

NeuroCarb

Artificial Neural Networks for NMR Structure Elucidation of Oligosaccharides

Inauguraldissertation

Zur Erlangung der Würde eines Doktors der Philosophie
vorgelegt der Philosophisch-Naturwissenschaftlichen Fakultät
der Universität Basel

von

Matthias Studer-Imwinkelried
aus Liestal BL und Langnau BE

Basel, 2006

Genehmigt von der Philosophisch-Naturwissenschaftlichen Fakultät
auf Antrag von

Prof. Dr. Beat Ernst, Institut für Molekulare Pharmazie, Universität Basel
Prof. Dr. Johann Gasteiger, Computer-Chemie-Zentrum und Institut für Organische Chemie,
Universität Erlangen-Nürnberg

Basel, den 5. Juli 2005

Prof. Dr. Hans-Jakob Wirz
Dekan

1. Summary	9
2. Abbreviations	11
3. Introduction	12
3.1. Glycoproteins	12
3.1.1. Glycoprotein structures and biosynthesis	13
3.1.2. Recombinant proteins	18
3.1.3. Main objectives of glycoprotein analysis	20
3.2. Carbohydrate structure elucidation by nuclear magnetic resonance (NMR)	21
3.2.1. Number of sugar residues	22
3.2.2. Constituent monosaccharides	22
3.2.3. Anomeric configuration	23
3.2.4. Linkage and sequence	24
3.2.5. Position of appended groups	24
3.2.6. Advantages and disadvantages of NMR	25
3.3. Artificial neural networks (ANN)	26
3.3.1. Short historical overview	26
3.3.2. Concise introduction to neural networks	27
3.3.3. Training of artificial neural networks	34
3.3.4. Learning in neural networks	34
3.3.5. Learning rules	36
3.3.6. Modifying patterns of connectivity	37
3.3.7. Advantages and disadvantages of neural networks	37
3.3.8. Application of neural networks	38
3.3.9. Application of neural networks to NMR and carbohydrates	39
3.3.10. Other computer-assisted structural analysis systems for carbohydrates	39
3.4. Integration of NeuroCarb into the EuroCarbDB	40
3.4.1. What is EuroCarbDB	40
3.5. The aims of this PhD thesis	42
4. Material and Methods	43
4.1. Used chemical compounds	43
4.1.1. Methyl pyranosides	43
4.1.2. Hindsgaul compounds	44
4.1.3. Disaccharide test compounds	45
4.1.4. Synthesis of β -D-glucopyranosyl-1-6- β -D-glucopyranosyl-1-6- β -D-glucopyranoside	46
4.1.5. ^{13}C -NMR Database	49
4.2. NMR equipment & experiments	52
4.3. Computer hardware	53
4.4. IUPAC JCAMP-DX	53
4.4.1. Summary	53
4.4.2. Detail insight into a JCAMP-DX file	53
4.4.3. The internal file format	54
4.4.4. Important LDRs for regaining the original NMR data (in ppm)	57
4.5. Multi-Layer Perceptrons (MLP) and the Back-propagation learning method	59
4.5.1. Problems of the Back-propagation learning method	61
4.5.2. Training with the Back-propagation learning method	62
4.5.3. Self organizing feature maps (SOM)	64
4.5.4. Counter-propagation Network	69
4.6. Error functions	71

4.6.1.	The Sum-of-squares error (SSE)	71
4.6.2.	Mean squared error (MSE)	71
4.6.3.	Cross entropy	71
4.7.	Modification generator (MG)	72
4.8.	Used neural network simulation software	72
4.8.1.	Statsoft Statistica ^[293]	72
4.8.2.	Stuttgart Neural Network Simulator (SNNS) V.4.2 ^[297]	73
4.8.3.	Java Neural Network Simulator (JavaNNS) V.1.1 ^[298]	75
4.9.	ANN PFG (Pattern File Generator)	76
4.9.1.	Introduction / Summary	76
4.9.2.	Input file formats	79
4.9.3.	Output file formats	80
4.9.4.	SNNS PFG V.0.1	81
4.9.5.	SNNS PFG V.0.2	83
4.9.6.	ANN PFG V.0.9	88
5.	Experiments	98
5.1.	Glycosylation shifts	98
5.1.1.	α -D-Glcp-OMe-xR	98
5.1.2.	β -D-Glcp-OMe-xR	99
5.1.3.	α -D-Glcp-xR	99
5.1.4.	β -D-Glcp-xR	100
5.1.5.	α -D-Manp-xR	100
5.1.6.	β -D-Manp-xR	101
5.1.7.	α -D-Manp-OMe-xR	101
5.1.8.	β -D-Manp-OMe-xR	102
5.1.9.	α -D-Galp-xR	102
5.1.10.	β -D-Galp-xR	103
5.1.11.	α -D-Galp-OMe-xR	103
5.1.12.	β -D-Galp-OMe-xR	104
5.2.	General definitions	105
5.3.	Methyl pyranosides approach	105
5.3.1.	¹ H-NMR data	105
5.3.2.	Conclusion	113
5.4.	¹³ C-NMR experiments	113
5.4.1.	Used dataset	113
5.4.2.	Comparison of different Back-propagation learning algorithms	115
5.4.3.	Comparison of different learning rates	116
5.4.4.	Comparison of different learning rates at 600 hidden units	117
5.4.5.	Hidden layer size comparison with additional noise	118
5.4.6.	Hidden layer size comparison without additional noise and block-pattern	119
5.4.7.	Classification comparison of different initial weight initialization values	120
5.4.8.	MSE comparison with different initial weight initialization values	122
5.4.9.	Hidden layer size comparison at learning rate 0.2	123
5.4.10.	Hidden layer size comparison at learning rate 0.7 and shift \pm 3 Hz	124
5.4.11.	Learning rate comparison without hidden layer and binary input patterns	126
5.4.12.	Conclusion	127
5.5.	Diploma work Alexej Moor	128
5.5.1.	Introduction	128
5.5.2.	Dataset	128
5.5.3.	Experiments & Results	129
5.5.4.	Discussion & conclusions	133
5.6.	Introduction of FileMaker ¹³ C-NMR database	133
5.7.	Kohonen feature maps	133

5.7.1.	Decay factor	133
5.7.2.	Data preparation	134
5.7.3.	Galactose	135
5.7.4.	Glucose	139
5.7.5.	Mannose	146
5.7.6.	Combination of galactose, glucose and mannose	150
5.7.7.	Discussion	151
5.8.	Statistica Approach	152
5.8.1.	Experiment-nomenclature	153
5.8.2.	Definitions	153
5.8.3.	Pattern file structure	154
5.8.4.	Data set	155
5.8.5.	Test files	156
5.8.6.	Preliminary experiments with Statsoft Statistica	160
5.8.7.	Glucose	168
5.8.8.	Galactose	170
5.8.9.	Mannose	172
5.8.10.	Combination of glucose, galactose and mannose (GAM)	174
5.9.	Ensemble approach	176
5.9.1.	The concept	176
5.9.2.	Glucose ensemble networks with one and two hidden layers	177
5.9.3.	Galactose ensemble networks with one and two hidden layers	180
5.9.4.	Mannose ensemble networks with one and two hidden layers	184
5.9.5.	Discussion of the ensemble approach	187
6.	Discussion summary & conclusions	189
7.	Outlook	193
8.	References	195
9.	Figure index	203
10.	Acknowledgements	207
11.	Appendix	209
11.1.	Peak lists of disaccharide test compounds	209
11.1.1.	Trehalose	209
11.1.2.	Gentiobiose	209
11.1.3.	Lactose	209
11.1.4.	Saccharose	210
11.2.	Regula Stingelin compounds	210
11.2.1.	β -D-pGlc-OMe	210
11.2.2.	β -D-pGlc-1-6- β -D-pGlc-OMe	211
11.3.	Monosaccharide test files	211
11.3.1.	Glucose	211
11.3.2.	Galactose	214
11.3.3.	Mannose	216
11.4.	GAM disaccharide test file	217

1. Summary

Recombinant proteins and monoclonal antibodies offer great promise as therapeutics for hundreds of diseases. Today, there are almost 400 biotechnology drugs in development for over 200 different conditions. Many of these drugs are glycoproteins for which the correct glycosylation patterns are important for their structure and function. Achieving and maintaining proper glycosylation is a major challenge in biotechnology manufacturing. Most recombinant therapeutic glycoproteins are produced in living cells. This method is used in an attempt to correctly match the glycosylation patterns found in the natural human form of the protein and achieve optimal in vivo functionality. However, utilizing cell systems to produce glycoproteins requires balancing the cells ability to produce the protein with its ability to attach the appropriate carbohydrates. One limitation of this approach is that the expression systems do not maintain complete glycosylation under high-volume production conditions. This results in low yields of usable product and contributes to the cost and complexity of producing these drugs. Incorrect glycosylation also affects the half-life of the drug. Low production yields are a significant contributor to the critical worldwide shortage of biotechnology manufacturing capacity.

To achieve higher production yields, the required quality standards to fulfill regulations by health authorities, fast, accurate and preferably inexpensive analytical methods are required. Nowadays the (routine) analysis of therapeutic glycoprotein is accomplished by analytical HPLC, MS or Lectin blotting and in conjunction with chemical derivatization, exo-glycosidases treatment, and/or other selective chemical cleavage reactions. The fact that different carbohydrates have very similar molecular weights and physicochemical properties makes the analysis of glycosylation slow and complex. Conventional glycoanalysis requires multiple steps to obtain the structure, sequence and prevalence of all glycans in a glycoprotein sample. Complete analysis typically takes several days and highly trained personnel. Therefore, the need for more efficient and rapid glycoanalysis methodology is fundamental to the success of biotechnologically produced drugs.

With this demand in the back of one's mind, a ^{13}C -NMR spectra analysis system for oligosaccharides based on multiple Back-propagation neural networks was developed during this thesis. Before the realization of the idea, some fundamental questions had to be posed:

1. Are the monosaccharide moieties, the anomeric configuration and the substitution pattern of an oligosaccharide shown in a NMR (^{13}C or ^1H) spectrum?
2. What kind of NMR data provides this information better (^1H or ^{13}C -NMR)?
3. How can spectroscopic data be processed, compressed and transferred into a neural network?
4. Which neural network architecture, learning algorithm and learning parameters lead to optimal results?

Preliminary experiments showed that the six chemical shifts of a monosaccharide moiety (from glucose, galactose and mannose) suffice to identify the monosaccharide itself, the anomeric configuration (if the anomeric carbon atom is substituted) and the substitution position(s). The experiments also revealed that these compounds could be almost completely separated by the help of Counter-propagation neural networks.

The main goal of the neural network approach was to recognize every single monosaccharide moiety in an oligosaccharide and train specialized separated networks for each monosaccharide moiety group. Therefore, the neural networks should be trained with the ^{13}C -NMR spectra of these monosaccharide moieties. During the test phase, the whole spectrum of an oligosaccharide will be presented to the network and the specialized networks should then only recognize the monosaccharide moieties they are trained for.

Initial attempts to train a Back-propagation neural network to identify six methyl pyranoside compounds failed. This lack of success was because the data set used was too small and an uncompressed NMR spectrum leads to too many input neurons. Therefore, the data foundation was changed and enlarged with 535 monosaccharide moieties (mostly galactose, glucose and mannose) from literature and a special data compression (JCAMP-DX for NMR files) and parsing software tool called ANN Pattern File Generator was developed. The entire dataset was normalized and stored in a FileMaker ^{13}C -NMR database. Further experiments with this new dataset, different Back-propagation network layouts and training parameters still did not achieve the designated recognition rate of unknown test compounds. The training performance of the neural networks seems to be insensible against major changes of training parameters. Tests with a new and enlarged dataset (1000 oligosaccharides and approx. 2500 monosaccharide moieties) with Kohonen networks highlighted, that separate Kohonen networks for each monosaccharide type yield to higher recognition rates than networks, which have to deal with all three monosaccharide types at once.

This cognition was transferred to separate back propagation networks, which now showed recognition rates higher than 90% for unknown compounds. This separated approach worked excellent for disaccharides with two different monosaccharide moieties. Disaccharides with similar or identical moieties cannot be identified because the designated neural network recognizes only one monosaccharide at once. Out of this disadvantage, the so-called '*ensemble*' or '*group of experts*' approach was developed. Here, one utilizes the fact, that no trained neural network shows exactly the same recognition characteristics. Different neural networks respond differently to the same test inputs. Twenty trained neural networks at a time were grouped into ensembles. All these networks are trained to recognize the same monosaccharide moiety. After presenting a test input (e.g. disaccharide) to this group of experts, one gets at the most extreme case, twenty different recognition results. Afterwards, the results can be statistically analyzed. In the case of a disaccharide with two monosaccharide moieties of the same carbohydrate (e.g. $\alpha\text{-D-Glcp-1-4-}\beta\text{-D-Glcp-OMe}$), the analysis will deliver both monosaccharide compounds because some networks recognized one and other networks the other part of the disaccharide.

The ensemble approach brought the final breakthrough of this thesis. Disaccharide recognition rates in the range of 85 – 96% (depending on the monosaccharide moiety – glucose, galactose or mannose) demonstrate the feasibility of the approach. The hit rates of the different ensembles can certainly be improved by a more subtle choice of the members of each ensemble. An ongoing diploma work shows a recognition improvement in this direction.

2. Abbreviations

Act	Activation Function
AFFN	ASCII Free Form Numeric
ANN	Artificial Neural Network
CASPER	Computer assisted spectrum evaluation of regular polysaccharides
COSY	Correlation spectroscopy
CSV	Comma-separated values
CHO	Chinese hamster ovary cells
DEPT	Distortionless Enhancement by Polarization Transfer
DQF-COSY	Double quantum filtered-COSY
ER	Endoplasmatic reticulum
FID	Free-induced decay
GAM	G lucose, G alactose and M annose
GUI	Graphical user interface
HMBC	Heteronuclear multiple bond correlation
HMQC	Heteronuclear multiple quantum coherence
HPLC	High pressure liquid chromatography
HSQC	Heteronuclear single quantum coherence
HU	Hidden units (neurons)
IPS	Intelligent problem solver (part of the Statsoft Statistica program)
IU	Input units (neurons)
IUPAC	The International Union of Pure and Applied Chemistry
JCAMP	Joint Committee on Atomic and Molecular Physical Data
LDR	Labeled data records (in JCAMP-DX files)
LINUXS	Linear Notation for Unique description of Carbohydrate Sequences
MALDI	Matrix-assisted laser desorption/ionization
MG	Modification Generator
MLP	Multi-layer perceptron (Neural network whit one or more hidden layers)
MS	Microsoft
MSE	Mean square error
NOE	Nuclear Overhauser Effect
NOESY	Nuclear Overhauser enhancement spectroscopy
ODBC	Open Database Connectivity, a standard database access method developed by the SQL Access group
OU	Output units (neurons)
PFG	Pattern File Generator (ANN PFG)
ROESY	Rotating frame Overhauser enhancement spectroscopy
SNNS	Stuttgart Neural Network Simulator
SOM	Self organizing feature maps – also called Kohonen feature maps
SQL	Structured query language. SQL is a standardized query language for requesting information from a database
TOCSY	Total Correlation Spectroscopy – a high resolution NMR technique
VBA	Visual Basic for Applications

3. Introduction

3.1. Glycoproteins

The human genome contains approx. 30'000 genes and encodes up to 40,000 proteins. A major challenge is to understand how post-translational events, such as glycosylation, affect the activities and functions of these proteins in health and disease. Glycosylated proteins are ubiquitous components of extracellular matrices and cellular surfaces where their oligosaccharide moieties are implicated in a wide range of cell-cell and cell-matrix recognition events. Most viruses and bacteria use cell-surface carbohydrates to gain entry into cells and initiate infection. Several human diseases and tumor metastasis are related to abnormalities in carbohydrate degradation and recognition. As a result, interest in glycobiology and characterization of carbohydrates has grown rapidly. However, the technology for carbohydrate analysis and sequencing has lagged behind this recent demand. One reason for this could be the distinct heterogeneity of oligosaccharide structures frequently found on a single polypeptide species. Hence, a single protein may exist as a complex collection of glycoproteins, which differ only in the amount or structure of attached carbohydrate moieties.

Unlike other structural biomolecules such as proteins and nucleic acids, synthesis of which is template-driven and well defined at a molecular level, oligosaccharides are not primary gene products ^[1].

For glycoproteins intended for therapeutic administration, it is important to have knowledge about the structure of the carbohydrate side chains. This will provide strategies to avoid cell systems that produce structures, which in humans can cause undesired reactions, e.g., immunologic and unfavorable serum clearance rate. Structural analysis of the oligosaccharide part of the glycoprotein requires instruments such as MS and/or NMR. However, before the structural analysis can be conducted, the carbohydrate chains have to be released from the protein and purified to homogeneity, which is often the most time-consuming step. Mass spectrometry and NMR play important roles in analysis of protein glycosylation. For oligosaccharides or glycoconjugates, the structural information from mass spectrometry is essentially limited to monosaccharide sequence, molecular weight, and only in exceptional cases glycosidic linkage positions can be obtained. To completely elucidate an oligosaccharide structure, several other structural parameters have to be determined, e.g., linkage positions, anomeric configuration and identification of the monosaccharide building blocks. One way to address these problems is to apply NMR spectroscopy (chapter 3.2).

Recombinant proteins and monoclonal antibodies offer great promise as therapeutics for many diseases. In 2002 there were more than 371 biotechnology drugs in development for nearly 200 different diseases ^[2]. Many of these drugs are glycoproteins. The process by which these carbohydrates are attached to proteins is called glycosylation. Glycosylation patterns are important to the structure and function of glycoproteins. Achieving and maintaining proper glycosylation is a major challenge in biotechnology manufacturing, and one that affects the industry's overall ability to maximize the clinical and commercial gains possible with these agents. Most recombinant therapeutic glycoproteins, including the well-known drugs AvonexTM (interferon beta 1- α) and EprexTM/EprexTM (epoetin α), are produced in living cells - Chinese hamster ovary (CHO) cells - in

an attempt to correctly match the glycosylation patterns found in the human form of the protein and achieve optimal *in vivo* functionality.

However, utilizing cell systems to produce glycoproteins requires balancing the cells ability to produce the protein with their ability to attach the appropriate carbohydrates. CHO cells engineered to produce large quantities of a specific protein often do not maintain the proper level of glycosylation. This results in low yields of usable product, which contributes to the cost and complexity of producing these drugs. Incorrect glycosylation also affects the immunogenicity^[3], plasma half-life, bioactivity and stability^[4] of a potential therapeutic product, resulting in the need to administer higher and more frequent doses.

Table 1: Some examples of the effect of glycosylation on therapeutic activity reported in the literature.

Protein	Change	Effect
erythropoietin	additional glycans; increased sialylation	longer half life; 5-fold reduction in dosing
follicle stimulating hormone	correct glycosylation	increased half-life
cerzyme/ceredase	increased exposure of mannose	better binding to mannose receptors; increased cell uptake to site of action
monoclonal antibodies	terminal galactose	mediation of effector function

These complications affect the cost of therapy, and potentially, the incidence of side effects. Low yields are a significant contributor to the critical worldwide shortage of biotechnology manufacturing capacity. Thus, the ability to manufacture these drugs is becoming an important strategic asset of pharmaceutical and biotechnology companies. Because of these issues, the pharmaceutical industry continues to search for better ways to manufacture and analyze glycoproteins. Alternative expression systems, such as transgenic animals and plants, have received industry and media attention because they offer the possibility to significantly increase product yields at lower cost. However, achieving the correct glycosylation patterns remains a problem with these systems and is a significant barrier to their widespread adaptation for manufacturing proteins for parenteral use.^[5]

3.1.1. Glycoprotein structures and biosynthesis

The structural variability of glycans is dictated by tissue specific regulation of glycosyltransferase genes, acceptor and sugar nucleotide availability in the Golgi, compartmentalization, and by competition between enzymes for acceptor intermediates during glycan elongation. Glycosyltransferases catalyze the transfer of a monosaccharide from specific sugar nucleotide donors onto a particular hydroxyl position of a monosaccharide in a growing glycan chain with a specific anomeric linkage (either α or β). The protein microenvironment of the immature glycan chain also affects glycosyltransferase catalytic efficiency, and leads to structural heterogeneity of glycans between glycoproteins - even between different glycosylation sites on individual glycoproteins produced by the same cells^[6]. The Oligosaccharide structures depend on the cell type and its enzymatic equipment, its developmental stage, and its nutritional or pathological state^[7]. The true structural diversity is enormous. This raises the question of using recombinant glycoproteins for therapeutic purposes, insofar as the oligosaccharide chains of the produced

glycoproteins have to be structurally close to those of the wild-type glycoproteins and compatible with the immune system

Oligosaccharides are covalently linked to proteins through *O*- (to Ser or Thr) or *N*- (to Asn) glycosidic bonds, respectively^[8]. In *O*-glycosylated proteins, the oligosaccharides range in size from 1 to 20 sugars. Therefore, they are displaying considerable structural (and antigenic) diversity. Moreover, these oligosaccharides are uniformly distributed along the peptide chain, or clustered in heavily glycosylated domains. *N*-Acetylgalactosamine (GalNAc) is invariably linked to Ser or Thr (Figure 1). Mannose residues have not been detected in mature *O*-glycans.

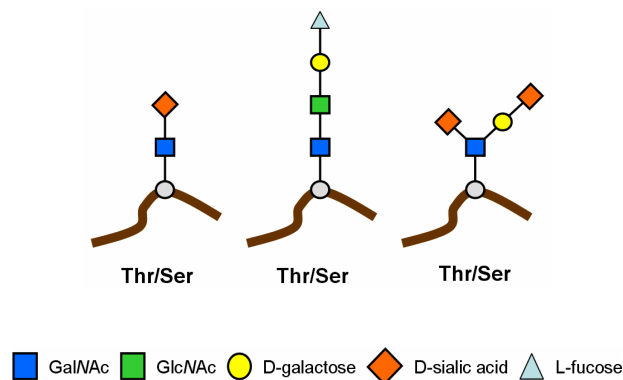


Figure 1: *O*-linked oligosaccharides

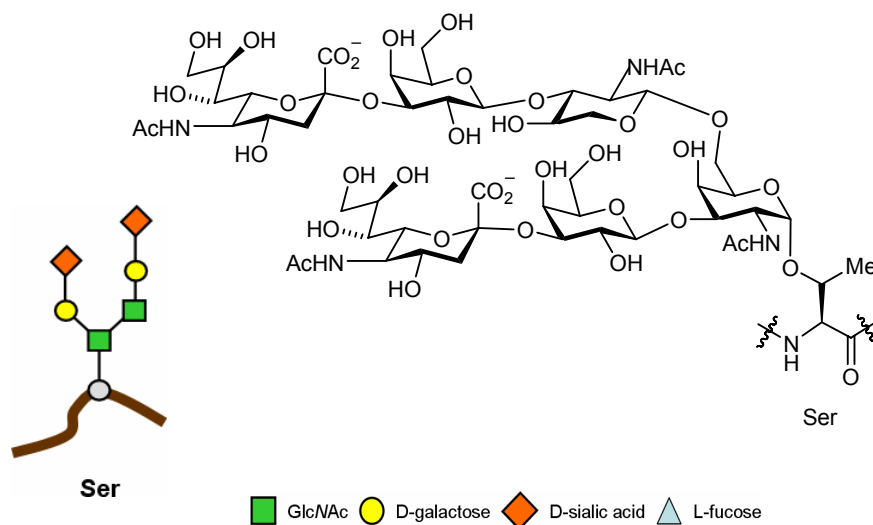


Figure 2: *O*-linked oligosaccharide in schematic illustration (left part) and the corresponding chemical structure (right)

N-Oligosaccharides have a common core structure of five sugars and differ in their outer branches. The first sugar residue, *N*-acetylglucosamine (GlcNAc) is bound to Asn being part of a specific tri-peptide sequence (Asn-X-Thr or Asn-X-Ser). *N*-Oligosaccharides are classified into three main categories: high mannose, complex, and hybrid (). High-mannose oligosaccharides have two to six additional mannoses linked to the pentasaccharide core and are forming branches. Hybrid oligosaccharides contain one branch that has the complex structure and one or more high-mannose branches. Complex-type oligosaccharides have two or more branches, each containing at least one GlcNAc, one Gal, and eventually a sialic acid (SA).

These branches can be bi-, tri-, or tetra-antennary (Figure 3). Glc residues have not been detected in mature complex *N*-oligosaccharides. Serum glycoproteins mostly consist of complex type *N*-oligosaccharides. *O*- and *N*-oligosaccharide chains may occur on the same peptide core [7].

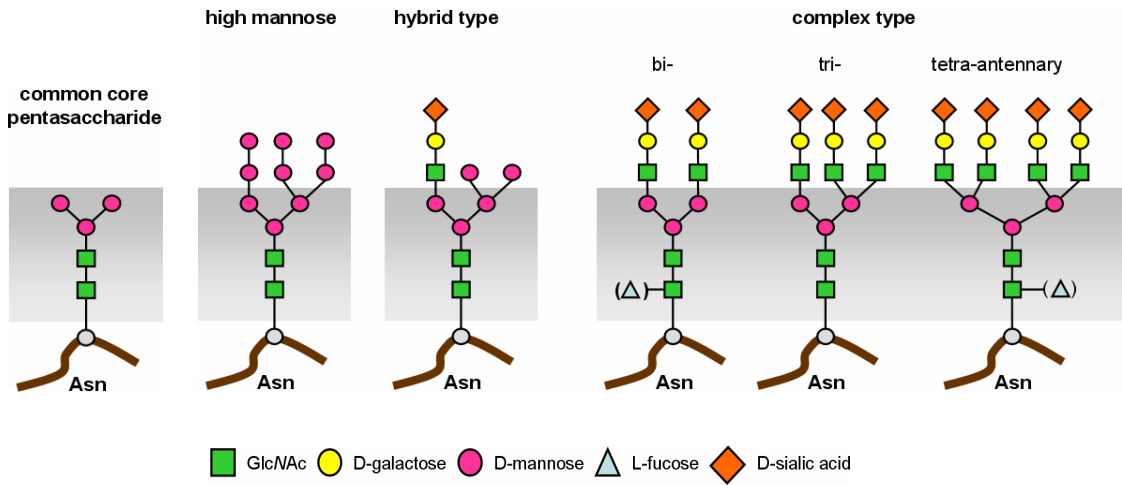


Figure 3: *N*-linked oligosaccharides

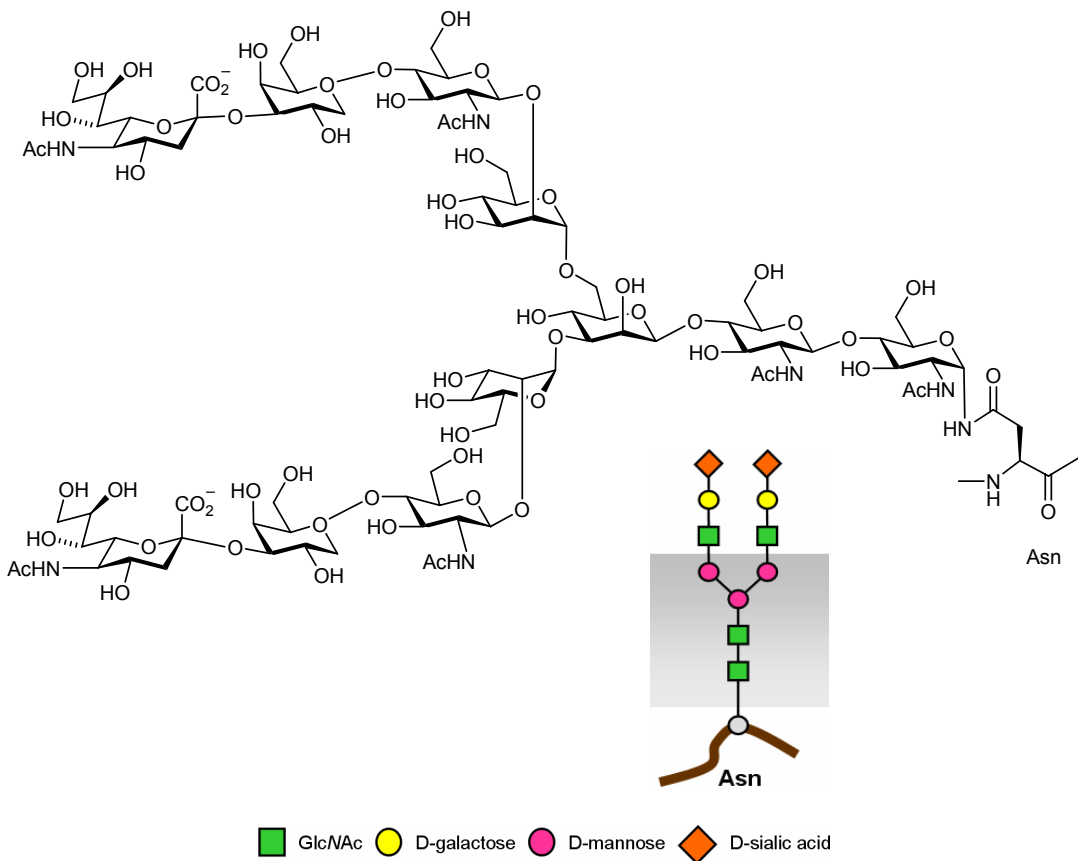


Figure 4: *N*-linked oligosaccharide in schematic illustration (bottom right) and the corresponding chemical structure (top)

O-Oligosaccharide biosynthesis begins in the *cis* Golgi with the transfer of the first sugar residue, GalNAc, from UPD-GalNAc by a specific polypeptide, O-GalNAc transferase, to a completed polypeptide chain. The glycan chain then grows by the addition of GlcNAc, Gal, and Fuc residues in the medial Golgi. Sialylation finally takes place throughout the trans Golgi. There are several possible pathways to construct O-glycans, depending on the substrate specificity and intracellular arrangement of glycosyltransferases. However, it is far less complex than the processing of N-oligosaccharides^[7].

The biosynthesis of N-oligosaccharides (Figure 5) begins in the ER with a large precursor oligosaccharide that contains 14 sugar residues. The inner five residues constitute the core, which is conserved in all structures of N-linked oligosaccharides (highlighted in figure 5). This precursor is linked to dolichol pyrophosphate, which acts as a carrier for the oligosaccharide.

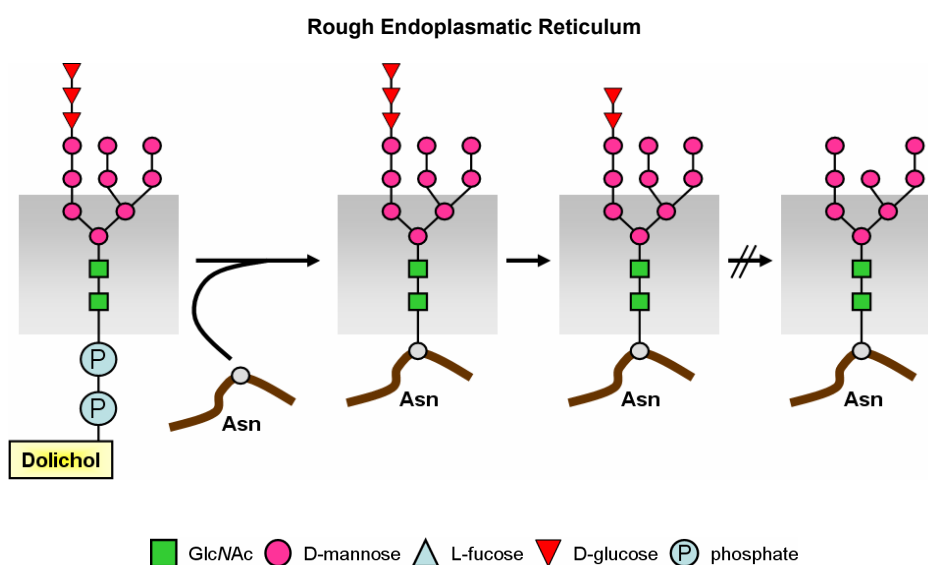


Figure 5: Processing of N-linked complex oligosaccharides (I)

In a next step, the lipid-linked oligosaccharide is transferred “en bloc” to an Asn residue on the growing polypeptide chain. While the nascent glycoprotein is still in the rough ER, all three GlcNAc residues and one mannose residue are removed by specific glycosidases, producing an oligosaccharide with 10 residues instead of 14. The subsequent maturation of the N-oligosaccharides takes place in the Golgi complex.

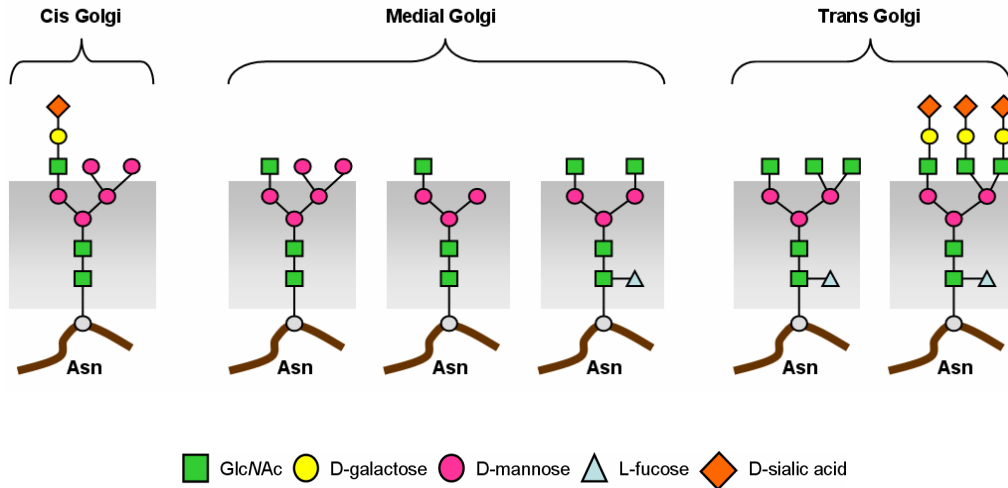


Figure 6: Processing of N-linked complex oligosaccharides (II)

This pathway involves a coordinated and sequential set of enzymatic reactions, which remove and add specific sugar residues. The enzymes involved (glycosidases and glycosyltransferases) are located in the cis, medial, and trans Golgi (figure 6). Many of these enzymes are extremely sensitive to stimuli within the cell, in which the glycoprotein is expressed. As a result, the specific sugars attached to an individual protein depend on the cell type in which the glycoprotein is expressed and its physiological status. The reaction product of one enzyme is the substrate for the next. When present, sialic acid residues are always at the terminal non-reducing ends of oligosaccharides. Missing terminal sialic acids on a glycoprotein expose underlying galactose residues, which are a signal for hepatic removal of the glycoprotein from circulation. The high-mannose and hybrid oligosaccharides appear as intermediates along the processing pathway.

The carbohydrate components of glycoproteins affect the functionality of the molecule by determining protein folding, oligomer assembly and secretion processes. Without the proper shape, the ability of the protein to interact correctly with its receptor is affected, possibly affecting function. Glycosylation may have additional biological roles by affecting solubility and preventing aggregation and metabolism.

3.1.2. Recombinant proteins

Recombinant proteins and monoclonal antibodies require a host organism for expression. Although protein expression systems produce correct amino acid sequences, the glycosylation remains (if unmodified) that of the host (Figure 7).

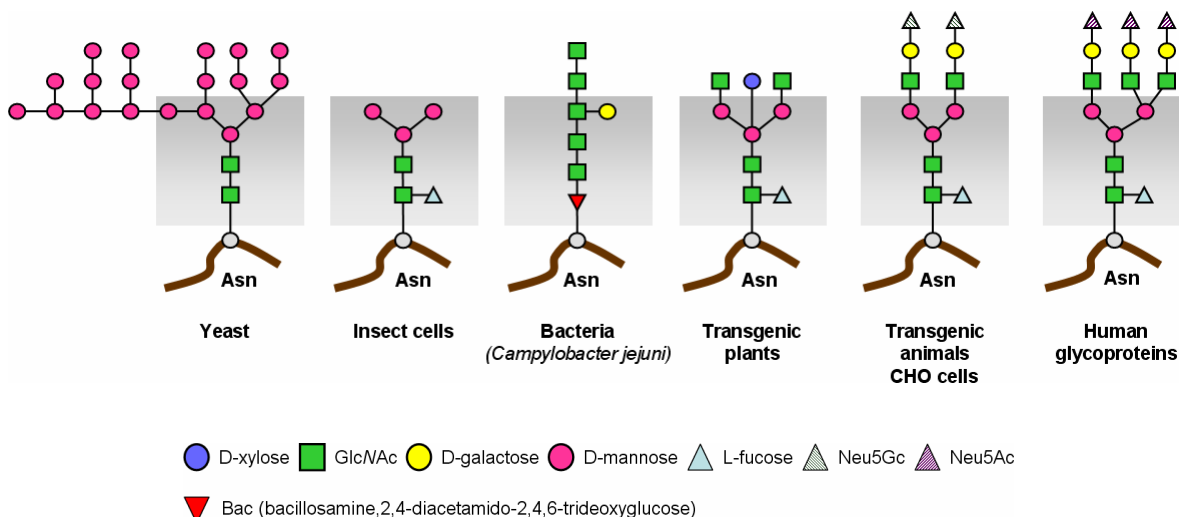


Figure 7: Comparison of *N*-glycosylation among alternate expression systems
Table 2: Comparison of expression systems ^[9]

Table 3: Different selected expression systems

Characteristics	Bacteria	Yeast	Insect cells	Mammalian cells
Cell growth	rapid (30 min)	rapid (90 min)	slow (18-24 h)	slow (24 h)
Complexity of growth medium	minimum	minimum	complex	complex
Cost of growth medium	low	low	high	high
Expression level	high	low - high	low - high	low - moderate
Extracellular expression	secretion to periplasm	secretion to medium	secretion to medium	secretion to medium
Posttranslational modifications	no eukaryotic post-translational modifications	most of the eukaryotic post-translational modifications	many of the post-translational modifications performed in mammalian cells	post-translational modifications
Protein folding	refolding usually required	refolding may be required	proper folding	proper folding
<i>N</i> -linked glycosylation	<i>Campylobacter jejuni</i> and many other bacteria have been identified as containing both <i>N</i> - and <i>O</i> -linked glycosylation systems	high mannose	simple, no sialic acid	complex
<i>O</i> -linked glycosylation		yes	yes	yes

Bacteria: The established paradigm that bacteria do not glycosylated proteins is no longer valid ^[10-13]. The human enteropathogenic bacterium *Campylobacter jejuni* and many other bacteria have been identified as containing both *N*- and *O*-linked glycosylation systems. But the details of the glycosylation biosynthetic process have not been determined in any of the bacteria systems ^[11].

Yeast: Researchers have shown that yeast (*pichia pastoris*) expression system can be genetically altered to produce therapeutic glycoproteins with human-like oligosaccharide structures ^[14]. This process involves the knockout of some of the endogenous glycosylation pathways, and recreation of the human sequential glycosylation machinery, which requires proper localization of active glycosyltransferases and mannosidases. Yeast and fungal expression systems offer a simple and cost effective production process with high yield and powerful secretory pathways.

Insect cell lines like the baculovirus/lepidopteran expression system ^[15, 16] attach shorter mannose chains to the parent protein than yeast ^[17] and cannot produce sialylated complex *N*-glycans. Again, while not likely immunogenic, these foreign patterns affect the properties of the recombinant proteins.

Plants: The published studies on the production of human proteins in plants indicate that plants often add simple *N*-glycan structures that lack galactose and terminal sialic acids. As a consequence their affinity is compromised.

CHO cells, the system most commonly used today for recombinant protein manufacturing, glycosylate close to human but do not maintain complete glycosylation under production conditions.

Transgenic animals are being studied as an alternative to traditional CHO cell production processes. Transgenic animals provide a potentially less expensive source of production for proteins compared to traditional cell culture systems. In recent years, the number of production systems has increased. While transgenic expression systems may solve the problems of protein production yields and may lower cost, they do not solve the problem of protein glycosylation. Another obstacle may be the presence of α 1-3 linked core fucose residues that are potentially immunogenic ^{[3] [18]}.

A potential concern is that most transgenic systems link a non-human form of sialic acid, *N*-glycolylneuraminic acid. Whether or not this is a problem may become evident as high-dose, chronic-use protein therapeutics become more widely used. A review of interferon gamma, a recombinant protein that has been expressed in three different systems, offers insight into the types of glycosylation differences that occur among expression systems. Interferon gamma produced in CHO cells contains a fucose residue and high mannose oligosaccharide chains. Finally, Interferon γ produced in transgenic mice shows considerable site-specific variation in *N*-glycan structures. Interferon γ produced from insect cell culture is associated with tri-mannosyl core structures. These differences highlight the importance of monitoring glycosylation patterns and noting the effect of variances in glycosylation on the structure and function of the recombinant protein ^[5].

To achieve these required quality standards and fulfill regulations by health authorities, fast, accurate and preferably inexpensive analytical methods are required. Nowadays the (routine) analysis of therapeutic glycoprotein is accomplished by analytical HPLC, MS or Lectin blotting and in conjunction with chemical derivatization, exo-glycosidases treatment, and/or other selective chemical cleavage reactions.

The complexity described above plus the fact that different carbohydrates have very similar molecular weights and physicochemical properties, makes the analysis of glycosylation slow and complex. Conventional glycoanalysis requires multiple steps to obtain the structure, sequence and prevalence of all glycans in a glycoprotein sample:

1. purification of the protein from culture medium
2. the chemical or enzymatic release of the glycans from the protein backbone
3. purification of the glycans
4. separation, labeling or other modification of the glycans
5. sequential cleavage of the terminal carbohydrates for some analytical methods
6. MS or NMR analysis

Complete analysis typically takes several days and highly trained personnel. This series of procedures and methods has several disadvantages:

1. Several of the steps can introduce anomalies that interfere with accurate analysis of the carbohydrates and the structure of the glycans
2. Once the glycans have been separated from the protein, it is not possible to determine the relationship of the glycans.

There is therefore clearly a need for more efficient and rapid glycol-analysis methodology.

3.1.3. Main objectives of glycoprotein analysis

Glycoprotein analysis is used in the following working fields

- clone profiling, selection and scale up in drug discovery
- monitoring of glycosylation changes during drug metabolism and pharmacokinetics in development
- stability analysis of glycosylation patterns during stability testing
- growth optimization and monitoring to reduce batch loss, save time and improve quality control in manufacturing

3.2. Carbohydrate structure elucidation by nuclear magnetic resonance (NMR)

There are several approaches to perform a primary structural analysis of a mono-, oligo-, or polysaccharide by NMR spectroscopy. Vliegenthart *et al.* [19] introduced the *structural-reporter-group* concept, which is based on signals outside the bulk region (3-4 ppm) in the ^1H -NMR spectra of carbohydrates. This approach is used to identify individual sugars or sequences of residues and can be used to identify structural motifs or specific sugars and linkage compositions found in relevant databases.

NMR- based structure elucidation is most often combined with data from mass spectrometry or chemical information, e.g. monosaccharide composition or methylation analysis [20]. Methylation analysis [21] provides information about which hydroxyl groups are substituted. Oligosaccharides were investigated in H_2O at temperatures below 0 °C, either by super cooling or addition of acetone- d_6 to prevent freezing [22]. During the studies the authors noticed [22] that the method can be used to identify positions in the monosaccharide residues of oligosaccharides which are glycosidically linked. The aliphatic protons at carbons with OH attached will show couplings to the OH group at low temperature and can be identified by comparison of spectra obtained in D_2O and H_2O using 1D TOCSY or by line broadening. The remaining aliphatic protons, often with sharper signals, will then correspond to substituted positions of the glycosidic linkages [23]. This method requires only small amounts of material compared to the amounts required for a full NMR structural analysis. If this indirect method fails to identify the glycosidic positions due to overlap, the positions bearing OH can be identified in a 2D COSY [24] by the correlation between OH protons and aliphatic protons. Similar experiments can be carried out in DMSO, where the exchange of OH-protons is slow even at room temperature [25].

Carbohydrates normally have at least two NMR-active nuclei, ^{13}C and ^1H . In addition, less frequently used nuclei like ^2H , ^{15}N , ^{17}O and ^{31}P can be used for studies of natural or synthetic oligosaccharides. The dispersion of resonances in the carbon spectra is favorable, but the amount of material needed to acquire such spectra is relatively high due to the low natural abundance of ^{13}C . However, advances in both hardware and pulse sequences have reduced the amount needed. In practical terms, about 100 μg of a pure trisaccharide is sufficient to perform a complete structural assignment by both ^1H and ^{13}C -NMR spectroscopy. When comparing chemical shift values and entering the data into a neural network, it is important that the reference data is measured at the same temperature and that the data are based on the same internal reference.

In the following chapters, the different NMR techniques to obtain the carbohydrate properties are discussed briefly.

3.2.1. Number of sugar residues

A good starting point for a structural analysis is the chemical shift of the anomeric proton. Integration of the anomeric resonances offers an initial estimate on the number of different monosaccharide residues present. The anomeric proton resonances are found in the shift range 4.4 - 5.5 ppm. The remaining ring proton resonances are found in the range 3 - 4.2 ppm in unprotected oligosaccharides. Additionally, the number of anomeric C₁ resonances present in a 1D ¹³C-NMR spectrum will confirm the number of different residues. (Such results can also be obtained from 2D ¹³C-¹H HSQC [26-28], HMQC [29-31] or HMBC [32-35] spectra, which in most cases are more sensitive than a 1D ¹³C spectrum).

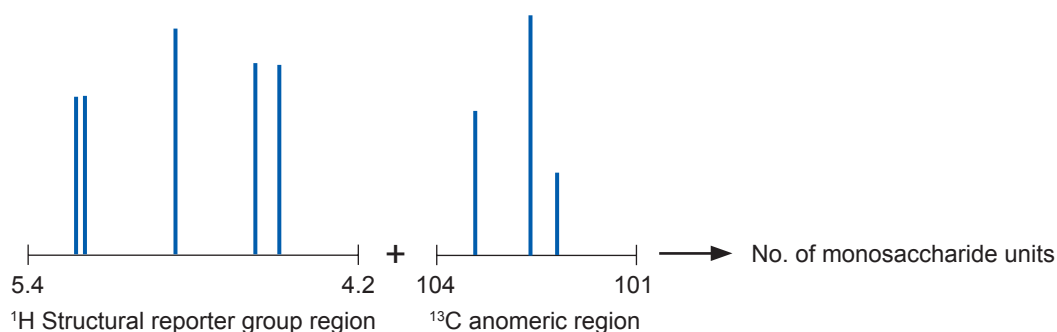


Figure 8: determination of the number of involved monosaccharide units (adapted from [25])

Illustrated examples used during this thesis are discussed in greater detail in chapter 5.1.

3.2.2. Constituent monosaccharides

Homonuclear TOCSY and DQF-COSY spectra are useful in the identification of individual monosaccharide residues. In TOCSY spectra of oligosaccharides acquired with a fairly long mixing time (>100 ms), it is often possible to measure the size of the coupling constants and the correlations to reveal the identity of the residue. In cases with significant overlap in the bulk region (3-4.2 ppm), a 1D selective TOCSY [36] may be useful in resolving ambiguities. Both ¹H and ¹³C chemical shifts for most monosaccharides are reported in literature (chapter 4.1.5) [25]. Based on such values, an assignment of the individual residues can be achieved with the help of neural networks. The ¹³C chemical shift values can easily be obtained from a HSQC or HMQC spectrum [29-31]. For carbohydrates without an anomeric proton (Figure 9 and Figure 10), characteristic signals as the H3equatorial or H3axial protons ($\delta_{\text{H3axial}} \sim 1.9$ ppm and $\delta_{\text{H3axial}} \sim 2.3$ ppm [37]) are a good starting point for the assignments.

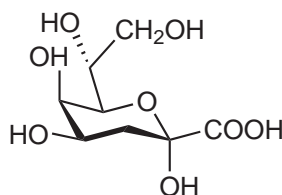
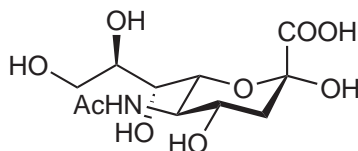


Figure 9: α -Kdo = 3-deoxy-D-manno-octulosonic acid

Figure 10: α -NeuAc

These experiments summarized in figure 11 are useful and give additional dispersion in the carbon dimension, which may facilitate the assignment of individual spin systems.

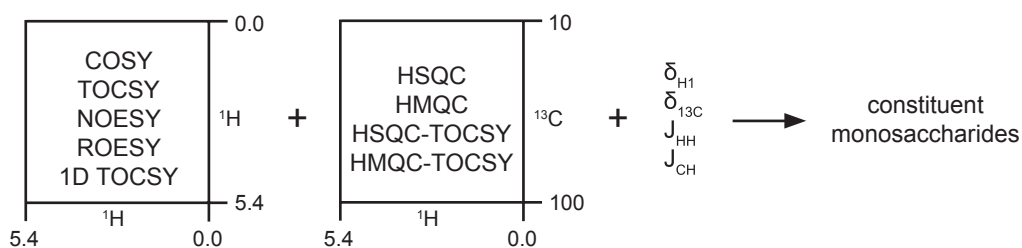


Figure 11: determination of the constituent monosaccharides (adapted from [25])

3.2.3. Anomeric configuration

Normally the α -anomer resonates downfield compared to the β -anomer in D-pyranoses in 4C1 conformation. The vicinal coupling constant between the anomeric H1 and the H2 indicates the relative orientation of the two protons. If they are both in an axial configuration in pyranose structures, a large coupling constant (7-8 Hz) is observed, whereas if they are equatorial-axial, this is smaller ($J_{1,2} \sim 4$ Hz), and for equatorial-equatorial oriented protons, even smaller coupling constants are observed (< 2 Hz) [38]. This principle can be used when assigning the relative orientation of protons in a hexopyranose ring as first demonstrated by Lemieux *et al* [39]. The ^{13}C chemical shift reveals the anomeric configuration in a manner similar to the proton chemical shifts, but most importantly the one bond ^{13}C - ^1H coupling constants in pyranoses can be used to determine the anomeric configuration unequivocally. For D sugars in the 4C1 conformation, a $^1J_{\text{C1},\text{H1}} \sim 170$ Hz indicates an α -anomeric sugar configuration whereas $^1J_{\text{C1},\text{H1}} \sim 160$ Hz indicates a β -anomeric sugar configuration [40]. This is reversed for L sugars. The use of one-bond coupling constants in furanose structures does not correlate in the same way with the anomeric structure. Several experiments can be used to measure these one-bond coupling constants, the simplest is to turn off the proton decoupling during the carbon acquisition.

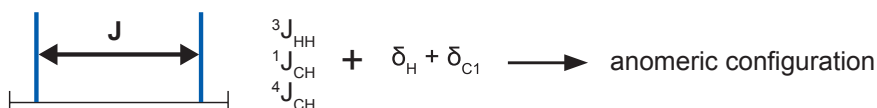


Figure 12: determination of the anomeric configuration (adapted from [25])

Illustrated examples used during this thesis are discussed in detail in chapter 5.1.

3.2.4. Linkage and sequence

Both the ^1H and the ^{13}C chemical shift may give an indication of the linkage type, if the chemical shifts for the specific linkage have been reported previously [25]. The effect of glycosylation depends on the linkage type, and the changes in the chemical shift are in general larger at the glycosylation site than at neighboring positions. Interresidue NOEs may give information about the glycosidic linkage, but it should be kept in mind that the strongest NOE might not be between the protons across the glycosidic linkage [41] [42]. A HMBC [32-35] experiment can also give linkage information, keeping in mind that both intra- and interresidue correlations are seen.

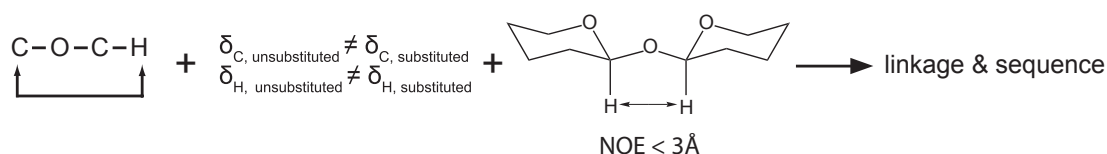


Figure 13: determination of linkage and sequence (adapted from [25])

Illustrated examples used during this thesis are discussed in detail in chapter 5.1.

3.2.5. Position of appended groups

The proton and carbon chemical shifts are sensitive to the attachment of a non-carbohydrate group like a methyl, acetyl, sulfate, or a phosphate group. Attachment of such groups will affect the proton and carbon resonances at the substitution position. Normally downfield shifts ~ 0.2 - 0.5 ppm are observed [25] for protons and higher $\Delta\delta$ values for ^{13}C . This shifts these resonances in a less crowded area of the spectra and helps the identification of modified residues. Such appended groups may also contain NMR-active nuclei, which may give rise to additional splitting due to couplings (e.g., ^{31}P - ^1H long-range couplings). The use of other homo- or hetero-nuclear correlations may help in the determination of their position. As pointed out above, many of the resonances are found in a narrow chemical shift range, and this can make it problematic to distinguish resonances which are close in chemical shift. Difficulty also arises when comparing different spectra or spectral regions.

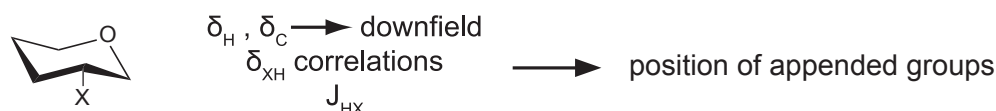


Figure 14: determination of the position of appended groups (adapted from [25])

Illustrated examples used during this thesis are discussed in detail in chapter 5.1.

3.2.6. Advantages and disadvantages of NMR

Because of the very large number of possible structural isomers ^[43], no structural elucidation technique is capable of providing a complete structural analysis, although nuclear magnetic resonance comes close in many cases. Unfortunately, NMR is very insensitive and normally needs relatively large sample amounts. However, with new special nano NMR sample tubes ^[25] and spectrometers with cryo heads, it is possible to reduce the amount of compound down to some milligrams. Even more complicated is the application of NMR analysis of a whole glycoprotein as a trustworthy routine monitoring method during production of therapeutic glycoproteins. Conventional glycoprofiling methods are complex, time consuming and therefore cost-intensive.

Recent trends in science have resulted in an explosive growth in the number of biotechnological medicines in development. These are largely driven by the rapidly growing number of known drug opportunities emerging from genomics and the improved ability to clone and express human proteins. Such developments are a major force in the growth of the pharmaceutical and biotech industries. However, expansion in this area is limited by manufacturing production capacities. Too much valuable material is rejected because of incorrect or missing glycosylation patterns provoked by slow analysis methods. These manufacturing limitations are likely to slow the growth of the biotech industry that could be realized if these issues were solved. Industry analysts have estimated that for every \$100 million of demand for a drug that goes unfilled, \$1 billion of the drug's market value is destroyed ^[44]


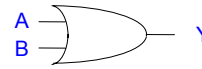
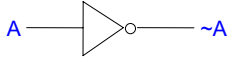
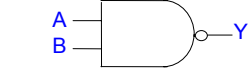

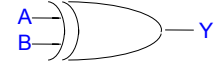
Therefore, new rapid, inexpensive and accurate analytical approaches such as the ANN approach proposed in this PhD thesis would be highly beneficial.

3.3. Artificial neural networks (ANN)

3.3.1. Short historical overview

The history of neural networks is almost as old as the first programmable computers and proceeds the history of the symbolic AI (artificial intelligence). In 1943, Warren McCulloch and Walter Pitts rudimentary characterized neural networks. They demonstrated that these networks could in principal compute every arithmetic or logic function ^[45].

Table 4: Basic logical functions and gates

<p>AND</p>  <p>$A \cdot B$</p> <table border="1" data-bbox="260 801 509 987"> <thead> <tr> <th>Input 1 \ Input 2</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>0</td> <td>0</td> </tr> <tr> <th>1</th> <td>0</td> <td>1</td> </tr> </tbody> </table> <p>produces a 'true' result whenever there is 'true' on both inputs</p>	Input 1 \ Input 2	0	1	0	0	0	1	0	1	<p>OR</p>  <p>$A \vee B$</p> <table border="1" data-bbox="614 801 863 987"> <thead> <tr> <th>Input 1 \ Input 2</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>0</td> <td>1</td> </tr> <tr> <th>1</th> <td>1</td> <td>1</td> </tr> </tbody> </table> <p>produces a 'true' result when there is a 'true' on either or both inputs</p>	Input 1 \ Input 2	0	1	0	0	1	1	1	1	<p>NOT</p>  <p>$\sim A$</p> <table border="1" data-bbox="949 835 1198 925"> <thead> <tr> <th>Input</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <td>Output</td> <td>1</td> <td>0</td> </tr> </tbody> </table> <p>Whatever logical state is applied to the input, the inverted state will appear at the output</p>	Input	0	1	Output	1	0			
Input 1 \ Input 2	0	1																											
0	0	0																											
1	0	1																											
Input 1 \ Input 2	0	1																											
0	0	1																											
1	1	1																											
Input	0	1																											
Output	1	0																											
<p>NAND</p>  <p>$\overline{A \cdot B}$</p> <table border="1" data-bbox="260 1238 509 1424"> <thead> <tr> <th>Input 1 \ Input 2</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>1</td> <td>0</td> </tr> <tr> <th>1</th> <td>0</td> <td>0</td> </tr> </tbody> </table> <p>When there are two false inputs, one gets a true result</p>	Input 1 \ Input 2	0	1	0	1	0	1	0	0	<p>NOR</p>  <p>$\overline{A \vee B}$</p> <table border="1" data-bbox="614 1238 863 1424"> <thead> <tr> <th>Input 1 \ Input 2</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>1</td> <td>1</td> </tr> <tr> <th>1</th> <td>1</td> <td>0</td> </tr> </tbody> </table> <p>When there is a 'false' input on one or both inputs, there is 'true' as the result</p>	Input 1 \ Input 2	0	1	0	1	1	1	1	0	<p>XOR</p>  <p>$A \oplus B$</p> <table border="1" data-bbox="949 1238 1198 1424"> <thead> <tr> <th>Input 1 \ Input 2</th> <th>0</th> <th>1</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>0</td> <td>1</td> </tr> <tr> <th>1</th> <td>1</td> <td>0</td> </tr> </tbody> </table> <p>Whenever there is a 'false' on one input, and a 'true' on the other input, a 'true' result is generated</p>	Input 1 \ Input 2	0	1	0	0	1	1	1	0
Input 1 \ Input 2	0	1																											
0	1	0																											
1	0	0																											
Input 1 \ Input 2	0	1																											
0	1	1																											
1	1	0																											
Input 1 \ Input 2	0	1																											
0	0	1																											
1	1	0																											

Independently, Donald O. Hebb described with the classical Hebbian learning ^[46] rule how neural assemblies can self-organize into feedback circuits capable of recognizing patterns (chapter 3.3.6). This rule can be found in its general form in almost every neural learning process. In the following years, the first successful applications of neural networks were demonstrated. Shortly after Frank Rosenblatt ^[47] constructed the first effective neuro-computer (Mark I Perceptron).

In 1969, Marvin Minsky and Seymour Papert ^[48] performed a detailed mathematical analysis of the Perceptron and showed deficiencies of the Perceptron model. They forecasted that the area of neural networks is a 'research dead-end'. In the following 15 years of little acknowledgement some scientists, famous today, laid the basis for the renaissance:

In 1972, Teuvo Kohonen ^[49] introduced a model of a linear associator. Paul Werbos proposed in 1974 in his PhD thesis ^[50, 51] the world's famous Back-propagation learning rule. However, his work attained great importance only approximately ten years later by the work of Rumelhart and McClelland ^[52]. Well-known names like Stephen Grossberg ^[53-55], John Hopfield ^[56-59] and Fukushima ^[60-74] followed in the next years. In the eighties, a period of main growth expansion followed. Often the influence of John Hopfield is quoted for the revival of the neural networks. He proved ^[58] that neural networks are able to solve the traveling salesman problem.¹

This result convinced many scientists of the potential benefits of ANN. Great influence had the final development and enhancement of the Back-propagation learning rule by Rumelhart, Hinton and Williams ^[52].

3.3.2. Concise introduction to neural networks

Artificial neural networks are an attempt at modeling the information processing of the nervous systems. Animal nervous systems are composed of thousands or millions of interconnected neurons. Each is a very complex arrangement, which deals with incoming signals in many different ways. However, neurons are rather slow when compared to their electronic analogues. Whereas the electronic simulation can achieve switching times of a few nanoseconds, biological neurons need several milliseconds to react to a stimulus. To accelerate this rather slow process, massively parallel and hierarchical networking of the brain is a prerequisite for its immense performance ^[75].

Table 5: Comparison between brain and computer ^[76]

Comparison between brain and computer

	brain	computer
number of processing elements	approx. 10^{11} neurons	approx. 10^9 transistors
Kind	Massively parallel	mainly serial
Storage	associative	referring to address
switching time of one element	approx. 1 ms (10^{-3} s)	approx. 1 ns (10^{-9} s)
"switching events" [Hz]	approx. 10^3 [Hz]	approx. 10^9 [Hz]
"switching events" altogether (theoretical)	approx. 10^{13} [Hz]	approx. 10^{18} [Hz]
"switching events" altogether (real)	approx. 10^{12} [Hz]	approx. 10^{10} [Hz]

¹ *Traveling salesman problem* (= TSP): Given a set of towns and the distances between them, determine the shortest path starting from a given town, passing through all the other towns and returning to the first town.

This is one of the most famous problems to test computationally different approaches (e.g. genetic algorithms, particle swarms, neural networks etc.). It has a variety of solutions of varying complexity and efficiency. The simplest solution (the brute force approach) generates all possible routes and takes the shortest. This becomes impractical as the number of towns, N, increases since the number of possible routes is $!(N-1)$. At this stage, only highly differentiated algorithms will succeed. Especially neural networks and particle swarms perform significantly better than other complex algorithms. Algorithms to solve the TPS problem are also used by phone companies to route telephone calls through their wire and wireless networks.

Today, the mechanisms for the production and transport of signals from a neuron to the next neuron are well-understood physiological phenomena. However, the mechanism by which these systems cooperate to form complex and extreme-parallel systems capable of incredible information processing feats has not yet been completely elucidated.

Biological neural networks are just one of many possible solutions to the problem of processing information. The main difference between neural networks and conventional computer systems is the massive parallelism and redundancy, which they exploit in order to deal with the unreliability of the individual computing units. Moreover, biological neural networks are self-organizing systems and each individual neuron is a delicate self-organizing structure capable of processing information in many different ways.

In biological neural networks, information is stored at the cell body. Nervous systems possess global architectures of variable complexity, but all are composed of neural cells or neurons.

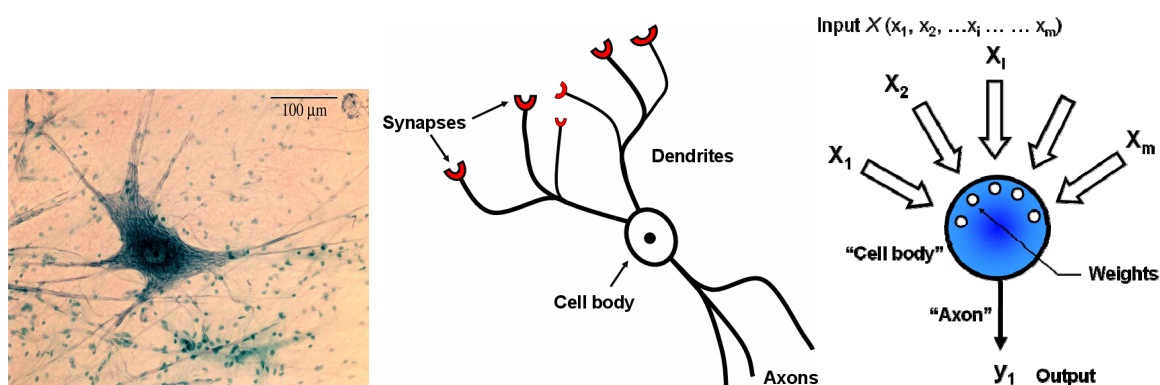


Figure 15: microscopic image of a biological neuron and Comparison between the biological and artificial neuron. The circle mimicking the neuron's cell body represents simple mathematical procedures to generate an output signal y_j from the set input signals represented by the multivariate input vector X (adapted from J. Zupan and J. Gasteiger)

Dendrites are the transmission channels for incoming information. They receive the signals at the contact regions (the synapses) with other nerve cells. The output signals are transmitted by the axon, of which each cell has mostly several. The elements of the biological system, dendrites, synapse, cell body and axon, are the minimal structure, which are adopted by the ANN from the biological model. Artificial neurons for computing have input channels, a cell body and an output channel. The synapses will be simulated by their so-called weights².

² The *weight* is the synaptic strength who determines the relative amount of the signal that enters the body of the neuron through the dendrites. In neural networks the term weight describes the factor by which the input is multiplied (Equation 1). Attenuating weights have values < 1 and amplifying weight need values > 1 .

Figure 16 shows the structure of an abstract neuron with four inputs (x) and four weights (w).

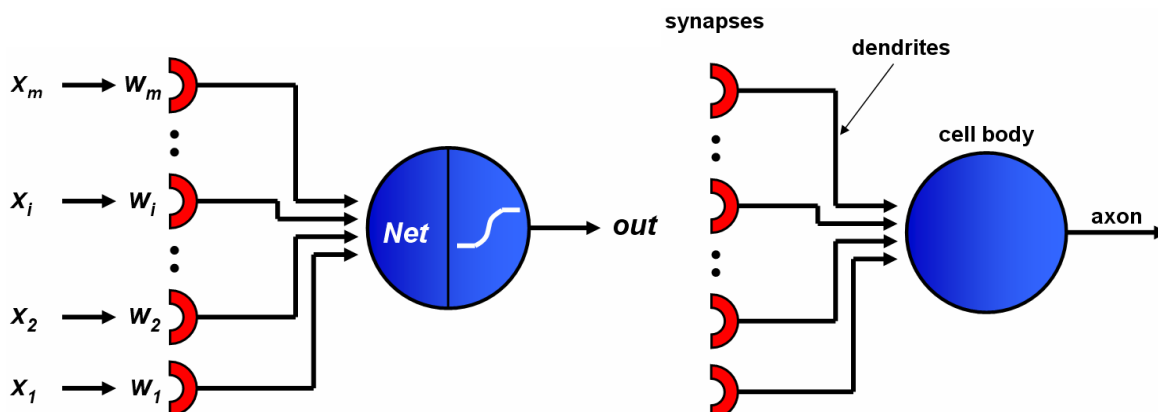


Figure 16: Similarities between biological and artificial neurons (adapted from J. Zupan and J. Gasteiger)

Each neuron normally has a large number of dendrites or synapses. Therefore, many signals can be received by the neuron simultaneously. The individual signals are labeled x_i and the corresponding weights, w_i .

The sum of the incoming signals becomes the net input **Net**: (Equation 1)

$$Net = w_1x_1 + w_2x_2 + \dots + w_ix_i + \dots + w_mx_m \quad \text{Equation 1}$$

The input signals are combined into a multivariate signal: a multidimensional vector \mathbf{X} , whose components are the individual input signals:

$$\mathbf{X} = (x_1, x_2, \dots, x_i, \dots, x_m) \quad \text{Equation 2}$$

The same way, all the weights can be described by a multidimensional weight vector \mathbf{W} :

$$\mathbf{W} = (w_1, w_2, \dots, w_i, \dots, w_m) \quad \text{Equation 3}$$

The **Net** is then the scalar product of a weight vector \mathbf{W} and a multivariate input vector \mathbf{X} representing an arbitrary object:

$$Net = \mathbf{W}\mathbf{X} + \vartheta = w_1x_1 + w_2x_2 + \dots + w_ix_i + \dots + w_mx_m + \vartheta \quad \text{Equation 4}$$

$$Net = \sum_{i=1}^m w_ix_i + \vartheta \quad \text{Equation 5}$$

In the present model, a neuron contains two steps in obtaining output from the incoming signals. In the first step the net input **Net** (as explained above) is evaluated and in the second step the net input signals **Net** is transformed nonlinearly. The second step tries to imitate the reaction of a real biological neuron. It only fires if the excitatory potential is reached, otherwise there is no stimulus passed ^[77].

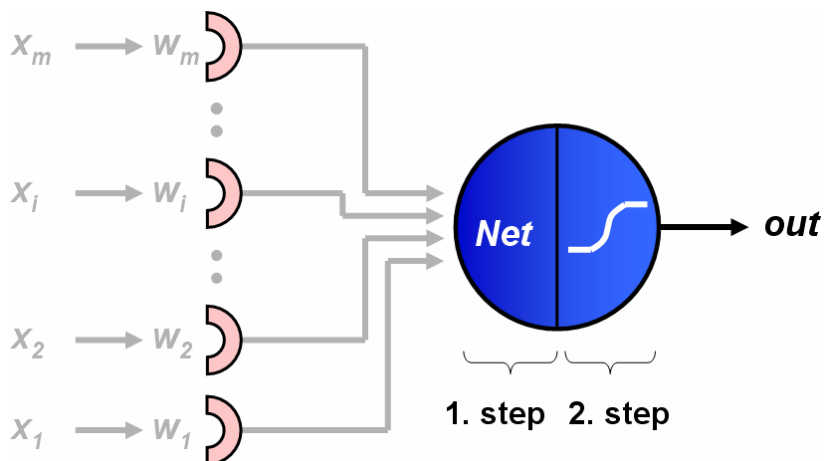


Figure 17: The first (evaluation of the *Net* input) and the second step (nonlinear transformation of *Net*) taking place in the artificial neuron

$$out = f(Net)$$

Equation 6

The transfer function is also called squashing function because it squashes the output into a small interval. Some frequently used transfer functions for the second step are represented in the following figures:

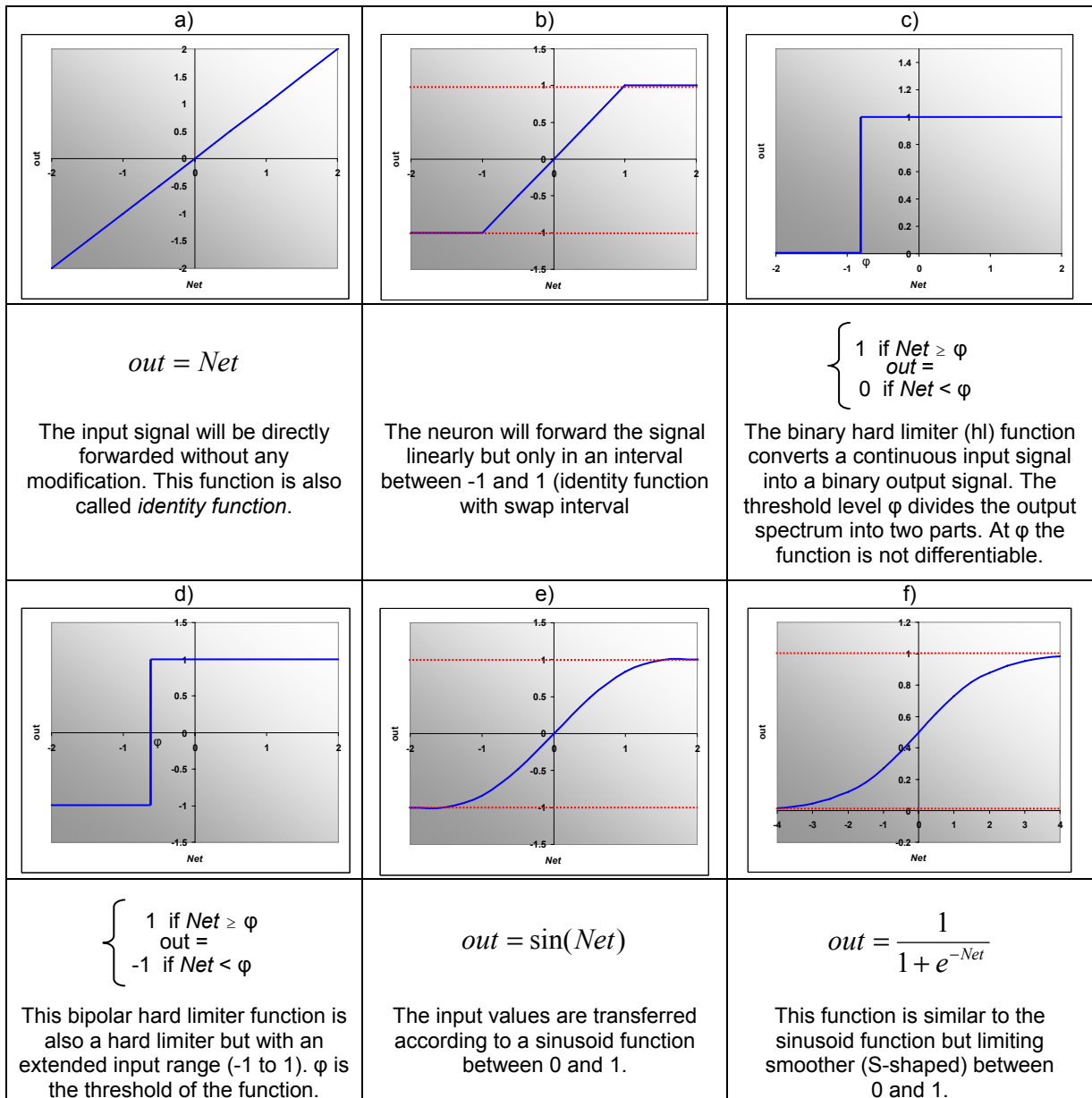


Figure 18: Transfer functions

The basic operation of a neuron is always the same. It collects a net input **Net** and transforms it into the output signal via one of the transfer of functions (Figure 18).

A layer is a group of neurons all of which have the same number of weights and all receive the same dimensional input signal simultaneously. The input "layer" does not change the input signals. That means that the input neurons have neither weights nor any kind of transfer function. These non-active input units (=input neurons) serve only as distributors of signals and do not play an active role in the network.

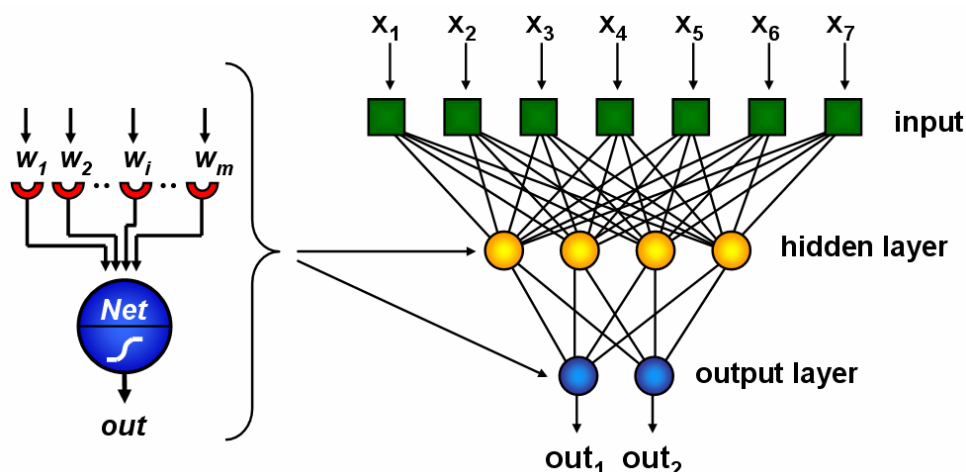


Figure 19: Full-connected feed forward sample network with one hidden layer

The layer(s) below the passive input layer are called *hidden layer(s)*, because they are not directly connected to the input or output signal. They only serve the information processing. More complex neural networks normally consist of more than one hidden layer. This is especially the case for higher dimensional problems. The layer of neurons that yields the output signals, is called output layer.

Neural networks differ in their network topology (architecture):

- the number of inputs and outputs
- the number of layers
- the number of neurons in each layer
- the number of weights in each neuron
- the way weights are linked together within or between the layers
- which neurons receive the correct input signals

We distinguish the following network topologies:

- **feed forward networks**
 - connected in layers (Figure 20a)
 - connected in layers and additional shortcut connections (Figure 20b)
- **feedback networks**
 - direct feedback (Figure 20c)
 - indirect feedback (Figure 20d)
 - lateral feedback (Figure 20e)
 - fully connected (Figure 20f) - Fully connected networks are rare special cases and became acquainted in particular with Hopfield [56-59] networks)

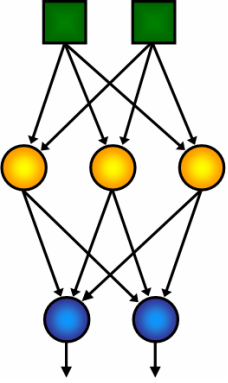
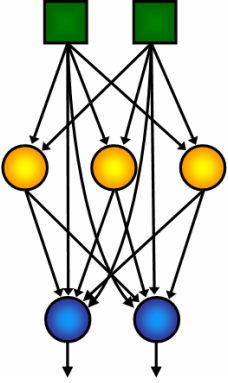
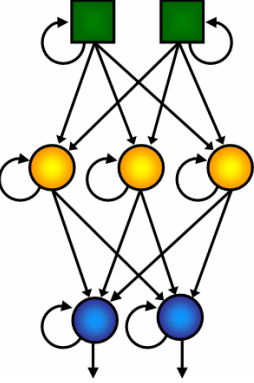
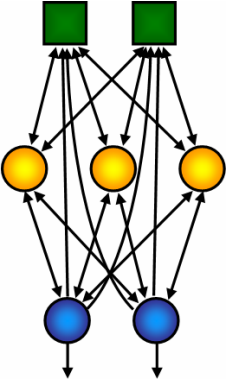
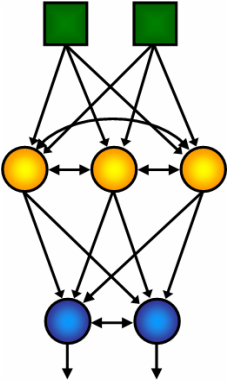
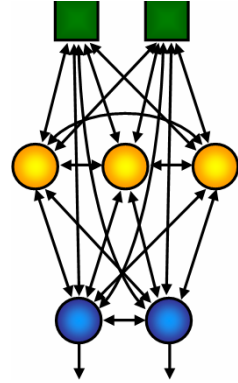
<p>a)</p> 	<p>b)</p> 	<p>c)</p> 
<p>These networks are divided into several levels (layers). There are only connections from a layer to the next.</p>	<p>With these networks there are connections between consecutive layers and connections, which jump over layers. For some problems, e.g. the Two-Spiral-Problem^[78] the shortcut connections are necessary.</p>	<p>These networks allow the adaptation of a neurons own activation over a connection from its exit to its entrance.</p>
<p>d)</p> 	<p>e)</p> 	<p>f)</p> 
<p>With these networks there is a feedback of neurons of higher levels to neurons of lower levels. This kind of feedback is necessary, if one wants to reach an increased sensitivity to certain ranges of input neurons or to certain input characteristics.</p>	<p>Networks with feedbacks within the same layer are often used for tasks, in which only one neuron in a group of neurons is to become active. Each neuron receives then restraining (in-hibitory) connections to other neurons and often still another activating (excitatory) direct feedback from itself. The neuron with the strongest activation (the winner) then restrains the other neurons, therefore such a topology is also called a <i>winner takes all</i> network.</p>	<p>Fully connected networks have connections between all neurons. They are in particular known as Hopfield networks^[56-59].</p>

Figure 20: Sample network topologies for feed forward and feedback networks

3.3.3. Training of artificial neural networks

A neural network has to be configured such that the application of a set of inputs produces (either 'direct' or via a relaxation process) the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using *a priori* knowledge. Alternatively, the neural network can be trained by feeding teaching patterns and allowing the weights to change according to some learning rules.

3.3.4. Learning in neural networks

As previously mentioned, the concept of learning in neural networks generally means that modifications of the combining weights occur in order to receive a better agreement between desired and actual output of the neural network. However, this presupposes that the desired output of the network must be known in advance.

