

Essays in Behavioural Economics

Inaugural-Dissertation
zur Erlangung des Grades eines Doktors
der Wirtschafts- und Gesellschaftswissenschaften
durch die
Rechts- und Staatswissenschaftliche Fakultät
der Rheinischen Friedrich-Wilhelms-Universität
Bonn

vorgelegt von
Mirko Seithe
aus Freckenhorst

Erscheinungsjahr 2011

Dekan: Prof. Dr. Klaus Sandmann
Erstreferent: Prof. Dr. Armin Falk
Zweitreferent: Prof. Dr. Sebastian Kube

Tag der mündlichen Prüfung: 13.12.2011

Diese Dissertation ist auf dem Hochschulschriftenserver der ULB Bonn
elektronisch publiziert.

Acknowledgements

This dissertation documents the results of the four research projects I have been working on in the past years. The projects presented herein were heavily influenced by colleagues and friends who gave support and inspiration throughout and to which I am very grateful.

First of all, I would like to thank my supervisor Armin Falk who inspired the projects presented in this document and taught me the practices of good and solid scientific work. Despite his filled schedule he always found the time to provide support for my projects and spawn new inspiration and ideas.

I would like to thank Urs Fischbacher and his research group at the Thurgauer Wirtschaftsinstitut (TWI) for both inviting and welcoming me to present my work. Despite the heresy I have committed, Urs not only sincerely supported me with my project but even agreed to become my referee, for which I am very grateful.

I am greatly indebted to the whole experimental economics research community in Bonn, in particular the participants of the experimental workshops and topics courses, who allowed me to present and discuss my work and provided a huge amount of valuable ideas in a friendly and constructive environment.

I am grateful for the financial support provided by the German National Research Foundation (DFG), which made this dissertation possible, as well as the Bonn Graduate School of Economics (BGSE) for providing interesting and challenging courses and seminars and a very supportive environment. I would like to thank all of my interesting and unique colleagues of the BGSE who have become great friends and never failed to provide fun and support.

Most importantly I would like to thank my family and friends for encouraging me during my dissertation process, especially Janetta, whose great support will never be forgotten.

Contents

Introduction	14
1 Locus of Control and Political Preferences	23
1.1 Introduction	23
1.2 Related Work	24
1.2.1 Locus of Control and Effort Provision	26
1.2.2 Locus of Control and Policy Preferences	26
1.3 Data	27
1.3.1 German Socio-Economic Panel and Locus of Control	27
1.3.2 The Locus of Control Variables	28
1.3.3 Party Preferences	30
1.4 Results	32
1.4.1 Interest in Politics and Voting Behaviour	32
1.4.2 Left-Right Voting and Locus of Control	35
1.4.3 Party Preferences	38
1.4.4 SPD, CDU and CSU	39
1.5 Effect of Changes in Locus of Control	45
1.5.1 Income and General Locus of Control	45
1.5.2 Detailed Locus of Control	47
1.6 Conclusion	47
1.A Appendix	49

1.A.1	Locus of Control Questions	49
2	Endogenous Work Norms	51
2.1	Introduction	51
2.2	Related Literature	52
2.2.1	Payment Schemes and Tournaments	53
2.2.2	Social Norms	55
2.2.3	Experimental Evidence for Social Norms	56
2.3	Experiment Design	58
2.3.1	Overview and Time Line	59
2.3.2	Real Effort Task	60
2.3.3	Treatments	61
2.3.4	Subject Pool	62
2.4	Theoretical Predictions	63
2.4.1	Selfish Optimum	63
2.4.2	Social Optimum	64
2.4.3	Individual Optimum with Norms	66
2.4.4	Summary	68
2.5	Results	69
2.5.1	Time and Performance	70
2.5.2	Regression Results	71
2.5.3	Differences in the Subject Pool	72
2.5.4	Individual Performance Differential	73
2.5.5	Subjective Perception of the Task	74
2.5.6	Norm Perception	75
2.6	Manager Stage	77
2.6.1	Design	77
2.6.2	Results	78
2.6.3	Summary	80
2.7	Conclusion	80

2.A	Instructions for the Main Experiment	82
2.A.1	Instructions Pretest	82
2.A.2	Getting-to-know-Task	83
2.A.3	Instructions Main Part	84
2.A.4	Norm Questions	88
2.A.5	Declaration Together	89
2.A.6	Declaration Separated	89
2.B	Manager Experiment	90
2.B.1	Instructions Pretest	90
2.B.2	Instructions Main Part	91
2.B.3	Main Computerised Part	95
3	Framing and Motivation	105
3.1	Introduction	105
3.2	Related Work	106
3.2.1	Motivational Crowding-Out	106
3.2.2	Feedback and Implied Control	107
3.2.3	Praise and Blame	108
3.2.4	Graphical Social Cues	108
3.3	Design	109
3.3.1	Procedure	109
3.3.2	Experiment Design	110
3.3.3	Treatment Groups	111
3.4	Model and Hypotheses	112
3.5	Results	114
3.5.1	Data	114
3.5.2	Influence of Positive and Negative Reinforcement	115
3.5.3	Influence of Field of Study and Time of Participation	117
3.5.4	Estimating Pay-Off and Cost Functions	118
3.5.5	Analysing the Influence of Feedback	121

3.5.6	Effect for Low- and High-Ability Agents	122
3.6	Discussion and Conclusion	123
3.A	Screenshots of the Experiment	126
3.A.1	Introductory Text and Questionnaire	126
3.A.2	Practice Table	127
3.A.3	Main Experiment	128
3.B	Positive and Negative Reinforcement	129
3.B.1	Positive Reinforcement	129
3.B.2	Negative Reinforcement	129
4	Bonn Experiment System	131
4.1	Introduction	131
4.1.1	Related Work	132
4.1.2	Introducing the Bonn Experiment System	134
4.1.3	Outline	136
4.2	Using the BoXS	137
4.2.1	Quick Start Tutorial	137
4.2.2	Starting an Experiment	140
4.2.3	The Experimenter View	141
4.2.4	Internet Experiments	143
4.2.5	Laboratory Experiments	143
4.2.6	Using an Offline Server	144
4.2.7	Autorun Experiments	145
4.2.8	Troubleshooting	146
4.2.9	Documentation	146
4.3	The BoXS Programming Language	147
4.3.1	Code Based and Graphical Approaches	147
4.3.2	Program Execution	148
4.3.3	Implemented Functionality	152
4.3.4	Basic Calculus	153

4.3.5	Variables	154
4.3.6	Displaying Information and User Input	157
4.3.7	Matching	161
4.4	Design and Implementation	162
4.4.1	Programming Language	163
4.4.2	Network Architecture	164
4.4.3	Communication Protocol	165
4.4.4	The Server	167
4.4.5	The Client	171
4.5	Limitations and Future Development	175
4.5.1	Feature Selection	175
4.5.2	Future Development	175
4.5.3	Limitations	176
4.6	Conclusion	176
4.A	List of all Functions in the BoXS Programming Language	178
4.A.1	Basic Operations and Calculations	178
4.A.2	Program Flow Control	179
4.A.3	Displaying Text and Graphics	182
4.A.4	Waiting	183
4.A.5	User Input	185
4.A.6	Matching	191
4.B	Example BoXS Programs	196
4.B.1	Questionnaire	196
4.B.2	Public Good Game	196
4.B.3	Chat Client	197
4.B.4	Dutch Auction	198
4.B.5	Localization	199
4.B.6	Real Effort Task	199

Literature

199

List of Figures

1.1	Locus of control components.	29
1.2	One-dimensional political attitude of voters.	36
1.3	Locus of control and parties 1.	40
1.4	Locus of control and parties 2.	41
1.5	Estimated relation between beliefs and party preference.	44
1.6	Reaction of party preferences to changes in locus of control.	46
2.1	The real effort task.	61
2.2	Overall distribution.	70
2.3	Distribution by treatment.	71
2.4	Performance differential between Bonn and Frankfurt.	73
2.5	Time controlled for individual factors.	74
2.6	Enjoyment of task.	75
2.7	Perceived norms.	76
2.8	Payment chosen by subjects.	79
2.9	Social interaction chosen by subjects.	79
3.1	The real effort task.	110
3.2	Examples for reinforcement types.	111
3.3	Time spent on the main task and pay-off (polynomial fit).	115
3.4	Performance by time of participation and field of study.	117
3.5	Performance by treatment.	118

3.6	Estimated pay-off and cost functions.	120
3.7	Number of tables solved by treatment and ability.	124
4.1	A simple questionnaire in the Bonn Experiment System.	135
4.2	The starting page.	140
4.3	The 'Available Setups' page.	141
4.4	The experimenter view.	142
4.5	Possible applications for the Bonn Experiment System.	144
4.6	Class diagram of the lexing/parsing classes.	150
4.7	Example questionnaire.	152
4.8	List of all functions implemented in the BoXS.	153
4.9	Examples using the <code>display</code> -command.	158
4.10	More examples using the <code>display</code> -command.	159
4.11	Class diagram of the main server classes.	170
4.12	The BoXS client.	172
4.13	Class diagram of the main client classes.	174

List of Tables

1.1	Sample validity.	31
1.2	Locus of control and intention to vote.	34
1.3	Locus of control and one-dimensional political preference.	37
2.1	The payment schemes.	62
2.2	Overview of the sessions.	62
2.3	Summary of theoretical predictions.	69
2.4	Regression results.	72
3.1	Treatment groups.	112
3.2	Regression results for number of tables.	116
3.3	Influence on fun, tediousness and estimated probability.	121
3.4	Number of tables solved by treatment and ability.	123
4.1	Local, group and global variable examples.	155
4.2	Array and matrix examples.	155

Introduction

The goal of behavioural economics is to improve classic microeconomic theory by introducing motives and concepts from related fields like psychology and sociology.

The driving paradigm of most neo-classical economic research is the concept of the Homo Oeconomicus, a human who approaches all problems in a rational and typically selfish way and who possesses boundless computational power and flawless reasoning. Despite the obvious oversimplification, the given assumptions allow the precise analysis of a large number of complex problems and have led to many interesting and often surprising findings and theories.

While the value of constructing theoretical economic models is beyond doubt, it is important to be aware that the simplifying assumptions made within limit the scope of the predictions made. The assumption that perfectly reasonable people interact in a strictly logical way often leads to conclusions which bear no resemblance to real-world observations.

The role of behavioural economic research is not to abandon theoretical research but to question and test the assumptions made by economic models, to identify contradictions to actual observations when they occur and to develop alternative models to capture apparent flaws in the models, or, as one might argue, flaws in human behaviour.

Examples for such flaws include loss aversion¹ and non-exponential discounting² which, despite being irrational from a theoretical perspective, seem to be prevalent themes in human behaviour. Social preferences play a role

¹Numerous experiments have shown that people value possible losses stronger than they value possible gains. For example, most people would decline a lottery in which they could earn or lose one Euro with equal probability. For a very detailed and exhaustive analysis of risk-preferences in Germany see Dohmen et al. (2005).

²Many experiments indicate that people seem to have difficulties in properly evaluating future gains and losses. Instead of using the mathematically correct exponential discounting, they tend to act in a myopic way, which is often referred to as hyperbolic discounting. For more information see for example Ainslie and Haslam (1992).

when people interact and social norms cause them to behave in a nice way when treated well or to reciprocate and punish their counterpart even at their own expense³.

Furthermore humans have difficulties when dealing with complex problems, which is referred to as bounded rationality⁴. People tend to make calculation mistakes, use rough approximations and imprecise simplifications when facing difficult problems⁵.

The first three chapters of this dissertation cover three different topics tied to behavioural economics. They connect concepts originating from psychology and sociology like intrinsic and extrinsic motivation and the so-called locus of control and apply them to microeconomic problems like the optimal effort provision in a principal-agent setting. The fourth chapter is strongly related to computer science as it describes the development of a computer system intended to simplify the design and conduction of economic experiments. While it is the project most distant to economics, it is arguably also the most ambitious of the four projects.

The remainder of this introduction includes short summaries of each of the four chapters.

Chapter 1 Many decisions in life are based on a person's belief in how much influence and control she has on her own life and her environment. A person failing an important test, for example, can perceive this as bad luck and fault of an ill-meaning teacher on the one hand, or as the result of a lack of learning effort or ability on the other hand. Both ways of interpretation can be subjectively justified to a certain degree and thus a large heterogeneity

³For more information on reciprocity see Fehr and Gächter (2000), Falk (2003) and Falk and Fischbacher (2006).

⁴For more information on bounded rationality see Rubinstein (1998).

⁵For example people tend to break complex problems into simple problems which they can solve individually. Unfortunately, this often leads to flawed results which is referred to as narrow bracketing. For more information see Read et al. (1999), Fehr et al. (2002) and Rabin and Weizsäcker (2009).

of this so-called locus of control belief exists within the population. This belief is likely to have a strong impact on peoples' lives as a person who believes that studying hard is guaranteed to benefit her in the future, for example, will study much harder than a person who believes that her ticket in the lottery of life has already been drawn.

Besides the heterogeneity in the locus of control belief within a population, there is a surprising difference between countries. Americans in particular have a much stronger 'belief in a just world' than Europeans despite the fact that the underlying environments are quite similar. In their popular paper on the subject Benabou and Tirole suggest a circular relationship between the peoples' beliefs and the society they inhabit which supports the between-country differences. The key idea is that people who believe that everyone is responsible for her own life will favour laws and policies which reduce the influence of the state and social institutions. In the resulting society, in which taxes are low and social security is loose, individual effort yields a high dividend, which can justify the initial beliefs as a way of motivation.

In the first chapter I use data from the German Socio-Economic Panel (GSOEP) to analyse the effect suggested by Benabou and Tirole that, *ceteris paribus*, a person who believes in a just world and an individual's influence on her life will favour parties who aim for a lower level of redistribution. I find that the locus of control has a strong and significant effect on a person's voting behaviour. In fact, the influence is even stronger than the effect of income and education, which are often considered as the main determinants for political preferences.

The results provide a new perspective to analysing political preferences, which reveals similarities between the political parties beyond the classical left-right scheme. On the one end of the spectrum, voters of the German liberal and green parties are, in spite of their parties' seemingly contradictory agendas, similar in their belief of an individual's influence on her life and her

surroundings. On the opposing end, the voters of far-left and far-right parties are surprisingly similar in their belief of their lives being controlled by others.

Chapter 2 Chapters two and three of this dissertation contribute to the research on so-called principal-agent relationships. These relationships are characterised by a principal who pays and employs an agent in order to work on a given task. As putting effort in the task is costly, an agent will, from a classical perspective, work as little as possible whenever she is not observed and does not have to fear any consequences. The principal faces the problem that observing the agent is both costly and often impractical. In order to motivate the agent to perform well she has to create an effective incentive system and a rewarding work environment⁶.

In the 'real world' many firms offer competitive payment schemes in which employees compete over bonuses, promotion chances and, in the worst case, keeping their jobs. While such incentive systems create strong incentives to exert effort at first glance, problems arise as cooperation and team work suddenly become detrimental to individual success and selfishness or even acts of sabotage are rewarded. Furthermore, social norms might play a role as socially aware agents might reduce their effort in order to prevent hurting their co-workers.

Previous work done by Bandiera et al. (2005) suggests that the latter effect is indeed prevalent. In a field study they find that introducing a relative payment scheme causes the observed fruit farmers, who are often good friends, to apply strong social pressure in order to prevent each other from performing too well.

Chapter two, which is based on joint work with Armin Falk, Leonie Gerhards and Michael Kosfeld, analyses the effects of social norms in the principal-agent setting. We analyse this effect by conducting a controlled

⁶For an analysis of the economic principal-agent problem see Grossman and Hart (1983).

laboratory environment using an incentivised real effort task in which we vary both the payment scheme and the level of social interaction.

We find that both social interaction and payment system have a significant effect on effort provision. Furthermore the results suggest that, given a certain level of social interaction, different payment systems may be optimal. In a close relationship, for example, team based cooperative payment schemes seem more advantageous whereas in anonymous environments competitive schemes may yield the best performance. The results derived from this chapter are suggestive and question many of the very competitive incentive systems which can be observed in the corporate world.

Chapter 3 In a typical principal-agent relationship an agent produces both successes and failures and the principal has to decide whether and how she should express her approval or disapproval. Many principals use explicit praise as a way of motivation. While such positive feedback might serve as an external reward, it also communicates to the agent that she is being supervised and not in control herself. This might hurt both the agent's intrinsic motivation of working on the task as well as her willingness to cooperate⁷. The use of blame seems likewise questionable as its potentially negative effect on the agent's motivation might outweigh the intended effect of discouraging mistakes.

In chapter three I analyse the straightforward question of how a principal should frame positive or negative feedback. I approach this question by conducting an internet experiment in which I vary the type of feedback subjects receive after success and failure and observe the productivity within an incentivised real-effort task. As a proxy for personal feedback a simple abstract smiley is displayed, which is happy upon success and sad upon failure, as well as a few other positive and negative images.

⁷For example, Falk and Kosfeld (2006) show that exerting control over an agent's decision can significantly decrease cooperation.

From a neo-classical microeconomic perspective the framing should have no tangible effect as the underlying effort provision problem is not affected. However, the behavioural perspective suggests two ways in which reinforcement could alter a subject's decision. First, both the costs of working on a task and the pleasure derived from solving it are largely subjective and might be influenced by praise and blame (hedonic effect). Second, subjects learn about their ability to solve the task and reinforcement might influence the learning process by magnifying the perception of positive or negative events (computational effect).

Despite the very small treatment variation I find that providing reinforced feedback causes surprisingly strong and significant effects. Especially negative reinforcement seems to have a strong effect and reduces subjects' performance by up to 20%. Positive reinforcement has an opposing, although much smaller effect and increases performance by about 5%. Further analysis reveals that the main effect driving these results is a change in the perceived fun and tediousness of working on the task. Finally, I find that high- and low-ability subjects react in a vastly different way to positive feedback.

The results from this chapter strongly advise against providing and reinforcing negative feedback whenever feasible.

Chapter 4 The previous two chapters analyse questions which are approached using experiments. The use of controlled laboratory experiments has become a vital part in behavioural economics and many other areas of research. Well-designed laboratory experiments make it possible to break down a complex problem in a way that allows it to be analysed in a clean and precise way. They provide control over the information a subject receives as well as the incentive system and allow for exogenous treatment assignment⁸.

⁸For more information on the advantages of laboratory experiments in general and in the context of economics see Falk and Fehr (2003) and Falk and Heckman (2009).

While classic laboratory experiments are conducted using pen-and-pencil methods, the use of computers is on the rise. Computerised experiments offer both theoretical advantages, like reducing the interaction between subject and experimenter, as well as practical advantages like greatly simplifying and accelerating the experiment conduction and the data collection process. Furthermore many experiments, for example market and auction simulations, provide a level of complexity and interaction which cannot reasonably be dealt with otherwise.

In the recent years both computers and networks have become vastly more powerful, reliable and versatile and most people in developed countries have acquired access to the internet. Likewise, so-called multimedia capabilities have vastly improved, allowing people to watch high-quality video, listen to sound or access other types of complex information.

The Bonn Experiment System (BoXS), which I will describe in chapter four, is arguably the most ambitious project approached in the process leading up to this dissertation. The BoXS is a software system allowing experimenters to easily design and conduct both laboratory and internet experiments and takes advantage of modern design architectures.

From a practical perspective, the BoXS allows experimenters without prior programming experience to design their own experiments in a relatively easy way. The implemented BoXS Programming Language is easy to learn due to a comprehensive documentation and allows simple experiments to be implemented using only a few lines of code.

From a technical perspective, the BoXS is based on HTML, CSS and Java. All the underlying technologies are platform independent and allow the BoXS to be executed on a large variety of devices, which was verified on different computers using the most popular operating systems Linux, MacOS and Windows. Furthermore, as mobile devices like phones and tablets become more powerful and popular, it is likely that they will be able to

support the BoXS in the future.

Since its conception as a simple prototype, the BoXS has been continually developed, enhanced and presented to different scientific communities. It has since been used in several actual laboratory and internet experiments to great success. Chapter four describes the basic ideas and design principles driving the development of the system, describes how the system works, offers some guidelines for experimenters using it and provides a technical documentation of its inner life.

Chapter 1

Locus of Control and Political Preferences

1.1 Introduction

The causes of many real life events can be interpreted in different ways. The exact same outcome, for example becoming unemployed, can be understood as the result of lacking effort or ability by one person, while it is perceived as bad luck or society's fault by another person. An individual's preference to blame events on internal or external factors is called the 'locus of control' in psychology literature.

A person's locus of control is likely to be based on observing and interpreting life events as well as shaped by parents, peers and the society in general. While an individual's resulting locus of control belief might be biased, one would expect the average locus of control of a population to converge to a similar level for all industrial countries. Interestingly, this does not seem to be the case. Despite the similarity between European countries and the USA, significant differences between it's peoples' locus of control seem to persist.

Benabou and Tirole propose a model in which they conjecture how dif-

ferent levels of locus of control can coexist. First, they argue that having an internal locus of control and believing that effort pays off is vital for success as it provides the necessary motivation to study and work hard. Second, having an internal locus of control might lead people to prefer a state with lower taxes and less social security as failure is seen as individual fault. If these preferences are widespread and result in a leaner state, individual effort becomes more important, which makes an internal locus of control more viable. The effect resulting from this cycle can explain persistent differences in peoples' locus of control when compared on a cross-country level.

In this paper I use the German Socio-Economic Panel, which provides a rich data set from representatively sampled German households, to estimate how differences in locus of control influence a person's voting behaviour. Using regression analyses I find that the locus of control has a significant and strong effect on a person's political preferences as well as her voting behaviour, which supports the assumptions made by Benabou and Tirole.

In the upcoming section I summarise related work on this subject. In section 1.3 I describe the data and the questions used in the survey. In section 1.4 I analyse the relation between locus of control and the likelihood to vote as well as party preferences. In section 1.5 I use the previously estimated models to make predictions about the likely political effects which would result from changes in locus of control. Section 1.6 summarises the results of this chapter.

1.2 Related Work

Several studies, for example Alesina et al. (2001), have shown that significant differences in the locus of control between different countries exist, most notably between the US and European countries. US citizens have a significantly stronger belief in the effect of their own actions than people in European countries. It is hard to imagine that this difference is based

entirely on real world differences.

Benabou and Tirole (2006) introduce a model in which they show how different levels of belief in a just world can coexist over time. The belief in a just world, in which hard work pays off and crime does not pay, is closely related to having an internal locus of control, which is the topic of this paper. The model of Benabou and Tirole assumes that people find it very hard to motivate themselves to do the unpleasant but necessary things in their life like working or getting education. If this is the case and people suffer from self-control problems, a realistic view of the world might yield an ineffectively low level of effort. In this case overestimating the effect of their own effort, despite signals to the opposite, might be a way of people tricking themselves into working harder and gaining an ex-post superior level of utility.

Furthermore Benabou and Tirole argue that the level of belief in a just world is likely to have an effect on society. If people think that a world is just in that effort pays off in success then poor people are seen as being at fault for their own failure. In effect peoples' support for redistributive policies is likely to decrease in favour of more market-oriented, laissez-faire policies. If this shift in policy preferences leads to lower tax rates and less social securities then the objective incentive to work hard increases. This in turn leads to a higher level of optimal effort, thus supporting the original biased beliefs.

The authors propose three ways in which people might sustain the difference between their belief and reality despite signals to the opposite. First, they argue that 'systematic brainwashing' might be done by people benefiting from laissez-faire policies. Second, individuals might settle for imprecise and incorrect beliefs if learning is costly. Third, people may simply prefer the belief in a just world as it is a more comforting concept.

1.2.1 Locus of Control and Effort Provision

Intuitively, differences in the locus of control are likely to have a strong influence on individual decision making. The effort exerted when searching for a new job, for example, is likely to be much lower if an individual thinks her effort does not effect the outcome and getting a good job is purely a matter of luck.

This intuition is supported by a study by Andrisani and Nestel (1976), who conduct a panel study in which they measure the locus of control of about 3000 respondents and observe how variables like job satisfaction and earnings develop afterwards. They find that a more internal locus of control induces a higher level of effort which is correlated to wage increases. In summary, this study supports both the influence of locus of control on effort, as well as its effect on the actual outcome which is in line with the assumptions of the model by Benabou and Tirole.

1.2.2 Locus of Control and Policy Preferences

Benabou and Tirole suggest that having an internal locus of control leads to a preference for lower taxes and less redistribution. Several publications analyse the relation between beliefs and preferences both on a cross-country level as well as on an individual level.

Alesina et al. (2001) analyse the influence of beliefs about the causes of poverty on political preferences. They find that people who believe that poverty is an effect of bad luck and not laziness are more likely to consider poor people worthy of monetary gifts and are generally more likely to be on the left side of the political spectrum. These effects were shown to be significant in cross-country comparisons.

Fong (2001) uses results from the Gallup Poll and analyses the support for redistribution. He finds that, unlike economic theory would predict, a person's own income is a surprisingly poor indicator for her support for

redistribution. Instead, a person's belief in the influences of effort, luck and opportunities in her life seem to be pivotal. This implies that monetary inequality does not necessarily lead to a strong demand for redistribution as long as people feel they are in control of their life.

A comparable analysis has been done by Alesina and Giuliano (2009), who use data from the US General Social Survey. They also find that other factors beside income like gender, age, education and religion play an important role to explain support for redistribution. They also find that traumata which occurred recently have a very strong influence.

Rainer and Siedler (2008) use the German Socio-Economic Panel and find that the preference for redistribution is not primarily determined by a person's income but by the perceived social mobility.

1.3 Data

1.3.1 German Socio-Economic Panel and Locus of Control

The German Socio-Economic Panel is a yearly recurring questionnaire of people living in representatively sampled German households, which is organized by the German Institute for Economic Research (DIW) and conducted by Infratest¹.

The most in-depth analysis of locus of control using the German Socio-Economic Panel so far has been done by Nolte et al. (1997), who do a vast analysis of the answers to the locus of control question block in the 1994 wave. They briefly analyse political preferences and find that people with a more internal locus of control are more likely to vote for the liberal party (FDP) while people with external locus of control tend to vote for the left party (PDS). This is in line with the results I find in this chapter. It should be noted, however, that the results from Nolte et al. are mere correlations

¹For more information see Wagner et al. (2007).

and do not account for income, gender and other factors which are likely to play a role for predicting party preference.

Since the 1994 wave the questions in the questionnaire concerning locus of control have been changed significantly, probably partly as a result of the fact that Nolte et al. point out several problems with the original questions. For example, they argue that some wording issues may have interfered with the questions as they cannot be arranged in a Guttman-scale as originally intended². Another important difference is that the 2005 wave of the questionnaire includes questions for the four specific components of locus of control, namely effort, ability, luck and the influence of society, which allows for the more in-depth analysis I do in this paper.

1.3.2 The Locus of Control Variables

In the 2005 wave of the questionnaire participants were asked to signal their agreement to 10 statements concerning their locus of control on a 7 point Likert-scale³. At first glance the locus of control can be viewed as a one-dimensional concept. In order to measure this general locus of control I use the subjects' agreement to the statement "My life's course depends on me". A person who has an internal locus of control would agree to this statement whereas a person with an external locus of control would disagree.

On a more detailed level one can distinguish four components determining the locus of control. These factors are luck and other people as external factors, as well as ability and effort as internal factors. In order to measure the first two factors I use the statements "What you can achieve depends on luck" and "Others make the crucial decisions in my life". For the latter ones I use the statements "Abilities are more important than effort" and "Success

²The idea of the Guttman scale, which was originally proposed in Guttman (1950), is to measure a person's agreement to an issue by presenting her with a set of increasingly more provocative statements and measure the point at which she disagrees.

³The Likert scale, the idea of which is to measure a person's agreement to a question not dichotomously but on a gradual discrete scale, was first proposed in Likert (1932).

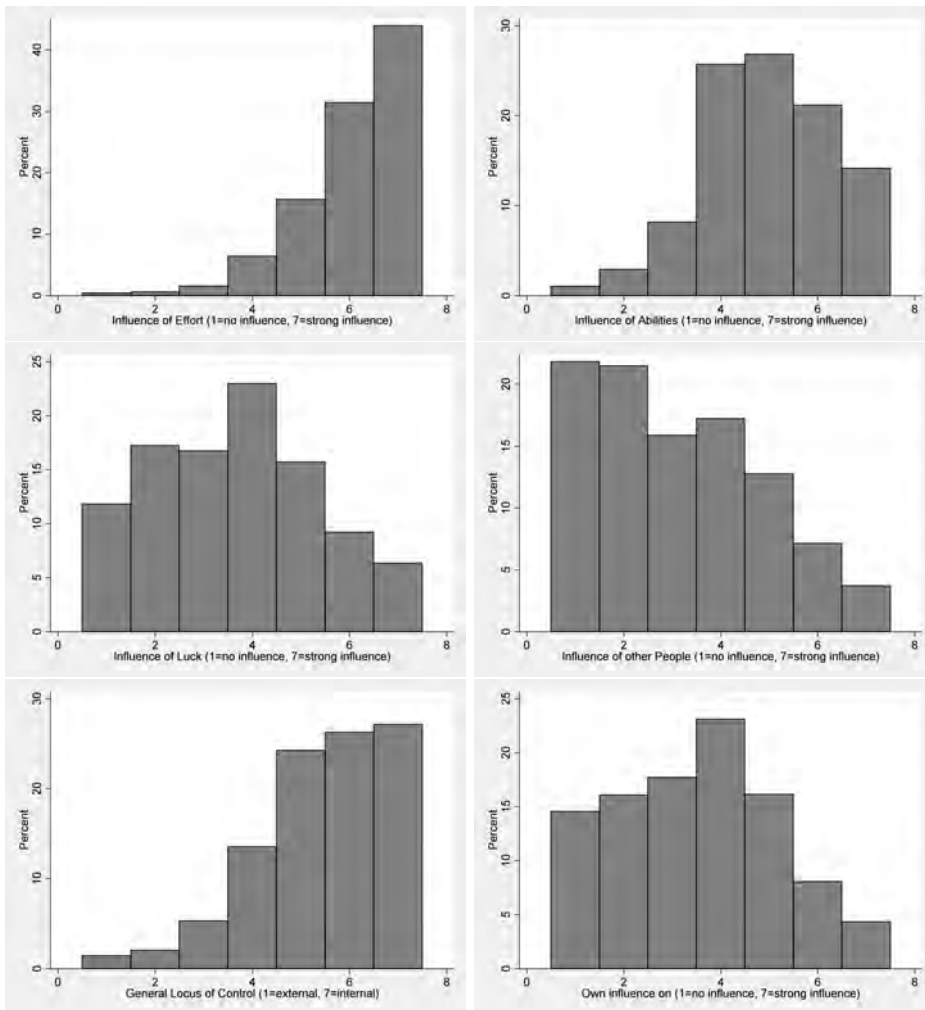


Figure 1.1: Locus of control components.

takes hard work". The distributions for these components are shown in figure 1.1.

The questionnaire also included a statement about how strongly social conditions can be influenced through individual involvement. While this statement does not directly belong to locus of control it is a closely related concept and is therefore included it in the following analysis.

In general it is hard to distinguish if a person is objectively externally controlled or if she is just perceiving her life like that. A person who has poor education or health issues, for example, might be objectively subject to more external control. In order to address this problem I perform multivariate regressions which control for factors which are likely to influence the objective locus of control.

1.3.3 Party Preferences

The question about party preference asked the subjects which political party they favour⁴. There are three possible problems with this question. First, it was possible for the subjects to select multiple parties at the same time. Since this issue is hard to deal with in multinomial models and only applies to 163 out of 21.105 subjects I decided to drop the corresponding observations from the analysis. Second, answering this question was optional and many subjects did in turn not give an answer. This might be a more severe problem as it applies to 11.781 out of 21.105 (56%) observations, especially if this results in a remaining sample which is not representative for the actual electorate. Third, there is no way to incentivise people participating in the survey to reveal their preferences correctly.

The above issues might be problematic if the individual likelihood of revealing one's true preferences is correlated with the favoured party. In order to approach the latter two issues I compare the answers given to this

⁴Note that the question still contained the PDS-party which has merged with the WASG to form the party 'Die Linke' in 2007.

		SPD	CDU/CSU	Grüne	FDP	PDS	Rep/DVU	Others	Total
Observed sample	Frequency	3407	3846	982	400	459	133	88	9315
	Percent	36,9%	41,7%	10,6%	4,3%	5,0%	1,4%	1,0%	
Actual result Bundestagswahl 2005 (Erststimmen)	Frequency	18129100	19280940	2538913	2208531	3764168	896455	1226027	48044134
	Percent	37,7%	40,1%	5,3%	4,6%	7,8%	1,9%	2,6%	
Actual result Bundestagswahl 2005 (Zweitstimmen)	Frequency	16194665	16631049	3838326	4648144	4118194	1014669	842941	47287988
	Percent	33,7%	34,6%	8,0%	9,7%	8,6%	2,1%	1,8%	

This table shows the preferred party as stated by the people within the SOEP sample as well as the actual election results from the Bundestagswahl 2005. Source for election results: www.bundeswahlleiter.de, Statistisches Bundesamt Wiesbaden.

Table 1.1: Sample validity.

question to the actual results of the Bundestagswahl which was held on October 2nd, 2005 to find out if there is a systematic misrepresentation in my sample. A first problem with this comparison is that while the election was held on one day, the questionnaire was done throughout the whole year. A second problem in doing this comparison is that most subjects only selected one party in the questionnaire whereas they have two votes in the German electoral process, which are counted in different ways⁵. In effect complex strategic considerations, which are hard to model, are likely to influence the actual voting decision.

The results of this comparison are shown in table 1.1. I find that the major parties SPD and CDU/CSU as well as B90/Grüne seem to be slightly over-represented while the FDP, PDS, Republicans and other parties seem to be under-represented. The misrepresentation is about 2% to 4% and indicates that the upcoming results for the smaller parties should be interpreted with care.

⁵The Erststimme (first vote) is used in a winner-takes-all format for each district. Since smaller parties are unlikely to win in this way people may be hesitant to make their vote ineffective by voting for them. Hence small parties are likely to be under-represented in the first vote results. The Zweitstimme (second vote) works as proportional vote and is not subject to this issue. However, many voters who favour a certain party combination might vote for a smaller party with their second vote, especially if that party is likely to end up near the five percent hurdle and the respective vote might become pivotal. For more information on this so-called ticket splitting see Pappi and Thurner (2002).

1.4 Results

In the first part of this section I analyse the individual propensity to participate in an election. In the second and third part I analyse individual political preferences on the classical left-right dimension as well as specific party preferences.

1.4.1 Interest in Politics and Voting Behaviour

Interest in politics and the resulting propensity to vote is likely to be determined by three factors. First, a person should be more likely to vote if she feels that the current political agenda is important for her. Second, she should be more likely to vote if she feels she can influence something. Third, she may perceive participating in an election as a social norms, regardless of the actual influence.

My main hypothesis for this analysis is that people who feel that they are unlikely to have an influence on society are less likely to vote. Also, people who think that luck and other people play a strong role should be less likely to vote.

In the SOEP questionnaire people were asked to answer whether they intend to participate in the next election on a 5 point scale between “(will vote) in any case” and “in no case”. For the following analysis I dichotomize the result and recode a person as voter if her intention is greater/equal than 3 and as non-voter if it is below 3. People who were not eligible to participate in an election, for example because they are no German citizens as well as participants who did not answer were excluded from the following analysis (1886 out of 21105 observations).

I do three separate probit regressions with being a voter as the dependent variable. Probit regressions allow the analysis of dichotomous variables and can be estimated using modern statistics software⁶. All three regressions

⁶For more information on the probit model see Finney (1947) and Hausman and Wise

use gender, age, education⁷, income, being born in the former GDR and subjective health as controlling variables.

The results of the regressions are shown in table 1.2. In the first regression I include the general locus of control as variable and find that the corresponding coefficient is positive and highly significant ($p < 0.001$), indicating that people with a more internal locus of control are more likely to vote.

In the second regression I drop the general locus of control and include the more specific locus of control questions. I find that people who believe in the influence of luck and other people on their lives are significantly less likely to vote. At the same time people who believe they have an influence on society are more likely to vote (all $p < 0.001$).

Since it is possible that the locus of control works as a proxy for other personal traits I do a third regression in which I include the Big Five personality measures⁸, which were also elicited in the 2005 wave, as well as a measure for religiousness. I find that while some of the newly introduced variables do have a significant influence, the qualitative results for the locus of control coefficients do not change.

The results seem to be in line with my initial hypotheses. First, people who believe in the influence of external factors, specifically the influence of luck and other people on their lives, are significantly less likely to participate in an election. Second, the perceived influence on social conditions seems to be a strong motivator for people to vote. In summary I can conclude that people with an external locus of control are less likely to vote and may

(1978). For a general overview over the econometric methods used in this and the subsequent chapters see Wooldridge (2002).

