



UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

# UIT-T

SECTEUR DE LA NORMALISATION  
DES TÉLÉCOMMUNICATIONS  
DE L'UIT

# G.711

**Appendice I**  
(09/99)

SÉRIE G: SYSTÈMES ET SUPPORTS DE  
TRANSMISSION, SYSTÈMES ET RÉSEAUX  
NUMÉRIQUES

Systèmes de transmission numériques – Equipements  
terminaux – Codage des signaux analogiques en  
modulation par impulsions et codage

---

Modulation par impulsions et codage des  
fréquences vocales

**Appendice I: Algorithme simple de haute qualité  
pour le masquage des pertes de paquets en  
codage G.711**

Recommandation UIT-T G.711 – Appendice I

(Antérieurement Recommandation du CCITT)

---

RECOMMANDATIONS UIT-T DE LA SÉRIE G

**SYSTÈMES ET SUPPORTS DE TRANSMISSION, SYSTÈMES ET RÉSEAUX NUMÉRIQUES**

|                                                                                                                                                           |                    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|
| CONNEXIONS ET CIRCUITS TÉLÉPHONIQUES INTERNATIONAUX                                                                                                       | G.100–G.199        |
| <b>SYSTÈMES INTERNATIONAUX ANALOGIQUES À COURANTS PORTEURS</b>                                                                                            |                    |
| CARACTÉRISTIQUES GÉNÉRALES COMMUNES À TOUS LES SYSTÈMES ANALOGIQUES À COURANTS PORTEURS                                                                   | G.200–G.299        |
| CARACTÉRISTIQUES INDIVIDUELLES DES SYSTÈMES TÉLÉPHONIQUES INTERNATIONAUX À COURANTS PORTEURS SUR LIGNES MÉTALLIQUES                                       | G.300–G.399        |
| CARACTÉRISTIQUES GÉNÉRALES DES SYSTÈMES TÉLÉPHONIQUES INTERNATIONAUX HERTZIENS OU À SATELLITES ET INTERCONNEXION AVEC LES SYSTÈMES SUR LIGNES MÉTALLIQUES | G.400–G.449        |
| COORDINATION DE LA RADIOTÉLÉPHONIE ET DE LA TÉLÉPHONIE SUR LIGNES                                                                                         | G.450–G.499        |
| <b>EQUIPEMENTS DE TEST</b>                                                                                                                                |                    |
| <b>CARACTÉRISTIQUES DES SUPPORTS DE TRANSMISSION</b>                                                                                                      |                    |
| <b>SYSTÈMES DE TRANSMISSION NUMÉRIQUES</b>                                                                                                                |                    |
| EQUIPEMENTS TERMINAUX                                                                                                                                     | G.700–G.799        |
| Généralités                                                                                                                                               | G.700–G.709        |
| <b>Codage des signaux analogiques en modulation par impulsions et codage</b>                                                                              | <b>G.710–G.719</b> |
| Codage des signaux analogiques par des méthodes autres que la MIC                                                                                         | G.720–G.729        |
| Principales caractéristiques des équipements de multiplexage primaires                                                                                    | G.730–G.739        |
| Principales caractéristiques des équipements de multiplexage de deuxième ordre                                                                            | G.740–G.749        |
| Caractéristiques principales des équipements de multiplexage d'ordre plus élevé                                                                           | G.750–G.759        |
| Caractéristiques principales des équipements de transcodage et de multiplication numérique                                                                | G.760–G.769        |
| Fonctionnalités de gestion, d'exploitation et de maintenance des équipements de transmission                                                              | G.770–G.779        |
| Caractéristiques principales des équipements de multiplexage en hiérarchie numérique synchrone                                                            | G.780–G.789        |
| Autres équipements terminaux                                                                                                                              | G.790–G.799        |
| RÉSEAUX NUMÉRIQUES                                                                                                                                        | G.800–G.899        |
| SECTIONS NUMÉRIQUES ET SYSTÈMES DE LIGNES NUMÉRIQUES                                                                                                      | G.900–G.999        |

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

**MODULATION PAR IMPULSIONS ET CODAGE DES FRÉQUENCES VOCALES**

**APPENDICE I**

**Algorithme simple de haute qualité pour le masquage  
des pertes de paquets en codage G.711**

**Résumé**

Les algorithmes de masquage des pertes de paquets (PLC, *packet loss concealment*), également appelés algorithmes de masquage d'effacement de trame, masquent les pertes de transmission d'un système audio dont le signal d'entrée est codé et mis en paquets par un émetteur, envoyé sur le réseau et reçu au récepteur qui décode les paquets et restitue leur contenu. Beaucoup de codeurs de la parole normalisés de type CELP ont des algorithmes PLC intégrés dans leur norme. L'algorithme décrit ci-après propose une méthode de masquage applicable à la Recommandation G.711.

**Source**

L'Appendice I à la Recommandation UIT-T G.711, élaboré par la Commission d'études 16 (1997-2000) de l'UIT-T, a été approuvé le 30 septembre 1999 selon la procédure définie dans la Résolution n° 1 de la CMNT.

## AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

La Conférence mondiale de normalisation des télécommunications (CMNT), qui se réunit tous les quatre ans, détermine les thèmes d'études à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution n° 1 de la CMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

## NOTE

Dans la présente Recommandation, le terme *exploitation reconnue (ER)* désigne tout particulier, toute entreprise, toute société ou tout organisme public qui exploite un service de correspondance publique. Les termes *Administration*, *ER* et *correspondance publique* sont définis dans la *Constitution de l'UIT* (Genève, 1992).

## DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

© UIT 2000

Droits de reproduction réservés. Aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'UIT.

## TABLE DES MATIÈRES

|                                                                                                               | <b>Page</b> |
|---------------------------------------------------------------------------------------------------------------|-------------|
| Appendice I – Algorithme simple de haute qualité pour le masquage des pertes de paquets en codage G.711 ..... | 1           |
| I.1 Introduction.....                                                                                         | 1           |
| I.2 Description de l'algorithme.....                                                                          | 1           |
| I.2.1 Trames correctes.....                                                                                   | 1           |
| I.2.2 Première trame erronée.....                                                                             | 2           |
| I.2.3 Détection de la tonie .....                                                                             | 2           |
| I.2.4 Génération du signal synthétique pour les dix premières millisecondes.....                              | 2           |
| I.2.5 Génération du signal synthétique après 10 millisecondes .....                                           | 3           |
| I.2.6 Atténuation .....                                                                                       | 3           |
| I.2.7 Première trame correcte suivant un effacement .....                                                     | 3           |
| I.2.8 Exemple .....                                                                                           | 4           |
| I.3 Description de l'algorithme avec code C++ annoté .....                                                    | 6           |
| I.3.1 Définition "typedef" et constantes.....                                                                 | 6           |
| I.3.2 Déclaration de classe .....                                                                             | 6           |
| I.3.3 Boucle principale.....                                                                                  | 8           |
| I.3.4 Fonctions membre utilitaire.....                                                                        | 8           |
| I.3.5 Constructeur.....                                                                                       | 9           |
| I.3.6 Addtohistory et savespeech .....                                                                        | 10          |
| I.3.7 Dofe .....                                                                                              | 11          |
| I.3.8 Détection de la tonie .....                                                                             | 14          |
| I.3.9 Production et atténuation du signal synthétique.....                                                    | 16          |
| I.3.10 Opérateurs de superposition/addition OLA.....                                                          | 17          |
| I.4 Complexité et délai .....                                                                                 | 19          |



## **MODULATION PAR IMPULSIONS ET CODAGE DES FRÉQUENCES VOCALES**

### **APPENDICE I**

#### **Algorithme simple de haute qualité pour le masquage des pertes de paquets en codage G.711**

*(Genève, 1999)*

#### **I.1 Introduction**

Les algorithmes de masquage des pertes de paquets (PLC, *packet loss concealment*), également appelés algorithmes de masquage d'effacement de trame, masquent les pertes de transmission d'un système audio dont le signal d'entrée est codé et mis en paquets par un émetteur, envoyé sur le réseau et reçu au récepteur qui décode les paquets et restitue leur contenu. Beaucoup de codeurs de la parole normalisés de type CELP tels que cités dans les Recommandations G.723.1 [1], G.728 [2], et G.729 [3] ont des algorithmes PLC intégrés dans leur norme. L'algorithme décrit ci-après propose une méthode de masquage applicable au codage G.711.