Generally, there are three different ways of learning in neural networks ^[76]:

- reinforcement learning
- unsupervised learning
- supervised learning

3.3.4.1. Reinforcement learning

In reinforcement learning, only the overall correct or wrong output is indicated (possibly also the degree of the correctness). However, there are no output values for each output neuron at hand. The learning process has to find the correct output of these neurons itself. This kind of learning procedures are neurobiologically and/or evolutionary more plausible than supervised learning: Observations of lower and of higher organisms showed that simple feedback mechanisms (punishment with wrong decisions, reward with correct) from the environment exist and improve the learning process. On the other hand, these learning procedures are much more time-consuming. Compared to a method in which one knows the desired output (supervised learning), reinforcement learning needs more time since it has less information for the correct modification of the weights.

3.3.4.2. Unsupervised learning

With unsupervised learning (also called self-organized learning), the training set only consists of input samples. There is no desired output or data whether the net classified the training samples correctly or not. Instead, the learning algorithm tries independently to identify and illustrate groups of similar or neighboring neurons (cluster) of similar input vectors. The most well-known class of unsupervised learning procedures are the self-organizing maps of Kohonen[79]. In the trained state of the organizing map, similar input vectors are mapped on topologically neighboring neurons.

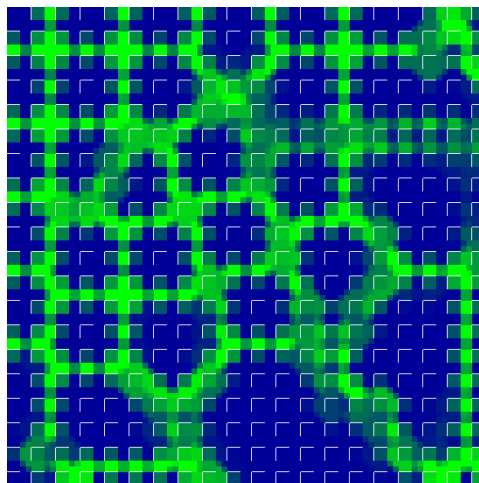


Figure 21: A sample Kohonen feature map (Euclidian distance map) made out of all monosaccharide units used in this thesis

Kohonen maps will be discussed in chapter 4.5.3. This class of learning procedures is the biologically most plausible one. Such topological maps have been found in the visual cortex of the mammalian brain ^[76].

Because unsupervised learning groups similar vectors into similar classes, these networks can be used for classification problems. For this purpose one only needs a number of reference vectors (training cases), whose transformation on the neurons is known in advance, and can then classify also unknown training samples according to their proximity to the next trained reference vector.

3.3.4.3. Supervised learning

With *supervised learning*, an external "teacher" indicates the correct and/or best output sample to each input sample of the training set. This means that for the network, a completely specified input sample and the correct and/or optimal completely specified output sample for this input is always available at the same time. The purpose of this learning procedure is to change the weights of the net in such a way that the net can make this association independently after repeated presentation of the input-output sample pairs. The network should also be able to recognize unknown, similar input samples (generalization). This kind of learning is usually the fastest and most used method to train a network for its task. The disadvantage of this approach is that it is biologically not plausible because no nervous system has its desired target neurons already activated in advance.

A typical supervised learning procedure, as for instance Back-propagation or its variations, accomplishes the following five steps for all pairs of input-output samples:

1. Presentation of the input pattern by appropriate activation of the input neurons (input unit).
2. Forward propagation of the input through the network; this produces a specific output pattern at the output neurons for the current input.
3. The comparison of the actual output with the desired output (teaching input) gives an error vector (difference, delta).

4. Back-propagation of the error from the output layer to the input layer provides changes of the combining weights, which serve to reduce the error vector.
5. Change of the weights of all neurons of the network by the error values computed in advance.

There are some variations in the details, particularly in the formulas for the computation of the weight changes. However, this pattern is in principle the basis of nearly all supervised learning procedures for non-recurrent networks (with no feedback connections).

3.3.5. Learning rules

The learning rule is the most interesting component of a neural network model because it allows the network to learn a given task only from its own examples. There are several possible ways of learning ^[76]:

- Development of new connections between neurons
- Deletion of connections
- Modification of the weight of a connection
- Modification of the threshold
- Modification of activation-, propagation- or output-function
- Creation of entirely new neurons
- Deletion of entire neurons

From these different alternatives, which can be used individually or in combination, the modification of the connection weights is by far most frequently used way of learning in neural networks. The development of a new connection between two neurons can be achieved relatively easy by modifying the connecting weight (from zero to some value >0). Similarly, the deletion of a connection is realized by changing the value of its weight to zero. The creation of new neurons finds their practical application in the cascade correlation neural networks ^[77].

The different learning rules used during this PhD thesis will be explained in chapter 4.

3.3.6. Modifying patterns of connectivity

All learning paradigms discussed above result in an adjustment of the weights of the connections between units according to some modification rule. Virtually all learning rules for models of this type can be considered as a variant of the Hebbian learning rule suggested in 1949 ^[46].

The basic idea is:

If neuron j receives an input from neuron i , and both neurons are strongly activated at the same time, then the weight w_{ij} of the connection from neuron i to neuron j is increased.

In the mathematical form, the Hebbian learning rule looks like show in the following equations:

$$\Delta w_{ij} = \eta o_i a_j \quad \text{Equation 7}$$

Thereby Δw_{ij} is the weight change of weight w_{ij} , η a constant (learning rate), o_i the output of the predecessor neuron i and a_j is the activation of the subsequent neuron.

According to Rumelhart and McClelland ^[52] the Hebbian learning rule in its general form looks like:

$$\Delta w_{ij} = \eta h(o_i, w_{ij}) g(a_j, t_j) \quad \text{Equation 8}$$

In this connection, the weight change Δw_{ij} is defined as the product of two functions:

1. The function $h(o_i, w_{ij})$ uses the output of the predecessor neuron o_i and the weight w_{ij} from neuron i to neuron j
2. The function $g(a_j, t_j)$ uses the activation of neuron a_j and the required activation (target output) of the neuron t_j

3.3.7. Advantages and disadvantages of neural networks

Considering as a whole, neural networks have many positive properties:

- **Learning aptitude:** Mostly neural networks are not programmed, but trained with a large class of training patterns. Thus, they are able to adapt their behavior to changing inputs.
- **A network learns the easiest features it can.**
- **Parallelism:** Neural networks are inherently highly parallel and therefore very suitable for an implementation on parallel computers.
- **Distributed knowledge presentation:** The "knowledge" of a neural network is saved within distribution of the weights. On one hand, this makes it possible to process the data in a parallel form and on the other, it results in a higher fault tolerance of the system against loss of single neurons or connections.

- **Associative storage of information:** here, information is stored oriented by its content. This associative method is not address-based as in conventional computer architectures. With neural networks, it is easy to recall a pattern that is only slightly similar to the entered test pattern.
- **Robustness against disturbance or noisy data:** Correctly trained neural networks respond less sensitive to noisy data or disturbances in the input pattern than conventional algorithms.

However, one must also consider the negative characteristics of neural networks:

- **Knowledge acquisition is only possible through training**
- **Learning is slow:** To analyze bigger problems with neural networks the amount of neurons and therefore weights is correspondingly larger. Many improvements of the known learning algorithms can reduce the problem but none solves it completely.

This issue will not be discussed here in detail. The different learning algorithms and error functions used in this PhD thesis will be discussed in chapter 4.5.

3.3.8. Application of neural networks

The fields of application of neural networks are normally those in which statistical and/or linear and also non-linear models can be used. In general, the use of neural networks provides better results than standard statistical techniques. Apart from science and engineering, some other fields in which neural networks are applied are shown below:

Table 6: Fields of application for neural networks

Finance	Index prediction, fraud detection, credit risk, classification, prediction of share profitability
Recognition of characters printed mechanically	Graphic recognition, recognition of hand-written characters, recognition of manual italic writing
Food	Odor and aroma analysis, customer profiling depending on purchase, product development, quality control
Energy	Electrical consumption prediction, distribution of water resources for electrical production, prediction of gas consumption
Manufacturing industry	Process control, quality control, control of robots
Medicine and health	Help to diagnosis, image analysis, medicine production, distribution of resources
Transports and Communications	Route optimization, optimization of the distribution of resources

3.3.9. Application of neural networks to NMR and carbohydrates

Intensive literature searches showed that there is relatively little cognition in this field of research. In SciFinder the keywords 'neural network', 'NMR' and 'carbohydrates' lead to only 14 hits for the time period between 1960 and 2002.

Meyer *et al.* ^[80-82] was the only researcher to perform fundamental experiments using neural networks and ¹H-NMR spectra of sugar alditols. He showed that a normal fully connected feed forward Back-propagation network could be trained with ¹H-NMR spectra and was able to recognize the trained data reliably. However, the dataset was highly limited consisting of only 24 training samples. Recall-tests with only parts of spectra (reporter groups) still led to good results. The whole work however remained fixed on the recall of already learned patterns. The real ability of a neural network – the generalization - was not tested and this contribution therefore must only be regarded as an initial attempt in the research field.

In 1998, Amendolia *et al.* ^[83] undertook further attempts in the area of sugar analytics and tried to quantify binary sugar alditol-mixtures with a set of neural networks on the basis of the ¹H-NMR spectra of mixtures thereof. They trained a separate network for each possible binary combination of the available alditols.

After these two contributions, the attempts witnessed limited attention for over 13 years. To date, no other researchers ever tried successfully to identify NMR spectra of oligosaccharides with the help of neural networks.

3.3.10. Other computer-assisted structural analysis systems for carbohydrates

Vliegthart and co-workers developed a ¹H and ¹³C-NMR database called SUGABASE, which combines the CarbBank^[84] and Complex Carbohydrate Structure Data (CCSD) with proton and carbon⁹² chemical shifts in a search routine^[85-88]. The search is based on the use of ¹H chemical shifts from the structural reporter groups^[19]. This concept is based on the fact that it is often sufficient to inspect only certain areas of a spectrum to ascertain the primary structure of a common glycoprotein carbohydrate structure. In the structural reporter group approach the region between 3 - 4 ppm is ignored and only the regions between 4 - 5.6 ppm and 1 - 3 ppm are inspected. The anomeric protons, methyl protons, protons attached to a carbon atom in the direct vicinity of a linkage position, and protons attached to deoxy carbon atoms are considered relevant structural reporter groups. The chemical shift values are used for a search in SUGABASE. The database is currently not being updated. The same is true for the CarbBank database^[84].

Jansson and Kenne developed the program CASPER (computer-assisted spectrum evaluation of regular polysaccharides) ^[38, 89-94]. This program has been developed to perform a structural analysis of both linear and branched oligo- and polysaccharides using ¹H and ¹³C chemical shift data and ¹J_{CH} or ³J_{HH} scalar coupling constants. The program allows both 1D and 2D data to be used for the spectra to be simulated. The database with the chemical shifts, different glycosylation shift, and correction sets for sterically strained structures will be more accurate with the increasing number of

assigned structural elements included, particularly with the addition of more data from branched molecules. CASPER can be used to extract glycosylation shifts and correction sets from newly assigned structures and incorporate them into the database.

3.4. Integration of NeuroCarb into the EuroCarbDB

3.4.1. What is EuroCarbDB

EuroCarbDBs is the abbreviation of 'distributed web-based European carbohydrate databases'. EuroCarbDB is a design study integrated in the 6th Framework Program for research and technology development (FP6) of the European Union. FP6 is a collection of actions at EU level to fund and promote research in transnational scientific projects.

EuroCarbDB is a union of researchers from five European countries:

- The German Cancer Research Centre and the University of Giessen in Germany
- The Bijvoet Center Utrecht in The Netherlands
- The Stockholm University in Sweden
- The University of Basel in Switzerland
- The European Bioinformatics Institute, Imperial College London and the University of Oxford in the UK.

The main reason why this union has been brought into being is the urgent need for an infrastructure, that will essentially improve the quality of the European carbohydrate research. EuroCarb will set up distributed data base systems for data exchange and new developments containing all kind of data about carbohydrates. In contrast to the genomic and proteomic area, no large data collections for carbohydrates have been compiled so far. However, the availability of such comprehensive data collections will be an important prerequisite to successfully perform large-scale glycomics projects aiming to decipher the biological functions of glycans. The definition of common protocols to enter new data will rapidly help to spread guidelines of good practice and quality criteria for experimental data, especially NMR-, MS- and HPLC-data, which are the key technologies for the identification and analysis of carbohydrates.

With the help of the internet, a global and interactive peer-to-peer communication for scientific data will be constituted. The initiative aims to overcome the existing fragmentation of European research in the area of bioinformatics for glycobiology through the development of standards, databases, algorithms and software components that are critical for the future of an excellent infrastructure. To guarantee maximal synergetic effects of the information contained in the newly created databases other available bioinformatics and biomedical resource have to be linked and cross-referenced in an efficient way.

The interpretation of both MS- and NMR spectra as well as HPLC profiles of glycans can be complicated without appropriate reference data. It is an urgent demand of ongoing high throughput

glycomics projects that efficient tools for automatic detection of glycan structures are provided. The development of appropriate algorithms, which enable a rapid and reliable automatic annotation and interpretation of MS- and NMR- spectra, is a major aim of the study. Therefore, the work described in this PhD thesis (NeuroCarb) is a more than suitable instrument for the solution of this problem and will be integrate in the EuroCarb union at best.

The EuroCarbDB has to be comprehensive and include the latest data to be attractive for the scientific community. However, to keep scientific data collections up-to-date and feed in continuously new data is one of the most time-consuming and thus expensive tasks to maintain an excellent database.

The existence of the EuroCarbDB infrastructure will provide high flexibility to include new types of experimental data, and encourage the development of new software tools and algorithms for data interpretation and analysis. Creating a GRID of distributed local databases will encourage people to input their recorded data prior to publication and keep it private. The data can be made available to the public after publication by the push of a button. In such a way, one potential source of error caused by extracting data from the literature will be eliminated. Additionally, the stored primary data will be more complete as can be guaranteed by any retrospective excerpt of experimental data. Another advantage of an open data base structure is that the ability to access primary experimental data in a digital format will attract researchers from outside the consortium to contribute to the development of the database, by creating new applications and algorithms and by including their own data.

To enable consistency between the various distributed databases a unique identifier for each carbohydrate structure must be defined. The LINUCS (Linear Notation for Unique description of Carbohydrate Sequences) notation has been proven suitable for this purpose. Based on the extended notations for complex carbohydrates as recommended by the IUPAC, the LINUCS-notation can be easily generated and looked up in a list of already existing LINUCS-codes, which is available from the master database.

The first goal will be to put together all required software modules (database management system, the data base design and the web-server). This can be installed quite easily on the distributed local hosts. The next step will be to develop standardized input options for NMR and MS, carbohydrates structures and references. These will be the first data to include as new entries into the database. The third step will be to include primary experimental data, which may originate from various instruments and will have different digital formats. The fourth step is then to provide standard exchange formats that allow access to these data across the Internet. This exchange format will be based on XML-definitions, which are sufficiently flexible to allow future extensions and require only a minimum effort to be adapted to existing databases. Future extensions of the database and new applications will be also based on the XML standard for data exchange.

3.5. The aims of this PhD thesis

The main objective was to develop a neural network based identification system capable of identifying monosaccharides from spectroscopic NMR data. In a further step, the system is to be extended in such a way that it can recognize monosaccharide moieties contained in disaccharides and still later oligosaccharides. Based on the initial efforts of Meyer *et al.* ^[80-82], the first task was to prove and improve this system, especially the generalization rate of the used networks.

Initially, the following fundamental questions have to be answered:

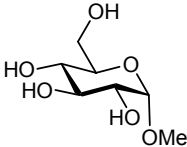
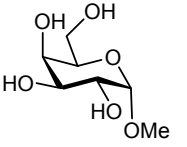
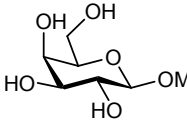
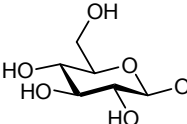
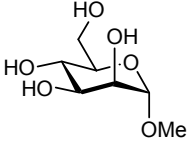
1. Is this information available through NMR spectroscopy? (monomer identity, anomeric configuration, substitution pattern).
2. What kind of NMR data provides this information (^1H or ^{13}C -NMR)?
3. How can spectroscopic data be transferred into a neural network?
4. Which network architecture, learning algorithm and learning parameters lead to optimal results?
5. Is an identification of monosaccharide moieties out of saccharide-mixture possible at all?

4. Material and Methods

4.1. Used chemical compounds

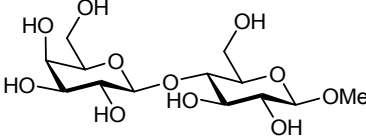
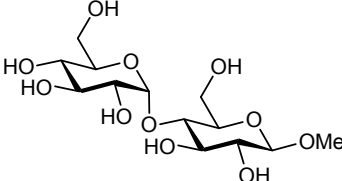
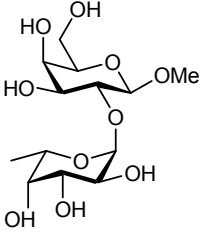
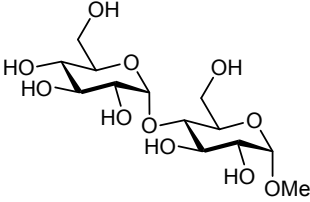
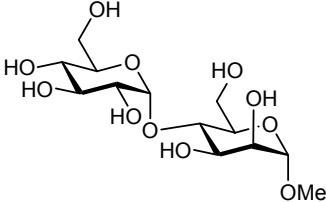
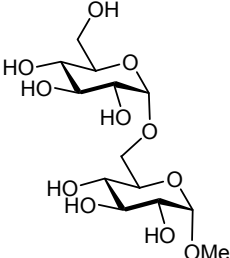
4.1.1. Methyl pyranosides

In house methyl pyranosides used for the training of SNNS neural networks.

<p>Compound 1: Methyl-α-D-glucopyranoside</p>  <p>Weighted sample: 0.0110 g</p>	<p>Compound 2: Methyl-α-D-galactopyranoside</p>  <p>Weighted sample: 0.0105 g</p>
<p>Compound 3: Methyl-β-D-galactopyranoside</p>  <p>Weighted sample: 0.0104 g</p>	<p>Compound 4: Methyl-β-D-glucopyranoside</p>  <p>Weighted sample: 0.0104 g</p>
<p>Compound 5: Methyl-α-D-mannopyranoside</p>  <p>Weighted sample: 0.0104 g</p>	

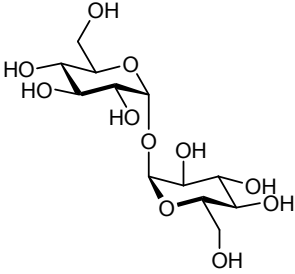
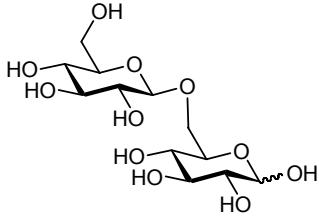
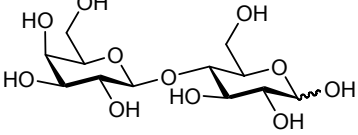
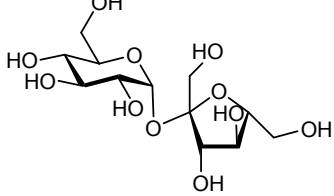
4.1.2. Hindsgaul compounds

(These compounds were kindly provided by Prof. Ole Hindsgaul (=OH), Carlsberg Research Center, Denmark)

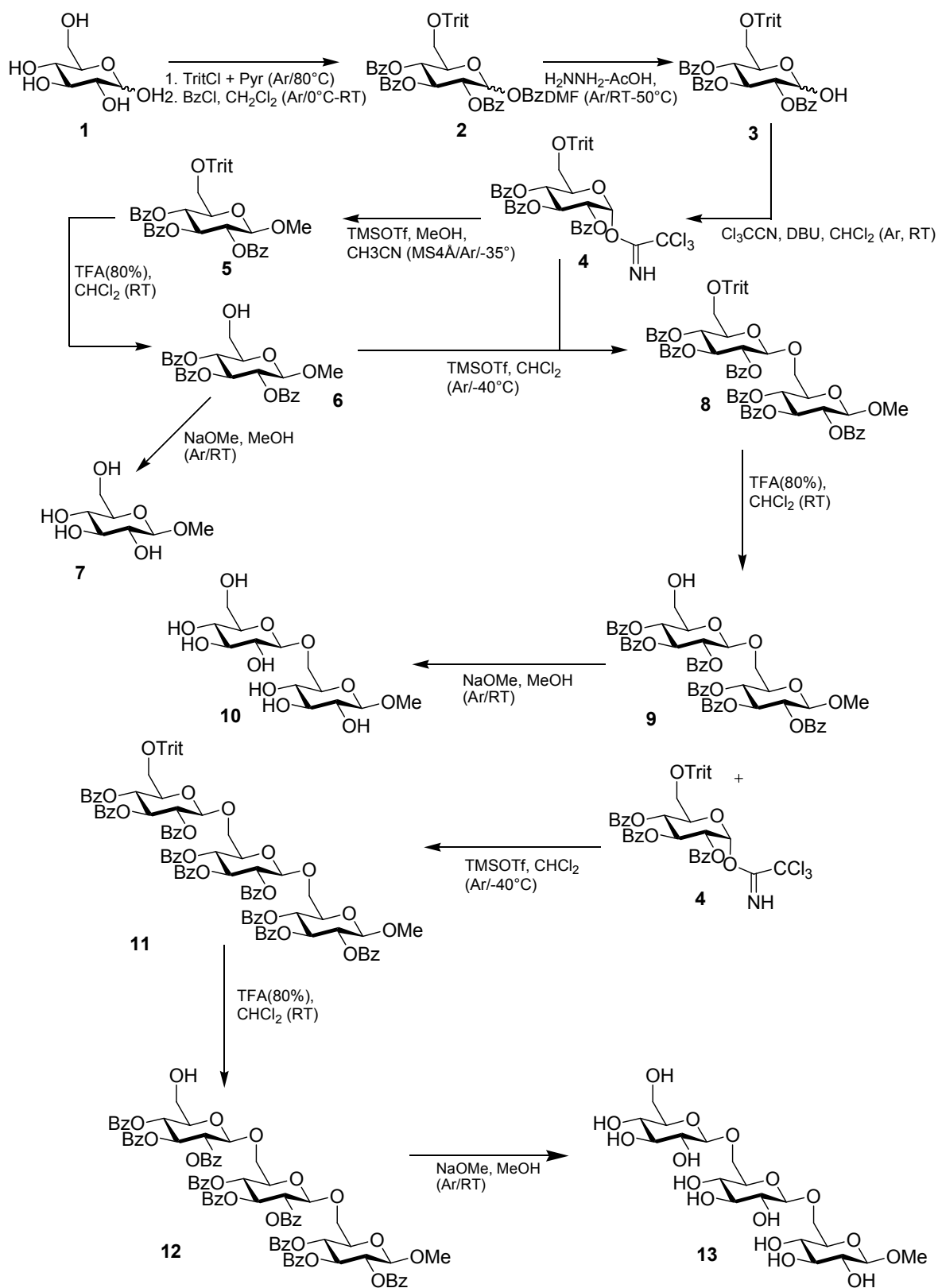
<p>Compound OH1: β-D-Galp-1-4-β-D-Glcp-OMe</p> 	<p>Compound OH7: α-D-Glcp-1-4-β-D-Glcp-OMe</p> 
<p>Compound OH3: α-D-Fucp-1-2-β-D-Galp-OMe</p> 	<p>Compound OH8: α-D-Glcp-1-4-α-D-Glcp-OMe</p> 
<p>Compound OH6: α-D-Glcp-1-4-α-D-Manp-OMe</p> 	<p>Compound OH9: α-D-Glcp-1-6-α-D-Glcp-OMe</p> 

4.1.3. Disaccharide test compounds

The following disaccharides were used as real test compounds for all neural networks trained and tested in chapter 5.8 and 5.9. The original NMR peak files can be found in appendix 11.1.

<p>Trehalose: α-D-Glcp-1-1-α-D-Glcp</p>  <p>Weighted sample: 0.0175 g</p>	<p>Gentiobiose: β-D-Glcp-1-6-β-D-Glcp</p>  <p>Weighted sample: 0.0201 g</p> <p>note: to use this disaccharide as a test compound the anomeric configuration of the second monosaccharide unit was artificially set to β configuration.</p>
<p>Lactose: β-D-Galp-1-4-β-D-Glcp</p>  <p>Weighted sample: 0.0216 g</p> <p>note: to use this disaccharide as a test compound the anomeric configuration of the second monosaccharide unit was artificially set to β configuration.</p>	<p>Saccharose: α-D-Glcp-1-2-β-Fruf</p>  <p>Weighted sample: 0.0216 g</p> <p>note: furanose forms of carbohydrates were not included in the neural network training. But this compound can serve as a positive and negative test all in one.</p>

4.1.4. Synthesis of β -D-glucopyranosyl-1-6- β -D-glucopyranosyl-1-6- β -D-glucopyranoside



4.1.4.1. Synthesis of O-Methyl β -D-glucopyranosyl-1-6- β -D-glucopyranosyl-1-6- β -D-glucopyranoside

2,3,4-tri-O-benzoyl-6-O-trityl- α -D-glucopyranosyl trichloroacetimidate (**4**):

D-Glucose (**1**, 5.03g, 27.9mmol) was coupled with tritylchloride (9.20g 33mmol, 1.2eq.) in a solution of dry pyridine (30ml) at 80°C during 15 h under Ar protection. The reaction mixture was cooled to 0°C, diluted with 20ml of CH₂Cl₂, and benzoylchloride (19.4ml, 167.4mmol, 6eq.) was added slowly. After stirring during 5h at RT, the solution was diluted and extracted with EtOAc (3 times), washed with H₂O and brine, dried (Na₂SO₄) and concentrated. The crude product was further purified on a silica gel column with 4:1 petroleum ether - EtOAc as eluent, to obtain 1,2,3,4-tetra-O-benzoyl-6-O-trityl-D-glucopyranoside (**2**, 18.30g, 78%).

2,3,4-tri-O-benzoyl-6-O-trityl-D-glucopyranose (**3**, 8.27g, 59%) was obtained by treating **2** (15.83g, 19mmol) with H₂NNH₂-AcOH (3.84g, 37.8mmol, 2 eq.) in DMF (200ml) under Ar protection for 4h at 50°C and a terminal purification as described for **2** (3:1 petroleum ether – EtOAc as eluent).

Compound **3** (8.2g, 11.2mmol) was dissolved in dry CH₂Cl₂ (100ml), CCl₃CN (5.6ml, 55.8mmol, 5eq.) and 1,8-Diazobicyclo-[5,4,0]-undec-7-en (0.83ml, 5.58mmol, 0.5 eq.) were added and the mixture was stirred for 6h at RT. After concentration and purification on a silica gel column with 6:1 petroleum ether-EtOAc as eluent, compound **4** (6.56g, 66.6%) was obtained.

Methyl β -D-glucopyranoside (**7**): (RS1)

Compound **4** (6.54g, 7.5mmol) was dissolved in dry CH₃CN (100ml). MeOH (0.6ml, 15mmol, 2eq.) was added and the solution was stirred during 1h over a molecular sieve (MS4Å) under Ar protection. After drop wise addition of TMSOTf (0.2ml, 1.125mmol, 0.15eq.) at -35°C stirring was continued during 1.5h. Then the mixture was neutralized with Et₃N and concentrated. Purification of the residue on a silica gel column with 6:1 petroleum ether-EtOAc as eluent, gave Methyl 2,3,4-tri-O-benzoyl-6-O-trityl- β -D-glucopyranoside (**5**, 3.48g, 64.4%). Cleaving of the trityl group was achieved by adding TFA (80%, 2.75ml) to a solution of **5** (3.44g, 4.6mmol) in CH₂Cl₂ (150ml). After stirring for 2h, saturated NaHCO₃ solution (70ml) was added. The resulting colorless reaction mixture was extracted with CH₂Cl₂ (3 times), washed with H₂O and brine, dried with Na₂SO₄, followed by purification by flash chromatography (4:1-1:1 petroleum ether-EtOAc as eluent) to yield Methyl 2,3,4-tri-O-benzoyl- β -D-glucopyranoside (**6**, 1.84g, 79%). Finally the benzoyl groups were cleaved by adding a catalytical amount of a freshly prepared Na-methanolate solution to a solution of **6** (108mg) in methanol (2ml) under Ar protection at RT to afford pH of 9 of the reaction mixture. After stirring for 1h the mixture was neutralized with acidic Amberlyste15, filtrated and concentrated. Purification on a silica gel column with 4:1:0.2 CH₂Cl₂-MeOH-H₂O as eluent to provided **7** (39mg, 95%).

Methyl β -D-glucopyranosyl-1-6- β -D-glucopyranoside (10): (RS2)

Compounds **6** (288mg, 0.568mmol) and **4** (502mg, 0.568mmol) were dissolved in dry CH₃CN (12ml) over activated MS4Å under Ar protection and stirred for 1h at RT. The solution was then cooled down to -40°C and TMSOTf (15 μ l, 0.15eq.) was added. The resulting yellow mixture was stirred at this temperature for 3h, then neutralized with Et₃N (~0.2ml) to produce a colorless solution, then filtered and concentrated. Purification on a silica gel column with 4:1-2:1 petroleum ether – EtOAc as eluent provided Methyl 2,3,4-tri-O-benzoyl-6-O-trityl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranoside (**8**, 460mg, 66%).

Methyl 2,3,4-tri-O-benzoyl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranoside (**9**, 228mg, 84.4%) was obtained by cleaving the trityl group from **8** (337mg, 0.275mmol) under the same conditions as described for the preparation of compound **6**. Methyl β -D-glucopyranosyl-1-6- β -D-glucopyranoside (**10**, 34mg, 81%) was obtained by cleaving the benzoyl groups from **9** (145mg, 0.118mmol) and neutralizing the mixture by the same method as described for **7**. Final purification of the crude product on a silica gel column with 10:4:0.8 CH₂Cl₂-MeOH-H₂O as eluent and in addition on a Sephadex™G15 column with H₂O as eluent was required to give **10** (39mg, 95%).

Methyl β -D-glucopyranosyl-1-6- β -D-glucopyranosyl-1-6- β -D-glucopyranoside (13):

Methyl 2,3,4-tri-O-benzoyl-6-O-trityl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranoside (**11**, 148mg, 48.6%) was obtained by coupling **9** (180mg, 0.183mmol) and **4** (177mg, 0.02mmol, 1.1eq.) with TMSOTf (7.5 μ l, 0.225eq.) under the same conditions as described for **8**. The crude product was purified on a silica column with 15:1 toluene-EtOAc as eluent. Methyl 2,3,4-tri-O-benzoyl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranosyl-(1 \rightarrow 6)-2,3,4-tri-O-benzoyl- β -D-glucopyranoside (**12**, 93mg, 78.1%) was obtained by cleaving the trityl group from **11** (140mg, 0.084mmol) under the same conditions as described for the preparation of compound **6**. The crude product was purified on a silica column with 6:1 toluene-EtOAc as eluent. Methyl β -D-glucopyranosyl-1-6- β -D-glucopyranosyl-1-6- β -D-glucopyranoside (**13**, 30mg, quant.) was obtained by cleaving the benzoyl groups from **12** (90mg, 0.063mmol) and neutralizing the mixture by the same method as described for **7**. Purification of the crude product was performed as described for **10**.

4.1.5. ¹³C-NMR Database

Due to the lack of good and well-maintained ¹³C-NMR databases (including solvent, standards and temperature) it was inevitable that an own database based on FileMaker 6 (FileMaker Inc.) had to be designed from scratch. A robust base training dataset is the most important and indispensable precondition for good generalization results of a neural network.

During an extensive literature research over one thousand different ¹³C-NMR peak lists of carbohydrates (mono-, di- and oligosaccharides) were collected [38, 73, 76, 77, 95-291].

Table 7: Oligosaccharide statistics of ¹³C-NMR FileMaker® database

Saccharides	Number
Monosaccharides	168
Disaccharides	381
Trisaccharides	255
Tetrasaccharides	83
Pentasaccharides	59
Hexasaccharides	44
Heptasaccharides	4
Nonasaccharides	2
Total monosaccharide units	2632

All peaks of the registered compounds in the database were corrected according to Gottlieb *et al.* [292]. Literature compounds with no available internal or external standards were fed into the database anyway but were marked accordingly and may be useful for later testing purposes (e.g. robustness) of trained neural networks,

Via the FileMaker's open database connectivity (ODBC) interface the NMR peaks are easily accessible also in other ODBC compatible programs as Microsoft Excel, Statsoft Statistica™ [293] or future releases of the ANN Pattern File Generator (chapter 4.9). The data records can also be exported into CSV Files (chapter 4.9.2.2) with the help of different FileMaker export scripts.

The database is accessible online within the Institute of Molecular Pharmacy. In connection with the *EuroCarbDB* project, the database will be placed at the disposal of all institutes united under *EuroCarbDB*. This will be done via a migration to a MySQL-server and an intuitive web interface (*EuroCarbDB* efforts in chapter 3.4).

13-C-NMR Datenbank

systematic_name: Beta-D-galactopyranosyl-1-4-(2-acetamido-2-deoxy)-beta-D-glucopyranosyl-1-2-(beta-D-galactopyranosyl-1-4-(2-acetamido-2-deoxy)-beta-D-glucopyranosyl-1-6)-alpha-D-mannopyranose.....

quickname: b-D-pGal-1-4-b-D-pGlcNAc-1-2-(b-D-pGal-1-4-b-D-pGlcNAc-1-6)-b

lit_ref: Book K. C. Pedersen, et al. (1984), "Carbon-13", 390 xis

entered_by: MS, mod_date: 30-8-2004

solvent: D2O, Ref. (ppm): 67.19

ext_ref: Dioxane, corr_ppm: -0.21

Chemical structure diagram showing a branched oligosaccharide chain with various linkages and substituents.

Diagram illustrating the numbering of sugar units in the main chain (1-5) and side chains (SA1, SA2, SA3, SA4, SA5).

Original peak values table:

	1	2	3	4	5
1 a-D-pMan	92.8	78.9	73.9	71.7	70.7
1A1 b-D-pGlcNAc	103.2	80.2	76.3	73.4	81.6
1A2 b-D-pGal	104.8	76.9	74.1	72.5	70.1
2 b-D-pGlcNAc	101.2	80.1	76.3	73.2	81.5
3 b-D-pGal	104.8	76.9	74.1	72.5	70.1

Figure 22: Main input layout of the 13C-NMR FileMaker database

4.1.5.1. Nomenclature

All monosaccharides are assumed to be in the D-configuration except for fucose and Iduronic acid, which are in the L-configuration. All glycosidically-linked monosaccharides assumed to be in the pyranose form. All monosaccharide glycosidic linkages are assumed to originate from the 1-position except for the sialic acids, which are linked from the 2-position

To enter a structure into the database, the quick names are assembled according to the following rules:

- The main chain is determined by the longest monosaccharide moieties chain
- If this rule is not applicable, the alphabet or the linkage is taken to assistance.
- The monosaccharide unit at the beginning of the main chain is determined by its free anomeric carbon atom (maybe substituted with OH or OMe)
- The monosaccharide units of the main chain are labeled with numbers starting from '1'
- Monosaccharide units of side chains are designated with capital letters starting with A
- The numbering of the units in a side chain starts again with the number '1'
- If there are several side chains starting at the same unit of the main chain, the one with the lowest initial letter (in alphabetical order) is designated A. The other chains with B, C, D and so on.

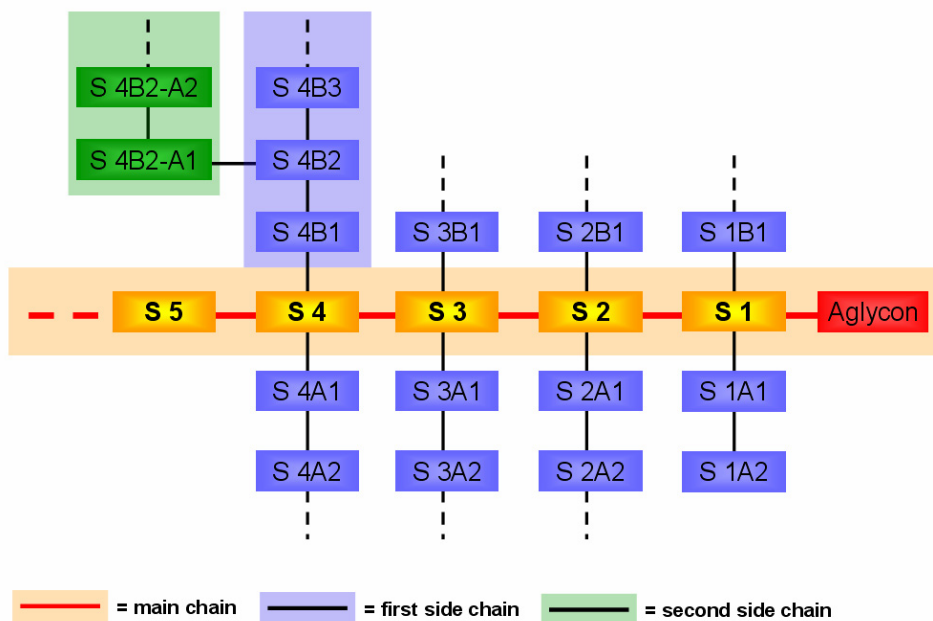


Figure 23: Oligosaccharide description scheme

The only simplification consists of the fact that only two side chains per monosaccharide unit of the main chain can be designated and entered into the FileMaker database. A fictive nomenclature example is outlined in figure 24.

α -D-mannopyranosyl-1-3-(α -D-mannopyranosyl-1-6)- α -D-mannopyranosyl-1-3-(α -D-mannopyranosyl-1-3-(α -D-mannopyranosyl-1-6)- α -D-mannopyranosyl-1-6)- α -D-mannopyranoside

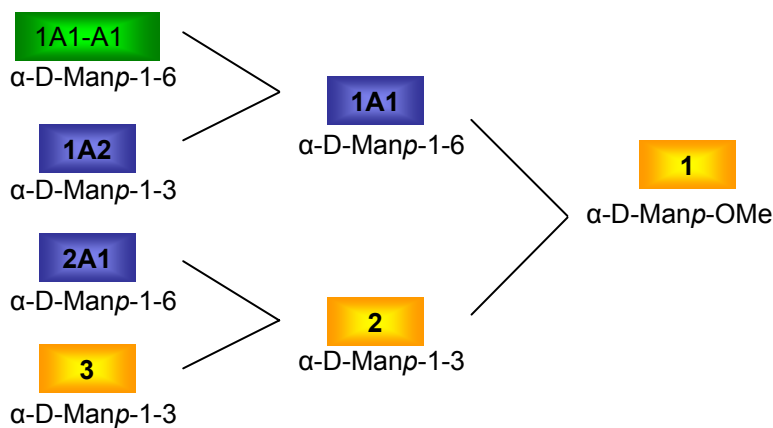
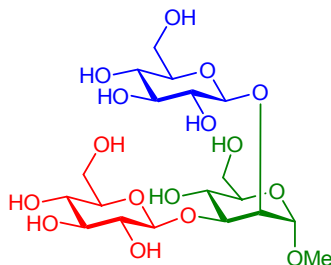
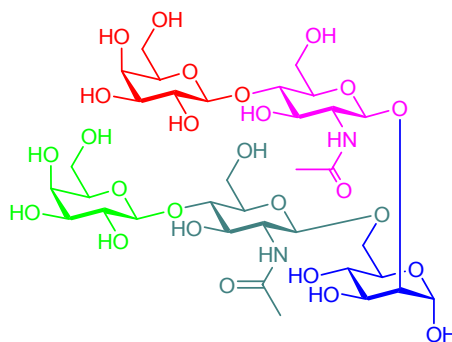


Figure 24: A fictive nomenclature example

4.1.5.2. Nomenclature examples for quick names



Structure 1: database record ID 32



Structure 2: database record ID 745

Figure 25: Two nomenclature examples for quick names

4.2. NMR equipment & experiments

All NMR experiments were acquired on the in-house Bruker™ 500 MHz UltraShield™ AVANCE™ Two-Bay Spectrometer. The Bruker XWIN-NMR Software package was used for spectrometer control, data acquisition, and processing. All experimental data was exported to JCAMP-DX for NMR files.

All NMR spectra (^1H and ^{13}C) were acquired in D_2O and at room temperature. ^1H spectra were scanned sixteen times and ^{13}C spectra 256 times.

4.3. Computer hardware

All neural networks were computed on the following hardware:

- Precision WorkStation with Dual Intel™ Xeon 2.0 GHz, 1GB RIMM Dual Channel PC800 ECC RAM - RedHat Linux 7.2
- Precision WorkStation with Dual Intel™ Xeon 2.4 GHz, 1GB RIMM Dual Channel PC800 ECC RAM - Microsoft Windows XP Professional
- Precision WorkStation with Dual Intel™ Xeon 3.4 GHz, 2 GB SDRAM – Microsoft Windows XP Professional

4.4. IUPAC JCAMP-DX

4.4.1. Summary

Because Bruker uses a proprietary and encoded NMR data storage format in its XWIN-NMR software package, it was not possible to extract the raw NMR data from the acquired NMR data and feed as input into a neural network. A solution was quick at hand: the JCAMP-DX exchange format. Fortunately, Bruker supports the export of collected NMR data into JCAMP-DX v.5.0 for NMR files.

JCAMP was an organization sponsored jointly by many scientific societies all over the world. This committee has been the source of several spectroscopic data exchange protocols. The first one for infrared spectroscopy [185] was published in 1988, other for chemical structure data [138], nuclear magnetic resonance spectroscopy [124] and mass spectrometry [178] followed later.

4.4.2. Detail insight into a JCAMP-DX file

The JCAMP-DX file is divided into three sections

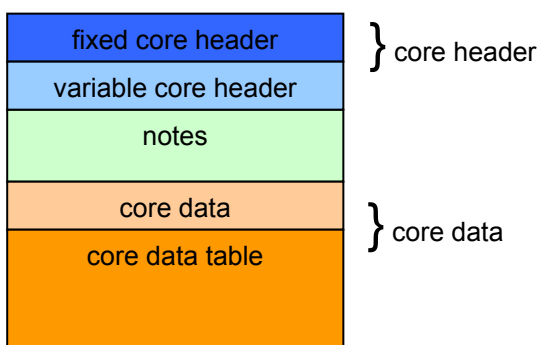


Figure 26: JCAMP-DX file structure overview

The intention is to separate the essential information to be parsed by the computer from the associated non-critical support data. The core is the irreducible minimum content of a JCAMP-DX file. The header contains all parameters defining the data set at the end of the file (core data) ^[252].

The core itself consists of four parts: 1) The first part, called the “Fixed Header Information”, contains labeled data records (LDRs), which are required for all JCAMP-DX files and which appear at the beginning of each file in a given order. 2) The “Variable Header Information” contains records which are data type specific (in this case NMR-specific) or which are used only in special types of JCAMP-DX files (e.g., compound files). Whether a particular LDR is required or not depends on the application. 3) The third section, “Core Data,” contains the spectral parameters for the fourth section, the “Data Table.” The type of data in the data table determines the parameters, which must appear in the core data. Only one data table may appear per JCAMP-DX block (a block being a part of the JCAMP-DX file starting with `##TITLE=` and ending with `##END=`)^[124].

4.4.3. The internal file format

All JCAMP-DX files are ASCII alphanumeric files consisting of lines of up to 80 characters long terminating in a carriage-return (CR) or linefeed (LF)^[252]. The entire file is made up of LDRs which all have the same basic structure:

```
##descriptor= xxx
```

The leading two hash signs tag the start of a new record. The *descriptor* is the label of a new data record. The following equal sign closes the data label. The LDR then continues with the data set until the parsing software reads the next LDR. Theoretically, a data record could run over more than just one line in the JCAMP-DX file. (`$$` indicates that the remainder of the line is a comment!)

Example fixed core header section of a Bruker JCAMP-DX v.5.0 for NMR file:

```
##TITLE= Name Gentiobiose / Project ANN / 20.1mg / c13cpdstd256 D2O
##JCAMPDX= 5.0
$$ Bruker NMR JCAMP-DX V1.0
##DATATYPE= NMR Spectrum
##DATACLASS= XYDATA
##ORIGIN= Bruker Analytik GmbH
##OWNER= mstuder
```

Data sets can consist of TEXT which is alphanumeric information of no predefined format such as in the example above for the title or as could be found in the `##OWNER` and `##ORIGIN` LDRs. They can also consist of alphanumeric data in the form of predefined values specified in the various JCAMP-DX protocols.

The JCAMP-DX for NMR guidelines have the same basic form like all the other JCAMP-DX protocols. The only major difference is the form which LDRs specific to NMR spectroscopy are written. The DATATYPE SPECIFIC information, whether belonging to the CORE or the NOTES section of the file have LDRs written in the following form:

```
##.labelname=
```

The additional period following the two hashes indicates a label specific to the type of spectroscopy identified in the `##DATATYPE` field.

The full data type specific LDR would read:

```
##NMR SPECTRUM.OBSERVE NUCLEUS= xxxx,
```

Concatenating the data type with the label name but the data type is left out to simplify matters.

A typical JCAMP-DX v.5.0 for NMR file ^[124] generated with Bruker XWIN-NMR from a 32k (32768 data points) ¹³C-NMR looks as follows (shortened in certain parts). Some LDRs of special importance for this thesis will be discussed later in this chapter

```
##TITLE= Name Gentiobiose / Project ANN / 20.1mg / c13cpdstd256 D20
##JCAMPDX= 5.0
$$ Bruker NMR JCAMP-DX V1.0
##DATA TYPE= NMR Spectrum
##DATA CLASS= XYDATA
##ORIGIN= Bruker Analytik GmbH
##OWNER= mstuder
$$ XWIN-NMR Version 3.0
$$ Mon Mar 24 14:39:38 2003 "MET (UT+1h)
##.OBSERVE FREQUENCY= 125.771571864236
##.OBSERVE NUCLEUS= ^13C
##.ACQUISITION MODE= SIMULTANEOUS
##.AVERAGES= 256
##.DIGITISER RES= 17
##SPECTROMETER/DATA SYSTEM= drx500
$$ Bruker specific parameters
$$ -----
##$DU= </z>
##$EXPNO= 20
##$NAME= <Mar21-2003>
##$EXP= <c13cpdstd256>
##$INSTRUM= <drx500>
##$SOLVENT= <D2O>
##$YMAX_p= 556486971
##$YMIN_p= -89040629
$$ End of Bruker specific parameters
$$ -----
##XUNITS= HZ
##YUNITS= ARBITRARY UNITS
##XFACTOR= 0.95970155584897
##YFACTOR= 1
##FIRSTX= 31446.5408805032
##LASTX= 0
##DELTAX= -0.95970155584897
##MAXY= 556486971
##MINY= -89040629
##NPOINTS= 32768 (32k Datapoints)
##FIRSTY= -22508377

##XYDATA= (X++ (Y..Y))
  32767.00000000    -22508377    -1367291    -5883613    3044087
  32763.00000000    -7614526     9764630     7765945    24592774
  32759.00000000    10390742     6682998    -5237532    10476673
  32755.00000000     6050057    -8171106    11107674    31413300
  32751.00000000    13364826   -18005683   -12936085   -6330169
  32747.00000000    11108544    15060429   -11885205   -18129920
  32743.00000000   -35745512   -8114840   -11074660   -3953471
  32739.00000000    12553316   -18400264   -11216800   -18484949
  32735.00000000   -16974031  -25466643     6790507    1051515
  32731.00000000     8564863    2882937   -18718790    11071256
  32727.00000000    18799416    13960988   -11076169   -355478
  ...
  19.00000000    -8952473    -4672043   -13698200   -36492374
  15.00000000   -13358629   -15377731   -31312215   -17676944
  11.00000000   -8440680   -3872342    6481776   -17295108
  7.00000000   -31234689  -15539587    17322022    14852773
  3.00000000     532254   -12077914   -13717156   -5059602
##END=
```

The LDR ##DATA TYPE= (third line of a JCAMP-DX file) affects the form of data that is stored in the last ##YXDATA= data record. E.g. NMR SPECTRUM, NMR PEAK TABLE, NMR FID or NMR PEAK SSIGNMENTS.

The `##XYDATA= LDR` is mostly a couple of thousand lines long (depending on the chosen spectrometer resolution: 8k, 16k and 32k) and contains the actual NMR data. The data block is terminated with an `##END=` tag. The NMR data can be stored either as ASCII free format numeric (AFFN), which can be in scientific notation, or in a compressed coded form called ASCII Squeezed Difference (ASDF). These two formats will be explained in the next chapters.

4.4.3.1. ASCII free format numeric (AFFN)

AFFN is similar to the free-form numeric I/O of BASIC and other popular computer languages. It is a combination of FORTRAN I, F, and E formats. An AFFN data item consists of a mantissa plus an optional exponential part. The mantissa can be an integer in FORTRAN I format, or a decimal in Fortran F format. The combination of mantissa and exponential is effectively Fortran E format. It is necessary to exclude the exponential term from the abscissa at the beginning of a line to prevent confusion with SQZ data items which start with E or e via `##XFACTOR=`. Thus, the data type of `##XYDATA=` is expressed as AFFN- or ASDF. The Bruker example above is written in AFFN.

Adjacent AFFN numeric fields are separated by blank(s), tab, comma, +, or –

Example:	12-3+4 (tab) 5,6,7 ,8,,
Translation:	1, 2, -3, 4, 5, 6, 7, 8, null entry.

Notice that "7 ," is interpreted as 7 not "7 + a null entry". In other words, when a numeric field containing at least one digit is terminated by blank, the scanner should skip ahead to the next non-blank. If that non-blank is comma, it should also be skipped. A blank field followed by a comma is interpreted as a NULL entry (i.e., no change in existing value).

The form of TABULAR DATA is represented symbolically as a *variable list* as follows:

$(X + (Y..Y))$, where `..` indicates indefinite repeat until the line is filled and `++` indicates that X is incremented by $(LASTX - FIRSTX) \div (DATAPOINTS - 1)$ between adjacent Ys.

X-values	Successive Y-values
7087.00000000	→ 48420955 → -5953663 → -19724977 → -13618317
7083.00000000	→ -20990412 → -32154519 → -22559434 → 25247761
7079.00000000	→ 1516879 → -5128452 → -23129053 → -5284456

In a table format, the above example section would look like this:

Table 8: JCAMP-DX (X+ + (Y..Y)) Data

x-values	y-values
7087	48420955
7086	-5953663
7085	-19724977
7084	-13618317
7083	-20990412
7082	-32154519
7081	-22559434
7080	25247761
7079	1516879
7078	-5128452
7077	-23129053
7076	-5284456

The (X+ + (Y..Y)) notation leads to a about ¼ smaller size of the JCAMP-DX file. However, it would be helpful to achieve higher compression rates, because a standard 32k JCAMP-DX NMR file normally has a file size of about 500 KB! Higher compression is only possible with a different compression coded form like ASDF (not discussed in this thesis).

After initial experiments with ASDF compressed JCAMP-DX files, the post data handling showed to be too complicated and time consuming and the compression was abandoned. There was no need to compress the data because of sufficient disk space and fast in house Ethernet network connections. Software compressing and decompressing of JCAMP-DX for NMR files consumes a lot of needless computing power.

JCAMP-DX files not used anymore can easily be stored in compressed TAR file archives with a compression factor ~3.

4.4.4. Important LDRs for regaining the original NMR data (in ppm)

The tabulated data points can be converted back to real ppm values (and vice versa) by dint of the following formulas:

$$DXpt = \frac{frequency[HZ] \times (pt[ppm] + offset) - FirstX}{XFactor} \quad \text{Equation 9}$$

$$pt[ppm] = \left(\frac{FirstX + (DXpt \times XFactor)}{Frequency[HZ]} \right) - offset \quad \text{Equation 10}$$

The following LDRs are important to regain the original recorded NMR spectrum:

Table 9: important LDRs for NeuroCarb

LDR	Unit	Description
##\$OFFSET=	[pt]	A Bruker specific parameter to shift the spectrum back to its original position
##MAXY= / ##MINY=	[AU]	The minimal and maximal peak intensities are needed for correct ordinate scaling of the NMR spectrum (Figure 57)
##XFACTOR=	-	These is the factor by which components of the tabulated abscissa values must be multiplied to obtain actual ppm values
##FIRSTX=	[pt]	Specifies the abscissa corresponding to the first value listed (Spectral data can be tabulated in order of <u>either</u> increasing or decreasing abscissa values!)
##NPOINTS=		Number of measured data points. This value determines the resolution of the NMR spectrum (typical values are: 32768 = 32k, 16384 = 16k, 8192 = 8k)
##.OBSERVE FREQUENCY	[MHz]	Observer Frequency of the instrument
##.OBSERVE NUCLEUS	-	Observed nucleus (e.g. ¹³ C, ¹ H)
##XYDATA= (X++ (Y..Y))		The actual tabulated measuring points as discussed above

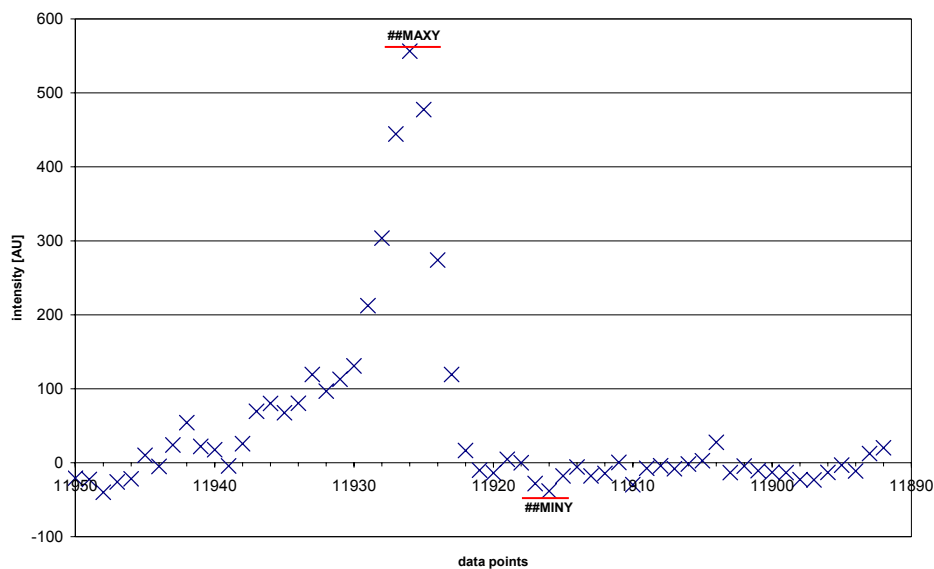


Figure 27: Graphical JCAMP-DX file x-y chart illustration of a sample ¹H-NMR peak

4.5. Multi-Layer Perceptrons (MLP) and the Back-propagation learning method

MLP is perhaps the most popular network architecture in use today. This is the type of network discussed briefly in the introduction: the units each perform a weighted sum of their inputs and pass this activation level through a transfer function to produce their output, and the units are arranged in a layered feed forward topology. The network thus has a simple interpretation as a form of input-output model, with the weights as the free parameters of the model. Such networks can model functions of almost arbitrary complexity, with the number of layers, and the number of units in each layer, determining the function complexity. Important issues in MLP design include specification of the number of hidden layers and the number of units in these layers^[77].

The number of input and output units is defined by the problem. The number of hidden units to use is far from clear. There are many rules of thumb, but the only way to determine the number of hidden layers and number of neurons is by testing as many network architectures as possible and compare their performance/error – trial and error.

Once the number of layers and number of units in each layer have been selected, the network's weights and thresholds must be set to minimize the prediction error made by the network. This is the role of the training algorithm that is used to automatically adjust the weights and thresholds in order to minimize this error. This process is equivalent to fitting the model represented by the network to the training data available. The error of a particular configuration of the network can be determined by running all the training cases through the network, comparing the actual output generated with the desired or target outputs (Figure 28). The differences are combined by an error function to give the network error.

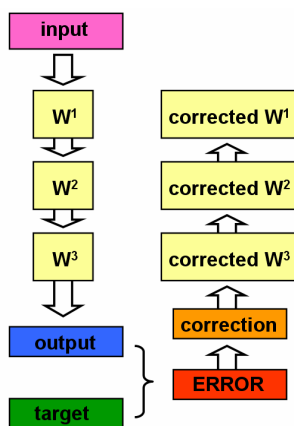


Figure 28: Schematic presentation of weight correction (Adapted from J. Zupan and J. Gasteiger^[232])

The most common error functions (chapter 4.6) are the sum-squared error (used for regression problems), where the individual errors of output units on each case are squared and summed together, and the cross entropy functions (used for maximum likelihood classification).

The Back-propagation learning method is a typical gradient method. All gradient methods compute the gradients of a target function (error function), and rise either orthogonal to gradients upward, until a maximum is reached or downward, until a minimum is reached. In the area of neural networks one tries to minimize the error by changing the weights by a negative fraction of the error function. A helpful concept here is the error surface. Each of the N weights and thresholds of the network (i.e., the free parameters of the model) is taken to be a dimension in space. The $N+1^{\text{th}}$ dimension is the network error. For any possible configuration of weights, the error can be plotted in the $N+1^{\text{th}}$ dimension, forming an error surface. The objective of network training is to find the lowest point in this many-dimensional surface.

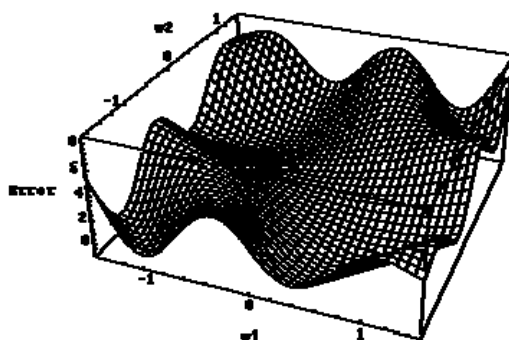


Figure 29: 3D error surface of a neural network as a function of weights w_1 and w_2 (adapted from ^[76])

Neural network error surfaces are much more complex as showed above (Figure 29), and are characterized by a number of unhelpful features, such as local minima (which are lower than the surrounding terrain, but above the global minimum), flat-spots and plateaus, saddle-points, and long narrow ravines (chapter 4.5.1).

It is not possible to determine analytically where the global minimum of the error surface is, and so neural network training is essentially an exploration of the error surface. From an initially random configuration of weights and thresholds (i.e., a random point on the error surface), the training algorithms incrementally seeks for the global minimum. Typically, the gradient (slope) of the error surface is calculated at the current point, and used to make a downhill move. Eventually, the algorithm stops in a low point, which may be a local minimum.

4.5.1. Problems of the Back-propagation learning method

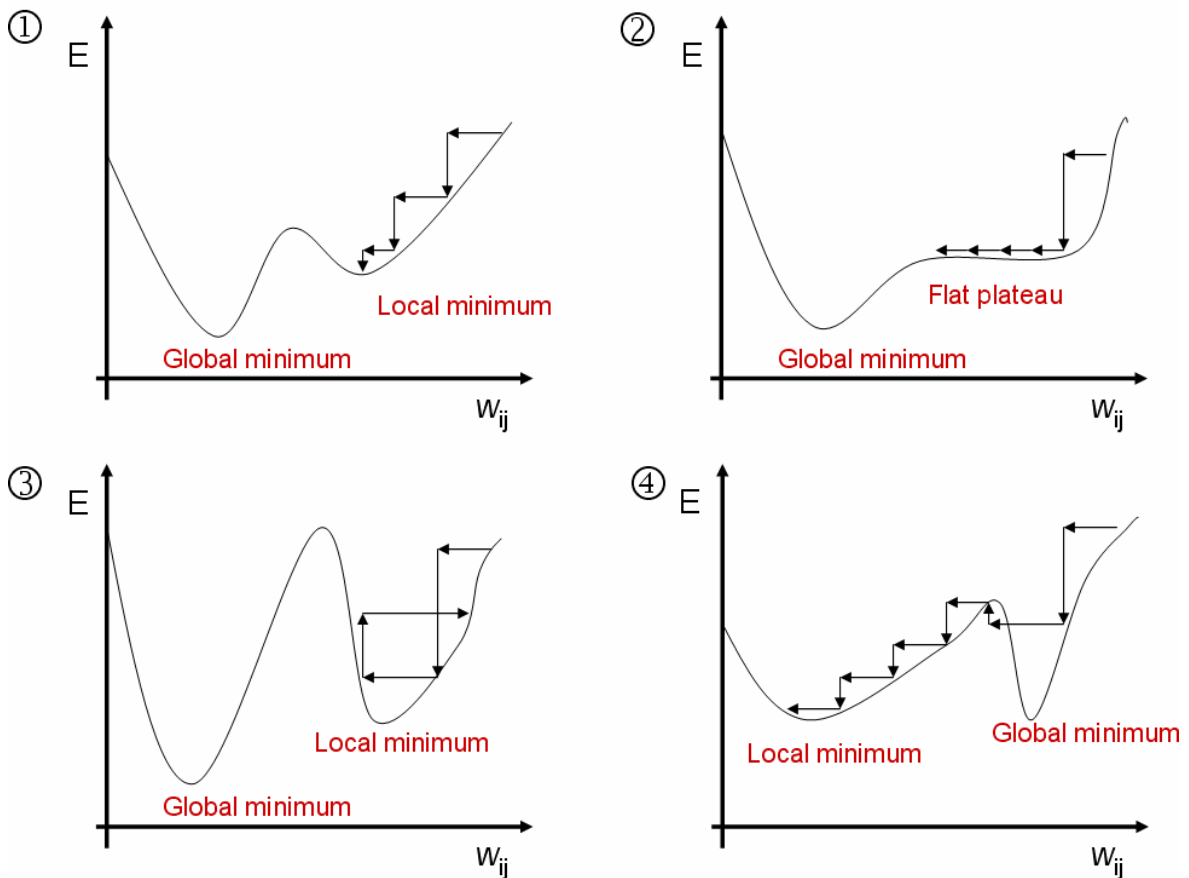


Figure 30: Problems of gradient methods - ① they only find local minima, ② they get stuck on flat plateaus, ③ oscillation in narrow ravines and ④ they leave good minima.

4.5.1.1. Local minima

Gradient methods all have the problem that they can get stuck in a local suboptimal minimum of the error surface (Figure 30 ①). The problem of neural networks is, that the error surface gets cliffy with increasing dimension of the network (with increasing number of connections and weights) and therefore the probability to find a local instead of the global minimum increase^[76]. For this there are little generally accepted procedures, how this problem can be solved.

4.5.1.2. Flat plateaus

Flat plateaus are a further problem of gradient methods. Since the size of the weight-change depends on the absolute value of the gradient, Back-propagation on flat plateaus stagnates, i.e. the learning procedure needs many iteration steps (Figure 30 ②). The learning procedure accomplishes no more weight changes on a completely flat plateau at all. With the introduction of the momentum term, the Back-propagation algorithm can overcome these flat plateaus (chapter 4.5.2.2).

4.5.1.3. Oscillation in narrow ravines

In steep ravines of the error surface, the learning process can oscillate (Figure 30 ③). This happens, if the gradient at the edge of a ravine is so large that via the weight change a jump on the opposite side of the ravine takes place. If the ravine there is just as steep, this causes a jump back to the starting position. Fortunately, the introduction of the momentum term can also absorb or damp the oscillation in steep ravines.

4.5.1.4. Leaving of good minima

If the learning rate (chapter 4.5.2.1) is too high, it can even occur that Back-propagation jumps out of a good into a sub-optimal minimum (Figure 30 ④). In practice, this happens very rarely.

4.5.2. Training with the Back-propagation learning method

The Back-propagation method (schematically shown in Figure 20), is a supervised learning method. Therefore it needs a set of pairs of objects (inputs \mathbf{X}_S , targets \mathbf{Y}_S). The weights are corrected to produce the specified target output for as many inputs as possible. The correction of weights is made after each individual new input. During learning, the input vector \mathbf{X} is presented to the network and the output vector **Out** is immediately compared with the target vector \mathbf{Y} , which is the correct output for the input \mathbf{X} . Once the error produced by the network is known, the weights will be adjusted accordingly.

The weight correction in the l -th layer is composed of two terms (Equation 14):

- The first term tends towards a fast steepest-descent convergence
- The second one is a long-range function that prevents the solution from being trapped in shallow local minima.

These two terms pull in opposite directions!

The weight correction is different in the output and hidden & input layer:

$$\delta_j^{last} = (y_j - out_j^{last}) out_j^{last} (1 - out_j^{last})$$

Equation 11: error of the output layer

$$\delta_j^l = \left(\sum_{k=1}^r \delta_k^{l+1} w_{kj}^{l+1} \right) out_j^l (1 - out_j^l)$$

Equation 12: error for all other layers (l=last -1 to 1)

Values from three layers influence the correction of weights in any one layer:

1. the output out_i^{l-1} of the layer above acting as the input i to the l -th layer
2. the out_j^l of the j -th neuron on the current layer l
3. the correction δ_k^{l+1} of the weight w_{kj}^{l+1} from layer $l+1$

$$\Delta w_{ji}^l = \eta \left(\sum_{k=1}^r \delta_k^{l+1} w_{kj}^{l+1} \right) out_j^l (1 - out_j^l) out_i^{l-1} + \mu \Delta w_{ji}^{l(\text{previous})}$$

Equation 13: final expression of the weight correction in a hidden layer

- l index of the current layer
- j current neuron
- i index of the input source (index of neuron in the upper layer)
- δ_j^l introduced error by the corresponding neuron
- η learning rate
- μ momentum term

$$\Delta w_{ji}^l = \eta \delta_j^l out_i^{l-1} + \mu \Delta w_{ji}^{l(\text{previous})}$$

Equation 14: final expression in condensed form

4.5.2.1. Learning rate η

The value of the learning rate determines only the speed at which the network attains a minimum on the criterion function (Equation 14), not the final weight values themselves. The step size is proportional to the slope (so that the algorithms settle down in a minimum) and to the learning rate. The correct setting for the learning rate is application-dependent, and is typically chosen by experiment; it may also be time-varying, getting smaller as the algorithm progresses ^[293].

The choice of the learning rate is decisive for the performance of the Back-propagation algorithm. Too large values of η cause strong jumps on the error surface and bring the risk of skipping narrow ravines and/or jumping out of them again. On the other hand, in the worst case the algorithm starts to oscillate. If the learning rate is too small, the amount of time needed to train the neural network is practically not acceptable. The choice of η depends primarily on the problem and the training data and in addition on the size and topology of the network. Therefore, it is not possible to choose the learning rate correctly in advance. The only way is to determine it experimentally.

4.5.2.2. Momentum term μ

As shown in Figure 30, error surfaces often have plateaus - regions in which the slope is very small. These can arise when there are too many weights and thus the error depends only weakly upon any one of them. The momentum term allows the network to learn more quickly when plateaus in the error surface exist. The approaches to alter the learning rule to include some fraction of the previous weight update ^[52]. Obviously, μ should not be negative and for stability μ must be less than 1.0 (Equation 14). If $\mu=1$, the change suggested by Back-propagation is ignored, and the weight vector moves with constant velocity ^[128].

4.5.3. Self organizing feature maps (SOM)

The self organizing maps (also known after their developer as Kohonen maps or Kohonen feature maps ^[79]) concern a special neural network, which is organized without teachers. As explained above in chapter 3.3.3, these teachers compare the produced output with the target output and adapt the weights if necessary. Whereas in supervised learning the training data set only contains cases featuring input variables together with the associated target outputs (and the network must infer a mapping from the inputs to the outputs), in unsupervised learning the training pattern file only contains input variables. At first glance, this may seem strange. Without outputs, what can the network learn? The answer is that the SOM network attempts to learn the structure of the data.

One possible use is therefore in exploratory data analysis. The SOM network can learn to recognize clusters of data, and can also relate similar classes to each other. The user can build up an understanding of the data, which is used to refine the network. As classes of data are recognized, they are labeled, so that the network becomes capable of classification tasks. SOM networks can also be used for classification when output classes are immediately available - the advantage in this case is their ability to highlight similarities between classes. A second possible use is in novelty detection. SOM networks can learn to recognize clusters in the training data, and respond to it. If new unseen data, unlike previous cases, is encountered, the network fails to recognize it and this indicates novelty.

A SOM network has only two layers: the input layer, and an output layer (also known as the topological map layer). The units in the topological map layer are laid out in space - typically in two dimensions.

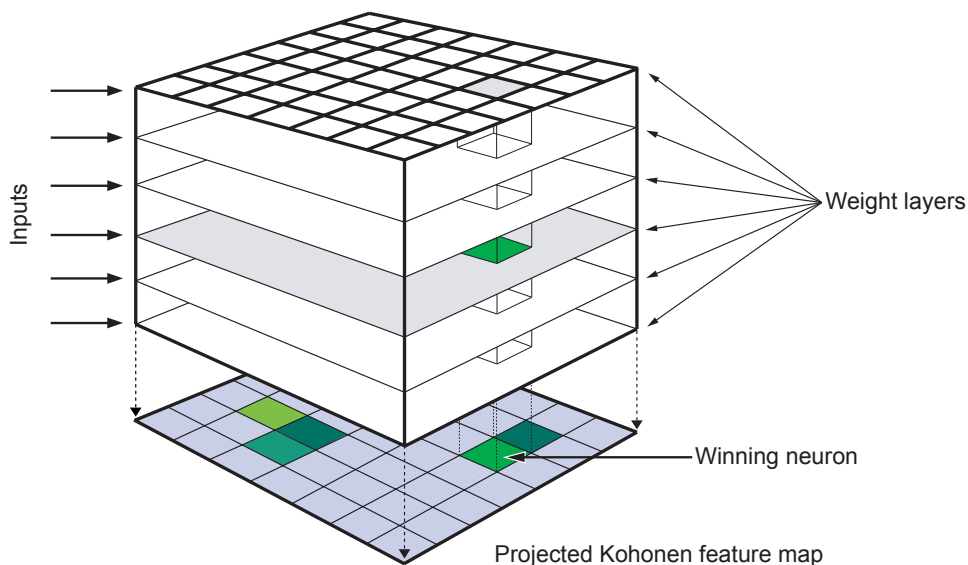


Figure 31: An illustration of a sample Kohonen feature map (49 neurons forming a 7x7x6 network) represented as a block containing neurons as columns and weights (line intersections) in levels (adapted from ^[294]).

The levels of weights are superimposed onto each other in a one-to-one-correspondence, hence the weights of each neuron are obtained by looking at the weights in all levels that are exactly aligned in a vertical column (from the top in (Figure 31)). There are as many weight levels in each Kohonen network as there are input variables describing the objects for which the network is designed.

SOM networks are trained using an iterative algorithm. Starting with an initially random set of weights, the algorithm gradually adjusts them to reflect the clustering of the training data. The iterative Kohonen training procedure tries to map the input so that similar signals excite neurons that are very close together on the topological map (in terms of spatial distance). You can think of the network's topological layer as a crude two-dimensional grid, which must be folded and distorted into the N-dimensional input space to preserve the original structure as far as possible. Clearly any attempts to represent an N-dimensional space in two dimensions will result in loss of detail; however, the technique can be worthwhile in allowing to visualize data that might otherwise be impossible to understand ^[295].

The basic iterative Kohonen algorithm simply runs through a number of epochs. During each epoch, the corresponding training case passes through the following steps:

1. The responses of all neurons are calculated.
2. The winning neuron is selected (the one whose center is nearest to the input case); no matter how close the other neurons are to this best one, they are left out of that cycle. This is also referred to as the winner takes all method ^[232].
3. After finding the neuron that best satisfies the selected training case, its weights are corrected to make its response larger and/or closer to the desired one.

- The weights of the arbitrarily defined neighboring neurons are corrected as well. These corrections are usually scaled down, depending on the distance from the winning neuron. For this reason, the scaling function is called a topology dependent function^[232].

The algorithm uses a time-decaying learning rate, which is used to perform the weighted sum and ensures that the alterations become subtler as the epochs pass. This ensures that the centers settle down to a compromise representation of the cases that cause that neuron to win.

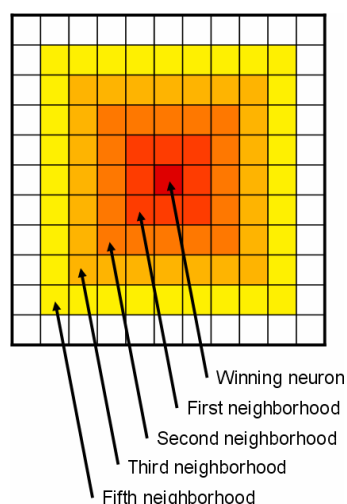


Figure 32: Illustration of square neighborhoods (adapted from J. Zupan and J. Gasteiger)

The topological ordering property is achieved by adding the concept of a neighborhood to the algorithm. The neighborhood is a set of neurons surrounding the winning neuron (Figure 32). The neighborhood, like the learning rate, decays over time, so that initially quite a large number of neurons belong to the neighborhood (perhaps almost the entire topological map); in the latter stages the neighborhood will be zero (i.e., consists solely of the winning neuron itself). In the Kohonen algorithm, the adjustment of neurons is actually applied to all the members of the current neighborhood, not just to the winning neuron.

The effect of this neighborhood update is that initially quite large areas of the network are dragged towards training cases - and dragged quite substantially. The network develops a crude topological ordering, with similar cases activating clumps of neurons in the topological map (Figure 35). As epochs pass, the learning rate and neighborhood both decrease, so that finer distinctions within areas of the map can be drawn, ultimately resulting in fine-tuning of individual neurons. Typically, training is deliberately conducted in two distinct phases: a relatively short phase with high learning rates and neighborhood, and a long phase with low learning rate and zero or near-zero neighborhoods.

Once the network has been trained to recognize structure in the data, it can be used as a visualization tool to examine the data. Win Frequencies (counts of the number of times each neuron wins when training cases are executed) can be examined to see if distinct clusters have formed on the map (Figure 33). Individual cases are executed and the topological map observed (Figure 35), to see if some meaning can be assigned to the clusters (this usually involves referring back to the

original application area, so that the relationship between clustered cases can be established). Once clusters are identified, neurons in the topological map are labeled (Figure 34) to indicate their meaning (sometimes individual cases may be labeled, too). Once the topological map has been built up in this way, new cases can be submitted to the network. If the winning neuron has been labeled with a class name, the network can perform classification. If not, the network is regarded as undecided.

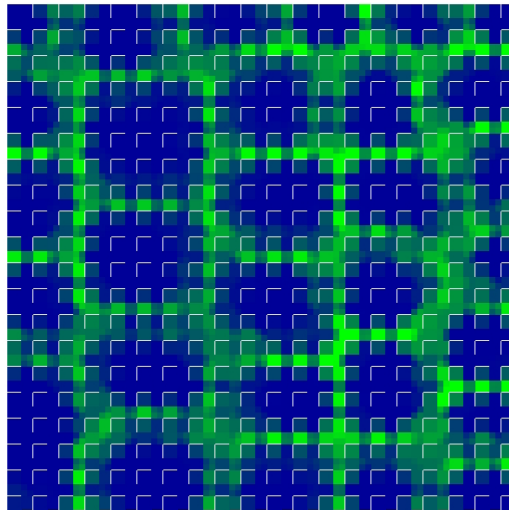


Figure 33: Sample Kohonen topological map (Euclidian distance between classes) trained with 30 different glucose monosaccharide classes after 20'000 learning cycles.

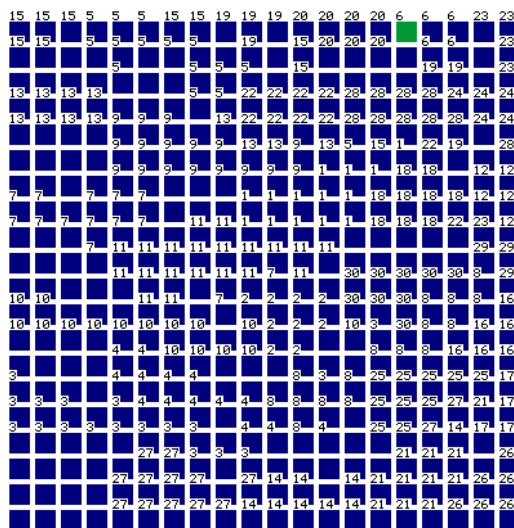


Figure 34: Winning neurons for each class after 10'000 learning cycles of the same network as illustrated in Figure 33

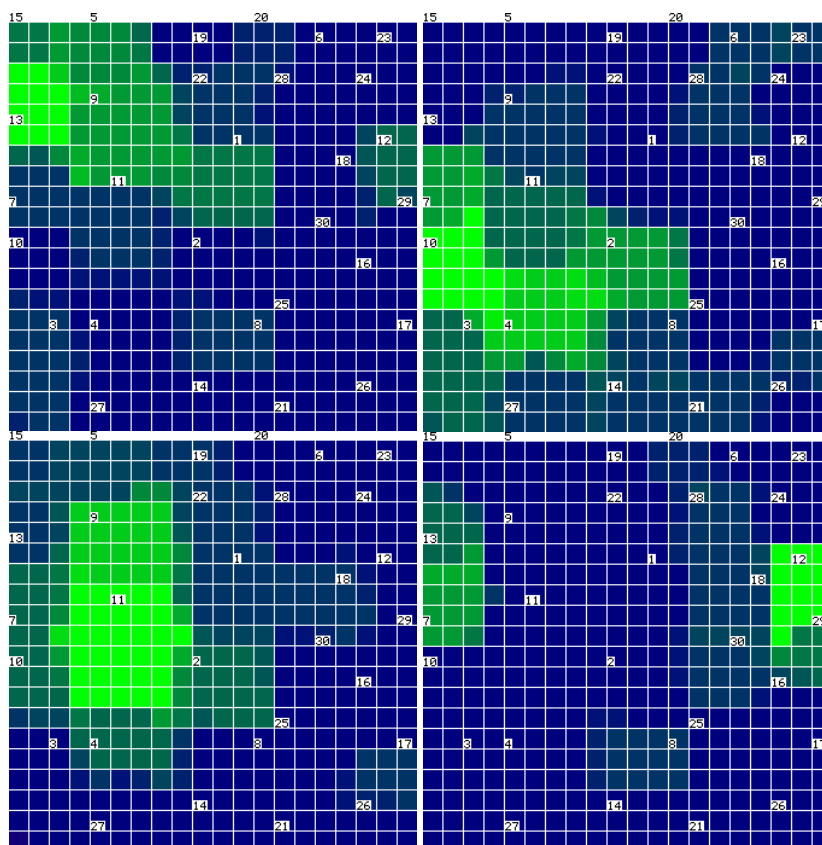


Figure 35: Graphical location of four different glucose monosaccharide residues (top left: α -D-Glcp-OMe-2R, top right: α -D-Glcp-OMe-3R, bottom left: α -D-Glcp-OMe-4R and bottom right: α -D-Glcp-OMe-6R). Light green areas indicate high, darker green regions indicate weaker similarity.

SOM networks also make use of an accept threshold, when performing classification. Since the activation level of a neuron in a SOM network is the distance of the neuron from the input case, the accept threshold acts as a maximum recognized distance. If the activation of the winning neuron is greater than this distance, the SOM network is regarded as undecided. Thus, by labeling all neurons and setting the accept threshold appropriately, a SOM network can act as a novelty detector (it reports undecided only if the input case is sufficiently dissimilar to all radial units).

SOM networks are inspired by some known properties of the brain. The cerebral cortex is actually a large flat sheet (about 0.5m squared; it is folded up into the familiar convoluted shape only for convenience in fitting into the skull) with known topological properties (for example, the area corresponding to the hand is next to the arm, and a distorted human frame can be topologically mapped out in two dimensions on its surface) ^[295].

4.5.4. Counter-propagation Network

The Counter-propagation network is a real multilayer network with more than just an input and an output layer. It consists of a Kohonen layer (as explained in chapter 4.5.3) followed by a fully connected Grossberg layer. The Counter-propagation network is trained with supervised competitive learning – the training process needs defined input and target output pairs.

