



UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

G.728

Annexe H
(05/99)

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

Systemes de transmission numériques – Equipements
terminaux – Codage des signaux analogiques par des
méthodes autres que la MIC

Codage de la parole à 16 kbit/s en utilisant la
prédiction linéaire à faible délai avec excitation par
code

**Annexe H: Fonctionnement de l'algorithme
LD-CELP à débit variable à des débits inférieurs
à 16 kbit/s principalement pour les équipements
DCME**

Recommandation UIT-T G.728 – Annexe H

(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
SYSTÈMES INTERNATIONAUX ANALOGIQUES À COURANTS PORTEURS	
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 RADIODÉLÉPHONIE ET DE LA TÉLÉPHONIE SUR LIGNES	G.450–G.499
EQUIPEMENTS DE TEST	
CARACTÉRISTIQUES DES SUPPORTS DE TRANSMISSION	
SYSTÈMES DE TRANSMISSION NUMÉRIQUES	
EQUIPEMENTS TERMINAUX	G.700–G.799
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.

RECOMMANDATION UIT-T G.728

CODAGE DE LA PAROLE À 16 kbit/s EN UTILISANT LA PRÉDICTION LINÉAIRE À FAIBLE DÉLAI AVEC EXCITATION PAR CODE

ANNEXE H

Fonctionnement de l'algorithme LD-CELP à débit variable à des débits inférieurs à 16 kbit/s principalement pour les équipements DCME

Résumé

La présente annexe contient les modifications apportées à l'algorithme LD-CELP figurant dans la Recommandation G.728. Ces modifications sont nécessaires pour réduire le débit de codage à 12,8 kbit/s et à 9,6 kbit/s. Elles portent également sur les répertoires de formes et de gains.

La présente édition fournit la description supplémentaire du calcul de GSTATE(1) ainsi que les valeurs supplémentaires des tableaux en relation avec le répertoire de gain.

La présente annexe comporte, sous format électronique, les vecteurs test destinés à valider les implémentations LD_CELP à faible débit.

Source

La Recommandation UIT-T G.728, Annexe H, révisée par la Commission d'études 16 (1997-2000) de l'UIT-T, a été approuvée le 27 mai 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 avait 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

	Page
H.1 Introduction.....	1
H.2 Principes de fonctionnement.....	1
H.2.1 Procédure de réduction du débit de codage.....	1
H.2.2 Principe du fonctionnement à 12,8 kbit/s.....	2
H.2.3 Principe du fonctionnement à 9,6 kbit/s.....	2
H.3 Modifications pour le fonctionnement à 12,8 kbit/s.....	2
H.3.1 Pseudo-code.....	2
H.3.2 Nouvelles tables de gains additionnels.....	7
H.3.3 Modification du paramètre de codeur.....	7
H.4 Modifications pour le fonctionnement à 9,6 kbit/s.....	8
H.4.1 Pseudo-code.....	8
H.4.2 Nouvelles tables de gains additionnels.....	12
H.4.3 Modification du paramètre de codeur.....	13

Fichiers électroniques joints:

- données de test pour le fonctionnement à 9,6 kbit/s
- données de test pour le fonctionnement à 12,8 kbit/s

Recommandation G.728

CODAGE DE LA PAROLE À 16 kbit/s EN UTILISANT LA PRÉDICTION LINÉAIRE À FAIBLE DÉLAI AVEC EXCITATION PAR CODE

ANNEXE H¹

Fonctionnement de l'algorithme LD-CELP à débit variable à des débits inférieurs à 16 kbit/s principalement pour les équipements DCME

(révisée en 1999)

H.1 Introduction

La présente annexe contient les modifications apportées à l'algorithme LD-CELP figurant dans la Recommandation G.728. Ces modifications sont nécessaires pour réduire le débit de codage à 12,8 kbit/s et à 9,6 kbit/s. Elles portent également sur les répertoires de formes et de gains.

La présente annexe part du principe que le lecteur s'est déjà familiarisé avec les spécifications contenues dans la Recommandation G.728, dont l'algorithme ne sera pas analysé ici. Seules les modifications à la Recommandation G.728 seront décrites.

La présente annexe se compose de quatre sous-paragraphes. Le sous-paragraph H.2 décrit le principe de fonctionnement pour la réduction du débit. Le sous-paragraph H.3 décrit les modifications requises pour le fonctionnement à 12,8 kbit/s. Le sous-paragraph H.4 décrit les modifications requises pour le fonctionnement à 9,6 kbit/s.

H.2 Principes de fonctionnement

H.2.1 Procédure de réduction du débit de codage

En réduisant la taille du répertoire, il est possible de réduire le débit sans changer notablement l'algorithme de codage. Dans le texte principal de la Recommandation G.728, l'index des répertoires se compose de 10 éléments binaires, correspondant à 1024 vecteurs de répertoire. Une réduction de 2 bits sur 10 de l'index de répertoire réduit le débit de codage de 16 kbit/s à 12,8 kbit/s; une réduction de 4 bits réduit le débit de codage à 9,6 kbit/s.

L'index de répertoire (du bit 9 au bit 0) est subdivisé en deux sections: 7 bits (du bit 9 au bit 3) pour le répertoire de formes et 3 bits (du bit 2 au bit 0) pour le répertoire de gains. Le répertoire de formes codé sur 7 bits comprend 128 vecteurs codes; le répertoire de gains codé sur 3 bits comprend 8 valeurs scalaires qui sont symétriques par rapport à zéro.

¹ Cette annexe est un complément facultatif de la Recommandation G.728. Elle n'est pas nécessaire pour implémenter correctement le codeur et le décodeur. Elle est destinée à améliorer les performances de l'algorithme G.728 dans certaines applications spécifiques, comme celles des équipements de multiplication de circuit numérique (DCME, *digital circuit multiplication equipment*). Le soin est laissé aux réalisateurs de choisir s'il y a lieu d'utiliser la présente annexe.

