

# NeuralWorks® User-Defined Neuro-Dynamics v5.5

Create custom-designed neural network architectures in *NeuralWorks Professional II/PLUS*.

*User-Defined Neuro-Dynamics* (UDND) is an advanced optional extension of *NeuralWorks Professional II/PLUS* that allows you to use the C programming language to define your own learning rules, transfer functions, error functions, and other aspects of processing in a neural network. These functions are compiled and linked with libraries to produce a customized super-set of *NeuralWorks Professional II/PLUS*.

## Fast, easy design

Using UDND you can modify *NeuralWorks Professional II/PLUS* learning rules, have the new network up and running in as little as 30 minutes, and know that it will work. Creating a completely new network type takes longer, but you will discover it is easier and quicker than other methods of network design. Equally as important, creating a new learning rule or network this way makes it possible to quickly and effectively compare it to other supported network types. This is an essential advantage only available with UDND.

## Customize these functions

In addition to the broad range of basic building blocks provided by NeuralWare, UDND makes it possible to customize the following functions:

Summation	Learn
Transfer	Noise
Output	Checkpoint processing
Error	Special event processing

When you have finished with your customization, UDND links the new functionality to create a custom *NeuralWorks Professional II/PLUS*. Properly constructed functions are also compatible with *NeuralWorks Designer Pack*.

## Only write the essentials

The UDND advantage is that it lets you modify only the building blocks particular to your application. In many instances, it takes only a few lines of code to construct a new paradigm. Through well defined and strictly followed interface specifications, all of the other building blocks fit seamlessly together into the new system. This also reduces debugging time, because the only routines which need to be tested are the new building blocks.

All of the capabilities of *NeuralWorks Professional II/PLUS*, including network editing tools, general analysis

tools, and utilities are available with the new functionality.

## How User-Defined Neuro-Dynamics fits in

NeuralWare has pioneered the building-block approach to neural network implementation. This structure uses four fundamental building blocks: Topology, Neuro-Dynamics, Control Strategy, and Learning/Recall Parameters.

**Topology.** The topology of a network is an object oriented description of each of the layers, processing elements, and connections. These form a hierarchical object-oriented database in which processing /elements inherit certain characteristics from the layer to which they belong. Connections likewise inherit certain characteristics from the processing elements they connect. Each processing element has an input terminal as well as an output terminal. Connections are always from the output terminal of one processing element to the input terminal of another. Connection topology is completely arbitrary. NeuralWare provides a script language, InstaNet, which is used to construct most common configurations.

**Neuro-Dynamics.** A variety of functions act on processing elements and their connections. These functions perform such operations as updating the internal activation of a processing element, modifying the connections into a processing element, selecting a “winner” among processing elements in a layer and so on. The Neuro-Dynamics are the core computational routines in any neural network paradigm. *User-Defined Neuro-Dynamics* makes it possible for you to add your own custom operation to these tables of functions. This modular objectoriented design makes it easy to change a single building block, such as a learning rule, creating a new network type, without rewriting all of the other building blocks. This approach also reduces the time required to debug a new algorithm.

**Control Strategy.** The Control Strategy determines the flow of control required to implement a neural network. It provides capabilities for performing input-output operations, activating certain kinds of check points, and selectively activating the Neuro-Dynamic functions associated with each layer. As described above, the Neuro-Dynamic functions act



NEURALWARE

on information in a processing element, and through the connections into a processing element. A control strategy is the *flow-chart* which is implemented to perform the algorithmic functions of a neural network.

Control Strategies may be very general and tolerant of network topology, such as the one for Back-Propagation. Other network types, such as Fuzzy ARTMAP, require very specific control strategies which require a very specific Network Topology. Control Strategies are implemented in an assembly-language-like syntax which is converted from source code to internal code at runtime.

**Learning / Recall Parameters.** Many of the Neuro-Dynamic functions require parameters which may vary over time. Learning / Recall Parameters are tables of parameters, the meaning of which is specific to the particular Neuro-Dynamics of a layer. These parameter tables may be specific to a single layer, group of layers, or the entire network. Annealing schedules, learning schedules, and a host of other steady-state and time-dependent functions can be implemented.

