VaribankFileMaker

Εργασία στη "Μουσική Πληροφορική" 2018-2019
Ανδρέας Απέργης

1. Εισαγωγή

Η παρούσα εργασία αφορά στη σχεδίαση και υλοποίηση μιας κλάσης για το περιβάλλον SuperCollider. Η κλάση έχει την ονομασία VaribankFileMaker και την ευθύνη της παραγωγής αρχείων συμβατών με το πρόγραμμα Varibank της συλλογής προγραμμάτων "Composers' Desktop Project" (CDP). Στη συνέχεια της παρούσας ενότητας επιχειρείται να γίνει μια εισαγωγή στη συλλογή CDP και στις κατηγορίες προγραμμάτων της. Στη 2η ενότητα παρουσιάζεται το πρόγραμμα Varibank και στην 3η ενότητα η κλάση VaribankFileMaker.

Γενικά Στοιχεία για τη Συλλογή Προγραμμάτων CDP

Η εφαρμογή της συλλογής CDP εστιάζει στο μετασχηματισμό ηχητικού υλικού για τις ανάγκες της σύνθεσης ηλεκτροακουστικής μουσικής και κατ' επέκταση του ηχητικού σχεδιασμού. Προέρχεται και συντηρείται από το ομώνυμο διεθνές συνεργατικό δίκτυο με έδρα το Ηνωμένο Βασίλειο που αποτελείται κυρίως από συνθέτες ηλεκτροακουστικής μουσικής και ερευνητές του πεδίου της ανάπτυξης λογισμικού για ηχητικές εφαρμογές (Chabade 1997: 138). Κάποια εξέχοντα ονόματα που συνέβαλαν στην ανάπτυξη της CDP είναι αυτά του T. Wishart που έχει συγγράψει το μεγαλύτερο μέρος του πηγαίου κώδικά της, του R. Dobson που είναι ο κύριος συντηρητής της και του A. Endrich που είναι υπεύθυνος για το μεγαλύτερο κομμάτι της τεκμηρίωσής της. Η συλλογή διανεμήθηκε δημοσίως για πρώτη φορά το 1987 για το λειτουργικό σύστημα των 16-bit υπολογιστών της εταιρείας Atari ενώ κατά τις επόμενες δεκαετίες ακολούθησαν διανομές για τα λειτουργικά συστήματα Windows και Mac OSX. Μέχρι το 2014 το λογισμικό της συλλογής ήταν κλειστό με διανομή επί πληρωμή. Από τον Οκτώβριο του ίδιου έτους διανέμεται ως ανοικτό λογισμικό συμπεριλαμβάνοντας και υποστήριξη για λειτουργικά συστήματα Linux. Παράλληλα, οι βιβλιοθήκες που το απαρτίζουν μετονομάζονται σε CDP-Wishart Libraries ως φόρο τιμης στον κύριο δημιουργό τους. Ο χρήστης της συλλογής έχει πρόσβαση στα προγράμματά της μέσω της γραμμής εντολών του εκάστοτε λειτουργικού συστήματος ενώ παρέχεται και η δυνατότητα περιορισμένης χρήσης των περισσοτέρων προγραμμάτων μέσω γραφικών διεπαφών. Ευρέως διαδεδομένες γραφικές διεπαφές αποτελούν τα προγράμματα Sound Shaper του R. Fraser (Windows) και Sound Loom του T. Wishart (Windows και Mac OS). Τέλος, σημαντικό χαρακτηριστικό της συλλογής αποτελεί η λειτουργία των προγραμμάτων της αποκλειστικά σε μη-πραγματικό χρόνο.

