

LOCAL REGRESSION MODELS: ADVANCEMENTS,
APPLICATIONS, AND NEW METHODS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Ryan P. Hafen

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2010

Purdue University

West Lafayette, Indiana

[Put dedication here.]

ACKNOWLEDGMENTS

[Put statement of appreciation or recognition of special assistance here.]

TABLE OF CONTENTS

	Page
LIST OF TABLES	vii
LIST OF FIGURES	viii
SYMBOLS	xv
ABBREVIATIONS	xix
ABSTRACT	xx
1 Background	1
1.1 Regression Models	1
1.2 Local Regression Models	1
1.2.1 Fitting a Local Regression Model	2
1.2.2 Loess as a Linear Operator	4
1.2.3 Statistical Properties	5
1.2.4 Diagnostics and Model selection	9
1.2.5 Computation	13
1.3 Time Series Models	14
1.3.1 ARMA Models	14
1.3.2 Nonstationarity and Trends	15
1.3.3 Linear Filters	16
1.3.4 Linear Filters in the Frequency Domain	17
1.4 Seasonal Trend Decomposition using Loess (STL)	20
1.4.1 The STL Method	20
1.4.2 STL Smoothing Parameters	24
1.4.3 STL Parameter Guidelines	25
1.4.4 Seasonal Parameter Selection	27
1.4.5 Post Smoothing	29
1.4.6 Computational Considerations	30
2 Advancements to the STL Method	31
2.1 The Operator Matrix of an STL Decomposition	31
2.1.1 Why Calculate the Operator Matrix?	31
2.1.2 How to Calculate the Operator Matrix	32
2.1.3 Variance Coefficients along the Design Space	36
2.1.4 Spectral Properties of Smoothing Near Endpoints	37
2.1.5 ARMA Modeling on Remainder Component	37
2.1.6 Predicting Ahead	39

	Page
2.1.7 Confidence and Prediction Intervals	42
2.1.8 Analysis of Variance	43
2.1.9 Model Selection	44
2.2 Local Quadratic Support and New Parameter Selection Guidelines .	45
2.2.1 Approximating the Critical Frequency for a Loess Filter . .	45
2.2.2 Calculating q Given ω , λ , and f_{crit}	48
2.2.3 Trend Filter Parameter Guidelines	48
2.2.4 High-Pass Filter Parameter Guidelines	49
2.3 The Revision Problem	49
2.3.1 Reproducing Kernel Hilbert Spaces with Loess	51
2.3.2 Blending to Lower Degree Polynomials at the Endpoints . .	52
2.3.3 Comparison of Blending and RKHS	60
2.4 Software Implementation: <code>stl2</code> and <code>operator</code> Packages	62
2.4.1 The <code>stl2</code> Package	63
2.4.2 The <code>operator</code> Package	64
2.4.3 Computational Considerations	65
2.4.4 Approximations	67
2.5 General Guidelines	68
3 An Application: Syndromic Surveillance	71
3.1 Background	71
3.2 Modeling the Data	72
3.2.1 Model Overview	73
3.2.2 Model Components and Parameter Selection	74
3.2.3 Model Diagnostics	76
3.2.4 Blending	76
3.2.5 Modeling on the Counts Scale	77
3.3 Outbreak Detection	78
3.3.1 Detection Method	78
3.3.2 Outbreak Model	79
3.3.3 Comparison of Methods	80
3.3.4 Outbreak Injection Scenarios	82
3.3.5 Choosing Cutoff Limits	83
3.4 Results	84
3.4.1 Results Summary	84
3.4.2 Examining the Difference in Detection Performance	86
3.4.3 False Positives	87
3.5 Discussion	87
3.6 Conclusions	88
3.7 Other Work	89
4 A New Density Estimation Method: Ed	91
4.1 Introduction	91

	Page
4.1.1 Background	91
4.1.2 Summary of Ed	92
4.1.3 Overview of the Chapter	93
4.2 Examples	94
4.3 Estimation Step 1: Raw Estimates	95
4.3.1 Choosing κ	97
4.3.2 Approximate Distribution of the Raw Estimates	98
4.4 Estimation Step 2: Loess Smoothing	100
4.5 Diagnostic Checking, Model Selection, and Inference	101
4.5.1 Evaluating Goodness-of-Fit	101
4.5.2 Mallows C_p Model Selection	102
4.5.3 Validating Model Assumptions	102
4.5.4 Inference	102
4.6 Comparison with Kernel Density Estimates	103
4.6.1 The Family-Income data	104
4.6.2 The Packet-Delay Data	105
4.6.3 The Normal-Mixture Data	105
4.7 Grid Augmentation	106
4.8 More Examples	108
4.9 Computational Methods	109
4.10 Discussion	110
5 Figures	113
5.1 Figures for Local Regression Modeling	113
5.2 Figures for Time Series Modeling	127
5.3 Figures for STL Advancements	139
5.4 Figures for Syndromic Surveillance	179
5.5 Figures for Ed	205
Appendix: R Package Documentation	243
LIST OF REFERENCES	273
VITA	279

LIST OF TABLES

Table		Page
2.1	Coefficient $\eta_{i,j}^{(\lambda)}$ values for $\lambda, i, j = 0, 1, 2$	47
3.1	Overall percentage of outbreaks detected by method	85

LIST OF FIGURES

Figure	Page
5.1 Data for loess example	113
5.2 Loess example - fit at $x = 0.5$	114
5.3 Loess fit for $\lambda = 2$, $\alpha = 0.4$	115
5.4 Variance coefficient along the design space for the density loess fit . .	116
5.5 Loess fitted values for several parameters	117
5.6 Loess residuals for several parameters	118
5.7 Mallows' C_p plot for the density data	119
5.8 Residual variance vs. ν for density data fits	120
5.9 Smoothing parameter α vs. equivalent number of parameters ν for den- sity data	121
5.10 Normal quantile plot for the density data loess fit with $\lambda = 2$, $\alpha = 0.4$	122
5.11 Residual serial correlation plot for density data loess fit	123
5.12 Residuals vs. design points for density data loess fit	124
5.13 Residuals vs. fitted values for density data loess fit	125
5.14 Loess fit for density data with exact fits at k -d tree vertices and inter- polation elsewhere	126
5.15 Two realizations of a unit-variance random walk process	127
5.16 Time series loess fit to simulated data	128
5.17 Power transfer functions for symmetric loess operators	129
5.18 CO ₂ concentration measurements at Mauna Loa, Hawaii from 1959 to 1997	130
5.19 STL decomposition for CO ₂ measurement time series	131
5.20 Power transfer function for each smoothing component of the CO ₂ STL decomposition	132
5.21 Seasonal diagnostic plot for CO ₂ decomposition	133

Figure	Page
5.22 Remainder component by cycle-subseries for CO ₂ decomposition . . .	134
5.23 Fitted seasonal component by cycle-subseries for CO ₂ data	135
5.24 Trend diagnostic plot for CO ₂ decomposition	136
5.25 CO ₂ decomposition with low-frequency post-trend smoothing	137
5.26 CO ₂ decomposition with two post-trend smoothing components . . .	138
5.27 Variance coefficients for CO ₂ decomposition components	139
5.28 Power transfer functions for STL operator matrix L^*	140
5.29 Autocorrelation function for CO ₂ remainder term and residual term after ARMA modeling	141
5.30 Normal quantile plot of remainder term of CO ₂ model after ARMA modeling on remainder term	142
5.31 Predicting the CO ₂ series ahead 3 years with 95% prediction intervals	143
5.32 Simulated data to mimic daily visit counts to an emergency department (ED)	144
5.33 C_p plot for CO ₂ decomposition with varying seasonal span $n_{(s)}$	145
5.34 Critical frequencies of a symmetric loess smooth vs. q^{-1}	146
5.35 Residuals from fitting critical frequencies vs. q^{-1} for a linear and quadratic model	147
5.36 Coefficients from quadratic polynomial fits of critical frequency vs. q^{-1} plotted against ω by λ	148
5.37 Plot of $n_{(p)}f_h^{(lower)}$ and $n_{(p)}f_h^{(upper)}$ vs. $\log_2 n_{(p)}$ for $\lambda_{(l)} = 0, 1, 2$ and $\omega = 0.05$	149
5.38 Variance coefficients along the design space for several choice of λ and q	150
5.39 Loess and RKHS loess operator weights for symmetric and endpoint fits with $q = 15$ and with $\lambda = 1$ and $\lambda = 2$	151
5.40 Symmetric $\lambda = 1$ span vs. endpoint $\lambda = 0$ span yielding equivalent variance	152
5.41 Symmetric $\lambda = 2$ span vs. endpoint $\lambda = 0$ and $\lambda = 1$ span yielding equivalent variance	153
5.42 Power transfer functions for loess fits with $\lambda = 0, 1, 2$ and q ranging from 7 to 35	154

Figure	Page
5.43 Simulated series from the sinusoid signal with added noise with $\sigma = 0.1$ and $\sigma = 1$	155
5.44 Simulated series from the ED signal with added noise with $\sigma = 0.1$ and $\sigma = 1$	156
5.45 Optimal proportion of blending δ vs. error standard deviation for sinusoid data	157
5.46 Optimal proportion of blending δ vs. error standard deviation for ED data	158
5.47 Optimal proportion of blending δ vs. number of points from end for sinusoid data	159
5.48 Optimal proportion of blending δ vs. number of points from end for ED data	160
5.49 MSRE loess / MSRE blend vs. σ for sinusoid data	161
5.50 MSRE loess / MSRE blend vs. σ for ED data	162
5.51 Power transfer functions for RKHS loess and loess with $\delta = 0, 0.5$, and 1	163
5.52 Phase shift function at the endpoint for RKHS loess and loess with $\delta = 0, 0.5$, and 1	164
5.53 Endpoint operator smoothing weights for RKHS loess and loess with $\delta = 0, 0.5$, and 1	165
5.54 Variance coefficients at and near the boundary for RKHS loess and loess with $\delta = 0, 0.5$, and 1	166
5.55 $\sqrt{\text{MSRE}}$ vs. σ for RKHS loess, traditional loess and loess blending with $\delta = 0.5$ for the sinusoid data	167
5.56 $\sqrt{\text{MSRE}}$ vs. σ for RKHS loess, traditional loess and loess blending with $\delta = 0.5$ for the ED data	168
5.57 MSRE RKHS / MSRE blend vs. σ for sinusoid data	169
5.58 MSRE RKHS / MSRE blend vs. σ for ED data	170
5.59 Endpoint operator power transfer functions for CO_2 decomposition fit operator for RKHS loess and blending with $\delta = 0$ and $\delta = 0.5$	171
5.60 Power transfer functions for STL operator matrix L^* with $\delta = 0.5$ blending for seasonal and trend.	172

Figure	Page
5.61 Variance coefficients for CO ₂ decomposition components with seasonal and trend blending with $\delta = 0.5$	173
5.62 STL decomposition for monthly CO ₂ concentration measurements with two years of data missing.	174
5.63 Run time for <code>stl2()</code> vs. run time for <code>stl()</code>	175
5.64 Run time vs. sample size for the <code>stl0p()</code> function, with a fitted cubic polynomial	176
5.65 Object storage size vs. sample size for the <code>stl0p()</code> function, with a fitted quadratic polynomial	177
5.66 Variance coefficient for CO ₂ approximate and actual STL decomposition components	178
5.67 STL decomposition of square root respiratory daily counts for an Indiana emergency department	179
5.68 Day-of-the-week component, d_t , for square root respiratory counts for 30 Indiana EDs.	180
5.69 Inter-annual component, t_t , for respiratory square root counts for 30 Indiana EDs	181
5.70 C_p plots for yearly-seasonal smoothing parameter for local quadratic fit to $\sqrt{y_t} - d_t - t_t$	182
5.71 Yearly-seasonal component, s_t , for square root respiratory daily counts for 30 Indiana EDs	183
5.72 Normal quantile plots for the noise component, n_t , of the square root respiratory counts for 30 EDs.	184
5.73 Sample autocorrelation functions for the noise component, n_t , of the square root respiratory counts for 30 EDs.	185
5.74 Day-of-the-week component diagnostic plot for one ED	186
5.75 Endpoint residuals from blending with $\delta = 0, 0.15$, and 0.5 for one ED	187
5.76 Residual standard deviation vs. mean daily count for each of the 30 EDs with and without the square root transformation.	188
5.77 Standard deviation for 100 simulated samples of size 10,000 from the square root Poisson distribution for various λ	189
5.78 Normal quantile plots for samples of size 500 from the square root Poisson distribution for various λ	190

Figure	Page
5.79 Distribution of $\hat{\sigma}_n$ for the 30 EDs.	191
5.80 Example of a randomly injected outbreak from the Sartwell model . .	192
5.81 Observed proportion of false positives vs. cutoff limit for the C1 method, by ED	193
5.82 EARS C1, C2, and C3 threshold levels for an empirical false positive rate of 0.03.	194
5.83 Observed false positive rate vs. cutoff limit for the GLM method . . .	195
5.84 Observed false positive rate vs. cutoff limit for the STL $q = 79$ method	196
5.85 Observed false positive rate vs. cutoff limit for the STL $q = 49$ method	197
5.86 GLM and STL threshold levels for an empirical false positive rate of 0.03.	198
5.87 Summary of proportion of outbreaks detected vs. outbreak magnitude by ED	199
5.88 Diagnostic plot of missed outbreaks for one ED by method	200
5.89 Residuals for model fits to daily respiratory counts for one ED	201
5.90 Fitted components for daily respiratory counts for one ED	202
5.91 Quantile plots of observed false positive rates for the STL and GLM methods based on $\rho = 0.03$	203
5.92 Temporal location of false positives for one ED by method.	204
5.93 Quantile plot for Packet-Delay data	205
5.94 Quantile plot for Family-Income data	206
5.95 Quantile plot for a sample of 3,000 observations from the Normal-Mixture example	207
5.96 Illustration of raw estimate calculation	208
5.97 Raw estimates for Packet-Delay data with $\kappa = 75$	209
5.98 Raw estimates for Family-Income data with $\kappa = 10$	210
5.99 Raw estimates for Normal-Mixture with $\kappa = 10$	211
5.100 Normal and log-gamma probability plots of the sample errors $\hat{y}_i^{(\kappa)} -$ $\log f(x_i^{(\kappa)})$ for the Normal-Mixture sample	212

Figure	Page
5.101 Variance of sample errors $\hat{y}_i^{(\kappa)} - \log f(x_i^{(\kappa)})$ for the Normal-Mixture samples	213
5.102 Loess fit to the Packet-Delay data on the log scale	214
5.103 Exponentiated loess fit for the Packet-Delay data	215
5.104 Loess fit with $\delta = 3$, $\alpha = 0.24$ to the Normal-Mixture raw estimates .	216
5.105 Exponentiated loess fit to the Normal-Mixture data	217
5.106 Residuals from loess fit for the Packet-Delay data	218
5.107 Residuals from loess fit for the Normal-Mixture data	219
5.108 Family-Income data log density fits for $\lambda = 2$ and different choices of α for local quadratic fitting.	220
5.109 Family-Income data log density residuals for $\lambda = 2$ and different choices of α for local quadratic fitting	221
5.110 C_p plot for the Family-Income data	222
5.111 Final loess fit with $\delta = 2$, $\alpha = 0.16$ for the Family-Income data. . . .	223
5.112 Final loess fit residuals with $\delta = 2$, $\alpha = 0.16$ for the Family-Income data.	224
5.113 Absolute residuals vs. income for Family-Income fit	225
5.114 Lag 1 correlation plot for Family-Income fit residuals	226
5.115 Normal quantile plot for Family-Income fit residuals	227
5.116 Pointwise 99% confidence limits for Family-Income log density estimate.	228
5.117 Kernel density estimate plot of Family-Income data	229
5.118 Final loess fit to Family-Income data on the identity scale, compared to the kernel density estimate with the Sheather-Jones bandwidth	230
5.119 Kernel density estimate plot of waiting time in queue for Packet-Delay data.	231
5.120 Kernel density estimate plots of 3,000 simulated values from the Normal-Mixture density	232
5.121 Plots of ed fit using grid augmentation	233
5.122 Ed raw estimates for VoIP call durations for attempted and connected calls.	234

Figure	Page
5.123 KDE estimates using Silverman's bandwidth on the log scale for VoIP call durations for attempted and connected calls	235
5.124 KDE estimates using bandwidths selected by unbiased cross-validation on the log scale for VoIP call durations for attempted and connected calls	236
5.125 KDE estimates using bandwidths selected by biased cross-validation on the log scale for VoIP call durations for attempted and connected calls	237
5.126 Log density raw estimates for bit-rate by call duration	238
5.127 28 benchmark densities	239
5.128 Ed raw estimates with true log density superposed for the 28 benchmark densities (page 1)	240
5.129 Ed raw estimates with true log density superposed for the 28 benchmark densities (page 2)	241
5.130 Ed raw estimates with true log density superposed for the 28 benchmark densities (page 3)	242

SYMBOLS

$'$	matrix transpose
\sim	distributed
$\lfloor \cdot \rfloor$	floor function
$\ \cdot \ $	L^2 norm
α	smoothing parameter (span); also, type I error rate
$A_{yx}(f)$	frequency response function for a linear operator a_t with input process y_t and output process y_t
$ A_{yx}(f) ^2$	power transfer function for a linear operator a_t with input process y_t and output process y_t
A	ARMA model operator matrix
$A^{(1:n)}$	matrix consisting of first n rows of some matrix A
$B(\cdot)$	bisquare weight function
$\hat{b}_i^{(\kappa)}$	balloon density estimate
χ^2	chi-square distribution
C_p	Mallows' estimator of M
C	cycle-subseries smoothing operator matrix
δ_1	trace of Λ
δ_2	trace of Λ^2
δ	proportion of blending
ϵ	vector of errors
$\hat{\epsilon}$	vector of residuals
e_i	white noise error process
f	Fourier frequencies; also, outbreak magnitude factor
f_{crit}	critical frequency
$f(\cdot)$	density function

$g_x(f)$	spectral density for a series x_t
$g_i^{(\kappa)}$	order statistic gaps
$\gamma(h)$	autocovariance function for values h unit apart in a series
H	high-pass smoothing operator matrix
I	$n \times n$ identity matrix
i	imaginary unit; also frequently used for vector subscripts
$K(\cdot)$	kernel function
κ	number of nearest neighbors for ed raw estimates
$l_i(x)$	loess operator weight for i th observation when fitting at x
$l(x)$	vector of loess operator weights when fitting at x
$\hat{l}^{(blend)}(x)$	linear operator coefficients for blended fit at x
L	loess operator matrix
$L_{\lambda,q}$	loess operator matrix from degree λ smoothing with span q
$L^{i,j}$	$n_{(p)} \times n_{(p)}$ matrix used to construct C
L^*	STL operator matrix
λ_i	mean of Poisson random variable at time i
λ	local polynomial degree
$\lambda_{(s)}$	STL parameter – degree for seasonal smoothing
$\lambda_{(t)}$	STL parameter – degree for trend smoothing
$\lambda_{(l)}$	STL parameter – degree for low-pass smoothing
λ_o, q_o	loess smoothing parameters blended from for a blended fit
λ_b, q_b	loess smoothing parameters blended to for a blended fit
Λ	$n \times n$ regulator matrix, equals $(I - L)'(I - L)$
M	moving average smoothing operator matrix; also, unit-less MSE: MSE/σ^2
\hat{M}	estimate of unit-less MSE, M
$\mu(x)$	mean function at point x
$\hat{\mu}(x)$	estimated mean function at point x
$\hat{\mu}^{(blend)}(x)$	estimated blended mean function at point x

n	number of observations
ν	trace of $L'L$, smoother degrees of freedom
$N(0, 1)$	standard normal distribution
O	outbreak magnitude for Sartwell model; also, Landau notation
ω	power transfer function cutoff for a critical frequency; also, grid augmentation parameter
p	number of parameters for parametric regression; also, order of an AR process
P	$n \times (n + 2n_{(p)})$ matrix of an $n \times n$ identity matrix surrounded by $n_{(p)}$ zero columns on either side
ϕ	vector of AR model parameters
$\Phi_{yx}(f)$	phase shift function for a linear operator with input process y_t and output process x_t
π_j	infinite autoregression coefficients for an ARMA process
q	number of observations per neighborhood, equals $\min(n, \alpha n)$; also, order of a MA process
$n_{(p)}$	STL parameter – number of observations per period
$n_{(s)}$	STL parameter – span for seasonal smoothing
$n_{(t)}$	STL parameter – span for trend smoothing
$n_{(l)}$	STL parameter – span for low-pass smoothing
r_i	remainder component of STL model at time i
\hat{r}_i	final estimate of remainder component of STL model at time i
ρ_i	STL robustness weights
s_i	seasonal component of STL model at time i
$s_i^{(k)}$	estimated seasonal component of STL model at k th iteration at time i
\hat{s}_i	final estimate of seasonal component of STL model at time i
S	seasonal smoothing operator matrix
S_k	seasonal smoothing operator matrix on iteration k

S^*	final seasonal smoothing operator matrix
σ^2	error variance
$\hat{\sigma}^2$	estimate of σ^2
t_i	trend component of STL model at time i
$t_i^{(k)}$	estimated trend component of STL model at k th iteration at time i
\hat{t}_i	final estimate of trend component of STL model at time i
T	trend smoothing operator matrix
T_k	trend smoothing operator matrix on iteration k
T^*	final trend smoothing operator matrix
$t_i^{(j)}$	j th post-trend component at time i
$T^{(j)}$	operator matrix for j th post-trend smoothing component
τ_x	variance coefficient for fitting at x
θ	vector of MA model parameters
$T(\cdot)$	tricube weight function
$w_i(x)$	weight of i th observation when fitting at x
W	weight matrix with diagonal entries $w_i(x)$
X	smoothing design matrix
X_i	zero-mean stochastic process
$x_i^{(\kappa)}$	location at which ed raw estimates are calculated
x_i	design points
y_i	response variable
$\hat{y}_i^{(\kappa)}$	ed raw estimate response variable

ABBREVIATIONS

ANOVA	analysis of variance
ARMA	autoregressive moving average
ARIMA	autoregressive integrated moving average
BLAS	basic linear algebra subroutines
E	expectation
ED	emergency department
EARS	early aberration reporting system
GLM	generalized linear model
i.i.d.	independent and identically distributed
KDE	kernel density estimate/estimation
MSE	mean squared error
MSRE	mean squared revision error
PHESS	public health emergency surveillance system
RHIPE	R Hadoop integrated processing environment
RSS	residual sum of squares
STL	seasonal trend decomposition using loess
tr	trace
Var	variance
VoIP	voice over IP

ABSTRACT

Hafen, Ryan P. Ph.D., Purdue University, May 2010. Local Regression Models: Advancements, Applications, and New Methods. Major Professor: William S. Cleveland.

Local regression methods model the relationship between an independent and dependent variable through weighted fitting of polynomials in local neighborhoods of the design space. Such methods can model a much wider class of functions than can be done with classical parametric models. A popular method, *loess*, is a local regression method with favorable statistical and computational properties. Loess modeling has been adapted to the modeling of time series data with the STL method (seasonal trend decomposition using loess). The first part of this work deals with some enhancements to the STL method. The second part presents an application of STL to syndromic surveillance data. Many of the improvements to STL were motivated by this application. Finally, a new modeling approach to nonparametric density estimation method, called *ed*, is presented which uses local regression to obtain density estimates.

While the STL method has many favorable properties such as simplicity and robustness, there are some aspects of its methodology and implementation that can be improved. Enhancements to the method presented in this work include support for local quadratic smoothing, missing values, and statistical inference. Also, a new method for dealing with smoothing at the endpoints is presented. This method, called blending, takes the endpoint fit to be a weighted average between the original endpoint fit and another fit of smaller degree. Blending is found to decrease the mean squared revision error (MSRE). Guidelines are given for the blending parameters. Software with these enhancements is also described.

Public health surveillance is the monitoring of data to detect and quantify unusual health events. Monitoring pre-diagnostic data, such as emergency department (ED) patient chief complaints, enables rapid detection of disease outbreaks. There are many sources of variation in such data; statistical methods need to accurately model them as a basis for timely and accurate disease outbreak methods. Methods for modeling daily chief complaint counts presented in this work are based on STL and were developed using data from the 76 EDs of the Indiana surveillance program from 2004 to 2008. Square root counts are decomposed into inter-annual, yearly-seasonal, day-of-the-week, and random-error components. Using this decomposition method, a new synoptic-scale (days to weeks) outbreak detection method was developed and evaluated by a simulation study, comparing detection performance to four well-known methods. The components of the STL decomposition reveal insights into the variability of the Indiana ED data. Day-of-the-week components tend to peak Sunday or Monday, fall steadily to a minimum Thursday or Friday, and then rise to the peak. Yearly-seasonal components show seasonal influenza, some with bimodal peaks. Some inter-annual components increase slightly due to increasing patient populations. A new outbreak detection method based on the decomposition modeling performs well with 90 days or more of data. Control limits were set empirically so that all methods had a specificity of 97%. STL had the largest sensitivity in all nine outbreak scenarios. The STL method also exhibited a well-behaved false positive rate when run on the data with no outbreaks injected. The STL decomposition method for chief complaint counts leads to a rapid and accurate detection method for disease outbreaks, and requires only 90 days of historical data to be put into operation. The visualization tools that accompany the decomposition and outbreak methods provide much insight into patterns in the data, which is useful for surveillance operations.

The ed method of density estimation for a univariate x takes a model building approach: an *estimation* method that can accurately fit many density patterns in data, and leads to *diagnostic* visual displays for checking fits for different values of the tuning parameters. The two-step estimator begins with a small-bandwidth bal-

loon density estimate, which is the inverse of the distance of x to the κ -th nearest neighbor. Next, loess is used to smooth the log of the balloon estimate as a function of x . This turns nonparametric density estimation into nonparametric regression estimation, allowing the full power of regression diagnostics, model selection, statistical inference, and computational methods. The theory of gaps of order statistics provides a tentative regression model with a known variance. For data with regions of x very sparse in data, pseudo-data can be added over the whole domain of x in the form of a grid, and the estimate corrected at the end. The ed method is straightforward. It deals well with problems encountered by other methods such as mis-fitting peaks and valleys, and it allows for simple identification and fitting of features such as discontinuities and boundary cut-offs all within the same framework.

1. BACKGROUND

This chapter introduces local regression and some time series modeling methods, with emphasis on a time series decomposition method: seasonal trend decomposition using loess (STL). The goal is to be brief but still cover the methods thoroughly, as much of the new work developed later depends heavily on these topics.

1.1 Regression Models

A common problem in statistics is describing the dependance of a response variable y on a predictor variable x . Given a data set with n pairs of observations, $(x_1, y_1), \dots, (x_n, y_n)$, consider a model of the form

$$y_i = \mu(x_i) + \epsilon_i \tag{1.1}$$

where $\mu(x)$ is an unknown function and the ϵ_i represent random errors in the observations. A common model is the normal regression model where the ϵ_i are independent normal random variables with mean 0 and variance σ^2 . All methods in this work are based on the normal model.

In the classical parametric regression setting, μ is specified as part of the model as a member of a parametric family, $\mu(x; \theta)$. For example, in the simple linear regression setting, $\mu(x; \theta) = \theta_0 + \theta_1 x_i + \epsilon_i$. The fitted regression function is found by obtaining an estimate, $\hat{\theta}$ for θ .

1.2 Local Regression Models

Often nature presents dependencies that are much more complex than can be captured with a simple parametric model. When the data suggest that $\mu(x)$ is a complex function, one may try to accommodate this by specifying a complex parametric

model with many parameters. Sometimes this can be successfully done, but finding the appropriate model can be very challenging and can result in difficult-to-interpret coefficients.

Another approach to modeling a complex μ that is much more simple and flexible is local regression. Instead of constraining $\mu(x)$ to have a parametric form, assume that the data *locally*, around some neighborhood of x , can be well-approximated by a parametric form – namely low order polynomials. If μ is a smooth differentiable function, then this assumption is simply an application of Taylor’s theorem. In this setting, obtaining the local regression fit, $\hat{\mu}(x)$, is based on choice of the size of the local neighborhood and the degree of the local polynomial.

An example that will be used throughout this section can be seen in figure 5.1. This data reflects a log density plus noise – a smoothing problem that presents itself in a new density estimation procedure (see chapter 4). There are 199 points. The difficulty of fitting a parametric model to this data is apparent, due to the curvature in the pattern of the data.

1.2.1 Fitting a Local Regression Model

While there are many ways to proceed in fitting local polynomials, this work focuses on a particular approach, *loess*. References for loess include Cleveland et al. (1988), Cleveland and Devlin (1988), Cleveland and Grosse (1991), Cleveland et al. (1992), and Cleveland and Loader (1996).

The loess fit at a particular point, say x , is obtained using weighted least squares to fit a polynomial of degree λ to the n_α observations nearest x . The quantity n_α is determined by a smoothing parameter α , where $n_\alpha = \min(n, \lfloor \alpha n \rfloor)$. A weight scheme is assigned to the n_α observations, with higher weight given to points closer to x . The weights are determined using the tricube weight function

$$T(u) = \begin{cases} (1 - u^3)^3 & \text{for } 0 \leq u < 1 \\ 0 & \text{for } u \geq 1. \end{cases} \quad (1.2)$$

Let $\Delta_\alpha(x)$ be the Euclidian distance from x to its n_α th nearest neighbor. Then the neighborhood weight, $w_i(x)$, for each x_i is obtained by

$$w_i(x) = T \left(\frac{\max(\alpha, 1)^{-1} |x_i - x|}{\Delta_\alpha(x)} \right) \quad (1.3)$$

which assigns weight 0 to observations outside of the α neighborhood.

Once the weights are obtained, a polynomial of degree λ is fit to the data, (x_i, y_i) , with weights w_i using weighted least squares regression. Local linear ($\lambda = 1$) or local quadratic ($\lambda = 2$) fitting are typically used. As an example, with local quadratic fitting, the model may be set up as

$$y = X\beta + \epsilon \quad (1.4)$$

locally around x , where $y = (y_1, \dots, y_n)'$, $\beta = (\beta_0, \beta_1, \beta_2)'$, and X is the design matrix centered at x

$$X = \begin{pmatrix} 1 & x_1 - x & (x_1 - x)^2 \\ 1 & x_2 - x & (x_2 - x)^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n - x & (x_n - x)^2 \end{pmatrix}. \quad (1.5)$$

With W as the weight matrix with diagonal entries w_i , minimizing the weighted least squares problem

$$(y - X\beta)'W(y - X\beta) \quad (1.6)$$

with respect to β yields the solution

$$\hat{\beta} = (X'WX)^{-1}X'Wy \quad (1.7)$$

Note that to simplify the computation, all rows of X , y , and W that have corresponding $w_i = 0$ (i.e. all rows corresponding observations that are not n_α nearest to x), can be excluded from the calculation, so that X is a $n_\alpha \times (\lambda + 1)$ matrix.

Now, since the design matrix is centered at x , the fit at x is simply

$$\hat{\mu}(x) = \hat{\beta}_0. \quad (1.8)$$

Figure 5.2 illustrates how the fit for the density data is obtained at $x = 0.5$ with $\lambda = 2$ and $\alpha = 0.4$. A local quadratic polynomial is fitted to the $q = 79$ nearest neighbors to $x = 0.5$, with weights as indicated in the top panel.

This covers how to compute the fit for a particular x . In practice, one usually wishes to obtain fits at the observed x_i or along a dense grid of points to obtain a regression surface. Efficient algorithms have been developed to obtain the fitted surface, explained briefly in section 1.2.5. A fit for the density example at all points is shown in figure 5.3.

A different parameterization of the loess smoothing span that is popular in time series applications is to express the span as an integer

$$q = \lfloor \alpha n \rfloor. \quad (1.9)$$

Note that when $\alpha \leq 1$, this is equivalent to the number of points in each smoothing neighborhood, n_α . This parameterization will be used the time series section of this chapter and in chapters 2 and 3.

1.2.2 Loess as a Linear Operator

A very important property of loess that will be frequently exploited throughout this work is that the fitted value $\hat{\mu}(x)$ can be written as a linear combination of the y_i

$$\hat{\mu}(x) = \sum_{i=1}^n l_i(x) y_i. \quad (1.10)$$

From (1.7) and (1.8), it can easily be seen that for $l(x) = (l_1(x), \dots, l_n(x))$,

$$l(x) = e_1' (X' W X)^{-1} X' W \quad (1.11)$$

where e_1 is a vector of length $\lambda + 1$, $e_1' = (1, 0, \dots, 0)$. From this it is clear that the $l_i(x)$ only depend on the design points.

An important case is when fitting at the design points, obtaining $\hat{y} = (\hat{\mu}(x_1), \dots, \hat{\mu}(x_n))'$. The fitted values can be written as

$$\hat{y} = Ly \tag{1.12}$$

where the i th row of L is $(l_1(x_i), \dots, l_n(x_i))$. L is called the *operator matrix*, and is similar to the hat matrix, H in parametric least-squares regression, the projection operator onto the space spanned by the fitting variables. L and H are similar in that they both relate the fitted values to the observed values through a linear transformation. Unlike H , the operator matrix L is neither symmetric ($L \neq L'$) nor idempotent ($L^2 \neq L$).

The virtue of the analogous roles of L and H in their respective settings is that many distributional results for the parametric setting are either the same or well-approximated in the local regression setting, enabling the use of familiar techniques for local regression estimates. These will be covered in the following section.

1.2.3 Statistical Properties

Here consider the normal errors model, with y having a multivariate normal distribution with mean vector $\mu = (\mu(x_1), \dots, \mu(x_n))'$ and covariance matrix $\sigma^2 I$, where I is the $n \times n$ identity matrix.

A critical assumption made throughout is that diagnostic checking has resulted in a model such that the bias $E[\hat{\mu}(x)] - \mu(x)$ is negligible. This is a reasonable assumption and is an assumption that is also made for all distributional results in parametric regression.

Normality of Fitted Values and Residuals

Due to linearity and normality, the fitted values

$$\hat{y} = Ly \tag{1.13}$$

and the residuals

$$\hat{\epsilon} = (I - L)y \quad (1.14)$$

have normal distributions with covariance matrices $\sigma^2 LL'$ and $\sigma^2(I - L)(I - L)'$ respectively. These results are the same as those for the parametric least-squares setting.

Distribution of the Residual Sum of Squares

The residual sum of squares (RSS) is defined as

$$\hat{\epsilon}'\hat{\epsilon} = y'\Lambda y \quad (1.15)$$

where

$$\Lambda = (I - L)'(I - L) \quad (1.16)$$

is the symmetric *regulator matrix*. The expected value of this quadratic form is

$$E[\hat{\epsilon}'\hat{\epsilon}] = \sigma^2 \text{tr}(\Lambda) + \mu'\Lambda\mu. \quad (1.17)$$

and the variance is

$$\text{Var}(\hat{\epsilon}'\hat{\epsilon}) = 2\sigma^4 \text{tr}(\Lambda^2) + 4\sigma^2 \mu'\Lambda^2\mu. \quad (1.18)$$

Under the assumption of $\hat{\mu}(x)$ estimating $\mu(x)$ with negligible or no bias, the second terms in the above expectation and variance vanish.

In the parametric setting, $\hat{\epsilon}'\hat{\epsilon}$ is the sum of p independent χ_1^2 random variables, resulting in a χ_p^2 distribution, where p is the rank of the design matrix X . In the local regression case, the distribution of $\hat{\epsilon}'\hat{\epsilon}$ is a linear combination of independent χ_1^2 random variables for which there are numerical algorithms to compute the distribution, such as [Farebrother \(1990\)](#). However, the distribution can be well-approximated by a χ^2 random variable. One approach is multiplying the RSS with a constant so that its first two central moments match those of the χ^2 distribution, akin to the idea by [Satterthwaite \(1946\)](#). With

$$\delta_1 = \text{tr}(\Lambda) \quad \text{and} \quad \delta_2 = \text{tr}(\Lambda^2) \quad (1.19)$$

one can construct a random variable

$$\zeta = \frac{\delta_1 \hat{\epsilon}' \hat{\epsilon}}{\delta_2 \sigma^2} \quad (1.20)$$

where from (1.17) and (1.18)

$$E[\zeta] = \delta_1^2 / \delta_2 \quad \text{and} \quad \text{Var}(\zeta) = 2\delta_1^2 / \delta_2. \quad (1.21)$$

Thus, ζ behaves approximately like a χ^2 distribution with δ_1^2 / δ_2 degrees of freedom.

Variance of the Fit at x and Confidence Intervals

To find the variance of $\hat{\mu}(x)$, use (1.10) and independence to obtain

$$\text{Var}(\hat{\mu}(x)) = \sigma^2 \sum_{i=1}^n l_i^2(x). \quad (1.22)$$

The quantity involving $l(x)$, call it

$$\tau_x = \sum_{i=1}^n l_i^2(x) \quad (1.23)$$

can be thought of as the *variance coefficient*, and it varies along the design space. Most notably, the variance coefficient increases dramatically at the boundaries of the design space. Dealing with this phenomenon will be of great importance in section 2.3. A plot of the variance coefficient along the design space for the density data can be seen in figure 5.4.

Estimating the variance of the fit is not complete without an estimate for the residual variance σ^2 . From (1.17), σ^2 is estimated by

$$\hat{\sigma}^2 = \hat{\epsilon}' \hat{\epsilon} / \text{tr}(\Lambda) \quad (1.24)$$

so that the estimate of the variance of $\hat{\mu}(x)$ is

$$\hat{\sigma}^2(x) = \hat{\sigma}^2 \sum_{i=1}^n l_i^2(x). \quad (1.25)$$

Now, from (1.20) and (1.24),

$$\zeta = \frac{\delta_1^2 \hat{\sigma}^2}{\delta_2 \sigma^2} \quad (1.26)$$

and since

$$\frac{\hat{\mu}(x) - \mu(x)}{\sigma} \sim N(0, 1) \quad (1.27)$$

$(\hat{\mu}(x) - \mu(x))/\hat{\sigma}$ can be approximated by a t distribution with δ_1^2/δ_2 degrees of freedom.

Thus, a level $1 - \alpha$ confidence interval for $\hat{\mu}(x)$ is

$$\hat{\mu}(x) \pm c\hat{\sigma}^2(x) \quad (1.28)$$

where c is the critical value of the t distribution with δ_1^2/δ_2 degrees of freedom such that $P(t < c) = \alpha/2$.

Smoother Degrees of Freedom

It is useful to be able to quantify the amount of smoothing that has been done by a loess smoothing operation. To borrow from parametric linear regression, this can be measured by the smoother degrees of freedom, or *equivalent number of parameters*, ν . There are multiple definitions, but the following will be used throughout this work

$$\nu = \text{tr}(L'L). \quad (1.29)$$

This definition is reasonable because for loess with independent errors

$$\sum_{i=1}^n \text{Var}(\hat{y}_i) = \sigma^2 \text{tr}(L'L), \quad (1.30)$$

and in the parametric regression case with p parameters

$$\sum_{i=1}^n \text{Var}(\hat{y}_i) = p\sigma^2. \quad (1.31)$$

Thus, ν is a loess analog of the number of parameters in parametric regression.

There are other definitions as well, such as $\nu = \text{tr}(L)$ and $\nu = \text{tr}(2L - L'L)$, based on analogs of other quantities involving p in parametric regression (Buja et al., 1989). Note that if L were symmetric and idempotent as in the parametric case, all of these definitions would be equivalent.

Analysis of Variance

Because of the approximate behavior of the RSS as a χ^2 random variable, statistical tests can be carried out to compare two models. Suppose N is the loess operator matrix for a model under a null hypothesis and A is the operator matrix for an alternative hypothesis. Let Λ_N and Λ_A be the regulator matrices (1.16) for N and A , respectively. Then $\text{RSS}_N = y'\Lambda_N y$ and $\text{RSS}_A = y'\Lambda_A y$. Analogous to parametric regression, testing the significance of the reduction in the residual sum of squares due to A can be done with the F test, with test statistic. Using the moment-matching approximation as done previously, let

$$\begin{aligned}\nu_1 &= \text{tr}(\Lambda_N - \Lambda_A) \\ \nu_2 &= \text{tr}((\Lambda_N - \Lambda_A)'(\Lambda_N - \Lambda_A)) \\ \delta_1 &= \text{tr}(\Lambda_A) \\ \delta_2 &= \text{tr}(\Lambda_A' \Lambda_A),\end{aligned}$$

Then the test statistic is

$$\hat{F} = \frac{(\text{RSS}_N - \text{RSS}_A)/\nu_1}{\text{RSS}_A/\delta_1} \quad (1.32)$$

whose approximate distribution is F with ν_1^2/ν_2 and δ_1^2/δ_2 degrees of freedom. A detailed study of the accuracy of the approximation used both here and in the section dealing with confidence intervals can be found in [Cleveland and Devlin \(1988\)](#).

Possible situations where such a test might be desirable could be the case where N is the operator matrix corresponding to parametric regression and it is desired to know whether a more flexible loess fit is necessary. Another application will be shown in chapter 2 when testing for periodicity in a time series.

1.2.4 Diagnostics and Model selection

Once a smooth L has been obtained, measures must be taken to ensure that it provides a good fit to the data and that assumptions have not been violated. The distributional results presented in this section depend on the following assumptions:

1. The fit follows the patterns in the data
2. The residuals are normally distributed
3. The residuals are independent
4. The residual variance is constant

Typically, one first finds the “best” fit by adjusting the smoothing parameters, satisfying (1), and then checks (3)–(4).

Visual Diagnostics

Recall that the smoothing parameters for loess are the degree of the local polynomial, λ , and the neighborhood bandwidth, α . These parameters can be chosen through trial-and-error with the help of some basic guidelines and visual diagnostics.

A general rule of thumb is to choose $\lambda = 1$ unless a scatterplot of the data indicates a pattern with substantial curvature such as peaks and valleys, in which case $\lambda = 2$ or possibly $\lambda = 3$ is recommended. For the bandwidth, if α is too small, the estimate will be too noisy, exhibiting superfluous wiggles. If α is too large, the estimate may miss key features due to oversmoothing, washing out small details. Typically a good fit is one in which the bias is as small as possible while keeping the fit as smooth as possible.

A powerful approach in regression for visualizing the bias-variance tradeoff is to view a scatterplot with the fit superposed in conjunction with a residual diagnostic plot (Cleveland, 1993). The fitted values scatterplot can help in determining if the fit is too smooth or too variable. The residual diagnostic plot can show local lack of fit, and one can judge the lack of fit based on knowledge of both the mechanism generating the data and knowledge of the performance of the smoothers used in the fitting.

Plots of fitted values and residuals for the density data for several smoothing parameters are shown in figures 5.5 and 5.6. Local linear, quadratic, and cubic fits

are considered with α ranging from small to large. Viewing the fitted values plot from top to bottom, the effects of undersmoothing to oversmoothing are apparent, with wiggly fits for small α and fits that miss the peaks and trough for large α . The residual plots have a solid line representing a loess smooth of the residuals to highlight the behavior of the mean of the residuals across the design space. A flat line in this plot indicates a good fit. Viewing the residual plot in conjunction with the fitted values plot, one might decide to go with a local quadratic or cubic model with α somewhere between 0.28 and 0.46. The local linear fits do not do well with the curvature, and the fits with small α are too noisy.

Mallows' C_p Model Selection Criterion

In addition to visual diagnostics, one may wish to use a model selection criterion when choosing parameters. One approach is generalizing Mallows' C_p (Mallows, 1973) to the local regression setting, as done by Cleveland et al. (1988). This criterion is based on the bias-variance decomposition of the mean squared error (MSE). For a given smoothing operator, L formed by smoothing parameters λ and α

$$\text{MSE} = \text{E}[(Ly - \mu)'(Ly - \mu)] \quad (1.33)$$

$$= (\text{E}[Ly] - \mu)'(\text{E}[Ly] - \mu) + \text{E}[y'L'Ly] - \text{E}[Ly]'\text{E}[Ly] \quad (1.34)$$

$$= \mu'\Lambda\mu + \sigma^2\nu \quad (1.35)$$

The first part of (1.35) is the bias sum of squares and the second part is the variance sum of squares. The bias depends on the unknown μ , so it must be estimated. Rearranging (1.17), the bias term can be written as

$$\mu'\Lambda\mu = \text{E}[\hat{\epsilon}'\hat{\epsilon}] - \sigma^2\text{tr}(\Lambda). \quad (1.36)$$

The C_p statistic, denoted by M , is simply the MSE divided by σ^2 , so with (1.35) and (1.36)

$$M = \frac{\text{E}[\hat{\epsilon}'\hat{\epsilon}]}{\sigma^2} - \text{tr}(\Lambda) + \nu. \quad (1.37)$$

This can be estimated by plugging in the observed RSS, $\hat{\epsilon}'\hat{\epsilon}$, and an unbiased estimate for the variance, $\hat{\sigma}^2$

$$\hat{M} = \frac{\hat{\epsilon}'\hat{\epsilon}}{\hat{\sigma}^2} - \text{tr}(\Lambda) + \nu. \quad (1.38)$$

The estimate $\hat{\sigma}^2$ can be obtained by smoothing the data with a small α .

When the fit follows the pattern in the data, the bias component becomes negligible and $E[\hat{M}] \approx \nu$. Good fits typically are ones for which \hat{M} is close to ν . However, the advised practice is to make a plot of \hat{M} vs. ν for several choices of α and λ and use this information to make better-informed parameter selection.

A C_p plot for the density example is shown in figure 5.7. The superiority of higher-degree local polynomials is evident for this data. Local cubic with degrees of freedom between 8.5 and 10 appears to be a good range fits – the bias is small, and smaller ν means smaller variance. Using figure 5.9 to translate ν into the corresponding neighborhood parameter α , this range covers fits around $\alpha = 0.45$. Figure 5.8 shows $\hat{\sigma}^2$ vs. ν . The residual variance begins to stabilize at $\nu > 8.5$ for local cubic, and this is where estimate for σ^2 used to calculate \hat{M} was obtained.

A very in-depth discussion on model selection in local regression can be found in [Tyner \(2007\)](#).

Assumption Validation

Once a good fit has been obtained, one must validate assumptions (2)–(4). Note that these are the same assumptions made in parametric regression and consequently many diagnostics used the parametric case can be used here. Item (2) can be checked by a normal probability plot. For (3), a serial correlation plot of ϵ_i vs. ϵ_{i-1} or in the case of time-series data, a plot of estimated autocorrelations of the ϵ_i can be used. For (4), one can make a plot of absolute residuals against design points to detect dependence of the residual variance on the predictor variable, or a plot of absolute residuals vs. the fitted values to detect dependence of the residual variance on the mean response.

Figures 5.10–5.13 show assumption validation plots for the density data fit shown previously with $\lambda = 2$ and $\alpha = 0.4$. From the C_p plot, these aren't a terrible choice for parameters, although local cubic would be better. Figure 5.10 supports the normality assumption. Figure 5.11 reveals a slight negative serial correlation for the higher residuals, but probably not strong enough to be concerned about. The residual variability seems to be roughly constant along the design space, as indicated in figure 5.12. It seems to have a slight dependence on the fitted values, shown in figure 5.13. The data are very sparse for the smaller fitted values, however. Overall, the assumptions for this particular fit seem to be reasonably valid.

1.2.5 Computation

A primer on loess would not be complete without a mention of computation. Direct calculation of local regression surfaces as described above can be very computationally expensive as the size of the data grows. Great care has been taken to make loess computationally feasible.

Without going too much into detail, the loess algorithm only directly computes the local regression estimate at a sufficiently dense set of points, and invokes a fast interpolation scheme to get the fitted surface everywhere else. The points at which to directly calculate the estimate are chosen as vertices on a k -d tree partition of the design space. The interpolation uses both the local fits at the vertices and, if $\lambda > 0$, the local slopes (which, from (1.7) are the $\hat{\beta}_1$ values). This approach keeps the computation time linear in n while still obtaining accurate estimates, making local regression feasible for large data sets.

Figure 5.14 shows the loess fit for the density data shown previously (figure 5.3) with dots indicating where the exact fit was obtained (the $k=d$ tree vertices), and a line indicating the interpolation. This data with $n = 199$ was fit with exact calculation at 17 nodes – significantly less computation than calculation at all 199 points or at a

dense grid of points along the design space. The figure also shows the exact fit along the design space. The interpolated fit is very true to the exact fit.

1.3 Time Series Models

A very brief background on some fundamental concepts in time series analysis will be given, focusing mainly on topics relevant to this work. These concepts include the autoregressive moving average (ARMA) family of models (Box et al., 1976), and some background in frequency domain time series analysis. Further reading on these topics include Cryer and Chan (2008), Shumway and Stoffer (2000), and Bloomfield (2004).

Consider a zero-mean stochastic process $\{X_i, i = 0, \pm 1, \pm 2, \dots\}$. The time units, i are equally-spaced and in a time series typically correspond to days, weeks, months, years, etc. Processes considered here are constrained to be stationary, meaning that the probability laws that govern the behavior of the process do not change over time.

A very common characteristic of time series data is the correlation of adjacent values in the series. The linear dependence between two values of a series h units apart by can be characterized by the *autocovariance function*, which for a zero-mean stationary series is

$$\gamma(h) = E[X_i X_{i-h}]. \quad (1.39)$$

If no correlation exists, classical statistical methods would suffice to analyze a series. When correlation does exist, models must account for this. A very useful and broadly applicable class of models to deal with correlation structures in time series data are ARMA models.

1.3.1 ARMA Models

Let $\{x_i, i = 0, \pm 1, \pm 2, \dots\}$ be an observed time series and $\{e_i, i = 0, \pm 1, \pm 2, \dots\}$ an unobserved white noise process of independent identically distributed zero-mean

random variables. One way to characterize the dependence of adjacent values in the time series is as a linear function of past values the of unobserved error process

$$x_i = e_i + \theta_1 e_{i-1} + \dots + \theta_q e_{i-q}. \quad (1.40)$$

Such a process is called a moving average process of order q , denoted as $MA(q)$.

Another way to characterize the dependence of adjacent values is by regressing the current observed value on past values

$$x_i = \phi_1 x_{i-1} + \phi_2 x_{i-2} + \dots + \phi_p x_{i-p} + e_i. \quad (1.41)$$

This is called an autoregressive process of order p , denoted by $AR(p)$.

When a series is a function of both its unobserved error process and its own past behavior, the $MA(q)$ and $AR(p)$ models can be mixed to obtain an autoregressive moving average $ARMA(p, q)$ model,

$$x_i = \sum_{j=1}^p \phi_j x_{i-j} + e_i + \sum_{j=1}^q \theta_j e_{i-j} \quad (1.42)$$

Selection of p and q and estimation of the θ_j and the ϕ_j are discussed in detail in several references such as [Shumway and Stoffer \(2000\)](#).

The ARMA class of models accommodates a large variety of real-world stationary time series. It will be used to model the remainder of decomposed time series in section [2.1.5](#).

1.3.2 Nonstationarity and Trends

Thus far, only stationary series have been considered. Nonstationary series can arise in a number of ways. Some nonstationary series exhibit what is called a stochastic “trend”. Such trends are called *nondeterministic*, i.e. the perceived fluctuations in the mean over time are stochastic and not intrinsic to the process. Perhaps the simplest example of this is a random walk process

$$y_i = y_{i-1} + e_i, \quad (1.43)$$

where the e_i are i.i.d. standard normal random variables. A realization of this process will exhibit what appears to be a trend, but the process has zero mean. The perceived trend is due to the strong correlation in the series and the variance that increases with i . Different realizations of the same process will exhibit very different “trends”. Figure 5.15 shows two simulated random walks to illustrate this.

The methods and applications in this work deal with time series that have trends that are considered to be *deterministic* – trends which typically can be explained by knowledge of the nature of the process. For example, consider the time series of monthly CO₂ concentrations measured at Mauna Loa, Hawaii from 1959 to 1997, measured in parts per million (ppm) (Keeling et al., 2001), shown in figure 5.18. There are two apparent trends – one is a long-term increase that looks to be linear, which is most-likely explainable by the increasing consumption of fossil fuels. Another trend is the seasonal oscillation within each year. A major portion of this work deals with methods for modeling time series with deterministic seasonal and trend components such as the CO₂ data.

1.3.3 Linear Filters

Deterministic trends are trends that after removed from the series result in a stationary process. One way to deal with deterministic trends is to estimate them first by regression and then remove them to model the stochastic properties of the remaining series. For example, consider the model

$$x_i = y_i + \varepsilon_i, \quad (1.44)$$

where y_i represents the mean function and ε_i is a zero-mean stationary process. Linear methods such as parametric regression or loess can be used to extract y_i . Any set of coefficients a_k that linearly transform an input series x_i to produce an output series y_i is called a *linear filter*,

$$y_i = \sum_{k=-\infty}^{\infty} a_k x_{i-k}. \quad (1.45)$$

Loess is a linear filter, where the coefficients a_k are computed as discussed in section 1.2.2.

Consider, for example, a simulated series x_i with deterministic trend y_i and with ε_i following an ARMA(1, 1) process with $\phi = -0.25$ and $\theta = 0.25$. Figure 5.16 shows such a process with the red dashed line representing the true mean function y_i and the solid black line representing the loess fit. The loess fit is shown for $q = 85$ and $q = 15$. In the following section, properties of smoothing time series with loess will be discussed, and this is why smoothed fits for two different parameters are shown here.

Note that the ε_i process in the above example does not satisfy the assumption of an i.i.d. normal process upon which loess operates. Research has been done in nonparametric regression with correlated errors. Opsomer et al. (2001) point out that when traditional nonparametric regression methods are applied to data with correlated errors, automatic smoothing parameter selection methods can easily break down due to the perception of these methods that all the structure in the data is due to the mean function, which is not true when the errors are correlated. However, if the researcher has an idea of what represents a reasonable fit, loess can still be applied and the choice of smoothing parameters can be dictated by an understanding of the nature of the data. This can be aided by visualization as discussed in section 1.2.4. Also, in section 2.1.5, an approach is proposed to be able to use numerical selection criteria such as C_p (section 1.2.4) when smoothing time series even when the errors are correlated.

1.3.4 Linear Filters in the Frequency Domain

Linear filters such as loess filters are used to extract a signal from noise, in effect eliminating higher frequencies from the data. Thus, it is useful to study linear filters by examining their effect on the series in terms of frequencies. To do so, the notion of the spectral density of a time series must be introduced.

The Spectral Density

The spectral density of a time series encapsulates the information from the autocovariance function (1.39) in terms of cycles instead of in terms of lags by way of the discrete Fourier transform. The spectral density for a series x_i with autocovariance function $\gamma_x(h)$ is computed as

$$g_x(f) = \sum_{h=-\infty}^{\infty} \gamma_x(h) e^{-2\pi f h} \quad 0 \leq f \leq 0.5. \quad (1.46)$$

A peak in the spectral density indicates an important contribution to the variance from the components at frequencies in the corresponding interval, e.g. if a monthly time series exhibits annual cycles, there will be a large value for the spectral density at the frequency $f = 1/12$.

The Frequency Response of a Linear Filter

Linear filters modify the spectral characteristics of a time series. The properties of a linear filter a_k (1.45) can be studied by analyzing its discrete Fourier transform

$$A_{yx}(f) = \sum_{k=-\infty}^{\infty} a_k e^{-2\pi i f k} \quad (1.47)$$

which is called the *frequency response function*. Let $g_x(f)$ be the spectral density of the input series x_i at frequency f and $g_y(f)$ be the spectral density of the output series y_i . Then

$$g_y(f) = |A_{yx}(f)|^2 g_x(f). \quad (1.48)$$

Thus, the frequency response function completely determines the effect of the filter on the spectral density of the new process, modifying the power or variance of the series at each frequency. The term

$$|A_{yx}(f)|^2 \quad (1.49)$$

will be referred to as the *power transfer function* for filter coefficients a_k , and is invaluable for studying the effect of a linear filter on the spectral density.

A simple way to think of $|A_{yx}(f)|^2$ is that for frequencies where this is close to 1, any cyclical behavior in x_i at that frequency is not altered by the filter, or is “passed” by the filter to the output series, y_i . Frequencies at which $|A_{yx}(f)|^2$ is near 0 are not passed, and frequencies at which $|A_{yx}(f)|^2 > 1$ are amplified.

Consider the loess filter applied to the example in section 1.3.3 (figure 5.16). Figure 5.17 shows the power transfer functions for the symmetric fit with $q = 85$ and the fit with $q = 15$. Both filters pass low frequencies, and thus loess is called a *low-pass filter*. The $q = 85$ filter cuts most of the frequency content above $f = 0.02$ cycles per point, eliminating periods shorter than $1/0.02 = 50$ points, while the $q = 15$ filter cuts most frequencies above $f = 0.14$ eliminating periods shorter than about 7 points. The effect of the $q = 15$ smooth not cutting off higher frequencies is evident when comparing two fits in figure 5.16.

Another spectral property of linear filters is the *phase shift*, which is the time shift between the filtered cycle and the unfiltered cycle. For

$$S = \sum_{k=-\infty}^{\infty} a_k \sin(2\pi f k) \quad \text{and} \quad C = \sum_{k=-\infty}^{\infty} a_k \cos(2\pi f k), \quad (1.50)$$

the phase shift is calculated as

$$\Phi_{yx}(f) = \begin{cases} \tan^{-1}(S/C) & \text{if } C \neq 0 \\ \pi/2 & \text{if } C = 0, S > 0 \\ -\pi/2 & \text{if } C = 0, S < 0 \end{cases} \quad (1.51)$$

When smoothing with loess in the symmetric case, where the number of observations is the same at either side of the point being smoothed, the phase shift as defined in (1.51) is zero at any frequency, meaning that the cycles in the smoothed output series from retain the same timing as the input series. However, loess smoothing at the endpoints of a series begins to have an effect on shifting the timing, when the neighborhoods begin to become asymmetric. The phase shift and power transfer functions will play an important role in studying methods for obtaining better endpoint estimates, which is discussed in section 2.3.

1.4 Seasonal Trend Decomposition using Loess (STL)

Seasonal trend decomposition using loess (STL) is a method for decomposing a time series into seasonal, trend, and remainder components using local regression techniques. It was designed with the goals of being simple, flexible, robust, fast, and easy to implement. An in-depth treatment can be found in [Cleveland et al. \(1990\)](#). Other methods with similar goals include X-11/X-11-ARIMA ([Dagum, 1978](#)), which is extensively used by the U.S. Census Bureau to adjust economic time series, and STL’s predecessor, SABL ([Cleveland et al., 1978](#)). One of the main reasons STL was conceived was because of the complexity and slow computational properties of these two methods. Also, it was created to make an open system available to create a fertile environment for new ideas and improvements.

Historically, seasonal adjustment methods such as X-11 have been devised for the sole purpose of removing the “nuisance” seasonal component from economic time series so that the seasonally adjusted data can be analyzed or presented. The purpose of time series decomposition presented in this work is well beyond seasonally adjusting a series – it is more about modeling the series and understanding all the components of variability in the series, of which the seasonal is just a part.

1.4.1 The STL Method

The goal of STL is to decompose a time series, y_i , $i = 1, \dots, n$ into seasonal, trend, and remainder components, s_i , t_i , r_i , respectively

$$y_i = s_i + t_i + r_i. \tag{1.52}$$

The procedure is illustrated using the monthly CO₂ concentration time series. A plot of a STL decomposition can be seen in figure [\(5.19\)](#). For this data set, an annual periodicity is apparent and the STL model was fit accordingly. The periodic behavior is very strong with the oscillations peaking each May and reaching the valley in September or October. The trend is steadily increasing.

STL works by iterating through smoothing of the seasonal and trend components. The seasonal component is defined based on the periodicity of the time series and how many observations there are per period, denoted by $n_{(p)}$. For example, the CO₂ has yearly periodicity with 12 observations per period, so $n_{(p)} = 12$. For a daily time series with weekly periodicity, $n_{(p)} = 7$, etc.

All smoothing operations done in STL use loess. STL is a procedure for regular time series, so the design points of the smoothing operation are equally-spaced, and without loss of generality can be thought of as $1, \dots, n$. For time series smoothing, q is always an odd integer. Each smoothing operation in STL requires smoothing parameters to be specified, but many of them have nice defaults.

For the seasonal smoothing, observations are separated into *cycle-subseries*, which for the CO₂ data, would be a subseries of all January values, another of all February values, etc. These subseries are separately smoothed and then recombined. The trend smoothing is then done on the data with the previously-fit seasonal component removed. This is iterated until convergence. STL also has an outer loop which, after each running of the inner loop, computes robustness weights that are passed on to the inner loop, in effort to reduce the influence of outlying values.

The Inner Loop

The inner loop is basically an iteration between seasonal smoothing (smoothing of the cycle-subseries) and trend smoothing. For example, the in the CO₂ decomposition, the seasonal was obtained by smoothing the cycle-subseries using local linear smoothing with $q = 35$ and the trend was obtained by smoothing the deseasonalized data using local linear smoothing with $q = 19$. However, it is not quite this simple, as the seasonal and trend smoothing compete for variation in the series at low frequencies. To fix this, some extra work needs to be done to the seasonal smoothing step. To illustrate, the top two panels of figure 5.20 show the power transfer functions (1.49) for the symmetric trend and cycle-subseries smoothing of the CO₂ data. Both

smoothing procedures pass power at the lowest frequencies. To keep the two smoothing operations from competing, a *high-pass filter* is applied to the cycle-subseries smooth before trend smoothing, which is composed of moving averages and loess. The transfer function for the high-pass filter for the CO₂ decomposition is shown in the third panel in figure 5.20, and applying this filter to the cycle-subseries filter yields the final seasonal smoothing filter, shown in the bottom panel of the figure.

The inner loop iterates $n_{(i)}$ times, each time updating the estimate for s_i and t_i . Let $s_i^{(k)}$ and $t_i^{(k)}$ be the seasonal and trend component estimates on the k th pass. The outline below details how the updated values $t_i^{(k+1)}$ and $s_i^{(k+1)}$ are obtained.

1. *Detrending*: The first step is to detrend the series, calculating $y_i - t_i^{(k)}$. If $k = 1$ then this step is skipped. If y_i has missing values, t_i also has missing values at the same positions.
2. *Cycle-subseries smoothing*: each cycle-subseries is smoothed by loess with a span of $q = n_{(s)}$ and local polynomial degree $\lambda = \lambda_{(s)}$. Smoothed values are computed at all time positions, including ones with missing values. Also, the smoothed values are computed one position before the first observation and one position after the last observation. For example, when smoothing the January cycle-subseries for the CO₂ data, which ranges from 1959 to 1997, there would also be an estimate computed for 1958 and 1998. After smoothing, the collection of cycle-subseries are recombined chronologically to form a series $c_i^{(k+1)}$ which consists of $n + 2n_{(p)}$ values.
3. *High-pass filtering of $c_i^{(k+1)}$ to obtain seasonal component*: It would seem logical to call $c_i^{(k+1)}$ the seasonal component and move to the trend smoothing. However, to prevent low-frequency power from entering the seasonal component, a high-pass filter is first applied to $c_i^{(k+1)}$. This filter is achieved by applying a low-pass filter to $c_i^{(k+1)}$ and subtracting the result from $c_i^{(k+1)}$. The low-pass filter consists of a moving average of length $n_{(p)}$ followed by another moving average of length $n_{(p)}$, followed by a moving average of length 3, followed by a

loess smoothing with $\lambda = \lambda_{(l)}$ and $q = n_{(l)}$. The moving averages drop values at the endpoints and this is why the series $c_i^{(k+1)}$ was extended beyond the range of the data in step 2. Now the low-pass filtered series $d_i^{(k+1)}$ consists of n values. The seasonal component is now obtained by $s_i^{(k+1)} = c_i^{(k+1)} - d_i^{(k+1)}$.

4. *Deseasonlizing*: The original series is deseasonalized by subtracting the seasonal component computed in the previous step $y_i - s_i^{(k+1)}$. If y_i has missing values, the deseasonalized series also has missing values in the corresponding positions.
5. *Trend smoothing*: The deseasonalized series is smoothed using loess with $q = n_{(t)}$ and $\lambda = \lambda_{(t)}$, and smoothed values are computed at all time positions.

The seasonal smoothing is comprised of steps 2 and 3 and the trend smoothing is step 5. The inner loop is carried out $n_{(i)}$ times. After running the inner loop $n_{(i)}$ times to get the estimates \hat{s}_i and \hat{t}_i , then the remainder is simply calculated as

$$\hat{r}_i = y_i - \hat{t}_i - \hat{s}_i. \quad (1.53)$$

If outlying values are apparent in the time series, one may wish to run the outer loop to down-weight these values.

The Outer Loop

The outer loop calculates robustness weights to downplay the influence of aberrant observations. Let

$$h = 6\text{median}(|r_i|). \quad (1.54)$$

Then the robustness weight at time i is calculated as

$$\rho_i = B(|r_i|/h) \quad (1.55)$$

where B is the bisquare weight function

$$B(u) = \begin{cases} (1 - u^2)^2 & \text{for } 0 \leq u < 1 \\ 0 & \text{for } u \geq 1. \end{cases} \quad (1.56)$$

With the robustness weights, the inner loop is repeated with the loess neighborhood weights now being multiplied by the robustness weights ρ_i . The outer loop is carried out $n_{(o)}$ times.

1.4.2 STL Smoothing Parameters

There are many parameters for the STL procedure but fortunately many of them have nice defaults. Aside from the number of observations per period, $n_{(p)}$, which should be evident from the data, and the number of iterations through the loops, $n_{(i)}$ and $n_{(o)}$, the rest of the parameters are loess smoothing parameters: $(n_{(s)}, \lambda_{(s)})$ for the cycle-subseries smoothing, $(n_{(l)}, \lambda_{(l)})$ for the low-pass filter smoothing, and $(n_{(t)}, \lambda_{(t)})$ for the trend smoothing. The main parameters that need to be carefully chosen, however, are only $n_{(s)}$, $\lambda_{(s)}$, and $\lambda_{(t)}$. In fact, the original STL implementation constrains the parameters $\lambda_{(t)} = 1$ and $\lambda_{(s)} = 1$ so that one only needs to focus on choosing $n_{(s)}$. There are guidelines for automatically selecting the other parameters, discussed in the following section. An extension allowing local quadratic trend and seasonal smoothing and automatic parameter selection rules for the trend smoothing span can be found in section 2.2.

The span parameters for each loess smoothing step of the procedure are always taken to be odd. This is done because it keeps things symmetric when smoothing in the interior of the series, which leads to favorable spectral smoothing properties.

The decomposition for the CO₂ data shown in figure 5.19 was done with parameters $n_{(p)} = 12$ (annual periodicity), $\lambda_{(s)} = 1$, and $n_{(s)} = 35$, which lead to automatic parameter choices of $\lambda_{(t)} = \lambda_{(l)} = 1$, $n_{(t)} = 19$, and $n_{(l)} = 13$. The iteration parameters were $n_{(i)} = 2$ and $n_{(o)} = 0$. Due to the results of the following section, typically the number of passes through the inner loop can be very small. The outer loop is only necessary if it has been judged that the series contains outliers.

1.4.3 STL Parameter Guidelines

The reason for automatic selection of the parameters other than the seasonal window is mainly based on the premise that the seasonal component is the main component of interest in the STL decomposition. Thus the trend smoothing and low-pass filter parameters are chosen so that they do not get in the way of the seasonal smoothing. Once a satisfactory seasonal component has been obtained, further trend smoothing can be done on the series with the seasonal removed, discussed in section 1.4.5.

The notion of the trend smoothing “getting in the way” of the seasonal smoothing is simple to define and study in the frequency domain (section 1.3.4). In Cleveland et al. (1990), the design goals for parameter selection for STL were to choose smoothing parameters so that the trend and seasonal components do not compete for variation at the same frequencies. They studied the transfer functions for symmetric loess filters with the trend $|T_{yx}(f)|^2$, cycle-subseries $|C_{yx}(f)|^2$, high-pass filter $|H_{yx}(f)|^2$, and seasonal $|S_{yx}(f)|^2$. These transfer functions were studied under simplified assumptions, ignoring end effects and thus only studying symmetric filters. More details on how these transfer functions were calculated can be read in Cleveland et al. (1990). The exact transfer functions without ignoring end effects can be calculated using results that are presented hereafter in section 2.1.2.

The design goals are probably best described by first referring to a plot of the transfer functions. Figure 5.20 shows the trend, cycle-subseries, high-pass, and seasonal filter transfer functions for the CO₂ decomposition in figure 5.19. The trend transfer function is simply the transfer function of a loess symmetric filter with $\lambda_{(t)} = 1$ and $n_{(t)} = 19$. As can be seen, it passes low frequencies and removes frequencies higher than about 0.08. A larger span would result in more smoothing and hence the transfer function would reduce to 0 faster. The cycle-subseries filter passes frequencies on and around $1/n_{(p)} = 1/12$ and its harmonics. From this plot it is clear to see that for the trend smoothing not to interfere with the cycle-subseries smoothing, the trend trans-

fer function needs to decrease to 0 before the seasonal smoothing at $1/n_{(p)}$ occurs. In this example, if $n_{(t)}$ were any smaller, it would begin to interfere with the seasonal smoothing. There still remains the problem of the low power frequency passing of the cycle-subseries smoothing (e.g. around $f = 0$). This is why the high-pass filter was introduced. Applying the high-pass filter to the cycle-subseries smoothed values yields the final seasonal filter shown in the bottom panel. The high-pass filter needs to filter out the low frequency, but needs to pass everything else corresponding to the cycle-subseries smoothing at $1/n_{(p)}$ and beyond.

The vertical dashed lines in figure 5.20 show the *critical frequencies*, frequencies at which the transfer functions decrease below some specified value, ω . Let $f_t^{(lower)}(\omega)$ be the critical frequency for the trend smoothing, the point at which $|T_{yx}(f)|^2$ decreases to below ω . Also, let $f_c^{(lower)}(\omega)$ be the frequency at which $|C_{yx}(f)|^2$ decreases to ω starting from $f = 0$ and moving in the positive direction and let $f_c^{(upper)}(\omega)$ be the frequency at which $|C_{yx}(f)|^2$ decreases to ω starting from $f = n_{(p)}^{-1}$ and moving in toward $f = 0$. Finally, define $f_h^{(lower)}(\omega)$ to be the lowest frequency such that $|H_{yx}(f)|^2 = \omega$ and let $f_h^{(upper)}(\omega)$ be the highest frequency such that $|H_{yx}(f)|^2 = 1 - \omega$. With the critical frequencies defined as this way, the design goals can be summarized as the following: chose parameters such that

1. $f_t^{(lower)}(\omega) < f_c^{(upper)}(\omega)$
2. $f_h^{(lower)}(\omega) > f_c^{(lower)}(\omega)$
3. $f_h^{(upper)}(\omega) < f_c^{(upper)}(\omega)$.

The critical frequencies in figure 5.20 were calculated with $\omega = 0.05$, and the design criteria have been met. Thus the seasonal and trend smoothing do not interfere with each other.

Cleveland et al. (1990) calculate the bandwidth guidelines using $\omega = 0.05$. They noticed that at $\omega = 0.05$, the critical frequency for a loess smooth with $\lambda = 1$ and a span of q is approximately $f \approx 1.5q^{-1}$. With this approximation,

$$f_t^{(lower)}(0.05) \approx 1.5n_{(t)}^{-1}. \quad (1.57)$$

Now, let $|L_{yx}(f)|^2$ be the transfer function for a symmetric loess filter with parameters $n_{(s)}$ and $\lambda_{(s)}$. Then

$$|C_{yx}(f)|^2 = |L_{yx}(n_{(p)}^{-1}f)|^2 \quad (1.58)$$

so that

$$f_c^{(lower)}(\omega) \approx 1.5n_{(p)}^{-1}n_{(s)}^{-1} \quad (1.59)$$

and

$$f_c^{(upper)}(\omega) \approx 1.5n_{(p)}^{-1}(1 - n_{(s)}^{-1}) \quad (1.60)$$

Hence, for design goal 1, the trend smoothing parameter, $n_{(t)}$ is chosen by

$$n_{(t)} \geq \frac{1.5n_{(p)}}{1 - 1.5n_{(s)}^{-1}} \quad (1.61)$$

For design goals 2 and 3, an empirical study revealed that as long as $n_{(s)} > 7$ choosing $n_{(l)}$ to be the smallest odd integer greater than or equal to $n_{(p)}$ satisfies the criteria. Recall that the high-pass filter is obtained from a low-pass filter of three moving averages followed by loess smoothing with $\lambda_{(l)} = 1$ and span $n_{(l)}$.

The constraints of $\lambda_{(t)} = 1$ and $\lambda_{(s)} = 1$ can be too restricting (e.g. a seasonal or trend component with large amounts of curvature), and in section (2.4.1), these constraints are removed with a new implementation of the STL procedure.

1.4.4 Seasonal Parameter Selection

With the results of the previous two sections, the analyst need mainly be concerned with choosing the seasonal span and degree and the trend degree. The trend degree, $\lambda_{(t)}$ can be chosen simply based on the analyst's judgement of the amount of curvature present in the trend component, which can be done from looking at a time series plot of the raw data. The seasonal smoothing parameters can be chosen based on knowledge of the mechanism generating the data and the goals of the analysis, and also with the help of visualization, as described in section 1.2.4.

Seasonal Diagnostic Plots

The *seasonal fitted values plot* is a visual tool that helps determine, after accounting for trend, how much variation in the data should go to the seasonal component and how much should go to the remainder. This plot consists of scatterplots for each cycle-subseries of the seasonal plus remainder, with the mean of the cycle-subseries seasonal component subtracted. The mean is subtracted so that the data for each cycle-subseries plot is centered at zero. A line is superposed on each plot of the fitted seasonal component with the mean subtracted. In conjunction with this plot, one can view the *seasonal residual plot*, which is the remainder term by month with a loess line superposed to highlight deviations from zero. These plots can be viewed to balance the bias/variance tradeoff, as discussed in section 1.2.4.

Figures 5.21 and 5.22 show the seasonal fitted values and residual plots for the CO₂ decomposition. The smoothing parameters seem to be adequate to describe the variation in the data due to the seasonal smoothing, although it appears there is more seasonal variation present in the months of May and June that may not have been accounted for. A smaller $n_{(s)}$ might be in order, although Cleveland et al. (1990) support these parameter choices based on the notion that with this particular data, one would expect a smooth evolution of the seasonal over the years.

It is interesting to note from the figures that the seasonal CO₂ level has been increasing over time for the Spring months and decreasing over time for the Fall months. This can be seen even more clearly by a *cycle-subseries plot*, which plots the fitted seasonal by cycle (now without the cycle-subseries mean removed), with vertical lines emanating from the cycle-subseries midmean. Such a plot for the CO₂ fit can be seen in figure 5.23.

Trend Diagnostic Plot

The automatic trend smoothing span criterion chooses the smallest possible $n_{(t)}$ such that the trend smoothing does not interfere with the seasonal smoothing. Some-

times, if the seasonal smoothing span has been chosen to be very small, it may constrain the trend smoothing span to be too large and not perform well in capturing the trend component of the data. One can verify that the trend smoothing is satisfactory by viewing a *trend diagnostic plot*, which is a scatterplot of the trend plus remainder with a line of the fitted trend component superposed. This is viewed in conjunction with the residuals. The trend diagnostic plot for the CO₂ decomposition is shown in figure 5.24.

1.4.5 Post Smoothing

An important characteristic of the trend smoothing step of the STL procedure to remember is that this step mainly exists to help extract the seasonal component. The trend smoothing bandwidth is chosen automatically so as not to interfere with the seasonal smoothing. Hence, the resulting trend component in most cases may not represent what the analyst wants to think of as the “trend estimate”. What the analyst defines as the trend estimate may differ based on the data or purpose of analysis. One situation may call for interest in a long-term trend, while another situation may warrant the trend estimate to capture all remaining low-frequency fluctuation in the data. A trend estimate that makes sense to the analyst can be obtained by subtracting the fitted seasonal component from the data and applying further smoothing.

As an example, consider the CO₂ data set. The trend component was obtained from automatic parameter selection, and as seen in figure 5.19, it is a fairly regular upward trend which exhibits some bumps over time. Suppose the analyst instead wants a very low-frequency representation of the rise of CO₂ levels over time. This can be achieved by subtracting the seasonal estimate obtained as in section 1.4.1 from the observed series and smoothing the result with a low-frequency filter. Figure 5.25 shows the decomposition of the CO₂ data with a trend post-smoothing with $q = 201$ and $\lambda = 1$. In this plot, the scales for each panel are set to the range of the

data so that the patterns in each component can be more easily seen. The seasonal component of figure 5.25 matches that of figure 5.19, but the STL trend has been replaced by the post-smoothing long-term trend estimate. The extra wiggles that are apparent in the the trend estimate of the original STL fit have been transferred to the remainder term in new decomposition.

The peaks and valleys present in the remainder component of the long-term trend decomposition for the CO₂ data have been found to be correlated to the Southern Oscillation, a measurement of the difference in atmospheric pressure between Easter Island in the South Pacific and Darwin, Australia (Bacastow, 1976). One might wish to describe the total CO₂ concentration as a function of seasonal, long-term trend, and Southern Oscillation behavior. This is obtained by two post-smoothings – first on the data minus the seasonal component as done above, and then a higher-frequency smoothing on the data minus the seasonal minus the long-term trend to obtain the component attributable to the Southern Oscillation. Using $q = 201$ and $\lambda = 1$ as above for the first post smoothing and $q = 35$ and $\lambda = 2$ for the second post smoothing, the decomposition in figure 5.26 is obtained.

1.4.6 Computational Considerations

As with loess, STL is a very fast procedure. Each loess operation in the procedure is only carried out at a fixed subset of time points and interpolated elsewhere. Unlike the traditional loess interpolation, the STL implementation only linearly interpolates between points. Since the design space is equally-spaced, the points at which direct calculation is made are also equally-spaced. For each loess smooth in the procedure, the number of points to skip between each direct calculation is computed as $q/10$, where q is the loess smoothing bandwidth.

2. ADVANCEMENTS TO THE STL METHOD

2.1 The Operator Matrix of an STL Decomposition

As shown in section 1.4, the STL decomposition is simply an iteration of linear smooths by loess and moving averages, and hence the entire decomposition procedure is linear. In other words, the fitted values

$$\hat{y}_i = \hat{s}_i + \hat{t}_i \quad (2.1)$$

can be obtained by a linear operator L^*

$$\hat{y} = L^* y. \quad (2.2)$$

The goal is to calculate L^* , but this is much more difficult to do than for loess.

2.1.1 Why Calculate the Operator Matrix?

Being able to calculate the operator matrix makes all of the statistical properties of loess (see section 1.2.3) readily available for STL, provided that the assumptions hold. Calculations are made possible such as

- variance coefficients along the design space
- studying properties of the procedure at endpoints
- confidence intervals
- predicting ahead
- equivalent number of parameters, residual standard error, residual degrees of freedom

- ANOVA for comparing models, particularly testing for seasonality
- model selection criteria, such as Mallows C_p

2.1.2 How to Calculate the Operator Matrix

To introduce STL operator matrix calculation, the simplified case of no missing values and no user-specified weights or robustness weights is assumed. Also, to keep the presentation lucid, assume that the total number of points, n , is divisible by $n_{(p)}$ (i.e. there are $n/n_{(p)}$ full periods in the data). Relaxing all of these is a minor technical detail, but would make presentation more complicated.

Without loss of generality, the design points of a time series $y = (y_1, \dots, y_n)$ can be thought of as the integers $x = (1, \dots, n)$. The span of the loess smooth will be denoted by q , which is the number of nearest neighbors to the fitting point including x itself. To keep neighborhoods symmetric, q is constrained to be odd.

Calculating the operator matrix for a STL decomposition is done by first obtaining the operator matrices corresponding to each step of the STL decomposition, outlined in section 1.4.1. Here, C denotes the cycle-subseries smoothing operator matrix, H denotes the high-pass operator, S denotes the seasonal smoothing operator, and, T denotes the trend smoothing operator. Also $L_{\lambda,q}$ will denote the loess operator matrix obtained from degree λ local smoothing with span q .

