# RRM

## 特性再構成モデル

Response Reconstruction Model

### 01 コンセプト

実機への忠実な再現よりも、音づくりの自由度と自然な操作感を重視する。
必要な特性を絞って軽量な設計を目指す、実用志向のモデル。

ニューラルネットワーク型のモデルでは、学習後に内部の音づくりを狙って変えることや、学習条件から離れた入力への対応が課題になる場合があります。RRMは、そうした弱点を補う選択肢として設計します。

評価の中心は、実機との一致度だけでなく、ゲインや入力の強さを変えたときの自然さ、調整のしやすさ、演奏に使える音の範囲です。

**操作感**

入力やゲインへの
自然な追従を設計

**自由度**

音の特徴を保ちながら
用途に合わせて変更

**軽量な設計**

必要な特性を絞り
処理を組み立てる

### 02 方式の概要と呼び方

RRMは、実機の測定結果を基に、歪み関数・前後のフィルター・音の履歴を扱う処理を組み合わせ、構造と係数を調整して作るモデルです。

ここでいう「特性」には、周波数による音の変化、入力の強さと歪みの関係、過去の音が現在の出力へ与える影響などを含みます。どこまで扱うかは、採用する構造によって決まります。

本資料では、この開発で用いる方式を「特性再構成モデル／RRM」と呼びます。

## 03 実機からモデルを作る流れ

### 1 記録する

同じテスト信号を使い、直結時と実機通過時の音を、そろえた入出力設定で記録します。

### 2 違いを調べる

測定系の影響と遅延を確認し、音色の変化や、入力レベルによる歪み方を調べます。

### 3 構造を選ぶ

前段フィルター、歪み関数、後段処理の組合せを決め、表したい特徴を整理します。

### 4 係数を合わせる

測定結果との違いを見ながら、関数やフィルターの係数を調整します。

### 5 確認する

調整に用いた信号だけでなく、別の信号や演奏でも、音の強弱と操作感を確認します。

## 04 モデルの内部構造

**入力**
レベル調整
→
**前段フィルター**
歪ませる帯域
→
**歪み関数**
tanhなど
→
**後段の複数経路**
成分ごとに整形
→
**合成・出力**
音色と音量

前段で「どの帯域を歪ませるか」を整え、関数で歪みを作り、後段で生まれた成分のバランスを調整します。
後段フィルターは過去のサンプルも使うため、処理全体には音の履歴が含まれます。