La présente annexe comporte, sous format électronique, des données destinées à tester les implémentations LD_CELP à faible débit.

Le nombre de vecteurs contenus dans le répertoire de formes peut être réduit comme suit. La probabilité de présence de vecteurs dans le répertoire de formes, obtenue à partir d'échantillons de parole normale, ne suit pas une loi de répartition uniforme. On observe que la probabilité des numéros 65 à 128 des index de répertoire est beaucoup plus grande que celle de numéros 1 à 64. En tirant parti de cette non-uniformité de répartition, il est possible de limiter le nombre de vecteurs de répertoire sans introduire beaucoup de dégradation qualitative des signaux vocaux. Par exemple, le bit 9 des index du répertoire de formes se prête à la réduction.

Une autre manière de diminuer le volume du répertoire de formes consiste à remodeler le répertoire optimisé pour chaque fonctionnement à débit réduit. Cela exigerait toutefois un plus grand nombre d'emplacements de mémoire et une modification notable des implémentations.

On peut réduire comme suit le nombre des valeurs de gain contenues dans le répertoire de gains. Pour diminuer le nombre d'éléments binaires utilisés pour coder le répertoire de gains, il est préférable de remodeler le répertoire de gains réduit, en l'optimisant pour le fonctionnement à débit réduit, parce que ce répertoire n'utilisera qu'une petite fraction de la mémoire. Chaque répertoire de gains de taille réduite devra être optimisé sur la base de la répartition des probabilités des valeurs de gain avant quantification.

La réduction du nombre de bits d'index de répertoire pour le fonctionnement à 12,8 kbit/s est décrite au H.2.2 et au H.2.3 pour le fonctionnement à 9,6 kbit/s.

H.2.2 Principe du fonctionnement à 12,8 kbit/s

Deux bits doivent être éliminés de l'index de répertoire codé sur 10 bits afin d'obtenir le fonctionnement à 12,8 kbit/s. Le bit 9 des index du répertoire de formes n'est pas utilisé et on choisit un répertoire de gains réduit à quatre valeurs. L'élimination du bit 9 des index de répertoire de formes réduit leur nombre à l'étendue de 65 à 128. Les quatre valeurs du répertoire de gains et les valeurs associées sont optimisées pour le fonctionnement à 12,8 kbit/s; elles sont représentées, au H.3.2, dans le Tableau H.1 (pour le calcul en virgule flottante) et dans le Tableau H.2 (pour le calcul en virgule fixe).

H.2.3 Principe du fonctionnement à 9,6 kbit/s

Quatre bits doivent être éliminés de l'index de répertoire codé sur 10 bits afin d'obtenir le fonctionnement à 9,6 kbit/s. Les bits 9, 8 et 5 sont éliminés des index du répertoire de formes et on réduit le répertoire de gains de 8 à 4 valeurs.

L'élimination des bits 9, 8 et 5 des index de répertoire de formes réduit leur nombre aux étendues de 97 à 100, 105 à 108, 113 à 116 et 121 à 124. Les quatre valeurs du répertoire de gains et les valeurs associées sont optimisées pour le fonctionnement à 9,6 kbit/s; elles sont représentées au H.4.2, dans le Tableau H.4 (pour le calcul en virgule flottante) et dans le Tableau H.5 (pour le calcul en virgule fixe).

H.3 Modifications pour le fonctionnement à 12,8 kbit/s

H.3.1 Pseudo-code

Seules les séquences d'exécution par blocs sont représentées; les détails de couche inférieure pour la transmission des paramètres ne sont pas décrits.

H.3.1.1 Blocs 17 et 18 – Calculateur d'erreur et sélecteur du meilleur index du répertoire

On trouvera dans le présent sous-paragraphe les pseudo-codes aussi bien en virgule flottante qu'en virgule fixe pour les blocs 17 et 18. Ces codes ont été modifiés pour le fonctionnement à 12,8 kbit/s et doivent remplacer les blocs 17 et 18 originaux qui figurent au 5.11/G.728. Le pseudo-code en virgule flottante est représenté en premier.

```

Initialiser DISTM au plus grand entier représentable par la machine
N1=NG_128/2
Pour J=65,66,...,NCWD, faire
  J1=(J-1)*IDIM
  COR=0.
  Pour K=1,2,...,IDIM, faire la ligne suivante
    COR=COR+PN(K)*Y(J1+K)          | calculer le produit interne  $P_j$ 

  Si COR > 0, alors faire les trois lignes suivantes
    IDXG=N1
    Si COR < GB_128(1)*Y2(J), alors faire la ligne suivante
      IDXG=1                          | meilleur gain positif trouvé

  Si COR ≤ 0, alors faire les trois lignes suivantes
    IDXG=NG_128
    Si COR > GB_128(3)*Y2(J), alors faire la ligne suivante
      IDXG=3                          | meilleur gain négatif trouvé

  D=-G2_128(IDXG)*COR+GSQ_128(IDXG)*Y2(J) | calculer la distorsion  $\hat{D}$ 

  Si D < DISTM, alors faire les trois lignes suivantes
    DISTM=D                          | sauvegarde de la plus faible distorsion
    IG=IDXG                          | et des meilleurs index du répertoire
    IS=J                              | trouvé jusqu'à maintenant.

Répéter le module en retrait ci-dessus pour le J suivant

IS1=IS-NCWD/2

ICHAN=(IS1-1)*NG_128+(IG-1)          | concaténer les index des
                                      | répertoires de formes et de gains.

```

Transmettre ICHAN par le canal de communication. Dans le cas d'une transmission en série, le bit le plus significatif de l'élément ICHAN sera transmis en premier. Si ICHAN est représenté par le mot de 8 bits b7, b6, b5, b4, b3, b2, b1, b0, alors l'ordre de transmission des bits sera b7, puis b6, b5, b4, b3, b2, b1, b0 (b7 étant le bit le plus significatif).

