Naïve Discriminative Learning:
Theoretical and Experimental Observations

Stefan Evert[1] & Antti Arppe[2]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
stefan.evert@fau.de

[2]University of Alberta, Edmonton, Canada
arppe@ualberta.ca

QITL-6, Tübingen, 6 Nov 2015

Outline

1. Introduction
   - Naïve Discriminative Learning
   - An example
2. Mathematics
   - The Rescorla-Wagner equations
   - The Danks equilibrium
   - NDL vs. the Perceptron vs. least-squares regression
3. Insights
   - Theoretical insights
   - Empirical observations
   - Conclusion

Outline

Objectives

- Explain the mathematical foundations of Naïve Discriminative Learning (NDL) in one place and in a consistent way
- Highlight the theoretical similarities of NDL with linear/logistic regression and the single-layer perceptron
- Present some empirical simulations of stochastic NDL learners, in light of the theoretical insights

Naïve Discriminative Learning

- Baayen (2011); Baayen *et al.* (2011)
- Incremental learning equations for direct associations between cues and outcomes (Rescorla and Wagner 1972)
- Equilibrium conditions (Danks 2003)
- Implementation as R package `ndl` (Arppe *et al.* 2014)

Naive: cue-outcome associations estimated separately for each outcome (this independence assumption is similar to a naive Bayesian classifier)

Discriminative: cues predict outcomes based on total activation level = sum of direct cue-outcome associations

Learning: incremental learning of association strengths

The Rescorla-Wagner equations (1972)

Represent incremental associative learning and subsequent on-going adjustments to an accumulating body of knowledge.

Changes in cue-outcome association strengths:

- No change if a cue is not present in the input
- Increased if the cue and outcome co-occur
- Decreased if the cue occurs without the outcome
- If outcome can already be predicted well (based on all input cues), adjustments become smaller

Only results of incremental adjustments to the cue-outcome associations are kept – no need for remembering the individual adjustments, however many there are.

Danks (2003) equilibrium conditions

- Presume an ideal stable "adult" state, where all cue-outcome associations have been fully learnt – further data points should then have no impact on the cue-outcome associations
- Provide a convenient short-cut to calculating the final cue-outcome association weights resulting from incremental learning, using relatively simple matrix algebra
- Most learning parameters of the Rescorla-Wagner equations drop out of the Danks equilibrium equation
- Circumvent the problem that a simulation of an R-W learner does usually not converge to a stable state unless the learning rate is gradually decreased

Traditional vs. linguistic applications of R-W

- Traditionally: simple controlled experiments on item-by-item learning, with only a handful of cues and perfect associations
- Natural language: full of choices among multiple possible alternatives – phones, words, or constructions – which are influenced by a large number of contextual factors, and which often show weak to moderate tendencies towards one or more of the alternatives rather than a single unambiguous decision
- These messy, complex types of problems are a key area of interest in modeling and understanding language use
- Application of R-W in the form of a Naïve Discriminative Learner to such linguistic classification problems is sucessful in practice and can throw new light on research questions

Related work

- R-W vs. perceptron (Sutton and Barto 1981, p. 155f)
- R-W vs. least-squares regression (Stone 1986, p. 457)
- R-W vs. logistic regression (Gluck and Bower 1988, p. 234)
- R-W vs. neural networks (Dawson 2008)

☞ similarities are also mentioned by many other authors . . .

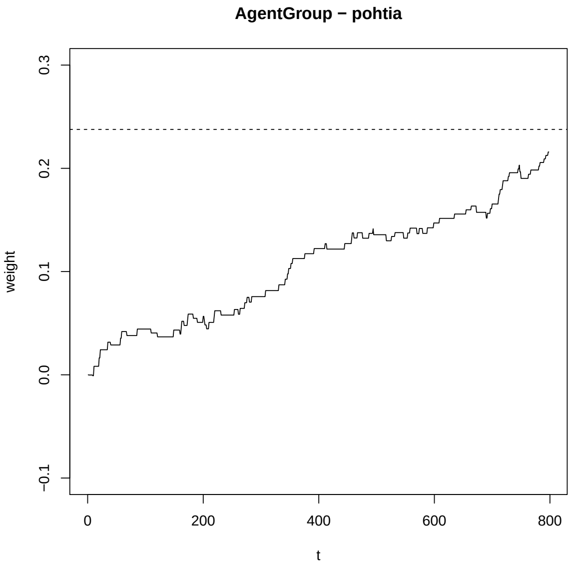
Outline

Simple vs. complex settings – QITL-1 revisited

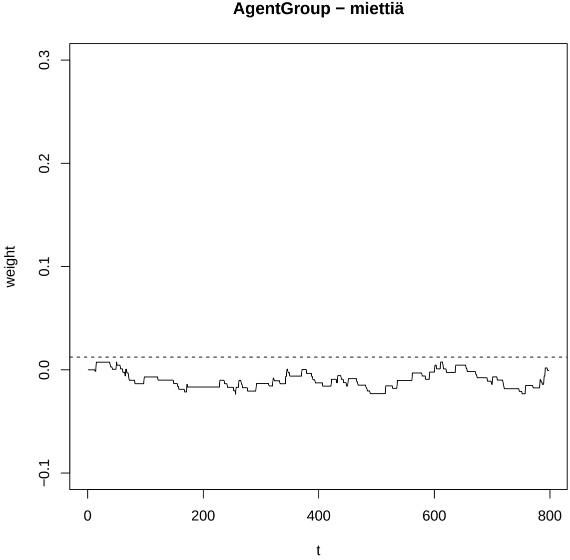
- Arppe and Järvikivi (2002, 2007)
- *Person* (FIRST PERSON SINGULAR or not) and *Countability* (COLLECTIVE or not) of AGENT/SUBJECT of Finnish verb synonym pair *miettiä* vs. *pohtia* 'think, ponder':

Forced-choice		Frequency (relative)	Acceptability	
Dispreferred	Preferred		Unacceptable	Acceptable
∅	miettiä+SG1 pohtia+COLL	Frequent	∅	miettiä+SG1 pohtiaä+COLL
miettiä+COLL pohtia+SG1	∅	Rare	miettiä+COLL	pohtia+SG1

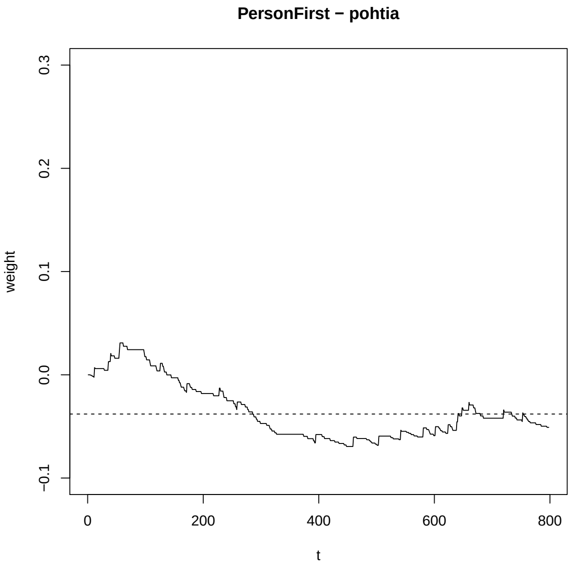
QITL-1 through the lens of NDL



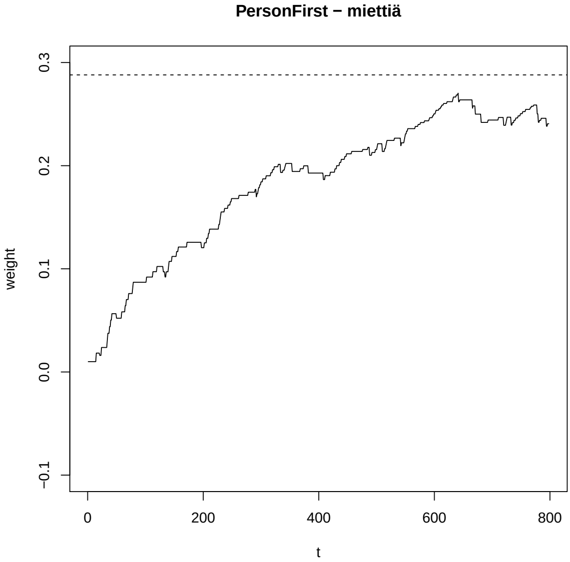
QITL-1 through the lens of NDL



QITL-1 through the lens of NDL

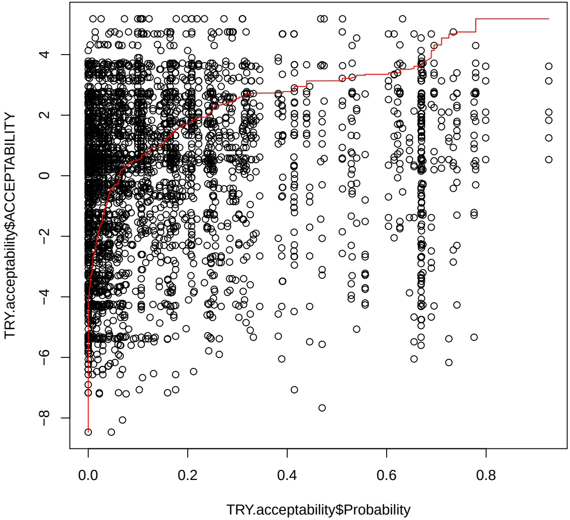


QITL-1 through the lens of NDL

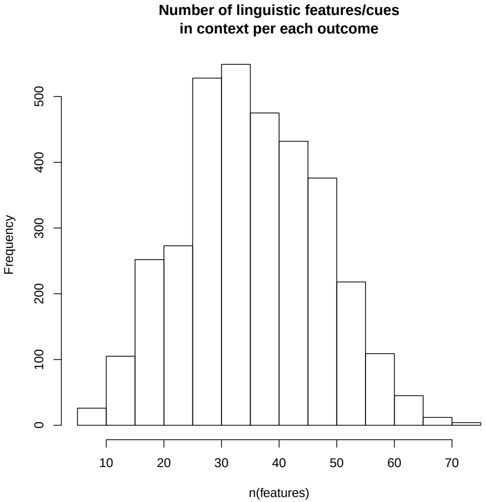


QITL-1 through the lens of QITL-6

(courtesy of Dagmar Divjak)



Simple vs. complex settings – QITL-2 revisited



QITL-4 revisited – NDL vs. statistical classifiers

	$\lambda_{\text{prediction}}$	$\tau_{\text{classification}}$	accuracy
Polytomous logistic regression (One-vs-rest)	0.368	0.488	**0.645**
Polytomous mixed logistic regression (Poisson reformulation)			
• 1|Section	0.360	0.482	0.640
• 1|Author	0.358	0.481	0.640
• 1|Section + 1—Author	0.358	0.481	0.640
Support Vector Machine	0.340	0.466	0.629
Memory-Based Learning (TiMBL)	0.286	0.422	0.599
Random Forests	0.326	0.455	0.621
Naive Discriminative Learning	0.346	0.471	0.632

Table: Classification diagnostics for models fitted to the Finnish data set ($n = 3404$).

