修 士 論 文

耐故障性を有する
階層型ニューラルネットワークの汎化能力

指導教官　日比野 靖 教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

武井 良太

1997 年 2 月 14 日

目 次

第 1 章

序論

1.1 本研究の背景と目的

ニューラルネットワークは生物の神経回路網を規範としていることから、その特徴として、学習能力や並列分散動作などがあげられる。さらに、汎化能力や耐故障性というものがあると言われている。

ニューラルネットワークにおける最大の特徴の一つである汎化能力は、学習されていない未学習のパターンに対して、どれだけ妥当な値をかえすことができるか、という能力のことである。この汎化能力を向上しようとする数多くの研究が報告されているが、「同じ情報源から標本化された未学習パタンに対する誤差」という観点から実験的に評価されることがほとんどである。これに対し、クロスバリデーション法のような統計的手法により評価の精度を向上させようとする試みや、汎化を学習曲線、最尤推定、正則化、逆射影フィルタなどの枠組で捉えようとする理論的試みが盛んになりつつあり、これらを応用した学習規則なども提案されるに至っている。

一方、ニューラルネットワークが何らかの耐故障性を有するという指摘も古くからなされてきた。しかしながら、従来のニューラルネットワークにおける耐故障性の研究では耐故障性ということば自体、故障が生じてもあまり誤りが大きくならないといった程度の意味合いで用いられることが多く、耐故障性の定義は曖昧であった。近年、ニューラルネットワークの有する耐故障性を積極的に利用しようとする立場から、フォールトトレラントコンピューティング (FTC) で用いられている「誤りが生じても仕様を満たし続ける性質」という定義に基づいた議論 [1] が行われるようになってきた。しかしながら、FTC で主

な対象とされる論理回路とニューラルネットワークとでは基本的な性質が大きく異なる。本論文では、[1] で用いられていた耐故障性の定義が必ずしも有効ではない簡単な例を挙げ、ニューラルネットワークに対して要求される耐故障性の定義について議論する。

また、従来のニューラルネットワークにおける耐故障性研究の多くでは、単一の故障のみを扱っていた。しかし、実世界の生物のシステムにおいては、単一故障よりは、複数の故障が発生していると考える方が自然であり、工学的ニューラルネットワークも複数故障に対してうまく対処できる能力があると期待される。

近年、丹ら、Neti らの研究により、耐故障性と汎化能力との間に密接な関係があることがわかってきた。しかしながら、このような関連性がどこから生じてきているかということについては、必ずしも明らかにはなっていない。

そこで、本研究では、学習により獲得されたネットワークにおいて、耐故障性に優れたネットワークは汎化能力においても優れているという従来の主張を確認し、さらに、各々のネットワークの特徴量を調べることにより、耐故障性、汎化能力、の関連性をもたらす原因についての一考察を与える。

これら二つの性質を支配する要因として、いくつかのネットワーク特徴量をとりあげ、シミュレーションにより、これらが耐故障性と汎化能力にどのように関わっているかを調べる。これらの特徴量として、学習パターンに対する誤差、出力関数の滑らかさ、中間層素子の微係数、中間層表現（中間層素子の平均・分散、中間層素子の分離直線）、結合荷重の統計的性質、出力分布、等を検討する。

1.2 本論文の構成

第 1 章では、本研究の背景と目的 及び 本論文の構成について述べる。

第 2 章では、階層型ニューラルネットワークについて簡単に説明する。本論文で対象としている 3 層の階層型ネットワークを中心に、ネットワーク構造、ネットワークの学習方法、ネットワークの特徴を示す。

第 3 章では、汎化能力の定義と評価方法について述べる 2 章のネットワークと学習法を用いて正規分布による帯状パターンという問題で汎化能力の実験を行なう。評価方法にはテストパターン法とクロスバリデーション法を使用し、誤差の大きさにより評価を与える。さらに、得られたネットワークの特徴量（内部表現）を調べる。特

徴量には、中間層表現、微係数、関数のなめらかさ、結合荷重、出力分布、等が考えられ、各々について見てみる。

ネットワークの学習において、学習率、慣性項係数などにより、形成されるネットワークは変わってくるため、基本となるこれらのパラメータから調査からはじめる。

第 4 章では、耐故障性について述べる。次に耐故障性について実験を行なう。XOR, 正規分布の帯状パターンという問題に ftbp 法を用いてネットワークを学習させる。今まで使われてきた必須結合数による評価の問題点について議論し、新しい評価方法として関数比較により耐故障性の評価を行なう。

本章でも 3 章と同様に、ネットワークの学習における基本的なパラメータ（学習率、慣性項係数など）の調査からはじめる。故障モデルとして、単一断線故障だけではなく、多重故障についても評価をおこなう。

耐故障化学習によりえられたネットワークの特徴量については、第 5 章において見ることにする。

第 5 章では、汎化能力と耐故障性の関係について考察する。以上より、評価されるべきネットワークが構築された。ここで、耐故障性に優れているネットワークは、汎化能力においても優れていることを示し、そのときのネットワークの特徴量がどのようになっているかを調べることから両者の関係について議論する。

第 6 章では、本研究で得られた成果と、今後の課題を述べる。

第 2 章

階層型ニューラルネットワーク

本章では、ニューラルネットワークの概要とその構造について簡単に解説し、本研究の対象となった階層型ネットワークモデル、およびその学習アルゴリズムについて詳しく述べることにする。

2.1 ニューラルネットワークの概略

ニューラルネットワークとは、単純な演算を行なうノード（素子）とそれらをつなぐリンクを多数結合させたネットワークである。ノード間の結合には重み付け（結合荷重）がなされており、各ノード間の出力は式 (2.1) で計算される。

$$a = f\left(\sum_i w_i x_i - \theta\right) \tag{2.1}$$

但し、a はノードの出力、w は重み、θは閾値（バイアス)、f は出力関数を表す。
出力関数には、閾値関数、

$$f(u) = \begin{cases} 1 & if \quad u \geq 0 \\ 0 & if \quad u < 0 \end{cases}$$

線形関数、

$$f(u) = u$$

シグモイド関数、

$$f(u) = \frac{1}{1 + exp(-u)} \tag{2.2}$$

などが用いられる。

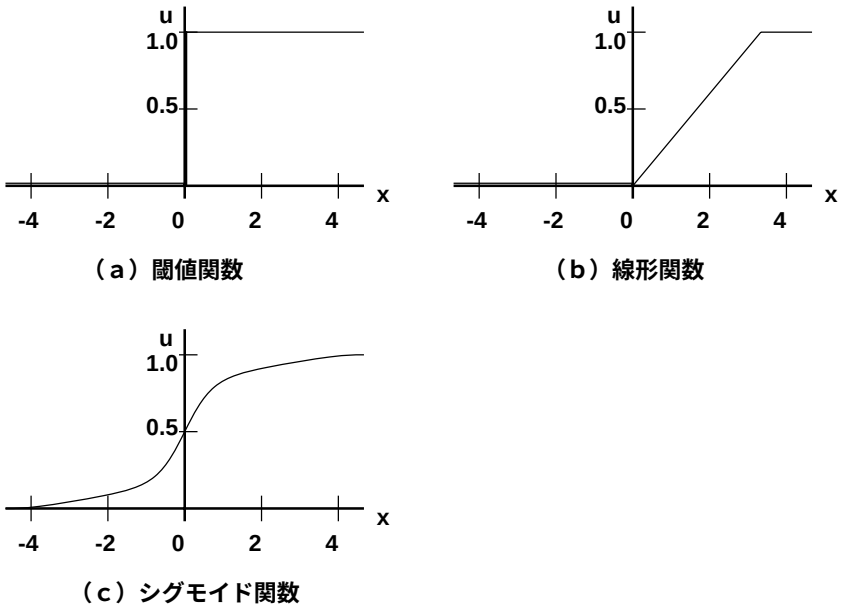

図 2.1: 出力関数

ニューラルネットワークは大きく分けて以下の 2 つに分類される。

- 相互結合型ネットワーク
- 階層型ネットワーク

相互結合型ネットワークは、通常すべてのノードが相互に結合したネットワークである。従って、素子出力のフィードバックループが形成されている。代表的なものにホップフィールドやボルツマンマシンがあり、巡回セールスマン問題等で使用される。

一方、階層型のネットワークはノードが層状に配置され、層の間にノードどうしの結合がある。結合の方向は、入力層、中間層、出力層への一方向であり、信号は一度しか流れない。本研究では 3 層の階層型ネットワークをしており、次にその構造を詳しく説明する。

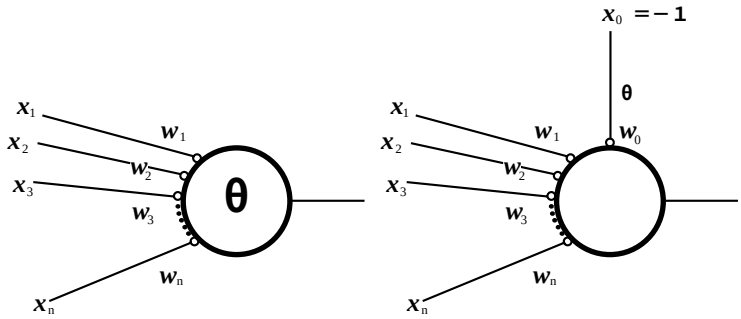

図 2.2: ノードモデル

2.2 階層型ネットワークの構造

階層型ネットワークは、ノードとそれらを結合するリンクから構成されている。それぞれの結合は重み（結合荷重）をもっており、この結合荷重をかけた入力値がその結合からのノードへの入力となる。図2.2に n 入力のノードを示す。ここで、x_iは i 番目の入力、w_iは i 番目の結合の荷重、θは閾値を示している。図のとおり、閾値は外からの入力に影響されない入力のバイアス値であり、仮想的に定数をとる入力と考え、他の結合と同様に扱うことができる。

この素子を層状に配置し、となりあった層の素子を完全結合することにより階層型ネットワークが実現される。図 2.3 に階層型ネットワークの例を示す。

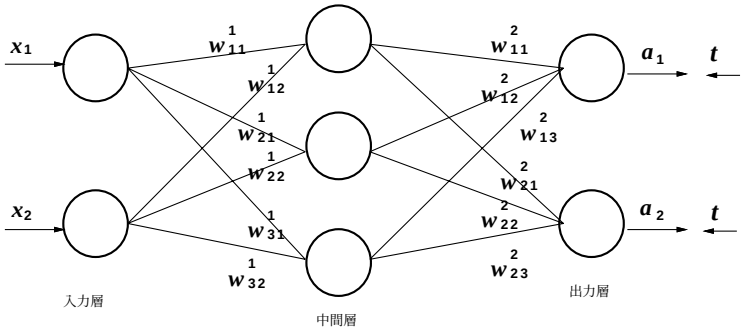

図 2.3: 階層型ネットワークの例

外部から入力を受ける層を入力層、外部に対して出力を行なう層を出力層、それらに挟

まれた層を中間層と呼ぶ。図 2.3 では入力層が 2 個、中間層が 3 個、出力層が 2 個の素子をもっている。このネットワークを 2-3-2 のネットワークと呼ぶことにする。

入力ベクトルが入力層に与えられると、結合荷重を通して中間層へ送られ、次いで出力層へと順にデータが送られて出力層の値が決まる。ネットワーク全体の入出力関係は結合荷重の値によって一意に決定される。中間層での出力分布状況は、ネットワークがどのように入出力関係を実現しているかということに大きくかかわってくる。これはネットワークの内部表現と呼ばれる。

2.3 学習

学習とは、ネットワークに与えられる入力に対して望ましい出力が得られるように結合荷重の値を変化させていくことを指す。ここでは、階層型ネットワークで最も一般的に使用されている誤差逆伝搬法 (BP 法) を紹介する。BP 法は Rumenlhart らにより提案され、出力層における誤差を最急降下によって減少させようとするもので、外部からは直接誤差を受け取ることができない中間層の結合荷重の値も学習することができる。BP 法により学習されるネットワークモデルは BP モデルと呼ばれる。BP モデルはフィードバック結合を持たず、信号は入力層から出力層へ一方向に伝わる。

BP 法による学習は次のように行なわれる。まず、ネットワークに実現させたい問題から、入力値とその入力に対して得られる出力値のパターンをサンプリングし、学習パターンセットを生成する。ネットワークは入力層に提示された入力パターンを出力層に伝播させ、実際に得られる出力と望ましい出力とを比較する。比較することにより得られる誤差が減少するように結合荷重の値を修正し、この誤差が一定値以下になるまで繰り返す。

ここで、第 p 番目の学習パターンを与えたときの出力層第 i 番目の素子が取るべき望ましい出力を $t_{p,i}$ とおき、実際にネットワークが出した値を $a_{p,i}$ とおくと、二乗誤差は、

$$E = \frac{1}{2}\sum_{p,i}(t_{p,i} - a_{p,i})^2 \tag{2.3}$$

この誤差 E が減少するように、勾配法を用いて結合荷重の値を修正してやる。第 l 番目の層の第 i 素子と、第 l-1 番目の層の第 j 素子との間の結合荷重の更新量 Δw_{ij}^{l} は次のように与えられる。

$$\Delta w_{ij}^{l} = -\eta \frac{\partial E}{\partial w_{ij}^{l}} \tag{2.4}$$

但し、η は学習率と呼ばれる正の定数である。ここでこの (2.4) 式を詳しく見てみることにする。まず、

$$\frac{\partial E}{\partial a_i} = t_i - a_i \tag{2.5}$$

$$\frac{\partial E}{\partial u_i} = \frac{\partial E}{\partial a_i}\frac{\partial a_i}{\partial u_i} \tag{2.6}$$

この$\frac{\partial a_i}{\partial u_i}$は、出力関数の微分に他ならない。ここで出力関数に前節で示したシグモイド関数をしようすると、

$$a = f(u) = \frac{1}{1 + exp(-u)} \tag{2.7}$$

このシグモイド関数を微分すると、

$$f'(u) = \frac{exp(-u)}{(1 + exp(-u))^2} = f(u)\,(1 - f(u)) \tag{2.8}$$

よって、(2.6) 式は

$$\frac{\partial E}{\partial u_i} = (t_i - a_i)a_i(1 - a_i) \tag{2.9}$$

$\frac{\partial E}{\partial u_i}$ が求まったので、つぎに目的である$\frac{\partial E}{\partial w_{ij}}$を求める。

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial u_i}\frac{\partial u_i}{\partial w_{ij}} \tag{2.10}$$

ここで、

$$u_i^{l} = \sum_j w_{ij}^{l} a_j^{l-1} \tag{2.11}$$

より

$$\frac{\partial u_i}{\partial w_{ij}} = a_j^{l-1} \tag{2.12}$$

(2.9)(2.10)(2.12) 式より

$$\begin{aligned} \frac{\partial E}{\partial w_{ij}^l} &= (t_i - a_i^l) a_i^l (1 - a_i^l) a_i^{l-1} (\text{出力層}) \\ &= a_i^l (1 - a_i^l) \sum_k w_{ki}^{l+1} \delta_k^{l+1} (\text{その他の層}) \end{aligned} \tag{2.13}$$

となる。これをすべてのパターンについて加算したものが実際にネットワークに適用されるべき結合荷重の更新量となるのだが、実際には収束速度を早めるために以下のような式を用いる。

$$\Delta w_{ij}^t = -\eta \frac{\partial E}{\partial w_{ij}} + \alpha \Delta w_{ij}^{t-1} \tag{2.14}$$

このように、前回の更新量を付加したものが使用される。この右辺の第 2 項は慣性項、α は慣性項係数とよばれるもので、1 未満の正の係数である。

2.4 階層型ニューラルネットワークの特徴

丹ら [1] によると、階層型ニューラルネットワークの特徴として、以下のような利点が考えられる。

2.4.1 ネットワークの利点

1. 非同期動作が可能
   本質的に、データがそろった時点で、各素子ごとに動作を開始して良いわけで、素子ごとの同期より大きな規模の同期は必要ではない。

2. 動作の並列性が高い
   各層ごとに着目してみると、数多くのデータ（パターン）を処理する際に並列性が

存在し、各素子で見てみると、同一層内で他の素子間とは無関係に処理をおこなっているところから並列性が存在し、その素子内部で見ても、入力の積和操作に並列性が存在している。

3. 機能が分散している
   普通、学習により得られたネットワークでは、どの部分が何をするという役割分担が明確ではない。

4. 記憶が分散している
   ニューラルネットワークではすべての情報は素子間の結合荷重の値として記憶されている。よってすべてのネットワークに分散して存在する。

5. フォールトトレランスの能力を持つ
   ニューラルネットワークにおいて、入力にノイズが混入しても、その能力が大きく低下するわけではない、といったようなことから、何らかのフォールトトレランスが存在しているということが知られている。しかし、積極的にフォールトトレランスを向上させる方法はあまり報告されておらず、さまざまな基本的な問題が残されている。これについては、第 4 章で詳しく述べることにする。

6. 汎化能力を持つ
   汎化能力とは、学習時に与えられた例題以外に対しても妥当な解を得る能力のことである。すなわち、与えられたパターンをまる暗記しているわけではなく、ネットワークがその問題の規則性、法則性を導き出しているために、未学習パターンの入力においても何らかの補間が行なわれる事が期待されるのである。

7. 学習能力を持つ
   前節で見たように、階層型ニューラルネットワークではネットワークの入出力関係を決定するために結合荷重の値の修正を、この学習という過程で行なっている。

2.4.2 誤差逆伝搬法の問題点

前節における誤差逆伝搬法（BP 法）には、いくつか問題があり、石川ら [17] は以下に示すような問題点を指摘している。

- 極小値問題： 結合荷重を修正するのに最急降下法を用いているため、解が二乗出力誤差和の極小値に陥る危険性がある。
- 学習所要時間が大： 二乗出力誤差和の最急降下法を使用しているため、解が最適点に近づくと傾斜が小さくなり、毎回の結合荷重の修正量が小さくなるため学習速度が遅くなる。ネットワークの規模が大きくなればなるほど問題は深刻化する。
- 試行錯誤的学習： ネットワークの大きさが過小であると十分学習できず、逆に過大であると過剰学習のためノイズまで学習してしまい汎化能力が減少する。
- 中間層の解釈が困難： BP 法の利点として、中間層上に規則性が出現し、その意味を理解できる例もあるのだが、多くの場合、中間層の解釈が困難なブラックボックス的なネットワークが得られる。

2.5 シミュレーションで使用する問題

2.5.1 XOR

XOR は、入力 2、出力 1、パターン数 4 とシンプルなため、実験を行なう際にたいへん便利なベンチマークである。教師パターンを表で表すと、

表 1:XOR 問題

Pat-No.	入力 パターン		教師パターン
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

である。数多くのニューラルネットワークの学習問題の例題として使用されており、3 層の階層型ニューラルネットワークにおいては、中間層が 2 つあればネットワーク構築可能であることも知られている。また、入出力関係がシンプルなところから、内部表現の考察が容易になるという点でも有益である。

同じような論理関数に AND や OR が存在するが、2 層のニューラルネットワークで実現可能であることから学習に使用する問題としては適切ではないと思われる。この理由か

らも、XOR を学習問題に採用した。具体的には、3.2.2 節の「影響を及ぼす特徴量」で示すことにする。

2.5.2 正規分布による帯状パターン

前節の XOR は論理関数であるため、これを学習させたネットワークには汎化の概念が適用できない。一般的に汎化能力を論ずる場合にはパターン認識や関数補間問題を用いる。学習したパターンに対して未学習のパターンが正しく識別できるか、ということを汎化能力の評価としたり、関数からサンプリングした値を使用して学習を行ない、その学習に用いなかったサンプリング値を正しくその関数の法則性にのっとって補間しているか、ということを汎化能力の評価に使用する。しかし、入出力次元が増加するほど内部状態や出力の分布状況を調べるのは困難となる。そこで本研究では、正規分布による帯状パターンという単純な問題を新たに作ることで、これらの解析を容易にすることにした。

正規分布による帯状パターンとは、下図に示すように y=0.5 を中心とした標準偏差 0.1 の正規分布から値をサンプリングする問題である。2 入力（2 次元）、1 出力（1 次元）のため様々な解析が容易である。

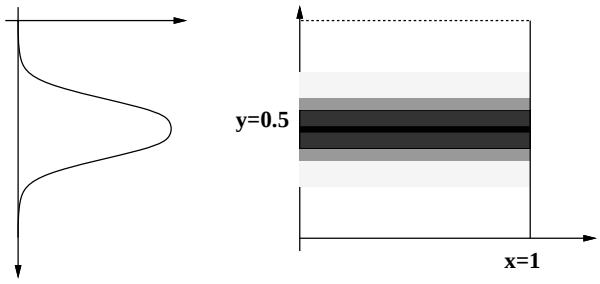

図 2.4: 新しい問題: 正規分布の帯状パターン

この正規分布において、サンプルポイントをどのように選択するかによって、2 通りの考え方が出てくる。

- 連続値サンプリング
- 離散値サンプリング

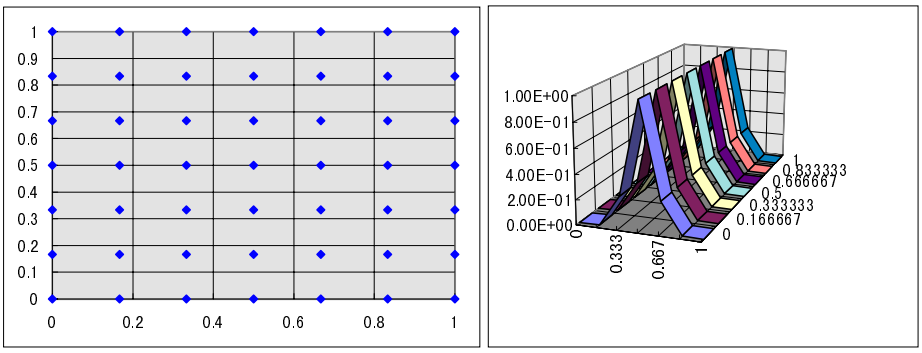

図 2.5: 帯状パターン問題の連続値サンプリング

帯状パターンの連続値サンプリングとは、定義域内 $(0 \leq x \leq 1)$ で等間隔にサンプリングされた点において、各々、正規分布関数の値をとるものである。この場合、連続した関数の形を学習する問題となる。サンプル点をとるべき位置は、関数の曲面と関連して決まることであり、関数と独立に選ぶ場合にはランダムでも等間隔でもあまり本質的な違いはない。よって、本研究では、x 軸、y 軸方向に 7 点を等間隔にとることとした。

一方、**帯状パターンの離散値サンプリング**とは、ランダムにサンプリングされた点の y 座標の値により、正規分布に従って、確率的に出力 0 と 1 を発生させたものである。これは、 0 と 1 の点を分離する「分布識別問題」となる。分離平面がどこを通るかに依存してサンプルの存在場所が重要になってくるという意味では連続値サンプリングの場合と同じである。しかし、サンプルポイントの値は、0 または 1 のいずれかの値であるため、連続値サンプリングのように中間値をとれない。よって変化の生じるところでは、1 と 0 のサンプルが混在するような領域ができる。また、値が確率的に決まることにより、ノイズを含ませていることになる。

実際に実験に使用したモデルを下図に示す。縦軸横軸ともに 0〜1 の間で、$y=0.5$ を中心とする正規分布にしたがって 100 ポイントをランダムに選んだ。図中の●は 1 を、○は 0 を表している。汎化能力を調べるために同様のデータを後 2 つ用意したが、それらについては付録にデータ値と共に示すことにする。

帯状パターン問題のシミュレーションについては、3 章、4 章で具体的な結果を示す。

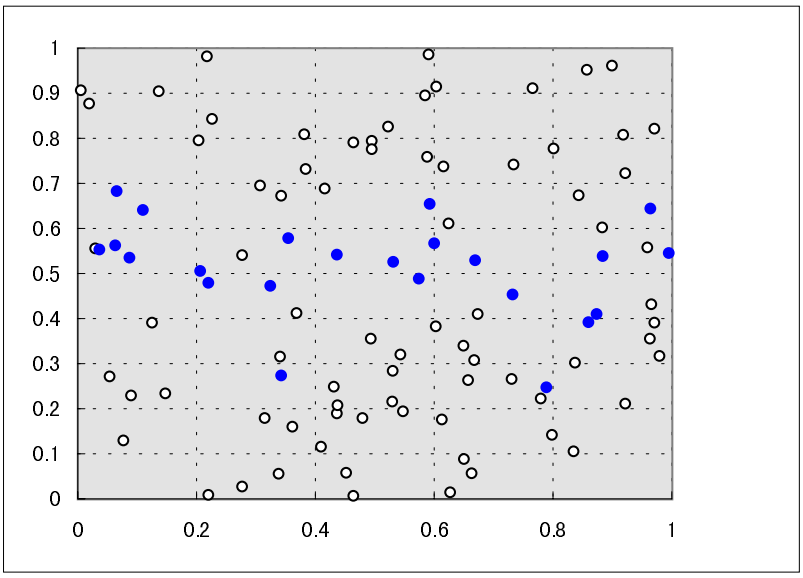

図 2.6: 帯状パターン問題の離散値サンプリング

第 3 章

汎化能力

本章では、汎化能力の定義、汎化能力に影響を与える根本的な要因について考察し、この根本要因に影響を与えるニューラルネットワークの特徴量について検討する。また、現在行なわれている汎化能力の向上手法を調べる。最後に、テストパターンとクロスバリデーションと呼ばれる 2 つの汎化能力評価方法について述べ、シミュレーション結果を示す。

3.1 汎化能力の定義

汎化能力とは、広くいうと、与えられたサンプルから問題の規則性、法則性を導き出す能力のことである。学習系においては、「学習時に与えられた例題以外に対しても適切な応答をするような性質」と定義され、未学習パターンに対する性能（誤差や正答率）を実験的に測定することにより評価されることが多い。

汎化能力は大まかにいって、耐ノイズ性と補間能力 (内挿、外挿) に分類できると考えられる。これら各々のケースに応じた汎化能力に関する検討が必要である。

本章では、正規分布の帯状パターン問題に対して 3 層の階層型ニューラルネットワークの特徴量と汎化能力の関係について検討する。

3.2 汎化能力の特徴

汎化能力は、工学的な応用に際して非常に重要であるため、多くの研究成果が報告されている。これらのほとんどは、汎化能力を低下させる根本要因を考察し、これらに影響を与えるような特徴をとりあげ、これを最適化することによって、汎化能力を向上させようとするものである。こうして提案された学習手法などは、ネットワークの特徴に影響を与えるため、汎化能力だけではなく、耐故障性をはじめとした他のネットワークの性質についても何らかの影響を与えるものと考えられる。

この節では、同様の流れに従い、まず、汎化能力の根本要因について考察し、それに影響を及ぼすであろう特徴量について調べ、続いて、今までに提案されている汎化能力向上手法について述べる。

3.2.1 汎化能力低下の根本要因

汎化能力に影響を与える要因を考える上で、そもそも汎化能力に差異を生じる原因ともいうべき根本要因は何かということが問題となる。これについて、「過剰学習」と「解の非一意性」の二つが考えられる。

1. **過剰学習**

   過剰学習とは、学習回数の増加と共に学習パタンに対する誤差が減少する一方で、未学習パターンに対する誤差が増加する現象である。

   十分な中間層ノード数を設定すれば 3 層の階層型ニューラルネットワークが任意の連続関数を任意の精度で近似できることが証明されている。このため、過剰な中間層ユニット数を設定した場合、入出力関係の表現方法が豊富になり過ぎるため、与えられた学習パタンに過剰に適応し、結果として汎化能力が劣化することになる。

   こうした過剰学習を抑制するには、冗長な中間層ユニットや荷重係数を削除してやればよい。従って、一般的に、中間層ユニットの少ないネットワークの方が、汎化能力に優れるといわれている。

2. **解の非一意性**

   一般に、与えられた学習パターンを満たすための関数は、無限に存在する。従って、学習パターン以外の値を評価しようとする汎化能力においては、求められている関

数と、学習系が実現した関数とが完全に一致することは一般に望めない。これが解の非一意性である。

階層型ニューラルネットワークの場合、与えられたパターンについての誤差が要求を満たすような結合荷重の組、すなわち、ネットワークによって実現される関数は無数に存在することに相当する。

3.2.2 影響を及ぼす特徴量

上述の根本要因に影響を与えるような特徴量（内部表現）として、本研究では過去3年の電子情報学会ニューロコンピューティング信学技法を文献調査のうえ、以下のようなものを取り上げた。

1. 中間層表現
2. 中間層素子の平均・分散（パターン方向と空間方向）
3. 分離直線（中間層素子による）
4. 中間層の微係数
5. 関数の滑らかさ
6. 結合荷重の統計的性質
7. 出力分布

本節では、XORを学習させた2-4-1のネットワークを用いて、具体的にこれらの特徴量の例を示し、結果を示す図の見方について説明する。

1. 中間層表現

ネットワークの結合荷重や中間層ノードの出力値は、学習によって自由に決定される。3層ネットワークの中間層出力の場合、分散的な表現と局所的な表現の2通りが考えられる。

分散表現とは、情報が複数の中間層素子により表現されており、その役割が均等であるもの、つまり、どの入力パターンに対しても偏りが存在せず、各々の素子が働いているよ

うな内部表現をいう。これは、出力素子から見て、各中間層素子の重要度がおおむね均等であることを意味している。

逆に**局所表現**とは、情報を一部の素子が持っているもの、つまり、ある入力パターンに対して、ネットワークの出力が少数の中間層素子からの入力によって決定される表現のことである。

ネットワークが局所表現されている場合、情報を一部の素子に偏った状態で持っているため、特定の素子からの情報の欠如や誤りにより、ある入力に対して誤りを起こしやすくなるものと考えられる。一方、分散表現では同じ情報を多数の素子が持っているため、全体としては大きな影響を及ぼさないものと期待される。しかしながら、実際に実験をおこなうと耐故障化学習 (FTBP 法) では局所解により完全に多重化されている方が故障に強いという結果が得られている [1]。

2. 中間層出力値の平均と分散

中間層の各素子が出力する値の平均と分散を調べる。この場合、ある入力パターンに対して、すべての中間層素子がどのような値を出力するのかを示す**空間方向**の平均・分散と、すべての入力パターンに対して各々の中間層がどのような値を出力するのかを示す**パターン方向**の平均・分散とが考えられる。

まず例として XOR 問題を使って学習したネットワークの中間層出力値を 3.1図に示す。ただし、実線が素子 1、太実線が素子 2、長点線が素子 3、短実線が素子 4 を表している。

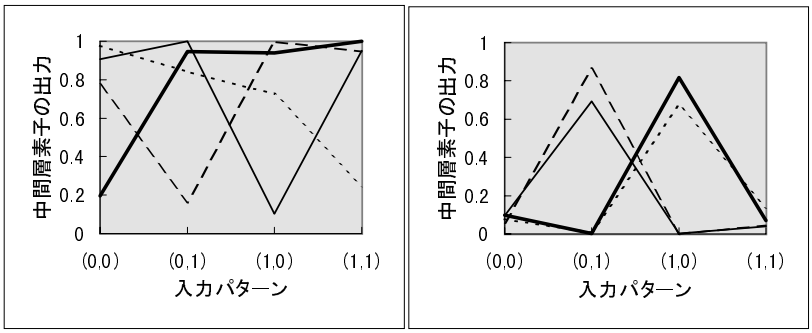

図 3.1: 中間層素子の出力

ここで、先ほどの 3.1図で示した 2 つのネットワーク例をつかって各パターンに対する中間層素子出力の平均と分散を比較してみる。

平均と分散の値は、以下の式により導かれる。

$$s^2 = \frac{S}{n-1} = \frac{1}{n-1}\sum_i^n (x_i - \bar{x})^2 \tag{3.1}$$

ただし、平均$\bar{x}$ および平方和 S は

$$\bar{x} = \frac{1}{n}(x_1 + x_2 + \cdots + x_n) = \frac{1}{n}\sum_i^n x_i \tag{3.2}$$

$$S = \sum_i^n (x_i - \bar{x})^2 = \sum_i^n x_i^2 - \frac{(\sum_i^n x_i)^2}{n} \tag{3.3}$$

空間方向の平均と分散

Pat-No.	入力　パターン	教師パターン	平均-左図	分散-左図	平均-右図	分散-右図
0	0　0	0	0.7135	0.1262	0.0831	0.000357
1	0　1	1	0.7376	0.1513	0.3911	0.204249
2	1　0	1	0.6911	0.1672	0.3724	0.187293
3	1　1	0	0.7857	0.1290	0.0734	0.002120

この分散の値は、どのくらい値がバラついているかを表しており、値が大きければ大きいほどあるパターンにおいて分散性が高いことを表し、逆に 0 に近ければ局所性が高いことを示している。

次に、各々の中間層の素子に注目してみる。いままでは、空間方向について議論してきたわけであるが、パターン方向について同様に平均と分散の値を調べてみると、

パターン方向の平均と分散

素子 No.	1	2	3	4
平均-左図	0.740068	0.770547	0.719788	0.697689
分散-左図	0.18143	0.147514	0.147083	0.100494
平均-右図	0.207936	0.247019	0.241373	0.223857
分散-右図	0.106534	0.145728	0.171801	0.0918071

空間方向に分散を見る場合、値が大きいということは素子が入力されるパターンに対して変化が激しい、つまりよく働いていることを表している。逆に、値が 0 に近いというこ

とは入力されるパターンに対してあまり働いていないということになる。つまり、与えられるパターンに関係なく一定の値を出力しているわけで、その素子はバイアス的な役割をはたしていることがわかる。

しかし、これらの値が直接、次の層（ここでは出力層）への入力となるわけではない。これらの値に結合荷重が掛けられた値が出力層へ入力されることになる。つまり、部分的に高い値をとっていても、その結合荷重が 0 に近ければ、その中間層素子は何もしていないことになる。

そこで、結合荷重を掛けた値、つまり出力層へ入力される値を調べ 3.2図に示してみた。

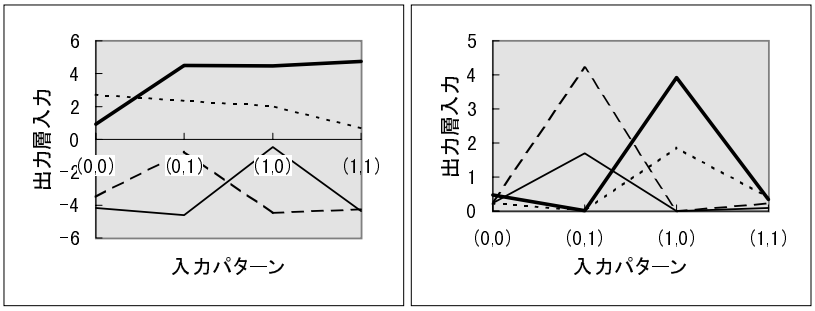

図 3.2: 出力層素子への入力

これらは中間層素子の出力に結合荷重をかけた値である。横軸には入力パターンを、縦軸には出力の値、すなわち出力層素子への入力値を表している。同じ素子からの入力は折れ線で結ばれ、パターンが変わったときの変化の様子がわかるようになっている。

3.2図（左）は複数のノードが競合し合うことにより構成されているのに対して、3.2図（右）ではあるパターンに対して特定の素子が高い値を示している。このように 3.2図からは中間層素子がどのような働きをしているかをつかむのに適している。最終的に出力層へ入力される値を知るということは、以後、故障の生じたネットワークを考えていく上で非常に有効な表現法となる。

3. 中間層素子による分離直線

次に中間層素子による分離直線の図の見方について説明する。本来、XOR は 0 と 1 からできた論理関数であるが、これを学習したネットワークは連続関数を実現する。ここで

は、XOR を学習し終ったネットワークに 0 または 1 以外のサンプルポイントを入力し、出力を評価する。ここで中間層素子が出力する値を 0〜0.5 の 0 領域と 0.5〜1 までの 1 領域に分離する直線をつくり、これを分離直線と呼ぶことにする。この分離直線を見ることにより、ネットワークが与えられたサンプルパターンをどのように分離しようとしているかをみることができる。AND と OR を使って分離直線の例を 3.3図に示す。ただし、●は 1 を、○は 0 を表しているものとする。

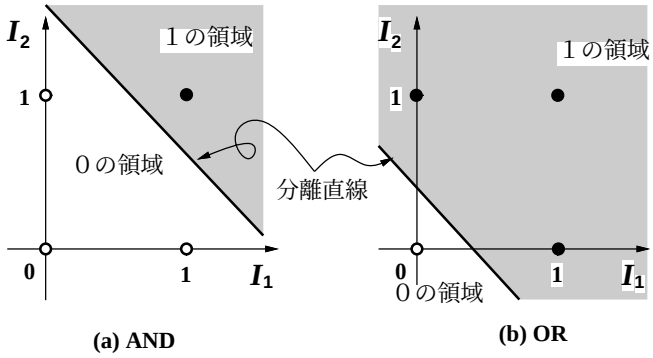

図 3.3: 中間層素子出力による分離直線の例

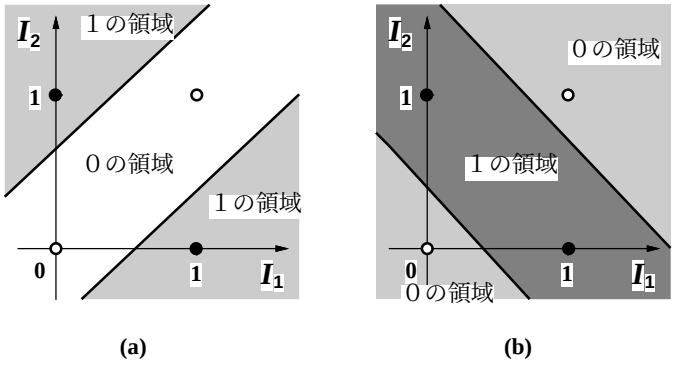

図 3.4: XOR の分離直線

2 入力 1 出力の場合、1 本の分離直線を引くことができる。3.3図からわかるように、AND や OR の場合、問題の性質から 1 本の分離直線で表現できる。つまり、2 つの入力素子と 1 つの出力素子があればネットワークを構築できることになり、中間層素子をまったく必要としないことになる。一方、XOR の場合は、3.4図に示すように、分離直線が 2

本以上なければ 0 または 1 の領域に分離できないことがわかる。つまり、中間層素子が 2 つ以上必要となるわけである。このような性質が本研究において XOR を使用した理由の一つである。

ここで、先ほど 3.1図に示した例がどのような分離直線をとっているかを示す。

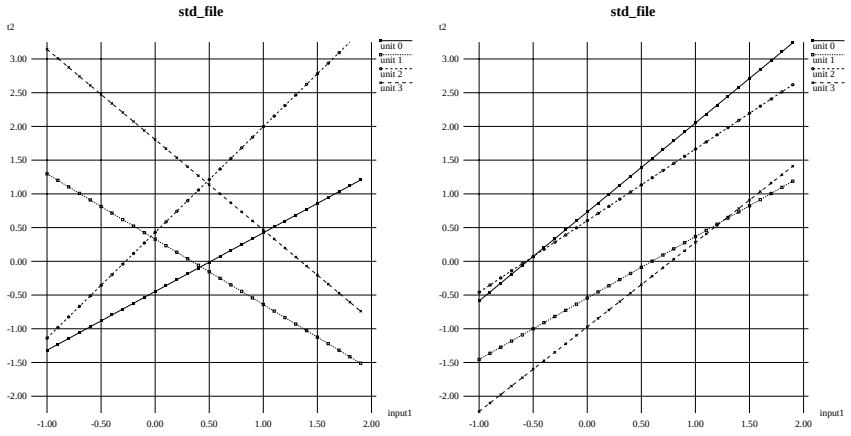

図 3.5: 中間層素子による分離直線

XOR の場合、分離直線が 2 本あれば領域を分離できることを示したが、3.5図の場合、分離直線は中間層素子の個数だけ得られるため、2-4-1 のネットワークでは 4 本の分離直線が引かれている。しかし、この分離直線は、中間層素子にのみ着目しているわけで、たとえ直線で区切られたとしても、出力層への結合荷重の値が 0 に近ければ、その中間層素子は働いていないことを表しており、つまりは分離直線も働いていないことになる。

4. 中間層の微係数

中間層素子の出力において、入力のばらつきに対して出力がいかに安定するかを表す指標として、この中間層微係数を導入する。式を用いて表すと、まず出力層からの出力は

$$o_i^3 = f(\sum_j w_{ij}^{32} o_j^2) = f\left(\sum_j w_{ij}^{32} f(\sum_k w_{jk}^{21} i_k^1)\right) \tag{3.4}$$

と表すことができる。ここで、

$$I = \sum_j w_{ij}^{32} f(\sum_k w_{jk}^{21} i_k^1) \tag{3.5}$$

$$J = \sum_k w_{jk}^{21} i_k^1 \tag{3.6}$$

とおくと、学習パターンにノイズが混入された場合の出力層出力の変化量は、

$$\frac{\partial o_i^3}{\partial i_k^1} = \frac{\partial f(I)}{\partial i_k^1} \tag{3.7}$$

$$= \frac{\partial f(I)}{\partial I}\frac{\partial I}{\partial i_k^1} \tag{3.8}$$

$$= \frac{\partial f(I)}{\partial I}\frac{\partial \sum_j w_{ij}^{32} 0_j^2}{\partial i_k^1} \tag{3.9}$$

$$= f'(I)\sum_j w_{ij}^{32}\frac{\partial f(J)}{\partial i_k^1} \tag{3.10}$$

ここで、$\frac{\partial f(J)}{\partial i_k^1}$ は

$$\frac{\partial f(J)}{\partial i_k^1} = \frac{\partial f(J)}{\partial J}\frac{\partial J}{\partial i_k^1} \tag{3.11}$$

$$= f'(J)w_{jk}^{21} \tag{3.12}$$

となる。よって、

$$\frac{\partial o_i^3}{\partial i_k^1} = f'(I)\sum_{j=1}^{n_2} w_{ij}^{32} w_{jk}^{21} f'(J) \tag{3.13}$$

と表すことができる。出力層出力の変化量を小さくすることができれば、そのネットワークは耐故障性に優れているといえる。つまり、これらの項それぞれが小さくなればよいわけであるが、ここでは中間層ユニット出力値に着目し、上記の条件のうち $f'(J)w_{jk}^{21}$ を小さくすることを汎化能力に対して望ましい特性の一つとして採用する。

5. 関数の滑らかさ

学習によ得られたネットワークが、与えられたサンプルパターンに対して、どのような値をとって実現しているかを調べることは重要である。前節で示した通り、一般的に緻密

に学習させすぎたネットワークは汎化能力は低下すると言われている。そこで、出力される関数がどのていど滑らかであるのか、「関数の滑らかさ」を表して一つの指標を与えることにした。関数の滑らかさを表す 1 つの方法は、スプライン関数の理論などでよく知られているように、関数の 2 階導関数のノルムの 2 乗の積分を用いることである。先ほどの微係数の式を利用して滑らかさを表すと、

$$\frac{\partial o_i^3}{\partial i_k^1} = \sum_j w_{ij}^{32} w_{jk}^{21} [f(I)f(J) - f(I)^2 f(J) - f(I)f(J)^2 + f(I)^2 f(J)^2] \tag{3.14}$$

と展開できる。ここで上式をもう一度微分してやると、

$$\begin{aligned} \{o_i^3\}'' \quad = \quad & \sum_j w_{ij}^{32} (w_{jk}^{21})^2 f(I)f(J)(1-f(J)) \\ & [(1-f(I)) \sum_j w_{ij}^{32} f(J) \\ & \quad \{1 - 2f(I) - f(J) + 2f(I)f(J)\} \\ & \quad + \{1 - f(I) - 2f(J) + 2f(I)f(J)\}] \end{aligned} \tag{3.15}$$

6. 結合荷重の統計的性質

1 の中間層表現の所でもふれたが、いかに複雑な中間層表現、分離直線を得たとしても出力層へ入力される結合荷重の値が 0 であれば、それらの働きはなされていないことになる。上述の図における分離直線出力を見ると、次の表のようになる。結合荷重という最も基本的かつ単純な統計値にネットワークの性質があらわれる可能性がある。

結合荷重の統計的性質

		平均	分散	歪度	尖度
A	入力層→中間層	3.241	1.249	25.076	85.428
	中間層→出力層	3.521	1.464	20.281	62.149
	ネットワーク全体	3.324	1.322	23.026	75.525
B	入力層→中間層	3.312	0.890	63.185	282.869
	中間層→出力層	3.665	1.008	59.161	257.593
	ネットワーク全体	3.416	0.940	59.285	260.493

7. 出力分布

実際に学習により得られたネットワークが出力する値を見る。0.5 以上を黒、それ未満を白とした。範囲は、0 と 1 のまわりの様子を観察しやすいように、縦軸、横軸ともに-1 から 2 までとした。

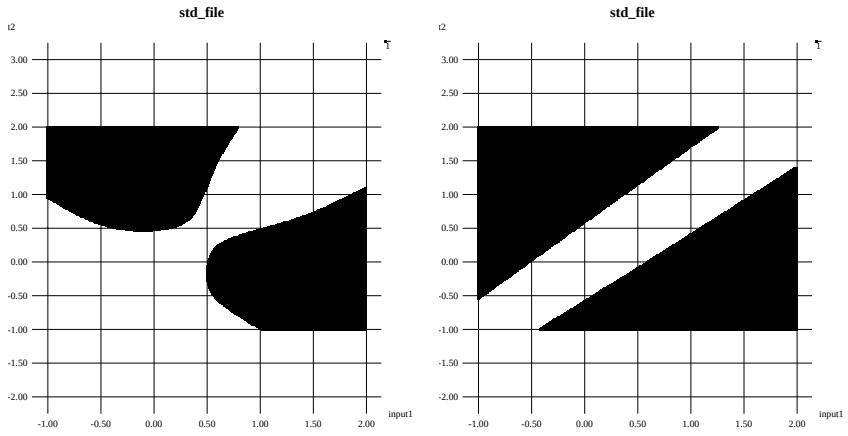

図 3.6: 中間層素子による出力分布

3.2.3 向上手法

汎化能力を向上させようとする手法として様々な学習法が提案されている。現在向上手法として提案されている、ネットワーク構造の最適化、構造化学習、DT 学習、正則化学習、射影学習について以下に簡単にまとめておく。

1. **ネットワーク構造の最適化**
   ネットワークの大きさをむやみに大きくしたり、層の数を増やしたりすると、過剰学習が起るものと考えられる。逆にネットワークの規模が小さすぎると、能力が不足して問題を十分に学習できない。層や中間層素子数を何らかの方法で最適な値にしてやることで、汎化能力の向上が期待できる。あらかじめ、ネットワーク構造を決めてから学習を行う立場と、学習中に構造も決めていこうとする立場があり、次に示す構造化学習は後者の方法である。

2. **構造化学習**

何らかの方法で適切な大きさのネットワークを学習によって求めることを構造化学習と呼ぶ。そのいくつかを簡単に述べる。なお、以下の式中における E は (2.3) 式に示した自乗誤差評価であり、BP 学習における評価関数である。

- **結合重みの自乗和最小化評価**

  Plaut ら [11] の提案によるもので、次式のように結合荷重の自乗和を付加的評価とし、Jが減少するように勾配法を用いて結合荷重の値を修正する。

$$J = E + \frac{h}{2}\sum_{ij} w_{ij}^2 \tag{3.16}$$

  ただし、w_{ij} はノード jからノード i への結合荷重値、h は付加的評価項の重みである。

  自乗和評価は直観的に分かりやすいが、重みの修正量が w_{ij} に比例することから、重みの絶対値が小さくなると結合荷重が 0 に近付く速度が比例的に小さくなり、完全に消滅するには時間がかかる。結合荷重の絶対値が大きいと結合荷重の減衰量が大きくなり、学習の劣化が著しくなるという問題点が指摘されている。

- **複雑さ最小化学習**

  Rumelhart ら [12] の提案によるもので、ネットワークの複雑さを表す次式のCを付加的評価とし、Jが減少するように勾配法を用いて結合荷重の値を修正する。

$$J = \lambda E + (1-\lambda)C \tag{3.17}$$

$$C = \sum_{ij} \frac{w_{ij}^2}{1+w_{ij}^2} + \sum_{ij} \frac{w_{ij}^2}{1+\sum_k w_{ij}^2} \tag{3.18}$$

  ただし、$(1-\lambda)$ は付加的評価項の相対的荷重である。

  (3.3) 式の第 1 項は結合荷重に関する複雑さを、第 2 項は結合の仕方に関する複雑さを表現している。結合荷重の絶対値が小さな場合、近似的に結合荷重の自乗和評価となるため、結合荷重が完全に 0 になるのに時間がかかるという問題点が指摘されている。

- **忘却付構造学習**

  石川ら [13] の提案によるもので、荷重の絶対値和を付加的評価とし、Jが減少するように勾配法を用いて結合荷重の値を修正する。

$$J = E + \varepsilon' \sum_{ij} |w_{ij}| \tag{3.19}$$

  ただし、ε' は付加的評価項の荷重であり、$\varepsilon = \varepsilon'\eta$ が忘却量に相当する。ここで、η は学習率である。この場合には、忘却量は一定値 η であり、結合荷重の自乗和最小化評価のような問題は無いと考えられる。

- **エントロピー最小化学習**

  内田ら [14] の提案によるもので、ネットワークの複雑さをエントロピーにより表現し、これを付加的評価とする。

$$J = E + \lambda H \tag{3.20}$$

$$H = -\sum_{ij} P_{ij} log_2 P_{ij} \tag{3.21}$$

$$P_{ij} = \frac{|w_{ij}|}{\sum_{ij} |w_{ij}|} \tag{3.22}$$

  ただし、λ は付加的評価項の荷重である。付加的評価の意味付けが明確である点に特徴がある。

- **側抑制学習**

  安井ら [15] の提案によるもので、各ノードへの入力結合間に相互抑制を導入するものである。

$$\Delta w'_{ij} = \Delta w_{ij} - \varepsilon sgn(w_{ij}) \sum_{k=1,\neq j}^{n} |w_{ik}| \tag{3.23}$$

  ただし、Δwij は BP 学習による結合荷重の修正量を、右辺第 2 項は付加的評価による結合荷重の修正量を、sgn(x) は x $>$ 0 で 1、x=0 で 0、x$<$0 で -1 をとる符合関数、εは付加項の相対的荷重である。

安井らは上式の結合荷重の修正量によって学習法を定義している。つまり最小化する評価関数 J を明示的に示すと、

$$J = E + \varepsilon' \sum_i \sum_{j \neq k, j<k} |w_{ij}||w_{ik}| \tag{3.24}$$

ただし、$\varepsilon' = \varepsilon\eta$ は付加的評価項の荷重である。

- **Optimal Brain Damage (OBD)**
  上記までの方法が、新たな評価項を付加する方法であったのに対して、このOBD では不要な結合を削除してしまうといった方法をとる。LeCun ら [16] の提案によるもので、BP 法で学習した後に各結合荷重の w_{ij} を削除した場合の評価関数の劣化を下式により予測し、これらのうち最も小さいものを削除して再び BP 法で学習を行なう。この手順を評価が大幅に劣化する直前まで繰り返す。

$$s_{ij} = \frac{1}{2}\frac{\partial^2 E}{\partial w_{ij}^2} w_{ij}^2 \tag{3.25}$$

  ここで、2 次微分行列が対角行列で近似可能であると仮定している。

3. **DT 学習**
   BP 学習法では、学習パターンと教師パターンとの比較を逐次行なうことで学習がなされ、新しくパターンを学習するには、それまでの学習経過に関係なく結合荷重の修正を行なっていた。しかし、この DT 学習法 (**D**issimilarity between two **T**arget patterns) は、教師パターン間の類似性と非類似性を考慮することでパターンの特徴的な部分を強調させて学習したもので、教師パターン相互間の関連性を学習に利用する。類似性と非類似性は、現在学習している教師パターンと前回学習した教師パターンの値の差で表され、BP 学習法の誤差に付加される。学習効率が向上することから過剰学習による汎化能力の低下が防げるため、パラメータの選び方によって汎化能力を向上させることができる。

4. **射影学習**
   最適汎化学習の一つに、小川ら [6] により考案された射影学習がある。作用素 A の値域を $R(A)$ で表し、A の共役作用素を A^*で表す。射影学習とは、任意の $f \in H$に対して、$f_0 = BAf$が f の $R(A^*)$ における最良近似となるような学習である。射影学習では、H に含まれる個々の関数 fに対して、f自身がわからないにもかかわらず、その最良近似 f_0を得ることができる。

5. **正則化学習**

正則化法とは、赤穂ら [22] によると、少ないデータしか得られない場合でもデータ点のまわりで、ぼかし行なうことによって確率密度関数を推定するという方法である。確率密度関数の推定問題は不良設定問題である。不良設定問題を解くのに古くから用いられている手法の一つが正則化法である。この正則化法を確率密度関数 $f(t)$ の推定に即して書くと、

$$Af(t) \equiv \int_{-\infty}^{\infty} u(x-t)f(t)dt = F(x) \tag{3.26}$$

を n 個のデータから解くことを考える (u(x) は unit step function)。確率分布関数は経験データから安定に近似できるが、それを直接上の微分方程式に入れてといても不安定な解しか得られない。そこで次の汎関数の最小化問題を考える。

$$R_r(\hat{f}, F) = \rho^2(A\hat{f}, F) + \gamma\Omega(\hat{f}) \tag{3.27}$$

ここでρは関数距離、Ωはいくつかの条件を満たす汎関数、γは正の実定数で正則化パラメタと呼ばれる。右辺第一項は誤差を表しており、第二項に正則化による評価関数が与えられており、この項を付加することにより正則化学習がなされる。この問題は安定に解くことができてFの近似の度合に応じてγ の値を十分ゆったり小さくして行けば漸近的に$\hat{f}$は $Af = F$の真の解に収束することが知られている。

他の学習法が汎化能力を向上させるために、ネットワークのサイズを縮小することにより過剰学習を抑制しようとする方法であるに対して、前項の射映学習と、この正則化学習は解の非一意性を減らすことにより汎化能力を向上させようとする方法といえる。

3.3 汎化能力の評価方法

上記でふれたとおり汎化能力の評価は、未学習パターンに対する性能、すなわち誤差や正答率を測定することにより行なわれる。その方法としては以下の 2 つが考えられる。

- テストパターン
- クロスバリデーション

テストパターンとは、汎化能力を評価するための最も一般的な方法であり、その手順には、以下のようなものがある。ネットワークに学習させる入力値、出力値に相当する全パターンの集合（これを母集団と呼ぶことにする）の中から、実際に学習を行ないネットワークを形成するためのパターンの集合を選び出す。このとき注意しなければならない事は、ただ無作為にパターンを選択するのではなく、例えば出力結果が 0 または 1 の二値であったとするならば、0 と 1 各々のカテゴリーからうまく（確率的に）選択してやらなければならない。選択パターンが決定すれば、そのパターンを使って学習を行なう。本節では BP 学習法を使用する。学習により得られたネットワークに、評価パターンを実行することで汎化能力の評価を行なう。ここでは、平均自乗和誤差の大きさを調べ、これが小さいネットワークほど汎化能力が高いものと考える。

クロスバリデーション [18] とは、テストパターンによる汎化能力の評価を複数繰り返す方法である。つまり、データの分割の仕方を変え、学習と評価を繰り返すことにより、もとのデータの持つ情報をできるだけ利用しようとするものである。3.7図にクロスバリデーション法の評価手順を示す。

3.4 シミュレーション

3.4.1 BP モデルの基本パラメータ

2.4.2 節でふれたとおり、誤差逆伝搬法（BP 法）では、学習率や慣性項係数、中間層ノード数といった基本的なパラメータを試行錯誤的に決定してやらなければならない。本研究においても、ここから始めることにする。

まず、帯状パターン問題の連続値サンプリングを使って実験をおこなった。中間層ノード数を 8 に固定し、学習率 η と慣性項係数 α を可変にとった。η、α ともに 0.1 から 0.2 きざみで 0.9 までとり、それぞれの値において 10 回づつ実験を繰り返した。許容誤差を 0.01 にとり、100000 回で打ち切り（タイムアウト）としたのだが、すべての組合せにおいて収束したため、その学習回数を評価に用いた。その結果を 3.8図に示す。

結果より、$\eta = 0.5$, $\alpha = 0.9$ が望ましい係数であると考えられる。以降、BP 法を使用した帯状パターン問題の連続値に関してはこの値を使用することにする。

同様に、帯状パターン問題の離散値サンプリングを使って実験を行なった。離散値の場

合、連続値と違って●のテリトリーの内部に○が入り込んでいたりするため、学習を収束させることはできなかった。そのため、すべてのηとαの組合せにおいて100000回でタイムアウトし、その平均自乗和誤差により評価することにした。その結果を以下の図に示す。

学習率と慣性項係数が決定したところで、その値を使用して次に中間層ノード数を決定する。中間層ノード数を 2〜10 まで 1 つづつノード数を増加させて実験を行なった。連続値、離散値ともにすべての中間層ノード数で 10 回づつ実験し、その平均と標準偏差を示した図をしめす。

結果より、連続値、離散値ともに中間層ノード数をばらつきが少なくなる 8 とすることに決めた。以降、BP 法で学習を行なう際には、パラメータとして学習率に 0.5 、慣性項係数に 0.9、中間層ノード数に 8 を使用する。

