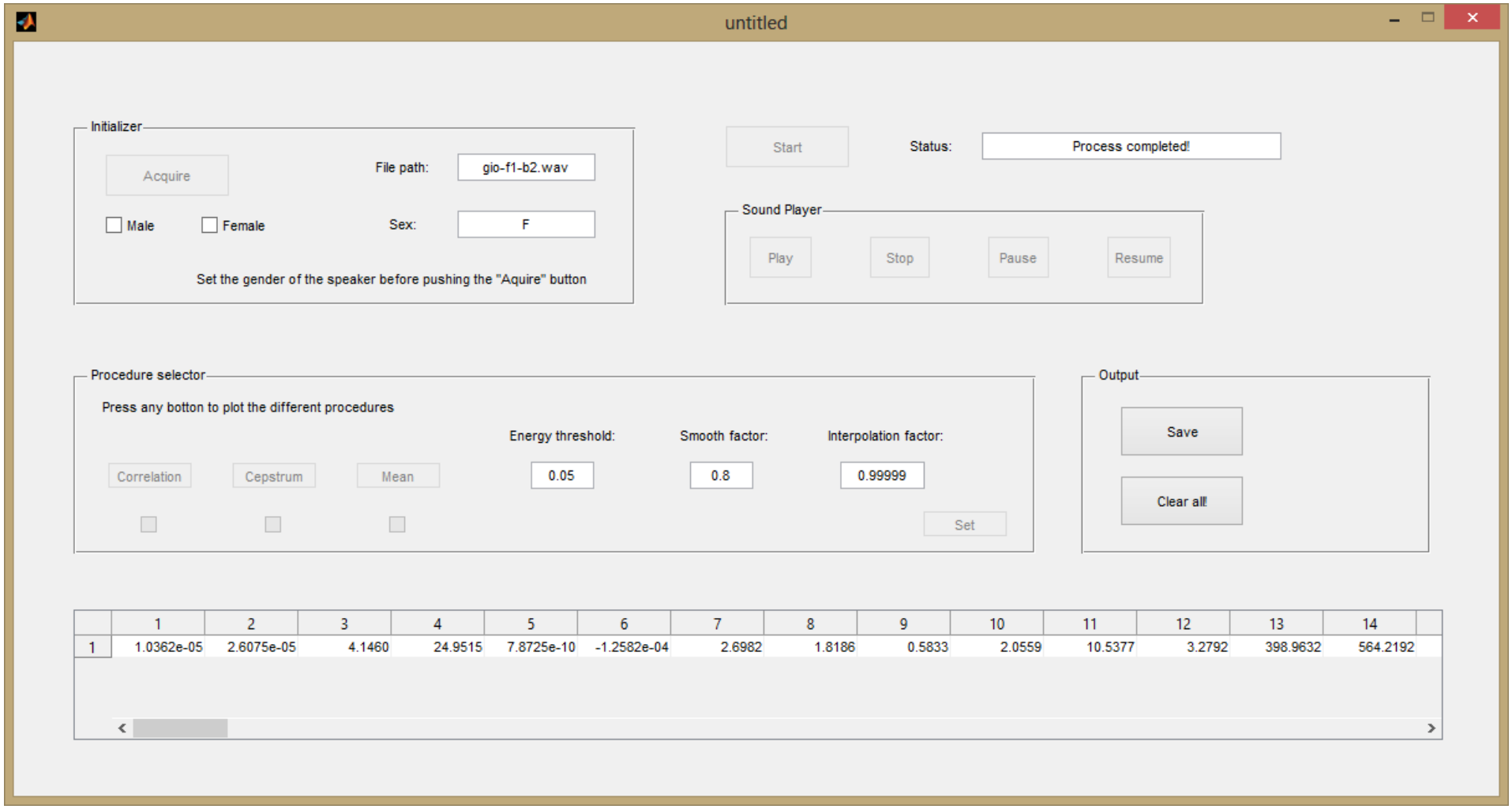

Prima di scegliere uno dei tre metodi disponibili è possibile visualizzare l'andamento del *pitch* tramite i pulsanti con scritto *Correlation*, *Cepstrum* o *Mean*. Modificando opportunamente i valori *Energy Threshold*, *Smooth factor* e *Interpolation factor* è possibile modificare l'andamento del *pitch* per il metodo scelto.

## Calcolo attraverso il metodo scelto



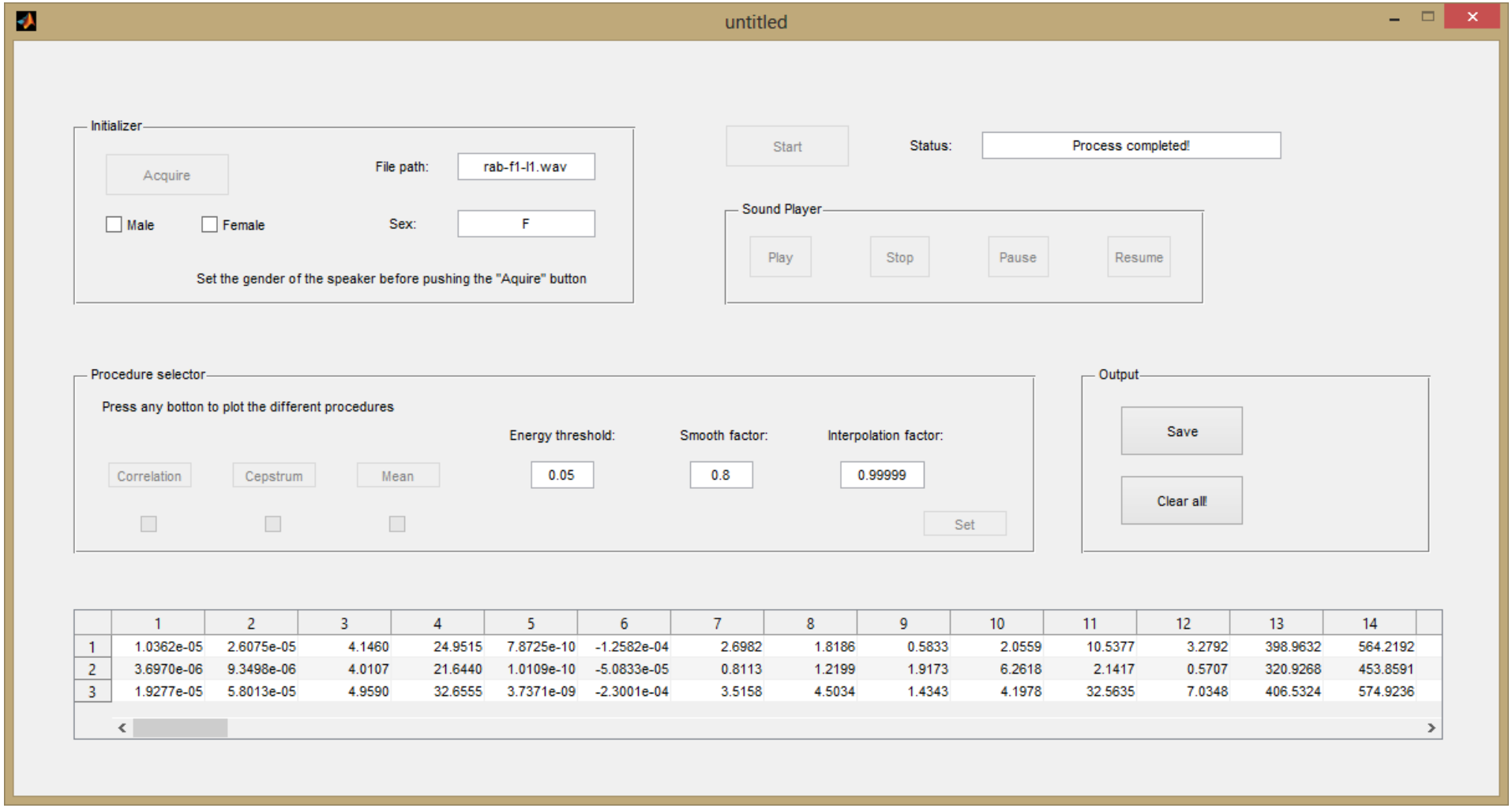

A questo punto per procedere con il calcolo bisogna spuntare il metodo scelto, modificare se necessario i valori *Energy Threshold*, *Smooth factor* e *Interpolation factor*, cliccare su *Set* ed infine su *Start*.