La version en virgule fixe du même module est reproduite ci-dessous. Ce pseudo-code à virgule fixe remplace le code à virgule fixe du bloc 17 original, figurant dans le G.3.9/G.728.

```

DISTM=2147483647
Pour J=65,66,...,NCWD, faire
  J1=(J-1)*IDIM
  AA0=0
  Pour K=1,2,...,IDIM, faire les deux lignes suivantes
    P=PN(K)*Y(J1+K)          | calculer le produit interne  $P_j$ 
    AA0=AA0+P                | NLS pour AA0 vaut 7+11=18
  Si AA0 < 0, alors affecter AA0=-AA0 | prendre la valeur absolue
  IDXG=1
  P=GB_128(1)*Y2(J)          | NLS pour P vaut 13+5=18

  Si AA0 ≥ P, alors affecter IDXG=IDXG+1

  AA0=AA0 >> 14              | NLS pour AA0=4
  Si AA0 > 32767, alors affecter |
  AA0=32767                  | écrêter AA0; AA0 est en mode de
                              | saturation
  AA1=GSQ_128(IDXG)*Y2(J)    | NLSGSQ_128=11, NLSY2=5, donc NLSAA1=16
  P=G2_128(IDXG)*AA0         | NLSG2_128=12, NLSAA0=4, donc NLSP=16
  AA1=AA1-P

```

```

Si AA1 < DISTM, alors faire les trois lignes suivantes
DISTM=AA1 | DISTM en double précision
IG=IDXG
IS=J
Répéter le module ci-dessus pour le J suivant

AA0=0 | trouver maintenant le bit de signe
J1=(IS-1)*IDIM
Pour K=1,2,...,IDIM, faire les deux lignes suivantes
P=PN(K)*Y(J1+K) | calculer le produit interne
AA0=AA0+P

Si AA0 ≤ 0, alors affecter IG=IG+2

IS1=NCWD
IS1=IS1 >> 1
IS1=IS-IS1

ICHAN=(IS1-1)*NG_128+(IG-1)

```

Dans le code ci-dessus, nous avons utilisé les quatre lignes suivantes:

```

AA0=AA0 >> 14 | NLS pour AA0=4
Si AA0 > 32767, alors affecter |
AA0=32767 | écrêter AA0
AA1=GSQ_128(IDXG)*Y2(J) | NLSGSQ_128=11, NLSY2=5, donc NLSAA1=16
P=G2_128(IDXG)*AA0 | NLSG2_128=12, NLSAA0=4, donc NLSP=16

```

Dans les puces de traitement DSP qui ont une fonction d'écrêtage, ces lignes peuvent être remplacées par le code suivant, qui donne exactement les mêmes résultats.

```

AA0=AA0 << 2 | NLS pour AA0=20
AA0=CLIP(AA0) | AA0 est en mode de saturation
AA0=AA0 >> 16 | prendre le mot le plus élevé; NLS pour
| AA0=4
AA1=GSQ_128(IDXG)*Y2(J) | NLSGSQ_128=11, NLSY2=5, donc NLSAA1=16
P=G2_128(IDXG)*AA0 | NLSG2_128=12, NLSAA0=4, donc NLSP=16

```

La fonction d'écrêtage (CLIP) et le mode de saturation se rapportent au principe de l'interdiction d'un débordement de la valeur AA0 lors de l'exécution de l'opération << 2. Au lieu de déborder, la valeur AA0 est mise au nombre positif ou négatif maximal, selon son signe d'origine. Dans ce cas, le nombre AA0 est toujours positif. Cette variante dépend du traitement DSP et peut nécessiter plus qu'un registre de 32 éléments binaires. La variante indiquée dans le pseudo-code principal peut toujours être implémentée.

H.3.1.2 Bloc 19 – Répertoire des vecteurs d'excitation quantifiés, et bloc 21 – Module de normalisation du gain

On trouvera dans le présent sous-paragraphe les deux pseudo-codes pour les blocs 19 et 21, en virgule flottante et en virgule fixe. Ces codes ont été modifiés pour le cas du fonctionnement à 12,8 kbit/s et doivent remplacer le bloc 19 original qui est décrit au 5.12/G.728. Voici la version en virgule flottante du pseudo-code pour le bloc 19, répertoire des vecteurs d'excitation quantifiés.

```

NN=(IS-1)*IDIM
Pour K=1,2,...,IDIM, faire la ligne suivante
YN(K)=GQ_128(IG)*Y(NN+K)

```

La version en virgule flottante du pseudo-code pour le bloc 21 (module de normalisation du gain) est donnée ci-dessous.

```

Pour K=1,2,...,IDIM, faire la ligne suivante
ET(K) =GAIN*YN(K)

```

La version en virgule fixe du même module est donnée ci-dessous. Elle remplace le code en virgule fixe original des blocs 19 et 21, figurant au G.3.10/G.728.

Pour le pseudo-code en virgule fixe, l'on combinera les blocs 19 et 21 afin de former un seul module. Les deux variables Y et GQ_128 ont des formats de type Q fixes, respectivement Q11 et Q13. La valeur de la variable GAIN est assortie de la variable NLSGAIN. Pour obtenir la précision maximale, le produit $GQ_{128}(IG)*GAIN$ est normalisé à 32 bits avant l'exécution de l'arrondissement aux 16 bits supérieurs. Soit $NNGQ_{128}(I)$ égal à (1 + le nombre de décalages à gauche nécessaires pour normaliser l'élément Q13 $GQ_{128}(I)$); donc $NNGQ_{128}(I)=3$ pour $I=1, 3$ et $NNGQ_{128}(I)=2$ pour $I=2, 4$. Le pseudo-code peut donc être écrit comme suit:

<pre>AA0=GQ_128(IG)*GAIN AA0=AA0 << NNGQ_128(IG)</pre>	<pre> AA0 possède NNGQ_128(IG) zéros en amorce décalons NNGQ_128(IG) bits afin de normaliser AA0</pre>
<pre>TMP=RND(AA0)</pre>	<pre> arrondissement aux 16 bits supérieurs et attribution au registre TMP</pre>
<pre>NLSAA0=13+NLSGAIN NLSTMP=NLSAA0+NNGQ_128(IG)-16</pre>	<pre> format Q du produit GQ_128(IG)*GAIN format Q de TMP, en raison du décalage à gauche du nombre AA0 de NNGQ_128(IG) bits, puis arrondissement et sélection des 16 bits supérieurs</pre>
<pre>NN=(IS-1)*IDIM</pre>	<pre> normalisation à 16 bits des vecteurs codes de forme sélectionnés</pre>
<pre>Call VSCALE(Y(NN+1), IDIM, IDIM, 14, TMP, NLS)</pre>	<pre> coder ces vecteurs sur 16 bits; et les mettre dans TEMP</pre>
<pre>Pour K=1,2,...,IDIM, faire les deux lignes suivantes</pre>	
<pre>AA0=TMP*TEMP(K)</pre>	<pre> TMP et TEMP sont tous deux normalisés à 16 bits, de façon que le produit ait 1 zéro d'amorce. L'arrondissement directement au mot élevé nous donne une table des ET sur 15 bits.</pre>
<pre>ET(K)=RND(AA0)</pre>	<pre> </pre>
<pre>NLSET=NLSTMP+11+NLS-16</pre>	<pre> calcul des NLS pour les ET.</pre>

H.3.1.3 Bloc 29 – Répertoire des vecteurs d'excitation quantifiés du décodeurs, et Bloc 31 – Module de normalisation du gain du décodeur

On trouvera dans le présent sous-paragraphe les deux pseudo-codes pour les blocs 29 et 31, en virgule flottante et en virgule fixe. Ces codes ont été modifiés pour le cas du fonctionnement à 12,8 kbit/s et doivent remplacer le bloc 29 original qui est décrit au 5.14/G.728. Voici la version en virgule flottante du pseudo-code pour le bloc 29, répertoire des vecteurs d'excitation quantifiés du décodeur.

Ce bloc extrait d'abord, du canal d'index à 8 bits reçu, les index du répertoire de gains codés sur 2 bits (IG) ainsi que les index du répertoire de formes codés sur 6 bits (IS). Le reste de l'opération est exactement comme pour le bloc 19 du codeur.

```
ITMP=partie entière de (ICHAN/NG_128)
IG=ICHAN-ITMP*NG_128+1
ITMP=ITMP+NCWD/2

NN=ITMP*IDIM
Pour K=1,2,...,IDIM, faire la ligne suivante
    YN(K)=GQ_128(IG)*Y(NN+K)
```

Le fonctionnement du bloc 31 (module de normalisation du gain du décodeur) est exactement le même que celui du bloc 21 pour le codeur.

La version en virgule fixe du même module est donnée ci-dessous:

pour le pseudo-code en virgule fixe, l'on combinera les blocs 29 et 31 afin de former un seul module. Les deux variables Y et GQ_128 ont des formats de type Q fixes, respectivement Q11 et Q13. La valeur de la variable GAIN est assortie de la variable NLSGAIN. Pour obtenir la précision maximale, le produit $GQ_{128}(IG) * GAIN$ est normalisé à 32 bits avant l'exécution de l'arrondissement aux 16 bits supérieurs. Soit $NNGQ_{128}(I)$ égal à (1 + le nombre de décalages à gauche nécessaires pour normaliser l'élément Q13 $GQ_{128}(I)$); donc $NNGQ_{128}(I)=3$ pour $I=1, 3$ et $NNGQ_{128}(I)=2$ pour $I=2, 4$. Le pseudo-code peut donc être écrit comme suit:

```

IS=ICHAN >> 2
IG=ICHAN-IS*NG_128+1
IS1=NCWD
IS1=IS1 >> 1
IS=IS+IS1+1

AA0=GQ_128(IG)*GAIN          | AA0 possède NNGQ_128(IG) zéros d'amorce
AA0=AA0 << NNGQ_128(IG)      | décalage à gauche de NNGQ_128(IG) bits
                               | pour normaliser AA0
TMP=RND(AA0)                  | arrondissement aux 16 bits supérieurs et
                               | attribution à TMP

NLSAA0=13+NLSGAIN             | formatage Q du produit GQ_128(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_128(IG)-16 | formatage Q de TMP en raison du décalage
                               | à gauche du nombre AA0 de NNGQ_128(IG)
                               | bits puis arrondissement et sélection des
                               | 16 bits supérieurs
NN=(IS-1)*IDIM                | normalisation à 16 bits
Call VSCALE(Y(NN+1), IDIM, IDIM, 14 ,TEMP, NLS) | des vecteurs codes de
                               | forme sélectionnés;
                               | et les mettre dans TEMP

Pour K=1,2,...,IDIM, faire les deux lignes suivantes
AA0=TMP*TEMP(K)                | TMP et TEMP sont tous deux normalisés à
ET(K)=RND(AA0)                 | 16 bits, de façon que le produit ait
                               | 1 zéro d'amorce. L'arrondissement
                               | directement au mot élevé nous donne une
                               | table des ET sur 15 bits.
NLSET=NLSTMP+11+NLS-16        | calcul des NLS pour les ET.

```

H.3.1.4 Blocs 96 et 97 – Additionneur et limiteur de termes de correction de gain logarithmique à -32 dB

Le présent sous-paragraphe donne les pseudo-codes en virgule fixe et en virgule flottante pour les blocs 96 et 97. Ces codes ont été modifiés pour un fonctionnement à 12,8 kbit/s et doivent remplacer les blocs 96 et 97 d'origine décrits au G.3.16/G.728.