Outline

1. Introduction
   - Naïve Discriminative Learning
   - An example
2. Mathematics
   - The Rescorla-Wagner equations
   - The Danks equilibrium
   - NDL vs. the Perceptron vs. least-squares regression
3. Insights
   - Theoretical insights
   - Empirical observations
   - Conclusion

The Rescorla-Wagner equations

- Goal of naïve discriminative learner: predict an outcome O based on presence or absence of a set of cues $C_1, \ldots, C_n$

The Rescorla-Wagner equations

- Goal of naïve discriminative learner: predict an outcome O based on presence or absence of a set of cues $C_1, \dots, C_n$
- An event $(\mathbf{c}, o)$ is formally described by indicator variables

$$c_i = \begin{cases} 1 & \text{if } C_i \text{ is present} \\ 0 & \text{otherwise} \end{cases} \qquad o = \begin{cases} 1 & \text{if } O \text{ results} \\ 0 & \text{otherwise} \end{cases}$$

The Rescorla-Wagner equations

- Goal of naïve discriminative learner: predict an outcome O based on presence or absence of a set of cues $C_1, \dots, C_n$
- An event $(\mathbf{c}, o)$ is formally described by indicator variables

$$c_i = \begin{cases} 1 & \text{if } C_i \text{ is present} \\ 0 & \text{otherwise} \end{cases} \qquad o = \begin{cases} 1 & \text{if } O \text{ results} \\ 0 & \text{otherwise} \end{cases}$$

- Given cue-outcome associations $\mathbf{v} = (V_1, \dots, V_n)$ of learner, the activation level of the outcome O is

$$\sum_{j=1}^{n} c_j V_j$$

The Rescorla-Wagner equations

- Goal of naïve discriminative learner: predict an outcome O based on presence or absence of a set of cues $C_1, \dots, C_n$
- An event $(\mathbf{c}, o)$ is formally described by indicator variables

$$c_i = \begin{cases} 1 & \text{if } C_i \text{ is present} \\ 0 & \text{otherwise} \end{cases} \qquad o = \begin{cases} 1 & \text{if } O \text{ results} \\ 0 & \text{otherwise} \end{cases}$$

- Given cue-outcome associations $\mathbf{v} = (V_1, \dots, V_n)$ of learner, the activation level of the outcome O is

$$\sum_{j=1}^{n} c_j^{(t)} V_j^{(t)}$$

- Associations $\mathbf{v}^{(t)}$ as well as cue and outcome indicators $(\mathbf{c}^{(t)}, o^{(t)})$ depend on time step t

The Rescorla-Wagner equations

- Rescorla and Wagner (1972) proposed the R-W equations for the change in associations given an event $(\mathbf{c}, o)$:

$$\Delta V_i = \begin{cases} 0 & \text{if } c_i = 0 \\ \alpha_i \beta_1 \big(\lambda - \sum_{j=1}^n c_j V_j \big) & \text{if } c_i = 1 \wedge o = 1 \\ \alpha_i \beta_2 \big(0 - \sum_{j=1}^n c_j V_j \big) & \text{if } c_i = 1 \wedge o = 0 \end{cases}$$

with parameters

$\lambda > 0$	target activation level for outcome O
$\alpha_i > 0$	salience of cue C_i
$\beta_1 > 0$	learning rate for positive ovents $(o = 1)$
$\beta_2 > 0$	learning rate for negative ovents $(o = 0)$

The Widrow-Hoff rule

- The W-H rule (Widrow and Hoff 1960) is a widely-used simplification of the R-W equations:

$$\Delta V_i = \begin{cases} 0 & \text{if } c_i = 0 \\ \alpha_i \beta_1 \left(\lambda - \sum_{j=1}^{n} c_j V_j\right) & \text{if } c_i = 1 \wedge o = 1 \\ \alpha_i \beta_2 \left(0 - \sum_{j=1}^{n} c_j V_j\right) & \text{if } c_i = 1 \wedge o = 0 \end{cases}$$

with parameters

$\lambda = 1$	target activation level for outcome O
$\alpha_i = 1$	salience of cue C_i
$\beta_1 = \beta_2$	global learning rate for positive and
$= \beta > 0$	negative events

The Widrow-Hoff rule

- The W-H rule (Widrow and Hoff 1960) is a widely-used simplification of the R-W equations:

$$\Delta V_i = \begin{cases} 0 & \text{if } c_i = 0 \\ \beta\bigl(1 - \sum_{j=1}^{n} c_j V_j\bigr) & \text{if } c_i = 1 \wedge o = 1 \\ \beta\bigl(0 - \sum_{j=1}^{n} c_j V_j\bigr) & \text{if } c_i = 1 \wedge o = 0 \end{cases}$$

with parameters

$\lambda = 1$	target activation level for outcome O
$\alpha_i = 1$	salience of cue C_i
$\beta_1 = \beta_2$	global learning rate for positive and
$= \beta > 0$	negative events

The Widrow-Hoff rule

- The W-H rule (Widrow and Hoff 1960) is a widely-used simplification of the R-W equations:

$$
\begin{aligned}
\Delta V_i &= \begin{cases} 0 & \text{if } c_i = 0 \\ \beta\left(1 - \sum_{j=1}^{n} c_j V_j\right) & \text{if } c_i = 1 \wedge o = 1 \\ \beta\left(0 - \sum_{j=1}^{n} c_j V_j\right) & \text{if } c_i = 1 \wedge o = 0 \end{cases} \\
&= c_i \beta\left(o - \sum_{j=1}^{n} c_j V_j\right)
\end{aligned}
$$

with parameters

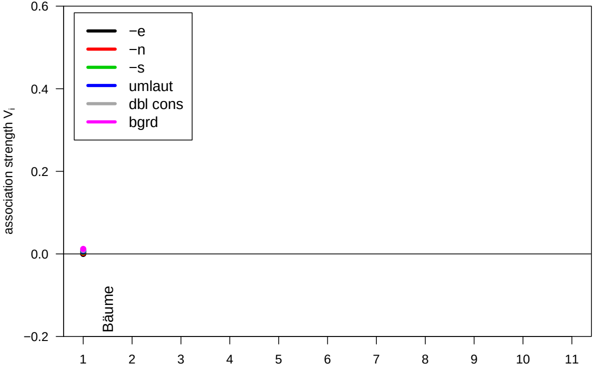
$\lambda = 1$	target activation level for outcome O
$\alpha_i = 1$	salience of cue C_i
$\beta_1 = \beta_2$	global learning rate for positive and
$= \beta > 0$	negative events

A simple example: German noun plurals

t	word	o pl?	c_1 $-e$	c_2 $-n$	c_3 $-s$	c_4 umlaut	c_5 dbl cons	c_6 bgrd
1	Bäume	1	1	0	0	1	0	1
2	Flasche	0	1	0	0	0	0	1
3	Baum	0	0	0	0	0	0	1
4	Gläser	1	0	0	0	1	0	1
5	Flaschen	1	0	1	0	0	0	1
6	Latte	0	1	0	0	0	1	1
7	Hütten	1	0	1	0	1	1	1
8	Glas	0	0	0	1	0	0	1
9	Bäume	1	1	0	0	1	0	1
10	Füße	1	1	0	0	1	0	1

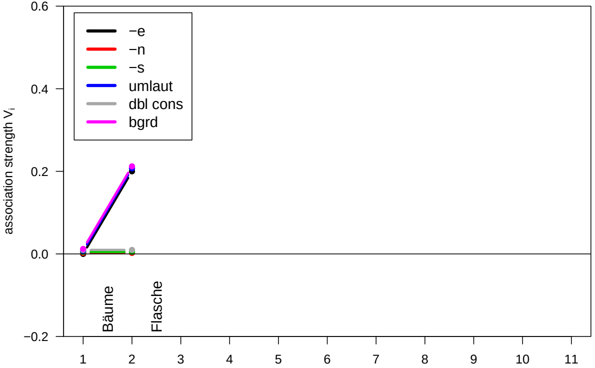
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
1	.000	.000	.000	.000	.000	.000	.000
Bäume	1	1	0	0	1	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



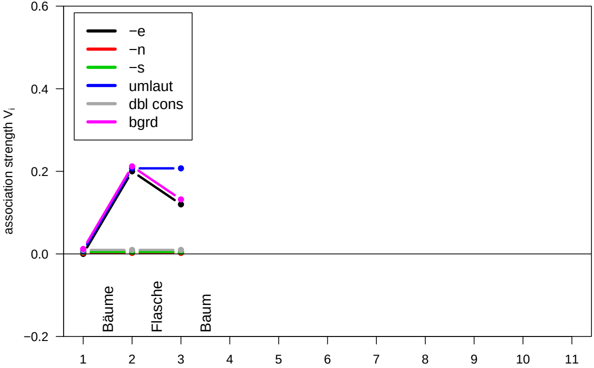
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
2	.400	.200	.000	.000	.200	.000	.200
Flasche	0	1	0	0	0	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



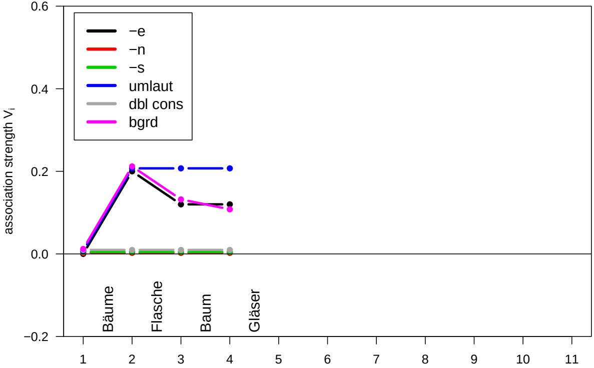
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
3	.120	.120	.000	.000	.200	.000	.120
Baum	0	0	0	0	0	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



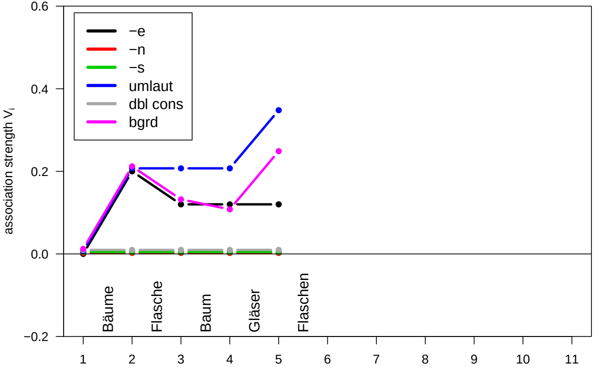
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
4	.296	.120	.000	.000	.200	.000	.096
Gläser	1	0	0	0	1	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



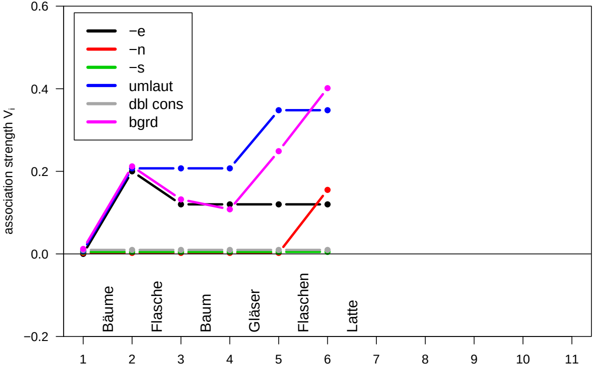
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
5	.237	.120	.000	.000	.341	.000	.237
Flaschen	1	0	1	0	0	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