## Completamento del processo




Si può notare tramite lo *Status* che il processo è andato a buon fine e che il programma ha creato una riga riguardante il file acquisito.

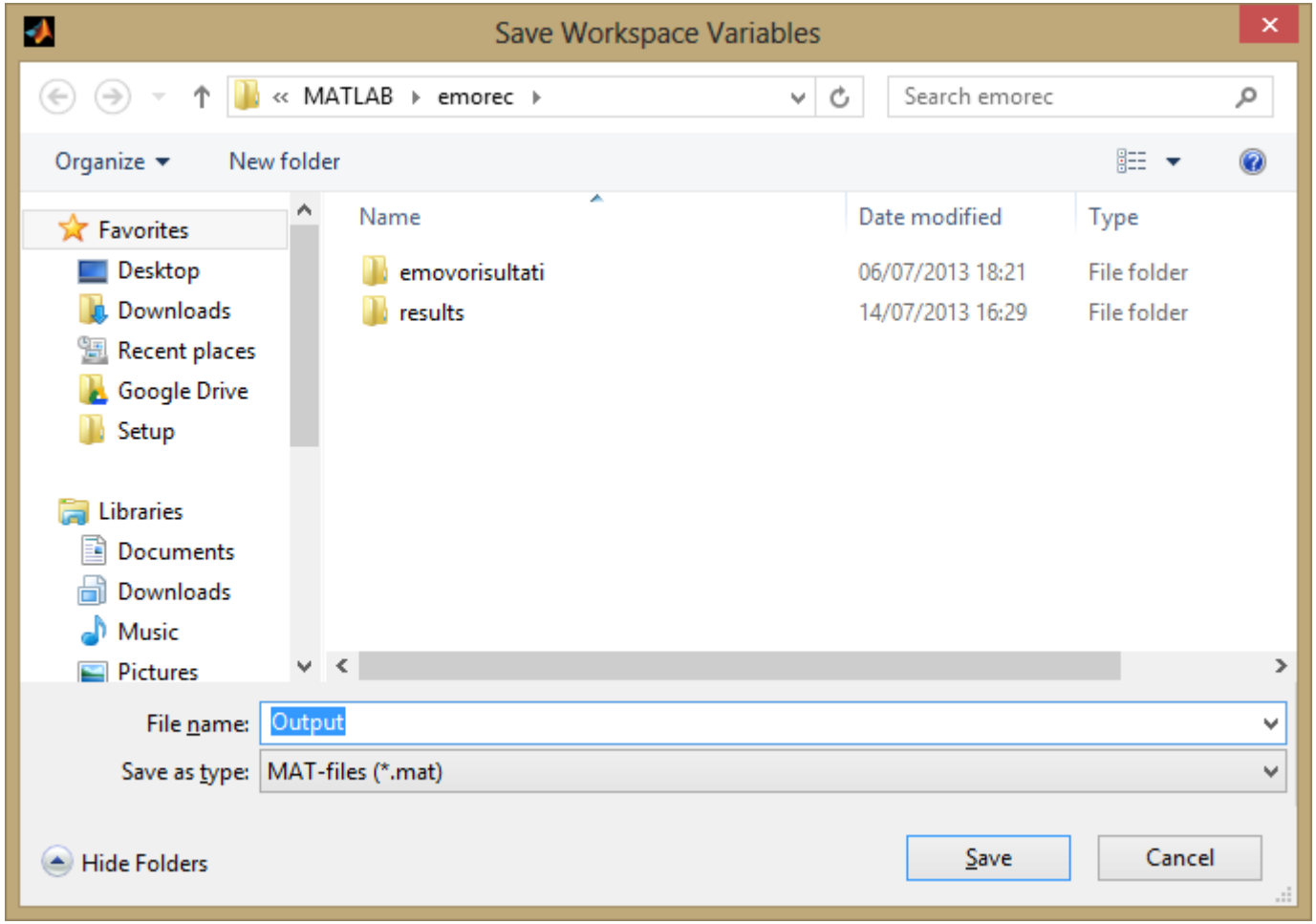
## Acquisizione di più file




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.0362e-05 | 2.6075e-05 | 4.1460 | 24.9515 | 7.8725e-10 | -1.2582e-04 | 2.6982 | 1.8186 | 0.5833 | 2.0559 | 10.5377 | 3.2792 | 398.9632 | 564.2192 |
| 2 | 3.6970e-06 | 9.3498e-06 | 4.0107 | 21.6440 | 1.0109e-10 | -5.0833e-05 | 0.8113 | 1.2199 | 1.9173 | 6.2618 | 2.1417 | 0.5707 | 320.9268 | 453.8591 |
| 3 | 1.9277e-05 | 5.8013e-05 | 4.9590 | 32.6555 | 3.7371e-09 | -2.3001e-04 | 3.5158 | 4.5034 | 1.4343 | 4.1978 | 32.5635 | 7.0348 | 406.5324 | 574.9236 |

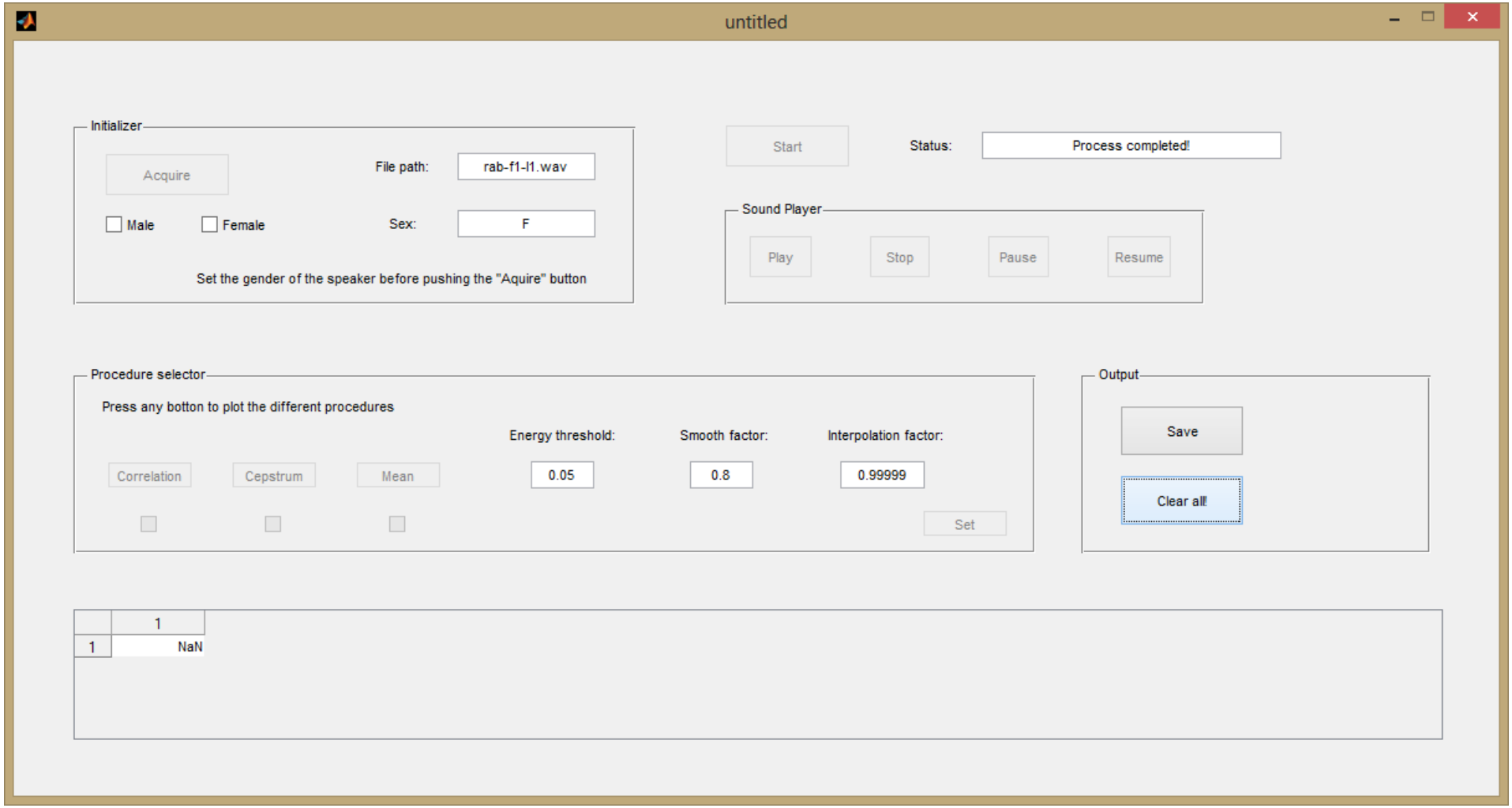
Ripercorrendo i passi fatti fino ad ora è possibile acquisire altri file audio che andranno ad aggiungersi nella riga successiva dell'ultimo file acquisito.

