PADERBORN UNIVERSITY
The University for the Information Society

Non- and Semiparametric Methods for Surface Estimation of Functional Time Series on a Lattice

Der Fakultät für Wirtschaftswissenschaften der Universität Paderborn zur Erlangung des
akademischen Grades Doktor der Wirtschaftswissenschaften - Doctor rerum politicarum -
vorgelegte Dissertation von

Bastian Schäfer M.Sc.

geboren am 8. Januar 1990 in Datteln

November 2021

Abstract

We consider nonparametric and semiparametric estimation of the mean or expectation surface of a functional time series on a lattice.

For (univariate) local regression, we propose a boundary modification procedure that allows smooth continuation of the estimates from the interior to the boundary. The proposed method for boundary modification is suitable for generating boundary kernel functions. Both the boundary modification and the kernel functions generated from it are important to the considered estimation of the expectation surface.

The nonparametric estimation of the expectation surface of a functional time series implies long computation times under the classical bivariate regression methods such as kernel regression or local regression. The double conditional smoothing (DCS) transforms a bivariate regression into two univariate regressions, and shortens the running time of the algorithm, especially for large data sets. We present a functional DCS for both kernel and local regression, under the assumption of both independent and dependent error terms. It is shown that, under certain assumptions, the DCS estimator is equivalent to bivariate regression. Integral formulas for the asymptotic expectation and the asymptotic variance of the DCS estimator are derived. This yields the asymptotically optimal bandwidths for estimating the regression surface or partial derivatives of it. These bandwidths are selected in a data-driven procedure by an iterative plug-in algorithm, which quickly converges to the optimal bandwidths. The developed algorithms and functions are implemented in the R package DCSmooth.

Zusammenfassung

Wir betrachten die nicht- und semiparametrische Schätzung der Erwartungsfläche einer funktionalen Zeitreihe.

Für die (univariate) lokale Regression wird ein Verfahren zur Randmodifikation vorgestellt, welches eine glatte Fortsetzung des Schätzers zu den Rändern hin ermöglicht. Die vorgeschlagene Methode zur Randmodifikation ist geeignet, um Randkernfunktionen zu erzeugen. Sowohl die Randmodifikation als auch die daraus erzeugten Kernfunktionen spielen in der Schätzung der Erwartungsfläche eine wichtige Rolle.

Die nichtparametrische Schätzung der Erwartungsfläche einer funktionalen Zeitreihe impliziert lange Rechenzeiten unter den klassischen bivariaten Regressionsmethoden wie der Kernregression oder der Lokalen Regression. Die doppelt bedingte Glättung (Double Conditional Smoothing, DCS) überführt eine bivariate Regression in zwei univariate Regressionen, und verkürzt die Laufzeit des Algorithmus insbesondere für große Datensätze. Wir stellen ein funktionales DCS für Kern- und lokale Regression vor, sowohl unter der Annahme von unabhängigen als auch von abhängigen Störtermen. Es wird gezeigt, dass, unter bestimmten Annahmen, der DCS-Schätzer äquivalent zur bivariaten Regression ist. Integralformeln für den asymptotischen Erwartungswert und die asymptotische Varianz des DCS-Schätzers werden hergeleitet. Daraus ergeben sich die asymptotisch optimalen Bandbreiten zur Schätzung der Regressionsfläche oder partieller Ableitungen dieser. Die datengestützte Wahl dieser Bandbreiten erfolgt mit einem iterativer Plug-In Algorithmus, welcher schnell die gegen die optimalen Bandbreiten konvergiert. Die entwickelten Algorithmen und Funktionen sind im R-Paket *DCSmooth* implementiert.

Preface

The thesis at hand was written during my time as research assistant to Prof. Dr. Yuanhua Feng at the professorship of Econometrics and Quantitative Methods of Empirical Economic Research, Paderborn University. This thesis considers the representation of functional time series on a lattice, the nonparametric regression of the mean surface of such a time series, and the related selection of optimal bandwidths. The explicit consideration of such lattice time series is not very common and suitable methods for estimating large data sets were rarely available in this or a comparable framework. This thesis aims at closing this gap further by providing suitable theoretical and practical methods for non- or semiparametric estimation of lattice data. The main application in this thesis is the lattice representation of financial time series. However, the proposed methods are not limited to this type of data, but can also be used for a range of other time series or in spatial smoothing applications.

In the course of the work on the topics of this thesis, the R-package DCSmooth was developed and is published on CRAN. This package implements the proposed methods and tools for application in R and provides a convenient way for nonparametric estimation of the mean surface of spatial time series on a lattice.

My sincere gratitude goes first to Prof. Dr. Yuanhua Feng for supervising my thesis and for all his extensive guidance and suggestions I have received over the years. In particular, I am thankful for the productive discussions and helpful comments on the joint work with him resulting in two cooperative papers which are part of this thesis. In addition to this professional aspect I always appreciated the cordial environment at the chair.

Furthermore, I would like to express my thanks to Prof. Dr. Yuanhua Feng, Prof. Dr. Hendrik Schmitz, Prof. Dr. Oliver Müller and Dr. Christian Peitz for agreeing to be members of my doctoral committee.

Contents

List of Figures **VIII**

List of Tables **IX**

List of Abbreviations and Acronyms **X**

1 Introduction **1**
- 1.1 Spatial Time Series and Nonparametric Regression 1
- 1.2 Contribution of this Thesis 4
- 1.3 Summary of the Contents 8

2 Boundary Modification in Local Regression **10**
- 2.1 Introduction 10
- 2.2 Modified Local Polynomial Regression 12
- 2.3 Equivalency of the Proposed Weighting Methods 14
- 2.4 Boundary Behavior of Local Regression 20
- 2.5 Areas of Application 24
- 2.6 Final Remarks 25
- 2.7 Appendix 26

3 Fast Computation and Bandwidth Selection Algorithms for the DCS **33**
- 3.1 Introduction 33
- 3.2 The Model and the Basic DCS Procedure 35
  - 3.2.1 A Semiparametric Functional Time Series Model 35
  - 3.2.2 The Double Conditional Smoothing 36
- 3.3 The Improved Double Conditional Smoothing 40
  - 3.3.1 Boundary Correction Under the DCS 40
  - 3.3.2 A Functional Smoothing Scheme 41
  - 3.3.3 Estimation of Derivatives 42
  - 3.3.4 Asymptotic Behavior of the Estimator 43
- 3.4 Bandwidth Selection 44
  - 3.4.1 Asymptotic Optimal Bandwidths 44
  - 3.4.2 The IPI Algorithm 45
- 3.5 Finite Sample Simulations 48
- 3.6 Application to Financial Data 51
- 3.7 Final Remarks 54
- 3.8 Appendix 55
  - 3.8.1 Proof of Theorem 3.1 55

3.8.2 Expectation and Variance of the DCS Estimator 56
3.8.3 Asymptotic Normality of the DCS Estimator 59
3.8.4 Derivation of the Optimal Bandwidths 59

4 Local Polynomial DCS under Dependent Errors 62
4.1 Introduction 62
4.2 The FDCS for Local Polynomial Estimators 65
4.2.1 Model and Assumptions 65
4.2.2 Extension to Local Polynomial Smoothers 66
4.2.3 Boundary Modification in the LP-DCS 69
4.2.4 Equivalent Kernels 70
4.3 Bandwidth Selection for the LP-DCS 71
4.3.1 Asymptotic Bias and Variance 71
4.3.2 Asymptotic Optimal Bandwidths 73
4.3.3 Bandwidth Selection by the IPI 74
4.4 Spatial Error Structure 76
4.4.1 Definition of the Spatial ARMA 76
4.4.2 Estimation of the SARMA 77
4.5 Simulation Study 79
4.6 Applications 81
4.7 Final Remarks 85
4.8 Appendix 89
4.8.1 Proof of Theorem 4.1 89
4.8.2 Proof of Theorem 4.2 89
4.8.3 Expectation and Variance 91
4.8.4 Optimal Bandwidths 94

5 Further Research Topics 97

6 Conclusion 103

A Appendix: *DCSmooth* Vignette 106
A.1 Introduction 106
A.2 Details of Functions, Methods and Data 108
A.2.1 Functions 108
A.2.2 Methods 116
A.2.3 Data 116
A.3 Application 117
A.3.1 Defining the Options 117
A.3.2 Application of the DCS with iid. Errors 119
A.3.3 Application of the DCS with SARMA Errors 121
A.3.4 Modeling Errors with Long Memory 125

A.3.5 Estimation of Derivatives 126
A.4 Mathematical Background 127
A.4.1 Double Conditional Smoothing 127
A.4.2 Bandwidth Selection 128
A.4.3 Boundary Modification 128
A.4.4 Spatial ARMA Processes 129
A.4.5 Estimation of SARMA Processes 130

Bibliography **XI**

List of Figures

1.1 Boundary Region in Spatial Regression. 5
1.2 Real-Time Monitoring of Wind Speed at Fairbanks, AK. 6
1.3 Run-Times for Bivariate and DCS Regression 7

2.1 Boundary Weighting Functions 14
2.2 Boundary Kernels 23

3.1 Kernel Regression DCS: Bandwidths of Simulation Study. 49
3.2 Kernel Regression DCS: Surfaces of Simulated Functions. 50
3.3 Kernel Regression DCS: Surfaces of SIE and BMW Data. 53

4.1 Local Polynomial DCS: Bandwidths of Simulation Study. 82
4.2 Local Polynomial DCS: Variance Coefficients of Simulation Study. 83
4.3 Local Polynomial DCS: Surfaces of Simulated Functions. 84
4.4 Local Polynomial DCS: Surfaces of ALV Data. 86
4.5 Local Polynomial DCS: Surfaces of SIE Data. 87

5.1 Surfaces of Temperature and Wind Speed in Yuma, AZ. 98
5.2 Spectral Densities for an SARMA process. 102

List of Tables

2.1 μ-smooth Boundary Kernels 32
2.2 (μ, μ')-smooth Boundary Kernels 32

3.1 Kernel Regression DCS: Bandwidth Statistics of Simulation Study 50
3.2 Kernel Regression DCS: Bandwidths for SIE and BMW 52

4.1 Local Polynomial DCS: Bandwidth Statistics of Simulation Study. 81
4.2 Local Polynomial DCS: Variance Coefficients of Simulation Study. 81
4.3 Bandwidth Selection for ALV and SIE under dependent errors 88
4.4 Estimated SARMA models for ALV. 95
4.5 Estimated SARMA models for SIE. 96

5.1 Estimated SARMA models for temperature and wind speed in Yuma. . . 101
5.2 Bandwidths Under a Nonparametric Error Assumption. 101
5.3 Bandwidths Under a Long-Memory Error Assumption. 102

List of Abbreviations and Acronyms

ALV	Allianz SE.
AMISE	Asymptiotic Mean Integrated Squared Error.
AMSE	Asymptotic Mean Squared Error.
ARCH	Autoregressive Conditional Heteroscedasticity.
ARMA	Autoregressive Moving Average (process).
BIC	Bayes Information Criterion.
BMW	Bayerische Motorenwerke AG.
BR	Boundary Region.
CRAN	The Comprehensive R Archive Network.
CTS	Calendar Time Sampling.
CV	Cross-Validation.
DCS	Double Conditional Smoothing.
EIM	Exponential Inflation Method.
FDCS	Functional Double Conditional Smoothing.
FPC	Functional Principal Component (regression).
GARCH	Generalized ARCH.
GM	Gasser-Müller (estimator).
HFF	High-Frequency Financial (data).
iid.	Independently and identically distributed.
IPI	Iterative Plug-In (algorithm).
IR	Interior Region.
LP-DCS	Local Polynomial DCS.
MDCS	Matrix Double Conditional Smoothing.
MIM	Multiplicative Inflation Method.
MISE	Mean Integrated Squared Error.
ML	Maximum Likelihood (estimation).
MSE	Mean Squared Error.
NOAA	National Oceanic and Atmospheric Administration.
NW	Nadaraya-Watson (estimator).
SAR	Spatial Autoregressive (process).
SARMA	Spatial Autoregressive Moving Average (process).
SD	Standard Deviation.
SFARIMA	Spatial Fractional Autoregressive Integrated Moving Average (process).
SIE	Siemens AG.

1 Introduction

1.1 Spatial Time Series and Nonparametric Regression

Univariate time series are a long- and well-established topic in statistics and econometrics. They serve as a main building block in many econometric analysis of time-related phenomena and have been extended to multivariate or multivariable models in parametric and nonparametric frameworks. Functional time series are a relatively new development whose incorporation into application in econometric models started not until computational power was sufficient for handling larger data sets. Functional data is, in the most general way, characterized as a set of functions observed in different states (Ferraty and Vieu, 2006). As Li et al. (2019a) point out, functional time series can in general be classified into two types: *"On the one hand, they can arise by separating an almost continuous time record into natural consecutive intervals. [...] On the other hand, functional time series can also arise when observations in a time period are considered as finite realizations of a continuous function."*

Examples for functional time series of the second type are, e.g., spectrometric curves from chemical analysis (Ferraty and Vieu, 2006), biometric observations of children's motions (Ramsay and Silverman, 2005), protein levels in the blood of patients over several years (Yao et al., 2005), AIDS incidences for age at diagnosis and year of diagnosis (Facer and Müller, 2003), and age-specific mortality and fertility rates as studied by Hyndman and Ullah (2007); Gao and Shang (2017); Shang and Hyndman (2017) where the age-specific death rate functions were obtained over several years. In general, this type of framework is characterized as set of functions that are observed in different states, representing different time points, locations or test persons in an experiment.

Estimation strategies for functional data of this type usually follow the same pattern; each individual function is estimated and then the dependency between the functions is modeled. The functions can be estimated e.g. by spline regression (Ramsay and Silverman, 2005; Hyndman and Ullah, 2007; Reiss and Ogden, 2007; Horváth and Kokoszka, 2012; Shang and Hyndman, 2017; Beyaztas and Shang, 2019) or functional partial least squares regression (Reiss and Ogden, 2007; Hyndman and Shang, 2009). Although the structure between the states might be modeled using a parametric approach (Horváth and Kokoszka (2012) use an autoregressive dependency between the function states based on the model by Bosq (2000) for example), the mainly used method is functional principal component analysis (FPC). The FPC is used e.g. by Hyndman and Ullah (2007); Hyndman and Shang (2009); Gao and Shang (2017); Shang and Hyndman (2017) to estimate and forecast the time series functions for mortality data. Among others, FPC is studied by Ramsay and Dalzell (1991); Rice and Silverman (1991); Hall et al. (2006) and Yao et al.

(2005) for sparse and irregular data, Li et al. (2019a) extend the functional time series concept to long memory data.

However, this thesis will only cover functional time series of the second type. For this type the functional time series under consideration is usually a univariate (discrete) time series or a (continuous) stochastic process which can be split up in naturally defined intervals. These intervals correspond to the states in the first definition of functional time series and might, e.g., be different days in which the time series can be split. Spatio-temporal time series can also be included in this class where time series are observed at the same time but different locations (or over other different entities). Some examples of functional data of this type are given by Horváth and Kokoszka (2012) from various scientific disciplines. They include environmental and geophysical data sets such as daily rainfall measurements in Australia (Delaigle and Hall, 2010) or pollution curves (Kaiser et al., 2002) for which the states are different places where observations are made. Specific examples of environmental functional time series are the intraday temperature and wind speed curves measured over a year in Figure 5.1 where the functional values are observed over the day and the days are the states in which the function occurs.

In financial econometrics, Andersen and Bollerslev (1997, 1998); Andersen et al. (2000) introduce a framework for return volatilities which can be represented by a functional time series. To study intraday volatility dynamics and interday volatility persistence simultaneously, Andersen and Bollerslev (1998) use a data set of five-minute return observations of the Deutsche Mark-Dollar spot exchange rate, spanning a full year from 1992 to 1993 with an intraday and interday index. This is essentially a functional time series on an equidistant lattice, with the intraday return volatility as the (possibly continuous) function and the interday indices as the (discrete) states of the functions. They identify and discuss three important effects in this functional time series of returns: daily ARCH effects, calendar effects, and macroeconomic announcement effects. A parametric multiplicative model designed for simultaneous estimation of the daily effects and the intraday calendar effects is used to isolate the purely stochastic volatility effects from calendar and announcement effects. Their findings suggest, that intraday volatility contains valuable information for interday volatility and hence, volatility measures only at the daily level lack information. Similar results are found by Andersen et al. (2000) for the Japanese Stock Market using data of the Nikkei index. Feng and McNeil (2008) used the decomposition into intraday and interday components to examine the long-term deterministic behavior under slowly changing volatility. Another approach was used by Engle and Sokalska (2012), who used a multiplicative component GARCH to forecast asset volatility by using daily, diurnal, and stochastic intraday components in their model. The explicit representation of the volatility surface is due to Feng (2013) and Peitz and Feng (2015), where the returns are assumed to follow a multiplicative respective additive model of a mean function and a volatility function.

If the observations of functional time series are cardinal variables and hence not only observations in the *function* direction, but also in the *state* direction follow a measurable order, the term *time series under a spatial representation* is used. If the observations of such a time series occur at equidistant points in all dimensions, we refer to this process as *time series on a lattice* or *lattice process*. For high-frequency financial data (HFF-data) observed at regular time intervals (e.g. 1-minute or 5-minute observations), the framework by Andersen and Bollerslev (1998); Peitz and Feng (2015) forms a spatial time series on a (regular) lattice.

Nonparametric estimation of the mean or expectation surface of a functional time series can simply be carried out by bivariate regression methods, which are well studied in the literature. Müller (1988, pp. 77-90) extends the Gasser-Müller (GM) kernel regression estimator (Gasser and Müller, 1979) to multivariate data, discusses the boundary problem and proposes a two-step estimation procedure using product kernels under a rectangular design. The GM-estimator is also applied by Müller and Prewitt (1993) for estimation of the regression function or a (partial) derivative of the functional time series. The authors give detailed asymptotic results for bias and variance and propose an adaptive bandwidth selection procedure based on minimization of the asymptotic mean squared error (AMSE). Optimal selection of global bandwidths for bivariate kernel regression is further studied by Herrmann et al. (1995). Based on minimization of the asymptotic mean integrated squared error (AMISE), formulas for the optimal bandwidths are derived and an iterative plug-in (IPI) algorithm for estimation of these bandwidths from a sample is presented. The authors assess their proposed bandwidth selection procedure at the example of some simulated Gaussian functions with a few hundred observations. Facer and Müller (2003) give detailed theoretical results for multivariate kernel regression under a fixed design assumption. Their spatial model is explicitly designed for finding local maxima of a surface, which represents a response function to a multivariate input under their framework. They used a cross-validation (CV) approach for bandwidth selection, however, this topic was not extensively discussed. Robinson (2011) applied the Nadaraya-Watson (NW) estimator in a spatial setting while allowing for conditional heteroscedasticity in the model. The consistency and asymptotic normality of this estimator under a spatial framework was shown, bandwidth selection for this model was not addressed. Spatial kernel regression of long-memory random fields was studied more recently by Wang and Cai (2010). Spatial kernel regression under random design was considered by Ghosh (2015).

Compared to kernel regression, local polynomial regression methods are advantageous in some ways. They solve the boundary problem (Fan and Gijbels, 1996) and allow for straightforward estimation of derivatives. An overview of local polynomial regression for multivariate local polynomial regression can be found in Scott (2015). Ruppert and Wand (1994) give profound theoretical results for local least squares regression. They study local linear and local quadratic regression under a random design assumption and possible heteroscedasticity in the errors. They extend their findings to higher-order polynomials and estimation of derivatives. A plug-in bandwidth selector for local linear regression is introduced by Yang and Tschernig (1999). Optimal bandwidths are obtained by minimization of the AMISE and the necessary second derivatives of the regression function are estimated using partial local cubic regression. However, their plug-in rule only consists of a single step, requiring an additional estimation of initial bandwidths. Hallin et al. (2004) also investigate the local linear estimation of a multidimensional mixing random field on a rectangular domain. They provide a comprehensive mathematical foundation for the asymptotic properties of this estimator but no bandwidth selection is considered. Wang and Wang (2009) used local linear regression for estimation of spatio-temporal models under local stationarity. An overview of nonparametric estimation of regression surfaces may be found in Ghosh (2018).

1.2 Contribution of this Thesis

A major issue in kernel regression is the boundary problem: at a boundary point, fewer observations are available than the bandwidth would require. This induces a bias in the estimates and leads to unreliable results in this region. A solution approach using specialized boundary kernels is discussed by Gasser and Müller (1979), some explicit formulas for these kernels for boundary correction, the so-called *boundary kernels* are given by Gasser et al. (1985); Müller (1988). However, the most useful and easy-to-construct kernel functions were proposed not until the μ-smooth boundary kernels by Müller (1991) and the (μ, μ')-smooth kernels by Müller and Wang (1994). These kernels allow for a smooth continuation of the interior estimates to the boundaries and ensure that the bias at the boundaries is of the same order of magnitude as in the interior. This type of boundary problem is automatically solved by local regression methods (Fan and Gijbels, 1992; Masry, 1996), hence these estimators are usually applied without concern to the boundaries. However, the naive use of simple truncated interior kernels at the boundaries leads to discontinuities in the regression at these boundaries. These discontinuities might become a severe issue if estimation at the boundaries dominates the complete estimation. For example, in real-time monitoring applications, each new observation is treated as a boundary point and hence, the complete estimation is carried out at the boundaries.

In multivariable regression, the role of the boundary region becomes more important, e.g. under the framework of spatial regression. In comparison with univariate time series, the proportion of the boundary region is increased in the two-dimensional setting of spatial time series. The ratio between boundary region (BR) and inner region (IR) for spatial surfaces is visualized in Figure 1.1. For $h_1 = h_2$, BR covers $4(h - h^2)$ of the total surface, while in the univariate case, it only covers $1 - 2h$. If $h = 0.1$, 36% of the spatial observations are in the boundary region compared to 20% for the univariate case, for $h = 0.2$, these numbers increase to 64%, respective 40%. Hence, in spatial regression, the boundary region should not be treated the same as the interior region, if continuity of the resulting estimates is required.

The findings in Chapter 2 now adapt the results of Müller (1991) and Müller and Wang (1994) to define boundary weighting functions for (univariate) local regression. As boundary effects in local regression do not affect the bias, the terminology *boundary*

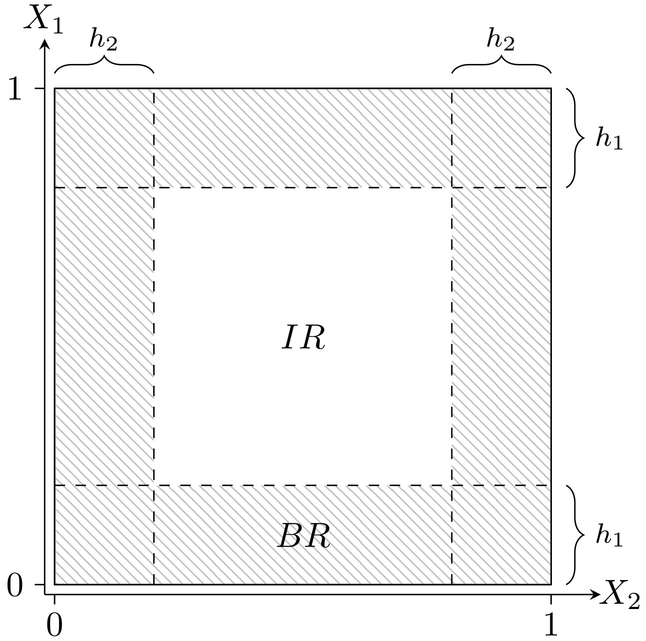


Figure 1.1: *Proportion of boundary region to the total regression surface in spatial regression.*

modification is used rather than *boundary correction*, which remains reserved for kernel regression. The proposed boundary modification in local regression leads to equivalent estimates as the common use of truncated weights in the interior but includes a smooth continuation to the boundaries. This can be demonstrated at an example of real-time monitoring; in Figure 1.2, the wind speed at Fairbanks, AK on 2021-02-01 is smoothed from left to right, where for each point only observations to the left were incorporated in the smoothing procedure. This effectively leads to boundary smoothing only as now each point is treated as an endpoint. While the smoothing results for the boundary modified local regression ("μ-smooth" and "(μ, μ')-smooth", see Chapter 2) exhibit a smooth continuous curve, the "Truncated" type without boundary modification exhibits much more discontinuity. The contribution of the boundary modification is abundantly clear in this example, where each point is treated as a boundary point. However, while restricted to only a few observations and thus probably less obvious, these effects remain for local regression of full data sets. In conclusion, accounting for boundary modification leads to smoother results compared to the naive use of truncated boundary kernels.

The proposed methods for boundary modification open a convenient way to generate boundary kernels of desired orders simply from the weighting functions. These boundary kernels prove useful for kernel regression of spatial time series in Chapter 3, while the boundary-modified local regression is utilized in Chapter 4.

The main contribution of Chapters 3 and 4 is the extension of the double conditional smoothing (DCS) originally proposed by Feng (2013) and its integration into R for practical use of the findings. Under the spatial time series scheme, estimation of surfaces is usually carried out using a bivariate kernel or local regression smoother as stated in

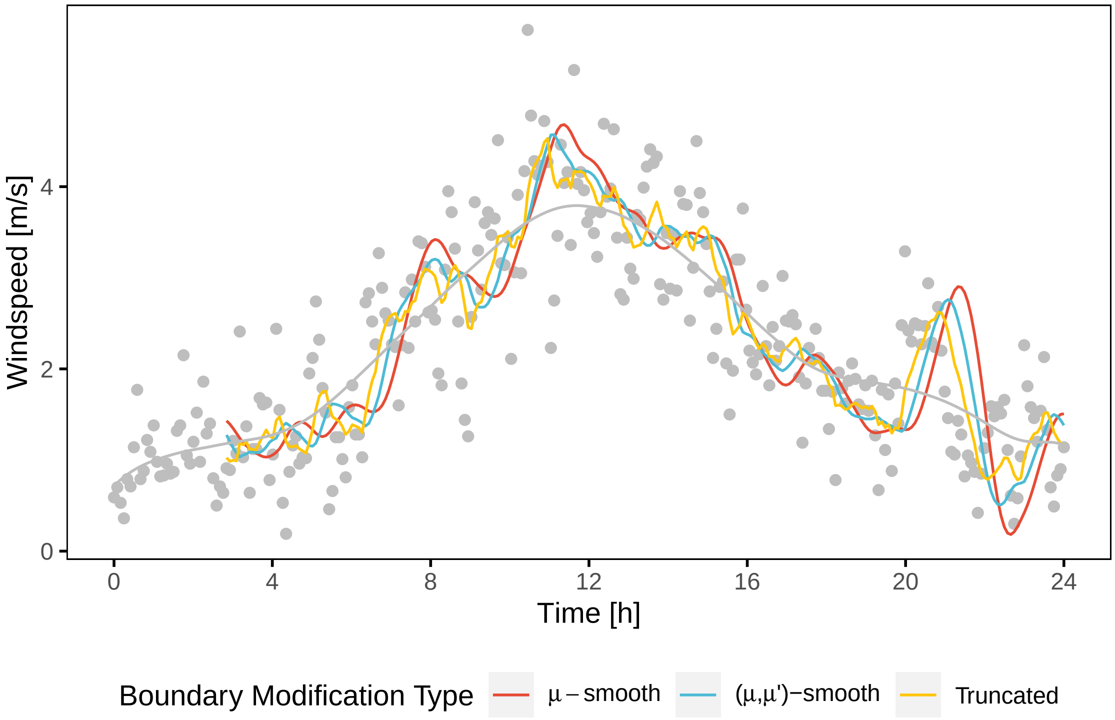


Figure 1.2: *Real-time monitoring of the wind speed at Fairbanks, AK on 2021-02-01 with three different boundary modification regimes. The employed estimation bandwidth is selected by the* `tsmooth` *function from the* smoots-package *to* $h = 0.122$ *using the full sample. Data source: NOAA/Diamond et al. (2013).*

section 1.1. The downside to this approach is, that these bivariate regressors are slow, which will be especially an issue if the size of the data sets increases. The DCS now divides the two-dimensional regression of the classical bivariate smoothers into two one-dimensional smoothers, effectively reducing a problem of $O(n^2)$ to $O(n)$. The run-times of a bivariate regression are compared to those of a DCS regression in Figure 1.3. The simulation employs different sample sizes using observations generated from the model $y_{i,j} = m(x_{1,i}, x_{2,j}) + \varepsilon_{i,j}$, where $m(x_1, x_2) = x_1^2 \cdot \sqrt{x_2}$ and $\varepsilon_{i,j}$ are iid. errors from the standard normal distribution. The values were simulated on a square grid with $n = n_1 \cdot n_2$ observations and $n_1 = n_2$. For each regression method and sample size, 25 simulations were measured. The graph indicates the much faster speed of the DCS and a better linear behavior compared to that of the bivariate regression.

The contributions of Chapter 3 for kernel regression and Chapter 4 for the newly developed local polynomial regression DCS can be described in two ways: in terms of *content* or *objectives*. The main *content* of Chapters 3 and 4 can be broken down into two parts: one regarding the DCS estimation and one regarding the selection of the associated bandwidths. Estimation procedures for kernel regression DCS were already used by Feng (2013) and Peitz and Feng (2015) but without data-driven bandwidth selection; the used bandwidths in these papers were set manually. In this thesis, the asymptotic optimal bandwidths are selected by minimization of the estimators' mean integrated squared error

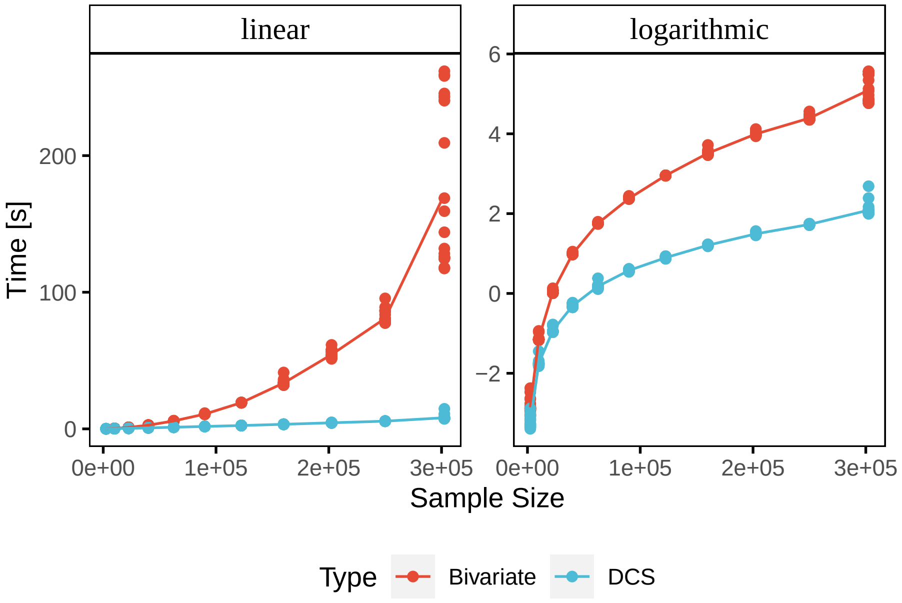


Figure 1.3: *Run-times for classical bivariate and DCS kernel regression for a simulation example with 11 different sample sizes and 25 iterations per sample size. The implementation of both algorithms in R only differs in the algorithm itself. Beside vectorization, no other measures for improving the efficiency were used. Run-times were measured using the* microbenchmark *package.*

(MISE), these optimal bandwidths are selected by a data-driven IPI algorithm. The IPI algorithm iteratively selects the bandwidths converging to the optimal values with certain properties. Thus, it does not depend on initial values or too many parameters. In addition, the number of steps required is much smaller than for other bandwidth selection methods such as CV. The choice of the used inflation method, exponential inflation instead of the more common multiplicative inflation method, further increases the speed of convergence of the IPI algorithm. The DCS estimation algorithm itself is refined by the introduction of the functional DCS (FDCS), which provides an efficient way for smoothing spatial time series on a lattice, by utilizing certain properties of the equidistant lattice under consideration. The FDCS is shown to have the same asymptotic properties as the DCS, for kernel regression as well as for local polynomial regression. The two main *objectives* of the extension of the DCS are to establish a theoretical foundation for the DCS estimators and to develop algorithms for implementation into computer-aided methods, e.g. in R. The theoretical foundation of the kernel regression DCS estimator was incomplete so far, important results such as asymptotic expectation and variance, and, from this, the MISE and the derived asymptotic optimal bandwidths were remaining to be found. This gap is closed by the presented findings. The known asymptotic theory for bivariate and multivariable nonparametric regression is shown to extend to the DCS estimators in general. Algorithms suitable for direct use in R are found with the FDCS for estimation of the mean or regression surface or its derivatives and the IPI bandwidth selection. These algorithms are available for practical use in the *DCSmooth* package for R (see Appendix A).

1.3 Summary of the Contents

In Chapter 2, the boundary modification methods in local regression are introduced. Although important results on their own, the findings of this chapter also serve as preparatory work for Chapters 3 and 4, as the methods for boundary modification in local regression and kernel generation are used later. The local regression is defined using μ-smooth and (μ, μ')-smooth boundary weights, similarly to the ideas of Müller (1991); Müller and Wang (1994) originally used for kernel regression. These boundary weights now allow for a smooth continuation of the estimates in the interior region to the boundary in local regression. The estimates in the interior using the proposed boundary modification are shown to be equivalent to the common method using common truncated methods. The results turn out to lead to a convenient way for generating boundary kernels of desired orders. This method is employed to obtain kernel functions which are used in Chapter 3.

The DCS for kernel regression under iid. errors is studied in Chapter 3. The general nonparametric functional or spatial model is introduced in Section 3.2 along with a summary of the basic ideas of the DCS procedure by Feng (2013); Peitz and Feng (2015). The main assumptions needed for derivation of asymptotic results and optimal bandwidths are set forth and explained. In Section 3.3, a matrix DCS (MDCS) and the FDCS are introduced to improve the computational efficiency of the original DCS algorithm. The FDCS utilizes the assumed equidistant lattice structure to avoid redundant computation of weights and is designed to use the faster vectorization algorithms of R. Section 3.4 treats the selection of optimal bandwidths. Asymptotic results of the expectation and the variance of the proposed estimator are obtained in Section 3.4.1 under the previously defined regularity assumptions. These results lead to the asymptotic optimal bandwidths for the regression by minimization of the MISE. For estimation of these optimal bandwidths, an IPI-algorithm (Gasser et al., 1991) is proposed in Section 3.4.2, which calculates the optimal bandwidths iteratively. This algorithm depends on some partial derivatives of the regression surface. Kernels generated from the method found in Chapter 2 are applied for estimation of these derivatives from the observed spatial time series. The methods developed in Chapter 3 are assessed in the simulation study of Section 3.5 by assessing the bandwidth selection algorithm using well-defined spatial functions with known optimal bandwidths. Examples of application to real HFF-data are shown in Section 3.6, where the algorithm is employed for regression of the expectation surface of the volatility time series of German companies BMW AG and Siemens AG and the corresponding volume surfaces during the financial crisis around 2009. Proofs and derivations of some results of Chapter 3 are moved to the appendix Section 3.8.

The previously defined DCS is extended to local polynomial regression under dependent errors in Chapter 4. Definitions for extension of the DCS to local polynomial smoothers and necessary assumptions to derive asymptotic results are in Section 4.2. The DCS (and FDCS) schemes are shown to work equivalently under local regression and can therefore be used to improve computational efficiency. The boundary modification for local regression proposed in Chapter 2 is applied to the spatial smoother in Section 4.2.3, equivalent kernel estimators are discussed in the following. These equivalent kernels are necessary for proving the equivalency between local regression and kernel regression under the DCS framework. Section 4.3 covers the derivation of the asymptotic bias and variance as well as the optimal bandwidths and, similar to Chapter 3, an IPI bandwidth selection algorithm under dependent errors with short memory. This error structure is explicitly modeled by a spatial ARMA (SARMA) structure (Section 4.4). Some estimation methods for parametric estimation of special classes of the SARMA model are proposed in Section 4.4.2. A simulation study employing simulations of spatial surfaces with SARMA errors is used to assess the proposed algorithms in Section 4.5, Section 4.6 includes an application to financial stock data. Our findings indicate that the bandwidths are usually increased when accounting for dependency structures in the errors and hence negation of dependent errors would lead to an undersmoothing.

It should be emphasized, that this thesis is a collection of the three articles in Chapters Chapter 2 - Chapter 4. Although these three articles form a somehow consecutive series, each one has a more or less different approach to the subject matter and should be able to be read as an independent contribution. Due to this approach, the notation might differ in some places between the chapters in order to meet the different requirements and ensure the internal consistency of each chapter.

Outside of the development of the algorithms for non- and semiparametric estimation of spatial surfaces for financial applications in Chapters 2 - 4, some alternative fields of application are presented in Chapter 5. Examples of environmental data illustrate the broad scope of the presented approach. Some extensions to the model include the introduction of long memory errors into the model or the nonparametric estimation of the variance coefficient used in bandwidth selection using spectral density estimation. Without claiming completeness, these topics are demonstrated at some examples in this chapter. Chapter 6 closes this thesis with some concluding remarks. The documentation of the *DCSmooth* package is given in the appendix, including application examples and descriptions of the methods and tools in the package.

All analyses and calculations provided in this thesis are carried out in *R* (R Core Team, 2021) using version 1.1.2 (2021-10-21) of the *DCSmooth*-package (Schäfer, 2021), specifically developed for smoothing functional time series on a lattice using the DCS procedure. Charts and diagrams shown are produced by the *ggplot2*-package (Wickham, 2016), the surface plots are created with the *plotly*-package (Sievert, 2020). Further, the packages *smoots* (Feng et al., 2021b) and *microbenchmark* (Mersmann, 2019) are used in creation of the examples in Figure 1.2 respective Figure 1.3.

2 Boundary Modification in Local Regression

This chapter is based on joint work with Yuanhua Feng and published with slight differences in the CIE Working Papers (144), Paderborn University, under the title ”Boundary Modification in Local Polynomial Regression”.

2.1 Introduction

An important aspect in nonparametric regression concerns estimation at the margins of the support and the associated boundary modification. Boundary modification is particularly well studied in kernel regression and kernel density estimation. If boundary modification is not considered in this context, the so-called *boundary problem* will affect both, the order of magnitude of the bias and the rate of convergence at a boundary point. The typical boundary modification method is to use some sort of boundary kernels. Different boundary kernels are proposed by Gasser and Müller (1979); Gasser et al. (1985); Granovsky and Müller (1991); Müller (1991); Müller (1993a,b); Müller and Wang (1994) and Kyung-Joon and Schucany (1998) among others. Also, the boundary correction proposed by Rice (1984) can be interpreted as an indirect use of certain boundary kernels, which is further studied by Cheng (2006). Recent developments and a summary of further boundary correction ideas in kernel density estimation may be found in Karunamuni and Alberts (2005) and Karunamuni and Zhang (2008). More recently, the application of boundary kernels in distribution function estimation is also addressed by Tenreiro (2013).

In contrast to this, there seems to be a lack of research on boundary modification in local polynomial regression. The important role of local polynomial regression in nonparametric estimation is well known, as there is extensive literature on this subject, e.g., in papers by Stone (1977); Cleveland (1979); Ruppert and Wand (1994) and Fan and Gijbels (1996). This approach exhibits several attractive features, that is, it is design adaptive (Fan and Gijbels, 1992), is an automatic kernel generator (Hastie and Loader, 1993) and also has automatic boundary correction. The last property implies that, under suitable regularity conditions, the magnitude order of the bias and the rate of convergence of a local polynomial estimator at a boundary point are the same as those at an interior point. It is hence commonly accepted that - as Masry (1996) points out - "*no boundary modification is required*" for local polynomial regression. Note, that the truncated part of the weight function used in the interior is usually employed at a boundary point. This will result in discontinuous estimates in the boundary region. If the continuity of the resulting estimates is required, some suitable method to achieve this property is demanded.

In this chapter, we will propose two new boundary modification methods for local polynomial regression by adapting the ideas of Müller (1991) for generating the so-called μ-smooth boundary kernels and those of Müller and Wang (1994) for generating the (μ, μ')-smooth boundary kernels. The parameter $\mu \geq 0$ is an integer, which quantifies the endpoint smoothness of a polynomial kernel function. In the first case, the resulting estimates in the boundary region are of the same order of smoothness. In the second case, the order of smoothness of the estimate at a boundary point is $\mu - 1$, if that of an estimate in the interior is μ. Consider the use of a common second-order (polynomial) kernel function as the weight function. In the interior, the weight function defined by the first idea is the same as the original weight function. The second method defines two additional (asymmetric) weight functions (if $\mu > 0$), whose standardized forms can be thought of as (a pair of) two *first order kernel functions*. It is shown that, under regularity conditions, the three quite different weighting methods in the interior are equivalent. The counterparts of the Epanechnikov kernel defined under the second method are two (half-formed) triangular kernels, whose form is unchanged from the interior to the corresponding endpoint. This shows that the best weight function at the endpoints is a natural extension of one of the best weight functions in the interior following the second boundary modification method. Moreover, consider a p-th order local polynomial estimator of the ν-th derivative of the regression function. The second kind of new weight functions is shown to generate kernel functions in the interior of order k such that $k - \nu$ is odd. The boundary behavior of local polynomial regression is also discussed in detail. The proposals are particularly useful when one-sided smoothing or detection of change points in nonparametric regression are considered.

The findings of this chapter are useful for the further development of the kernel regression DCS in Chapter 3 and the new development of the local polynomial DCS (LP-DCS) in Chapter 4. Hence, this chapter can be viewed as preparatory work, although the findings presented here are important on their own. Chapter 2 is organized as follows: the proposed new boundary modification methods are introduced and discussed in Section 2.2. The main results on the equivalency of the (μ, μ')-smooth estimators to the commonly used estimators at an interior point are stated in Section 2.3. The proofs for the special cases of local linear, local quadratic and local cubic regression are given. Section 2.4 investigates the boundary behavior of local polynomial regression. We discuss some areas, in which the proposed methods might improve certain results in Section 2.5 and conclude the chapter with some final remarks in Section 2.6. Proofs of auxiliary results and the equivalency of the proposed estimators for $p > 3$ are moved to the chapters appendix in Section 2.7 along with some formulas for equivalent regression kernels.

2.2 Modified Local Polynomial Regression

For simplicity, we consider the smoothing of the time series $\{Y_t\}, t = 1, \dots, n$ under an equidistant design. We employ the widely used simple nonparametric regression model:

$$Y_t = g(x_t) + \varepsilon_t, \tag{2.2.1}$$

where $x_t = t/n \in [0,1]$, g is a smooth regression function on $[0,1]$ and ε_t is a sequence of iid. random variables with zero mean and finite variance $\mathrm{var}(\varepsilon_t) = \sigma_\varepsilon^2 > 0$. Assume that g is at least $(p+1)$-times differentiable at a point x_0. We have

$$g(x) = g(x_0) + g'(x_0)(x - x_0) + \cdots + g^{(p)}(x_0)(x - x_0)^p/p! + R_p \tag{2.2.2}$$

for x in a neighborhood of x_0, where R_p is a remainder term. Given observations y_1, …, y_n, the local polynomial estimator of $g^{(\nu)}(x)$, i.e. the ν-th derivative of g at a point x, is obtained by solving the locally weighted least squares problem

$$\arg\min_{\beta} Q,\ Q = \sum_{t=1}^{n} \left[y_t - \sum_{j=0}^{p} \beta_j (x_t - x)^j \right]^2 W\left(\frac{x_t - x}{h} \right),$$

where h is the bandwidth and W is the weight function. Let $\widehat{\beta} = (\widehat{\beta}_0, \widehat{\beta}_1, \dots, \widehat{\beta}_p)^{\mathrm{T}}$. From (2.2.2) we can see that $\nu!\widehat{\beta}_\nu$ is an estimator of $g^{(\nu)}(x)$, $\nu = 0, 1, \dots, p$.

Usually, the weight function W is a symmetric density $W(u)$ on $[-1,1]$. For a right boundary point $x = 1 - qh$, define the truncated weight function $W_q^0(u) = W(u)\mathbb{1}_{[-1,q]}(u)$ for $0 \leq q \leq 1$, where the discussion at a left boundary point for weights $W_{q,L}(u)$ is analogously. The weight function is assumed to be non-negative throughout this chapter, but it is not required for the weight function to be normalized to one. For $q = 1$, W_q^0 coincides with W. The truncated boundary weight function $W_q^0(u)$ is the naive boundary modification method in local regression, which is often used in the literature without any explanation. A clear drawback of this method is that the weight function used at a boundary point is discontinuous at the endpoint so that the corresponding asymptotically equivalent kernel at a boundary point is also discontinuous. This property will be taken over by the resulting estimators and $\widehat{g}^{(\nu)}$ obtained in this manner is always discontinuous. This deficiency is more clear if change point detection based on local polynomial regression is considered (see e.g. Loader, 1996) because now each point is treated as an endpoint. To overcome this problem, the estimation at boundaries in local regression should be improved, with various ways to define a reasonable boundary modification method. In the following we will consider two of them, corresponding to two well-known classes of boundary kernels.

The first method is obtained by adapting the idea of Müller (1991) for constructing μ-optimal smooth boundary kernels. Following this idea, the weight function at a boundary point is obtained by transforming $W(u)$, the weight function used in the interior, into a

μ-smooth weight function on the support $[-1, q]$, which is symmetric around the central point of the support $u_q = (q-1)/2$. Such a weight function is given by (Müller, 1993b), with

$$W_q^a(u) = W\left(\frac{u - \frac{q-1}{2}}{\frac{q+1}{2}}\right)\mathbb{1}_{[-1,q]}(u) = W\left(\frac{2u + (1-q)}{1+q}\right)\mathbb{1}_{[-1,q]}(u).$$

As pointed out by Müller (1991), $W_q^a(u)$ (and also $W_q^b(u)$ to be stated in (2.2.6)) is only defined for a μ-optimal second order kernel with $W^{(r)}(-1) = W^{(r)}(1) = 0$ for $0 \leq r < \mu$. The weights W_q^a are μ-th smooth for any $0 \leq q \leq 1$ and the resulting estimate $\widehat{g}^{(\nu)}$ at the boundary has the same degree of smoothness as in the interior. For any weight function W, W_q^a coincides with W_q in the interior ($q = 1$), i.e. $W_1^a \equiv W_1 \equiv W$. The mean squared error (MSE) of $\widehat{g}^{(\nu)}$ at an endpoint using W_q^a is however much larger than the MSE of an estimator using W_q.

This problem can be solved as follows. Consider weight functions of the form

$$W(u) = (1-u^2)^{\mu}\mathbb{1}_{[-1,1]} = (1+u)^{\mu}(1-u)^{\mu}\mathbb{1}_{[-1,1]}(u), \tag{2.2.3}$$

for some integer $\mu \geq 0$, which is defined based on some second order kernel function, but is not necessarily normalized. Now, the boundary weight function W_q can be rewritten as

$$W_q(u) = (1+u)^{\mu}(1-u)^{\mu}\mathbb{1}_{[-1,q]}(u) \tag{2.2.4}$$

and W_q^a can be equivalently defined by

$$W_q^a(u) = (1+u)^{\mu}(q-u)^{\mu}\mathbb{1}_{[-1,q]}(u). \tag{2.2.5}$$

For the second boundary modification method, let $\mu' = \max(\mu - 1, 0)$, i.e. $\mu' = 0$ for $\mu = 0$ and $\mu' = \mu - 1$ for $\mu \geq 1$. Following Müller and Wang (1994), a (μ, μ')-smooth boundary weight function on $[-1, q]$ is defined by

$$W_q^b(u) = (1+u)^{\mu}(q-u)^{\mu'}\mathbb{1}_{[-1,q]}(u). \tag{2.2.6}$$

$W_q^a(u)$ and $W_q^b(u)$ are the weight functions associated with the μ- and $(\mu, \mu-1)$-smooth boundary kernels given in Müller (1991) and Müller and Wang (1994) respectively, which can all be automatically generated by means of local polynomials with a proper weight function.

If the uniform weight function is used, the three boundary modification methods coincide with each other. For $\mu = 1$, W_q^a is smooth at $u = q$, but W_q^b is not. The weight functions W_q, W_q^a and W_q^b are shown in Figure 2.1 for $q = 0, 1/3, 2/3$ and 1 for $\mu = 1$ and $\mu = 2$, respectively. Note that for $q = 1$, W_1 and W_1^a coincide with each other.