Le masquage PLC consiste à générer un signal vocal synthétique destiné à remplacer les données manquantes (effacées) d'un train binaire. Ce signal synthétique aura de préférence le même timbre et les mêmes caractéristiques spectrales que le signal manquant et ne produira pas de défauts. Etant donné que les signaux vocaux ont souvent une légère persistance, il est possible de se référer à l'historique des signaux précédents pour produire un son synthétique raisonnablement proche de celui correspondant au segment manquant. Si l'effacement n'est pas trop long et s'il ne survient pas dans une zone comportant des variations rapides du signal, le masquage peut le rendre imperceptible.

#### **I.2 Description de l'algorithme**

L'adjonction d'un algorithme PLC à un système G.711 dépourvu de masquage des pertes de paquets ne requiert des modifications qu'au niveau du récepteur. Les données audio codées G.711 sont échantillonnées à 8 kHz, et dans le présent appendice, on prend pour hypothèse un découpage en trames de 10 ms (80 échantillons). Une intervention sur quelques paramètres permet d'appliquer l'algorithme à d'autres tailles de paquet et taux d'échantillonnage.

##### **I.2.1 Trames correctes**

En fonctionnement normal (transmission de trames ou de paquets corrects) le récepteur décode le paquet reçu et envoie le résultat à la sortie audio. Deux modifications mineures interviennent au niveau du récepteur lorsqu'il traite des trames correctes en vue de réaliser le masquage PLC:

- 1) il enregistre une copie du résultat décodé dans un tampon historique de 48,75 ms (390 échantillons). Ce tampon sert à calculer la période fondamentale du moment et à extraire des signaux persistant un effacement. Cette mise en tampon n'introduit aucun retard dans le signal de sortie;
- 2) le signal restitué est retardé de 3,75 ms (30 échantillons) avant d'être envoyé à la sortie audio. Ce retard introduit par l'algorithme permet de faire, au début d'un effacement, une superposition/admission (OLA, *overlap add*), assurant une transition douce entre les signaux réels et les signaux synthétiques.

### **I.2.2 Première trame erronée**

Au début de l'effacement, le tampon historique circulaire est copié dans un tampon non circulaire, appelé le tampon fondamental, plus simple d'emploi. Le contenu de ce tampon est utilisé pendant toute la durée de l'effacement. Une copie additionnelle du quart de période fondamentale la plus récente, appelée le tampon *lastq*, est prévue pour le cas où l'effacement aurait une durée supérieure à 10 ms.

### **I.2.3 Détection de la tonie**

En premier lieu on évalue la période fondamentale en déterminant la crête de l'intercorrélation normalisée entre les 20 ms les plus récentes des sons vocaux dans le tampon historique et les sons vocaux précédents au moyen de prélèvements de 5 (40 échantillons) à 15 ms (120 échantillons). Cela correspond à des fréquences de 200 à 66 Hz. L'étendue tonale a été choisie sur la base de celle utilisée pour le postfiltre du codage G.728. Alors que ce dernier utilise une limite inférieure de 2,5 ms (20 échantillons), cette limite est portée dans le cas présent à 40 échantillons afin que la même étendue tonale ne soit pas répétée plus de deux fois dans une même trame effacée de 10 ms. Pour simplifier l'opération, la tonie est évaluée en deux phases, par une recherche approximative sur un signal sous-échantillonné au rapport 2:1, ensuite par une recherche précise au voisinage de la crête obtenue par la recherche approximative. On peut diminuer la complexité au prix d'une légère dégradation de la qualité en omettant cette recherche précise. Dans le texte qui suit, on utilise le terme *longueur d'onde* notamment pour désigner la valeur obtenue par ce calcul étant donné que le signal manquant peut correspondre à de la parole voisée ou de la parole non voisée.

La superposition/addition par déplacement du signal (WSOLA, *waveform shift overlap add*) a montré que la fonction d'intercorrélation normalisée peut être remplacée par une autre, non normalisée, ou par une fonction de différence des grandeurs entre moyennes (AMDF, *average magnitude difference function*) donnant des résultats analogues au niveau de la qualité de fonctionnement global.

### **I.2.4 Génération du signal synthétique pour les dix premières millisecondes**

Pour les 10 premières millisecondes de l'effacement, on obtient les meilleurs résultats en générant un signal synthétique sur la base de la dernière période fondamentale non atténuée. Seule la partie correspondant à 1,25 période fondamentale la plus récente du tampon tonal est utilisée au cours de ces dix premières millisecondes. On obtient une transition douce entre le signal réel et le signal synthétique, ou lorsque la période fondamentale est répétée à plusieurs reprises, par une superposition/addition (OLA) au moyen d'une fenêtre triangulaire d'un quart de période fondamentale, entre la dernière et l'avant-dernière périodes. Pour un quart de longueur d'onde, le signal commençant à 1,25 période fondamentale de la fin du tampon tonal est multiplié par une rampe montante et ensuite ajouté au dernier quart de période fondamentale dans la mémoire tampon *lastq* multiplié par une rampe descendante. Si une complexité plus grande n'est pas un obstacle, les fenêtres triangulaires peuvent être remplacées par des fenêtres de Hanning dans toutes les opérations OLA.

Le résultat de l'opération OLA remplace la partie terminale du tampon tonal et celle du tampon historique. Il est également reproduit par le récepteur au cours de la partie terminale de la dernière trame correcte en remplacement du signal original. Ceci introduit le retard dû à l'algorithme, la partie terminale de la dernière trame ne pouvant être restituée qu'au moment de savoir si la trame suivante est effacée. S'il se produit un effacement, le signal de la partie terminale de la dernière trame correcte est modifié par une opération OLA pour que le passage au signal synthétique ait lieu en douceur.

Pour générer le signal synthétique destiné aux 10 premières millisecondes de l'effacement, on place un indicateur à une période fondamentale derrière l'extrémité du tampon tonal, pour ensuite envoyer des copies des échantillons vers la sortie. Si la période fondamentale est inférieure à 10 ms l'indicateur, lorsqu'il quitte l'extrémité de la période fondamentale, recule d'une période exactement



avant la poursuite de l'opération. Si la période fondamentale est courte (la fréquence est élevée), la dernière période fondamentale contenue dans le tampon tonal est répétée à plusieurs reprises au cours de l'effacement de 10 ms.

A mesure que l'effacement se poursuit, le tampon historique est mis à jour au moyen du signal synthétique sortant. Ainsi, le tampon historique contient toujours un signal progressif, continu. Cette continuité est importante lorsque une séquence se présente dans l'ordre "trame erronée, trame correcte, trame erronée".

### **I.2.5 Génération du signal synthétique après 10 millisecondes**

Si la trame suivante est également perdue, l'effacement durera au moins 20 ms et il convient d'y remédier. Si la répétition d'une même période fondamentale donne de bons résultats pour les effacements de courte durée (10 ms par exemple), elle introduit dans les effacements plus longs des défauts harmoniques (bip), qui sont surtout perceptibles si l'effacement se termine dans une zone non voisée du signal vocal ou dans une zone de transition rapide telle qu'un arrêt. Des expériences ont montré que ces défauts sont considérablement réduits par l'augmentation du nombre de périodes fondamentales utilisées pour synthétiser le signal à mesure que l'effacement se poursuit. Le fait d'utiliser davantage de périodes fondamentales augmente la variation dans le signal. Même si les périodes fondamentales ne sont pas restituées dans l'ordre dans lequel elles surviennent dans le signal original, le résultat obtenu sera toujours naturel. A 10 ms dans l'effacement, le nombre de périodes fondamentales utilisées pour synthétiser la parole est porté à deux, et à 20 ms une troisième est ajoutée. Pour les effacements durant plus de 20 ms, aucune autre modification n'est apportée au tampon tonal.

Lorsque le nombre de périodes fondamentales utilisées dans le tampon tonal augmente, il importe que les transitions dans le signal synthétique soient douces. On obtient cela en poursuivant la restitution du tampon tonal existant pendant un quart de période fondamentale au début de la deuxième et de la troisième trame effacée, ce qui met à jour le tampon tonal et maintient l'indicateur de tampon synchronisé avec la phase correcte, pour ensuite effectuer une superposition/addition (OLA) au moyen de la production d'un nouveau tampon tonal.

Le tampon tonal est mis à jour exactement comme cela s'est passé au cours de la première trame effacée, à cela près que le nombre de périodes fondamentales est augmenté. Par exemple, au départ de la deuxième trame effacée, le signal commençant à 2,25 périodes fondamentales de la fin du tampon tonal est multiplié, pendant un quart de longueur d'onde, par une rampe montante pour être ensuite ajouté au quart de longueur d'onde dans le tampon `lastq` multiplié par une rampe descendante. Le résultat de l'opération OLA remplace le dernier quart de longueur d'onde du tampon tonal. Pour conserver la phase de l'indicateur de sortie en vigueur, on soustrait les périodes fondamentales de l'indicateur jusqu'à qu'il corresponde à la première période fondamentale utilisée.