3.4.2 テストパターン

正規分布の帯状パターン離散値問題を用いて汎化能力を調べる。正規分布に基づいて確率的に 0 と 1 をランダムに 300 パターン発生させた。そのうち 100 パターンを学習パターンに使用して BP 法を用いて学習させ、残りの 200 パターンにおいて評価をおこなった。学習率に 0.5、慣性項係数に 0.9 を使用し、100000 回でタイムアウトとした。300 パターンのうち 100 パターンを用いて学習させるため、同様の実験が 3 つできることになる。この 3 種類の実験結果を中間層ノード数を 2〜10 まで変化させて 3.12図に示す。まったく同じ確率の正規分布に基づいてサンプルポイントを定めているが、その値にかなりの差が存在していることがわかる。

3.4.3 クロスバリデーション

テストパターンにより得られた結果をもとに、これら 3 種類の結果を平均したものがクロスバリデーションとなる。3.13図に結果を示す。テストパターンに比べてかなり差が抑えられていることがわかる。3.14 図は 3.13図の最大と最小をとったもので、各中間層ノード数におけるネットワークの値が取り得る範囲を示している。3.14図を見る限り、中間層ノード数が少な過ぎるとその表現方法が不十分となり誤差が大きくなり、逆に大き過ぎても過剰学習により汎化能力が低下していることがわかる。

3.4.4 補間能力

学習により得られたニューラルネットワークにおいて実現される関数が、学習において使用したサンプルポイント以外、つまりポイントとポイントの間の値はどのような値を取り得るのか、という能力を補間能力と呼ぶ。本節では、正規分布の帯状パターン連続値サ

ンプリング問題をニューラルネットワークに学習させ、この補間能力をもって汎化能力を評価した。

まず、帯状分布の連続値において、BP 法を用いて学習させた。学習率に 0.5、慣性項係数に 0.9 を用い、許容誤差を 0.001 としたがすべてにおいて学習は収束せず、100000 回で打ち切りとした。7 × 7 でサンプリングされた値を学習パターンとしてネットワークを学習させ、補間能力を見るために 100 × 100 でサンプリングされた値を評価パターンとして与えた。中間層ノード数を 2〜10 まで変化させ、それぞれの中間層ノード数の平均自乗和誤差を取り、比較してみた。3.15図に結果を示す。各々の中間層ノード数において、初期値をランダムに変えることにより 10 個づつネットワークを発生させ、その平均と標準偏差の値を示している。中間層ノード数が少なすぎると学習が不十分なため、汎化能力はおろか学習パターンに対する誤差まで大きくなる。7 × 7 サンプリングにおいては、中間層ノード数は 6 で最小となる。

次に学習に用いるサンプルポイントにおいて、どの程度のサンプリング数が妥当であるかを調べる。中間層ノード数を 6 に定め、3.15図と同じパラメータで BP により実験を行なった。学習に使用するサンプリング数として、3 × 3、5 × 5、7 × 7、9 × 9、11 × 11、13 × 13、15 × 15 を用いた。評価には先ほどと同様に 100 × 100 のサンプルポイントを使用し、その平均自乗和誤差により評価を行なった。3.16図に結果を示す。学習に使用するサンプルポイント数が少なければ少ないほど、学習そのものは容易となる。そのため、学習による誤差はサンプルポイント数が少なければ少ないほど良い。サンプル数が増加するにつれて、学習が容易にはいかなくなり、誤差が増えているものと思われる。しかしながら、学習に用いるサンプリング数が少なければ、補間能力は低下する。逆に学習に用いるサンプリング数が多ければ、学習そのものが不十分となり、補間能力も低下している。

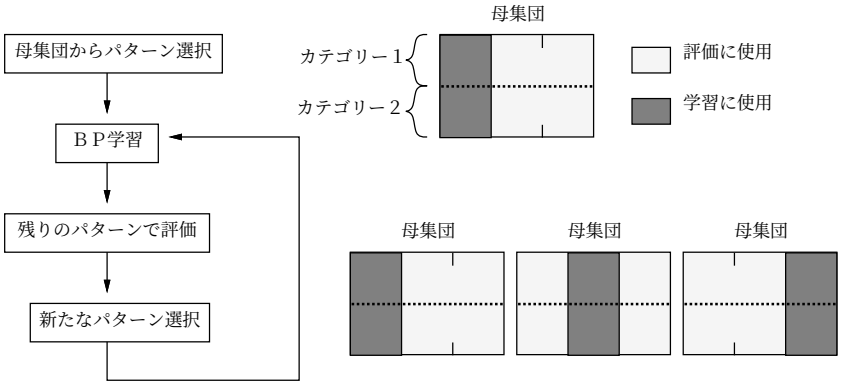

図 3.7: 汎化能力の評価手順

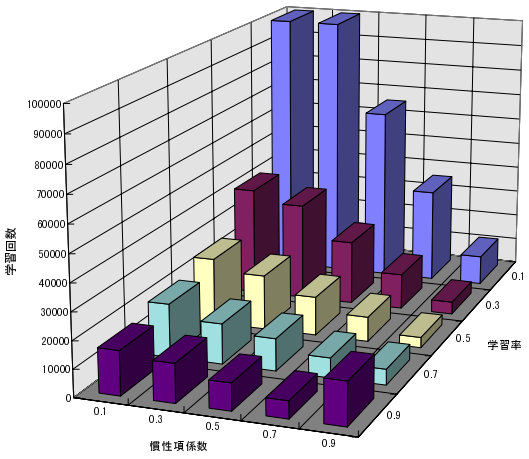

図 3.8: 学習率と慣性項係数に対する平均自乗和誤差の変化（帯状パターン連続）

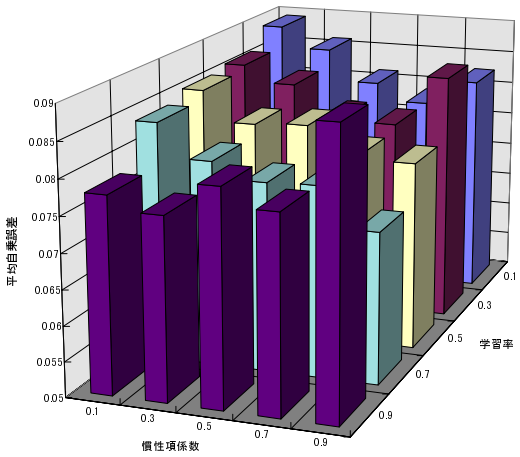

図 3.9: 学習率と慣性項係数に対する平均自乗和誤差の変化（帯状パターン離散）

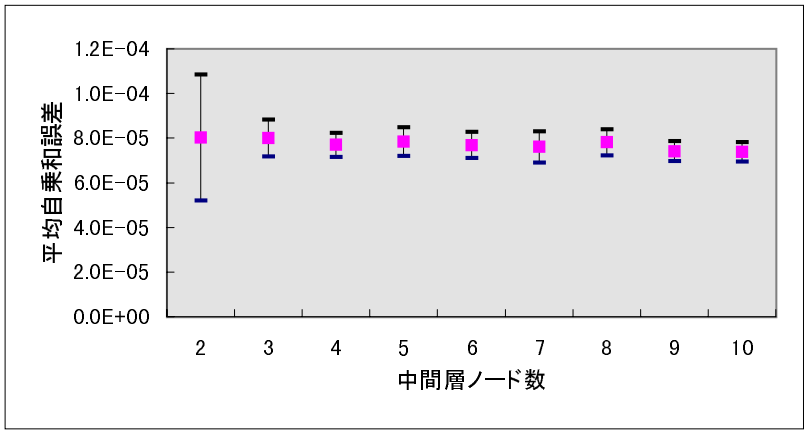

図 3.10: BP で帯状パターン連続を学習した時の平均・標準偏差

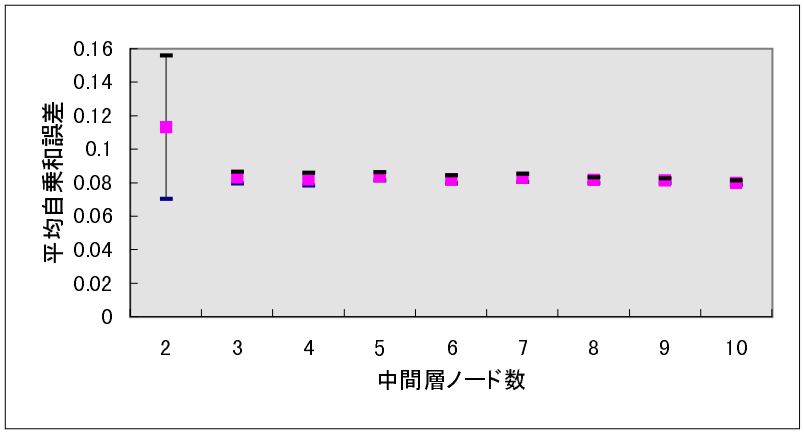

図 3.11: BP で帯状パターン離散を学習した時の平均・標準偏差

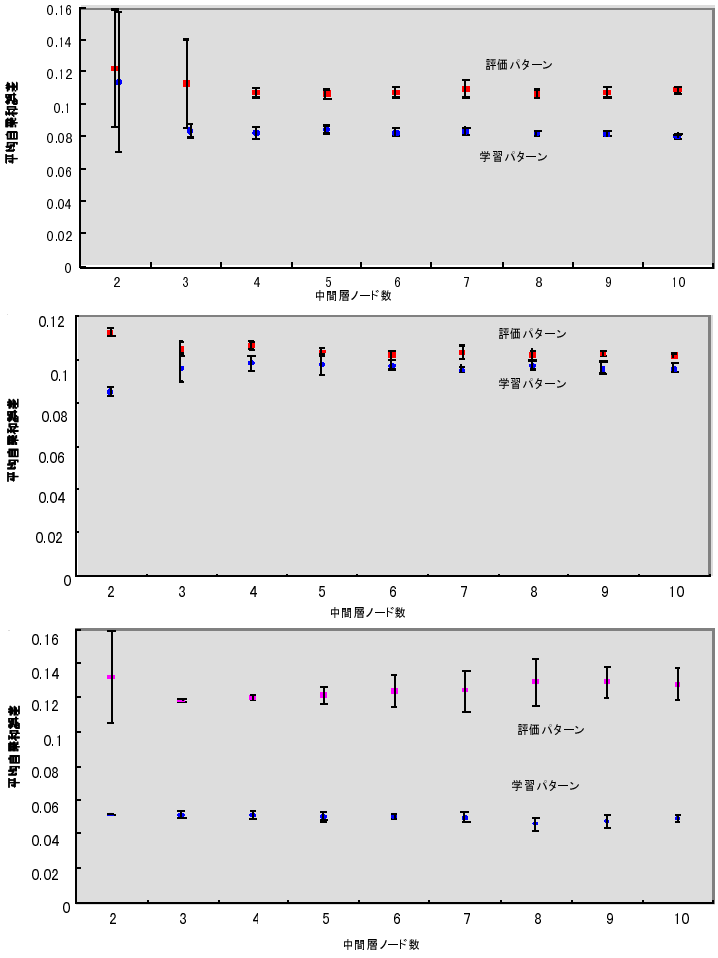

図 3.12: 中間層素子を変化させた時のテストパターンによる汎化能力

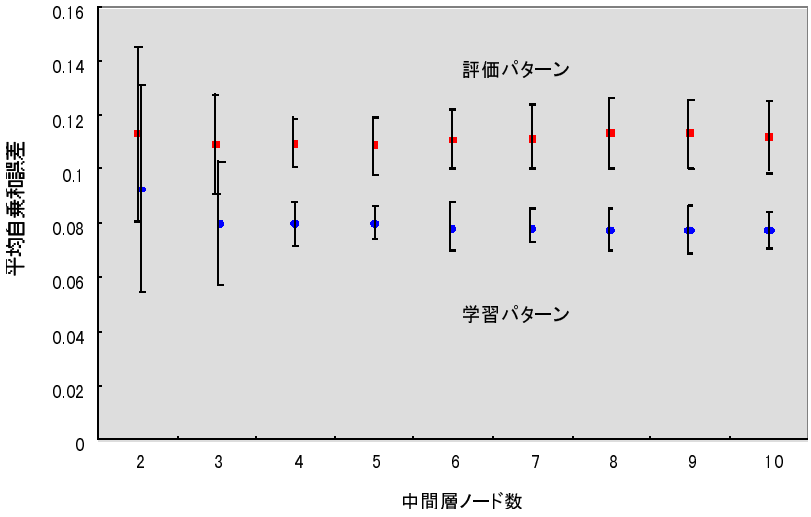

図 3.13: 中間層素子を変化させた時のクロスバリデーションによる汎化能力

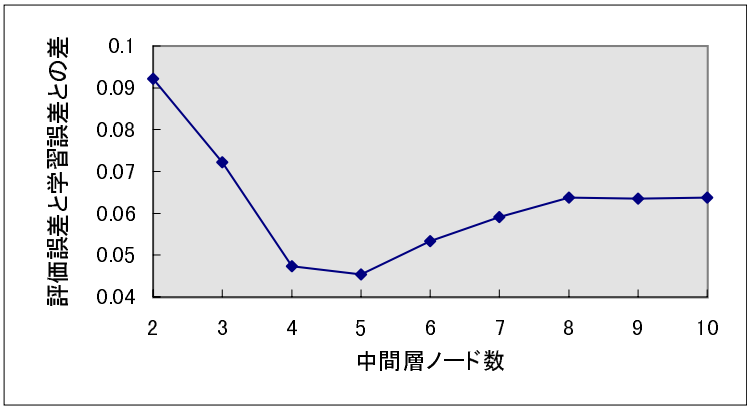

図 3.14: 中間層素子数を変化させた時のクロスバリデーションにおいて取り得る範囲

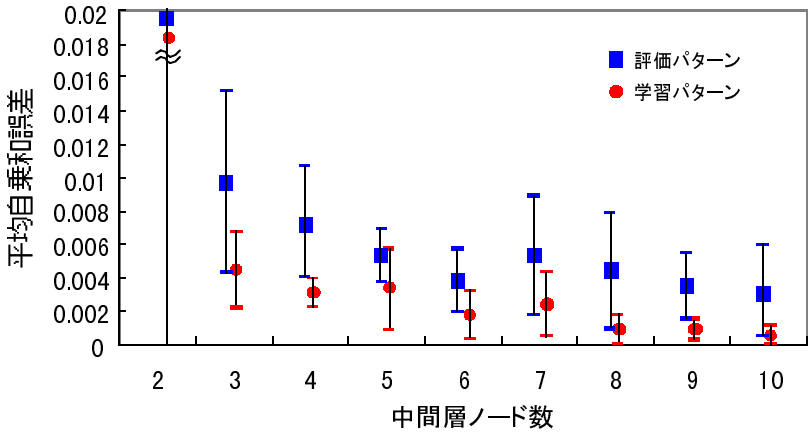

図 3.15: 帯状分布の連続値サンプリング 7 × 7 の補間能力

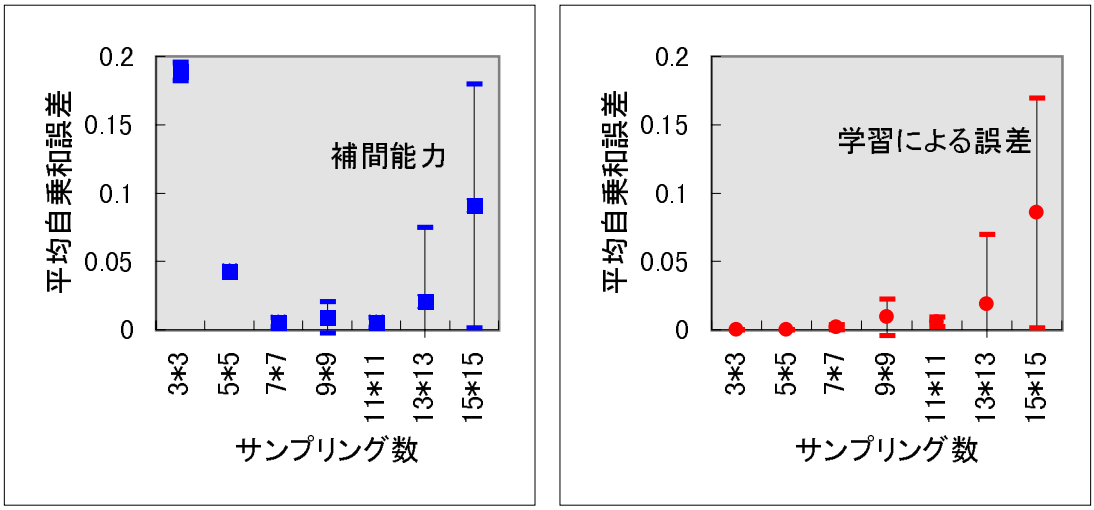

図 3.16: 帯状分布連続のサンプル数に対する補間能力の違い

第 4 章

耐故障性

ニューラルネットワークにおいても古くから、結合荷重にノイズを加えてもネットワークの誤差はあまり大きくならない、といった指摘はなされていた。しかしながら、これは特定の問題における特定の状況での誤差の大きさを議論しているわけであり、耐故障性の議論としては問題やネットワーク構造に依存しない形で行なえる必要がある。本章では、今まで用いられてきた耐故障性の定義では必ずしも適切ではないことを示し、それに変わる定義と評価方法を提案する。

4.1 FTC における耐故障性の定義

FTC(Fault-Tolerant Computing) においては、「ある仕様 Sで記述されるシステムが、ある故障集合 Fに含まれるいかなる故障 $f \in F$が生じた場合においても S を満たし続けるとき、このシステムは Fに対して耐故障性 (Fault-tolerance) を有する」と定義されており、この定義に基づくと、耐故障性があれば故障時にも最初に決めた性能が保証される、という問題に依存しない共通の意味合いを持つことになる。

階層型ニューラルネットワークにおける仕様とは、ネットワークを学習により得る際のプログラム停止条件にあたる。

この定義の場合、仕様とともに故障モデルというものを明確に与える必要が出てくる。

4.2 故障モデル

4.2.1 階層型ＮＮの故障モデル

階層型NNの場合、故障モデルの要素として以下のようなことがらを考える必要がある。

1. 故障の部位　(素子、結合、荷重、など)
2. 故障の種類　(縮退故障、ノイズ、など)
3. 同時に起こりうる数　(単一、多重)

結合は、信号の伝達経路であるため、断線が考えられる。また、故障ではないが、信号に雑音がのることも考えられる。素子の故障には、出力関数の変化や入力信号の欠落といった事が考えられる。荷重においては、重みの変動や荷重自身の欠落が考えられる。欠落とは値が 0 に縮退する故障であり、値の変動とはノイズの混入である。さらに、このような故障が 1 つ発生すれば単一故障であるし、複数で発生すれば多重故障となる。

4.2.2 断線故障

断線故障は、故障モデルで言う**故障の部位**に属し、ニューラルネットワークモデルにおける結合荷重値の 0 縮退故障を意味している。素子故障の多くは複数の結合故障で表現可能である、ネットワークの構成要素には結合が最も多いという理由から、断線故障は最初に取り上げるべきモデルとして適切であると考えられる。

本研究では単一の断線故障と、それが複数発生する多重故障を考えた。多重故障の場合は、断線故障を発生させる本数をリンクの本数の組合せの分だけ存在する。リンクの本数の分だけ断線させればよい単一断線故障と比較して、多重故障の場合の計算量は膨大なものとなる。そのため、本実験においては 3 重の故障までにおいて議論することにする。

4.2.3 必須結合

次に説明するように、単一の断線故障を故障モデルとした場合に便利な概念として、必須結合というものが提案されている。

ある仕様 Sを満たしているネットワーク Nのある結合 lを断線させることによって Nが Sを満たさなくなるとき、lを Nの (Sの下での) 必須結合であるという。単一断線故障

モデルを想定する場合、「ネットワークがフォールトトレラント」であることと「ネットワーク内に必須結合がない」ということは等しい意味を持つ。つまり耐故障ネットワークを得るという目的を必須結合のないネットワークを得るということに言い換えることが出来る。

一つのネットワークに存在する必須結合の数を必須結合数といい、ネットワークの耐故障性を評価する 1 つの尺度を与えられるものと考えられる。

4.3 耐故障化学習

以上のように耐故障性の尺度が与えられたことで、これを学習の目的関数として耐故障性を学習により獲得するニューラルネットワークモデルが実現できる。現在、GA(Genetic Algolithm) を用いた学習法や、勾配法による FTBP(Fault-Tolerant Back Propagation) 学習法 [1] などが提案されている。ここでは、本研究で使用する FTBP 法を示す。

BP 法の場合では、最適化すべき誤差評価関数は

$$E_{BP} = \sum_{p}\sum_{i}(t-a)^2 \tag{4.1}$$

という関数であった。FTBP 法の場合、学習規則において故障 (fault) を発生させているため、BP 法の評価関数に fault を考慮した誤差評価関数を付加すれば、BP 法における評価関数をうまく利用できる。式で表現すると、

$$E = \sum_{f}\sum_{p}\sum_{i}(t-a_f)^2 \tag{4.2}$$

と表せる。ただし、a_fは故障が発生したニューラルネットワークにおける実際の出力を表している。上式を書き直すと、

$$\begin{aligned} E &= \sum_{f=0}^{F}\sum_{p}\sum_{i}(t-a_f)^2 && (4.3)\\ &= \sum_{f=0}^{F}\sum_{p}\sum_{i}(t-a_f)^2 - \sum_{f=0}^{F}\sum_{p}\sum_{i}(t-a_0)^2 + \sum_{f=0}^{F}\sum_{p}\sum_{i}(t-a_0)^2 && (4.4)\end{aligned}$$

$$= \sum_{f=0}^{F} \sum_{p} \sum_{i} \{(t-a_f)^2 - (t-a_0)^2\} + (F+1) \sum_{p} \sum_{i} (t-a_0)^2 \tag{4.5}$$

これは、(4.2) 式がフォールトトレランスと学習パターンの両方についての評価関数となっていることを示している [1]。ただし、$f = 0$ は故障がないことを表している。

ここで、故障が発生した場合のネットワークの誤差の方が故障の発生しなかったネットワークの誤差よりも大きいと仮定する。式で表すと、

$$(t-a_f)^2 > (t-a_0)^2 \tag{4.6}$$

となり、通常、学習パターンの獲得が実現されていれば成り立つことから、学習のごく初期を除いては妥当なものであると考えられる。よって、(4.5) 式の第 1 項は

$$\sum_{f=0}^{F} \sum_{p} \sum_{i} |(t-a_f)^2 - (t-a_0)^2| \tag{4.7}$$

と書き換えることができる。よって

$$E \approx \sum_{f=0}^{F} \sum_{p} \sum_{i} |(t-a_f)^2 - (t-a_0)^2| + (F+1)(t-a_0)^2 \tag{4.8}$$

となる。第 1 項は故障の有無による違いを表す評価関数であり、第 2 項は通常の BP 法の学習規則で用いられている学習パターンの獲得についての評価関数である。つまり (4.5) 式はこの 2 つの評価関数が線形結合されたようなものと考えることができる [1]。

これで (4.5) 式を誤差評価関数として学習規則を導けば、目的とするネットワークを学習によって得られるような階層型ニューラルネットワークが実現可能であることが提示できた。具体的に学習規則を式で表すと、

$$E = \sum_{f} E_{BP} \tag{4.9}$$

より、改めて導出するまでもなく、

$$\Delta w^{FTBP} = \sum_{f} \Delta w_f^{BP} \tag{4.10}$$

で与えられる。故障の集合の中から順に選んだ故障を生じさせながら、通常の BP 学習規則に従った学習を行なえば良いわけである。

4.4 定義および尺度に関する問題点

前節までに示した議論の流れにより、故障時にも最初に指定した性能を保つ耐故障ニューラルネットワークというものを学習により構成することが可能となった。しかしながら、耐故障性の評価という観点に立ち返って見るとニューラルネットワークでは、FTC で用いられる耐故障性の定義が必ずしも適切ではないことを以下に示す。

ニューラルネットワークは論理回路と異なり連続値をとるため、仕様を満たさなくなったときでも、完全に異なった値になるわけではなく、もとの値からの逸脱の度合というものが存在する。また、故障によりネットワークが実現する関数の細部が失われても、大まかな関数形は保存される可能性があり、故障の度合に応じて誤りが増加する graceful degradation の能力を有するものと期待されるにもかかわらず、これが FTC で用いられる耐故障性の定義に反映されていない。さらにＮＮでは、汎化に代表されるように、仕様として与えられていない能力に対しても期待される部分が存在する。例えば、学習時には最大誤差を見て終了判定したネットワークに、平均的な誤差も小さくなっていることを期待する、というものがある。

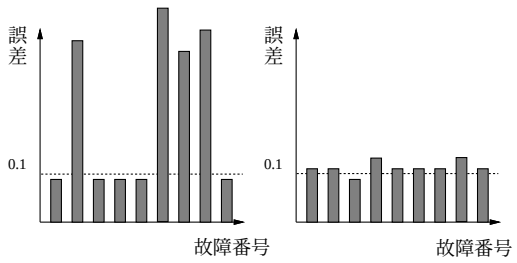

図 4.1: 必須結合数と平均誤差

これは故障時にも同様であり、この図にあるように、最大誤差 0.1 という仕様のネットワークの場合、点線を超えると必須結合とみなされる。図で、左側のネットワークに比べて右側のものの方がはるかに必須結合が多いのにもかかわらず、平均的な誤差という意味では右図の方が望ましいと考えることができる。

このような状況が実際に起こりうることは簡単なシミュレーションを行うことにより示すことができる。

表 1:シミュレーション結果

No.	必須結合数	故障時 MSE	無故障時 MSE
1	16	0.043	0.000079
2	19	0.014	0.000077
3	15	0.046	0.000087
4	15	0.039	0.000093
5	15	0.018	0.000077
6	13	0.027	0.000089
7	17	0.029	0.000087
8	16	0.013	0.000085

この表は、中間層ノード 8 個の BP モデルで XOR を学習したものである。初期値をランダムに変えることにより 8 回学習を行ない、最大誤差が 0.1 未満となったところで終了した。

No.1 と No.2 では無故障時の誤差和はほぼ等しいが故障時には No.2 がより多くの必須結合を有しながら No.1 より少ない誤差和を有している。No.6 は最も必須結合が少ないにもかかわらず、誤差和は最も必須結合が多い No.2 よりも大きい。

このように、誤差和というものに注目した場合、必須結合数は一般に期待されるような耐故障性の評価を与えていないということがわかる。

4.5 耐故障性の新たな定義

4.5.1 新たな仕様と定義

上記の問題への対処方法として以下の 2 つが考えられる。

1. 仕様にすべてを記述

耐故障性の定義はそのままで、仕様において必要な用件をすべて記述する。上記の例であれば、ネットワークの仕様の中に最大誤差のみならず誤差和も含める。この方法は、FTC の議論との整合性は高いものの、すべての要求が記述可能であるかという点に問題

がある。また、記述は可能であったとしても評価にコストがかかりすぎる場合などでは、現実的な評価方法、学習方法が構築できない可能性がある。さらに記述されなかった性質については何も保証されないことは従来と同じである。

2. 新たな定義を導入する

故障時と無故障時で関数自体があまり変わらないことを耐故障性の定義とする。ネットワーク自身が実現する関数を対象とし、最初に学習したときの仕様には直接言及しない。関数自体の比較には、汎化能力において取り上げられている議論を流用する。この定義の場合、故障時にはNNが一部仕様を満たさなくなる可能性もあるが、仕様として与えられなかった部分も含め、無故障時の性質を故障時にも多く引き継ぐことが期待される。

本論文では、後者の立場から、次節に述べるような評価方法を提案する。

4.5.2 関数比較

本節では必須結合数による評価に代わる関数比較という新しい評価方法を提案する。これは、故障が発生したネットワークの出力値から故障無しのネットワークの出力値の差のノルムで表される。学習により得られたネットワークの関数を f、そのネットワークに故障が発生した場合の関数を f_f と置いて式で表すと、

$$||f - f_f|| = \int\int (f - f_f)^2 \tag{4.11}$$

つまり、故障ネットワークと故障無しネットワークの差の 2 乗の積分である。その一例として、XOR を学習させたNNの関数比較を 4.2図に示す。

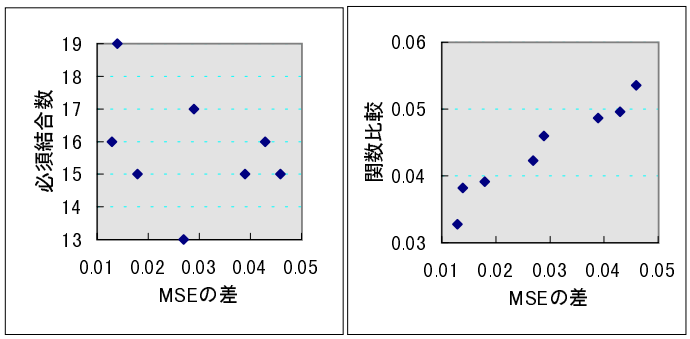

図 4.2: 必須結合数と関数比較

そこでXORの学習を完了したネットワークに対して、縦軸、横軸ともに0.1きざみでメッシュを切り、それぞれ値を入力していった。その際、(0,0),(0,1),(1,0),(1,1)の4点のまわりを注意深く観察したいという考えから、サンプリングの範囲を縦軸、横軸ともに−1.0から2.0まで拡張して評価を行なった。0.1きざみのサンプリングで連続関数をうまくマスクできているかどうか疑問であったため、その細かさを半分の0.5、さらに半分の0.25、さらに半分の0.125という単位でメッシュを切って実験したが、いずれの値においても0.1きざみメッシュと変わらない値を示したため、計算器の時間上最もサンプル数の少ない0.1きざみメッシュを採用して実験をおこなった。

4.2図からもわかるように、必須結合数と平均自乗和誤差の間には関連性はなさそうだが、この関数比較と平均自乗和誤差との間には正の相関があることを示している。

この関数比較という評価方法を用いると、4.5.1節で提案した「新たな定義」を導入することができる。今までの、必須結合数による耐故障性の評価では、単一断線故障に対する必須結合の数で評価していたため、耐故障化学習が必ず終了しなければならなかった。しかし、関数比較という評価方法を用いると、学習は必ずしも終了する必要はない。さらに、必須結合を探すために断線故障は単一に限定されていたものが、関数比較においては単一である必要はなく、多重故障にも適応できることを意味している。

4.6 シミュレーション

4.6.1 FTBPモデルの基本パラメータ

FTBP法の場合、冗長性をもたせるために中間層ノード数をBP法の場合よりも多くもたなければならない。そこでまず、学習率ηと慣性項係数αを適当に決めておき、中間層ノード数を4〜20まで2つ刻みで変化させて、平均自乗和誤差を評価関数として最適な中間層ノード数を選ぶことにした。帯状分布の連続値、離散値の結果を4.3図、4.4図に示す。結果より、連続値、離散値ともばらつきがおさまり、平均自乗和誤差の減少率から考えて、中間層ノード数は16が妥当であると考える。

次に最適な学習率と慣性項係数を決定する。中間層ノード数を16として学習率と慣性項係数を変化させた。結果を4.5図、4.6図に示す。結果より、帯状パターン離散では学習率0.3、慣性項係数0.9、帯状パターン連続では学習率0.5、慣性項係数0.5に決定する。

4.6.2 関数比較

前節と同様に多重故障についても実験を行なった。帯状パターン離散問題を FTBP 法で学習することにより得られたネットワークを使用して、中間層ノード数 16 のネットワークに対する実験を行なった。初期値をランダムに変えることにより 10 個のネットワークが存在している。まずこの 10 個のネットワークに対して基本評価をおこなった。

	a	b	c	d	e	f	g	h	i	j
無故障	7.745	7.720	7.819	7.719	7.685	7.716	7.721	7.745	7.806	7.763
単一断線故障	9.177	9.062	9.073	9.054	8.976	9.016	9.031	8.998	9.071	9.032
関数比較	1.376	0.766	1.123	0.464	1.060	0.508	1.153	0.656	0.504	1.185
必須結合数	2	0	0	0	0	0	0	2	0	0

※ただし、値は 10^{-2}

4.7図に帯状分布離散を FTBP 法で学習したネットワークの結果を示した。無故障時の平均自乗和誤差と単一断線故障を発生させた時の平均自乗和誤差との差を MSE の差とした。この MSE の差をもとにすべての値を昇順にソートしてみた。関数比較の値と比べてみると、やはり正の相関があることがわかる。

ここでこれら 10 個のネットワークに対して多重故障を発生させてみた。結果を以下に示す。

故障数	a	b	c	d	e	f	g	h	i	j
0	0.7745	0.7720	0.7819	0.7719	0.7685	0.7716	0.7721	0.7745	0.7806	0.7763
1	0.9177	0.9062	0.9073	0.9054	0.8976	0.9016	0.9031	0.8998	0.9071	0.9032
2	1.0553	1.0303	1.0264	1.0291	1.0176	1.0226	1.0274	1.0175	1.0265	1.0236
3	1.1871	1.1453	1.1397	1.1442	1.1306	1.1357	1.1454	1.1283	1.1394	1.1380

※ただし、値は 10^{-1}

4.8図は各ネットワークに多重の断線故障が発生したときの平均自乗和誤差の値を示している。故障の数が増加すると誤差の値も増加することは明白である。

次に、故障数が 0〜3 までの断線故障において、各々の故障数に対する 10 個のネットワークの値を 4.9図に示す。4.9図の値は、各断線故障時における最小の誤差値を出力するネットワークとそれ以外のネットワークとの差の値を表している。つまり、各故障数において最も低い誤差を出力するネットワークの値が 0 になることになる。

4.9図では、誤差 0.001 あたりで混線しているため、まったく同じ図を平均自乗和誤差 0〜0.002 の間にしぼって示してみた。

4.10図は、断線故障が増加する際のネットワークの故障の仕方の特性を表している。4.10図より以下の 4 つのタイプのネットワークに興味を持った。

- すべての故障において良いネットワーク : e
- すべての故障において悪いネットワーク : a
- 故障の増加とともに最良NNとの差が悪化するネットワーク : b
- 故障の増加とともに最良NNとの差が良くなるネットワーク : h

発生する故障の数が増加するということは、故障していない部分のみでネットワークを構成しなければならないため、誤差も増加してしかるべきである。しかし、同じ故障条件のもとでネットワークの出力に差が生まれる。この原因をさぐることは重要であると思われるが、多重断線故障の場合、発生させる故障数のすべての組み合せにおいて検討しなければならない。

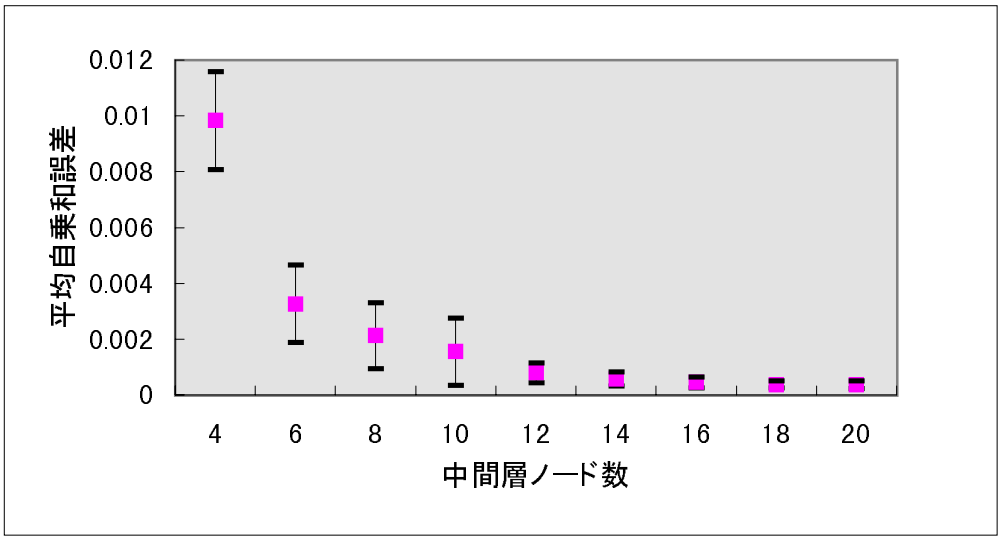

図 4.3: 中間層素子数を変化させて FTBP で帯状パターン連続を学習した時の平均・標準偏差

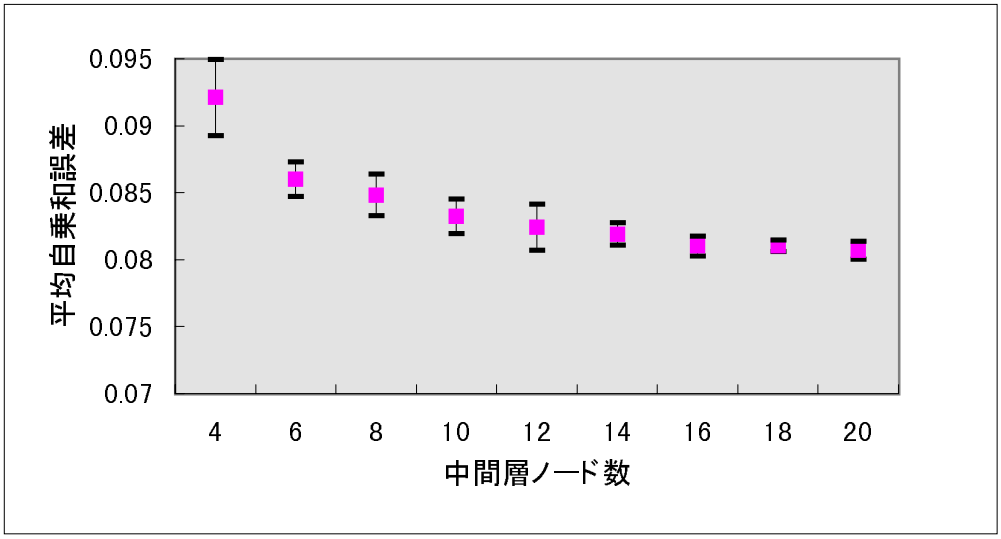

図 4.4: 中間層素子数を変化させて FTBP で帯状パターン離散を学習した時の平均・標準偏差

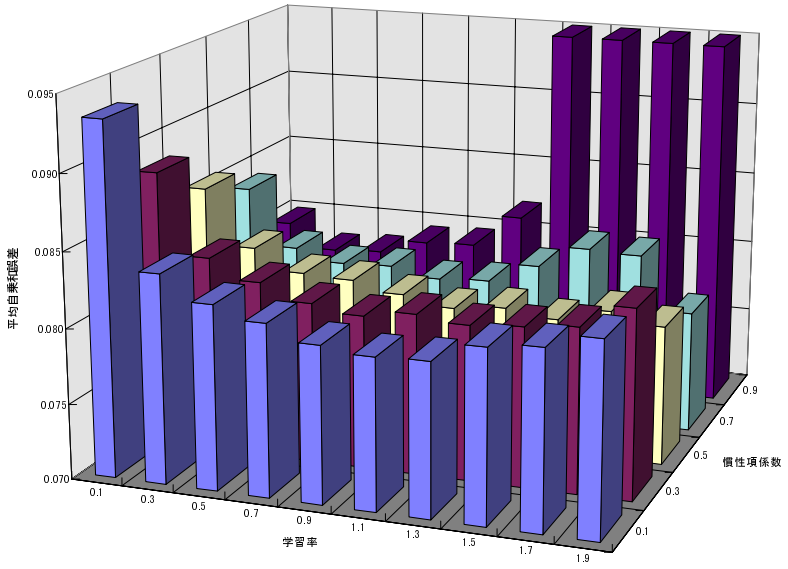

図 4.5: FTBP で帯状パターン離散を学習した時の学習率と慣性項係数

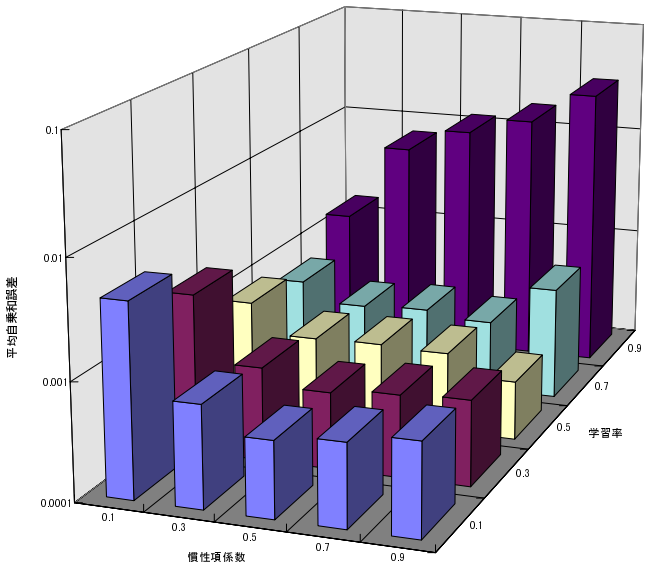

図 4.6: FTBP で帯状パターン連続を学習した時の学習率と慣性項係数

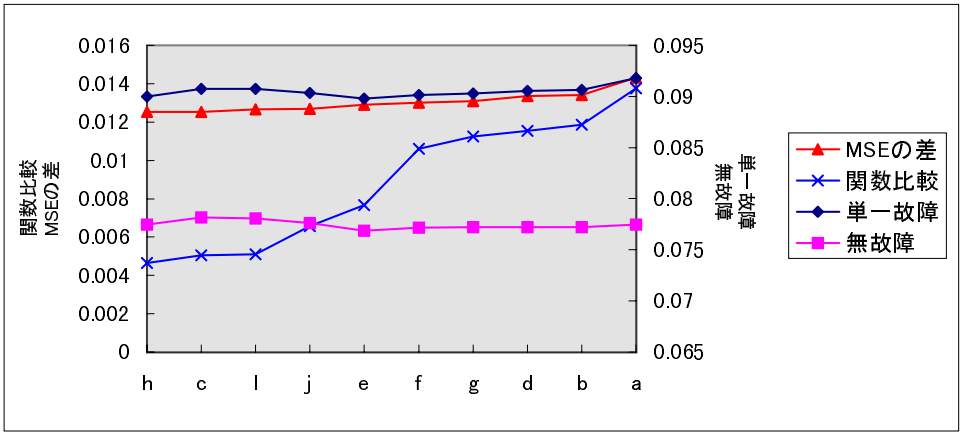

図 4.7: 中間層ノード数 16 の性能評価

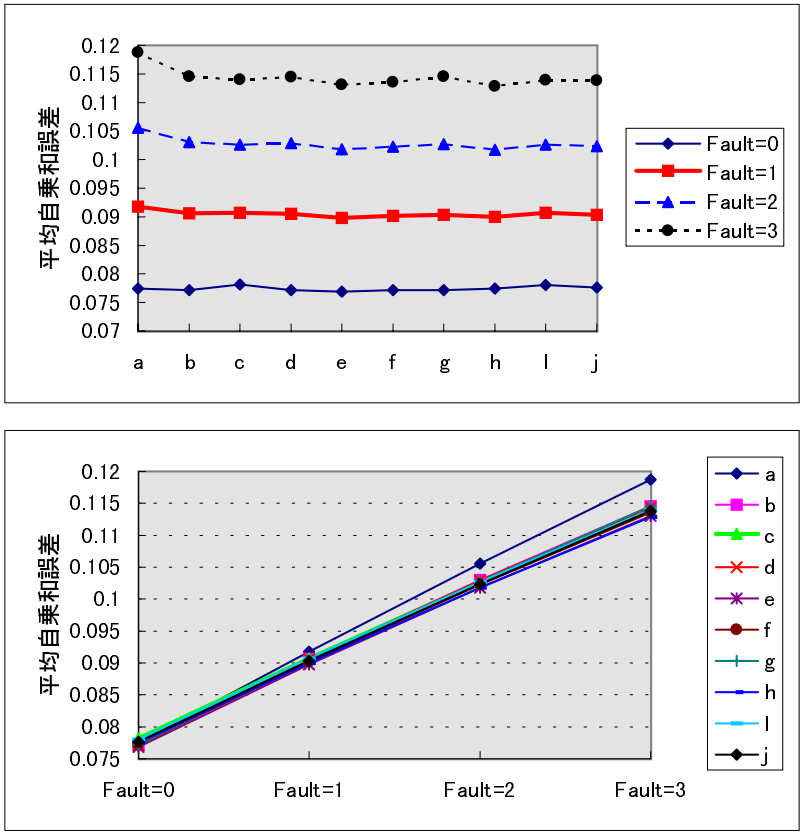

図 4.8: 多重故障

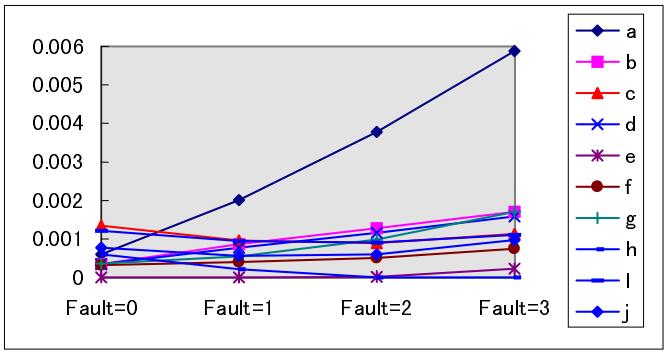

図 4.9: 多重故障（最良NNとの差）

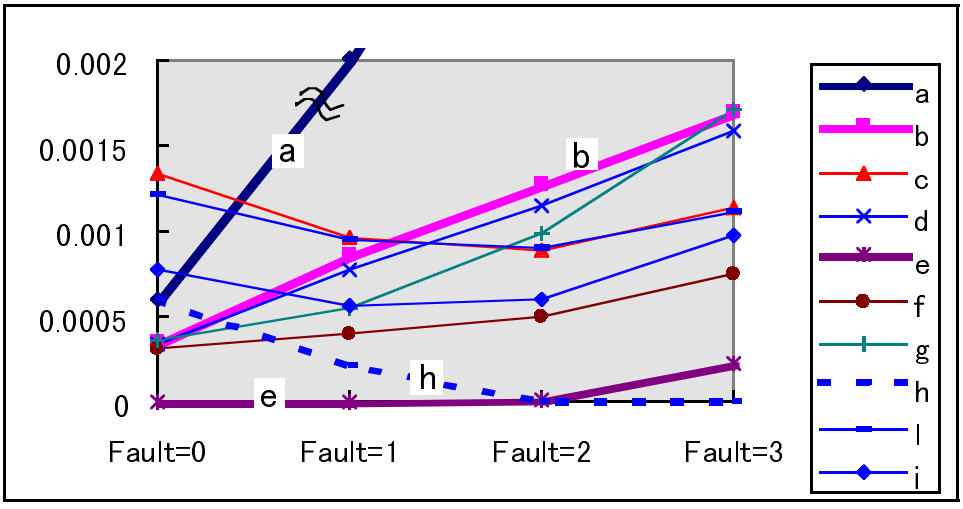

図 4.10: 多重故障（最良NNとの差）

第 5 章

耐故障性と汎化能力

本章では、第 3 章と第 4 章で述べた耐故障性と汎化能力の関係について検討する。まず、BP 法と FTBP 法によって得られるネットワークの耐故障性と汎化能力を調べ、従来の論文で主張されていたこれらの間の関係について検討する。次に、ネットワークの特徴量として、中間層素子の微係数と関数のなめらかさの評価を行い、これらと耐故障性、汎化能力との関係を議論する。

5.1 耐故障性の評価

BP 法と FTBP 法により得られたネットワークにおいて、中間層ノード数を 4〜20 まで変化させ、関数比較により耐故障性の評価を行なった。学習させる問題には正規分布の帯状パターン連続と離散を使用した。BP 法は第 3 章で求めた学習率 0.5、慣性項係数 0.9 を使用し、FTBP 法では第 4 章で求めた学習率 0.3、慣性項係数 0.9（帯状パターン離散)、学習率 0.5、慣性項係数 0.5（帯状パターン連続）を使用した。結果を 5.1図と 5.2図に示す。それぞれ初期値をランダムに変えることにより 10 個のネットワークを構築し、10 個の平均と標準偏差の値を示している。

耐故障性には、ある程度の冗長性を必要とするため、中間層ノード数がある程度必要である。やはり、中間層ノード数が十分ではないところでは、関数比較の値も悪くなっているが、12 個以上の中間層素子を持つあたりから値は安定しはじめている。正規分布の帯状パターン連続、離散、両方の問題において、耐故障化学習を行なっている FTBP 法の方が、すべての中間層ノード数において BP 法より故障に強いことを示している。

実際に単一の断線故障を BP 法と FTBP 法で発生させ、その自乗和誤差の値を調べてみた。同様のパラメータにおいて、正規分布の帯状パターン連続と離散について実験を行ない、5.3図と 5.4図に結果を示す。やはり BP 法の方が故障に弱いことがわかる。

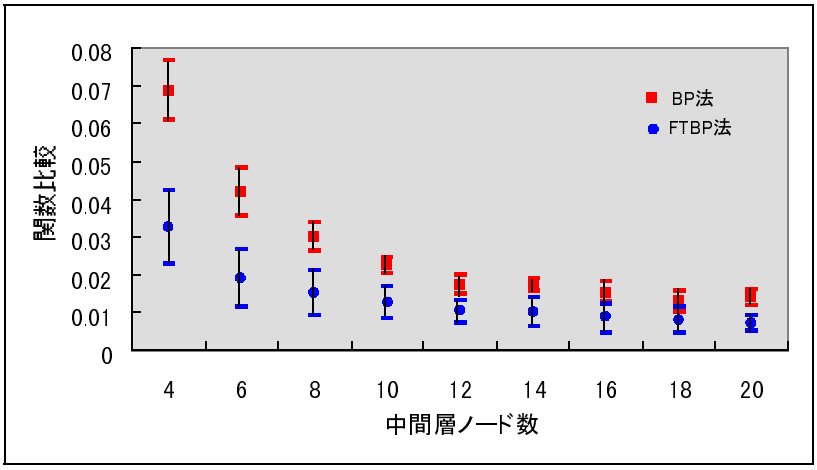

図 5.1: 中間層ノード数を変化させた時の BP と FTBP の関数比較（帯状パターン連続）

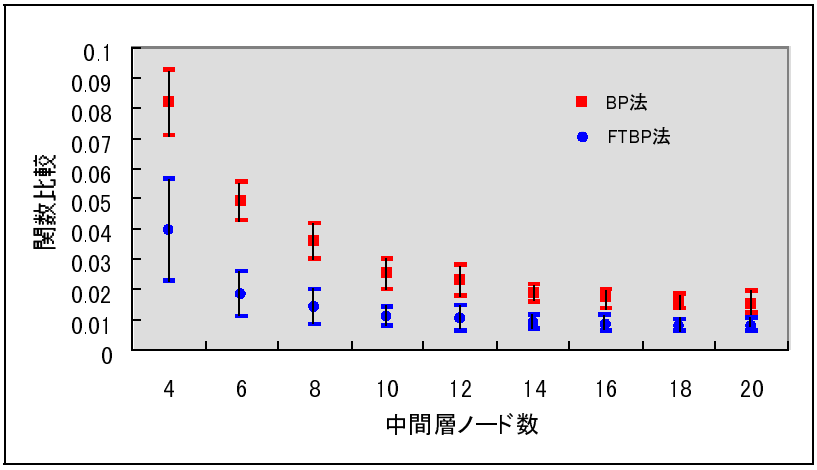

図 5.2: 中間層ノード数を変化させた時の BP と FTBP の関数比較（帯状パターン離散）

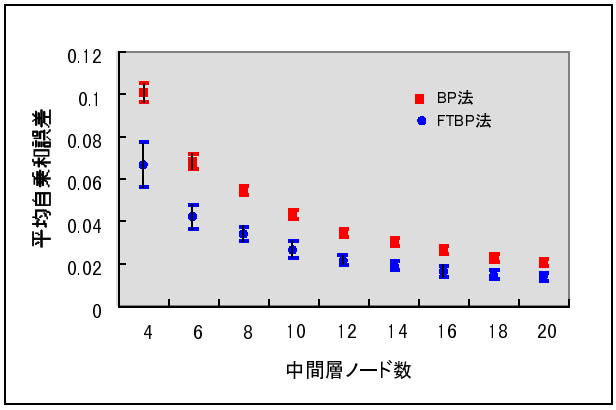

図 5.3: 単一断線故障を発生させた時の BP と FTBP の自乗和誤差（帯状パターン連続）

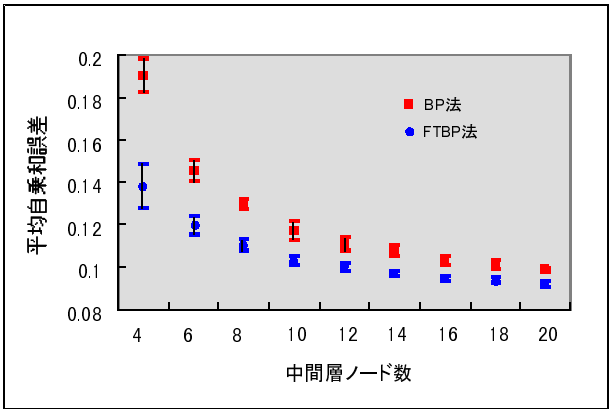

図 5.4: 単一断線故障を発生させた時の BP と FTBP の自乗和誤差（帯状パターン離散）

5.2 汎化能力の評価

続いて BP 法と FTBP 法において帯状パターンの離散値を学習したネットワークに対する汎化能力の評価を行なった。基本的なパラメータは上記と同じとし、中間層ノード数を 4 から 20 まで変化させて実験を行なった。結果を 5.5図に示す。すべての中間層ノード数の場合において、FTBP 法の方が誤差が少ないことから汎化能力が高いことを表している。これは、耐故障性に優れたネットワークは汎化能力にも優れるという従来の主張

と一致する。

しかしながら、次に同様のネットワークにおける補間能力を、帯状パターンの連続値を用いて調べてみたところ、今度は逆に FTBP 法よりも BP 法の方が優れた結果を示すことがわかった。結果を図に示す。この結果については次節で考察する。

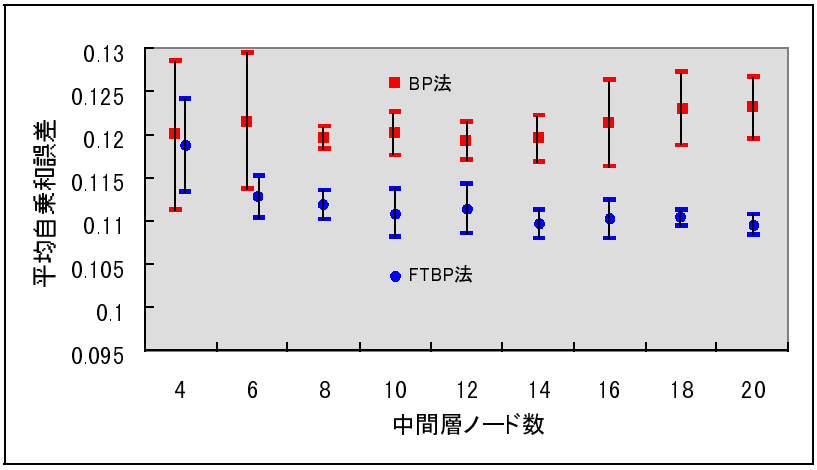

図 5.5: BP 法と FTBP 法の汎化能力の比較（帯状パターン離散）

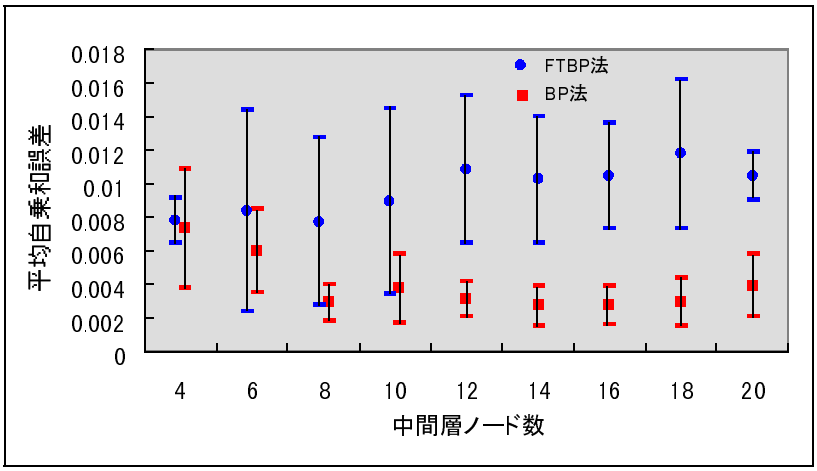

図 5.6: BP 法と FTBP 法の補間能力の比較（帯状パターン連続）

5.3　耐故障性と汎化能力の評価

5.1 節において、2 つの問題何れにおいても FTBP 法の方が故障に対して強いことは明らかとなったが、5.2 節では、帯状分布の離散値問題においては従来言われていたように FTBP 法の方が汎化能力が高いものの、帯状分布の連続値問題については、BP 法の方が良い結果を出すことがわかった。