The larger μ is, the larger is the difference between W_q and W_q^a. On the other hand, the larger μ is, the smaller is the difference between W_q^a and W_q^b. As shown by Feng (1999),

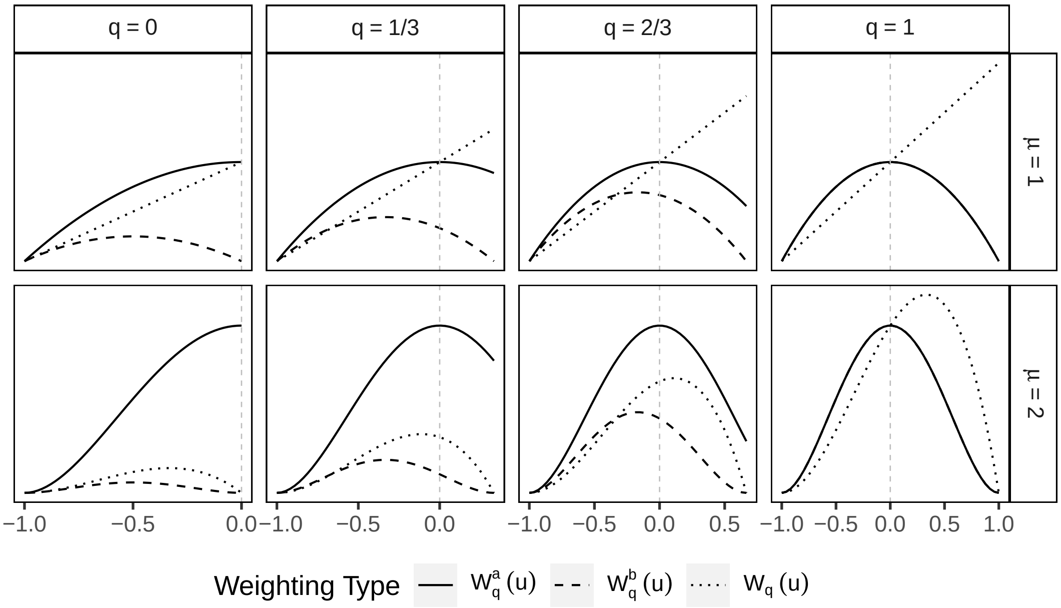


Figure 2.1: *Boundary weighting functions* $W_q^a(u), W_q^b(u)$ *and* $W_q^0(u)$ *corresponding to the* μ*-smooth,* (μ,μ')*-smooth and truncated type for* $q = 0, 1/3, 2/3, 1$ *and* $\mu = 1, 2$.

if the Gaussian kernel (with non-compact support) is used, W_q^a and W_q^b will reduce to the same weight function, because the Gaussian kernel corresponds to the limit case of the weight function defined in (2.2.3) as $\mu \to \infty$.

2.3 Equivalency of the Proposed Weighting Methods

The newly proposed boundary modification methods for local polynomial regression are defined by the use of the μ-smooth weights $W_q^a(u)$ in (2.2.5) and the (μ,μ')-smooth weights $W_q^b(u)$ in (2.2.6). The conventional method without boundary modification employs the truncated kernel $W_q(u)$ in (2.2.4). In the interior, for $q = 1$, it holds that $W_1(u) = W_1^a(u) = W(u)$, hence these weights will only lead to different estimates at boundary points. On the other hand, W, W_1^b and $W_{1,L}^b$ are three quite different weight functions, if $\mu > 0$. However, under some regularity conditions, the resulting estimates in the interior using those different weight functions are exactly the same. That is, as interior weight functions, they are equivalent to each other and the third boundary modification method is also logically defined. For simplicity, denote $W_*(u) \equiv W_1^b(u)$ and $W_{**}(u) \equiv W_{1,L}^b(u)$. Consider at first the estimates obtained with weight functions $W(u)$ and $W_*(u)$, which will be denoted by $\widehat{g}^{(\nu)}(x)$ and $\widehat{g}_*^{(\nu)}(x)$, respectively. Let $h < x < 1-h$ be an interior point and define the matrix of observations by

$$\mathbf{X} = \begin{bmatrix} 1 & u_1 & \cdots & u_1^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & u_n & \cdots & u_n^p \end{bmatrix},$$

where $u_t = (x_t - x)/h$. Let $\mathbf{W}$ and $\mathbf{W}_*$ be the diagonal matrices with entries $W(u_t)$ respectively $W_*(u_t)$. Then $\widehat{g}^{(\nu)}(x)$ and $\widehat{g}_*^{(\nu)}(x)$ are both linear smoothers with the weights

$$\{\mathbf{w}^{\nu}(x)\}^{\mathrm{T}} = \nu!\mathbf{e}_{\nu+1}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W} \tag{2.3.1}$$

and

$$\{\mathbf{w}_*^{\nu}(x)\}^{\mathrm{T}} = \nu!\mathbf{e}_{\nu+1}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{W}_*\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{W}_*, \tag{2.3.2}$$

respectively, where $\mathbf{e}_j$, $j = 1, \ldots, p+1$, denote the j-th $(p+1)\times 1$ unit vector. The following theorem shows that $\widehat{g}^{(\nu)}(x) = \widehat{g}_*^{(\nu)}(x)$ under given conditions:

Theorem 2.1. *Consider the equidistant nonparametric regression of model* (2.2.1). *Let* $0 < h < 0.5$ *be a given bandwidth and* $h < x = x_{i_0} < 1-h$ *be an observation point in the interior. Let* $\widehat{g}^{(\nu)}(x)$ *and* $\widehat{g}_*^{(\nu)}(x)$ *be the estimates obtained with the weight functions* $W(u)$ *and* $W^*(u)$*, respectively. Assume further that* $p - \nu > 0$ *is odd. Then we have* $\widehat{g}^{(\nu)}(x) = \widehat{g}_*^{(\nu)}(x)$ *for any observation vector* $\mathbf{y}$.

The subsequent proof of Theorem 2.1 is structured as follows: we state some notation and the auxiliary results Lemma 2.1 and Lemma 2.2 below. Then, Theorem 2.2 gives sufficient and necessary conditions for Theorem 2.1 to hold and is proved for the important special cases of local linear, local quadratic and local cubic regression. In the appendix in Section 2.7, we propose another necessary and sufficient condition in Theorem 2.5 and prove Theorem 2.1 for $p > 3$. Proofs of the auxiliary Lemmas 2.1, 2.2 as well as that of Theorem 2.2 are given in the appendix as well.

For a matrix C with elements c_{ij}, we denote the submatrix obtained by erasing the i-th row and j-th column of C by C_{ij}. Hence, $|C_{ij}|$ is the corresponding minor determinant. We further denote the cofactor of element c_{ij} by $\widetilde{c}_{ij} = (-1)^{i+j}|C_{ij}|$.

Define $c_j = \sum\limits_{|u_t|\leq 1} W(u_t)u_t^j$ and $c_j^* = \sum\limits_{|u_t|\leq 1} W_*(u_t)u_t^j$, which are the weighted sums of the j-th order powers of u_t. Lemma 2.1 gives some important relationships between these quantities.

Lemma 2.1. *For the quantities* c_j *and* c_j^**, it holds that*

1) $c_j = 0$ *for* j *odd,*

2) $c_j^* - c_{j+1}^* = c_j$,

3) $c_j^* = c_{j+1}^*$ *for* j *odd and*

4) $c_j^* = c_j + c_{j+2}^*$ *for* j *even.*

Define the $(p+1)\times(p+1)$ matrices $C=(c_{ij})=(\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X})$ and $C_*=(c^*_{ij})=(\mathbf{X}^{\mathrm{T}}\mathbf{W}_*\mathbf{X})$, where c_{ij} and c^*_{ij} with double indices denote the (i,j)-th elements of C and C_* respectively. We have

$$C=\begin{pmatrix} c_0 & 0 & c_2 & \cdots & c_{p-1} & 0 \\ 0 & c_2 & 0 & \cdots & 0 & c_{p+1} \\ c_2 & 0 & c_4 & \cdots & c_{p+1} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{p-1} & 0 & c_{p+1} & \cdots & c_{2p-2} & 0 \\ 0 & c_{p+1} & 0 & \cdots & 0 & c_{2p} \end{pmatrix} \tag{2.3.3}$$

and

$$C_*=\begin{pmatrix} c^*_0 & c^*_2 & c^*_2 & \cdots & c^*_{p-1} & c^*_{p+1} \\ c^*_2 & c^*_2 & c^*_4 & \cdots & c^*_{p+1} & c^*_{p+1} \\ c^*_2 & c^*_4 & c^*_4 & \cdots & c^*_{p+1} & c^*_{p+3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c^*_{p-1} & c^*_{p+1} & c^*_{p+1} & \cdots & c^*_{2p-2} & c^*_{2p} \\ c^*_{p+1} & c^*_{p+1} & c^*_{p+3} & \cdots & c^*_{2p} & c^*_{2p} \end{pmatrix} \tag{2.3.4}$$

for p odd and

$$C=\begin{pmatrix} c_0 & 0 & c_2 & \cdots & 0 & c_p \\ 0 & c_2 & 0 & \cdots & c_p & 0 \\ c_2 & 0 & c_4 & \cdots & 0 & c_{p+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & c_{p+2} & 0 & \cdots & c_{2p-2} & 0 \\ c_p & 0 & c_{p+2} & \cdots & 0 & c_{2p} \end{pmatrix} \tag{2.3.5}$$

and

$$C_*=\begin{pmatrix} c^*_0 & c^*_2 & c^*_2 & \cdots & c^*_p & c^*_p \\ c^*_2 & c^*_2 & c^*_4 & \cdots & c^*_p & c^*_{p+2} \\ c^*_2 & c^*_4 & c^*_4 & \cdots & c^*_{p+2} & c^*_{p+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c^*_p & c^*_{p+2} & c^*_{p+2} & \cdots & c^*_{2p-2} & c^*_{2p} \\ c^*_{p+2} & c^*_{p+2} & c^*_{p+4} & \cdots & c^*_{2p} & c^*_{2p} \end{pmatrix} \tag{2.3.6}$$

for p even.

For any observation vector $\mathbf{y}$, the equality $\widehat{g}^{(\nu)}=\widehat{g}^{(\nu)}_*$ holds if and only if $\mathbf{w}^{\nu}=\mathbf{w}^{\nu}_*$ under the conditions of Theorem 2.1. Define $D=(d_{ij})=C^{-1}$ and $D_*=(d^*_{ij})=C_*^{-1}$,

$i, j = 1, 2, \dots, p+1$. As C, C_* are symmetric matrices, i.e. $C^T = C, C_*^T = C_*$, the elements of D, D_* in terms of the cofactors of C_{ij}, C_{ij}^* are

$$d_{ij} = \frac{\widetilde{c}_{ij}}{|C|}, \quad d_{ij}^* = \frac{\widetilde{c}_{ij}^*}{|C_*|}. \tag{2.3.7}$$

Denote the $(\nu+1)$-th rows of D and D_* by $D^{\nu+1}$ and $D_*^{\nu+1}$ respectively. From (2.3.1) and (2.3.2), the condition $D^{\nu+1}\mathbf{X}^{\mathrm{T}}\mathbf{W} = D_*^{\nu+1}\mathbf{X}^{\mathrm{T}}\mathbf{W}_*$ is equivalently to $\mathbf{w}^{\nu} = \mathbf{w}_*^{\nu}$. Let $\widetilde{\mathbf{X}}$ be the $n \times n$ diagonal matrix with elements $(1-u_t)$, $t = 1, \dots, n$. Then we have $\widetilde{\mathbf{X}}\mathbf{W}_* = \mathbf{W}$ and $\mathbf{w}^{\nu} = \mathbf{w}_*^{\nu}$ if and only if

$$D_*^{\nu+1}\mathbf{X}^{\mathrm{T}} = D^{\nu+1}\mathbf{X}^{\mathrm{T}}\widetilde{\mathbf{X}}.$$

Using 4) of Lemma 2.2, we obtain

$$D^{\nu+1} = (d_{\nu+1,1}, 0, d_{\nu+1,3}, 0, \dots, d_{\nu+1,p}, 0)$$

for p odd and any $\nu < p$ even and

$$D^{\nu+1} = (0, d_{\nu+1,2}, 0, \dots, d_{\nu+1,p}, 0)$$

for p even and any $\nu < p$ odd. Theorem 2.1 hence reduces to

Theorem 2.2. *Theorem 2.1 holds if, under the same conditions,*

$$D_*^{\nu+1} = (d_{\nu+1,1}, -d_{\nu+1,1}, d_{\nu+1,3}, -d_{\nu+1,3}, \dots, d_{\nu+1,p}, -d_{\nu+1,p}) \tag{2.3.8}$$

for p odd and any $\nu < p$ even, and

$$D_*^{\nu+1} = (0, d_{\nu+1,2}, -d_{\nu+1,2}, d_{\nu+1,4}, -d_{\nu+1,4}, \dots, d_{\nu+1,p}, -d_{\nu+1,p}) \tag{2.3.9}$$

for p even and any $\nu < p$ odd.

We now give the proofs for the special cases of local linear ($p = 1$), local quadratic ($p = 2$) and local cubic ($p = 3$) regression. A proof for $p > 3$ can be found in Section 2.7.

Local Linear

Proof. Let $C = (c_{ij}) = (\mathbf{X}^T\mathbf{W}\mathbf{X})$ and $C_* = (c_{ij}^*) = (\mathbf{X}^T\mathbf{W}_*\mathbf{X})$. For local linear regression we have

$$C = \begin{pmatrix} c_0 & 0 \\ 0 & c_2 \end{pmatrix} \text{ and } C_* = \begin{pmatrix} c_0^* & c_2^* \\ c_2^* & c_2^* \end{pmatrix}.$$

We have $p = 1$ and only $\nu = 0$ (i.e. for estimating g itself) satisfies the condition $p - \nu$ odd. Using (2.3.7) and Lemma 2.1 leads to

$$d^*_{1,1} = \frac{c^*_2}{c^*_2(c^*_0 - c^*_2)} = \frac{1}{c^*_0 - c^*_2} = \frac{1}{c_2} = d_{1,1},$$

because $c^*_0 - c^*_2 = c_2$ and

$$d^*_{1,2} = \frac{-c^*_2}{c^*_2(c^*_0 - c^*_2)} = -d^*_{1,1} = -d_{1,1}.$$

□

Local Quadratic

Proof. For local quadratic regression we have

$$C = \begin{pmatrix} c_0 & 0 & c_2 \\ 0 & c_2 & 0 \\ c_2 & 0 & c_4 \end{pmatrix}$$

and

$$C_* = \begin{pmatrix} c^*_0 & c^*_2 & c^*_2 \\ c^*_2 & c^*_2 & c^*_4 \\ c^*_2 & c^*_4 & c^*_4 \end{pmatrix}.$$

Now, only $\nu = 1$ (i.e. for estimating g') satisfies the condition $p - \nu$ odd. Observing (2.3.7) and Lemma 2.1, straightforward calculation leads to

$$|C| = c_2(c_0c_4 - c_2^2) \quad \text{and} \quad |C_*| = c_2(c_0c^*_4 - c_2c^*_2).$$

It is easy to see that $|C^*_{2,1}| = 0$ and hence $d^*_{2,1} = 0$. Furthermore we have $|C_{2,2}| = c_0c_4 - c_2^2$ and $|C^*_{2,2}| = c_0 * c^*_4 - c_2 * c^*_2$. This leads to

$$d^*_{2,2} = \frac{1}{c_2} = d_{2,2}.$$

Finally, it is clear that $|C^*_{2,3}| = |C^*_{2,2}|$. This results in $d^*_{2,3} = -d^*_{2,2} = -d_{2,2}$. □

Local Cubic

Proof. For local cubic regression we have

$$C = \begin{pmatrix} c_0 & 0 & c_2 & 0 \\ 0 & c_2 & 0 & c_4 \\ c_2 & 0 & c_4 & 0 \\ 0 & c_4 & 0 & c_6 \end{pmatrix}$$

and

$$C_* = \begin{pmatrix} c_0^* & c_2^* & c_2^* & c_4^* \\ c_2* & c_2^* & c_4^* & c_4^* \\ c_2^* & c_4^* & c_4^* & c_6^* \\ c_4^* & c_4^* & c_6^* & c_6^* \end{pmatrix}.$$

Now, $\nu = 0$ and $\nu = 2$ satisfy the condition $p - \nu$ odd. Only the calculation in the case with $\nu = 2$, i.e. for estimating the second derivatives with local cubic regression, will be given in detail. The results for $\nu = 0$ are similar and omitted.

Following (2.3.7) and Lemma 2.1, we have

$$|C| = (c_0c_4 - c_2^2)(c_2c_6 - c_4^2) \quad \text{and} \quad |C_*| = (c_0c_4 - c_2^2)(c_2c_6^* - c_4c_4^*).$$

It is easy to show that $d_{3,1} = -\frac{c_2}{c_0c_4 - c_2^2}$. Straightforward calculation leads to

$$|C_{3,1}^*| = \begin{vmatrix} c_2^* & c_2^* & c_4^* \\ c_2^* & c_4^* & c_4^* \\ c_4^* & c_6^* & c_6^* \end{vmatrix} = \begin{vmatrix} 0 & c_2 & c_4^* \\ c_2 & 0 & c_4^* \\ c_4 & 0 & c_6^* \end{vmatrix} = -c_2(c_2c_6^* - c_4c_4^*).$$

That is $d_{3,1}^* = \frac{|C_{3,1}^*|}{|C_*|} = -\frac{c_2}{c_0c_4 - c_2^2} = d_{3,1}$. Furthermore, we have

$$|C_{3,2}^*| = \begin{vmatrix} c_0^* & c_2^* & c_4^* \\ c_2^* & c_4^* & c_4^* \\ c_4^* & c_6^* & c_6^* \end{vmatrix} = - \begin{vmatrix} c_2 & c_2 & c_4^* \\ c_2 & 0 & c_4^* \\ c_4 & 0 & c_6^* \end{vmatrix} = -c_2(c_2c_6^* - c_4c_4^*) = |C_{3,1}^*|,$$

and hence $d_{3,2}^* = -d_{3,1}^* = -d_{3,1}$. Note that, although the two (1,1)-th elements of $C_{3,1}^*$ and $C_{3,2}^*$ are different, they have however both zero cofactors. Similar calculations can be carried out for $C_{3,3}^*$ and $C_{3,4}^*$. □

Theorem 2.1 gives some exact, finite sample results for equidistant local polynomial regression. It shows (together with Theorem 2.3) that some local regression estimators with quite different weight functions can be identical. For given p, these results hold for all $\nu < p$ with $p - \nu$ odd. For instance, if $p = 3$, they hold for $\nu = 0$ and $\nu = 2$, i.e. for the estimation of g and g''.

Note that the assumption of $p - \nu$ being odd is often made in local regression so that the bias in the interior and at the boundary is of the same order. Theorem 2.1 does not hold for local polynomial estimates $\widehat{g}^{(\nu)}$ and $\widehat{g}_*^{(\nu)}$ obtained with $p - \nu$ even. For instance, it is easy to show that local constant estimates $\widehat{g}$ and $\widehat{g}_*$ in the interior are different.

For a nonparametric regression under equidistant design, the assumption that $h < x = x_{i_0} < 1 - h$ is an observation point in the interior, implies that $q = 1$ and all weights $W(u_t) \neq 0$ are symmetric around $W(u_{i_0})$ and the u_t^j for $|u_t| \leq 1$ are either symmetric (for j even) or asymmetric (for j odd) around $u_{i_0} = 0$. These facts are used in the proof of Theorem 2.1. At an interior point x, which is not an observation point, or for non-equidistant design nonparametric regression, results of Theorem 2.1 only hold approximately.

For estimation at the left boundary, let $\widehat{g}_{**}(x)$ denote the local polynomial estimator obtained with the weight function $W^{**}(u)$ in the interior, where $W^{**}(u) \equiv W^b_{1,L}(u)$ as above. This leads to

Theorem 2.3. *Let $\widehat{g}_{**}(x)$ denote the local polynomial estimator obtained with the weight function $W_{**}(u)$ at an observation point in the interior. Then, under the assumptions of Theorem 2.1, we have $\widehat{g}^{(\nu)}(x) = \widehat{g}_{**}^{(\nu)}(x)$.*

The proof of Theorem 2.3 is analogously to the proof of Theorem 2.1 and is omitted. The third alternative boundary modification method using (2.2.6) is shown to be also well-defined and reasonable by Theorems 2.1 and 2.3. The results presented in this section show in particular that some local regression estimators obtained using different weight functions are equivalent to each other under certain assumptions. Other reasonable boundary modification methods in local regression might exist as well.

2.4 Boundary Behavior of Local Regression

In the following, we will discuss the asymptotic boundary behavior of a local polynomial estimator. It is well known that local polynomial regression has automatic boundary correction property without explicit use of boundary kernels the boundary kernels will be generated automatically. Local regression with the two boundary modification methods introduced above will be utilized for straightforward generation of the two classes of boundary kernels discussed by Müller (1991) and Müller and Wang (1994)

Let $g^{(\nu)}$ be estimated by a p-th order local regression estimator with $p - \nu$ odd. At first, the naive boundary modification method with $W_q(u)$ is used. Then, a local regression estimator is asymptotically equivalent to some kernel estimator of order $k = p + 1$, not only in the interior but also at a boundary point. Following Müller (1987), we have

$$\lim_{n\to\infty} \sup_{1 \leq t \leq n} \left| \frac{w_t^\nu}{w_{tk}^\nu} - 1 \right| = 0, \quad \text{defining } \frac{0}{0} \equiv 1, \tag{2.4.1}$$

where $\{w_t^\nu\}$ denotes the weighting system obtained by a local regression and $\{w_{tk}^\nu\}$ those obtained by some kernel method. Related results may also be found e.g. in Lejeune

(1985); Lejeune and Sarda (1992) and Ruppert and Wand (1994). Let K_q denote the corresponding (equivalent) boundary kernel, which does not belong to the classes of μ-smooth or (μ, μ')-smooth boundary kernels described by Müller (1991) and Müller and Wang (1994). These boundary kernels may be called optimal $(\mu, 0)$ smooth boundary kernels, since they are non-smooth at the endpoint $u = q$ for $q < 1$. If locally unweighted regression (with the Uniform kernel as the weight function) is used, then the generated boundary kernels are the so-called minimum variance kernels introduced by Gasser and Müller (1979). If the Epanechnikov kernel is used as the weight function in the interior, the resulting kernels K_q are the so-called optimal boundary kernels proposed by Gasser et al. (1985).

In the following, we will extend the above results to a general boundary modification method. Let W_q denote any reasonable weight function at the boundary with support $[-1, q]$, which can, for instance, be one of W_q^0, W_q^a or W_q^b, $q \in [0, 1]$ defined above. For convenience, assume that $W_q(u)$ is standardized with integral one. By extending the results in (2.4.1), we obtain

Theorem 2.4. *Let $\widehat{g}^{(\nu)}$ be a p-th order local regression estimator of $g^{(\nu)}$ with $p - \nu$ odd. Let $k = p + 1$. Assume that the bandwidth h satisfies $h \to 0$, $nh^{2\nu-1} \to \infty$ as $n \to \infty$. Then $\widehat{g}^{(\nu)}$ is asymptotically equivalent to a kernel estimator with the boundary kernel*

$$K_q(u) = (a_0 + a_1 u + \cdots + a_{k-1} u^{k-1}) W_q(u), \tag{2.4.2}$$

where a_0, a_1, …, a_{k-1} are the unique solutions of the system of k linear equations

$$\mathbf{N}_{pq} \mathbf{a} = \nu! \mathbf{e}_{\nu+1},$$

where

$$\mathbf{N} = \begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_p \\ \mu_1 & \mu_2 & \cdots & \mu_{p+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mu_p & \mu_{p+1} & \cdots & \mu_{2p} \end{bmatrix},$$

where $\mu_j = \int_{-1}^{q} u^j W_q(u) du$ is the j-th moment of W_q, $\mathbf{a} = (a_0, a_1, \ldots, a_p)^{\mathrm{T}}$ and $\mathbf{e}_{\nu+1}$ is as defined before.

The proof of Theorem 2.4 can be obtained by adapting known results in the literature and is hence omitted. It is clear that the solutions of this linear system are unique.

The boundary kernels $K_q(u)$ generated by local polynomial regression depend strongly on the form of the boundary weight function, through W_q^o on the r.h.s. of (2.4.2) and the change in the solutions of a_0, …, a_p. If W_q^0 is used as weight function, then the boundary kernels generated by local polynomial regression are discontinuous at the point $x = q$ for $q \neq 1$ due to the discontinuity of W_q^0 itself, no matter how smooth $W(u)$ in the interior is.

This drawback is overcome by using the two smooth boundary modification methods introduced in Section 2.2. Now, due to the uniqueness of the solutions, the boundary kernels proposed by Müller (1991) and Müller and Wang (1994) will be generated respectively[1]. Another advantage by using the ideas in Müller (1991); Müller and Wang (1994) is that, in this case, the solution (2.4.2) may be represented more easily by means of orthonormal polynomials associated with the weight function W_q^a or W_q^b (see Müller (1991) for a general explicit solution). Such a representation would be arduous if the boundary weight function W_q^0 is used, because, for general μ, the solutions of the orthonormal polynomials associated with W_q are highly complex.

We provide a special solution to a $(2,0,0)$-kernel. Some further μ-smooth and (μ,μ')-smooth boundary kernels generated via Theorem 2.4 from $W_q^a(u)$ and $W_q^b(u)$ are given in Table 2.1 and Table 2.2 and plotted in Figure 2.2 for $q = 0, 0.5, 1$. Consider the simple case for estimation of g with local linear regression ($p = 1$, $k = 2$), the asymptotically equivalent boundary kernel using a boundary weight function W_q is given by

$$\begin{aligned} & K_q(u) = (a_0 + a_1 u) W_q(u), \\ \text{with} \quad & \alpha_0 = \frac{\mu_2}{\mu_2 - \mu_1^2} \quad \text{and} \quad a_1 = \frac{-\mu_1}{\mu_2 - \mu_1^2}. \end{aligned} \tag{2.4.3}$$

For the Uniform kernel ($\nu = 0$), the solutions for all of the three boundary modification methods are the same with $W_q(u) = \mathbb{1}_{[-1,q]}(u)$,

$$\frac{\mu_1}{\mu_0} = \frac{q-1}{2} \quad \text{and} \quad \frac{\mu_2}{\mu_0} = \frac{q^2 - q + 1}{3}.$$

Inserting these into (2.4.3), we obtain

$$a_0 = 1 + 3\frac{(1-q)^2}{(1+q)^2} \quad \text{and} \quad a_1 = 6\frac{1-q}{(1+q)^2}.$$

This leads to

$$\begin{aligned} K_q(u) &= \frac{1}{1+q}\left[1 + 3\left(\frac{1-q}{1+q}\right)^2 + 6\frac{1-q}{(1+q)^2}u\right]\mathbb{1}_{[-1,q]}(u) \\ &= \frac{1}{(q+1)^3}\left[(4q^2 - 4q + 4) + (-6q + 6)u\right]\mathbb{1}_{[-1,q]}(u). \end{aligned}$$

---

[1] The arguments of the kernel functions used in kernel estimators and local polynomial estimators are often different. The $K_q(u)$ generated from Theorem 2.4 is indeed a right boundary kernel, if μ_j follows the definition of the kernel properties by Müller (1991); Müller (1993b); Müller and Wang (1994).

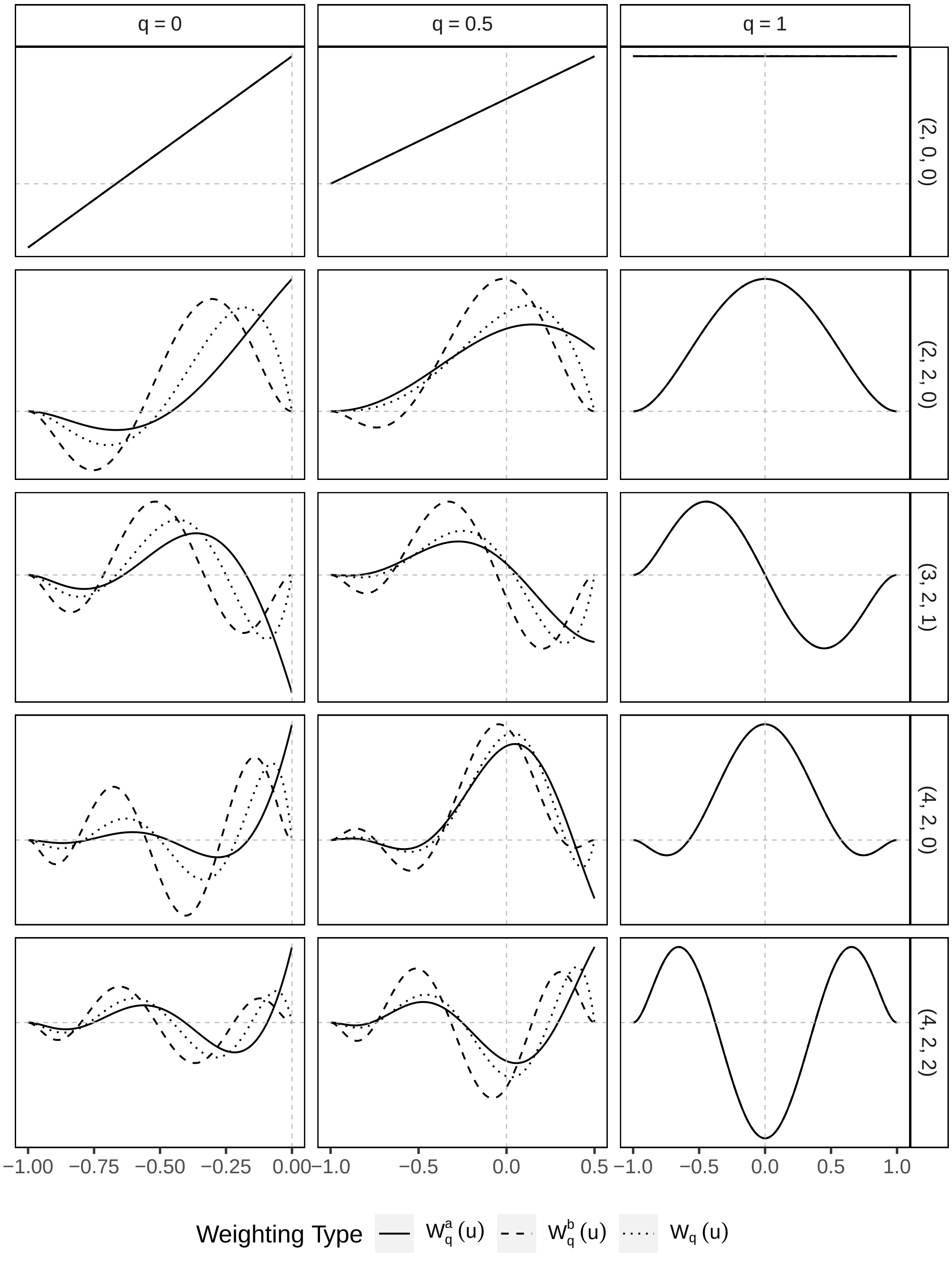


Figure 2.2: *Kernels of order (k, μ, ν) generated by the equivalent kernel method of Theorem 2.4. Values are displayed for $q = 0, 0.5, 1$ and all three boundary modification methods.*

2.5 Areas of Application

Although local polynomial regression solves the boundary problem, which affects the bias and convergence rate of kernel regression, some boundary effects are still present. In this chapter, we focused on the degree of smoothness at the boundaries and were interested in a smooth continuation to the boundary regions of the weighted regression in the interior. A boundary modification in this sense might lead to improved results compared to the naive local polynomial regression, notably in cases, in which the boundary region dominates the interior region.

In real-time or online monitoring applications, data points are evaluated as they occur, and every new value is a boundary point. If the continuous estimation of a smooth trend is required, boundary modification as proposed in this chapter is necessary. If truncated kernels are used, the estimated trend function would be discontinuous at every point as a sequence of endpoints cannot be estimated smoothly. Examples for such applications might be temperature or pollution monitoring, the tracking of financial measures such as stock or commodity prices, or technical measurements in industrial facilities.

A similar argument can be made for nonparametric change point detection, which was investigated, for instance, by Müller (1992) and Loader (1996). For a given set of observations, one-sided regression is applied from both sides, where every observation in the interior is considered as the endpoint of the observations left to it and as the endpoint of the observations right to it.

The ratio of the boundary regression increases with the dimensionality of the data. Hence, for the same bandwidth, the boundary effects are more severe. Consider for example a bandwidth $h = 0.2$ and a boundary region $BR_i = \{0 \leq x_i \leq h_i \cup 1 - h_i \leq x_i \leq 1\}$, where i is the dimension index (e.g. $i = 1$ in the univariate case). For univariate regression, the boundary ratio is 0.4, while in a bivariate regression model, the boundary ratio $BR_1 \cup BR_2$ becomes 0.64 if $h_1 = h_2 = 0.2$. Thus, for a bandwidth of $h = 0.2$, the boundary region covers more than half of all observations. If estimation of derivatives of the regression surface is considered, the bandwidth increases in general, leading to much larger boundary ratios. These derivatives frequently appear in plug-in bandwidth selection rules, as proposed e.g. by Gasser et al. (1991); Herrmann et al. (1992) and Herrmann et al. (1995). If these plug-in or iterative plug-in algorithms are applied to local polynomial regression, smooth extension to the boundaries improves the stability of the results. We use this application of the proposed boundary modification methods in the subsequent chapters.

2.6 Final Remarks

In this chapter, we propose two new methods for boundary modification in local regression, which correspond to the boundary kernels of Müller (1991) and Müller and Wang (1994). This type of boundary modification allows for a smooth continuation of estimates at the boundary region. The newly found estimation methods are shown to be equivalent to the conventional use of truncated kernels in the interior. The proposed methods establish a convenient way for obtaining boundary correction kernels in the sense of Müller (1991) and Müller and Wang (1994) simply from their regression weights, which is especially useful for computing higher-order kernels.

Our focus is on the continuous extension of the smoother to boundary points, optimal kernels are not in the scope of this chapter. However, Müller and Wang (1994) showed that the (optimal) MSE using the $(\mu, \mu-1)$-th smooth boundary kernel is generally smaller than that using the μ-th smooth boundary kernels. It is expected that the MSE obtained using the naive boundary modification method would still be smaller. A more interesting question is which weight function is optimal. Cheng et al. (1997) obtained related results for estimation at the endpoints. Their results indicate that, e.g., for a local linear estimator of g, none of the three methods is optimal. A general answer to this question at an arbitrary boundary point with $q \in (0, 1)$ is still unknown.

2.7 Appendix

Proof of Theorem 2.1 Note that C in (2.3.3) or (2.3.5) is a special matrix, whose (i,j)-th entries with $i+j$ odd are all zero. For all p, it holds that $C = C^T$ and $C_* = C_*^T$. The following lemma provides some useful properties of such a matrix.

Lemma 2.2. *Let $\{E\}_{ij} = (e_{ij})$ be an $m \times m$ matrix with $e_{ij} = 0$ for all elements where $i+j$ is odd. Then*

1) *$|E| = |A| \cdot |B|$, where A is the matrix obtained from E on erasing the even rows and even lines and B is that on erasing the odd rows and odd lines.*
2) *For i and j both odd, let $r = (i+1)/2$, $s = (j+1)/2$ and A_{rs} denote the (r,s)-th minor of the matrix A defined in 1). Then the cofactor of e_{ij} is either $|A_{rs}||B|$ or $-|A_{rs}||B|$.*
3) *For i and j both even, let $r = i/2$, $s = j/2$ and B_{rs} denote the (r,s)-th minor of the matrix B defined in 1). Then the cofactor of e_{ij} is either $|A||B_{rs}|$ or $-|A||B_{rs}|$.*
4) *For $i+j$ odd, the cofactor of the element e_{ij} (i.e. of a zero element) is zero. Then E_{ij} corresponding to a nonzero element is zero.*

Using the notation of cofactors from (2.3.7), Theorem 2.2 can be reduced further.

Theorem 2.5. *The results given in* (2.3.8) *and* (2.3.9) *are respectively equivalent to*
Case 1:

a) $\frac{|C^*_{\nu+1,i}|}{|C_*|} = \frac{|C_{\nu+1,i}|}{|C|}$

b) $|C^*_{\nu+1,i+1}| = |C^*_{\nu+1,i}|$, $i = 1, 3, \ldots, p$,

for p odd and ν even with $0 \leq \nu < p$.
Case 2:

a) $|C^*_{\nu+1,1}| = 0$

b) $\frac{|C^*_{\nu+1,i}|}{|C_*|} = \frac{|C_{\nu+1,i}|}{|C|}$c) $|C^*_{\nu+1,i+1}| = |C^*_{\nu+1,i}|$, $i = 2, 4, \ldots, p$,

for p even and ν odd with $1 \leq \nu < p$.

The proof of Theorem 2.5 is straightforward and is omitted. Note that $|C^*_{\nu+1,i+1}| = |C^*_{\nu+1,i}|$ means that these two cofactors have the same absolute value but with different sign. We are now in position to prove Theorem 2.1:

Proof of Theorem 2.1. From the considerations above, Theorem 2.5, is a sufficient (an necessary) condition for Theorem 2.1 to hold.

Note that the matrix C is a special case of those considered in Lemma 2.2. Hence we have $|C| = |A| \cdot |B|$, where A is the matrix obtained from C on erasing the even rows and lines and B is that on erasing the odd rows and lines. We assume $p \geq 3$, so that both of A and B are at least of size 2×2.

Case 1: ν is even and hence $\nu+1$ is odd. Note that C_* only consists of c_j for j even. From the relationship $c_j^* = c_j + c_{j+2}^*$ for j even and by subtracting the $(j+1)$ th column from the j-th column of C_* for all $j = 1, 2, \ldots, p$, we obtain

$$\overline{C}_* = \begin{pmatrix} c_0 & 0 & c_2 & \cdots & c_{p-1} & c_{p+1}^* \\ 0 & c_2 & 0 & \cdots & 0 & c_{p+1}^* \\ c_2 & 0 & c_4 & \cdots & c_{p+1} & c_{p+3}^* \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{p-1} & 0 & c_{p+1} & \cdots & c_{2p-2} & c_{2p}^* \\ 0 & c_{p+1} & 0 & \cdots & 0 & c_{2p}^* \end{pmatrix}$$

and

$$|C_*| = |\overline{C}_*|.$$

Following 4) of Lemma 2.2, the cofactor of the $(j,p+1)$-th element in the $(p+1)$-th column of $\overline{C}_*$ is zero, if j is odd and these elements can be replaced by zero without affecting the determinant of $\overline{C}_*$. Doing this and carrying out similar interchanges as for C, we get

$$\begin{aligned} |C_*| &= |\overline{C}_*| \\ &= \begin{vmatrix} c_0 & 0 & c_2 & \cdots & c_{p-1} & 0 \\ 0 & c_2 & 0 & \cdots & 0 & c_{p+1}^* \\ c_2 & 0 & c_4 & \cdots & c_{p+1} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{p-1} & 0 & c_{p+1} & \cdots & c_{2p-2} & 0 \\ 0 & c_{p+1} & 0 & \cdots & 0 & c_{2p}^* \end{vmatrix} \\ &= \begin{vmatrix} A & \mathbf{0} \\ \mathbf{0} & B_* \end{vmatrix} \\ &= |A| \cdot |B_*|, \end{aligned}$$

where A is the same as that for C and B_* is obtained from B by replacing the last column with $(c_{p+1}^*, c_{p+3}^*, \ldots, c_{2p}^*)^{\mathrm{T}}$. This leads to

$$\frac{|C|}{|C_*|} = \frac{|B|}{|B_*|}. \tag{2.7.1}$$

To show that the ratio between the cofactors $\widetilde{c}_{\nu+1,i}$ and $\widetilde{c}_{\nu+1,i}^*$ for i odd is that in (2.7.1), let i be odd and define $r = (\nu+2)/2$ and $s = (i+1)/2$. From 2) of Lemma 2.2, either $\widetilde{c}_{\nu+1,i} = |A_{rs}||B|$ or $\widetilde{c}_{\nu+1,i} = -|A_{rs}||B|$, where A_{rs} is the same as in 2) of Lemma 2.2 for the matrix C. To calculate $\widetilde{c}^*_{\nu+1,i}$, we transform the submatrix $C^*_{\nu+1,i}$ by subtracting and interchanging rows and columns to

$$\overline{C}^*_{\nu+1,i} = \begin{pmatrix} A_{rs} & \widetilde{B}_* \\ \mathbf{0} & B_* \end{pmatrix} \tag{2.7.2}$$

where A_{rs} and B_* are as defined above. Following (2.7.6) the cofactor of $c^*_{\nu+1,i}$ is either

$$|\overline{C}^*_{\nu+1,i}| = |A_{rs}||B| \text{ or } |\overline{C}^*_{\nu+1,i}| = -|A_{rs}||B|.$$

Furthermore, the required number of interchanges is the same for calculation of the cofactors of $c_{\nu+1,i}$ and $c^*_{\nu+1,i}$, i.e. the sign is either both changed or both unchanged. We obtain the ratio between these two cofactors

$$\frac{\widetilde{c}^*_{\nu+1,i}}{\widetilde{c}_{\nu+1,i}} = \frac{|A_{rs}||B_*|}{|A_{rs}||B|} = \frac{|B_*|}{|B|} = \frac{|C_*|}{|C|}.$$

This proves the results in a) of Case 1.

Furthermore, the two submatrices $C^*_{\nu+1,i}$ and $C^*_{\nu+1,i+1}$ differ from each other only in one column. Analogously, $C^*_{\nu+1,i+1}$ can be transformed by subtracting and interchanging rows and columns to yield

$$\tilde{C}^*_{\nu+1,i+1} = \begin{pmatrix} A_{rs} & \check{B}_* \\ \check{A}_{rs} & B_* \end{pmatrix},$$

where A_{rs} and B_* are the same as in (2.7.2), all elements of $\check{B}_*$ are zero except for those in the last column and all elements of $\check{A}_{rs}$ are zero except for those in the s-th column, i.e. $\check{A}_{rs}$ has at least one column with all zero, which ensures that the any element of $\check{B}_*$ has zero cofactor and, for calculating the determinant of $\overline{C}^*_{\nu+1,i+1}$, $\check{B}_*$ can be replaced by a zero block. This leads to either

$$|\tilde{C}^*_{\nu+1,i+1}| = |A_{rs}||B| \text{ or } |\tilde{C}^*_{\nu+1,i+1}| = -|A_{rs}||B|.$$

Again, the required numbers of interchanges by the transformations to calculate the cofactors $\widetilde{c}^*_{\nu+1,i}$, $\widetilde{c}^*_{\nu+1,i+1}$ are the same, implying $|C^*_{\nu+1,i+1}| = |C^*_{\nu+1,i}|$. Results of Case 1 hold.

Case 2: a) Now, p is even and $1 \leq \nu < p$ is odd. We carry out a similar transformation from C_* to $\overline{C}_*$ as in Case 1. Because $c^*_{\nu+1,1}$ is in the first column of C_*, $C^*_{\nu+1,1}$ can be transformed similarly. The transformed matrix is now equal the corresponding minor of $\tilde{C}_*$, denoted by $\tilde{C}^*_{\nu+1,1}$. Note that, generally, this does not hold for the cofactor of an element in other column. Hence we have $|C^*_{\nu+1,1}| = \pm|\tilde{C}^*_{\nu+1,1}| = 0$. That is $d^*_{\nu+1,1} = 0$.

The proof of results in b) and c) of Case 2 is analogous to that for Case 1. We will only give a sketched proof in this case. Let i be even. We can obtain $|C_*| = |A_*| \cdot |B|$, where

B is the same as that for C and A_* is obtained from A by replacing the last column with $(c_p^*, c_{p+2}^*, \ldots, c_{2p}^*)^{\mathrm{T}}$. Similar to the analysis given above, we obtain

$$\frac{|C^*_{\nu+1,i}|}{|C_{\nu+1,i}|} = \frac{|A_*||B_{\nu+1,i}|}{|A||B_{\nu+1,i}|} = \frac{|A_*|}{|A|} = \frac{|C_*|}{|C|}.$$

And further, it can also be shown that $|C^*_{\nu+1,i+1}| = |C^*_{\nu+1,i}|$. Hence, Theorem 2.1 is proved. □

The proof of Theorem 2.3 is obtained in the same manner.

Additional Proofs

Proof of Lemma 2.1. The result in 1) holds, since, for j odd, u_t^j for $|u_t| \leq 1$ are asymmetric around u_{i_0} and k_t for $k_t \neq 0$ are symmetric around k_{i_0}.

2) For the difference between c_j^* and c_{j+1}^* we have

$$\begin{aligned} c_j^* - c_{j+1}^* &= \sum_{|u_t|\leq 1} k_t^*[u_t^j - u_t^{j+1}] \\ &= \sum_{|u_t|\leq 1} k_t^*(1-u_t)u_t^j \\ &= \sum_{|u_t|\leq 1} k_t u_t^j = c_j, \end{aligned}$$

since $k_t = k_t^*(1-u_t)$.

3) For j odd we have $c_j^* - c_{j+1}^* = c_j = 0$, i.e. $c_j^* = c_{j+1}^*$.

4) For j even, we have $j+1$ is odd and $c_{j+1}^* = c_{j+2}^*$. Results in 4) follows further from that in 2). With this Lemma 2.1 is proved. □

Proof of Lemma 2.2. 1) Let m' denote the integer part of $(m+1)/2$ and define the permutation matrix P_m with elements p_{ij} by

$$p_{ij} = \begin{cases} 1 & \text{for } j = 2i-1, 1 \leq i \leq m', j = 2(i-m'), m' < i \leq m \\ 0 & \text{otherwise} \end{cases}. \tag{2.7.3}$$

We obtain the matrix $E' = P_m E P_m^T$ with

$$E' = \begin{pmatrix} A & \mathbf{0} \\ \mathbf{0} & B \end{pmatrix}, \tag{2.7.4}$$

where **0** a block of zeros with corresponding rows and columns.[2] Now, A consists purely of elements with odd indices and B of elements with even indices. Hence, A is actually

---

[2] A similar matrix is obtained, if we interchange the i-th row with the $(2i-1)$-th row for $2 \leq i \leq n_1$ and then interchange the j-th column with the $(2j-1)$-th column.

the matrix obtained from E on erasing the even rows and even lines and B is that on erasing the odd rows and odd lines. Since $|P_m|^2 = 1$ is even, we yield

$$|E| = |E'| = \begin{vmatrix} A & \mathbf{0} \\ \mathbf{0} & B \end{vmatrix} = |A| \cdot |B|. \tag{2.7.5}$$

2) By (2.7.3), after the transformation of 1), the r-th row of E consists of the same elements as the i-th row of E, the same is true for columns s of A and j of E. Hence, E_{rs} is the same as E_{ij}, the same is true for the cofactors $\widetilde{e}'_{rs} = \widetilde{e}_{ij}$. Using (2.7.5), the submatrix E_{rs} has the diagonal form of (2.7.5) with blocks A_{rs} and B. The number of total interchanges required for the the transformation of the minor must not necessarily be even and hence, $\widetilde{e}_{rs}$ is either $|A_{rs}||B|$ or $-|A_{rs}||B|$.

3) The proof of 3) is analogous to that of 2).

4) The result in this part is well known for such a special matrix. Let A_1 be an $n_1 \times n_1$ matrix, A_2 be an $n_2 \times n_2$ matrix and A_3 an arbitrary matrix with corresponding rows and columns. Then, note that

$$\begin{vmatrix} A_1 & A_3 \\ \mathbf{0} & A_2 \end{vmatrix} = |A_1||A_2| \text{ and } \begin{vmatrix} A_1 & \mathbf{0} \\ A_3 & A_2 \end{vmatrix} = |A_1||A_2|. \tag{2.7.6}$$

With the same rationale as for 2), erasing the i-th row and and j-th column of E corresponds to erasing the r-th row and s-th column of E'. The erased element is now in one of the **0**'s in (2.7.4). Hence, elements of A and B are erased and we obtain one of the two forms given in (2.7.6) with either $|A_1| = 0$ or $|A_2| = 0$. □

Proof of Theorem 2.2. Note that

$$\mathbf{X}^{\mathrm{T}}\widetilde{\mathbf{X}} = \begin{bmatrix} (1-u_1) & (1-u_2) & \cdots & (1-u_{n-1}) & (1-u_n) \\ (1-u_1)u_1 & (1-u_2)u_2 & \cdots & (1-u_{n-1})u_{n-1} & (1-u_n)u_n \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ (1-u_1)u_1^{p-1} & (1-u_2)u_2^{p-1} & \cdots & (1-u_{n-1})u_{n-1}^{p-1} & (1-u_n)u_n^{p-1} \\ (1-u_1)u_1^{p} & (1-u_2)u_2^{p} & \cdots & (1-u_{n-1})u_{n-1}^{p} & (1-u_n)u_n^{p} \end{bmatrix}.$$

Let p be odd and $\nu < p$ be even. Straightforward calculation shows that the t-th element of $D^{\nu+1}\mathbf{X}^{\mathrm{T}}\tilde{\mathbf{X}}$, denoted by s_t, is given by

$$s_t = (1-u_t) \sum_{j=1}^{(p+1)/2} d_{\nu+1,2j-1} u_t^{2j-2}.$$

Denote the t-th element of $D_*^{\nu+1}\mathbf{X}^{\mathrm{T}}$ by s_t^*. If (2.3.8) holds, we have

$$\begin{aligned} s_t^* &= \sum_{j=1}^{(p+1)/2} \left(d_{\nu+1,2j-1}u_t^{2j-2} - d_{\nu+1,2j-1}u_t^{2j-1}\right) \\ &= \sum_{j=1}^{(p+1)/2} d_{\nu+1,2j-1}u_t^{2(j-1)}(1-u_t) \\ &= s_t. \end{aligned}$$

Hence, the first part of Theorem 2.2 is proved. The second part can be proved analogously.