Cycle-Subseries Operator

The cycle-subseries operator matrix, C is simply a collection of loess operators for each cycle-subseries. Let $m = n/n_{(p)}$, the number of observations in each cycle-subseries. For example, a monthly time series with $n_{(p)} = 12$ and $n = 120$ observations would have $m = 10$ observations in each cycle-subseries (e.g. 10 January values, etc.). Each cycle-subseries is smoothed by a matrix, $L_{\lambda_{(s)},n_{(s)}}$.

Let the design points of a cycle-subseries be $1, \dots, m$. The operator matrix as defined in section 1.2.2 would be a $m \times m$ matrix, but here, fitted values are needed not just at $1, \dots, m$, but at $0, \dots, m+1$. Adding a row to the top and bottom of the operator matrix corresponding to the extrapolated fits one time unit before and after the series yields a $(m+2) \times m$ matrix $L_{\lambda(s), n(s)}$.

Let $l_{i,j}$ denote the (i, j) th element of this matrix with $i = 0, \dots, m+1$ and $j = 1, \dots, m$. Each cycle-subseries is smoothed by $L_{\lambda(s), n(s)}$ and then the resulting fits are put back in order to obtain the cycle-subseries component. Thus, defining $L^{i,j}$ as the $n_{(p)} \times n_{(p)}$ matrix with diagonal entries $l_{i,j}$ and zeros elsewhere, the cycle-subseries operator is a $(n + 2n_{(p)}) \times n$ matrix

$$C = \begin{pmatrix} L^{0,0} & L^{0,1} & \dots & L^{0,m} \\ L^{1,0} & L^{1,1} & \dots & L^{1,m} \\ \vdots & \vdots & \ddots & \vdots \\ L^{m+1,0} & L^{m+1,1} & \dots & L^{m+1,m} \end{pmatrix} \quad (2.3)$$

High-Pass Filter Operator

The operator for the high-pass filter is obtained from a low-pass filter of three moving average operations – two of length $n_{(p)}$ and one of length 3 – followed by a loess smooth with parameters $(\lambda_{(l)}, n_{(l)})$. The operator for the moving averages will be a $n \times (n + 2n_{(p)})$ matrix with a diagonal band obtained as follows. Let a and b be sequences with elements 1 through $n_{(p)}$ equal to $1/n_{(p)}$ with all other entries equal to zero. Now let h be the convolution of a and b , i.e.

$$h_i = \sum_{k=-\infty}^{\infty} a_k b_{i-k}, \quad i = -\infty, \dots, \infty. \quad (2.4)$$

Now, let c be a sequence with elements 1 through 3 equal to $1/3$, with all other entries equal to zero, and let d be the convolution of c and h . The sequence d will have $2n_{(p)} + 1$ non-zero elements. These elements comprise the diagonal band of the $n \times (n + 2n_{(p)})$ moving average matrix, M .

The final step in the low-pass filter is to apply loess smoothing using the $n \times n$ operator, $L_{\lambda_{(l)}, n_{(l)}}$, so that the low-pass filter is obtained by $L_{\lambda_{(l)}, n_{(l)}} M$, which is a $n \times (n + 2n_{(p)})$ matrix. Now, let P be a $n \times (n + 2n_{(p)})$ matrix whose first $n_{(p)}$ columns consist of zeros, whose next n columns are the identity matrix, I , and whose last $n_{(p)}$ columns consist of zeros. Then the high-pass filter is simply computed as

$$H = P - L_{\lambda_{(l)}, n_{(l)}} M \quad (2.5)$$

which is a $n \times (n + 2n_{(p)})$ matrix.

The Seasonal Component Operator

The $n \times n$ seasonal operator, S is simply obtained

$$S = HC. \quad (2.6)$$

The Trend Component Operator

The trend operator matrix is simply the $n \times n$ loess operator

$$T = L_{\lambda_{(t)}, n_{(t)}}. \quad (2.7)$$

While this is the trend operator, the actual trend component is calculated by first removing the seasonal, and would be obtained by the operator

$$T(I - S) \quad (2.8)$$

Iteratively Obtaining the Fit

The matrices S and T defined in (2.6) and (2.7) only depend on the smoothing parameters. Now, the final STL fit is obtained by iterating through seasonal and trend fits. Let S_k and T_k denote the seasonal and trend operator matrices on the k th iteration. Let T_0 be a $n \times n$ matrix with all entries equal to 0. Then for $k \geq 1$,

$$S_k = S(I - T_{k-1}) \quad (2.9)$$

and

$$T_k = T(I - S_k). \quad (2.10)$$

Now let S^* and T^* be the final matrices obtained upon convergence. Then the seasonal component is calculated as

$$\hat{s} = S^*y \quad (2.11)$$

and the trend component is calculated as

$$\hat{t} = T^*y \quad (2.12)$$

Also of interest will be the overall operator matrix

$$L^* = S^* + T^* \quad (2.13)$$

from which the fitted values (2.2) are calculated.

Post-Smoothing

If further trend smoothing is desired, the operator matrix for additional trend component(s) will simply be loess operator matrices, and will operate on the data with the seasonal component removed, i.e. $(I - S^*)y$.

For example, with the CO₂ example in section 1.4.5 where two additional trend smooths were obtained, the final decomposition is of the form

$$\hat{y}_i = \hat{s}_i + \hat{t}_i^{(1)} + \hat{t}_i^{(2)} \quad (2.14)$$

where $\hat{t}_i^{(1)}$ corresponds to the estimated long term trend and $\hat{t}_i^{(2)}$ corresponds to the trend related to the southern oscillation. Let $T^{(1)}$ be the loess operator matrix for obtaining $\hat{t}^{(1)}$ and $T^{(2)}$ be the loess operator matrix for obtaining $\hat{t}^{(2)}$. Then in operator notation, these additional trend components are obtained by

$$\hat{t}^{(1)} = T^{(1)}(I - S^*)y \quad (2.15)$$

and

$$\hat{t}^{(2)} = T^{(2)}(I - S^* - T^{(1)})y \quad (2.16)$$

and in this case, the overall “fitted value” operator matrix for the decomposition would be

$$L^* = S^* + T^{(1)}(I - S^*) + T^{(2)}(I - S^* - T^{(1)}). \quad (2.17)$$

2.1.3 Variance Coefficients along the Design Space

Having the operator matrices for the seasonal, trend, and overall fit components enables calculation of the contribution of each component to the variance of the fit at each point. The variance coefficient (1.23) can be calculated at each design point using the operator matrices, where the i th row of the matrix corresponds to the smoothing coefficients at design point i .

Of course (1.23) depends on the assumption of independence of the data being smoothed which generally isn’t the case for time series data. However, it can be useful to study the variance coefficients along the design space under the assumption of independence regardless of whether it is a valid assumption to get a better understanding the effect of the different smoothing components of the STL decomposition on the variance. Here consider the variance of the smoothing matrices when applying them to independent mean-zero unit-variance Gaussian noise.

From (1.23), the variance coefficient for design point i for the overall STL fit is obtained by

$$\tau_i = \sum_{j=1}^n L_{ij}^{*2} \quad (2.18)$$

and can be likewise calculated for the seasonal and trend parts of the fit. Figure 5.27 shows the variance coefficient for the final seasonal smoothing matrix S^* , trend smooth T^* , and the overall fit L^* . The variance attributable to the trend smoothing increases drastically at the endpoints and accounts for most of the variance at the endpoints, while the seasonal variance begins increasing much earlier. This is due to smoothing a small number of observations ($n = 39$) per cycle-subseries with $n_{(s)} = 35$,

which allows only a few points in each subseries to be smoothed with a symmetric filter.

2.1.4 Spectral Properties of Smoothing Near Endpoints

One of the most beneficial aspects of computing the STL operator matrix is the ability to study the spectral properties of the smoothing operations at and near the endpoints of the series. Recall from section 1.4.3 that the original presentation of the STL procedure ignored end effects when studying the characteristics of the smoothing operators in the frequency domain. Since the endpoints are such a critical part of a time series model, it is important to understand what is really happening at these points.

The frequency response function for the overall STL fit at the design point i can be calculated by applying (1.47) using the i th row of the STL operator L^* . Figure 5.28 shows the power transfer function computed from the frequency response for the STL fit for the CO₂ data at design points 1, 10, and 234 (the middle of the series). At the endpoint, the power of the frequencies related to both the trend and seasonal smoothing are greatly amplified, and the trend smoothing is bleeding into the seasonal. At design point 10, the trend part of the smoothing is behaving well, but the seasonal filter is still behaving poorly. At the center of the series, where everything is symmetric, the transfer function looks as it should (see figure 5.20 for comparison).

The problems revealed in figure 5.20 are a significant concern. Methods for dealing with this problem are described in section 2.3

2.1.5 ARMA Modeling on Remainder Component

Many of the statistical properties of linear filters covered in section 1.2.3 operate under the assumption of independence. This assumption typically isn't met when modeling time series and limits the simplicity of the inference that can be done

in the independent case. One possibility to consider is that after removing all of the deterministic components of the series through STL decomposition, suppose the remaining noise, r_i , can be described as a stationary ARMA process. An ARMA model can be expressed in linear operator form, and such modeling can account for the dependence in r_i and be incorporated into the overall operator matrix. This leaves an independent error term and allows inference to be done.

Suppose that r_i follows an ARMA(p, q) process. Then from (1.42)

$$r_i = \sum_{j=1}^p \phi_j r_{i-j} + e_i + \sum_{j=1}^q \theta_j e_{i-j} \quad (2.19)$$

where the e_i are i.i.d. normal random variables. This model can be written in inverted form as an infinite autoregression

$$r_i = \pi_1 r_{i-1} + \pi_2 r_{i-2} + \pi_3 r_{i-3} + \dots + e_i \quad (2.20)$$

where the π_j are calculated recursively as

$$\pi_j = \begin{cases} \sum_{i=1}^{\min(j,q)} \theta_i \pi_{j-i} + \phi_j & 1 \leq j \leq p \\ \sum_{i=1}^{\min(j,q)} \theta_i \pi_{j-i} & j > p \end{cases} \quad (2.21)$$

with $\pi_0 = -1$. Thus, the ARMA model can be expressed as a linear function of past values and the fitted values can be expressed as

$$\hat{r}_i = \hat{\pi}_1 r_{i-1} + \hat{\pi}_2 r_{i-2} + \hat{\pi}_3 r_{i-3} + \dots \quad (2.22)$$

Thus for the vector $r = (r_1, \dots, r_n)$, an $n \times n$ operator matrix can be constructed

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & \dots \\ \hat{\pi}_1 & 0 & 0 & 0 & \dots \\ \hat{\pi}_2 & \hat{\pi}_1 & 0 & 0 & \dots \\ \hat{\pi}_3 & \hat{\pi}_2 & \hat{\pi}_1 & 0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \quad (2.23)$$

such that

$$\hat{r} = Ar. \quad (2.24)$$

This is only approximate because the series is made up of only a finite number of observations. For example, $\hat{r}_2 = \hat{\pi}_1 r_1$ although it should be $\hat{r}_2 = \hat{\pi}_1 r_1 + \hat{\pi}_2 r_0 + \hat{\pi}_3 r_{-1} + \dots$. Only the first few fitted values in the series are noticeably effected by this. A more proper approach would be to backcast the series and substitute the backcasted values for r_i , $t \leq 0$, but the ARMA operator in (2.23) is reasonable.

If an appropriate ARMA model has been found, the residuals

$$\hat{e}_i = \hat{r}_i - \hat{\pi}_1 r_{i-1} - \hat{\pi}_2 r_{i-2} - \hat{\pi}_3 r_{i-3} - \dots \quad (2.25)$$

should behave like i.i.d. normal random variables. The STL decomposition now looks like

$$\hat{y}_i = \hat{s}_i + \hat{t}_i + \hat{r}_i + \hat{e}_i. \quad (2.26)$$

The operator matrix of the overall STL fit, including the ARMA modeling of the remainder is

$$L^* = S^* + T^* + A(I - S^* - T^*) \quad (2.27)$$

and all of the statistical tools of section 1.2.3 should now be available.

The remainder component of the CO₂ STL model originally presented ($\lambda_{(s)} = 1$ and $n_{(s)} = 35$) exhibited autocorrelation, and an ARMA(2, 1) model with $\phi_1 = 1.11$, $\phi_2 = -0.33$, and $\theta_1 = -0.98$ removed this autocorrelation. Figure 5.29 shows the estimated autocorrelation function of \hat{r}_i and of \hat{e}_i after ARMA modeling. The ARMA model successfully removes the dependence. Also, figure 5.30 shows a normal quantile plot of the \hat{e}_i . The STL model combined with ARMA modeling on the remainder for this data have given an overall operator matrix L^* to which the tools of section 1.2.3 can be applied.

2.1.6 Predicting Ahead

The operator matrix formulation facilitates predicting future values of the time series based on the decomposition model. This is simply done by extrapolating the loess fits beyond the last data point. Of course the main assumption in doing so is that

the components of variation will continue to behave in the future as their most recent behavior suggests. A strong case for this may often be made for a seasonal component that exhibits very predictable behavior over time, but may be more difficult to make for a trend component that has exhibited a lot of fluctuation over time. Regardless of the data, predicting ahead a few points in time in this manner is reasonable.

Suppose one wishes to predict a time series N units into the future. The goal is to obtain a $(n+N) \times n$ operator matrix for the overall STL fit, L^* . To do so, consider the $(n+N) \times n$ versions of the loess smoothing matrices described in section 2.1.2 used to obtain L^* , where the extra N rows of each matrix correspond to extrapolating the loess fit N units beyond n . The general procedure outlined in section 2.1.2 is redefined below with the extrapolated matrices. Attention must be paid throughout to keeping the matrix operations conformable. The procedure is modified as follows.

Cycle-Subseries Operator

For simplicity, suppose that the number of units to predict ahead, N , is a multiple of $n_{(p)}$. Recall also the simplification of n being divisible by $n_{(p)}$, with the number of observations in each cycle-subseries $m = n/n_{(p)}$. The modified cycle-subseries operator matrix, C is obtained from a loess operator $L_{\lambda_{(s)}, n_{(s)}}$ which provides fitted values at $0, 1, \dots, m + N/n_{(p)} + 1$, resulting in a $(m + N/n_{(p)} + 2) \times m$ matrix. Now C is constructed as in (2.3), and is a $(n + N + 2n_{(p)}) \times n$ matrix.

High-Pass Filter Operator

The high-pass filter (2.5) is constructed from matrices P , M , and $L_{\lambda_{(l)}, n_{(l)}}$. The new versions of these matrices are simply obtained from extending them to accommodate the $n + N$ values instead of n . Thus M and P are $(n + N) \times (n + N + 2n_{(p)})$ matrices and $L_{\lambda_{(l)}, n_{(l)}}$ is a $(n + N) \times (n + N)$ matrix, so that H is a $(n + N) \times (n + N + 2n_{(p)})$ matrix.

The Seasonal, Trend, and Iterating

The seasonal operator S is obtained as in (2.6), yielding a $(n + N) \times n$ matrix. The trend operator, T is the $(n + N) \times n$ loess operator $L_{\lambda_{(t)}, n_{(t)}}$ for design points $1, \dots, n + N$.

Obtaining the iterated fit is the same as done previously, except that the matrix multiplications in (2.9) and (2.10) will not work with the new matrix dimensions. Let $A^{(1:n)}$ denote the first n rows of a $m \times n$ matrix, A , $m > n$. With this notation, and with T_0 as the zero $(n + N) \times n$ matrix, the iterated fit is obtained for $k \geq 1$

$$S_k = S(I - T_{k-1}^{(1:n)}) \quad (2.28)$$

and

$$T_k = T(I - S_k^{(1:n)}) \quad (2.29)$$

which yields the $(n + N) \times n$ matrices S^* and T^* .

ARMA Fit to Residuals

ARMA prediction is done by simply using the π weights (2.22) to obtain new fitted values. For example,

$$\hat{r}_{n+1} = \hat{\pi}_1 r_n + \hat{\pi}_2 r_{n-1} + \hat{\pi}_3 r_{n-2} + \dots \quad (2.30)$$

Predictions beyond $n + 1$ require the use of past predicted values where observed values are unavailable, e.g.

$$\hat{r}_{n+2} = \hat{\pi}_1 \hat{r}_{n+1} + \hat{\pi}_2 r_n + \hat{\pi}_3 r_{n-1} + \dots \quad (2.31)$$

To express ARMA prediction in terms of an operator matrix, the matrix A (2.23) is appended with N rows. From (2.30), the $(n+1)$ th row is simply made up of entries $(\hat{\pi}_n, \hat{\pi}_{n-1}, \dots, \hat{\pi}_1)$. For the $(n+2)$ th row, from (2.31),

$$\hat{r}_{n+2} = \hat{\pi}_1 \hat{r}_{n+1} + \sum_{i=2}^{n+1} \hat{\pi}_i r_{n-i+2} \quad (2.32)$$

$$= \hat{\pi}_1 \sum_{i=1}^n \hat{\pi}_i r_{n-i+1} + \sum_{i=1}^n \hat{\pi}_i r_{n-i+1} \quad (2.33)$$

$$= \sum_{i=1}^n (\hat{\pi}_1 \hat{\pi}_i + \hat{\pi}_{i+1}) r_{n-i+1} \quad (2.34)$$

and hence the $(n-i+1)$ th element of the $(n+2)$ th row of A is $(\hat{\pi}_1 \hat{\pi}_i + \hat{\pi}_{i+1})$. Similar recursion is carried out to obtain each additional row of A .

Obtaining Predicted Values

Once the $(n+N) \times n$ matrices S^* , T^* , and A (if necessary) are obtained, STL predictions are calculated simply by

$$\hat{y} = L^* y \quad (2.35)$$

which yields a vector of length $n+N$, the first n values corresponding to the fitted values, and the last N values corresponding to the predictions. L^* is calculated as either the sum of S^* and T^* if no ARMA modeling is done on the remainder, or

$$L^* = S^* + T^* + A(I - (S^* - T^*)^{(1:n)}) \quad (2.36)$$

if ARMA modeling is done.

2.1.7 Confidence and Prediction Intervals

With the overall decomposition matrix L^* , procedures for confidence and prediction intervals are straightforward. If the residuals $(I - L^*)y$ behave like independent identically distributed normal random variables, the results of section 1.2.3 can be applied. For confidence intervals about the true mean at any point in the series $1, \dots, n$,

the material in section 1.2.3 can be directly applied using L^* , with $l(i)$ being the i th row of L^* . Thus a level $1 - \alpha$ confidence interval for \hat{y}_i is

$$\hat{y}_i \pm c\hat{\sigma}||l(i)|| \quad (2.37)$$

where $\hat{\sigma}$ is calculated by (1.24) and c is the critical value of a t distribution with δ_1^2/δ_2 (1.19) degrees of freedom such that $P(t < c) = \alpha/2$.

Intervals for predictions in the range $n + 1, \dots, n + N$ can be calculated in the same manner, but now the interval is not for the mean, but for a new value, so the standard error term needs to take the variability of the error into account along with the uncertainty in \hat{y}_i ,

$$\text{Var}(y_i - \hat{y}_i) = \sigma^2(1 + ||l_i||^2). \quad (2.38)$$

Thus a level $1 - \alpha$ prediction interval for \hat{y}_i is

$$\hat{y}_i \pm c\hat{\sigma}\sqrt{1 + ||l(i)||^2}. \quad (2.39)$$

Figure 5.31 shows the CO₂ series predicted ahead 3 years with 95% prediction intervals. The model is the original STL model with ARMA modeling on the remainder, as described previously.

2.1.8 Analysis of Variance

The STL operator matrix can be used to test hypotheses. A natural application is testing whether a hypothesized seasonal term should be in the model or not. The seasonal component in the CO₂ data is very prominent, but consider another example of square root daily counts of visits to an emergency department (ED). Such a series is shown in figure 5.32. More background on this type of data is presented in chapter 3. Here the interest is whether there is a significant day-of-the-week effect. Model fitting and visualization strongly confirm this, but a formal test can also be reassuring.

Visual diagnostics have resulted in an STL decomposition with $n_{(p)} = 7$, $\lambda_{(t)} = 1$, $n_{(t)} = 39$, and with the seasonal day-of-week component strictly periodic. Note that

$\lambda_{(t)} = 2$ would be a better choice here due to the multiple peaks and valleys in the seasonal term, which will be discussed in section 2.2. Under the null hypothesis of no seasonality, a simple loess fit of $\lambda = 1$, $q = 39$ is used. Figure 5.32 shows the fitted values under the null and alternative hypothesis.

Let N be the operator matrix under the null hypothesis and A be the operator matrix under the alternative of seasonality. To test for seasonality, define $R_N = (I - N)'(I - N)$ and $R_A = (I - A)'(I - A)$ to obtain the residual sum of squares $y'R_N y$ and $y'R_A y$. Now, let $\nu_1 = \text{tr}(R_N - R_A)$, $\nu_2 = \text{tr}(R_N - R_A)^2$, $\delta_1 = \text{tr} R_A$, and $\delta_2 = \text{tr} R_A^2$. Then the test statistic is

$$\hat{F} = \frac{(y'R_N y - y'R_A y)/\nu_1}{(y'R_A y)/\delta_1} \quad (2.40)$$

with an approximate F distribution with degrees of freedom ν_1^2/ν_2 and δ_1^2/δ_2 .

It is important to stress that to use the test, the assumptions must be verified. For the disease counts example, the residuals from the STL fit behave like i.i.d. normal random variables with mean 0 and standard deviation 0.52.

The above test resulted in a p -value of 2.2×10^{-16} , which is overwhelming evidence for the existence of the day-of-the-week component.

2.1.9 Model Selection

With L^* in hand, model selection tools such as C_p can be used in addition to visual diagnostics for parameter selection. Typically, the seasonal and trend local polynomial degree are quite evident from the data. The parameter one may be most interested in is the choice of $n_{(s)}$, since $n_{(t)}$ is calculated automatically. In some cases, one may also be interested in making a custom choice of $n_{(t)}$ as well.

For the CO₂ data, the C_p statistic as discussed in section 1.2.4 was calculated for several choices of $n_{(s)}$, ranging from 25 to 125. This was done with and without ARMA modeling of the remainder to see how the results agree. Figure 5.33 shows the C_p plot with the no-ARMA modeling on the left and ARMA modeling on the right. The point through which the line crosses in both plots corresponds to $n_{(s)} = 45$ and

in both cases is the parameter that results in the lowest variance. Both plots suggest that perhaps a larger $n_{(s)}$ would be preferable to the $n_{(s)} = 35$ that has been used in previous models.

2.2 Local Quadratic Support and New Parameter Selection Guidelines

The seasonal and trend components of a time series may exhibit behavior that suggests the use of local quadratic smoothing. Allowing this as an option warrants a new look at recommendations for the smoothing parameters.

Along the lines of section 1.4.3, the goal is to provide automatic parameter selection guidelines. The same simplification of ignoring end effects is applied, noting that if there are concerns about end effects (as there should be), the user can view the frequency response functions of the smooths at the endpoints as discussed in section 2.1.4. Also, work in section 2.3 will help the frequency response to behave at the endpoints more like it does in the symmetric case. Ignoring endpoints greatly simplifies derivation and presentation of the guidelines.

2.2.1 Approximating the Critical Frequency for a Loess Filter

Recall that for a loess filter with transfer function $|L_{yx}(f)|^2$, the critical frequency f_{crit} is the smallest frequency for which

$$|L_{yx}(f_{crit})|^2 = \omega \quad (2.41)$$

The transfer function for symmetric loess filters is 1 at $f = 0$ and trails off to 0 as f increases. The cutoff level, ω , is a quantity suggesting at what point the amount of power passed by the transfer function is negligible. Cleveland et al. (1990) only considered $\omega = 0.05$. As noted by Findley and Monsell (1990), this cutoff is rather arbitrary and it would be nice for the guidelines to allow more flexibility in this regard. This work allows ω to be specified for a range from 0.05 to 0.2.

The guidelines presented here come from theory-based empirical study. Although it is easy to obtain the critical frequency for a loess filter directly for any given degree λ , span q , and cutoff ω , the final goal will be to choose q for a given λ and ω . Solving for q is difficult to do analytically, but there are simple approximate relationships between these variables that make things simple.

Let $f_{crit}(q; \omega, \lambda)$ be the critical frequency of a loess smooth as a function of the loess span q for a given cutoff ω and local polynomial degree λ . Examining f_{crit} for loess filters with ω , ranging from 0.05 to 0.2 in increments of 0.03 and $\lambda \in \{0, 1, 2\}$ (note that $\lambda = 0$ and $\lambda = 1$ yield equivalent results) and $q = 2i + 1$ for $i = 4, \dots, 45$, the relationship between the critical frequency and q^{-1} is nearly linear. Figure 5.34 shows the critical frequencies vs. q^{-1} . The fitted lines are quadratic polynomial fits to the data

$$f_{crit}(q; \omega, \lambda) \approx \beta_0^{(\omega, \lambda)} + \beta_1^{(\omega, \lambda)} q^{-1} + \beta_2^{(\omega, \lambda)} q^{-2} \quad (2.42)$$

In Cleveland et al. (1990), a more rough approximation of a linear polynomial with no intercept was used to arrive at the approximation of $f_{crit}(q; \omega = 0.05, \lambda = 1) \approx 1.5q^{-1}$. Although the relationship looks quite linear, a quadratic gives a much better approximation without too much additional effort. For example, figure 5.35 shows the residual critical frequency using a quadratic polynomial vs. using a linear with no intercept.

The β_i coefficients depend on ω and λ . While it is possible to calculate these for many choices of ω , it turns out that there is a nice relationship between ω and the β_i . Figure 5.36 shows the β_i s obtained from the regressions shown in figure 5.34 vs. the ω at each λ . For β_0 , a linear polynomial seems to work well. For β_1 and β_2 , however, a quadratic polynomial fit works better. This yields

$$\beta_i^{(\omega, \lambda)} \approx \eta_{i,0}^{(\lambda)} + \eta_{i,1}^{(\lambda)} \omega + \eta_{i,2}^{(\lambda)} \omega^2, \quad i = 0, 1, 2 \quad (2.43)$$

The coefficients $\eta_{i,j}^{(\lambda)}$ are given in table 2.1.

Now, to obtain an approximation to f_{crit} for a specified ω , q , and λ , the coefficients in table 2.1 can be used to get the β_i s and then the β_i s can be used to get f_{crit} . For

Table 2.1
Coefficient $\eta_{i,j}^{(\lambda)}$ values for $\lambda, i, j = 0, 1, 2$.

λ	i	$j = 0$	$j = 1$	$j = 2$
0, 1	0	1.0335×10^{-4}	-2.1665×10^{-4}	0.0
	1	1.426860	-3.150382	5.074818
	2	1.665341	-3.877194	6.469529
2	0	3.8109×10^{-6}	7.0850×10^{-4}	0.0
	1	2.240896	-3.304353	5.080994
	2	2.331143	-1.831482	1.854315

example, with $\omega = 0.05$, $q = 23$, and $\lambda = 2$, the coefficient $\beta_1^{(0.05,2)}$ would be calculated as

$$\begin{aligned}
 \beta_1^{(0.05,2)} &\approx \eta_{1,0}^{(2)}(0.05) + \eta_{1,1}^{(2)} + \eta_{1,2}^{(2)}(0.05)^2 \\
 &= 2.240896 + (-3.304353)(0.05) + (5.080994)(0.05)^2 \\
 &= 2.088381.
 \end{aligned}$$

Similarly, the coefficients $\beta_0^{(0.05,2)} \approx 3.92359 \times 10^{-5}$ and $\beta_2^{(0.05,2)} \approx 2.244205$, so that

$$\begin{aligned}
 f_{crit} &\approx \beta_0^{(0.05,2)} + \beta_1^{(0.05,2)}/23 + \beta_2^{(0.05,2)}/23^2 \\
 &= 3.92359 \times 10^{-5} + 2.088381/23 + 2.244205/23^2 \\
 &= 0.09508076.
 \end{aligned}$$

The true critical frequency is 0.09516264.

2.2.2 Calculating q Given ω , λ , and f_{crit}

The benefit of the approximation method outlined above is that it is now simple to calculate q for a given ω , λ , and f_{crit} by solving (2.42) for q

$$q = \frac{-\beta_1^{(\omega, \lambda)} - \sqrt{(\beta_1^{(\omega, \lambda)})^2 - 4(\beta_0^{(\omega, \lambda)} - f_{crit})\beta_2^{(\omega, \lambda)}}}{2(\beta_0^{(\omega, \lambda)} - f_{crit})}. \quad (2.44)$$

The actual span is calculated to be the next odd integer of $\lfloor q \rfloor$. Suppose, for example, that a value of $f_{crit} = 1/12$ is desired with $\omega = 0.05$ and $\lambda = 2$. Then by (2.44), $q = 25$.

Although the empirical study above considered a finite range of q , subsequent analysis using $q > 91$ still led to excellent approximations.

2.2.3 Trend Filter Parameter Guidelines

Allowing local quadratic smoothing for the trend and seasonal components adds two more parameters for the user to specify up-front – $\lambda_{(s)}$ and $\lambda_{(t)}$. Recall that these were constrained to 1 previously. These can be chosen based on the user's judgment of the curvature present in these components based on preliminary analysis. Also, as before, the user needs to specify $n_{(s)}$, the seasonal smoothing window.

Recall from section 1.4.3 the goal for choosing the trend filter parameters, requiring $f_t^{(lower)}(\omega) < f_c^{(upper)}(\omega)$. Given $\lambda_{(s)}$, $\lambda_{(t)}$, and $n_{(s)}$, first approximate $f_{crit}(n_{(s)}; \omega, \lambda_{(s)})$ using (2.42). Then by (1.58)

$$f_c^{(upper)} = n_{(p)}^{-1}(1 - f_{crit}(n_{(s)}; \omega, \lambda_{(s)})). \quad (2.45)$$

Now, since $f_t^{(lower)}$ is simply $f_{crit}(n_{(t)}; \omega, \lambda_{(t)})$, the rule $f_t^{(lower)}(\omega) < f_c^{(upper)}(\omega)$ is satisfied using (2.44) to obtain

$$n_{(t)} \geq \frac{-\beta_1^{(\omega, \lambda_{(t)})} - \sqrt{(\beta_1^{(\omega, \lambda_{(t)})})^2 - 4(\beta_0^{(\omega, \lambda_{(t)})} - f_c^{(upper)})\beta_2^{(\omega, \lambda_{(t)})}}}{2(\beta_0^{(\omega, \lambda_{(t)})} - f_c^{(upper)})}. \quad (2.46)$$

2.2.4 High-Pass Filter Parameter Guidelines

The high-pass filter depends on the loess smoothing parameters $\lambda_{(l)}$ and $n_{(l)}$. A good rule for $\lambda_{(l)}$ is to choose

$$\lambda_{(l)} = \lambda_{(t)} \quad (2.47)$$

since the high-pass smoothing needs to capture curvature in the trend if it is present.

The guideline given in section 1.4.3 of choosing $n_{(l)}$ to be the smallest odd integer greater than or equal to $n_{(p)}$ is now studied with the local quadratic trend and seasonal smoothing capabilities to see if it still holds. To investigate this guideline, consider a range of $n_{(p)}$ values

$$n_{(p)} = 2^{2+i/2}, \quad i = 0, \dots, 13 \quad (2.48)$$

rounded to the nearest integer. Then $f_h^{(lower)}$ and $f_h^{(upper)}$ are calculated as described in section 1.4.3. Figure 5.37 plots the values of $n_{(p)}f_h^{(lower)}$ and $n_{(p)}f_h^{(upper)}$ vs. $\log_2 n_{(p)}$ for $\lambda_{(l)} = 0, 1, 2$ and $\omega = 0.05$. The horizontal lines correspond to $n_{(p)}f_c^{(lower)}$ and $n_{(p)}f_c^{(upper)}$ for $\lambda_{(s)} = 0$ or 1 in the first panel and $\lambda_{(s)} = 2$ in the second panel. Each line corresponds to a different value of $n_{(s)}$, with

$$n_{(s)} = 7 + 2i, \quad i = 0, \dots, 12. \quad (2.49)$$

The horizontal lines with $n_{(s)}$ closest to the middle of the y -axis correspond to upper and lower critical frequencies with $n_{(s)} = 7$, with $n_{(s)}$ increasing as the lines move further from the middle. As can be seen, the design goals of $f_h^{(lower)}(\omega) > f_c^{(lower)}(\omega)$ and $f_h^{(upper)}(\omega) < f_c^{(upper)}(\omega)$ are mostly satisfied in the $\lambda_{(s)} = 0$ or 1 case, but that in the $\lambda_{(s)} = 2$ case, it is satisfied for the most part when $n_{(s)} \geq 13$.

Thus, the guidelines for choosing $n_{(l)}$ do not change, except the added caution to use $n_{(s)} \geq 13$ when using local quadratic trend or seasonal smoothing.

2.3 The Revision Problem

Commonly, the most important parts of a time series are its endpoints. For example, the government is usually most interested in the latest seasonally-adjusted

unemployment rate. Thus a seasonal-trend decomposition method must perform well at the endpoints. When the STL method was first introduced, there were many who pointed out concerns about the reliability of the component estimation at the endpoints (Gray and Thomson, 1990; Findley and Monsell, 1990; Wallgren and Wallgren, 1990).

The difficulty of smoothing at the endpoints is inherent in all local regression procedures because the first and last observations of the series are not smoothed with the same symmetric weights that are used in the center of the series. One consequence is the increase in variance at the endpoints, as discussed in section 1.2.3. Figure 5.38 shows, for example, the variance coefficient (1.23) for several loess fits along an equally-spaced design space.

The estimates for endpoint observations are revised when new data is available, and the difference between the initial endpoint estimate and subsequent estimates with more data are known as *revisions*. Large revisions are undesirable for a number of reasons, and there are a few ways to deal with the problem:

1. Alter the smoothing procedure at the endpoints to reduce the mean-squared estimation error.
2. Extrapolate the time series and run the procedure on the extended series.

This section addresses the first approach. Section 2.3.1 discusses a modification to loess that has been proposed to deal with the endpoint problem, while section 2.3.2 introduces a new approach, blending, where the estimates at the endpoint are blended down to a lower-degree polynomial fit. Section 2.3.3 compares the performance of these.

In the second approach, autoregressive integrated moving average (ARIMA) models are used to extrapolate the original series, as done in the X11-ARIMA method. This has led to a great performance increase in terms of revisions with the X-11 system. While such ideas could be applied to STL, this work does not investigate this possibility.

2.3.1 Reproducing Kernel Hilbert Spaces with Loess

Some interesting recent work by [Bianconcini \(2007\)](#) looks at a reproducing kernel hilbert space (RKHS) representation of the loess smoother and pays particular attention to the asymmetric smoothing case. A RKHS is a Hilbert space of functions characterized by a kernel that reproduces every function of the space by inner products. RKHS kernels were found corresponding to local linear and local quadratic loess fitting with the tricube weight function.

For local linear loess, the second order tricube kernel is

$$K(t) = \begin{cases} \frac{70}{81}(1 - |t|^3)^3 & \text{for } 0 \leq t < 1 \\ 0 & \text{for } t \geq 1 \end{cases} \quad (2.50)$$

and for local quadratic loess, the third order tricube kernel is

$$K(t) = \begin{cases} \frac{70}{81}(1 - |t|^3)^3 \left(\frac{539}{293} - \frac{3719}{638}t^2 \right) & \text{for } 0 \leq t < 1 \\ 0 & \text{for } t \geq 1. \end{cases} \quad (2.51)$$

In the symmetric smoothing case, these kernels give the exact same smoothing coefficients as loess. For example, consider a series of 15 observations at times $t = 1, \dots, 15$. The symmetric local linear loess fit at $i = 8$ with span $q = 15$ is obtained from the weights $l(8)$ from (1.11). Using the RKHS kernel (2.50), the weights

$$w_j = \frac{K((j - 8)/7)}{\sum_{k=1}^{15} K((k - 8)/7)}, \quad j = 1, \dots, 15 \quad (2.52)$$

are exactly the same as the weights $l(8)$. The same is true in the symmetric local quadratic case.

The RKHS approach differs from loess with respect to asymmetric smoothing. The loess smooth will always involve q observations, while the neighborhood for the RKHS smooth gets smaller as the endpoint is reached, down to $(q + 1)/2$ observations at the endpoint. For example, the RKHS fit corresponding to a loess fit with $q = 15$ at $i = 1$ would involve 8 observations, with weights calculated as

$$w_j = \frac{K((j - 1)/7)}{\sum_{k=1}^8 K((k - 1)/7)}, \quad j = 1, \dots, 8 \quad (2.53)$$

Figure 5.39 shows the weights obtained by loess and RKHS loess for the symmetric and endpoint cases with $q = 15$ and with $\lambda = 1$ and $\lambda = 2$.

Some, such as Gray and Thomson (1990), may view the fact that the span of the filter does not increase upon encountering the boundaries of the data, as is the case with RKHS but not with loess, as favorable. There are some problems with the RKHS loess approach with regard to practical application. One is that it is difficult to incorporate user-defined or robustness weights into the fitting. In the traditional loess setting, these weights enter in during the weighted local regression. In the RKHS approach, one would be constrained to apply these weight adjustments to the final weights, which is not the same. Also, there would be difficulties dealing with missing values and handling cases where $q > n$. More will be done to investigate RKHS properties in the following sections.

2.3.2 Blending to Lower Degree Polynomials at the Endpoints

A new approach to deal with the endpoint problem is blending to a lower degree polynomial at the endpoints. Blending simply means averaging the endpoint fit with an endpoint fit of smaller degree. By observing figure 5.38, one might see why this would be a good idea in terms of reducing the variance at the endpoints. Degrees 1 and 2 smoothing have much higher variance coefficients at the endpoints than in the interior, while degree 0 has variance that stays nearly constant. Averaging an endpoint degree 1 smooth with an endpoint degree 0 smooth, for example, would reduce the endpoint variance. Reducing the variance isn't the only consideration, but is a good initial motivation for studying this approach.

Assuming that the analyst has already chosen loess smoothing parameters that perform well in the interior region of the data, call these (λ_o, q_o) (“o” for “original”), the task is to average the endpoint fits with another fit with parameters (λ_b, q_b) (“b” for “blend”). With the equally-spaced design space $x = (1, \dots, n)$, obtaining the

blended fit $\hat{\mu}^{blend}(n)$ at the right endpoint ($i = n$) is achieved simply by a weighted average of the original fit weighted by $(1 - \delta)$ and the blending fit weighted by δ

$$\hat{\mu}^{blend}(n) = (1 - \delta)\hat{\mu}(n; \lambda_o, q_o) + \delta\hat{\mu}(n; \lambda_b, q_b). \quad (2.54)$$

It is beneficial to view the blended fit in terms of the operator weights (1.11) so that properties such as the variance coefficients and frequency response of the blended fit can be studied. For the blended fit, let $l^{blend}(x)$ be the operator weights so that

$$\hat{\mu}^{blend}(x) = \sum_{i=1}^n l_i^{blend}(x) y_i \quad (2.55)$$

$$= \sum_{i=1}^n ((1 - \delta)l_i(x; \lambda_o, q_o) + \delta l_i(x; \lambda_b, q_b)) y_i. \quad (2.56)$$

What Span and Degree to Blend to?

Since there are many possibilities for how to blend, it is useful to develop guidelines based on different aspects of the behavior of the blending results. These behaviors are mostly based on calculations involving $l^{blend}(x)$. The main aspects considered are the effect of parameter choices on the bias and variance of the blended filter, as well as its spectral frequency-passing properties.

Loess smoothing in the STL context is always done at either $\lambda_o = 1$ or $\lambda_o = 2$ ($\lambda_o = 0$ is also an option but not recommended and it is not different from $\lambda_o = 1$ smoothing in the interior in the equally-spaced design setting). Thus, when smoothing with $\lambda_o = 2$, the options for λ_b are 0 or 1, and when smoothing with $\lambda_o = 1$, the only option is $\lambda_b = 0$. Assuming λ_b has been chosen, the question arises regarding the choice of the endpoint blending span, q_b . Definitely a choice of $q_b > q_o$ is not a good one because this means the fit being blended to involves more data than the original fit and at a lower degree polynomial and will definitely lead to a huge increase in bias.

When considering choice of q_b , it is informative to look at values of (λ_b, q_b) that yield an *endpoint* variance coefficient equivalent to that obtained by the (λ_o, q_o) in the *symmetric* case. A look at figure 5.38 shows that, for example, that the endpoint

variance coefficient for $\lambda = 0$ look to be the same as the interior variance coefficient for $\lambda = 1$ of the same span. For a more comprehensive study, figure 5.40 shows the span needed for the variance of degree 0 endpoint smoothing to match the variance of a given span for degree 1 symmetric smoothing, considering odd spans from 11 to 1001. It is clear that a $\lambda = 1$ smoothing has symmetric variance equal to the endpoint variance of a $\lambda = 0$ smooth with the same span. Thus, if one chose to blend 100% from degree 1 to degree 0 smoothing with $q_b = q_o$, the endpoint variance would be the same at the endpoints as it is in the interior.

Figure 5.41 compares symmetric $\lambda = 2$ smoothing to endpoint $\lambda = 0$ and $\lambda = 1$ smoothing. Here, achieving equivalent endpoint variance when smoothing the endpoints with $\lambda = 1$, one would need to double the span of the endpoint smoothing relative to the span of the symmetric smoothing, which is not desirable. On the other hand, smoothing the endpoints with $\lambda = 0$, one would need to use half the span of the symmetric smoothing to obtain equivalent variance.

These results favor the use of $\lambda_b = 0$ regardless of the choice of λ_o . Of course, one would not want to blend 100% to a lower degree, as this would introduce too much bias. It is too much to ask for the endpoint variance to remain the same as what was achieved in the interior, but studying the results above that do achieve this goal gives good insight into choice of q_b . The suggestion is to choose $q_b = q_o$ when $\lambda_o = 1$ and to choose q_b to be the next odd integer of $(q_o - 1)/2$ when $\lambda_o = 2$.

Further support for the “always blend to 0” rule is given by looking at the power transfer function, $|L_{yx}(f)|^2$, (1.49) of loess endpoint fits for several parameter combinations. Figure 5.42 shows $|L_{yx}(f)|^2$ for fits with $\lambda = 0, 1, 2$ and q ranging from 7 to 35. Only the $\lambda = 0$ endpoint fits have power transfer functions that do not amplify low-frequency signals. The $\lambda = 2$ functions have the highest magnitude, and blending to $\lambda = 1$ from $\lambda = 2$ will not fix this. Blending to $\lambda = 0$, however, can help the power transfer function of the blend behave more appropriately.

To summarize, the results above suggest nice default choices for the blending smoothing parameters. These choices are:

- If $\lambda_o = 1$, blend to $\lambda_b = 0$ with $q_b = q_o$.
- If $\lambda_o = 2$, blend to $\lambda_b = 0$ with q_b as the next odd integer of $(q_o - 1)/2$.

Graduating the Blend from Endpoint to Symmetric

So far, blending only at the endpoints $i = 1$ and $i = n$ has been considered. The endpoint problem exists, although not nearly as serious, at points near the boundaries as well. If the span is q , let $m = (q - 1)/2$. Then the points effected at the beginning of the series are $1, \dots, m$. Points beyond m are fitted with symmetric weights. The same applies at the end of the series.

A simple rule to deal with all of the endpoints is to diminish the amount of blending in a linear fashion until the interior symmetric region of the design space is reached, as this is where the variance is stabilized. For example, at the beginning of the series, the first point would be blended with weight δ to the lower degree smooth, and each subsequent point would be blended down with weight

$$\delta_i = \frac{m + 1 - i}{m} \delta, \quad i = 1, \dots, m. \quad (2.57)$$

Blending as a Shrinkage Estimator

It is worthwhile to note that the blended estimate can be viewed as a shrinkage estimator. It is useful to think of blending in the shrinkage context in that it gives hope of reducing, at the sacrifice of some bias, the mean squared error of the endpoint estimates.

Let X_λ be the design matrix for the local regression of degree λ at the left endpoint, $i = 1$. Then, for example,

$$X_2 = \begin{pmatrix} 1 & 0 & 0^2 \\ 1 & 1 & 1^2 \\ \vdots & \vdots & \vdots \\ 1 & q-2 & (q-2)^2 \\ 1 & q-1 & (q-1)^2 \end{pmatrix} \quad (2.58)$$

Define weight matrix W as usual with diagonal $W_{ii} = T((i-1)/(q-1))$ for $i = 1, \dots, q$ where T is the tricube function (1.2).

Consider the case of blending from $\lambda = 1$ to $\lambda = 0$ at the endpoint. Recall that the guidelines call for q to remain the same for both estimates. Using the design and weight matrices, from (2.55) and (1.11),

$$l(1) = (1 - \delta)[e'_1(X'_1 W X_1)^{-1} X'_1 W] + \delta[(X'_0 W X_0)^{-1} X'_0 W] \quad (2.59)$$

where $e'_1 = (1, 0)$. This yields the fit

$$\hat{\mu}(1) = l'(1)y. \quad (2.60)$$

This blending from $\lambda = 1$ to $\lambda = 0$ can be viewed as a local ridge regression with ridge penalty c on the slope coefficient, where the fit at $i = 1$ is obtained from solving

$$\hat{\beta} = \underset{\beta}{\operatorname{argmax}} (y - X_1 W \beta)'(y - X_1 W \beta) + c\beta_1^2. \quad (2.61)$$

where $\beta' = (\beta_0, \beta_1)$. For correctly chosen c , the ridge estimate $\hat{\beta}_0$ is equivalent to the blended fit $l(1)$.

To show the relationship between the ridge estimate (2.61) and the blending estimate (2.59), the solution to (2.61) for $\hat{\beta}_0$ is

$$\hat{\beta}_0 = (e'_1(X'_1 W X_1 + cI_1)^{-1} X'_1 W)'y \quad (2.62)$$

where

$$I_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}. \quad (2.63)$$

Thus, for $\hat{\beta}_0 = \hat{\mu}(1)$ to be true, let

$$A = (X_1' W X_1)^{-1} \quad (2.64)$$

then from (2.59), (2.60), and (2.62)

$$e_1'(A^{-1} + cI_1)^{-1} X_1' W = (1 - \delta) e_1' A X_1' W + \delta (X_0' W X_0)^{-1} X_0' W \quad (2.65)$$

must hold. Now, using the Sherman-Morrison formula,

$$(A^{-1} + cI_1)^{-1} = A - A e_2 (1/c + e_2' A e_2)^{-1} e_2' A \quad (2.66)$$

$$= A - \eta A I_1 A \quad (2.67)$$

where $e_2' = (0, 1)$ and $\eta = 1/(1/c + e_2' A e_2)$, a constant. So

$$e_1'(A^{-1} + cI_1)^{-1} X_1' W = e_1'(A - \eta A I_1 A) X_1' W \quad (2.68)$$

$$= (1 - \delta) e_1' A X_1' W + \delta e_1' (I - \delta^{-1} \eta A I_1) A X_1' W. \quad (2.69)$$

Thus for (2.65) to hold,

$$e_1' (I - \delta^{-1} \eta A I_1) A X_1' W = (X_0' W X_0)^{-1} X_0' W \quad (2.70)$$

which is true when

$$c = (e_2' A e_2 / \delta - e_2' A e_2)^{-1}. \quad (2.71)$$

Thus, blending from $\lambda = 1$ to $\lambda = 0$ with blending proportion δ is equivalent to a local linear ridge regression with penalty c on the slope parameter.

In the case of blending from $\lambda = 2$ to $\lambda = 0$, a sort of quasi-ridge regression analog is available, with the least-squares problem now being penalized as

$$\hat{\beta} = \underset{\beta}{\operatorname{argmax}} (y - X_2 W \beta)' (y - X_2 W \beta) + c_1 \beta_1^2 + c_2 \beta_2^2 + c_{12} \beta_1 \beta_2. \quad (2.72)$$

A Data-Based Approach to Selecting the Amount of Blending

The discussion thus far has been about the choice of λ_b and q_b . The final parameter to specify is δ , the proportion of blending. Choosing $\delta = 0$ is equivalent to no blending and choosing $\delta = 1$ is equivalent to local constant smoothing at the boundary. Some metrics can be used to assist with choosing δ .

Mean Squared Revision Error

The parameter δ can be chosen to minimize the mean squared revision error (MSRE), which is the average squared difference between the asymmetric smooth and the symmetric smooth at any given data point. If the span of the smoother q is much smaller than the number of observations in the series n , then the revision at each point i in the interior of the series can be calculated by comparing the asymmetric fit at i and the symmetric fit at i , provided there is enough data at either side of i to do so.

For example, if a series has $n = 100$ observations and $q = 11$, then the first revision error can be calculated at $i = 11$. The asymmetric fit is calculated only using data at points $i \leq 11$. Then the symmetric fit is calculated using the 5 observations on each side of $i = 11$ and the value at $i = 11$. The difference between these two fitted values is the revision error at $i = 11$. This can be calculated all the way across the design space until point $i = 95$, but not any higher because the symmetric smoother always needs 5 observations on both sides. The mean of the squared differences of the symmetric and asymmetric fitted values at each point is the mean squared revision error.

To account for blending in the MSRE, let a_i denote the asymmetric loess estimate at point i using λ_b and q_b and let b_i denote the asymmetric loess estimate at point i using λ_o and q_o so that the blended fit is

$$\hat{\mu}^{blend}(i) = \delta a_i + (1 - \delta)b_i. \quad (2.73)$$

Finally, let c_i denote the symmetric loess fit at i using λ_o and q_o . Then the MSRE is

$$\text{MSRE} = \frac{1}{m} \sum_{i=q_o}^{n-(q_o-1)/2} (\delta a_i + (1 - \delta)b_i - c_i)^2 \quad (2.74)$$

where $m = n - (q_o - 1)/2 - q_o + 1$ is the number of values summed.

Minimizing the MSRE

A completely data-based criterion for choosing δ is obtained by choosing δ that minimizes the MSRE. Minimizing (2.74),

$$\hat{\delta} = \frac{-\sum(a_i - b_i)(b_i - c_i)}{\sum(a_i - b_i)^2}. \quad (2.75)$$

Empirical Study

An empirical study can help get a better understanding of the properties of δ . Consider two simulated data sets. One is simply a sinusoidal signal with i.i.d. normal noise added, of the form

$$\mu(j) = \sin(2\pi j/100) + \epsilon_j, \quad j = 1, \dots, 500. \quad (2.76)$$

The second example is simulated to follow the ED visit pattern from section 2.1.8 (see figure 5.32). The trend extracted from the null model of no seasonality is used as a baseline and i.i.d. normal noise is added to the signal to obtain simulated series. Varying magnitudes of noise are added to each baseline to investigate the behavior of $\hat{\delta}$ with the standard deviation of the normal distribution σ ranging from 0.01 to 1 in increments of 0.1. Diagnostics found that the sinusoid data is on average best estimated using local quadratic smoothing with $q = 75$ and the simulated ED data using local quadratic with $q = 91$.

Figures 5.43 and 5.44 show a simulated series from each example with the standard deviation of the added noise at its lowest and highest values, 0.01 and 1. Lines are plotted for the symmetric fit, the endpoint asymmetric fit with $(\lambda_o = 2, q_o = 75)$ for the sinusoid data and $(\lambda_o = 2, q_o = 91)$ for the ED data, and the endpoint asymmetric fit with $(\lambda_b = 0, q_b = 37)$ for the sinusoid data and $(\lambda_b = 0, q_b = 45)$ for the ED data. Note the clear phase shift in the endpoint local constant fitting.

Each series was simulated 10 times per error standard deviation, and the MSRE-optimal δ was determined for endpoint revisions and for revisions at 1, 2, and 3 points from the endpoint. Figures 5.45 and 5.46 show the resulting δ values for both

examples. The main focus is on the endpoint smoothing (0 points from end). The optimal amount of blending increases as the series gets more noisy. Also, the optimal amount of blending decreases as the fit becomes more symmetric. To get a better view of this, figures 5.47 and 5.48 show the δ values vs. number of points from the end. The decline looks quite linear in each case, which agrees with the criteria set forth in (2.57) to linearly graduate the blend down as the interior of the space is reached. However, these results indicate that it may be better to graduate down to $\delta = 0$ much more quickly. As the fitting moves away from the endpoint, however, the term “optimal” becomes less meaningful because the MSREs to choose from are all already much smaller than at the endpoint.

Figures 5.49 and 5.50 compare the MSRE under the optimal blending to the MSRE using the unmodified endpoint loess fit. For the sinusoidal data, the endpoint MSRE for blending is nearly half that for regular loess. For the ED data, the benefit of blending over no blending increases as the noise level in the data increases, nearly halving the endpoint MSRE at $\sigma = 1$.

2.3.3 Comparison of Blending and RKHS

The blending method will be compared to the RKHS method both in terms of spectral properties as well as in practical application. First, consider the spectral properties of the endpoint smoothing methods. Properties are presented for both local linear and quadratic endpoint fits for the unmodified loess endpoint smooth ($\delta = 0$), the halfway blended endpoint smooth ($\delta = 0.5$), the fully blended endpoint smooth ($\delta = 1$), and the RKHS endpoint smooth.

It is worthwhile to assess the power transfer function properties of each method of endpoint smoothing. Figure 5.51 shows the power transfer function for each method for smoothing with $q = 17$. The dotted line shows the transfer function for the symmetric fit, and each gray line represents the transfer function as the fit becomes more asymmetric, until it reaches the endpoint, which is represented by the solid line.

The $\delta = 0$ blending exhibits the problems discussed in section 2.3.2, namely that the endpoint smoothing excessively amplifies power at the lower frequencies, and passes much more noise than the symmetric smoother. With $\delta = 0.5$, this problem is eliminated, and parameter choice rules for the fit blended to seem to be satisfactory with respect to power transfer. For $\lambda = 1$, it is interesting how similar the symmetric and asymmetric transfer functions are for $\delta = 1$ blending. The asymmetric RKHS transfer functions seem to trail off more slowly than the blending transfer functions.

The phase shift plots corresponding to the frequency response functions for the above setting can be seen in figure 5.52. Compared to RKHS, the blending methods have slightly smaller phase shift effect.

Figure 5.53 shows the endpoint smoothing weights for each method. Figure 5.54 shows the variance coefficients at the boundary of the design space for each method. The RKHS and $\delta = 0.5$ blend variance look very similar.

To see how the methods compare when applied to actual data, figures 5.55 and 5.56 show the MSRE for the sinusoid and ED simulated data, respectively. For the sinusoidal data, the blended endpoint smoothing has much smaller MSRE, and the traditional loess endpoint smoothing even has smaller MSRE than RKHS in some cases. For the ED data, the blending and RKHS methods appear to perform similarly, both doing better than the traditional approach. A better comparison of blending and RKHS can be made by looking at the ratio of the two. Figures 5.57 and 5.58 show the RKHS MSRE divided by the blending MSRE. For the sinusoid data, the RKHS MSRE is always at least 1.5 times the blending MSRE. For the ED data, the endpoint MSRE for both methods is quite similar, with the blending MSRE being slightly larger in most cases.

Finally, some results are presented on applying these methods to STL modeling. The power transfer functions for an STL operator matrix with $n_{(p)} = 7$, $n_{(s)} = 15$, and $\lambda_{(s)} = 1$ are shown using the different revision techniques in figure 5.59. The revision techniques are used in all loess smoothing stages of the STL procedure. The goal is for the transfer function at the endpoint to look as much as possible like the

symmetric transfer function. The traditional no-blending transfer function has some unfavorable properties, such as amplifying noise and trend smoothing interfering with seasonal smoothing. The $\delta = 0.5$ blending and RHKS transfer functions look much better. The trend is still slightly bleeding into the seasonal. Recall that the guidelines for automatic STL parameter selection in section 2.2 are based on symmetric transfer functions. Blending helps the same guidelines hold at the endpoints.

Now, to see the results of this section in action with a real data set modeled by STL, consider the CO₂ data. This is an example of a data set where the blending parameter choice is difficult to choose. The variability is very small, which means a small δ will be optimal. For the trend smoothing, the trend is increasing and any amount of blending to local constant smoothing will lead to considerable bias and therefore be penalized when minimizing the MSRE. Using (2.75), the optimal blending parameter was chosen to be $\hat{\delta} = 0.04$. There is a tradeoff here between having nice spectral characteristics, which requires larger δ , or having lower bias, which requires smaller δ . Entertaining the idea of increasing δ , figure 5.60 shows the transfer functions and figure 5.61 shows the variance coefficients for using $\delta = 0.5$ for the seasonal and trend loess smooths. Such a large δ is not optimal in terms of MSRE, in fact, $\delta = 0.5$ gives $\text{MSRE} \approx 0.1$ for the trend smoothing, while the MSRE for no blending is 0.022. Note though that the RKHS MSRE for the trend smoothing is also about 0.1, and since the RKHS and $\delta = 0.5$ blending have variance that behaves very similar, they both have about the same bias. Compare figures 5.60 and 5.61 to figures 5.28 and 5.27 to see the improvements in the spectral and variance properties when blending with $\delta = 0.5$.

2.4 Software Implementation: stl2 and operator Packages

Almost as important as new developments in methodology is their implementation and accessibility for practical application. This section outlines some attempts at achieving this for the STL methodology.

The original STL implementation was done in FORTRAN. Much attention was given to the speed of the procedure. The current STL implementation in R, available by `stl()`, is just a wrapper around the FORTRAN code. This implementation does not allow for missing values and is missing several of the developments described in this work. S-plus also has an implementation of STL that does not depend on the FORTRAN code and is a bit more flexible. Two new R packages were created to make the results of this chapter available, “stl2” and “operator”. Some brief discussion is given below, but the full reference manuals for these packages are available in the appendix.

2.4.1 The stl2 Package

Several improvements have been made to the STL implementation, packaged as “stl2”. Great attention has been paid to speed. The computationally intense parts of the procedure are implemented in C. All the non-speed critical components have been moved to R. This makes extendability much easier without much sacrifice for speed, making modifications such as trading day, multiplicative models, etc. more simple to add. This keeps in the spirit of one of the original intents of STL – an open system available to create a fertile environment for new ideas and improvements.

Beyond the architecture of the code, many small enhancements have been made to the procedure. The main function of the package is `stl2()`, which has the same arguments as the original `stl()` function. In addition, it offers the following

- local quadratic support for seasonal, trend, and low-pass loess smoothing by specifying `s.window=2`, etc.
- automatic choice of parameters based on new guidelines in section 2.2, and allowing ω to be specified, e.g. `critfreq=0.1`.
- blending for seasonal, trend, and low-pass loess smoothing by specifying δ , e.g. `s.blend=0.5`, etc.

- further trend smoothing by specifying vectors `fc.window` and `fc.degree` for the spans and degrees of the post-trend components to be estimated. These are carried out in the order they are specified.

The `stl2` function can handle missing values. For example, figure 5.62 shows the CO₂ decomposition with two years of data missing.

A fitted model from `stl2()` is of class "`stl2`", and several methods have been created to simplify working with these objects. Methods `seasonal()`, `trend()`, and `remainder()` were created for both the original "`stl`" class and the "`stl2`" class, and extract the corresponding components. The method `fc()` extracts the post-trend smoothing components, if any. Also, a method `fitted()` returns the seasonal plus trend, and `time()` returns the time vector, if specified.

Also, several plot functions have been created to simplify plotting the results, using the trellis/lattice framework (cite...). These are plots for the decomposition and diagnostics. All decomposition plots in this work were created using these functions. They all take an object of class "`stl2`" and return an object of class "`trellis`", and allow the user control over all aspects of the trellis plot. The documentation can be viewed for examples of these functions.

2.4.2 The operator Package

In conjunction with the `stl2` package, the `operator` package was created to handle computation of the STL operator matrices and perform inference functions. This package is much more computationally demanding than the `stl2` package and should only be used when further study of statistical properties of an STL model is desired. Details and examples can be seen in the appendix.

This package offers the same improvements as `stl2`, such as blending, local quadratic support, and post-trend smoothing. Also, it is possible for the user to specify an ARMA model to model the remainder. The two main functions in the package are `stlop()` and `loessOp()`, which return objects of class "`stlop`" and "`op`"

containing among other things operator matrices for loess and STL models according to the parameters specified.

The inference-related functions in this package include a `predict()` method, an `anova()` method, and `cp()` methods for predicting ahead, calculating prediction and confidence intervals, testing hypotheses, and calculating C_p statistics. All inference is carried out using the final fit matrix L^* , which incorporates the seasonal, trend (or multiple trend if post-smoothing was specified), and ARMA fits, if specified.

In addition to inference functions, the `operator` package provides functions for frequency-domain plots such as plots of the power transfer function. The `frequ()` function obtains the frequency response function for a specified STL model for the symmetric smoothing case. Another function `frequlo()` calculates the frequency response function for a general linear filter matrix, and accepts objects of class `"op"`. This function can calculate the frequency response for asymmetric fits and can be used with the overall STL model matrix L^* . Results from these functions are of class `"frequ"`, and there are plot methods for viewing the power transfer function and phase shift function.

Methods have been available to make going from one package to another easy. For example, there is a `stlop()` method for objects of class `"stl"` and `"stl2"`, as well as a `stl2()` method for objects of class `"stl"`.

2.4.3 Computational Considerations

With respect to practical application, it is worthwhile to examine the computational properties of these methods. Most focus is on the `operator` package. All results below were obtained using an Intel Core2Duo 2ghz machine with 2 gigabytes of RAM.

Computational Complexity

The `stl2()` function in the `stl2` package manages to stay within the same order of time complexity as the original `stl()` implementation. There is an overhead due to much of the code residing in R, but it is not much of a cost. Figure 5.63 shows the results of running time for `stl()` vs. `stl2()` for sample sizes ranging from 2,000 to 15,000, with 10 samples at each sample size. The relationship between the two functions is linear, and the slope coefficient of the fitted line interestingly is almost exactly 2. So on average, the `stl2()` function is twice as slow as `stl`. This is not a big deal especially considering that even at samples of size 15,000 it only takes under a quarter of a second to run. The constant factor of increase in time complexity is also acceptable because of the ease of maintaining and enhancing the R-level code.

The added functionality of calculating the STL operator matrix comes with a cost, namely many unavoidable $n \times n$ matrix multiplications, which results in less-than-desirable computational complexity on the order of $O(n^3)$. Although many of the matrices at some point may be rather sparse, after a few multiplications, the sparseness is lost, so no effort has been made to speed calculation using sparse matrices.

Computation would be simple if all of the filters were symmetric (the data had no endpoints) and if there was no need to accommodate other things, such as user-specified weights, or ARMA modeling of the remainder. These requirements are too restricting though.

All matrix multiplication is carried out using basic linear algebra subroutines (BLAS) in R. To give an idea of the running time, with R configured to use a parallel BLAS, figure 5.64 shows the results of running `stl0p()` for 3 replicates of various n and a fixed set of parameters, with the auxiliary statistics also being computed. This is just to get a general feel for running time, as it will vary greatly based on the machine and the specified parameters. The fitted line is a cubic polynomial, and the fit is extrapolated to $n = 3000$. Even for $n = 3000$, the computation time is not

extremely inconvenient, and may be faster with a machine with more cores. However, no matter how long one is willing to wait for computation of a large series, storage perhaps becomes a bigger issue with large n .

Storage

The expectation for storage space for STL operator objects is to be of order n^2 . The storage requirements for various n for `stl0p()` are shown in figure 5.65. A quadratic polynomial fits the points. Extrapolating out to $n = 3000$, approximately 1/4 of a gigabyte is required. And this is just for storage, not counting the workspace.

Thus storage will be an issue at some point. The good news though is that for manageable data sets (say, $n < 2000$), computation time isn't bad and storage should be feasible.

2.4.4 Approximations

If inference such as predicting ahead is desired for longer series and the user determines that the decomposition model is only heavily dependent on the recent history of the data (say the latest 1000 observations, for example), the `operator` package could still be used by only fitting the recent series. This idea is quite reasonable if the loess smoothing spans are not very large.

Also, approximations can be made to the overall STL fit matrix when the parameters are chosen such that the seasonal and trend smoothing are nearly spectrally orthogonal. If this is the case, the STL fit operator matrix can be approximated by only one iteration using (2.3), (2.5), and (2.7) to obtain

$$\tilde{L}^* = HC + T(I - HC) \quad (2.77)$$

as opposed to iterating through seasonal and trend smooths until convergence. This approximation can be fast because of the sparseness of the matrices.

For example, with the CO₂ model with blending, the high-pass filter matrix H and the cycle-subseries smoothing matrix C only have about 7.2% non-zero entries, although their product is 87.5% non-zero. The trend smoothing matrix T is banded with only about 3.6% non-zero entries. Thus sparse matrix routines could be used to approximate L^* and perhaps drop an order of magnitude in complexity.

To compare some quantities with the approximate and true operator matrices for the CO₂ model, see figure 5.66, which shows the approximate and true variance coefficient across the design space for the true and estimated seasonal, trend, and overall fit matrices. The variance has the largest discrepancy at the endpoints. Thus predictions can be made with this model using approximations, but the standard errors will be somewhat unreliable. Also, compare some auxiliary statistics for this example such as $\text{tr}((L^*)'(L^*)) = 58.325$ vs. $\text{tr}((\tilde{L}^*)'(\tilde{L}^*)) = 58.4$.

2.5 General Guidelines

All of the modifications to the STL method presented here may have given the impression that the modeling method is considerably complicated for practical use. To organize the points of the original methodology and enhancements added, below are some guidelines to modeling time series with STL.

1. Fit an initial simple model, specifying $n_{(p)}$, $\lambda_{(s)}$, and $\lambda_{(t)}$ using `stl2()`.
2. Evaluate the fit using visual diagnostic methods (section 1.4.4).
3. Alter the model according to the findings in step 2. Mainly this will include whether or not to switch to local quadratic smoothing for either the seasonal or trend and how to adjust the seasonal smoothing parameter, $n_{(s)}$. If a custom $n_{(t)}$ is desired, check the resulting power transfer plots to ensure that the trend smoothing is not interfering with the seasonal smoothing. There may be several iterations between this step and step 2, and additional confirmation of parameter choices can also be done in step 5.

4. If further trend smoothing is desired, fit these components and carry out visual diagnostics.
5. If endpoint effects are important, re-fit the model using blending parameters for the seasonal and trend smoothing. Specify these either by an arbitrary blending proportion or using a data-based approach.
6. Evaluate the autocorrelation properties of the remainder component. If any statistical or inferential work (such as C_p parameter selection, confidence intervals, predicting ahead) is desired, try to model the remainder as an ARMA process.
7. If desired and if assumptions are met, carry out inference using the final model and using the `operator` package.

3. AN APPLICATION: SYNDROMIC SURVEILLANCE

3.1 Background

The development of statistical methods for accurately modeling disease count time series for the early detection of bioterrorism and pandemic events is a very active area of research in syndromic surveillance. Some relevant work in this area includes [Burkom \(2003\)](#); [Burkom et al. \(2007\)](#); [Reis and Mandl \(2003\)](#); [Hutwagner et al. \(2003\)](#); [Wallenstein and Naus \(2004\)](#); [Brillman et al. \(2005\)](#); [Kleinman et al. \(2005\)](#); [Wong et al. \(2002\)](#); [Burr et al. \(2006\)](#); [Stroup et al. \(1993\)](#); [Hutwagner et al. \(1997\)](#); [Farrington et al. \(1996\)](#); [Stern and Lightfoot \(1999\)](#); [Simonsen \(1997\)](#).

Early detection requires accurate modeling of the data. A challenge to modeling disease count time series is systematic components of time variation whose patterns have a level of predictability — inter-annual (long-term trend), day-of-the-week, and yearly-seasonal components [Burr et al. \(2006\)](#); [Burkom et al. \(2007\)](#). The variation can be ascribed to causal factors: for the inter-annual component the factor can be an increase or decrease in the population of people from which patients come; for the yearly-seasonal component it is changing weather and human activities over the course of a year; and for the day-of-the-week component it is a changing propensity of patients to seek medical attention according to the day of the week. In addition to these components is a noise component: random errors not assignable to causal factors that often can be modeled as independent random variables.

The most effective approach to early outbreak detection is to account for the systematic components of variation and the noise component through a model, and then base a detection method on values of the systematic components and the statistical properties of the random errors. Methods that do not accurately model the components can suffer in performance. The model predicts the systematic recurring behavior

of the systematic components so it can be discounted by the detection method, and specifies the statistical properties of the counts through the stochastic modeling of the noise component. The two dangers are lack of fit — patterns in the systematic components are missed — and overfitting — the fitted values are unnecessarily noisy. Both are harmful. Lack of fit results in increased prediction squared bias, and overfitting results in increased prediction variance. The prediction mean-squared error is the sum of squared bias and variance.

Many modeling methods require large amounts of historical data. Examples are the extended baseline methods [Stroup et al. \(1993\)](#); [Hutwagner et al. \(1997\)](#); [Farrington et al. \(1996\)](#); [Stern and Lightfoot \(1999\)](#); [Simonsen \(1997\)](#) of the Early Aberration Reporting System (EARS) [Hutwagner et al. \(2003\)](#), which require at least 3 years of data. Another popular example is a seasonal autoregressive integrated moving average (ARIMA) model in [Reis and Mandl \(2003\)](#) that is fitted to 8 years of data. Such methods are not useful for many surveillance systems that have recently come online, so methods have been proposed that require little historical data [Hutwagner et al. \(2005\)](#). They typically employ moving averages of recent data, such as C1-MILD (C1), C2-MEDIUM (C2), and C3-ULTRA (C3), which are used in EARS. But such methods do not provide optimal performance because they do not exploit the full information in the data; they smooth out some component effects and operate locally to avoid others, rather than accounting for the components.

3.2 Modeling the Data

The goal of this work is accurate modeling of components of variation, but without requiring a large amount of historical data. The research was carried as part of an analysis of the daily counts of chief complaints from emergency departments (EDs) of the Indiana Public Health Emergency Surveillance System (PHESS) [Grannis et al. \(2006\)](#). The complaints are divided into eight classifications using a Bayesian classifier, CoCo [Olszewski \(2003\)](#). Data for the first EDs go back to November 2004,

and new EDs have come online continually since then. There are now 76 EDs in the system.

There are many ways to proceed in modeling chief-complaint time series. One is to develop parametric models. Modeling methods here are based on STL (see chapters 1 and 2) because of the ability for nonparametric methods to account for a much wider range of component patterns than any single parametric model. Just as importantly, it is found that it can effectively be used with as little as 90 days of data.

The STL decomposition was run on all Indiana EDs for respiratory-related counts. Chief complaints categorized as respiratory are fundamental markers for a number of naturally occurring diseases, and research has shown that diseases from bioweapons have early characterization of influenza-like illness [Franz et al. \(1997\)](#), which typically results in respiratory complaints.

Modeling was carried out on respiratory time series for the 30 EDs that had a mean daily count greater than 10. The STL method only performs satisfactorily when the ED has a large enough mean daily count. These series end in April 2008, and start at times ranging from November 2004 to September 2005. To maintain anonymity of the EDs, names have been replaced by three letter words in the figures

3.2.1 Model Overview

STL modeling was used to decompose the square root of ED daily counts into four components:

$$\sqrt{y_i} = t_i + s_i + d_i + r_i, \quad (3.1)$$

where i is time in units of years and increments daily, y_i is the respiratory count on day i , t_i is an inter-annual component that models long-term trend, s_i is a yearly-seasonal component, d_i is a day-of-the-week component, and r_i is a random-error noise component. The reason for modeling on the square root scale is discussed in section 3.2.5.

Figure 5.67 shows a decomposition of $\sqrt{y_i}$ for one ED. The long-term trend component t_i is nearly constant. The yearly-seasonal component s_i has peaks due to seasonal influenza: single peaks in early 2005, 2007, and 2008, and a double peak in late 2006 and early 2007. While some of these yearly-seasonal effects are visible in the raw data, the effects are much more effectively seen in s_i because other effects including the noise are not present. Variation in the day-of-the-week component, d_i is small compared with the total variation of $\sqrt{y_i}$, but it cannot be ignored in the modeling because its variation is not small compared with the initial growth of critical disease outbreaks. The noise component r_i is irregular in behavior, and was found to be accurately modeled as independent, identically distributed normal random variables with mean 0.

3.2.2 Model Components and Parameter Selection

The STL decomposition (3.1) is obtained by first extracting the day-of-the-week component and then removing it to perform post-trend smoothing to obtain the inter-annual and yearly-seasonal components.

The day-of-the-week component, d_i

The model fitting begins with computation of a day-of-the-week component, d_i by direct application of STL. Diagnostics found that d_i is a strictly periodic component, so that the seasonal smoothing step of the STL procedure simply consists of averaging the cycle-subseries. The parameters for the trend component to extract d_i were $\lambda_{(t)} = 2$ and $n_{(t)} = 39$. Local quadratic smoothing was chosen because of the fluctuating nature of the trend.

The fitted day-of-the-week components, d_i are shown for each ED in figure 5.68. The components tend to peak on Sunday or Monday, fall to a minimum Thursday or Friday, and then rise. For some EDs, d_i is very high over the weekend but for others is considerably lower.

The inter-annual component, t_i

After obtaining d_i , the inter-annual component t_i is obtained by a very low-frequency post-trend smoothing of the deseasonalized data $\sqrt{y_i} - d_i$. The parameters chosen for this component are local linear fitting with a bandwidth of 1001 days.

Figure 5.69 shows the inter-annual component for the 30 EDs. Some t_i show an increase and others are nearly constant. The increase may be attributable to a growing population using an ED, and no growth to a stable population.

The yearly-seasonal component, s_i

For the yearly-seasonal component, s_i , a critical idea of the method is not pooling values across years with the same time of year — values of January 1, values of January 2, etc. — as is often done. If enough data was available, STL could be used to pool across years. Because one goal is for methods to work with limited data, s_i is obtained by loess using data in a neighborhood of i , and not across years. It is possible that this would be better in many surveillance applications even with substantially more data if the phase and shape are sufficiently variable from one year to the next. For example, for some Indiana EDs, seasonal influenza peaks are unimodal in some years and bimodal in others. The modeling presented here tracks this accurately, but averaging across years could easily distort the patterns, for example, making the bimodal peaks merge into unimodal ones. Other research has also found that the “one season fits all” assumption, leading to pooling across years, is not suitable for disease surveillance data [Burr et al. \(2006\)](#).

The yearly-seasonal is a low-frequency post-trend smoothing component that is obtained by smoothing the values $\sqrt{y_i} - d_i - t_i$. Due to the many peaks and valleys present in the raw series, local quadratic smoothing is used.

The bandwidth choice for the yearly-seasonal component is critical as the peaks and valleys throughout the course of a year need to be accurately accounted for. In addition to visual diagnostics, a C_p plot was made for each ED for various bandwidths

q with $\lambda = 2$. These plots are shown in figure 5.70. The vertical line at $\nu = 54.5$ corresponds to $q = 79$ and the vertical line at $\nu = 89.4$ corresponds to $q = 49$. The bandwidth $q = 79$ is a good choice for almost all EDs, but there are a few, such as “Act” and “Apt” where this is not the best choice from the C_p perspective. Ideally, the bandwidth should be as large as possible while still yielding a good fit to prevent overfitting. Moving the bandwidth down to $q = 49$ makes the fit suitable for all EDs, but at the sacrifice of perhaps being unnecessarily small for several EDs.

Figure 5.71 shows the yearly seasonal component using $q = 79$. The flu season peaks for 2004-2005 and 2007-2008 tend to be larger than those for 2005-2006 and 2006-2007. Many EDs show a double peak for 2006-2007.

3.2.3 Model Diagnostics

The parameters for the decomposition were chosen by use of visual diagnostics (section 1.2.4 and 1.4.4). Figure 5.72, shows normal quantile plots of the r_i , and indicates that the normal is a good approximation of the noise component distribution. Plots of the autocorrelation functions of the r_i are shown in figure 5.73. Although there are a few slightly significant autocorrelation estimates, surprisingly the overall behavior is indicative of independence. Thus, the noise component can be modeled as independent random variables. Loess smoothing of the r_i against i verified that no variation that should have been in t_i or s_i leaked into the noise component r_i . Separate loess plots of the r_i for each day-of-the-week verified that all significant day-of-the-week effects were captured by d_i . For example, figure 5.74 shows r_i by day-of-the-week for one ED. The superposed lines are loess fits. The lines are nearly flat for each day, supporting the strictly periodic nature of d_i .

3.2.4 Blending

Blending (section (2.3.2)) was applied to the yearly-seasonal smoothing and to the trend smoothing component of the model used to extract d_i to enhance the endpoint

MSRE and to result in a more constant variance. A good optimal blending proportion for each ED was found to be $\delta = 0.15$. Figure 5.75 compares the endpoint residuals for smoothing with $\delta = 0, 0.15$, and 0.5 . The endpoint residuals are calculated as difference between the observed value of the series $\sqrt{y_i}$ and the fitted value from fitting up to point i . The loess fit in the figure highlights deviations from zero, an indication of endpoint bias. Blending with $\delta = 0.15$ gives slightly less-biased endpoint residuals, while blending halfway results in very unfavorable results.

3.2.5 Modeling on the Counts Scale

The square root transformation results in much simpler statistical modeling properties. For example, the standard deviations of the r_i are nearly constant within and between hospitals, and the marginal distribution is nearly normal with mean 0 and variance σ_n^2 . Neither are true for the untransformed counts, as illustrated in figure 5.76, which shows the residual standard deviation after fitting the model vs. the mean daily count for each of the 30 EDs with and without the square root transformation. The behavior of the noise component r_i after transformation is not surprising because the raw counts can be thought of as Poisson random variables, and the square root of a Poisson random variable whose mean is not too small is approximately normal with a standard deviation of 0.5 [Johnson et al. \(1992\)](#).

To illustrate the behavior of the standard deviation and normality of the square root of a Poisson random variable, this distribution is explored through some simulations. Figure 5.77 shows the results from calculating the standard deviation of 100 samples of size 10,000 from a square root Poisson distribution for various λ . The distribution of the standard deviation is biased above 0.5 and the bias decreases as λ increases. Figure 5.78 shows normal quantile plots for square root Poisson samples of size 500. To compare these results with the properties of the noise component, figure 5.79 shows a plot of the distribution of the observed sample standard deviations $\hat{\sigma}_n$ for the 30 EDs. These standard deviations are all greater than 0.5, indicating a sys-

tematic departure, but are not far from 0.5. The consistency of the values can be seen in figure 5.72; the slopes of the lines on the display are nearly the same. The near normality of r_i and the closeness of the $\hat{\sigma}_n$ to 0.5 are consistent with the y_i having a distribution that is well approximated by the Poisson with mean λ_i . However, in carrying out outbreak detection as outlined in the following section, $\hat{\sigma}_n$ is used in place of the theoretical value 0.5 to provide a measure of robustness to the Poisson model.

3.3 Outbreak Detection

STL modeling provides public health officials with a clear view of yearly-seasonal variation for visual monitoring of the onset and magnitude of seasonal peaks. This is facilitated by the smoothness of s_i as opposed to the raw data.

Another critical task is the detection of outbreaks that rise up on the order of several days to several weeks and are not part of the systematic patterns in disease counts; the causes are bioterrorism and uncommon highly pernicious diseases. Following the terminology of meteorology, these will be called “synoptic time-scale outbreaks”.

3.3.1 Detection Method

The synoptic-scale detection method proposed here uses a simple control chart based on the assumption that the daily counts are independently distributed Poisson random variables with a changing mean λ_i . The variability in the systematic components t_i , s_i , and d_i is taken to be negligible which is reasonable since they are relatively low-frequency components. These components are treated as deterministic. The statistical variability in $\sqrt{y_i}$ is chiefly the result of r_i , modeled as independent,

identically distributed normal random variables with variance σ_n^2 . Thus the sample mean of y_i is λ_i , which from (3.1) is calculated as

$$\lambda_i = E[(y_i)] = (t_i + s_i + d_i)^2 + \sigma_n^2 \quad (3.2)$$

where σ_n^2 is estimated by the sample standard deviation of the r_i .