Κατηγορίες Προγραμμάτων της Συλλογής CDP

Τα προγράμματα της συλλογής είναι περισσότερα από 500 και για την διευκόλυνση του χρήστη είναι κατηγοριοποιημένα βάσει του τύπου των μετασχηματισμών που υποβάλλεται το ηχητικό υλικό. Οι κυριότερες κατηγορίες με τις κωδικές ονομασίες τους και μια σύντομη περιγραφή τους είναι οι ακόλουθες:

- **BLUR**: Διεργασίες για το "θόλωμα" του ηχητικού υλικού (λειτουργούν στο πεδίο του χρόνου)
- **COMBINE**: Διεργασίες για το συνδυασμό φασμάτων διαφορετικού ηχητικού υλικού.
- **DISTORT**: Διεργασίες για την παραμόρφωση του ηχητικού υλικού μέσω wavesets (Wishart 1994: 50).
- **EXTEND**: Διεργασίες για την επιμήκυνση και την κατάτμηση του ηχητικού υλικού.
- **FILTER**: Διεργασίες για το φιλτράρισμα του ηχητικού υλικού.

- **FOCUS**: Διεργασίες για τον εστιασμό στην φασματική πληροφορία του ηχητικού υλικού.
- **GRAIN**: Διεργασίες για την παραγωγή κοκκώδους υφής από το ηχητικό υλικό.
- **MODIFY**: Διεργασίες για πιο γενική τροποποίηση του ηχητικού υλικού.
- **MORPH**: Διεργασίες για τη μεταμόρφωση κατά τη μετάβαση από έναν ήχο σε κάποιον άλλο.
- **REPITCH**: Διεργασίες για τη διαμόρφωση της τονικότητας του ηχητικού υλικού.
- **SPECEDIT**: Βασικές λειτουργίες επέμβασης στην φασματική πληροφορία του ηχητικού υλικού.
- **SPECINFO**: Διεργασίες ανάκτησης περαιτέρω πληροφορίας από την φασματική πληροφορία του ηχητικού υλικού.
- **STRANGE**: Διεργασίες για την παραγωγή ιδιαίτερων σχέσεων μεταξύ των μερικών συχνοτήτων του ηχητικού υλικού.
- **STRETCH**: Διεργασίες για την χρονική ή φασματική επέκταση του ηχητικού υλικού.
- **TEXTURE**: Διεργασίες για την παραγωγή σύνθετων υφών από το ηχητικό υλικό.

2. Το Πρόγραμμα Varibank

Το πρόγραμμα Varibank της κατηγορίας FILTER υλοποιεί μία τράπεζα ζωνοδιαβατών φίλτρων που επιδέχονται δυναμικό στον χρόνο έλεγχο των παραμέτρων της κεντρικής συχνότητας (και ανεξάρτητο για κάθε φίλτρο) και της ποιότητας (παράμετρος Q). Σχετικά με τις μουσικές εφαρμογές του, όπως χαρακτηριστικά αναφέρεται στην τεκμηρίωση της συλλογής CDP:

> ...A tool like FILTER VARIBANK makes it possible gradually to focus complex sonic material into specific pitches, thus creating links with other pitched elements, such as a broader harmonic scheme and the use of acoustic musical instruments... (Endrich and Fraser 2015)

Χρήση του Προγράμματος

Τα αρχεία ελέγχου των παραμέτρων παρέχονται στο πρόγραμμα από τον χρήστη με τη μορφή αρχείων κειμένου `.txt`. Το αρχείο ελέγχου της ποιότητας των φίλτρων περιέχει σε κάθε γραμμή την χρονική στιγμή που συμβαίνει η αλλάγη της παραμέτρου και την τιμή που αυτή λαμβάνει (ένα τυπικό αρχείο από breakpoints) και είναι προαιρετικό. Το αρχείο ελέγχου των κεντρικών συχνοτήτων των φίλτρων είναι πιο σύνθετο από το προηγούμενο και η μορφή του διαφοροποιείται ανάλογα με την εκδοχή του προγράμματος που επιλέγει να εκτελέσει ο χρήστης.

