Niigata University
Short-term study program for students from abroad
Related subject "Information Engineering 2"
2nd period on November 26 and December 3 (Monday), 2012
(For a student Javier Guzman)

Invitation to Machine Learnig and Evolutionary Computation

Tatsuya MOTOKI
Dept. of Information Engineering
Niigata University
Ikarashi 2-8050
Niigata 950-2181, Japan
E-mail: motoki@ie.niigata-u.ac.jp

Contents

2nd period on November 26, 2012

2nd period on December 3, 2012

(For Teaching Yourself)

I. A Survey — Computers that Learn —

1 When do we say that a computer learn?

1–1 Inductive Learning

This word means that someone uses "logical reasoning that a general law exists because particular cases that seem to be examples of it exist." (The Oxford Paperback Dictionary.)

In the first learning paradigm, called inductive learning, we (or computers) construct something that explains given examples.

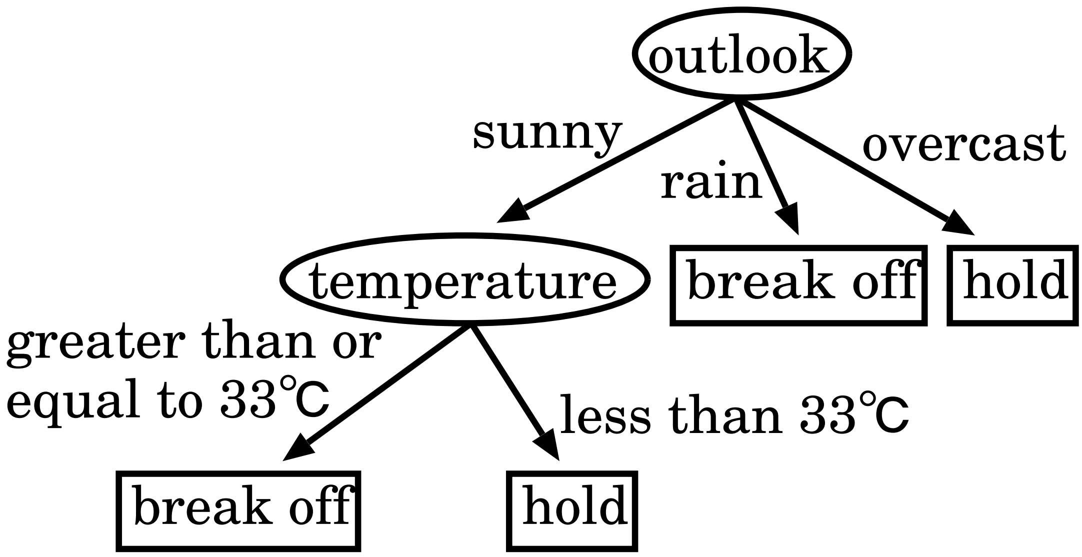
Example1 (Learning Decision Tree)

Suppose that we have to decide wether the athletic meeting should be held based on the outlook, the temterature and the humidity. Then, in the inductive learning paradigm, we try to build a general decision criterion from some decision instances.

Suppose, for example, that we are given following cases:

	attribute of case			decision of whether the athletic meeting is held
	outlook	temperature	humidity	
case1	sunny	40°C	low	break off
case2	sunny	30°C	high	hold
case3	overcast	20°C	low	hold
case4	rain	15°C	middle	break off

We can build the following kind of structure, called a decision tree, that explains the given cases.



This decision tree says that for deciding an arbitrarily given case,

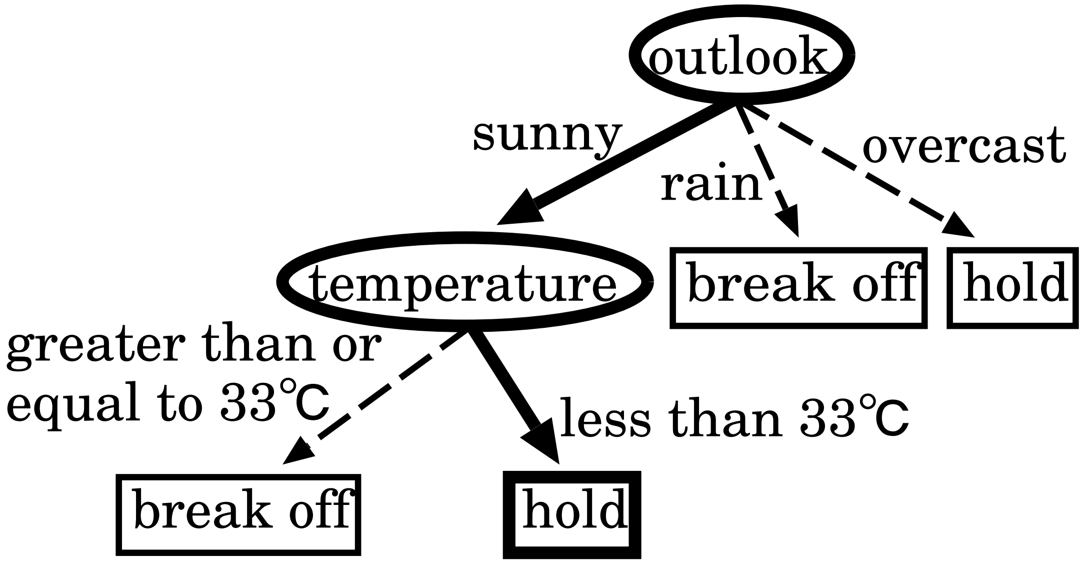
> **we should first check up the outlook attribute and infer that**
>
> **if outlook="rain" then the decision is "break off",**
> **if outlook="overcast" then the decision is "hold", and**
> **if outlook="sunny" then**
>
> > **we should next check up the temperature attribute and infer that**
> >
> > **if temperature$\geq$ 33°C then**
> > **the decision is "break off", and**
> > **if temperature$<$ 33°C then**
> > **the decision is "hold."**

For example, given a case

attribute of case			decision of whether the athletic meeting is held
outlook	temperature	humidity	
sunny	25°C	middle	hold

,

we can use the above tree to derive a decision as follows:



As a second example, suppose that we have to find a (logic) program that can deduce given instance facts. In such a case, we can construct or search a logic program that satisfies the given condition.

This is also an example of inductive learning, (although I do not give a detailed explanation.)

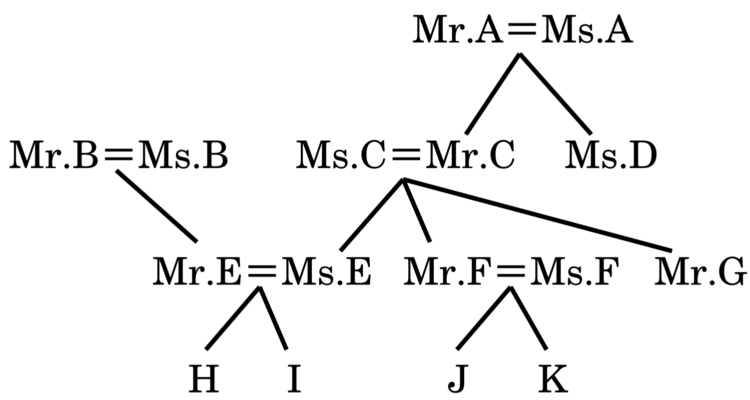
Example2 (Inductive Logic Programming)

[functions whose ranges are {true, false}]

Given several facts about some predicate whose definition is unknown, we can search a logic program that defines this predicate.

Suppose, for example, that we are given the following facts about blood relation:

father(Mr.A, Mr.C), father(Mr.A, Ms.D), ...
mother(Ms.A, Mr.C), ...
married(Mr.A, Ms.A), ...
male(Mr.A), ...
female(Ms.A), ...



And suppose that we are also given the following facts about the target predicate "grandparent":

grandparent(Mr.A, Ms.E), ...,
grandparent(Ms.C, K), ...
¬ grandparent(Mr.A, H), ...,
¬ grandparent(Mr.B, J), ...

⇨ **Then we can search a logic program that defines the target predicate "grandparent."**

Example of a solution program:

```
parent(X,Y) :- mother(X,Y).
parent(X,Y) :- father(X,Y).
grandparent(X,Y) :- parent(X,Z),
                    parent(Z,Y).
```

logical meaning

A person x is a parent of y if x is a mother of y.
A person x is a parent of y if x is a father of y.
A person x is a grandparent of y if x is a parent of z and z is a parent of y.

If these rules are established, we can deduce any grandparent-relation from facts on father- and mother-relations.

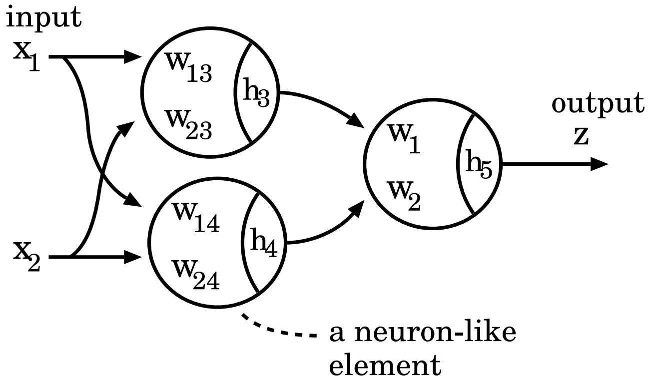
1–2 Adaptive Learning

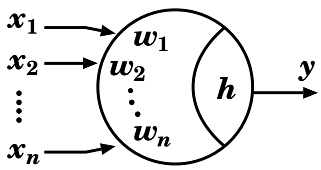
In the second learning paradigm, called **adaptive learning**, we (or computers) repeatedly improve some system parameters so that the behaviour of the system will be better.

Example3 (Training Artificial Neural Networks)

Consider artificial neural networks, which are typically built by hierarchically (or mutually) combining neuron-like elements. Suppose that we want a network that behaves as we intend. Then, in the adaptive learning paradigm, we assume some structure of the target network, and repeatedly adjust some parameters in the network so that the network behaves well.

Suppose, for example, we assume the following structure for the target network.

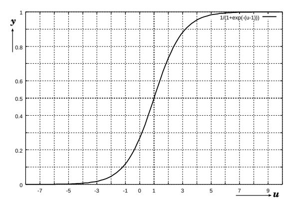




The output y of a neuron-like element is calculated as follows:

$$u = \Sigma_{i=1}^{2} x_i w_i - h$$

$$y = \frac{1}{1 + e^{-u}}$$



And suppose that we wish the network to behave as follows:

input		output
x_1	x_2	z
0	0	(nearly equal to) 0
0	1	(nearly equal to) 1
1	0	(nearly equal to) 1
1	1	(nearly equal to) 0

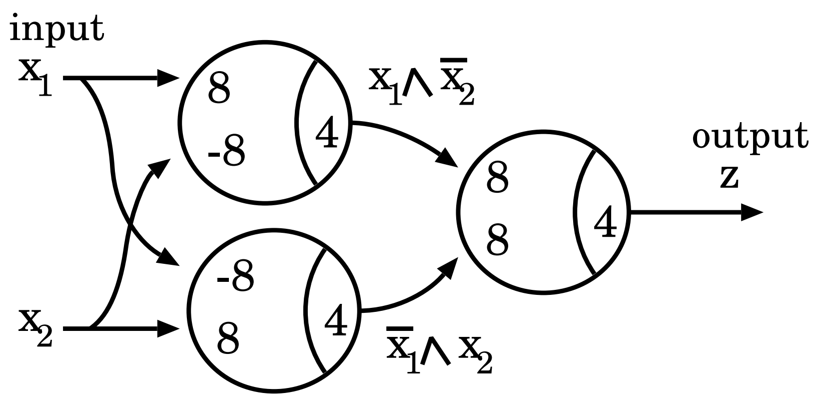
Exclusive OR function

⇨ **Then we repeat the following training operations:**

(1) for some given input assignment, we compare the output of the network with the desired output;

(2) if these are different, we adjust parameters w_{ij} and h_j so that the difference will be extinguished (or reduced).

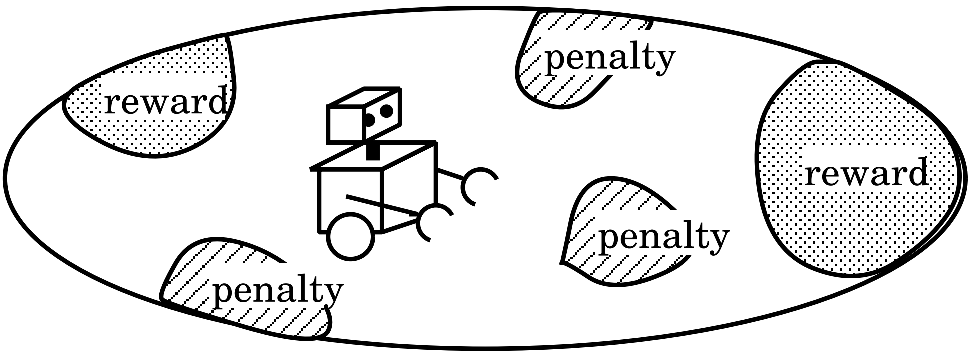
Finally we get a network like follows.



Example4 (Reinforcement Learning)

Consider a robot that works in some unknown environment. Initially the robot has no knowledge about the environment; but in response to his action, he receives feedback signal (e.g. reward or penalty). So, hopefully, from his experience he will gradually learn which action brings a good result.

In contrast with the case of artificial neural networks, nobody teaches the robot the correct action.

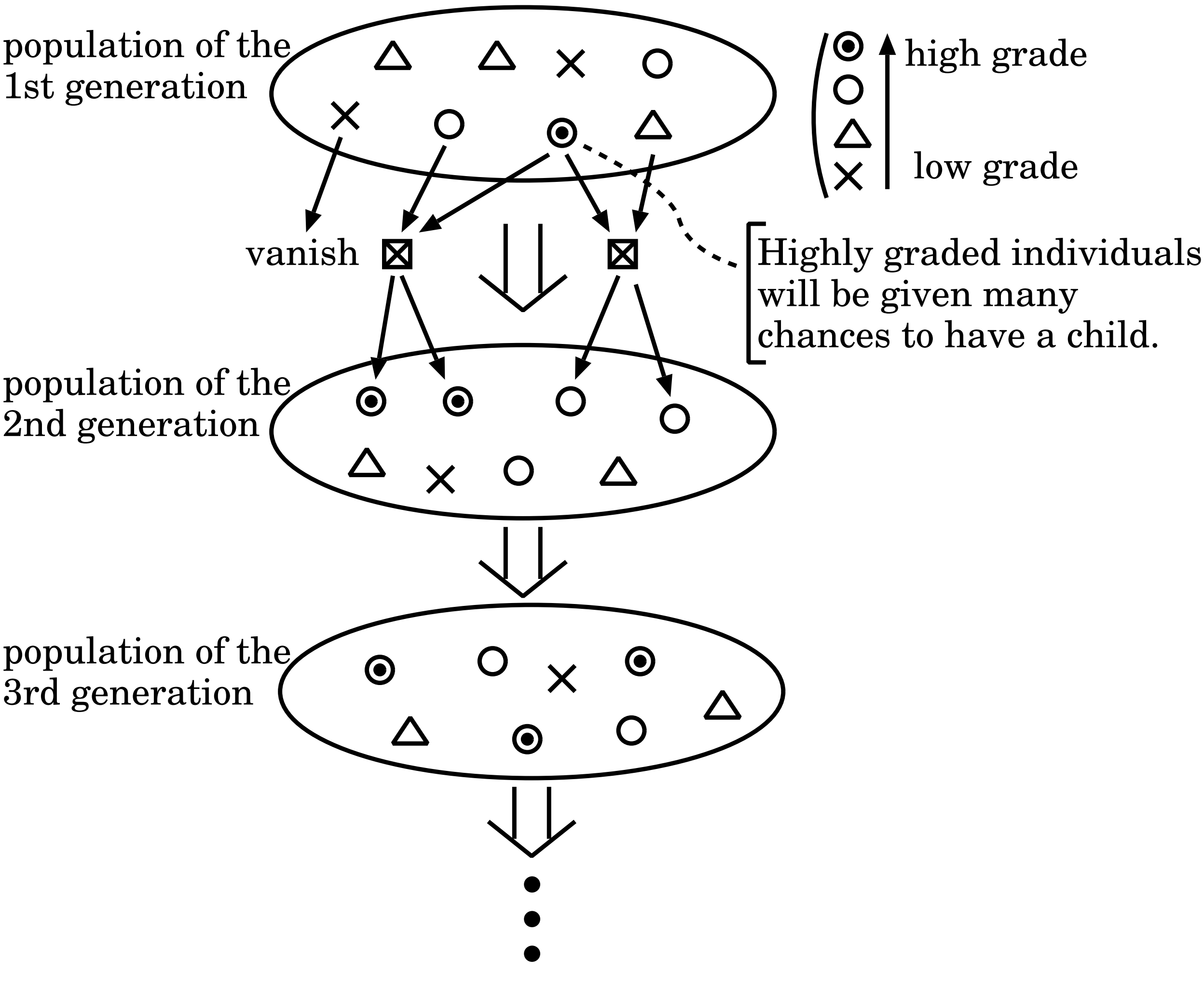


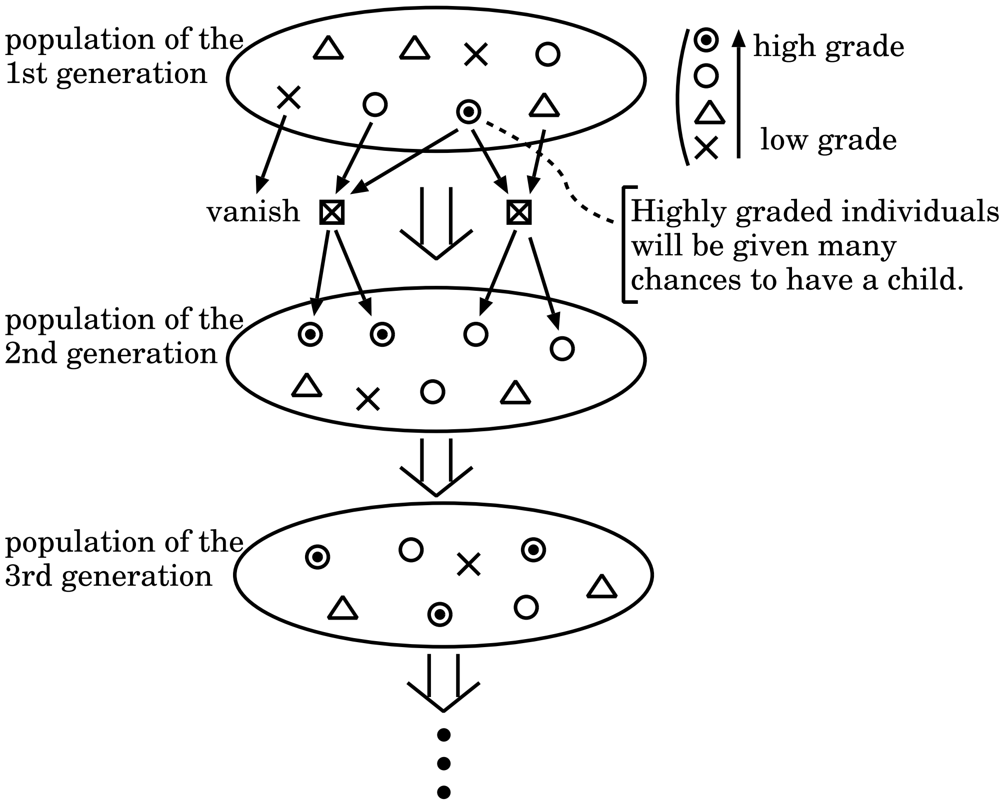
1–3 Evolutionary Learning

In the third learning paradigm, called evolutionary learning, we search a good solution based on the mechanics of natural selection and natural genetics; that is, we keep in mind how a population of animals and plants evolves, and search a good solution by alternately repeating two operations:

(1) evaluation of current search points (i.e. current candidates for good solution), and

(2) creation of new search points (candidates for good solution) from current-good search points.





- **We regard candidates for good solution as individuals in a populaion.**
- **In each time, we have several candidates for good solution as search points.**

- **Individuals of low grade will disappear in the next generation.**
- **Individuals of high grade will be utilized to create new individuals in the next generation.**

- **The creation of new individuals are usually accomplished by imitating real animal's crossover or mutation events.**

⇨ **Each individual does not learn, but the population of individuals learns.**

1–4 Summary

- **Inductive Learning**
  ⋯ **We construct something that explains given examples.**

- **Adaptive Learning**
  ⋯ **We repeatedly improve system parameters so that the behaviour of the system will be better.**

- **Evolutionary Learning**
  ⋯ **We search a good solution based on the mechanics of natural selection and natural genetics.**

In each learning paradigm, we fix the system model (e.g. decision tree, neural network, ...) that gives a search space, and try to find a good solution within that model.

This shows a difference from our inborn learning ability. We (human) does not restrict to a particular model.

II. Inductive Learning

2 Learning Decision Trees — C4.5 System —

References:
J.R.Quinlan(1993), C4.5: Programs for Machine Learning, Morgan Kaufmann.
P.H.Winston(1992), Artificial Intelligence 3rd Edition, Addison-Wesley.
S.Russell&P.Norvig(1995), Artificial Intelligence A Modern Approach, Prentice hall.

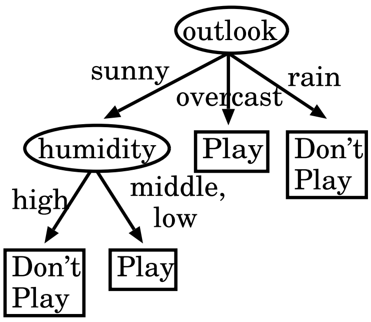
2–1 What sort of decision tree is desirable?

Example5 (Desirable Decision Tree)

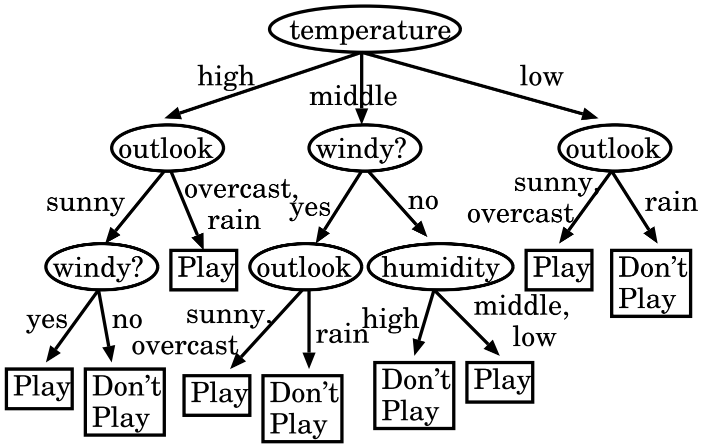
Suppose, for example, that we are given the following sample cases:

	attribute of case				resulting class
	outlook	temperature	humidity	windy?	
case1	sunny	middle	low	yes	Play
case2	sunny	high	high	no	Don't Play
case3	sunny	middle	high	no	Don't Play
case4	sunny	middle	middle	no	Play
case5	sunny	high	middle	yes	Play
case6	overcast	high	high	no	Play
case7	overcast	low	middle	yes	Play
case8	overcast	high	middle	no	Play
case9	rain	middle	high	yes	Don't Play
case10	rain	low	middle	yes	Don't Play
case11	rain	middle	high	no	Don't Play

Then, both the following decision trees can be used to classify the given cases.



Decision Tree A



Decision Tree B

Between these trees, decision Tree A seems more reasonable, because Decision Tree B is considered to contain many unessential tests.

Generally, we accept a principle, calld Occam's razor:

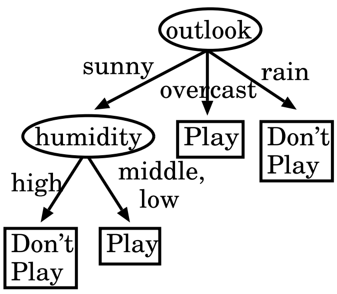
> **The world is inherently simple.**
> **Therefore the simplest decision tree that is consistent with the samples is the one that is most likely to classify unknown objects correctly.**

2–2 How do we build a desirable decision tree?

Unfortunately, finding the smallest decision tree is a computationarily intractable problem.

⇨ **We try to find a sufficiently small one with the following simple heuristics:**

- **Decision trees are built from root to leaves.**



**In decision trees,
the term "node" is used to denote a oval box that contains a test or a rectangle box that contains a decision,
the term "root" is used to denote the top node that will be check up first, and
the term "leaf" is used to denote a decision node that is depicted as a rectangle box.**

- **In each time, we choose the test that seems to make the tree smallest.**

- **The building process proceeds greedily;**

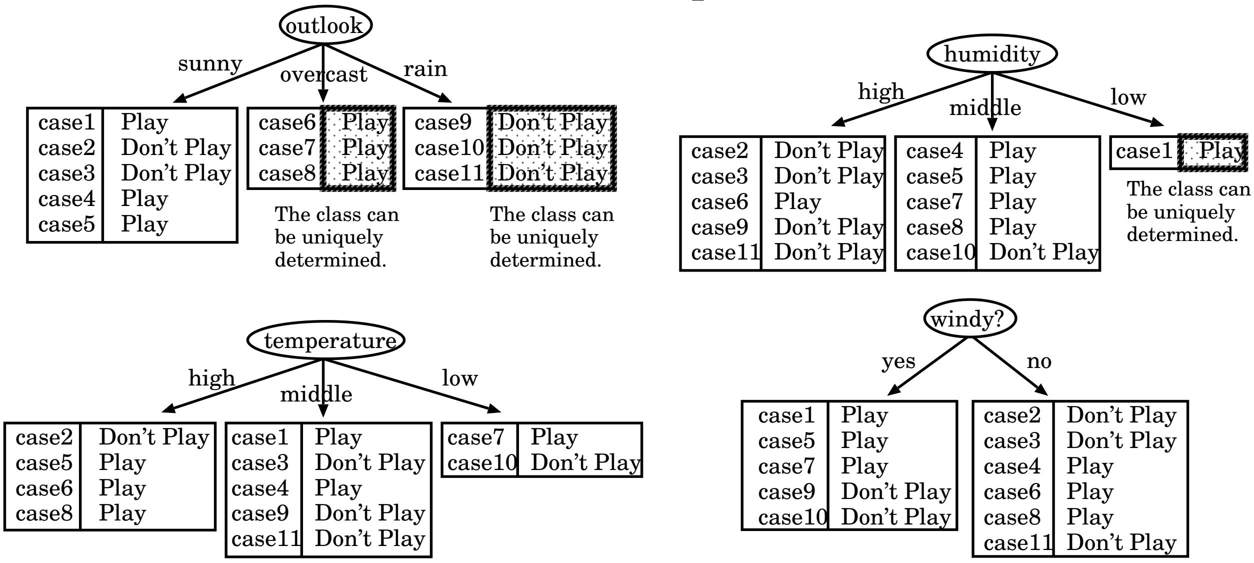
**that is,
once a test has been choosed to build a decision tree, that choice is fixed in the subsequent building process.**

- **In each node of tree under construction, we consider the set of sample cases that are distributed to that node by the above tests.**

Example6 (Building a Decision Tree)

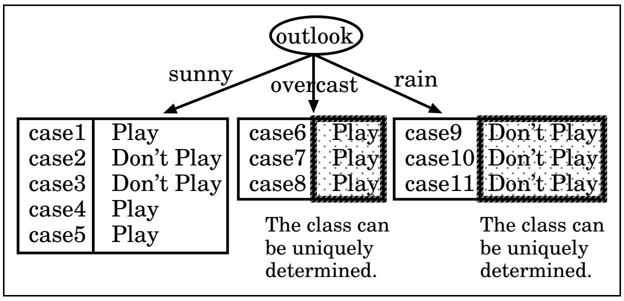
For the table of sample cases given in page 12, the above greedy algorithm works as follows:

Step1 **To choose the first test, we check up how each possible test divide the set of sample cases:**



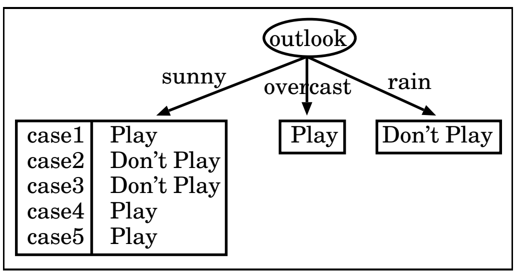
Among these, the outlook test seems to be best (i.e. seems to make the final tree smallest).

⇨ **We choose the outlook test; so we renew the tree under construction as follows:**



Step2 **Every node in which all distributed sample cases belong to the same class is replaced by an answering node identifying that class.**

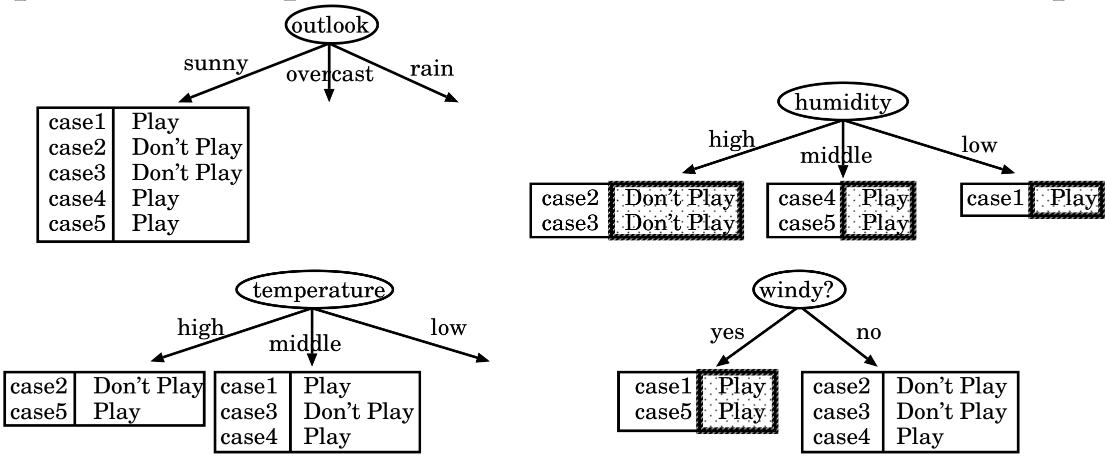
⇨ **We renew the tree under construction as follows:**



Step3 Now, it remains to build a subtree that correctly classifies 5 cases:

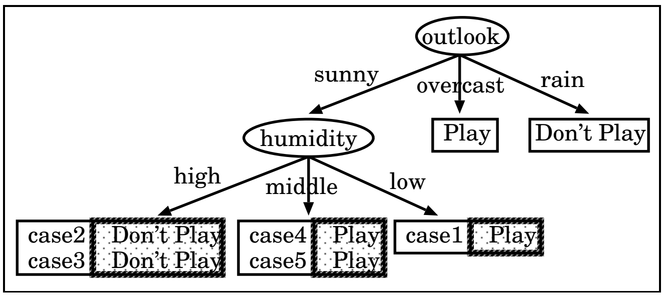
case1	Play
case2	Don't Play
case3	Don't Play
case4	Play
case5	Play

To choose the second test for this classification, we check up how each possible test divide the set of sample cases:



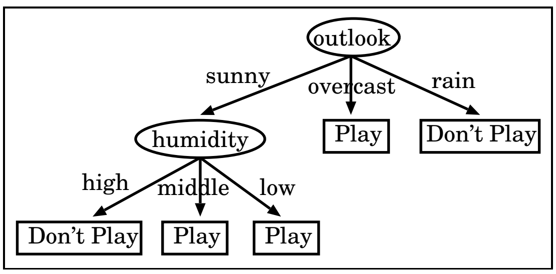
The humidity test seems to be best.

⇨ We choose the humidity test; so we renew the tree under construction as follows:



Step4 Every node in which all distributed sample cases belong to the same class is replaced by an answering node identifying that class.

⇨ We renew the tree under construction as follows:



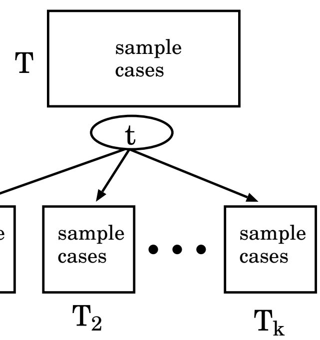
which is the resulting decision tree.

2–3 Which test should we next choose in building a decision tree?

When we saw some heuristics in the greedy tree-building algorithm, any criterion is not given for choosing a good test that seems to make the final tree smallest. Since it is generally unlike that we unhesitatingly choose a good test, we need a powerful way to valuate possible tests.

Among such ways, Quinlan(1979)'s one is well-known and illustrated below. But, there is not enough time to get into details on it right now.

⇨ **For evaluating a given test, Quinlan(1979) proposed a way to measure the total disorder in the subsets of sample cases produced by the test.**



Formally, given any test t that splits the set T of sample cases into $T_1, T_2, ..., T_k$, Quinlan defines a measure of disorder after that test:

$$\text{average\_disorder(t)} = \sum_{b=1}^{k} \left\{ \frac{|T_b|}{|T|} \sum_{c\ \text{:class}} \left(-\frac{\text{freq}(T_b, c)}{|T_b|} \log_2 \frac{\text{freq}(T_b, c)}{|T_b|} \right) \right\},$$

where

$|S|$ **denotes the cardinality of a set** S,
(i.e. the number of elements in S,**)**

$\text{freq}(T_b, c) =$ **the number of cases in** T_b **that belongs to class** c.

Then, he proposed to <u>choose the test that minimizes the amount average_disorder(t).</u>

Example7 (Quinlan's Measure of Disorder)

Looking back at Step1 in page 12, we have

$$
\begin{aligned}
&\textbf{average\_disorder(outlook)} \\
&= \frac{5}{11}\left(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}\right) \\
&\quad +\frac{3}{11}\left(-\frac{3}{3}\log_2\frac{3}{3}\right) \\
&\quad +\frac{3}{11}\left(-\frac{3}{3}\log_2\frac{3}{3}\right) \\
&= 0.4413411\cdots
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{average\_disorder(temperature)} \\
&= \frac{4}{11}\left(-\frac{3}{4}\log_2\frac{3}{4}-\frac{1}{4}\log_2\frac{1}{4}\right) \\
&\quad +\frac{5}{11}\left(-\frac{2}{5}\log_2\frac{2}{5}-\frac{3}{5}\log_2\frac{3}{5}\right) \\
&\quad +\frac{2}{11}\left(-\frac{1}{2}\log_2\frac{1}{2}-\frac{1}{2}\log_2\frac{1}{2}\right) \\
&= 0.9181694\cdots
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{average\_disorder(humidity)} \\
&= \frac{5}{11}\left(-\frac{1}{5}\log_2\frac{1}{5}-\frac{4}{5}\log_2\frac{4}{5}\right) \\
&\quad +\frac{5}{11}\left(-\frac{4}{5}\log_2\frac{4}{5}-\frac{1}{5}\log_2\frac{1}{5}\right) \\
&\quad +\frac{1}{11}\left(-\frac{1}{1}\log_2\frac{1}{1}\right) \\
&= 0.6562981\cdots
\end{aligned}
$$

$$
\begin{aligned}
&\textbf{average\_disorder(windy)} \\
&= \frac{5}{11}\left(-\frac{3}{5}\log_2\frac{3}{5}-\frac{2}{5}\log_2\frac{2}{5}\right) \\
&\quad +\frac{6}{11}\left(-\frac{3}{6}\log_2\frac{3}{6}-\frac{3}{6}\log_2\frac{3}{6}\right) \\
&= 0.9867956\cdots
\end{aligned}
$$

So, based on Quinlan's criterion, the outlook test is decided to be best as the root test.

(The second best is humidity test, the third best is temperature test, and the worst is windy test.)

2–4 Utilizing the C4.5 System

Prof. Quinlan developed a system, called C4.5, that builds decision trees by the above algorithm.

> **Ross Quinlan is a professor in University of Sydney.**
>
> ---
>
> **Besides building decision trees for given sample cases with discrete attribute, the C4.5 system works in various manners; e.g.**
>
> - **it can treat attributes with continuous values,**
> - **it can treat sample cases that has missing attribute values,**
> - **it can construct sets of production rules,**
> - **it can predict the accuracy of the resulting decision tree (or set of rules),**
> - **it can simplify decision trees by discarding one or more subtrees and replacing them with leaves,**
> - **it can run interactively,**
> - **..............................**

According to the C4.5 system,

- **a book(1993) that contains the source code (about 8800 lines) and user's guide, and**
- **a diskette that contains the source code,**

are distributed by Morgan Kaufmann Publishers.