今回の中心部を表す概念式

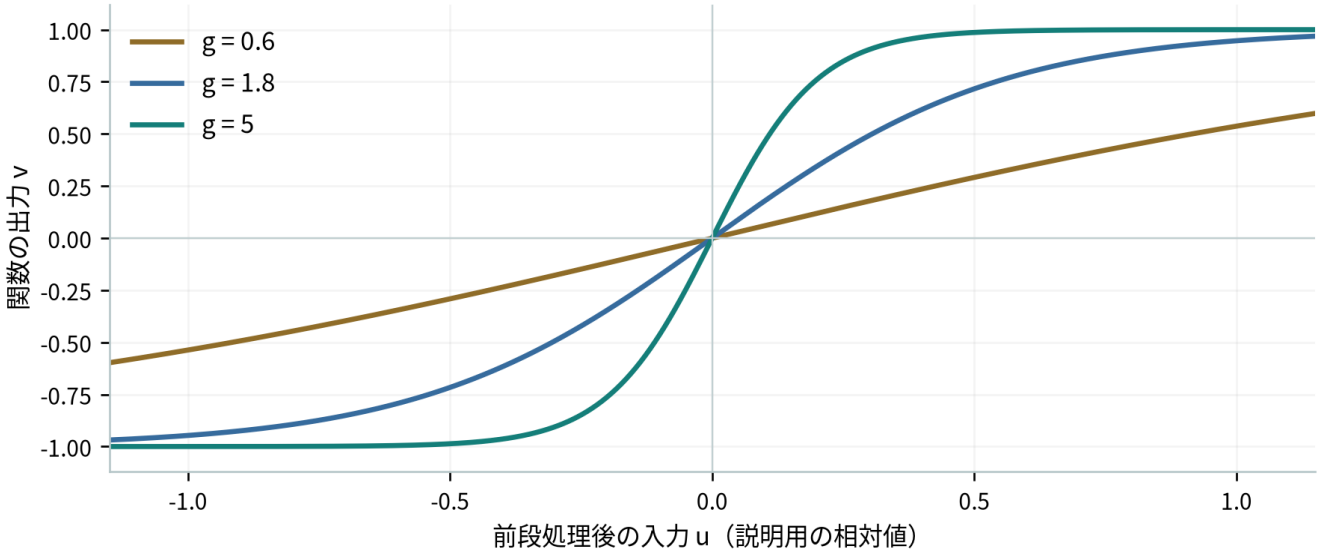
$$v = \tanh(g \cdot u)$$

$$y = F1(v) + F2(v^2)$$

uは前段処理後の入力、gは歪みの強さ、vは関数の出力です。F1・F2は履歴を持つフィルターで、vと$v^2$の経路を別々に整えて合成します。式は中心部の説明用で、レベル調整などは省略しています。

## 05 ゲインを変えたときの振る舞い

歪み関数は、入力が小さいときには比較的穏やかに、大きいときには飽和へ近づく形を持ちます。入力ゲインを変えることで、その関数のどの領域を使うかが連続的に変わります。




図はtanh関数の説明用です。実機や製品の測定値ではありません。gは概念図の係数で、製品ノブの表示値とは異なります。

ギターのボリュームやピッキングの強さに対しても、同じ構造の中で反応を変えられます。狙うのは、広い操作範囲で音がつながり、演奏に使える変化が得られることです。

関数として連続していても、どの入力・設定でも自然な音になるとは限りません。強弱やゲインの範囲を広く試し、聴感と弾き心地で確認します。

## 06 実機の範囲を超える音づくり

実機から得た特徴を土台に、歪み量や前後の帯域を独自に調整できます。ゲインだけでなく、どの成分を歪ませるか、歪ませた後に何を残すかも設計対象です。

| 調整する部分 | 音づくりの方向 |
| --- | --- |
| 入力と歪み関数 | 軽い歪みから強い歪みまで、使う範囲を広げる。 |
| 前段の帯域 | 低域の混み合い方や、歪みへ入る成分を整える。 |
| 後段の帯域・応答 | 高域の角、音の芯、必要な強弱の反応を設計する。 |

入力ゲインの倍率と、聴感上の歪み量は比例しません。大きく変更した領域は、独自の楽器として使いやすいかを評価します。

## 07 機械学習モデルとの違い

ここでは、NAMなどのニューラルネットワーク型モデルを比較対象とします。NAMは学習済みの重みを持ち、WaveNetやLSTMなどの構造を扱います。[1][2]

| 比較点 | RRM（今回の構成） | ニューラルネットワーク型 |
|---|---|---|
| 構成 | 歪み関数と前後のフィルターなど、役割を決めた処理。 | ネットワークの構造と、学習済みの重み。 |
| 実機への合わせ方 | 用意した処理構造の中で、関数やフィルターの係数を調整。 | 入出力データを用いて、ネットワークの重みを学習。 |
| 音の変更 | 役割の分かる関数・帯域・処理構造を個別に変更。 | 学習構成に応じた制御や前後のDSPを利用。内部の変更には再学習等が必要になる場合もある。 |
| 入力条件の変化 | 構造に沿った応答を設計し、範囲外も試奏で評価。 | 学習条件とモデル構造に応じて応答。適用範囲を確認する。 |

RRMは、調整対象の役割を明確にし、目的に合わせて変更しやすくする設計です。一方、表現できる挙動は用意した構造に依存します。モデルの種類や学習条件による違いを踏まえ、方式全体の優劣と、個別モデルの出来を分けて考えます。

## 08 現在の制約と今後の可能性

・基準になるのは、特定の実機・設定・測定レベルで取得したデータです。
・測定には接続機器やレベル調整の影響も含まれます。
・ゲイン操作の自然さと、実機のGAINノブの再現性は別々に評価します。
・フィルターの履歴を持っていても、実機の非線形な圧縮や時間変化まで自動的に再現できるわけではありません。
・測定範囲を超えるほど、元の特性との釣り合いが変わる可能性があります。

今後は、必要な応答を追加したり、異なる歪みへ構造を展開したりしながら、自由に調整できる実用的な音の範囲を広げていきます。

**参考資料（ニューラルネットワーク型モデルの説明）**

[1] NAM公式リポジトリー：README（学習とモデルの出力）
[2] NAM公式資料：.namファイル仕様（構造と重み）