Το πρόγραμμα έχει δύο εκδοχές με τις ονομασίες `varibank` και `varibank2` αντίστοιχα. Η εκδοχή `varibank` αναμένει από το αρχείο να περιέχει σε κάθε γραμμή την χρονική στιγμή καθώς και ζεύγη αποτελούμενα από την τιμή της κεντρικής συχνότητας (είτε σε Hz, είτε σε αρίθμηση νοτών MIDI) και την τιμή του πλάτους της (είτε σε κλίμακα από το 0.001 έως το 10000, είτε σε dB). Επίσης το πλήθος των ζευγών οφείλει να είναι το ίδιο σε όλες τις γραμμές (χρονικές στιγμές). Η εκδοχή `varibank2` αναμένει από το αρχείο να περιέχει δύο περιοχές που διαχωρίζονται μεταξύ τους με μία γραμμή που περιέχει τον χαρακτήρα `#`. Η άνω περιοχή οφείλει να έχει τη μορφή του συμβατού με το `varibank` αρχείου που αναφέρθηκε προηγουμένως. Η κάτω περιοχή σε κάθε γραμμή την χρονική στιγμή και ζεύγη αποτελούμενα από τον αύξοντα αριθμό της μερικής συχνότητας του φάσματος και του πλάτους της (και εδώ απαιτείται τήρηση του ίδιου πλήθους ζευγών σε κάθε γραμμή). Πιο αναλυτικές πληροφορίες για τη μορφή των αρχείων βρίσκονται στην τοποθεσία της τεκμηρίωσής τους (Endrich and Fraser 2016a).

Εφόσον ο χρήστης έχει στη διάθεσή το αρχείο που περιέχει τον ήχο που επιθυμεί να φιλτραριστεί από το πρόγραμμα (`infile`) και το αρχείο ελέγχου των κεντρικών συχνοτήτων (`freqsfile`) μπορεί να εκτελέσει το πρόγραμμα `varibank` ή `varibank2` εισάγοντας στη γραμμή εντολών αντίστοιχα:

```
filter varibank mode infile outfile freqsfile Q gain
```

ή

```
filter varibank2 mode infile outfile freqsfile Q gain
```

Ο όρος `mode` μπορεί να λάβει την τιμή `1` (συχνότητες φίλτρων σε Hz) ή `2` (συχνότητες φίλτρων σε αρίθμηση νοτών MIDI) αναλόγως της μορφής του αρχείου ελέγχου των κεντρικών συχνοτήτων. Ο όρος `outfile` αναφέρεται στο αρχείο που θα εξαχθεί από το πρόγραμμα. Ο όρος `Q` μπορεί να είναι είτε το αρχείο ελέγχου της ποιότητας των φίλτρων είτε μια σταθερά (ποιότητα των φίλτρων σταθερή στον χρόνο). Τέλος, ο όρος `gain` αφορά την συνολική ένταση του εξαχθέντος αρχείου και είναι μια σταθερά. Τόσο για τον όρο `Q` όσο και για τον όρο `gain` το εύρος τιμών είναι το σύνολο [0.001, 10000.0].

Διαθέσιμες Μέθοδοι Παραγωγής του Αρχείου Ελέγχου Κεντρικών Συχνοτήτων

Το αρχείο ελέγχου των κεντρικών συχνοτήτων για τις δύο εκδοχές του προγράμματος Varibank μπορεί να παραχθεί είτε προφανώς με τη δια χειρός συμπλήρωση ενός κενού αρχείου κειμένου σύμφωνα με τους κανόνες που αναφέρθηκαν στην προηγούμενη υποενότητα είτε αυτόματα από άλλα προγράμματα της συλλογής CDP. Συγκεκριμένα, για την εκδοχή `varibank` το αρχείο μπορεί να παραχθεί μέσω του προγράμματος `peak extract` της κατηγορίας SPECINFO, ενώ για την εκδοχή `varibank2` μέσω του προγράμματος `get_partials harmonic` της ίδιας κατηγορίας. Μέσω των προγραμμάτων δημιουργείται ένα φασματικό μοντέλο από το δείγμα της εισόδου τους βάσει του οποίου θα "κινηθούν" τα φίλτρα της τράπεζας Varibank. Με αυτόν τον τρόπο παρέχεται η δυνατότητα φιλτραρίσματος ενός ήχου από το φασματικό μοντέλο ενός άλλου.