## Salvataggio



Tramite il pulsante *Save* è possibile esportare la matrice dei file acquisiti che è visualizzata nell'interfaccia. La matrice viene salvata con estensione *mat*

## Pulizia del workspace




Cliccando sul pulsante *Clear all* si eliminano i risultati ottenuti tramite la pulizia del *workspace*.

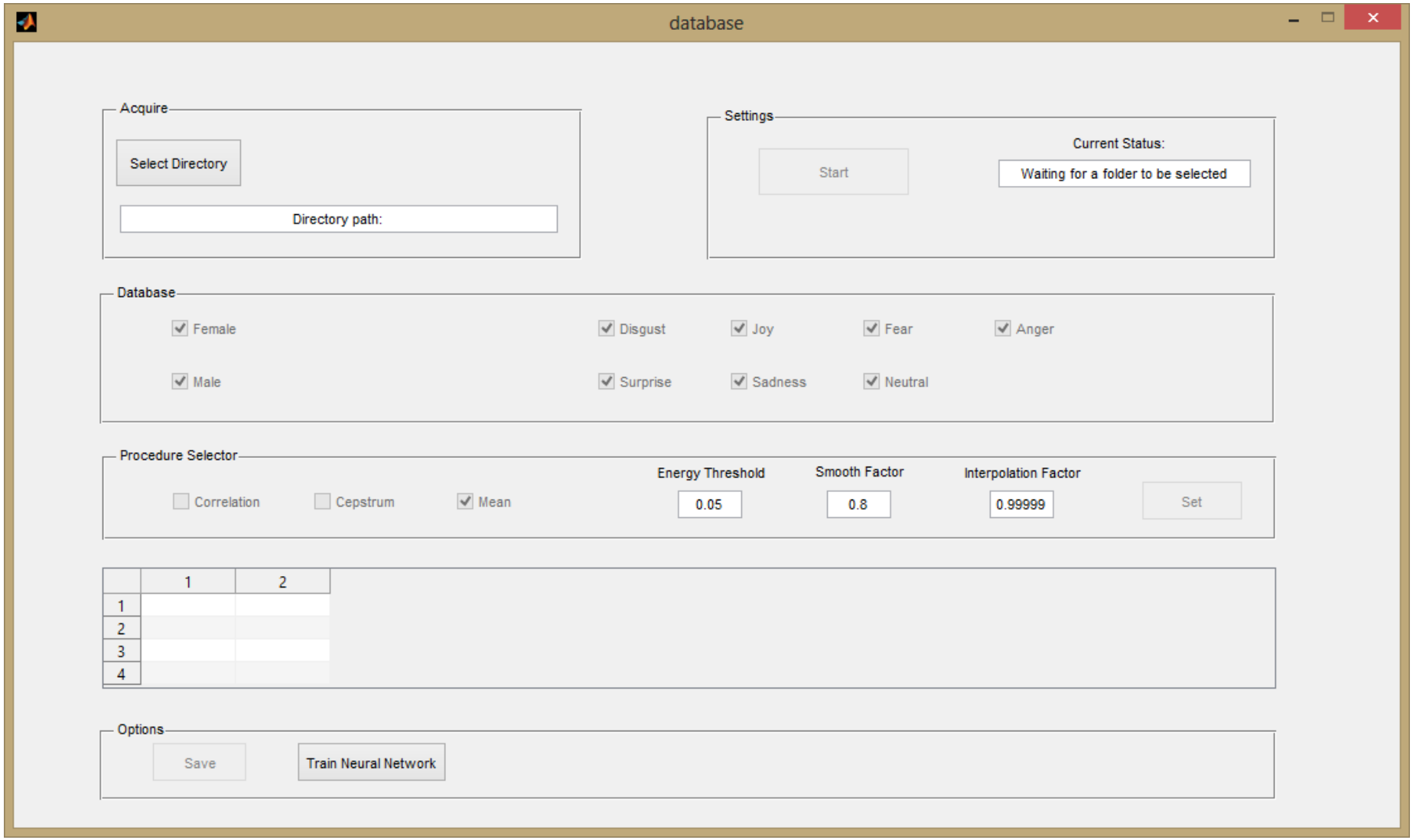
## L’analisi del database audio

Al contrario della precedente questa guida è destinata all'utente che vuole acquisire una cartella contente file audio tramite interfaccia grafica. Per il corretto funzionamento devono essere presenti nella stessa cartella da cui viene lanciata l'interfaccia i due file che servono per l'avvio della GUI e le funzioni che essa richiama:

- *feat_mom.m*
- *feat_sound.m*
- *filtro_ch2.m*
- *filtro_pb.m*
- *initMixture.m*
- *my_speech_proc.m*
- *pitch_feat_gmm.m*
- *pitch_feat_tr.m*
- *select_sentence.m*
- *set_emotion.m*
- *setB_sex.m*
- *smooth_pitch.m*

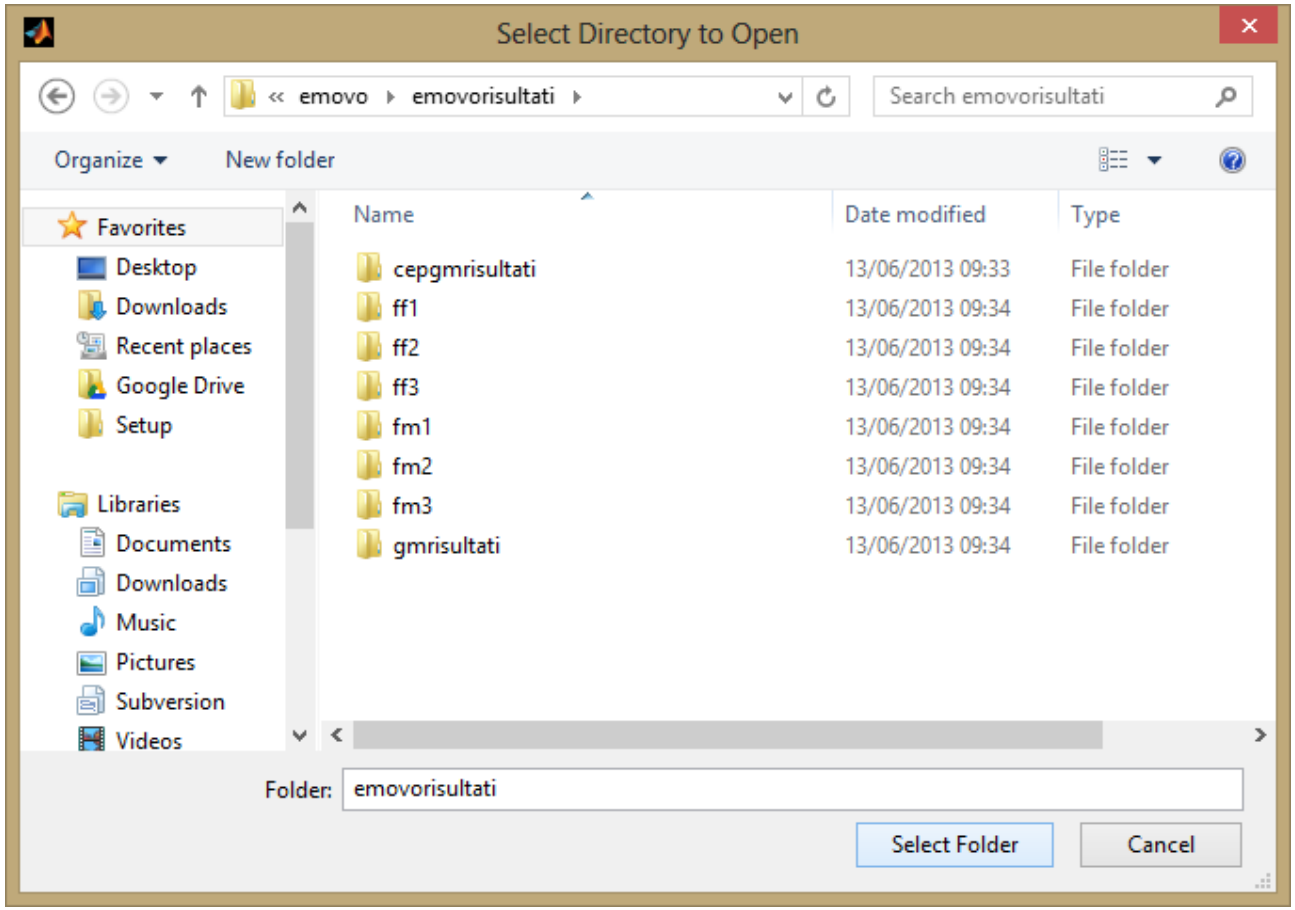
Dopo aver verificato la presenza dei file all'interno della cartella è possibile procedere con l'acquisizione seguendo passo dopo passo i punti sottostanti.