Le pseudo-code en virgule flottante est le suivant:

```

GSTATE(1) = LOGGAIN + GCBLG_128(IG) + SHAPELG(IS)
If GSTATE(1) < -32.0, set GSTATE(1) = -32.0

```

Le pseudo-code en virgule fixe est le suivant:

```

AA0=LOGGAIN << 7              | Aligner la virgule au niveau de la
AA0=AA0 + (GCBLG_128(IG) << 5) | frontière entre les mots de poids
AA0=AA0 + (SHAPELG(IS) << 5)   | fort et de poids faible de
                               | l'accumulateur.
AA0=AA0 >> 7                    | Décalage à droite pour revenir au
                               | format Q9

```

If AA0 < -16384, set AA0=-16384	Vérifier la limite inférieure
GSTATE(1)=AA0	Le mot de 16 bits de poids faible est sauvegardé

H.3.2 Nouvelles tables de gains additionnels

On trouvera dans le présent sous-paragraphe les valeurs du répertoire de gains pour le fonctionnement à 12,8 kbit/s. Les valeurs en virgule flottante sont données en premier. Voir Tableau H.1.

Tableau H.1/G.728 – Valeurs en virgule flottante des tables relatives aux répertoires de gains

Index de table	1	2	3	4
GQ_128	0,525824	1,562449	-0,525824	-1,562449
GB_128	0,869912	*	-0,869912	*
G2_128	1,051648	3,124898	-1,051648	-3,124898
GSQ_128	0,276491	2,441247	0,276491	2,441247
GCBLG_128	-5,5831919	3,8761170	-5,5831919	3,8761170

Les valeurs en virgule fixe sont données ci-après. Voir Tableau H.2.

Tableau H.2/G.728 – Valeurs en virgule fixe des tables relatives aux répertoires de gains

Index de table	1	2	3	4
GQ_128(Q13)	4 308	12 800	-4 308	-12 800
GB_128(Q13)	7 126	*	-7 126	*
G2_128(Q12)	4 308	12 800	-4 308	-12 800
GSQ_128(Q11)	566	5 000	566	5 000
NNGQ_128(Q0)	3	2	3	2
GCBLG_128(Q11)	-11 434	7 938	-11 434	7 938

H.3.3 Modification du paramètre de codeur

Le présent sous-paragraphe décrit le nouveau paramètre NG_128. Ce paramètre a été dérivé du paramètre NG (valeur = 8) de la Recommandation G.728, pour l'adapter au fonctionnement à 12,8 kbit/s. Voir Tableau H.3.

Tableau H.3/G.728 – Paramètre de base du codeur LD-CELP

Nom	Valeur	Description
NG_128	4	Dimension du répertoire de gains (nombre de niveaux de gain)

H.4 Modifications pour le fonctionnement à 9,6 kbit/s

H.4.1 Pseudo-code

Seules les séquences d'exécution par blocs sont représentées; les détails de couche inférieure pour la transmission des paramètres ne sont pas décrits.

H.4.1.1 Blocs 17 et 18 – Calculateur d'erreur et sélecteur du meilleur index du répertoire

On trouvera dans le présent sous-paragraphe les pseudo-codes aussi bien en virgule flottante qu'en virgule fixe pour les blocs 17 et 18. Ces codes ont été modifiés pour le fonctionnement à 9,6 kbit/s et doivent remplacer les blocs 17 et 18 originaux qui figurent au 5.11/G.728. Le pseudo-code en virgule flottante est représenté en premier.

```
Initialiser DISTM au plus grand entier représentable par la machine
N1=NG_96/2
Pour K=1,2,3,4, faire
  Pour K1=97,98,99,100, faire
    J=(K-1)*8+K1
    J1=(J-1)*IDIM
    COR=0.
    Pour K2=1,2,...,IDIM, alors faire la ligne suivante
      COR=COR+PN(K2)*Y(J1+K2)      | calculer le produit interne Pj

    Si COR > 0, alors faire les trois lignes suivantes
      IDXG=N1
      Si COR < GB_96(1)*Y2(J), alors faire la ligne suivante
        IDXG=1                      | meilleur gain positif trouvé

    Si COR ≤ 0, alors faire les trois lignes suivantes
      IDXG=NG_96
      Si COR > GB_96(3)*Y2(J), alors faire la ligne suivante
        IDXG=3                      | meilleur gain négatif trouvé

    D=-G2_96(IDXG)*COR+GSQ_96(IDXG)*Y2(J) | calculer la distorsion  $\hat{D}$ 
    Si D < DISTM, alors faire les trois lignes suivantes
      DISTM=D                       | sauvegarde de la plus faible
      IG=IDXG                       | distorsion et des meilleurs index
      IS=J                           | du répertoire trouvé jusqu'à maintenant

  Répéter le module ci-dessus pour le K1 suivant.

Répéter le module ci-dessus pour le K suivant.

IS1=IS-(NCWD/2+NCWD/4)
IS2= partie entière de (IS1/8)
IS2=IS2*4
IS3=IS1-IS2*2
IS1=IS2+IS3
ICHAN=(IS1-1)*NG_96+(IG-1)      | concaténer les index des
                                | répertoires de formes et de gains.
```

Transmettre ICHAN par le canal de communication. Dans le cas d'une transmission en série, le bit le plus significatif de l'élément ICHAN sera transmis en premier. Si ICHAN est représenté par le mot de 6 bits b5, b4, b3, b2, b1, b0, alors l'ordre de transmission des bits sera b5, puis b4, b3, b2, b1, b0 (b5 étant le bit le plus significatif).