⁷It is generally assumed that higher education is correlated to a higher level of support for democratic systems. For a detailed analysis for German citizens see Siedler (2009).

⁸The so-called Five Factor Model, which is popular among psychologists, proposes the factors 'openness to experience', 'agreeableness', 'neuroticism', 'conscientiousness' and 'extraversion' as dimensions of people's personalities. See, for example, Costa Jr and McCrae (1992).

VARIABLES	(1) (Probit)	(2) (Probit)	(3) (Probit)
Gender (male)	0.0449** (0.0206)	0.0330 (0.0210)	0.0463** (0.0220)
Age	0.0307*** (0.00339)	0.0288*** (0.00347)	0.0277*** (0.00352)
Age (power 2)	-0.000119*** (3.49e-05)	-0.000102*** (3.58e-05)	-9.51e-05*** (3.62e-05)
Education: Hauptschule	0.517*** (0.0307)	0.540*** (0.0315)	0.555*** (0.0318)
Education: Realschule	0.807*** (0.0323)	0.790*** (0.0330)	0.788*** (0.0333)
Education: FHR	1.017*** (0.0551)	0.971*** (0.0562)	0.953*** (0.0566)
Education: Abitur	1.334*** (0.0383)	1.286*** (0.0394)	1.264*** (0.0397)
Income (log)	0.0591*** (0.0162)	0.0599*** (0.0166)	0.0616*** (0.0167)
Income missing	0.321*** (0.120)	0.318*** (0.122)	0.325*** (0.123)
Born in GDR	-0.239*** (0.0252)	-0.225*** (0.0258)	-0.202*** (0.0262)
Infl. of luck (1-7)		-0.0568*** (0.00625)	-0.0569*** (0.00631)
Infl. of other people (1-7)		-0.0264*** (0.00592)	-0.0216*** (0.00608)
Infl. of own effort (1-7)		0.0363*** (0.00943)	0.0289*** (0.00991)
Infl. of own abilities (1-7)		0.0214*** (0.00778)	0.0137* (0.00793)
Infl. on soc. (1-7)		0.0963*** (0.00627)	0.0886*** (0.00638)
Goes to church regularly			0.158*** (0.0280)
Health (1=good, 5=poor)	-0.109*** (0.0114)	-0.102*** (0.0117)	-0.0816*** (0.0122)
Big Five: Openness			0.0244*** (0.00305)
Big Five: Conscientiousness			-0.000959 (0.00398)
Big Five: Extraversion			-0.00168 (0.00327)
Big Five: Agreeableness			0.00863** (0.00364)
Big Five: Neuroticism			-0.00803*** (0.00302)
High Income sample (G)	0.384*** (0.0452)	0.349*** (0.0459)	0.334*** (0.0460)
Immigrant sample (D)	0.0749 (0.0543)	0.0611 (0.0554)	0.0417 (0.0558)
General LoC (1=ext., 7=int.)	0.0694*** (0.00703)		
Constant	-1.706*** (0.132)	-1.641*** (0.146)	-1.955*** (0.152)
Observations	20,937	20,523	20,429

Robust standard errors in parentheses

*** p<0.01, ** p<0.05, * p<0.1

Table 1.2: Locus of control and intention to vote.

therefore be under-represented in elections.

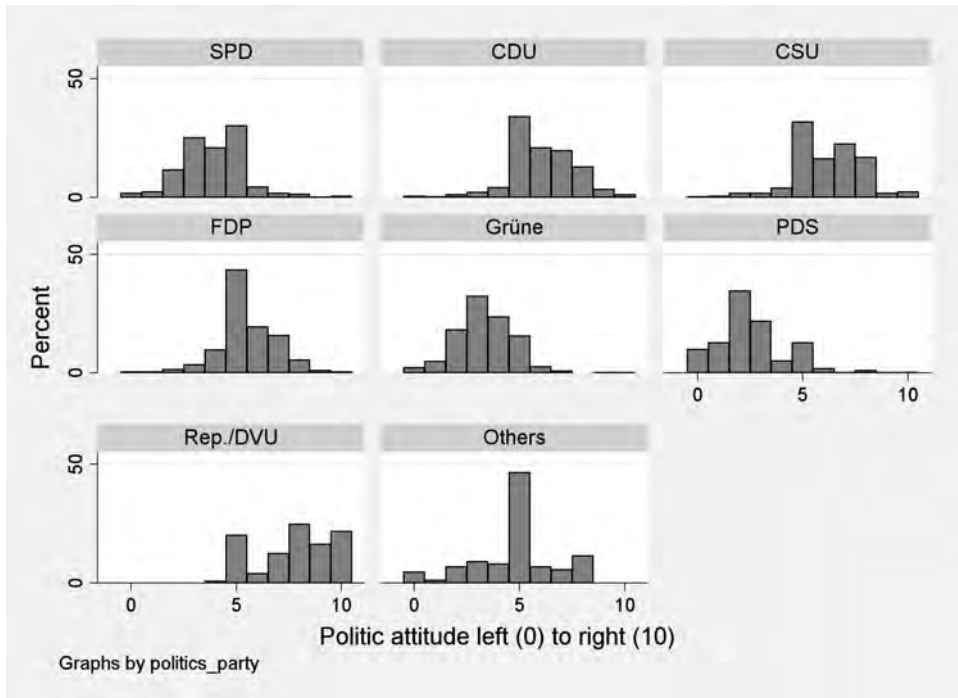
1.4.2 Left-Right Voting and Locus of Control

In this section I analyse individual policy preferences on the classical left-right dimension. The dependent variable in the following regressions is the subjects' reported political attitude on an eleven point scale between left (0) and right (10).

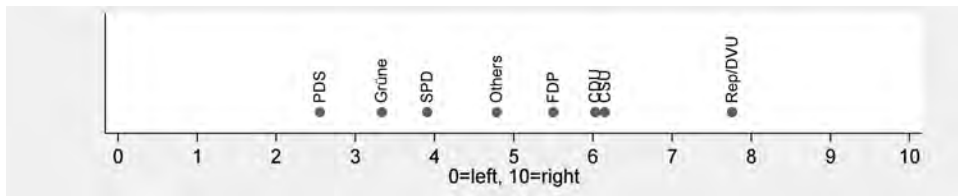
When aligning the German political parties accordingly, the parties favouring more state intervention, i.e. the social democrats (SPD, mean 3.97), the green party (B90/Grüne, mean 3.40) and the left party (PDS, mean 2.83) are considered to be on the left wing while parties like the Christian democrats (CDU/CSU, means 6.05 and 6.17) and the liberals (FDP, mean 5.51) are considered to be on the right, as are the nationalist parties (Rep./NPD/DVU, mean 7.76). The distribution of the left-right preference as given by the participants and grouped by the favoured party is shown in figure 1.2.

Following the literature described in the first section I have two main hypotheses about peoples' preferences for redistribution. First, people who are unemployed or earn only a small wage should favour a higher amount of redistribution as they would benefit from it the most and hence tend to vote for left parties. Second, people who have a more external locus of control should favour redistribution and vote left as they perceive inequality more as an effect of chance and not as an effect of effort.

The answer to this question, which is the dependent variable for the subsequent regressions, is both discrete and ordered. In this special case the use of ordinary least squares regressions is discouraged. Instead, I use ordered probit regressions for the following estimations. The results of the regressions are shown in table 1.3. As in the previous section I begin with a regression including gender, age, education, income, health, GDR and the general locus of control. I find that people with higher education are more



(a) Distribution within parties.



(b) Mean within parties.

Figure 1.2: One-dimensional political attitude of voters.

	(1)	(2)	(3)
	(OProbit)	(OProbit)	(OProbit)
Gender (male)	0.169*** (10.92)	0.174*** (11.11)	0.159*** (9.79)
Age	0.00176 (0.70)	0.000744 (0.29)	-0.00254 (-0.99)
Age (power 2)	0.0000437* (1.70)	0.0000525** (2.03)	0.0000733*** (2.80)
Health (1=good, 5=poor)	-0.0163* (-1.88)	-0.0266*** (-3.00)	-0.0118 (-1.28)
Born in GDR	-0.344*** (-17.98)	-0.363*** (-18.61)	-0.340*** (-17.18)
General LoC (1=ext., 7=int.)	0.0172*** (3.11)		
High Income sample (G)	0.0734*** (2.65)	0.0970*** (3.48)	0.101*** (3.62)
Immigrant sample (D)	0.0713* (1.75)	0.0471 (1.16)	0.0372 (0.91)
Infl. of luck (1-7)		0.0319*** (6.68)	0.0327*** (6.81)
Infl. of other people (1-7)		0.0118** (2.56)	0.0159*** (3.36)
Infl. of own effort (1-7)		0.0546*** (7.75)	0.0565*** (7.66)
Infl. of own abilities (1-7)		0.0140** (2.40)	0.0184*** (3.12)
Infl. on soc. (1-7)		-0.0347*** (-7.37)	-0.0347*** (-7.24)
Goes to church regularly			0.216*** (11.00)
Big Five: Openness			-0.0173*** (-7.48)
Big Five: Conscientiousness			0.0189*** (6.25)
Big Five: Extraversion			0.0138*** (5.57)
Big Five: Agreeableness			-0.0230*** (-8.24)
Big Five: Neuroticism			-0.00859*** (-3.84)
Controls for Education	Yes	Yes	Yes
Controls for Income	Yes	Yes	Yes
Observations	20180	19899	19811

t statistics in parentheses

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

Table 1.3: Locus of control and one-dimensional political preference.

likely to have a leftist political attitude, as are people who were born in the former GDR and people who do not attend church regularly. I also find that women are more likely to vote for left parties than men, which is in line with Schmitt (2000), who analyses voting behaviour in Europe.

I find that people with a more internal locus of control are more likely to vote for parties on the right side of the political spectrum. Note that while the influence of locus of control is strongly significant ($p < 0.001$), the influence of income is not ($p > 0.1$). This result supports the results from Fong, who also finds that income alone is no good predictor for political attitudes.

When introducing the more detailed locus of control measures and dropping the general locus of control, I find that people who believe more strongly in the effect of their own effort ($p < 0.001$) and ability ($p < 0.01$) are more likely to favour parties on the right side, which is in line with my hypotheses. I also find that belief in the influence of luck seems to contribute to voting to the right ($p < 0.001$). These results remain significant when I introduce the big five in the third regression.

In summary I find that people who believe more strongly in internal control factors like effort and ability are more likely to be on the right side of the political spectrum, which is in line with my initial hypotheses. Furthermore income is positively correlated with right-wing-voting in all three regressions as hypothesized, but the effect does not seem to be significant.

1.4.3 Party Preferences

In this section I take a detailed look at the German parties and analyse the individuals voting for them. My goal is to identify the factors which make people prefer one party over the other with special focus on the locus of control. In order to do this I use multinomial logit regressions, which allow

the analysis of nominal dependent variables⁹, with the favoured party as dependent variable. As a baseline I use the SPD since this was the most common choice in the 2005 questionnaire.

First, I look at the major parties in Germany who usually score above 20% in the elections. These are the social democrats (SPD) and the Christian democrats (CDU/CSU¹⁰). Next are the liberal (FDP) and green parties (B90/Grüne), which usually score between 5 and 15%. Finally I look at the socialist (PDS) and nationalistic parties (Rep./DVU/NPD). The regression results for the following analysis are shown in the tables 1.3 and 1.4.

1.4.4 SPD, CDU and CSU

When comparing SPD and CDU/CSU to the other parties, I find that they are strongest with voters who have an average income and average education.

When trying to distinguish between these parties I find that both CDU and CSU are significantly stronger than the SPD with people who have a higher income ($p \leq 0.01$ for CDU and $p \leq 0.1$ for CSU). Both CDU and CSU are also more popular with people who attend church regularly (both $p < 0.001$), which seems reasonable as the Christian parties are generally thought of as the product of the cleavage between Christian and non-Christian citizens. Looking at the age relation I find that the SPD seems to be stronger with middle-aged voters while CDU/CSU are stronger with both younger and older people. For the CSU there also seems to be a gender effect as men are significantly more likely to vote for them as opposed to the SPD ($p < 0.001$). The CDU is significantly stronger with people who were born in East Germany ($p < 0.01$).

When looking at the general locus of control I find that people with a more internal locus of control favour the CDU over the SPD ($p < 0.001$).

⁹For more information on the multinomial logit model see Hausman and McFadden (1984).

¹⁰The CSU can only be voted for in Bavaria, while the CDU can only be voted for in the other federal states.

VARIABLES	(1) CDU	(2) CSU	(3) FDP	(4) Grüne	(5) PDS	(6) Rep./DVU	(7) Others
Gender (male)	0.0365 (0.0550)	0.409*** (0.0936)	0.327*** (0.119)	-0.525*** (0.0857)	-0.00803 (0.115)	1.237*** (0.232)	0.525** (0.237)
Age	-0.0226** (0.00966)	-0.0204 (0.0159)	-0.0300 (0.0197)	0.0681*** (0.0194)	-0.135*** (0.0215)	-0.0391 (0.0533)	0.0603 (0.0512)
Age (power 2)	0.000306*** (9.48e-05)	0.000266* (0.000153)	0.000330* (0.000192)	-0.00102*** (0.000211)	0.00119*** (0.000204)	-0.000522 (0.000639)	-0.000882* (0.000522)
Education: Hauptschule	-0.343*** (0.0971)	0.229 (0.174)	-0.591*** (0.229)	-1.423*** (0.180)	-0.427 (0.277)	0.993*** (0.368)	0.261 (0.421)
Education: Realschule	0.102 (0.100)	0.531*** (0.178)	0.187 (0.219)	-0.354** (0.148)	-0.0341 (0.274)	0.998*** (0.361)	0.324 (0.435)
Education: FHR	0.0430 (0.136)	0.364 (0.231)	0.115 (0.293)	0.306* (0.182)	0.143 (0.365)	-0.631 (0.792)	0.213 (0.601)
Education: Abitur	0.00945 (0.101)	0.184 (0.184)	0.758*** (0.211)	0.708*** (0.134)	0.469* (0.265)	-1.307** (0.531)	-0.314 (0.484)
Income (log)	0.123*** (0.0414)	0.141* (0.0737)	0.126 (0.0979)	-0.0152 (0.0581)	-0.349*** (0.0799)	-0.577*** (0.131)	-0.196 (0.171)
Income missing	0.765** (0.318)	0.899 (0.574)	0.661 (0.764)	-0.0897 (0.442)	-2.441*** (0.600)	-3.705*** (0.931)	-0.770 (1.262)
General LoC (1=ext., 7=int.)	0.0747*** (0.0198)	0.0377 (0.0327)	0.130*** (0.0457)	-0.0556* (0.0303)	-0.0928** (0.0410)	0.0520 (0.0738)	-0.0471 (0.0839)
Infl. on soc. (1-7)	-0.110*** (0.0156)	-0.158*** (0.0261)	-0.125*** (0.0359)	0.112*** (0.0253)	-0.0362 (0.0338)	-0.223*** (0.0708)	-0.244*** (0.0757)
Goes to church regularly	0.908*** (0.0658)	1.373*** (0.0974)	0.0938 (0.157)	0.303*** (0.107)	-1.708*** (0.340)	-0.914* (0.516)	0.584** (0.285)
Health (1=good, 5=poor)	-0.0574* (0.0308)	-0.0518 (0.0540)	-0.0823 (0.0676)	-0.0503 (0.0488)	0.138** (0.0635)	0.319*** (0.100)	0.0269 (0.122)
Born in GDR	0.276*** (0.0718)	-1.728*** (0.241)	-0.157 (0.166)	-0.150 (0.127)	3.269*** (0.137)	1.493*** (0.241)	0.661** (0.259)
Constant	-0.707* (0.375)	-2.572*** (0.645)	-2.784*** (0.834)	-1.666*** (0.562)	2.840*** (0.751)	1.572 (1.135)	-2.677* (1.481)
Observations	8897	8897	8897	8897	8897	8897	8897
R-squared

Robust standard errors in parentheses

Baseline: SPD

*** p<0.01, ** p<0.05, * p<0.1

This table shows the results of a multinomial logit regression with the preferred party as dependent variable and the general locus of control as explanatory variable. The omitted party is the SPD as it was the most common choice in the sample.

Figure 1.3: Locus of control and parties 1.

VARIABLES	(1) CDU	(2) CSU	(3) FDP	(4) Grüne	(5) PDS	(6) Rep./DVU	(7) Others
Gender (male)	0.0304 (0.0555)	0.395*** (0.0943)	0.332*** (0.119)	-0.524*** (0.0866)	-0.0360 (0.116)	1.209*** (0.234)	0.531** (0.236)
Age	-0.0252*** (0.00972)	-0.0220 (0.0159)	-0.0313 (0.0198)	0.0700*** (0.0198)	-0.131*** (0.0219)	-0.0471 (0.0529)	0.0608 (0.0509)
Age (power 2)	0.000318*** (9.53e-05)	0.000288* (0.000153)	0.000348* (0.000194)	-0.00102*** (0.000216)	0.00116*** (0.000210)	-0.000404 (0.000634)	-0.000900* (0.000517)
Education: Hauptschule	-0.321*** (0.0976)	0.247 (0.175)	-0.559** (0.229)	-1.391*** (0.183)	-0.424 (0.279)	1.046*** (0.372)	0.254 (0.419)
Education: Realschule	0.153 (0.101)	0.530*** (0.180)	0.180 (0.219)	-0.411*** (0.151)	-0.0553 (0.276)	1.044*** (0.371)	0.335 (0.431)
Education: FHR	0.117 (0.137)	0.358 (0.235)	0.0585 (0.298)	0.204 (0.185)	0.0458 (0.372)	-0.278 (0.659)	0.260 (0.593)
Education: Abitur	0.0701 (0.102)	0.158 (0.188)	0.712*** (0.211)	0.597*** (0.137)	0.423 (0.270)	-1.246** (0.538)	-0.271 (0.472)
Income (log)	0.131*** (0.0417)	0.138* (0.0736)	0.112 (0.0968)	-0.0269 (0.0587)	-0.357*** (0.0802)	-0.602*** (0.137)	-0.196 (0.173)
Income missing	0.816** (0.320)	0.888 (0.573)	0.536 (0.755)	-0.162 (0.447)	-2.457*** (0.601)	-3.919*** (0.966)	-0.782 (1.276)
Infl. of luck (1-7)	0.0179 (0.0167)	-0.0693** (0.0289)	-0.0533 (0.0357)	-0.0824*** (0.0277)	-0.142*** (0.0378)	-0.0188 (0.0636)	0.0855 (0.0734)
Infl. of other people (1-7)	-0.0360** (0.0159)	0.00801 (0.0260)	-0.0607* (0.0365)	-0.0399 (0.0263)	0.101*** (0.0330)	0.158*** (0.0604)	-0.0901 (0.0710)
Infl. of own effort (1-7)	0.172*** (0.0265)	0.153*** (0.0448)	0.146** (0.0589)	-0.195*** (0.0334)	0.0529 (0.0559)	-0.0971 (0.0891)	0.0343 (0.0964)
Infl. of own abilities (1-7)	-0.00332 (0.0205)	-0.0560* (0.0333)	-0.0786* (0.0424)	-0.0621** (0.0307)	-0.00905 (0.0425)	0.00404 (0.0759)	-0.0619 (0.0942)
Infl. on soc. (1-7)	-0.105*** (0.0157)	-0.154*** (0.0260)	-0.116*** (0.0363)	0.119*** (0.0259)	-0.0432 (0.0335)	-0.216*** (0.0716)	-0.243*** (0.0762)
Goes to church regularly	0.919*** (0.0664)	1.398*** (0.0979)	0.0997 (0.157)	0.302*** (0.109)	-1.601*** (0.339)	-0.964* (0.515)	0.598** (0.286)
Health (1=good, 5=poor)	-0.0577* (0.0310)	-0.0547 (0.0541)	-0.0903 (0.0682)	-0.0178 (0.0492)	0.154** (0.0652)	0.274*** (0.101)	0.0557 (0.117)
Born in GDR	0.264*** (0.0728)	-1.763*** (0.241)	-0.178 (0.170)	-0.118 (0.128)	3.214*** (0.138)	1.441*** (0.246)	0.696*** (0.258)
Constant	-1.294*** (0.409)	-2.755*** (0.692)	-2.064** (0.889)	-0.195 (0.592)	2.168*** (0.795)	2.317* (1.187)	-2.937* (1.600)
Observations	8824	8824	8824	8824	8824	8824	8824
R-squared

*** p<0.01, ** p<0.05, * p<0.1
Robust standard errors in parentheses
Baseline: SPD

This table shows the results of a multinomial logit regression with the preferred party as dependent variable and the four locus of control components as explanatory variable. The omitted party is the SPD as it was the most common choice in the sample.

Figure 1.4: Locus of control and parties 2.

On a more detailed level I find that people who believe in the influence of effort are significantly more likely to vote for CDU and CSU ($p < 0.001$) while people who believe that they have a strong influence on society are more likely to vote for SPD ($p < 0.001$).

FDP and B90/Grüne

The liberal party (FDP) generally favours a laissez-faire type of economy with less redistribution. The green party (B90/Grüne) usually has environmental and humanistic goals on their agenda.

Both parties are similar in that they are more popular with highly educated people as opposed to people with only basic education (both $p < 0.001$). This result is in line with Schmitt who finds that people with high education are more likely to vote for B90/Grüne, PDS and FDP. While the FDP seems to be more popular with men, B90/Grüne are more popular with women and people who attend church regularly (all $p < 0.01$). ss When looking at the general locus of control I find that people with an internal locus of control are significantly more likely to vote for the liberals ($p < 0.01$), which is in line with my hypothesis. I also find that people who vote for B90/Grüne believe more strongly in their influence on society while I find the opposite effect for the FDP (both $p < 0.01$).

PDS and Republicans/DVU/NPD

The former socialist party PDS is located on the very left side of the political spectrum. In 2007 it has merged with the WASG and is now known as Die Linke. The nationalistic Republicans, the DVU and the NPD are situated on the very right side of the political spectrum. Both the PDS and the Rep./DVU/NPD are considered to be on the opposing extremes of the German political spectrum. Interestingly enough though, there are many similarities between these parties' voters regardless.

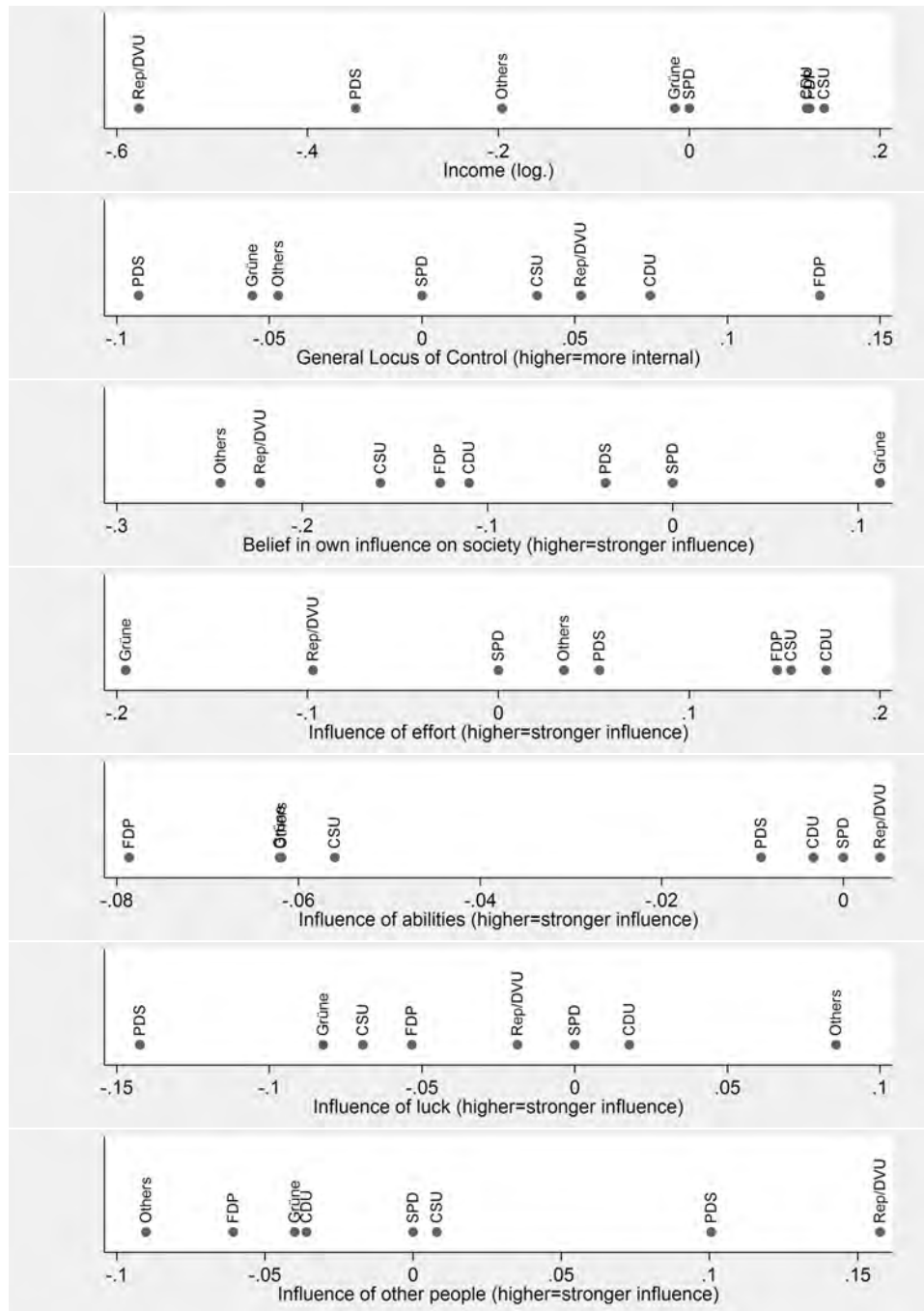
First, both parties are similar in that they are popular among people who have a relatively low income ($p < 0.001$). Both parties are significantly stronger with people who were born in the former GDR ($p < 0.001$). Furthermore both parties are more popular with people who are not religious ($p < 0.01$ for PDS and $p < 0.1$ for Rep.) and with people who have poor health ($p \leq 0.01$). The Rep./DVU/NPD seem to be more popular with male voters ($p < 0.001$) as well as with voters who have a more basic education ($p < 0.01$ for Haupt- and Realschule).

When looking at the general locus of control I find that supporters of the PDS believe less strongly in the influence of luck ($p < 0.001$) while supporters of the Rep./DVU/NPD believe that they have little influence on society ($p < 0.001$). Both parties are similar in that their voters believe that other people have a strong influence on their lives ($p < 0.01$). This is likely related to both parties' agendas as they generally see either rich capitalists or immigrants as the cause for most social problems.

Summary

In summary, I find that the locus of control has a vital influence on how people vote. As hypothesized, an internal locus of control favours right-wing parties like CDU/CSU and FDP, while an external locus of control favours the left-wing parties SPD, B90/Grüne and PDS. Voters of the more extreme parties PDS and Rep./DVU/NPD believe that their lives are strongly influenced by other people and that they have little influence on society.

When considering figure 1.5, which plots the coefficients estimated in the regressions, I find that aligning the parties according to the locus of control components enables one to distinguish between them in a more detailed way than the classical one-dimensional left-right approach can. The perceived influences of effort and other people seem to be two especially promising candidates in this respect.



This figure shows the coefficients taken from the multinomial logit regression displayed in table 1.3. Each graph shows which parties would win (lose) voters if the respective belief were to become stronger (weaker).

Figure 1.5: Estimated relation between beliefs and party preference.

1.5 Effect of Changes in Locus of Control

The multinomial logit analysis in the previous section enables us to analyse how peoples' voting decisions might differ if their locus of control would change. I consider an increase/decrease of each locus of control component as well as the log-income by one standard deviation. For the income variable the standard deviation was calculated based on observations for which the income was not missing. I assume that no strategic considerations bias the individual voting decisions and that an extrapolation by one standard deviation does not generate an excessive forecasting error. In table 1.6 I provide these hypothetical voting results.

1.5.1 Income and General Locus of Control

First I consider a change in the individuals' net income. I find that as the income increases CDU/CSU and FDP become stronger while SPD and PDS lose votes. This effect is moderately strong. An increase of all incomes by one standard deviation would gain CDU/CSU/FDP about 3.5 percentage points compared to SPD/Grüne who would lose 1.6 points. Note that it is above all the PDS and Rep./DVU/NPD who would 'suffer' politically from peoples' increased income.

The effect of a change of the general locus of control seems to be equally strong. It is above all CDU and FDP who would gain from a more internal locus of control, while B90/Grüne and PDS would suffer the most. In fact the effect on the political outcome of an overall variation in the general locus of control is almost as strong as an equally large change (one standard deviation) in income.

		SPD	CDU	CSU	FDP	Grüne	PDS	Rep.	Others	SPD and B90/Grüne	CDU/CSU and FDP
Net Income	Decrease	37,7%	31,3%	6,9%	3,9%	10,5%	6,4%	2,3%	1,2%	48,1%	42,0%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Increase	35,7%	36,6%	8,0%	4,6%	9,7%	3,7%	0,9%	0,8%	45,4%	49,2%
General Locus of Control	More external	38,0%	31,7%	7,4%	3,7%	11,1%	5,6%	1,3%	1,1%	49,1%	42,8%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	More internal	35,6%	36,2%	7,5%	4,9%	9,1%	4,3%	1,5%	0,9%	44,7%	48,6%
Influence of Luck	Weaker infl.	35,8%	32,1%	8,3%	4,5%	11,1%	5,9%	1,4%	0,8%	47,0%	44,9%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Stronger infl.	37,6%	35,8%	6,8%	4,0%	9,1%	4,1%	1,4%	1,2%	46,8%	46,6%
Influence of other people	Weaker infl.	36,0%	35,3%	7,2%	4,6%	10,5%	4,2%	1,1%	1,1%	46,5%	47,1%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Stronger infl.	37,5%	32,6%	7,8%	3,9%	9,7%	5,8%	1,8%	0,9%	47,2%	44,4%
Influence of effort	Weaker infl.	38,9%	30,1%	6,8%	3,9%	12,8%	4,9%	1,6%	1,0%	51,6%	40,8%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Stronger infl.	34,4%	37,8%	8,1%	4,7%	7,9%	4,9%	1,2%	1,0%	42,3%	50,6%
Influence of ability	Weaker infl.	36,1%	33,4%	7,9%	4,6%	10,7%	4,9%	1,4%	1,1%	46,7%	45,9%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Stronger infl.	37,5%	34,5%	7,1%	3,9%	9,6%	4,9%	1,4%	0,9%	47,1%	45,6%
Influence on society	Weaker infl.	33,7%	36,9%	8,7%	4,8%	7,9%	4,8%	1,8%	1,3%	41,6%	50,4%
	Baseline	36,9%	34,0%	7,5%	4,3%	10,1%	4,9%	1,4%	0,9%	47,0%	45,7%
	Stronger infl.	39,7%	30,8%	6,3%	3,7%	12,7%	5,0%	1,1%	0,7%	52,4%	40,8%

I simulate the effect of a change in peoples' beliefs using the previous estimation results. This table shows the distribution of the predicted party preferences if the beliefs were to change by one standard deviation in each direction. The political coalition which would be most likely to win is highlighted in the rightmost two columns.

Figure 1.6: Reaction of party preferences to changes in locus of control.

1.5.2 Detailed Locus of Control

The most significant locus of control component seems to be the perceived effect of effort. An increase by one standard deviation would increase the result for CDU/CSU/FDP by 4.9% while SPD/Grüne would lose 4.6%. This result is in line with the literature in that the locus of control does have a stronger effect on political preferences than income does.

Another important factor seems to be an individual's belief in her ability to change social conditions. I find that SPD, B90/Grüne and PDS would benefit from this while the other parties would lose support. Note that both PDS and Rep./DVU/NPD benefit strongly from more pronounced beliefs in the influence of other people.

1.6 Conclusion

In this chapter I analyse the effect of the locus of control on individual political preferences using data from the German Socio-Economic Panel. I find that the locus of control does have a strong and significant effect both on the likelihood of participating in the political system as well as on the political parties a person is likely to favour. In fact the effect of the locus of control seems to be even stronger than the effect of income.

All parties want to increase the German peoples' wealth and well-being and want to ensure that effort pays off for everyone. Interestingly enough, however, not all parties would benefit politically from a world in which this would be the case. Likewise, all parties propagate the democratic system and want to give the impression that participation can make a difference while again only some parties would really benefit if peoples' beliefs were to change in such a way.

It is above all the parties on the very extremes of the political spectrum who react most strongly to shifts in individual beliefs. The PDS is strong

with people who believe that their lives are mainly determined by other, presumably more wealthy and powerful, people. Likewise the nationalist parties are most popular with people who believe that they are influenced by other people, presumably immigrants, and that they have no influence on society.

The results of this paper imply that changing peoples' preferences and beliefs might be an effective way to change their political preferences. While it is not clear how locus of control beliefs are formed and how strongly they can be influenced, it seems likely that political parties might try to change peoples' beliefs in their favour. Likewise organisations who regard the popularity of extremist parties as a problem might want to focus on the underlying problem that many people feel like they lack control over their own lives.

1.A Appendix

1.A.1 Locus of Control Questions

- 1 My life's course depends on me
- 2 Haven't achieved what I deserve
- 3 What you achieve depends on luck
- 4 Influence on social conditions through involvement
- 5 Others make the crucial decisions in my life
- 6 Success takes hard work
- 7 Doubt my abilities when problems arise
- 8 Possibilities are defined by social conditions
- 9 Abilities are more important than effort
- 10 Little control over my life

Chapter 2

Endogenous Work Norms

2.1 Introduction

The relation between an employer (principal) and her employees (agents) is generally considered an incomplete contract in which the principal's ability to monitor and control the agents is both limited and costly. One common way for the principal to circumvent this problem is to motivate her agents by linking their wage to their productivity. Performance dependent payment schemes like piece rates can incentivise effort if productivity can be observed reliably and is sufficiently correlated to the agents' effort. Payment schemes based on team performance and revenue sharing are often used when an individual's contribution to a project is hard to measure and values of team work are to be emphasised. Many companies also use tournaments in which employees compete for monetary bonuses or job promotion using their effort.

In most companies employees work together in permanent teams, meet each other every work day and often form strong social ties. These ties and the corresponding social pressure can play a vital role when the payment scheme incorporated by the principal interacts with social norms. In revenue sharing schemes individual effort increases the whole team's pay-off and the social norm of helping others enhances the effect of the monetary in-

centives. In tournament situations individual effort decreases the co-workers' chances to win in which case the social norm of not hurting someone else can counteract the intended incentives. It is reasonable to assume that the magnitude of these social norm effects strongly depends on the amount of social interaction between the affected employees as people may, for example, be more willing to hurt a complete stranger than a known person.

In this paper we analyse these effects in a clean laboratory environment in which subjects work on a tedious cognitive real effort task for a chosen period of time. We exogenously vary both the payment scheme and the level of social interaction and observe both the productivity of the subjects as well as the chosen work time. We find that social norms do have a strong effect on a subject's work motivation and can both increase or decrease her performance. Furthermore we find that in order for social norms to have a significant effect social interaction before and after the actual work task is required. In an auxiliary experiment we show that subjects who are to assume the role of an employer are aware of the relevant social norms and a majority is able to choose a performance maximising environment for their imaginary employees.

In section 2.2 an overview of related literature covering payment schemes and social norms is given. Section 2.3 describes the design of our experiment and in section 2.4 we derive theoretical predictions for subject behaviour. The results of our main experiment are presented in section 2.5. The auxiliary manager experiment as well as its results are described in section 2.6. Section 2.7 discusses the results and concludes this chapter.

2.2 Related Literature

There are two branches of literature which are of particular interest for our research question. The first branch analyses the effects of different payment systems and tournament types on individual work motivation and team be-

haviour. The second branch analyses the requirements for and the effects of social norms using theoretical models and experiments.

2.2.1 Payment Schemes and Tournaments

Many publications analyse the incentives resulting from different payment systems and their theoretical and practical implications on agent behaviour.

One common problem with relative payment schemes is that while they do provide incentives to exert effort, they may as well lead to disharmony among the employees up to the point of sabotage. When several agents are paid according to their relative performance and effort cannot be observed, sabotaging other agents may be more cost efficient than doing actual work. The employer faces the dilemma that she cannot incentivise effort without incentivising sabotage as well.