With Y_i as a Poisson random variable with mean λ_i , an outbreak alarm is declared when Y_i results in an observed value y_i for which

$$P(Y_i \geq y_i; \lambda_i) < \alpha \quad (3.3)$$

where α is a designated type I error threshold. To evaluate this synoptic-scale outbreak method, historical data is used for baselines and artificial outbreaks are added, as has been done in other research [Jackson et al. \(2007\)](#); [Mandl et al. \(2004\)](#).

3.3.2 Outbreak Model

The outbreak model used here is a lognormal epicurve of Sartwell [Sartwell \(1950\)](#). An outbreak start day is selected and becomes day 1. A total number of cases, O , is selected; the bigger the value of O , the more readily the outbreak can be detected. The day of each case is a random draw, rounded to the nearest positive integer, from a lognormal distribution whose minimum is 0.5. Due to different representations, here the lognormal density is defined as

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-\frac{(\log x - \zeta)^2}{2\sigma^2}}, \quad (3.4)$$

using $\zeta = 2.401$ and $\sigma = 0.4626$ to approximate temporal behavior of an anthrax outbreak [Meselson et al. \(1994\)](#). The lognormal density times a constant is shown in figure 5.80, which will be fully explained later.

Detectability at a point in time depends on the the number of outbreak cases per day compared with the variability of the baseline counts over the outbreak period. As has been emphasized earlier, the variability of an ED count time series is larger

when the level of the series is larger than when the level is smaller. Because O is fixed, detectability changes through time.

Three values of O were chosen for each baseline, selected based on a method in [Burkom \(2003\)](#). There are many ways to make the selection, but this one was chosen to remain consistent with past work. The maximum of the lognormal epicurve is 0.087 and occurs at $e^{\zeta - \sigma^2} = 8.9$ days. Supposing the density is this value for 8.9 ± 0.5 days, then the expected number of cases in this peak interval is $0.087O$. Detectability can be controlled by making the expected peak-interval cases equal to a magnitude factor f times the sample standard deviation of the residual counts $y_i - (t_i + s_i + d_i)^2$. The parameter f controls the detectability. The values of O for each baseline in this evaluation were chosen by this method with $f = 1, 1.5$, and 2 . For example, one baseline has standard deviation of the residual counts as 3.324. For $f = 2$, $O = 2 \times 3.324 / 0.087 = 76.414$, and rounding to the nearest integer, $O = 76$. This case is shown in figure [5.80](#). The curve is the lognormal density times 76, and the histogram shows 76 draws from the lognormal.

3.3.3 Comparison of Methods

The synoptic-scale method described in section [3.3.1](#) was compared to other methods. Because some surveillance EDs have a rather short history, a comparison was made to the EARS C1, C2, and C3 methods [Hutwagner et al. \(2003\)](#). These methods are based on a simple control chart approach with the daily test statistic calculated as

$$S_t = \max(0, y_t - (\mu_t + \sigma_t) / \sigma_t), \quad (3.5)$$

where μ_t and σ_t are a 7-day baseline mean and standard deviation. For the C1 method, the baseline window for μ_t and σ_t is days $t - 7, \dots, t - 1$ and for the C2 method, the baseline window is days $t - 9, \dots, t - 3$. The test statistic for the C3 method is $S_t + S_{t-1} + S_{t-2}$, where S_{t-1} and S_{t-2} are added if they do not exceed the threshold.

Since some EDs have more than 3 years of data, some of the EARS longer historical baseline methods were considered. However, none of these were implemented in this evaluation since it has been shown that these do not offer much of an advantage over the limited baseline methods [Hutwagner et al. \(2005\)](#). Instead, a comparison was made with a Poisson generalized linear model (GLM) method [Jackson et al. \(2007\)](#), which has terms for day-of-the-week, month, linear trend, and holidays. Holidays were not included since details of its implementation could not be found. Another method considered was a seasonal ARIMA model [Reis and Mandl \(2003\)](#) but its results were found to be unsatisfactory, so it is not included. This could be due to insufficient data for pooling across years, or that such pooling through the seasonal terms of the model cannot accommodate the substantial change from one year to the next in the seasonal patterns.

The STL modeling in section [3.2](#) revealed two possible choices for the smoothing span q for smoothing the yearly-seasonal component, $q = 49$ and $q = 79$. From a detection standpoint, choosing q to be larger may be beneficial because it will be more difficult for an evolving outbreak to become absorbed into s_i . However, if q is too large, the variation attributable to s_i may not be adequately captured. Both values of q are tested.

An investigation was made into the minimum amount of data needed by the STL method to achieve good performance. Because the long-term trend t_i is very stable, and the window for yearly-seasonal component s_i is not very large, one would expect STL to do well with a relatively short baseline. To test this, the outbreak detection method was run at each time point using just the most recent 90 days of baseline data for each outbreak scenario.

Thus, there are 8 methods to be compared:

- The EARS C1, C2, and C3 methods
- A Poisson GLM with terms for day-of-week, month, and linear trend, denoted as “GLM”

- STL decomposition with yearly-seasonal smoothing span $q = 79$, denoted as “STL $q = 79$ ”
- STL decomposition with yearly-seasonal smoothing span $q = 49$, denoted as “STL $q=49$ ”
- STL $q = 79$ with only a 90 day baseline, denoted as “STL(90) $q = 79$ ”
- STL $q = 49$ with only a 90 day baseline, denoted as “STL(90) $q = 49$ ”

3.3.4 Outbreak Injection Scenarios

Outbreak detection was tested using all days of each of the 30 baselines except for the first 365 days, which were used as historical data for the GLM method to have sufficient data to estimate coefficients. Each baseline series had outbreaks of the 3 magnitudes injected into the series starting at day 366 and then sliding daily across the series until the end of the series. Each scenario was tested by each method to determine whether the outbreak could be detected and how quickly. Applying this to all baselines required about 75,000 test cases, which required almost 7 million loess function calls. Running the tests sequentially took about 3.5 days, but spreading the computations out over 64 cores sped up the testing to 1.5 hours. Computations were spread out using the R-Hadoop Integrated Processing Environment (RHIPE), (cite).

Each outbreak was a different random sample from the lognormal epicurve. One sample is shown in figure 5.80. The counts of each outbreak were added to the baseline counts to form y_i . Each method was run through each day of an outbreak as would be done in real practice, keeping track of whether the outbreak was detected and, if so, how long it took to detect. Time until detection is measured by the number of days after first exposure until the day of detection. Any evaluation that did not result in detection on or before day 16 of the outbreak was classified as not-detected.

3.3.5 Choosing Cutoff Limits

The cutoff limits for all methods were chosen so that the observed false positive rate was fixed at 0.03. This was achieved empirically by choosing a cutoff, c , for each method from the historical data with no outbreaks injected. For the C1, C2, and C3 methods, this means choosing c to be the 97th percentile of the daily test statistic values. Figure 5.81 shows the observed proportion of false positives vs. the cutoff limit for the C1 method, by ED. The plots for C2 and C3 are similar. The dashed lines correspond to an observed false positive rate of 0.03 (horizontal line) and the corresponding empirical cutoff limit, c (vertical line). Figure 5.82 compares the cutoff limits for each of the EARS methods for each ED.

In practice, limits for the STL and GLM methods would be set by using the theoretical false positive rate α . If a model fits the data, the observed rates should be consistent with the theoretical rate. Since the STL and GLM methods update past fitted values as they progress, choosing the empirical cutoff for these methods cannot be done by simply fitting the entire time series and retrospectively choosing the cutoff. Instead, the fitted values obtained at the last day of each fit over time were used to obtain empirical cutoffs, since it is the last-day fits that will be used in the outbreak detection algorithm on each day. Each method yields a last-day Poisson distribution parameter estimate λ_i , and the empirical cutoff c was chosen such that the proportion of days for which $P(Y_y \geq y_i; \lambda_i) < c$ was equal to 0.03.

Figure 5.83 shows the observed false positive rate vs. the cutoff limit for the GLM method. The dashed lines carry the same meaning as in figure 5.81. The solid diagonal line is a guide indicating when the observed and theoretical false positive rates are equal. The solid vertical line corresponds to the true false positive rate, $\alpha = 0.03$, which one would hope for the observed rate to be close to. The observed false positive rate fluctuates wildly from one ED to the next, and in most cases is much less than 0.03. This in effect penalizes the method for its lack of fit.

Figures 5.84 and 5.85 show the observed false positive rates vs. the cutoff limit for the STL $q = 79$ and the STL $q = 49$ methods. The results are similar to their equivalent 90-day baseline methods. The STL $q = 79$ observed false positive values do a very good job of yielding cutoff values close what the true false positive rate should be. The cutoff levels for the GLM and all STL methods are shown in figure 5.86.

It is worth noting that even globally fixing the thresholds will not ensure a completely commensurate comparison between methods since a poorly-fitting method may exhibit a changing false positive behavior over the course of the series. The outbreak detection results in the following section will be accompanied by additional evaluation of the methods.

3.4 Results

3.4.1 Results Summary

Table 3.1 shows the overall results for each outbreak detection method, averaging over EDs and outbreak magnitudes. The percentage of outbreaks detected and the average days until detection are reported. The STL methods clearly outperform the others, with the $q = 49$ version performing best. Also, the limited baseline STL methods essentially perform just as well as their full baseline counterparts, indicating that the STL method can be used with as little as 90 days of historical data available.

The days until detection are recorded including the incubation period, and referring to figure 5.80, outbreak cases usually do not begin until day 3 or 4. Detection before day 6 or day 7 is before the expected outbreak peak. The C1 method typically has the earliest detection, but has the worst sensitivity.

Figure 5.87 plots the proportion of outbreaks detected vs. outbreak magnitude for each ED for the C3, GLM, and STL $q = 49$ methods. While the GLM method is very competitive, the STL method is dominant in nearly every case and is much more consistent. Comparisons of sensitivity across baselines within each method and

Table 3.1
Overall percentage of outbreaks detected by method.

Method	Percentage Detected	Mean Days until Detection
EARS C1	61.62	5.59
EARS C2	64.59	5.80
EARS C3	64.43	6.03
Poisson GLM	72.23	6.41
STL, $q = 79$	76.32	6.44
STL(90), $q = 79$	75.46	6.45
STL, $q = 49$	80.73	6.70
STL(90), $q = 49$	79.99	6.77

magnitude show quite a bit of variability. This may be due to unique characteristics of each ED, the random outbreak generation, or the fact that the cutoff values were set empirically within each baseline.

Figure 5.88 shows where in time the methods failed to detect an injected outbreak for one ED with outbreaks injected with magnitude $f = 2$. The yearly-seasonal component is plotted and a vertical line is overplotted corresponding to the start day of outbreaks that the methods failed to detect. The EARS and GLM methods exhibit long runs of failed detections and long runs of successful detections, indicating probable bias in the fitted values or detection algorithm. Most striking is the inability of these two methods to detect outbreaks that begin during the decline of the seasonal flu. The STL methods have a more spread-out distribution of failed detections, although they also exhibit small amounts of bunching in time. Also, the STL methods do much better at detecting outbreaks that occur during the seasonal flu decline.

3.4.2 Examining the Difference in Detection Performance

To better understand the difference in performance across outbreak methods, the model fitted values (predicted systematic values) and residuals (data minus the fits) from each method were investigated. As has been emphasized, the methods of statistical modeling upon which each detection method is based have a large impact on performance. For the EARS methods, the fitted value for day i is the seven-day baseline raw data mean μ_i from Equation 3.5. For the GLM and the STL methods, the fitted value for day i is the evaluation of the systematic components at time i for a fit through time i , just as it would be used in practice for outbreak detection. GLM bases this on data for a number of years.

Figure 5.89 shows residual diagnostic plots to check for lack of fit in the EARS, GLM, and STL methods for the daily respiratory counts for one ED. The last-day residuals against i for each model; a loess curve shows the local average fluctuations of the residuals around 0. Substantial lack of fit is evident for the GLM method, systematic fluctuations from 0. This is due to pooling across years to estimate the yearly seasonal component, which can be quite different from one year to the next. Lack of fit revealed for EARS and STL is minor.

Figure 5.90 checks the variability in the EARS, GLM, and STL fitted values for the same ED of the previous two figures. The fitted values should be as smooth as possible without introducing lack of fit. For EARS, the fitted values are graphed against i . To make comparisons of EARS with GLM and STL commensurate, the GLM and STL fitted values minus their day-of-week components are plotted against i . The EARS fit is slightly noisier than that of STL, the result of overfitting. In particular there is a large amount of synoptic scale variation, making it more difficult to detect synoptic-scale outbreaks. The EARS fit also appears to be slightly more delayed in time than the STL fits. The GLM fits are locally quite smooth at most points in time, but have large precipitous jumps at certain time points because the

yearly seasonal component is modeled as a step function, constant within months, but changing discontinuously from one month to the next.

3.4.3 False Positives

Practical application of syndromic surveillance methods is dependent on reliable methods that can effectively detect outbreaks but are not prone to false alarms. Trust will be lost in a system that gives off far too many false alarms and therefore proper attention might not be given when an alarm is correctly identified. Hence, a study is also done examining the false positive behavior of these methods for a fixed theoretical cutoff limit.

In section 3.3.5, theoretical thresholds were determined that resulted in a fixed observed false positive rate of 0.03 when run on the raw baselines. In applied settings, typically the opposite is done, setting the cutoff based on a theoretical threshold. Thus, here for a set theoretical threshold $\alpha = 0.03$, the observed false positive rate is computed on the raw baselines.

Figure 5.91 shows quantile plots of the observed false positive rates from the GLM and STL methods for the historical data with no outbreaks present and a theoretical false positive rate of 0.03 for all 30 EDs. The observed rates for the GLM methods deviate from 0.03 by far more than for STL. Running the GLM method at a theoretical rate of $\alpha = 0.03$ typically results in an observed false positive rate of 0.05 and ranges up to 0.1. The STL methods appear to yield fewer observed false positives on average than the theoretical limit. Figure 5.92 shows the location of where the false positives occurred in time for one ED.

3.5 Discussion

For this data, the same smoothing parameters were successfully used across all EDs. Others interested in applying these methods to other syndromic data should perform thorough parameter selection and model validation and not rely on the pa-

rameters chosen for this data. For outbreak detection, this data behaved like Poisson random variables, but this may not be the case in other applications. This assumption should be checked and if it does not hold, other possibilities should be investigated such as the over-dispersed Poisson. Another alternative is validating distributional properties of the remainder term and using this term for monitoring.

Although the detection performance results presented here favor the STL method, this does not necessarily mean that it is the best method available. There are certainly other methods with which comparisons could have been made. The EARS methods were chosen mainly because they are widely used and can be applied to very short disease time series. This detection study was limited to one type of outbreak. Further use and study of this method will determine its merit. The visualization capabilities of STL modeling make it a useful tool for visual analytics.

The STL method should not be used for small counts close to and including zero. While they work well with the large counts of the respiratory and gastro-intestinal categories, many other categories such as botulinic have counts that are too small for the square roots to be approximately normally distributed. Future work can investigate employing STL ideas for small counts by replacing the square root normal distribution and local least-squares fitting with the Poisson distribution and local Poisson maximum likelihood fitting.

3.6 Conclusions

The STL decomposition methods presented here effectively model chief complaint counts for syndromic surveillance without significant lack of fit or undue noise, and lead to a synoptic time-scale disease outbreak detection method that in these testing scenarios performs better than other methods to which is it compared — EARS C1, C2, and C3 methods [Hutwagner et al. \(2003\)](#), as well as a Poisson GLM modeling method [Jackson et al. \(2007\)](#). The methods can be used for disease time series as short as 90 days, which is important because many surveillance systems have started only

recently and have a limited number of observations. Visualization of the components of variation — inter-annual, yearly-seasonal, day-of-the-week, and random-error — provides much insight into the properties of disease time series.

3.7 Other Work

Other work has been done with this data in cooperation with the Purdue University Rendering and Perceptualization Lab (PURPL). This work includes incorporation of these methods into visual analytics systems, generating synthetic data sets, and spatio-temporal modeling. References include [Maciejewski et al. \(2008\)](#); [Hafen et al. \(2009\)](#); [Maciejewski et al. \(2008\)](#).

4. A NEW DENSITY ESTIMATION METHOD: ED

4.1 Introduction

This chapter introduces a new univariate nonparametric density estimation, *ed* which is based on local regression methods.

4.1.1 Background

Nonparametric density estimation has received substantial attention in the research literature of statistics and machine learning. A search of “density estimation” the ISI Web of Knowledge from 1950 to 2008 revealed over 22,000 papers. The principal reason is that the density of a distribution, univariate or multivariate, is such a fundamental statistical property.

Density estimation got its start as a major research area with the work of [Rosenblatt \(1956\)](#) and [Parzen \(1962\)](#) on fixed bandwidth kernel density estimation. A kernel, which is a probability density such as the normal, is centered at each point in the data. The unnormalized density estimate at any point is the average of the kernels at that point. There are many excellent reviews centering on kernel methods ([Silverman, 1986](#); [Izenman, 1991](#); [Scott, 1992](#); [Jones et al., 1996](#); [Sheather, 2004](#)).

The kernel approach is simple and allows very fast computational algorithms ([Silverman, 1982](#); [Gray and Moore, 2003a,b](#)). However, as discussed widely in the research literature, performance can be poor for a number of density patterns occurring in applications. For example, peaks can be chopped and valleys filled in, and significant bias can occur at a boundary for x when there is a sharp cut-off of the density at the boundary. Improvements to kernel density estimates have focused on reducing the

bias through higher-order kernels and allowing the bandwidth to be adjusted locally, but results for this hard problem have not been entirely successful (Loader, 2004).

Diagnostic methods for assessing whether a kernel estimate follows the density patterns in data have been put forward (Loader, 1999; Marron and Udina, 1999; Farmen and Marron, 1999), but the form of the estimation limits the possibilities, so there does not exist a set of comprehensive diagnostic tools, as there are, for example, for regression diagnostics (Daniel et al., 1980). The problem is that there is not a notion of local goodness-of-fit that can be exploited in a visual diagnostic tool, such as viewing scatterplots with fitted lines and viewing residuals in the regression setting. Automated model selection criteria for kernel estimates can help in finding an estimate that fits the data (Sheather, 2004), but the criteria are global and average out local places where lack of fit or overfitting occur.

4.1.2 Summary of Ed

The ed method has been designed to provide a model building approach to density estimation. The ed *estimation* method can readily fit a wide range of density patterns in data, and it enables the use of a rich set of visual displays for *diagnostic* checking of different ed fits and assumptions accompanying the fits. Ed does this by transforming density estimation into regression estimation, making available the models, methods, diagnostic checks, and computational methods of regression. The ed approach also includes a theoretical framework based on the theory of gaps for order statistics. The framework is useful for the fitting process and making statistical inferences based on the ed estimator.

The two-step estimator begins with a very small bandwidth balloon density estimate at x , which is the inverse of distance to the κ -th nearest neighbor of x (Loftsgaarden and Quesenberry, 1965; Tukey and Tukey, 1981). The balloon density estimate is computed at certain values of x . The log of the balloon estimate and the values of x provide a response and a factor that are treated as a univariate regression that theory

suggests can be treated as having normal errors with constant known variance. A κ of about 10 typically works well to ensure this error distribution. Next, a regression curve is fitted using loess (section 1.2). Polynomial fitting results in good bias and variance properties (Fan, 1993). The normal approximation can be checked in any application, and if it is warranted, the loess mechanism for statistical inference with normal errors provides characterizations of uncertainty (section 1.2.3).

Fitting polynomials to the log of the density is not new. Loader (1996) introduced a maximum likelihood method used directly on the raw data that has excellent variance-bias properties. But such direct maximum likelihood estimation does not facilitate powerful regression diagnostics. The ed approach works without sacrificing efficiency because just the small amount of smoothing of the balloon estimate followed by the log transformation tends to create a normal-error regression without introducing more than negligible bias.

The log transformation can perform poorly in certain applications when the data are sparse over a significant portion of the interval within which the data lie. One method to overcome this is gridding: pseudo-data are added over the interval in the form of a grid, and the estimate corrected at the end.

4.1.3 Overview of the Chapter

Section 4.2 introduces three data sets that will be used throughout to illustrate ed and problems that the traditional kernel density estimator can encounter. Section 4.3 deals with the raw estimates, based on the κ -th nearest neighbor balloon estimate. Approximate distributional properties of the estimate are derived and the choice of κ is discussed. Section 4.4 addresses smoothing the raw estimates using loess, and section 4.5 applies methods of model selection, model diagnostic checking, and inference. Section 4.6 illustrates difficulties the traditional kernel density estimate can encounter and compares the kernel estimates to the ed estimates for the example

data sets. Section 4.7 outlines a grid augmentation approach for estimating densities with sparse regions.

4.2 Examples

In this section, three data sets — named “Packet-Delay”, “Family-Income”, and “Normal-Mixture” — are introduced to illustrate the ed method as well as the well-known problems that the traditional kernel density estimator can encounter.

Packet-Delay is from a simulation study of quality of service for voice over the Internet (VoIP). The data are the 264,583 positive queueing delays for packets arriving for transmission on a router output link. The simulation recreates the router queueing mechanism, and arrivals at the queue are generated by a validated model for VoIP packet arrival times (Xi et al., 2009). Some interesting features in the density of this data are possible discontinuities and a sharp cutoff of the density at 0 ms. A quantile plot for this data is shown in figure 5.93.

Another example that has had appearances in the density estimation literature is the family income data from the Family Expenditure Survey in the United Kingdom. This data was studied in detail in Marron and Schmitz (1992). The data consists of 7,201 family incomes for the year 1975, scaled to have a mean of 1. A preliminary analysis revealed large outliers, which were removed, reducing the size of the data to 7,162 observations. It is justifiable to remove these observations since it is not realistic to suppose that good estimates can be obtained in the extremities of the tails of the distribution by nonparametric methods. A quantile plot of the data with the outliers removed is shown in figure 5.94. Of interest in this data is the existence of a class system, indicated by multiple modes in the density. Different smoothing parameter choices lead to different conclusions.

The final example is a simulated mixture of normal random variables

$$\frac{1}{4} (N(0, 0.25) + N(3.5, 0.25) + N(5, 0.75) + N(10, 1.5)),$$

using the notation $N(\mu, \sigma)$ to specify the normal distribution with mean μ and standard deviation σ . This density exhibits multiple modes of varying magnitude. It is used to illustrate problems with fixed bandwidth kernel density estimation methods, and also for simulation comparisons in which knowing the true density is desirable. A quantile plot of a sample of size 3,000 from this density is shown in figure 5.95.

4.3 Estimation Step 1: Raw Estimates

The first step in the ed procedure is calculating the raw estimates. Consider a random sample of independent observations x_1, \dots, x_m from a random variable X with unknown density f and order statistics denoted $x_{(1)}, \dots, x_{(m)}$. The raw estimates are based on non-overlapping gaps of groups of κ order statistics,

$$g_i^{(\kappa)} = x_{(i\kappa+1)} - x_{((i-1)\kappa+1)}, \quad i = 1, \dots, n, \quad (4.1)$$

where $n = \lfloor (m-1)/\kappa \rfloor$. For example, if $\kappa = 10$, then

$$g_1^{(10)} = x_{(11)} - x_{(1)}, \quad (4.2)$$

$$g_2^{(10)} = x_{(21)} - x_{(11)}, \quad (4.3)$$

$$\dots \quad (4.4)$$

Using the gaps, a quantity is computed that [Tukey and Tukey \(1981\)](#) call the *balloon density estimate*,

$$\hat{b}_i^{(\kappa)} = \frac{\kappa}{mg_i^{(\kappa)}}, \quad i = 1, \dots, n. \quad (4.5)$$

The balloon density estimate will serve as the basis of the response variable in a regression problem. Each estimate, $\hat{b}_i^{(\kappa)}$, is positioned at the midpoints of the gap intervals,

$$x_i^{(\kappa)} = \frac{x_{(i\kappa+1)} + x_{((i-1)\kappa+1)}}{2}. \quad (4.6)$$

Note that when positioned at $x_i^{(\kappa)}$, the estimates $\hat{b}_i^{(\kappa)}$ are equivalent to the uniform-weight nearest neighbor density estimation estimates proposed by [Loftsgaarden and](#)

Quesenberry (1965) evaluated at $x_i^{(\kappa)}$. Figure 5.96 illustrates a simple case of calculating the raw estimates $(x_i^{(\kappa)}, \hat{b}_i^{(\kappa)})$ for a sample $x = (1, 3, 7, 2, 15, 17, 19)$ with $\kappa = 2$.

Favorable distributional properties result when working with the raw estimates on the log scale. It turns out that $\log \hat{b}_i^{(\kappa)}$ are approximately independent and follow a non-standard log-gamma distribution, with

$$E[\log \hat{b}_i^{(\kappa)}] = \log f(x_i^{(\kappa)}) + \log \kappa - \psi_0(\kappa), \quad (4.7)$$

$$\text{Var}(\log \hat{b}_i^{(\kappa)}) = \psi_1(\kappa), \quad (4.8)$$

where ψ_0 and ψ_1 are the digamma and trigamma functions. A derivation of this is provided in section 4.3.2.

With this result, the raw estimate is computed as

$$\hat{y}_i^{(\kappa)} = \log \hat{b}_i^{(\kappa)} - \log \kappa + \psi(\kappa) \quad (4.9)$$

so that

$$E[\hat{y}_i^{(\kappa)}] = \log f(x_i^{(\kappa)}). \quad (4.10)$$

Figures 5.97–5.99 show plots of $(x_i^{(\kappa)}, \hat{y}_i^{(\kappa)})$ for the three examples described in section 4.2. For the Packet-Delay data, there was one waiting time with as many as 67 ties, so a good choice is $\kappa = 75$. For this data, a value of $\kappa = 75$ is quite small relative to the sample size of 264,583. For the Normal-Mixture data and the Family-Income data, $\kappa = 10$ was chosen. More on the selection of κ will be discussed in section 4.3.1.

These figures demonstrate the merit of viewing the raw estimates as a visualization technique, especially in the case of the Packet-Delay data. A discontinuity at 0.16 ms is prominent in the raw estimate plot, and another appears to be present at 0.32 ms. It would be much more difficult to unequivocally declare or even notice these points as discontinuities through some other technique like a histogram or a kernel density estimate. Further investigation revealed a simple explanation for these discontinuities – the service time of a packet is a constant time of 0.16, so packets that arrive when there is another packet being served will wait in the queue at most

0.16 time units, while packets that arrive when there is already one packet in the queue and another being served will wait in the queue at least 0.16 but at most 0.32 time units, etc. While this phenomenon should have been obvious from the start, visualizing the density via the raw estimates gave a quick understanding of what should have already been known about the data.

Figure 5.98 reveals the bimodal nature of the income distribution. The two peaks will become more evident after smoothing the raw estimates.

Figure 5.99 shows that the theoretical log density fits through the raw estimates for the Normal-Mixture sample nicely, affirming that the raw estimates are leading in the right direction.

4.3.1 Choosing κ

There are two considerations for the choice of κ . First, κ should be small enough that there is as little distortion of the density as possible by the averaging that occurs, since the assumption is that the density is approximately uniform in small neighborhoods. Second, κ should be large enough that $\hat{y}_i^{(\kappa)}$ is approximately normal. Choosing $\kappa = 10$ works quite well for approximate normality, and if m is not small, is adequate for not distorting the density.

To illustrate some of the consequences of choice of κ , data was simulated from the Normal-Mixture data and raw estimates were computed with different κ . Since f is known, the errors $\hat{y}_i^{(\kappa)} - \log f(x_i^{(\kappa)})$ can be studied. Figure 5.100 compares the sample order statistics of the errors with the expected order statistics of the normal with variance $\psi_1(\kappa)$ and the non-standard log-gamma distribution for different values of κ , for a random sample of size 3,000 from the Normal-Mixture data. The normal approximation seems to work well for $\kappa = 10$, but not for $\kappa = 1$ and $\kappa = 4$.

With proper choice of κ , the raw estimates, $(x_i^{(\kappa)}, \hat{y}_i^{(\kappa)}), i = 1, \dots, n$, provide a regression problem, where the observations are independent, the error is approximately normal with constant known variance, and the expected value of $\hat{y}_i^{(\kappa)}$ is the log den-

sity $\log f(x_i^{(\kappa)})$. Smoothing the raw estimates will result in a good estimate of the log density.

4.3.2 Approximate Distribution of the Raw Estimates

For a sequence of m independent identically distributed random variables X_1, \dots, X_m with continuous density f and with order statistics denoted as $X_{(1)}, \dots, X_{(m)}$, consider the non-overlapping gaps of groups of κ order statistics,

$$G_i^{(\kappa)} = X_{(i\kappa+1)} - X_{((i-1)\kappa+1)}, \quad i = 1, \dots, n, \quad (4.11)$$

where $n = \lfloor (m-1)/\kappa \rfloor$, which is a random quantity. An alternate way to view this is as the sum of distances to consecutive order statistics,

$$G_i^{(\kappa)} = \underbrace{(X_{((i-1)\kappa+2)} - X_{((i-1)\kappa+1}))}_{\Delta_1^{(i)}} + \dots + \underbrace{(X_{(i\kappa+1)} - X_{(i\kappa)})}_{\Delta_\kappa^{(i)}}. \quad (4.12)$$

Recall that $G_i^{(\kappa)}$ is used to calculate an estimate of the density positioned at the interval midpoints,

$$X_i^{(\kappa)} = \frac{X_{(i\kappa+1)} + X_{((i-1)\kappa+1)}}{2}. \quad (4.13)$$

When κ is small, the density of f around the neighborhood of $X_i^{(\kappa)}$ is approximately uniform, so that each of the κ distances, $\Delta_1^{(i)}, \dots, \Delta_\kappa^{(i)}$ behave approximately like an exponential distribution with rate $mf(X_i^{(\kappa)})$. With this distributional assumption for each Δ ,

$$E[G_i^{(\kappa)}] \approx \frac{\kappa}{mf(X_i^{(\kappa)})}. \quad (4.14)$$

For observed data, replacing the expected value with the observed $g_i^{(\kappa)}$ and solving the above for $f(x_i^{(\kappa)})$ yields the balloon estimate in equation 4.9.

With the approximate distribution of $G_i^{(\kappa)}$ being gamma, the balloon estimate

$$\hat{B}_i^{(\kappa)} = \frac{\kappa}{mG_i^{(\kappa)}}, \quad i = 1, \dots, n \quad (4.15)$$

has an approximate inverse gamma distribution whose variance depends on $f(X_i^{(\kappa)})$.

Now, to simplify notation, use B and G instead of $\hat{B}_i^{(\kappa)}$, and $G_i^{(\kappa)}$ and consider the transformation $Y = \log B$. With this,

$$\begin{aligned} F_Y(y) &= P(\log B \leq y) \\ &= P(\log \kappa - \log m - \log G \leq y) \\ &= 1 - F_G(\exp\{\log \kappa - \log m - y\}). \end{aligned} \quad (4.16)$$

The gap, G , has an approximate gamma distribution shape parameter κ and rate parameter $\lambda = mf(x_i^{(\kappa)})$, with probability density function

$$f_G(y; \kappa, \lambda) = \frac{\lambda^\kappa}{\Gamma(\kappa)} y^{\kappa-1} e^{-\lambda y}, \quad x > 0. \quad (4.17)$$

Now let $a = \log \kappa - \log m$. Differentiating (4.16),

$$\begin{aligned} f_Y(y) &= -f_G(e^{a-y})(-e^{a-y}) \\ &= \frac{1}{\Gamma(\kappa)} \exp\{-\kappa(y - a - \log \lambda) - e^{-(y-a-\log \lambda)}\} \\ &= \frac{1}{\Gamma(\kappa)} \exp\left\{-\kappa(y - \log \kappa - \log f(X_i^{(\kappa)})) - e^{-(y-\log \kappa - \log f(X_i^{(\kappa)}))}\right\}, \end{aligned} \quad (4.18)$$

where the last step was obtained by substituting back in $a = \log \kappa - \log m$ and $\lambda = mf(X_i^{(\kappa)})$. Equation 4.18 is the probability density function for the non-standard log-gamma distribution (Kotz and Nadarajah, 2000), with

$$E[Y] = \log f(x_i^{(\kappa)}) + \log \kappa - \psi_0(\kappa) \quad (4.19)$$

$$\text{Var}(Y) = \psi_1(\kappa), \quad (4.20)$$

where ψ_0 and ψ_1 are the digamma and trigamma functions, respectively.

Thus the log transformation yields an unbiased estimate, up to an additive known constant, of the log density at $x_i^{(\kappa)}$, and that the variance is now constant. Of course, these results are approximate, based on the assumption of near-uniformity of the density in small neighborhoods.

Figure 5.99 gives some empirical validation of the expectation property (4.19) of the raw estimates since the true log density fits nicely through the raw estimates.

To give some empirical support to the variance property (4.20) of the raw estimates, figure 5.101 plots the variance of the errors $\hat{y}_i^{(\kappa)} - \log f(x_i^{(\kappa)})$ for several samples of size 3,000 from the Normal-Mixture data at different values of κ . The black line joining through the circles represents the trigamma function, and it passes nicely through the median variance value for each κ . Also, figure 5.100 provides empirical validation of the normality property of the raw estimates for the Normal-Mixture data.

4.4 Estimation Step 2: Loess Smoothing

Local regression methods are used to smooth the raw estimates, $(x_i^{(\kappa)}, \hat{y}_i^{(\kappa)})$, $i = 1, \dots, n$. Since the approximate distribution of the response variable is known to be a non-standard log-gamma, local likelihood methods could be used (Hastie and Tibshirani, 1987) to obtain a log-density estimate. However, for large enough κ , the distribution of $\hat{y}_i^{(\kappa)}$ tends to the normal distribution, enabling the use of loess smoothing (section 1.2). Throughout the presentation of smoothing the raw estimates, the simplified notation of (x_i, y_i) is used in place of $(x_i^{(\kappa)}, \hat{y}_i^{(\kappa)})$.

From section 4.3, for large enough κ , y_i is approximately normal with expected value equal to the log of the density at x_i . Thus, the model is

$$y_i = \ell(x_i) + \epsilon_i, \quad i = 1, \dots, n, \quad (4.21)$$

where $\ell(x_i) = \log f(x_i)$ and ϵ_i is approximately normal with

$$E[\epsilon_i] = 0, \quad \text{Var}(\epsilon_i) = \psi_1(\kappa), \quad (4.22)$$

where ψ_1 is the trigamma function. Since the raw estimates are computed at disjoint intervals, the ϵ_i are also independent.

Once the loess fit $\hat{\ell}(x_0)$ is obtained, the ed density estimate is calculated by exponentiating,

$$\hat{f}(x_0) = \exp(\hat{\ell}(x_0)). \quad (4.23)$$

Although the ed estimate is not guaranteed to integrate to one, experience has shown that a good fit results in an estimate whose numeric integral is extremely close

to one. If one desires to have an estimate that integrates exactly to one, it is possible to renormalize.

Figure 5.102 and 5.104 show loess fits to the Packet-Delay and Normal-Mixture raw estimates, and figures 5.103 and 5.105 show the exponentiated fits. The Family-Income data will be studied separately in the next section, where it receives more attention in terms of parameter selection and model diagnostics.

The Packet-Delay data was fitted separately on each interval of continuity. Although in general the assumption is that the density is continuous, if discontinuities are noticed in the raw estimates as in this case, each interval of continuity can be fit separately. Loess smoothing was applied to the raw estimates on each interval, using $\delta = 1$, $\alpha = 0.75$, for the lowest interval, $\delta = 2$, $\alpha = 0.5$, for the middle interval, and $\delta = 1$, $\alpha = 1.5$ for the highest interval.

The Normal-Mixture data was fit using $\delta = 3$ and $\alpha = 0.24$. As can be seen in the figure, the resulting fit is quite good. Selection of δ and α was done Mallows' C_p (section 1.2.4).

4.5 Diagnostic Checking, Model Selection, and Inference

With density estimation set as a regression problem, the multitude of visual and numeric diagnostic tools are available as described in section 1.2.4 to evaluate the goodness-of-fit of the ed estimate, aid in parameter selection, and check the validity of model assumptions.

4.5.1 Evaluating Goodness-of-Fit

In the density estimation setting, typically peaks and valleys are present so one can usually expect λ to be 2 (local quadratic fitting) or 3 (local cubic fitting). Diagnostics are carried out on the fits and residuals, where the residuals are calculated as

$$\hat{\epsilon}_i = y_i - \hat{\ell}(x_i). \quad (4.24)$$

Figures 5.106 and 5.107 show the residuals for the Packet-Delay and Normal-Mixture fits shown in figures 5.102 and 5.104. The Packet-Delay residuals indicate that perhaps a smaller bandwidth should be used in the 0–0.16 ms range.

Throughout the remainder of this section, attention will be given to the Family-Income data. Figures 5.108 and 5.109 show the fitted density and residuals for 6 different choices of α for local quadratic fitting ($\lambda = 2$) to the Family-Income raw estimates. The fit is very rough for small α and gets smoother as α increases. However, as α gets too large, bumps begin to be apparent in the residuals, as emphasized by a loess smooth, indicating that the larger bandwidths are cutting off the peaks. Here, $\alpha = 0.14$ looks best in terms of smoothness of the fit and well-behaving residuals.

4.5.2 Mallows C_p Model Selection

A C_p plot for the Family-Income data can be seen in figure 5.110. After viewing the C_p plot and the fitted value and residual diagnostic plots, parameter choices $\lambda = 2$, $\alpha = 0.16$ seem to be suitable, although a local cubic model will work well also. Plots of the final loess fit and residuals for these parameters can be seen in figure 5.111.

4.5.3 Validating Model Assumptions

To validate the assumptions made in modeling, several diagnostic plots were made as described in section 1.2.4. Figures 5.113–5.115 show plots checking the assumptions of constant variance, independence variance, and approximate normality for the residuals of the Family-Income ed fit with $\lambda = 2$ and $\alpha = 0.16$. It appears that the assumptions hold, making inference possible.

4.5.4 Inference

The approximate normality of the residuals and the capabilities of loess enable construction of confidence intervals for the ed density estimate. Doing so assumes

that the estimate is unbiased. The diagnostic methods presented in section 4.5 assist in determining whether there is bias present in the loess modeling, but this does not account for bias that may have been introduced when computing the raw estimates. If the diagnostics look good, it is safe to assume the bias is negligible.

Pointwise confidence limits can be computed as outlined in section 1.2.3. Figure 5.116 shows the ed fit to the British income data density with 99% pointwise confidence limits.

4.6 Comparison with Kernel Density Estimates

The traditional fixed-bandwidth kernel density estimate (KDE) was introduced over a half-century ago by Rosenblatt (1956) and Parzen (1962) and is arguably still the most commonly used density estimation procedure, next to the histogram (Silverman, 1986). This is most-likely due to the simplicity of the method and its widespread availability in software implementations.

Given data x_1, \dots, x_m from a density f , the KDE $\hat{f}(x)$ is calculated as

$$\hat{f}(x) = \frac{1}{mh} \sum_{i=1}^m K\left(\frac{x - x_i}{h}\right) \quad (4.25)$$

where K is a kernel function that integrates to 1 and is typically a symmetric probability density function. The parameter h is the bandwidth smoothing parameter that controls the variance of the kernel and hence the amount of smoothing.

The quality of a KDE is much more dependent on choice of h than choice of K . The estimated density curve inherits all smoothness and differentiability properties of the kernel function, but the efficiency of the estimator quite similar across many kernel choices (Silverman, 1986). In this section, the gaussian kernel is used with mean 0 and variance 1.

Choosing h has been a major area of research and there are many automatic selection criteria, four of which are utilized in this section: Silverman's rule (Silverman, 1986), least squares (unbiased) and biased cross-validation (Scott and Terrell, 1987), and the Sheather-Jones method (Sheather and Jones, 1991). Typically the

Sheather-Jones bandwidth is recommended. An excellent discussion of these bandwidth selection methods can be found in [Sheather \(2004\)](#). A problem with these and other bandwidth selection methods is that they rely on global and/or asymptotic properties.

The three examples are used in this section to highlight some problems with the traditional kernel approach, namely problems with: fixed bandwidths, boundaries and discontinuities, bias, and choosing optimal smoothing parameters. These difficulties are contrasted with the simplicity with which `ed` deals with these problems. There have been several improvements to kernel density estimation methods to address these problems, such as variable bandwidths, boundary kernels, and higher-order kernels ([Silverman, 1986](#); [Wand and Jones, 1995](#); [Karunamuni and Alberts, 2005](#)), but `ed` can manage all of these issues within a simple framework, while providing insight to the analyst as to whether these problems are present or not.

4.6.1 The Family-Income data

Figure [5.117](#) shows kernel density estimates for 6 different bandwidths for the income data, ranging from $h = e^{-4}$ to $h = e^{-1}$. The largest bandwidth yields a unimodal estimate, while the smallest bandwidth exhibits a multi-modal estimate. Thus different choice of bandwidth can lead to different conclusions on whether the income distribution indicates the presence of an income class system. Here one would definitely want to know the “correct” bandwidth with confidence.

Another possible problem that is brought to attention is that if a small bandwidth is desired to adequately capture the first peak, the smoothness in the tail is sacrificed. Also, positive probability is estimated for negative incomes for some of the bandwidth choices. This could be corrected with a boundary kernel, or with a power transformation of income as is done in [Wand and Jones \(1995\)](#). Finally, sharp peaks in densities are usually chopped with KDEs. This can be explained by the equivalence of the KDE to local constant fitting of the log density ([Loader, 1999](#)),

which cannot adequately capture curvature. Thus considerable bias can be expected in this and all densities with peaks.

The ed fit of this data as studied in section 4.5 has given confidence in the bimodal nature of this density and a satisfactory fit has been obtained. Figure 5.118 shows the exponentiated final ed fit compared to the fit obtained from kernel density estimation using the Sheather-Jones bandwidth selection criterion. The ed fit has eliminated the wiggles in the upper tail, and the first peak is more pronounced.

4.6.2 The Packet-Delay Data

Figure 5.119 shows the KDE for the Packet-Delay data, using a bandwidth chosen by Silverman's rule. Due to the large size of the data, the other selection criteria took a very long time to compute and did not find a suitable parameter choice. This data illustrates the problem kernel density estimates can have with discontinuities and boundaries. The KDE trails off at 0 ms and gives positive density for negative waiting times. From figure 5.97, it is clear that the density cuts off abruptly at 0 ms. The problem with the KDE lies in the fact that at the boundary, there is no data on the left side of the kernel.

The KDE hints at the discontinuity at 0.16 ms. One may try fitting a KDE separately on each interval, as was done with the ed fit, but doing so would lead to the boundary problem at the endpoints of each interval. Of course a boundary kernel method could be used, but these problems are more easily exploited and modeled with ed.

4.6.3 The Normal-Mixture Data

Kernel density estimate plots for the Normal-Mixture data are shown in figure 5.120. The solid gray line in the density plots is the true density. The bandwidths used in the density plots were calculated based on the four criteria discussed above. The density plot shows how extremely different results can be obtained from the

different bandwidth selection criteria. Silverman's rule and biased cross-validation indicate a tri-modal density. Relying on one of these methods could lead to confusion over which features are really present. Also note that the Sheather-Jones and unbiased cross-validation bandwidths are more faithful to the true density, especially capturing the narrowest peak, but exhibit too many wiggles for the first and fourth peak. This is a case where a variable bandwidth would be beneficial.

While a multitude of density estimation methods exist that may be able to deal with many of the issues that have been identified in these examples, or while extra care may avoid some of the problems discussed, the ed method lends itself as a simple comprehensive approach to visualizing and modeling these densities.

4.7 Grid Augmentation

One problem with fitting on the log scale is the fact that it is difficult to estimate densities when they are close to 0. For example, in the Normal-Mixture ed fit shown in figure 5.105, the difficulty with estimating the troughs is evident, although for this example it is not as evident when viewing the exponentiated fit.

To deal with low density regions, the data can be augmented with a grid of uniform points, creating a mixture of the form

$$g(x) = (1 - \omega)f(x) + \omega u(x), \quad (4.26)$$

where f is the true density, u is a uniform density over the range of the data plus a buffer if the support of the density is not finite on either end, and ω is the desired proportion of mixing. The ed procedure is then applied to the augmented data and the estimate of $\log g(x)$ is used to obtain the estimate of $f(x)$.

The density $u(x)$ is uniform over an interval (a, b) . Choice of a and b depends on the data at hand. In the case of a density that does not trail off at one or both tails, one can choose a and/or b to be the minimum or maximum of the observed data, or a value that makes sense in the context of the data (such as $a=0$ for non-negative data). If the tails of the density trail on either end, the range of the uniform should extend

beyond the range of raw estimate design points to capture the transition of the f into u . Furthermore, the range of the uniform should extend even further so that loess estimation at the tails of f incorporates data on both sides of the estimation point. The uniform limits are computed as

$$a = 2x_{(1)} - x_{(\lceil \alpha m \rceil)} \quad \text{and} \quad b = 2x_{(m)} - x_{(m - \lceil \alpha m \rceil + 1)}. \quad (4.27)$$

To obtain a grid-augmented fit, the user specifies a desired proportion of blending ω . A grid of equally-spaced points is generated (not a random uniform sample) over (a, b) . The number of points to generate is

$$m_g = \left\lfloor \frac{\omega m}{1 - \omega} \right\rfloor, \quad (4.28)$$

so that the original data x_i , $i = 1, \dots, m$ is augmented with values

$$u_i = a + (i - 1) \frac{b - a}{m_g - 1}, \quad i = 1, \dots, m_g. \quad (4.29)$$

Putting both data sets together yields a new sample, x_i^* , $i = 1, \dots, m^*$, where $m^* = m + m_g$, from which the raw estimates can be calculated.

With the u_i defined as above, a slight adjustment needs to be made to ω to ensure that in regions where the uniform density dominates, the raw estimates obtained from the augmented data agree exactly with the log of the uniform density $u(x; a, b)$. For a value x a region where all κ neighbors are from the uniform grid, the raw estimate will be

$$\hat{\ell}(x) = \log \frac{k}{m(u_{\kappa+1} - u_1)} - \log \kappa + \psi(\kappa). \quad (4.30)$$

Since x lies in a region surrounded by only uniform points, assume that $f(x) = 0$, so that $\log g(x) = \log(\omega u(x))$. Then the raw estimates in this region need to match $\log g$, which gives

$$\omega^* = \frac{b - a}{e^{\hat{\ell}(x)}}. \quad (4.31)$$

So for a user-specified proportion of grid augmentation ω , the x_i^* follow a distribution of the form

$$g(x) = (1 - \omega^*)f(x) + \omega^*u(x). \quad (4.32)$$

Now the ed estimate, $\hat{g}(x)$, is calculated using loess. While $\log g(x) \geq \ell(x)$ should always hold, loess may give estimates at certain points where this inequality does not hold. At any such point, x , simply set $\log \hat{g}(x) = \hat{\ell}(x)$. Now, since $u(x)$ is known, the estimate for f can be calculated

$$\hat{f}(x) = \frac{\hat{g}(x) - \omega^* u(x)}{1 - \omega^*}. \quad (4.33)$$

Figure 5.121 shows a plot of a grid augmentation fit to the mixture of normals with $\kappa = 10$, $\lambda = 3$, $\alpha = 0.16$, and $\omega = 0.4$ (resulting in $\omega^* = 0.38$). The first panel shows the raw estimates with $\log \hat{g}$, and the second panel shows \hat{f} . The last panel shows the residuals on the log scale for the mixture.

Note that when using grid augmentation, the numeric integral has, from experience, consistently come up slightly less than 1. When using grid augmentation, normalizing is recommended.

4.8 More Examples

Before concluding this chapter, a few more examples are presented for a greater appreciation of the method. These examples particularly highlight the power of simply visualizing the raw estimates in gaining insight into the properties of a density, even without fitting the raw estimates to obtain a density estimate.

The first set of examples comes another part of the study on VoIP quality of service (Xi et al., 2009). One distribution of interest was the distribution of the duration of attempted and connected phone calls. Figure 5.122 plots raw density estimates for the duration on the log base 2 seconds scale for 33,361 attempted calls and 44,689 connected calls. There are some very interesting features in the raw estimate plots, specifically large spikes in the attempted calls distribution. Figures 5.123 – 5.125 show kernel density estimates on the log scale with the same plotting scales as that of figure 5.122 using Silverman’s rule, unbiased, and biased cross-validation. These KDEs come nowhere near showing the same amount of information that can be seen in the raw estimates.

Another example from the VoIP project is a study of the distribution of the bit-rate of the calls by caller and callee and duration. Of interest is how far into the call the caller and callee’s bit-rates to reach a similar distribution. Figure 5.126 plots ed raw density estimates for caller and callee bit-rates by call duration intervals. Early on in the call, the caller’s bit-rate is typically much smaller than that of the callee. There are some interesting patterns that occur in the 60–80 second range that would require domain knowledge to understand. The distributions of caller and callee bit-rate seem to reach a similar distribution after about 100 seconds. The plotted raw density estimates are very helpful in visualizing the patterns in the densities.

A final set of examples is from a batch of “benchmark densities” put forth and studied by Devroye and Berlinet (1994), mainly for the purpose of showing several types of patterns that can arise in a density and how the ed raw estimates can be helpful in exposing these patterns. Figure 5.127 shows a plot of the 28 densities. Figures 5.128 – 5.130 show the resulting ed raw estimates from samples of size 5,000 from each density with the true log density superposed. The ed raw estimates do a very good job of giving the analyst an idea of the patterns in a density and provide a simple approach to obtaining the density estimate by regression methods.

4.9 Computational Methods

There are two major computational steps in obtaining the ed density estimate. The first is the computation of the raw estimates, and the second is smoothing the raw estimates. Each is discussed below.

The main computational chore in calculating the raw estimates is obtaining the order statistics. A simplistic way to do this is to sort the data and extract the order statistics as needed. This is the current approach and the quicksort algorithm is used, which on average takes $O(m \log m)$ time although its worst case is $O(m^2)$. Since only every κ th order statistic is needed, however, computation time is wasted sorting the values in between, and a better approach might be to recursively apply a

selection algorithm. There are selection algorithms that operate in worst-case linear time (Blum et al., 1973).

After the raw estimates are computed, there are n values to smooth, where $n = \lfloor (m - 1)/\kappa \rfloor$. Loess is a computationally efficient method, where direct evaluation of the local fit is evaluated at a fixed and sufficiently dense set of points from a k -d tree, and interpolated elsewhere. Typically, a user would like to compute the final density estimate at a dense grid of equally-spaced points, enabling plotting of the density or numerical integration for normalization. Once k -d tree has been constructed, interpolation on the grid is performed in linear time. Overall runtime of loess is proportional to n (Cleveland and Grosse, 1991).

Another computational requirement in loess smoothing is estimating the standard error of the estimates, as discussed in section 4.5.4. Direct calculation of the standard error estimate and corresponding degrees of freedom is computationally burdensome. In the loess implementation, there is a clever approach to estimating these quantities efficiently. In this application, the theoretical variance of the residuals is known, so that this calculation is unnecessary.

4.10 Discussion

Similar work and ideas include Marron and Uдина (1999), where the idea is briefly mentioned of using residuals to indicate when performance of the density estimate is poor, by binning the data to a fine grid and using the difference between the kernel estimate and the bin frequencies as residuals. Also, Farnen and Marron (1999) discuss binning the data and then fitting locally using a smoothing method that works in the presence of heteroscedastic noise. Cleveland et al. (1993) briefly discuss a notion similar to ed.

There are some existing visualization tools for density estimation. SiZer (Chaudhuri and Marron, 1999) addresses the question of what features of a density are significant, examining a family of different fixed bandwidth kernel density estimates.

It provides analysts with a visual tool to quantitatively decide which features are really there. Mode trees (Minnotte and Scott, 1993) is a similar tool. These tools are very useful for exploring the features of a density, but are not helpful in obtaining a final density estimate, as they only consider families of fixed-bandwidth estimates, and most densities cannot be accurately estimated using fixed bandwidths.

Ed shows promise as a useful tool for exploring densities with moderate or large samples. Simply viewing the raw estimates alone can be extremely insightful. The procedure is very straightforward, and problems such as discontinuities, boundary cut-offs, and mis-fitting peaks and valleys can be detected and dealt with very easily all within the same framework. In the traditional kernel approach, on the other hand, one would need to resort to boundary kernels, variable bandwidths, higher-order kernels, or other means to deal with these problems. The low computational complexity of ed is also very attractive from a practical standpoint.

Future work includes extending ed to higher dimensions, more work on grid augmentation, and determining more rigorously the theoretical properties of the bias and variance. Another possible research direction would be work on automating parameter selection for ed. Although the method has been presented here as a modeling tool, which implies an iterative human-guided approach, the favorable bias and variance properties along with the sufficiently fast computation time may make this method a useful tool in machine learning algorithms.

5. FIGURES

Visualization is very useful in statistical analysis and exposition and as such, there are many figures presented in this work. All figures are collected here as a separate chapter instead of distributing them throughout the document. The reader may set this chapter apart to be read in parallel with the text.

5.1 Figures for Local Regression Modeling

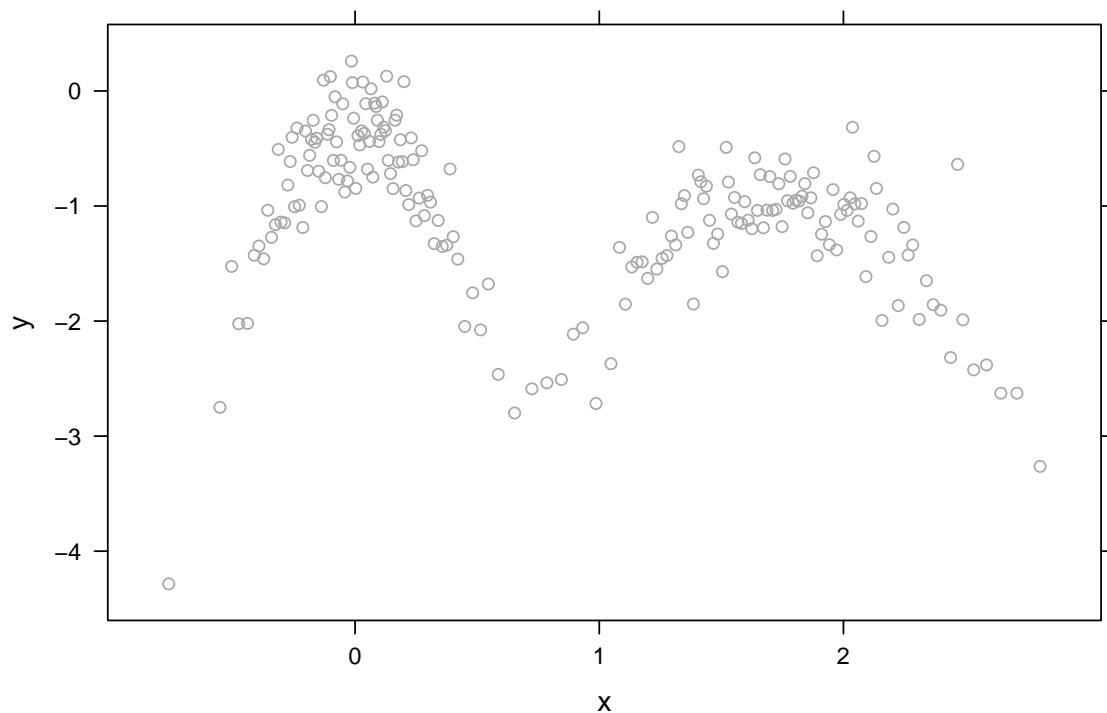


Figure 5.1. Data for loess example. There are 199 points. This data is from a nonparametric density estimation problem (see chapter 4).

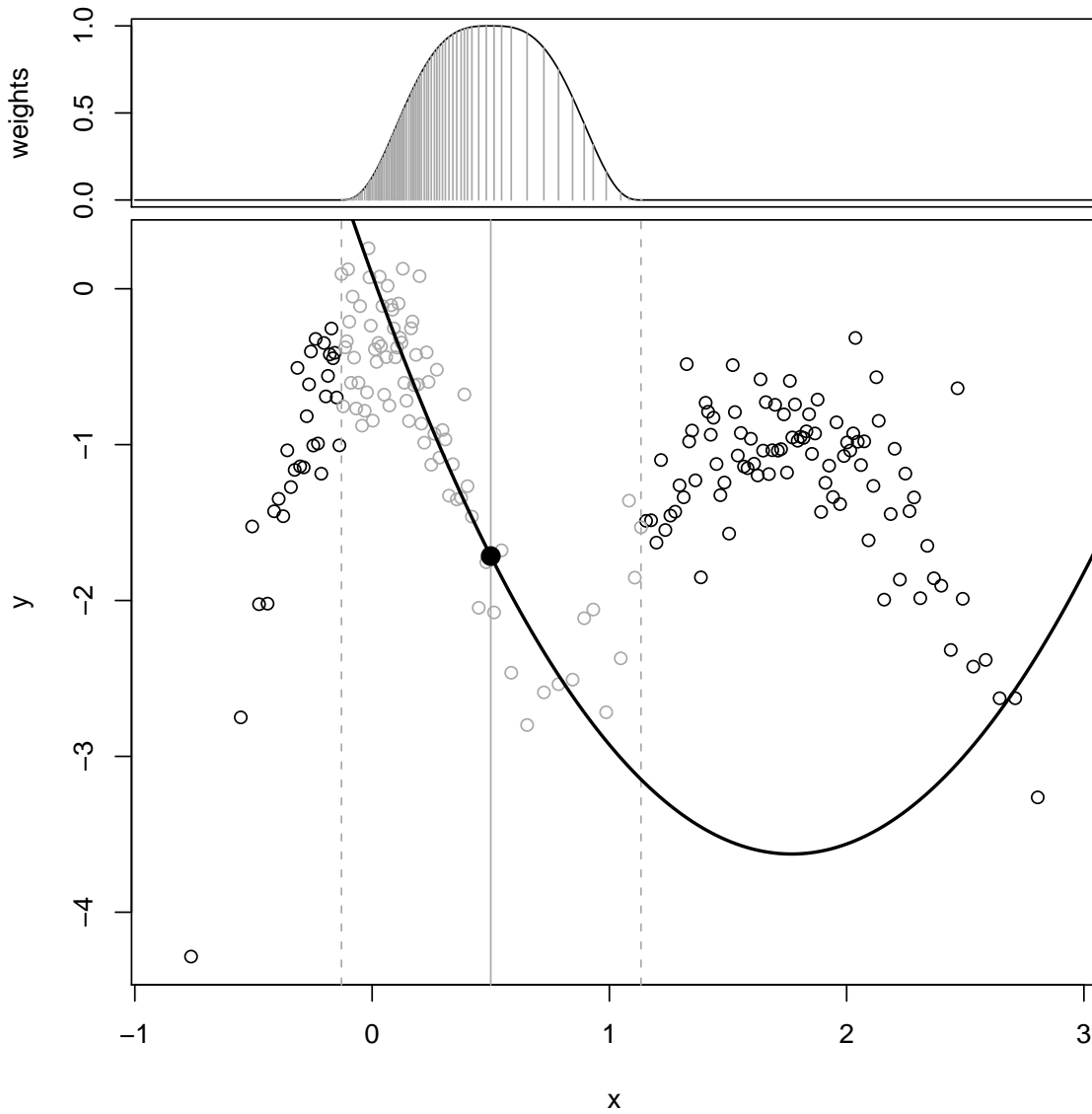


Figure 5.2. Loess fit at $x = 0.5$, with $n = 199$, $\lambda = 2$, and $\alpha = 0.4$. The gray points indicate the 79 points that fall within the neighborhood around $x = 0.5$. The weights corresponding to each neighboring x are shown in the top panel. The solid black line is the fitted weighted least-squares quadratic polynomial and the solid dot is the loess fit at $x = 0.5$.

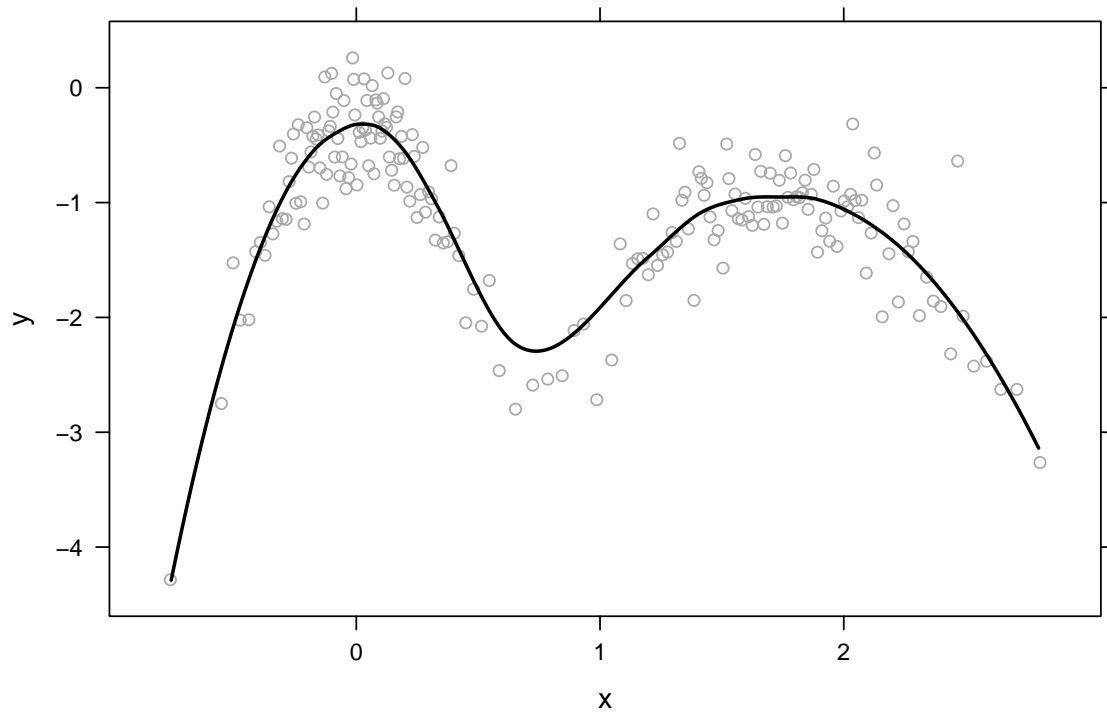


Figure 5.3. Loess fit for the density example data with $\lambda = 2$, $\alpha = 0.4$.

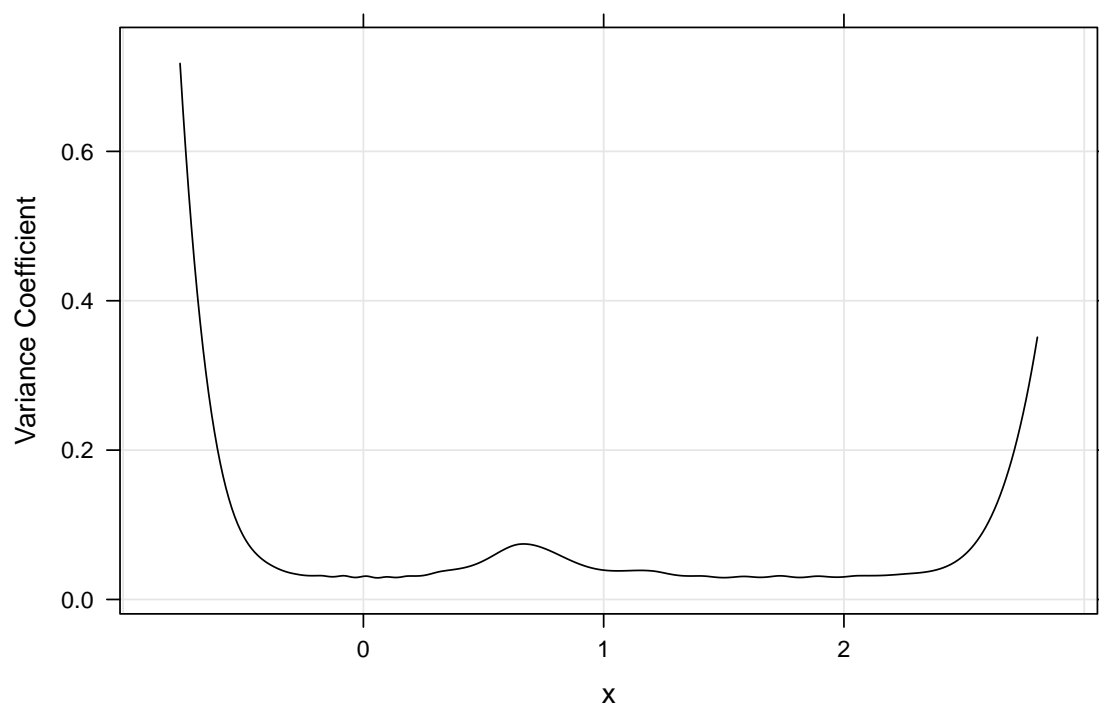


Figure 5.4. Variance coefficient along the design space for the density loess fit shown in figure 5.3.

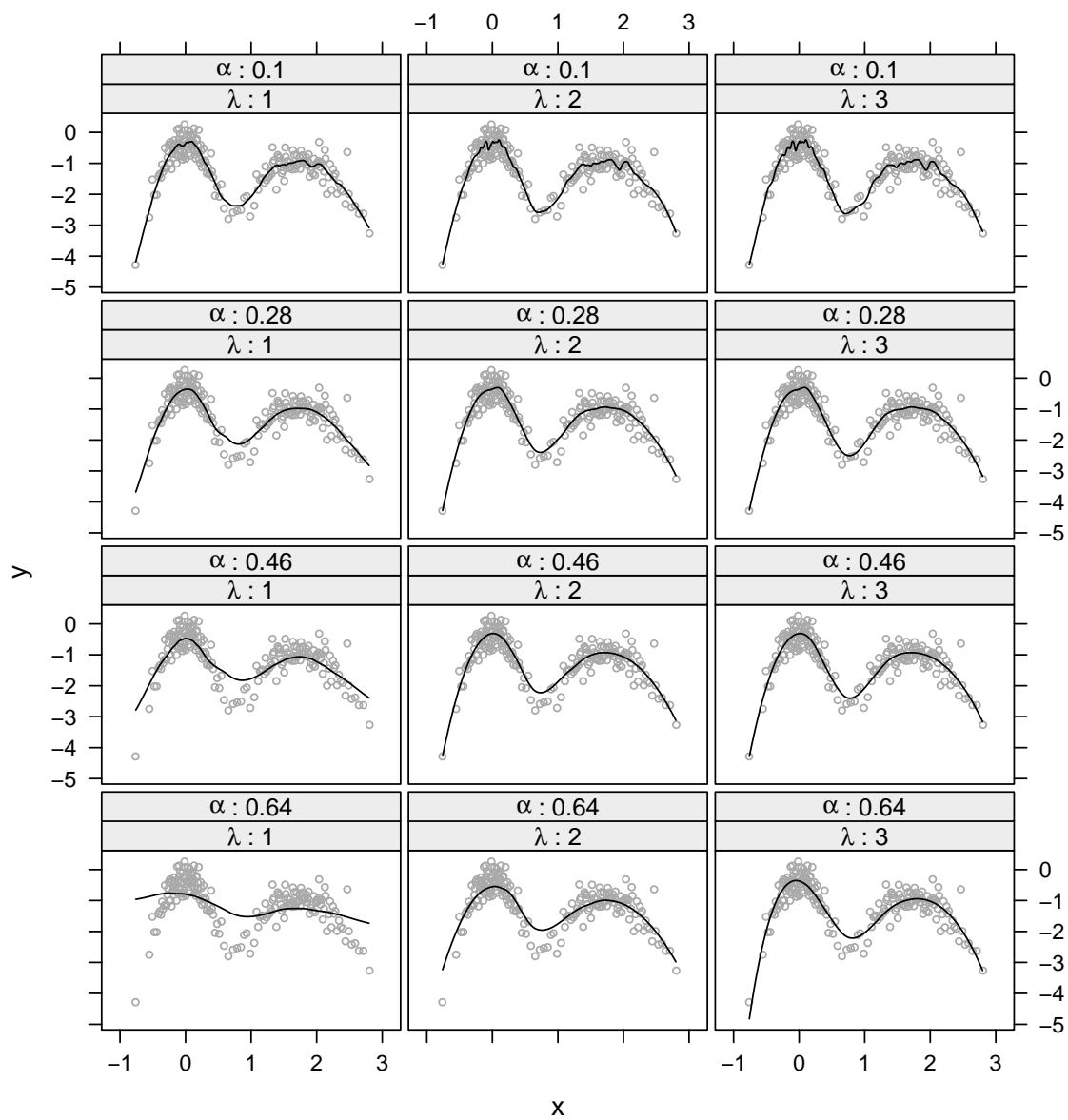


Figure 5.5. Loess fitted values for several values of α and λ .

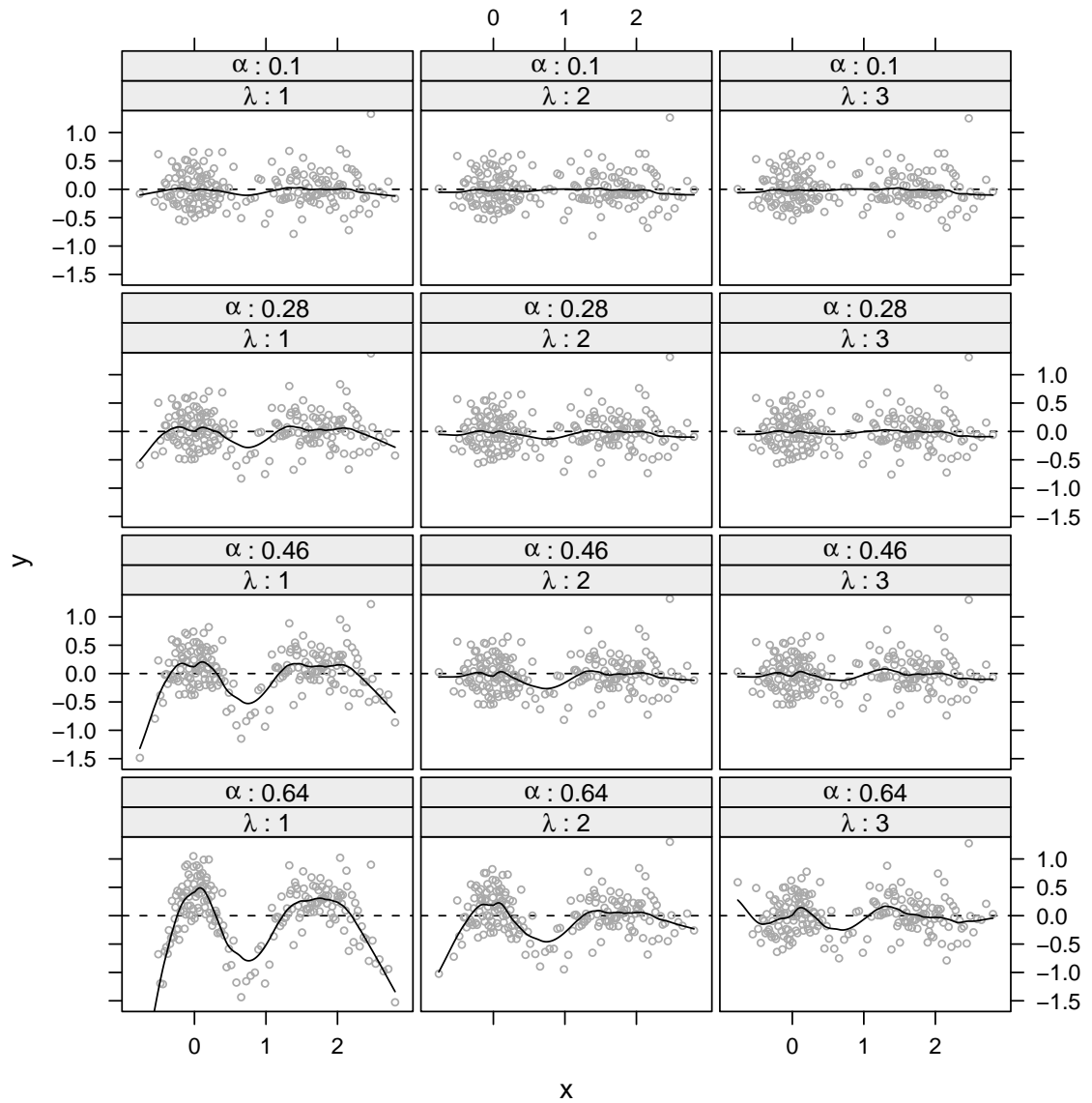


Figure 5.6. Loess residuals for several parameters. The solid line is a loess smooth of the residuals to highlight the behavior of the residual mean across the design space.

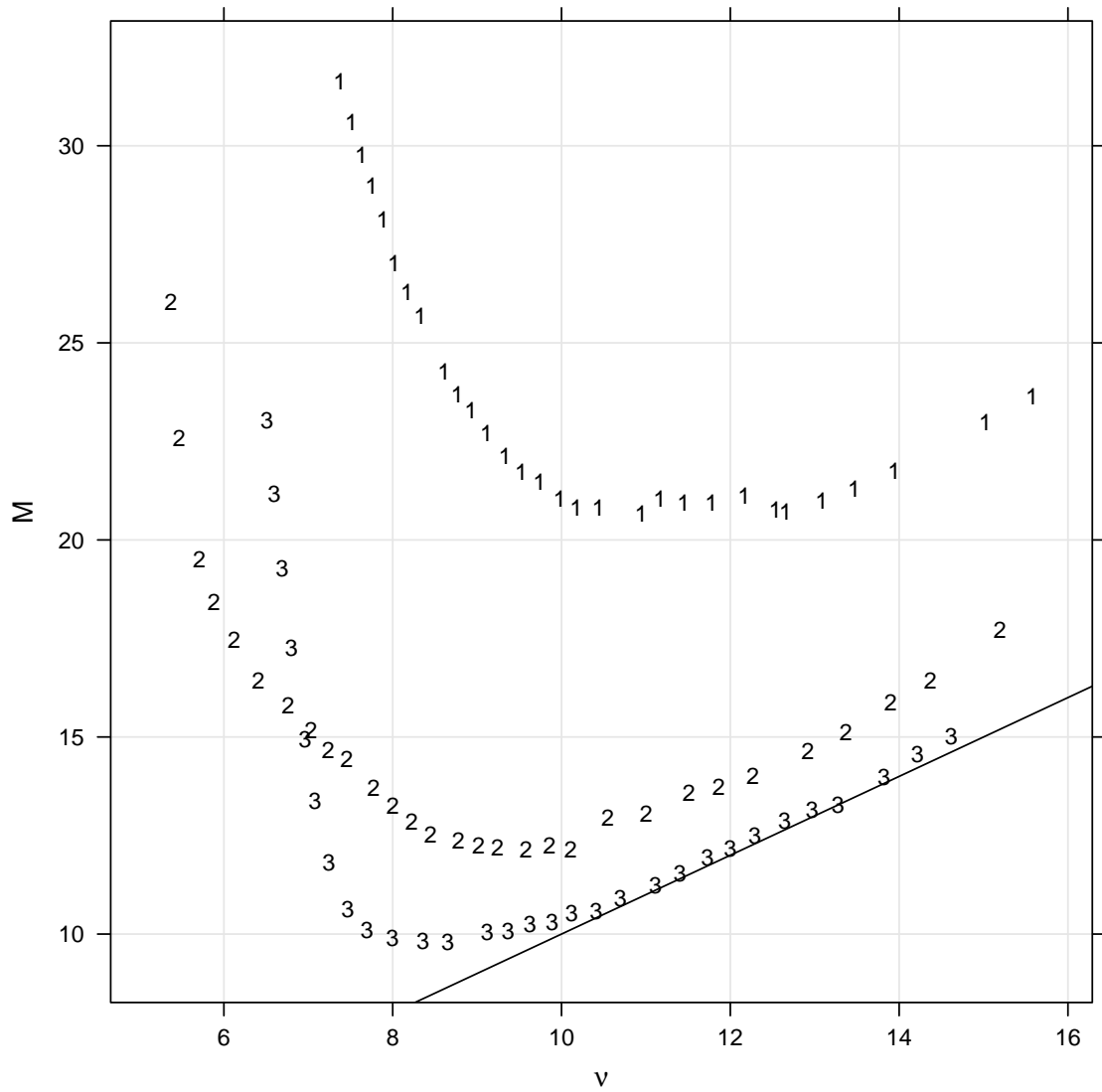


Figure 5.7. Mallows' C_p plot for the density data. The scaled MSE, M is plotted against the equivalent number of parameters, ν .

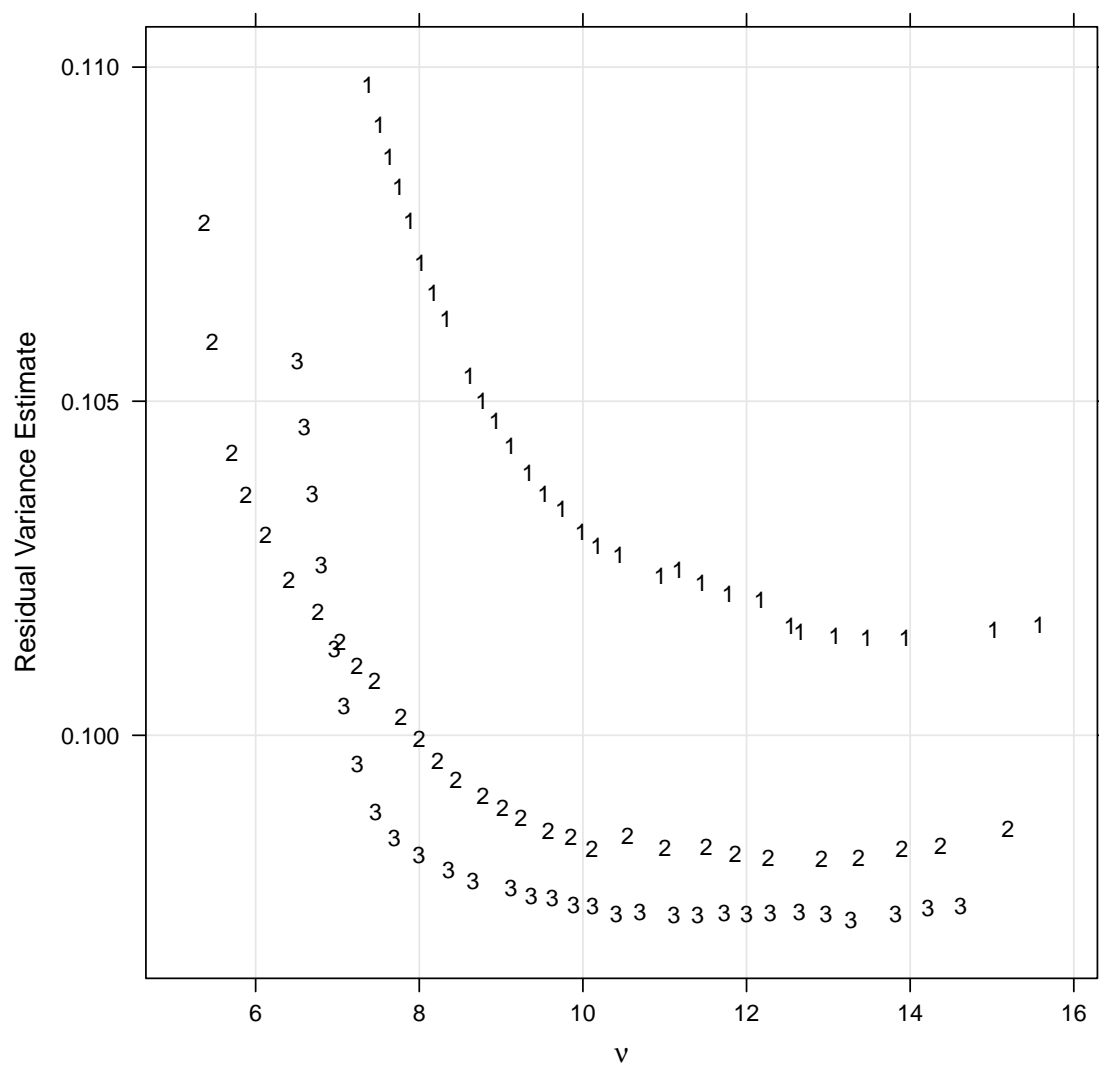


Figure 5.8. Residual variance vs. ν for density data fits. The variance begins to stabilize for $\lambda = 3$ around $\nu = 8.5$.