そこで、本節では、まず同じ FTBP 法で学習された同一中間層ノード数を持つ個々のネットワークについて汎化能力と耐故障性の関係を調べ、これらのネットワークにおいてこの二つの性質の関係はどのようになるかを調べる。用いたのは帯状分布の離散値問題を学習することにより得られた中間層ノード数 16 のネットワークである。初期値をランダムにかえることにより 10 個のネットワークが学習により得られた。横軸に評価パターンに対する誤差、つまり汎化能力を表し、縦軸に断線故障が発生したときの出力誤差を表した図を示す。

	a	b	c	d	e	f	g	h	i	j
故障	0.9177	0.9062	0.9073	0.9054	0.8976	0.9016	0.9031	0.8998	0.9071	0.9032
汎化	1.2006	1.1941	1.1717	1.1917	1.1751	1.1760	1.1844	1.1704	1.1718	1.1741

※ただし、値は 10^{-1}

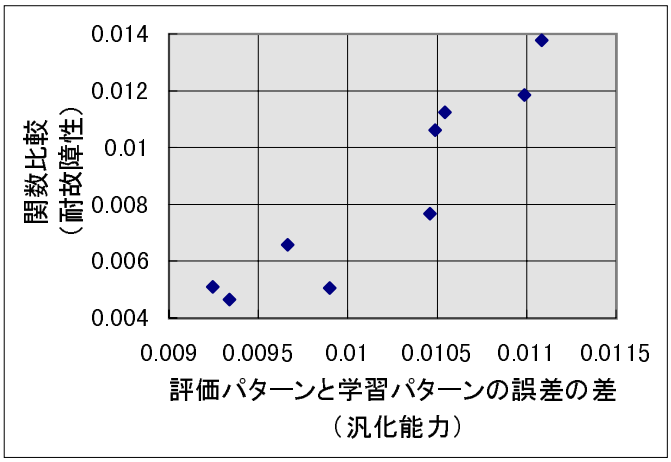

図 5.7: 同じ中間層を持つネットワークの耐故障性と汎化能力

5.7図から、あきらかに耐故障性と汎化能力との間には正の相関があることがわかる。これにより、少なくとも正規分布の帯状パターン離散値問題においては、耐故障性に優れるネットワークは汎化能力においても優れると考えることができる。

本節では更に、ネットワークの特徴量として、中間層素子の微係数とネットワークが実現する関数のなめらかさを測定し、耐故障性と汎化能力の両方に共通する要因を探ることにする。

3 章で論じたように、中間層の微係数は、入力の変動に対する中間層素子出力の安定性をみる特徴量である。この微係数の値が小さいほど安定性があるといえるため、耐ノイズという意味での汎化においては、小さいほうが望ましいといえる。また、入力層から中間層への結合の故障は入力層の変動と類似した効果を持つため、耐故障性に関しても、小さいほうが有利であると考えられる。

帯状分布の離散で BP と FTBP を比較した結果を 5.11図に示す。中間層ノード数が増加するにつれて微係数の値も増加しているものの、FTBP の方が優れた値を示していることがわかる。

一方、5.9 図に示す帯状分布連続においては BP の方が望ましい値を得ている。

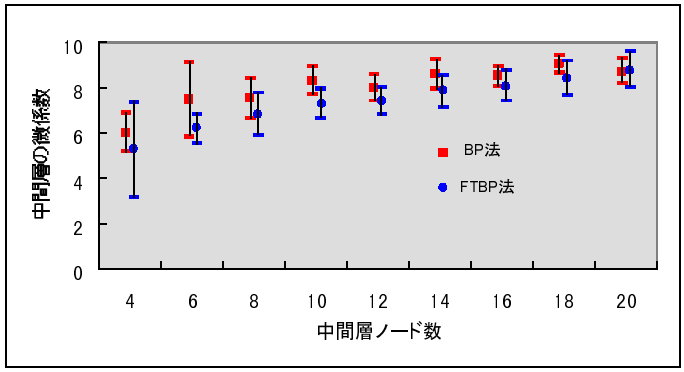

図 5.8: 帯状分布離散を学習した BP と FTBP の中間層素子の微係数

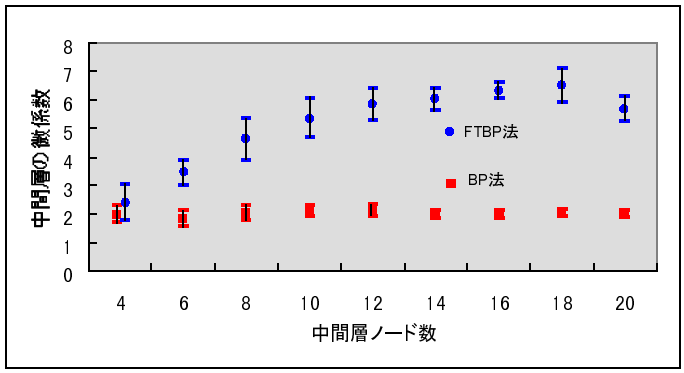

図 5.9: 帯状分布連続を学習した BP と FTBP の中間層素子の微係数

次に、関数の滑らかさに関する結果を示す。

この値が小さいほどネットワークが実現している関数は滑らかとなる。つまり入力に対する出力の変化は小さいわけで、中間層素子の微係数値同様、小さい方が望ましい。

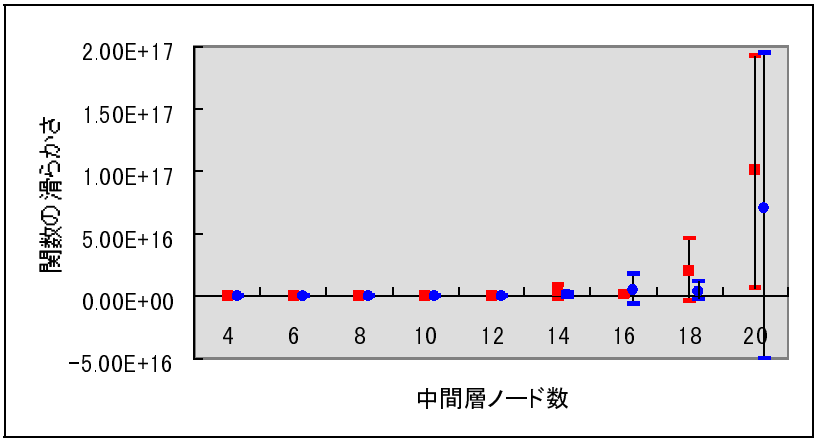

図 5.10: 帯状分布離散を学習した BP と FTBP の関数の滑らかさ

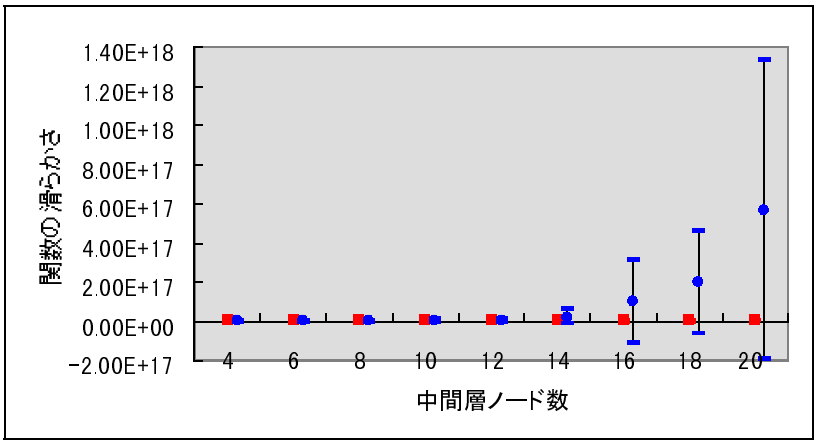

図 5.11: 帯状分布連続を学習した BP と FTBP の関数の滑らかさ

この結果からは、汎化能力や中間層素子の微係数のときと同様、離散の場合にはFTBPの方が望ましい結果を得ているが、連続の場合にはむしろ BP の方が優れた結果を出しているといえる。

5.4 考察

正規分布の帯状パターン離散値問題においては、BP 法よりも FTBP 法の方が耐故障性はもとより汎化能力にも優れることがわかった。さらに同じ中間層素子数をもつ個々のネットワークにおける耐故障性と汎化能力の関係についての結果から、故障時の誤差が少ないネットワークの方が未学習パターンに対する誤差も少ないことがわかった。

これらの結果は耐故障性に優れたネットワークは汎化能力においても優れているという従来の報告に沿ったものとなっている。

しかしながら、連続値の問題においては、逆の結果が得られることになった。FTBP 法で得られたネットワークの方が耐故障性には優れるものの、汎化能力においては通常の BP 法のものに劣り、かつ、特徴量として取り上げた値についても、より望ましいと考えられる値をとったのは BP の方であった。

以上の結果から、まず、ここで取り上げた二つの特徴量は、汎化能力においては予想されたような性質を示す量となる可能性はあるが、耐故障性に対しては必ずしも有効ではないことが明らかとなった。これで、これらの特徴量を仲立ちとした汎化能力と耐故障性との関連性は議論できないことがわかる。

本論文では詳細を取り上げなかった (採取したデータは付録として巻末に収録する) が、ここで取り上げたもの以外の特徴量においては、ほとんど耐故障性や汎化能力との関連性を見い出すことができなかった。これは、こうしたプリミティブな特徴量では汎化や耐故障性といった高度な性質を説明づけるには不十分であったものと考えられる。

関数のなめらかさはネットワークの実現している関数そのものの性質であり、中間層素子出力の微係数はこれに準ずるものであるため、汎化能力や耐故障性と関わりが比較的深かったと考えられるものの、十分な手がかりを与えるものとはならなかった。

今後、同様のアプローチでより高度な特徴量を探していく方向も考えられるが、今回のように、文献調査をもとにした偶発的なやりかたではなく、何らかの裏付けを持って特徴量を探す必要があると思われる。

次に、本章で取り上げた二つの問題で極めて大きな違いを生じていることが明らかとなった。これは、これらの問題の性質が大きく異なっているものと考えられる。

帯状パターン離散値は、0 と 1 を分離する問題であり、パタン認識などと同様の性質を持っているものと考えられる。汎化能力を調べる際に使われるベンチマークには何らかのパタン認識が使われることが多いことを考えると、従来の耐故障性と汎化能力との関係

を述べた研究において、扱われていた問題が全てこの類のものであった可能性は否定できない。

丹 [1] らによると、こうした離散値を出力としてとる問題の場合、FTBP は中間層から出力層への結合荷重を大きくとり、出力層素子をシグモイド関数の不活性 (飽和) 領域で使うようなネットワークを形成する傾向にある。シグモイド関数の不活性領域では入力の変動が出力に影響を与えにくくなるため、こうした結合荷重をとることで、故障の際の値の変動を抑えることが可能となる。

故障による変動を抑えることは、入力の変動を抑えることにもつながり、耐ノイズの意味での汎化能力には有効に働くものと考えられる。

一方、帯状パターン連続値問題のような、出力が連続値をとる問題は、連続値の関数の形を素子の出力関数であるシグモイド関数の重ね合せで表現しようとする問題であると考えられる。

この類の問題では、大まかな形はシグモイドの組み合わせで容易に形成できても、要求されている連続値をネットワーク出力として正確にとるためには、シグモイド関数の活性領域を引き伸ばした緩やかなカーブを重ね合せて調整する必要が出てくるものと考えられる。

このような要求は上述のような FTBP の性質とは相容れないものであると考えられ、こうした連続値を要求する問題については、FTBP はうまく対処ができていないのではないかと思われる。

詳細については、連続値問題におけるネットワークの内部表現の解析や、BP ネットワークにおける連続値問題における耐故障性と汎化能力の関係の評価などが必要とされる。

第 6 章

結論

6.1 本研究の成果

本研究では、まず、汎化能力の根本要因について考察し、文献調査を元に、汎化能力と関連の深いと思われるネットワーク特徴量、および今までに提案されてきた汎化能力を向上させる手法についてまとめた。

次に、従来使用されてきた必須結合数による耐故障性の評価方法は、必ずしも有効でないことを示し、それに代わる耐故障性の評価尺度として「関数の比較」を提案し、シミュレーションにより有効性を示した。また、この尺度に基づき、本来の仕様はさておき、故障時と無故障時で変化が少ないこと、という新たな耐故障性の定義方法を提案した。

これにより、従来の必須結合数による議論とは異なり、必ずしも全パタンについて学習が終了する必要はなく、さらに故障モデルが評価に含まれていないため、多重故障への適用がそのまま可能となることを示した。

汎化能力を評価するためのベンチマークとして正規分布の帯状パターン離散値/ 連続値問題を提案し、シミュレーションを行った。

その結果、離散値については、耐故障性に優れるネットワークの方が、汎化能力に優れるという従来の報告通りの結果が得られたが、連続値問題においては、逆の結果が得られた。これは、連続値問題が、パタン認識などと類似した性質を持つと思われる離散値問題とは異なった中間層素子の役割を要求するためであると考えられる。

また、ネットワークの特徴量として、ネットワークが実現する関数の滑らかさと、中間層素子の微係数を検討し、汎化能力と耐故障性に共通する要因を探ろうとしたが、これら

は汎化能力とは同調した動きを見せたものの、耐故障性に対しては同様のことはいえず、これらの特徴量を仲立ちとした議論はできないことがわかった。

6.2 今後の課題

本論文で提案したような特徴量では、耐故障性と汎化能力に共通する要因を得ることはできなかったが、このアプローチ自体が否定されたわけではなく、今後、新たな特徴量を検討することが課題として残っている。但し、何らかの裏付けに基づいた特徴量の探索が必要になると思われる。

本論文で提案した新しい定義と評価尺度により多重故障への適応が可能となったが、今回は多重故障と汎化能力との評価については行なっていない。今後、これらの関係についても検討を行なう必要がある。

謝辞

日頃あたたかく御指導下さいました、指導教官日比野靖教授、丹康雄助手に心より感謝致します。研究方針からプログラムに至るまで指導して頂いたおかげでなんとか論文をまとめることができました。

横田治夫助教授には、取り扱うテーマにおける御助言をいただき、厚く御礼申し上げます。

参考文献

[1] 丹 康雄, 南谷 崇, フォールトトレランスを有する階層型ニューラルネットワークとその性質, 電子情報通信学会論文誌, E76-DI, PP.380-389, (1993).

[2] 渡辺 栄治, 関数近似問題に対する階層型ニューラルネットワークの内部表現と汎化能力の関係に関する一検討, 電子情報通信学会, 信学技法, NC95-164,PP.77-84 (1996-03).

[3] 渡辺 栄治, パターン分類問題に対する階層型ニューラルネットワークの内部表現と汎化能力の関係電子情報通信学会, 信学技法, NC95-59, PP.79-86 (1995-10).

[4] 芳我 尚秀, 石川 眞澄, 各種構造学習法の構造化および汎化能力の比較, 電子情報通信学会, 信学技法,NC93-20,PP.63-70 (1993-06).

[5] 伊藤 顕一郎, 金岡 泰保, 富田 眞吾, パターンの非類似性を用いたバックプロパゲーション学習法, 電子情報通信学会, 信学技法,NC93-75, PP.57-63 (1994-02).

[6] 小川 栄光, 船田 純一, 最適汎化ニューラルネットワークの誤差最小構成法, 電子情報通信学会, 信学技法,NC93-127, PP.115-122 (1994-03).

[7] 真島 憲仁, 渡辺 晃子, 吉村 明, 永野 俊, OBD(Optimal Brain Damage) による中間層素子削減方法の有効性, 電子情報通信学会, 信学技法,NC93-129, PP.131-136 (1994-03).

[8] 上坂 吉則, 山岸 恵理, データ復元におけるヒトの予備知識について, 電子情報通信学会, 信学技法,NC95-11, PP.81-88 (1995-05).

[9] 田辺 佳彦, 松田 勘一, 出力誤差関数の微係数を利用した誤差逆伝搬法電子情報通信学会, 信学技法,NC95-36, PP.55-62 (1995-07).

[10] 石原 誠, 岸田 悟, 徳高 平蔵, 藤村 喜久郎, 階層型ニューラルネットワークにおける隠れユニット自動削減システムの検討, 電子情報通信学会, 信学技法,NC93-130, PP.137-143 (1994-03).

[11] D. C. Plaut, S. J. Nowlan and G. E. Hinton, Experiment on learning by back propagation, Carnegie-Mellon Univ., CMUCS-86-126 (1986).

[12] D.E.Rumelhart, International Conference on Neural Networks, Plenary Session (1988).

[13] 石川 眞燈, 忘却を用いたコネクショニストモデルの構造学習アルゴリズム, 人工知能学会誌, Ver5, No.5, pp.595-630 (1990).

[14] M. Ishikawa and H. Uchida, International joint conference on neural networks, II pp.375-367, Beijing (1992).

[15] S. Yasui, A new method to remove redundant connections in backpropagation neural networks, Introduction of of "parametric lateral inhibition fields," International joint conference on neural networks, II pp.360-367, Beijing (1992).

[16] Y.LeCun, J.S.Denker and S.A.Solla, Optimal brain damage, in D.S.Touretzky (ed.) Advances in Neural Infomation Processing Systems, 2, pp.598-605, Morgan Kaufman (1990).

[17] 松岡 清利 編著, ニューロコンピューティング -基礎と応用-, 朝倉書店, 1992,

[18] 和田 安弘, 川人 光男, 新しい情報量基準と Cross Validation による汎化能力の推定, 電子情報通信学会論文誌 D-II,Vol.J74-D-II No.7 pp.955-965,(1991).

[19] Efron B., Bootstrap Methods: Another Look at the Jackknife, The Annals of Statistics, 7, 1, pp.1-26 (1979).

[20] 坂上 勝彦, 横矢 直和, 弛緩法と正則化, 情報処理, Vol. 30, No. 9, pp.1047-1057 (1989).

[21] 松岡 清利, An Approach to the Generalization Problem in the Backpropagation Method, 電子情報通信学会論文誌 D-II,Vol.J73-D-II, No.6, pp.897-905,(1990).

[22] 赤穂 昭太郎, 正則化法によるニューラルネットの学習について, ニューロコンピューティング, NC 91-76, pp.85-90 (1991).

[23] Nait Charif Hammadi, Hideo Ito, A Learning Algorithm for Fault Tolerant Feedforward Neural, Networks, IEICE TRANS. INF. and SYST. ,VOL. E80-D, NO. 1 JANUARY (1997).

[24] 中澤 真介, 小川 英光, 最適汎化ニューラルネットワークの最適構成法, 電子情報通信学会, 信学技法,NC96-60, PP.17-24 (1996-12).

[25] C. Neti, M.H. Schneider, and E.D. Young, Maximally Fault Tolerant Neural Networks IEEE Trans. Neural Networks, 3, pp. 14-23 (1992).

付録 A. 評価器

NAME
ニューラルネットワークの評価器

SYNOPSIS

nnftn, browser, unit, mse, nnft2, msen, link, func, hidden,
multi, deriv, smooth,

DESCRIPTION

nnft は、ある問題に対する学習が終了し、確立されたニュラールネットワークに
おいて、USER が求めたい値を得るためのプログラムである。USER が指定したい結合
に故障を発生させた場合の内部状態、誤差、無故障時との比較等を見る事ができる。

OPTIONS

nnftn - パターンエラーを表示する。全く同じパターン（同じ入力値）なのにもかか
わらず、理想出力の値が違っている、といった矛盾をしめす。エラーがない
場合は、[pattern error :0] と表されるだけであるが、エラー発生の場合は
　総エラー数、故障リンク、誤りのあるパターン番号、出力、誤差、の順で
表示される。
browser - 入力層、中間層、出力層の全ノードの出力を表示する。リンクの故障

（単一断線）と、表示したいパターン番号を指定できる。単一断線故障を
発生させる場合は、[Faulty link ID?] の後にその"リンクの番号"を入力
する。故障なしの場合は、同様に"-1"を入力すればよい。ある特定の
パターンを表示させたい場合は、[pattern #?] の後に、その"パターン番号"
を入力すればよい。すべてのパターンを表示させたい場合には、同様に"-1"
を入力すればよい。

unit - エッセンシャルユニットを表示する。出力表示は、パターン番号、出力層のユニット番号、中間層のユニット番号、出力誤差、の順で表示されている。

mse - 故障発生（リンクの単一断線）の平均自乗和誤差を表示する。すべてのリンクを断線させた場合の各パターンにおける誤差を出力し、最後に平均自乗和誤差を出力する。オプションの [-m] を使用すれば多重故障も表示可能である。この場合すべての故障の組合せ（総あたり）の各パターンの誤差を表示し、最後にその平均自乗和誤差を表示する。

nnft2 - 必須結合の数を表示する。各リンクを 1 本づつ断線させ（0 にすることにより）すべてのパターンでフォアードし、誤差 0.1 を越えたものを必須結合とみなす。
出力は、必須結合番号、リンク番号、パターン番号、出力、誤差、の順で示される。最後に必須結合数が表示されるが、[-v] オプションを使用すると、この必須結合数のみが表示される。

msen - 無故障時の各パターンにおける誤差の値と、平均自乗誤差を表示する。

link - 多重リンク故障における必須結合の値を表示する。この値は nnft2 において [-m] オプションを使用した場合と同値となる。

func - 故障時と無故障時の関数の比較。適当にサンプルされた空間での出力の比較をおこなう。デフォルトとして、x,y 領域共に 2〜-1 の範囲が指定されている。オプション、[-x][-y] を使用することによって範囲を指定することが可能。さらに、サンプルポイントは、デフォルトとして 0.1 間隔になっているが、これも、[-s] オプションを使用することによって自由に変えることができる。

hidden - 中間層における各ノードの出力をパターンごとに表示。パターンに対する平均と分散、各ノードごとにおける平均と分散を表示。相対するノード出力の相関係数（総あたり）を表示する。故障が発生した場合も表示可能で、
[Faulty link ID?] で断線させたいリンクの番号を入力する。故障なしの

値を得たい場合は"-1"を入力する。また、表示させたいパターンを指定する
こともでき、[pattern #?] でパターンの番号を入力すればよい。全パターン
を出力させたい場合は、"-1"を入力すればよい。

multi - 多重故障を発生させ、パターンに対する最小誤差、パターンに対する最小誤差、トータル誤差、平均自乗和誤差の順に出力する。

deriv - 中間層の微係数の値を出力する。微係数の値のみが出力される。

smooth - 関数の滑らかさを出力する。滑らかさの値のみが出力される。

-a accuracy ：0 から 1 までの値を入力。この値は許容誤差を示しており、これを越える値のエラーをかえす。

-i filename ：評価のために入力するファイル名

-t evaluate_function ：数値代入により、各評価関数を呼び出すことができる。対応する数値は、nnftn=2, browser=3, unit=4, mse=5, nnft2=6, msen=7, link=8, func=9, hidden=10, となっている。

-p pattern_filename ：評価したいパターンのファイルを入力できる。

-v verbose_switch ：すべてのデータを出力するのではなく必要な最小限度の情報を表示する。[-v 0] を指定する。

-l fault type ：故障のタイプ選択。以下の 0〜4 を選択できる。

```
case 0: /* stuck-at-0 */
w1[f_link] = 0.;
case 1: /* dilution */
w1[f_link] = w1[f_link] * dilution_rate;
case 2: /* random noise */
case 3: /* stuck-at-plus-infinity */
case 4: /* stuck-at-minus-infinity */
```

-m fault_number ：リンクの多重故障を発生させる。1 からリンク数まで入力可能。
1 より少ない値を入力した場合、"number of faults must be >0"
と表示される。

-d dilution_rate ：故障による得られる値の割合を指定する。0〜1 までの間。
-x range_of_x ：関数比較 func において、その x 軸の領域を指定するオプション。
3〜-2 までの範囲を指定したい場合、func -x 3 -2 と入力。
ただし、[-y] オプションと同時に使用。
-y range_of_y ：関数比較 func において、その y 軸の領域を指定するオプション。
3〜-2 までの範囲を指定したい場合、func -x 3 -2 と入力。
ただし、[-x] オプションと同時に使用。
-s split_rate ：関数比較 func で指定した領域において、対象とするサンプル
ポイントの間隔を設定できる。しかし、指定領域で等間隔にならない
ようなサンプルポイントを設定した場合は、"Not applopliate!
Change split_rate!"と表示される。
（ex, 3.0 × 3.0 領域において間隔 0.7 でサンプリングの場合など）

	-a	-i	-t	-p	-v	-l	-m	-d	-x	-y	-s
nnftn	○	○									
browser		○	○	○							
unit	○	○	○	○	○						
mse		○	○	○	○	○	○	△			
nnft2	○	○	○		○						
msen		○	○	○	○						
link	○	○	○	○	○	○	○	△			
func		○	○		○				○	○	○
hidden		○	○	○							
multi		○	○								
deriv		○	○								
smooth		○	○								

以上、○が使用可能を示している。使用可能であれば、同時に複数のオプション、
つまり、nnft -v 0 -p pat_filename という形で使うこともできる。

※ [-d] オプションは、[-l] オプションの case 1 と同時に使用したときのみ有効である。つまり、mse -l 1 -d (0.0〜1.0) 又は link -l 1 -d (0.0〜1.0)