Lazear (1989) analyses how different agent types, so-called doves and hawks, compete for a prize in an environment in which effort cannot be observed and the individual output is partly influenced by luck. The hawks differ from the doves in that their marginal sabotage costs are lower. Lazear shows that both hawks and doves devote a significant amount of their resources to sabotage their co-workers. As the hawks have lower costs of sabotaging they spend more effort on sabotage and less effort on being productive. In effect they free ride on the doves which are more productive.

Companies prefer the more productive dove-type agents but face the problem that the self-selection of agents does not work. Dove-dominated companies generate greater profits, which makes them attractive for both hawks and doves while hawk-dominated companies are attractive for neither type. In order to solve this problem it may be profitable for the company to use screening mechanisms (personality testing) which allow them to distinguish between the types in order to prevent hawks from free riding on their firm.

Nalbantian and Schotter (1997) analyse the optimal effort level of agents working under different payment schemes like gain sharing, revenue sharing and competitive teams. They use a repeated non-real effort task and ask their subjects how much effort they were willing to exert. They find that significant shirking and free-riding behaviour occurs in all treatments. When comparing the different payment schemes they find that subjects tend to provide the most effort under the relative payment systems. The contributions were highest when the subjects were monitored and could be punished when caught shirking. However, the probability and therefore the costs of monitoring had to be substantial in order to provide sufficient incentives for the subjects to behave according to their contract.

In the real world different payment schemes exist in different business areas and people can influence their future payment scheme to a certain extent by their job choice. Dohmen and Falk (2006) analyse the performance of subjects in a cognitive real effort task where the subjects were allowed to choose one out of four different payment schemes (fixed payment, piece rate, revenue sharing and tournament). They find that the performance of subjects working under a non-fixed payment scheme is significantly higher. The driving force behind this result is that the sorting of the subjects is by no means random and depends strongly on individual ability, self-assessment and various preferences. High ability individuals are much more likely to select the performance dependent schemes, which partly explains the increased performance in the relative schemes. They also find that women, risk-avoiding people and people with strong social preferences tend to be less willing to select themselves into competitive payment schemes, which may lead to inefficient sorting in the sense that sought-after high ability workers might hesitate to participate.

2.2.2 Social Norms

Norms describe types of behaviour which are widely recognised within a society and enforced, for example by means of peer pressure. Kandel and Lazear (1992) analyse different kinds of peer pressure and their effect on an agent's motivation. Peer pressure can be modelled both as guilt, which works within the agent herself, as well as shame, which requires an agent's norm-violating behaviour to be observed and punished by others. Unlike guilt, shame requires the co-workers to actively monitor and punish defecting agents in order to become an effective motivator. They interpret social norms as an expected effort level and social pressure as the mechanism which supports this level as equilibrium behaviour.

Kandel and Lazear find that peer pressure can be an important motivator for agents working under team based payment schemes. Furthermore they show that members of a team have to be able to monitor their team-mates in order to ensure an effective effort level. If this monitoring and punishing process is itself costly, monitoring free riding problems may arise in sufficiently large groups, which implies that social norms may be most effective in small or medium sized teams.

Dur and Sol (2010) model an agent's utility as the sum of monetary payoff, effort costs and social attention received from her co-workers. Each agent can contribute to both productive and social activities. The main feature of the model is that investing into social ties increases the recipients' interest in the agent's well-being and may in effect influence the recipients' future decisions in her favour. Therefore committing to social activities may benefit even selfish individuals as it allows them to incentivise more social behaviour in their colleagues. They show that agents may invest too little in social ties in the absence of economic incentives and find that appropriate payment schemes can support the formation of social ties, which may improve both the agents' utility and the employer's profit.

Huck et al. (2003) present a framework which allows the analysis of situations in which both economic incentives and social norms play a role. The main feature of their framework is a social utility function which depends on the externalities caused and received by an agent. Externalities are defined as the loss/gain in utility caused in another agent given an action as compared to the utility which would have been caused by the socially optimal action. They show that social norms can both increase or decrease a rational agent's optimal effort. Furthermore, even situations with multiple equilibria may arise, which they demonstrate using a public good game in which both a low and a high effort level can be maintained. The framework they introduce is applied in the theoretical section 2.4 of this paper.

2.2.3 Experimental Evidence for Social Norms

Several publications analyse the effect of social norms and the conditions required for them to work.

Falk and Ichino (2006) analyse how an agent's effort depends on peer pressure using an experiment employing a manual real effort task involving the packaging of envelopes. In the baseline treatment subjects work alone and have no information about other subjects' performance. In two further treatments the subjects could see a small transparent box supposedly containing the piles of envelopes produced by previous subjects on their desk. They find that this pile size seems to serve as a reference point for the subjects as a high pile increases the average performance as compared to the baseline while a low pile decreases performance. Finally, in a fourth treatment two subjects work in the same room at the same time, which is shown to significantly increase the average subject performance. The chosen treatments are interesting in that they show that both the presence of another person and the mere indication of another persons' performance are sufficient to influence a subject's effort decision.

Krupka and Weber (2009) analyse the conditions necessary for social norms to have an effect. They conduct a laboratory experiment in which subjects play an incentivised dictator game allowing them to choose between social and selfish actions. Besides the baseline treatment, in which no information and no additional information is provided, they employ two so-called focusing treatments, in which the subjects are asked how they think other subjects decide (descriptive focus) or how one should decide (injunctive focus), as well as an informational treatment in which subjects receive information on actual previous subjects' decisions. They find that all three treatments yield a cooperation rate which is significantly higher than in the baseline treatment. Cooperation was highest in the injunctive treatment in which subjects should describe how one should behave. The main result is that it seems to be necessary to make subjects reflect about social norms for them to be effective.

Gächter and Fehr (1999) analyse how social interaction influences cooperation by letting subjects play a repeated public good game and exogenously changing the interaction before and after the game. In the anonymous baseline treatment subjects do not see each other neither before nor after the game and are therefore playing against complete strangers. In a group identity treatment and a social exchange treatment subjects meet before or after the game. Finally in a fourth treatment subjects meet before and after the game. Their main result is that the effect of the social norms supporting the cooperation is by far the strongest in the treatment in which the subjects meet twice. Meeting only before or after the game seems to be insufficient for creating a contribution level which is substantially above the anonymous baseline. A questionnaire which was conducted alongside the experiment indicates that social pressure seems to be the driving force behind the cooperation as many subjects reportedly fear being revealed as a free rider.

Bandiera et al. (2005) are, to our knowledge, so far the only researchers to analyse the effects of different payment systems and effort in a non-laboratory environment over a longer period of time. They cooperate with a UK fruit farm which allows them to exogenously change their workers' payment scheme from a relative payment scheme to a piece rate scheme while maintaining the marginal pay-off effects. A relative payment scheme as employed by the fruit farm inherits the problem that each worker's effort creates negative externalities on her co-workers. In an environment in which strong ties between the workers exist, they may internalise these externalities, either because of feeling sympathy or because of fearing social retribution. They find that the change towards a piece rate vastly increases the workers' performance and argue that this effect can be explained by the fact that the negative externalities are eliminated by the exogenous change and the social norms inhibiting the workers' productivity are made irrelevant. In order to analyse the effect of social pressure they furthermore differentiate between the harvesting of low growing fruit, which allow for easy co-worker monitoring, as well as dense high shrubs, which may conceal individual performance. They find that shrub size does matter as the treatment difference is strongest for 'high visibility' fruit.

In summary the related literature indicates that social norms can significantly increase or decrease the agents' work motivation in the presence of externalities. The magnitude of this effect seems to depend on the social distance as interaction before, during or after the effort decision seems to be necessary for social pressure to be effective.

2.3 Experiment Design

Our experiment consists of a real effort task, which we use to measure a subject's work motivation, and several questionnaires, both of which are described in detail below. We employ six different treatments which differ

in the level of social interaction (separated or together) and in the payment scheme (piece rate, team payment, relative payment).

2.3.1 Overview and Time Line

The basic set-up of the experiment is as follows:

- Upon arrival, each subject is welcomed and led to an individual room. (vacated offices in Frankfurt, empty lecture rooms in Bonn).
- Subjects receive information on the real effort task. After reading the instructions, each subject practises this task for 5 minutes.
- In the together-treatment the subjects in each group are introduced to each other and asked to find personal similarities (e.g. hobbies) for 5 minutes. They are also informed that they are paid on a different day and a corresponding appointment is made. Afterwards, the subjects are separated again and do not see each other until the designated pay-off appointment. In the separate-treatment this step is skipped and the group mates see each other neither before nor after the task.
- Now the subjects receive the instructions for the main task as well as some control questions and questions about social norms. The main task consists of the same real effort task as in the beginning and subjects can work between 0 and 60 minutes on the task. Their later pay-off depends on the number of tables they and their group mates count, depending on the treatment.
- When the subjects decide to end the task a brief questionnaire is displayed. Afterwards the subjects can leave immediately.
- The subjects receive their pay-off on a different appointment approximately one week after the main experiment. The reason for the separate pay-off appointment is that the final individual pay-off depends on

the group mate's effort which is often not known by the time a subject ends the experiment.

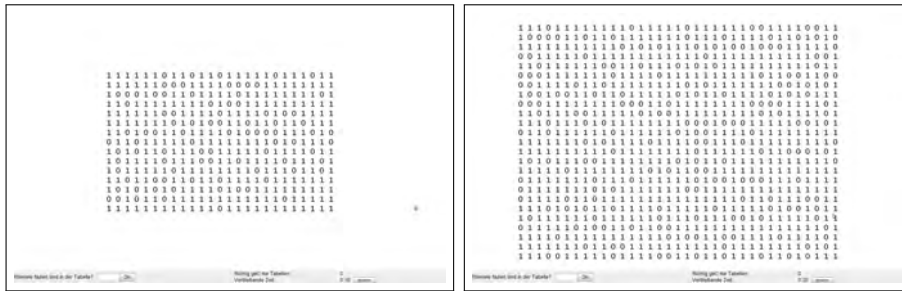
The counting task for the experiment is implemented using Java and the questionnaires are implemented in the Bonn Experiment System (BoXS)¹. The main variables which we are interested in are the time subjects spend on the incentivised main task and how many tables they solve. The complete set of instructions used in this experiment is provided in appendix 2.A.

2.3.2 Real Effort Task

The real effort task used in our experiment consists of counting zeroes in large tables of binary digits. This task is intended to be both boring and uninteresting in order to reduce any intrinsic motivation for doing the task. Furthermore, subjects are explicitly informed in the instructions that the task is useless and only relevant for their pay-off. The reason for this information is that feedback from several subjects participating in a pretest indicates that they continue to work on the task despite wanting to quit in order not to 'endanger' our experiment.

If the hourly wage were constant, subjects should, theoretically seen, either quit immediately or work the maximum time possible, depending on whether the wage is higher (lower) than their reservation wage. While one might assume fatigue to be a driving force in increasing the costs over time, pretests show that this is not the case. In fact even slight evidence for learning effects over time can be found. As a consequence we decide to increase the size of the tables over time. By increasing the table size the time required per table increases and in effect, as the payment per table remains the same, the marginal wage of working on the tables for another minute decreases. Examples for these tables are displayed in figure 2.1.

¹See chapter four.



(a) 15x25 (minutes 0-5)

(b) 25x35 (minutes 55-60)

The size of the tables increases every 5 minutes by one row and one column.

Figure 2.1: The real effort task.

By increasing the table size we enforce a convex cost function and therefore an inner optimum resulting in a working time between 0 and 60 minutes for most subjects. The experiment starts with tables of size 15x25 (425 digits) and every five minutes one additional row and one column are added up to the size of 25x35 (875 digits) after 55 minutes. The payment, which is 0.06 Euro per table, and the rate at which the tables increase over time are calculated in order to provide an average hourly wage of about 3 Euro/hour in the beginning which decreases to about 1.5 Euro/hour in the end, depending on the individual subject's performance. Additionally a show-up fee of about 10-12 Euros, depending on the treatment, is paid.

2.3.3 Treatments

In our experiment we use 3 different payment schemes and 2 social interaction modes resulting in 6 separate treatments.

As payment schemes we employ a piece rate (PR), a team (T) and a relative performance (RP) scheme. The exact pay-off formulae are shown in table 2.1. Note that all three schemes have an equal marginal pay-off β to solving an additional table, which simplifies the upcoming analysis and allows us to compare the results later on. What does change, however, is the externality generated by solving tables, which is positive in the team

Piece rate (PR)	$w_i = \alpha + \beta \cdot x_i$
Team (T)	$w_i = \alpha + \beta \cdot (x_i + x_j)$
Relative performance (RP)	$w_i = \alpha + \beta \cdot (x_i - x_j)$

x_i denotes subject i 's individual performance,
 x_j is the performance of i 's group mate.

Table 2.1: The payment schemes.

Payment scheme	Interaction	Location	Subjects
Relative performance	Together	Frankfurt	25
Relative performance	Separated	Frankfurt	22
Team	Together	Bonn	27
Team	Separated	Bonn	26
Piece rate	Together	Bonn	18
Piece rate	Separated	Bonn	18

Table 2.2: Overview of the sessions.

treatment, negative in the relative performance treatment and neutral in the piece rate treatment.

We also have two interaction treatments. In the *separated* treatment (s) subjects do not meet each other neither before nor after the experiment. In the *together* treatment (t) the subjects within a group get to know each other and solve a small task together before the main task. They are then separated, introduced to the main task and solve the real effort task alone and without being able to communicate and without having any indication of how long their group mate is spending on the task. They do meet again, however, at the pay-off appointment and are then asked to briefly explain their performance to each other. This is common knowledge and communicated to the subjects before the main task.

2.3.4 Subject Pool

We started the conduction of our experiment in Frankfurt, where we could use vacated university offices in order to separate our subjects. Unfortunately the experiment subject pool, which was just started around that time, was

not sufficient to provide enough subjects for all treatments. The fact that many subjects were unfamiliar with laboratory experiments and did not show up in time or at all was also detrimental. Therefore the remaining sessions were done in Bonn, where empty lecture rooms during the semester break were used. All subjects were recruited using the respective laboratories' subject pools and the Orsee-software².

During the sessions, which took up to 90 minutes, a maximum of 4 subjects were working on the task at the same time resulting in a total of 12 to 16 observations per day.

2.4 Theoretical Predictions

In this section we derive several predictions for the subjects' behaviour for each of the three payment schemes introduced above. For each of these payment schemes we calculate the optimal effort level for selfish individuals, the social optimum and the optimal effort when assuming an utility function including social norms.

2.4.1 Selfish Optimum

If we assume that the individual is purely selfish and does not take their group mate's utility or pay-off into account, their optimal behaviour can be calculated easily. We assume a standard utility function which is linear in the wage and includes a cost function:

$$u_i = w_i(x_i) - c_i(x_i) \tag{2.1}$$

The wage function depends on the payment scheme.

$$w_i(x_i) = \alpha + \beta \cdot x_i \quad (\text{PR}) \tag{2.2}$$

$$w_i(x_i) = \alpha + \beta \cdot x_i + \beta \cdot x_j \quad (\text{T}) \tag{2.3}$$

²See Greiner (2004)

$$w_i(x_i) = \alpha + \beta \cdot x_i - \beta \cdot x_j \text{ (RP)} \quad (2.4)$$

Note that all three pay-off functions are chosen in a way that ensures that the derivative of the wage, and therefore the utility, with respect to the number of tables is the same. Using straightforward maximisation we find that the optimal number of calculated tables for selfish individuals is the same for all payment schemes and characterised by:

$$c'_i(x_i) = w'_i(x_i) = \beta \quad (2.5)$$

2.4.2 Social Optimum

In the following we refer to an allocation of actions as socially optimal if it maximises a social welfare function. While it is possible to derive a social optimum for a variety of welfare functions, including Bergson-Samuelson-type functions³, we use the utilitarian welfare function $w = u_1 + u_2$ in which each individuals' utility function is weighted equally.

Piece Rate Treatment

$$\max_{x_1, x_2} \alpha + \beta \cdot x_1 - c_1(x_1) + \alpha + \beta \cdot x_2 - c_2(x_2) \quad (2.6)$$

When the piece rate payment scheme, which does not create externalities, is used, the social maximisation problem yields the following two conditions:

$$c'_1(x_1) = \beta \quad c'_2(x_2) = \beta \quad (2.7)$$

Hence, in the piece rate scheme the socially optimal action coincides with the selfish action.

³Bergson-Samuelson welfare functions define social welfare as the arbitrarily weighted sum of the individual utility functions. The main difference to the utilitarian welfare function is that different individuals can be treated as more or less important. For more information see Samuelson (1977).

Team Payment Treatment

$$\max_{x_1, x_2} \alpha + \beta \cdot x_1 + \beta \cdot x_2 - c_1(x_1) + \alpha + \beta \cdot x_2 + \beta \cdot x_1 - c_2(x_2) \quad (2.8)$$

In the team payment scheme every additional table solved by a player creates twice the welfare pay-off as in the piece rate treatment. Hence we find that both individuals should work until the cost of solving another table exceeds twice the marginal pay-off:

$$c'_1(x_1) = 2\beta \quad c'_2(x_2) = 2\beta \quad (2.9)$$

If the cost of solving a table is linear in the table size this would mean that subjects should solve tables up to twice the size as in the piece rate treatment.

Relative Payment Treatment

$$\max_{x_1, x_2} \alpha + \beta \cdot x_1 - \beta \cdot x_2 - c_1(x_1) + \alpha + \beta \cdot x_2 - \beta \cdot x_1 - c_2(x_2) \quad (2.10)$$

In the relative payment scheme every additional table solved by a player transfers the amount β from her group mate's pay-off to her own pay-off. The overall pay-off and therefore the utilitarian welfare does not change.

$$c'_1(x_1) = 0 \quad c'_2(x_2) = 0 \quad (2.11)$$

The socially optimal action is to work only as long as $c'(x) < 0$ and hence, as we assume solving tables to be costly ($c'(x) > 0$), not working at all is socially optimal.

2.4.3 Individual Optimum with Norms

In this section we use the approach of Huck/Kübler/Weibull to determine the optimal actions of an agent who gets a social pay-off or a social penalty for behaving according to the social norm. The main idea is that the overall utility of an agent is the sum of her monetary pay-off u_i and a social pay-off v_i . Note that while the original model by Huck/Kübler/Weibull allows for any number of players, the formulae used here are simplified for the two player case relevant to our experiment.

$$U_i(x_i, x_j) = u_i(x_i, x_j) + v_i(x_i, x_j) \quad (2.12)$$

$$v_i(x_i, x_j) = G(\psi_i(x_i, \hat{x}_j), \psi_j(\hat{x}_i, x_j)) \quad (2.13)$$

The social pay-off v_i is defined as a function G which depends on the overall externality caused by the individual i on others as well as the externality caused onto individual i . It is reasonable to assume that the derivative of G with respect to the externality on others is positive which means that causing positive externalities increases an individual's utility, often referred to as warm glow⁴.

$$\psi_i(x_i, x_j) = u_j(x_i, \hat{x}_b) - u_j(\hat{x}_i, \hat{x}_b) \quad (2.14)$$

$\psi_i(x_i, x_j)$ denotes the externality caused by player i playing x_i given that player j plays x_j . This externality is measured by calculating the difference between player j 's utility when player i plays the socially optimal action \hat{x}_i and when she plays her actual choice x_i . If a player's choice is identical to the social norm the externality is zero by definition. If a player's choice leads to a higher (lower) utility for the other player than the social norm, it yields a positive (negative) externality.

⁴The idea of the warm glow is that the mere act of contributing to another person's welfare increases a person's utility. See Andreoni (1990).

Piece Rate Treatment In the case of the piece rate treatment there is no externality.

$$\psi_1(x_1, x_2) = u_2(x_1, \hat{x}_2) - u_2(\hat{x}_1, \hat{x}_2) = \alpha + \beta\hat{x}_2 - \alpha - \beta\hat{x}_2 = 0 \quad (2.15)$$

$$\psi_2(x_1, x_2) = u_1(\hat{x}_1, x_2) - u_1(\hat{x}_1, \hat{x}_2) = \alpha + \beta\hat{x}_1 - \alpha - \beta\hat{x}_1 = 0 \quad (2.16)$$

In effect, the externality and hence the social pay-off function is constant in each player's action. Hence a player's optimal action does not change by adding norm awareness as compared to the selfish case.

Team Payment Treatment

$$\psi_1(x_1, x_2) = \alpha + \beta\hat{x}_2 + \beta x_1 - \alpha - \beta\hat{x}_2 - \beta\hat{x}_1 = \beta(x_1 - \hat{x}_1) \quad (2.17)$$

$$\psi_2(x_1, x_2) = \alpha + \beta\hat{x}_1 + \beta x_2 - \alpha - \beta\hat{x}_1 - \beta\hat{x}_2 = \beta(x_2 - \hat{x}_2) \quad (2.18)$$

In the team payment scheme the social pay-off increases in the number of tables a player solves. If the number of tables corresponds to the number in the social optimum, the externality is 0. If it is higher (lower) the externality is positive (negative). Solving the above formulae for the individual utility maximising effort choice yields:

$$\beta + \beta \cdot G'_1(\beta(x_1 - \hat{x}_1), \beta(x_2 - \hat{x}_2)) = c'_1(x_1) \quad (2.19)$$

$$\beta + \beta \cdot G'_1(\beta(x_2 - \hat{x}_2), \beta(x_1 - \hat{x}_1)) = c'_2(x_2) \quad (2.20)$$

Since $G'_1 := \frac{\partial G_1}{\partial x_i}$ is assumed to be strictly positive the optimal effort is now bigger than in the selfish treatment. If we furthermore assume that $G'_1 < 1$, which means that the marginal externality is weighted smaller than the marginal individual pay-off, we find that the optimal effort lies between the selfish and the socially optimal effort.

Relative Payment Treatment

$$\psi_1(x_1, x_2) = \alpha + \beta\hat{x}_2 - \beta x_1 - \alpha - \beta\hat{x}_2 + \beta\hat{x}_1 = \beta(\hat{x}_1 - x_1) \quad (2.21)$$

$$\psi_2(x_1, x_2) = \alpha + \beta\hat{x}_1 - \beta x_2 - \alpha - \beta\hat{x}_1 + \beta\hat{x}_2 = \beta(\hat{x}_2 - x_2) \quad (2.22)$$

In the relative payment scheme the social pay-off decreases in the number of tables a player solves. If the number of tables corresponds to the number in the social optimum the externality is 0, if it is higher (lower) the externality is negative (positive). Solving for the individual utility maximising effort choice yields:

$$\beta + \beta G'_1(\beta(\hat{x}_1 - x_1), \beta(\hat{x}_2 - x_2)) = c'_1(x_1) \quad (2.23)$$

$$\beta + \beta G'_1(\beta(\hat{x}_2 - x_2), \beta(\hat{x}_1 - x_1)) = c'_2(x_2) \quad (2.24)$$

Using the same assumptions as above we find that the players' optimal effort is now lower⁵ than in the selfish treatment.

For the preceding analysis it was not necessary to specify the function G besides making some very basic assumptions. Depending on the function G , situations in which multiple equilibria exist could arise. While it is not necessary to dwell on this any further at this point as we do not need to calculate the exact equilibria, it is important to note that a player's optimal action is likely to depend on their group mate's action and on how they expect her to behave.

2.4.4 Summary

Our main hypothesis concerning the separated and together treatments is that we expect social norms to play a bigger role if the social distance between the group mates decreases, in other words we expect G'_1 to increase when

⁵Note that the derivative of the inner term $\hat{x}_1 - x_1$ w.r.t. x_1 is negative, which yields the negative sign.

$c'(x) = \dots$	Relative payment	Piece rate	Team payment
Selfish optimum	β	β	β
Social optimum	0	β	2β
Optimum w. norms	$\in (0, \beta)$	β	$\in (\beta, 2\beta)$

Table 2.3: Summary of theoretical predictions.

the subjects meet each other.

Table 2.3 summarises the theoretical results from this section. The order of the numbers of tables we expect our subjects to count is characterised by⁶:

$$0 \leq x_{RP,t} < x_{RP,s} < x_{PR,t} = x_{PR,s} < x_{T,s} < x_{T,t} \leq 2 \cdot x_{PR,t} \quad (2.25)$$

2.5 Results

In this section we analyse the results from the experiment. What we are interested in studying is the subjects' motivation to work which is proxied both by the time until a subject leaves as well as the number of correctly solved tables.

The number of tables solved by a subject depends on both the time spent on the task as well as the subject's individual productivity. Unfortunately, even though the counting task is designed to be rather simple, the individual abilities differ substantially between the subjects. Therefore differences in performance can be caused both by different levels of effort as well as different ability. The time spent on the task may be a better proxy for effort. However, the time is also likely affected by individual abilities as high ability implies an increased hourly wage and therefore an increased optimum working time.

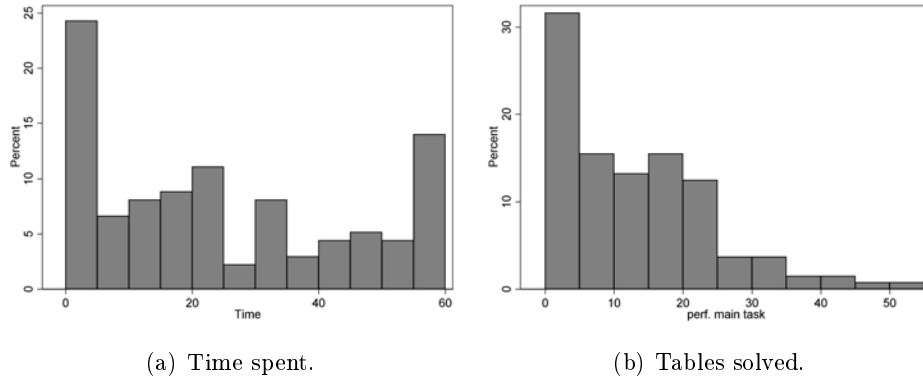


Figure 2.2: Overall distribution.

2.5.1 Time and Performance

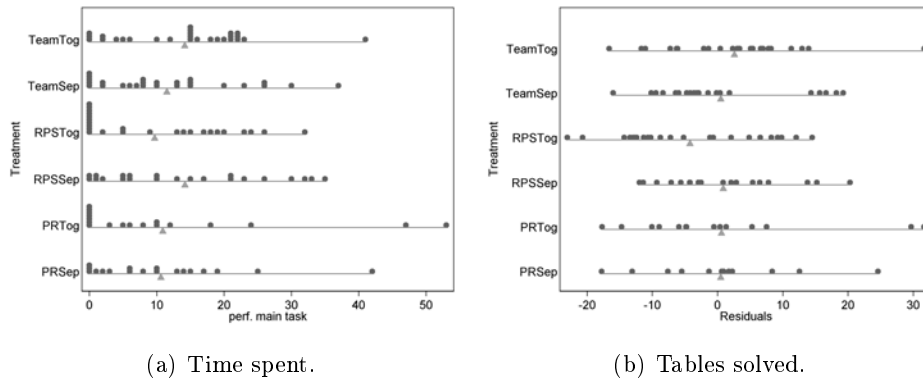
Figures 2.2a and 2.2b show the overall distributions of the time spent on the main task as well as the number of tables correctly solved by the subjects. We find that the distribution of the time is censored on both ends. 25 subjects decided to quit the task immediately and 15 subjects chose the upper bound (60 minutes). The remaining 96 subjects worked between 0 and 60 minutes.

The distribution of the number of tables solved is censored on the left at zero. The maximum number of tables solved in 60 minutes was 53, which was quite impressive.

Figures 2.3a and 2.3b show the distributions separated by treatment. We find that meeting the respective group-mate does seem to affect the time spent on the task in the team payment and the relative payment schemes. In the team payment scheme the average time is 6.6 minutes longer when subjects meet each other. In the relative payment scheme the average time is about 7.5 minutes shorter when subjects meet each other. In the piece rate treatments the difference was less than 2 minutes.

The same is true for the number of solved tables. In the relative (team) payment scheme, subjects solved 4.6 tables less (2.7 tables more) when meeting their group-mate. It seems that social norms do have an effect in the

⁶Note: RP=relative payment, PR=piece rate, T=team, s=separated, t=together



This figure shows beam-plots⁷ for the time spent on the real effort task and the number of tables solved by treatment.

Figure 2.3: Distribution by treatment.

previously conjectured directions.

2.5.2 Regression Results

Table 2.4 shows regressions for each payment scheme and for both the time spent on the task as well as for the number of tables solved. Note that as the distribution of the dependent variable is censored, appropriate Tobit regressions are used instead of the ordinary least squares regressions. So-called Tobit regressions allow for the analysis of data which is normally distributed but censored on one or both sides⁸. We do separate regressions for each payment scheme and are interested in whether the together-dummy, which reflects whether subjects met each other, does have a significant impact on subject performance. We control for gender, age and ability as measured by the individuals' performance in the pretest. We also control for the field of study and the students' income in order to account for possible differences in reservation wage.

In the team treatment we find that subjects spend significantly more time on the task when meeting each other ($p < 0.05$ for the one-sided test). In

⁸Common examples include the visitors of a cinema which has a maximum seating capacity. For more information see McDonald and Moffitt (1980).

	(1)	(2)	(3)	(4)	(5)	(6)
	Time/Team	Time/PR	Time/RPS	Tab./Team	Tab./PR	Tab./RPS
model						
together	15.49* (1.95)	11.34 (1.12)	-26.12** (-2.55)	4.491 (1.20)	3.468 (0.51)	-8.677** (-2.16)
male	-7.666 (-0.72)	-5.430 (-0.51)	-11.68 (-0.91)	1.590 (0.38)	-3.416 (-0.42)	-2.420 (-0.48)
age	1.598* (1.77)	-0.912 (-0.84)	1.726 (0.75)	0.192 (0.95)	-0.612 (-0.89)	0.205 (0.21)
perf. pretest	-1.866 (-0.58)	7.544** (2.77)	8.405* (1.96)	1.094 (0.80)	7.684*** (3.54)	4.212** (2.31)
income	Yes	Yes	Yes	Yes	Yes	Yes
field of study	Yes	Yes	Yes	Yes	Yes	Yes
Observations	42	26	41	42	26	41

t statistics in parentheses

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

The first three columns of this table show the results of three regressions with the time spent on the task as dependent variable. As the dependent variable was censored on the left (at zero) and at the right (maximum was 60 minutes) tobit regressions were used instead of ordinary least squares. The last three columns show the results of three tobit regressions with the number of tables solved as dependent variable (censored on the left at zero). Omitted from this table are the coefficients for the dummy variables used to control for income and field of study.

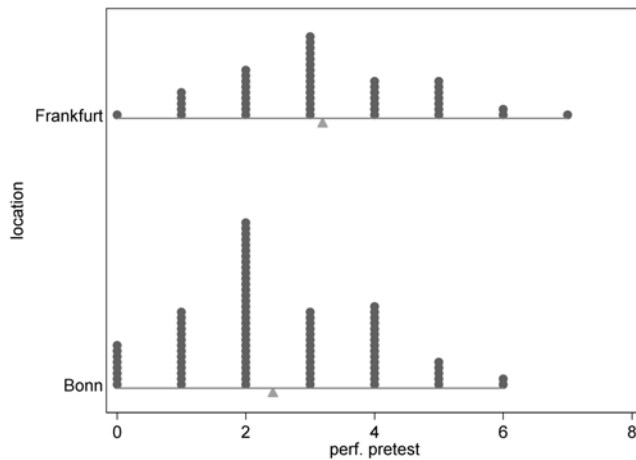
Table 2.4: Regression results.

the tournament treatment subjects spend significantly less time on the task when meeting each other ($p < 0.01$ for the one-sided test) and in effect solve fewer tables ($p < 0.05$ for the one-sided test). For the piece rate treatments we find no significant treatment effects.

2.5.3 Differences in the Subject Pool

One unfortunate finding of our analysis is that there is a significant difference in ability between the students in Bonn and Frankfurt. This can be seen clearly in figure 2.4 which displays the subjects' performance in the pretest and shows that subjects in Frankfurt solve 0.8 tables more on average than subjects in Bonn.

We are not certain about the origin of these differences. Maybe the



This figure shows the performance of the subjects in the pretest, in which they were asked to solve as many tables as possible in five minutes.

Figure 2.4: Performance differential between Bonn and Frankfurt.

different set-up of computers or the different working environment has an effect. It is also possible that the different subject pool, i.e. very experienced and arguably less excited experiment participants in Bonn and first-time participants in Frankfurt has an effect. This is clearly an effect we did not have in mind when designing our experiment.

Note however, that these differences between the two cities, while not particularly convenient, are not problematic for our main analysis as each payment scheme was done in the same city and the treatment effects analysed above could not have arisen from these subject pool differences.

2.5.4 Individual Performance Differential

So, which treatment should the clever and social-norm aware manager choose? In order to answer this question, consider figure 2.5 which improves upon figure 2.3 by controlling for gender, age, individual ability and income⁹. We find that the team-together-treatment seems to yield the highest effort and

⁹Note that controlling for the individual ability also compensates for the city differences discussed above.

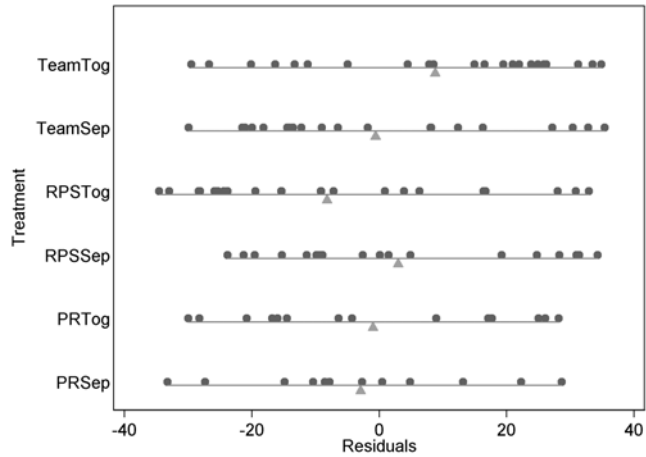


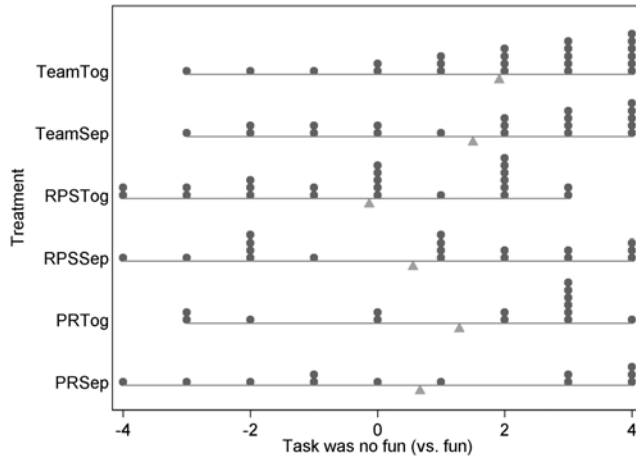
Figure 2.5: Time controlled for individual factors.

increases the time spent by 8.8 minutes compared to the overall mean. The worst treatment seems to be the relative-together-treatment which decreases the time spent by 8.2 minutes. Note that this ranking is in line with our hypotheses derived above.

2.5.5 Subjective Perception of the Task

Another interesting question is how the subjects perceive the different treatments and whether they enjoy working under the respective payment scheme. Since worker motivation might suffer in the long run if the task is perceived as unpleasant, taking worker satisfaction into account is of importance.

Figure 2.6 shows how strongly the subjects enjoy working as elicited by a question in the questionnaire at the end. The results from this figure seem to coincide with the previous results in that subjects seem to be most satisfied working in the team-together task (average +1.9) and least satisfied in the relative-together task (average -0.1).



After the real effort task the subjects were asked as how much fun they perceive the task. The answer can be given on a 9-point scale from -4 (no fun) to +4 (very fun).

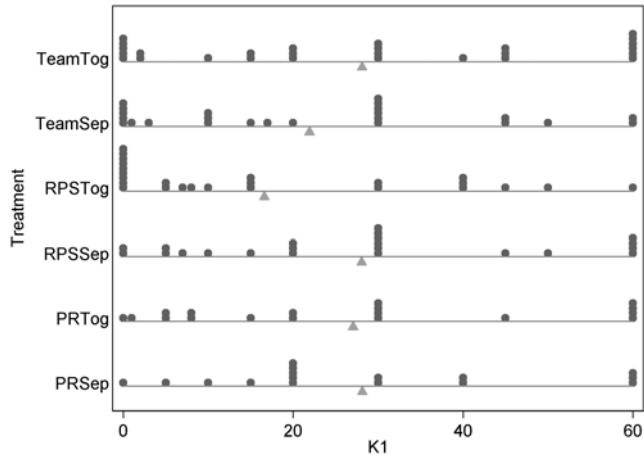
Figure 2.6: Enjoyment of task.