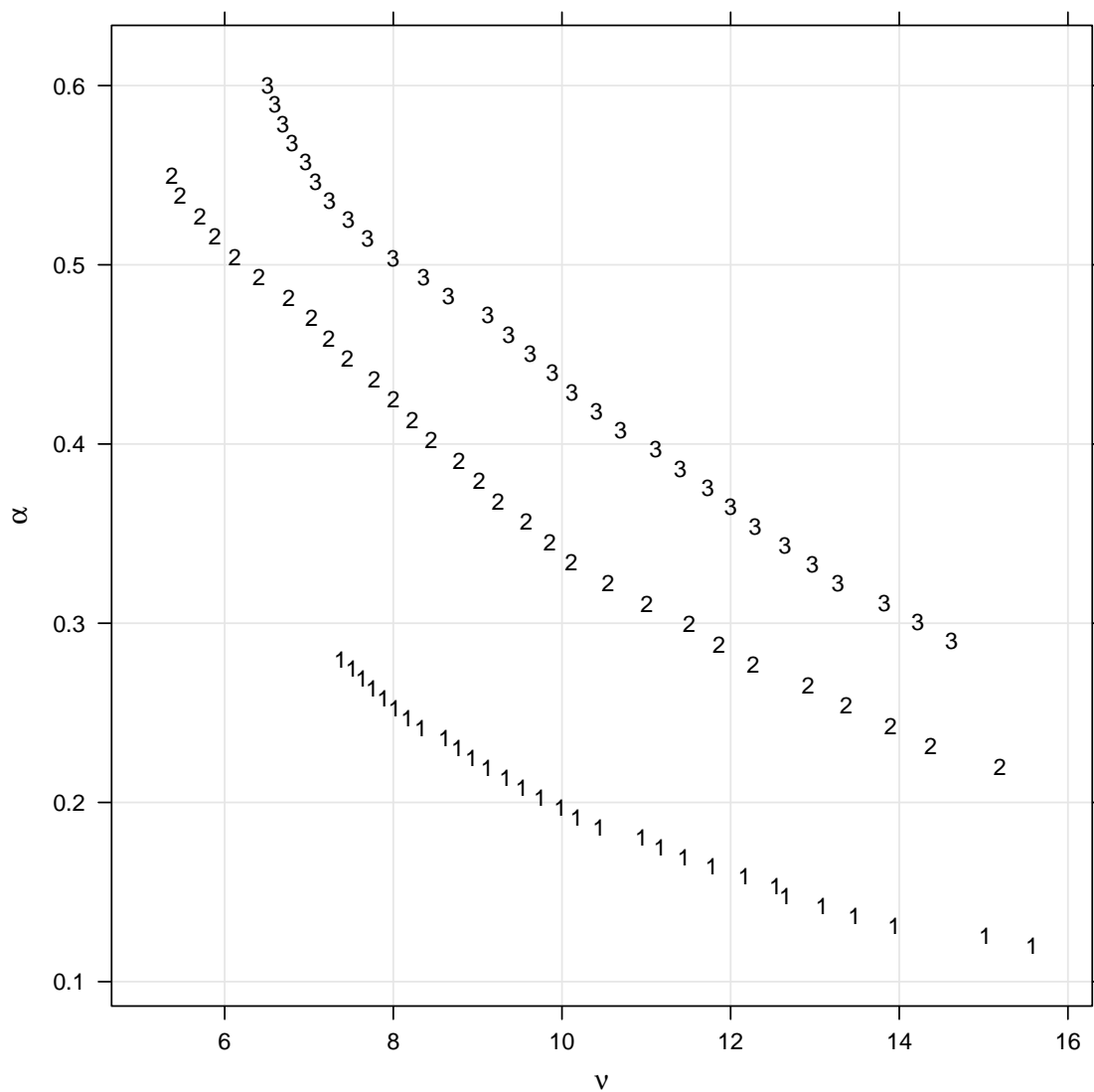


Figure 5.9. Smoothing parameter α vs. equivalent number of parameters ν for density data. This helps translate ν into α when looking at a C_p plot (figure 5.7).

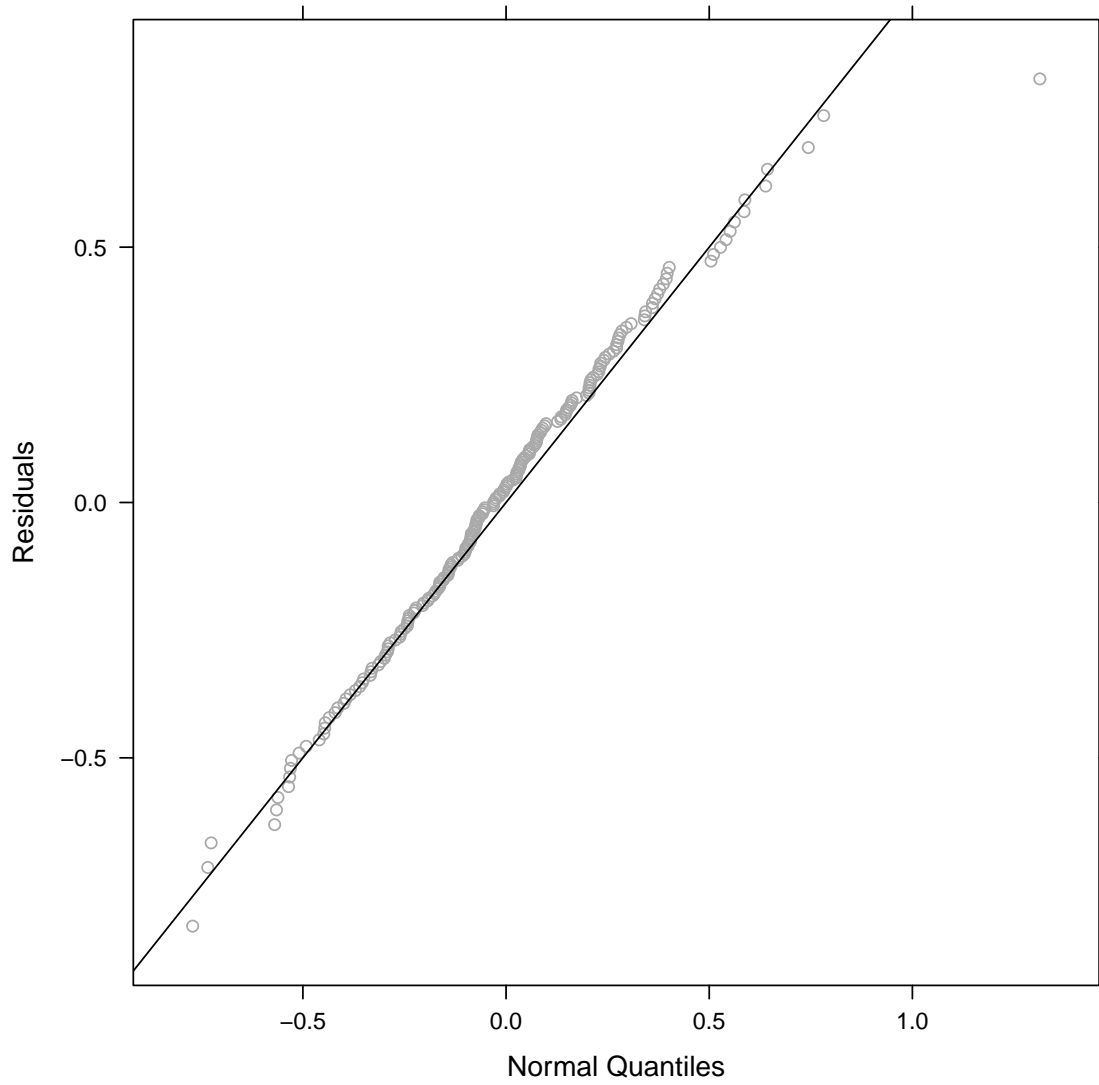


Figure 5.10. Normal quantile plot for the density data fit with $\lambda = 2$, $\alpha = 0.4$.

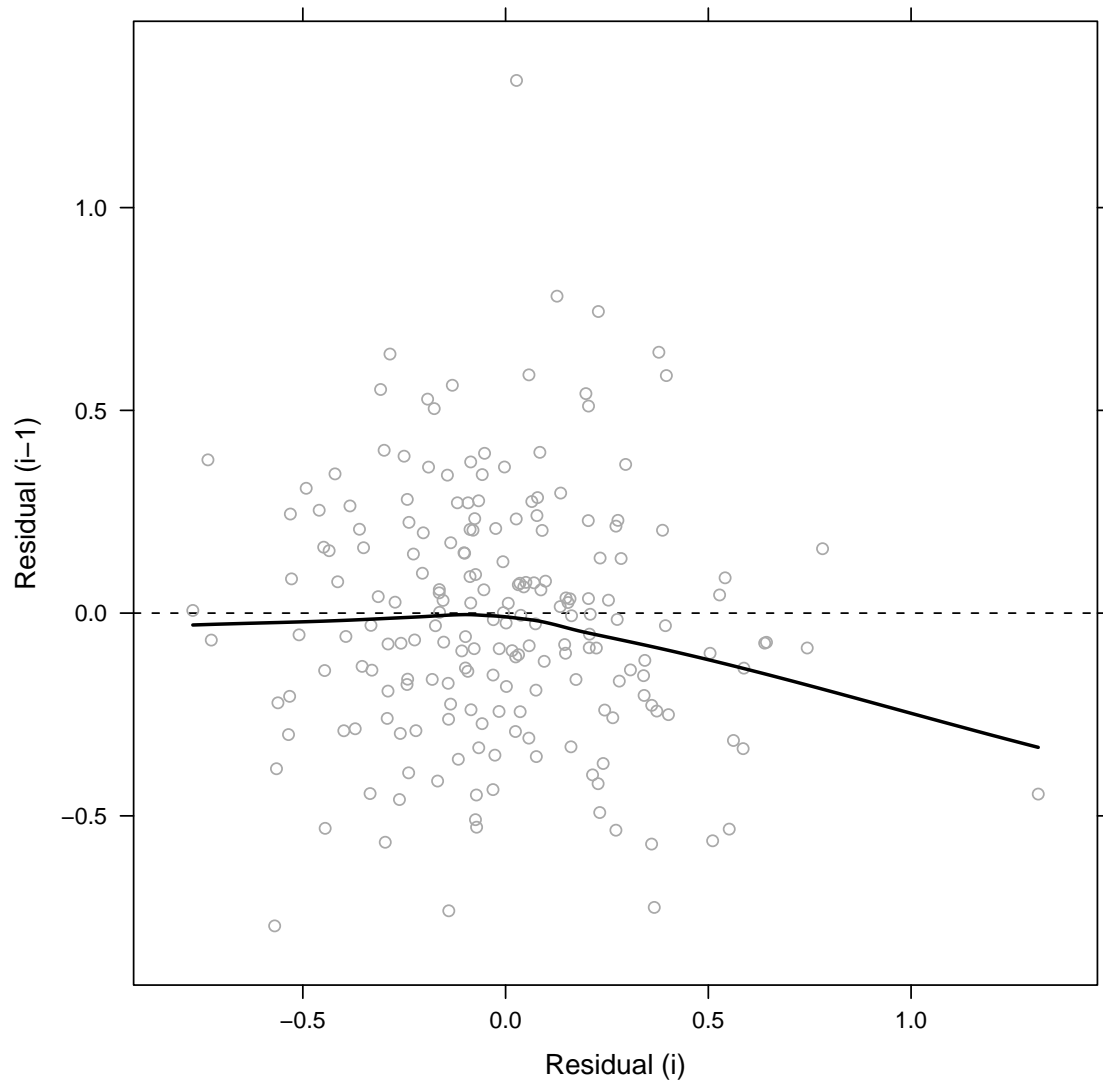


Figure 5.11. Residual serial correlation plot for density data loess fit with $\lambda = 2$, $\alpha = 0.4$.

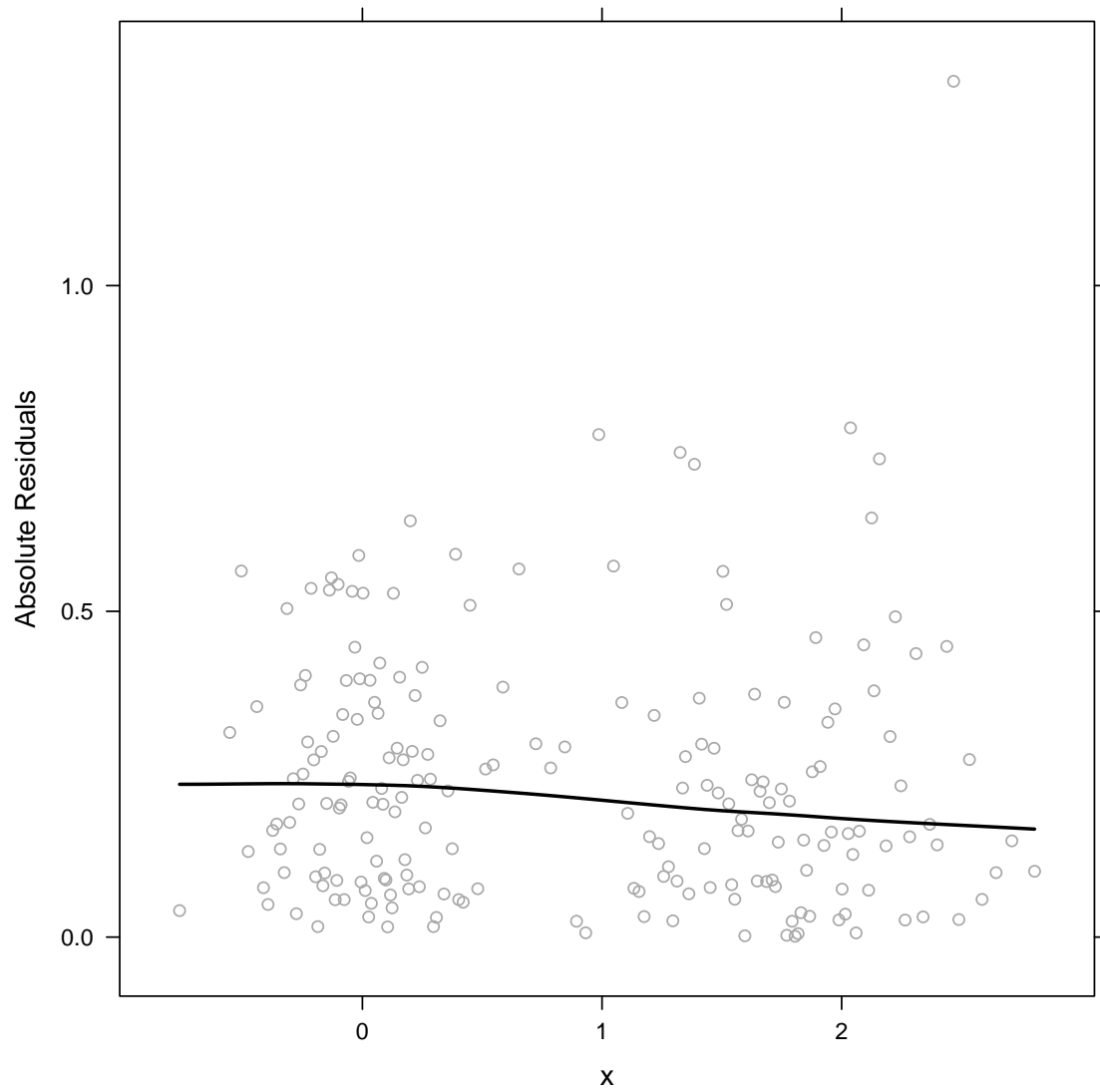


Figure 5.12. Residuals vs. design points for density data loess fit with $\lambda = 2$, $\alpha = 0.4$. The solid line is a loess fit to highlight any signs of changing variance across the design space.

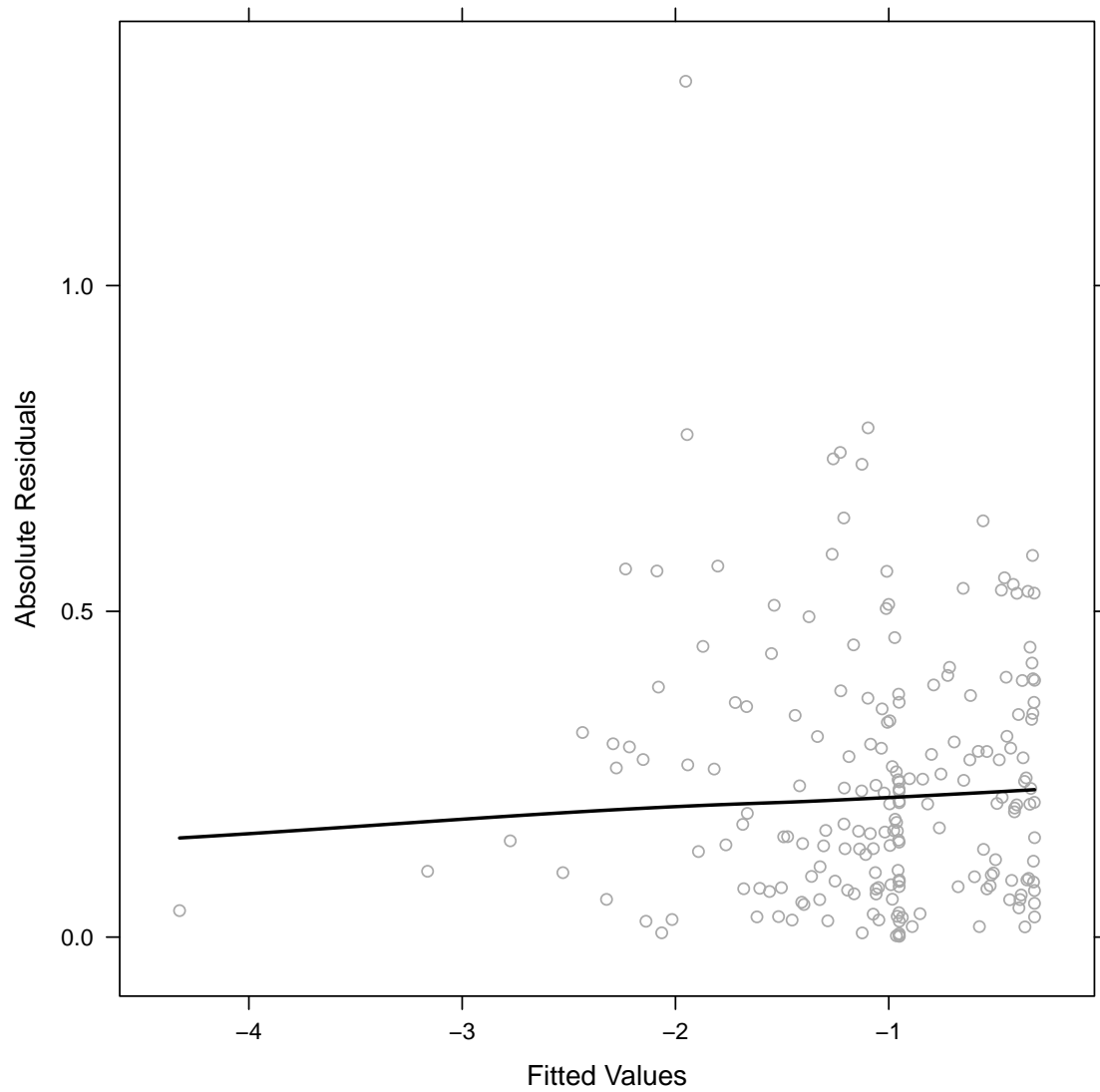


Figure 5.13. Residuals vs. fitted values for density data loess fit with $\lambda = 2$, $\alpha = 0.4$. The solid line is a loess fit to highlight any dependence variance on fitted values.

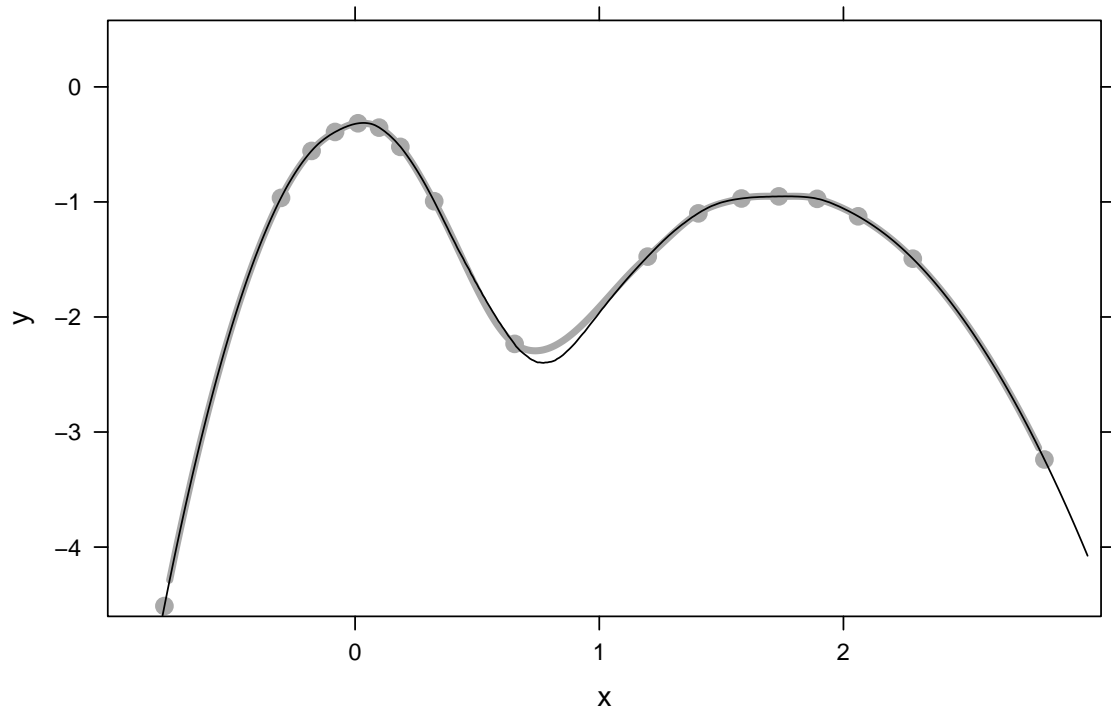


Figure 5.14. Loess fit for density data with exact fits at k -d tree vertices and interpolation elsewhere. The solid gray dots are the fits at the vertices, the thick gray line is the interpolated fit, and the black line is the exact loess fit.

5.2 Figures for Time Series Modeling

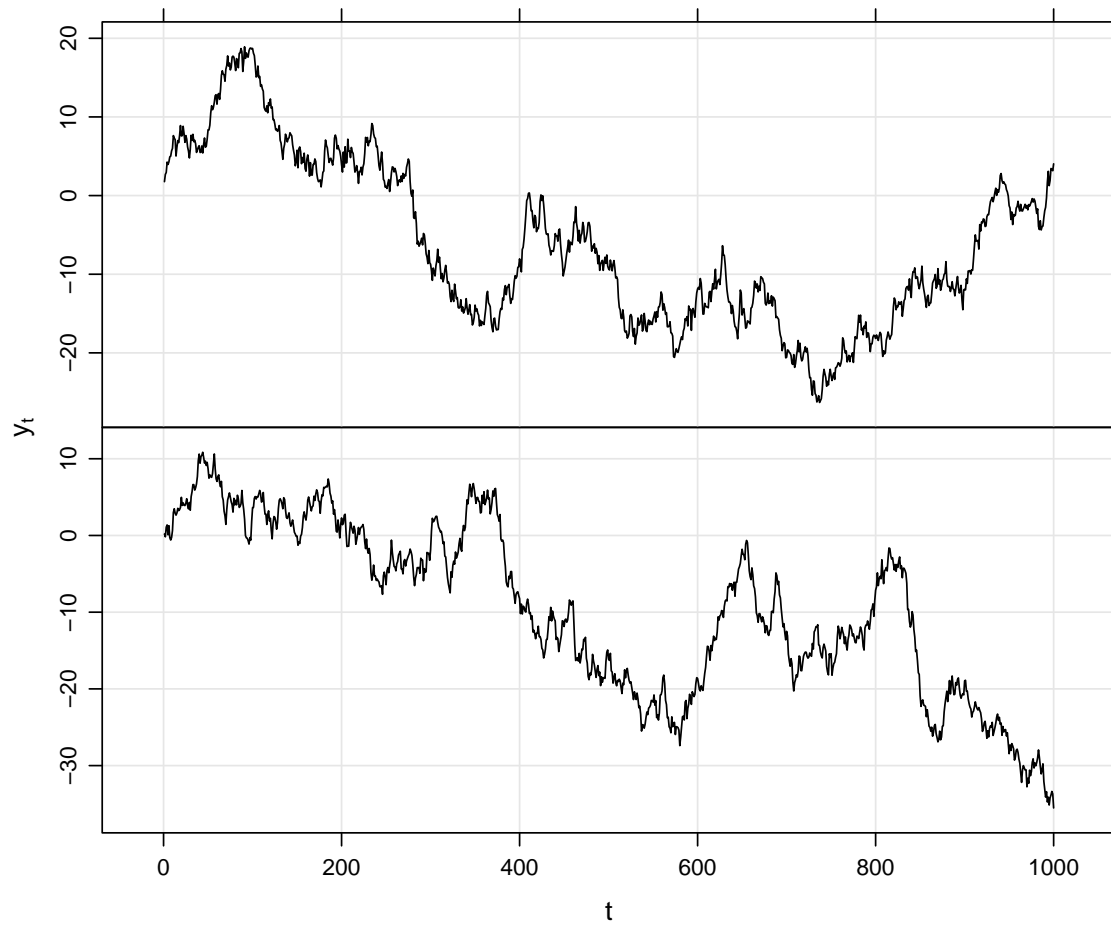


Figure 5.15. Two realizations of a unit-variance random walk process.

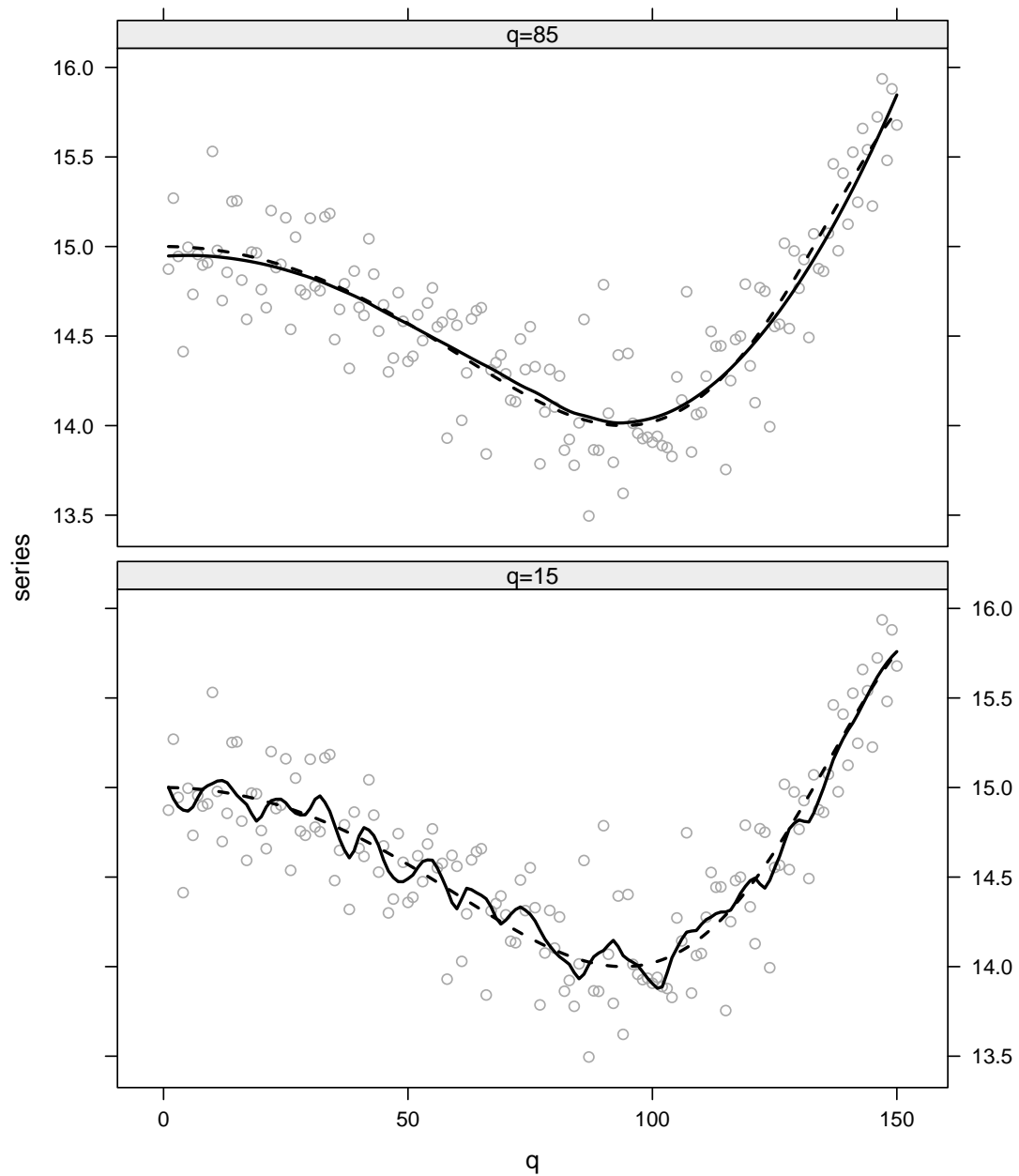


Figure 5.16. Time series loess fit to simulated data with a deterministic trend and an ARMA(1,1) error process. The true trend is shown by the dashed line and fits are shown for $q = 85$ (top) and $q = 15$ (bottom) as the solid black line. In both cases, local quadratic fitting is used.

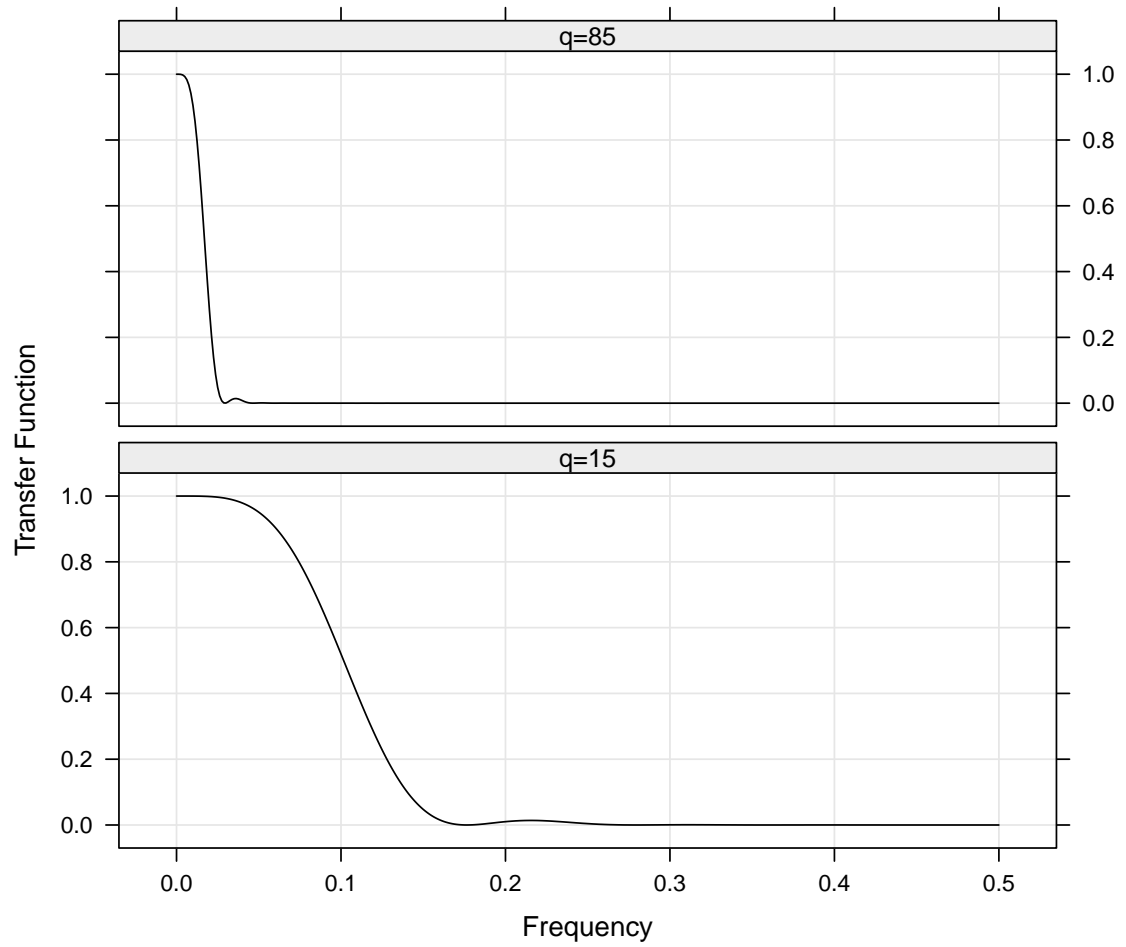


Figure 5.17. Power transfer functions for symmetric loess operators with $\lambda = 2$ and $q = 85$ and $q = 15$.

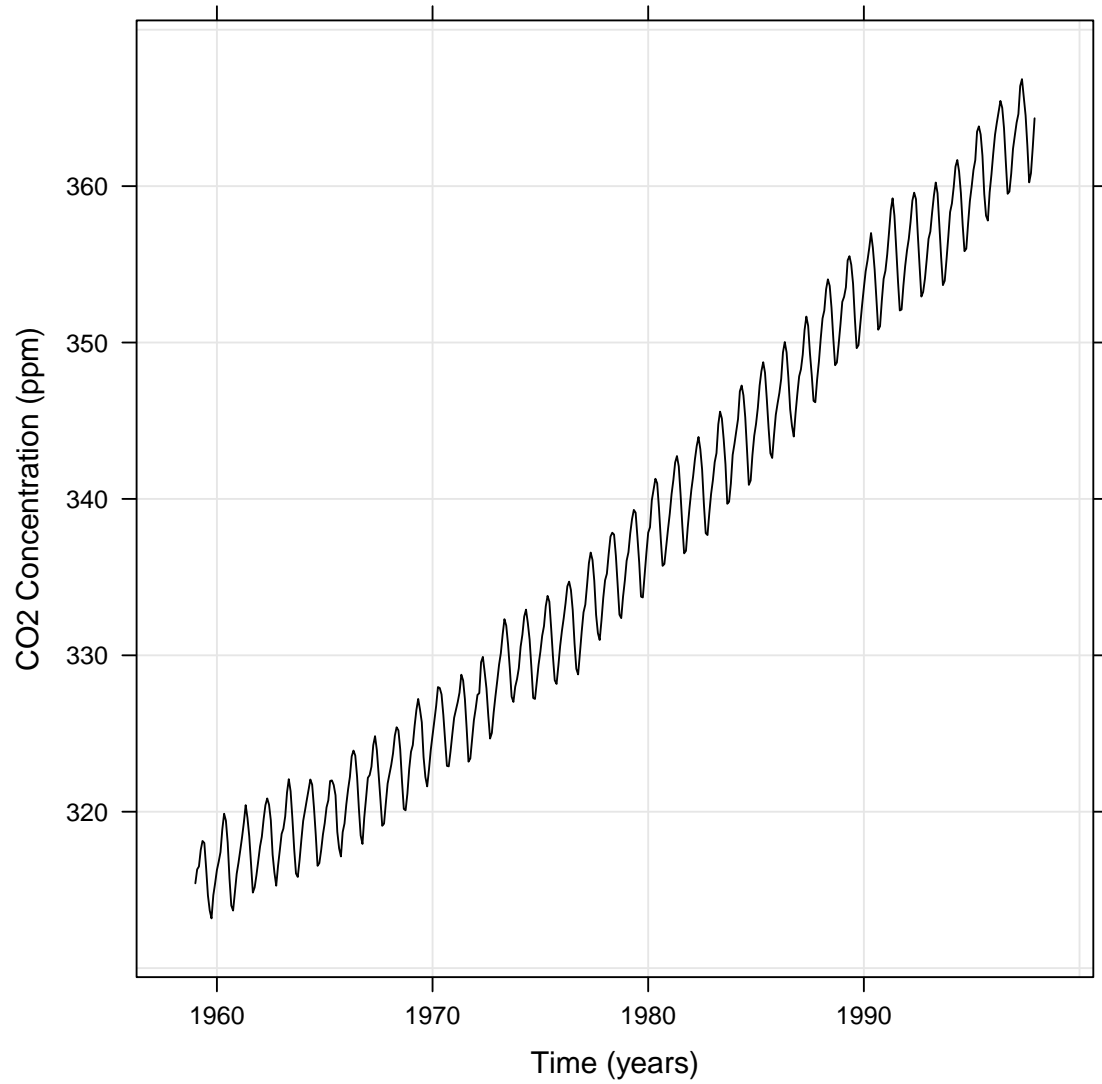


Figure 5.18. CO₂ concentration measurements in parts per million at Mauna Loa, Hawaii from 1959 to 1997, measured monthly.

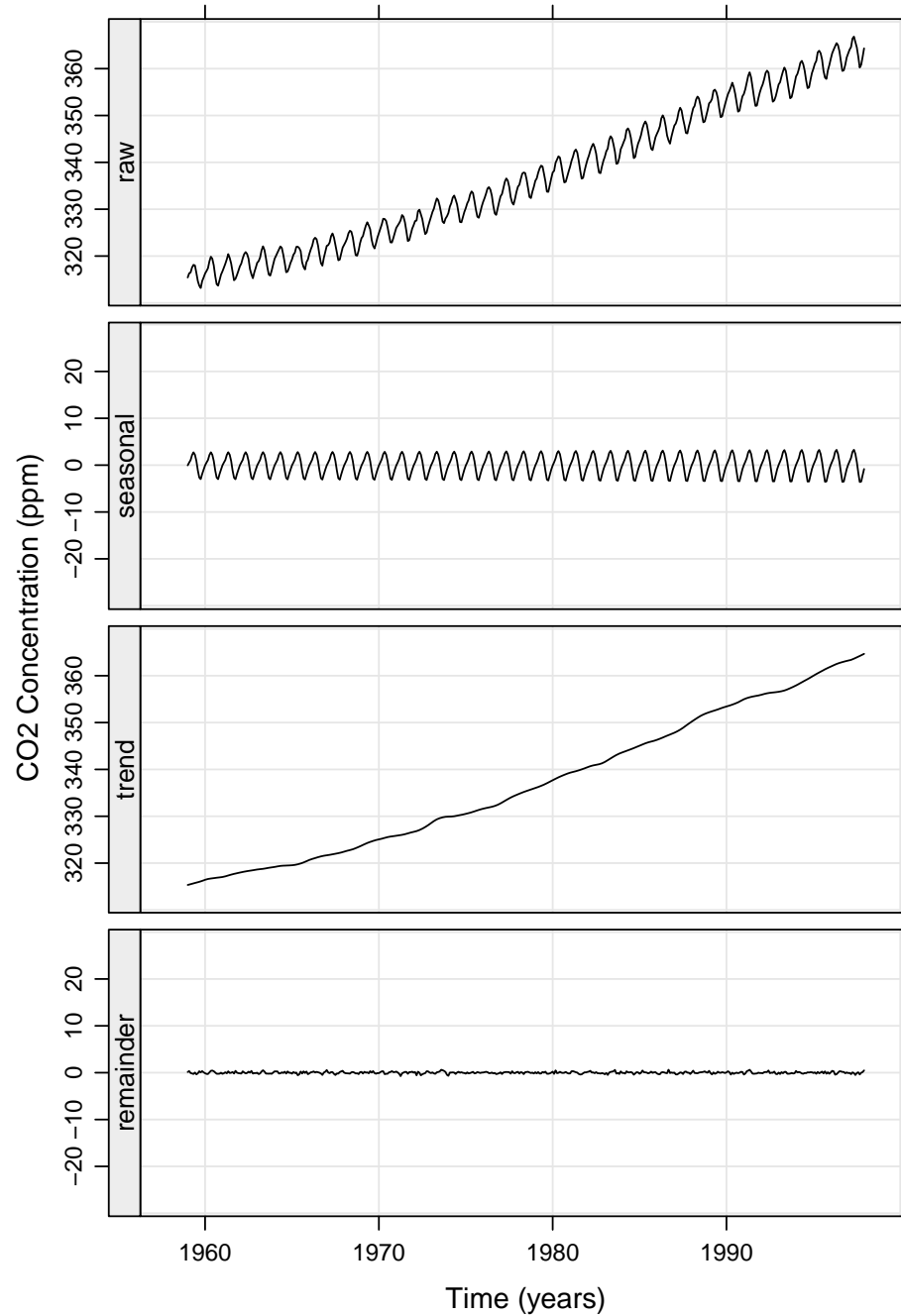


Figure 5.19. STL decomposition for monthly CO₂ measurement time series with seasonal smoothing parameters ($\lambda = 1$, $q = 35$) and trend smoothing parameters ($\lambda = 1$, $q = 19$). All components are plotted on the same scale.

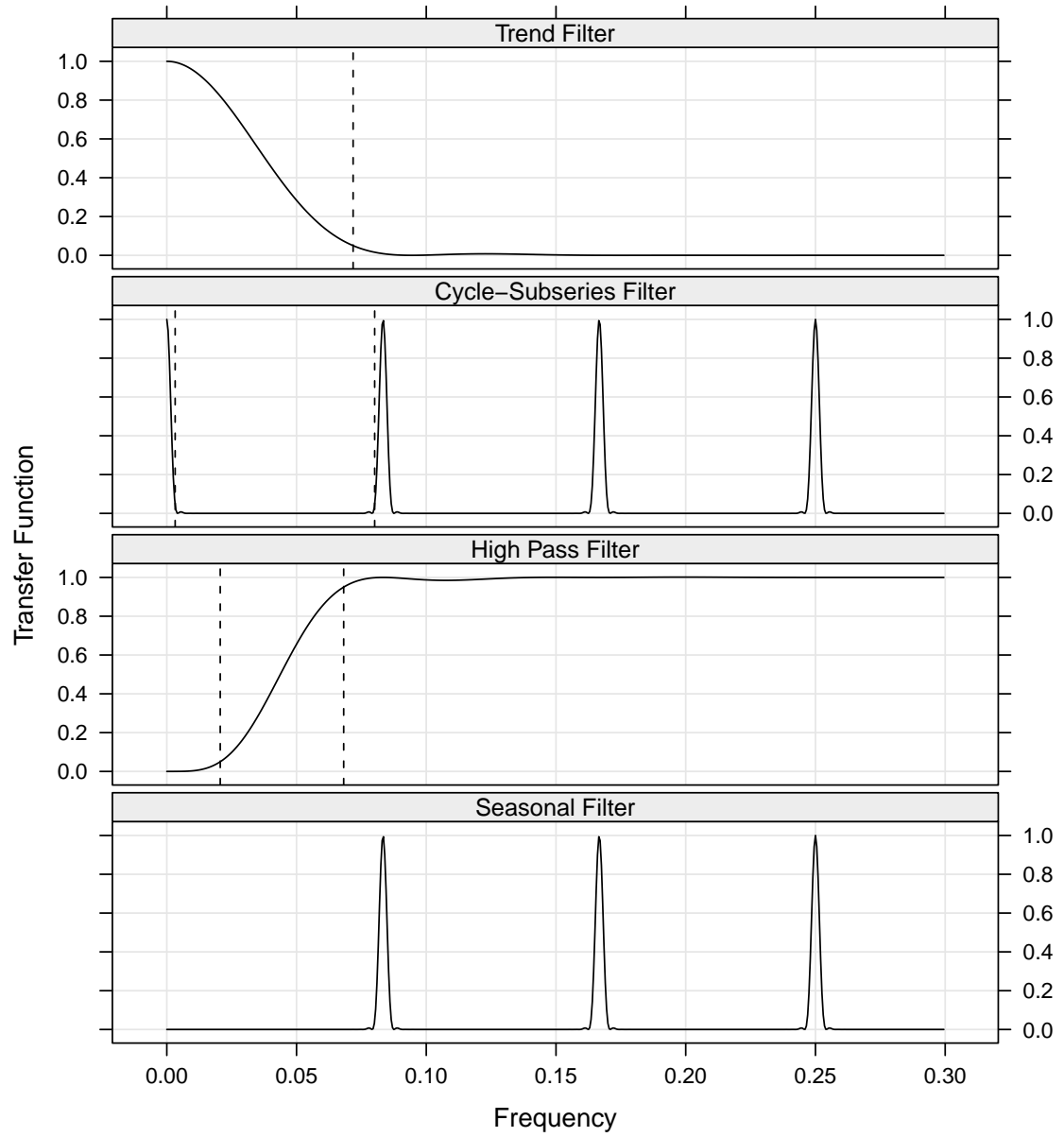


Figure 5.20. Power transfer function for each smoothing component of the CO₂ STL decomposition. The vertical dashed lines correspond to critical frequencies with $\omega = 0.5$.

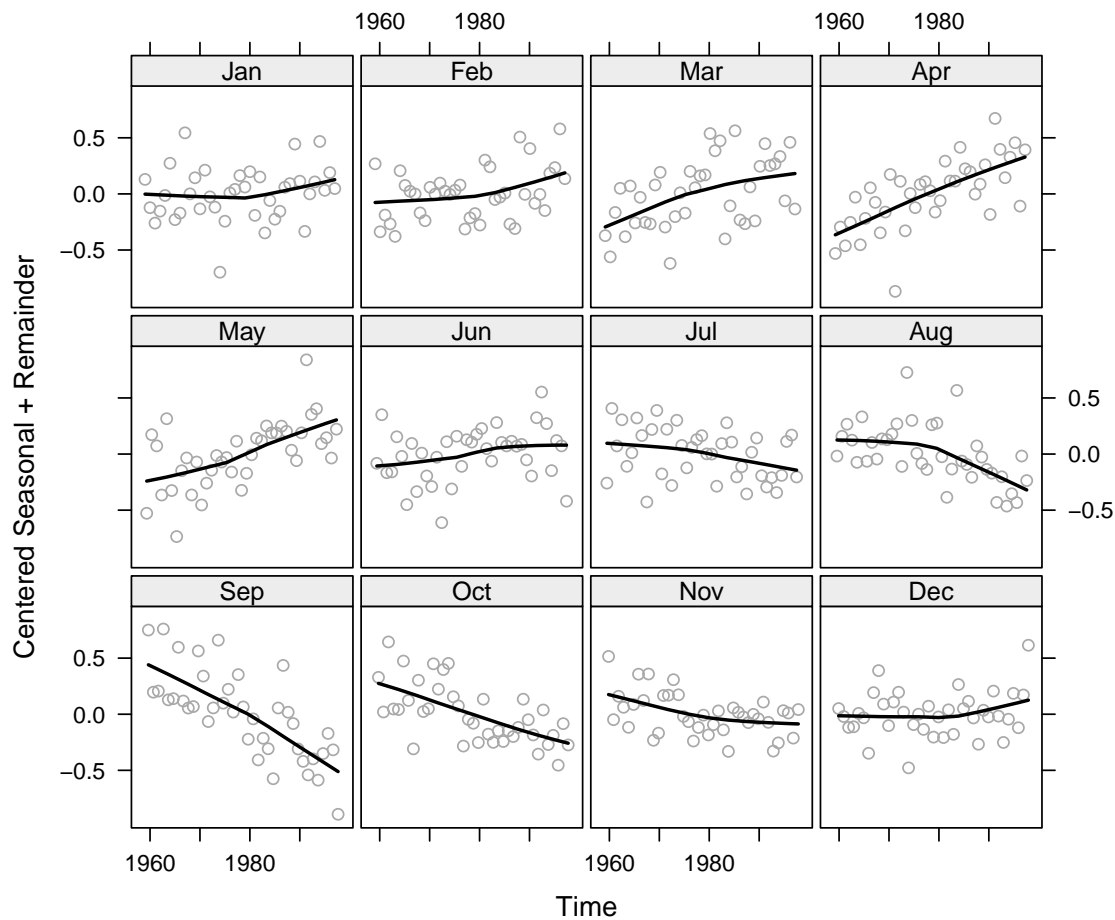


Figure 5.21. Seasonal diagnostic plot for CO_2 decomposition. Each cycle-subseries is centered around zero. The dots correspond to the centered $\hat{s}_t + \hat{r}_t$ and the lines correspond to the centered \hat{s}_t .

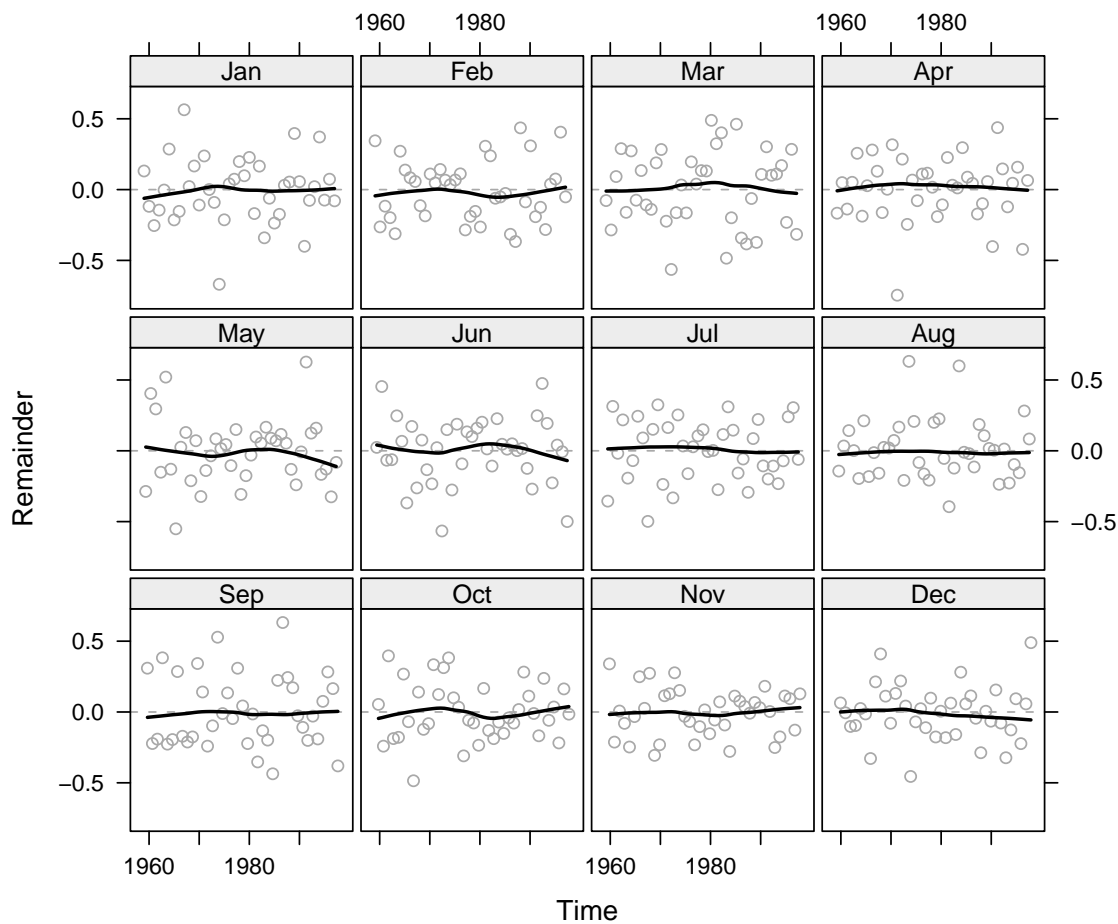


Figure 5.22. Remainder component by cycle-subseries for CO₂ decomposition. The solid line is a loess fit to highlight deviations from zero.

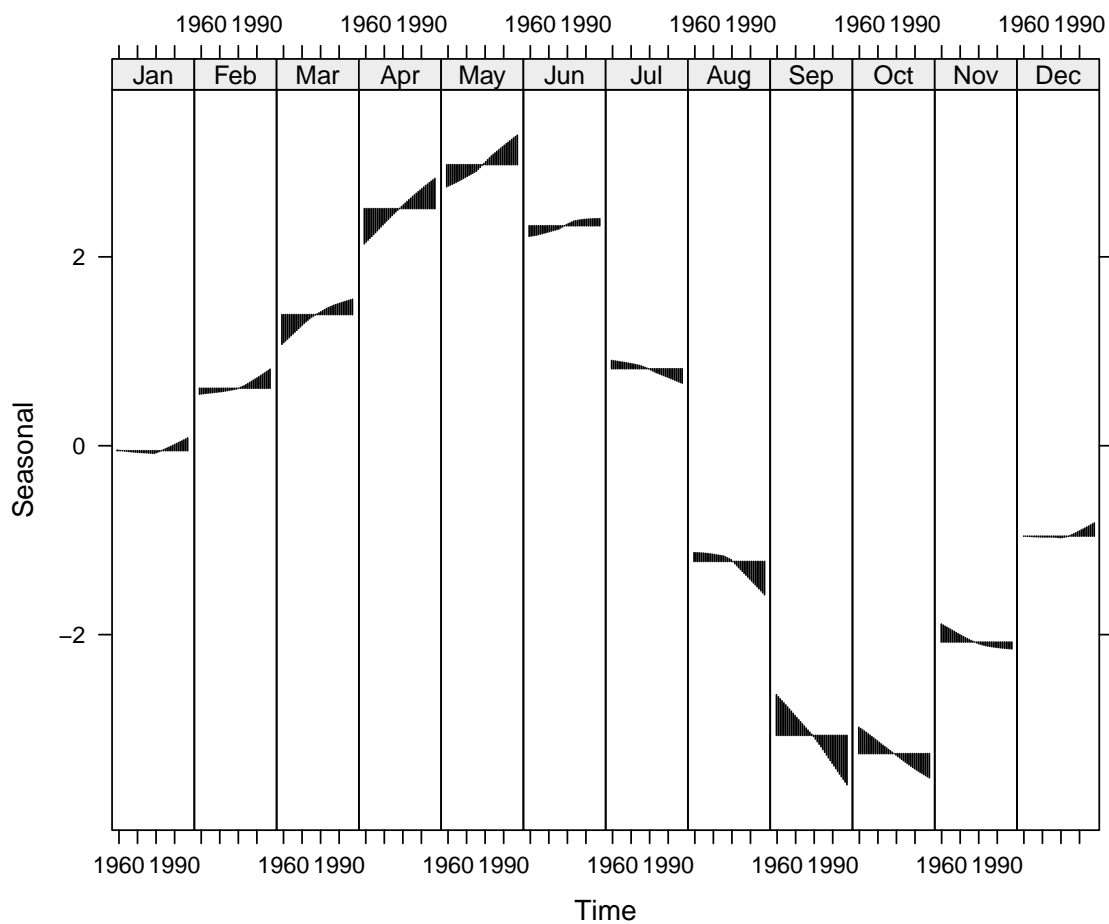


Figure 5.23. Fitted seasonal component by cycle-subseries for CO₂ data, with vertical lines emanating from the cycle-subseries midmean.

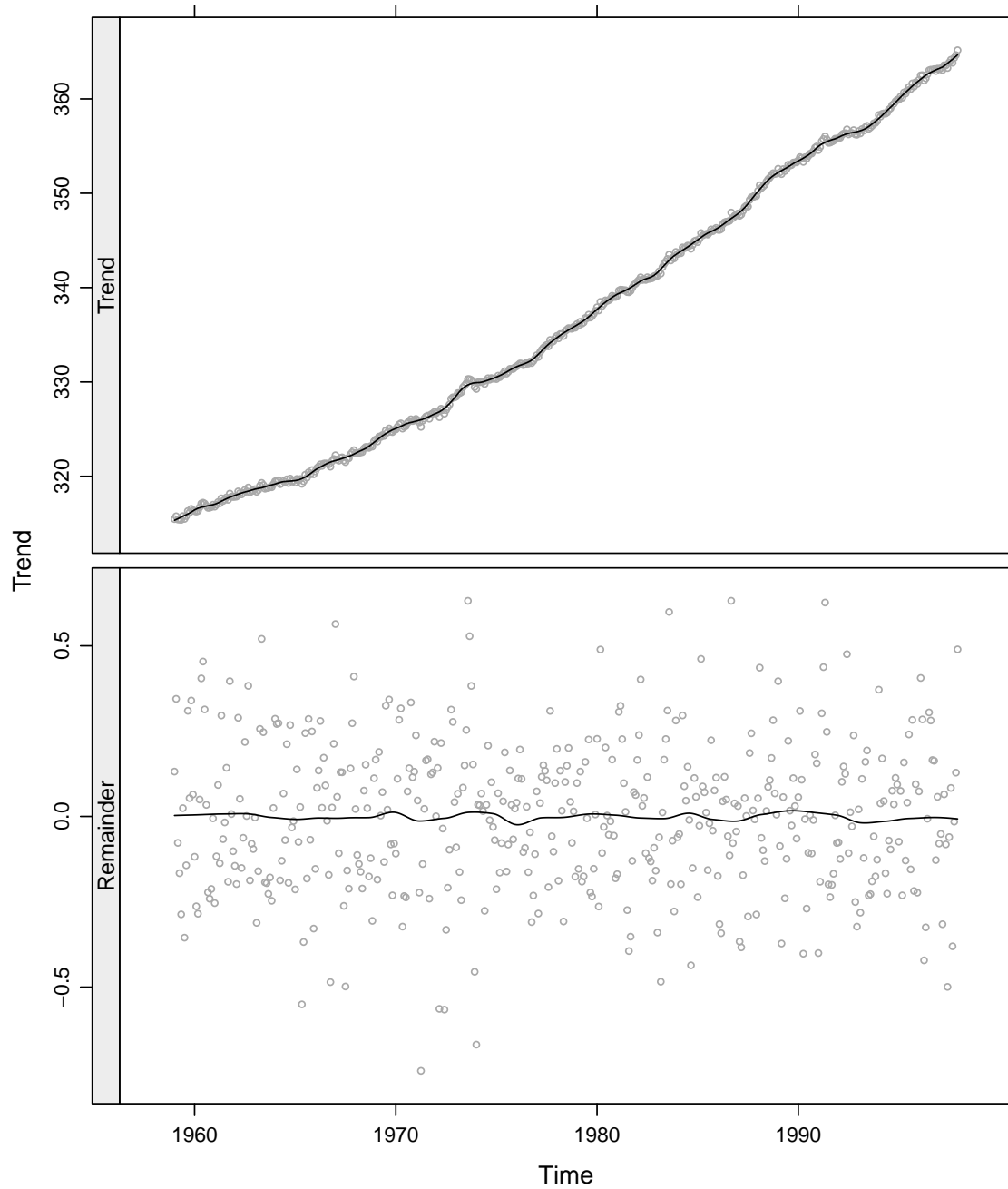


Figure 5.24. Trend diagnostic plot for CO_2 decomposition. The dots in the top panel are $\hat{t} + \hat{r}$ and the line is \hat{t} . The line in the bottom panel is a loess fit to highlight deviations from zero in the residuals.

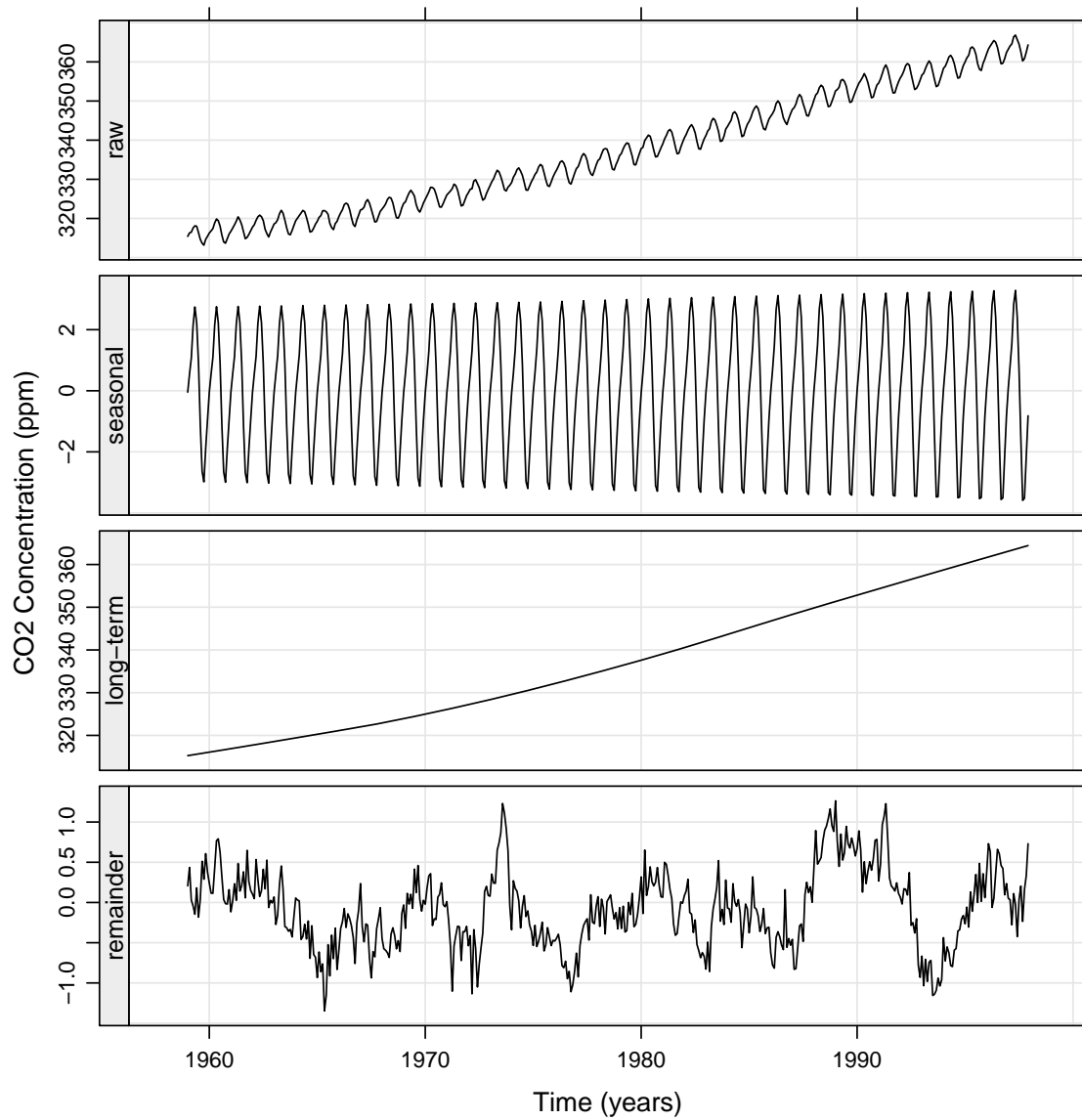


Figure 5.25. CO₂ decomposition with low-frequency post-trend smoothing. The components in this plot are not plotted on the same scale. The long-term trend was computed by applying a loess filter with $\lambda = 1$ and $q = 201$ to $y_t - \hat{s}_t$. The remainder term exhibits fluctuation consistent with the southern oscillation.

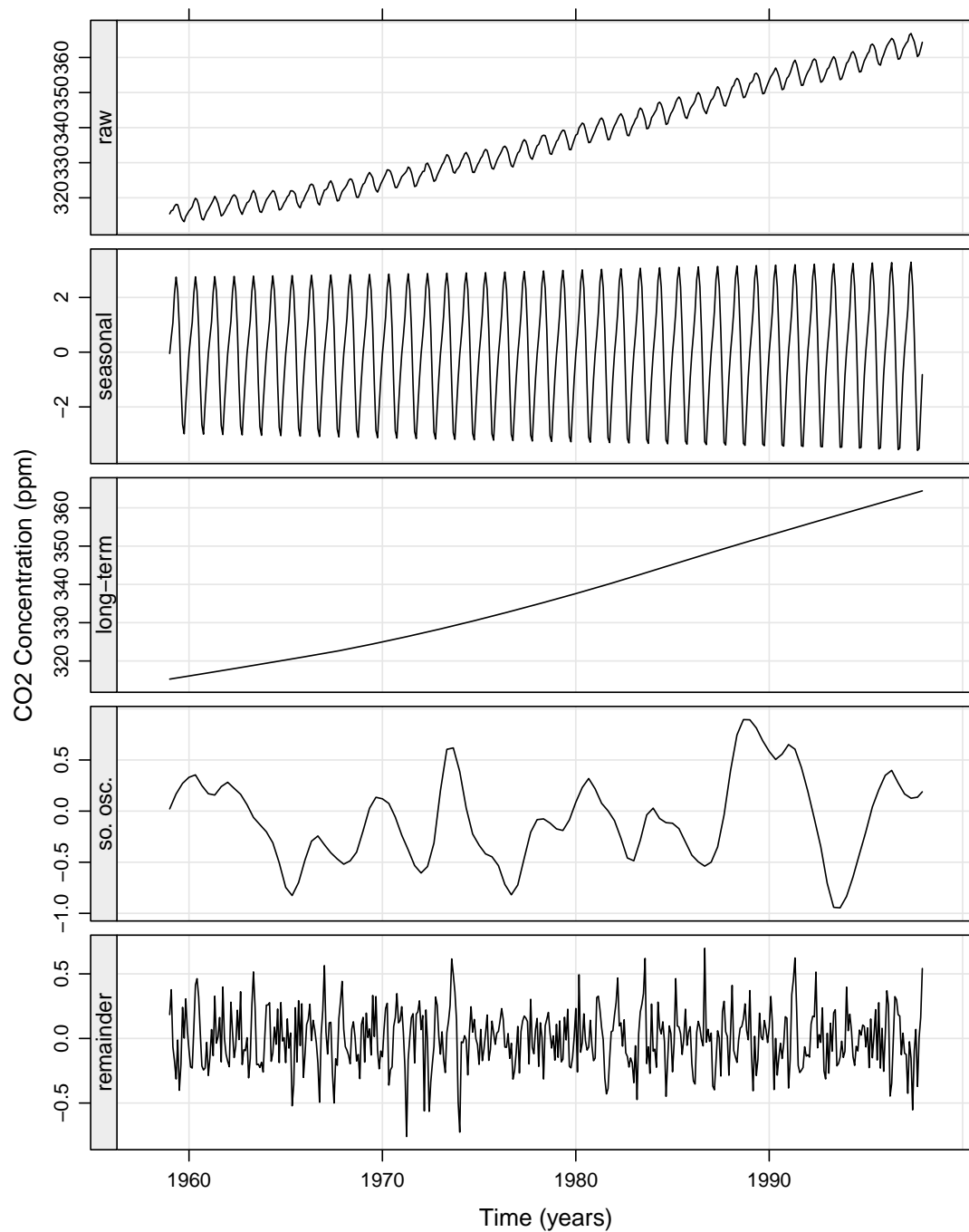


Figure 5.26. CO₂ decomposition with a low-frequency post-trend smoothing of $\lambda = 1$ and $q = 201$ to obtain a long-term trend component followed by a smoothing of $\lambda = 1$ and $q = 35$ to obtain a southern oscillation component.

5.3 Figures for STL Advancements

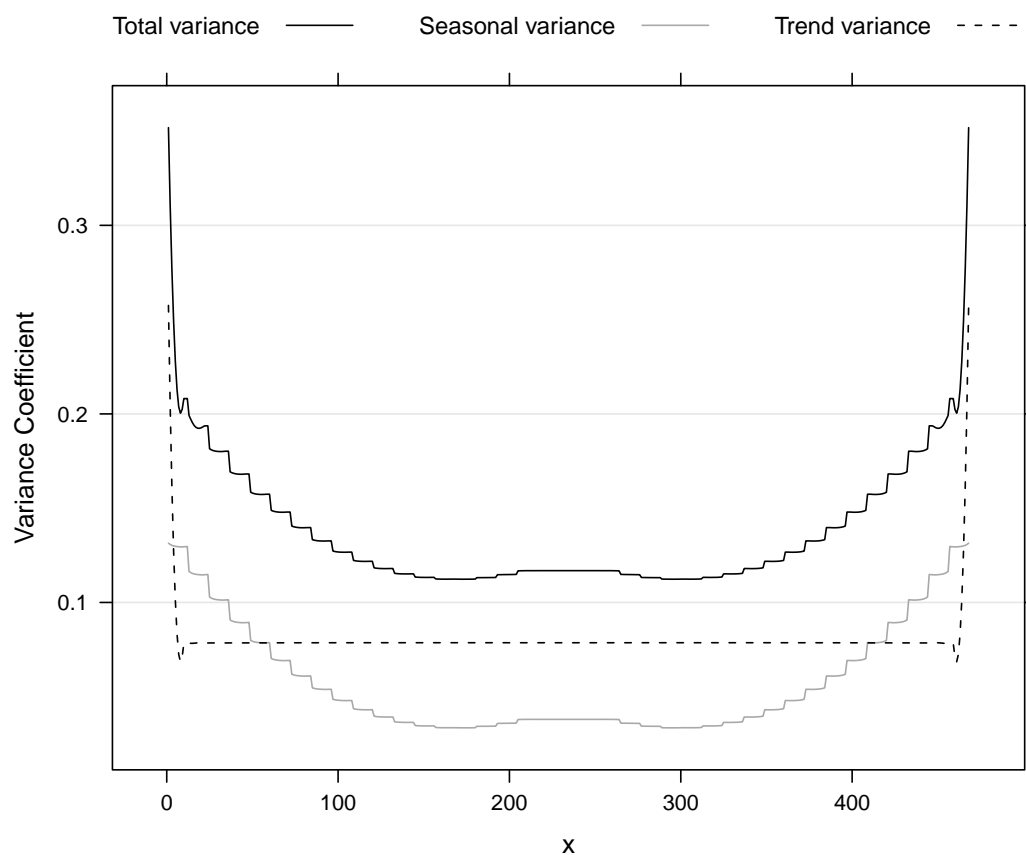


Figure 5.27. Variance coefficients for CO_2 decomposition components for original CO_2 decomposition (see figure 5.19).

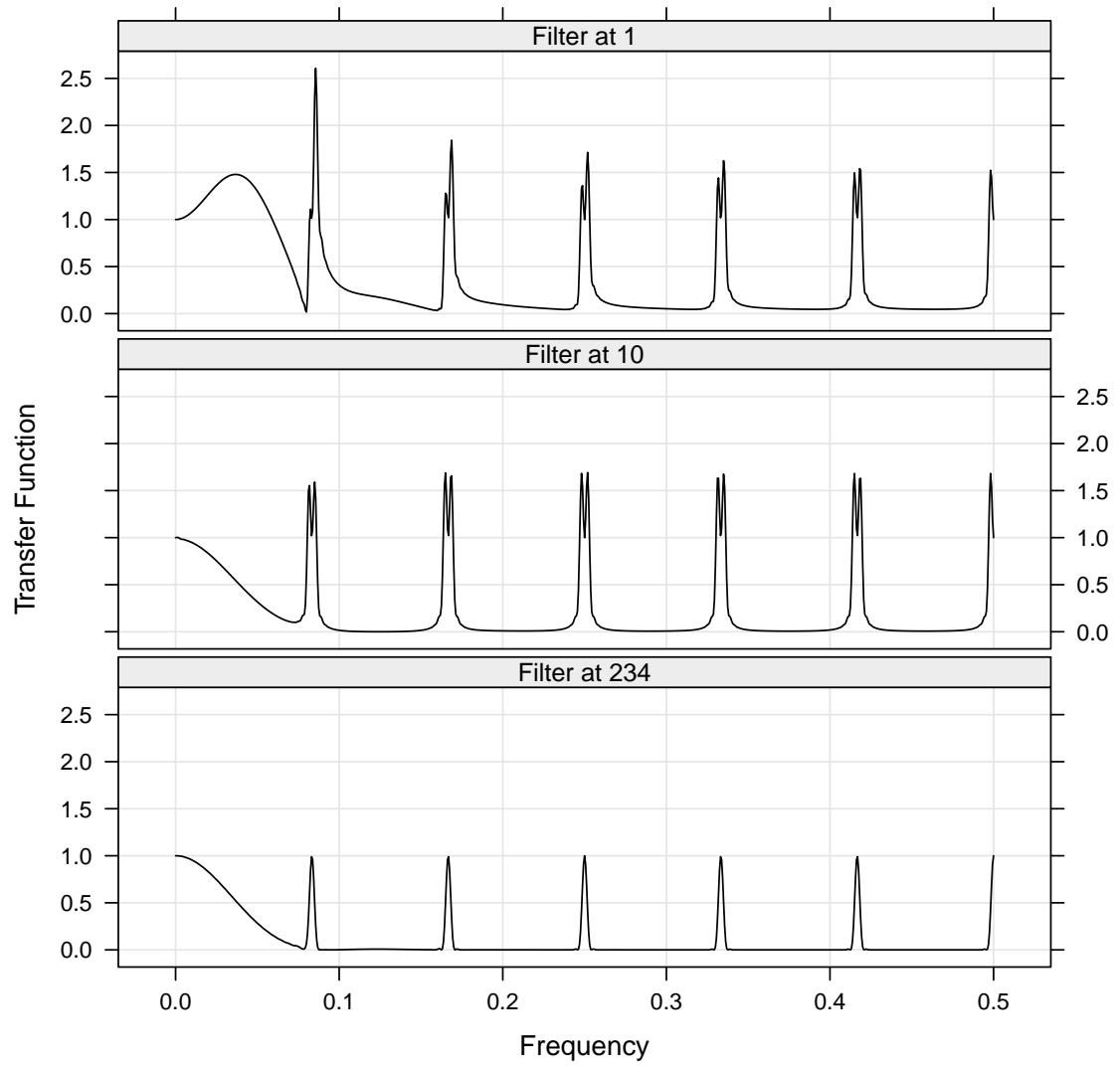


Figure 5.28. Power transfer functions for STL operator matrix L^* at design points 1 (endpoint), 10, and 234 (middle).

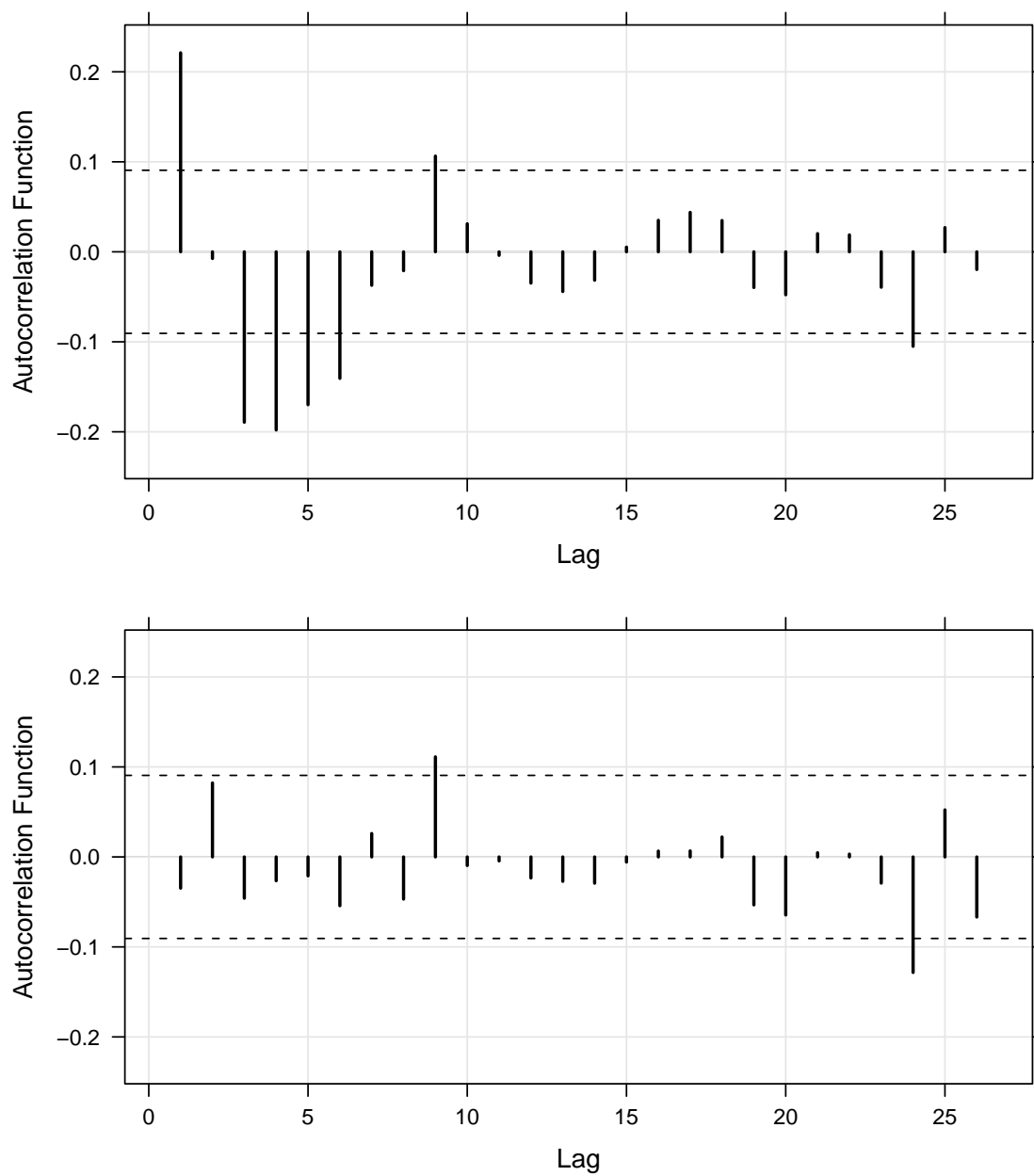


Figure 5.29. Autocorrelation functions for CO₂ remainder term (top) and residual term after ARMA modeling to remove dependence in the remainder term (bottom).