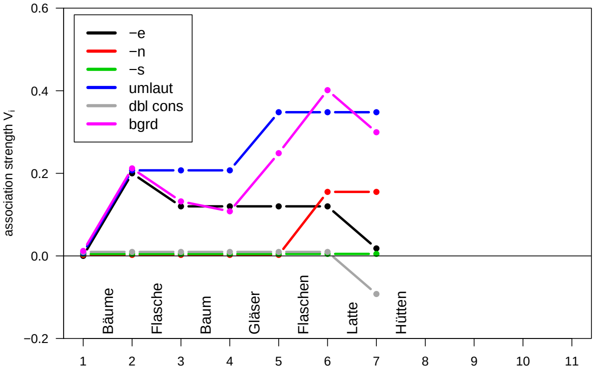
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
6	.509	.120	.153	.000	.341	.000	.389
Latte	0	1	0	0	0	1	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



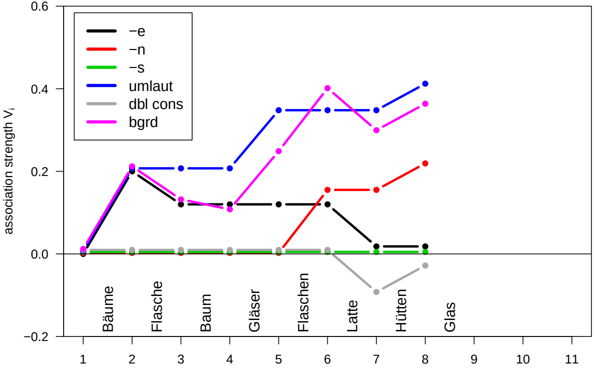
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
7	.679	.018	.153	.000	.341	-.102	.288
Hütten	1	0	1	0	1	1	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



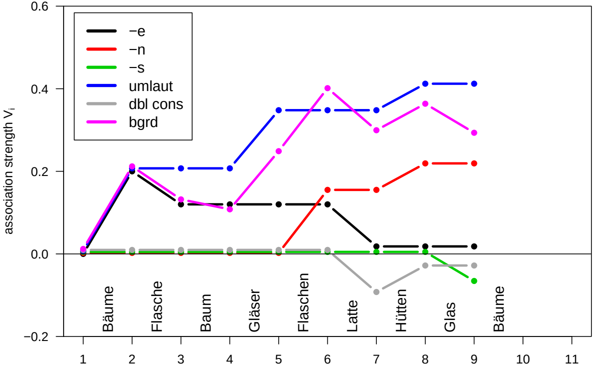
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
8	.352	.018	.217	.000	.405	-.038	.352
Glas	0	0	0	1	0	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



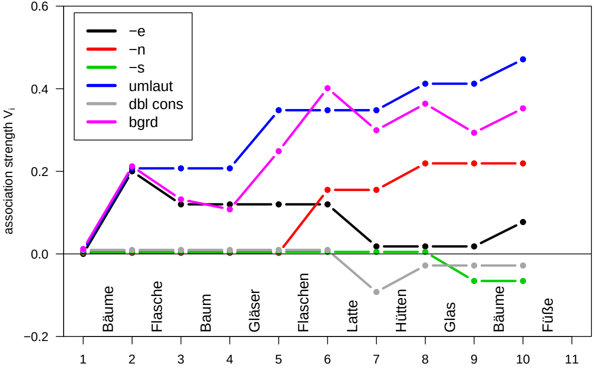
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
9	.704	.018	.217	-.070	.405	-.038	.281
Bäume	1	1	0	0	1	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



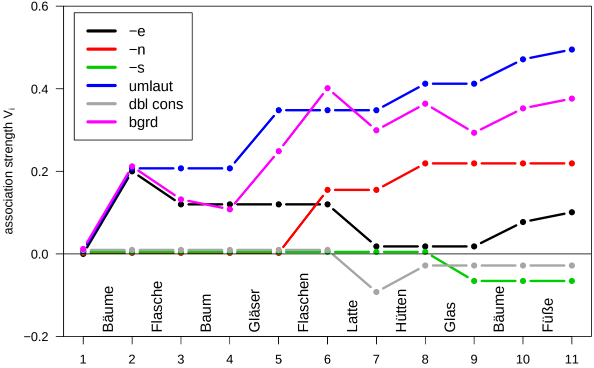
A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
10	.882	.077	.217	-.070	.464	-.038	.340
Füße	1	1	0	0	1	0	1
	o	c_1	c_2	c_3	c_4	c_5	c_6



A simple example: German noun plurals

t	$\sum c_j V_j$	V_1	V_2	V_3	V_4	V_5	V_6
11		.101	.217	-.070	.488	-.038	.364
	o	c_1	c_2	c_3	c_4	c_5	c_6



A stochastic NDL learner

- A specific event sequence $(\mathbf{c}^{(t)}, o^{(t)})$ will only be encountered in controlled experiments

A stochastic NDL learner

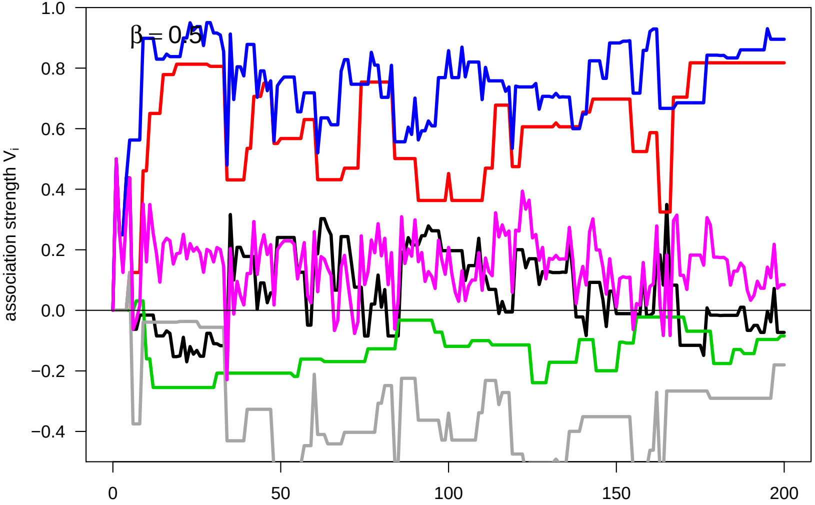
- A specific event sequence $(\mathbf{c}^{(t)}, o^{(t)})$ will only be encountered in controlled experiments
- For applications in corpus linguistics, it is more plausible to assume that events are randomly sampled from a population of event tokens $(\mathbf{c}^{(k)}, o^{(k)})$ for $k = 1, \ldots, m$
  - ☞ event types listed repeatedly proportional to their frequency

A stochastic NDL learner

- A specific event sequence $(\mathbf{c}^{(t)}, o^{(t)})$ will only be encountered in controlled experiments
- For applications in corpus linguistics, it is more plausible to assume that events are randomly sampled from a population of event tokens $(\mathbf{c}^{(k)}, o^{(k)})$ for $k = 1, \ldots, m$
  - ☞ event types listed repeatedly proportional to their frequency
- I.i.d. random variables $\mathbf{c}^{(t)} \sim \mathbf{c}$ and $o^{(t)} \sim o$
  - ☞ distributions of $\mathbf{c}$ and o determined by population
- NDL can now be trained for arbitrary number of time steps, even if population is small (as in our example)
  - study asymptotic behaviour of learners
  - convergence ➜ stable "adult" state of associations