付録 B. BP 法の分離直線

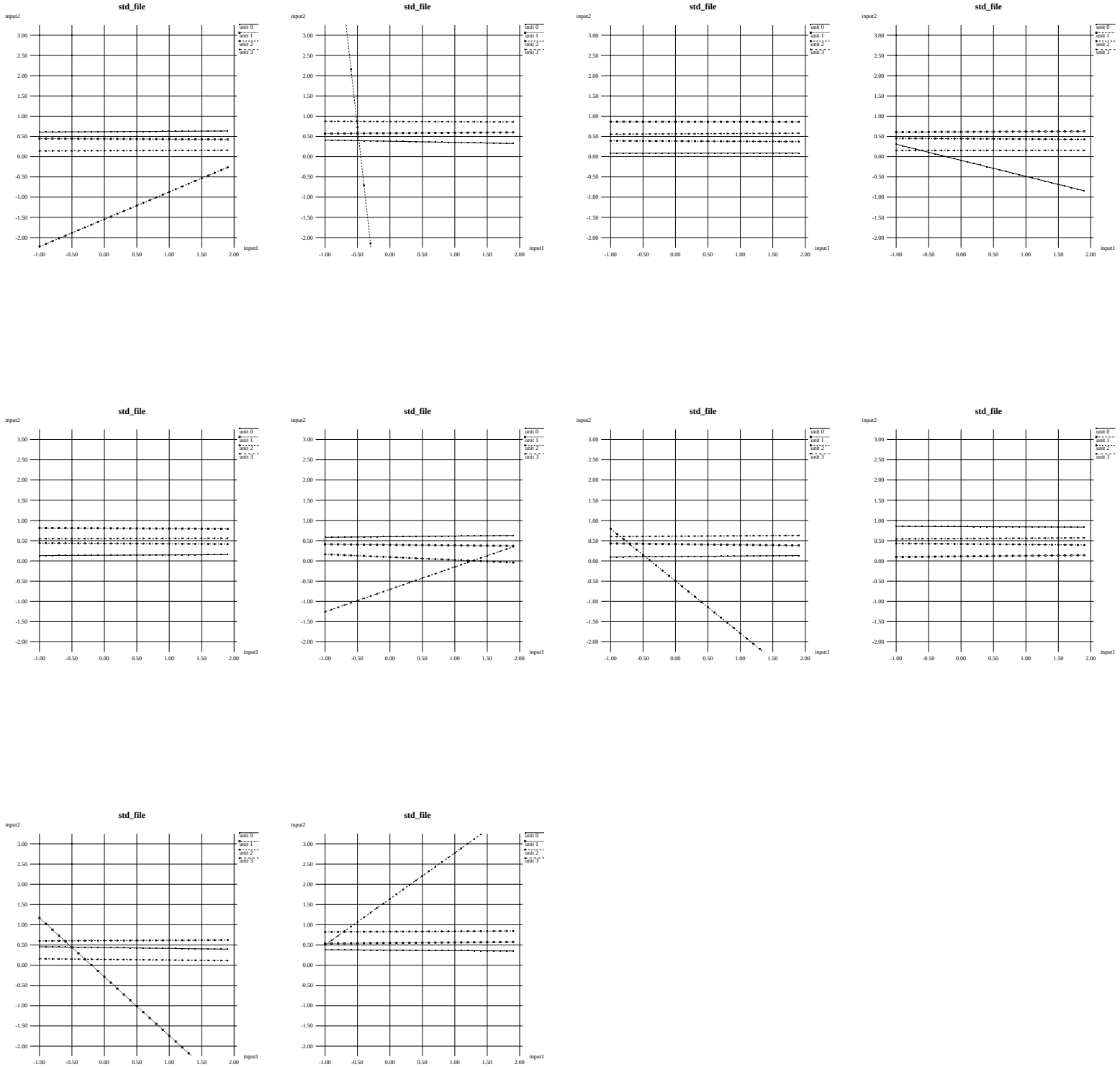

図 6.1: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 4）

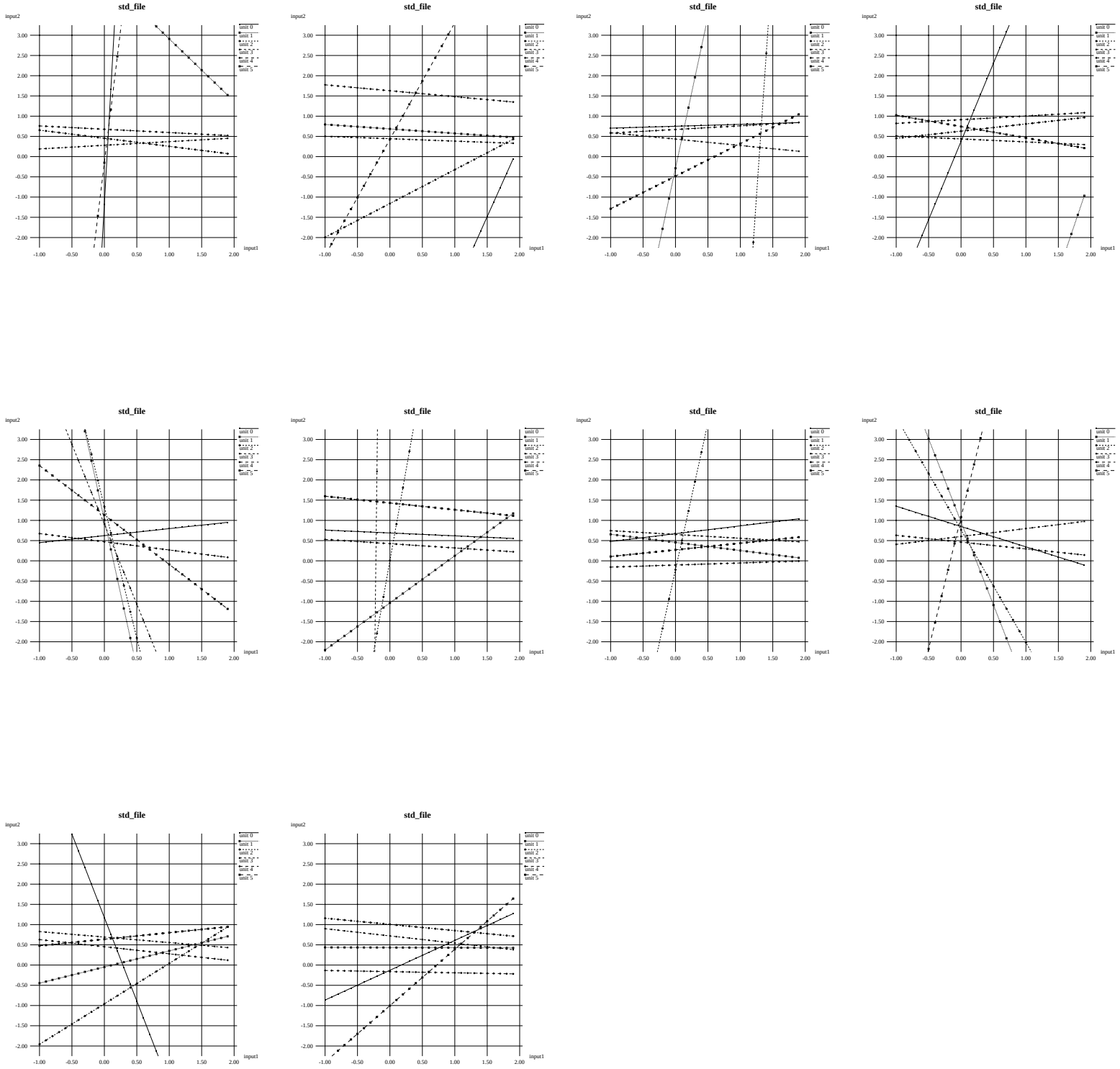

図 6.2: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 6）

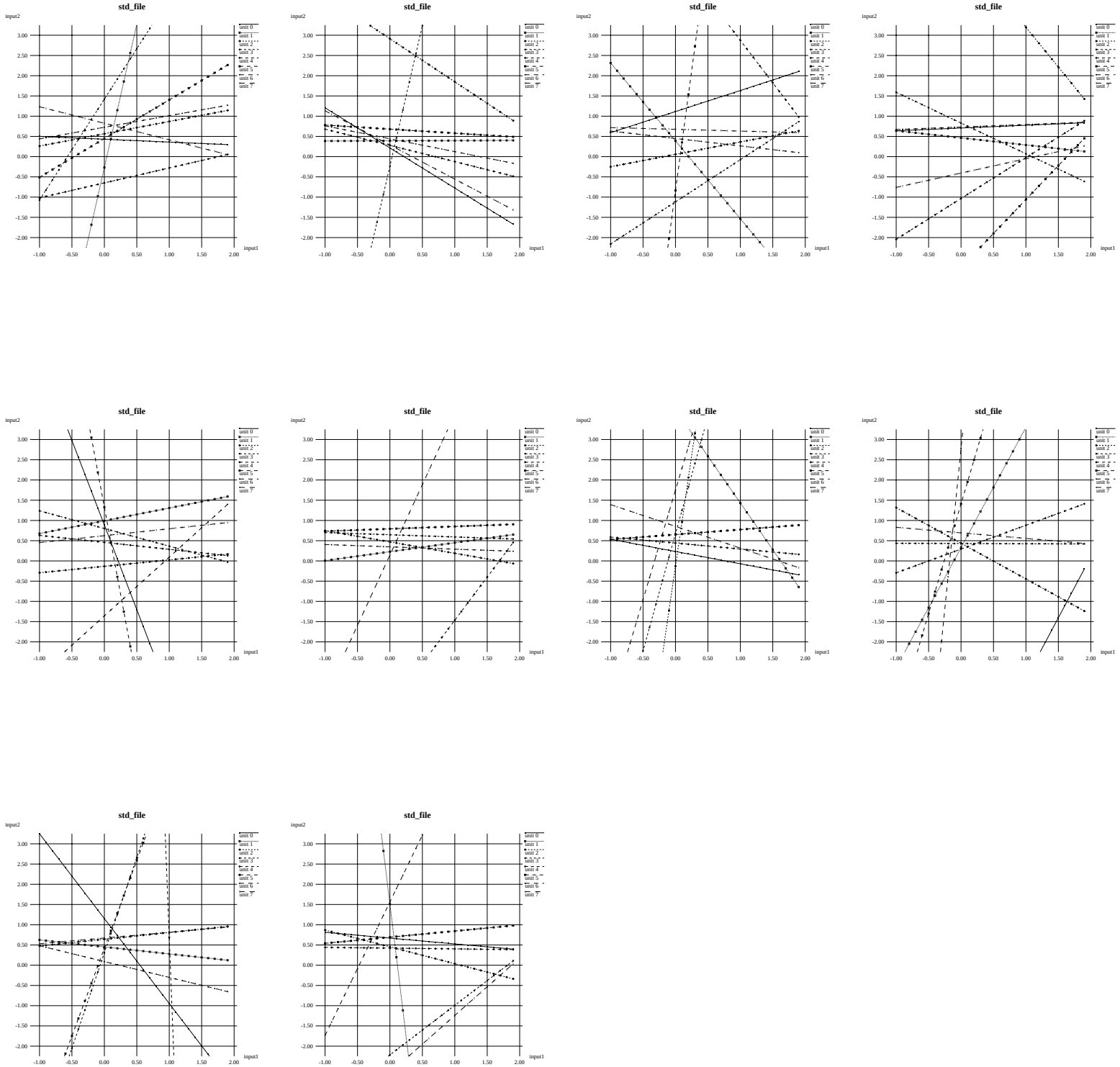

図 6.3: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 8）

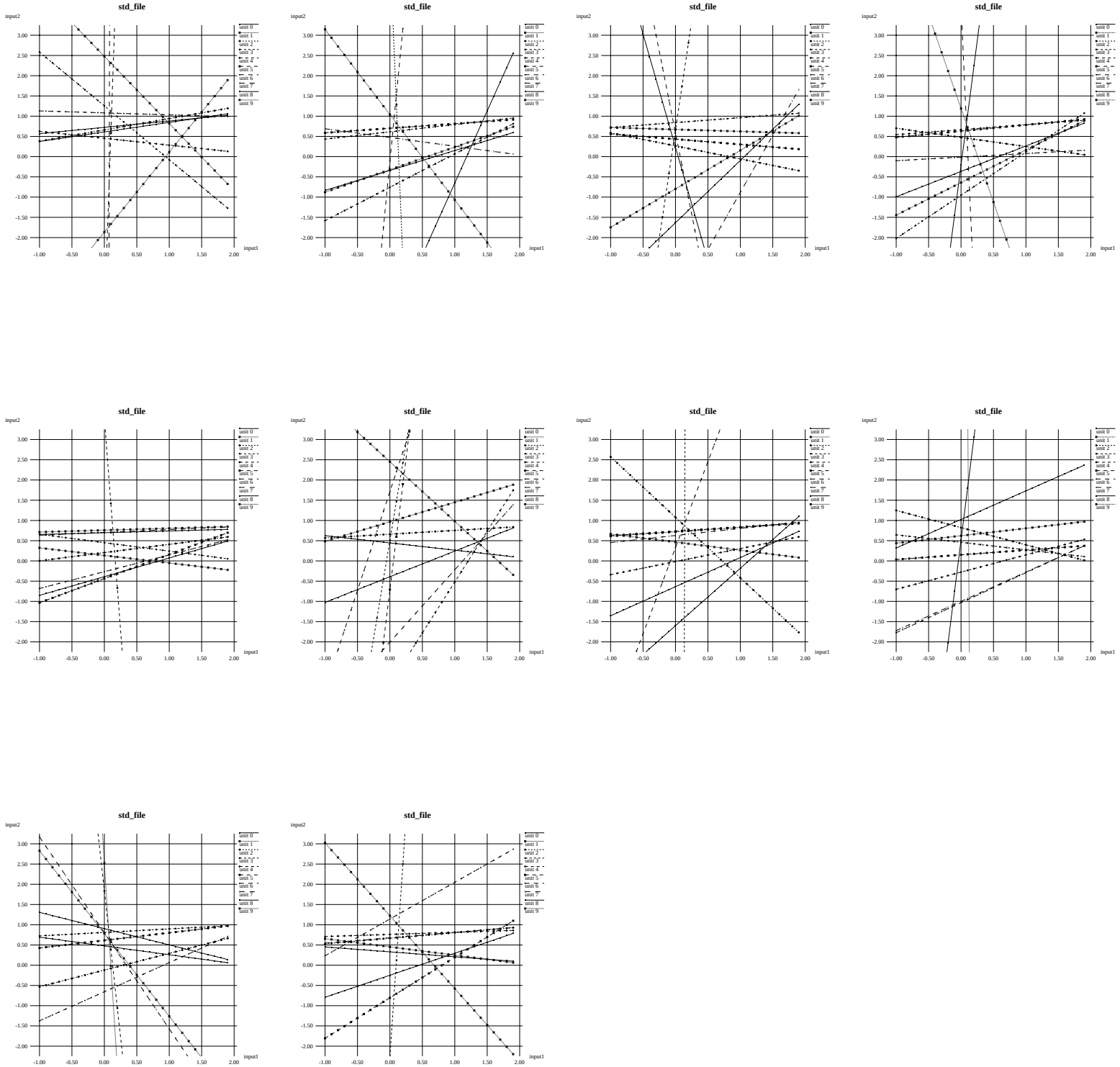

図 6.4: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 10）

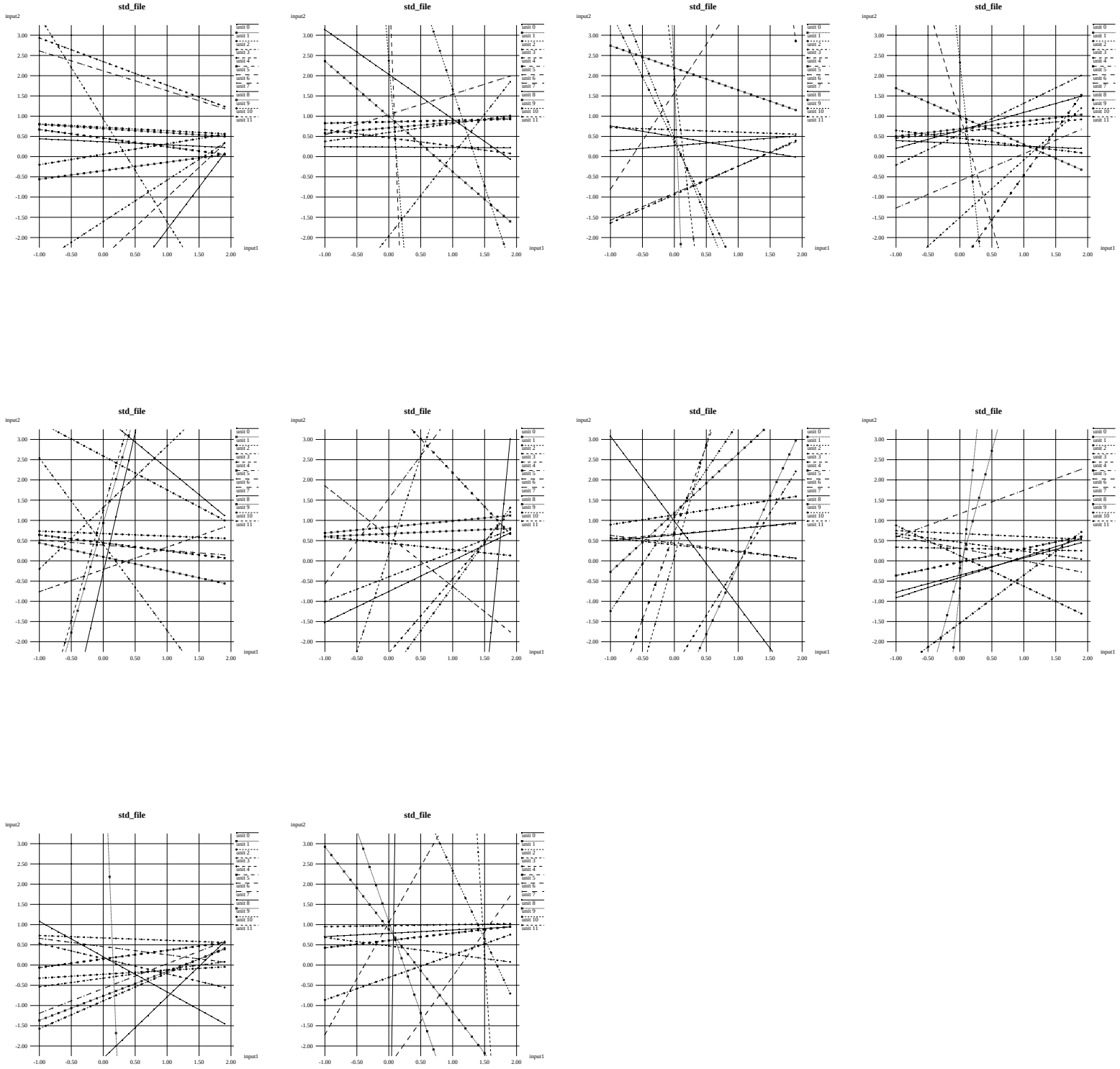

図 6.5: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 12）

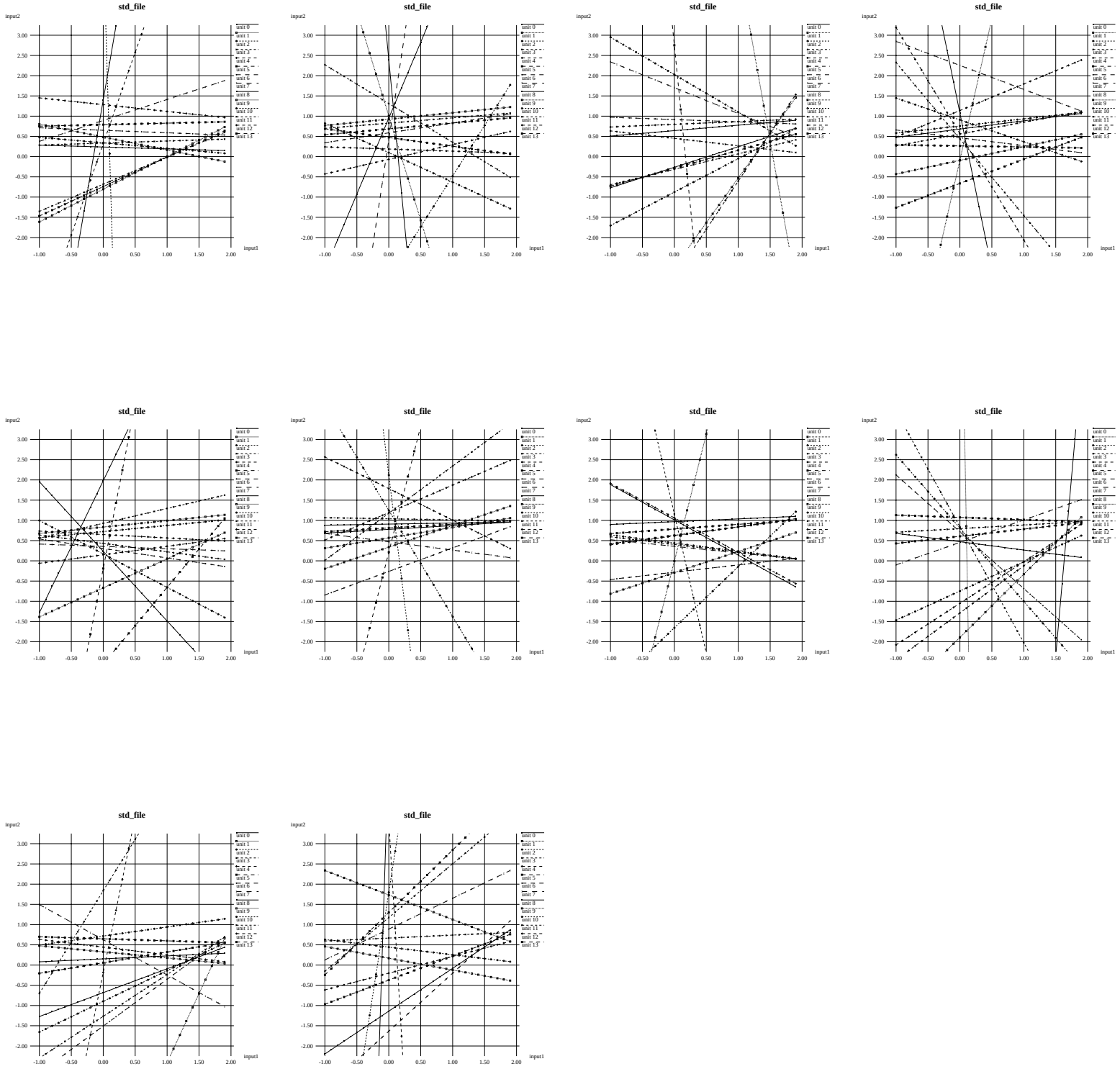

図 6.6: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 14）

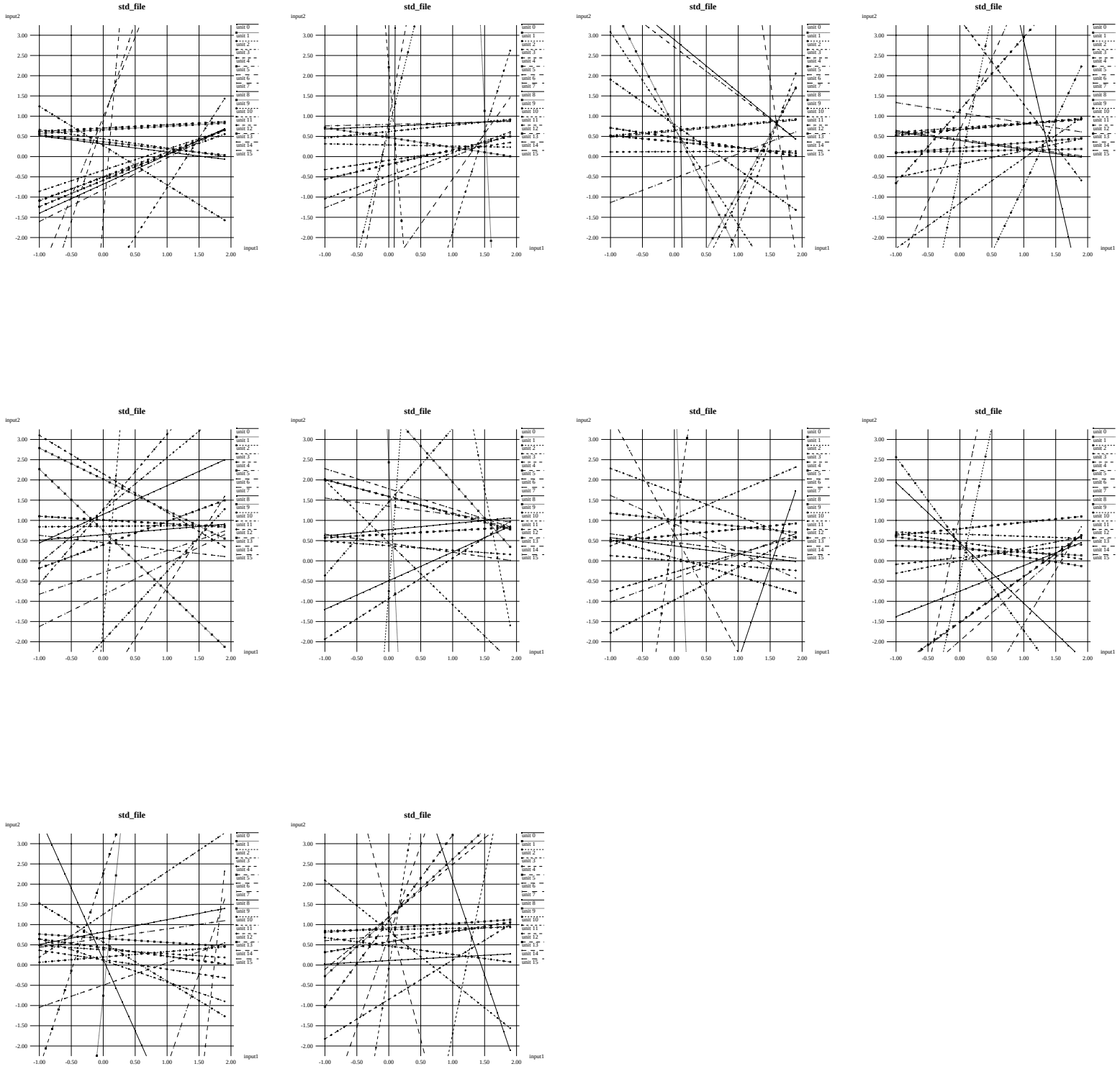

図 6.7: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 16）

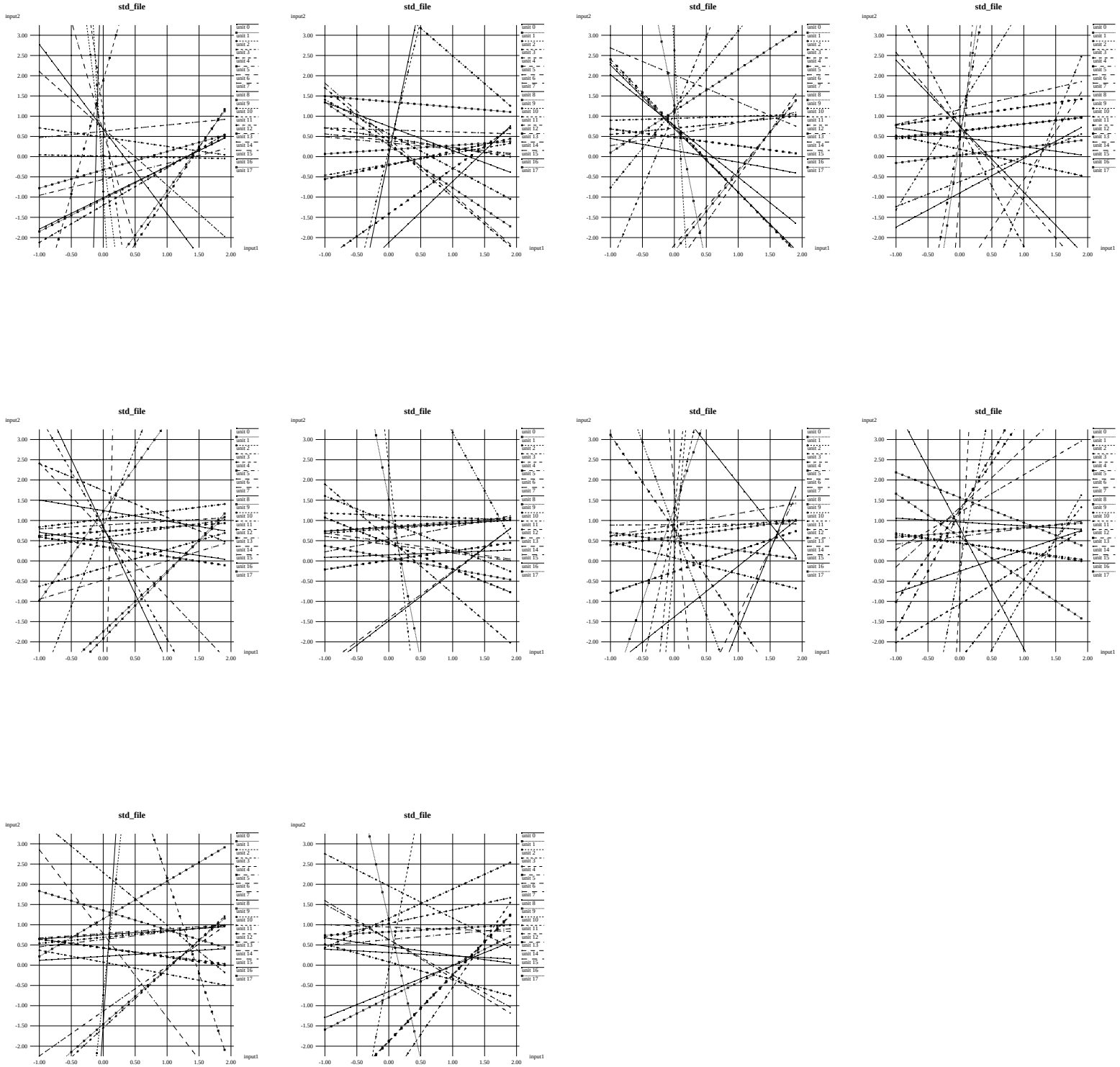

図 6.8: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 18）

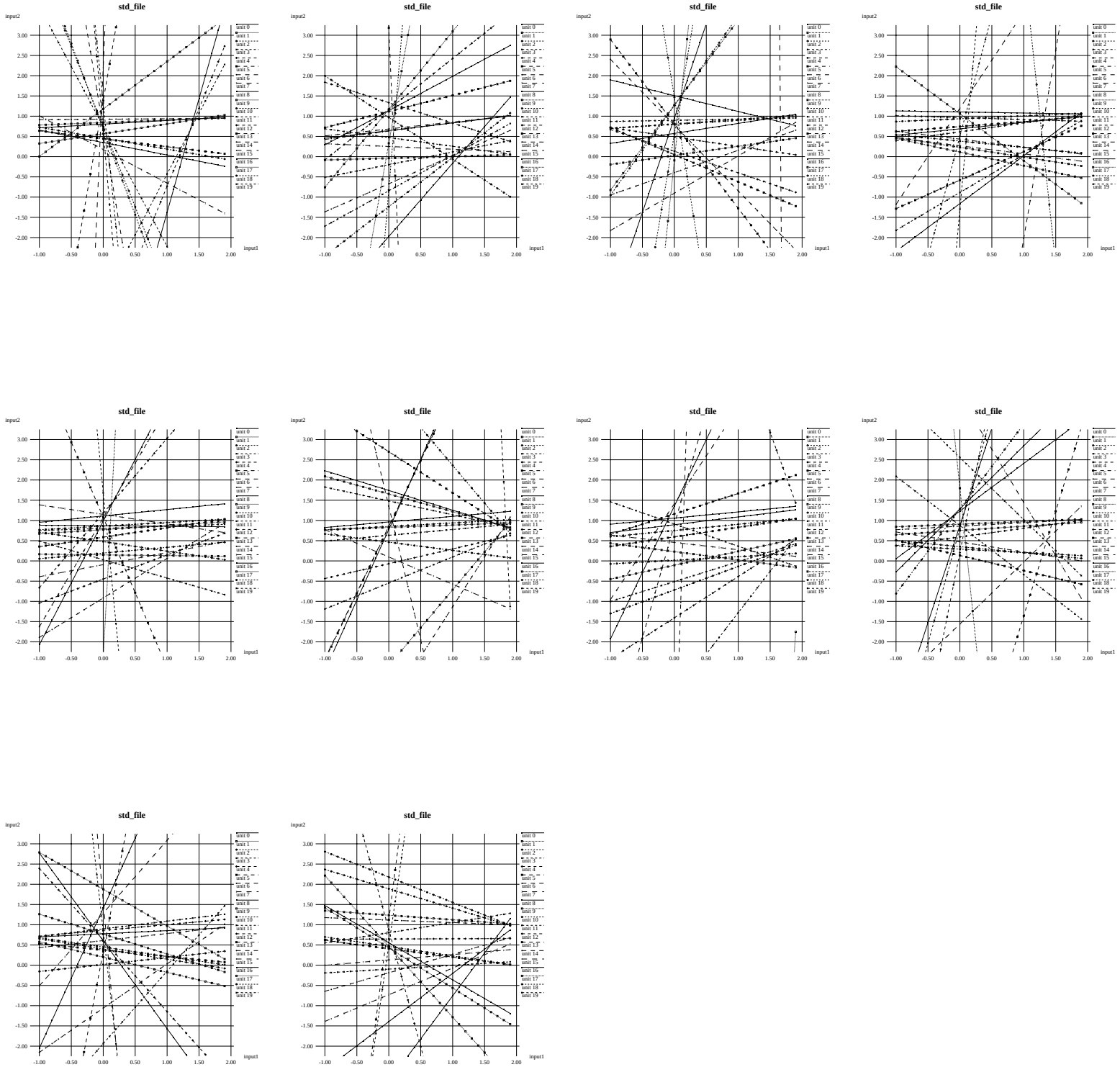

図 6.9: 帯状パターン離散を BP で学習した時の分離直線（中間層ノード 20）

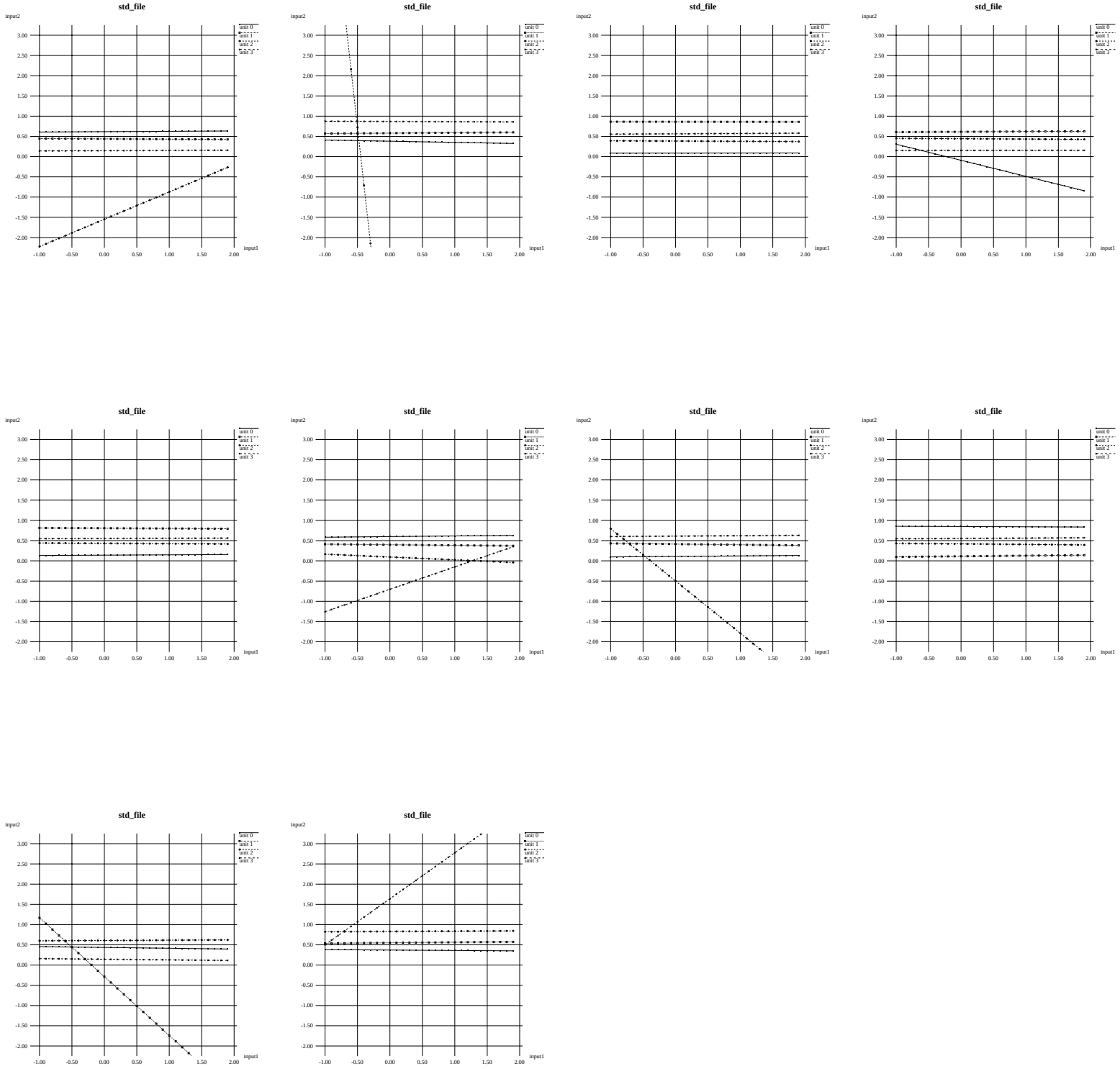

図 6.10: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 4）

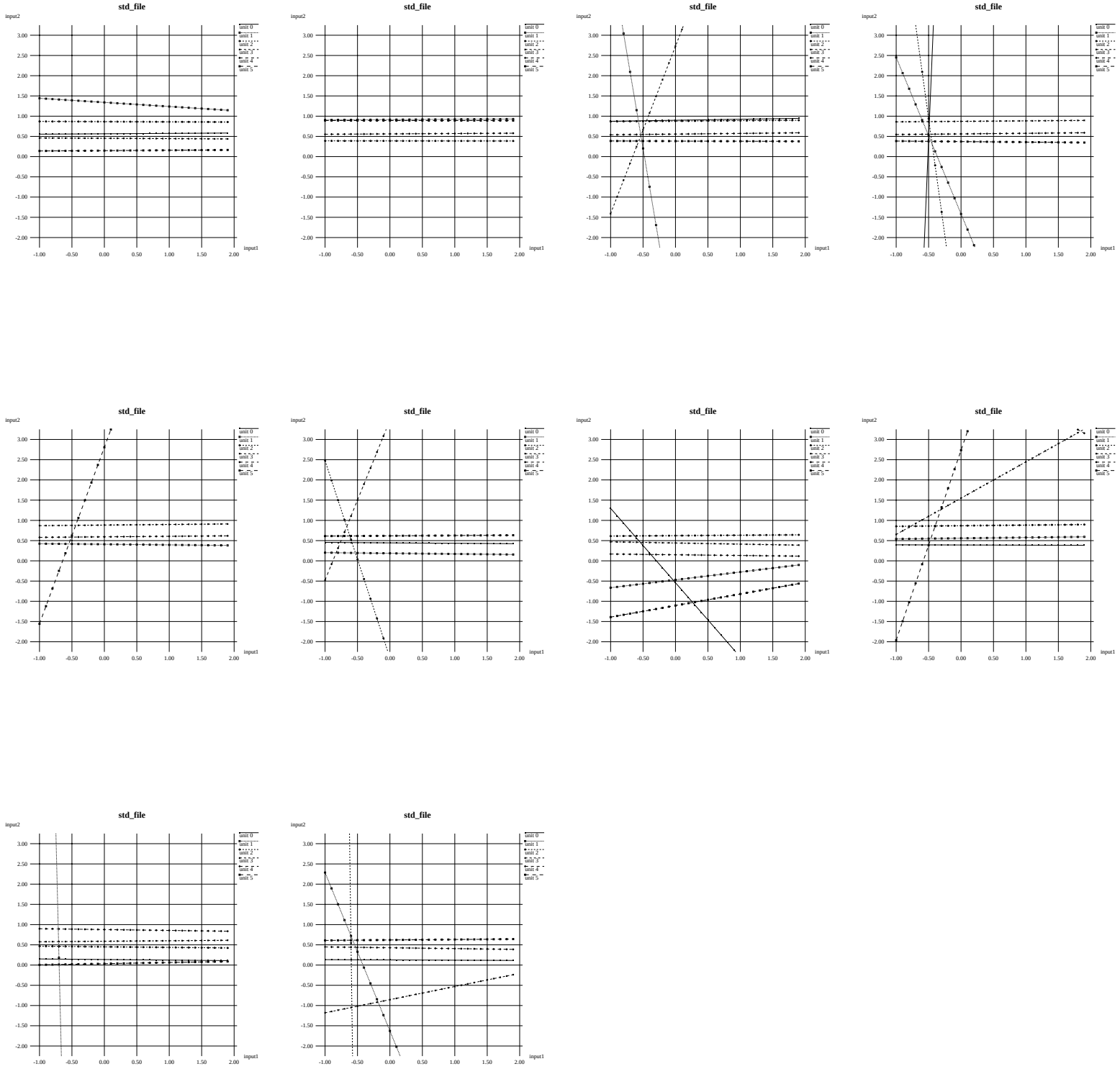

図 6.11: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 6）

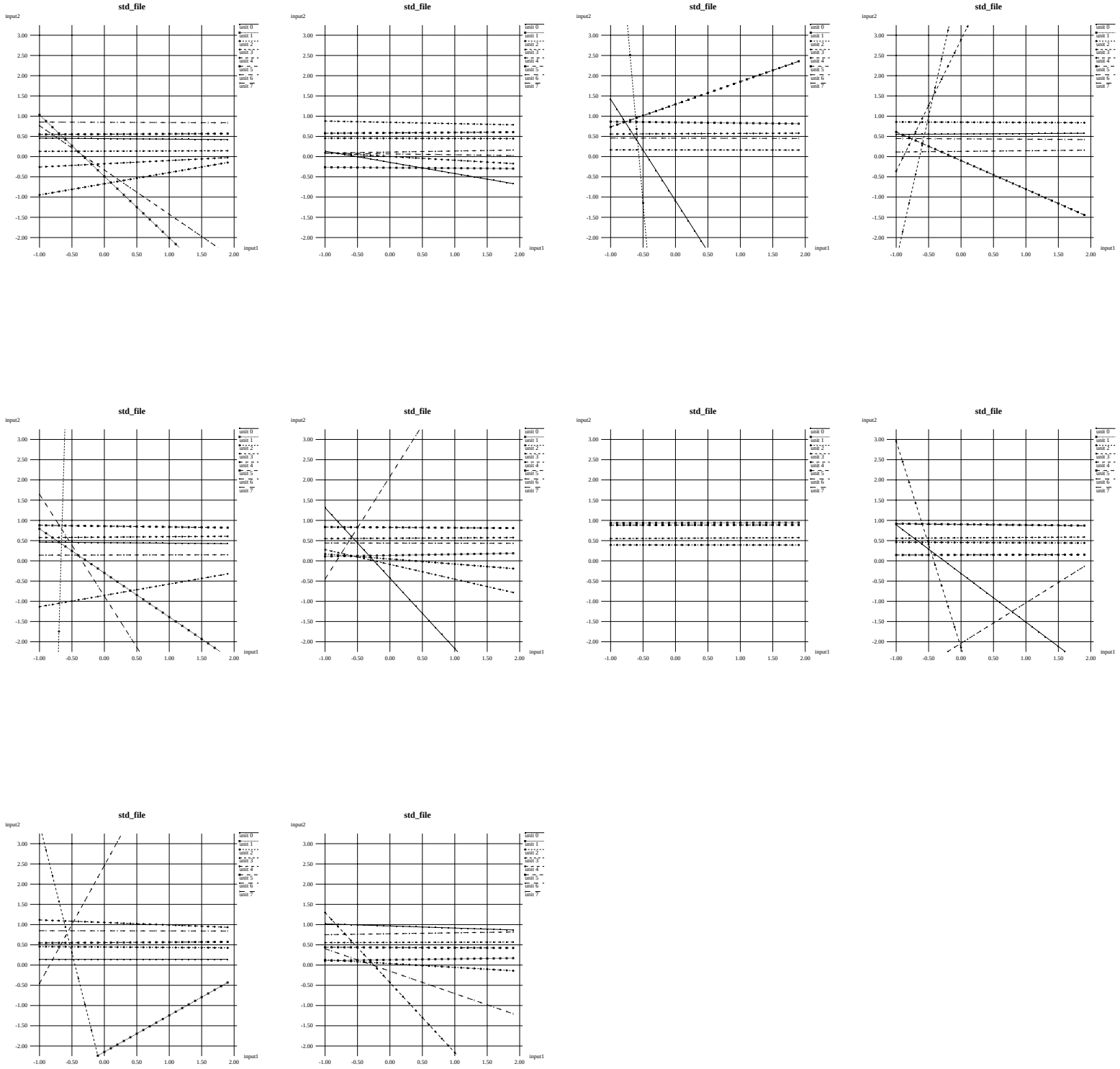

図 6.12: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 8）

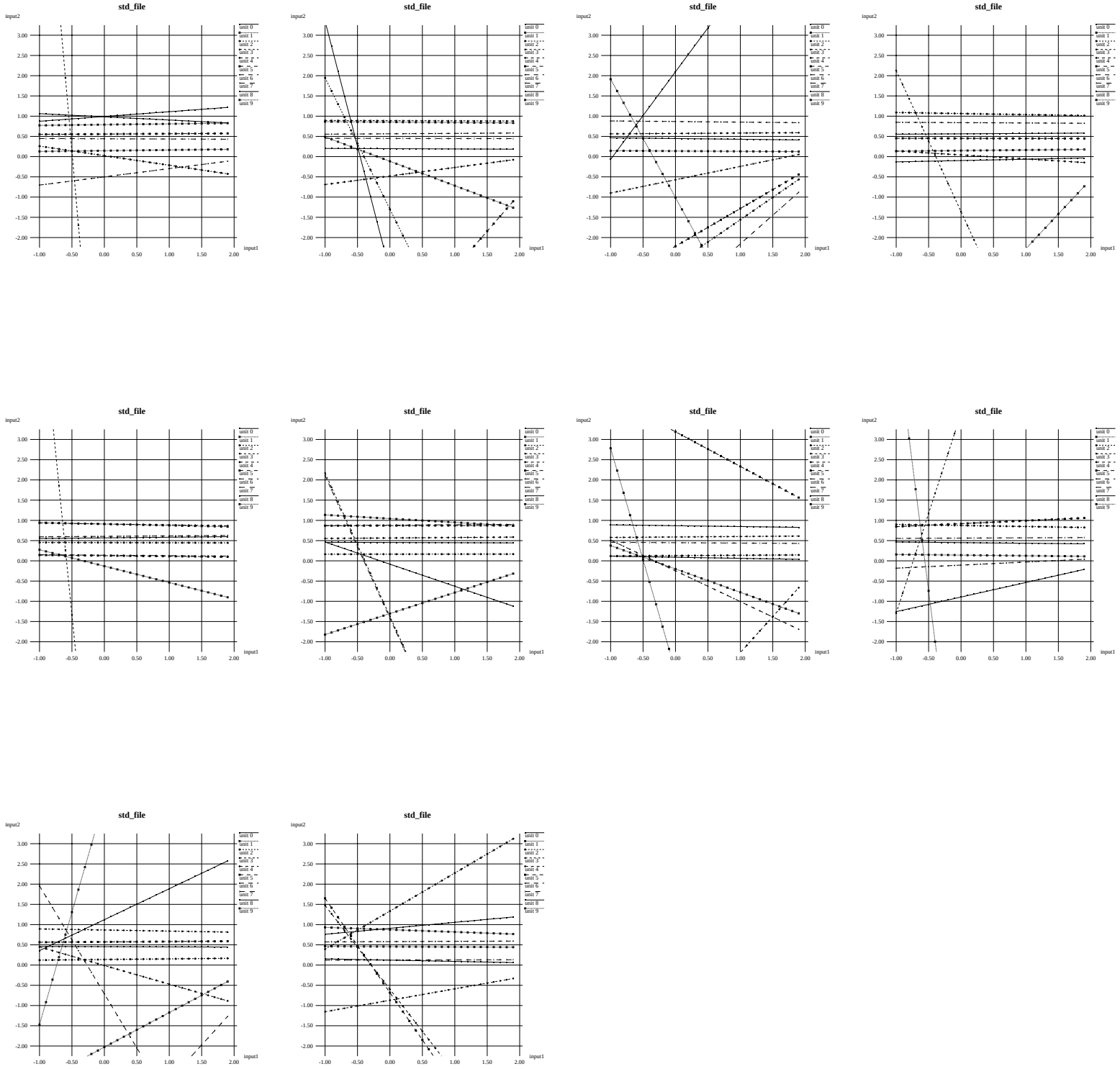

図 6.13: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 10）

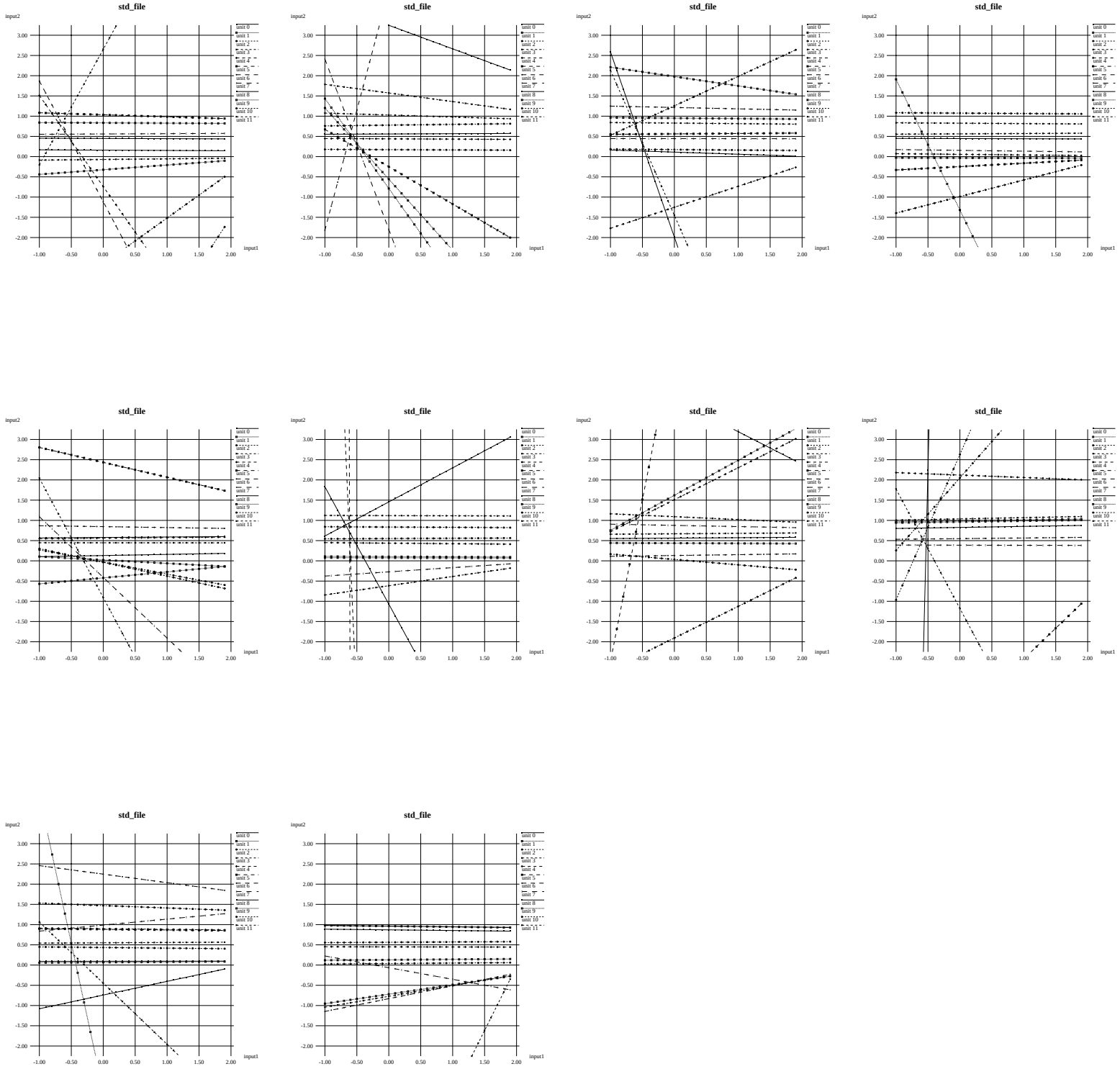

図 6.14: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 12）

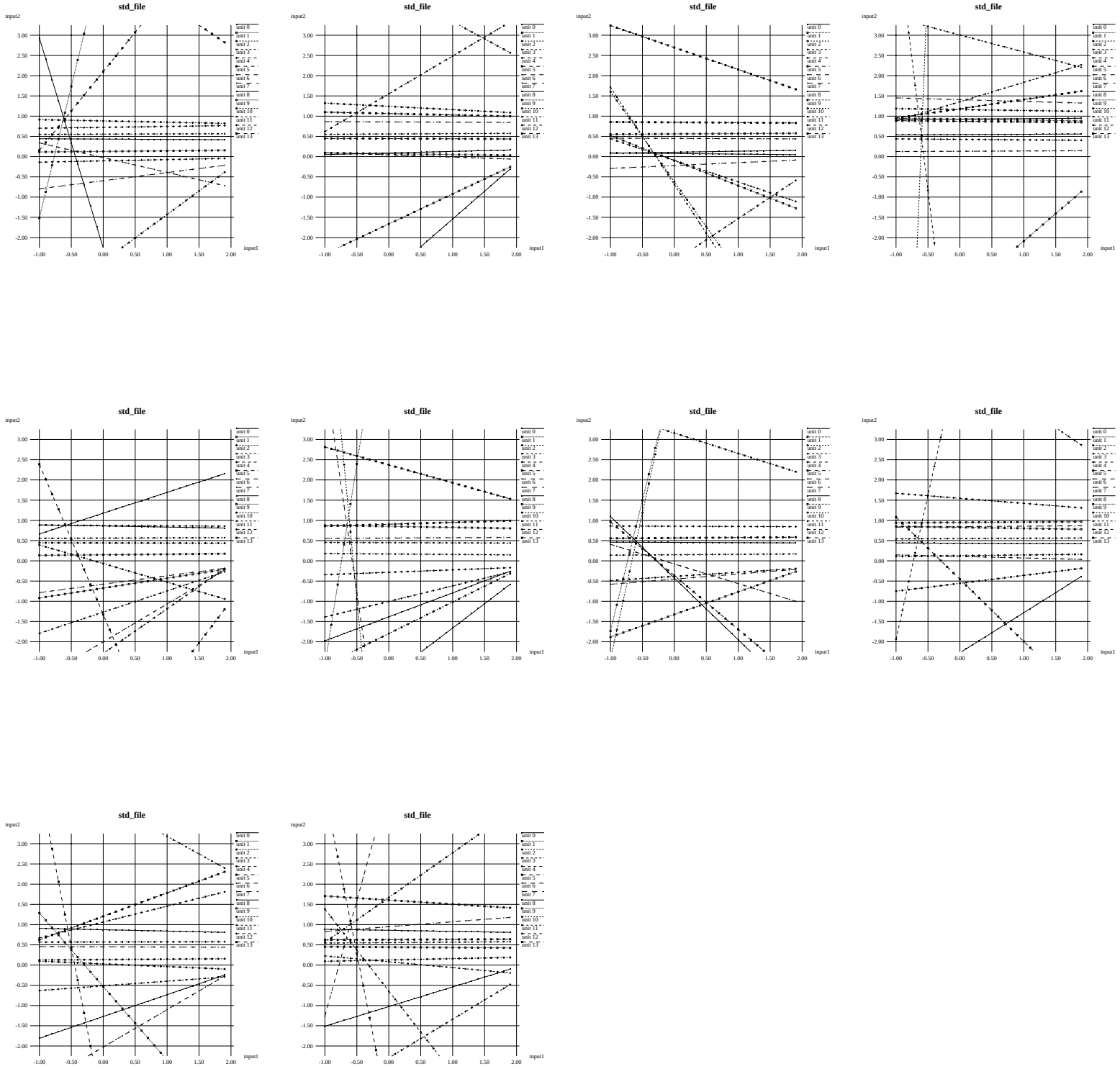

図 6.15: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 14）

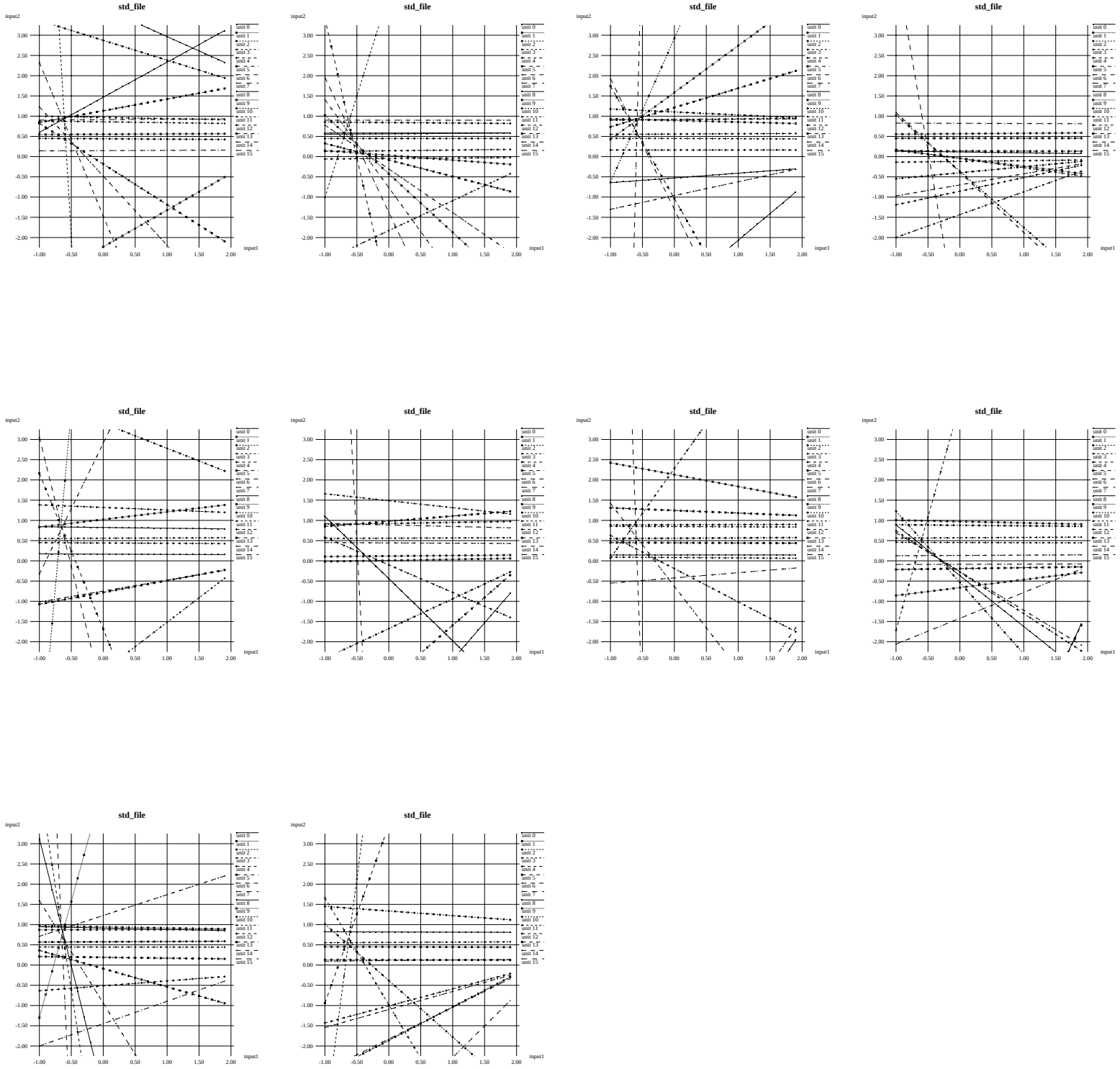

図 6.16: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 16）

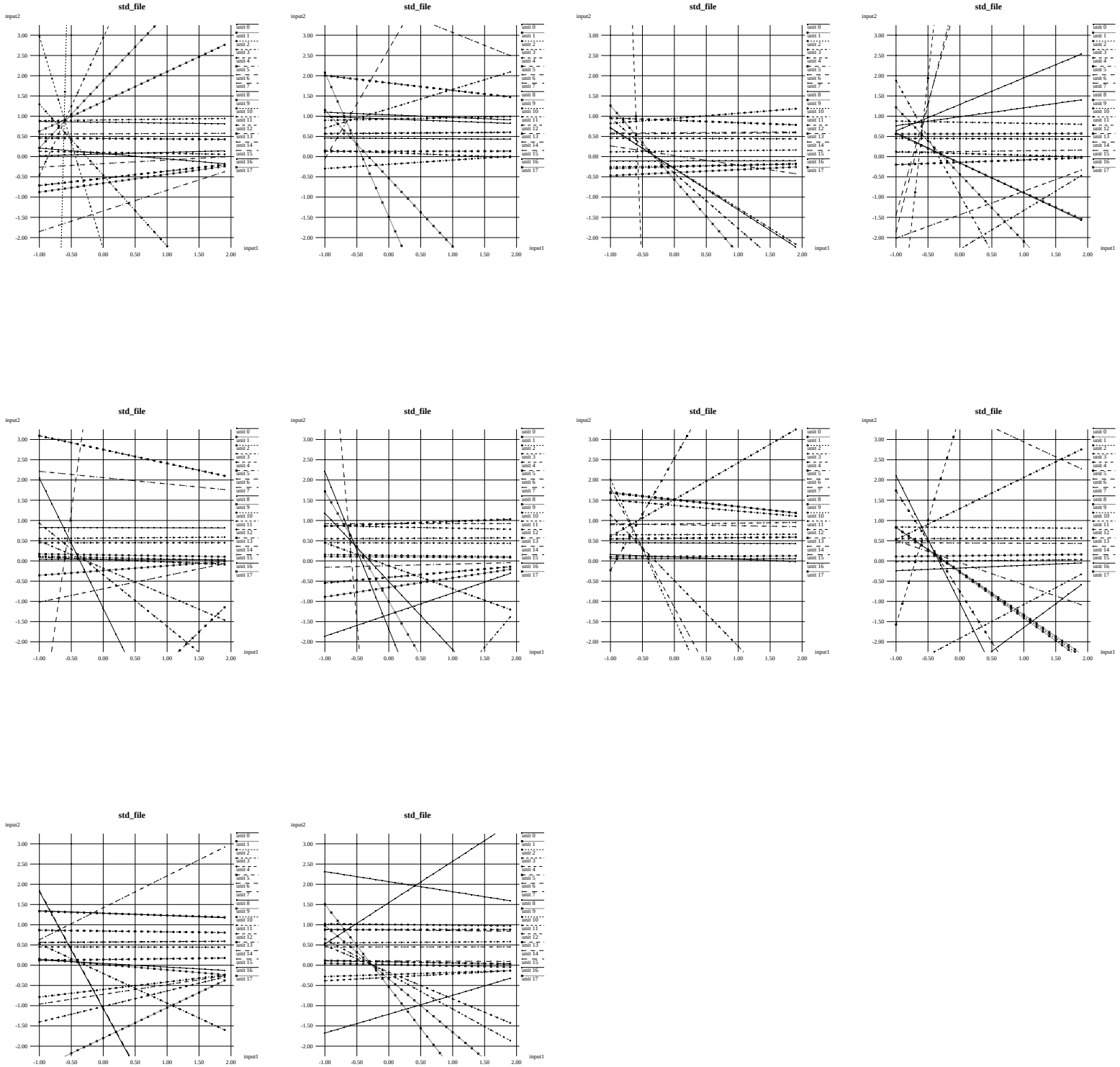

