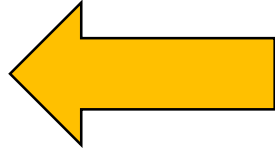


第5回
サウンドメディア論
および演習 講義編

今日やること

- 音を揃える

- リミッタ
- コンプレッサ



- 音を揺らす

- トレモロ
- ビブラート

- 音を広げる

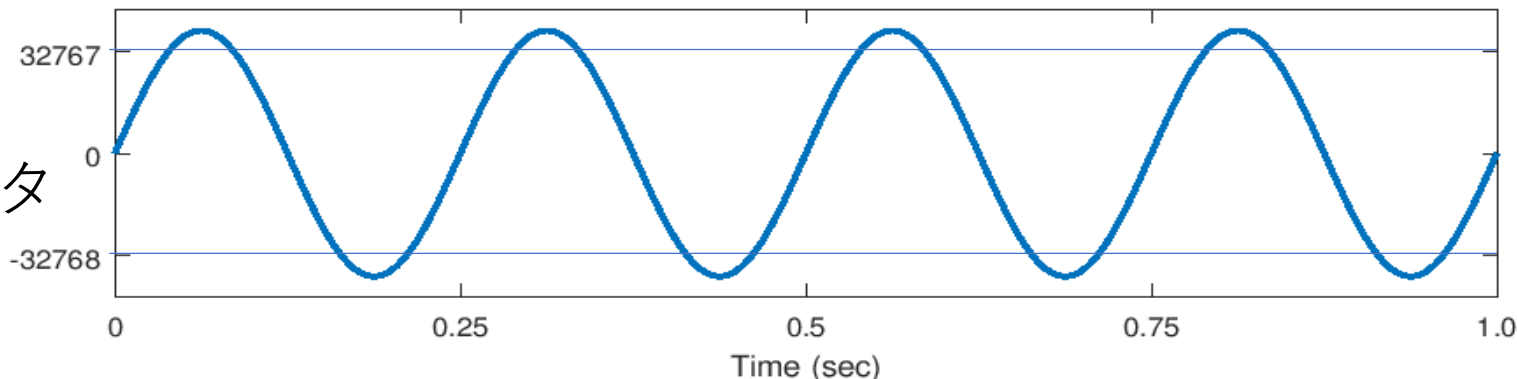
- コーラス
- フランジャ

リミッタとは

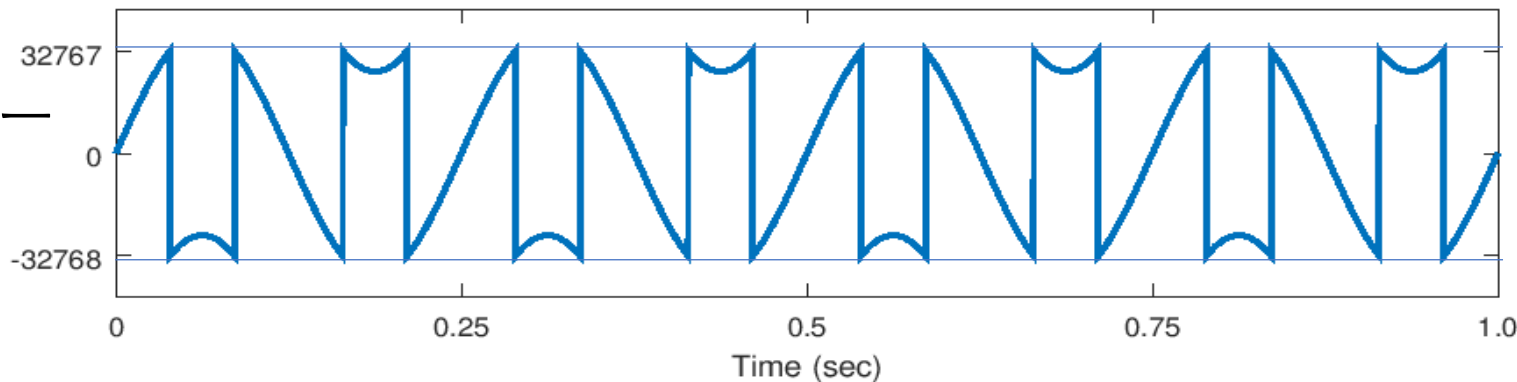
- 振幅の上限・下限内に音データを収める

⇒ 正の最大値を上回ると負の値に逆転、負の最小値を下回ると正の値に逆転(オーバーフロー)

本来の
音データ

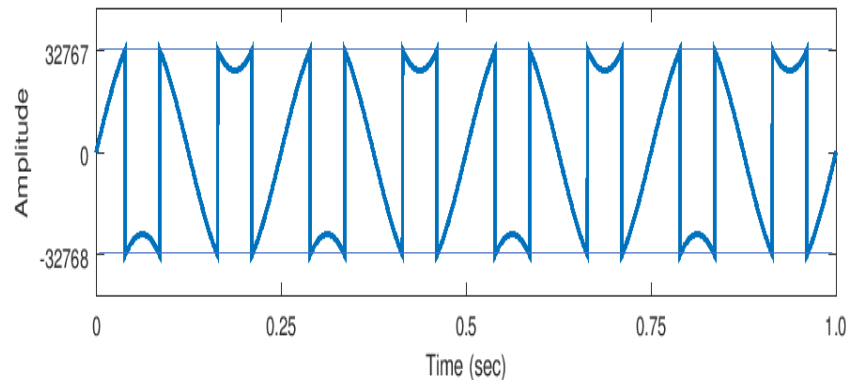
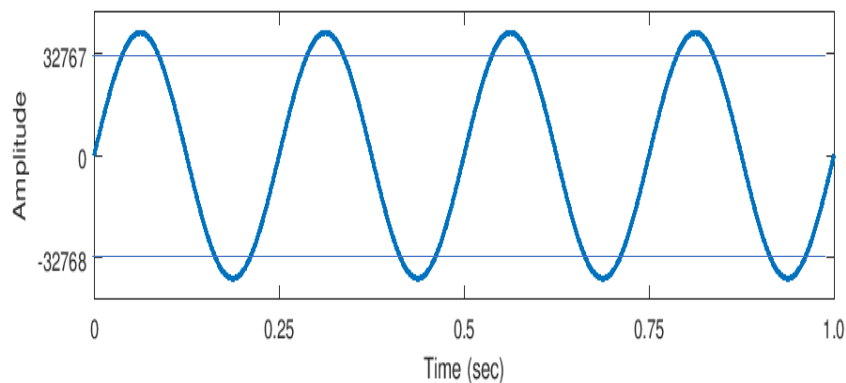


オーバー
フロー



リミッタとは

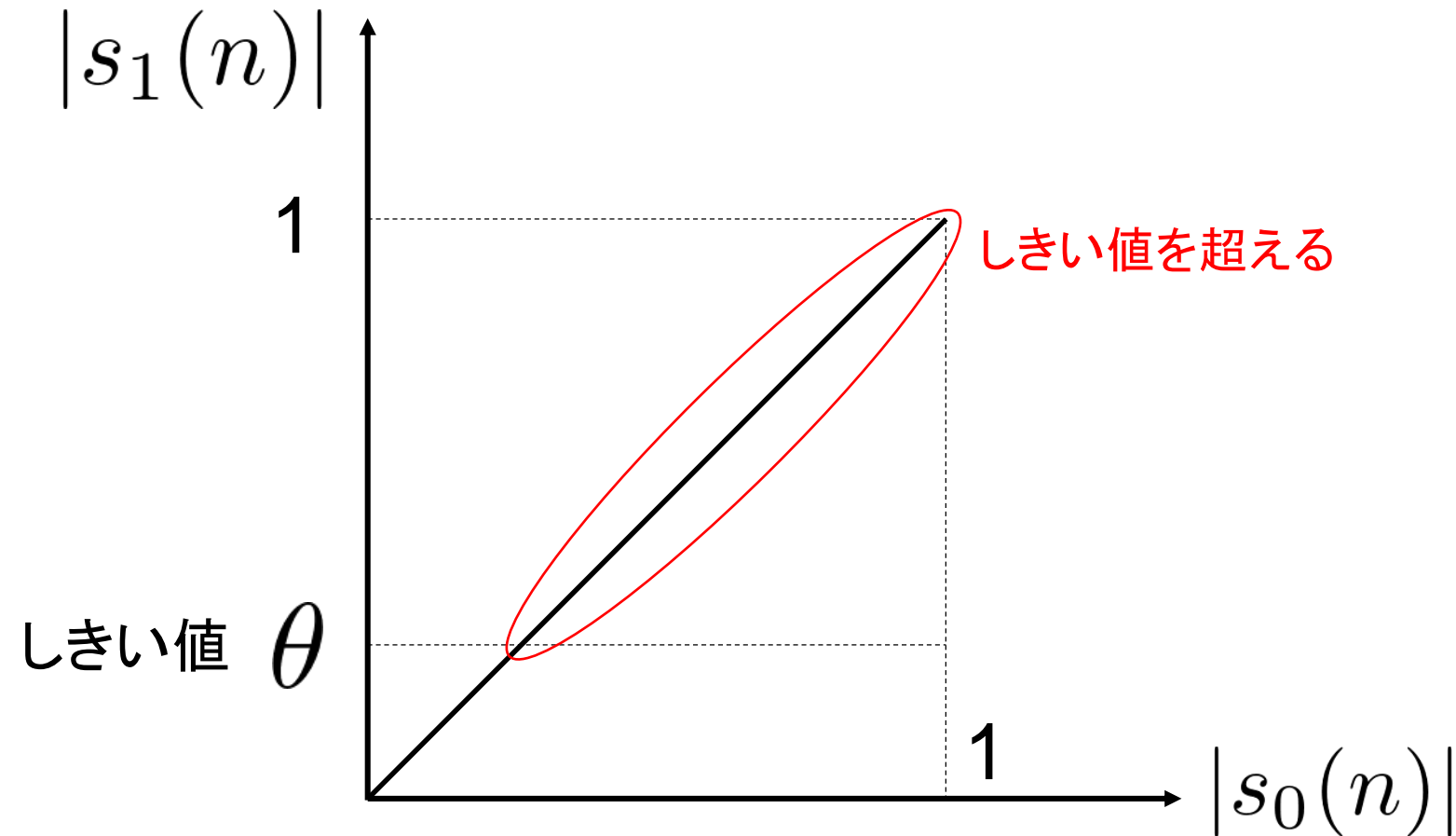
- 振幅の上限・下限内に音データを収める
⇒ 正の最大値を上回ると負の値に逆転、負の最小値を下回ると正の値に逆転(**オーバーフロー**)