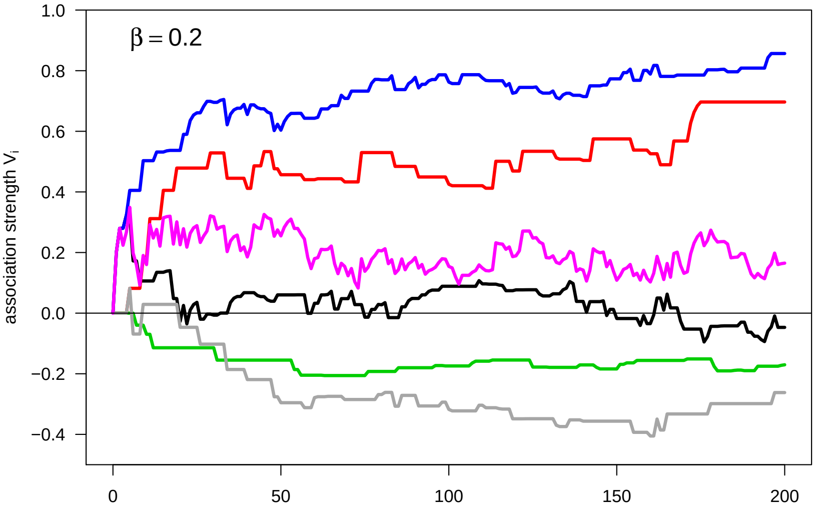
A stochastic NDL learner

Effect of the learning rate β



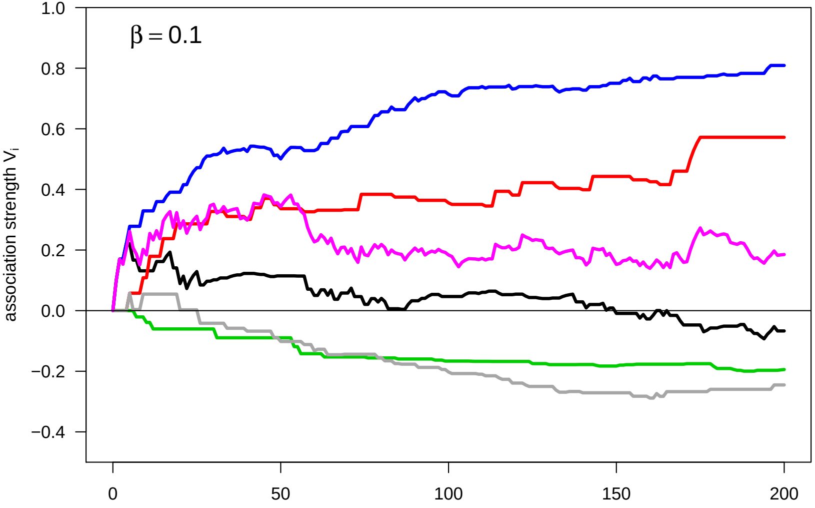
A stochastic NDL learner

Effect of the learning rate β



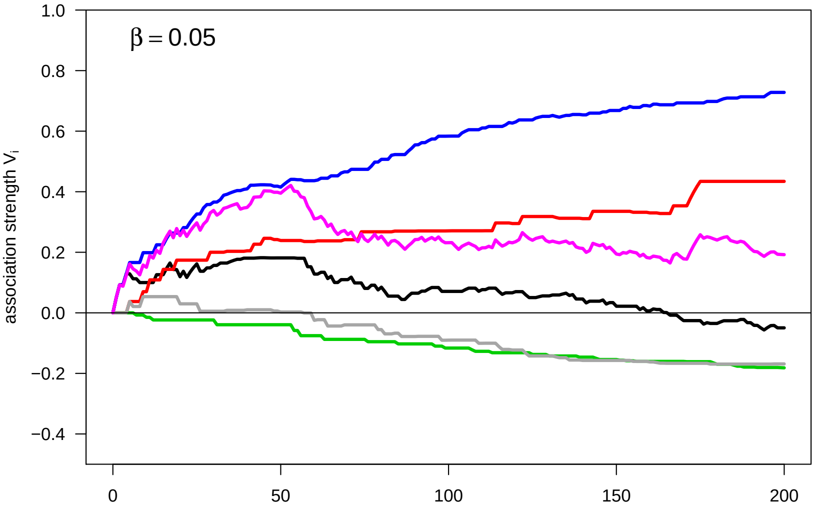
A stochastic NDL learner

Effect of the learning rate β



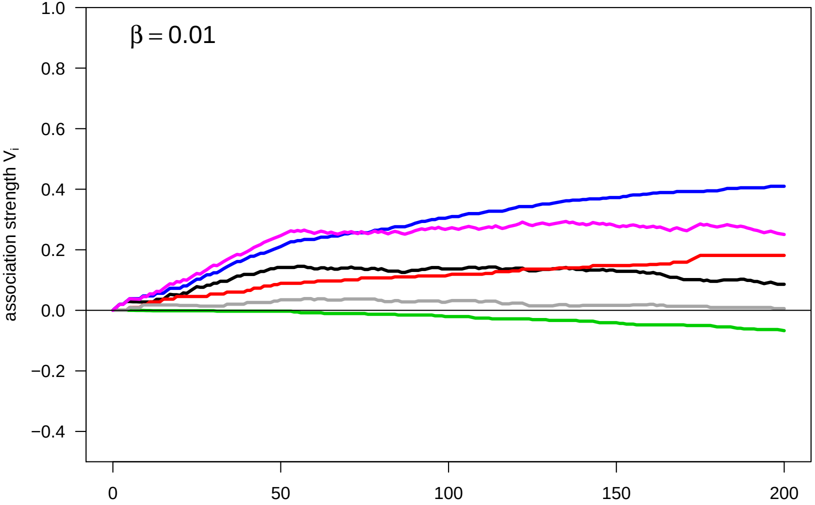
A stochastic NDL learner

Effect of the learning rate β



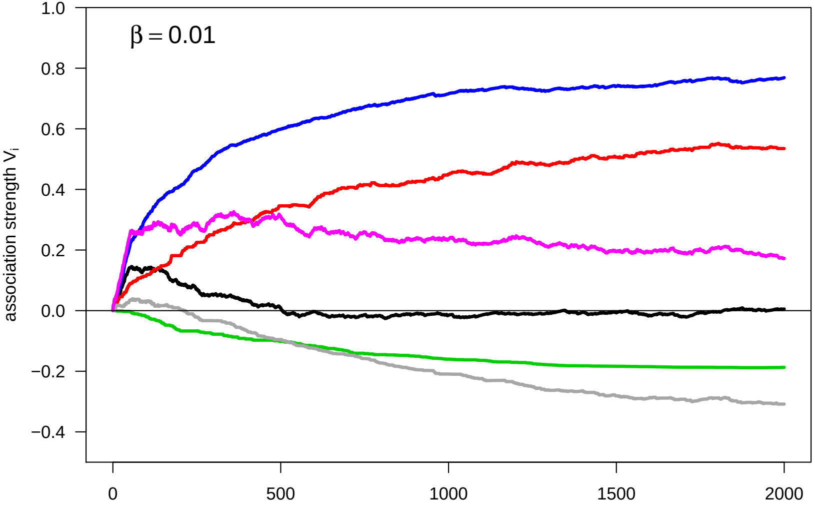
A stochastic NDL learner

Effect of the learning rate β



A stochastic NDL learner

Effect of the learning rate β



Outline

Expected activation levels

- Since we are interested in the general behaviour of a stochastic NDL, it makes sense to average over many individual learners to obtain expected associations $\mathrm{E}[V_j^{(t)}]$

$$\mathrm{E}[V_{j+1}^{(t)}] = \mathrm{E}[V_j^{(t)}] + \mathrm{E}[\Delta V_j^{(t)}]$$

$$\mathrm{E}[\Delta V_j^{(t)}] = \mathrm{E}\left[c_i \beta \bigl(o - \textstyle\sum_{j=1}^{n} c_j V_j^{(t)}\bigr)\right]$$

Expected activation levels

- Since we are interested in the general behaviour of a stochastic NDL, it makes sense to average over many individual learners to obtain expected associations $\mathrm{E}[V_j^{(t)}]$

$$\mathrm{E}[V_{j+1}^{(t)}] = \mathrm{E}[V_j^{(t)}] + \mathrm{E}[\Delta V_j^{(t)}]$$

$$\begin{aligned}\mathrm{E}[\Delta V_j^{(t)}] &= \mathrm{E}\left[c_i \beta \big(o - \textstyle\sum_{j=1}^n c_j V_j^{(t)}\big)\right] \\ &= \beta \cdot \mathrm{E}[c_i o] - \beta \cdot \mathrm{E}\left[c_i \textstyle\sum_{j=1}^n c_j V_j^{(t)}\right]\end{aligned}$$

Expected activation levels

- Since we are interested in the general behaviour of a stochastic NDL, it makes sense to average over many individual learners to obtain expected associations $\mathrm{E}[V_j^{(t)}]$

$$\mathrm{E}[V_{j+1}^{(t)}] = \mathrm{E}[V_j^{(t)}] + \mathrm{E}[\Delta V_j^{(t)}]$$

$$\begin{aligned}\mathrm{E}[\Delta V_j^{(t)}] &= \mathrm{E}\left[c_i \beta \left(o - \textstyle\sum_{j=1}^n c_j V_j^{(t)}\right)\right] \\ &= \beta \cdot \mathrm{E}[c_i o] - \beta \cdot \textstyle\sum_{j=1}^n \mathrm{E}[c_i c_j V_j^{(t)}]\end{aligned}$$

- c_i and c_j are independent from $V_j^{(t)}$

Expected activation levels

- Since we are interested in the general behaviour of a stochastic NDL, it makes sense to average over many individual learners to obtain expected associations $\mathrm{E}[V_j^{(t)}]$

$$\mathrm{E}[V_{j+1}^{(t)}] = \mathrm{E}[V_j^{(t)}] + \mathrm{E}[\Delta V_j^{(t)}]$$

$$\begin{aligned}\mathrm{E}[\Delta V_j^{(t)}] &= \mathrm{E}\left[c_i \beta \left(o - \textstyle\sum_{j=1}^n c_j V_j^{(t)}\right)\right] \\ &= \beta \cdot \mathrm{E}[c_i o] - \beta \cdot \textstyle\sum_{j=1}^n \mathrm{E}[c_i c_j] \mathrm{E}[V_j^{(t)}]\end{aligned}$$

- c_i and c_j are independent from $V_j^{(t)}$
- indicator variables: $\mathrm{E}[c_i o] = \Pr(C_i, O)$; $\mathrm{E}[c_i c_j] = \Pr(C_i, C_j)$

Expected activation levels

- Since we are interested in the general behaviour of a stochastic NDL, it makes sense to average over many individual learners to obtain expected associations $\mathrm{E}\big[V_j^{(t)}\big]$

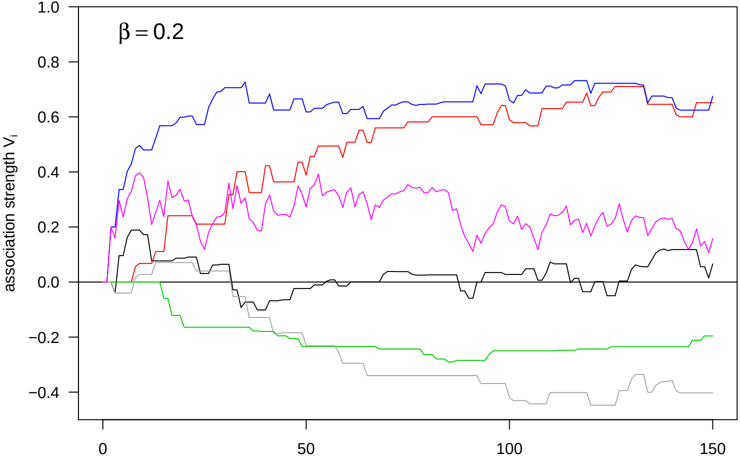
$$\mathrm{E}\big[V_{j+1}^{(t)}\big] = \mathrm{E}\big[V_j^{(t)}\big] + \mathrm{E}\big[\Delta V_j^{(t)}\big]$$

$$\begin{aligned}\mathrm{E}\big[\Delta V_j^{(t)}\big] &= \mathrm{E}\left[c_i \beta \big(o - \textstyle\sum_{j=1}^n c_j V_j^{(t)}\big)\right] \\ &= \beta \cdot \Big(\mathrm{Pr}(C_i, O) - \textstyle\sum_{j=1}^n \mathrm{Pr}(C_i, C_j) \mathrm{E}\big[V_j^{(t)}\big]\Big)\end{aligned}$$

- c_i and c_j are independent from $V_j^{(t)}$
- indicator variables: $\mathrm{E}[c_i o] = \mathrm{Pr}(C_i, O)$; $\mathrm{E}[c_i c_j] = \mathrm{Pr}(C_i, C_j)$

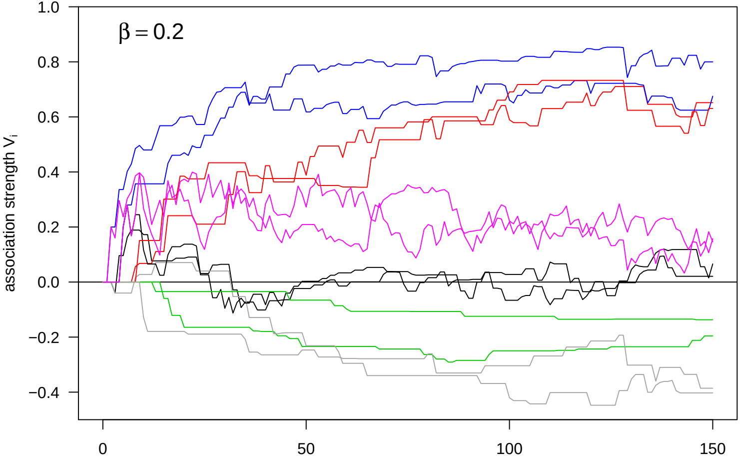
Expected activation levels

$$\Delta V_j^{(t)} = c_i^{(t)} \beta \big(o^{(t)} - \textstyle\sum_{j=1}^{n} c_j^{(t)} V_j^{(t)} \big)$$