図 6.17: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 18）

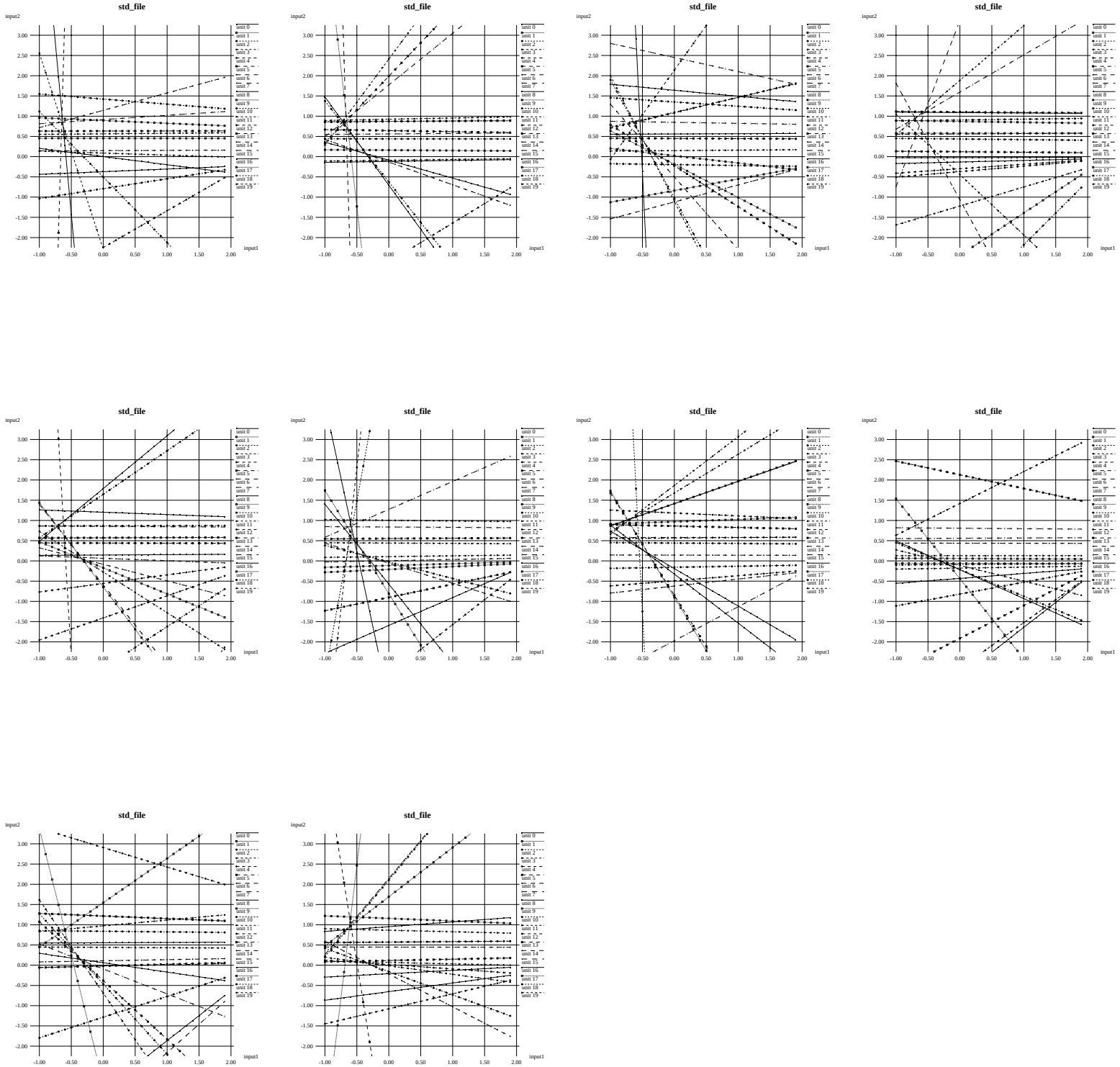

図 6.18: 帯状パターン連続を BP で学習した時の分離直線（中間層ノード 20）

付録 C. FTBP 法の分離直線

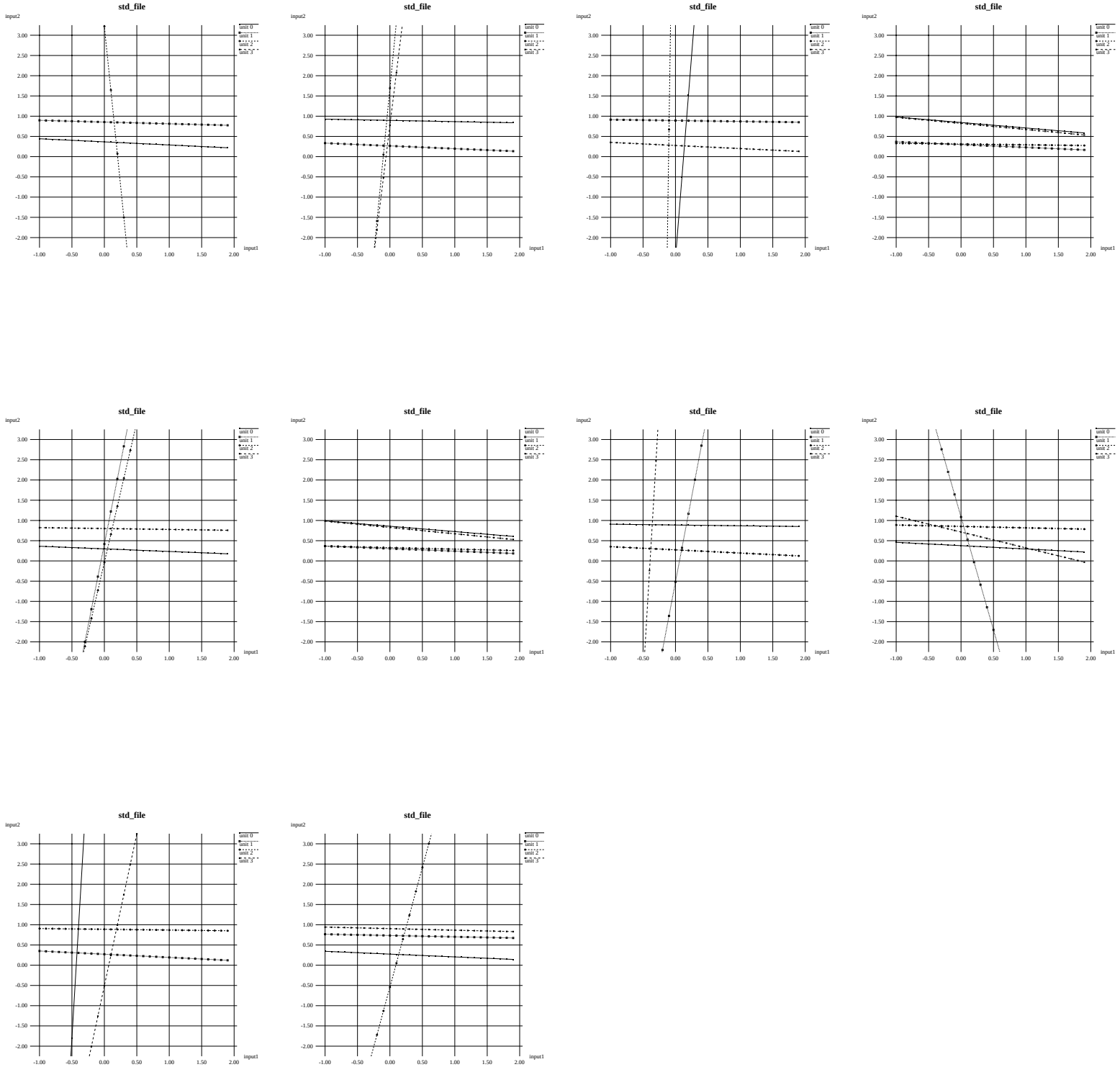

図 6.19: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 4）

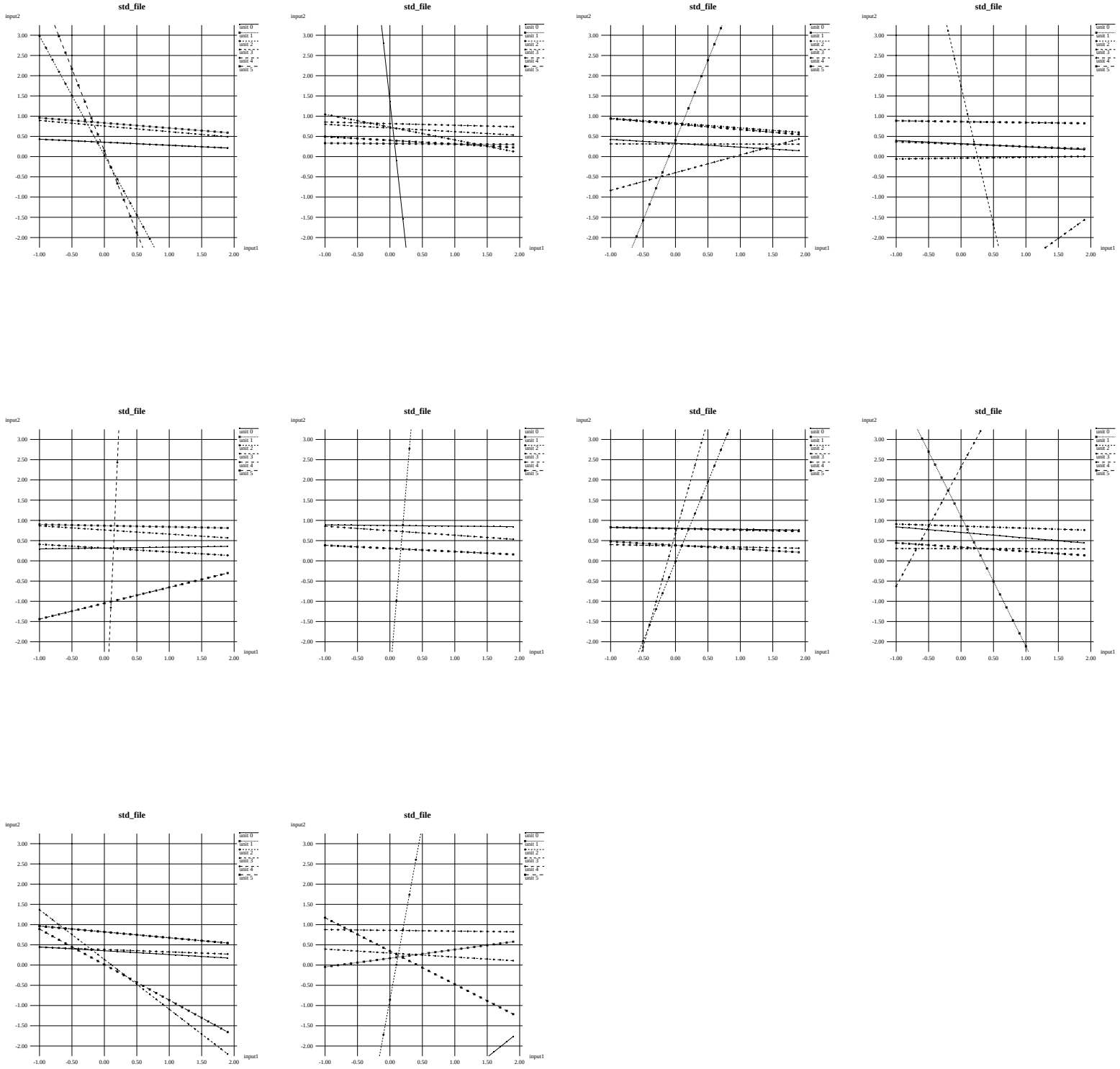

図 6.20: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 6）

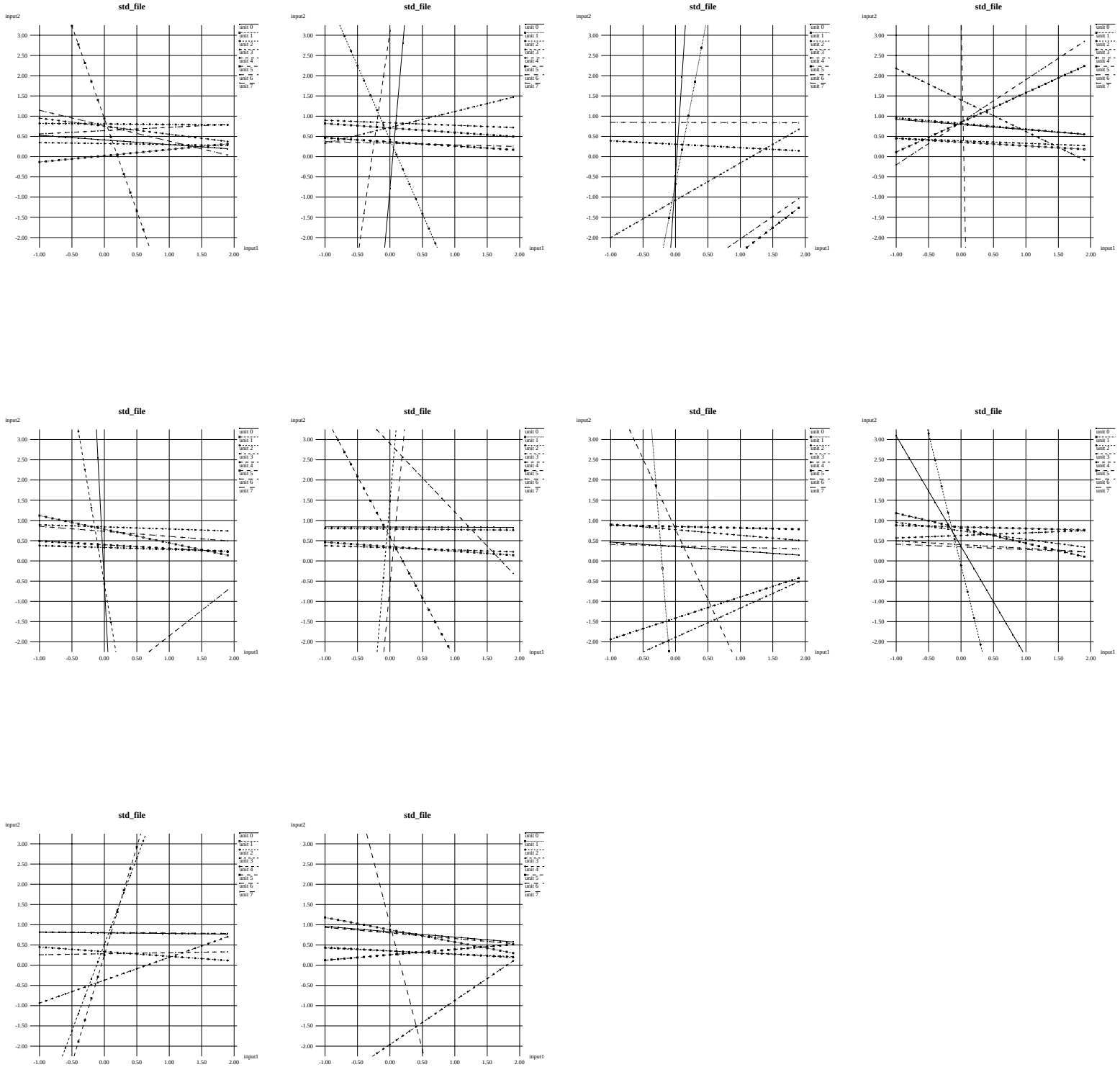

図 6.21: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 8）

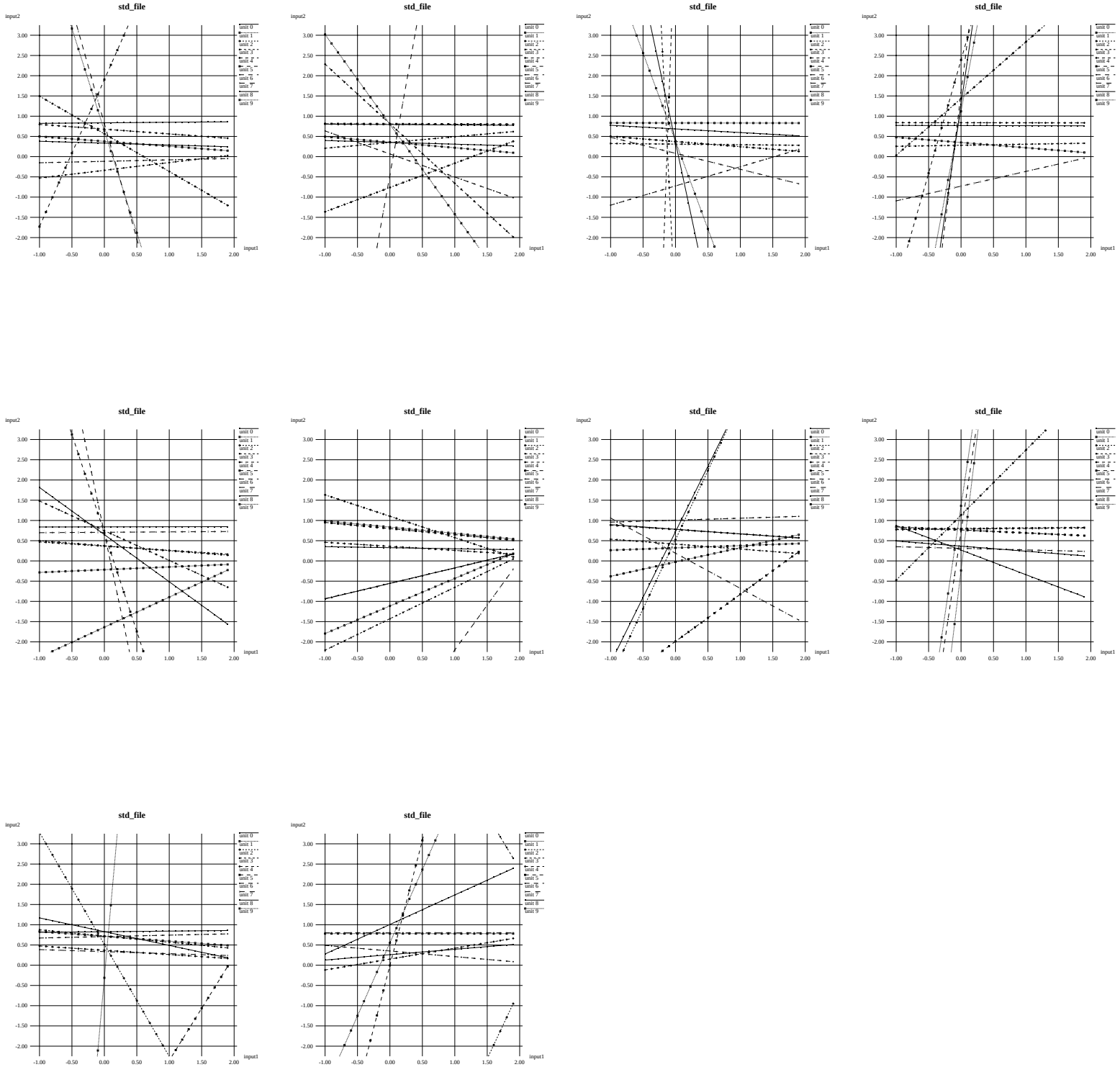

図 6.22: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 10）

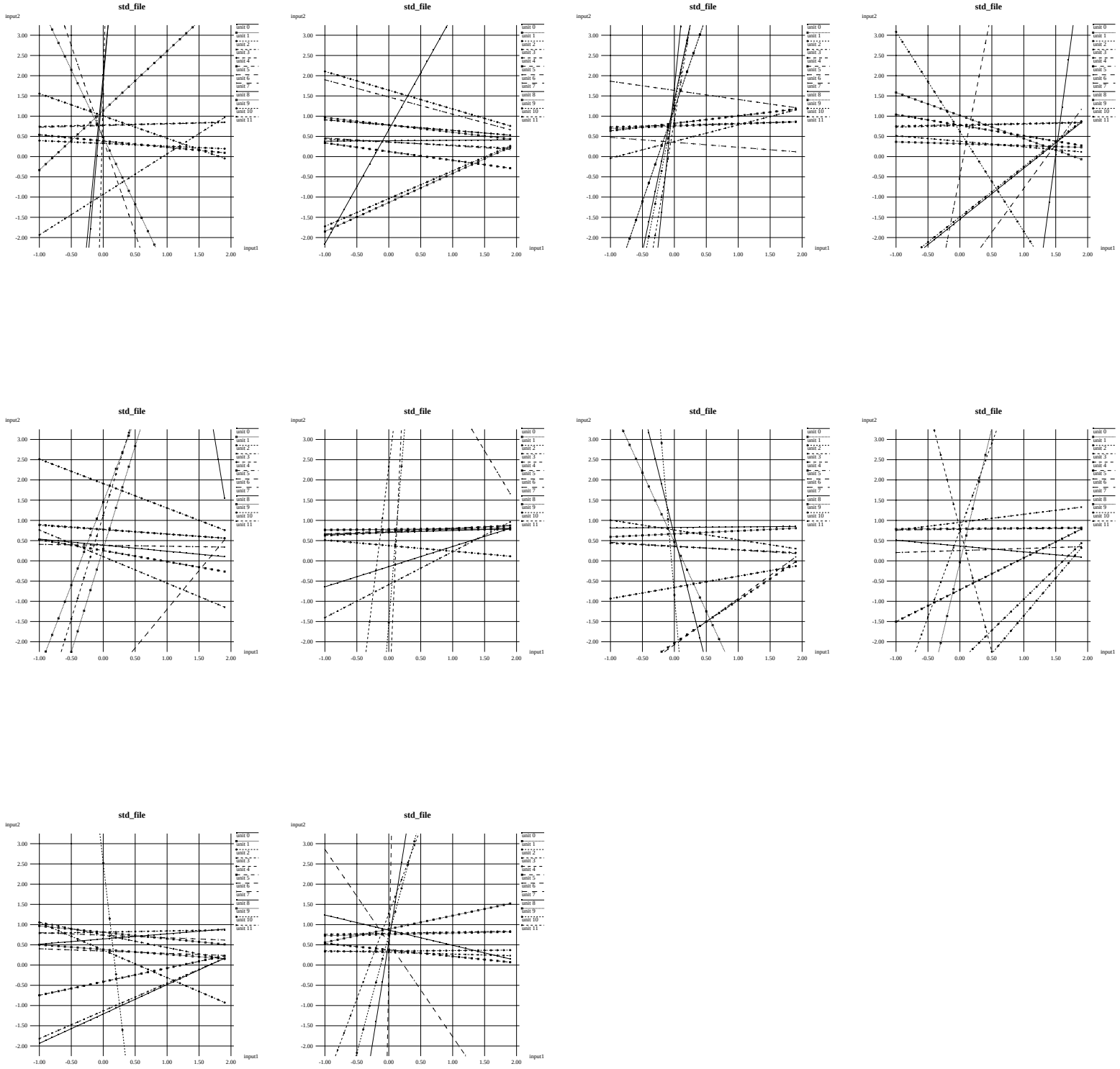

図 6.23: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 12）

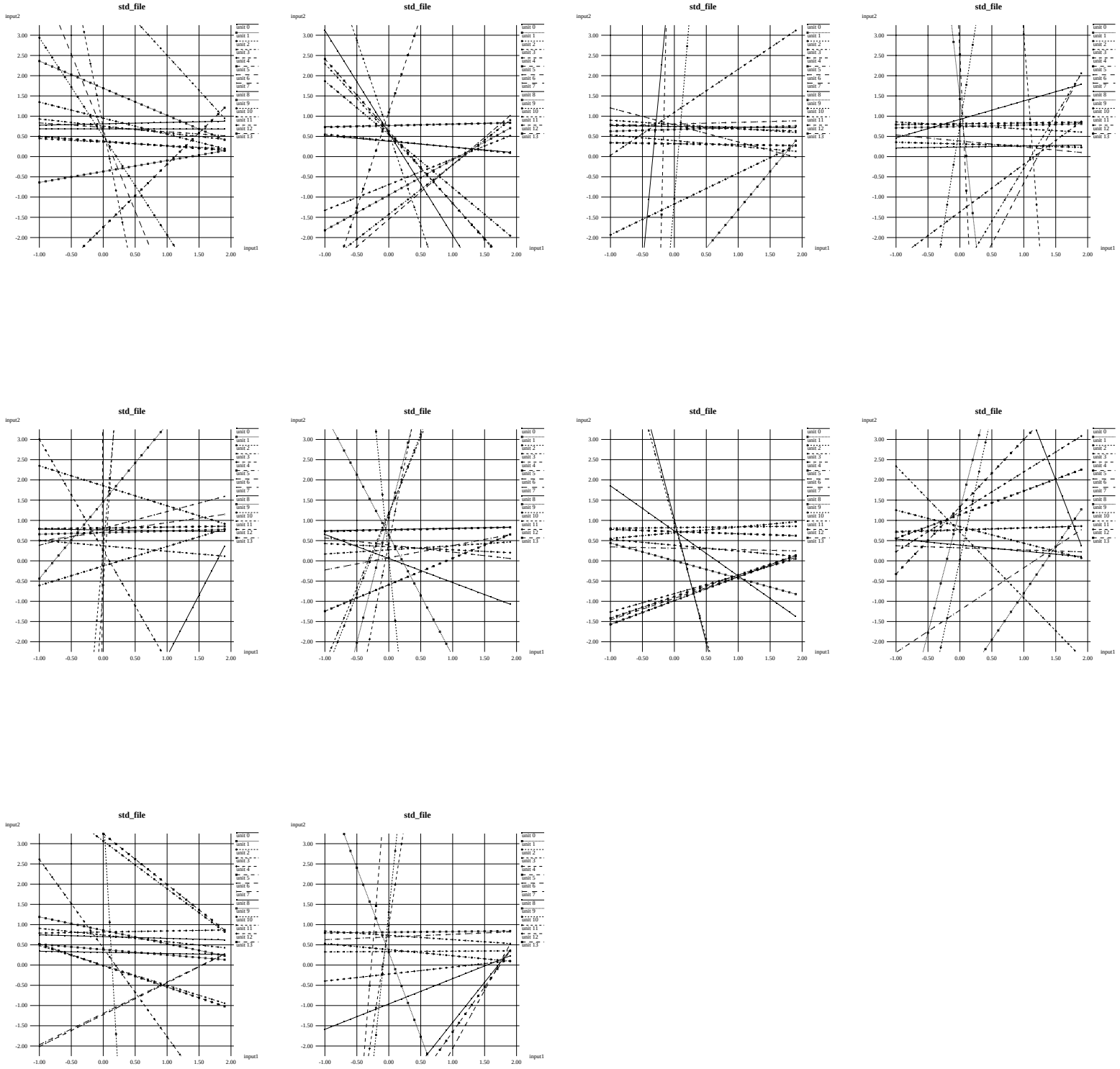

図 6.24: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 14）

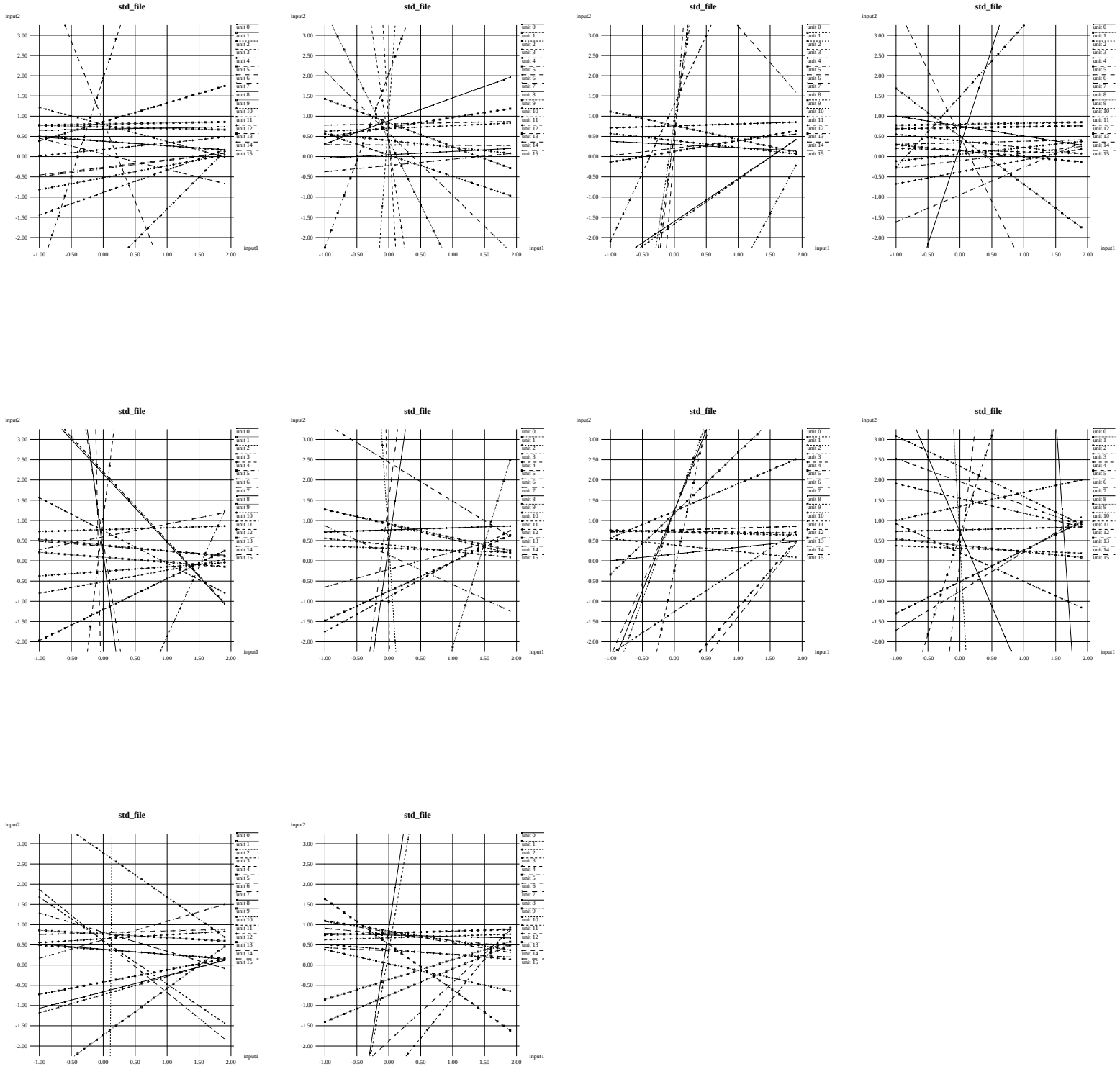

図 6.25: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 16）

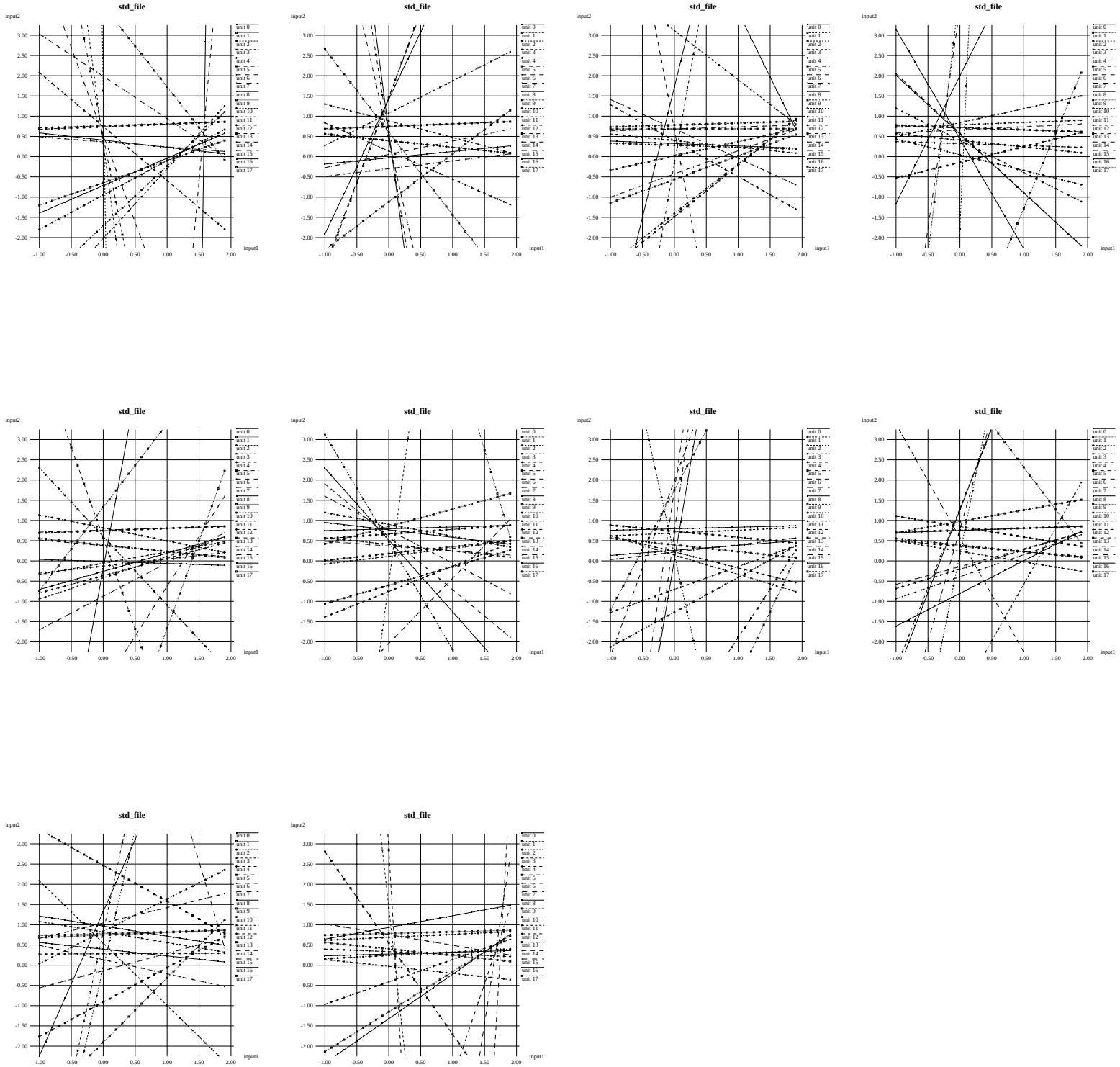

図 6.26: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 18）

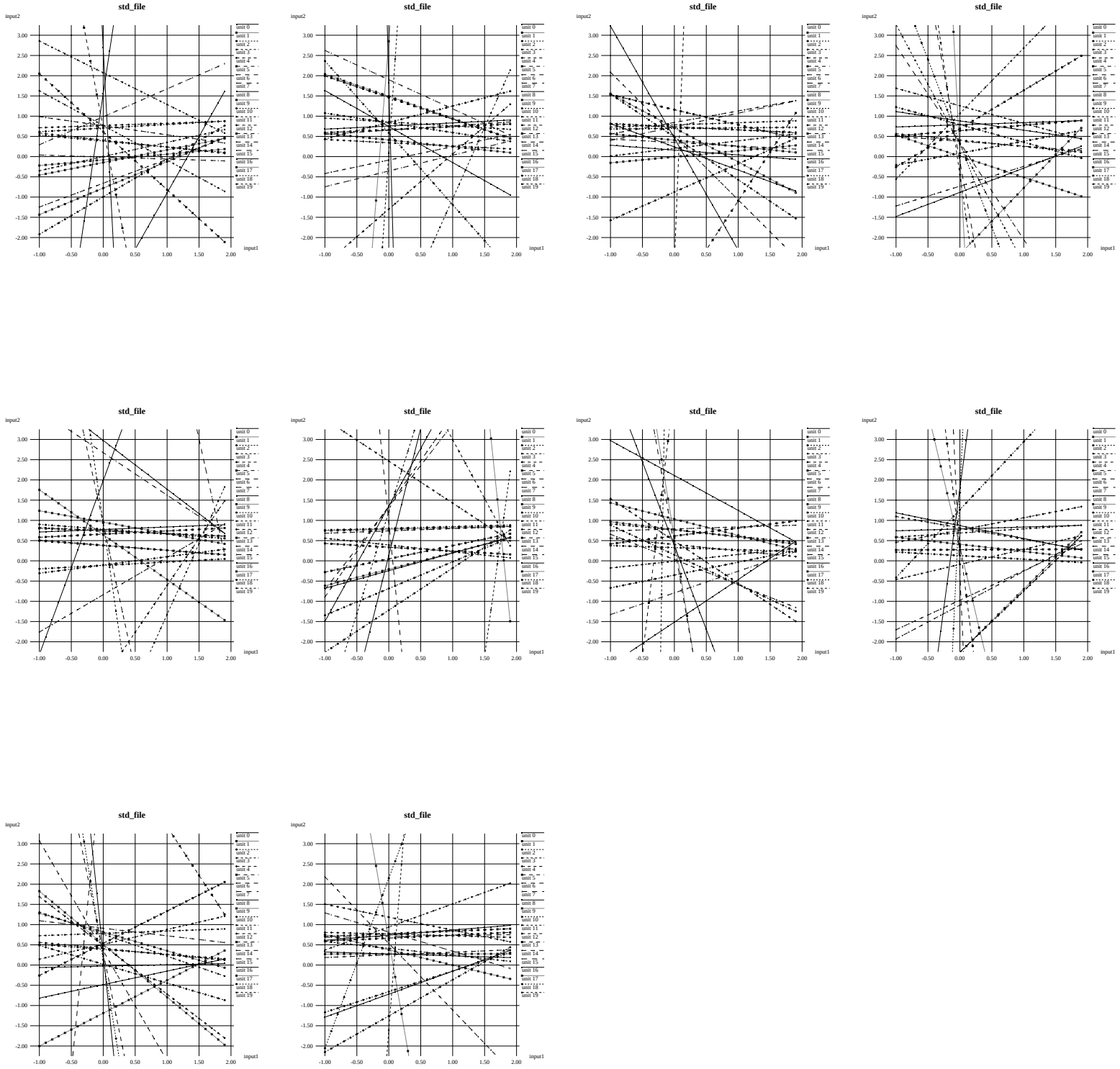

図 6.27: 帯状パターン離散を FTBP で学習した時の分離直線（中間層ノード 20）

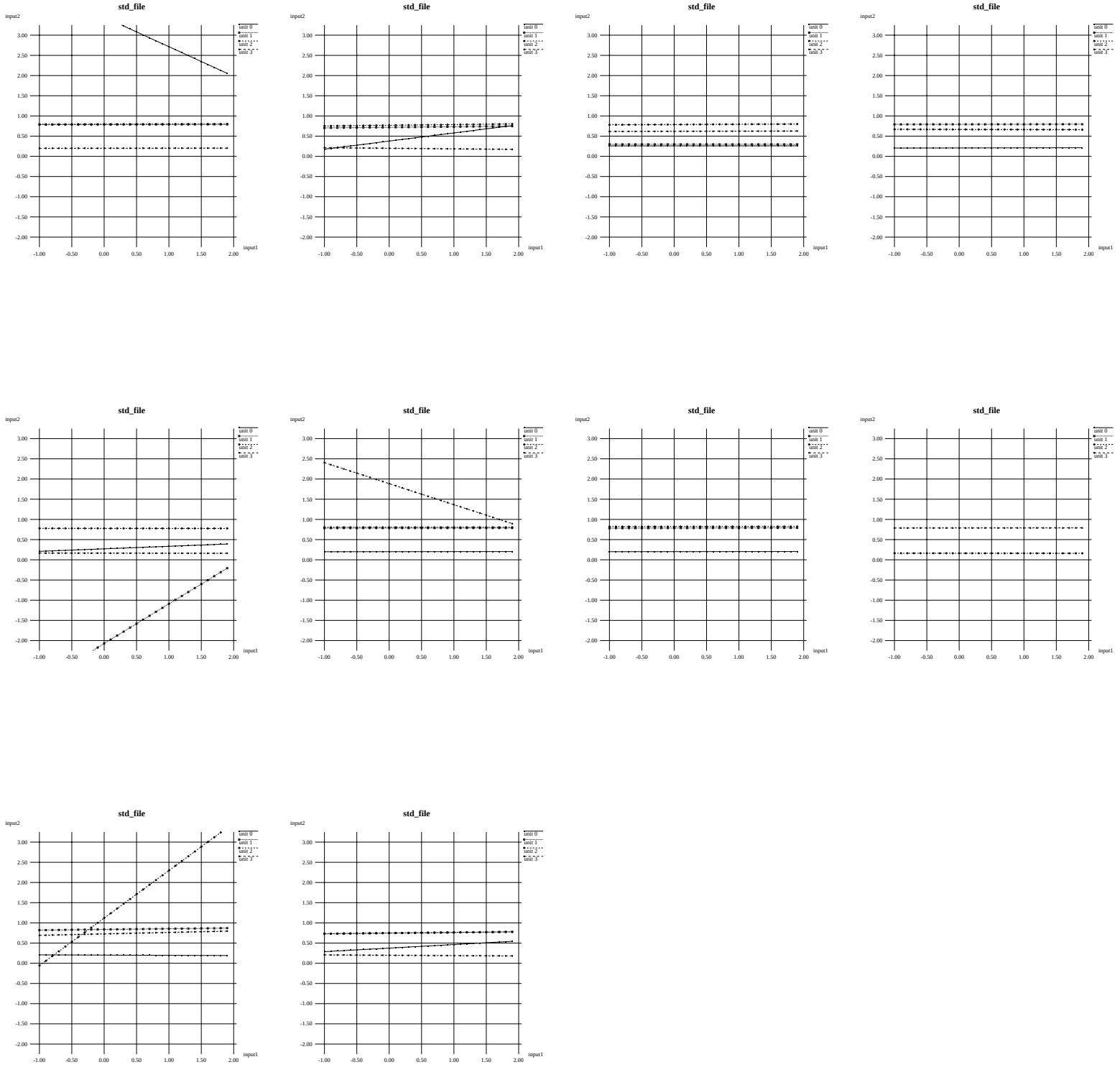

図 6.28: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 4）

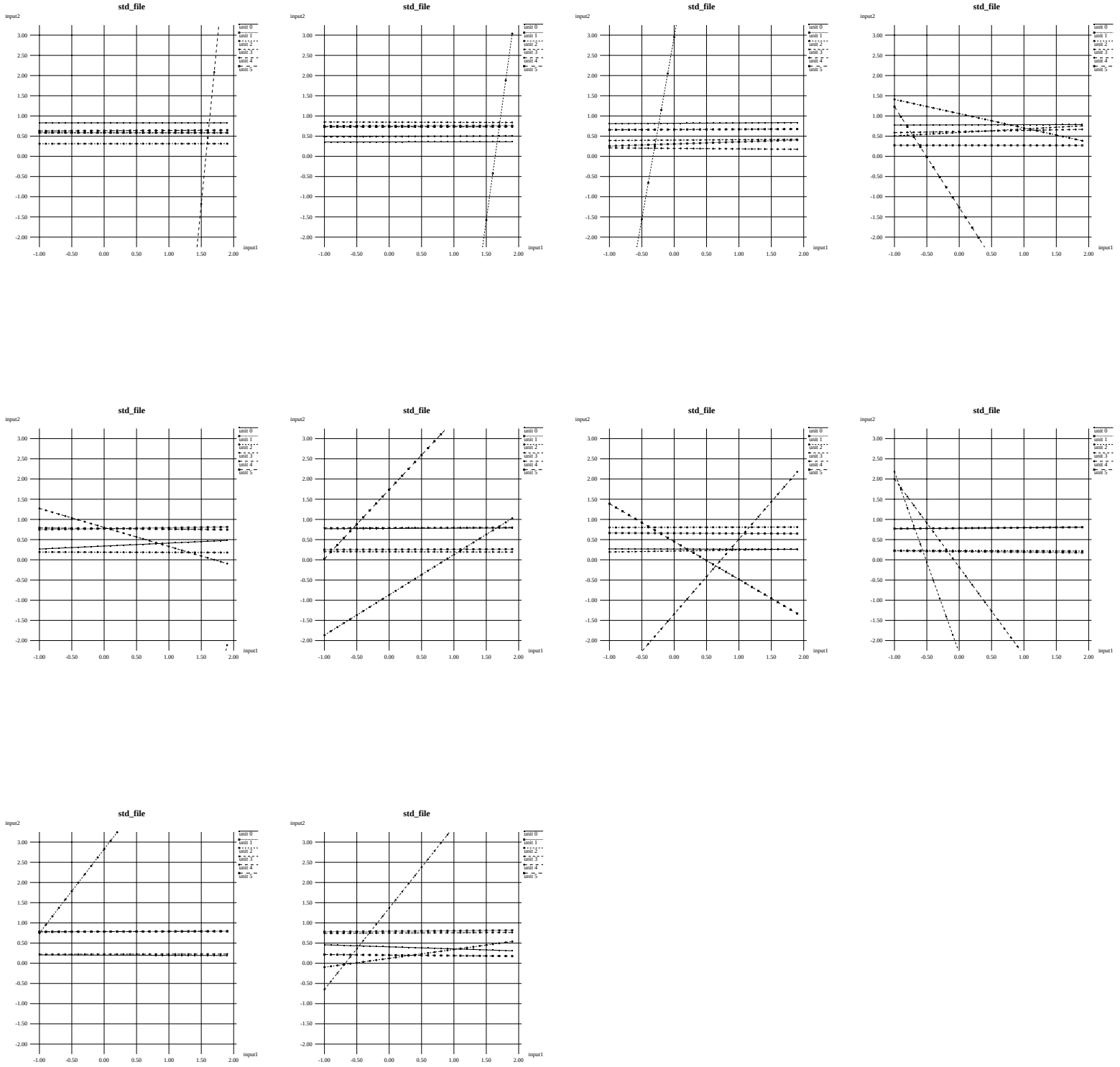

図 6.29: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 6）

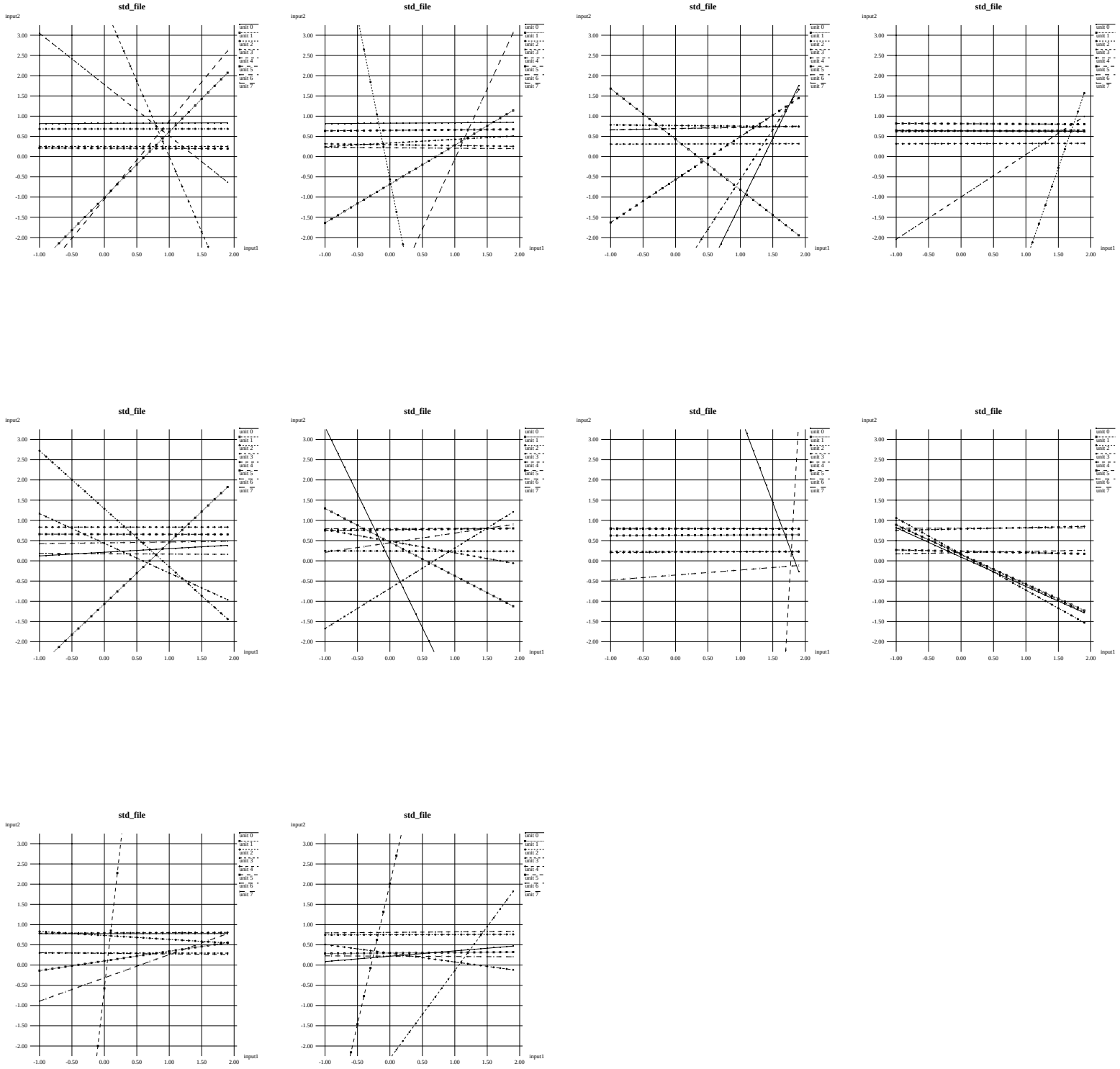

図 6.30: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 8）

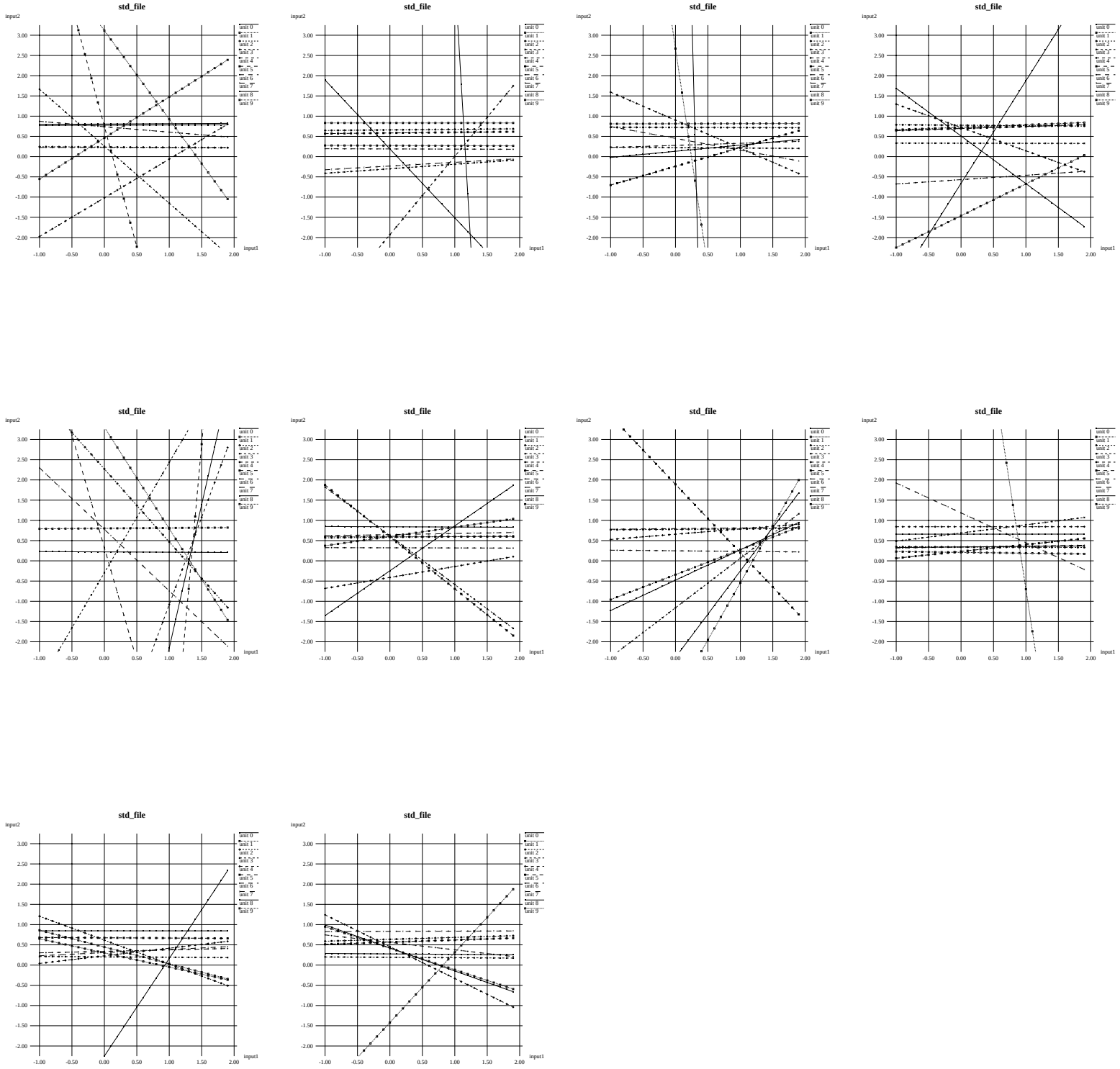

図 6.31: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 10）

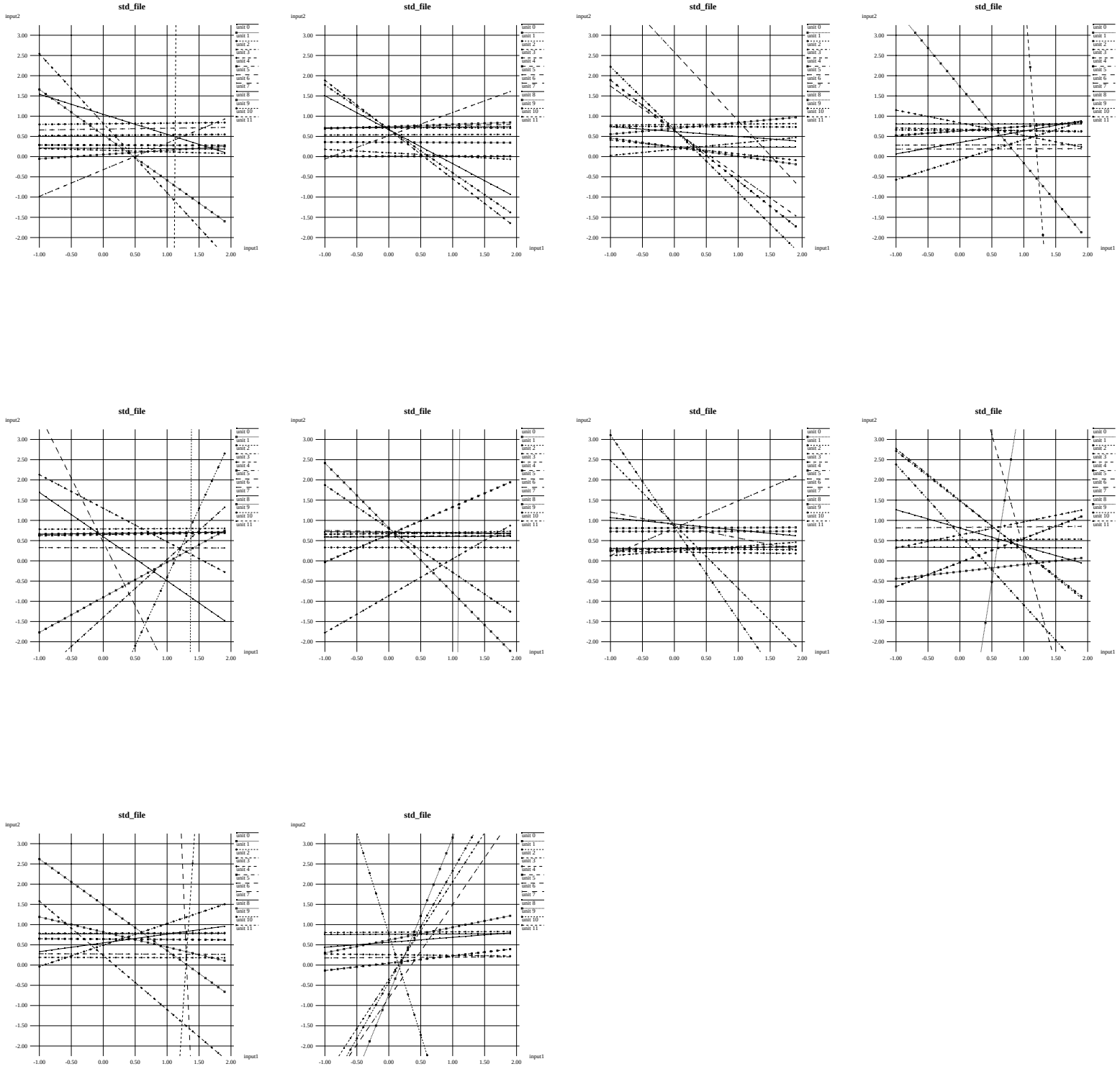

図 6.32: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 12）

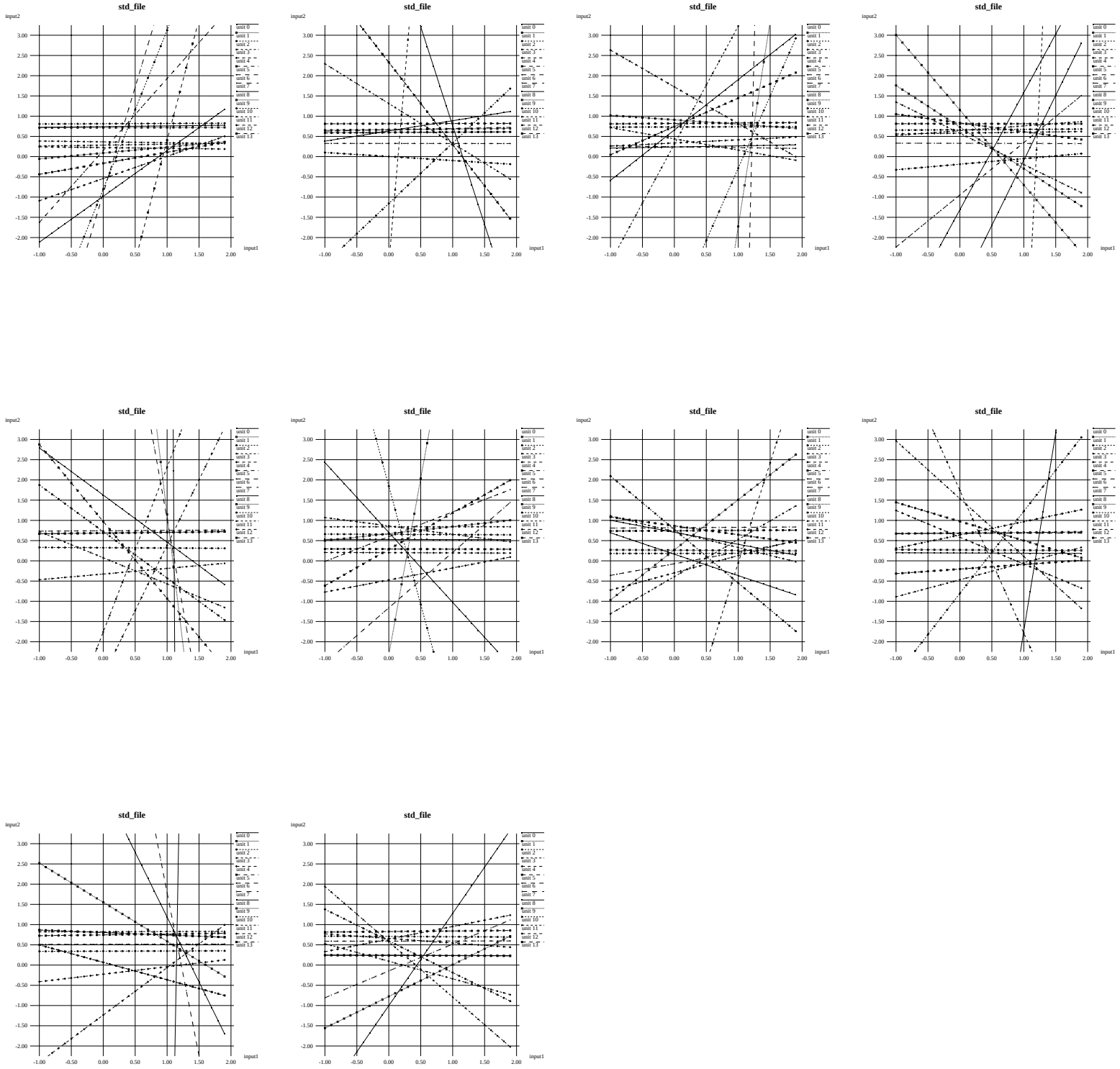

図 6.33: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 14）