La version en virgule fixe du même module est reproduite ci-dessous. Ce pseudo-code à virgule fixe remplace le code à virgule fixe du bloc 17 original, figurant au G.3.9/G.728.

```

DISTM=2147483647
Pour K=1,2,3,4, faire
  Pour K1=97,98,99,100, faire
    J=(K-1)*8+K1
    J1=(J-1)*IDIM
    AA0=0
    Pour K2=1,2,...,IDIM, faire les deux lignes suivantes
      P=PN(K2)*Y(J1+K2) | calculer le produit interne Pj
      AA0=AA0+P | NLS pour AA0 vaut 7+11=18
      Si AA0 < 0, alors affecter |
      AA0=-AA0 | prendre la valeur absolue
      IDXG=1
      P=GB_96(1)*Y2(J) | NLS pour P vaut 13+5=18
      Si AA0 ≥ P, alors affecter |
      IDXG=IDXG+1
      AA0=AA0 >> 14 | NLS pour AA0=4
      Si AA0 > 32767, alors affecter |
      AA0=32767 | écrêter AA0; AA0 est en mode saturation
      AA1=GSQ_96(IDXG)*Y2(J) | NLSGSQ_96=11, NLSY2=5, donc NLSAA1=16
      P=G2_96(IDXG)*AA0 | NLSG2_96=12, NLSAA0=4, donc NLSP=16
      AA1=AA1-P

      Si AA1 < DISTM, alors faire les trois lignes suivantes
        DISTM=AA1 | DISTM en double précision
        IG=IDXG
        IS=J

```

Répéter le module ci-dessus pour le K1 suivant.

Répéter le module ci-dessus pour le K suivant.

```

AA0=0 | trouver maintenant le bit de signe
J1=(IS-1)*IDIM
Pour K=1,2,..., IDIM, faire les deux lignes suivantes
  P=PN(K)*Y(J1+K) | calculer le produit interne
  AA0=AA0+P
Si AA0 ≤ 0, alors affecter IG=IG+2
IS2=NCWD
IS1=IS2 >> 1
IS2=IS2 >> 2
IS1=IS-(IS1+IS2)
IS2=IS1 >> 3
IS2=IS2 << 2
IS3=IS2 << 1
IS3=IS1-IS3
IS1=IS2+IS3
ICHAN=(IS1-1)*NG_96+(IG-1)

```

Dans le code ci-dessus, nous avons utilisé les quatre lignes suivantes:

```

AA0=AA0 >> 14 | NLS pour AA0=4
If AA0 > 32767, set AA0=32767 | écrêter AA0
AA1=GSQ_96(IDXG)*Y2(J) | NLSGSQ_96=11, NLSY2=5, donc NLSAA1=16
P=G2_96(IDXG)*AA0 | NLSG2_96=12, NLSAA0=4, donc NLSP=16

```

Dans les puces de traitement DSP qui ont une fonction d'écrêtage, ces lignes peuvent être remplacées par le code suivant, qui donne exactement les mêmes résultats:

AA0=AA0 << 2	NLS pour AA0=20
AA0=CLIP(AA0)	AA0 est en mode de saturation
AA0=AA0 >> 16	prendre le mot le plus élevé;
	NLS pour AA0=4
AA1=GSQ_96(IDXG)*Y2(J)	NLSGSQ_96=11, NLSY2=5, donc NLSAA1=16
P=G2_96(IDXG)*AA0	NLSG2_96=12, NLSAA0=4, donc NLSP=16

La fonction d'écrêtage (CLIP) et le mode de saturation se rapportent au principe de l'interdiction d'un débordement de la valeur AA0 lors de l'exécution de l'opération << 2. Au lieu de déborder, la valeur AA0 est mise au nombre positif ou négatif maximal, selon son signe d'origine. Dans ce cas, le nombre AA0 est toujours positif. Cette variante dépend du traitement DSP et peut nécessiter plus qu'un registre de 32 éléments binaires. La variante indiquée dans le pseudo-code principal peut toujours être implémentée.

H.4.1.2 Bloc 19 – Répertoire des vecteurs d'excitation quantifiés, et bloc 21 – Module de normalisation du gain

On trouvera dans le présent sous-paragraphe les deux pseudo-codes pour les blocs 19 et 21, en virgule flottante et en virgule fixe. Ces codes ont été modifiés pour le cas du fonctionnement à 9,6 kbit/s et doivent remplacer le bloc 19 original qui est décrit au 5.12/G.728. Voici la version en virgule flottante du pseudo-code pour le bloc 19, répertoire des vecteurs d'excitation quantifiés. La version en virgule flottante du pseudo-code pour le bloc 19 est présentée en premier.

```

NN=(IS-1)*IDIM
Pour K=1,2,..., IDIM, faire la ligne suivante
  YN(K)=GQ_96(IG)*Y(NN+K)

```

La version en virgule flottante du pseudo-code pour le bloc 21 (module de normalisation du gain) est donnée ci-dessous.

```

Pour K=1,2,..., IDIM, faire la ligne suivante
  ET(K)=GAIN*YN(K)

```

La version en virgule fixe du même module est donnée ci-dessous. Elle remplace le code en virgule fixe original des blocs 19 et 21, figurant au G.3.10/G.728.

Pour le pseudo-code en virgule fixe, l'on combinera les blocs 19 et 21 afin de former un seul module. Les deux variables Y et GQ_96 ont des formats de type Q fixes, respectivement Q11 et Q13. La valeur de la variable GAIN est assortie de la variable NLSGAIN. Pour obtenir la précision maximale, le produit GQ_96(IG)*GAIN est normalisé à 32 bits avant l'exécution de l'arrondissement aux 16 bits supérieurs. Soit NNGQ_96(I) égal à (1 + le nombre de décalages à gauche nécessaires pour normaliser l'élément Q13 GQ_96(I)); donc NNGQ_96(I)=3 pour I=1, 3 et NNGQ_96(I)=2 pour I=2, 4. Le pseudo-code peut donc être écrit comme suit:

AA0=GQ_96(IG)*GAIN	AA0 possède NNGQ_96(IG) zéros en amorce
AA0=AA0 << NNGQ_96(IG)	décalons NNGQ_96(IG) bits afin de
	normaliser AA0
TMP=RND(AA0)	arrondissement aux 16 bits supérieurs et
	attribution au registre TMP
NLSAA0=13+NLSGAIN	format Q du produit GQ_96(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_96(IG)-16	format Q de TMP, en raison du décalage à
	gauche du nombre AA0 de NNGQ_96(IG)
	bits, puis arrondissement et sélection
	des 16 bits supérieurs
NN=(IS-1)*IDIM	normalisation à 16 bits des vecteurs
	codes de forme sélectionnés

Call VSCALE(Y(NN+1), IDIM, IDIM, 14, TEMP, NLS)	coder ces vecteurs sur
	16 bits; et les mettre
	dans TEMP
Pour K=1,2,..., IDIM, faire les deux lignes suivantes	
AA0=TMP*TEMP(K)	TMP et TEMP sont tous deux normalisés à
	16 bits, de façon que le produit ait
ET(K) =RND(AA0)	1 zéro d'amorce.
	L'arrondissement directement au mot élevé
	nous donne une table des ET sur 15 bits.
NLSET=NLSTMP+11+NLS-16	calcul des NLS pour les ET.