The Grossberg layer acts as a kind of visualization and classification aid for the output of the Kohonen layer. Once the Counter-propagation network is trained, each time a winner neuron in the Kohonen layer is selected, the corresponding output neuron of the Grossberg layer is activated. The user now only sees to which class his input pattern belongs to and no longer the geographical region on the Kohonen feature map. This information is hidden.

Counter-propagation networks are best used to generate lookup tables, where all the required answers are known in advance. The network learns to build the Kohonen map and in the same step to connect all the neurons of a cluster on the Kohonen layer with its corresponding target output neuron of the Grossberg layer.

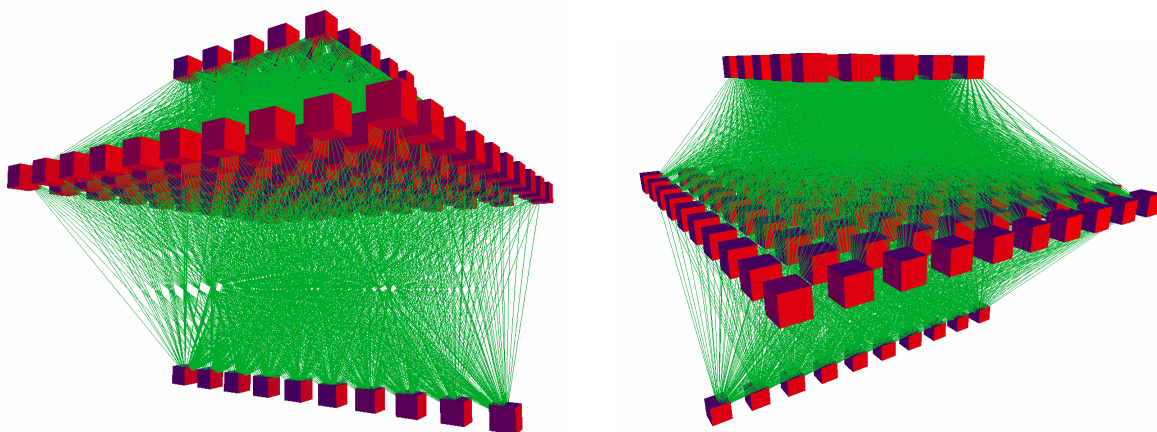


Figure 36: Fully connected sample Counter-propagation network in SNNS 3D illustration. Input Units on top, Kohonen layer in the middle and the Grossberg layer at the bottom.

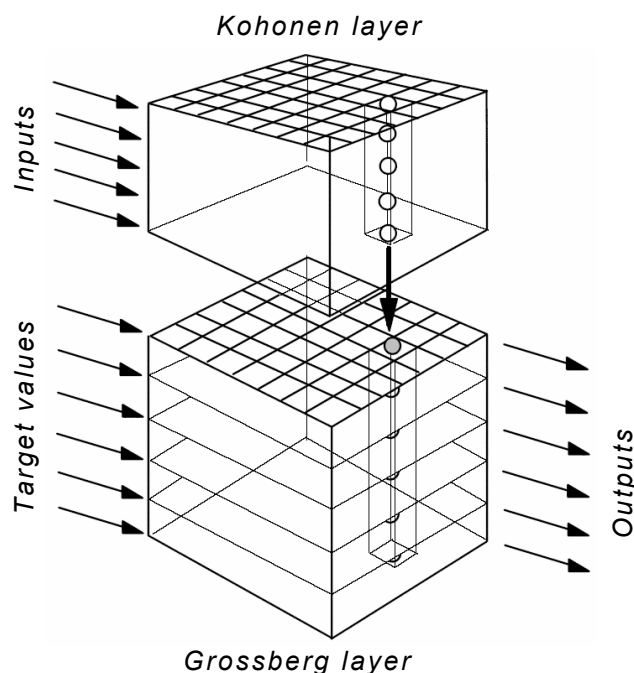


Figure 37: An illustration of a sample Counter-propagation network. On top the Kohonen layer and at the bottom the Grossberg ^[53-55] layer (adapted from ^[294]).

During a learning cycle (epoch) the following steps are executed ^[296]:

1. The responses of all neurons are calculated.
2. The winning neuron is selected (the one whose center is nearest to the input case)
3. After finding the neuron that best satisfies the selected training case, its weights are corrected to make its response larger and/or closer to the desired one.
4. The output at the Grossberg layer is computed and compared to the target output.
5. **Only** the weights between the winner and the output layer are updated. The weights in the Kohonen layer are **not** adapted.

It is hard to formalize the types of predictions which can be accomplished by a Counter-propagation network. They can be of very different types. The simplest are those classifying multidimensional objects into proper categories like NMR spectra into monosaccharide units. More complex predictions involve content-dependent retrievals, where incomplete or fuzzy data are entered and the originals are recovered. Therefore, this type of network was used for the identification of monosaccharide units. Also incomplete input pattern (e.g. peaks missing) lead to a correct classification (chapter 5.5).

The problem with the Counter-propagation network is that it needs large quantities of training data covering all possible answers. The number of different classes the network can distinguish is limited by the size of the network ^[232].

4.6. Error functions

The central goal in network training is not to memorize the training data but rather to model the underlying generator of the data, so that the best possible predictions for the output vector can be made when the trained network is subsequently presented with a new value for the input vector ^[77]. However, to direct this process in the right direction we need some penalty criteria – the error functions. Based on their output the neural network training algorithm can determine his position on the error surface and take the necessary steps to reduce the error.

4.6.1. The Sum-of-squares error (SSE)

The Sum-of-squares error is the sum over the output units of the squared difference between the desired output t_k given by a teacher and the actual output z_k :

$$J(w) \equiv \frac{1}{2} \sum_{k=1}^c (t_k - z_k)^2 = \frac{1}{2} \|t - z\|^2 \quad \text{Equation 15: The sum-of-squares error function}$$

Where t and z are the target and the network output vectors of length c and w represents all the weights in the network.

This is the standard error function used in regression problems. It can also be used for classification problems, giving robust performance in estimating discriminant functions, although arguably entropy functions are more appropriate for classification (on the assumption that the generating distribution is drawn from the exponential family), and allow outputs to be interpreted as probabilities.

4.6.2. Mean squared error (MSE)

The mean squared error is equal the SSE divided by the number of patterns (training or test cases).

4.6.3. Cross entropy

The following formula is applied when one is dealing with a conventional classification problem involving mutually exclusive classes (number of classes is greater than two). One output for each class appearing in the input pattern file with the coding scheme 1-of- c and a winner-takes-all activation model (the unit with the largest input has output 1 while all other units have output 0).

$$J_{ce}(w) = \sum_{m=1}^n \sum_{k=1}^c t_{mk} \ln(t_{mk} / z_{mk}) \quad \text{Equation 16: The cross entropy error function}$$

Where n = number of patterns, t_{mk} = target output of unit k for pattern m and z_{mk} = actual output of unit k for pattern m

4.7. Modification generator (MG)

The MG is a VBA Excel macro programmed by Andreas Stöckli. The main purpose of the macro is the artificial inflation of the NMR training peak data. Since certain monosaccharide moieties are under-represented in the database, it is necessary to generate further artificial NMR peaks from these sugars. From each monosaccharide moiety included in the NMR FileMaker database, the program calculates the mean value and the standard error. With these numbers, the macro generates a user requested quantity of artificial modifications of a certain monosaccharide moiety. The mean values of each peak are then randomly shifted upward and downward in the range of the standard deviation. With the MG it is possible to generate an equalized data basis for the training and recognition process of a neural network.

4.8. Used neural network simulation software

4.8.1. Statsoft Statistica [293]

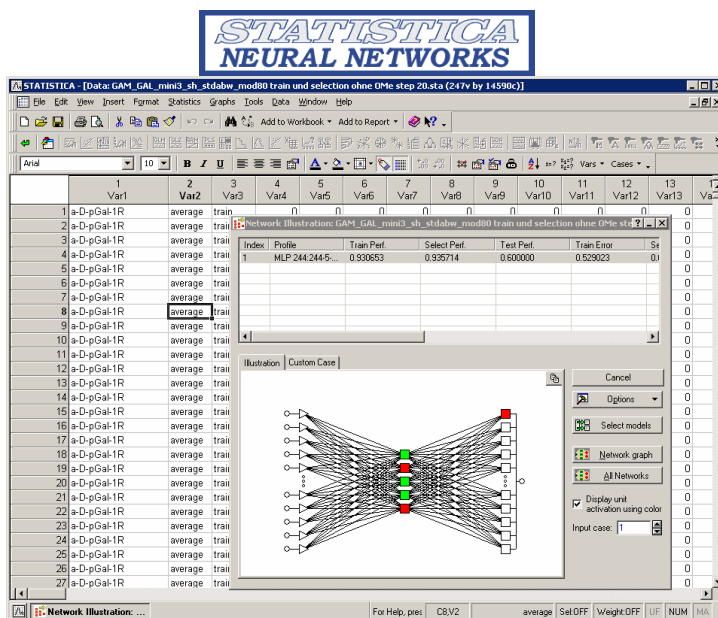


Figure 38: Statsoft Statistica main working area

Statistica is a suite of analytics software products and provides an array of data analysis, data management, data visualization and data mining procedures. Its techniques include a wide selection of predictive modeling, clustering, classification and exploratory techniques in one software platform.

The subprogram for neural networks includes many architectures and algorithms. like regression, classification, time series, cluster analysis and feature selection; MLP, RBF, PNN, SOM, linear, PCA, cluster networks and ensembles; Back-propagation, conjugate gradient descent, quasi-newton, Levenberg-Marquardt, quick-propagation, delta-bar-delta, LVQ, PCA, pruning and feature selection algorithms (including forward & backward selection and genetic algorithms).

As input data for a neural network the program accepts almost any imaginable table data format used today. The most applicable format in connection with the ANN Pattern File Generator is a

comma separated value (CSV) file in ASCII format (chapter 4.9.2.2 and 4.9.3.1). These files are not compressed and can be edited with a multitude of ASCII editors or spreadsheet programs like Microsoft Excel. It is also possible to read the training data directly out of a database with an ODBC or MySQL interface or from an online web form over the internet.

4.8.2. Stuttgart Neural Network Simulator (SNNS) V.4.2 ^[297]

The Stuttgart Neural Network Simulator is a simulator for neural networks developed at the Institute for Parallel and Distributed High Performance Systems (Institut für Parallele und Verteilte Höchstleistungsrechner, IPVR) at the University of Stuttgart since 1989.

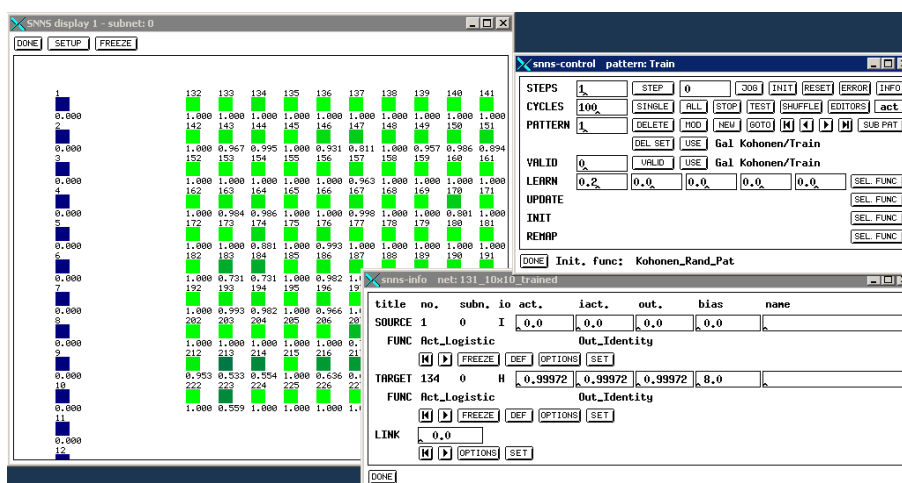


Figure 39: SNNS V.4.2 working area

The SNNS simulator consists of four main components that are depicted in Figure 40: Simulator kernel, graphical user interface, batch execution interface batchman, and network compiler snns2c. The simulator kernel operates on the internal network data structures of the neural networks and performs all operations on them. The graphical user interface XGUI1, built on top of the kernel, gives a graphical representation of the neural networks and controls the kernel during the simulation run. In addition, the user interface can be used to directly create, manipulate and visualize neural networks in various ways. Complex networks can be created quickly and easily.

SNNS is implemented completely in ANSI-C. During this thesis the kernel was compiled under RedHat Linux 7.1, and 7.2. A precompiled version of kernel V.4.2 was used to run simulations in a Microsoft Windows environment like Windows 2000 and Windows XP.

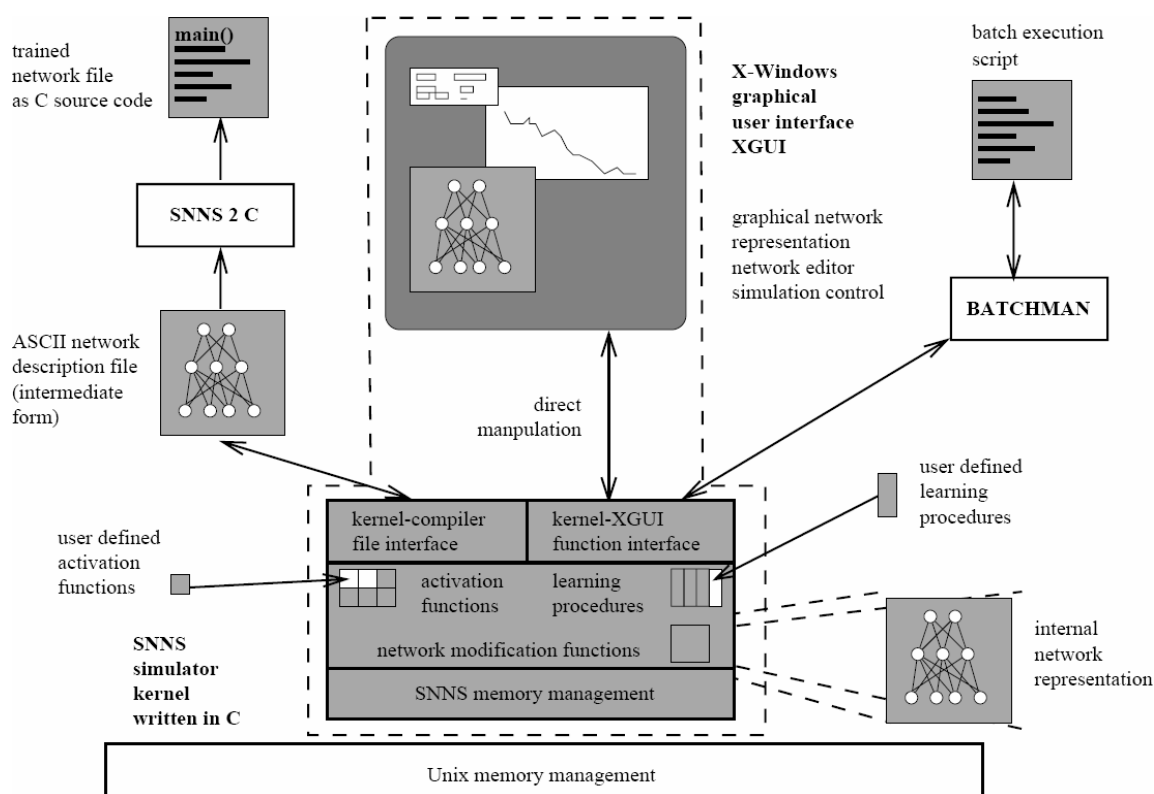


Figure 40: SNNs components: simulator kernel, graphical user interface xgui, batchman and network compiler ssn2c (adapted from ^[296])

In SNNs the following architectures and learning procedures are included ^[296]:

- Back-propagation (BP) for feedforward networks
- Counter-propagation
- Quickprop
- Backpercolation 1
- RProp
- Generalized radial basis functions (RBF)
- ART1
- ART2
- ARTMAP
- Cascade Correlation
- Recurrent Cascade Correlation
- Back-propagation through time (for recurrent networks)
- Quickprop through time (for recurrent networks)
- Self-organizing maps (Kohonen maps)
- TDNN (time-delay networks) with Back-propagation
- Jordan networks
- Elman networks and extended hierarchical Elman networks
- Associative Memory

As input data, the program only accepts a specific predefined pattern file format explained in detail in chapter 4.9.3.2.

4.8.3. Java Neural Network Simulator (JavaNNS) V.1.1 ^[298]

JavaNNS is the Java implementation with almost the same features as SNNS. It is based on the SNNS kernel V.4.2. The required input data format of the pattern files is the same as for SNNS (chapter 4.9.3.2).

JavaNNS was used because of the good visualization tools and Java applications can be executed on almost any operating system like UNIX, Linux, Windows and Mac OSX. Therefore, the calculated neural networks and their recognition results can be compared among the operating systems.

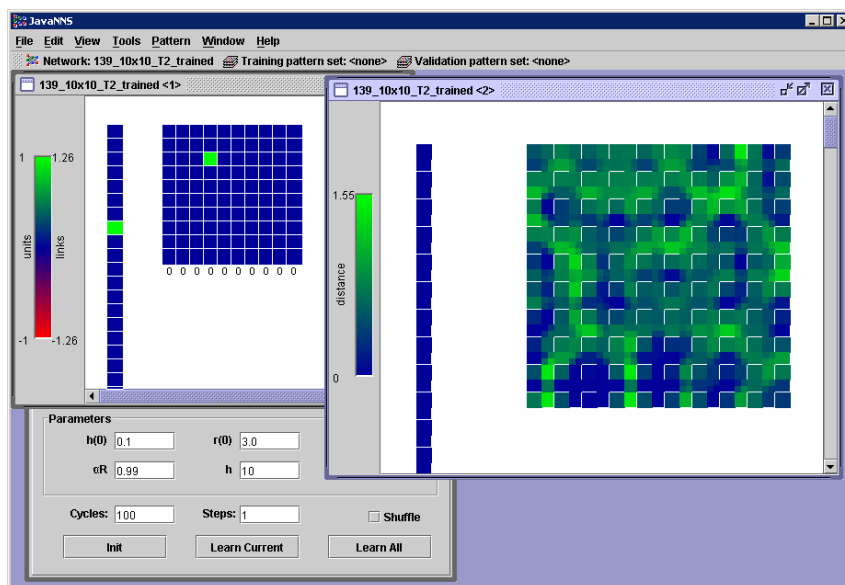


Figure 41: JavaNNS V.1.1 main window

4.9. ANN PFG (Pattern File Generator)

4.9.1. Introduction / Summary

The ANN NMR pattern file generator (or just PFG) is a powerful and multifunctional software tool developed and programmed in Microsoft Visual Basic 6.0 during this PhD thesis. The main goal of the program is the conversion and compression of any kind of JCAMP-DX (but specially JCAMP-DX for NMR (^{13}C and ^1H -NMR)) file or a predefined CSV file containing NMR peak lists (chapter 4.9.2.2) into an every time intimately pattern file. The conversion is executed with user-defined parameters and in later versions a custom peak mask (chapter 4.9.6.2). All processing parameters can be saved in a configuration file. The output pattern file is an ASCII text file containing the input pattern presented to the input neurons of the neural network in a compressed form. For Back-propagation networks, the file also contains the corresponding teaching output patterns. With the ANN PFG it is possible to provide an absolutely identical pattern for each use.

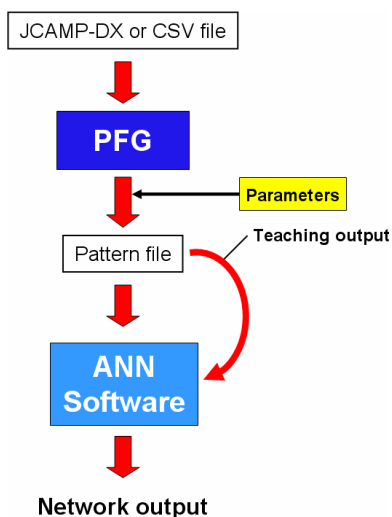


Figure 42: ANN PFG - coarse data flow

Another reason for the development of this software was the immense amount of ^{13}C -NMR data one had to deal with. An exported JCAMP-DX NMR spectrum normally contains 32k (32768) data points (chapter 4.4) depending on the resolution and the dimensions of the spectrum (1D or 2D) at hand. Feeding uncompressed 32k NMR data points into a neural network would lead to a network with 32768 input units. Already the fact that a carbohydrate ^{13}C -NMR spectrum only contains about 1% peak values, makes data compression inevitable. To work with the 99% non-peak data with no information content is needless and would increase the amount of training data in no relation. Not speaking of the immense amount of computer training time it takes to train neural networks with several million weights. A major problem with networks with too many degrees of freedom (weights) is over-fitting and the resulting lack of generalization capability ^[299-301]. Therefore, the used neural network should be as small as possible. An algorithm capable of distinguishing between peak and non-peak data and data conversion into a format suitable as an input for neural networks had to be

implemented into the PFG. The three different major releases (V.0.1, 0.2 and 0.9) of the software use different approaches to fulfill these needs.

In version 0.1 there is no proper data reduction algorithm implemented – instead the only way of data reduction is the definition of a region of interest; a starting ppm-value and an end ppm-value (Figure 43). The fixed (hard coded) scaled intensity (from 0 to 1) of each measuring point in this defined region is then feed directly into the corresponding input neuron of the network to train (Figure 44). The resulting network has as many input units as there are data points in the defined ppm range. But with this method regions with no peak data will be included nevertheless.

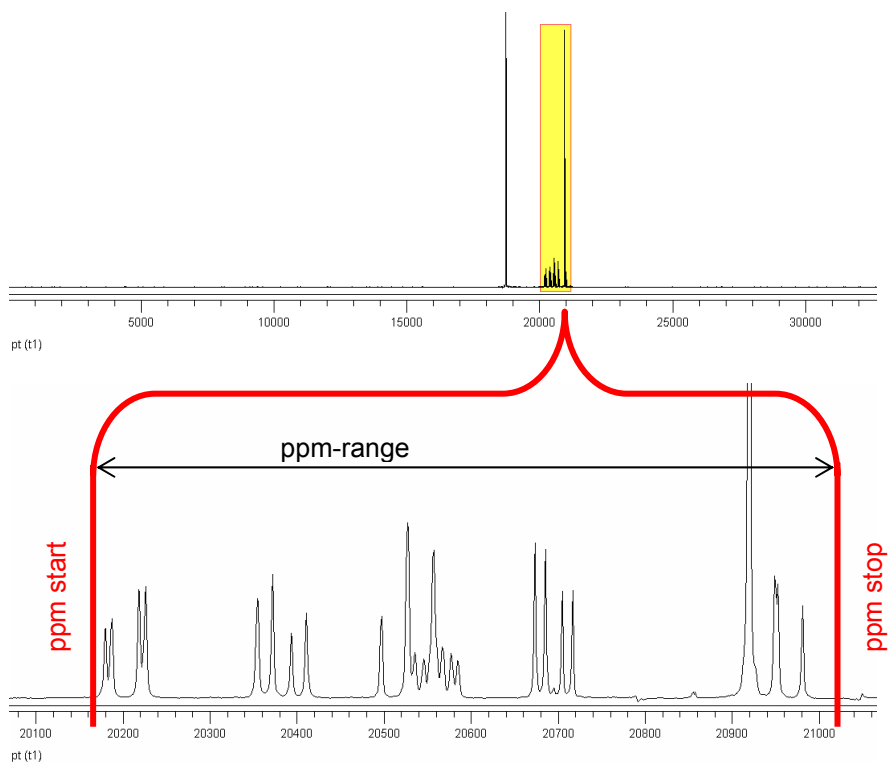


Figure 43: Illustration of data reduction in PFG V.0.1 and partly from V.0.2

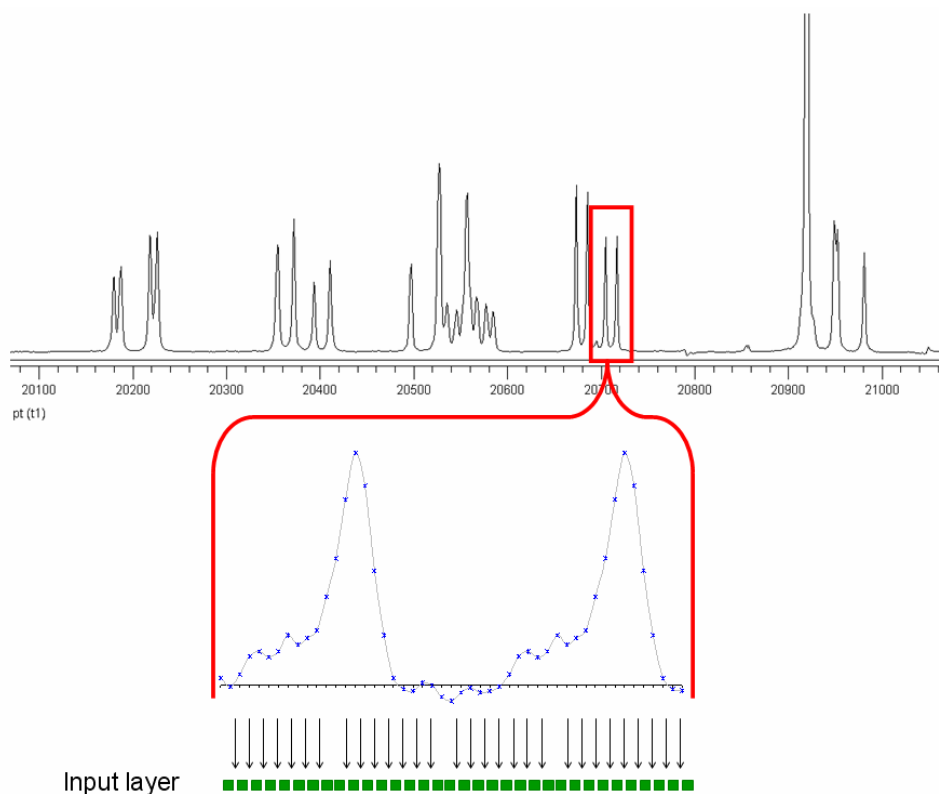


Figure 44: Input data flow illustration; how ^1H -NMR data enters the neural network

In version 0.2 the data reduction approach from version 0.1 was retained but extended with an individual adjustable reading step size – only every x^{th} data point is processed and the missed out data points are not used as input to the final input pattern for the neural network. The biggest danger of this approach is to read over important peaks if the step size is bigger than the dimension of a peak in a JCAMP-DX file. A normal ^1H peak allocates approximately 20 points in a JCAMP-DX file. A peak in a ^{13}C -NMR file allocates approximately 10 data points. Therefore, the step size has to be adjusted according to the underlying type of data. To avoid this problem version 0.2 was extended with a new feature: the *block-patterns*. In this approach, the software checks in the original JCAMP-DX input file after every reading step it, if a peak was missed out. If there was a peak in the gap exceeding the threshold (Figure 56) the algorithm will include the signal in the generated pattern. For details, see description of the algorithm in chapter 4.9.5.

In the latest stable version of the PFG (V.0.9), a different approach was chosen. The program still works with a starting ppm-value and an end ppm-value but the values are determined automatically by reading all the NMR input data in advance. In a second step another algorithm is superimposing all input NMR spectra and generates a so-called *peak mask* of all used data points in the ppm-range. Data points where there is no input level exceeding the threshold level (Figure 57 and Figure 58) are not included in the final pattern file. The idea of the peak mask has also the advantage of filtering out uninteresting peak regions. If the peak mask was generated, e.g. only out of glucose data then the algorithm will not care about peaks in the region of 17 – 19 ppm if a rhamnose is presented. The peak mask serves as a kind of primitive input filter. Details are explained in chapter 4.9.6.

Because the used neural networks cannot change their size (number of neurons in the different layers) during the training or the test phase, all input and output vectors of the pattern file have to be kept at a fixed dimensionality. This task is taken over by the ANN Pattern File Generator in advance during the data input data processing. Also for later predictions based on an earlier trained neural network, the input dimensionality of the test pattern has to match exactly the number of input and output units provided by the used network.

The ANN PFG was developed in three major and independent releases suited for the particular problems one had to deal with in each stage of this thesis. The different not already explained releases will be discussed and outlined in two separated chapters. The detailed algorithms will only be discussed in the final version 0.9, as they are quite similar in each version of the program.

4.9.2. Input file formats

An input file contains the data one wants a neural network to learn from or make predictions from. For training purposes, an input file also has to contain the corresponding teaching output. As input files, the different versions of the ANN PFG accept two fundamental different input file types.

- JCAMP-DX for NMR files
- CSV Files (Comma-separated values file)

4.9.2.1. JCAMP-DX for NMR files

This file format is discussed and explained in chapter 4.4

4.9.2.2. CSV Files

A CSV file contains the values in a table as a series of ASCII text lines organized so that each column value is separated by a predefined delimiter (freely selectable in the PFG) from the next column's value and each row starts a new line.

A CSV file is a way to collect the data from any table so that it can be conveyed as input to another table-oriented application such as a relational database application. Microsoft Excel, a leading spreadsheet or relational database application, can read (and write) CSV files.

To serve as an input file for the ANN PFG V.0.9 the CVS file must keep the file structure according to the following defaults:

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
Data origin (NMR, FileMaker, Literature ...)	Compound name	Subset (Train, Test or Selection)	Peak 1 [ppm]	Peak 2 [ppm]	Peak 3 [ppm]	Peak x [ppm]

Each compound has to start on a new line. The peaks starting at Column 4 do not have to be sorted. The ANN PFG will bubble sort them internally. The peak list should be filled up with 0-peaks – as many 0-peaks as there are peak values in the longest record in the whole table.

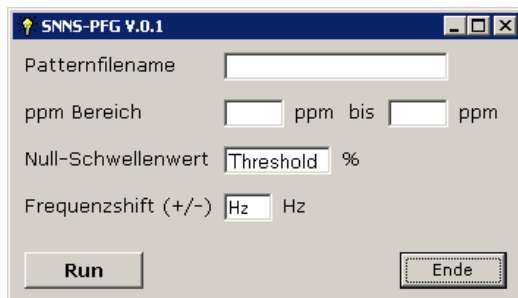


Figure 48: First generation SNNS-PFG GUI

To increase and artificially multiply the input data, the first version of the program was able to generate artificial modifications of the original JCAMP-DX files. Thereto the desired modification can be coded directly in the filename of the JCAMP-DX file. It is also possible to combine different reasonable modifications.

The following codes (and their combinations) in the input file name are accepted:

Table 10: Modification codes and their explanation

Code	explanation
<i>A</i>	no variation
<i>B</i>	halve spectrum intensity
<i>C</i>	add Gaussian noise (40dB)
<i>D</i>	add Gaussian noise (60dB)
<i>E</i>	right shift whole spectrum (+1Hz)
<i>F</i>	left shift whole spectrum (-1Hz)

Table 11: Some example input file names

File name	explanation
<i>1_a.dx</i>	Compound 1 with no changes
<i>2_b.dx</i>	Compound 2 is halved in its peak intensity
<i>4_de.dx</i>	Compound 4 is shifted to the right (+1Hz) and 60dB noise is added
<i>5_bf.dx</i>	Compound 5 is shifted to the left (-1Hz) and reduced in its intensity by 50%

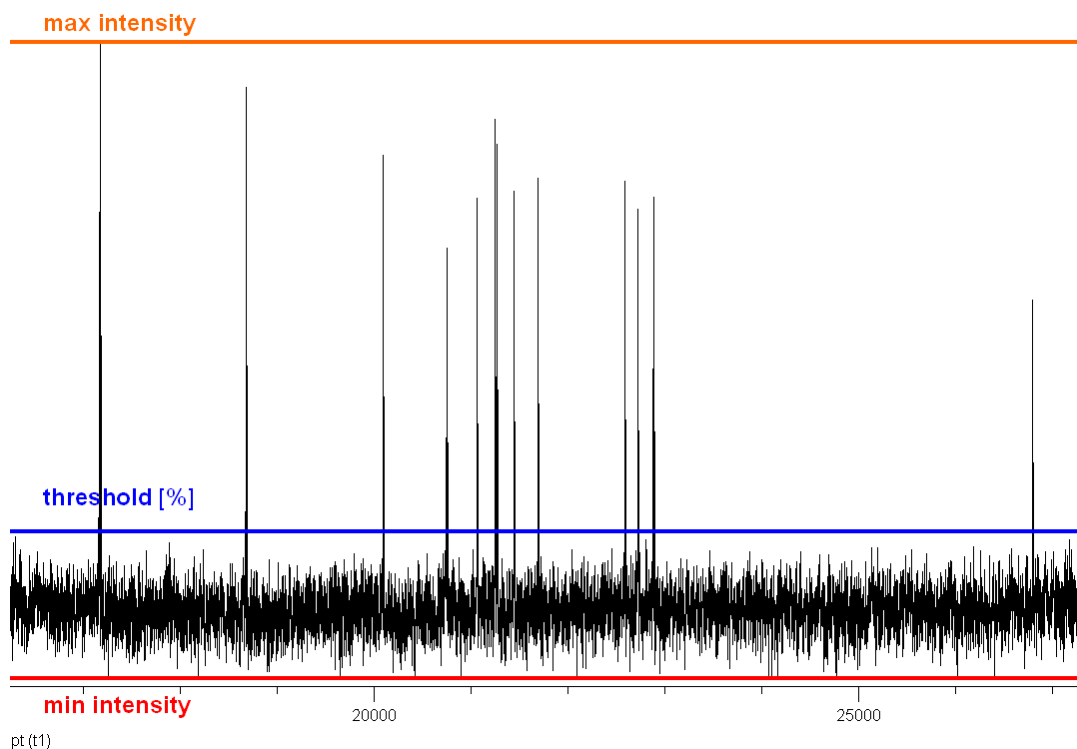


Figure 49: Noise threshold [%] and max / min intensity parameter by the example of a disaccharide ^{13}C -NMR spectrum.

4.9.5. SNNS PFG V.0.2

After all, PFG version 0.2 is actually based on the same ideas and algorithms as version 0.1. These basic features are discussed in chapter 4.9.1. A new version of the PFG was necessary because the training/test dataset was expanded with ^{13}C -NMR peak lists from literature. Because these peak lists are not available as JCAMP-DX files, a subprogram to generate JCAMP-DX out of normal ^{13}C -NMR peak lists was implemented in this version (chapter 4.9.5.1). The training pattern files of version 0.1 still had too many inputs and for this reason, the training and generalization results of the tested networks were absolutely unsatisfying (chapter 5.3). Therefore, the compression algorithm had to be improved and extended (chapter 4.9.5.3). Several new features like SNNS Kohonen input pattern, binary input pattern and pattern files in CSV file format were also implemented.

4.9.5.1. JCAMP-DX file generator subprogram

The JCAMP-DX file generator subprogram converts peak lists of (literature) ^1H or ^{13}C -NMR spectra back into JCAMP-DX for NMR files. The subprogram also offers the possibility to generate user defined artificial modifications of the original files and add a custom percentage of random Gaussian noise to the data. The JCAMP-DX files can be generated each with different resolutions (8k, 16k and 32k). It is also possible to split up the generated modifications into different directories for test and training cases.

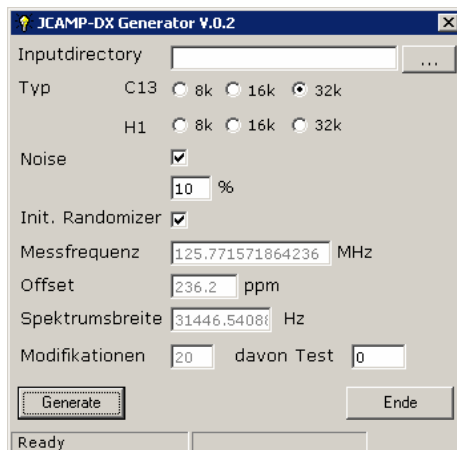
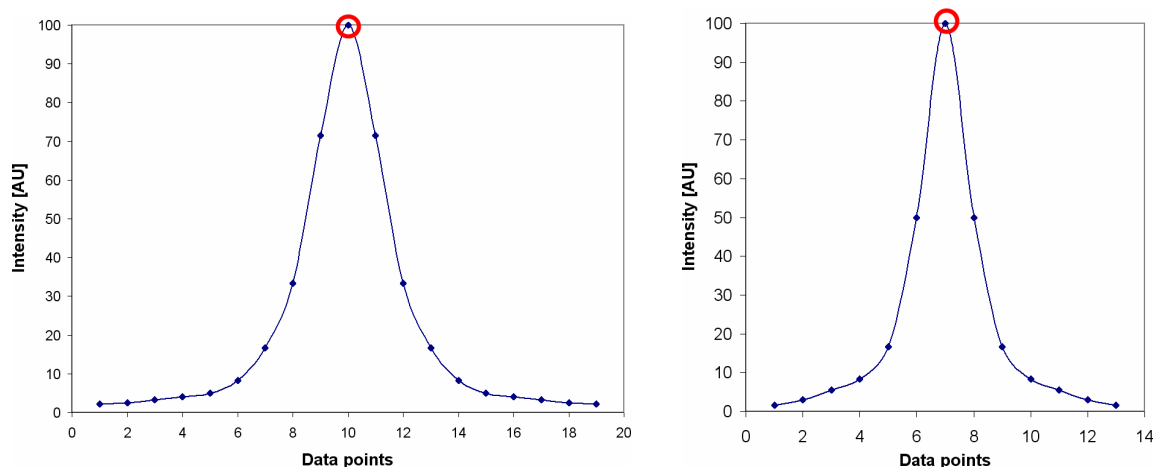


Figure 50: JCAMP-DX file generator V.0.2 GUI

To model and simulate ^1H and ^{13}C -NMR peaks in the output JCAMP-DX file, about 100 peaks of each type were isolated from real NMR spectra. The individual data points were averaged to gain an average shape of each peak type (Figure 51). This information was then hard coded into the source code of the JCAMP-DX file generator subprogram. The arbitrary intensity value of 100 is assigned to the original data point in the input peak list. This point represents the symmetrical center of the peak. Real NMR spectra normally contain slightly unsymmetrical peaks but the JCAMP-DX file generator does not simulate this to simplify matters.

Figure 51: Detail shape view of simulated NMR peaks (left graph: ^1H -NMR, right graph: ^{13}C -NMR). The **red circle** marks the original data point in the input peaklist (now the symmetrical center of the peak).

As input, the JCAMP-DX file generator accepts peak lists in the CSV file format. The compound name is the same as the filename and will be used as output file name with the DX file extension. E.g., the CSV input file *a-D-Manp-1R.csv* becomes *a-D-Manp-1R_0001.dx* whereas a four-digit suffix will be attached to distinguish possible custom modifications.

```
101.06;73.42;71.42;71.22;67.26;61.50;0.00
```

Figure 52: Sample file content of a-D-Manp-1R.csv

The resulting output JCAMP-DX file only contains the LDRs who are important for later processing in other subroutines in the PFG (chapter 4.4.4). The JCAMP-DX files cannot be reimported into a NMR software suite like Bruker XWIN-NMR because spectrometer specific LDRs can no be artificially created. The JCAMP-DX generator in its final version is able to convert 50'000 JCAMP-DX files at most.

4.9.5.2. Main pattern file generation subprogram

The main subprogram is responsible for the actual training or test pattern creation out of JCAMP-DX files (generated with the JCAMP-DX file generator or real JCAMP-DX files from NMRs). Features like the user definable input range (in ppm) and the zero-level threshold were taken over from version 0.1.

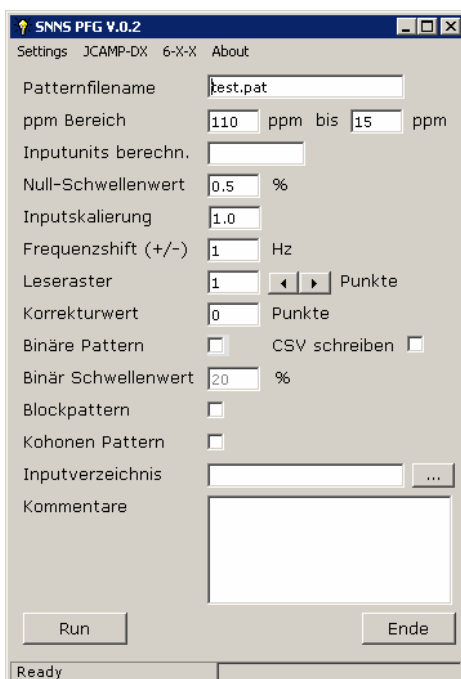


Figure 53: SNNS PFG V.0.2 main subprogram GUI

New features introduced with version 0.2 are:

- A user-defined input intensity scaling factor (default value 1); the largest appearing peak intensity in all input JCAMP-DX files is assigned to this value.
- A user-defined frequency shift for desired artificial modifications
- A user-defined reading step size for a more efficient data compression. This value defines the number of data points the reading subroutine skips in each reading cycle (Figure 54). The biggest problem of this compression approach is the possibility that the subroutine skips complete peaks if the step size is bigger than expansion of a single NMR peak (second reading step in Figure 54). To solve this problem for larger step sizes, the so called *block-pattern* approach was introduced (this approach is discussed in greater detail in chapter 4.9.5.3).

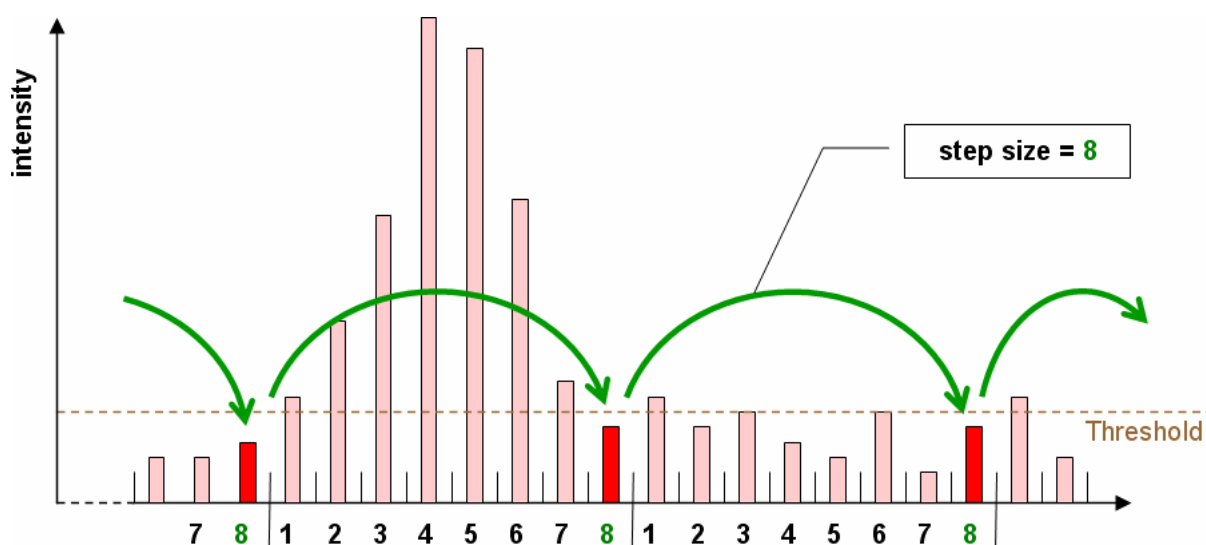


Figure 54: Explanation of the variable step size approach – only the peaks marked in red are processed and taken over into the final output pattern file. Peaks colored in pink will not be processed.

- A partly solution for the peak read-over problem is the user-defined reading offset. This value shifts the reading grid rightwards.
- Version 0.2 allows also the creation of binary pattern files. Thereto all intensities exceeding the binary threshold level are represented by '1'. Intensities below the binary threshold are assigned '0'.
- If desired version 0.2 can write Kohonen SNNS pattern files (as described in chapter 4.9.3.2)
- If the JCAMP-DX input files are used to train a Back-propagation network, the necessary target output patterns will be retrieved from an external CSV file called *output.csv*. This file acts as a kind of lookup table for the PFG. After every processed input JCAMP-DX file, the subroutine browses through the *output.csv* file and copies the corresponding output pattern into the pattern file.

The fact that this file is not hard coded into the source code makes it easier to insert new compounds into the pattern files.

The *output.csv* file has the following structure:

```

12
a-D-Glc-1R;1 0 0 0 0 0 0 0 0 0 0 0
b-D-Glc-1R;0 1 0 0 0 0 0 0 0 0 0 0
a-D-Glc-2R;0 0 1 0 0 0 0 0 0 0 0 0
b-D-Glc-2R;0 0 0 1 0 0 0 0 0 0 0 0
a-D-Glc-3R;0 0 0 0 1 0 0 0 0 0 0 0
b-D-Glc-3R;0 0 0 0 0 1 0 0 0 0 0 0
a-D-Glc-4R;0 0 0 0 0 0 1 0 0 0 0 0
b-D-Glc-4R;0 0 0 0 0 0 0 1 0 0 0 0
...

```

Figure 55: A sample cutout of the *output.csv* file

Line 1 contains the number of output neurons the PFG subroutine should write into the pattern file. The following lines contain the compound name followed by its (binary) output pattern. Compound name and output pattern are separated by a standard CSV delimiter (e.g. semicolon). If desired, the digits of the output can be optionally separated by a space to improve the readability of the file.

- All described values and settings can be saved in an individual configuration file. These settings can be reloaded later to reproduce exactly the same pattern as for the first time.

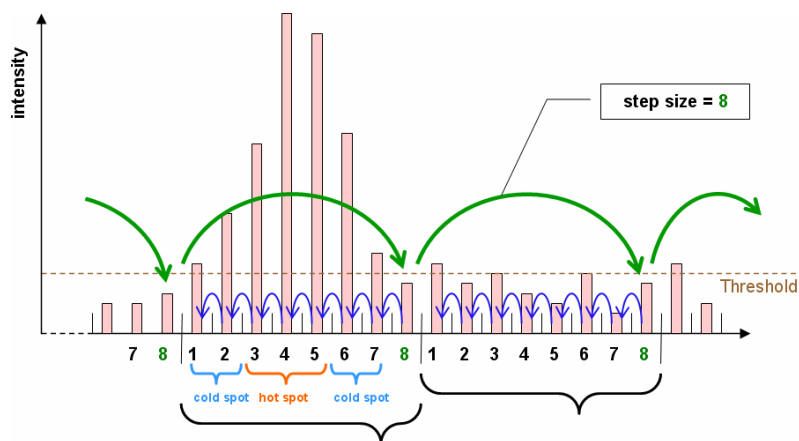
4.9.5.3. Data compression and block-pattern

The block-pattern approach is an effective solution to overcome the problem of peak loss if the reading step size entered by the user is too big. A detailed overview is given in Figure 56. The block-pattern algorithm can only be used for even step sizes.

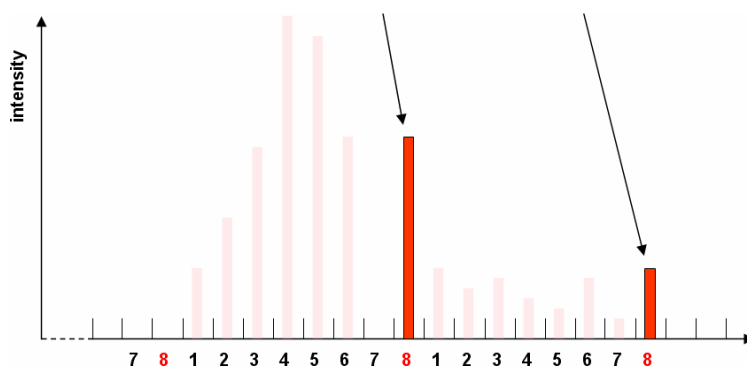
If the block-pattern option is chosen, the algorithm checks (blue arrows in step ①) after each reading step (green arrows in step ①), if an intensity of a data point exceeding the threshold level was skipped. If there is a value exceeding this threshold, the exact location within the last step is determined. The area of the skipped data points is divided into three similar regions. The first and the last third of the region are called the *cold spots*. The central region is named *hot spot* accordingly. If the exceeding data point of interest is located in the hot spot area, then all intensity values exceeding the threshold in this area (step size) will be averaged (step ②) and the average value is taken over into the final pattern file. If the data point of interest is located in one of the cold spot areas the exact intensity value is directly taken over into the pattern file.

Step ③ shows the final generated and compressed pattern file without the skipped data points with a step size eight.

Step ①



Step ②



Step ③

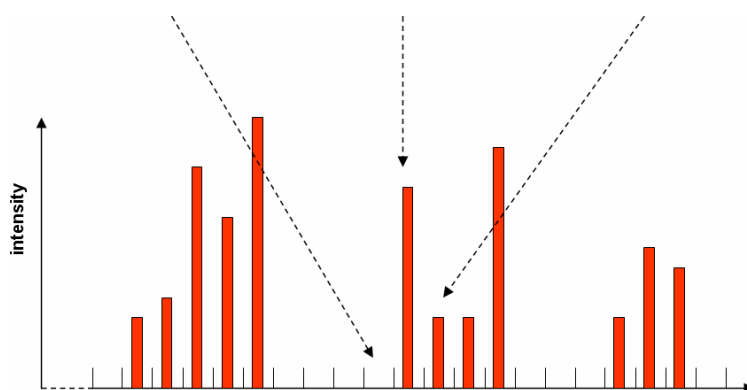


Figure 56: Formation path of a sample block-pattern with step size = 8

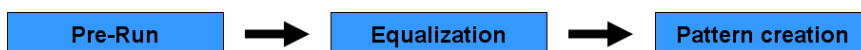
4.9.6. ANN PFG V.0.9

The UML sequence diagram is shown in Figure 69.

The final release of the ANN PFG is actually a summary of all important and well-proved functions of the predecessor versions. It accomplishes all requirements of the experiments at the state of the thesis. Because of still bad generalization abilities and too big neural networks, the data compression approach was radically redesigned and the resulting compression algorithm is called peak mask. This approach will be explained in the next chapter 4.9.6.1.

The support of JCAMP-DX files was discontinued because ^{13}C / ^1H -NMR data is now stored in the FileMaker ^{13}C -NMR database and exported as peak lists or directly taken over via ODBC from any database application. Non-peak data (e.g. noise) is unimportant for the generalization abilities of the tested neural networks and all experiments are based only upon literature peak lists without non-peak data.

Version 0.9 was equipped with a detailed preview window that displays the imported NMR peak lists and the generated peak mask (Figure 61). Located on the right side of the main GUI the user will find detailed statistics about the generated peak mask (Figure 59), the read in input peak list file and the estimated size of the generated pattern file. The workflow of the PFG is divided into three different work steps:



In the *Pre-Run* phase, the program generates the peak mask and collects all variables out of the input file necessary to proceed to the next step. Determined variables are ppm-max, ppm-min, max-intensity and min-intensity (Figure 57). The calculated peak mask and all NMR peaks of the input file will be displayed in the NMR preview windows (Figure 61).

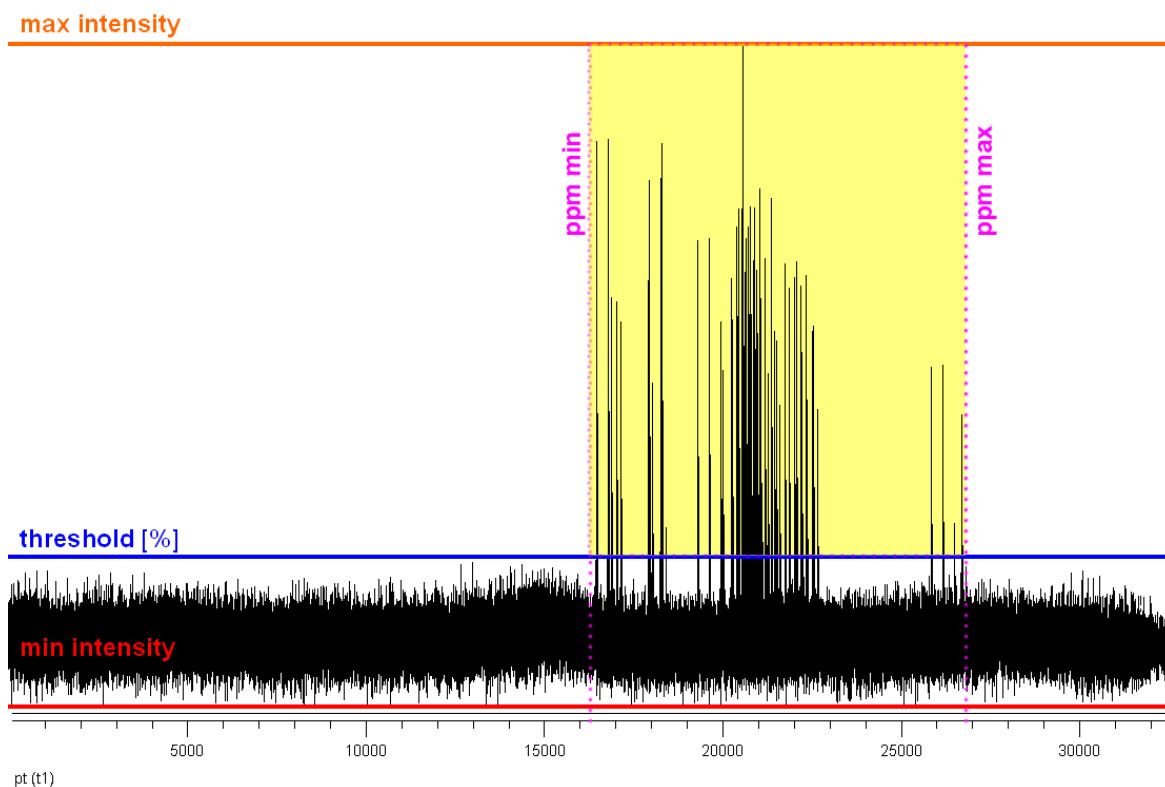


Figure 57: ANN PFG V 0.9.40 variables explanation

In the Equalization step, the user can fine-tune the parameters of the computer calculated peak mask and enter desired parameters needed to create the final pattern file. The last step consists of the pattern creation itself.

The input CSV file format the new PFG version can process is strictly specified and has to follow the following rules:

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
Data origin (NMR, FileMaker, Literature ...)	Compound name	Subset (Train, Test or Selection)	Peak 1 [ppm]	Peak 2 [ppm]	Peak 3 [ppm]	Peak x [ppm]

Each compound starts on a new line. The peaks starting at column 4 do not have to be sorted. The ANN PFG will sort them internally with the help of a bubble-sort algorithm. The peak list must be filled up with 0-peaks – as many 0-peaks as there are peak values in the longest record in the whole table.

```
average;a-D-Manp-1R;selection;101.06;73.42;71.42;71.22;67.26;61.50;0.00
average;a-D-Manp-OH;selection;94.85;73.15;71.49;71.08;67.86;61.75;0.00
average;a-D-Manp-OH-2R;selection;93.14;79.56;73.26;70.34;67.23;61.48;0.00
average;a-D-Manp-OH-4R;selection;94.91;77.74;72.24;71.04;69.88;61.37;0.00
average;a-D-Manp-OH-6R;selection;95.18;73.21;71.59;71.62;70.13;67.89;0.00
average;a-D-Manp-OMe;selection;101.70;73.28;71.36;70.73;67.78;61.68;55.71
average;a-D-Manp-OMe-2R;selection;99.98;79.19;73.40;70.84;67.76;61.58;55.50
average;a-D-Manp-OMe-3R;selection;101.39;79.31;7.40;68.68;66.51;61.42;55.50
....
```

Figure 58: Sample input CSV file for ANN PFG V.0.9

4.9.6.1. Pre-Run phase

After the selection of an input CSV file and choosing the desired NMR type the user clicks on the Pre-Run button and the following steps will be executed:

- The input CSV file is imported line by line into an array (the input CSV array). Each line is then separated into its columns. The whole input CSV file is now stored in a two-dimensional x/y array in the computers memory.
- Since there is the possibility, that the individual peaks are not arranged according to size, the peak list of each line will be sorted in descending order by a bubble-sort algorithm.
- In a next step, the whole input CSV array is transformed into the CSV point array: Each peak value (in ppm) is converted into its corresponding JCAMP-DX data point (^1H or ^{13}C -NMR with 8k, 16k and 32k resolution as defined by the user in the input options GUI). In the end, there is a second array (CSV point array) containing the same information but with the exception that the peaks are no longer stored in ppm-values but in JCAMP-DX data points.
- The values for $\text{ppm-max}_{\text{global}}$ and $\text{ppm-min}_{\text{global}}$ are determined (peak intensities are not available in literature peak data). Each peak is assigned the arbitrary intensity of 1'000'000.
- The peak mask is created with all data and values carried together so far. The procedure is explained in the next chapter.

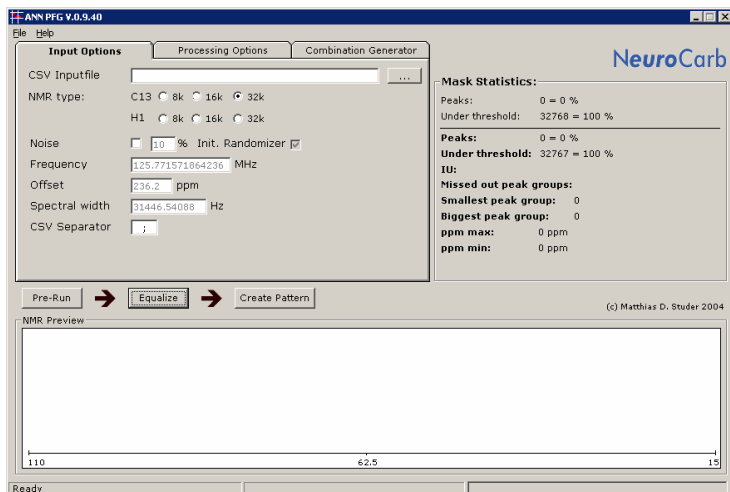


Figure 59: ANN PFG GUI - input and Pre-Run options tab

4.9.6.2. Peak mask

The peak mask is actually a simple bit mask (or a one-dimensional binary array). It has always the dimension of the NMR resolution the user requests (8k, 16k or 32k). The fundamental idea of the peak mask is to create a one-dimensional map or copy of **all** peaks contained in the compounds in the input CSV file. And this map serves as a template or filter for the final output pattern file.

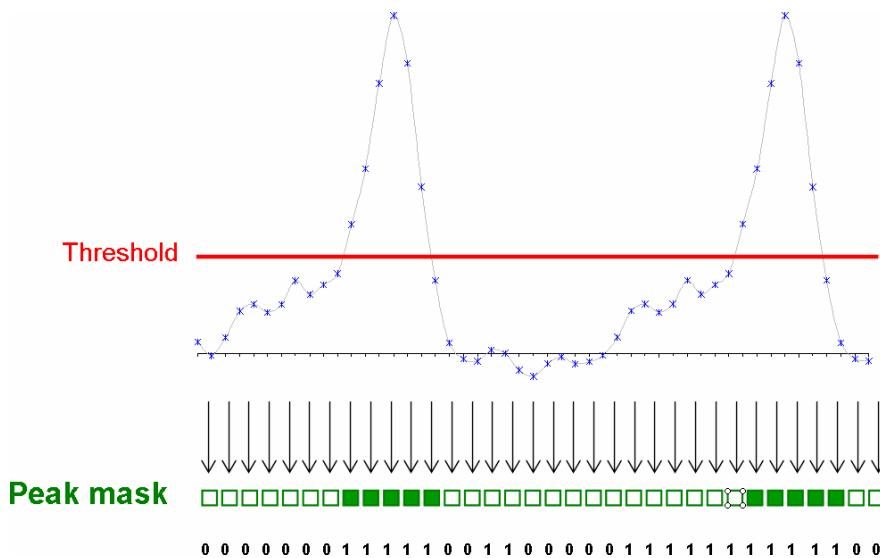


Figure 60: Formation of the binary peak mask

For these purposes, only peaks exceeding the threshold level (adjusted by the user) will be mapped on the peak mask. Peaks exceeding this level will be mapped in binary form; above threshold = 1 and below threshold = 0 (Figure 60). Thus, peaks are not mapped by their intensity but only by their dispersion (only their "shadow" by perpendicular "illumination"). All other information is lost – and not necessary for the application of the peak mask. The mask can also be regarded as a kind of superimposed dot mask filter containing all the peaks available (as holes) in the input file (Figure 61).

During every reading cycle in the CSV point array, the algorithm checks up in the peak mask if the corresponding bit is set to 1 or 0. If the bit of the processed data point is set to 1, the peak information will be written into the final pattern file. Otherwise, the information of the CSV point array at the corresponding data point will not be processed and written in the pattern file. This procedure shows very well, how the mask can be used for input data compression. Regions with no peaks (regions with bits set to 0 in the peak mask) won't appear in the final pattern file. And on the other hand the pattern file only contains dense peak information used as input for the neural network.

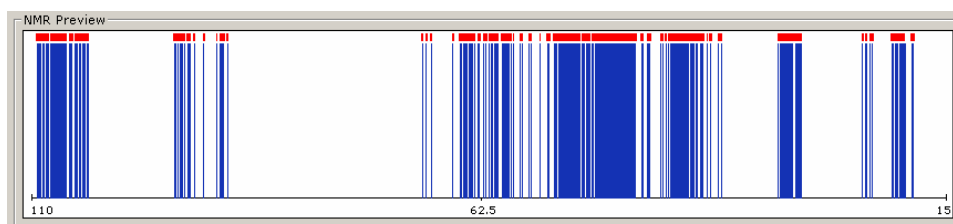


Figure 61: Preview of overlaid NMR spectra (blue) and calculated peak mask (red)

Consider the following example to show how the peak mask also acts as an input filter: The initially "closed" peak mask (all bits set to 0) will be opened or "perforated" (bits set to 1) only just at the regions where the compounds in the input CSV file contains peak information. The peak mask created during the Pre-Run can certainly be saved to a configuration-file. In later applications the same saved peak mask can be reloaded (e.g. to create an identical pattern to test a specific neural network) and will filter the new test compounds insofar as only "known" compounds will pass the peak mask filter. Disturbing impurities or other foreign substances will be ignored.

A peak mask built from a certain carbohydrate (e.g. glucose) will act as a filter for other compounds or disturbing impurities. Peaks in regions, where the peak mask built from e.g. glucose is set to 0, will not appear in the final pattern file for the neural network to train or test. Therefore, a peak mask created with glucose will not allow peaks from e.g. rhamnose to enter the final pattern file. But rhamnose peaks in regions overlapping with glucose won't be filtered. This cannot be prevented.

To equalize and refine the peak mask, two different approaches are also implemented into the PFG V.0.9:

1. With values entered in the Mask Equalization field in the GUI, it is possible to delete small gaps between adjacent peaks in the peak mask. The bits of gaps smaller than the specified value will be set to 1. This leads to a smoother mask and "opens" the mask for peak information not already known at the time of the peak mask creation. E.g. poorly shifted NMR spectra.



Figure 62: Peak mask equalizing (4 points)

2. Another approach is the introduction of a tolerance factor (the input field is also located on the GUI). This subroutine enlarges the peak mask regions with bits set to one (the holes) on every side.



Figure 63: Peak mask with tolerance = ± 2

4.9.6.3. Pattern creation

The final pattern creation step consists of filtering the input CSV file through the peak-mask created in the Pre-Run step or loaded from a file. If desired, the default values for the ppm-range, the zero threshold level or the input unit activation values, can be adjusted by the user. If nothing different is selected, binary pattern files (data point not exceeding zero threshold level = 0, data point exceeding zero threshold level = 1) are created. The ppm-range values ppm-max and ppm-min are automatically determined out of the input CSV file.

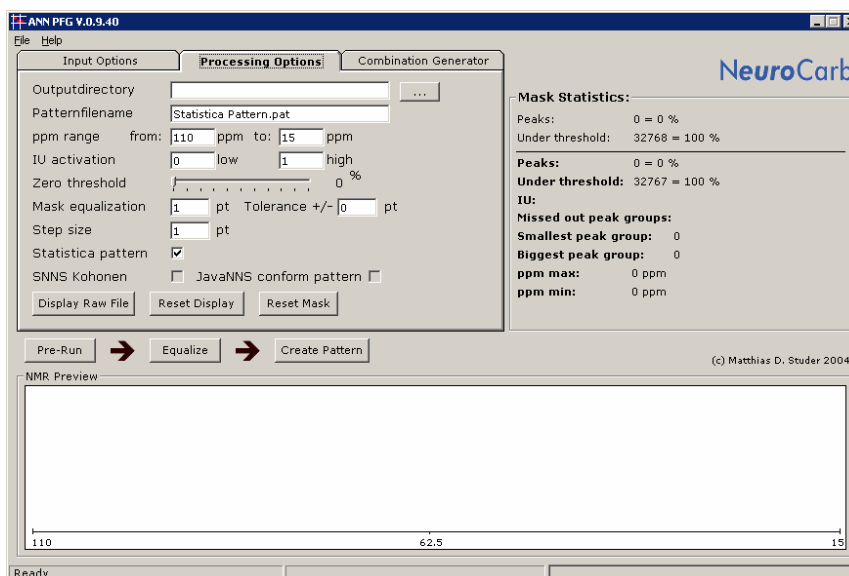


Figure 64: ANN PFG GUI - processing options tab

The algorithm also checks if the step size is larger than the biggest gap in the peaks mask. If this is the case the according peak would be wrapped. The block-pattern approach was abandoned. If desired the subprogram can create output files in the SNNS (Backprop or Kohonen) or Statistica format. As SNNS can't work with clear text output values the target output values for SNNS pattern files have to be encoded in a binary format. This task is fully automated and integrated in PFG V.0.9. Each different compound found in the input CSV file is assigned a consecutive binary number. This binary number has as many digits as there are different compounds in the input CSV file. Leading digits not used will be filled up by zero (example in Figure 63)

1:	1	0	0	0	0	0	0	0	0	0	0	0	0
2:	0	1	0	0	0	0	0	0	0	0	0	0	0
3:	0	0	1	0	0	0	0	0	0	0	0	0	0
4:	0	0	0	1	0	0	0	0	0	0	0	0	0
5:	0	0	0	0	1	0	0	0	0	0	0	0	0
6:	0	0	0	0	0	1	0	0	0	0	0	0	0
7:	0	0	0	0	0	0	1	0	0	0	0	0	0
8:	0	0	0	0	0	0	0	1	0	0	0	0	0
9:	0	0	0	0	0	0	0	0	1	0	0	0	0
10:	0	0	0	0	0	0	0	0	0	1	0	0	0
11:	0	0	0	0	0	0	0	0	0	0	1	0	0
12:	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 65: A sample binary output-coding matrix for 12 different compounds for SNNS pattern files

The final output files look like the examples described in chapter 4.9.3.1 and 4.9.3.2.

4.9.6.4. Combination generator

The combination generator is a subprogram added in a later stage to PFG V.0.9. It was originally designed to create pattern files of single compounds just by entering the peak list in the designated fields. And not to enter the single compound via an extra prepared CSV input file. However, the problems discussed in chapter 5.8.8 lead to the invention and introduction of the combination generator.

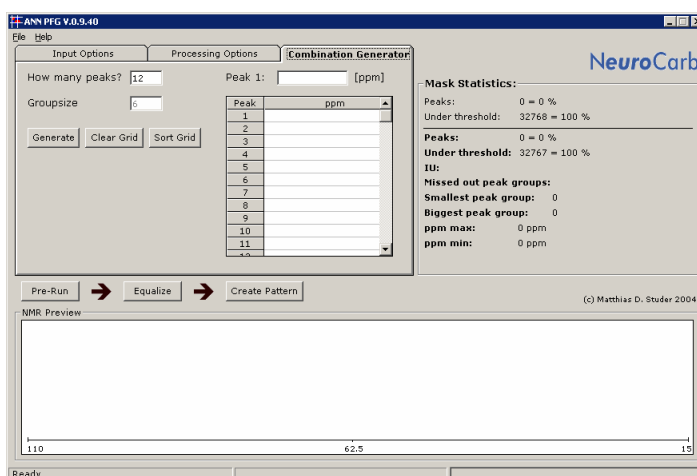
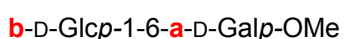
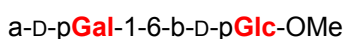


Figure 66: ANN PFG GUI - peak combination generator tab

When feeding e.g. twelve peaks (a disaccharide) into a neural network, it is sometimes impossible to assign all peaks to the corresponding monosaccharide unit. Test compounds (e.g. **a-D-Glcp-1-6-b-D-Galp-OMe**) are often understandably recognized as follows:



α, β interchanged



Monosaccharide moieties interchanged



This can be explained thereby that a neural network has no knowledge about the ring membership (spin system) of a single peak and the false positive results cannot be regarded as real faults. The assignment of the ring membership could also be a problem for a human NMR interpreter.

To overcome this problem the combination generator subprogram generates an array of all possible combinations of how to group n peaks into groups of g peaks. For a disaccharide with e.g. 12 peaks, there are only **two** reasonable combinations out of 924. The remaining 922 combinations are senseless.

Peaks:	1 2 3 4 5 6	7 8 9 10 11 12	assumption: peak 6 = α and 12 = β
924 possible combinations of 6 peaks:			
	1 2 3 4 5 6		✓
	1 2 3 4 5 7		✗
	1 2 3 4 5 8		✗
	1 2 3 4 5 9		✗
	1 2 3 4 5 12		→ false positive combination (Peak 6 and 7 interchanged)
		
	3 4 7 10 11 12		✗
		
	6 7 8 9 10 11		→ second false positive compound
	6 7 9 10 11 12		✗
	6 8 9 10 11 12		✗
	7 8 9 10 11 12		✓

Figure 67: Combinations example with 12 peaks / 6 peaks per group and α/β confusion (anomeric peaks 6 and 12)

The array with all possible peak combinations is computed with the following formula:

$$c = \frac{n!}{g \times (n-g)!} \quad \text{Equation 17}$$

c = number of possible combinations
n = number of peaks in NMR spectrum
g = group size (normally g=6 for a monosaccharide)

This internal array with all combinations is then processed exactly like a normal CSV input file and finally converted into a neural network test pattern file. If this pattern file is tested with different pre-trained neural networks, most networks will statistically recognize the correct two combinations. The minority will also recognize the false positive or some wrong combinations.

In the following example, 40 neural networks were tested with 924 peak combinations of a single disaccharide peak list containing 12 peaks.

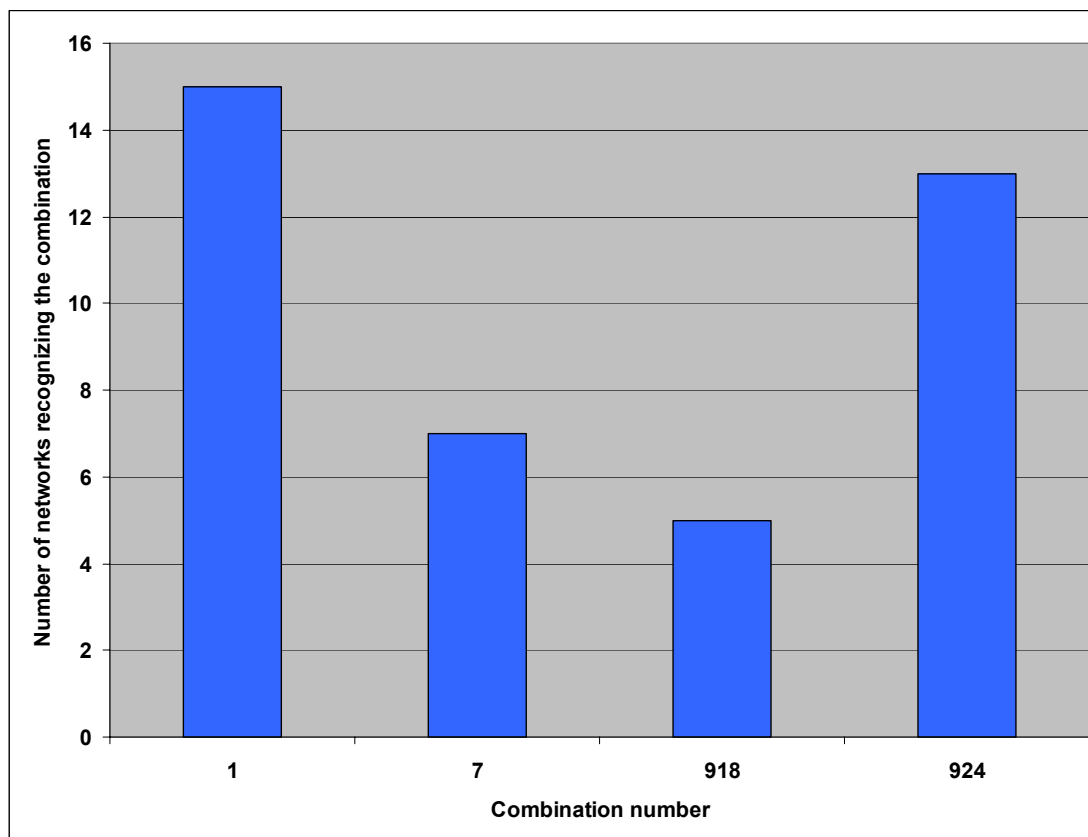


Figure 68: Test results overview of 40 neural networks tested with 924 peak combinations

The results show, that the two right combinations are recognized by the majority of the 40 neural networks tested with all of the 924 combinations. The false positive combinations are recognized as well but with lower incidence.

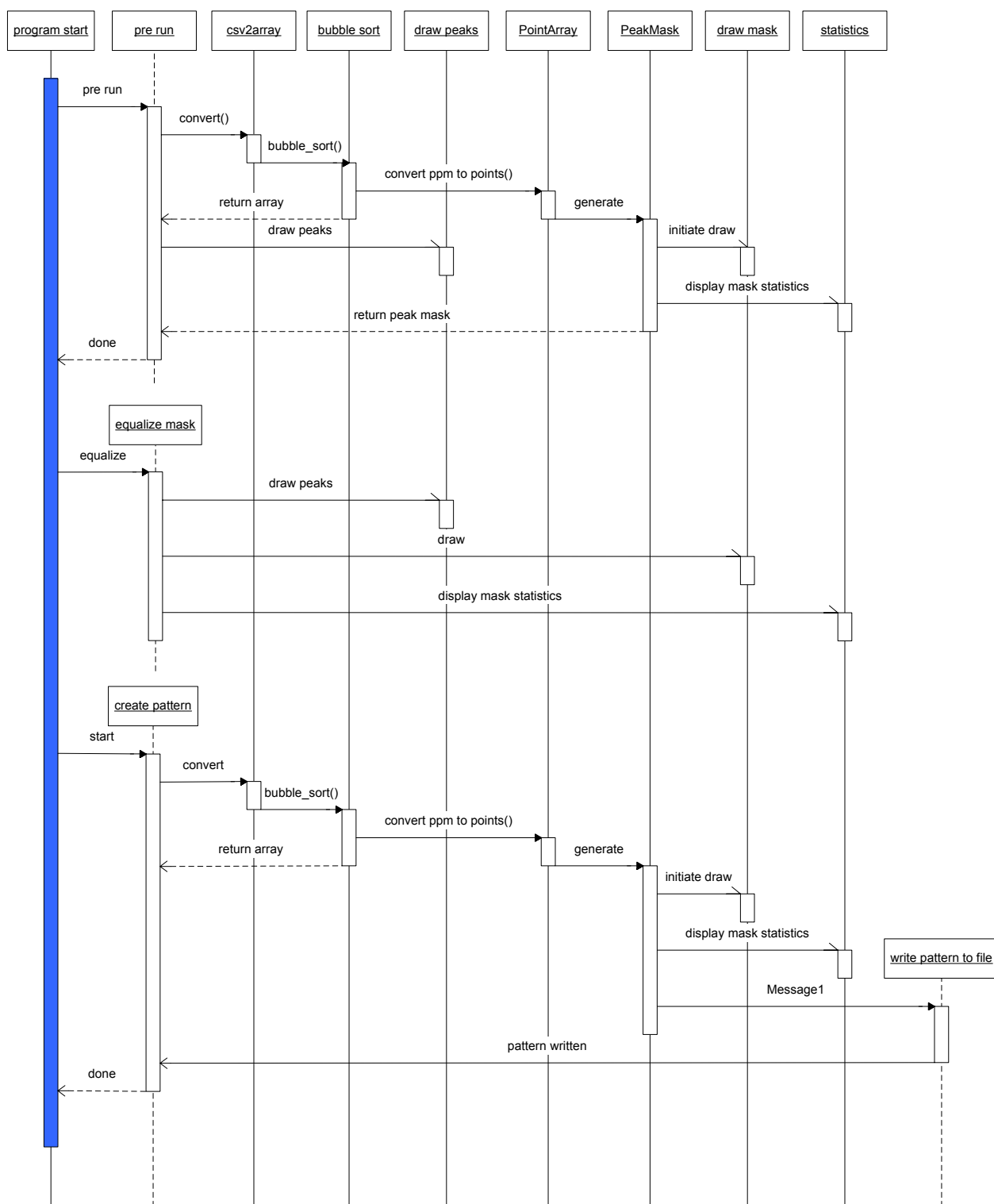


Figure 69: ANN PFG V.0.9 UML sequence diagram

5. Experiments

5.1. Glycosylation shifts

The following graphs illustrate the glycosylation shifts of substituents R at different ring positions. The analysis was done for glucose, galactose and mannose. The used ppm-values are average shift values of all corresponding compounds in the FileMaker ^{13}C -NMR database (chapter 4.1.5).