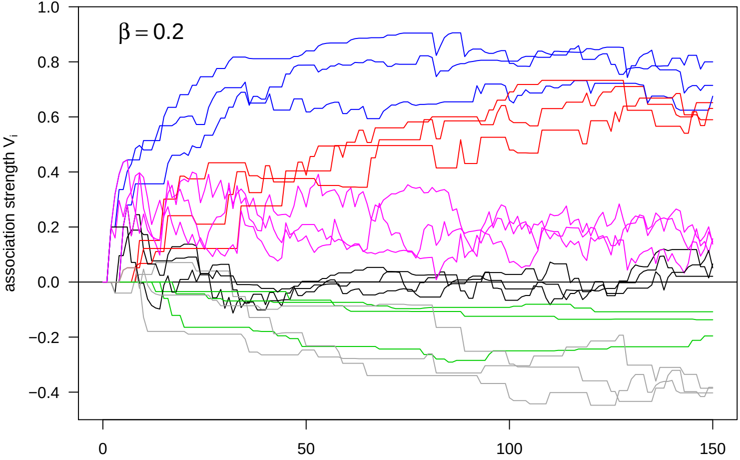
Expected activation levels

$$\Delta V_j^{(t)} = c_i^{(t)} \beta \left(o^{(t)} - \sum_{j=1}^{n} c_j^{(t)} V_j^{(t)} \right)$$



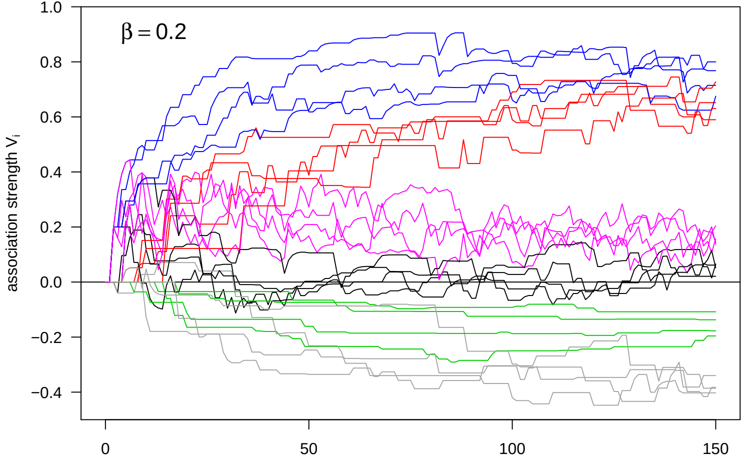
Expected activation levels

$$\Delta V_j^{(t)} = c_i^{(t)} \beta \left(o^{(t)} - \sum_{j=1}^{n} c_j^{(t)} V_j^{(t)} \right)$$



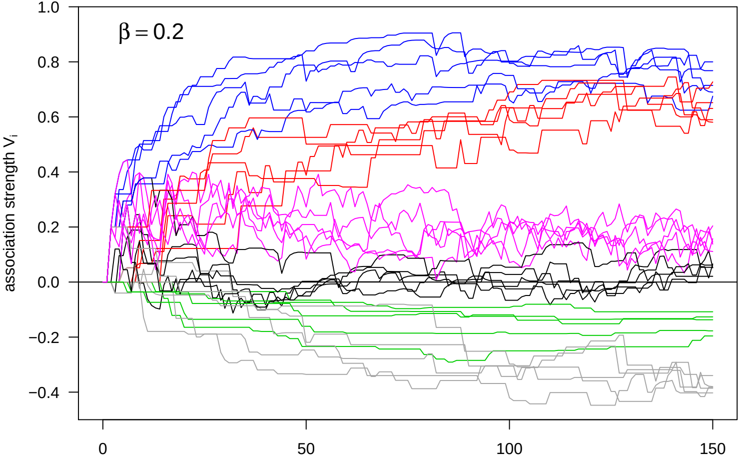
Expected activation levels

$$\Delta V_j^{(t)} = c_i^{(t)} \beta \left(o^{(t)} - \sum_{j=1}^{n} c_j^{(t)} V_j^{(t)} \right)$$



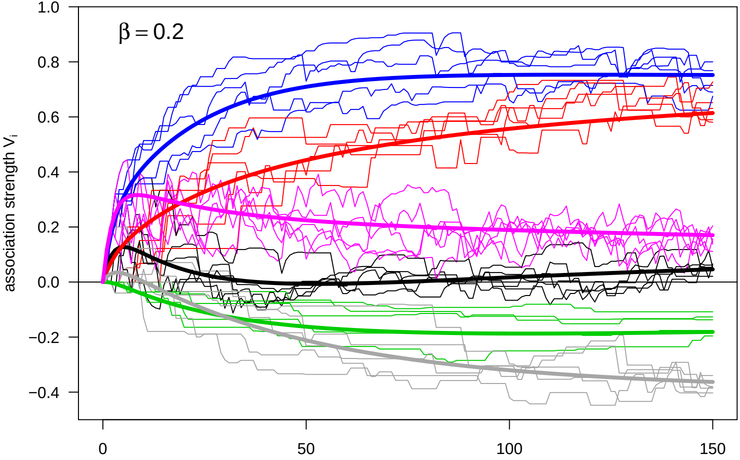
Expected activation levels

$$\Delta V_j^{(t)} = c_i^{(t)} \beta \left(o^{(t)} - \sum_{j=1}^{n} c_j^{(t)} V_j^{(t)} \right)$$



Expected activation levels

$$\mathrm{E}[\Delta V_j^{(t)}] = \beta \cdot \left(\Pr(C_i, O) - \sum_{j=1}^{n} \Pr(C_i, C_j)\mathrm{E}[V_j^{(t)}]\right)$$



The Danks equilibrium

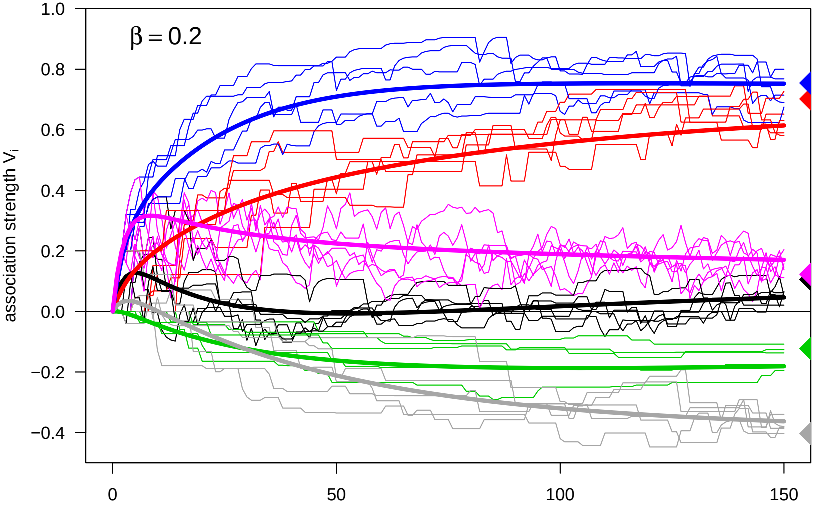
- If $\mathrm{E}\big[V_i^{(t)}\big]$ converges, the asymptote $V_i^* = \lim_{t\to\infty} \mathrm{E}\big[V_i^{(t)}\big]$ must satisfy the Danks equilibrium conditions $\mathrm{E}\big[\Delta V_i^*\big] = 0$, i.e.

$$\mathrm{Pr}(C_i, O) - \textstyle\sum_{j=1}^{n} \mathrm{Pr}(C_i, C_j) V_j^* = 0 \quad \forall i$$

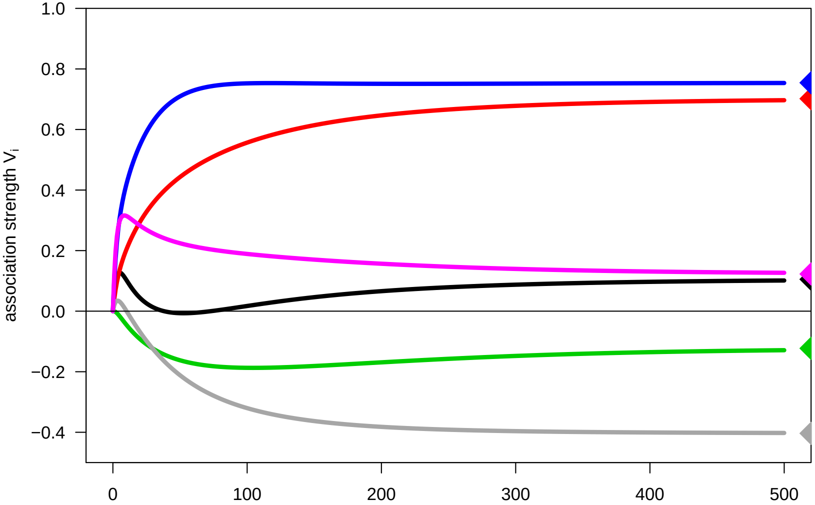
  (Danks 2003, p. 113)

- Now there is a clear interpretation of the Danks equilibrium as the stable average associations reached by a community of stochastic learners with input from the same population
  - ☞ allows us to compute the "adult" state of NDL without carrying out a simulation of the learning process

The Danks equilibrium



The Danks equilibrium



Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} f(C_1, O) \\ \vdots \\ f(C_n, O) \end{bmatrix} = \mathbf{X}^T \mathbf{z}$$

Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} f(C_1, O) \\ \vdots \\ f(C_n, O) \end{bmatrix} = \mathbf{X}^T \mathbf{z} \qquad \begin{bmatrix} f(C_1, C_1) & \cdots & f(C_1, C_n) \\ \vdots & & \vdots \\ f(C_n, C_1) & \cdots & f(C_n, C_n) \end{bmatrix} = \mathbf{X}^T \mathbf{X}$$

Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} \Pr(C_1, O) \\ \vdots \\ \Pr(C_n, O) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{z} \qquad \begin{bmatrix} \Pr(C_1, C_1) & \cdots & \Pr(C_1, C_n) \\ \vdots & & \vdots \\ \Pr(C_n, C_1) & \cdots & \Pr(C_n, C_n) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{X}$$

Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} \Pr(C_1, O) \\ \vdots \\ \Pr(C_n, O) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{z} \qquad \begin{bmatrix} \Pr(C_1, C_1) & \cdots & \Pr(C_1, C_n) \\ \vdots & & \vdots \\ \Pr(C_n, C_1) & \cdots & \Pr(C_n, C_n) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{X}$$

Danks equilibrium: $\frac{1}{m}\mathbf{X}^T\mathbf{z} - \frac{1}{m}\mathbf{X}^T\mathbf{X}\mathbf{w}^* = \mathbf{0}$

Matrix notation

$$\mathbf{X} = \begin{bmatrix} c_1^{(1)} & \cdots & c_n^{(1)} \\ c_1^{(2)} & \cdots & c_n^{(2)} \\ \vdots & & \vdots \\ c_1^{(m)} & \cdots & c_n^{(m)} \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} o^{(1)} \\ o^{(2)} \\ \vdots \\ o^{(m)} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} \Pr(C_1, O) \\ \vdots \\ \Pr(C_n, O) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{z} \qquad \begin{bmatrix} \Pr(C_1, C_1) & \cdots & \Pr(C_1, C_n) \\ \vdots & & \vdots \\ \Pr(C_n, C_1) & \cdots & \Pr(C_n, C_n) \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{X}$$