### **I.2.6 Atténuation**

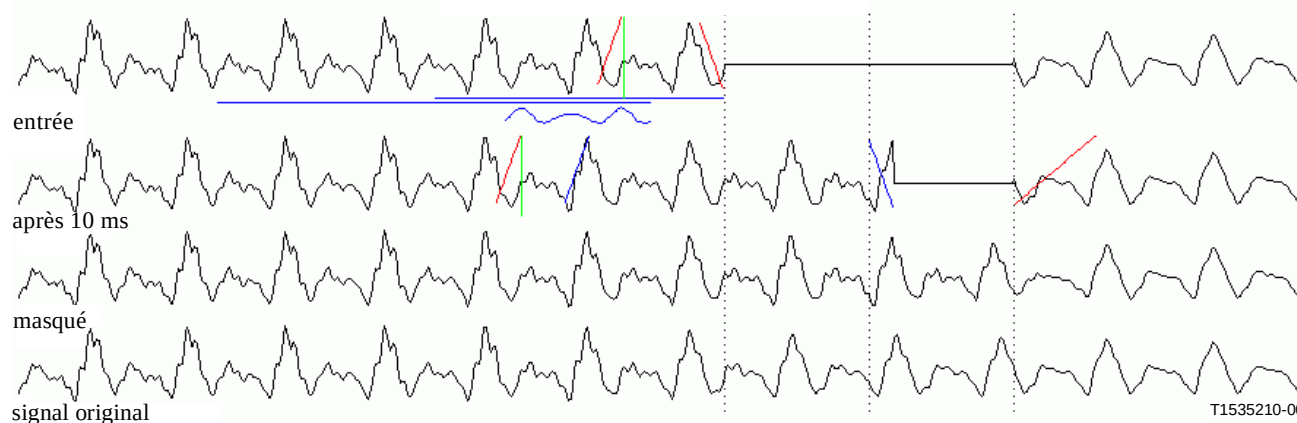
Comme pour d'autres algorithmes PLC, ceux des Recommandations G.729 et G.728 Annexe I par exemple, il convient, en cas d'effacement important, d'atténuer le signal à mesure que l'effacement se poursuit. Lorsque l'effacement se prolonge, le signal synthétique est davantage susceptible de s'écarter du signal réel. L'absence d'atténuation a pour effet de créer des défauts étranges résultant du maintien prolongé de certains types de son, même si le segment du signal synthétique paraît naturel lorsqu'on l'écoute isolément. Le signal n'est pas atténué pendant les 10 premières millisecondes d'un effacement. Au début des 10 ms suivantes, le signal synthétique est atténué linéairement au moyen d'une rampe au taux de 20% par 10 ms. Après 60 ms, le signal synthétique est nul.

### **I.2.7 Première trame correcte suivant un effacement**

A l'arrivée de la première trame correcte suivant un effacement, il faut une transition douce entre le signal synthétique et le signal réel. A cet effet, il convient de prolonger le signal synthétique du tampon tonal au-delà de la fin de l'effacement pour le mixer avec le signal réel au moyen d'une

opération OLA. La longueur de cette dernière dépend à la fois de la période fondamentale et de la longueur de l'effacement. Pour les effacements de courte durée, de 10 ms par exemple, on utilise une fenêtre d'un quart de longueur d'onde. Pour les effacements plus longs, cette fenêtre est prolongée de 4 ms par 10 ms d'effacement jusqu'au maximum de la trame, soit 10 ms.

### I.2.8 Exemple



**Figure I.1/G.711 – Algorithme de masquage de perte de trame pour codage G.711**

La Figure I.1 représente graphiquement le comportement de l'algorithme dans le cas d'un effacement de 20 ms (c'est-à-dire deux trames) sur un segment de parole voisée d'une voix masculine. La courbe supérieure représente le signal d'entrée. L'effacement est délimité par les trois traits verticaux pointillés à 10 ms d'intervalle. La parole qui précède l'effacement sur la courbe "entrée" est le contenu du tampon historique au moment où l'effacement commence. La partie du signal qui suit l'effacement représente 20 ms de parole non altérée.

Sous le signal "entrée" on trouve deux traits horizontaux qui représentent les fenêtres utilisées pour la détection de la tonie au moyen de la corrélation croisée normalisée. Le trait court supérieur montre les 20 dernières millisecondes de parole précédant l'effacement; il s'agit du signal de référence. Le trait inférieur représente la fenêtre de 20 ms qui recule par étapes de 40 à 120 échantillons. La petite courbe au-dessous du trait horizontal est le résultat de l'intercorrélation normalisée. La crête de cette courbe, représentée par le trait vertical, est l'estimation de la tonie. Ce trait vertical se trouve à une période fondamentale avant le début de l'effacement.

Le contenu du quart de longueur d'onde qui précède l'effacement est enregistré dans le tampon `lastq` pour usage ultérieur. Pour créer le tampon tonal, le quart de longueur d'onde qui précède l'effacement subit une opération OLA au moyen d'une fenêtre triangulaire au quart de longueur d'onde de la période fondamentale précédente. La pondération et l'emplacement des fenêtres sont représentés par les deux rampes du signal "entrée". Le résultat produit par l'opération OLA remplace le quart de longueur d'onde du signal précédant l'effacement. Pour les 10 premières millisecondes de l'effacement, le signal synthétique est généré par répétition du tampon comportant une seule période fondamentale, par exemple la zone commençant au trait vertical, à la crête de l'intercorrélation, et se terminant à la fin de l'effacement, autant de fois qu'il le faut.

Les résultats de ce qui précède sont montrés dans la deuxième courbe, "après 10 ms". Dans celle-ci, le signal synthétique est prolongé d'un quart de longueur d'onde au-delà de la limite correspondant aux 10 premières millisecondes de l'effacement étant donné que cela est nécessaire au cours de l'opération OLA. Le décalage dans le tampon à période fondamentale unique, à la fin des 10 ms, est sauvegardé dans une variable appelée `poffset`. Si l'effacement se termine après 10 ms, il suffit de

superposer (OLA) ce quart de longueur d'onde au signal d'entrée pour terminer l'algorithme de masquage. Dans le cas présent, l'effacement dure plus de 10 ms, raison pour laquelle la période fondamentale est allongée afin d'accroître la variation du signal.

La variation du signal est augmentée par l'adjonction d'une autre période fondamentale au tampon tonal. Un trait vertical a été tracé sur la courbe à deux périodes fondamentales précédant le début de l'effacement. Le quart de longueur d'onde précédant ce trait, représenté par la rampe montante, subit une opération OLA au moyen du tampon `lastq` (la zone au-dessous de la rampe descendante et la courbe d'entrée) et placée dans le tampon tonal. Pour la deuxième période de 10 ms, le tampon tonal est donc la zone comprise entre la fin de la première rampe montante et le début de l'effacement dans la courbe "après 10 ms". Le quart de longueur d'onde qui précède immédiatement l'effacement est différent dans le cas du tampon tonal de ce que montre sur la courbe étant donné que celle-ci a fait l'objet d'une opération OLA.

La transition douce entre les tampons à simple et à double période fondamentale nécessite une opération OLA pour un quart de longueur d'onde au début de la deuxième période de 10 ms de l'effacement. La zone se trouvant sous la rampe descendante, à 10 ms à l'intérieur de l'effacement et combinée à la zone se trouvant sous la rampe montante, près de la première crête de signal dans le tampon à deux périodes fondamentales et opération OLA. Le résultat prend la place de la zone se trouvant sous la rampe descendante. L'emplacement de la rampe montante est calculé à partir de l'indicateur `poffset` enregistré en ôtant des périodes fondamentales jusqu'à ce que l'indicateur tonal se trouve dans la première longueur d'onde de la partie du tampon tonal utilisée à ce moment. Cela garantit que la phase correcte de l'onde est maintenue alors que le nombre de périodes dans le tampon tonal augmente. Pendant la durée de la deuxième trame de 10 ms effacée, l'onde synthétique est produite par simple copie du signal se trouvant dans le tampon tonal.