5.1.1. α -D-Glcp-OMe-xR

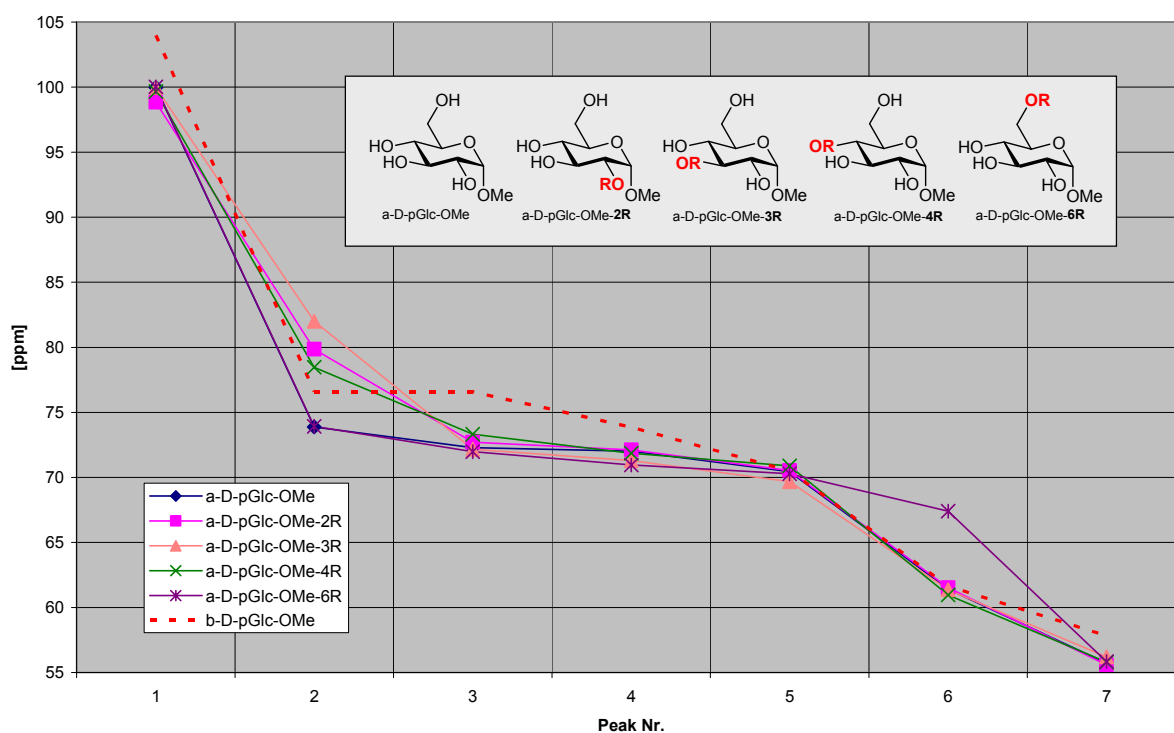


Figure 70: Influence of substituents at different ring positions for α -D-Glcp-OMe

5.1.2. β -D-Glcp-OMe-xR

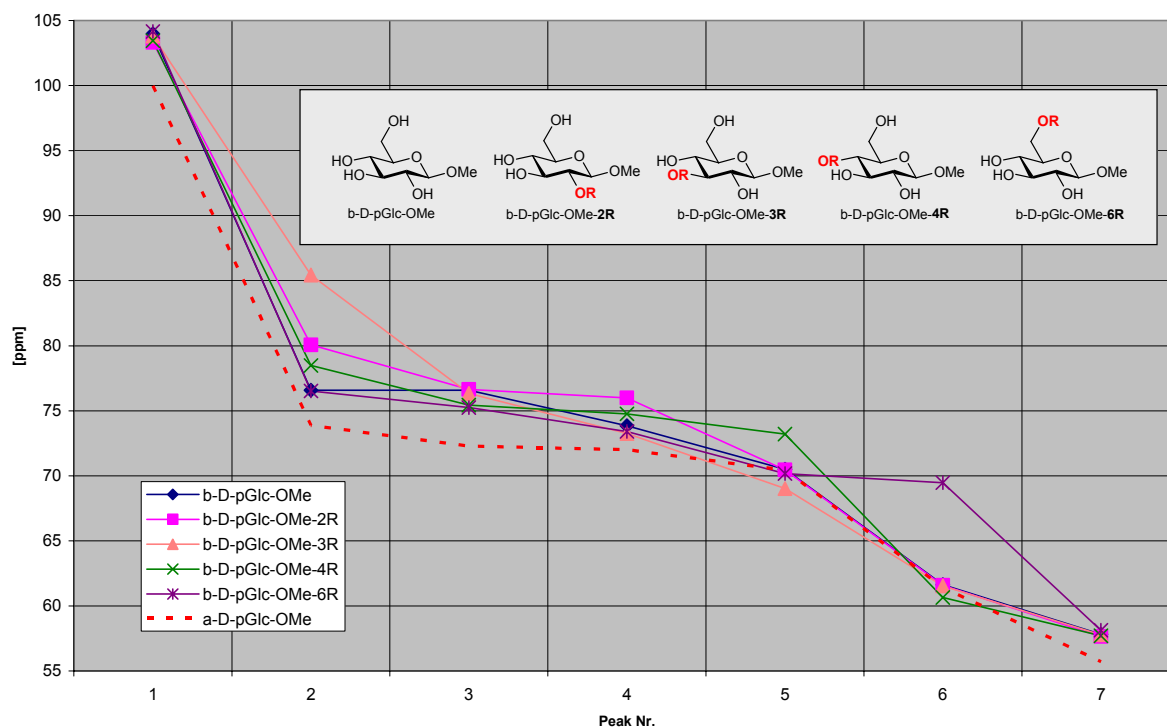


Figure 71: Influence of substituents at different ring positions for β -D-Glcp-OMe

5.1.3. α -D-Glcp-xR

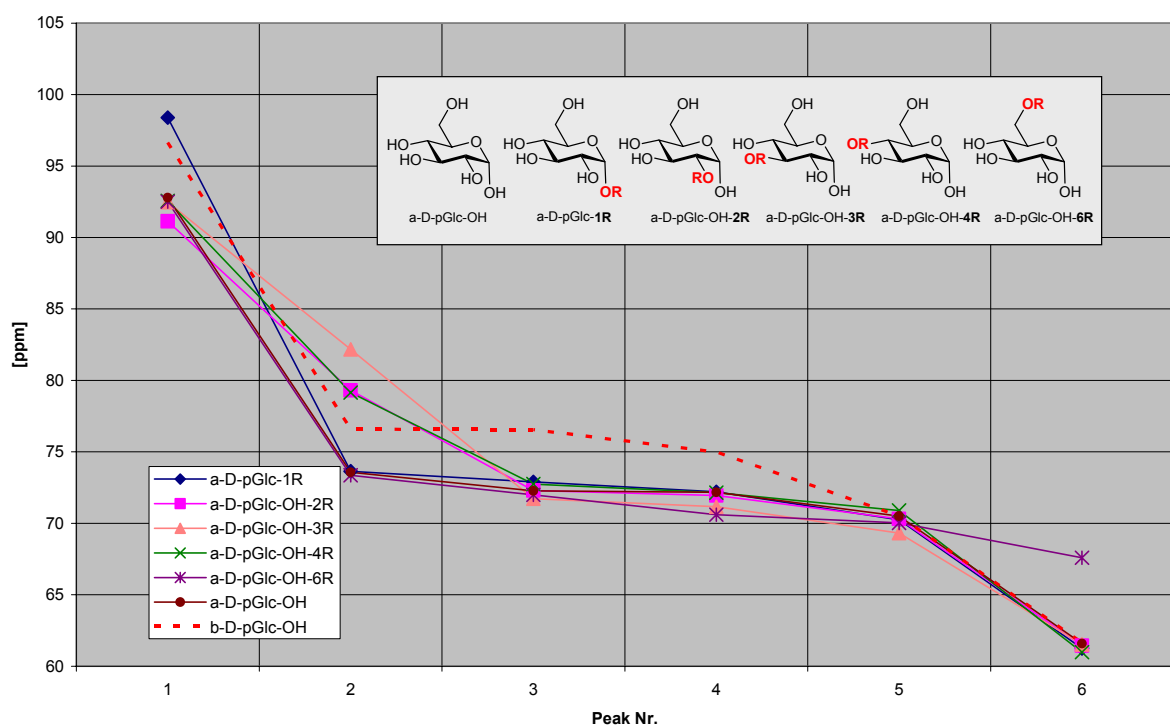


Figure 72: Influence of substituents at different ring positions for α -D-Glcp

5.1.4. β -D-Glcp-xR

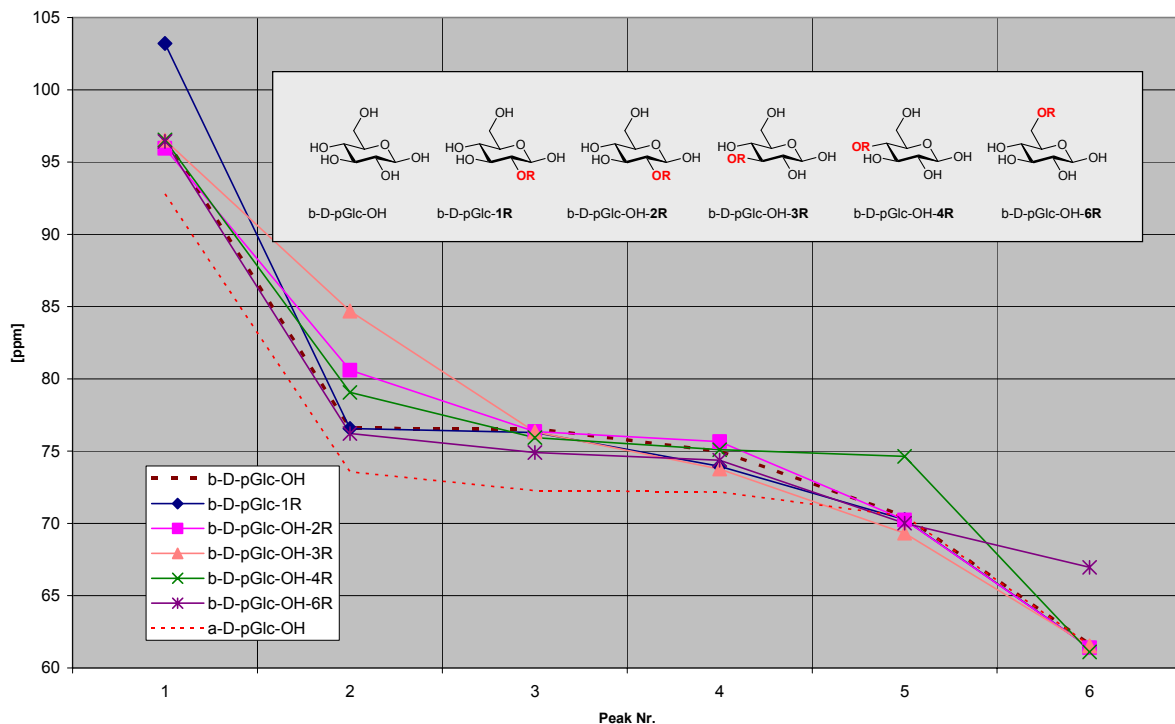


Figure 73: Influence of substituents at different ring positions for β -D-Glcp

5.1.5. α -D-Manp-xR

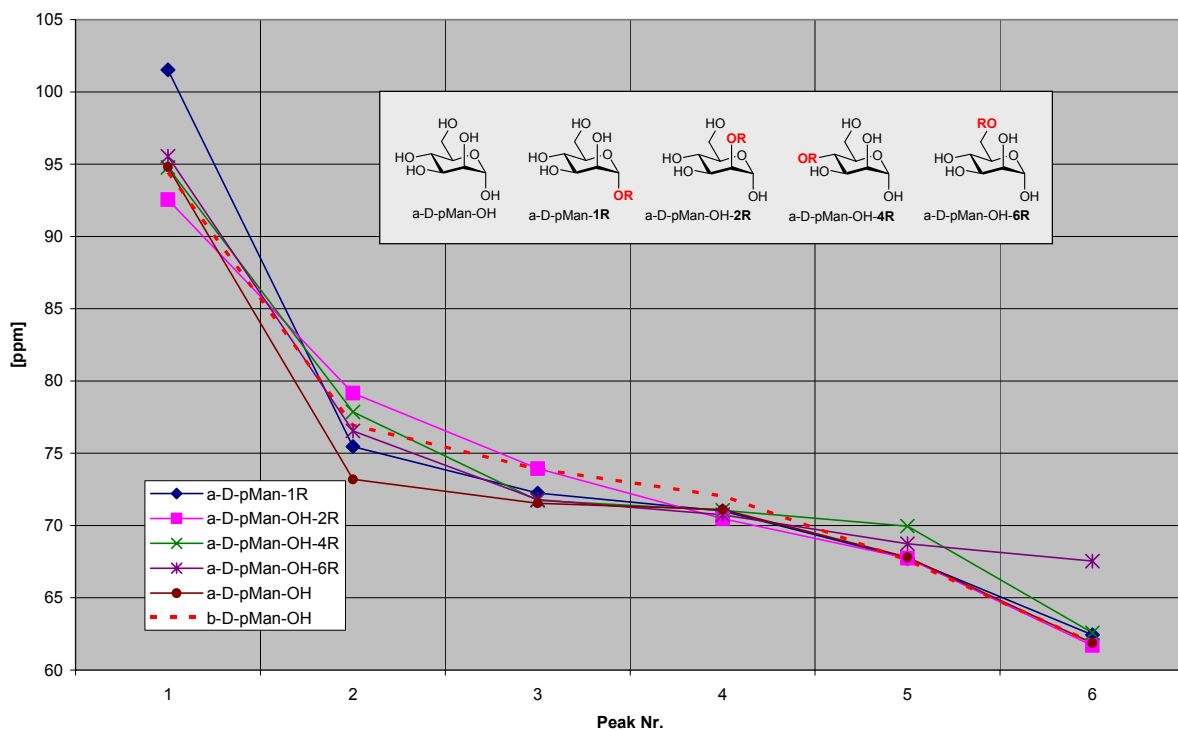
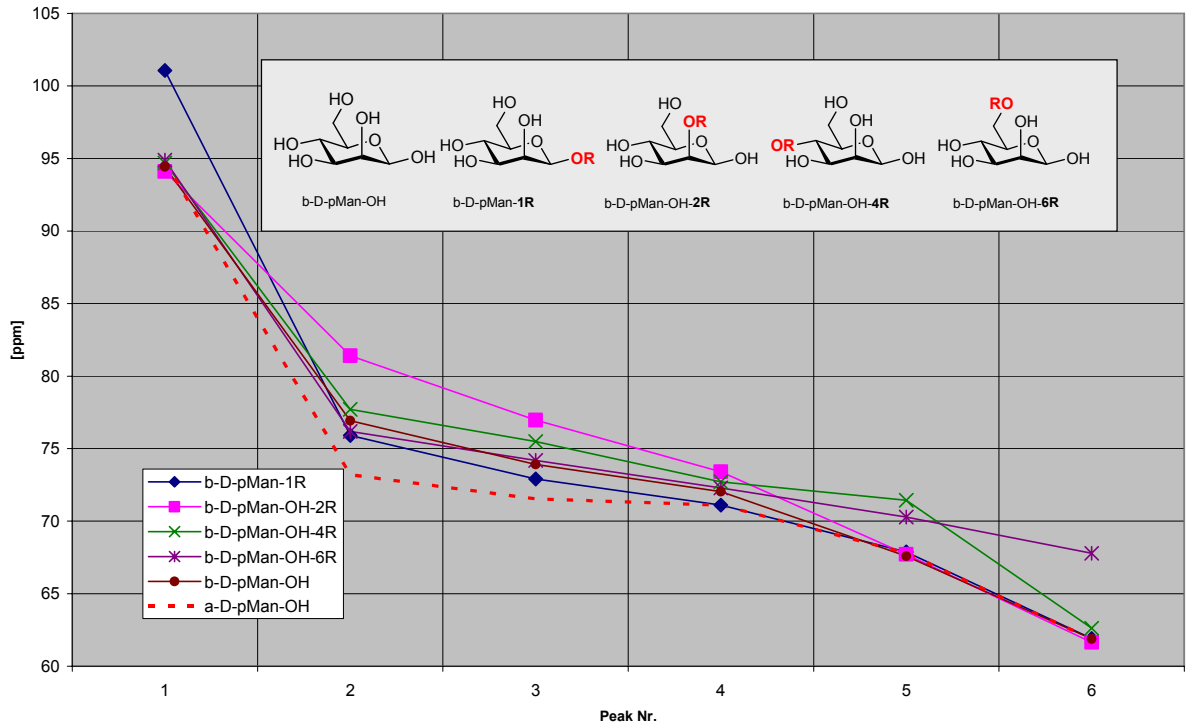
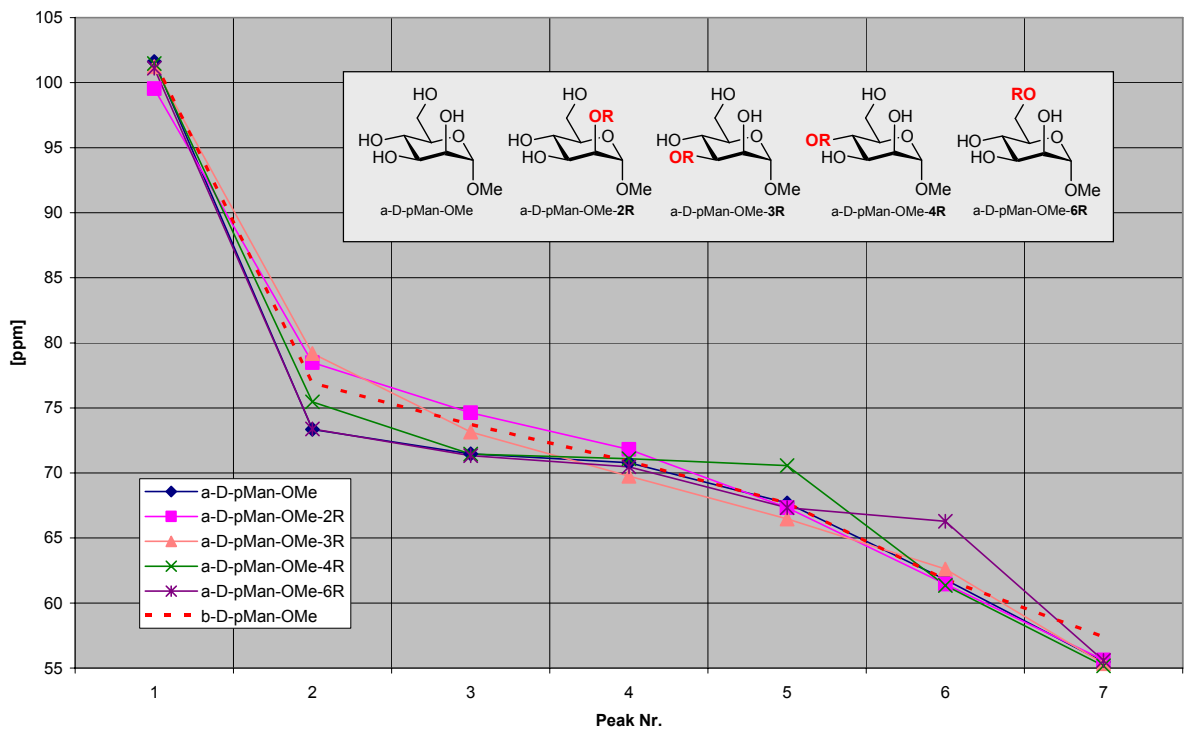


Figure 74: Influence of substituents at different ring positions for α -D-Manp

5.1.6. β -D-Manp-xRFigure 75: Influence of substituents at different ring positions for β -D-Manp5.1.7. α -D-Manp-OMe-xRFigure 76: Influence of substituents at different ring positions for α -D-Manp-OMe

5.1.8. β -D-Manp-OMe-xR

There is no data available in the FileMaker ^{13}C -NMR database.

5.1.9. α -D-Galp-xR

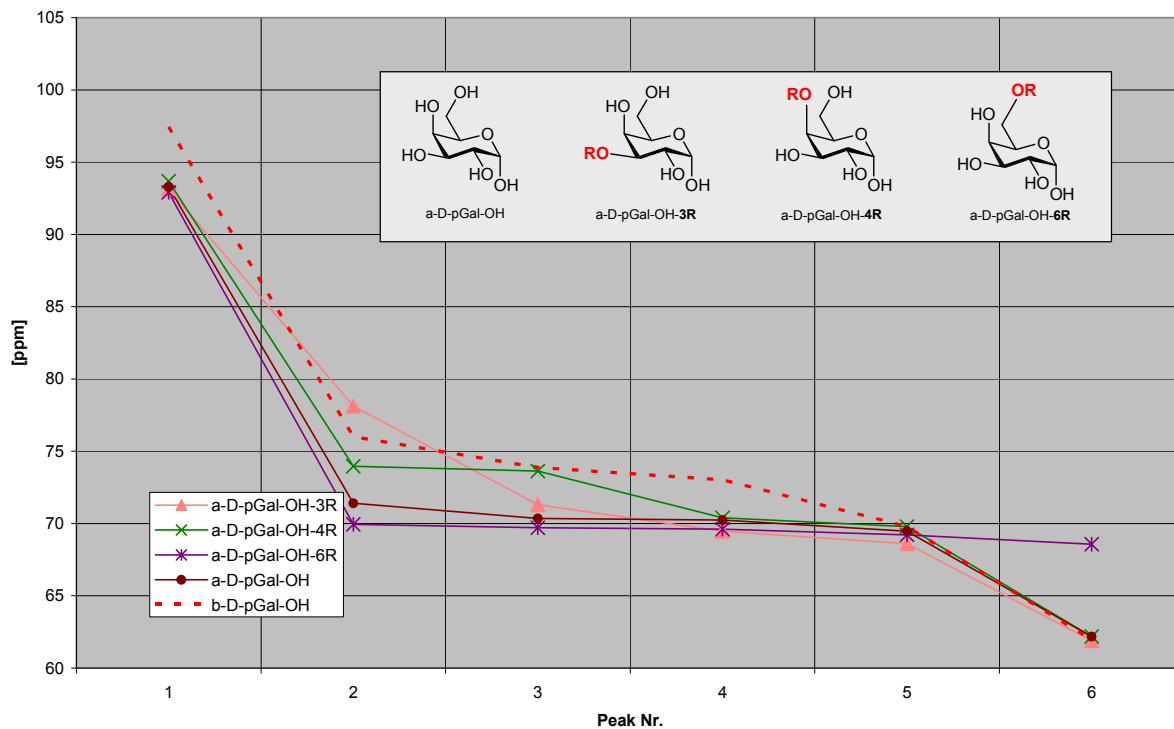
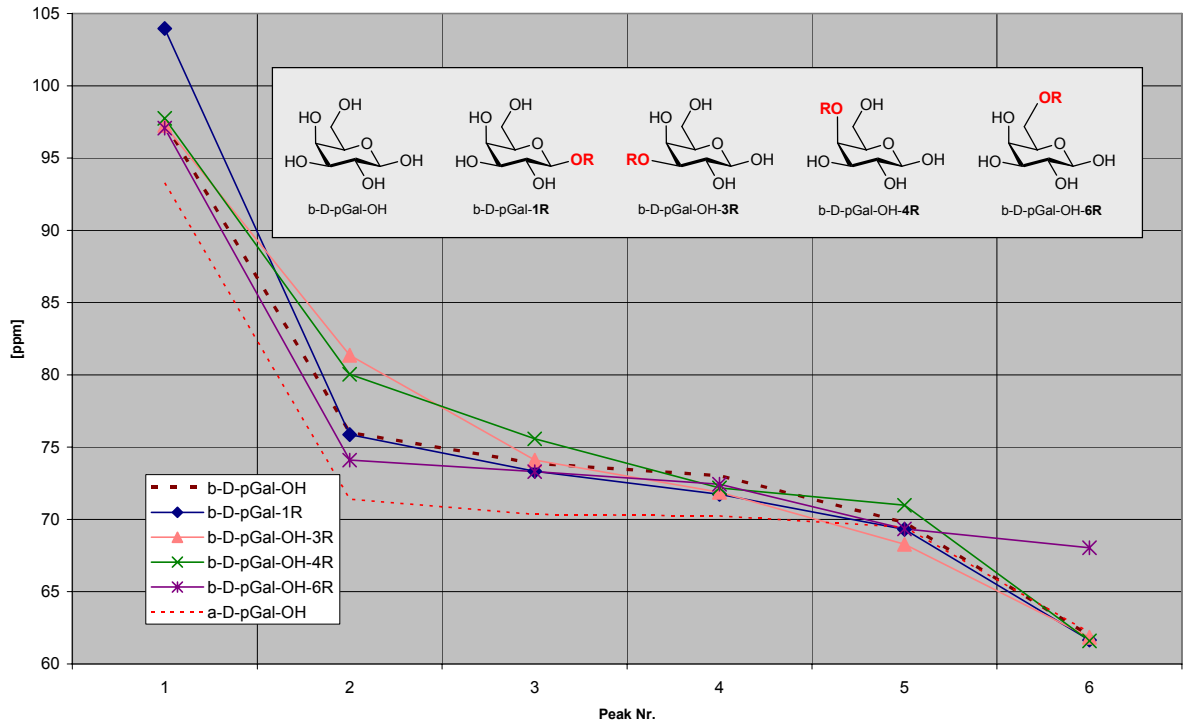
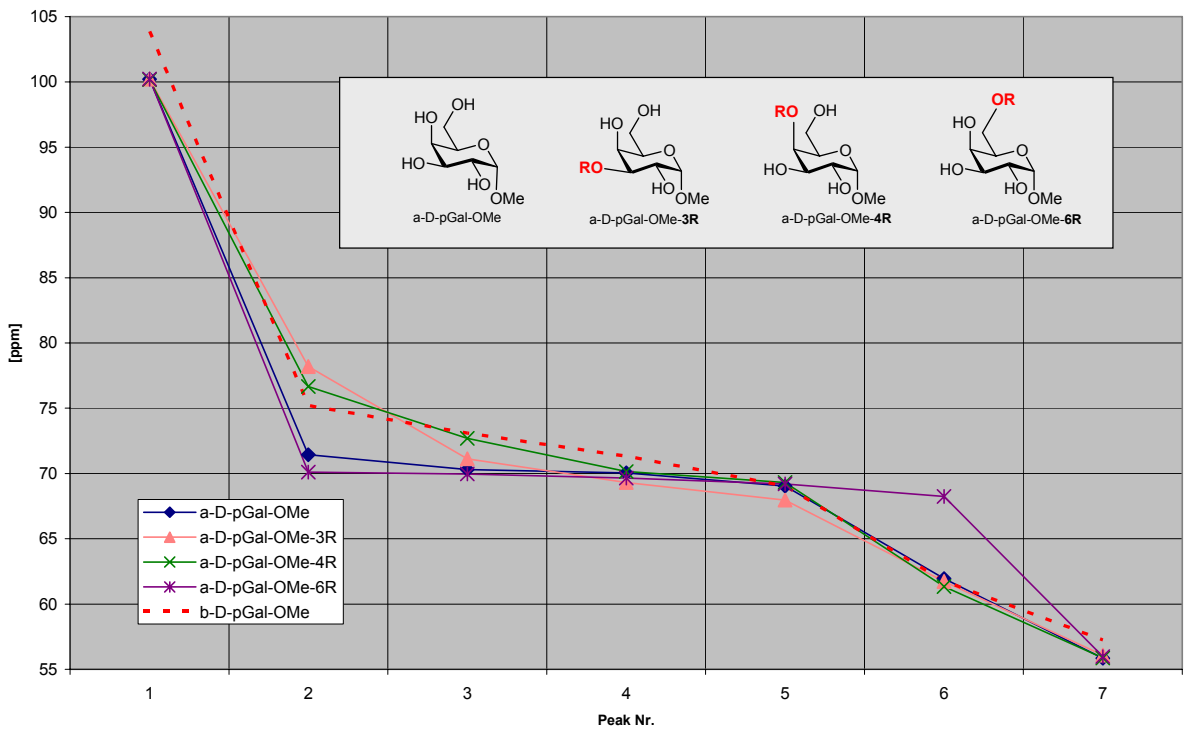
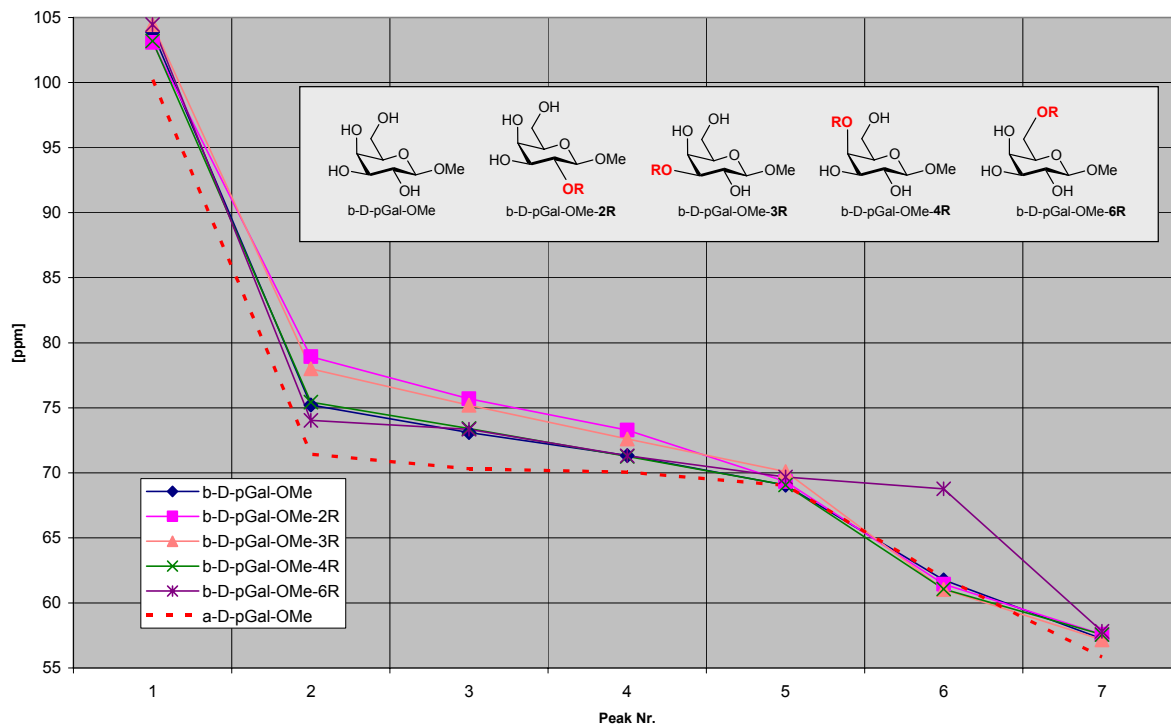


Figure 77: Influence of substituents at different ring positions for α -D-Galp

5.1.10. β -D-Galp-xRFigure 78: Influence of substituents at different ring positions for β -D-Galp5.1.11. α -D-Galp-OMe-xRFigure 79: Influence of substituents at different ring positions for α -D-Galp-OMe

5.1.12. β -D-Galp-OMe-xRFigure 80: Influence of substituents at different ring positions for β -D-Galp-OMe

5.2. General definitions

Term	Definition / Explanation
Generalization	A measure of how well a network can respond to new data on which it has not been trained but which are related in some way to the training patterns. An ability to generalize is crucial to the decision-making ability of the network.
Recall	A measure of how well a network can respond to the data it was trained with. If the recall rate reaches 100% network training can be stopped because no further improvement in generalization can be expected.
Valid set	Other expression for test data set to test the generalization ability of the trained neural network. This test can be done after the training or during the training process.
d_{max}	The maximum difference $d_j = t_j - i_j$ between a teaching value t_j and an output o_j of an output unit which is tolerated, i.e. which is propagated back as $d_j = 0$. ^[296]
Threshold	Activation level an output neuron has to exceed to be classified as a correct output. This is especially important when binary teaching values are used.

5.3. Methyl pyranosides approach

The idea of these experiments was to apply the approach of Meyer *et al.*^[80-82] to glycopyranoses, because of the fact that there were no sugar alditols available. The compounds used were five methyl pyranosides presented in chapter 4.1.1. The methyl pyranosides have the advantage that they are clearly defined at the anomeric carbon and the information of the difference between α and β configuration can be included into the training of the neural networks. For the identification of disaccharides, the anomeric configuration is indispensable.

5.3.1. ¹H-NMR data

All five compounds were measured under the conditions explained in chapter 4.2. The amount of sample is indicated under 4.1.1.

To artificially amplify the data set, each compound was measured four times and then twelve modifications (according to Table 12) were created with the ANN PFG V.0.1.

Train: 5 compounds x 4 NMR measurements x 12 modifications = 240 compounds

To gain the valid data set, the procedure was repeated with only six modifications per NMR experiment.

Valid: 5 compounds x 4 NMR measurements x 6 modifications = 120 compounds

Before creating the pattern file, the twenty acquired NMR spectra were superimposed. To include all peaks in the twenty samples, a peak range from 4.4 to 3.2 ppm had to be included into the final

pattern files. After processing the JCAMP DX files with the ANN PFG V.0.1 the resulting pattern files (training and test) contained 2620 input neurons.

Table 12: Modification codes and their explanation

Mod	code	explanation
1	a	no variation
2	b	half intensity
3	c	add 40db noise
4	d	right shift 1Hz
5	e	left shift 1Hz
6	bc	half intensity + Noise
7	bd	half intensity + right shift 1Hz
8	be	half intensity + left shift 1Hz
9	cd	add 40db noise + right shift 1Hz
10	ce	add 40db noise + left shift 1Hz
11	bcd	half intensity + add 40db noise + right shift 1Hz
12	bce	half intensity + add 40db noise + left shift 1Hz

5.3.1.1. Comparison of different learning rates

This experiment was the first test run made during this thesis. As a starting point, all parameters were chosen according to default values proposed in the SNNS manual ^[296] and the Neural Network book from J. Gasteiger ^[232]. The main aim of this experiment was to determine if it is possible to separate the five used methyl pyranosides and if it is possible to recognize unseen but similar ¹H-NMR spectra.

ANN PFG V.0.1 parameters	NMR type Shift Zero threshold Input range	¹ H – 32k data points ± 1 Hz 10 4.4 - 3.2 ppm
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	2620 input neurons 200 hidden neurons 5 output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (±0.5)	
Learning rate	Variable	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	1	
Pattern shuffling	Activated	

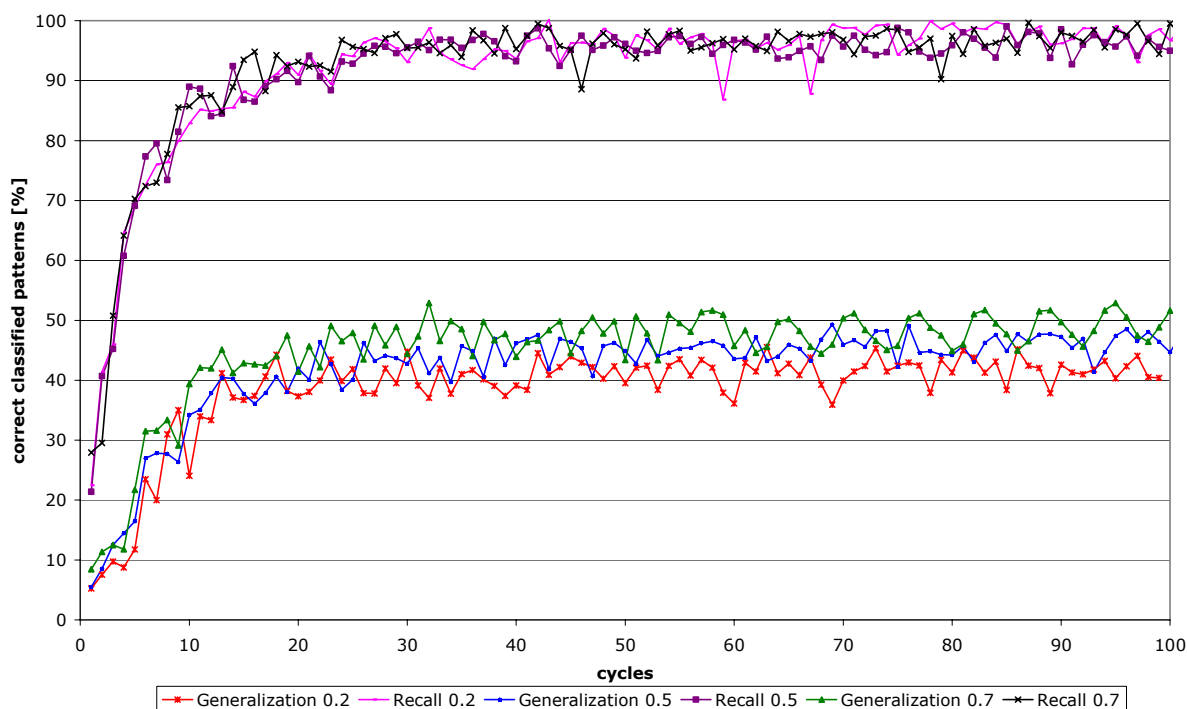


Figure 81: Learning rate comparison

Figure 81 clearly shows, that the generalization rate of the networks is independent from the chosen learning rate. A 100% recall rate is reached after ~30 cycles. An improvement of the generalization rate was not expected because the recall curve already reached its maximum and these two curves always climb in parallel.

5.3.1.2. Hidden layer size comparison

The next parameter expected to have a great influence, is the size of the hidden layer. A range from zero to 800 hidden units seems to be appropriate. Bigger networks would take too long to compute.

ANN PFG V.0.1 parameters	NMR type Shift Zero threshold Input range	¹ H – 32k data points ± 1 Hz 10 4.4 - 3.2 ppm
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	2620 input neurons variable hidden neurons 5 output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (±0.5)	
Learning rate	0.2	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

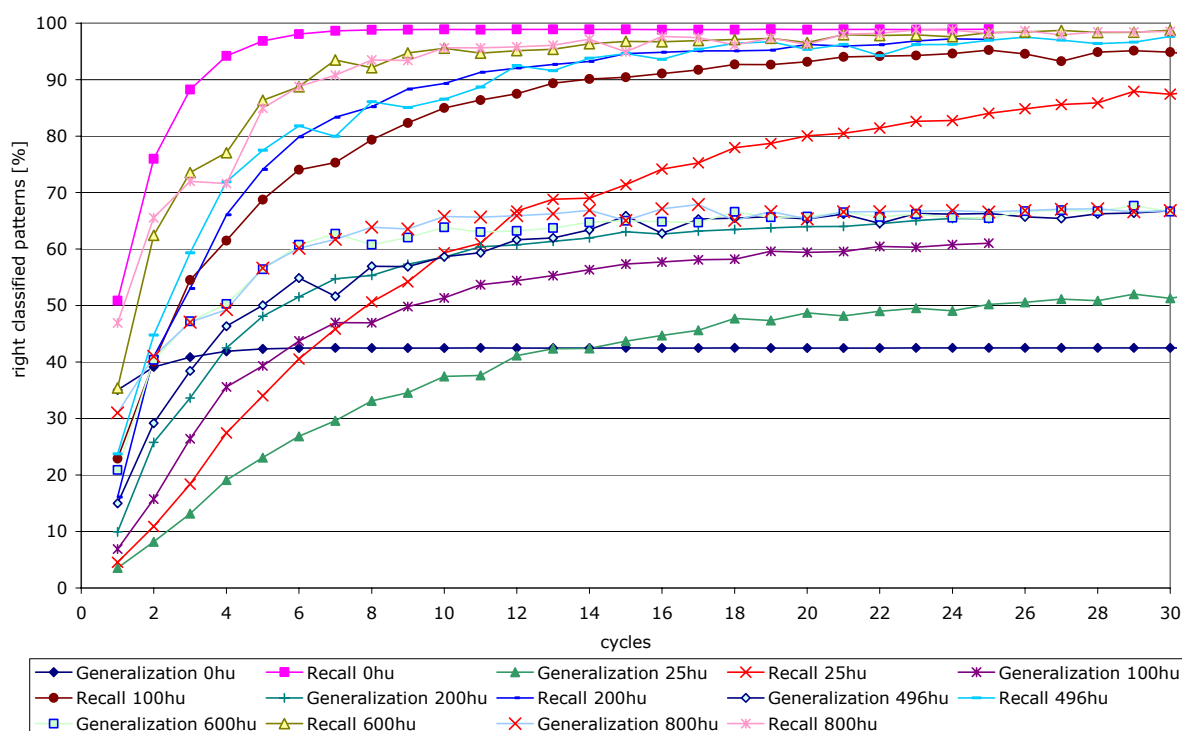


Figure 82: Hidden layer size comparison

- Figure 82 shows a saturation of the generalization rate starting at the size of about 200 hidden units. Networks with more hidden units cannot improve the generalization rate.
- Networks with more than 800 hidden units were not trained because of the immense amount of time needed to train these networks. A network with 2620 input units, 800 hidden units and 5 output units has a total of 2'100'000 weights ($2620 \cdot 800 + 800 \cdot 5 = 2'100'000$) and it took 288 hours to train this network up to 100 cycles.
- Linear networks do not seem to be suitable to predict unseen test data.
- The optimal point to stop the training is reached after about 10 training cycles.
- The results of this experiment highlight the urgent need of more compounds to train the network. The artificially generated modifications are not sufficient to give good generalization results. Literature says that 10 times as many training cases as input units are needed to get satisfying results. This may not be enough for the highly complex functions at hand. For classification problems, the number of cases in the smallest class should be at least several times the number of input units ^[302]. According to this rule of thumb, the networks used in this experiment should be trained with at least 26'200 training cases. This demand cannot be achieved with twelve simple modifications of five NMR compounds.

5.3.1.3. MSE and classification comparison with 246hu

In a next step, the output coding values for an activated output unit were changed from 1 to 0.75 and from 0 to 0.25 for a deactivated output unit (new values are shown in dotted red lines in Figure 83). This change is reasonable because the curve of the logistic activation function has its steepest part between 0.75 and 0.25. This means, that small changes of the net input result in stronger output changes. If the target output levels are set to 1 and 0 accordingly, it takes indefinitely many training cycles to reach the desired target output because the logistic function is not defined for 1 and 0. Therefore, the network can never reach the required target output values.

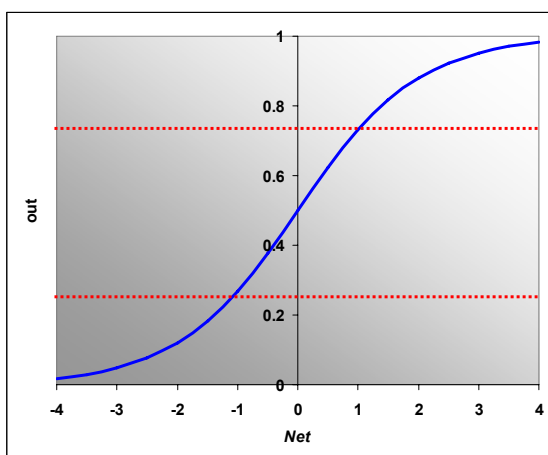


Figure 83: Act Logistic activation function

ANN PFG V.0.1 parameters	NMR type Shift Zero threshold Input range	¹ H – 32k data points ± 1 Hz 10 4.4 - 3.2 ppm
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	2620 input neurons 246 hidden neurons 5 output neurons	
Output coding	0.75 = activated / 0.25 = deactivated	
Training cycles	100	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (±0.5)	
Learning rate	0.1	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

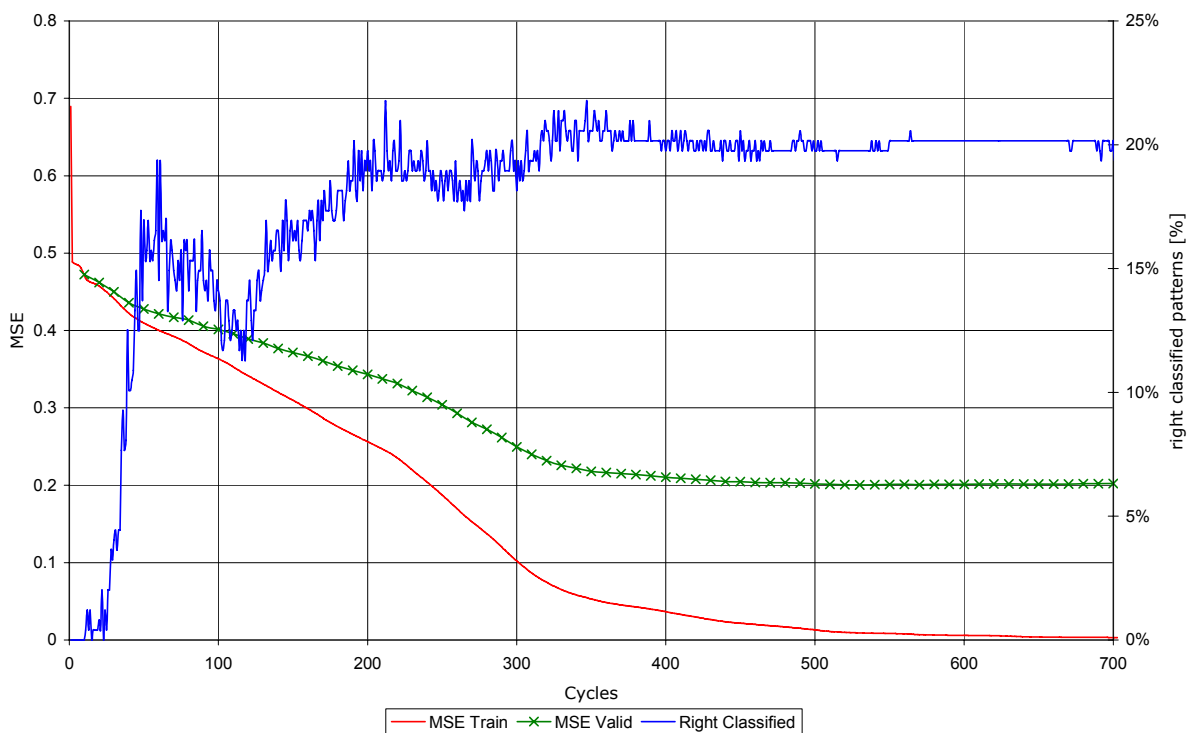


Figure 84: MSE and classification comparison for target output values 0.75 and 0.25

The blue colored classification curve depicted in Figure 84 flattens after about 400 training cycles. The best generalization rate of 67% of experiment 5.3.1.2 was far not reached. The red training MSE curve drops steadily to zero. Therefore, no further improvements in generalization can be expected. Why the generalization rate never exceeds 22%, cannot be explained.

5.3.1.4. MSE and classification comparison with 100hu

The following experiment was carried out with exactly the same setup as the previous experiment 5.3.1.3. The only difference is a smaller hidden layer size of 100 units.

ANN PFG V.0.1 parameters	NMR type Shift Zero threshold Input range	^1H – 32k data points ± 1 Hz 10 4.4 – 3.2 ppm
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	2620 input neurons 100 hidden neurons 5 output neurons	
Output coding	0.75 = activated / 0.25 = deactivated	
Training cycles	100	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (± 0.5)	
Learning rate	0.1	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

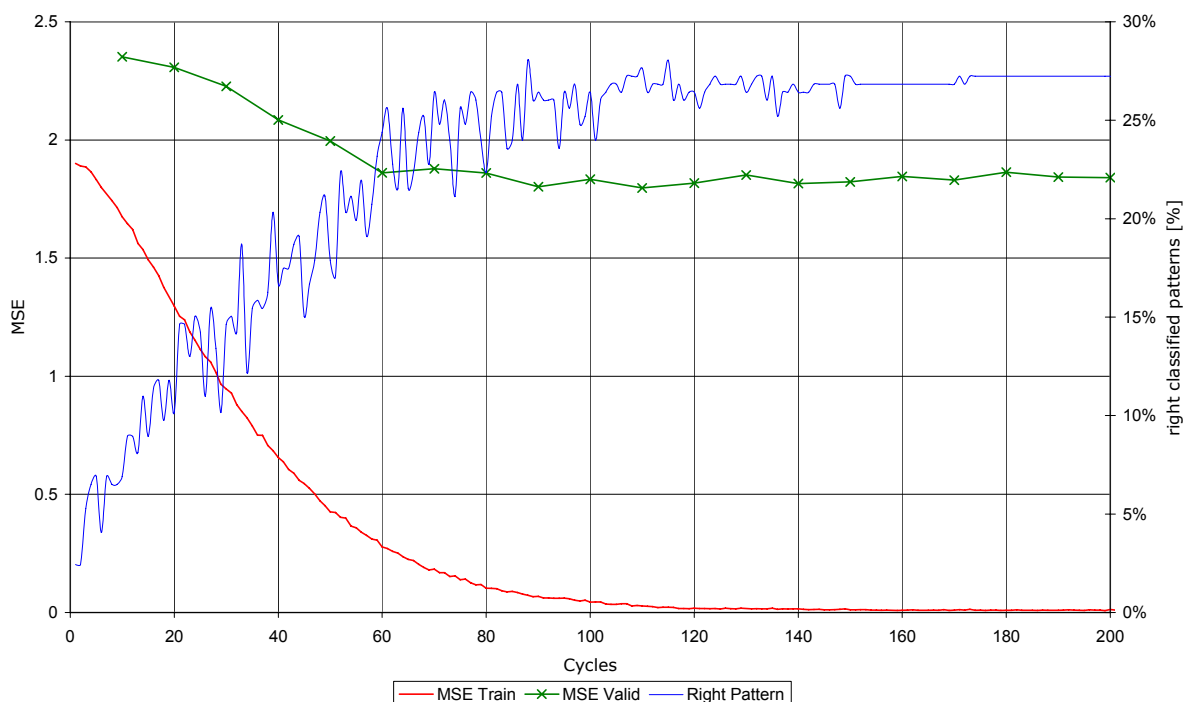


Figure 85: Combined MSE and classification comparison - patterns without methyl-peak

The reduced hidden layer size improves the generalization rate to 27%. Deeper inspection of the weights connecting the input with the hidden layer showed that the trained neural network only considered the methyl peak at 3.4 ppm. The remaining peaks of the ^1H spectra were ignored, respectively the weights in these regions were all set to a level between 0.85 and 0.93.

5.3.1.5. Classification comparison of networks without methyl peaks at 3.4ppm

To avoid the findings of the previous experiment, all methyl peaks at 3.4 ppm were manually deleted in the training and valid pattern files. The network was trained with the same parameters used in experiment 5.3.1.2.

ANN PFG V.0.1 parameters	NMR type Shift Zero threshold Input range	^1H – 32k data points ± 1 Hz 10 4.4 - 3.2 ppm (without 3.4 ppm)
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	2620 input neurons 100 hidden neurons 5 output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (± 0.5)	
Learning rate	0.1 and 0.2	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

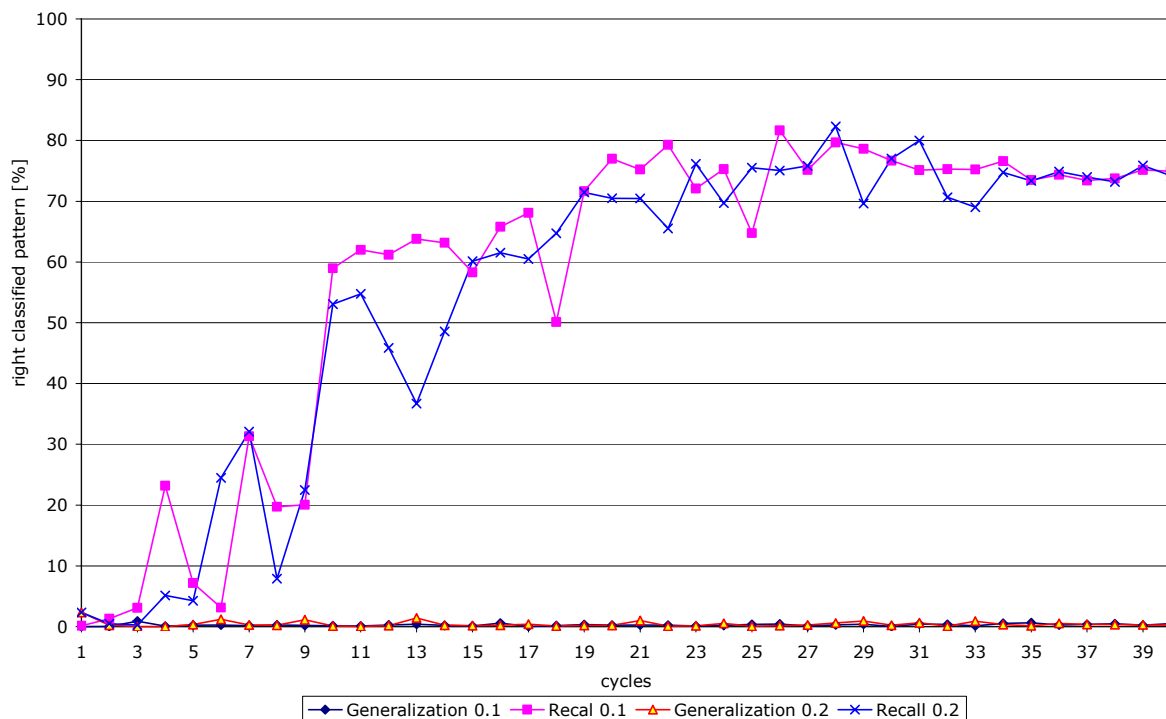


Figure 86: Classification comparison of networks without methyl peaks at 3.4ppm

Surprisingly, the trained neural network was unable to correctly predict more than two test compounds. The two recall curves flatten after about 20 training cycles. This indicates that the network is unable to find an approximation function for the given in- output training pairs.

5.3.2. Conclusion

According to the conducted experiments in the last chapter allow drawing the following conclusions are drawn:

- The learning rate does not seem to have an influence on the generalization rate of the trained neural networks.
- Networks with more than 200 hidden units cannot distinctively improve the generalization rate.
- It is not possible to train a neural network with a satisfying generalization rate with only five compounds. Either the amount of training compounds has to be extended or the size of the used neural network has to be reduced.
- The conclusions from experiment 5.3.1.5 suggest to abandon the ^1H -NMR approach and switch to ^{13}C -NMR spectra. This consideration is supported by the fact that protons in a ^1H -NMR spectrum interact with each other. However, to correctly identify monosaccharides, one only needs the carbon-"core" and the substitution pattern. This information is best accessibly via ^{13}C -NMR data.
- To accomplish the new targets, a new version of the ANN PFG was necessary.

5.4. ^{13}C -NMR experiments

5.4.1. Used dataset

The underlying ^{13}C -NMR data was obtained from literature [73, 96, 97, 101, 103, 106, 107, 109, 113, 119, 121, 129, 132, 133, 135-137, 145, 149, 151, 155-157, 159, 162, 164, 166, 174-177, 181, 183, 188, 191, 193, 202, 210, 211, 221, 260, 290, 303]. The data set contained 535 monosaccharides consisting of 55 different monosaccharide moieties (Table 13). To prepare the processing with the ANN PFG V.0.2 the, the average peak values of each group were calculated and saved into 55 separate CSV files. Afterwards, these files were converted into JCAMP DX files (32k and no Gaussian noise).

To enlarge the dataset, the JCAMP DX generator V.0.2 created 200 modifications (JCAMP DX files) for each monosaccharide-group. Thereof 40 modifications were used as test data and the remaining 160 modifications were saved into the training pattern file. The parameters used for the ANN PFG V.0.2 are indicated at the respective experiment.

Table 13: Monosaccharide moieties contained in the first ¹³C-NMR literature data set

Nr.	Monosaccharide moiety	Nr.	Monosaccharide moiety
1	<i>a-D-pMan-1R</i>	29	<i>b-D-pGal-OH-6R</i>
2	<i>a-D-pMan-OH-2R</i>	30	<i>b-D-pGal-OMe</i>
3	<i>a-D-pMan-OH-4R</i>	31	<i>b-D-pGal-OMe-2R</i>
4	<i>a-D-pMan-OH-6R</i>	32	<i>b-D-pGal-OMe-3R</i>
5	<i>a-D-pMan-OMe</i>	33	<i>b-D-pGal-OMe-6R</i>
6	<i>a-D-pMan-OMe-2R</i>	34	<i>a-D-pGlc</i>
7	<i>a-D-pMan-OMe-3R</i>	35	<i>a-D-pGlc-1R</i>
8	<i>a-D-pMan-OMe-4R</i>	36	<i>a-D-pGlc-OH-2R</i>
9	<i>a-D-pMan-OMe-6R</i>	37	<i>a-D-pGlc-OH-3R</i>
10	<i>b-D-pMan-1R</i>	38	<i>a-D-pGlc-OH-4R</i>
11	<i>b-D-pMan-OH-2R</i>	39	<i>a-D-pGlc-OH-6R</i>
12	<i>b-D-pMan-OH-4R</i>	40	<i>a-D-pGlc-OMe</i>
13	<i>b-D-pMan-OH-6R</i>	41	<i>a-D-pGlc-OMe-2R</i>
14	<i>b-D-pMan-OMe</i>	42	<i>a-D-pGlc-OMe-3R</i>
15	<i>b-D-pMan-OMe-2R</i>	43	<i>a-D-pGlc-OMe-4R</i>
16	<i>b-D-pMan-OMe-4R</i>	44	<i>a-D-pGlc-OMe-6R</i>
17	<i>a-D-pGal-1R</i>	45	<i>b-D-pGlc</i>
18	<i>a-D-pGal-OH-3R</i>	46	<i>b-D-pGlc-1R</i>
19	<i>a-D-pGal-OH-4R</i>	47	<i>b-D-pGlc-OH-2R</i>
20	<i>a-D-pGal-OH-6R</i>	48	<i>b-D-pGlc-OH-3R</i>
21	<i>a-D-pGal-OMe</i>	49	<i>b-D-pGlc-OH-4R</i>
22	<i>a-D-pGal-OMe-2R</i>	50	<i>b-D-pGlc-OH-6R</i>
23	<i>a-D-pGal-OMe-3R</i>	51	<i>b-D-pGlc-OMe</i>
24	<i>a-D-pGal-OMe-4R</i>	52	<i>b-D-pGlc-OMe-2R</i>
25	<i>a-D-pGal-OMe-6R</i>	53	<i>b-D-pGlc-OMe-3R</i>
26	<i>b-D-pGal-1R</i>	54	<i>b-D-pGlc-OMe-4R</i>
27	<i>b-D-pGal-OH-3R</i>	55	<i>b-D-pGlc-OMe-6R</i>
28	<i>b-D-pGal-OH-4R</i>		

5.4.2. Comparison of different Back-propagation learning algorithms

ANN PFG V.0.2 parameters	ppm range Zero threshold Input scaling Shift Raster Binary patterns	100 - 15 ppm 0% 1.0 ± 1 Hz 5 points yes/no
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	6225 input neurons 100 hidden neurons 55 output neurons	
Training cycles	280	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (± 0.5)	
Learning rate	0.1 - 1	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

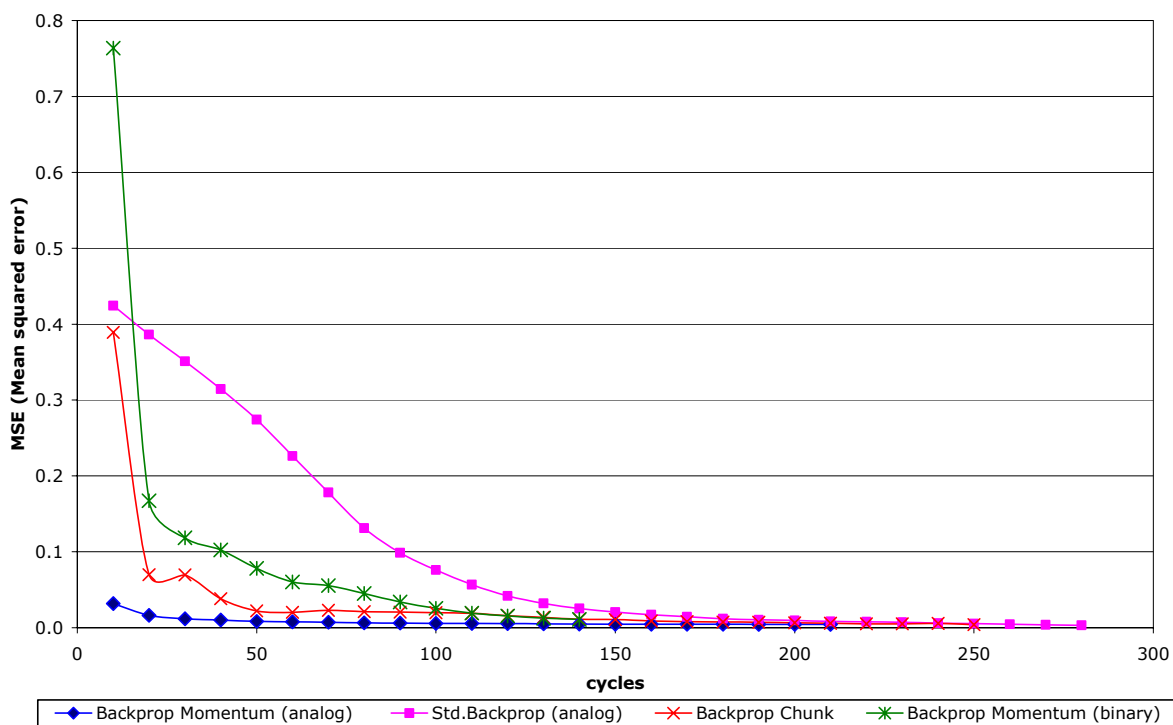


Figure 87: Different Backprop learning algorithms overview

As depicted in Figure 87, the different modifications of the Back-propagation learning algorithm tend to find the same solution and the MSE drops asymptotically after approx. 150 cycles to zero. The Back-propagation modification with a momentum term finds the minimum fastest. It is not possible to testify if binary or analog input coding should be used. This will be explored by further separate experiments.

5.4.3. Comparison of different learning rates

ANN PFG V.0.2 parameters	ppm range Zero threshold Input scaling Shift Raster Binary patterns	100 - 15 ppm 0% 1.0 ± 1 Hz 5 points yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	6225 input neurons 100 hidden neurons 55 output neurons	
Training cycles	up to 30	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (0.25 - 0.75)	
Learning rate	0.3	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

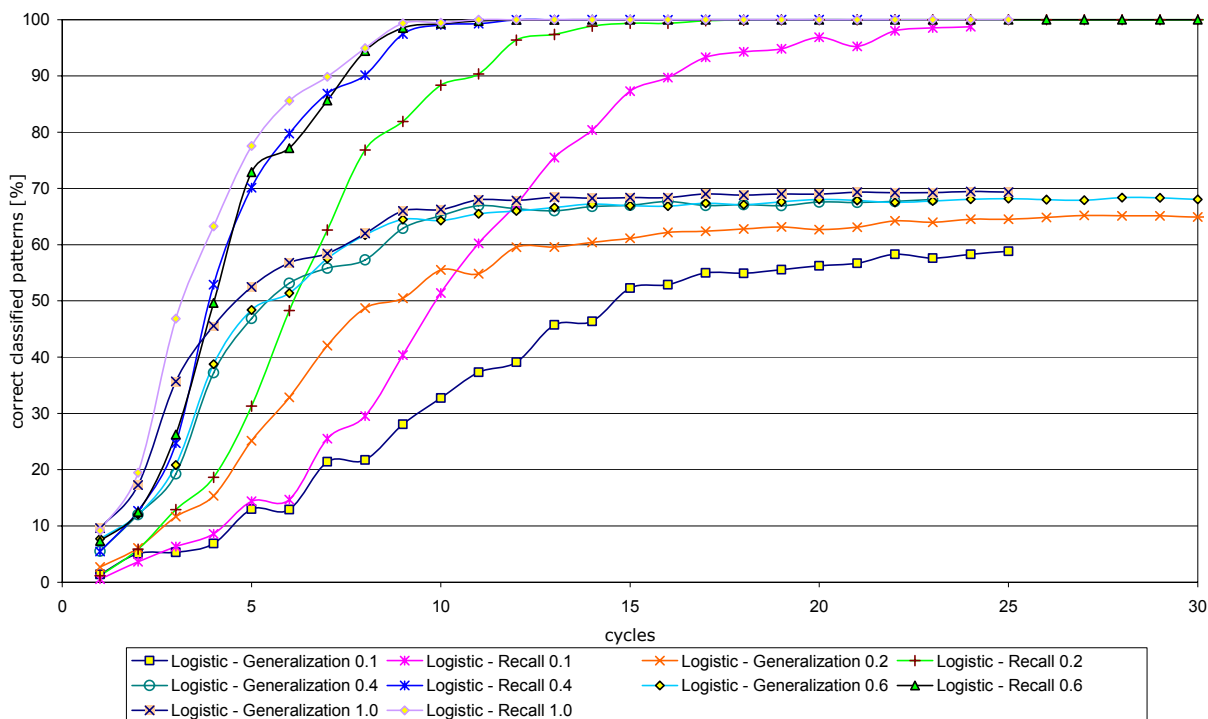


Figure 88: Learning rate comparison with a logistic transfer function and a fixed hidden layer size of 100 neurons

The results show a similar picture as in Figure 81. A linear increase of the learning rate does not result in a similar increase in generalization rate. Therefore, in the following experiments the learning rate will be set > 0.4 . The relatively low generalization rates still indicates fundamental network topology or data set problems. With the following experiments we tried to cover all possible solutions.

5.4.4. Comparison of different learning rates at 600 hidden units

ANN PFG V.0.2 parameters	ppm range	100 - 15 ppm
	Zero threshold	0%
	Input scaling	1.0
	Shift	± 1 Hz
	Raster	5 points
	Binary patterns	yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	6225 input neurons 600 hidden neurons 55 output neurons	
Training cycles	25	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (0.25 - 0.75)	
Learning rate	0.2 and 0.4	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

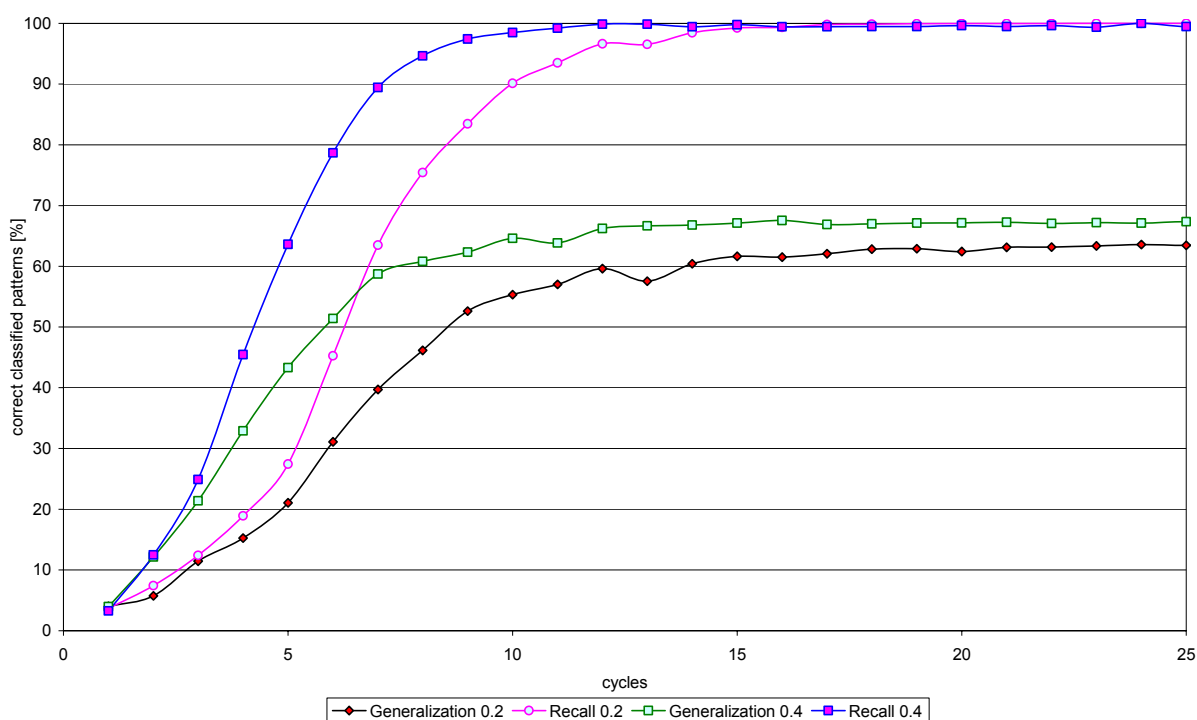


Figure 89: Learning rate comparison with a logistic transfer function and fixed hidden layer size of 600 neurons

An increase of the hidden layer size to 600 hidden units does not solve the problem. The generalization and recall rates climb approximately to the same levels as in the previous experiments. The only major difference is the time it takes to reach the plateau; ~15 cycles with 600 hidden units and ~10 cycles with 100 hidden units. As the training time rises with the number of weights and connections, it is not advisable to choose networks with large hidden layer sizes (already discussed in 5.3.1.2).

5.4.5. Hidden layer size comparison with additional noise

Many publications [64, 65, 77, 80-82, 302, 304-307] propose to add artificial noise (=jitter) to the input values of the training data to improve generalization with small training sets.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm 20% 0% 1.0 ± 1 Hz 5 points no yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	6225 input neurons variable hidden neurons 55 output neurons	
Training cycles	40	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (0.25 - 0.75)	
Learning rate	0.4	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

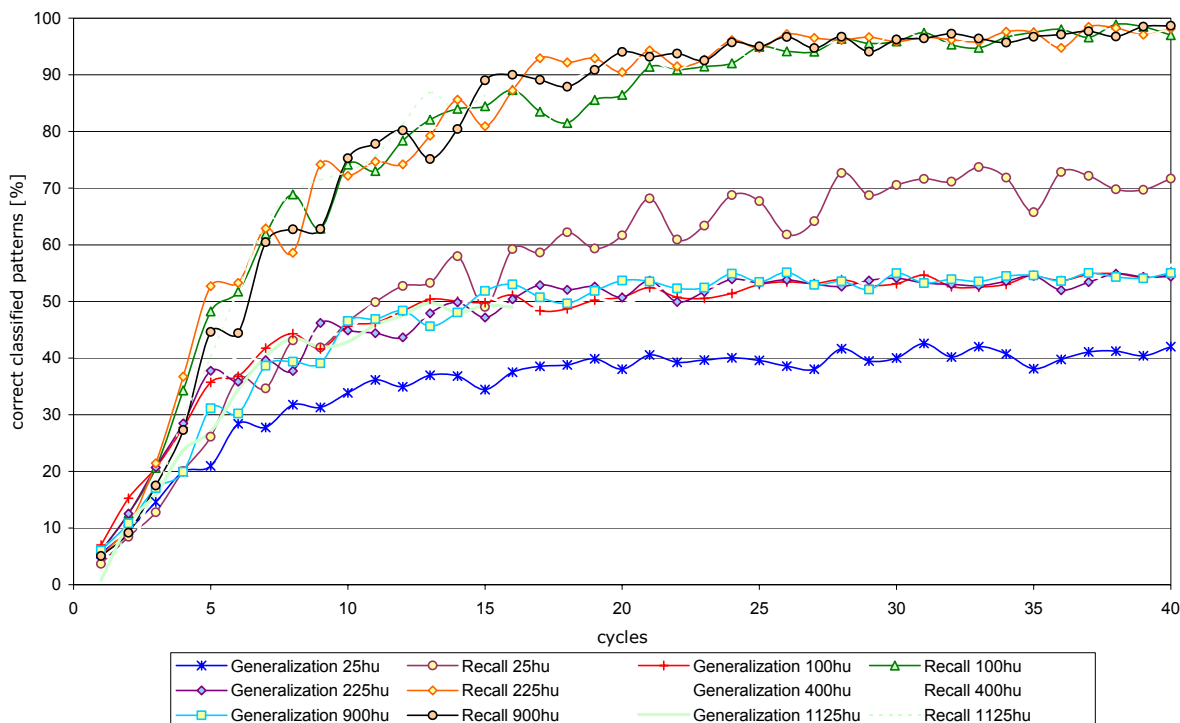


Figure 90: Hidden layer size comparison with additional noise and block pattern

The addition of 20% artificial noise seems to have a contra productive effect on the recall and the generalization rate. The saturation point is shifted back to higher cycles. The graph shows again, that the network size has a minor influence on the performance of the neural network. A hidden layer size of about 100 hidden units is sufficient for passable results.

5.4.6. Hidden layer size comparison without additional noise and block-pattern

Since it was shown that noise did not have a positive influence on the results, this approach was given up. Therefore the newly proposed block-pattern approach (explained and presented in chapter 4.9.5.3) was used for the first time. The size of the input layer could be reduced from 6225 to 4280 neurons by increasing the raster size from five to 10 points.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm none 0% 1.0 ± 1 Hz 10 points yes yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	4280 input neurons variable hidden neurons 55 output neurons	
Training cycles	40	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (0.25 - 0.75)	
Learning rate	0.4	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

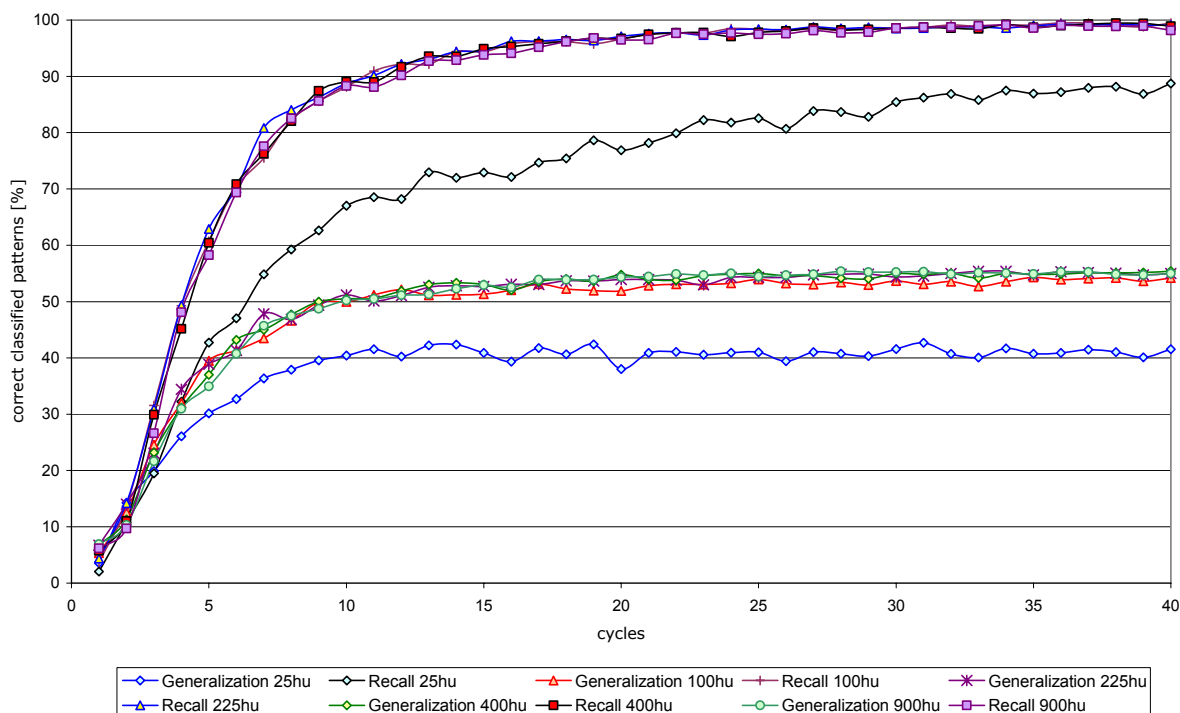


Figure 91: Hidden layer size comparison with a logistic transfer function, without noise and block pattern

Figure 91 shows an increased homogeneity of the calculated recall and generalization rates. The networks seem to be insensible against minor changes of the hidden layer size. The results confirm the suspicion, that all learning parameters have only a minor influence on the generalization performance of the trained neural networks. Good training results seem to depend on an equalized and big enough dataset.

5.4.7. Classification comparison of different initial weight initialization values

As proposed in literature^[232, 302], the next adjustable training parameters is the choice of the weight initialization value. This value should be chosen according to the activation function – the values should lie in the defined range of the function. For testing purpose the range of tested init values ranges from ± 0.001 to ± 1.3 , even though the sinus function is not defined at values of 1.3.

The hidden layer size was set to 100 hidden units, because of the results from the previous experiments. In addition, the block patterns were maintained.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm none 0% 1.0 ± 1 Hz 5 points yes yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	4280 input neurons 100 hidden neurons 55 output neurons	
Training cycles	25	
Activation function	Sinus (as described in Figure 18e)	
Output function	Identity	
Init function	variable from ± 0.001 – ± 1.3	
Learning rate	0.5	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

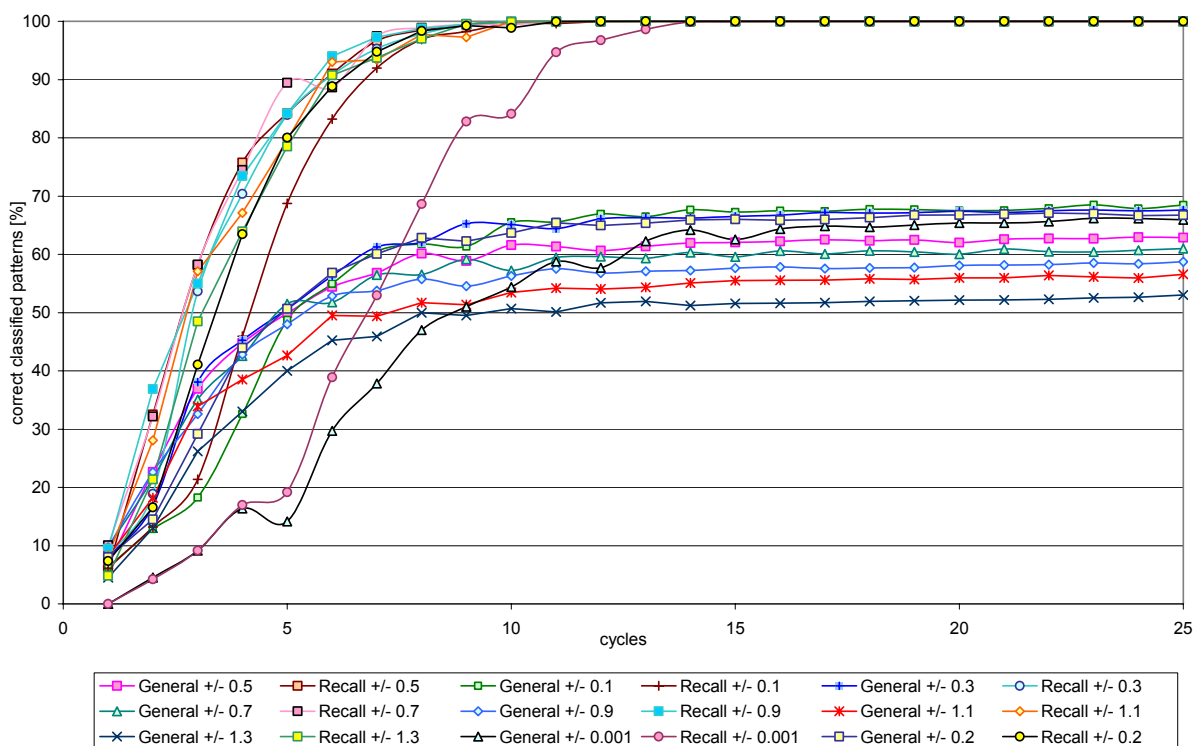


Figure 92: Weight init values comparison at a learning rate 0.4, sinus transfer function, without noise and fixed hidden layer size of 100 neurons

The achieved generalization rate almost reaches 70% again. The optimal initialization values seem to be ± 0.1 . Higher values decrease the generalization rate. Values $< \pm 0.1$ reach the same rate, but it takes comparatively longer (more cycles) until the curve climbs to the same level. This experiment proves the results from experiment 5.4.3.

5.4.8. MSE comparison with different initial weight initialization values

For the sake of completeness, the experiment 5.4.7 was repeated but only the MSE values were recorded and are displayed in Figure 93.