## Avvio dell'interfaccia



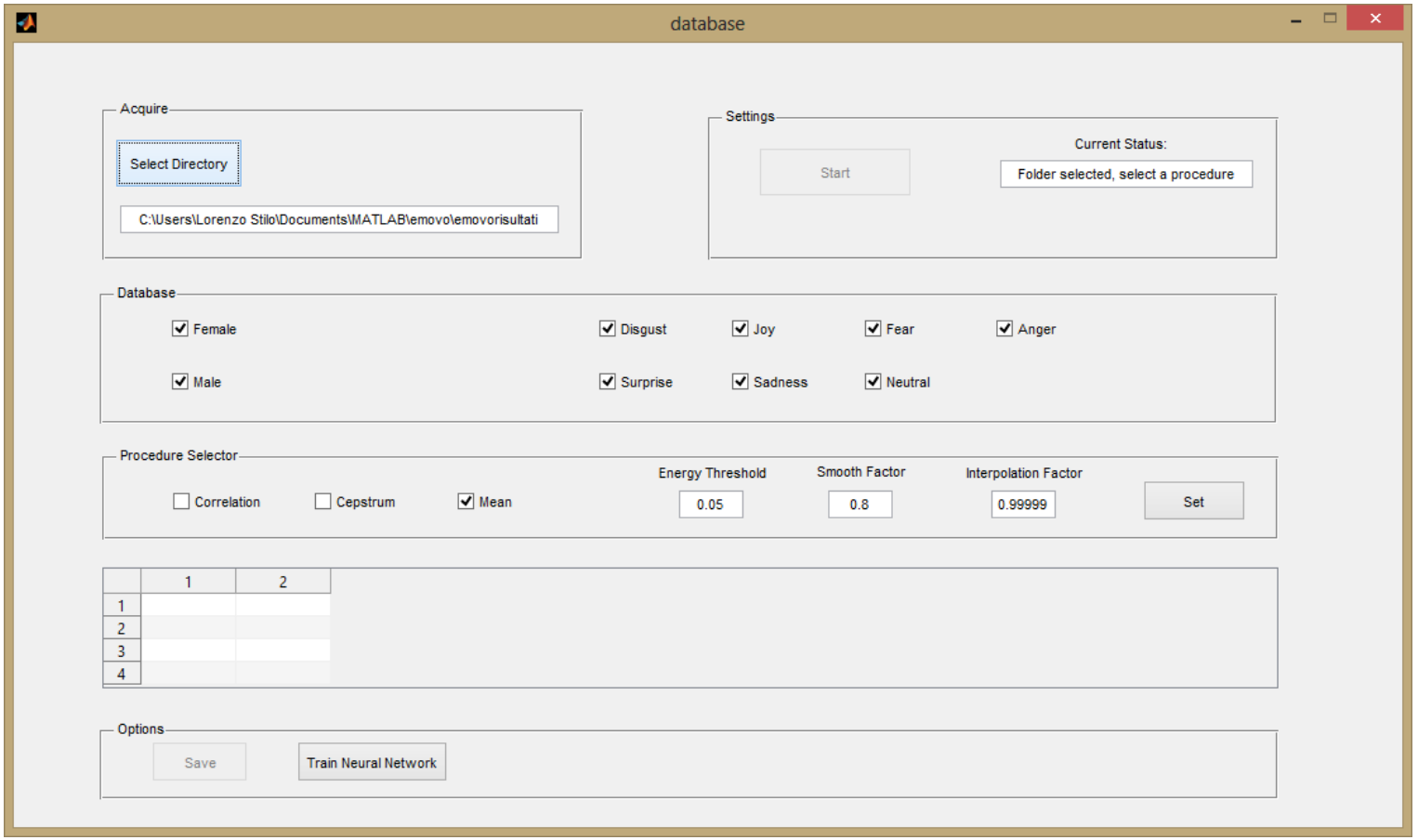

La figura soprastante è la *GUI* appena avviata.

## Selezione della cartella




Cliccando su pulsante *Select Directory* apparirà la finestra soprastante che permette di sfogliare fra le cartelle e selezionare tramite il pulsante *Select Folder* la cartella che contiene i file audio desiderati.