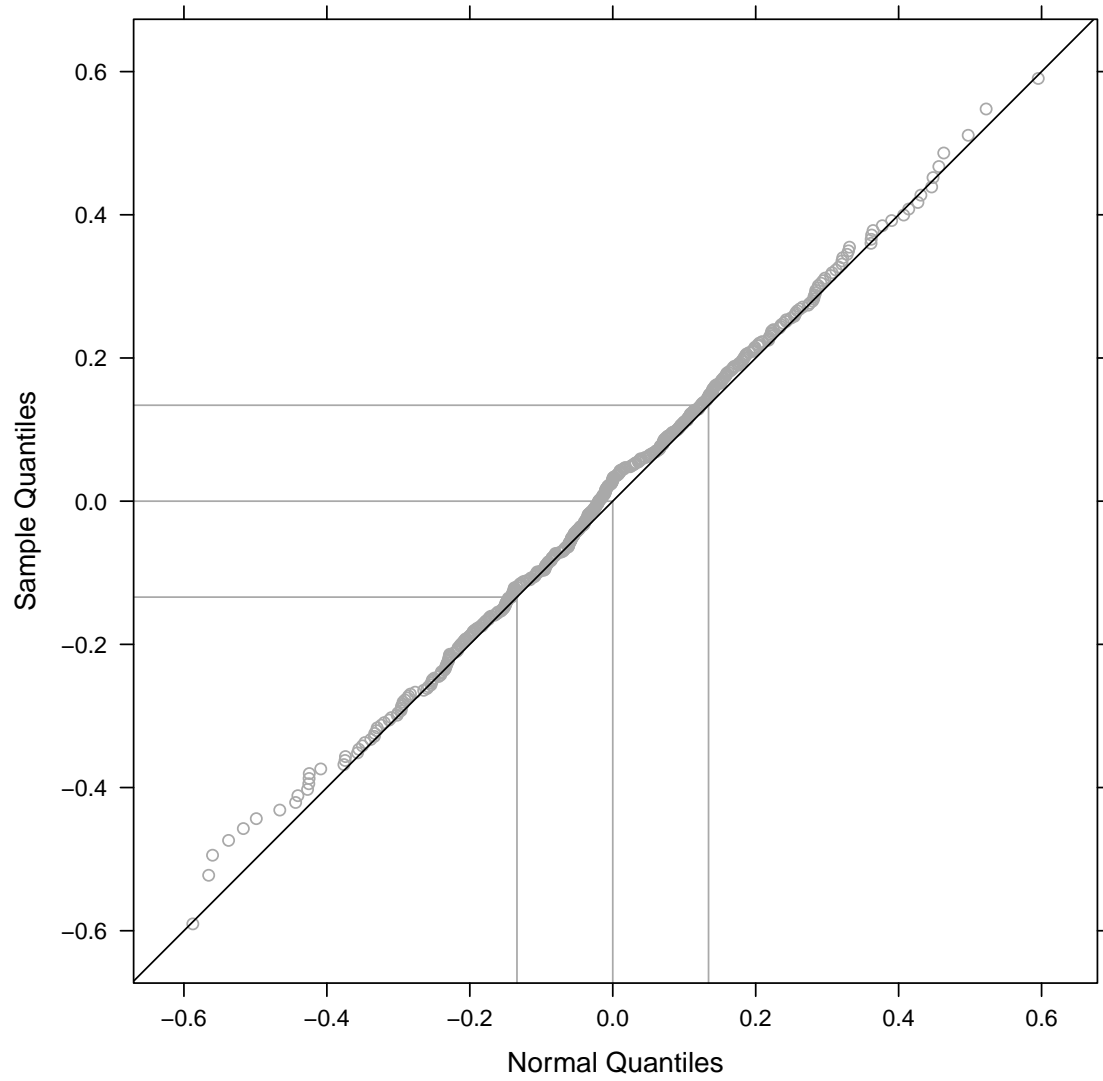


Figure 5.30. Normal quantile plot of remainder term of CO₂ model after ARMA modeling on remainder term.

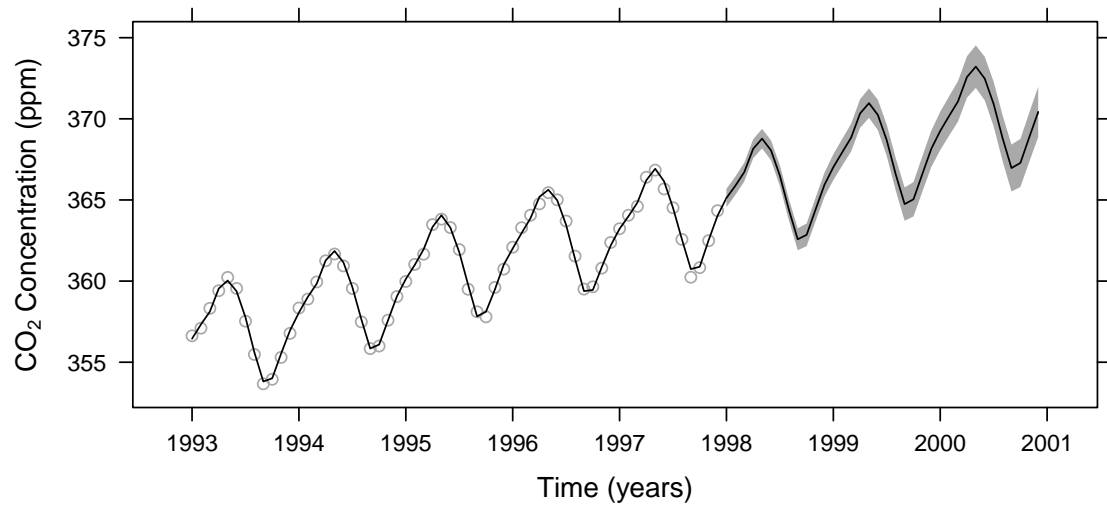


Figure 5.31. Predicting the CO₂ series ahead 3 years with 95% prediction intervals.

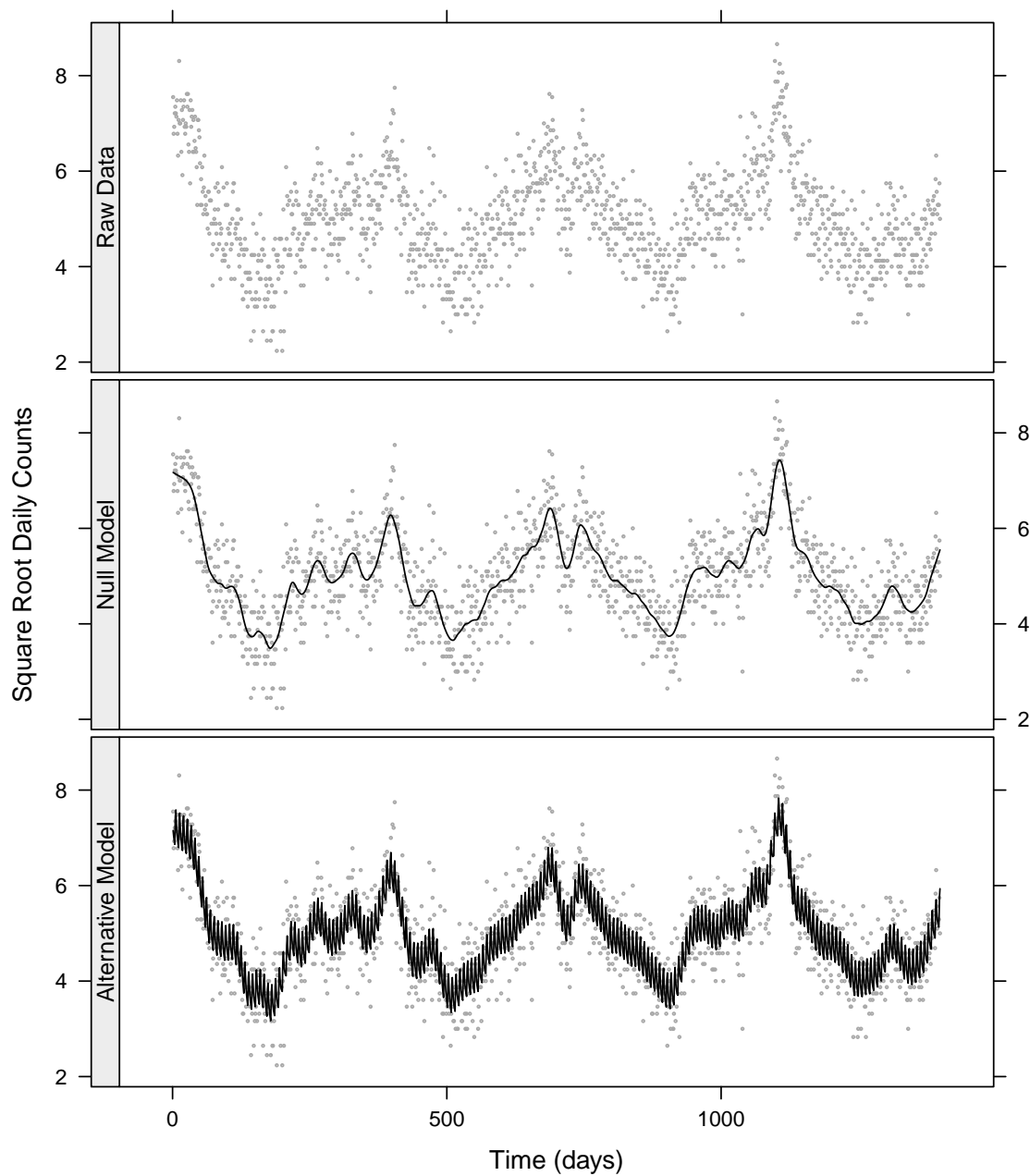


Figure 5.32. Simulated data to mimic daily visit counts to an emergency department (ED), with fitted values under the null model of no seasonality and under the alternative of a seasonal day-of-the-week component.

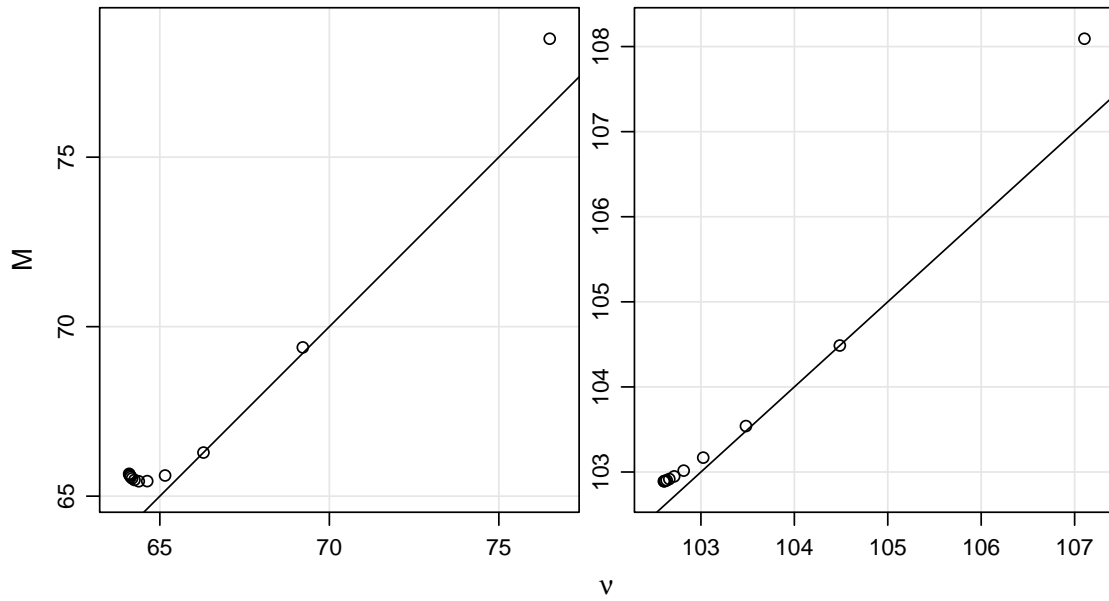


Figure 5.33. C_p plot for CO_2 decomposition with varying seasonal span $n_{(s)}$ with (right) and without (left) ARMA modeling of the remainder.

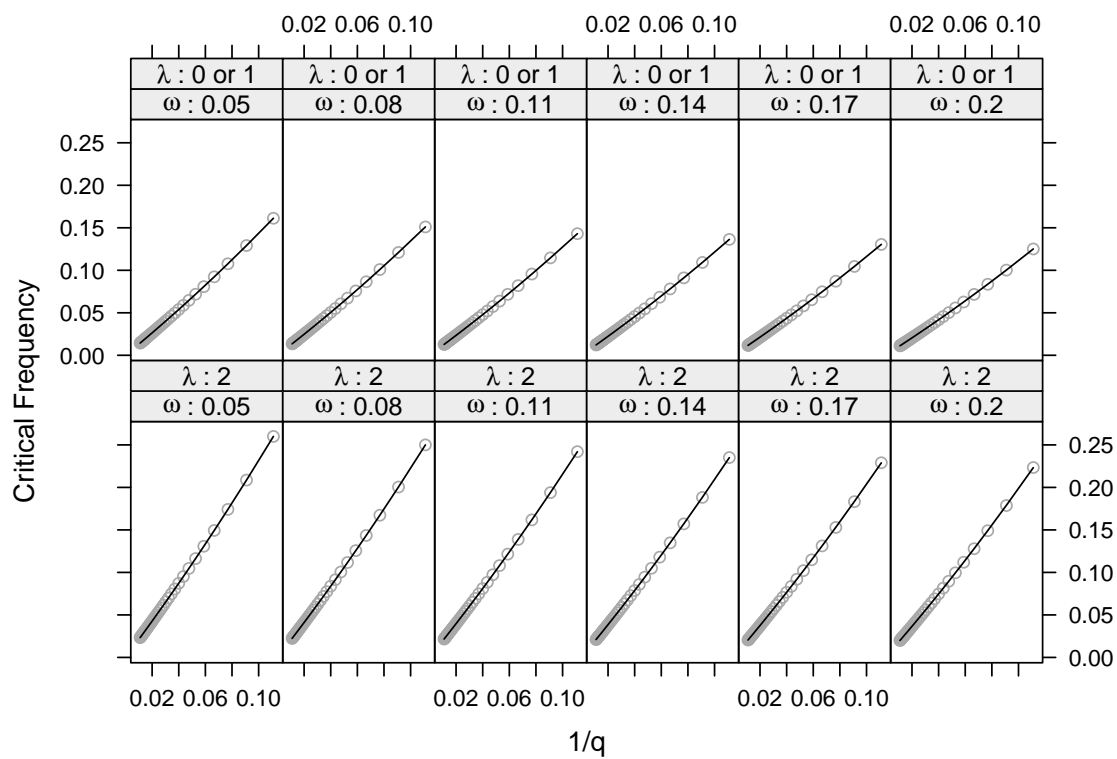


Figure 5.34. Critical frequencies of a symmetric loess smooth vs. q^{-1} by λ and ω with a fitted quadratic polynomial superposed.

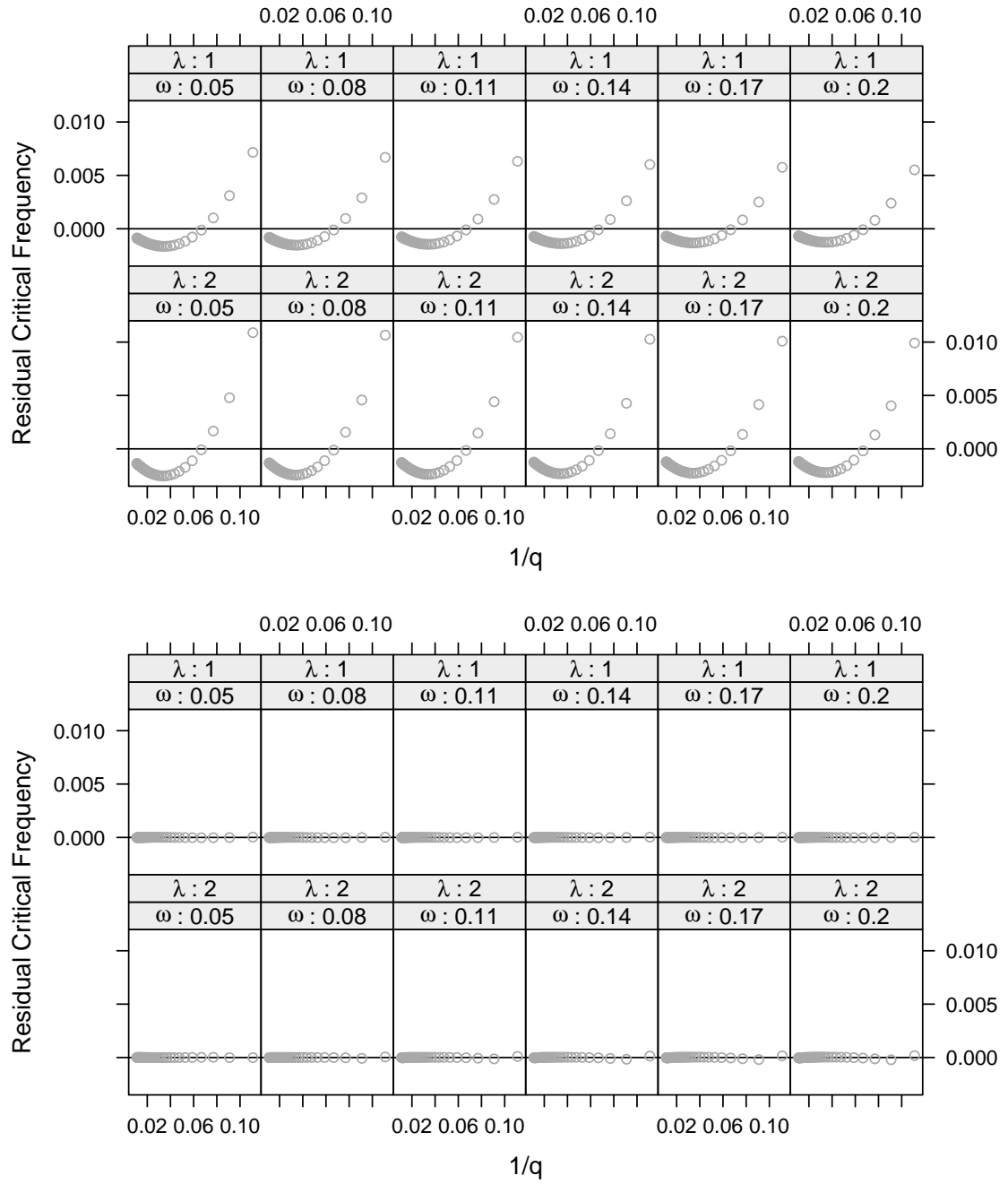


Figure 5.35. Residuals from fitting critical frequencies vs. q^{-1} for a linear fit with no intercept (top) and quadratic model (bottom). The bottom residuals correspond to the fits in figure 5.34.

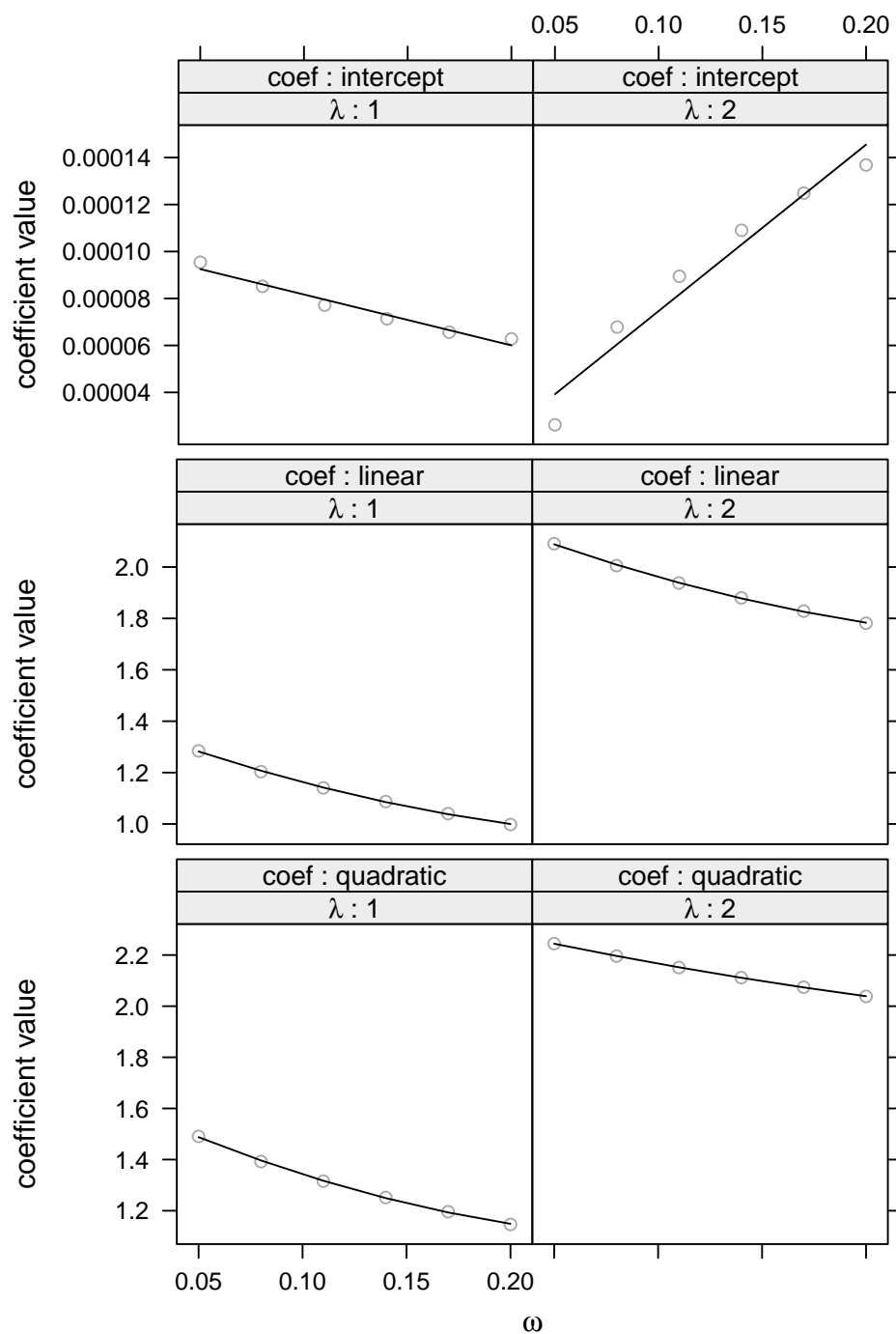


Figure 5.36. Coefficients from quadratic polynomial fits of critical frequency vs. q^{-1} plotted against ω by λ . The fitted line is a linear fit for the intercept coefficient and a quadratic polynomial fit for the linear and quadratic coefficient.

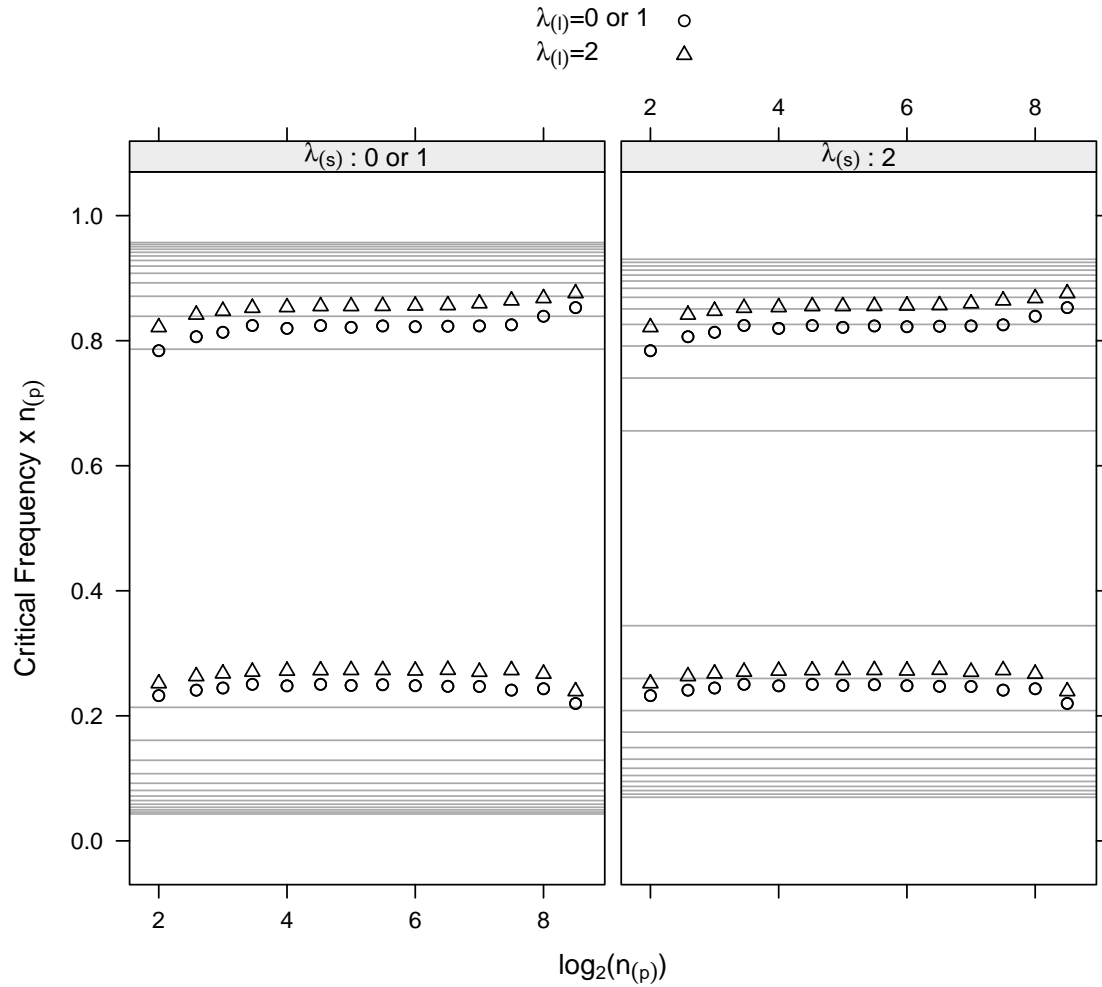


Figure 5.37. $n_{(p)}f_h^{(lower)}$ and $n_{(p)}f_h^{(upper)}$ vs. $\log_2 n_{(p)}$ for $\lambda_{(l)} = 0, 1, 2$ and $\omega = 0.05$. The horizontal lines correspond to $n_{(p)}f_c^{(lower)}$ and $n_{(p)}f_c^{(upper)}$ for $\lambda_{(s)} = 0$ or 1 in the first panel and $\lambda_{(s)} = 2$ in the second panel. Each line corresponds to a different value of $n_{(s)}$, with $n_{(s)} = 7 + 2i$, $i = 0, \dots, 12$. The horizontal lines with $n_{(s)}$ closest to the middle of the y -axis correspond to upper and lower critical frequencies with $n_{(s)} = 7$, with $n_{(s)}$ increasing as the lines move further from the middle.

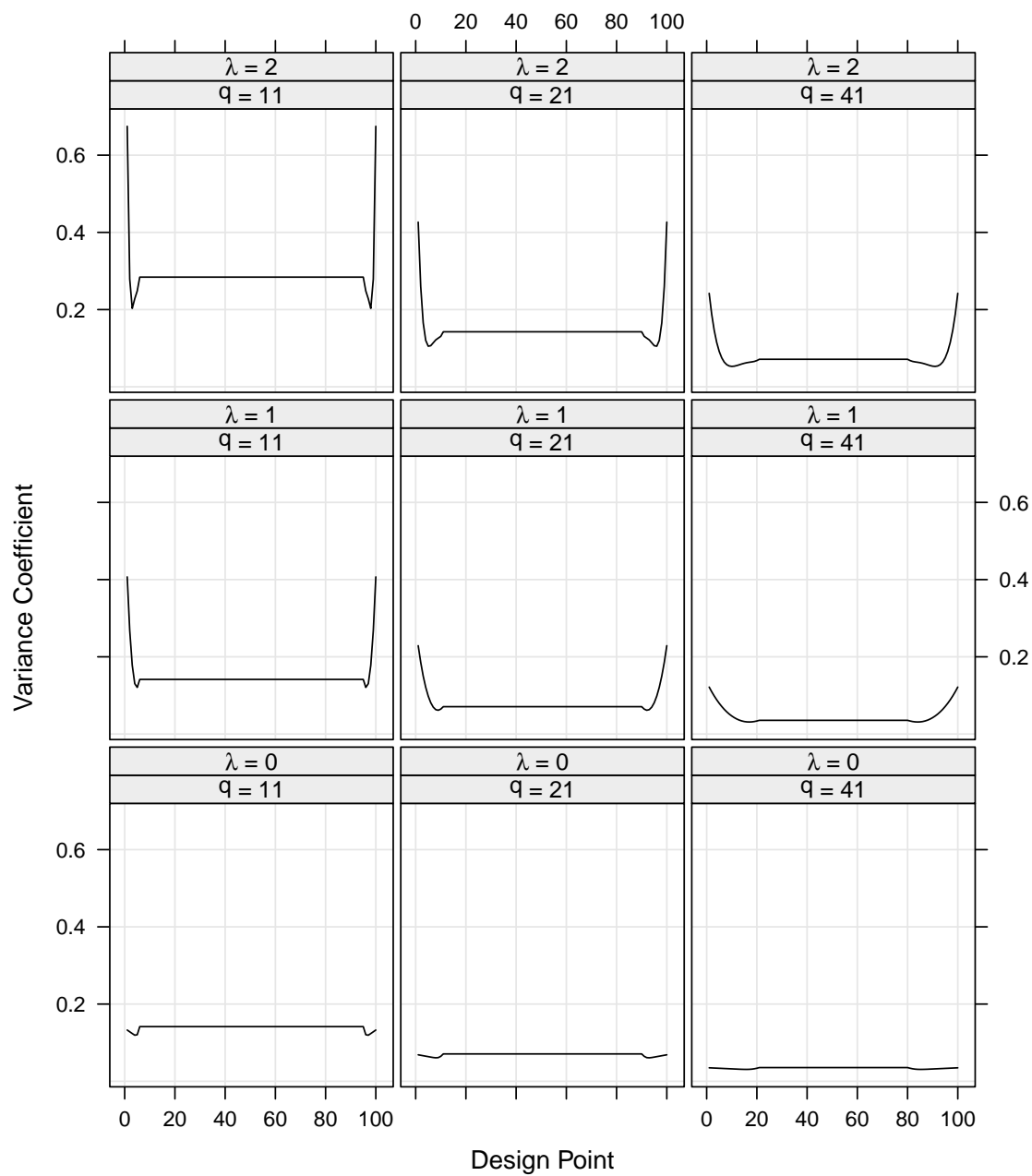


Figure 5.38. Variance coefficients along the design space for several choice of λ and q .

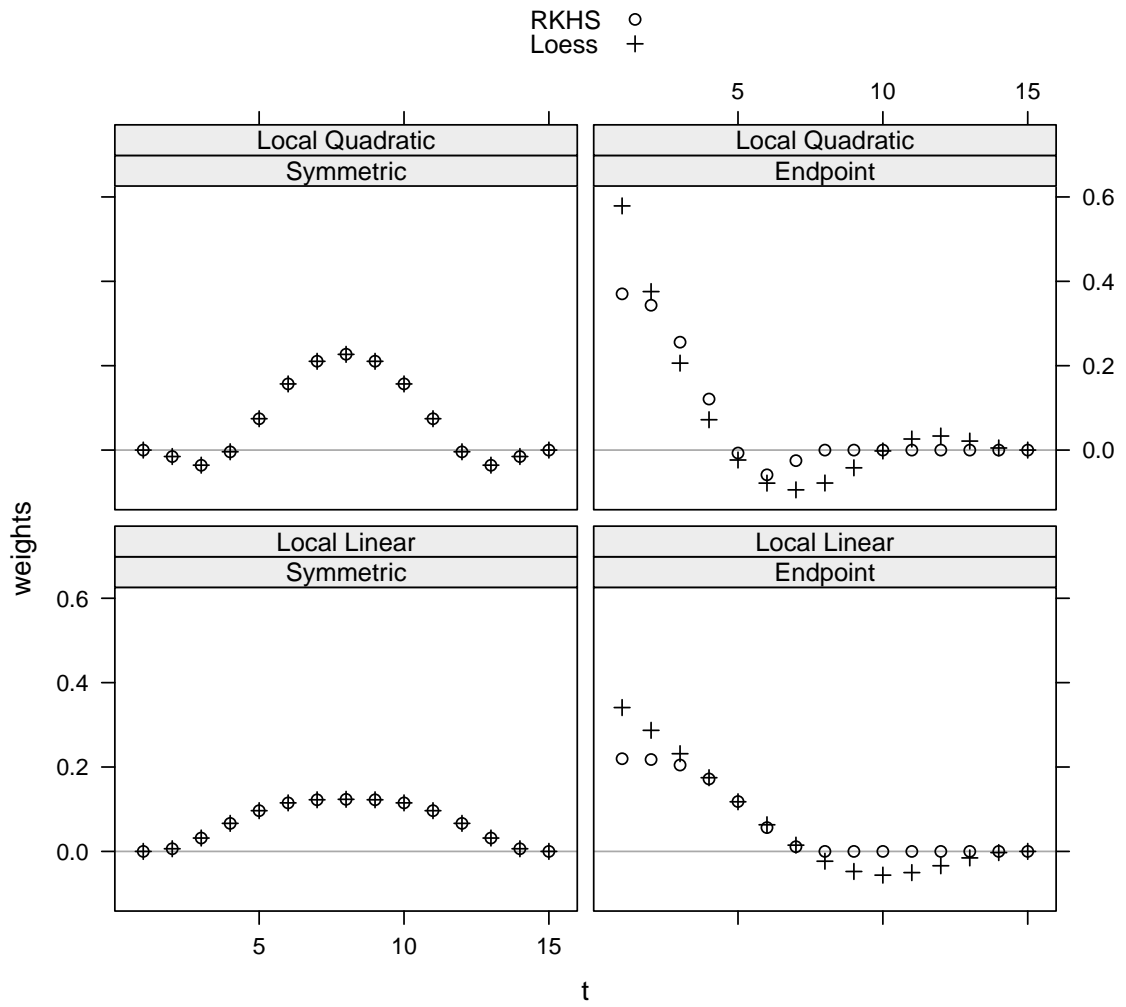


Figure 5.39. Loess and RKHS loess operator weights for symmetric and endpoint fits with $q = 15$ and with $\lambda = 1$ and $\lambda = 2$.

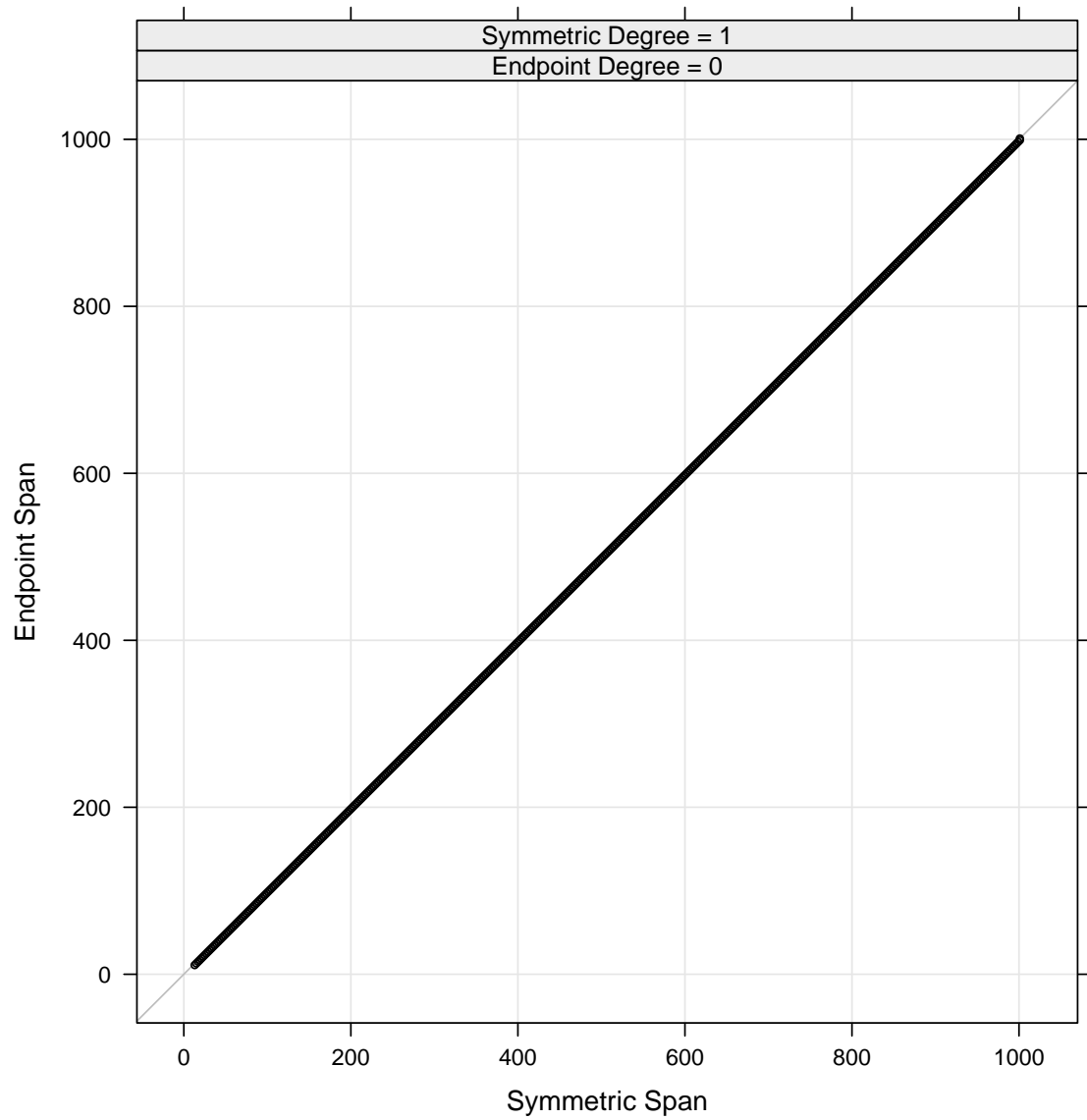


Figure 5.40. Symmetric $\lambda = 1$ span vs. endpoint $\lambda = 0$ span yielding equivalent variance.

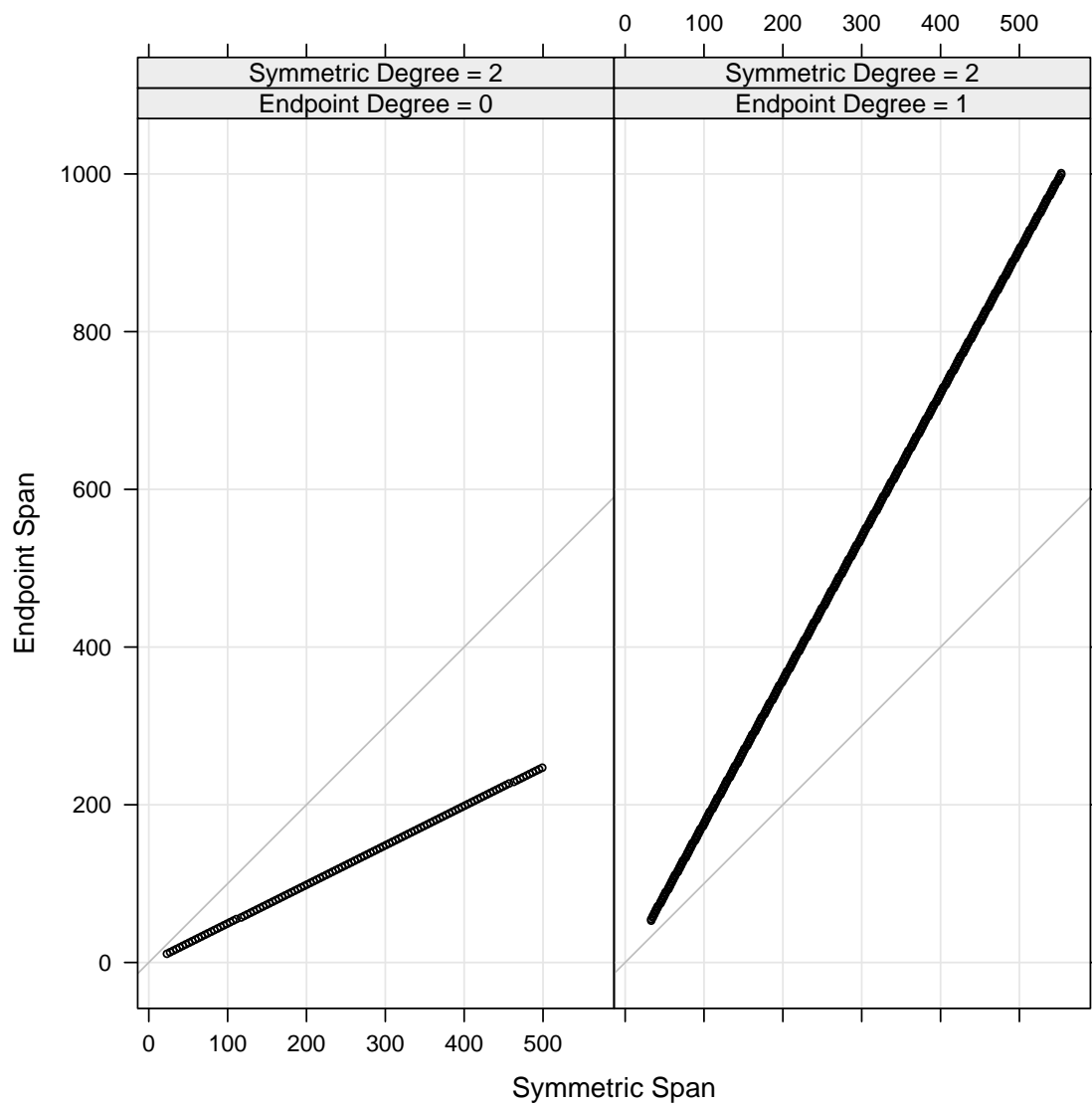


Figure 5.41. Symmetric $\lambda = 2$ span vs. endpoint $\lambda = 0$ and $\lambda = 1$ span yielding equivalent variance.

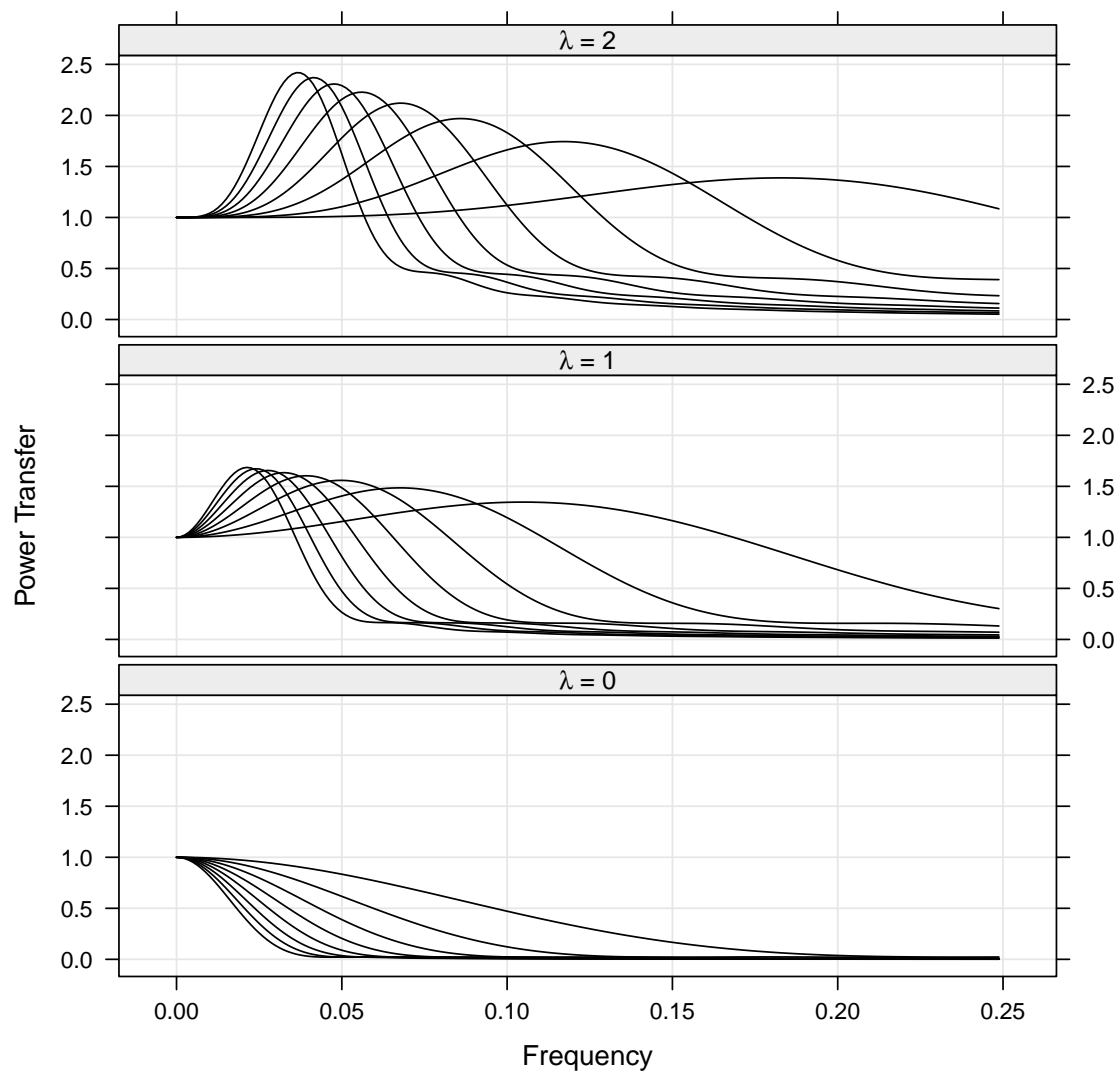


Figure 5.42. Power transfer functions for loess fits with $\lambda = 0, 1, 2$ and q ranging from 7 to 35.

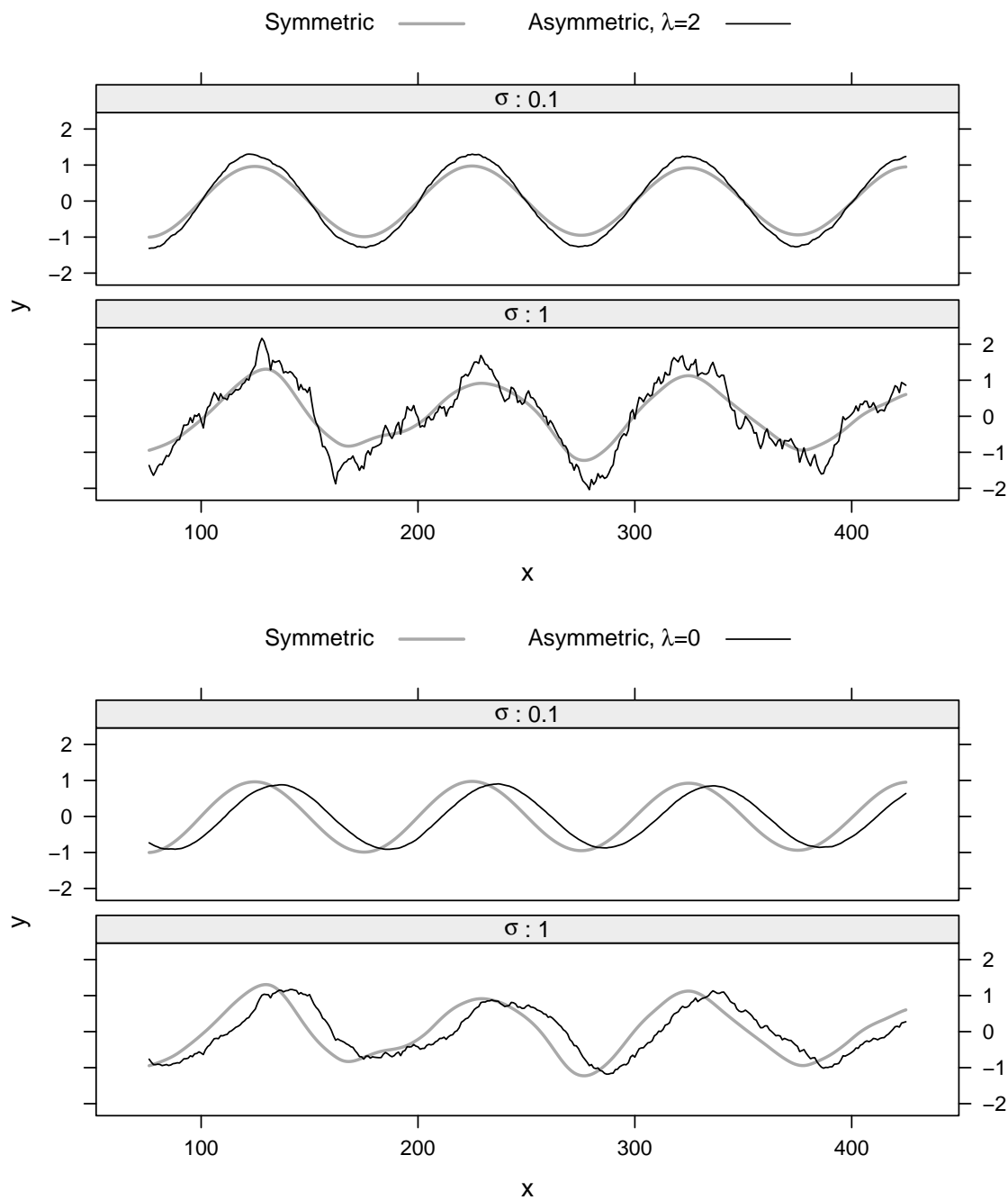


Figure 5.43. Simulated series from the sinusoid signal with added noise with $\sigma = 0.1$ and $\sigma = 1$. The gray line is the symmetric fit and the black line is the endpoint asymmetric fit with $(\lambda_o = 2, q_o = 91)$ fit (top), and the endpoint asymmetric $(\lambda_b = 0, q_b = 45)$ fit (bottom).

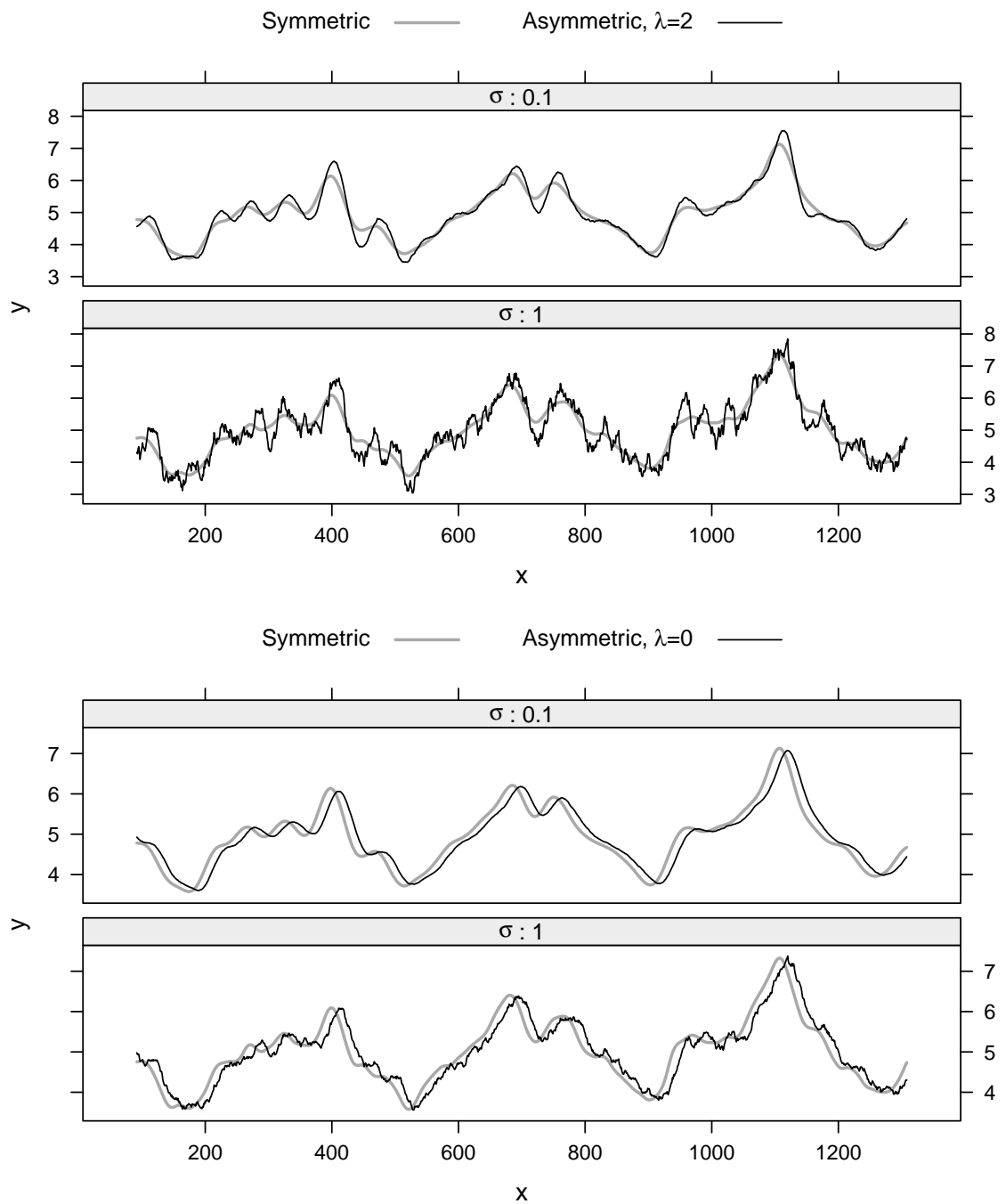


Figure 5.44. Simulated series from the ED signal with added noise with $\sigma = 0.1$ and $\sigma = 1$. The gray line is the symmetric fit and the black line is the endpoint asymmetric fit with $(\lambda_o = 2, q_o = 75)$ fit (top), and the endpoint asymmetric $(\lambda_b = 0, q_b = 37)$ fit (bottom).

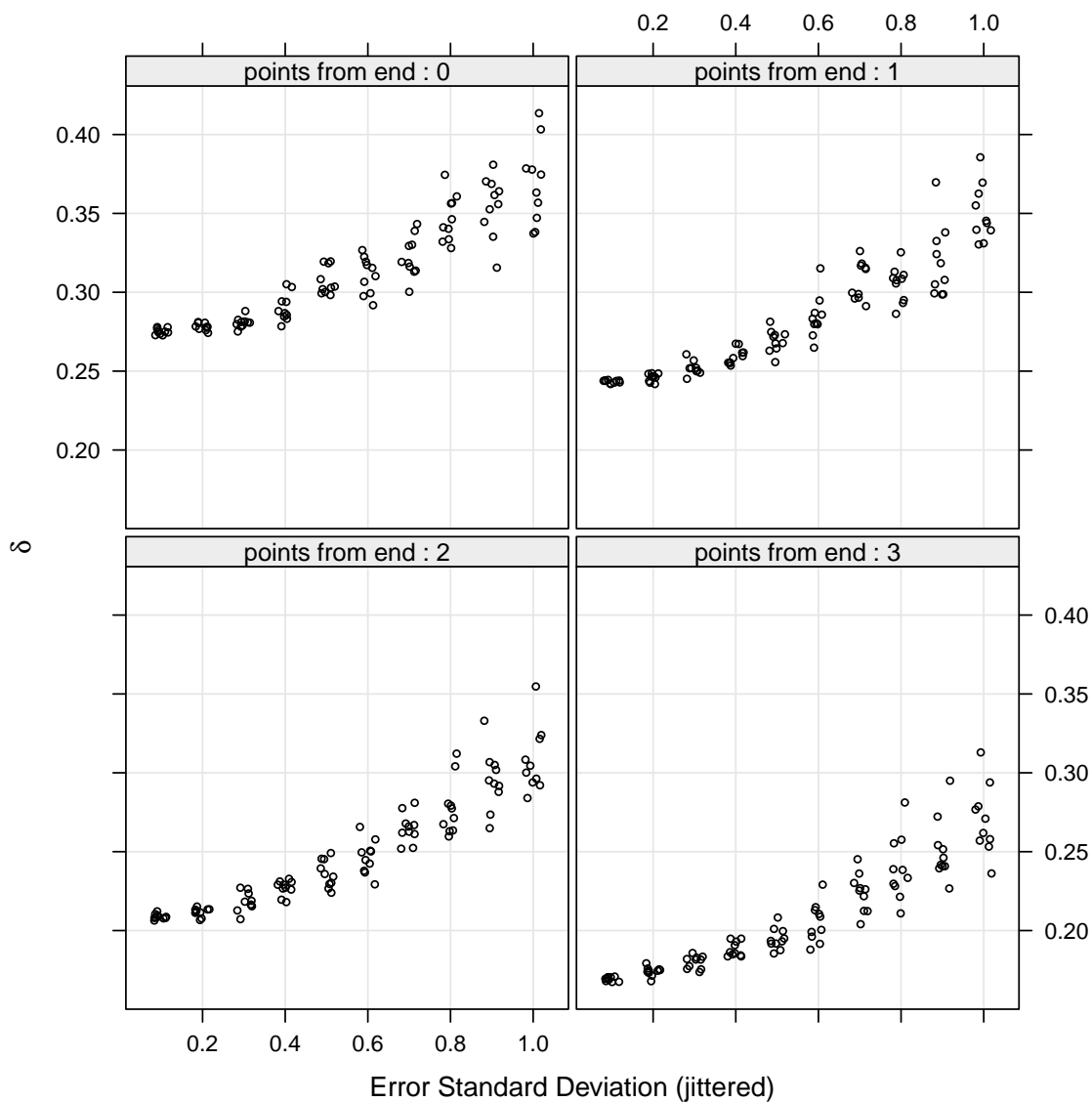


Figure 5.45. Optimal proportion of blending δ vs. error standard deviation for sinusoid data, with 10 replicates per standard deviation. “Points from end : 0” corresponds to the endpoint fit.

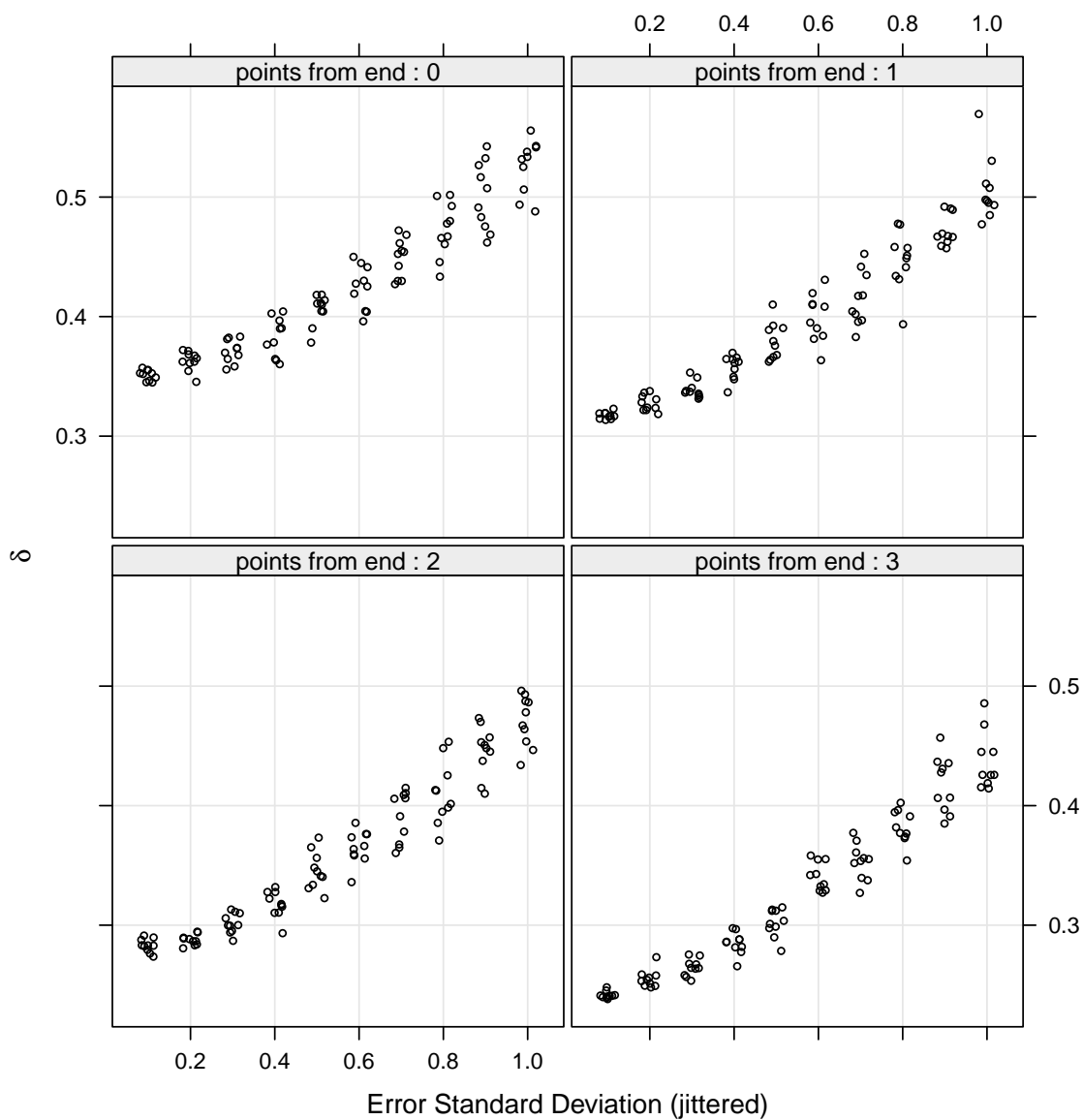


Figure 5.46. Optimal proportion of blending δ vs. error standard deviation for ED data, with 10 replicates per standard deviation. “Points from end : 0” corresponds to the endpoint fit.

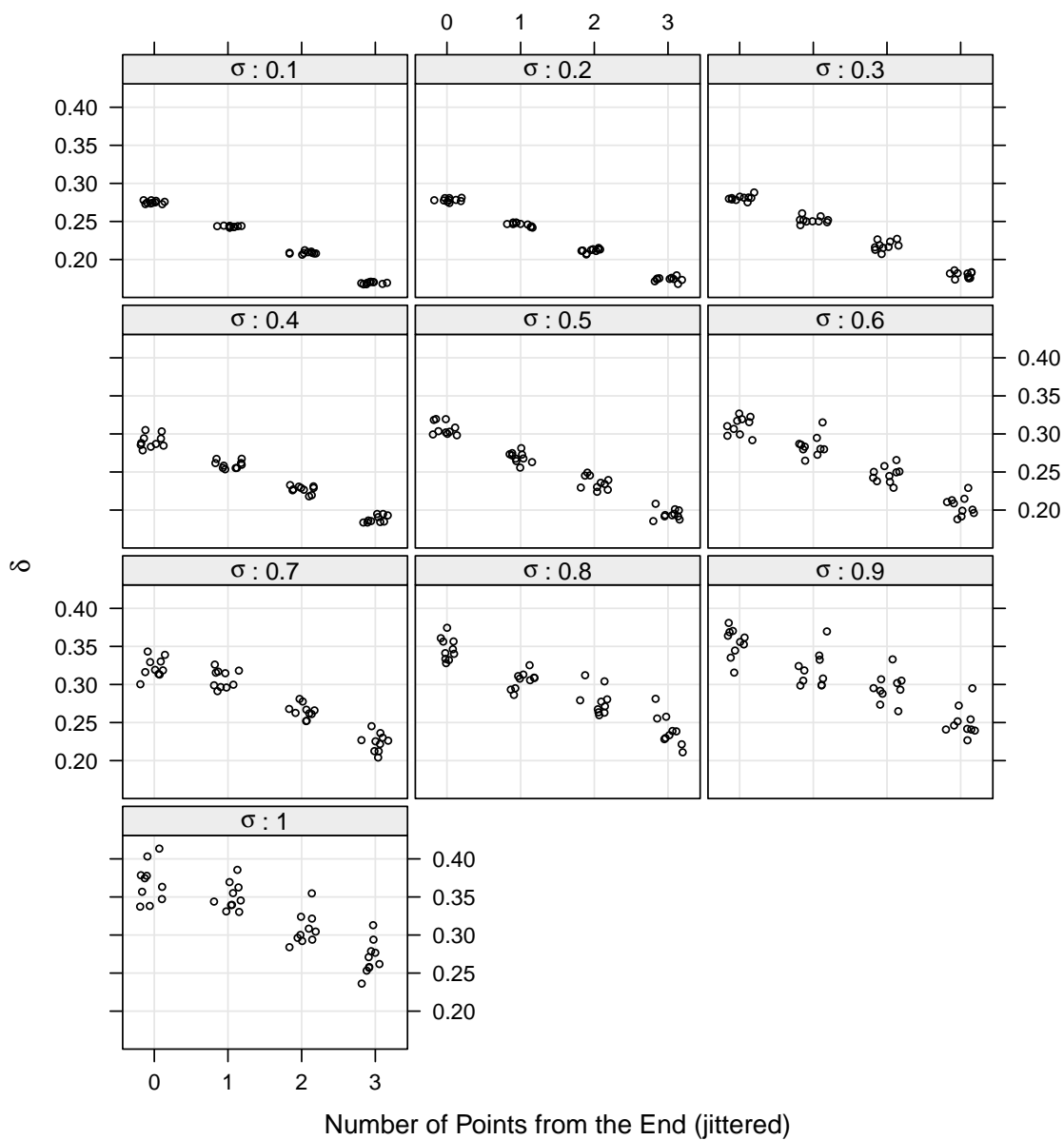


Figure 5.47. Optimal proportion of blending δ vs. number of points from end for sinusoid data.

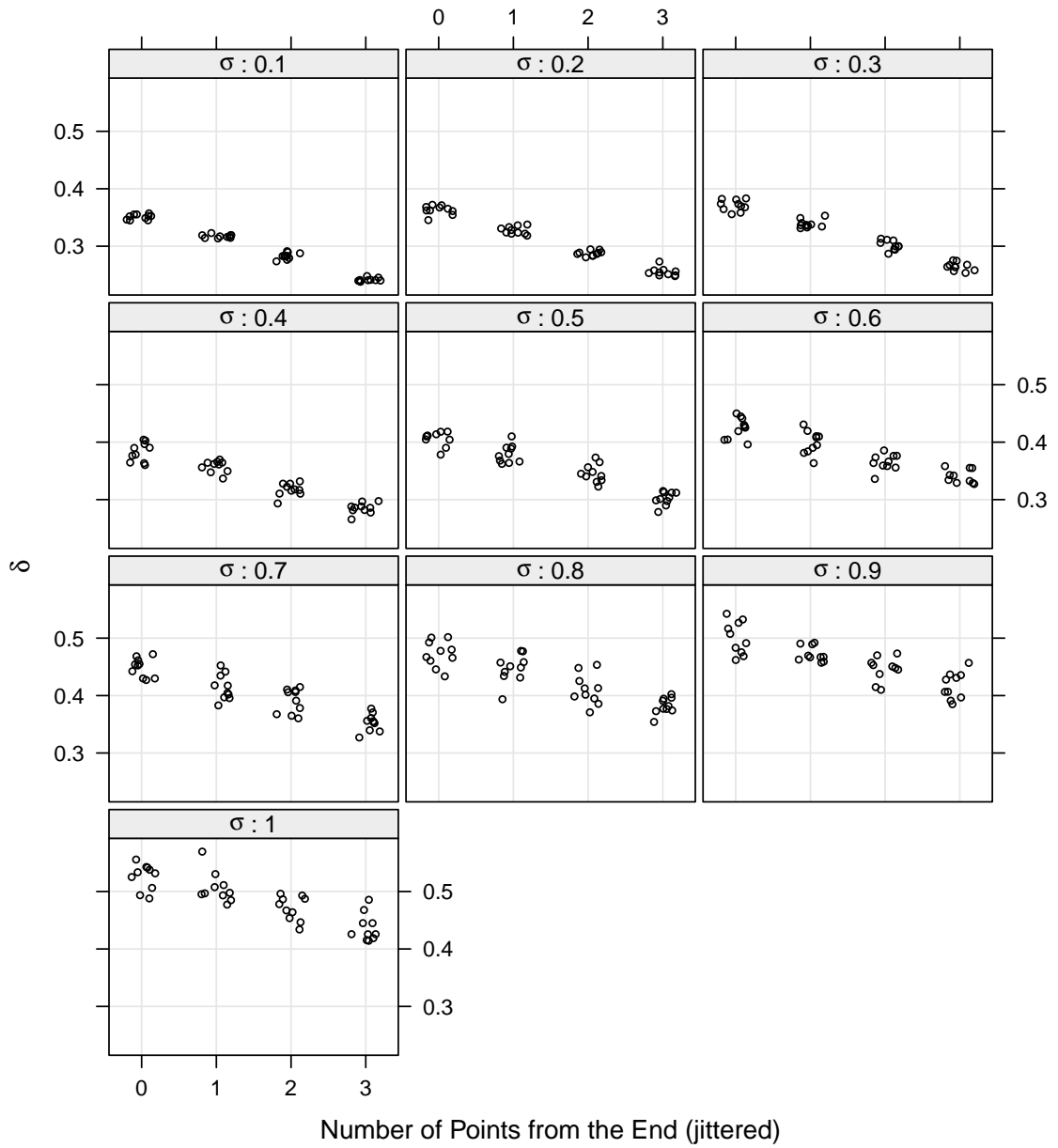


Figure 5.48. Optimal proportion of blending δ vs. number of points from end for ED data.

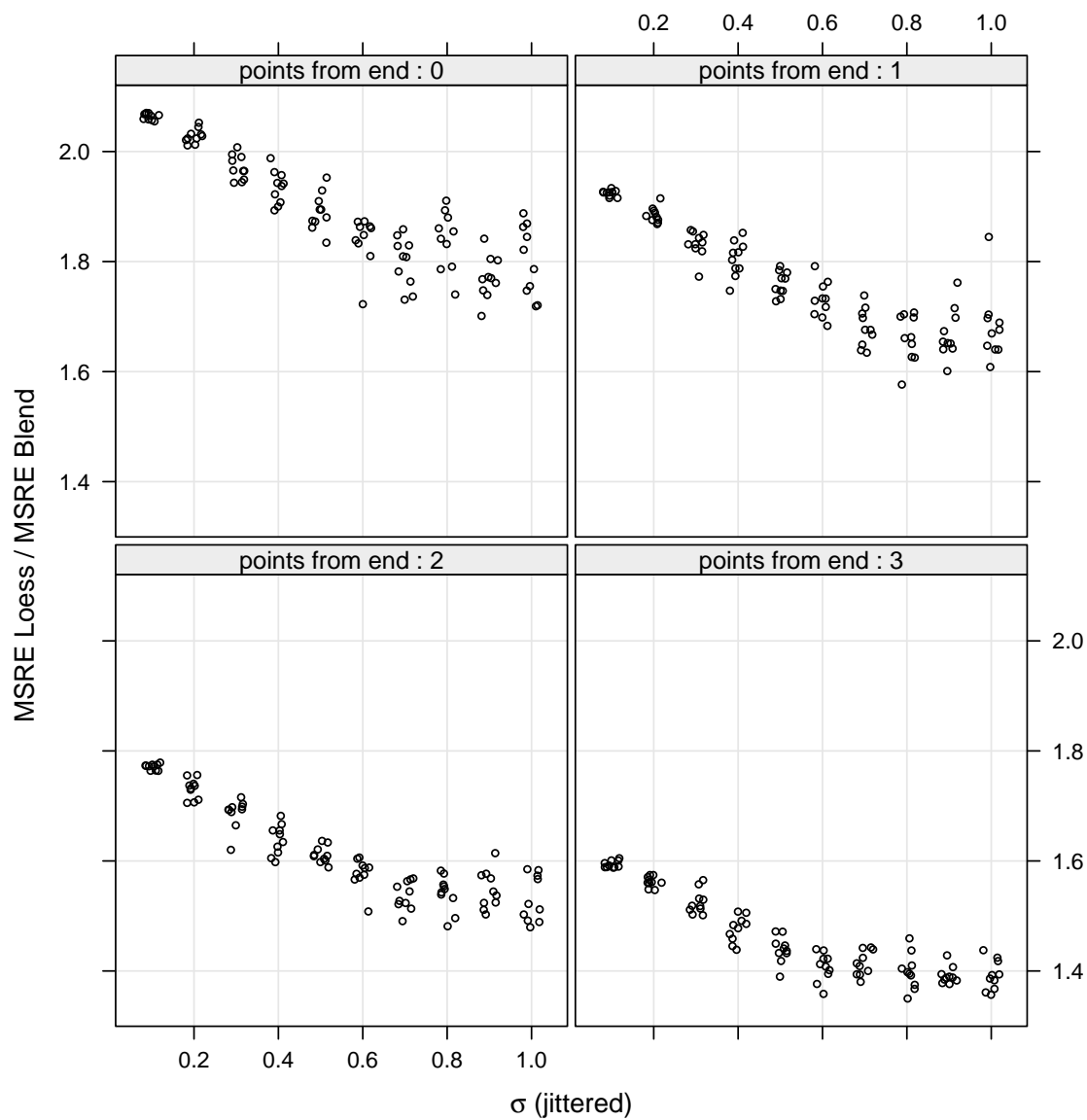


Figure 5.49. MSRE loess / MSRE blend vs. σ for sinusoid data.

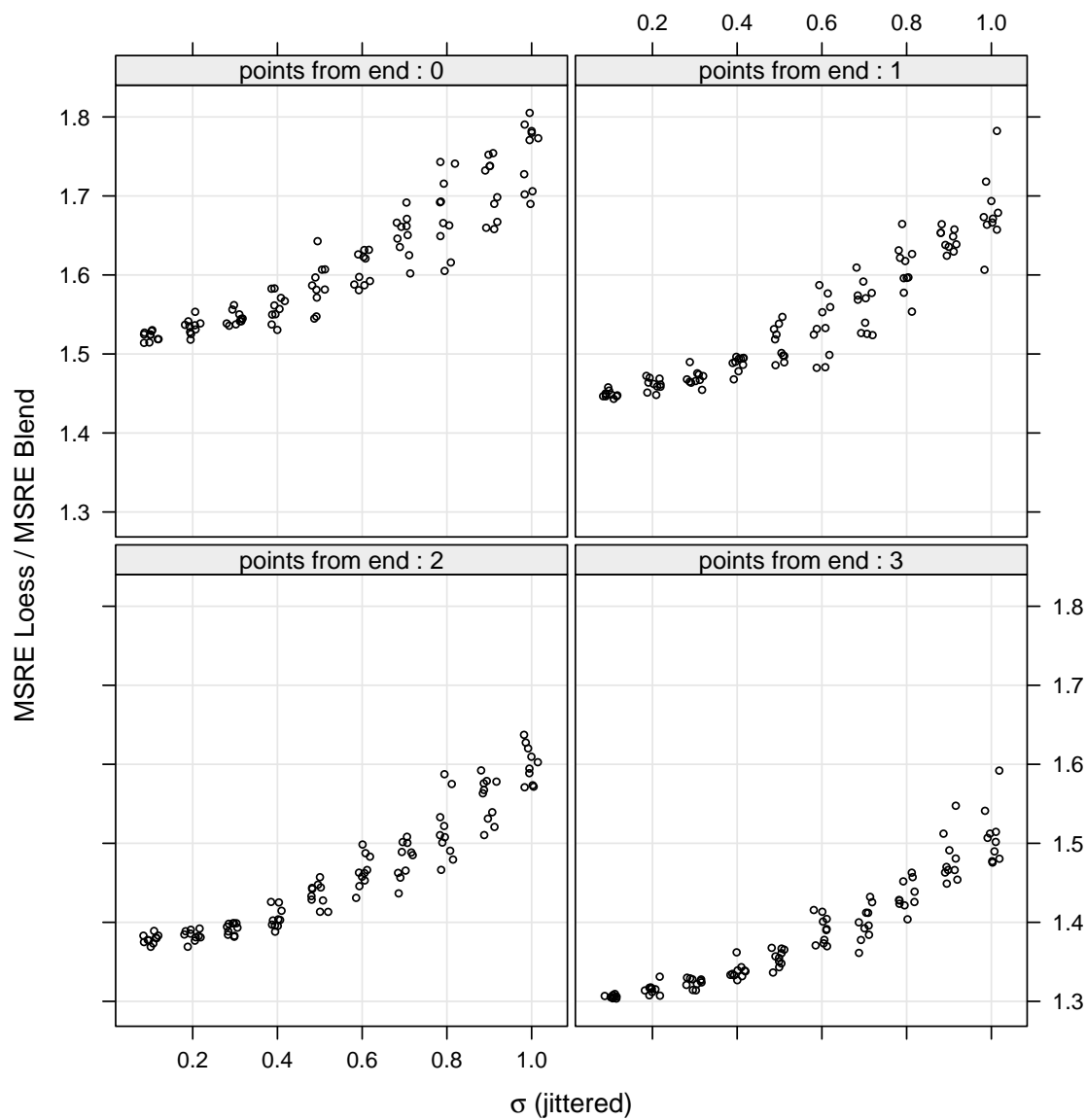


Figure 5.50. MSRE loess / MSRE blend vs. σ for ED data.

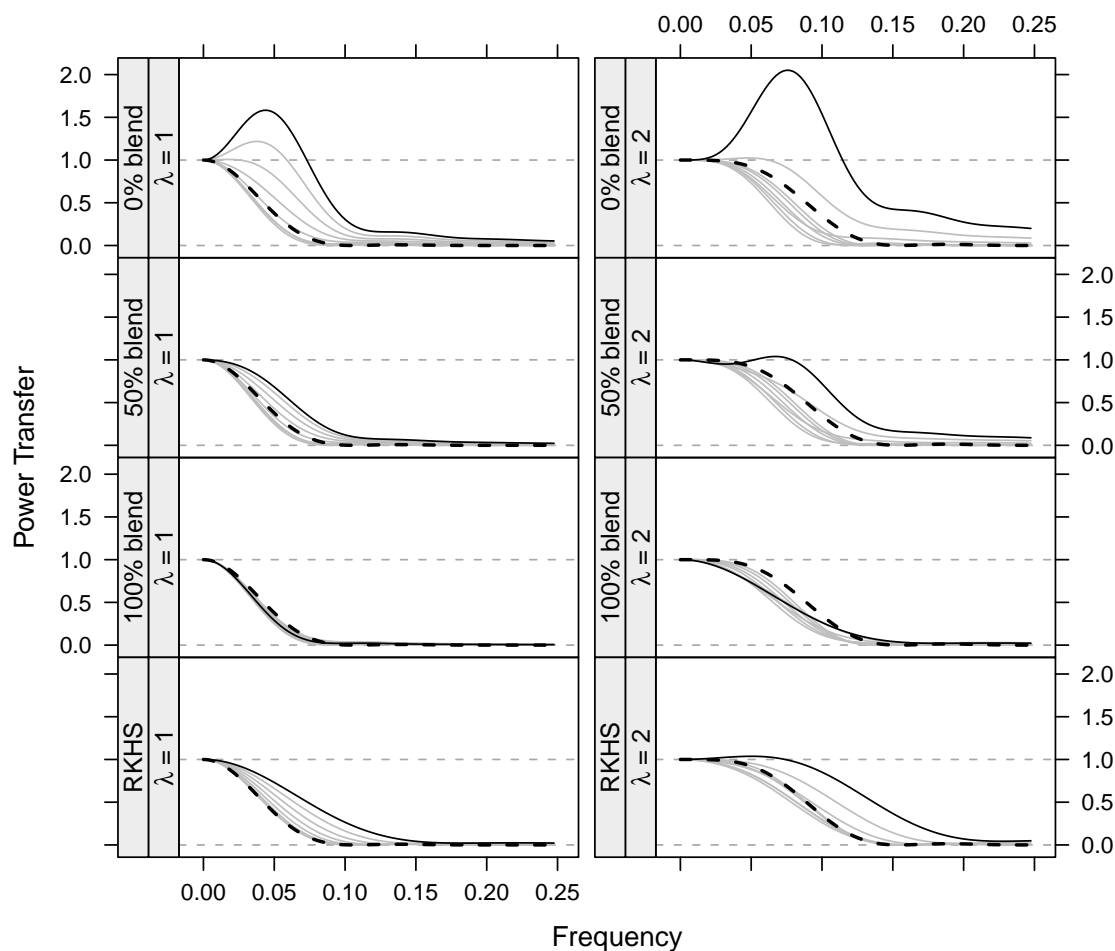


Figure 5.51. Power transfer functions for RKHS loess and loess with $\delta = 0, 0.5$, and 1 for smoothing with $q = 17$. The dashed line shows the transfer function for the symmetric fit, and each gray line represents the transfer function as the fit becomes more asymmetric, until it reaches the endpoint, which is represented by the solid line.