□

Table 2.1: *Selected right μ-smooth (Müller, 1991) boundary kernels $K_q(u)$ for $u \in [-1, q]$ and $q \in [0, 1]$.*

Order	**Formula**
$(2,2,0)$	$\frac{60(1+u)^2(q-u)^2}{(q+1)^7}\big[(4q^2-6q+4)+(-7q+7)u\big]$
$(3,1,0)$	$\frac{60(1+u)(q-u)}{(q+1)^7}\big[(2q^4-8q^3+15q^2-8q+2)$ $+(-8q^3+27q^2-27q+8)u+(7q^2-21q+7)u^2\big]$
$(3,2,1)$	$\frac{840(1+u)^2(q-u)^2}{(q+1)^9}\big[(-5q^3+16q^2-16q+5)+(22q^2-40q+22)u$ $+(-21q+21)u^2\big]$
$(4,2,0)$	$\frac{840(1+u)^2(q-u)^2}{(q+1)^{11}}\big[(4q^6-30q^5+96q^4-136q^3+96q^2-30q+4)$ $+(-27q^5+171q^4-396q^3+396q^2-171q+27)u$ $+(54q^4-300q^3+480q^2-300q+54x^2)u^2$ $+(-33q^3+165q^2-165q+33)u^3\big]$
$(4,2,2)$	$\frac{5040(1+u)^2(q-u)^2}{(q+1)^{11}}\big[(18q^4-100q^3+160q^2-100q+18)$ $+(-139q^3+455q^2-455q+139)u$ $+(304q^2-580q+304)u^2+(-198q+198)u^3\big]$

Table 2.2: *Selected right (μ, μ')-smooth (Müller and Wang, 1994) boundary kernels $K_q(u)$ for $u \in [-1, q]$ and $q \in [0, 1]$.*

Order	**Formula**
$(2,2,0)$	$\frac{60(1+u)^2(q-u)}{(q+1)^6}\big[(1-2q+2q^2)+(2-3q)u\big]$
$(3,1,0)$	$\frac{12(1+u)}{(q+1)^6}\big[(6q^4-16q^3+21q^2-6q+1)+(-20q^3+45q^2-30q+5)u$ $+(15q^2-30q+5)u^2\big]$
$(3,2,1)$	$\frac{420(1+u)^2(q-u)}{(q+1)^8}\big[(-5q^3+12q^2-9q+2)+(19q^2-26q+11)u$ $+(-16q+12)u^2\big]$
$(4,2,0)$	$\frac{420(1+u)^2(q-u)}{(q+1)^{10}}\big[(5q^6-30q^5+77q^4-84q^3+45q^2-10q+1)$ $+(-30q^5+152q^4-280q^3+216q^2-70q+8)u$ $+(54q^4-240q^3+300q^2-144q+18)u^2$ $+(-30q^3+120q^2-90q+12)u^3\big]$
$(4,2,2)$	$\frac{5040(1+u)^2(q-u)}{(q+1)^{10}}\big[(9q^4-40q^3+50q^2-24q+3)$ $+(-61q^3+160q^2-127q+30)u$ $+(119q^2-182q+77)u^2+(-70q+56)u^3\big]$

3 Fast Computation and Bandwidth Selection Algorithms for the DCS

This chapter is based on joint work with Yuanhua Feng and published with slight differences in the CIE Working Papers (146), Paderborn University, under the title ”Fast Computation and Bandwidth Selection Algorithms for Smoothing Functional Time Series”.

3.1 Introduction

Smoothing, i.e. estimation of the mean surface of curve- or functional time series, is a topic that arises in many research areas, including environmental science, biology, demography, and finance (see e.g. Aneiros-Pérez and Vieu, 2008; Chiou and Müller, 2009; Bathia et al., 2010; Hyndman and Shang, 2010; Shang and Hyndman, 2017; Li et al., 2019a). The mean surface of a functional time series is estimated under a well-known nonparametric regression model (see Hyndman and Ullah, 2007; Aneiros-Pérez and Vieu, 2008; Hyndman and Shang, 2010; Gao and Shang, 2017, among others), assuming the data are observed on a regular lattice. Smoothing of random fields under a fixed design is closely related (see e.g. Machkouri and Stoica, 2010; Wang and Wang, 2009; Yue and Speckman, 2010; Li et al., 2019b). In this chapter we introduce fast computation procedures and develop a suitable data-driven algorithm for estimating the mean surface of large functional time series with millions of observations. Our findings can be generalized to smoothing functional time series or random fields with irregular design in one dimension. Potential extensions include nonparametric regression with spatial- or spatial-temporal data (Hallin et al., 2004; Robinson, 2011) or bivariate kernel regression (Müller, 1988; Müller and Prewitt, 1993; Facer and Müller, 2003) under suitable regularity conditions on the design.

The mean surface of a functional time series can simply be estimated by traditional bivariate (2D) nonparametric regression approaches (see e.g. Müller, 1988; Ruppert and Wand, 1994; Härdle and Müller, 2013; Scott, 2015). However, the common 2D-smoothing techniques run so slowly, up to the point of practical inapplicability, if the number of observations in both dimensions grow very large. This problem becomes even more severe if data-driven selection of the bandwidths is taken into account for which iterative smoothing is required. Some fast computation algorithms for multivariate nonparametric regression were suggested in the literature, including approximate binned kernel estimates (Wand, 1994) or a grid with much fewer estimation points. However, those approaches can only provide approximate or partial smoothing results with certain information loss.

For functional time series defined on a lattice, Feng (2013) and Peitz and Feng (2015) proposed a (Nadaraya-Watson-type) double conditional smoothing (DCS) procedure where univariate kernel estimates are calculated in the first stage, conditioning on observations in the other dimension. Then, the intermediate results are smoothed in the second dimension conditioning on observations in the first dimension. An important feature of the DCS is its equivalency to a 2D-kernel smoother with a product kernel, thus, no loss of information will occur. The main idea of the DCS is to divide a 2D-smoother into two sequential univariate smoothing procedures which hence improves the computational efficiency strongly compared to classical bivariate smoothing. The resulting estimates do not depend on the order of the two univariate smoothing procedures. Moreover, the intermediate smoothing results might also provide valuable information about the trends of the functional time series in one direction which correspond to the smoothing results for a functional time series as used, e.g., by Shang and Hyndman (2017) and Gao and Shang (2017). The first stage of the DCS can be carried out in both dimensions to discover detailed features of the data, examples are given by Feng (2013) and Peitz and Feng (2015). The DCS can be combined with other ideas to further reduce the computation time and might also be extended to higher-dimensional cases. We introduce a new functional DCS (FDCS) scheme to calculate the functional curve at once (per dimension). Again, this procedure is equivalent to the DCS and the 2D-kernel smoother but runs faster than the DCS procedure.

Note that the Nadaraya-Watson (NW) kernel regression is subject to the boundary problem. Spatial observations further worsen the boundary problem, since the ratio of the boundary region to the number of total observations is in general much higher than in the univariate case. In this chapter, we use product boundary kernels, i.e. products of two univariate boundary kernels as proposed by Müller (1991) and Müller and Wang (1994) to correct the boundary effect. For the iterative plug-in (IPI) bandwidth selection rule by Gasser et al. (1991) presented in Section 3.4.2, estimation of derivatives of the regression surface is necessary. Definitions of boundary kernel functions for estimation of these partial derivatives may be found e.g. in Müller (1988) and Facer and Müller (2003). Some useful closed-form formulas of univariate boundary kernels for estimating the derivatives are given in Table 2.1 and Table 2.2 of Chapter 2.

Different data-driven algorithms for bandwidth selection in multivariate kernel regression, including plug-in (Herrmann et al., 1995; Yang and Tschernig, 1999; Koláček and Horová, 2017), Cross-Validation (CV, Zhang et al., 2009; Koláček and Horová, 2017), bootstrap (Manteiga et al., 2004) and Bayesian (Zhang et al., 2009) approaches, are proposed in the literature. We will adjust the 2D-IPI algorithm of Herrmann et al. (1995) for selection of the bandwidths. For this purpose, necessary asymptotic results for the FDCS approach are obtained and investigated under independent and identically distributed (iid.) errors. In particular, the asymptotically optimal bandwidths for the FDCS are shown to be the same as given in the literature since the FDCS is equivalent to the 2D-kernel smoother. An IPI algorithm is developed by plugging suitable estimates of the unknown variance and bias factors into the asymptotically optimal bandwidths starting from fixed initial bandwidths. Bandwidth selection rules based on a search procedure, like CV or bootstrap, run too slowly and are not suitable for selecting the bandwidths in the current context of large data sets and are not considered in this thesis.

The proposed methods are applied to functional time series of high frequency financial (HFF) returns for estimating the spot-volatility surface as well as the surface of trading volumes. Following Andersen and Bollerslev (1997), Andersen and Bollerslev (1998) and Andersen et al. (2000), those data can be indicated by an interday (daily) and an intraday index. We obtain a functional time series with the trading day as the *time* dimension and the intraday trading time as the *temporal* dimension which is a continuous variable indeed. With equidistant intraday observations, a functional time series on a regular lattice received. Exemplary 3D-plots of such data can be found e.g. in Feng (2013); Peitz and Feng (2015), and Li et al. (2019b). The IPI algorithm is applied to one-minute returns of the German companies Siemens AG and BMW AG and the corresponding trading volumes. All time series include more than one million observations over multiple years around the 2008/2009 financial crisis. We estimate the spot-volatility surface from the return data which reflects the joint long-term and intraday volatility dynamics. In particular, it exhibits a volatility saddle around the financial crisis as a combination of the very high volatility peak and the daily volatility smiling (see Figure 3.3a); the volume surface exhibits a similar pattern. The developed methods allow us to estimate and remove a possible non-stationary volatility component from the HFF-returns. The standardized returns can be further analyzed using known parametric functional time series models. The real data examples show that the proposed algorithm works well in practice.

We define nonparametric regression for functional time series and the DCS in Section 3.2. The boundary correction, the FDCS and the estimation of the derivatives are proposed in Section 3.3, asymptotic optimal bandwidths are obtained and applied to the IPI algorithm in Section 3.4. We present our simulation study in Section 3.5 and an application to real data examples in Section 3.6. We close with final remarks in Section 3.7. Additional proofs are given in the appendix.

3.2 The Model and the Basic DCS Procedure

3.2.1 A Semiparametric Functional Time Series Model

Let $y_{i,j}$ be the observations of a functional time series $Y_{i,j}$, obtained on a regular lattice $X_1 \times X_2$ defined by fixed design points $x_{1,i}$, $i = 1, ..., n_1$ in the time dimension and $x_{2,j}$, $j = 1, ..., n_2$ in the temporal dimension, according to given design densities. The total number of observations is $n = n_1 \cdot n_2$. We study the nonparametric regression of a possible deterministic mean surface in those data. For this purpose, we assume the data follow the semiparametric regression model for a functional time series:

$$Y_{i,j} = m(x_{1,i}, x_{2,j}) + \varepsilon_{i,j},\ i = 1, ..., n_1;\ j = 1, ..., n_2, \tag{3.2.1}$$

where $m(\bullet, \bullet)$ is a smooth nonparametric regression function and $\{\varepsilon_{i,j}\}$ is a stationary (possibly parametric) random field defined on the lattice determined by $x_{1,i}$ and $x_{2,j}$ with zero-mean and $\mathrm{var}(\varepsilon_{i,j}) = \sigma^2$. Note that it is only assumed that $x_{1,i}$ or $x_{2,j}$ are fixed design points and their values are taken independently of each other, it is not required that $x_{1,i}$ or $x_{2,j}$ are equally spaced. This model can also be applied to fixed design nonparametric regression with spatial or spatial-temporal data. Throughout this chapter, we assume both variables X_1 and X_2 to be discrete and equidistantly distributed, where we use the rescaled variables $x_{1,i} = i/n_1$ and $x_{2,j} = j/n_2$ on the range $[0, 1] \times [0, 1]$. As specific example for model (3.2.1), we will consider equidistant HFF time series under a functional representation, where X_1 stands for the trading day and X_2 for the intraday trading time. The suggested fast computation procedures can also be extended to the case where x_2 follows an irregular or random design.

As the proposed approaches below do not hinge on the dependency structure of the stationary part, these smoothing procedures are applicable to nonparametric regression for functional time series with iid., short- or long-range dependent errors. Further necessary assumptions on the dependency structure of the lattice process $\{\varepsilon_{i,j}\}$ will be introduced during the discussion on the asymptotic properties of the proposed estimators in Section 3.3.3 and the development of the IPI algorithm in Section 3.4.2.

3.2.2 The Double Conditional Smoothing

Bi- and multivariate kernel regression under model (3.2.1) was studied among others by Ruppert and Wand (1994), Herrmann et al. (1995), Härdle and Müller (2013), and Scott (2015), where also bandwidth selection is covered. A crucial issue in bivariate kernel regression of HFF data is, that these data regularly include a huge number of observations and the common kernel regression estimator runs too slowly. To overcome this problem and for reduction of computation time, the DCS method was proposed by Feng (2013), application examples were given by Peitz and Feng (2015). The DCS provides an equivalent definition to the common bivariate semiparametric regression model:

$$Y_t = m(\widetilde{x}_t) + \varepsilon_t, \quad \widetilde{x}_t = (x_{1,t}, x_{2,t}), \tag{3.2.2}$$

where $m(\widetilde{x})$ is a smooth mean function in $\widetilde{x}$ and $Y_t, t = 1, ..., n$ is a single time series depending on a two-dimensional covariate variable $\widetilde{x}_t = (x_{1,t}, x_{2,t})$. The innovations $\{\varepsilon_t\}$ form a sequence of random variables with zero mean and variance σ^2 which might depend on the point $\widetilde{x}_t$ allowing for heteroscedasticity. The mean or expectation function $m(\widetilde{x}) =$

$\mathbb{E}(Y|\widetilde{x})$ of model (3.2.2) given observations y_t can be estimated using a bivariate kernel regression (see e.g. Müller, 1988; Müller and Prewitt, 1993; Facer and Müller, 2003) with

$$\widehat{m}(x_1, x_2) = \sum_{t=1}^{n} w_t\, y_t,$$
$$w_t = K\left(\frac{x_{1,t}-x_1}{h_1}, \frac{x_{2,t}-x_2}{h_2}\right)\left[\sum_{t=1}^{n} K\left(\frac{x_{1,t}-x_1}{h_1}, \frac{x_{2,t}-x_2}{h_2}\right)\right]^{-1}. \tag{3.2.3}$$

In (3.2.3), $K(u,v)$ is a bivariate kernel function and h_1, h_2 are the bandwidths over X_1 and X_2 respectively. Following Facer and Müller (2003), a bivariate kernel function $K(u,v)$ of order $(k, |\nu|)$, for estimation of a partial derivative $m^{(\nu)}(\widetilde{x}_t)$ with $\nu = (\nu_1, \nu_2)$, has the following properties for $r, s \in \mathbb{N} \cup 0$:

$$\int_{-1}^{1}\int_{-1}^{1} K(u,v) u^r v^s \,\mathrm{d}u\,\mathrm{d}v = \begin{cases} 0 & \text{for } 0 \le r+s \le |\nu|, (r,s) \neq \nu \\ \nu! & \text{for } (r,s) = \nu \\ 0 & \text{for } |\nu| < r+s < k. \end{cases} \tag{3.2.4}$$

Bivariate kernel functions in the sense of (3.2.4) can be formulated in several ways. An important special case is a product kernel, where $K(u,v)$ is the product of two univariate kernels K_1, K_2 of order (k_1, ν_1) and (k_2, ν_2) given in Definition 3.4. These orders are related to that of (3.2.4) by $k = \delta + \nu_1 + \nu_2$, where $\delta \equiv k_1 - \nu_1 = k_2 - \nu_2$. Throughout this chapter, we consider the use of such product kernels as stated in Assumption A5.

Consider the estimation in (3.2.1) with observations $y_{i,j}$ of $Y_{i,j}$, at a point (x_{1,i_0}, x_{2,j_0}), $x_{1,i_0} = i_0/n_1$ and $x_{2,j_0} = j_0/n_2$, where $0 \le i_0 \le n_1$ and $0 \le j_0 \le n_2$ are two integers. We establish the following assumptions for the regression model:

A1. The functional time series under consideration is equidistant with observations $y_{i,j}$, $i = 1, ..., n_1$ and $j = 1, ..., n_2$. Model (3.2.1) is defined as a (two-dimensional) triangular array in $n = n_1 \cdot n_2$ with $m(x_1, x_2)$ on $[0,1]^2$. The equidistant design points $(x_{1,i}, x_{2,j})$ are given by the rescaled variables $x_{1,i} = i/n_1$ and $x_{2,j} = j/n_2$.

A2. The mean surface $m(x_1, x_2)$ is a smooth and Lipschitz continuous function which is at least (k_1, k_2) times continuously differentiable on $[0,1]^2$.

A3. In the limit, the bandwidths satisfy the conditions $h_1, h_2 \to 0$, $n_1 h_1, n_2 h_2 \to \infty$ and $n_1 h_1^{\nu_1} h_2^{\nu_2}, n_2 h_1^{\nu_1} h_2^{\nu_2} \to \infty$, as $n_1, n_2 \to \infty$ at the same time.

A4. The error terms $\{\varepsilon_{i,j}\}$ form an iid. random field with zero mean and common variance $\mathrm{var}(\varepsilon_{i,j}) = \sigma^2$.

A5. A bivariate product kernel $K(u,v) = K_1(u) \cdot K_2(v)$ is used, where K_1, K_2 are (boundary) kernels of order (k_1, ν_1) and (k_2, ν_2) fulfilling the well-known regularity condition in Definition 3.4. For simplification of the results, we further assume that $k_1 - \nu_1 = k_2 - \nu_2 \equiv \delta$, with δ odd.

Although the proposed DCS estimator is with boundary correction, its bias is still affected by the design. Under A1, the formula of the asymptotic bias is simplified. This result does not hold for non-equidistant design, as the kernel estimator used in this chapter is not design adaptive. For non-equidistant fixed design, either the Gasser-Müller estimator (Gasser and Müller, 1979, see (3.2.8)) or local polynomial regression should be adopted. The second part of A1 is a counterpart of a well-known, somehow artificial model assumption in nonparametric regression for time series which assumes that the mean surface is not affected by n_1 or n_2. Without this assumption, we cannot discuss the asymptotic bias and $m(x_1, x_2)$ cannot be estimated consistently. A2 and A3 are common assumptions in bivariate kernel regression. In this chapter, we will only investigate the asymptotic variance under the iid. Assumption A4, the results can be extended to cases with stationary dependent errors. The Regularity conditions on the kernel function of A5 are stated in Definition 3.4 (see e.g. Müller, 1988).

Under the product kernel Assumption A5 and equidistant design of X_1, X_2, the bivariate kernel estimator can be rewritten as

$$\widehat{m}(x_1, x_2) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} w_{1,i}\, w_{2,j}\, y_{i,j}, \tag{3.2.5}$$

with weights[1] defined by

$$\begin{gathered} w_{1,i} = \frac{1}{n_1 h_1 \widehat{f}(x_1)} K_1\left(\frac{x_{1,i} - x_2}{h_1}\right) \quad \text{and} \quad w_{2,j} = \frac{1}{n_2 h_2 \widehat{f}(x_2)} K_2\left(\frac{x_{2,j} - x_2}{h_2}\right), \\ \text{where} \quad \widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x_i - x}{h}\right) \quad \text{for } x_1 \text{ and } x_2. \end{gathered} \tag{3.2.6}$$

The weights in (3.2.7) are of the NW-type. In the equidistant case, we can simplify the weight function by replacing $\widehat{f}$ by the known design density $f = 1$. This leads to the weights proposed by Mack and Müller (1989)

$$w_{1,i} = \frac{1}{n_1 h_1} K_1\left(\frac{x_{1,i} - x_1}{h_1}\right) \quad \text{and} \quad w_{2,j} = \frac{1}{n_2 h_2} K_2\left(\frac{x_{2,j} - x_2}{h_2}\right). \tag{3.2.7}$$

These also allow for straightforward extension to estimation of derivatives as described in Section 3.3.3. Thus, we use the definition (3.2.7) over the NW-type ones from (3.2.6). As mentioned above, both estimators are not undconditionally design adaptive, for non-equidistant or random design densities one should prefer using the weights of Gasser and Müller (1979), with

$$w_{1,i} = \frac{1}{h_1} \int_{r_{i-1}}^{r_i} K_1\left(\frac{u - x_1}{h_1}\right) \mathrm{d}u \quad \text{and} \quad w_{2,j} = \frac{1}{h_2} \int_{s_{j-1}}^{s_j} K_2\left(\frac{v - x_2}{h_2}\right) \mathrm{d}v, \tag{3.2.8}$$

---

[1] The definition of the argument in $K(\bullet)$ differs among the literature and among different estimation methods. We use the notation $u = (x_i - x)/h$ such that values to the left of x get a negative sign in u. While this notation is uncommon in classical kernel regression, it is widely used for local regression and we opt for consistency with respect to local regression.

where $r_1 = x_{1,1}, r_i = (x_{1,i} + x_{1,i-1})/2$ for $2 \leq i \leq n_1 - 1$ and $r_{n_1} = x_{1,n_1}$, s_j is defined analogously for x_2. All three estimators (3.2.6), (3.2.7) and (3.2.8) are asymptotically equivalent under Assumption A1 by the following Theorem 3.1

Theorem 3.1. ***(Equivalency of the Kernel Weights)*** *Let $w^a_{i,j} = w^a_{1,i} w^a_{2,j}$ be the product kernel weights with $w^a_{\bullet,i}$ as defined in* (3.2.7)*, $w^b_{i,j} = w^b_{1,i} w^b_{2,j}$ be the Nadaraya-Watson product weights with $w^b_{\bullet,i}$ from* (3.2.6) *and $w^c_{i,j} = w^c_{1,i} w^c_{2,j}$ be the Gasser-Müller product weights with $w^c_{\bullet,i}$ from* (3.2.8)*, all with the same boundary kernel $K_q(u)$. Under equidistant fixed design of X_1, X_2 on $[0,1]^2$, all estimators are equivalent in the sense that the relation holds*

$$\lim_{n_1,n_2\to\infty} \sup_{\substack{1\leq i\leq n_1\\ 1\leq j\leq n_2}} \left| \frac{w^*_{i,j}}{w^{**}_{i,j}} - 1 \right| = 0 \qquad \text{defining} \qquad \frac{0}{0} \equiv 1,$$

for any $w^*_{i,j}, w^{**}_{i,j} = \{w^a_{i,j}, w^b_{i,j}, w^c_{i,j}\}$.

A proof is given in the appendix to this chapter.

Equation (3.2.5) allows us to write $y_{i,j}$ under a spatial representation on a $n_1 \times n_2$ lattice instead of the single vector of (3.2.2). The idea of the double conditional smoothing is, to not estimate (3.2.5), but to carry out two smoothing procedures over i and j sequentially. This transforms a bivariate kernel smoother into two univariate approaches.

Definition 3.1. ***(Double Conditional Smoothing).*** *Let $y_{i,j}$ be the observations of a functional time series $Y_{i,j}$ observed on an equidistant lattice spanned by $X_1 \times X_2$ and $w_{1,i}, w_{2,j}$ be some appropriate weights. The double conditional smoothing at an observation point (x_1, x_2) is defined by:*

$$\widehat{m}(x_1, x_2) = \sum_{i=1}^{n_1} w_{1,i}\, \widehat{m}(x_2|x_{1,i}) \text{ or equivalently} \tag{3.2.9}$$

$$\widehat{m}(x_1, x_2) = \sum_{j=1}^{n_2} w_{2,j}\, \widehat{m}(x_1|x_{2,j}), \tag{3.2.10}$$

$$\text{with} \quad \widehat{m}(x_2|x_{1,i}) = \sum_{j=1}^{n_2} w_{2,j}\, y_{i,j} \quad \text{and} \quad \widehat{m}(x_1|x_{2,j}) = \sum_{i=1}^{n_1} w_{1,i}\, y_{i,j}. \tag{3.2.11}$$

Both formulas (3.2.9) and (3.2.10) are equivalent, that is, the direction of the double conditional smoothing will not affect the results. Under model (3.2.1) and Assumption A5, they are equivalent to the common bivariate kernel regression. In particular, the double conditional smoothing offers a quick and convenient way to reduce computing effort of bivariate kernel regression and it might also deliver useful intermediate results. The first-stage estimate in (3.2.11) is the smoother which conditions on $x_{1,i}$ (respective $x_{2,j}$) and therefore contains the smoothed time series over each day (respective the time series over the days at a specific intraday time).

The DCS can be carried out under model (3.2.1) at any observation point and is suitable for smoothing a functional time series, provided that the assumptions stated above are fulfilled. This procedure runs much faster than the common bivariate kernel estimator. When the estimation at an interior point is considered, the bivariate regression involves the calculation of a sum with about $4(n_1h_1 \cdot n_2h_2)$ non-zero weights, while the DCS is a sum of $2(n_1h_1 + n_2h_2)$ non-zero weights. The larger the bandwidths are, the larger the difference between the run-times of the two approaches is. However, the DCS is not well-defined at a non-observation point. The DCS can be extended to the case when e.g. X_2 is irregularly spaced or even a random variable, if a design adaptive estimator is used. In this case, smoothing is done over X_2 conditioning on X_1 at first, where suitable fixed estimation points $x_{2,j}$ are chosen independently of $x_{1,i}$. The second stage can then be carried out conditioning on these points in X_2.

3.3 The Improved Double Conditional Smoothing

3.3.1 Boundary Correction Under the DCS

Nonparametric kernel regression suffers from biased estimates at the margins of the definition space, regardless of its dimensionality, the so-called boundary effects. This problem arises, because for the outer observations, there are fewer data to one side used for smoothing than the actual bandwidth would require. This induces a bias in the estimates. The use of specialized boundary kernels was proposed by Gasser and Müller (1979) and later refined by Müller (1991) and Müller and Wang (1994) for different types of kernel estimators. According to their work, we define an interior region and a boundary region on the lattice spanned by X_1 and X_2. Thus, the boundary region contains all observations within an h_1 or h_2 distance from the margins. Define

$$B_1 = \{x_1 : 0 \leq x_1 < h_1 \cup 1 - h_1 < x_1 \leq 1\},$$
$$B_2 = \{x_2 : 0 \leq x_2 < h_2 \cup 1 - h_2 < x_2 \leq 1\},$$

where B_1 defines the boundary range in the x_1-direction and B_2 the same for x_2. The boundary region (BR) of the kernel regression under consideration is given by $BR = \{B_1 \cup B_2\}$ and the interior region is defined as $IR = [0,1]^2 \setminus \{B_1 \cup B_2\}$. In the current context, the ratio BR/IR might be very large. Compared to the boundary region of a one-dimensional estimator, which is $2h$, the boundary region for the spatial model under consideration has size $4h - 4h^2$. For an NW-type estimator, the bias of an estimate in the BR is of a lower order of magnitude in terms of the bandwidths and the mean integrated squared error (MISE) will be dominated by estimates in the BR. Thus, the correction of the estimates in the BR is necessary. Following Müller (1988), the product of two univariate boundary kernels forms a bivariate boundary kernel function and the boundary correction in the DCS reduces to the use of corresponding univariate boundary kernels in (3.2.9) - (3.2.11). For $q_1 \in [0,1]$ a right boundary kernel $K_{q_1}(u)$ has support $[-1, q_1]$, where q_1 is $(1 - x_1)/h_1$, provided that $x_1 \in B_1$. The corresponding kernel to the left is $K_{q_1,L}(u) = K_{q_1}(-u)$ on $[-q_1, 1]$, $q_1 = x_1/h_1$ for $x_1 \in B_1$. For $q_1 = 1$, the boundary kernels reduce to the interior kernel $K(u)$. Similar definitions hold for x_2. Throughout this chapter, we will use the $(\mu, \mu - 1)$-smooth boundary kernels of Müller and Wang (1994) in the form given in Chapter 2 rather than the μ-optimal kernels by Müller (1991), as the latter are not centered around the point of estimation and thus have a larger bias. In the following, all kernels K_1, K_2 are assumed to be boundary kernels of the correct side without explicit labels, whenever $(x_1, x_2) \in BR$.

3.3.2 A Functional Smoothing Scheme

The previously defined double conditional smoothing can still be improved in efficiency to a faster functional smoothing scheme. This scheme provides a way to avoid redundant computations and reduces the runtime of an implementation of the algorithm further. We assume that the weights for smoothing in the x_1-dimension do not depend on the given value of x_2, and vice versa. Then, the DCS procedure can be written in matrix form, where $\mathbf{Y}$ explicitly denotes the matrix of observations with components $y_{i,j}$ and $\mathbf{W}_1, \mathbf{W}_2$ are the smoothing matrices containing the (column-) vectors of weights for each column or row from $\mathbf{Y}$. That is, $\{\mathbf{W}_1\}_{i,1\leq j\leq n_2} = w_1(i)^T$ and $\{\mathbf{W}_2\}_{1\leq i\leq n_1,j} = w_2(j)$, where $w_1(i_0)$ is the vector of weights obtained from (3.2.6), (3.2.7) or (3.2.8) for estimation at a point $x_1 = x_{1,i_0}$ with elements $w_{1,i}$. The vector $w_2(j)$ is defined analogously. From Definition 3.1 we can directly derive the matrix DCS (MDCS):

Definition 3.2. ***(Matrix Double Conditional Smoothing).*** *Let $\mathbf{Y}$ be the $n_1 \times n_2$ matrix of ordered observations of a time series $Y_{i,j}$ on an equidistant lattice spanned by $X_1 \times X_2$ and $\mathbf{W}_1, \mathbf{W}_2$ be some appropriate weighting matrices. Then, the MDCS is given by the equations*

$$\widehat{\mathbf{M}} = \mathbf{W}_1 \cdot \widehat{\mathbf{M}}_{x_2|x_1} \text{ or equivalently}$$
$$\widehat{\mathbf{M}} = \widehat{\mathbf{M}}_{x_1|x_2} \cdot \mathbf{W}_2,$$
$$\text{with} \quad \widehat{\mathbf{M}}_{x_2|x_1} = \mathbf{Y}\,\mathbf{W}_2 \quad \text{and} \quad \widehat{\mathbf{M}}_{x_1|x_2} = \mathbf{W}_1\,\mathbf{Y}.$$

The double conditional smoothing can also be represented in a single step estimator using matrix notation which is equivalent to (3.2.5):

$$\widehat{\mathbf{M}} = \mathbf{W}_1\,\mathbf{Y}\,\mathbf{W}_2.$$

Although the matrix smoothing scheme provides an elegant definition from a theoretical point of view, the calculation of the product of two huge matrices can cause computational problems. We propose to divide the matrix product into corresponding products of vectors of weights and the data matrix. We obtain a smoothed curve (or function) over all (conditional) $x_{2,j}$ values for given $x_{1,i}$ and vice versa. This idea can easily be implemented into a computer aided algorithm and will be called a *functional DCS smoothing scheme* (FDCS):

Definition 3.3. ***(Functional Double Conditional Smoothing).****Let $\mathbf{Y}$ be the $n_1 \times n_2$ matrix of ordered observations of a time series $Y_{i,j}$ on an equidistant lattice spanned by $X_1 \times X_2$ and $w_1(i), w_2(j)$ be some appropriate vectors of weights. Then, the FDCS calculates each row respective column of the resulting matrices at once by the equations*

$$\begin{aligned} \left\{\widehat{\mathbf{M}}\right\}_{i,1\le j\le n_2} &= w_1(i)^T\widehat{\mathbf{M}}_{x_2|x_1} \text{ or equivalently} \\ \left\{\widehat{\mathbf{M}}\right\}_{1\le i\le n_1,j} &= \widehat{\mathbf{M}}_{x_1|x_2}w_2(j) \\ \text{with} \quad \left\{\widehat{\mathbf{M}}_{x_2|x_1}\right\}_{1\le i\le n_1,j} &= \mathbf{Y}w_2(j) \quad \text{and} \quad \left\{\widehat{\mathbf{M}}_{x_1|x_2}\right\}_{i,1\le j\le n_2} = w_1(i)^T\mathbf{Y}. \end{aligned}$$

Both procedures are carried out over all i and j, respectively. Note that the DCS, MDCS and FDCS are three equivalent computation schemes with different implementation methods in a program. Hence, the resulting estimates are all exactly the same and indeed the same as those obtained by the common 2D-kernel regression under the regularity conditions used in this chapter.

3.3.3 Estimation of Derivatives

The formulas for the optimal bandwidths h_1 and h_2 for the double conditional smoothing of the regression surface in Proposition 3.1 include partial derivatives of the surface $m(x_{1,i}, x_{2,j})$. In the IPI algorithm, these derivatives need to be calculated explicitly. In addition, the estimation of the partial derivatives itself is an important topic in theory and practice of nonparametric regression and of particular interest itself. Note that the boundary problem is even more severe when estimation of the partial derivatives is considered, as this usually requires larger bandwidths than those used in the regression surface estimation. Boundary kernels for estimating the derivatives in univariate kernel regression are well studied in the literature, e.g. by Gasser et al. (1985); Müller (1988); Müller (1991); Müller and Wang (1994) and Feng (2004). We denote[2] such a kernel (with suitable extension to the boundary points[3]) by $K_\nu(u)$. Then $K_{(\nu_1,\nu_2)}(u,v) = K_{\nu_1}(u) \cdot K_{\nu_2}(v)$ defines a product kernel for estimation of the (ν_1,ν_2)-th derivative of the expectation surface $m(x_1,x_2)$. The corresponding kernel estimator of Mack and Müller (1989) is defined by

$$\hat{m}^{(\nu_1,\nu_2)}(x_1,x_2) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} w_{i,j}^{\nu_1,\nu_2} y_{i,j}, \tag{3.3.1}$$

---

[2] In the remainder of this chapter, all kernel functions are assumed to be of the desired derivative without explicit indication.

[3] Note that, if $K_\nu(u)$ is a right boundary kernel, the corresponding kernel at the left is $K_{\nu,L}(u) = (-1)^\nu K_\nu(-u)$.

where $w_{i,j}^{\nu_1,\nu_2} = w_{1,i}^{\nu_1} w_{2,j}^{\nu_2}$ with

$$w_{1,i}^{\nu_1} = \frac{1}{n_1 h_1^{\nu_1+1}} K_{\nu_1}\left(\frac{x_{1,i}-x_1}{h_1}\right) \text{ and } w_{2,j}^{\nu_2} = \frac{1}{n_2 h_2^{\nu_2+1}} K_{\nu_2}\left(\frac{x_{2,j}-x_2}{h_2}\right), \tag{3.3.2}$$

where h_1 and h_2 are suitably chosen bandwidths for estimating the partial derivative of given orders (see Section 3.4.2). The partial derivative surface $\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)$ can also be calculated by means of the DCS or FDCS schemes proposed in this chapter. Useful explicit formulas of $K_\nu(u)$ are obtained in Chapter 2 based on the results of Feng (2004) which will be used for the practical implementation of the proposed IPI algorithm in Section 3.4.2.

3.3.4 Asymptotic Behavior of the Estimator

Let the kernels in the weights (3.3.2) for the derivative estimator (3.3.1) be of order (k_1,ν_1) for K_1 and (k_2,ν_2) for K_2. Then, under Assumptions A1 to A4, the expectation of the estimator in (3.3.1) (and asymptotically of that in (3.2.5), (3.2.7)) at an interior point is given by

$$\begin{aligned}\mathbb{E}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} &= m^{(\nu_1,\nu_2)}(x_1,x_2) + B_m(x_1,x_2)[1+o(1)] \\ &\quad + O\left(\frac{1}{n_2 h_1^{\nu_1} h_2^{\nu_2}}\right) + O\left(\frac{1}{n_1 h_1^{\nu_1} h_2^{\nu_2}}\right),\end{aligned} \tag{3.3.3}$$

where, using the kernel constants $b_i = \int K_i(u)u^k \,\mathrm{d}u$ (see Definition 3.4 in Section 3.8),

$$B_m(x_1,x_2) = \frac{b_1}{k_1!} h_1^{k_1-\nu_1} m^{(k_1,\nu_2)}(x_1,x_2) + \frac{b_2}{k_2!} h_2^{k_2-\nu_2} m^{(\nu_1,k_2)}(x_1,x_2). \tag{3.3.4}$$

The boundary correction ensures, that the order of magnitude of the bias at a boundary point is the same as that of B_m. The variance is

$$\operatorname{var}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} = \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} \left[R(K_1)R(K_2) + o(1)\right], \tag{3.3.5}$$

with $R(K) = \int K^2(u)\,\mathrm{d}u$. Proofs of (3.3.3) and (3.3.5) can be found in Section 3.8.2, these results are similar to the findings of Müller and Prewitt (1993) and Facer and Müller (2003) only for different weighting methods. In Section 3.8.3 it is shown that the estimator is asymptotically normal distributed with:

$$\sqrt{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} \left(\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) - m^{(\nu_1,\nu_2)}(x_1,x_2)\right) \xrightarrow{d} N(0,\sigma^2).$$

3.4 Bandwidth Selection

3.4.1 Asymptotic Optimal Bandwidths

In nonparametric regression, commonly used measures of the goodness-of-fit are the mean squared error (MSE) and the MISE. Following Herrmann et al. (1995), we will call the minimizers of the asymptotic MISE (AMISE) the optimal bandwidths. Under Assumptions A1-A5, the MISE (and the AMISE) of the regression surface ($\nu_1 = \nu_2 = 0$) at an interior point (x_1, x_2) is given by

$$\begin{aligned}\text{MISE } &= \text{AMISE}\\ &+ o\left(h_1^{2k_1} + h_2^{2k_2} + n_1^{-1}n_2^{-1}h_1^{-1}h_2^{-1}\right)\\ &+ O\left(n_1^{-1}\right) + O\left(n_2^{-1}\right)\end{aligned}$$

For simplification, we set $K_1(u) = K_2(u) \equiv K(u)$ and assume $K(u)$ is a kernel of order $k = 2$, a generalization is in Section 3.8.2. As mentioned above, we assume that corresponding boundary kernels for K_1, K_2 are used at a boundary point. Thus, the orders of magnitude of the bias at a boundary point are ensured to be the same as in the interior. Hence, the AMISE is

$$\begin{aligned}\text{AMISE } &= \frac{b^2}{4}\left[h_1^4 \cdot I_{11} + 2h_1^2h_2^2 \cdot I_{12} + h_2^4 \cdot I_{22}\right]\\ &+ \frac{R(K)^2 \cdot \sigma^2}{n_1n_2h_1h_2}\\ &+ o(h_1^4 + h_2^4) + o\left((n_1n_2h_1h_2)^{-1}\right),\end{aligned} \tag{3.4.1}$$

with the integrals

$$\begin{aligned}I_{11} &= \int_0^1\int_0^1 \left[m^{(2,0)}(x_1,x_2)\right]^2 \mathrm{d}x_1\,\mathrm{d}x_2,\\ I_{22} &= \int_0^1\int_0^1 \left[m^{(0,2)}(x_1,x_2)\right]^2 \mathrm{d}x_1\,\mathrm{d}x_2,\\ \text{and}\quad I_{12} = I_{21} &= \int_0^1\int_0^1 m^{(2,0)}(x_1,x_2)\,m^{(0,2)}(x_1,x_2)\,\mathrm{d}x_1\,\mathrm{d}x_2.\end{aligned} \tag{3.4.2}$$

Note in particular, that the MISE is calculated on the complete support $[0,1]^2$. The boundary correction ensures that the effect of the estimates in the boundary region is asymptotically negligible under A3. This is not true for the NW-type estimator without boundary correction.

Proposition 3.1. ***(Asymptotic Optimal Bandwidths).*** *Let* $K(u)$ *be a kernel of order* $(k = 2, \nu = 0)$ *with boundary correction. Under Assumptions A1 to A5, the asymptotically optimal bandwidths for estimation of the regression surface, which minimize the AMISE* (3.4.1)*, are given by*

$$h_{1,A} = \left(\frac{R(K)^2 \cdot \sigma^2}{n_1 n_2 b^2 \left[\left(\frac{I_{11}}{I_{22}}\right)^{\frac{3}{4}} \left(\sqrt{I_{11} I_{22}} + I_{12}\right) \right]} \right)^{\frac{1}{6}} \quad \textit{and} \tag{3.4.3}$$

$$h_{2,A} = \left(\frac{R(K)^2 \cdot \sigma^2}{n_1 n_2 b^2 \left[\left(\frac{I_{22}}{I_{11}}\right)^{\frac{3}{4}} \left(\sqrt{I_{11} I_{22}} + I_{21}\right) \right]} \right)^{\frac{1}{6}} . \tag{3.4.4}$$

The two asymptotically optimal bandwidths are related by:

$$h_{1,A} = h_{2,A} \cdot \left(\frac{I_{22}}{I_{11}}\right)^{\frac{1}{4}} \quad \textit{and} \quad h_{2,A} = h_{1,A} \cdot \left(\frac{I_{11}}{I_{22}}\right)^{\frac{1}{4}} .$$

General optimal bandwidth formulas for estimation of partial derivatives and different kernel orders can be found in Section 3.8.4.

3.4.2 The IPI Algorithm

The selection of the optimal bandwidths based on (3.4.3) and (3.4.4) involves the development of suitable estimates of the integrals and the innovation variance required in those formulas. We propose to estimate σ^2 simply from the residuals. The above integrals are simply estimated by summation from

$$\widehat{I}_{11} = \frac{1}{n_1^0 n_2^0} \sum_{i=n_1^o+1}^{n_1-n_1^o} \sum_{j=n_2^o+1}^{n-n_2^o} \left[\widehat{m}^{(2,0)}(x_{1,i}, x_{2,j}) \right]^2 , \tag{3.4.5}$$

$$\widehat{I}_{22} = \frac{1}{n_1^0 n_2^0} \sum_{i=n_1^o+1}^{n_1-n_1^o} \sum_{j=n_2^o+1}^{n_2-n_2^o} \left[\widehat{m}^{(0,2)}(x_{1,i}, x_{2,j}) \right]^2 ,$$

$$\widehat{I}_{12} = \widehat{I}_{21} = \frac{1}{n_1^0 n_2^0} \sum_{i=n_1^o+1}^{n_1-n_1^o} \sum_{j=n_2^o+1}^{n_2-n_2^o} \widehat{m}^{(2,0)}(x_{1,i}, x_{2,j})\, \widehat{m}^{(0,2)}(x_{1,i}, x_{2,j}), \tag{3.4.6}$$

where $\widehat{m}^{(\nu_1,\nu_2)}$ is estimated following (3.3.1) using bandwidths $\widetilde{h}_1$ and $\widetilde{h}_2$ which, in general, differ from the bandwidths h_1, h_2 for estimation of the regression surface. The IPI algorithm uses a subset of the observations with $n_1^o = [\lambda_1 n_1]$, $n_2^o = [\lambda_1 n_2]$, where λ_1, λ_2 are zero or a small positive number and $[\bullet]$ denotes the integer part. For most applications, $\lambda_1 = \lambda_2 \equiv \lambda$ works well. Although the proposed estimators are with boundary correction, the estimation of $\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)$ near the endpoints might be unstable (see examples in

Figure 3.2). In order to calculate the integrals, the estimates at some boundary points can be excluded if this improves the stability of the selected bandwidths. Then, only estimated values of the partial derivatives within $[\lambda_1, 1-\lambda_1] \times [\lambda_2, 1-\lambda_2]$ are used for calculation of the integrals.

A crucial problem is the choice of the bandwidths $\widetilde{h}_2$ and $\widetilde{h}_1$ used in derivative estimation. The IPI algorithm proposed by Gasser et al. (1991) computes the auxiliary bandwidths from h_1 and h_2 selected in each iteration step, by means of some so-called *inflation method*. The purpose is to achieve h_1 and h_2 with a certain optimal property at the end of the procedure. The IPI algorithm has been extended to correlated data by Herrmann et al. (1992), to long-range dependencies by Ray and Tsay (1997) and Beran and Feng (2002a), and to bivariate data by Herrmann et al. (1995).

The inflation method computes the $\widetilde{h}$ from the h by specifying a functional relation between these two. The mainly used method is the multiplicative inflation method (MIM) which was introduced by Gasser et al. (1991) and was also employed by Herrmann et al. (1995). The MIM links the bandwidths by a multiplicative relation, that is, in the s-th iteration step $\widetilde{h}^{(\nu)}_{1,s} = c_{(\nu)} \cdot h_{1,s-1} n^{\alpha_1}$. The bandwidth $\widetilde{h}^{(\nu)}_{1,s}$ is suited for estmation of the ν-th derivative over X_1 and $c_{(\nu)}$ is a scaling factor allowing for tuning the bandwidths conditional on the order of the derivative. Another approach is the exponential inflation method (EIM) used by Beran and Feng (2002a,b). In contrast to the MIM, the EIM uses an exponential relation to compute the auxiliary bandwidths: $\widetilde{h}^{(\nu)}_{1,s} = c_{(\nu)} \cdot h^{\alpha_1}_{1,s-1}$. This may result in a faster speed of convergence and therefore to less iterations required for the IPI (see Beran and Feng, 2002b). The scaling parameter used by Beran and Feng (2002a) was $c = 1$, while Herrmann et al. (1995) set $c_{(2)} = 1.5$ for the bandwidth in the direction of the second derivative and $c_{(0)} = 0.25$ in the other. We found in our simulations that a choice of $c_{(2)} = 2, c_{(0)} = 1$ fits the best for estimation of the regression surface, but optimal values might depend on the derivative order (ν_1, ν_2) under consideration, the form of regression function itself or the error term variance. Two ways have been widely used to obtain optimal exponents α_1: Minimize $\mathrm{MISE}\,(\widehat{I}_{11})$ or minimize $\mathrm{MISE}\,(\widehat{m}^{(2,0)})$. Bandwidths for derivatives in x_2 direction, i.e. for calculation of $\widehat{m}^{(0,2)}$ are obtained in the same manner. From $k_1 - \nu_1 = k_2 - \nu_2 = \delta$ in Assumption A5, it follows that $\alpha_1 = \alpha_2 \equiv \alpha$. As Beran and Feng (2002a) pointed out, $\alpha = 1/2$ is the most stable choice for α for the EIM which corresponds to the use of $\alpha = 1/12$ for the MIM as proposed by Herrmann et al. (1995). Note that both of $h_{1,A}$ and $h_{2,A}$ are of the order $O(n^{-1/6})$. The idea is to inflate the bandwidths for estimation of $\widehat{I}_{11}$, $\widehat{I}_{22}$, and $\widehat{I}_{12}$ to those of the order $O(n^{-1/12})$, so that the variances of both estimators will achieve the lowest rate of convergence $O(n^{-1/2})$ (Herrmann et al., 1995). Thus, we will use this choice of α in the current chapter, as the stability of the selected bandwidths plays an important role in practice, in particular in two-dimensional kernel smoothing. The other two choices of α will not be considered in the current chapter, because they may result in much smaller selected bandwidths. Initial values $(h_{1,0}, h_{2,0})$ of the algorithm can be chosen quite arbitrarily, as the results do not depend on these starting values.

Proposition 3.2. ***(IPI Bandwidth Selection Algorithm).*** *Let* $(\widehat{h}_{1,s}, \widehat{h}_{2,s})$ *be the bandwidths obtained in the* s*-th iteration step. Then, the IPI algorithm processes as follows:*

1. *Choose initial values* $(\widehat{h}_{1,0}, \widehat{h}_{2,0})$.
2. *In the* s*-th iteration step:*
   i. *Define*

$$\begin{aligned}\widetilde{h}_{1,s}^{(2)} &= c_{(2)}(\widehat{h}_{1,s-1})^{\alpha}, \quad \widetilde{h}_{1,s}^{(0)} = c_{(0)}(\widehat{h}_{1,s-1})^{\alpha},\\ \widetilde{h}_{2,s}^{(2)} &= c_{(0)}(\widehat{h}_{2,s-1})^{\alpha}, \quad \widetilde{h}_{2,s}^{(0)} = c_{(2)}(\widehat{h}_{2,s-1})^{\alpha}.\end{aligned}$$

   ii. *Compute* $\widehat{m}^{(2,0)}(x_1, x_2)$ *using bandwidths* $\left(\widetilde{h}_{1,s}^{(2)}, \widetilde{h}_{2,s}^{(0)}\right)$ *and* $\widehat{m}^{(0,2)}(x_1, x_2)$ *using bandwidths* $\left(\widetilde{h}_{1,s}^{(0)}, \widetilde{h}_{2,s}^{(2)}\right)$.

   iii. *Compute the corresponding integrals* $\widehat{I}_{11}, \widehat{I}_{22}, \widehat{I}_{12}$ *applying* (3.4.5) - (3.4.6) *and the optimal bandwidths* $\left(\widehat{h}_{1,s}, \widehat{h}_{2,s}\right)$ *by* (3.4.3), (3.4.4).
3. *Stop if the distance between* $(\widehat{h}_{1,s}, \widehat{h}_{2,s})$ *and* $(\widehat{h}_{1,s-1}, \widehat{h}_{2,s-1})$ *is smaller than some desired threshold. Otherwise, return to 2.*

The bandwidths yielded in step 3 *are then called the asymptotic optimal bandwidths.*

Calculation of bandwidths via Proposition 3.1 requires computation or estimation of additional values. The kernel constants $R(K)$ and b can be computed straightforward either by analytic integration of the kernel or by numerical approximation. The estimation of the variance σ^2 is carried out in each iteration based on the residuals

$$\widehat{\sigma}_s^2 = \frac{1}{n}\sum_{i=1}^{n_1}\sum_{i=1}^{n_2}\left[Y_{i,j} - \widehat{m}_s(x_{1,i}, x_{2,j})\right]^2,$$

where $\widehat{m}_s(x_{1,i}, x_{2,j})$ is calculated in each iteration using the bandwidths $(\widehat{h}_{1,s-1}, \widehat{h}_{2,s-1})$ from the previous iteration step. An alternative estimator of σ^2 is given by Herrmann et al. (1995) based on the differences, which provides an initial estimation before the iteration process, but is presumably more imprecise.

It can be shown that $\widehat{\sigma}^2$ is $\sqrt{n}$-consistent. Following the results in Herrmann et al. (1995), it can be shown that the rates of convergence of the selected bandwidths are of the order $O(n^{-1/6})$ for MIM and EIM inflation methods.