Comme pour les 10 premières millisecondes, l'onde synthétique s'étend au-delà de la limite des 10 ms suivantes en vue de l'opération OLA avec le segment suivant. Si l'effacement se poursuit, un quart de longueur d'onde suffit et le nombre de périodes dans le tampon tonal est à nouveau augmenté. Si l'effacement se termine, la longueur de la fenêtre de l'opération OLA est augmentée de 4 ms par 10 ms additionnelles d'effacement (jusqu'à concurrence de 10 ms) étant donné la probabilité de désaccord de phase entre les signaux synthétique et réel. La fenêtre OLA pour la fin de l'effacement de 20 ms (un quart de longueur d'onde + 4 ms) est représentée sous la forme de rampe montante à la fin de l'effacement dans la courbe "après 10 ms". Au cours des deuxièmes 10 ms d'un effacement, l'onde est également atténuée au moyen d'une rampe linéaire. Dans le cas d'un effacement long, cette atténuation conduit le signal synthétique à 0 après 60 ms.

La courbe "Masqué" fait apparaître le résultat final de l'algorithme. Pour des besoins de comparaison, le signal original, sans effacement, est reproduit en dessous. La voix synthétique est très proche de la parole avant l'effacement et elle est une bonne approximation de l'onde originale. Un examen poussé de l'onde synthétique révèle que la première période fondamentale dans l'effacement provient de la dernière période précédant l'effacement, la deuxième période de l'effacement provient de deux périodes précédant l'effacement et la troisième période de l'effacement répète la première période avant l'effacement. De même, en raison d'un changement de tonie au cours de l'effacement, la crête de la dernière période du signal synthétique n'est pas exactement alignée sur la dernière période du signal original. Pour cette raison, la fenêtre OLA à la fin de l'effacement doit être élargie à mesure que l'effacement se prolonge.

### I.3 Description de l'algorithme avec code C++ annoté

L'algorithme PLC a été mis en œuvre en virgule flottante avec une classe C++<sup>1</sup>. Ce code, ainsi qu'une explication de la manière dont il fonctionne, est présenté dans le présent sous-paragraphe. La mise en œuvre complète représente quelque 360 lignes de code C++, observations comprises. Le code C++ montré dans la présente annexe est montré à des fins d'illustration seulement.

#### I.3.1 Définition "typedef" et constantes

Afin de permettre le passage entre les calculs à double précision et à précision normale à virgule flottante, le type suivant est défini.

```
typedef float Float;
```

Pour passer au mode double précision, remplacer "float" par "double". Le code définit les constantes suivantes du préprocesseur:

```
#define PITCH_MIN      40                /* tonie minimale permise, 200 Hz */
#define PITCH_MAX      120               /* tonie maximale permise, 66 Hz */
#define PITCHDIFF      (PITCH_MAX - PITCH_MIN)
#define POVERLAPMAX    (PITCH_MAX >> 2) /* fenêtre OLA av. tonie maximale */
#define HISTORYLEN     (PITCH_MAX * 3 + POVERLAPMAX) /* longueur de tampon */
#define NDEC           2                 /* sous-échantillonnage 2:1 */
#define CORRLLEN       160               /* longueur de corrélation 20 ms */
#define CORRBUFLEN     (CORRLLEN + PITCH_MAX) /* longueur du tampon de corrélation */
#define CORRMINPOWER   ((Float)250.)    /* puissance minimale */
#define EOVERLAPINCR   32                /* terminer l'incrément OLA par trame, 4 ms */
#define FRAMESZ        80                /* 10 ms à 8 KHz */
#define ATTENFAC((Float).2) /* facteur d'atténuation par trame de 10 ms frame */
#define ATTENINCR      (ATTENFAC/FRAMESZ) /* atténuation par échantillon */
```

#### I.3.2 Déclaration de classe

```
1 class LowCFE {
2 public:
3     LowCFE();
4     void dofe(short *s); /* synthétisé la parole pour l'effacement */
5     void addtohistory(short *s); /* ajouter une trame correcte au tampon historique */
6 protected:
7     int eraseCnt; /* trames effacées consécutives */
8     int poverlap; /* superposition basée sur la tonie */
9     int poffset; /* départ dans la période fondamentale */
```

---

<sup>1</sup> Ce programme est également disponible en langage C dans le module G.711 de la Bibliothèque des outils logiciels de l'UIT (Rec. UIT-T G.191).

```

10      int      pitch;                /* évaluation de la tonie */
11      int      pitchblen;            /* longueur de période fondamentale du
                                         moment */
12      Float    *pitchbufend;          /* fin de tampon tonal */
13      Float    *pitchbufstart;        /* commencement de tampon tonal */
14      Float    pitchbuf[HISTORYLEN]; /* tampon pour cycles de parole */
15      Float    lastq[POVERLAPMAX];    /* dernier quart d'onde enregistré */
16      short    history[HISTORYLEN];   /* tampon historique */
17
18      void      scalespeech(short *out);
19      void      getfespeech(short *out, int sz);
20      void      savespeech(short *s);
21      int       findpitch();
22      void      overlapadd(Float *l, Float *r, Float *o, int cnt);
23      void      overlapadd(short *l, short *r, short *o, int cnt);
24      void      overlapaddatend(short *s, short *f, int cnt);
25      void      convertsf(short *f, Float *t, int cnt);
26      void      convertfs(Float *f, short *t, int cnt);
27      void      copyf(Float *f, Float *t, int cnt);
28      void      copys(short *f, short *t, int cnt);
29      void      zeros(short *s, int cnt);
30 };

```

La classe est appelée **LOWCFE**, soit "Low Complexity Frame Erasure concealment" (masquage simple de pertes de trame). Son interface est définie par trois fonctions publiques. Le constructeur de la ligne 3 initialise les variables internes. La fonction **addtohistory()** de la ligne 5 traite les trames correctes. Son argument, un indicateur désignant un ensemble d'entiers courts de longueur **FRAMESZ**, doit contenir une trame de données qui a été décodée. Le programme suppose qu'un entier court contient une donnée signée sur 16 bits avec signe et que le décodeur restitue des données à 16 bits uniformes. La fonction **dofe()** produit la voix synthétique au cours d'un effacement.

La partie restante de la classe est protégée, ce qui signifie que ses membres et fonctions sont inaccessibles à l'utilisateur de la classe. La variable **erasescnt** repère le nombre de trames effacées consécutives dans un effacement. Sa valeur, qui commence à 0, est augmentée de 1 toutes les 10 ms que dure un effacement; elle est remise à 0 à la première trame correcte qui suit un effacement.

La variable **poverlap** est la longueur de la fenêtre OLA et correspond au quart de la tonie du moment. La variable **poffset**, à la ligne 9, est le décalage dans le tampon tonal en vigueur; elle est utilisée pour produire une onde synthétique. La variable **pitch**, à la ligne 10, est l'estimation du moment en ce qui concerne la tonie. La variable **pitchblen** est la longueur de la partie du tampon tonal qui est utilisée sur le moment. Cette longueur change si l'effacement dure plus d'une trame. La variable **pitchbufend** désigne l'extrémité terminale du tampon tonal. La variable **pitchbufstart** est un indicateur désignant le commencement d'une partie du tampon tonal qui est en cours d'utilisation. La variable **pitchbuf**, à la ligne 14, contient le tampon historique au départ d'un effacement. Contrairement à celui qui correspond au dernier quart de longueur d'onde, ce tampon reste statique pendant la durée de l'effacement. Le tampon **lastq** contient une copie cachée du dernier quart de longueur d'onde du signal précédant le début de l'effacement; il est utilisé pour

les opérations OLA. Le tampon `history` de la ligne 16 contient l'historique récent du signal. Il est mis à jour pendant les trames correctes et les trames erronées.

Les fonctions protégées des lignes 18 à 29 mettent en œuvre la partie centrale de l'algorithme et sont décrites plus loin.

### I.3.3 Boucle principale

La boucle de traitement principale d'un programme qui utilise la classe `LowCFE` avec codage G.711 est montrée ci-dessous. Elle part du principe que la fonction `receiveframe()` est "vraie" si une trame de 10 ms d'un train binaire G.711 a été reçue sans erreur et qu'elle est "erronée" si un effacement survient. La fonction `g711dec()` convertit le train binaire G.711 en un produit MIC uniforme à 16 bits, et `output()` restitue la parole avec les effacements masqués. La mise en œuvre de ces fonctions ne relève pas du présent appendice et n'est donc pas traitée.

```
1 void process()
2 {
3     char    bitstream[FRAMESZ];
4     short   speech[FRAMESZ];
5     LowCFE  fec;
6     bool    frameisgood;
7
8     for(;;) {
9         frameisgood = receiveframe(bitstream);
10        if (frameisgood) {
11            g711dec(bitstream, speech);
12            fec.addtohistory(speech);
13        } else
14            fec.dofe(speech);
15        output(speech);
16    }
17 }
```

La ligne 10 comporte un test visant à déterminer si la trame d'entrée est effacée. Si la trame est correcte, elle est envoyée au décodeur à la ligne 11 et le résultat de celui-ci est soumis à l'algorithme PLC avec la fonction de membre `addtohistory()`. Si la trame est effacée, la fonction membre `dofe()` est appelée pour générer le signal vocal synthétique.