- リミッタは**オーバーフローの対策の1つ**
決められた振幅の範囲の中に音データを収める処理

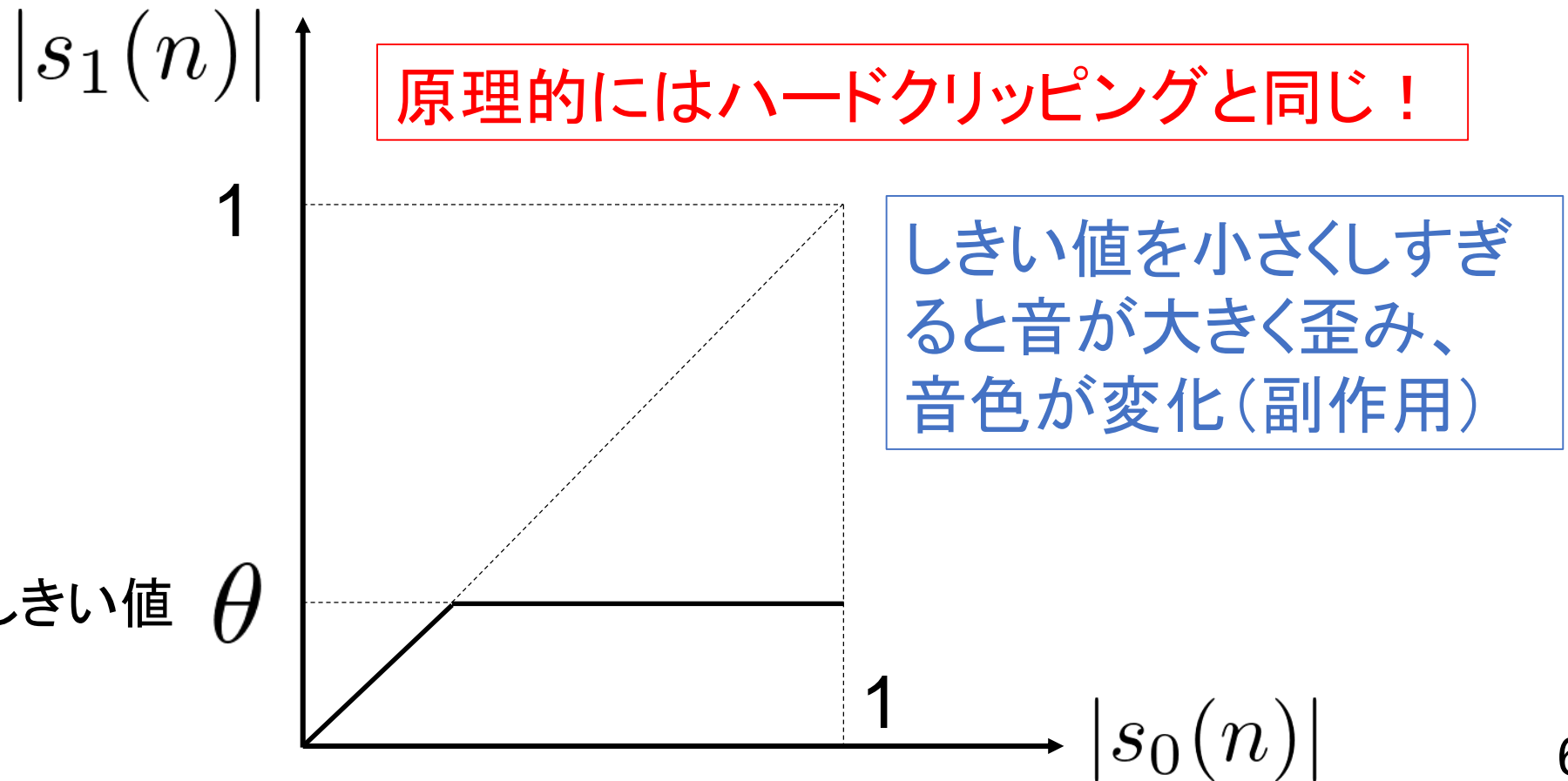
リミッタの仕組み

入力信号の振幅がしきい値よりも大きい場合、
出力信号の振幅を切りそろえる



リミッタの仕組み

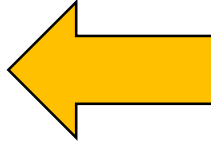
入力信号の振幅がしきい値よりも大きい場合、
出力信号の振幅を切りそろえる



今日やること

- 音を揃える

- リミッタ
- コンプレッサ



- 音を揺らす

- トレモロ
- ビブラート

- 音を広げる

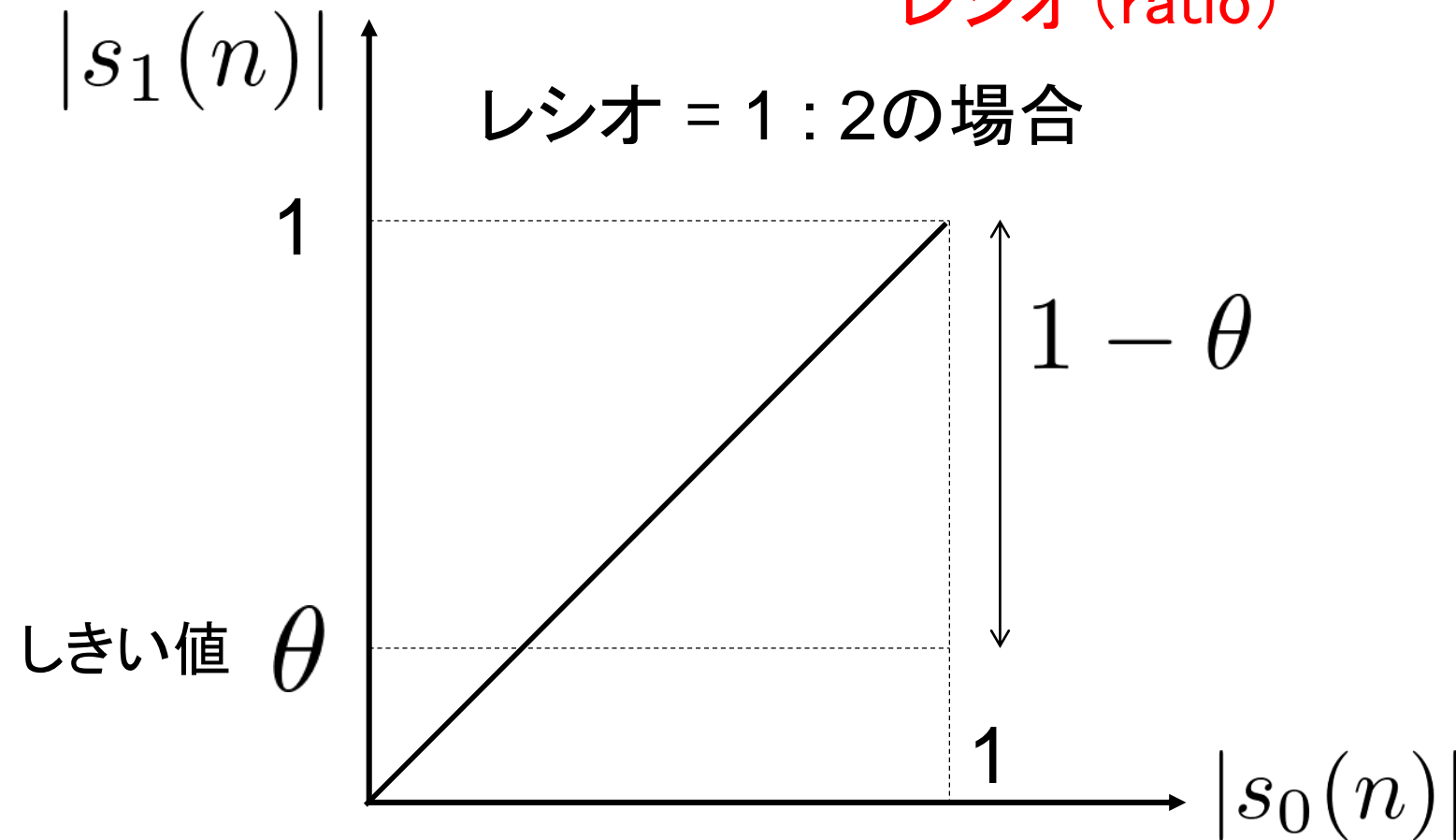
- コーラス
- フランジャ

コンプレッサとは？

- 振幅の切り揃えによる音色変化を音作りに利用
- しきい値を超えた振幅を規定の割合で圧縮

レシオ (ratio)

レシオ = 1 : 2 の場合

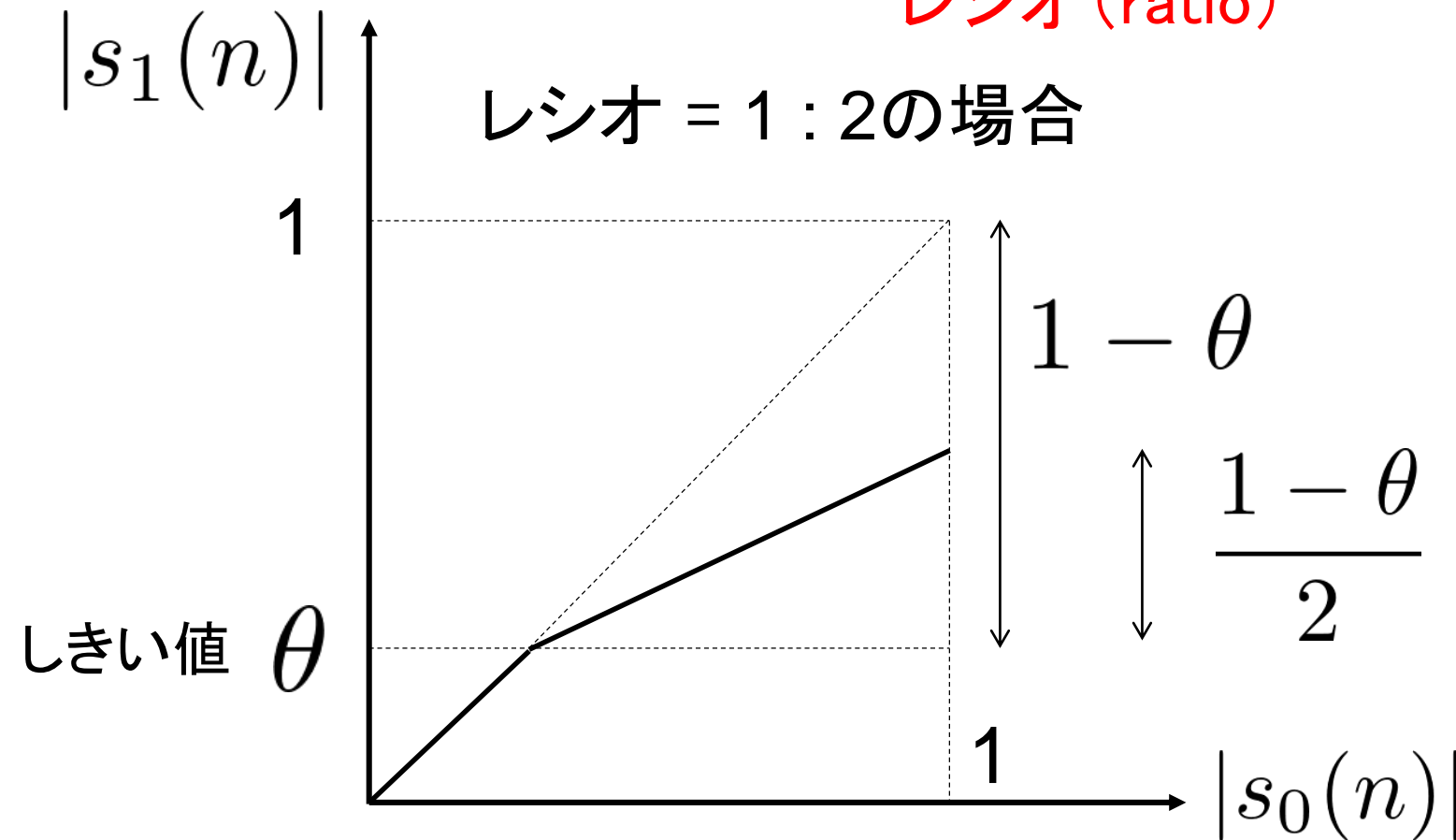


コンプレッサとは？

- 振幅の切り揃えによる音色変化を音作りに利用
- しきい値を超えた振幅を規定の割合で圧縮

レシオ (ratio)

レシオ = 1 : 2 の場合

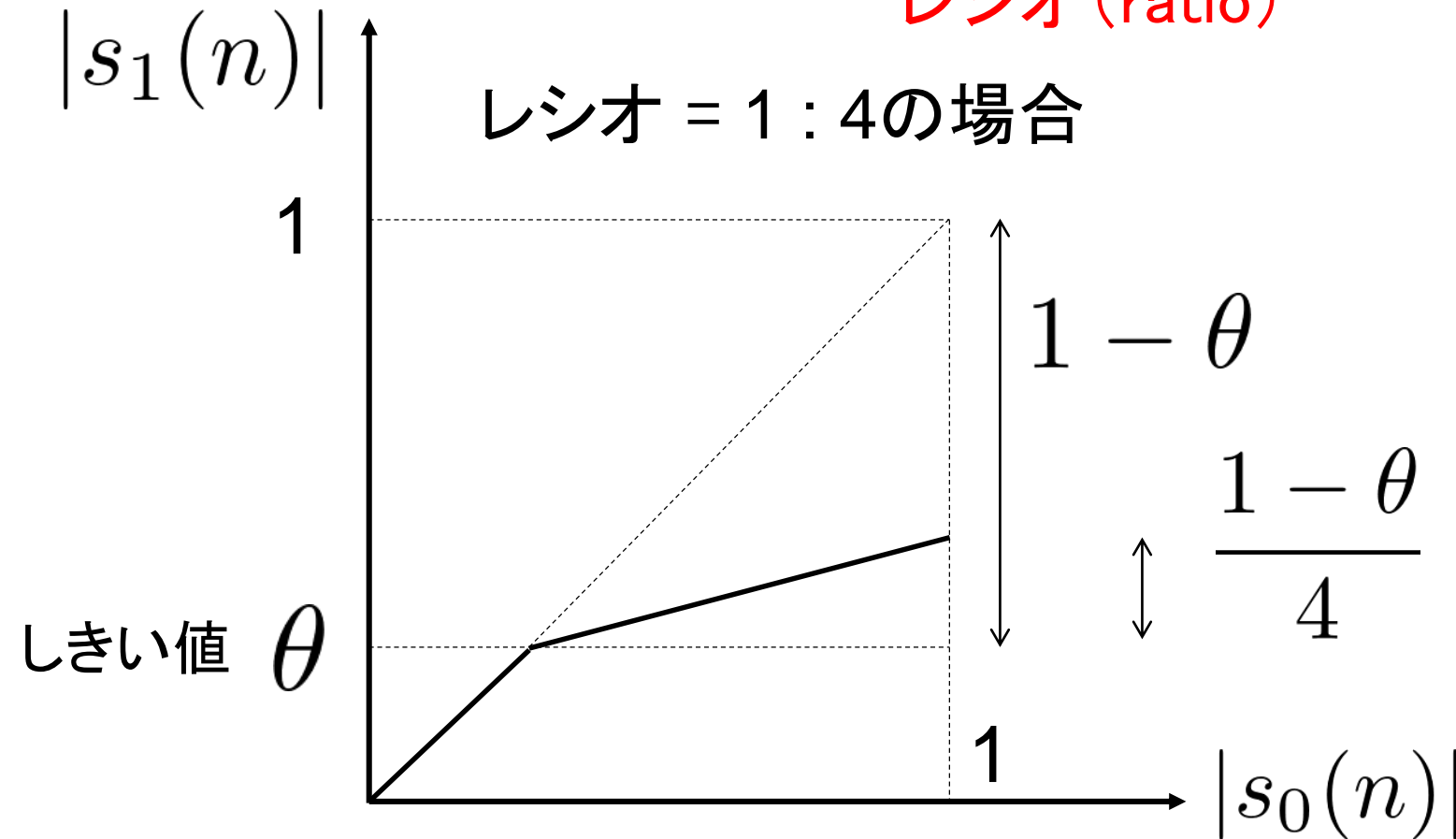


コンプレッサとは？

- 振幅の切り揃えによる音色変化を音作りに利用
- しきい値を超えた振幅を規定の割合で圧縮

レシオ (ratio)