⇨ **We will see how to utilize that system.**

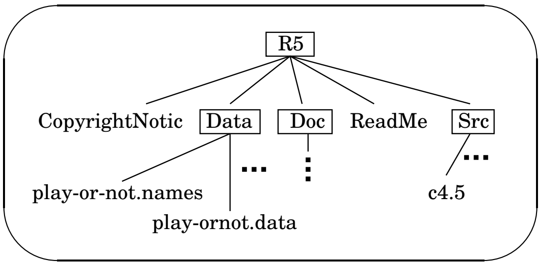
Example8 (Utilizing the C4.5 system)

For building a decision tree from the table of sample cases in page 12, we can utilize the C4.5 system as follows:

Step1 **We give a brief name “play-or-not” to the task.**

Step2 **We construct a text file “play-or-not.names” that defines the possible classes and attributes.**

```
[motoki@x205a]$ cd R5                          change the current directory
/home/motoki/R5
[motoki@x205a]$ ls                             display file names in the current directory
CopyrightNotic  Data/  Doc/  ReadMe  Src/
[motoki@x205a]$ cd Data                        change the current directory
/home/motoki/R5/Data
[motoki@x205a]$ cat play-or-not.names          display contents in the file "play-or-not.names"
play, don't play.

outlook:     sunny, overcast, rain.
temperature: high, middle, low.
humidity:    high, middle, low.
windy:       yes, no.
[motoki@x205a]$
```



Step3 **We construct a text file “play-or-not.data” that describes the sample cases from which a decision tree will be built.**

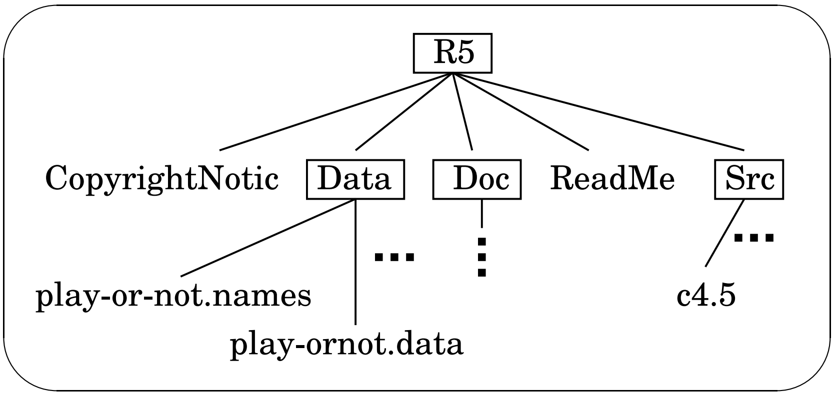
```
[motoki@x205a]$ pwd                            print the name of the current working directory
/home/motoki/R5/Data
[motoki@x205a]$ cat play-or-not.data           display contents in the file "play-or-not.data"
sunny,    middle, low,    yes, play.
sunny,    high,   high,   no,  don't play.
sunny,    middle, high,   no,  don't play.
sunny,    middle, middle, no,  play.
sunny,    high,   middle, yes, play.
overcast, high,   high,   no,  play.
overcast, low,    middle, yes, play.
overcast, high,   middle, no,  play.
rain,     middle, high,   yes, don't play.
rain,     low,    middle, yes, don't play.
rain,     middle, high,   no,  don't play.
[motoki@x205a]$
```

Step4 If we do not yet complete the compilation&linkage of the source code, we accomplish this:

```
[motoki@x205a]$ cd ../Src            change directory; ".." means the parent directory
/home/motoki/R5/Src
[motoki@x205a]$ ls                   display file names in the current directory
Makefile       c4.5rules.c  discr.c     header.c     siftrules.c  types.i
Modifications  classify.c   extern.i    info.c       sort.c       userint.c
average.c      confmat.c    genlogs.c   makerules.c  st-thresh.c  xval-prep.c
besttree.c     consult.c    genrules.c  prune.c      stats.c      xval.sh
build.c        consultr.c   getdata.c   prunerule.c  subset.c
buildex.i      contin.c     getnames.c  rules.c      testrules.c
c4.5.c         defns.i      getopt.c    rulex.i      trees.c
[motoki@x205a]$ make all             compile related source codes and link them
make c4.5
make[1]: 入ります ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Enter to the directory '/home/motoki/R5/Src')
cc -O2 -c c4.5.c
cc -O2 -c besttree.c
cc -O2 -c build.c
cc -O2 -c info.c
cc -O2 -c discr.c
cc -O2 -c contin.c
cc -O2 -c subset.c
cc -O2 -c prune.c
cc -O2 -c stats.c
cc -O2 -c st-thresh.c
cc -O2 -c classify.c
cc -O2 -c confmat.c
cc -O2 -c sort.c
cc -O2 -c getnames.c
getnames.c: In function 'GetNames':
getnames.c:151: warning: assignment makes pointer from integer without a cast
getnames.c:176: warning: assignment makes pointer from integer without a cast
getnames.c:193: warning: assignment makes pointer from integer without a cast
getnames.c:213: warning: cast to pointer from integer of different size
getnames.c: At top level:
getnames.c:268: warning: type mismatch with previous implicit declaration
getnames.c:176: warning: previous implicit declaration of 'CopyString'
getnames.c:268: warning: 'CopyString' was previously implicitly declared to return 'int'
cc -O2 -c getdata.c
cc -O2 -c trees.c
cc -O2 -c getopt.c
cc -O2 -c header.c
cc -o c4.5 c4.5.o besttree.o build.o info.o discr.o contin.o subset.o prune.o stats.o st-thresh
make[1]: 出ます ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Exit from the directory '/home/motoki/R5/Src')
make c4.5rules
make[1]: 入ります ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Enter to the directory '/home/motoki/R5/Src')
cc -O2 -c c4.5rules.c
c4.5rules.c:35: warning: data definition has no type or storage class
cc -O2 -c rules.c
cc -O2 -c genlogs.c
cc -O2 -c genrules.c
```

```
cc -O2 -c makerules.c
cc -O2 -c prunerule.c
cc -O2 -c siftrules.c
cc -O2 -c testrules.c
cc -o c4.5rules c4.5rules.o rules.o genlogs.o genrules.o makerules.o prunerule.o siftrules.o te
make[1]: 出ます ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Exit from the directory '/home/motoki/R5/Src')
make consult
make[1]: 入ります ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Enter to the directory '/home/motoki/R5/Src')
cc -O2 -c consult.c
cc -O2 -c userint.c
cc -o consult consult.o userint.o getnames.o getdata.o trees.o getopt.o header.o -lm
make[1]: 出ます ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Exit from the directory '/home/motoki/R5/Src')
make consultr
make[1]: 入ります ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Enter to the directory '/home/motoki/R5/Src')
cc -O2 -c consultr.c
cc -o consultr consultr.o rules.o userint.o getnames.o getdata.o trees.o getopt.o header.o -lm
make[1]: 出ます ディレクトリ '/home/motoki/R5/Src'
        (***Comment*** Exit from the directory '/home/motoki/R5/Src')
cc -o xval-prep xval-prep.c -lm
cc -o average average.c -lm
[motoki@x205a]$ ls                        (display file names in the current directory)
Makefile       c4.5rules.c  contin.o     getopt.c     rules.o      testrules.o
Modifications  c4.5rules.o  defns.i      getopt.o     rulex.i      trees.c
average*       classify.c   discr.c      header.c     siftrules.c  trees.o
average.c      classify.o   discr.o      header.o     siftrules.o  types.i
besttree.c     confmat.c    extern.i     info.c       sort.c       userint.c
besttree.o     confmat.o    genlogs.c    info.o       sort.o       userint.o
build.c        consult*     genlogs.o    makerules.c  st-thresh.c  xval-prep*
build.o        consult.c    genrules.c   makerules.o  st-thresh.o  xval-prep.c
buildex.i      consult.o    genrules.o   prune.c      stats.c      xval.sh
c4.5*          consultr*    getdata.c    prune.o      stats.o
c4.5.c         consultr.c   getdata.o    prunerule.c  subset.c
c4.5.o         consultr.o   getnames.c   prunerule.o  subset.o
c4.5rules*     contin.c     getnames.o   rules.c      testrules.c
[motoki@x205a]$
```

Step5 We invoke the executable program code c4.5 to build a decision tree:




```
[motoki@x205a]$ pwd
```

print the name of the current working directory

```
/home/motoki/R5/Data
[motoki@x205a]$ ls
```

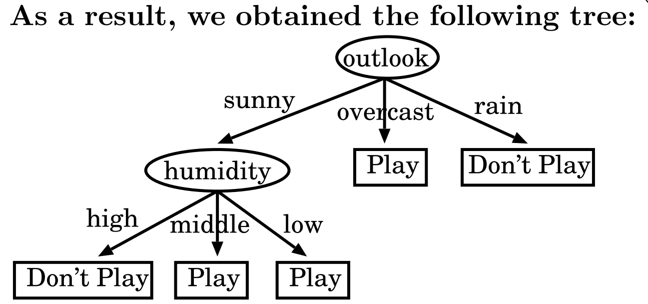
display file names in the current directory

```
crx.data     golf.unpruned    labor-neg.test  monk2.test        soybean.data
crx.names    hypo.data        monk1.data      monk3.data        soybean.names
crx.test     hypo.names       monk1.names     monk3.names       vote.data
golf.data    hypo.test        monk1.test      monk3.test        vote.names
golf.names   labor-neg.data   monk2.data      play-or-not.data  vote.test
golf.tree    labor-neg.names  monk2.names     play-or-not.names
[motoki@x205a]$ ../Src/c4.5 -f play-or-not
```

invoke the executable code "c4.5"

```
C4.5 [release 5] decision tree generator Mon Nov 12 10:11:26 2001
----------------------------------------

    Options:
        File stem <play-or-not>

Read 11 cases (4 attributes) from play-or-not.data

Decision Tree:

outlook = overcast: play (3.0)
outlook = rain: don't play (3.0)
outlook = sunny:
|   humidity = high: don't play (2.0)
|   humidity = middle: play (2.0)
|   humidity = low: play (1.0)


Tree saved
```

As a result, we obtained the following tree:




```
Evaluation on training data (11 items):

         Before Pruning           After Pruning
        ----------------   ---------------------------
        Size      Errors   Size      Errors   Estimate

           7    0( 0.0%)      7    0( 0.0%)    (45.2%)   <<
[motoki@x205a]$
```

Remark:
To avoid that the decision tree with little predictive power is built, the C4.5 system require in default that any test used in the tree must have at least two branches with 2 or more cases.
Therefore, the C4.5 system does not necessarily build a decision tree that classifies all given sample cases. For example, if case5 did not belong to the table of sample cases in page 12, the C4.5 system would work as follows:

```
[motoki@x205a]$ ../Src/c4.5 -f play-or-not

C4.5 [release 5] decision tree generator  Mon Nov 12 14:38:31 2001
----------------------------------------


    Options:
        File stem <play-or-not>

Read 10 cases (4 attributes) from play-or-not.data

Decision Tree:

outlook = sunny: play (4.0/2.0)
outlook = overcast: play (3.0)
outlook = rain: don't play (3.0)


Tree saved


Evaluation on training data (10 items):

         Before Pruning           After Pruning
        ----------------   ---------------------------
        Size      Errors   Size      Errors   Estimate
           4   2(20.0%)       4    2(20.0%)    (53.0%)   <<
[motoki@x205a]$
```

III. Evolutionary Computation

Today,

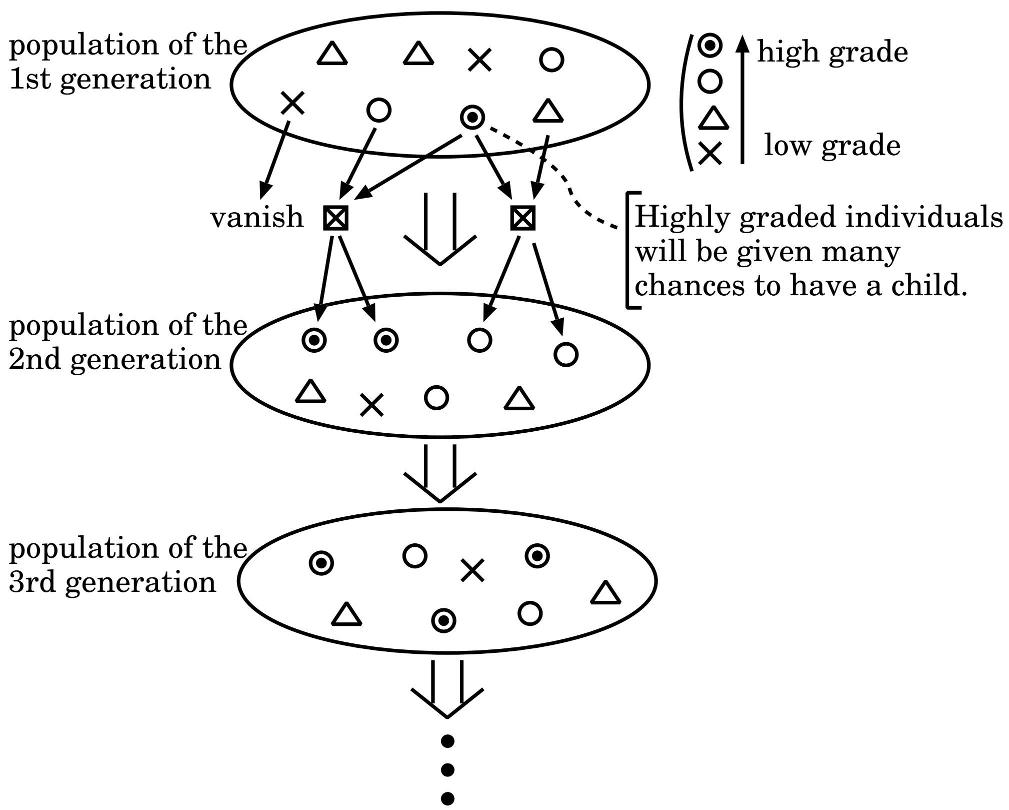
I'm going to share with you the basic feature of evolutionary learning, through optimization of a simple function of one variable, and next through search for an arithmetic expression that fits the given input-output relations.

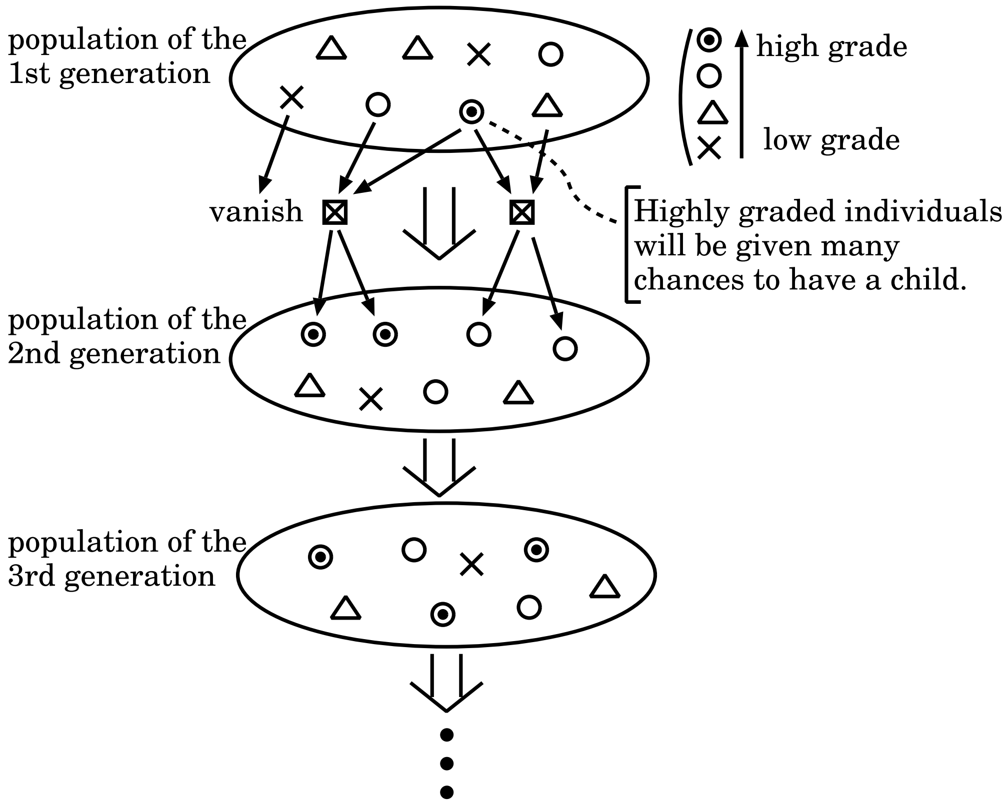
At the beginning, let us review a last talk on evolutionary learning.

In the evolutionary learning paradigm, we search a good solution based on the mechanics of natural selection and natural genetics, that is, we keep in mind how a population of animals and plants evolves, and search a good solution by alternately repeating two operations:

(1) valuation of current search points (i.e. current candidates for good solution), and

(2) creation of new search points (candidates for good solution) from current-good search points.





Getting into details,

- **we regard candidates for good solution as individuals in a populaion.**
- **In each time, we have several candidates for good solution as search points.**

- **Individuals of low grade will disappear in the next generation.**
- **Individuals of high grade will be utilized to create new individuals in the next generation.**

- **The creation of new individuals are usually accomplished by imitating real animal's crossover or mutation events.**

3 We can evolutionarily find x that maximizes x*sin(10*pi*x)+1.

References:
Z.Michalewictz(1994), Genetic Algorithms + Data Structures = Evolution Programs Second Edition, Springer-Verlag.

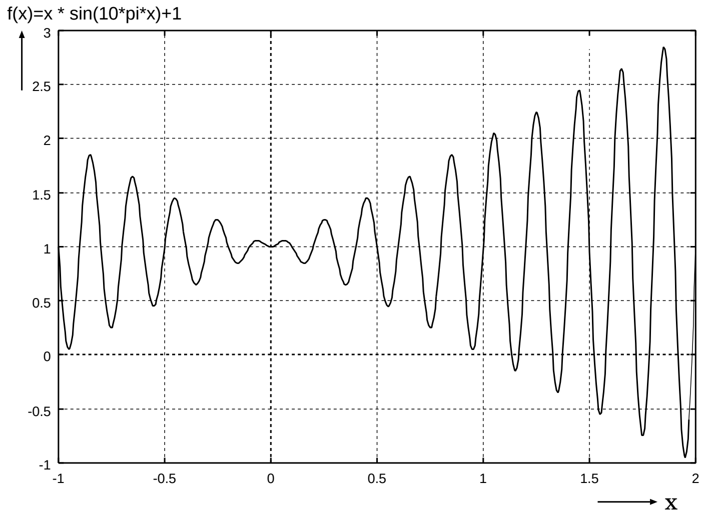
First, we will see how we can apply the evolutionary search scheme to a simple problem.

3–1 A Problem of Optimizing a Simple Function of One Variable

Consider

A Function Optimization Problem:
Find such a value of x from the range $[-1, 2]$ that maximize the function $f(x) = x\sin(10\pi x) + 1$.

As a matter of fact, this function behaves as follows:

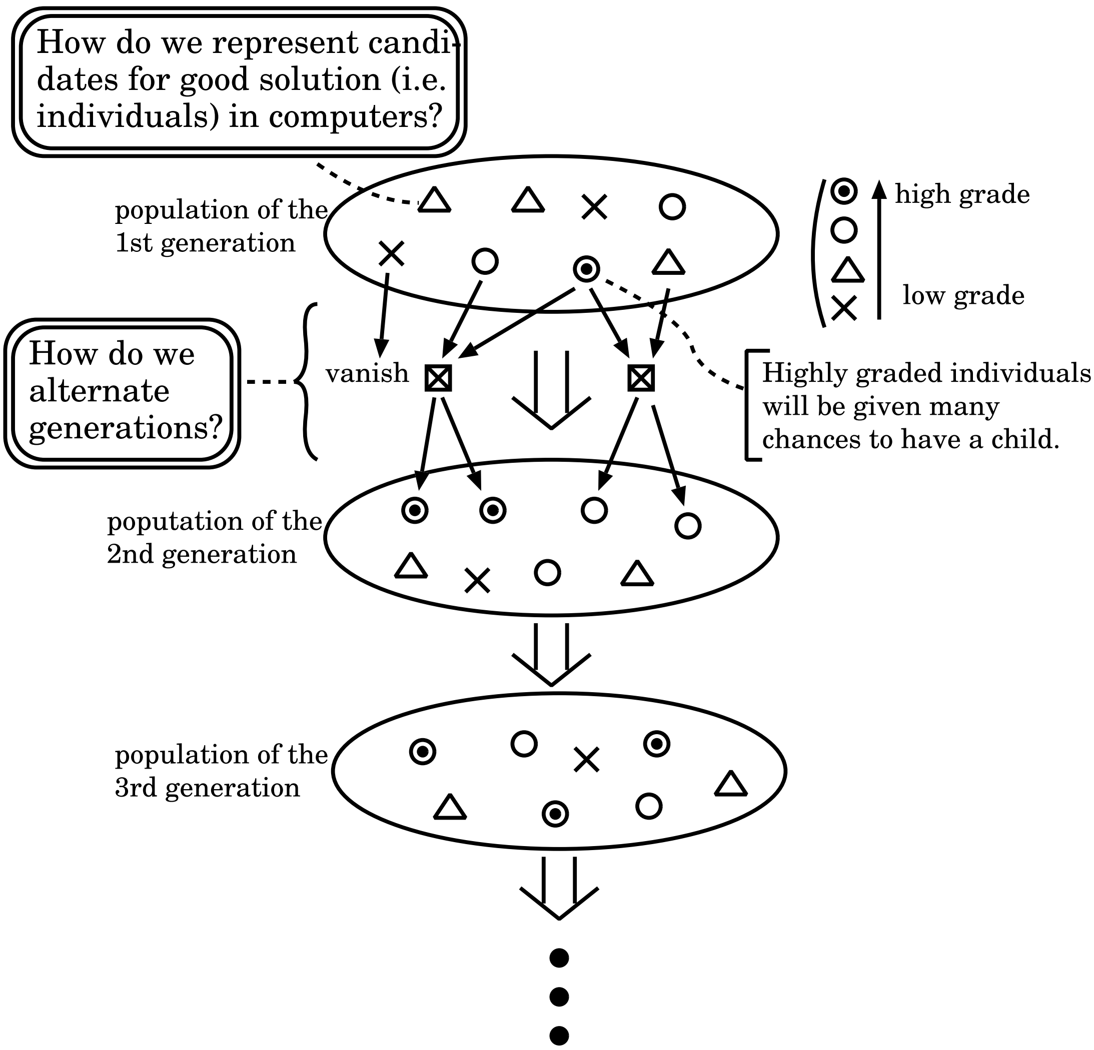


⇨ **In the domain $[-1, 2]$, the function f reaches its maximum $f(x) = 2.850274\cdots$ for $x = 1.850542\cdots$.**

3–2 How do we evolutionarily search x that maximize the function f(x)=x*sin(10*pi*x)+1

To follow the search scheme reviewed a little while ago, we must clarify

- how to represent candidates for good solution, and
- how to alternate generations.



⇨ About these issues, we follow the scheme in the Michalewictz's book (pp.18-22), although we can choose many alternatives.

3–3 A Method for Representing Candidates for Good Solution

We are to find a real number in an interval $[-1, 2]$, but we cannot encode infinitely many objects by a fixed sized memory. So, we discretize the search space $[-1, 2]$ by dividing into many equally-sized small intervals.

⇨ • **We search a finite space**

$$\left\{-1 + \frac{3}{2^{22}-1}k \,\middle|\, k \text{ is an integer, } 0 \leq k \leq 2^{22}-1\right\},$$

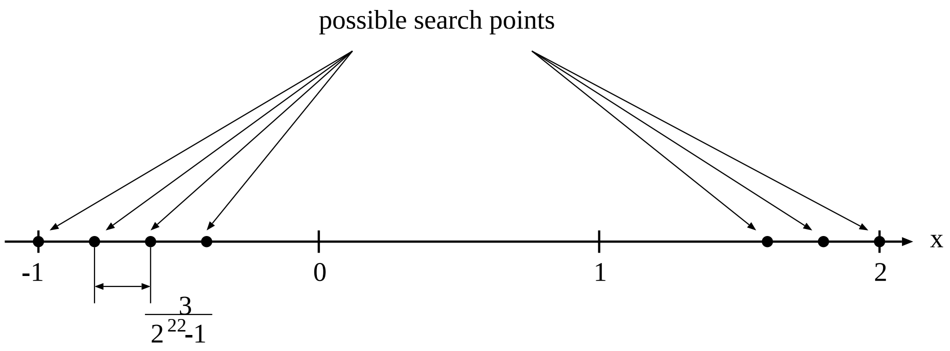
instead of an infinite space $[-1, 2]$.

• **We use a binary string**

$$b_{21}b_{20}b_{19}\cdots b_1 b_0$$

as an individual to represent a search point

$$x = -1 + \frac{3}{2^{22}-1}\left(\sum_{i=0}^{21} b_i \times 2^i\right) \quad \in [-1, 2].$$

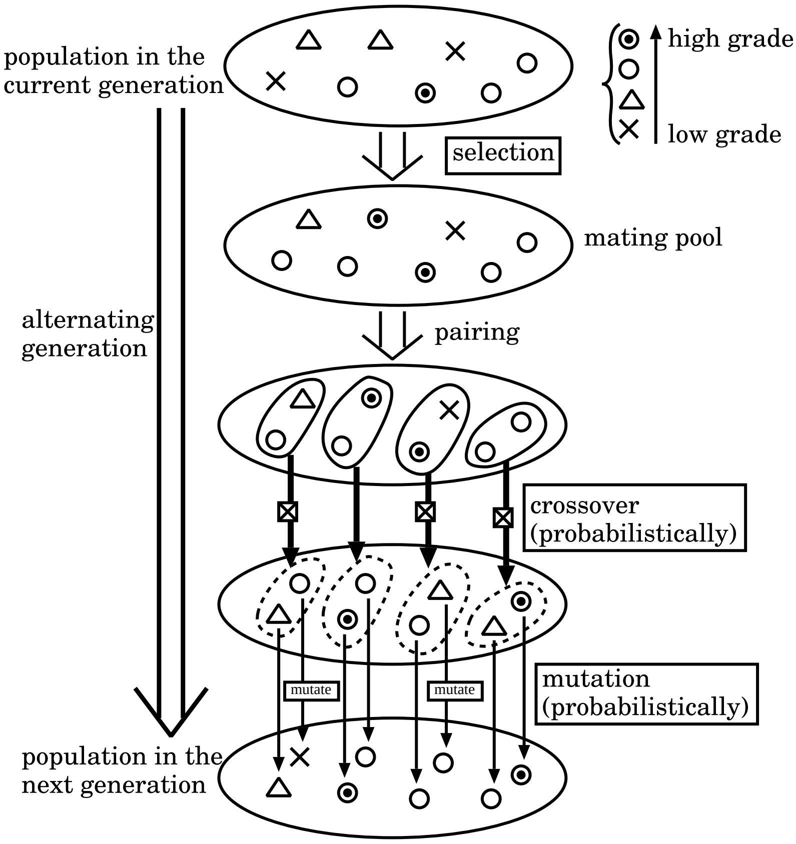


3–4 A Method for Creating Initial Individuals

All 22 bits for each individual are randomly initialized.

3–5 A Method for Alternating Generations

We alternate generations as follows:



Given a population of individuals in the current generation, we first <u>select</u> good individuals that are utilized to create new individuals. For this selection operation, we need a measure of goodness of individuals, called fitness.

Fitness

Now, since we are to find the value of $x = -1 + \frac{3}{2^{22}-1}\left(\sum_{i=0}^{21} b_i \times 2^i\right)$ **that maximize the function** $f(x)$**, we consider that**

$$\begin{aligned} &\textbf{the fitness of individual } b_{21}b_{20}\cdots b_1b_0 \\ &\quad = f(\textbf{the search point that } b_{21}b_{20}\cdots b_1b_0 \textbf{ represent}) \\ &\quad = f(x) \end{aligned}$$

So in this problem, the larger the fitness, the better the individual.

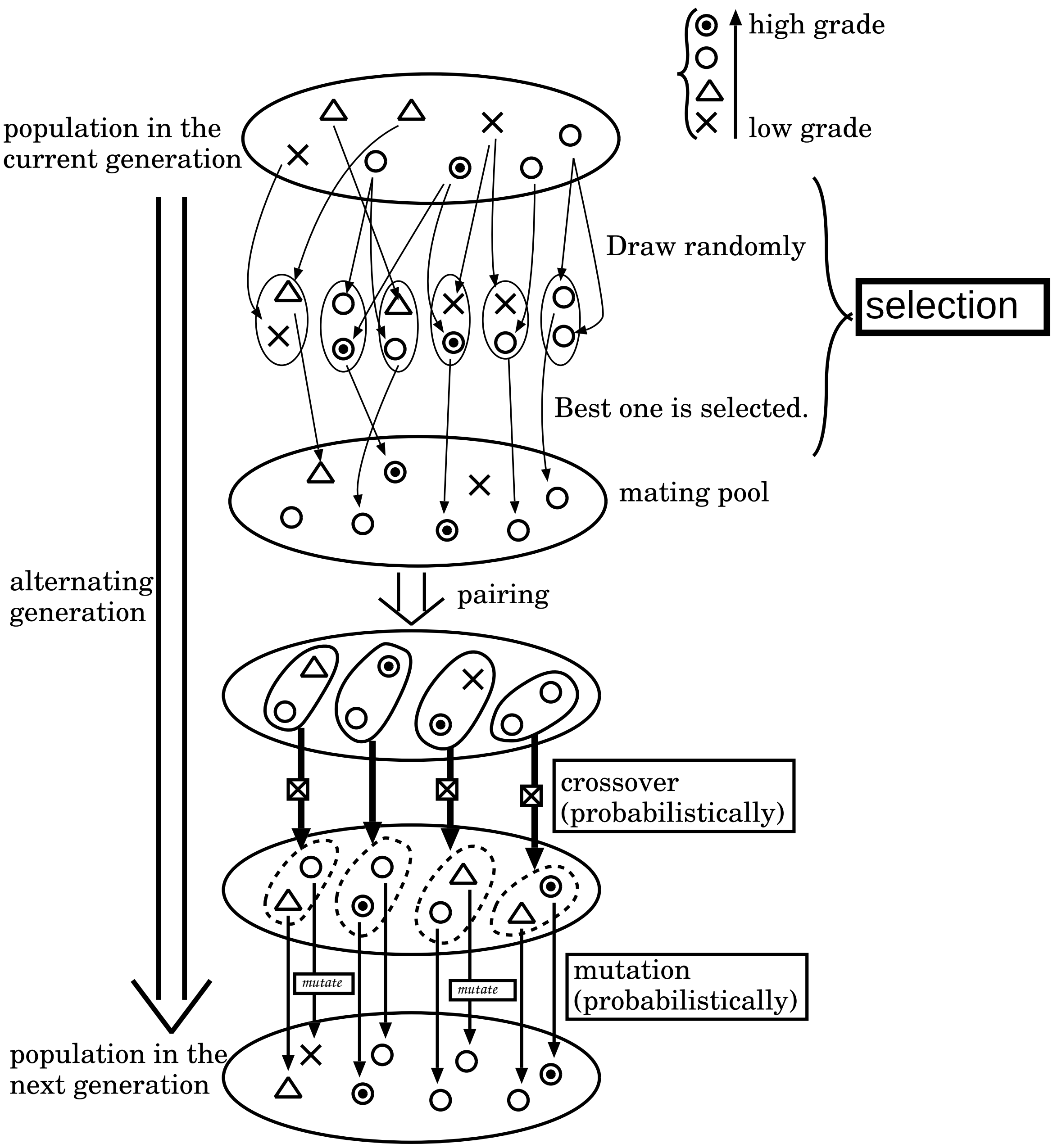
Selection

Based on the fitness, we repeat the following selection process so many times as is equal to the population size:

(1) Two individuals are randomly drawn from the population with replacement.

(2) The best of them is selected into the mating pool.

tournament selection



After the selection process, we next create new individuals from the selected individuals. Such creation operations are accomplished by imitating real animal's crossover and mutation events; so we also call such operations crossover and mutation, respectively.

Crossover

Given two individuals (called parents), we perform the following operations with some probability (called crossover rate):

(1) An integer k is randomly chosen from {1, 2, 3, ..., 21}.

(2) Swap final k bits of parents inclusively to generate two new individuals (called children).

one-point crossover

parents

00000|01110000000010000
11100|00000111111000101

↑ $k = 17$ (crossover point)

⇒ crossover

children

00000|00000111111000101
11100|01110000000010000

Mutation

Given an individual, we independently flip each bit in it with a low probability (called mutation rate).

11100000000111111000101
↓mutate
11101000000111111000101

Flip bitwisely with a low probability.

3–6 Implementation in Fortran77

We now give a Fortran77 computer code that implements the evolutionary search procedure described earlier, where