Danks equilibrium: $\mathbf{X}^T\mathbf{z} = \mathbf{X}^T\mathbf{X}\mathbf{w}^*$

Matrix notation: German noun plurals

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

Matrix notation: German noun plurals

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} 3 \\ 2 \\ 0 \\ 5 \\ 1 \\ 6 \end{bmatrix} = \mathbf{X}^T \mathbf{z}$$

Matrix notation: German noun plurals

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} 3 \\ 2 \\ 0 \\ 5 \\ 1 \\ 6 \end{bmatrix} = \mathbf{X}^T \mathbf{z} \qquad \begin{bmatrix} 5 & 0 & 0 & 3 & 1 & 5 \\ 0 & 2 & 0 & 1 & 1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 3 & 1 & 0 & 5 & 1 & 5 \\ 1 & 1 & 0 & 1 & 2 & 2 \\ 5 & 2 & 1 & 5 & 2 & 10 \end{bmatrix} = \mathbf{X}^T \mathbf{X}$$

Matrix notation: German noun plurals

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} V^{(1)} \\ \vdots \\ V^{(n)} \end{bmatrix}$$

$$\begin{bmatrix} .3 \\ .2 \\ .0 \\ .5 \\ .1 \\ .6 \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{z} \qquad \begin{bmatrix} .5 & .0 & .0 & .3 & .1 & .5 \\ .0 & .2 & .0 & .1 & .1 & .2 \\ .0 & .0 & .1 & .0 & .0 & .1 \\ .3 & .1 & .0 & .5 & .1 & .5 \\ .1 & .1 & .0 & .1 & .2 & .2 \\ .5 & .2 & .1 & .5 & .2 & 1 \end{bmatrix} = \frac{1}{m}\mathbf{X}^T\mathbf{X}$$

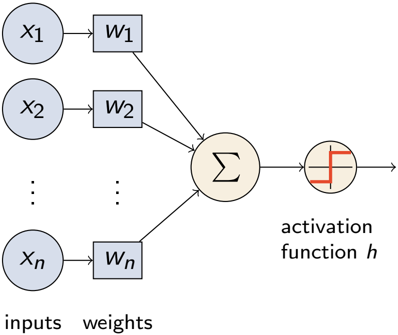
Outline

The single-layer perceptron (SLP)

SLP (Rosenblatt 1958) is most basic feed-forward neural network

- numeric inputs $x_1, \ldots, x_n$
- output activation $h(y)$ based on weighted sum of inputs

$$y = \sum_{j=1}^{n} w_j x_j$$



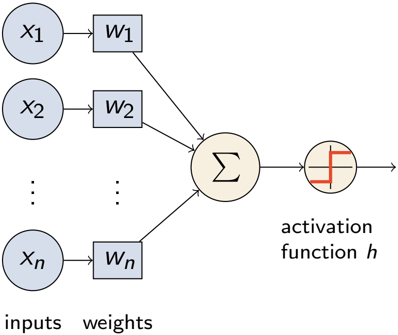
The single-layer perceptron (SLP)

SLP (Rosenblatt 1958) is most basic feed-forward neural network

- numeric inputs $x_1, \ldots, x_n$
- output activation $h(y)$ based on weighted sum of inputs

$$y = \sum_{j=1}^{n} w_j x_j$$

- $h =$ Heaviside step function in traditional SLP



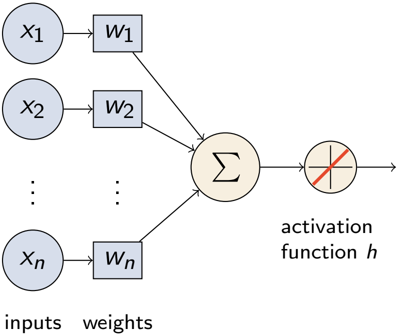
The single-layer perceptron (SLP)

SLP (Rosenblatt 1958) is most basic feed-forward neural network

- numeric inputs $x_1, \ldots, x_n$
- output activation $h(y)$ based on weighted sum of inputs

$$y = \sum_{j=1}^{n} w_j x_j$$

- $h =$ Heaviside step function in traditional SLP
- even simpler model: $h(y) = y$



The single-layer perceptron (SLP)

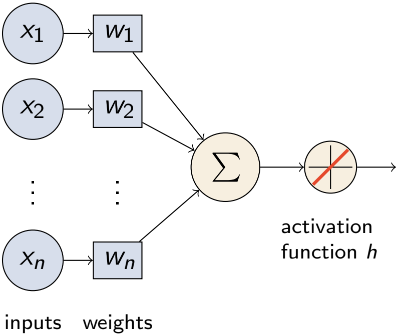
SLP (Rosenblatt 1958) is most basic feed-forward neural network

- numeric inputs $x_1, \ldots, x_n$
- output activation $h(y)$ based on weighted sum of inputs

$$y = \sum_{j=1}^{n} w_j x_j$$

- h = Heaviside step function in traditional SLP
- even simpler model: $h(y) = y$
- cost wrt. target output z:

$$E(\mathbf{w}, \mathbf{x}, z) = \left(z - \sum_{j=1}^{n} w_j x_j\right)^2$$



SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\Delta w_i = -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i}$$

SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\begin{aligned}\Delta w_i &= -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i} \\ &= -\delta \frac{\partial}{\partial w_i} \left(z - \sum_{j=1}^{n} w_j x_j \right)^2\end{aligned}$$

SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\begin{aligned}\Delta w_i &= -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i} \\ &= -2\delta \left(z - \sum_{j=1}^{n} w_j x_j \right) (-x_i)\end{aligned}$$

SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\begin{aligned}\Delta w_i &= -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i} \\ &= -2\delta \left(z - \sum_{j=1}^{n} w_j x_j \right) (-x_i) \\ &= \beta c_i \left(o - \textstyle\sum_{j=1}^{n} c_j V_j \right)\end{aligned}$$

SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\begin{aligned}
\Delta w_i &= -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i} \\
&= 2\delta x_i \left(z - \sum_{j=1}^{n} x_j w_j \right) \\
&= \beta c_i \left(o - \textstyle\sum_{j=1}^{n} c_j V_j \right)
\end{aligned}$$

SLP training: the delta rule

- SLP weights are learned by gradient descent training: for a single training item $(\mathbf{x}, z)$ and learning rate $\delta > 0$

$$\Delta w_i = -\delta \frac{\partial E(\mathbf{w}, \mathbf{x}, z)}{\partial w_i}$$
$$= 2\delta x_i \left(z - \sum_{j=1}^{n} x_j w_j \right)$$
$$= \beta c_i \left(o - \sum_{j=1}^{n} c_j V_j \right)$$

- Perfect correspondence to W-H rule with

$$V_i = w_i \qquad c_i = x_i \qquad o = z \qquad \beta = 2\delta$$

Batch training

- Neural networks often use batch training, where all training data are considered at once instead of one item at a time
- The corresponding batch training cost is

$$E(\mathbf{w}) = \frac{1}{m} \sum_{k=1}^{m} E(\mathbf{w}, \mathbf{x}^{(k)}, z^{(k)})$$

Batch training

- Neural networks often use batch training, where all training data are considered at once instead of one item at a time
- The corresponding batch training cost is

$$E(\mathbf{w}) = \frac{1}{m} \sum_{k=1}^{m} E(\mathbf{w}, \mathbf{x}^{(k)}, z^{(k)})$$

- Similar to stochastic NDL, batch training computes the expected weights $\mathrm{E}\left[\mathbf{w}^{(t)}\right]$ for an SLP with stochastic input

Batch training

- Neural networks often use batch training, where all training data are considered at once instead of one item at a time
- The corresponding batch training cost is

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{m} \sum_{k=1}^{m} E(\mathbf{w}, \mathbf{x}^{(k)}, z^{(k)}) \\
&= \frac{1}{m} \sum_{k=1}^{m} \left(z^{(k)} - \textstyle\sum_{j=1}^{n} w_j x_j^{(k)} \right)^2
\end{aligned}
$$

- Similar to stochastic NDL, batch training computes the expected weights $\mathrm{E}\big[\mathbf{w}^{(t)}\big]$ for an SLP with stochastic input
- Minimization of $E(\mathbf{w})$ = linear least-squares regression

Linear least-squares regression

- Matrix formulation of the linear least-squares problem:

$$E(\mathbf{w}) = \frac{1}{m} \sum_{k=1}^{m} \left(z^{(k)} - \textstyle\sum_{j=1}^{n} w_j x_j^{(k)} \right)^2$$

Linear least-squares regression

- Matrix formulation of the linear least-squares problem:

$$
\begin{aligned}
E(\mathbf{w}) &= \frac{1}{m} \sum_{k=1}^{m} \left(z^{(k)} - \textstyle\sum_{j=1}^{n} w_j x_j^{(k)} \right)^2 \\
&= \frac{1}{m} \left(\mathbf{z} - \mathbf{X}\mathbf{w} \right)^T \left(\mathbf{z} - \mathbf{X}\mathbf{w} \right)
\end{aligned}
$$

Linear least-squares regression

- Matrix formulation of the linear least-squares problem:

$$E(\mathbf{w}) = \frac{1}{m}\sum_{k=1}^{m}\left(z^{(k)} - \textstyle\sum_{j=1}^{n} w_j x_j^{(k)}\right)^2$$
$$= \frac{1}{m}(\mathbf{z} - \mathbf{X}\mathbf{w})^T(\mathbf{z} - \mathbf{X}\mathbf{w})$$

- Minimum of $E(\mathbf{w})$, the L_2 solution, must satisfy $\nabla E(\mathbf{w}^*) = \mathbf{0}$, which leads to the normal equations

$$\mathbf{X}^T\mathbf{z} = \mathbf{X}^T\mathbf{X}\mathbf{w}^*$$

Linear least-squares regression

- Matrix formulation of the linear least-squares problem:

$$E(\mathbf{w}) = \frac{1}{m}\sum_{k=1}^{m}\left(z^{(k)} - \textstyle\sum_{j=1}^{n} w_j x_j^{(k)}\right)^2$$
$$= \frac{1}{m}(\mathbf{z} - \mathbf{X}\mathbf{w})^T(\mathbf{z} - \mathbf{X}\mathbf{w})$$

- Minimum of $E(\mathbf{w})$, the L_2 solution, must satisfy $\nabla E(\mathbf{w}^*) = \mathbf{0}$, which leads to the normal equations

$$\mathbf{X}^T\mathbf{z} = \mathbf{X}^T\mathbf{X}\mathbf{w}^*$$

- Normal equations = Danks equilibrium conditions

Linear least-squares regression

- Matrix formulation of the linear least-squares problem:

$$E(\mathbf{w}) = \frac{1}{m} \sum_{k=1}^{m} \left(z^{(k)} - \sum_{j=1}^{n} w_j x_j^{(k)} \right)^2$$
$$= \frac{1}{m} \left(\mathbf{z} - \mathbf{X}\mathbf{w} \right)^T \left(\mathbf{z} - \mathbf{X}\mathbf{w} \right)$$