3.5 Finite Sample Simulations

The performance and precision of the IPI bandwidth selector derived in Section 3.4.2 is assessed via a simulation study using two Gaussian peaks as exemplary functions for the mean surface $m(x_1, x_2)$:

$$m_1(x_1, x_2) \sim N\left(\begin{bmatrix}0.5\\0.5\end{bmatrix}, \begin{bmatrix}0.05 & 0\\0 & 0.05\end{bmatrix}\right) + \sigma\,\eta, \tag{3.5.1}$$

$$m_2(x_1, x_2) \sim N\left(\begin{bmatrix}0.5\\0.3\end{bmatrix}, \begin{bmatrix}0.1 & 0\\0 & 0.1\end{bmatrix}\right) + N\left(\begin{bmatrix}0.2\\0.8\end{bmatrix}, \begin{bmatrix}0.05 & 0\\0 & 0.05\end{bmatrix}\right) + \sigma\,\eta. \tag{3.5.2}$$

N denotes the bivariate normal distribution, η is an iid. random field following a standard normal distribution and σ^2 is a variance. Hence, m_1 represents a symmetric single peak and m_2 is an asymmetric double peak function. We employ 10,000 simulations for each function and variances $\sigma_a^2 = 1, \sigma_b^2 = 0.25$ on $m : [0,1] \times [0,1] \to \mathbb{R}$ with $n_1 = n_2 = 101$. The second partial derivatives for both functions can be calculated analytically and thus, the true optimal (MISE minimizing) bandwidths $(h_{1,true}, h_{2,true})$ are known.

For estimation of the regression surface, we use kernels of order $(2, 2, 0)$ whenever $\nu_i = 0$ and a $(4, 2, 2)$ kernel if the 2nd derivative is considered. All kernels are of the (μ, μ')-smooth type by Müller and Wang (1994). We set the parameters in the IPI algorithm to $c_{(2)} = 2, c_{(0)} = 1, \alpha = 0.5$, the margins are trimmed by $\lambda = 0.05$.

Figure 3.1 displays the distributions of the estimated bandwidths. The overall precision of the bandwidth selector depends on the functional form of the surface $m(x_1, x_2)$, the variance of the errors and also on the choice of the inflation parameters for h_1, h_2 discussed in Section 3.4.2. Hence, the slight over- or undersmoothing displayed in the histograms is due to the functional form chosen for m_1, m_2 and the choice of parameters but not an artifact of the bandwidth selection estimator itself. Other choices of test functions might feature the same or contrary behaviour. The p-values to the null hypothesis $H_0 : h = h_{true}$ under a normal distribution are given in Table 3.1, along with the means, true values, and standard deviations. Despite the use of boundary kernels to reduce the bias, the estimation at the margins is still unstable to some degree as illustrated in the smoothed examples of Figure 3.2.

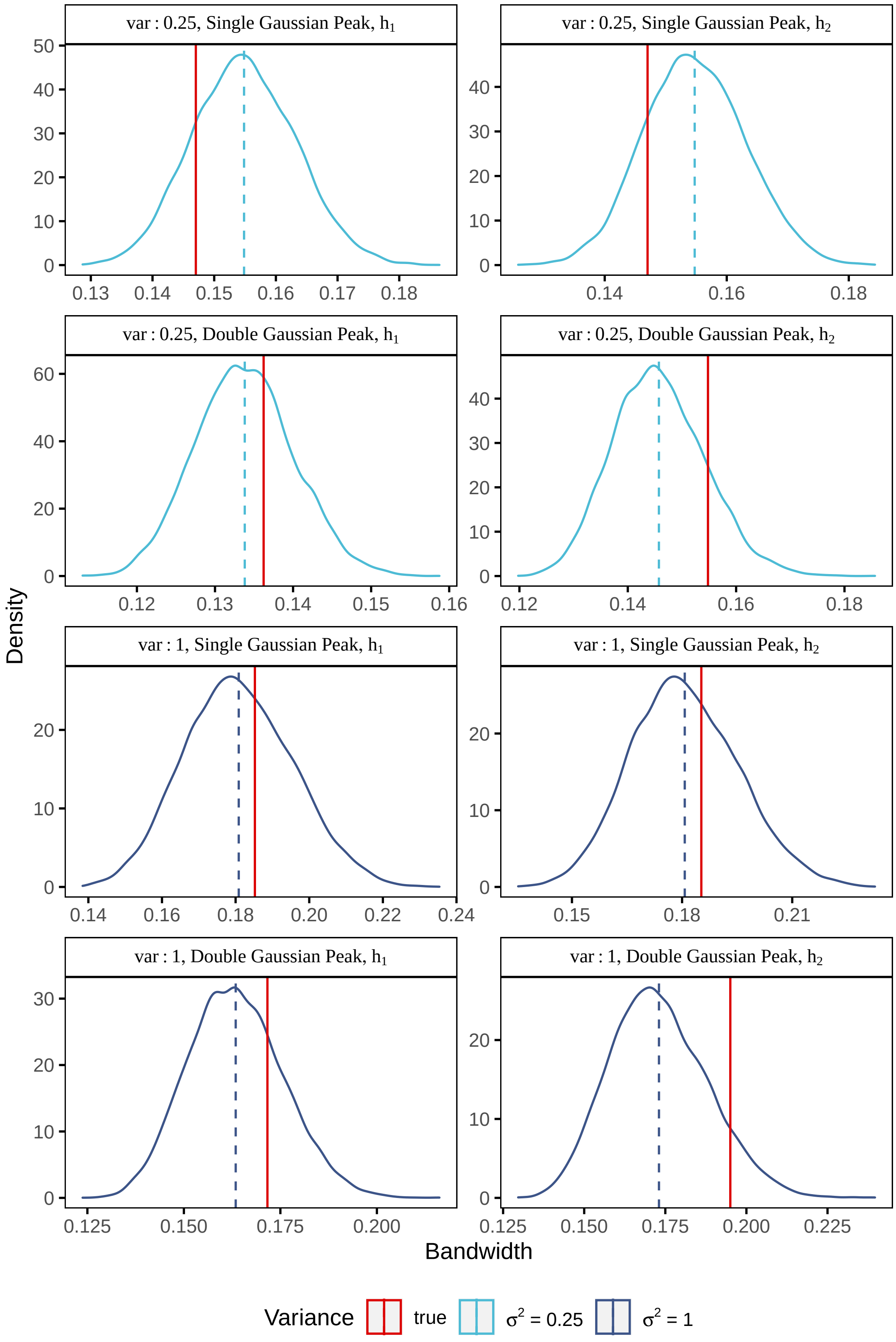


Figure 3.1: *Distribution of the bandwidth estimates* h_1, h_2 *obtained in the simulation study. Simulated are functions* m_1, m_2 *from* (3.5.1), (3.5.2) *under iid. standard normal errors for* σ_a^2 *and* σ_b^2.

Table 3.1: *Bandwidth statistics of the simulation study under iid. errorterms using the kernel regression DCS. Values are obtained from a sample of 10,000 simulations for functions* m_1*,* m_2 *from* (3.5.1)*,* (3.5.2)*. The p-value corresponds to* $H_0 : h = h_{true}$*.*

	Function		**h$_{t}$rue**	**mean(h)**	**sd(h)**	**p-value**
$\sigma^2 = 0.25$	f_1	h_1	0.1470	0.1548	0.0083	0.2566
		h_2	0.1470	0.1547	0.0083	0.2578
	f_2	h_1	0.1362	0.1338	0.0062	0.3701
		h_2	0.1548	0.1457	0.0085	0.2264
$\sigma^2 = 1$	f_1	h_1	0.1852	0.1808	0.0147	0.3815
		h_2	0.1852	0.1807	0.0146	0.3803
	f_2	h_1	0.1716	0.1634	0.0122	0.3186
		h_2	0.1950	0.1730	0.0150	0.1366

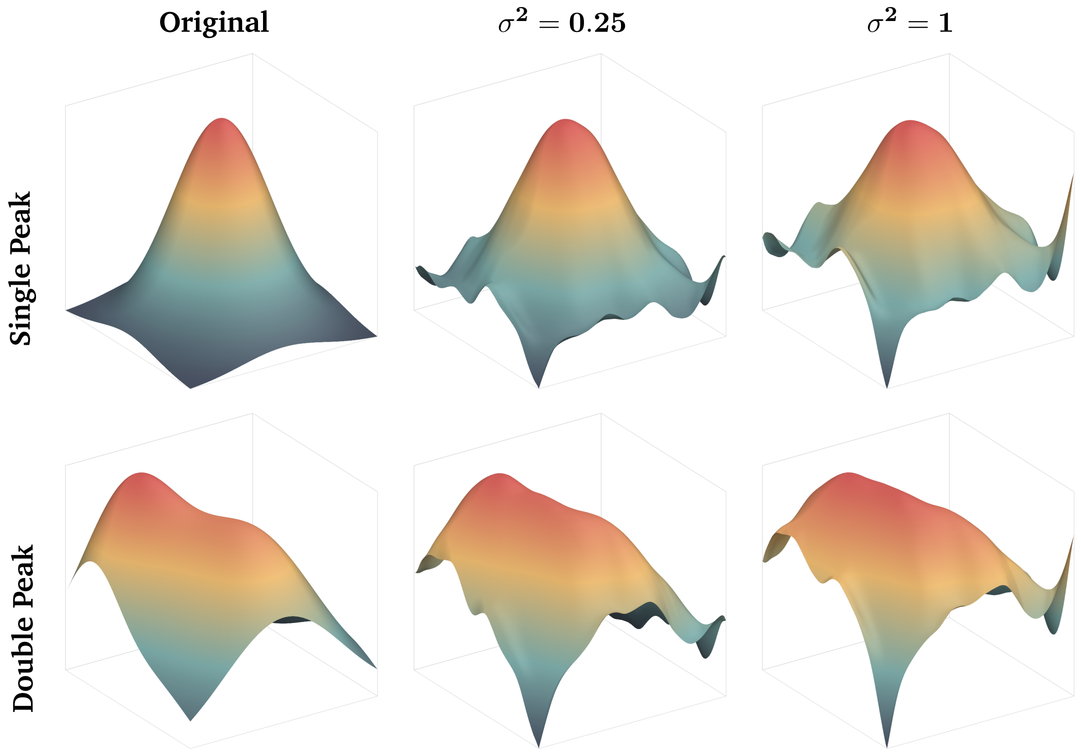


Figure 3.2: *Simulated and estimated surfaces for the functions* m_1*,* m_2 *from* (3.5.1)*,* (3.5.2) *under iid. standard normal errors* η *and* $\sigma^2 = 0.25$*,* $\sigma^2 = 1$*. Used bandwidths are:* $(0.1382, 0.1612)$ *for* m_1 *and* σ_1^2*,* $(0.1512, 0.1950)$ *for* m_1 *and* σ_b^2*,* $(0.1415, 0.1521)$ *for* m_2 *and* σ_a^2 *and* $(0.1654, 0.1922)$ *for* m_2 *and* σ_b^2*. Scale of the vertical axis might differ across the plots.*

3.6 Application to Financial Data

In this section, the previously developed bandwidth selector is applied to HFF time series data. We compute the stock price volatility surface as well as the trading volume surface for the stocks of German companies Siemens AG (SIE) and BMW AG (BMW). The data consist of the 1-minute aggregated stock prices and trading volumes from 2004-01-02 to 2014-09-30 with about 1.39 million observations[4]. The trading volumes are directly applied to the model (3.2.1). The spot-volatility surface is computed from the demeaned log-returns $\overline{R}_{i,j}$ of the functional time series for the price data. Models for simultaneously analysis of interday effects (e.g. ARCH or GARCH effects) and intraday volatility were proposed by Andersen and Bollerslev (1998); Feng (2013) and Peitz and Feng (2015). We employ a simplified version of their models, where both, the ARCH effects and intraday volatility, are captured in the nonparametric volatility surface $\sigma_R(x_1, x_2)$. The nonparametric regression model is then

$$\overline{R}_{i,j} = \sigma_R(x_{1,i}, x_{2,i})\eta_{i,j},$$

where $\eta_{i,j}$ an iid. random field following a standard normal distribution. The volatility surface is estimated from

$$\ln\left(\overline{R}^2_{i,j}\right) = \ln\left(\sigma^2_R(x_{1,i}, x_{2,j})\right) + \varepsilon_{i,j}. \tag{3.6.1}$$

Note that $\varepsilon_{i,j}$ is an iid. random field with zero mean since $\mathbb{E}\{\ln(\eta^2_{i,j})\} = 0$.

As expected, the smoothed volatility and volume plots in figures 3.3a and 3.3b clearly show the influence of the 2008 financial crisis as a large peak and the 2012 euro currency crisis as the minor peak for both surfaces. The intraday time series exhibit the typical U-shape volatility smile which is the pattern suggested by economic theory (see e.g. Lockwood and Linn, 1990; Andersen and Bollerslev, 1997; Goodhart and O'Hara, 1997).

For selecting the optimal bandwidths, we use the same setup as in Section 3.5, the numerical results obtained via the IPI bandwidth selection are given in Table 3.2. By comparing the resulting bandwidths a clear pattern arises: the interday (x_1) and intraday (x_2) bandwidths of both companies are close to each other for the volatility, the same is true for the volume bandwidths. On the other hand, the results are quite different between volume and volatility, especially for the intraday bandwidth where the volume bandwidth is twice as large as the volatility bandwidth for both companies. From this, one might hypothesize that volatility surfaces from different companies over the same time span have some more explanatory power over each other, than the volume surface of the same company and vice versa. This result is somehow supported by the correlations between the surfaces given below. However, the evidence supporting this hypothesis in

---

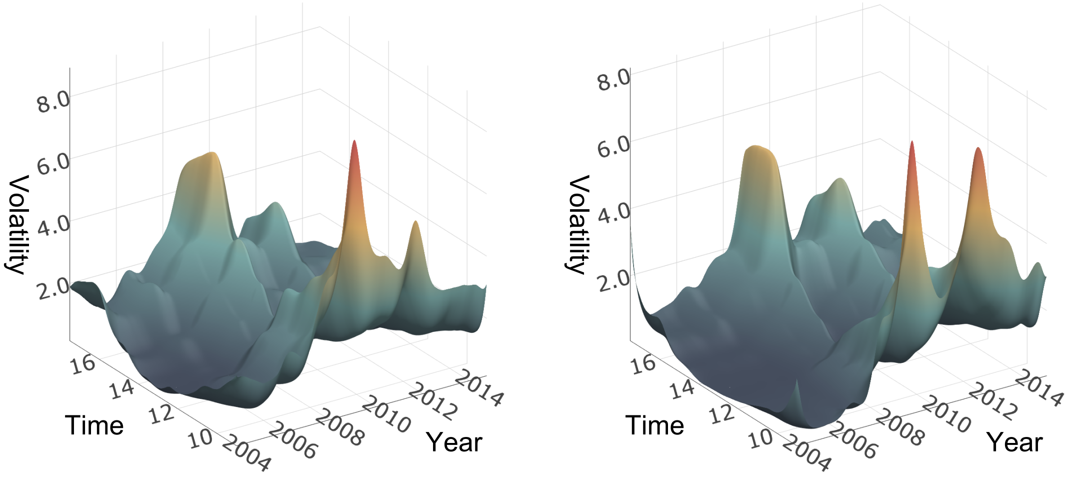
[4] The data was aggregated by calendar time sampling (CTS). Original Data was obtained from Thomson Reuters.

Table 3.2: *Selected bandwidths and estimated variance factors c_f for volatility and volume data of Siemens (SIE) and BMW under an iid. error term assumption.*

Data	$\mathbf{n}$	$\mathbf{h}$	$\mathbf{n \cdot h}$	$\mathbf{c_f}$
SIE (Volatility)	2738 510	0.06521 0.07663	179 Days 39 Minutes	28.597
BMW (Volatility)	2738 510	0.06340 0.07946	174 Days 41 Minutes	38.329
SIE (Volume)	2738 510	0.07371 0.15990	202 Days 82 Minutes	6.332E+08
BMW (Volume)	2738 510	0.07511 0.16448	206 Days 84 Minutes	1.165E+08

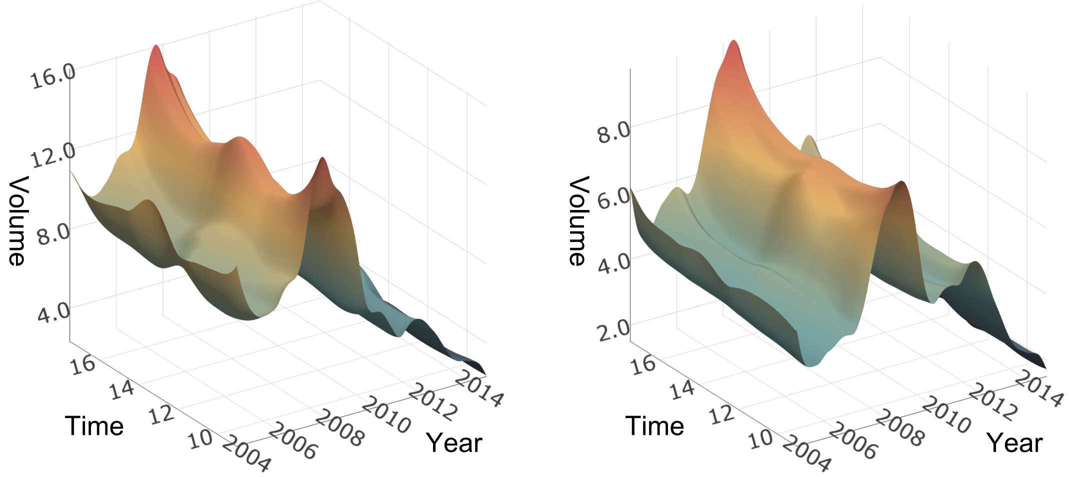
this application examples is sparse and no conclusions can be drawn from these findings here. Further research is necessary.

The volatility and volume surfaces show similar patterns and exhibit a correlation of 0.226 between volatility and volume surface for SIE and 0.209 for BMW. The correlation between the volatility surfaces of the companies is 0.848, between the volumes 0.869. Cross-correlations are -0.142 between SIE volatility and BMW volumes and 0.513 for its counterpart. The relation between volatility and trading volume is a well-known topic in finance (see e.g. Karpoff, 1987; Brailsford, 1996; Lee and Rui, 2002). The proposed methods for smoothing HFF-surfaces provide useful results for further analysis of the intraday and interday correlations of stock price volatility and trading volume or volatility (respective volume) correlations between different companies. The volatility surfaces seem to exhibit a slight undersmoothing which might be caused by misspecification of model (3.2.1) for stock returns and trading volumes as the assumption of iid. error terms is likely not met by financial data. However, for the volume surfaces this problem appears to be less severe.



(a) Spot volatility of SIE

(b) Spot volatility of BMW



(c) Trading volume surface of SIE

(d) Trading volume surface of BMW

Figure 3.3: *Estimated spot volatility and trading volume surfaces of Siemens AG (SIE) and BMW AG using the bandwidths in Table 3.2. Values of volatility are in* 1E−04*, of volumes in* 1E+03*. The volatility surface is retransformed from the model* (3.6.1).

3.7 Final Remarks

We propose a double conditional smoothing scheme to improve the efficiency of nonparametric estimation for functional time series on a regular lattice. The bandwidth selection by minimization of the AMISE is considered and asymptotic formulas for the bandwidths are derived. We use an IPI method based on these asymptotic optimal bandwidths and employ the functional double conditional smoothing for fast computation of the bandwidths by the plug-in method. This newly developed methods allow for much faster computation of the mean surfaces or its derivatives than the classical bivariate smoothers and thus, for a much faster bandwidth selection. The proposed functional scheme will be particularly helpful for smoothing large data sets and is not limited to time series applications.

The model used throughout this chapter only considers non-dependent errors. An extension of the model and smoothing scheme to a dependent error structure with short and long memory is of interest as these effects regularly occur in functional time series. The boundary problem can be solved through other nonparametric smoothing techniques such as local polynomial regression or spline regression.

3.8 Appendix

3.8.1 Proof of Theorem 3.1

Proof. Note that $\widehat{f}(x)$ in the NW-weights (3.2.6) is a kernel density estimator for the design density of X with expectation

$$\mathbb{E}\{\widehat{f}(x)\} = \frac{1}{h}\int_0^1 K_q\left(\frac{s-x}{h}\right) f(s)\,\mathrm{d}s = \int_{-1}^{q} K_q(u) f(x-uh)\,\mathrm{d}u.$$

In the equidistant case it is $f(x) = 1$ and, from Definition 3.4, the kernel function $K_q(u)$ is normalized to 1. A Taylor expansion yields then

$$= \int_{-1}^{q} K(u)[f(x) + f'(x)hu + \ldots]\,\mathrm{d}u = \int_{-1}^{q} K(u)\,\mathrm{d}u = 1$$

and hence, $\mathbb{E}\{\widehat{f}(x)\} = 1 + o(1)$. Then, Theorem 3.1 for $w^a_{i,j}$ and $w^b_{i,j}$ follows directly from comparing $w^a_{\bullet,i}$ and $w^b_{\bullet,i}$ in the limit $n \to \infty$.

For the univariate Gasser-Müller weights (3.2.8) under equidistant design, define

$$s_i = \frac{x_{i-1} + x_i}{2} = \frac{2i-1}{2n} \quad \text{for} \quad 1 < i < n,$$

for $i = 0, n$ similar arguments hold. From the mean value theorem, there exists a $z_i \in [i-1, i]$, such that

$$\frac{1}{h}\int_{s_{i-1}}^{s_i} K_q\left(\frac{u-x}{h}\right)\,\mathrm{d}u = \frac{1}{nh} K_q\left(\frac{z_i/n - x}{h}\right).$$

Using the Lipschitz continuity of K_q where $L = const.$ and setting $x = i_0/n$, we arrive at

$$\begin{aligned}
|w^c_i - w^a_i| &= \left|\frac{1}{h}\int_{s_{i-1}}^{s_i} K_q\left(\frac{u-x}{h}\right)\,\mathrm{d}u - \frac{1}{nh}K_q\left(\frac{i-i_0}{nh}\right)\right| \\
&= \left|\frac{1}{nh}K_q\left(\frac{z_i - i_0}{nh}\right) - \frac{1}{nh}K_q\left(\frac{i-i_0}{nh}\right)\right| \\
&\leq \frac{L}{nh}\left|\frac{z_i - i}{nh}\right| \leq \left(\frac{1}{nh}\right)^2 \\
\Rightarrow |w^c_i - w^a_i| &= O\left(\left(\frac{1}{nh}\right)^2\right).
\end{aligned}$$

For the product weights, it follows

$$\begin{aligned}
w^c_{i,j} &= w^c_{1,i} w^c_{2,j} \\
&= \left[w^a_{1,i} + O\left((n_1 h_1)^{-2}\right)\right]\left[w^a_{2,j} + O\left((n_2 h_2)^{-2}\right)\right] \\
&= w^a_{1,i} w^a_{2,j} + O\left((n_1 h_1)^{-2}\right) + O\left((n_2 h_2)^{-2}\right) \\
&= w^a_{i,j} + O\left((n_1 h_1)^{-2}\right) + O\left((n_2 h_2)^{-2}\right)
\end{aligned} \tag{3.8.1}$$

By observing Assumption A3, $nh \to \infty$ when $n \to \infty$, Theorem 3.1 follows for $w^c_{i,j}$ and $w^a_{i,j}$ from (3.8.1). The equivalency of $w^b_{i,j}$ and $w^c_{i,j}$ results directly from the above considerations. □

3.8.2 Expectation and Variance of the DCS Estimator

Definition 3.4. ***(Boundary Kernel Function).*** *A function $K(u) : [-1, q] \to \mathbb{R}$ is called a right boundary kernel function of order $(k, \nu), k > \nu$ for $q \in [0, 1]$, if it satisfies the following properties:*

$$\int_{-1}^{q} K(u)u^j \, \mathrm{d}u = \begin{cases} 0 & \textit{for } j = 0, \dots, k-1; j \neq \nu \\ \nu! & \textit{for } j = \nu \\ b_k \neq 0 & \textit{for } j = k. \end{cases}$$

Without loss of generality, we consider a kernel on the right boundary ($q \in [0, 1]$) or the interior region ($q = 1$). The results also hold for left boundary kernels with $K_L(u) = (-1)^\nu K(-u)$. Let $u = (\widetilde{u} - x_1)/h_1$ and $v = (\widetilde{v} - x_2)/h_2$. The integral approximation of the expectation of the DCS estimator (3.3.1) is similar to that given by Gasser and Müller (1984) or Müller and Prewitt (1993); Facer and Müller (2003)

$$\begin{aligned} &\mathbb{E}\left\{\widehat{m}^{(\nu_1,\nu_2)}\ (x_1, x_2)\right\} \\ &\quad = \frac{1}{h_1^{\nu_1} h_2^{\nu_2}} \int_{-1}^{q_1} \int_{-1}^{q_2} K_1(u) K_2(v) m(x_1 + uh_1, x_2 + vh_2) \, \mathrm{d}u \, \mathrm{d}v \\ &\qquad + O\left(\frac{1}{n_2 h_1^{\nu_1} h_2^{\nu_2}}\right) + O\left(\frac{1}{n_1 h_1^{\nu_1} h_2^{\nu_2}}\right). \end{aligned} \tag{3.8.2}$$

Gasser and Müller (1984) also showed, that the integral approximation for expectation and variance remain valid at the boundary in the univariate case.

Proof. Let Assumptions A1 to A5 hold. Then

$$\begin{aligned} &\left| \mathbb{E}\left[\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)\right] - \frac{1}{h_1^{\nu_1+1} h_2^{\nu_2+1}} \int_0^1 \int_0^1 K_1\left(\frac{\widetilde{u} - x_1}{h_1}\right) K_2\left(\frac{\widetilde{v} - x_2}{h_2}\right) m(u, v) \, \mathrm{d}\widetilde{u} \, \mathrm{d}\widetilde{v} \right| \\ &\leq \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left| \frac{1}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} K_1\left(\frac{x_{1,i} - x_1}{h_1}\right) K_2\left(\frac{x_{2,j} - x_2}{h_2}\right) m(x_{1,i}, x_{2,j}) \right. \\ &\qquad \left. - \frac{1}{h_1^{\nu_1+1} h_2^{\nu_2+1}} \int_{r_{i-1}}^{r_i} \int_{s_{j-1}}^{s_j} K_1\left(\frac{\widetilde{u} - x_1}{h_1}\right) K_2\left(\frac{\widetilde{v} - x_2}{h_2}\right) m(\widetilde{u}, \widetilde{v}) \, \mathrm{d}\widetilde{u} \, \mathrm{d}\widetilde{v} \right| \\ &= \frac{1}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left| K_1\left(\frac{x_{1,i} - x_1}{h_1}\right) K_2\left(\frac{x_{2,j} - x_2}{h_2}\right) m(x_{1,i}, x_{2,j}) \right. \\ &\qquad \left. - K_1\left(\frac{\xi_{1,i} - x_1}{h_1}\right) K_2\left(\frac{\xi_{2,j} - x_2}{h_2}\right) m(\xi_{1,i}, \xi_{2,j}) \right|, \end{aligned}$$

where $\xi_{1,i} \in [r_{i-1}, r_i]$, $\xi_{2,j} \in [s_{j-1}, s_j]$ are suitable mean values. Let r_i, s_j be suitable partitions on $[0, 1]$, such that, under equidistant design, $r_i - r_{i-1} = n_1^{-1}$ and $s_j - s_{j-1} = n_2^{-1}$

for $1 < i < n_1$, $1 < j < n_2$. Denote the set of all integers i, j leading to non-zero weights in $K_1(\bullet), K_2(\bullet)$ by Q_1^*, Q_2^* where

$$Q_1^* = \{i : -n_1 h_1 \le i \le q_1 n_1 h_1, i \in \mathbb{Z}\},$$
$$Q_2^* = \{j : -n_2 h_2 \le j \le q_2 n_2 h_2, j \in \mathbb{Z}\}.$$

The number of elements in Q_1^*, Q_2^* is $|Q_1^*| = O(n_1 h_1)$ and $|Q_2^*| = O(n_2 h_2)$. Using the Lipschitz continuitiy of K_1, K_2, m and summation over non-zero values, we arrive at

$$\begin{aligned}
&\le \frac{L}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \sum_{i \in Q_1^*} \sum_{j \in Q_2^*} \left(|x_{1,i} - \xi_{1,i}| + |x_{2,j} - \xi_{2,j}|\right) \\
&\le \frac{L}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \sum_{i \in Q_1^*} \sum_{j \in Q_2^*} \left(\frac{1}{n_1} + \frac{1}{n_2}\right) \\
&= O\left(\frac{1}{n_2 h_1^{\nu_1} h_2^{\nu_2}}\right) + O\left(\frac{1}{n_1 h_1^{\nu_1} h_2^{\nu_2}}\right). \qquad \square
\end{aligned}$$

where L is a suitable constant.

A Taylor expansion of $m(x_1 + h_1 u, x_2 + h_2 v)$ in (3.8.2) around a point (x_1, x_2) yields, after applying Definition 3.4,

$$\begin{aligned}
\mathbb{E}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)\right\} &= m^{(\nu_1,\nu_2)}(x_1, x_2) + B_m + R_m \\
&= m^{(\nu_1,\nu_2)}(x_1, x_2) + B_m[1 + o(1)],
\end{aligned}$$

where $B_m(x_1, x_2)$ is defined by (3.3.4). Using $\tilde{\mu}_j(K) = \int_{-1}^{q} K(u) u^j / j! \,\mathrm{d}u$ and observing that $(-1)^j \tilde{\mu}_j(K) \ge 0$, we can assess the order of magnitude of $R_m(x_1, x_2)$:

$$\begin{aligned}
|R_m(x_1, x_2)| = &\Bigg| \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) h_1^{k_1-\nu_1} h_2^{k_2-\nu_2} m^{(k_1,k_2)}(x_1, x_2) \\
&+ \sum_{\substack{r \ge \nu_1, s \ge \nu_2 \\ \max(r-k_1, s-k_2) > 0}} \tilde{\mu}_r(K_1) \tilde{\mu}_s(K_2) h_1^{r-\nu_1} h_2^{s-\nu_2} m^{(r,s)}(x_1, x_2) \Bigg| \\
&\le \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) \left(h_1^{k_1-\nu_1} + h_2^{k_2-\nu_2}\right) \\
&\cdot \Bigg| m^{(k_1,k_2)}(x_1, x_2) + \sum_{\substack{r \ge \nu_1, s \ge \nu_2 \\ \max(r-k_1, s-k_2) > 0}} m^{(r,s)}(x_1, x_2) \Bigg| \\
&= O\left(h_1^{k_1-\nu_1}\right) + O\left(h_2^{k_2-\nu_2}\right) = O\left(B_m(x_1, x_2)\right).
\end{aligned}$$

Regarding the variance, note that from model (3.2.1) and Assumption A4 of iid. error terms, $\operatorname{var}(y_{i,j}) = \mathbb{E}(\varepsilon_{i,j}^2) = \sigma^2$ and $\operatorname{cov}(\varepsilon_{i,j}, \varepsilon_{r,s}) = 0$ for $(i,j) \neq (r,s)$. The variance can hence be approximated in the following way

$$
\begin{aligned}
&\operatorname{var}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} \\
&\quad= \frac{1}{(n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1})^2} \mathbb{E}\left\{\left(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} K_1\left(\frac{x_{1,i}-x_1}{h_1}\right) K_2\left(\frac{x_{2,j}-x_2}{h_2}\right)\varepsilon_{ij}\right)^2\right\} \\
&\quad= \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\left[R(K_1)R(K_2)+o(1)\right] \\
&\quad= \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} R(K_1)R(K_2) + O\left(\frac{1}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\right) \\
&\quad= V_m + O\left(\frac{1}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\right).
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&\left|\operatorname{var}\left\{m^{(\nu_1,\nu_2)}(x_1,x_2)\right\} - \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} R(K_1)R(K_2)\right| \\
&= \frac{\sigma^2}{n_1^2 n_2^2 h_1^{2\nu_1+2} h_2^{2\nu_2+2}}\left|\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\left[K_1^2\left(\frac{x_{1,i}-x_1}{h_1}\right) K_2^2\left(\frac{x_{2,j}-x_2}{h_2}\right)\right.\right. \\
&\qquad\left.\left. - K_1^2\left(\frac{\xi_{1,i}-x_1}{h_1}\right) K_2^2\left(\frac{\xi_{2,j}-x_2}{h_2}\right)\right]\right|
\end{aligned}
$$

for suitable mean values $\xi_{1,i}, \xi_{2,j}$. Using Lipschitz continuity of the kernel functions K, Q_1^*, Q_2^* as defined above and noting that $x_i - \xi_i \leq 1/n.$, we assess the order of magnitude of the variance to

$$
\begin{aligned}
&\leq \frac{\sigma^2}{n_1^2 n_2^2 h_1^{2\nu_1+2} h_2^{2\nu_2+2}} \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\left|K_1^2\left(\frac{x_{1,i}-x_1}{h_1}\right) K_2^2\left(\frac{x_{2,j}-x_2}{h_2}\right)\right. \\
&\qquad\left. - K_1^2\left(\frac{\xi_{1,i}-x_1}{h_1}\right) K_2^2\left(\frac{\xi_{2,j}-x_1}{h_1}\right)\right| \\
&\leq \frac{\sigma^2}{n_1^2 n_2^2 h_1^{2\nu_1+2} h_2^{2\nu_2+2}} \sum_{i\in Q_1^*}\sum_{j\in Q_2^*}\left(|x_{1,i}-\xi_{1,i}| + |x_{2,j}-\xi_{2,j}|\right) \\
&\leq \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\left(\frac{1}{n_1}+\frac{1}{n_2}\right) \\
&\leq \frac{\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} = O\left(\frac{1}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\right).
\end{aligned}
$$

□

3.8.3 Asymptotic Normality of the DCS Estimator

The estimator (3.3.1) can be written as triangular array in $n = n_1 \cdot n_2$ of elements $z_{i,j,n}$

$$\widehat{m}_n^{(\nu_1,\nu_2)}(x_1, x_2) = \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} z_{i,j,n} \quad \text{with}$$
$$z_{i,j,n} = \frac{1}{\widetilde{h}_1^{\nu_1+1} \widetilde{h}_2^{\nu_2+1}} K_1\left(\frac{x_{1,i} - x_1}{h_1}\right) K_2\left(\frac{x_{2,j} - x_2}{h_2}\right) y_{i,j}.$$

Note that $\operatorname{var}(z_{i,j,n}) = O\left((h_1^{2\nu_1+1} h_2^{2\nu_2+1})^{-1}\right)$. With

$$\begin{aligned}\mathbb{E}\left(|z_{i,j,n} - \mathbb{E}(z_{i,j,n})|^{2+r}\right) &\leq\ 2^{1+r}\, \mathbb{E}\left(|z_{i,j,n}|^{2+r} + |\mathbb{E}(z_{i,j,n})|^{2+r}\right) \\ &\leq\ 2^{2+r}\, \mathbb{E}\left(|z_{i,j,n}|^{2+r}\right) = O\left(\frac{1}{\left(h_1^{\nu_1+1} h_2^{\nu_2+1}\right)^{2+r}}\right)\end{aligned}$$

and $s_n^2 = \sum_{i=1}^{n_2} \sum_{j=1}^{n_2} \operatorname{var}(z_{in}) = n_1 n_2 \operatorname{var}(z_{i,j,n})$, the Lyapunov condition

$$\lim_{n\to\infty} \frac{1}{s_n^2} \sum_{i=1}^{n_2} \sum_{j=1}^{n_2} \mathbb{E}\left(|z_{in} - \mathbb{E}(Z_{in})|^{2+r}\right) = \lim_{n\to\infty} O\left(\frac{1}{(n_1 n_2 h_1 h_2)^{\frac{r}{2}}}\right) = 0$$

holds with Assumption A3. Hence we yield by the Lyapunov CLT

$$\sqrt{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \left(\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2) - m^{(\nu_1,\nu_2)}(x_1, x_2)\right) \xrightarrow{d} \mathcal{N}(0, \sigma^2).$$

A neccessary condition for the Lyapunov CLT is, that the z_{in} are independently distributed. We can justify this assumption similar to Gasser and Müller (1984): for two disjunct points $(x_{1,i}, x_{2,j})$ and $(x_{1,k}, x_{2,l})$, $i \neq k, j \neq l$, there exists a finite n_0 such that $\widehat{m}_n^{(\nu_1,\nu_2)}(x_{1,i}, x_{2,j})$ and $\widehat{m}_n^{(\nu_1,\nu_2)}(x_{1,k}, x_{2,l})$ are independent for $n > n_0$. This is a consequence of $h \to 0$ as $n \to \infty$ and the compactness of the used kernels.

3.8.4 Derivation of the Optimal Bandwidths

The optimal bandwidths for estimation of the regression surface $m(x_1, x_2)$ or its partial derivatives $m^{(\nu_1,\nu_2)}(x_1, x_2)$ on $(x_1, x_2) \in [0, 1]^2$ are obtained by minimizing the AMISE (3.3.3) (see e.g. Herrmann et al., 1995). For simplification of the results, we assume that $k_1 - \nu_2 = k_2 - \nu_2 \equiv \delta$ and note that $\tilde{\mu}_k(K) = b/k!$ given a kernel $K(u)$ of order (k, ν) according to Definition 3.4.

$$\begin{aligned}\text{MISE } &= \text{AMISE } + o\left(h_1^{2(k_1-\nu_1)} + h_2^{2(k_2-\nu_2)} + n_1^{-1} n_2^{-1} h_1^{-(2\nu_1+1)} h_2^{-(2\nu_2+1)}\right) \\ &\quad + O\left(n_1^{-1} h_1^{-\nu_1} h_2^{-\nu_2}\right) + O\left(n_2^{-1} h_1^{-\nu_1} h_2^{-\nu_2}\right),\end{aligned}$$

$$\begin{aligned}
\text{AMISE} &= \int_0^1 \int_0^1 B_m^2(x_1, x_2)\,\mathrm{d}x_1\,\mathrm{d}x_2 + V_m \\
&= \tilde{\mu}_{k_1}^2(K_1) h_1^{2\delta} I_{11} + \tilde{\mu}_{k_2}^2(K_2) h_2^{2\delta} I_{22} + 2(-1)^{k_1+k_2} \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) h_1^{\delta} h_2^{\delta} I_{12} \\
&\quad + \frac{R(K_1)R(K_2)\sigma^2}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}},
\end{aligned}$$

with kernel constants $R(K)$, $\tilde{\mu}_j(K)$ as defined above and σ being the variance of the iid. error terms $\varepsilon_{i,j}$. More generally than in (3.4.2), we now define the integrals $I_{r,s}$ by

$$\begin{aligned}
I_{r,s} &= \int_0^1 \int_0^1 m^{(\tau_r)}(x_1, x_2)\, m^{(\tau_s)}(x_1, x_2\,\mathrm{d}x_1\,\mathrm{d}x_2 \\
\text{where} \quad \tau_i &= \begin{cases} (k_1, \nu_2) & \text{for } i = 1 \\ (\nu_1, k_2) & \text{for } i = 2. \end{cases}
\end{aligned}$$

The first order condition for minimizing the AMISE is then

$$\begin{aligned}
\frac{\partial \text{AMISE}}{\partial h_1} &= 2\delta \tilde{\mu}_{k_1}^2(K_1) I_{11} h_1^{2\delta-1} + 2\delta \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) I_{12} h_1^{\delta-1} h_2^{\delta} \\
&\quad - (2\nu_1 + 1) \frac{R(K_1)R(K_2)\sigma^2}{n_1 n_2 h_1^{2\nu_1+2} h_2^{2\nu_2+1}} = 0.
\end{aligned} \tag{3.8.3}$$

Multiplying (3.8.3) by $h_1/(2\nu_1 + 1)$ and equalizing the equation with its counterpart $\partial \text{AMISE}\,/\partial h_2 = 0$ results in

$$\frac{\tilde{\mu}_{k_1}^2(K_1)}{2\nu_1 + 1} I_{11} h_1^{2\delta} - \frac{\tilde{\mu}_{k_2}^2(K_2)}{2\nu_2 + 1} I_{22} h_2^{2\delta} + (\nu_2 - \nu_1) \frac{\tilde{\mu}_{k_1}(K_1)\tilde{\mu}_{k_2}(K_2)}{(2\nu_1 + 1)(2\nu_2 + 1)} I_{12} h_1^{\delta} h_2^{\delta} = 0,$$

which is a quadratic equation in h_1^{δ}, h_2^{δ}, with solution

$$\begin{aligned}
h_1^{\delta} &= A^{\delta} h_2^{\delta}, \\
A^{\delta} &= \frac{\tilde{\mu}_{k_2}(K_2)}{\tilde{\mu}_{k_1}(K_1)} \left(\frac{\nu_1 - \nu_2}{2\nu_2 + 1} \frac{I_{12}}{I_{11}} \pm \left[\frac{(\nu_1 - \nu_2)^2}{(2\nu_2 + 1)^2} \frac{I_{12}^2}{I_{11}^2} + \frac{2\nu_1 + 1}{2\nu_2 + 1} \frac{I_{22}}{I_{11}} \right]^{\frac{1}{2}} \right),
\end{aligned}$$

such that $A^{\delta} \geq 0, A \geq 0$. The relation between h_1 and h_2 along with the first order condition yields the formulas for the optimal bandwidths

$$\begin{aligned}
h_{1,A} &= \left[\frac{2\nu_1 + 1}{2\delta} \frac{R(K_1)R(K_2)\sigma^2}{n_1 n_2 A^{-(2\nu_2+1)} C_1} \right]^{\frac{1}{2(\delta+\nu_1+\nu_2+1)}}, \\
&\text{with} \quad C_1 = \tilde{\mu}_{k_1}^2(K_1) I_{11} + \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) I_{12} A^{-\delta}
\end{aligned} \tag{3.8.4}$$

$$\begin{aligned}
h_{2,A} &= \left[\frac{2\nu_2 + 1}{2\delta} \frac{R(K_1)R(K_2)\sigma^2}{n_1 n_2 A^{(2\nu_1+1)} C_2} \right]^{\frac{1}{2(\delta+\nu_1+\nu_2+1)}} \\
&\text{with} \quad C_2 = \tilde{\mu}_{k_2}^2(K_2) I_{22} + \tilde{\mu}_{k_1}(K_1) \tilde{\mu}_{k_2}(K_2) I_{12} A^{\delta},
\end{aligned} \tag{3.8.5}$$

where (3.8.5) follows from symmetry. The second derivatives $\partial^2 \mathrm{AMISE}/\partial h_i^2$ can be shown to be positive for $\delta \geq 1$ by strainghtforward calculation, hence the optimal bandwidths (3.8.4), (3.8.5) constitute a minimum indeed. If the regression surface is considered ($\nu_1 = \nu_2 = 0$) and $k_1 = k_2 = 2$, the equations (3.4.3) and (3.4.4) of Proposition 3.1 follow directly.

4 Local Polynomial Double Conditional Smoothing under Dependent Errors

This chapter is published with slight differences in the CIE Working Papers (143), Paderborn University, under the title ”Bandwidth selection for the Local Polynomial Double Conditional Smoothing under Spatial ARMA Errors”.

4.1 Introduction

Functional data arise in many research areas such as physics, geography, biology, and also in economics and finance. This data appear in various forms, from spatial land survey data to functional time series with a time- and a temporal dimension. Examples for such time series are the intraday observations of temperatures or pollution over several days in a specific spot and other series, where a quantitiy of interest can be measured in intraday time intervals over several days (see e.g. Ramsay and Silverman, 2005; Chiou and Müller, 2009; Hyndman and Shang, 2010; Horváth and Kokoszka, 2012; Li et al., 2019a). For financial applications, functional or spatial representation of volatility surfaces was proposed by Feng (2013) and Peitz and Feng (2015) based on results from Andersen and Bollerslev (1997, 1998) and Andersen et al. (2000). Nonparametric regression for a spatial representation on a lattice was investigated, among others, by Hyndman and Ullah (2007); Aneiros-Pérez and Vieu (2008); Horváth and Kokoszka (2012) and Gao and Shang (2017).

We consider nonparametric estimation of the expectation surface under a functional time series on a regular lattice. The idea of the double conditional smoothing (DCS) proposed by Feng (2013) and investigated under kernel regression and iid. errors in Chapter 3 will be extended to local polynomial estimation. This type of smoother provides additional accuracy and straightforward estimation of derivatives of functions and surfaces. The advantages of adopting local polynomial regression for the DCS come at the cost of increased computation time compared to the kernel regression methods proposed in Chapter 3. Local polynomial regression for multivariate data was investigated, for instance, by Yang and Tschernig (1999); Hallin et al. (2004) and Wang and Wang (2009). Ruppert and Wand (1994) considered the asymptotic behavior of these estimators and derived formulas for the optimal bandwidth selection based on minimization of the mean integrated squared error (MISE). An overview on spatial local regression can be found in Scott (2015) or Ghosh (2018). We motivate the use of local polynomial regression by a higher degree of precision and direct estimation of the derivatives of the surface under consideration which is directly useful for the bandwidth selection procedure as higher order derivatives are required in the formulas. Further, it solves the boundary problem of kernel regression. This is especially helpful when dealing with two-dimensional data as now the ratio of the boundary region to the interior is usually larger than in the univariate case. Kernel regression, as discussed in Chapter 3 and in the literature, e.g., by Müller and Prewitt (1993); Herrmann et al. (1995) and Facer and Müller (2003), is subject to the boundary problem, where the order of the bias at the boundary differs from the bias in the interior. This flaw has to be corrected by using elaborate boundary kernels, proposed by Gasser and Müller (1979) and, for instance, given by Müller (1991) or Müller and Wang (1994). Even with boundary correction kernels, estimation at the boundary is unstable, especially for the estimation of derivatives. This problem can be overcome, or at least mitigated, by using local polynomial regression instead of kernel regression which has an automatic boundary correction (see e.g. Fan and Gijbels, 1992, 1996).

We address the bandwidth selection for the local regression DCS by employing a data-driven iterative plug-in (IPI) procedure (see Section 3.4.2 and Gasser et al., 1991; Ruppert et al., 1995). The asymptotic optimal bandwidths are found from minimization of the asymptotic MISE (AMISE) of the estimator. The formulas for the optimal bandwidths include partial derivatives of the regression surface under consideration which are estimated via local polynomial DCS (LP-DCS) itself and hence, auxiliary bandwidths are required. Under the IPI these bandwidths are obtained from the optimal bandwidths of the regression surface and the estimates of the partial derivatives are iteratively plugged into the formulas for the optimal bandwidth in each step.

In many applications the innovations of a time series are subject to some kind of dependency structure. This is especially true in financial econometrics where dependency structures are utilized to estimate volatility (see Andersen and Bollerslev, 1998; Andersen et al., 2003, 2004) or the risk for stock prices, but dependency arises in other fields too, e.g., when studying weather or climate dynamics (see Chapter 5). One-dimensional kernel regression under correlated errors is a well-researched topic, considered among others by Altman (1990, 1993); Hart (1991) and also by Feng (2013) for the DCS. Local polynomial estimators under correlated errors are a more recent field of research, e.g., conducted by Francisco-Fernández and Vilar-Fernández (2001), Opsomer et al. (2001) and Brabanter et al. (2018). Local regression under long-memory dependencies was considered by Feng et al. (2021a) for a spatial FARIMA model; other recent work in this area was done by Robinson (2020) and Li et al. (2019a).

We employ a parametric model for modeling the error structure under our functional framework by using a spatial ARMA (SARMA) process. The SARMA offers a well-researched parametric way to incorporate dependency structures into nonparametric surface regression. Random fields on a lattice or spatial stochastic processes proved useful for various applications and are studied in-depth in the literature. A general review of statistical techniques for analyzing spatial patterns was done by Bartlett (1975) for early developments in this area. Specific spatial processes were investigated, e.g., by Tjøstheim (1978) and Martin (1979) where the analysis is primarily focused on spatial patterns in fields such as geography, ecology and agriculture (Martin, 1990) and urban economics (Fisher, 1971). However, these results are also useful in financial applications under a time-series context.

Estimation of the parameters of SARMA models is closely related to the parameter estimation of one-dimensional ARMA processes; most univariate estimation procedures can be extended to lattice processes or do already exist. Common maximum-likelihood estimators might run too slowly when applied to large data sets, notably if estimation is required multiple times, for either order selection or in the IPI bandwidth selection algorithm. Parameter estimation for SARMA processes was researched, among others, by Yao and Brockwell (2006) who propose a spatial variant of the innovations algorithm by Brockwell and Davis (1991). However, this algorithm runs slowly for large data sets. Faster approaches might be least squares estimators, e.g., a spatial version of the estimation algorithm by Hannan and Rissanen (1982). A special case of the SARMA process is the separable SARMA which can be written as the product of two univariate ARMA processes in the respective directions. Hence, making an analogy with the DCS, estimation of the parameters of the two-dimensional process is reduced to estimation of two one-dimensional processes. We propose a convenient matrix notation for SARMA models and point out some estimation methods for separable SARMA processes. For pure SAR processes, i.e. spatial AR processes, a two-dimensional version of the Yule-Walker estimator is presented, allowing for fast estimation of dependency models where the MA-part is zero.

The finite sample behavior of the local polynomial regression DCS is assessed by a simulation study which shows that the proposed estimators work well under some conditions. Moreover, the introduction of dependent errors does no lead to a distinct decrease in estimation efficiency of the optimal bandwidths, although the distribution of the variance coefficient estimates for the SARMA model employs a higher variance than in the iid. case. Application of the proposed algorithms to high-frequency financial (HFF) data of the stock returns of Allianz SE and Siemens AG indicate that accounting for dependency leads to increased bandwidths for this data sets. Hence, an incorrectly made iid. assumption is likely to cause an over- or undersmoothing of the expectation surface of the data.

We extend the DCS and functional DCS (FDCS) scheme from Feng (2013) and Chapter 3 to local polynomial regression in Section 4.2 the asymptotic properties of this new estimator are discussed in Section 4.3.1. Optimal bandwidth selection using an IPI algorithm for the local polynomial DCS is addressed in Section 4.3.2. Section 4.4 considers the dependency structure of the error terms and defines the SARMA, some estimation procedures are suggested. In Section 4.5, the proposed bandwidth selection algorithm is assessed along with the SARMA estimator, application to financial data is done in Section 4.6. We close with some final remarks in Section 4.7. The appendix Section 4.8 contains some additional proofs and derivations.