ANN PFG V.0.2 parameters	ppm range	100 - 15 ppm
	Noise	none
	Zero threshold	0%
	Input scaling	1.0
	Shift	± 1 Hz
	Raster	5 points
	Block pattern	yes
	Binary patterns	yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	4280 input neurons 100 hidden neurons 55 output neurons	
Training cycles	25	
Activation function	Sinus (as described in Figure 18e)	
Output function	Identity	
Init function	variable from $\pm 0.001 - \pm 1.3$	
Learning rate	0.4	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

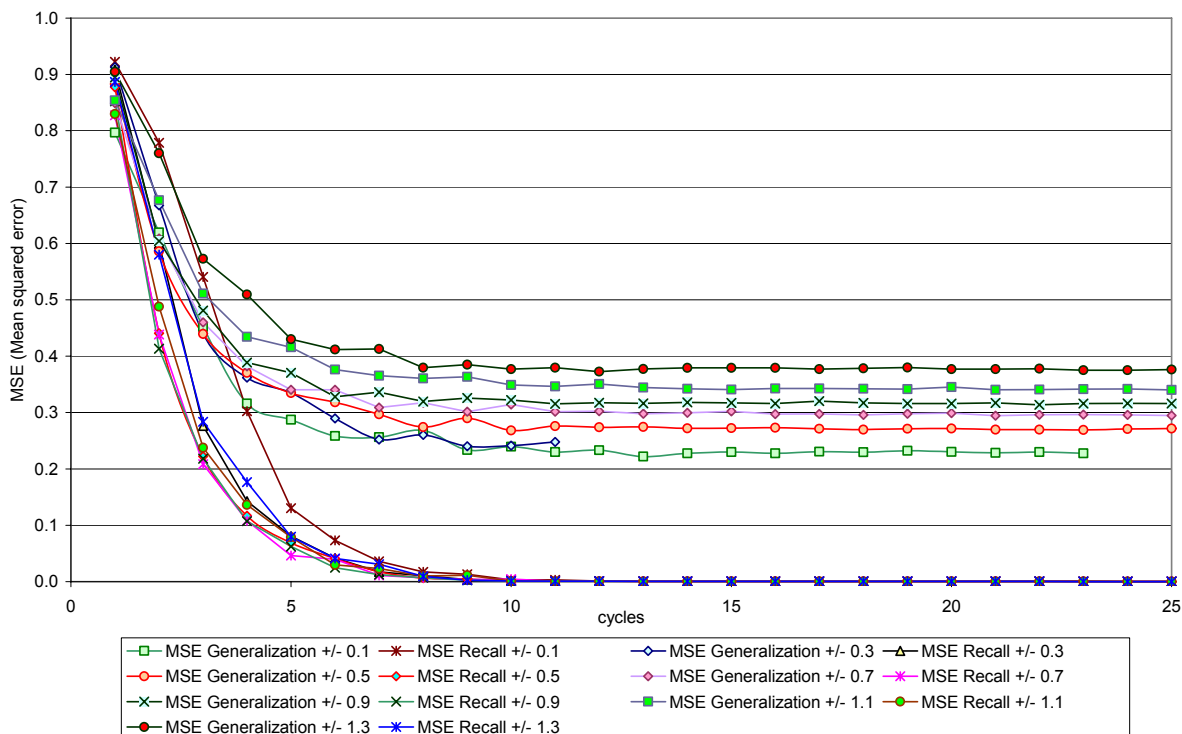


Figure 93: Weight init values comparison with Backprop Momentum at a learning rate 0.4, logistic transfer function, without noise and fixed hidden layer size of 100 neurons

5.4.9. Hidden layer size comparison at learning rate 0.2

To exclude the possibility of the Back-propagation algorithm being trapped in a local minimum, another set of six networks was trained with the Back-propagation momentum algorithm. This modification of the Back-propagation algorithm showed good and particularly fast results in experiment 5.4.2.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm none 0% 1.0 ± 1 Hz 5 points no yes
Software	SNNS V4.2	
Training algorithm	Back-propagation momentum	
Network size	6225 input neurons variable hidden neurons 55 output neurons	
Training cycles	25	
Activation function	Logistic (as described in Figure 18f)	
Output function	Identity	
Init function	Random (± 0.5)	
Learning rate	0.2	
Momentum term	0.5	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

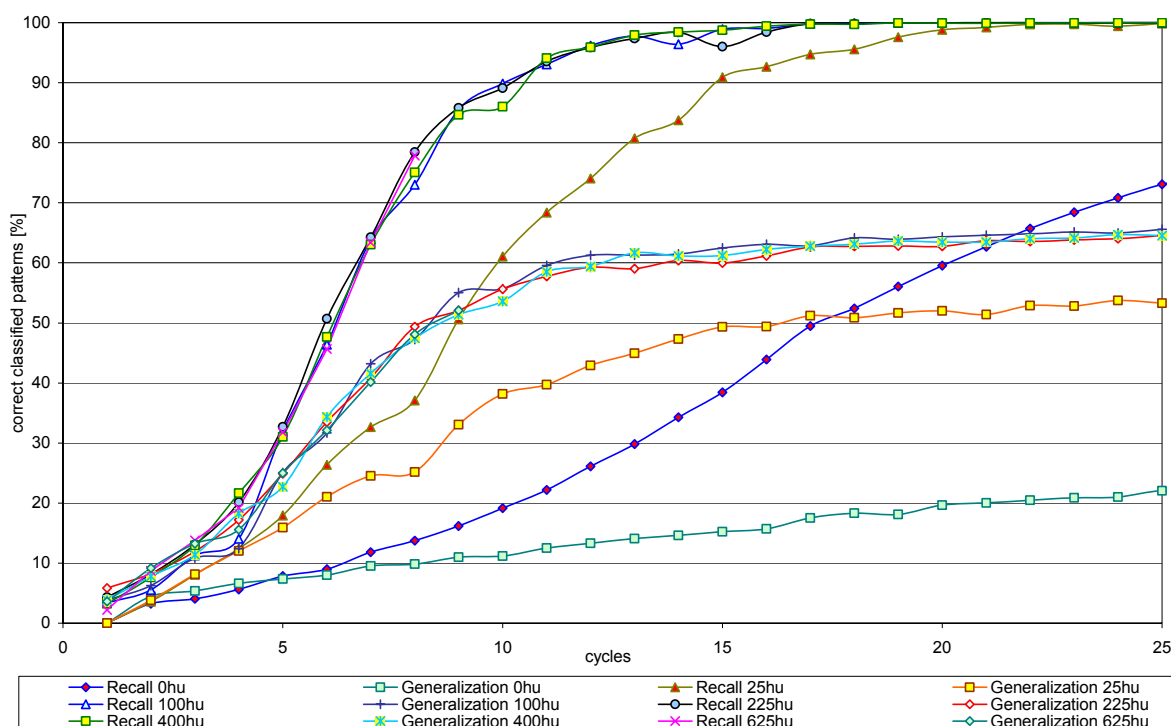


Figure 94: Comparison of different hidden layer sizes with Backprop Momentum at a fixed learning rate 0.2, logistic transfer function and without noise.

The results did not become better than the average of the preceding experiments. The generalization rate of most tested networks exceeded the 64% limit only marginal. The lower generalization rate could be led back on the block pattern subprogram that was not activated in the ANN PFG.

5.4.10. Hidden layer size comparison at learning rate 0.7 and shift ± 3 Hz

To prevent high recall and low generalization rates the shift size of the ANN PFG V.0.2 was slightly increased to ± 3 Hz. The idea of this approach was to generate a bigger variability of the training patterns. These pattern should now activate a wider range of input neurons of each peak in the original NMR peak list. The effect of memorization (high recall rates) should be avoided.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm none 0% 1.0 ± 3 Hz 5 points no yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	4280 input neurons variable hidden neurons 55 output neurons	
Training cycles	40	
Activation function	StepFunc (as described in Figure 18d)	
Output function	Identity	
Init function	Random (± 0.5)	
Learning rate	0.7	
Momentum term	0.2	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

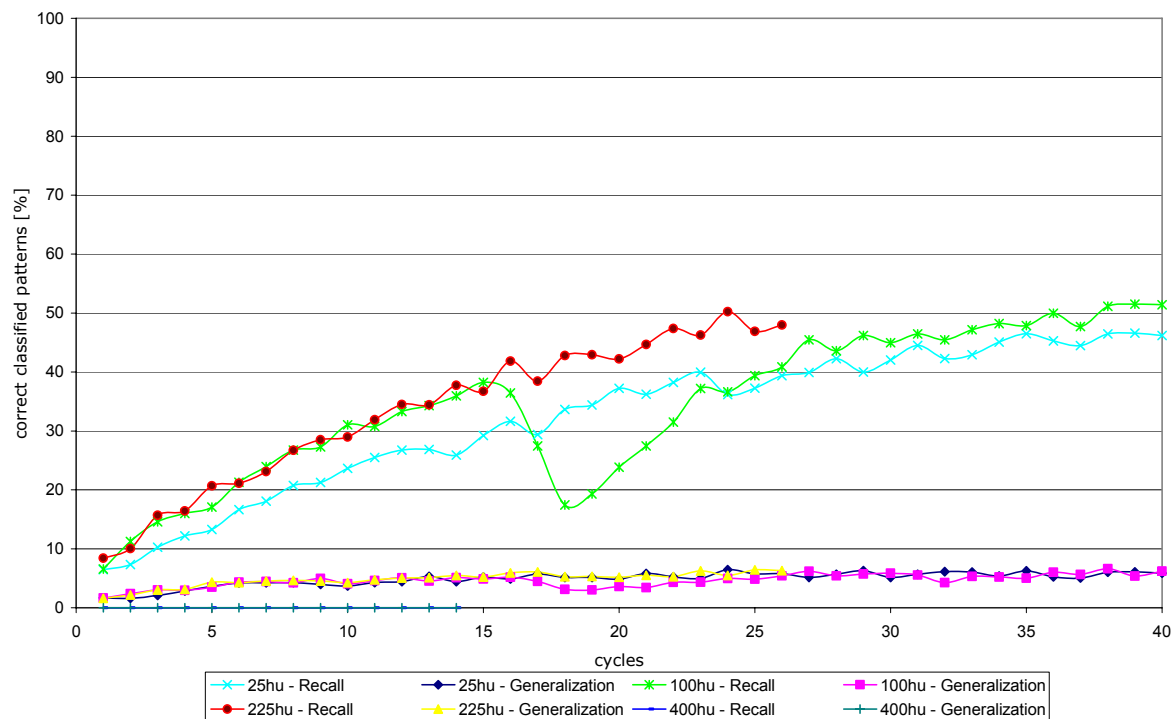


Figure 95: Comparison of different hidden layer sizes with Backprop Momentum at a fixed learning rate 0.7, StepFunc transfer function and binary patterns.

As depicted in Figure 95, the experiment was abandoned because of very bad generalization results. The generalization rate may climb to levels reached in former experiments, but the amount of time needed to eventually reach generalization rates > 60% is disproportional.

5.4.11. Learning rate comparison without hidden layer and binary input patterns

Another approach would be to test, if the classification problem is linearly separable. This kind of problems can be best approached with neural networks without hidden layers. For this purpose, the training and test pattern were written in binary from with the ANN PFG V.0.2. Every peak exceeding the 30% threshold is coded as 1. Peaks below the threshold are coded as 0. With the use of a step function as activation function, the network is binary.

ANN PFG V.0.2 parameters	ppm range Noise Zero threshold Input scaling Shift Raster Block pattern Binary patterns	100 - 15 ppm none 30% 1.0 ± 1 Hz 5 points yes yes
Software	SNNS V4.2	
Training algorithm	Standard Back-propagation	
Network size	4280 input neurons 0 hidden neurons 55 output neurons	
Training cycles	40	
Activation function	StepFunc (as described in Figure 18d)	
Output function	Identity	
Init function	Random (± 0.1)	
Learning rate	0.1 - 1	
Momentum term	0.2	
d_{max}	0.1	
Flat spot elimination value	0.1	
Threshold	0.7	
Pattern shuffling	activated	

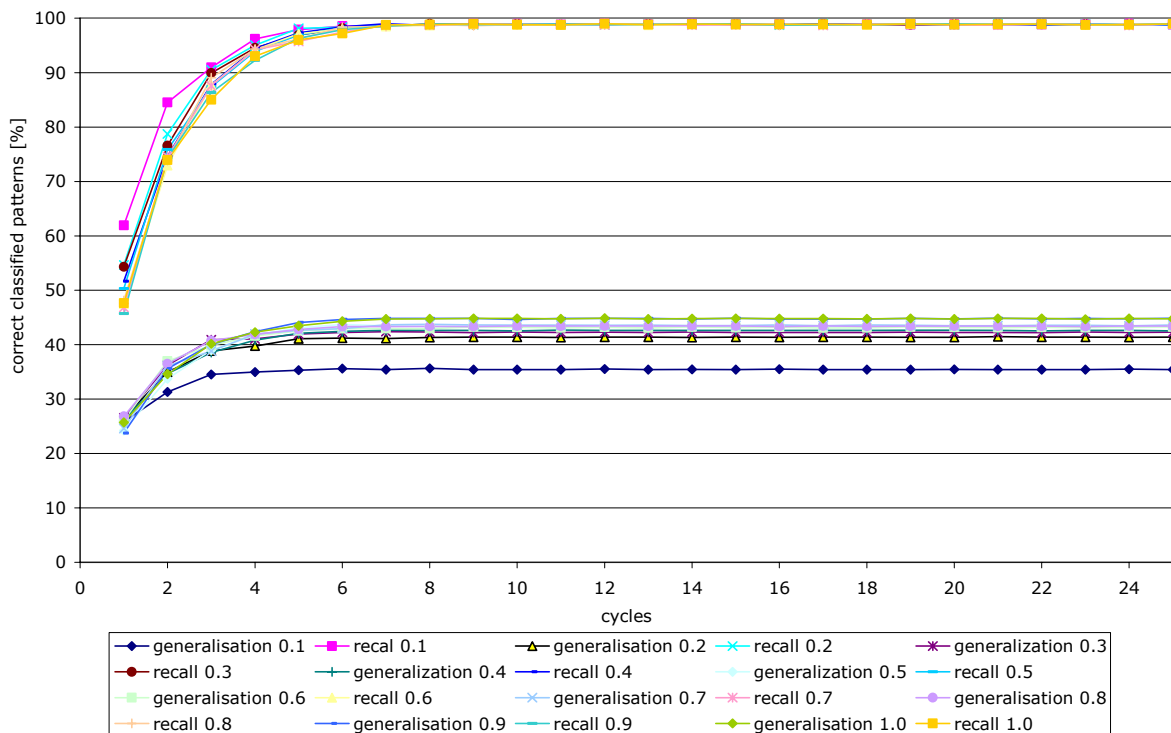


Figure 96: Comparison of different learning rates with Backprop Momentum without hidden layer, StepFunc transfer function and binary patterns.

The results of 5.4.11 are not satisfying. All tested learning rates lead to similar recall and generalization rates. But the generalization rate never exceeds levels $> 45\%$. The task of separating and identifying monosaccharide moieties is only partly accomplished. The fact, that all depicted curves are almost overlaid, indicates that good solutions with networks without hidden layers are independent from the used learning rate. Therefore, the idea of training neural networks without hidden layers was abandoned.

5.4.12. Conclusion

The experiments of this section showed again, that none of the learning parameters could really improve the generalization rate. A point of improvement would be the size of networks. I.e. fewer input units by increasing the reading step size or activating the block pattern subprogram of the ANN PFG. However, this improvement possibility will mostly reduce the training time and not the generalization performance. Therefore, the main attention should be turned to the dataset. The set should be drastically expanded and equalized.

Maybe the training task is "overstrained" with too many classification groups (Table 13) and the training data should be separated into smaller groups. This approach would necessitate to train an own neural networks for each carbohydrate species.

5.5. Diploma work Alexej Moor

5.5.1. Introduction

The main goal of this diploma work was to prove that the information contained in only six ^{13}C -NMR peaks of a monosaccharide (only mannose, glucose and galactose) are sufficient to classify the following properties of a carbohydrate:

- Carbohydrate species (mannose, glucose and galactose)
- Anomeric configuration (α or β)
- Linkage position (if the monosaccharide is linked to another carbohydrate)

These aims should be achieved with a simple supervised or non-supervised learning method. Kohonen feature maps and Counter-propagation networks were taken into close consideration. The most suitable network type and training algorithm should be elicited during this diploma work.

5.5.2. Dataset

The carbohydrate compounds used were found in literature [73, 96, 97, 101, 103, 106, 107, 109, 113, 119, 121, 129, 132, 133, 135-137, 145, 149, 151, 155-157, 159, 162, 164, 166, 174-177, 181, 183, 188, 191, 193, 202, 210, 211, 221, 260, 290, 303]. They were collected in the first version of the FileMaker ^{13}C -NMR database (chapter 5.4.1). The final dataset contained 585 different monosaccharide units with only one, two or three linkage positions. The moieties were randomly subdivided in a training set containing 275 units and a test set containing 323 datasets. Finally we had 46 different monosaccharide units (= groups or output units) (Figure 97).

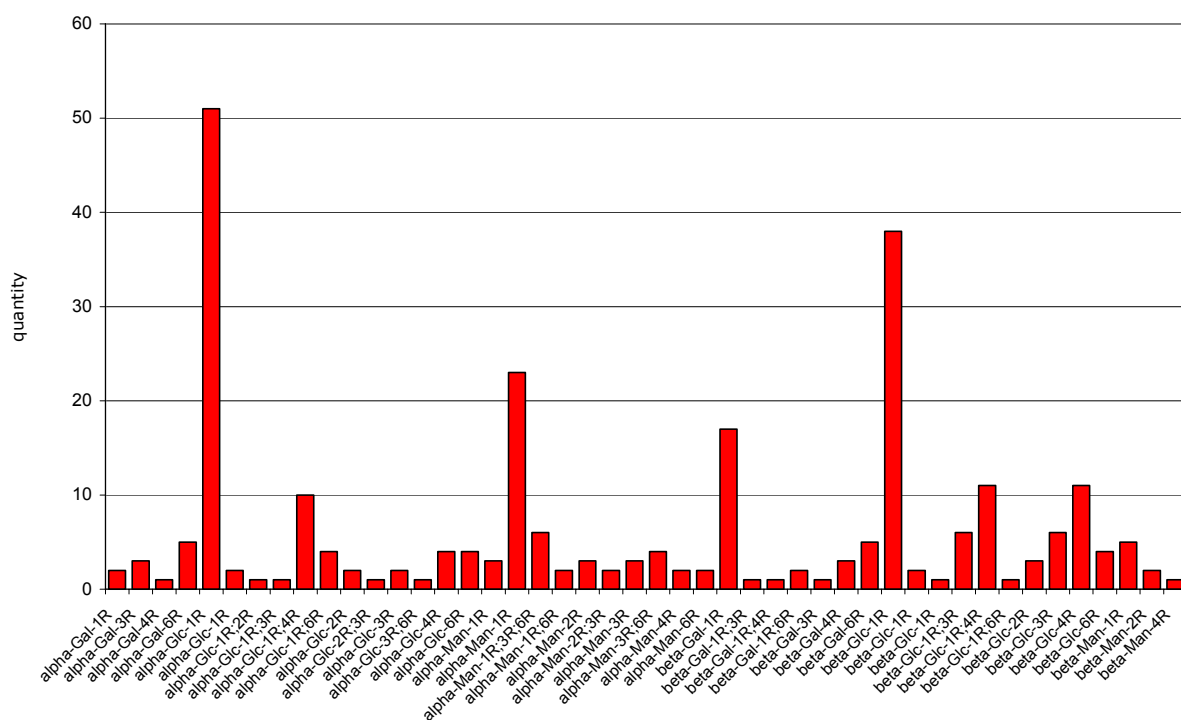


Figure 97: Data distribution of all groups contained in the data set

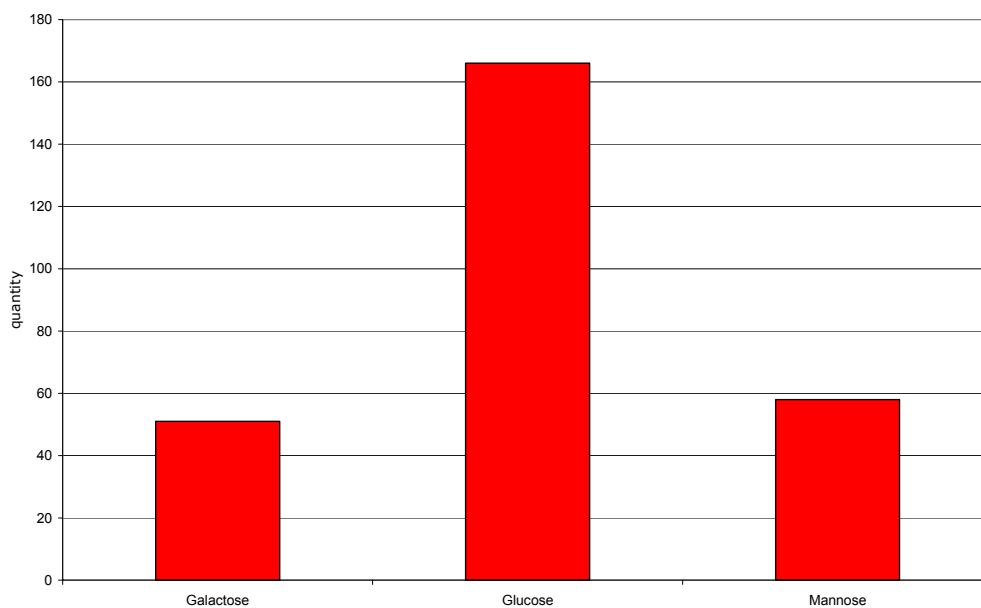


Figure 98: Data set carbohydrate distribution

5.5.3. Experiments & Results

The raw chemical shifts from the peak list (in ppm values) out of the FileMaker ^{13}C -NMR database were used as direct input to the neural network. There was no normalization or remapping of the ppm-values and the peak list was not processed with the ANN PFG.

5.5.3.1. Determination of the anomeric configuration (α / β)

Software	SNNS V4.2
Network size	6 input neurons 16 x 16 Kohonen neurons 2 Output neurons
Output coding	binary (1= activated / 0 = deactivated)
Training cycles	1000
Activation function	Logistic
Output function	Identity
Init function	Random (± 1)
Learn rate for Kohonen layer	0.3
Learn rate for Grossberg layer	0.5
Threshold	0
Pattern shuffling	activated

Table 14: α / β discrimination

	Train (containing 275 patterns)			Test (containing 323 patterns)		
	<i>wrong</i>	<i>unknown</i>	<i>correct</i>	<i>wrong</i>	<i>unknown</i>	<i>correct</i>
Try 1	0.0%	0.0%	100.0%	0.0%	0.0%	100.0%
Try 2	0.0%	0.7%	99.3%	0.0%	0.6%	99.4%
Try 3	0.0%	0.7%	99.3%	0.0%	0.6%	99.4%
Try 4	0.0%	0.0%	100.0%	0.0%	0.0%	100.0%
Try 5	0.0%	0.7%	99.3%	0.0%	0.6%	99.4%
	0.0%	0.4%	99.6%	0.0%	0.4%	99.6%

5.5.3.2. Determination of the carbohydrate identity

Software	SNNS V4.2
Network size	6 input neurons 16 x 16 Kohonen neurons 3 Output neurons
Output coding	binary (1= activated / 0 = deactivated)
Training cycles	1000
Activation function	Logistic
Output function	Identity
Init function	Random (± 1)
Learn rate for Kohonen layer	0.3
Learn rate for Grossberg layer	0.5
Threshold	0
Pattern shuffling	activated

Table 15: Carbohydrate discrimination

	Train (containing 275 patterns)			Test (containing 323 patterns)		
	<i>wrong</i>	<i>unknown</i>	<i>correct</i>	<i>wrong</i>	<i>unknown</i>	<i>correct</i>
Try 1	0.0%	0.7%	99.3%	0.0%	1.9%	98.1%
Try 2	0.0%	0.7%	99.3%	0.0%	0.6%	99.4%
Try 3	0.0%	0.7%	99.3%	0.0%	0.9%	99.1%
Try 4	0.0%	0.0%	100.0%	0.3%	0.0%	99.7%
Try 5	0.0%	1.1%	98.9%	0.0%	2.2%	97.8%
Try 6	0.0%	0.0%	100.0%	0.0%	0.3%	99.7%
Try 7	0.0%	0.0%	100.0%	0.0%	0.3%	99.7%
	0.0%	0.5%	99.5%	0.0%	0.9%	99.1%

5.5.3.3. Linkage determination

Software	SNNS V4.2
Network size	6 input neurons 16 x 16 Kohonen neurons 12 Output neurons
Output coding	binary (1= activated / 0 = deactivated)
Training cycles	1000
Activation function	Logistic
Output function	Identity
Init function	Random (± 1)
Learn rate for Kohonen layer	0.3
Learn rate for Grossberg layer	0.5
Threshold	0
Pattern shuffling	activated

Table 16: Linkage discrimination

	Train (containing 275 patterns)			Test (containing 323 patterns)		
	<i>wrong</i>	<i>unknown</i>	<i>correct</i>	<i>wrong</i>	<i>unknown</i>	<i>correct</i>
Try 1	0.0%	1.8%	98.2%	5.3%	1.9%	92.9%
Try 2	0.0%	0.7%	99.3%	5.3%	0.6%	94.1%
Try 3	0.0%	1.5%	98.5%	5.3%	0.0%	94.7%
Try 4	0.0%	0.7%	99.3%	5.3%	0.6%	94.1%
Try 5	0.0%	1.5%	98.5%	5.6%	1.2%	93.2%
	0.0%	1.2%	98.8%	5.3%	0.9%	93.8%

5.5.3.4. Combination of all used features (groups)

Software	SNNS V4.2
Network size	6 input neurons 16 x 16 Kohonen neurons 46 Output neurons
Output coding	binary (1= activated / 0 = deactivated)
Training cycles	1000
Activation function	Logistic
Output function	Identity
Init function	Random (± 1)
Learn rate for Kohonen layer	0.3
Learn rate for Grossberg layer	0.5
Threshold	0
Pattern shuffling	activated

Table 17: Combination discrimination

	Train (containing 275 patterns)			Test (containing 323 patterns)		
	<i>wrong</i>	<i>unknown</i>	<i>correct</i>	<i>wrong</i>	<i>unknown</i>	<i>correct</i>
Try 1	0.0%	2.9%	97.1%	6.5%	0.9%	92.6%
Try 2	0.0%	2.9%	97.1%	5.9%	3.7%	90.4%
Try 3	0.0%	0.7%	99.3%	7.1%	0.0%	92.9%
Try 4	0.0%	1.5%	98.5%	6.8%	0.6%	92.6%
Try 5	0.0%	1.5%	98.5%	5.3%	3.7%	91.0%
	0.0%	1.9%	98.1%	6.3%	1.8%	91.9%

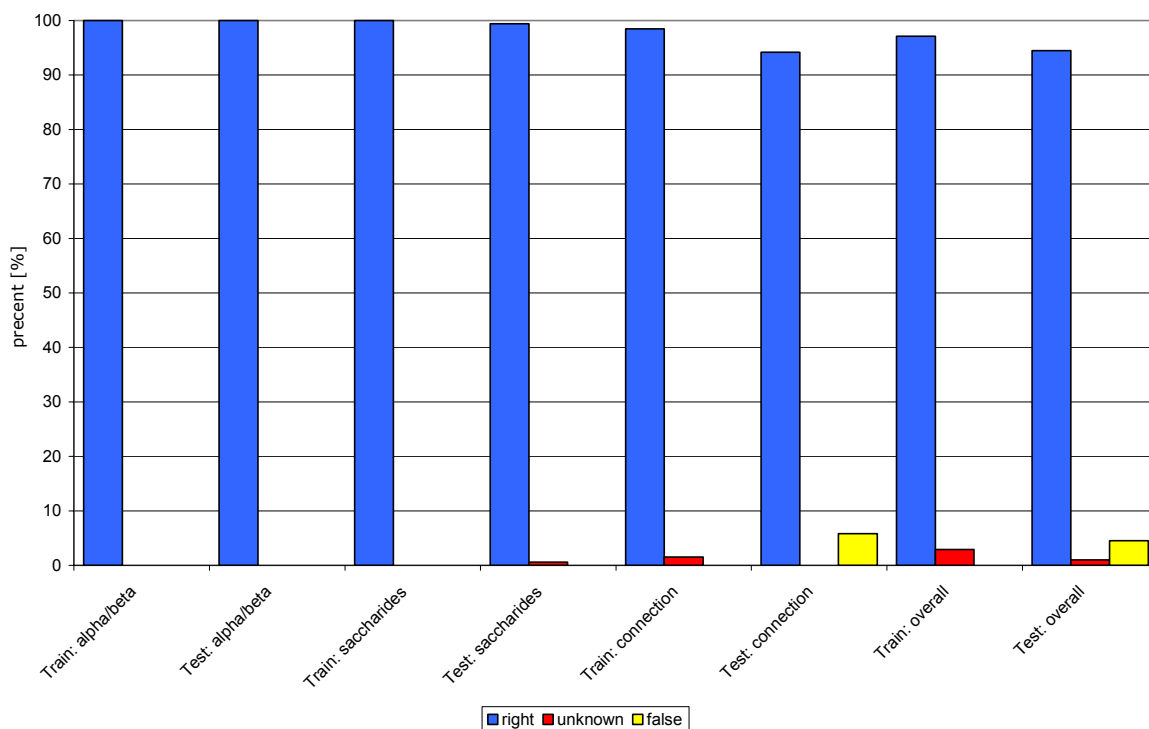


Figure 99: Graphical results overview

With these results, the following decision tree (Figure 100) for monosaccharide units was proposed. The Counter-propagation network with the best separation quality (α / β discrimination –Figure 13) will be used as a first entity to separate the test data in a first run. The following downstream Counter-propagation networks will be specially trained to recognize the carbohydrate identity and the linkage pattern of a monosaccharide unit. In this way, it should be possible to achieve a very high hit rate of >90% correct classified monosaccharide units (Table 17).

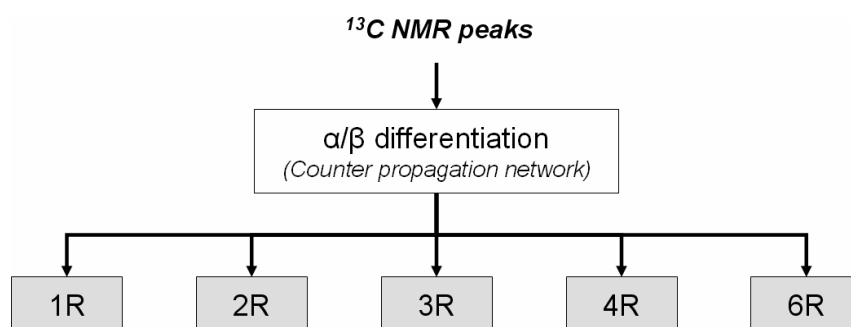


Figure 100: Proposed counter propagation networks decision tree for automated monosaccharide moiety identification

5.5.4. Discussion & conclusions

The presented results generated with the proposed decision tree show clearly, that an automated separation of the underlying dataset is possible. Different learning rates and initial random weights do not play an important role. Six ^{13}C -NMR peaks contain enough information to make a reliable assignment. The approach shows also that Kohonen and Counter-propagation networks are suitable for this task. The monosaccharide moieties of the compounds OH1, OH6, OH7, OH8 and OH9 from Ole Hindsgaul (chapter 4.1.2) were correctly classified. However, with this approach it will never be possible to process other saccharides than monosaccharides because the Kohonen layer of the first network contains only six input units and cannot deal with variable input data (of e.g. disaccharides or carbohydrates with more than six peaks in the peak list).

The work clearly proofed that the underlying data set still has to be enlarged to form a sufficiently big training and test set. The diploma work also highlighted that there are numerous mistakes published in ^{13}C -NMR peak lists in literature.

In all following experiments, only separated neural networks for each monosaccharide species (glucose, galactose and mannose) will be trained.

5.6. Introduction of FileMaker ^{13}C -NMR database

All the following experiments are based on the fully expanded FileMaker ^{13}C -NMR database explained in chapter 4.1.5.

5.7. Kohonen feature maps

The main purpose of the following experiments was to check if all GAM monosaccharide moieties can be separated by a neural network. The approach would also prove the robustness of our classification scheme. The new pattern files for Statsoft Statistica will be based on the results of the following experiments.

Another positive effect of the Kohonen networks is the possibility to find errors in the peak lists of the FileMaker ^{13}C -NMR database. These errors can originate from the literature or from the human data entry into the database.

5.7.1. Decay factor

The decay factor is a number, by which the learning and the neighboring function are multiplied (decreased) after every training cycle. As the factor is < 1 , the decay is dropping asymptotically against zero. Kohonen feature maps trained with small decay factors ($d < 0.5$) are very fast in training but often don't find a good local or the global minimum. The best decay factor for each trained Kohonen feature map was determined experimentally in advance by analyzing the separation ability after 5'000 training cycles at a time. These preliminary tests are not published. For details about Kohonen networks see chapter 4.5.3.

With the following equations (Equation 18 and Equation 19) it is possible to calculate the necessary decay factor for a given amount of learning cycles. Or the training cycles needed with a fixed decay rate.

$$d = -\frac{1}{c} + 1 \quad \text{Equation 18: Kohonen feature map decay factor calculation}$$

$$c = \frac{1}{1-d} \quad \text{Equation 19: Kohonen feature map cycles calculation}$$

d = decay factor
 c = planned training cycles

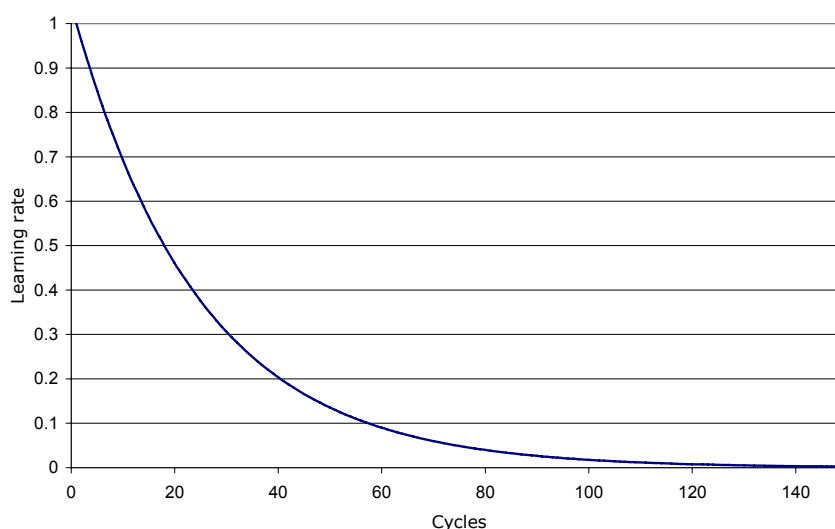


Figure 101: Sample decay curve

5.7.2. Data preparation

All peaks lists of the all monosaccharide moieties in the FileMaker ^{13}C -NMR database (final version) containing glucose, galactose and mannose were exported via ODBC directly into a Microsoft Excel spreadsheet. Monosaccharide moieties belonging together were merged (Table 23: Special characteristics and cohesions of the mannose Kohonen feature map). The whole data set finally contained 69 different monosaccharide moieties (= groups). From every group the average peak values and the associated standard deviation were calculated. The average ppm-peak-values were then randomly shifted ten times in the range of the according standard deviation. This led to an evenly distributed data set containing 690 peak lists of 69 monosaccharide moieties. The Excel spreadsheet was then saved into a CSV file and processed with the ANN PFG. The used parameters for the ANN PFG and SNNS are indicated under the following respective experiments.

5.7.3. Galactose

ANN PFG parameters	NMR type Step size Zero threshold Input coding	¹³ C – 32k data points 15 0 binary
Software	SNNS V4.2	
Network size	261 input neurons 20 x 20 Kohonen neurons 27 Output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100'000	
Activation function	Logistic	
Output function	Identity	
Init function	Random (±1)	
Kohonen adaptation height	0.3	
Kohonen adaptation radius	0.5	
Height decrease factor	0.99999	
Radius decrease factor	0.99999	
Horizontal size	20	
Threshold	0	
Pattern shuffling	activated	

Table 18: Galactose group allocation

1	4-deoxy-b-D-Galp-OMe-6R	15	a-D-Galp-OMe-4R
2	a-D-Galp-1R	16	a-D-Galp-OMe-6R
3	a-D-GalpA-1R	17	b-D-Galp-1R
4	a-D-GalpNAc-1R	18	b-D-GalpNAc-1R
5	a-D-GalpNAc-OH-6R	19	b-D-Galp-OH
6	a-D-GalpNAc-OMe-3R	20	b-D-Galp-OH-3R
7	a-D-GalpNAc-OMe	21	b-D-Galp-OH-4R
8	a-D-Galp-OH	22	b-D-Galp-OH-6R
9	a-D-Galp-OH-3R	23	b-D-Galp-OMe
10	a-D-Galp-OH-4R	24	b-D-Galp-OMe-2R
11	a-D-Galp-OH-6R	25	b-D-Galp-OMe-3R
12	a-D-Galp-OMe	26	b-D-Galp-OMe-4R
13	a-D-Galp-OMe-2R	27	b-D-Galp-OMe-6R
14	a-D-Galp-OMe-3R		

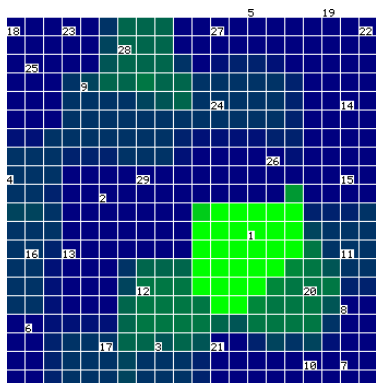


Figure 102: 4-deoxy-b-D-Galp-OMe-6R

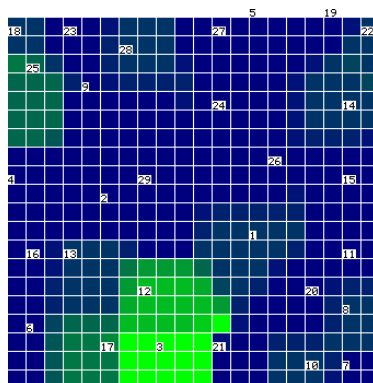


Figure 103: α-D-Galp-1R

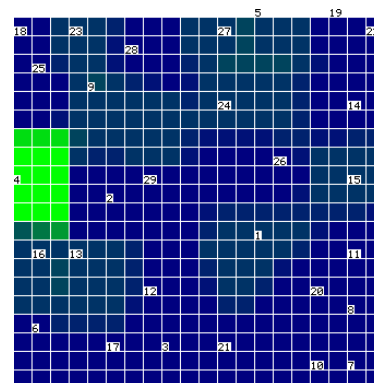


Figure 104: α-D-GalpA-1R

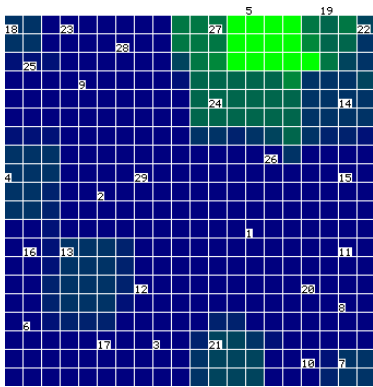


Figure 105: α -D-GalpNAc-1R

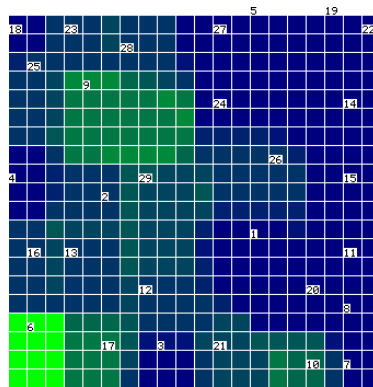


Figure 106: α -D-GalpNAc-OH-6R

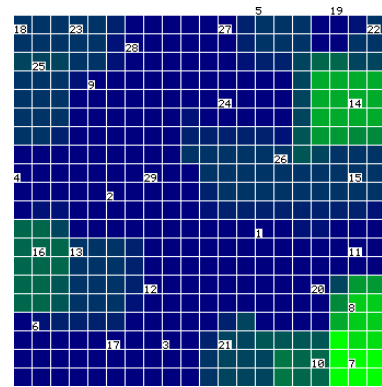


Figure 107: α -D-GalpNAc-OMe-3R

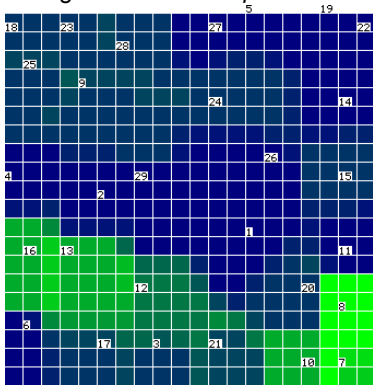


Figure 108: α -D-GalpNAc-OMe

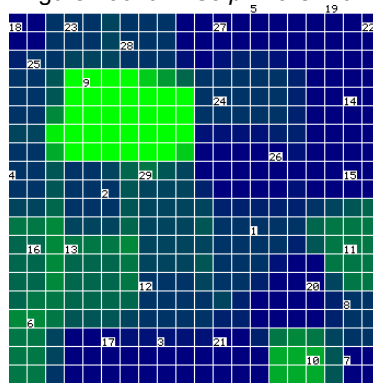


Figure 109: α -D-Galp-OH

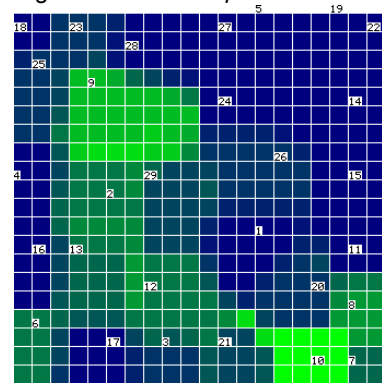


Figure 110: α -D-Galp-OH-3R

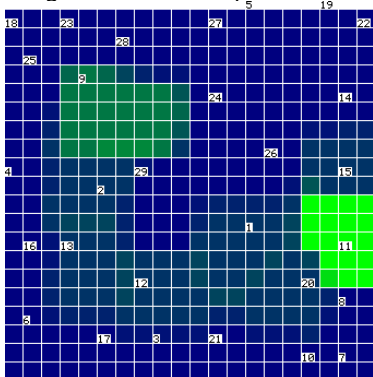


Figure 111: α -D-Galp-OH-4R

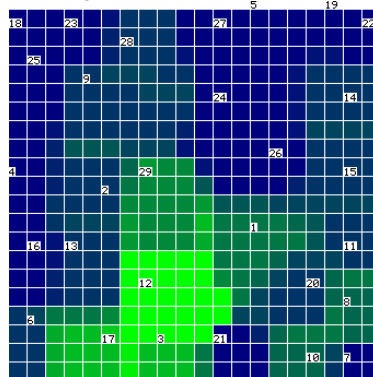


Figure 112: α -D-Galp-OH-6R

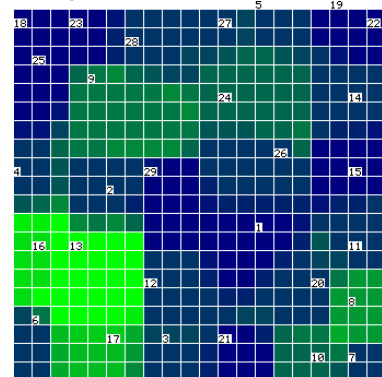


Figure 113: α -D-Galp-OMe

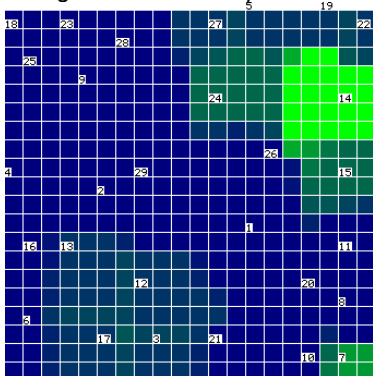


Figure 114: α -D-Galp-OMe-2R

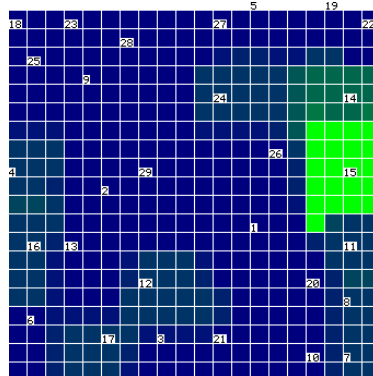


Figure 115: α -D-Galp-OMe-3R

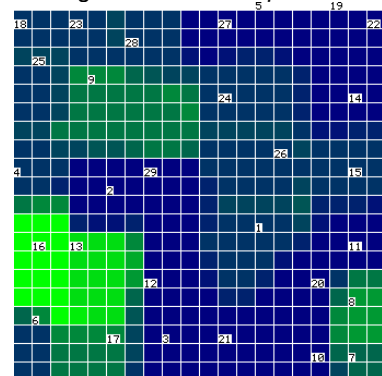
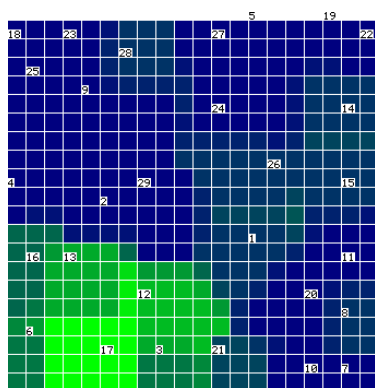
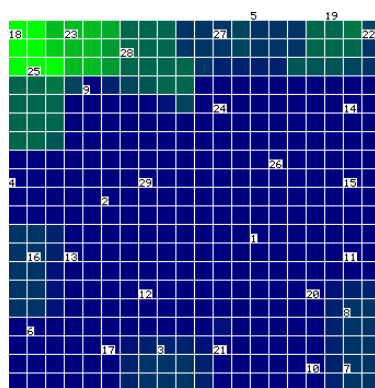
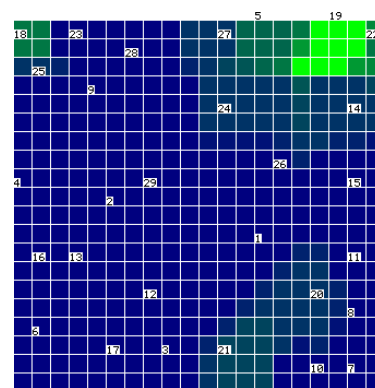
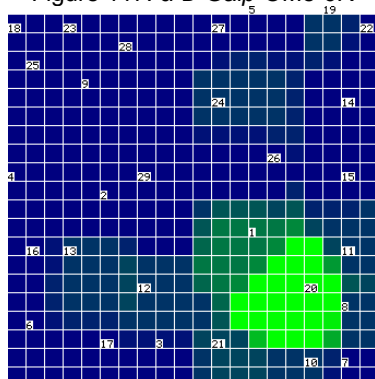
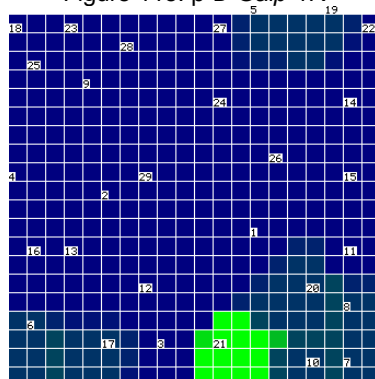
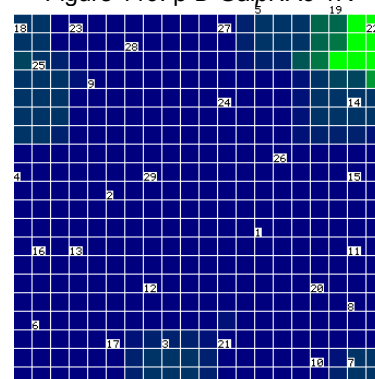
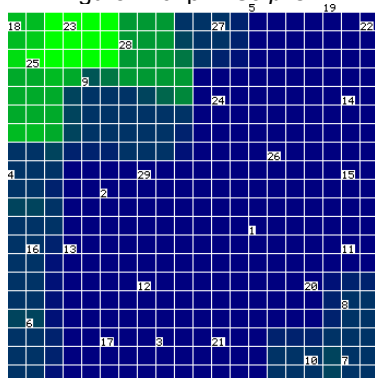
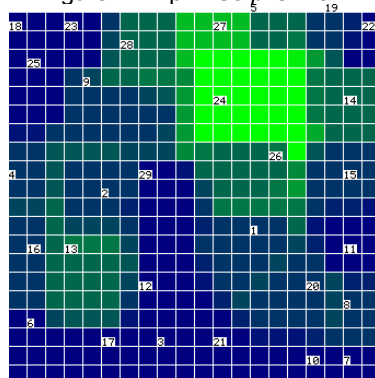
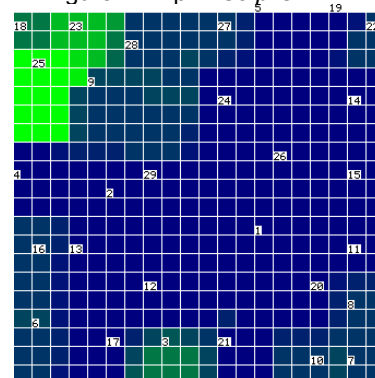
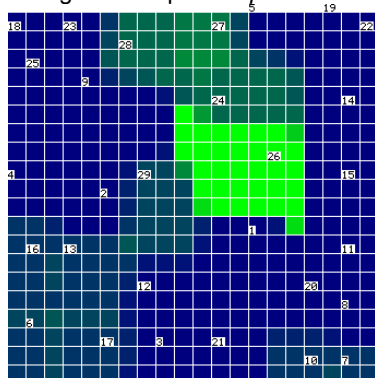
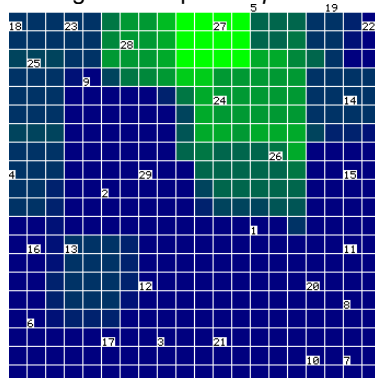
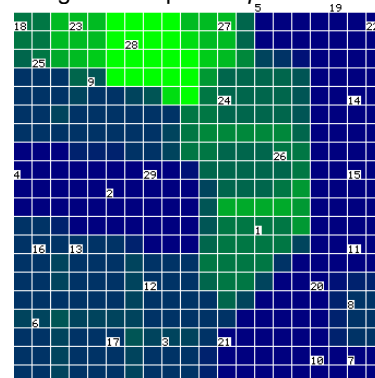


Figure 116: α -D-Galp-OMe-4R

Figure 117: α -D-Galp-OMe-6RFigure 118: β -D-Galp-1RFigure 119: β -D-GalpNAc-1RFigure 120: β -D-Galp-OHFigure 121: β -D-Galp-OH-3RFigure 122: β -D-Galp-OH-4RFigure 123: β -D-Galp-OH-6RFigure 124: β -D-Galp-OMeFigure 125: β -D-Galp-OMe-2RFigure 126: β -D-Galp-OMe-3RFigure 127: β -D-Galp-OMe-4RFigure 128: β -D-Galp-OMe-6R

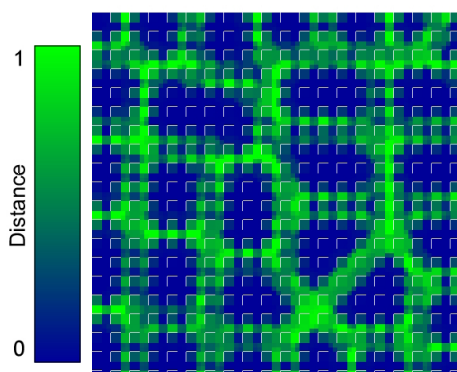


Figure 129: Euclidian distance map

5.7.3.1. Discussion & Conclusion

The following Table 19 highlights similarities between the different groups used to train the Kohonen feature map.

Table 19: Special characteristics and cohesions in the galactose Kohonen feature map

Primary activated group	Secondary activation	Description
α -D-Galp-1R	α -D-Galp-OH-6R α -D-Galp-OMe-6R	All adjacent and forming a tight cluster.
α -D-GalpNAc-OH-6R	α -D-Galp-OH	Both are similar but spatially separated.
α -D-GalpNAc-OMe-3R	α -D-GalpNAc-OMe α -D-Galp-OMe-2R	The first two groups are forming a cluster. α -D-Galp-OMe-2R is spatially separated.
α -D-GalpNAc-OMe	α -D-GalpNAc-OMe-3R α -D-Galp-OMe α -D-Galp-OMe-4R	
α -D-Galp-OH	α -D-Galp-OH-3R α -D-Galp-OH-4R α -D-GalpNAc-OH-6R	They all form spatially separated patches
α -D-Galp-OH-6R	α -D-Galp-OMe-6R	
α -D-Galp-OMe	α -D-Galp-OMe-4R α -D-Galp-OMe-6R α -D-GalpNAc-OMe (weak)	α -D-Galp-OMe, α -D-Galp-OMe-4R and α -D-Galp-OMe-6R are adjacent and are forming a compact cluster.
α -D-Galp-OMe-2R	α -D-Galp-OMe-3R	Adjacent but not interacting with each other
β -D-Galp-OMe	β -D-Galp-OMe-4R β -D-Galp-OMe-6R	All adjacent and forming a tight cluster.
β -D-Galp-OMe-2R	β -D-Galp-OMe-3R	Spatially not related

Compounds substituted at the C₂ position also provoke a simultaneous activation in the activation area of the same compound substituted at the C₃ position. Compounds substituted at the C₁ position also provoke activation in the activation area of the same compound substituted at the C₆ position and sometimes in the area of compounds substituted at the C₄ position. Therefore, it is maybe advisable to use different neural networks to distinguish between these compounds and evade possible separation problems with Back-propagation algorithm.

A graphically α / β differentiation is not visible but possible, because there are no overlapping and interacting activation patches containing α or β compounds. This proves as commonly known, that the different substitution patterns have no effect on the anomeric configuration. In other words, the anomeric configuration can not be used to determine the substitution at the other carbon atoms of a carbohydrate moiety.

5.7.4. Glucose

ANN PFG parameters	NMR type Step size Zero threshold	¹³ C – 32k data points 15 0
Software	SNNS V4.2	
Network size	261 input neurons 20 x 20 Kohonen neurons 28 Output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100'000	
Activation function	Logistic	
Output function	Identity	
Init function	Random (± 1)	
Kohonen adaptation height	0.3	
Kohonen adaptation radius	0.5	
Height decrease factor	0.99999	
Radius decrease factor	0.99999	
Horizontal size	20	
Threshold	0	
Pattern shuffling	activated	

Table 20: Glucose group allocation

1	a-D-Glcp-1R	15	b-D-GlcpNAc-1R
2	a-D-GlcpN-1R	16	b-D-GlcpNAc-OH-4R
3	a-D-Glcp-OH	17	b-D-GlcpNAc-OMe-3R
4	a-D-Glcp-OH-2R	18	b-D-GlcpNAc-OMe-4R
5	a-D-Glcp-OH-3R	19	b-D-Glcp-OH
6	a-D-Glcp-OH-4R	20	b-D-Glcp-OH-2R
7	a-D-Glcp-OH-6R	21	b-D-Glcp-OH-3R
8	a-D-Glcp-OMe	22	b-D-Glcp-OH-4R
9	a-D-Glcp-OMe-2R	23	b-D-Glcp-OH-6R
10	a-D-Glcp-OMe-3R	24	b-D-Glcp-OMe
11	a-D-Glcp-OMe-4R	25	b-D-Glcp-OMe-2R
12	a-D-Glcp-OMe-6R	26	b-D-Glcp-OMe-3R
13	b-D-Glcp-1R	27	b-D-Glcp-OMe-4R
14	b-D-GlcpN-1R	28	b-D-Glcp-OMe-6R

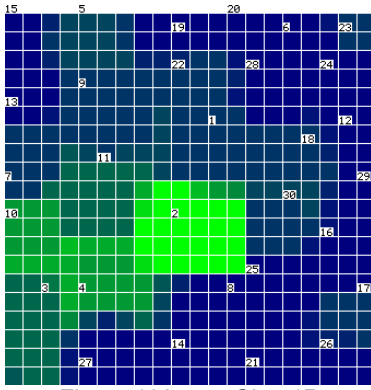


Figure 130: α -D-Glcp-1R

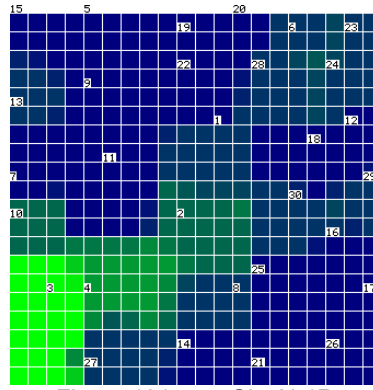


Figure 131: α -D-GlcpN-1R

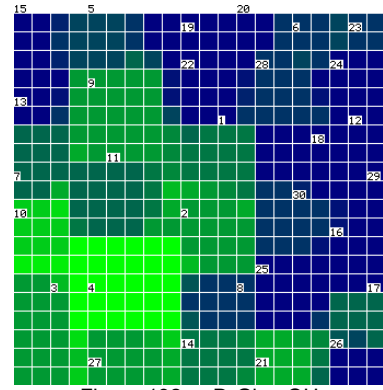


Figure 132: α -D-Glcp-OH

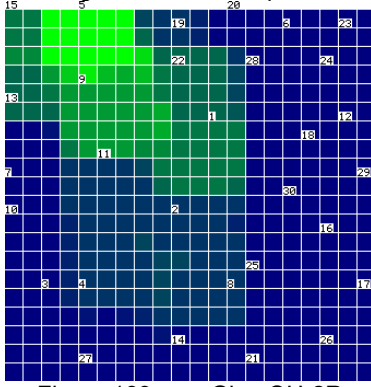


Figure 133: α -D-Glcp-OH-2R

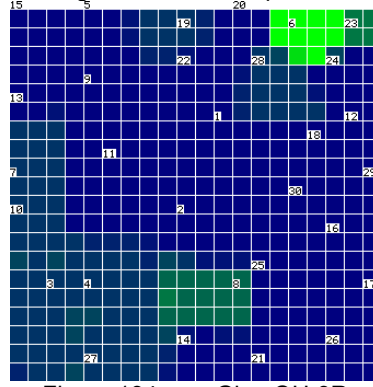


Figure 134: α -D-Glcp-OH-3R

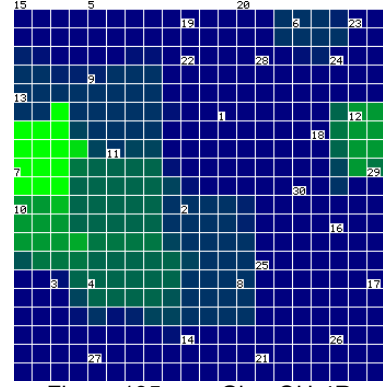


Figure 135: α -D-Glcp-OH-4R

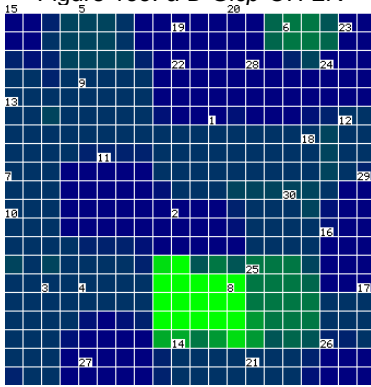


Figure 136: α -D-Glcp-OH-6R

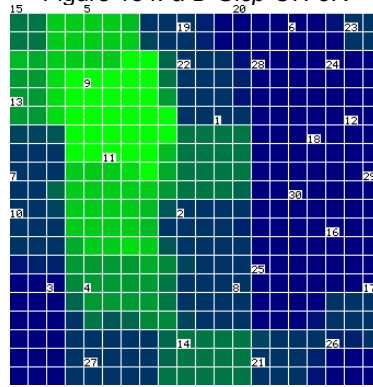


Figure 137: α -D-Glcp-OMe

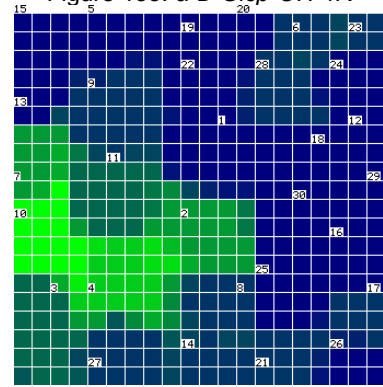


Figure 138: α -D-Glcp-OMe-2R

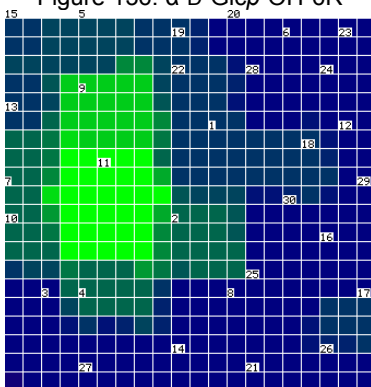


Figure 139: α -D-Glcp-OMe-3R

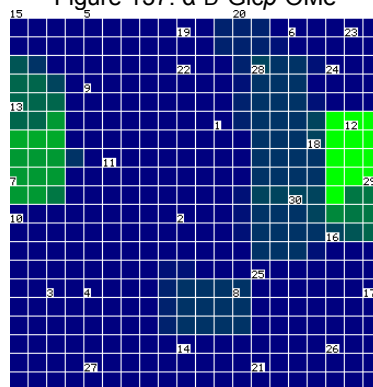


Figure 140: α -D-Glcp-OMe-4R

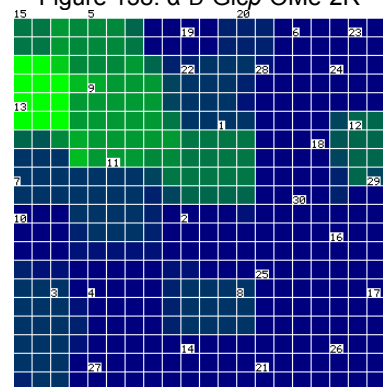
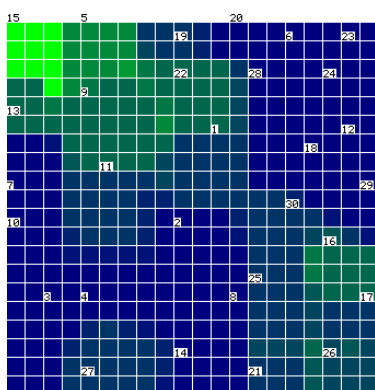
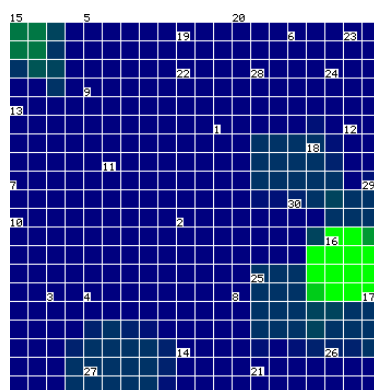
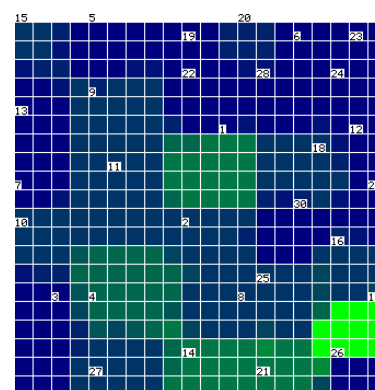
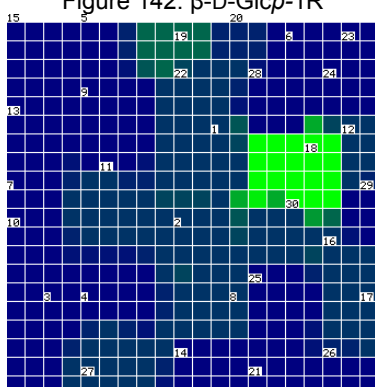
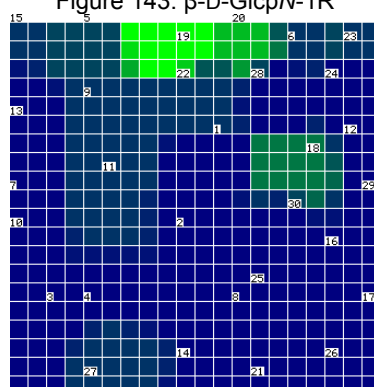
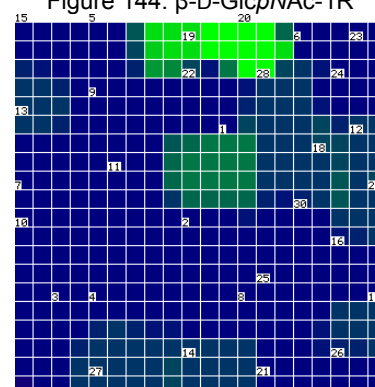
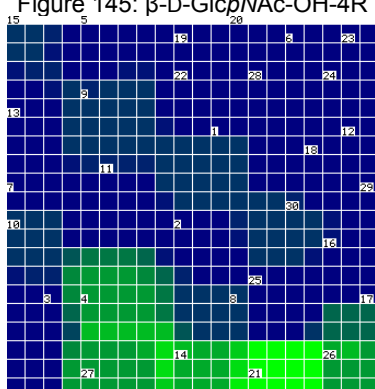
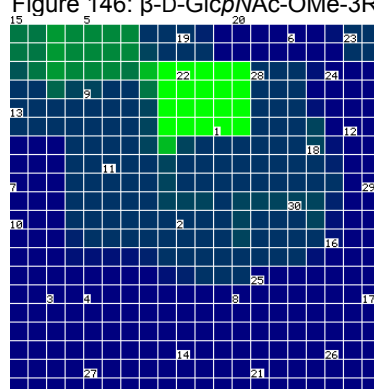
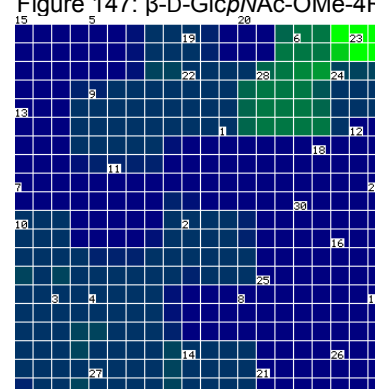
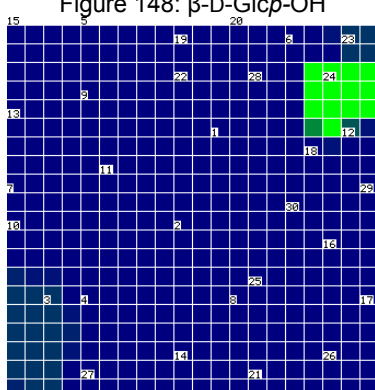
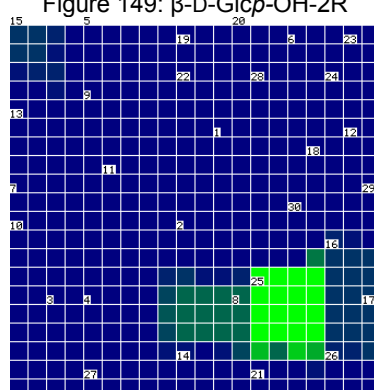
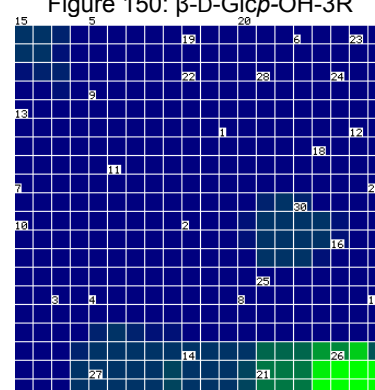


Figure 141: α -D-Glcp-OMe-6R

Figure 142: β -D-Glcp-1RFigure 143: β -D-GlcpN-1RFigure 144: β -D-GlcpNAc-1RFigure 145: β -D-GlcpNAc-OH-4RFigure 146: β -D-GlcpNAc-OMe-3RFigure 147: β -D-GlcpNAc-OMe-4RFigure 148: β -D-Glcp-OHFigure 149: β -D-Glcp-OH-2RFigure 150: β -D-Glcp-OH-3RFigure 151: β -D-Glcp-OH-4RFigure 152: β -D-Glcp-OH-6RFigure 153: β -D-Glcp-OMe

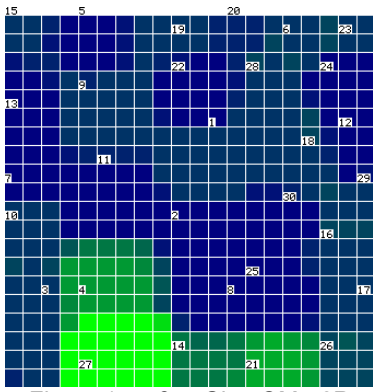


Figure 154: β -D-Glcp-OMe-2R

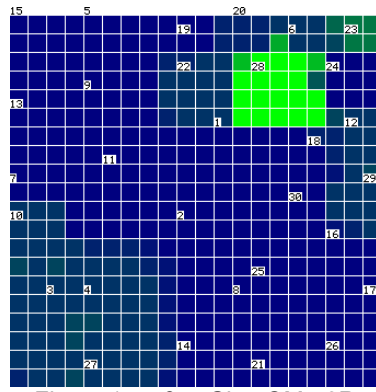


Figure 155: β -D-Glcp-OMe-3R

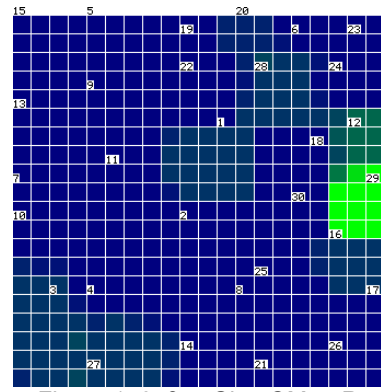


Figure 156: β -D-Glcp-OMe-4R

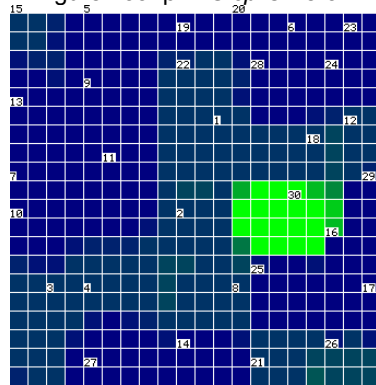


Figure 157: β -D-Glcp-OMe-6R

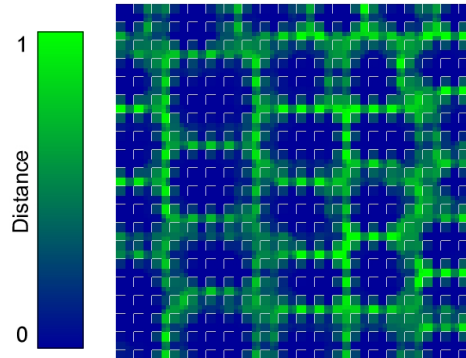


Figure 158: Euclidian distance map

5.7.4.1. Discussion & Conclusion

The following Table 19 highlights similarities between the different glucose groups used to train the Kohonen feature map.

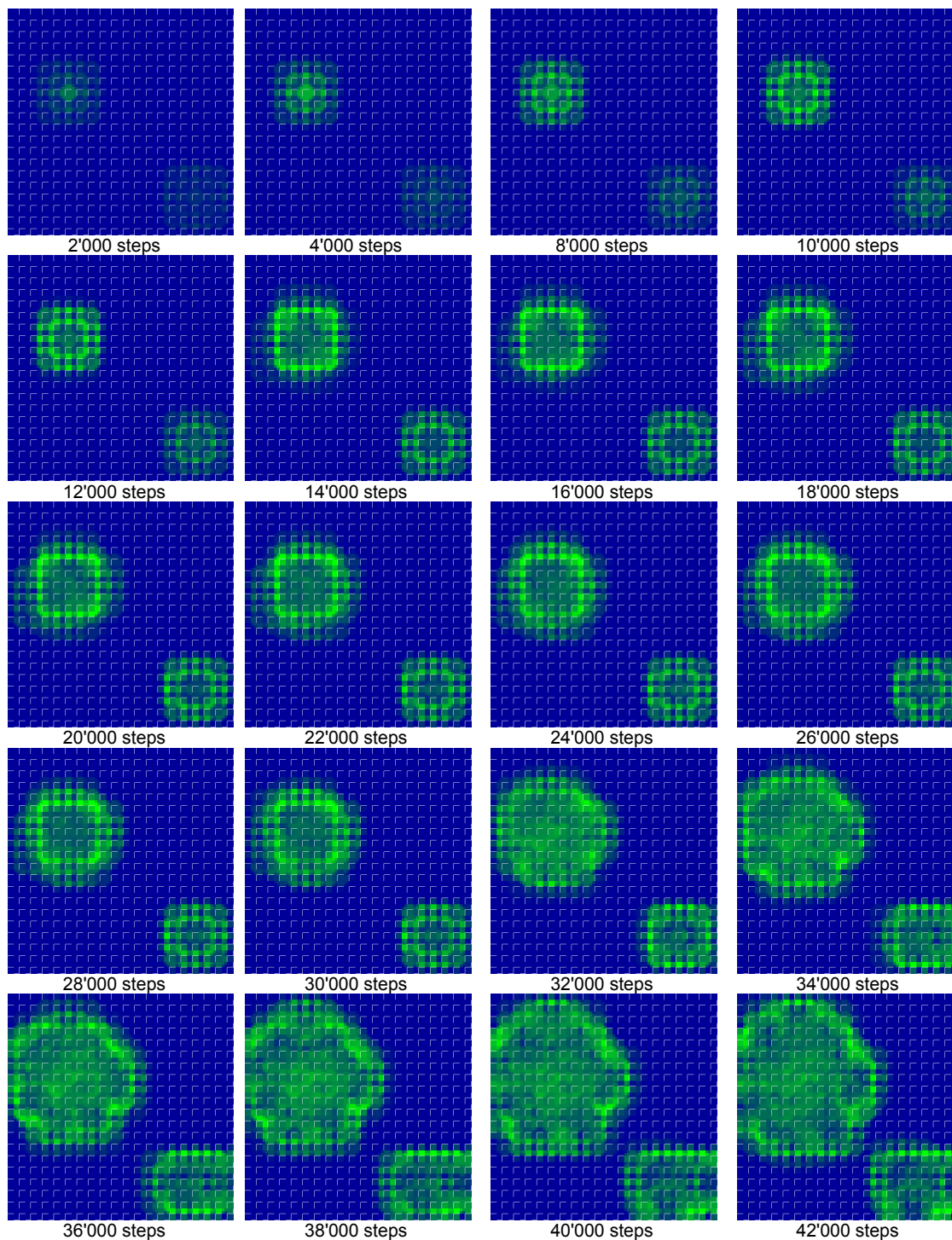
Table 21: Special characteristics and cohesions of the glucose Kohonen feature map

Primary activated group	Secondary activation	Description
α -D-Glcp-1R	α -D-Glcp-OH	Slightly blurred patches but still spatially separated.
α -D-Glcp-OH	α -D-Glcp-OMe α -D-Glcp-OMe-2R β -D-Glcp-OH β -D-Glcp-OMe-2R	They all form blurred and adjacent patches
α -D-Glcp-OH-3R	β -D- <i>p</i> -Glc-OH-3R	Side by side but clearly separated
α -D-Glcp-OMe	α -D-Glcp-OH α -D-Glcp-OH-2R α -D-Glcp-OMe-3R	They are all forming a big but clearly separated cluster
α -D-Glcp-OMe-4R	α -D-Glcp-OH-4R	Spatially totally separated patches
α -D-Glc-OMe-6R	α -D-Glcp-OMe	Adjacent and clearly separated clusters
β -D-GlcpNAc-OMe-3R	β -D-GlcpNAc-OMe-4R	Clearly separated patches lying side by side
β -D-Glcp-OH	α -D-Glcp-OH β -D-Glcp-OMe β -D-Glcp-OMe-2R	Forming one big slightly blurred cluster

The glucose Kohonen feature map does not show similarities between compounds substituted at the C₂ and the C₃, C₁ and the C₆ position. Instead, the feature map shows similarities between C₁ and C₂ substituted compounds apart from the attached group. An α / β differentiation is also not visible.