## Impostazione dei parametri

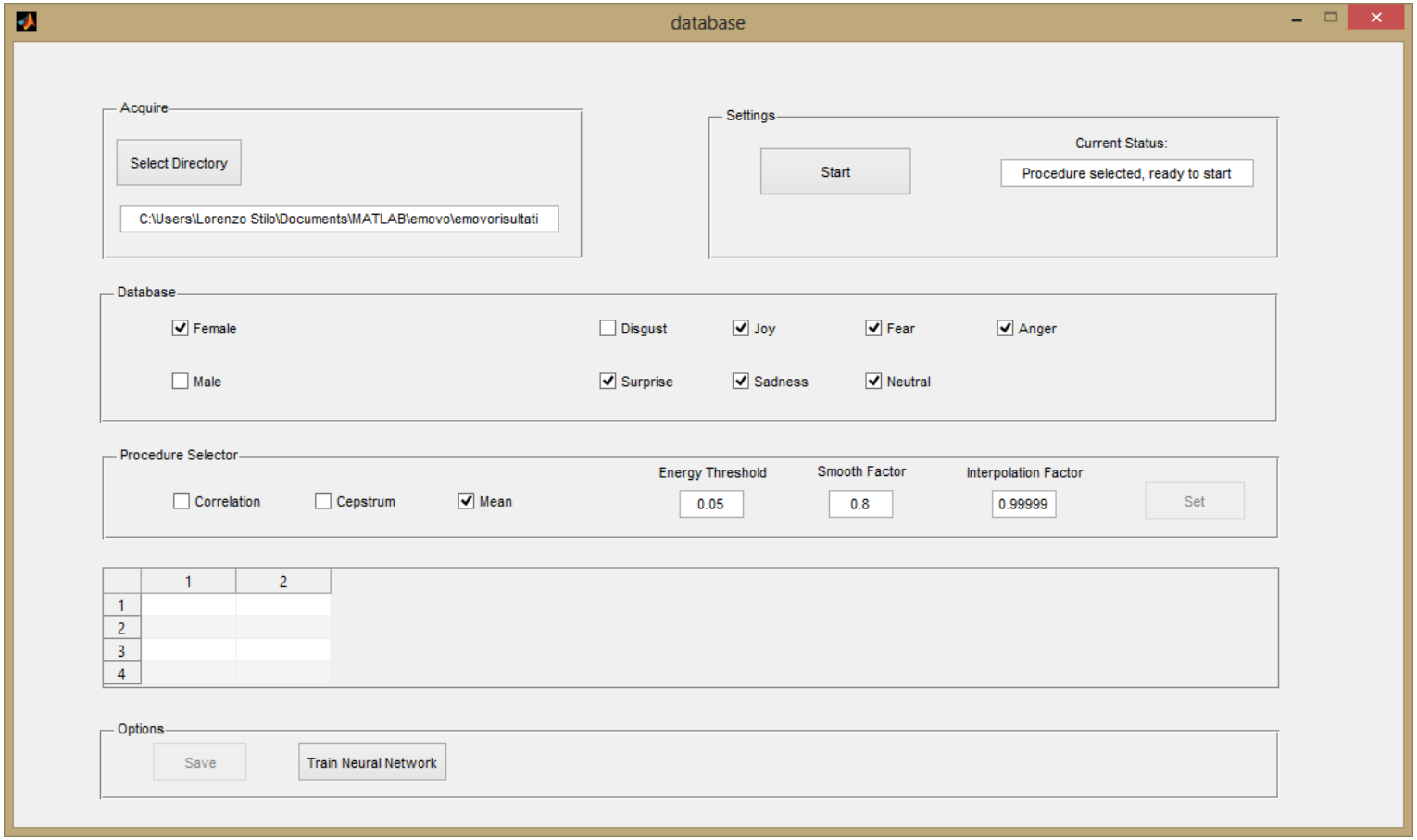



Una volta acquisito l'indirizzo della cartella si potrà leggere in *Current Status* che la cartella è stata selezionata e che si può proseguire. Si attiverà la sezione 'Database' dove sarà possibile spuntare uno o entrambi i sessi dei parlatori e le emozioni dei file audio.

Dopo aver spuntato i sessi e le emozioni è possibile settare i parametri *Energy Threshold*, *Smooth factor* e *Interpolation factor* e si deve scegliere la procedure tra *Correlation*, *Cepstrum* o *Mean*.

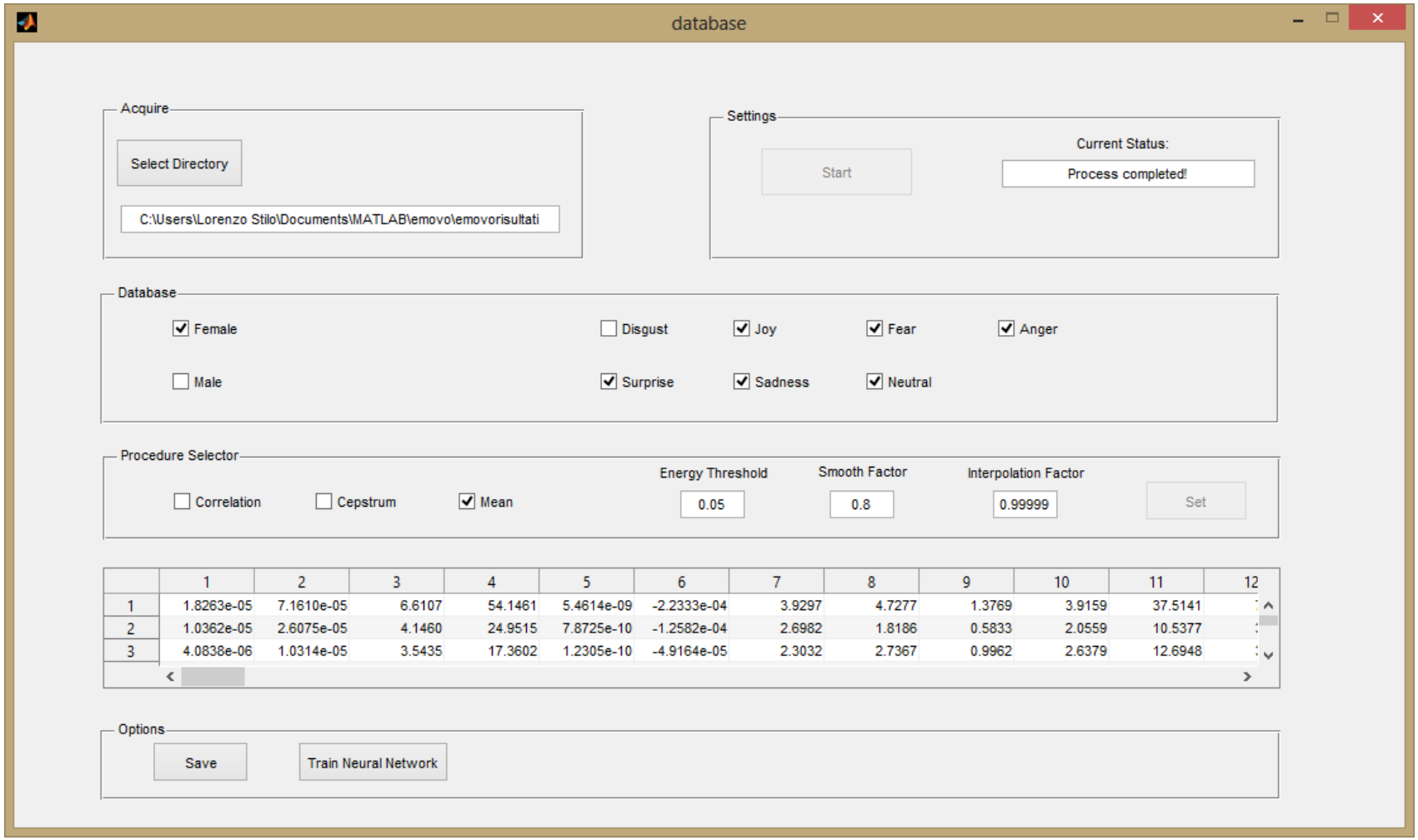
A questo punto bisogna cliccare sul pulsante *Set* per settare i parametri scelti.