4.2 The FDCS for Local Polynomial Estimators

4.2.1 Model and Assumptions

A general bivariate non- or semiparametric spatial regression model requires regression on two covariates simultaneously. The DCS splits this estimation into two separate estimation procedures, effectively reducing a two-dimensional problem into two sequencial one-dimensional estimation procedures. This provides a computational advantage over the classical bivariate smoothers, as only univariate regression is considered. In the works by Feng (2013) and Peitz and Feng (2015) as well as in Chapter 3, the DCS is proposed for kernel regression, an extension to local polynomial smoothers will be the scope of this chapter.

Let $Y_{i,j}$ be functional time series observed on a regular lattice spanned by components $X_1 = \{x_{1,i}\}$, $i = 1, \ldots, n_1$ and $X_2 = \{x_{2,j}\}$, $j = 1, \ldots, n_2$, which are assumed to be equidistant. Extension to the case where one component is non-equidistant or even has random design is possible, due to the design adaptivity of local regression. The spatial nonparametric model is

$$Y_{i,j} = m(x_{1,i}, x_{2,j}) + \varepsilon_{i,j}, \tag{4.2.1}$$

where $m(x_1, x_2)$ is a deterministic mean or expectation surface and $\{\varepsilon_{i,j}\}$ is a stationary random field with zero mean. We aim at estimation of the mean surface $m(x_1, x_2)$ or a (partial) derivative $m^{(\nu_1,\nu_2)}(x_1, x_2)$. To derive the local polynomial DCS (LP-DCS) estimator and its asymptotic properties, we establish the following assumptions:

A1. The random field $Y_{i,j}$ of model (4.2.1) forms a triangular array on an equidistant lattice on $[0,1]^2$ spanned by sets of design points $X_1 = \{x_{1,i}\}$, $i = 1, \ldots, n_1$ and $X_2 = \{x_{2,j}\}$, $j = 1, \ldots, n_2$. The points $x_{1,i}, x_{2,j}$ are given by rescaled variables $x_{1,i} = i/n_1$ and $x_{2,j} = j/n_2$.

A2. The expectation surface $m(x_1, x_2)$ is a smooth and Lipschitz continuous function defined on $[0,1]^2$, which is at least $(p_1 + 1, p_2 + 1)$ times continuously differentiable with respect to (x_1, x_2).

A3. The bandwidths h_1, h_2 satisfy the conditions $h_1, h_2 \to 0$, $n_1 h_1^{\nu_1+1}, n_2 h_2^{\nu_2+1} \to \infty$ and $n_1 h_1^{\nu_1} h_2^{\nu_2}, n_2 h_1^{\nu_1} h_2^{\nu_2} \to \infty$, as $n_1, n_2 \to \infty$.

A4. The error function $\{\varepsilon_{i,j}\}$ is a stationary random field or functional time series with zero-mean and covariance $\operatorname{cov}(\varepsilon_{i,j}, \varepsilon_{i-s,j-t}) = \mathbb{E}(\varepsilon_{i,j}\varepsilon_{i-s,j-t}) = \gamma(s,t)$, where $\sum_s \sum_t \gamma(s,t) < \infty$.

A5. Bivariate product weights $W(u,v) = W_1(u)W_2(v)$ are used, where the weights W_1, W_2 are with boundary modification.

Assumption A1 ensures that the form of the expectation surfaces is not affected by n_1 or n_2. This assumption is required for establishing the consistency and asymptotic normality of the estimator. Under A2, the integral approximation of bias and variance are finite and the optimal bandwidth derived from the AMISE exist and are well-defined, A3 ensures the asymptotic unbiasedness of the estimator and is a variant of a common assumption in nonparametric regression. In this chapter, dependency in the error terms is allowed, however, stationarity of the error terms is assumed and the dependency structure is restricted to short-memory processes by Assumption A4. The crucial assumption for the DCS to work, is the product kernel Assumption A5.

4.2.2 Extension to Local Polynomial Smoothers

Consider the Taylor expansion of the surface $m(x_1, x_2)$ around a point (x_{1,i_0}, x_{2,j_0}):

$$m(x_1, x_2) = \sum_{s=0}^{\infty}\sum_{t=0}^{\infty} \frac{(x_1 - x_{1,i_0})^s (x_2 - x_{2,j_0})^t}{s!\; t!} \left(\frac{\partial^{s+t} m}{\partial x_1^s \partial x_2^t}\right)(x_{1,i_0}, x_{2,j_0}). \tag{4.2.2}$$

Let $y_{i,j}$ be the observations in a sample of $Y_{i,j}$ for $1 \leq i \leq n_1$, $1 \leq j \leq n_2$ and X_1, X_2 be the (rescaled) covariate vectors from Assumption A1. Then, the (p_1, p_2)-th order local polynomial estimator of $m^{(\nu_1,\nu_2)}(x_1, x_2)$ is given by the coefficient β_{ν_1,ν_2} from minimization of the locally weighted least squares problem

$$\begin{aligned} &\arg\min_{\beta} Q(x_1, x_2) \quad \text{with} \\ &Q(x_1, x_2) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \left[y_{i,j} - \sum_{s=0}^{p_1}\sum_{t=0}^{p_2} \frac{(x_{1,i} - x_1)^s (x_{2,j} - x_2)^t}{s!\; t!} \beta_{s,t} \right]^2, \\ &\qquad\qquad \cdot W\left(\frac{x_{1,i} - x_1}{h_1}, \frac{x_{2,j} - x_2}{h_2}\right) \end{aligned}$$

where h_1, h_2 are the bandwidths corresponding to X_1 and X_2 and W is a bivariate weighting function. Let y be the vector constructed from vectorization of the observations $y_{i,j}$ of length $n = n_1 n_2$, $\mathbf{W}$ be the $n \times n$ matrix of suitable weights W and define the matrix of covariates by

$$\mathbf{C} = \begin{bmatrix} 1 & (x_{1,1} - x_1) & (x_{1,1} - x_1)^2 & \dots & (x_{1,1} - x_1)^{p_1}(x_{2,1} - x_2)^{p_2} \\ 1 & (x_{1,2} - x_1) & (x_{1,2} - x_1)^2 & \dots & (x_{1,2} - x_1)^{p_1}(x_{2,2} - x_2)^{p_2} \\ \vdots & \ddots & & & \\ 1 & (x_{1,n} - x_1) & & \dots & (x_{1,n} - x_1)^{p_1}(x_{2,n} - x_2)^{p_2} \end{bmatrix}.$$

Further, let e_i be the i-th unit vector and $\widehat{\beta}$ be the vector that solves $\arg\min_{\beta} Q(x_1, x_2)$ with

$$\widehat{\beta} = (\mathbf{C}^T \mathbf{W} \mathbf{C})^{-1} \mathbf{C}^T \mathbf{W} y.$$

Then, the local polynomial estimator for $m^{(\nu_1,\nu_2)}(x_1,x_2)$ is

$$\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) = \nu_1!\nu_2! e^T_{\nu_1+\nu_2+1}\widehat{\beta}. \tag{4.2.3}$$

The estimator in (4.2.3) constitutes a single-step bivariate local polynomial regression. The expansion in (4.2.2) can be rewritten as

$$m(x_1,x_2) = \sum_{s=0}^{\infty} \frac{(x_1 - x_{1,0})^s}{s!} \frac{\partial^s}{\partial x_1^s} \left(\sum_{t=0}^{\infty} \frac{(x_2 - x_{2,0})^t}{t!} \frac{\partial^t m(x_{1,0}, x_{2,0})}{\partial x_2^t} \right).$$

From this, we can directly define the DCS for local polynomial regression, if product weights (Assumption A5) are used.

Definition 4.1. ***(Local Polynomial DCS, LP-DCS).*** *Let Assumptions A1-A5 hold and $W_1(u), W_2(v)$ be some suitable weight functions. The LP-DCS estimator of $m^{(\nu_1,\nu_2)}(x_1,x_2)$ is*

$$\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) = \nu_1! e^T_{\nu_1+1}\widehat{\beta}_1 \text{ or equivalently} \tag{4.2.4}$$

$$\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) = \nu_2! e^T_{\nu_2+1}\widehat{\beta}_2, \tag{4.2.5}$$

where $\widehat{\beta}_1, \widehat{\beta}_2$ are solutions to

$$\arg\min_{\beta_1} \sum_i \left[\widehat{m}^{(\nu_2)}(x_2|x_1) - \sum_{r=1}^{p_1} \beta_{1,r} \frac{(x_{1,i} - x_1)^r}{r!} \right]^2 W_1\left(\frac{x_{1,i} - x_1}{h_1}\right), \tag{4.2.6}$$

$$\arg\min_{\beta_2} \sum_j \left[\widehat{m}^{(\nu_1)}(x_2|x_1) - \sum_{s=1}^{p_2} \beta_{2,s} \frac{(x_{2,j} - x_2)^s}{s!} \right]^2 W_2\left(\frac{x_{2,j} - x_2}{h_2}\right) \tag{4.2.7}$$

and e_i is the i-th unit vector as above. The intermediate results used in (4.2.6), (4.2.7) *are the estimators of $m^{(\nu_2)}(x_2)$ conditioning on x_1 respective of $m^{(\nu_2)}(x_1)$ conditioning on x_2. They are given by*

$$\widehat{m}^{(\nu_2)}(x_2|x_{1,i}) = \nu_2! e^T_{\nu_2+1}\widehat{\beta}_{2|1} \text{ and}$$

$$\widehat{m}^{(\nu_1)}(x_1|x_{2,j}) = \nu_1! e^T_{\nu_1+1}\widehat{\beta}_{1|2},$$

where $\widehat{\beta}_{2|1}$ and $\widehat{\beta}_{1|2}$ are solutions to

$$\arg\min_{\beta_{2|1}} \sum_j \left[y_{i,j} - \sum_{s=0}^{p_2} \beta_{2|1,s} \frac{(x_{2,j} - x_2)^s}{s!} \right] W_2\left(\frac{x_{2,j} - x_2}{h_2}\right),$$

$$\arg\min_{\beta_{1|2}} \sum_i \left[y_{i,j} - \sum_{r=0}^{p_1} \beta_{1|2,r} \frac{(x_{1,i} - x_1)^r}{r!} \right] W_1\left(\frac{x_{1,i} - x_1}{h_1}\right).$$

A more detailed survey of weight functions and boundary modification for the weights $W(u)$ is given in Section 4.2.3 and Chapter 2. The DCS is symmetric in X_1, X_2, hence the order of estimation is interchangeable. Closed-form solutions to the LP-DCS from Definition 4.1 can be derived easily:

Definition 4.2. ***(Local Polynomial Functional DCS).*** *Define* $\mathbf{Y}$ *as the* $n_1 \times n_2$ *matrix with entries* $\{\mathbf{Y}\}_{i,j} = y_{i,j}$ *and the covariate matrices at a point* $(x_{1,i}, x_{2,j})$ *by*

$$\begin{aligned} \mathbf{X}_{1,i} &= \begin{bmatrix} 1 & (X_1 - x_{1,i}) & (X_1 - x_{1,i})^2 & \dots & (X_1 - x_{1,i})^{p_1} \end{bmatrix} \quad \textit{and} \\ \mathbf{X}_{2,j} &= \begin{bmatrix} 1 & (X_2 - x_{2,j}) & (X_2 - x_{2,j})^2 & \dots & (X_2 - x_{2,j})^{p_2} \end{bmatrix}. \end{aligned}$$

The diagonal matrices of the weights[1] *are* $\mathbf{W}_{1,i}, \mathbf{W}_{2,j}$ *with elements* $W_1((X_1 - x_{1,i})/h_1)$ *respective* $W_2((X_2 - x_{2,j})/h_2)$*. The functional LP-DCS equivalent to* (4.2.4)*,* (4.2.5) *is then*

$$\begin{aligned} \left\{\widehat{\mathbf{M}}\right\}_{\substack{i \\ 1\leq j\leq n_2}} &= \nu_1! e_{\nu_1+1}^T (\mathbf{X}_{1,i}^T \mathbf{W}_{1,i} \mathbf{X}_{1,i})^{-1} \mathbf{X}_{1,i}^T \mathbf{W}_{1,i} \widehat{\mathbf{M}}_2 \quad \textit{and} \\ \left\{\widehat{\mathbf{M}}\right\}_{\substack{1\leq i\leq n_1 \\ j}} &= \nu_2! e_{\nu_2+1}^T (\mathbf{X}_{2,i}^T \mathbf{W}_{2,i} \mathbf{X}_{2,i})^{-1} \mathbf{X}_{2,i}^T \mathbf{W}_{2,i} \widehat{\mathbf{M}}_1, \end{aligned}$$

where estimation is carried out over all rows i *and all columns* j*. The intermediate results are obtained from*

$$\begin{aligned} \left\{\widehat{\mathbf{M}}_2^T\right\}_{\substack{j \\ 1\leq i\leq n_1}} &= \nu_2! e_{\nu_2+1}^T (\mathbf{X}_{2,j}^T \mathbf{W}_{2,j} \mathbf{X}_{2,j})^{-1} \mathbf{X}_{2,j}^T \mathbf{W}_{2,j} \mathbf{Y}^T \quad \textit{and} \\ \left\{\widehat{\mathbf{M}}_1\right\}_{\substack{i \\ 1\leq j\leq n_1}} &= \nu_1! e_{\nu_1+1}^T (\mathbf{X}_{1,i}^T \mathbf{W}_{1,i} \mathbf{X}_{1,i})^{-1} \mathbf{X}_{1,i}^T \mathbf{W}_{1,i} \mathbf{Y}. \end{aligned}$$

In Definition 4.2, we utilize the equidistancy of the lattice spanned by X_1, X_2 to estimate the i-th row or j-th column of $\widehat{\mathbf{M}}$ in a single step. This feature is called *functional smoother* or FDCS (functional double conditional smoothing) in Chapter 3. Further, we can avoid redundant computation by noting that the non-zero weights are the same for all rows (or columns) in the interior region. The elements of $\widehat{\mathbf{M}}$ are given by

$$\left\{\widehat{\mathbf{M}}\right\}_{i,j} = \widehat{m}^{(\nu_1,\nu_2)}(x_{1,i}, x_{2,j}),$$

and the intermediate results are

$$\left\{\widehat{\mathbf{M}}_1\right\}_{i,j} = \widehat{m}^{(\nu_1)}(x_{1,i}|x_{2,j}) \quad \text{and} \quad \left\{\widehat{\mathbf{M}}_2\right\}_{i,j} = \widehat{m}^{(\nu_2)}(x_{2,j}|x_{1,i}),$$

Theorem 4.1. ***(Equivalency to Bivariate Local Regression).*** *Under the conditions A1-A5, the DCS-estimators from Definitions 4.1 and 4.2 are equivalent to the classical bivariate local polynomial estimator in* (4.2.3).

A proof to Theorem 4.1 is given in Section 4.8.1.

---

[1] In the remainder of the chapter, the dependency of the matrices $\mathbf{X}_{1,i}, \mathbf{X}_{2,j}, \mathbf{W}_{1,i}, \mathbf{W}_{2,j}$ on i or j is always assumed and hence not explicitly denoted.

4.2.3 Boundary Modification in the LP-DCS

It is a well-known result that the boundary problem of kernel regression is solved by local polynomial regression (see e.g. Fan and Gijbels, 1992, 1996). However, boundary modification in local polynomial regression might be necessary to avoid discontinuities in the bias between the boundary and interior regions. This type of boundary modification in local regression is extensively discussed by Feng (2004) and in Chapter 2.

We define the boundary region (BR) as in Chapter 3 and similar to Müller and Wang (1994). The BR contains all points in X_1 and X_2 that have a distance not greater than the bandwidth from the boundaries at 0 or 1, i.e. all points that are not in the interior region. That is, the boundary region is defined by

$$\mathrm{BR} = [0,1]^2 \setminus \{x_1 : h_1 \leq x_1 \leq 1 - h_1 \cup x_2 : h_2 \leq x_2 \leq 1 - h_2\}.$$

Due to the increased share of the BR in the total area of support in two-dimensional models compared to univariate models, the effect of the boundaries cannot be neglected without careful consideration, although local regression is used. While local polynomial methods solve the boundary problem indeed, naive truncation of weight functions might introduce discontinuities in the estimation, as discussed in Chapter 2. In the remainder of this chapter, we consider local polynomial regression with boundary modification, that is, we use the boundary modified weighting functions proposed in Chapter 2, which are based upon the boundary kernels by Müller (1991) and Müller and Wang (1994). At a boundary point $x_{\bullet,i} \in \mathrm{BR}$, for $q \in [0,1]$ and a smoothness parameter μ, these weights are

$$W_q^0(u) = (1+u)^\mu (1-u)^\mu, \quad u \in [-1,q] \tag{4.2.8}$$

$$W_q^a(u) = (1+u)^\mu (q-u)^\mu, \quad u \in [-1,q] \tag{4.2.9}$$

$$W_q^b(u) = (1+u)^\mu (q-u)^{\mu'}, \quad u \in [-1,q], \mu' = \min(1, \mu - 1), \tag{4.2.10}$$

for the right boundary. The corresponding left boundary weighting functions have support $u \in [-q,1]$. The weights in (4.2.8) are the truncated weights commonly used in local regression. The μ-smooth weights (4.2.9) and the (μ,μ')-smooth weights (4.2.10) are the boundary modification weights proposed in Chapter 2. The findings of Chapter 2 show that, despite the use of different weighting methods, the local regression estimator in the interior ($q = 1$) is the same for all of these three weighting functions. In the following, we use the $(\mu, \mu-1)$-smooth weights W_q^b corresponding to the boundary kernels proposed by Müller and Wang (1994). They provide the lowest bias among all three weighting methods, as they are centered around the estimation point (see Chapter 2). Note that all results hold for the other weighting functions as well, provided that the use of boundary weights is consistent, hence, we will not indicate the use of boundary kernels explicitely.

4.2.4 Equivalent Kernels

The asymptotic results, as well as the optimal bandwidths, include some quantities, which need to be calculated from the corresponding kernel functions. We use the method introduced in Chapter 2 to generate these corresponding kernels from the weighting function.

At first, we define the local polynomial weights from Definition 4.2 via

$$w^L_{1,i} = \left\{\nu_1! e^T_{\nu_1+1}(\mathbf{X}_1^T\mathbf{W}_1\mathbf{X}_1)^{-1}\mathbf{X}_1^T\mathbf{W}_1\right\}_i \tag{4.2.11}$$

$$w^L_{2,j} = \left\{\nu_2! e^T_{\nu_2+1}(\mathbf{X}_2^T\mathbf{W}_2\mathbf{X}_2)^{-1}\mathbf{X}_2^T\mathbf{W}_2\right\}_j, \tag{4.2.12}$$

where $\mathbf{W}$ is constructed from one of the weighting functions (4.2.8) - (4.2.10) with boundary modification if necessary. The LP-DCS can be written as weighted sum of the observations

$$\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} w^L_{1,i} w^L_{2,j} y_{i,j}. \tag{4.2.13}$$

A method for generating equivalent (boundary) kernel weights w^K to the w^L of (4.2.11), (4.2.12) is proposed in Chapter 2, the resulting kernels correspond to the kernels from Müller (1991) and Müller and Wang (1994). These equivalent kernel weights of order (k,ν) with $k=p+1$ are defined by

$$w^K_i = \frac{1}{nh^{\nu+1}} K_{q,\nu}(u_i), \quad K_{q,\nu}(u_i) = \left(\sum_{l=0}^{k-1} a_{l,\nu} u_i^l\right) W_q(x_i) \tag{4.2.14}$$

for $u_i = (x_i - x)/h$. The coefficients $a_{l,\nu}$ are obtained from

$$\mathbf{N}_{p,q} a_\nu = \nu! e_{\nu+1}, \tag{4.2.15}$$

where $\mathbf{N}_{p,q}$ is the $p \times p$-matrix of moments of $W_q(u)$ and $a_\nu = (a_{0,\nu}, a_{1,\nu}, \ldots, a_{k-1,\nu})$. Again, the weighting functions $W_q(u)$ can be any of the three boundary modification weights (4.2.8) - (4.2.10). We establish the equivalency between w^L and w^K by

Theorem 4.2. ***(Equivalency of Local and Kernel Regression).*** *Let $w^L_{1,i}$, $w^L_{2,j}$ be the local polynomial weights defined by* (4.2.11) *and* (4.2.12) *and $w^K_{1,i}$, $w^K_{2,j}$ the kernel weights defined by* (4.2.14). *Under Assumptions A1 - A4, $w^L_{1,i}w^L_{2,j}$ and $w^K_{1,i}w^K_{2,j}$ are equivalent in the sense that*

$$\lim_{n_1,n_2\to\infty} \sup_{\substack{1\le i\le n_1\\ 1\le j\le n_2}} \left| \frac{w^L_{1,i}w^L_{2,j}}{w^K_{1,i}w^K_{2,j}} - 1 \right| = 0 \quad \textit{defining} \quad \frac{0}{0} \equiv 1.$$

A helpful intermediate result from the proof of Theorem 4.2 given in 4.8.2 , is the following approximation for w^L:

$$\nu! e_{\nu+1}^T (\mathbf{X}^T\mathbf{W}, \mathbf{X})^{-1}\mathbf{X}^T\mathbf{W} = \frac{1}{nh^{\nu_1+1}} K_{q,\nu}\left(\frac{x_i - x}{h}\right) + O\left(\frac{1}{nh^{\nu+1}}\right),$$

for $\mathbf{X} = \begin{bmatrix} 1 & (x_i - x)^1 & \dots & (x_i - x)^p \end{bmatrix}$.

4.3 Bandwidth Selection for the LP-DCS

4.3.1 Asymptotic Bias and Variance

Asymptotic expressions for the bias and variance of the proposed local polynomial estimator for functional surfaces on a lattice can be derived by attributing two-dimensional estimators to one-dimensional estimators, which is a main advantage of the DCS. The first step of the DCS from Definition 4.2 forms a functional time series on its own; it is a sequence of univariate time series. Hence, taking the expectation of (4.2.13), we yield

$$\begin{aligned}\mathbb{E}\left\{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} w_{1,i}^L w_{2,j}^L y_{i,j}\right\} &= \sum_{i=1}^{n_1} w_{1,i}^L \mathbb{E}\left\{\sum_{j=1}^{n_2} w_{2,j}^L y_{i,j}\right\} \\ &= \sum_{i=1}^{n_1} w_{1,i}^L \mathbb{E}\left\{\widehat{m}^{(\nu_2)}(x_2|x_{1,i})\right\},\end{aligned} \tag{4.3.1}$$

where $\widehat{m}^{(\nu_2)}(x_2|x_{1,i})$ denotes the estimator of the ν_2-th derivative of $m(x_1, x_2)$ for fixed x_1, i.e. for the i-th row of $y_{i,j}$.

The expectation and variance of local polynomial estimators are standard results in nonparametric estimation theory and can be found e.g. in Ruppert and Wand (1994) or Fan and Gijbels (1996). At in interior point, the expectation of $\widehat{m}^{(\nu_2)}(x_2|x_{1,i})$ is

$$\begin{aligned}\mathbb{E}\left\{\widehat{m}^{(\nu_2)}(x_2|x_{1,i})\right\} &= \mathbb{E}\left\{\nu_2! e_{\nu_2+1}^T (\mathbf{X}_2^T\mathbf{W}_2\mathbf{X}_2)^{-1}\mathbf{X}_2^T\mathbf{W}_2 y_{i,\cdot}^T\right\} \\ &= m^{(\nu_2)}(x_2|x_{1,i}) + S_{2|i} + R_{2|i},\end{aligned}$$

with the Taylor series remainder R_2 and $S_{2|1}$ the leading term of the expansion with

$$\begin{aligned}S_{2|i} &= \nu_2! e_{\nu_2+1}^T (\mathbf{X}_2^T\mathbf{W}_2\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}_2 \begin{pmatrix} (x_{2,1} - x_2)^{p_2+1} \\ \vdots \\ (x_{2,n_2} - x_2)^{p_2+1}\end{pmatrix} \frac{m^{(p_2+1)}(x_2|x_{1,i})}{(p_2+1)!} \\ &= \sum_{j=1}^{n_2} w_{2,j}^L (x_{2,j} - x_2)^{p_2+1} \frac{m^{(p_2+1)}(x_2|x_{1,i})}{(p_2+1)!},\end{aligned} \tag{4.3.2}$$

for the expectation of $\widehat{m}(\nu_1)(x_1, x_2)$, a similar result holds. From (4.3.1), the expectation of the regression surface or any of its derivatives is

$$\begin{aligned}\mathbb{E}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} &= \nu_1! e_{\nu_1+1}^T(\mathbf{X}_1^T\mathbf{W}_1\mathbf{X}_1)^{-1}\mathbf{X}_1^T\mathbf{W}_1\mathbb{E}\left\{\widehat{m}^{(\nu_2)}(x_2|x_1)\right\}\\ &= m^{(\nu_1,\nu_2)}(x_1,x_2) + S_1 + S_2 + R.\end{aligned}$$

Again, R is the Taylor series remainder and S_1, S_2 are the the leading term of the expansion conditional on i or j respectively, with

$$S_1 = \sum_{i=1}^{n_1} w_{1,i}^L \frac{(x_{1,i}-x_1)^{p_1+1}}{(p_1+1)!} m^{(p_1+1,\nu_2)}(x_1,x_2), \tag{4.3.3}$$

$$S_2 = \sum_{j=1}^{n_2} w_{2,j}^L \frac{(x_{2,j}-x_2)^{p_2+1}}{(p_2+1)!} m^{(\nu_1,p_2+1)}(x_1,x_2). \tag{4.3.4}$$

From Theorem 4.2, we can attribute the integral approximation of the local polynomial estimator to that of the kernel estimator using equivalent kernels. Then, the integral appoximation of the expectation is

$$\begin{aligned}\mathbb{E}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} = m^{(\nu_1,\nu_2)}(x_1,x_2) &+ \tilde{\mu}_{p_1+1}(K_1)h_1^{p_1-\nu_1+1}m^{(p_1+1,\nu_2)}(x_1,x_2)\\ &+ \tilde{\mu}_{p_2+1}(K_2)h_2^{p_2-\nu_2+1}m^{(\nu_1,p_2+1)}(x_1,x_2)\\ &+ O\left(h_1^{p_1-\nu_1+1}\right) + O\left(h_1^{p_1-\nu_1+1}\right) +\\ &+ O\left(\frac{1}{n_1h_1^{\nu_1}h_2^{\nu_2}}\right) + O\left(\frac{1}{n_2h_1^{\nu_1}h_2^{\nu_2}}\right)\end{aligned} \tag{4.3.5}$$

where $\tilde{\mu}_r(K) = \int K_{q,\nu}(u)u^r/r!\,\mathrm{d}u$.

Similar to the univariate expectation, the univariate variance is a standard result in local regression theory. The asymptotic variance under dependent errors is given by Altman (1990) for the univariate case and in the bivariate case by Feng (2013) for the kernel regression DCS. Under the more general Assumption A4 of dependent error terms we the variance is given by

$$\operatorname{var}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} = \sum_{i=1}^{n_1}\sum_{i^*=1}^{n_1}\sum_{j=1}^{n_2}\sum_{j^*=1}^{n_2} w_{1,i}^L w_{1,i^*}^L w_{2,j}^L w_{2,j^*}^L \operatorname{cov}(\varepsilon_{i,j},\varepsilon_{i^*,j^*}). \tag{4.3.6}$$

The following approximation is proved in Section 4.8.3

$$\operatorname{var}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} = \frac{\sum\limits_{s=-\infty}^{\infty}\sum\limits_{t=-\infty}^{\infty}\gamma(s,t)}{n_1n_2h_1^{2\nu_1+1}h_2^{2\nu_2+1}}\left[R(K_1)R(K_2)+o(1)\right]. \tag{4.3.7}$$

The spatial autocovariance function γ is given by

$$\gamma(s,t) = \operatorname{cov}(\varepsilon_{i,j},\varepsilon_{i+s,j+t}), \tag{4.3.8}$$

and the kernel roughness is $R(K) = \int_{-1}^{1} K^2(u)\,\mathrm{d}u$.

4.3.2 Asymptotic Optimal Bandwidths

The optimal bandwidths for the local polynomial regression are found by minimization of the AMISE. For simplification of the findings, we restrict the polynomial orders by an addendum to Assumption A5:

A5′ : The orders of the polynomial regression p_1, p_2 are chosen such that $p_1 - \nu_1 = p_2 - \nu_2 = \delta$, with δ odd.

Let $c_f = \sum_s \sum_t \gamma(s,t)$ be the sum of autocovariances of the lattice process $\varepsilon_{i,j}$ in (4.2.1) and $R(K)$, $\tilde{\mu}_j(K)$ defined as above. Further, define the integrals over the products of the regression surfaces or its respective derivatives by

$$I_{ij} = \int_0^1 \int_0^1 m^{(r_i)}(x_1,x_2) m^{(r_j)}(x_1,x_2)\,\mathrm{d}x_1\,\mathrm{d}x_2 \tag{4.3.9}$$
$$\text{with} \quad r = \{(p_1+1,\nu_2),(\nu_1,p_2+1)\}.$$

Using (4.3.5) and (4.3.7), the (asymptotic) MISE is

$$\begin{aligned}
\operatorname{MISE}(\widehat{m}^{(\nu_1,\nu_2)}, h_1, h_2) &= \Big[\mathbb{E}\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\} - \widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\Big]^2 \\
&\quad + \operatorname{var}\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\} \\
&= \operatorname{AMISE}(\widehat{m}^{(\nu_1,\nu_2)}, h_1, h_2) \\
&\quad + O\left(\frac{1}{n_1 h_1^{\nu_1} h_2^{\nu_2}}\right) + O\left(\frac{1}{n_2 h_1^{\nu_1} h_2^{\nu_2}}\right) \\
&\quad + O\left(\frac{1}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}}\right) \\
&\quad + O\left(h_1^{p_1-\nu_1+1} + h_2^{p_2-\nu_2+1}\right), \\
\operatorname{AMISE}(h_1,h_2) &= \tilde{\mu}^2_{p_1+1}(K_1) h_1^{2(\delta+1)} I_{11} + \tilde{\mu}^2_{p_2+1}(K_2) h_2^{2(\delta+1)} I_{22} \\
&\quad + 2\tilde{\mu}_{p_1+1}(K_1)\tilde{\mu}_{p_2+1}(K_2) h_1^{\delta+1} h_2^{\delta+1} I_{12} \\
&\quad + \frac{R(K_1)R(K_2)}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} c_f.
\end{aligned} \tag{4.3.10}$$

Optimal bandwidths are the bandwidths h_1, h_2 which are the joint minimizers of (4.3.10). They are given by:

Proposition 4.1. ***(Asymptotic Optimal Bandwidths).*** *Let conditions A1-A4 and A5′ hold. Then, the AMISE minimizing, asymptotic optimal bandwidths $h_{1,A}, h_{2,A}$ for estimation of $m^{(\nu_1,\nu_2)}(x_1,x_2)$ are given by the equations*

$$h_{1,A} = \left[\frac{2\nu_1+1}{2(\delta+1)} \frac{R(K_1)R(K_2)\,c_f}{n_1 n_2 A_1^{2\nu_2+1} C_1}\right]^{\frac{1}{2(\delta+\nu_1+\nu_2+2)}}, \tag{4.3.11}$$
$$C_1 = \left[\tilde{\mu}^2_{p_1+1}(K_1) I_{11} + \tilde{\mu}_{p_1+1}(K_1)\tilde{\mu}_{p_2+1}(K_2) I_{12} A_1^{\delta+1}\right]$$

and

$$\begin{aligned} h_{2,A} &= \left[\frac{2\nu_2+1}{2(\delta+1)}\frac{R(K_1)R(K_2)\,c_f}{n_1 n_2 A_2^{2\nu_1+1} C_2}\right]^{\frac{1}{2(\delta+\nu_1+\nu_2+2)}}, \\ C_2 &= \left[\tilde{\mu}^2_{p_2+1}(K_2)I_{22} + \tilde{\mu}_{p_1+1}(K_1)\tilde{\mu}_{p_2+1}(K_2)I_{12}A_2^{\delta+1}\right]. \end{aligned}$$

The relation between h_1 and h_2 is given by the relation factor A, where

$$\begin{aligned} & h_1 = A_1 h_2,\ h_2 = A_2 h_1,\ A_1 = A_2^{-1} \\ & A_1 = \left(\frac{\tilde{\mu}(K_2)}{\tilde{\mu}(K_1)}\left[\frac{I_{12}}{I_{11}}\frac{(\nu_1-\nu_2)}{(2\nu_2+1)} \pm \left(\frac{I_{12}^2}{I_{11}^2}\frac{(\nu_1-\nu_2)^2}{(2\nu_2+1)^2} + \frac{I_{22}}{I_{11}}\frac{(2\nu_1+1)}{(2\nu_2+1)}\right)^{\frac{1}{2}}\right]\right)^{\frac{1}{\delta+1}}, \end{aligned} \qquad (4.3.12)$$

where the sign in (4.3.12) *is chosen such that $A_1 > 0$ and hence $A_2 > 0$.*

If $\nu_1 = \nu_2$ ($\Rightarrow p_1 = p_2$ from A5′), the relation factor simplifies to $A_1 = (I_{11}/I_{22})^{2(\delta+1)}$ and for the special case of local linear estimation of the regression surface ($\nu_1 = \nu_2 = 0, p_1 = p_2 = 1$) (4.3.11) reduces to

$$h_1 = \left(\frac{R^2(K)\,c_f}{4n_1 n_2\,\tilde{\mu}^2(K)\left(\frac{I_{11}}{I_{22}}\right)^{\frac{1}{4}}\left[I_{11} + I_{12}\left(\frac{I_{11}}{I_{22}}\right)^{\frac{1}{2}}\right]}\right)^{\frac{1}{6}},$$

which is equal to the bandwidth for the regression surface under kernel regression in Section 3.4. The formula for h_2 is analogously.

The optimal bandwidths are of order $O\left(n^{-1/(2\delta+2\nu_1+2\nu_2+4)}\right)$. For $\widehat{m}^{(\nu_1,\nu_2)}$, it holds that $\mathrm{MISE} = O\left(h_A^{2(\delta+1)}\right)$. This leads to an order of convergence of $O\left(n^{-(\delta+1)/(\delta+\nu_1+\nu_2+2)}\right)$ and to a global convergence rate of $O\left(n^{-(\delta+1)/(2\delta+2\nu_1+2\nu_2+4)}\right)$. In particular, the global convergence rate of bandwidths for local linear regression of the mean surface is of the order $O\left(n^{-1/3}\right)$.

4.3.3 Bandwidth Selection by the IPI

The formulas for the optimal bandwidths of Proposition 4.1 require computation of higher order derivatives of $m(x_1, x_2)$. Although these derivatives can simply be estimated by the LP-DCS, this estimation requires some optimal bandwidths itself, which, in general, differ from those optimal for the regression surface. These auxiliary bandwidths are iteratvely obtained by an IPI-algorithm, as proposed by Gasser et al. (1991) and used by Herrmann et al. (1995) and in Chapter 3 for two-dimensional data. In this algorithm, the bandwidths $\widetilde{h}_1^{(p_1+1)}, \widetilde{h}_2^{(\nu_2)}$ and $\widetilde{h}_1^{(\nu_1)}, \widetilde{h}_2^{(p_2+1)}$ required for estimation of $\widehat{m}^{(p_1+1,\nu_1)}(x_1, x_2)$ and $\widehat{m}^{(\nu_1,p_2+1)}$ are computed from the bandwidths h_1, h_2 for estimation of $\widehat{m}(x_1, x_2)$ via an inflation method. This inflation method specifies a functional relation between the h and $\widetilde{h}$ values (see also Section 3.4.2). We use the exponential inflation method (EIM) proposed by Beran and Feng (2002a,b), as the EIM is shown to offer a better rate of convergence than the multiplicative method employed by Herrmann et al. (1995). The EIM sets $\widetilde{h} \propto h^\alpha$ instead of the multiplicative method, where $\widetilde{h} \propto n^\alpha h$. The exponent α can be defined in several ways. The choice of $\alpha = 0.5$ leads to the most stable bandwidth estimation (Beran and Feng, 2002a) and is hence a common used value, e.g., it is used in Chapter 3. However, if estimation of a derivative of $\widehat{m}(x_1, x_2)$ is considered, and hence at least one $\nu \neq 0$, alternative choices might be preferable. We select α to minimize $MISE(\widehat{m}^{(p_1+1,\nu_2)}, \widetilde{h}_1, \widetilde{h}_2)$ respective $MISE(\widehat{m}^{(\nu_1,p_2+1)}, \widetilde{h}_1, \widetilde{h}_2)$. Note that from (4.3.9), the required derivatives are of order (p_1+1, ν_2) and (ν_1, p_2+1). From Proposition 4.1 and B5′, we get

$$\begin{aligned} &h_1, h_2 \propto (n_1 n_2)^{-\frac{1}{2(\delta+\nu_1+\nu_2+2)}} \quad \text{and} \quad \widetilde{h}_1, \widetilde{h}_2 \propto (n_1 n_2)^{-\frac{1}{2(p_1+p_2+2)}} \\ &\Rightarrow \widetilde{h}_1 \propto h_1^\alpha, \widetilde{h}_2 \propto h_2^\alpha, \quad \text{with} \quad \alpha = \frac{\delta+\nu_1+\nu_2+2}{p_1+p_2+3}. \end{aligned} \tag{4.3.13}$$

For the exponent α, it holds that $0.5 < \alpha < 1$. In the special case of local linear regression of the mean surface ($\nu_1 = \nu_2 = 0, p_1 = p_2 = 1, \delta = 1$), we have $\alpha = 0.6$.

The IPI algorithm now computes the auxiliary bandwidths $\widetilde{h}_1, \widetilde{h}_2$, the integrals $\widehat{I}_{11}, \widehat{I}_{22}, \widehat{I}_{12}$ and from this, the asymptotic optimal bandwidths conditional on the iteration step $h_{1,A}, h_{2,A}$ in each step.

Proposition 4.2. ***(IPI Bandwidth Selection Algorithm).*** *Let $(\widehat{h}_{1,s}, \widehat{h}_{2,s})$ be the bandwidths for estimation of $m^{(\nu_1,\nu_2)}(x_1, x_2)$ obtained in the s-th iteration step. Then, the IPI algorithm processes as follows:*

1. *Choose initial values $(\widehat{h}_{1,0}, \widehat{h}_{2,0})$.*
2. *In the s-th iteration step:*
   i. *Define*

   $$\begin{aligned} \widetilde{h}_{1,s}^{(p_1+1)} &= c_{(p_1+1)}(\widehat{h}_{1,s-1})^\alpha, \quad & \widetilde{h}_{1,s}^{(\nu_2)} &= c_{(\nu_2)}(\widehat{h}_{1,s-1})^\alpha, \\ \widetilde{h}_{2,s}^{(p_2+1)} &= c_{(p_2+1)}(\widehat{h}_{2,s-1})^\alpha, \quad & \widetilde{h}_{2,s}^{(\nu_1)} &= c_{(\nu_1)}(\widehat{h}_{2,s-1})^\alpha, \end{aligned}$$

   ii. *Compute*

   $$\begin{aligned} &\widehat{m}^{(p_1+1,\nu_2)}(x_1, x_2) \text{ using bandwidths } \left(\widetilde{h}_{1,s}^{(p_1+1)}, \widetilde{h}_{2,s}^{(\nu_2)}\right) \text{ and} \\ &\widehat{m}^{(\nu_1,p_2+1)}(x_1, x_2) \text{ using bandwidths } \left(\widetilde{h}_{1,s}^{(\nu_1)}, \widetilde{h}_{2,s}^{(p_2+1)}\right) \end{aligned}$$

   and from this the integrals $\widehat{I}_{11}, \widehat{I}_{22}, \widehat{I}_{12}$ applying (4.3.9).

   iii. *Compute $\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)$ using bandwidths $(\widehat{h}_{1,s-1}, \widehat{h}_{2,s-1})$ and estimate the variance factor c_f from the residuals $\widehat{\varepsilon}_{i,j} = \widehat{m}(x_{1,i}, x_{2,j}) - y_{i,j}$ using an appropriate method.*

   iv. *Obtain the optimal bandwidths $\left(\widehat{h}_{1,s}, \widehat{h}_{2,s}\right)$ from Proposition 4.1.*

3. *Stop if the distance between* $(\widehat{h}_{1,s}, \widehat{h}_{2,s})$ *and* $(\widehat{h}_{1,s-1}, \widehat{h}_{2,s-1})$ *is smaller than some desired threshold. Otherwise, return to 2.*

The bandwidths yielded in step 3 *are then called the optimal bandwidths.*

If $\nu_1 \neq 0$ or $\nu_2 \neq 0$, the optimal bandwidths for estimation of $\widehat{m}(x_1, x_2)$ are not obtained in the IPI-process. We suggest to calculate c_f in advance by applying the IPI for the regression surface previously to bandwidth selection of the desired derivative $\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)$.

4.4 Spatial Error Structure

4.4.1 Definition of the Spatial ARMA

In most application cases, whether financial econometrics or environmental sciences, dependency in the error terms is a more realistic approach than assuming an iid. distribution of the errors. From Assumption A4, the error terms or innovations $\varepsilon_{i,j}$ of model (4.2.1) are assumed to follow a certain dependency structure with short memory. This ensures the dependency structure of the errors to be finite and influences the asymptotical optimal bandwidths of Proposition 4.1 only through the variance factors c_f. For long memory dependency, more sophisticated bandwidth formulas are used, as now the long-memory parameters directly influence the bandwidth (see Feng et al., 2021a). In the following, we study an SARMA process to model the errors $\varepsilon_{i,j}$ of (4.2.1).

The following definition of a causal SARMA process is commonly used:

Definition 4.3. ***(SARMA-Process).*** *Let* $\eta_{i,j}$ *be an iid. random field, with zero mean and variance* σ_η^2 *and let the polynomials* $\phi(z_1, z_2)$ *and* $\psi(z_1, z_2)$ *have all roots outside the unit circle. The process* $\varepsilon_{i,j}$ *is called an* $\mathrm{SARMA}\left((r_1, r_2), (q_1, q_2)\right)$ *process, if it fulfills the equation*

$$\phi(B_1, B_2)\varepsilon_{i,j} = \psi(B_1, B_2)\eta_{i,j}, \tag{4.4.1}$$

where the lag operators are defined by $B_1\varepsilon_{i,j} = \varepsilon_{i-1,j}$ *and* $B_2\varepsilon_{i,j} = \varepsilon_{i,j-1}$ *and*

$$\phi(z_1, z_2) = \sum_{s=0}^{r_1}\sum_{t=0}^{r_2} \phi_{s,t} z_1^s z_2^t, \ \psi(z_1, z_2) = \sum_{s=0}^{q_1}\sum_{t=0}^{q_2} \psi_{s,t} z_1^s z_2^t,$$

with $\phi_{0,0} = 1$*,* $\psi_{0,0} = 1$.

The autocovariance generating function $g_\varepsilon(z_1, z_2)$ and spectral density $f(\omega_1, \omega_2)$ of the general SARMA process in Definition 4.3 given, for instance, by Martin (1996), are

$$g_\varepsilon(z_1, z_2) = \sum_{s=-\infty}^{\infty}\sum_{t=-\infty}^{\infty} \gamma(s,t) z_1^s z_2^t = \sigma_\eta^2 \frac{\psi(z_1, z_2)\psi(z_1^{-1}, z_2^{-1})}{\phi(z_1, z_2)\phi(z_1^{-1}, z_2^{-1})},$$

$$f(\omega_1, \omega_2) = \left(\frac{1}{2\pi}\right)^2 \sum_{s=-\infty}^{\infty}\sum_{t=-\infty}^{\infty} \gamma(s,t) e^{-\mathrm{i}\omega_1 s - \mathrm{i}\omega_2 t} = \left(\frac{1}{2\pi}\right)^2 g_\varepsilon\left(e^{-\mathrm{i}\omega_1 s}, e^{-\mathrm{i}\omega_2 t}\right)$$

where $\gamma(s,t)$ is the autocovariance function. The covariance factor c_f can be computed easily given the SARMA parameters from the spectral density

$$c_f = 4\pi^2 f(0,0) = 4\pi^2 g_\varepsilon(1,1) = \sigma_\eta^2 \frac{\psi^2(1,1)}{\phi^2(1,1)}. \tag{4.4.2}$$

We propose the following matrix notation for observations of the SARMA process from Definition 4.3, which allows for a direct visual assignment of the coefficients to the respective lags:

Definition 4.4. ***(Matrix Notation for SARMA-Processes).*** *Let* $\mathbf{E}$ *be the* $(n_1 \times n_2)$*-matrix containing the observed values of* $\varepsilon_{i,j}$*,* $\mathbf{Z}$ *be the* $(n_1 \times n_2)$*-matrix containing the innovations* $\xi_{i,j}$*. We define the submatrices* $\mathbf{E}^*(i^*,j^*), \mathbf{Z}^*(i^*,j^*)$ *of lagged values at* (i,j) *by*

$$\mathbf{E}^*(i^*,j^*) = \{E_{i,j}\}_{\substack{i^* \geq i \geq i^*-p_1 \\ j^* \geq j \geq j^*-p_2}}, \quad \mathbf{Z}^*(i^*,j^*) = \{Z_{i,j}\}_{\substack{i^* \geq i \geq i^*-q_1 \\ j^* \geq j \geq j^*-q_2}}.$$

For $\max(p_1,q_1) < i \leq n_1$ *and* $\max(p_2,q_2) < j \leq n_2$*, the SARMA-process from* (4.4.1) *can be written as*

$$\operatorname{vec}^T(\boldsymbol{\phi}) \operatorname{vec}(\mathbf{E}^*(i,j)) = \operatorname{vec}^T(\boldsymbol{\psi}) \operatorname{vec}(\mathbf{Z}^*(i,j))$$

where the coefficient matrices are

$$\boldsymbol{\phi} = \begin{pmatrix} \phi_{0,0} & \cdots & \phi_{0,p_2} \\ \vdots & \ddots & \\ \phi_{p_1,0} & & \phi_{p_1,p_2} \end{pmatrix}, \quad \boldsymbol{\psi} = \begin{pmatrix} \psi_{0,0} & \cdots & \psi_{0,q_2} \\ \vdots & \ddots & \\ \psi_{q_1,0} & & \psi_{q_1,q_2} \end{pmatrix}$$

with $\phi_{0,0} = 1, \psi_{0,0} = 1$

4.4.2 Estimation of the SARMA

In general, the various methods for parameter estimation of univariate ARMA models can be easily extended to estimation of $\boldsymbol{\phi}$, $\boldsymbol{\psi}$, and σ^2 of the SARMA in Definition 4.4. Most widely used procedures are maximum likelihood (ML) estimation or least squares estimation from the residuals. ML estimation for a stationary SARMA is e.g. investigated by Yao and Brockwell (2006) based on the innovations algorithm proposed by Brockwell and Davis (1991). Illig and Truong-Van (2006) propose an estimation procedure for the AR-part as well as a method for selection of the $\mathrm{SARMA}\,(r,q)$ orders $r = (r_1, r_2)$ and $q = (q_1, q_2)$. For none of these methods, closed-form solutions exist and numerical optimization is required. In the spatial framework under consideration, where large data sets are observed, this will lead to an increased computation time. In the worst case, the efficiency increase by the DCS in the nonparametric estimation of the mean surface is dominated by the numerical estimation of the error terms. A much faster approach is a two-dimensional version of the algorithm by Hannan and Rissanen (1982), who use a

Yule-Walker estimate of the AR-part of the SARMA for the initial estimation of the innovations and a linear regression for estimation of the SARMA-parameters. However, this method provided less good fits in our simulations compared to other numerical methods. In the following, we will propose estimation strategies for two special cases of the SARMA; a separable process and a spatial AR-process (SAR).

In the case of a separable process, the polynomials in Definition 4.3 reduce to

$$\phi(z_1, z_2) = \phi_1(z_1)\phi_2(z_2) \quad \text{and} \quad \psi(z_1, z_2) = \psi_1(z_1)\psi_2(z_2)$$

$$\phi_1(z_1) = 1 - \sum_{s=1}^{p_1} \phi_{s,0} z_1^s, \quad \phi_2(z_2) = 1 - \sum_{t=1}^{p_2} \phi_{0,t} z_1^t, \tag{4.4.3}$$

$$\psi_1(z_1) = 1 + \sum_{s=1}^{q_1} \psi_{s,0} z_1^s, \quad \psi_2(z_2) = 1 + \sum_{t=1}^{q_2} \psi_{0,t} z_1^t. \tag{4.4.4}$$

In this case, the spatial process is just the product of two univariate ARMA processes and the formulas for the autocovariance and spectral density reduce to a separable form. The parameter matrices are the products of the parameter vectors

$$\begin{aligned} \boldsymbol{\phi} &= (1, \phi_{1,0}, \phi_{2,0}, \ldots, \phi_{p_1,0})^T (1, \phi_{0,1}, \phi_{0,2}, \ldots, \phi_{0,p_2}) \\ \boldsymbol{\psi} &= (1, \psi_{1,0}, \psi_{2,0}, \ldots, \psi_{q_1,0})^T (1, \psi_{0,1}, \psi_{0,2}, \ldots, \psi_{0,q_2}). \end{aligned}$$

and the variance factor can be computed by

$$c_f = 4\pi^2 \sigma_\eta^2 \frac{\psi_1^2(1)\psi_2^2(1)}{\phi_1^2(1)\phi_2^2(1)}.$$

Under a separable SARMA process, estimation of the parameters can be reduced to estimation of two univariate ARMA-processes (Martin, 1979) in the two directions, which bears a resemblance to the DCS procedure. Given a matrix $\mathbf{E}$ of observations of the process $\varepsilon_{i,j}$, the ARMA coefficients can be estimated in two ways. Either, every row and column of $\mathbf{E}$ can be treated as a random sample of the respective ARMA model and individually estimated. This approach was discussed by Beran et al. (2009) for the SARMA-part of long-memory SFARIMA model. The estimates $\widehat{\phi}_1, \widehat{\psi}_1$ would then be the averages over the parameters $\widetilde{\phi}_1(j), \widetilde{\psi}_1(j)$ estimated from $\{\mathbf{E}\}_{1 \le i \le n_1, j}$ and $\widehat{\phi}_2, \widehat{\psi}_2$ the averages over $\widetilde{\phi}_2(i), \widetilde{\psi}_2(i)$ estimated from $\{\mathbf{E}\}_{i, 1 \le j \le n_2}$. The second approach brings the averaging forward and treats $\mathrm{vec}(\mathbf{E})$ as the only random sample of the column-wise time series and $\mathrm{vec}(\mathbf{E}^T)$ as the only random sample of the row-wise time series. Hence, the parameters $\widehat{\phi}_1, \widehat{\psi}_1$ can be directly estimated from $\mathrm{vec}(\mathbf{E})$ and $\widehat{\phi}_2, \widehat{\psi}_2$ from $\mathrm{vec}(\mathbf{E}^T)$. Although both approaches are suitable for estimation of the parameters of a separable SARMA process, it depends on the circumstances, which one to prefer. We found the second one more precise in our simulations, however, the first one might be faster for very large data sets, depending on the estimation algorithm used.