5.7.4.2. Deconvolution of the glucose Kohonen feature map

For illustration purpose the deconvolution steps (2'000 to 90'000) of the glucose Kohonen feature map are depicted in Figure 159



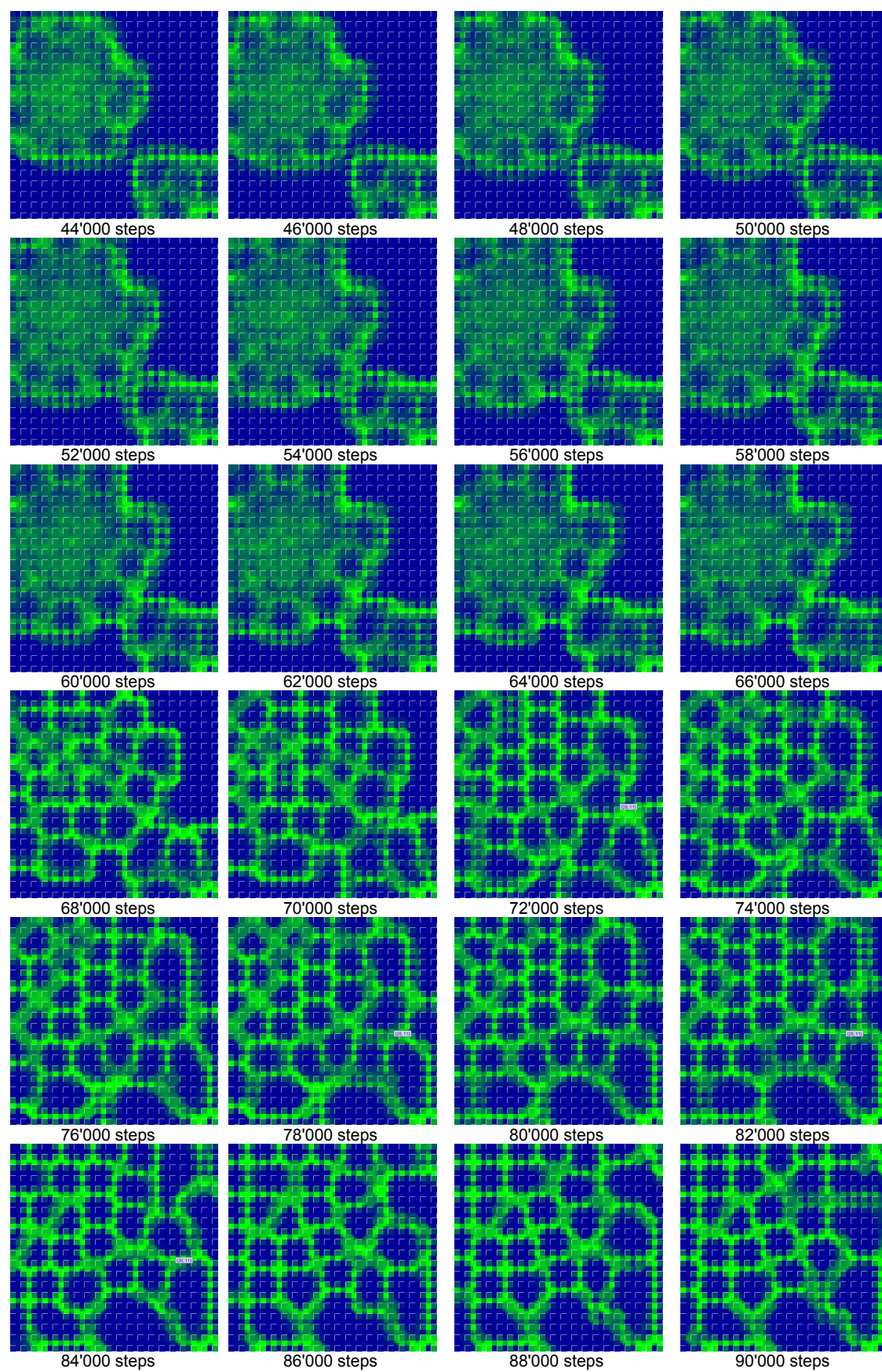


Figure 159: Deconvolution of the glucose Kohonen feature map

5.7.5. Mannose

ANN PFG parameters	NMR type Step size Zero threshold	¹³ C – 32k data points 15 0
Software	SNNS V4.2	
Network size	261 input neurons 15 x 15 Kohonen neurons 20 Output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	100'000	
Activation function	Logistic	
Output function	Identity	
Init function	Random (±1)	
Kohonen adaptation height	0.3	
Kohonen adaptation radius	0.5	
Height decrease factor	0.99999	
Radius decrease factor	0.99999	
Horizontal size	15	
Threshold	0	
Pattern shuffling	activated	

Table 22: Mannose group allocation

1	a-D-Manp-1R	12	a-D-Manp-OMe-6R
2	a-D-ManpNAc-1R	13	b-D-Manp-1R
3	a-D-Manp-OH	14	b-D-ManpNAc-1R
4	a-D-Manp-OH-2R	15	b-D-Manp-OH
5	a-D-Manp-OH-4R	16	b-D-Manp-OH-2R
6	a-D-Manp-OH-6R	17	b-D-Manp-OH-4R
7	a-D-Manp-OMe	18	b-D-Manp-OH-6R
8	a-D-Manp-OMe-2R	19	b-D-Manp-OMe
9	a-D-Manp-OMe-3R	20	b-D-Manp-OMe-2R
10	a-D-Manp-OMe-4R	21	b-D-Manp-OMe-4R
11	a-D-Manp-OMe-6R		

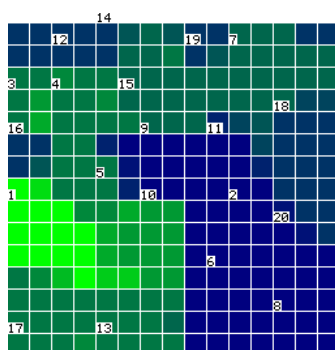
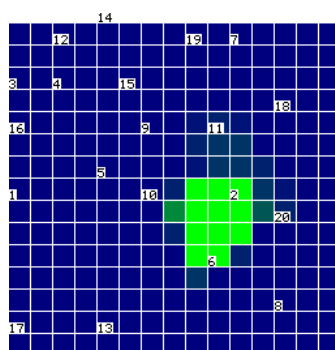
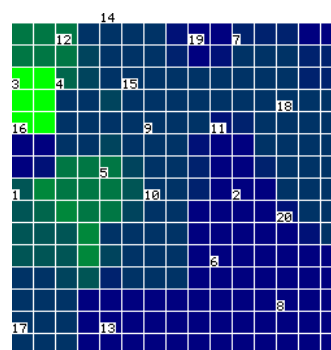
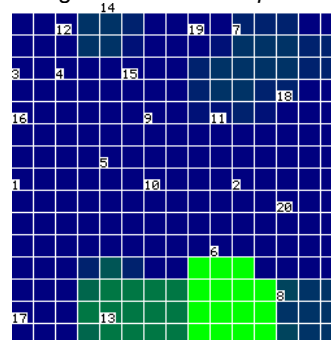
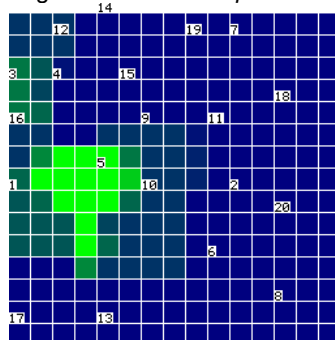
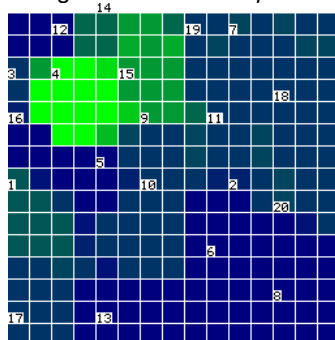
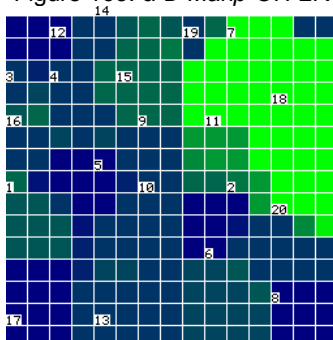
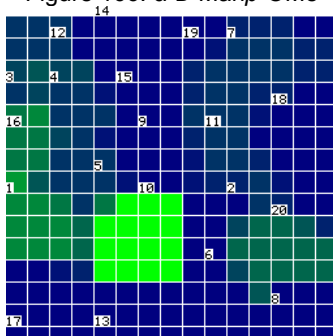
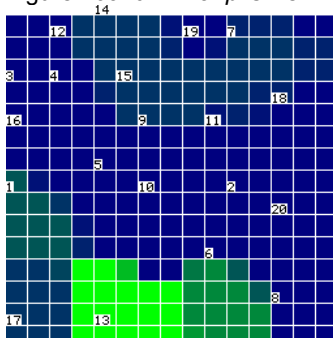
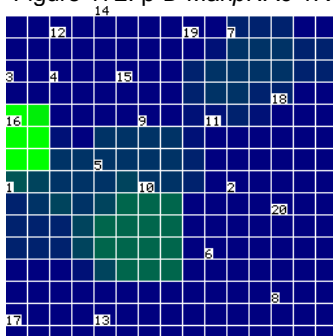
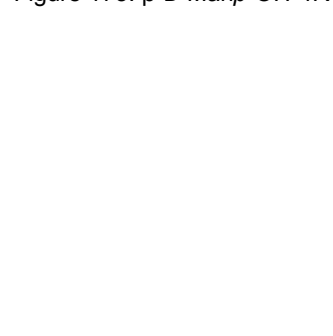
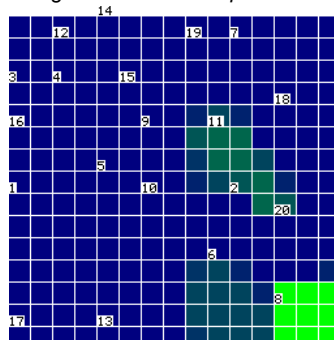
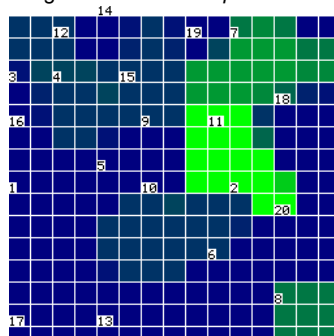
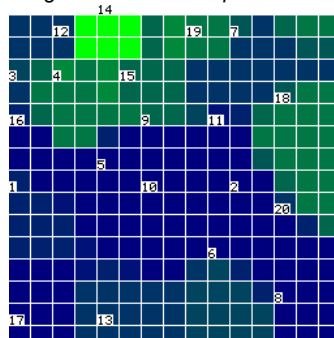
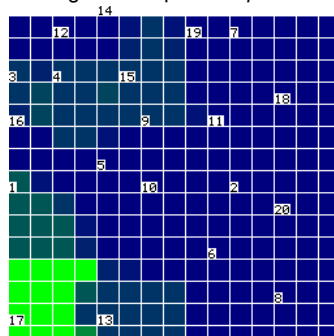
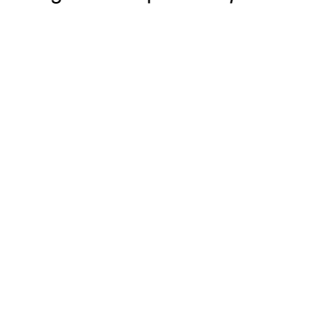
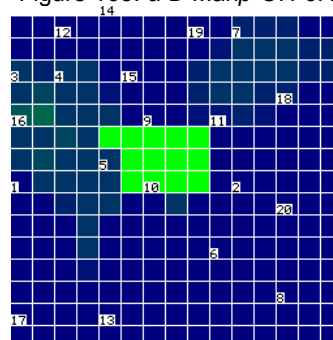
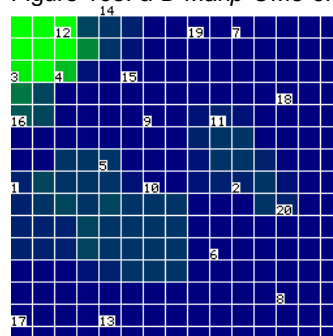
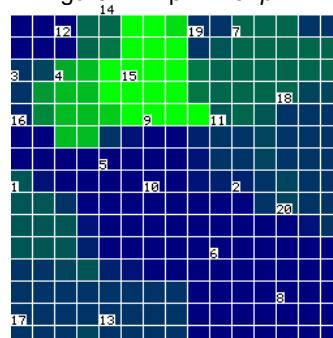
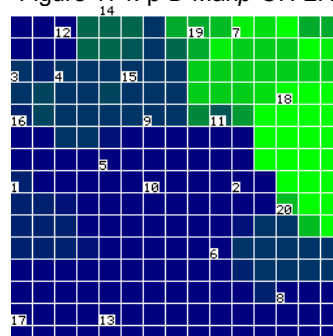
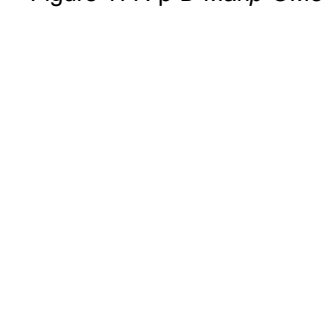
Figure 160: α -D-Manp-1RFigure 161: α -D-ManpNAc-1RFigure 162: α -D-Manp-OH

Figure 163: α -D-Manp-OH-2RFigure 166: α -D-Manp-OMEFigure 169: α -D-Manp-OME-4RFigure 172: β -D-ManpNAc-1RFigure 175: β -D-Manp-OH-4RFigure 164: α -D-Manp-OH-4RFigure 167: α -D-Manp-OME-2RFigure 170: α -D-Manp-OME-6RFigure 173: β -D-Manp-OHFigure 176: β -D-Manp-OH-6RFigure 165: α -D-Manp-OH-6RFigure 168: α -D-Manp-OME-3RFigure 171: β -D-Manp-1RFigure 174: β -D-Manp-OH-2RFigure 177: β -D-Manp-OME

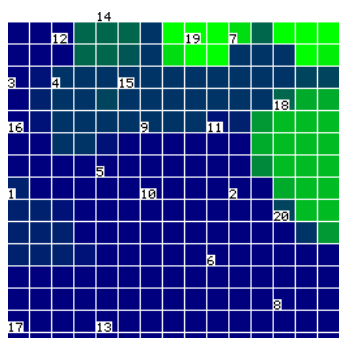


Figure 178: β -D-Manp-OMe-2R

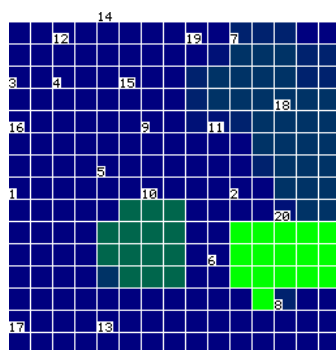


Figure 179: β -D-Manp-OMe-4R

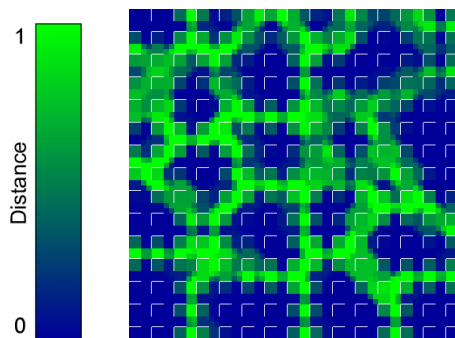


Figure 180: Euclidian distance map

5.7.5.1. Discussion & Conclusion

Table 23: Special characteristics and cohesions of the mannose Kohonen feature map

Primary activated group	Secondary activation	Description
α -D-Manp-1R	α -D-Manp-OH α -D-Manp-OH-2R α -D-Manp-OH-4R α -D-Manp-OMe α -D-Manp-OMe-4R β -D-ManpNAc-1R β -D-Manp-OH-2R β -D-Manp-OH-6R	Large slightly blurred and connected patch
α -D-Manp-OH	α -D-Manp-OH-4R	Spatially separated patches
α -D-Manp-OH-2R	β -D-Manp-OH-2R	Adjacent patches
α -D-Manp-OMe	β -D-Manp-OMe	Adjacent patches with almost indistinguishable activation.
α -D-Manp-OMe-6R	α -D-Manp-OMe α -D-Manp-OMe-2R	Adjacent activation areas.
β -D-Manp-OH	β -D-Manp-OMe	Spatially separated patches lying on opposing sites of the network.
β -D-Manp-OMe-2R	β -D-Manp-OMe	Adjacent activation areas

The classification of the mannose monosaccharide moieties seem to be the most complicated task for a Kohonen feature map. The fact that presenting an α -D-Manp-1R to the trained Kohonen feature map also activates eight other mannose moieties suggests that the network is not trained sufficiently. However, another 50'000 training cycles did not change the separation abilities of the Kohonen feature map. α -D-Manp-1R seems to be a problematic monosaccharide moiety.

The α -form of a monosaccharide moiety mostly activates also the β -form (e.g. α -D-Manp-OMe also activates β -D-Manp-OMe and vice versa). A test compound, which is substituted at C₁ also activates other areas of compounds with similar substitution at the same carbon atom (e.g. β -D-Manp-OMe also activates β -D-Manp-OH).

However, this cognition could not be applied to all used mannose test cases. The "rules" are not as clear as with the other Kohonen feature maps of glucose and galactose). If it would also be the case for future test compounds like fucose, xylose etc. cannot be estimated based on the presented results of glucose, galactose and mannose.

5.7.6. Combination of galactose, glucose and mannose

In a final test, all previous pattern files (galactose, mannose and glucose) were combined into one single pattern file and a 25x25 Kohonen feature map was trained with it. The learning parameters were slightly adapted to match the new network and pattern file size.

ANN PFG parameters	NMR type Step size Zero threshold	¹³ C – 32k data points 15 0
Software	SNNS V4.2	
Network size	261 input neurons 25 x 25 Kohonen neurons 69 Output neurons	
Output coding	binary (1= activated / 0 = deactivated)	
Training cycles	400'000	
Activation function	Logistic	
Output function	Identity	
Init function	Random (±1)	
Kohonen adaptation height	0.5	
Kohonen adaptation radius	0.5	
Height decrease factor	0.999995	
Radius decrease factor	0.999995	
Horizontal size	25	
Threshold	0	
Pattern shuffling	activated	

Table 24: GAM Group allocation

1	a-D-Manp-1R	36	b-D-Galp-OMe
2	a-D-ManpNAc-1R	37	b-D-Galp-OMe-2R
3	a-D-Manp-OH	38	b-D-Galp-OMe-3R
4	a-D-Manp-OH-2R	39	b-D-Galp-OMe-6R
5	a-D-Manp-OH-4R	40	a-D-Glcp
6	a-D-Manp-OH-6R	41	a-D-Glcp-1R
7	a-D-Manp-OMe	42	a-D-GlcpN-1R
8	a-D-Manp-OMe-2R	43	a-D-Glcp-OH
9	a-D-Manp-OMe-3R	44	a-D-Glcp-OH-2R
10	a-D-Manp-OMe-4R	45	a-D-Glcp-OH-3R
11	a-D-Manp-OMe-6R	46	a-D-Glcp-OH-4R
12	b-D-Manp-1R	47	a-D-Glcp-OH-6R
13	b-D-ManpNAc-1R	48	a-D-Glcp-OMe
14	b-D-Manp-OH	49	a-D-Glcp-OMe-2R
15	b-D-Manp-OH-2R	50	a-D-Glcp-OMe-3R
16	b-D-Manp-OH-4R	51	a-D-Glcp-OMe-4R
17	b-D-Manp-OH-6R	52	a-D-Glcp-OMe-6R
18	b-D-Manp-OMe	53	b-D-Glcp
19	b-D-Manp-OMe-2R	54	b-D-Glcp-1R
20	b-D-Manp-OMe-4R	55	b-D-GlcpN-1R
21	a-D-Galp-1R	56	b-D-GlcpNAc-1R
22	a-D-Galp-OH	57	b-D-GlcpNAc-OH-4R
23	a-D-Galp-OH-3R	58	b-D-GlcpNAc-OMe-3R
24	a-D-Galp-OH-4R	59	b-D-GlcpNAc-OMe-4R
25	a-D-Galp-OH-6R	60	b-D-Glcp-OH
26	a-D-Galp-OMe	61	b-D-Glcp-OH-2R
27	a-D-Galp-OMe-2R	62	b-D-Glcp-OH-3R
28	a-D-Galp-OMe-3R	63	b-D-Glcp-OH-4R

29	a-D-Galp-OMe-4R	64	b-D-Glcp-OH-6R
30	a-D-Galp-OMe-6R	65	b-D-Glcp-OMe
31	b-D-Galp-1R	66	b-D-Glcp-OMe-2R
32	b-D-Galp-OH	67	b-D-Glcp-OMe-3R
33	b-D-Galp-OH-3R	68	b-D-Glcp-OMe-4R
34	b-D-Galp-OH-4R	69	b-D-Glcp-OMe-6R
35	b-D-Galp-OH-6R		

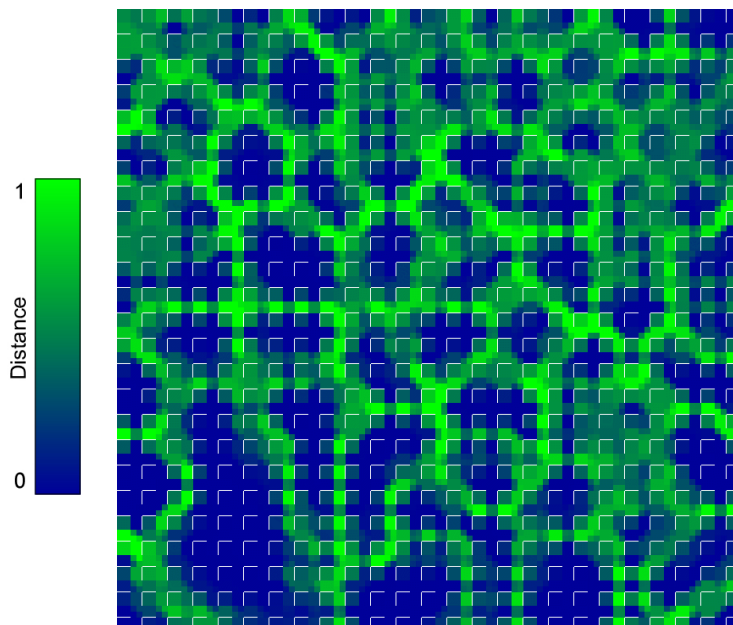


Figure 181: Euclidian distance map of the GAM Kohonen network (25 x 25 neurons) after 400'000 training cycles

5.7.7. Discussion

After 400'000 training cycles the Kohonen feature map was still unable to distinguish clearly between all 69 patterns contained in the used training pattern file (blurred regions in Figure 181). Too many groups are overlapping and therefore the approach to classify all monosaccharide units with only one neural network was abandoned, as expected before. Whereas the approach with separated networks for each carbohydrate species (glucose, galactose and mannose) was perpetuated in the following experiments. Each Kohonen network was afterwards tested with the test pattern files of the two carbohydrate species not involved in the training process (e.g. the glucose Kohonen map was tested with the galactose and mannose test pattern file) and the three Kohonen feature maps were unable to predict monosaccharides they were not trained with. Therefore, it can be concluded, that the different sugars included in the FileMaker ^{13}C -NMR database can be clearly differentiated with separated neural networks specialized for only one carbohydrate group.

However, the experiments showed again, that the full dataset of all monosaccharide units contained in the in the FileMaker ^{13}C -NMR database can be classified by means of their full ^{13}C -NMR spectrum. The used group allocation is correct and will be applied to all further experiments.

5.8. Statistica Approach

The data flow from the FileMaker ^{13}C -NMR database to Statsoft Statistica is structured in three different major steps (depicted in Figure 182).

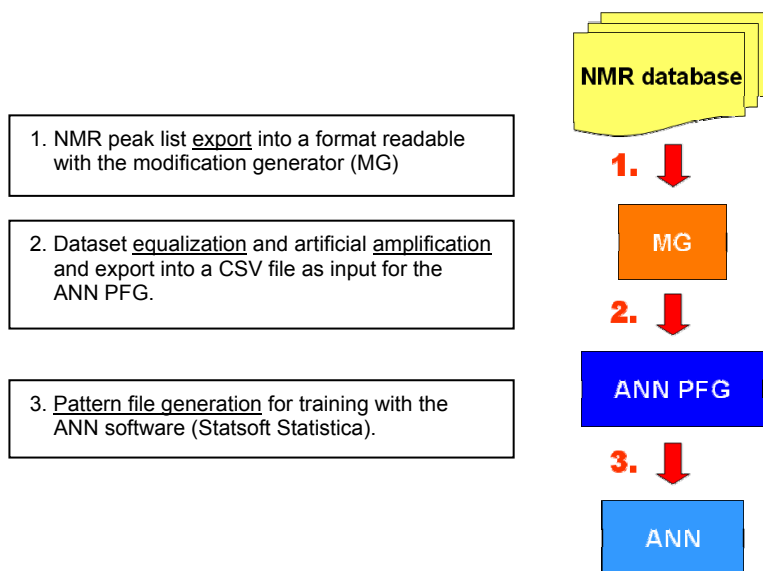


Figure 182: Coarse workflow of the Statistica approach

- In the first step, the peak lists of the desired Carbohydrates are exported into a CSV file and then manually divided into their individual monosaccharide moieties.
- In the second step, these monosaccharide units are equalized and artificially amplified (shifted) with the MG (chapter 4.7) and saved into a CSV file formatted (for exact definitions see chapter 4.9.2.2) to serve as input for the ANN PFG.
- In the final third step, this CSV file is processed and converted into a training pattern file suitable to be used as an input for Statsoft Statistica (for exact format definitions see chapter 4.9.2.2).

5.8.1. Experiment-nomenclature

The pattern files generated with the ANN PFG and the corresponding neural networks are named according to the following scheme:

Carbohydrate_MGversion_shift_modifications_stepsize

e.g. gal_mini4_sh05_mod80_step20

<i>Carbohydrate</i>	<i>MG version</i>	<i>shift size</i>	<i>modifications</i>	<i>step size</i>
the first three letters represent the used carbohydrate group abbreviated with gal, glc, man, GAM, fuc, etc.	the four different version of the MG are labeled with mini1 – mini4	the shift size [ppm] in a two digit form whereas the first digit represents the number before and the second digit the number after the decimal point. e.g. 05 = 0.5ppm 10 = 1.0ppm stdabw = standard deviation	the number of artificial modifications made out of each monosaccharide moiety.	the reading step size used for the ANN PFG

5.8.2. Definitions

- *Performance:* For nominal variables (classification outputs), the performance measure is the proportion of cases correctly classified. This takes no account of doubt options, and so a classification network with conservative accept and reject thresholds (confidence limits) may have a low apparent performance, as many cases are not correctly classified.
- *Error:* The error of the network on the subsets used during training. This is less interpretable than the performance measure, but is the figure actually optimized by the training algorithm (at least, for the training subset). This is the RMS of the network errors on the individual cases, where the individual errors are generated by the network error function, which is either a function of the observed and expected output neuron activation levels (usually sum-squared or a cross-entropy measure - see chapter 4.6) for more details.
- *Confidence:* Confidence levels define the accept and reject thresholds for the classifications task. Correct classifications result from output neuron activations higher than the accept threshold confidence level. False classifications result from levels below the reject confidence level. In all conducted experiments of the following chapters, the confidence levels were set to 0.75 as accept and 0.25 as reject threshold.

Patterns used for training with Statsoft Statistica are normally subdivided into three different subsets (the number of compounds in each subset is indicated in the next section).

- *Train.* Observations in the Training set will be used to train the network (i.e., to estimate the network weights and other parameters).

- *Test:* Observations in the test-set will be used to perform an "independent check" of the network performance during training, to avoid over-fitting the data (i.e., to determine when to terminate training the network).

- *Selection:* Observations in the selection-set will not be used during training of the network (estimation procedure) at all, but the fully trained network will be applied to those cases as a final independent check of the final network performance (also called the generalization).

After the training step, the neural networks are classified by means of their best selection performance.

5.8.3. Pattern file structure

As Statistica is able to deal with three different subsets (*train*, *test* and *selection*), the training pattern file was subdivided as follows:

Two thirds of the modifications generated with the MG were classified as training compounds. The remaining third belongs to the test subset.



Figure 183: General pattern file structure

The selection subset consists of the averaged peak lists of all monosaccharide moieties contained in the training pattern file and 200 randomly chosen monosaccharide peak lists directly out of the ^{13}C -NMR database (only from the trained monosaccharide).

5.8.4. Data set

For all the following experiments data ^[38, 73, 76, 77, 95-291] of the FileMaker ¹³C-NMR database in its final stage of expansion was used (chapter 4.1.5). The following monosaccharide moiety classification of each of the three carbohydrates galactose, glucose and mannose were used for the training of the neural networks.

Table 25: Used galactose monosaccharide moieties

1	a-D-Galp-1R	11	b-D-Galp-1R
2	a-D-Galp-OH	12	b-D-Galp-OH
3	a-D-Galp-OH-3R	13	b-D-Galp-OH-3R
4	a-D-Galp-OH-4R	14	b-D-Galp-OH-4R
5	a-D-Galp-OH-6R	15	b-D-Galp-OH-6R
6	a-D-Galp-OMe	16	b-D-Galp-OMe
7	a-D-Galp-OMe-2R	17	b-D-Galp-OMe-2R
8	a-D-Galp-OMe-3R	18	b-D-Galp-OMe-3R
9	a-D-Galp-OMe-4R	19	b-D-Galp-OMe-6R
10	a-D-Galp-OMe-6R		

Table 26: Used glucose monosaccharide moieties

1	a-D-Glcp-1R	12	b-D-Glcp-1R
2	a-D-Glcp-OH	13	b-D-Glcp-OH
3	a-D-Glcp-OH-2R	14	b-D-Glcp-OH-2R
4	a-D-Glcp-OH-3R	15	b-D-Glcp-OH-3R
5	a-D-Glcp-OH-4R	16	b-D-Glcp-OH-4R
6	a-D-Glcp-OH-6R	17	b-D-Glcp-OH-6R
7	a-D-Glcp-OMe	18	b-D-Glcp-OMe
8	a-D-Glcp-OMe-2R	19	b-D-Glcp-OMe-2R
9	a-D-Glcp-OMe-3R	20	b-D-Glcp-OMe-3R
10	a-D-Glcp-OMe-4R	21	b-D-Glcp-OMe-4R
11	a-D-Glcp-OMe-6R	22	b-D-Glcp-OMe-6R

Table 27: Used Mannose monosaccharide moieties

1	a-D-Manp-1R	11	a-D-Manp-OMe-6R
2	a-D-Manp-OH	12	b-D-Manp-1R
3	a-D-Manp-OH-2R	13	b-D-Manp-OH
4	a-D-Manp-OH-4R	14	b-D-Manp-OH-2R
5	a-D-Manp-OH-6R	15	b-D-Manp-OH-4R
6	a-D-Manp-OMe	16	b-D-Manp-OH-6R
7	a-D-Manp-OMe-2R	17	b-D-Manp-OMe
8	a-D-Manp-OMe-3R	18	b-D-Manp-OMe-2R
9	a-D-Manp-OMe-4R	19	b-D-Manp-OMe-4R
10	a-D-Manp-OMe-6R		

For the final combination experiments with GAM, the monosaccharide moieties of glucose, galactose and mannose were combined. The resulting pattern file contained 60 groups.

5.8.5. Test files

5.8.5.1. Monosaccharide test

The monosaccharide test files were always included directly into the training pattern file and marked as *selection* subset. This subset is always applied to the fully trained network at the end of the training process. The data distribution is shown in the following tables. The detailed composition of the test files can be found in appendix 11.3.

Table 28: Data distribution of the glucose monosaccharide moiety test file

frequency	monosaccharide moiety
19.69%	a-D-Glcp-1R
2.15%	a-D-Glcp-OH
1.85%	a-D-Glcp-OH-2R
2.46%	a-D-Glcp-OH-3R
6.77%	a-D-Glcp-OH-4R
4.92%	a-D-Glcp-OH-6R
2.77%	a-D-Glcp-OMe
0.92%	a-D-Glcp-OMe-2R
0.62%	a-D-Glcp-OMe-3R
0.92%	a-D-Glcp-OMe-4R
2.15%	a-D-Glcp-OMe-6R
25.85%	b-D-Glcp-1R
2.77%	b-D-Glcp-OH
1.85%	b-D-Glcp-OH-2R
2.46%	b-D-Glcp-OH-3R
7.38%	b-D-Glcp-OH-4R
4.92%	b-D-Glcp-OH-6R
1.54%	b-D-Glcp-OMe
0.62%	b-D-Glcp-OMe-2R
1.23%	b-D-Glcp-OMe-3R
4.92%	b-D-Glcp-OMe-4R
1.23%	b-D-Glcp-OMe-6R

Table 29: Data distribution of the galactose monosaccharide moiety test file

frequency	monosaccharide moiety
14.62%	a-D-Galp-1R
1.17%	a-D-Galp-OH
4.68%	a-D-Galp-OH-3R
0.58%	a-D-Galp-OH-4R
1.75%	a-D-Galp-OH-6R
5.85%	a-D-Galp-OMe
0.58%	a-D-Galp-OMe-2R
5.26%	a-D-Galp-OMe-3R
1.75%	a-D-Galp-OMe-4R
4.09%	a-D-Galp-OMe-6R
39.18%	b-D-Galp-1R
1.17%	b-D-Galp-OH
6.43%	b-D-Galp-OH-3R
1.17%	b-D-Galp-OH-4R
1.75%	b-D-Galp-OH-6R
4.68%	b-D-Galp-OMe
1.17%	b-D-Galp-OMe-2R
0.58%	b-D-Galp-OMe-3R
3.51%	b-D-Galp-OMe-6R

Table 30: Data distribution of the mannose monosaccharide moiety test file

frequency	monosaccharide moiety
42.95%	a-D-Manp-1R
2.68%	a-D-Manp-OH
5.37%	a-D-Manp-OH-2R
3.36%	a-D-Manp-OH-4R
0.67%	a-D-Manp-OH-6R
2.68%	a-D-Manp-OMe
5.37%	a-D-Manp-OMe-2R
7.38%	a-D-Manp-OMe-3R
2.68%	a-D-Manp-OMe-4R
4.03%	a-D-Manp-OMe-6R
12.08%	b-D-Manp-1R
2.01%	b-D-Manp-OH
1.34%	b-D-Manp-OH-2R
3.36%	b-D-Manp-OH-4R
0.67%	b-D-Manp-OH-6R
0.67%	b-D-Manp-OMe
2.01%	b-D-Manp-OMe-2R
0.67%	b-D-Manp-OMe-4R

5.8.5.2. Self-measured test compounds

The four disaccharides Trehalose, Gentiobiose, Lactose and Saccharose (chapters 4.1.3 and 11.1 for more details), the compounds RS1 and RS2 from Regula Stingelin (chapter 4.1.4) and the compounds OH1, OH3, OH5, OH7, OH8 and OH9 from Ole Hindsgaul (chapter 4.1.2) were used as a combined positive and negative (the Fructofuranose unit of the Saccharose was never included in a training pattern file) test set. All compounds were kindly measured and resolved by Brian Cutting.

5.8.5.3. Disaccharide test file

For the disaccharide test file, all literature disaccharides contained in the FileMaker database were exported. Non-glucose, galactose and mannose compounds were deleted and excess disaccharides with too high frequency (like α -D-Glcp-1R) were reduced.

The test file finally contained 175 evenly distributed literature disaccharide peak lists (350 monosaccharide moieties). The data distribution is shown in Table 31. A detailed composition of the GAM test file can be found in appendix 11.4.³

³ It is not possible to exclude the possibility that there are still incorrect literature peaks in the ¹³C-NMR database. Therefore the disaccharide test performance is maybe a little bit lower than for the own measured in-house NMR compounds.

Many mistakes in published literature data have already been discovered with the help of the trained Kohonen feature maps.

Table 31: Disaccharide test set data distribution

frequency	monosaccharide moiety	frequency	monosaccharide moiety
5.71%	a-D-Galp-1R	1.43%	a-D-Galp-OMe-3R
5.71%	b-D-Galp-1R	1.14%	a-D-Galp-OMe-4R
17.71%	a-D-Glcp-1R	1.43%	a-D-Galp-OMe-6R
15.71%	b-D-Glcp-1R	0.57%	b-D-Galp-OMe-2R
6.29%	a-D-Manp-1R	0.57%	b-D-Galp-OMe-3R
1.43%	b-D-Manp-1R	0.57%	b-D-Galp-OMe-4R
0.57%	a-D-Galp-OH-3R	1.71%	b-D-Galp-OMe-6R
0.57%	a-D-Galp-OH-6R	0.86%	a-D-Glcp-OMe-2R
0.86%	b-D-Galp-OH-3R	0.57%	a-D-Glcp-OMe-3R
0.57%	b-D-Galp-OH-4R	0.57%	a-D-Glcp-OMe-4R
0.57%	b-D-Galp-OH-6R	1.43%	a-D-Glcp-OMe-6R
2.00%	a-D-Glcp-OH-2R	0.57%	b-D-Glcp-OMe-2R
2.29%	a-D-Glcp-OH-3R	0.86%	b-D-Glcp-OMe-3R
2.29%	a-D-Glcp-OH-4R	3.14%	b-D-Glcp-OMe-4R
2.57%	a-D-Glcp-OH-6R	1.43%	b-D-Glcp-OMe-6R
2.00%	b-D-Glcp-OH-2R	2.00%	a-D-Manp-OMe-2R
2.29%	b-D-Glcp-OH-3R	2.00%	a-D-Manp-OMe-3R
2.86%	b-D-Glcp-OH-4R	0.86%	a-D-Manp-OMe-4R
2.57%	b-D-Glcp-OH-6R	1.14%	a-D-Manp-OMe-6R
0.57%	a-D-Manp-OH-2R	0.86%	b-D-Manp-OMe-2R
0.57%	a-D-Manp-OH-4R		
0.57%	b-D-Manp-OH-4R		

number	fraction	carbohydrate
77	22.00%	Galactose
216	61.71%	Glucose
57	16.29%	Mannose

5.8.5.4. Negative test

The respective pattern files, the network was not trained with, were used as a negative test.

- The galactose networks were tested with the selection subset of the glucose and mannose pattern files.
- The glucose networks were tested with the selection subset of the galactose and mannose pattern files.
- The mannose networks were tested with the selection subset of the glucose and galactose pattern files

5.8.6. Preliminary experiments with Statsoft Statistica

To prepare systematic experiments, some preliminary test had to be done. Factorial design would have been advisable, but was not used because there was already good knowledge from the preceding experiments and their results.

The introduction of new features in the MG and the ANN PFG mad it inevitable to test again some good findings of the NMR and ANN problem like the amount of modification, the step size, the learning rate, shift etc.

5.8.6.1. Modification comparison

To estimate the pattern file size necessary to give good generalization results, different pattern files for galactose were generated containing 10, 20, 40 60, 80, 100, 150 and 200 modifications.

All eight networks were trained 10 times each with 2000 cycles Back-propagation (learning rate 0.1, 0.01 noise and pattern shuffling enabled) followed by 1000 cycles Conjugated-Gradient (=CG).

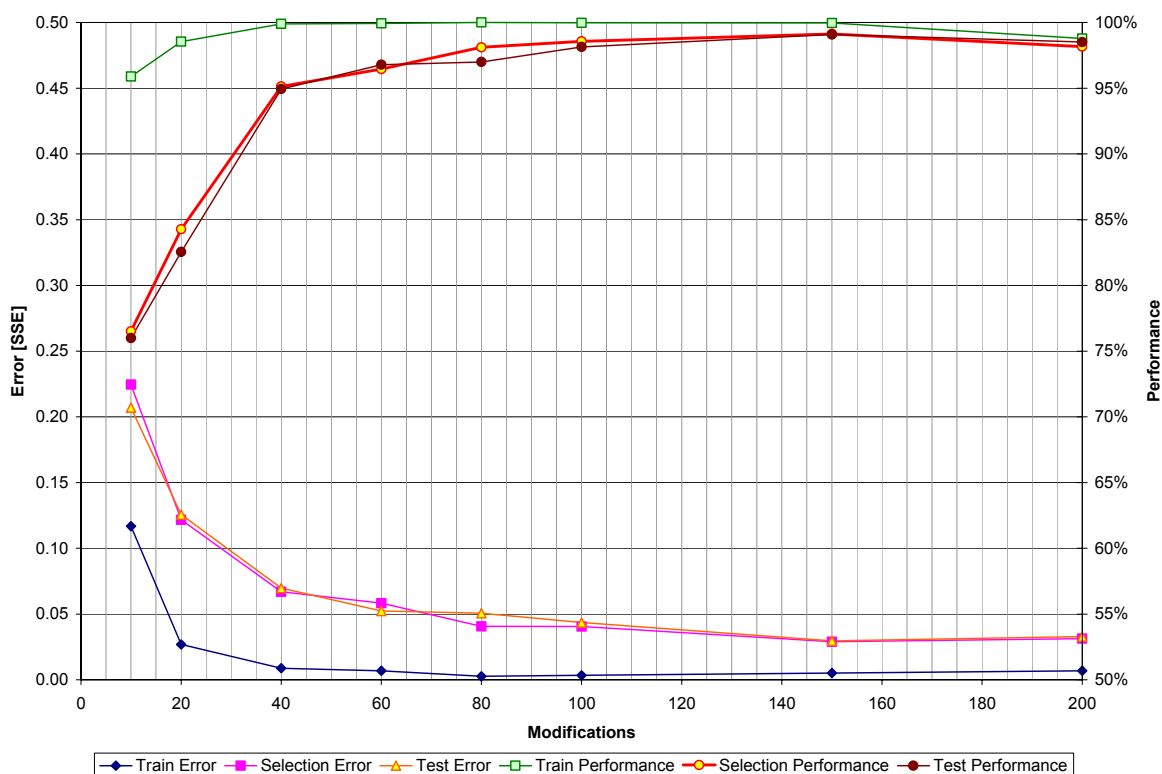


Figure 184: Average Error and performance values for different numbers of modifications (gal_mini3_sh05_modxxx)

To have an idea of the location of the best performance, the average selection performance values of all ten tested networks are drawn against the number of hidden units and are depicted in Figure 185.

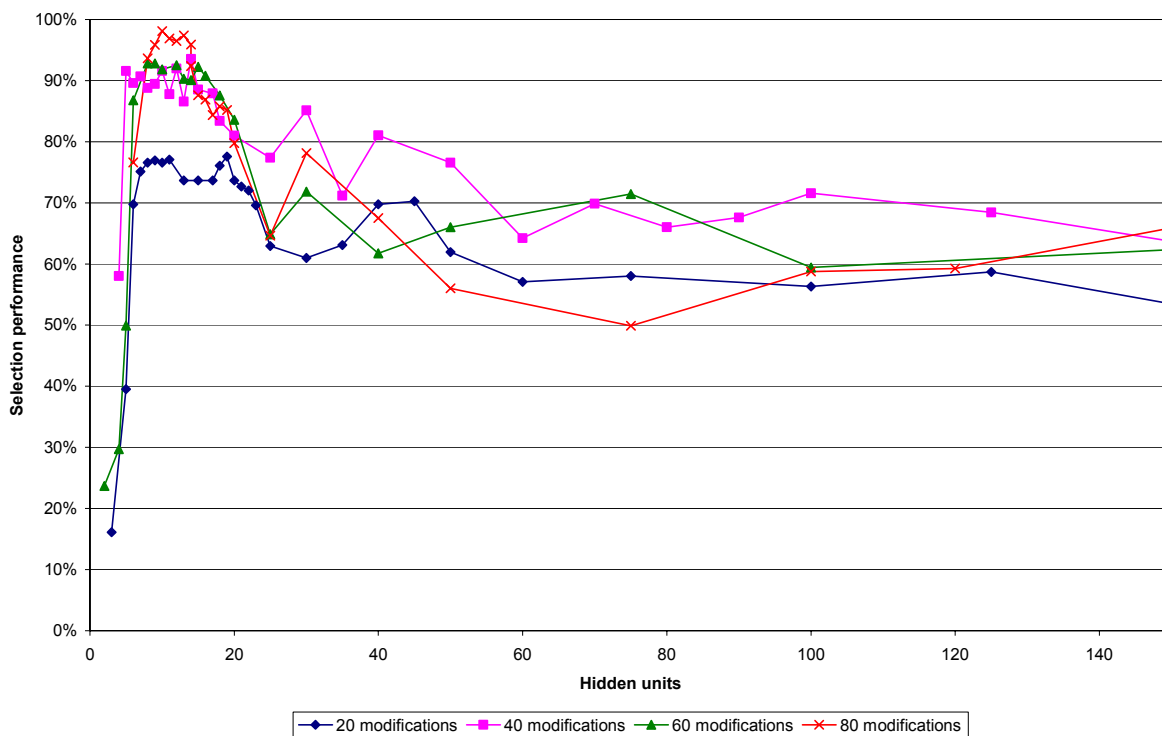


Figure 185: Average selection performance of different numbers of modifications (gal_mini3_sh05_modxxx)

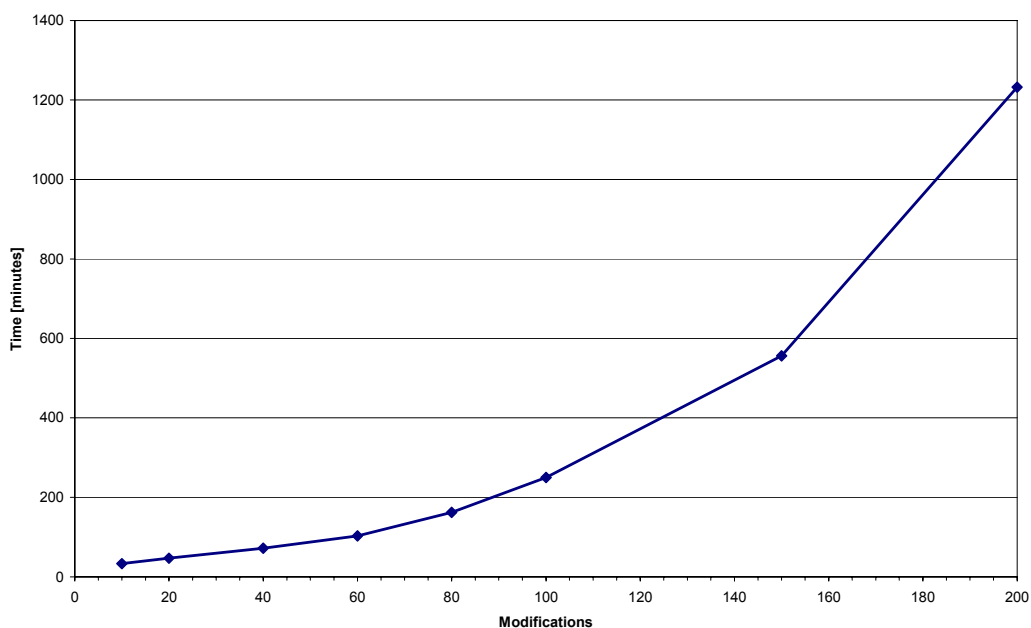


Figure 186: Average training time (1 – 200 hidden units) for a Back-propagation neural network

Eighty modifications will be taken as an optimal standard in all following experiments if not noted. More modifications lead only to a minor improvement of the selection performance. Moreover, computational times of over 20 hours cannot be realized. Not mentioning the amount of time needed to generate the pattern files at all. The processing of 200 modifications with the ANN PFG V.0.9 takes about 8 hours.

In a first approximation, an ideal network hidden layer size seems to be around 10 - 20 hidden units and pattern files with 80 modifications of each monosaccharide moiety lead to good network generalization rates (selection performance).

5.8.6.2. Learning rate comparison with 40 and 80 modifications

To find the range of the best learning rate, 19 neural networks with different learning rates were trained for 2000 cycles (1000 cycles Back-propagation, learning rate 0.1, shuffling enabled and followed by 1000 cycles CG) 10 times each. The hidden layer size was kept fixed at 10 hidden units (because of the preceding experiment).

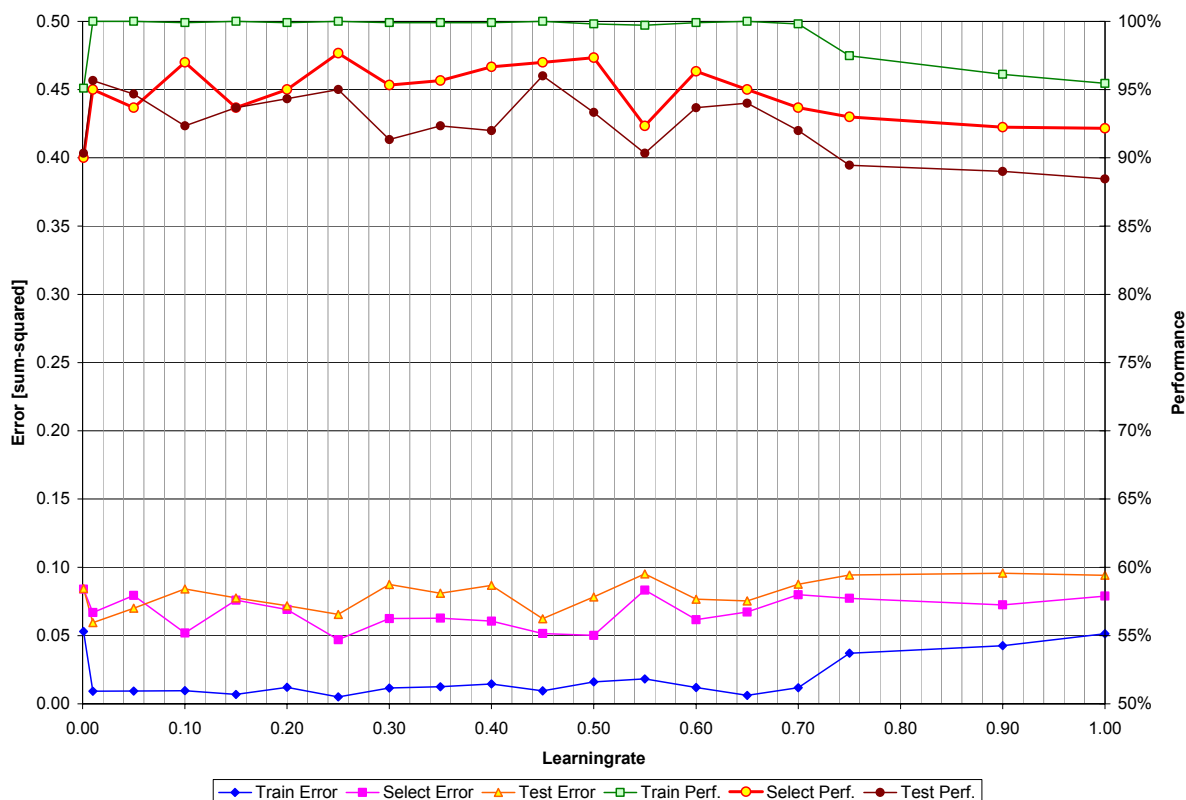


Figure 187: Lowest error and best performance values for different learning rates (gal_mini2_sh05_mod40 10hu)

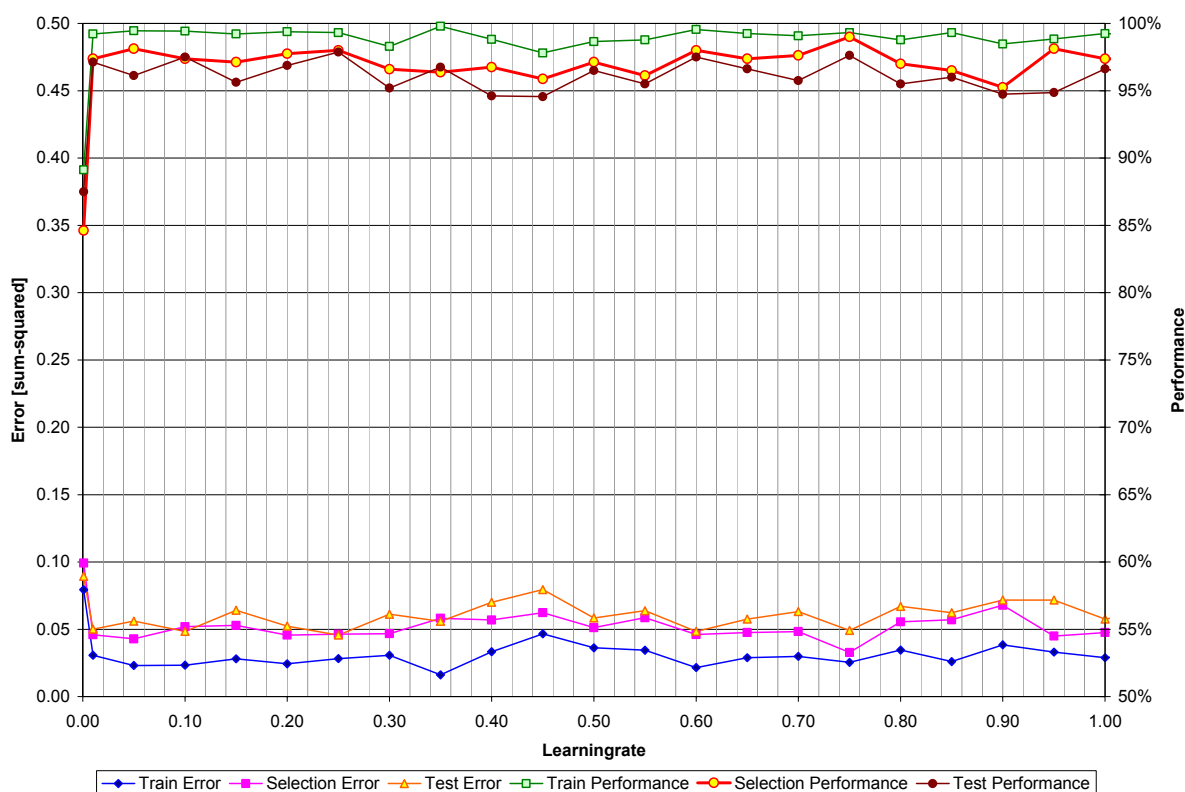


Figure 188: Learning rate overview
(gal_mini2_sh05_mod80 10hu)

As concluded in experiment 5.4.3, the results of these two experiments show again, that the classification of carbohydrate moieties is not learning rate dependent. Whereas the experiments show again, that a larger number of modifications slightly improves the selection performance. The performance and error curves move closer together. The selection performance increases only about 3%.

5.8.6.3. Momentum term comparison

Another parameter closely related to the learning rate is the momentum term. It is theoretically possible that every newly trained neural network finds another local minimum (or even the global minimum). To raise the chance of finding a better minimum, different networks with momentum terms from 0.1 – 0.9 were trained for 2000 cycles (1500 cycles Back-propagation, learning rate 0.01, shuffling enabled and followed by 500 cycles CG) 10 times each.

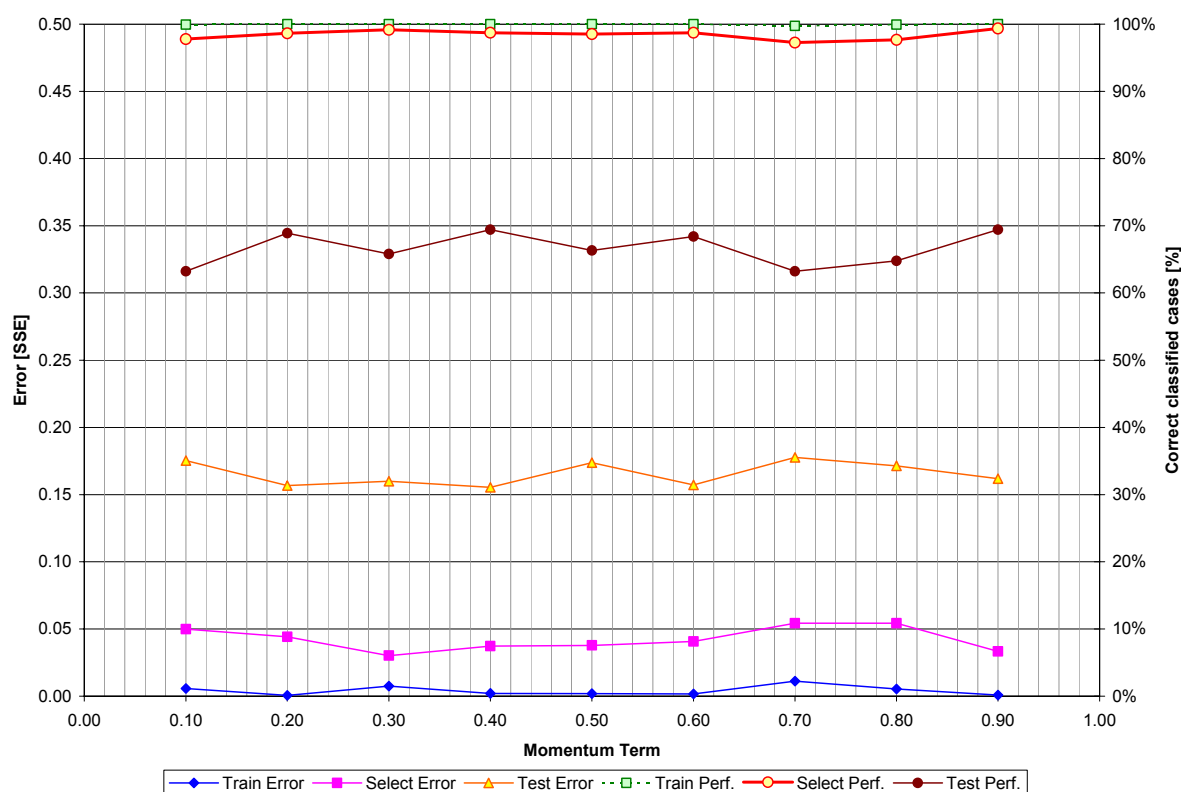


Figure 189: Momentum term comparison
(gal_mini3_sh05_mod80 step 20)

Figure 189 shows clear without ambiguity, that the classification task does not depend on the used momentum term during training.

5.8.6.4. Noise values

As shown in literature ^[300, 305, 308-311], the addition of Gaussian noise to the input values of the training pattern file can improve the generalization ability of a neural network. Therefore 26 neural networks with different noise values and 10 hidden units were trained 10 times (1000 cycles Back-propagation, learning rate 0.1, pattern shuffling enabled and followed by 1000 cycles CG).

Noise can also be regarded as vertical shift. Shifting the ppm-values up and down results in a horizontal shift.

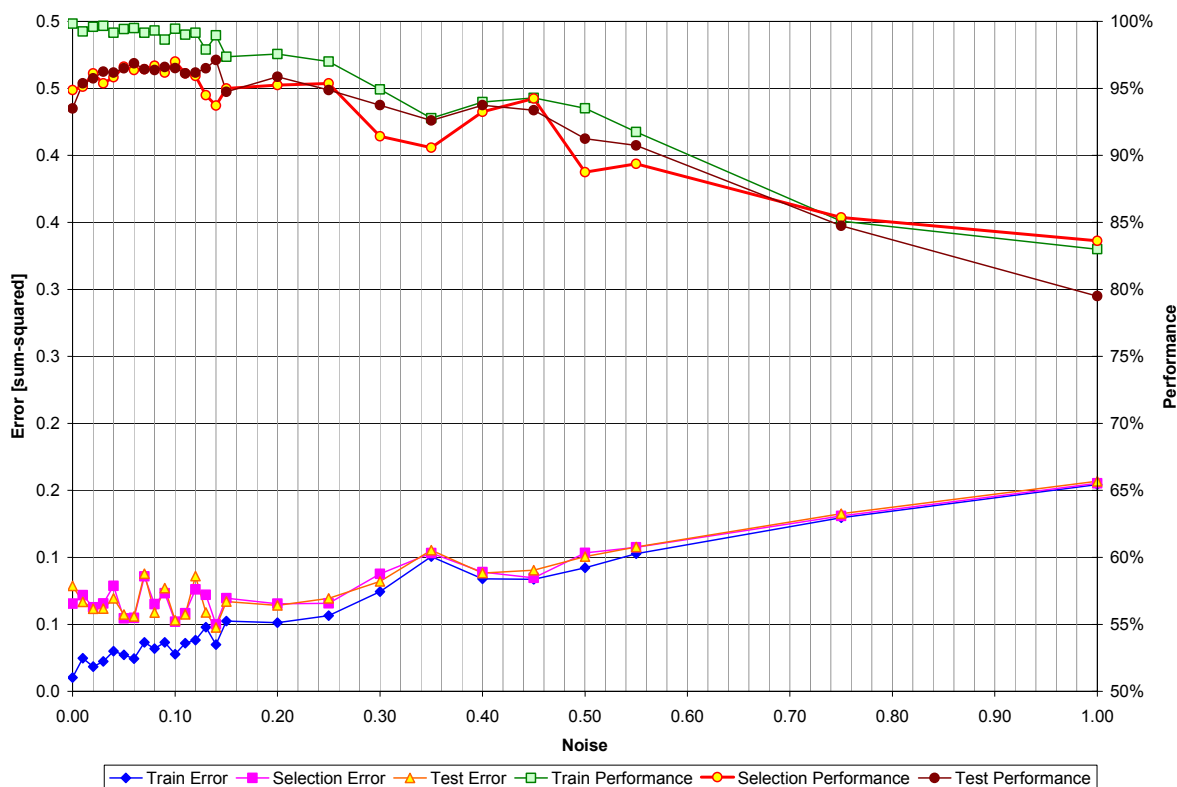


Figure 190: Performance and error comparison of different noise levels
(Exp11: gal_mini3_sh05_mod80 step 20)

To much noise starts to "disturb" the learning process of a Back-propagation neural network as shown in the figure by the decreasing performance and increasing error values. Whereas the performance of networks trained without any noise is significantly lower than networks trained with only a little noise. An optimal value for noisy training seems to be around 0.1.

5.8.6.5. Optimal pattern step size determination

The step size defines the number of data points, by which an NMR peak is represented in the pattern file. Or defines the number of adjacent input units who are activated by one NMR peak. The step size is the only ANN PFG parameter to directly affect the input layer size of the neural network. Therefore, an optimal step size value has to be determined experimentally.

For this purpose five neural networks (step size 10, 15, 20, 25 and 30) were trained 10 times (2000 cycles Back-propagation, learning rate 0.1, pattern shuffling enabled and followed by 1000 cycles CG). The whole setup was repeated with pattern files with ± 0.5 ppm horizontal shift.

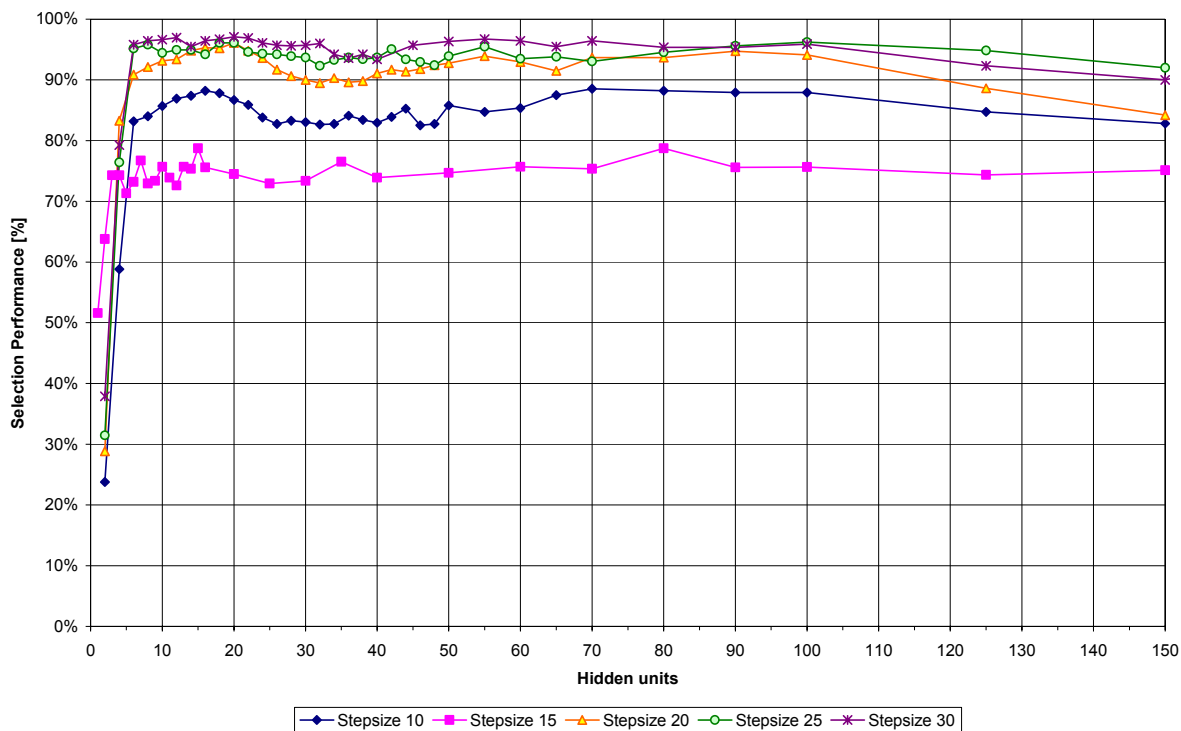


Figure 191: Selection performance of different pattern step sizes (gal_mini4_sh01_mod80)

All testes pattern files have a selection performance maximum around 15 – 22 hidden units. The maximum values of the curves differ only a little around 20 hidden units. It cannot be excluded, that step sizes > 20 data points will miss out important NMR peaks. Therefore, all future pattern files will be processed with a step size of 20 at most.

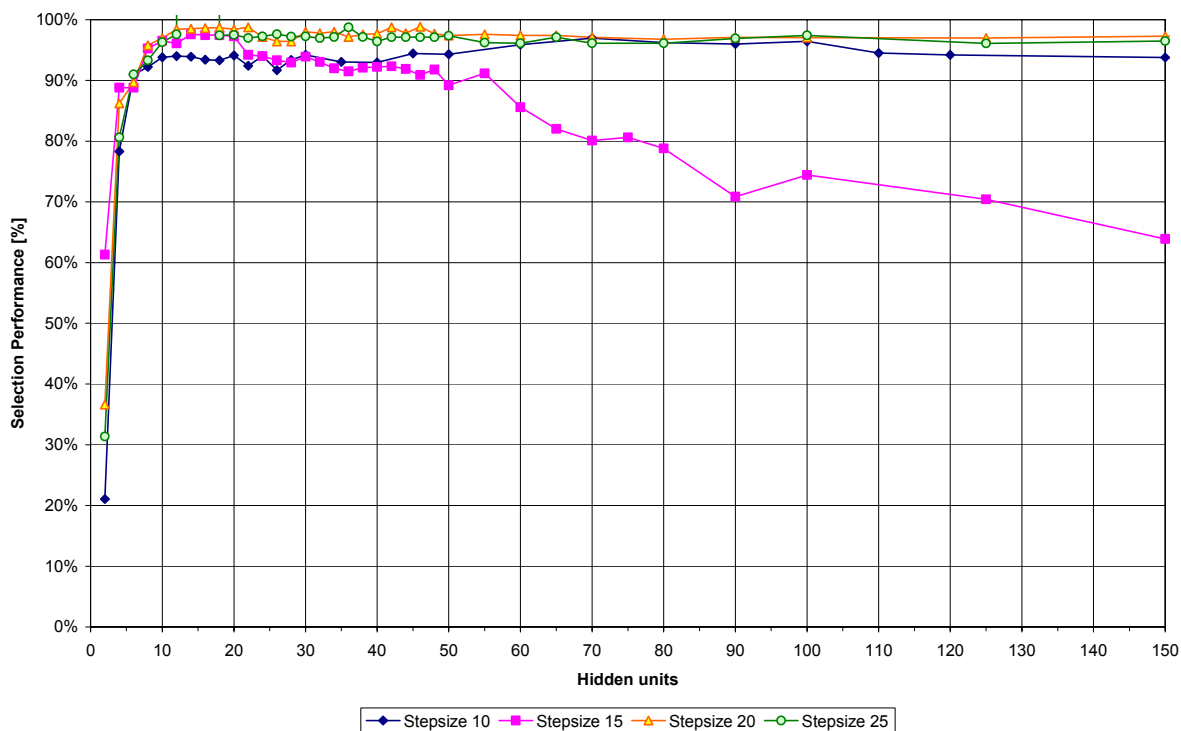


Figure 192: Selection performance of different pattern step sizes (gal_mini4_sh05_mod80)

As seen in the previous experiment with 0.1 ppm shift, the curves show the same order. But they are settled in a narrower region between 90% and 100% selection performance. The best performance is also achieved in the region of 20 hidden units. A step size of 20 points seems to be the best choice. Smaller sizes lead to bigger networks with more input units (and longer training times) and larger step sizes hold the possibility of missing peaks.

5.8.6.6. Conclusion of the preliminary experiments

The experiments in chapter 5.8.6 lead to the following conclusions:

- The network hidden layer size will be defined in the region of 10 – 20 hidden neurons
- The pattern files will contain 80 modifications of each monosaccharide moiety. Patterns with more than 100 modifications lead to a slightly better network performance but the amount of time needed to generate the pattern files is too big.
- The 80 modifications will be shifted in the range of the standard deviation of the corresponding peak of the same monosaccharide moiety.
- The ANN PFG will parse the input file with a grid size of 20 points
- The noise level during training will be fixed at ± 0.1
- The learning rate will be kept constant at 0.1 and likewise the momentum term

5.8.7. Glucose

Table 32: Training parameters for glucose network

Training pattern file	Glucose (Table 25)
Test pattern file	Glucose (Table 28)
Modifications	80
Shift	standard deviation
ANN PFG V.0.9 reading step size	20 points
Total training cycles	3000 cycles 2000 cycles Back-propagation 1000 cycles conjugated gradient
Learning rate	0.1
Momentum term	0.5
Noise level	± 0.1
Pattern shuffling	enabled
Number of networks trained	32

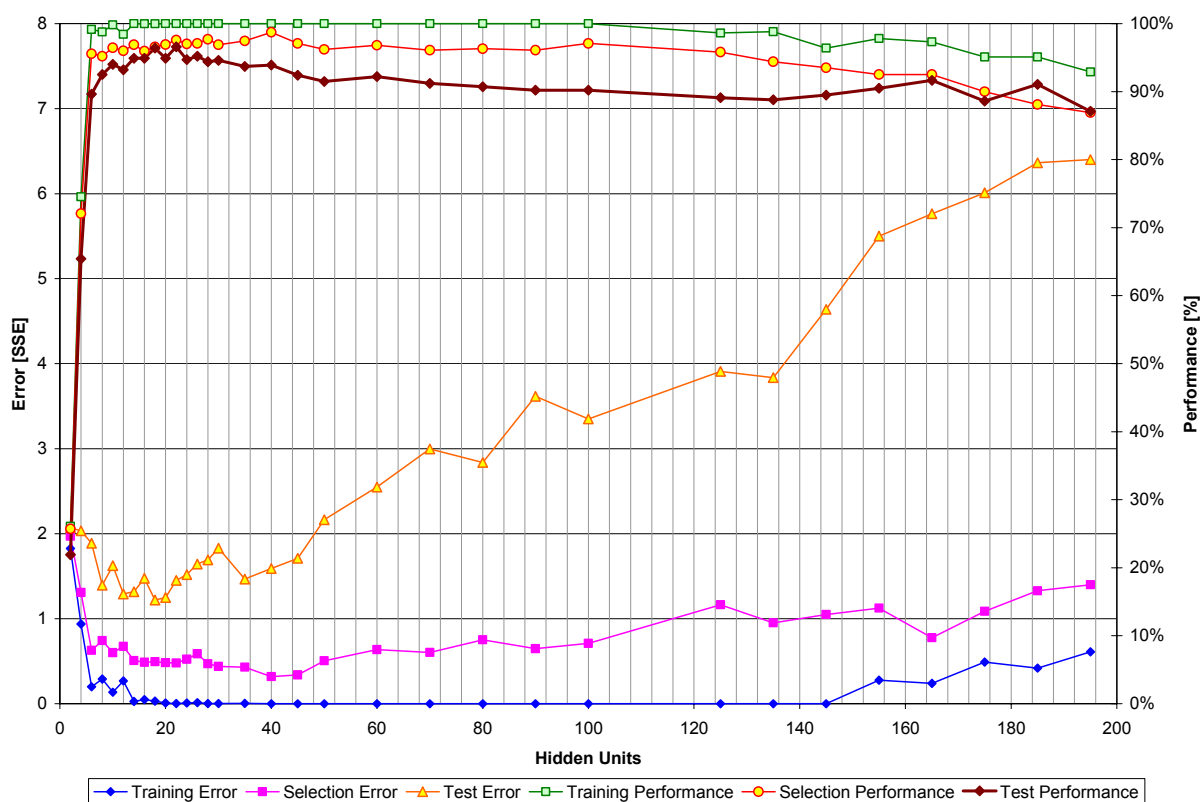


Figure 193: Glucose performance and error visualization chart (glc_mini3_sh_stdabw_mod80 step 20)

Surprisingly the best performing network had a hidden layer size of 40 hidden units. The selection performance of the glucose monosaccharide moiety test file was 98.72% with a selection error of 0.32. The following tests were carried out with this trained neural network.

Table 33: Glucose disaccharide test results of self measured test compounds

	Reference output	Recognition	Confidence
OH1	β -D-Galp-1-4- β -D-Glcp-OMe	β -D-Glcp-OMe-4R	0.8725
OH2	α -L-Fucp-1-3-(α -L-Fucp-1-2)- β -D-GlcpNAc-OMe	β -D-Glcp-OMe-4R	0.8622
OH3	α -L-Fucp-1-3- β -D-Galp-OMe	-	-
OH5	α -D-Fucp-1-3- β -D-GlcpNAc-OMe	-	-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	α -D-Glcp-OH-4R	0.7689
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe	α -D-Glcp-1R	0.9547
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	α -D-Glcp-OMe-6R	0.9136
RS1	β -D-Glcp-OMe	β -D-Glcp-OMe	0.9712
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	β -D-Glcp-1R	0.8657
Trehalose	α -D-Glcp-1-1- α -D-Glcp	α -D-Glcp-1R	0.9829
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp	α -D-Glcp-OH	0.7761
Lactose	β -D-Galp-1-4- β -D-Glcp	-	-
Saccharose	α -D-Glcp-1-2- β -Fruf	α -D-Glcp-1R	0.8108

The orange highlighted test compounds OH7, OH8, OH9, RS2 and Gentiobiose show the biggest problem of this approach. If disaccharides with two monosaccharide moieties from the same sugar (like Glc-Glc in OH7, OH8, OH9, RS2 and Gentiobiose) are presented to the trained galactose neural network, only one monosaccharide moiety exceeds the confidence level of 0.75 because the corresponding output neuron is the winner of this test run. The other monosaccharide moiety is also activated but is not exceeding the confidence level. This finding also explains the relatively poor test results of the following disaccharide test results in Table 34.

Table 34: Disaccharide test analysis for glucose network

Positive test	Total test moieties	Correct	Not recognized	false positive
Glucose	216	100 (= 46.30%)	116 (=53.70%)	25
Negative test				
Mannose	167			19
Galactose	189			21

5.8.8. Galactose

Table 35: Training parameters for galactose network

Training pattern file	Galactose (Table 25)
Test pattern file	Galactose (Table 29)
Modifications	80
Shift	standard deviation
ANN PFG V.0.9 reading step size	20 points
Total training cycles	3000 cycles 2000 cycles Back-propagation 1000 cycles conjugated gradient
Learning rate	0.1
Momentum term	0.5
Noise level	± 0.1
Pattern shuffling	enabled
Number of networks trained	36

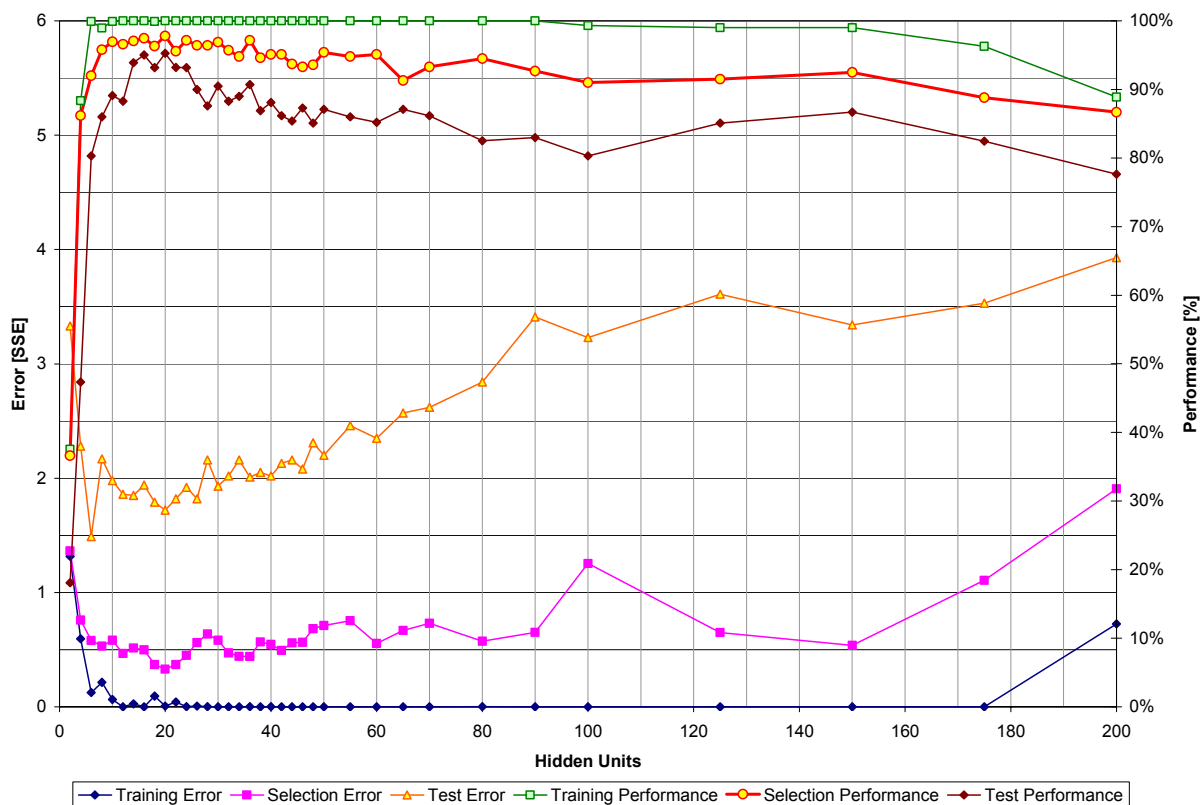


Figure 194: Galactose performance and error visualization chart (gal_mini3_sh_stdabw_mod80 step 20)

The visualization of the galactose networks reflects the findings of the preliminary experiments. The best selection performance of 97.8% (selection error 0.33) is reached with a network of 20 hidden units. The following tests were carried out with this network architecture.

Table 36: Galactose disaccharide test results of self-measured test compounds

	Reference output	Recognition	Confidence
OH1	β -D-Galp-1-4- β -D-Glcp-OMe	β -D-Galp-OMe	0.9924
OH2	α -L-Fucp-1-3-(α -L-Fucp-1-2)- β -D-GlcpNAC-OMe	-	-
OH3	α -L-Fucp-1-3- β -D-Galp-OMe	α -D-Galp-OMe-3R	0.7810
OH5	α -D-Fucp-1-3- β -D-GlcpNAC-OMe	-	-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	α -D-Galp-OMe-2R	0.8023
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe	-	-
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	-	-
RS1	β -D-Glcp-OMe	-	-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	-	-
Trehalose	α -D-Glcp-1-1- α -D-Glcp	β -D-Galp-1R	0.7629
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp	-	-
Lactose	β -D-Galp-1-4- β -D-Glcp	β -D-Galp-1R	0.7546
Saccharose	α -D-Glcp-1-2- β -Fruf	-	-

The OH1 disaccharide test compound in Table 36 shows other difficulties of this single network approach. The neural network cannot separate the ^{13}C signals according to their spin system. In the example of OH1 the network doesn't know if the OMe peak belongs to the galactose or the glucose moiety. A possibility to master this problem is the introduction of the combination generator into the ANN PFG V.0.9 as explained in chapter 4.9.6.4. This subprogram was not used during this PhD thesis. Ongoing experiments of Andreas Stoeckli are very promising to solve the spin system separation problem.

Table 37: Disaccharide test analysis for galactose network

Positive test	Total test moieties	Correct	Not recognized	false positive
Galactose	189	73 (= 38.62%)	116 (=61.38%)	14
Negative test				
Mannose	167			17
Glucose	216			28

5.8.9. Mannose

Table 38: Training parameters for mannose network

Training pattern file	Mannose (Table 26)
Test pattern file	Mannose (Table 29)
Modifications	80
Shift	standard deviation
ANN PFG V.0.9 reading step size	20 points
Total training cycles	3000 cycles 2000 cycles Back-propagation 1000 cycles conjugated gradient
Learning rate	0.1
Momentum term	0.5
Noise level	± 0.1
Pattern shuffling	enabled
Number of networks trained	40

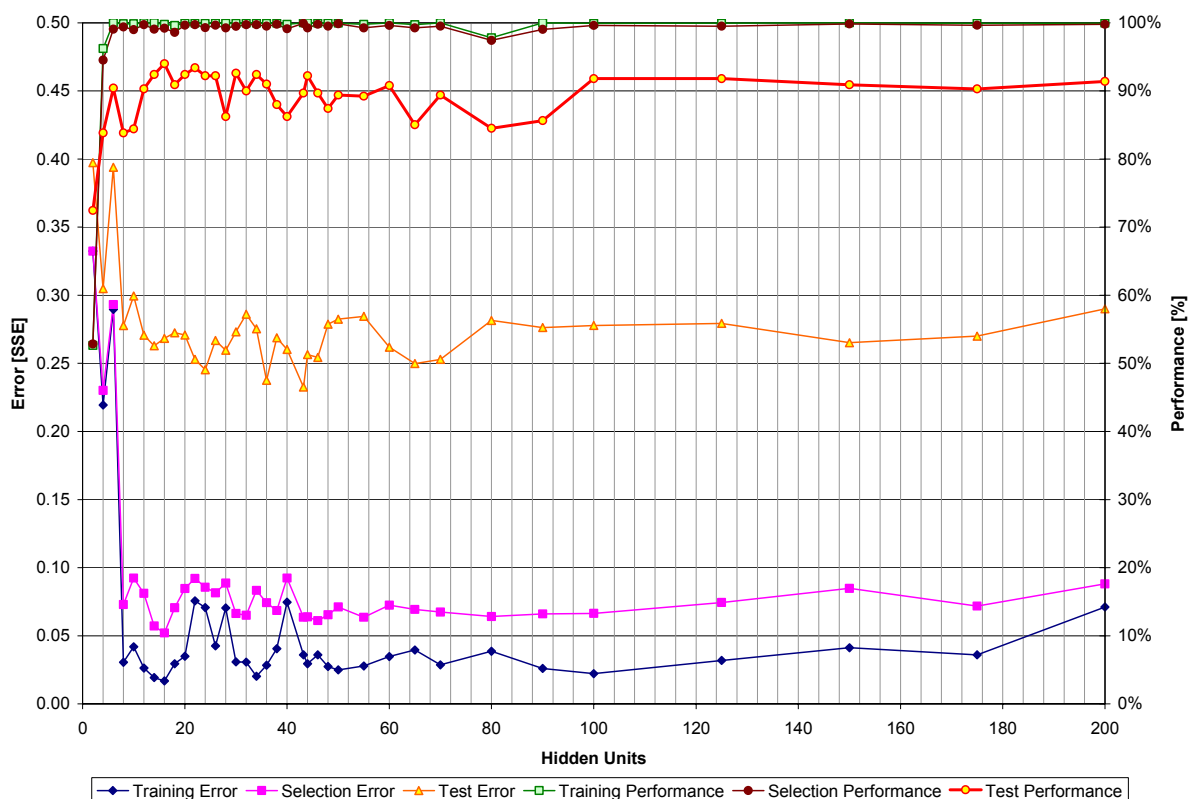


Figure 195: Mannose performance and error visualization chart (man_mini3_sh_stdabw_mod80 step 20)

Figure 195 reflects again the findings of the preliminary experiments. The best selection performance of 94.15% (selection error 0.051) is reached with a network of 20 hidden units. The following tests were carried out with this network architecture.

Table 39: Mannose disaccharide test results of self-measured test compounds

	Reference output	Recognition	Confidence
OH1	β -D-Galp-1-4- β -D-Glcp-OMe	-	-
OH2	α -L-Fucp-3-(α -L-Fucp-1-2)- β -D-GlcpNAc-OMe	-	-
OH3	α -L-Fucp-1-3- β -D-Galp-OMe	-	-
OH5	α -D-Fucp-1-3- β -D-GlcpNAc-OMe	-	-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	-	-
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe	α -D-Manp-1R	0.8842
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	-	-
RS1	β -D-Glcp-OMe	-	-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	-	-
Trehalose	α -D-Glcp-1-1- α -D-Glcp	-	-
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp	α -D-Manp-OH	0.7925
Lactose	β -D-Galp-1-4- β -D-Glcp	-	-
Saccharose	α -D-Glcp-1-2- β -Fruf	β -D-Manp-OMe-4R	0.7502

Because there are no mannose disaccharides in the test file, the results are not completely comparable with the two previous disaccharide test evaluations of glucose and galactose. The three false positive mannose recognitions and their high confidence level cannot be explained.