## Calcolo della matrice



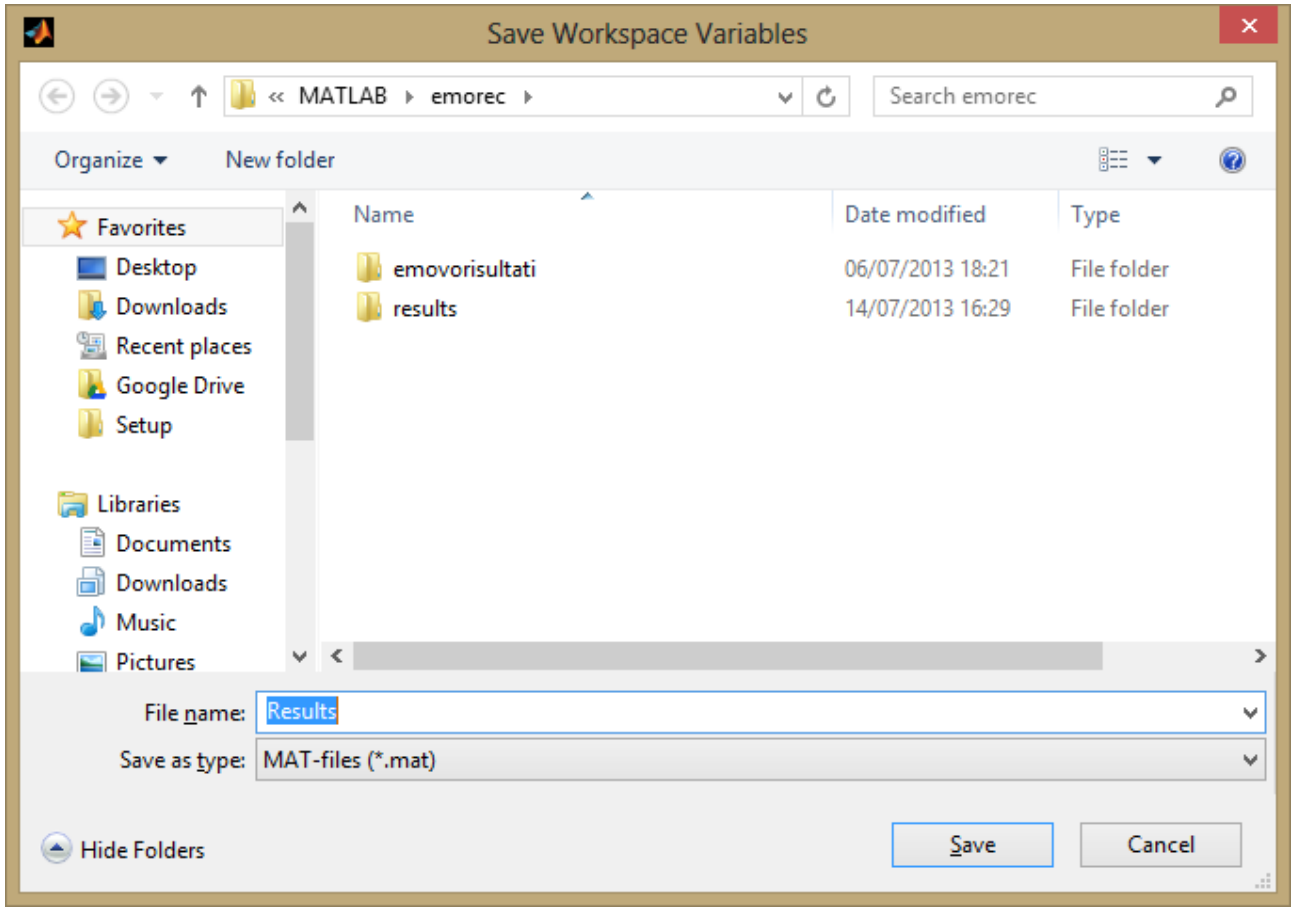

Si nota che nel *Current Status* che il programma è pronto ad generare la matrice che servirà per addestrare la rete neurale. A questo punto basterà cliccare su *Start*.

## Esportazione della matrice generata



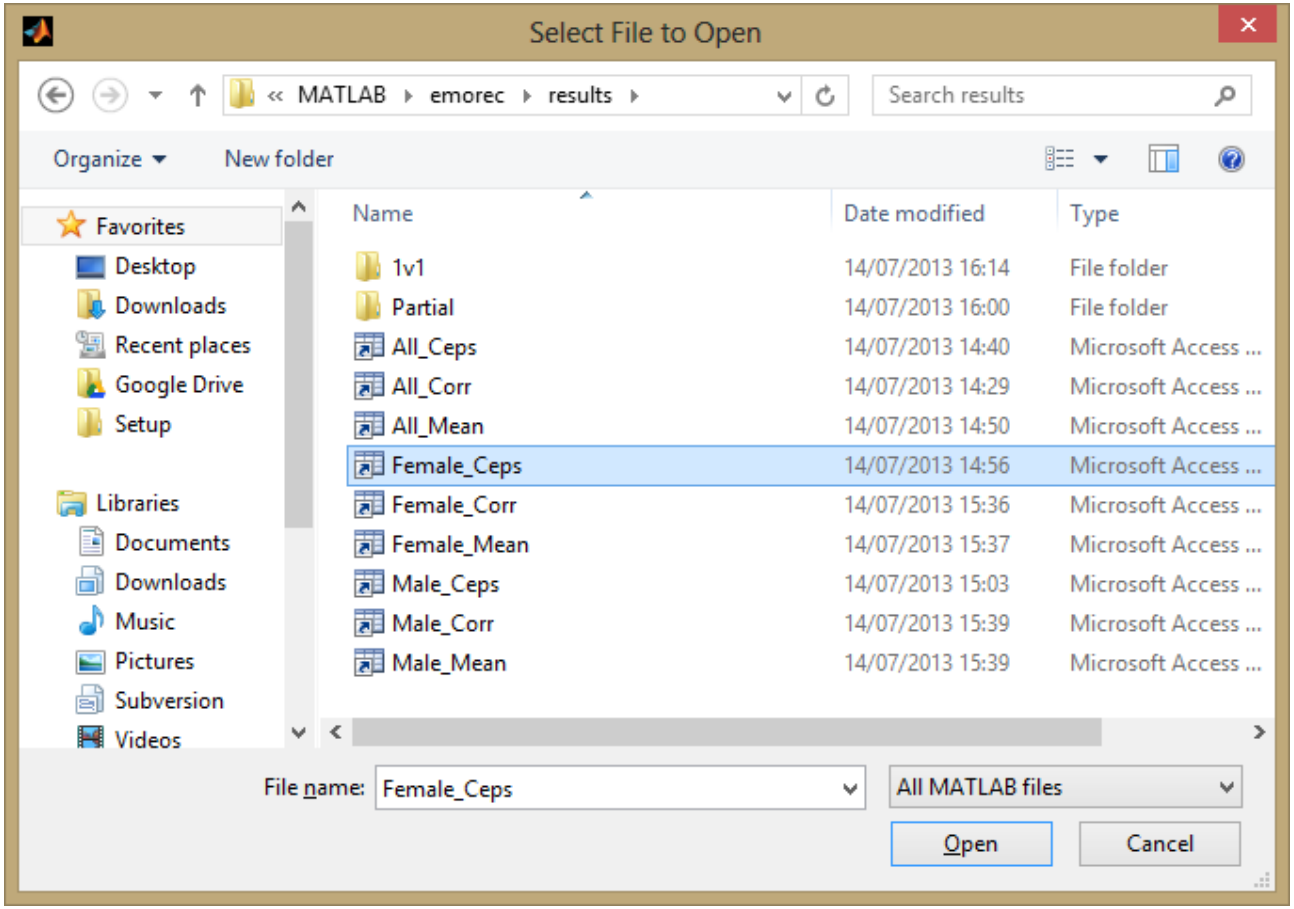

La matrice è stata generata ed è visualizzata a schermo. A questo è necessario esportarla premendo sul pulsante *Save* prima di poter proseguire.

## Locazione di salvataggio



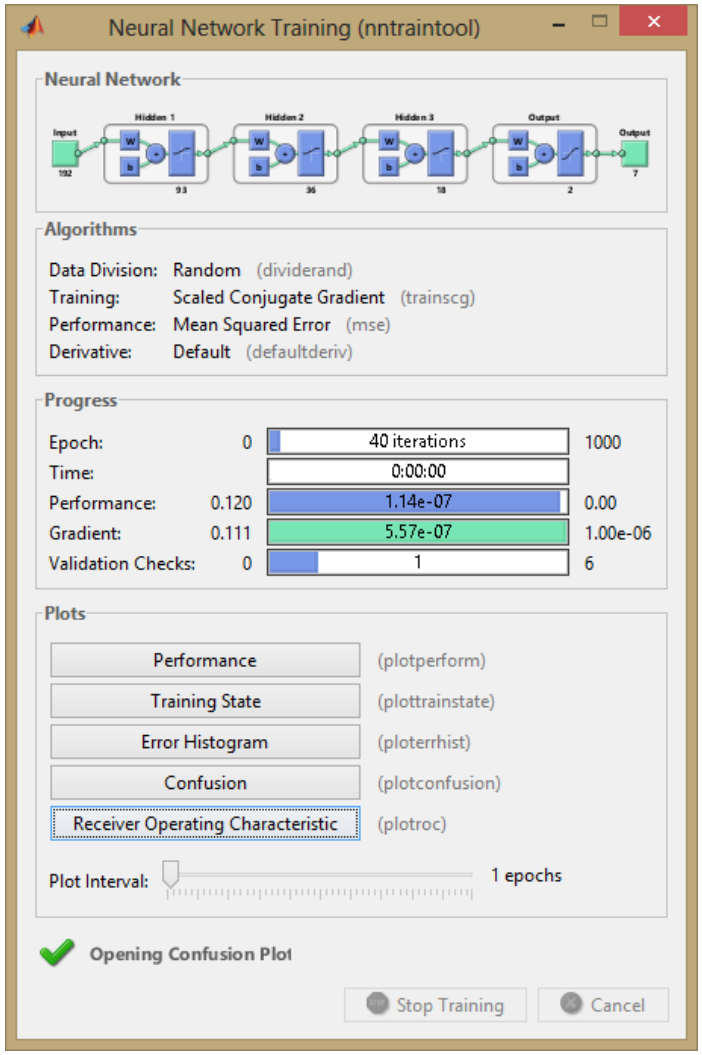

Bisogna scegliere dove salvare la matrice che stiamo esportando con estensione *MAT*.