population_size	=	**50,**
maximum_generation_number	=	**200,**
crossover_rate	=	**0.25 and**
mutation_rate	=	**0.01.**

```
xcspc60_41% cat applying_GA_to_optimize_func.f  (display the Fortran77 source code)
*************************************************************************
* A Fortran77 program that implements a GA search of                     *
*               finding such a value of x from the range [-1,2] that     *
*               maximize the function f(x)=x*sin(10*pi*x)+1.             *
*-----------------------------------------------------------------------*
* Note: (1)We use lower 22 bits of an INTEGER-type variable b (or array   *
*          element) as an individual to represent a search point         *
*                  x=-1+(b[0]+b[1]*2+...+b[21]*2**21)*3/(2**22-1),       *
*          where b[i] denotes an (i+1)-th lowest bit.                    *
*       (2)All 22 bits for each individual are randomly initialized.     *
*       (3)We fix the population size to be 50, and the maximum generation*
*          number to be 200.                                             *
*       (4)we use the tournament selection with tournament size 2.       *
*       (5)We use the one-point crossover with crossover rate 0.25.      *
*       (6)As a mutation operation, we independently flip each bit in a  *
*          given individual with probability 0.01.                       *
*       (7)For simplicity, we use a pseudo-random generator that can be  *
*          disasterous in many circumstances.                            *
*          [Reference: W.H.Press, et al.(1992), Numerical Recipes in C,  *
*                      Cambridge University Press.                       *
*                      M.Matsumoto&T.Nishimura(1998), Mersenne Twister: A *
*                      623-dimensionally Equidistributed Uniform         *
*                      Pseudorandom Number Generator, ACM Trans. on Model.*
*                      Comput. Simul., Vol.8, pp.3-30.]                  *
*************************************************************************
      PROGRAM GA_OPT
      INTEGER POPSIZE, MAXGEN
      REAL    CROSS_RATE, MUT_RATE
      PARAMETER (POPSIZE=50, MAXGEN=200, CROSS_RATE=0.25, MUT_RATE=0.01)
      INTEGER SELECT, RAND_INT,
     &        IND(1:POPSIZE), MATE_POOL(1:POPSIZE), GEN, BESTIND, I, K
      REAL    F, RAND_REAL,
```

```
     &       FITNESS(1:POPSIZE), BESTSOFAR

* Initialize a Pseudo-random Generator
      CALL RAND_INITIALIZE( )

* Create Initial Individuals
      DO 20 K=1, POPSIZE
         IND(K)=0
         DO 10 I=0,21
            IND(K)=IND(K)*2+RAND_INT(2)
   10    CONTINUE
         FITNESS(K)=F(-1.0+3.0/REAL(2**22-1)*IND(K))
   20 CONTINUE

      WRITE(*,100)
      BESTSOFAR=-100.0
      CALL OBSERVE(IND, FITNESS, POPSIZE, 0, BESTSOFAR)

* Evolutionary Simulation
      DO 80 GEN=1,MAXGEN
*        -------Selection-------
         DO 30 K=1,POPSIZE
            MATE_POOL(K)=IND(SELECT(FITNESS, POPSIZE))
 30      CONTINUE
*        -------Crossover-------
         DO 40 K=1,POPSIZE,2
            IF (RAND_REAL( ).LE.CROSS_RATE) THEN
               CALL CROSSOVER(MATE_POOL(K), MATE_POOL(K+1))
            END IF
   40    CONTINUE
*        -------Mutation-------
         DO 60 K=1,POPSIZE
            DO 50 I=0,21
               IF (RAND_REAL( ).LE.MUT_RATE) THEN
                  CALL FLIP(MATE_POOL(K), I)
               END IF
   50       CONTINUE
   60    CONTINUE
*        -------Alternate Generation and Evaluate-------
         DO 70 K=1,POPSIZE
            IND(K)=MATE_POOL(K)
            FITNESS(K)=F(-1.0+3.0/REAL(2**22-1)*IND(K))
   70    CONTINUE
         CALL OBSERVE(IND, FITNESS, POPSIZE, GEN, BESTSOFAR)
   80 CONTINUE

  100 FORMAT(' ------------------------------------------------'/
```

```
     & ' We now apply genetic algorithm to the problem of'/
     & ' finding such a value of x from the range [-1,2]'/
     & ' that maximize the function f(x)=x*sin(10*pi*x)+1.'/
     & ' -----------------------------------------------'//
     & ' Gen.#        Best_ind         (its_val)  Best_x   Best_f(x)'
     &                                               , ' Average_f(x)'/
     & ' -----  ----------------------  -------  ---------  --------- '
     &                                               , ' --------')
      END

*-------------------------------------------------------------------------*
* A target function to be maximized                                       *
*-------------------------------------------------------------------------*
      REAL FUNCTION F(X)
      REAL PI
      PARAMETER (PI=3.141593)
      REAL X

      F=X*sin(10.0*PI*X)+1.0

      END

*-------------------------------------------------------------------------*
* A function that selects an individual number                            *
*                by the tournement selection of tournament size 2         *
*-------------------------------------------------------------------------*
      INTEGER FUNCTION SELECT(FITNESS, POPSIZE)
      INTEGER POPSIZE, RAND_INT, CANDIDATE1, CANDIDATE2,
     &        CHECKCOUNT
      DATA    CHECKCOUNT/0/
      SAVE    CHECKCOUNT
      REAL    FITNESS(1:POPSIZE)
      CHARACTER REL*2

      CANDIDATE1=RAND_INT(POPSIZE)+1
      CANDIDATE2=RAND_INT(POPSIZE)+1
      IF (FITNESS(CANDIDATE1).GE.FITNESS(CANDIDATE2)) THEN
         SELECT=CANDIDATE1
      ELSE
         SELECT=CANDIDATE2
      END IF

*     -----Check whether the SELECT routine works-----
      IF (CHECKCOUNT.LT.3) THEN
         CHECKCOUNT=CHECKCOUNT+1
         IF (FITNESS(CANDIDATE1).GE.FITNESS(CANDIDATE2)) THEN
            REL='>='
```

```
         ELSE
            REL='< '
         END IF
         WRITE(*,100) POPSIZE, CANDIDATE1, FITNESS(CANDIDATE1), REL,
     &                FITNESS(CANDIDATE2), CANDIDATE2, SELECT
      END IF

 100  FORMAT(' (Check the routine SELECT(FITNESS, popsize=', I2, '):'/
     &       '   | FITNESS(', I2, ')=', F9.6, 1X, A2, 1X,
     &       F9.6, '=FITNESS(', I2, ') ==> We select the ', I2,
     &       '-th individual.)')

      END

*-------------------------------------------------------------------------*
* A subroutine that recombinates given two parent individuals             *
*                 by one-point crossover                                  *
*-------------------------------------------------------------------------*
      SUBROUTINE CROSSOVER(IND1, IND2)
      INTEGER IND1, IND2, RAND_INT, TAIL1, TAIL2, CROSSPOINT, I,
     &        CHECKCOUNT, BIT_IND1(0:21), BIT_IND2(0:21),
     &        CHILD1, CHILD2, BIT_CHILD1(0:21), BIT_CHILD2(0:21),
     &        NUM1, NUM2, NUM_CHILD1, NUM_CHILD2
      DATA    CHECKCOUNT/0/
      SAVE    CHECKCOUNT
      CHARACTER*1 CHAR_CODE(0:1)
      DATA    CHAR_CODE/'0', '1'/

      CROSSPOINT=RAND_INT(21)+1
      TAIL1=MOD(IND1, 2**CROSSPOINT)
      TAIL2=MOD(IND2, 2**CROSSPOINT)
      CHILD1=IND1-TAIL1+TAIL2
      CHILD2=IND2-TAIL2+TAIL1

*     -----Check whether the CROSSOVER routine works-----
      IF (CHECKCOUNT.LT.3) THEN
         CHECKCOUNT=CHECKCOUNT+1
         NUM1=IND1
         NUM2=IND2
         NUM_CHILD1=CHILD1
         NUM_CHILD2=CHILD2
         DO 10 I=0,21
            BIT_IND1(I)=MOD(NUM1,2)
            BIT_IND2(I)=MOD(NUM2,2)
            BIT_CHILD1(I)=MOD(NUM_CHILD1,2)
            BIT_CHILD2(I)=MOD(NUM_CHILD2,2)
            NUM1=NUM1/2
```

```
            NUM2=NUM2/2
            NUM_CHILD1=NUM_CHILD1/2
            NUM_CHILD2=NUM_CHILD2/2
   10    CONTINUE
         WRITE(*,100) IND1, IND2, CROSSPOINT,
     &      (CHAR_CODE(BIT_IND1(I)),I=21,CROSSPOINT,-1), ' ',
     &         (CHAR_CODE(BIT_IND1(I)),I=CROSSPOINT-1,0,-1),
     &      (CHAR_CODE(BIT_CHILD1(I)),I=21,CROSSPOINT,-1), ' ',
     &         (CHAR_CODE(BIT_CHILD1(I)),I=CROSSPOINT-1,0,-1),
     &      (CHAR_CODE(BIT_IND2(I)),I=21,CROSSPOINT,-1), ' ',
     &         (CHAR_CODE(BIT_IND2(I)),I=CROSSPOINT-1,0,-1),
     &      (CHAR_CODE(BIT_CHILD2(I)),I=21,CROSSPOINT,-1), ' ',
     &         (CHAR_CODE(BIT_CHILD2(I)),I=CROSSPOINT-1,0,-1),
     &      (' ',I=21,CROSSPOINT,-1), '^',
     &         (' ', I=CROSSPOINT-1,0,-1),
     &      (' ',I=21,CROSSPOINT,-1), '^',
     &         (' ', I=CROSSPOINT-1,0,-1)
      END IF
*     -----End of check-----

      IND1=CHILD1
      IND2=CHILD2

  100 FORMAT(' (Check the routine CROSSOVER(parent1=', I7,
     &                                  ', parent2=',I7, '):'/
     &       '   | CROSSPOINT=', I2/
     &       '   | Parent1: ', 23A1, ' \\/ ==> Child1: ', 23A1/
     &       '   | Parent2: ', 23A1, ' /\\     Child2: ', 23A1/
     &       '   |          ', 23A1,   '                ', 23A1, ')')
      END

*--------------------------------------------------------------------------*
* A subroutine that flips the (I+1)-th lowest bit of given individual     *
*--------------------------------------------------------------------------*
      SUBROUTINE FLIP(IND, FLIP_PLACE)
      INTEGER IND, FLIP_PLACE, I
     &        CHECKCOUNT, BIT_IND(0:21),
     &        AFTER_FLIP, BIT_AFTER_FLIP(0:21), NUM, NUM_AFTER_FLIP
      DATA    CHECKCOUNT/0/
      SAVE    CHECKCOUNT

      IF (MOD(IND/2**FLIP_PLACE,2).EQ.1) THEN
         AFTER_FLIP=IND-2**FLIP_PLACE
      ELSE
         AFTER_FLIP=IND+2**FLIP_PLACE
      END IF
```

```
*     -----Check whether the FLIP routine works-----
      IF (CHECKCOUNT.LT.3) THEN
         CHECKCOUNT=CHECKCOUNT+1
         NUM=IND
         NUM_AFTER_FLIP=AFTER_FLIP
         DO 10 I=0,21
            BIT_IND(I)=MOD(NUM,2)
            BIT_AFTER_FLIP(I)=MOD(NUM_AFTER_FLIP,2)
            NUM=NUM/2
            NUM_AFTER_FLIP=NUM_AFTER_FLIP/2
   10    CONTINUE
         WRITE(*,100) IND, FLIP_PLACE,
     &      (BIT_IND(I),I=21,0,-1), (BIT_AFTER_FLIP(I),I=21,0,-1),
     &      (' ',I=21,FLIP_PLACE+1,-1), '^', (' ',I=FLIP_PLACE-1,0,-1),
     &      (' ',I=21,FLIP_PLACE+1,-1), '^', (' ',I=FLIP_PLACE-1,0,-1)
      END IF
*     -----End of check-----

      IND=AFTER_FLIP

 100  FORMAT(' (Check the routine FLIP(ind=', I7,
     &                                ', flip_place=', I2, '):'/
     &       '   | Before_flip: ', 22I1, ' ==> After_flip: ', 22I1/
     &       '   |              ', 22A1, '                 ', 22A1, ')')
      END

*------------------------------------------------------------------------*
* A function that printouts the current state of the search              *
*------------------------------------------------------------------------*
      SUBROUTINE OBSERVE(IND, FITNESS, POPSIZE, GENERATION, BESTSOFAR)
      INTEGER POPSIZE, IND(1:POPSIZE), GENERATION,
     &        BEST, K, I, BIT(0:21), NUM
      REAL    FITNESS(1:POPSIZE), BESTSOFAR, SUM

      BEST=1
      SUM=FITNESS(1)
      DO 10 K=2, POPSIZE
         SUM=SUM+FITNESS(K)
         IF(FITNESS(K).GT.FITNESS(BEST)) BEST=K
   10 CONTINUE

      NUM=IND(BEST)
      DO 20 I=0,21
         BIT(I)=MOD(NUM,2)
         NUM=NUM/2
   20 CONTINUE
```

```
      IF (FITNESS(BEST).GT.BESTSOFAR) THEN
         BESTSOFAR=FITNESS(BEST)
         WRITE(*,100) GENERATION, (BIT(K),K=21,0,-1), IND(BEST),
     &                -1.0+3.0/REAL(2**22-1)*IND(BEST), FITNESS(BEST),
     &                SUM/POPSIZE
      END IF

 100  FORMAT(1X, I5, 2X, 22I1, 2X, I7, 2X, F8.6, 2X, F8.6, 2X, F8.6)
      END



****************************************************************************
* A module for generating pseudo-random numbers                            *
****************************************************************************

*--------------------------------------------------------------------------*
* A subroutine that initialize the random seed.                            *
*--------------------------------------------------------------------------*
      SUBROUTINE RAND_INITIALIZE( )
      COMMON /RAND/NEXT
      SAVE   /RAND/
      DOUBLE PRECISION NEXT
      INTEGER SEED

      WRITE(*,*) 'Type in a random seed (positive integer).'
      READ(*,*) SEED
      NEXT=DBLE(SEED)

      END

*--------------------------------------------------------------------------*
* A function that generates a integer                                      *
*              between 0 and (given_parameter_value -1)                    *
*--------------------------------------------------------------------------*
      INTEGER FUNCTION RAND_INT(TO)
      COMMON /RAND/NEXT
      SAVE   /RAND/
      DOUBLE PRECISION NEXT
      INTEGER TO
      REAL A

      RAND_INT=INT(RAND_REAL( )*TO)

      END

*--------------------------------------------------------------------------*
```

```
* A function that generates a real number in [0, 1)                       *
*-------------------------------------------------------------------------*
      REAL FUNCTION RAND_REAL( )
      COMMON /RAND/NEXT
      SAVE   /RAND/
      DOUBLE PRECISION NEXT
      INTEGER NUM

      NEXT=DMOD(NEXT*1103515245D0+12345D0, 2147483647D0)
      NUM=MOD(INT(NEXT)/65536, 32768)
      RAND_REAL=REAL(NUM)/32768.0

      END
```

3–7 Experimental Results

The program in the previous section 3–6 reports the best individual, the best fitness and the average fitness in each time when the best-so-far fitness is improved. It also reports how the selection process works, how the crossover operation works, and how the mutation operation works.

Executing that program, we obtain the following results:

```
xcspc60_42% g77 applying_GA_to_optimize_func.f
                 (compile the source code and generate an executable code "a.out")
xcspc60_43% a.out                              (invoke the executable code "a.out")
 Type in a random seed (positive integer).
1                                        (input to the executing program "a.out")
 ------------------------------------------------
 We now apply genetic algorithm to the problem of
 finding such a value of x from the range [-1,2]
 that maximize the function f(x)=x*sin(10*pi*x)+1.
 ------------------------------------------------

 Gen.#       Best_ind         (its_val)  Best_x   Best_f(x)  Average_f(x)
```

```
-----  ----------------------  -------  --------  --------  --------
    0  1111000111010111001101  3962317  1.834071  2.609172  0.969633
(Check the routine SELECT(FITNESS, popsize=50):
  | FITNESS(22)= 1.639175 >=  1.001883=FITNESS(15) ==> We select the 22-th individual.)
(Check the routine SELECT(FITNESS, popsize=50):
  | FITNESS( 6)= 2.448672 >=  1.011820=FITNESS(34) ==> We select the  6-th individual.)
(Check the routine SELECT(FITNESS, popsize=50):
  | FITNESS(22)= 1.639175 >=  1.606463=FITNESS(50) ==> We select the 22-th individual.)
(Check the routine CROSSOVER(parent1= 495585, parent2=3428462):
  | CROSSPOINT=11
  | Parent1: 00011110001 11111100001 \/ ==> Child1: 00011110001 00001101110
  | Parent2: 11010001010 00001101110 /\     Child2: 11010001010 11111100001
  |                     ^                                      ^          )
(Check the routine CROSSOVER(parent1=1506017, parent2=3672623):
  | CROSSPOINT= 3
  | Parent1: 0101101111101011100 001 \/ ==> Child1: 0101101111101011100 111
  | Parent2: 1110000000101000101 111 /\     Child2: 1110000000101000101 001
  |                             ^                                      ^   )
(Check the routine CROSSOVER(parent1=3437851, parent2=1268266):
  | CROSSPOINT=10
  | Parent1: 110100011101 0100011011 \/ ==> Child1: 110100011101 1000101010
  | Parent2: 010011010110 1000101010 /\     Child2: 010011010110 0100011011
  |                      ^                                      ^         )
(Check the routine FLIP(ind=2850477, flip_place= 4):
  | Before_flip: 1010110111111010101101 ==> After_flip: 1010110111111010111101
  |                                ^                                      ^    )
(Check the routine FLIP(ind=1646233, flip_place= 4):
  | Before_flip: 0110010001111010011001 ==> After_flip: 0110010001111010001001
  |                                ^                                      ^    )
(Check the routine FLIP(ind=2350858, flip_place= 0):
  | Before_flip: 1000111101111100001010 ==> After_flip: 1000111101111100001011
  |                                    ^                                      ^)
    5  1110000111010110111101  3700157  1.646559  2.636949  2.190265
    6  1110000111010111111010  3700218  1.646603  2.637234  2.284685
    7  1110000111110110111101  3702205  1.648024  2.644850  2.409237
    8  1110000111110111111101  3702269  1.648070  2.645041  2.519771
    9  1111001111000111110010  3994098  1.856802  2.814564  2.491488
   13  1111001011000111110010  3977714  1.845083  2.823120  2.682922
   16  1111001110000110111101  3989949  1.853835  2.840398  2.674170
   18  1111001110000110101101  3989933  1.853823  2.840466  2.778359
   19  1111001110000011010010  3989714  1.853667  2.841381  2.760096
   23  1111001100000110101101  3981741  1.847964  2.844185  2.705675
   28  1111001100100001111101  3983485  1.849211  2.848644  2.662642
   33  1111001100100001111111  3983487  1.849213  2.848647  2.833734
   37  1111001100101001111101  3983997  1.849577  2.849415  2.663872
   38  1111001100111001111101  3985021  1.850310  2.850222  2.742151
   43  1111001100111011111101  3985149  1.850401  2.850254  2.670041
```

```
   44  1111001100111101111101  3985277  1.850493  2.850271  2.772511
   49  1111001100111111111101  3985405  1.850585  2.850272  2.796093
   54  1111001100111111101101  3985389  1.850573  2.850273  2.588252
   55  1111001100111111100101  3985381  1.850567  2.850273  2.743095
   57  1111001100111110111111  3985343  1.850540  2.850274  2.673532
xcspc60_44% a.out                                  invoke the executable code "a.out"
 Type in a random seed (positive integer).
3                                              input to the executing program "a.out"
 -------------------------------------------------
 We now apply genetic algorithm to the problem of
 finding such a value of x from the range [-1,2]
 that maximize the function f(x)=x*sin(10*pi*x)+1.
 -------------------------------------------------

 Gen.#        Best_ind         (its_val)  Best_x   Best_f(x) Average_f(x)
 -----  ----------------------  -------  --------  --------  --------
     0  1111001010110110010111  3976599  1.844286  2.814652  0.931059
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS(20)= 0.840346 >=  0.675106=FITNESS(45) ==> We select the 20-th individual.)
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS(47)=-0.701067 <   0.510389=FITNESS(31) ==> We select the 31-th individual.)
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS( 9)= 2.115118 >=  1.109483=FITNESS(11) ==> We select the  9-th individual.)
 (Check the routine CROSSOVER(parent1=3059904, parent2= 248745):
   | CROSSPOINT=11
   | Parent1: 10111010110 00011000000 \/ ==> Child1: 10111010110 01110101001
   | Parent2: 00001111001 01110101001 /\     Child2: 00001111001 00011000000
   |                     ^                                      ^          )
 (Check the routine CROSSOVER(parent1=3976599, parent2=3965452):
   | CROSSPOINT= 9
   | Parent1: 1111001010110 110010111 \/ ==> Child1: 1111001010110 000001100
   | Parent2: 1111001000001 000001100 /\     Child2: 1111001000001 110010111
   |                       ^                                      ^        )
 (Check the routine CROSSOVER(parent1=3965452, parent2= 785691):
   | CROSSPOINT= 7
   | Parent1: 111100100000100 0001100 \/ ==> Child1: 111100100000100 0011011
   | Parent2: 001011111111010 0011011 /\     Child2: 001011111111010 0001100
   |                         ^                                      ^      )
 (Check the routine FLIP(ind=3361079, flip_place= 4):
   | Before_flip: 1100110100100100110111 ==> After_flip: 1100110100100100100111
   |                                 ^                                  ^    )
 (Check the routine FLIP(ind=3976204, flip_place= 1):
   | Before_flip: 1111001010110000001100 ==> After_flip: 1111001010110000001110
   |                                    ^                                  ^ )
 (Check the routine FLIP(ind= 605647, flip_place= 4):
   | Before_flip: 0010010011110111001111 ==> After_flip: 0010010011110111011111
   |                                 ^                                  ^    )
```

```
    4  1111001011000000001011  3977227  1.844735  2.819561  2.234270
    5  1111001100001100010011  3982099  1.848220  2.845331  2.310440
    7  1111001100100011000000  3983552  1.849259  2.848758  2.645174
    8  1111001100101100000100  3984132  1.849674  2.849577  2.716642
   10  1111001100101100010011  3984147  1.849685  2.849594  2.657524
   12  1111001100101111000000  3984320  1.849808  2.849775  2.728778
   14  1111001100111111000001  3985345  1.850542  2.850274  2.800497
xcspc60_45% a.out                                   (invoke the executable code "a.out")
 Type in a random seed (positive integer).
3333                                            (input to the executing program "a.out")
 -------------------------------------------------
 We now apply genetic algorithm to the problem of
 finding such a value of x from the range [-1,2]
 that maximize the function f(x)=x*sin(10*pi*x)+1.
 -------------------------------------------------


 Gen.#        Best_ind         (its_val)  Best_x   Best_f(x) Average_f(x)
 -----  ----------------------  -------  --------  --------  --------
    0  1101111110110111010111  3665367  1.621675  2.020905  0.872255
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS(40)= 1.228418 >=  1.100637=FITNESS( 8) ==> We select the 40-th individual.)
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS(41)= 0.987819 >=  0.931672=FITNESS(19) ==> We select the 41-th individual.)
 (Check the routine SELECT(FITNESS, popsize=50):
   | FITNESS(32)= 0.757835 <   1.000196=FITNESS(35) ==> We select the 35-th individual.)
 (Check the routine CROSSOVER(parent1=2015470, parent2=1724211):
   | CROSSPOINT=21
   | Parent1: 0 111101100000011101110 \/ ==> Child1: 0 110100100111100110011
   | Parent2: 0 110100100111100110011 /\     Child2: 0 111101100000011101110
   |           ^                                       ^                     )
 (Check the routine CROSSOVER(parent1=1767779, parent2=2427505):
   | CROSSPOINT= 7
   | Parent1: 011010111110010 1100011 \/ ==> Child1: 011010111110010 1110001
   | Parent2: 100101000010100 1110001 /\     Child2: 100101000010100 1100011
   |                         ^                                       ^       )
 (Check the routine CROSSOVER(parent1=1065732, parent2=2015470):
   | CROSSPOINT=18
   | Parent1: 0100 000100001100000100 \/ ==> Child1: 0100 101100000011101110
   | Parent2: 0111 101100000011101110 /\     Child2: 0111 000100001100000100
   |              ^                                       ^                  )
 (Check the routine FLIP(ind= 813159, flip_place=13):
   | Before_flip: 0011000110100001100111 ==> After_flip: 0011000100100001100111
   |                       ^                                       ^           )
 (Check the routine FLIP(ind=4060168, flip_place=19):
   | Before_flip: 1111011111010000001000 ==> After_flip: 1101011111010000001000
   |                 ^                                       ^                 )
 (Check the routine FLIP(ind=2015470, flip_place= 4):
```

```
   | Before_flip: 01111011000000111101110 ==> After_flip: 01111011000000111111110
   |                               ^                                     ^     )
      2  1110001000111101101100  3706732  1.651262  2.649964  1.380827
      3  1110001000110001101100  3705964  1.650713  2.650299  1.543338
      9  1110001000101101011001  3705689  1.650516  2.650299  2.503086
     10  1110001000101101101001  3705705  1.650527  2.650301  2.528122
     11  1111001000111101011011  3968859  1.838750  2.725101  2.594279
     14  1111001001110101101001  3972457  1.841323  2.773342  2.590566
     16  1111001001111101011001  3972953  1.841678  2.779099  2.414793
     17  1111001100110101011001  3984729  1.850101  2.850092  2.680887
     22  1111001100110101101001  3984745  1.850112  2.850101  2.724118
     23  1111001100110111101001  3984873  1.850204  2.850166  2.760757
     27  1111001100111101101001  3985257  1.850479  2.850269  2.833693
     29  1111001100111101101101  3985261  1.850482  2.850270  2.828703
     30  1111001100111101111101  3985277  1.850493  2.850271  2.808124
     32  1111001100111110111101  3985341  1.850539  2.850273  2.743597
     40  1111001100111111001000  3985352  1.850547  2.850274  2.833087
xcspc60_46%
```

4 We can evolutionarily search arithmetic expressions.

References:
J.R.Koza(1992), Genetic Programming: on the programming of computers by means of natural selection, MIT Press.

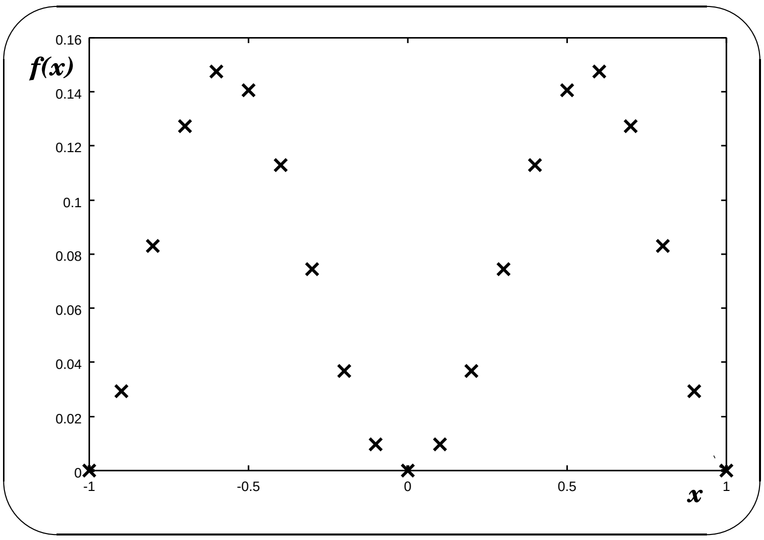
In this section, we see that we can evolutionarily search 2-dimensionally structured objects.

4–1 The Symbolic Regression Problem

Consider

A Symbolic Regression Problem:
Find such a function $f(x)$, in symbolic form, that fits the following sample of input-output relations:

x	f(x)
-1.0	0.0
-0.9	0.029241
-0.8	0.082944
-0.7	0.127449
-0.6	0.147456
-0.5	0.140625
-0.4	0.112896
-0.3	0.074529
-0.2	0.036864
-0.1	0.009801
0.0	0.0
0.1	0.009801
0.2	0.036864
0.3	0.074529
0.4	0.112896
0.5	0.140625
0.6	0.147456
0.7	0.127449
0.8	0.082944
0.9	0.029241
1.0	0.0

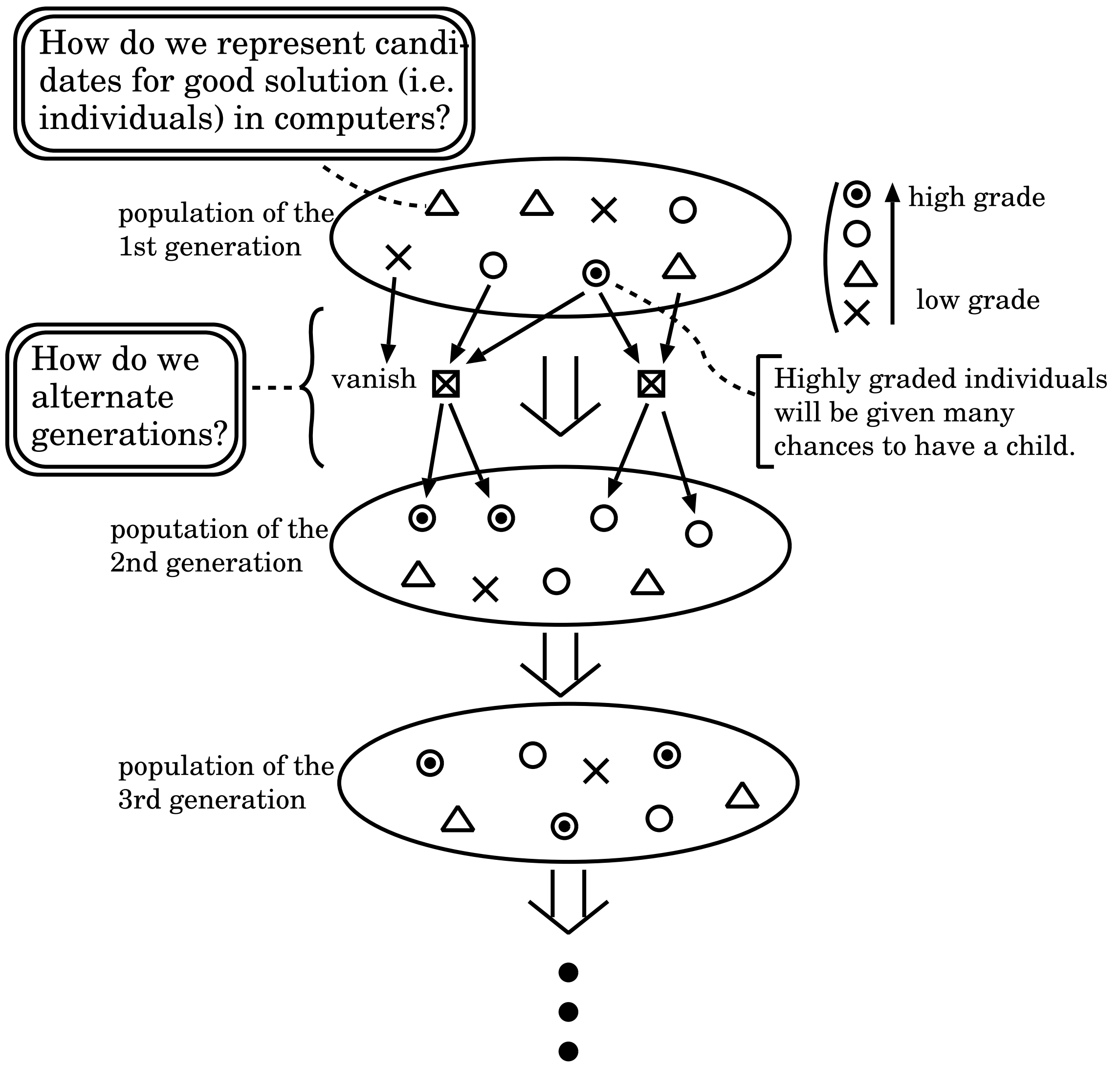


4–2 How do we evolutionarily solve the symbolic regression problem?

We can proceed in the same manner as before; that is,

To follow the search scheme reviewed a little while ago, we must clarify

- how to represent candidates for good solution, and
- how to alternate generations.



⇨ **About these issues, we mainly follow the scheme described in the Koza(1992)'s book.**

4–3 A Method for Representing Candidates for Good Solution

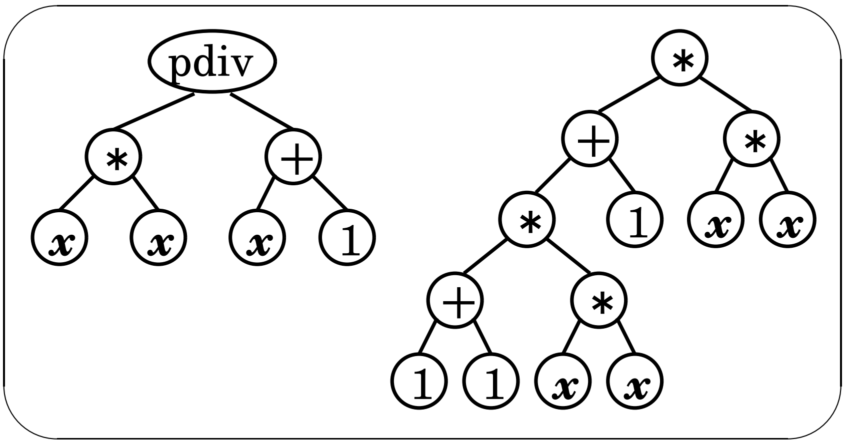
We are to find a function of variable x in symbolic form. So, we can consider many possible search points, e.g. $\frac{x*x}{x+1}, (2x^2+1)*x^2, x\sin(10\pi x)+1, \exp(\log x+\sin x), \ldots$ But to make the search steadily, we cannot help fixing the search space.

⇨ • **We fix the set F of possible primitive function (or operation) symbols to be $\{+, -, *, \mathbf{pdiv}, \mathbf{if\_lte}\}$, and fix the set T of possible primitive terminal symbols to be $\{x, 1\}$.**

$$\mathbf{pdiv}(a,b) = \begin{cases} 1 & \textbf{if } b=0 \\ a/b & \textbf{otherwise} \end{cases}$$

$$\mathbf{if\_lte}(a,b,c,d) = \begin{cases} c & \textbf{if } a \le b \\ d & \textbf{otherwise} \end{cases}$$

• **We consider an expression constructed from 4 arithmetic operations "+", "−", "∗", "pdiv", function "if_lte", variable name "x", and constant name "1" as an individual. So, individuals have so-called "<u>tree</u>" structures, that spread 2 dimentionally.**



4–4 A Method for Creating Initial Individuals

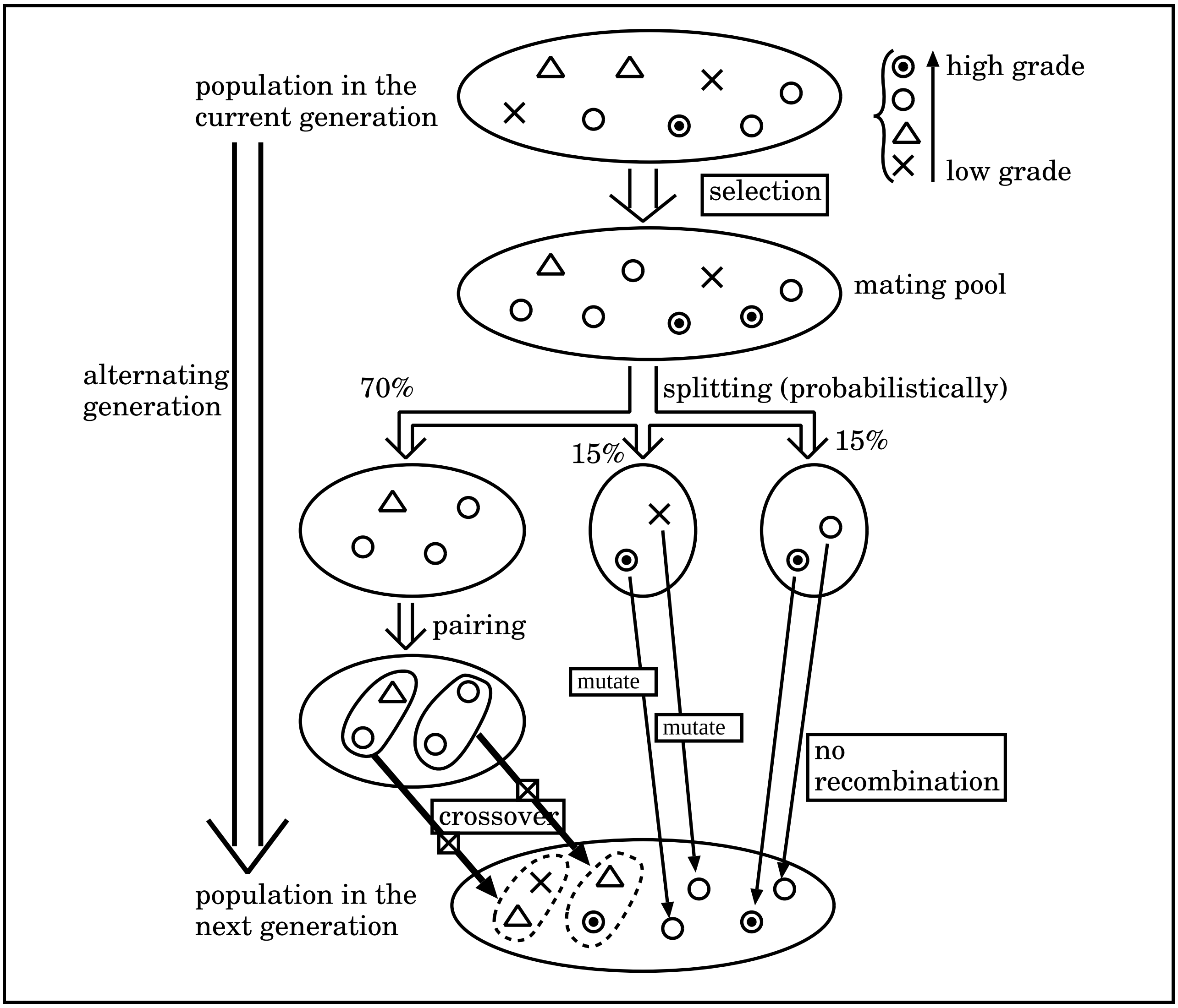
For each integer k between 3 and 25 (called "size"), we randomly create an equal number of expressions that contains k primitive symbols.

- $\frac{\text{Popsize}}{23}$ expressions that contains 3 primitive symbols
  $\frac{\text{Popsize}}{23}$ expressions that contains 4 primitive symbols
  $\frac{\text{Popsize}}{23}$ expressions that contains 5 primitive symbols
  ....
  $\frac{\text{Popsize}}{23}$ expressions that contains 25 primitive symbols
  } Initial Population

- **Trees with various shapes and labels are all given an equal probability to be created.**

4–5 A Method for Alternating Generations

We alternate generations as follows:



Given a population of individuals in the current generation, we first select good individuals that are utilized to create new individuals.
For this selection operation, we need a measure of goodness of individuals, also called fitness.

Fitness

Suppose that we are given a sample of input-output relations $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$.

Since we are to find a hypothetical function $h(x)$ **that fits the given sample relations as tightly as possible, we consider that**

$$\text{the fitness of } \boxed{\text{individual } h(x)} = \frac{1}{n}\sum_{i=1}^{n} |h(x_i) - y_i| .$$

So in this problem, the smaller the fitness, the better the individual.

x	target function (unknown)	hypothetical function $h(x)$	|error|
x_1	y_1	$h(x_1)$	$\|h(x_1) - y_1\|$
x_2	y_2	$h(x_2)$	$\|h(x_2) - y_2\|$
x_3	y_3	$h(x_3)$	$\|h(x_3) - y_3\|$
$\cdots$	$\cdots$	$\cdots$	$\cdots\cdots$
x_n	y_n	$h(x_n)$	$\|h(x_n) - y_n\|$
			$\frac{1}{n}\sum_{i=1}^{n}\|h(x_i) - y_i\|$ (Average)

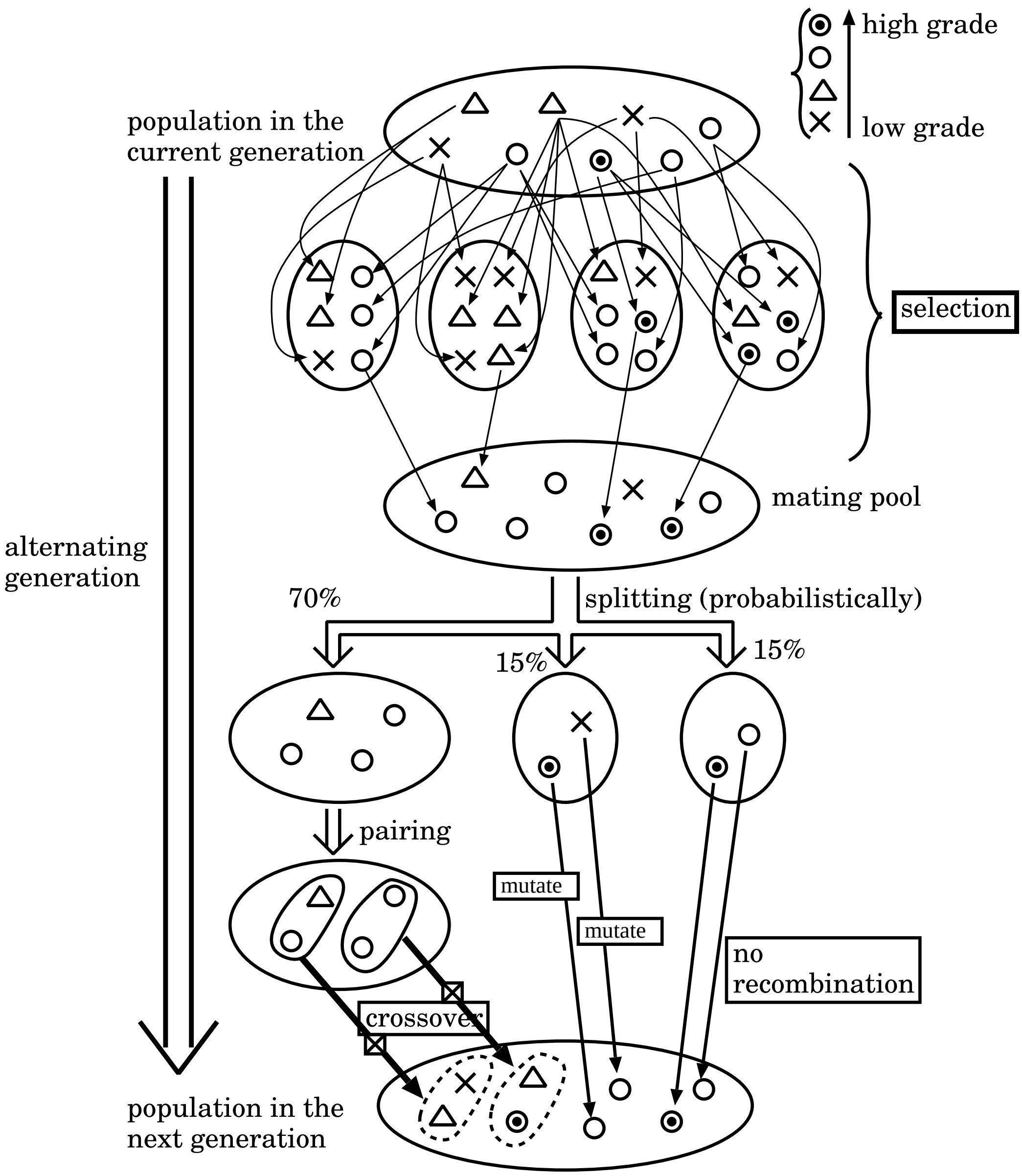
Selection

Based on the fitness, we also adopt the tournament selection scheme as in section 3 ; but we now fix the tournament size to be 6. So,

we repeat the following selection process so many times as is equal to the population size:

(1) Six individuals are randomly drawn from the population with replacement.