Il faut noter que la fonction `addtohistory` de la ligne 12 fait davantage que transférer une copie de la parole au tampon historique. Elle modifie également le contenu de la matrice de signaux vocaux. Le signal de sortie est retardé par les échantillons `POVERLAPMAX`. De plus, à la première trame correcte suivant un effacement, le signal d'entrée fait l'objet d'une opération OLA avec la voix synthétique avant de retourner.

### I.3.4 Fonctions membre utilitaire

Les fonctions utilitaires des lignes 25 à 29 de la déclaration de classe convertissent simplement les entiers courts en réels (`convertsf`) et inversement (`convertfs()`), copient des matrices de réels (`copyf()`) et d'entiers courts (`copys()`) et mettent à zéro une matrice d'entiers courts (`zeros()`). Elles sont présentées en premier parce qu'elles sont utilisées par les autres routines.

```

1 void LowcFE::convertsf(short *f, Float *t, int cnt)
2 {
3     for (int i = 0; i < cnt; i++)
4         t[i] = (Float)f[i];
5 }
6
7 void LowcFE::convertfs(Float *f, short *t, int cnt)
8 {
9     for (int i = 0; i < cnt; i++)
10         t[i] = (short)f[i];
11 }
12
13 void LowcFE::copyf(Float *f, Float *t, int cnt)
14 {
15     for (int i = 0; i < cnt; i++)
16         t[i] = f[i];
17 }
18
19 void LowcFE::copys(short *f, short *t, int cnt)
20 {
21     for (int i = 0; i < cnt; i++)
22         t[i] = f[i];
23 }
24
25 void LowcFE::zeros(short *s, int cnt)
26 {
27     for (int i = 0; i < cnt; i++)
28         s[i] = 0;
29 }

```

Il n'y a pas de saturation ou d'arrondi dans la routine `convertsf`.

### I.3.5 Constructeur

Le constructeur initialise les membres internes de la classe.

```

1 LowcFE::LowcFE()
2 {
3     eraseCnt = 0;
4     pitchBufEnd = &pitchBuf[HISTORYLEN];
5     zeros(history, HISTORYLEN);
6 }

```

A la ligne 3, `erasecnt` est mis à 0 afin que le code sache que cela n'est pas courant dans un effacement. Ensuite, `pitchbufend` est réglé de manière à designer la fin du tampon. Ensuite, le tampon historique est effacé pour éviter la production de défauts si un effacement se produit au départ du signal.

### I.3.6 Addtohistory et savespeech

Ensuite nous présentons la fonction interface publique `addtohistory()` ainsi qu'une fonction protégée, `savespeech`, appelée de manière interne par `addtohistory()`. `addtohistory()` est appelé par l'application après le décodage d'une trame correcte mais avant la restitution du signal.

```

1 /*
2  *   Enregistrer l'équivalent d'une trame de signaux vocaux dans le tampon
   historique.
3  *   Renvoyer la parole restituée retardée par POVERLAPMAX.
4  */
5 void LowcFE::savespeech(short *s)
6 {
7     /* faire de la place pour un nouveau signal */
8     copys(&history[FRAMESZ], history, HISTORYLEN - FRAMESZ);
9     /* copier dans la nouvelle trame */
10    copys(s, &history[HISTORYLEN - FRAMESZ], FRAMESZ);
11    /* copier hors de la trame retardée */
12    copys(&history[HISTORYLEN - FRAMESZ - POVERLAPMAX], s, FRAMESZ);
13 }
14
15 /*
16  *   Une trame correcte a été reçue et décodée.
17  *   Si cela se produit directement après un effacement, faire une
   superposition avec le signal synthétique.
18  *   Ajouter la trame au tampon historique.
19  */
20 void LowcFE::addtohistory(short *s)
21 {
22     if (erasecnt) {
23         short overlapbuf[FRAMESZ];
24         /*
25          *   les effacements longs nécessitent une superposition
26          *   longue pour lisser la transition entre le signal
27          *   synthétique et le signal réel.
28          */
29         int olen = poverlap + (erasecnt - 1) * EOVERLAPINCR;
30         if (olen > FRAMESZ)
31             olen = FRAMESZ;
32         getfespacech(overlapbuf, olen);

```



```

33         overlapaddatend(s, overlapbuf, olen);
34         erasecnt = 0;
35     }
36     savespeech(s);
37 }

```

A la ligne 22, `addtohistory()` vérifie s'il s'agit de la première trame correcte suivant un effacement. Si un effacement s'est produit dans la trame précédente, `erasecnt` contient le nombre de trames de 10 ms formant l'effacement. Si la trame précédente n'a pas été effacée, `erasecnt` est 0.

Si la trame précédente a été effacée, les lignes 23-34 continuent de produire le signal synthétique, font une opération OLA entre le signal synthétique et le signal d'entrée, puis effacent `erasecnt`. La ligne 29 détermine la longueur de la fenêtre OLA. La variable `poverlap` contient le nombre d'échantillons d'un quart de la période fondamentale estimée du moment. Si l'effacement ne dure que 10 ms (`erasecnt == 1`), la longueur de la fenêtre OLA est mise à `poverlap`. Si plusieurs trames ont été effacées, la longueur de la fenêtre OLA est augmentée de 4 ms pour chaque 10 ms d'effacement. Les lignes 30-31 font que la fenêtre OLA n'excède pas 10 ms en cas d'effacement long. La ligne 32 produit la voix synthétique dans le tampon temporaire `overlapbuf`. A la ligne 33, le signal synthétique fait l'objet d'une opération OLA avec le signal d'entrée. Le résultat est remis dans `s` en remplacement du signal d'entrée original.

La ligne 36 appelle `savespeech()` pour enregistrer le signal dans le tampon historique et ajouter le retard dû à l'algorithme. S'il ne s'agit pas de la première trame suivant un effacement, la parole originale est enregistrée. Sinon, c'est le résultat de l'opération OLA à la ligne 33.

La fonction `savespeech()`, aux lignes 5 à 13, enregistre la parole dans le tampon historique. Dans un traitement DSP cela pourrait être un tampon circulaire, mais pour les besoins de la simulation il est plus simple de déplacer le contenu. La ligne 8 déplace le tampon pour laisser la place à une nouvelle trame. Celle-ci est copiée dans la partie terminale du tampon à la ligne 10, et la ligne 12 remplace le contenu de l'ensemble d'entrée par le signal retardé par des échantillons `POVERLAPMAX` (30). Cela introduit le délai dû à l'algorithme, soit 3,75 ms.

### I.3.7 Dofe

La fonction membre public `dofe` produit le signal synthétique au cours de l'effacement et contient l'ensemble de l'algorithme PLC.

```

1  /*
2   * Produire le signal synthétique.
3   * Au début d'un effacement, déterminer la tonie et extraire une période
4   * fondamentale de la partie terminale du signal. Effectuer une opération OLA
5   * sur un quart de la tonie pour lisser le signal. Ensuite, répéter le signal
6   * extrait pendant toute la durée de l'effacement. Si l'effacement
7   * se poursuit pendant plus de 10 ms, augmenter le nombre de périodes dans le
8   * tampon tonal. A la fin d'un effacement, effectuer l'opération OLA avec le
9   * début de la première trame correcte. Le gain décroît à mesure que
10    l'effacement se prolonge.
11 */
12 void LowCFE::dofe(short *out)