For $q_1 = q_2 = 0$, the SARMA of Definition 4.3 reduces to an SAR process with order (r_1, r_2). Unlike the SARMA, there exist explicit formulas for estimation of the parameters of an SAR process in form of the (spatial) Yule-Walker equations (see Ha and Newton, 1993; Illig and Truong-Van, 2006). We use the definition of (4.3.8) for the autocovariance function $\gamma(s,t)$ of the SAR process. The spatial Yule-Walker equation for the $\mathrm{SAR}\,(r_1, r_2)$ is then

$$\mathbf{\Gamma}\ \mathrm{vec}(\boldsymbol{\phi}) = 0$$

where $\mathbf{\Gamma}$ denotes the full autocovariance matrix:

$$\mathbf{\Gamma} = \begin{pmatrix} \gamma(0,0) & \gamma(1,0) & \dots & \gamma(r_1,0) & \gamma(0,1) & \dots & \gamma(r_1,r_2) \\ \gamma(1,0) & \gamma(0,0) & & & & & \gamma(r_1-1,r_2) \\ \vdots & & \ddots & & & & \vdots \\ \gamma(r_1,r_2) & \gamma(r_1-1,r_2) & \dots & & & & \gamma(0,0) \end{pmatrix}$$

Ha and Newton (1993) proposed the following bias-corrected sample autocovariance function:

$$\widehat{\gamma}(s,t) = \frac{n_1 n_2}{(n_1-s)(n_2-t)} \sum_{i=1}^{n_1-s} \sum_{j=1}^{n_2-t} \varepsilon_{i,j}\varepsilon_{i+s,j+t}$$

$$\widehat{\gamma}(s,-t) = \frac{n_1 n_2}{(n_1-s)(n_2-t)} \sum_{i=1}^{n_1-s} \sum_{j=t+1}^{n_2} \varepsilon_{i,j}\varepsilon_{i+s,j-t},$$

with the relations $\widehat{\gamma}(-s,-t) = \widehat{\gamma}(s,t)$ and $\widehat{\gamma}(-s,t) = \widehat{\gamma}(s,-t)$.

4.5 Simulation Study

The proposed algorithm is assessed at the example of three gaussian functions. We study the distribution of the selected bandwidths and of the variance factor c_f under dependent errors following an SARMA model. Let $N(\mu, \Sigma)$ denote the bivariate normal distribution with mean vector μ and covariance matrix Σ, then the example surface functions are

$$m_1 \sim N\left(\begin{pmatrix}0.5\\0.5\end{pmatrix}, \begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}\right) \tag{4.5.1}$$

$$m_2 \sim N\left(\begin{pmatrix}0.5\\0.3\end{pmatrix}, \begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}\right) + N\left(\begin{pmatrix}0.2\\0.8\end{pmatrix}, \begin{pmatrix}0.05 & 0\\0 & 0.05\end{pmatrix}\right) \tag{4.5.2}$$

$$m_3 \sim N\left(\begin{pmatrix}0.25\\0.75\end{pmatrix}, \begin{pmatrix}0.01 & 0\\0 & -0.1\end{pmatrix}\right) + N\left(\begin{pmatrix}0.75\\0.5\end{pmatrix}, \begin{pmatrix}0.01 & 0\\0 & -0.1\end{pmatrix}\right), \tag{4.5.3}$$

where (4.5.1) constitutes a symmetric single peak, (4.5.2) an asymmetric double peak and (4.5.3) forms two ridges intended to resemble the volatility surfaces found in Section 4.6. The error structure is chosen such that $\varepsilon_{i,j}$ follows a separable $\mathrm{SARMA}\left((1,1),(1,1)\right)$ process with coefficients

$$\boldsymbol{\phi} = \begin{pmatrix} 1 & -0.50 \\ -0.20 & 0.10 \end{pmatrix}, \ \boldsymbol{\psi} = \begin{pmatrix} 1 & 0.20 \\ 0.30 & 0.06 \end{pmatrix}, \ \sigma^2 = 0.25. \tag{4.5.4}$$

From (4.4.2), the variance coefficient for the SARMA is then $c_f = 3.8025$. Our study includes 2,500 simulations for each function where observations are simulated on a $[0,1]^2$ grid of size $n_1 = n_2 = 101$. We use a trim parameter $\lambda_1 = \lambda_2 = \lambda = 0.05$ to stabilize estimation of bandwidths. Estimation of the expectation surface is done under a correctly specified SARMA model for estimation of the error structure and misspecified model under an iid. assumption. This allows to assess the effect of not accounting for dependency in the errors.

We employ a local linear estimator for estimation of the regression surface, with the (μ, μ')-smooth boundary modification weights of (4.2.10) and $\mu = 2$. Hence, the required partial derivatives for the optimal bandwidths of Proposition 4.1 are $m^{(2,0)}(x_1, x_2)$ and $m^{(0,2)}(x_1, x_2)$, which are estimated by local linear and local cubic regression in the dimension of the second derivative. The parameters in the EIM were chosen to be $c_{(2)} = 2$, $c_{(2)} = 1$ for the exemplary functions m_1, m_2 and $c_{(2)} = c_{(2)} = 1$ for m_3. We chose to differ in the latter case, as here the structure is somehow finer than in the other cases and a too large bandwidth for derivative estimation is susceptible for oversmoothing[2]. The exponent was chosen according to (4.3.13) to $\alpha = 3/5$.

The distributions of the estimated bandwidths are displayed in Figure 4.2. The results indicate that the goodness-of-fit of bandwidth selection depends on the underlying function $m(x_1, x_2)$. Finer structures will lead to a rapid decrease in variance of the estimator making a bias in the estimation more severe. This becomes obvious from Table 4.1 where the p-values measure the deviation of the estimates from the true values. For all functions the true bandwidth is estimated with a certain confidence under the correctly specified model. Parameter tuning for further optimization of the bandwidth selection algorithm might aim at adjusting the parameters λ or c_0, c_1. However, to our knowledge, a data driven method for selection of the inflation parameters is not available, so these values can only be selected from a simulation with similar functions.

In any case the selected bandwidths for the correct model are close to the true bandwidths, while the misspecified model clearly underestimates the bandwidths in this case. This behavior depends on the values of c_f and σ^2, hence, incorrect specification of the error model might also lead to oversmoothing for different data. The original surfaces can

---

[2] For practical applications, we propose to run a pilot smoothing with manually specified bandwidths to gain insights on the functional form of the surface under consideration. Parameters of the EIM might be choosen according to these findings as long as no data-driven method for selecting this parameters exist.

Table 4.1: *Bandwidth statistics of the simulation study using SARMA errors under local polynomial regression DCS. Values are obtained from a sample of 2,500 simulations for the functions given in* (4.5.1) - (4.5.3). *The p-value corresponds to* $H_0 : h = h_{true}$.

Function	**Type**		$\mathbf{h_1}$			$\mathbf{h_2}$	
		mean	**SD**	**p**	**mean**	**SD**	**p**
m_1	true	0.23142	-	-	0.23142	-	-
	SARMA	0.23299	0.01611	0.92246	0.23345	0.01588	0.89853
	iid.	0.16120	0.01759	0.00007	0.16153	0.01768	0.00008
m_2	true	0.21443	-	-	0.24367	-	-
	SARMA	0.19555	0.01817	0.29877	0.22742	0.03027	0.59150
	iid.	0.14004	0.01375	0.00000	0.15164	0.01831	$> 1\text{E}{-}05$
m_3	true	0.08842	-	-	0.11392	-	-
	SARMA	0.09332	0.00390	0.20989	0.11729	0.00632	0.59469
	iid.	0.06745	0.00255	$> 1\text{E}{-}05$	0.08444	0.00446	$> 1\text{E}{-}05$

Table 4.2: *Summary statistics of the estimated variance coefficients in the simulation study. The true value is* 3.8025 *for all models, the p-value corresponds to* $H_0 : c_f = c_{f,true}$.

Function	**Errors**	**mean**	**SD**	**p**
m_1	SARMA	3.42496	0.25953	0.14576
	iid.	0.50153	0.01328	$> 1\text{E}{-}05$
m_2	SARMA	3.35608	0.25459	0.07952
	iid.	0.49777	0.01321	$> 1\text{E}{-}05$
m_3	SARMA	2.83352	0.21125	$> 1\text{E}{-}05$
	iid	0.45225	0.01210	$> 1\text{E}{-}05$

be found in the first column of Figure 4.3 along with the smoothed surfaces for the iid. and SARMA data.

4.6 Applications

We illustrate the proposed algorithm at the example of stock price volatility and the corresponding trading volumes observed from 2004 to 2014 covering the financial crisis around 2008, with data from German companies Allianz SE (ALV) and Siemens AG (SIE). Some exemplary smoothed surfaces are given in Figure 4.4 for ALV and Figure 4.5 for SIE. The impact of the financial crisis is clearly visible in the spike in 2008 which exists in all surface plots. Another insight we can draw from our application is that accounting for dependency in the errors leads to an increase in bandwidths for our examples (see Table 4.3).

We use stock price returns and volume data from Allianz SE and Siemens AG. All four data sets are observed from 2004-01-02 to 2014-09-30 and include 2739 days with 510

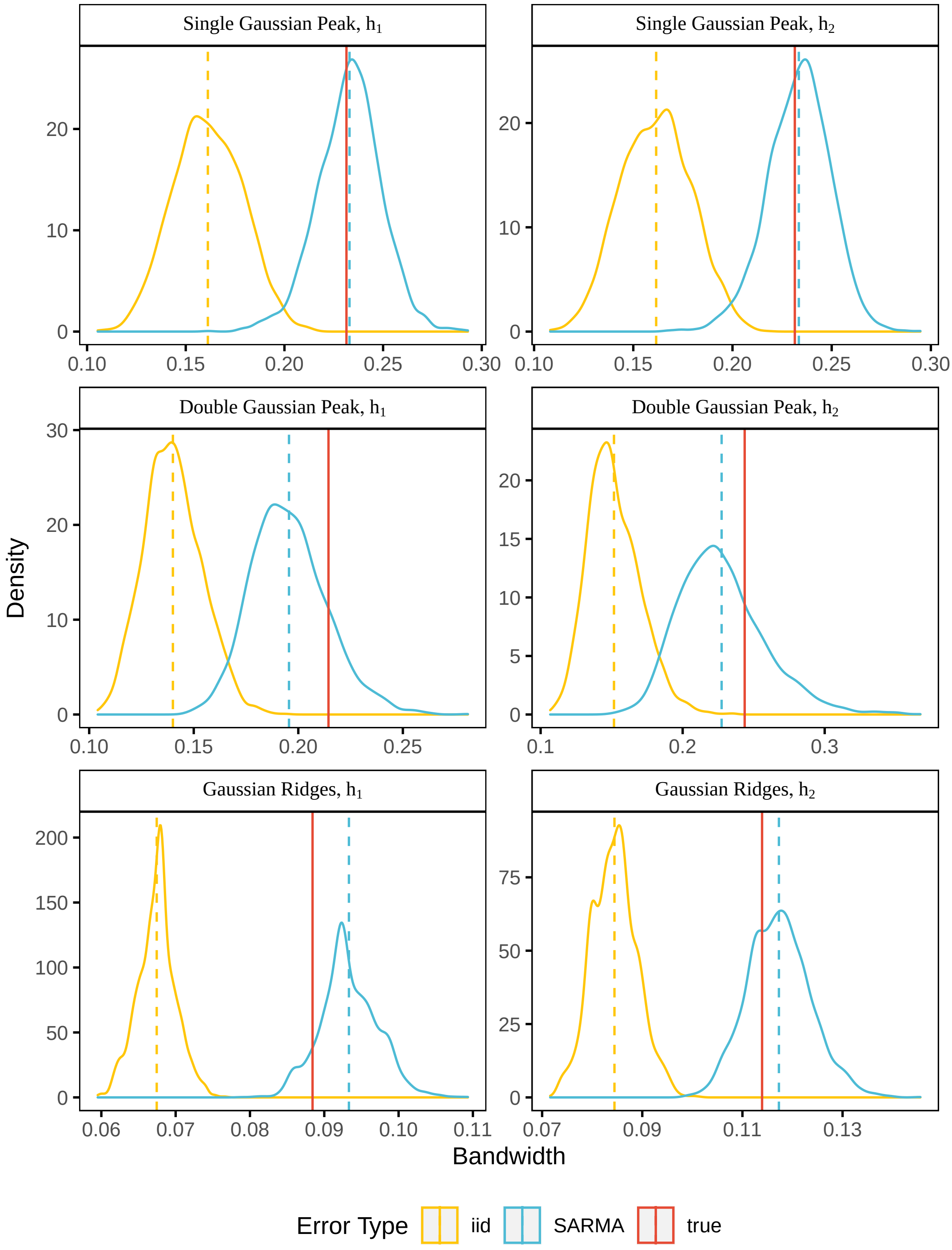


Figure 4.1: *Distribution of the bandwidth estimates* h_1*,* h_2 *obtained in the simulation study with 2,500 observations. Simulated are functions* m_1*,* m_2*,* m_3 *of* (4.5.1) - (4.5.3) *under the SARMA model* (4.5.4)*. Bandwidths are selected under an iid. and an* $\mathrm{SARMA}\,((1,1),(1,1))$ *assumption.*

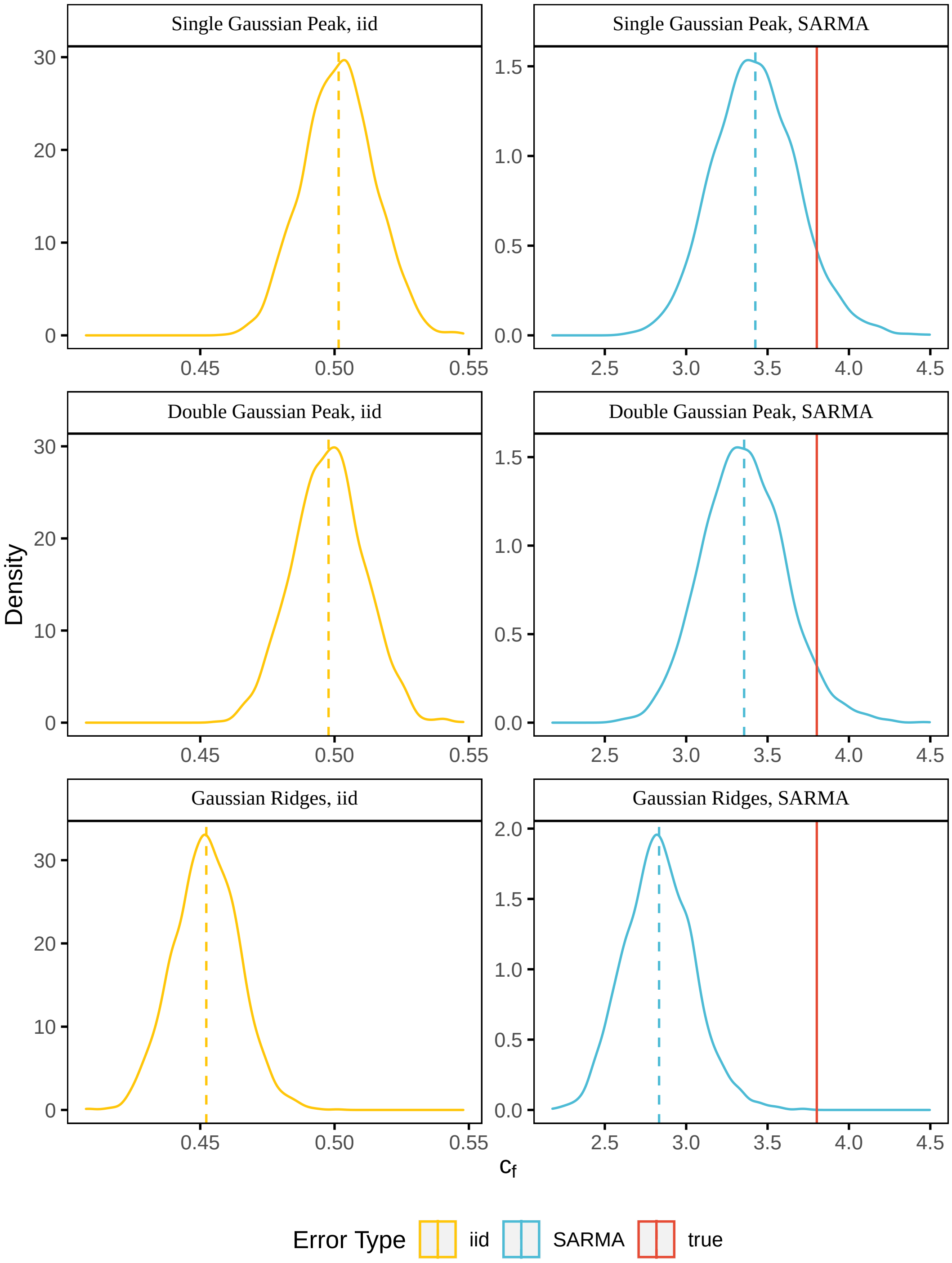


Figure 4.2: *Distribution of the estimated variance coefficients* c_f *in the simulation study with 2,500 observations. Simulated are functions* m_1, m_2, m_3 *of* (4.5.1) - (4.5.3) *under the SARMA model* (4.5.4). *Variance coefficients* c_f *are obtained under an iid. assumption and an* $\mathrm{SARMA}\,((1,1),(1,1))$ *assumption for the error terms.*

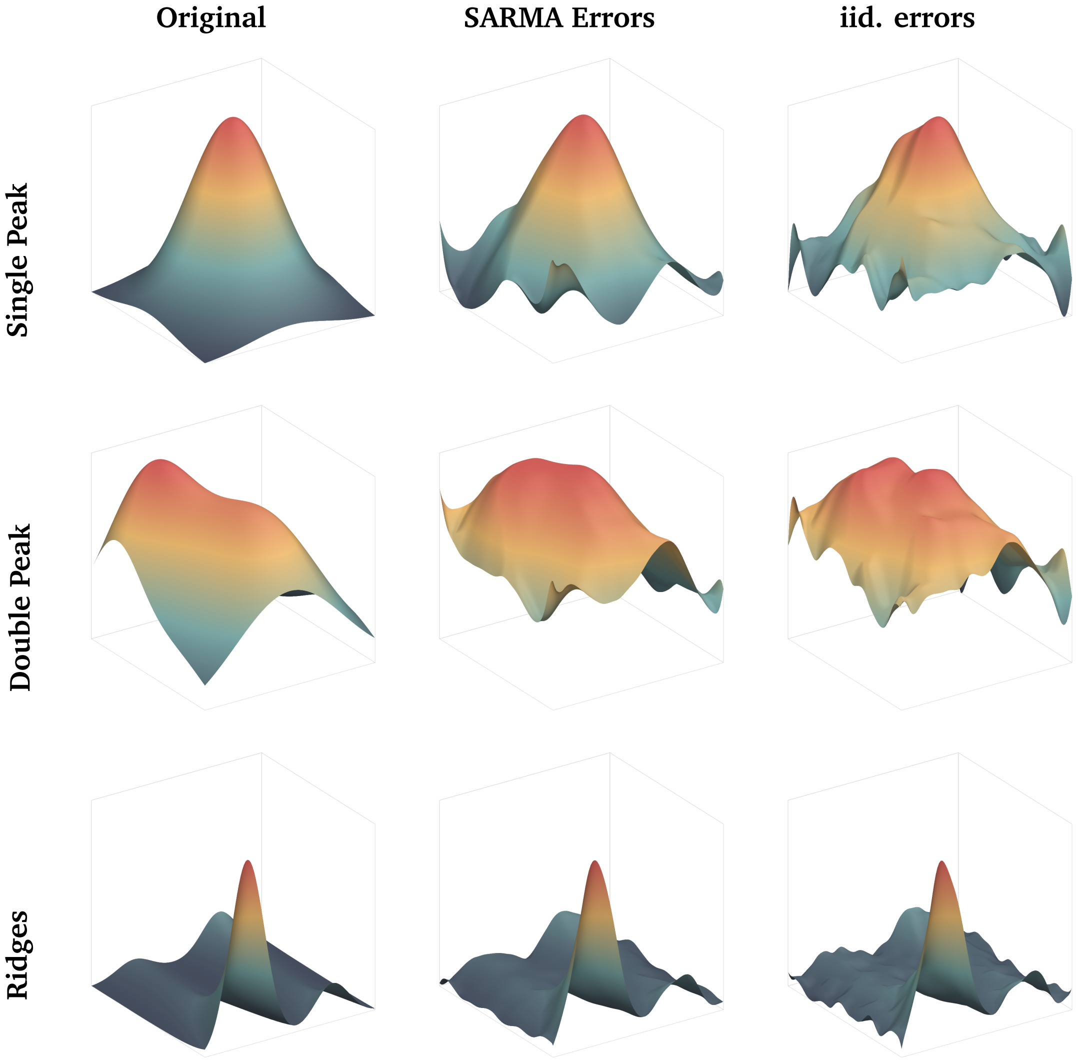


Figure 4.3: *Original and estimated surfaces for the functions* (4.5.1) - (4.5.3) *under SARMA errors from model* (4.5.4). *The following bandwidths are used:* $m_1 : (0.2384, 0.2497), m_2 : (0.2138, 0.2474), m_3 : (0.0971, 0.1245)$ *under the* $\text{SARMA}\,((1,1),(1,1))$ *assumption and* $m_1 : (0.1750, 0.1496), m_2 : (0.1647, 0.1692), m_3 : (0.0694, 0.0868)$ *under the iid. assumption (misspecification). The scale of the vertical axis might differ across the plots.*

per-minute observations leading to a total number of observations just below 1.4 million[3]. While no further preprocessing was done for the volume data, we use the squared and demeaned returns for estimation of the volatility surface. While Peitz and Feng (2015) proposed to use another nonparametric surface estimation for demeaning the returns, we found the arithmetic mean to be equally appropriate as the difference between both approaches is minor in this context. We use a similar model as in Chapter 3 or Feng (2013); Peitz and Feng (2015). Let $\overline{R}_{i,j}$ be the centralized returns, $\sigma_R(x_1, x_2)$ the spot volatility surface and $\widetilde{\eta}_{i,j}$ a random field. We model the returns by

$$\overline{R}_{i,j} = \sigma_R(x_{1,i}, x_{2,j})\widetilde{\eta}_{i,j}$$

with transformation into an additive model given by

$$\ln\left(\overline{R}^2_{i,j}\right) = \ln\left(\sigma^2_R(x_{1,i}, x_{2,j})\right) + \varepsilon_{i,j}, \tag{4.6.1}$$

where $\varepsilon_{i,j} = \ln(\widetilde{\eta}^2_{i,j})$ is assumed to be a stationary random field with short memory. Model (4.6.1) can be estimated using nonparametric regression, where $\ln\left(\sigma^2_R(x_1, x_2)\right)$ is the mean surface. The volatility surface is obtained via retransformation.

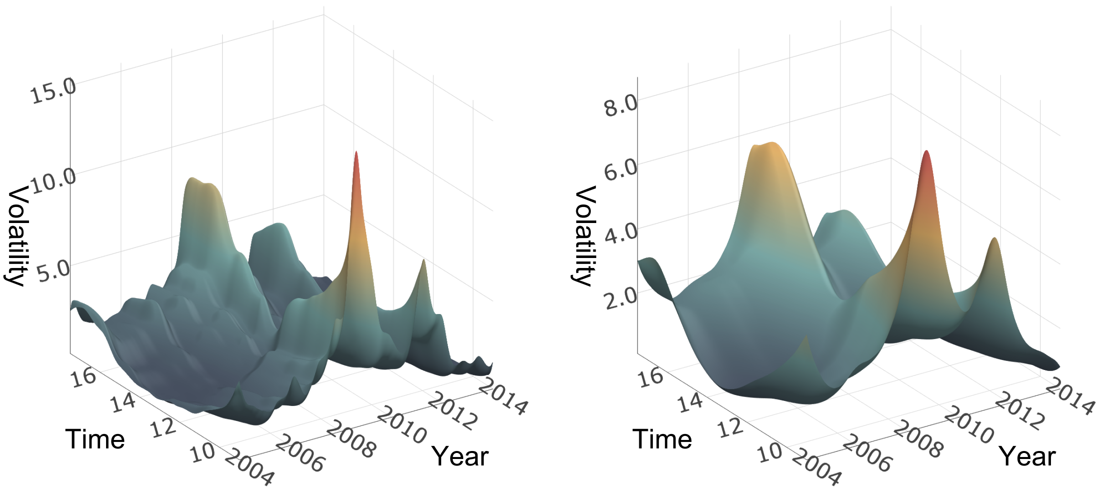
Four different approaches are used for modeling the error terms $\varepsilon_{i,j}$ of (4.6.1): iid., $\mathrm{SARMA}\,((1,1),(1,1))$, $\mathrm{SARMA}\,(r,s)$ (SARMA*) with selection of the orders r, s by minimization of the BIC (with maximum orders $(2,2)$ for the AR- and MA-part) and $\mathrm{SAR}\,(3,3)$. The resulting optimal bandwidths are displayed in Table 4.3, the estimated variance models in Table 4.4 (Allianz SE) and Table 4.5 (Siemens AG). For SARMA models with nonzero MA-part the bandwidths are substantially larger than under the assumption of iid. or SAR-error terms where the latter only exhibit a small increase to the iid. case. This leads to much smoother surface estimates in Figure 4.4 and Figure 4.5 when accounting for dependent data.

4.7 Final Remarks

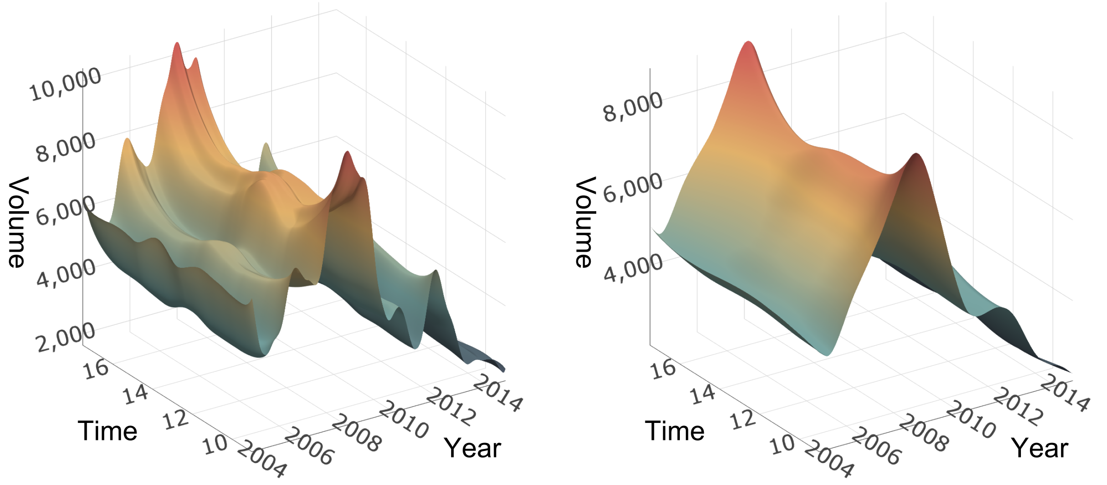
The DCS, as introduced by Feng (2013) and further developed in Chapter 3, is extended to local polynomial smoothers in this chapter. Local polynomial estimators have some clear advantages over kernel regression; they solve the boundary problem and allow for easier estimation of derivatives. These properties are utilized in the bandwidth selection procedure by an IPI algorithm where estimation of partial derivatives is necessary. We obtain optimal bandwidths in Proposition 4.1 and propose an IPI algorithms based on these optimal bandwidths in Proposition 4.2. We introduce a certain dependency structure in the semiparametric model, which is explicitely addressed by an SARMA model. Some fast estimation procedures for the SARMA parameters are suggested and a order

---

[3] The data was aggregated by calendar time sampling (CTS). Original Data was obtained from Thomson Reuters.

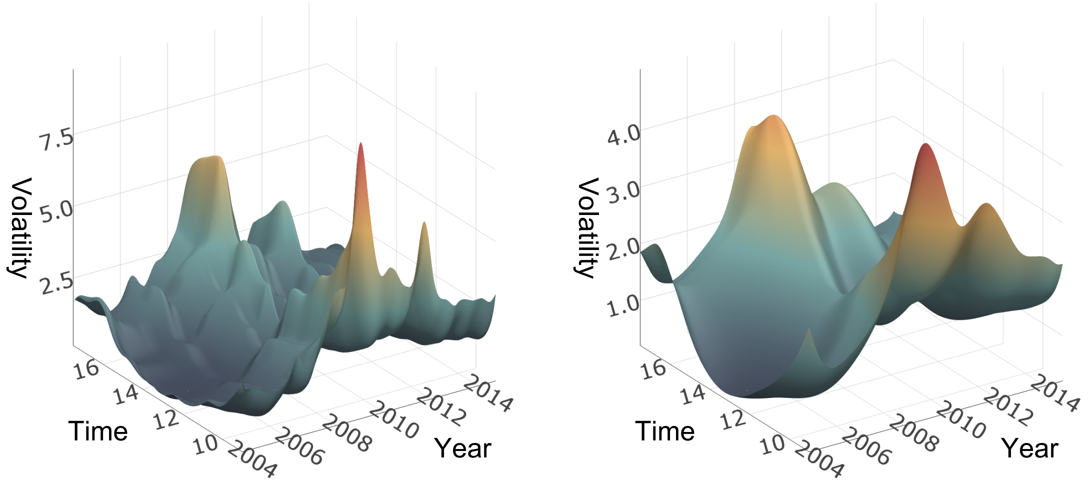


(a) Volatility, iid. errors (b) Volatility, SARMA errors



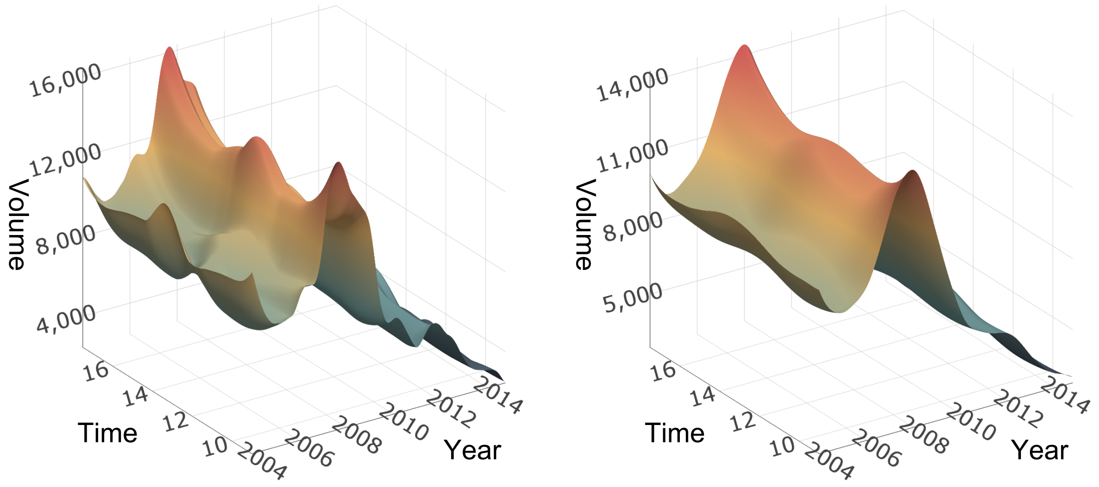
(c) Volumes, iid. errors (d) Volumes, SARMA errors

Figure 4.4: *Estimated spot volatility and trading volume surfaces of Allianz SE using the bandwidths in Table 4.3. Values of volatility are in* $1\mathrm{E}{-}04$*, of volumes in* $1\mathrm{E}{+}03$.



(a) Volatility, iid. errors

(b) Volatility, SARMA errors



(c) Volumes, iid. errors

(d) Volumes, SARMA errors

Figure 4.5: *Estimated spot volatility and trading volume surfaces of Siemens AG using the bandwidths in Table 4.3. Values of volatility are in* 1E−04, *of volumes in* 1E+03.

Table 4.3: *Bandwidths of the estimated models for spot volatility and trading volumes of Allianz SE and Siemens AG under assumption of independent and dependent errors and local polynomial regression DCS. Estimated models are iid.,* $\mathrm{SARMA}\,((1,1),(1,1))$*, SARMA* with BIC order selection and* $\mathrm{SAR}\,(1,1)$.

Data	**Err. Type**	$\mathbf{h_1}$	$\mathbf{h_2}$	c_f
Allianz SE Volatility	iid.	0.03848	0.07412	20.74
	SARMA	0.09967	0.16900	2469.84
	SARMA*	0.13684	0.25297	18358.91
	SAR	0.04833	0.09017	57.19
Allianz SE Volumes	iid.	0.05376	0.13552	1.845E+08
	SARMA	0.14757	0.30166	9.506E+09
	SARMA*	0.14930	0.30525	1.022E+10
	SAR	0.05926	0.15380	3.161E+08
Siemens AG Volatility	iid.	0.05033	0.07178	28.53
	SARMA	0.13934	0.18254	6376.02
	SARMA*	0.28156	0.34702	148123.74
	SAR	0.06004	0.08519	68.274
Siemens AG Volumes	iid.	0.06465	0.13751	6.330E+08
	SARMA	0.13894	0.29217	2.369E+10
	SARMA*	0.13992	0.29384	2.467E+10
	SAR	0.06936	0.15102	9.417E+08

selection algorithm is presented. The application to financial data indicates that incorporating dependent errors terms increase the bandwidth substantially for the data sets under consideration.

It should be emphasized that, though our algorithm is initially developed for financial applications, it is not limited to those. For any spatial data model or functional time series on a lattice which meets the assumptions, the semiparametric DCS offers a fast method of estimating the expectation surface for large data sets.

Further research on the DCS can be conducted in various directions. Crucial to the bandwidth selection algorithm is the estimation of the derivatives. These estimates are heavily influenced by the choice of the inflation parameters and inflation exponents. Refinement of the local polynomial DCS might employ a data driven choice of these inflation parameters and thus reducing both, bias and variance of the estimator.

The use of an SARMA process for modeling the error term structure might not be appropriate in all cases. In some applications, this model might be clearly misspecified or another model might suit the specific framework better. In the presence of long-memory effects in the data, the SARMA might be replaced with an appropriate model such as a spatial FARIMA model. An extension to nonparametric estimation of the variance factor would convert the semiparametric model treated in this chapter into a fully nonparametric model.

4.8 Appendix

4.8.1 Proof of Theorem 4.1

Proof. Let $\mathbf{U}_1 = (\mathbf{X}_1^T \mathbf{W}_1 \mathbf{X}_1)^{-1} \mathbf{X}_1^T \mathbf{W}_1$ and $\mathbf{U}_2 = (\mathbf{X}_2^T \mathbf{W}_2 \mathbf{X}_2)^{-1} \mathbf{X}_2^T \mathbf{W}_2$, then the local polynomial DCS estimator from Definition 4.2 expressed as a single step is $\mathbf{U}_1(\mathbf{U}_2\mathbf{Y})^T$. This can be written in terms of vectorization and Kronecker multiplication:

$$\mathbf{U}_1(\mathbf{U}_2\mathbf{Y})^T = \mathbf{U}_1\mathbf{Y}^T\mathbf{U}_2^T = (\mathbf{U}_2 \otimes \mathbf{U}_1)\text{vec}(\mathbf{Y}) \tag{4.8.1}$$

Let $\mathbf{C}, \mathbf{S}, Y$ be defined as in Section 4.2 and note that $\mathbf{C} = \mathbf{X}_1 \otimes \mathbf{X}_2$, $\mathbf{S} = \mathbf{W}_1 \otimes \mathbf{W}_2$ and $Y = \text{vec}(\mathbf{Y})$. For suitable matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$, the identity $\mathbf{AC} \otimes \mathbf{BD} = (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D})$ holds, iterated use of this identity on (4.8.1) leads to:

$$\mathbf{U}_1(\mathbf{U}_2\mathbf{Y})^T = (\mathbf{C}^T\mathbf{S}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{S}Y$$

where the rhs. is the classical bivariate local polynomial smoother as in (4.2.3). □

4.8.2 Proof of Theorem 4.2

This proof[4] is similar to that of Müller (1987), but adapted for spatial regression methods under the DCS framework. From (4.2.13), the nonparametric estimator for $\widehat{m}^{(\nu_1,\nu_2)}(x_1, x_2)$ can be written as weighted sum, with either the local polynomial weights w^L or (equivalent) kernel weights w^K. By Assumption A4, product weights are used. We consider kernels of order (k, μ, ν) as defined by (4.2.14) and (4.2.15) for a weighting function $W_q(u)$ of (4.2.8) - (4.2.10) as in Chapter 2.

Without loss of generality, we only consider right boundary kernels defined on $[-1, q]$, for $q \in [0, 1]$ or interior kernels ($q = 1$), the results also hold for left boundary kernels defined on $[-q, 1]$.

Define $\mathbf{X}$ and $\mathbf{W}$ as above. Further, define the symmetric matrix

$$\widetilde{\mathbf{N}}_{p,q} = (\mathbf{X}^T\mathbf{W}_q\mathbf{X}) = \begin{pmatrix} c_0 & c_1 & \cdots & c_p \\ c_1 & c_2 & & c_{p+1} \\ \vdots & & \ddots & \vdots \\ c_p & c_{p+1} & \cdots & c_{2p} \end{pmatrix},$$

where the i, j-th element is indicated by $r = i + j - 2$. The weights w_ν^L can be written as $w_\nu^L = e_{\nu+1}\widetilde{\mathbf{N}}_{p,q}^{-1}\mathbf{X}^T\mathbf{W}_q$ with $e_{\nu+1}$ being the $(\nu + 1)$-th unit vector. Further, denote the set of all integers l leading to non-zero weights in $W_q(\bullet)$ by Q^* where

$$Q^* = \{l : -nh \leq l \leq qnh, l \in \mathbb{Z}\}.$$

---

[4] Parts of this proof can also be adoped to proof Theorem 2.4 in Chapter 2.

For any $x \in [0, 1-h]$, where h is a bandwidth, the elements c_r are given by

$$c_r = \sum_{l \in Q^*} (x_l - x)^r W_q\left(\frac{x_l - x}{h}\right) = nh^{r+1} \int_{-1}^{q} u^r W_q(u)\,\mathrm{d}u + O(h^r). \tag{4.8.2}$$

Proof. Define the r-th moment of $W_q(u)$ by $\mu_r(u) = u^r W_q(u)$ and note that μ_r is Lipschitz continuous as $W_q(u)$ is Lipschitz continuous. For the integral in (4.8.2) we have

$$\int_{-1}^{q} \mu_{q,r}(u)\,\mathrm{d}u = \sum_{l \in Q^*} \int_{\frac{l-1}{[hn]}}^{\frac{l}{[hn]}} f_{q,r}(u)\,\mathrm{d}u.$$

By the mean value theorem, there exists an $\xi_l \in \left(\frac{l-1}{[nh]}, \frac{l}{[nh]}\right)$, such that

$$\int_{-1}^{q} \mu_r(u)\,\mathrm{d}u = \frac{1}{[nh]} \sum_{l \in Q^*} \mu_r(\xi_l).$$

Using the equidistancy of the x's and Lipschitz continuity of $f_{q,r}(u)$, it follows that

$$\begin{aligned}
&\left| \sum_{l \in Q^*} (x_l - x) W_q\left(\frac{x_l - x}{h}\right) - nh^{r+1} \int_{-1}^{q} u^r W_q(u)\,\mathrm{d}u \right| \\
&= \left| h^r \sum_{l \in Q^*} \mu_r\left(\frac{l}{nh}\right) - h^r \sum_{l \in Q^*} \mu_r(\xi_l) \right| \\
&\leq h^r \sum_{l \in Q^*} \left| \mu_r\left(\frac{l}{nh}\right) - \mu_r(\xi_l) \right| \\
&\leq Lh^r \sum_{l \in Q^*} \left| \frac{l}{nh} - \xi_l \right| \\
&\leq Lh^r \sum_{l \in Q^*} \left| \frac{1}{nh} \right| = O(h^r)
\end{aligned}$$

as $|Q^*| = O(nh)$. □

Hence, $\widetilde{\mathbf{N}}_{p,q} = nh\mathbf{H}[\mathbf{N}_{p,q} + o(\mathbf{1})]\mathbf{H}$, where $\mathbf{H}$ is the diagonal matrix with elements $1, h, h^2, \ldots, h^{p-1}$. Taking the inverse, it follows for the i-th element of w_ν^L

$$\begin{aligned}
w_{\nu,i}^L &= \frac{1}{nh} \nu! e_{\nu+1}^T (\mathbf{H}^{-1}[\mathbf{N}_{p,q} + o(\mathbf{1})]^{-1}\mathbf{H}^{-1})\mathbf{X}^T \mathbf{W}_q e_i \\
&= \frac{1}{nh^{\nu+1}} \sum_{r=0}^{p} \frac{1}{h^r} [a_{\nu+r} + o(1)]\, x_i^r\, W_q\left(\frac{x_i - x}{h}\right) \\
&= \frac{1}{nh^{\nu+1}} K_{\nu,q}\left(\frac{x_i - x}{h}\right) + O\left(\frac{1}{nh^{\nu+1}}\right),
\end{aligned}$$

with the equivalent kernel $K_{\nu,q}(u)$. The notation $o(\mathbf{1})$ indicates a matrix whose elements are all of order $o(1)$. The weights are hence linked by $w_{\nu,i}^L = w_{\nu,i}^K[1 + o(1)]$.

With $u_i = (x_{1,i} - x_1)/h_1$ and $v_j = (x_{2,j} - x_2)/h_2$ we yield for the product weights with

$$\begin{aligned} w^L_{\nu_1,i} w^L_{\nu_2,j} &= \left(\sum_{r_1=0}^{p_1} \sum_{r_2=0}^{p_2} [a_{1,r_1} + o(1)][a_{2,r_2} + o(1)]\, u_i^{r_1} v_j^{r_2} \right) \frac{W_{1,i} W_{2,j}}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \\ &= \left(\sum_{r_1=0}^{p_1} \sum_{r_2=0}^{p_2} [a_{1,r_1} a_{2,r_2} + o(1)] u_i^{r_1} v_j^{r_2} \right) \frac{W_{1,i} W_{2,j}}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \end{aligned} \tag{4.8.3}$$

In the kernel regression case it is clear from (4.2.14) that

$$w^K_{\nu_1,i} w^K_{\nu_2,j} = \frac{1}{n_1 n_2 h_1^{\nu_1+1} h_2^{\nu_2+1}} \sum_{r_1=0}^{p_1} \sum_{r_2=0}^{p_2} a_{1,\nu_1+r_1} a_{2,\nu_2+r_2} u_{1,i}^{r_1} u_{2,j}^{r_2} W_{1,i} W_{2,j} \tag{4.8.4}$$

By setting the kernel order to $k = p + 1$, Theorem 4.2 follows directly from (4.8.3) and (4.8.4).

4.8.3 Expectation and Variance

As expectation and variance of a (bivariate) local regression estimator are standard results in the literature (see, e.g, Ruppert and Wand, 1994; Fan and Gijbels, 1996; Feng, 2013), we only give sketched proofs for the integral approximations of the expectation and variance of the DCS estimator. From Theorem 4.2, we can also adopt the results for kernel regression from Müller and Prewitt (1993); Facer and Müller (2003) or that of Chapter 3.

Univariate Expectation. The univariate asymptotic expectation of the local polynomial estimator can be calculated strainghtforward by using $\mathbb{E}\{y_{i,j}|x_{1,i}\} = m(x_{2,j}|x_{1,i})$ to

$$\begin{aligned} \mathbb{E}\left\{\widehat{m}^{(\nu_2)}(x_2)|x_{1,i}\right\} &= \mathbb{E}\left\{\nu_2! e_{\nu_2+1} (\mathbf{X}_2^T \mathbf{W}_2 \mathbf{X}_2)^{-1} \mathbf{X}_2^T \mathbf{W}_2 m(x_{2,j}) | x_{1,i}\right\} \\ &= \nu_2! e_{\nu_2+1} (\mathbf{X}_2^T \mathbf{W}_2 \mathbf{X}_2)^{-1} \mathbf{X}_2^T \mathbf{W}_2 \mathbf{X} \begin{bmatrix} m(x_{2,j}|x_{1,i}) \\ m^{(1)}(x_{2,j}|x_{1,i}) \\ \vdots \\ m^{p_2}(x_{2,j}|x_{1,i}) \end{bmatrix} + S_{2|1} + R_2 \\ &= m^{(\nu)}(x_{2,j}|x_{1,i}) + S_{2|1} + R_2. \end{aligned}$$

The leading term of the expansion is as defined in (4.3.2), the Taylor series remainder is

$$R_2 = \sum_{r_2=p_2+2}^{\infty} \sum_{j=1}^{n_2} w^L_{2,j} (x_{2,j} - x_2)^{r_2} \frac{m^{(r_2)}(x_2|x_{1,i})}{r_2!}$$

An analogous argumentation holds for $S_{1|2}$.

Expectation of the DCS Estimator. The leading terms of the Taylor expansion are given by S_1 and S_2 of (4.3.3), (4.3.4). Their integral approximation is

$$S_1 = h_1^{p_1-\nu_1+1}\tilde{\mu}_{p+1}(K_1)m^{(p_1+1,\nu_2)}(x_1,x_2) + O\left(\frac{1}{n_1 h_1^{\nu_1}}\right) + O\left(h_1^{p_1-\nu_1+1}\right) \quad \text{and}$$
$$S_2 = h_2^{p_2-\nu_2+1}\tilde{\mu}_{p_2+1}(K)m^{(\nu_1,p_2+1)}(x_1,x_2) + O\left(\frac{1}{n_2 h_2^{\nu_1}}\right) + O\left(h_2^{p_2-\nu_2}\right).$$

Note that we define the kernel moment function $\tilde{\mu}$ by

$$\tilde{\mu}_j(K) = \int_{-1}^{q} K(u)u^j \,\mathrm{d}u, \quad q \in [0,1]$$

Variance of the DCS Estimator. We adopt the findings by Feng (2013) for the local polynomial regression to give a sketched proof of the variance approximation (4.3.7).