### Selezione del dataset




Ora si può passare all'addestramento della rete neurale. Cliccando sul pulsante *Train Neural Network* apparirà la finestra soprastante che permetterà di scegliere una matrice da noi generata.

## Addestramento della rete




Questa finestra apparirà automaticamente e sta ad indicare il processo dell'addestramento.

## Scelta del Classificatore

Al fine della scelta di un classificatore adatto ai nostri scopi si è optato per una rete neurale. Le principali motivazioni di questa scelta sono state due: la prima è il valore didattico di questo genere di classificatori la seconda è la versatilità degli stessi.

### Le reti neurali

Le reti neurali sono caratterizzate da differenti tipologie sia per l'architettura della rete stessa che per la tipologia delle funzioni di trasferimento dei neuroni che la compongono.

Le scelte possibili erano molteplici e non disponendo di un algoritmo scientifico per il dimensionamento e la scelta dell'architettura si è proceduto per lo più in maniera euristica fino al conseguimento del miglior risultato possibile.

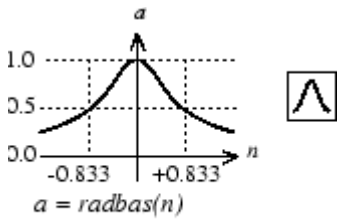
### Architettura della rete neurale

La rete neurale che abbiamo realizzato è di tipo *feed forward* ed è costituita da uno strato di neuroni in ingresso costituito da un numero che corrisponde alle cifre del vettore di caratterizzazione.

A questa segue uno strato definito *nascosto* che è il vero cuore del classificatore: la scelta del comportamento e del dimensionamento di questo strato influiscono in egual misura sulle prestazioni della rete e sulla sua complessità della stessa. Inoltre è opportuno sottolineare che il sovradimensionamento di questo strato di neuroni può portare all'inutilizzabilità del classificatore.

Tenendo conto di quanto sopra si è scelto di porre 180 neuroni in un unico strato nascosto per la caratterizzazione delle sette classi emozionali in esame.

Per quanto riguarda la funzione di trasferimento di questo strato nascosto si è scelto di utilizzare le *Radial Basis Function* che sono caratterizzate da gaussiane il cui comportamento è riportato in seguito.




Radial Basis Function

Infine lo strato d'uscita è vincolato alla tipologia della rete stessa. Nei test che abbiamo condotto sono state utilizzate due diverse tipologie di rete.

La prima esclusivamente *feed forward* ha uno strato di neuroni d'uscita la cui funzione di trasferimento è di tipo puramente lineare. La seconda finalizzata al riconoscimento di *pattern* ha uno strato in uscita di tipo *sigmoidale* con un andamento che ha minimo in -1 e massimo in +1.

## Modalità di addestramento e realizzazione del Classificatore

Il classificatore è stato realizzato mediante l'ambiente di sviluppo *MATLAB* già spesso citato in questo documento.

Uno dei fattori più importanti per la scelta del classificatore è risultato quello del dimensionamento dei dati in ingresso alla rete[1].

[1] '*Dataset*' della rete neurale.

Si è scelto di procedere in questo modo: dopo aver caratterizzato l'intero database EMOVO tramite l'utilizzo dell'interfaccia grafica da noi realizzata, si è proceduto alla suddivisione randomica[2] dei dati ottenuti per la creazione di tre diverse matrici che MATLAB utilizzerà rispettivamente per il *Training*, per il *Validation* e per il *Test*. Questa divisione randomica è realizzata in modo tale che le tre sottomatici generate siano sempre la prima il 70% del totale la seconda il 15% e la terza il 15%. In questo modo abbiamo quindi dimensionato le tre fasi della creazione del classificatore.

Inoltre è stato necessario definire la tipologia di addestramento della rete. Tra le varie possibilità offerteci dall'ambiente di *MathWorks* si è scelto *Gradient descent with momentum backpropagation*.I parametri utilizzati per la scelta delle epoche e il minimo errore accettabile sono in seguito definiti nello script qui allegato.

```
%% Chiudo tutte le finestre aperte e pulisco il Command window e il
Workspace
close all
clear all
clc

%% Acquisisco la il mio file di estensione 'mat'
[file,path]=uigetfile('*.mat*');     % Scelgo il mio file
load([path,file]);                   % Acquisisco un file con i risultati
clear path file;                     % Pulisco 'path' e 'file' che non sono
più necessari

%% Creo l'input e il target per la rete neurale
inputs = feat_speech;

label_speech=zeros(7,length(gt_speech));    % Crea la matrice dei label
per l'addestramento della rete neurale
for i=1:length(gt_speech)
    for j=1:7
        if (gt_speech(i)==j)
            [j i]
            label_speech(j,i)=1;
        end
    end
end
targets = label_speech;

clear i j;      % Pulisco gli indici 'i' e 'j'

%% Creo la rete neurale, divido il dataset e definisco le funzioni degli
strati
net = feedforwardnet([180]);        % Creo la rete neurale

net.divideFcn = 'dividerand';           % Divido il dataset in modo
casuale
net.divideParam.trainRatio = 0.8;
net.divideParam.valRatio = 0.1;
net.divideParam.testRatio = 0.1;

net.layers{1}.transferFcn = 'radbas';   % Scelgo il tipo di funzione per
lo strato

%% Definisco il tipo di addestramento e i suoi parametri
net.trainFcn = 'traingdm';
```

[2] E' stata utilizzata la funzione *MATLAB* '*dividerand*'.

```
net.trainParam.show = 25;
net.trainParam.goal = 1e-5;
net.trainParam.epochs = 10000;
net.trainParam.mc = 0.7;

%% Addestro la rete e vedo graficamente i risultati
[net,tr] =
train(net,inputs,targets,'showResources','yes','useParallel','yes','useGPU
','yes');

outputs = net(inputs);
errors = gsubtract(targets,outputs);
performance = perform(net,targets,outputs);

%view(net)
plotconfusion(targets,outputs)
```

## Risultati

Trattando una rete neurale il cui *dataset* è composto randomicamente i risultati ottenuti possono variare di molto in base alla divisione che in quel momento compone le matrici di addestramento, validazione e test. Per ovviare a questo problema si è scelto di eseguire un numero consistente di prove e mediare sulle prestazioni ottenute da ciascuna di esse.

Le prove effettuate si sono suddivide in diverse tipologie:

- Intero database senza distinzioni sul sesso del parlatore
- Intero database creando due diversi classificatori in base al sesso del parlatore
- Database mancante della classe emozionale di *disgusto*
- Confronto tra sole due emozioni senza distinzione sul sesso del parlatore
- Confronto tra sole due emozioni creando due diversi classificatori in base al sesso del parlatore

Per ciascuna tipologia l'architettura della rete è stata leggermente modificata per adattarsi al differente numero di input.

I risultati ottenuti hanno evidenziato ottime prestazioni quando al classificatore è chiesto di distinguere tra sole due emozioni, con un andamento tendente al peggioramento all'introduzione di più possibili stati emozionali. Nonostante questo, agendo sul dimensionamento della rete e su una più fine suddivisione del *dataset* le risposte corrette sono risultate comunque multo superiori alla metà dei test in esame.

Le prestazioni aumentano ulteriormente se, come citato, si sceglie di suddividere il classificatore in base al sesso del parlatore acquisendo due diversi *pesi sinaptici*.