(2) The best of them is selected into the mating pool.



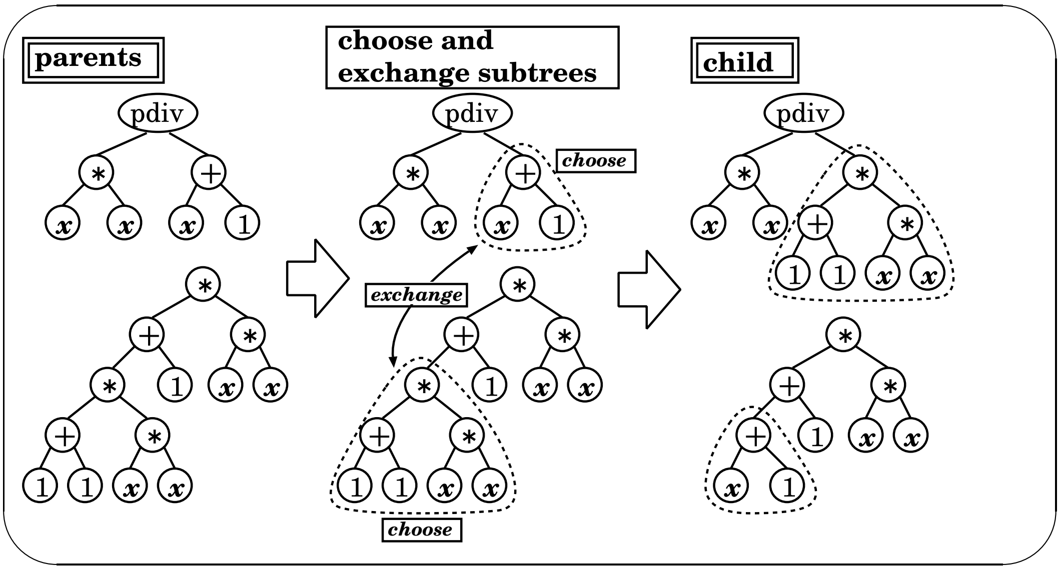
After the selection process, we next create new individuals from the selected individuals. Such creation operations are accomplished by imitating real animal's crossover and mutation events; so we also call such operations crossover and mutation, respectively.

Crossover

Given two parent individuals, we perform the following operations:

(1) For each parent individual, choose one subtree randomly.

(2) Exchange the two chosen subtrees.

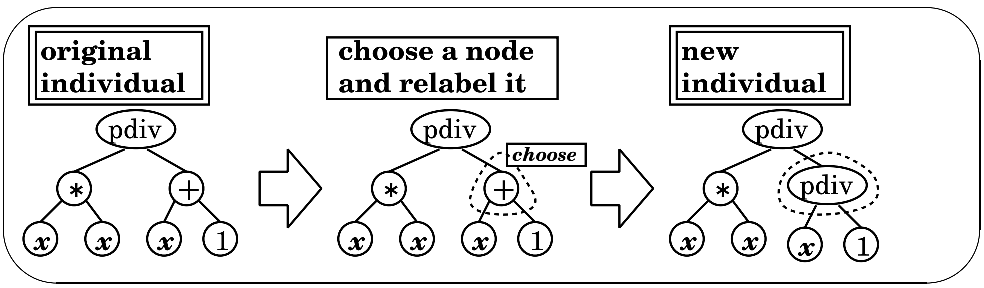


Mutation

Given an individual, we perform the following operations:

(1) Choose a node randomly.

(2) Reassign any possible function (or terminal) symbol to the chosen node.



4–6 Implementation in C-language

We now give a C program that implements the evolutionary search procedure described earlier.

Note: We use some terminologies with different meanings from those in section **3** ; we now define

$$\text{Crossover\_rate} = \frac{\text{the number of new individuals by crossover}}{\text{population size}}$$

$$\text{Mutation\_rate} = \frac{\text{the number of new individuals by mutation}}{\text{population size}}$$

$$\text{Reproduction\_rate} = \frac{\text{the number of individuals that survives}}{\text{population size}}$$

```
/**********************************************************************/
/*   PlainGPC/Reg-sample-run-2/main.c                                 */
/*--------------------------------------------------------------------*/
/* This is a core part of programs that implements the following evolution-*/
/* ary computation for the symbolic regression problem:               */
/*                                                                    */
/*   * Rough Sketch of the Computation:                               */
/*       initialize all program modules                               */
/*            (e.g. setting parameters, memory allocation, etc.);     */
/*       for (run=1; run<=num_runs ; run++)                           */
/*         generate a population of individuals by the ramped uniform method;*/
/*         calculate fitness for each individual;                     */
/*         for (k=1; k<=max_gen; k++){                                */
/*           for (i=0; i<pop_size; ){                                 */
/*             if ((rnd=pseudo_random_number_in_[0,1))                 */
/*                   < branch_prob_crossover_func_pt  && i<pop_size-1) {  */
/*               select two individuals by the tournament selection    */
/*                  and copy those individuals to child[i] and child[i+1];*/
/*               crossover child[i] and child[i+1]                    */
/*                  by exchanging any subtrees that have two or more nodes;*/
/*               i+=2 ;                                               */
/*             }else if (rnd < branch_prob_crossover && i<pop_size-1){  */
/*               select two individuals by the tournament selection    */
/*                  and copy those individuals to child[i] and child[i+1];*/
/*               crossover child[i] and child[i+1]                    */
/*                  by exchanging any subtrees;                       */
/*               i+=2 ;                                               */
/*             }else if (rnd < branch_prob_crossover_or_mutation) {    */
```

```
/*              select one individual by the tournament selection       */
/*                 and copy this individual to child[i];                */
/*              mutate child[i] by relabeling any node randomly;        */
/*              ++i ;                                                   */
/*            }else {                                                   */
/*              select one individual by the tournament selection       */
/*                 and copy this individual to child[i];                */
/*              ++i ;                                                   */
/*            }                                                         */
/*          }                                                           */
/*                                    { At this point, the new population   */
/*                                      of individuals has been completed.} */
/*          alternate the generation by deleting the current-generation  */
/*             individuals and supposing the newly-generated individuals */
/*             to be the current ones;                                    */
/*          calculate fitness for each new (current) individuals;         */
/*          calculate and record the best fitness, the average fitness,   */
/*             the hit number of the best individual, etc.                */
/*             on the new population;                                     */
/*          if (the best fitness < epsilon)                              */
/*            break;                                                     */
/*        }                                                              */
/*        record a summary about (run)-th run;                           */
/*      }                                                                */
/*      generate a text file that records how the computation proceeds;  */
/*                                                                       */
/*   * Target Function:                                                  */
/*      We only consider unary functions; for example,                   */
/*      f1(x)=x*x/2, f2(x)=x^2+4*x+3, ......                             */
/*                                                                       */
/*   * We use expressions constructed from          |                    */
/*     5 function symbols                           |                    */
/*         +, -, *, /,if-less-than-or-equal         | ==>                */
/*    and 2 or 3 terminal symbols                   |  Read also the module*/
/*        x, 1  (, some ephemeral-random-constants) |  "../REGRESSION/    */
/*    as individuals to represent search points.    |   fitness_linked_1.c."*/
/*                                                  |                    */
/*   * We represent individuals by the "Linked"     |                    */
/*     data structure.                              |                    */
/*                                                                       */
/*   * We adopt the tournament selection scheme with default tournament  */
/*     size 6.                                                           */
/*                                                                       */
/*   * We adopt the point mutation that chooses any node in a given tree */
/*     and relabels it randomly.                                         */
/************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include "lib.h"

#define  MAX_ARITY  4
#include "linked.h"
#include "reg_linked.h"
                                    /*----------------------------------*/
double target_function(double x);   /* We assume that the target function */
```

```
                                    /* has this function name and type.   */
                                    /*------------------------------------*/
static void initialize_all_modules(void);


/* Alias Names for Parameters that Control Runs */
#define  RANDOM_SEED                 Param.random_seed

#define  NUM_RUNS                    Param.num_runs
#define  MAX_GENERATION              Param.max_generation
#define  POP_SIZE                    Param.pop_size

#define  CROSSOVER_RATE_FUNC_PT      Param.crossover_rate_func_pt
#define  CROSSOVER_RATE_ANY_PT       Param.crossover_rate_any_pt
#define  REPRODUCTION_RATE           Param.reproduction_rate
#define  MUTATION_RATE               Param.mutation_rate

                                   /*---------------------------------*/
static GP_parameters Param;        /* A table of parameters           */
static char Report_file[100];      /* The name of the report file     */
                                   /* will be recorded in this array. */
                                   /*---------------------------------*/
static Boolean Best_individual_needed;
                   /*-------------------------------------------------------*/
                   /* A variable that records whether the program outputs */
                   /* all best-of-run individuals to the report file.     */
                   /*-------------------------------------------------------*/
static Boolean Observe_with_display;
                   /*-------------------------------------------------------*/
                   /* A variable that records whether the program opens a */
                   /* window for observing the progress of the search.    */
                   /*-------------------------------------------------------*/

/*------------------------------------------------------*/
/* A List of Parameter Variables that are Specially Used */
/* in the Symbolic Regression Problem                   */
/*--------------------------------------------------------------------------*/
#ifdef NUM_FITNESS_CASES                               /* We can set these parameter*/
  static int   Num_fitness_cases = NUM_FITNESS_CASES; /* variables by specifying  */
#else                                                  /* the corresponding macros */
  static int   Num_fitness_cases = 21;                 /* in the makefile or the   */
#endif                                                 /* command line.            */
#ifdef VAR_LOWER_LIMIT                                 /*--------------------------*/
  static float Var_lower_limit   = VAR_LOWER_LIMIT;
#else
  static float Var_lower_limit   = -1.0;
#endif
#ifdef VAR_UPPER_LIMIT
  static float Var_upper_limit   =  VAR_UPPER_LIMIT;
#else
  static float Var_upper_limit   =  1.0;
#endif
#ifdef MAX_ERROR_FOR_HIT
  static float Max_error_for_hit =  MAX_ERROR_FOR_HIT;
                     /*--------------------------------------------------*/
                     /* We will use this parameter variable when we      */
```

```
                     /* decide whether an individual fit a fitness case. */
                     /*--------------------------------------------------*/
#else
  static float Max_error_for_hit =  0.01;
#endif
static Boolean Smaller_fitness_is_better = TRUE;

/*---------------------------------------------------------*/
/* Core Variables for Proceeding an Evolutionary Computation */
/*---------------------------------------------------------*/
static Tree   *Individual;
static float *Raw_fitness;
static int    *Num_of_hits;
static int    *Ind_size;
static int    *Ind_height;
static int    Best_of_gen_ind_num;
static float Current_ave_fitness;
static float Current_ave_size;
static float Current_ave_height;

static Tree  *Next_individual;

static int *Arity_table;
static int Num_func;
static int Num_terminals;

/*-----------------------------------------------*/
/* Variables for Recording the Progress of GP Runs */
/*------------------------------------------------------------*/
static Tree   *Best_of_run_individual; /* A storage for recording */
                                       /* best-of-run individuals */
                                       /*------------------------*/
static float *Best_of_run_raw_fitness;
                                  /*-----------------------------*/
static int    Best_so_far_run;    /* A variable for recording the */
                                  /* run number that gives the best*/
                                  /* individual among all runs    */
                                  /*-----------------------------*/
static char *Comment_on_overall_runs;


/*--------------------------------------------------------------------------*/
/* A main routine that proceeds an evolutionay computation               */
/*--------------------------------------------------------------------------*/
/* (the format of parameter specification in the command line) :        */
/*   We assume that parameters will be specified in the command line     */
/*   as follows:                                                         */
/*        format    :       comments                                     */
/*      -param filename   or   -p filename:                              */
/*                     This declares that a detailed specification of     */
/*                     GP-parameters will be found in the file "filename."*/
/*                     A specification of parameter in  the command line  */
/*                     will be prior to one in this file.                 */
/*      -report filename   or   -r filename :                            */
/*                     This declares that the progress of the computation */
/*                     (e.g. how the best fitness varies with generation, */
```

```
/*                   how the average fitness varies with generation, the*/
/*                   best individual, the average progress of runs, etc.)*/
/*                   should be output to the file "filename."        */
/*                   If this specification is missing, the default file */
/*                   name "report_file.default" is assume.           */
/*                   If the file name "stdout" is specified, then all   */
/*                   the record will be output to the standard out file.*/
/*      -best_ind_needed   or   -b :                                 */
/*                   This declares that all the best-of-run individuals */
/*                   should be output to the file specified by the     */
/*                   format "-report filename" (or the file           */
/*                   "report_file.default").                          */
/*      -seed integer   or   -s integer :                            */
/*                   This declares that the initial value of the random */
/*                   seed should be set to  "integer."                */
/*      -no_observe   or   -n :                                      */
/*                   This declares that the program should not open any */
/*                   window for observing the progress of the computation.*/
/*      -popsize integer  :                                          */
/*                   This declares that the population size should be   */
/*                   set to "integer."                                */
/*      -max_gen integer  :                                          */
/*                   This declares that the maximum number of generation*/
/*                   should be set to "integer."                      */
/*      -num_runs integer :                                          */
/*                   This declares that the number of runs should be    */
/*                   set to "integer."                                */
/*      -help   or   -h   or   -H :                                  */
/*                   If this is specified, the program will not proceed */
/*                   the GP run, but give an information about how to   */
/*                   specify command line options.                    */
/*-------------------------------------------------------------------*/
/* (the format of the file to specify parameters) :                  */
/*   We assume that parameters will be specified in the file as follows: */
/*     * Every line that begins with the character "#" will be treated as */
/*       a comment line.                                             */
/*     * Lines that specifies parameters should have one of the following */
/*       forms, where the blank characters may be inserted in any place.  */
/*       (Any order of parameters is possible.)                      */
/*                                                      default value */
/*                                                                ↓ */
/*          num_runs = any positive integer                          1 */
/*          pop_size = any positive integer                        500 */
/*          max_generation = any positive integer                   50 */
/*          random_seed  = any nonnegative integer                   1 */
/*          allow_ephe_ran_const = yes  or  no                      no */
/*          min_ephe_ran_const = any real number                   0.0 */
/*          max_ephe_ran_const = any real number                   1.0 */
/*          restrict_num_ephe_ran_const = yes  or  no               no */
/*          num_ephe_ran_const = any positive integer              100 */
/*          max_height_of_initial_tree = any integer >=2             6 */
/*          max_height_after_crossover = any positive integer       17 */
/*          max_height_of_replacement_subtree = any positive integer   4 */
/*          min_size_of_initial_tree = any positive integer          3 */
/*          max_size_of_initial_tree = any positive integer         25 */
/*          thinning_rate_for_larger_ind = any real number in [0,1] 0.21 */
```

```
/*          tournament_size = any positive integer                     6 */
/*          control_param_adj_fit = any nonnegative real number      1.0 */
/*          crossover_rate_func_pt = any real number in [0,1]       0.20 */
/*          crossover_rate_any_pt  = any real number in [0,1]       0.50 */
/*          reproduction_rate = any real number in [0,1]            0.15 */
/*          mutation_rate = any real number in [0,1]                0.15 */
/*          report_file = any file name              report_file.default */
/*          best_individual_needed = yes  or  no                      no */
/*          observe_with_display = yes  or  no                       yes */
/*    * Any character that occurs after 100-th column is treated as a    */
/*      part of a comment.                                               */
/*    * If two or more specification appear in one line, the second and  */
/*      the later ones will be ignored.                                  */
/* Remark :  About the choice of random_generation_method,               */
/* ~~~~~~   initial_pop_creation_method and selection_method, this       */
/*          routine knows by directly hearing from running modules.      */
/*-----------------------------------------------------------------------*/
main(int argc, char *argv[])
{
  int   i, num_ephe_ran_const, run, generation, best_ind_num,
        sum_of_size, sum_of_height;
  float best_so_far_fitness, sum_of_raw_fitness, sum_of_rate, fraction,
        branch_prob_crossover_func_pt, branch_prob_crossover,
        branch_prob_crossover_or_mutation;
  Tree  best_so_far_individual, *temp;
  FILE  *report_fp, *gnuplot_data_fp;
  char  function_name[50];

  get_parameters(argc, argv,              /*-----------------------------*/
                 &Param,                  /* read in values of parameters */
                 Report_file,             /*-----------------------------*/
                 &Best_individual_needed,
                 &Observe_with_display);

                   /* check whether the parameters CROSSOVER_RATE_FUNC_PT */
                   /* etc. is consistently specified.                     */
  sum_of_rate = CROSSOVER_RATE_FUNC_PT + CROSSOVER_RATE_ANY_PT
                + REPRODUCTION_RATE + MUTATION_RATE;
  if (sum_of_rate<0.9999 || 1.0001<sum_of_rate) {
    printf("Error: The parameter file specified\n"
           "            crossover_rate_func_pt = %e,\n"
           "            crossover_rate_any_pt  = %e,\n"
           "            reproduction_rate      = %e and\n"
           "            mutation_rate          = %e;\n"
           "       so crossover_rate_func_pt + crossover_rate_any_pt + "
           "reproduction_rate != 1.0, a curious setting.\n",
           CROSSOVER_RATE_FUNC_PT, CROSSOVER_RATE_ANY_PT,
           REPRODUCTION_RATE, MUTATION_RATE);
    exit(1);
  }

              #ifdef SMALL
              POP_SIZE = 10;
              MAX_GENERATION = 2;
              Param.max_height_of_initial_tree = 4;
              printf("--------------------------------------------\n"
```

```
                  "To see how the program behaves,"
                  "three GP parameters are changed as follows:\n"
                  "    pop_size                  = %d,\n"
                  "    max_generation            = %d,\n"
                  "    min_size_of_initial_tree  = %d,\n"
                  "    max_size_of_initial_tree  = %d;\n"
                  "then the execution trace are reported.\n\n",
                  POP_SIZE, MAX_GENERATION,
                  Param.min_size_of_initial_tree,
                  Param.max_size_of_initial_tree);
            #endif
                             /*---------------------------------------*/
initialize_all_modules();    /* initialize all modules (e.g. informing */
                             /* parameter values to related modules,   */
                             /* memory allocation, initialization of   */
                             /* local variables)                       */
                             /*---------------------------------------*/
branch_prob_crossover_func_pt
  = CROSSOVER_RATE_FUNC_PT
     / (2.0 - CROSSOVER_RATE_FUNC_PT - CROSSOVER_RATE_ANY_PT);
branch_prob_crossover
  = (CROSSOVER_RATE_FUNC_PT + CROSSOVER_RATE_ANY_PT)
     / (2.0 - CROSSOVER_RATE_FUNC_PT - CROSSOVER_RATE_ANY_PT);
branch_prob_crossover_or_mutation
  = (CROSSOVER_RATE_FUNC_PT + CROSSOVER_RATE_ANY_PT + 2.0*MUTATION_RATE)
     / (2.0 - CROSSOVER_RATE_FUNC_PT - CROSSOVER_RATE_ANY_PT);

/*--------------------*/
/*   repeat GP runs   */
/*--------------------*/
for (run=0; run<NUM_RUNS; run++) {
            #if defined(TRACEIND) || defined(TRACEFIT) || defined(TRACESEL) \
                     || defined(TRACECROSS) || defined(TRACEMUT) \
                     || defined(TRACERESULT) || defined(TRACEALL)
            printf("\n\n"
                  "*************************************************\n"
                  "*    Run (%d)\n"
                  "*************************************************\n",
                  run);
            #endif
  num_ephe_ran_const = create_initial_pop_ramped_uniform(Individual, POP_SIZE);
  create_terminal_value_table(num_ephe_ran_const);
  initialize_more_on_mutation_point(num_ephe_ran_const);

  for (generation=0;  ; generation++) {
            #if defined(TRACEIND) || defined(TRACEFIT) || defined(TRACESEL) \
                     || defined(TRACECROSS) || defined(TRACEMUT) \
                     || defined(TRACERESULT) || defined(TRACEALL)
            printf("\n"
                  "=====================================\n"
                  "   Generation  %d\n"
                  "=====================================\n",
                  generation);
            #endif

    /*-----------------------------------------*/
```

```
/* recognize the current state of the search */
/*-------------------------------------------*/
for (i=0; i<POP_SIZE; i++) {
        #if defined(TRACEIND) || defined(TRACEALL) /*----------------------*/
        printf("Individual[%d]:\n", i);            /* display all individuals*/
                                                   /* and their fitnesses  */
                                                   /*----------------------*/
        print_an_individual_linked(stdout, Individual[i]);
        #endif
  calculate_fitness_and_hits_linked(Individual[i], &Raw_fitness[i],
                                  &Num_of_hits[i]);
  Ind_size[i]   = num_of_nodes(Individual[i]);
  Ind_height[i] = height(Individual[i]);
        #if defined(TRACEIND) || defined(TRACEALL)
        printf("  ==> fitness          = %e\n"
               "      num_of_hits      = %d\n"
               "      num_of_nodes     = %d\n"
               "      num_of_func_nodes = %d\n"
               "      height           = %d\n"
               "\n------------------------------\n",
               Raw_fitness[i], Num_of_hits[i],
               Ind_size[i],
               num_of_function_nodes(Individual[i]),
               Ind_height[i]);
        #endif
}
                                      /*------------------------------*/
best_ind_num = 0;                     /* the best individual, the     */
sum_of_raw_fitness = Raw_fitness[0];  /* average fitness, the average */
sum_of_size        = Ind_size[0];     /* size, and the average height */
sum_of_height      = Ind_height[0];   /* in the current generation    */
for (i=1; i<POP_SIZE; i++) {          /*------------------------------*/
  if (Raw_fitness[i] < Raw_fitness[best_ind_num])
    best_ind_num = i;
  sum_of_raw_fitness += Raw_fitness[i];
  sum_of_size        += Ind_size[i];
  sum_of_height      += Ind_height[i];
}
Best_of_gen_ind_num = best_ind_num;
Current_ave_fitness = sum_of_raw_fitness/POP_SIZE;
Current_ave_size    = (float) sum_of_size/POP_SIZE;
Current_ave_height  = (float) sum_of_height/POP_SIZE;
record_the_current_degree_of_evolution(                /*-------------*/
                     Raw_fitness[Best_of_gen_ind_num], /* report to   */
                     Num_of_hits[Best_of_gen_ind_num], /* the module  */
                     Current_ave_fitness,              /*"observe_run"*/
                     Current_ave_size,                 /*-------------*/
                     Current_ave_height);

                           /*----------------------------------*/
                           /* renew the best-so-far individual */
                           /* and the best-so-far fitness      */
if (generation==0) {       /*----------------------------------*/
  best_so_far_fitness = Raw_fitness[Best_of_gen_ind_num];
  best_so_far_individual = copy_tree(Individual[Best_of_gen_ind_num]);
}else if (Raw_fitness[Best_of_gen_ind_num]<best_so_far_fitness) {
```

```
    best_so_far_fitness = Raw_fitness[Best_of_gen_ind_num];
    free_tree(best_so_far_individual);
    best_so_far_individual = copy_tree(Individual[Best_of_gen_ind_num]);
  }

  /*----------------------------------------------*/
  /* decide wether the search should be terminated */
  /*----------------------------------------------*/
  if (Num_of_hits[Best_of_gen_ind_num]==Num_fitness_cases
      || generation >= MAX_GENERATION)
    break;

  /*----------------------*/
  /* alternate generation */
  /*----------------------*/
  for (i=0; i<POP_SIZE; i++)
                                      /*-----------*/
                                      /* selection */
                                      /*-----------*/
    Next_individual[i] = copy_tree(Individual[select_min_by_tournament(Raw_fitness)]);
          #if defined(TRACEIND) || defined(TRACEFIT) || defined(TRACESEL) \
              || defined(TRACECROSS) || defined(TRACEMUT) || defined(TRACEALL)
          printf("------------------------------\n\n");
          #endif

  for (i=0; i<POP_SIZE; ) {
                     /*-----------------------------------------------*/
                     /* perform one operation                         */
                     /* among crossover, mutation and reproduction */
                     /*-----------------------------------------------*/
    fraction = rand_float();
    if (fraction < branch_prob_crossover_func_pt && i<POP_SIZE-1) {
                                      /*----------------------------*/
                                      /* exchange any subtrees that */
                                      /* have two or more nodes     */
                                      /*----------------------------*/
          #if defined(TRACECROSS) || defined(TRACEALL)
          printf("***** Next_individual[%d] and Next_individual[%d]"
                 " are mated. *****\n"
                 "  ==> crossover_at_func_point\n"
                 , i, i+1);
          #endif
      crossover_at_func_point(&Next_individual[i], &Next_individual[i+1]);
      i += 2;
    }else if (fraction < branch_prob_crossover && i<POP_SIZE-1) {
                                      /*----------------------*/
                                      /* exchange any subtrees */
                                      /*----------------------*/
          #if defined(TRACECROSS) || defined(TRACEALL)
          printf("***** Next_individual[%d] and Next_individual[%d]"
                 " are mated. *****\n"
                 "  ==> crossover_at_any_point\n"
                 , i, i+1);
          #endif
      crossover_at_any_point(&Next_individual[i], &Next_individual[i+1]);
      i += 2;
```

```
      }else if (fraction < branch_prob_crossover_or_mutation) {
                                    /*--------*/
                                    /* mutate */
                                    /*--------*/
          #if defined(TRACEMUT) || defined(TRACEALL)
          printf("***** Next_individual[%d] is mutated."
                 "(mutation_point) *****\n", i);
          #endif
        mutation_point(&Next_individual[i]);
        ++i;
      }else {                       /*----------*/
                                    /* reproduce */
                                    /*-----------*/
          #if defined(TRACEMUT) || defined(TRACEALL)
          printf("***** Next_individual[%d] is reproduced. *****\n", i);
          #endif
        ++i;
      }
    }
                                            /*----------------------*/
    for (i=0; i<POP_SIZE; i++)              /* alternate generation */
      free_tree(Individual[i]);             /*----------------------*/
    temp            = Individual;
    Individual      = Next_individual;
    Next_individual = temp;
          #if defined(TRACEIND) || defined(TRACEFIT) || defined(TRACESEL) \
              || defined(TRACECROSS) || defined(TRACEMUT) || defined(TRACEALL)
          printf("\n\n***** Each Individual[i] is replaced by"
                 " Next_individual[i]. *****\n\n");
          #endif
  }

  /*----------------------------*/
  /* post-processing after a run */
  /*----------------------------*/
  for (i=0; i<POP_SIZE; i++)
    free_tree(Individual[i]);
  Best_of_run_individual[run]  = best_so_far_individual;
  Best_of_run_raw_fitness[run] = best_so_far_fitness;
  if (best_so_far_fitness < Best_of_run_raw_fitness[Best_so_far_run])
    Best_so_far_run = run;      /*------------------------------------*/
  post_processing_after_a_run(); /* report to the module "observe_run" */
}                              /*------------------------------------*/

/*-------------------------------------------------------------*/
/* generate a text file that records how the GP runs proceed */
/*-------------------------------------------------------------*/
print_the_target_function(function_name);       /*----------------------*/
sprintf(Comment_on_overall_runs,                /* A storage of 500 byte */
        "### Problem ###\n\n"                   /* is available now.     */
        "Symbolic Regression (Target:  %s)\n"   /* ==>Notice that the    */
        "Num_fitness_cases = %d\n"              /*    comment should be  */
        "Var_lower_limit   = %e\n"              /*    shorter than 500 byte.*/
        "Var_upper_limit   = %e\n\n"            /*----------------------*/
        "### Additional Parameters ###\n\n"
        "Max_error_for_hit = %e\n"
```

```
          "mutaion_method    = point mutation\n",
          function_name, Num_fitness_cases,
          Var_lower_limit, Var_upper_limit,
          Max_error_for_hit);
  report_on_runs(Report_file, Comment_on_overall_runs, NULL);

  /*----------------------------*/
  /* output the best individual */
  /*----------------------------*/
  if (strcmp(Report_file, "stdout") == 0) {
    report_fp = stdout;
  }else if ((report_fp=fopen(Report_file, "a")) == NULL) {
    printf("Error: The report file \"%s\" cannot be opened. (Append)\n"
           "==> Execution was aborted.\n", Report_file);
    exit(1);
  }
  fprintf(report_fp,
          "#################################################################\n"
          "#   Best Individual(s)   #\n"
          "##########################\n\n"
          "### Best of Overall Runs ###\n"
          "(Raw Fitness = %e)\n",
          Best_of_run_raw_fitness[Best_so_far_run]);
  print_an_individual_linked(report_fp,
                             Best_of_run_individual[Best_so_far_run]);
  if (Best_individual_needed)
    for (run=0; run<NUM_RUNS; run++) {
      fprintf(report_fp,
              "\n### Best of %d-th run ###\n"
              "(Raw Fitness = %e)\n",
              run, Best_of_run_raw_fitness[run]);
      print_an_individual_linked(report_fp, Best_of_run_individual[run]);
    }

  if (strcmp(Report_file, "stdout") != 0)
    fclose(report_fp);
}


/*----------------------------------------------------------------------*/
/* a function   "initialize_all_modules"                                */
/* that initialize all modules (e.g. informing parameter values to      */
/* related modules, memory allocation, initialization of local variables)*/
/*----------------------------------------------------------------------*/
/* (parameters) num_fitness_cases : thenumber of fitness case           */
/*              var_lower_limit : the lower bound of variable of the     */
/*                                target function, which is used when    */
/*                                fitness cases are generated            */
/*              var_upper_limit : the upper bound of variable of the     */
/*                                target function, which is used when    */
/*                                fitness cases are generated            */
/*----------------------------------------------------------------------*/
static void initialize_all_modules(void)
{
  int i;
```

```
  get_symbol_information(&Arity_table, &Num_func, &Num_terminals);

  /*---------------------------------*/
  /* initialize other program modules */
  /*---------------------------------*/
  initialize_create_initial_pop_ramped_uniform(Arity_table,
                              Num_func, Num_terminals,
                              Param.min_size_of_initial_tree,
                              Param.max_size_of_initial_tree,
                              Param.allow_ephe_ran_const,
                              Param.restrict_num_ephe_ran_const,
                              Param.num_ephe_ran_const);
  initialize_fitness_linked_1(Param.max_ephe_ran_const,
                            Param.min_ephe_ran_const, /*---------------*/
                            Num_fitness_cases,   /* The target function*/
                            target_function,     /* and the related    */
                            Var_lower_limit,     /* parameters specified*/
                            Var_upper_limit,     /* by external static */
                            Max_error_for_hit);  /* variables will be  */
                                                 /* informed.          */
                                                 /*--------------------*/
  initialize_crossover(Param.max_height_after_crossover);
  initialize_mutation_point(MAX_ARITY, Arity_table,
                          Num_func, Num_terminals);
  initialize_observe_runs(&Param,
                        Smaller_fitness_is_better,
                        Observe_with_display);
  initialize_tournament_selection(Param.pop_size, Param.tournament_size);
  initialize_randomizer(Param.random_seed);

  /*---------------------------------*/
  /* initialize this module "main.c" */
  /*---------------------------------*/
  Individual       = (Tree *) malloc(sizeof(Tree)*POP_SIZE);
  Raw_fitness      = (float *) malloc(sizeof(float)*POP_SIZE);
  Num_of_hits      = (int *) malloc(sizeof(int)*POP_SIZE);
  Ind_size         = (int *) malloc(sizeof(int)*POP_SIZE);
  Ind_height       = (int *) malloc(sizeof(int)*POP_SIZE);
  Next_individual  = (Tree *) malloc(sizeof(Tree)*POP_SIZE);
  Best_of_run_individual  = (Tree *) malloc(sizeof(Tree)*NUM_RUNS);
  Best_of_run_raw_fitness = (float *) malloc(sizeof(float)*NUM_RUNS);
  Comment_on_overall_runs = (char *) malloc(sizeof(char)*500);
}
```

We omit to display all other program modules.