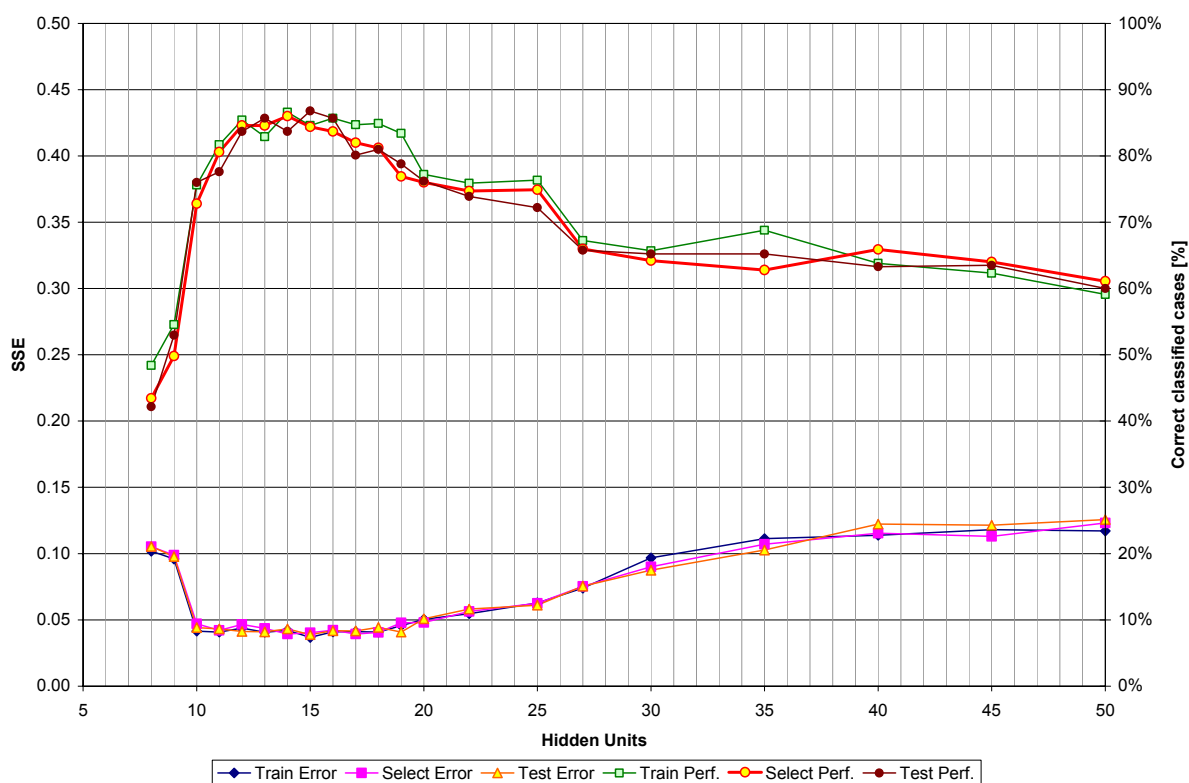
Table 40: Disaccharide test analysis for mannose network

Positive test	Total test moieties	Correct	Not recognized	false positive
Mannose	167	91 (= 54.49%)	76 (=45.51%)	27
Negative test				
Glucose	261			18
Galactose	189			21

5.8.10. Combination of glucose, galactose and mannose (GAM)

As a final experiment of this section, all training and test pattern files were merged into one single pattern file.

Training pattern file	Combination of galactose, glucose and mannose data sets
Test pattern file	Combination of galactose, glucose and mannose test data sets
Modifications	80
Shift	standard deviation
ANN PFG V.0.9 reading step size	20 points
Peak Mask	GAM
Total training cycles	2000 cycles 1500 cycles Back-propagation 500 cycles conjugated gradient
Learning rate	0.1
Momentum term	0.5
Noise level	± 0.1
Pattern shuffling	enabled
Number of networks trained	36



As already shown in experiment 5.7.6, it is not possible to classify all monosaccharide moieties of galactose, glucose and mannose with one single Kohonen feature map. The same task cannot be satisfyingly accomplished with a single Back-propagation neural network. The best performing networks are located in the region of 10 – 20 hidden units.

Table 41: Mannose disaccharide test results of self-measured test compounds

	Reference output	Recognition	Confidence
OH1	β -D-Galp-1-4- β -D-Glcp-OMe	-	-
OH2	α -L-Fucp-1-3-(α -L-Fucp-1-2)- β -D-GlcpNAc-OMe	-	-
OH3	α -L-Fucp--1-3- β -D-Galp-OMe	-	-
OH5	α -D-Fucp--1-3- β -D-GlcpNAc-OMe	-	-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	-	-
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe	α -D-Manp-1R	0.8842
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	-	-
RS1	β -D-Glcp-OMe	-	-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	-	-
Trehalose	α -D-Glcp-1-1- α -D-Glcp	-	-
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp	α -D-Manp-OH	0.7925
Lactose	β -D-Galp-1-4- β -D-Glcp	-	-
Saccharose	α -D-Glcp-1-2- β -Fruf	β -D-Manp-OMe-4R	0.7502

The results in Table 41 show the same problems already discussed in chapter 5.8.7 and 5.8.8. The single neural network trained with galactose, glucose and mannose can only recognize one monosaccharide moiety at once. The other moiety is not exceeding the confidence level of 0.75. And the problem of the spin system separation persists.

A partly explanation of the bad selection performance is depicted in the following figure:

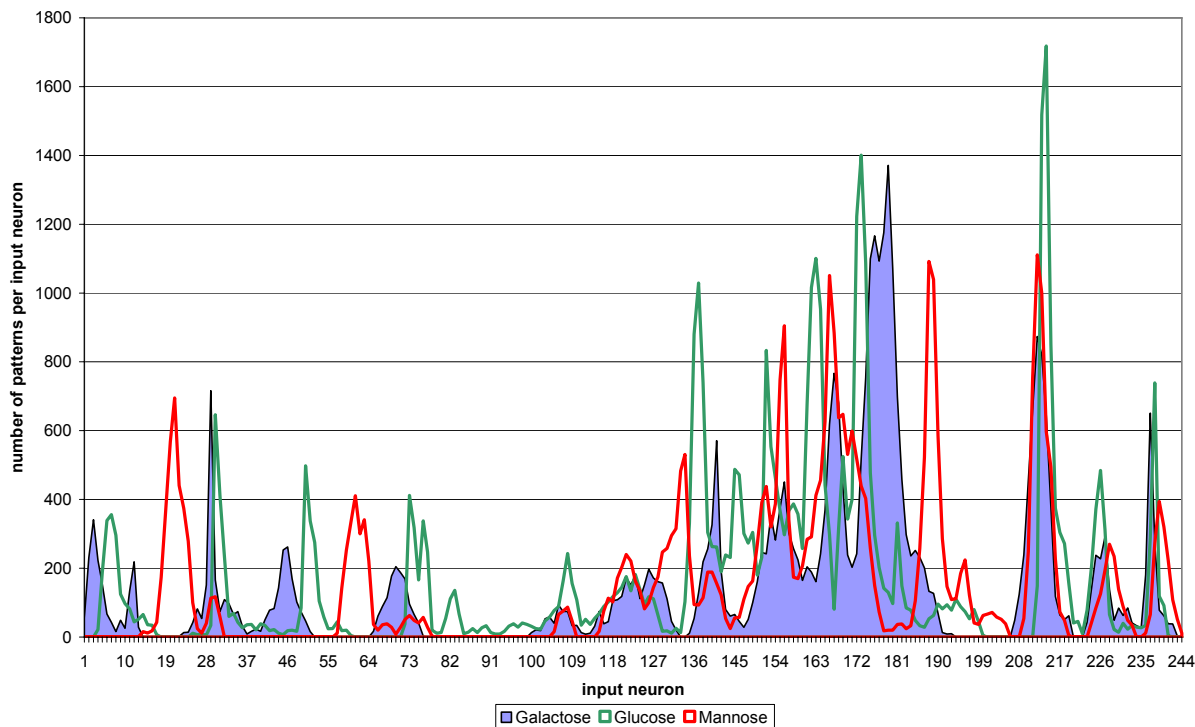


Figure 196: Number of activating patterns per input neuron

This figure shows the accumulated input activations per input neuron of all modifications of the combined GAM training and test patten file. The region between input neuron 100 and 190 shows heavy overlaps of all three sugars. Therefore, the neural network cannot consult this region for its decision making and has to depend on the remaining input regions. As the results of the preceding

experiments show clearly, this problem does not occur if separated neural networks are trained for each sugar. Another approach to help to overcome the problem would be to generate selective peak masks for every sugar in the ANN PFG. Therefore, it's possible to exclude at least some peaks regions of the other monosaccharide types not needed in the pattern file.

The combined approach was now completely abandoned.

5.9. Ensemble approach

5.9.1. The concept

The basic idea of the so called ensemble approach is the feature of Statsoft Statistica to build an ensemble of a group of similar trained neural networks. The intelligent problem solver (IPS) is an algorithm who creates networks and automatically trains them for a certain time or until a certain performance level is reached. The user decides how many networks should finally be retained and stored in an ensemble. The classification threshold when to accept or the reject a network can also be chosen individually.

When a neural network is trained several times with the same training pattern file and the same training parameters, its performance (generalization ability) will be different for every network because of different random starting weight values, Gaussian noise and pattern shuffling. The training algorithm will find a different minimum on the error surface. Therefore, the idea arose to train at least 20 similar neural networks for each monosaccharide (glucose, galactose and mannose). Every single trained network of this ensemble will be a kind of a specialist for certain monosaccharide moieties. When a test compound is simultaneously presented to all the networks of the ensemble, every single "expert" network will recognize its favorite monosaccharide moiety. There will also be false predictions but the prediction with the highest frequency will be assessed as the winner. Therefore, all the predictions of the ensemble can be statistically analyzed and a likelihood can be calculated.

In each of the following experiments, the IPS was running for 48 hours and the 20 networks with the best selection performance were retained and joined into an ensemble.

All experiments were carried out with a maximum limit of 3000 cycles per network (2000 cycles Back-propagation and 1000 cycles CG). The pattern files were generated with 22-point raster size and a combined GAM mask (combination of glucose, galactose and mannose mask file).

5.9.2. Glucose ensemble networks with one and two hidden layers

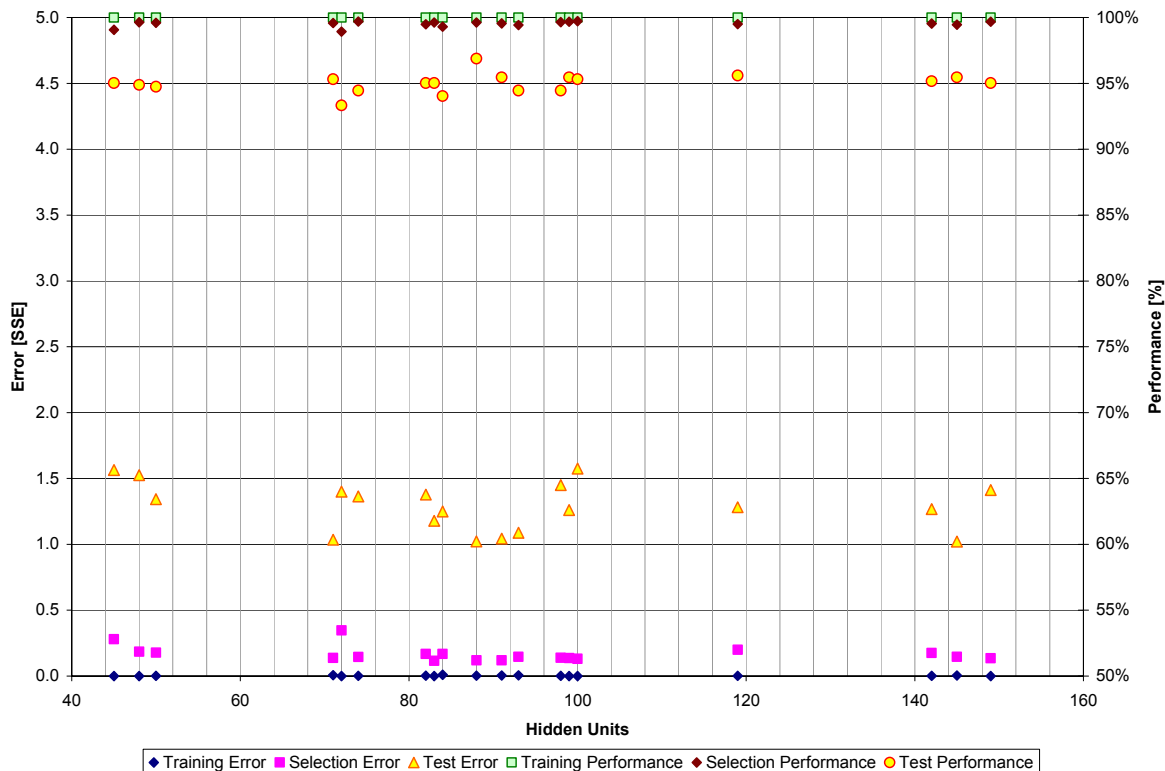


Figure 197: Graphical IPS performance and error comparison for glucose ensemble networks with one hidden layer

Table 42: Glucose IPS ensemble performance and error summary (one hidden layer) sorted by hidden layer size.

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-45-23	100.00%	99.07%	95.03%	0.000003	0.280000	1.564000	BP2000,CG44b
MLP 244-48-23	100.00%	99.64%	94.89%	0.000271	0.185791	1.525683	BP2000,CG37b
MLP 244-50-23	100.00%	99.60%	94.74%	0.001723	0.178278	1.343558	BP2000,CG32b
MLP 244-71-23	100.00%	99.58%	95.31%	0.007425	0.137773	1.034577	BP2000,CG38b
MLP 244-72-23	100.00%	98.93%	93.32%	0.000001	0.347000	1.400000	BP2000,CG41b
MLP 244-74-23	100.00%	99.69%	94.46%	0.001823	0.144796	1.363462	BP2000,CG45b
MLP 244-82-23	100.00%	99.49%	95.03%	0.002369	0.167917	1.378030	BP2000,CG37b
MLP 244-83-23	100.00%	99.63%	95.03%	0.000689	0.115678	1.178307	BP2000,CG35b
MLP 244-84-23	100.00%	99.30%	94.03%	0.009855	0.169066	1.249409	BP2000,CG24b
MLP 244-88-23	100.00%	99.63%	96.88%	0.003035	0.119123	1.023239	BP2000,CG30b
MLP 244-91-23	100.00%	99.55%	95.45%	0.004790	0.119619	1.043912	BP2000,CG26b
MLP 244-93-23	100.00%	99.43%	94.46%	0.004948	0.146625	1.088620	BP2000,CG25b
MLP 244-98-23	100.00%	99.66%	94.46%	0.000904	0.138716	1.450450	BP2000,CG37b
MLP 244-99-23	100.00%	99.67%	95.45%	0.000423	0.136588	1.260370	BP2000,CG25b
MLP 244-100-23	100.00%	99.72%	95.31%	0.000312	0.130668	1.576228	BP2000,CG30b
MLP 244-119-23	100.00%	99.50%	95.60%	0.001541	0.199001	1.282266	BP2000,CG32b
MLP 244-142-23	100.00%	99.53%	95.17%	0.001119	0.176200	1.267603	BP2000,CG25b
MLP 244-145-23	100.00%	99.46%	95.45%	0.003791	0.146485	1.021856	BP2000,CG26b
MLP 244-149-23	100.00%	99.67%	95.03%	0.000353	0.134548	1.413274	BP2000,CG26b

The single layer neural networks show a very good performance distribution. The selection performance is almost independent from the hidden layer size and stays in a very narrow band around 95%. All networks were trained within the requested 3000 cycles maximum training time. A rising selection error curve cannot be noticed. The twenty trained neural networks in Table 42 form an optimal ensemble for the glucose recognition task.

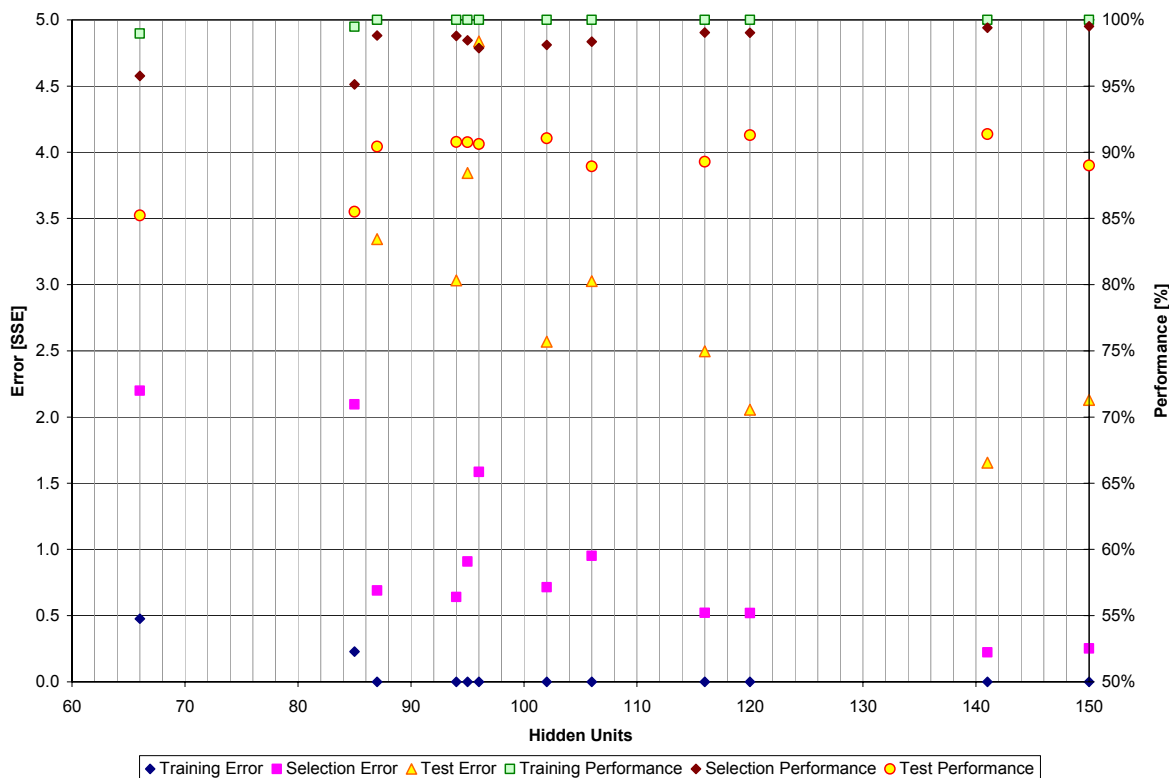


Figure 198: Graphical IPS performance and error comparison for glucose ensemble networks with two hidden layers

The networks with more free parameters (= more hidden layers) do not reach the selection performance of the single hidden layer networks trained with exactly the same pattern and test files. The two-hidden layer networks show a bigger scatter of the error and performance values. In contrast to the preliminary experiments, the selection error curve is slightly decreasing instead of increasing to higher error values. Because of the relatively large test errors, no dual layer network was used in the ensemble for the following tests.

Table 43: Glucose IPS ensemble performance and error summary (two hidden layers) sorted by hidden layer size.

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-66-70-23	98.97%	95.76%	85.23%	0.476894	2.199564	5.837237	BP2000,CG29b
MLP 244-85-74-23	99.48%	95.12%	85.51%	0.228416	2.095198	6.735819	BP2000,CG33b
MLP 244-87-56-23	100.00%	98.82%	90.42%	0.000003	0.689957	3.344393	BP2000,CG45b
MLP 244-94-63-23	100.00%	98.79%	90.77%	0.000001	0.641492	3.032390	BP2000,CG41b
MLP 244-95-61-23	100.00%	98.45%	90.77%	0.000001	0.909111	3.843496	BP2000,CG47b
MLP 244-96-80-23	100.00%	97.87%	90.63%	0.000005	1.586145	4.837241	BP2000,CG37b
MLP 244-102-64-23	100.00%	98.11%	91.05%	0.000111	0.715020	2.569745	BP2000,CG32b
MLP 244-106-60-23	100.00%	98.35%	88.94%	0.000001	0.952447	3.026520	BP2000,CG43b
MLP 244-116-90-23	100.00%	99.04%	89.29%	0.000004	0.520583	2.497200	BP2000,CG41b
MLP 244-120-68-23	100.00%	99.02%	91.29%	0.000045	0.520457	2.055503	BP2000,CG35b
MLP 244-141-85-23	100.00%	99.39%	91.38%	0.000149	0.222354	1.655640	BP2000,CG33b
MLP 244-150-111-23	100.00%	99.52%	89.00%	0.000026	0.252739	2.129605	BP2000,CG34b

Table 44: Glucose disaccharide test results of self-measured test compounds

	Reference output	Recognition			
OH1	β -D-Galp-1-4- β -D-Glcp-OMe		-	20	β -D-Glcp-OMe-4R
OH2	α -L-Fucp--1-3-(α -L-Fucp--1-2)- β -D-GlcpNAC-OMe		-		-
OH3	α -L-Fucp--1-3- β -D-Galp-OMe		-		-
OH5	α -D-Fucp--1-3- β -D-GlcpNAC-OMe		-		-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	8	β -D-Glcp-OMe-4R	8	α -D-Glcp-1R
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe		-		-
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	9	α -D-Glcp-1R	3	α -D-Glcp-OMe-6R
RS1	β -D-Glcp-OMe	12	β -D-Glcp-OMe		-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	7	β -D-Glcp-1R	4	β -D-Glcp-OMe-6R
Trehalose	α -D-Glcp-1-1- α -D-Glcp	8	α -D-Glcp-1R		
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp		??	6	β -D-Glcp-OH-6R
Lactose	β -D-Galp-1-4- β -D-Glcp	5	β -D-Galp-1R	12	α -D-pGlc-OH-3R
Saccharose	α -D-Glcp-1-2- β -Fruf	6	α -D-Glcp-1R		

The outstanding disaccharide test results prove that the new ensemble approach leads to the desired network recognition performance. The test analysis in Table 44 shows only one false recognition in the Lactose test compound and a missing monosaccharide moiety in the Gentiobiose test compound. However, it must be said, that the first monosaccharide moiety (β -D-Glcp-1R) nearly reached the necessary confidence level of 0.75 (it was 0.723). The Trehalose compound shows the only known drawback of the ensemble approach: if a disaccharide consists of two identical monosaccharide moieties, it is not possible to recognize them as two separate units. The corresponding NMR peak list only consists out of six peaks and contains no information about a second monosaccharide moiety. A possible solution would be to take the peak intensities into account. But this information is hardly available in literature data and was not included in the whole concept of this thesis.

Table 45: Disaccharide test analysis for glucose networks

Positive test	Total test moieties	Correct	Not recognized	false positive
Glucose	216	163 (= 75.46%)	53 (=24.54%)	8
Negative test				
Mannose	167			9
Galactose	189			5

In comparison with Table 34, there is a significant improvement of the number of correct recognized glucose test compounds.

5.9.3. Galactose ensemble networks with one and two hidden layers

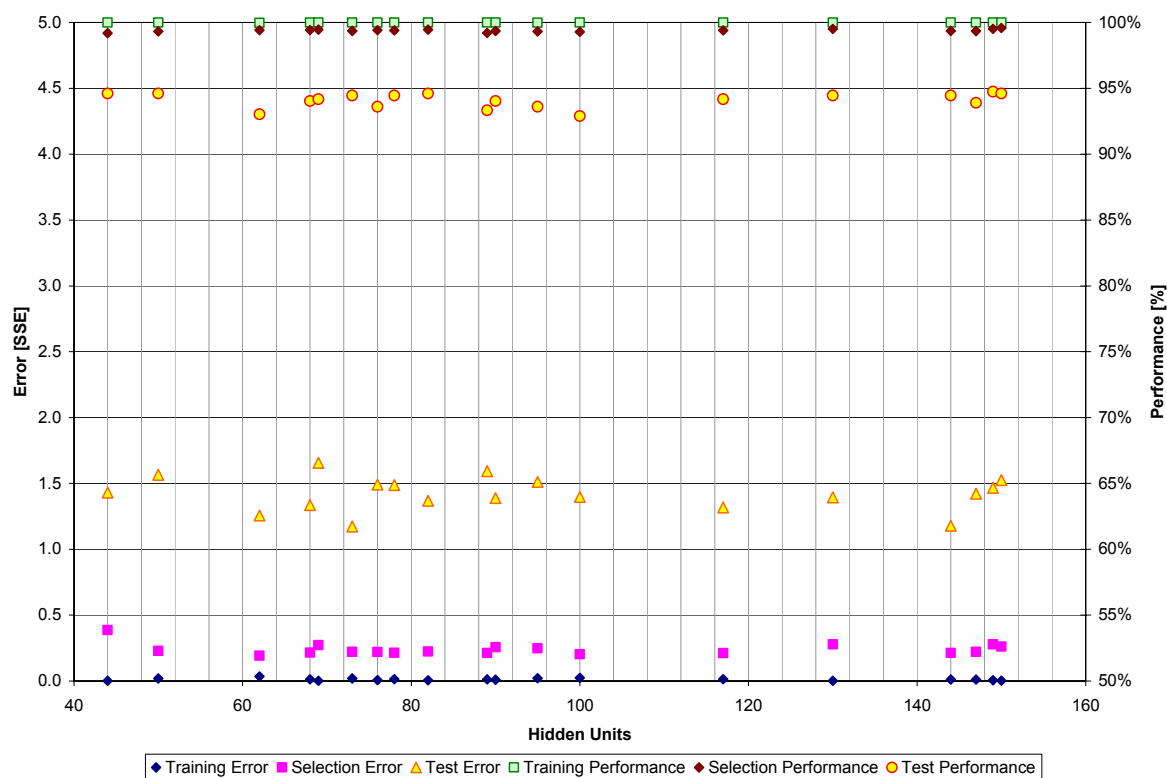


Figure 199: Graphical IPS performance and error comparison for galactose ensemble networks with one hidden layer

Table 46: Galactose IPS ensemble performance and error summary (one hidden layer)

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-44-20	100.00%	99.19%	94.60%	0.000732	0.387476	1.431000	BP2000,CG35b
MLP 244-50-20	100.00%	99.32%	94.60%	0.019897	0.228701	1.566552	BP2000,CG21b
MLP 244-62-20	99.97%	99.41%	93.04%	0.034192	0.191590	1.256100	BP2000,CG21b
MLP 244-68-20	100.00%	99.43%	94.03%	0.012840	0.215844	1.335803	BP2000,CG23b
MLP 244-69-20	100.00%	99.46%	94.18%	0.000744	0.273424	1.657000	BP2000,CG33b
MLP 244-73-20	100.00%	99.36%	94.46%	0.019675	0.221567	1.172396	BP2000,CG20b
MLP 244-76-20	100.00%	99.41%	93.61%	0.007247	0.220668	1.490012	BP2000,CG23b
MLP 244-78-20	100.00%	99.39%	94.46%	0.013372	0.214837	1.488324	BP2000,CG24b
MLP 244-82-20	100.00%	99.46%	94.60%	0.005956	0.224582	1.368620	BP2000,CG25b
MLP 244-89-20	100.00%	99.21%	93.32%	0.012719	0.213383	1.593564	BP2000,CG21b
MLP 244-90-20	100.00%	99.36%	94.03%	0.009454	0.256370	1.388920	BP2000,CG25b
MLP 244-95-20	99.99%	99.32%	93.61%	0.021049	0.248335	1.510969	BP2000,CG21b
MLP 244-100-20	100.00%	99.29%	92.90%	0.023423	0.203255	1.396639	BP2000,CG11b
MLP 244-117-20	100.00%	99.39%	94.18%	0.014260	0.211702	1.317706	BP2000,CG20b
MLP 244-130-20	100.00%	99.52%	94.46%	0.001453	0.278101	1.394156	BP2000,CG27b
MLP 244-144-20	100.00%	99.36%	94.46%	0.011766	0.212916	1.178078	BP2000,CG22b
MLP 244-147-20	100.00%	99.35%	93.89%	0.011529	0.220940	1.422639	BP2000,CG20b
MLP 244-149-20	100.00%	99.52%	94.74%	0.005050	0.279242	1.466692	BP2000,CG20b
MLP 244-150-20	100.00%	99.58%	94.60%	0.003345	0.261268	1.525276	BP2000,CG20b

Figure 199 shows a very similar distribution of the error and performance values like the glucose ensemble. The selection performance of all networks lies around a very good value of 95% while the selection error stays at a low level of 0.25. No networks have to be excluded from the ensemble and will be used for the following disaccharide tests.

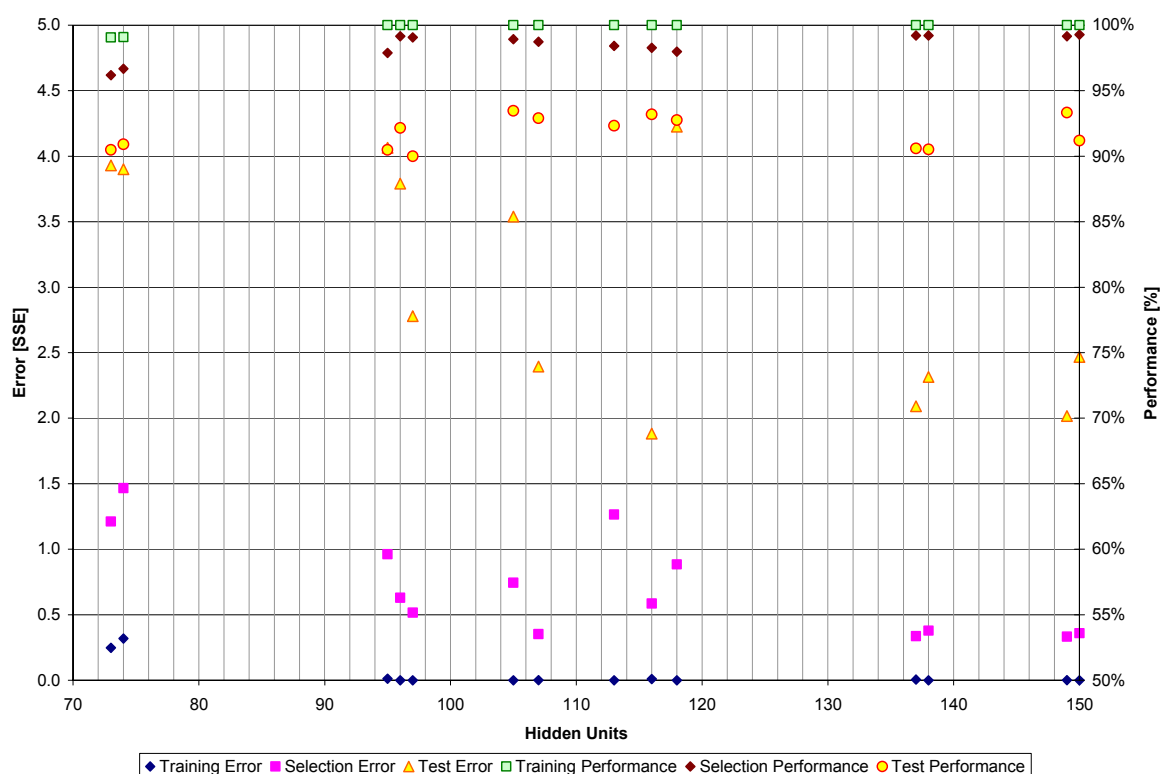


Figure 200: Graphical IPS performance and error comparison for galactose ensemble networks with two hidden layers

Table 47: Galactose IPS ensemble performance and error summary (two hidden layer)

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-73-72-20	99.07%	96.18%	90.48%	0.247851	1.211898	3.929880	BP2000,CG44b
MLP 244-74-67-20	99.08%	96.68%	90.91%	0.319151	1.466726	3.901390	BP2000,CG34b
MLP 244-95-61-20	100.00%	97.89%	90.48%	0.012841	0.962914	4.063470	BP2000,CG30b
MLP 244-96-62-20	100.00%	99.15%	92.16%	0.000001	0.630087	3.790584	BP2000,CG41b
MLP 244-97-65-20	100.00%	99.07%	90.00%	0.000230	0.516452	2.779150	BP2000,CG33b
MLP 244-105-66-20	100.00%	98.93%	93.47%	0.000011	0.745294	3.539168	BP2000,CG37b
MLP 244-107-77-20	100.00%	98.73%	92.90%	0.001576	0.352067	2.394478	BP2000,CG28b
MLP 244-113-85-20	100.00%	98.42%	92.33%	0.000001	1.265037	5.035661	BP2000,CG41b
MLP 244-116-70-20	100.00%	98.28%	93.18%	0.009099	0.586528	1.882266	BP2000,CG23b
MLP 244-118-95-20	100.00%	97.98%	92.76%	0.000429	0.884356	4.225017	BP2000,CG30b
MLP 244-137-86-20	100.00%	99.21%	90.59%	0.005148	0.336823	2.090991	BP2000,CG22b
MLP 244-138-85-20	100.00%	99.21%	90.51%	0.000636	0.378535	2.315195	BP2000,CG26b
MLP 244-149-93-20	100.00%	99.15%	93.32%	0.001094	0.332417	2.016792	BP2000,CG23b
MLP 244-150-98-20	100.00%	99.27%	91.20%	0.000495	0.359954	2.467834	BP2000,CG25b

The two-hidden layer networks lack of the same bad selection error and a somewhat lower selection performance. This ensemble will not be used for the following disaccharide test.

Table 48: Galactose disaccharide recognition test results of self-measured test compounds

	Reference output	Recognition			
OH1	β -D-Galp-1-4- β -D-Glcp-OMe		-	4	β -D-Galp-1R
OH2	α -L-Fucp--1-3-(α -L-Fucp--1-2)- β -D-GlcpNAc-OMe		-		-
OH3	α -L-Fucp--1-3- β -D-Galp-OMe		-	4	β -D-Galp-OMe-3R
OH5	α -D-Fucp--1-3- β -D-GlcpNAc-OMe		-		-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe		-		-
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe		-		-
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe		-		-
RS1	β -D-Glcp-OMe		-		-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe		-		-
Trehalose	α -D-Glcp-1-1- α -D-Glcp	4	α -D-pGal-OH-4R		-
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp		-		-
Lactose	β -D-Galp-1-4- β -D-Glcp	8	β -D-Galp-1R		-
Saccharose	α -D-Glcp-1-2- β -Fruf		-		-

The galactose disaccharide test summary shows outstanding recognition capabilities. Only for the Trehalose test disaccharide α -D-Glcp-1-1- α -D-Glcp a number of four networks deliver a false positive recognition.

Table 49: Disaccharide test analysis for galactose networks

Positive test	Total test moieties	Correct	Not recognized	false positive
Galactose	77	63 (= 82.82%)	14 (=18.18%)	4
Negative test				
Mannose	167			2
Glucose	347			4

As expected from the glucose test results, the ensemble approach also seems to be the right approach for the recognition of galactose monosaccharide moieties.

5.9.4. Mannose ensemble networks with one and two hidden layers

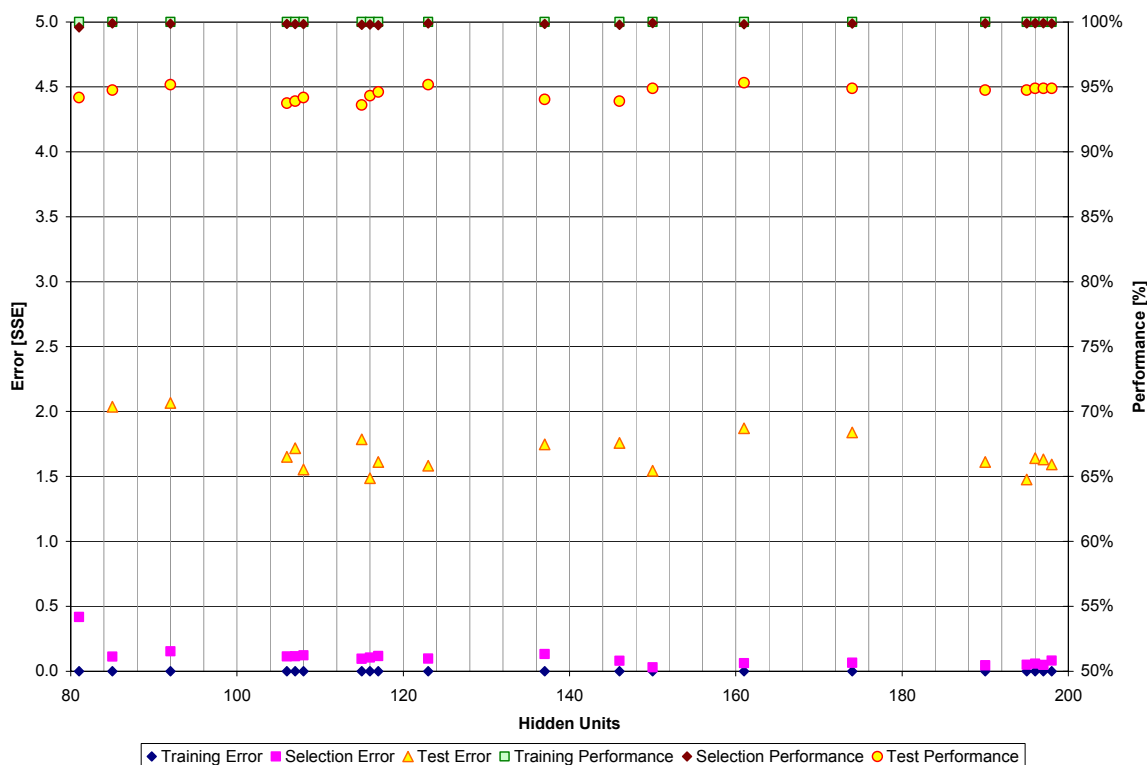


Figure 201: Graphical IPS performance and error comparison for mannose ensemble networks with one hidden layer

Table 50: Mannose IPS ensemble performance and error summary (one hidden layer)

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-81-20	100.00%	99.58%	94.18%	0.000001	0.418552	5.696214	BP2000,CG42b
MLP 244-85-20	100.00%	99.89%	94.74%	0.000000	0.113067	2.037000	BP2000,CG40b
MLP 244-92-20	100.00%	99.86%	95.17%	0.000001	0.153221	2.066000	BP2000,CG41b
MLP 244-106-20	100.00%	99.84%	93.75%	0.000001	0.114024	1.651000	BP2000,CG43b
MLP 244-107-20	100.00%	99.83%	93.89%	0.000001	0.115503	1.718000	BP2000,CG46b
MLP 244-108-20	100.00%	99.83%	94.18%	0.000001	0.123083	1.554000	BP2000,CG45b
MLP 244-115-20	100.00%	99.78%	93.61%	0.000067	0.095984	1.786000	BP2000,CG34b
MLP 244-116-20	100.00%	99.80%	94.32%	0.000011	0.105275	1.486000	BP2000,CG43b
MLP 244-117-20	100.00%	99.75%	94.60%	0.000199	0.117472	1.612000	BP2000,CG33b
MLP 244-123-20	100.00%	99.89%	95.17%	0.000041	0.098102	1.583000	BP2000,CG37b
MLP 244-137-20	100.00%	99.84%	94.03%	0.000041	0.132934	1.747000	BP2000,CG37b
MLP 244-146-20	100.00%	99.78%	93.89%	0.000492	0.080875	1.758495	BP2000,CG32b
MLP 244-150-20	100.00%	99.92%	94.89%	0.000001	0.030058	1.544000	BP2000,CG47b
MLP 244-161-20	100.00%	99.81%	95.31%	0.000162	0.063055	1.871560	BP2000,CG35b
MLP 244-174-20	100.00%	99.88%	94.89%	0.000368	0.065395	1.838493	BP2000,CG28b
MLP 244-190-20	100.00%	99.89%	94.74%	0.000089	0.046283	1.612000	BP2000,CG35b
MLP 244-195-20	100.00%	99.89%	94.74%	0.000056	0.049694	1.477000	BP2000,CG34b
MLP 244-196-20	100.00%	99.91%	94.89%	0.000002	0.058156	1.641000	BP2000,CG39b
MLP 244-197-20	100.00%	99.91%	94.89%	0.000005	0.048571	1.631000	BP2000,CG33b
MLP 244-198-20	100.00%	99.88%	94.89%	0.000003	0.082562	1.593000	BP2000,CG34b

The test and performance summary shows the best values of all tested single-hidden layer networks (glucose and galactose). The selection performance values stay at a comparable high level of about 95% like glucose and galactose but the corresponding selection error is relatively low. No network had to be rejected from the mannose ensemble.

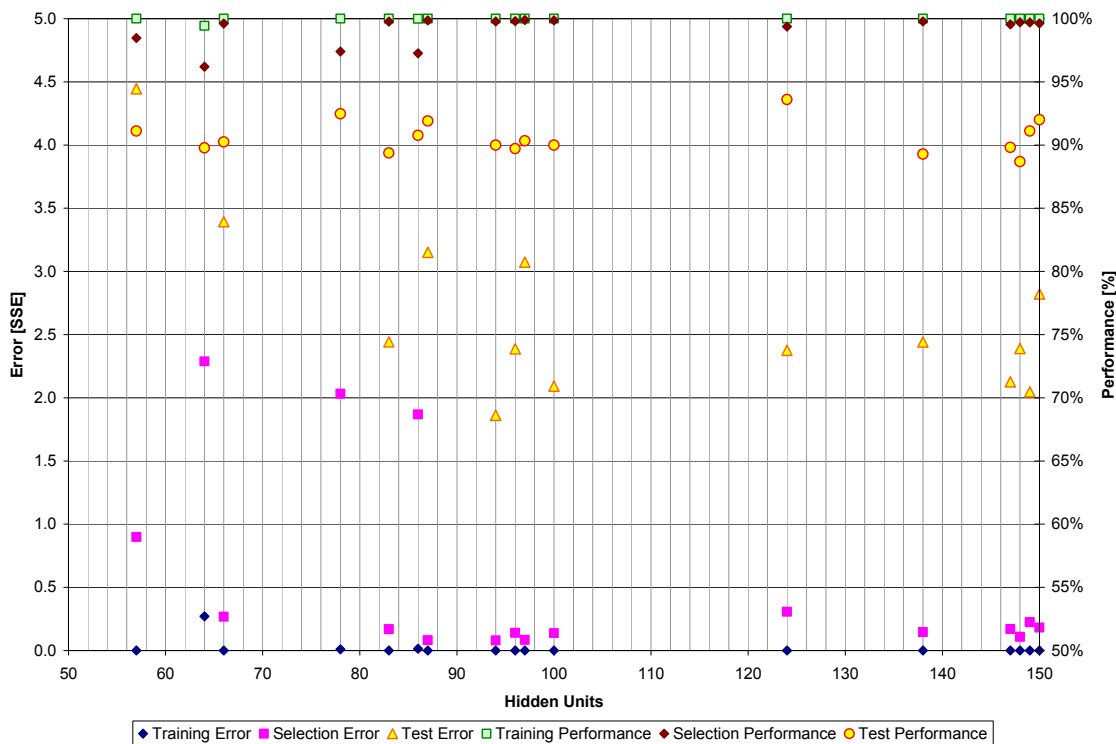


Figure 202: Graphical IPS performance and error comparison for mannose ensemble networks with two hidden layers

Table 51: Mannose IPS ensemble performance and error summary (two hidden layers)

Profile	Training Perf.	Selection Perf.	Test Perf.	Training Error	Selection Error	Test Error	Training
MLP 244-48-50-20	99.90%	96.52%	90.77%	0.089775	2.010963	8.827342	BP2000,CG45b
MLP 244-57-46-20	100.00%	98.46%	91.11%	0.000008	0.898265	4.445000	BP2000,CG40b
MLP 244-64-58-20	99.43%	96.20%	89.77%	0.270075	2.288524	6.584868	BP2000,CG45b
MLP 244-66-41-20	100.00%	99.61%	90.24%	0.000001	0.267535	3.393818	BP2000,CG53b
MLP 244-78-74-20	100.00%	97.39%	92.47%	0.009916	2.031911	5.974671	BP2000,CG33b
MLP 244-83-63-20	100.00%	99.77%	89.37%	0.000003	0.168075	2.442332	BP2000,CG39b
MLP 244-86-73-20	99.99%	97.25%	90.77%	0.013263	1.868208	8.073456	BP2000,CG26b
MLP 244-87-58-20	100.00%	99.86%	91.90%	0.000001	0.081578	3.151437	BP2000,CG42b
MLP 244-94-72-20	100.00%	99.78%	90.00%	0.000235	0.081471	1.862420	BP2000,CG29b
MLP 244-96-67-20	100.00%	99.80%	89.72%	0.000007	0.138711	2.386843	BP2000,CG39b
MLP 244-97-73-20	100.00%	99.89%	90.33%	0.000000	0.084114	3.072368	BP2000,CG39b
MLP 244-100-80-20	100.00%	99.86%	90.00%	0.000102	0.137767	2.091864	BP2000,CG26b
MLP 244-124-76-20	100.00%	99.38%	93.61%	0.000099	0.306600	2.376244	BP2000,CG30b
MLP 244-138-77-20	100.00%	99.78%	89.29%	0.000015	0.146612	2.441308	BP2000,CG35b
MLP 244-147-92-20	100.00%	99.53%	89.81%	0.000413	0.170073	2.126683	BP2000,CG24b
MLP 244-148-93-20	100.00%	99.70%	88.68%	0.000006	0.108296	2.389680	BP2000,CG38b
MLP 244-149-93-20	100.00%	99.69%	91.11%	0.000023	0.223935	2.047243	BP2000,CG31b
MLP 244-150-100-20	100.00%	99.63%	92.00%	0.000013	0.181686	2.821365	BP2000,CG30b

The mannose two hidden layer summary shown in Figure 202 displays the same disorder like the two predecessor two-hidden layer networks from glucose and galactose. The networks have too many degrees of freedom and do not make up

Table 52: Mannose disaccharide recognition test results of self-measured compounds

	Reference output	Recognition		
OH1	β -D-Galp-1-4- β -D-Glcp-OMe	-	-	-
OH2	α -L-Fucp--1-3-(α -L-Fucp--1-2)- β -D-GlcpNAc-OMe	-	-	-
OH3	α -L-Fucp--1-3- β -D-Galp-OMe	-	-	-
OH5	α -D-Fucp--1-3- β -D-GlcpNAc-OMe	-	-	-
OH7	α -D-Glcp-1-4- β -D-Glcp-OMe	-	-	-
OH8	α -D-Glcp-1-4- α -D-Glcp-OMe	-	2	α -D-pMan-OMe-2R
OH9	α -D-Glcp-1-6- α -D-Glcp-OMe	-	-	-
RS1	β -D-Glcp-OMe	-	-	-
RS2	β -D-Glcp-1-6- β -D-Glcp-OMe	-	-	-
Trehalose	α -D-Glcp-1-1- α -D-Glcp	-	-	-
Gentiobiose	β -D-Glcp-1-6- β -D-Glcp	-	-	-
Lactose	β -D-Galp-1-4- β -D-Glcp	-	-	-
Saccharose	α -D-Glcp-1-2- β -Fruf	-	-	-

As expected, The mannose neural network ensemble shows an outstanding recognition performance. Except for one false positive recognition, the networks show no confusion with other moieties from glucose or galactose. In addition, the fucose and the Fructofuranose do not interfere with the mannose recognition.

Table 53: Disaccharide test analysis for mannose networks

Positive test	Total test moieties	Correct	Not recognized	false positive
Mannose	37	32 (= 86.49%)	5 (=13.51%)	10
Negative test				
Glucose	347			5
Galactose	189			2

The mannose disaccharide test delivers by far the best test recognition rate. The false positive classified galactose and glucose test compounds are very low.

5.9.5. Discussion of the ensemble approach

The ensemble networks finally brought the long looked-for breakthrough in the disaccharide recognition. There is still room for some performance improvements by expanding and normalizing the underlying literature dataset or picking the trained neural networks to be combined into the final ensemble by hand.

The results of the literature disaccharide test are only partly comparable because there are not the same numbers of test compounds of each carbohydrates in the respective test set (61.7%, 22% galactose and 16.3% mannose). Nevertheless, the test sets were good enough to discover the drawbacks of the ensemble method. The ensemble approach seems to be the only way to identify monosaccharide moieties out of disaccharides or oligosaccharides in a later stage.

6. Discussion summary & conclusions

As all experiments are already discussed in the corresponding chapters, this section has to be considered as a summary of all achievements and problems. The main objective to develop a neural network based identification system capable of identifying monosaccharide moieties out of disaccharides from spectroscopic ^{13}C -NMR data has been fully achieved. The success of this PhD thesis is to be judged on the basis of the aims formulated in chapter 3.5:

- *Reproduction of the results and the neural network approach to identify ^1H -NMR spectra of five alditols of Meyer et al. [80-82].*

Because the sugar alditols were not available, the compounds were replaced with five methyl pyranosides (chapter 4.1.1). The methyl pyranosides had the advantage to be clearly defined at the anomeric carbon and the information of the difference between α and β configuration could be included into the training data of the neural networks. It turned out quickly, that the five methyl pyranosides did not suffice to train a neural network with good generalization rates in spite of the artificial modifications made with the ANN PFG V.0.1. The presence of the methyl peak was a distinct feature for the monosaccharide recognition. Input regions of other remaining ^1H -NMR peaks did not take part in the recognition process (the weights of the associated input neurons were set to negative values).

- *What kind of NMR data provide information about the anomeric configuration and the substitution pattern of a carbohydrate (^1H or ^{13}C -NMR)?*

The idea to develop a structure elucidation system for ^1H -NMR spectra was abandoned shortly after the start of the PhD thesis because of the better signal to noise ratio of ^{13}C -NMR spectra, because there are no disturbing water peaks in a ^{13}C -NMR spectrum and because of the clear identification of the anomeric configuration. A ^{13}C -NMR spectrum can easily be converted into a binary format because of the very sharp and narrow peaks. Whereas the wide peaks of a ^1H -NMR spectrum produce extremely large binary files and would need to be Fourier or Hadamard transformed, before they can be used as an input for a neural network.

The neural networks trained with methyl pyranosides zeroed in on the methyl peak (chapter 5.3). This peak had the biggest intensity of all peaks and was characteristic for the identification of the selected compounds.

- *What information is available/accessible through NMR spectroscopy? (monomer identity, anomeric configuration, substitution pattern).*

The Diploma thesis of Alexej Moor (chapter 5.5) and the Kohonen feature maps trained in chapter 5.7 proved that the information content of a ^{13}C -NMR peak list of a monosaccharide is high enough to clearly identify the compound. Apart from the monosaccharide identity, it is possible to recognize the anomeric configuration and the substitution pattern of the moiety. The big drawback of this approach is the fact, that a Kohonen network architecture can never identify monosaccharide moieties from disaccharides because the algorithm only selects one winning neuron after the input pattern is presented. A second monosaccharide moiety can therefore not be detected. An obvious solution would be to convert the peak list back into an JCAMP DX for NMR file (chapter 4.9.5.1) and present it point by point to the network instead of a peak list with six numbers. This conversion and later data compression task can be handled with the ANN PFG software, developed during this PhD thesis.

However, an advantage of the Counter-propagation networks is the supervised learning method and the easy interpretation of the test results because of the Grossberg layer.

- *How can spectroscopic data be transferred into a neural network?*

As discussed in the previous sections, there are several ways to feed spectroscopic data (especially ^{13}C -NMR) into a neural network. The most important point to consider is that a neural network has a fixed number of input units. A test pattern file has to have the exact same dimension as the pattern file the neural network was trained with. Thus, similar features of the training data should always activate similar input neurons. Therefore, it is obvious, that an NMR peak list can only serve as a direct input for a neural network if there are always the same numbers of peaks in the file. Methylated monosaccharides and compounds without methyl peak cannot be analyzed with the same network. The only fast reasonable way to input spectroscopic data from an NMR into a neural network would be to use the spectrum itself and assign each input neuron to a certain part (ppm or Hz range) of the spectrum. An ideal format to save and handle spectroscopic data is the IUPAC JCAMP-DX data exchange protocol (chapter 4.4).

As mentioned in the previous section, the spectroscopic data handling can be accomplished with the ANN PFG software (chapter 4.9). The software is an indispensable tool for generating and reading of JCAMP DX files and for the proper data compression and writing of the training and test pattern files. The development of the different versions of the ANN PFG and its complimentary subprograms was the major part of this PhD thesis. All necessary features needed during this work are included in the final version 0.9.

- *Which network architecture, learning algorithm and learning parameters lead to optimal results?*

The identification of monosaccharides is achievable with the help of multi-layer perceptrons, Kohonen feature maps and Counter-propagation networks. The monosaccharide peak list can serve directly as inputs to a neural network and does not have to be processed with a software tool like the ANN PFG. However, monosaccharide moieties out of disaccharides can only be identified with the help of multi-layer perceptrons (MLP) and the Back-propagation learning algorithm. But the final breakthrough was achieved only with an ensemble of at least 20 optimal trained neural networks. Each of these networks acts as a little expert and the "opinion" of all experts can then be statistically interpreted.

As demonstrated with all different major approaches, the whole monosaccharide moiety recognition problem depends almost exclusively on the underlying dataset. The amount of correct literature data sets and the number of modifications are essential for good recognition results. Learning parameters have only a minor influence on the performance. The network size plays a secondary role, however it is increasing the training time.

- *Is an identification of monosaccharide moieties out of saccharide-mixture possible at all?*

As proved in the newest ensemble approach (chapter 5.9), it is possible to identify monosaccharide moieties out of ^{13}C -NMR disaccharide spectra with a very high recognition rate between 80% and 90%. The trained networks were not optimized at all. The last section of the thesis has to be regarded as a proof-of-concept approach only. Andreas Stöckli showed in his ongoing PhD thesis, that specific optimized neural networks for e.g. fucose recognition can reach monosaccharide moiety identification rates of >95%.

The limits of the methods are not identifiable yet. No other compounds than disaccharides have been tested with the ensemble neural networks. A possible problem could become the identification of non-linear (branched) oligosaccharides, if the substitutions are located at two adjacent carbon atoms and their effect on the neighboring carbon atoms is superimpose. But this is only a hypothesis and can hopefully be disproved.

The identification of tri- or later oligosaccharides was not an aim of this work. Therefore, the only test compounds were isolated mono- and disaccharides. Test with mixtures of different carbohydrates or with interfering compounds and ions were not accomplished and will be part of future experiments.

The question whether monosaccharide moieties can be identified out of mixtures could not be answered completely. All carried out experiments with ensembles of neural networks indicate that the identification is possible.

7. Outlook

In order to successfully resume the performed work, the following steps are considered:

- To avoid problems with disaccharides with two identical monosaccharide units the combination approach will be advanced. First results from the PhD thesis of Andreas Stöckli are very promising.
- The algorithm to generate all possible combinations of n peaks will be optimized and accelerated.
- The FileMaker database should be expanded with further carbohydrates like fructose, xylose, and rhamnose etc.
- More literature data for GlcNAc, GlcN, GalNAc and ManNAc should be included into the training process of the existing ensembles.
- Two-fold substituted monosaccharide units will be included into the training process for the identification of tri- or oligosaccharides.
- To gain some insight into the knowledge of the trained networks, methods of feature extraction^[312-316] should be applied. This will help to understand, what properties or parts of a certain ^{13}C -NMR spectrum are important for a correct recognition of the monosaccharide moieties. The feature extraction method will also help to develop further data compression algorithms and speed up the training and test phases in general. Based on the extracted features an optimized network architecture (e.g. with feedback neurons) for each type of carbohydrate could possibly be developed. With the knowledge of important or unimportant parts of the input layer, these regions can be strengthened or attenuated by activating or inhibiting feedback neurons (Figure 20).
- Finally, all steps of the data preparation, pattern file generation, training and recognition process will be unified within one single application.
- This application will be equipped with a user friendly web interface to offer online access for other research institutes united under EuroCarb DB.

8. References

- [1] K. J. Yarema, C. R. Bertozzi, *Genome Biol* **2001**, 2, REVIEWS0004.
- [2] PhRMA, *Pharmaceutical Research and Manufacturers of America (PhRMA)* **2002**.
- [3] V. Tretter, F. Altmann, V. Kubelka, L. Marz, W. M. Becker, *Int Arch Allergy Immunol* **1993**, 102, 259.
- [4] M. H. Goldman, D. C. James, M. Rendall, A. P. Ison, M. Hoare, A. T. Bull, *Biotechnol Bioeng* **1998**, 60, 596.
- [5] D. Zopf, G. Vergis, *Pharmaceutical Visions* **2003**.
- [6] J. W. Dennis, M. Granovsky, C. E. Warren, *Bioessays* **1999**, 21, 412.
- [7] G. Durand, N. Seta, *Clin Chem* **2000**, 46, 795.
- [8] J. Montreuil, *Biol Cell* **1984**, 51, 115.
- [9] J. M. Fernandez, J. P. Hoeffler, *Gene Expression Systems. Using nature for the art of expression*, Academic Press, San Diego, **1999**.
- [10] I. Benz, M. A. Schmidt, *Mol Microbiol* **2002**, 45, 267.
- [11] P. M. Power, M. P. Jennings, *FEMS Microbiol Lett* **2003**, 218, 211.
- [12] P. Messner, C. Schaffer, *Fortschr Chem Org Naturst* **2003**, 85, 51.
- [13] C. Schaffer, M. Graninger, P. Messner, *Proteomics* **2001**, 1, 248.
- [14] S. R. Hamilton, P. Bobrowicz, B. Bobrowicz, R. C. Davidson, H. Li, T. Mitchell, J. H. Nett, S. Rausch, T. A. Stadheim, H. Wischnewski, S. Wildt, T. U. Gerngross, *Science* **2003**, 301, 1244.
- [15] N. Tomiya, M. J. Betenbaugh, Y. C. Lee, *Acc Chem Res* **2003**, 36, 613.
- [16] N. Tomiya, S. Narang, Y. C. Lee, M. J. Betenbaugh, *Glycoconj J* **2004**, 21, 343.
- [17] F. Altmann, E. Staudacher, I. B. Wilson, L. Marz, *Glycoconj J* **1999**, 16, 109.
- [18] J. K. Ma, M. B. Hein, *Trends Biotechnol* **1995**, 13, 522.
- [19] J. F. G. Vliegthart, L. Dorland, H. Van Halbeek, *Advances in Carbohydrate Chemistry and Biochemistry* **1983**, 41, 209.
- [20] B. Lindberg, J. Lonngren, *Methods Enzymol* **1978**, 50, 3.
- [21] D. Rolf, J. A. Bennek, G. R. Gray, *Carbohydrate Research* **1985**, 137, 183.
- [22] S. Sheng, R. Cherniak, H. van Halbeek, *Anal Biochem* **1998**, 256, 63.
- [23] E. F. Hounsell, D. Bailey, *Glycopeptides and Related Compounds* **1997**, 631.
- [24] D. Marion, K. Wuthrich, *Biochem Biophys Res Commun* **1983**, 113, 967.
- [25] J. O. Duus, C. H. Gotfredsen, K. Bock, *Chemical Reviews (Washington, D. C.)* **2000**, 100, 4589.
- [26] G. Bodenhausen, D. J. Ruben, *Chemical Physics Letters* **1980**, 69, 185.
- [27] L. E. Kay, P. Keifer, T. Saarinen, *Journal of the American Chemical Society* **1992**, 114, 10663.
- [28] A. G. Palmer, III, J. Cavanagh, P. E. Wright, M. Rance, *Journal of Magnetic Resonance (1969-1992)* **1991**, 93, 151.
- [29] A. Bax, R. H. Griffey, B. L. Hawkins, *Journal of Magnetic Resonance (1969-1992)* **1983**, 55, 301.
- [30] R. E. Hurd, B. K. John, *Journal of Magnetic Resonance (1969-1992)* **1991**, 91, 648.
- [31] L. Mueller, *Journal of the American Chemical Society* **1979**, 101, 4481.
- [32] A. Bax, M. F. Summers, *Journal of the American Chemical Society* **1986**, 108, 2093.
- [33] W. Willker, D. Leibfritz, R. Kerssebaum, W. Bermel, *Magnetic Resonance in Chemistry* **1993**, 31, 287.
- [34] J. Ruiz-Cabello, G. W. Vuister, C. T. W. Moonen, P. Van Gelderen, J. S. Cohen, P. C. M. Van Zijl, *Journal of Magnetic Resonance (1969-1992)* **1992**, 100, 282.
- [35] A. Meissner, D. Moskau, N. C. Nielsen, O. W. Soerensen, *Journal of Magnetic Resonance* **1997**, 124, 245.
- [36] C. Roumestand, C. Delay, J. A. Gavin, D. Canet, *Magnetic Resonance in Chemistry* **1999**, 37, 451.
- [37] S. Prytulla, J. Lambert, J. Lauterwein, M. Klessinger, J. Thiem, *Magnetic Resonance in Chemistry* **1990**, 28, 888.
- [38] P.-E. Jansson, L. Kenne, G. Widmalm, *Carbohydrate Research* **1987**, 168, 67.
- [39] J. E. Lemieux, *Union Med Can* **1958**, 87, 1447.
- [40] K. Bock, C. Pedersen, *Journal of the Chemical Society, Perkin Transactions 2: Physical Organic Chemistry (1972-1999)* **1974**, 293.

- [41] R. U. Lemieux, K. Bock, L. T. J. Delbaere, S. Koto, V. S. Rao, *Canadian Journal of Chemistry* **1980**, *58*, 631.
- [42] E. V. Vinogradov, B. O. Petersen, J. E. Thomas-Oates, J. Duus, H. Brade, O. Holst, *Journal of biological chemistry* **1998**, *273*, 28122.
- [43] R. A. Laine, *Glycobiology* **1994**, *4*, 759.
- [44] R. Andresson, R. Mynahan, *In Vivo: The Business and Medicine Report* **2001**, *1*.
- [45] W. S. McCulloch, W. Pitts, *Bulletin of Mathematical Biophysics* **1943**, *5*, 115.
- [46] P. M. Milner, *Sci Am* **1993**, *268*, 124.
- [47] F. Rosenblatt, *Psychol Rev* **1958**, *65*, 386.
- [48] M. Minsky, S. Papert, *MIT Press, Cambridge, MA* **1969**.
- [49] T. Kohonen, *IEEE Transactions on Computers* **1972**, *C-21*, 353.
- [50] P. Werbos, Harvard University (Harvard), **1974**.
- [51] P. Werbos, *The Roots of Backpropagation - From Ordered Derivatives to Neural Networks and Political Forecasting*, John Wiley & Sons, New York, **1994**.
- [52] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *MIT Press, Cambridge, MA* **1986**, 318.
- [53] S. Grossberg, *Biol Cybern* **1976**, *21*, 145.
- [54] S. Grossberg, *Biol Cybern* **1976**, *23*, 187.
- [55] S. Grossberg, *Biol Cybern* **1976**, *23*, 121.
- [56] J. J. Hopfield, *Proc Natl Acad Sci U S A* **1982**, *79*, 2554.
- [57] J. J. Hopfield, *Proc Natl Acad Sci U S A* **1984**, *81*, 3088.
- [58] J. J. Hopfield, D. W. Tank, *Biological Cybernetics* **1985**, *52*, 141.
- [59] J. J. Hopfield, D. W. Tank, *Science* **1986**, *233*, 625.
- [60] K. Fukushima, *Biol Cybern* **1975**, *20*, 121.
- [61] K. Fukushima, *Neural Netw* **2004**, *17*, 37.
- [62] K. Fukushima, *Biol Cybern* **2001**, *84*, 251.
- [63] K. Fukushima, *Neural Netw* **1999**, *12*, 791.
- [64] K. Fukushima, *Acta Neurochir Suppl (Wien)* **1987**, *41*, 51.
- [65] K. Fukushima, *Biol Cybern* **1986**, *55*, 5.
- [66] K. Fukushima, *Biol Cybern* **1984**, *50*, 105.
- [67] K. Fukushima, *Iyodenshi To Seitai Kogaku* **1981**, *19*, 319.
- [68] K. Fukushima, *Biol Cybern* **1980**, *36*, 193.
- [69] K. Fukushima, M. Kikuchi, *Neural Netw* **1996**, *9*, 1417.
- [70] K. Fukushima, S. Miyake, *Biol Cybern* **1978**, *28*, 201.
- [71] Y. Hirai, K. Fukushima, *Biol Cybern* **1978**, *31*, 209.
- [72] S. Miyake, K. Fukushima, *Biol Cybern* **1984**, *50*, 377.
- [73] H. Okada, E. Fukushi, S. Onodera, T. Nishimoto, J. Kawabata, M. Kikuchi, N. Shiomi, *Carbohydrate Research* **2003**, *338*, 879.
- [74] M. Okada, Y. Yamaguchi, K. Fukushima, *Neural Netw* **1997**, *10*, 971.
- [75] R. Rojas, *Neural Networks - A Systematic Introduction*, Springer-Verlag, **1996**.
- [76] A. Zell, *Simulation Neuronaler Netze*, Universität Tübingen, **1998**.
- [77] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, **1995**.
- [78] K. J. Lang, M. J. Witbrock, *Proceedings of the connectionist Models summer School* **1988**, 52.
- [79] T. Kohonen, *Biological Cybernetics* **1982**, *43*, 59.
- [80] B. Meyer, T. Hansen, D. Nute, P. Albersheim, A. Darvill, W. York, J. Sellers, *Science* **1991**, *251*, 542.
- [81] J. P. Radomski, H. van Halbeek, B. Meyer, *Nat Struct Biol* **1994**, *1*, 217.
- [82] J. U. Thomsen, B. Meyer, *Journal of Magnetic Resonance* **1989**, *84*, 212.
- [83] S. R. Amendolia, A. Doppiu, M. L. Ganadu, G. Lubinu, *Analytical Chemistry* **1998**, *70*, 1249.
- [84] S. Doubet, K. Bock, D. Smith, A. Darvill, P. Albersheim, *Trends in Biochemical Sciences* **1989**, *14*, 475.
- [85] J. A. Van Kuik, J. F. G. Vliegthart, *Trends in Glycoscience and Glycotechnology* **1991**, *3*, 229.
- [86] J. A. Van Kuik, K. Hard, J. F. G. Vliegthart, *Carbohydrate Research* **1992**, *235*, 53.
- [87] D. S. M. Bot, P. Cleij, H. A. Van 'T Klooster, H. Van Halbeek, G. A. Veldink, J. F. G. Vliegthart, *Journal of Chemometrics* **1988**, *2*, 11.
- [88] B. R. Leeflang, E. J. Faber, P. Erbel, J. F. G. Vliegthart, *Journal of Biotechnology* **2000**, *77*, 115.

- [89] P. E. Jansson, L. Kenne, G. Widmalm, *Journal of Chemical Information and Computer Sciences* **1991**, *31*, 508.
- [90] P. E. Jansson, L. Kenne, G. Widmalm, *Analytical Biochemistry* **1991**, *199*, 11.
- [91] K. Hermansson, P. E. Jansson, L. Kenne, G. Widmalm, F. Lindh, *Carbohydrate Research* **1992**, *235*, 69.
- [92] P. E. Jansson, L. Kenne, G. Widmalm, *Carbohydrate Research* **1989**, *188*, 169.
- [93] R. Stenutz, B. Erbing, G. Widmalm, P.-E. Jansson, W. Nimmich, *Carbohydrate Research* **1997**, *302*, 79.
- [94] R. Stenutz, P.-E. Jansson, G. Widmalm, *Carbohydrate Research* **1998**, *306*, 11.
- [95] K. Adelhorst, K. Bock, *Acta Chemica Scandinavica* **1992**, *46*, 1114.
- [96] K. Adelhorst, K. Bock, H. Pedersen, S. Refn, *Acta Chemica Scandinavica* **1988**, *Ser. B42*, 196.
- [97] K. Ajisaka, H. Fujimoto, *Carbohydrate Research* **1990**, *199*, 227.
- [98] F. Akiyama, R. L. Stevens, S. Hayashi, D. A. Swann, J. P. Binette, B. Caterson, K. Schmid, H. van Halbeek, J. H. Mutsaers, G. J. Gerwig, *Archives of Biochemistry and Biophysics* **1987**, *252*, 574.
- [99] Y. Akiyama, S. Eda, K. Kato, H. Tanaka, *Carbohydrate Research* **1984**, *133*, 289.
- [100] Y. Arakatsu, G. Ashwell, E. A. Kabat, *The Journal of Immunology* **1966**, *97*, 858.
- [101] L. V. Backinowsky, P. I. Abronina, A. S. Shashkov, A. A. Grachev, N. K. Kochetkov, S. A. Nepogodiev, J. F. Stoddart, *Chemistry--A European Journal* **2002**, *8*, 4412.
- [102] L. V. Backinowsky, A. R. Gomtsyan, N. E. Bairamova, N. K. Kochetkov, *Bioorganicheskaya Khimiya (Russian Journal of Bioorganic Chemistry)* **1984**, *10*, 79.
- [103] I. Backman, B. Erbing, P.-E. Jansson, L. Kenne, *Journal of the Chemical Society, Perkin Transactions 1* **1988**, *4*, 889.
- [104] N. L. Bakh, E. M. Beier, N. V. Bovin, S. E. Zurabyan, G. Y. Vidershain, *Doklady Akademii Nauk* **1980**, *255*, 996.
- [105] M. Bartelt, A. S. Shashkov, H. Kochanowski, B. Jann, K. Jann, *Carbohydrate Research* **1994**, *254*, 203.
- [106] M. Bartelt, A. S. Shashkov, H. Kochanowski, B. Jann, K. Jann, *Carbohydrate Research* **1993**, *248*, 233.
- [107] H. Baumann, B. Erbing, P.-E. Jansson, L. Kenne, *Journal of the Chemical Society, Perkin Transactions 1* **1989**, *12*, 2153.
- [108] H. Baumann, P.-E. Jansson, L. Kenne, *Journal of the Chemical Society, Perkin Transactions 1* **1991**, *9*, 2229.
- [109] H. Baumann, P.-E. Jansson, L. Kenne, *Journal of the Chemical Society, Perkin Transactions 1* **1988**, *2*, 209.
- [110] W. H. Binder, H. Kählig, W. Schmid, *Tetrahedron* **1994**, *50*, 10407.
- [111] K. Bock, J. Arnarp, J. Loengren, *European Journal of Biochemistry* **1982**, *129*, 171.
- [112] K. Bock, C. Pedersen, *Advances in Carbohydrate Chemistry and Biochemistry* **1983**, *41*, 27.
- [113] K. Bock, C. Pedersen, H. Pedersen, *Advances in Carbohydrate Chemistry and Biochemistry* **1984**, *42*, 193.
- [114] K. Bock, S. Refn, *Acta Chemica Scandinavica* **1989**, *43*, 373.
- [115] J. B. Bouwstra, J. Kerékgyártó, J. P. Kamerling, J. F. G. Vliegthart, *Carbohydrate Research* **1989**, *186*, 39.
- [116] J.-R. Brisson, F. M. Winnik, J. J. Krepinsky, J. P. Carver, *Journal of Carbohydrate Chemistry* **1983**, *2*, 41.
- [117] A. V. Bukharov, I. M. Skvortsov, V. V. Ignatov, A. S. Shashkov, Y. A. Knirel, J. Dabrowski, *Carbohydrate Research* **1993**, *241*, 309.
- [118] P. Cescutti, R. Toffanin, B. J. Kvam, S. Paoletti, G. G. Dutton, *European Journal of Biochemistry* **1993**, *213*, 445.
- [119] N. W. H. Cheetham, G. Teng, *Carbohydrate Research* **1985**, *144*, 169.
- [120] V. Chiffolleau-Giraud, P. Spangenberg, C. Rabiller, *Tetrahedron: Asymmetry* **1997**, *8*, 2017.
- [121] D. D. Cox, E. K. Metzner, L. W. Cary, E. J. Reist, *Carbohydrate Research* **1978**, *67*, 23.
- [122] S. M. T. D'Arcy, S. L. Carney, T. J. Howe, *Carbohydrate Research* **1994**, *255*, 41.
- [123] J. W. Date, *Danish Medical Bulletin* **1966**, *13*, 98.
- [124] A. N. Davies, P. Lampen, *Applied Spectroscopy* **1993**, *47*, 1093.
- [125] A. H. de Bruin, H. Parolis, L. A. S. Parolis, *Carbohydrate Research* **1992**, *235*, 199.
- [126] M.-H. Du, U. Spohr, R. U. Lemieux, *Glycoconjugate Journal* **1994**, *11*, 443.

- [127] V. K. Dua, C. A. Bush, *Analytical Biochemistry* **1983**, 133, 1.
- [128] O. R. Duda, E. P. Hart, G. D. Stork, *John Wiley & Sons, Inc.* **2001**.
- [129] J. O. Duus, N. E. Nifant'ev, A. S. Shashkov, E. A. Khatuntseva, K. Bock, *Carbohydrate Research* **1996**, 288, 25.
- [130] P. Edebrink, P.-E. Jansson, G. Widmalm, W. Nimmich, *Carbohydrate Research* **1994**, 257, 107.
- [131] H. Egge, H. von Nicolai, F. Zilliken, *FEBS Letters* **1974**, 39, 341.
- [132] P. Fernández, J. Jiménez-Barbero, *Journal of Carbohydrate Chemistry* **1994**, 13, 207.
- [133] P. Fernández, J. Jiménez-Barbero, M. Martín-Lomas, D. Solís, T. Díaz-Mauriño, *Carbohydrate Research* **1994**, 256, 223.
- [134] W. Fischer, K. Winkler, *Hoppe-Seyler's Zeitschrift für Physiologische Chemie* **1969**, 350, 1137.
- [135] L. A. Flugge, J. T. Blank, P. A. Petillo, *Journal of the American Chemical Society* **1999**, 121, 7228.
- [136] M. Forsgren, P.-E. Jansson, L. Kenne, *Journal of the Chemical Society, Perkin Transactions 1* **1985**, 2383.
- [137] T. Fujiwara, T. Takeda, Y. Ogihara, *Carbohydrate Research* **1985**, 141, 168.
- [138] J. Gasteiger, B. M. P. Hendriks, P. Hoever, C. Jochum, H. Somberg, *Applied Spectroscopy* **1991**, 45, 4.
- [139] P. Gellerfors, K. Axelsson, A. Helander, S. Johansson, L. Kenne, S. Lindqvist, B. Pavlu, A. Skottner, L. Fryklund, *Journal of Biological Chemistry* **1989**, 264, 11444.
- [140] P. A. J. Gorin, M. Mazurek, *Canadian Journal of Chemistry* **1975**, 53, 1212.
- [141] H. D. Grimmecke, Y. A. Knirel, B. Kiesel, M. Voges, E. T. Rietschel, *Carbohydrate Research* **1994**, 259, 45.
- [142] L. Grimmonprez, G. Takerkart, M. Monsigny, J. Montreuil, *Comptes Rendus Hebdomadaires des Seances de l'Academie des Sciences: Serie D* **1967**, 265, 2124.
- [143] G. Gronberg, P. Lipniunas, T. Lundgren, F. Lindh, B. Nilsson, *Archives of Biochemistry and Biophysics* **1992**, 296, 597.
- [144] M. R. Grue, H. Parolis, L. A. S. Parolis, *Carbohydrate Research* **1993**, 246, 283.
- [145] M. Gruter, B. Didier, P. de Waard, J. Kuiper, J. P. Kamerling, J. F. G. Vliegthart, *Journal of Carbohydrate Chemistry* **1994**, 13, 363.
- [146] A. Gunnarsson, B. Svensson, B. Nilsson, S. Svensson, *European Journal of Biochemistry* **1984**, 145, 463.
- [147] S. R. Haseley, L. Galbraith, S. G. Wilkinson, *Carbohydrate Research* **1994**, 258, 199.
- [148] S. R. Haseley, S. G. Wilkinson, *Carbohydrate Research* **1994**, 264, 73.
- [149] D. L. Hendrix, Y.-a. Wei, *Carbohydrate Research* **1994**, 253, 329.
- [150] O. Hindsgaul, D. P. Khare, M. Bach, R. U. Lemieux, *Canadian Journal of Chemistry* **1985**, 63, 2653.
- [151] R. E. Hoffman, J. C. Christofides, D. B. Davies, C. J. Lawson, *Carbohydrate Research* **1986**, 153, 1.
- [152] K. Ishikawa, I. Matsui, S. Kobayashi, H. Nakatani, K. Honda, *Biochemistry* **1993**, 32, 6259.
- [153] B. Jann, A. S. Shashkov, H. Kochanowski, K. Jann, *Carbohydrate Research* **1994**, 263, 217.
- [154] B. Jann, A. S. Shashkov, H. Kochanowski, K. Jann, *Carbohydrate Research* **1994**, 264, 305.
- [155] P.-E. Jansson, L. Kenne, I. Kolare, *Carbohydrate Research* **1994**, 257, 163.
- [156] P.-E. Jansson, L. Kenne, H. Ottosson, *Journal of the Chemical Society, Perkin Transactions 1* **1990**, 7, 2011.
- [157] P.-E. Jansson, L. Kenne, E. Schweda, *Journal of the Chemical Society, Perkin Transactions 1* **1988**, 10, 2729.
- [158] P.-E. Jansson, L. Kenne, E. Schweda, *Journal of the Chemical Society, Perkin Transactions 1* **1987**, 377.
- [159] P.-E. Jansson, A. Kjellberg, T. Rundlöf, G. Widmalm, *Journal of the Chemical Society, Perkin Transactions 2* **1996**, 1, 33.
- [160] P.-E. Jansson, B. Lindberg, *Carbohydrate Research* **1980**, 82, 97.
- [161] P.-E. Jansson, B. Lindberg, J. Lönnngren, C. Ortega, *Carbohydrate Research* **1984**, 131, 277.
- [162] P.-E. Jansson, G. Widmalm, *Journal of the Chemical Society, Perkin Transactions 2* **1992**, 7, 1085.