Το πρόγραμμα `get_partials harmonic` εξάγει τις εντάσεις των πολλαπλάσιων μιας θεμελίου συχνότητας που υπάρχουν στο φάσμα της εισόδου του. Παράγει αρχείο συμβατό (αν και ελαφρώς παραλλαγμένο σε σχέση με το πρότυπο της προηγούμενης υποενότητας) με το `varibank2` επιτρέποντας στο τελευταίο να εξάγει ικανοποιητικά φιλτραρισμένο υλικό. Εντούτοις, το `get_partials harmonic` αναλύει μόνο ένα στιγμιότυπο του φάσματος της εισόδου του και αντίστοιχα παράγει ένα σταθερού φάσματος αρχείο για το `varibank2` μη αξιοποιώντας με αυτόν τον τρόπο τις επεξεργαστικές δυνατότητες του τελευταίου. Παράλληλα, το `get_partials harmonic` απαιτεί από τον χρήστη να γνωρίζει εκ των προτέρων την τιμή της θεμελίου συχνότητας του στιγμιοτύπου που θα αναλυθεί, που δεν είναι πάντα εφικτό.

Το πρόγραμμα `peak extract` εξάγει τις κορυφές των φασμάτων της εισόδου του. Παράγει αρχείο συμβατό με το `varibank` που δε διασφαλίζει πάντα την εξαγωγή ικανοποιητικά φιλτραρισμένου υλικού από το τελευταίο. Το `peak extract`, σε αντίθεση με το `get_partials harmonic`, αναλύει το σύνολο των επιμέρους φασμάτων της εισόδου του παράγοντας αντίστοιχα ένα δυναμικού φάσματος αρχείο για το `varibank`. Εντούτοις, η ποιότητα του παραγόμενου από το `peak extract` αρχείο εξαρτάται σε μεγάλο βαθμό τόσο από την φύση του φάσματος της εισόδου του όσο και από τις ρυθμίσεις της ανάλυσης αυτού του φάσματος. Αυτό έχει ως αποτέλεσμα τη λήψη από το `varibank` φιλτραρισμένου υλικού που τις περισσότερες φορές εμφανίζει ευκόλως αντιληπτές ασυνέχειες κατά την κίνηση των κεντρικών συχνοτήτων των φίλτρων.

Αξίζει να σημειωθεί ότι η παραγωγή αρχείων για τα `varibank` και `varibank2` από τα `peak extract` και `get_partials harmonic` αντίστοιχα δεν είναι παρά μια μόνο από τις πολλές δυνατότητες των ανωτέρω προγραμμάτων η οποία επιλέγεται από τον χρήστη τους με την χρήση κατάλληλου flag πριν την εκτέλεσή τους. Η πλήρης λίστα δυνατοτήτων των προγραμμάτων αυτών μπορεί να επιθεωρηθεί στην τεκμηρίωσή τους (Endrich and Fraser 2016c, 2016b).

3. Η Κλάση VaribankFileMaker

Για την αποφυγή της δια χειρός παραγωγής του συμβατού με το Varibank αρχείου καθώς και για την επιδίωξη συνέπειας κατά κάποιον αυτόματο τρόπο παραγωγής του, αναζητήθηκαν λύσεις σε εργαλεία εκτός της