レシオ = 1 : 4 の場合

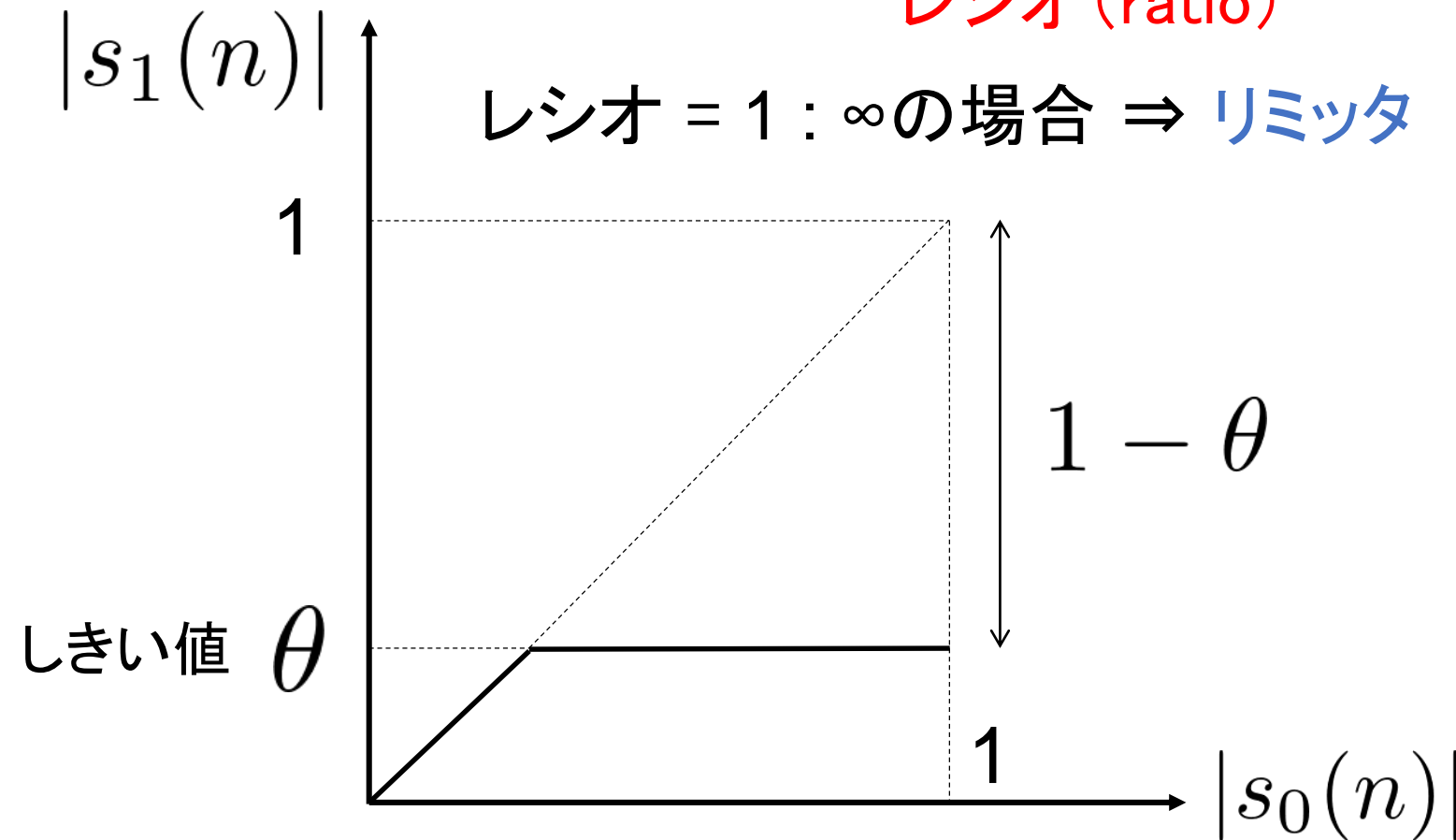


コンプレッサとは？

- 振幅の切り揃えによる音色変化を音作りに利用
- しきい値を超えた振幅を規定の割合で圧縮

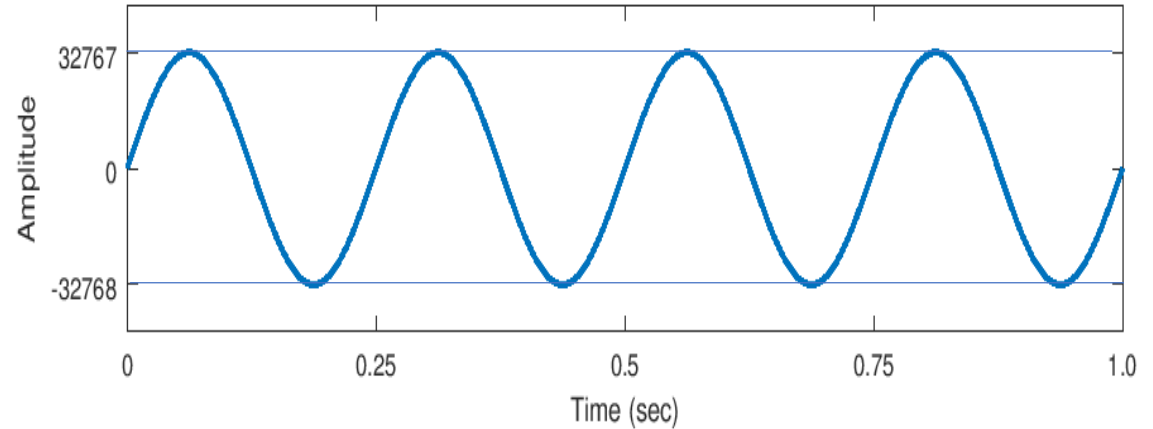
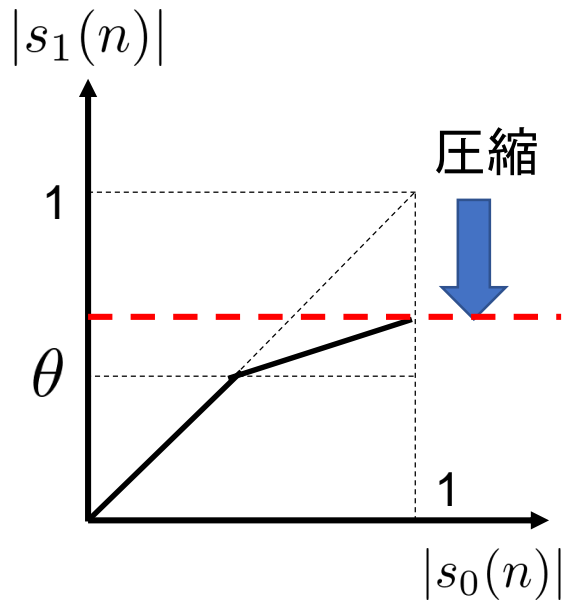
レシオ (ratio)

レシオ = $1 : \infty$ の場合 \Rightarrow リミッタ

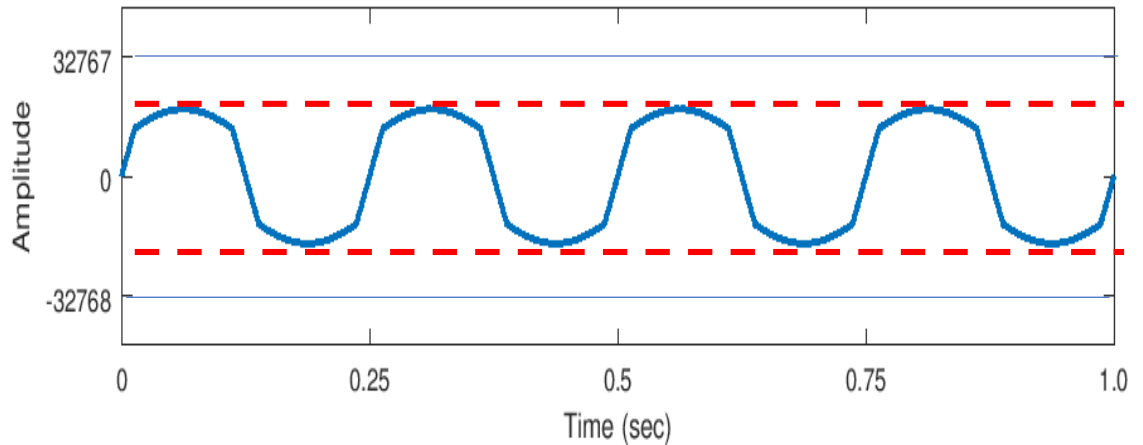


コンプレッサによる振幅の増幅(1/3)

- 圧縮の例

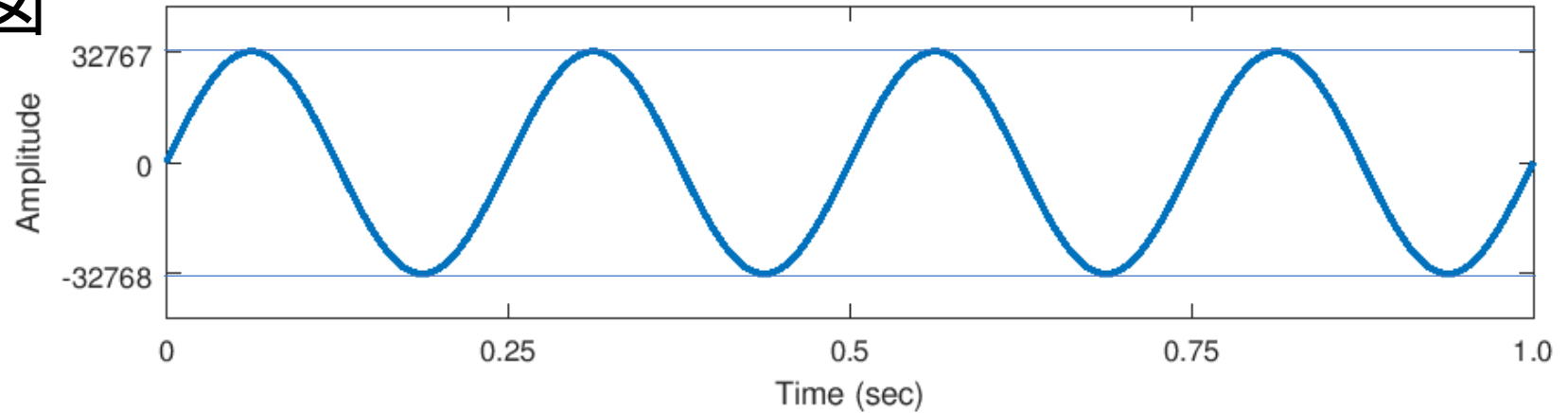


圧縮



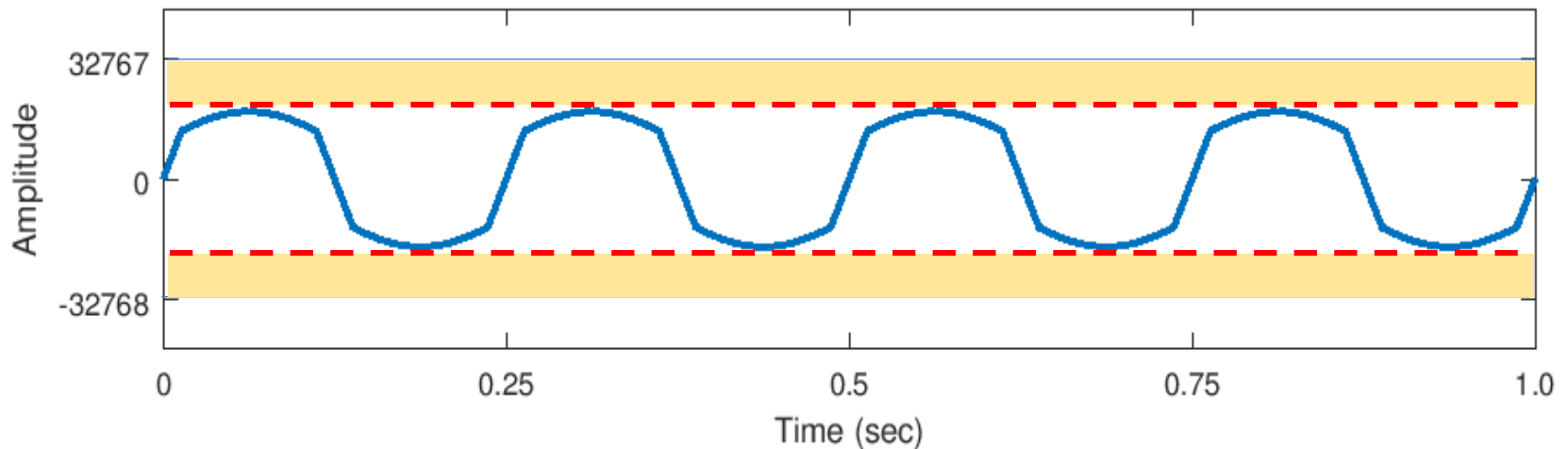
コンプレッサによる振幅の増幅(1/3)