## Full Customization

*NeuralWorks Professional II/PLUS* used alone provides the ability to customize Network Topology and Control Strategies. Limited customization of a particular network is possible by using non-standard building blocks for error functions, learning rules, etc. User-Defined Neuro-Dynamics provides the final step in full customization - creation of new building blocks.

## UDND Functions

A UDND function is written in C code. Though inherited from the layer, it acts on one processing element at a time. Each time a UDND function is called, it is provided a pointer to the current processing element, and the layer to which that processing element belongs. From the layer pointer, it is possible to obtain a pointer to the Learning and Recall Parameters. Within each processing element, there are several fields which define the current state of the processing element, as well as a handle to the connection list.

Access to the connection list provides the essential capabilities for implementing learning rules. Though this is the only time when the connection list is typically updated, under special circumstances, other functions may modify the connection list as well. The learning rule specified in the layer is used to determine the number of fields in the connection list. When building a network, it is essential to set the learning rule in the layer prior to making any connections to processing elements in the layer. This usually requires modification of the appropriate InstaNet script.

Functions which only modify fields internal to the processing element may not require any changes to the Control Strategy or InstaNet script used to construct the network. When UDND is used to implement a radically new paradigm, several new functions may need to be created, as well as a new Control Strategy and InstaNet script.

As an aid to implementing new functions, a variety of existing Neuro-Dynamics functions are provided. For the majority of applications, these can be modified to create the desired enhancements.

## Other Features

**Event Handling.** UDND allows you to write an event handler to intercept a variety of events such as: Network Loaded, Network Saved, Learn Start, Learn Complete, Recall Start, Recall Complete. This class of event handling is useful for dynamically allocating or freeing memory for special purposes. It can also be used to open or close files for logging information.

**Check-Points.** The check point option allows the ability to perform specific actions when certain iteration-driven events occur. One of the most popular uses is to periodically jog or limit the magnitude of weights during training.

**Initialization.** Whenever the Network Initialize function is called, the user-defined initialization routine is called for each processing element which contains a user-defined routine. This provides the capability to initialize local variables.

**Graphical Analysis Tools.** The graphical analysis capabilities of *NeuralWorks Professional II/PLUS* work seamlessly with properly constructed UDND

routines. The analysis package is not specific to any one paradigm but is general in nature. It can be used to probe, dynamically display, and record any combination of processing element and connection fields, applying a variety of transformations to them.

**Integrated User IO.** High-performance User IO routines can be linked directly into *NeuralWorks Professional II/PLUS* with UDND. This makes it possible to build high-performance systems with customized input-output interfaces.

## Requirements

*NeuralWorks Professional II/PLUS* version 5.5  
NeuralWorks User-Defined NeuroDynamics version 5.5. (This includes *NeuralWorks Professional II/PLUS* libraries, source code sample modules, and example make or batch files.)

C Compiler, linker and libraries for each platform. NeuralWare supports the most current compilers for the platforms we support. Examples of this are Microsoft Visual C for Windows, and most GNU C Compilers. This is not an all-inclusive list. Contact NeuralWare for specific computers.

To find how User-Defined Neuro-Dynamics can provide a solution that allows you to define your own learning rules, transfer functions, error functions, and other aspects of processing in a neural networks contact NeuralWare at 412-278-6288, FAX us at 412-278-6289 or email us at [sales@neuralware.com](mailto:sales@neuralware.com).

## For More Information

To find out how *NeuralWorks UDND* can provide a solution for your neural network design and development, contact NeuralWare at (412) 278-6288, FAX us at (412) 278-6289 or EMAIL us at [sales@neuralware.com](mailto:sales@neuralware.com).



**NeuralWare**  
409 Elk Street  
Suite 200  
Carnegie PA 15106  
USA  
Phone (412) 278-6288  
FAX (412) 278-6289  
Email [sales@neuralware.com](mailto:sales@neuralware.com)  
<http://www.neuralware.com>