συλλογής CDP. Με αυτόν τον τρόπο θα μπορούσαν να συνδυαστούν οι ώριμες και ποιοτικές επεξεργαστικές δυνατότητες της CDP και συγκεκριμένα του Varibank με σύγχρονα εργαλεία προγραμματικής ανάλυσης ηχητικού υλικού. Κατόπιν σχετικής έρευνας αξίζει να αναφερθεί ότι εντοπισμός του προβλήματος και ανάλογη πρόταση για λύση του έχει στο παρελθόν παρουσιαστεί από το project "brkmaker" (Higgins 2016). Το project έχει υλοποιηθεί με τη γλώσσα Pure Data (Puckette 1996) και εστιάζει στην παραγωγή αρχείων συμβατών με τη CDP και με περιεχόμενο του τύπου breakpoint (ανάλογο εκείνου για τον έλεγχο της ποιότητας Q των φίλτρων του Varibank). Ως δευτερεύουσα λειτουργικότητα του project έχει προστεθεί η παραγωγή αρχείων συμβατών με το Varibank, που σύμφωνα με τον δημιουργό του αποτελεί μια "rough" εκδοχή του προγράμματος παραγωγής breakpoint αρχείων. Για την δημιουργία μιας μονάδας λογισμικού με βασική λειτουργικότητα την παραγωγή αρχείων για το Varibank κρίθηκε κατάλληλο το περιβάλλον SuperCollider (McCartney 1996). Η καταλληλότητα του εργαλείου για το σκοπό αυτό προκύπτει λόγω της διαθεσιμότητας υπηρεσιών παραγωγής αρχείων και πολύ περισσότερο για τις παροχές του σε εργαλεία ανάλυσης. Η επιζητούμενη λειτουργικότητα περιλήφθηκε στην κλάση `VaribankFileMaker`. Η κλάση επιλέχθηκε να παράγει αρχείο για την εκδοχή `varibank` ως πιο γενική σε σχέση με την `varibank2`.

Δομή της Κλάσης

Η λειτουργικότητα της κλάσης διαμοιράστηκε σε τρεις μεθόδους: `.loadSoundFile`, `.writeParameterFile` και `.createFilteredSoundFile`.

Η μέθοδος `.loadSoundFile` εξυπηρετεί τον χρήστη στη φόρτωση ενός αρχείου ήχου εμφανίζοντας κατάλληλο διάλογο επιλογής αρχείων. Μέσω καταλλήλων μηνυμάτων διασφαλίζεται η φόρτωση αποκλειστικά και μόνο αρχείων ήχου και το επιλεγμένο αρχείο ήχου αποθηκεύεται σε ένα buffer.

Η μέθοδος `.writeParameterFile` αναλαμβάνει την αναπαραγωγή και ταυτόχρονη ανάλυση του αποθηκευμένου ήχου, και την αρχική δημιουργία και εγγραφή του αρχείου για το Varibank. Τα δύο αυτά ζεύγη δραστηριοτήτων συνυπάρχουν γιατί η εκτέλεση τους οφείλει να είναι παράλληλη. Διαφορετικά, τα δεδομένα της ανάλυσης θα έπρεπε να έχουν αποθηκεύτει σε κάποιον χώρο μνήμης πριν εγγραφούν στο αρχείο. Αναλογιζόμενοι το μέγεθός τους μια τέτοια προσέγγιση αν και εκλεπτυσμένη δεν θα ήταν πρακτική. Για τον λόγο αυτό, η λειτουργικότητα της μεθόδου διαμοιράζεται σε τρεις μεθόδους που θα εκτελεστούν παράλληλα: `.runFFTSynth`, `.makefilterBankParamFile` και `.runWriteRoutine`.

Η μέθοδος `.runFFTSynth` ορίζει και εκτελεί ένα SynthDef με τις ευθύνες της αναπαραγωγής και ανάλυσης του αποθηκευμένου ήχου.

Η μέθοδος `.makefilterBankParamFile` δημιουργεί το αρχείο για το Varibank προς εγγραφή.

Η μέθοδος `.runWriteRoutine` ορίζει και εκτελεί μια Routine για την εγγραφή του αρχείου για το Varibank. Ως κατηγόρημα δέχεται τον χρόνο για τον οποίο ο ήχος που θα επεξεργαστεί το Varibank θα μείνει ως έχει (μετρώντας από την αρχή του). Ο χρόνος αυτός εγγράφεται ως η αρχική πληροφορία στο αρχείο και κατόπιν εγγράφονται σε αυτό τα δεδομένα της ανάλυσης. Ο τρόπος με τον οποία τα δεδομένα διαχειρίζονται και εγγράφονται ανατίθεται στη μέθοδο `.get1AnalysisFrameContent`.