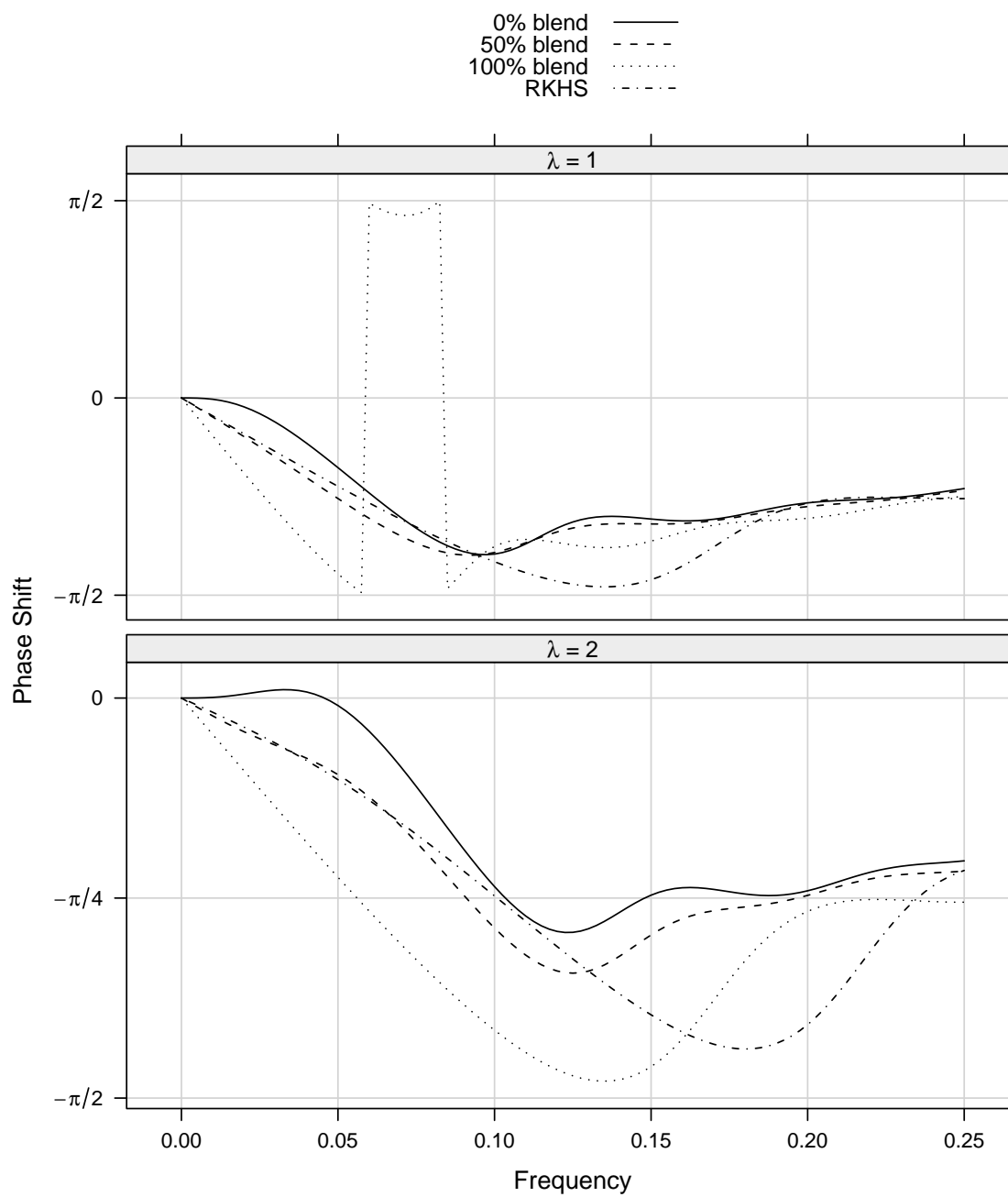


Figure 5.52. Phase shift function at the endpoint for RKHS loess and loess with $\delta = 0, 0.5$, and 1 for smoothing with $q = 17$.

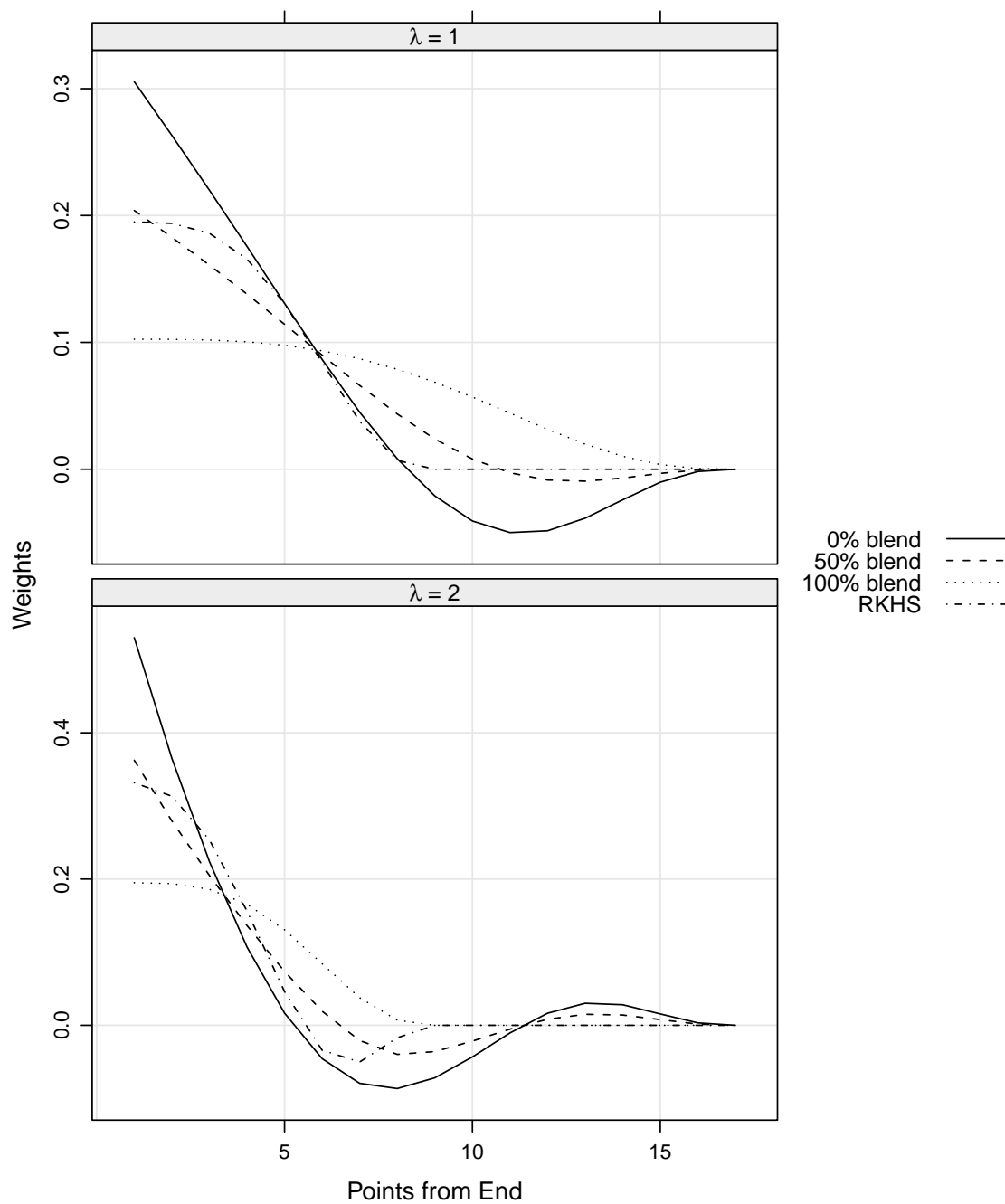


Figure 5.53. Endpoint operator smoothing weights for RKHS loess and loess with $\delta = 0, 0.5, \text{ and } 1$ for smoothing with $q = 17$.

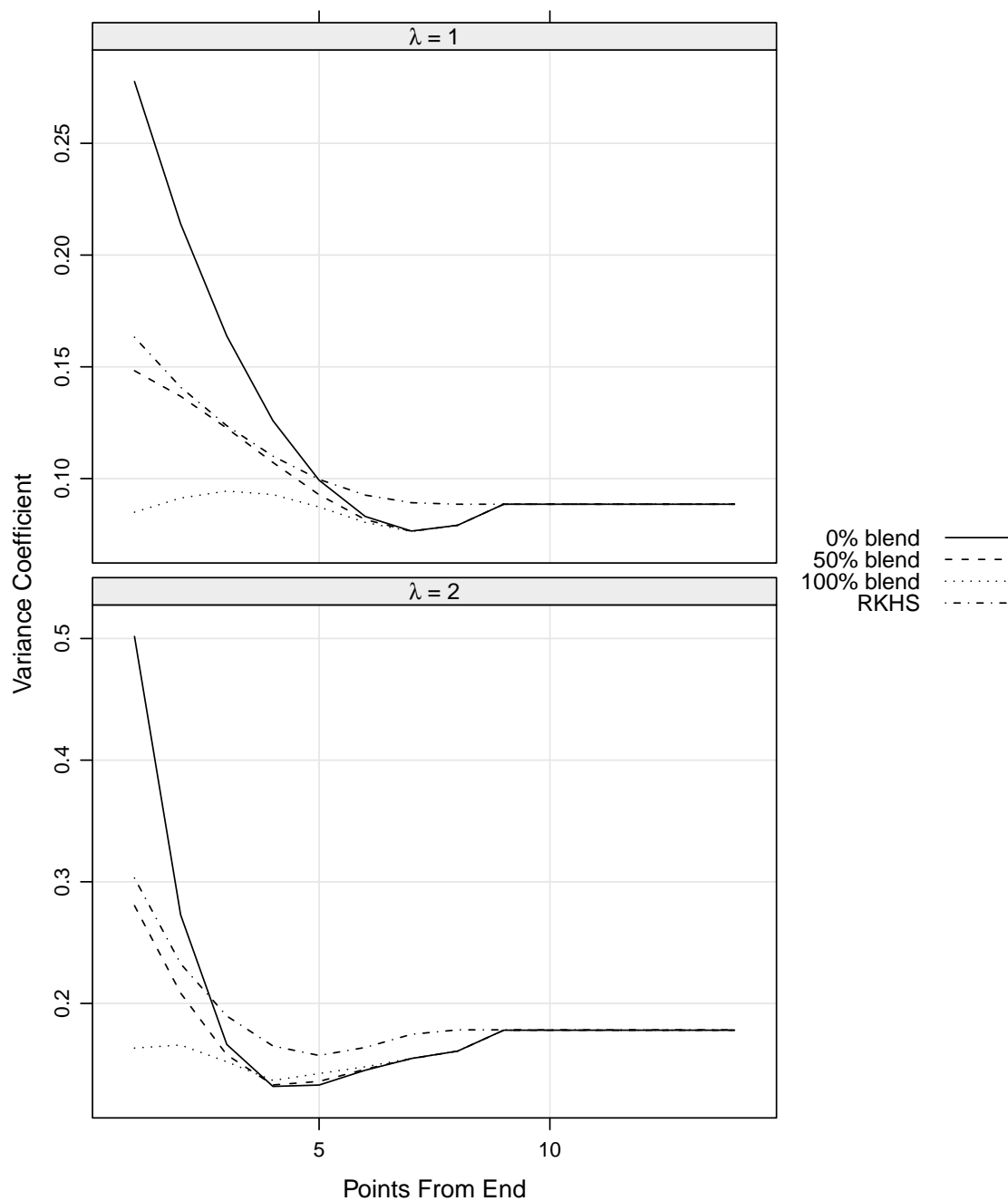


Figure 5.54. Variance coefficients at and near the boundary for RKHS loess and loess with $\delta = 0, 0.5$, and 1 for smoothing with $q = 17$.

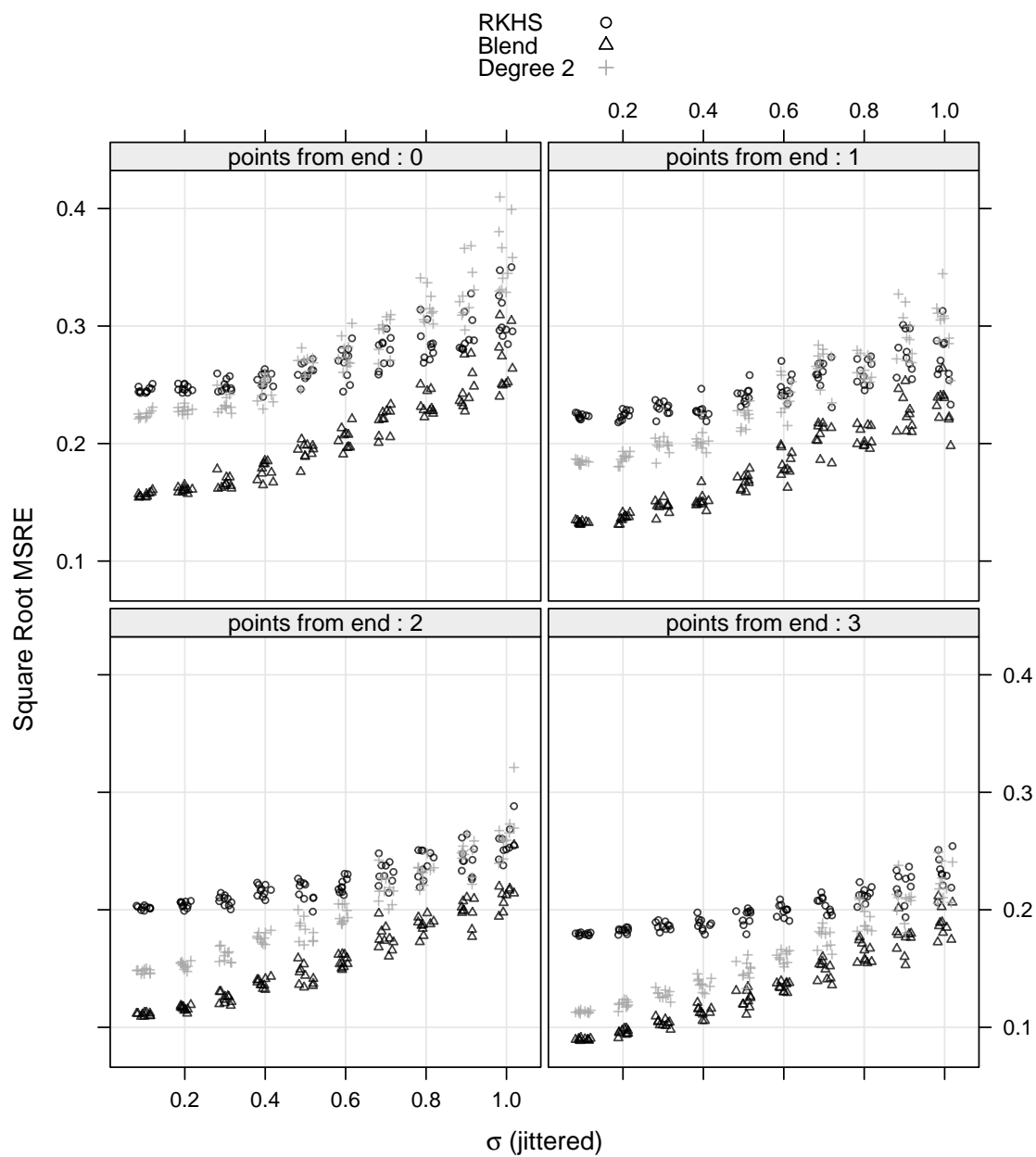


Figure 5.55. $\sqrt{\text{MSRE}}$ vs. σ for RKHS loess, traditional loess and loess blending with $\delta = 0.5$ for the sinusoid data.

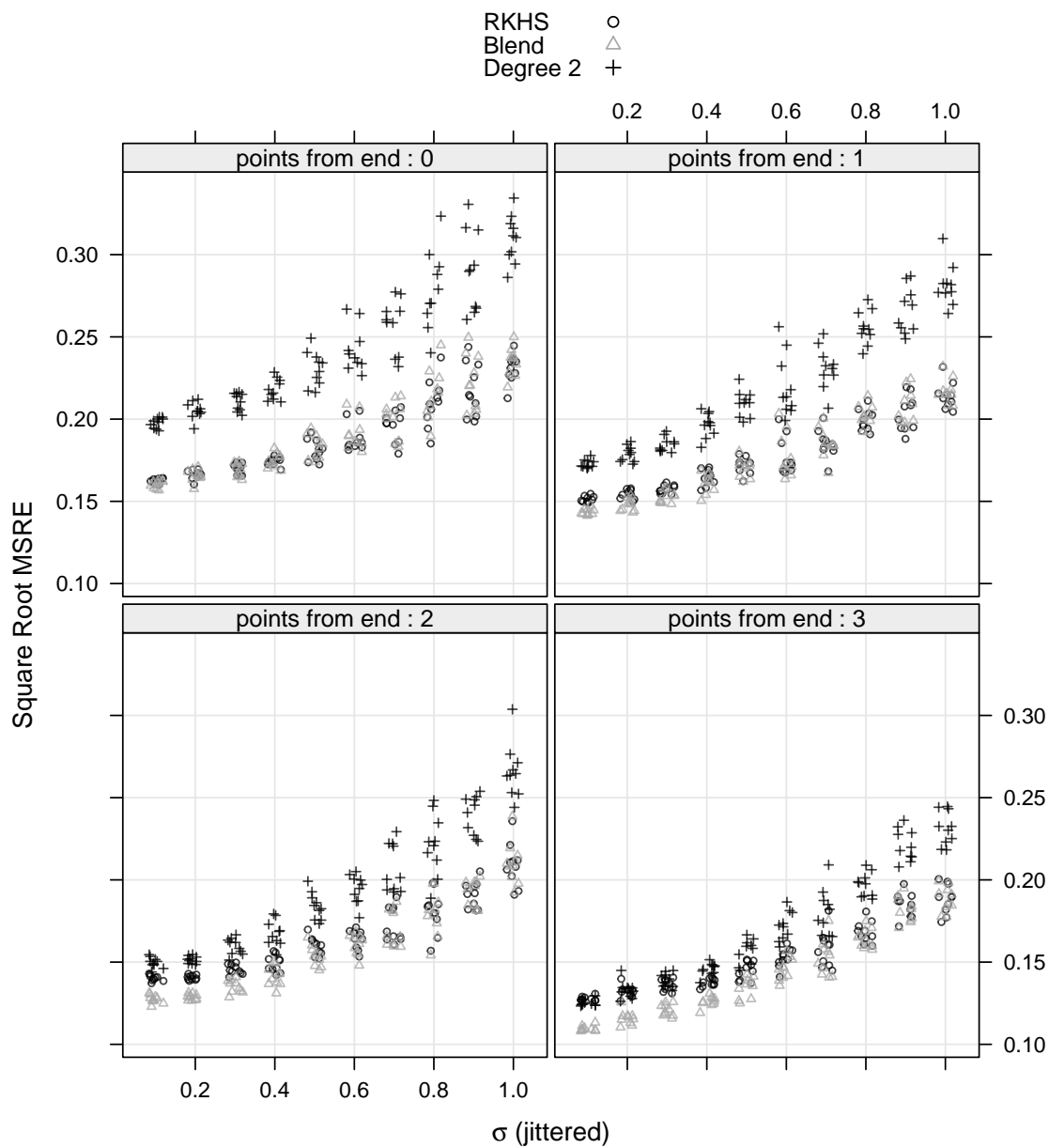


Figure 5.56. $\sqrt{\text{MSRE}}$ vs. σ for RKHS loess, traditional loess and loess blending with $\delta = 0.5$ for the ED data.

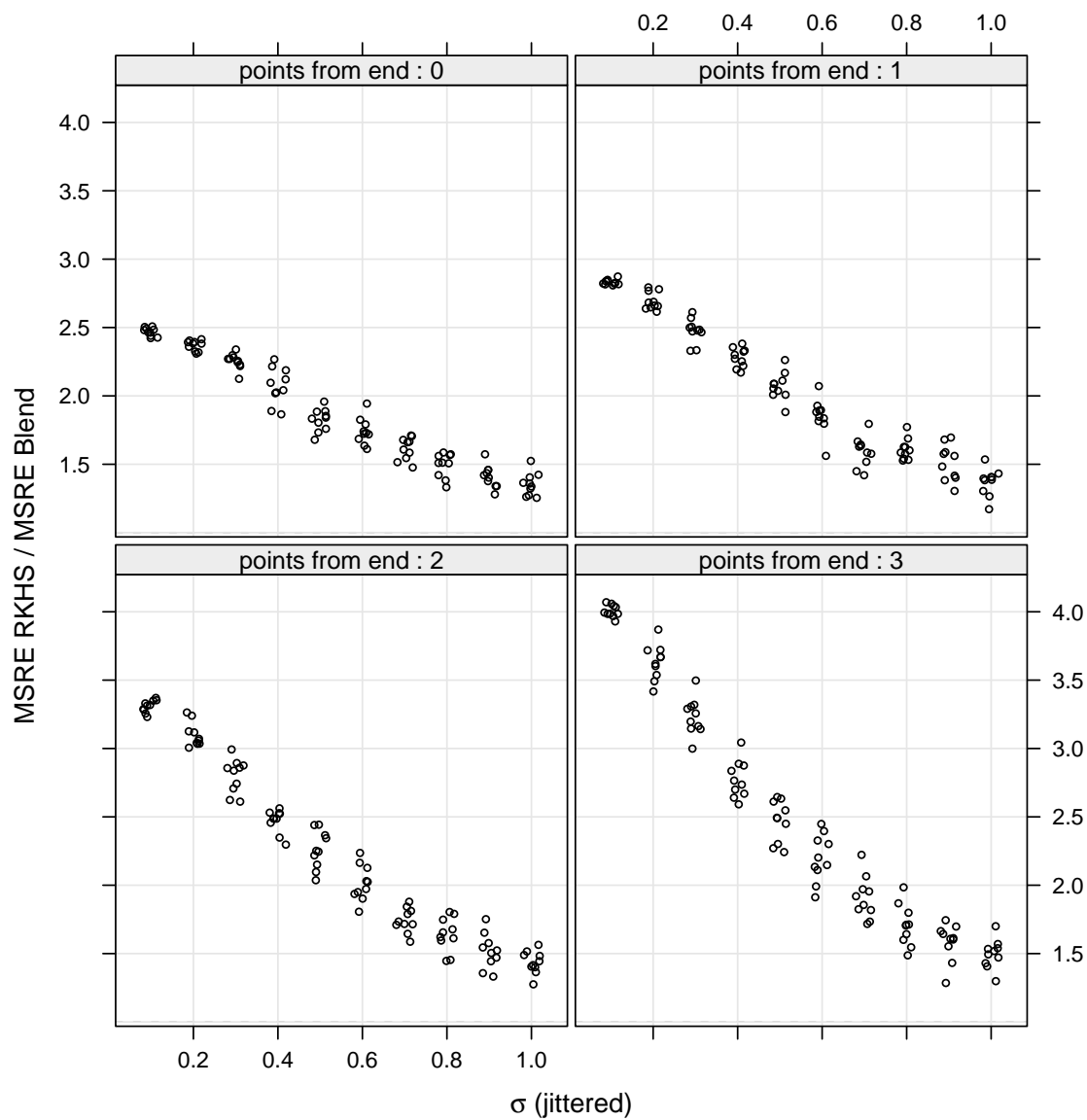


Figure 5.57. MSRE RKHS / MSRE blend vs. σ for sinusoid data.

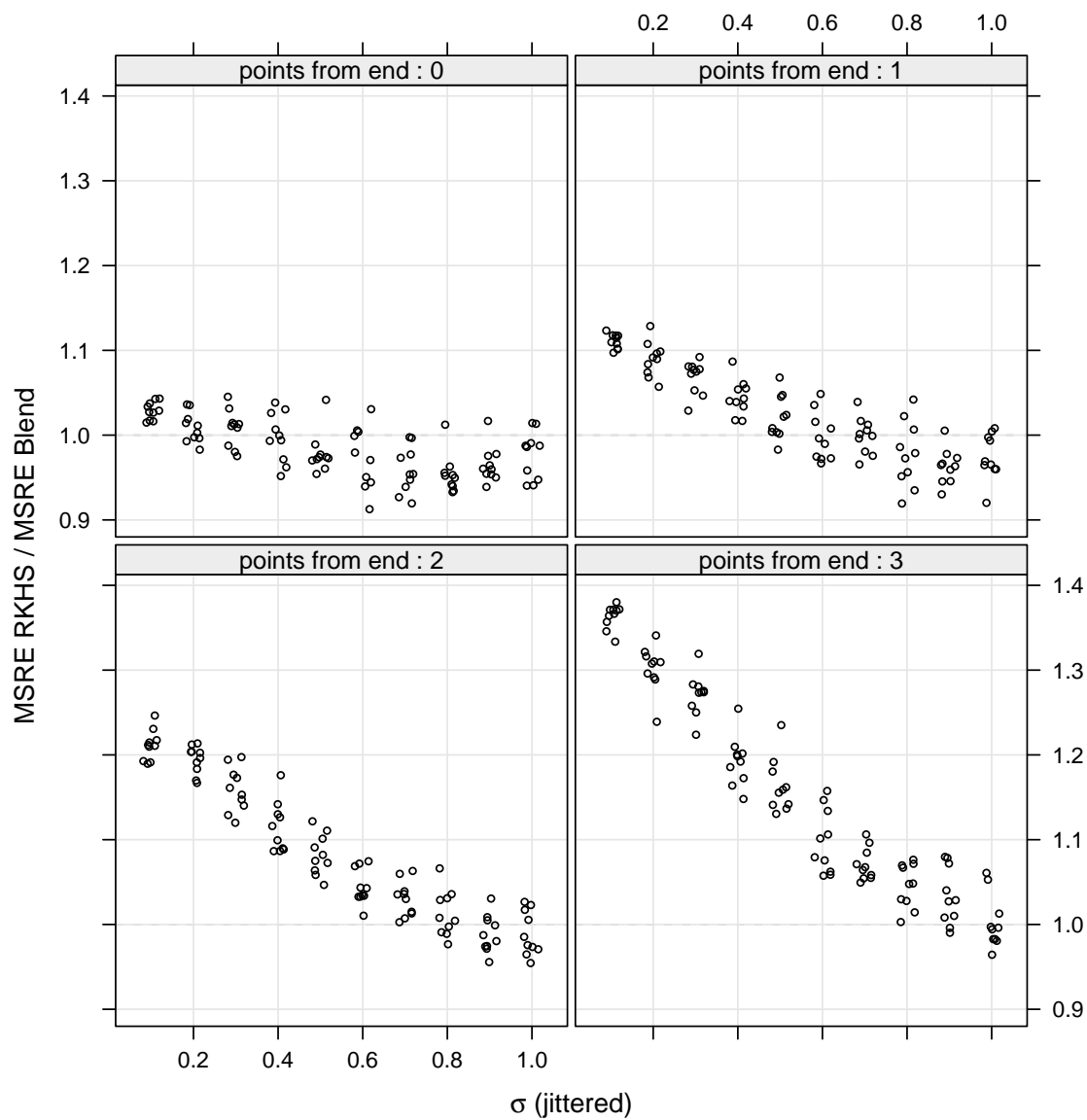


Figure 5.58. MSRE RKHS / MSRE blend vs. σ for ED data.

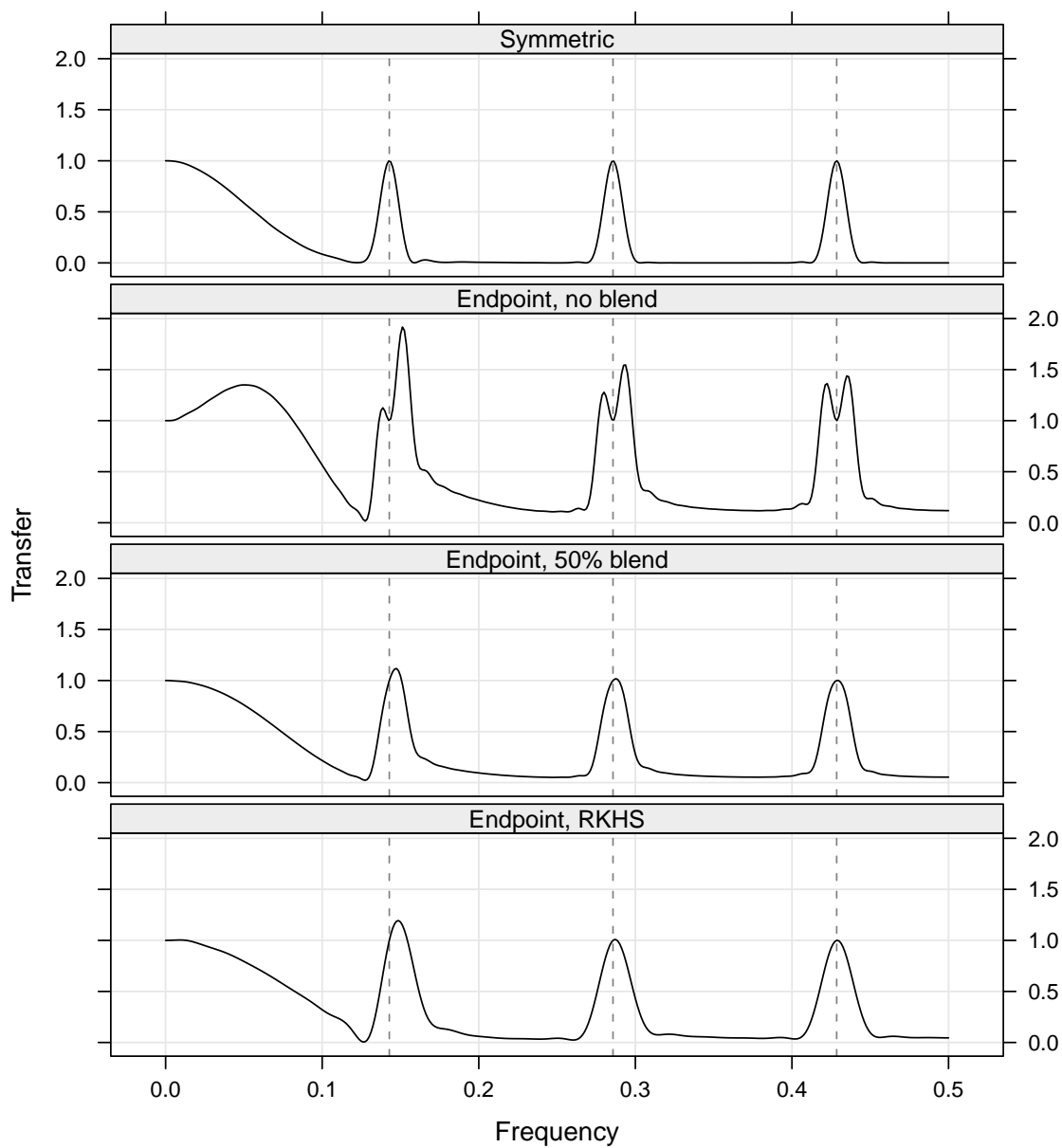


Figure 5.59. Endpoint operator power transfer functions for CO_2 decomposition fit operator for RKHS loess and blending with $\delta = 0$ and $\delta = 0.5$.

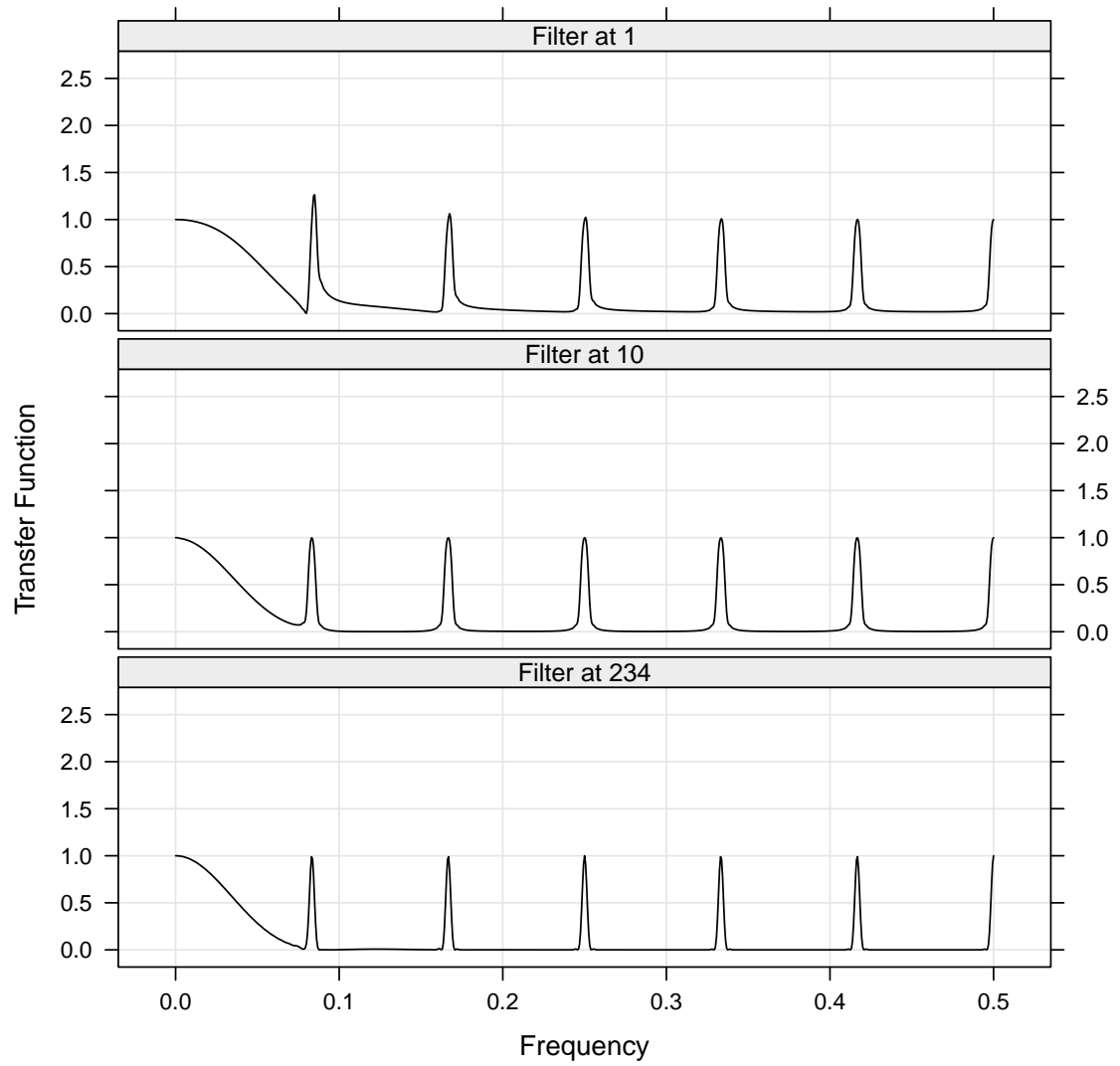


Figure 5.60. Power transfer functions for STL operator matrix L^* with $\delta = 0.5$ blending for seasonal and trend at design points 1 (endpoint), 10, and 234 (middle). Compare to figure 5.28

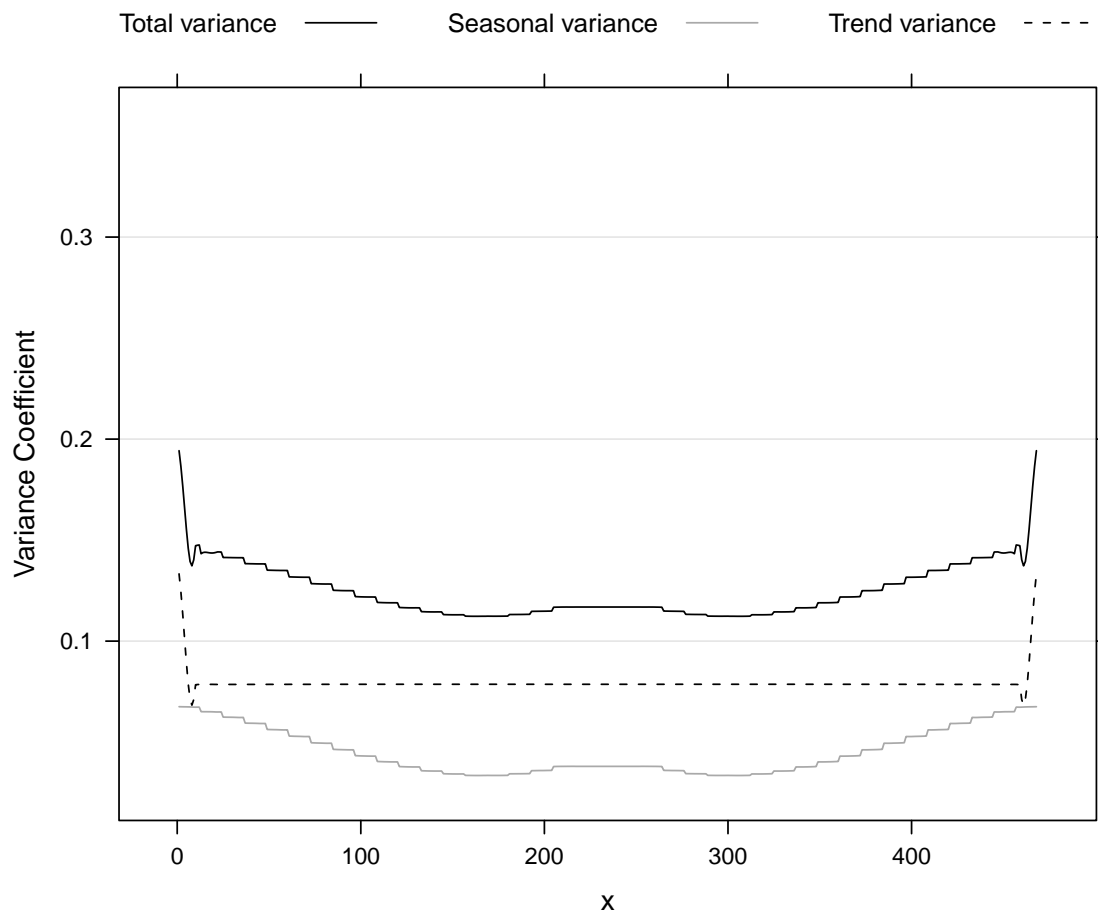


Figure 5.61. Variance coefficients for CO_2 decomposition components with seasonal and trend blending with $\delta = 0.5$ Compare to figure 5.27.

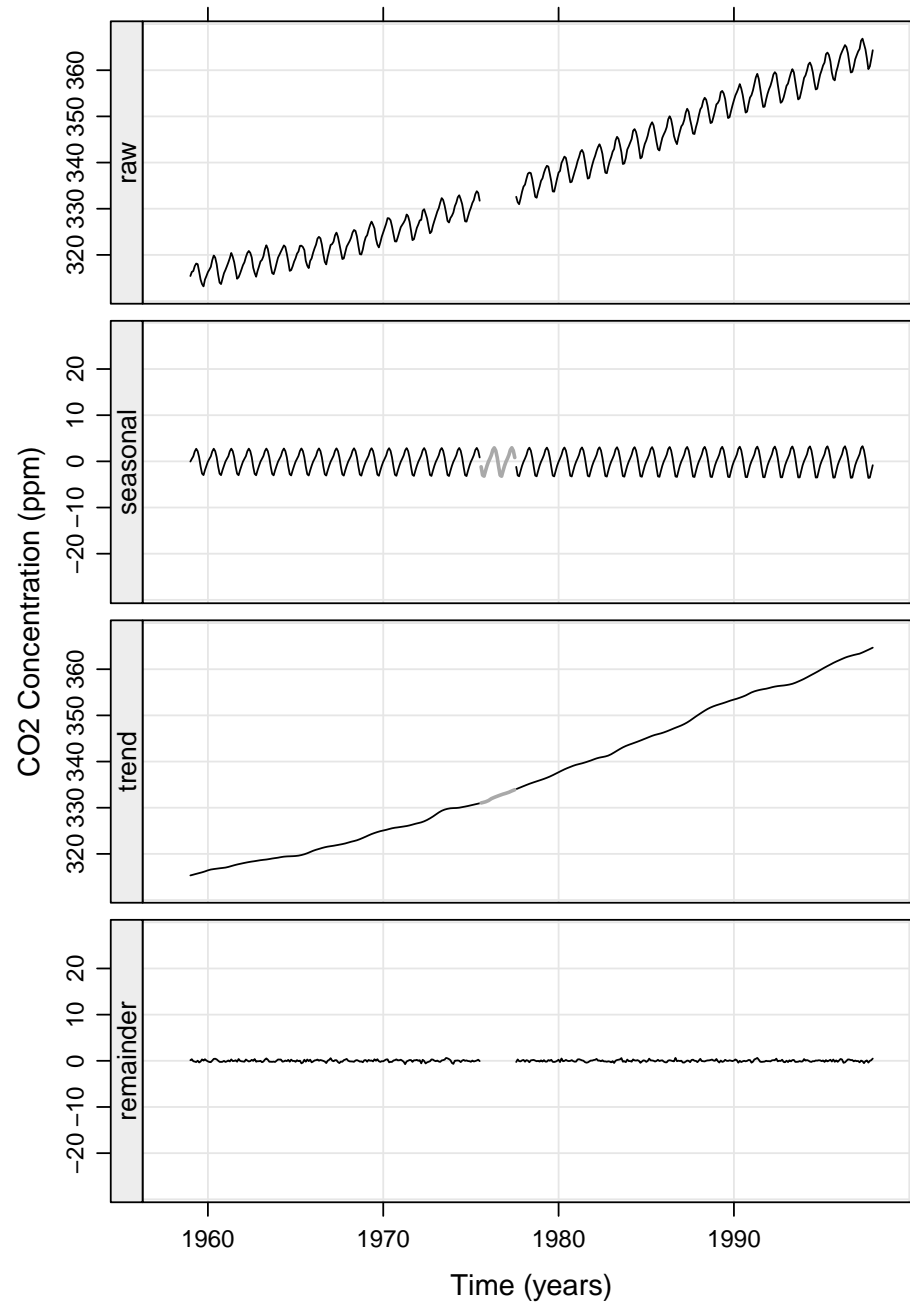


Figure 5.62. STL decomposition for monthly CO₂ concentration measurements with two years of data missing. The gray line in the seasonal and trend components are the missing-data estimates.

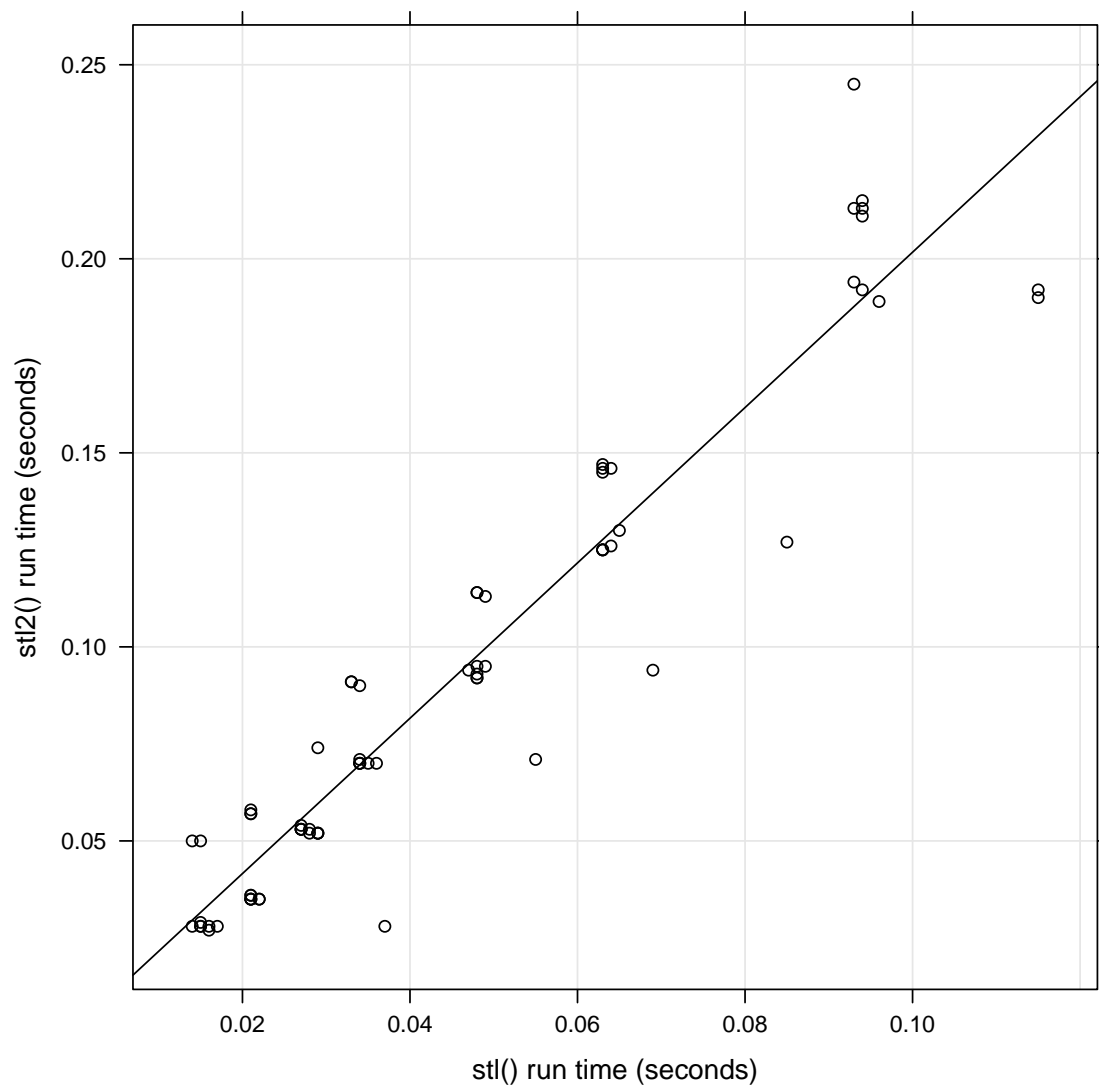


Figure 5.63. Run time for `stl2()` vs. run time for `stl()` for sample sizes ranging from 2000 to 15000, with 10 replicates at each sample.

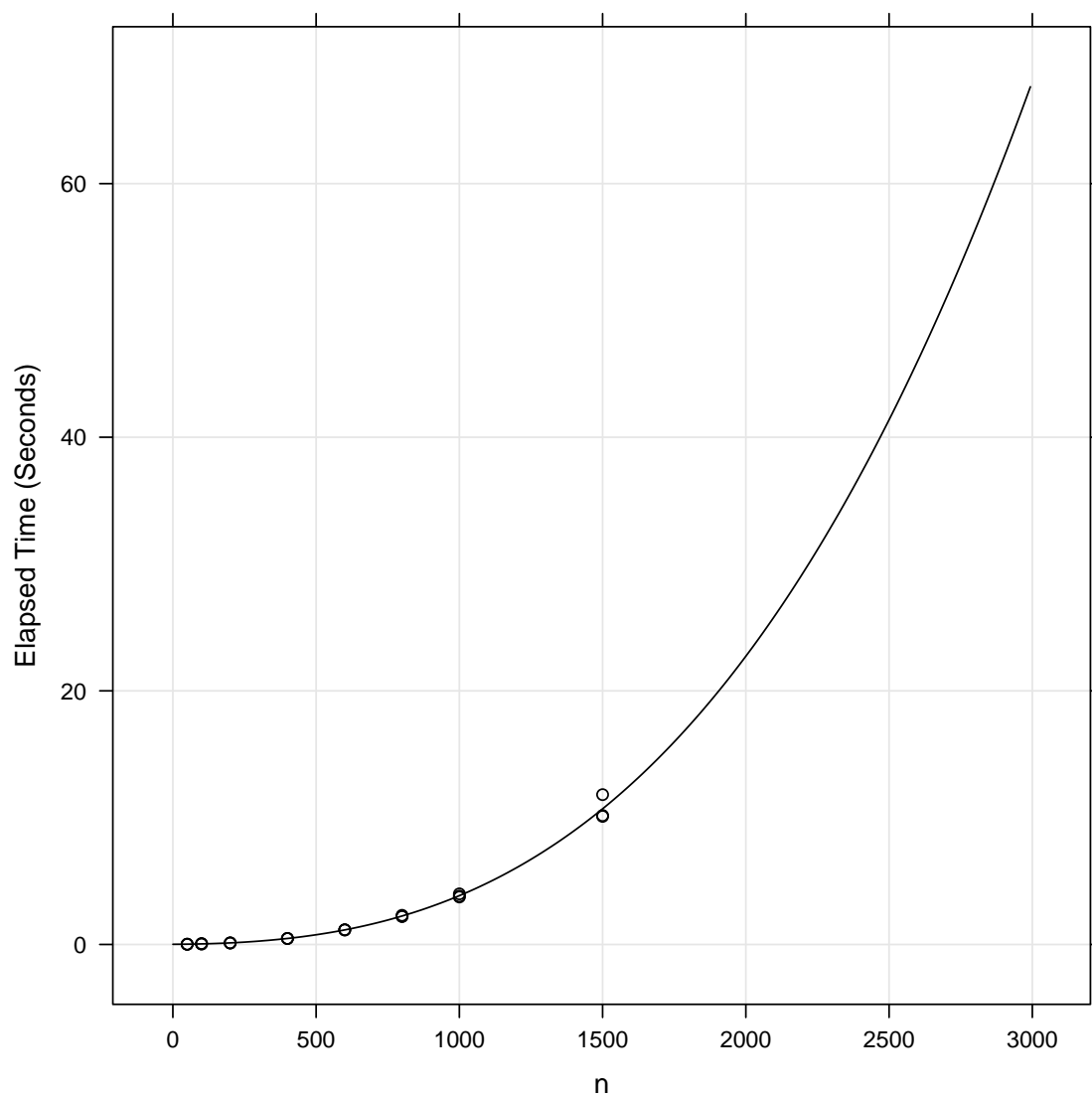


Figure 5.64. Run time vs. sample size for the `st10p()` function, with a fitted cubic polynomial. There are 3 replicates per sample size.

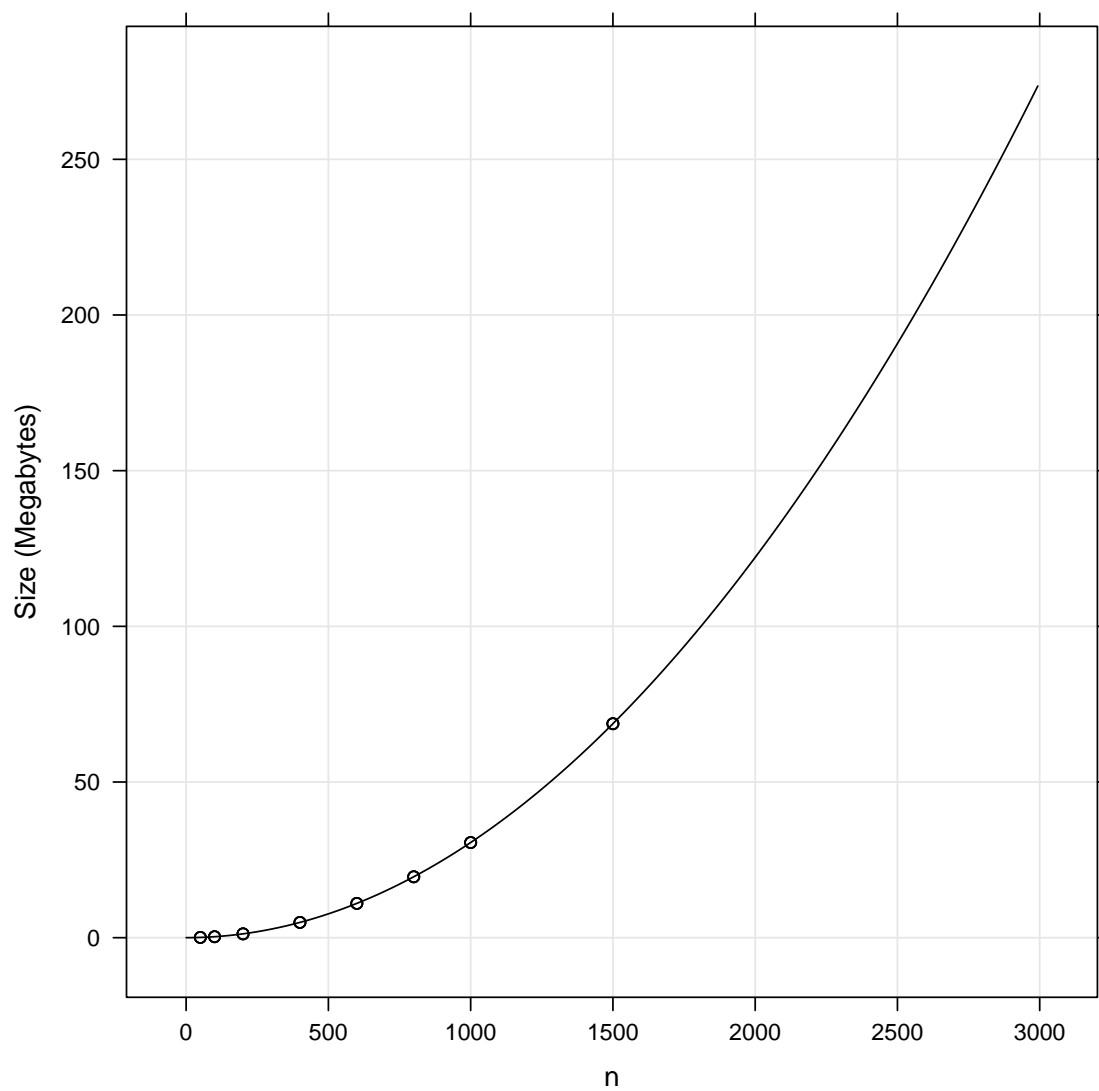


Figure 5.65. Object storage size vs. sample size for the `st10p()` function, with a fitted quadratic polynomial.

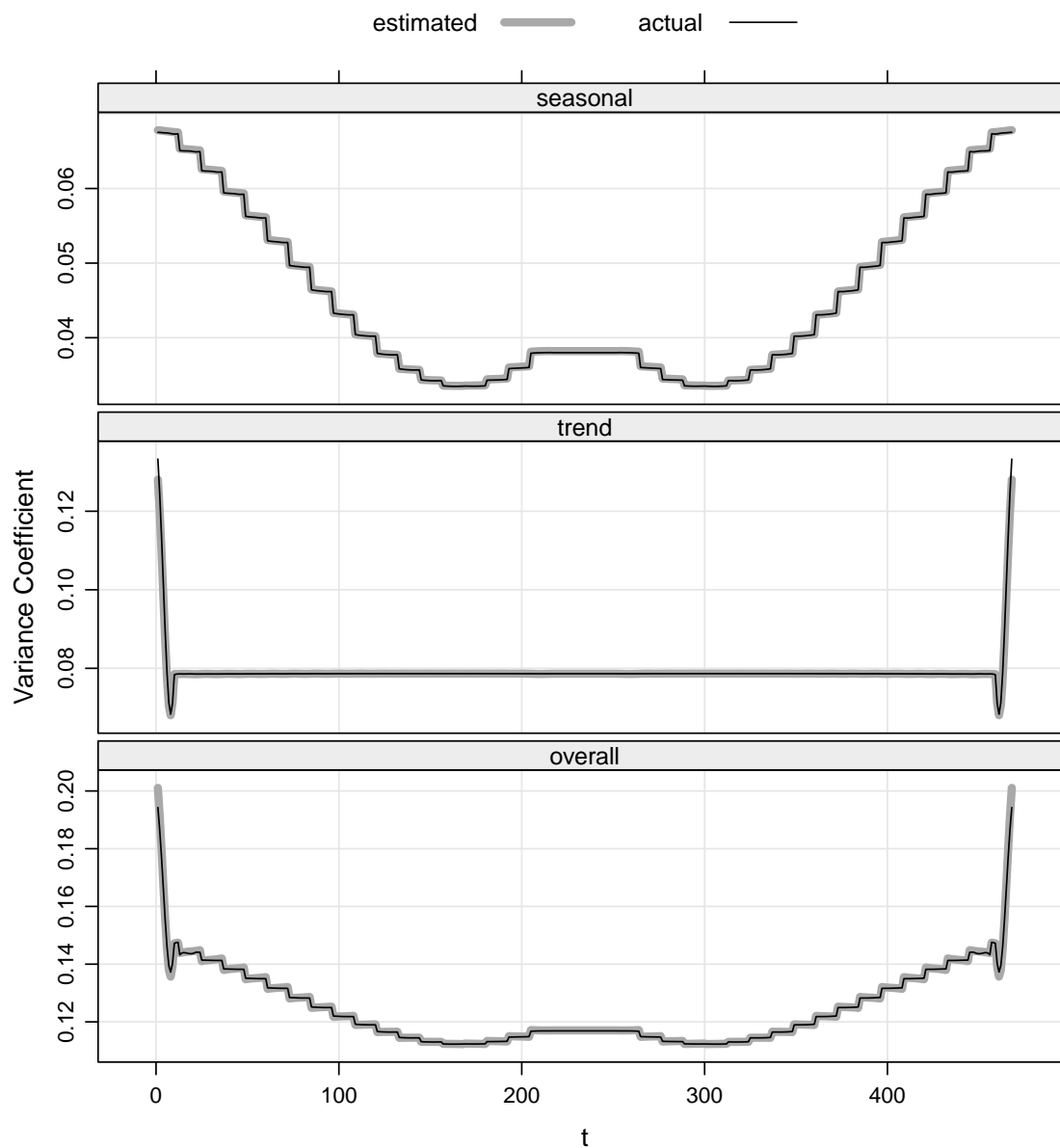


Figure 5.66. Variance coefficient for CO₂ approximate and actual STL decomposition components.

5.4 Figures for Syndromic Surveillance

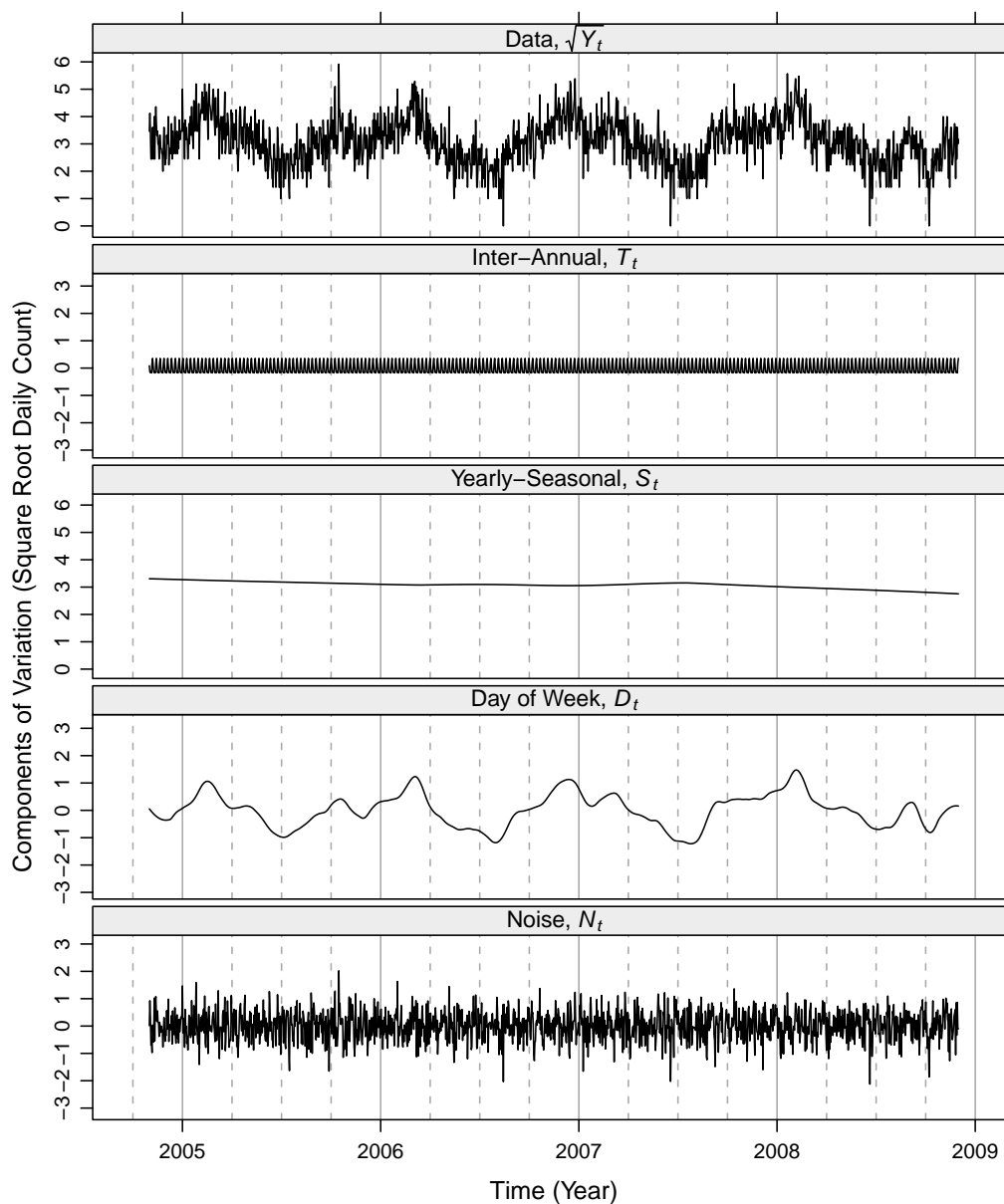


Figure 5.67. Respiratory square root daily counts (centered around 0 on the plot) and four components of variation of the STL decomposition for an Indiana emergency department (ED). The four components sum to the square root counts. The solid vertical lines show January 1 and the dashed vertical lines show April 1, July 1, and October 1.

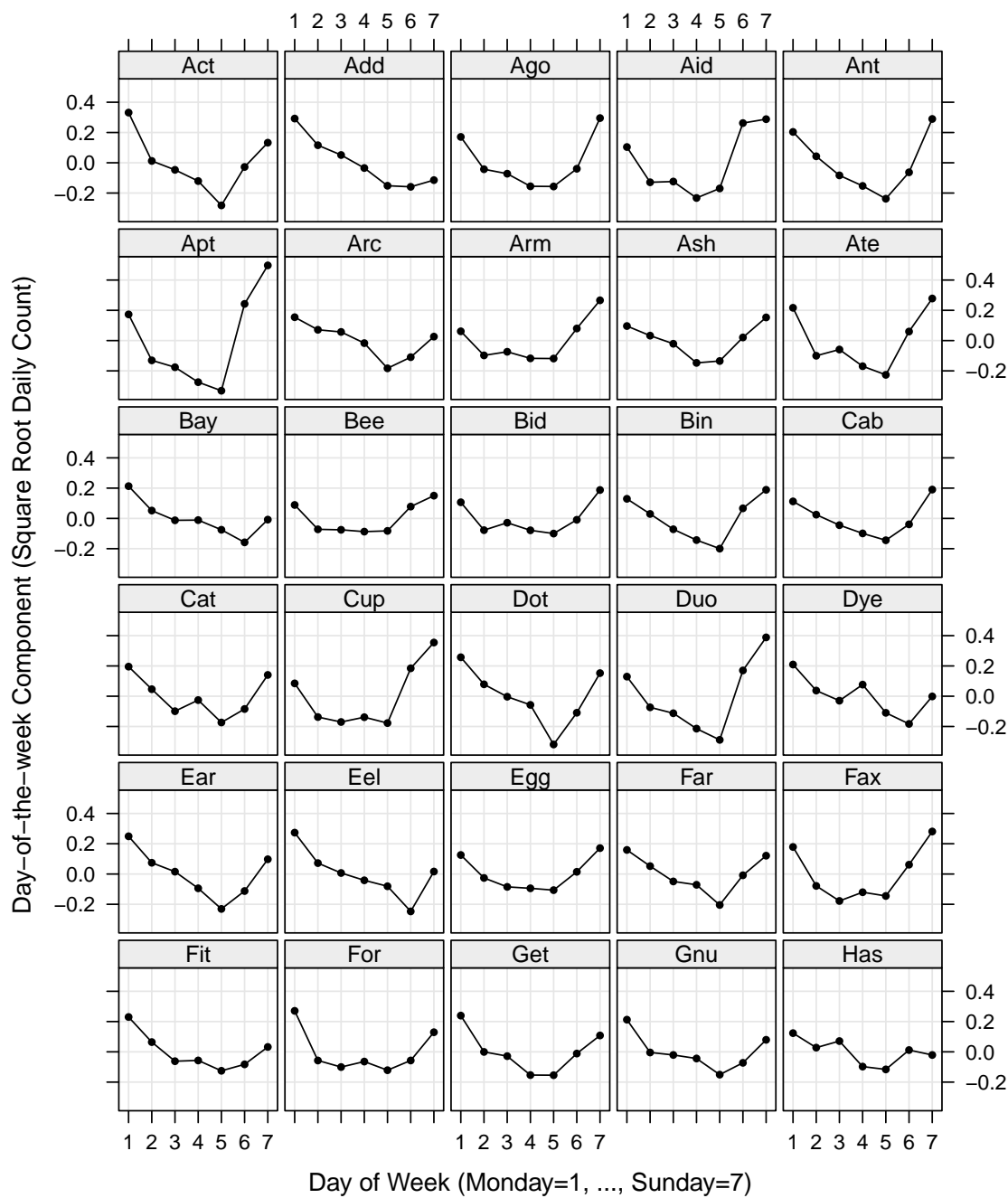


Figure 5.68. Day-of-the-week component, d_t , for square root respiratory counts for 30 Indiana EDs.

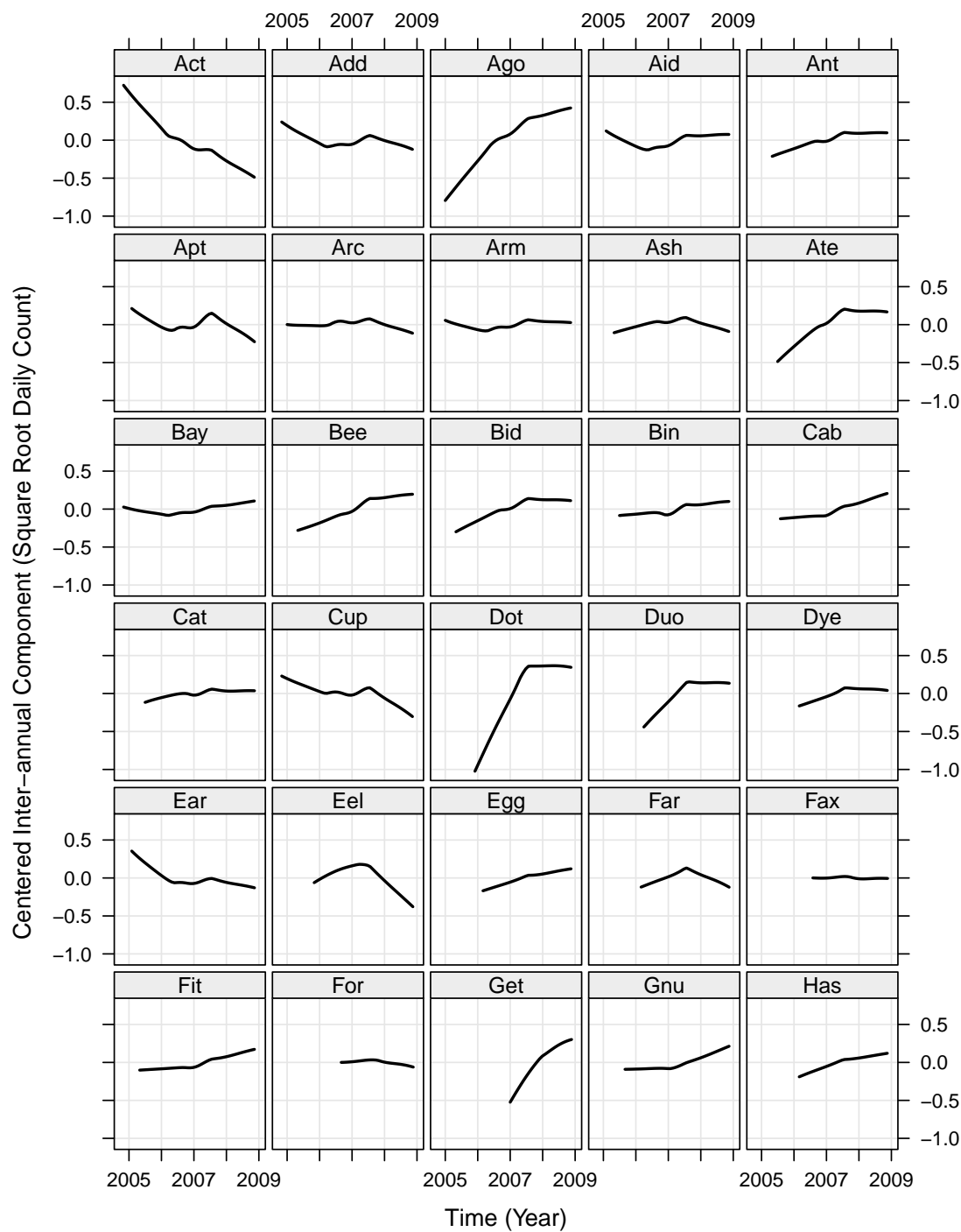


Figure 5.69. Inter-annual component, t_t , for respiratory square root counts for 30 Indiana EDs. Each component has been centered around zero to protect anonymity.

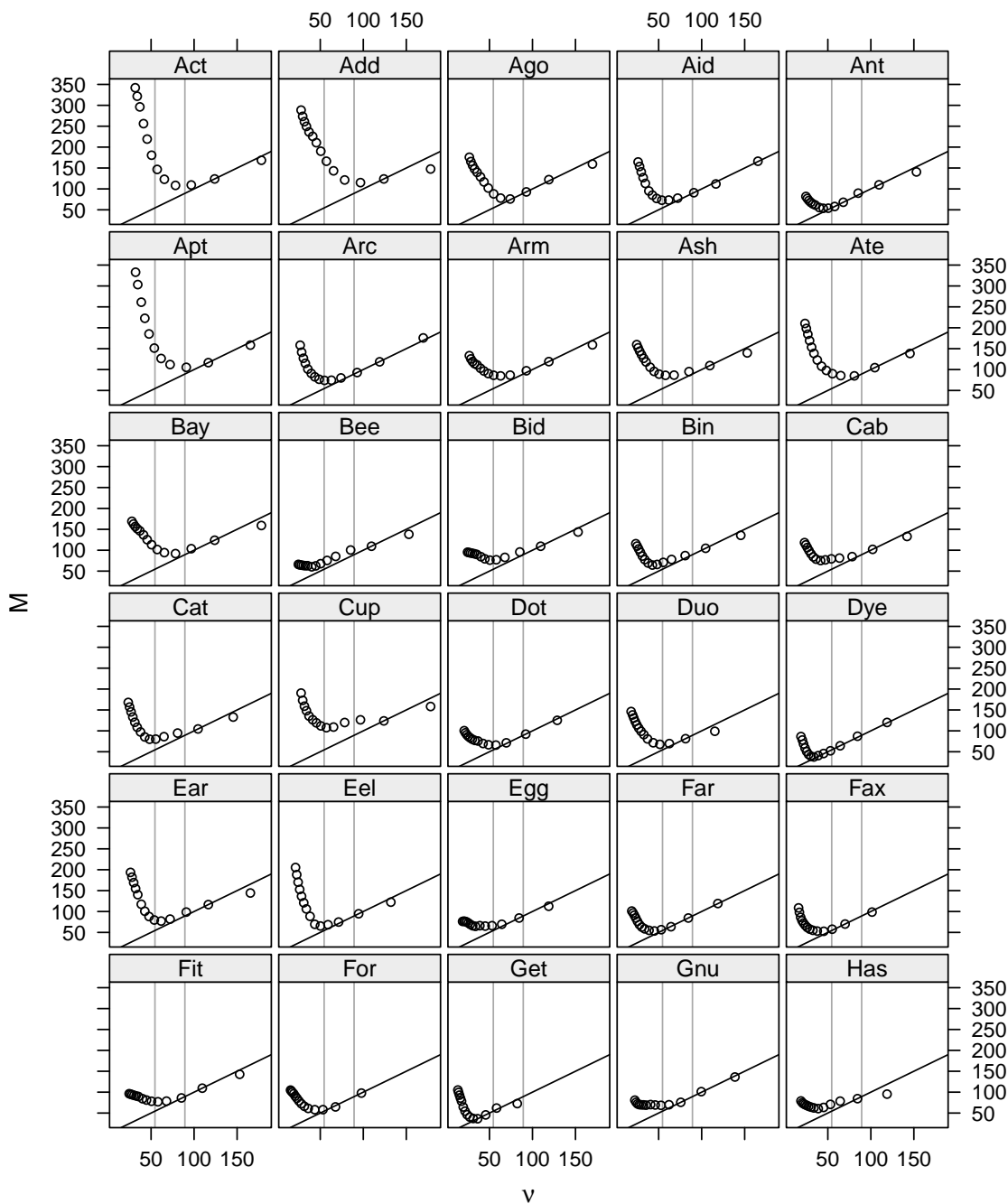


Figure 5.70. C_p plots for yearly-seasonal smoothing parameter for local quadratic fit to $\sqrt{y_t} - d_t - t_t$. The vertical line at $\nu = 54.5$ corresponds to $q = 79$ and the line at $\nu = 89.4$ corresponds to $q = 49$.

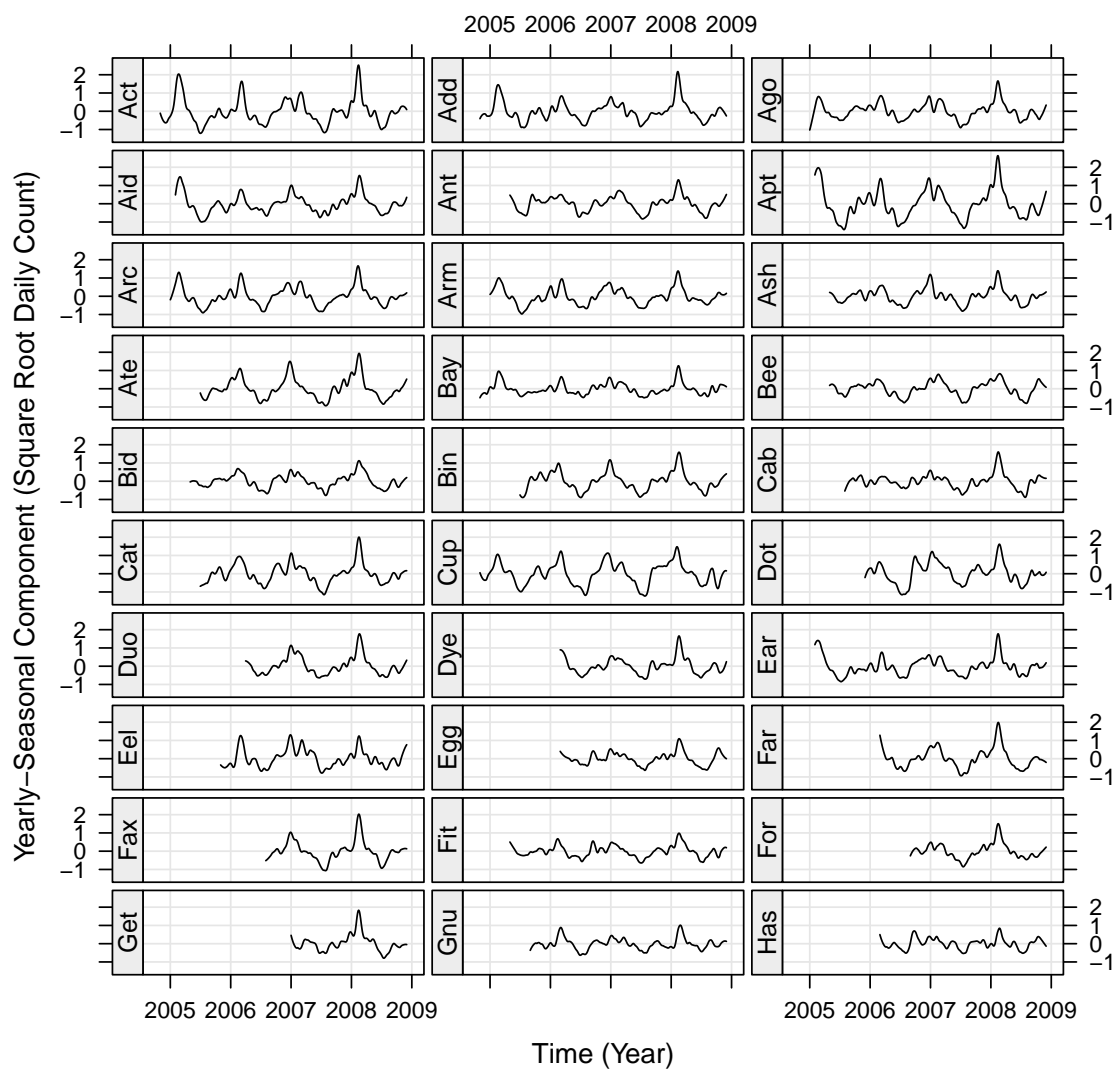


Figure 5.71. Yearly-seasonal component, s_t , for square root respiratory daily counts for 30 Indiana EDs.

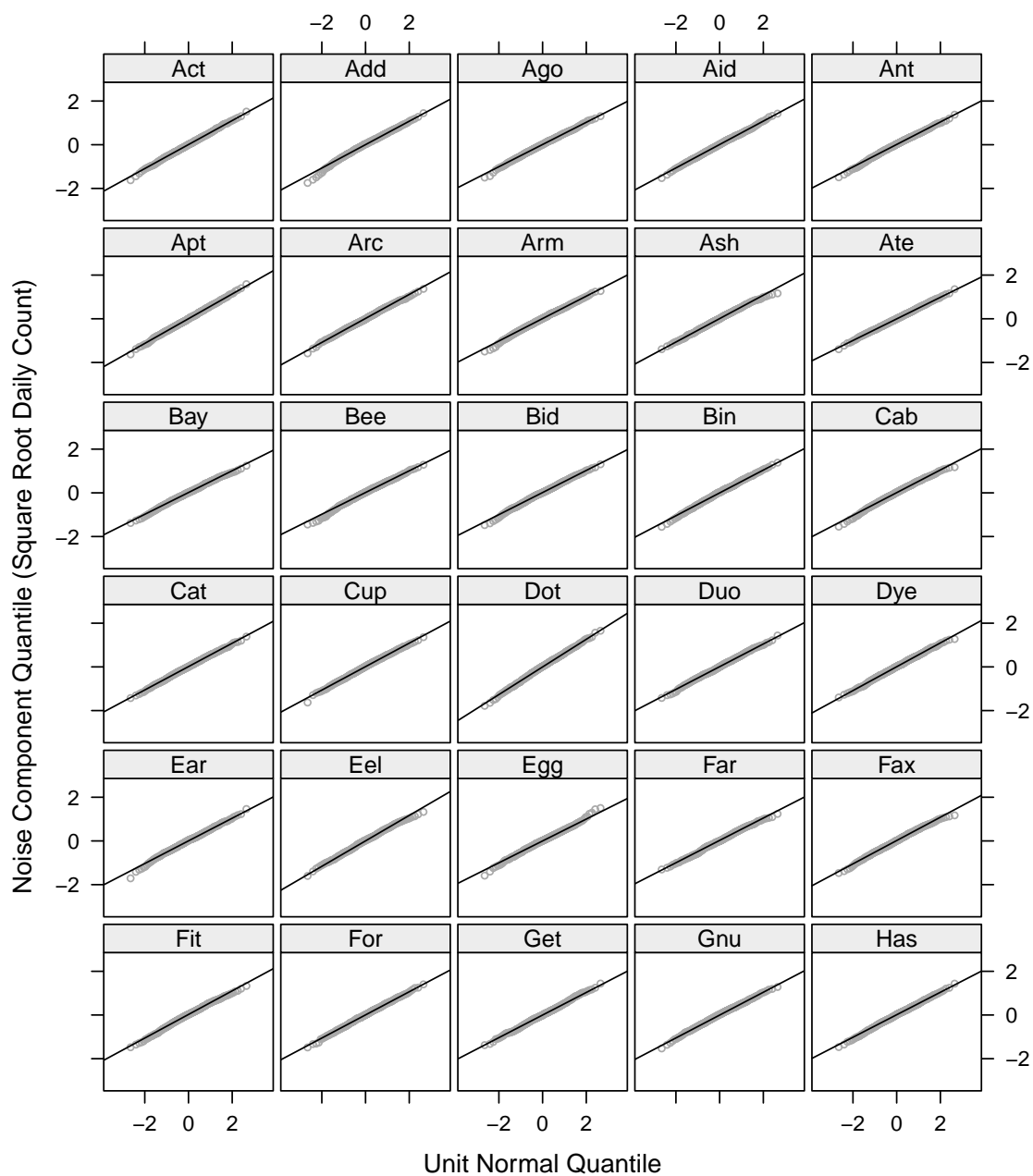


Figure 5.72. Normal quantile plots for the noise component, n_t , of the square root respiratory counts for 30 EDs.