2.5.6 Norm Perception

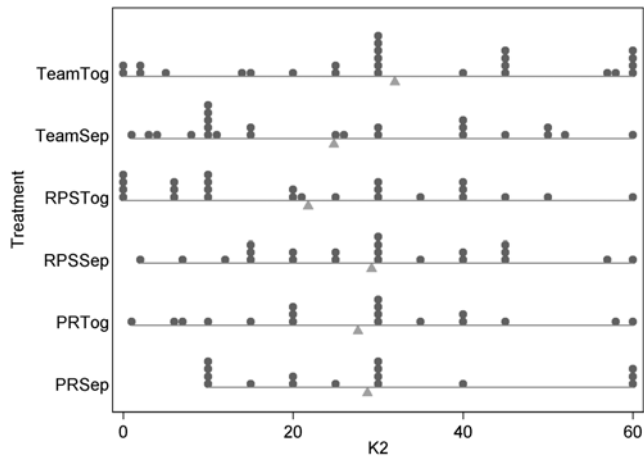
The results from the previous sections indicate that norms do have an effect on individual effort. In order to analyse the cause of this overall effect this section aims to analyse two driving forces.

The first force is that social interaction may lead subjects to realise that there are social norms in the first place which they would miss otherwise. While the instructions were formulated in an easy way and control questions ensured that every subject understood the pay-off mechanism correctly, the situation might still be too abstract for some people to fully understand if they never see the person affected by their actions. In order to measure this effect we ask the subject how long one should work on the task. The result is shown in figure 2.7a. We find that people seem to be much more likely to take the social norm into account when meeting each other.

The second force is, as noted in the theoretical section, that a subject's ideal response in an environment with social norms can depend on how their group mate behaves. Meeting their group mate may lead them to perceive



(a) How long do you think one should work on the task?



(b) How long do you think others will work on the task?

Before the real effort task was started, the subjects were asked how long one should work on the task (a) and how long they expect others to do so (b). The answer could be provided in minutes from 0 to 60.

Figure 2.7: Perceived norms.

her as more cooperative and change their behaviour in effect. In order to estimate this effect we ask the subjects how much they estimate other people to work on average. While this question does not exactly capture the subjects' estimation for their group mate's behaviour, it may serve as a proxy. The result is shown in figure 2.7b. We find that people do seem to estimate other people to behave more norm conforming if they meet each other both in the team and in the relative payment scheme¹⁰.

2.6 Manager Stage

After finding strong evidence for the role of social norms in work environments in the main experiment we are interested in the question of whether people are aware of the social norms playing a role in this situation. In order to find out about this we conduct a second experiment in which subjects assume the roles of managers deciding on a work contract.

2.6.1 Design

This experiment consists of three parts. In the first part the subjects have the opportunity to familiarise themselves with the real effort task from the previous experiment by solving a small, a medium and a large-sized table from the original experiment.

In the second part the subjects are instructed to assume the roles of managers who are to choose the pay-off scheme and the social interaction for their employees in eight different situations. In the first six situations the social interaction is given (i.e. together or separated) and the subjects can choose between one of two pay-off schemes. In the remaining two situations the payment scheme is given (i.e. team or relative) and subjects can choose the level of social interaction.

¹⁰Note that these results can not be fully explained as a self rationalisation mechanism as the corresponding questions were asked before the main experiment was conducted.

The subjects' choice is incentivised by randomly selecting two of the decisions and paying the subjects according to the average work done by the subjects working in the chosen treatments in the main experiment. Therefore the subjects can improve their pay-off by selecting the treatments in which the previous subjects solve the most tables in the main experiment.

Before this part of the experiment is conducted, instructions, examples and control questions are handed out and checked by the experimenters in order to ensure that the subjects understood the task correctly.

The third part of the experiment consists of several questions concerning the subjects' personal situation, their interests and preferences as well as some questions regarding empathy and social preferences. The instructions and questionnaires are supplied in appendix 2.B.

2.6.2 Results

The experiment was performed in the BonnEconLab in three sessions and with a total of 71 subjects recruited from the Bonn subject pool using Orsee¹¹. We ensured that none of the subjects already participated in the main experiment in order to avoid influences and biases due to previous experiences. The experiment was conducted using the Bonn Experiment System (BoXS)¹².

The first result is shown in figure 2.8, which displays the payment systems chosen by the subjects given a level of social interaction.

When choosing between piece rate and team payment (first column), we find that the majority of subjects prefers the team payment in the together setting while preferring the piece rate payment in the separated setting. This difference can be analysed using an appropriate McNemay-test and is highly significant ($p < 0.001$).

When choosing between relative and team payment (third column), we

¹¹See Greiner (2004).

¹²See chapter 4.

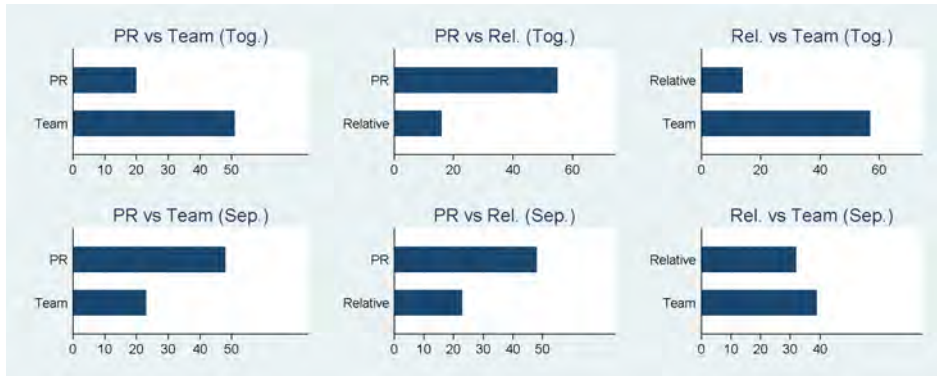


Figure 2.8: Payment chosen by subjects.

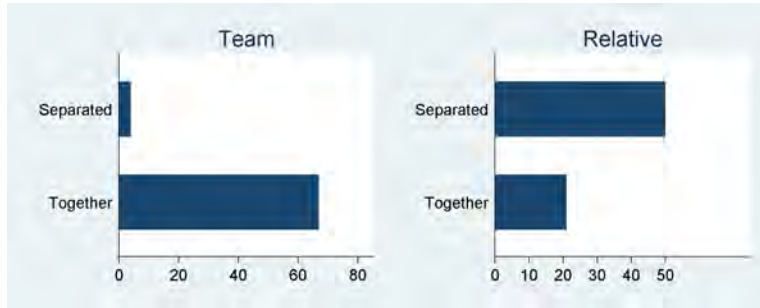


Figure 2.9: Social interaction chosen by subjects.

find that the majority of subjects prefers the team payment in both settings. However, this preference is much stronger in the together setting ($p < 0.001$).

We find no significant difference between piece rate and relative payment as the piece rate was preferred in either case (second column).

The second result is shown in figure 2.9 which displays the social interaction chosen by the subjects given a payment system. We find that when a team payment system is given 67 of 71 subjects choose the together setting while 50 of 71 subjects prefer the separated setting given the relative payment scheme. This difference is highly significant ($p < 0.001$).

2.6.3 Summary

We find that many subjects do seem to be aware of the social norms playing a role in our main experiment and chose the pay-off system or the social interaction which is the most work-promoting given the predictions from our theoretical model and the results from our main experiment.

2.7 Conclusion

In real companies both team based payment schemes as well as competitive schemes are widely used. While these schemes create obvious pecuniar incentives to work hard, they may also cause externalities and in turn trigger social norms which can support or counteract the intended incentives. In this chapter we analyse the question of how the effect of social norms depend on social interaction. In order to answer this question we design and conduct a series of laboratory experiments. We exogenously change both the payment scheme and the amount of social interaction and observe the time the subjects spend working on an incentivised but tedious cognitive real effort task.

We find that social norms can have both very positive and negative effects on individual effort. The effects found are both statistically significant and show a high relevance as the time spent on the task on average decreased or increased by up to one third. These effects are in line with the predictions derived from a theoretical approach. We also find that the effect of social norms strongly depends on the amount of social interaction. In fact, in order for the social norms to have a tangible and significant effect it seems to be necessary for the subjects to meet before and after the work task in question. Furthermore we can show, using an auxiliary experiment, that most subjects are aware of the relevant social norms and are able to choose a productivity maximising environment for their imaginary employees when

given the chance.

While the results presented in this chapter are already significant and show strong effects of social norms, they are likely to underestimate their effect at 'real' work places. The incentives in our experiment are very low, the task is played with a total stranger and no real retribution for violating norms has to be expected. In a real work environment in which workers compete for significant stakes, meet each other every day and strong possibilities for enforcing social norms are in place, the effect of social norms is arguably much stronger and should be taken into account when designing and analysing organisation and payment schemes.

2.A Instructions for the Main Experiment

2.A.1 Instructions Pretest

Introduction

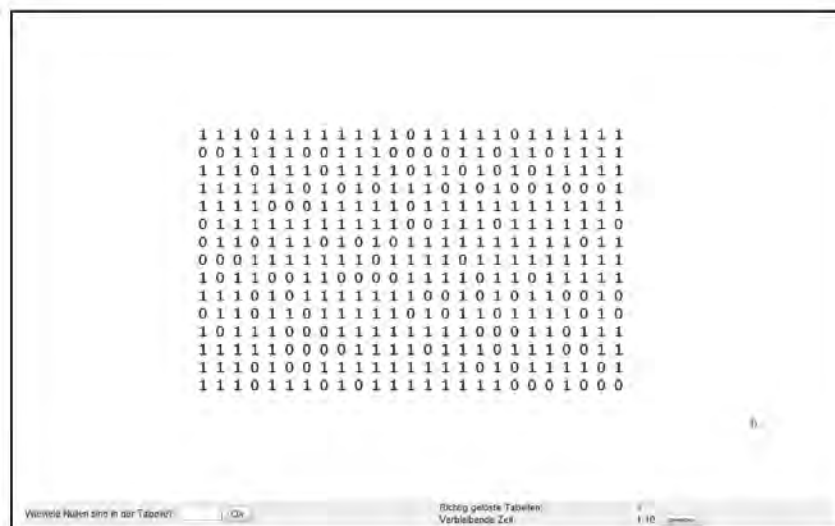
The experiment is composed of several parts. To begin with please read the explanations of the first part. The instructions for the subsequent parts will be provided in the course of the experiment.

You receive 11 Euro for showing up on time (Piece Rate Treatment) [12 Euro in Rel. Performance Treatment, 10 Euro in team Treatment]. You can earn additional money by making decisions in the course of the experiment. The amount of your earnings depends on your decisions. The full amount of your payment will be paid out to you in cash by the experimenters at the previously arranged date.

Instructions for the first part

During this part of the experiment several tables are going to appear successively on your computer screen. Your task is to count the zeros in each of the tables correctly. **Please solve as many tables as possible.** You are given **5 minutes** in total.

The computer screen you are going to use looks like the following:



Please fill in for the number of zeros that you counted to the intended data field at the bottom left of the screen (each table separately). Confirm your input by clicking on the "OK" button.

Afterwards a message informs you whether you entered the correct or wrong number of zeros. The next table appears after clicking the "Ok" button below that message (note: this may take a while).

At the bottom right of the computer screen you can read how many tables you have already solved correctly and how much time is left for completing the task.

Do you have any more questions?

The experimenter is soon coming to you once again. Afterwards the task will start automatically.

2.A.2 Getting-to-know-Task

A Game to get to know each other

Please introduce yourself to the other person in this room. Take down at least 5 similarities that you and the other person share on this sheet of paper. These might include your hobbies, favorite films or preferred vacation spots. There are no limits in finding your similarities!

After approximately 5 minutes, the experimenter is going to bring you back to your room where the experiment will continue.

Enjoy finding your similarities!

Your answers:

- 1.
- 2.
- 3.
- 4.
- 5.
- ⋮

2.A.3 Instructions Main Part

Instructions for the Second Part

Please read these instructions as well as the sheets distributed along with these instructions carefully.

In this part of the experiment your task is once again to work on tables similar to the first part. As before, counting the tables correctly is only relevant for your payoff and has no further use.

Unlike in the first part you can now choose how long you would like to work on the tables. The maximum possible time is 60 minutes. In order to finish the task just click on the respective button on the screen and follow the instructions given on the screen. Finishing the task early does not result in any disadvantages for you. You can leave the experiment immediately after and will receive your full payoff at the designated appointment.

How is your payoff calculated?

SEPARATED TREATMENT

At the same time another person is working on the counting task. You will not meet this person, neither during nor after this experiment.

TOGETHER TREATMENT

At the same time the person you have just met is working on the same counting task.

PIECE RATE TREATMENT

You receive **6 Euro-cents for every table you counted correctly.**

TEAM TREATMENT

You are paid according to the **joint performance of you and the other person**. This means that you receive 6 Euro-cents for each table you or the other person **counts correctly**. At the same time the other person also receives 6 Euro-cents for every table solved by her or by you.

Your payoff is calculated as:

$$0,06 \text{ €} * (\text{„own correct tables“} + \text{„correct tables by other person“})$$

REALTIVE PAYMENT TREATMENT

You are paid according to your performance **relative to the performance of the other person**.

This means that you earn 6 Euro-cents for every correct table you solve more than the other person. If you solve fewer tables than the other person 6 Euro-cents will be deducted from your payoff for every table you solve less.

The payoff for the other person is exactly the other way around.

Your payoff is calculated as:

$$0,06 \text{ €} * (\text{„own correct tables“} - \text{„correct tables by other person“})$$

Please notice: The total amount earned jointly by you and the other person does not depend on the number of tables either one of you solves.

Erroneously counted tables have no influence on your payoff.

PIECE RATE TREATMENT

The following examples are intended to illustrate the payoff.

Example 1:

You work on the task for 20 minutes and count 15 tables correctly. The other person works on the task for 60 minutes and counts 35 tables correctly..

You earn 11 € for showing up on time and $0,06 \text{ €} \cdot 15 = 0,90 \text{ €}$ for the second part of the experiment resulting in a total of 11,90 €. The other person earns 11 € for showing up and $0,06 \text{ €} \cdot 35 = 2,10 \text{ €}$ for the second part resulting in a total of 13,10 €.

Example 2:

You work on the task for 40 minutes and count 25 tables correctly. The other person works on the task for 50 minutes and counts 25 tables correctly.

You receive 11 € for showing up on time and $0,06 \text{ €} \cdot 25 = 1,50 \text{ €}$ for the second part of the experiment resulting in a total of 12,50 €. The other person also earns a total of 12,50 €.

Example 3:

You work on the task for 60 minutes and count 30 tables correctly. The other person works on the task for 60 minutes and counts 35 tables correctly.

You earn 11 € for showing up on time and $0,06 \text{ €} \cdot 30 = 1,80 \text{ €}$ for the second part of the experiment resulting in a total of 12,80 €. The other person earns 11 € for showing up and $0,06 \text{ €} \cdot 35 = 2,10 \text{ €}$ for the second part resulting in a total of 13,10 €.

TEAM TREATMENT

The following examples are intended to illustrate the payoff.

Example 1:

You finish the task immediately and count 0 tables correctly. The other person works on the task for 5 minutes and counts 5 tables correctly.

You earn 10 € for showing up on time and $0,06 \text{ €} \cdot (0+5) = 0,30 \text{ €}$ for the second part of the experiment resulting in a total of 10,30 €. The other person also earns 10 € for showing up and 2,10 € for the second part resulting in a total of 10,30 €.

Example 2:

You work on the task for 60 minutes and count 35 tables correctly. The other person finishes immediately and counts 0 tables correctly.

You earn 10 € for showing up on time and $0,06 \text{ €} \cdot (35+0) = 2,10 \text{ €}$ for the second part of the experiment resulting in a total of 12,10 €. The other person also earns a total of 12,10 €.

Example 3:

You work on the task for 60 minutes and count 30 tables correctly. The other person also works on the task for 60 minutes and calculates 35 tables correctly.

You earn 10 € for showing up on time and $0,06 \text{ €} \cdot (30+35) = 3,90 \text{ €}$ for the second part of the experiment resulting in a total of 13,90 €. The other person also earns a total of 13,90 €.

2.A.4 Norm Questions

Tell us your opinion:

In the following we would like to know which behavior you regard as adequate.

1. How long should one work on this task?

Time in minutes: _____

2. Please estimate the average answer to the question above given by participants in this experiment.

Average time in minutes: _____

If your guess is within 5% of the correct answer, you receive an additional grant of 5 Euro.

3. Please estimate the actual average time spent on this task by participants in this experiment.

Average time in minutes: _____

If your guess is within 5% of the correct answer, you receive an additional grant of 5 Euro.

2.A.5 Declaration Together

Declaration

Please fill out this form after completing the task.

At the previously arranged payment date you will see the person you are matched to again. This declaration form then is going to serve as a basis for discussion on how long and how much you have worked today.

Your statements:

- ∞ Time in minutes that you have worked on the table task: _____

- ∞ Amount of tables that you solved (correctly): _____

- ∞ Why did you decide to work exactly this length of time and solved that amount of tables? Please give reasons for your behavior.

2.A.6 Declaration Separated

Declaration

Please fill out this form after completing the task.

Your statements:

- ∞ Time in minutes that you have worked on the table task: _____

- ∞ Amount of tables that you solved (correctly): _____

- ∞ Why did you decide to work exactly this length of time and solved that amount of tables? Please give reasons for your behavior.

2.B Manager Experiment

2.B.1 Instructions Pretest

Introduction

The experiment is composed of several parts. To begin with please read the explanations of the first part. The instructions for the subsequent parts will be provided in the course of the experiment.

In this experiment you can earn money by making decisions. The amount of your earnings depends on your decisions. The full amount of your payment will be paid out to you in cash by the experimenters at the end of the experiment.

Instructions for the first part

During this part of the experiment three tables are going to appear successively on your computer screen. Your task is to count the zeros in each of the tables correctly. Please solve the tables as fast as possible. The first table has a size of 15*25 digits, the second one has 20*30 digits and the last one is composed of 25*35 digits.

The computer screen you are going to use (here an example of a table with 15*25 digits) looks like the following:

1	1	1	0	1	0	0	0	0	1	1	0	1	1	0	1	0	0	1	1	1	1	1	1	0		
1	0	1	1	0	0	1	1	1	0	1	0	1	0	1	1	1	0	1	1	1	1	1	1	1	0	
1	1	0	1	0	1	1	1	0	1	1	0	1	0	1	0	0	1	0	0	1	0	1	1	0		
1	1	0	0	1	1	0	1	1	0	1	1	0	1	1	0	1	0	0	1	1	1	1	1	0		
1	1	1	1	1	1	0	0	1	1	1	1	0	1	1	1	0	1	0	1	1	0	0	0	1		
1	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	1	1	1	1	0	0	1	0		
0	0	0	1	0	0	1	0	1	1	1	0	1	0	1	1	0	0	0	1	1	1	1	0	0		
1	1	0	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	1	0	1		
0	1	0	0	1	1	0	1	1	0	0	0	1	0	0	1	0	1	1	0	1	1	0	1	1		
1	1	0	1	1	1	0	1	0	0	0	1	1	1	1	0	1	0	1	0	0	1	1	0	1		
1	0	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0		
1	0	1	1	0	0	0	0	0	0	1	1	0	1	1	0	1	1	0	1	1	0	0	1	1	0	
1	0	0	0	1	1	0	0	0	1	0	0	1	0	1	1	1	0	0	1	1	1	1	1	0		
1	1	0	0	1	0	0	0	1	1	1	0	0	0	1	0	1	1	1	0	1	1	0	1	1		
0	0	0	1	0	1	1	0	1	0	1	0	1	0	1	0	0	0	0	0	0	1	0	0	0		

Wieviele Nullen sind in der Tabelle?

Weiter

Please fill in for the number of zeros that you counted to the intended data field at the bottom left of the screen (each table separately). Confirm your input by clicking on the "Weiter" button. Afterwards a message informs you if you have entered the correct or wrong number of zeros. The next table appears after clicking the "Weiter" button below that message (note: this may take a while).

Do you have any more questions? If so, please raise your hands. The experimenter is there to help. Afterwards the task will start automatically.

2.B.2 Instructions Main Part

Instructions for the second part

Please read the instructions carefully and take your decisions subsequently. Questions concerning the decisions will be answered by the experimenters at any time. Please raise your hands and wait for the experimenter to come to you.

What is your role in this experiment?

In this experiment you are given the role of an employer who decides on a remuneration system of two employees.

Which task do the employees work on?

The employees work on a similar task as you just did at your computer. They count in tables full of ones and zeros, the inscribed zeros. At the beginning of the task the tables have a size of 15*25 digits (as the first table that you just saw). They increase every five minutes by one row and one column until they reach a size of 25*35 digits (corresponding to the third table you just saw). The employees can work up to 60 minutes on that task. However, they are also given the opportunity to terminate the task and leave anytime they want to.

What decision do you take now?

You now have to indicate under which remuneration system you want to hire your employees as an employer. Your payoff thereby is determined by the average performance under the particular remuneration system. Performance here is defined as the amount of tables solved correctly.

You can choose from three remuneration systems:

- ∞ **Payment depending on individual performance:** Each employee is paid only for his own performance. Both persons receive (independently of each other) 6 Euro cents for each table they solve correctly:

Person A earns: $0.06 \text{ €} \cdot \text{Person A's performance}$,

Person B earns: $0.06 \text{ €} \cdot \text{Person B's performance}$

- ∞ **Payment depending on joint performance:** For each table that one of the two persons solves correctly, both employees receive 6 Euro cents:

Person A earns: $0,06 \text{ €} \cdot (\text{Person A's performance} + \text{Person B's performance})$

Person B earns: $0,06 \text{ €} \cdot (\text{Person A's performance} + \text{Person B's performance})$

- ∞ **Payment depending on relative Performance:** For each table that one employee solves more than the other person, he or she receives 6 Euro cents. From the person's payment, who solves fewer tables than the other employee, 6 Euro cents are withdrawn for each table solved less:

Person A earns: $0,06 \text{ €} \cdot (\text{Person A's performance} - \text{Person B's performance})$

Person B earns: $0,06 \text{ €} \cdot (\text{Person B's performance} - \text{Person A's performance})$

How is your payment calculated?

The employees have already worked on their task under the various remuneration systems in a preceding experiment. They already received their payment, thus you do not have to pay them anymore. At that time the employees did not know about your existence as employer.

Your payment today is determined by the average performance, that the two participants jointly showed under the remuneration system you choose today. You receive 20 Euro cents per average amount of tables solved correctly.

Your payment =

$0,20 \text{ €} \cdot \text{average performance shown by a pair of participants under the remuneration system you choose}$

For your information: The average performance of a pair of participants for all remuneration systems is an amount of 24 tables solved correctly. The lowest joint performance of a pair of participants are 0 tables, the highest joint performance are 65 correctly solved tables.

What next?

Following the control questions you are asked to indicate under which remuneration system you want to hire your employees in various situations as an employer. For this purpose please use the input mask at your computer. Under which remuneration system do you expect to see the highest performance?

At the end of the experiment two decisions from the following eight decision situations are randomly selected and will serve as a basis for your payment.

Please start now by answering the control questions. If you solved all control questions correctly you can start answering the questions on your computer screen.

Do you have any more questions? If so, please raise your hands. The experimenter is there to help. Afterwards you can start answering the control questions.

Control Questions

1) In the following example you are asked to calculate the earnings of two employees (person A and person B) depending on their performance under the three different remuneration systems. Please use the calculation formula given in the instructions for this purpose.

Person A solves 5 tables correctly. Person B solves 45 tables properly. How much do person A and Person B earn depending on their performance...

...if being paid based on individual performance?

Person A earns:

Person B earns:

... if being paid depending on joint performance?

Person A earns:

Person B earns:

... if being paid depending on relative performance?

Person A earns:

Person B earns:

2) Please mark with a cross, which of the following statements explain this experiment correctly in your opinion.

- The employees for whom I choose the remuneration system have already exhibited their performance on the task under the corresponding remuneration system in the past.
- The employees work today simultaneously on the task.
- My payment is determined by the average performance of the employee pairs.
- My payment is determined by the average performance of each individual employee.
- My payment is not determined by the average performance, but by the concrete performance of a randomly selected pair of participants.

3) A participant pair's average performance over all three remuneration systems in the preceding experiment amounts to 24 correctly solved tables. Assume you decide to choose a remuneration system, under which the average performance is 40 correctly solved tables (Note: This number is entirely fictitious).

How much do you earn as an employer in this case?

Please wait until the experimenter has checked the correctness of your answers to the control questions. Subsequently you can start answering the questions on your computer screen.

2.B.3 Main Computerised Part

Decision Sheet

Please read the following decision situations carefully. Please decide for each situation, which payment system you would choose for your employees. Which payment system yielded the highest performance?
At the end of the experiment two of the following eight situations will be randomly selected and will then decide your payoff. You will receive 20 Euro-Cents per average performance of the employees given the selected payment system.

Start

Decision Sheet

Situation 1/8

Your employees **get to know each other** before they learn about their task. Afterwards they work spatially separated. They meet again at the payoff and discuss their performance. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

individual performance joint performance

Situation 2/8

Your employees **get to know each other** before they learn about their task. Afterwards they work spatially separated. They meet again at the payoff and discuss their performance. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

individual performance relative performance

Situation 3/8

Your employees **get to know each other** before they learn about their task. Afterwards they work spatially separated. They meet again at the payoff and discuss their performance. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

joint performance relative performance

Continue

Decision Sheet

Situation 4/8

Your employees **do not get to know each other** neither before nor after the task. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

- individual performance joint performance

Situation 5/8

Your employees **do not get to know each other** neither before nor after the task. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

- individual performance relative performance

Situation 6/8

Your employees **do not get to know each other** neither before nor after the task. Which payment system would you choose for the employees in this case? Please select one of the two options.

Payment based on...

- joint performance relative performance

Continue

Decision Sheet

In the following the payment system for your employees is already chosen, which means that they will be paid either according to their joint or their relative performance. The employees will work on the task while being spatially separated from each other. You as the employer can choose if your employees get to meet each other before and after the task.

Situation 7/8

Assume your employees are paid according to their **joint performance**.

Please select the situation under which you want your employees to work for you:

Situation a: Your employees **meet each other** before working on the task and meet again after the task to discuss their performance.

Situation b: Your employees **do not meet each other** and will meet neither before nor after the task.

- Situation a Situation b

Situation 8/8

Assume your employees are paid according to their **relative performance**.

Please select the situation under which you want your employees to work for you:

Situation a: Your employees **meet each other** before working on the task and meet again after the task to discuss their performance.

Situation b: Your employees **do not meet each other** and will meet neither before nor after the task.

- Situation a Situation b

Continue

Decision Sheet

In the following the payment system for your employees is already chosen, which means that they will be paid either according to their joint or their relative performance. The employees will work on the task while being spatially separated from each other. You as the employer can choose if your employees get to meet each other before and after the task.

Situation 7/8

Assume your employees are paid according to their **joint performance**.

Please select the situation under which you want your employees to work for you:

Situation a: Your employees **meet each other** before working on the task and meet again after the task to discuss their performance.

Situation b: Your employees **do not meet each other** and will meet neither before nor after the task.

Situation a Situation b

Situation 8/8

Assume your employees are paid according to their **relative performance**.

Please select the situation under which you want your employees to work for you:

Situation a: Your employees **meet each other** before working on the task and meet again after the task to discuss their performance.

Situation b: Your employees **do not meet each other** and will meet neither before nor after the task.

Situation a Situation b

Continue

Questionnaire

1a) Did you know the counting task which your employees did from a previous experiment?

yes no

1b) Did you hear about the counting task from friends or colleagues?

yes no

Continue

Questionnaire

2) Under which payment system would you have liked to work as an employee if you would never meet the other employee? Payment based on ...

- individual performance joint performance relative performance

3) Under which payment system would you have liked to work as an employee if you would meet the other employee before and after the task? Payment based on ...

- individual performance joint performance relative performance

4) Did the meeting between the employees influence their performance in your opinion?
Please elaborate your answer....

Continue

Questionnaire

5) Please answer the following questions:

Situation 1: Performance based on **individual performance**, your employees **know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Situation 2: Performance based on **individual performance**, your employees **do not know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Situation 3: Performance based on **joint performance**, your employees **know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Situation 4: Performance based on **joint performance**, your employees **do not know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Situation 5: Performance based on **relative performance**, your employees **know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Situation 6: Performance based on **relative performance**, your employees **do not know each other**.

How long did the employees on average work in your opinion? (in minutes)

If you were to work on the task as an employee, how long **should one** work? (in minutes)

If you were to work on the task as an employee, how long **would you** work? (in minutes)

Do you think the payment system is fair towards the employees in this situation?

yes no

Questionnaire

6) How old are you?

7) Are you...

male female

8) Do you have siblings?

yes no

9) What are you studying?

Continue

Questionnaire

10a) Do you have a job beside studying?

yes no

10b) How many hours does this job take per week?

Did you work before?

yes

no

11) If you already worked, according to which payment systems have you been paid?

Fixed hourly wage

yes

no

Variable payment based on individual performance

yes

no

Variable payment based on joint performance

yes

no

Variable payment based on relative performance

yes

no

Other payment system:

Continue

Questionnaire

12) How would you describe your political attitude?
(On a scale from 0 (left) to 10 (right))

0 1 2 3 4 5 6 7 8 9 10

13) Have you been in a youth group as a child (e.G.: boy scouts, sports club etc.)?

yes no

14a) Do you do sports?

yes no

14b) If yes: Would you consider your sport a team sport?

yes no

Questionnaire

15) Are you in general a risk taking person or do you try to avoid risks?
(On a scale from 1 (not willing to take risks) to 10 (willing to take risks))

0 1 2 3 4 5 6 7 8 9 10

Questionnaire

16) Which of the following activities do you do in your free time?

Visiting cultural activities, e.G. concerts, theatre, talks

weekly monthly less often never

Cinema, pop concerts, disco, dance

weekly monthly less often never

Active sports

weekly monthly less often never

Artistic and musical activities (making music, dancing, theatre, painting, photography)

weekly monthly less often never

Socialise with friends, relatives and neighbors

weekly monthly less often never

Helping friends, relatives and neighbors when there is something to do

weekly monthly less often never

Charity work in institutions and social services

weekly monthly less often never

Participation in political parties and policy

weekly monthly less often never

Visiting church, attending religious activities

weekly monthly less often never

Continue

Questionnaire

17) You will now be presented a number of statements which describe certain general human traits. Please answer for each statement how much it applies to you. Note that there are no right or wrong answers. Please answer as truthfully as possible.

I have warm feeling for people less fortunate than me.

never seldom sometimes often always

I can vividly imagine the feelings of a person in a novel.

never seldom sometimes often always

In emergencies I feel afraid and anxious.

never seldom sometimes often always

In a dispute, I try to understand both sides before making a decision.

never seldom sometimes often always

If I see someone being abused I feel like I need to protect him/her.

never seldom sometimes often always

I feel helpless when finding myself in very emotional situations.

never seldom sometimes often always

After watching a movie I feel like being one of the characters from the movie.

never seldom sometimes often always

Being in a tense emotional situation frightens me.

never seldom sometimes often always

I am touched by things I only observe.

never seldom sometimes often always

I believe every problem has two sides and try to account for both of them.

never seldom sometimes often always

I would consider myself a warmhearted person.

never seldom sometimes often always

When watching a movie I can put myself in the main character's position.

never seldom sometimes often always

In tense situations I tend to lose control over myself.

never seldom sometimes often always

If someone's behaviour appears strange to me I try to put myself in his/her position.

never seldom sometimes often always

When reading an interesting story or a good book I try to imagine how I would feel if the described events would happen to me.

never seldom sometimes often always

Before criticising someone I try to imagine how I would feel in his/her place.

never seldom sometimes often always

Continue

Chapter 3

Framing and Motivation

3.1 Introduction

In a work relationship an agent typically produces both successes and failures and the principal faces the question of whether and how she should communicate the corresponding positive or negative feedback. While positive feedback in the form of praise might serve as a kind of non-monetary reward and reinforce an agent's motivation, literature suggests that its overall long-term effect is ambiguous. Negative feedback in the form of criticism and blame might hurt an agent's motivation and performance even more.

I am interested in the influence of the framing of positive and negative feedback on an agent's motivation and effort provision. I conduct an internet experiment in which subjects work on an incentivised, tedious and uninteresting real effort task. Depending on the treatment the subjects receive different types of reinforcing feedback upon the successful or unsuccessful completion of a task as a simple form of praise and blame. While praise and blame in real work environments typically imply tangible consequences, using an experiment allows me to isolate the pure motivational effect.

I find that the subjects' performance increases significantly when positive reinforcement is provided upon success and that it decreases strongly when

negative reinforcement is given upon failure. This result is based mainly on a hedonic rather than a calculatory effect. Finally, I can show that the effect of positive reinforcement is vastly different for high- and low-ability subjects. While low-ability subjects are more likely to be encouraged by feedback, high-ability subjects tend to be less appreciative.

Section 3.2 of this chapter provides a brief overview of the related literature. Sections 3.3 and 3.4 present the design of my experiment and include a basic model for the agents' behaviour. Section 3.5 summarises the results, which are then discussed in the concluding section 3.6.

3.2 Related Work

Psychologists explain motivation using the Cognitive Evaluation Theory, which distinguishes between intrinsic and extrinsic motivation. Intrinsic motivation stems from the satisfaction of autonomously doing something useful and using one's skills. Extrinsic motivation is derived from outside factors and includes the wage and possible social pressure. Praise and blame from a principal are specific forms of extrinsic motivation.

3.2.1 Motivational Crowding-Out

When extrinsic rewards like payment or praise are introduced, exerted effort typically increases for as long as the rewards persist. However, when the rewards are withdrawn, performance often drops below the level of a non-rewarded control group. The first experiment to demonstrate this so-called motivational crowding-out was conducted by Deci (1971)¹. The common explanation for this effect is that a person is initially uncertain about her motivation for doing a task. Once an extrinsic reward is introduced, the perceived locus of control for doing the task is shifted towards the external

¹For more literature on motivational crowding-out see for example Frey and Oberholzer-Gee (1997), Frey and Jegen (2001) and the meta-study by Deci et al. (1999).

side, leading to a reduced fulfilment of the autonomy component required for intrinsic motivation.

Since extrinsic rewards might diminish intrinsic motivation they should, according to the above findings, only be offered for tasks with little existing intrinsic value, which is in line with the so-called motivation work cycle match as proposed by Amabile (1993).

3.2.2 Feedback and Implied Control

Benabou and Tirole (2003) analyse how these findings influence whether a principal should provide feedback to an agent. They argue that while extrinsic rewards can undermine the feeling of autonomy, they can also be used to enlarge the agent's perceived competence and hence have positive effects. Assuming that an agent is uncertain about her motivation an interesting optimisation problem arises for the principal, the solution of which depends on the direct cost of the reward and its effects on the agent's behaviour and beliefs. They find that while rewards can serve as effective short-term motivators, they might also negatively influence the agent's beliefs by signalling the principal's lack of trust in the agent's abilities and motivation.

Falk and Kosfeld (2006) conjecture that exerting control can incur a 'hidden cost of control' by signalling a lack of faith in the agent's trustworthiness. They conduct a laboratory experiment in which a principal can choose to exert control over the agent's behaviour by setting a minimum contribution amount in a simple investment game². They find that, even though neo-classical models would predict no difference, the agent's contribution decreases significantly when the principal chooses to restrict her decision.

²In an investment game one subject, denoted as investor, is endowed with a certain sum of money. She can invest a part of this money in another subject, the trustee, who can then send back an arbitrary amount. The amount of money invested is multiplied by a factor greater than one, which implies that cooperation can lead to an ex-post superior level of income for both subjects.

3.2.3 Praise and Blame

The overall effect of so-called feedback intervention, which incorporates providing an agent with information on her task performance and both praise and blame, is controversial. As Kluger and DeNisi (1996) summarise, the direction of the effect seems to strongly depend on whether the feedback is perceived as a way of control by the agent.

Xiao and Houser (2007) and Ellingsen and Johannesson (2008) analyse the role of anticipated feedback on cooperation in the ultimatum game³. Using a laboratory experiment they allow the receiving player to send a message to the dividing player. They analyse the resulting proposals and compare them to a control treatment in which no message could be sent. They find that the mere anticipation of feedback is sufficient to increase altruism and cause more equal offers.

A special type of feedback which is often used as an inexpensive way to reward employees is the distribution of non-monetary awards and titles. Frey and Neckermann (2006) argue that awards and titles can serve as extremely cheap and effective motivators. Given that the precise value of an award is unknown, they might be especially useful in situations in which only incomplete information on an agent's performance is available.