Η μέθοδος `.get1AnalysisFrameContent` καλείται από την `.runWriteRoutine` ανά τακτά χρονικά διαστήματα που καθορίζονται από το πλαίσιο (frame) της ανάλυσης. Όταν γίνονται διαθέσιμα τα δεδομένα από το buffer της ανάλυσης μεταφέρονται από το επίπεδο του SCSynth στο επίπεδο της SCLang μέσω του μηνύματος `.loadToFloatArray` στον buffer. Κατόπιν λαμβάνεται η πληροφορία πλάτους (magnitude) των φασματικών δειγμάτων και εγγράφεται στο αρχείο για το Varibank.

Τέλος, η μέθοδος `.createFilteredSoundFile` εξυπηρετεί τον χρήστη στην επιλογή αρχείου προς επεξεργασία από το Varibank. Μέσω διαλόγου διασφαλίζεται ότι πρόκειται για αρχείο ήχου και στο directory του εκτελείται το Varibank εξάγοντας εκεί το επεξεργασμένο αρχείο ήχου.

Παράδειγμα Χρήσης

Ακολουθεί ένα παράδειγμα ενδεικτικής χρήσης της κλάσης `VaribankFileMaker` στο περιβάλλον SuperCollider:

```
//Δημιουργία ενός νέου αντικειμένου
a = VaribankFileMaker.new;
```

```
//Επιλογή αρχείου ήχου προς ανάλυση
a.loadSoundFile;
```

```
/*
Έναρξη της ανάλυσης και εγγραφή του
συμβατού με το VARIBANK αρχείου
*/
a.writeParameterFile;
```

```
/*
Εναλλακτικά, μπορεί να επιλεχθεί
χρονική μετατόπιση (σε δευτερόλεπτα)
για την εγγραφή των δεδομένων της ανάλυσης
*/
a.writeParameterFile(offset: 0.5);
```

```
/*
Επιπλέον, μπορεί να επιλεχθεί
συγκεκριμένο όνομα για το αρχείο
(προεπιλεγμένο είναι το ίδιο
όνομα με αυτό του αρχείου ήχου)
*/
a.writeParameterFile(name: "myBankOfFilters");
/*
Αν το αρχείο με αυτό το όνομα
προυπάρχει, το παλαίο αρχείο
αντικαθίσταται από το νέο
*/
```

```
/*
Επιλογή του αρχείου ήχου προς
επεξεργασία από το VARIBANK
*/
a.createFilteredSoundFile(quality: 500, volume: 100);
```

```
/*
Και εδώ, μπορεί να επιλεχθεί συγκεκριμένο
όνομα για το αρχείο που θα εξαχθεί από το
VARIBANK (προεπιλεγμένο είναι το όνομα του
αρχείου ήχου με την κατάληξη '_filtered')
*/
a.createFilteredSoundFile(quality: 500, volume: 100, name: "myFilteredSound");
```

Αποτελέσματα Χρήσης

Η χρήση της κλάσης VaribankFileMaker απέφερε ικανοποιητικά αποτελέσματα εφόσον εξυπηρέτησε πλήρως

τις ανάγκες της επεξεργασίας ηχητικού υλικού για την παράλληλη της παρούσας εργασία στη "Σύνθεση Ηλεκτροακουστικής Μουσικής" (Apergis 2019).

Αναφορές

Apergis, A. 2019. *Άνοιγμα, Προς Το*. Audio File. https://soundcloud.com/andreas-apergis/ql34psamh7mu (προσπελάστηκε την 1η Απριλίου 2019)

Chabade, J. 1997. *Electric sound: the past and promise of electronic music*. Upper Saddle River, N.J.: Prentice Hall

Endrich, A., and R. Fraser. 2015.*FILTER VARIBANK and VARIBANK2 – User-defined time-varying filterbank, with time-varying Q*. http://www.ensemble-software.net/CDPDocs/html/cgrofilt.htm#VARIBANK (προσπελάστηκε την 1η Απριλίου 2019)

Endrich, A., and R. Fraser. 2016a.*Datafile for FILTER VARIBANK or VARIBANK2* http://www.ensemble-software.net/CDPDocs/html/filestxt.htm#FILTERVFILES (προσπελάστηκε την 1η Απριλίου 2019)