拡大図



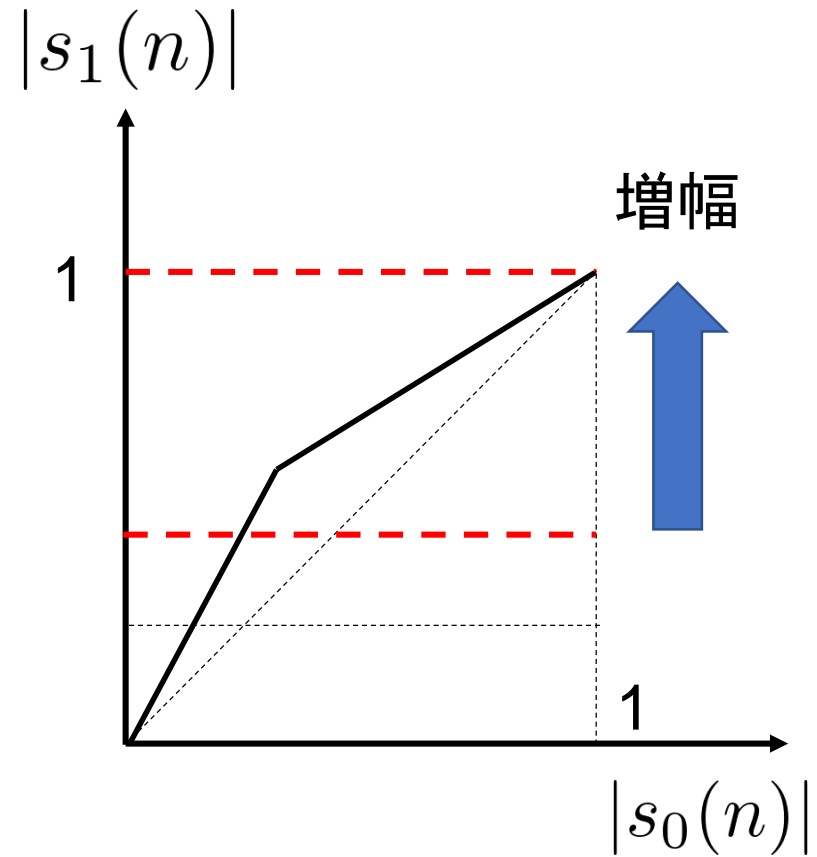
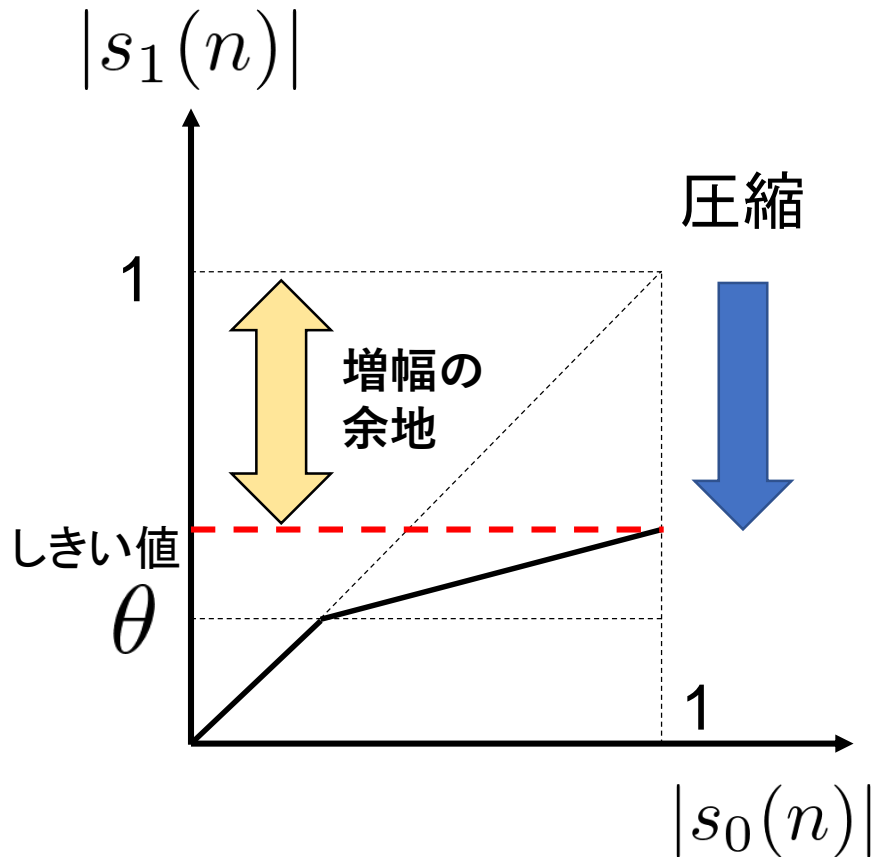
振幅の
圧縮

圧縮によって、「隙間」が発生
⇒ 振幅を増幅する余地



コンプレッサによる振幅の増幅(2/3)

振幅を圧縮することで**増幅の余地**が生まれる

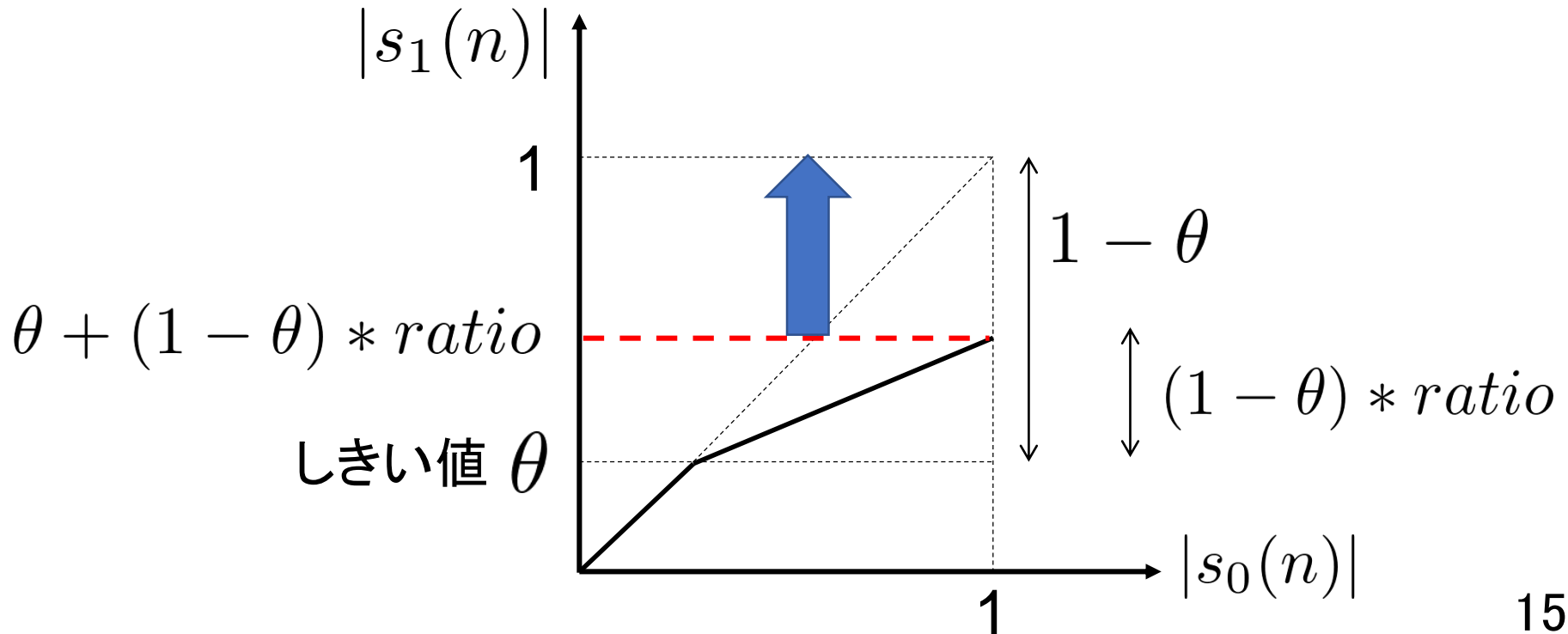


コンプレッサによる振幅の増幅 (3/3)

- 圧縮後、次式に従い振幅を増幅

$$s_1(n) = gain * s_0(n) \quad \text{ただし} \quad gain = \frac{1}{\theta + (1 - \theta) * ratio}$$

- 増幅率 $gain$ の意味

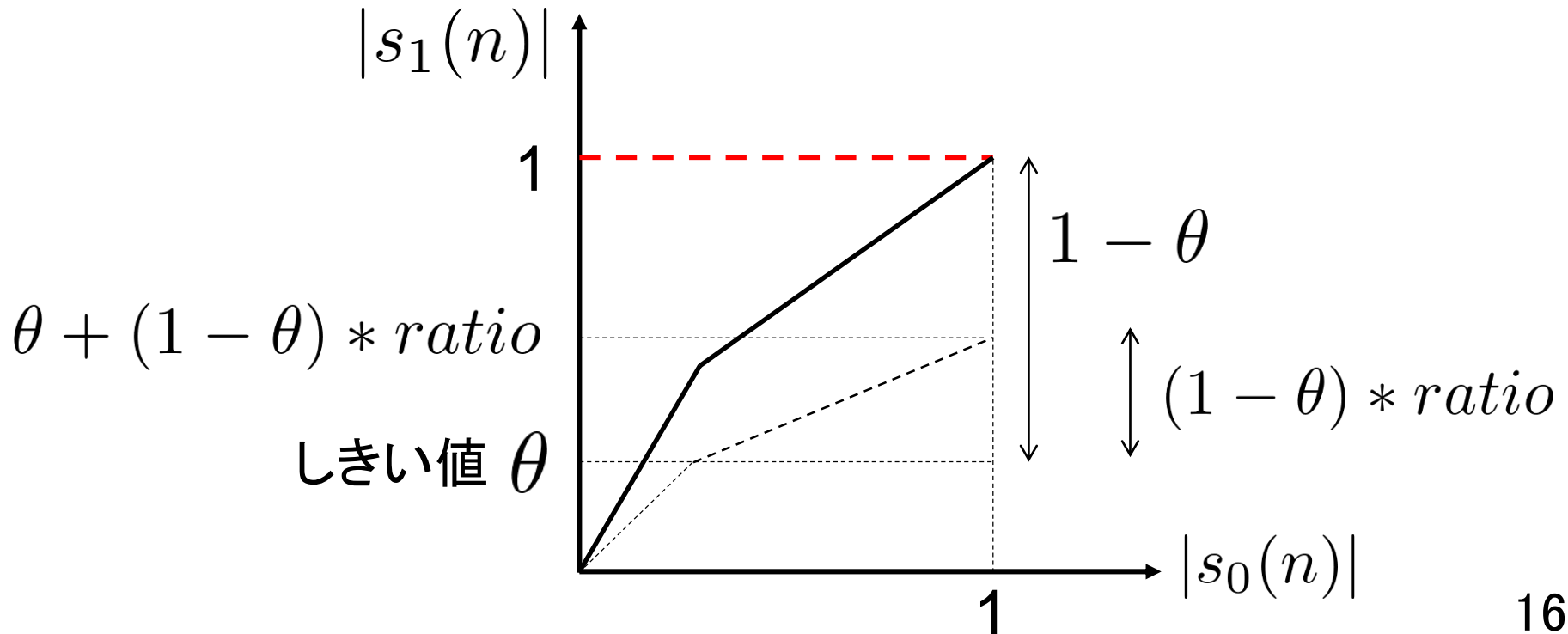


コンプレッサによる振幅の増幅 (3/3)

- 圧縮後、次式に従い振幅を増幅

$$s_1(n) = gain * s_0(n) \quad \text{ただし} \quad gain = \frac{1}{\theta + (1 - \theta) * ratio}$$