```

```

12 {
13     if (erasescnt == 0) {
14         convertsf(history, pitchbuf, HISTORYLEN); /* obtenir
15                                                    l'historique */
16         pitch = findpitch(); /* déterminer la tonie */
17         poverlap = pitch >> 2; /* OLA sur un quart de longueur
18                                d'onde */
19         /* enregistrer les derniers échantillons poverlap originaux */
20         copyf(pitchbufend - poverlap, lastq, poverlap);
21         poffset = 0; /* créer un tampon tonal avec 1 période */
22         pitchblen = pitch;
23         pitchbufstart = pitchbufend - pitchblen;
24         overlapadd(lastq, pitchbufstart - poverlap,
25                   pitchbufend - poverlap, poverlap);
26         /* mettre à jour le dernier quart de longueur d'onde dans le
27            tampon historique */
28         convertfs(pitchbufend - poverlap, &history[HISTORYLEN-poverlap],
29                   poverlap);
30         getfespace(out, FRAMESZ); /* obtenir la voix
31                                   synthétique */
32     } else if (erasescnt == 1 || erasescnt == 2) {
33         /* estimation de la partie terminale de la tonie précédente */
34         short tmp[POVERLAPMAX];
35         int saveoffset = poffset; /* enregistrer le décalage pour
36                                   l'opération OLA */
37         getfespace(tmp, poverlap); /* poursuivre avec l'ancien
38                                   tampon tonal */
39         /* ajouter des périodes au tampon tonal */
40         poffset = saveoffset;
41         while (poffset > pitch)
42             poffset -= pitch;
43         pitchblen += pitch; /* ajouter une période */
44         pitchbufstart = pitchbufend - pitchblen;
45         overlapadd(lastq, pitchbufstart - poverlap,
46                   pitchbufend - poverlap, poverlap);
47         /* superposer (OCA) le nouveau tampon tonal à l'ancien */
48         getfespace(out, FRAMESZ);
49         overlapadd(tmp, out, out, poverlap);
50         scalespeech(out);
51     } else if (erasescnt > 5) {
52         zeros(out, FRAMESZ);
53     } else {
54         getfespace(out, FRAMESZ);
55         scalespeech(out);
56     }
57 }

```

```

51     erasecnt++;
52     savespeech(out);
53 }

```

A la ligne 13, `erasecnt` est testé pour déterminer s'il est égal à 0. Si c'est le cas, il s'agit de la première trame d'un effacement et le code des lignes 14 à 27 s'exécute. La ligne 14 copie le contenu du tampon historique dans le tampon tonal et le convertit en valeur à virgule flottante dans le processus. Exception faite du dernier quart de longueur d'onde, le contenu du tampon tonal ne change pas pendant la durée de l'effacement. La ligne 15 appelle `findpitch` pour effectuer une corrélation croisée normalisée pour évaluer la tonie. La fonction `findpitch()` renvoie une valeur entre `MIN_PITCH(40)` et `MAX_PITCH(120)`. `findpitch()` limite la complexité de l'algorithme et n'est appelé que pour la première trame d'un effacement. La ligne 16 détermine la longueur de la fenêtre OLA à un quart de la période tonale. A la ligne 18, le dernier quart de longueur d'onde du tampon tonal est enregistré dans `lastq` si l'effacement dure plus d'une trame. Les lignes 19 à 21 mettent ensuite le tampon tonal de telle manière que seule la dernière période du tampon est utilisée pour produire la voix synthétique. A la ligne 22, contenu de `lastq` est une opération OLA avec le quart de période commençant à 1,25 période en retrait dans le tampon tonal. Cela produit une transition douce entre la parole originale et la voix synthétique au départ d'un effacement, tout en assurant une transition douce si le tampon tonal à une seule période est répété au cours de la première trame, par exemple si la période est inférieure à 80 échantillons (10 ms).

Le résultat de cette opération OLA est placé dans la partie postérieure du tampon tonal et remplace le dernier quart de période du tampon historique à la ligne 25. La mise à jour du tampon historique a pour effet de produire la parole traitée OLA lorsque `savespeech` est appelé à la ligne 52 et fait en sorte que le tampon historique ne contient aucune discontinuité. La ligne 27 produit ensuite la voix synthétique pour la trame par l'appel de `getfspeech()`. Cette fonction `getfspeech()` copie simplement la dernière période du tampon tonal dans l'ensemble de sortie, répétant la période autant de fois qu'il le faut pour combler les 80 échantillons. Après la ligne 27, le code passe à la ligne 51 au moyen d'incrément `erasecnt`. La ligne 52 met à jour le tampon historique au moyen de la voix synthétique. La fonction `savespeech()` retarde également le signal; aussi, quand `savespeech()` renvoie le résultat de l'opération OLA à la ligne 22, ce résultat sera présent dans les premiers 3,75 ms de la restitution.

Le branchement des lignes 29-44 intervient sur les deuxième et troisième trames d'un effacement. Ce branchement augmente le nombre de périodes utilisées dans le tampon tonal pour synthétiser la tonie. Le fait d'augmenter le nombre de périodes fondamentales augmente la variation dans le signal ce qui, à son tour, diminue le nombre de défauts harmoniques artificiels (bip) qui surviennent si l'on utilise qu'une seule période fondamentale. Aux lignes 30 à 32, la restitution du tampon tonal utilisée dans la trame effacée précédente se poursuit pendant un quart de période pour être ensuite placée dans un tampon provisoire. Le signal est soumis à une opération OLA avec le contenu du nouveau tampon tonal élargi pour aboutir à une transition douce du signal de sortie, le nombre de périodes fondamentales du tampon tonal ayant augmenté.

A la ligne 31, le décalage dans le tampon tonal est enregistré dans `saveoffset`. Lorsque l'ancien tampon tonal a produit l'onde de la ligne 32, `poffset` est rétabli en `saveoffset` à la ligne 34, ce qui permet de maintenir la phase du signal synthétique. Les lignes 35 à 36 ôtent des périodes fondamentales de `poffset` jusqu'à ce que celui-ci désigne la première période.

La ligne 37 ajoute une période au tampon tonal. La ligne 38 met à jour `pitchbufstart`, l'indicateur désignant le départ de la partie utilisée du tampon tonal. La ligne 39 effectue une opération OLA entre les données enregistrées dans `lastq` et le quart de période précédant `pitchbufstart`, plaçant les résultats dans le dernier quart de période à l'extrémité arrière du

tampon tonal. Comme dans la première trame, ceci a pour effet de lisser le tampon tonal s'il est répété à plusieurs reprises dans le signal de sortie.

A la ligne 42, le signal synthétique provenant du nouveau tampon tonal est extrait pour la durée de la trame, et le quart de période au départ de ces données est soumis à une opération OLA avec le tampon provisoire créé à la ligne 32. A la ligne 44, le signal synthétique est atténué au moyen d'une rampe linéaire au moyen d'un appel lancé à `scalespeech`. Cette atténuation se fait au taux de 20% par 10 ms et commence au début de la deuxième trame effacée. On notera que le signal synthétique n'est pas atténué au cours de la première trame d'un effacement.

Pendant les quatrième, cinquième et sixième trames d'un effacement, le traitement est simplifié étant donné que le tampon tonal reste statique, comme le montrent les lignes 48-49. Le signal synthétique est généré au moyen d'un appel à `getfspeech()` pour être ensuite atténué au moyen de `scalespeech()`. Au delà de 60 ms, le signal synthétique est mis à 0 à la ligne 46.

### I.3.8 Détection de la tonie

Le détecteur de tonie est la seule partie de l'algorithme qui présente une relative complexité. Pour que celle-ci reste faible, une recherche approximative est d'abord effectuée sur un signal sous-échantillonné. Ensuite une recherche précise est effectuée au voisinage de la crête du résultat de la recherche approximative. Les dernières 20 ms du signal sont intercorrélées avec le signal antérieur à des retards de PITCH\_MIN à PITCH\_MAX.

```
1 /*
2  * Estimer la tonie.
3  * l - désigner le premier échantillon dans les 20 dernières ms de la parole.
4  * r - désigner l'échantillon PITCH_MAX précédant l
5  */
6 int LowcFE::findpitch()
7 {
8     int    i, j, k;
9     int    bestmatch;
10    Float  bestcorr;
11    Float  corr;          /* corrélation */
12    Float  energy;        /* énergie courante */
13    Float  scale;         /* corrélation d'échelle par puissance moyenne */
14    Float  *rp;           /* segment de correspondance */
15    Float  *l = pitchbufend - CORRLLEN;
16    Float  *r = pitchbufend - CORRBULEN;
17
18    /* recherche approximative */
19    rp = r;
20    energy = 0.f;
21    corr = 0.f;
22    for (i = 0; i < CORRLLEN; i += NDEC) {
23        energy += rp[i] * rp[i];
24        corr += rp[i] * l[i];
25    }
```

```

26     scale = energy;
27     if (scale < CORRMINPOWER)
28         scale = CORRMINPOWER;
29     corr = corr / (Float)sqrt(scale);
30     bestcorr = corr;
31     bestmatch = 0;
32     for (j = NDEC; j <= PITCHDIFF; j += NDEC) {
33         energy -= rp[0] * rp[0];
34         energy += rp[CORRLEN] * rp[CORRLEN];
35         rp += NDEC;
36         corr = 0.f;
37         for (i = 0; i < CORRLEN; i += NDEC)
38             corr += rp[i] * l[i];
39         scale = energy;
40         if (scale < CORRMINPOWER)
41             scale = CORRMINPOWER;
42         corr /= (Float)sqrt(scale);
43         if (corr >= bestcorr) {
44             bestcorr = corr;
45             bestmatch = j;
46         }
47     }
48     /* recherche précise */
49     j = bestmatch - (NDEC - 1);
50     if (j < 0)
51         j = 0;
52     k = bestmatch + (NDEC - 1);
53     if (k > PITCHDIFF)
54         k = PITCHDIFF;
55     rp = &r[j];
56     energy = 0.f;
57     corr = 0.f;
58     for (i = 0; i < CORRLEN; i++) {
59         energy += rp[i] * rp[i];
60         corr += rp[i] * l[i];
61     }
62     scale = energy;
63     if (scale < CORRMINPOWER)
64         scale = CORRMINPOWER;
65     corr = corr / (Float)sqrt(scale);
66     bestcorr = corr;
67     bestmatch = j;