Endrich, A., and R. Fraser. 2016b.*GET_PARTIALS HARMONIC*. http://www.ensemble-software.net/CDPDocs/html/cspecinf.htm#GETPARTIALS (προσπελάστηκε την 1η Απριλίου 2019)

Endrich, A., and R. Fraser. 2016c.*PEAK EXTRACT*. http://www.ensemble-software.net/CDPDocs/html/cspecinf.htm#PEAKEXTRACT (προσπελάστηκε την 1η Απριλίου 2019)

Higgins, J. 2016. *brkmaker*. Source Code Repository. https://github.com/j-p-higgins/brkmaker (προσπελάστηκε την 1η Απριλίου 2019)

McCartney, J. 1996. SuperCollider: A New Real-Time Synthesis Language. In *Proceedings of the International Computer Music Conference, Hong Kong, August 19-24, 1996*, 257-58. San Francisco, CA: International Computer Music Association

Puckette, M. 1996. Pure Data: Another Integrated Computer Music Environment. In *Proceedings of the International Computer Music Conference, Hong Kong, August 19-24, 1996*, 224-27. San Francisco, CA: International Computer Music Association

Wishart, T. 1994. *Audible design: a plain and easy introduction to practical sound composition*. York, UK: Orpheus the Pantomime.

Παράρτημα Α. (Πηγαίος Κώδικας)

```
/*
VaribankFileMaker:
Analyse a sound file and output a text file compatible with program VariBank of
the 'CDP' suite
Usage:
1. loadSoundFile : Select file to analyze throug a dialog
2. writeParameterFile : Analyse the file and write parameters for analysis in
text file.
3. createFilteredSoundFile : Create sound file by invoking Varibank through
command line (needs a local CDP installation)
*/
VaribankFileMaker {
    var sampleFilterModelFilePath, filterBankParamFilePath,
  sampleInputFilePath, sampleOutputFilePath;
    var buffer, fftBuffer, file, fftSynth;
    loadSoundFile {
```

```
    FileDialog (
        { arg path;
            sampleFilterModelFilePath = PathName(path[0]);
            if (sampleFilterModelFilePath.extension == "wav",
                {
                    this.loadSampleBuffer;
                },
                {
                    sampleFilterModelFilePath = nil;
                    postln( "Please select a soundfile next time" );
                }
            );
        },
        {
            postln("Made no selection");
            if (sampleFilterModelFilePath.notNil && buffer.notNil,
                {postln(", but" + sampleFilterModelFilePath.fileName +
      "is still loaded")}
            );
        }
    )
}
loadSampleBuffer {
    buffer !? { buffer.clear };
    buffer = Buffer.read(Server.default, sampleFilterModelFilePath.fullPath);
    postln("SoundFile" + sampleFilterModelFilePath.fileName +
"was selected (to filter another with)");
}
makefilterBankParamFile { arg fileName;
    filterBankParamFilePath =
PathName.new(sampleFilterModelFilePath.pathOnly ++ fileName ++ ".txt");
    if (File.exists(filterBankParamFilePath.fullPath),
{File.delete(filterBankParamFilePath.fullPath)});
    file = File(filterBankParamFilePath.fullPath, "w");
}
runFFTSynth {
    fftBuffer = Buffer.alloc(Server.default, 2048, 1);
    fftSynth = {
        var input, chain;
        input = PlayBuf.ar(buffer.numChannels, buffer, BufRateScale.kr(buffer));
        chain = FFT(fftBuffer, input, hop: 1/4, wintype: 0, winsize:
    (fftBuffer.numFrames/2).asInt);
        //chain = PV_MagSquared(chain);
    }.play;
}
get1AnalysisFrameContent {arg maxFreqNum, scalingCoef;
    var content = "";
    fftBuffer.loadToFloatArray
    (
        action:
        {arg currentFrame;
            var z, x;
            var magnitudes;
            z = currentFrame.clump(2).flop;
            z = [Signal.newFrom(z[0]), Signal.newFrom(z[1])];
            x = Complex(z[0], z[1]);
            magnitudes = x.magnitude[1..maxFreqNum]/scalingCoef;
            magnitudes.do
            {arg item, index;
                var ampInDB = item.ampdb;
                var threshold = -60;
                if ( ampInDB < threshold,
                    {ampInDB = -inf},
                    {ampInDB = ampInDB + 20}
                );
```

```
                content = content +
      (index+1*buffer.sampleRate/fftBuffer.numFrames).asString;
                content = content + ampInDB ++ "dB";