- Minimum of $E(\mathbf{w})$, the L_2 solution, must satisfy $\nabla E(\mathbf{w}^*) = \mathbf{0}$, which leads to the normal equations

$$\mathbf{X}^T \mathbf{z} = \mathbf{X}^T \mathbf{X} \mathbf{w}^*$$

- Normal equations = Danks equilibrium conditions
- Regression theory shows that batch training / stochastic NLP converges to the unique* solution of the L_2 problem

What have we learned?

$$\boxed{\begin{array}{ccccc} \text{stochastic} & = & \text{batch} & = & L_2 \text{ regression} \\ \text{NDL} & = & \text{SLP} & & \end{array}}$$

☞ These equivalences also hold for the general R-W equations with arbitrary values of α_i, β_1, β_2 and λ (see paper)

Outline

Effects of R-W parameters

$\beta > 0$: learning rate ➜ convergence of individual learners

Effects of R-W parameters

$\beta > 0$: learning rate ➜ convergence of individual learners

$\lambda \neq 1$: linear scaling of associations / activation (obvious)

Effects of R-W parameters

$\beta > 0$: learning rate ➜ convergence of individual learners

$\lambda \neq 1$: linear scaling of associations / activation (obvious)

$\alpha_i \neq 1$: salience of cue C_i determines how fast associations are learned, but does not affect the final stable associations (same L_2 regression problem)

Effects of R-W parameters

$\beta > 0$: learning rate ➜ convergence of individual learners

$\lambda \neq 1$: linear scaling of associations / activation (obvious)

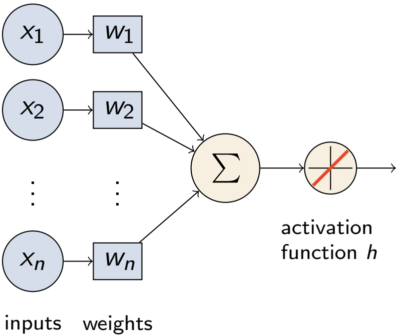
$\alpha_i \neq 1$: salience of cue C_i determines how fast associations are learned, but does not affect the final stable associations (same L_2 regression problem)

$\beta_1 \neq \beta_2$: different positive/negative learning rates *do* affect the stable associations; closely related to prevalence of positive and negative events in the population

What about logistic regression?

Logistic regression is the standard tool for predicting a categorical response from binary features

- can be expressed as SLP with probabilistic interpretation

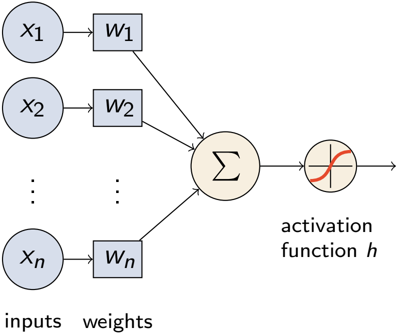


What about logistic regression?

Logistic regression is the standard tool for predicting a categorical response from binary features

- can be expressed as SLP with probabilistic interpretation
- uses logistic activation function

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

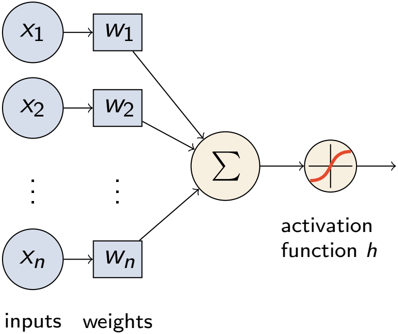


What about logistic regression?

Logistic regression is the standard tool for predicting a categorical response from binary features

- can be expressed as SLP with probabilistic interpretation
- uses logistic activation function

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



- and Bernoulli cost

$$E(\mathbf{w}, \mathbf{x}, z) = \begin{cases} -\log h(y) & \text{if } z = 1 \\ -\log(1 - h(y)) & \text{if } z = 0 \end{cases}$$

What about logistic regression?

- Gradient descent training leads to delta rule that corresponds to a modified version of the R-W equations

$$\Delta V_i = \begin{cases} 0 & \text{if } c_i = 0 \\ \beta\left(1 - h\left(\sum_{j=1}^{n} c_j V_j\right)\right) & \text{if } c_i = 1 \land o = 1 \\ \beta\left(0 - h\left(\sum_{j=1}^{n} c_j V_j\right)\right) & \text{if } c_i = 1 \land o = 0 \end{cases}$$

What about logistic regression?

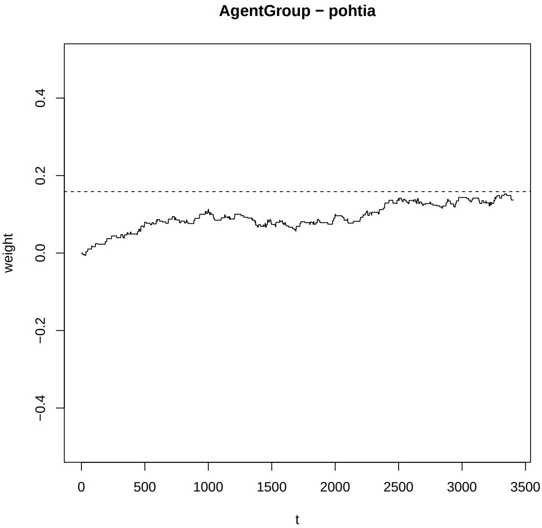
- Gradient descent training leads to delta rule that corresponds to a modified version of the R-W equations

$$\Delta V_i = \begin{cases} 0 & \text{if } c_i = 0 \\ \beta\left(1 - h\left(\sum_{j=1}^{n} c_j V_j\right)\right) & \text{if } c_i = 1 \wedge o = 1 \\ \beta\left(0 - h\left(\sum_{j=1}^{n} c_j V_j\right)\right) & \text{if } c_i = 1 \wedge o = 0 \end{cases}$$

- Same as original R-W, except that activation level is now transformed into probability $h(y)$
- But no easy way to analyze stochastic learning process (batch training $\neq$ expected value of single-item training)
- Less robust for highly predictable outcomes ➜ **w** diverges

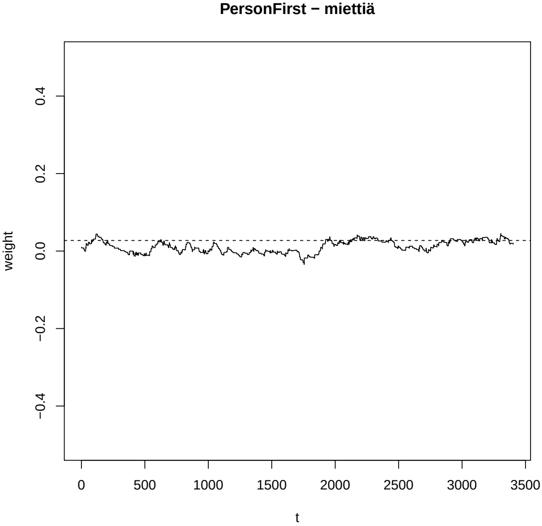
Outline

Some NDL simulation runs



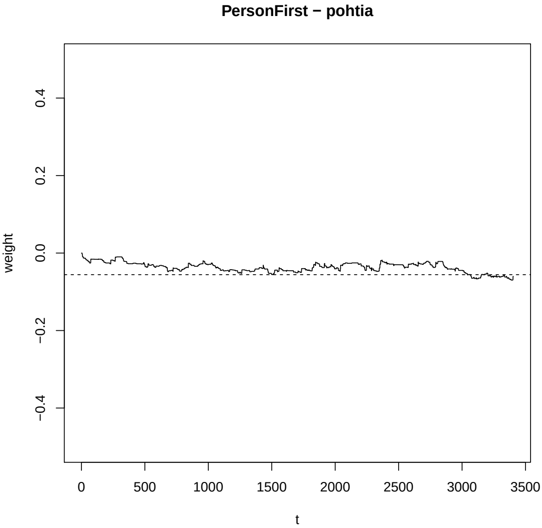
moderate positive association ➜ convergence

Some NDL simulation runs



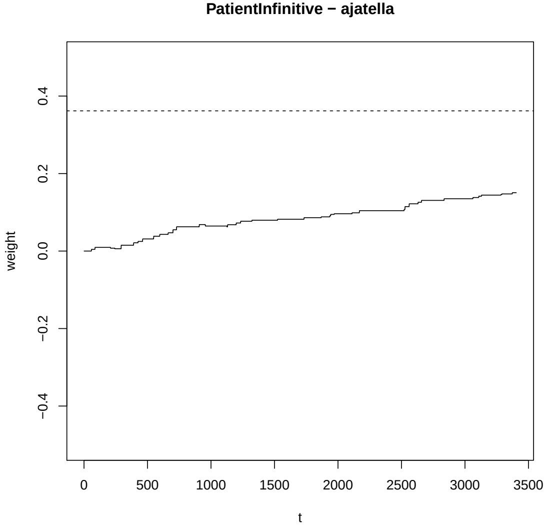
equivocal association ➜ convergence

Some NDL simulation runs



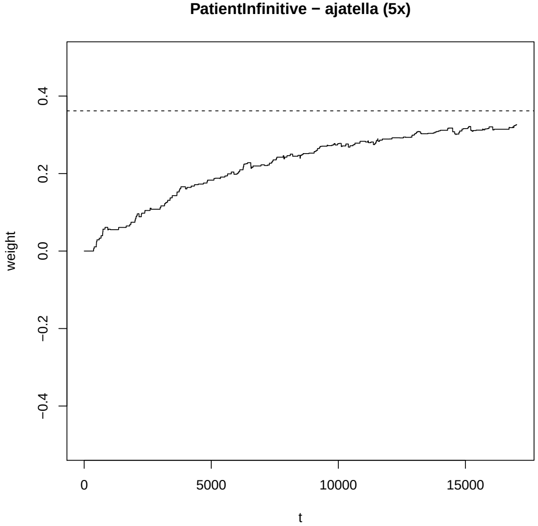
equivocal association ➜ convergence

Some NDL simulation runs



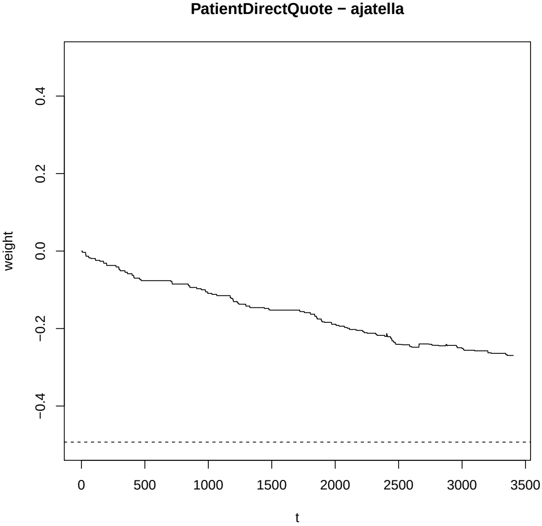
near-perfect positive association ➜ non-convergence with 1× data