- 増幅率 $gain$ の意味



コンプレッサの処理のまとめ

- 振幅の圧縮

$$s_1(n) = \begin{cases} -\theta + (s_0(n) + \theta) * ratio & (s_0(n) < -\theta) \\ s_0(n) & (-\theta \leq s_0(n) \leq \theta) \\ \theta + (s_0(n) - \theta) * ratio & (s_0(n) > \theta) \end{cases}$$

ただし $s_0(n)$: 入力 $s_1(n)$: 出力 θ : しきい値 (正の値)

- 振幅の増幅

$$s_1(n) = gain * s_0(n) \quad \text{ただし} \quad gain = \frac{1}{\theta + (1 - \theta) * ratio}$$

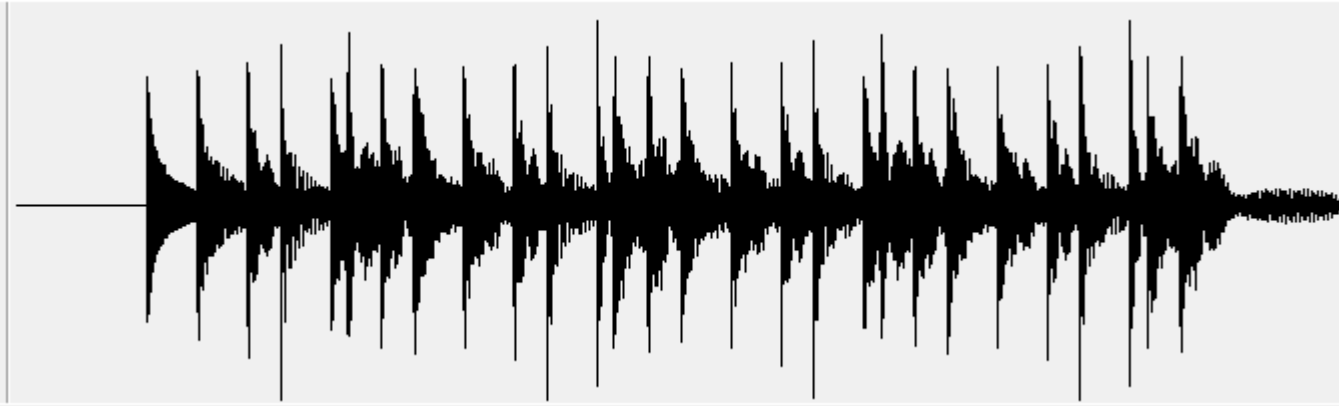
※圧縮後に増幅

コンプレッサのデモンストレーション

- コンプレッサをかける前

30369

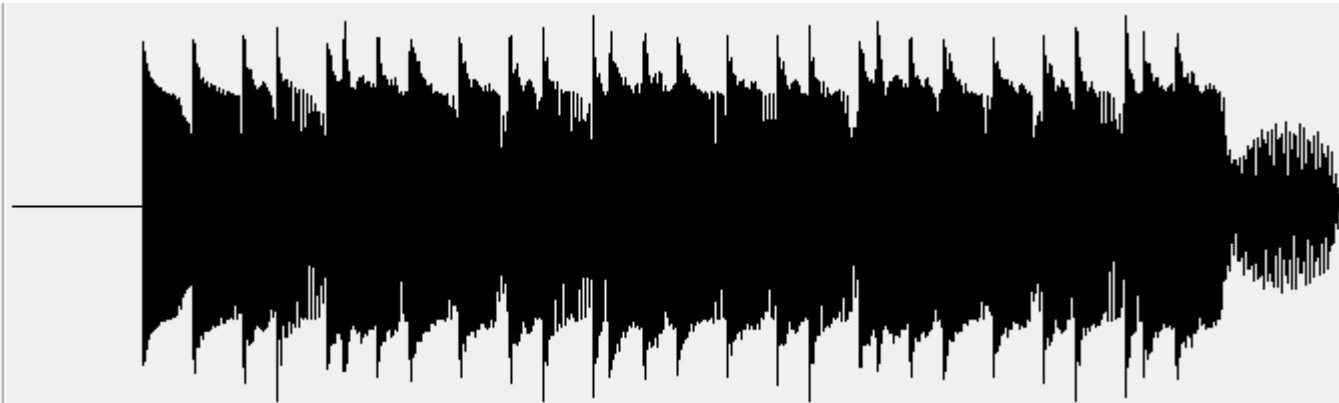
-32766



- コンプレッサをかけた後（増幅なし）

5682

-5898

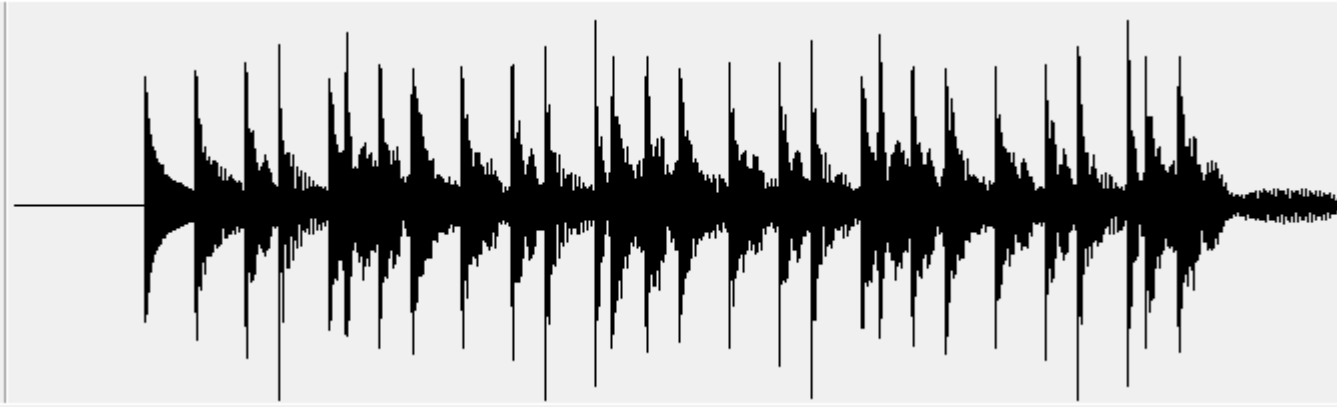


コンプレッサのデモンストレーション

• コンプレッサをかける前

30369

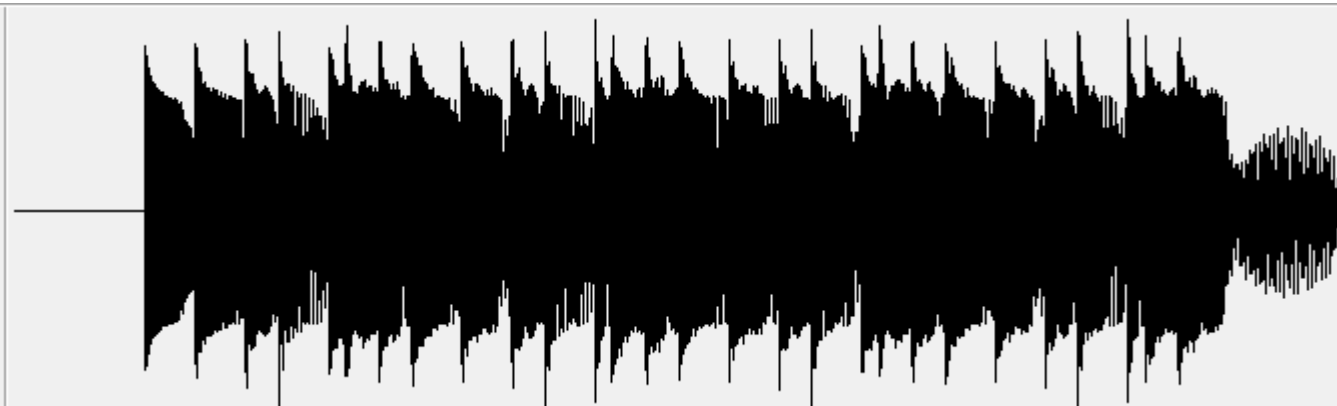
-32766



• コンプレッサをかけた後（増幅あり）

29907

-31043



コンプレッサの実装

擬似コード

N : 音データの長さ, F_s : 標本化周波数

s_0 : 入力の音データ, s_1 : 出力の音データ

$\text{threshold} \leftarrow$ しきい値, $\text{ratio} \leftarrow$ レシオ

$\text{gain} \leftarrow$ (スライドを参考にして設定)

for $n = 0$ to $N - 1$ **do** // 各時刻のデータに対して処理

$s_1(n) \leftarrow s_0(n)$

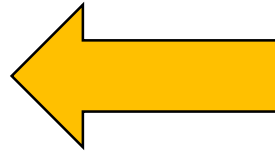
 /* スライドを参考にして振幅の圧縮処理を書く */

$s_1(n) \leftarrow \text{gain} * s_1(n)$

end for

今日やること

- 音を揃える
 - リミッタ
 - コンプレッサ
- 音を揺らす
 - トレモロ
 - ビブラート
- 音を広げる
 - コーラス
 - フランジャ



トレモロとは？

- トレモロ奏法に由来

- ギターやマンドリンで、同じ音を繰り返し早引きすることで音の大きさが震えているように聞かせる奏法



村治佳織 さん

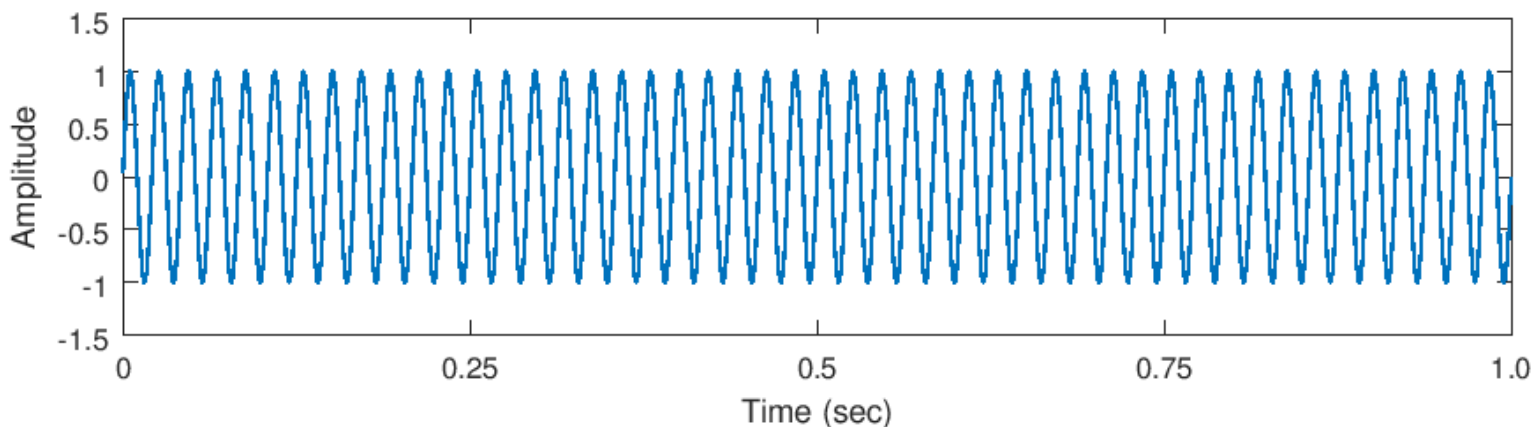
<https://www.youtube.com/watch?v=LwySHIWl9qY>