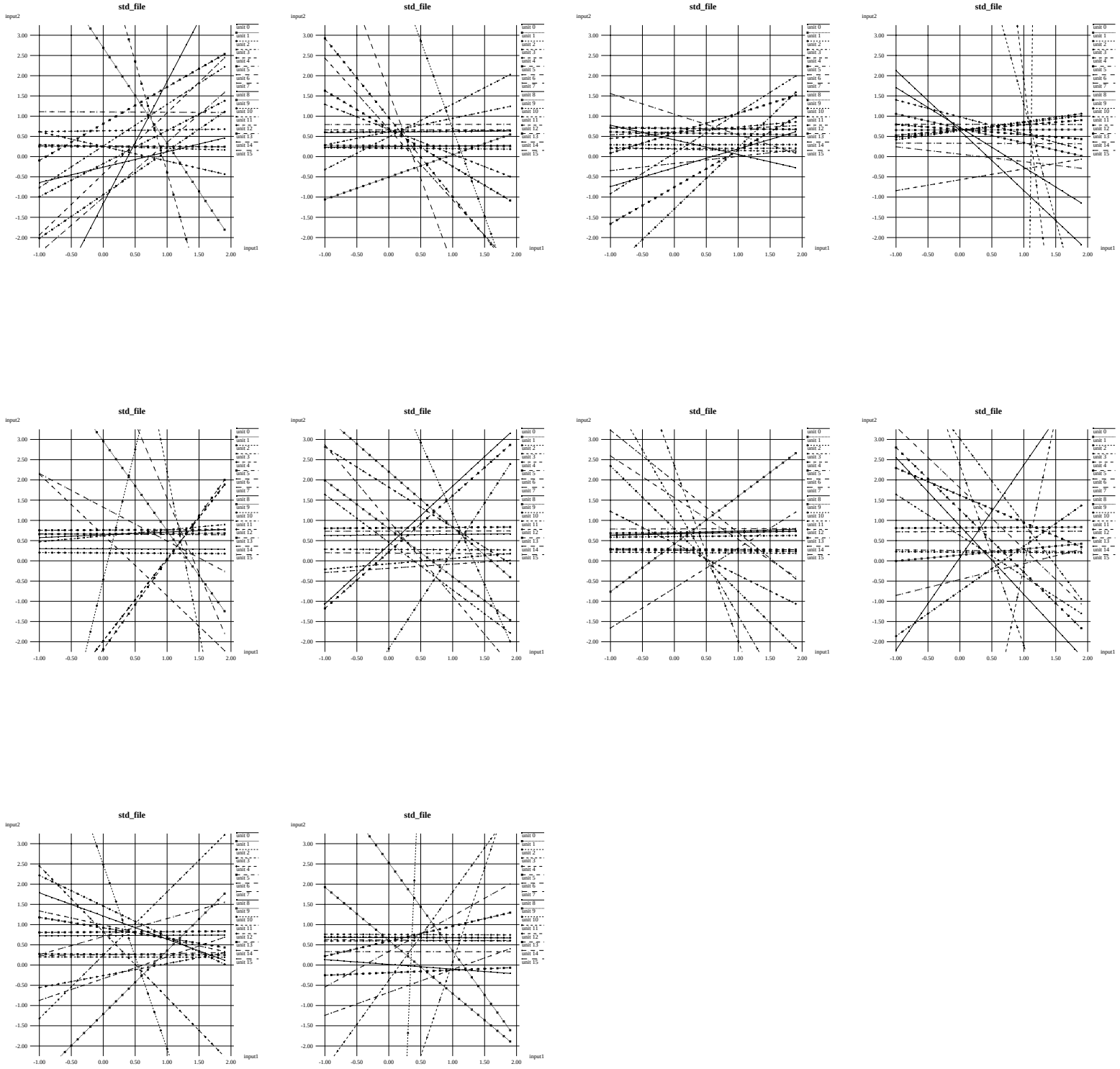

図 6.34: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 16）

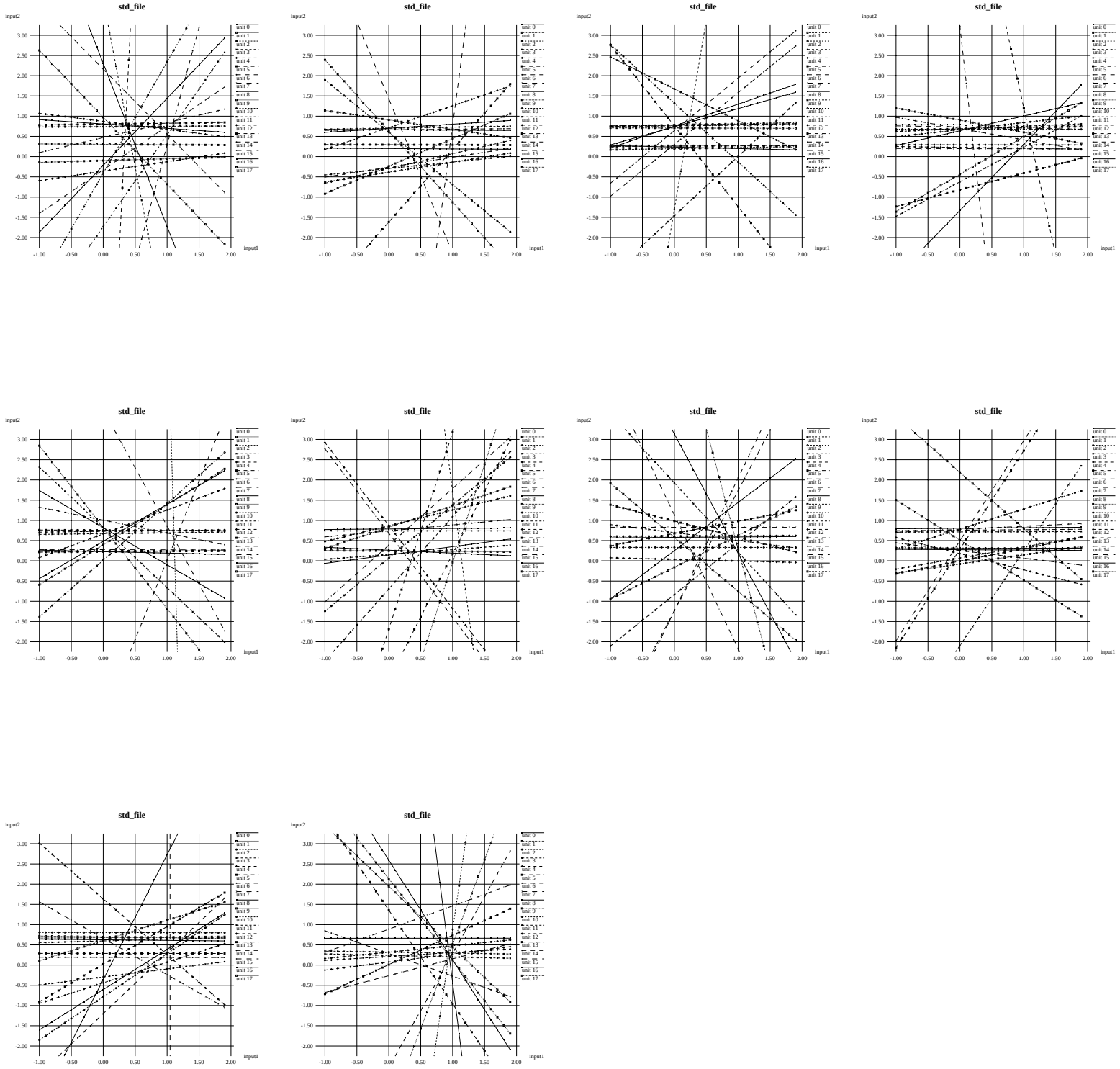

図 6.35: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 18）

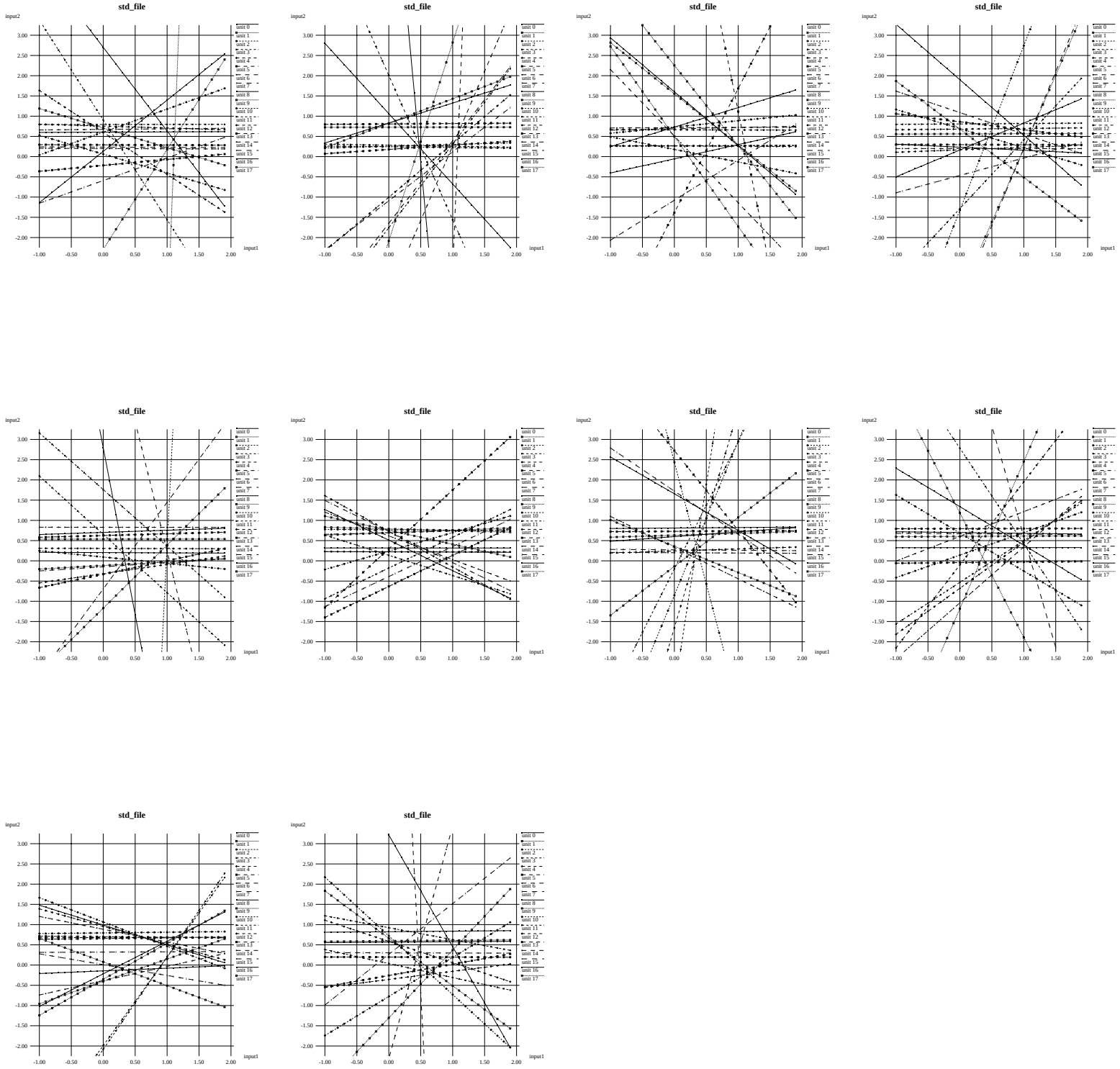

図 6.36: 帯状パターン連続を FTBP で学習した時の分離直線（中間層ノード 20）

付録 D. BP 法の出力分布

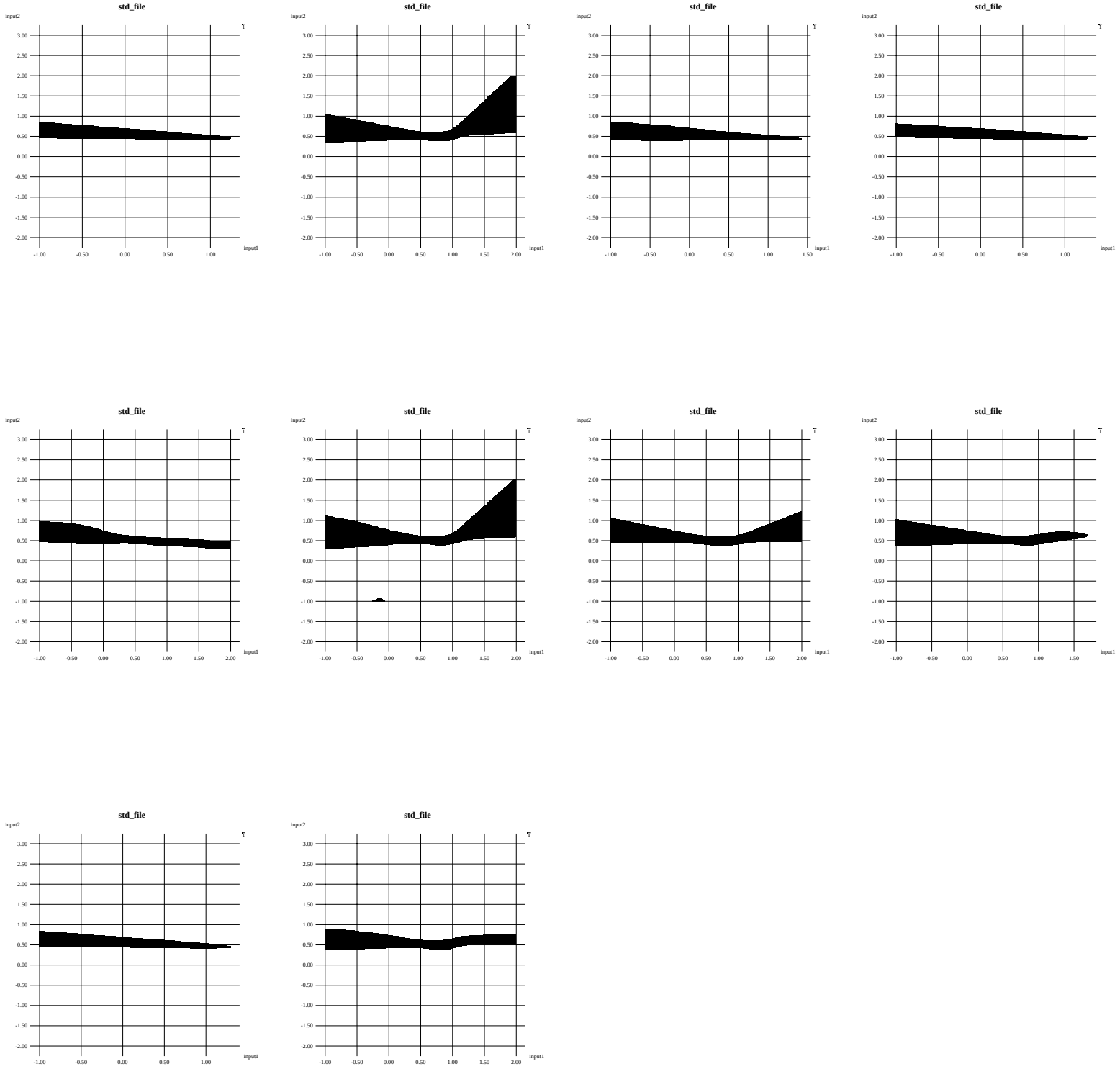

図 6.37: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 4）

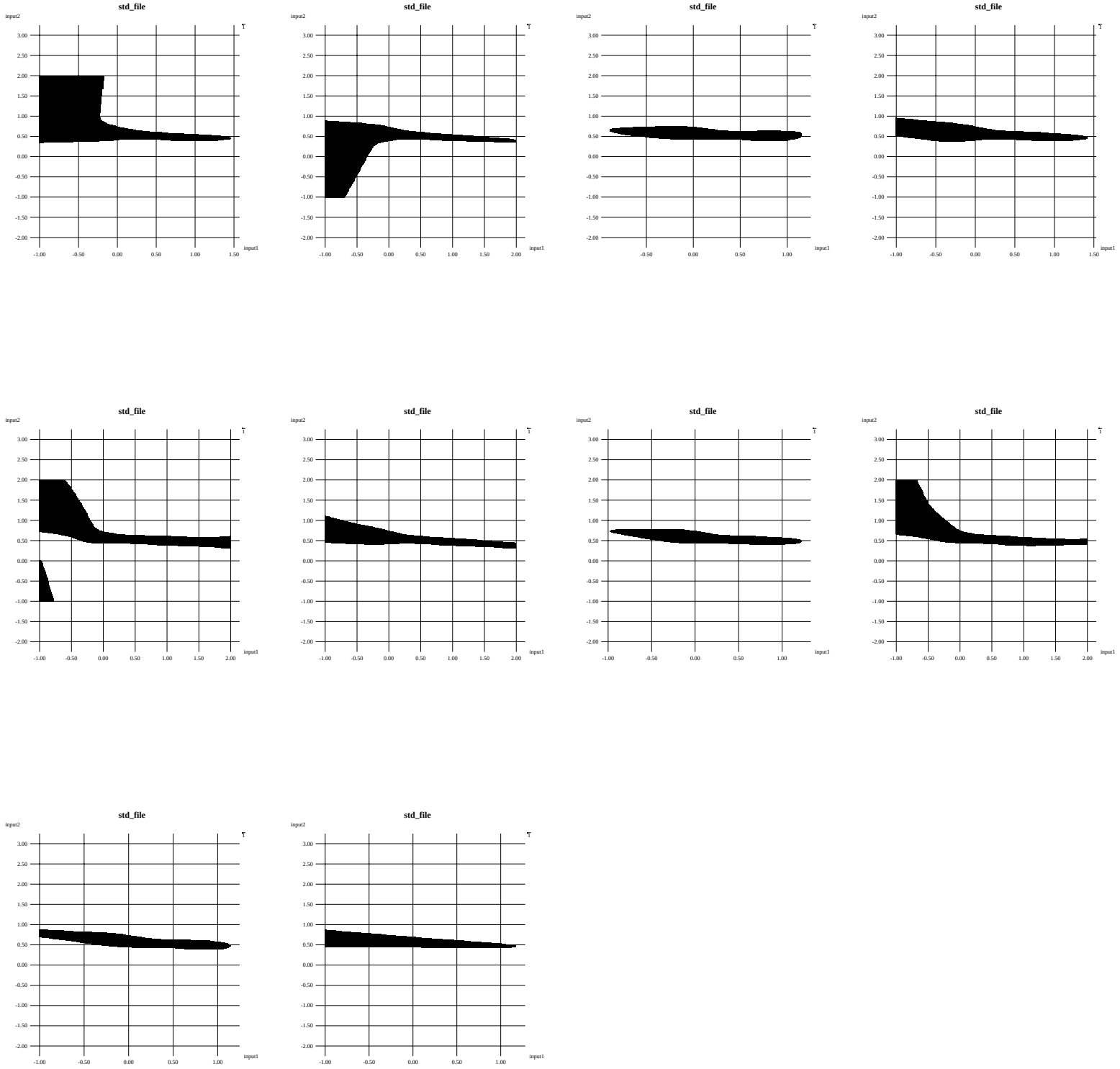

図 6.38: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 6）

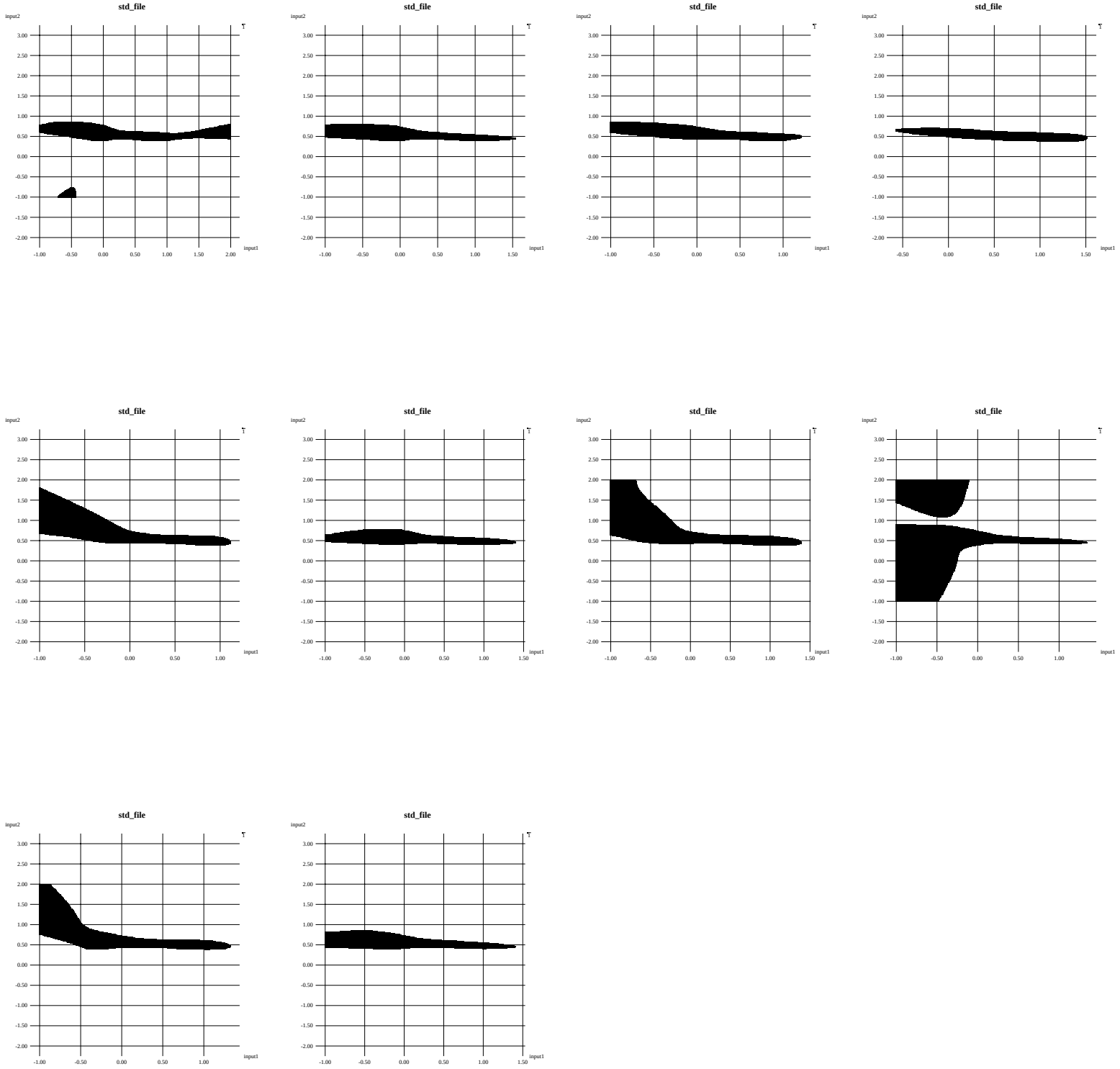

図 6.39: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 8）

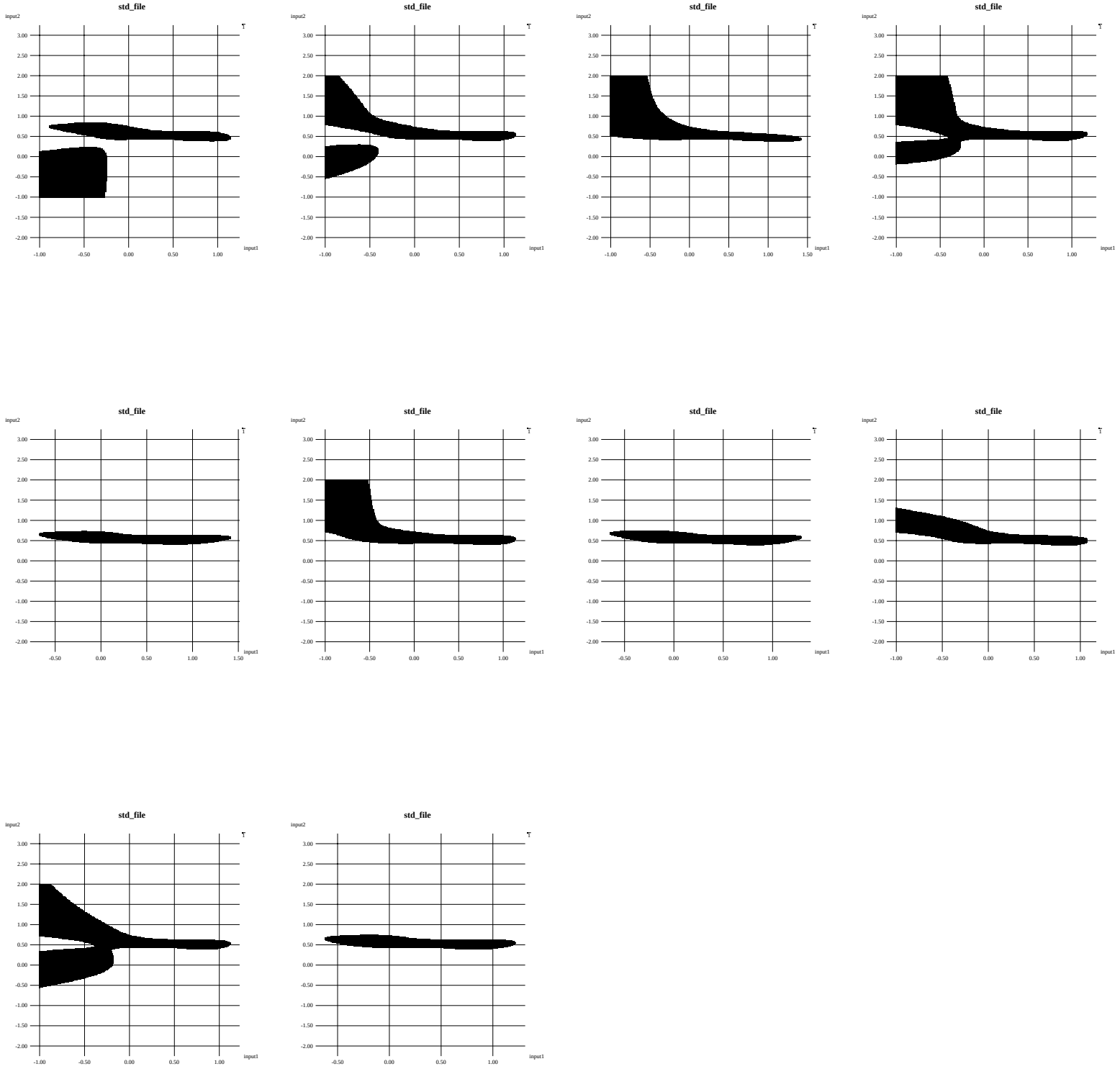

図 6.40: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 10）

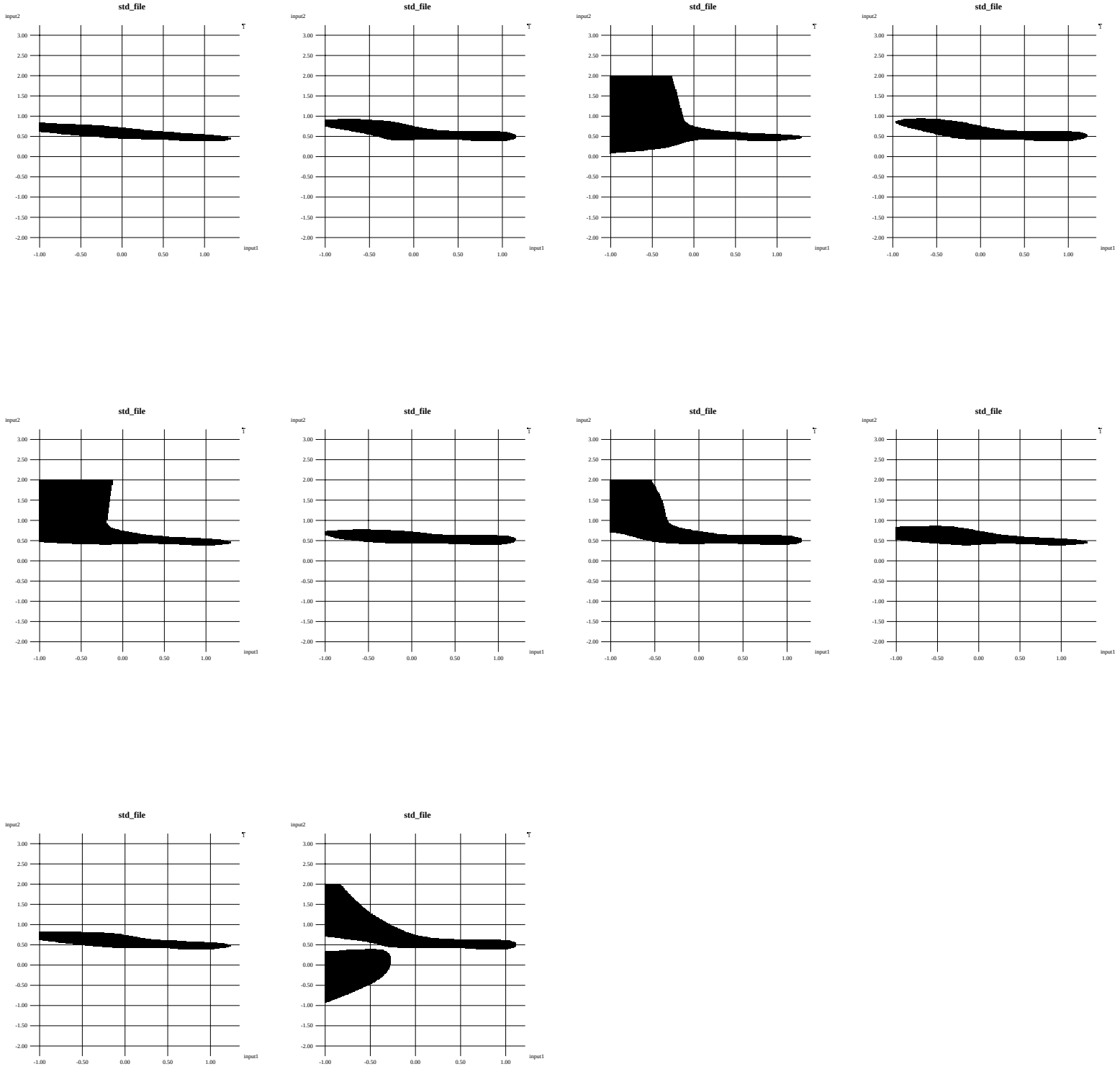

図 6.41: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 12）

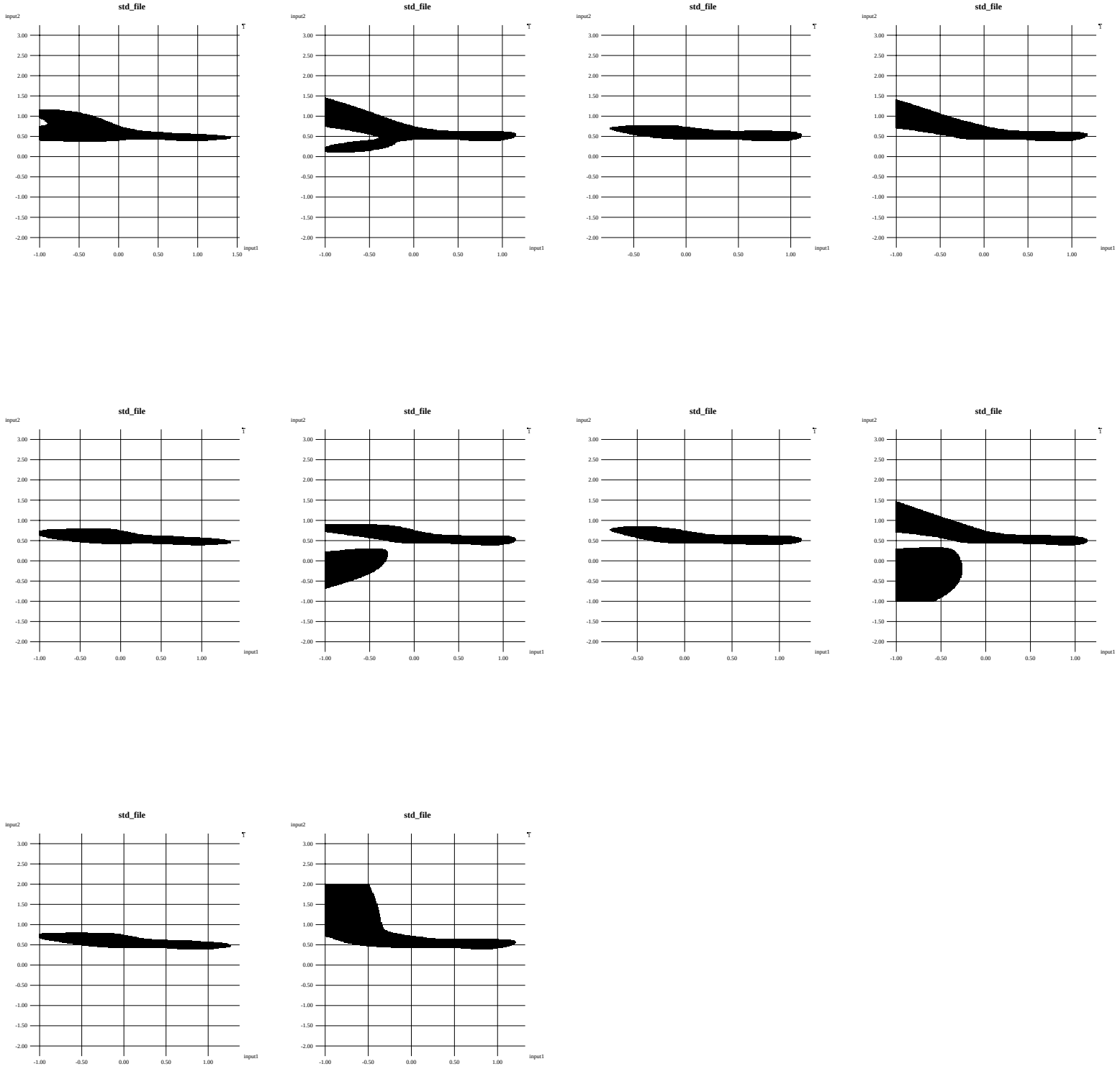

図 6.42: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 14）

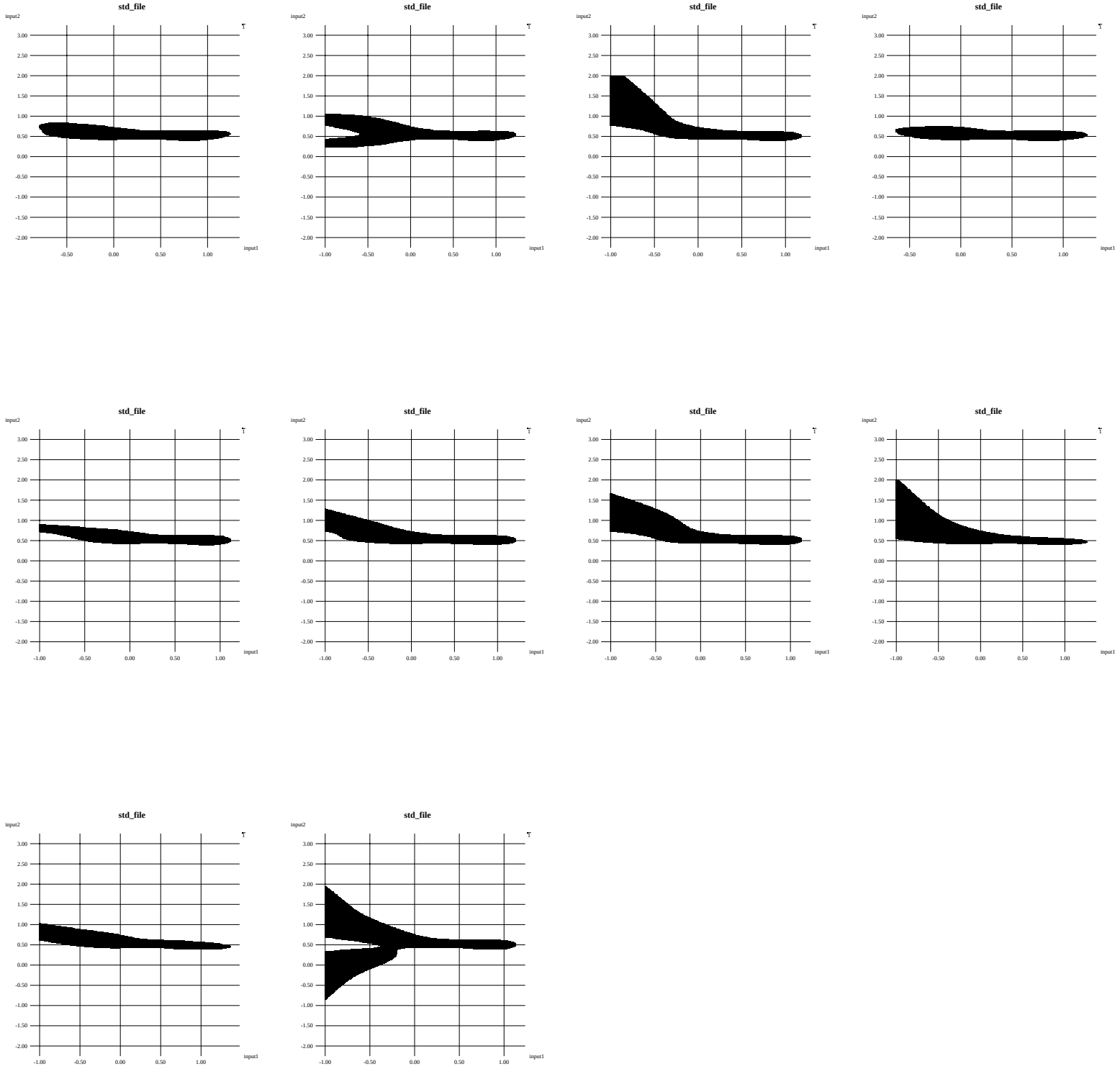

図 6.43: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 16）

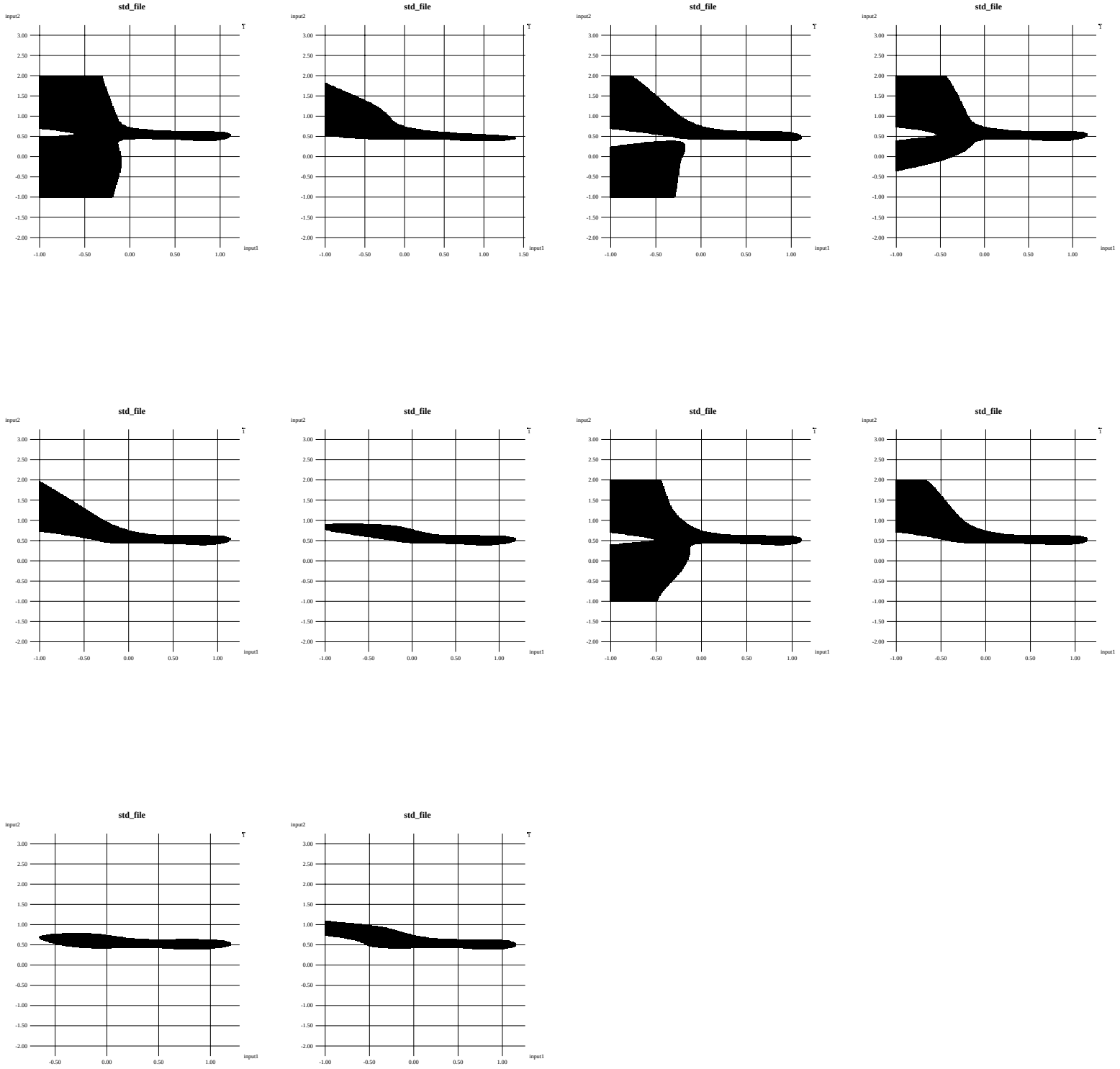

図 6.44: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 18）

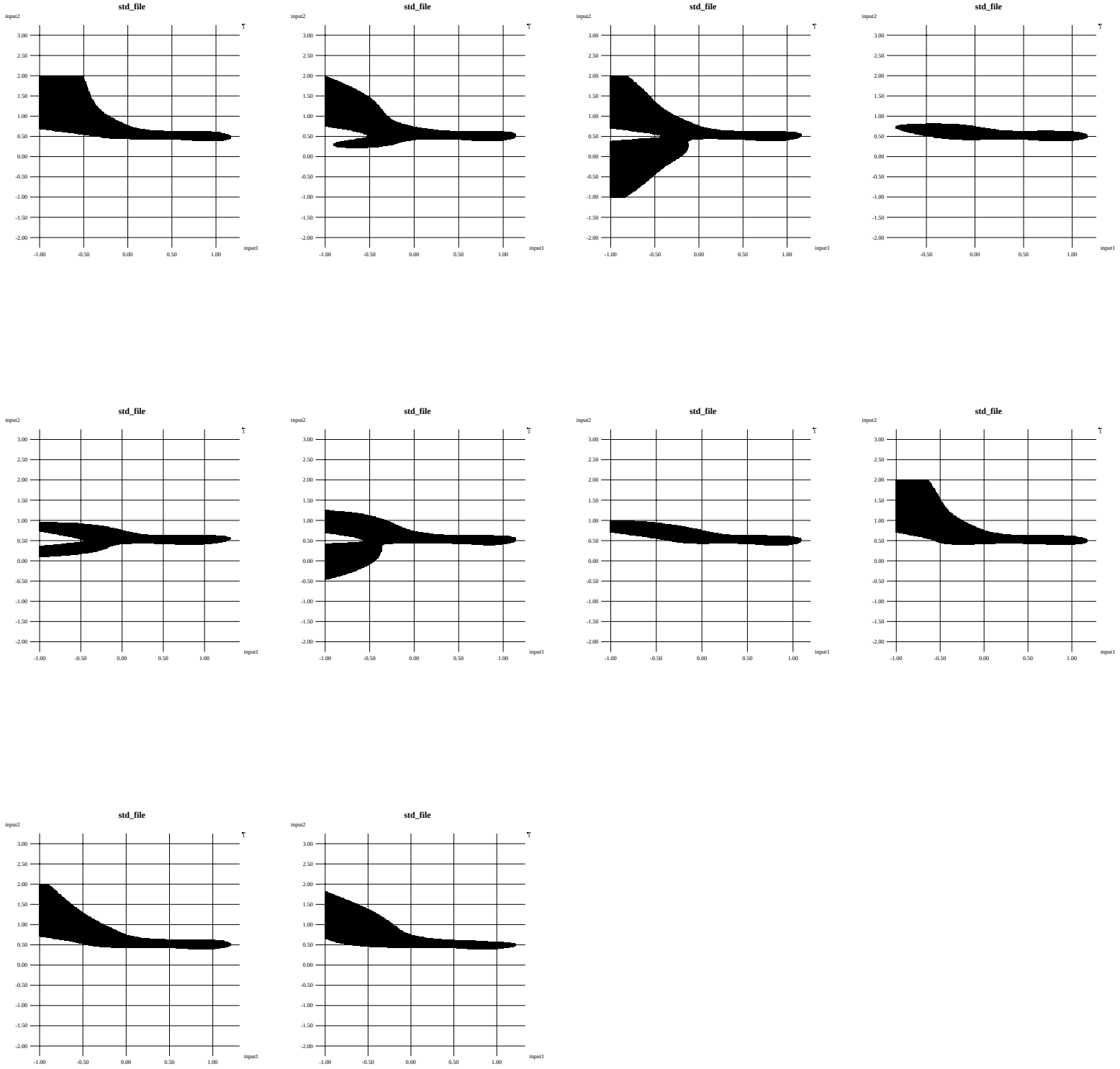

図 6.45: 帯状パターン離散を BP で学習した時の出力分布（中間層ノード 20）

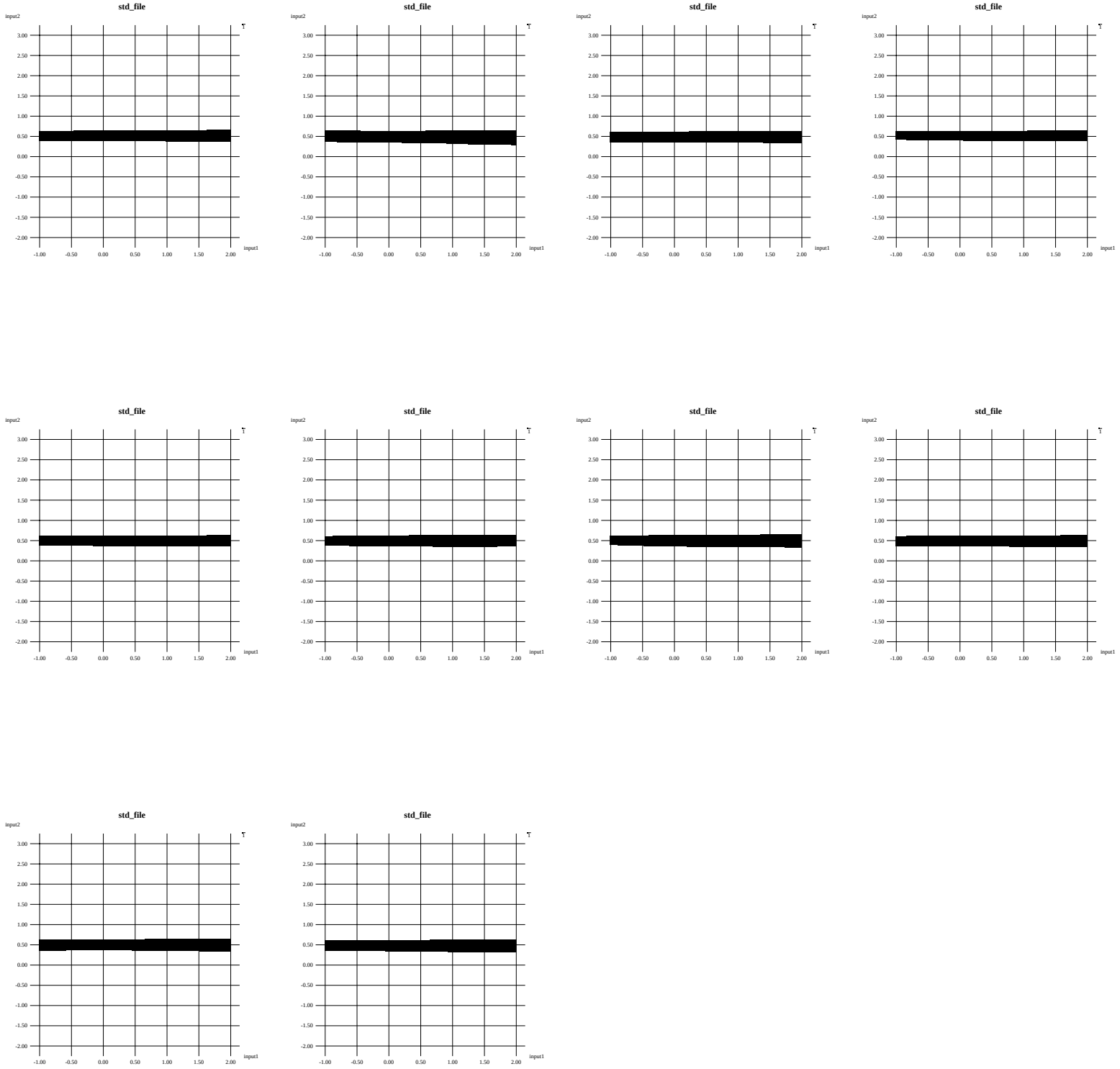

図 6.46: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 4）

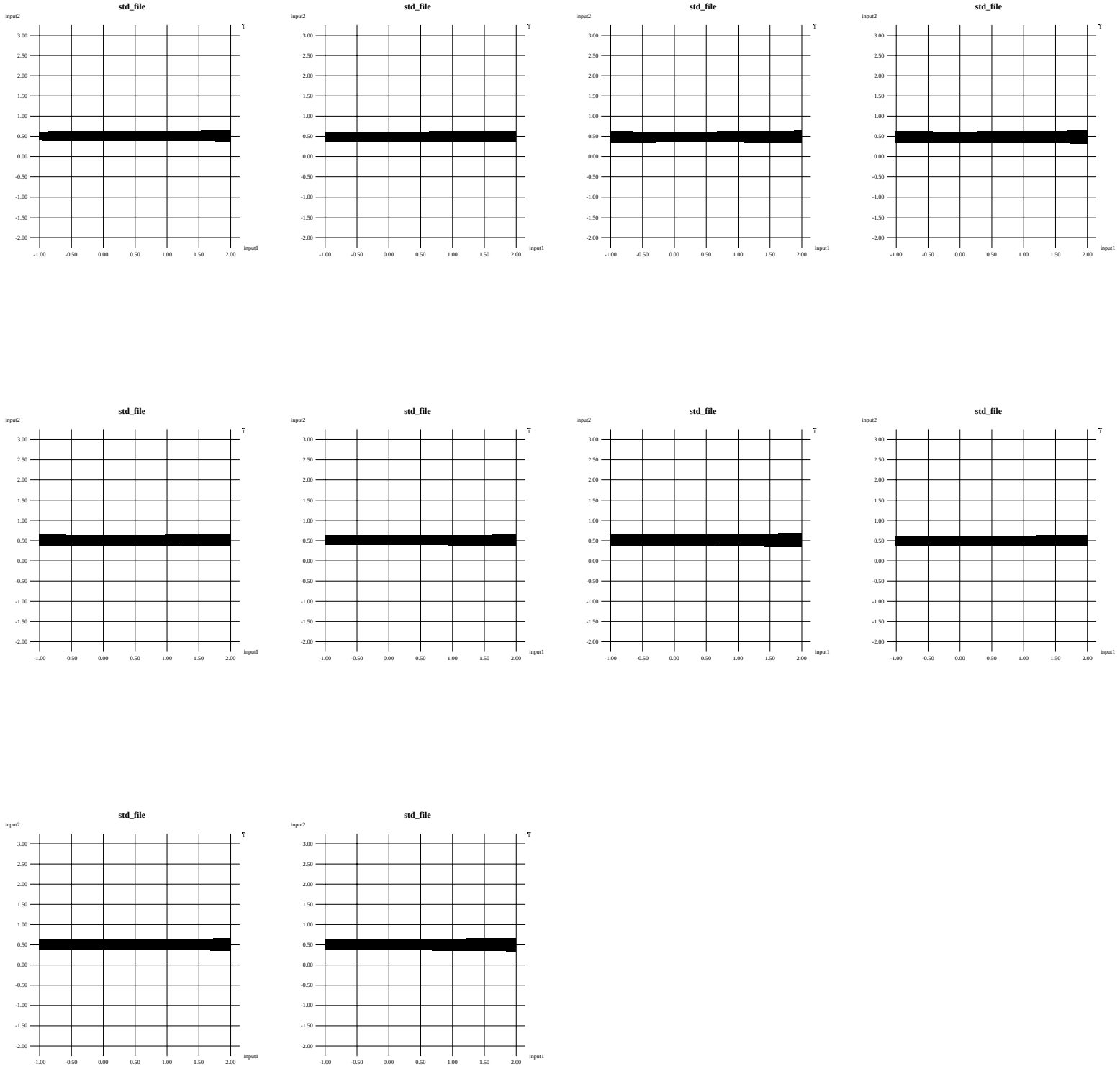

図 6.47: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 6）

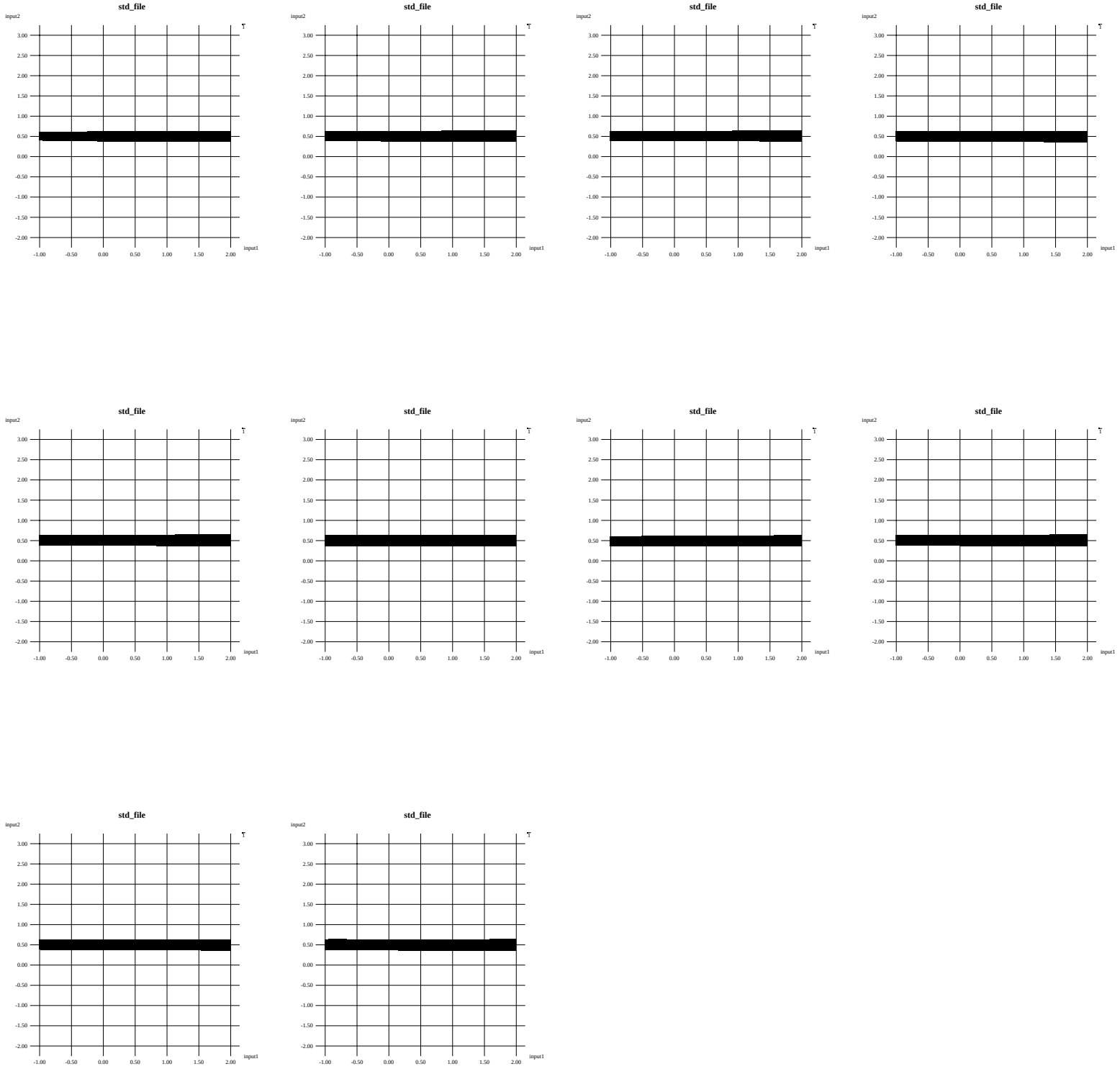

図 6.48: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 8）

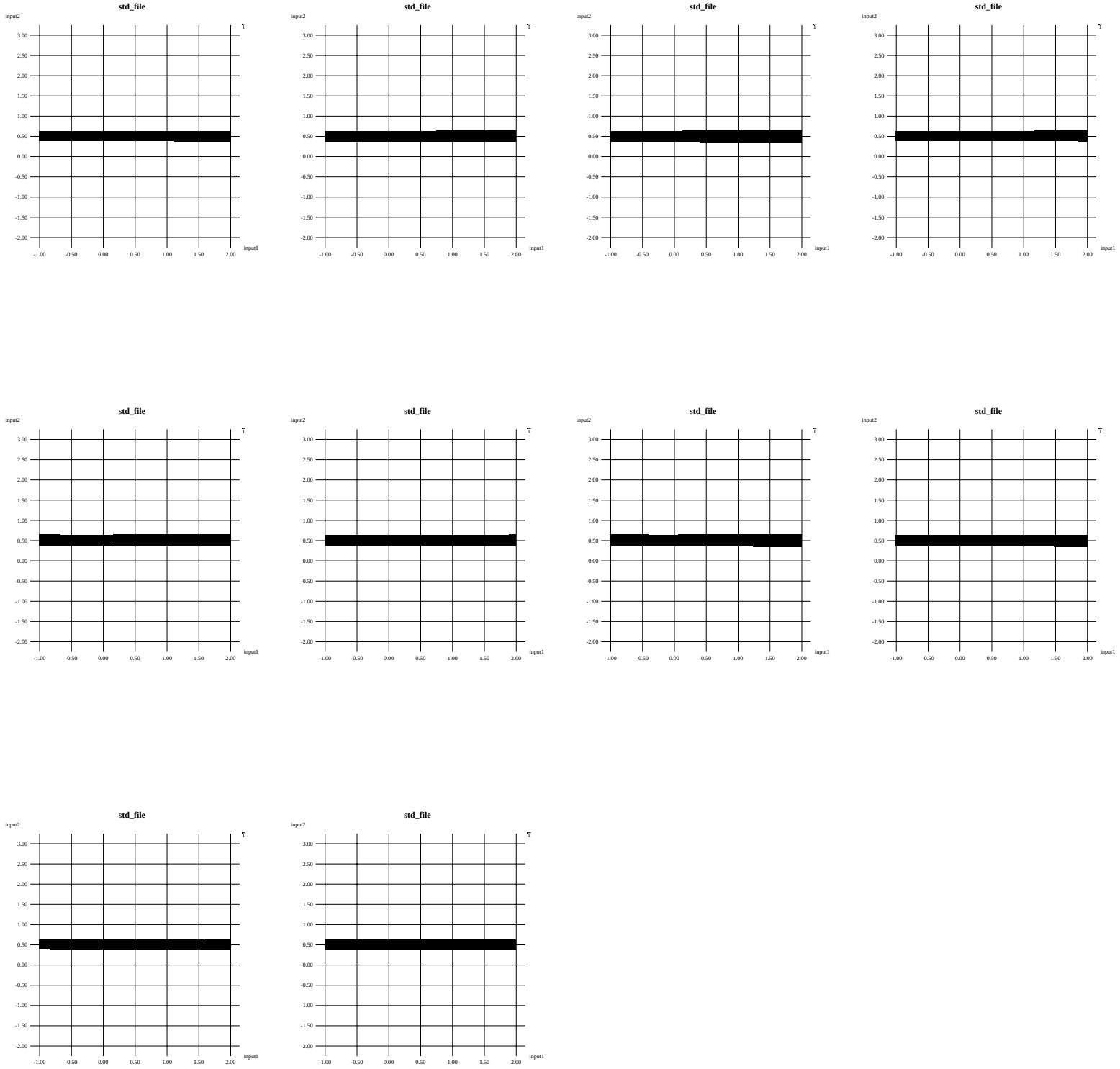

図 6.49: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 10）

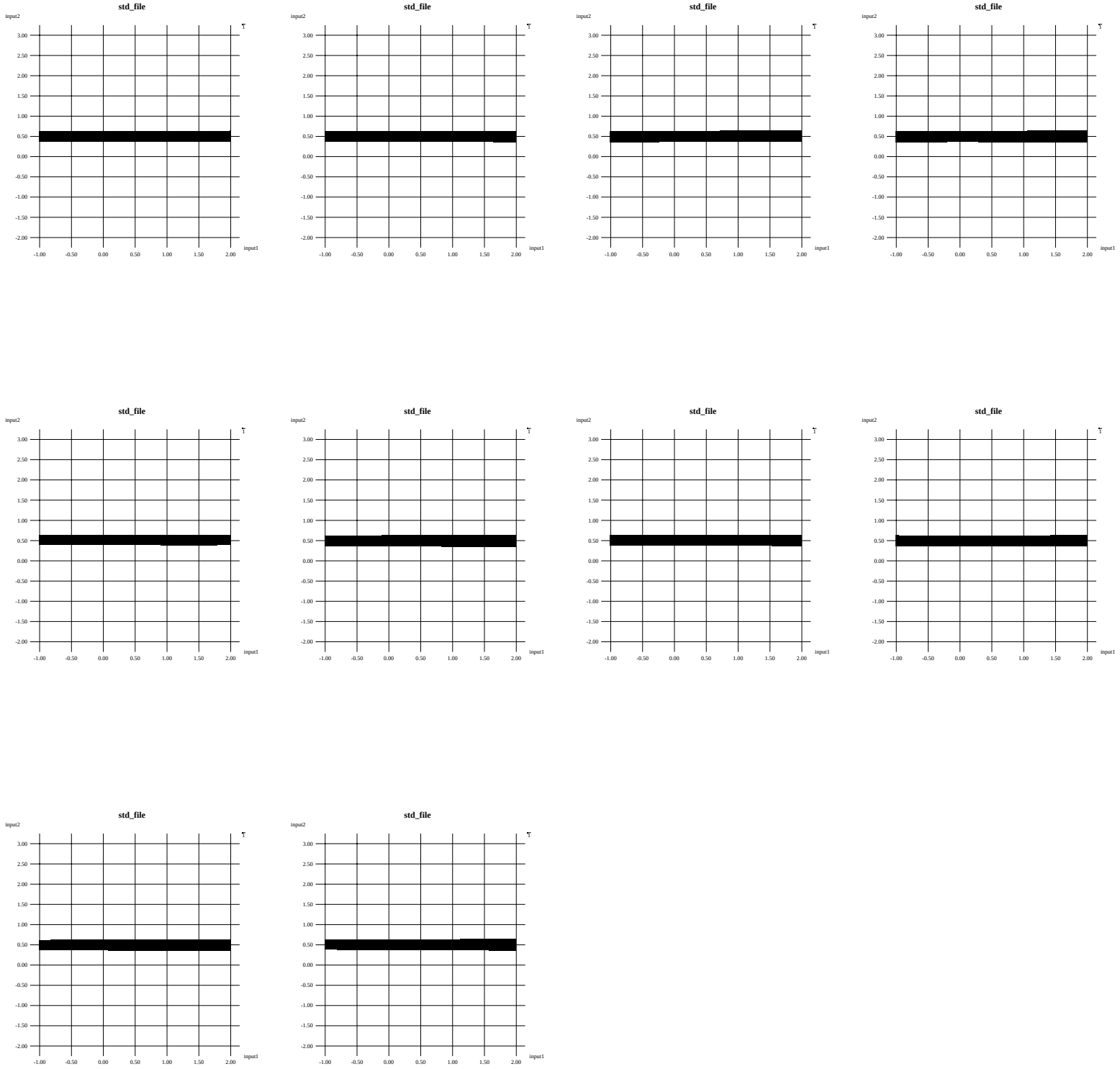