In seguito sono riportati i risultati ottenuti tramite il comando *plotconfusion* di MATLAB. Queste tabelle mostrano come sono stati classificati gli audio analizzati. In seguito sono riportati tre esempi e per ognuno di essi ci sono quattro tabelle che mostrano come la rete neurale è riuscita a classificare l'emozione per le seguenti fasi:

- *Training*
- *Validation*
- *Test*
- *All*

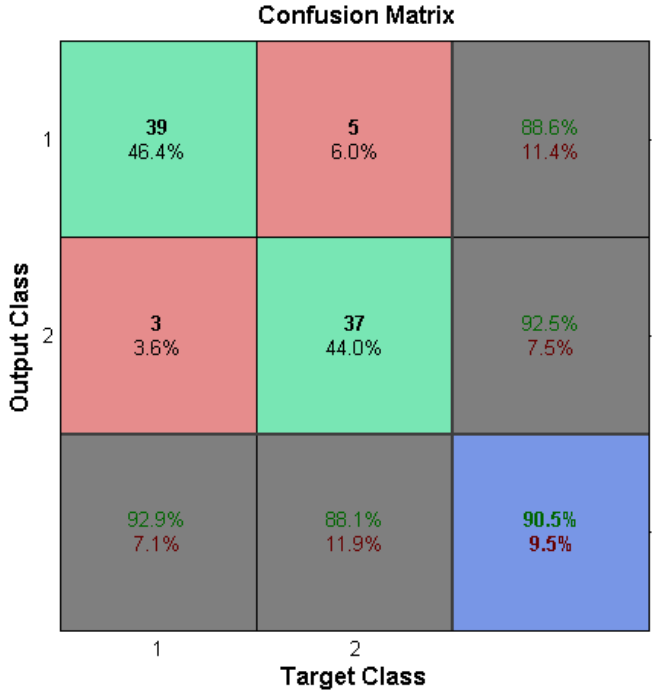
Per ogni esempio trattato è stato scelto l'algoritmo di *Backpropagation* con momento pari a 0.7 con un numero di epoche pari a 10000 che molto spesso non veniva raggiunto.

La rete è formata da uno strato con una *Squashing Function* di tipo *radbas* con un numero di neuroni opportuno alla grandezza del *dataset* che si analizza ed uno strato di uscita con una *Squashing Function* di tipo *purelin*

Il *dataset* analizzato è stato diviso in modo casuale tramite la funzione *dividerand* con le seguenti percentuali:

- 60% per la fase del *Training*: con tale processo controllo la risposta fornita dalla rete quando in input vengono inseriti valori di variabili indipendenti per cui si conosce l'esatto valore della variabile dipendente.
- 15% per la fase di *Validation*: mentre la rete viene addestrata, ed i pesi modificati, contemporaneamente si monitora la differenza tra i risultati ottenuti sottoponendo alla rete questa porzione del *dataset*.
- 25% per la fase di *Test*: dopo la fase di addestramento effettuata con la fase di training, l'efficacia della rete viene testata su questo insieme di dati.

Facendo riferimento alla tabella della fase di *test* della terza classificazione cerchiamo di capire come leggere la tabella:




Le caselle verdi corrispondono alle classificazioni corrette mentre quelle rosse alle classificazioni sbagliate. In questo caso sono stati classificati correttamente 39 su 44 gli elementi della classe 1 e 37 su 40 elementi della classe 2.

Il processo di classificazione ha *confuso* 5 elementi della classe 1, classificandoli erroneamente nella classe 2. E 3 elementi delle classe 2 classificandoli nella classe 1.

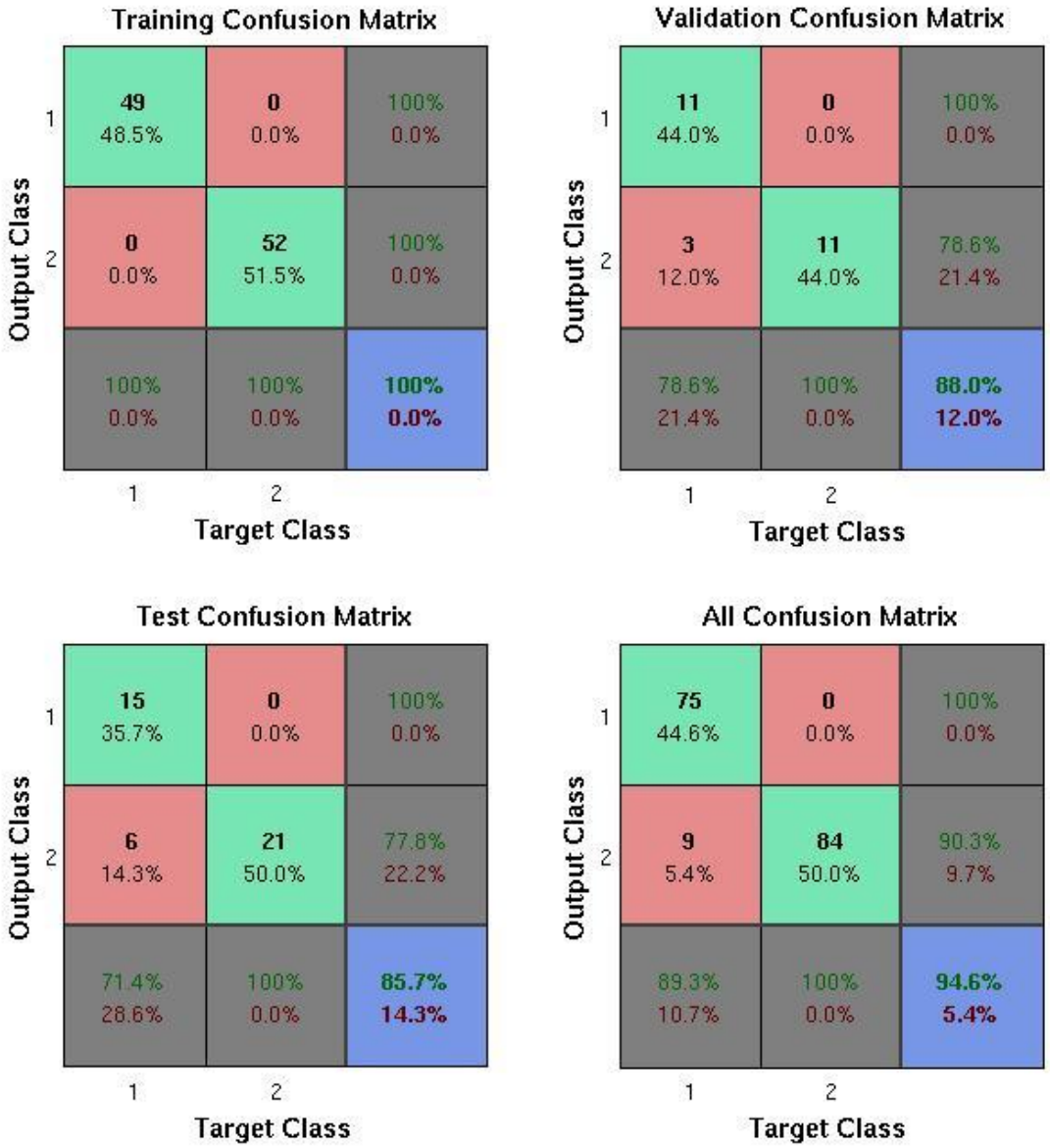
Le caselle grigie sono semplicemente dei totali. Nell'ultima casella della seconda riga si può osservare che c'è una corretta classificazione con il 77.8% ed un 22.2% di errore causati dalla "confusione" dei 6 elementi che è stata fatta dalla classe 1.

Nella casella blu c'è il risultato complessivo della classificazione, indipendentemente dal tipo di errore che è stato commesso.

## Prima classificazione

### Gioia contro Tristezza

Riconoscimento tra la classe emozionale *gioia*, indentificata con il valore 1 e la classe emozionale *tristezza* identificata con il valore 2.



## Seconda classificazione

### Intero database EMOVO

In questa classificazione viene analizzato l'intero database composto da tutte le classi emozionali