4–7 Tracing a Run with Small Population Size

By activating all lines for debugging, we can make the program in section 4–6 verbose, and so observe how the program works in detail:

```
[motoki@x205a]$ make sym_reg3 "CC=gcc -DSMALL -DTRACERESULT -DTRACEIND   \
                                 -DTRACESEL -DTRACECROSS -DTRACEMUT"
                                    compile related source codes and link them
```

We omit to display some lines written by "make" command.

```
[motoki@x205a]$ i386_arity4/sym_reg3          invoke the executable code "sym_reg3"
--------------------------------
GP parameters are set as follows:
--------------------------------
num_runs       = 1
pop_size       = 500
max_generation = 50
random_seed               = 1
random_generation_method = randomizer mt by matsumoto & nishimura
allow_ephe_ran_const        = no
min_ephe_ran_const          = 0.000000e+00
max_ephe_ran_const          = 1.000000e+00
restrict_num_ephe_ran_const = no
num_ephe_ran_const          = 100
max_height_of_initial_tree        = 6
max_height_after_crossover        = 17
max_height_of_replacement_subtree = 4
initial_pop_creation_method  = ramped uniform
min_size_of_initial_tree     = 3
max_size_of_initial_tree     = 25
thinning_rate_for_larger_ind = 0.210000
selection_method          = tournament selection
tournament_size           = 6
control_param_for_adj_fit = 1
crossover_rate_func_pt = 0.200000
crossover_rate_any_pt  = 0.500000
reproduction_rate      = 0.150000
mutation_rate          = 0.150000
Report_file            = report_file.default
Best_individual_needed = no
Observe_with_display   = yes

--------------------------------------------
To see how the program behaves,three GP parameters are changed as follows:
    pop_size                   = 10,
    max_generation             = 2,
    min_size_of_initial_tree   = 3,
```

```
    max_size_of_initial_tree   = 25;
then the execution trace are reported.

------------------------------------------------
Target function is f(x) = x^6 - 2*x^4 + x^2.
==> Fitness cases are generated as follows:
        x              f(x)
-1.000000e+00    0.000000e+00
-9.000000e-01    2.924101e-02
-8.000000e-01    8.294399e-02
-7.000000e-01    1.274490e-01
-6.000000e-01    1.474560e-01
-5.000000e-01    1.406250e-01
-4.000000e-01    1.128960e-01
-3.000000e-01    7.452901e-02
-2.000000e-01    3.686400e-02
-1.000000e-01    9.801000e-03
0.000000e+00    0.000000e+00
1.000000e-01    9.801000e-03
2.000000e-01    3.686400e-02
3.000000e-01    7.452901e-02
4.000000e-01    1.128960e-01
5.000000e-01    1.406250e-01
6.000000e-01    1.474560e-01
7.000000e-01    1.274490e-01
8.000000e-01    8.294399e-02
9.000000e-01    2.924101e-02
1.000000e+00    0.000000e+00
------------------------------------------------
From a given arity table, the following Table_of_possible_functions is obtained:
Table_of_possible_functions[0] = {}
Table_of_possible_functions[1] = {}
Table_of_possible_functions[2] = { 0  1  2  3 }
Table_of_possible_functions[3] = {}
Table_of_possible_functions[4] = { 4 }



*************************************************
*    Run (0)
*************************************************

======================================
   Generation  0
======================================
Individual[0]:
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(1,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
```

Here are individuals in the current generation.

```
             1  )  )  )  )  )
  ==> fitness           = 1.778540e-01
      num_of_hits       = 0
      num_of_nodes      = 23
      num_of_func_nodes = 11
      height            = 10

-------------------------------
Individual[1]:
+(*(-(1,
      -(-(1,
          *(*(x,
              *(1,
                +(x,
                  x  )  )  ),
            1  )  ),
        x  )  ),
    x  ),
  1  )
  ==> fitness           = 1.294114e+00
      num_of_hits       = 1
      num_of_nodes      = 19
      num_of_func_nodes = 9
      height            = 9

-------------------------------
Individual[2]:
iflte(pdiv(-(*(x,
               pdiv(x,
                    -(1,
                      pdiv(x,
                           *(1,
                             -(1,
                               1  )  )  )  )  )  ),
             x  ),
           x  ),
      x,
      1,
      x  )
  ==> fitness           = 7.380952e-01
      num_of_hits       = 1
      num_of_nodes      = 21
      num_of_func_nodes = 9
      height            = 9

-------------------------------
Individual[3]:
+(x,
  iflte(+(x,
          1  ),
        iflte(1,
              +(1,
                +(x,
                  1  )  ),
              iflte(x,
                    1,
```

```
                    x,
                    1  ),
              +(x,
                1  )  ),
        1,
        x  )  )
  ==> fitness           = 1.047619e+00
      num_of_hits       = 1
      num_of_nodes      = 23
      num_of_func_nodes = 8
      height            = 5

-------------------------------
Individual[4]:
+(pdiv(1,
       1  ),
  x  )
  ==> fitness           = 9.274472e-01
      num_of_hits       = 1
      num_of_nodes      = 5
      num_of_func_nodes = 2
      height            = 2

-------------------------------
Individual[5]:
iflte(x,
      *(1,
        -(+(-(-(1,
                x  ),
              +(x,
                1  )  ),
            1  ),
          pdiv(x,
               -(-(1,
                   1  ),
                 x  )  )  )  ),
      x,
      -(1,
        x  )  )
  ==> fitness           = 3.904762e-01
      num_of_hits       = 2
      num_of_nodes      = 25
      num_of_func_nodes = 11
      height            = 6

-------------------------------
Individual[6]:
-(pdiv(+(1,
         +(*(*(x,
               *(+(1,
                   1  ),
                 1  )  ),
             pdiv(1,
                  1  )  ),
           x  )  ),
       +(x,
         1  )  ),
```

```
  x  )
  ==> fitness           = 2.047754e+00
      num_of_hits       = 0
      num_of_nodes      = 21
      num_of_func_nodes = 10
      height            = 8

-------------------------------
Individual[7]:
pdiv(pdiv(x,
          pdiv(-(x,
                 iflte(-(x,
                         x  ),
                       x,
                       1,
                       x  )  ),
               1  )  ),
     pdiv(-(+(x,
              1  ),
            1  ),
          -(x,
            1  )  )  )
  ==> fitness           = 2.322194e+00
      num_of_hits       = 0
      num_of_nodes      = 23
      num_of_func_nodes = 10
      height            = 6

-------------------------------
Individual[8]:
-(pdiv(1,
       x  ),
  +(*(-(x,
        x  ),
      pdiv(-(1,
             x  ),
           x  )  ),
    1  )  )
  ==> fitness           = 2.789494e+00
      num_of_hits       = 2
      num_of_nodes      = 15
      num_of_func_nodes = 7
      height            = 5

-------------------------------
Individual[9]:
*(-(1,
    1  ),
  +(x,
    +(1,
      1  )  )  )
  ==> fitness           = 7.255287e-02
      num_of_hits       = 5
      num_of_nodes      = 9
      num_of_func_nodes = 4
      height            = 3
```

```
-------------------------------
0-th Generation of 0-th Run:
===> Best_of_gen_fitness           = 7.255287e-02
     Hits_num_of_best_of_gen ind. = 5
     Ave_of_gen_fitness            = 1.180760e+00
     Ave_of_gen_size               = 1.840000e+01
     Ave_of_gen_height             = 6.300000e+00
--------------------------------------
*** Best_so_far_fitness_of_current_run is renewed. ***
```

Selection operation started.

```
Tournament selection of size 6.  ==> Indididual[5] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[0] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[2] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[0] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[4] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[9] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[0] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[5] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[9] is selected and reproduced.
Tournament selection of size 6.  ==> Indididual[4] is selected and reproduced.
-------------------------------
```

Reproduction

```
***** Next_individual[0] is reproduced. *****
```

Crossover by exchanging any two subtrees that have two or more nodes

```
***** Next_individual[1] and Next_individual[2] are mated. *****
  ==> crossover_at_func_point
Parent (1)
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(1,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
             1  )  )  )  )  )
------
Randomly generated node number is 0.
==> Selected subtree is as follows:
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(1,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
```

```
           +(1,
             1  )  )  )  )  )
------
Parent (2)
iflte(pdiv(-(*(x,
               pdiv(x,
                    -(1,
                      pdiv(x,
                           *(1,
                             -(1,
                               1  )  )  )  )  )  ),
             x  ),
           x  ),
      x,
      1,
      x  )
------
Randomly generated node number is 6.
==> Selected subtree is as follows:
pdiv(x,
     *(1,
       -(1,
         1  )  )  )
------
===> Child (1)
pdiv(x,
     *(1,
       -(1,
         1  )  )  )
------
===> Child (2)
iflte(pdiv(-(*(x,
               pdiv(x,
                    -(1,
                      pdiv(1,
                           -(1,
                             -(x,
                               +(pdiv(1,
                                      +(pdiv(1,
                                             *(x,
                                               -(1,
                                                 *(1,
                                                   1  )  )  )  ),
                                        x  )  ),
                                 +(1,
                                   1  )  )  )  )  )  )  )  ),
             x  ),
           x  ),
      x,
      1,
      x  )
--------------------------------------------------
```

Mutation

```
***** Next_individual[3] is mutated.(mutation_point) *****

Before mutation:
pdiv(1,
     -(1,
```

```
       -(x,
         +(pdiv(1,
                +(pdiv(1,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
             1  )  )  )  )  )
------
Randomly generated node number is 11.
==> Selected subtree is as follows:
1
------
By point mutation, the subtree is changed to:
x
------
==> After mutation:
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(x,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
             1  )  )  )  )  )
------------------------------------
***** Next_individual[4] and Next_individual[5] are mated. *****
  ==> crossover_at_any_point
```

Crossover by exchanging any two subtrees

```
Parent (1)
+(pdiv(1,
       1  ),
  x  )
------
Randomly generated node number is 1.
==> Selected subtree is as follows:
pdiv(1,
     1  )
------
Parent (2)
*(-(1,
    1  ),
  +(x,
    +(1,
      1  )  )  )
------
Randomly generated node number is 7.
==> Selected subtree is as follows:
1
------
===> Child (1)
```

```
+(1,
  x  )
------
===> Child (2)
*(-(1,
    1  ),
  +(x,
    +(pdiv(1,
           1  ),
      1  )  )  )
--------------------------------------------------
***** Next_individual[6] is reproduced. *****
```

Reproduction

```
***** Next_individual[7] is reproduced. *****
```

Reproduction

```
***** Next_individual[8] is mutated.(mutation_point) *****
```

Mutation

```
Before mutation:
*(-(1,
    1  ),
  +(x,
    +(1,
      1  )  )  )
------
Randomly generated node number is 3.
==> Selected subtree is as follows:
1
------
By point mutation, the subtree is changed to:
x
------
==> After mutation:
*(-(1,
    x  ),
  +(x,
    +(1,
      1  )  )  )
------------------------------------
***** Next_individual[9] is mutated.(mutation_point) *****
```

Mutation

```
Before mutation:
+(pdiv(1,
       1  ),
  x  )
------
Randomly generated node number is 2.
==> Selected subtree is as follows:
1
------
By point mutation, the subtree is changed to:
1
------
==> After mutation:
+(pdiv(1,
       1  ),
  x  )
```

```
------------------------------------


***** Each Individual[i] is replaced by Next_individual[i]. *****


======================================
   Generation  1
======================================
Individual[0]:
iflte(x,
      *(1,
        -(+(-(-(1,
                x  ),
              +(x,
                1  )  ),
            1  ),
          pdiv(x,
               -(-(1,
                   1  ),
                 x  )  )  )  ),
      x,
      -(1,
        x  )  )
  ==> fitness           = 3.904762e-01
      num_of_hits       = 2
      num_of_nodes      = 25
      num_of_func_nodes = 11
      height            = 6

-------------------------------
Individual[1]:
pdiv(x,
     *(1,
       -(1,
         1  )  )  )
  ==> fitness           = 9.274470e-01
      num_of_hits       = 0
      num_of_nodes      = 7
      num_of_func_nodes = 3
      height            = 3

-------------------------------
Individual[2]:
iflte(pdiv(-(*(x,
               pdiv(x,
                    -(1,
                      pdiv(1,
                           -(1,
                             -(x,
                               +(pdiv(1,
                                      +(pdiv(1,
                                             *(x,
                                               -(1,
                                                 *(1,
                                                   1  )  )  )  ),
                                        x  )  ),
```

Here are individuals in the current generation.

```
                                +(1,
                                  1  )  )  )  )  )  )  )  ),
               x  ),
             x  ),
        x,
        1,
        x  )
    ==> fitness           = 8.798280e-01
        num_of_hits       = 1
        num_of_nodes      = 37
        num_of_func_nodes = 17
        height            = 16

-------------------------------
Individual[3]:
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(x,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
             1  )  )  )  )  )
    ==> fitness           = 1.778540e-01
        num_of_hits       = 0
        num_of_nodes      = 23
        num_of_func_nodes = 11
        height            = 10

-------------------------------
Individual[4]:
+(1,
  x  )
    ==> fitness           = 9.274472e-01
        num_of_hits       = 1
        num_of_nodes      = 3
        num_of_func_nodes = 1
        height            = 1

-------------------------------
Individual[5]:
*(-(1,
    1  ),
  +(x,
    +(pdiv(1,
           1  ),
      1  )  )  )
    ==> fitness           = 7.255287e-02
        num_of_hits       = 5
        num_of_nodes      = 11
        num_of_func_nodes = 5
        height            = 4
```

```
-------------------------------
Individual[6]:
pdiv(1,
     -(1,
       -(x,
         +(pdiv(1,
                +(pdiv(1,
                       *(x,
                         -(1,
                           *(1,
                             1  )  )  )  ),
                  x  )  ),
           +(1,
             1  )  )  )  )  )
  ==> fitness           = 1.778540e-01
      num_of_hits       = 0
      num_of_nodes      = 23
      num_of_func_nodes = 11
      height            = 10

-------------------------------
Individual[7]:
iflte(x,
      *(1,
        -(+(-(-(1,
                x  ),
              +(x,
                1  )  ),
            1  ),
          pdiv(x,
               -(-(1,
                   1  ),
                 x  )  )  )  ),
      x,
      -(1,
        x  )  )
  ==> fitness           = 3.904762e-01
      num_of_hits       = 2
      num_of_nodes      = 25
      num_of_func_nodes = 11
      height            = 6

-------------------------------
Individual[8]:
*(-(1,
    x  ),
  +(x,
    +(1,
      1  )  )  )
  ==> fitness           = 1.560781e+00
      num_of_hits       = 1
      num_of_nodes      = 9
      num_of_func_nodes = 4
      height            = 3

-------------------------------
```

```
Individual[9]:
+(pdiv(1,
       1  ),
  x  )
  ==> fitness           = 9.274472e-01
      num_of_hits       = 1
      num_of_nodes      = 5
      num_of_func_nodes = 2
      height            = 2

-------------------------------
1-th Generation of 0-th Run:
===> Best_of_gen_fitness          = 7.255287e-02
     Hits_num_of_best_of_gen ind. = 5
     Ave_of_gen_fitness           = 6.432163e-01
     Ave_of_gen_size              = 1.680000e+01
     Ave_of_gen_height            = 6.100000e+00
--------------------------------------
```

We omit to display many intermediate lines.

```
-------------------------------
2-th Generation of 0-th Run:
===> Best_of_gen_fitness          = 7.255287e-02
     Hits_num_of_best_of_gen ind. = 5
     Ave_of_gen_fitness           = 7.050346e-01
     Ave_of_gen_size              = 1.440000e+01
     Ave_of_gen_height            = 4.900000e+00
--------------------------------------
***** 0-th Run is terminated at 2th Generation. *****

*** Best_so_far_fitness is renewed. ***
```

4–8 Experimental Results

The program in section 4–6 generate a report file that shows the run parameters we adopted, the best individual, and how the best (-of-generation) fitness, the average fitness and the average size vary with generation.

Executing the program on a parameter environment

$$\begin{aligned} \textbf{the\_number\_of\_runs} &= \textbf{100}, \\ \textbf{population\_size} &= \textbf{500}, \\ \textbf{maximum\_generation\_number} &= \textbf{50}, \\ &\cdots\cdots\cdots, \end{aligned}$$

we obtain the following report file:

```
[motoki@x205a]$ more report_file.reg3_need_best_ind
                        display contents in the file "report_file.reg3_need_best_ind"
#####################
#___GP_Parameters___#
#####################

num_runs       = 100
pop_size       = 500
max_generation = 50
random_seed              = 1
random_generation_method = randomizer mt by matsumoto & nishimura
allow_ephe_ran_const        = no
min_ephe_ran_const          = 0.000000e+00
max_ephe_ran_const          = 1.000000e+00
restrict_num_ephe_ran_const = no
num_ephe_ran_const          = 100
max_height_of_initial_tree        = 6
max_height_after_crossover        = 17
max_height_of_replacement_subtree = 4
initial_pop_creation_method  = ramped uniform
min_size_of_initial_tree     = 3
max_size_of_initial_tree     = 25
thinning_rate_for_larger_ind = 0.210000
selection_method          = tournament selection
tournament_size           = 6
control_param_for_adj_fit = 1
crossover_rate_func_pt = 0.200000
crossover_rate_any_pt  = 0.500000
reproduction_rate      = 0.150000
mutation_rate          = 0.150000

################################
#___Comments_on_Overall_Runs___#
################################

### Problem ###

Symbolic Regression (Target:  f(x) = x^6 - 2*x^4 + x^2)
Num_fitness_cases = 21
Var_lower_limit   = -1.000000e+00
Var_upper_limit   = 1.000000e+00

### Additional Parameters ###
```

```
Max_error_for_hit = 1.000000e-02
mutaion_method    = point mutation


#################################################################
#___Results_on_Runs___#
#######################

###_0-th_Run_###

#Gene-    _             Hits#_of
#ration  Best_Fitness  best_ind.  Ave._Fitness    Ave.Size    Ave.Height
    0      7.255e-02       5        1.759e+03    1.447e+01    4.846e+00
    1      6.768e-02       6        6.654e-01    1.273e+01    4.274e+00
    2      6.768e-02       6        7.990e+03    1.216e+01    4.174e+00
    3      6.306e-02       6        3.727e-01    1.280e+01    4.324e+00
    4      5.285e-02       9        3.610e-01    1.508e+01    4.898e+00
    5      5.356e-02       7        3.541e-01    1.822e+01    5.498e+00
```

We omit to display many intermediate lines.

```
###_90-th_Run_###

#Gene-    _             Hits#_of
#ration  Best_Fitness  best_ind.  Ave._Fitness    Ave.Size    Ave.Height
    0      7.255e-02       5        2.197e+01    1.389e+01    4.598e+00
    1      5.285e-02       9        2.238e+03    1.205e+01    4.078e+00
    2      5.285e-02       9        4.379e-01    1.245e+01    4.108e+00
    3      5.285e-02       9        3.765e-01    1.384e+01    4.504e+00
    4      5.285e-02       9        2.904e+04    1.614e+01    5.090e+00
    5      5.285e-02       9        1.617e+05    1.992e+01    6.190e+00
    6      5.285e-02       9        2.682e-01    2.395e+01    7.058e+00
    7*     2.161e-09      21        7.989e+03    2.624e+01    7.414e+00

###_91-th_Run_###

#Gene-    _             Hits#_of
#ration  Best_Fitness  best_ind.  Ave._Fitness    Ave.Size    Ave.Height
    0      7.164e-02       5        9.802e+01    1.394e+01    4.632e+00
    1      5.285e-02       9        2.619e+00    1.357e+01    4.308e+00
    2      6.429e-02       5        4.433e-01    1.295e+01    4.090e+00
    3      6.310e-02       6        3.380e-01    1.425e+01    4.206e+00
    4      6.120e-02       5        2.586e-01    1.833e+01    4.748e+00
    5      4.435e-02       7        2.211e-01    2.585e+01    6.220e+00
    6      3.823e-02       9        1.429e-01    3.269e+01    8.090e+00
    7      3.823e-02       9        1.480e-01    3.450e+01    8.814e+00
    8      3.823e-02       9        1.363e-01    3.556e+01    9.034e+00
    9      3.823e-02       9        1.452e-01    3.717e+01    8.878e+00
   10      3.444e-02       7        1.485e-01    3.868e+01    8.738e+00
   11      3.444e-02       7        1.270e-01    3.707e+01    8.352e+00
   12      3.320e-02      11        1.213e-01    3.600e+01    8.186e+00
   13      1.247e-02      14        1.292e-01    3.327e+01    7.878e+00
   14      1.247e-02      14        1.214e-01    3.556e+01    8.408e+00
```

15	1.247e-02	14	1.075e-01	4.199e+01	9.700e+00
16*	5.515e-09	21	9.391e-02	4.879e+01	1.037e+01

###_92-th_Run_###

#Gene- #ration	_ Best_Fitness	Hits#_of best_ind.	Ave._Fitness	Ave.Size	Ave.Height
0	7.255e-02	5	4.677e+00	1.421e+01	4.710e+00
1	7.255e-02	5	1.246e+04	1.317e+01	4.300e+00
2	7.255e-02	5	4.810e-01	1.236e+01	3.894e+00
3	7.255e-02	5	4.410e-01	1.223e+01	3.742e+00
4	7.255e-02	5	3.772e-01	1.426e+01	4.088e+00
5	7.255e-02	5	5.011e-01	1.878e+01	4.870e+00
6	7.255e-02	5	5.217e-01	2.225e+01	5.434e+00
7	7.255e-02	5	4.678e-01	2.595e+01	6.024e+00
8	7.255e-02	5	2.823e+04	2.770e+01	6.350e+00
9	7.255e-02	5	1.198e+00	2.766e+01	6.214e+00
10	7.255e-02	5	1.563e+00	3.043e+01	6.540e+00
11	7.255e-02	5	1.080e+01	3.425e+01	7.340e+00
12	7.255e-02	5	1.537e+01	3.976e+01	8.534e+00
13	7.233e-02	5	2.132e+01	5.078e+01	1.036e+01
14	7.233e-02	5	6.611e+01	5.936e+01	1.135e+01
15	7.233e-02	5	9.394e+02	6.002e+01	1.143e+01
16	7.233e-02	5	7.319e+03	7.283e+01	1.297e+01
17	7.233e-02	5	2.006e+04	8.172e+01	1.410e+01
18	7.212e-02	5	1.364e+04	7.608e+01	1.369e+01
19	7.212e-02	5	3.479e+04	7.350e+01	1.347e+01
20	7.177e-02	5	2.146e+05	7.281e+01	1.352e+01
21	7.177e-02	5	3.389e+06	7.590e+01	1.393e+01
22	7.177e-02	5	2.933e+05	7.827e+01	1.407e+01
23	7.177e-02	5	8.922e+06	7.820e+01	1.382e+01
24	7.174e-02	5	2.886e+05	8.052e+01	1.389e+01
25	7.174e-02	5	5.018e+04	8.252e+01	1.400e+01
26	7.170e-02	5	4.689e+05	8.454e+01	1.422e+01
27	7.169e-02	5	2.850e+05	8.636e+01	1.435e+01
28	7.167e-02	5	3.453e+05	8.760e+01	1.439e+01
29	7.167e-02	5	1.026e+05	8.580e+01	1.432e+01
30	7.167e-02	5	1.440e+11	8.561e+01	1.412e+01
31	7.167e-02	5	5.851e+12	8.564e+01	1.395e+01
32	7.167e-02	5	1.937e+11	8.924e+01	1.421e+01
33	7.164e-02	5	1.781e+12	9.285e+01	1.403e+01
34	7.164e-02	5	1.638e+12	9.503e+01	1.398e+01
35	7.162e-02	5	4.263e+06	9.770e+01	1.420e+01
36	7.149e-02	5	2.365e+06	9.849e+01	1.415e+01
37	7.149e-02	5	5.663e+05	1.008e+02	1.441e+01
38	7.147e-02	5	4.218e+05	1.042e+02	1.455e+01
39	7.141e-02	5	5.840e+06	1.071e+02	1.467e+01
40	7.084e-02	5	3.028e+06	1.087e+02	1.465e+01
41	7.080e-02	5	1.278e+08	1.105e+02	1.483e+01
42	7.023e-02	5	2.476e+07	1.133e+02	1.480e+01
43	7.023e-02	5	2.998e+08	1.206e+02	1.504e+01
44	7.023e-02	5	3.631e+07	1.299e+02	1.533e+01
45	6.965e-02	5	1.430e+08	1.376e+02	1.581e+01
46	6.929e-02	5	5.618e+09	1.404e+02	1.595e+01
47	6.880e-02	6	2.498e+08	1.453e+02	1.623e+01
48	6.879e-02	6	5.608e+09	1.505e+02	1.654e+01

```
   49      6.849e-02        6       2.039e+09    1.581e+02    1.670e+01
   50*     6.845e-02        6       2.622e+09    1.647e+02    1.675e+01

###_93-th_Run_###
```

We omit to display many intermediate lines.

```
###_Best_Run_###

  90-th Run
  Best Fitness =2.161157e-09

###_Average_Run_###

  #Note: (1)We assume that Best_Fitness and Hits#_of_best_ind. of
  #         final generation are kept after termination of run;
  #         so according to these items, the averages are
  #         calculated under this assumption.
  #      (2)According to the remaining items Ave.Fitness, Ave.Size
  #         and Ave.Height, the averages are calculated over all
  #         populations that was really arisen in some GP run.

#Gene-   _             Hits#_of
#ration  Best_Fitness  best_ind.  Ave._Fitness    Ave.Size    Ave.Height
    0     7.056e-02    5.290e+00   9.133e+03    1.400e+01    4.620e+00
    1     6.705e-02    5.920e+00   9.898e+02    1.307e+01    4.211e+00
    2     6.524e-02    6.020e+00   1.214e+03    1.312e+01    4.178e+00
    3     6.390e-02    6.170e+00   4.029e+03    1.410e+01    4.402e+00
    4     6.295e-02    6.100e+00   6.763e+03    1.621e+01    4.980e+00
    5     6.014e-02    6.620e+00   1.148e+04    1.869e+01    5.654e+00
    6     5.909e-02    6.660e+00   2.764e+08    2.102e+01    6.253e+00
    7     5.345e-02    7.820e+00   1.183e+05    2.340e+01    6.803e+00
    8     5.088e-02    8.560e+00   2.233e+04    2.592e+01    7.327e+00
    9     4.951e-02    8.820e+00   1.382e+05    2.790e+01    7.746e+00
   10     4.703e-02    9.300e+00   2.423e+08    3.051e+01    8.238e+00
   11     4.547e-02    9.430e+00   4.997e+10    3.393e+01    8.815e+00
   12     4.312e-02    1.002e+01   1.527e+09    3.700e+01    9.261e+00
   13     4.090e-02    1.043e+01   3.024e+09    3.999e+01    9.708e+00
   14     3.848e-02    1.081e+01   3.384e+09    4.301e+01    1.011e+01
   15     3.753e-02    1.096e+01   5.586e+13    4.629e+01    1.062e+01
   16     3.667e-02    1.117e+01   1.137e+09    4.995e+01    1.109e+01
   17     3.618e-02    1.148e+01   3.899e+08    5.355e+01    1.152e+01
   18     3.582e-02    1.144e+01   5.522e+08    5.733e+01    1.190e+01
   19     3.544e-02    1.156e+01   9.662e+08    6.059e+01    1.217e+01
   20     3.484e-02    1.168e+01   1.452e+12    6.315e+01    1.243e+01
   21     3.337e-02    1.197e+01   7.745e+08    6.535e+01    1.263e+01
   22     3.302e-02    1.199e+01   2.024e+11    6.797e+01    1.290e+01
   23     3.224e-02    1.235e+01   1.580e+13    7.171e+01    1.321e+01
   24     3.169e-02    1.249e+01   6.013e+16    7.479e+01    1.338e+01
   25     3.149e-02    1.257e+01   1.241e+16    7.611e+01    1.354e+01
   26     3.106e-02    1.271e+01   6.958e+14    7.795e+01    1.365e+01
   27     3.099e-02    1.275e+01   3.158e+16    8.119e+01    1.386e+01
   28     3.089e-02    1.278e+01   5.263e+13    8.437e+01    1.403e+01
```

29	3.067e-02	1.288e+01	3.376e+12	8.758e+01	1.422e+01
30	3.058e-02	1.284e+01	3.944e+13	9.071e+01	1.435e+01
31	3.048e-02	1.284e+01	2.511e+14	9.340e+01	1.445e+01
32	3.033e-02	1.284e+01	1.129e+13	9.569e+01	1.458e+01
33	3.010e-02	1.289e+01	1.212e+13	9.782e+01	1.468e+01
34	3.001e-02	1.299e+01	1.454e+11	9.954e+01	1.477e+01
35	2.983e-02	1.294e+01	2.613e+10	1.019e+02	1.481e+01
36	2.971e-02	1.308e+01	1.054e+12	1.052e+02	1.490e+01
37	2.947e-02	1.311e+01	1.336e+11	1.079e+02	1.498e+01
38	2.935e-02	1.314e+01	1.046e+12	1.096e+02	1.497e+01
39	2.921e-02	1.324e+01	1.159e+12	1.114e+02	1.499e+01
40	2.903e-02	1.326e+01	1.456e+10	1.131e+02	1.505e+01
41	2.894e-02	1.330e+01	1.105e+13	1.153e+02	1.509e+01
42	2.884e-02	1.332e+01	8.085e+10	1.179e+02	1.519e+01
43	2.874e-02	1.340e+01	1.115e+14	1.208e+02	1.530e+01
44	2.861e-02	1.344e+01	1.125e+13	1.237e+02	1.536e+01
45	2.850e-02	1.356e+01	1.588e+15	1.261e+02	1.545e+01
46	2.842e-02	1.360e+01	8.090e+11	1.273e+02	1.553e+01
47	2.838e-02	1.367e+01	1.754e+11	1.274e+02	1.557e+01
48	2.833e-02	1.369e+01	4.251e+10	1.282e+02	1.558e+01
49	2.830e-02	1.374e+01	3.398e+09	1.299e+02	1.564e+01
50	2.817e-02	1.380e+01	1.155e+13	1.312e+02	1.566e+01

###_Information_of_Performance_Curves_###

#	_	Expected_Num.of_ind.
#	_	to_be_processed
#Generation	Success_Rate	for_over_99%_success
0	0.0000000	-inf
1	0.0100000	4.590e+05
2	0.0100000	6.885e+05
3	0.0100000	9.180e+05
4	0.0100000	1.148e+06
5	0.0300000	4.560e+05
6	0.0300000	5.320e+05
7	0.1200000	1.480e+05
8	0.1600000	1.215e+05
9	0.1700000	1.250e+05
10	0.2000000	1.155e+05
11	0.2100000	1.200e+05
12	0.2400000	1.105e+05
13	0.2600000	1.120e+05
14	0.2800000	1.125e+05
15	0.2900000	1.120e+05
16	0.3000000	1.105e+05
17	0.3200000	1.080e+05
18	0.3300000	1.140e+05
19	0.3300000	1.200e+05
20	0.3300000	1.260e+05
21	0.3400000	1.320e+05
22	0.3500000	1.265e+05
23	0.3600000	1.320e+05
24	0.3800000	1.250e+05
25	0.3800000	1.300e+05
26	0.3900000	1.350e+05
27	0.3900000	1.400e+05

```
      28        0.3900000      1.450e+05
      29        0.3900000      1.500e+05
      30        0.3900000      1.550e+05
      31        0.3900000      1.600e+05
      32        0.3900000      1.650e+05
      33        0.3900000      1.700e+05
      34        0.3900000      1.750e+05
      35        0.3900000      1.800e+05
      36        0.3900000      1.850e+05
      37        0.3900000      1.900e+05
      38        0.3900000      1.950e+05
      39        0.3900000      2.000e+05
      40        0.3900000      2.050e+05
      41        0.3900000      2.100e+05
      42        0.4000000      2.150e+05
      43        0.4000000      2.200e+05
      44        0.4000000      2.250e+05
      45        0.4000000      2.300e+05
      46        0.4200000      2.115e+05
      47        0.4200000      2.160e+05
      48        0.4200000      2.205e+05
      49        0.4200000      2.250e+05
```

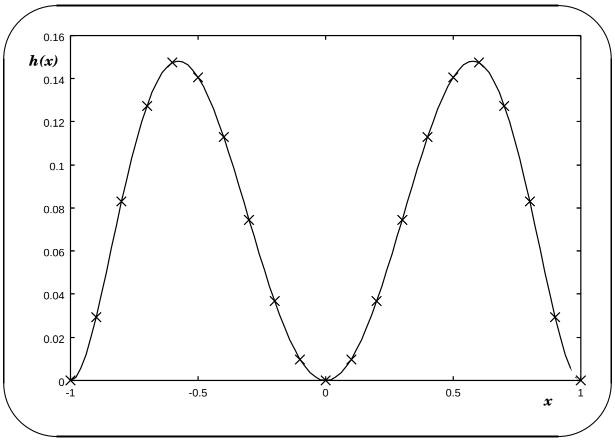
We have a 42% chance of finding a good solution.

```
#################################################################
#___Elapsed_Time___#
####################

                  Process Time     Real Time
Total Time        2.108e+02 sec   2.110e+02 sec
Time/Run          2.108e+00 sec   2.110e+00 sec
Time/Generation   5.851e-02 sec   5.856e-02 sec

###__Run_Environment_###

Date:  Tue Nov 20 16:15:03 2001

Host:  x205a.ml.ie.niigata-u.ac.jp


#################################################################
#   Best Individual(s)   #
##########################

### Best of Overall Runs ###
(Raw Fitness = 2.161157e-09)
*(*(*(-(x,
        *(x,
          *(1,
            *(*(1,
                x  ),
              x  )  )  )  ),
      -(1,
        x  )  ),
    1  ),
  *(x,
    +(1,
```

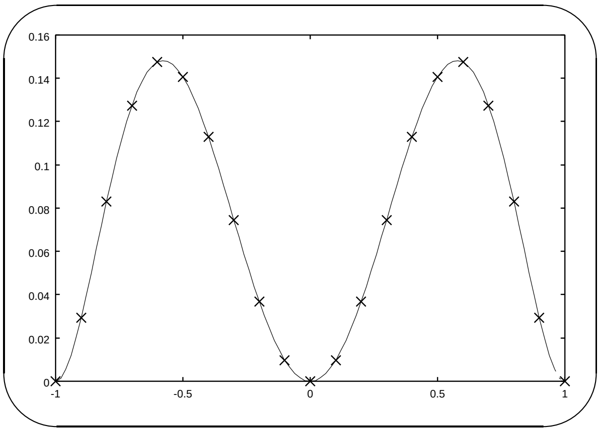
```
      -(*(1,
          x  ),
        -(1,
          1  )  )  )  )  )

### Best of 0-th run ###
(Raw Fitness = 3.385495e-02)
*(iflte(*(x,
          +(x,
            +(*(x,
```

We omit to display many intermediate lines.

```
### Best of 90-th run ###
(Raw Fitness = 2.161157e-09)
*(*(*(-(x,
        *(x,
          *(1,
            *(*(1,
                x  ),
              x  )  )  )  ),
      -(1,
        x  )  ),
    1  ),
  *(x,
    +(1,
      -(*(1,
          x  ),
        -(1,
          1  )  )  )  )  )
```

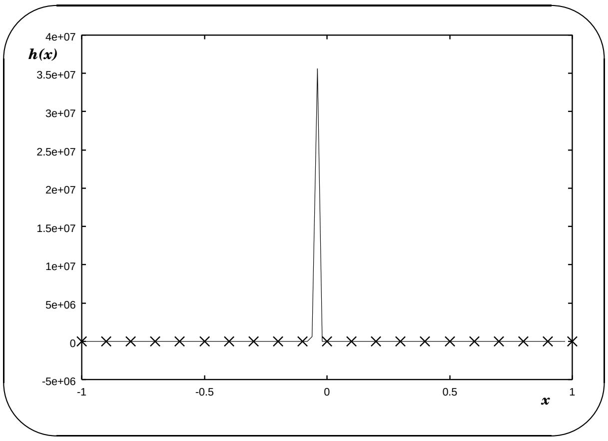



```
### Best of 91-th run ###
(Raw Fitness = 5.515259e-09)
*(*(iflte(pdiv(x,
               pdiv(x,
                    -(1,
                      *(x,
                        x  )  )  )  ),
          1,
          -(1,
            *(x,
              -(+(pdiv(1,
                       *(*(1,
                           1  ),
                         1  )  ),
                   x  ),
                 1  )  )  ),
          *(x,
            -(1,
              x  )  )  ),
    *(x,
      -(1,
```



```
        *(x,
          -(+(pdiv(+(1,
                     x  ),
                   *(x,
                     *(-(iflte(+(*(x,
                                   +(1,
                                     x  )  ),
                                 1  ),
                               x,
                               1,
                               x  ),
                         x  ),
                       x  )  )  ),
              x  ),
            1  )  )  )  )  ),
  x  )
```

```
### Best of 92-th run ###
(Raw Fitness = 6.844731e-02)
```

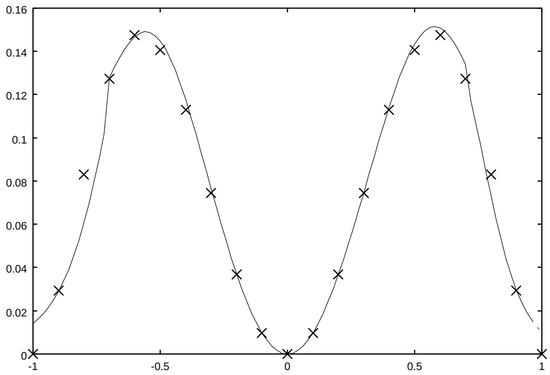



```
*(-(x,
    +(*(-(x,
          +(*(pdiv(-(+(pdiv(x,
                            1  ),
                       iflte(-(+(1,
                                 x  ),
                               -(x,
                                 x  )  ),
                             x,
                             x,
                             pdiv(pdiv(1,
                                       *(x,
                                         pdiv(x,
                                              pdiv(pdiv(1,
                                                        1  ),
                                                   x  )  )  )  ),
                                  x  )  )  ),
                     x  ),
                   x  ),
              +(-(-(x,
                    1  ),
                  x  ),
                iflte(-(-(+(1,
                            x  ),
                          1  ),
                        1  ),
                      x,
                      1,
                      pdiv(pdiv(+(1,
                                  x  ),
                                1  ),
                           x  )  )  )  ),
            -(+(1,
                x  ),
              1  )  )  ),
        pdiv(pdiv(-(+(*(x,
                        1  ),
```

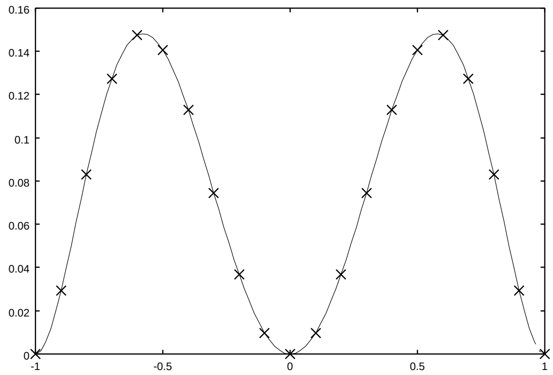
```
                    -(-(1,
                        1  ),
                    1  )  ),
                1  ),
            x  ),
        pdiv(x,
            1  )  )  ),
  -(+(1,
      x  ),
    1  )  )  ),
pdiv(pdiv(-(pdiv(pdiv(-(+(*(x,
                            1  ),
                        -(-(x,
                            1  ),
                        1  )  ),
                    1  ),
                x  ),
            pdiv(x,
                1  )  ),
        -(+(x,
            x  ),
          -(1,
            +(*(-(x,
                    1  ),
                +(-(-(x,
                        1  ),
                    x  ),
                iflte(-(-(+(x,
                            x  ),
                        1  ),
                    1  ),
                    x,
                    1,
                    pdiv(pdiv(1,
                            1  ),
                        x  )  )  )  ),
            -(-(+(*(x,
                    x  ),
                -(x,
                    1  )  ),
                1  ),
            1  )  )  )  )  ),
    x  ),
pdiv(1,)
    1  )  )  )

### Best of 93-th run ###
```

We omit to display many later lines.

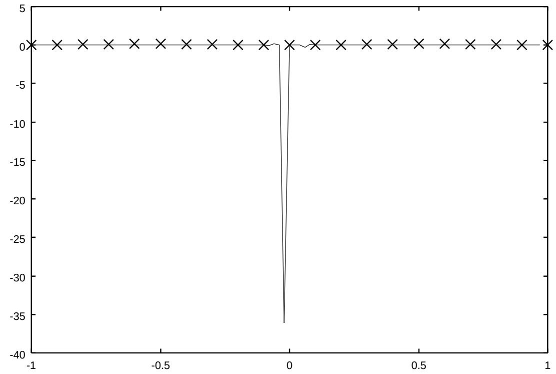
Best of 93-th run



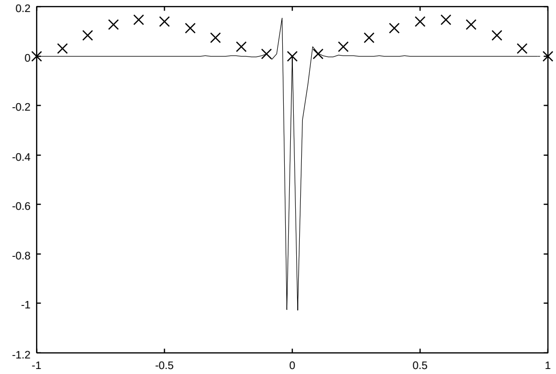
Best of 94-th run



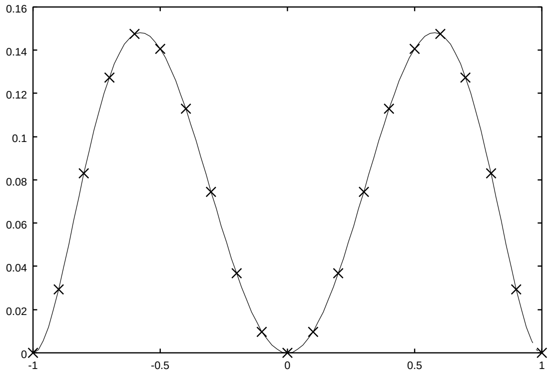
Best of 95-th run



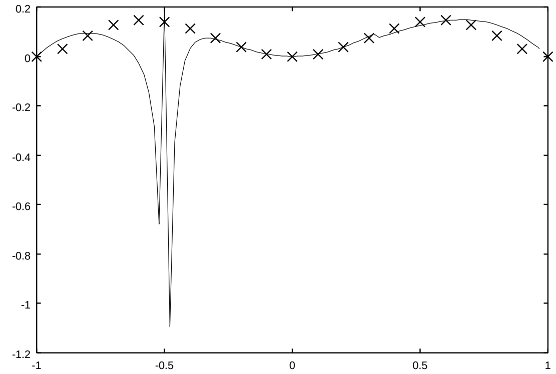
Best of 96-th run



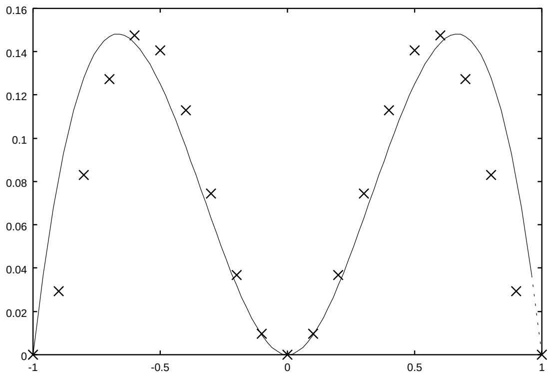
Best of 97-th run



Best of 98-th run



Best of 99-th run



IV. Adaptive Learning

Today, I'm going to share with you the basic feature of adaptive learning on artificial neural networks.