図 6.50: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 12）

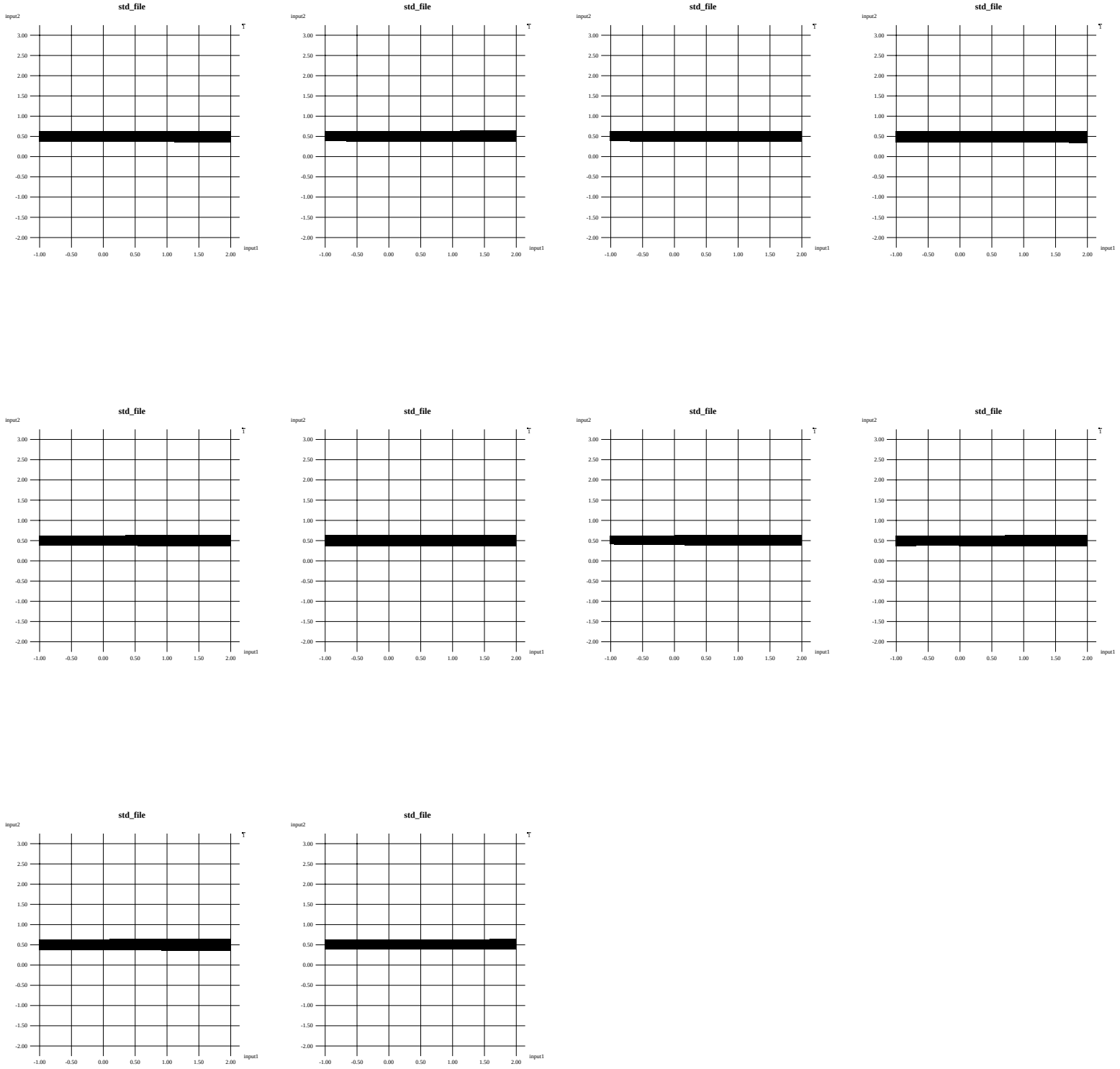

図 6.51: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 14）

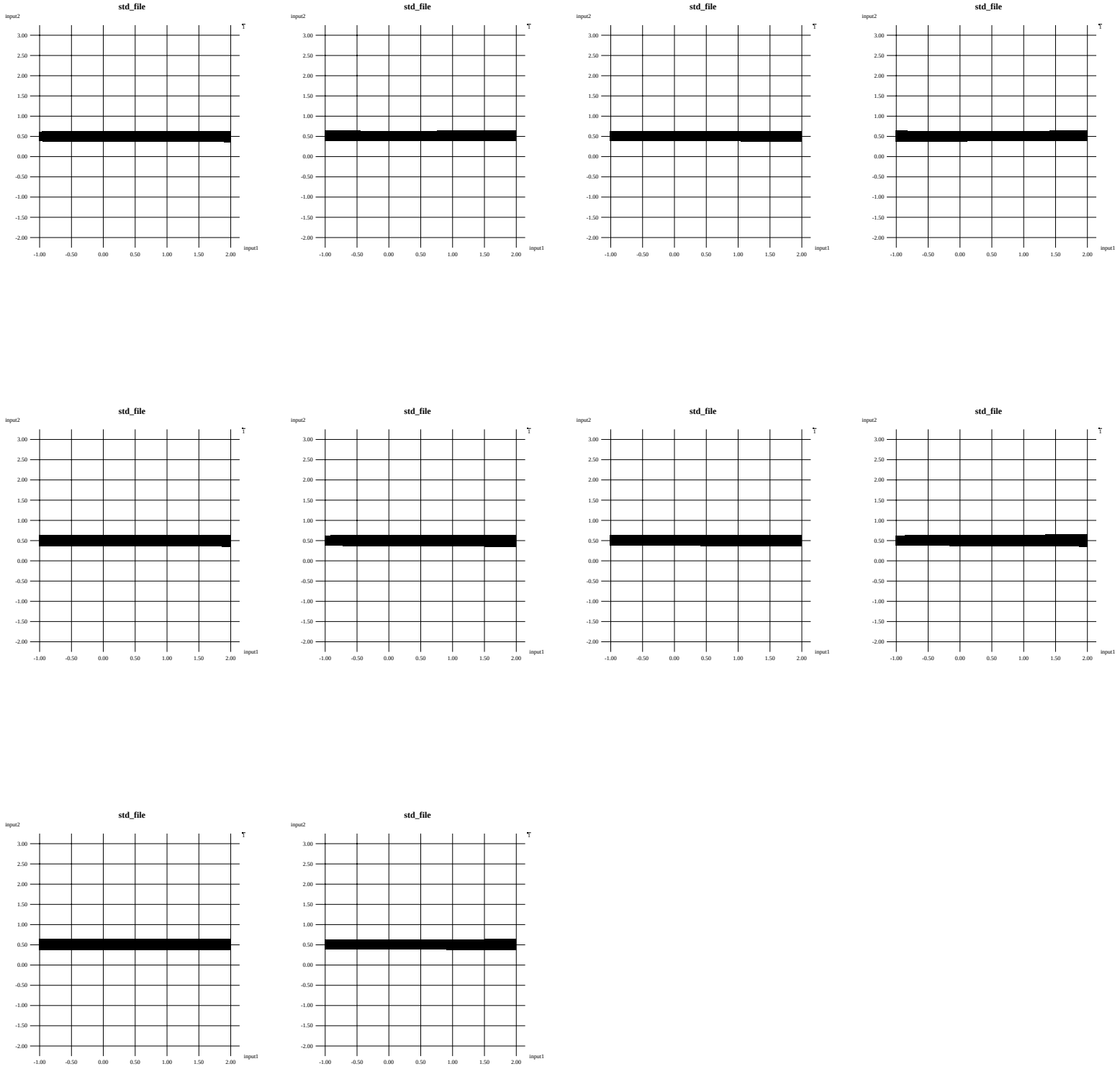

図 6.52: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 16）

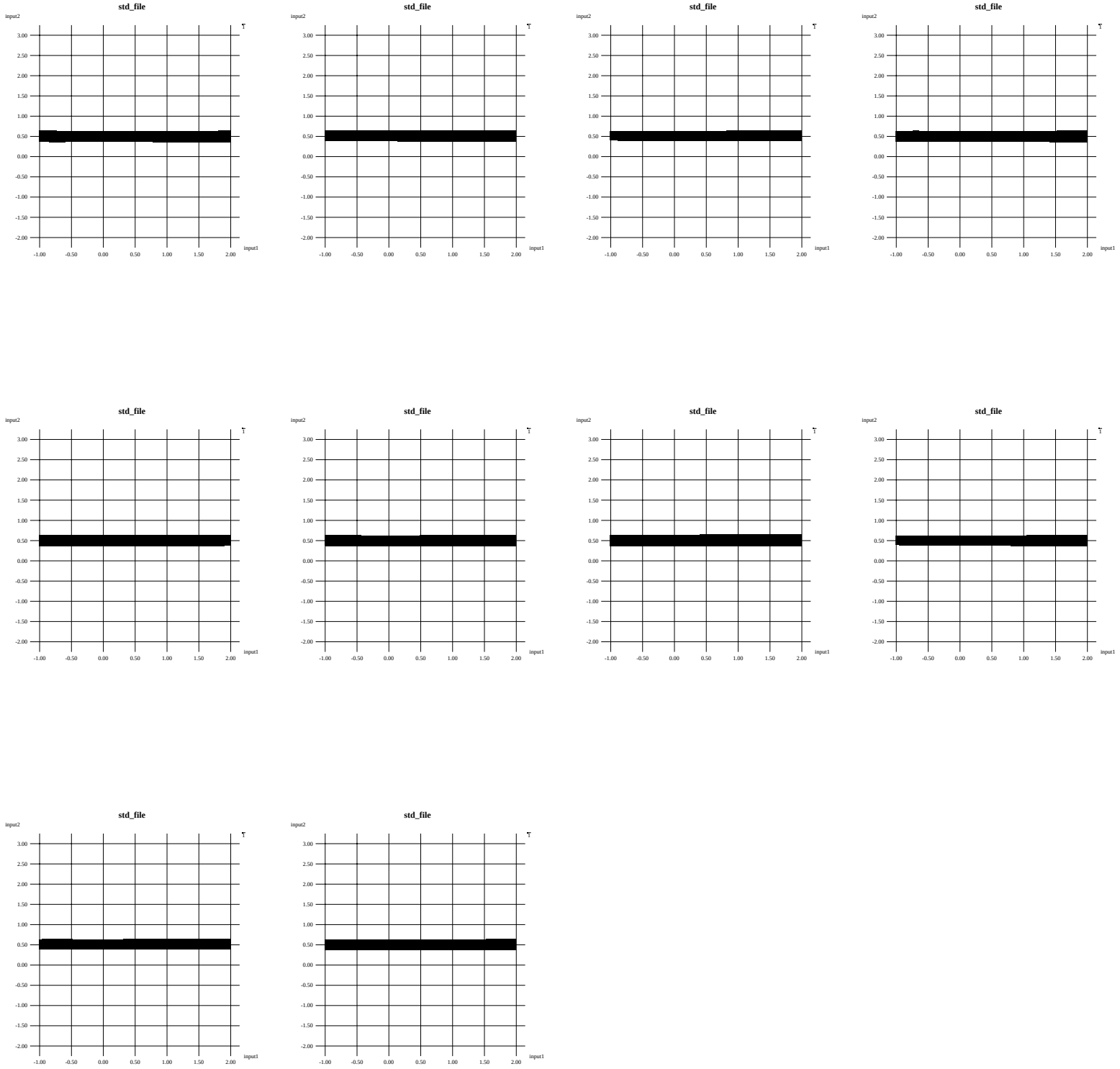

図 6.53: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 18）

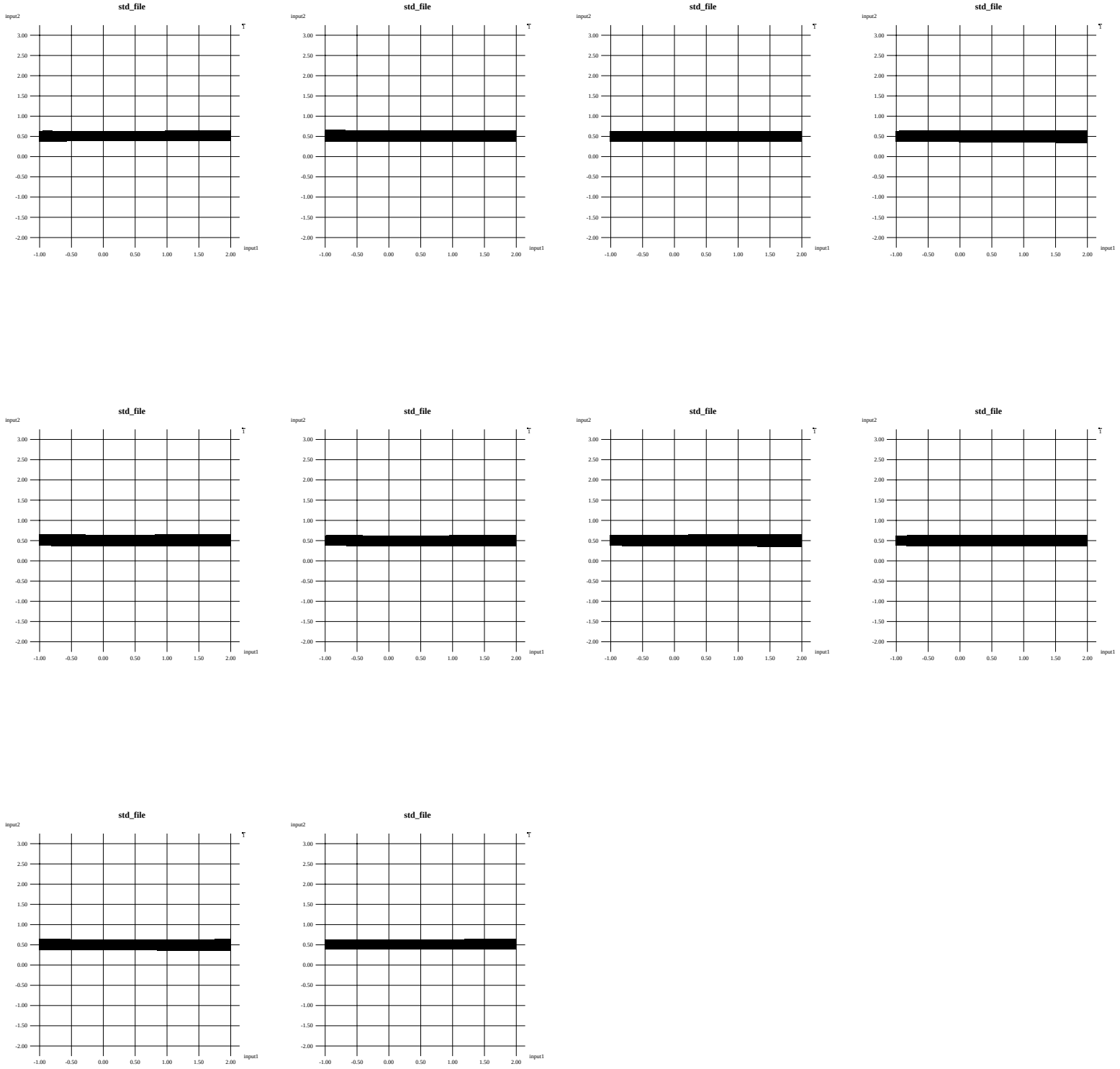

図 6.54: 帯状パターン連続を BP で学習した時の出力分布（中間層ノード 20）

付録 E. FTBP 法の出力分布

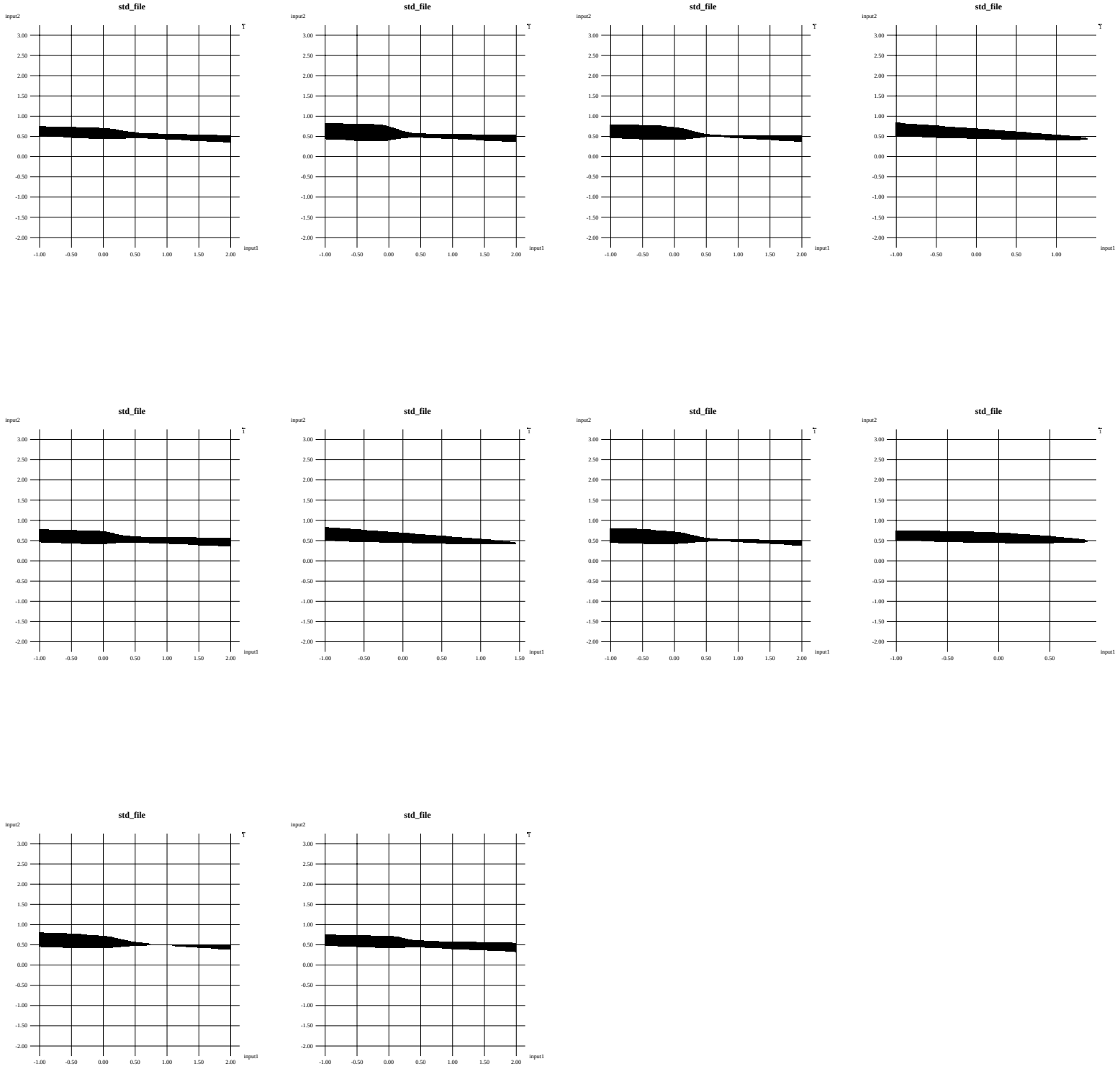

図 6.55: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 4）

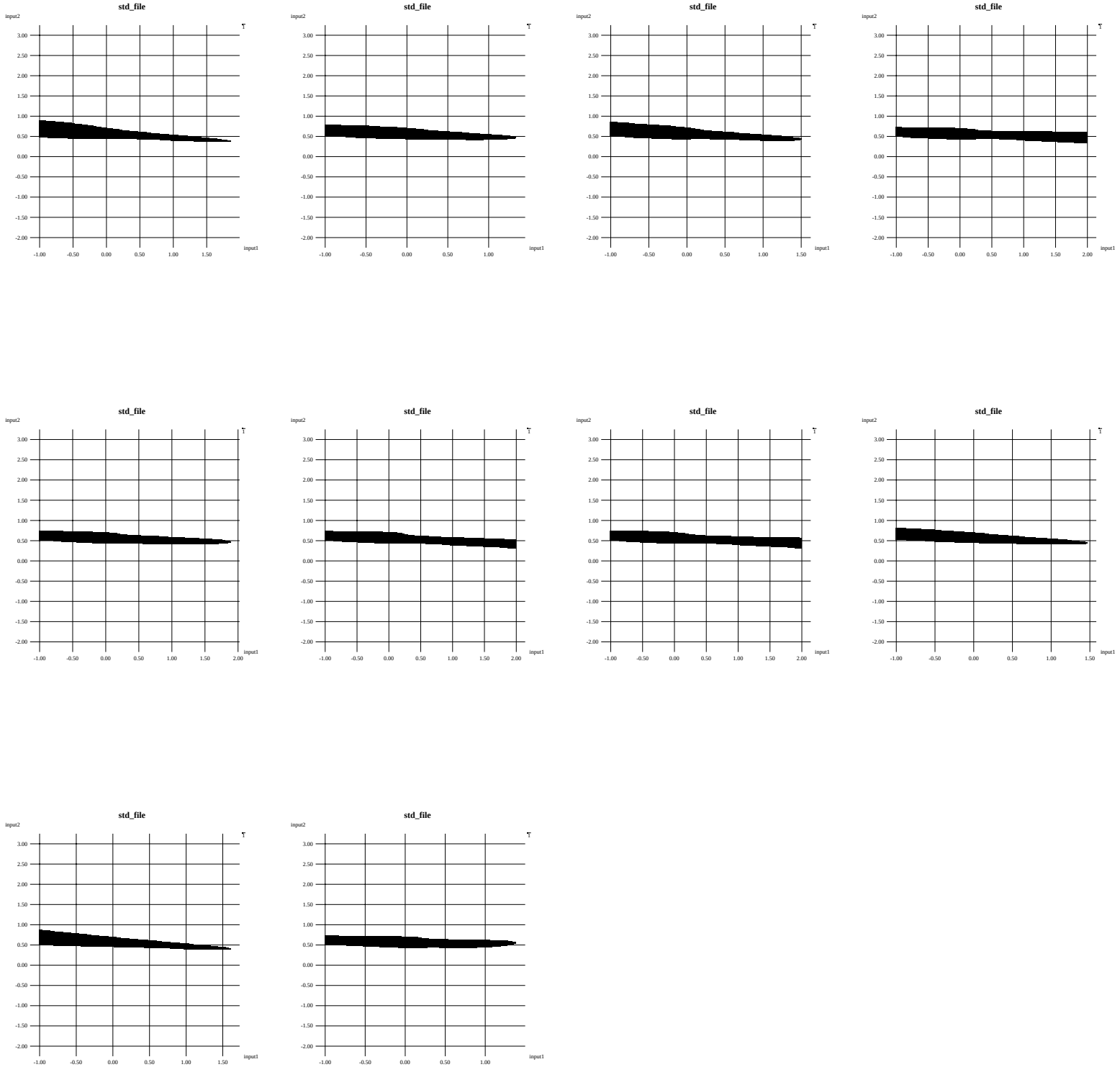

図 6.56: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 6）

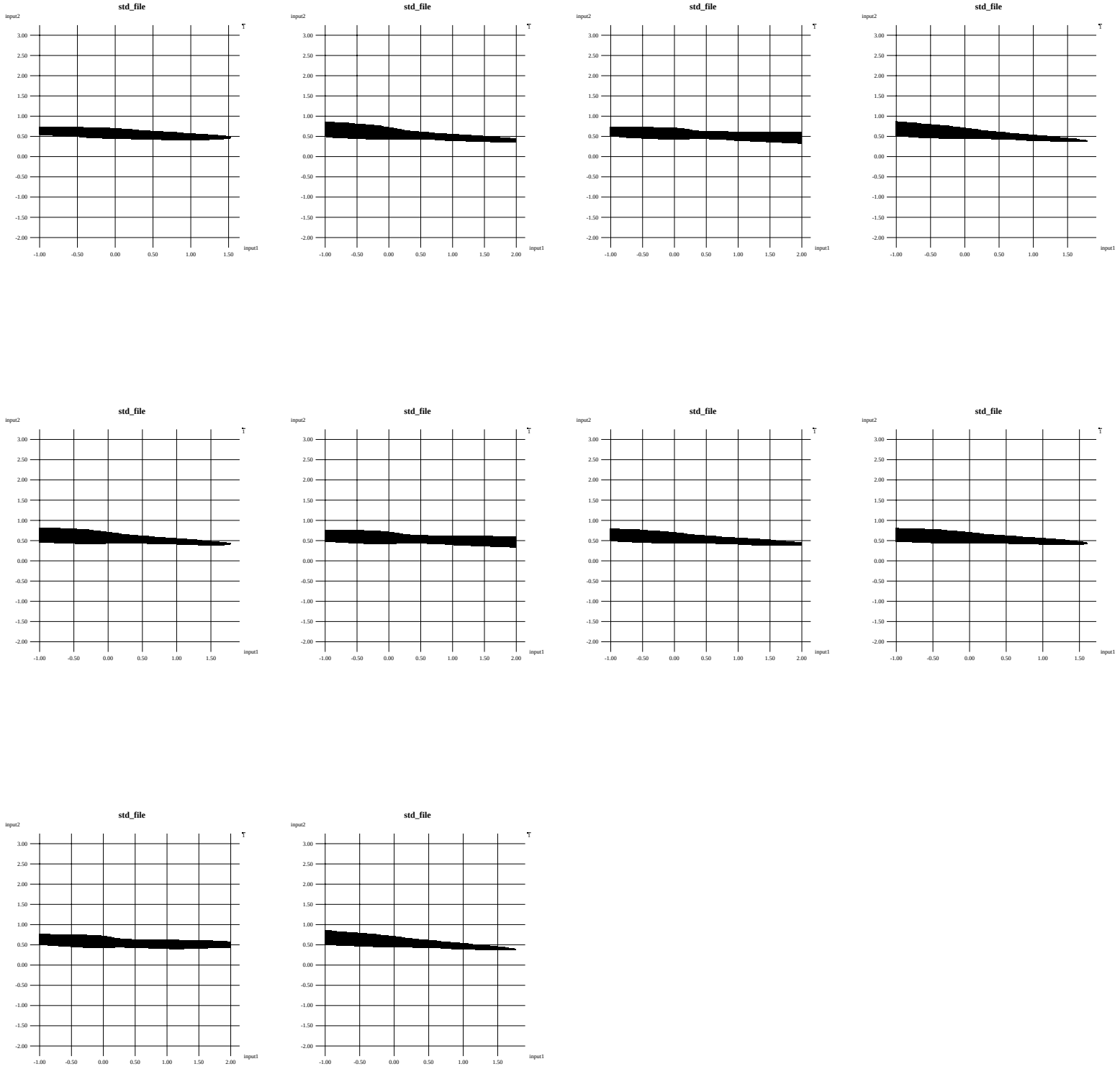

図 6.57: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 8）

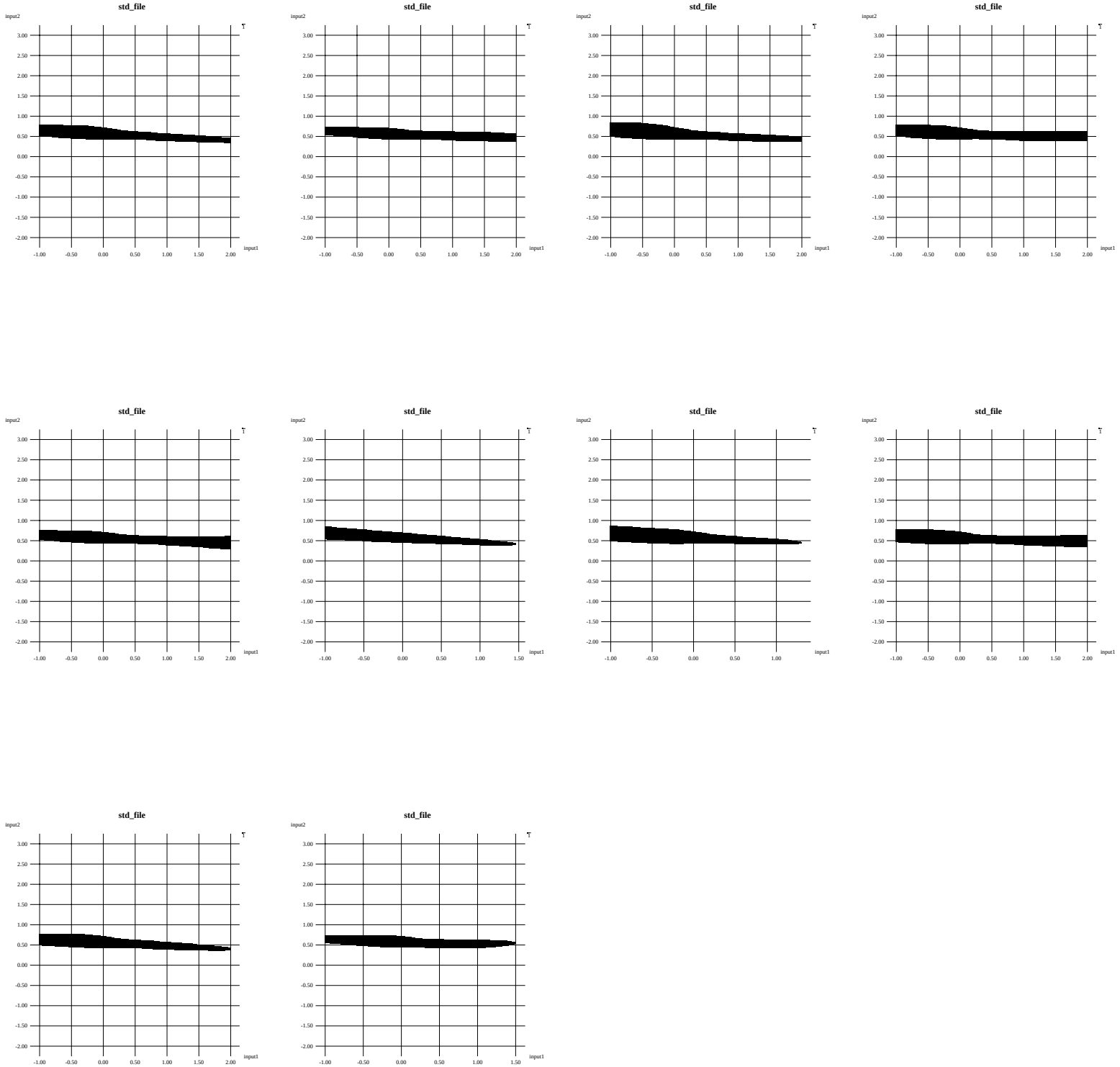

図 6.58: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 10）

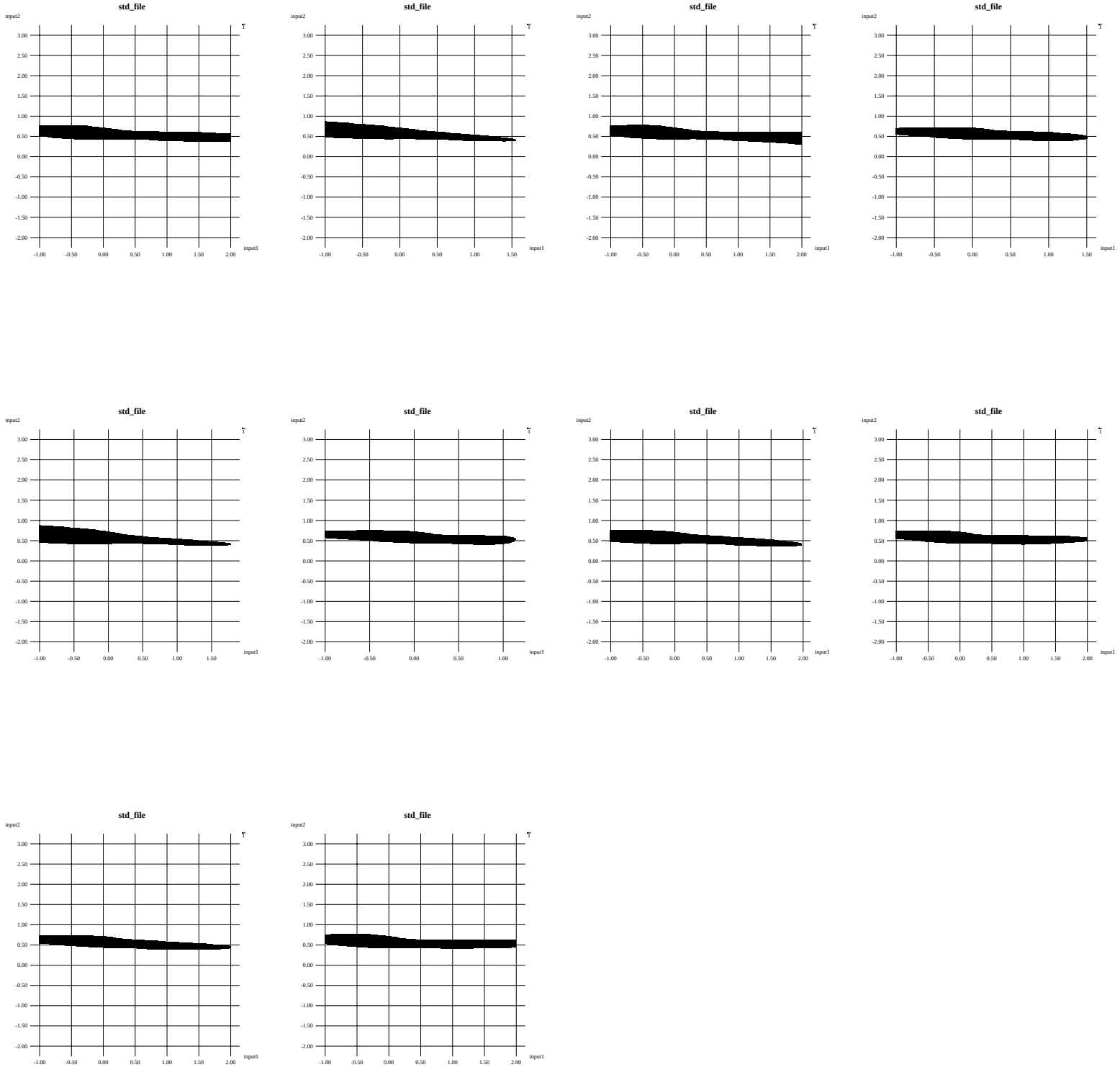

図 6.59: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 12）

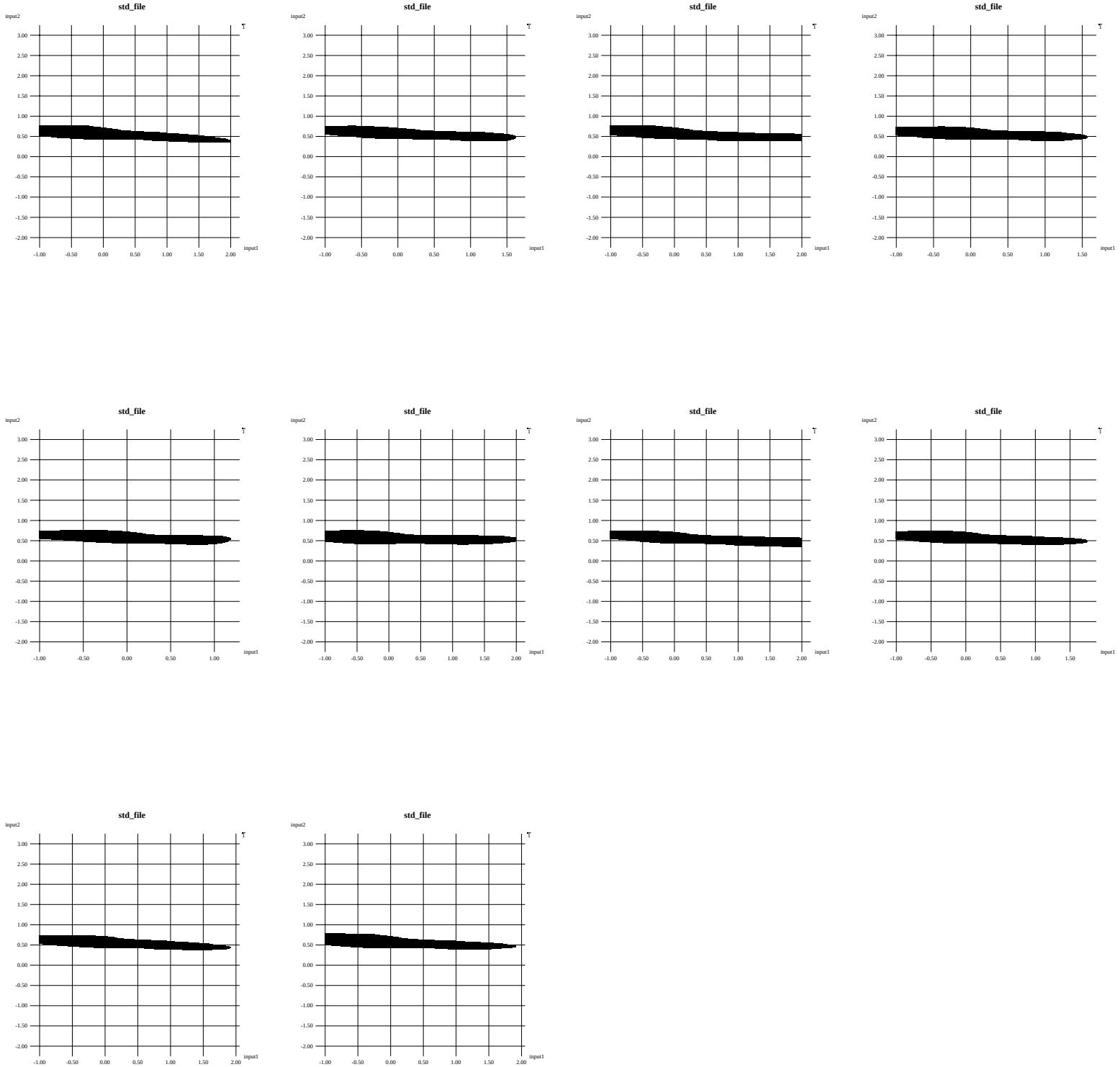

図 6.60: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 14）

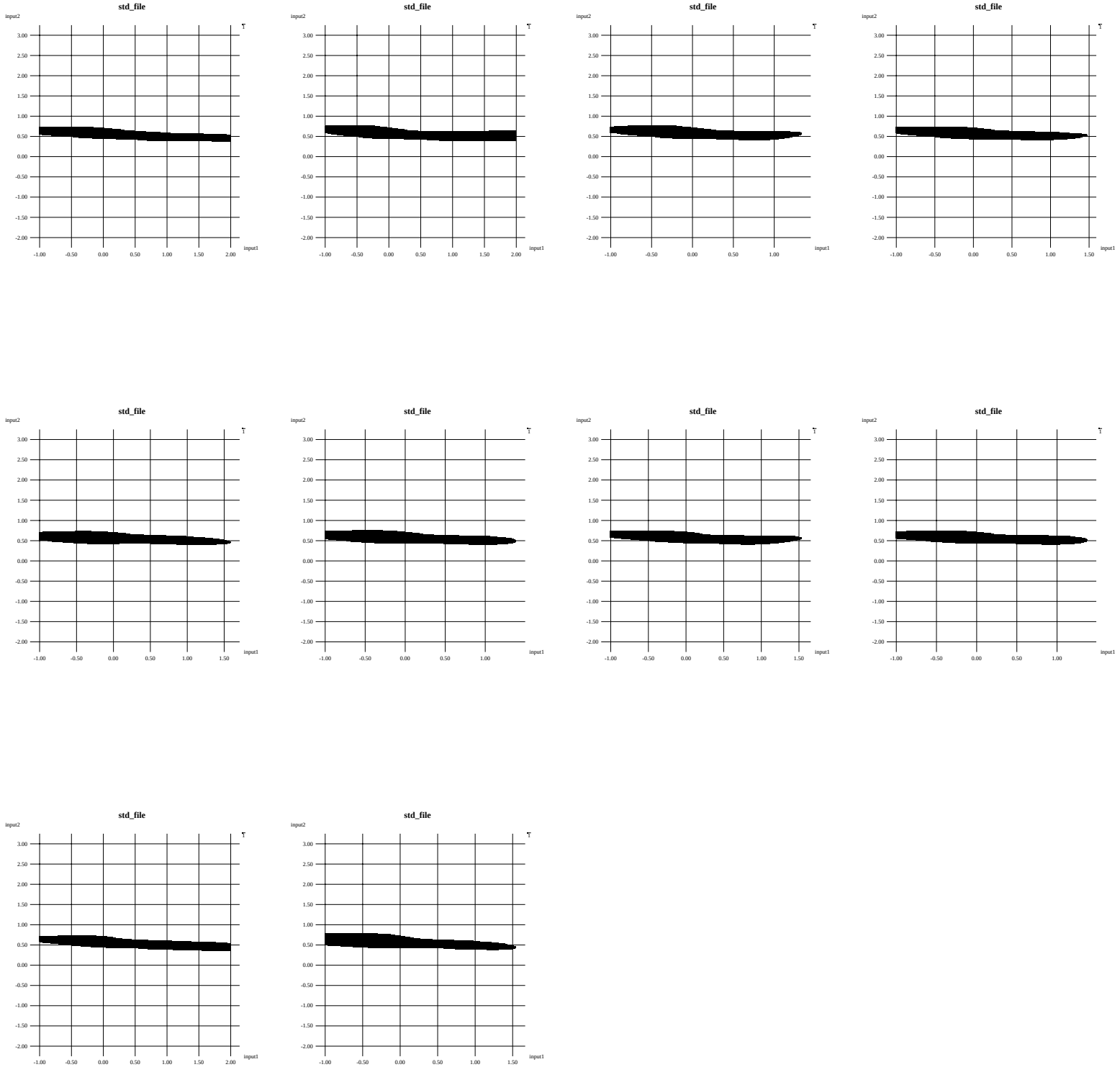

図 6.61: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 16）

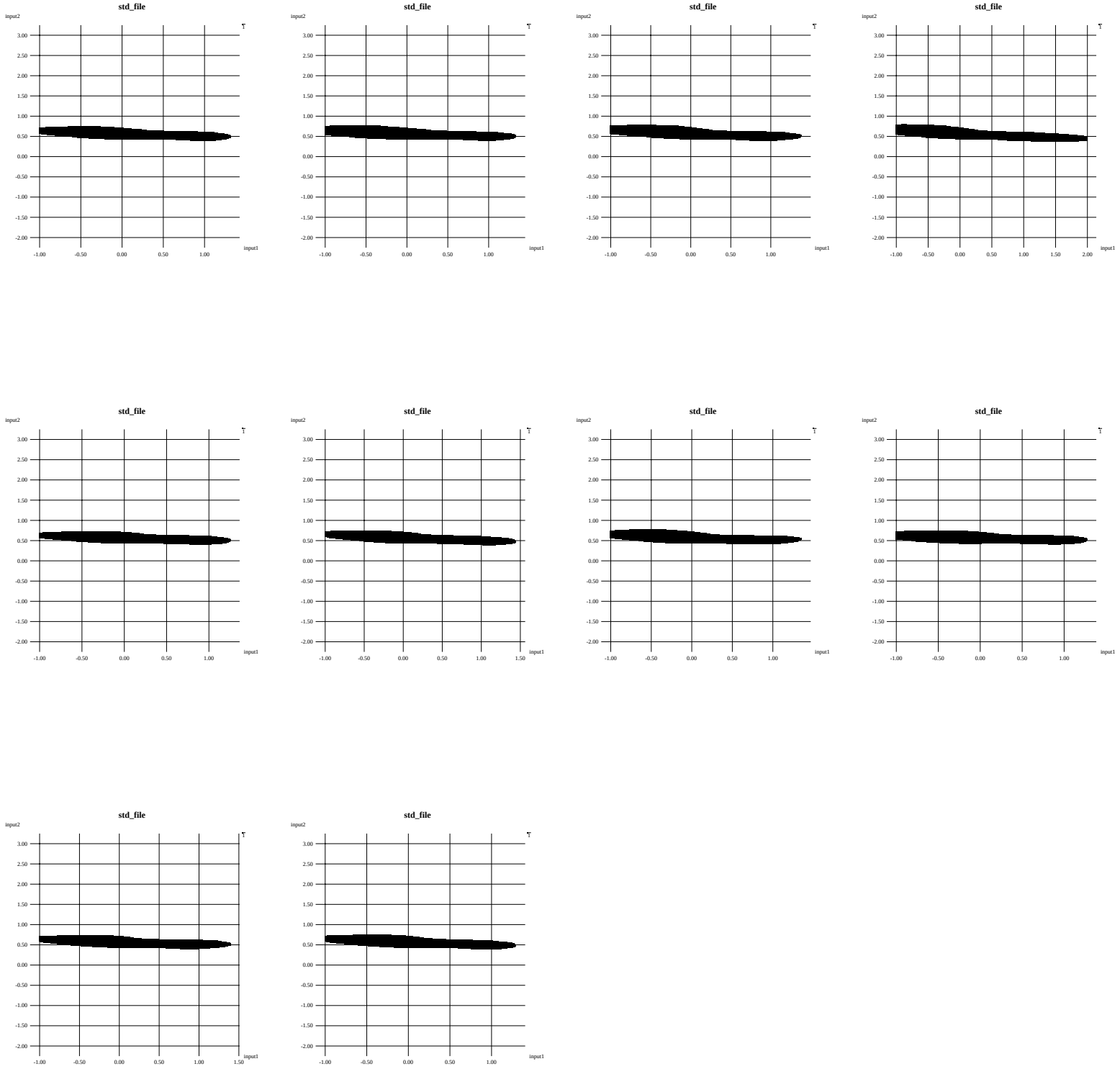

図 6.62: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 18）

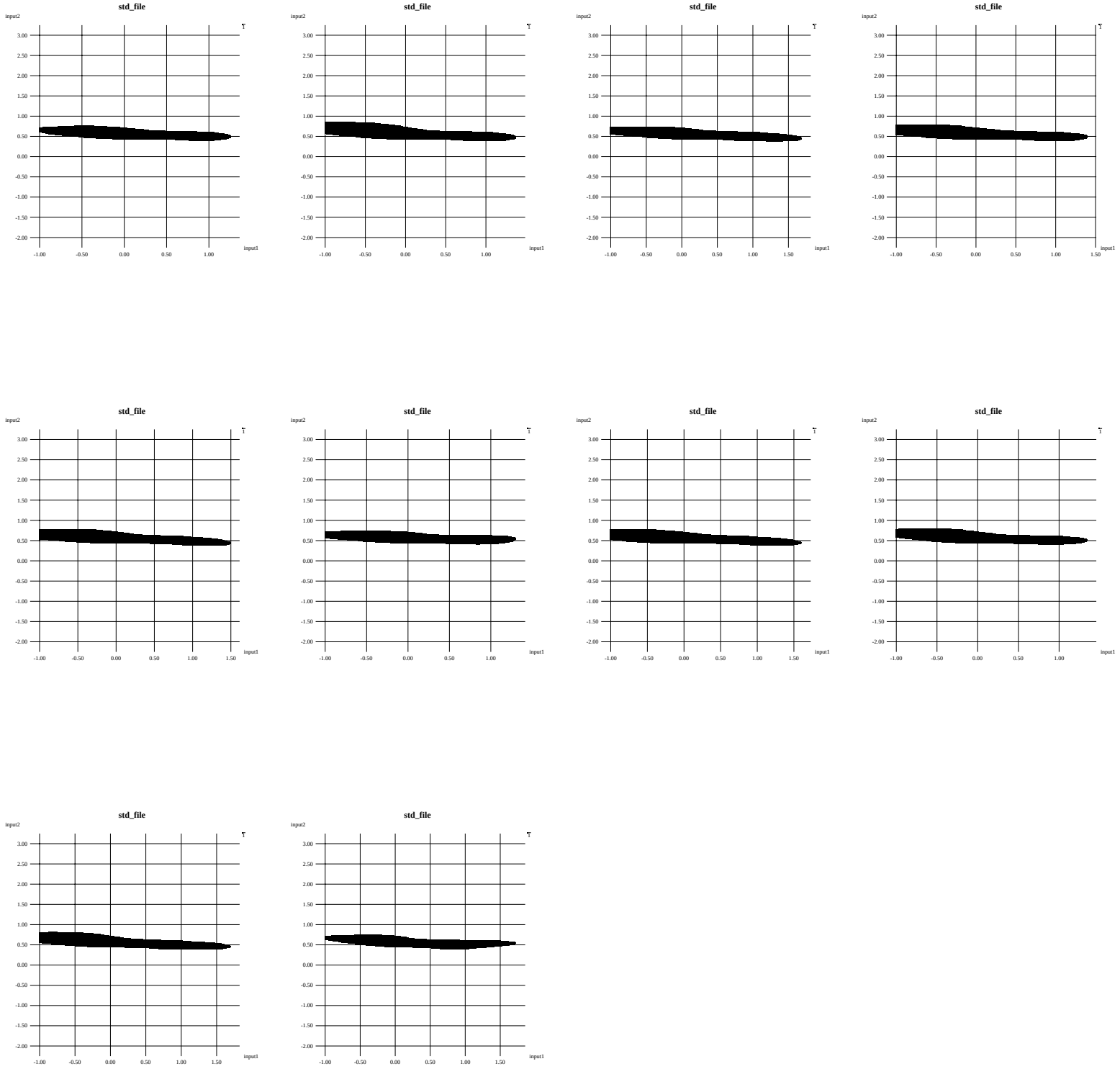

図 6.63: 帯状パターン離散を FTBP で学習した時の出力分布（中間層ノード 20）

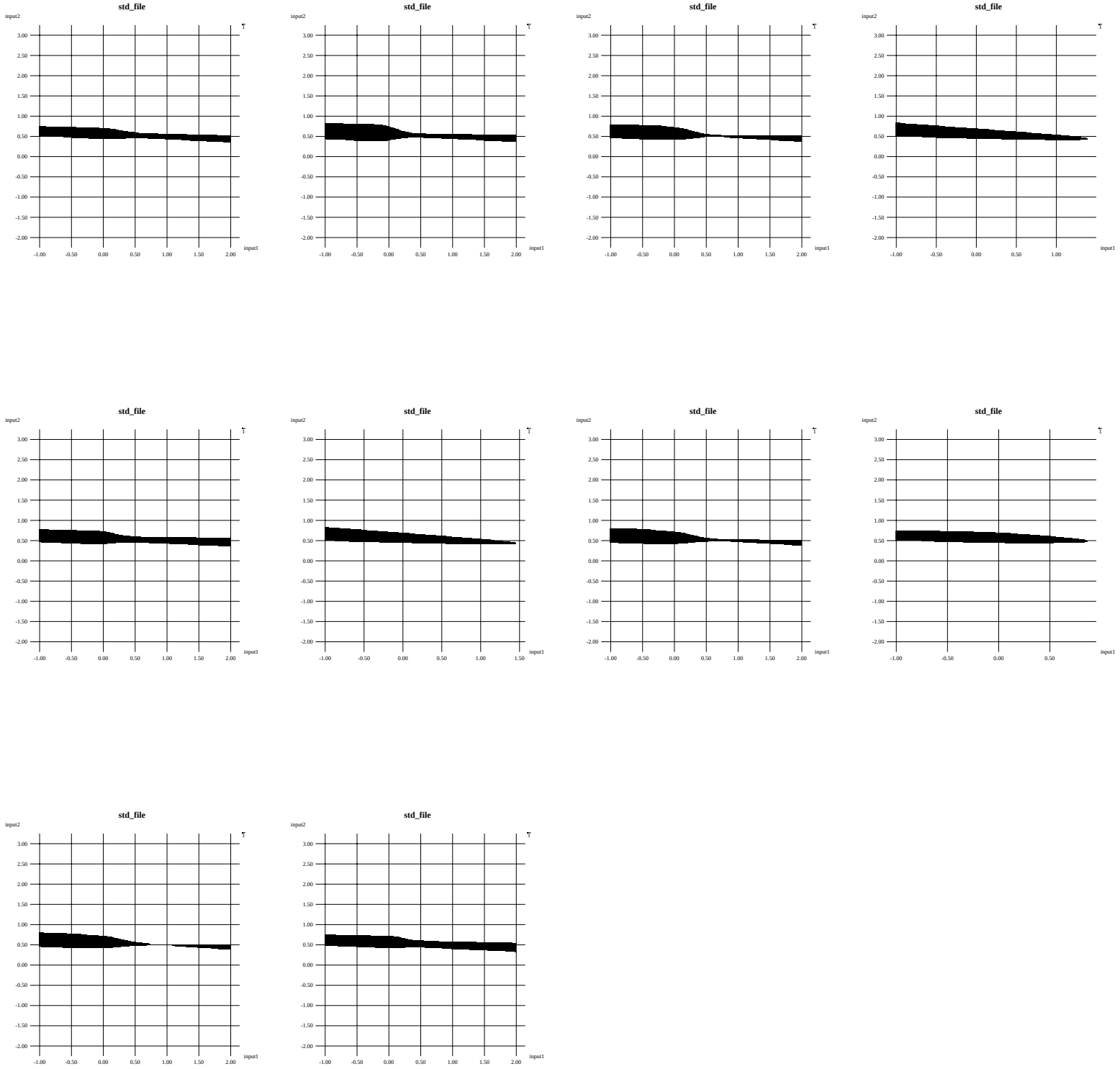

図 6.64: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 4）

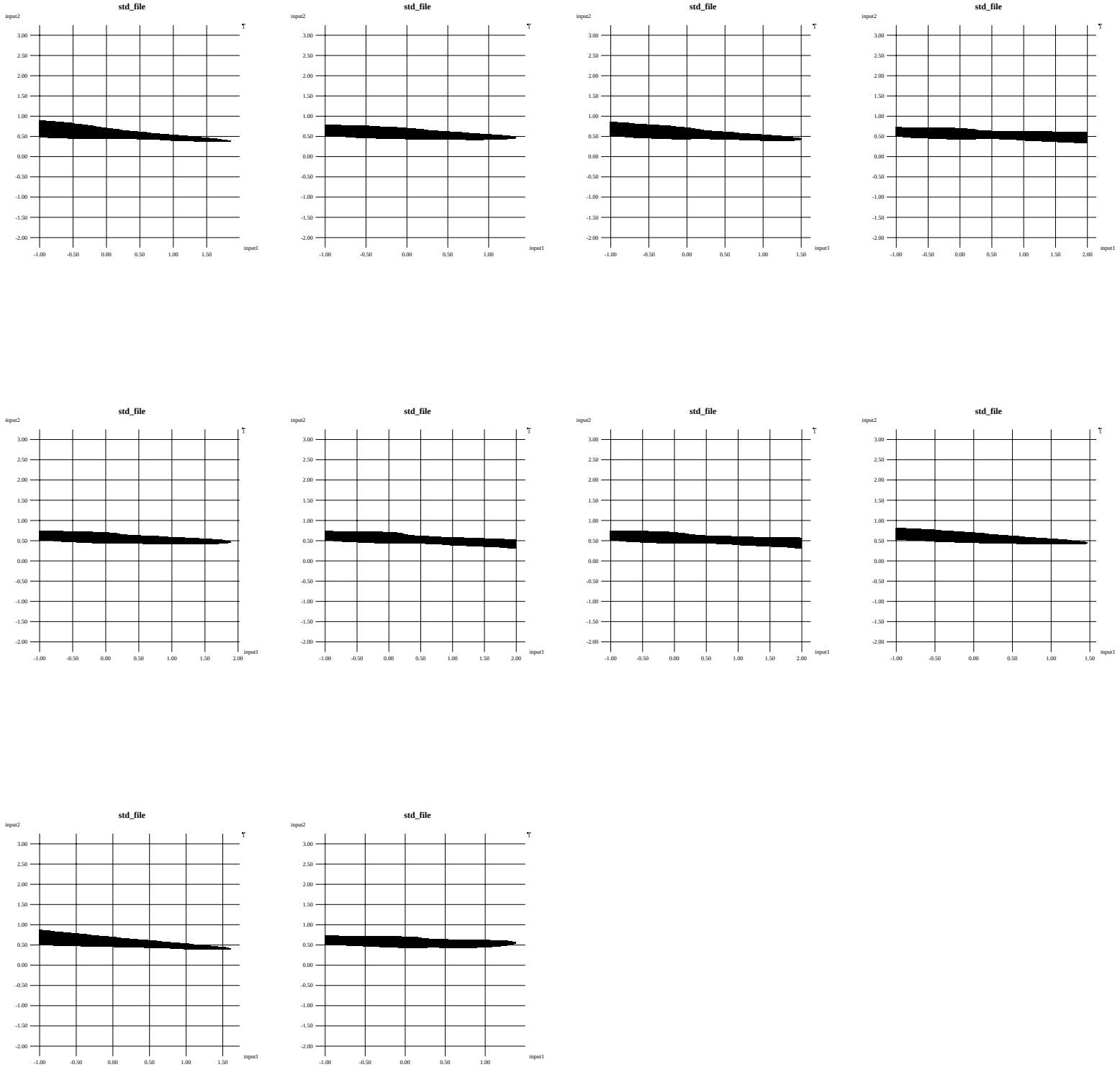

図 6.65: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 6）

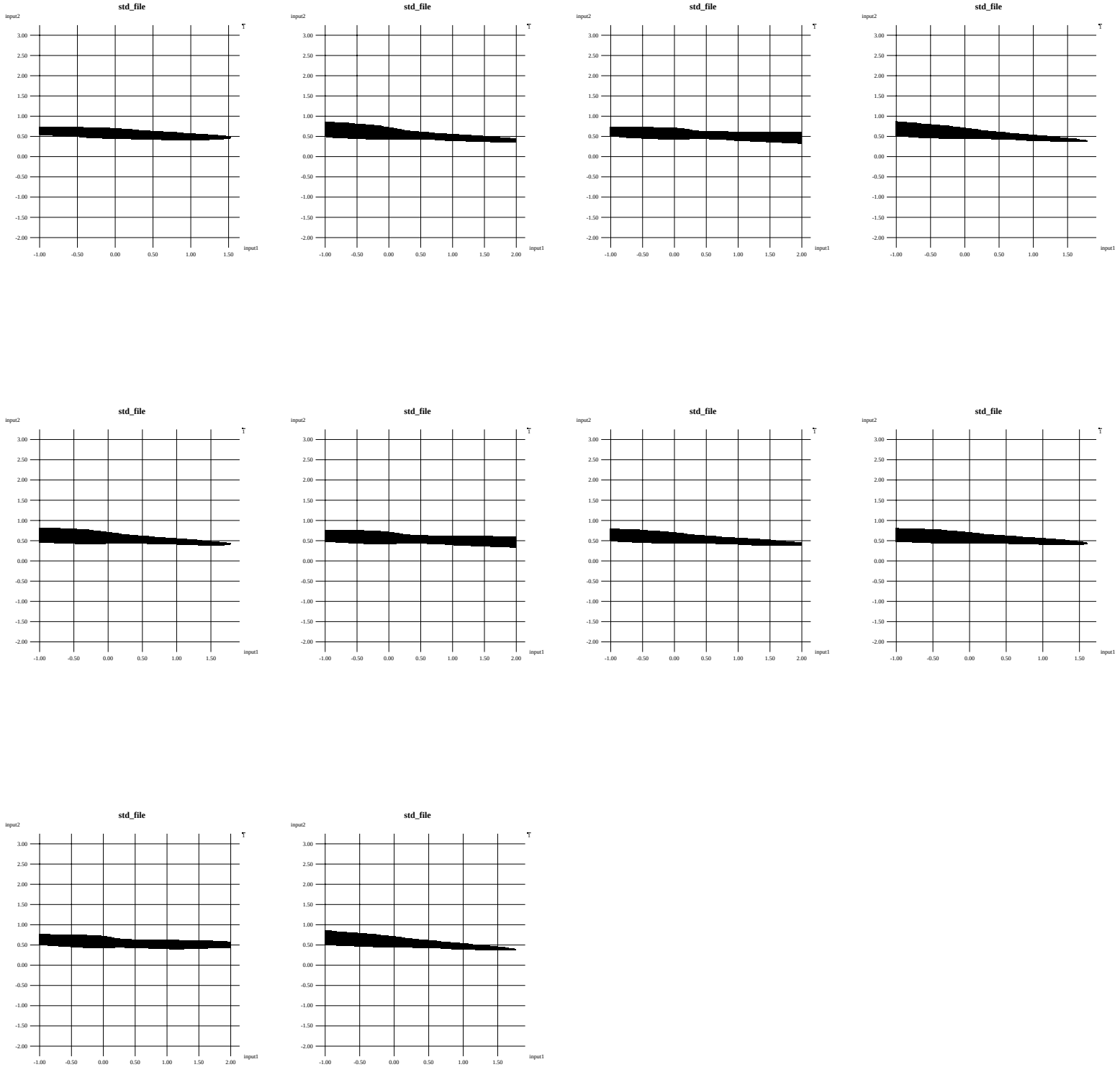

図 6.66: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 8）