3.2.4 Graphical Social Cues

Several previous studies use smilies or related images as a proxy for human interaction. Eckel and Wilson (2003) show smilies to experiment subjects prior to participating in a cooperation-based game. They find that, depending on the displayed smiley, the subjects' behaviour changes. They argue that seeing facial expressions provides valuable information about another

³The ultimatum game, first proposed by Güth et al. (1982), is a two-player game in which one player (divider) can divide a given sum of money between herself and the other player (receiver). The receiver can either accept the proposed offer, in which case it becomes effective, or refuse the offer, in which case both players receive nothing.

person's intentions and is therefore crucial in situations involving social exchange.

Krupka and Croson (2011) show that the hint of a human face can strongly reinforce the effect of prevalent social norms. In a field experiment they analyse how much money people are donating to support a local library and find that simple graphical cues, in their experiment three dots indicating a human face, have a surprisingly strong effect on the average donation amount.

3.3 Design

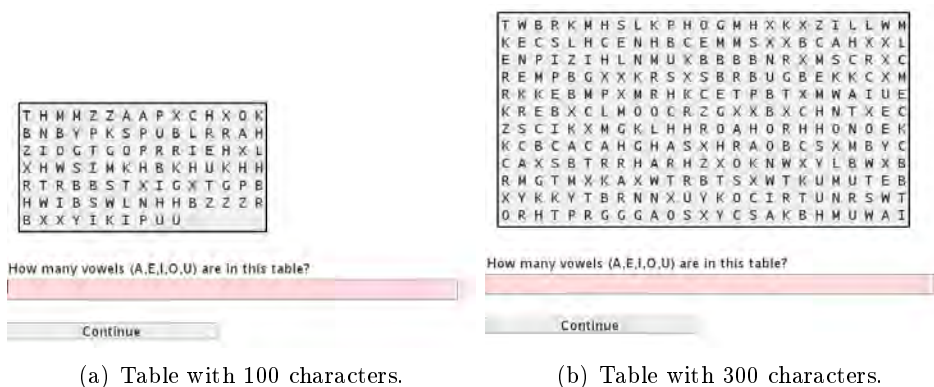
3.3.1 Procedure

I am interested in the effect of reinforced positive and negative feedback on effort provision. In order to analyse this effect I conduct an internet experiment in which subjects participate using their home computers. Internet experiments have the advantage of allowing subjects to participate without the need to travel to the lab. Subjects can quit at any time and have attractive outside options as they do not need to schedule time in advance, which is a big advantage for this type of experiment.

The subjects receive an invitation email and can participate at any time during a period of about 5 days by clicking on a link provided in the email. The experiment is programmed and conducted using the Bonn Experiment System⁴. Upon starting the experiment the subjects are informed about their show-up fee of 3 Euros, asked to fill out a brief questionnaire and provide information on their preferred payment. In order to keep the transaction costs for participating in the experiment as low as possible, I offer to either send the money by mail or by bank transfer⁵. The subjects are informed about the payment options in the invitation email and asked to only participate if

⁴See chapter four.

⁵About 2/3 of the subjects preferred the bank transfer.



This figure shows two example tables of the real effort task. The first table in the experiment contains only 5 characters while the last table contains 299. The number of characters increases by 2 after each table regardless of whether the previous table is solved correctly or incorrectly. After the table containing 299 characters is solved the experiment ends.

Figure 3.1: The real effort task.

they agree with them.

3.3.2 Experiment Design

The real effort task of the experiment consists of counting the number of vowels in tables of randomly generated letters. For each correctly solved table subjects receive 0.10 Euros while 0.02 Euro are deducted for every incorrectly solved table. The subjects are allowed to quit the experiment after each table by selecting the corresponding option, in which case they receive the money earned so far without negative consequences.

The number of letters in each table increases over time, which increases both the time required to solve a table as well as the probability to make a counting error. The consequence of this is that it is relatively attractive to solve the easy tables in the beginning while the task becomes less attractive over time. Two tables of different sizes are shown in figure 3.1 and the full set of instructions is provided in appendix 3.A.

The counting task is deliberately chosen to be tedious and uninteresting and subjects are informed that solving the tables only affects their pay-off

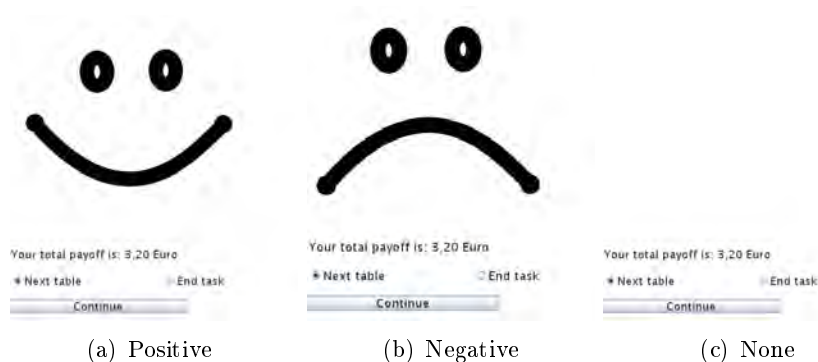


Figure 3.2: Examples for reinforcement types.

and is of no further use. The reason for this statement is that previous experiments indicate that subjects often feel pressured to work on the task in order not to 'risk the results of the experiment'.

The main variable I observe is the number of tables solved by the subjects as a proxy for effort and productivity. Additionally, after every fifth table a brief questionnaire asks the subjects whether they perceive the task as fun or tedious on a 7-point Likert scale and how many of the following five tables they estimate to solve correctly.

3.3.3 Treatment Groups

The treatment variation in the experiment is the way the subjects receive feedback after solving each table. In all treatments information about the total pay-off so far is given, which implicitly includes the information whether the previous table was counted correctly or incorrectly⁶.

Additionally, in three of the four treatments a random positive image is shown on success and/or a negative image is shown on failure, which I refer to as positive or negative reinforcement. The image shown upon counting successfully can be a happy smiley, a thumbs-up, a star or the text 'RICHTIG' (correct). Upon counting incorrectly a sad smiley, a thumbs-down, a light-

⁶If the total pay-off increases by 0.10 Euro (decreases by 0.02 Euro) as compared to the last pay-off, the input was correct (incorrect).

Name	Subjects	Display on success	Display on failure
–	20	Total pay-off	Total pay-off
P-	25	Total pay-off & Pos. reinf.	Total pay-off
-N	24	Total pay-off	Total pay-off & Neg. reinf.
PN	25	Total pay-off & Pos. reinf.	Total pay-off & Neg. reinf.

Table 3.1: Treatment groups.

ning bolt or the text 'FALSCH' (incorrect) is displayed. An example is shown in figure 3.2 and all possible images are provided in appendix 3.B⁷.

3.4 Model and Hypotheses

Neo-Classical Perspective

Using the standard microeconomic model as a foundation one can formulate an agent's utility function as $u(n) = w(n_C, n_I) - c(n, a, t)$, where n_C and n_I denote the numbers of correctly and incorrectly solved tables. $w(n_C, n_I)$ denotes the corresponding wage. The expected marginal wage of working on another table depends on the probability $p(n)$ of solving it correctly, which decreases as the tables become more difficult ($\frac{\partial p}{\partial n} < 0$).

The function $c(n, a, t)$ represents the cost of solving the tables expressed as a pecuniar value which depends on the number of the table n , the individual ability a and the perceived tediousness of the task t . I assume that a subject always tries to solve a table correctly and that solving a table incorrectly is as costly as solving it correctly. I furthermore assume that the ability level a is exogenous and that higher levels of ability decrease the marginal cost of working on the task ($\frac{\partial^2 c}{\partial n \partial a} < 0$). An increase in the level of perceived tediousness t should increase the costs ($\frac{\partial^2 c}{\partial n \partial t} > 0$). Since the difficulty of the tables increases as they get bigger, $c(n, a, t)$ is a strictly monotonously increasing and convex function in n ($\frac{\partial c}{\partial n} > 0$, $\frac{\partial^2 c}{\partial n^2} > 0$).

⁷When only the total pay-off is to be displayed an empty white image is loaded alongside in order to eliminate possible effects from image-treatments taking longer to load and display.

As the second derivatives to all terms of the utility function are negative, one can conclude that the optimal effort for each subject is an inner maximum as characterised by equation 3.1:

$$u'(n^*, a, t) = p(n^*) \cdot 0.1 - (1 - p(n^*)) \cdot 0.02 - c'(n^*, a, t) = 0 \quad (3.1)$$

Behavioural Perspective

From a neo-classical perspective differences in the framing of the feedback should have no effect on the optimal effort provision as the above optimality condition is not affected. However, from a behavioural perspective two effects might occur:

- Subjects may have an unclear perception of their ability a and the probability $p(n)$ of solving upcoming tables correctly. Positive (negative) reinforcement might make the subjects more aware of their successes (failures) and bias their perception of $p(n)$ and a upwards (downwards). I refer to this as the computational effect.
- Subjects may also be unclear about how much fun and how tedious the task is and giving them positive (negative) reinforcement might decrease (increase) the perceived tediousness t . I refer to this as the hedonic effect.

In order to be able to disentangle the computational and hedonic effects three questions are asked during the experiment. A question asking how many of the upcoming tables the subject expects to solve correctly is intended to capture the computational effect. Two questions of how fun and how tedious the task is perceived to be are intended to measure the hedonic effect.

Summary

Both the computational and the hedonic effect would imply a higher effort provision when positive reinforcement is provided and a lower effort provision when negative reinforcement is given. No clear prediction can be made for treatments including both positive and negative reinforcement. In summary, the rank-order of the subjects' productivity we would expect is characterised by:

$$n_{P-}^* > n_{--}^* > n_{-N}^* \quad \text{and} \quad n_{P-}^* > n_{PN}^* > n_{-N}^* \quad (3.2)$$

3.5 Results

This section summarises the results from my experiment. Subsection 3.5.1 describes the data gathered in the experiment, which is then used to estimate the hypothesised framing effects in 3.5.2 and 3.5.3. In 3.5.4 the data is used to estimate the subjects' cost and wage functions and thus provide another perspective on the treatment effect. In subsection 3.5.5 the data from the questionnaire is used to analyse whether the treatment differences are primarily driven by a computational or a hedonic effect. Finally, subsection 3.5.6 presents additional insights derived from examining high- and low-ability subjects separately.

3.5.1 Data

I invited about 300 subjects who were registered at the BonnEconLab using ORSEE⁸. Out of these subjects, a total of 94 subjects participated and were distributed randomly across the 4 treatments (see table 3.1). 8 more subjects participated but experienced technical difficulties during the experiment⁹ and were excluded from the upcoming analysis.

⁸See Greiner (2004).

⁹For example computer crashes, problems with the internet browser, problems with the internet connection etc.

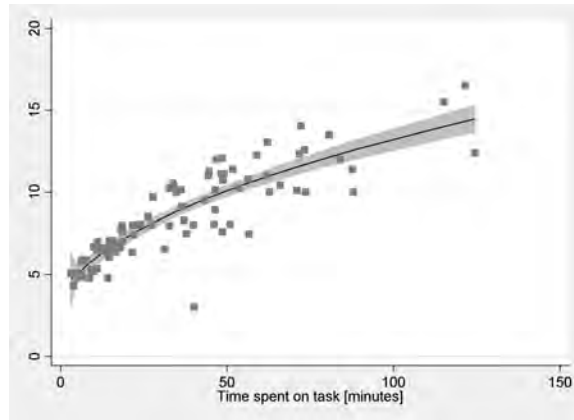


Figure 3.3: Time spent on the main task and pay-off (polynomial fit).

The subjects spent an average of about 35 minutes on the task and earned an average of 8.35 Euros in total. Figure 3.3 shows the relation between the time spent on the experiment and the pay-off, which is concave due to the increasing difficulty of the tables over time. While the first 5 Euros¹⁰ could usually be earned in less than 10 minutes, earning 10 Euros took about 50 minutes on average. Earning 15 Euros was possible but took about 2 hours, which was apparently not very attractive for most subjects.

3.5.2 Influence of Positive and Negative Reinforcement

I am primarily interested in the number of tables solved as a proxy for effort and productivity. The number of solved tables is count data and does therefore not follow the normal distribution, which discourages the use of the usual ordinary least squares regression. Instead, Poisson regressions, which assume Poisson distributed dependent variables, can and should be used. Modern statistics packages like Stata can estimate Poisson regression models efficiently using the maximum-likelihood estimation method. Table 3.2 shows the results of 5 different Poisson regressions on the number of tables solved including different sets of explanatory variables.

¹⁰including the show-up fee

	(1)	(2)	(3)	(4)	(5)
	Tables solved	Tables solved	Tables solved	Tables solved	Tables solved
Tables solved					
Positive reinforcement	-0.0327 (-1.26)	0.00464 (0.17)	-0.0267 (-1.01)	0.0418 (1.45)	0.0613** (2.08)
Negative reinforcement	-0.112*** (-4.33)	-0.105*** (-4.04)	-0.170*** (-6.34)	-0.199*** (-7.21)	-0.257*** (-9.00)
Male		-0.142*** (-5.21)		-0.120*** (-4.00)	-0.190*** (-5.66)
Age		-0.0129*** (-3.00)		-0.0158*** (-3.27)	-0.0126*** (-2.59)
Participated in sim. exp.		0.0251 (0.90)		-0.0207 (-0.70)	-0.0507* (-1.69)
First 23 tables correct			2.213*** (10.17)	2.487*** (10.89)	2.700*** (11.59)
First 23 tables avg. time			0.00201* (1.65)	0.00189 (1.44)	0.00365*** (2.64)
Constant	4.229*** (183.59)	4.572*** (43.61)	2.197*** (10.75)	2.004*** (8.69)	2.047*** (8.81)
Time	No	No	No	Yes	Yes
Field of Study	No	No	No	No	Yes
Observations	94	94	85	85	85

t statistics in parentheses

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

This table shows the results of 5 Poisson regressions with the number of solved tables (both correctly and incorrectly counted tables) as dependent variable. The subjects' performance in the first 23 tables is used as a proxy for individual ability (columns 3-5). Further control variables include dummy variables for the time of the day when the experiment was started by the subject in four-hour-sections (columns 4-5) as well as 7 dummy variables for the category of study (column 5).

Table 3.2: Regression results for number of tables.

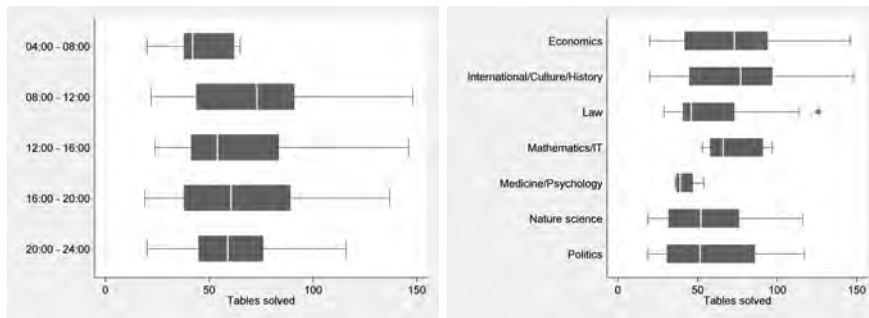
The first column shows a simple regression using only the dummy variables of whether positive and/or negative reinforcement was provided. Columns two and three add controls for gender, age and previous participation in counting experiments as well as proxies for the subjects' ability¹¹.

I find that negative reinforcement has a significant effect and seems to decrease performance by about 10 to 15%¹². Positive reinforcement does not

¹¹I use the percentage of correct tables and the time spent on the first 23 tables as proxies for individual ability. The first 23 tables include the tables containing 5 to 50 characters.

¹²The coefficient estimates of the Poisson regressions can be interpreted as the differences in logs of the dependent variable. This allows us to calculate the approximate treatment differences based on the coefficients:

$$\log n_{*N} = \log n_{*-} - 0.105 \Leftrightarrow n_{*N} = n_{*-} / \exp(0.105) \approx 0.90 \cdot n_{*-}$$



(a) Time of participation

(b) Field of study

Figure 3.4: Performance by time of participation and field of study.

seem to have a significant effect.

3.5.3 Influence of Field of Study and Time of Participation

As the experiment was an internet experiment, subjects could execute it at any day- or night time. Figure 3.4a shows that performance was lowest between 4 am and 8 am, which could be related to the subjects being in a hurry at that time or sample selection effects, with for example the more busy students participating in the morning. In the regression shown in column 4 I control for the time of participation and find that the coefficient for negative reinforcement becomes both stronger and more significant.

Finally, I also control for the field in which the participant is studying and find that the coefficient for positive reinforcement becomes significant. The resulting fifth regression implies that, *ceteris paribus*, positive reinforcement increases performance by about 6%¹³ while negative reinforcement decreases performance by 23%¹⁴.

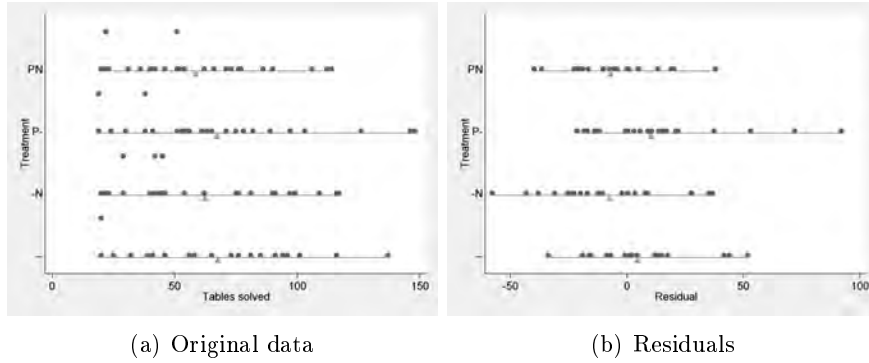


Figure (a) shows a beam-plot of the number of tables solved by treatment. Using the findings from the previous regressions (see table 3.2) I can control for gender, age, individual ability, time of participation and field of study. Figure (b) shows the residuals remaining from a corresponding regression on the number of tables.

Figure 3.5: Performance by treatment.

Graphical Representation of Treatment Differences

The beam plot provided in figure 3.5a shows the number of tables solved by treatment. I find that while the treatments including negative reinforcement (-N and PN) seem to yield somewhat less performance, the effect does not seem to be very strong. Using the Poisson regression analysis to control for the factors explained above, I can filter the results which yields figure 3.5b and shows clear differences in the performance levels between the different treatments.

3.5.4 Estimating Pay-Off and Cost Functions

Using the data collected in our experiment I can estimate both the expected pay-off as well as the cost functions, which allows the analysis of the agents' effort provision problem from a different perspective.

$$\log n_{*N} = \log n_{*-} - 0.170 \Leftrightarrow n_{*N} = n_{*-} / \exp(0.170) \approx 0.84 \cdot n_{*-}$$

$$^{13} \log n_{P*} = \log n_{-*} + 0.061 \Leftrightarrow n_{P*} = n_{-*} * \exp(0.061) \approx 1.06 \cdot n_{-*}$$

$$^{14} \log n_{*N} = \log n_{*-} - 0.257 \Leftrightarrow n_{*N} = n_{*-} / \exp(0.257) \approx 0.77 \cdot n_{*-}$$

Estimating the Pay-Off Function

While the piece rate remains constant during the experiment, the probability of solving a table correctly decreases as the tables get bigger. Assuming that counting each character yields a small error probability f , one can write the probability of solving a table correctly as $p(n) = (1 - f)^{n^{15}}$. This equation and its parameter f can be estimated as a non-linear model based on the data gathered in the experiment. For the estimation I use the average error rate of the first 23 tables. The resulting pay-off function is given by:

$$\hat{E}[u'_i(n)] = 0.1 \cdot (1 - \hat{f}_i)^n - 0.02 \cdot (1 - (1 - \hat{f}_i)^n) \quad (3.3)$$

Estimating the Cost Function

The table q at which a subject quits the experiment indicates the point in time when the expected marginal pay-off from participating in the experiment is surpassed by the subject's marginal cost for participation:

$$\hat{c}'(q, a, t) = \hat{E}[u'_i(q)] \quad (3.4)$$

Using this equation and the assumption that the cost for working on the task is proportional to the required time, one can formulate the marginal cost of solving a table as a function of the required time $d(n, a)$:

$$\hat{c}'(n, a, t) = \frac{d(n, a)}{d(q, a)} \cdot \hat{c}'(q, a, t) \quad (3.5)$$

This marginal cost can be interpreted as the combination of the subject's marginal outside option and the exhaustion resulting from working on the task. Given that the subjects' outside options should be unrelated to the treatment, all treatment differences should indicate changes in the latter component.

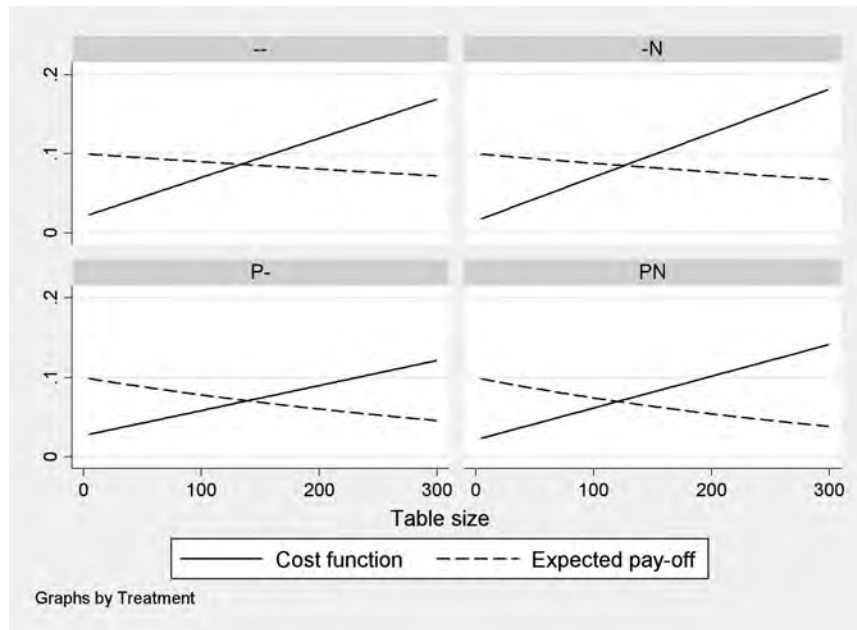


Figure 3.6: Estimated pay-off and cost functions.

Summary

Figure 3.6 shows the estimated ex-ante utility- and cost-functions for an average subject within each treatment. The size of the table at which the average subject quits is located at the intersections of the marginal expected pay-off and the marginal cost. The main difference in the four graphs is the slope of the marginal cost functions. While the slope in the positive-framing treatment is relatively flat, resulting in higher performance, the slope in the negative framing treatment is almost twice as steep. Negative reinforcement seems to make the task more exhausting for the subjects which explains the lower performance.

¹⁵For simplicity I do not account for the rare case of several counting mistakes cancelling each other out.

	(1)	(2)	(3)
	Fun	Tedious	Est. prob.
Count time	-0.000000126 (-0.07)	-0.00000402** (-2.11)	0.000000811 (0.83)
Size of table	-0.0187*** (-14.89)	-0.0185*** (-13.60)	-0.00601*** (-9.25)
Size of table x Positive feedback	0.00233* (1.70)	0.00360** (2.51)	0.000390 (0.57)
Size of table x Negative feedback	-0.00180 (-1.31)	-0.00468*** (-3.24)	0.000216 (0.32)
Constant	1.393*** (19.40)	1.848*** (24.99)	4.713*** (138.27)
Observations	954	987	1139

t statistics in parentheses

* $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

After every fifth table subjects are asked how much fun/how tedious the task is, as well as how many of the upcoming five tables they expect to solve correctly. This table shows the results of fixed-effects panel regressions for each of the three variables. Explanatory variables are the time required to count the current table, the size of the current table as well as the interaction terms between table size and positive/negative framing.

Table 3.3: Influence on fun, tediousness and estimated probability.

3.5.5 Analysing the Influence of Feedback

On a more detailed level, I approach the question of how the effect of reinforcement works by examining the answers to the questionnaires, which were conducted after every fifth table. I use a fixed-effects panel regression in which I model the effect of positive and negative reinforcement as an effect over time. The idea is that while, for example, the fun of solving the tables always diminishes as the tables get bigger, this diminishment might occur at a faster or slower rate when negative or positive reinforcement is provided. The results of these regressions are displayed in table 3.3.

I find that negative reinforcement decreases the perceived fun and significantly increases perceived tediousness. Positive reinforcement has the reverse effect and is significant for both measures. The scale of these effects is about one point on a seven-point Likert scale for fun and tediousness for every 100 tables.

Unlike fun and tediousness, the estimated probability of solving the upcoming tables correctly does not seem to be affected significantly by either

type of framing. Based on these results one can conclude that the framing effect measured in the previous sections is primarily a hedonic rather than a computational effect.

3.5.6 Effect for Low- and High-Ability Agents

The previous sections analyse the framing effect jointly for all subjects regardless of their ability. In this section I partition the subjects according to their ability and analyse the treatment effect for each group. In order to do this I use the subjects' performance in the first 23 tables to estimate their ability. I classify a subject as low (high) ability type if the time required for the first tables is above (below) the median time.

Table 3.4 shows the results of two separate regressions for each ability type. Surprisingly, I find that the effect of positive reinforcement seems to differ strongly for the two types. While the provision of positive reinforcement seems to benefit the performance of low ability types it seems to strongly hurt the performance of high types. Both effects are strongly significant ($p < 0.001$ for both coefficients) in opposing directions. The effect for negative reinforcement is unambiguously negative for both types and likewise significant ($p < 0.001$).

This finding implies that high ability types should perform worst when both types of reinforcement are provided. The combined coefficient for positive and negative reinforcement is -0.552 , which implies a decrease in performance by more than 40%¹⁶ for this treatment compared to the no-reinforcement-treatment, which is a surprisingly huge effect. In fact, the negative effect of the reinforcement treatments is so strong that high ability subjects end up solving fewer tables on average than low ability subjects.

Figure 3.7 shows beam-plots for both subject types. As the previous regression indicates, a clear trend can be seen for high ability types: The

¹⁶ $\log n_{PN} = \log n_{--} - 0.552 \Leftrightarrow n_{PN} = n_{--} / \exp(0.552) \approx 0.58 \cdot n_{--}$

	(1)	(2)
	High	Low
Tables solved		
Male	-0.0197 (-0.32)	-0.132*** (-2.61)
Age	-0.000246 (-0.03)	-0.0165** (-2.09)
Participated in sim. exp.	0.0436 (0.87)	-0.133*** (-2.87)
First 23 tables correct	2.753*** (8.31)	1.707*** (3.81)
First 23 tables avg. time	0.00378 (0.27)	0.000236 (0.13)
Positive reinforcement	-0.222*** (-3.96)	0.297*** (6.97)
Negative reinforcement	-0.330*** (-6.14)	-0.130*** (-2.89)
Constant	1.712*** (5.57)	2.404*** (5.21)
Time	Yes	Yes
Field of Study	Yes	Yes
Observations	42	43

t statistics in parentheses
* p<0.1, ** p<0.05, *** p<0.01

This table shows the regression results for the number of solved tables estimated separately for high- and low-ability subjects. A subject was classified as low (high) ability type if she required more (less) than the median average time for the first 23 tables.

Table 3.4: Number of tables solved by treatment and ability.

more reinforcing feedback is provided, the worse the subject's performance becomes. Performance is highest in the treatment in which no feedback is provided and lowest when both types of reinforcement are given.

For low ability types the effect of positive reinforcement seems to be motivating and performance enhancing. It even seems to slightly outweigh the effect of negative reinforcement. The performance for low ability types seems to be highest when only positive feedback is provided.

3.6 Discussion and Conclusion

I confront experiment subjects with an incentivised real effort task, which can be ended at any time, and exogenously vary the type of reinforcement

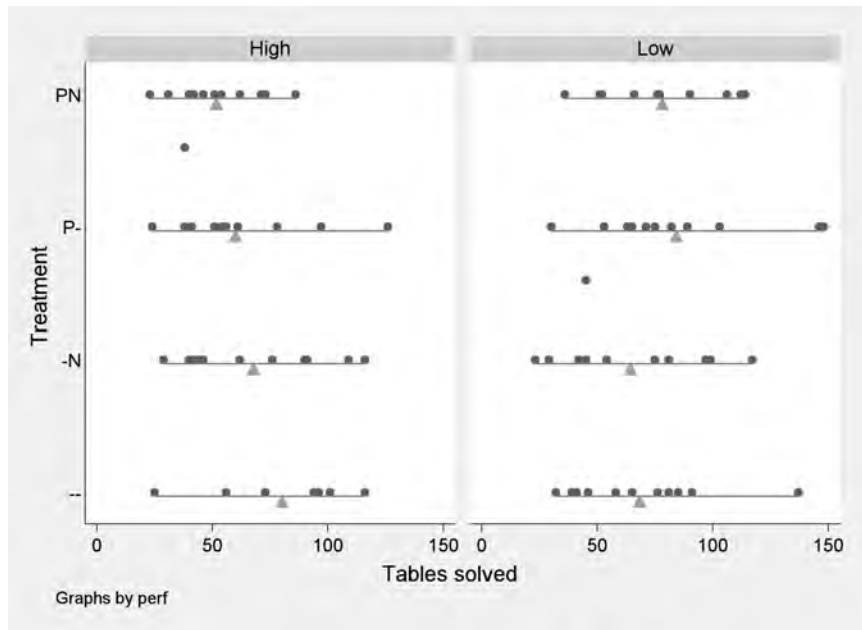


Figure 3.7: Number of tables solved by treatment and ability.

the subjects receive after each table. While every subject receives the information of whether a table is solved correctly or not, some subjects receive additional positive and/or negative images as reinforcement.

I find that the reinforcing images have a significant and strong effect on the perceived tediousness of the task. As a result, administering negative reinforcement decreases the subjects' average performance by up to 20% while giving positive reinforcement increases it by about 5%. Furthermore I find that the treatment effects differ substantially for high- and low-ability subject types. Subjects with low ability seem to appreciate positive feedback and become encouraged by it. Subjects with high ability seem to disregard any kind of feedback and sharply reduce their effort when confronted with it.

While these effects are already surprisingly strong given that the treatment variation is relatively small, it can be argued that the treatment effect estimated in this chapter is likely to underestimate the effect of real world

reinforcement, where feedback is given by a real and well-known person as opposed to a computer program and often bears tangible consequences.

The implications for real work environments are straightforward. While there are obviously cases when negative feedback has to be given in order to avoid future mistakes, the results suggest that explicit negative feedback upon failure should be kept to a minimum and blame should be avoided. At the same time, while positive feedback might be a useful and likewise inexpensive way to increase performance, the results suggest that even praise can be demotivating for some agent types.

Given the data at hand it remains unclear why different subject types react to the reinforcing feedback in a seemingly contradictory way. An interesting aspect for further research seems to be the analysis of the subjects' personality types, for example using the Five Factor Model. Additional questionnaires regarding the subjects' motivation might also yield more insight into the question for which types of people and in which situations reinforcing feedback can be advantageous.

3.A Screenshots of the Experiment

3.A.1 Introductory Text and Questionnaire

Introduction (translated)

Introduction

Welcome to this internet experiment.
You are now participating in an economic experiment. For starting the experiment as well as answering the following questions you receive a payoff of 3 Euro. You can earn additional money in the main part of this experiment.

You can receive the payoff of this experiment by either bank transfer or mail.

Please choose your preferred payoff method:

bank transfer mail

[Continue](#)

Payment Data

<p>Please enter the data of the bank account to which the payoff should be transferred. Please note that your account data will be treated confidentially and will be discarded after the payoff.</p> <p>Kontoinhaber (bank account owner)</p> <input type="text"/>	<p>Please enter the address to which your payoff should be sent. Please note that your address will be treated confidentially and will be discarded after the payoff.</p> <p>First and last name</p> <input type="text"/>
<p>Kontonummer (bank account id number)</p> <input type="text"/>	<p>Street and house number</p> <input type="text"/>
<p>Name der Bank (name of bank)</p> <input type="text"/>	<p>Postal code and City</p> <input type="text"/>
<p>Bankleitzahl (id number of bank)</p> <input type="text"/>	<input type="text"/>
<p>Continue</p>	<p>Continue</p>

Questionnaire

Please tell us something about yourself. Note that the data which is collected in the course of this experiment is treated confidentially and will be analysed in an anonymised way.

Please tell us your age:

Are you...

Male Female

What do you study?

In how many experiments have you participated so far?

none 1-5 5-20 more than 20

Have you participated in an experiment involving the counting of letters or digits?

yes no

3.A.2 Practice Table

Description

In the main part of this experiment your task will be to count the vowels, which is the letters 'A', 'E', 'I', 'O' and 'U', in tables of letters. In order to get to know this task please solve the following practice table.

Hint: If you cannot click on 'Continue' on the next screen this means that your input was incorrect. In this case please try again.

Example Table

Practice table

X	H	K	Y	K	P
Y	E	H	O	Y	Z
M	B	O	O	W	H
B	O	G	B	U	T

How many vowels (A,E,I,O,U) are in this table?

3.A.3 Main Experiment

Description

Main experiment

In the following part of the experiment you receive **10 Euro-Cents** for each correctly solved table which will be added to your payoff. For each incorrectly solved table **2 Euro-Cents** will be deducted. The first table is very easy to solve. The size of the tables increases the more tables you solve. In effect they become more difficult.

Important: After every fifth table you will be asked to briefly answer how difficult the task is and how much fun solving the tables is. Please answer these questions carefully. After each table you have the choice to **quit the main task or solve more tables**.

Also important: The correct number of vowels is already known and **solving the tables has no further use for us**. This is also not an ability or intelligence test. The number of tables which you solve only affects your payoff. Please end this task at will.

When you are ready to start the experiment click on 'Start'.

Start

Task Questionnaire

The task is ...

no fun -3 -2 -1 0 1 2 3 very fun

The task is ...

very tedious -3 -2 -1 0 1 2 3 very easy

Out of the next 5 tables I will (would) probably solve ...

0 1 2 3 4 5

Click on 'Continue', to start the next table:

Continue

Final Screen

Thank you for your participation!

You will receive your payoff of 3,20 Euro in the course of the upcoming weeks by bank transfer.

If you have questions please write to us at: mseithe@uni-bonn.de

(You can now close this window.)

3.B Positive and Negative Reinforcement

Note: The smilies and the text were created using OpenOffice and gimp, the thumbs-up is a royalty free image taken from Wikimedia commons, see:
http://kamelopedia.mormo.org/index.php/Datei:Symbol_thumbs_up.svg

3.B.1 Positive Reinforcement



3.B.2 Negative Reinforcement



Chapter 4

Bonn Experiment System

4.1 Introduction

Behavioural experiments have become a vital part of economic research in the preceding decade as they allow researchers to study actual human behaviour beyond the predictions of theoretical models. Most major economics departments now run dedicated laboratories which centralise the recruitment of experiment subjects and simplify the experiment conduction.

Nowadays most experiments are conducted using computers instead of pen-and-pencil methods, which brings both theoretical and practical advantages for experimenters: 1) Computers allow for experiments involving complex real-time interaction between subjects, e.g. in market or auction related experiments, which would be extremely tedious to conduct otherwise. 2) Using computers to interact with the subjects reduces possible experimenter effects and makes reproducing an experiment easier. 3) An experiment which was programmed once can be easily documented and shared amongst researchers. 4) The data generated by an experiment can be automatically collected and exported to spreadsheet and statistics programs.

In the course of the last decade, computer technology has vastly improved, affecting both the abilities of modern computers as well as their

possible applications. Fast and stable internet connections are widely spread among both institutional and private users and computers have become able to display high quality audio and video files. At the same time the acceptance of computers has increased with most users as has their sophistication in using them.

4.1.1 Related Work

In the very beginning of computerised experiments no experiment software existed which would help experimenters design and run their experiments. This required every experimenter to implement her experiment from scratch using complex programming languages like C++, which in turn required the experimenter to either acquire significant programming skills or delegate the implementation to a professional programmer. While many experiment designs are easy to explain, they may be very hard to implement. Especially the programming of network communication and the graphical user interface can be very complicated and tedious and often outweighs the advantages of using computers in the first place.

The first major improvement on this situation came in the form of RatImage, developed by Abbink and Sadrieh (1995), which is a library of common functions required for most experiments, for example user interface design. While RatImage still required the experimenter to program his experiment in low-level programming languages, many tasks could be vastly simplified by using its predefined routines. Unfortunately, RatImage, which was designed for the outdated MS-DOS operating system, seems to be neither supported nor available any more.