```

```

68     for (j++; j <= k; j++) {
69         energy -= rp[0] * rp[0];
70         energy += rp[CORRLEN] * rp[CORRLEN];
71         rp++;
72         corr = 0.f;
73         for (i = 0; i < CORRLEN; i++)
74             corr += rp[i] * l[i];
75         scale = energy;
76         if (scale < CORRMINPOWER)
77             scale = CORRMINPOWER;
78         corr = corr / (Float)sqrt(scale);
79         if (corr > bestcorr) {
80             bestcorr = corr;
81             bestmatch = j;
82         }
83     }
84     return PITCH_MAX - bestmatch;
85 }

```

A l'appel de `findpitch()`, le contenu du tampon tonal correspond au contenu du tampon historique. La ligne 15 met le signal de référence (1) à l'échantillon de 20 ms précédant le début de l'effacement. Auparavant, la ligne 16 met le signal de retard à des échantillons `MAX_PITCH`. Les lignes 19-29 calculent l'intercorrélation normalisée au retard `MAX_PITCH` sur un signal décimal de 2:1. Les lignes 26-28 verrouillent l'énergie à un niveau minimal pour éviter une division par zéro et la perte d'intérêt pour les zones à très faible énergie.

Les lignes 32-47 répètent le calcul d'intercorrélation normalisée pour chaque autre décalage. Seuls les décalages pairs sont examinés. La somme des produits de l'énergie est établie de telle manière qu'il suffit de deux opérations multiplication-accumulation (MAC) pour la mettre à jour par itération. La crête de la corrélation est contenue dans `bestcorr`, et le décalage correspondant est contenu dans `bestmatch`.

Les lignes 48-54 calculent la zone utilisée pour la recherche détaillée. Si la crête obtenue par la recherche approximative n'est pas au décalage minimum ou maximum, la recherche détaillée porte sur 3 décalages. Les lignes 55-83 recommencent le calcul effectué dans la recherche approximative, mais sans sous-échantillonnage. La crête de la recherche détaillée est renvoyée en tant qu'estimation de la tonie à la ligne 84.

### I.3.9 Production et atténuation du signal synthétique

La fonction `getfespeech()` extrait l'onde synthétique du tampon tonal, alors que `scalespeech` applique la rampe d'atténuation à la voix synthétique.

```

1 /*
2  *   Obtenir des échantillons dans le tampon tonal circulaire. Mettre à jour
   poffset
3  *   pour que le signal continu à l'effacement des trames subséquentes
4  */
5 void LowCFE::getfespeech(short *out, int sz)

```

```

6 {
7     while (sz) {
8         int cnt = pitchblen - poffset;
9         if (cnt > sz)
10             cnt = sz;
11         convertfs(&pitchbufstart[poffset], out, cnt);
12         poffset += cnt;
13         if (poffset == pitchblen)
14             poffset = 0;
15         out += cnt;
16         sz -= cnt;
17     }
18 }
19
20 void LowcFE::scalespeech(short *out)
21 {
22     Float g = (Float)1. - (erasecnt - 1) * ATTENFAC;
23     for (int i = 0; i < FRAMESZ; i++) {
24         out[i] = (short)(out[i] * g);
25         g -= ATTENINCR;
26     }
27 }

```

`getfespeech()` des lignes 5-18 copie essentiellement l'onde du tampon tonal vers l'ensemble de sortie. Si la taille demandée est plus grande que le tampon tonal, l'indicateur de sortie `poffset` revient au début du tampon et continue à partir de ce point. Le tampon tonal commence à `pitchbufstart` et sa longueur est de `pitchblen` échantillons. La variable `poffset` est la position du moment dans le tampon. `poffset` est mis à jour à chaque itération à travers la boucle et aux points de retour à l'échantillon suivant qui doit être produit. De cette manière la génération du signal synthétique peut se poursuivre si `getfespeech()` est appelé à nouveau.

La fonction `scalespeech()` des lignes 20-27 applique la rampe d'atténuation à la valeur correspondant à une trame de voix synthétique. Elle n'est pas appelée au cours de la première trame d'un effacement. Dans la deuxième trame, `erasecnt` sera 1 étant donné qu'il n'y a pas encore eu d'incrément, et le gain démarrera donc à 1 et descendra jusqu'à 0,8. L'atténuation se poursuit à 20% par trame pour les trames effacées subséquentes.

### I.3.10 Opérateurs de superposition/addition OLA

Les opérateurs de superposition/addition complètent le code source. Trois routines sont fournies. Deux d'entre elles sont identiques sauf en ce qui concerne leur type d'argument et surchargent le même nom de fonction membre, `overlapadd`. La version flottante est appelée pour les opérations OLA effectuées sur le tampon tonal, alors que la version abrégée est appelée pour les opérations OLA sur le signal de sortie.

```

1 /*
2  * Opération OLA côtés gauche et droit
3  */

```

```

4 void LowcFE::overlapadd(Float *l, Float *r, Float *o, int cnt)
5 {
6     Float incr = (Float)1. / cnt;
7     Float lw = (Float)1. - incr;
8     Float rw = incr;
9     for (int i = 0; i < cnt; i++) {
10         Float t = lw * l[i] + rw * r[i];
11         if (t > 32767.)
12             t = 32767.;
13         else if (t < -32768.)
14             t = -32768.;
15         o[i] = t;
16         lw -= incr;
17         rw += incr;
18     }
19 }
20
21 void LowcFE::overlapadd(short *l, short *r, short *o, int cnt)
22 {
23     Float incr = (Float)1. / cnt;
24     Float lw = (Float)1. - incr;
25     Float rw = incr;
26     for (int i = 0; i < cnt; i++) {
27         Float t = lw * l[i] + rw * r[i];
28         if (t > 32767.)
29             t = 32767.;
30         else if (t < -32768.)
31             t = -32768.;
32         o[i] = (short)t;
33         lw -= incr;
34         rw += incr;
35     }
36 }

```

L'opération OLA utilise des fenêtres triangulaires qui sont calculées dans la routine basée sur la longueur de l'argument `cnt`. La ligne 6 calcule l'incrément de fenêtre par échantillon, alors que les lignes 7 et 8 initialisent les pondérations des fenêtres côté gauche et côté droit. Les lignes 10-17 appliquent les pondérations à chaque échantillon,aturent la valeur à un entier à 16 bits, produisent le résultat puis mettent à jour les pondérations pour l'itération suivante.

Une routine de OLA additionnelle, `overlapaddatend()`, est appelée à la première trame correcte suivant un effacement. Cette routine diffère de celle ci-dessus en ce sens qu'elle change l'échelle de la voix synthétique au moyen du facteur d'atténuation avant de la combiner au signal d'entrée.



```

1 /*
2  * Faire une opération OLA à la fin de l'effacement au moyen du début de la
   première trame correcte
3  * Mettre à l'échelle la voix synthétique au moyen du facteur de gain avant
   l'opération OLA.
4  */
5 void LowcFE::overlapdatend(short *s, short *f, int cnt)
6 {
7     Float incr = (Float)1. / cnt;
8     Float gain = (Float)1. - (erasecnt - 1) * ATTENFAC;
9     if (gain < 0.)
10         gain = (Float)0.;
11     Float incrg = incr * gain;
12     Float lw = ((Float)1. - incr) * gain;
13     Float rw = incr;
14     for (int i = 0; i < cnt; i++) {
15         Float t = lw * f[i] + rw * s[i];
16         if (t > 32767.)
17             t = 32767.;
18         else if (t < -32768.)
19             t = -32768.;
20         s[i] = (short)t;
21         lw -= incrg;
22         rw += incr;
23     }
24 }

```

#### I.4 Complexité et délai

On estime que la complexité de l'algorithme présente un taux crête d'environ 1/2 MIPS de traitement DSP. La moyenne est considérablement plus basse. La complexité est dominée par le calcul du terme d'intercorrélacion dans la routine de détection de la tonie. Ce calcul ne survient qu'au cours de la première trame d'un effacement. Pour toutes les autres trames, la complexité est très faible, de l'ordre de quelques instructions multipliées-accumulées (MAC, *multiply accumulate*) par échantillon.