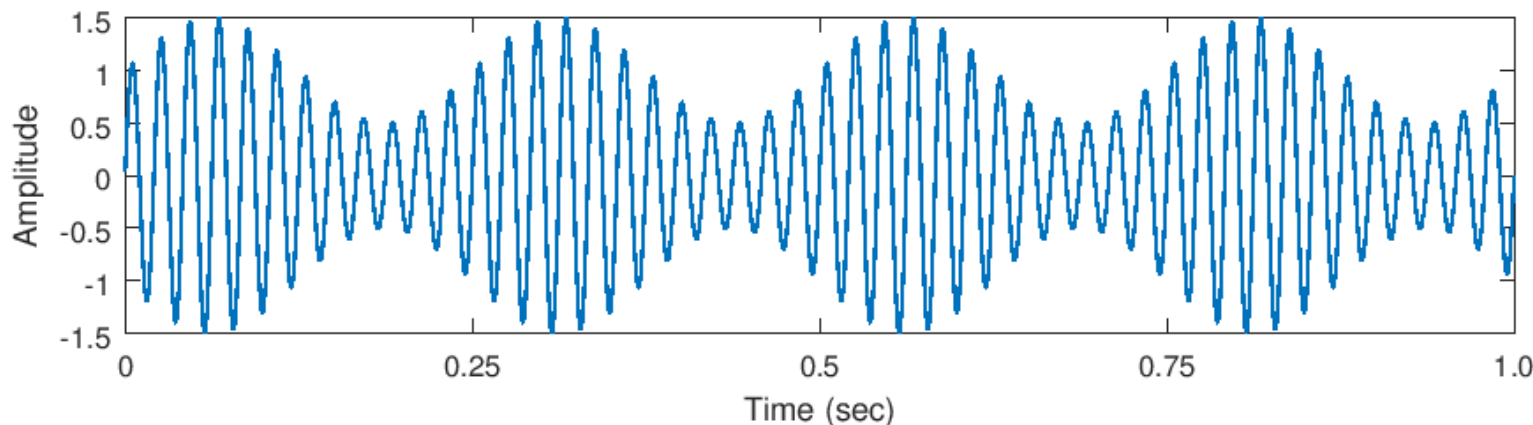
トレモロとは

- トレモロ奏法にヒントを得たサウンドエフェクト
周期的な信号(例:正弦波)によって音の大きさを変化
⇒音の大きさを周期的に揺らし、音を震わせる

トレモロを
かける前



トレモロを
かけた後

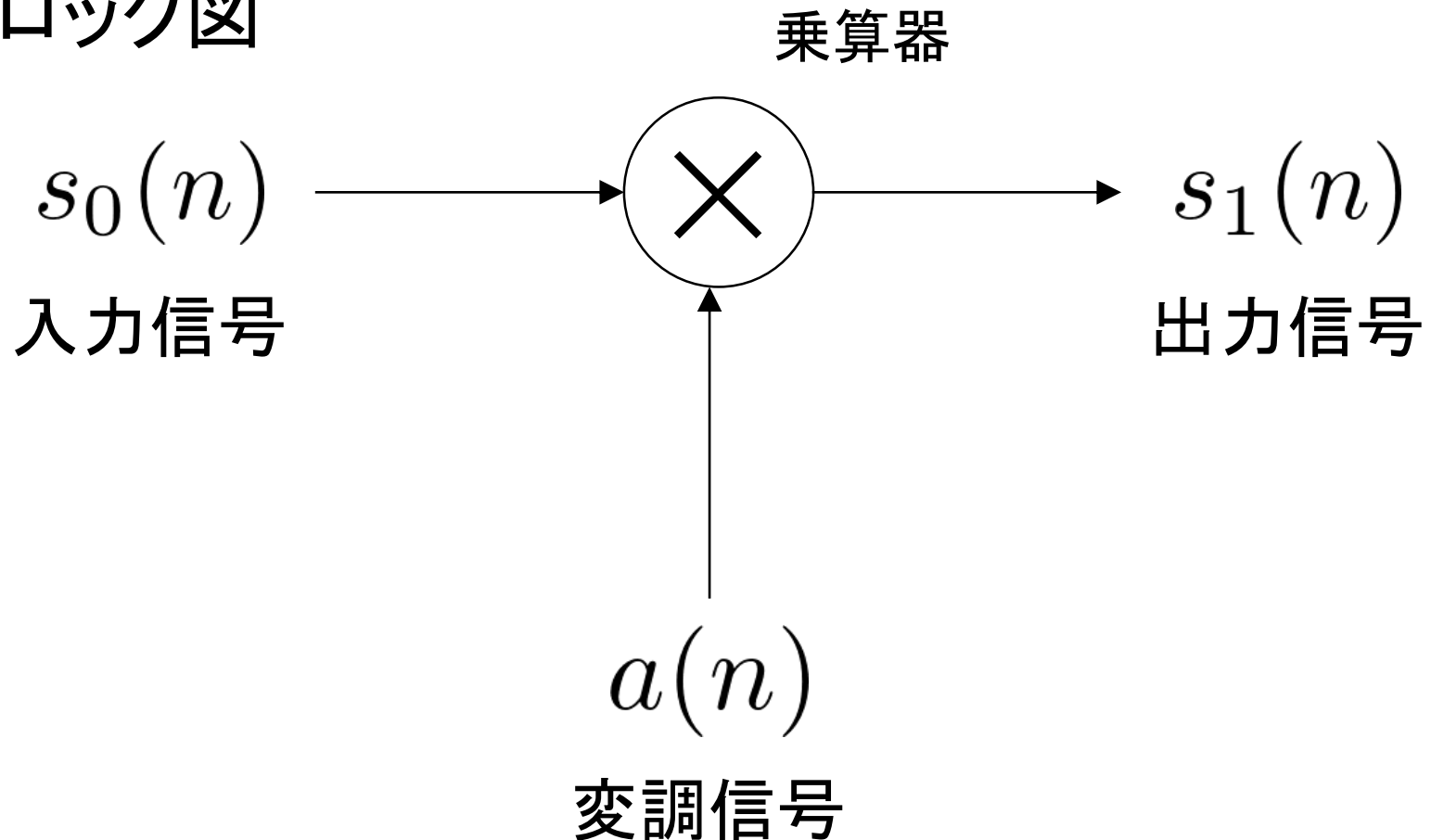


トレモロの原理

- ラジオの通信方式「AM」と同じ

AM = Amplitude Modulation (振幅変調)

- ブロック図



トレモロの原理

- 次式で定義

$$\frac{s_1(n)}{\text{出力信号}} = \frac{a(n)}{\text{変調信号}} \times \frac{s_0(n)}{\text{入力信号}}$$

トレモロのパラメータ

$$a(n) = 1 + \boxed{\text{depth}} \cdot \sin \left(\frac{2\pi \cdot \boxed{\text{rate}} \cdot n}{F_s} \right)$$

揺れの「深さ」

揺れの「細かさ」

※ $a(n)$ が負にならないように、 depth の絶対値は1以下

トレモロのデモンストレーション

- トレモロをかける前



- トレモロをかけた後 (depth = 0.5)

Rate (Hz)	1.0	3.0	5.0	7.0	9.0
音					

トレモロの実装

擬似コード

N : 音データの長さ

M_PI : 円周率 π

Fs : 標本化周波数

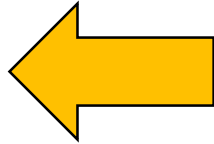
s0 : 入力の音データ, s1 : 出力の音データ

depth \leftarrow 揺れの深さ, rate \leftarrow 揺れの細かさ

```
for n = 0 to N -1 do           // 各時刻のデータに対して処理
    a  $\leftarrow$  (振幅変調)
    s1(n)  $\leftarrow$  a * s0(n)
end for
```

今日やること

- 音を揃える
 - リミッタ
 - コンプレッサ
- 音を揺らす
 - トレモロ
 - ビブラート
- 音を広げる
 - コーラス
 - フランジャ



ビブラートとは

- ビブラート奏法に由来

弦を押さえている指を小刻みに揺らすことで、音が震えているように聞かせる奏法

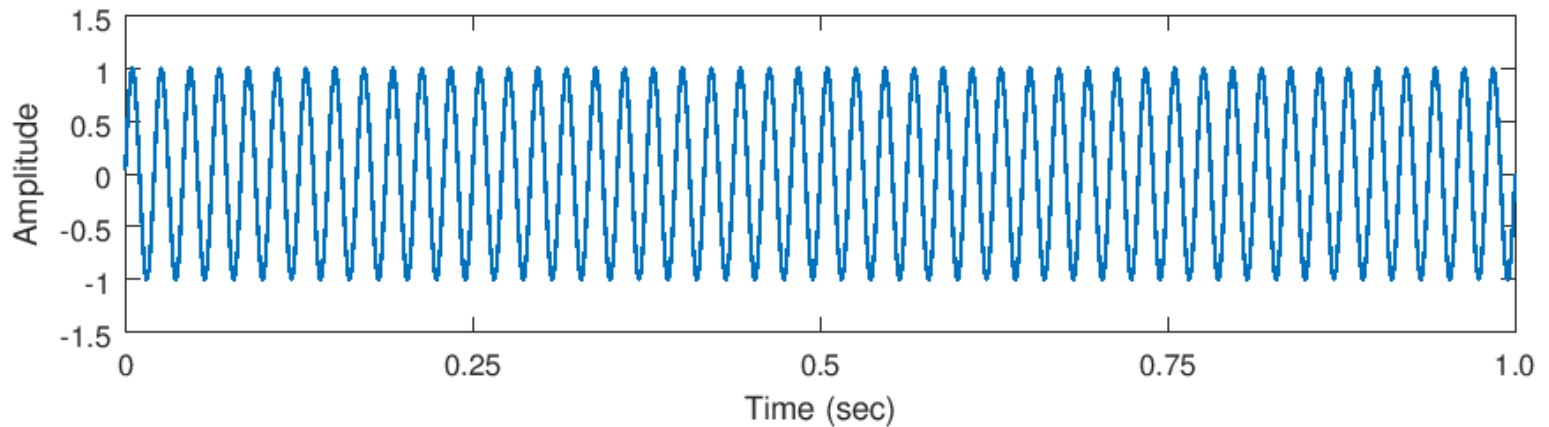


<https://www.youtube.com/watch?v=d-ve7ZzMlaQ>

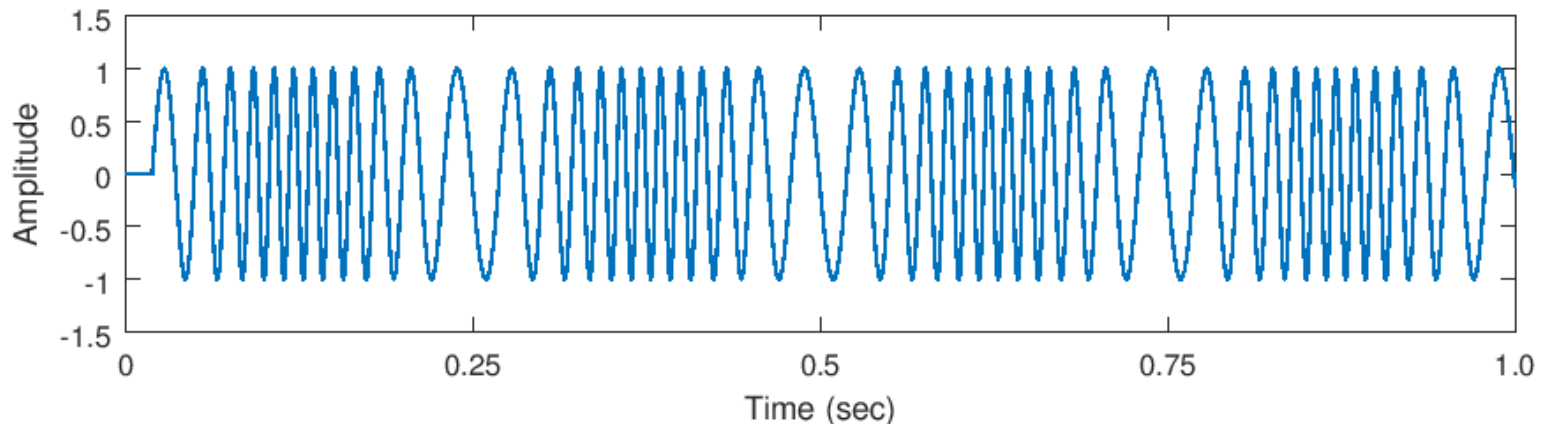
ビブラートとは

- ビブラート奏法にヒントを得たサウンドエフェクト
正弦波など周期的な信号によって音の高さを変化させることで音の高さを周期的に揺らし、音を震わせる

ビブラートを
かける前



ビブラートを
かけた後

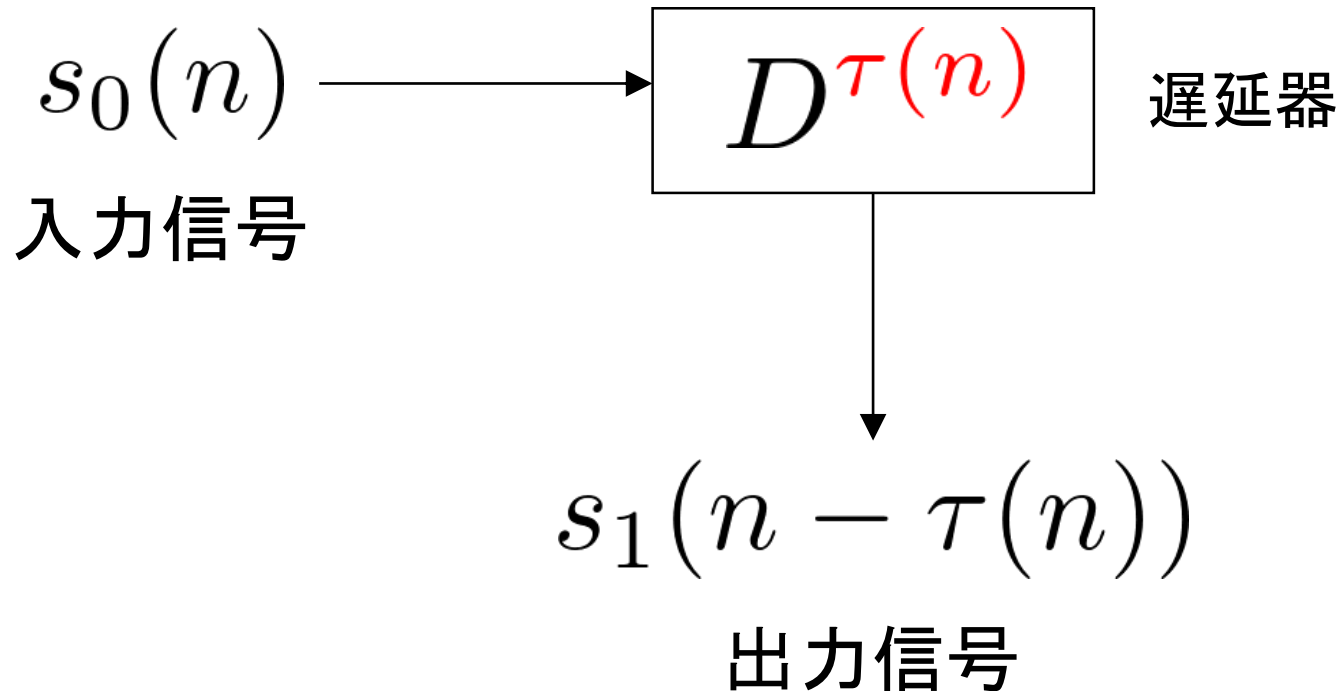


ビブラートの原理(1/4)