The next major improvement was z-Tree, which was introduced by Fischbacher (1999)¹ and has been steadily supported and improved ever since. Based on the citation count it is probably the most relevant experiment

¹See also Fischbacher (2007).

software to date, especially for economists. The main feature of z-Tree is that it allows the experimenter to design many experiments without writing any program code. It provides an extensive graphical user interface which makes all the important functions accessible and allows the user to design experiments by arranging basic components like text fields and buttons on a tree-like structure. By providing this simplified approach z-Tree allows experimenters with no prior programming experience to implement and run an experiment, while at the same time providing a feature set extensive enough to allow for the implementation of most experiment types. Z-Tree is designed for the Microsoft Windows operating system and provides both server (zTree) and a client (zLeaf) applications which communicate using the TCP-IP protocol. The most recent version of z-Tree implements graphics, both for presentation and interaction, the support for external hardware and chat functionality.

Regate, designed by Zeiliger (2009), is another experiment software system for Windows which enables experimenters to program and conduct computerised experiments². It provides an elaborate and complex user interface which experimenters can use to program and supervise their experiments. Programs consist of several script statements which are inserted in a tree structure. Debugging and testing are simplified in Regate by a) enabling the experimenter to play several subjects on the same computer and screen at the same time and b) providing the possibility to simulate subjects' behaviour by having the software make random choices in a specified range. Regate includes an online documentation and provides several sample programs.

Finally, Kirchkamp (2004) provides a good overview on how internet experiments can be implemented. He explains both how to use existing experiment software like z-Tree and RatImage in an internet environment as

²Since no published paper on Regate is available yet, this paragraph is based on the presentation and the manual available at the official homepage: <http://www.gate.cnrs.fr/~zeiliger/regate/regate.htm>

well as the more basic programming approach based on HTML and PHP.

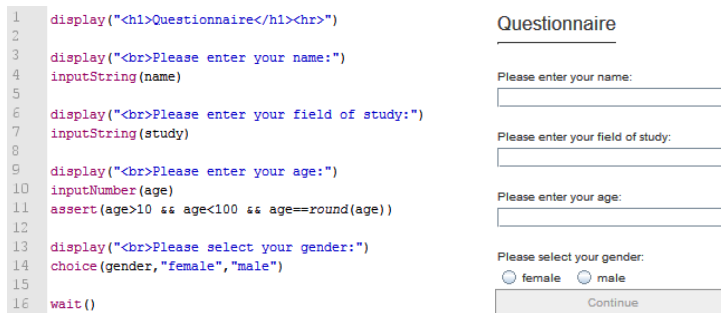
All the mentioned platforms have weaknesses. First, they are designed for laboratory experiments only and are often not designed for mobile or internet experiments. They also heavily rely on Windows as their only supported platform and cannot be used on devices like mobile phones. Second, they are often not very user friendly, not very easy to learn and often lack a comprehensive and up-to-date documentation. Finally, many of the mentioned platforms are no longer supported and often cannot be used on recent computers.

4.1.2 Introducing the Bonn Experiment System

This chapter introduces the Bonn Experiment System (BoXS), which provides a platform for experimenters which is both flexible and easy to use.

The flexibility of the Bonn Experiment System arises from two facts. First, the system is based on the Java platform, which allows it to be used on a wide variety of platforms, including both different device types like netbooks and mobile devices as well as different operating systems like Windows, Linux and MacOS. Second, while it is still possible to download and use the BoXS in an offline environment, it can use the internet as a medium to connect the computers participating in an experiment, which enables any computer worldwide to participate in an experiment without requiring the experimenters to set up their own network structure. This allows for a variety of experiment environments:

- Laboratory experiments, both using an official server (which is easier to use) or an offline server (which allows for experiments without an internet connection).
- Internet experiments in which subjects participate using their private computers at home.



(a) Program code.

(b) Resulting screen.

Figure 4.1: A simple questionnaire in the Bonn Experiment System.

- International experiments where subjects from different countries participate using computers connected over the internet.
- Mobile experiments using netbooks, laptops or Java-compatible mobile phones connected over wireless internet.
- Cross-platform experiments involving Windows, Linux and MacOS.

The Bonn Experiment System also introduces useful features like the simple measurement of response times and the tracking of a input history for each variable, which may be interesting for researchers interested in choice revision behaviour or the individual decision process.

Besides being flexible, the BoXS is also very robust. When a subject's computer or even the experimenter's computer crashes, the experiment continues and the affected subjects/experimenter can simply reconnect and resume the experiment at the point before the crash while all previous data is preserved.

The BoXS is easy to use for several reasons. The programming language implemented in the BoXS is designed to be compact, easy to learn and intuitive to use and resembles popular programming languages like Java. The BoXS also features extensive documentation including an online manual, example programs, a tutorial, a site answering frequent questions, a discussion

group where questions can be posted and, coming soon, video tutorials for the most common questions. The user feedback from experimenters writing their first experiments using the BoXS has so far been very positive. Furthermore, the BoXS does not require any installation on a computer. This makes setting up even complex experiment environments easy as inviting someone to participate in an experiment only requires sending a link. Testing and debugging is also easy as the BoXS allows the easy simulation of a large number of subjects.

4.1.3 Outline

Section 4.2 of this chapter is intended for experimenters who have not used the BoXS before and want to learn about its features. It starts with a brief tutorial and provides information on how to use the BoXS in different environments.

Section 4.3 describes the BoXS Programming Language (BoXSPL), which is introduced by the BoXS and is intended to provide a simple way for non-programmers to design experiments. The section describes how programs are executed and how the most important commands work.

Section 4.4 provides a more in-depth technical description of the underlying network architecture and communication, as well as on how the server and client software is realised. It is primarily intended for readers with a computer science background who are interested in how the Bonn Experiment System works.

The last two sections discuss the current state and the possible future development of the BoXS. Finally, a full documentation of the BoXSPL as well as several example programs are provided in the appendix.

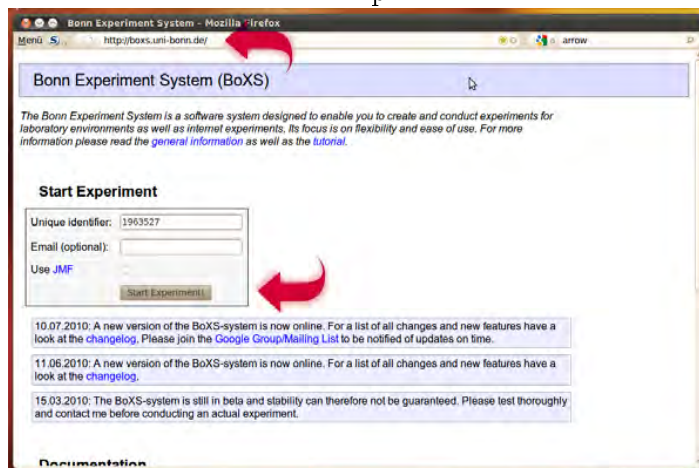
4.2 Using the BoXS

This section provides practical tips for experimenters considering to use the Bonn Experiment System (BoXS). It begins by providing a brief tutorial which demonstrates how to write a simple experiment, proceeds with a description of the user interface and explains how to use the BoXS in laboratory and internet experiments.

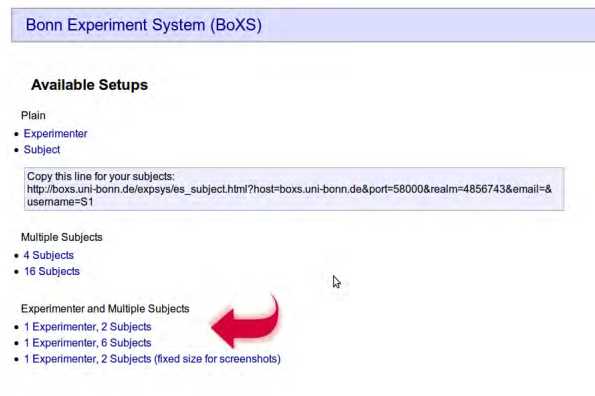
4.2.1 Quick Start Tutorial

This tutorial explains how to write the quintessential "Hello World"-program in less than 5 minutes. For this tutorial to work an internet connection, an internet browser and the Java plug-in for the browser are required.

1. Launch a web browser and open the site `boxs.uni-bonn.de`.



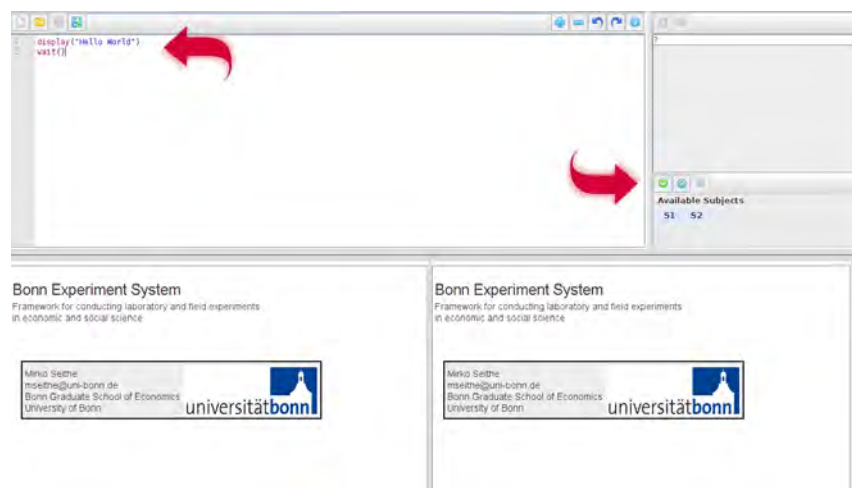
2. Click on "Start Experiment!".
3. Click on "1 Experimenter, 2 Subjects".



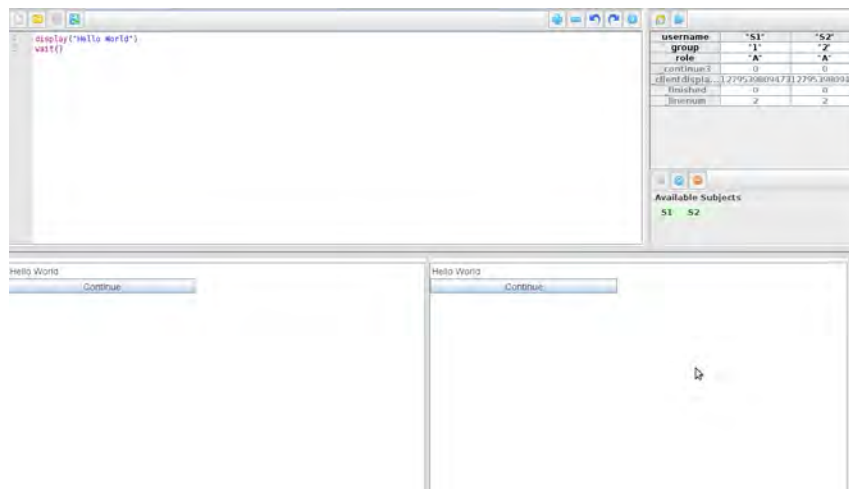
4. When asked for a password, just click on "Ok".

5. The top half of the screen contains the experimenter view. The bottom part contains two subject views for testing purposes. Click on the large white area in the top and enter the following program:

```
display("Hello World")  
wait()
```



6. When done, click on the green start-icon on the right.



The Hello World program is successfully compiled by the server and executed on the two simulated subject clients in the bottom. When you click on "Continue" in the subject views the experiment ends and you can write and start a new experiment. Feel free to experiment by editing and expanding the example program.

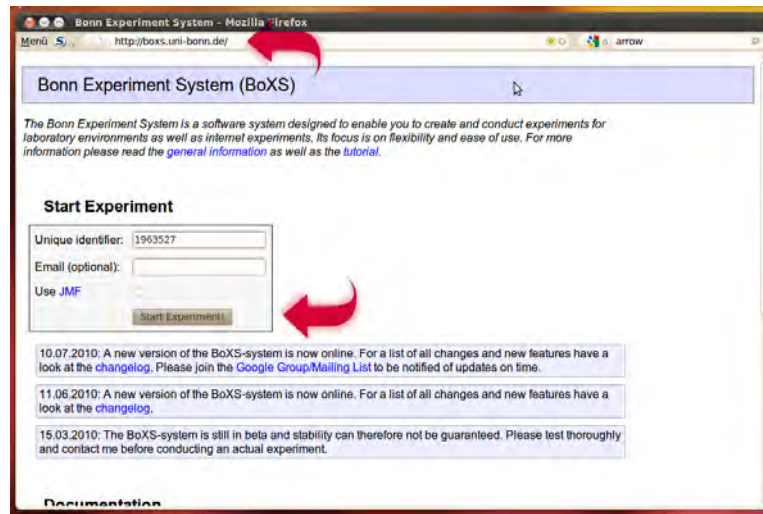


Figure 4.2: The starting page.

4.2.2 Starting an Experiment

Figure 4.2 shows the BoXS web site which is typically used to start an experiment. At the beginning, the experimenter has to specify a realm id and, if required, her email address. In the BoXS, each experiment is uniquely identified by its realm id which ensures that your subjects do not get mixed up with other experiments. By default the realm id is a generated random number which is sufficient for most cases. Alternatively, it can be set to the experimenter's name, institution or her experiment's name. Specifying an email address enables the BoXS to automatically send results to the experimenter's email account. Note that this is completely optional as data can be exported without using this mail option.

Upon clicking on 'Start Experiment!', the 'Available Setups'-page shown in figure 4.3 is displayed. This page offers a large number of possible display set-ups for the experiment, which each include an experimenter view and/or one or more subject views. The quick start tutorial uses one experimenter view and two subject views on the same page, which is useful for testing purposes. Other available set-ups include pure experimenter or subject views,

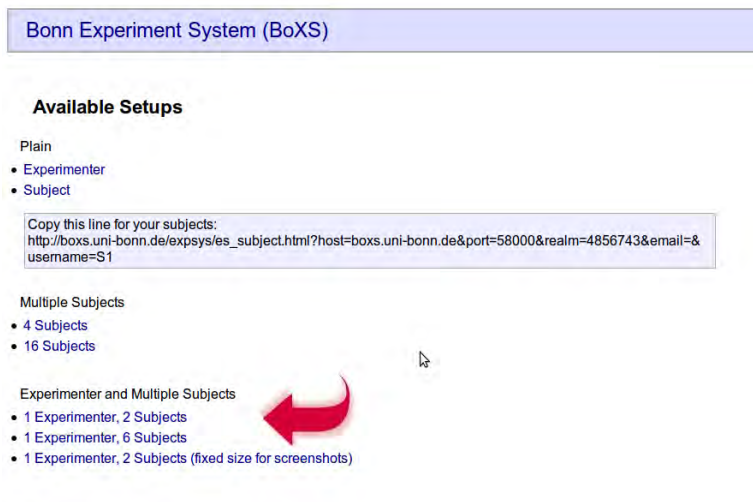


Figure 4.3: The 'Available Setups' page.

which are useful for actual experiments, as well as pages with up to 16 subjects each, which are intended for testing and debugging. If more than 16 subject views are required, the respective page can be opened multiple times. Every set-up opened from this page is automatically associated with the created experiment and shows up on the corresponding experimenter's screen.

Note that while an arbitrary number of subjects can be active at the same time, only one experimenter can. When another experimenter client is started, all previously connected experimenter clients for the respective realm are disconnected automatically.

4.2.3 The Experimenter View

Figure 4.4 shows the main experimenter view which allows her to write programs as well as start, supervise and cancel experiments.

When the experimenter view is first displayed, the experimenter is asked for a password. By default each new realm is created without a password. In order to specify a password it can be entered at this point and experimenters

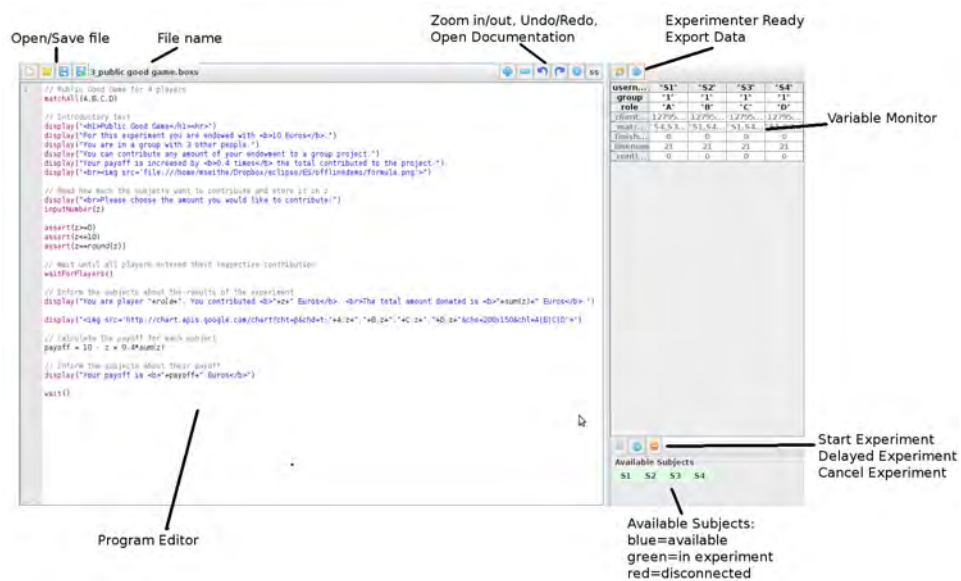


Figure 4.4: The experimenter view.

are required to enter it whenever they reconnect to this realm in the future. Setting a password is *strongly recommended* as ill-meaning and well-informed subjects could specify it otherwise and prevent the legitimate experimenter from accessing her experiment.

The main component of the experimenter view is the program editor in the left part into which experimenters can enter their programs. The program editor allows copy and paste as well as undo and redo. It also provides automatic syntax highlighting, which enhances the readability of the program code. The buttons on top of the program editor allow the experimenter to load and save programs and to change the font size of the editor.

On the bottom right of the client view a list of all the subjects who are connected to the experiment is displayed. This list is updated whenever new subjects join the experiment or the status of a subject changes. Subjects who are available for an experiment are displayed in green, subjects in an experiment are displayed as blue and subjects who were disconnected during

an experiment and suspended are shown in red. On top of the subject list buttons to start a normal experiment, to start a delayed experiment and to cancel an experiment are provided.

The variable list in the top right area displays all data generated by the currently running experiment in a table with each subject in a separate column. The data displayed in this table can be exported to a comma-separated values file (CSV) by clicking on the export button on top of it.

The separation between the program editor and the variable view can be dragged by the experimenter to suit her layout preferences.

4.2.4 Internet Experiments

In the tutorial both the experimenter and the subject clients are executed on the same computer. Conducting a real experiment with other people is relatively straightforward. In order to conduct an internet experiment, one can copy the subject link displayed on the 'Available Setups'-page and send it to the desired subjects. For example:

```
http://boxs.uni-bonn.de/expsys/es_subject.html?host=boxs.uni-  
bonn.de&port=58000&realm=1963527&email=&username=new
```

The link contains the realm id and the server data required for participating in the experiment. If another person opens this link in her web browser, she shows up in the subject list with the user name specified in the link. The experiment can then be started by pressing the start-button in the experimenter's view as described in the tutorial. Information on starting the experiment automatically is provided in section 4.2.7.

4.2.5 Laboratory Experiments

Laboratory experiments using the BoXS work very similar to internet experiments. In the beginning the subject link copied from the 'Available Setups'-page has to be opened on each computer in the laboratory. The user name

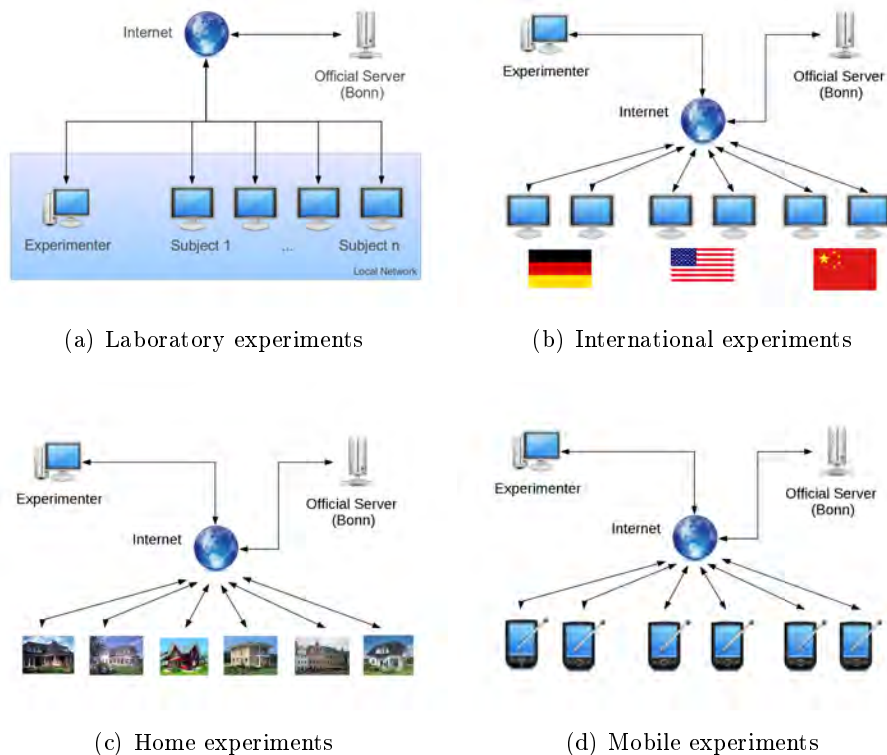


Figure 4.5: Possible applications for the Bonn Experiment System.

should be changed to reflect each computers' cubicle/room number in order to correctly identify the subjects and their computers later on. As copying this link can be quite tedious, it is generally a good idea to bookmark the link on every computer so that it can be reused for future experiments.

Like in the internet experiment example the subjects show up in the experimenter's available subjects list and the experiment can be started by clicking on the start-button.

4.2.6 Using an Offline Server

Usually the official BoXS server is recommended for all experiments as it is the most convenient way. There are some cases, however, in which the set-up and use of a local BoXS server can be advantageous. Experiments for which no internet connection is available, for instance due to technical restrictions

or restrictive laboratory policies, are a good example. Another kind of situation are high-frequency experiments where extremely fast reaction times and very low latencies are required.

The package required for running a local BoXS server can be downloaded from the general information section of the homepage which also includes some tips on how to set it up. In a nutshell, the experimenter needs to execute the downloaded BoXS server on one of the computers. Then the official server's name (`boxs.uni-bonn.de`) on the participation links has to be substituted by the IP address of the computer running the server. Afterwards, everything should work like when using the official internet server with the exception of the email functionality.

4.2.7 Autorun Experiments

Some experiments, especially internet experiments, require to be run while the experimenter is not available. Consider the case in which the experimenter wants participants to fill out an online questionnaire during a certain time period. Doing this with the methods discussed previously would require the experimenter to sit in front of her computer and manually start an experiment whenever a participant connects to the BoXS.

In order to simplify this process so-called autorun experiments have been implemented. An autorun experiment is created by writing a program as usual and clicking the blue 'autorun'-button when done. The experiment is now stored on the server and the experimenter can turn off her computer without affecting it.

Whenever a subject with the appropriate realm id logs onto the server, the stored experiment is automatically executed. The data of the experiment, including all previous observations, is sent to the experimenter by email after each completed observation. Note that a valid email address *has to be specified* at the beginning of the experiment in this case.

4.2.8 Troubleshooting

The following two problems are encountered frequently when using the BoXS and can be solved easily:

- If nothing is displayed after clicking on a link on the 'Available Subjects'-page, the Java plug-in is probably not properly installed. The Java plug-in is available for free and most web browsers notify the user in case it is missing and aid her in its installation. Otherwise it can be installed manually by visiting the Java homepage at www.java.com and downloading and installing the Java Standard Edition Runtime Environment (JRE).
- If a message claiming that clients cannot connect to the server is displayed despite a working internet connection, the experimenter's institute's firewall is probably at fault. In order to resolve this, the corresponding IT department should be kindly requested to open the ports 58000 and 58001, which are used by the BoXS, for TCP connections.

4.2.9 Documentation

Several ways are available to learn more about how to use the Bonn Experiment System:

- The appendix of this chapter as well as the largely equivalent online documentation provide an elaborate documentation for each command available in the BoXSPL:

<http://boxs.uni-bonn.de/documentation/index.html>

- The documented example programs, which are printed in the appendix of this chapter and can be downloaded on the web site provide examples for how the BoXS can be used and how common experiment types can be realised:

<http://boxs.uni-bonn.de/examples/index.html>

- The frequently asked questions section on the web site contains a big list of answered questions and is a good place to start when problems and questions are encountered:

<http://boxs.uni-bonn.de/general/index.html>

- A public mailing list exists where all users can ask questions and are invited to contribute to the general discussion:

<http://groups.google.com/group/bonn-experiment-system>

- Video tutorials demonstrating the basic features of the BoXS are available on the homepage and demonstrate how to do the most common tasks using the BoXS.

4.3 The BoXS Programming Language

In this section I describe the thought process behind the design decisions met concerning the BoXS Programming Language (BoXSPL). The goal of the BoXSPL is to create a language which is easy to learn for novice users while still allowing the implementation of most experiment types. This section intends to provide an overview of the BoXSPL. For more information on the commands and concepts described in this section please refer to the appendix or the official homepage where more elaborate documentation is available.

4.3.1 Code Based and Graphical Approaches

While most professional programming languages like C++ and Java are purely text based programming languages, languages designed for novice programmers like z-Tree or Regate provide strong graphical user interfaces for designing a program. The advantage of graphical approaches is that they may be easier to learn and less intimidating for novice users as standard experiment types like questionnaires can often be created without even writing a single line of code. In more complex experiments, however, the

experimenter is usually required to write program code at some point either way.

Text based languages provide advantages for advanced users as it is usually faster to type a desired command using the keyboard than to create it using a graphical interface. Sophisticated users may furthermore take advantage of features like copying and pasting and are free to choose any text editor they like. Another advantage of text based languages is that their programs can be easily shared and archived, as they are compatible across versions and platforms, or published, as they can be easily printed.

With the BoXSPL I introduce a text based programming language. In order to ease the learning curve for novice users I provide a rich documentation, several sample programs and an editor with syntax highlighting. I also provide a tutorials and videos to reduce the time and effort required to get new users started with the BoXS and create a first experiment.

4.3.2 Program Execution

This section describes how the BoXS server processes a program written by an experimenter and how it is executed.

Lexing and Parsing

In computer science, a lexical analyser (lexer) is an algorithm which reads a given text string and translates it into a set of tokens, for example string tokens, numbers and operators³. These tokens are then handed over to a syntactic analyser (parser), which analyses and structures the tokens and, as a final step, arranges and translates them to a format which can be executed⁴.

When the BoXS project was initiated as a small prototype, a hand-made

³See Wikipedia, http://en.wikipedia.org/w/index.php?title=Lexical_analysis&oldid=366935008

⁴See Wikipedia, <http://en.wikipedia.org/w/index.php?title=Parsing&oldid=373059757>

simple lexing/parsing-algorithm was implemented. As the complexity of the language increased and more test cases were created, the stability, quality and performance of the lexing/parsing process has been steadily enhanced and improved. An alternative to hand-made lexers are so-called lexer- and parser-generators for Java, for example JLex and CUP⁵, which are freely available. These generators process a given language specification, which can be enhanced and changed later on, and create lexer- and parser-code which can be included in any program.

The main advantage of using a such a professional lexer/parser generator is the high reliability and robustness of the resulting algorithm. Furthermore it simplifies the future documentation and enhancement of the underlying language specification. In the long run the transition to a lexing/parser generator seems advantageous. However, as the required changes would likely incur a lot of initial instability as the lexing/parsing process is vital for the BoXS, the migration process has a relatively low priority at the moment.

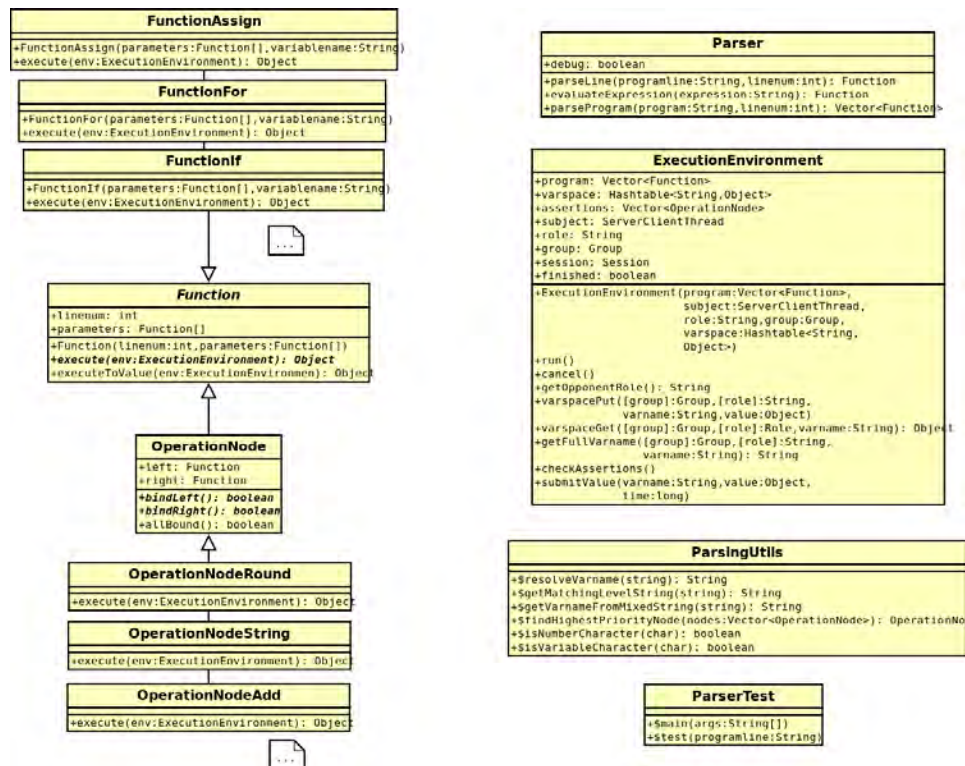
Internal Implementation

Lexing, parsing and execution are done in several steps. An overview of the classes involved in the process is shown in figure 4.6. First, the complete program is partitioned based on the matching and flow control commands (if, for and while) contained in it. In the next step, the resulting code is scanned line by line and translated into corresponding Function-objects. These objects are then filled with the data required for their proper execution at runtime.

For instance, the line `var=round(15/4+6)` is translated into the following tree-like structure:

```
FunctionAssign ("var", OperationNodeRound (OperationNodeAdd
      (OperationNodeDivide (OperationNodeDouble:15.0,
```

⁵See <http://www.cs.princeton.edu/~appel/modern/java/JLex/>.



Note: Some attributes, operations and classes are omitted from the diagram in order to improve readability.

Figure 4.6: Class diagram of the lexing/parsing classes.

OperationNodeDouble:4.0),OperationNodeDouble:6.0)))

The FunctionAssign-object, which is on top of the hierarchy, assigns a value to the variable named var. In order to calculate the correct value for this it executes the OperationNodeRound-object which in turn executes and evaluates objects further down in the hierarchy.

As a result of the lexing/parsing process, a program which is entered as a text string is converted into a vector of Function-objects, which can each reference one or more related Function-objects. In the first versions of the BoXS this conversion, which is arguably the most computationally intensive and complex process in the BoXS, was done while the experiment was running. In order to improve performance this process is now done before the experiment is executed, which vastly improves the execution performance

in more complex experiments. At runtime, the BoXS calls the `execute`-methods of all Function-objects, which are implemented as very fast and efficient operations.

In order to ensure the proper functioning of the BoXS lexer/parser a suit of critical test terms and expressions has been collected which is executed and tested before a change is incorporated into the official BoXS server. Every time an internal error in the BoXS is found, a corresponding expression is added to this test collection in order to ensure that this error is not accidentally reintroduced in a future version.

Error Handling

Unfortunately not all programs written by experimenters are flawless. There are two categories of errors which can occur when executing a user-written program. The first category contains so-called compile-time errors which prevent the program from being lexed and parsed correctly, for instance misspelled commands, missing brackets or other types of syntax errors. The second category consists of so-called runtime errors which occur and can only be detected while the program is executed, for example the referencing of undefined variables or an invalid mathematical operation.

Both compile-time and runtime errors which occur when running a program in the BoXS are reported to the experimenter and displayed in a separate window, including the line which caused the error. Error messages serve the purpose of informing the experimenter about mistakes in her program and making her aware of possible implications.

When compile-time errors are encountered, the BoXS only shows the error message and does not start the experiment. When runtime-errors occur, the philosophy of the BoXS is to keep the experiment running whenever possible and only halt the execution for subjects who are directly affected by the error.

```
display("Please enter your age:")
inputNumber(age)
assert(age>=10 && age<=100)
display("Please enter your gender:")
choice(gender,"male","female")
wait()
```

Figure 4.7: Example questionnaire.

Furthermore two specific types of possible errors do not raise error messages: 1) Referencing an undefined variable does not result in an error but returns the numerical value 0. The rationale for this is that it makes programs significantly shorter by eliminating the need to initialise every variable (for example `counter=0`). 2) Some questionable mathematical expressions, for instance `var=1/0`, does not result in an error message but in the pseudo-value `Infty` (infinity), which may produce odd results when used for further calculations.

4.3.3 Implemented Functionality

One important process in creating a programming language is to find the right compromise between its accessibility and its generality. While a simple language with only a few commands might be very appealing to novice users, a lack of functionality would narrow down its possible applications.

Before a description of the functions implemented in the BoXSPL is provided, consider the questionnaire example program shown in figure 4.7 for an impression of how a typical BoXS-program looks. A typical program includes `display`-commands to display instructions and questions, includes some input commands like `inputNumber` and `choice` and ends with a `wait`-command.

Figure 4.8 shows the set of functions which are implemented in the first version of the BoXS Programming Language (BoXSPL) grouped by function.

Displaying on the subjects' screens display, video
Requesting input from the subjects inputString, inputNumber, choice, assert
Basic algebra + - / * % < > >= <= == != !
Mathematical functions log exp round sin cos tan
Random number generation randomUniform, randomUniformInteger, randomGauss
Controlling the program flow if, while, for
Waiting for the subjects wait, waitForPlayers, waitForExperimenter, waitTime
Matching subjects into groups matchManual, matchAll, matchStranger, matchPerfectStranger

Figure 4.8: List of all functions implemented in the BoXS.

The functions allow for most experiment types and questionnaires. Each function is designed to have a clear purpose and be easy to understand.

4.3.4 Basic Calculus

On the most basic level the BoXSPL includes the most common mathematical functions as well as string concatenation. It can evaluate arithmetic expressions, calculate with integer and real numbers at double precision and understands the use of brackets. Furthermore the BoXS can generate uniformly and normally distributed pseudo-random numbers based on the linear congruential generator implemented in Java⁶. It also provides program flow control commands in the form of an **if**-command for conditional execution as well as a **for**- and a **while**-command for repeated execution.

⁶See http://download.oracle.com/docs/cd/E17476_01/javase/1.4.2/docs/api/java/util/Random.html.

4.3.5 Variables

The BoXS provides a very flexible data structure which allows for variables with different scopes, i.e. local, group-specific and global types, as well as arrays and matrices of arbitrary dimensions.

Internal Data Representation

The BoXS uses a so-called HashMap-object to store all data generated by each experiment as it provides a very flexible way of data storage. A map in computer science is a general data structure which can store an arbitrary number of key-value pairs. The HashMap-class, as provided by the Java programming language, provides a very efficient implementation of such a map by generating hash codes for each key in order to reduce the time required to access stored data.

The keys used in this map are the variable names, which are stored as a string, and the corresponding values are arbitrary objects. In the current version these objects are either strings or double precision numbers. In future versions this might be used to store more complex objects like lists or images.

Local, Group and Global Variables

In order to ensure that all data is stored unambiguously, a variable name needs to be transformed and resolved internally before a variable is stored. The variable name `payoff`, for example, would be problematic as it would be unclear to which subject the payoff belongs. In order to avoid this, each variable name is internally prefixed by the respective subject's username, which is always unique⁷ for each experiment⁸.

⁷The server always ensures that the usernames of the subjects are unique. If several subjects login with the same username, they are renamed internally by adding the suffix `_number`.

⁸In the first versions of the BoXS, variable names stored in the format of `group.role.varname`. This turned out to be problematic, however, as the re-matching of subjects would lead to all variables getting mixed up.

(Suppose there is one group (1) with two subjects (S1 and S2) in roles A and B.)