On évalue la complexité de la manière suivante: à la première trame de la partie effacée, l'algorithme doit évaluer la tonie et effectuer une opération OLA pour un quart de période fondamentale. Comme la valeur maximale d'une période est de 120 échantillons, cela revient donc à 30 échantillons. Les routines `findpitch()` et `overlapadd()` présentent les nombres d'opérateurs suivants:

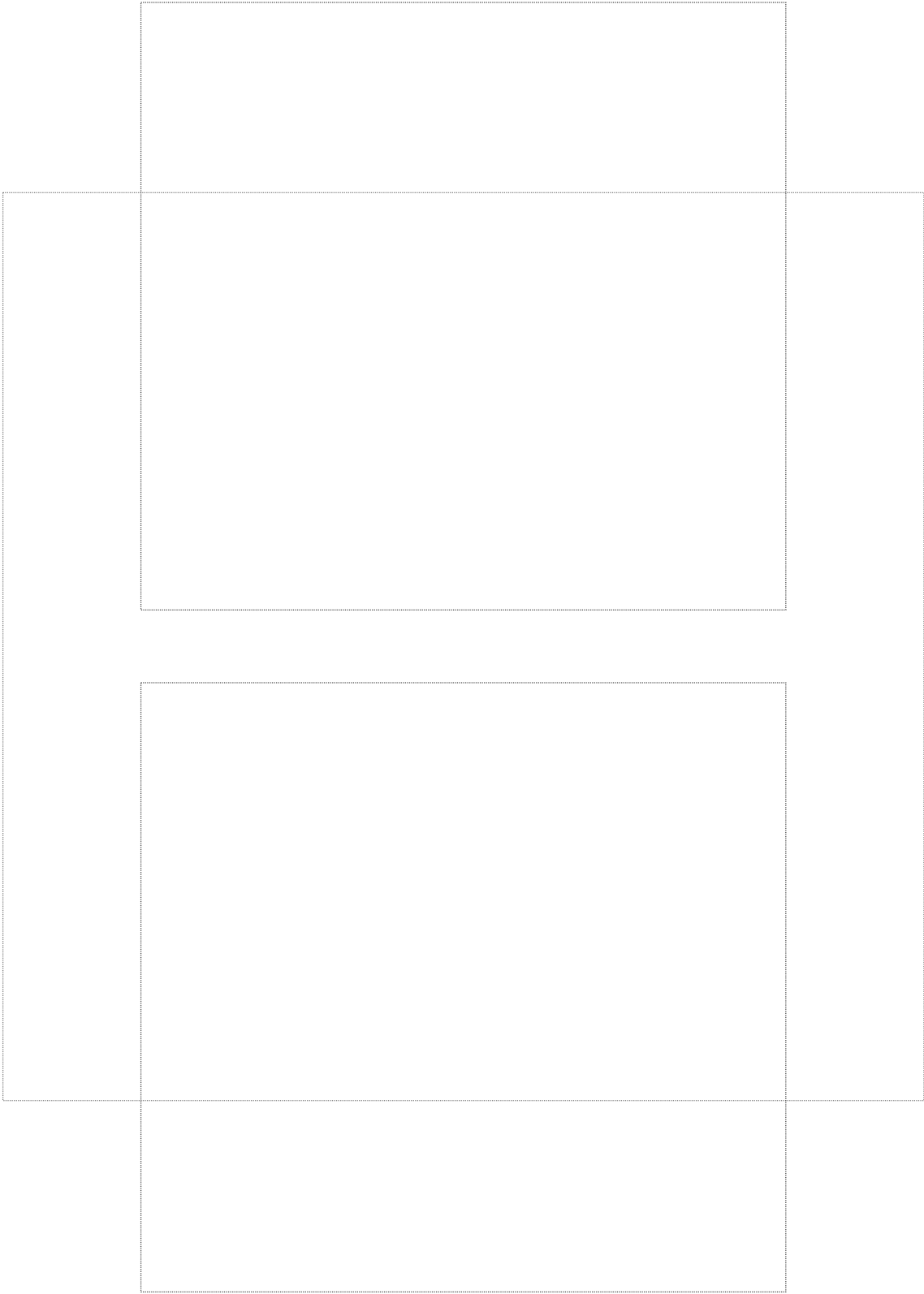
| Routine                   | MAC  | comp. | div. | racine |
|---------------------------|------|-------|------|--------|
| <code>findpitch()</code>  | 3764 | 86    | 44   | 44     |
| <code>overlapadd()</code> | 121  | 0     | 1    | 0      |
| <b>Total</b>              | 3885 | 86    | 45   | 44     |

A supposer 2 cycles pour une comparaison et 10 pour une division ou une racine carrée on obtient  $3885 + 86 * 2 + (45 + 44) * 10 = 4947$  cycles. Etant donné que cela se produit dans une trame de 10 ms, on multiplie par 100 pour obtenir 0,5 MIPS.

Comme indiqué précédemment, l'algorithme introduit un retard de 3,75 ms. Pour réduire autant que possible les retards produits par le calcul, on peut exécuter l'algorithme PLC après chaque trame correcte, avant de savoir si la trame suivante est effacée, cela au prix d'un peu de mémoire. Dans ce cas, si la trame suivante est effacée, le signal synthétique est disponible immédiatement. Si la trame suivante n'est pas effacée, le signal synthétique est simplement rejeté.

### **Bibliographie**

- [1] Recommandation UIT-T G.723.1 (1996), *Codeur de signaux vocaux à double débit pour communications multimédias acheminées à 5,3 kbit/s et à 6,3 kbit/s.*
- [2] Recommandation CCITT G.728 (1992), *Codage de la parole à 16 kbit/s en utilisant la prédiction linéaire à faible délai avec excitation par code.*
- [3] Recommandation UIT-T G.729 (1996), *Codage de la parole à 8 kbit/s par prédiction linéaire avec excitation par séquences codées à structure algébrique conjuguée.*



## SERIES DES RECOMMANDATIONS UIT-T

|                |                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| Série A        | Organisation du travail de l'UIT-T                                                                                                              |
| Série B        | Moyens d'expression: définitions, symboles, classification                                                                                      |
| Série C        | Statistiques générales des télécommunications                                                                                                   |
| Série D        | Principes généraux de tarification                                                                                                              |
| Série E        | Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains                                            |
| Série F        | Services de télécommunication non téléphoniques                                                                                                 |
| <b>Série G</b> | <b>Systèmes et supports de transmission, systèmes et réseaux numériques</b>                                                                     |
| Série H        | Systèmes audiovisuels et multimédias                                                                                                            |
| Série I        | Réseau numérique à intégration de services                                                                                                      |
| Série J        | Transmission des signaux radiophoniques, télévisuels et autres signaux multimédias                                                              |
| Série K        | Protection contre les perturbations                                                                                                             |
| Série L        | Construction, installation et protection des câbles et autres éléments des installations extérieures                                            |
| Série M        | RGT et maintenance des réseaux: systèmes de transmission, de télégraphie, de télécopie, circuits téléphoniques et circuits loués internationaux |
| Série N        | Maintenance: circuits internationaux de transmission radiophonique et télévisuelle                                                              |
| Série O        | Spécifications des appareils de mesure                                                                                                          |
| Série P        | Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux                                                             |
| Série Q        | Commutation et signalisation                                                                                                                    |
| Série R        | Transmission télégraphique                                                                                                                      |
| Série S        | Equipements terminaux de télégraphie                                                                                                            |
| Série T        | Terminaux des services télématiques                                                                                                             |
| Série U        | Commutation télégraphique                                                                                                                       |
| Série V        | Communications de données sur le réseau téléphonique                                                                                            |
| Série X        | Réseaux pour données et communication entre systèmes ouverts                                                                                    |
| Série Y        | Infrastructure mondiale de l'information et protocole Internet                                                                                  |
| Série Z        | Langages et aspects informatiques généraux des systèmes de télécommunication                                                                    |