After a variable transformation $s = i - i^*$, $t = j - j^*$ and defining $\kappa_1 = [n_1h_1]$ and $\kappa_2 = [n_2h_2]$, where $[\bullet]$ denotes the integer part, the variance in (4.3.6) may be rewritten as

$$\operatorname{var}\left\{\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)\right\} = \sum_{s=-2\kappa_1}^{2\kappa_1}\sum_{t=-2\kappa_2}^{2\kappa_2}\left[\sum_{i-i^*=s}\sum_{j-j^*=t} w_{1,i}^L w_{1,i-s}^L w_{2,j}^L w_{2,j-t}^L\right]\gamma(s,t).$$

Following Theorem 4.2 we use the equivalent kernel weights w^K to write

$$\begin{aligned}= \frac{1}{(n_1n_2h_1^{\nu_1+1}h_2^{\nu_2+1})^2}\sum_{s=-2\kappa_1}^{2\kappa_1}\sum_{t=-2\kappa_2}^{2\kappa_2}&\left[\sum_{i-i^*=s} K_1\left(\frac{i}{n_1h_1}\right)K_1\left(\frac{i-s}{n_1h_1}\right)\right.\\ &\left.\cdot\sum_{j-j^*=t} K_2\left(\frac{j}{n_2h_2}\right)K_2\left(\frac{j-t}{n_2h_2}\right)\right]\gamma(s,t).\end{aligned} \tag{4.8.5}$$

Define the autocovariance response function for a kernel $K(u)$(Feng, 2013) by

$$G_K(u) = \begin{cases} \int\limits_{-1}^{u+1} K(v)K(v-u)\,\mathrm{d}v & \text{for } -2 \le u \le 0 \\ \int\limits_{u-1}^{1} K(v)K(v-u)\,\mathrm{d}v & \text{for } 0 \le u \le 2 \\ 0 & \text{otherwise.} \end{cases} \tag{4.8.6}$$

This function quantifies the contribution of the autocovariance to the overall variance of the estimator $\widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2)$. Defining $u = s/\kappa$, it holds that

$$\frac{1}{n_1h_1}\sum_{i-i^*=s} K_1\left(\frac{i}{n_1h_1}\right)K_1\left(\frac{i-s}{n_1h_1}\right) = G_{K_1}(u) + O\left(\frac{1}{n_1h_1}\right)$$

with a similar result for $G_{K_2}(v)$

Proof. Let $-2\kappa \leq s \leq 0$ and define $v = \widetilde{v}/\kappa$ and u as above. Then we have

$$\begin{aligned}
&\left| \frac{1}{nh} \sum_{-\kappa}^{s+\kappa} K\left(\frac{i}{nh}\right) K\left(\frac{i-s}{nh}\right) - \int_{-1}^{u+1} K(v)K(v-u)\,\mathrm{d}v \right| \\
&= \left| \frac{1}{nh} \sum_{-\kappa}^{s+\kappa} K\left(\frac{i}{nh}\right) K\left(\frac{i-s}{nh}\right) - \frac{1}{nh} \sum_{-\kappa}^{s+\kappa} \int_{r_{i-1}}^{r_i} K\left(\frac{\widetilde{v}}{nh}\right) K\left(\frac{\widetilde{v}-s}{nh}\right) \right| \\
&\leq \frac{1}{nh} \sum_{-\kappa}^{s+\kappa} \left| K\left(\frac{i}{nh}\right) K\left(\frac{i-s}{nh}\right) - K\left(\frac{\xi_i}{nh}\right) K\left(\frac{\xi_i - s}{nh}\right) \right|
\end{aligned}$$

where the r_i form a suitable partition and ξ are adequate mean values. Then, we can use the Lipschitz continuity of $K(u)$ and obtain

$$\leq \frac{L}{nh} \sum_{-\kappa} s + \kappa \left(\left| \frac{i - \xi_i}{nh} \right| + \left| \frac{i - s - \xi_i + s}{nh} \right| \right) = O\left(\frac{1}{nh}\right).$$

□

With the autocovariance response function$G_K(u)$ from (4.8.6), the term for the variance in (4.8.5) is written as

$$\begin{aligned}
&\operatorname{var}\left\{ \widehat{m}^{(\nu_1,\nu_2)}(x_1,x_2) \right\} \qquad (4.8.7) \\
&= \frac{1}{n_1 n_2 h_1^{2\nu_1+1} h_2^{2\nu_2+1}} \sum_{s=-2\kappa_1}^{2\kappa_1} \sum_{t=-2\kappa_2}^{2\kappa_2} \left[G_{K_1}\left(\frac{s}{n_1 h_1}\right) G_{K_2}\left(\frac{t}{n_2 h_2}\right) + o(1) \right] \gamma(s,t).
\end{aligned}$$

In the asymptotic limit by, Assumption A3, it holds that $n_1 h_1, n_2 h_2 \to \infty$ as $n_1, n_2 \to \infty$. Applying the same rationale as Feng (2013), we may divide the double sums of (4.8.7) into two double sums, for all $|s| \leq 1/h_1$, $|t| \leq 1/h_2$ and the other for the remaining terms. Then (4.8.7) becomes

$$\begin{aligned}
&= \sum_{|s|\leq 1/h_1} \sum_{|t|\leq 1/h_2} G_{K_1}\left(\frac{s}{n_1 h_1}\right) G_{K_2}\left(\frac{t}{n_2 h_2}\right) \gamma(s,t) \\
&\quad + \sum_{|s|> 1/h_1} \sum_{|t|> 1/h_2} G_{K_1}\left(\frac{s}{n_1 h_1}\right) G_{K_2}\left(\frac{t}{n_2 h_2}\right) \gamma(s,t).
\end{aligned}$$

Now, since $h_1, h_2 \to 0$ as $n_1, n_2 \to \infty$, the second term vanishes in the limit. Employing Assumption A3 again, the arguments in G_K approach zero, where it holds that $G_K(0) = R(K)$, with $R(K)$ as defined above. We arrive at the asymptotic approximation in (4.3.6).

4.8.4 Optimal Bandwidths

Let Assumptions A1 to A4 as well as A5′ hold. From the AMISE (4.3.10), we get the first order condition for the optimal bandwidth h_1

$$\begin{aligned}\frac{\partial AMISE(h_1,h_2)}{\partial h_1} &= 2(\delta+1)\tilde{\mu}^2_{p_1+1}(K_1)I_{11}h_1^{2\delta+1}\\ &\quad + 2(\delta+1)\tilde{\mu}_{p_1+1}(K_1)\tilde{\mu}_{p_2+1}(K_2)I_{12}h_1^{\delta}h_2^{\delta+1}\\ &\quad - \frac{(2\nu_1+1)R(K_1)R(K_2)}{n_1n_2h_1^{2\nu_1+2}h_2^{2\nu_2+1}}c_f = 0.\end{aligned} \tag{4.8.8}$$

Note that the first order condition for the derivative with respect to h_2 is analogous. Solving (4.3.10) for the variance term and equalizing both first order conditions yields

$$\begin{aligned}&\frac{\tilde{\mu}_{p_1+1}(K_1)I_{11}}{2\nu_1+1}h_1^{2(\delta+1)} - \frac{\tilde{\mu}_{p_2+1}(K_2)I_{22}}{2\nu_2+1}h_2^{2(\delta+1)}\\ &\qquad + \tilde{\mu}_{p_1+1}(K_1)\tilde{\mu}_{p_2+1}(K_2)I_{12}\left(\frac{\nu_2-\nu_1}{(2\nu_1+1)(2\nu_2+1)}\right)h_1^{\delta+1}h_2^{\delta+1} = 0,\end{aligned}$$

which is a quadratic equation in $h_1^{\delta+1}$. The positive solution is the linear relation $h_1 = A_1h_2$ with A given in (4.3.12). Applying this relation to the first order condition (4.8.8) leads to the optimal bandwidth formula.

Table 4.4: *Financial application: Estimated SARMA models for volatility and volumes of Allianz SE*

Data	**Err. Type**	σ	ϕ	ψ
Allianz SE Volatility	iid.	4.612	-	-
	SARMA	4.503	$\begin{pmatrix} 1 & -0.9876 \\ -0.9872 & 0.9749 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.9507 \\ -0.965 & 0.9174 \end{pmatrix}$
	SARMA*	6.525	$\begin{pmatrix} 1 & -1.1242 & 0.1279 \\ -1.0045 & 1.1292 & -0.1285 \\ 0.0143 & -0.016 & 0.0018 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.9769 \\ -0.9686 & 0.9463 \end{pmatrix}$
	SAR	4.528	$\begin{pmatrix} 1 & -0.1639 & -0.032 & -0.0466 \\ -0.0268 & -0.0166 & -0.0126 & -0.008 \\ -0.0227 & -0.0135 & -0.0092 & -0.0042 \\ -0.0229 & -0.0087 & -0.0056 & -0.0048 \end{pmatrix}$	$\begin{pmatrix} 1 \end{pmatrix}$
Allianz SE Volumes	iid.	12790	-	-
	SARMA	12650	$\begin{pmatrix} 1 & -0.9984 \\ -0.012 & 0.012 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.9876 \\ 0.0105 & -0.0104 \end{pmatrix}$
	SARMA*	12640	$\begin{pmatrix} 1 & -1.0113 & 0.0128 \\ -0.0226 & 0.0228 & -3\text{E}-04 \\ -0.0161 & 0.0163 & -2\text{E}-04 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.988 \end{pmatrix}$
	SAR	12770	$\begin{pmatrix} 1 & -0.0491 & -0.0435 & -0.0418 \\ -0.0145 & -0.0142 & -0.0144 & -0.0143 \\ -0.0093 & -0.0085 & -0.0088 & -0.0091 \\ -0.0063 & -0.0063 & -0.0068 & -0.0061 \end{pmatrix}$	$\begin{pmatrix} 1 \end{pmatrix}$

Table 4.5: *Financial application: estimated models for volatility and volumes of Siemens AG*

Data	**Err. Type**	σ^2	ϕ	ψ
Siemens AG Volatility	iid.	5.322	-	-
	SARMA	5.211	$\begin{pmatrix} 1 & -0.9847 \\ -0.9976 & 0.9823 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.9486 \\ -0.9876 & 0.9368 \end{pmatrix}$
	SARMA*	30.18	$\begin{pmatrix} 1 & -1.1759 & 0.1805 \\ -1.0086 & 1.1861 & -0.182 \\ 0.0125 & -0.0147 & 0.0023 \end{pmatrix}$	$\begin{pmatrix} 1 & -1.0426 & 0.0659 \\ -0.9813 & 1.0231 & -0.0646 \end{pmatrix}$
	SAR	5.246	$\begin{pmatrix} 1 & -0.1456 & -0.0342 & -0.0408 \\ -0.0261 & -0.013 & -0.0111 & -0.0042 \\ -0.0229 & -0.0114 & -0.0072 & -0.0027 \\ -0.0203 & -0.0095 & -0.0058 & -0.0018 \end{pmatrix}$	(1)
Siemens AG Volumes	iid.	23940	-	-
	SARMA	23760	$\begin{pmatrix} 1 & -0.9983 \\ -0.008 & 0.008 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.989 \\ 0.0074 & -0.0073 \end{pmatrix}$
	SARMA*	23720	$\begin{pmatrix} 1 & -1.0057 & 0.0074 \\ -0.0154 & 0.0155 & -1\text{E}{-04} \\ -0.0092 & 0.0093 & -1\text{E}{-04} \end{pmatrix}$	$\begin{pmatrix} 1 & -0.9892 \end{pmatrix}$
	SAR	23910	$\begin{pmatrix} 1 & -0.0365 & -0.034 & -0.0338 \\ -0.0111 & -0.0116 & -0.011 & -0.0107 \\ -0.0059 & -0.0059 & -0.0058 & -0.0059 \\ -0.0044 & -0.0043 & -0.0042 & -0.0044 \end{pmatrix}$	(1)

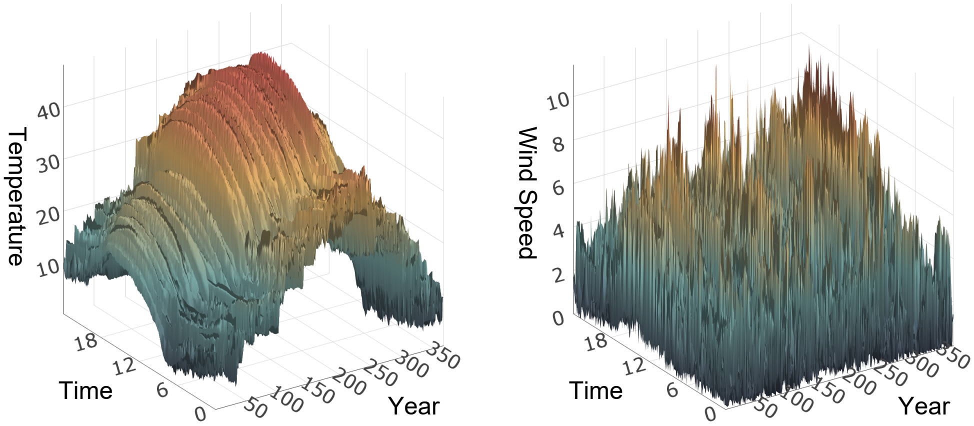
5 Further Research Topics

The methods and algorithms presented in this thesis aim at further development of the DCS technique for estimation of mean surfaces of functional time series. Naturally, there are plenty of ways in which the proposals can be extended. In general, these extensions can be divided into two areas; extension of the scope of applications or extension of the methods themselves. These areas are intertwined, some new applications require development of new methods as well, to account for special features of the applications of interest, if the methods available are not suitable for these considerations.

In the previous chapters the methods and applications aimed at financial applications. However, the findings can be generalized to a broader scope of functional data as described in Section 1.1. A direct transition of the methods is possible to lattice time series from non-financial areas of the type considered in this thesis. As electronic and digital monitoring and measuring becomes cheaper and widely available detailed time series over a long period become available as well. Under the spatial time series framework many of them can be divided into an intraday and an interday component if they are recorded over several days.

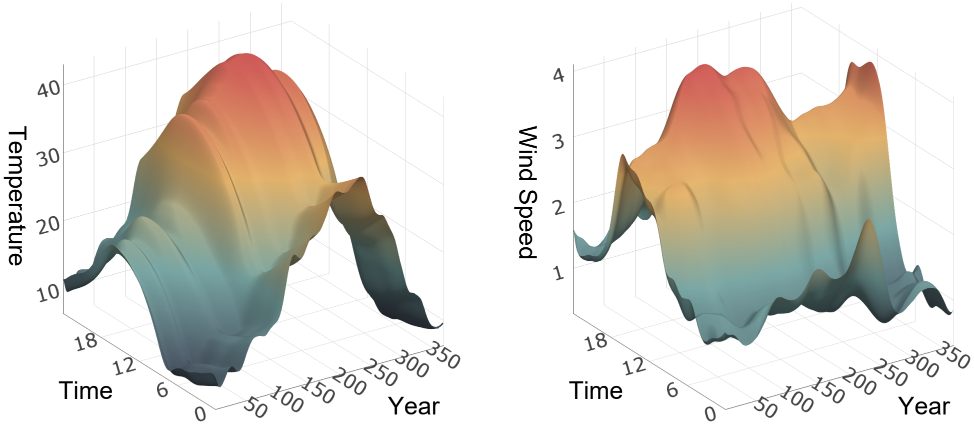
Monitoring of environmental data is a topic of increasing importance especially in the face of the climate crisis. The database of the NOAA (National Oceanic and Atmospheric Administration) (Diamond et al., 2013) provides plenty of environmental data which can be represented in our functional time series framework. Exemplary surfaces of the time series of temperature and wind speed in Yuma, AZ during 2020 are shown in Figure 5.1, the data is collected in 5-minute intervals. Although the general structure of the data is visible from the raw data, smoothing uncovers the mean structure very clearly. The use of an SARMA assumption to model the errors leads to a major increase in the selected bandwidths as expected, due to a strong autoregressive component in the data. The estimated SARMA models for the errors in the temperature and wind speed example are given in Table 5.1. The estimated coefficients from the SARMA models show substantial autoregressive components which is expected due to temperatures and also wind speed to some degree are not subject to a very fast change over the intraday observations. For temperatures, this effect is stronger in the intraday and interday direction, while for the wind speed, the interday dependency is much weaker. These findings somehow correspond to the common experience of weather.

The assumption of a parametric SARMA for modeling the dependency structure in the error terms $\varepsilon_{i,j}$ might be too restrictive in some applications. More flexibility in the model assumption for the dependency structure is be allowed by introducing a nonparametric estimation method for c_f, which is the spectral density of the error term process at the origin. Let $\varepsilon_{i,j}$ be the observations of the error term process of the model with dependent



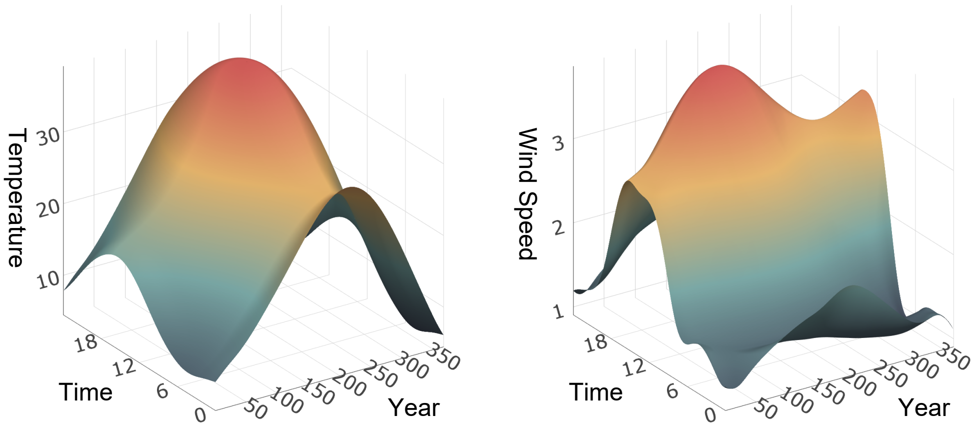
(a) Temperature, observations

(b) Wind speed, observations



(c) Temperature, iid. errors

(d) Wind speed, iid. errors



(e) Temperature, SARMA errors

(f) Wind speed, SARMA errors

Figure 5.1: *Observed and estimated surfaces of temperature* $[C^\circ]$ *and wind speed* $[m/s]$ *in Yuma, AZ in 2020. Surfaces are smoothed under iid. and* $\mathrm{SARMA}\,((1,1),(1,1))$ *error assumptions.*

errors in (4.2.1). The corresponding spectral density $f(\omega_1, \omega_2)$ is estimated by the Fourier transform of the autocovariance function of the $\varepsilon_{i,j}$, e.g. given by Robinson (2007)

$$\widehat{f}(\omega_1, \omega_2) = \left(\frac{1}{2\pi}\right)^2 \sum_{k_1=1-n_1}^{n_1-1} \sum_{k_2=1-n_2}^{n_2-1} w\left(\frac{k_1}{\lambda_1}\right) w\left(\frac{k_1}{\lambda_2}\right) \widehat{\gamma}(k_1, k_2) e^{-\mathrm{i}k_1\omega_1 - \mathrm{i}k_2\omega_2} \qquad (5.0.1)$$

where $\omega_1, \omega_2 \in [-\pi, \pi]$, $w(\bullet)$ is a suitable lag window weighting function and the autocovariances are estimated from

$$\widehat{\gamma}(k_1, k_2) = \frac{1}{n_1 n_2} \sum_{i^*=1}^{n_1-|k_1|} \sum_{j^*=1}^{n_2-|k_2|} \varepsilon_{i,j}\varepsilon_{i-i^*, j-j^*}.$$

There exist several types of lag window weights $w(\bullet)$, a simple but useful variant is the Bartlett weighting function

$$w(u) = \begin{cases} 1 - |u|, & |u| \leq 1 \\ 0, & \text{otherwise} \end{cases}.$$

Bühlmann (1996) proposed an IPI bandwidth selection algorithm for the bandwidth of an univariate estimator which can be extended to the two dimensional case for estimation of λ_1, λ_2. This algorithm differs from those in Propositions 3.2 and 4.2 as the optimal bandwidths are selected locally, depending on ω_1, ω_2. In Figure 5.2 the true and estimated spectral density surfaces of the SARMA process used in Section 4.5 with coefficient matrices in (4.5.4) are shown. A nonparametric estimator for the variance coefficient of the error process is then $c_f = \widehat{f}(0, 0)$. For the example surfaces, we get $f(0, 0) = 6.129\text{E}{-04}$ and $\widehat{f}(0, 0) = 6.273\text{E}{-04}$. In Table 5.2, the estimated bandwidths along with c_f are tabulated for examples of the volatility and volume surfaces of Allianz SE and BMW AG using nonparametric estimation of c_f. Contrary to the examples in Section 4.6, data sampled at the 5-minute level was used, over the time span from 2007-01-02 to 2010-12-30[1].

Introducing long-memory errors into the semiparametric regression model (4.2.1) is a more challenging topic, as now the asymptotic contribution of the autocovariances on the variance of the local estimator is not negligible. Hence, a complete separation of the variance coefficient c_f as in Proposition 4.1 is not possible; the optimal bandwidths are explicitly a function of the long memory parameters d_1, d_2. A (separable) SFARIMA model is given by Beran et al. (2009), where the same notation as in (4.4.3) and (4.4.4) is used

$$\phi_1(B_1)\phi_2(B_2)(1 - B_1)^{d_1}(1 - B_2)^{d_2}\varepsilon_{i,j} = \psi_1(B_1)\psi_2(B_2)\eta_{i,j},$$

with $d_1, d_2 \in [0, 0.5)$. The formulas for the optimal bandwidths differ from those in Propositions 3.1 and 4.1. Some exemplary results of bandwidth selection under SFARIMA errors

---

[1] The data was aggregated by calendar time sampling (CTS). Original Data was obtained from Thomson Reuters.

are given in Table 5.3, for the larger data sets of 1-minuite observations from 2004-2014 with $n_1 = 2738$, $n_2 = 510$. The larger bandwidths of the short-memory bandwidth selection might be due to a misspecification in the model, which results from unaccounted long-memory effects.

Table 5.1: *Estimated* $\mathrm{SARMA}\,((1,1),(1,1))$ *models for temperature and wind speed in Yuma, AZ.*

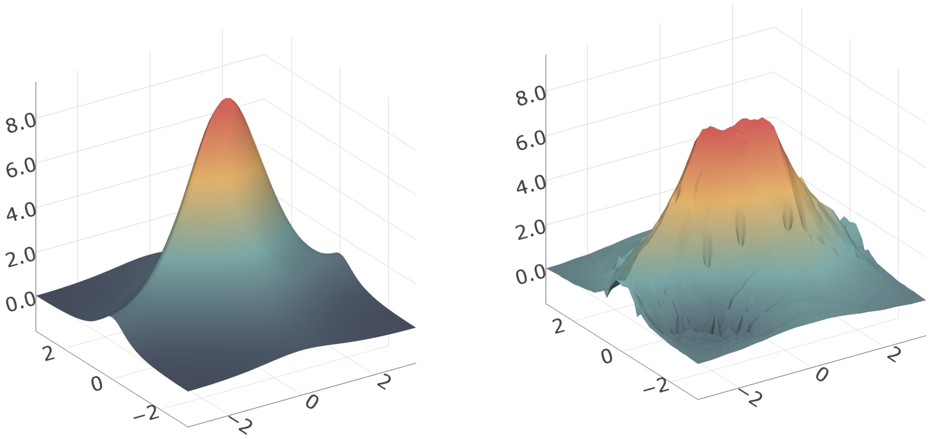
	Model	$\mathbf{h_1}$	$\mathbf{h_2}$	σ^2	$\boldsymbol{\phi}$	$\boldsymbol{\psi}$
Temperature	iid.	0.0408	0.0604	7.7212	-	-
	SARMA	0.3235	0.3648	0.2056	$\begin{pmatrix} 1 & -0.9966 \\ -0.8215 & 0.8187 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.0258 \\ -0.0556 & 0.0014 \end{pmatrix}$
Wind Speed	iid.	0.0767	0.0725	1.7573	-	-
	SARMA	0.2334	0.1444	0.2712	$\begin{pmatrix} 1 & -0.9664 \\ -0.0427 & 0.0412 \end{pmatrix}$	$\begin{pmatrix} 1 & -0.3333 \\ 0.1169 & -0.0390 \end{pmatrix}$

Table 5.2: *Bandwidths for smoothing the Allianz SE and Siemens AG volatility surfaces of 5-minute data from 2007-2010 with* $n_1 = 1016$*,* $n_2 = 101$ *observations. The bandwidths under a short memory assumption for the error terms are obtained using an iid. model, an* $\mathrm{SARMA}\,(1,1),(1,1))$ *model and a nonparametric spectral density estimate for* c_f*.*

Volatility of	**iid.**			**SARMA**			**Nonparametric**		
	h_1	h_2	c_f	h_1	h_2	c_f	h_1	h_2	c_f
ALV Volatility	0.0739	0.1219	5.44E+00	0.1738	0.2803	3.87E+02	0.1620	0.2685	2.94E+02
BMW Volatility	0.0850	0.1192	5.54E+00	0.1571	0.2127	1.04E+02	0.1737	0.2433	1.83E+02
ALV Volumes	0.0433	0.1092	2.16E−01	0.2127	0.4735	1.74E+02	0.1860	0.3908	7.00E+01
BMW Volumes	0.0534	0.1232	2.47E−01	0.1597	0.4397	6.29E+01	0.1514	0.4049	4.22E+01

Table 5.3: *Bandwidths for smoothing the Allianz SE and Siemens AG volatility surfaces of 1-minute data from 2004-2014 used in Chapter 4 under a long-memory assumption for the error terms. The bandwidths are selected under an* $\mathrm{SARMA}\,((1,1),(1,1))$ *and an* $\mathrm{SFARIMA}\,((1,1),(1,1))$.

Volatility of		**SARMA**			**SFARIMA**	
	h_1	h_2	c_f	h_1	h_2	c_f
ALV Volatility	0.09967	0.16900	2.46984E+03	0.10771	0.12905	2.44919E−01
BMW Volatility	0.08311	0.11721	4.81752E+02	0.12261	0.12309	5.95235E−01
ALV Volumes	0.14757	0.30166	9.50627E+09	0.08568	0.15508	1.54287E+06
BMW Volumes	0.12555	0.34013	6.26523E+09	0.09792	0.15641	8.17673E+05



(a) true (b) estimated

Figure 5.2: *True and estimated spectral density of the SARMA process* (4.5.4). *The bandwidths for the nonparametric estimate by applying* (5.0.1) *are selected by the spatial version of the algorithm by Bühlmann (1996).*

6 Conclusion

The major topic of this dissertation is the further development of the DCS method to estimate the mean surface of a functional time series on a lattice.

The proposed representation of time series on a lattice provides a method for detecting and estimating of certain features in this data. Throughout this thesis, the most prominent example was a lattice time series with an intraday and interday index, as this type of time series has a wide range of applications. The surface of time series of this type can hence be used for exploring the behavior over the days and inside the days and especially the relationship between those two components. Another type of functional lattice time series can be constructed from univariate time series with a regular seasonal pattern. The spatial representation and estimation of the corresponding mean surface may be utilized to filter out the seasonal component. Further lattice time series in which only one component actually represents time can be defined under the proposed framework. Examples are the time series of temperatures measured at different locations in a solid-state body in a physics experiment or the time series of the number of cars on different sections of a street monitored over a day. These examples are somehow closer to the second definition of functional time series by Li et al. (2019a) cited in Chapter 1. The necessary assumption for the presented functional regression framework is that all distances between observations are well defined on a cardinal scale, be it in time units, spatial units or some other dimensions.

In Chapter 2 boundary modification methods in local regression are treated. Local regression is known to solve the boundary problem from which kernel regression suffers and due to this, admittedly very important, characteristic of local regression, the further discussion of the boundaries is often neglected. This is especially problematic in cases where boundary regions dominate the total estimation area. The example of real-time monitoring in Section 1.2, where each point is treated as a boundary point, illustrates this issue. Adopting the ideas of the boundary kernels by Müller (1991) and Müller and Wang (1994) the concept of boundary correction is extended to a boundary modification scheme for local polynomial regression estimators. Two new classes of boundary-modified local regression weights are introduced corresponding to the kernel regression weights proposed by Müller (1991) and Müller and Wang (1994). It is shown that the use of these weighting functions leads to estimates in the interior region which are exactly equivalent to the common use of truncated kernels. However, the new kernels provide a smooth continuation of estimates in the inner region to estimates in the boundary region. The findings emphasize the direct correspondence between local polynomial regression and the generation of kernels for kernel regression (see e.g. Hastie and Loader, 1993). The proposed methods for boundary modification in local regression can be utilized for generating the classes of boundary kernels defined by Müller (1991) and Müller and Wang (1994) of desired orders. Explicit formulas of kernels generated by these methods can be found in Table 2.1 and Table 2.2.

The DCS for kernel regression is studied and further developed in Chapter 3. A boundary correction scheme is introduced in Section 3.3.1 applying the boundary kernels generated from the methods proposed in Chapter 2. In previous applications of the DCS by Feng (2013) or Peitz and Feng (2015) boundary correction was not considered explicitly. Further improvement in the efficiency of the DCS algorithm is achieved via the FDCS defined in Section 3.3.2 which utilizes the assumed lattice structure to avoid redundant computations. The FDCS has equivalent properties to the DCS (under a regular lattice structure) and is designed for direct implementation of the algorithm in a program such as R. From the asymptotic properties of the DCS/FDCS estimator found in Section 3.3.4, the optimal bandwidths are derived by minimization of the AMISE in Section 3.4.1. These bandwidths are selected with the IPI-algorithm (Gasser et al., 1991) described in Section 3.4.2. This algorithm is the preferred bandwidth selection method for this framework, as other bandwidth selection methods like CV are too slow for large data sets. The IPI iteratively computes certain partial derivatives of the regression function to obtain intermediate bandwidth estimates which are then used for regression again, converging towards the true asymptotic optimal bandwidths. To estimate the derivatives required in each step appropriate kernels for derivative estimation from the kernel generating method of Chapter 2 are used. The bandwidth selection algorithm is assessed in Section 3.5 at two simulated spatial surface functions. The results indicate that the bandwidth selector is subject to a certain variance, depending on the function under consideration. However, the true optimal bandwidths are found with 9% confidence on average. The results also indicate, that the overall precision of the algorithm strongly depends on the data and function under consideration. The application of the methods to financial data in Section 3.6 demonstrates that the proposed algorithms are well-suited for estimating the mean or expectation surface from noisy data. The spot volatility surfaces clearly show the influence of the financial crisis 2008 in the expected ways, however, the surface seems to be undersmoothed by a small amount, as some finer structures are still visible in the surfaces.

The DCS under local polynomial regression (LP-DCS) is newly developed in Chapter 4. Local polynomial regression has some advantages over kernel regression, e.g., it solves the boundary problem and provides a clean derivative estimation method, which is useful for the IPI bandwidth selection. It is shown that the LP-DCS shares some properties with the kernel regression DCS such that the FDCS is also applicable here. Similar to the boundary correction in kernel regression, the boundary modification scheme introduced in Chapter 2 is employed for the estimation of the spatial surfaces under the LP-DCS context. Again, the optimal bandwidths are obtained by an IPI algorithm. The estimation of derivatives is simplified under local regression, which is an advantage for the calculation of the partial derivatives necessary in the IPI algorithm. The assumption of iid. error terms is given up in favor of a dependency structure modeled by a spatial ARMA (SARMA) model. Some estimation procedures are suggested for parametric estimation of the SARMA along with a convenient matrix notation. In the simulation study of Section 4.5 it becomes evident that neglecting dependency in the errors leads to incorrect estimation of the bandwidths. When accounting for SARMA errors in the volatility surface estimations of Section 4.6, the selected bandwidths increase clearly from the bandwidths under an iid. assumption for the error terms. Hence, an incorrect model assumption for the error terms would lead to an undersmoothing of the mean surface. This behavior also holds for the environmental data presented in Chapter 5.

In summary, the DCS technique provides an efficient way of estimating the mean surface of a spatial time series in a nonparametric way. The proposed methods for nonparametric estimation and bandwidth selection under iid. and short memory dependent errors work well under the spatial model and are suitable for estimation of mean surfaces of the data. Although the focus was on the estimation of time series in a financial context, the findings are applicable to a broader scope of data structures from other areas of research.

A Appendix: *DCSmooth* Vignette

The R package DCSmooth *gathers the methods and tools developed in this thesis for practical application. This slightly modified vignette of this package gives an overview of the package. Note that some changes in notation compared to the theoretical chapters 2 - 4 are made, for easier programming and usage of the package. This concerns especially the notation of the covariate variables x_1, x_2, which are replaced by* `x` *and* `t` *in the code. The package is published on CRAN (Schäfer, 2021).*

A.1 Introduction

This vignette describes the use of the *DCSmooth*-package and its functions. The *DCSmooth* package provides some tools for non- and semiparametric estimation of the mean surface m of an observed sample of some function

$$y(x,t) = m(x,t) + \varepsilon(x,t).$$

The *DCSmooth* contains the following functions, methods and data sets:

Functions	
`set.options()`	Define options for the `dcs()`-function.
`dcs()`	Nonparametric estimation of the expectation function of a matrix `Y`. Includes automatic iterative plug-in bandwidth selection.
`surface.dcs()`	3d-plot for the surfaces of an `"dcs"`-object.
`sarma.sim()`	Simulate a SARMA-model.
`sarma.est()`	Estimate the parameters of a SARMA-model.
`sfarima.sim()`	Simulate a SFARIMA-model.
`sfarima.est()`	Estimate the parameters of a SFARIMA-model.
`kernel.assign()`	Assign a pointer to a kernel function.
`kernel.list()`	Print list of kernels available in the package.

Methods/Generics	
`summary.dcs()`	Summary statistics for an object of class `"dcs"`.
`print.dcs()`	Print an object of class `"dcs"`.
`plot.dcs()`	Plot method for an `"dcs"`-object, returns contour plot.
`residuals.dcs()`	Returns the residuals of the regression from an `"dcs"`-object.
`print.summary_dcs()`	Print an object of class `"summary_dcs"`, which inherits from `summary.dcs()`.
`print.set_options()`	Prints an object of class `"dcs_options"`, which inherits from `set.options()`
`summary.sarma()`	Summary statistics for an object of class `"sarma"`
`print.summary_sarma()`	Prints an object of class `"summary_sarma"`, which inherits from `summary.sarma()`
`summary.sfarima()`	Summary statistics for an object of class `"sfarima"`
`print.summary_sfarima()`	Prints an object of class `"summary_sfarima"`, which inherits from `summary.sfarima()`

Data	
`y.norm1`	A surface with a single gaussian peak.
`y.norm2`	A surface with two gaussian peaks.
`y.norm3`	A surface with two gaussian ridges.
`temp.nunn`	Temperatures in Nunn, CO observed in 2020 in 5 minute intervals. (Source: NOAA)
`temp.yuma`	Temperatures in Yuma, AZ observed in 2020 in 5 minute intervals. (Source: NOAA)
`wind.nunn`	Windspeed in Nunn, CO observed in 2020 in 5 minute intervals. (Source: NOAA)
`wind.yuma`	Windspeed in Yuma, AZ observed in 2020 in 5 minute intervals. (Source: NOAA)
`returns.alv`	5 minute returns of Allianz SE from 2007 to 2010
`volumes.alv`	5 minute volumes of Allianz SE from 2007 to 2010

A.2 Details of Functions, Methods and Data

A.2.1 Functions

`set.options()`

This auxiliary function is used to set the options for the `dcs` function. An object of class `dcs_options` is created and should be used as `dcs_options`- argument in the `dcs` function.

Arguments of `set.options()` are

- **type** Specifies the regression type. Supported methods are kernel regression (`"KR"`) and local polynomial regression (`"LP"`), which is the default value.
- **kerns** A character vector of length 2 stating the identifiers for the kernels in each dimension to use. The first element corresponds to the smoothing conditional on rows, the second conditional on columns. The identifiers are of the form $X\_k\mu\nu$, where X indicates the smoothing method to use, either one of `M`, `MW` or `T`. The value k is the kernel order, μ is the smoothness degree and ν the derivative estimated by the kernel, which must match the order of derivative `drv`. For more information on the kernels see section 4.3, a list of available kernels is given in A.1. The default kernels are `"MW_220"` for both dimensions.
- **drv** Derivative (ν_x, ν_t) of $m(x, t)$ to be estimated. Note that $k \geq \nu + 2$, hence, only estimation of derivatives corresponding to kernels available is possible.
- **var_model** Specifies the model assumption and estimation method for the errors/innovations $\varepsilon(x, t)$ in the regression model. The model is selected in the form `"model_method"`. Currently available are
  - `"iid"` (independently identically distributed errors with variance estimation from the residuals, set as default).
  - `"sarma_sep"` (separable spatial ARMA (SARMA) process, two univariate processes in both directions are estimated via `stats::arima`).
  - `"sarma_HR"` (fast estimation of an SARMA process by the Hannan-Rissanen algorithm).
  - `"sarma_RSS` (estimation of an separable SARMA process by numerical minimization of the RSS).
  - `"sfarima_RSS"` (estimation of a separable spatial FARIMA (SFARIMA) model by numerical minimization of the RSS).

The models and estimation methods are described in more detail in Section 4.4.

- `...` Additional arguments passed to `set.options`. The default values of these options typically depend on other options and thus are put in an ellipsis. Accepted arguments are
  - **IPI_options** Advanced options for tuning the parameters of the iterative plug-in algorithm of the bandwidth selection. These options include 2-element vectors

for the inflation parameters (`infl_par`), the inflation exponents (`infl_exp`) and trimming parameters for stabilized estimation of the necessary derivatives (`trim`). Another option to further stabilize estimation of derivatives at the boundaries is the use of a constant estimation window at the boundaries setting the logical flag `const_window` to `TRUE`. The default values for the IPI-options depend partly on the regression type and error model selected and are given below.

- **model_order** controls the order of the parametric error term model if an SARMA or SFARIMA model is used. This can be either a list of the form `list(ar = c(1, 1), ma = c(1, 1))` (the default for SARMA and SFARIMA) specifying the model order, or any of `c("aic", "bic", "gpac")` specifying an order selection criterion. Note that `gpac` does not work under SFARIMA errors.
- **order_max** Controls the maximum order if an order selection process is chosen in `model_order`. Is a list of the form `list(ar = c(1, 1), ma = c(1, 1))` (the default).

`set.options()` returns an object of class `"dcs_options` including the following values

- **type** Inherited from input.
- **kerns** Inherited from input.
- **drv** Inherited from input.
- **p_order** A numeric vector of length 2, computed from `drv` by $p_k = \nu_k + 1, k = x, t$.
- **var_model** Inherited from input.
- **IPI_options** Options for the iterative-plug in algorithm for bandwidth selection. If unchanged, values are set conditional on `type` (see default values for `KR` and `LP` below).
- **add_options** A list containing the additional options `model_order` and `order_max` if available.

Every argument of the `set.options` function has a default value. Hence, just using `set.options()` will produce a complete set of options for double conditional smoothing regression in `dcs` (which is also implemented as default options in `dcs`, if the argument `dcs_options` is omitted).

Default options for kernel regression (`type = "KR"`) are

```
summary(set.options(type = "KR"))
#> dcs_options
#> ---------------------------------------
#> options for DCS   rows    cols
#> ---------------------------------------
#> type: kernel regression
#> kernels used:        MW_220  MW_220
#> derivative:      0   0
#> variance model:
#> ---------------------------------------
#> IPI options:
#> inflation parameters     2   1
#> inflation exponents  0.5 0.5
#> trim             0.05    0.05
#> constant window width    FALSE
#> ---------------------------------------
```

Default options for local polynomial regression (`type = "LP"`) are

```
summary(set.options(type = "LP"))
#> dcs_options
#> ---------------------------------------
#> options for DCS   rows    cols
#> ---------------------------------------
#> type: local polynomial regression
#> kernel order:        MW_220  MW_220
#> derivative:      0   0
#> polynomial order:    1   1
#> variance model:  iid
#> ---------------------------------------
#> IPI options:
#> inflation parameters     1   1
#> inflation exponents  auto
#> trim             0.05    0.05
#> constant window width    FALSE
#> ---------------------------------------
```

`dcs()`

The `dcs()`-function is the main function of the package and includes IPI-bandwidth selection and non-parametric smoothing of the observations `Y` using the selected bandwidths. This function creates an object of class `dcs`, which includes the results of the DCS procedure.

Arguments of `dcs()` are

- **Y** The matrix of observations to be smoothed via the DCS procedure. This matrix should only contain numeric values and no missing observations. For computational reasons, `Y` has to have at least five rows and columns, however, for reliable results the size should be larger.

- **dcs_options** The options used for the smoothing and bandwidth selection. This should be an object of class `"dcs_options"` created by `set.options()`. This argument is optional, if omitted, all options will be set to their default values from the `set.options()` function.
- **h** Either a two-value vector of positive numeric bandwidths or `"auto"` if bandwidth selection should be employed (the default).
- **parallel** A logical flag if parallelization should be used for computation of the smoothed surfaces and its derivatives. If the order of the variance model is automatically selected, parallelization affects also this.
- `...` Further arguments to be passed to the function. This includes the equidistant covariates `X` and `T` which should be ordered numerical vectors whose length matches the number of rows of `Y` for `X` and the number of columns of `Y` for `T`.

`dcs` returns an object of class `"dcs` including the following values

- **X, T** Vectors of covariates inherited from input or calculated to be equidistant on $[0, 1]$ if these are omitted in the input.
- **Y** Matrix of observations inherited from input.
- **M** Matrix of smoothed values. If the argument `h = "auto` is used in `dcs`, the bandwidths are optimized via the IPI-algorithm, if `h` is set to fixed values, these bandwidth are used.
- **R** Matrix of residuals computed from $R = Y - M$.
- **h** Bandwidths used for smoothing of `Y`. Either obtained by IPI bandwidth selection or given as argument in `dcs`.
- **c_f** The estimated variance factor used in the last iteration of the bandwidth selection algorithm. Is set to `NA`, if no bandwidth selection is used.
- **var_est** The estimated model obtained for the error terms (residuals) $\varepsilon(x, t)$, i.e. the matrix `$R`. The output depends on the model specified in `dcs_options$var_model`. For `var_model = "iid"`, it contains the estimated standard deviation of the residuals and an indicator for stationarity, which is true by assumption. For dependent models `dcs_options$var_model = c("sarma_sep", "sarma_RSS", "sarma_HR")`, it contains the estimated model in an object of class `"sarma"` including the coefficient matrices `$ar` and `$ma`, the standard deviation `$sigma` as well as an stationarity indicator `$stnry`. For `dcs_options$var_model = "sfarima_RSS"`, the output is of class `"sfarima"`, with similar contents as `"sarma"` and the addition of the estimated long memory parameter vector `$d`.
- **dcs_options** An object of class `"dcs_options"` containing the options used in the function.
- **iterations** An integer reporting the number of iterations of the IPI algorithm. Is set to `NA`, if no bandwidth selection is used.
- **time_used** A number reporting the time (in seconds) used for the IPI algorithm (total including all iterations). Is set to `NA`, if no bandwidth selection is used.

`surface.dcs()`

This function is a convenient wrapper for the `plotly::plot_ly()` function of the *plotly* package, for easy displaying of the considered surfaces. Direct plotting is available for any object of class `"dcs"` or any numeric matrix `Y`.

Arguments of `surface.dcs()` are

- **Y** Either an object of class `"dcs"`, inheriting from a call to `dcs()` or a numeric matrix, which is then directly passed to `plotly::plot_ly()`.
- **plot_choice** Only used, if `Y` is an object of class `"dcs"`. Specifies the surface to be plotted, `1` for the original observations, `2` for the smoothed surface and `3` for the residual surface. If `plot_choice` is omitted and `Y` is an `"dcs"`-object, a choice dialogue will be prompted to the console, which asks to state one of the available options.
- **trim** A two-value vector which gives the percentage (between 0 and 0.5) of boundaries to leave out in the plot. Useful, if estimation at boundaries is unstable and has too high values compared to the inner, e.g. useful when estimation of derivatives is considered.
- `...` Further arguments to be passed to the `plotly::plot_ly()` function.

`surface.dcs()` returns an object of class `"plotly"`.

`sarma.sim()`

Simulation of a spatial ARMA process (SARMA). This function returns an object of class `"sarma"` with attribute `"subclass" = "sim"`. The simulated innovations are created from a normal distribution with specified standard deviation σ. This function uses a burn-in period for more consistent results.

Arguments of `sarma.sim()` are

- **n_x**, **n_t** The dimension of the resulting matrix of observations, where `n_x` specifies the number of rows and `n_t` the number of columns. Initially, a matrix `Y'` of size $2n_x \times 2n_t$ is simulated, for which simulation points with $i \leq n_x$ or $j \leq n_t$ are discarded (burn-in period).
- **model** A list containing the model parameters to be used in the simulation. The argument should be a list of the form `list(ar, ma, sigma)`. The values `ar` and `ma` are matrices of size $(p_x + 1) \times (p_t + 1)$ respective $(q_x + 1) \times (q_t + 1)$ and contain the coefficients in ascending lag order, so that the upper left entry is equal to 1 (for lag 0 in both dimensions). The standard deviation of the iid. innovations with zero mean is `sigma`, which should be a single positive number. See the examples in the application part 3.2 and the notation of SARMA models in Section 4.4.

`sarma.sim()` returns an object of class `"sarma"` with attribute `"subclass" = "sim"` including the following values:

- **Y** The matrix of simulated values with size $n_x \times n_t$ (determined by function arguments `n_x`, `n_t`). The matrix Y is the lower left $n_x \times n_t$ submatrix of the actually simulated matrix Y' of size $2n_x \times 2n_t$ to avoid effects from setting the initial values (burn-in period).
- **innov** The $n_x \times n_t$ matrix of iid. normally distributed innovations/errors of the SARMA model with zero mean and variance σ^2 (determined by the function argument `model\$sigma`). As with `Y`, the original matrix has size $2n_x \times 2n_t$.
- **model** The model used for simulation, inherited from input.
- **stnry** A flag indicating whether the simulated process is stationary.

`sarma.est()`

Estimate the parameters of an SARMA of given order. It returns an object of class `"sarma"` with attribute `"subclass" = "est"`. For estimation, three methods are available.

Arguments of `sarma.est()` are

- **Y** A (demeaned) matrix of observations, which contains only numeric values and no missing observations.
- **method** A character string specifying the method for estimation. Currently supported methods are the Hannan-Rissanen algorithm (`"HR"`), a separable model using two univariate estimations via `stats::arima` (`"sep"`) and a separable model which minimizes the residual sum of squares (RSS) of the model (`"RSS"`).
- **model_order** A list specifying the order of the SARMA to be estimated. This list should be of the form `list(ar = c(1, 1), ma = c(1, 1))`, where all orders should be non-negative integers. A SARMA$((1,1),(1,1))$ model is estimated by default, if `model_order` is omitted.

`sarma.est()` returns an object of class `"sarma"` with attribute `"subclass" = "est"` including the following values:

- **Y** The matrix of observations inherited from input.
- **innov** The matrix of estimated innovations (residuals).
- **model** A list of estimated model coefficients containing the matrices `ar` of autoregressive coefficients, the matrix `ma` of moving average coefficients as well as the standard deviation of residuals `sigma`.
- **stnry** A flag indicating whether the estimated process is stationary.

`sfarima.sim()`

Simulation of a (separable) spatial fractional ARIMA (SFARIMA) process. This function returns an object of class `"sfarima"` with attribute `"subclass" = "sim"`. The simulated innovations are created from a normal distribution with specified standard deviation σ. This function uses a burn-in period for more consistent results.

Arguments of `sfarima.sim()` are

- **n_x, n_t** The dimension of the resulting matrix of observations, where `n_x` specifies the number of rows and `n_t` the number of columns. Initially, a matrix `Y'` of size $2n_x \times 2n_t$ is simulated, for which simulation points with $i \leq n_x$ or $j \leq n_t$ are discarded (burn-in period).
- **model** A list containing the model parameters to be used in the simulation of the form `list(ar, ma, d, sigma)`. The values `ar` and `ma` are matrices of size $(p_x+1) \times (p_t+1)$ respective $(q_x+1) \times (q_t+1)$ and containing the coefficients in ascending lag order, so that the upper left entry is equal to 1 (for lag 0 in both dimensions). The long-memory parameters d_x, d_t are stored in `d`, a numeical vector of length 2, with $0 < d_x, d_t < 0.5$. The standard deviation of the iid. innovations with zero mean is `sigma`, which should be a single positive number. See the examples in the application part 3.3 and the notation of the short memory SARMA part in Section 4.4.

`sfarima.sim()` returns an object of class `"sfarima"` with attribute `"subclass" = "sim"` including the following values:

- **Y** The matrix of simulated values with size $n_x \times n_t$ (determined by function arguments `n_x`, `n_t`). The matrix Y is the lower left $n_x \times n_t$ submatrix of the actually simulated matrix Y' of size $2n_x \times 2n_t$ to avoid effects from setting the initial values (burn-in period).
- **innov** The $n_x \times n_t$ matrix of iid. normally distributed innovations/errors of the SFARIMA model with zero mean and variance σ^2 (determined by the function argument `model\$sigma`). As with `Y`, the original matrix has size $2n_x \times 2n_t$.
- **model** The model used for simulation, inherited from input.
- **stnry** A flag indicating whether the simulated process is stationary.

`sfarima.est()`

Estimation of an SFARIMA process. This function minimizes the residual sum of squares (RSS) to estimate the SFARIMA-parameters of a given order. It returns an object of class `"sfarima"` with attribute `"subclass" = "est"`.

Arguments of `sfarima.est()` are

- **Y** A (demeaned) matrix of observations, which contains only numeric values and no missing observations.
- **model_order** A list specifying the order of the SFARIMA to be estimated. This list should be of the form `list(ar = c(1, 1), ma = c(1, 1))`. All orders should be non-negative integers. A SFARIMA$((1,1),(1,1))$ model is estimated by default, if `model_order` is omitted.

`sfarima.est()` returns an object of class `"sfarima"` with attribute `"subclass" = "est"` including the following values:

- **Y** The matrix of observations inherited from input.
- **innov** The matrix of estimated innovations (residuals).
- **model** A list of estimated model coefficients containing the matrices `ar` of autoregressive coefficients, the matrix `ma` of moving average coefficients as well as the vector `d` holding the long-memory parameters and the standard deviation of residuals `sigma`.
- **stnry** A flag indicating whether the estimated process is stationary.

`kernel.assign()`

This function sets an external pointer to a specified boundary kernel available in the DCSmooth package. These kernels are functions $K(u, q)$, where u is a vector on $[q, -1]$ and $q \in [0, 1]$. The boundary kernels are as proposed by Müller and Wang (1994); Müller (1991) and constructed via the method described in Chapter 2. Available types are Müller-Wang (`MW`), Müller (`M`) and truncated kernels (`T`).

Arguments of `kernel.assign()` are

- **kernel_id** The identifier for the kernel to be assigned. It is a character string of the form `"X_abc"`, where `X` is specifies the type (`MW`, `M`, `T`), `a` i the kernel order k, `b` is the degree of smootheness μ and `c` is the order of derivative ν. A list of currently useable kernel identifiers can be accessed with the function `kernel.list()`.

`kernel.assign()` returns an object of class `"function"`, which points to a precompiled kernel function.

`kernel.list()`

`kernel.list()` prints the available identifiers for use in `kernel.assign()`.

The argument of `kernel.list()` is

- **print** A logical value indicating if the list of available kernels should be printed to the console.

`kernel.list()` returns a list including the available kernels as character strings, if the argument is `print = FALSE`.