Assignment for ...	Program Line	Internal Representation
... current subject	<code>var=5</code>	<code>S1.var=5</code>
... subject A in current group	<code>A.var=5</code>	<code>S1.var=5</code>
... subject B in current group	<code>B.var=5</code>	<code>S2.var=5</code>
... subject B in current group 1	<code>1.B.var=5</code>	<code>S2.var=5</code>
... all subjects in current group	<code>*.var=5</code>	<code>S1.var=5, S2.var=5</code>
... all subjects in group 1	<code>1.*.var=5</code>	<code>S1.var=5, S2.var=5</code>
... all subjects in all groups	<code>*.*.var=5</code>	<code>S1.var=5, S2.var=5</code>

Table 4.1: Local, group and global variable examples.

(Suppose there is one group (1) with two subjects (S1 and S2) in roles A and B.)

	Program Line	Internal Representation
specific index	<code>var[3]=5</code>	<code>S1.var[3]=5</code>
calculated index	<code>var[1+2]=5</code>	<code>S1.var[3]=5</code>
string index	<code>var["A"]=5</code>	<code>S1.var[A]=5</code>
string index	<code>hello["german"]="Willkommen..."</code>	<code>S1.hello[german]=...</code>
variable index	<code>var[experimentround]=5</code>	<code>S1.var[3]=5</code>
variable index	<code>var[role]=5</code>	<code>S1.var[A]=5</code>
3-dimensional	<code>var[1][2][3]=5</code>	<code>S1.var[1][2][3]=5</code>

Table 4.2: Array and matrix examples.

If no specific prefix is specified by the experimenter, a variable is treated as a local variable which means that it only applies to the current subject. Therefore the line `payoff=5` only sets the current subject's payoff to 5. In order to change another players variables or to do group-specific or global⁹ changes a prefix has to be used. This allows the experimenter to create pseudo-global variables and share variables among subjects. Table 4.1 shows some examples how this can be done in the BoXSPL.

Arrays and Matrices

The BoXSPL allows arrays and matrices of arbitrary dimension to be stored. Table 4.2 shows some examples of what can be done with this.

The reason why the BoXS is so flexible with respect to arrays is that they are not stored as arrays internally. They are stored in the very same HashMap where all data is stored. The array indices are evaluated at runtime

⁹Note that this so-called global level is specific to the current realm. Cross-realm communication is not possible for obvious security reasons.

and appended to the variable name. In effect, a one-dimensional array with a length of 5 is stored like 5 separate simple variables.

Arrays do not need to be defined ahead of time and their dimensions can be arbitrarily changed at runtime. Furthermore both number and string indices are allowed, which is very useful in some situations.

Automatically Generated Variables

The BoXS automatically creates several variables during the execution of an experiment. While some of these variables are only required for the internal execution process, some variables may be interesting for experimenters. Most automatically generated variables are prefixed with “_” in order to avoid confusion.

- `_linenum`: The number of the line in the program which is currently being executed (usually a wait-command).
- `_finished`: 1 if the experiment has finished for this subject, 0 otherwise.
- `_continue<linenum>`: 1 if the subject has clicked successfully on the wait-button specified in the given line, 0 otherwise.
- `_clientdisplaytime<linenum>`: The exact time¹⁰ at which a stage was displayed on the respective subject’s screen. This may be useful for experimenters in order to synchronise the BoXS to other devices based on the time. The line number specifies the wait-command which triggered the stage in question.
- `_inputhistory_<varname>`: This variable stores every input made by the subjects. This allows the experimenter to learn about the decision

¹⁰In milliseconds since January 1, 1970, 00:00:00 GMT, see: http://download.oracle.com/docs/cd/E17476_01/javase/1.4.2/docs/api/java/util/Date.html.

process and possible choice revisions, as well as the response times. The data is stored as a comma-separated string where each action is formatted as `<time>.<input>` and where `time` is the number of milliseconds since the current stage was displayed on the subject's screen and `input` is the value entered by the subject at that time. This feature can be disabled using the `disableInputHistory()`-command if the data is not required.

4.3.6 Displaying Information and User Input

An experiment software needs to enable the experimenter to both present instructions or questions on the subjects' screens as well as receive their input. In the BoXSPL several commands are available to achieve this.

Each command is processed on the server by evaluating variables and solving calculations and distributed to the clients where it triggers the creation of a corresponding graphical components like a text boxes or a buttons. After creating all components for a screen they are, by default, vertically aligned and displayed. In general the BoXS client tries to recycle components and realise each stage with as few changes as required in order to increase the performance and reduce possible flicker effects in experiments where the information to be displayed changes frequently, for example in market experiments.

In case the components do not fit the screen's height, a vertical scrollbar is displayed, which allows the subjects to view components which do not fit on the screen. Horizontal scrollbars are not shown.

Displaying Text and Graphics

For displaying instructions, graphics and other types of data the BoXSPL provides the `display`-command. The `display`-command, as well as several other commands, supports the Hyper-Text Mark-up Language (HTML) and

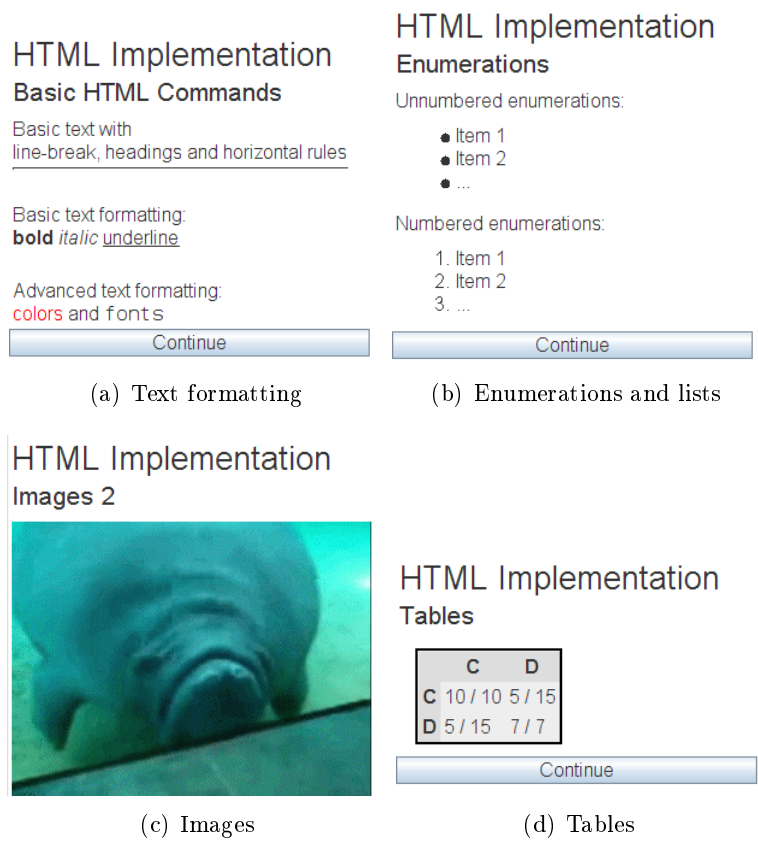


Figure 4.9: Examples using the `display`-command.

provides a great amount of flexibility.

The HTML format, which is also used for website programming, is both popular, flexible and relatively easy to learn. Besides simple text, HTML allows for text formatting, lists, enumerations, tables and images. The image formats which can be used include the standard formats JPEG and PNG as well as animated GIFs which can be used for displaying moving images on the subject’s screen. Figure 4.9 shows some examples of what can be achieved by using HTML formatting.

By default, a modern style using sans-serif fonts and a compact layout is used in the BoXS, which is likely sufficient for most experimenters. If required, more advanced experimenters can use advanced Cascading Style Sheet (CSS) formatting to further customize the BoXS’s appearance. CSS

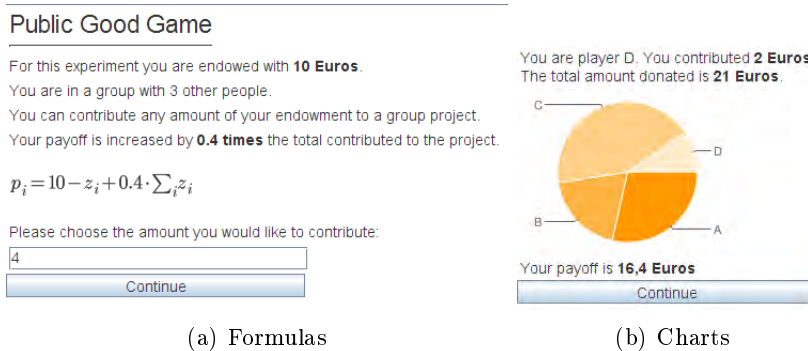


Figure 4.10: More examples using the `display`-command.

code can either be used within a `display`-command, which changes the appearance for this exact element, or globally by using the `style`-command, which is not discussed in this document.

Since the BoXS is usually run over the internet, several kinds of services which are available on the internet can be used within an experiment to gain access to additional functionality. In the example program shown in figure 4.10 this is used to include a mathematical formula, which is generated from TEX-code using a Google service, as well as a chart based on data generated in an experiment and visualised using the Google Chart API¹¹.

Videos

The BoXSPL also includes an experimental `video`-command which can be used to include video and audio files into an experiment. In order to use it the experimenter needs to provide a video or audio file in a format that is compatible to the Java Media Framework (JMF)¹². Using the JMF has the advantage of true platform-compatibility but unfortunately also introduces some restrictions.

First, using the `video`-command requires the subject clients to be able to

¹¹See <http://code.google.com/intl/de-DE/apis/chart/>.

¹²See <http://java.sun.com/javase/technologies/desktop/media/jmf/2.1.1/formats.html>.

access the Java Media Framework, which has to be specified when starting the clients and can slow down the starting process significantly. Second, the video and audio codecs supported by the JMF are not very satisfying as they only support relatively dated and inefficient compression algorithms, which leads to poor video quality, a big transfer size or both. Third, the default Java security settings are very strict and forbid applets to access local videos which requires the the experimenter to edit the Java security settings on each subjects' computer¹³.

For the above reasons the `video`-command is to be considered experimental at this stage and its use is generally discouraged. Moving pictures without sound can be easily achieved using animated GIFs in the `display`-command.

Unfortunately the basic problems regarding the JMF is not likely to be solved in the near future. However, as the HTML format is currently being expanded to include video, it is likely that future versions of Java may deliver a less complicated way to use videos.

Subject Input

The current version the BoXSPL provides four commands for requesting subject input, that is the `inputString`- and `inputNumber`-commands for string and numerical input, as well as the `choice`- and a `checkbox`-commands for selections. These commands create appropriate graphical components on the subject's screen and send every input made by a subject to the server where it is processed. A list of all available input commands as well as documentation on their usage is provided in the appendix.

¹³For instructions on how to do this see the online documentation of the `video`-command.

Waiting and Assertions

When executing an experiment the BoXS continues until it encounters a `wait`-command. When a `wait`-command is encountered all previous statements in the program are executed and displayed to the subject, as well as a 'continue'-button. The experiment execution is then halted until the subject enters the required information of the respective stage and clicks on the 'continue'-button. The `assert`-command can be used to specify additional restrictions, for example a maximum value for an input variable.

Besides the `wait`-command the BoXSPL includes a `waitTime`-command, which waits for a specific time, a `waitForExperimenter`-command, which waits until the experimenter clicks on a button and a `waitForPlayers`-command, which waits until all subjects of a subject group have clicked on their respective 'continue'-buttons.

4.3.7 Matching

Matching is the process by which an experiment system assigns subjects to groups and gives them roles which are unique for each group. In an economic trust game, for example, the subjects would be partitioned into groups of two where each group designates one subject as the 'investor' and the other subject as the 'trustee'.

The most basic matching type which is provided by the BoXS is the manual matching (`matchManual(username,group,role)`), which allows the experimenter to precisely specify a group and a role for each subject. While this approach allows for the most customisation, it becomes increasingly messy and impractical as the number of subjects in an experiment increases.

The second and most common matching type is the alphabetical matching (`matchAll(roles)`), which sorts the subjects based on their user names and assigns them in alphabetic order. The experimenter only needs to specify the names of the roles and the BoXS automatically creates as many groups

as possible. Note that the subjects are always assigned to the same groups if they are rematched using the same command.

Some experiment designs require a so-called stranger matching which ensures that subjects are matched to random subjects in subsequent parts of the experiment. The so-called perfect stranger matching furthermore requires that a subject is never matched into a group which contains a previous 'group-mate'. The BoXS provides the `matchStranger(roles)`- and the `matchPerfectStranger(roles)`-commands to execute these types of matching.

A perfect stranger matching requires a surprisingly high amount of calculation in order to determine the matching order which guarantees the most possible matches. Due to this computational complexity the BoXS uses matching tables, which drastically reduces the time required for the matching but restricts the matching to combinations which have been pre-calculated. A list of all pre-calculated perfect stranger matches is provided in section 4.A.6 in the appendix of this chapter.

The matching specified by any of the above matching commands is preserved until the `matchClear()`-command is called. Afterwards a new matching can be started.

If no matching is specified by the experimenter, the BoXS by default assigns all subjects into groups with one player each.

4.4 Design and Implementation

This section describes the technical aspects of the Bonn Experiment System (BoXS). In the first part of this section I discuss basic decisions made in designing the Bonn Experiment System, that is the choices of the programming language, the network architecture and the communication protocol. The following sections describe how both the server and the client of the BoXS have been designed and how they work internally.

4.4.1 Programming Language

The task of a programming language is to allow human programmers to write computer programs which can then be translated (compiled) to a native format which is executable by computers. Today several hundred programming languages exist, each designed to satisfy certain needs, for example high performance, platform independence or the support of complex scientific functions¹⁴.

The most popular and mature programming languages which are used for application programming at the time of writing this chapter are C++, Visual Basic and Java.

C++, first designed in 1979 by Bjarne Stroustrup, is probably the most widely used programming language for applications and video games today. Program code written in C++ can be compiled for most platforms and usually performs very well. However, the program code has to be specifically compiled into native code and distributed for every target platform. For example, a program compiled for Microsoft Windows can not be executed under Linux or MacOS, or even on some other Windows versions.

Visual Basic is developed by Microsoft and is designed to be easier to use than C++. At the same time it is the most restrictive programming language as programs developed using it are restricted to the Microsoft Windows operating system and can not be used on other platforms.

Java, which was first published by Sun in 1995 is based on the “Write Once, Run Anywhere”-philosophy, which allows the programmer to write and compile a program once and execute it on every platform. In order to make this work Java programs are not compiled to native code but to an intermediate byte code. This byte code is then executed by the so-called Java Virtual Machine, which is available for almost all platforms. Today Java

¹⁴See Wikipedia entry for “Programming Language”:
http://en.wikipedia.org/w/index.php?title=Programming_language&oldid=371608777

is very popular, especially for internet applications, and the Java Virtual Machine, which is required for executing Java applications, is pre-installed on most computers and can be installed for free otherwise.

One common misconception about Java is that it is slower than other languages because of the additional translation process required during the program execution. While this criticism was justified for early Java versions, the modern Java Virtual Machines have become much faster and perform just-in-time-compilation, which means that the parts of the program which are most important for its performance are automatically compiled into native code at runtime¹⁵.

The BoXS uses Java as the programming language for both the server and the clients in order to ensure full cross-platform compatibility, even within the same experiment. This has the advantage that it allows the BoXS to be used in internet environments. It also allows laboratories using the BoXS to freely choose the operating system for its computers, for example allowing the use of open operating systems like Linux, which may be used to reduce the costs required to set-up and administrate the laboratory computers.

4.4.2 Network Architecture

There are two approaches to design a network architecture. The client-server approach designates one central server computer to which all so-called clients connect. If clients want to share information in this architecture, they have to send it to the server, which then processes and/or distributes it to the appropriate receivers. The peer-to-peer approach tries to minimise the role of the server and emphasises direct connections between different clients. It has become very popular for file sharing as it provides a high bandwidth and reduces the need for powerful and costly servers. In general, the peer-to-peer

¹⁵See Wikipedia entry for 'Java':
[http://en.wikipedia.org/w/index.php?title=Java_\(programming_language\)&oldid=372353698](http://en.wikipedia.org/w/index.php?title=Java_(programming_language)&oldid=372353698)

approach does allow a higher bandwidth as well as a slightly lower latency.

The BoXS uses a client-server architecture for reasons similar to the ones described in Fischbacher (2007). First of all, the need for a server in an experiment system is hard to eliminate as subject registration, the matching of the subjects, the distribution of the experiment programs and the collection of the resulting data are intrinsically central processes and are best implemented using a server. While it would be possible to add peer-to-peer elements to the network architecture, the slight advantages in speed would probably not justify the resulting increases in complexity and effort. High bandwidth is not an important requirement for most experiments and the latency is usually low enough in the client-server approach to be hardly noticeable both in local networks and over the internet.

Basically the network structure resembles that of z-Tree with one exception. While the z-Tree program (as opposed to the z-Leaf) includes the server as well as the experimenters' user interface, the two roles are separated in the BoXS. The BoXS server, which is described below, can therefore be executed either on the experimenters' computer or on a separate computer, for example on the official server.

4.4.3 Communication Protocol

The internet and most local networks support two major communication protocols. The Transmission Control Protocol (TCP) and its extension TCP/IP are widely used for most internet applications like web browsing and sending emails. It provides a high degree of reliability and guarantees the arrival of the transmitted data packages between sender and receiver in the right order. The disadvantage of TCP/IP is that it incurs a significant latency, especially if packages become corrupted or delayed¹⁶. The User Datagram

¹⁶See Wikipedia entry for 'Transmission Control Protocol (TCP)':
http://en.wikipedia.org/w/index.php?title=Transmission_Control_Protocol&oldid=371744160

Protocol (UDP), as opposed to the TCP, does not guarantee the correct order or even the correct arrival of data packages. Instead it provides a fast transmission speed and a low latency. The UDP is widely used for real-time applications like live audio or video streams and online games. The reasoning for this is that for a game or a voice transmission a missing package may not be perceived as bad as a constant lag which would result in a delayed playback¹⁷.

Java supports both TCP and UDP sockets and connections. As communication based on the UDP protocol does not guarantee the correct arrival of sent packages, the programmer using it has to provide additional algorithms to account for cases in which packages were transmitted erroneously. Packages would have to be checked on arrival, unordered packages would have to be sorted and missing packages would have to be requested and sent again. As the correct and robust implementation of these functions is both tedious and non-trivial, using the TCP protocol was the obvious choice for the BoXS. Besides convenience, it is questionable if programming a secure data connection based on UDP can improve upon the corresponding mechanisms which are already implemented in the TCP.

The Bonn Experiment System uses two connections between the server and each client. While one connection for each client would be sufficient for both directions in principle, experience in developing the BoXS has shown that both performance, stability and latency of the connections can be improved by using separate connections for both directions as they allow for asynchronous data transmission.

¹⁷See Wikipedia entry for 'User Datagram Protocol (UDP)':
http://en.wikipedia.org/w/index.php?title=User_Datagram_Protocol&oldid=372018803

4.4.4 The Server

The main task of the server is to keep track of all its connected clients, to ensure the correct transmission of data within the system and recover connections in case of connection issues. Additionally the server has to parse and execute experiment programs, correctly match and assign subjects to the correct experiment sessions and provide a way for experimenters to control and manage their experiments.

One key feature of the BoXS is that it provides official servers which can be accessed over the internet and eliminate the need for experimenters to run and administrate their own server. In order to make this possible and attractive the server has to meet particularly strong requirements concerning robustness and security.

Server Robustness

In order to ensure the highest robustness possible, I decided to implement the BoXS as a highly multi-threaded architecture. A thread in programming is a part of a process which can be executed separately¹⁸. Programs can create several threads which are then 'forked' and executed independent from each other and at the same time.

One major advantage of using threads is that the crash of one thread does not necessarily affect the other threads. Furthermore multi-threaded programs take advantage of modern computer processors, which possess multiple processor cores and have the potential to run much faster as a result. The downside of multi-threading is that it requires a lot of sophisticated programming techniques to ensure that the threads are synchronised correctly and do not disturb each other or incur non-deterministic behaviour.

In the case of the BoXS server the main process's only task is to wait for

¹⁸See Wikipedia entry for 'Thread (computer science)':
http://en.wikipedia.org/w/index.php?title=Thread_%28computer_science%29&oldid=371822693

and accept incoming connection attempts from clients. After a client connects successfully, all subsequent communication is handled by a communication thread which is immediately forked and started. Additional threads are forked for each experiment and each subjects' role in an experiment. Therefore the malfunctioning of one thread can effect neither the vital functions of the server nor the execution of other experiments.

In order to ensure that the resources of the server are shared evenly across the different experiments, several mechanisms are in place to detect and interrupt programs which get trapped in an infinite loop and consume too much processing power as a result.

So far the server program has proven to be very reliable. It should be noted, however, that no severe stress tests have been done to date. If many experiments with high levels of interaction were to run at the same time, the speed of the server might decrease and the available memory might get depleted. For such experiments the use of a separate server is recommended.

Connection Robustness

One worrying thought might be that subjects or even the experimenter temporarily lose their internet connection during an experiment. While the mechanisms described in the previous section already ensure that this does not affect the remaining subjects, the thought of the subjects' data being lost is not pleasing.

The BoXS offers the possibility to reconnect both subjects and experimenters who lost their connection and resume the experiment at almost the exact same position where they left. In order to do this, the BoXS suspends and stores each client session which gets interrupted during an experiment for up to 24 hours. When a client tries to connect to the BoXS and provides realm and subject ids which match those of a suspended session, the session gets reassigned to the client and resumed. The server then ensures that the

reconnected subjects' clients are updated by sending them the most recent experiment state.

Security

Both experiment designs and experiment results contain a lot confidential data both from the experimenter and from the subjects. This is especially important for the BoXS as it a) uses the internet as its medium, b) several experimenters work on the same server at the same time and c) the subjects use the same software client as the experimenter.

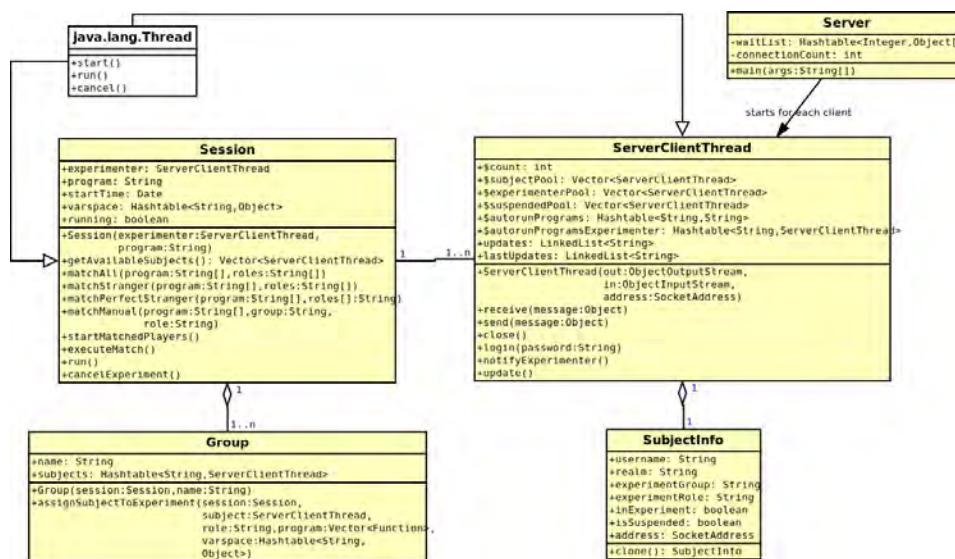
The first mechanism implemented to ensure that this data cannot be accessed by unauthorized persons is the so-called realm id, which is a string or a number specified by the experimenter at the beginning of the experiment. This exact realm id has to be entered on every other computer which is intended to enter the experiment. This ensures that the subjects and experimenters as well as the corresponding data of different experiments do not get mixed up.

The second mechanism is an experimenter password which can be specified by the experimenter and ensures that only the experimenter creating the realm or someone entrusted with the password can access the subjects' clients and their data.

Limitations While these two mechanisms provide sufficient security for most environments, experimenters should be aware that no extensive security checks were done on the BoXS.

In the current version of the BoXS data is transmitted between the server and the clients without being encrypted, which might allow a man-in-the-middle attack by a sufficiently sophisticated and motivated hacker.

Additional security features might be implemented in future versions of the BoXS.



Note: Some attributes, operations and classes are omitted from the diagram in order to improve readability.

Figure 4.11: Class diagram of the main server classes.

Notes on the Implementation

Figure 4.11 shows a Unified Modelling Language (UML) diagram of the most important server classes. The main process is the main()-function of the Server-class, which is called when the server is started and opens the TCP ports 58000 and 58001 in order to wait for incoming connections. Whenever a client connects to both of these ports, a ServerClientThread is created, forked from the main process and started.

The ServerClientThread-object provides all functions required for the server to communicate with a specific client and manages the login process as well as the information about the connected client, which is stored in a SubjectInfo-object. While being executed, the ServerClientThread-object waits for and processes data sent by the client as it arrives, for example input generated by the subject or an experimenter's program.

Whenever an experiment is started by an experimenter, a Session-object is created which contains both the experiment program as a string as well

as a variable space in which all data generated by the experiment is stored. The Session-object also contains the matching methods which create Group-objects and fill them with available subjects based on a specified matching rule.

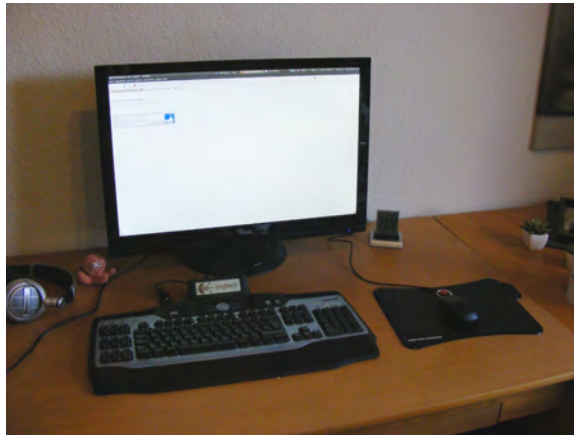
4.4.5 The Client

The client is the software which runs on both the experimenters' and the subjects' computers and is designed to provide an easily usable interface for both experimenters and subjects. For experimenters it must provide the means to write, execute and supervise an experiment, as well as the possibility to receive and store data generated by an experiment. For subjects it must graphically display the current stage of the experiment, as specified by its experimenter's program, as well as receive and transmit the user input generated by the subject to the server, where it is processed. Note that both the experimenter and the subject clients are reliant on their connection to the server to fulfil their task.

Implementation as Java Applet

As one aim of this project is to make the system as universal and flexible as possible, the client software is implemented using the Java Programming Language, more specifically as a Java Applet. As previously described, the Java Programming Language allows the generation of program code which can be run on every operating system and every platform. Figure 4.12 shows the same client running in different environments and on different operating systems.

An applet is a program which can be executed within a web browser without prior installation. Java is the most common choice for programming applets and is supported by most internet browsers and used by many web sites to provide advanced functionality ranging from small tools like stock



(a) Desktop PC, Ubuntu Linux



(b) Netbook, Windows



(c) MacBook, MacOS

Figure 4.12: The BoXS client.

and news tickers to large applications like text and spreadsheet applications or even games.

In the case of the BoXS, implementing the clients as Java applets allows them to be run on every operating system and every type of device as long as they have an internet connection and an internet browser supporting Java¹⁹, both of which are available on most computers.

Several applets can be executed at the same time, not only on the same computer but even within the same web page. This allows multiple instances to be run side-by-side, which is useful for testing in the BoXS as several subject applets can be easily simulated at once.

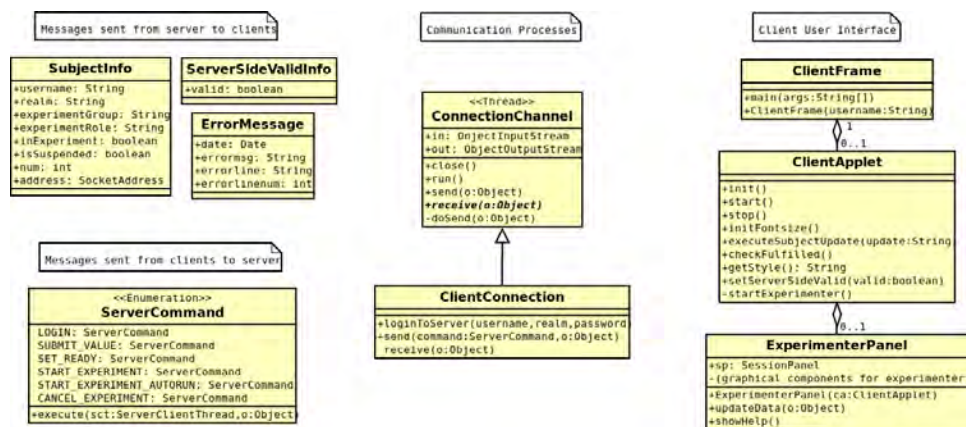
In the case of the BoXS, both the experimenters and the subjects use the same client applet. Whether the applet belongs to an experimenter or a subject is evaluated at runtime based on the user name and on whether the password provided by the user is correct. The size of the applet is approximately 80 KB which is sufficiently small to be loaded without delay on most computers.

Internal Implementation

Figure 4.13 displays an UML diagram of the client classes. The main class for the clients is the ClientApplet-class, an object of which is created for every BoXS client applet that is started. Alternatively it is possible to start the BoXS as an independent program without a surrounding internet browser by executing the ClientFrame's main()-function. In this case a window is created which contains a ClientApplet-object and behaves like an applet otherwise.

The ClientApplet-object manages both the internal behaviour of the

¹⁹At the time of writing this chapter, most web browsers designed for desktop PCs and notebooks already support Java. While this is not necessarily true for mobile devices yet, it is very likely that full Java support arrives within the next few years as both the computational power of these devices as well as their operating systems are quickly advancing.



Note: Some attributes, operations and classes are omitted from the diagram in order to improve readability.

Figure 4.13: Class diagram of the main client classes.

client as well as its graphical representation. When started it tries to connect to the BoXS server and either displays an experimenter’s graphical user interface, as specified by the ExperimenterPanel-class, or a subject’s user interface, depending on the login data. The ClientApplet also creates a ClientConnection-object which is the client analogue to the ServerClient-Thread and handles the connection between the client and the server.

The diagram also shows the objects used for the server-client communication in both directions. The SubjectInfo-object contains information about a subject’s identity and its current status. The ServerSideValid-object is sent by the server to signal whether the input made by a subject is valid. An ErrorMessage-object contains information on the line and the description of an error which was detected while parsing or executing a program. The ServerCommand is the only object sent from a client to the server and is used to start or cancel an experiment as well as to submit values entered by a subject.

4.5 Limitations and Future Development

I have received a lot of feedback and was able to implement most of the proposed features and improve many aspects of the Bonn Experiment System based on it. While I highly appreciate this feedback, I am unfortunately not able to implement all proposals made.

4.5.1 Feature Selection

The obvious reason is that every additional feature and every change requires a significant amount of time and effort to implement, document and test. Especially the latter should not be underestimated as even a slight change can have widespread effects which are often hard to anticipate. In order to decide whether a certain feature is implemented, I try to estimate the likely number of affected users (Is every experimenter affected or only a very small subset?), the severity of the lack of the feature (Does it make some experiment designs infeasible or is it merely an inconvenience?) as well as the expected implementation effort (Is it done in two hours or two weeks?).

Another reason why I am hesitant to implement some proposals is that every new feature adds to the overall complexity of the system. One of the major advantages of the Bonn Experiment System is that it allows a large number of experiments while requiring the experimenter only to learn a small set of commands. The more buttons, commands and tweaking possibilities a system has, the harder and more intimidating it might become.

4.5.2 Future Development

Obviously it is hard to predict how the development of the BoXS advances in the future. I am dedicated to get the current version error free and intend to continue developing it in the future. If I will no longer be able to support and maintain the BoXS any further, I will try to find a way to ensure possible

further development, either by publishing the software as open source or by handing it over to another researcher or programmer willing to take care of its future development.

4.5.3 Limitations

There are several features which were already requested and have made it to the wish list for future versions.

- **Functions:** Currently it is not possible for experimenters to create user-defined functions and procedures. While this is not very relevant for short experiments, the lack of user-defined functions can lead to unnecessarily long and messy programs in some cases, for example if the randomisation of stages is required.
- **External devices:** At the moment the BoXS offers no possibility to connect external devices like for example medical devices and input devices like joysticks.
- **Delay:** A slight lag exists between the execution of the program on a server and the point in time when it is displayed on the subjects' screens. While this lag is usually sufficiently small for non-time-critical experiments, it may be of importance for some experiments.

4.6 Conclusion

The Bonn Experiment System provides a novel and attractive way of designing and conducting laboratory and internet experiments. The possibilities to easily set-up and run experiments over the internet, to include web based content as well as advanced features like the measurement of response times allow for many new and exciting experiment designs and environments.

The possibility to execute the BoXS client applets without prior installation eases the set-up in lab environments and allows for experiments which

use existing infrastructure outside labs, including the subjects' computers. True cross-platform compatibility provides freedom of choice and possible support for mobile devices in the future.

The programming language BoXSPL is easy to use and easy to learn. The small number of commands as well as the available online documentations, tutorials and sample programs make the learning process easy for novice users and provide rich possibilities for advanced users. Several experiments have been conducted using the BoXS and the feedback received from the experimenters was very positive.

One exciting and unexpected example for how the BoXS expands the space of experiment possibilities was provided by a kind professor who wrote me about how he used the BoXS in his lecture to teach about experimental methods by programming ad-hoc experiments together with his students who could participate using their laptops.

4.A List of all Functions in the BoXS Programming Language

4.A.1 Basic Operations and Calculations

Basic Calculus

The BoXS compiler correctly evaluates $+$, $-$, $/$, $*$ and the modulus ($\%$). It also correctly derives the priority from brackets as required.

Example

```
1 display("<h2>Calculating</h2>")
2 display("12 + 8 = "+(12+8))
3 display("12 + 8 * 3 = "+(12+8*3))
4 display("18 / 6 = "+(18/6))
5 display("18 / (6-2) = "+(18/(6-2)))
6 display("5 * 3 + 2 = "+(5*3+2))
7 display("5 * (3 + 2) = "+(5*(3+2)))
8 display("15 % 4 = "+(15%4))
9 wait()
```

Calculating
12 + 8 = 20
12 + 8 * 3 = 36
18 / 6 = 3
18 / (6-2) = 4.5
5 * 3 + 2 = 17
5 * (3 + 2) = 25
15 % 4 = 3

Continue

More Calculus and Trigonometric Functions

The BoXS can calculate the natural logarithm (\log), as well as the exponential function (\exp), sine (\sin), cosine (\cos) and tangent (\tan). It also provides the functions round , round1 and round2 to round a number to 0, 1 or 2 decimals.

Example

```
1 display("<h2>Mathematics</h2>")
2 display("log(30)="+log(30))
3 display("exp(10)="+exp(10))
4 display("sin(5)="+sin(5))
5 display("cos(PI/4)="+cos(PI/4))
6 display("tan(1)="+tan(1))
7 display("round(log(30))="+round(log(30)))
8 wait()
```

Mathematics
log(30)=3.4011973816621555
exp(10)=22026.465794806718
sin(5)=-0.9589242746631385
cos(PI/4)=0.7071067811865476
tan(1)=1.5574077246549023
round(log(30))=3

Continue

Boolean Algebra

The BoXS can check for equality (==), inequality (!=) and compare (< , > , <= , >=). It knows the logic operations AND (\&\&) and OR (\|\|).

Example

```
1 display("<h2>Logic operations</h2>")
2 a=15
3 b=3
4 c=5
5 display("Equal:<br>(a == b * c) = "+(a==b*c))
6 display("Not equal:<br>(a != b) = "+(a!=b))
7 display("Less than/equal:<br>(a <= b) = "+(a<=b))
8 display("And:<br>(1 && 0) = "+(1&&0))
9 display("Or:<br>(1 || 0) = "+(1||0))
10 wait()
```

Logic operations
Equal:
(a == b * c) = 1
Not equal:
(a != b) = 1
Less than/equal:
(a <= b) = 0
And:
(1 && 0) = 0
Or:
(1 || 0) = 1

Notes

- The BoXS internally uses the number 0 as false while the number 1 is treated as true.

Random Number Generation

Currently uniformly and normally distributed random numbers are supported.

Example

```
1 display("<h2>Uniformly distributed in [0,1]</h2>")
2 display(randomUniform())
3 display(randomUniform())
4 display("<h2>Uniform integers in [5,10]</h2>")
5 display(randomUniformInteger(5,10))
6 display(randomUniformInteger(5,10))
7 display("<h2>Gauss distributed</h2>")
8 display(randomGauss())
9 display(randomGauss())
10 wait()
```

Uniformly distributed in [0,1]
0.6372758448375837
0.13205509632194734

Uniform integers in [5,10]
6
8

Gauss distributed
-0.4541906306469427
1.1714924549772257

Notes

- The random numbers are different for each subject. If they are supposed to be the same, they can be assigned to global variables.
- The numbers are generated using the internal Java random number generator, which is based on a linear congruential generator and produces pseudo-random numbers.