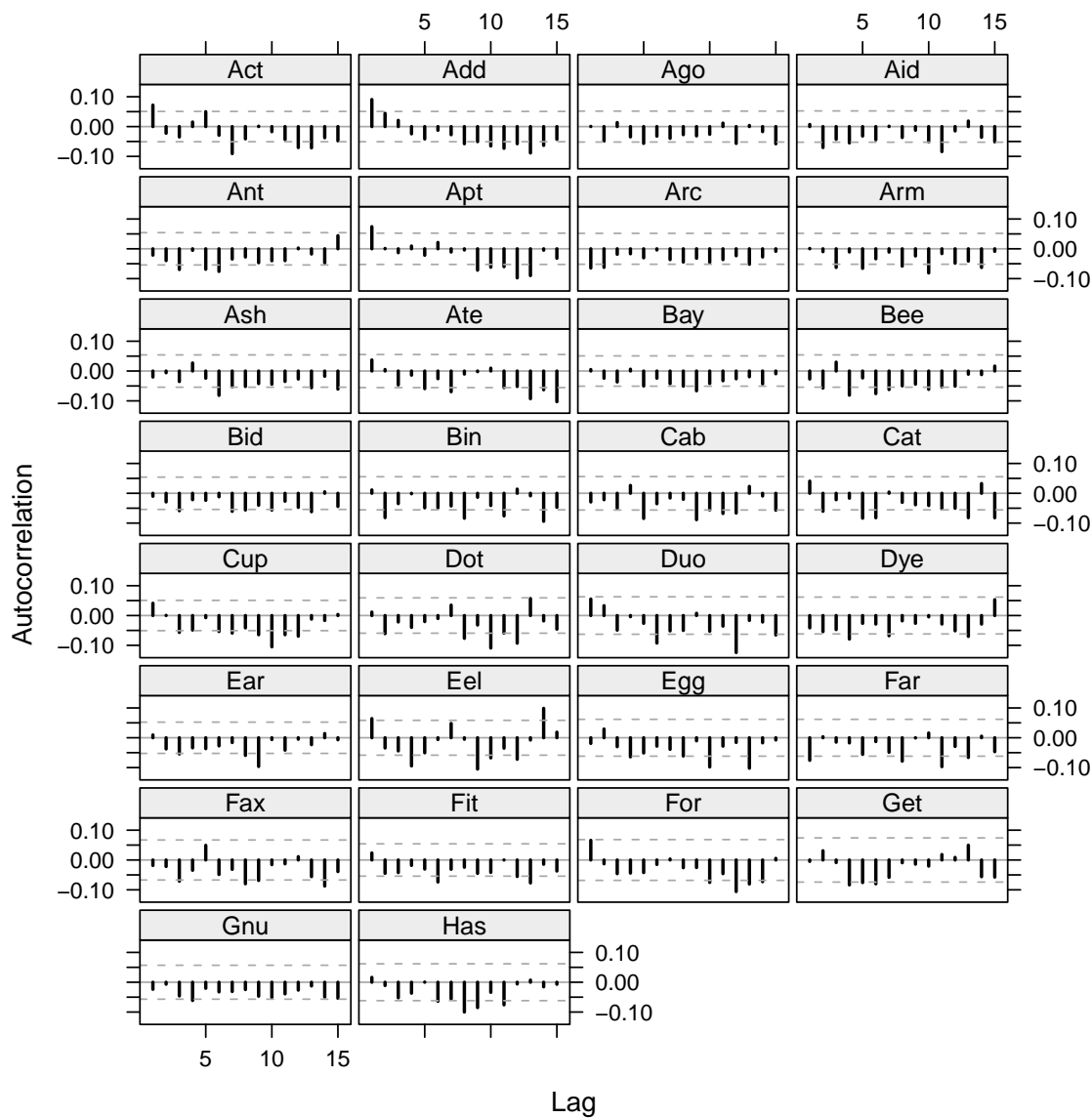


Figure 5.73. Sample autocorrelation functions for the noise component, n_t , of the square root respiratory counts for 30 EDs.

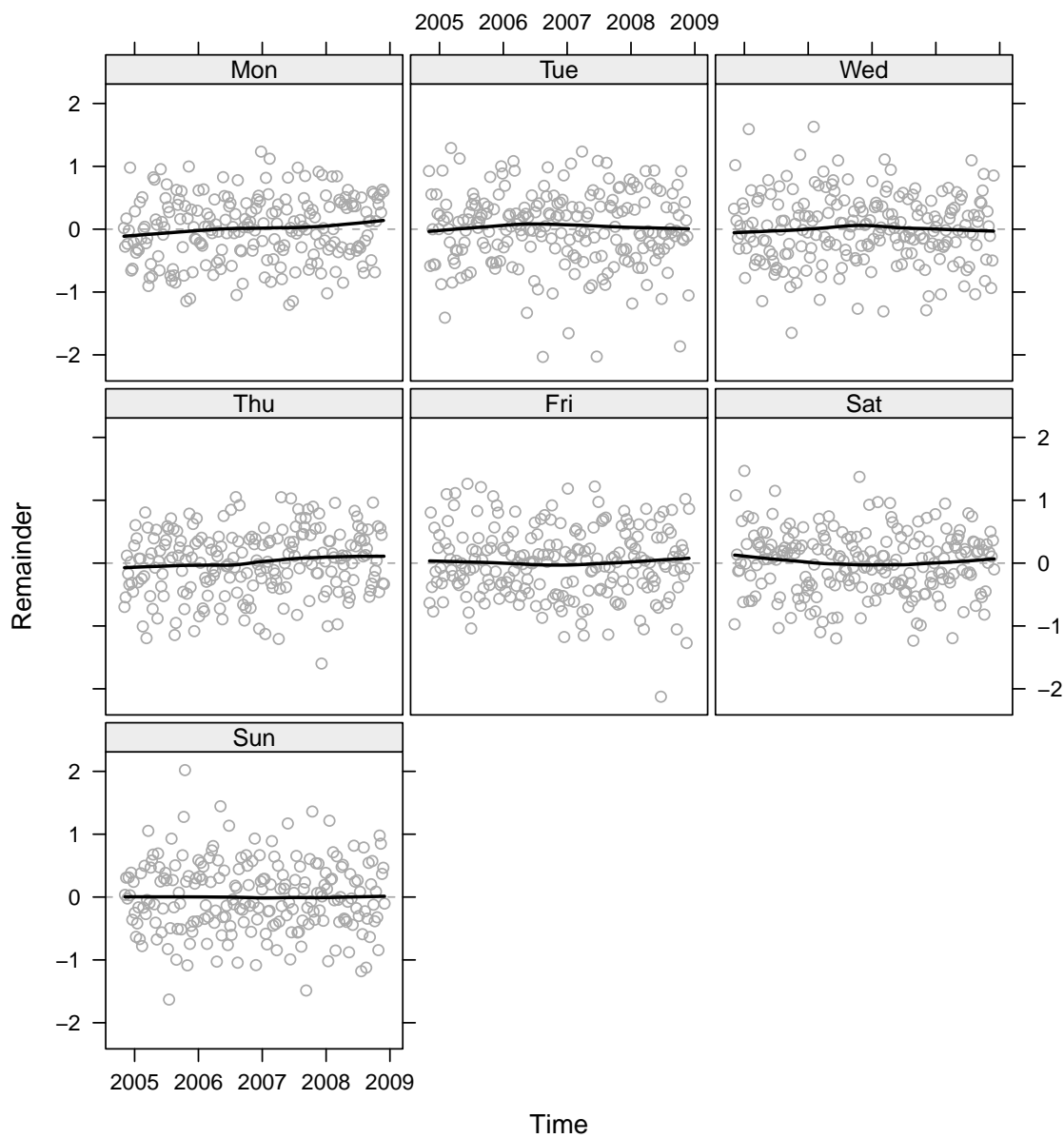


Figure 5.74. Day-of-the-week component diagnostic plot for one ED. The remainder term is plotted with a loess fit.

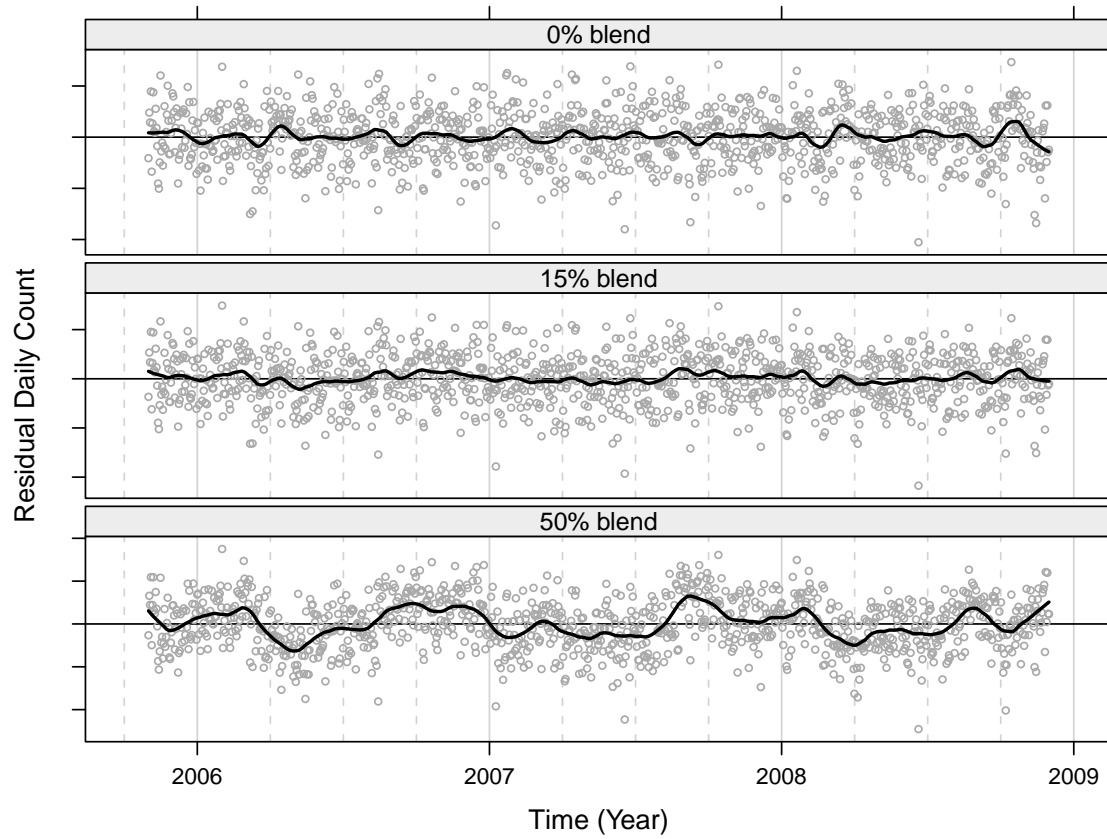


Figure 5.75. Endpoint residuals for one ED from blending s_t and the trend smoothing used to extract d_t with $\delta = 0, 0.15$, and 0.5 . The loess fit highlights deviations from zero.

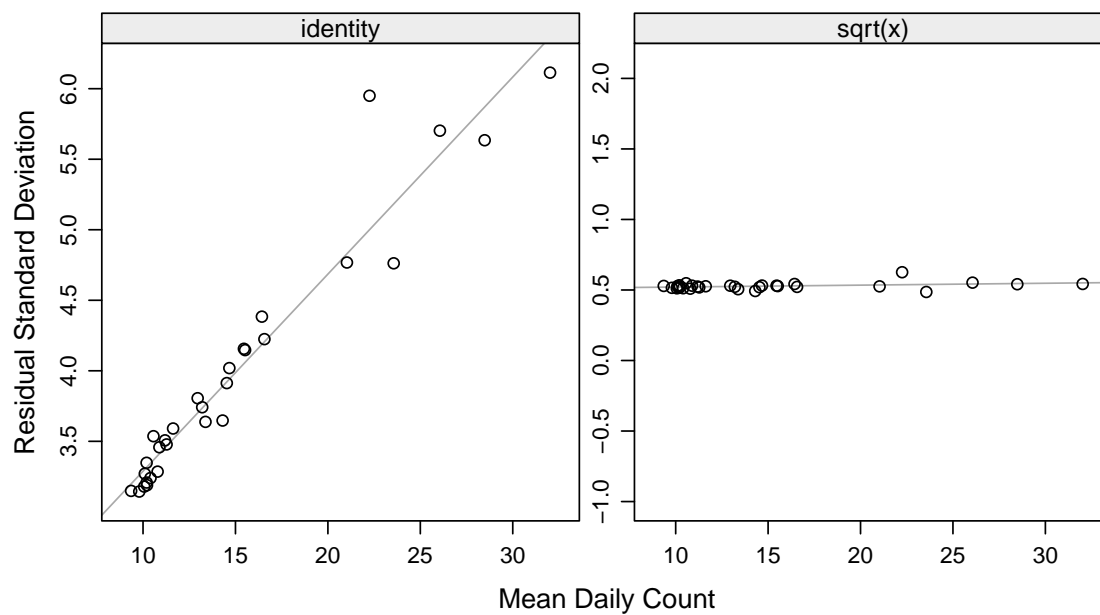


Figure 5.76. Residual standard deviation vs. mean daily count for each of the 30 EDs with and without the square root transformation.

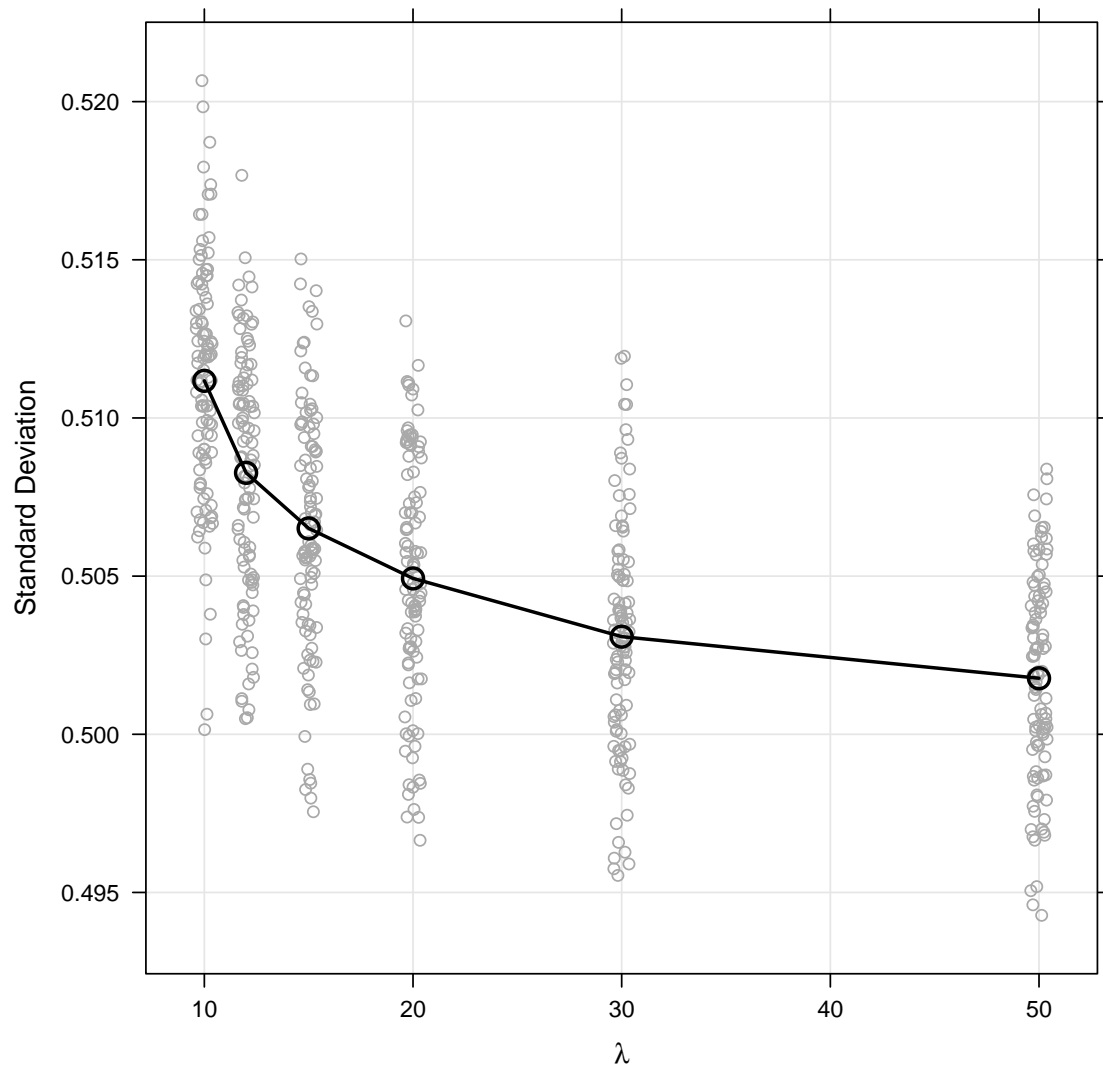


Figure 5.77. Standard deviation for 100 simulated samples of size 10,000 from the square root Poisson distribution for various λ .

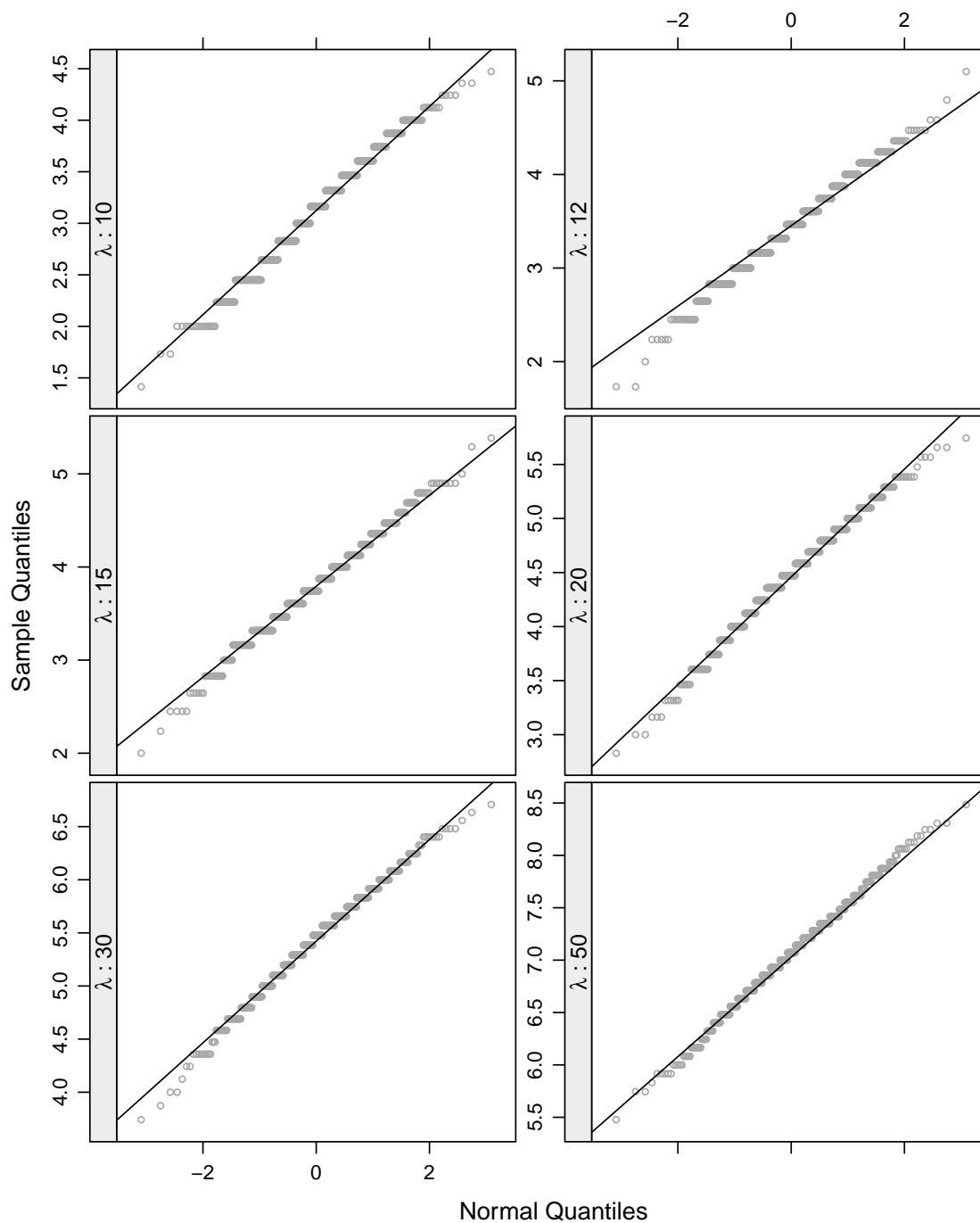


Figure 5.78. Normal quantile plots for samples of size 500 from the square root Poisson distribution for various λ .

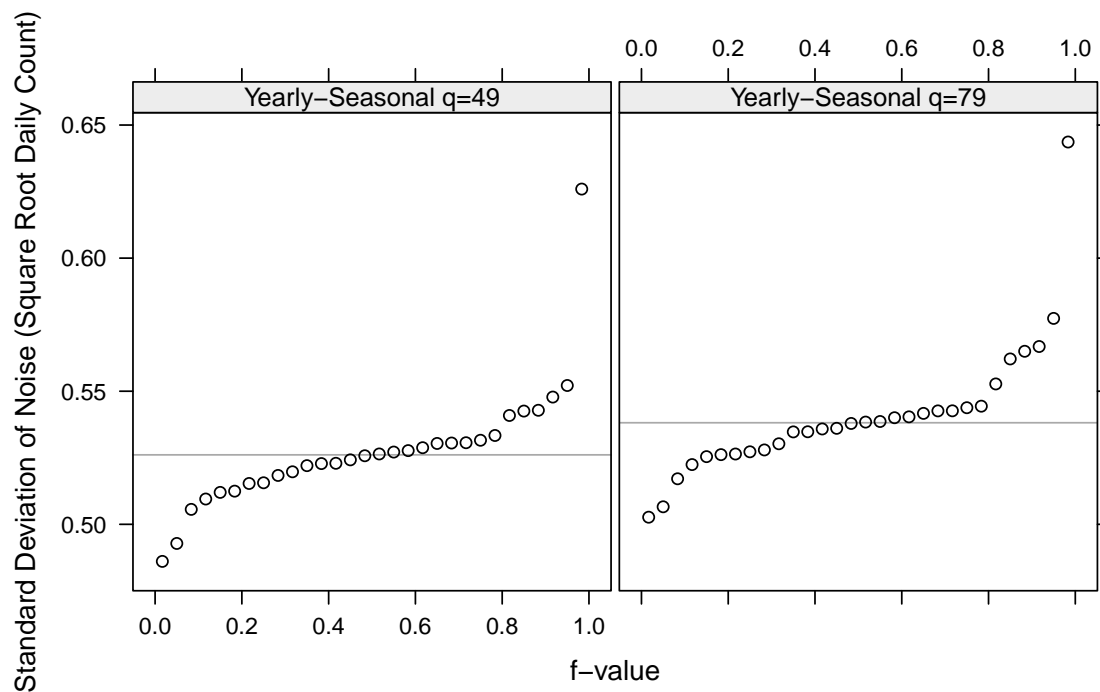


Figure 5.79. Distribution of $\hat{\sigma}_n$ for the 30 EDs.

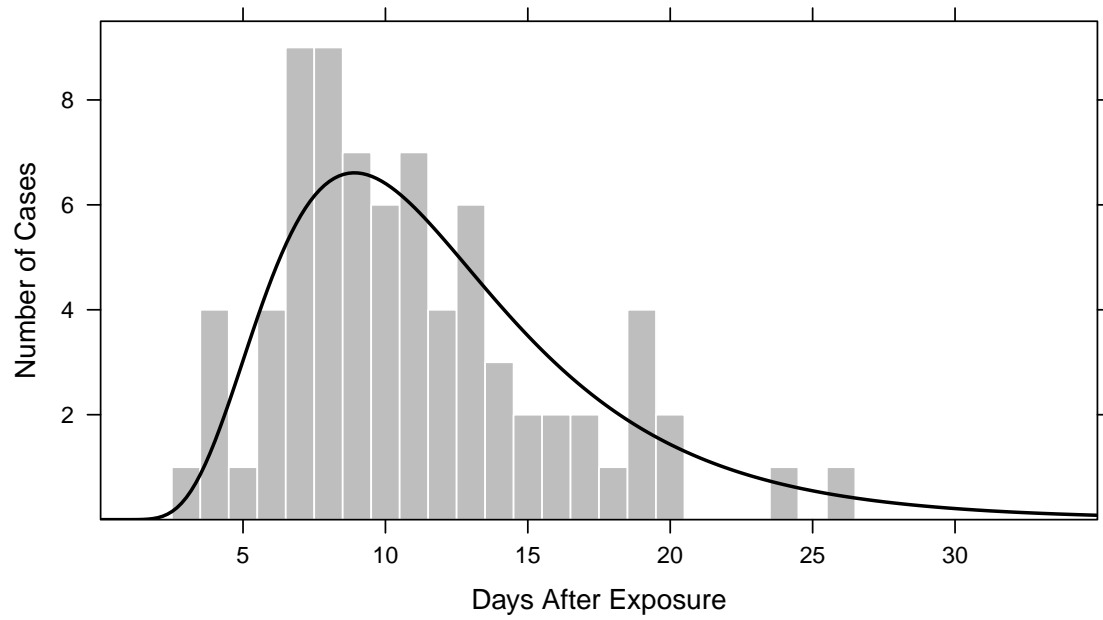


Figure 5.80. Randomly generated outbreak of 76 cases injected according to the Sartwell model [Sartwell \(1950\)](#) is shown by the histogram. The log-normal density of the model multiplied by 76 is shown by the curve. The parameters of the lognormal are the mean, $\zeta = 2.401$, and the standard deviation, $\sigma = 0.4626$, both on the natural log scale.

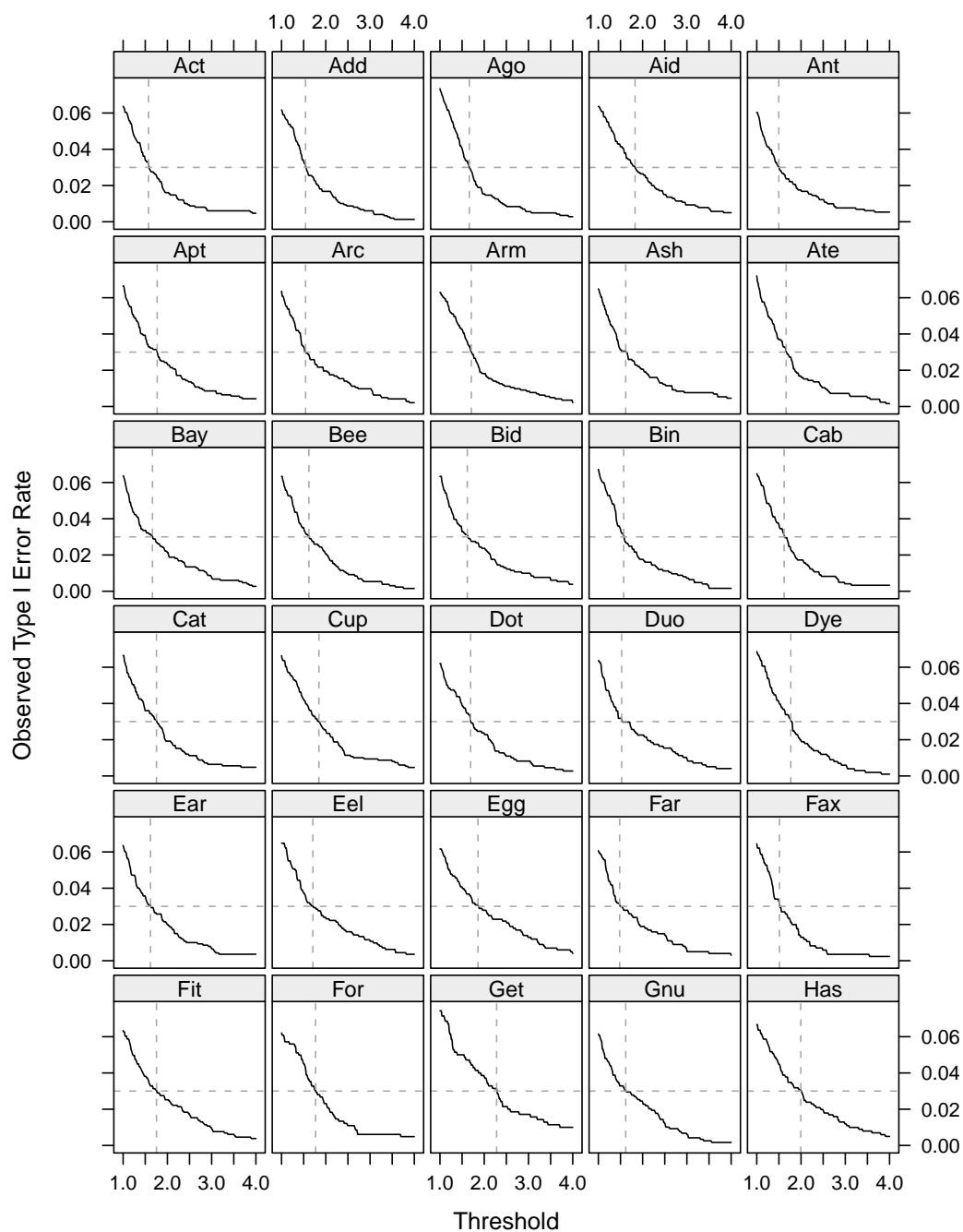


Figure 5.81. Observed proportion of false positives vs. cutoff limit for the C1 method, by ED. The dashed lines correspond to an observed false positive rate of 0.03 (horizontal) and the corresponding empirical cutoff limit, c (vertical).

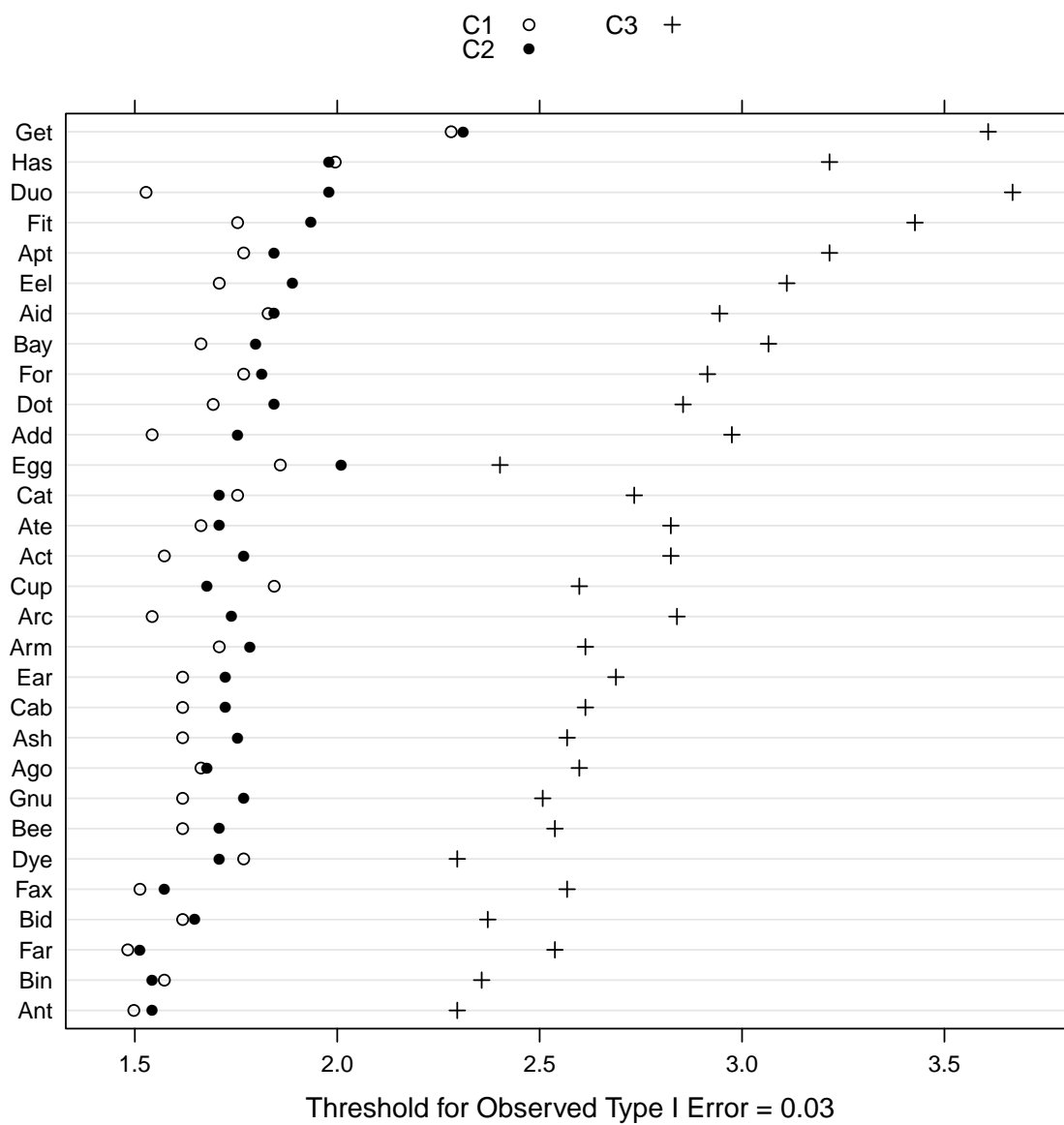


Figure 5.82. EARS C1, C2, and C3 threshold levels for an empirical false positive rate of 0.03.

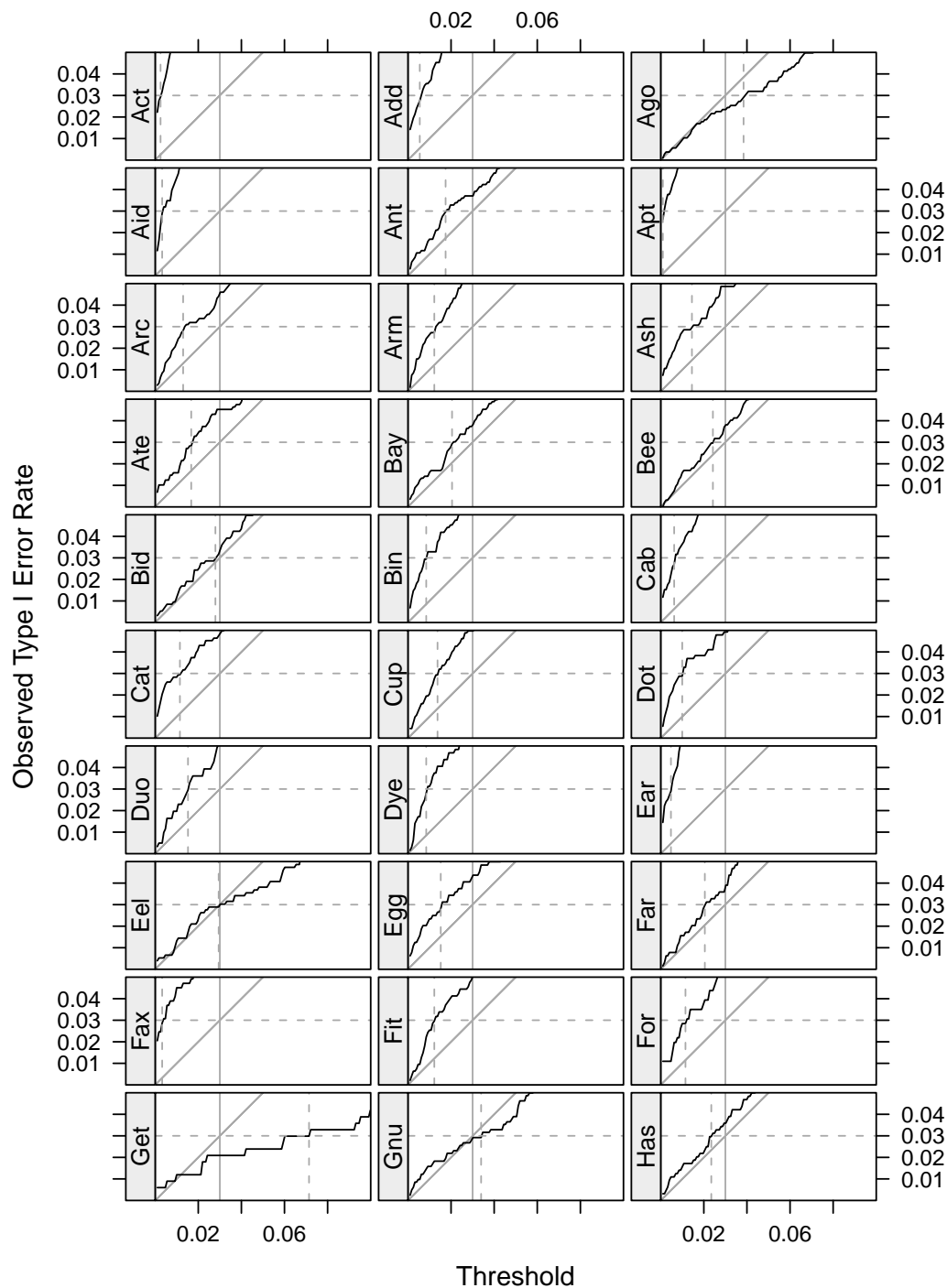


Figure 5.83. Observed false positive rate vs. cutoff limit for the GLM method. The dashed lines correspond to an observed false positive rate of 0.03 (horizontal) and the corresponding empirical cutoff limit, c (vertical). The solid vertical line corresponds to the true false positive rate, $\rho = 0.03$

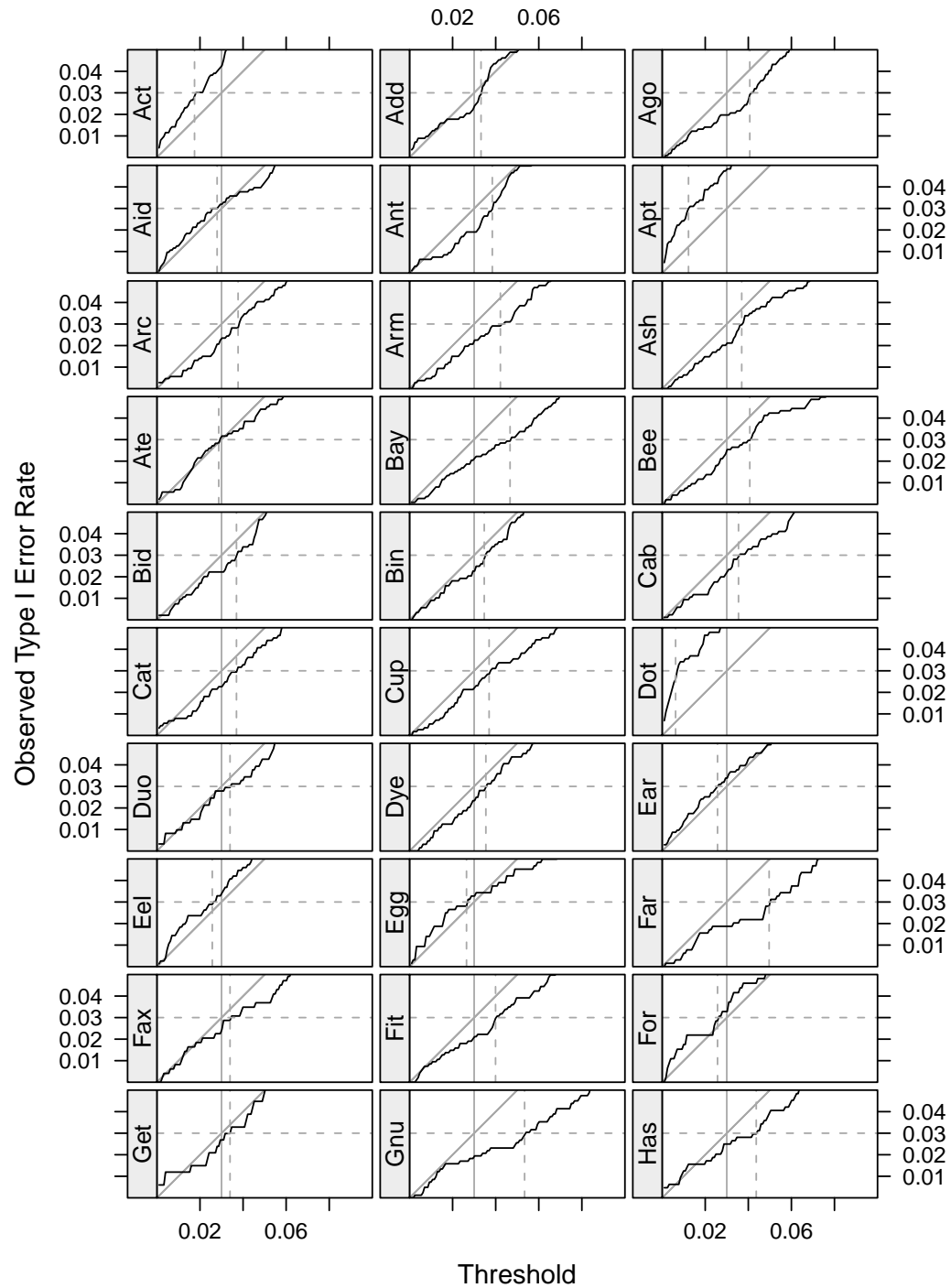


Figure 5.84. Observed false positive rate vs. cutoff limit for the STL $q = 79$ method. The dashed lines correspond to an observed false positive rate of 0.03 (horizontal) and the corresponding empirical cutoff limit, c (vertical). The solid vertical line is the true false positive rate, $\rho = 0.03$.

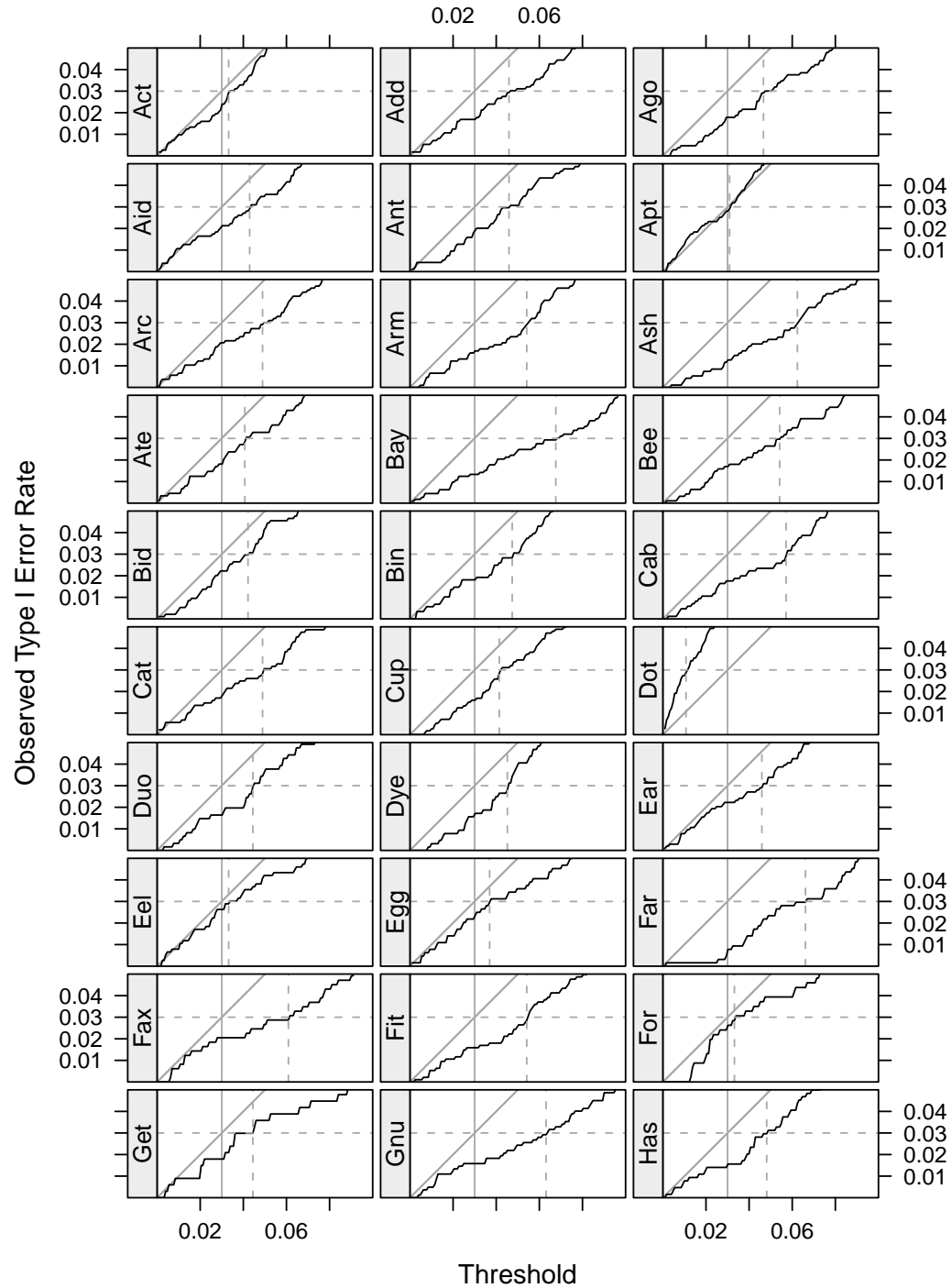


Figure 5.85. Observed false positive rate vs. cutoff limit for the STL $q = 49$ method. The dashed lines correspond to an observed false positive rate of 0.03 (horizontal) and the corresponding empirical cutoff limit, c (vertical). The solid vertical line is the true false positive rate, $\rho = 0.03$

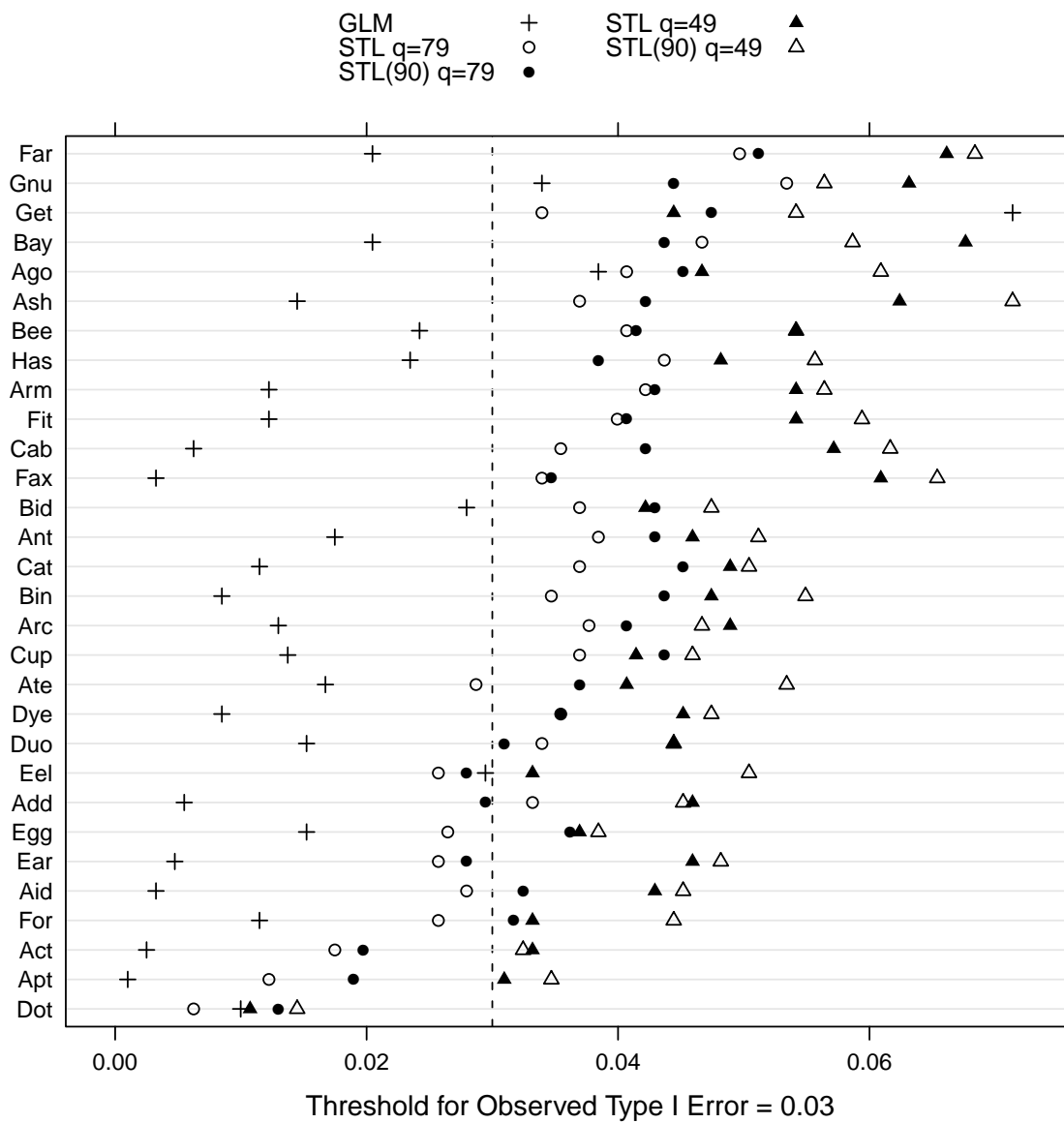


Figure 5.86. GLM and STL threshold levels for an empirical false positive rate of 0.03.

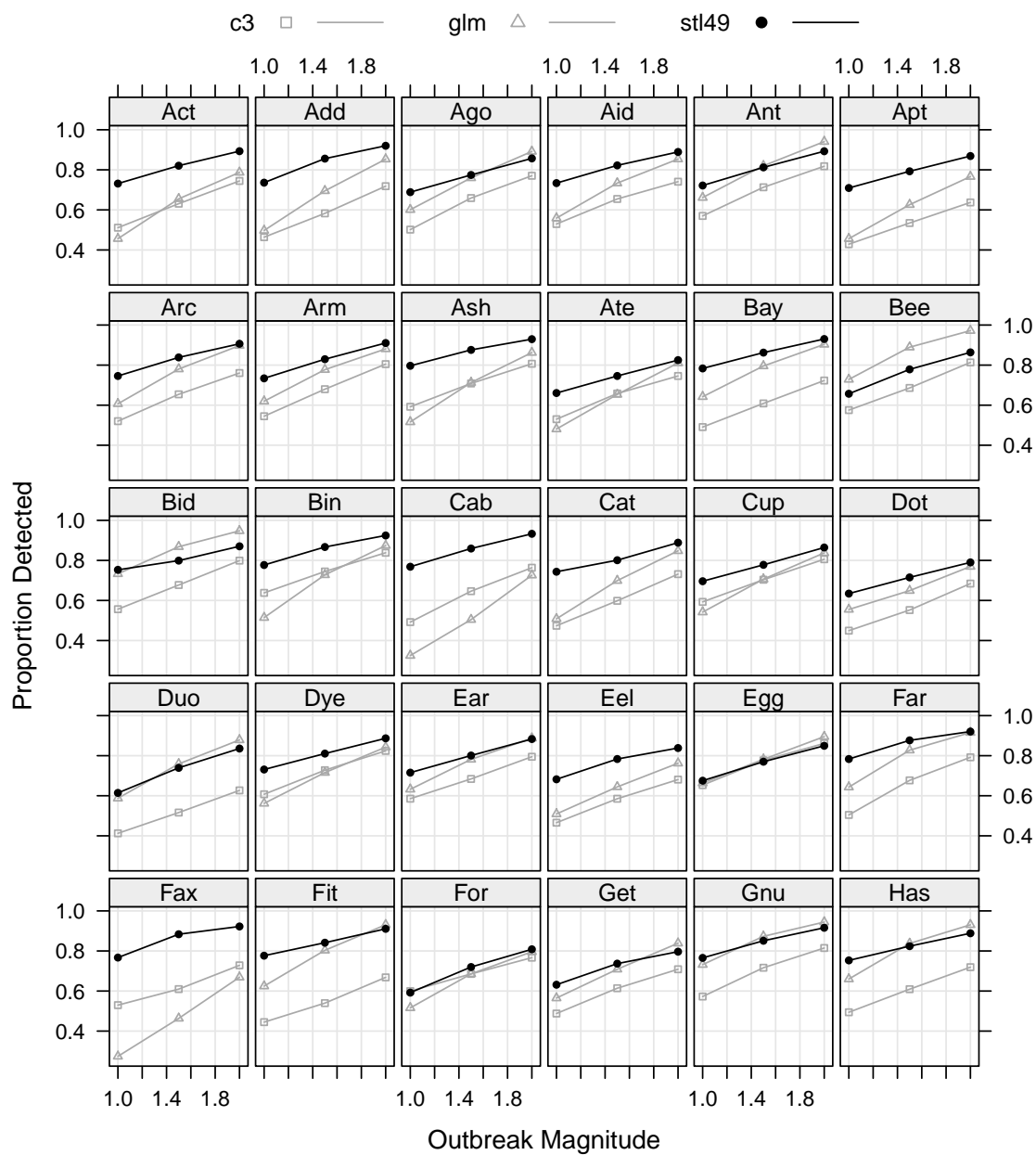


Figure 5.87. Summary of proportion of outbreaks detected vs. outbreak magnitude by ED, comparing the EARS C3, GLM, and STL $q = 49$ methods.

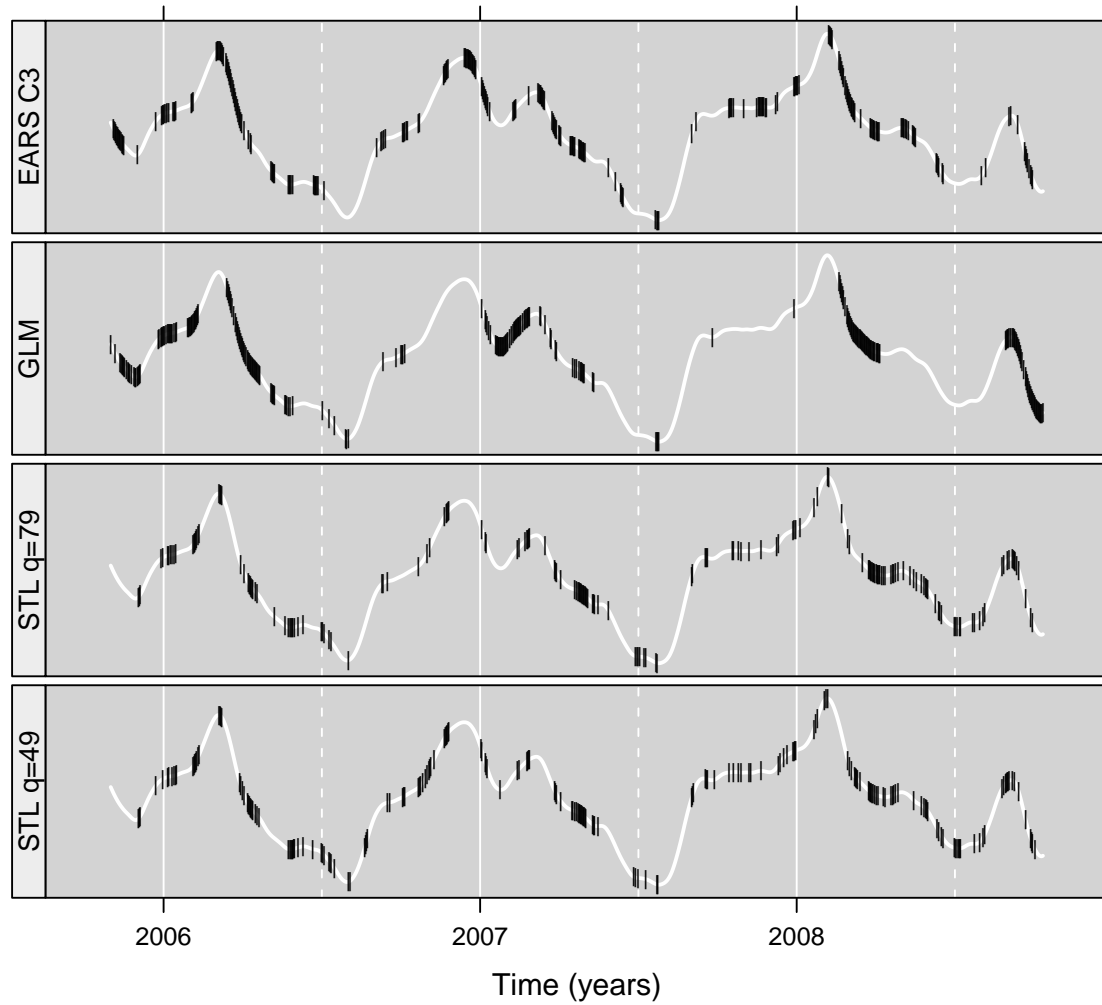


Figure 5.88. Diagnostic plot of missed outbreaks for one ED by method for outbreaks injected with magnitude $f = 2$, plotted over the yearly-seasonal component. The black symbols signify the starting date of a missed outbreak.

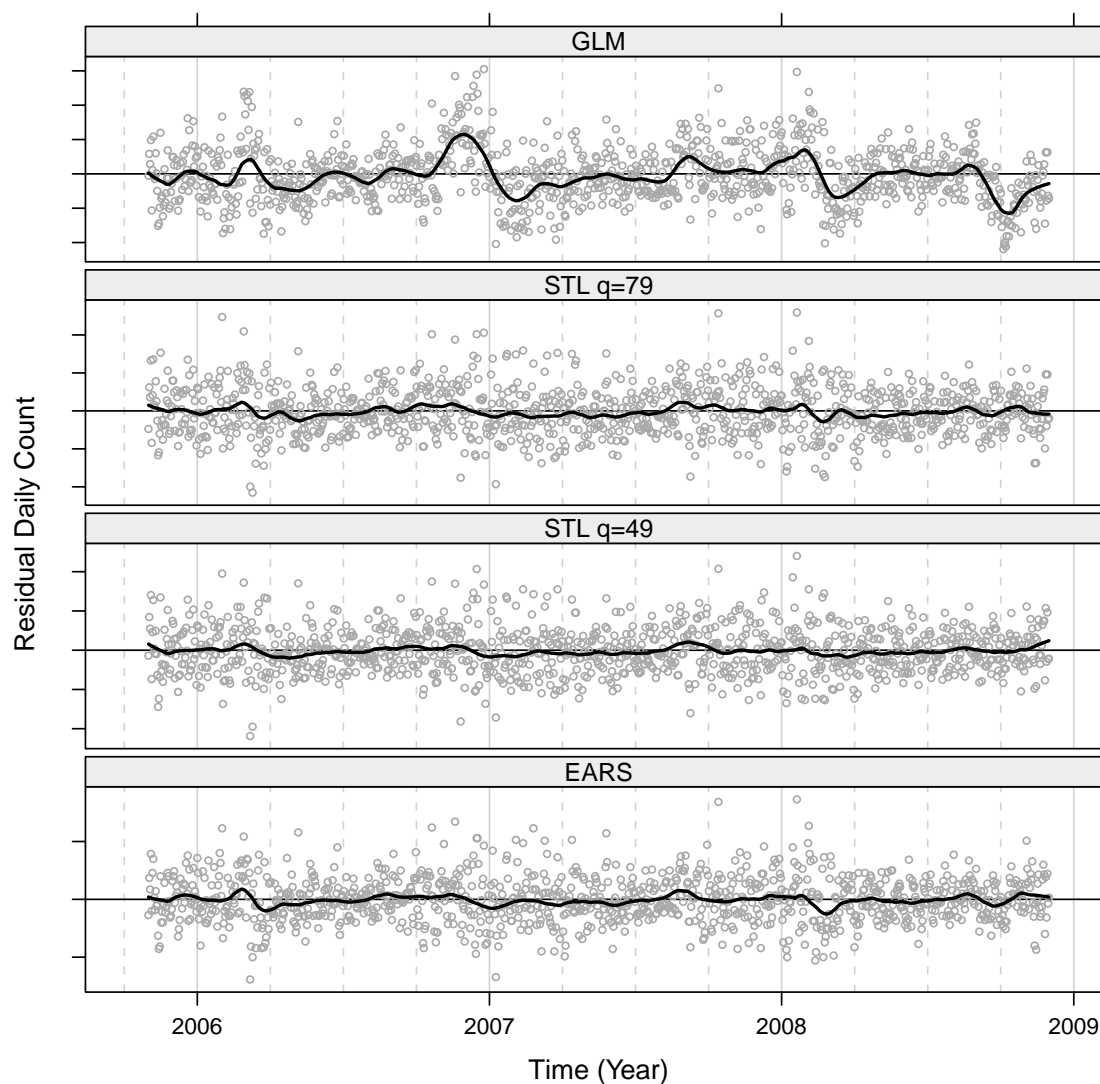


Figure 5.89. Residuals for model fits to daily respiratory counts for one ED. The EARS residuals are the observed count minus the 7 day baseline mean with lag 2. The GLM and STL residuals are obtained from the model predicted values. The smooth curve is the local mean of the residuals.

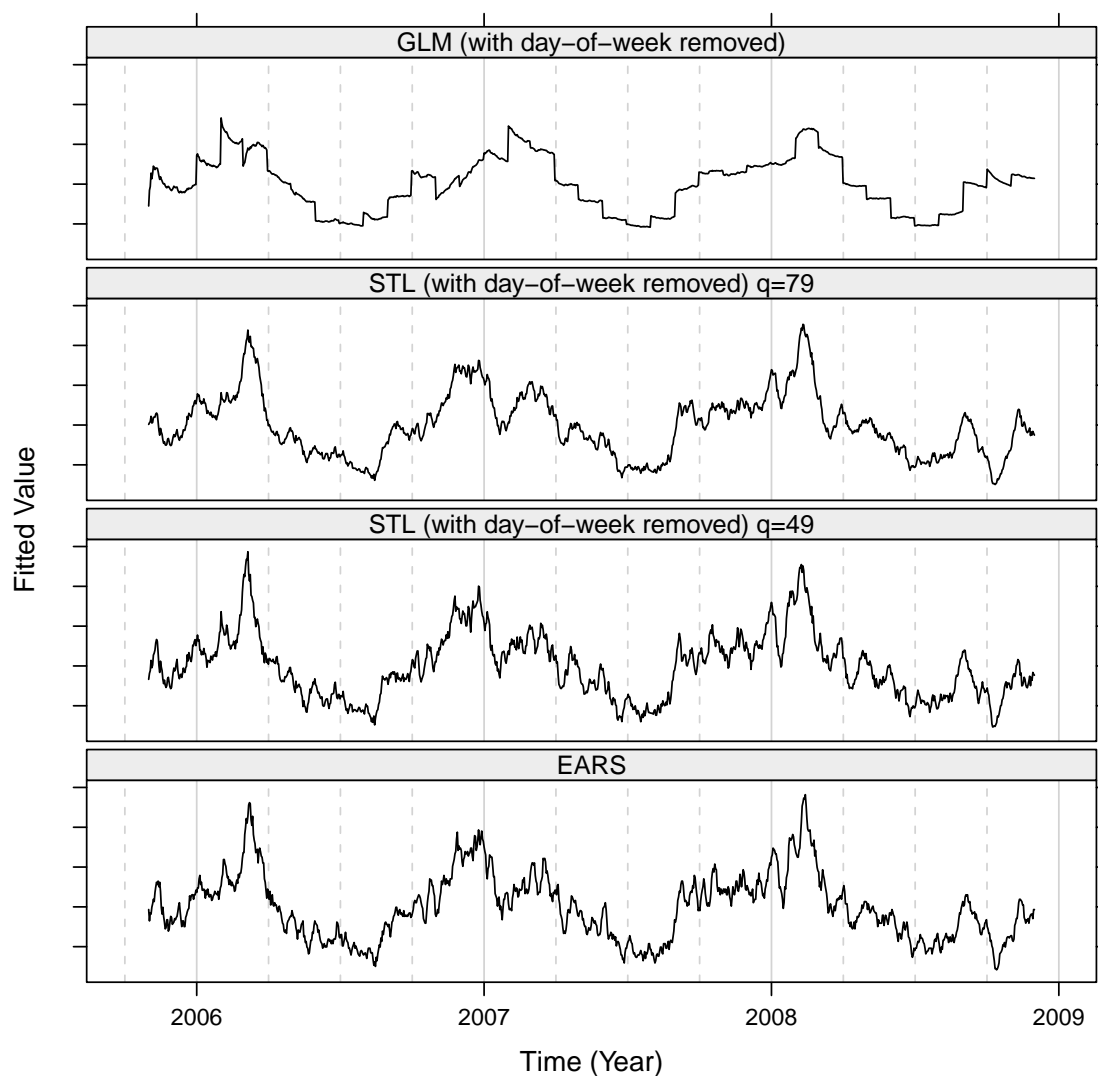


Figure 5.90. Fitted components for daily respiratory counts for one ED. The EARS fitted value at day t is the the 7 day baseline mean with lag 2. The GLM and STL fitted values are the predicted values from fitting the models up to day t but with the day-of-the-week components removed to make comparison with the variability of the EARS fitted values commensurate.

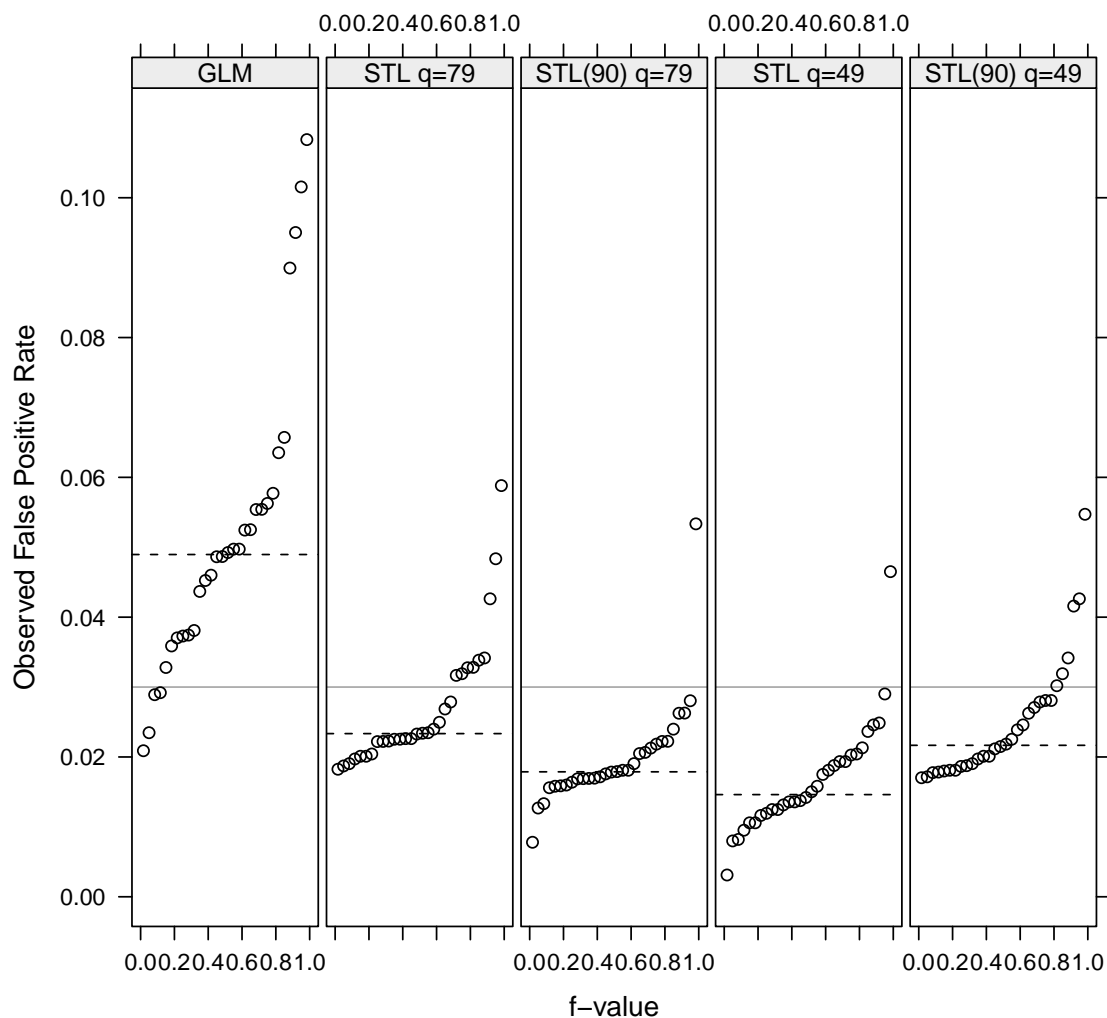


Figure 5.91. Quantile plots of observed false positive rates for the STL and GLM methods based on a theoretical false positive rate of $\rho = 0.03$, from respiratory counts for each of the 30 EDs. The dashed lines represent the median value for each method.

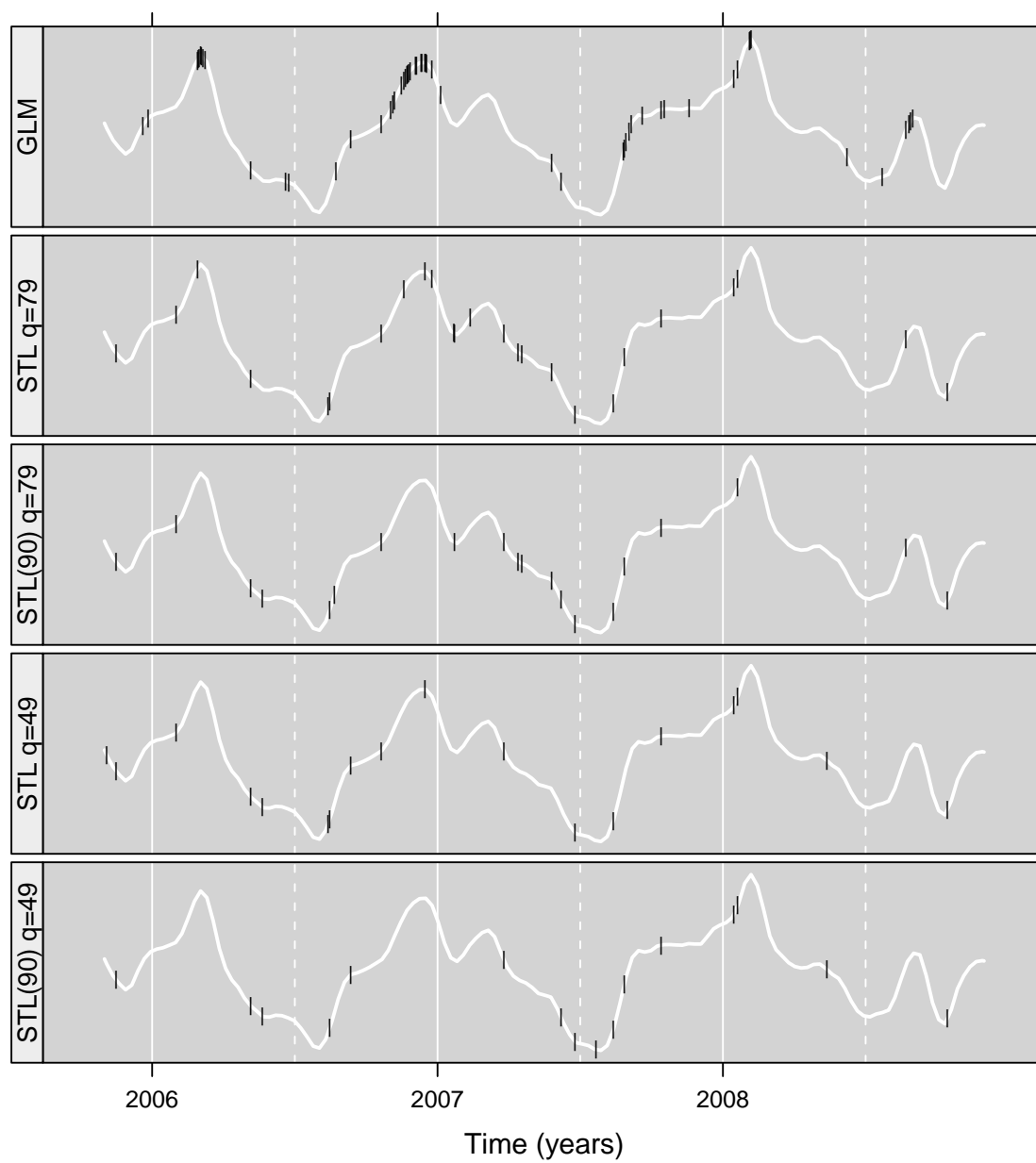


Figure 5.92. Temporal location of false positives for one ED by method.

5.5 Figures for Ed

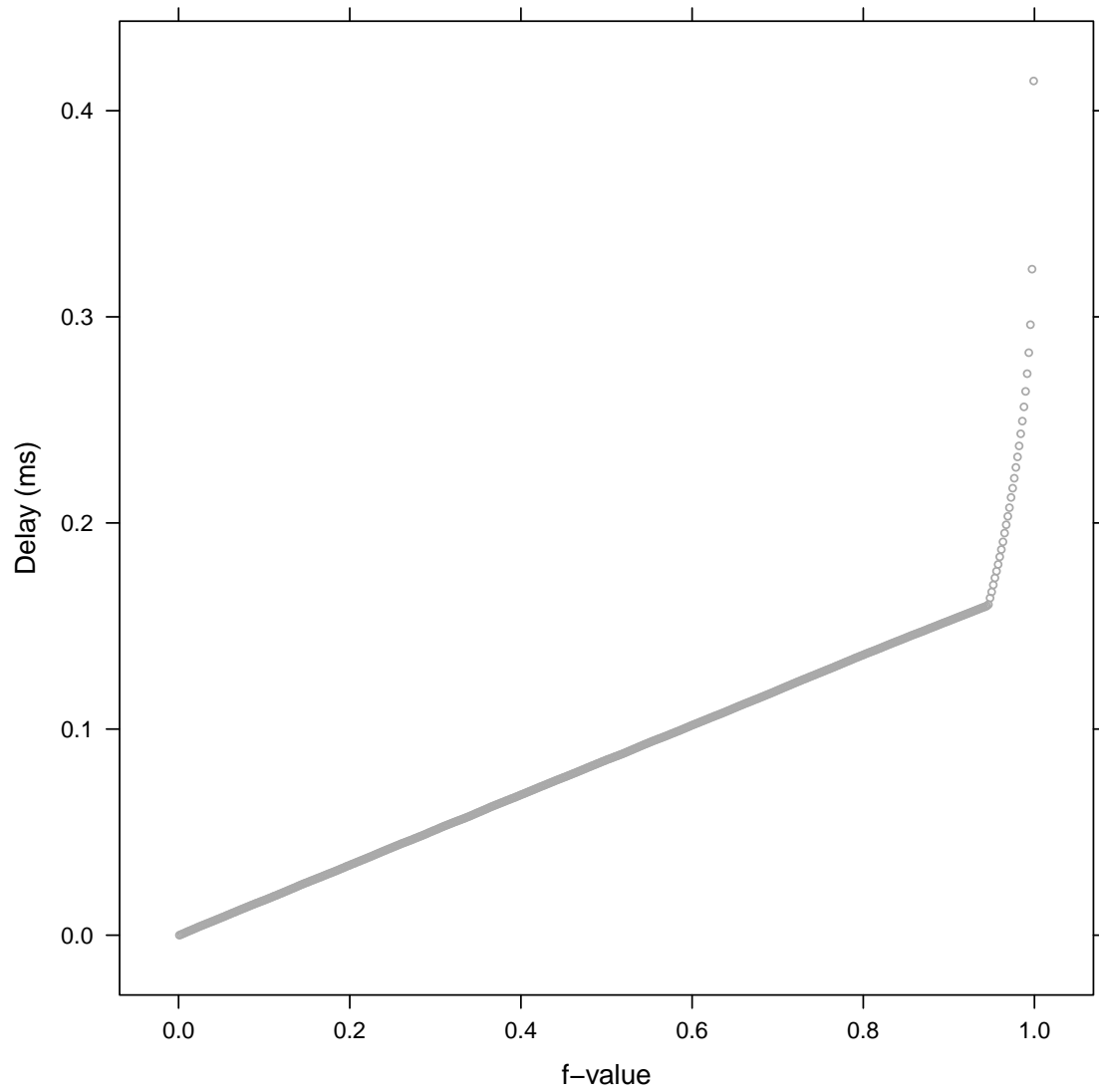


Figure 5.93. Quantile plot for Packet-Delay data.