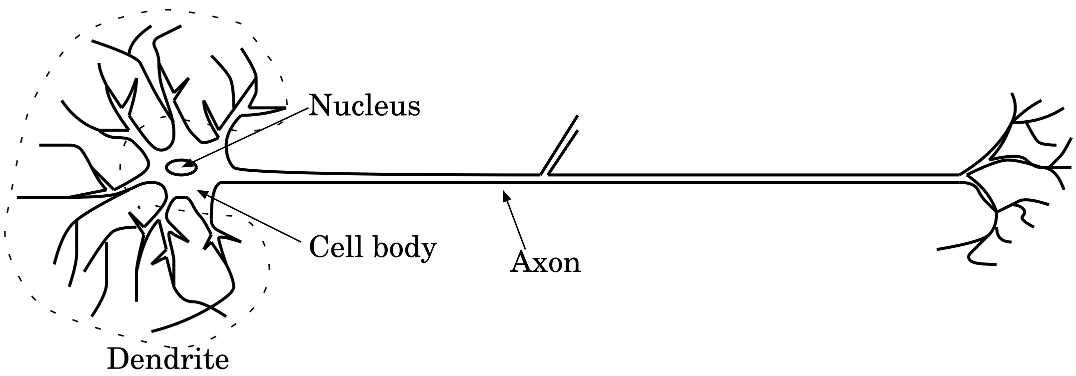
5 Training Perceptrons

References:
P.H.Winston(1992), Artificial Intelligence 3rd Edition, Addison-Wesley.
S.Russell&P.Norvig(1995), Artificial Intelligence A Modern Approach, Prentice hall.

5–1 How the Brain Works

The brain consists of many simple functional units called neurons.

A neuron consists of a cell body plus one "axon" and many "dendrites."



- **Dendrites are bushy trees around the cell body that receive influences from other neurons.**
- **The axon is a single long fiber that delivers the neuron's output to connections with other neurons.**
- **The Axon eventually branches into strands that connect to the dendrites or cell bodies of other neurons.**

  The connecting junction is called a synapse.

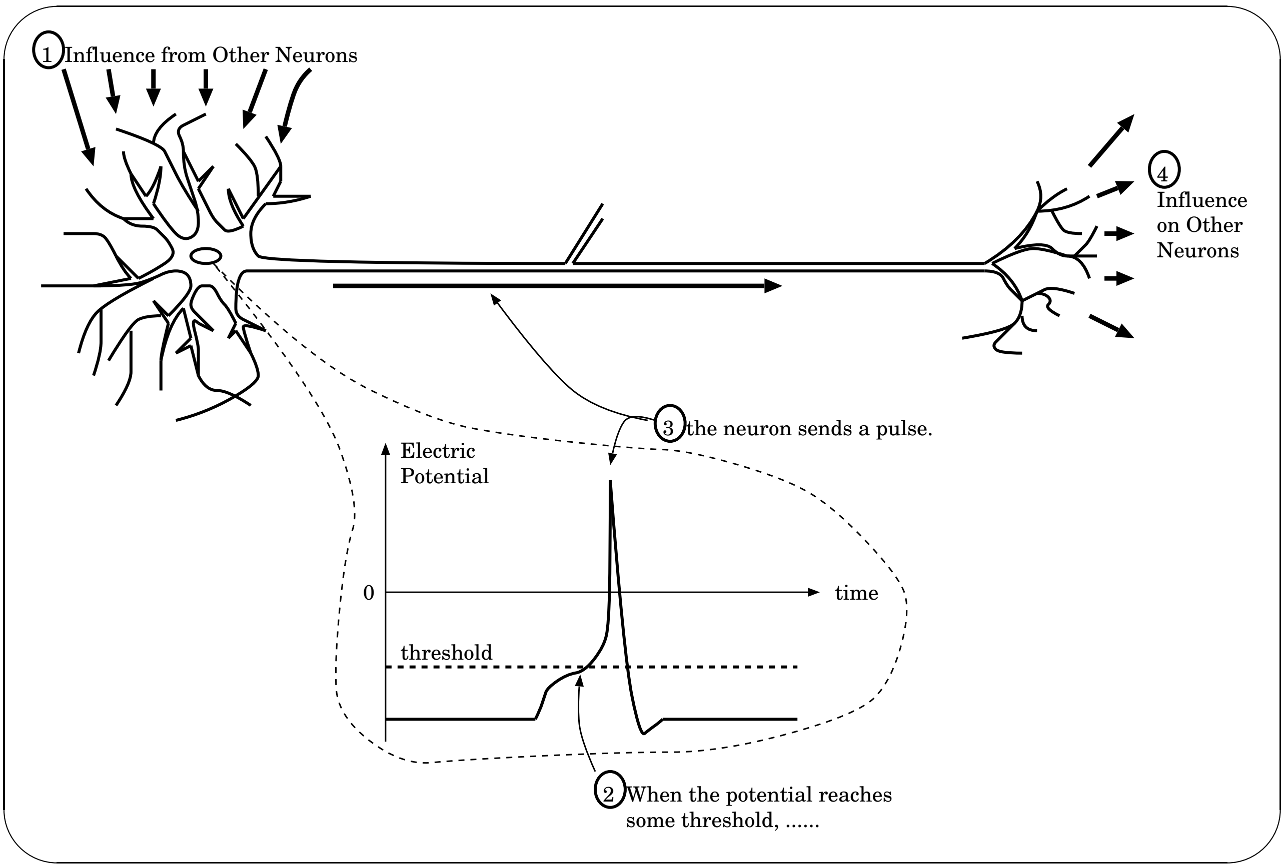
Each neuron works as follows :

(0) The neuron usually produce no output.

(1) The cell body has some electrical potential that are raised or lowered by signals from other neurons.

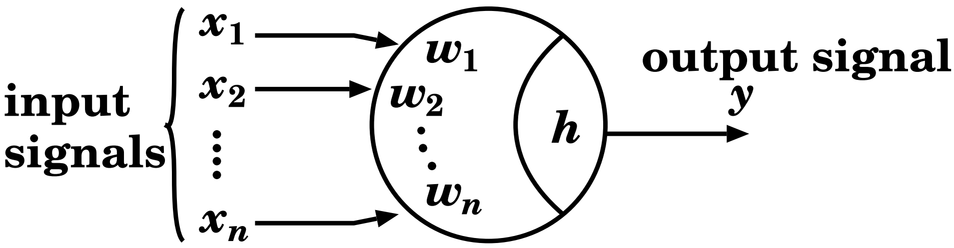
(2) When the potential reaches some threshold, the neuron sends a full-strength electrical pulse to the axon.

(This pulse will have influence on other neurons.)



5–2 A Neuron-like Element

We now consider a simple computing element that simulates a neuron:



$$u = \sum_{i=1}^{n} x_i w_i$$

$$y = \begin{cases} 1 & \textbf{if } u \geq h \\ 0 & \textbf{otherwise} \end{cases}$$

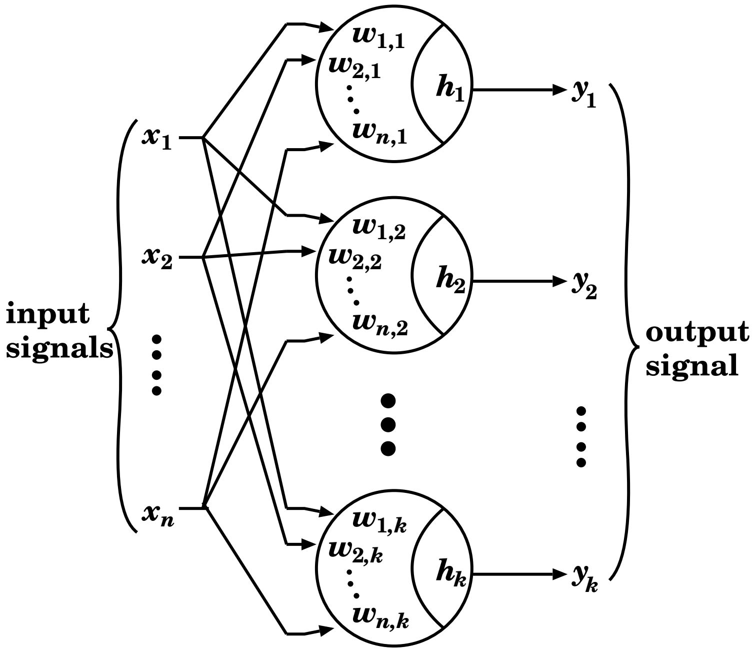
- **The quantity** x_i **models the stimulation from other neurons.**
- **The weights** w_i **model synaptic properties of whether and to what degree the stimulation** x_i **raise the potential of the neuron.**
- **The quantity** u **models the electric potential of the neuron.**
- **The quantity** h **models the threshold that gives a criterion of when the neuron send a electric pulse to an axon.**
- **The quantity** y **medels the stimulation that will influence on other neurons.**

But unlike real neurons,

- **This pseudo-neuron continues to give an output.**

5–3 Perceptrons — Single-Layer Feed-Forward Networks —

We first consider a class of networks, called perceptrons, that have the following network structure:

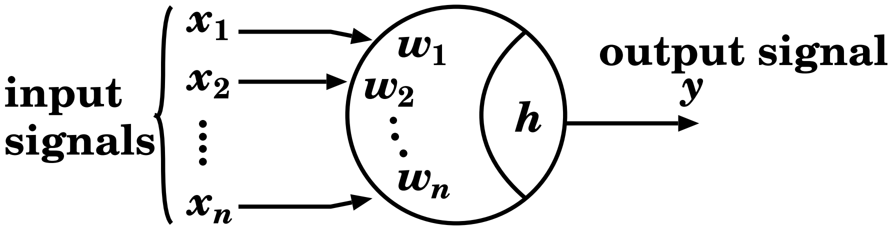


Notice that each output y_i is independent of the others.

Because each weight only affects one of the outputs.

⇨ **For each output y_j, we can adjust a group of weights and threshold $\{\ w_{1,j}, w_{2,j}, \ldots, w_{n,j}, h_j\ \}$ separately from other groups of weights and threshold.**

⇨ **With no loss of generality, we can limit an adjustment algorithm to perceptrons with single output.**



5–4 Adjusting Weights in Perceptrons

Most neural network learning algorithms follow the "current-best-hypothesis" scheme:

function NEURAL-NETWORK-LEARNING(*examples*) **returns** *network*

network ← a netwok with randomly assigned weights ;
repeat
  for each *e* **in** *examples* **do**
    O ← NEURAL-NETWORK-OUTPUT(*network, e*) ;
    T ← the observed(required) output values from *e* ;
    update the weights in *network* based on *e*, **O**, and **T**
  end
until all examples correctly predicted or stopping criterion is reached;
return *network*

(from S.Russell&P.Norvig,1995; p.577, Fig.19.11)

- **They search for a network that is consistent with the given examples (i.e. input-output relations).**
- **They hold a hypothesis network and repeatedly adjust it.**
- **When some given example is found to be unsatisfied by the current hypothesis network, small adjustments in weights are made to reduce the difference between the output of the network and the desired output.**

How to update the weights :

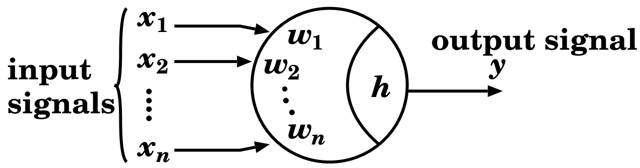
For perceptrons, it suffices to update the weights and threshold with the following rule:

$$\begin{aligned} w_j &\leftarrow w_j + \alpha \times x_j \times (t - y) \\ h &\leftarrow h - \alpha \times (t - y), \end{aligned}$$

where

- α **is a small positive constant called the learning rate,**
- x_j **denotes the j-th input value of an example,**
- t **denotes the required output for the inputs $(x_1, x_2, \ldots, x_n)$, and**
- y **denotes the output of the current hypothesis network for the inputs $(x_1, x_2, \ldots, x_n)$.**

These rules are seen to work correctly by the following way:



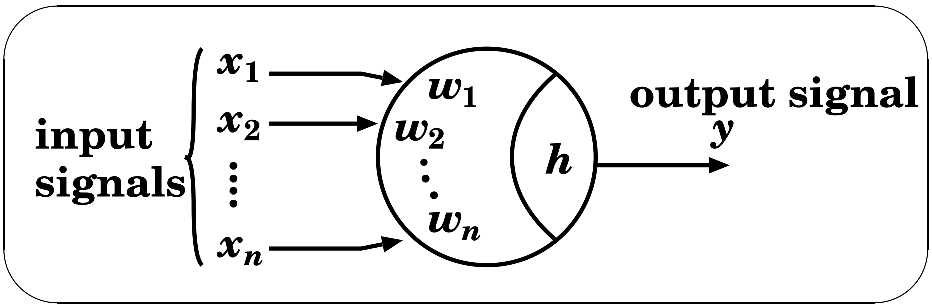
- If $t - y > 0$ and $x_j > 0$, then the above update contributes to
  an increase of w_j,
  so an increase of total input $\sum_{i=1}^{n} x_i w_i$,
  so a tendency to increase y,
  and so a tendency to reduce an error $t - y$.

- If $t - y < 0$ and $x_j > 0$, then the above update contributes to
  a decrease of w_j,
  so a decrease of total input $\sum_{i=1}^{n} x_i w_i$,
  so a tendency to decrease y,
  and so a tendency to reduce an error $t - y$.

⇨ **We obtain the following perceptron learning algorithm:**

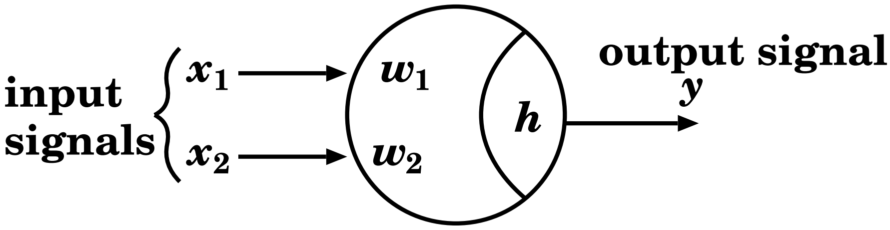
function PERCEPTRON-LEARNING(*examples*) **returns** *network*

network $\leftarrow$ a perceptron with randomly assigned weights w_i's and h;
repeat
num_of_disagree $\leftarrow$ 0;
$\Delta w_i \leftarrow 0$ for each i;
$\Delta h \leftarrow 0$;
for each (inputs:$x_1, \ldots, x_n$; required_output: t) **in** *examples* **do**
$y \leftarrow$ the output of the current perceptron on inputs $(x_1, \ldots, x_n)$;
if $y \neq t$ **then**
num_of_disagree $\leftarrow$ *num_of_disagree* +1;
$\Delta w_i \leftarrow \Delta w_i + \alpha x_i(t - y)$ for each i;
$\Delta h \leftarrow \Delta h - \alpha(t - y)$
end
end
$w_i \leftarrow w_i + \Delta w_i$ for each i;
$h \leftarrow h + \Delta h$
until *num_of_disagree*=0 or some stopping criterion is satisfied;
return *network*



5–5 Implementation in Fortran77

For a target perceptron structure



and the required input-output relations

inputs		output
x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	1

,

logical OR

we give a Fortran77 program that implements the perceptron learning algorithm described in the previous section 5–4:

```
xcspc60_41% cat training_perceptron_OR.f        display the contents of the file
****************************************************************************
* A Fortran77 program that implements the perceptron learning algorithm   *
*              to the problem of training a perceptron with one neuron-   *
*              like element so that it behaves as the logical OR function.*
****************************************************************************
      PROGRAM PERCEPTRON
      INTEGER NUM_OF_EXAMPLES, MAX_EPOCH
      REAL    ALPHA
      PARAMETER (NUM_OF_EXAMPLES=4, MAX_EPOCH=200, ALPHA=0.01)
      INTEGER EPOCH, EXAMPLE, NUM_OF_DISAGREE, Y,
     &        X1(1:NUM_OF_EXAMPLES), X2(1:NUM_OF_EXAMPLES),
     &        T(1:NUM_OF_EXAMPLES)
      REAL    W1, W2, H, DELTA_W1, DELTA_W2, DELTA_H
      DATA    X1/0,0,1,1/, X2/0,1,0,1/, T/0,1,1,1/

* Initialize a Pseudo-random Generator
```

```
      CALL RAND_INITIALIZE( )

* Generate a Perceptron Randomly (i.e. Initialize Weights and threshold)
      W1=RAND_REAL( )-0.5
      W2=RAND_REAL( )-0.5
      H=RAND_REAL( )-0.5

* Printout a Headline
      WRITE(*,100)

* Training the Perceptron
      DO 20 EPOCH=1,MAX_EPOCH
         WRITE(*,200) EPOCH-1, W1, W2, H
         NUM_OF_DISAGREE=0
         DELTA_W1=0.0
         DELTA_W2=0.0
         DELTA_H=0.0
         DO 10 EXAMPLE=1,NUM_OF_EXAMPLES
            IF (X1(EXAMPLE)*W1+X2(EXAMPLE)*W2 .GE. H) THEN
               Y=1
            ELSE
               Y=0
            END IF
            IF (Y .NE. T(EXAMPLE)) THEN
               NUM_OF_DISAGREE=NUM_OF_DISAGREE+1
               DELTA_W1=DELTA_W1+ALPHA*X1(EXAMPLE)*(T(EXAMPLE)-Y)
               DELTA_W2=DELTA_W2+ALPHA*X2(EXAMPLE)*(T(EXAMPLE)-Y)
               DELTA_H=DELTA_H-ALPHA*(T(EXAMPLE)-Y)
               WRITE(*,300) EXAMPLE, X1(EXAMPLE), X2(EXAMPLE),
     &                      T(EXAMPLE), Y,
     &                      ALPHA*X1(EXAMPLE)*(T(EXAMPLE)-Y),
     &                      ALPHA*X2(EXAMPLE)*(T(EXAMPLE)-Y),
     &                      -ALPHA*(T(EXAMPLE)-Y)
            ELSE
               WRITE(*,400) EXAMPLE, X1(EXAMPLE), X2(EXAMPLE),
     &                      T(EXAMPLE), Y
            END IF
   10    CONTINUE
         IF (NUM_OF_DISAGREE .EQ. 0) THEN
            WRITE(*,500)
            STOP
         END IF
         W1=W1+DELTA_W1
         W2=W2+DELTA_W2
         H=H+DELTA_H
   20 CONTINUE
      WRITE(*,600)MAX_EPOCH
```

```
 100 FORMAT(' -------------------------------------------------'/
    &       ' We now search a perceptron with one neuron-like'/
    &       ' element that behaves as the logical OR function,'/
    &       ' by the perceptron learning algorithm.'/
    &       ' -------------------------------------------------'//
    &       ' Epoch    W1      W2       H'/
    &       ' -----  ------  ------  ------')
 200 FORMAT(1X, I4, 2X, F6.3, 2X, F6.3, 2X, F6.3)
 300 FORMAT(4X, '| Example',I1,':X1=',I1,' X2=',I1,' T=',I1,' Output:',
    &       I1,' ==> ', 'W1+=',F6.3, ' W2+=',F6.3, ' H+=',F6.3)
 400 FORMAT(4X, '| Example',I1,':X1=',I1,' X2=',I1,' T=',I1,' Output:',
    &       I1,' ==> It behaves correctly.')
 500 FORMAT(/' *** We found a required perceptron. ***')
 600 FORMAT(/' *** We trained a perceptron ', I3, ' epochs,      ***'/
    &        ' *** but couldn''t find a required perceptron. ***')
     END

****************************************************************************
* A module for generating pseudo-random numbers                            *
****************************************************************************
```

We omit to display the code, since it is already given in the section 3–6.

5–6 Experimental Results

The program in the previous section 5–5 reports how the weights w_i and the threshold h are updated so that the hypothesis perceptron will become consistent with the given 4 examples.

Executing that program, we obtain the following results:

```
xcspc60_42% g77 training_perceptron_OR.f
                 (compile the source code and generate an executable code "a.out")
xcspc60_43% a.out                                 (invoke the executable code "a.out")
 Type in a random seed (positive integer).
1                                             (input to the executing program "a.out")
 -------------------------------------------------
 We now search a perceptron with one neuron-like
 element that behaves as the logical OR function,
 by the perceptron learning algorithm.
 -------------------------------------------------


 Epoch    W1      W2       H
 -----  ------  ------  ------
    0   0.014  -0.060   0.136
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    1   0.034  -0.040   0.106
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    2   0.054  -0.020   0.076
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    3   0.074   0.000   0.046
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
    4   0.074   0.010   0.036
```

```
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
    5   0.074   0.020   0.026
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
    6   0.074   0.030   0.016
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.

 *** We found a required perceptron. ***
xcspc60_44% a.out                          (invoke the executable code "a.out")
 Type in a random seed (positive integer).
3                                      (input to the executing program "a.out")
 --------------------------------------------------
 We now search a perceptron with one neuron-like
 element that behaves as the logical OR function,
 by the perceptron learning algorithm.
 --------------------------------------------------

 Epoch    W1      W2       H
 -----  ------  ------  ------
    0   0.042  -0.491   0.176
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    1   0.062  -0.471   0.146
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    2   0.082  -0.451   0.116
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:0 ==> W1+= 0.010 W2+= 0.000 H+=-0.010
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
    3   0.102  -0.431   0.086
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
```

```
 4   0.112  -0.411   0.066
 | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
 5   0.122  -0.391   0.046
 | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
 6   0.132  -0.371   0.026
 | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
 7   0.142  -0.351   0.006
 | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
 8   0.152  -0.331  -0.014
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
 9   0.162  -0.311  -0.024
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
10   0.172  -0.291  -0.034
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
11   0.182  -0.271  -0.044
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
12   0.192  -0.251  -0.054
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:0 ==> W1+= 0.010 W2+= 0.010 H+=-0.010
13   0.202  -0.231  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
```

```
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
14   0.202  -0.221  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
15   0.202  -0.211  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
16   0.202  -0.201  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
17   0.202  -0.191  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
18   0.202  -0.181  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
19   0.202  -0.171  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
20   0.202  -0.161  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
21   0.202  -0.151  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
22   0.202  -0.141  -0.064
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
```

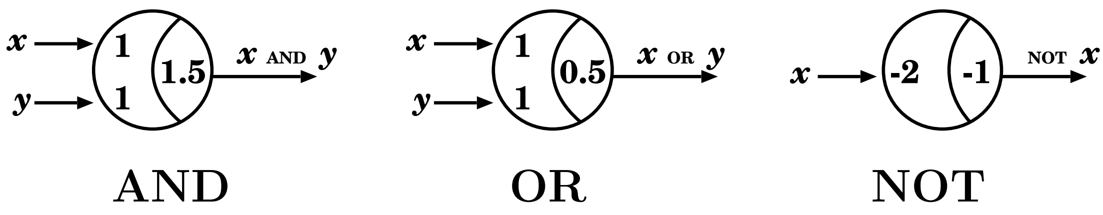
```
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
23   0.202  -0.131  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
24   0.202  -0.121  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
25   0.202  -0.111  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
26   0.202  -0.101  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
27   0.202  -0.091  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
28   0.202  -0.081  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
29   0.202  -0.071  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
30   0.202  -0.061  -0.064
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
31   0.202  -0.061  -0.054
  | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
  | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
  | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
  | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
32   0.202  -0.051  -0.054
```

```
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
33   0.202  -0.051  -0.044
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
34   0.202  -0.041  -0.044
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
35   0.202  -0.041  -0.034
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
36   0.202  -0.031  -0.034
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
37   0.202  -0.031  -0.024
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
38   0.202  -0.021  -0.024
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
39   0.202  -0.021  -0.014
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
40   0.202  -0.011  -0.014
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
 | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
 | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
41   0.202  -0.011  -0.004
 | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
 | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
```

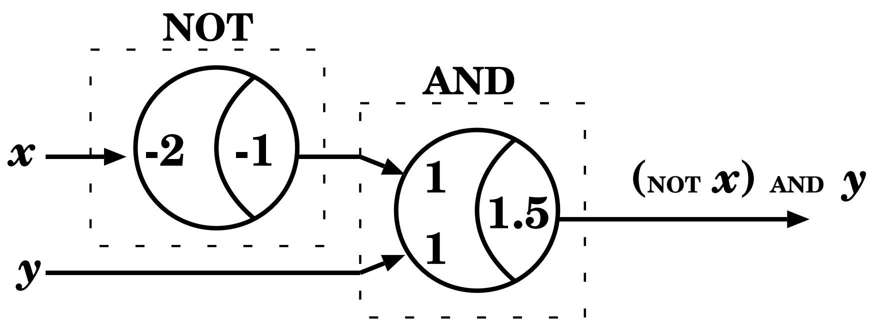
```
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
   42   0.202  -0.001  -0.004
    | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
    | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
   43   0.202  -0.001   0.006
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:0 ==> W1+= 0.000 W2+= 0.010 H+=-0.010
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
   44   0.202   0.009  -0.004
    | Example1:X1=0 X2=0 T=0 Output:1 ==> W1+= 0.000 W2+= 0.000 H+= 0.010
    | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.
   45   0.202   0.009   0.006
    | Example1:X1=0 X2=0 T=0 Output:0 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:1 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:1 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=1 Output:1 ==> It behaves correctly.

 *** We found a required perceptron. ***
xcspc60_45%
```

5–7 What kind of Boolean function can perceptron represent?

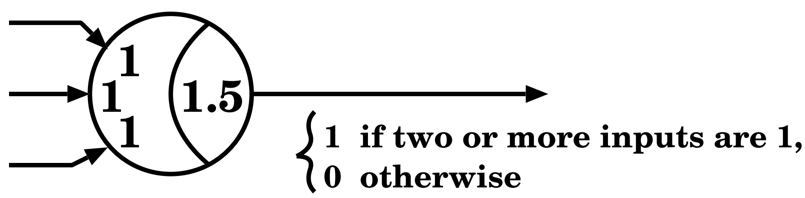
- Perceptrons can represent the simple Boolean functions AND, OR and NOT.



⇨ Every Boolean function can be represented by layering neuron-like units suitably.



- Perceptrons can represent some complex Boolean functions, e.g. the majority function which outputs 1 if and only if more than half of its inputs are 1.



But,

- **Perceptrons cannot represent some simple Boolean functions, e.g. ExclusiveOR.**

inputs x_1	x_2	ExclusiveOR
0	0	0
0	1	1
1	0	1
1	1	0

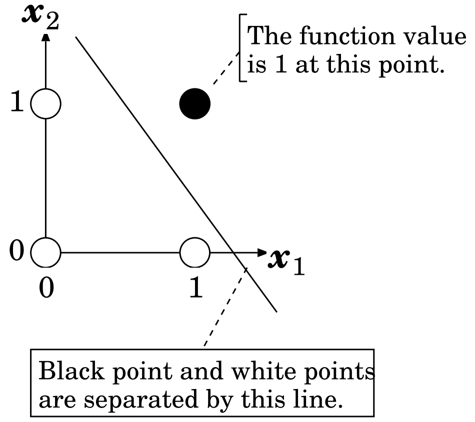
⇨ **Generally,**
perceptrons can represent a Boolean function if and only if that function is "linearly separable."

An n-ary Boolean function f is said to be **linearly separable** if two sets of input vectors

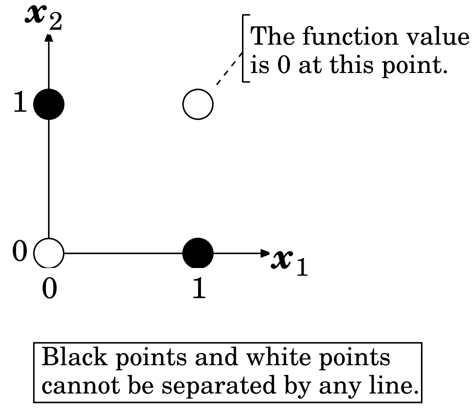
$$\{\vec{x} \mid f(\vec{x}) = 0\} \quad \text{and} \quad \{\vec{x} \mid f(\vec{x}) = 1\}$$

can be separated by some plane.

The AND function is linearly separable.



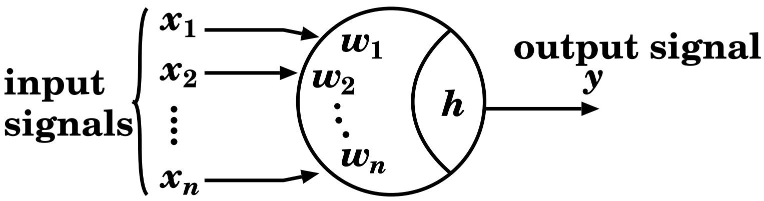
The ExclusiveOR function is not linearly separable.



6 Training Multilayer Neural Networks — back-propagation —

6–1 An Alternative Neuron-like Element

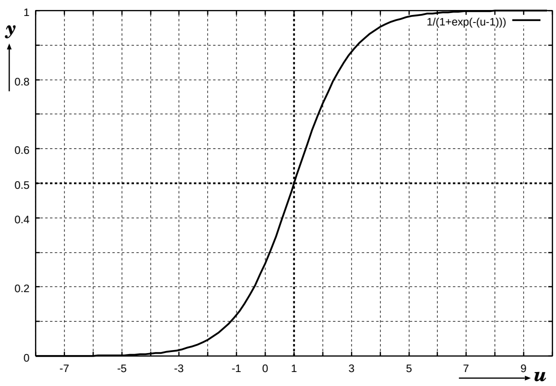
To avoid the difficulty of finding a good way to update the weights of "multilayer" neural networks, we next consider the other kind of simple computing elements that simulate a neuron:



$$u = \sum_{i=1}^{n} x_i w_i$$

$$y = \frac{1}{1+e^{-(u-h)}}$$

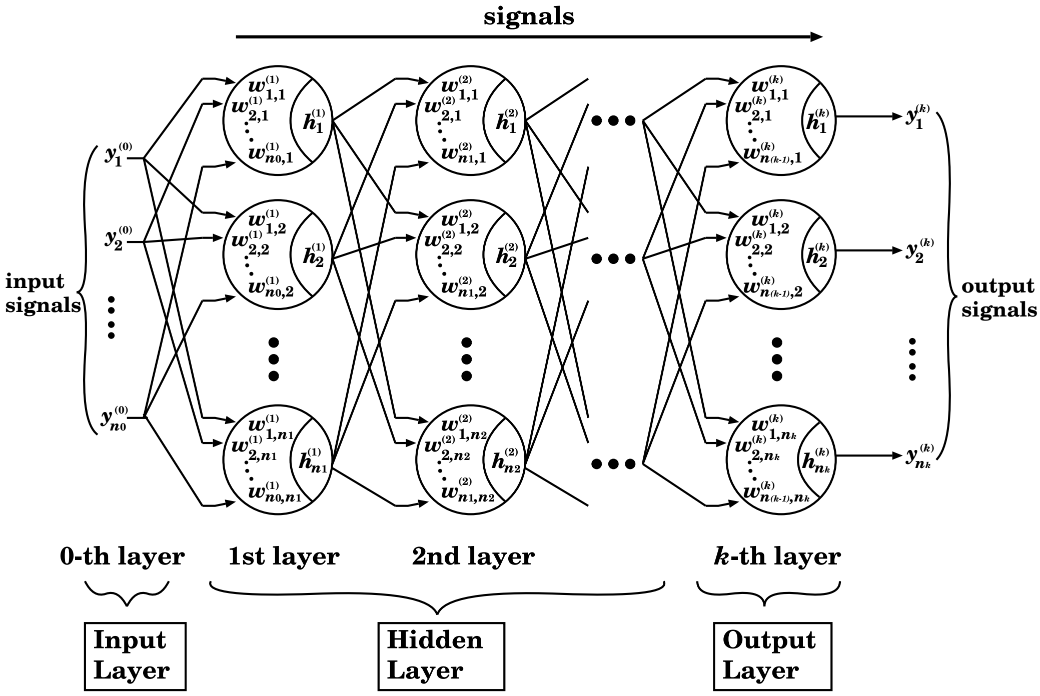
An "activation" function $y = \frac{1}{1+\exp\{-(u-1)\}}$**:**



- **The output** y **varies in the open interval (0,1).**
- **The derivative** $\frac{dy}{du}$ **exists.**

6–2 Multilayer Feed-Forward Networks

We consider a class of networks, called multilayer neural networks, that have the following network structure:



For each pair $m \in \{1, 2, ..., k\}$ **and** $j \in \{1, 2, ..., n_m\}$**, We define**

$u_j^{(m)}$ = **the total input (i.e. electric potential) of the** j**-th unit in the** m**-th layer, and**

$y_j^{(m)}$ = **the output of the** j**-th unit in the** m**-th layer.**

⇨ **From the assumption of the neuron-like elements in the previous subsection, we obtain**

$$u_j^{(m)} = \sum_{i=1}^{n_m} y_i^{(m-1)} w_{i,j}^{(m)} \quad \text{for any } m \geq 1 \text{ and } j, \quad \text{and}$$

$$y_j^{(m)} = g(u_j^{(m)} - h_j^{(m)}) \quad \text{for any } m \geq 1 \text{ and } j,$$

where $g(x) = \dfrac{1}{1 + e^{-x}}$.

6–3 Backpropagation Learning

We also follow the scheme of keeping and repetitively updating the current-best-hypothesis:

How to update the weights and threshold :

For multilayer networks, it is usually adopted to update the weights and thresholds with the following rule:

$$w_{i,j}^{(m)} \leftarrow w_{i,j}^{(m)} + \alpha \times y_i^{(m-1)} \times \mathcal{E}rr_j^{(m)}$$
$$h_j^{(m)} \leftarrow h_j^{(m)} - \alpha \times \mathcal{E}rr_j^{(m)}$$

for every $m \in \{1, 2, ..., k\}, i \in \{1, 2, ..., n_{m-1}\}$,
and $j \in \{1, 2, ..., n_m\}$,

where

α **is a small positive constant (the learning rate),**

$\mathcal{E}rr_j^{(m)}$ **can be regarded as a quantity of error that the** j**-th unit in the** m**-th layer committed.**

The quantities $\mathcal{E}rr_i^{(m)}$ **are calculated** **backward** **(i.e. from the** k**-th layer to the 1st layer) as follows:**

$$\mathcal{E}rr_i^{(k)} = t_i - y_i^{(k)} \qquad \text{for every } i \in \{1, 2, ..., n_k\}$$
$$\mathcal{E}rr_i^{(m)} = g'(u_i^{(m)} - h_i^{(m)}) \sum_{j=1}^{n_{m+1}} w_{i,j}^{(m+1)} \mathcal{E}rr_j^{(m+1)}$$

for every $m \in \{1, 2, ..., k-1\},\ i \in \{1, 2, ..., n_m\}$,

where

t_i **denotes the required output of the** i**-th unit (in the output layer) for the inputs** $(y_1^{(0)}, y_2^{(0)}, ..., y_{n_0}^{(0)})$,

g' **denotes the derivative of the (activation) function** $g(x) = \frac{1}{1+e^{-x}}$, **and so**

$$g'(u_i^{(m)} - h_i^{(m)}) = g(u_i^{(m)} - h_i^{(m)}) \left(1 - g(u_i^{(m)} - h_i^{(m)})\right)$$
$$= y_i^{(m)}(1 - y_i^{(m)})$$

We obtain the following backpropagation learning algorithm:

function BACKPROPAGATION-LEARNING(*examples*) **returns** *network*

network $\leftarrow$ a multilayer neural network
with randomly assigned weights $w_{i,j}^{(m)}$ and thresholds $h_j^{(m)}$;

repeat

num_of_disagree $\leftarrow 0$;

$\Delta w_{i,j}^{(m)} \leftarrow 0$ for each i, j, m;

$\Delta h_j^{(m)} \leftarrow 0$ for each j, m;

for each (inputs:$x_1, \ldots, x_{n_0}$;
required_output: $t_1, \ldots, t_{n_k}$) **in** *examples* **do**

$y_j^{(0)} \leftarrow x_j$ for each j;

for each $m = 1, 2, 3, ..., k$ **do**

$y_j^{(m)} \leftarrow$ the output of the j-th unit in the m-th layer for each j;

end

if ($|t_1 - y_1^k| > \epsilon$ or $|t_2 - y_2^k| > \epsilon$ or $\cdots$ or $|t_{n_k} - y_{n_k}^k| > \epsilon$) **then**

num_of_disagree $\leftarrow$ *num_of_disagree* +1;

$\mathcal{E}rr_i^{(k)} \leftarrow t_i - y_i^{(k)}$ for each i;

for each $m = k-1, k-2, ..., 2, 1$ **do**

$\mathcal{E}rr_i^{(m)} \leftarrow y_i^{(m)}(1 - y_i^{(m)}) \sum_{j=1}^{n_{m+1}} w_{i,j}^{(m+1)} \mathcal{E}rr_j^{(m+1)}$ for each i;

end

$\Delta w_{i,j}^{(m)} \leftarrow \Delta w_{i,j}^{(m)} + \alpha y_i^{(m-1)} \mathcal{E}rr_j^{(m)}$ for each m, i, j;

$\Delta h_j^{(m)} \leftarrow \Delta h_j^{(m)} - \alpha \mathcal{E}rr_j^{(m)}$ for each m, j

end

end

$w_{i,j}^{(m)} \leftarrow w_{i,j}^{(m)} + \Delta w_{i,j}^{(m)}$ for each m, i, j;

$h_j^{(m)} \leftarrow h_j^{(m)} + \Delta h_j^{(m)}$ for each m, j

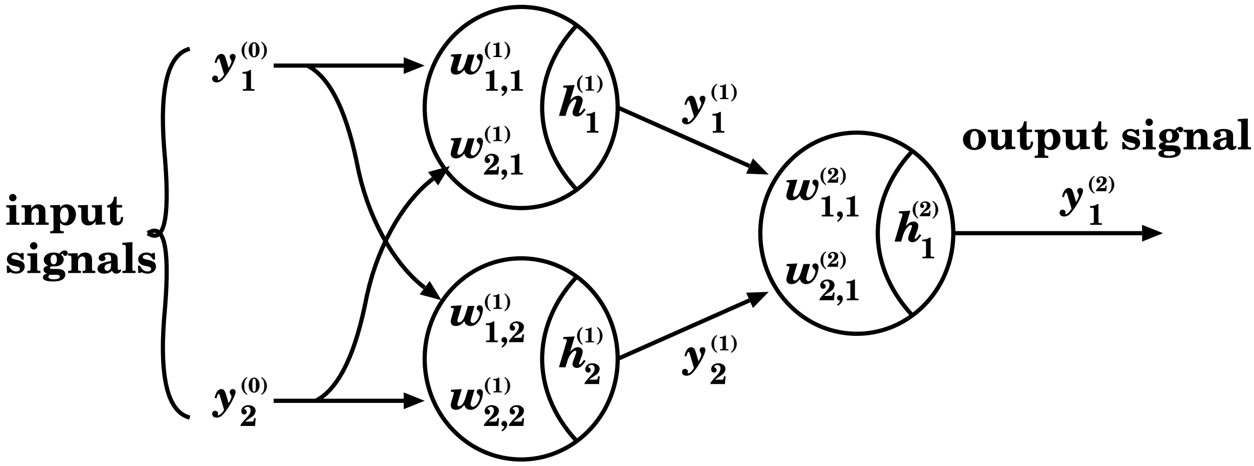
until *num_of_disagree*=0 or some stopping criterion is satisfied;

return *network*

This learning algorithm is not guaranteed to converge to a global optimum.