            };
            ^content;
        }
    );
}
runWriteRoutine{arg initialTimeStamp;
    {
        var content = "";
        var timeStamp = initialTimeStamp;
        var fftSize = fftBuffer.numFrames;
        var fftSizeOver2 = (fftSize/2).asInt;
        var numFreqs = fftSizeOver2-1;
        var rate = 2*fftSize/buffer.sampleRate;
        var numFrames =
  ((buffer.numFrames / buffer.sampleRate) / rate).ceil.asInt;
        "Please, bear with me...".postln;
        if (timeStamp != 0.0,
            {
                content = "0.0";
                numFreqs.do
                {
                    arg i;
                    content = content + (i+1*buffer.sampleRate/fftSize).asString;
                    content = content + "-infdB";
                };
                file.write(content);
                content = "\n";
            }
        );
        numFrames.do
        {
            content =  content ++ timeStamp;
            content = content +
    this.get1AnalysisFrameContent(numFreqs, fftSizeOver2);
            content = content + "\n";
            file.write(content);
            content ="\n";
            timeStamp = timeStamp + rate;
        };
        //wait a little more
        0.1.wait;
        file.close;
        fftSynth.free;
        fftBuffer.free;
        "See? I'm done!".postln;
    }.fork;
}
writeParameterFile { arg offset=0.0, name="variTextFile";
    if (sampleFilterModelFilePath.notNil,
        {
            var fileName;
            if (name == "variTextFile",
                {fileName = sampleFilterModelFilePath.fileNameWithoutExtension},
                {fileName = name}
            );
            this.makefilterBankParamFile(fileName);
            this.runFFTSynth;
            this.runWriteRoutine(offset);
        },
        { postln("Please, select a soundfile to be analyzed")}
    );
}
createFilteredSoundFile { arg quality=5, volume=1, name="variFiltered";
```

```
    createFilteredSoundFile { arg quality=3, volume=1, name="variFiltered";
        if (filterBankParamFilePath.notNil,
            {
                FileDialog (
                    { arg path;
                        sampleInputFilePath = PathName(path[0]);
                        if (sampleInputFilePath.extension == "wav",
                            {
                                var fileName;
                                var command = "cd" +
                sampleInputFilePath.pathOnly.escapeChar($ ) ++
                "; filter varibank 1" + sampleInputFilePath.fileName;
                                postln(sampleInputFilePath.fileName +
                  "was selected (to be filtered)");
                                if (name == "variFiltered",
                                    {fileName =
                    sampleInputFilePath.fileNameWithoutExtension ++
                    "_filtered"},
                                    {fileName = name}
                                );
                                sampleOutputFilePath =
                PathName.new(sampleInputFilePath.pathOnly ++
                  fileName ++ ".wav");
                                if (File.exists(sampleOutputFilePath.fullPath),
                {File.delete(sampleOutputFilePath.fullPath)});
                                command = command + sampleOutputFilePath.fileName +
                filterBankParamFilePath.fullPath.escapeChar($ ) +
                quality + volume;
                                command.runInTerminal;
                            },
                            {sampleInputFilePath = nil;
                postln( "Please select a (.wav) soundfile next time" )}
                        );
                    },
                    {
                        postln("Made no selection");
                        if (sampleInputFilePath.notNil,
                            {postln("," + sampleInputFilePath.fileName +
              "is your previously filtered sound")}
                        );
                    }
                )
            },
            { postln("Please, first analyze a sound")}
        );
    }
}
```