- ラジオの通信方式「FM」と同じ

FM = Frequency Modulation (周波数変調)

- ブロック図



ビブラートの原理(2/4)

- 次式で定義

$$\boxed{s_1(n)} = \boxed{s_0}(t) \quad \boxed{t} = n - \tau(n)$$

出力信号 入力信号

$$\tau(n) = \overset{\text{オフセット}}{\boxed{d}} + \boxed{\text{depth}} \cdot \sin \left(\frac{2\pi \cdot \boxed{\text{rate}} \cdot n}{F_s} \right)$$

揺れの「細かさ」

揺れの「深さ」

※ $\tau(n)$ が負にならないように、depth の絶対値は d 以下

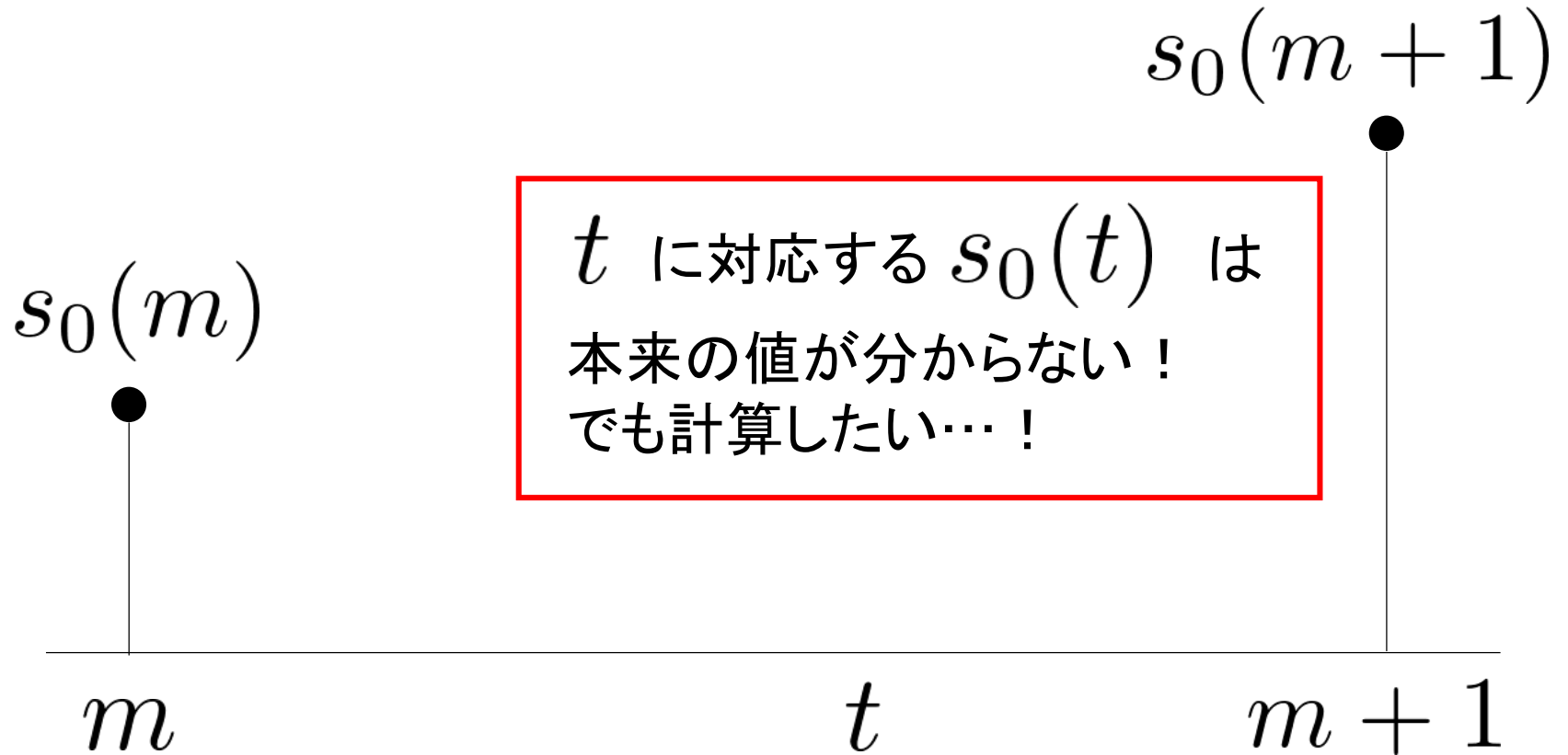
ビブラートの原理(3/4)

- ディレイを利用したサウンドエフェクト
 - 遅延時間が**時間的に一定**: ディレイ
 - 遅延時間が**時間的に変化**: ビブラート

$$s_1(n) = s_0(t) \quad t = n - \tau(n)$$

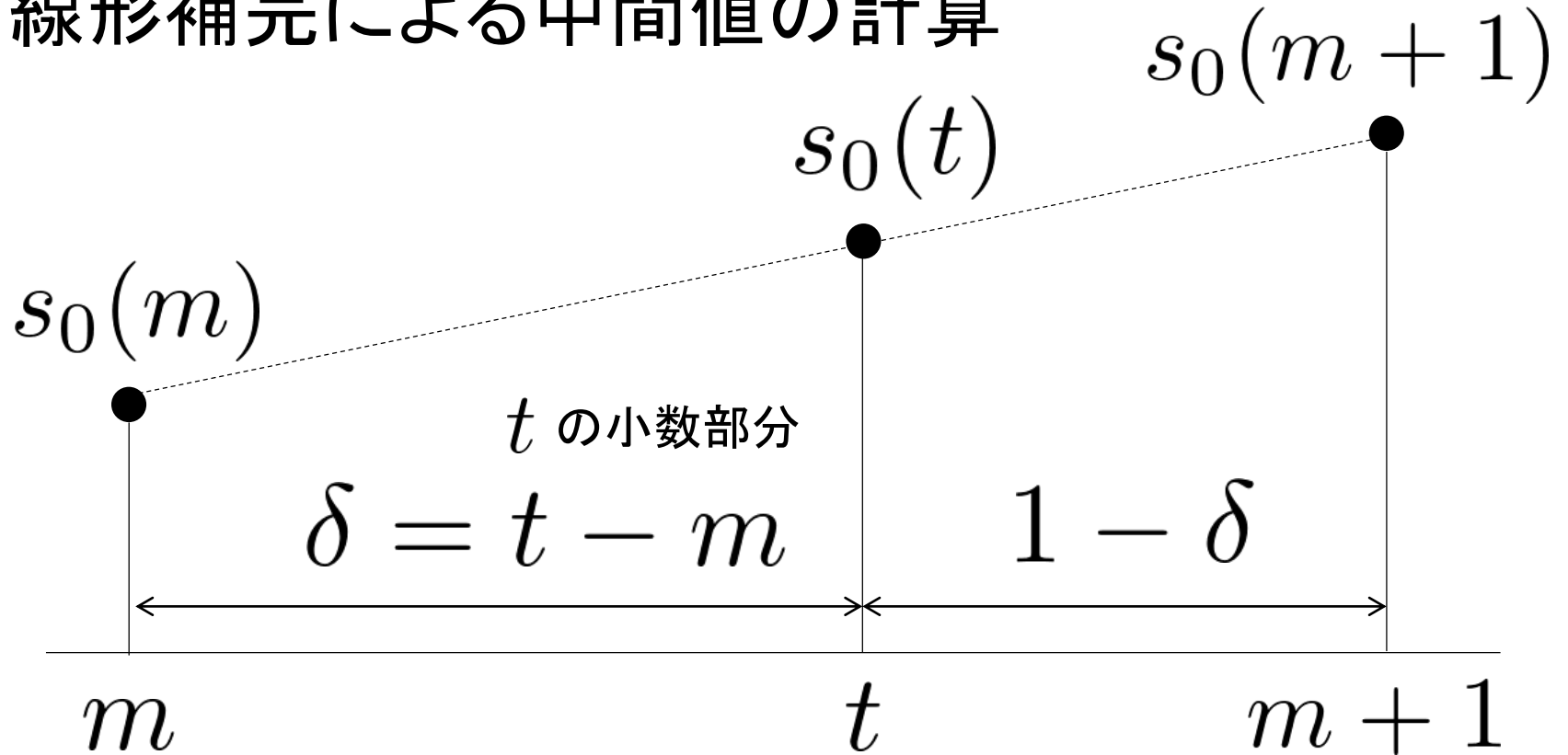
- 遅延時間が標本化周期の整数倍とは限らない
サンプルとサンプルの**中間値を求める処理**が必要
⇒次スライドで説明

ビブラートの原理(4/4)



ビブラートの原理(4/4)

- 線形補完による中間値の計算



$$s_0(t) = \delta \cdot s_0(m+1) + (1 - \delta)s_0(m)$$

$$m \leq t < m+1$$

ビブラートのデモンストレーション

- ビブラートをかける前



- ビブラートをかけた後 ($d = 2\text{ms}$, $\text{depth} = 2\text{ms}$)

Rate (Hz)	1.0	3.0	5.0	7.0	9.0
音					

ビブラートの実装

擬似コード

N : 音データの長さ, F_s : 標本化周波数, M_PI : 円周率 π

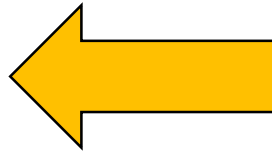
s_0 : 入力の音データ, s_1 : 出力の音データ

$d \leftarrow$ オフセット, $depth \leftarrow$ 揺れの深さ, $rate \leftarrow$ 揺れの細かさ

```
for n = 0 to N - 1 do      // 各時刻のデータに対して処理
    tau  $\leftarrow$  (遅延時間) // スライド資料の $\tau(n)$ 
    t  $\leftarrow$  n - tau      // 中間点 (double型)
    m  $\leftarrow$  t            // 変数t (double)から変数m(int) への代入により整数部分を抽出
                           // →暗黙の型キャスト
    delta  $\leftarrow$  t - m    // tの小数部分
    if m  $\geq$  0 かつ m < N then
        /*  $s_0$ の線形補完により $s_1$ を計算 */
    end if
end for
```

今日やること

- 音を揃える
 - リミッタ
 - コンプレッサ
- 音を揺らす
 - トレモロ
 - ビブラート
- 音を広げる
 - コーラス
 - フランジャ



コーラスとは？

合唱の効果を作り出すサウンドエフェクト



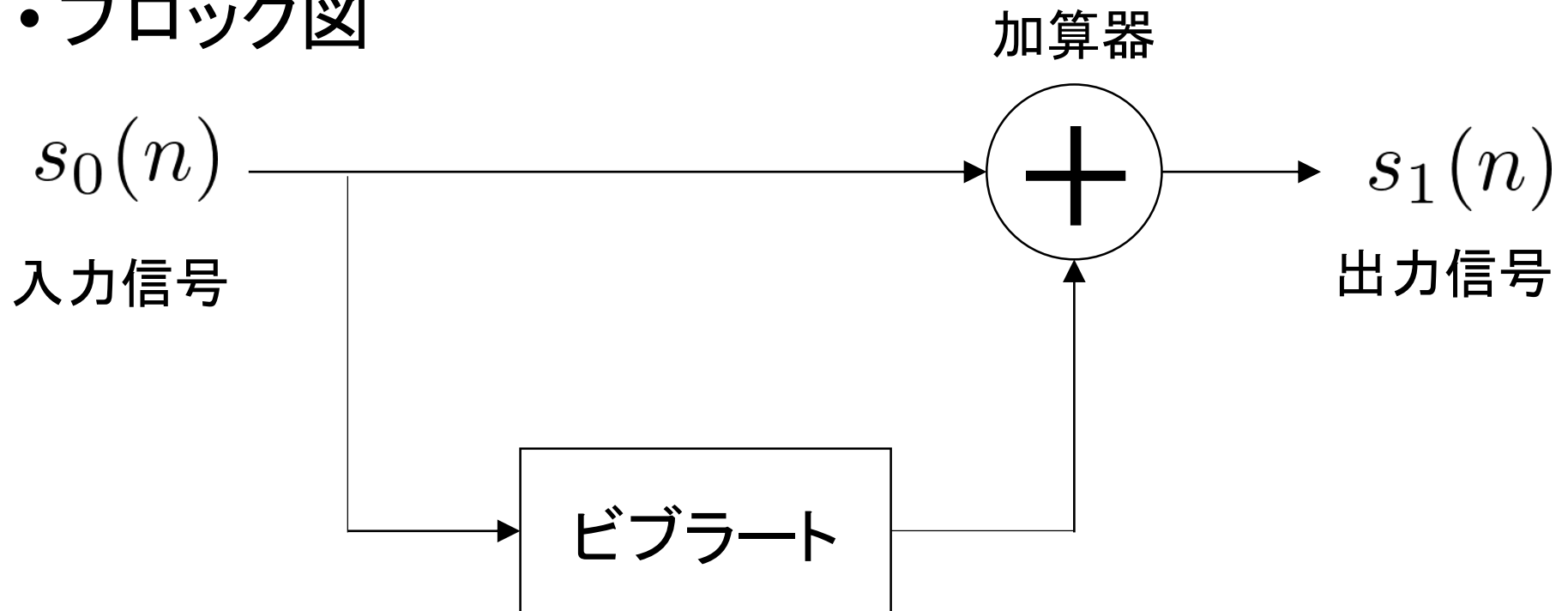
全員が正確な高さで
歌っているつもりでも
微妙にずれる
⇒合唱の歌声の独特な
ハーモニーの正体