6–4 Implementation in Fortran77

For a target multilayer network structure



and the required input-output relations

inputs x_1	x_2	output t
0	0	0
0	1	1
1	0	1
1	1	0

,

logical ExclusiveOR

we give a Fortran77 program that implements the back-propagation learning algorithm described in the previous section 6–3:

```
xcspc60_41% cat training_multilayer_net_XOR.f        display the Fortran77 source code
***************************************************************************
* A Fortran77 program that implements the backpropagation learning       *
*              algorithm to the problem of training a multilayer network  *
*              with two inputs, two 1st-layer units and one 2nd-layer unit*
*              so that it behaves as the logical ExclusiveOR function.    *
***************************************************************************
      PROGRAM BACKPROPAGATION
      INTEGER MAX_LAYER, MAX_UNIT_NUM, NUM_OF_EXAMPLES, MAX_EPOCH,
     &        LAYER0, LAYER1, LAYER2, IN1, IN2, UNIT1, UNIT2
      REAL    ALPHA, EPSILON
      PARAMETER (MAX_LAYER=2, MAX_UNIT_NUM=2, NUM_OF_EXAMPLES=4,
     &           MAX_EPOCH=10000,
     &           LAYER0=0, LAYER1=1, LAYER2=2,
     &           IN1=1, IN2=2, UNIT1=1, UNIT2=2,
```

```
     &           ALPHA=0.9, EPSILON=0.05)
      INTEGER NUM_OF_UNITS(0:MAX_LAYER),
     &        LAYER, UNIT, EPOCH, EXAMPLE, NUM_OF_DISAGREE,
     &        X1(1:NUM_OF_EXAMPLES), X2(1:NUM_OF_EXAMPLES),
     &        T(1:NUM_OF_EXAMPLES)
      DATA    NUM_OF_UNITS/2,2,1/,
     &        X1/0,0,1,1/, X2/0,1,0,1/, T/0,1,1,0/
      REAL    Y(0:MAX_LAYER, 1:MAX_UNIT_NUM),
     &        W(0:MAX_LAYER, 1:MAX_UNIT_NUM, 1:MAX_UNIT_NUM),
     &        H(0:MAX_LAYER, 1:MAX_UNIT_NUM),
     &        ERR(1:MAX_LAYER, 1:MAX_UNIT_NUM),
     &        DELTA_W(0:MAX_LAYER, 1:MAX_UNIT_NUM, 1:MAX_UNIT_NUM),
     &        DELTA_H(0:MAX_LAYER, 1:MAX_UNIT_NUM)

* Initialize a Pseudo-random Generator
      CALL RAND_INITIALIZE( )

* Generate a Multilayer Network Randomly (i.e. Initialize Weights and threshold)
      DO 20 LAYER=1,2
         DO 10 UNIT=1,NUM_OF_UNITS(LAYER)
            W(LAYER,IN1,UNIT)=RAND_REAL( )-0.5
            W(LAYER,IN2,UNIT)=RAND_REAL( )-0.5
            H(LAYER,UNIT)=RAND_REAL( )-0.5
   10    CONTINUE
   20 CONTINUE

* Printout a Headline
      WRITE(*,1000)

* Training the Multilayer Network
      DO 100 EPOCH=1,MAX_EPOCH
         WRITE(*,2000) EPOCH-1,
     &      W(LAYER1,IN1,UNIT1),W(LAYER1,IN2,UNIT1),H(LAYER1,UNIT1),
     &      W(LAYER1,IN1,UNIT2),W(LAYER1,IN2,UNIT2),H(LAYER1,UNIT2),
     &      W(LAYER2,IN1,UNIT1),W(LAYER2,IN2,UNIT1),H(LAYER2,UNIT1)
         NUM_OF_DISAGREE=0
         DO 40 LAYER=1,2
            DO 30 UNIT=1,NUM_OF_UNITS(LAYER)
               DELTA_W(LAYER,IN1,UNIT)=0.0
               DELTA_W(LAYER,IN2,UNIT)=0.0
               DELTA_H(LAYER,UNIT)=0.0
   30       CONTINUE
   40    CONTINUE

         DO 70 EXAMPLE=1,NUM_OF_EXAMPLES
*           ---Calculate the Output of each Unit in each Layer---
            Y(LAYER0,UNIT1)=REAL(X1(EXAMPLE))
            Y(LAYER0,UNIT2)=REAL(X2(EXAMPLE))
            Y(LAYER1,UNIT1)=
     &           1.0/(1.0+exp(-(Y(LAYER0,UNIT1)*W(LAYER1,IN1,UNIT1)
     &                         +Y(LAYER0,UNIT2)*W(LAYER1,IN2,UNIT1)
     &                         -H(LAYER1,UNIT1))))
            Y(LAYER1,UNIT2)=
     &           1.0/(1.0+exp(-(Y(LAYER0,UNIT1)*W(LAYER1,IN1,UNIT2)
     &                         +Y(LAYER0,UNIT2)*W(LAYER1,IN2,UNIT2)
     &                         -H(LAYER1,UNIT2))))
```

```
            Y(LAYER2,UNIT1)=
     &           1.0/(1.0+exp(-(Y(LAYER1,UNIT1)*W(LAYER2,IN1,UNIT1)
     &                         +Y(LAYER1,UNIT2)*W(LAYER2,IN2,UNIT1)
     &                         -H(LAYER2,UNIT1))))
*           ---Check wether the Output is Correct---
            IF (ABS(REAL(T(EXAMPLE))-Y(LAYER2,UNIT1)) .GT. EPSILON) THEN
               NUM_OF_DISAGREE=NUM_OF_DISAGREE+1
*              ---Estimate the Error of each Unit in each Layer Backward---
               ERR(LAYER2,UNIT1)=REAL(T(EXAMPLE))-Y(LAYER2,UNIT1)
               ERR(LAYER1,UNIT1)=Y(LAYER1,UNIT1)*(1-Y(LAYER1,UNIT1))
     &                          *W(LAYER2,IN1,UNIT1)*ERR(LAYER2,UNIT1)
               ERR(LAYER1,UNIT2)=Y(LAYER1,UNIT2)*(1-Y(LAYER1,UNIT2))
     &                          *W(LAYER2,IN2,UNIT1)*ERR(LAYER2,UNIT1)
*              ---Calculate the Adjustment on Weights and Thresholds---
               DO 60 LAYER=1,2
                  DO 50 UNIT=1,NUM_OF_UNITS(LAYER)
                     DELTA_W(LAYER,IN1,UNIT)=DELTA_W(LAYER,IN1,UNIT)
     &                  + ALPHA*Y(LAYER-1,IN1)*ERR(LAYER,UNIT)
                     DELTA_W(LAYER,IN2,UNIT)=DELTA_W(LAYER,IN2,UNIT)
     &                  + ALPHA*Y(LAYER-1,IN2)*ERR(LAYER,UNIT)
                     DELTA_H(LAYER,UNIT)=DELTA_H(LAYER,UNIT)
     &                  - ALPHA*ERR(LAYER,UNIT)
   50             CONTINUE
   60          CONTINUE
               WRITE(*,3000)
     &            EXAMPLE, X1(EXAMPLE), X2(EXAMPLE), T(EXAMPLE),
     &            Y(LAYER1,UNIT1), Y(LAYER1,UNIT2), Y(LAYER2,UNIT1),
     &            ERR(LAYER2,UNIT1),ERR(LAYER1,UNIT1),ERR(LAYER1,UNIT2)
            ELSE
               WRITE(*,4000)
     &            EXAMPLE, X1(EXAMPLE), X2(EXAMPLE), T(EXAMPLE),
     &            Y(LAYER1,UNIT1), Y(LAYER1,UNIT2), Y(LAYER2,UNIT1)
            END IF
   70    CONTINUE

*        ---Check whether We Found a Required Multilayer Network---
         IF (NUM_OF_DISAGREE .EQ. 0) THEN
            WRITE(*,5000)
            STOP
         END IF
*        ---Update Weights and Thresholds---
         DO 90 LAYER=1,2
            DO 80 UNIT=1,NUM_OF_UNITS(LAYER)
               W(LAYER,IN1,UNIT)=
     &            W(LAYER,IN1,UNIT)+DELTA_W(LAYER,IN1,UNIT)
               W(LAYER,IN2,UNIT)=
     &            W(LAYER,IN2,UNIT)+DELTA_W(LAYER,IN2,UNIT)
               H(LAYER,UNIT)=H(LAYER,UNIT)+DELTA_H(LAYER,UNIT)
   80       CONTINUE
   90    CONTINUE
  100 CONTINUE

      WRITE(*,6000)MAX_EPOCH

 1000 FORMAT(' -----------------------------------------------------'/
     &       ' We now search a multilayer network with two inputs,'/
```

```
     &       ' two 1st-layer units and one 2nd-layer unit that'/
     &       ' behaves as the logical ExclusiveOR function,'/
     &       ' by the backpropagation learning algorithm.'/
     &       ' ---------------------------------------------------'//
     &       '       ---(Layer1,Unit1)---  ---(Layer1,Unit2)---',
     &                                  '  ---(Layer2,Unit1)---'/
     &       ' Epoch    W1     W2      H      W1     W2      H  ',
     &                                  '    W1     W2      H  '/
     &       ' -----  ------ ------ ------  ------ ------ ------',
     &                                  '  ------ ------ ------')
 2000 FORMAT(1X, I4, 1X, 3(2X, F6.3, 1X, F6.3, 1X, F6.3))
 3000 FORMAT(4X, '| Example',I1,':X1=',I1,' X2=',I1,' T=',I1,
     &       ' Output:Y1_1=',F5.2,' Y1_2=',F5.2,' ->Y2_1=',F5.2,
     &       ' ==> Err2_1=',F5.2,' ->Err1_1=',F5.2,' Err1_2=',F5.2)
 4000 FORMAT(4X, '| Example',I1,':X1=',I1,' X2=',I1,' T=',I1,
     &       ' Output:Y1_1=',F5.2,' Y1_2=',F5.2,' ->Y2_1=',F5.2,
     &       ' ==> It behaves correctly.')
 5000 FORMAT(/' *** We found a required multilayer network. ***')
 6000 FORMAT(/
     & ' *** We trained a multilayer network ', I5, ' epochs,    ***'/
     & ' *** but couldn''t find a required multilayer network. ***')
      END

***************************************************************************
* A module for generating pseudo-random numbers                           *
***************************************************************************
```

We omit to display the code, since it is already given in the section 3–6.