Some NDL simulation runs



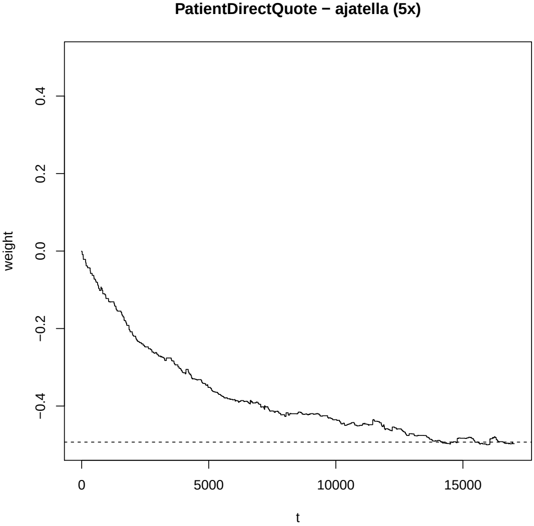
near-perfect positive association ➜ convergence with 5× data

Some NDL simulation runs



near-perfect negative association ➜ non-convergence with 1× data

Some NDL simulation runs

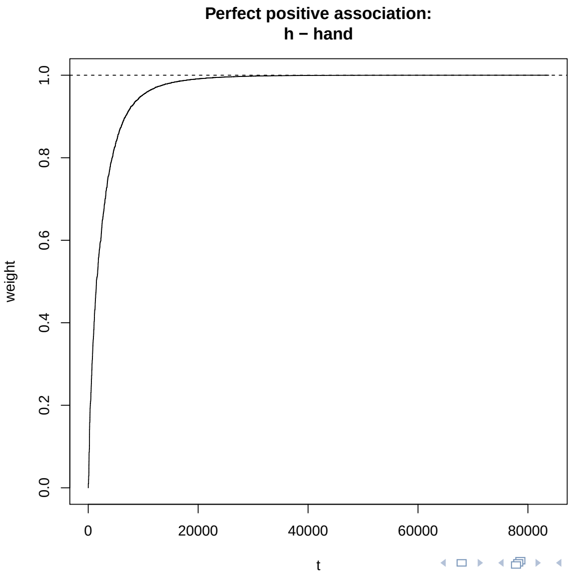


near-perfect negative association ➜ convergence with 5× data

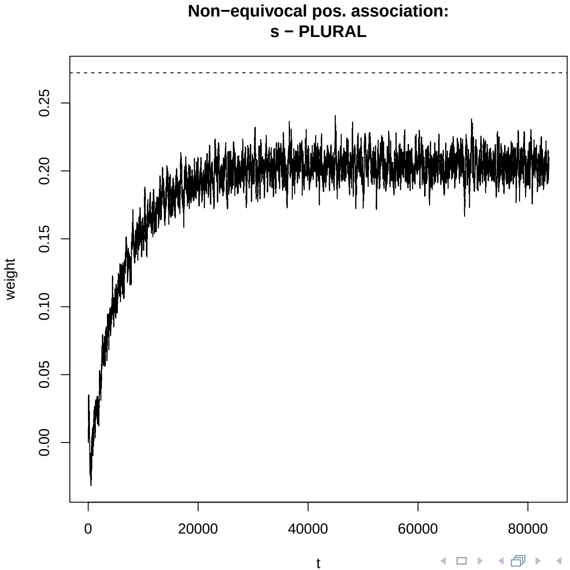
Convergence vs. non-convergence – artificial data

word form	frequency	outcomes	cues
hand	10	hand_NIL	h_a_n_d
hands	20	hand_PLURAL	h_a_n_d_s
land	8	land_NIL	l_a_n_d
lands	3	land_PLURAL	l_a_n_d_s
and	35	and_NIL	a_n_d
sad	18	sad_NIL	s_a_d
as	35	as_NIL	a_s
lad	102	lad_NIL	l_a_d
lad	54	lad_PLURAL	l_a_d
lass	134	lass_NIL	l_a_s_s

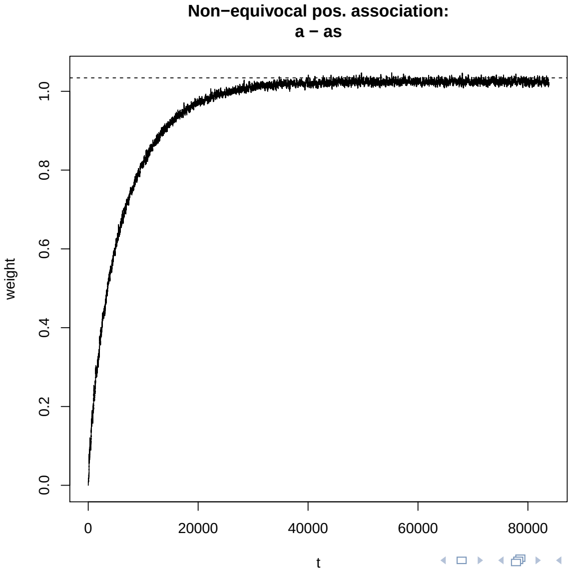
Perfect positive association ➜ convergence



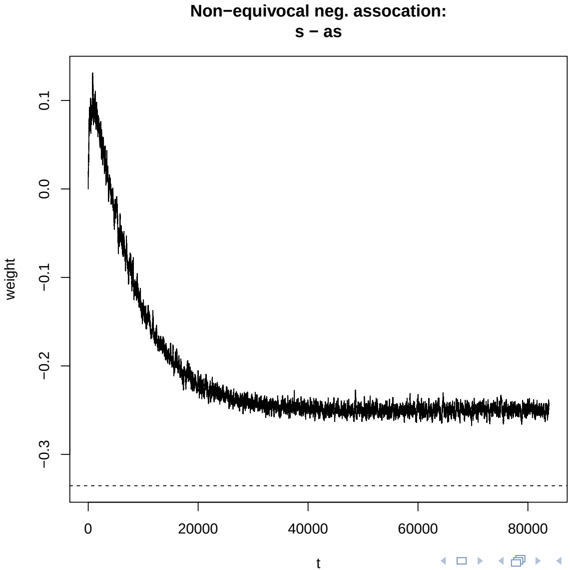
Moderate positive association ➜ non-convergence



Perfect positive association ➜ convergence



Moderate negative association ➜ non-convergence



Outline

1. Introduction
   - Naïve Discriminative Learning
   - An example
2. Mathematics
   - The Rescorla-Wagner equations
   - The Danks equilibrium
   - NDL vs. the Perceptron vs. least-squares regression
3. Insights
   - Theoretical insights
   - Empirical observations
   - Conclusion

Summary & next steps

$$
\boxed{\begin{array}{ccccc} \text{stochastic} & = & \text{batch} & = & L_2 \text{ regression} \\ \text{NDL} & = & \text{SLP} & & \end{array}}
$$

Summary & next steps

$$\begin{array}{ccccc} \text{stochastic} & = & \text{batch} & = & L_2 \text{ regression} \\ \text{NDL} & = & \text{SLP} & & \end{array}$$

- How many training steps are needed for a stochastic NDL learner to converge to the Danks equilibrium?

Summary & next steps

stochastic	=	batch	=	L_2 regression
NDL	=	SLP		

- How many training steps are needed for a stochastic NDL learner to converge to the Danks equilibrium?
- Are there cases of non-convergence? If yes, why?

Summary & next steps

stochastic	=	batch	=	L_2 regression
NDL	=	SLP		

- How many training steps are needed for a stochastic NDL learner to converge to the Danks equilibrium?
- Are there cases of non-convergence? If yes, why?
- Does NDL accuracy always improve with more cues and more training data? If not, why?

Summary & next steps

stochastic	=	batch	=	L_2 regression
NDL	=	SLP		

- How many training steps are needed for a stochastic NDL learner to converge to the Danks equilibrium?
- Are there cases of non-convergence? If yes, why?
- Does NDL accuracy always improve with more cues and more training data? If not, why?
- How does logistic regression behave as incremental learner?

Summary & next steps

$$\boxed{\begin{array}{ccccc} \text{stochastic} & = & \text{batch} & = & L_2 \text{ regression} \\ \text{NDL} & = & \text{SLP} & & \end{array}}$$

- How many training steps are needed for a stochastic NDL learner to converge to the Danks equilibrium?
- Are there cases of non-convergence? If yes, why?
- Does NDL accuracy always improve with more cues and more training data? If not, why?
- How does logistic regression behave as incremental learner?
- Which sequences / patterns in the input data lead to significantly different behaviour from stochastic learner?

Acknowledgements 1/2

Follow me on Twitter: @RattiTheRat

The mathematical analysis was fuelled by large amounts of coffee and cinnamon rolls at Cinnabon, Harajuku, Tokyo

Acknowledgements 2/2

The empirical analyses were conducted in the natural environment of Ninase, Saaremaa, Estonia.

References I

Arppe, Antti and Järvikivi, Juhani (2002). Verbal synonymy in practice: Combining corpus-based and psycholinguistic evidence. Presentation at the Workshop on Quantitative Investigations in Theoretical Linguistics (QITL-1).

Arppe, Antti and Järvikivi, Juhani (2007). Every method counts: Combining corpus-based and experimental evidence in the study of synonymy. *Corpus Linguistics and Linguistic Theory*, **3**(2), 131–159.

Arppe, Antti; Hendrix, Peter; Milin, Petar; Baayen, R. Harald; Shaoul, Cyrus (2014). *ndl: Naive Discriminative Learning*. R package version 0.2.16.

Baayen, R. Harald (2011). Corpus linguistics and naive discriminative learning. *Brazilian Journal of Applied Linguistics*, **11**, 295–328.

Baayen, R. Harald; Milin, Petar; Đurđević, Dusica Filipović; Hendrix, Peter; Marelli, Marco (2011). An amorphous model for morphological processing in visual comprehension based on naive discriminative learning. *Psychological Review*, **118**(3), 438–81.

Danks, David (2003). Equilibria of the Rescorla-Wagner model. *Journal of Mathematical Psychology*, **47**, 109–121.

Dawson, Michael R. W. (2008). Connectionism and classical conditioning. *Comparative Cognition & Behavior Reviews*, **3**.

References II

Gluck, Mark A. and Bower, Gordon H. (1988). From conditioning to category learning: An adaptive network model. *Journal of Experimental Psychology: General*, **117**(3), 227–247.

Rescorla, Robert A. and Wagner, Allen R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. H. Black and W. F. Prokasy (eds.), *Classical Conditioning II: Current Research and Theory*, chapter 3, pages 64–99. Appleton-Century-Crofts, New York.

Rosenblatt, Frank (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, **65**(6), 386–408.

Stone, G. O. (1986). An analysis of the delta rule and the learning of statistical associations. In D. E. Rumelhart and J. L. McClelland (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, chapter 11, pages 444–459. MIT Press, Cambridge, MA.

Sutton, Richard S. and Barto, Andrew G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, **88**(2), 135–170.

Widrow, Bernard and Hoff, Marcian E. (1960). Adaptive switching circuits. In *IRE WESCON Convention Record*, pages 96–104, New York. IRE.