H.4.1.3 Bloc 29 – Répertoire des vecteurs d'excitation quantifiés du décodeur, et bloc 31 – Module de normalisation du gain du décodeur

On trouvera dans le sous-paragraphe les deux pseudo-codes pour les blocs 29 et 31, en virgule flottante et en virgule fixe. Ces codes ont été modifiés pour le cas du fonctionnement à 9,6 kbit/s et doivent remplacer le bloc 29 original qui est décrit au 5.14/G.728. Voici la version en virgule flottante du pseudo-code pour le bloc 29, répertoire des vecteurs d'excitation quantifiés du décodeur.

Ce bloc extrait d'abord, du canal d'index à 6 bits reçu, les index du répertoire de gains codés sur 2 bits (IG) ainsi que les index du répertoire de formes codés sur 4 bits (IS). Le reste de l'opération est exactement comme pour les blocs 19 et 21 du codeur.

```

ITMP=partie entière de (ICHAN/NG_96)
IG=ICHAN-ITMP*NG_96+1
ITMP1=integer part of (ITMP/4)
ITMP2=ITMP-ITMP1*4
ITMP1=ITMP1*8
ITMP=ITMP1+ITMP2
ITMP=ITMP+(NCWD/2+NCWD/4)

NN=ITMP*IDIM
Pour K=1,2,...,IDIM, faire la ligne suivante
  YN(K)=GQ_96(IG)*Y(NN+K)

```

Le fonctionnement du bloc 31 (module de normalisation du gain du décodeur) est exactement le même que celui du bloc 21 pour le codeur.

La version en virgule fixe du même module est donnée ci-dessous.

Pour le pseudo-code en virgule fixe, l'on combinera les blocs 29 et 31 afin de former un seul module. Les deux variables Y et GQ_96 ont des formats de type Q fixes, respectivement Q11 et Q13. La valeur de la variable GAIN est assortie de la variable NLSGAIN. Pour obtenir la précision maximale, le produit GQ_96(IG)*GAIN est normalisé à 32 bits avant l'exécution de l'arrondissement aux 16 bits supérieurs. Soit NNGQ_96(I) égal à (1 + le nombre de décalages à gauche nécessaires pour normaliser l'élément Q13 GQ_96(I)); donc NNGQ_96(I)=3 pour I=1, 3 et NNGQ_96(I)=2 pour I=2, 4. Le pseudo-code peut donc être écrit comme suit:

```

IS=ICHAN >> 2
IG=ICHAN-IS*NG_96+1
IS1=IS >> 2
IS2=IS-IS1*4
IS1=IS1 << 3
IS=IS1+IS2
IS2=NCWD
IS1=IS2 >> 1
IS2=IS2 >> 2
IS=IS+IS1+IS2+1

```

AA0=GQ_96(IG)*GAIN	AA0 possède NNGQ_96(IG) zéros d'amorce
AA0=AA0 << NNGQ_96(IG)	décalage à gauche de NNGQ_96(IG) bits
	pour normaliser AA0
TMP=RND(AA0)	arrondissement aux 16 bits supérieurs et
	attribution à TMP
NLSAA0=13+NLSGAIN	formatage Q du produit GQ_96(IG)*GAIN
NLSTMP=NLSAA0+NNGQ_96(IG)-16	formatage Q de TMP en raison du décalage
	à gauche du nombre AA0 de NNGQ_96(IG)
	bits puis arrondissement et sélection des
	16 bits supérieurs
NN=(IS-1)*IDIM	normalisation à 16 bits des vecteurs
	codes de forme sélectionnés
Call VSCALE(Y(NN+1), IDIM, IDIM, 14 ,TEMP, NLS)	coder ces vecteurs à
	16 bits; et les mettre
	dans TEMP
Pour K=1,2,...,IDIM, faire les deux lignes suivantes	
AA0=TMP*TEMP(K)	TMP et TEMP sont tous deux normalisés à
ET(K)=RND(AA0)	16 bits, de façon que le produit ait
	1 zéro d'amorce.
	L'arrondissement directement au mot élevé
	nous donne une table des ET sur 15 bits.
NLSET=NLSTMP+11+NLS-16	calcul des NLS pour les ET.

H.4.1.4 Blocs 96 et 97 – Additionneur et limiteur de termes de correction de gain logarithmique à -32 dB

Le présent sous-paragraphe donne les pseudo-codes en virgule fixe et en virgule flottante pour les blocs 96 et 97. Ces codes ont été modifiés pour un fonctionnement à 9,6 kbit/s et doivent remplacer les blocs 96 et 97 d'origine décrits au G.3.16/G.728.

Le pseudo-code en virgule flottante est le suivant:

```
GSTATE(1) = LOGGAIN + GCBLG_96(IG) + SHAPELG(IS)
If GSTATE(1) < -32.0, set GSTATE(1) = -32.0
```

Le pseudo-code en virgule fixe est le suivant:

AA0=LOGGAIN << 7	Aligner la virgule au niveau de la
AA0=AA0 + (GCBLG_96(IG) << 5)	frontière entre les mots de poids
AA0=AA0 + (SHAPELG(IS) << 5)	fort et de poids faible de
	l'accumulateur.
AA0=AA0 >> 7	Décalage à droite pour revenir au
	format Q9
If AA0 < -16384, set AA0=-16384	Vérifier la limite inférieure
GSTATE(1)=AA0	le mot de 16 bits de poids faible
	est sauvegardé

H.4.2 Nouvelles tables de gains additionnels

On trouvera dans le présent sous-paragraphe les valeurs du répertoire de gains pour le fonctionnement à 9,6 kbit/s. Les valeurs en virgule flottante sont données en premier. Voir Tableau H.4.