6–5 Experimental Results

The program in the previous section 6–4 reports how the weights $w_{i,j}^{(m)}$ and the thresholds $h_j^{(m)}$ are updated so that the hypothesis multilayer network will become consistent with the given 4 examples. Executing that program, we obtain the following results:

```
xcspc60_42% g77 training_multilayer_net_XOR.f        (compile the source code and generate an executable code "a.out")
xcspc60_43% a.out                                    (invoke the executable code "a.out")
 Type in a random seed (positive integer).
55                                                   (input to the executing program "a.out")
 --------------------------------------------------
 We now search a multilayer network with two inputs,
 two 1st-layer units and one 2nd-layer unit that
 behaves as the logical ExclusiveOR function,
 by the backpropagation learning algorithm.
 --------------------------------------------------

        ---(Layer1,Unit1)---  ---(Layer1,Unit2)---  ---(Layer2,Unit1)---
 Epoch    W1     W2      H      W1     W2      H      W1     W2      H
 -----  ------ ------ ------  ------ ------ ------  ------ ------ ------
    0   -0.237  0.311  0.037  -0.371 -0.479  0.348  -0.432 -0.203 -0.096
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.05 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.30 ->Y2_1= 0.45 ==> Err2_1= 0.55 ->Err1_1=-0.06 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.33 ->Y2_1= 0.46 ==> Err2_1= 0.54 ->Err1_1=-0.06 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.23 ->Y2_1= 0.46 ==> Err2_1=-0.46 ->Err1_1= 0.05 Err1_2= 0.02
    1   -0.244  0.302  0.053  -0.378 -0.486  0.356  -0.348 -0.155 -0.262
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.50 Y1_2= 0.23 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.01
    2   -0.241  0.304  0.048  -0.380 -0.488  0.355  -0.366 -0.173 -0.225
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.05 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.05 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.50 Y1_2= 0.23 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.05 Err1_2= 0.02
    3   -0.240  0.304  0.048  -0.383 -0.491  0.357  -0.360 -0.176 -0.235
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.50 Y1_2= 0.23 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.05 Err1_2= 0.02
    4   -0.239  0.305  0.046  -0.386 -0.494  0.358  -0.360 -0.183 -0.234
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.50 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.05 Err1_2= 0.02
    5   -0.237  0.305  0.045  -0.390 -0.497  0.359  -0.358 -0.188 -0.236
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.05 Err1_2= 0.02
    6   -0.236  0.306  0.044  -0.393 -0.500  0.360  -0.357 -0.194 -0.237
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    7   -0.234  0.306  0.042  -0.397 -0.503  0.361  -0.356 -0.200 -0.238
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    8   -0.233  0.307  0.041  -0.401 -0.506  0.362  -0.354 -0.207 -0.240
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.30 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
```

```
9   -0.231  0.307  0.040  -0.404 -0.510  0.363  -0.353 -0.213 -0.241
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
10   -0.230  0.308  0.038  -0.409 -0.514  0.364  -0.352 -0.219 -0.242
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.32 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
11   -0.229  0.308  0.037  -0.413 -0.517  0.366  -0.351 -0.226 -0.244
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.31 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
12   -0.227  0.309  0.036  -0.417 -0.521  0.367  -0.350 -0.232 -0.245
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.31 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.02
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
13   -0.226  0.310  0.035  -0.422 -0.526  0.369  -0.348 -0.239 -0.246
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.02
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.31 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
14   -0.225  0.310  0.033  -0.427 -0.530  0.370  -0.347 -0.246 -0.248
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.31 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
15   -0.223  0.311  0.032  -0.432 -0.534  0.371  -0.346 -0.253 -0.249
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.31 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
16   -0.222  0.312  0.031  -0.437 -0.539  0.373  -0.345 -0.260 -0.251
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.31 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.51 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.04 Err1_2= 0.02
17   -0.221  0.312  0.030  -0.442 -0.544  0.375  -0.344 -0.268 -0.252
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.29 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.31 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
18   -0.219  0.313  0.029  -0.448 -0.549  0.376  -0.342 -0.275 -0.254
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.49 ==> Err2_1= 0.51 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
19   -0.218  0.314  0.027  -0.453 -0.554  0.378  -0.341 -0.283 -0.255
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
20   -0.217  0.315  0.026  -0.460 -0.560  0.380  -0.340 -0.291 -0.257
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.03
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
21   -0.215  0.315  0.025  -0.466 -0.565  0.382  -0.339 -0.299 -0.258
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
22   -0.214  0.316  0.024  -0.472 -0.571  0.384  -0.338 -0.307 -0.260
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.41 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
23   -0.213  0.317  0.023  -0.479 -0.577  0.386  -0.337 -0.316 -0.261
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.30 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.02
24   -0.211  0.318  0.022  -0.486 -0.584  0.388  -0.336 -0.324 -0.263
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.29 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
25   -0.210  0.319  0.021  -0.494 -0.591  0.390  -0.335 -0.333 -0.265
```

```
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.29 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
26   -0.209  0.320  0.019  -0.501 -0.597  0.392  -0.334 -0.342 -0.266
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.57 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.29 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
27   -0.207  0.321  0.018  -0.509 -0.605  0.395  -0.333 -0.352 -0.268
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.03
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.29 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
28   -0.206  0.322  0.017  -0.518 -0.612  0.397  -0.331 -0.362 -0.270
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.29 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.52 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
29   -0.205  0.323  0.016  -0.526 -0.620  0.400  -0.330 -0.372 -0.271
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
30   -0.204  0.324  0.015  -0.535 -0.629  0.402  -0.329 -0.382 -0.273
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.04
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.17 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
31   -0.202  0.325  0.014  -0.545 -0.637  0.405  -0.328 -0.393 -0.275
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.17 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
32   -0.201  0.326  0.013  -0.555 -0.646  0.408  -0.327 -0.404 -0.277
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.28 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.17 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
33   -0.200  0.327  0.012  -0.565 -0.656  0.410  -0.326 -0.415 -0.279
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.16 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
34   -0.198  0.328  0.011  -0.576 -0.665  0.413  -0.325 -0.427 -0.281
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.25 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.16 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
35   -0.197  0.330  0.010  -0.587 -0.676  0.416  -0.324 -0.440 -0.283
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.25 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.16 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
36   -0.196  0.331  0.009  -0.599 -0.686  0.419  -0.323 -0.452 -0.284
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.25 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.27 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.15 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
37   -0.194  0.332  0.008  -0.611 -0.698  0.423  -0.322 -0.465 -0.287
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.04 Err1_2= 0.05
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.25 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.15 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
38   -0.193  0.333  0.007  -0.624 -0.709  0.426  -0.321 -0.479 -0.289
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.24 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.04
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.15 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
39   -0.191  0.335  0.006  -0.637 -0.722  0.429  -0.320 -0.493 -0.291
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.24 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.26 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.53 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
40   -0.190  0.336  0.005  -0.651 -0.735  0.433  -0.320 -0.508 -0.293
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.24 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.25 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
41   -0.189  0.338  0.004  -0.666 -0.748  0.436  -0.319 -0.524 -0.295
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
```

```
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.23 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.25 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
42   -0.187  0.339  0.003  -0.681 -0.762  0.440  -0.318 -0.540 -0.297
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.23 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.25 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.13 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
43   -0.186  0.341  0.002  -0.697 -0.777  0.443  -0.317 -0.557 -0.299
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.06
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.23 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.24 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.13 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.04 Err1_2= 0.03
44   -0.184  0.342  0.001  -0.714 -0.793  0.447  -0.316 -0.574 -0.302
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.07
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.24 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
45   -0.183  0.344  0.000  -0.731 -0.809  0.451  -0.315 -0.592 -0.304
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.07
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.23 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
46   -0.181  0.346 -0.001  -0.749 -0.826  0.455  -0.314 -0.611 -0.307
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.07
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.22 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.23 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
47   -0.180  0.347 -0.001  -0.768 -0.844  0.458  -0.314 -0.631 -0.309
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.07
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.23 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.11 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
48   -0.178  0.349 -0.002  -0.788 -0.863  0.462  -0.313 -0.652 -0.312
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.04 Err1_2= 0.07
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.22 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.11 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
49   -0.176  0.351 -0.003  -0.809 -0.882  0.466  -0.312 -0.674 -0.314
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.08
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.21 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.22 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.54 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
50   -0.175  0.353 -0.004  -0.831 -0.903  0.469  -0.312 -0.697 -0.317
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.08
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.20 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.21 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
51   -0.173  0.355 -0.005  -0.854 -0.925  0.473  -0.311 -0.721 -0.319
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.08
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.20 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.21 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
52   -0.171  0.357 -0.006  -0.878 -0.947  0.476  -0.310 -0.746 -0.322
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.08
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.19 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.21 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.09 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
53   -0.169  0.359 -0.006  -0.903 -0.971  0.479  -0.310 -0.772 -0.325
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.09
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.19 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.09 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
54   -0.167  0.362 -0.007  -0.929 -0.996  0.482  -0.309 -0.799 -0.328
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.09
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.19 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.20 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.08 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
55   -0.165  0.364 -0.008  -0.957 -1.022  0.485  -0.309 -0.828 -0.331
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.04 Err1_2= 0.09
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.18 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.08 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.04 Err1_2= 0.03
56   -0.163  0.366 -0.009  -0.985 -1.049  0.487  -0.308 -0.858 -0.334
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.46 ==> Err2_1=-0.46 ->Err1_1= 0.04 Err1_2= 0.09
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.18 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.19 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.07 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
57   -0.161  0.369 -0.010  -1.015 -1.077  0.488  -0.308 -0.890 -0.338
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.46 ==> Err2_1=-0.46 ->Err1_1= 0.04 Err1_2= 0.10
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.17 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
```

```
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.55 Y1_2= 0.07 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
58   -0.159  0.372 -0.010  -1.046 -1.107  0.490  -0.307 -0.923 -0.341
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.46 ==> Err2_1=-0.46 ->Err1_1= 0.04 Err1_2= 0.10
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.17 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.18 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.07 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
59   -0.157  0.374 -0.011  -1.078 -1.138  0.490  -0.307 -0.957 -0.345
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.46 ==> Err2_1=-0.46 ->Err1_1= 0.04 Err1_2= 0.10
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.16 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.17 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.06 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
60   -0.154  0.377 -0.012  -1.112 -1.170  0.490  -0.307 -0.993 -0.348
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.16 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.17 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.06 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
61   -0.152  0.380 -0.012  -1.147 -1.204  0.489  -0.306 -1.031 -0.352
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.16 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.16 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.06 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
62   -0.150  0.383 -0.013  -1.183 -1.239  0.487  -0.306 -1.071 -0.356
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.15 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.16 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.05 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
63   -0.147  0.386 -0.013  -1.221 -1.275  0.484  -0.306 -1.112 -0.361
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.15 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.15 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.05 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.04 Err1_2= 0.03
64   -0.144  0.390 -0.014  -1.260 -1.313  0.480  -0.306 -1.155 -0.365
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.44 ==> Err2_1=-0.44 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.14 ->Y2_1= 0.50 ==> Err2_1= 0.50 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.15 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.56 Y1_2= 0.05 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.03
65   -0.142  0.393 -0.014  -1.300 -1.352  0.475  -0.306 -1.200 -0.370
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.44 ==> Err2_1=-0.44 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.57 Y1_2= 0.04 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.03
66   -0.139  0.397 -0.015  -1.342 -1.392  0.468  -0.305 -1.246 -0.375
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.38 ->Y2_1= 0.44 ==> Err2_1=-0.44 ->Err1_1= 0.03 Err1_2= 0.13
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.13 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.14 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.57 Y1_2= 0.04 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.03
67   -0.136  0.400 -0.015  -1.385 -1.434  0.460  -0.305 -1.295 -0.381
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.43 ==> Err2_1=-0.43 ->Err1_1= 0.03 Err1_2= 0.13
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.13 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.14 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.57 Y1_2= 0.04 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.02
68   -0.132  0.404 -0.016  -1.430 -1.477  0.451  -0.306 -1.345 -0.386
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.43 ==> Err2_1=-0.43 ->Err1_1= 0.03 Err1_2= 0.14
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.13 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.13 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.07
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.57 Y1_2= 0.03 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.02
69   -0.129  0.408 -0.016  -1.476 -1.522  0.440  -0.306 -1.398 -0.392
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.42 ==> Err2_1=-0.42 ->Err1_1= 0.03 Err1_2= 0.14
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.12 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.13 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.57 Y1_2= 0.03 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.04 Err1_2= 0.02
70   -0.126  0.413 -0.016  -1.523 -1.568  0.427  -0.306 -1.452 -0.399
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.39 ->Y2_1= 0.42 ==> Err2_1=-0.42 ->Err1_1= 0.03 Err1_2= 0.15
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.12 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.07
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.58 Y1_2= 0.03 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.04 Err1_2= 0.02
71   -0.122  0.417 -0.017  -1.572 -1.616  0.413  -0.306 -1.508 -0.406
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.41 ==> Err2_1=-0.41 ->Err1_1= 0.03 Err1_2= 0.15
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.12 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.58 Y1_2= 0.03 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.04 Err1_2= 0.02
72   -0.118  0.422 -0.017  -1.622 -1.665  0.396  -0.306 -1.566 -0.413
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.40 ->Y2_1= 0.41 ==> Err2_1=-0.41 ->Err1_1= 0.03 Err1_2= 0.15
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.11 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.12 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.58 Y1_2= 0.02 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.04 Err1_2= 0.02
73   -0.114  0.427 -0.017  -1.673 -1.715  0.378  -0.307 -1.626 -0.421
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.41 ->Y2_1= 0.40 ==> Err2_1=-0.40 ->Err1_1= 0.03 Err1_2= 0.16
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.11 ->Y2_1= 0.51 ==> Err2_1= 0.49 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.11 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
```

```
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.58 Y1_2= 0.02 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.04 Err1_2= 0.02
74   -0.110  0.432 -0.017  -1.726 -1.767  0.358  -0.307 -1.687 -0.430
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.41 ->Y2_1= 0.40 ==> Err2_1=-0.40 ->Err1_1= 0.03 Err1_2= 0.16
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.11 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.11 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.58 Y1_2= 0.02 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.04 Err1_2= 0.02
75   -0.106  0.437 -0.017  -1.780 -1.820  0.336  -0.308 -1.751 -0.439
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.42 ->Y2_1= 0.39 ==> Err2_1=-0.39 ->Err1_1= 0.03 Err1_2= 0.17
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.11 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.59 Y1_2= 0.02 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.04 Err1_2= 0.02
76   -0.101  0.443 -0.017  -1.835 -1.874  0.313  -0.309 -1.816 -0.448
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.42 ->Y2_1= 0.38 ==> Err2_1=-0.38 ->Err1_1= 0.03 Err1_2= 0.17
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.10 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.59 Y1_2= 0.02 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.04 Err1_2= 0.02
77   -0.096  0.448 -0.016  -1.892 -1.929  0.288  -0.310 -1.882 -0.459
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.43 ->Y2_1= 0.38 ==> Err2_1=-0.38 ->Err1_1= 0.03 Err1_2= 0.17
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.10 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.59 Y1_2= 0.02 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.04 Err1_2= 0.02
78   -0.091  0.455 -0.016  -1.949 -1.986  0.261  -0.311 -1.950 -0.469
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.44 ->Y2_1= 0.37 ==> Err2_1=-0.37 ->Err1_1= 0.03 Err1_2= 0.18
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.10 ->Y2_1= 0.52 ==> Err2_1= 0.48 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.10 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.59 Y1_2= 0.01 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.04 Err1_2= 0.02
79   -0.086  0.461 -0.016  -2.008 -2.044  0.233  -0.312 -2.020 -0.481
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.44 ->Y2_1= 0.36 ==> Err2_1=-0.36 ->Err1_1= 0.03 Err1_2= 0.18
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.09 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.10 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.60 Y1_2= 0.01 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.04 Err1_2= 0.02
80   -0.080  0.468 -0.015  -2.068 -2.103  0.204  -0.314 -2.090 -0.493
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.45 ->Y2_1= 0.35 ==> Err2_1=-0.35 ->Err1_1= 0.03 Err1_2= 0.18
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.09 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.09 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.60 Y1_2= 0.01 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.04 Err1_2= 0.01
81   -0.074  0.475 -0.014  -2.129 -2.163  0.173  -0.316 -2.162 -0.505
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.46 ->Y2_1= 0.34 ==> Err2_1=-0.34 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.09 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.09 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.60 Y1_2= 0.01 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.04 Err1_2= 0.01
82   -0.068  0.482 -0.014  -2.190 -2.224  0.142  -0.319 -2.234 -0.519
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.46 ->Y2_1= 0.34 ==> Err2_1=-0.34 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.09 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.09 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.61 Y1_2= 0.01 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.04 Err1_2= 0.01
83   -0.062  0.490 -0.013  -2.253 -2.286  0.110  -0.322 -2.308 -0.532
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.47 ->Y2_1= 0.33 ==> Err2_1=-0.33 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.08 ->Y2_1= 0.53 ==> Err2_1= 0.47 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.09 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.61 Y1_2= 0.01 ->Y2_1= 0.58 ==> Err2_1=-0.58 ->Err1_1= 0.04 Err1_2= 0.01
84   -0.055  0.498 -0.012  -2.316 -2.349  0.078  -0.326 -2.382 -0.547
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.48 ->Y2_1= 0.32 ==> Err2_1=-0.32 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.08 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.08 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.61 Y1_2= 0.01 ->Y2_1= 0.58 ==> Err2_1=-0.58 ->Err1_1= 0.04 Err1_2= 0.01
85   -0.047  0.507 -0.010  -2.380 -2.412  0.045  -0.330 -2.456 -0.562
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.49 ->Y2_1= 0.31 ==> Err2_1=-0.31 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.08 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.08 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.62 Y1_2= 0.01 ->Y2_1= 0.58 ==> Err2_1=-0.58 ->Err1_1= 0.05 Err1_2= 0.01
86   -0.040  0.516 -0.009  -2.444 -2.476  0.013  -0.335 -2.530 -0.577
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.50 ->Y2_1= 0.30 ==> Err2_1=-0.30 ->Err1_1= 0.03 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.08 ->Y2_1= 0.54 ==> Err2_1= 0.46 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.08 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.62 Y1_2= 0.01 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.05 Err1_2= 0.01
87   -0.032  0.526 -0.007  -2.509 -2.540 -0.020  -0.341 -2.605 -0.593
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.50 ->Y2_1= 0.29 ==> Err2_1=-0.29 ->Err1_1= 0.02 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.07 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.08 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.62 Y1_2= 0.01 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.05 Err1_2= 0.01
88   -0.023  0.536 -0.006  -2.573 -2.605 -0.052  -0.348 -2.679 -0.609
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.51 ->Y2_1= 0.28 ==> Err2_1=-0.28 ->Err1_1= 0.02 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.07 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.07 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.63 Y1_2= 0.01 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.05 Err1_2= 0.01
89   -0.014  0.546 -0.004  -2.638 -2.669 -0.083  -0.356 -2.753 -0.626
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.52 ->Y2_1= 0.27 ==> Err2_1=-0.27 ->Err1_1= 0.02 Err1_2= 0.19
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.07 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.07 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.63 Y1_2= 0.01 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.05 Err1_2= 0.01
```

```
 90   -0.005  0.557 -0.002  -2.703 -2.734 -0.114  -0.364 -2.827 -0.643
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.53 ->Y2_1= 0.26 ==> Err2_1=-0.26 ->Err1_1= 0.02 Err1_2= 0.18
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.64 Y1_2= 0.07 ->Y2_1= 0.55 ==> Err2_1= 0.45 ->Err1_1=-0.04 Err1_2=-0.08
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.07 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.08
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.64 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.05 Err1_2= 0.01
 91    0.005  0.569  0.001  -2.767 -2.798 -0.144  -0.374 -2.900 -0.660
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.54 ->Y2_1= 0.25 ==> Err2_1=-0.25 ->Err1_1= 0.02 Err1_2= 0.18
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.64 Y1_2= 0.07 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.07 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.08
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.64 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.05 Err1_2= 0.01
 92    0.015  0.581  0.004  -2.831 -2.862 -0.174  -0.385 -2.972 -0.678
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.54 ->Y2_1= 0.24 ==> Err2_1=-0.24 ->Err1_1= 0.02 Err1_2= 0.18
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.64 Y1_2= 0.06 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.07 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.08
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.64 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.05 Err1_2= 0.01
 93    0.026  0.594  0.007  -2.895 -2.926 -0.202  -0.397 -3.043 -0.696
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.55 ->Y2_1= 0.24 ==> Err2_1=-0.24 ->Err1_1= 0.02 Err1_2= 0.18
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.64 Y1_2= 0.06 ->Y2_1= 0.56 ==> Err2_1= 0.44 ->Err1_1=-0.04 Err1_2=-0.08
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.06 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.04 Err1_2=-0.08
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.65 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.05 Err1_2= 0.01
 94    0.037  0.608  0.010  -2.958 -2.989 -0.230  -0.410 -3.113 -0.715
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.56 ->Y2_1= 0.23 ==> Err2_1=-0.23 ->Err1_1= 0.02 Err1_2= 0.17
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.06 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.08
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.06 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.04 Err1_2=-0.08
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.65 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.06 Err1_2= 0.01
 95    0.049  0.622  0.014  -3.021 -3.052 -0.256  -0.425 -3.183 -0.734
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.56 ->Y2_1= 0.22 ==> Err2_1=-0.22 ->Err1_1= 0.02 Err1_2= 0.17
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.06 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.06 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.04 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.66 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.06 Err1_2= 0.01
 96    0.062  0.636  0.018  -3.082 -3.113 -0.282  -0.441 -3.251 -0.753
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.50 Y1_2= 0.57 ->Y2_1= 0.21 ==> Err2_1=-0.21 ->Err1_1= 0.02 Err1_2= 0.17
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.06 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.06 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.66 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.06 Err1_2= 0.01
 97    0.075  0.652  0.023  -3.143 -3.175 -0.306  -0.458 -3.318 -0.773
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.58 ->Y2_1= 0.20 ==> Err2_1=-0.20 ->Err1_1= 0.02 Err1_2= 0.16
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.05 ->Y2_1= 0.57 ==> Err2_1= 0.43 ->Err1_1=-0.04 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.06 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.67 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.06 Err1_2= 0.00
 98    0.088  0.668  0.029  -3.203 -3.235 -0.330  -0.477 -3.383 -0.794
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.58 ->Y2_1= 0.20 ==> Err2_1=-0.20 ->Err1_1= 0.02 Err1_2= 0.16
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.05 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.05 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.67 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.06 Err1_2= 0.00
 99    0.102  0.685  0.035  -3.262 -3.294 -0.353  -0.497 -3.448 -0.814
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.59 ->Y2_1= 0.19 ==> Err2_1=-0.19 ->Err1_1= 0.02 Err1_2= 0.16
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.05 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.05 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.68 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.07 Err1_2= 0.00
100    0.117  0.702  0.042  -3.320 -3.353 -0.375  -0.519 -3.511 -0.836
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.59 ->Y2_1= 0.18 ==> Err2_1=-0.18 ->Err1_1= 0.02 Err1_2= 0.15
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.05 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.05 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.69 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.07 Err1_2= 0.00
101    0.132  0.720  0.049  -3.377 -3.410 -0.396  -0.542 -3.573 -0.858
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.60 ->Y2_1= 0.18 ==> Err2_1=-0.18 ->Err1_1= 0.02 Err1_2= 0.15
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.05 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.05 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.05 Err1_2=-0.07
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.69 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.07 Err1_2= 0.00
102    0.147  0.739  0.058  -3.433 -3.467 -0.417  -0.566 -3.634 -0.880
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.49 Y1_2= 0.60 ->Y2_1= 0.17 ==> Err2_1=-0.17 ->Err1_1= 0.02 Err1_2= 0.15
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.05 ->Y2_1= 0.58 ==> Err2_1= 0.42 ->Err1_1=-0.05 Err1_2=-0.07
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.05 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.06 Err1_2=-0.06
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.70 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.07 Err1_2= 0.00
103    0.163  0.758  0.067  -3.488 -3.523 -0.436  -0.592 -3.693 -0.903
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.48 Y1_2= 0.61 ->Y2_1= 0.16 ==> Err2_1=-0.16 ->Err1_1= 0.02 Err1_2= 0.14
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.05 Err1_2=-0.06
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.05 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.06 Err1_2=-0.06
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.70 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.08 Err1_2= 0.00
104    0.180  0.778  0.078  -3.541 -3.577 -0.455  -0.619 -3.751 -0.927
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.48 Y1_2= 0.61 ->Y2_1= 0.16 ==> Err2_1=-0.16 ->Err1_1= 0.02 Err1_2= 0.14
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.06 Err1_2=-0.06
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.06 Err1_2=-0.06
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.71 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.08 Err1_2= 0.00
105    0.197  0.799  0.090  -3.594 -3.631 -0.474  -0.648 -3.808 -0.951
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.48 Y1_2= 0.62 ->Y2_1= 0.15 ==> Err2_1=-0.15 ->Err1_1= 0.02 Err1_2= 0.14
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.06 Err1_2=-0.06
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.06 Err1_2=-0.06
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.71 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.08 Err1_2= 0.00
106    0.214  0.820  0.103  -3.646 -3.684 -0.492  -0.679 -3.864 -0.976
```

```
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.47 Y1_2= 0.62 ->Y2_1= 0.15 ==> Err2_1=-0.15 ->Err1_1= 0.03 Err1_2= 0.14
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.06 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.07 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.72 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.09 Err1_2= 0.00
107    0.232  0.841  0.118  -3.696 -3.735 -0.509  -0.710 -3.919 -1.002
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.47 Y1_2= 0.62 ->Y2_1= 0.14 ==> Err2_1=-0.14 ->Err1_1= 0.03 Err1_2= 0.13
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.06 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.07 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.72 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.09 Err1_2= 0.00
108    0.250  0.863  0.134  -3.746 -3.786 -0.526  -0.744 -3.973 -1.028
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.47 Y1_2= 0.63 ->Y2_1= 0.14 ==> Err2_1=-0.14 ->Err1_1= 0.03 Err1_2= 0.13
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.07 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.07 Err1_2=-0.06
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.73 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.09 Err1_2= 0.00
109    0.268  0.886  0.152  -3.794 -3.836 -0.543  -0.779 -4.026 -1.055
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.46 Y1_2= 0.63 ->Y2_1= 0.14 ==> Err2_1=-0.14 ->Err1_1= 0.03 Err1_2= 0.13
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.04 ->Y2_1= 0.59 ==> Err2_1= 0.41 ->Err1_1=-0.07 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.07 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.73 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.09 Err1_2= 0.00
110    0.287  0.909  0.172  -3.842 -3.884 -0.559  -0.815 -4.078 -1.082
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.46 Y1_2= 0.64 ->Y2_1= 0.13 ==> Err2_1=-0.13 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.07 Err1_2=-0.06
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.08 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.10 Err1_2= 0.00
111    0.307  0.932  0.193  -3.888 -3.932 -0.575  -0.853 -4.129 -1.110
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.45 Y1_2= 0.64 ->Y2_1= 0.13 ==> Err2_1=-0.13 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.08 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.04 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.08 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.10 Err1_2= 0.00
112    0.326  0.955  0.217  -3.934 -3.979 -0.590  -0.893 -4.179 -1.139
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.45 Y1_2= 0.64 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.08 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.03 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.08 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.62 ==> Err2_1=-0.62 ->Err1_1= 0.10 Err1_2= 0.00
113    0.346  0.979  0.243  -3.978 -4.026 -0.606  -0.934 -4.229 -1.168
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.44 Y1_2= 0.65 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.08 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.53 Y1_2= 0.03 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.09 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.11 Err1_2= 0.00
114    0.366  1.003  0.271  -4.022 -4.071 -0.621  -0.977 -4.277 -1.198
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.43 Y1_2= 0.65 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.03 Err1_2= 0.12
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.68 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.09 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.03 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.09 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.11 Err1_2= 0.00
115    0.387  1.027  0.302  -4.064 -4.115 -0.636  -1.022 -4.325 -1.229
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.43 Y1_2= 0.65 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.09 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.03 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.09 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.12 Err1_2= 0.00
116    0.409  1.051  0.335  -4.106 -4.159 -0.651  -1.069 -4.373 -1.259
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.42 Y1_2= 0.66 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.09 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.52 Y1_2= 0.03 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.10 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.12 Err1_2= 0.00
117    0.431  1.076  0.371  -4.147 -4.202 -0.666  -1.117 -4.419 -1.291
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.41 Y1_2= 0.66 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.10 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.03 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.10 Err1_2=-0.05
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.13 Err1_2= 0.00
118    0.454  1.100  0.410  -4.187 -4.245 -0.681  -1.168 -4.466 -1.323
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.40 Y1_2= 0.66 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.67 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.10 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.03 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.10 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.13 Err1_2= 0.00
119    0.477  1.124  0.451  -4.227 -4.286 -0.696  -1.220 -4.512 -1.355
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.39 Y1_2= 0.67 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.03 Err1_2= 0.11
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.03 ->Y2_1= 0.60 ==> Err2_1= 0.40 ->Err1_1=-0.11 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.51 Y1_2= 0.03 ->Y2_1= 0.65 ==> Err2_1= 0.35 ->Err1_1=-0.11 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.61 ==> Err2_1=-0.61 ->Err1_1= 0.13 Err1_2= 0.00
120    0.502  1.149  0.496  -4.265 -4.327 -0.711  -1.275 -4.557 -1.388
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.38 Y1_2= 0.67 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.66 Y1_2= 0.03 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.11 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.03 ->Y2_1= 0.65 ==> Err2_1= 0.35 ->Err1_1=-0.11 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.14 Err1_2= 0.00
121    0.528  1.173  0.544  -4.303 -4.367 -0.727  -1.332 -4.602 -1.421
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.37 Y1_2= 0.67 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
 | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.03 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.12 Err1_2=-0.05
 | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.50 Y1_2= 0.03 ->Y2_1= 0.65 ==> Err2_1= 0.35 ->Err1_1=-0.12 Err1_2=-0.04
 | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.15 Err1_2= 0.00
122    0.555  1.197  0.595  -4.340 -4.407 -0.742  -1.391 -4.647 -1.454
 | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.36 Y1_2= 0.68 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
```

```
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.65 Y1_2= 0.02 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.12 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.49 Y1_2= 0.03 ->Y2_1= 0.66 ==> Err2_1= 0.34 ->Err1_1=-0.12 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.60 ==> Err2_1=-0.60 ->Err1_1= 0.15 Err1_2= 0.00
123    0.583  1.221  0.650  -4.377 -4.446 -0.758  -1.453 -4.691 -1.487
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.34 Y1_2= 0.68 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.64 Y1_2= 0.02 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.13 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.03 ->Y2_1= 0.66 ==> Err2_1= 0.34 ->Err1_1=-0.12 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.16 Err1_2= 0.00
124    0.614  1.245  0.708  -4.412 -4.485 -0.774  -1.517 -4.736 -1.521
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.33 Y1_2= 0.68 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.63 Y1_2= 0.02 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.14 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.48 Y1_2= 0.03 ->Y2_1= 0.66 ==> Err2_1= 0.34 ->Err1_1=-0.13 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.16 Err1_2= 0.00
125    0.646  1.268  0.770  -4.447 -4.523 -0.790  -1.585 -4.780 -1.555
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.32 Y1_2= 0.69 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.62 Y1_2= 0.02 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.14 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.47 Y1_2= 0.03 ->Y2_1= 0.67 ==> Err2_1= 0.33 ->Err1_1=-0.13 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.59 ==> Err2_1=-0.59 ->Err1_1= 0.17 Err1_2= 0.00
126    0.681  1.292  0.835  -4.482 -4.560 -0.807  -1.655 -4.824 -1.589
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.30 Y1_2= 0.69 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.61 Y1_2= 0.02 ->Y2_1= 0.61 ==> Err2_1= 0.39 ->Err1_1=-0.15 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.46 Y1_2= 0.02 ->Y2_1= 0.67 ==> Err2_1= 0.33 ->Err1_1=-0.14 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.58 ==> Err2_1=-0.58 ->Err1_1= 0.18 Err1_2= 0.00
127    0.718  1.315  0.904  -4.516 -4.597 -0.824  -1.728 -4.869 -1.623
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.29 Y1_2= 0.70 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.60 Y1_2= 0.02 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.16 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.02 ->Y2_1= 0.67 ==> Err2_1= 0.33 ->Err1_1=-0.14 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.58 ==> Err2_1=-0.58 ->Err1_1= 0.18 Err1_2= 0.00
128    0.758  1.338  0.977  -4.549 -4.633 -0.842  -1.805 -4.913 -1.657
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.27 Y1_2= 0.70 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.59 Y1_2= 0.02 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.17 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.45 Y1_2= 0.02 ->Y2_1= 0.68 ==> Err2_1= 0.32 ->Err1_1=-0.14 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.19 Err1_2= 0.00
129    0.801  1.361  1.054  -4.582 -4.669 -0.860  -1.885 -4.957 -1.691
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.26 Y1_2= 0.70 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.58 Y1_2= 0.02 ->Y2_1= 0.62 ==> Err2_1= 0.38 ->Err1_1=-0.17 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.44 Y1_2= 0.02 ->Y2_1= 0.68 ==> Err2_1= 0.32 ->Err1_1=-0.15 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.57 ==> Err2_1=-0.57 ->Err1_1= 0.20 Err1_2= 0.00
130    0.847  1.384  1.134  -4.615 -4.704 -0.878  -1.969 -5.002 -1.726
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.24 Y1_2= 0.71 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.56 Y1_2= 0.02 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.18 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.43 Y1_2= 0.02 ->Y2_1= 0.68 ==> Err2_1= 0.32 ->Err1_1=-0.15 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.21 Err1_2= 0.00
131    0.896  1.407  1.219  -4.647 -4.739 -0.897  -2.056 -5.047 -1.760
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.23 Y1_2= 0.71 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.10
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.55 Y1_2= 0.02 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.19 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.42 Y1_2= 0.02 ->Y2_1= 0.69 ==> Err2_1= 0.31 ->Err1_1=-0.16 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.56 ==> Err2_1=-0.56 ->Err1_1= 0.22 Err1_2= 0.00
132    0.948  1.431  1.306  -4.678 -4.773 -0.916  -2.147 -5.092 -1.794
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.21 Y1_2= 0.71 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.53 Y1_2= 0.02 ->Y2_1= 0.63 ==> Err2_1= 0.37 ->Err1_1=-0.20 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.41 Y1_2= 0.02 ->Y2_1= 0.69 ==> Err2_1= 0.31 ->Err1_1=-0.16 Err1_2=-0.04
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.55 ==> Err2_1=-0.55 ->Err1_1= 0.22 Err1_2= 0.00
133    1.004  1.456  1.398  -4.709 -4.806 -0.936  -2.242 -5.138 -1.829
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.20 Y1_2= 0.72 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.51 Y1_2= 0.02 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.20 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.40 Y1_2= 0.02 ->Y2_1= 0.69 ==> Err2_1= 0.31 ->Err1_1=-0.17 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.54 ==> Err2_1=-0.54 ->Err1_1= 0.23 Err1_2= 0.00
134    1.063  1.482  1.492  -4.740 -4.839 -0.957  -2.340 -5.183 -1.863
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.18 Y1_2= 0.72 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.50 Y1_2= 0.02 ->Y2_1= 0.64 ==> Err2_1= 0.36 ->Err1_1=-0.21 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.39 Y1_2= 0.02 ->Y2_1= 0.70 ==> Err2_1= 0.30 ->Err1_1=-0.17 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.53 ==> Err2_1=-0.53 ->Err1_1= 0.24 Err1_2= 0.00
135    1.124  1.510  1.589  -4.771 -4.872 -0.978  -2.443 -5.230 -1.898
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.17 Y1_2= 0.73 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.48 Y1_2= 0.02 ->Y2_1= 0.65 ==> Err2_1= 0.35 ->Err1_1=-0.21 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.39 Y1_2= 0.02 ->Y2_1= 0.70 ==> Err2_1= 0.30 ->Err1_1=-0.17 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.52 ==> Err2_1=-0.52 ->Err1_1= 0.25 Err1_2= 0.00
136    1.188  1.539  1.689  -4.801 -4.903 -0.999  -2.548 -5.276 -1.933
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.16 Y1_2= 0.73 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.46 Y1_2= 0.02 ->Y2_1= 0.66 ==> Err2_1= 0.34 ->Err1_1=-0.22 Err1_2=-0.04
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.38 Y1_2= 0.02 ->Y2_1= 0.70 ==> Err2_1= 0.30 ->Err1_1=-0.18 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.51 ==> Err2_1=-0.51 ->Err1_1= 0.25 Err1_2= 0.00
137    1.255  1.570  1.792  -4.831 -4.935 -1.022  -2.658 -5.323 -1.968
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.14 Y1_2= 0.74 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.44 Y1_2= 0.02 ->Y2_1= 0.66 ==> Err2_1= 0.34 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.37 Y1_2= 0.02 ->Y2_1= 0.71 ==> Err2_1= 0.29 ->Err1_1=-0.18 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.50 ==> Err2_1=-0.50 ->Err1_1= 0.26 Err1_2= 0.00
138    1.323  1.604  1.896  -4.860 -4.965 -1.044  -2.770 -5.370 -2.004
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.13 Y1_2= 0.74 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.43 Y1_2= 0.02 ->Y2_1= 0.67 ==> Err2_1= 0.33 ->Err1_1=-0.22 Err1_2=-0.03
```

```
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.36 Y1_2= 0.02 ->Y2_1= 0.71 ==> Err2_1= 0.29 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.49 ==> Err2_1=-0.49 ->Err1_1= 0.26 Err1_2= 0.00
139    1.392  1.641  2.002  -4.889 -4.995 -1.067  -2.885 -5.418 -2.041
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.12 Y1_2= 0.74 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.41 Y1_2= 0.02 ->Y2_1= 0.68 ==> Err2_1= 0.32 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.35 Y1_2= 0.02 ->Y2_1= 0.71 ==> Err2_1= 0.29 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.48 ==> Err2_1=-0.48 ->Err1_1= 0.27 Err1_2= 0.00
140    1.463  1.680  2.109  -4.918 -5.024 -1.090  -3.002 -5.466 -2.078
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.11 Y1_2= 0.75 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.03 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.39 Y1_2= 0.02 ->Y2_1= 0.69 ==> Err2_1= 0.31 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.34 Y1_2= 0.02 ->Y2_1= 0.72 ==> Err2_1= 0.28 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.47 ==> Err2_1=-0.47 ->Err1_1= 0.27 Err1_2= 0.00
141    1.533  1.722  2.216  -4.947 -5.053 -1.114  -3.121 -5.515 -2.115
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.10 Y1_2= 0.75 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.38 Y1_2= 0.02 ->Y2_1= 0.70 ==> Err2_1= 0.30 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.34 Y1_2= 0.02 ->Y2_1= 0.72 ==> Err2_1= 0.28 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.45 ==> Err2_1=-0.45 ->Err1_1= 0.27 Err1_2= 0.00
142    1.604  1.766  2.325  -4.975 -5.081 -1.139  -3.242 -5.564 -2.154
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.09 Y1_2= 0.76 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.36 Y1_2= 0.02 ->Y2_1= 0.70 ==> Err2_1= 0.30 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.33 Y1_2= 0.02 ->Y2_1= 0.73 ==> Err2_1= 0.27 ->Err1_1=-0.20 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.44 ==> Err2_1=-0.44 ->Err1_1= 0.27 Err1_2= 0.00
143    1.675  1.813  2.433  -5.004 -5.108 -1.163  -3.364 -5.613 -2.193
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.08 Y1_2= 0.76 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.35 Y1_2= 0.02 ->Y2_1= 0.71 ==> Err2_1= 0.29 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.32 Y1_2= 0.02 ->Y2_1= 0.73 ==> Err2_1= 0.27 ->Err1_1=-0.20 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.42 ==> Err2_1=-0.42 ->Err1_1= 0.27 Err1_2= 0.00
144    1.745  1.862  2.541  -5.031 -5.135 -1.188  -3.486 -5.662 -2.232
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.07 Y1_2= 0.77 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.34 Y1_2= 0.02 ->Y2_1= 0.72 ==> Err2_1= 0.28 ->Err1_1=-0.22 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.31 Y1_2= 0.02 ->Y2_1= 0.74 ==> Err2_1= 0.26 ->Err1_1=-0.20 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.74 Y1_2= 0.00 ->Y2_1= 0.41 ==> Err2_1=-0.41 ->Err1_1= 0.27 Err1_2= 0.00
145    1.814  1.913  2.649  -5.058 -5.161 -1.213  -3.609 -5.712 -2.272
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.07 Y1_2= 0.77 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.32 Y1_2= 0.02 ->Y2_1= 0.73 ==> Err2_1= 0.27 ->Err1_1=-0.21 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.30 Y1_2= 0.02 ->Y2_1= 0.74 ==> Err2_1= 0.26 ->Err1_1=-0.20 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.40 ==> Err2_1=-0.40 ->Err1_1= 0.27 Err1_2= 0.00
146    1.882  1.965  2.756  -5.085 -5.187 -1.238  -3.731 -5.762 -2.313
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.06 Y1_2= 0.78 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.02 Err1_2= 0.09
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.31 Y1_2= 0.02 ->Y2_1= 0.74 ==> Err2_1= 0.26 ->Err1_1=-0.21 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.29 Y1_2= 0.02 ->Y2_1= 0.75 ==> Err2_1= 0.25 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.38 ==> Err2_1=-0.38 ->Err1_1= 0.27 Err1_2= 0.00
147    1.948  2.018  2.862  -5.112 -5.211 -1.263  -3.854 -5.812 -2.353
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.05 Y1_2= 0.78 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.02 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.30 Y1_2= 0.02 ->Y2_1= 0.75 ==> Err2_1= 0.25 ->Err1_1=-0.20 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.29 Y1_2= 0.02 ->Y2_1= 0.76 ==> Err2_1= 0.24 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.37 ==> Err2_1=-0.37 ->Err1_1= 0.26 Err1_2= 0.00
148    2.013  2.072  2.966  -5.138 -5.236 -1.288  -3.975 -5.862 -2.394
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.05 Y1_2= 0.78 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.02 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.29 Y1_2= 0.02 ->Y2_1= 0.76 ==> Err2_1= 0.24 ->Err1_1=-0.20 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.28 Y1_2= 0.02 ->Y2_1= 0.76 ==> Err2_1= 0.24 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.75 Y1_2= 0.00 ->Y2_1= 0.35 ==> Err2_1=-0.35 ->Err1_1= 0.26 Err1_2= 0.00
149    2.077  2.127  3.068  -5.163 -5.259 -1.314  -4.095 -5.913 -2.435
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.04 Y1_2= 0.79 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.28 Y1_2= 0.02 ->Y2_1= 0.76 ==> Err2_1= 0.24 ->Err1_1=-0.20 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.27 Y1_2= 0.02 ->Y2_1= 0.77 ==> Err2_1= 0.23 ->Err1_1=-0.19 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.34 ==> Err2_1=-0.34 ->Err1_1= 0.26 Err1_2= 0.00
150    2.139  2.181  3.169  -5.188 -5.283 -1.339  -4.214 -5.963 -2.475
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.04 Y1_2= 0.79 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.27 Y1_2= 0.02 ->Y2_1= 0.77 ==> Err2_1= 0.23 ->Err1_1=-0.19 Err1_2=-0.03
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.26 Y1_2= 0.02 ->Y2_1= 0.78 ==> Err2_1= 0.22 ->Err1_1=-0.18 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.33 ==> Err2_1=-0.33 ->Err1_1= 0.25 Err1_2= 0.00
151    2.200  2.236  3.267  -5.212 -5.305 -1.364  -4.331 -6.013 -2.515
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.04 Y1_2= 0.80 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.26 Y1_2= 0.02 ->Y2_1= 0.78 ==> Err2_1= 0.22 ->Err1_1=-0.18 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.26 Y1_2= 0.02 ->Y2_1= 0.78 ==> Err2_1= 0.22 ->Err1_1=-0.18 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.76 Y1_2= 0.00 ->Y2_1= 0.31 ==> Err2_1=-0.31 ->Err1_1= 0.24 Err1_2= 0.00
152    2.259  2.289  3.364  -5.236 -5.327 -1.389  -4.446 -6.063 -2.555
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.03 Y1_2= 0.80 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.25 Y1_2= 0.02 ->Y2_1= 0.79 ==> Err2_1= 0.21 ->Err1_1=-0.18 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.25 Y1_2= 0.02 ->Y2_1= 0.79 ==> Err2_1= 0.21 ->Err1_1=-0.17 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.77 Y1_2= 0.00 ->Y2_1= 0.30 ==> Err2_1=-0.30 ->Err1_1= 0.24 Err1_2= 0.00
153    2.317  2.343  3.457  -5.259 -5.349 -1.413  -4.559 -6.113 -2.595
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.03 Y1_2= 0.80 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.08
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.25 Y1_2= 0.02 ->Y2_1= 0.79 ==> Err2_1= 0.21 ->Err1_1=-0.17 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.24 Y1_2= 0.02 ->Y2_1= 0.80 ==> Err2_1= 0.20 ->Err1_1=-0.17 Err1_2=-0.03
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.77 Y1_2= 0.00 ->Y2_1= 0.29 ==> Err2_1=-0.29 ->Err1_1= 0.23 Err1_2= 0.00
154    2.372  2.395  3.549  -5.282 -5.370 -1.437  -4.669 -6.163 -2.634
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.03 Y1_2= 0.81 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.24 Y1_2= 0.02 ->Y2_1= 0.80 ==> Err2_1= 0.20 ->Err1_1=-0.17 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.24 Y1_2= 0.02 ->Y2_1= 0.80 ==> Err2_1= 0.20 ->Err1_1=-0.17 Err1_2=-0.02
```

```
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.77 Y1_2= 0.00 ->Y2_1= 0.27 ==> Err2_1=-0.27 ->Err1_1= 0.23 Err1_2= 0.00
155    2.427  2.446  3.638  -5.304 -5.391 -1.461  -4.777 -6.212 -2.673
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.03 Y1_2= 0.81 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.23 Y1_2= 0.02 ->Y2_1= 0.81 ==> Err2_1= 0.19 ->Err1_1=-0.16 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.23 Y1_2= 0.02 ->Y2_1= 0.81 ==> Err2_1= 0.19 ->Err1_1=-0.16 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.77 Y1_2= 0.00 ->Y2_1= 0.26 ==> Err2_1=-0.26 ->Err1_1= 0.22 Err1_2= 0.00
156    2.479  2.497  3.724  -5.326 -5.411 -1.484  -4.883 -6.261 -2.711
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.82 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.23 Y1_2= 0.02 ->Y2_1= 0.82 ==> Err2_1= 0.18 ->Err1_1=-0.16 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.22 Y1_2= 0.02 ->Y2_1= 0.82 ==> Err2_1= 0.18 ->Err1_1=-0.16 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.78 Y1_2= 0.00 ->Y2_1= 0.25 ==> Err2_1=-0.25 ->Err1_1= 0.21 Err1_2= 0.00
157    2.530  2.546  3.808  -5.347 -5.430 -1.507  -4.987 -6.309 -2.749
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.82 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.22 Y1_2= 0.02 ->Y2_1= 0.82 ==> Err2_1= 0.18 ->Err1_1=-0.15 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.22 Y1_2= 0.02 ->Y2_1= 0.82 ==> Err2_1= 0.18 ->Err1_1=-0.15 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.78 Y1_2= 0.00 ->Y2_1= 0.24 ==> Err2_1=-0.24 ->Err1_1= 0.21 Err1_2= 0.00
158    2.580  2.594  3.889  -5.368 -5.450 -1.530  -5.087 -6.358 -2.786
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.82 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.21 Y1_2= 0.02 ->Y2_1= 0.83 ==> Err2_1= 0.17 ->Err1_1=-0.15 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.21 Y1_2= 0.02 ->Y2_1= 0.83 ==> Err2_1= 0.17 ->Err1_1=-0.15 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.78 Y1_2= 0.00 ->Y2_1= 0.23 ==> Err2_1=-0.23 ->Err1_1= 0.20 Err1_2= 0.00
159    2.628  2.641  3.968  -5.388 -5.468 -1.552  -5.186 -6.405 -2.822
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.83 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.07
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.21 Y1_2= 0.02 ->Y2_1= 0.83 ==> Err2_1= 0.17 ->Err1_1=-0.14 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.21 Y1_2= 0.02 ->Y2_1= 0.83 ==> Err2_1= 0.17 ->Err1_1=-0.14 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.79 Y1_2= 0.00 ->Y2_1= 0.22 ==> Err2_1=-0.22 ->Err1_1= 0.19 Err1_2= 0.00
160    2.674  2.686  4.044  -5.408 -5.487 -1.573  -5.281 -6.453 -2.857
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.83 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.20 Y1_2= 0.02 ->Y2_1= 0.84 ==> Err2_1= 0.16 ->Err1_1=-0.14 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.20 Y1_2= 0.02 ->Y2_1= 0.84 ==> Err2_1= 0.16 ->Err1_1=-0.14 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.79 Y1_2= 0.00 ->Y2_1= 0.21 ==> Err2_1=-0.21 ->Err1_1= 0.19 Err1_2= 0.00
161    2.719  2.730  4.118  -5.427 -5.504 -1.594  -5.374 -6.499 -2.892
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.02 Y1_2= 0.83 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.20 Y1_2= 0.02 ->Y2_1= 0.84 ==> Err2_1= 0.16 ->Err1_1=-0.13 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.20 Y1_2= 0.02 ->Y2_1= 0.84 ==> Err2_1= 0.16 ->Err1_1=-0.13 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.79 Y1_2= 0.00 ->Y2_1= 0.20 ==> Err2_1=-0.20 ->Err1_1= 0.18 Err1_2= 0.00
162    2.763  2.773  4.189  -5.446 -5.522 -1.615  -5.465 -6.545 -2.926
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.83 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.20 Y1_2= 0.02 ->Y2_1= 0.85 ==> Err2_1= 0.15 ->Err1_1=-0.13 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.19 Y1_2= 0.02 ->Y2_1= 0.85 ==> Err2_1= 0.15 ->Err1_1=-0.13 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.79 Y1_2= 0.00 ->Y2_1= 0.20 ==> Err2_1=-0.20 ->Err1_1= 0.18 Err1_2= 0.00
163    2.805  2.815  4.259  -5.464 -5.539 -1.635  -5.553 -6.591 -2.959
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.84 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.01 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.19 Y1_2= 0.02 ->Y2_1= 0.85 ==> Err2_1= 0.15 ->Err1_1=-0.12 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.19 Y1_2= 0.02 ->Y2_1= 0.85 ==> Err2_1= 0.15 ->Err1_1=-0.12 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.80 Y1_2= 0.00 ->Y2_1= 0.19 ==> Err2_1=-0.19 ->Err1_1= 0.17 Err1_2= 0.00
164    2.846  2.855  4.326  -5.482 -5.555 -1.655  -5.639 -6.636 -2.992
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.84 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.00 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.19 Y1_2= 0.02 ->Y2_1= 0.86 ==> Err2_1= 0.14 ->Err1_1=-0.12 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.19 Y1_2= 0.02 ->Y2_1= 0.86 ==> Err2_1= 0.14 ->Err1_1=-0.12 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.80 Y1_2= 0.00 ->Y2_1= 0.18 ==> Err2_1=-0.18 ->Err1_1= 0.16 Err1_2= 0.00
165    2.886  2.894  4.390  -5.500 -5.572 -1.674  -5.722 -6.681 -3.024
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.84 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.18 Y1_2= 0.02 ->Y2_1= 0.86 ==> Err2_1= 0.14 ->Err1_1=-0.12 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.18 Y1_2= 0.02 ->Y2_1= 0.86 ==> Err2_1= 0.14 ->Err1_1=-0.12 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.80 Y1_2= 0.00 ->Y2_1= 0.17 ==> Err2_1=-0.17 ->Err1_1= 0.16 Err1_2= 0.00
166    2.924  2.932  4.453  -5.517 -5.588 -1.692  -5.804 -6.725 -3.055
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.84 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.06
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.18 Y1_2= 0.02 ->Y2_1= 0.87 ==> Err2_1= 0.13 ->Err1_1=-0.11 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.18 Y1_2= 0.02 ->Y2_1= 0.87 ==> Err2_1= 0.13 ->Err1_1=-0.11 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.80 Y1_2= 0.00 ->Y2_1= 0.17 ==> Err2_1=-0.17 ->Err1_1= 0.15 Err1_2= 0.00
167    2.961  2.969  4.514  -5.534 -5.603 -1.711  -5.883 -6.768 -3.086
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.85 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.18 Y1_2= 0.02 ->Y2_1= 0.87 ==> Err2_1= 0.13 ->Err1_1=-0.11 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.87 ==> Err2_1= 0.13 ->Err1_1=-0.11 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.80 Y1_2= 0.00 ->Y2_1= 0.16 ==> Err2_1=-0.16 ->Err1_1= 0.15 Err1_2= 0.00
168    2.997  3.005  4.573  -5.550 -5.618 -1.728  -5.960 -6.811 -3.116
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.85 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.11 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.11 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.81 Y1_2= 0.00 ->Y2_1= 0.16 ==> Err2_1=-0.16 ->Err1_1= 0.14 Err1_2= 0.00
169    3.032  3.040  4.630  -5.566 -5.633 -1.745  -6.034 -6.853 -3.145
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.85 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.10 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.10 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.81 Y1_2= 0.00 ->Y2_1= 0.15 ==> Err2_1=-0.15 ->Err1_1= 0.14 Err1_2= 0.00
170    3.066  3.073  4.685  -5.581 -5.648 -1.762  -6.107 -6.895 -3.173
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.85 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.10 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.17 Y1_2= 0.02 ->Y2_1= 0.88 ==> Err2_1= 0.12 ->Err1_1=-0.10 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.81 Y1_2= 0.00 ->Y2_1= 0.14 ==> Err2_1=-0.14 ->Err1_1= 0.14 Err1_2= 0.00
```

```
171    3.099  3.106  4.739  -5.597 -5.662 -1.778  -6.178 -6.936 -3.201
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.10 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.10 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.81 Y1_2= 0.00 ->Y2_1= 0.14 ==> Err2_1=-0.14 ->Err1_1= 0.13 Err1_2= 0.00
172    3.131  3.138  4.791  -5.612 -5.676 -1.794  -6.247 -6.976 -3.229
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.09 Err1_2=-0.02
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.09 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.81 Y1_2= 0.00 ->Y2_1= 0.13 ==> Err2_1=-0.13 ->Err1_1= 0.13 Err1_2= 0.00
173    3.162  3.168  4.841  -5.626 -5.690 -1.810  -6.315 -7.016 -3.255
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.09 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.09 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.13 ==> Err2_1=-0.13 ->Err1_1= 0.12 Err1_2= 0.00
174    3.192  3.198  4.890  -5.641 -5.703 -1.825  -6.380 -7.055 -3.282
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.16 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.09 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.89 ==> Err2_1= 0.11 ->Err1_1=-0.09 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.13 ==> Err2_1=-0.13 ->Err1_1= 0.12 Err1_2= 0.00
175    3.221  3.227  4.937  -5.655 -5.716 -1.839  -6.444 -7.094 -3.307
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.09 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.09 Err1_2=-0.02
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.12 Err1_2= 0.00
176    3.249  3.256  4.983  -5.668 -5.729 -1.854  -6.506 -7.132 -3.332
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.86 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.08 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.08 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.12 ==> Err2_1=-0.12 ->Err1_1= 0.11 Err1_2= 0.00
177    3.277  3.283  5.028  -5.682 -5.742 -1.868  -6.567 -7.170 -3.357
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.08 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.90 ==> Err2_1= 0.10 ->Err1_1=-0.08 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.11 Err1_2= 0.00
178    3.304  3.310  5.072  -5.695 -5.754 -1.881  -6.627 -7.207 -3.381
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.08 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.15 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.08 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.82 Y1_2= 0.00 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.11 Err1_2= 0.00
179    3.330  3.336  5.114  -5.708 -5.766 -1.894  -6.685 -7.244 -3.404
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.08 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.08 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.10 Err1_2= 0.00
180    3.355  3.361  5.155  -5.720 -5.778 -1.907  -6.741 -7.280 -3.427
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.07 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.91 ==> Err2_1= 0.09 ->Err1_1=-0.07 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.10 Err1_2= 0.00
181    3.380  3.386  5.197  -5.733 -5.790 -1.883  -6.796 -7.277 -3.495
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.05
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.11 ==> Err2_1=-0.11 ->Err1_1= 0.10 Err1_2= 0.00
182    3.411  3.417  5.226  -5.744 -5.801 -1.902  -6.854 -7.316 -3.501
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.10 Err1_2= 0.00
183    3.434  3.440  5.264  -5.755 -5.811 -1.919  -6.908 -7.354 -3.516
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.01 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.10 ==> Err2_1=-0.10 ->Err1_1= 0.09 Err1_2= 0.00
184    3.455  3.461  5.302  -5.767 -5.822 -1.934  -6.959 -7.390 -3.533
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.87 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.14 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.07 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.09 Err1_2= 0.00
185    3.477  3.482  5.340  -5.778 -5.832 -1.912  -7.010 -7.387 -3.594
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.83 Y1_2= 0.00 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.09 Err1_2= 0.00
186    3.504  3.509  5.367  -5.788 -5.842 -1.932  -7.063 -7.426 -3.597
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.09 Err1_2= 0.00
187    3.524  3.530  5.402  -5.798 -5.852 -1.950  -7.112 -7.464 -3.608
```

```
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.92 ==> Err2_1= 0.08 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.08 Err1_2= 0.00
188    3.543  3.549  5.438  -5.808 -5.862 -1.930  -7.159 -7.461 -3.666
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.87 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.09 ==> Err2_1=-0.09 ->Err1_1= 0.08 Err1_2= 0.00
189    3.568  3.573  5.464  -5.818 -5.870 -1.951  -7.209 -7.500 -3.666
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.08 Err1_2= 0.00
190    3.586  3.592  5.496  -5.827 -5.879 -1.970  -7.255 -7.538 -3.673
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.06 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.08 Err1_2= 0.00
191    3.604  3.609  5.530  -5.837 -5.889 -1.951  -7.299 -7.535 -3.728
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.84 Y1_2= 0.00 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.08 Err1_2= 0.00
192    3.626  3.631  5.555  -5.846 -5.897 -1.972  -7.346 -7.574 -3.727
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.93 ==> Err2_1= 0.07 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.07 Err1_2= 0.00
193    3.643  3.649  5.586  -5.855 -5.906 -1.955  -7.389 -7.571 -3.777
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.08 ==> Err2_1=-0.08 ->Err1_1= 0.07 Err1_2= 0.00
194    3.665  3.670  5.610  -5.863 -5.913 -1.978  -7.434 -7.611 -3.773
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.13 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.07 Err1_2= 0.00
195    3.681  3.686  5.639  -5.872 -5.922 -1.998  -7.476 -7.649 -3.776
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.07 Err1_2= 0.00
196    3.696  3.701  5.670  -5.880 -5.930 -1.981  -7.516 -7.646 -3.826
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.07 Err1_2= 0.00
197    3.715  3.720  5.693  -5.888 -5.937 -2.003  -7.558 -7.685 -3.822
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.07 Err1_2= 0.00
198    3.731  3.736  5.721  -5.897 -5.945 -1.987  -7.597 -7.683 -3.869
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.07 Err1_2= 0.00
199    3.749  3.754  5.743  -5.904 -5.952 -2.010  -7.638 -7.721 -3.863
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.06 Err1_2= 0.00
200    3.764  3.769  5.770  -5.912 -5.960 -1.995  -7.676 -7.719 -3.907
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.07 ==> Err2_1=-0.07 ->Err1_1= 0.06 Err1_2= 0.00
201    3.782  3.786  5.791  -5.919 -5.967 -2.018  -7.715 -7.758 -3.900
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.04 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.04 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
202    3.795  3.800  5.817  -5.927 -5.974 -2.003  -7.752 -7.756 -3.943
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
  | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
  | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
  | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
203    3.812  3.817  5.838  -5.934 -5.981 -2.027  -7.790 -7.795 -3.934
  | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
```

```
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  204    3.825  3.830  5.863  -5.941 -5.987 -2.013  -7.825 -7.794 -3.976
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  205    3.841  3.846  5.883  -5.948 -5.994 -2.037  -7.862 -7.832 -3.967
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  206    3.854  3.859  5.907  -5.955 -6.001 -2.024  -7.896 -7.831 -4.007
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.86 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  207    3.905  3.909  5.856  -5.955 -6.001 -2.060  -7.942 -7.871 -3.909
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.04 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.94 ==> Err2_1= 0.06 ->Err1_1=-0.05 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.88 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.
  208    3.859  3.863  5.948  -5.963 -6.008 -2.044  -7.928 -7.869 -4.016
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  209    3.911  3.916  5.896  -5.963 -6.008 -2.044  -7.974 -7.869 -3.962
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.05 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.05 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.87 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.
  210    3.869  3.874  5.979  -5.970 -6.015 -2.030  -7.962 -7.867 -4.061
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  211    3.924  3.928  5.924  -5.970 -6.015 -2.067  -8.009 -7.908 -3.960
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.04 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.05 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.05 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.87 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.
  212    3.882  3.887  6.007  -5.978 -6.023 -2.053  -7.997 -7.906 -4.058
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  213    3.935  3.940  5.954  -5.978 -6.023 -2.053  -8.043 -7.906 -4.005
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.87 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.
  214    3.897  3.901  6.031  -5.985 -6.029 -2.039  -8.032 -7.904 -4.098
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.88 ->Y2_1= 0.05 ==> Err2_1=-0.05 ->Err1_1= 0.00 Err1_2= 0.04
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.96 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.96 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  215    3.950  3.955  5.977  -5.985 -6.029 -2.076  -8.078 -7.945 -3.998
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.04 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> Err2_1= 0.05 ->Err1_1=-0.04 Err1_2=-0.01
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.87 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.
  216    3.911  3.916  6.055  -5.992 -6.036 -2.062  -8.067 -7.943 -4.092
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.11 Y1_2= 0.02 ->Y2_1= 0.96 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.10 Y1_2= 0.02 ->Y2_1= 0.96 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.85 Y1_2= 0.00 ->Y2_1= 0.06 ==> Err2_1=-0.06 ->Err1_1= 0.06 Err1_2= 0.00
  217    3.963  3.967  6.003  -5.992 -6.036 -2.062  -8.111 -7.943 -4.040
    | Example1:X1=0 X2=0 T=0 Output:Y1_1= 0.00 Y1_2= 0.89 ->Y2_1= 0.05 ==> It behaves correctly.
    | Example2:X1=0 X2=1 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example3:X1=1 X2=0 T=1 Output:Y1_1= 0.12 Y1_2= 0.02 ->Y2_1= 0.95 ==> It behaves correctly.
    | Example4:X1=1 X2=1 T=0 Output:Y1_1= 0.87 Y1_2= 0.00 ->Y2_1= 0.05 ==> It behaves correctly.

 *** We found a required multilayer network. ***
xcspc60_44%
```