コーラスの原理(1/2)

- ビブラートの応用で実現

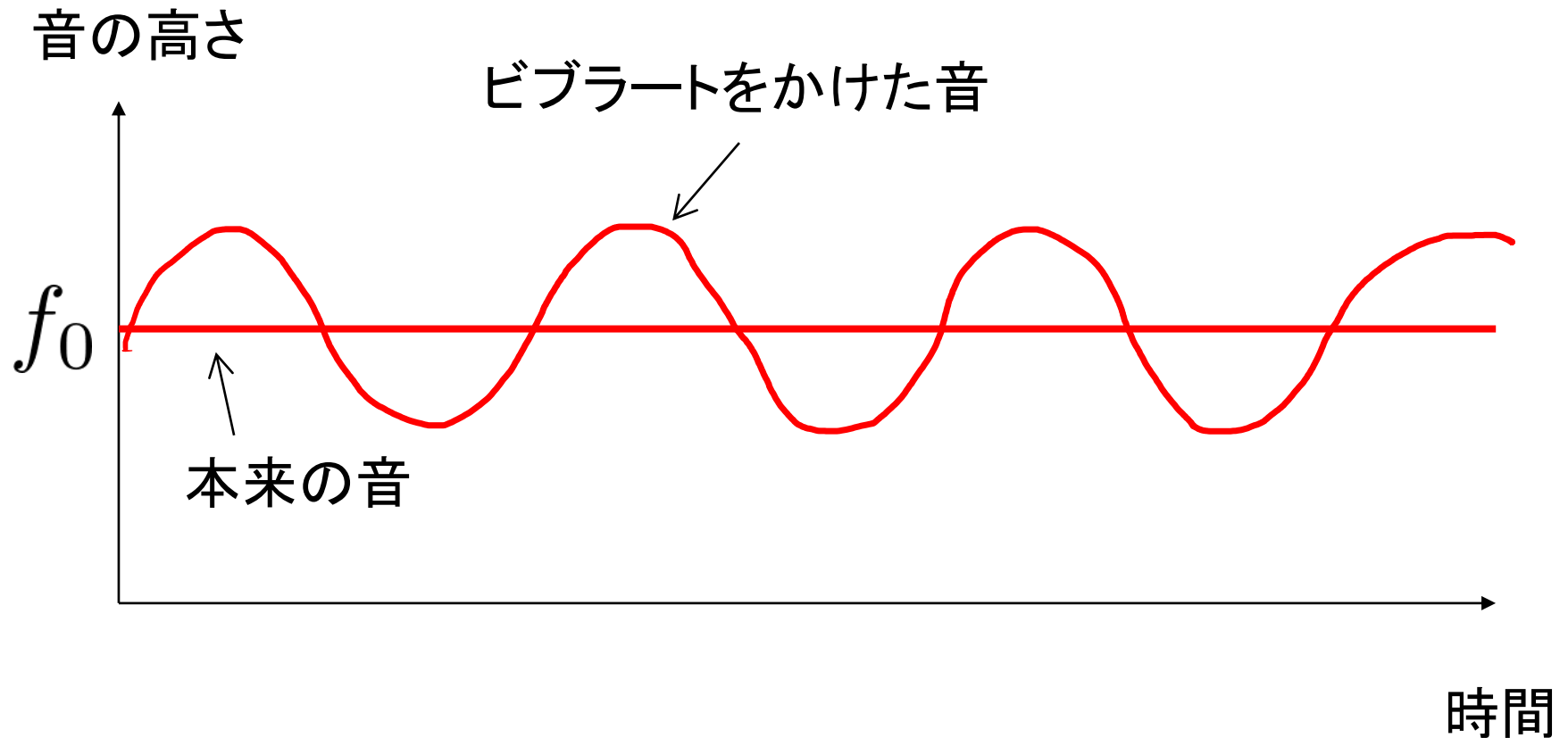
音の高さのずれた音データを本来の音データに重ねる

- ブロック図



コーラスの効果

本来の音の高さを中心として音の高さが広がった音データを作り出す











コーラスのデモンストレーション

- コーラスをかける前



- コーラスをかけた後 ($d = 25\text{ms}$, $\text{depth} = 10\text{ms}$)

Rate (Hz)	0.0	0.1	0.3	0.5	0.7	0.9	2.0	5.0
音								

コーラスの実装

擬似コード

N : 音データの長さ, F_s : 標本化周波数, M_{PI} : 円周率 π

s_0 : 入力の音データ, s_1 : 出力の音データ

$d \leftarrow$ オフセット, $depth \leftarrow$ 揺れの深さ, $rate \leftarrow$ 揺れの細かさ

for $n = 0$ to $N - 1$ **do** // 各時刻のデータに対して処理

$s_1(n) \leftarrow s_0(n)$

 /* ビブラートと同様のため、省略 */

if $0 \leq m < N$ **then**

 /* ブロック図を参考にビブラートのプログラムを修正 */

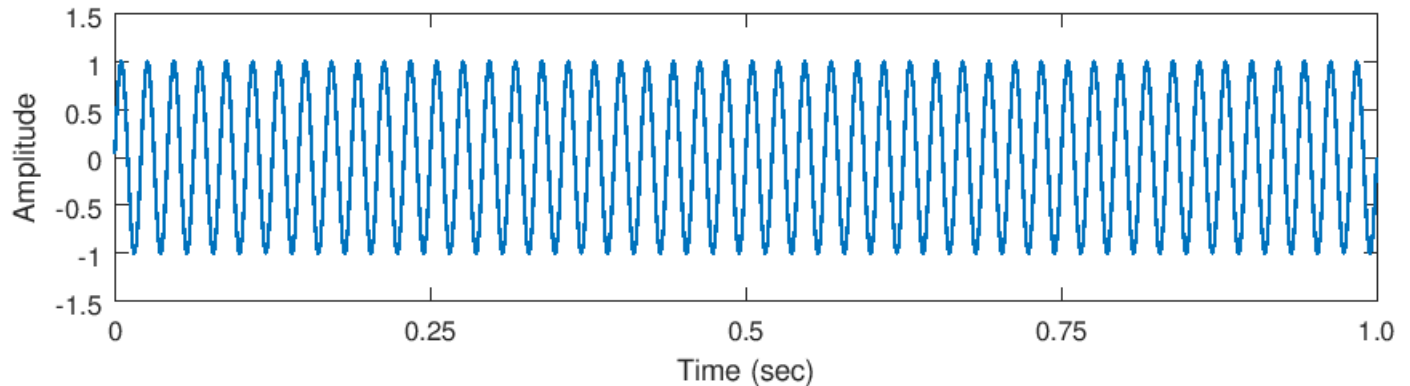
end if

end for

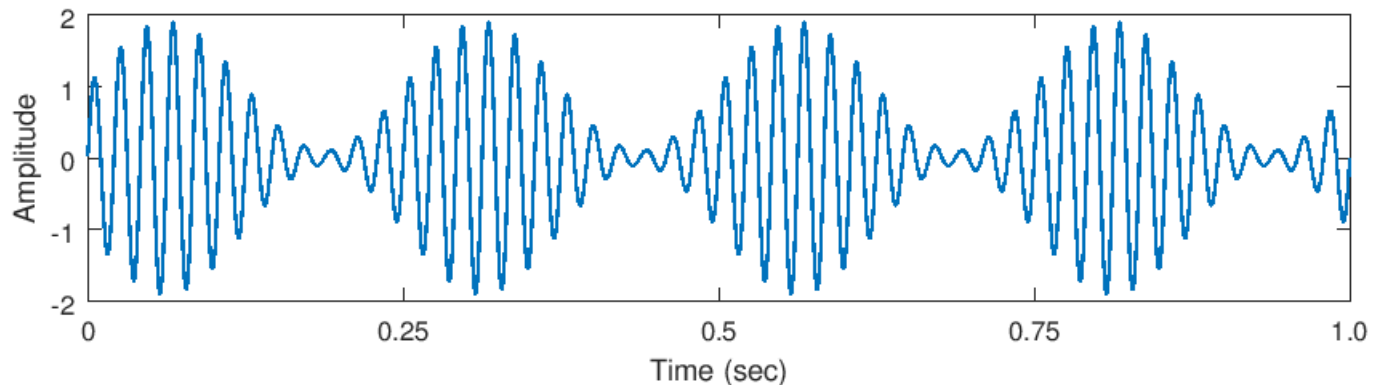
フランジヤとは？

- 原理的にはコーラスとまったく同じ
 - depthを大きく設定⇒コーラス
 - depthを小さく設定⇒フランジヤ
- 「うなり」を発生させるエフェクト

フランジヤを
かける前



フランジヤを
かけた後











フランジヤのデモンストレーション

- フランジヤをかける前



- フランジヤをかけた後 ($d = 2\text{ms}$, $\text{depth} = 2\text{ms}$)

Rate (Hz)	0.0	0.1	0.3	0.5	0.7	0.9	2.0	5.0
音								

実装について

- コーラスと同じなので省略

まとめ

- 音を揃える: リミッタ、コンプレッサ
 - リミッタ: しきい値を超えた振幅をしきい値に置換
 - コンプレッサ: しきい値を超えた振幅を圧縮 & 増幅
- 音を揺らす: トレモロ、ビブラート
 - トレモロ: 振幅を揺らす
 - ビブラート: 周波数を揺らす; ディレイの応用
- 音を広げる: コーラス、フランジヤ
 - コーラス: ビブラートの応用
 - フランジヤ: コーラスと同様; 「うなり」の効果

第5回
サウンドメディア論
および演習 演習編

準備

1. 20180511.zipをMoodleからダウンロードせよ
2. 適宜、適当なフォルダの下に解凍せよ
3. ターミナル上でmakeコマンドを実行し、コンパイルが通ることを確認せよ

演習 (1/3)

- コンプレッサのプログラムを作成せよ (compressor.c)
 - しきい値 threshold, レシオ ratioはともに小さめ (0.1~) の値に設定すると, より増幅がかかることを確認せよ
- TREMOLOのプログラムを作成せよ (tremolo.c)
 - depthは0.5に固定して, rateを色々と変えてみることでTREMOLOの効果を確認せよ
 - depthの値を変える場合、値が1以下になるように
⇒1以上にするとオーバーフローが発生

演習 (2/3)

- ビブラートのプログラムを作成せよ (vibrato.c)
オフセット d と 遅延の揺れ $depth$ は、最初は $2 \text{ ms} \times$ (標本化周波数) で固定し、 $rate$ を変えながら (0.1~)、ビブラートの効果を確認せよ。その後、 d や $depth$ の値を変えてみよ。 d と $depth$ は同じ値を設定するのが簡単。
- コーラスのプログラムを作成せよ (chorus.c)
 - ビブラートのプログラムおよびコーラスのブロック図を参考にプログラムを完成せよ
 - d , $depth$ はそれぞれ 25 ms , 10 ms で設定し、 $rate$ を変えて (0.1~) コーラスの効果確かめよ
 - その後、好みの値に調整せよ

演習 (3/3)

- フランジャのプログラム (flanger.c) はコーラスのプログラム完成後、中身をそのままコピーすることで完成する
 - dとdepthを小さい値 (例えば 2 ms)に設定し、rateを変えて効果を確認せよ

提出

- コンプレッサ compressor.c
- トレモロ tremolo.c
- ビブラート vibrato.c
- コーラス chorus.c

※dやdepthやrateなどには各自で値を設定

※フランジャ flanger.cは提出の必要なし

提出方法

1. フォルダを作成

フォルダ名は「学籍番号_0511」とする

例: 学籍番号がK123456ならば「K123456_0511」

2. 提出するファイルをその中に入れる

3. Finder上でフォルダをCtrl+左クリックし、圧縮ファイル(zip)を作成する

4. Moodleにアップロードして課題提出