Available kernels are

k	μ	ν	Truncated Kernels	Müller Kernels	Müller-Wang Kernels
2	0	0		`M_200`	`MW_200`
2	1	0		`M_210`	`MW_210`
2	2	0	`T_220`	`M_220`	`MW_220`
3	2	1	`T_321`	`M_321`	`MW_320`
4	2	0	`T_420`	`M_420`	`MW_420`
4	2	1		`M_421`	`MW_421`
4	2	2	`T_422`	`M_422`	`MW_422`

A.2.2 Methods

The *DCSmooth* package contains the following methods

Function	Methods/Generics available
`dcs_options`	`print`, `summary`
`dcs`	`plot`, `print`, `print.summary`, `residuals`, `summary`
`sarma`	`print.summary`, `summary`
`sfarima`	`print.summary`, `summary`

A.2.3 Data

This package contains three simulated example data sets and six data sets of environmental spatial time series.

Each of the three simulated example data sets is a matrix of size 101×101 computed on $[0,1]^2$ for the following functions, where $N(\mu, \Sigma)$ is the bivariate normal distribution with mean vector μ and covariance matrix Σ:

- **y.norm1**

$$N\left(\begin{pmatrix}0.5\\0.5\end{pmatrix}, \begin{pmatrix}0.05 & 0\\0 & 0.05\end{pmatrix}\right)$$

- **y.norm2**

$$N\left(\begin{pmatrix}0.5\\0.3\end{pmatrix}, \begin{pmatrix}0.1 & 0\\0 & 0.1\end{pmatrix}\right) + \mathcal{N}\left(\begin{pmatrix}0.2\\0.8\end{pmatrix}, \begin{pmatrix}0.05 & 0\\0 & 0.05\end{pmatrix}\right)$$

- **y.norm3**

$$N\left(\begin{pmatrix}0.25\\0.75\end{pmatrix}, \begin{pmatrix}0.01 & 0\\0 & -0.1\end{pmatrix}\right) + \mathcal{N}\left(\begin{pmatrix}0.75\\0.5\end{pmatrix}, \begin{pmatrix}0.01 & 0\\0 & -0.1\end{pmatrix}\right)$$

The environmental application data sets features the temperature and wind speed asurfaces of Nunn, CO (`temp.nunn`, `wind.nunn`) and Yuma, AZ (`temp.yuma`, `wind.yuma`). The observations are taken in 2020 in 5-minute intervals. The temperatures are given in *Celsius* and wind speed in *m/s*. All data sets consist therefore of 288 columns (the intraday observations) and 366 rows (the days). The data is taken from the U.S. Climate Reference Network database at www.ncdc.noaa.gov. (see Diamond et. al. (2013), doi:10.1175/BAMS-D-12-00170.1.

For examples of financial applications, the return and volume data of German insurance company Allianz SE is available in the package. The data is aggregated to the 5-minute level over the years 2007-2010, hence, the financial crisis 2008 is covered. The matrices consist of 1016 rows representing the days and 101 (returns) respective 102 (volumes) columns for the intraday 5-minute intervals.

A.3 Application

The application of the package is demonstrated at the example of the simulated function `y.norm1`, which represents a gaussian peak on $[0,1]^2$ with $n_x = n_t = 101$ evaluation points. Different models are simulated and estimation using `dcs` is demonstrated. Whenever default options are used, they are not explicitly used as function arguments, instead only when deviating from the defaults, the options are changed.

A.3.1 Defining the Options

In order to set specific options use the `set.options()` function to create an object of class `"dcs_options"`.

```
opt1 = set.options(type = "KR", kerns = c("M_220", "M_422"),
                   drv = c(0, 2),
                   var_model = "sarma_RSS",
                   IPI_options = list(trim = c(0.1, 0.1),
                                      infl_par = c(1, 1),
                                      infl_exp = c(0.7, 0.7),
                                      const_window = TRUE),
                   model_order = list(ar = c(1, 1), ma = c(0, 0)))
summary(opt1)
#> dcs_options
#> ---------------------------------------
#> options for DCS     rows    cols
#> ---------------------------------------
#> type: kernel regression
#> kernels used:       M_220   M_422
#> derivative:          0      2
#> variance model:      sarma_RSS
#> ---------------------------------------
#> IPI options:
#> inflation parameters 1   1
#> inflation exponents  0.7 0.7
#> trim                 0.1 0.1
#> constant window width    TRUE
#> ---------------------------------------
class(opt1)
#> [1] "dcs_options"
```

The contents of the advanced option `IPI_options` can be set directly as argument in `set.options()`. Changing these options might lead to non convergent bandwidths.

```
opt2 = set.options(type = "KR", kerns = c("M_220", "M_422"),
                   drv = c(0, 2),
                   var_model = "sarma_RSS", trim = c(0.1, 0.1),
                   infl_par = c(1, 1), infl_exp = c(0.7, 0.7),
                   const_window = TRUE,
                   model_order = list(ar = c(1, 1), ma = c(0, 0)))
summary(opt2)
#> dcs_options
#> ---------------------------------------
#> options for DCS   rows    cols
#> ---------------------------------------
#> type: kernel regression
#> kernels used:     M_220   M_422
#> derivative:       0      2
#> variance model:   sarma_RSS
#> ---------------------------------------
#> IPI options:
#> inflation parameters  1   1
#> inflation exponents   0.7 0.7
#> trim                  0.1 0.1
#> constant window width    TRUE
#> ---------------------------------------
class(opt2)
#> [1] "dcs_options"
```

When using a model selection procedure, the additional option `order_max` is available:

```
opt3 = set.options(var_model = "sarma_sep", model_order = "bic",
                   order_max = list(ar = c(0, 1), ma = c(2, 2)))
summary(opt3)
#> dcs_options
#> ---------------------------------------
#> options for DCS   rows    cols
#> ---------------------------------------
#> type: local polynomial regression
#> kernel order:        MW_220  MW_220
#> derivative:          0       0
#> polynomial order:    1       1
#> variance model:  sarma_sep
#> ---------------------------------------
#> IPI options:
#> inflation parameters     1     1
#> inflation exponents      auto
#> trim                     0.05  0.05
#> constant window width    FALSE
#> ---------------------------------------
```

A.3.2 Application of the DCS with iid. Errors

The example data set is simulated by using iid. errors:

```
y_iid = y.norm1 + matrix(rnorm(101^2), nrow = 101,
                         ncol = 101)
```

Kernel regression with iid. errors. While local linear regression has some clear advantages over kernel regression, kernel regression is the faster method.

```
opt_iid_KR = set.options(type = "KR")
dcs_iid_KR = dcs(y_iid, opt_iid_KR)

# print results
dcs_iid_KR
#> dcs
#> --------------------------------------
#> DCS with automatic bandwidth selection
#> --------------------------------------
#>  Selected Bandwidths:
#>      h_x: 0.18855
#>      h_t: 0.19259
#>  Variance Factor:
#>      c_f: 0.99379
#> ---------------------------------------

# print options used for DCS procedure
dcs_iid_KR$dcs_options
#> dcs_options
#> ----------------------------------------
#> options for DCS  rows    cols
#> ----------------------------------------
#> type: kernel regression
#> kernels used:    MW_220  MW_220
#> derivative:      0    0
#> variance model:  iid
#> ----------------------------------------
```

The summary of the `"dcs"`-object provides some more detailed information:

```
summary(dcs_iid_KR)
#> summary_dcs
#> -------------------------------------------
#> DCS with automatic bandwidth selection:
#> -------------------------------------------
#> Results of kernel regression:
#> Estimated Bandwidths: h_x:    0.1886
#>                       h_t:    0.1926
#> Variance Factor:      c_f:    0.9938
#> Iterations:           4
#> Time used (seconds):  0.0379
#> -------------------------------------------
#> Variance Model:   iid
#> -------------------------------------------
#> sigma:        0.99689
#> stationary:   TRUE
#> -------------------------------------------
#> See used parameter with "$dcs_options".
```

Local polynomial regression with iid. errors. This is the default method, specification of options is not necessary. Note that local polynomial regression requires the bandwidth to cover at least the number of observations of the polynomial order plus one. For small bandwidths or too few observation points in one dimension, local polynomial regression might fail ("Bandwidth h must be larger for local polynomial regression."). It is suggested to use kernel regression in this case.

```
dcs_LP_iid = dcs(y_iid)
dcs_LP_iid
#> dcs
#> ------------------------------------------
#> DCS with automatic bandwidth selection:
#> ------------------------------------------
#>  Selected Bandwidths:
#>      h_x: 0.17265
#>      h_t: 0.19012
#>  Variance Factor:
#>      c_f: 0.99364
#> ------------------------------------------

dcs_LP_iid
#> dcs
#> ------------------------------------------
#> DCS with automatic bandwidth selection
#> --------------------------------------
#>  Selected Bandwidths:
#>      h_x: 0.17265
#>      h_t: 0.19012
#>  Variance Factor:
#>      c_f: 0.99364
#> ------------------------------------------
```

A.3.3 Application of the DCS with SARMA Errors

A matrix containing innovations following a SARMA$((p_x, p_t), (q_x, q_t))$ process can be obtained by using the `sarma.sim()` function. We use the following SARMA$((1, 1), (1, 1))$-process as example:

$$AR = \begin{pmatrix} 1 & 0.4 \\ -0.3 & -0.12 \end{pmatrix}, \quad MA = \begin{pmatrix} 1 & -0.5 \\ -0.2 & 0.1 \end{pmatrix} \quad \text{and} \quad \sigma^2 = 0.25$$

```
ar_mat = matrix(c(1, -0.3, 0.4, 0.12), nrow = 2, ncol = 2)
ma_mat = matrix(c(1, -0.2, -0.5, 0.1), nrow = 2, ncol = 2)
sigma  = sqrt(0.25)
```

```
model_list = list(ar = ar_mat, ma = ma_mat, sigma = sigma)
sim_sarma = sarma.sim(n_x = 101, n_t = 101,
                      model = model_list)
# SARMA observations

y_sarma = y.norm1 + sim_sarma$Y
```

Estimation of an SARMA process for a given order is implemented via the `sarma.est()` function (note that the simulated matrix can be accessed via `$Y`):

```
est_sarma = sarma.est(sim_sarma$Y, method = "HR",
                      model_order =
                      list(ar = c(1, 1), ma = c(1, 1)))

summary(est_sarma)
#> ------------------------------------------
#> Estimation of SARMA((1,1),(1,1))
#> ------------------------------------------
#> sigma:        0.5008
#> stationary:   TRUE
#> ar:
#>         lag 0  lag 1
#> lag 0  1.0000 0.4042
#> lag 1 -0.3051 0.1291
#>
#> ma:
#>         lag 0    lag 1
#> lag 0  1.0000 -0.49690
#> lag 1 -0.1804  0.07785
```

Local polynomial regression with specified SARMA order. The `dcs()`-command is used with the default SARMA$((1,1),(1,1))$ model (correctly specified) and with an SARMA$((1,1),(0,0))$ (i.e. SAR$(1,1)$) model. The chosen estimation procedure is `"sep"`:

```
# SARMA((1, 1), (1, 1))
opt_sarma_1 = set.options(var_model = "sarma_sep")
dcs_sarma_1 = dcs(y_sarma, opt_sarma_1)

summary(dcs_sarma_1$var_est)
#> ------------------------------------------
#> Estimation of SARMA((1,1),(1,1))
#> ------------------------------------------
#> sigma:        0.5441
#> stationary:   TRUE
#> ar:
#>          lag 0   lag 1
#> lag 0  1.0000  0.5001
#> lag 1 -0.5246 -0.2623
#>
#> ma:
#>          lag 0    lag 1
#> lag 0  1.0000 -0.47060
#> lag 1 -0.1524  0.07174

# SARMA((1, 1), (0, 0))
opt_sarma_2 = set.options(var_model = "sarma_sep",
                          model_order =
                          list(ar = c(1, 1), ma = c(0, 0)))
dcs_sarma_2 = dcs(y_sarma, opt_sarma_2)

summary(dcs_sarma_2$var_est)
#> ------------------------------------------
#> Estimation of SARMA((1,1),(0,0))
#> ------------------------------------------
#> sigma:        0.5734
#> stationary:   TRUE
#> ar:
#>         lag 0   lag 1
#> lag 0  1.000  0.7089
#> lag 1 -0.397 -0.2814
#>
#> ma:
#>        lag 0
#> lag 0     1
```

Local polynomial regression with automated order selection. Automated order selection is used with `model_order = c("aic", "bic", "gpac")` in `set.options()`. The first one minimizes the AIC, the second one the BIC and the third uses a generalized partial autocorrelation function for order selection (not available for SFARIMA estimation). Order selection for large data sets is slowly in general, however, the `"gpac"` might be slightly faster than the other two. If automatic order selection is used, the argument `order_max` sets the maximum orders to be tested in the same way as `model_order` is usually defined as a list.

```
# BIC
opt_sarma_3 = set.options(var_model = "sarma_HR",
                          model_order = "bic",
                          order_max =
                          list(ar = c(2, 2), ma = c(2, 2)))
dcs_sarma_3 = dcs(y_sarma, opt_sarma_3)

summary(dcs_sarma_3$var_est)
#> ------------------------------------------
#> Estimation of SARMA((1,1),(1,1))
#> ------------------------------------------
#> sigma:        0.4994
#> stationary:   TRUE
#> ar:
#>         lag 0  lag 1
#> lag 0  1.0000 0.4027
#> lag 1 -0.3089 0.1267
#>
#> ma:
#>         lag 0    lag 1
#> lag 0  1.0000 -0.50360
#> lag 1 -0.1898  0.07345

# gpac
opt_sarma_4 = set.options(var_model = "sarma_HR",
                          model_order = "gpac",
                          order_max =
                          list(ar = c(2, 2), ma = c(2, 2)))
dcs_sarma_4 = dcs(y_sarma, opt_sarma_4)

summary(dcs_sarma_4$var_est)
#> ------------------------------------------
#> Estimation of SARMA((2,2),(2,0))
#> ------------------------------------------
#> sigma:        0.5065
#> stationary:   TRUE
#> ar:
#>          lag 0    lag 1    lag 2
#> lag 0  1.00000  0.85080  0.29000
#> lag 1 -0.44690 -0.13470 -0.04776
#> lag 2  0.01464 -0.02736 -0.01037
#>
#> ma:
#>          lag 0
#> lag 0  1.00000
#> lag 1 -0.32150
#> lag 2 -0.01187
```

A.3.4 Modeling Errors with Long Memory

This package includes a bandwidth selection algorithm when the errors $\varepsilon(x,t)$ follow an SFARIMA$((p_x,p_t),(q_x,q_t))$ process with long memory. Order selection for SFARIMA models works exactly as in the SARMA case. We use the same SARMA model as in 3.2 with long-memory parameters $d = (0.3, 0.1)$:

```
ar_mat = matrix(c(1, -0.3, 0.4, 0.12), nrow = 2, ncol = 2)
ma_mat = matrix(c(1, -0.2, -0.5, 0.1), nrow = 2, ncol = 2)
d = c(0.3, 0.1)
sigma  = sqrt(0.25)

model_list = list(ar = ar_mat, ma = ma_mat, d = d,
                  sigma = sigma)
sim_sfarima = sfarima.sim(n_x = 101, n_t = 101,
                          model = model_list)

# SFARIMA surface observations
y_sfarima = y.norm1 + sim_sfarima$Y

opt_sfarima = set.options(var_model = "sfarima_RSS")
dcs_sfarima = dcs(y_sfarima, opt_sfarima)

summary(dcs_sfarima$var_est)
#> ------------------------------------------
#> Estimation of SFARIMA((1,1),(1,1))
#> ------------------------------------------
#> d:            0.3149 0.1015
#> SD (sigma):   0.4966
#> stationary:   TRUE
#> ar:
```

```
#>           lag 0    lag 1
#> lag 0  1.00000  0.38640
#> lag 1 -0.09455 -0.03653
#>
#> ma:
#>          lag 0     lag 1
#> lag 0  1.0000 -0.521900
#> lag 1 -0.0157  0.008192
```

A.3.5 Estimation of Derivatives

Local polynomial estimation is suitable for estimation of derivatives of a function or a surface. While estimation of derivatives works as well under dependent errors, the example uses the iid. model from 3.1. Derivatives can be computed for any derivative vector `drv`, if the values are non-negative and an appropriate kernel function is chosen (such that the derivative order of the kernel matches the derivative order in `drv`). Note that the order of the polynomials for the νth derivative is chosen to be $p_i = \nu_i + 1, i = x, t$. As bandwidths increase with the order of the derivatives, the bandwidth might be large for higher derivative orders.

The estimator for $m^{(1,0)}(x, t)$ is

```
opt_drv_1 = set.options(drv = c(1, 0), kerns = c("MW_321", "MW_220"))
opt_drv_1$IPI_options$trim = c(0.1, 0.1)
dcs_drv_1 = dcs(y_iid, opt_drv_1)

dcs_drv_1
#> dcs
#> ------------------------------------------
#> DCS with automatic bandwidth selection:
#> ------------------------------------------
#>  Selected Bandwidths:
#>      h_x: 0.16354
#>      h_t: 0.23423
#>  Variance Factor:
#>      c_f: 0.99431
#> ------------------------------------------

# surface.dcs(dcs_drv_1, trim = c(0.1, 0.1), plot_choice = 2)
```

The estimator for $m^{(0,2)}(x, t)$ is

```
opt_drv_2 = set.options(drv = c(0, 2),
                        kerns = c("MW_220", "MW_422"))
opt_drv_2$IPI_options$trim = c(0.1, 0.1)

dcs_drv_2 = dcs(y_iid, opt_drv_2)

dcs_drv_2
#> dcs
#> ------------------------------------------
#> DCS with automatic bandwidth selection:
#> ------------------------------------------
#>  Selected Bandwidths:
#>      h_x: 0.21999
#>      h_t: 0.12777
#>  Variance Factor:
#>      c_f: 0.99431
#> ------------------------------------------
```

A.4 Mathematical Background

A.4.1 Double Conditional Smoothing

The double conditional smoothing (DCS, see Feng (2013)) is a spatial smoothing technique which effectively reduces the twodimensional estimation to two one-dimensional estimation procedures. The DCS is defined for kernel regression as well as for local polynomial regression.

Classical bivariate (and multivariate) regression has been considered e.g. by Herrmann et al. (1995) (kernel regression) and Ruppert and Wand (1994) (local polynomial regression). The DCS provides now a faster and, especially for equidistant data, more efficient smoothing scheme, which leads to reduced computation time. For the DCS procedure implemented in this package, consider a $(n_x \times n_t)$-matrix $\mathbf{Y}$ of non-empty observations $u_{i,j}$ and equidistant covariates X, T on $[0, 1]$, where X has length n_x and T has length n_t. The model is then

$$y_{i,j} = m(x_i, t_j) + \varepsilon_{i,j}$$

where $m(x, t)$ is the mean or trend function, $x_i \in X$, $t_j \in T$ and ε is a random error function with zero mean. The model in matrix form is $\mathbf{Y} = \mathbf{M}_1 + \mathbf{E}$ at the observation points.

The main assumption of the DCS is that of product kernels, i.e. the weights in the respective methods are constructed by $K(u, v) = K_1(u)K_2(v)$. Now, a two stage smoother can be constructed by either the kernel weights directly (kernel regression) or by using locally weighted regression with kernels K_1, K_2, in any case, the weights are called $\mathbf{W}_x$ and $\mathbf{W}_t$. The DCS procedure implemented in *DCSmooth* smoothes over rows (conditioning on X) first and then over columns (conditioning on T), although switching the smoothing order is exactly equivalent. Hence, the DCS is given by the following equations:

$$\widehat{\mathbf{M}}_0[, j] = \mathbf{Y} \cdot \mathbf{W}_t[, j]$$
$$\widehat{\mathbf{M}}_1[i,] = \mathbf{W}_x[i,] \cdot \widehat{\mathbf{M}}_0$$

A.4.2 Bandwidth Selection

The bandwidth vector $h = (h_x, h_t)$ is selected via an iterative plug-in (IPI) algorithm (Gasser et al., 1991). The IPI selects the optimal bandwidths by minimizing the mean integrated squared error (MISE) of the estimator. As the MISE includes derivatives of the regression surface $m(x, t)$, auxiliary bandwidths for estimation of these derivatives are calculated via an inflation method. These inflation method connects the bandwidths of $m(x, t)$ with that of a derivative $m^{(\nu_x, \nu_t)}(x, t)$ by

$$\widetilde{h}_k = c_k \cdot h_k^{\alpha}, \quad k = x, t$$

and is called exponential inflation method (EIM). The values of c_k are chosen on simulations, that of $alpha$ are subject to the derivative of interest. The IPI now starts with an initial bandwidth h_0 (chosen to be $h_0 = (0.1, 0.1)$) and calculates in each step s the auxiliary bandwidths $\widetilde{h}_{k,s}$ from h_{s-1} and h_s from the smoothed derivative surfaces using $\widetilde{h}_{k,s}$. The iteration process finishes until a certain threshold is reached.

A.4.3 Boundary Modification

In kernel regression, the boundary problem exists, which leads to biased estimated at the boundaries of the regression surface. This problem can (partially) be solved by means of suitable boundary kernels as introduced by Müller (1991) and Müller and Wang (1994). These boundary kernels differ in their degrees of smoothness and hence lead to different estimation results at the boundaries. However, all kernels are similar to the classical kernels in the interior region of the regression.

Following Chapter 2, a boundary modification is also defined for local polynomial regression. In the *DCSmooth* package, the local polynomial regression is always with boundary modification weights. Kernel types available (either for kernel regression or local polynomial regression) are Müller-type, Müller-Wang-type and truncated kernels, denoted by `M`, `MW` and `T`. In most applications, the Müller-Wang type are the preferred weighting functions.

For observations $X = x_i$, $x_i, i = 1, \dots, n_x$ and a given bandwidth h, define $u_r = \frac{x_r - X}{h} \in [-1, q]$. A (left) boundary kernel function $K_q^l(u)$ of order (k, μ, ν) is defined on $[-1, q]$, for $q \in [0, 1]$ and has the following properties

$$\int_{-1}^{q} u^j K_q^l(u) \mathrm{d}u = \begin{cases} 0 & \text{for } 0 \leq j < k, j \neq \nu \\ (-1)^\nu \nu! & \text{for } j = \nu \\ \beta \neq 0 & \text{for } j = k \end{cases}$$

The corresponding right boundary kernels can be calculated by $K_q^r(u) = (-1)\nu K_q(-u)$. The boundary kernels assigned by `kernel.assign()` are left boundary kernels.

A.4.4 Spatial ARMA Processes

The SARMA process $\varepsilon_{i,j}$ is given by the following equations:

$$\phi(B_1, B_2)\varepsilon_{i,j} = \psi(B_1, B_2)\eta_{i,j},$$

where the lag operators are $B_1 \varepsilon_{i,j} = \varepsilon_{i-1,j}$ and $B_2 \varepsilon_{i,j} = \varepsilon_{i,j-1}$, $\xi \underset{iid.}{\sim} \mathcal{N}(0, \sigma^2)$ and

$$\phi(z_1, z_2) = \sum_{m=0}^{p_1} \sum_{n=0}^{p_2} \phi_{m,n} z_1^m z_2^n, \ \psi(z_1, z_2) = \sum_{m=0}^{q_1} \sum_{n=0}^{q_2} \psi_{m,n} z_1^m z_2^n.$$

The coefficients $\psi_{m,n}$ and $\phi_{m,n}$ are written in matrix form

$$\boldsymbol{\phi} = \begin{pmatrix} \phi_{0,0} & \cdots & \phi_{0,p_2} \\ \vdots & \ddots & \\ \phi_{p_1,0} & & \phi_{p_1,p_2} \end{pmatrix} \text{ and } \boldsymbol{\psi} = \begin{pmatrix} \psi_{0,0} & \cdots & \psi_{0,q_2} \\ \vdots & \ddots & \\ \psi_{q_1,0} & & \psi_{q_1,q_2} \end{pmatrix},$$

where Φ is the AR-part (`$var_model$ar`) and Ψ is the MA-part (`$var_model$ma`). The example from 3.2,

$$\boldsymbol{\phi} = \begin{pmatrix} 1 & 0.4 \\ -0.3 & -0.12 \end{pmatrix} \text{ and } \boldsymbol{\psi} = \begin{pmatrix} 1 & -0.2 \\ -0.5 & 0.1 \end{pmatrix},$$

would then reduce to the process

$$\varepsilon_{i,j} = 0.4\varepsilon_{i,j-1} - 0.3\varepsilon_{i-,j} + 0.2\varepsilon_{i-1,j-1} + 0.2\xi_{i,j-1} + 0.2\xi_{i-1,j} - 0.5\xi_{i-1,j-1} + \xi_{i,j}.$$

Note that this process can be written as product of two univariate processes in the sense that

$$\phi_1(B_1)\phi_2(B_2)^T \varepsilon_{i,j} = \psi_1(B_1)\psi_2(B_2)^T \eta_{i,j},$$

with

$$\phi_1 = \begin{pmatrix} 1 \\ -0.3 \end{pmatrix}, \quad \phi_2 = \begin{pmatrix} 1 \\ 0.4 \end{pmatrix}, \quad \psi_1 = \begin{pmatrix} 1 \\ -0.5 \end{pmatrix}, \quad \psi_2 = \begin{pmatrix} 1 \\ -0.2 \end{pmatrix}.$$

Hence, these process forms a separable SARMA. Estimation of separable SARMA models can be reduced to estimation of univariate ARMA models.

A.4.5 Estimation of SARMA Processes

For estimation of SARMA models, three methods are implemented in *DCSmooth*:

Estimation of a Separable SARMA by ML-Estimation. This method is only available under the assumption of a separable model. Define two univariate time series

$$\varepsilon_{1,r} = \{\varepsilon_1\}_r = \{\varepsilon_{i,j}\}_{i+n_t(j-1)}, \quad \varepsilon_{2,s} = \{\varepsilon_2\}_s = \{\varepsilon_{i,j}\}_{j+n_x*(i-1)}$$

for $r, s = 1, \ldots, n_x \cdot n_t$, $i = 1, \ldots, n_x$, $j = 1, \ldots, n_t$. The parameters ϕ_1, ψ_1 of $\varepsilon_{1,r}$ and ϕ_2, ψ_2 of $\varepsilon_{2,s}$ can then be estimated by well-known maximum likelihood estimators.

Least Squares Estimation using the RSS. The SARMA model can be rewritten as

$$\eta_{i,j} = \psi(B_1, B_2)^{-1}\phi(B_1, B_2)\varepsilon_{i,j},$$

which allows for an $AR(\infty)$-representation of the SARMA model

$$\eta_{i,j} = \sum_{r,s=0}^{\infty} \theta_{r,s}^{AR}\varepsilon_{i-r,s-j}$$

From this, we can define the residual sum of squares (RSS) and get an estimate for the vector of coefficients $\theta = c(\phi_{1,0}, \phi_{0,1}, \ldots, \psi_{1,0}, \psi_{0,1}, \ldots$ by

$$\widehat{\theta} = \arg\min RSS \approx \arg\min \sum_{i,j=0}^{\infty} \eta_{i,j}^2.$$

Calculation of the $AR(\infty)$ representation of an SARMA model is difficult for a general SARMA but for a separable SARMA, the known univariate formulas hold. These procedure can be directly used for SFARIMA models, if the long memory parameter d is included in θ (see Beran et al., 2009).

The Hannan-Rissanen Algorithm. The previously defined estimation methods require numeric optimization of some quantities and hence take more time for calculation on a computer. The Hannan-Rissanen algorithm (Hannan and Rissanen, 1982) provides a much faster estimation procedure. An extension to SARMA models is straightforward:

The main idea Hannan-Rissanen algorithm is to use a high-order SAR auxiliary model for initial estimation of the unobservable innovations sequence. Then, a linear regression model is applied, which yields the SARMA-parameters from the observations and estimated innovations.

Let $\{\varepsilon\}_{i,j}$ be the the ordered observations of the $SARMA((p_x, p_t), (q_x, q_t))$ process $\varepsilon(x, t)$ with $i = 1, \ldots, n_x, j = 1, \ldots, n_t$. The SARMA parameters ϕ, ψ are then estimated by the modified Hannan-Rissanen algorithm: 1. Obtain the auxiliary residuals $\widetilde{\eta}_{i,j}$ by fitting a high-order autoregressive model with $(\widetilde{p}_x, \widetilde{p}_t) \geq (p_x, p_t)$ to the observations:

$$\widetilde{\eta}_{i,j} = \begin{cases} \sum\limits_{m=0}^{\widetilde{p}_1} \sum\limits_{n=0}^{\widetilde{p}_2} \widetilde{\phi}_{m,n} \varepsilon_{i-m,j-n}, & \widetilde{p}_1 < i \leq n_1, \widetilde{p}_2 < j \leq n_2 \\ 0, & 1 \leq i \leq \widetilde{p}_1, 1 \leq j \leq \widetilde{p}_2 \end{cases}$$

where $\widetilde{\phi}_{m,n}$ is estimated by the Yule-Walker equations and $\phi_{0,0} = 1$. 2. Obtain $\widehat{\phi}_{m,n}$ and $\widehat{\psi}_{m,n}$ and the estimated innovations $\widehat{\eta}_{i,j}$ by linear regression from

$$\varepsilon_{i,j} = - \sum_{\substack{m=0 \\ m \neq n = 0}}^{p_1} \sum_{\substack{n=0 \\ n \neq m = 0}}^{p_2} \widehat{\phi}_{m,n} \varepsilon_{i-m,j-n} + \sum_{\substack{m=0 \\ m \neq n = 0}}^{q_1} \sum_{\substack{n=0 \\ n \neq m = 0}}^{q_2} \widehat{\psi}_{m,n} \widetilde{\xi}_{i-m,j-n} + \widehat{\xi}_{i,j}.$$

The resulting coefficients $\widehat{\phi}$, $\widehat{\psi}$ are then the estimates for the parameters.

The autocovariance function of the SARMA-process is $\gamma(s, t) = \mathbb{E}(\varepsilon_{i,j} \varepsilon_{i+s,j+t})$. For $\widetilde{p}_x, \widetilde{p}_t$, the spatial Yule-Walker equation for the $SAR(\widetilde{p}_x, \widetilde{p}_t)$ approximation of the SARMA is then

$$\mathbf{\Gamma} \ \mathrm{vec}(\boldsymbol{\phi}) = 0$$

where $\mathbf{\Gamma}$ denotes the full autocovariance matrix:

$$\mathbf{\Gamma} = \begin{pmatrix} \gamma(0,0) & \gamma(1,0) & \ldots & \gamma(r_1,0) & \gamma(0,1) & \ldots & \gamma(r_1,r_2) \\ \gamma(1,0) & \gamma(0,0) & & & & & \gamma(r_1-1,r_2) \\ \vdots & & \ddots & & & & \vdots \\ \gamma(r_1,r_2) & \gamma(r_1-1,r_2) & \ldots & & & & \gamma(0,0) \end{pmatrix}$$

Bibliography

Altman, N. S. (1990). Kernel smoothing of data with correlated errors. *Journal of the American Statistical Association*, 85:749–759.

Altman, N. S. (1993). Estimating error correlation in nonparametric regression. *Statistics & Probability Letters*, 18:213–218.

Andersen, T. and Bollerslev, T. (1997). Intraday periodicity and volatility persistence in financial markets. *Journal of Empirical Finance*, 4(2-3):115–158.

Andersen, T. and Bollerslev, T. (1998). Deutsche mark-dollar volatility: Intraday activity patterns, macroeconomic announcements, and longer run dependencies. *Journal of Finance*, 53:219–265.

Andersen, T., Bollerslev, T., and Cai, J. (2000). Intraday and interday volatility in the japanese stock market. *Journal of International Financial Markets, Institutions and Money*, 10(2):107–130.

Andersen, T., Bollerslev, T., Diebold, F. X., and Labys, P. (2003). Modeling and forecasting realized volatility. *Econometrica*, 71:579–625.

Andersen, T. G., Bollerslev, T., and Meddahi, N. (2004). Analytical evaluation of volatility forecasts. *International Economic Review*, 45(4):1079–1110.

Aneiros-Pérez, G. and Vieu, P. (2008). Nonparametric time series prediction: A semi-functional partial linear modeling. *Journal of Multivariate Analysis*, 99(5):834 – 857.

Bartlett, M. S. (1975). *The Statistical Analysis of Spatial Pattern*, volume 15 of *Ettore Majorana International Science Series*. Springer Netherlands, 1 edition.

Bathia, N., Yao, Q., and Ziegelmann, F. (2010). Identifying the finite dimensionality of curve time series. *The Annals of Statistics*, 38:3352–3386.

Beran, J. and Feng, Y. (2002a). Iterative plug-in algorithms for semifar models – definition, convergence, and asymptotic properties. *Journal of Computational and Graphical Statistics*, 11(3):690–713.

Beran, J. and Feng, Y. (2002b). Local polynomial fitting with long-memory, short memory and antipersistent errors. *Annals of the Institute of Statistical Mathematics*, 54(2):291–311.

Beran, J., Ghosh, S., and Schell, D. (2009). On least squares estimation for long-memory lattice processes. *Journal of Multivariate Analysis*, 100(10):2178–2194.

Beyaztas, U. and Shang, H. L. (2019). Forecasting functional time series using weighted likelihood methodology. *Journal of Statistical Computation and Simulation*, 89(16):3046–3060.

Bühlmann, P. (1996). Locally adaptive lag-window spectral estimation. *Journal of Time Series Analysis*, 17(3):247–270.

Bosq, D. (2000). *Linear Processes in Function Spaces*. Lecture Notes in Statistics. Springer, New York, 1 edition.

Brabanter, K., Cao, F., Gijbels, I., and Opsomer, J. (2018). Local polynomial regression with correlated errors in random design and unknown correlation structure. *Biometrika*, 105:681–690.

Brailsford, T. J. (1996). The empirical relationship between trading volume, returns and volatility. *Accounting & Finance*, 36(1):89–111.

Brockwell, P. J. and Davis, R. A. (1991). *Time Series: Theory and Methods*. Springer Series in Statistics. New York, NY : Springer New York, second edition edition.

Cheng, M.-Y. (2006). Choice of the bandwidth ratio in rice's boundary modification. *Journal of the Chinese Statistical Association*, 44:235–251.

Cheng, M.-Y., Fan, J., and Marron, J. S. (1997). On automatic boundary corrections. *The Annals of Statistics*, 25(4):1691 – 1708.

Chiou, J.-M. and Müller, H.-G. (2009). Modeling hazard rates as functional data for the analysis of cohort lifetables and mortality forecasting. *Journal of the American Statistical Association*, 104:572–585.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74(368):829–836.

Delaigle, A. and Hall, P. (2010). Defining probability density for a distribution of random functions. *The Annals of Statistics*, 38(2):1171 – 1193.

Diamond, H. J., Karl, T. R., Palecki, M. A., Baker, C. B., Bell, J. E., Leeper, R. D., Easterling, D. R., Lawrimore, J. H., Meyers, T. P., Helfert, M. R., Goodge, G., and Thorne, P. W. (2013). U.s. climate reference network after one decade of operations: Status and assessment. *Bulletin of the American Meteorological Society*, 94:485–498.

Engle, R. F. and Sokalska, M. E. (2012). Forecasting intraday volatility in the us equity market. multiplicative component garch. *Journal of Financial Econometrics*, 10(1):54–83.

Facer, M. R. and Müller, H.-G. (2003). Nonparametric estimation of the location of a maximum in a response surface. *Journal of Multivariate Analysis*, 87(1):191–217.

Fan, J. and Gijbels, I. (1992). Variable bandwidth and local linear regression smoothers. *The Annals of Statistics*, 20:2008–2036.

Fan, J. and Gijbels, I. (1996). *Local polynomial modelling and its applications*, volume 66 of *Monographs on statistics and applied probability*. Chapman & Hall, London, 1 edition.

Feng, Y. (1999). *Kernel- and Locally Weighted Regression - with Application to Time*. Verlag für Wissenschaft und Forschung, Berlin, 1 edition.

Feng, Y. (2004). *Non- and Semiparametric Regression with Fractional Time Series Errors*. habilitation, University of Konstanz.

Feng, Y. (2013). Double-conditional smoothing of high-frequency volatility surface in a spatial multiplicative component garch with random effects. *Center for International Economics Working Paper*, 415:643–652.

Feng, Y., Beran, J., Gosh, S., Schäfer, B., and Letmathe, S. (2021a). A semiparametric generalization of spatial fractional arima processes. Preprint, Paderborn University.

Feng, Y., Letmathe, S., and Schulz, D. (2021b). *smoots: Nonparametric Estimation of the Trend and Its Derivatives in TS*. R package version 1.1.3.

Feng, Y. and McNeil, A. J. (2008). Modelling of scale change, periodicity and conditional heteroskedasticity in return volatility. *Economic Modelling*, 25(5):850–867.

Ferraty, F. and Vieu, P. (2006). *Nonparametric Functional Data Analysis*. Springer Series in Statistics. Springer, New York.

Fisher, W. D. (1971). Econometric estimation with spatial dependence. *Regional and Urban Economics*, 1(1):19–40.

Francisco-Fernández, M. and Vilar-Fernández, J. M. (2001). Local polynomial regression estimation with correlated errors. *Communications in Statistics - Theory and Methods*, 30(7):1271–1293.

Gao, Y. and Shang, H. L. (2017). Multivariate functional time series forecasting: Application to age-specific mortality rates. *Risks*, 5(2).

Gasser, T., Kneip, A., and Köhler, W. (1991). A flexible and fast method for automatic smoothing. *Journal of the American Statistical Association*, 86(415):643–652.

Gasser, T. and Müller, H.-G. (1979). Kernel estimation of regression functions. In Gasser, T. and Rosenblatt, M., editors, *Smoothing Techniques for Curve Estimation*, pages 23–68, Berlin, Heidelberg. Springer.

Gasser, T. and Müller, H.-G. (1984). Estimating regression functions and their derivatives by the kernel method. *Scandinavian Journal of Statistics*, 11:171–185.

Gasser, T., Müller, H.-G., and Mammitzsch, V. (1985). Kernels for nonparametric curve estimation. *Journal of the Royal Statistical Society*, 47:238–252.

Ghosh, S. (2015). Surface estimation under local stationarity. *Journal of Nonparametric Statistics*, 27(2):229–240.

Ghosh, S. (2018). *Kernel Smoothing: Principles, Methods and Applications*. Wiley, Hoboken, NJ.

Goodhart, C. A. and O'Hara, M. (1997). High frequency data in financial markets: Issues and applications. *Journal of Empirical Finance*, 4(2):73–114. High Frequency Data in Finance, Part 1.

Granovsky, B. L. and Müller, H.-G. (1991). Optimizing kernel methods: A unifying variational principle. *International Statistical Review / Revue Internationale de Statistique*, 59(3):373–388.

Ha, E. and Newton, J. H. (1993). The bias of estimators of causal spatial autoregressive processes. *Biometrika*, 80(1):242–245.

Hall, P., Müller, H.-G., and Wang, J.-L. (2006). Properties of principal component methods for functional and longitudinal data analysis. *The Annals of Statistics*, 34(3):1493–1517.

Hallin, M., Lu, Z., and Tran, L. T. (2004). Local linear spatial regression. *The Annals of Statistics*, 32(6):2469—-2500.

Hannan, E. J. and Rissanen, J. (1982). Recursive estimation of mixed autoregressive-moving average order. *Biometrika*, 69(1):81–94.

Hart, J. D. (1991). Kernel regression estimation with time series errors. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(1):173..187.

Hastie, T. and Loader, C. (1993). Local Regression: Automatic Kernel Carpentry. *Statistical Science*, 8(2):120 – 129.

Herrmann, E., Engel, J., Gasser, T., and Wand, M. P. (1995). A bandwidth selector for bivariate kernel regression. *Journal of the Royal Statistical Society*, 57:171–180.

Herrmann, E., Gasser, T., and Kneip, A. (1992). Choice of bandwidth for kernel regression when residuals are correlated. *Biometrika*, 79:783–795.

Horváth, L. and Kokoszka, P. (2012). *Inference for Functional Data with Applications*. Springer Series in Statistics. Springer, New York.

Härdle, W. and Müller, M. (2013). Multivariate and semiparametric kernel regression. In Schimek, M., editor, *Smoothing and Regression: Approaches, Computation, and Application*, Wiley Series in Probability and Statistics, pages 357–391. Wiley, New York.

Hyndman, R. J. and Shang, H. L. (2009). Forecasting functional time series. *Journal of the Korean Statistical Society*, 38:199–211.

Hyndman, R. J. and Shang, H. L. (2010). Rainbow plots, bagplots, and boxplots for functional data. *Journal of Computational and Graphical Statistics*, 19:29–45.

Hyndman, R. J. and Ullah, M. S. (2007). Robust forecasting of mortality and fertility rates: A functional data approach. *Computational Statistics & Data Analysis*, 51:4942–4956.

Illig, A. and Truong-Van, B. (2006). Asymptotic results for spatial arma models. *Communications in Statistics - Theory and Methods*, 35:671–688.

Kaiser, M. S., Daniels, M. J., Furakawa, K., and Dixon, P. (2002). Analysis of particulate matter air pollution using markov random field models of spatial dependence. *Environmetrics*, 13(5-6):615–628.

Karpoff, J. M. (1987). The relation between price changes and trading volume: A survey. *The Journal of Financial and Quantitative Analysis*, 22(1):109–126.

Karunamuni, R. J. and Alberts, T. (2005). On boundary correction in kernel density estimation. *Statistical Methodology*, 2(3):191–212.

Karunamuni, R. J. and Zhang, S. (2008). Some improvements on a boundary corrected kernel density estimator. *Statistics & Probability Letters*, 78(5):499–507.

Koláček, J. and Horová, I. (2017). Bandwidth matrix selectors for kernel regression. *Computational Statistics*, 32:1027—1046.

Kyung-Joon, C. and Schucany, W. R. (1998). Nonparametric kernel regression estimation near endpoints. *Journal of Statistical Planning and Inference*, 66(2):289–304.

Lee, B.-S. and Rui, O. M. (2002). The dynamic relationship between stock returns and trading volume: Domestic and cross-country evidence. *Journal of Banking & Finance*, 26(1):51–78.

Lejeune, M. (1985). Estimation non-paramétrique par noyaux: Régression polynômiale mobile. *Revue de Statistiques Appliqées*, 33:43–68.

Lejeune, M. and Sarda, P. (1992). Smooth estimators of distribution and density functions. *Computational Statistics & Data Analysis*, 14(4):457–471.

Li, D., Robinson, P. M., and Shang, H. L. (2019a). Long-range dependent curve time series. *Journal of the American Statistical Association*.

Li, L., Lu, K., and Xiao, Y. (2019b). Wavelet thresholding in fixed design regression for gaussian random fields. *Journal of Fourier Analysis and Applications*, 25:3184–3213.

Loader, C. R. (1996). Change point estimation using nonparametric regression. *The Annals of Statistics*, 24(4):1667 – 1678.

Lockwood, L. J. and Linn, S. C. (1990). An examination of stock market return volatility during overnight and intraday periods, 1964-1989. *The Journal of Finance*, 45(2):591–601.

Machkouri, M. E. and Stoica, R. (2010). Asymptotic normality of kernel estimates in a regression model for random fields. *Journal of Nonparametric Statistics*, 22:955—971.

Mack, Y. P. and Müller, H.-G. (1989). Derivative estimation in nonparametric regression with random predictor variable. *Sankhyā, The Indian Journal of Statistics, Series A*, 51:59–72.

Manteiga, W. G., Miranda, M. M., and González, A. P. (2004). The choice of smoothing parameter in nonparametric regression through wild bootstrap. *Computational Statistics & Data Analysis*, 47:487—515.

Martin, R. J. (1979). A subclass of lattice processes applied to a problem in planar sampling. *Biometrika*, 66(2):209–217.

Martin, R. J. (1990). The use of time-series models and methods in the analysis of agricultural field trials. *Communications in Statistics - Theory and Methods*, 19(2):55–81.

Martin, R. J. (1996). Some results on unilateral arma lattice processes. *Journal of Statistical Planning and Inference*, 50(3):395–411.

Masry, E. (1996). Multivariate local polynomial regression for time series: Uniform strong consistency and rates. *Journal of Time Series Analysis*, 17(6):571–599.

Mersmann, O. (2019). *microbenchmark: Accurate Timing Functions*. R package version 1.4-7.

Müller, H.-G. (1987). Weighted local regression and kernel methods for nonparametric curve fitting. *Journal of the American Statistical Association*, 82(397):231–238.

Müller, H.-G. (1988). *Nonparametric Regression Analysis of Longitudinal Data*. Lecture Notes in Statistics. Springer, Berlin Heidelberg, 1 edition.

Müller, H.-G. (1992). Change-points in nonparametric regression analysis. *The Annals of Statistics*, 20(2):737–761.

Müller, H.-G. (1993a). [local regression: Automatic kernel carpentry]: Comment. *Statistical Science*, 8(2):134–139.

Müller, H.-G. (1993b). On the boundary kernel method for non-parametric curve estimation near endpoints. *Scandinavian Journal of Statistics*, 20(4):313–328.

Müller, H.-G. and Prewitt, K. A. (1993). Mulitparameter bandwidth processes and adaptive surface smoothing. *Journal of Multivariate Analysis*, 47:1–21.

Müller, H.-G. (1991). Smooth optimum kernel estimators near endpoints. *Biometrika*, 78:521–530.

Müller, H.-G. and Wang, J.-L. (1994). Hazard rate estimation under random censoring with varying kernels and bandwidths. *Biometrics*, 50:61–76.

Opsomer, J., Wang, Y., and Yang, Y. (2001). Nonparametric regression with correlated errors. *Statistical Science*, 16(2):134–153.

Peitz, C. and Feng, Y. (2015). Double conditional smoothing of high-frequency volatility surface under a spatial model. In Beran, J., Feng, Y., and Hebbel, H., editors, *Empirical Economic and Financial Research*, pages 341–356. Springer, Heidelberg.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ramsay, J. O. and Dalzell, C. J. (1991). Some tools for functional data analysis. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):539–572.

Ramsay, J. O. and Silverman, B. W. (2005). *Functional Data Analysis*. Springer Series in Statistics. Springer, New York.

Ray, B. K. and Tsay, R. S. (1997). Bandwidth selection for kernel regression with long-range dependent errors. *Biometrika*, 84:791–802.

Reiss, P. T. and Ogden, T. (2007). Functional principal component regression and functional partial least squares. *Journal of the American Statistical Association*, 102(479):984–996.

Rice, J. (1984). Bandwidth Choice for Nonparametric Regression. *The Annals of Statistics*, 12(4):1215 – 1230.

Rice, J. A. and Silverman, B. W. (1991). Estimating the mean and covariance structure nonparametrically when the data are curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(1):233–243.

Robinson, P. M. (2007). Nonparametric spectrum estimation for spatial data. *Journal of Statistical Planning and Inference*, 137(3):1024–1034. Special Issue on Nonparametric Statistics and Related Topics: In honor of M.L. Puri.

Robinson, P. M. (2011). Asymptotic theory for nonparametric regression with spatial data. *Journal of Econometrics*, 165:5—19.

Robinson, P. M. (2020). Spatial long memory. *Japanese Journal of Statistics and Data Science*, 3:243–256.

Ruppert, D., Sheather, S. J., and Wand, M. P. (1995). An effective bandwidth selector for local least squares regression. *Journal of the American Statistical Association*, 90(432):1257–1270.

Ruppert, D. and Wand, M. P. (1994). Multivariate locally weighted least squares regression. *The Annals of Statistics*, 22(3):1346–1370.

Schäfer, B. (2021). *DCSmooth: Nonparametric Regression and Bandwidth Selection for Spatial Models*. R package version 1.1.2.

Scott, D. (2015). *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley Series in Probability and Statistics. Wiley, Hoboken, 2 edition.

Shang, H. L. and Hyndman, R. J. (2017). Grouped functional time series forecasting: An application to age-specific mortality rates. *Journal of Computational and Graphical Statistics*, 26:330–343.

Sievert, C. (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. Chapman and Hall/CRC.

Stone, C. J. (1977). Consistent nonparametric regression. *The Annals of Statistics*, 5(4):595–620.

Tenreiro, C. (2013). Boundary kernels for distribution function estimation. *REVSTAT Statistical Journal*, 11:169–190.

Tjøstheim, D. (1978). Statistical spatial series modelling. *Advances in Applied Probability*, 10:130–154.

Wand, M. P. (1994). Fast computation of multivariate kernel estimators. *Journal of Computational and Graphical Statistics*, 3:433–445.

Wang, H. and Wang, J. (2009). Estimation of the trend function for spatio-temporal models. *Journal of Nonparametric Statistics*, 21:567—-588.

Wang, L. and Cai, H. (2010). Asymptotic properties of nonparametric regression for long memory random fields. *Journal of Statistical Planning and Inference*, 140(3):837–850.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

Yang, L. and Tschernig, R. (1999). Multivariate bandwidth selection for local linear regression. *Journal of the Royal Statistical Society: Series B (Methodological)*, 61:793–815.

Yao, F., Müller, H.-G., and Wang, J.-L. (2005). Functional linear regression analysis for longitudinal data. *The Annals of Statistics*, 33(6):2873–2903.

Yao, Q. and Brockwell, P. J. (2006). Gaussian maximum likelihood estimation for ARMA models II: Spatial processes. *Bernoulli*, 12(3):403 – 429.

Yue, Y. and Speckman, P. L. (2010). Nonstationary spatial gaussian markov random fields. *Journal of Computational and Graphical Statistics*, 19:96–116.

Zhang, X., Brooks, R. D., and King, M. L. (2009). A bayesian approach to bandwidth selection for multivariate kernel regression with an application to state-price density estimation. *Journal of Econometrics*, 153:21—32.