Tableau H.4/G.728 – Valeurs en virgule flottante des tables relatives aux répertoires de gains

Index du tableau	1	2	3	4
GQ_96	0,564657	1,937714	-0,564657	-1,937714
GB_96	1,007492	*	-1,007492	*
G2_96	1,129314	3,875428	-1,129314	-3,875428
GSQ_96	0,318838	3,754736	0,318838	3,754736
GCBLG_96	-4,9643057	5,7457935	-4,9643057	5,7457935

Les valeurs en virgule fixe sont données ci-après. Voir Tableau H.5.

Tableau H.5/G.728 – Valeurs en virgule fixe des tables relatives aux répertoires de gains

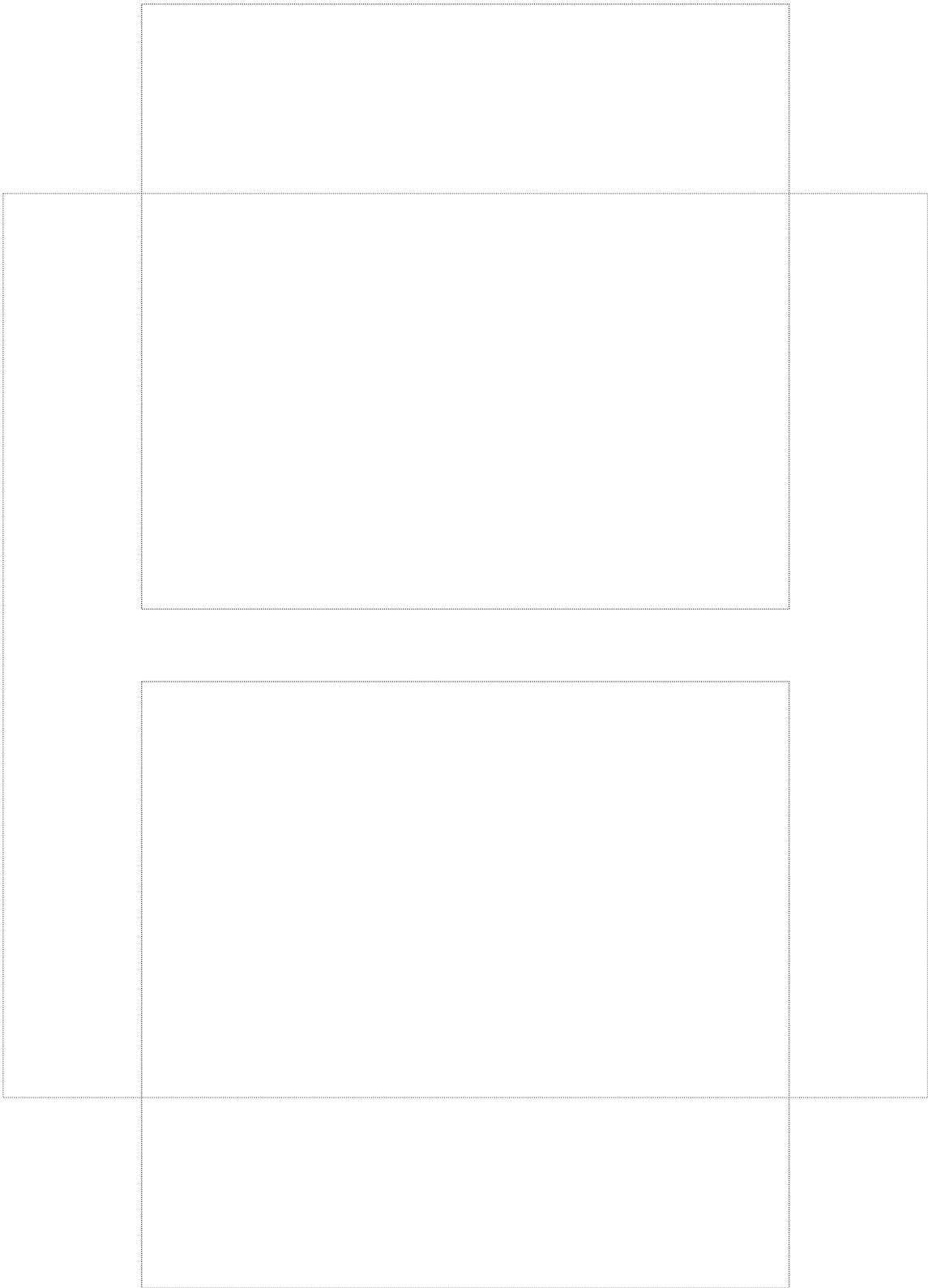
Index du tableau	1	2	3	4
GQ_96(Q13)	4 626	15 874	-4 626	-15 874
GB_96(Q13)	8 253	*	-8 253	*
G2_96(Q12)	4 626	15 874	-4 626	-15 874
GSQ_96(Q11)	653	7 690	653	7 690
NNGQ_96(Q0)	3	2	3	2
GCBLG_96(Q11)	-10 167	11 767	-10 167	11 767

H.4.3 Modification du paramètre de codeur

Le présent sous-paragraphe décrit le nouveau paramètre NG_96 dérivé du paramètre NG valeur 8 de la Recommandation G.728, pour l'adapter au fonctionnement à 9,6 kbit/s. Voir Tableau H.6.

Tableau H.6/G.728 – Paramètre de base du codeur LD-CELP

Nom	Valeur	Description
NG_96	4	Dimension du répertoire de gains (nombre de niveaux de gain)



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
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
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