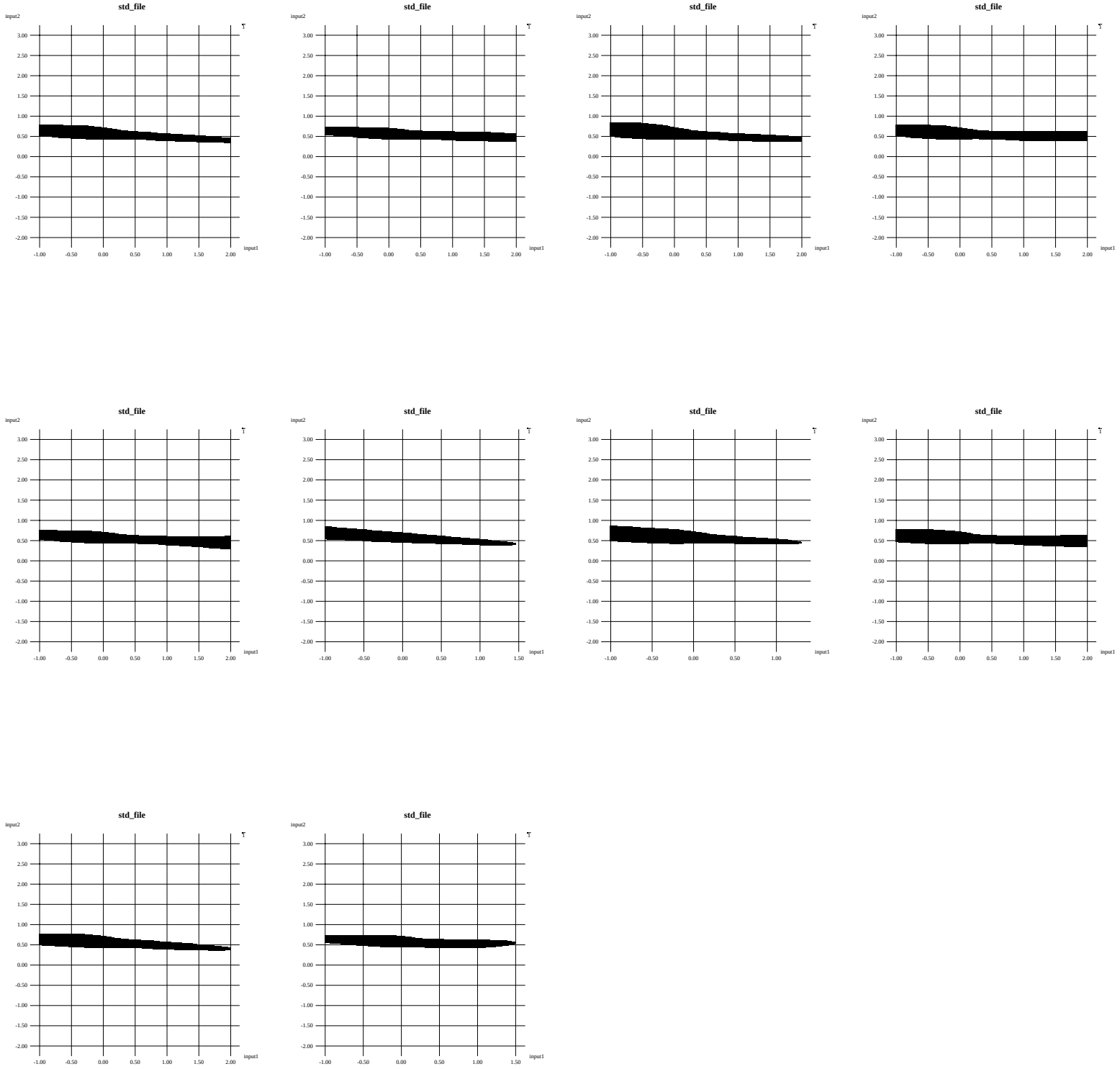

図 6.67: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 10）

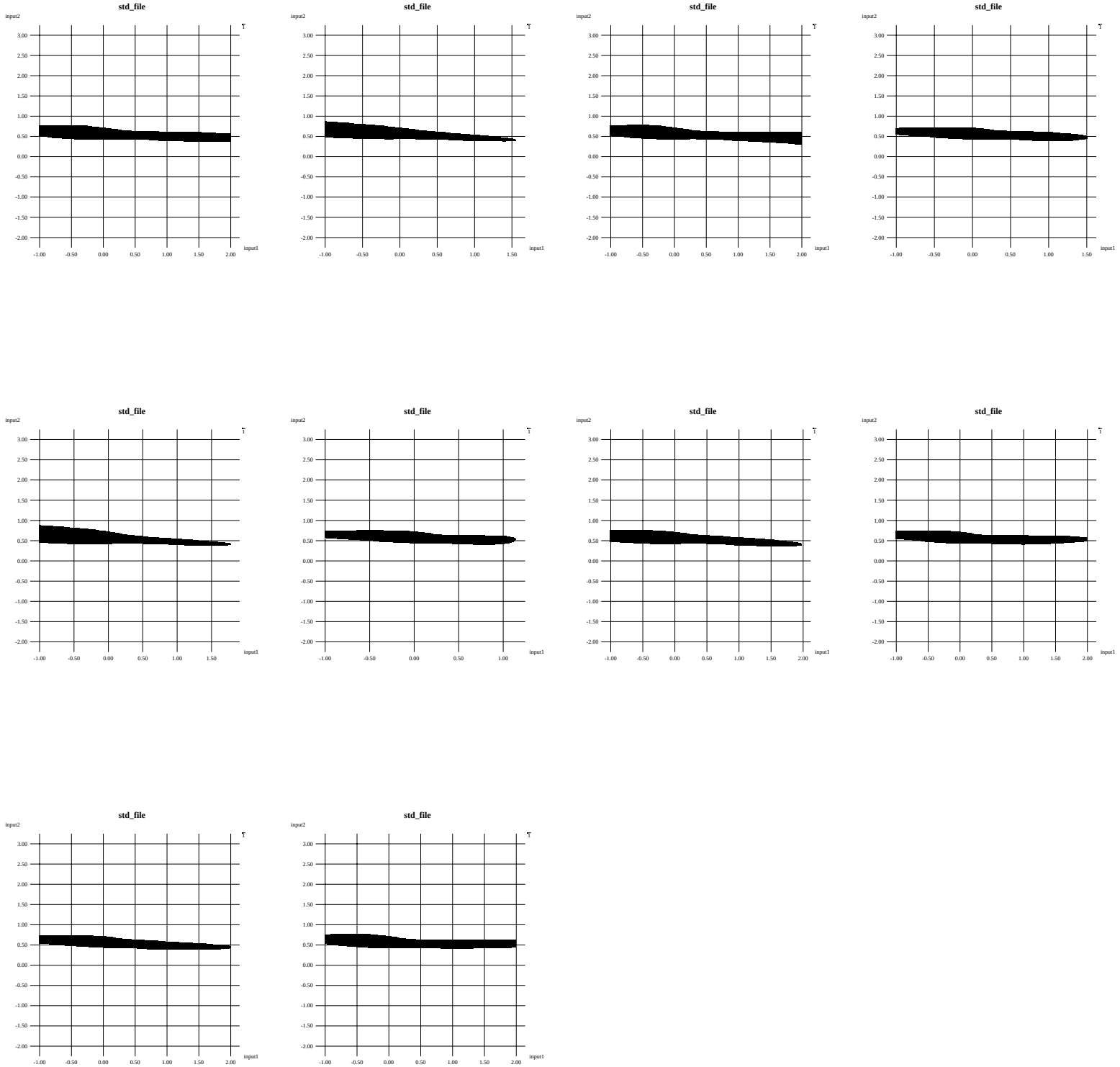

図 6.68: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 12）

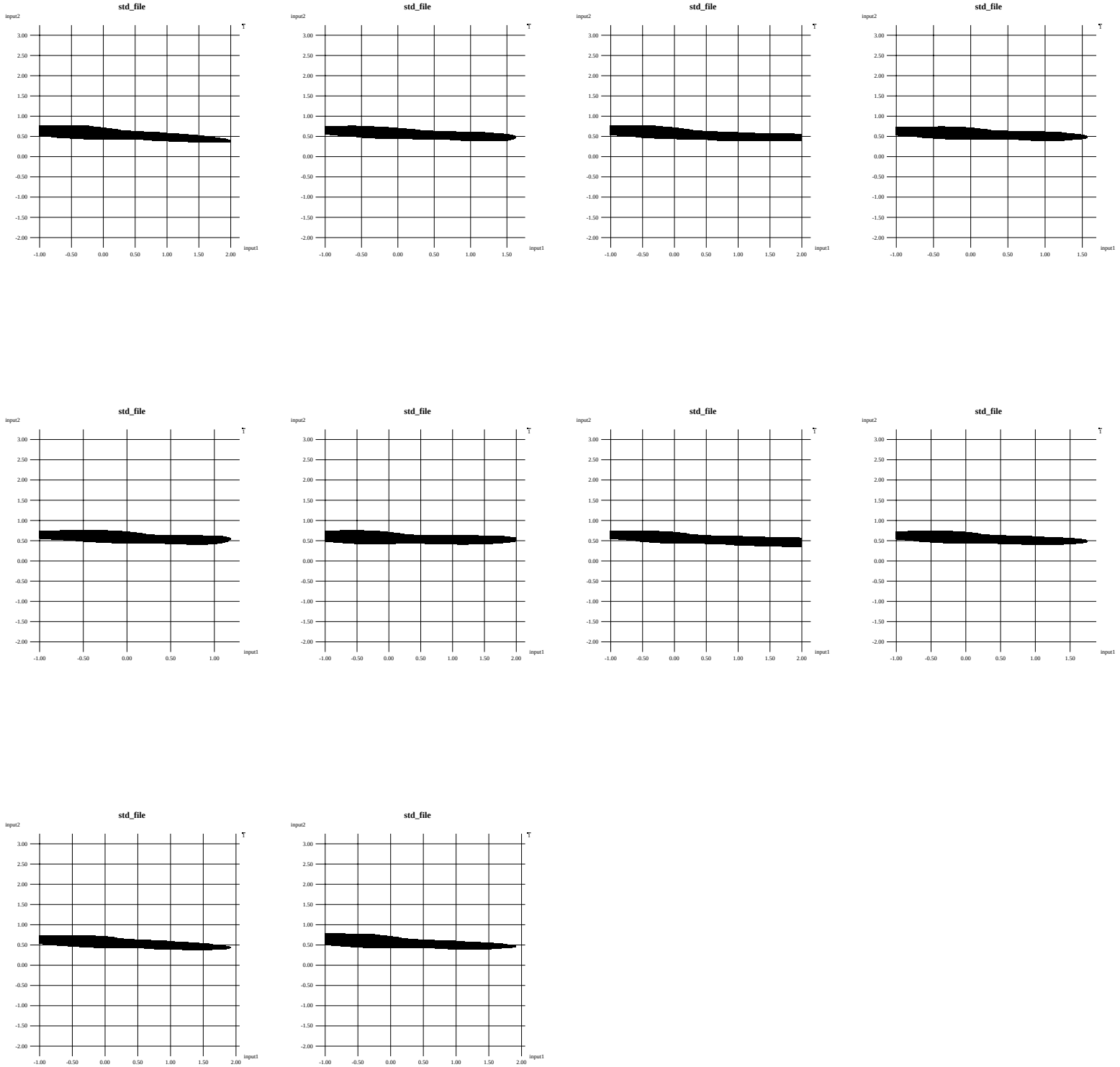

図 6.69: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 14）

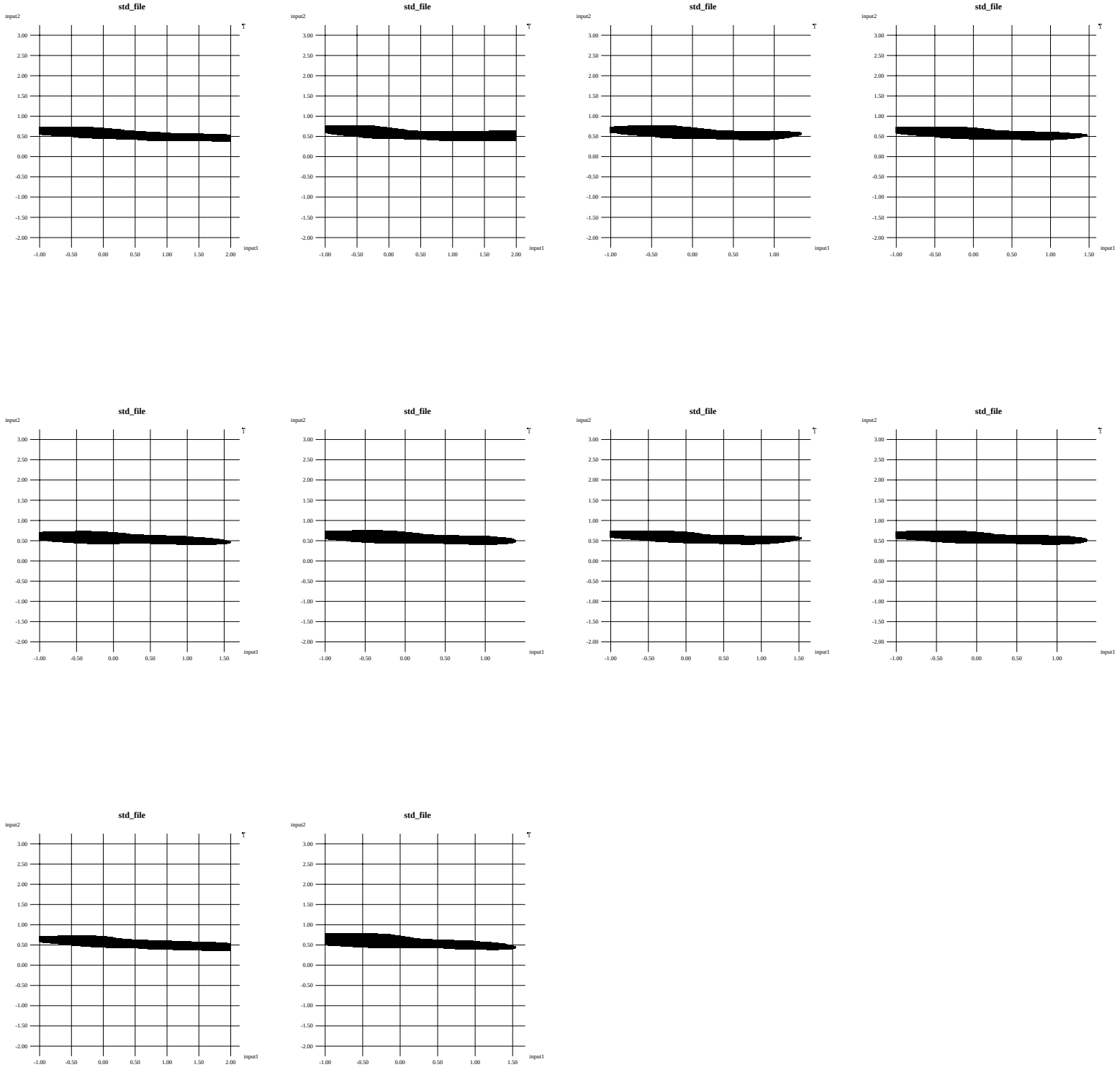

図 6.70: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 16）

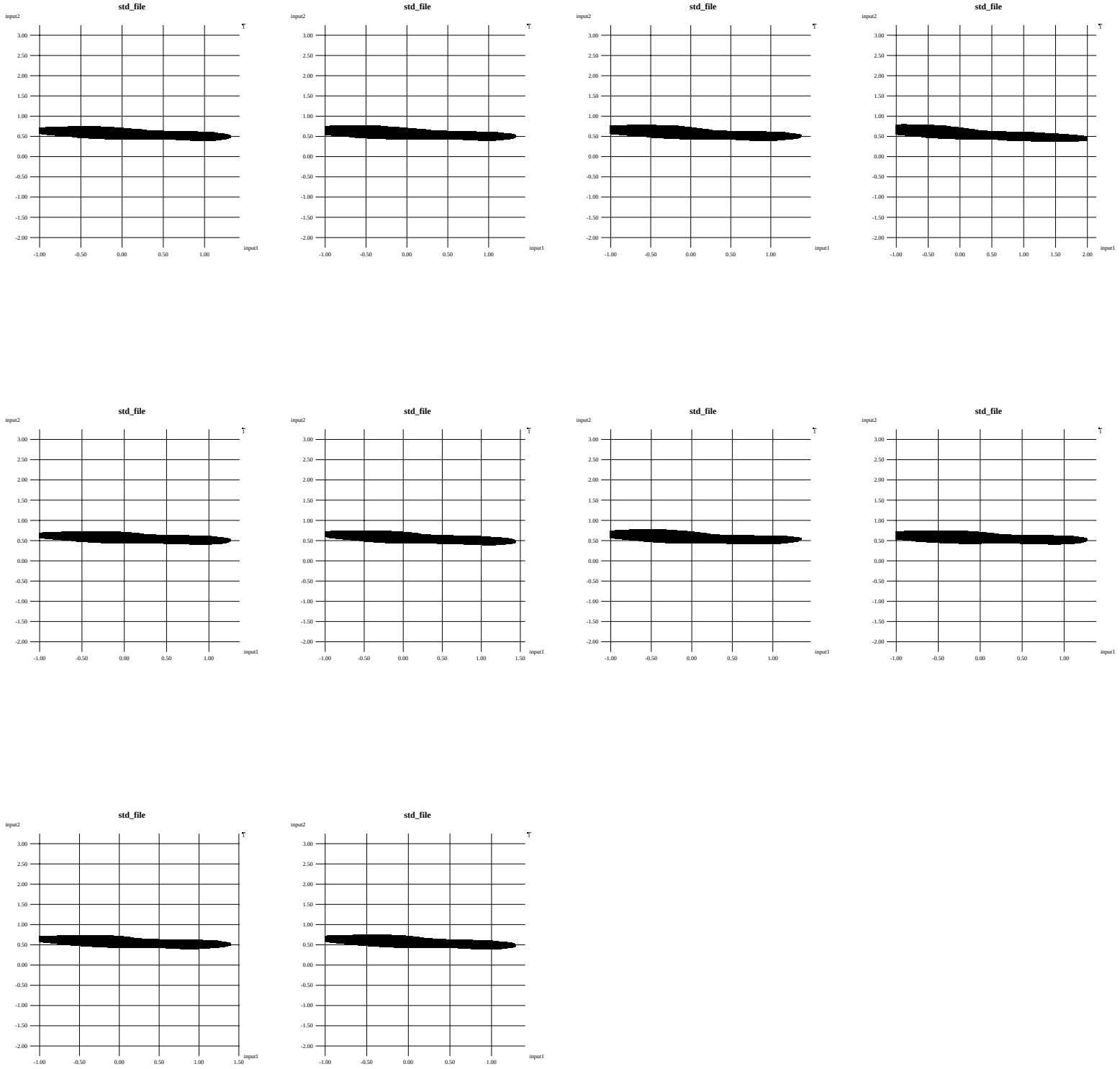

図 6.71: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 18）

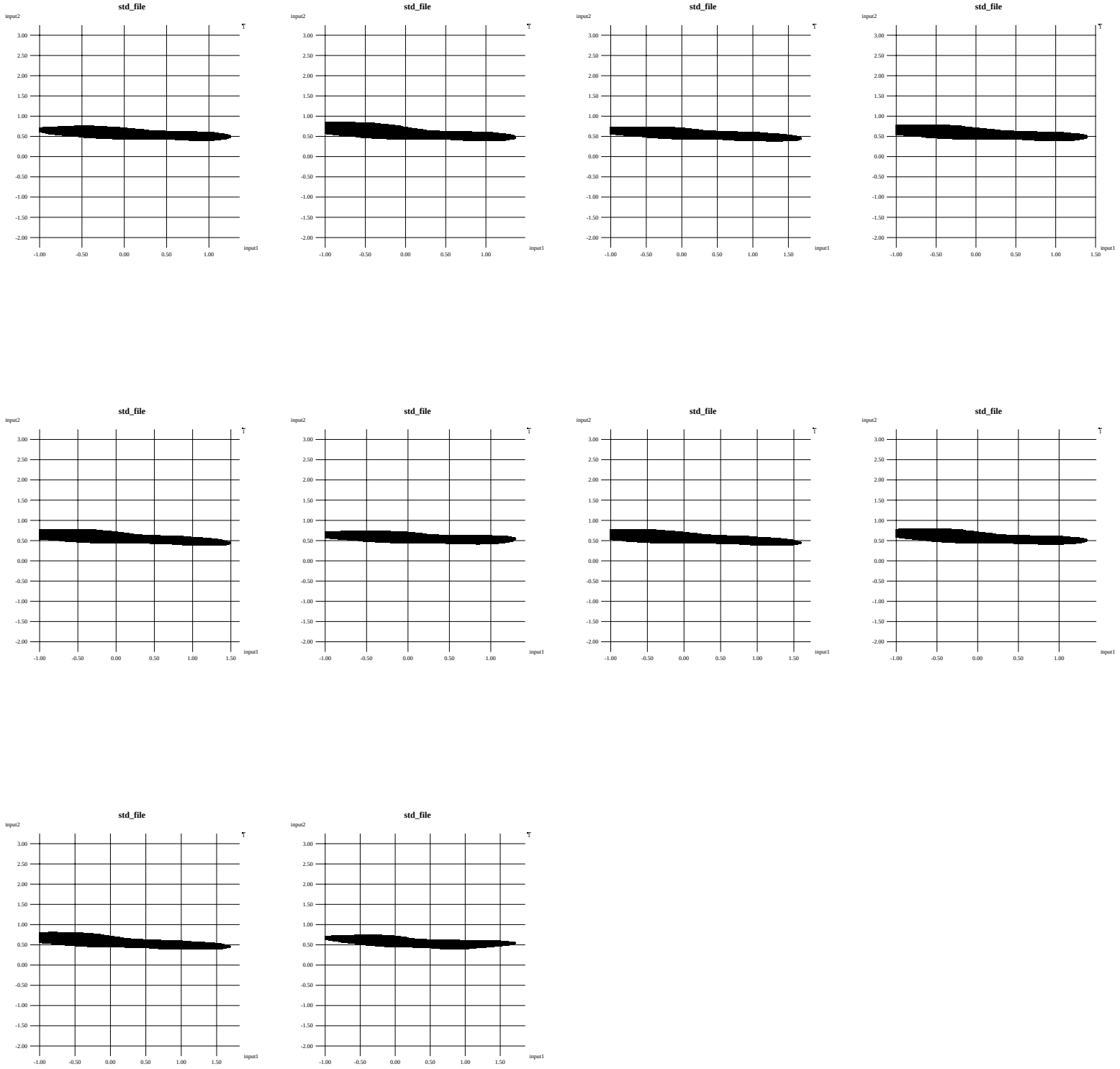

図 6.72: 帯状パターン連続を FTBP で学習した時の出力分布（中間層ノード 20）

付録 F.

関数比較に対する結合荷重

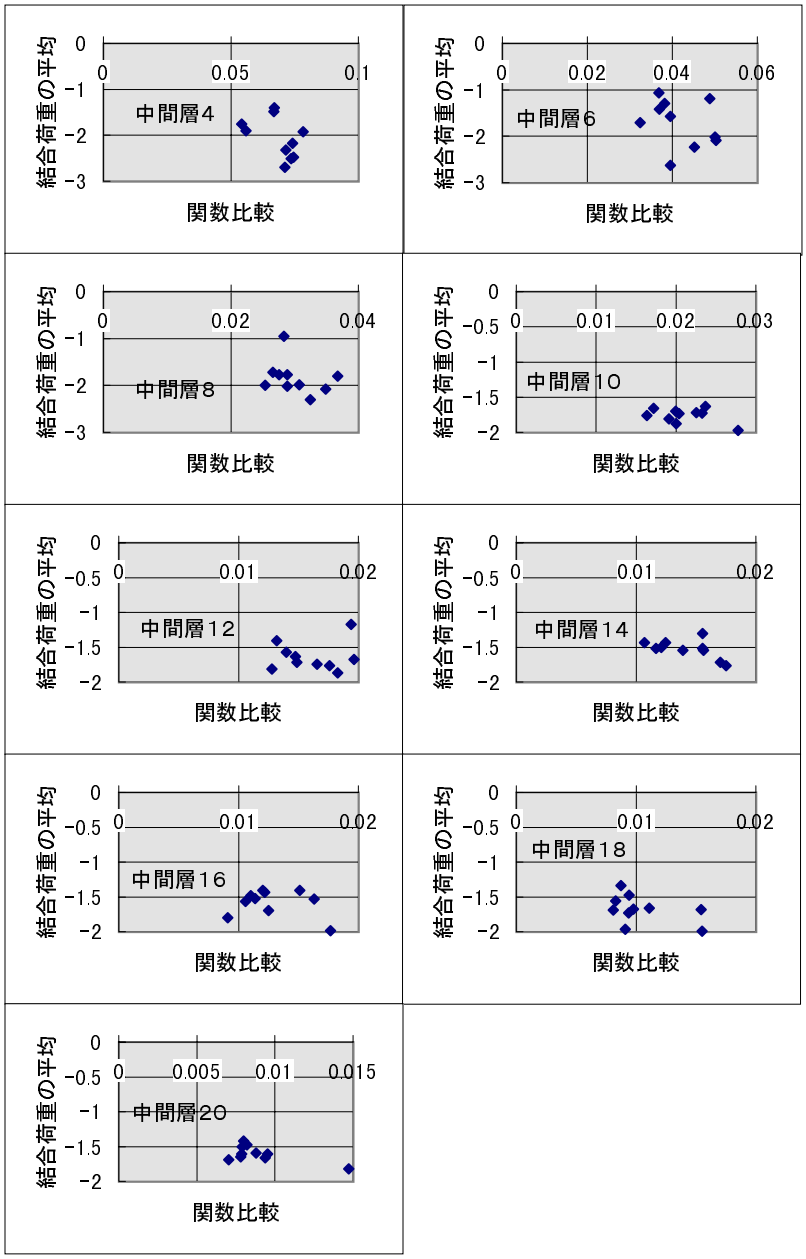

図 6.73: BP で帯状パターン連続を学習した時の関数比較に対する結合荷重

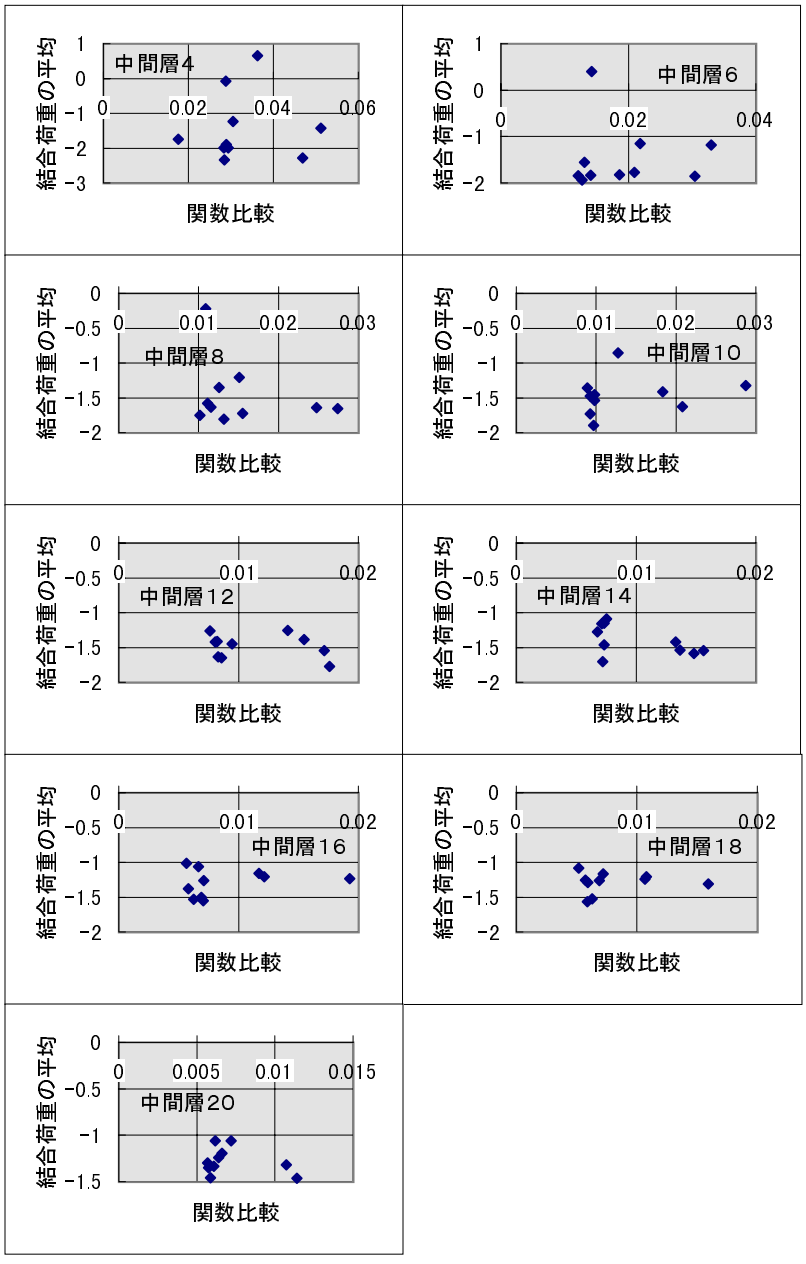

図 6.74: BP で帯状パターン離散を学習した時の関数比較に対する結合荷重

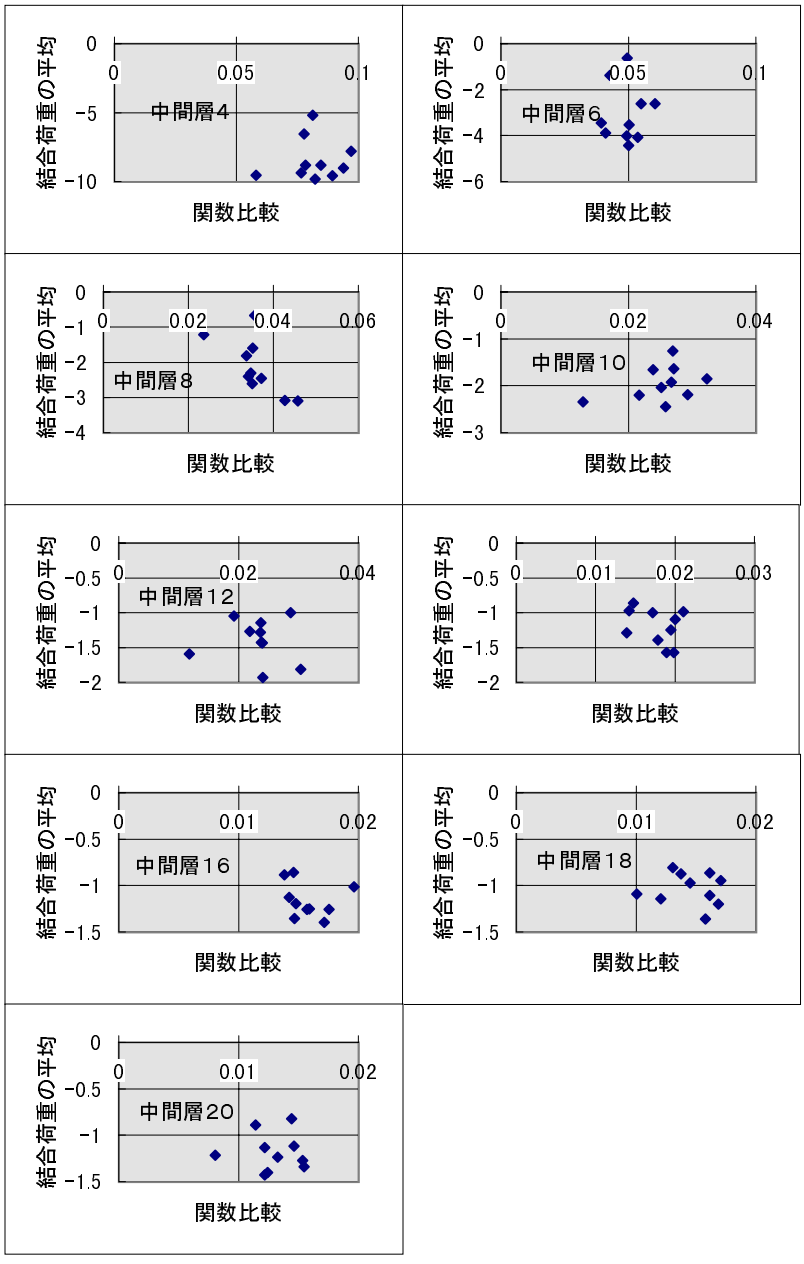

図 6.75: BP で帯状パターン連続を学習した時の関数比較に対する結合荷重

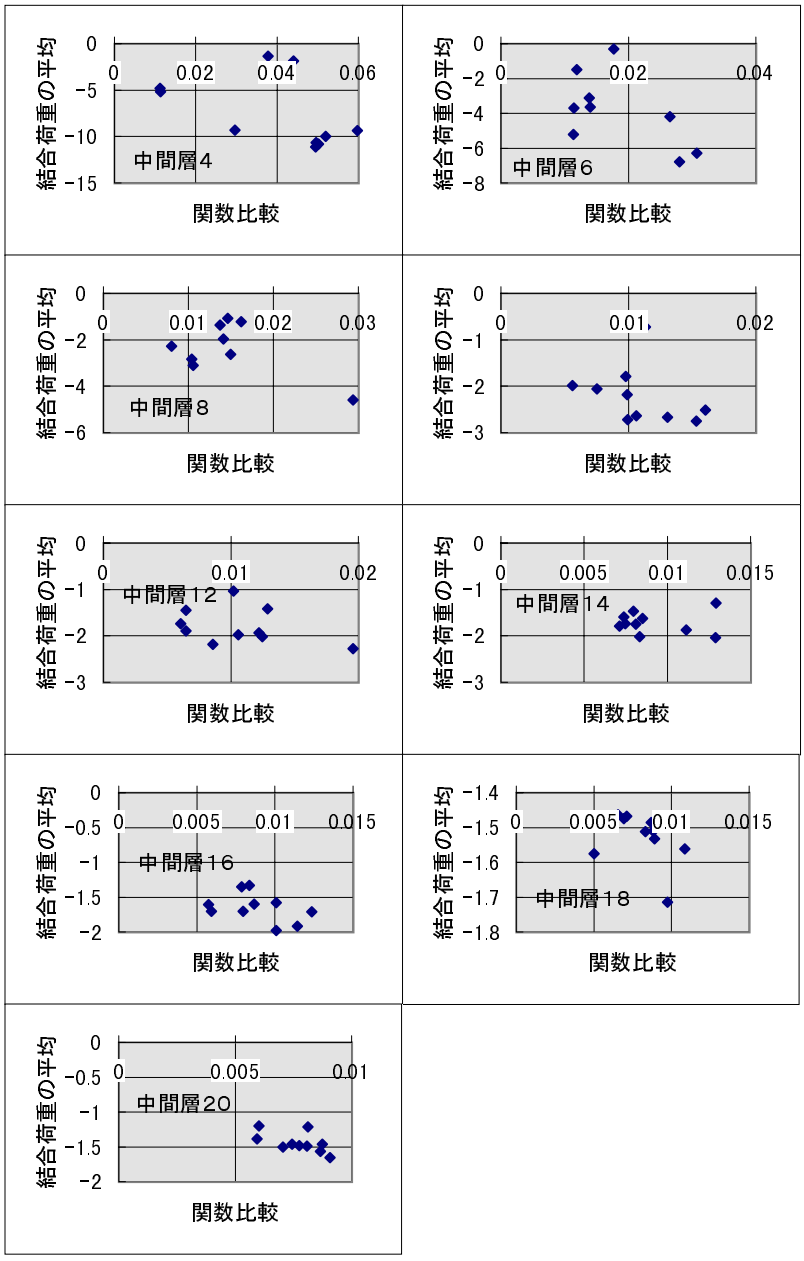

図 6.76: BP で帯状パターン離散を学習した時の関数比較に対する結合荷重

付録 G

関数比較に対する中間層の微係数

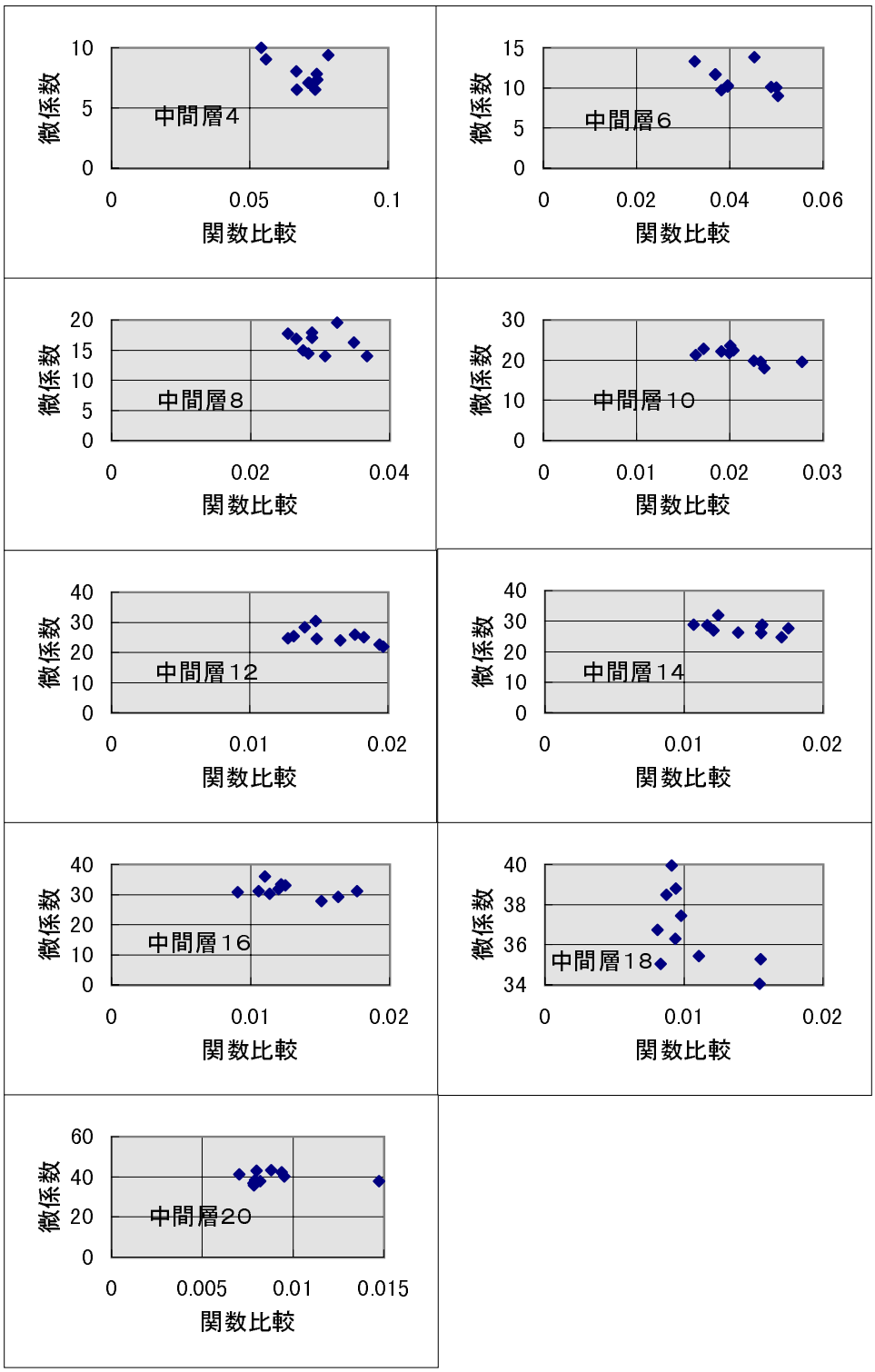

図 6.77: BP で帯状パターン連続を学習した時の関数比較に対する中間層の微係数

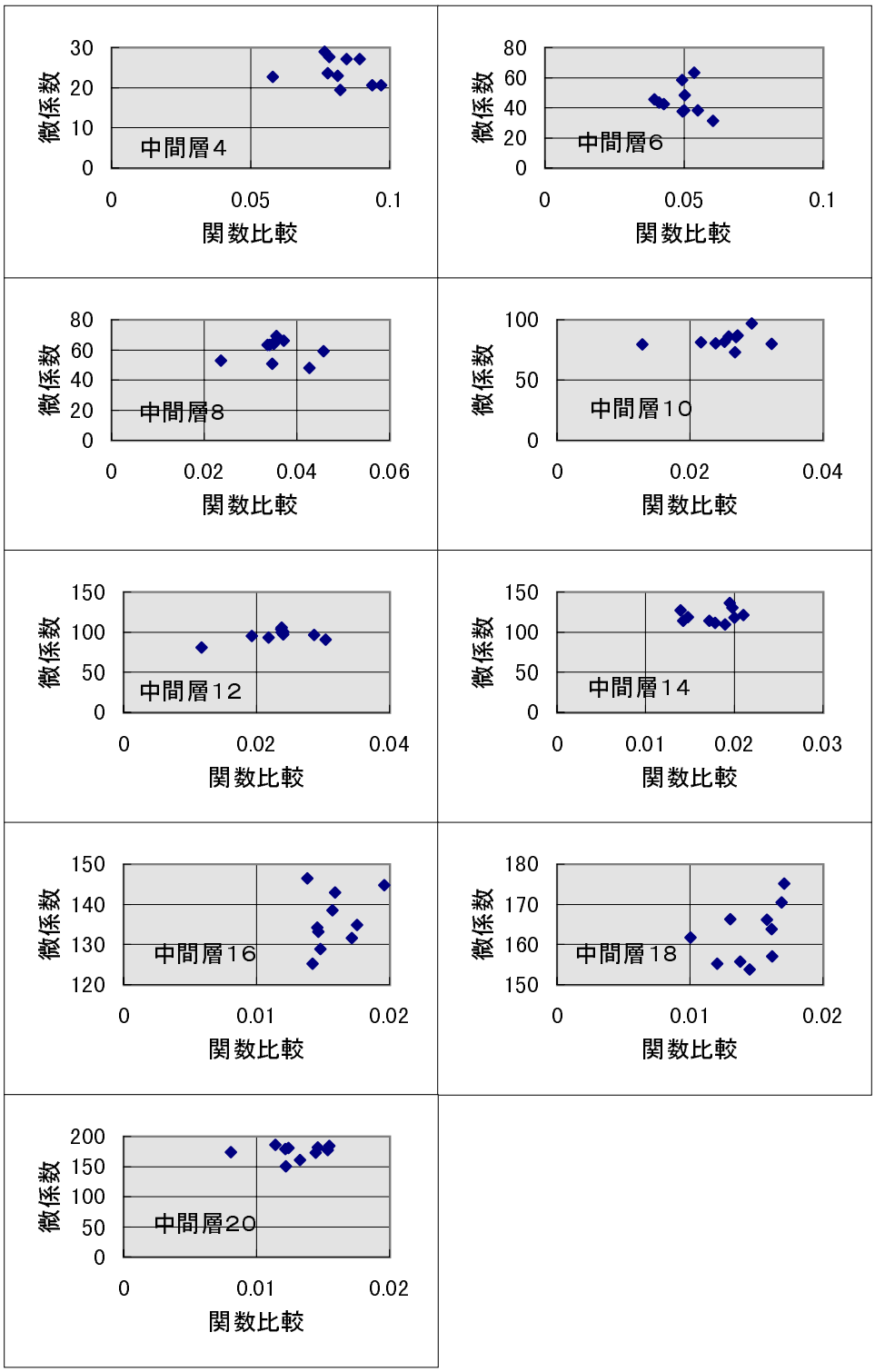

図 6.78: BP で帯状パターン離散を学習した時の関数比較に対する中間層の微係数

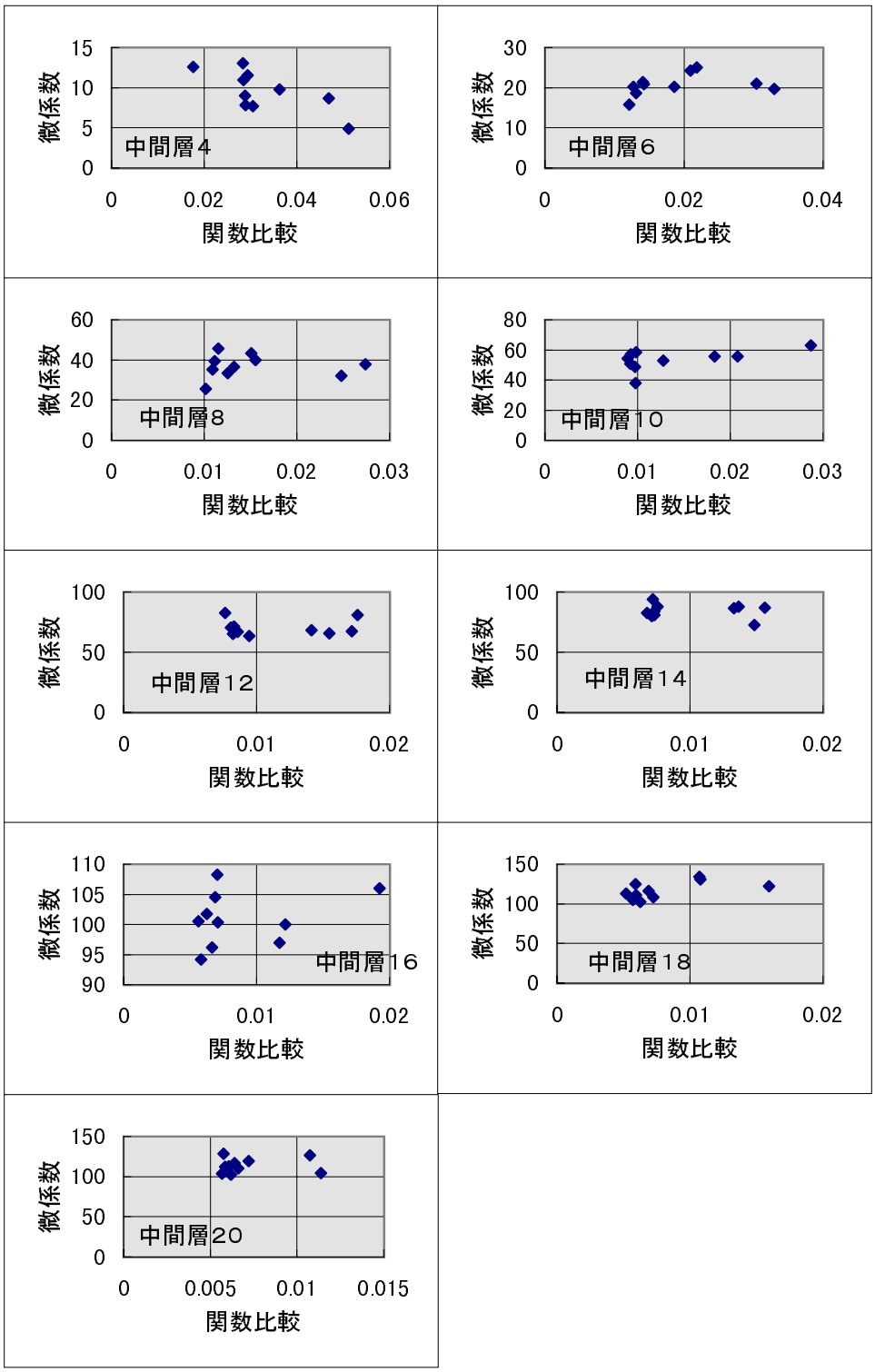

図 6.79: FTBP で帯状パターン連続を学習した時の関数比較に対する中間層の微係数

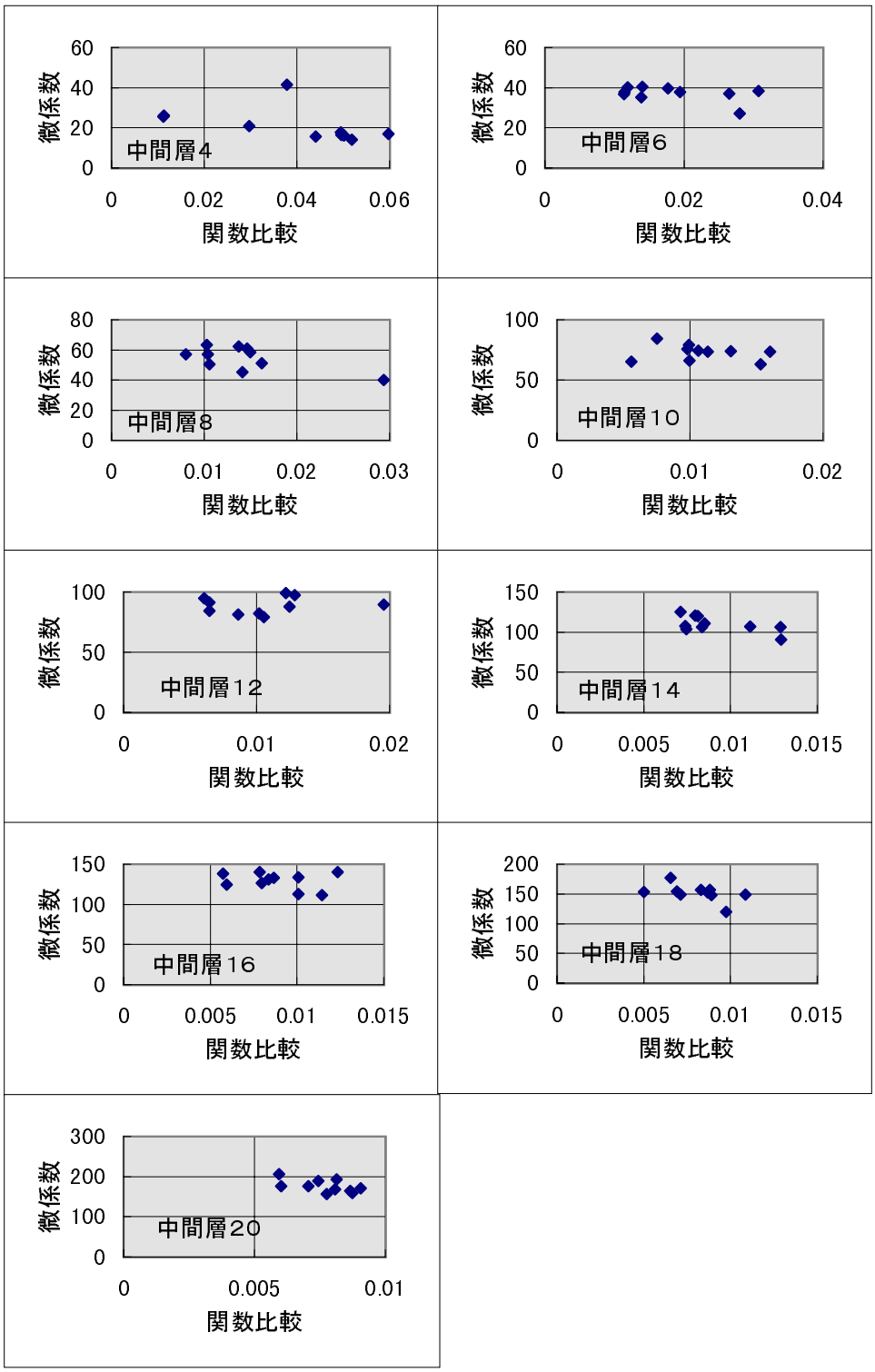

図 6.80: FTBP で帯状パターン離散を学習した時の関数比較に対する中間層の微係数

付録 H

関数比較に対する関数の滑らかさ

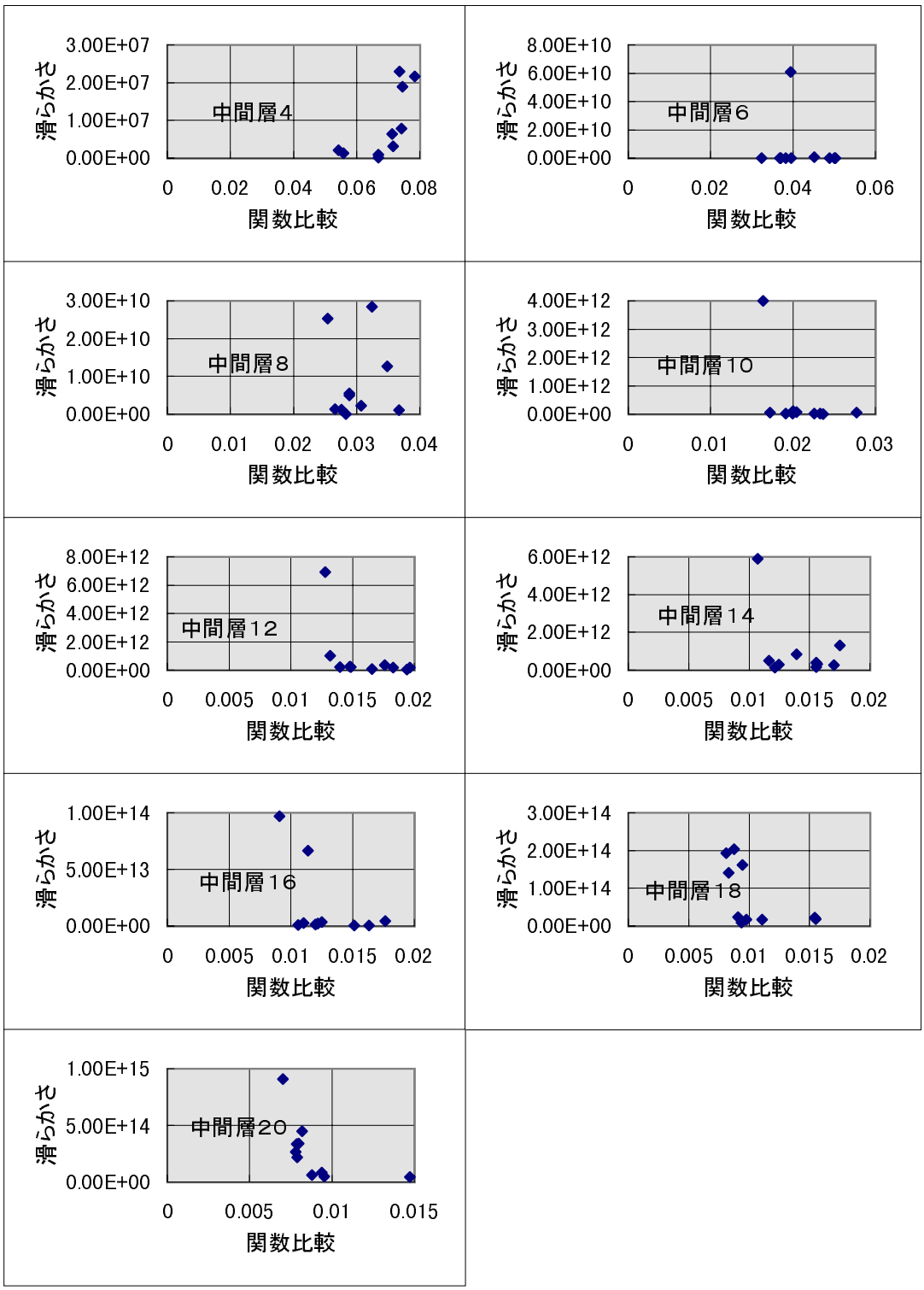

図 6.81: BP で帯状パターン連続を学習した時の関数比較に対する関数の滑らかさ

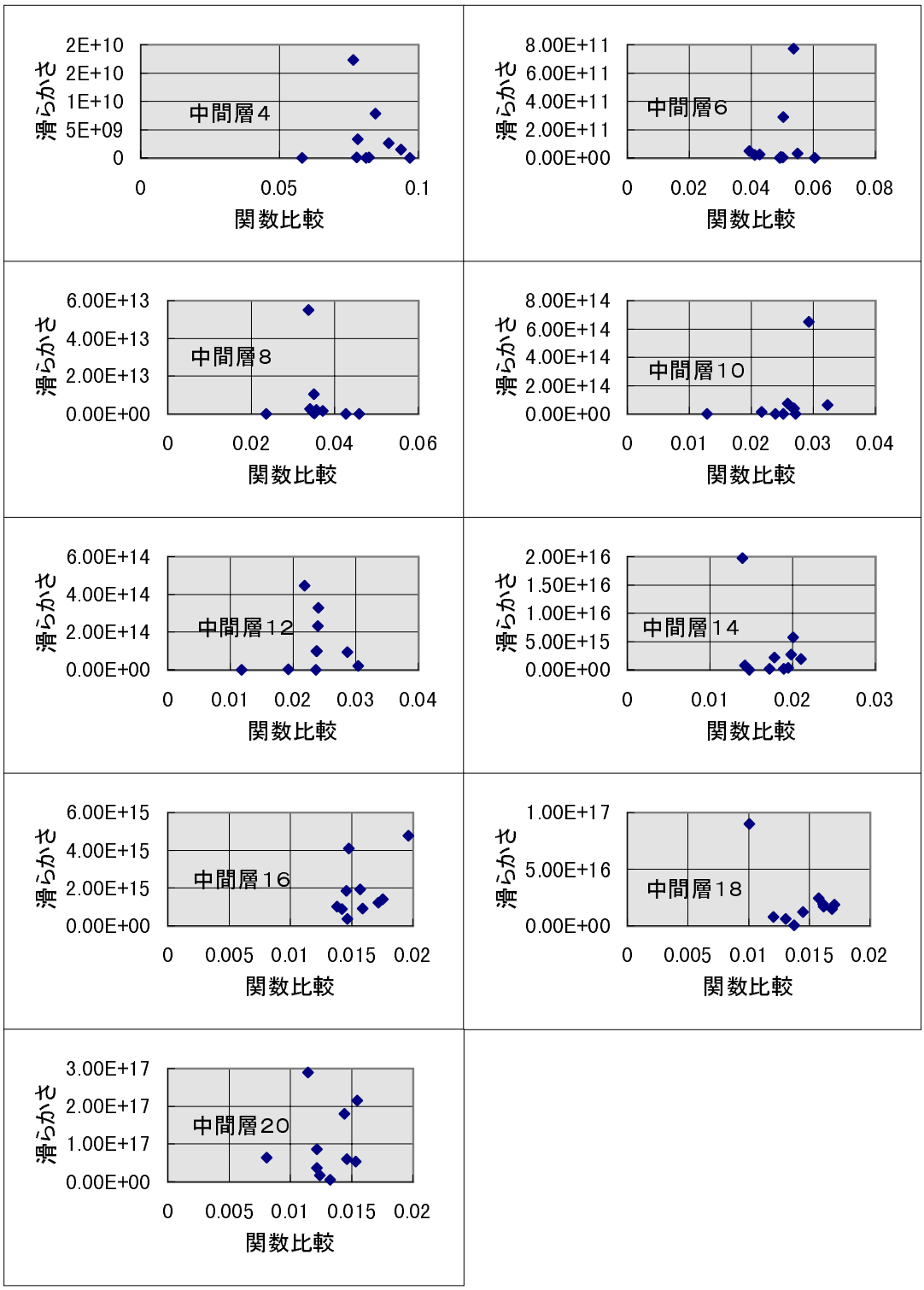

図 6.82: BP で帯状パターン離散を学習した時の関数比較に対する関数の滑らかさ

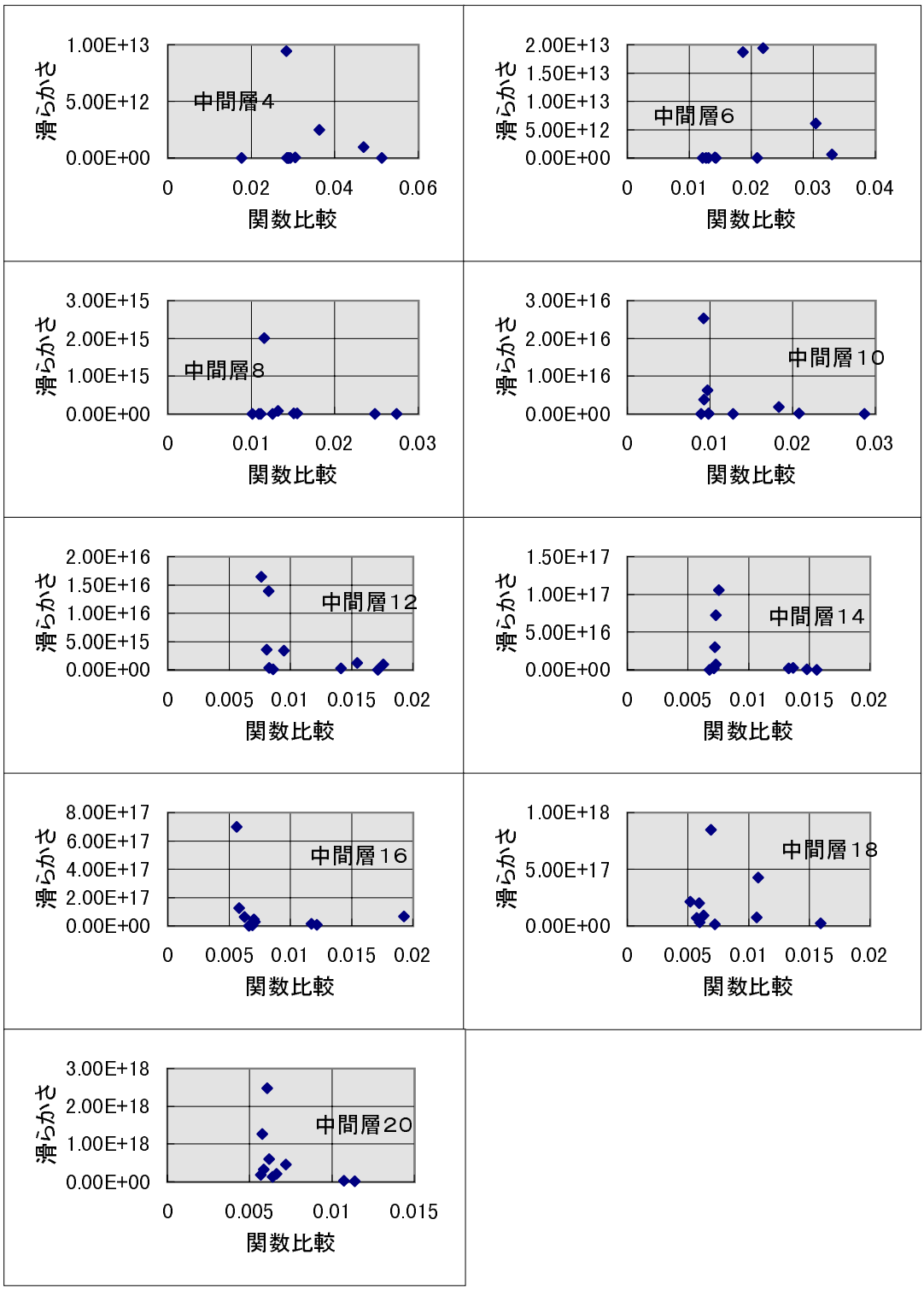

図 6.83: FTBP で帯状パターン連続を学習した時の関数比較に対する関数の滑らかさ

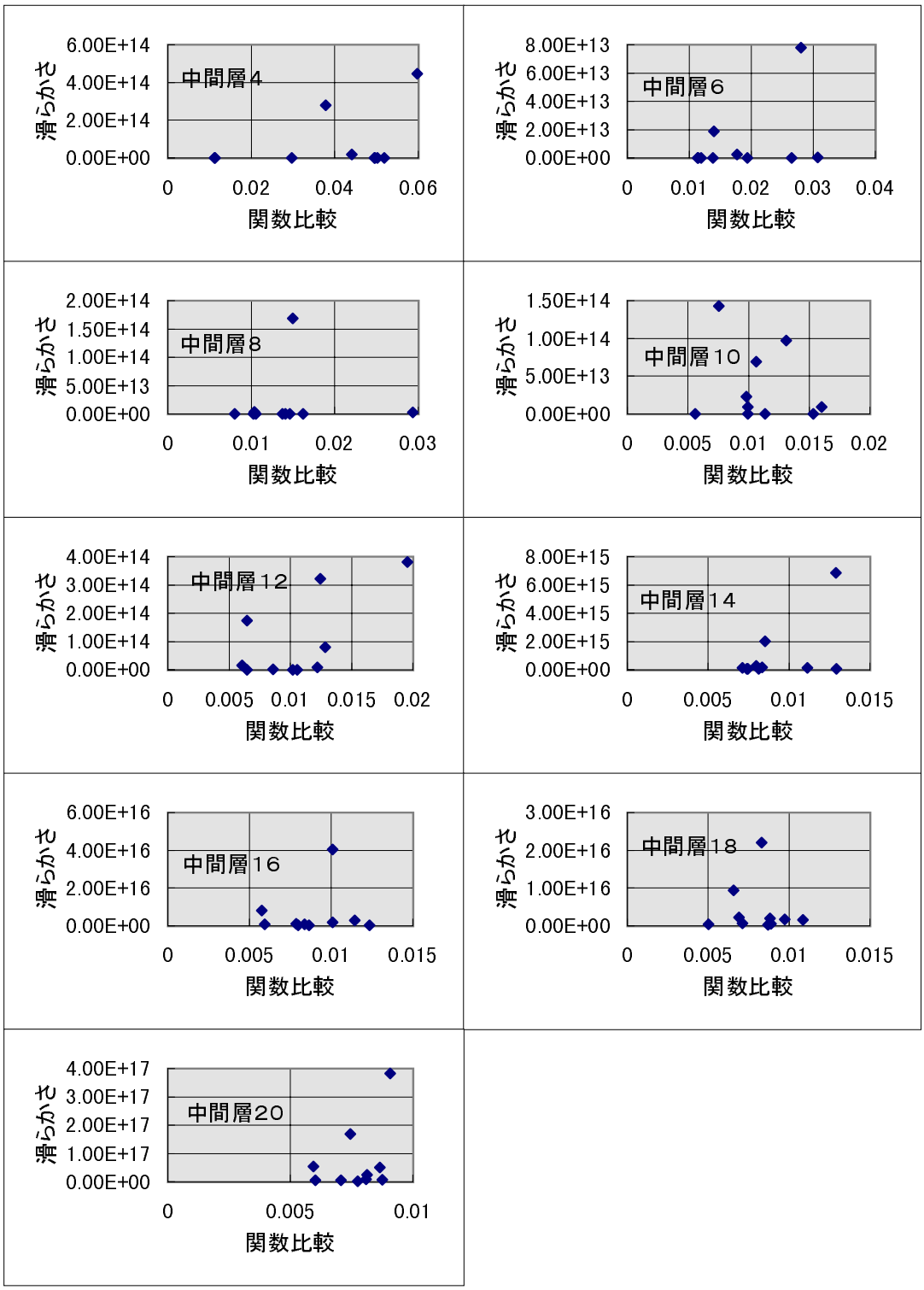

図 6.84: FTBP で帯状パターン離散を学習した時の関数比較に対する関数の滑らかさ