4.A.2 Program Flow Control

if(expression) { ... }

Tests if the expression is fulfilled (i.e. not equal to zero) and executes the code in brackets only if the expression is met.

Parameters

expression The expression which must be fulfilled.

Notes

- Note that each curly bracket needs to be in a single line of code.

while(expression) { ... }

The while-command executes a part of your program repeatedly for as long as a given expression is fulfilled (i.e. not equal to zero). Compared to the for-command it is slightly more versatile.

Parameters

expression The expression which must be fulfilled for the loop to be continued.

Example

```
1 display("<h1>Square numbers</h1>")
2
3 i=1
4 while(i<=10)
5 {
6     display(i+" * "+i+" = "+(i*i))
7     i=i+1
8 }
9
10 wait()
```

Square numbers

1*1 = 1
2*2 = 4
3*3 = 9
4*4 = 16
5*5 = 25
6*6 = 36
7*7 = 49
8*8 = 64
9*9 = 81

Example

```
1 i=1
2 s="<table>"
3 while(i<=10)
4 {
5     j=1
6     s=s+"<tr>"
7     while(j<=10)
8     {
9         s=s+"<td align=center width=40>"+(i*j)+"</td>"
10        j=j+1
11    }
12    s=s+"</tr>"
13    i=i+1
14 }
15 display(s)
16 wait()
```

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Notes

- Note that each curly bracket needs to be in a single line of code.

- In order to avoid infinite loops, execution is aborted when too many repetitions occur. An error message is given in this case.

for(initialization; condition; iteration) { ... }

The for-command executes a part of your program repeatedly for as long as a given expression is fulfilled. Compared to the while-command it is usually more compact and easier to use.

Parameters

initialization	Initialization code which is executed before the loop. Usually this is used to initialize a counting variable.
condition	The expression which must be fulfilled for the loop to be continued. Usually this is used to check if the counting value exceeds the number of desired repetitions.
iteration	Code which is executed after each repetition. Usually this is used to increase the counting variable.

Example

```

1 display("<h1>Square numbers</h1>")
2
3 for(i=1; i<=9; i=i+1)
4 {
5     display(i+" * "+i+" = "+(i*i))
6 }
7
8 wait()

```

Square numbers

```

1 * 1 = 1
2 * 2 = 4
3 * 3 = 9
4 * 4 = 16
5 * 5 = 25
6 * 6 = 36
7 * 7 = 49
8 * 8 = 64
9 * 9 = 81

```

Continue

Example

```
1 s="<table>"
2 for(i=1; i<=9; i=i+1)
3 {
4   s=s+"<tr>"
5   for(j=1; j<=9; j=j+1)
6   {
7     s=s+"<td align=center width=40>"+(i*j)+"</td>"
8   }
9   s=s+"</tr>"
10 }
11 display(s)
12 wait()
```

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

Continue

Notes

- Note that each curly bracket needs to be in a single line of code.
- In order to avoid infinite loops, execution is aborted when too many repetitions occur. An error message is given in this case.

4.A.3 Displaying Text and Graphics

display([message])

The display-command shows a message on the subject's screen. Due to its support of HTML commands it is very versatile and can be used to display most types of information.

Parameters

message The message to be displayed. The message can be both a constant string or a variable which is evaluated on the server.

Example

```
1 payoff=5+3*0.25
2 display("Your payoff is "+payoff+" Euros")
3 wait()
```

Your payoff is 5.75 Euros

Continue

Example

```
1 display("<h2>Formatting text</h2>")
2 display("Make text <b>bold</b>")
3 display(" or <i>italic</i>")
4 display(" or <u>underlined</u>")
5 display("Use <font color=red>colors</font>")
6 display(" and <font face=Courier>font</font>")
7 wait()
```

Formatting text

Make text **bold**

or *italic*

or underlined

Use **colors**

and **fonts**

Continue

Example

```
1 display("<h2>Using images on the internet</h2>")
2 file="http://www3.uni-bonn.de/die-universitaet/"
3 file=file+"uniartikel/stadtplan_kl.jpg"
4 display("<img src='"+file+"'>")
5 wait()
```



Example

```
1 display("<h2>Using tables</h2>")
2 table = "<tr><th></th><th></th><th></th><th></th></tr>"
3 table = table + "<tr><th>C</th><td>10,10</td><td>5,15</td></tr>"
4 table = table + "<tr><th>D</th><td>5,15</td><td>7,7</td></tr>"
5 display("<table>"+table+"</table>");
6 wait()
```

C	D
10,10	5,15
5,15	7,7

Notes

- This command supports full HTML-syntax for formatting the output if required. You can also use CSS formatting.
- You can include local and remote images (including animated GIFs). You can also use internet services like e.g. Google Charts to include additional functionality to your programs.
- Text is scaled to fit the resolution of the client screen. Images are not scaled, however.

4.A.4 Waiting

`wait([message],[messageafterclick])`

The wait-command creates a button on each subject's display. The experiment does not continue for the subject until the button is pressed. If the current subject screen requires the subject to do something, for example enter a number, the button is disabled until she does so.

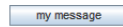
Parameters

`message` The message to be displayed on the button. If no parameter is given, a standard message is displayed (Continue).

messageafterclick The message to be displayed after the subject clicked on the button. If no parameter is given, a standard message is displayed (Please wait for the experiment to continue).

Example

```
1 waitForPlayers("my message")
```



Notes

- The button is only enabled (i.e. 'clickable') when all assertions are fulfilled and all required input elements are filled out.
- The wait-command is subject specific. When a subject clicks on the wait button, the experiment continues for this subject even if the other subjects have not finished yet.

waitForPlayers([message],[messageafterclick])

The waitForPlayers-command is similar to the wait-Command. The difference is that the experiment only continues after this command when all subjects of the a subject's group have pressed the button. This command can be used to synchronise groups of subjects before experiment parts which require each subject to have reached a certain point in the program.

Parameters See wait.

waitTime(time)

The waitTime-command halts the execution of the program for the specified time.

Parameters

time The time the program waits in milliseconds (!).

Notes

- No button or message is displayed for the subjects which would indicate that the experiment is waiting for a certain time. You might want to point this out in the instructions in order to avoid confusion.

waitForExperimenter()

Sometimes the experimenter needs the experiment to wait until he has done something, for example until she has explained the next stage to the subjects. The `waitForExperimenter`-command halts the execution of the experiment for all subjects until the experimenter presses the 'Ready'-button on the experimenter's client.

Notes

- No button or message is displayed for the subjects which would indicate that the experiment is waiting for the experimenter. You might want to point this out in the instructions in order to avoid confusion.

4.A.5 User Input

inputString(variablename)

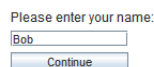
Displays a text-field in which the subject can enter a value. This value is stored in a variable with the specified name. The user can type in all characters, including numbers and foreign characters.

Parameters

`variablename` The name of the variable in which the result is stored.

Example

```
1 display("Please enter your name:")
2 inputString(name)
3 wait()
4
5 display("Hello "+name)
6 wait()
```



Notes

- The default text is an empty string. If you want to specify a default, assign a value to the variable before the `inputString`-command.
- If you want this command to be non-compulsive, which means that the subject can continue without entering something, you can use the `inputStringNC`-command. The syntax is equivalent.

inputNumber(variablename)

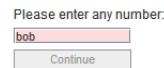
Displays a text-field in which the subject can enter a numeric value. This value is stored in a variable with the specified name. The BoXS enforces the entered text to be a natural or real number.

Parameters

variablename The name of the variable in which the result is stored.

Example

```
1 display("Please enter any number:")
2 inputNumber(num)
3 wait()
4
5 display("You entered "+num)
6 display("The square of "+num+" is "+(num*num))
7 wait()
```



Notes

- The default text is an empty string. If you want to specify a default, assign a value to the variable before the inputNumber-command.
- If the text entered is not a number the text-field is highlighted and the subject cannot continue until a correct value is entered.
- Both real and integer numbers can be entered.
- If you want this command to be non-compulsive, which means that the subject can continue without entering something, you can use the inputNumberNC-command. The syntax is equivalent.

choice(varname,values)

The choice-command displays a group of radio buttons. The user can only select one option at a time. When the user selects a value, it is stored in a variable with the specified name.

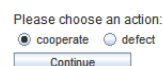
Parameters

variablename The name of the variable in which the result is stored.

values A comma-separated list of strings or numbers.

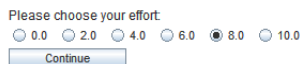
Example

```
1 display("Please choose an action:")
2 choice(action,"cooperate","defect")
3 wait()
4
5 display("You selected "+action)
6 wait()
```



Example

```
1 display("Please choose your effort:")
2 choice(effort,0,2,4,6,8,10)
3 wait()
4
5 payoff = 5 + effort - effort * effort / 10
6 display("Your effort was "+effort)
7 display("Your payoff is "+payoff+" Euros")
8 wait()
```



Notes

- By default no option is selected. If you want to specify a default, assign a value to the variable before the choice-command.
- In some cases it is nice to have the experiment system automatically randomize the order of the choice options. The choiceRandomize(...) function fulfils this role and allows for randomization without additional programming.
- If you want this command to be non-compulsive, which means that the subject can continue without entering something, you can use the choiceNC-command or the choiceRandomizeNC-command. The syntax is equivalent.

checkbox(varname,description)

The check-command displays a single checkbox along with the specified description. The user can select the checkbox or leave it unchecked. When the user selects the checkbox, the value 1 is stored in a variable with the specified name.

Parameters

variablename The name of the variable in which the result is stored.

description The description shown alongside the checkbox.

Notes

- By default no option is selected. If you want to specify a default, assign a value to the variable before the choice-command.
- A checkbox is always non-compulsive.

assert(expression)

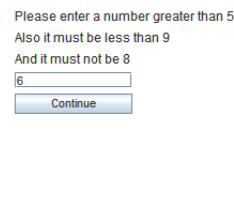
The assert-command restricts the subject's possibilities when she is faced with input fields. Using the assert-command any amount of assertions on the variables can be added.

Parameters

expression The assertion. The assertion can reference other variables.

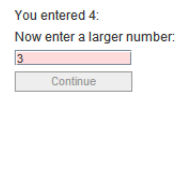
Example

```
1  display("Please enter a number greater than 5")
2  assert(val>5)
3
4  display("Also it must be less than 9")
5  assert(val<9)
6
7  display("And it must not be 8")
8  assert(val!=8)
9
10 inputNumber(val)
11 wait()
```



Example

```
1  display("Please enter any number:")
2  inputNumber(a)
3  wait()
4
5  display("You entered "+a+":")
6  display("Now enter a larger number:")
7  inputNumber(b)
8  assert(b>a)
9  wait()
```



Notes

- If an assertion is violated by user input, the user cannot continue until she chooses a valid input.
- You can impose multiple assertions on one variable. In this case user input is only allowed if it satisfies all assertions.
- The assert-command requires a wait/waitForPlayers- or button-command to have an effect.
- The assert-command does not give specific feedback to the subject on why she cannot continue when an assertion is violated. Therefore the

assumptions should be made clear and communicated to the subject in the experiment instructions and/or the experiment program itself.

style(text)

While the default style is sufficiently attractive for most experiments, experimenters might run into situations where they need more control over how things are formatted. The style-command allows to specify a style in the cascading style sheet (CSS) format which is automatically applied to all following commands. This style can specify every format aspect ranging from font name and size up to color, transparency effects, borders etc.. It is particularly useful for the display commands but also effects buttons etc.

Parameters

text The desired style which can be specified in the CSS format.

Example

```
1  mystyle="body{ padding: 0px; font-size: 16px; font-family: 'Ubuntu','Verdana','Arial',serif }"
2  mystyle=mystyle+h1{font-size: 130%; margin-top: 0px; margin-bottom: 3px; font-weight: normal; }"
3  mystyle=mystyle+h2{font-size: 115%; margin-top: 0px; margin-bottom: 3px; font-weight: normal; }"
4  mystyle=mystyle+table{background-color: #e0e0e0; border:solid; border-width:1px; border-color: #000000; margin:5px; margin-left:10px;}
5  mystyle=mystyle+td,th{padding:5px;text-align: center;}
6  mystyle=mystyle+th{background-color: #d0d0d0; }"
7
8  style(mystyle)
9
10 display("<h2>Einleitung</h2>")
11
12 display("Herzlich Willkommen zu diesem Internetexperiment. ")
13 ...
```

Einleitung

Herzlich Willkommen zu diesem Internetexperiment.

Sie nehmen nun an einem wirtschaftswissenschaftlichen Experiment teil. Für das Starten des Experimentes sowie das Beantworten der folgenden Fragen erhalten Sie 0 Euro. Zusätzlich können Sie im Hauptteil dieses Experimentes weiteres Geld verdienen. Sie können das Geld, welches Sie in diesem Experiment verdienen, ab der nächsten Woche werktags zwischen 10 und 12 Uhr am Lehrstuhl für empirische Wirtschaftsforschung abholen:

Lehrstuhl für empirische Wirtschaftsforschung
Lennéstr. 43
53113 Bonn

Ihre Teilnehmernummer ist: **655239**

Bitte notieren Sie sich diese Adresse und ihre Teilnehmernummer bevor Sie fortfahren.

Notes

- A lot of information on how to create cascading style sheets is available on the internet.

manualLayout()

By default all components are arranged vertically by the BoXS (automatic layout). When this is not sufficient, the manualLayout-command can be used to specify the exact position of each component. While this is sort of cumbersome, it allows for a great amount of freedom in designing the visual appearance of an experiment.

In order to do this, the first line of a screen needs to be manualLayout() in order to disable the automatic layout. Afterwards, display and every input command take 4 additional parameters which specify the horizontal and vertical position as well as the width and the height of the respective component.

Example

```

1 manualLayout()
2 display("<div width=370><h2>Risk Preferences<hr>",10,0,400,50)
3
4 display("<b>Fixed Pay",20,50,70,30)
5 display("<b>Lottery",120,50,100,30)
6 display("<b>Your Choice",240,50,100,30)
7
8 display("200",20,80,100,30)
9 display("0 with 50%<br>500 with 50%",120,80,100,30)
10 choice(c1,"fixed","lottery",240,80,150,30)
11
12 display("250",20,120,100,30)
13 display("0 with 50%<br>500 with 50%",120,120,100,30)
14 choice(c2,"fixed","lottery",240,120,150,30)
15
16 display("300",20,160,100,30)
17 display("0 with 50%<br>500 with 50%",120,160,100,30)
18 choice(c3,"fixed","lottery",240,160,150,30)
19
20
21 wait("Continue","Please wait",300,220,100,20)

```

Risk Preferences

Fixed Pay	Lottery	Your Choice
200	0 with 50% 500 with 50%	<input type="radio"/> fixed <input type="radio"/> lottery
250	0 with 50% 500 with 50%	<input type="radio"/> fixed <input type="radio"/> lottery
300	0 with 50% 500 with 50%	<input type="radio"/> fixed <input type="radio"/> lottery

Continue

Notes

- The four additional parameters must be supplied. Otherwise the component in question is not displayed.

- The origin of the coordinate system (0,0) is the top left edge of the client applet.
- The `manualLayout` command is only effective for one screen. If the next screen should follow a manual layout as well the command needs to be repeated on that screen. Otherwise the BoXS defaults to automatic layout.

Non-compulsory Input

In some cases it is required to have input components which are non-compulsory, i.e. the subject should be allowed to proceed even if she did not fill out a component. An example for this would be a text field for an email address or a comment, which is optional. In order to allow for this, the BoXS includes non-compulsory versions of all input commands. These non-compulsory commands have the same syntax as the usual commands and end with the letters 'NC':

- `inputStringNC(...)`
- `inputNumberNC(...)`
- `choiceNC(...)`
- `choiceRandomizeNC(...)`

Default values

It is possible to specify default values for all input components. In order to do so this one can simply assign a value to the variable in question before the corresponding input component is created. For example:

```
name="Bob"
inputString(name)
wait()
```

4.A.6 Matching

Matching is the process by which the subjects are assigned to groups and roles. You can use automatic matching, including (perfect) stranger matching, as well as manual matching.

Notes

- **Implicit matching:** You do not need to specify matching information. If you specify no matching information, every subject is automatically allocated to a separate group and receives the role "A", which is sufficient for experiments which require no interaction among the subjects.

matchAll(roles)

The matchAll-command distributes the subjects in alphabetical order of their username to the desired groups/roles.

Parameters

roles A comma-separated list of all roles.

Example

```
1 matchAll("investor", "trustee")
2 display("Your name is <b>"+username+"</b>")
3 display("You are the <b>"+role+"</b> in group <b>"+group+"<")
4 display("Your opponent is <b>"+opponent.role+"</b>")
5 wait()
```

Your name is S1
You are the "investor" in group 1
Your opponent is "trustee"

Notes

- The role names can be letters (A,B,C) or arbitrary strings (investor, trustee).

matchPerfectStranger(roles)

The matchPerfectStranger-command distributes the subjects to the desired groups/roles in a way that ensures that no subject are matched to the same subject again.

Parameters

roles A comma-separated list of all roles.

Example

```
1 matchPerfectStranger(A,B,C,D)
2 // Experiment Part 1
3
4 matchPerfectStranger(A,B,C,D)
5 // Experiment Part 2
6
7 matchPerfectStranger(A,B,C,D)
8 // Experiment Part 3
9
10 matchPerfectStranger(A,B,C,D)
11 // Experiment Part 4
```


Notes

- Perfect stranger matching is an extremely computationally expensive operation. Therefore the system uses pre-calculated tables. Use the following table to figure out how many matches you can achieve given subject- and rolecount.

		Number of Roles														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Number of Subjects	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	4	-	3	-	-	-	-	-	-	-	-	-	-	-	-	-
	5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	6	-	5	1	-	-	-	-	-	-	-	-	-	-	-	-
	7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	8	-	7	1	-	-	-	-	-	-	-	-	-	-	-	-
	9	-	-	4	-	-	-	-	-	-	-	-	-	-	-	-
	10	-	9	-	-	1	-	-	-	-	-	-	-	-	-	-
	11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	12	-	11	3	1	-	1	-	-	-	-	-	-	-	-	-
	13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	14	-	-	13	-	-	-	1	-	-	-	-	-	-	-	-
	15	-	-	4	-	1	-	-	-	-	-	-	-	-	-	-
	16	-	-	15	-	5	-	-	1	-	-	-	-	-	-	-
	17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	18	-	-	17	5	-	-	1	-	1	-	-	-	-	-	-
	19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	20	-	-	19	-	4	1	-	-	-	1	-	-	-	-	-
	21	-	-	7	-	-	-	1	-	-	-	-	-	-	-	-
	22	-	-	21	-	-	-	-	-	-	-	1	-	-	-	-
	23	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	24	-	-	?	7	5	-	1	-	1	-	-	-	1	-	-
	25	-	-	-	-	3	-	-	-	-	-	-	-	-	-	-
	26	-	-	?	-	-	-	-	-	-	-	-	-	-	1	-
	27	-	-	-	8	-	-	-	-	1	-	-	-	-	-	-
	28	-	-	?	-	5	-	-	1	-	-	-	-	-	-	1
	29	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	30	-	-	?	-	-	-	-	-	-	-	-	-	-	-	-

- You can only do perfect stranger matching until all possible matches are use. If you try doing more matches, an error is thrown.
- Use the command `matchHistoryClear` to reset the perfect stranger matching algorithm.
- The role names can be letters (A,B,C) or arbitrary strings (investor, trustee).

`matchStranger(roles)`

The `matchStranger`-command distributes the subjects randomly to the desired groups/roles.

Parameters

`roles` A comma-separated list of all roles.

Notes

- The role names can be letters (A,B,C) or arbitrary strings (investor, trustee).

matchManual(username,group,role)

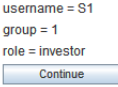
By using the matchManual-command you can manually specify how the subjects are to be matched. Each command matches exactly one subject. The first argument is the username of the subject which shall be assigned, the second and third argument are group and role.

Parameters

username	The username of the subject which shall be matched.
group	The desired group.
role	The desired role.

Example

```
1 matchManual(S1,1,investor)
2 matchManual(S2,1,trustee)
3
4 display("username = "+username)
5 display("group = "+group)
6 display("role = "+role)
7
8 wait()
9
```



Notes

- The role names can be letters (A,B,C) or arbitrary strings (investor, trustee).
- Important: In the most recent version of the BoXS username, group and role can be variables which vastly expands the possibilities of the matchManual-command. This implies that the corresponding values *must be contained in quotation marks*. For example:

```
matchManual("S1", "1", "A")
```

matchDone()

If you want to change the matching at some point in the experiment, use the `matchDone()`-command. The experiment halts until all subjects have reached this point.

4.B Example BoXS Programs

4.B.1 Questionnaire

This example demonstrates how to implement a simple questionnaire.

```
1 display("<h1>Questionnaire</h1><br>")
2
3 display("<br>Please enter your name:")
4 inputString(name)
5
6 display("<br>Please enter your field of study:")
7 inputString(study)
8
9 display("<br>Please enter your age:")
10 inputNumber(age)
11 assert(age>10 && age<100 && age==round(age))
12
13 display("<br>Please select your gender:")
14 choice(gender,"female","male")
15
16 wait()
```

4.B.2 Public Good Game

The public good game, as implemented in this example, is a game in which 4 players can contribute a part of their initial endowment to a group project which benefits everyone. Has a special feature it uses the Google Chart API in order to graphically display the distribution of the contributions.

```
1 // Public Good Game for 4 players
2 matchAll(A,B,C,D)
3
4 // Introductory text
5 display("<h1>Public Good Game</h1><br>")
6 display("For this experiment you are endowed with <b>10 Euros</b>.")
7 display("You are in a group with 3 other people.")
8 display("You can contribute any amount of your endowment to a group project.")
9 display("Your payoff is increased by <b>0.4 times</b> the total contributed to the project.")
10 display("<br><img src='http://chart.apis.google.com/chart?cht=tx&chl=p_i=10%20-%20z_i%20*2%20%200.4%20%20\cdot%20\sum_i\{z_i\}'>")
11
12 // Read how much the subjects want to contribute and store it in z
13 display("<br>Please choose the amount you would like to contribute:")
14 inputNumber(z)
15
16 assert(z>=0)
17 assert(z<=10)
18 assert(z==round(z))
19
20 // Wait until all players entered their respective contribution
21 waitForPlayers()
22
23 // Inform the subjects about the results of the experiment
24 display("You are player "+role+". You contributed <b>"+z+"</b> Euros</b>. <br>The total amount donated is <b>"+sum(z)+"</b> Euros</b>.")
25
26 display("<br><img src='http://chart.apis.google.com/chart?cht=p&chd=t:"+A.z+", "+B.z+", "+C.z+", "+D.z+"&chs=200x150&chl=A|B|C|D'>")
27
28 // Calculate the payoff for each subject
29 payoff = 10 - z + 0.4*sum(z)
30
31 // Inform the subjects about their payoff
32 display("Your payoff is <b>"+payoff+"</b> Euros</b>")
33
34 wait()
```

4.B.3 Chat Client

The chat client implemented in this example allows two subjects to send messages to each other.

```
1 // Colorful chat for two players, A and B
2 matchAll(A,B)
3 A.color="#33bb33"
4 B.color="#5555bb"
5
6 *.chat=""
7 quit=0
8 // Repeat until the user presses the leave-button
9 while(quit==0)
10 {
11     // Display previous chat and a textbox where the user can type
12     display("<h1>Chat window</h1>")
13     display(chat)
14     display("<br><h2>Enter your message:</h2>")
15     inputString(message)
16
17     // Buttons to send the typed message or leave the chat
18     send=0
19     button(send,"Send message")
20     button(quit,"Leave chat")
21
22     // Refresh every second
23     waitTime(1000)
24
25     // If the user pressed the send-button, append his text to the chat
26     if(send)
27     {
28         *.chat=chat+role+": <font color="+color+">"+message+"</font><br>"
29     }
30 }
```

4.B.4 Dutch Auction

In a Dutch auction two subjects watch the price of a good decrease over time and can buy it at the current price by clicking on the corresponding button.

```
1 matchAll(A,B)
2
3 // Welcome screen
4 display("<h1>Dutch auction</h1>")
5 display("Click when you are ready:")
6 waitForPlayers("Ready!")
7
8 // Initialize variables
9 *.winner=0
10 price=30
11
12
13 // Repeat this until the price is 0 or a winner was found
14 while(price > 0 && winner==0)
15 {
16 // Display the current price and offer a button to bid
17 clicked=0
18 display("Current price: "+price)
19 button(clicked,"bid")
20
21 // Wait one second
22 waitTime(1000)
23
24 // If the player clicked, he/she will be the winner
25 if (clicked)
26 {
27 *.winner=role
28 *.winnerprice=price
29 }
30 price=price-1
31 }
32
33 // Display outcome
34 display("Auction finished.")
35
36 if (winner==0)
37 {
38 display("No one bought the good.")
39 }
40 if (winner!=0)
41 {
42 display("Player "+winner+" bought the good for "+winnerprice+".")
43 }
44
45 waitForPlayers()
```

4.B.5 Localization

Sometimes an experiment has to work in different languages at the same time. This example shows an easy way to implement such a feature.

```
1 // Language selection screen
2 display("Please select your language:")
3 choice(language, "Deutsch", "English")
4 wait()
5
6 // Begin of actual experiment
7 welcomemessage["Deutsch"] = "Willkommen zu unserem Experiment"
8 welcomemessage["English"] = "Welcome to our Experiment"
9 display(welcomemessage[language])
10
11 question01["Deutsch"] = "Bitte geben Sie Ihr Alter ein:"
12 question01["English"] = "Please enter your age:"
13 display(question01[language])
14
15 inputNumber(age)
16 assert(age>10 && age<100 && age==round(age))
17
18 continue["Deutsch"] = "Weiter..."
19 continue["English"] = "Continue..."
20 continue2["Deutsch"] = "Bitte warten sie bis das Experiment weiter geht"
21 continue2["English"] = "Please wait for the experiment to continue"
22 wait(continue[language], continue2[language])
```

4.B.6 Real Effort Task

In this tedious real effort task subjects have to count the number of ones in a table of digits.

```
1 correct=0
2 total=0
3
4 for (repetition=0; repetition<5; repetition=repetition+1)
5 {
6 // generate 01-table
7 table=""
8 count=0
9
10 for (row=0; row<8; row=row+1)
11 {
12 for (col=0; col<20; col=col+1)
13 {
14 c=randomUniformInteger(0,1)
15 count=count+c
16 table=table+c+ " "
17 }
18 table=table+"<br>"
19 }
20
21 // Display table and choices
22 display("Correct: <b>"+correct+"</b> Wrong: <b>"+(total-correct))
23 display("<br><pre>" +table)
24 pad=randomUniformInteger(-2,2)
25 display("How many ones are there?")
26 choice(userc, count+pad-2, count+pad-1, count+pad, count+pad+1, count+pad+2)
27 wait()
28
29 // Was user choice right?
30 total=total+1
31 if (userc==count)
32 {
33 correct=correct+1
34 }
35 }
36 }
```


Bibliography

- Abbink, K. and Sadrieh, A. (1995). Ratimage - research assistance toolbox for computer-aided human behavior experiments. *SFB 303 Discussion Paper B-325, University of Bonn*.
- Ainslie, G. and Haslam, N. (1992). Hyperbolic discounting. *Choice over time*, pages 57–92.
- Alesina, A. and Giuliano, P. (2009). Preferences for Redistribution. *NBER Working Paper*.
- Alesina, A., Gläser, E., Sacerdote, B., and Center, L. (2001). Why doesn't the US have a European-style welfare system? *NBER working paper*.
- Amabile, T. (1993). Motivational synergy: Toward new conceptualizations of intrinsic and extrinsic motivation in the workplace. *Human Resource Management Review*, 3(3):185–201.
- Andreoni, J. (1990). Impure altruism and donations to public goods: a theory of warm-glow giving. *The Economic Journal*, 100(401):464–477.
- Andrisani, P. and Nestel, G. (1976). Internal-external control as contributor to and outcome of work experience. *Journal of Applied Psychology*, 61(2):156–165.
- Bandiera, O., Barankay, I., and Rasul, I. (2005). Social Preferences and the

- Response to Incentives: Evidence from Personnel Data*. *The Quarterly Journal of Economics*, 120(3):917–962.
- Benabou, R. and Tirole, J. (2003). Intrinsic and extrinsic motivation. *Review of Economic Studies*, 70(3):489–520.
- Benabou, R. and Tirole, J. (2006). Belief in a Just World and Redistributive Politics*. *Quarterly Journal of Economics*, 121(2):699–746.
- Costa Jr, P. and McCrae, R. (1992). Revised neo personality inventory (neo pi-r) and neo five-factor inventory (neo-ffi). *Psychological Assessment Resources, Odessa, Fla. (PO Box 998, Odessa 33556)*.
- Deci, E. (1971). Effects of externally mediated rewards on intrinsic motivation. *Journal of personality and Social Psychology*, 18(1):105–115.
- Deci, E., Köstner, R., and Ryan, R. (1999). A meta-analytic review of experiments examining the effects of extrinsic rewards on intrinsic motivation. *Psychological bulletin*, 125(6):627.
- Doane, D. and Tracy, R. (2000). Using beam and fulcrum displays to explore data. *The American Statistician*, 54(4):289–290.
- Dohmen, T. and Falk, A. (2006). Performance pay and multi-dimensional sorting: Productivity, preferences and gender. *Institute for the Study of Labor (IZA); University of Bonn-Economic Science Area; CESifo (Center for Economic Studies and Ifo Institute for Economic Research)*.
- Dohmen, T., Falk, A., Huffman, D., Schupp, J., Sunde, U., and Wagner, G. (2005). *Individual risk attitudes: New evidence from a large, representative, experimentally-validated survey*. IZA.
- Dur, R. and Sol, J. (2010). Social interaction, co-worker altruism, and incentives. *Games and Economic Behavior*, 69(2):293–301.

- Eckel, C. and Wilson, R. (2003). The human face of game theory. *Trust and Reciprocity: Interdisciplinary Lessons from Experimental Research*. Russell Sage Foundation Series on Trust. Russell Sage Foundation, New York, pages 245–274.
- Ellingsen, T. and Johannesson, M. (2008). Anticipated verbal feedback induces altruistic behavior. *Evolution and Human Behavior*, 29(2):100–105.
- Falk, A. (2003). Homo oeconomicus versus homo reciprocans: Ansätze für ein neues wirtschaftspolitisches leitbild? *Perspektiven der Wirtschaftspolitik*, 4(1):141–172.
- Falk, A. and Fehr, E. (2003). Why labour market experiments? *Labour Economics*, 10(4):399–406.
- Falk, A. and Fischbacher, U. (2006). A theory of reciprocity. *Games and Economic Behavior*, 54(2):293–315.
- Falk, A. and Heckman, J. (2009). Lab experiments are a major source of knowledge in the social sciences. *Science*, 326(5952):535.
- Falk, A. and Ichino, A. (2006). Clean evidence on peer pressure. *Journal of Labor Economics*, 24:39–57.
- Falk, A. and Kosfeld, M. (2006). The hidden costs of control. *The American economic review*, 96(5):1611–1630.
- Fehr, E., Fischbacher, U., and Gächter, S. (2002). Strong reciprocity, human cooperation, and the enforcement of social norms. *Human nature*, 13(1):1–25.
- Fehr, E. and Gächter, S. (2000). Fairness and retaliation: The economics of reciprocity. *The Journal of Economic Perspectives*, 14(3):159–181.
- Finney, D. (1947). Probit analysis; a statistical treatment of the sigmoid response curve.

- Fischbacher, U. (1999). z-tree. toolbox for readymade economic experiments. *IEW Working Paper*, 21.
- Fischbacher, U. (2007). z-tree: Zurich toolbox for ready-made economic experiments. *Experimental Economics*, 10:171–178.
- Fong, C. (2001). Social preferences, self-interest, and the demand for redistribution. *Journal of Public Economics*, 82(2):225–246.
- Frey, B. and Jegen, R. (2001). Motivation crowding theory. *Journal of Economic surveys*, 15(5):589–611.
- Frey, B. and Neckermann, S. (2006). Auszeichnungen: ein vernachlässigter Anreiz. *Perspektiven der Wirtschaftspolitik*, 7(2):271–284.
- Frey, B. and Oberholzer-Gee, F. (1997). The cost of price incentives: An empirical analysis of motivation crowding-out. *The American economic review*, 87(4):746–755.
- Greiner, B. (2004). An online recruitment system for economic experiments.
- Grossman, S. and Hart, O. (1983). An analysis of the principal-agent problem. *Econometrica: Journal of the Econometric Society*, pages 7–45.
- Guttman, L. (1950). The basis for scalogram analysis. *Measurement and prediction*, 4:60–90.
- Gächter, S. and Fehr, E. (1999). Collective action as a social exchange. *Journal of Economic Behavior & Organization*, 39(4):341–369.
- Güth, W., Schmittberger, R., and Schwarze, B. (1982). An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior & Organization*, 3(4):367–388.

- Hausman, J. and McFadden, D. (1984). Specification tests for the multinomial logit model. *Econometrica: Journal of the Econometric Society*, pages 1219–1240.
- Hausman, J. and Wise, D. (1978). A conditional probit model for qualitative choice: Discrete decisions recognizing interdependence and heterogeneous preferences. *Econometrica: Journal of the Econometric Society*, pages 403–426.
- Huck, S., Kübler, D., and Weibull, J. (2003). Social norms and economic incentives in firms.
- Kandel, E. and Lazear, E. (1992). Peer pressure and partnerships. *Journal of Political Economy*, 100(4):801–817.
- Kirchkamp, O. (2004). Www experiments for economists, a technical introduction.
- Kluger, A. and DeNisi, A. (1996). Effects of feedback intervention on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological bulletin*, 119(2):254.
- Krupka, E. and Croson, R. (2011). What Gives: Spurring generosity with targeted cues.
- Krupka, E. and Weber, R. (2009). The focusing and informational effects of norms on pro-social behavior. *Journal of Economic Psychology*, 30(3):307–320.
- Lazear, E. (1989). Pay equality and industrial politics. *Journal of Political Economy*, 97(3):561–580.
- Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology*.

- McDonald, J. and Moffitt, R. (1980). The uses of tobit analysis. *The review of economics and statistics*, 62(2):318–321.
- Nalbantian, H. and Schotter, A. (1997). Productivity under group incentives: An experimental study. *The American Economic Review*, pages 314–341.
- Nolte, H., Weischer, C., Wilkesmann, U., Maetzel, J., and Tegethoff, H. (1997). Kontrolleinstellungen zum Leben und zur Zukunft – Auswertung eines neuen, sozialpsychologischen Itemblocks im sozio-oekonomischen Panel. *Diskussionspapier der Fakultät für Sozialwissenschaft der Ruhr-Universität Bochum*.
- Pappi, F. and Thurner, P. (2002). Electoral behaviour in a two-vote system: Incentives for ticket splitting in german bundestag elections. *European Journal of Political Research*, 41(2):207–232.
- Rabin, M. and Weizsäcker, G. (2009). Narrow bracketing and dominated choices. *The American economic review*, 99(4):1508–1543.
- Rainer, H. and Siedler, T. (2008). Subjective income and employment expectations and preferences for redistribution. *Economics Letters*, 99(3):449–453.
- Read, D., Loewenstein, G., and Rabin, M. (1999). Choice bracketing. *Journal of Risk and uncertainty*, 19(1):171–197.
- Rubinstein, A. (1998). *Modeling bounded rationality*. The MIT Press.
- Samuelson, P. (1977). Reaffirming the existence of "reasonable" bergson-samuelson social welfare functions. *Economica*, pages 81–88.
- Schmitt, H. (2000). Zur vergleichenden Analyse des Einflusses gesellschaftlicher Faktoren auf das Wahlverhalten: Forschungsfragen, Analysestrategien und einige Ergebnisse. *Wahlen und Wähler. Analysen aus Anlaß der Bundestagswahl 1998*.

- Siedler, T. (2009). Schooling and Citizenship: Evidence from Compulsory Schooling Reforms. *IZA Discussion Paper*.
- Wagner, G., Frick, J., and Schupp, J. (2007). The german socio-economic panel study (soep): Scope, evolution and enhancements. *SOEPpapers*.
- Wooldridge, J. (2002). *Econometric analysis of cross section and panel data*. The MIT press.
- Xiao, E. and Houser, D. (2007). Emotion expression and fairness in economic exchange. *University of Pennsylvania*.
- Zeiliger, R. (2009). Regate 9.22 - internet based software for experimental economics.