- [163] J. Kerékgyártó, Z. Szurmai, A. Lipták, *Carbohydrate Research* **1993**, *245*, 65.
- [164] V. Kéry, S. Kucár, M. Matulová, J. Haplová, *Carbohydrate Research* **1991**, *209*, 83.
- [165] E. A. Khatuntseva, A. S. Shashkov, N. E. Nifant'ev, *Magnetic Resonance in Chemistry* **1997**, *35*, 414.
- [166] L. L. Kiefer, W. S. York, P. Albersheim, A. G. Darvill, *Carbohydrate Research* **1990**, *197*, 139.
- [167] Y. A. Knirel, N. K. Kochetkov, *Biokhimiia* **1994**, *59*, 1784.
- [168] N. A. Kocharova, Y. A. Knirel, A. S. Shashkov, N. E. Nifant'ev, N. K. Kochetkov, L. D. Varbanets, N. V. Moskalenko, O. S. Brovarkaya, V. A. Muras, J. M. Young, *Carbohydrate Research* **1993**, *250*, 275.
- [169] N. A. Kocharova, A. Maszewska, G. V. Zatonsky, A. Torzewska, O. V. Bystrova, A. S. Shashkov, Y. A. Knirel, A. Rozalski, *Carbohydrate Research* **2004**, *339*, 415.
- [170] G. Kogan, G. Haraguchi, S. I. Hull, R. A. Hull, A. S. Shashkov, B. Jann, K. Jann, *European Journal of Biochemistry* **1993**, *214*, 259.
- [171] G. Kogan, A. S. Shashkov, B. Jann, K. Jann, *Carbohydrate Research* **1993**, *238*, 261.
- [172] H. Kogelberg, T. J. Rutherford, *Glycobiology* **1994**, *4*, 49.
- [173] P. Kovác, C. P. J. Glaudemans, R. B. Taylor, *Carbohydrate Research* **1985**, *142*, 158.
- [174] P. Kovác, L. Lerner, *Carbohydrate Research* **1988**, *184*, 87.
- [175] P. Kovác, E. A. Sokolowski, C. P. J. Glaudemans, *Carbohydrate Research* **1984**, *128*, 101.
- [176] P. Kovác, R. B. Taylor, *Carbohydrate Research* **1987**, *167*, 153.
- [177] Y. Kurokawa, T. Takeda, Y. Komura, Y. Ogihara, *Carbohydrate Research* **1988**, *175*, 144.
- [178] P. Lampen, H. Hillig, A. N. Davies, M. Linscheid, *Applied Spectroscopy* **1994**, *48*, 1545.
- [179] G. M. Lipkind, N. E. Nifant'ev, A. S. Shashkov, N. K. Kochetkov, *Canadian Journal of Chemistry* **1990**, *68*, 1238.
- [180] N. H. Low, D. L. Nelson, P. Sporns, *Journal of Apicultural Research* **1988**, *27*, 245.
- [181] R. Madiyalakan, M. S. Chowdhary, S. S. Rana, K. L. Matta, *Carbohydrate Research* **1986**, *152*, 183.
- [182] J. L. Magnani, B. Nilsson, M. Brockhaus, D. Zopf, Z. Steplewski, H. Koprowski, V. Ginsburg, *Journal of Biological Chemistry* **1982**, *257*, 14365.
- [183] A. Maranduba, A. Veyrieres, *Carbohydrate Research* **1986**, *151*, 105.
- [184] J.-R. Marino-Albernas, S. L. Harris, V. Varma, B. M. Pinto, *Carbohydrate Research* **1993**, *245*, 245.
- [185] R. S. McDonald, P. A. Wilks, Jr., *Applied Spectroscopy* **1988**, *42*, 151.
- [186] O. A. Nechaev, V. I. Torgov, V. N. Shibaev, *Bioorganicheskaya Khimiya (Russian Journal of Bioorganic Chemistry)* **1988**, *14*, 1224.
- [187] N. E. Nifant'ev, V. Y. Amochaeva, A. S. Shashkov, *Bioorganicheskaya Khimiya (Russian Journal of Bioorganic Chemistry)* **1992**, *18*, 562.
- [188] N. E. Nifant'ev, A. S. Shashkov, G. M. Lipkind, N. K. Kochetkov, *Carbohydrate Research* **1992**, *237*, 95.
- [189] A. Nixon Anderson, L. A. S. Parolis, H. Parolis, *Carbohydrate Research* **1994**, *265*, 41.
- [190] P. Odonmazig, A. Ebringerová, E. Machová, J. Alföldi, *Carbohydrate Research* **1994**, *252*, 317.
- [191] T. Ogawa, T. Kaburagi, *Carbohydrate Research* **1982**, *103*, 53.
- [192] T. Ogawa, K. Sasajima, *Tetrahedron* **1981**, *37*, 2787.
- [193] T. Ogawa, K. Sasajima, *Carbohydrate Research* **1981**, *97*, 205.
- [194] E. Parra, J. Jiménez-Barbero, M. Bernabe, J. A. Leal, A. Prieto, B. Gómez-Miranda, *Carbohydrate Research* **1994**, *251*, 315.
- [195] H. Paulsen, B. Sumfleth, *Chemische Berichte* **1979**, *112*, 3203.
- [196] J. H. Pazur, F. J. Miskiel, B. Liu, *Analytical Biochemistry* **1988**, *174*, 46.
- [197] M. B. Perry, L. A. Babiuk, *Canadian Journal of Biochemistry and Cell Biology* **1984**, *62*, 108.
- [198] D. E. Portlock, G. S. Lubey, B. Borah, *Journal of Organic Chemistry* **1989**, *54*, 2327.
- [199] V. Pozsgay, J.-R. Brisson, H. J. Jennings, *Canadian Journal of Chemistry* **1987**, *65*, 2764.
- [200] S. Rio, J.-M. Beau, J.-C. Jacquinet, *Carbohydrate Research* **1983**, *244*, 295.
- [201] G. W. Robijn, J. R. Thomas, H. Haas, D. J. C. van den Berg, J. P. Kamerling, J. F. G. Vliegthart, *Carbohydrate Research* **1995**, *276*, 137.
- [202] T. E. C. L. Ronnow, M. Meldal, K. Bock, *Journal of Carbohydrate Chemistry* **1995**, *14*, 197.
- [203] T. E. C. L. Ronnow, M. Meldal, K. Bock, *Tetrahedron: Asymmetry* **1995**, *5*, 2109.
- [204] D. Schwarzenbach, R. W. Jeanloz, *Carbohydrate Research* **1981**, *90*, 193.

- [205] W. B. Severn, J. C. Richards, *Carbohydrate Research* **1993**, *240*, 277.
- [206] A. S. Shashkov, N. E. Nifant'ev, V. Y. Amochaeva, N. K. Kochetkov, *Magnetic Resonance in Chemistry* **1993**, *31*, 599.
- [207] A. Shimamuro, Y. Uezono, H. Tsumori, H. Mukasa, *Carbohydrate Research* **1992**, *233*, 237.
- [208] N. Shiomi, *Journal of Plant Physiology* **1989**, *134*, 151.
- [209] P. Söderman, P.-E. Jansson, G. Widmalm, *Journal of the Chemical Society, Perkin Transactions 2* **1998**, *3*, 639.
- [210] P. Spangenberg, C. Andre, V. Langlois, M. Dion, C. Rabiller, *Carbohydrate Research* **2002**, *337*, 221.
- [211] V. K. Srivastava, S. J. Sondheimer, C. Schuerch, *Carbohydrate Research* **1980**, *86*, 203.
- [212] G. Strecker, J.-M. Wieruszkeski, M. D. Fontaine, Y. Plancke, *Glycobiology* **1994**, *4*, 605.
- [213] G. Strecker, J.-M. Wieruszkeski, Y. Plancke, B. Boilly, *Glycobiology* **1995**, *5*, 137.
- [214] T. Takeda, T. Kanemitsu, M. Ishiguro, Y. Ogihara, M. Matsubara, *Carbohydrate Research* **1994**, *256*, 59.
- [215] E. Tarelli, S. F. Wheeler, *Carbohydrate Research* **1994**, *261*, 25.
- [216] A. Temeriusz, B. Piekarska, J. Radomski, J. Stepinski, *Carbohydrate Research* **1982**, *108*, 298.
- [217] T. Uchiyama, *Biochimica et Biophysica Acta* **1975**, *397*, 153.
- [218] T. Usui, S. Morimoto, Y. Hayakawa, M. Kawaguchi, T. Murata, Y. Matahira, *Carbohydrate Research* **1996**, *285*, 29.
- [219] T. Usui, T. Murata, Y. Yabuuchi, K. Ogawa, *Carbohydrate Research* **1993**, *250*, 57.
- [220] R. W. Veh, J. C. Michalski, A. P. Corfield, M. Sander-Wewer, D. Gies, R. Schauer, *Journal of Chromatography* **1981**, *212*, 313.
- [221] E. V. Vinogradov, K. Bock, *Carbohydrate Research* **1998**, *309*, 57.
- [222] H. von Nicolai, R. Drzeniek, F. Zilliken, *Zeitschrift für Naturforschung B: Journal of Chemical Sciences* **1971**, *26*, 1049.
- [223] T. Watanabe, M. Shida, T. Murayama, Y. Furuyama, T. Nakajima, K. Matsuda, K. Kainuma, *Carbohydrate Research* **1984**, *129*, 229.
- [224] A. Weintraub, K. Leontein, G. Widmalm, P. A. Vial, M. M. Levine, A. A. Lindberg, *European Journal of Biochemistry* **1993**, *213*, 859.
- [225] D. V. Whittaker, L. A. S. Parolis, H. Parolis, *Carbohydrate Research* **1994**, *262*, 323.
- [226] A. M. Winn, L. Galbraith, G. S. Temple, S. G. Wilkinson, *Carbohydrate Research* **1993**, *247*, 249.
- [227] K. Yamashita, Y. Tachibana, A. Kobata, *Archives of Biochemistry and Biophysics* **1977**, *182*, 546.
- [228] K. Yamashita, Y. Tachibana, K. Mihara, S. Okada, H. Yabuuchi, A. Kobata, *Journal of Biological Chemistry* **1980**, *155*, 5126.
- [229] J. H. Yoon, G. H. Ryu, P. Finch, J. S. Rhee, *Journal of Molecular Catalysis B: Enzymatic* **2001**, *15*, 191.
- [230] N. M. Young, I. B. Jocius, M. A. Leon, *Biochemistry* **1971**, *10*, 3457.
- [231] T. Ziegler, H. Sutoris, C. P. J. Glaudemans, *Carbohydrate Research* **1992**, *229*, 271.
- [232] J. Zupan, J. Gasteiger, *Neural Networks in Chemistry and Drug Design - Second Edition*, Wiley-VCH Verlag, **1999**.
- [233] S. E. Zurabyan, V. A. Markin, V. V. Pimenova, B. V. Rozynov, V. L. Sadovskya, A. Y. Khorlin, *Bioorganicheskaya Khimiya (Russian Journal of Bioorganic Chemistry)* **1978**, *4*, 928.
- [234] K. Agoston, A. Dobó, J. Rákó, J. Kerékgyártó, Z. Szurmai, *Carbohydrate Research* **2001**, *330*, 183.
- [235] A. Allerhand, E. Berman, *Journal of the American Chemical Society* **1984**, *106*, 2400.
- [236] N. Asano, K. Matsui, S. Takeda, Y. Kono, *Carbohydrate Research* **1992**, *243*, 71.
- [237] I. Backman, B. Erbing, P.-E. Jansson, L. Kenne, *J. Chem. Soc., Perkin Trans. 1* **1988**, *4*, 889.
- [238] M. Baron, P. A. J. Gorin, M. Iacomini, *Carbohydrate Research* **1988**, *177*, 235.
- [239] R. W. Bassily, R. I. El-Sokkary, B. A. Silvanis, A. S. Nematalla, M. A. Nashed, *Carbohydrate Research* **1993**, *239*, 197.
- [240] G. Batta, K. E. Kövér, *Carbohydrate Research* **1999**, *320*, 267.
- [241] M. K. Bhattacharjee, R. M. Mayer, *Carbohydrate Research* **1993**, *242*, 191.
- [242] C. M. Bishop, *Oxford University Press* **1995**.

- [243] L. Chen, Y. Zhu, F. Kong, *Carbohydrate Research* **2002**, 337, 383.
- [244] J. Dahmén, T. Frejd, T. Lave, F. Lindh, G. Magnusson, G. Noori, K. Palsson, *Carbohydrate Research* **1983**, 113, 219.
- [245] J. Dahmén, T. Frejd, G. Magnusson, G. Noori, A.-S. Carlström, *Carbohydrate Research* **1984**, 127, 27.
- [246] J. Dahmén, G. Gnosspelius, A.-C. Larsson, T. Lave, G. Noori, K. Palsson, *Carbohydrate Research* **1985**, 138, 17.
- [247] I. Damager, C. E. Olsen, A. Blennow, K. Denyer, B. Lindberg Moller, M. S. Motawia, *Carbohydrate Research* **2003**, 338, 189.
- [248] S. K. Das, N. Roy, *Carbohydrate Research* **1995**, 271, 177.
- [249] M. Fajjes, J. K. Fairweather, H. Driguez, A. Planas, *Chemistry--A European Journal* **2001**, 7, 4651.
- [250] J. Fang, X. Chen, W. Zhang, A. Janczuk, P. G. Wang, *Carbohydrate Research* **2000**, 329, 873.
- [251] P. J. Garegg, H. Hultberg, *Carbohydrate Research* **1982**, 110, 261.
- [252] J. Gasteiger, **2003**.
- [253] S. J. Gebbie, I. Gosney, P. R. Harris, I. M. F. Lacan, W. R. Sanderson, J. P. Sankey, *Carbohydrate Research* **1998**, 308, 345.
- [254] P. A. J. Gorin, *Carbohydrate Research* **1982**, 101, 13.
- [255] M. L. Hayes, A. S. Serianni, R. Barker, *Carbohydrate Research* **1982**, 100, 87.
- [256] P.-E. Jansson, L. Kenne, K. Persson, G. Widmalm, *Journal of the Chemical Society, Perkin Transactions 1* **1990**, 591.
- [257] G. A. Jeffrey, R. Nanni, *Carbohydrate Research* **1985**, 137, 21.
- [258] S. H. Khan, C. F. Piskorz, K. L. Matta, *Journal of Carbohydrate Chemistry* **1994**, 13, 1025.
- [259] S. Koto, H. Haigoh, S. Shichi, M. Hirooka, T. Nakamura, C. Maru, M. Fujita, A. Goto, T. Sato, M. Okada, S. Zen, K. Yago, F. Tomonaga, *Bulletin of the Chemical Society of Japan* **1995**, 68, 2331.
- [260] L. M. J. Kroon-Batenburg, J. Kroon, B. R. Leeftang, J. F. G. Vliegthart, *Carbohydrate Research* **1993**, 245, 21.
- [261] R. U. Lemieux, U. Spohr, M. Bach, D. R. Cameron, T. P. Frandsen, B. B. Stoffer, B. Svensson, M. M. Palcic, *Canadian Journal of Chemistry* **1996**, 74, 319.
- [262] L. J. Liotta, R. Capotosto, R. A. Garbitt, B. M. Horan, P. J. Kelly, A. P. Koleros, L. M. Brouillette, A. M. Kuhn, S. Targontsidis, *Carbohydrate Research* **2001**, 331, 247.
- [263] G. M. Lipkind, A. S. Shashkov, S. S. Mamyran, N. K. Kochetkov, *Carbohydrate Research* **1988**, 181, 1.
- [264] K. Mizutani, R. Kasai, M. Nakamura, O. Tanaka, H. Matsuura, *Carbohydrate Research* **1989**, 185, 27.
- [265] C. Morat, F. R. Tavel, M. R. Vignon, *Carbohydrate Research* **1987**, 163, 265.
- [266] V. Moreau, J.-L. Viladot, E. Samain, A. Planas, H. Driguez, *Bioorganic & Medicinal Chemistry* **1996**, 4, 1849.
- [267] N. E. Nifant'ev, V. Y. Amochaeva, A. S. Shashkov, N. K. Kochetkov, *Carbohydrate Research* **1993**, 250, 211.
- [268] T. Ogawa, K. Sasajima, *Carbohydrate Research* **1981**, 93, 67.
- [269] T. Peters, *Liebigs Annalen der Chemie* **1991**, 135.
- [270] T. Rundlöf, A. Kjellberg, C. Damberg, T. Nishida, G. Widmalm, *Magnetic Resonance in Chemistry* **1998**, 36, 839.
- [271] A. S. Shashkov, G. M. Lipkind, N. K. Kochetkov, *Carbohydrate Research* **1986**, 147, 175.
- [272] S.-i. Shoda, T. Kawasaki, K. Obata, S. Kobayashi, *Carbohydrate Research* **1993**, 249, 127.
- [273] S.-i. Shoda, K. Obata, O. Karthaus, S. Kobayashi, *Journal of the Chemical Society, Chemical Communications* **1993**, 1402.
- [274] B. W. Sigurskjold, B. Duus, K. Bock, *Acta Chemica Scandinavica* **1991**, 45, 1032.
- [275] P. Spangenberg, V. Chiffoleau-Giraud, C. André, M. Dion, C. Rabiller, *Tetrahedron: Asymmetry* **1999**, 10, 2905.
- [276] B. A. Spronk, A. Rivera-Sagredo, J. P. Kamerling, J. F. G. Vliegthart, *Carbohydrate Research* **1995**, 273, 11.
- [277] StatSoft, *STATISTICA Manual* **2004**.
- [278] Z. Szurmai, J. Kerékgyártó, J. Harangi, A. Liták, *Carbohydrate Research* **1987**, 164, 313.
- [279] K. Takeo, T. Imai, *Carbohydrate Research* **1987**, 165, 123.
- [280] K. Takeo, S. Matsuzaki, *Carbohydrate Research* **1983**, 113, 281.

- [281] K. Takeo, S. Tei, *Carbohydrate Research* **1986**, *145*, 307.
- [282] A. Temeriusz, B. Piekarska, J. Radomski, J. Stepinski, *Polish Journal of Chemistry (formerly Roczniki Chemii)* **1982**, *56*, 141.
- [283] M. Upreti, D. Ruhela, R. A. Vishwakarma, *Tetrahedron* **2000**, *56*, 6577.
- [284] A. M. P. van Steijn, J. P. Kamerling, J. F. G. Vliegthart, *Carbohydrate Research* **1992**, *225*, 229.
- [285] R. Verduyn, M. Douwes, P. A. M. van der Klein, E. M. Mösinger, G. A. van der Marel, J. H. van Boom, *Tetrahedron* **1993**, *49*, 7301.
- [286] P. Wang, G.-J. Shen, Y.-F. Wang, Y. Ichikawa, C.-H. Wong, *Journal of Organic Chemistry* **1993**, *58*, 3985.
- [287] J. C. Wilson, M. J. Kiefel, S. Albouz-Abo, M. von Itzstein, *Bioorganic & Medicinal Chemistry Letters* **2000**, *10*, 2791.
- [288] A. Zell, *Universität Tübingen* **1998**.
- [289] X.-X. Zhu, P. Y. Ding, M.-S. Cai, *Tetrahedron: Asymmetry* **1996**, *7*, 2833.
- [290] T. Ziegler, P. Kovác, C. P. J. Glaudemans, *Carbohydrate Research* **1990**, *203*, 253.
- [291] J. Zupan, J. Gasteiger, *Wiley-VCH Verlag* **1999**.
- [292] H. E. Gottlieb, V. Kotlyar, A. Nudelman, *J Org Chem* **1997**, *62*, 7512.
- [293] StatSoft, Version 6 ed., StatSoft, **2004**.
- [294] J. Zupan, *Acta Chimica Slovenica* **1994**, *41*, 327.
- [295] StatSoft, Version 6 ed., StatSoft, **2004**.
- [296] A. Zell, G. Mamier, M. Vogt, *SNNS: Stuttgart Neural Network Simulator User Manual*, University of Tübingen, Stuttgart, **2002**.
- [297] A. Zell, G. Mamier, M. Vogt, N. Mache, R. Hübner, S. Döring, K.-U. Herrmann, T. Soye, M. Schmalzl, T. Sommer, A. Hatzigeorgiou, D. Posselt, T. Schreiner, B. Kett, G. Clemente, J. Wieland, J. Gatter, in *University of Tübingen, University of Stuttgart and University of Tübingen*, **2002**.
- [298] A. Zell, G. Mamier, M. Vogt, N. Mache, T. Sommer, R. Hübner, M. Schmalzl, T. Soye, S. Döring, D. Posselt, K.-U. Herrmann, A. Hatzigeorgiou, Version 1.1 ed., University of Tübingen, Tübingen, **2002**.
- [299] J. Zhang, *Proceedings of The 2002 International Joint Conference on Neural Networks, Honolulu, Hawaii, U.S.A., 12 - 17 May, 2002* **2002**, Vol.1, PP800.
- [300] S. Lawrence, C. L. Giles, A. C. Tsoi, *online PDF* **1996**.
- [301] P. L. Rosin, F. Freddy, *Proceedings of IGARSS'95, Firenze, Italy, July 1995* **1995**.
- [302] W. S. Sarle, Cary, NC, USA, **2002**.
- [303] V. Pozsgay, P. Nanasi, A. Neszmelyi, *Carbohydrate Research* **1979**, *75*, 310.
- [304] L. Massone, E. Bizzi, *Biol Cybern* **1989**, *61*, 417.
- [305] I. A. Basheer, M. Hajmeer, *Journal of microbiological methods* **2000**, *43*, 3.
- [306] D. P. Berrar, C. S. Downes, W. Dubitzky, *Pacific Symposium on Biocomputing 2003, Kauai, HI, United States, Jan. 3-7, 2003* **2003**, 5.
- [307] P. Berrar Daniel, C. S. Downes, W. Dubitzky, *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing* **2003**, 5.
- [308] K. Kang, J. H. Oh, C. Kwon, Y. Park, *Physical Review. E. Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics* **1993**, *48*, 4805.
- [309] H. M. A. Andree, W. Lourens, A. Taal, J. C. Vermeulen, *Nuclear Instruments & Methods in Physics Research, Section A: Accelerators, Spectrometers, Detectors, and Associated Equipment* **1995**, *355*, 589.
- [310] K. Kang, J. H. Oh, C. Kwon, Y. Park, *Physical Review. E. Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics* **1996**, *54*, 1811.
- [311] R. Caruana, S. Lawrence, L. Giles, *Neural Information Processing Systems, Denver, Colorado November 28-30, 2000* **2000**.
- [312] R. Setiono, *Artif Intell Med* **2000**, *18*, 205.
- [313] R. Setiono, *Neural Comput* **1997**, *9*, 205.
- [314] R. Setiono, *Neural Comput* **1997**, *9*, 185.
- [315] R. Setiono, *Artif Intell Med* **1996**, *8*, 37.
- [316] R. Setiono, *Neural Comput* **2001**, *13*, 2865.

9. Figure index

Figure 1: O-linked oligosaccharides	14
Figure 2: O-linked oligosaccharide in schematic illustration (left part) and the corresponding chemical structure (right)	14
Figure 3: N-linked oligosaccharides	15
Figure 4: N-linked oligosaccharide in schematic illustration (bottom right) and the corresponding	15
Figure 5: Processing of N-linked complex oligosaccharides (I)	16
Figure 6: Processing of N-linked complex oligosaccharides (II)	17
Figure 7: Comparison of N-glycosylation among alternate expression systems	18
Figure 8: determination of the number of involved monosaccharide units (adapted from ^[25])	22
Figure 9: α -Kdo = 3-deoxy-D-manno-octulosonic acid	22
Figure 10: α -NeuAc	23
Figure 11: determination of the constituent monosaccharides (adapted from ^[25])	23
Figure 12: determination of the anomeric configuration (adapted from ^[25])	23
Figure 13: determination of linkage and sequence (adapted from ^[25])	24
Figure 14: determination of the position of appended groups (adapted from ^[25])	24
Figure 15: microscopic image of a biological neuron and Comparison between the biological	28
Figure 16: Similarities between biological and artificial neurons (adapted from J. Zupan and J. Gasteiger)	29
Figure 17: The first (evaluation of the <i>Net</i> input) and the second step (nonlinear transformation of <i>Net</i>) taking place in the artificial neuron	30
Figure 18: Transfer functions	31
Figure 19: Full-connected feed forward sample network with one hidden layer	32
Figure 20: Sample network topologies for feed forward and feedback networks	33
Figure 21: A sample Kohonen feature map (Euclidian distance map) made out of all monosaccharide units used in this thesis	35
Figure 22: Main input layout of the ¹³ C-NMR FileMaker database	50
Figure 23: Oligosaccharide description scheme	51
Figure 24: A fictive nomenclature example	51
Figure 25: Two nomenclature examples for quick names	52
Figure 26: JCAMP-DX file structure overview	53
Figure 27: Graphical JCAMP-DX file x-y chart illustration of a sample ¹ H-NMR peak	58
Figure 28: Schematic presentation of weight correction (Adapted from J. Zupan and J. Gasteiger ^[232])	59
Figure 29: 3D error surface of a neural network as a function of weights w_1 and w_2 (adapted from ^[76])	60
Figure 30: Problems of gradient methods - ① they only find local minima, ② they get stuck on flat plateaus, ③ oscillation in narrow ravines and ④ they leave good minima.	61
Figure 31: An illustration of a sample Kohonen feature map (49 neurons forming a 7x7x6 network) represented as a block containing neurons as columns and weights (line intersections) in levels (adapted from ^[294]).	65
Figure 32: Illustration of square neighborhoods (adapted from J. Zupan and J. Gasteiger)	66
Figure 33: Sample Kohonen topological map (Euclidian distance between classes) trained with 30 different glucose monosaccharide classes after 20'000 learning cycles.	67
Figure 34: Winning neurons for each class after 10'000 learning cycles of the same network	67
Figure 35: Graphical location of four different glucose monosaccharide residues (top left: α -D-Glcp-OMe-2R, top right: α -D-Glcp-OMe-3R, bottom left: α -D-Glcp-OMe-4R and bottom right: α -D-Glcp-OMe-6R). Light green areas indicate high, darker green regions indicate weaker similarity.	68
Figure 36: Fully connected sample Counter-propagation network in SNNS 3D illustration. Input Units on top, Kohonen layer in the middle and the Grossberg layer at the bottom.	69
Figure 37: An illustration of a sample Counter-propagation network. On top the Kohonen layer and at the bottom the Grossberg ^[53-55] layer (adapted from ^[294]).	70
Figure 38: Statsoft Statistica main working area	72
Figure 39: SNNS V.4.2 working area	73
Figure 40: SNNS components: simulator kernel, graphical user interface xgui, batchman and network compiler snns2c (adapted from ^[296])	74
Figure 41: JavaNNS V.1.1 main window	75
Figure 42: ANN PFG - coarse data flow	76
Figure 43: Illustration of data reduction in PFG V.0.1 and partly from V.0.2	77
Figure 44: Input data flow illustration; how ¹ H-NMR data enters the neural network	78
Figure 45: Sample input CSV file for ANN PFG	80
Figure 46: Example Statistica output pattern file in CSV format	80
Figure 47: SNNS sample input pattern file	81
Figure 48: First generation SNNS-PFG GUI	82
Figure 49: Noise threshold [%] and max / min intensity parameter by the example of a disaccharide	83
Figure 50: JCAMP-DX file generator V.0.2 GUI	84
Figure 51: Detail shape view of simulated NMR peaks (left graph: ¹ H-NMR, right graph: ¹³ C-NMR). The red circle marks the original data point in the input peaklist (now the symmetrical center of the peak).	84
Figure 52: Sample file content of α -D-Manp-1R.csv	84
Figure 53: SNNS PFG V.0.2 main subprogram GUI	85
Figure 54: Explanation of the variable step size approach – only the peaks marked in red are processed and taken over into the final output pattern file. Peaks colored in pink will not be processed.	86
Figure 55: A sample cutout of the <i>output.csv</i> file	87
Figure 56: Formation path of a sample block-pattern with step size = 8	88
Figure 57: ANN PFG V 0.9.40 variables explanation	89
Figure 58: Sample input CSV file for ANN PFG V.0.9	90
Figure 59: ANN PFG GUI - input and Pre-Run options tab	91
Figure 60: Formation of the binary peak mask	91
Figure 61: Preview of overlaid NMR spectra (blue) and calculated peak mask (red)	92

Figure 62: Peak mask equalizing (4 points)	92
Figure 63: Peak mask with tolerance = ± 2	93
Figure 64: ANN PFG GUI - processing options tab	93
Figure 65: A sample binary output-coding matrix for 12 different compounds for SNNs pattern files	94
Figure 66: ANN PFG GUI - peak combination generator tab	94
Figure 67: Combinations example with 12 peaks / 6 peaks per group and α/β confusion	95
Figure 68: Test results overview of 40 neural networks tested with 924 peak combinations	96
Figure 69: ANN PFG V.0.9 UML sequence diagram	97
Figure 70: Influence of substituents at different ring positions for α -D-Glcp-OMe	98
Figure 71: Influence of substituents at different ring positions for β -D-Glcp-OMe	99
Figure 72: Influence of substituents at different ring positions for α -D-Glcp	99
Figure 73: Influence of substituents at different ring positions for β -D-Glcp	100
Figure 74: Influence of substituents at different ring positions for α -D-Manp	100
Figure 75: Influence of substituents at different ring positions for β -D-Manp	101
Figure 76: Influence of substituents at different ring positions for α -D-Manp-OMe	101
Figure 77: Influence of substituents at different ring positions for α -D-Galp	102
Figure 78: Influence of substituents at different ring positions for β -D-Galp	103
Figure 79: Influence of substituents at different ring positions for α -D-Galp-OMe	103
Figure 80: Influence of substituents at different ring positions for β -D-Galp-OMe	104
Figure 81: Learning rate comparison	107
Figure 82: Hidden layer size comparison	108
Figure 83: Act Logistic activation function	109
Figure 84: MSE and classification comparison for target output values 0.75 and 0.25	110
Figure 85: Combined MSE and classification comparison - patterns without methyl-peak	111
Figure 86: Classification comparison of networks without methyl peaks at 3.4ppm	112
Figure 87: Different Backprop learning algorithms overview	115
Figure 88: Learning rate comparison with a logistic transfer function and a fixed hidden layer size of 100 neurons	116
Figure 89: Learning rate comparison with a logistic transfer function and fixed hidden layer size of 600 neurons	117
Figure 90: Hidden layer size comparison with additional noise and block pattern	118
Figure 91: Hidden layer size comparison with a logistic transfer function, without noise and block pattern	120
Figure 92: Weight init values comparison at a learning rate 0.4, sinus transfer function, without noise and fixed hidden layer size of 100 neurons	121
Figure 93: Weight init values comparison with Backprop Momentum at a learning rate 0.4, logistic transfer function, without noise and fixed hidden layer size of 100 neurons	122
Figure 94: Comparison of different hidden layer sizes with Backprop Momentum at a fixed learning rate 0.2, logistic transfer function and without noise.	123
Figure 95: Comparison of different hidden layer sizes with Backprop Momentum at a fixed learning rate 0.7, StepFunc transfer function and binary patterns.	125
Figure 96: Comparison of different learning rates with Backprop Momentum without hidden layer, StepFunc transfer function and binary patterns.	127
Figure 97: Data distribution of all groups contained in the data set	129
Figure 98: Data set carbohydrate distribution	129
Figure 99: Graphical results overview	132
Figure 100: Proposed counter propagation networks decision tree for automated	132
Figure 101: Sample decay curve	134
Figure 102: 4-deoxy-b-D-Galp-OMe-6R	135
Figure 103: α -D-Galp-1R	135
Figure 104: α -D-GalpA-1R	135
Figure 105: α -D-GalpNAc-1R	136
Figure 106: α -D-GalpNAc-OH-6R	136
Figure 107: α -D-GalpNAc-OMe-3R	136
Figure 108: α -D-GalpNAc-OMe	136
Figure 109: α -D-Galp-OH	136
Figure 110: α -D-Galp-OH-3R	136
Figure 111: α -D-Galp-OH-4R	136
Figure 112: α -D-Galp-OH-6R	136
Figure 113: α -D-Galp-OMe	136
Figure 114: α -D-Galp-OMe-2R	136
Figure 115: α -D-Galp-OMe-3R	136
Figure 116: α -D-Galp-OMe-4R	136
Figure 117: α -D-Galp-OMe-6R	137
Figure 118: β -D-Galp-1R	137
Figure 119: β -D-GalpNAc-1R	137
Figure 120: β -D-Galp-OH	137
Figure 121: β -D-Galp-OH-3R	137
Figure 122: β -D-Galp-OH-4R	137
Figure 123: β -D-Galp-OH-6R	137
Figure 124: β -D-Galp-OMe	137
Figure 125: β -D-Galp-OMe-2R	137
Figure 126: β -D-Galp-OMe-3R	137
Figure 127: β -D-Galp-OMe-4R	137
Figure 128: β -D-Galp-OMe-6R	137
Figure 129: Euclidian distance map	138
Figure 130: α -D-Glcp-1R	140

Figure 131: α -D-GlcpN-1R	140
Figure 132: α -D-Glcp-OH	140
Figure 133: α -D-Glcp-OH-2R	140
Figure 134: α -D-Glcp-OH-3R	140
Figure 135: α -D-Glcp-OH-4R	140
Figure 136: α -D-Glcp-OH-6R	140
Figure 137: α -D-Glcp-OMe	140
Figure 138: α -D-Glcp-OMe-2R	140
Figure 139: α -D-Glcp-OMe-3R	140
Figure 140: α -D-Glcp-OMe-4R	140
Figure 141: α -D-Glcp-OMe-6R	140
Figure 142: β -D-Glcp-1R	141
Figure 143: β -D-GlcpN-1R	141
Figure 144: β -D-GlcpNAc-1R	141
Figure 145: β -D-GlcpNAc-OH-4R	141
Figure 146: β -D-GlcpNAc-OMe-3R	141
Figure 147: β -D-GlcpNAc-OMe-4R	141
Figure 148: β -D-Glcp-OH	141
Figure 149: β -D-Glcp-OH-2R	141
Figure 150: β -D-Glcp-OH-3R	141
Figure 151: β -D-Glcp-OH-4R	141
Figure 152: β -D-Glcp-OH-6R	141
Figure 153: β -D-Glcp-OMe	141
Figure 154: β -D-Glcp-OMe-2R	142
Figure 155: β -D-Glcp-OMe-3R	142
Figure 156: β -D-Glcp-OMe-4R	142
Figure 157: β -D-Glcp-OMe-6R	142
Figure 158: Euclidian distance map	142
Figure 159: Deconvolution of the glucose Kohonen feature map	145
Figure 160: α -D-Manp-1R	146
Figure 161: α -D-ManpNAc-1R	146
Figure 162: α -D-Manp-OH	146
Figure 163: α -D-Manp-OH-2R	147
Figure 164: α -D-Manp-OH-4R	147
Figure 165: α -D-Manp-OH-6R	147
Figure 166: α -D-Manp-OMe	147
Figure 167: α -D-Manp-OMe-2R	147
Figure 168: α -D-Manp-OMe-3R	147
Figure 169: α -D-Manp-OMe-4R	147
Figure 170: α -D-Manp-OMe-6R	147
Figure 171: β -D-Manp-1R	147
Figure 172: β -D-ManpNAc-1R	147
Figure 173: β -D-Manp-OH	147
Figure 174: β -D-Manp-OH-2R	147
Figure 175: β -D-Manp-OH-4R	147
Figure 176: β -D-Manp-OH-6R	147
Figure 177: β -D-Manp-OMe	147
Figure 178: β -D-Manp-OMe-2R	148
Figure 179: β -D-Manp-OMe-4R	148
Figure 180: Euclidian distance map	148
Figure 181: Euclidian distance map of the GAM Kohonen network (25 x 25 neurons) after	151
Figure 182: Coarse workflow of the Statistica approach	152
Figure 183: General pattern file structure	154
Figure 184: Average Error and performance values for different numbers of modifications (gal_mini3_sh05_modxxx)	160
Figure 185: Average selection performance of different numbers of modifications	161
Figure 186: Average training time (1 – 200 hidden units) for a Back-propagation neural network	161
Figure 187: Lowest error and best performance values for different learning rates (gal_mini2_sh05_mod40 10hu)	162
Figure 188: Learning rate overview	163
Figure 189: Momentum term comparison	164
Figure 190: Performance and error comparison of different noise levels	165
Figure 191: Selection performance of different pattern step sizes (gal_mini4_sh01_mod80)	166
Figure 192: Selection performance of different pattern step sizes (gal_mini4_sh05_mod80)	166
Figure 193: Glucose performance and error visualization chart	168
Figure 194: Galactose performance and error visualization chart	170
Figure 195: Mannose performance and error visualization chart	172
Figure 196: Number of activating patterns per input neuron	175
Figure 197: Graphical IPS performance and error comparison for glucose ensemble networks with one hidden layer	177
Figure 198: Graphical IPS performance and error comparison for glucose ensemble networks with two hidden layers	178
Figure 199: Graphical IPS performance and error comparison for galactose ensemble networks with one hidden layer	180
Figure 200: Graphical IPS performance and error comparison for galactose ensemble networks with two hidden layers	182
Figure 201: Graphical IPS performance and error comparison for mannose ensemble networks with one hidden layer	184
Figure 202: Graphical IPS performance and error comparison for mannose ensemble networks with two hidden layers	185

10. Acknowledgements

My sincere thanks go to ...

- *Prof. Beat Ernst* for giving me the chance to carry out my Ph.D. thesis in his wonderful and multinational group at the Institute of Molecular Pharmacy and the opportunity to work on such an exciting and fascinating project. I am also grateful he gave me the freedom to develop my own ideas and for supporting me in each way he could in their realization. I learned so much during these years.
- *Andreas Stöckli* who joined my one-man team as a diploma student several years ago and now became one of my best friends. Thank you for your support, your ideas, your jokes and the many fruitful and sometimes needless discussions we had all over the years. Our nighttime working and gaming sessions, the many beers and our famous coffee breaks. I will miss his moods and, the countless genial and sometimes absolutely useless new ideas we had together. He was with me whenever we faced a new abyss of our project. And we both climbed up the hill again! And we did it. Thank you.
- *Daniel Ricklin* for the endless scientific and fruitful discussions.
- *Zorica Dragic* who was really always there when I had serious personal problems. I'll miss her good mood and our laughs right on the battlefield when we both saw no light at the end of the tunnel. We fought hard all these years.
- *Bea Wagner* for her endurance with an NMR beginner like me.
- *Regula Stingelin* for the precious test oligosaccharides she synthesized for us (chapter 4.1.4).
- *Brian Cutting* for all the NMR experiments he did for us and his proof-reading of my thesis.
- *Prof. Ole Hindsgaul* from the University of Alberta for his valuable test compounds he sent us (chapter 4.1.2)
- *Alexej Moor* for his excellent diploma work.
- The present and past members of our group for the great atmosphere and the wonderful time we had over the past four years.
- My parents and my new family in Wallis for their support.
- *Nicola Van der Linden* for all her ChemDraw structures and the valuable EndNote library with all our literature compounds.
- My dear friends all over the world who tried to keep patient in the case I missed another appointment because important experiments and programming wanted to be done during nighttime.
- And the greatest thanks go to *Regula* my beloved wife. Without here endless support I would have never gotten through all this. I love you!

11. Appendix

11.1. Peak lists of disaccharide test compounds

11.1.1. Trehalose

DU=/z, USER=Matthias, NAME=Mar21-2003, EXPNO=10, PROCNO=1
 F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY		INTENSITY
		[Hz]	[PPM]	
1	18537.1	11764.879	93.5519	18.10
2	21250.5	9160.900	72.8456	11.69
3	21298.3	9115.061	72.4811	16.76
4	21443.8	8975.376	71.3704	13.34
5	21620.2	8806.070	70.0241	13.56
6	22822.7	7652.064	60.8477	11.58
7	26789.8	3844.975	30.5745	4.33

11.1.2. Gentiobiose

DU=/z, USER=Matthias, NAME=Mar21-2003, EXPNO=20, PROCNO=1
 F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY		INTENSITY
		[Hz]	[PPM]	
1	17295.7	12956.201	103.0251	14.55
2	18175.3	12112.034	96.3124	11.61
3	20803.3	9590.095	76.2585	15.58
4	20841.1	9553.790	75.9698	18.10
5	20939.9	9458.955	75.2157	14.34
6	21052.3	9351.135	74.3583	8.12
7	21175.6	9232.776	73.4172	13.77
8	21630.1	8796.618	69.9489	13.13
9	21649.3	8778.212	69.8026	10.88
10	21736.5	8694.454	69.1365	10.96
11	22795.9	7677.824	61.0525	10.16
12	26789.2	3845.588	30.5793	10.04

11.1.3. Lactose

DU=/z, USER=Matthias, NAME=Mar21-2003, EXPNO=30, PROCNO=1
 F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY		INTENSITY
		[Hz]	[PPM]	
1	17267.2	12983.592	103.2429	15.02
2	18719.0	11590.327	92.1639	10.51
3	20475.3	9904.800	78.7610	12.21
4	20875.9	9520.402	75.7043	18.10
5	21247.1	9164.148	72.8714	16.66
6	21392.1	9024.991	71.7649	12.52
7	21427.2	8991.291	71.4969	12.94
8	21451.0	8968.426	71.3151	15.16
9	21564.0	8859.996	70.4529	14.87
10	21765.4	8666.732	68.9161	12.27
11	22750.5	7721.332	61.3985	13.08
12	22895.9	7581.831	60.2892	10.52
13	26789.9	3844.835	30.5733	10.41

11.1.4. Saccharose

DU=/z, USER=Matthias, NAME=Mar21-2003, EXPNO=40, PROCNO=1
F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY [Hz]	INTENSITY [PPM]
1	17167.8	13078.979	104.0014
2	18675.1	11632.410	92.4986
3	20091.1	10273.515	81.6929
4	20745.8	9645.195	76.6966
5	21060.7	9343.015	74.2938
6	21245.8	9165.380	72.8812
7	21267.2	9144.867	72.7181
8	21441.3	8977.755	71.3893
9	21685.2	8743.742	69.5285
10	22582.1	7882.967	62.6838
11	22718.9	7751.672	61.6397
12	22879.1	7597.959	60.4174
13	26789.9	3844.867	30.5736

11.2. Regula Stingelin compounds

11.2.1. β -D-pGlc-OMe

DU=/z, USER=Matthias, NAME=Mar20-2003, EXPNO=10, PROCNO=1
F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY [Hz]	INTENSITY [PPM]
1	17278.8	13078.979	103.9211
2	18851.1	9636.936	76.5718
3	19685.1	9635.589	76.5611
4	20472.8	9286.405	73.7866
5	20773.7	8870.731	70.4838
6	21394.8	7754.071	61.6112
7	21984.2	7259.071	57.6781
8	26789.9	3844.867	30.5736

11.2.2. β -D-pGlc-1-6- β -D-pGlc-OMe

DU=/z, USER=Matthias, NAME=Mar20-2003, EXPNO=20, PROCNO=1
 F1=230.000ppm, F2=-10.000ppm, MI=0.00cm, MAXI=10000.00cm, PC=1.400

#	ADDRESS	FREQUENCY		INTENSITY
		[Hz]	[PPM]	
1	17167.8	13114.467	104.2914	18.01
2	18675.1	13052.674	103.7986	15.72
3	20091.1	9681.857	76.9929	14.78
4	20745.8	9681.064	76.9866	13.14
5	21060.7	9543.645	75.8938	10.78
6	21245.8	9279.243	73.7912	13.87
7	21267.2	9217.236	73.2981	14.61
8	21441.3	8928.161	70.9993	13.41
9	21685.2	8902.911	70.7985	16.02
10	22582.1	7833.445	62.2938	13.44
11	22718.9	7368.912	58.5997	12.89
12	26789.9	3844.867	30.5736	9.58

11.3. Monosaccharide test files

FM Ref. = Internal FileMaker ¹³C-NMR database record number

11.3.1. Glucose

Table 54: Detailed composition of the glucose monosaccharide test file

FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety
4	a-D-Glcp-1R	163	a-D-Glcp-OH-6R	798	b-D-Glcp-1R
19	a-D-Glcp-1R	165	a-D-Glcp-OH-6R	813	b-D-Glcp-1R
23	a-D-Glcp-1R	167	a-D-Glcp-OH-6R	814	b-D-Glcp-1R
24	a-D-Glcp-1R	173	a-D-Glcp-OH-6R	831	b-D-Glcp-1R
42	a-D-Glcp-1R	177	a-D-Glcp-OH-6R	831	b-D-Glcp-1R
43	a-D-Glcp-1R	179	a-D-Glcp-OH-6R	873	b-D-Glcp-1R
46	a-D-Glcp-1R	187	a-D-Glcp-OH-6R	874	b-D-Glcp-1R
51	a-D-Glcp-1R	795	a-D-Glcp-OH-6R	879	b-D-Glcp-1R
52	a-D-Glcp-1R	797	a-D-Glcp-OH-6R	880	b-D-Glcp-1R
64	a-D-Glcp-1R	868	a-D-Glcp-OH-6R	881	b-D-Glcp-1R
65	a-D-Glcp-1R	870	a-D-Glcp-OH-6R	883	b-D-Glcp-1R
68	a-D-Glcp-1R	874	a-D-Glcp-OH-6R	908	b-D-Glcp-1R
70	a-D-Glcp-1R	880	a-D-Glcp-OH-6R	909	b-D-Glcp-1R
77	a-D-Glcp-1R	905	a-D-Glcp-OH-6R	263	b-D-Glcp-OH
78	a-D-Glcp-1R	265	a-D-Glcp-OMe	272	b-D-Glcp-OH
79	a-D-Glcp-1R	275	a-D-Glcp-OMe	279	b-D-Glcp-OH
80	a-D-Glcp-1R	280	a-D-Glcp-OMe	284	b-D-Glcp-OH
81	a-D-Glcp-1R	300	a-D-Glcp-OMe	406	b-D-Glcp-OH
96	a-D-Glcp-1R	397	a-D-Glcp-OMe	416	b-D-Glcp-OH
115	a-D-Glcp-1R	417	a-D-Glcp-OMe	432	b-D-Glcp-OH
120	a-D-Glcp-1R	433	a-D-Glcp-OMe	438	b-D-Glcp-OH
123	a-D-Glcp-1R	443	a-D-Glcp-OMe	470	b-D-Glcp-OH
124	a-D-Glcp-1R	444	a-D-Glcp-OMe	75	b-D-Glcp-OH-2R
152	a-D-Glcp-1R	11	a-D-Glcp-OMe-2R	83	b-D-Glcp-OH-2R
155	a-D-Glcp-1R	15	a-D-Glcp-OMe-2R	152	b-D-Glcp-OH-2R
156	a-D-Glcp-1R	708	a-D-Glcp-OMe-2R	154	b-D-Glcp-OH-2R
159	a-D-Glcp-1R	13	a-D-Glcp-OMe-3R	788	b-D-Glcp-OH-2R
160	a-D-Glcp-1R	17	a-D-Glcp-OMe-3R	790	b-D-Glcp-OH-2R
163	a-D-Glcp-1R	8	a-D-Glcp-OMe-4R	77	b-D-Glcp-OH-3R

164	a-D-Glcp-1R	19	a-D-Glcp-OMe-4R	85	b-D-Glcp-OH-3R
167	a-D-Glcp-1R	21	a-D-Glcp-OMe-4R	156	b-D-Glcp-OH-3R
168	a-D-Glcp-1R	24	a-D-Glcp-OMe-6R	158	b-D-Glcp-OH-3R
175	a-D-Glcp-1R	28	a-D-Glcp-OMe-6R	184	b-D-Glcp-OH-3R
176	a-D-Glcp-1R	67	a-D-Glcp-OMe-6R	792	b-D-Glcp-OH-3R
177	a-D-Glcp-1R	68	a-D-Glcp-OMe-6R	794	b-D-Glcp-OH-3R
178	a-D-Glcp-1R	69	a-D-Glcp-OMe-6R	877	b-D-Glcp-OH-3R
189	a-D-Glcp-1R	70	a-D-Glcp-OMe-6R	79	b-D-Glcp-OH-4R
190	a-D-Glcp-1R	113	a-D-Glcp-OMe-6R	87	b-D-Glcp-OH-4R
202	a-D-Glcp-1R	4	b-D-Glcp-1R	160	b-D-Glcp-OH-4R
203	a-D-Glcp-1R	27	b-D-Glcp-1R	162	b-D-Glcp-OH-4R
411	a-D-Glcp-1R	28	b-D-Glcp-1R	170	b-D-Glcp-OH-4R
413	a-D-Glcp-1R	29	b-D-Glcp-1R	172	b-D-Glcp-OH-4R
581	a-D-Glcp-1R	31	b-D-Glcp-1R	176	b-D-Glcp-OH-4R
582	a-D-Glcp-1R	31	b-D-Glcp-1R	182	b-D-Glcp-OH-4R
706	a-D-Glcp-1R	32	b-D-Glcp-1R	186	b-D-Glcp-OH-4R
746	a-D-Glcp-1R	32	b-D-Glcp-1R	190	b-D-Glcp-OH-4R
777	a-D-Glcp-1R	33	b-D-Glcp-1R	192	b-D-Glcp-OH-4R
785	a-D-Glcp-1R	42	b-D-Glcp-1R	194	b-D-Glcp-OH-4R
788	a-D-Glcp-1R	62	b-D-Glcp-1R	773	b-D-Glcp-OH-4R
792	a-D-Glcp-1R	63	b-D-Glcp-1R	777	b-D-Glcp-OH-4R
796	a-D-Glcp-1R	82	b-D-Glcp-1R	779	b-D-Glcp-OH-4R
851	a-D-Glcp-1R	83	b-D-Glcp-1R	871	b-D-Glcp-OH-4R
852	a-D-Glcp-1R	84	b-D-Glcp-1R	875	b-D-Glcp-OH-4R
859	a-D-Glcp-1R	85	b-D-Glcp-1R	881	b-D-Glcp-OH-4R
868	a-D-Glcp-1R	86	b-D-Glcp-1R	883	b-D-Glcp-OH-4R
869	a-D-Glcp-1R	87	b-D-Glcp-1R	885	b-D-Glcp-OH-4R
870	a-D-Glcp-1R	88	b-D-Glcp-1R	910	b-D-Glcp-OH-4R
871	a-D-Glcp-1R	89	b-D-Glcp-1R	915	b-D-Glcp-OH-4R
872	a-D-Glcp-1R	100	b-D-Glcp-1R	923	b-D-Glcp-OH-4R
907	a-D-Glcp-1R	100	b-D-Glcp-1R	928	b-D-Glcp-OH-4R
932	a-D-Glcp-1R	107	b-D-Glcp-1R	81	b-D-Glcp-OH-6R
934	a-D-Glcp-1R	108	b-D-Glcp-1R	89	b-D-Glcp-OH-6R
935	a-D-Glcp-1R	110	b-D-Glcp-1R	164	b-D-Glcp-OH-6R
955	a-D-Glcp-1R	117	b-D-Glcp-1R	166	b-D-Glcp-OH-6R
262	a-D-Glcp-OH	118	b-D-Glcp-1R	168	b-D-Glcp-OH-6R
271	a-D-Glcp-OH	121	b-D-Glcp-1R	174	b-D-Glcp-OH-6R
278	a-D-Glcp-OH	122	b-D-Glcp-1R	178	b-D-Glcp-OH-6R
405	a-D-Glcp-OH	124	b-D-Glcp-1R	180	b-D-Glcp-OH-6R
415	a-D-Glcp-OH	125	b-D-Glcp-1R	188	b-D-Glcp-OH-6R
437	a-D-Glcp-OH	125	b-D-Glcp-1R	796	b-D-Glcp-OH-6R
464	a-D-Glcp-OH	153	b-D-Glcp-1R	798	b-D-Glcp-OH-6R
74	a-D-Glcp-OH-2R	154	b-D-Glcp-1R	867	b-D-Glcp-OH-6R
82	a-D-Glcp-OH-2R	157	b-D-Glcp-1R	869	b-D-Glcp-OH-6R
151	a-D-Glcp-OH-2R	158	b-D-Glcp-1R	873	b-D-Glcp-OH-6R
153	a-D-Glcp-OH-2R	161	b-D-Glcp-1R	879	b-D-Glcp-OH-6R
787	a-D-Glcp-OH-2R	162	b-D-Glcp-1R	904	b-D-Glcp-OH-6R
789	a-D-Glcp-OH-2R	165	b-D-Glcp-1R	266	b-D-Glcp-OMe
76	a-D-Glcp-OH-3R	166	b-D-Glcp-1R	276	b-D-Glcp-OMe
84	a-D-Glcp-OH-3R	171	b-D-Glcp-1R	281	b-D-Glcp-OMe
155	a-D-Glcp-OH-3R	172	b-D-Glcp-1R	398	b-D-Glcp-OMe
157	a-D-Glcp-OH-3R	179	b-D-Glcp-1R	471	b-D-Glcp-OMe
183	a-D-Glcp-OH-3R	180	b-D-Glcp-1R	12	b-D-Glcp-OMe-2R
791	a-D-Glcp-OH-3R	185	b-D-Glcp-1R	16	b-D-Glcp-OMe-2R
793	a-D-Glcp-OH-3R	186	b-D-Glcp-1R	7	b-D-Glcp-OMe-3R
878	a-D-Glcp-OH-3R	188	b-D-Glcp-1R	14	b-D-Glcp-OMe-3R

78	a-D-Glcp-OH-4R	191	b-D-Glcp-1R	18	b-D-Glcp-OMe-3R
86	a-D-Glcp-OH-4R	192	b-D-Glcp-1R	44	b-D-Glcp-OMe-3R
159	a-D-Glcp-OH-4R	193	b-D-Glcp-1R	20	b-D-Glcp-OMe-4R
161	a-D-Glcp-OH-4R	194	b-D-Glcp-1R	22	b-D-Glcp-OMe-4R
169	a-D-Glcp-OH-4R	200	b-D-Glcp-1R	45	b-D-Glcp-OMe-4R
171	a-D-Glcp-OH-4R	218	b-D-Glcp-1R	47	b-D-Glcp-OMe-4R
175	a-D-Glcp-OH-4R	412	b-D-Glcp-1R	48	b-D-Glcp-OMe-4R
181	a-D-Glcp-OH-4R	413	b-D-Glcp-1R	50	b-D-Glcp-OMe-4R
185	a-D-Glcp-OH-4R	414	b-D-Glcp-1R	64	b-D-Glcp-OMe-4R
189	a-D-Glcp-OH-4R	414	b-D-Glcp-1R	65	b-D-Glcp-OMe-4R
191	a-D-Glcp-OH-4R	429	b-D-Glcp-1R	66	b-D-Glcp-OMe-4R
193	a-D-Glcp-OH-4R	429	b-D-Glcp-1R	90	b-D-Glcp-OMe-4R
774	a-D-Glcp-OH-4R	430	b-D-Glcp-1R	195	b-D-Glcp-OMe-4R
776	a-D-Glcp-OH-4R	430	b-D-Glcp-1R	246	b-D-Glcp-OMe-4R
778	a-D-Glcp-OH-4R	431	b-D-Glcp-1R	247	b-D-Glcp-OMe-4R
872	a-D-Glcp-OH-4R	591	b-D-Glcp-1R	249	b-D-Glcp-OMe-4R
876	a-D-Glcp-OH-4R	599	b-D-Glcp-1R	815	b-D-Glcp-OMe-4R
882	a-D-Glcp-OH-4R	677	b-D-Glcp-1R	832	b-D-Glcp-OMe-4R
886	a-D-Glcp-OH-4R	773	b-D-Glcp-1R	9	b-D-Glcp-OMe-6R
911	a-D-Glcp-OH-4R	774	b-D-Glcp-1R	30	b-D-Glcp-OMe-6R
916	a-D-Glcp-OH-4R	786	b-D-Glcp-1R	35	b-D-Glcp-OMe-6R
929	a-D-Glcp-OH-4R	786	b-D-Glcp-1R	813	b-D-Glcp-OMe-6R
80	a-D-Glcp-OH-6R	790	b-D-Glcp-1R		
88	a-D-Glcp-OH-6R	794	b-D-Glcp-1R		

11.3.2. Galactose

Table 55: Detailed composition of the galactose monosaccharide test file

FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety
1	a-D-Galp-1R	630	a-D-Galp-OMe-3R	767	b-D-Galp-1R
39	a-D-Galp-1R	814	a-D-Galp-OMe-3R	768	b-D-Galp-1R
40	a-D-Galp-1R	120	a-D-Galp-OMe-4R	778	b-D-Galp-1R
41	a-D-Galp-1R	122	a-D-Galp-OMe-4R	779	b-D-Galp-1R
49	a-D-Galp-1R	816	a-D-Galp-OMe-4R	806	b-D-Galp-1R
98	a-D-Galp-1R	23	a-D-Galp-OMe-6R	808	b-D-Galp-1R
603	a-D-Galp-1R	27	a-D-Galp-OMe-6R	815	b-D-Galp-1R
662	a-D-Galp-1R	37	a-D-Galp-OMe-6R	817	b-D-Galp-1R
689	a-D-Galp-1R	41	a-D-Galp-OMe-6R	903	b-D-Galp-1R
721	a-D-Galp-1R	55	a-D-Galp-OMe-6R	928	b-D-Galp-1R
724	a-D-Galp-1R	427	a-D-Galp-OMe-6R	929	b-D-Galp-1R
763	a-D-Galp-1R	428	a-D-Galp-OMe-6R	939	b-D-Galp-1R
804	a-D-Galp-1R	6	b-D-Galp-1R	940	b-D-Galp-1R
807	a-D-Galp-1R	7	b-D-Galp-1R	941	b-D-Galp-1R
816	a-D-Galp-1R	8	b-D-Galp-1R	961	b-D-Galp-1R
832	a-D-Galp-1R	9	b-D-Galp-1R	962	b-D-Galp-1R
887	a-D-Galp-1R	10	b-D-Galp-1R	964	b-D-Galp-1R
904	a-D-Galp-1R	34	b-D-Galp-1R	965	b-D-Galp-1R
905	a-D-Galp-1R	35	b-D-Galp-1R	968	b-D-Galp-1R
917	a-D-Galp-1R	37	b-D-Galp-1R	969	b-D-Galp-1R
918	a-D-Galp-1R	44	b-D-Galp-1R	1021	b-D-Galp-1R
919	a-D-Galp-1R	45	b-D-Galp-1R	1022	b-D-Galp-1R
920	a-D-Galp-1R	47	b-D-Galp-1R	295	b-D-Galp-OH
966	a-D-Galp-1R	48	b-D-Galp-1R	488	b-D-Galp-OH
967	a-D-Galp-1R	54	b-D-Galp-1R	242	b-D-Galp-OH-3R
435	a-D-Galp-OH	55	b-D-Galp-1R	765	b-D-Galp-OH-3R
482	a-D-Galp-OH	56	b-D-Galp-1R	853	b-D-Galp-OH-3R
766	a-D-Galp-OH-3R	59	b-D-Galp-1R	855	b-D-Galp-OH-3R
854	a-D-Galp-OH-3R	60	b-D-Galp-1R	857	b-D-Galp-OH-3R
856	a-D-Galp-OH-3R	61	b-D-Galp-1R	862	b-D-Galp-OH-3R
858	a-D-Galp-OH-3R	71	b-D-Galp-1R	863	b-D-Galp-OH-3R
864	a-D-Galp-OH-3R	72	b-D-Galp-1R	865	b-D-Galp-OH-3R
866	a-D-Galp-OH-3R	90	b-D-Galp-1R	906	b-D-Galp-OH-3R
981	a-D-Galp-OH-3R	97	b-D-Galp-1R	982	b-D-Galp-OH-3R
984	a-D-Galp-OH-3R	99	b-D-Galp-1R	983	b-D-Galp-OH-3R
986	a-D-Galp-OH-4R	105	b-D-Galp-1R	907	b-D-Galp-OH-4R
768	a-D-Galp-OH-6R	106	b-D-Galp-1R	985	b-D-Galp-OH-4R
988	a-D-Galp-OH-6R	195	b-D-Galp-1R	767	b-D-Galp-OH-6R
998	a-D-Galp-OH-6R	196	b-D-Galp-1R	987	b-D-Galp-OH-6R
282	a-D-Galp-OMe	247	b-D-Galp-1R	997	b-D-Galp-OH-6R
282	a-D-Galp-OMe	248	b-D-Galp-1R	214	b-D-Galp-OMe
299	a-D-Galp-OMe	259	b-D-Galp-1R	283	b-D-Galp-OMe
299	a-D-Galp-OMe	564	b-D-Galp-1R	301	b-D-Galp-OMe
399	a-D-Galp-OMe	564	b-D-Galp-1R	301	b-D-Galp-OMe
399	a-D-Galp-OMe	571	b-D-Galp-1R	400	b-D-Galp-OMe
454	a-D-Galp-OMe	572	b-D-Galp-1R	400	b-D-Galp-OMe
454	a-D-Galp-OMe	574	b-D-Galp-1R	489	b-D-Galp-OMe
483	a-D-Galp-OMe	575	b-D-Galp-1R	489	b-D-Galp-OMe
483	a-D-Galp-OMe	576	b-D-Galp-1R	817	b-D-Galp-OMe-2R
288	a-D-Galp-OMe-2R	577	b-D-Galp-1R	833	b-D-Galp-OMe-2R
40	a-D-Galp-OMe-3R	583	b-D-Galp-1R	97	b-D-Galp-OMe-3R

54	a-D-Galp-OMe-3R	589	b-D-Galp-1R	25	b-D-Galp-OMe-6R
119	a-D-Galp-OMe-3R	623	b-D-Galp-1R	29	b-D-Galp-OMe-6R
121	a-D-Galp-OMe-3R	686	b-D-Galp-1R	98	b-D-Galp-OMe-6R
290	a-D-Galp-OMe-3R	725	b-D-Galp-1R	99	b-D-Galp-OMe-6R
391	a-D-Galp-OMe-3R	765	b-D-Galp-1R	105	b-D-Galp-OMe-6R
392	a-D-Galp-OMe-3R	766	b-D-Galp-1R	106	b-D-Galp-OMe-6R

11.3.3. Mannose

Table 56: Detailed composition of the mannose monosaccharide test file

FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety	FM Ref.	Monosaccharide moiety
3	a-D-Manp-1R	826	a-D-Manp-1R	394	a-D-Manp-OMe-3R
3	a-D-Manp-1R	853	a-D-Manp-1R	431	a-D-Manp-OMe-3R
36	a-D-Manp-1R	854	a-D-Manp-1R	738	a-D-Manp-OMe-3R
39	a-D-Manp-1R	855	a-D-Manp-1R	824	a-D-Manp-OMe-3R
50	a-D-Manp-1R	856	a-D-Manp-1R	848	a-D-Manp-OMe-3R
51	a-D-Manp-1R	888	a-D-Manp-1R	93	a-D-Manp-OMe-4R
53	a-D-Manp-1R	910	a-D-Manp-1R	103	a-D-Manp-OMe-4R
91	a-D-Manp-1R	911	a-D-Manp-1R	739	a-D-Manp-OMe-4R
92	a-D-Manp-1R	912	a-D-Manp-1R	825	a-D-Manp-OMe-4R
93	a-D-Manp-1R	925	a-D-Manp-1R	36	a-D-Manp-OMe-6R
94	a-D-Manp-1R	930	a-D-Manp-1R	94	a-D-Manp-OMe-6R
101	a-D-Manp-1R	978	a-D-Manp-1R	104	a-D-Manp-OMe-6R
102	a-D-Manp-1R	1038	a-D-Manp-1R	253	a-D-Manp-OMe-6R
103	a-D-Manp-1R	1042	a-D-Manp-1R	740	a-D-Manp-OMe-6R
104	a-D-Manp-1R	293	a-D-Manp-OH	826	a-D-Manp-OMe-6R
111	a-D-Manp-1R	439	a-D-Manp-OH	114	b-D-Manp-1R
112	a-D-Manp-1R	441	a-D-Manp-OH	857	b-D-Manp-1R
113	a-D-Manp-1R	476	a-D-Manp-OH	858	b-D-Manp-1R
253	a-D-Manp-1R	287	a-D-Manp-OH-2R	859	b-D-Manp-1R
254	a-D-Manp-1R	418	a-D-Manp-OH-2R	860	b-D-Manp-1R
254	a-D-Manp-1R	420	a-D-Manp-OH-2R	861	b-D-Manp-1R
255	a-D-Manp-1R	912	a-D-Manp-OH-2R	913	b-D-Manp-1R
255	a-D-Manp-1R	925	a-D-Manp-OH-2R	914	b-D-Manp-1R
256	a-D-Manp-1R	930	a-D-Manp-OH-2R	919	b-D-Manp-1R
256	a-D-Manp-1R	931	a-D-Manp-OH-2R	920	b-D-Manp-1R
257	a-D-Manp-1R	1045	a-D-Manp-OH-2R	921	b-D-Manp-1R
257	a-D-Manp-1R	909	a-D-Manp-OH-4R	922	b-D-Manp-1R
257	a-D-Manp-1R	913	a-D-Manp-OH-4R	923	b-D-Manp-1R
258	a-D-Manp-1R	917	a-D-Manp-OH-4R	924	b-D-Manp-1R
258	a-D-Manp-1R	921	a-D-Manp-OH-4R	926	b-D-Manp-1R
258	a-D-Manp-1R	926	a-D-Manp-OH-4R	927	b-D-Manp-1R
567	a-D-Manp-1R	741	a-D-Manp-OH-6R	979	b-D-Manp-1R
568	a-D-Manp-1R	297	a-D-Manp-OMe	980	b-D-Manp-1R
569	a-D-Manp-1R	407	a-D-Manp-OMe	440	b-D-Manp-OH
570	a-D-Manp-1R	434	a-D-Manp-OMe	442	b-D-Manp-OH
573	a-D-Manp-1R	477	a-D-Manp-OMe	480	b-D-Manp-OH
573	a-D-Manp-1R	53	a-D-Manp-OMe-2R	419	b-D-Manp-OH-2R
576	a-D-Manp-1R	91	a-D-Manp-OMe-2R	421	b-D-Manp-OH-2R
577	a-D-Manp-1R	101	a-D-Manp-OMe-2R	908	b-D-Manp-OH-4R
582	a-D-Manp-1R	111	a-D-Manp-OMe-2R	914	b-D-Manp-OH-4R
595	a-D-Manp-1R	115	a-D-Manp-OMe-2R	918	b-D-Manp-OH-4R
596	a-D-Manp-1R	117	a-D-Manp-OMe-2R	922	b-D-Manp-OH-4R
597	a-D-Manp-1R	737	a-D-Manp-OMe-2R	927	b-D-Manp-OH-4R
597	a-D-Manp-1R	823	a-D-Manp-OMe-2R	849	b-D-Manp-OH-6R
627	a-D-Manp-1R	33	a-D-Manp-OMe-3R	408	b-D-Manp-OMe
627	a-D-Manp-1R	52	a-D-Manp-OMe-3R	116	b-D-Manp-OMe-2R
699	a-D-Manp-1R	92	a-D-Manp-OMe-3R	118	b-D-Manp-OMe-2R
823	a-D-Manp-1R	102	a-D-Manp-OMe-3R	594	b-D-Manp-OMe-2R
824	a-D-Manp-1R	112	a-D-Manp-OMe-3R	10	b-D-Manp-OMe-4R
825	a-D-Manp-1R	393	a-D-Manp-OMe-3R		

11.4. GAM disaccharide test file

Table 57: Detailed composition of the GAM disaccharide test file

a-D-Galp-1-1-a-D-Galp	a-D-Glcp-1-4-a-D-Galp-OMe	b-D-Galp-1-3-a-D-Galp	b-D-Glcp-1-3-a-D-Galp-OMe
a-D-Galp-1-3-a-D-Galp-OMe	a-D-Glcp-1-4-a-D-Glcp	b-D-Galp-1-3-a-D-Galp-OMe	b-D-Glcp-1-3-a-D-Glcp
a-D-Galp-1-4-a-D-Galp-OMe	a-D-Glcp-1-4-a-D-Glcp	b-D-Galp-1-3-b-D-Galp	b-D-Glcp-1-3-a-D-Glcp
a-D-Galp-1-4-a-D-Galp-OMe	a-D-Glcp-1-4-a-D-Glcp	b-D-Galp-1-3-b-D-Galp	b-D-Glcp-1-3-a-D-Glcp
a-D-Galp-1-4-b-D-Galp-OMe	a-D-Glcp-1-4-a-D-Glcp-OMe	b-D-Galp-1-3-b-D-Galp-OMe	b-D-Glcp-1-3-a-D-Glcp
a-D-Galp-1-4-b-D-Glcp	a-D-Glcp-1-4-b-D-Galp	b-D-Galp-1-3-b-D-Galp-OMe	b-D-Glcp-1-3-a-D-Glcp-OMe
a-D-Galp-1-4-b-D-Glcp-OMe	a-D-Glcp-1-4-b-D-Glcp	b-D-Galp-1-3-b-D-Glcp-OMe	b-D-Glcp-1-3-a-D-Manp-OMe
a-D-Galp-1-6-a-D-Galp-OMe	a-D-Glcp-1-4-b-D-Glcp	b-D-Galp-1-4-a-D-Glc	b-D-Glcp-1-3-a-D-Manp-OMe
a-D-Galp-1-6-a-D-Glcp	a-D-Glcp-1-4-b-D-Glcp	b-D-Galp-1-4-a-D-Glcp	b-D-Glcp-1-3-b-D-Glcp
a-D-Galp-1-6-a-D-Glcp	a-D-Glcp-1-4-b-D-Glcp-OMe	b-D-Galp-1-4-b-D-Galp-OMe	b-D-Glcp-1-3-b-D-Glcp
a-D-Galp-1-6-b-D-Galp-OMe	a-D-Glcp-1-4-b-D-Glcp-OMe	b-D-Galp-1-4-b-D-Glcp	b-D-Glcp-1-3-b-D-Glcp
a-D-Galp-1-6-b-D-Glcp	a-D-Glcp-1-6-a-D-Galp-OMe	b-D-Galp-1-4-b-D-Glcp	b-D-Glcp-1-3-b-D-Glcp
a-D-Galp-1-6-b-D-Glcp	a-D-Glcp-1-6-a-D-Glcp	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-3-b-D-Glcp-OMe
a-D-Glcp-1-1-a-D-Glcp	a-D-Glcp-1-6-a-D-Glcp	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-4-a-D-Galp-OMe
a-D-Glcp-1-1-a-D-Glcp	a-D-Glcp-1-6-a-D-Glcp	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-4-a-D-Glcp
a-D-Glcp-1-1-a-D-Glcp	a-D-Glcp-1-6-a-D-Glcp-OMe	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-4-a-D-Glcp
a-D-Glcp-1-1-a-D-Glcp	a-D-Glcp-1-6-a-D-Glcp-OMe	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-4-a-D-Glcp
a-D-Glcp-1-1-a-D-Glcp	a-D-Glcp-1-6-b-D-Galp-OMe	b-D-Galp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-4-a-D-Glcp-OMe
a-D-Glcp-1-1-b-D-Glcp	a-D-Glcp-1-6-b-D-Glcp	b-D-Galp-1-6-a-D-Galp	b-D-Glcp-1-4-a-D-Manp
a-D-Glcp-1-2-a-D-Glcp	a-D-Glcp-1-6-b-D-Glcp	b-D-Galp-1-6-a-D-Galp-OMe	b-D-Glcp-1-4-b-D-Glcp
a-D-Glcp-1-2-a-D-Glcp	a-D-Glcp-1-6-b-D-Glcp	b-D-Galp-1-6-a-D-Galp-OMe	b-D-Glcp-1-4-b-D-Glcp
a-D-Glcp-1-2-a-D-Glcp	a-D-Glcp-1-6-b-D-Glcp-OMe	b-D-Galp-1-6-b-D-Galp	b-D-Glcp-1-4-b-D-Glcp
a-D-Glcp-1-2-a-D-Glcp	a-D-Manp-1-1-a-D-Galp	b-D-Galp-1-6-b-D-Galp-OMe	b-D-Glcp-1-4-b-D-Glcp-OMe
a-D-Glcp-1-2-a-D-Glcp-OMe	a-D-Manp-1-1-a-D-Manp	b-D-Galp-1-6-b-D-Galp-OMe	b-D-Glcp-1-4-b-D-Manp
a-D-Glcp-1-2-a-D-Manp-OMe	a-D-Manp-1-2-a-D-Manp	b-D-Galp-1-6-b-D-Galp-OMe	b-D-Glcp-1-6-a-D-Galp-OMe
a-D-Glcp-1-2-b-D-Glcp	a-D-Manp-1-2-a-D-Manp-OMe	b-D-Galp-1-6-b-D-Glcp-OMe	b-D-Glcp-1-6-a-D-Glcp
a-D-Glcp-1-2-b-D-Glcp	a-D-Manp-1-2-a-D-Manp-OMe	b-D-Galp-1-6-b-D-Glcp-OMe	b-D-Glcp-1-6-a-D-Glcp
a-D-Glcp-1-2-b-D-Glcp	a-D-Manp-1-2-a-D-Manp-OMe	b-D-Glcp-1-1-a-D-Glcp	b-D-Glcp-1-6-a-D-Glcp
a-D-Glcp-1-2-b-D-Glcp	a-D-Manp-1-2-a-D-Manp-OMe	b-D-Glcp-1-1-a-D-Glcp	b-D-Glcp-1-6-a-D-Glcp
a-D-Glcp-1-2-b-D-Glcp-OMe	a-D-Manp-1-2-a-D-Manp-OMe	b-D-Glcp-1-1-a-D-Glcp	b-D-Glcp-1-6-a-D-Glcp-OMe
a-D-Glcp-1-2-b-D-Manp-OMe	a-D-Manp-1-3-a-D-Manp-OMe	b-D-Glcp-1-1-b-D-Glcp	b-D-Glcp-1-6-b-D-Galp-OMe
a-D-Glcp-1-3-a-D-Galp-OMe	a-D-Manp-1-3-a-D-Manp-OMe	b-D-Glcp-1-1-b-D-Glcp	b-D-Glcp-1-6-b-D-Glcp
a-D-Glcp-1-3-a-D-Glcp	a-D-Manp-1-3-a-D-Manp-OMe	b-D-Glcp-1-2-a-D-Glcp	b-D-Glcp-1-6-b-D-Glcp
a-D-Glcp-1-3-a-D-Glcp	a-D-Manp-1-3-a-D-Manp-OMe	b-D-Glcp-1-2-a-D-Glcp	b-D-Glcp-1-6-b-D-Glcp
a-D-Glcp-1-3-a-D-Glcp	a-D-Manp-1-4-a-D-Manp-OMe	b-D-Glcp-1-2-a-D-Glcp	b-D-Glcp-1-6-b-D-Glcp
a-D-Glcp-1-3-a-D-Glcp	a-D-Manp-1-4-a-D-Manp-OMe	b-D-Glcp-1-2-a-D-Glcp-OMe	b-D-Glcp-1-6-b-D-Glcp-OMe
a-D-Glcp-1-3-a-D-Glcp-OMe	a-D-Manp-1-4-a-D-Manp-OMe	b-D-Glcp-1-2-a-D-Glcp-OMe	b-D-Glcp-1-6-b-D-Glcp-OMe
a-D-Glcp-1-3-a-D-Manp-OMe	a-D-Manp-1-4-b-D-Glcp-OMe	b-D-Glcp-1-2-a-D-Manp-OMe	b-D-Manp-1-2-b-D-Manp-OMe
a-D-Glcp-1-3-b-D-Galp	a-D-Manp-1-6-a-D-Glcp-OMe	b-D-Glcp-1-2-b-D-Glcp	b-D-Manp-1-4-a-D-Glcp
a-D-Glcp-1-3-b-D-Glcp	a-D-Manp-1-6-a-D-Manp-OMe	b-D-Glcp-1-2-b-D-Glcp	b-D-Manp-1-4-a-D-Manp
a-D-Glcp-1-3-b-D-Glcp	a-D-Manp-1-6-a-D-Manp-OMe	b-D-Glcp-1-2-b-D-Glcp	b-D-Manp-1-4-b-D-Glcp
a-D-Glcp-1-3-b-D-Glcp	a-D-Manp-1-6-a-D-Manp-OMe	b-D-Glcp-1-2-b-D-Glcp-OMe	b-D-Manp-1-4-b-D-Manp
a-D-Glcp-1-3-b-D-Glcp	a-D-Manp-1-6-a-D-Manp-OMe	b-D-Glcp-1-2-b-D-Manp-OMe	b-D-Manp-1-6-a-D-Glcp-OMe
a-D-Glcp-1-3-b-D-Glcp-OMe	b-D-Galp-1-2-b-D-Galp-OMe	b-D-Glcp-1-3-a-D-Galp-OMe	

Curriculum Vitae

Matthias Dominik Studer-Imwinkelried
eidg. dipl. pharm

Born: 11th June 1974 in Basel

May 2005 – May 2006

Postdoctoral fellow
University of Basel

Bioinformatics / Chemoinformatics

Design Studies related to the development of distributed, Web-based European carbohydrate databases (**EUROCarbDB**)
www.eurocarbdb.org

Education

September 2005

Ph.D. exam (magna cum laude)

January 2001- May 2005

Ph.D. thesis - University of Basel

In the field of Bioinformatics (Artificial Neural Networks / Pharmaceutical Chemistry / NMR spectroscopy / Carbohydrates)
In the group of Prof. Beat Ernst – Institute of Molecular Pharmacy

Title "*NeuroCarb - Artificial Neural Networks for NMR Structure Elucidation of Oligosaccharides*" www.neurocarb.ch

November 2000

Swiss federal diploma in pharmacy

Diploma work

Focused on molecular modeling - In the group of Prof. Beat Ernst

Title "*Homology Modeling und Molecular Dynamics Studien von E-Selectin*"

1994 – 2000

Pharmacy studies - University of Basel

December 1993

Federal maturity exam (economics) - Grammar school Liestal

Teaching experience

2000 – 2004

Supervision of undergraduate students

Molecular modeling for students of pharmaceutical sciences

Prof. Beat Ernst
Prof. Angelo Vedani

2000 – 2003

Lectures

"*Homology Modeling*"

In the context of the lecture course
"Advanced Molecular Modeling"

Prof. Angelo Vedani
Biographics Laboratory 3R

since 2000

Computer administrator

- Responsible for IT infrastructure of the Institute of Molecular Pharmacy
- Further training courses and seminars for students, graduate student and postdocs.

Pof. Beat Ernst

Further university training

2004 - 2005

"venture challenge"

venturelab FJ Institut für Jungunternehmen

Semester course in entrepreneurship (~60 lectures) Company analysis, marketing, communication, sales, law financing and business plans

Internships

1994

F. Hoffmann-La Roche AG, Basel - Solida POMF-IP

In process control of solid drug formulations Dr. Werner Erni (interpharma)
Dr. Gregor Wolany †

Work experience

1997 – 1998

Birs Apotheke Birsfelden

Internship in a public pharmacy during pharmacy studies Ursula Refardt

1998

Kantonsspital Bruderholz

Clinical internship Dr. Hans-Martin Grünig

1996 – 1997

F. Hoffmann-La Roche AG, Basel - Liquida PTFP-IP

- In process control of i.v. formulations Dr. Werner Erni
- Qualification of high sterile production lines (class 100) (interpharma)
- Validation and documentation of computer systems Dr. Gregor Wolany †
- Documentation of computer systems
- Collaboration in a GMP-laboratory

since 1990

Adler Apotheke Liestal & Apotheke Bubendorf

Pharmacist H.J. & U. Studer-Schweizer

Language and computer skills

Languages	German	mother tongue
	English	fluent (oral and written)
	French	good knowledge

Computer Skills	UNIX (SGI IRIX)* Linux (RedHat 7 - 9 ES/WS)* Mac OS X* Windows (all versions)* Apache Web Server*, IIS Web Server, Samba File Server*, SSH*/SFTP/VPN, FileMaker Server*, MySQL
-----------------	--

Software

- Molecular Modeling	Tripes SYBYL* Schroedinger Macromodel Biograf: Yeti, PrGen, Quasar
- Neural Networks	SNNS*, JavaNNS*, (Statsoft Statistica)*
- NMR	Bruker XWIN-NMR
- Data analysis software	Statsoft Statistica
- Office	Microsoft Office (incl. Visio)*
- Development platforms	Microsoft Visual Studio Eclipse
- Chemistry	SciFinder Scholar Beilstein Crossfire

Web Technologies	XML/HTML*, PHP, CDML*
------------------	-----------------------

Web Development Tools	Macromedia Dreamweaver* Macromedia Fireworks*
-----------------------	--

Programming languages	Visual Basic .NET C++
-----------------------	--------------------------

* High degree of proficiency

Publications

Januar 2006

Bioinformatics for Glycobiology and Glycomics (Wiley – in press)
Dr. Claus-Wilhelm von der Lieth (DKFZ Heidelberg)

Chapter : Neural Networks for structure elucidation of
oligosaccharides

Summer 2006

Publication in preparation (Journal of Organic Chemistry):
*Artificial neural networks for NMR structure elucidation of
disaccharides* / M. Studer, A. Stoeckli and B. Ernst

Posters & Public lectures

Posters:

- 2002 World Congress on Computational Intelligence
Honolulu, Hawaii (Mai 2002)
- Swiss Chemical Society - Fall Meeting, Lausanne
(October 2003)

Public lectures:

- DKFZ, Heidelberg (April 2005)
- Bijvoet Center for Biomolecular Research, Utrecht
(December 2005)
- Computer Chemistry Center, Erlangen (December 2005)

Basel, 2006



M. Studer