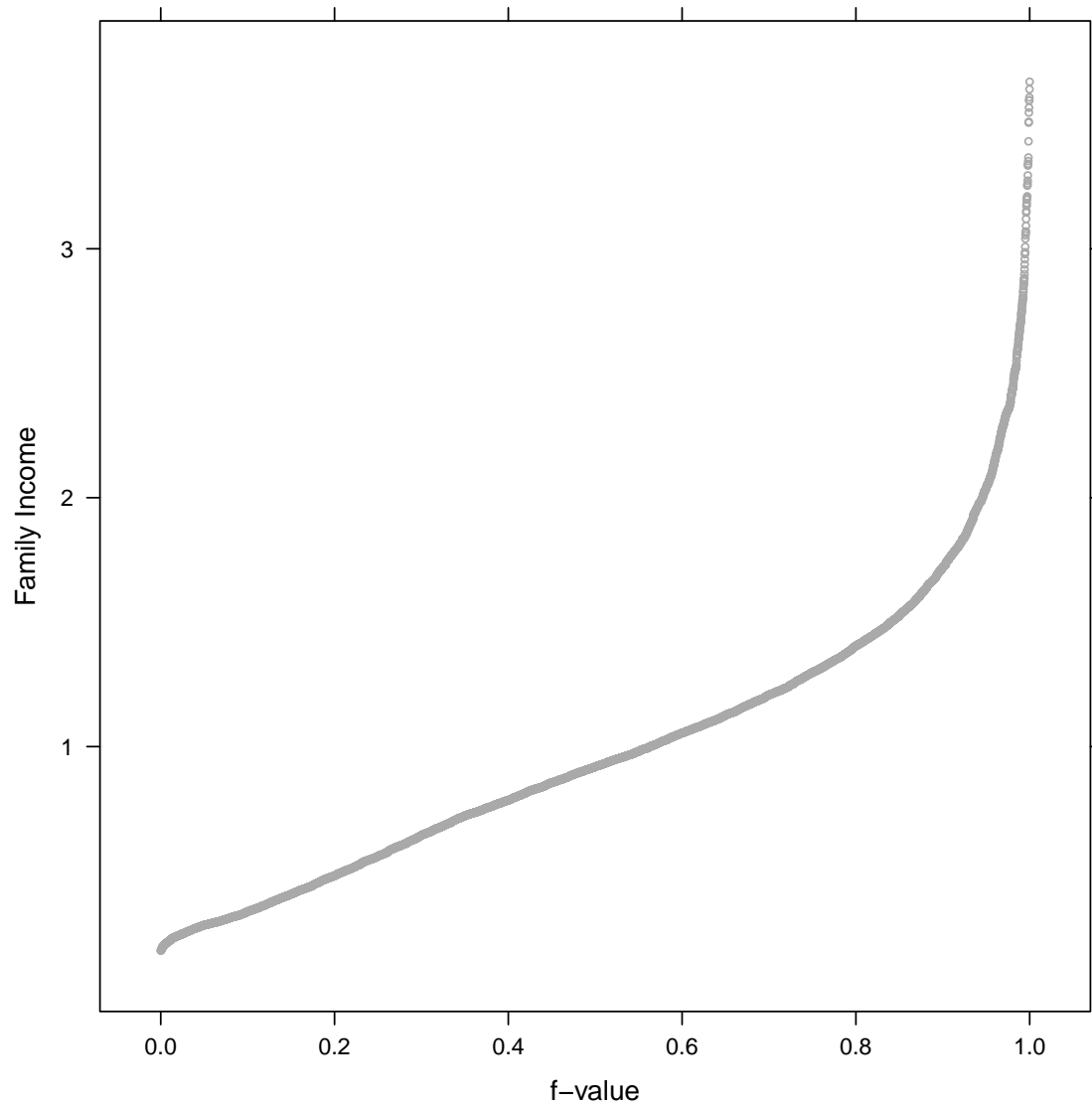


Figure 5.94. Quantile plot for Family-Income data with 39 outliers removed.

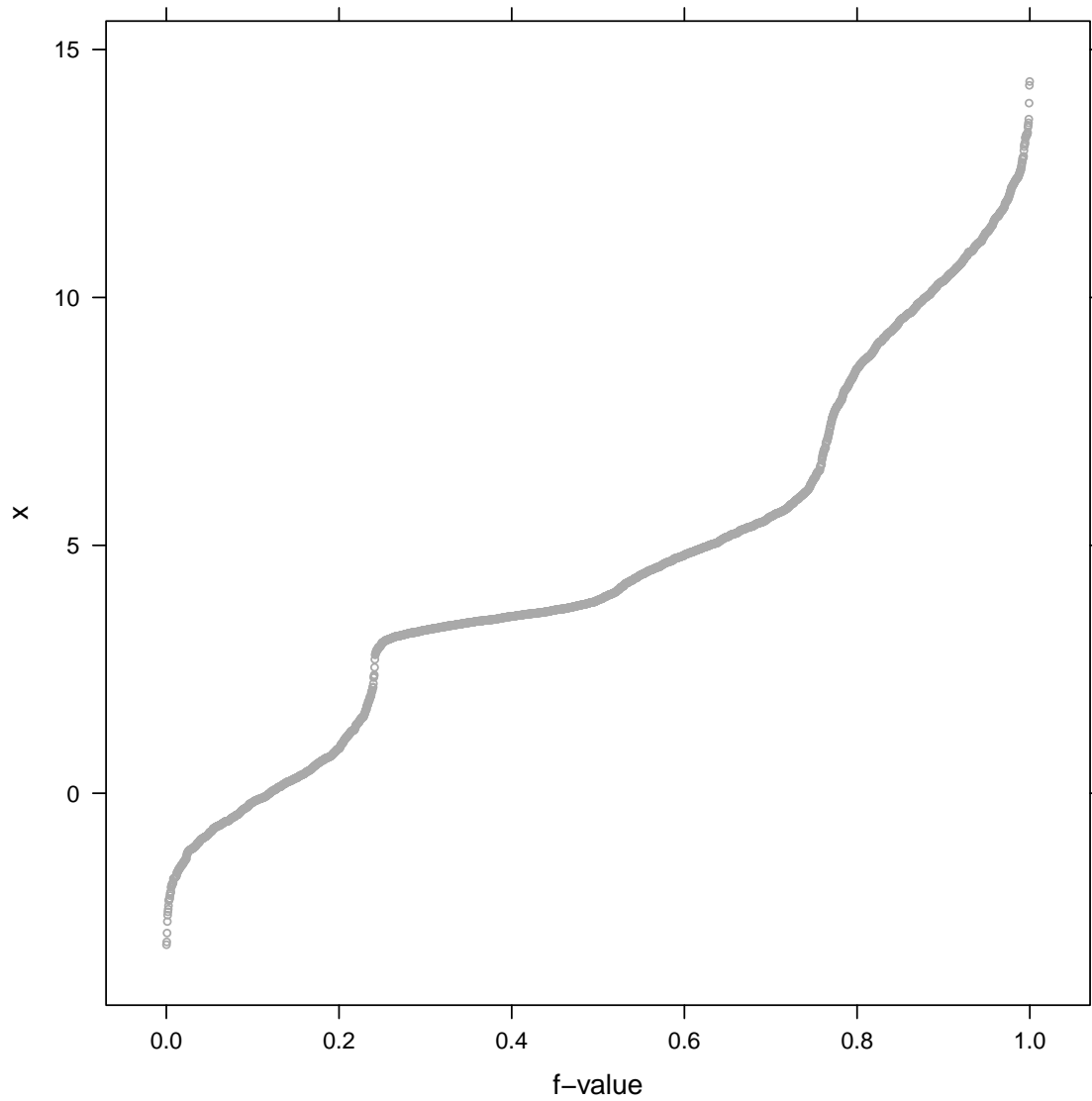


Figure 5.95. Quantile plot for a sample of 3,000 observations from the Normal-Mixture example.

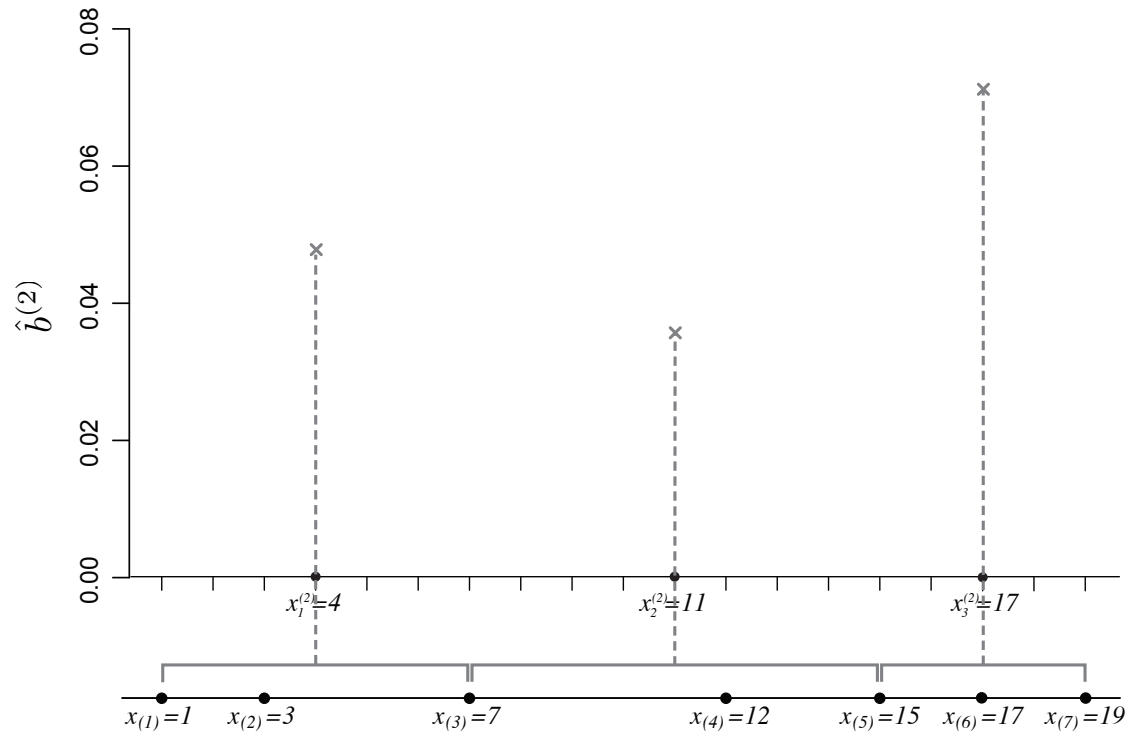


Figure 5.96. Illustration of raw estimate calculation for a sample $x = (1, 3, 7, 2, 15, 17, 19)$ with $\kappa = 2$. The order statistics $x_{(1)}, \dots, x_{(7)}$ are shown on a number line below the grid and the location points for the raw estimates $x_1^{(2)}, \dots, x_3^{(2)}$ are indicated by the dots on the x -axis. The “x” plotting characters denote the raw estimates $(x_i^{(2)}, \hat{b}_i^{(2)})$.

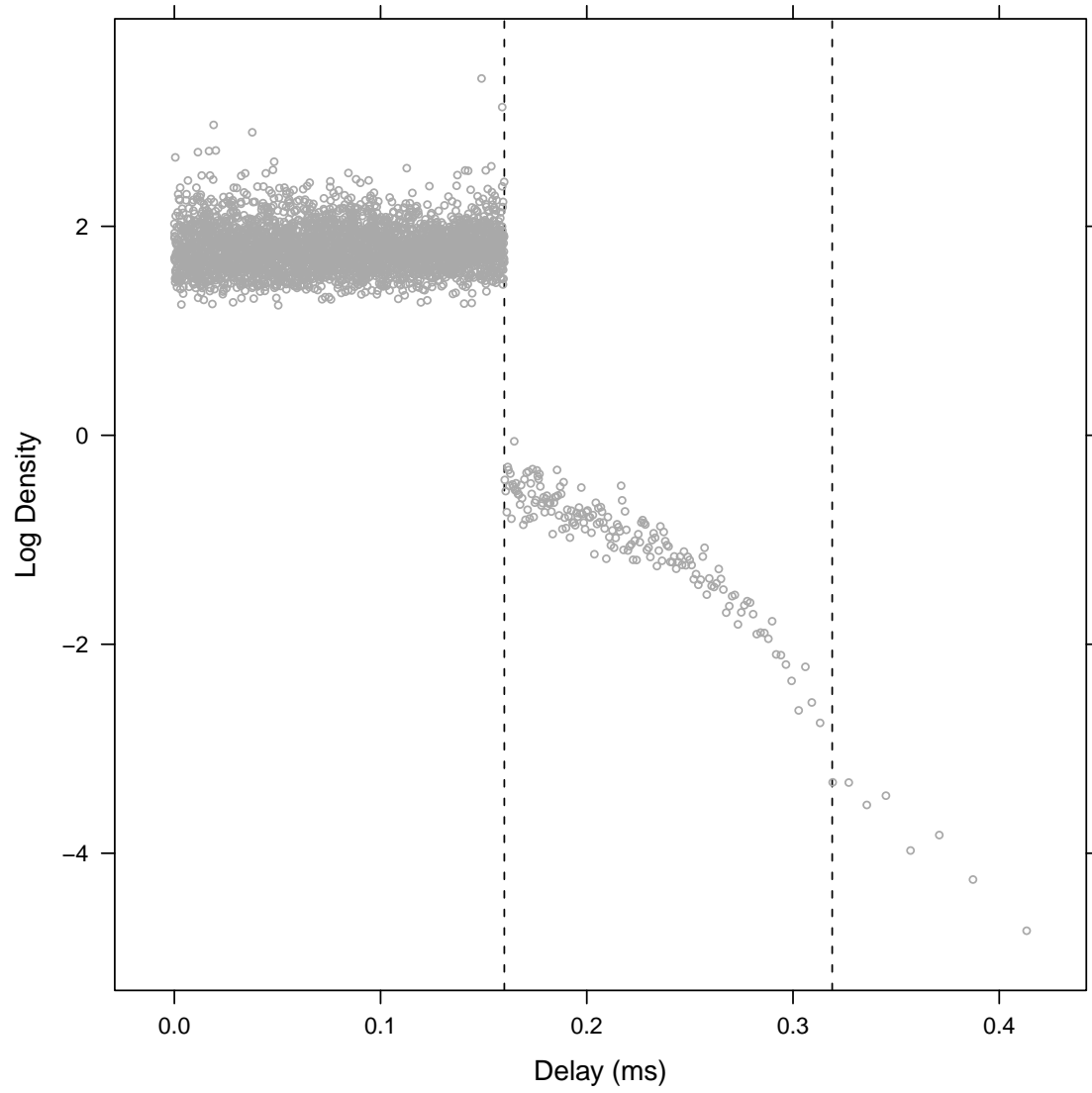


Figure 5.97. Raw estimates for Packet-Delay data with $\kappa = 75$. The plot pronounces the discontinuities in the density, at 0.16 and 0.32 (vertical lines).

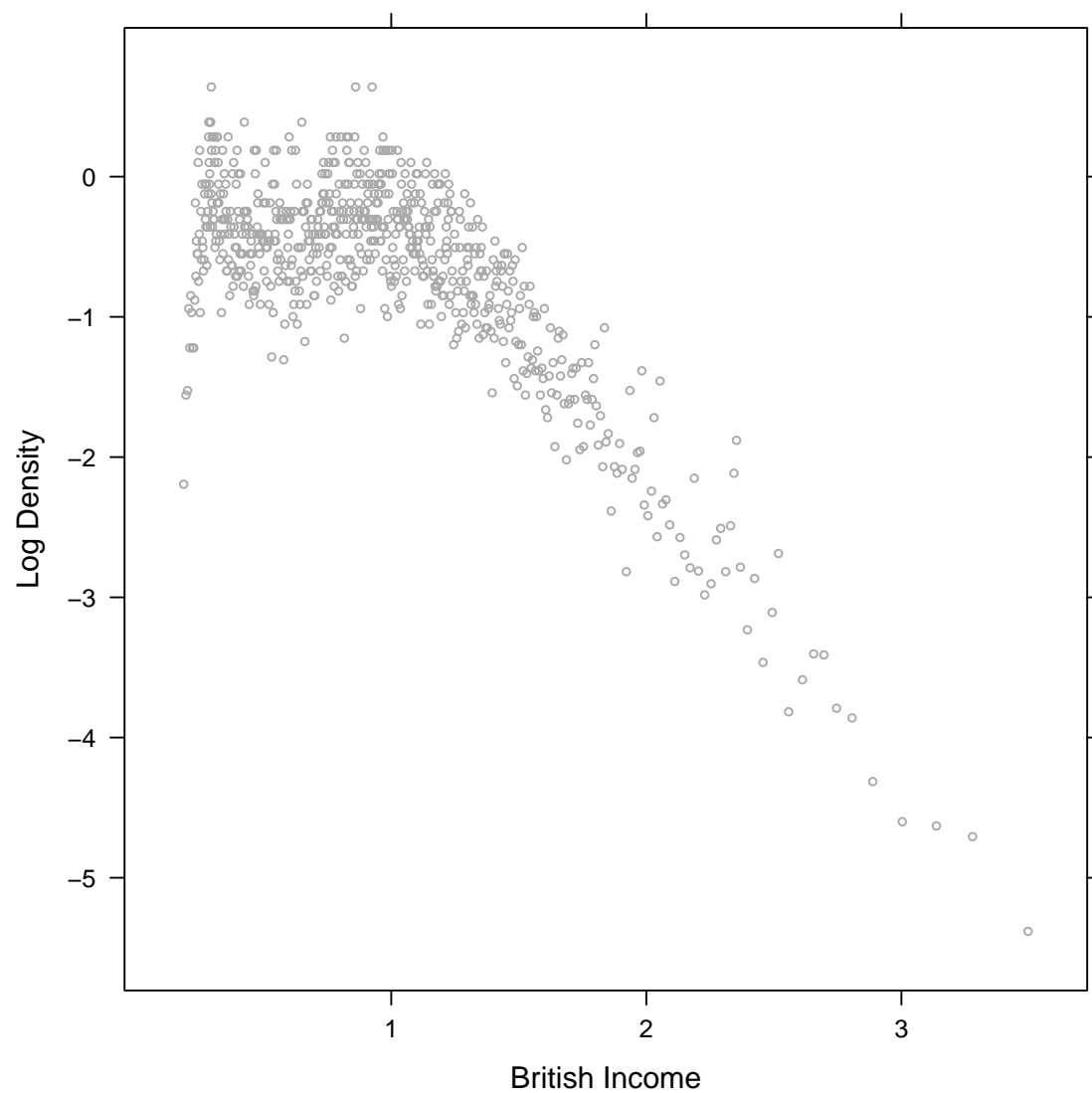


Figure 5.98. Raw estimates for Family-Income data with $\kappa = 10$.

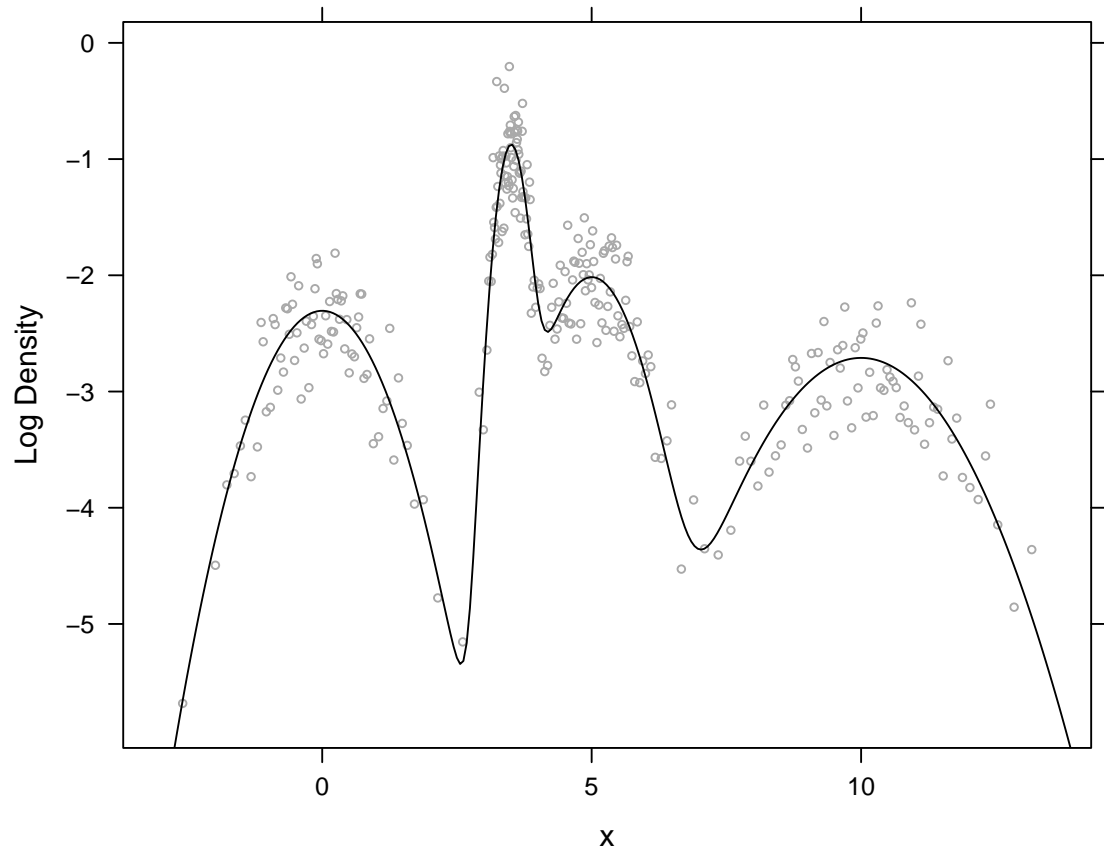


Figure 5.99. Raw estimates for Normal-Mixture with $\kappa = 10$. The solid line is the log theoretical density.

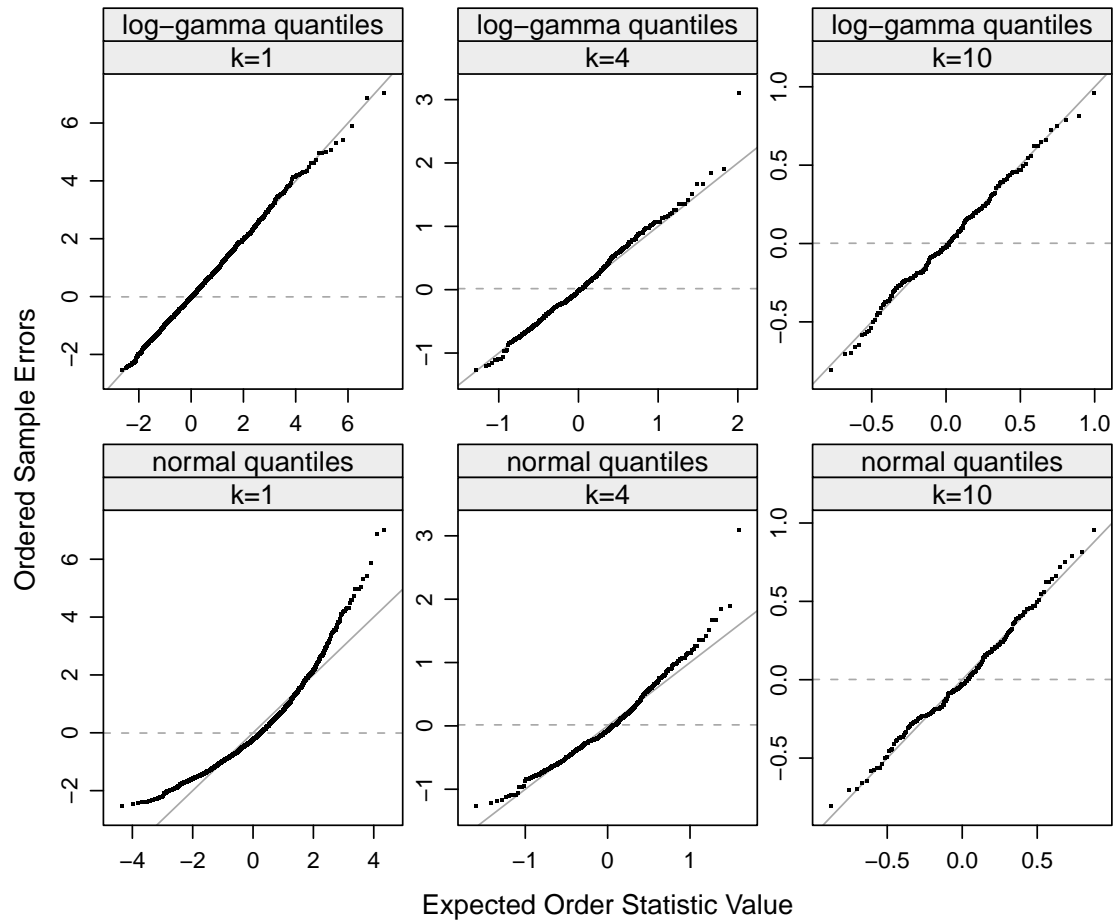


Figure 5.100. Normal and log-gamma probability plots of the sample errors $\hat{y}_i^{(\kappa)} - \log f(x_i^{(\kappa)})$ for the Normal-Mixture sample, where normal distribution has variance $\psi_1(\kappa)$. The dashed line is the mean of the errors.

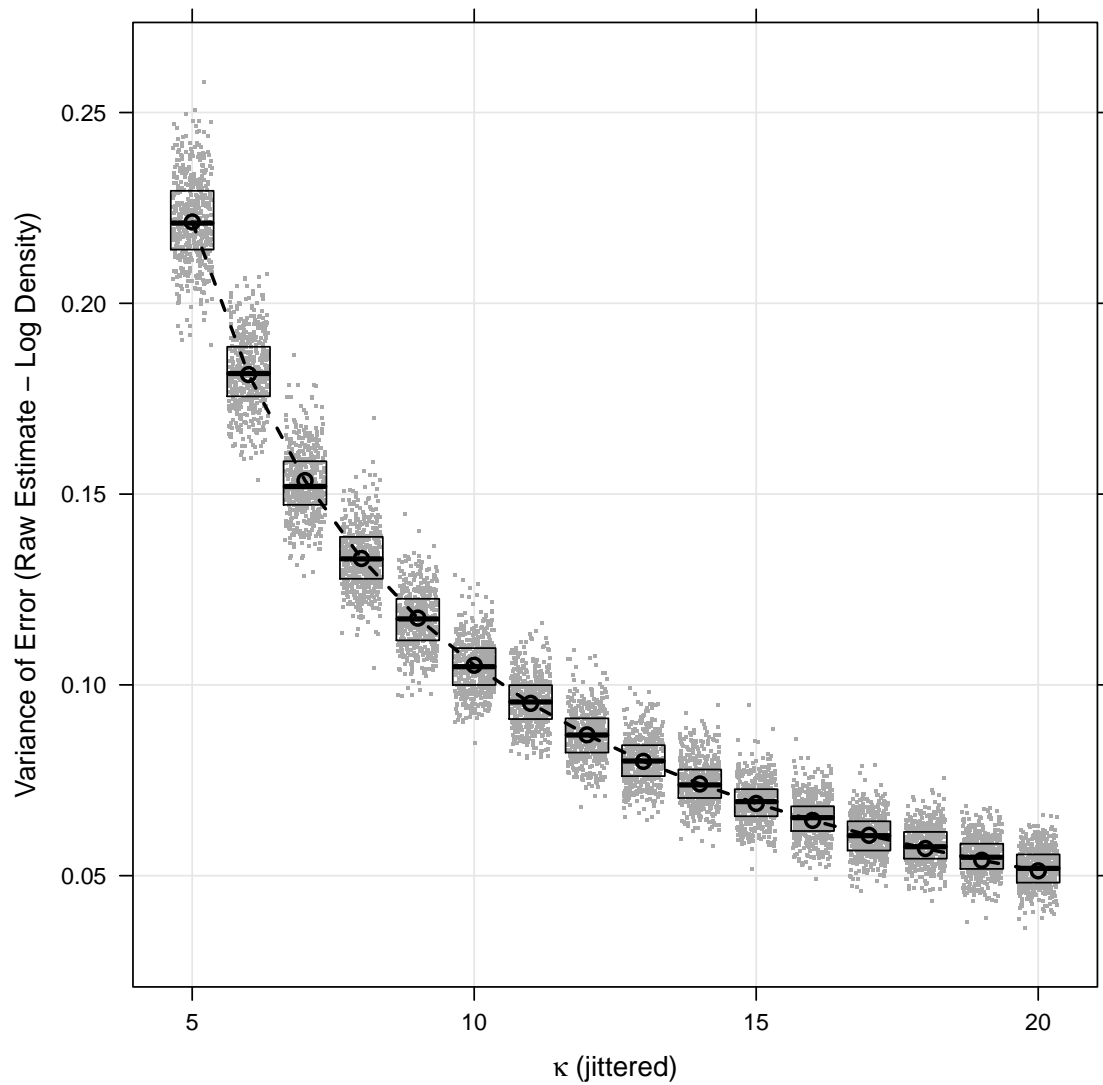


Figure 5.101. Variance of sample errors $\hat{y}_i^{(\kappa)} - \log f(x_i^{(\kappa)})$ for several Normal-Mixture samples of size 3,000 and for κ ranging from 5 to 20. The boxes represent the interquartile range for the variances at each κ with the horizontal line in each box representing the median value. The points with the dashed line connecting them represent the $\psi_1(\kappa)$, the theoretical variance for the raw estimates.

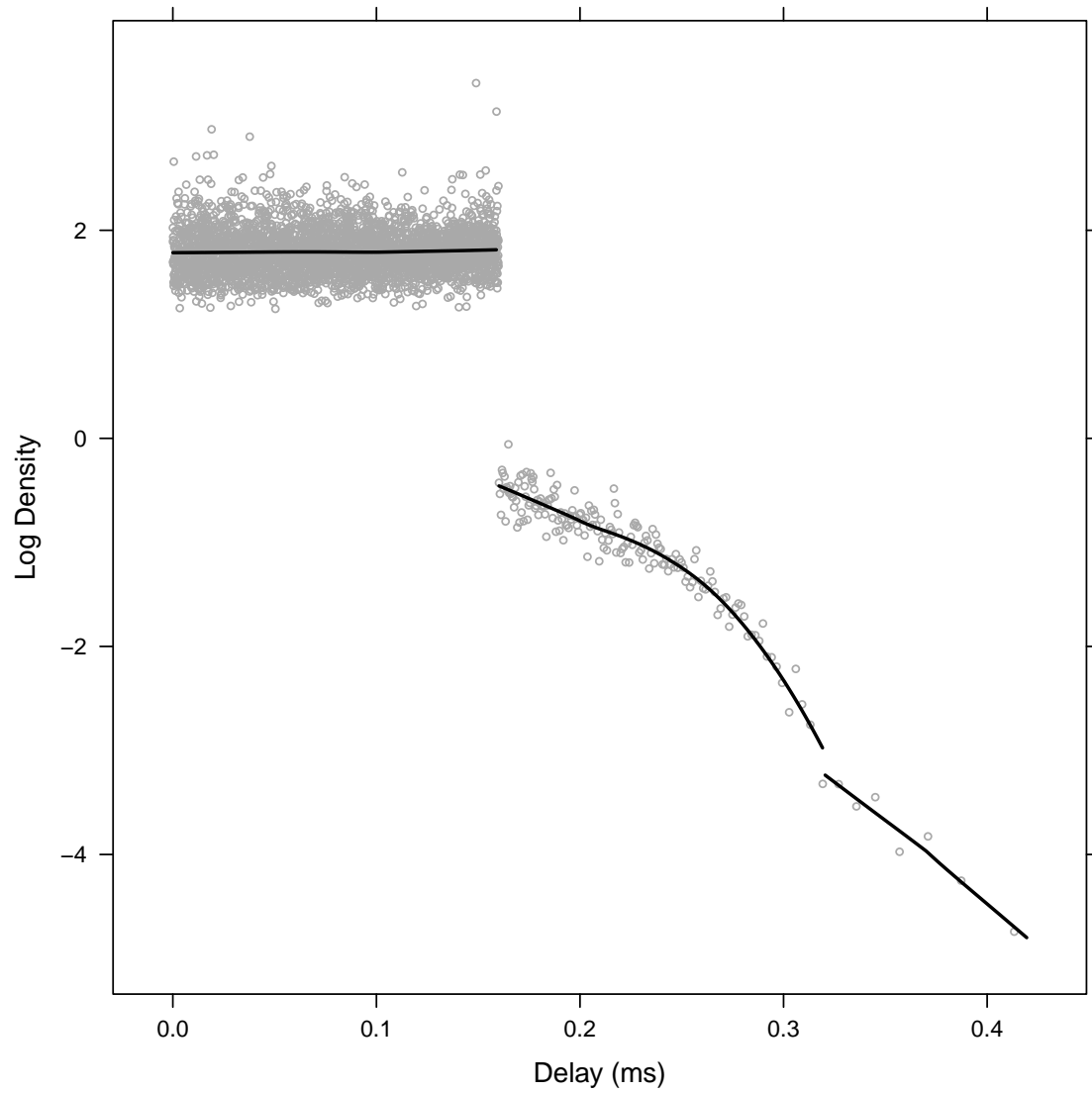


Figure 5.102. Loess fit to the Packet-Delay data on the log scale. Smoothing parameters used were: $\delta = 1$ and $\alpha = 0.75$ for the lowest interval, $\delta = 2$ and $\alpha = 0.5$ for the middle interval, and $\delta = 1$ and $\alpha = 1.5$ for the highest interval.

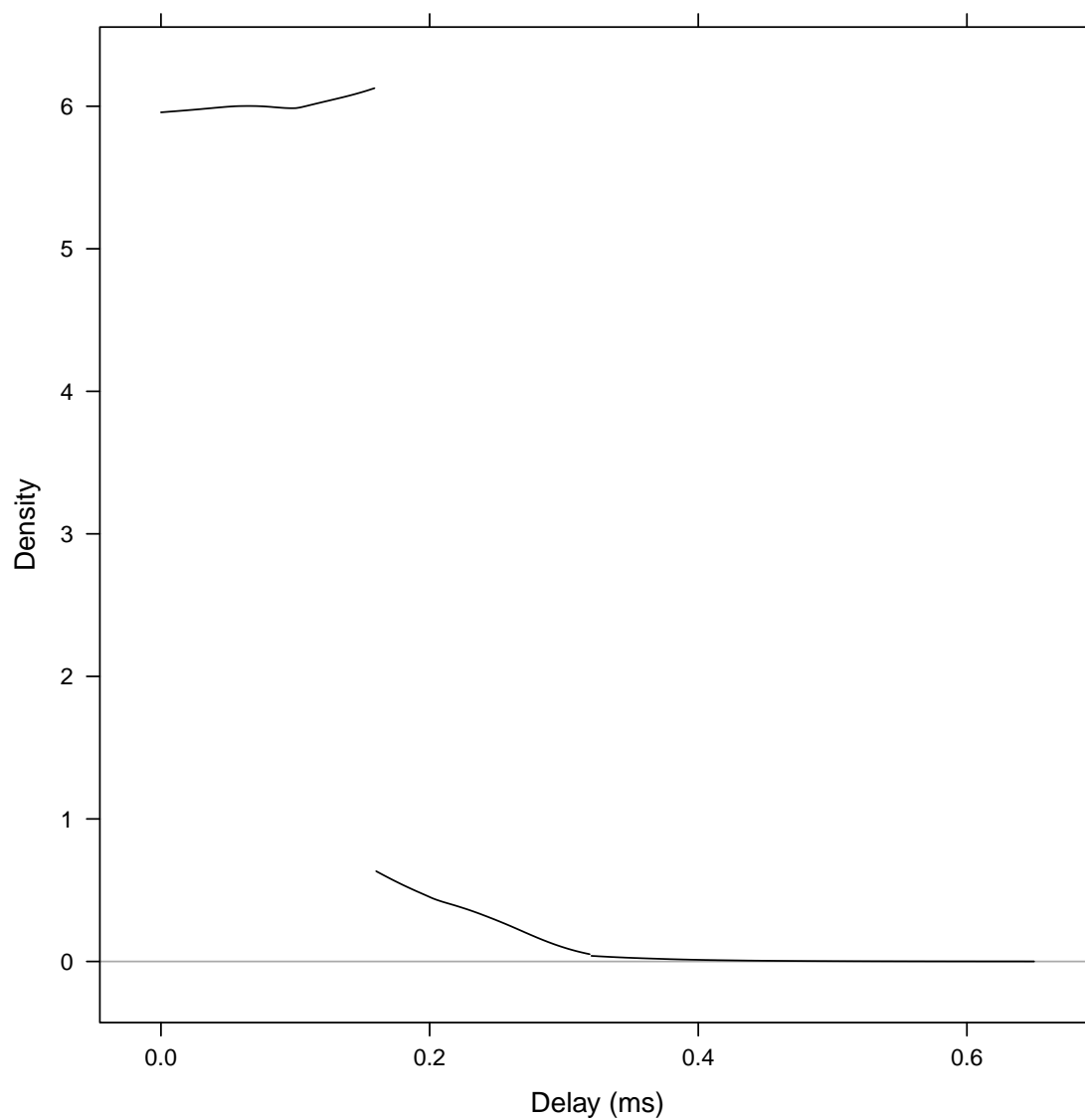


Figure 5.103. Exponentiated loess fit for the Packet-Delay data.

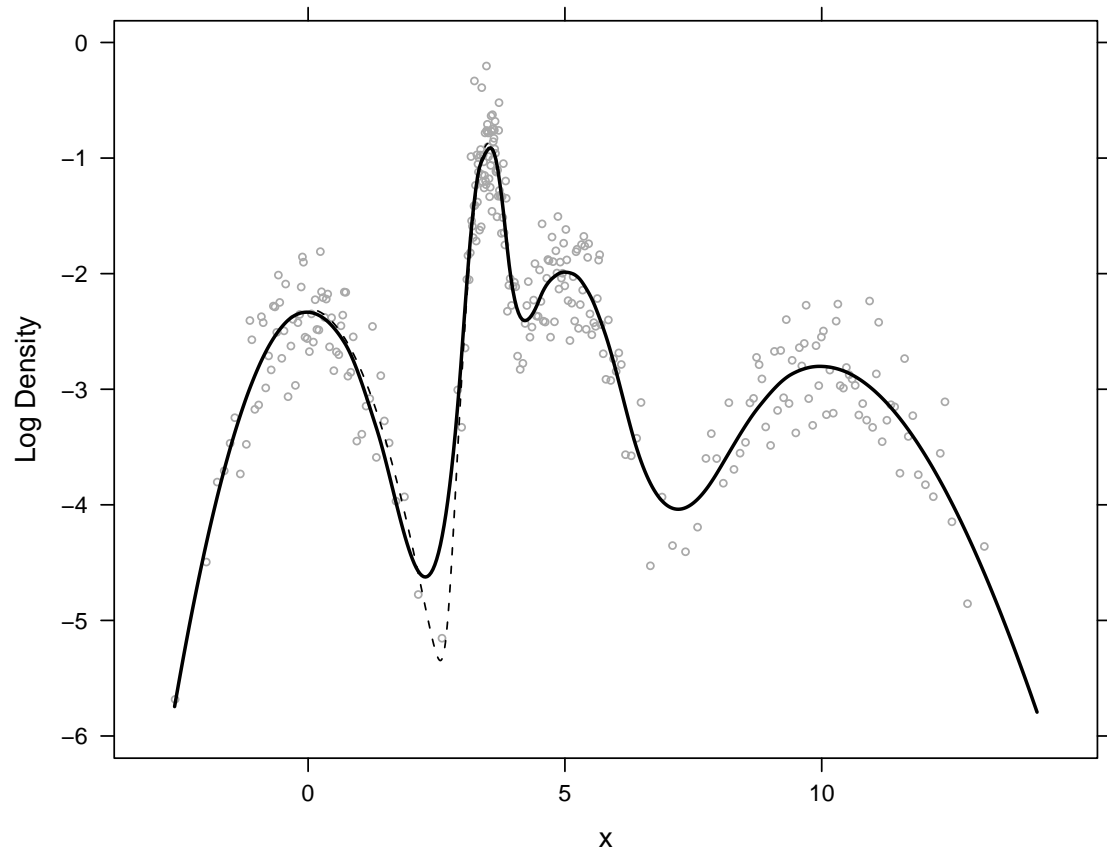


Figure 5.104. Loess fit with $\delta = 3$, $\alpha = 0.24$ to the Normal-Mixture raw estimates. The dashed line represents the true log density.

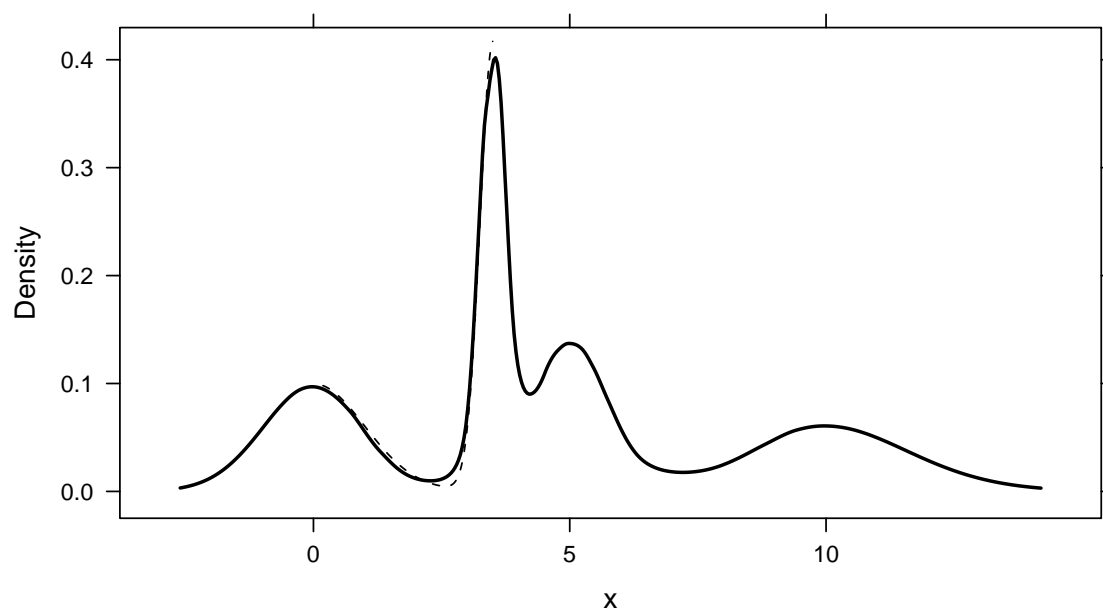


Figure 5.105. Exponentiated loess fit to the Normal-Mixture data. The dashed line represents the true density.

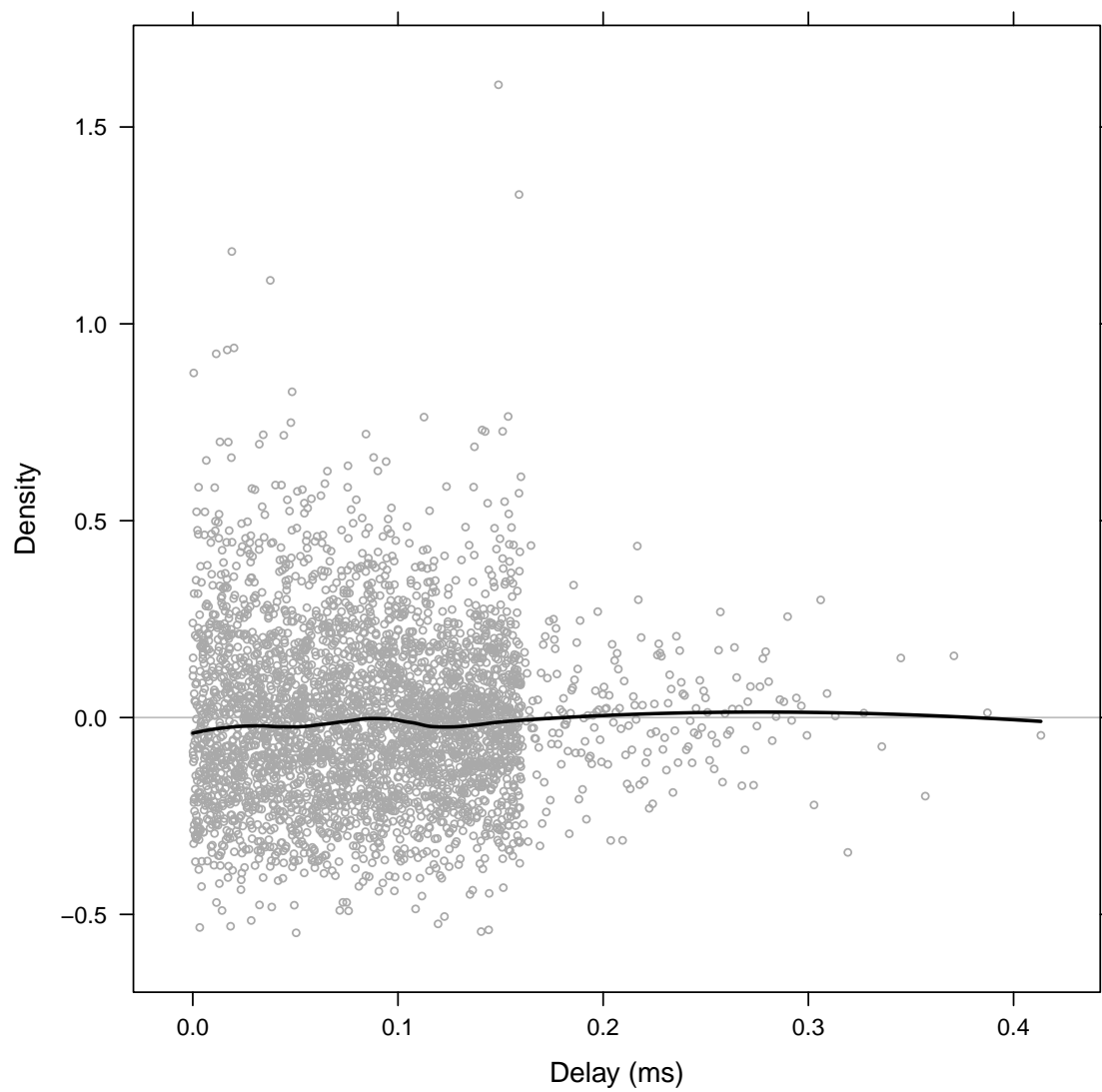


Figure 5.106. Residuals from loess fit for the Packet-Delay data.

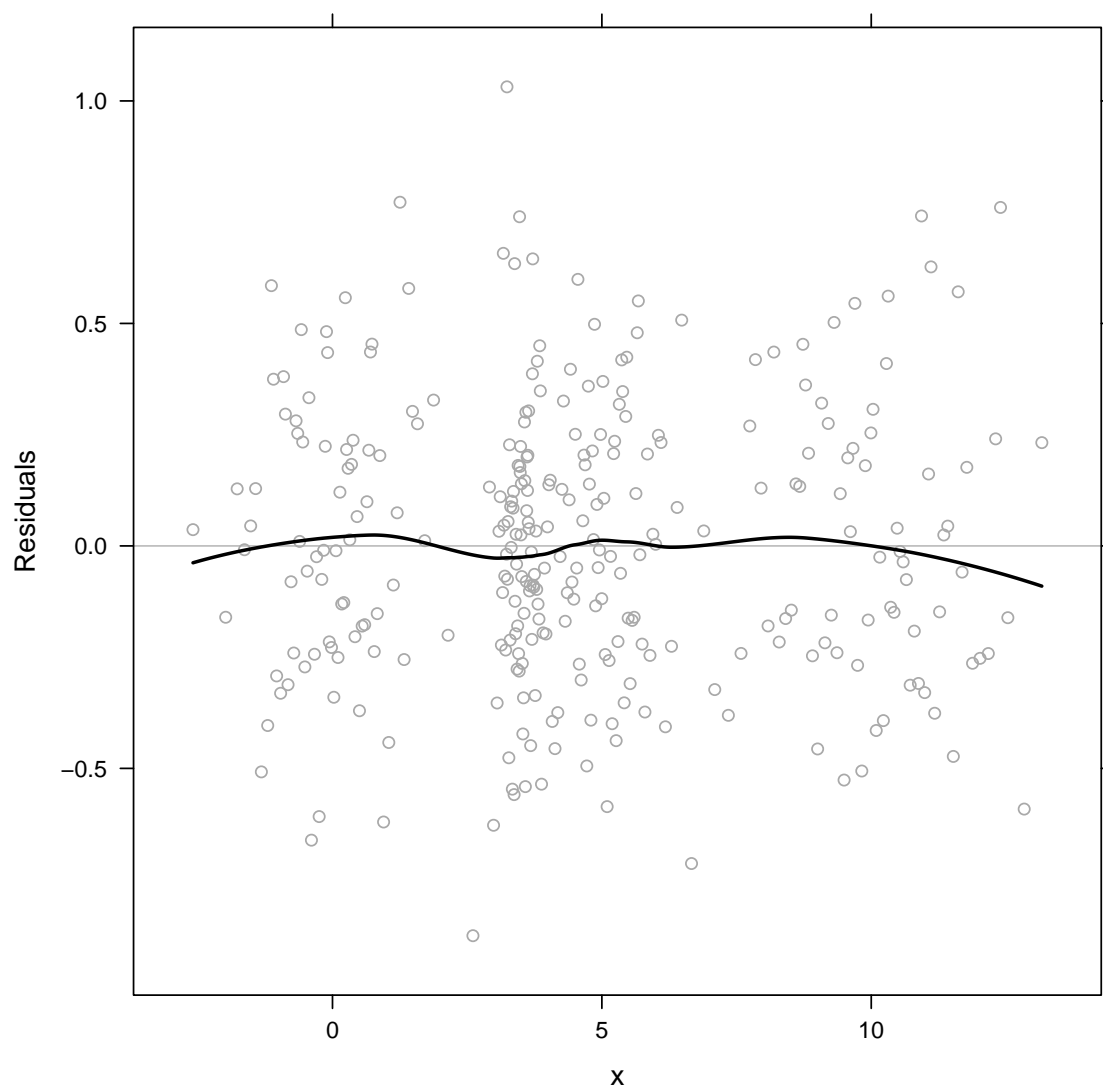


Figure 5.107. Residuals from loess fit for the Normal-Mixture data.

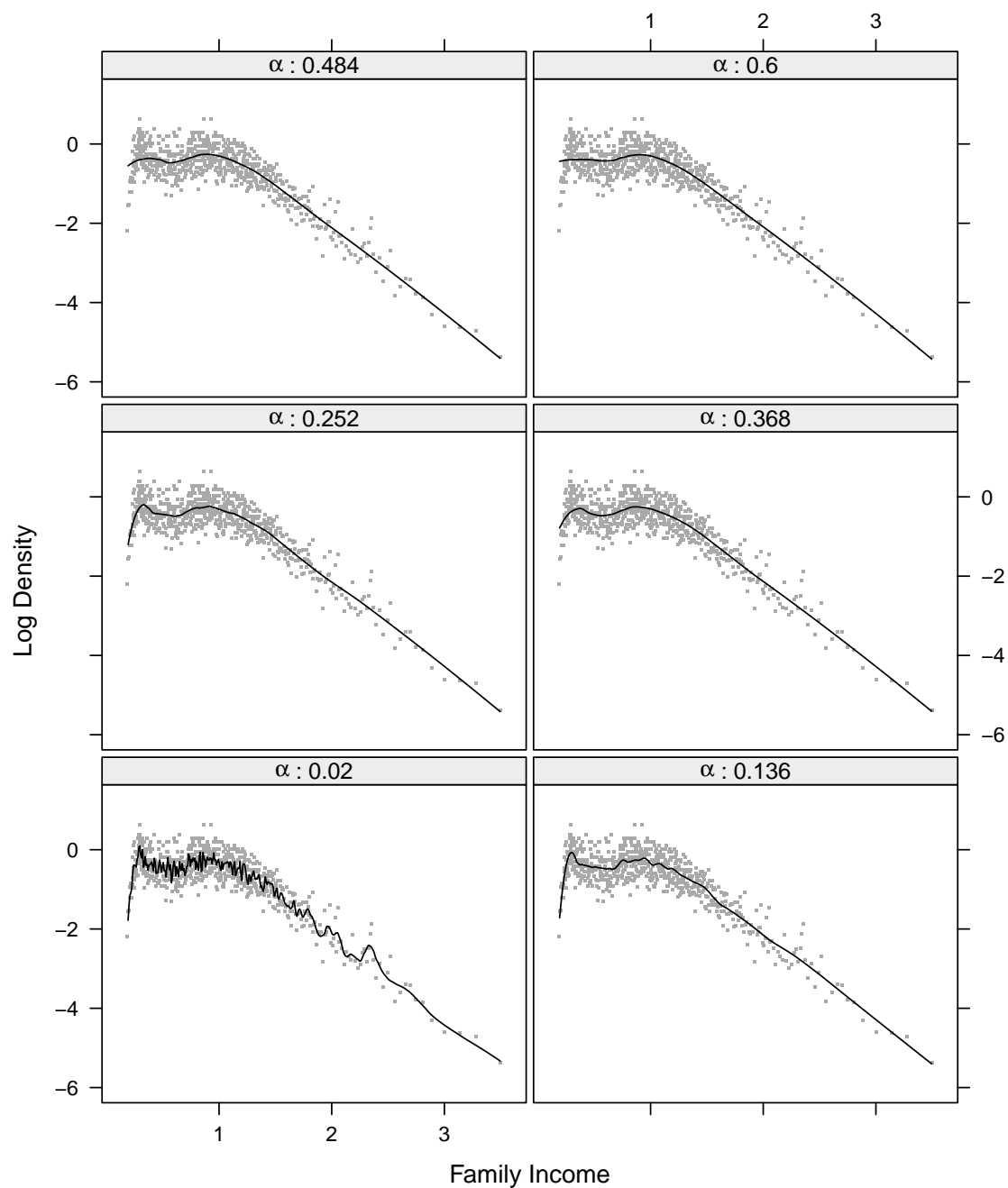


Figure 5.108. Family-Income data log density fits for $\lambda = 2$ and different choices of α for local quadratic fitting.

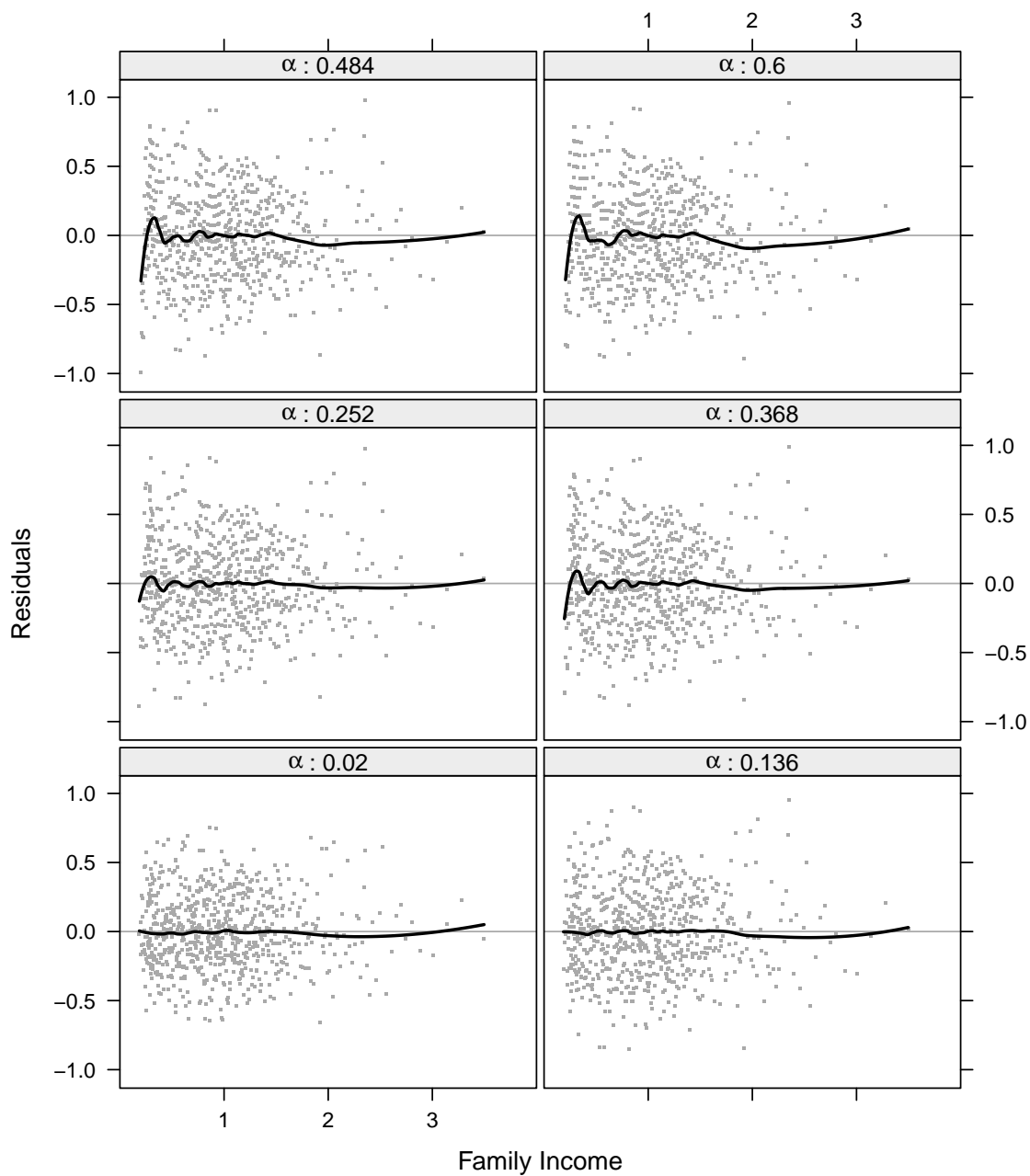


Figure 5.109. Family-Income data log density residuals for $\lambda = 2$ and different choices of α for local quadratic fitting. A loess smooth is added to highlight deviations from 0.

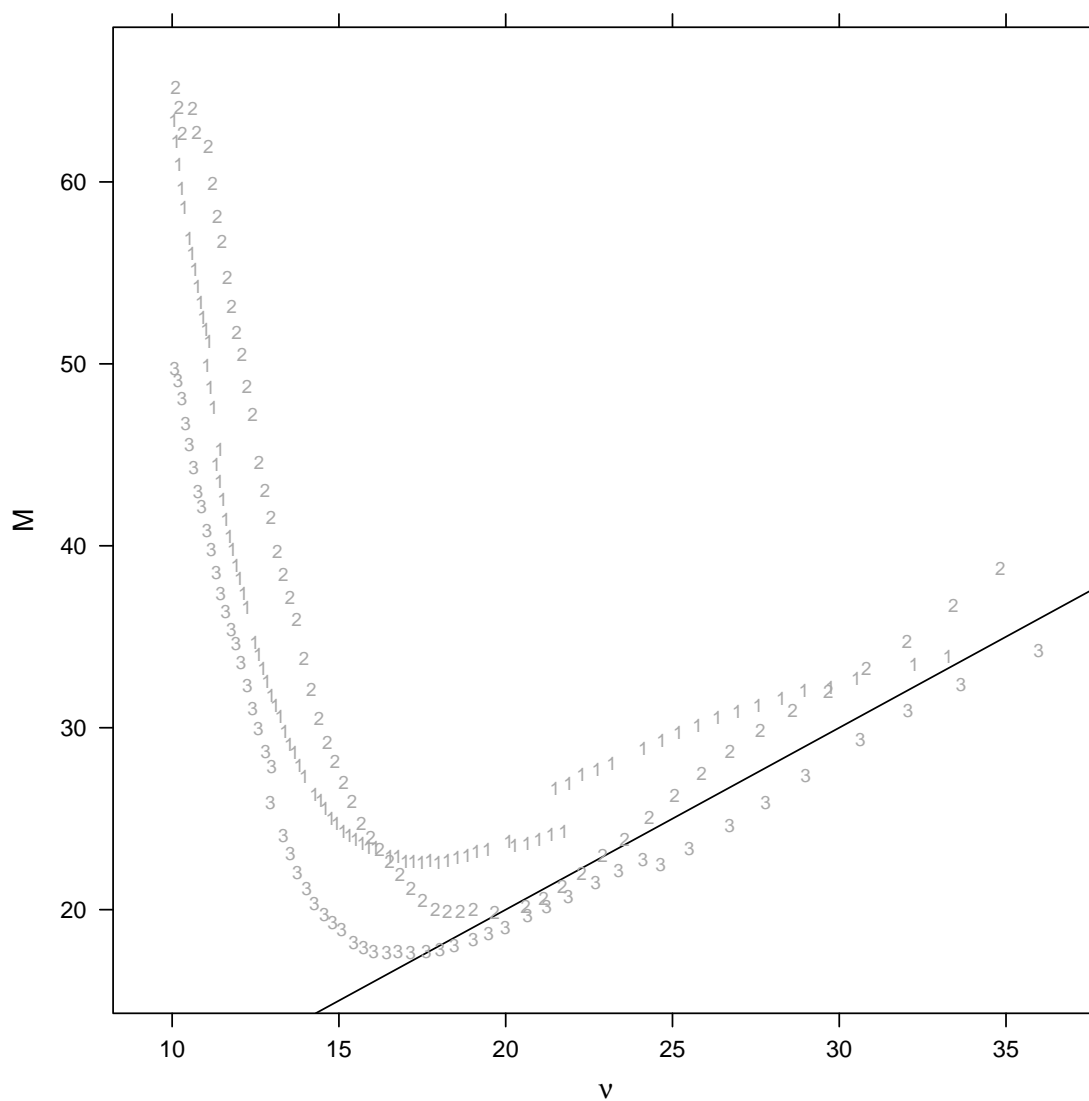


Figure 5.110. C_p plot for the Family-Income data. The plotting characters correspond to the degree of the fit.

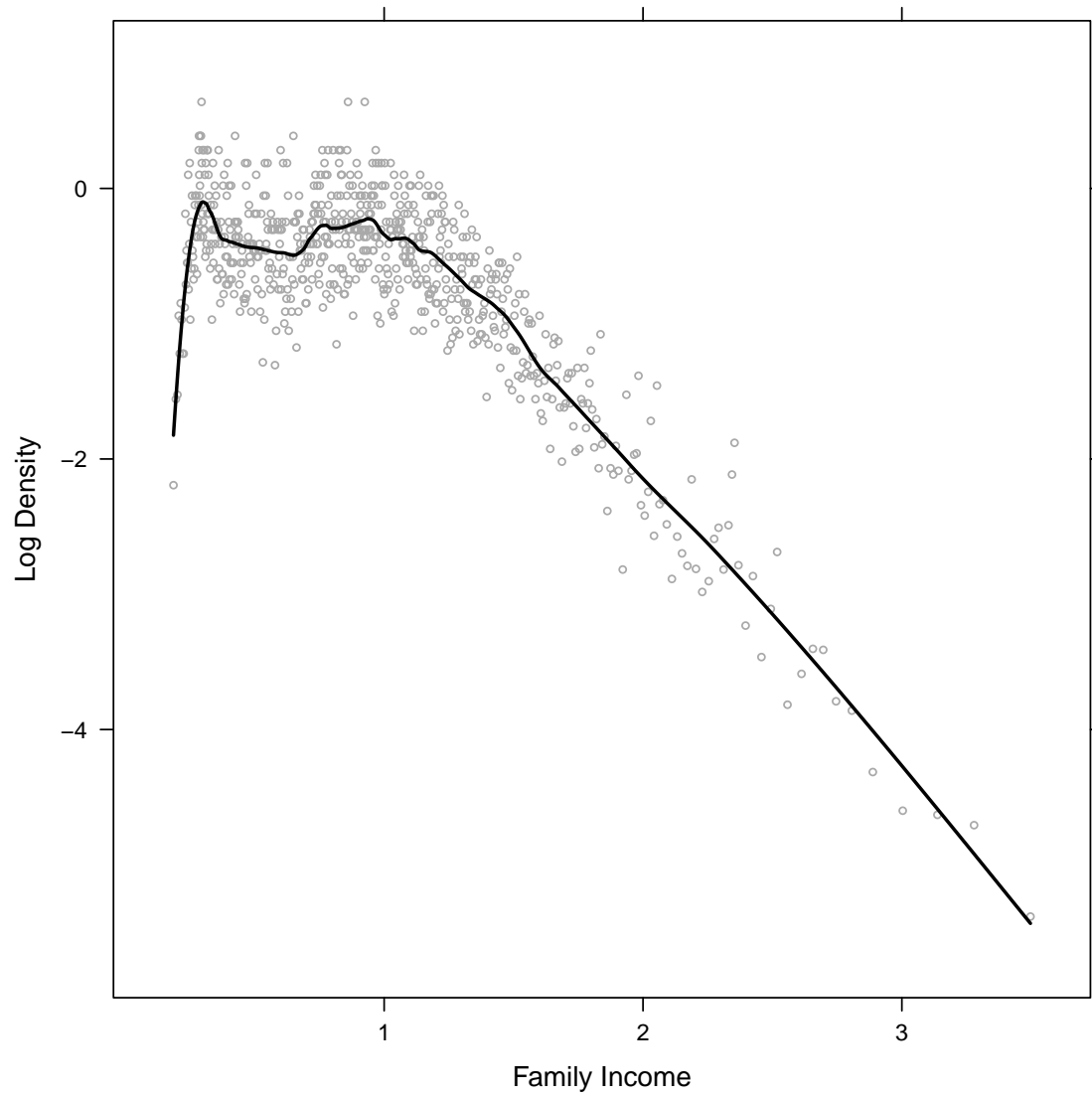


Figure 5.111. Final loess fit with $\delta = 2$, $\alpha = 0.16$ for the Family-Income data.

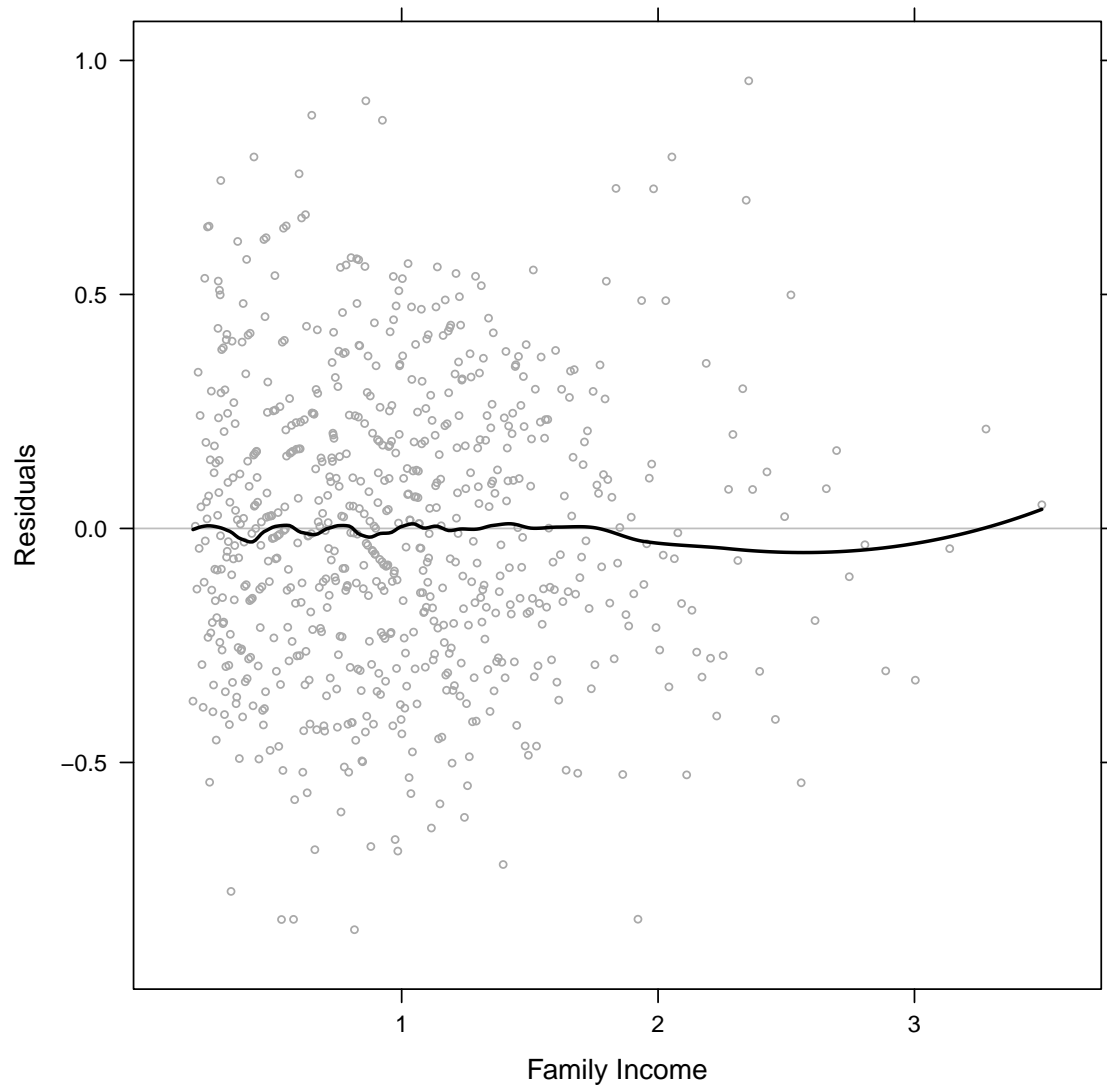


Figure 5.112. Final loess fit residuals with $\delta = 2$, $\alpha = 0.16$ for the Family-Income data.

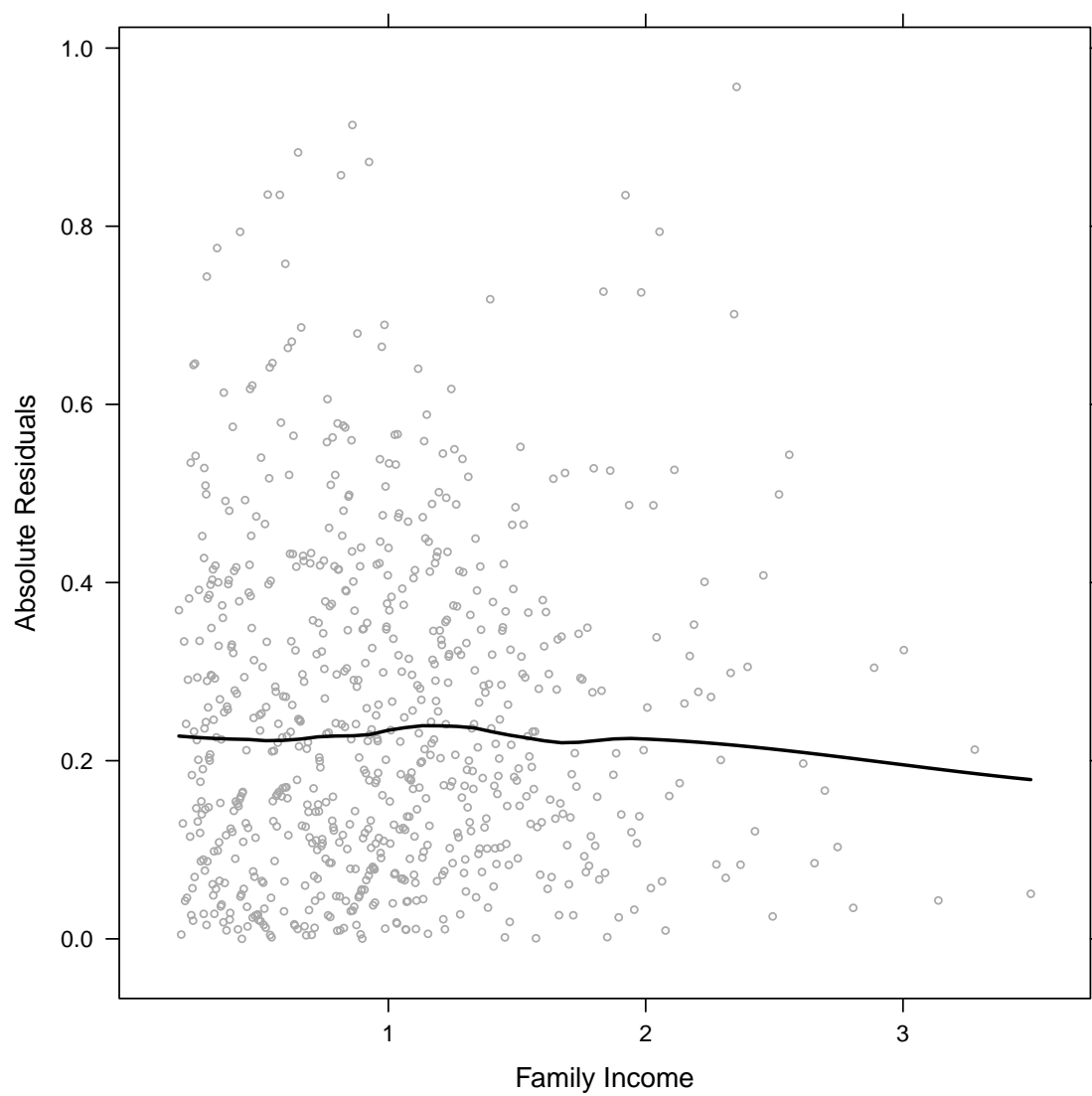


Figure 5.113. Absolute residuals vs. income for Family-Income fit.

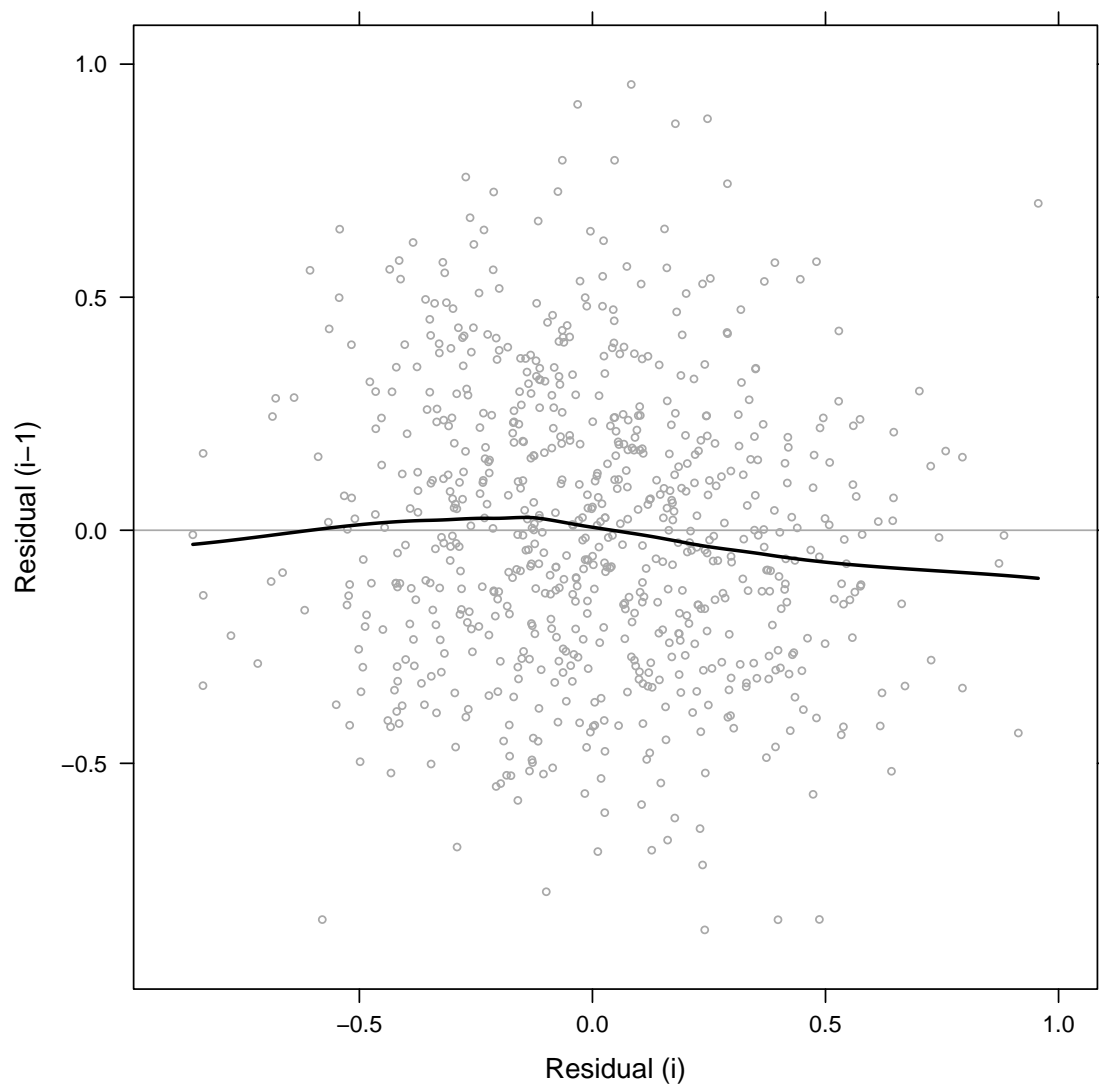


Figure 5.114. Lag 1 correlation plot for Family-Income fit residuals.

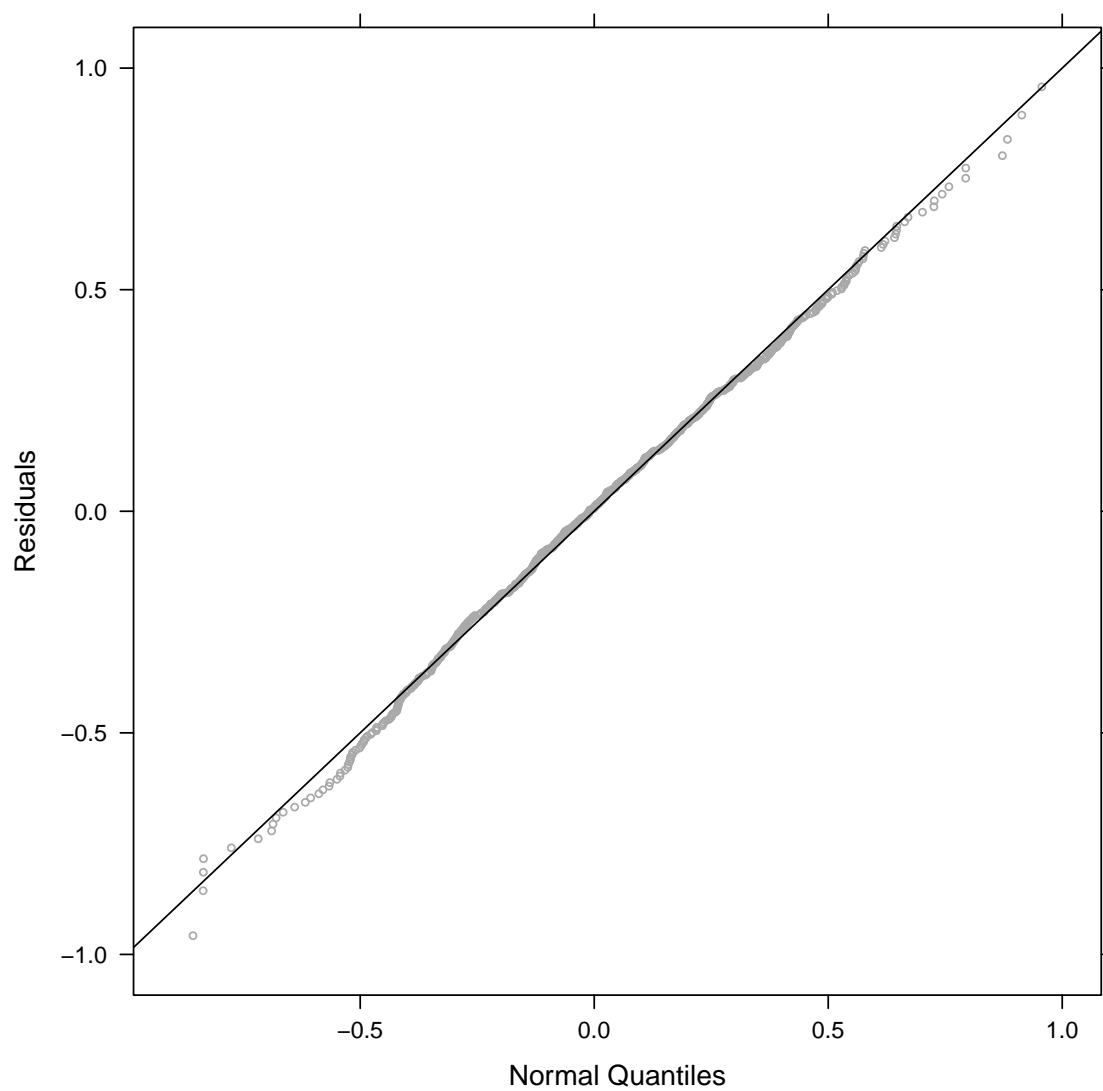


Figure 5.115. Normal quantile plot for Family-Income fit residuals.

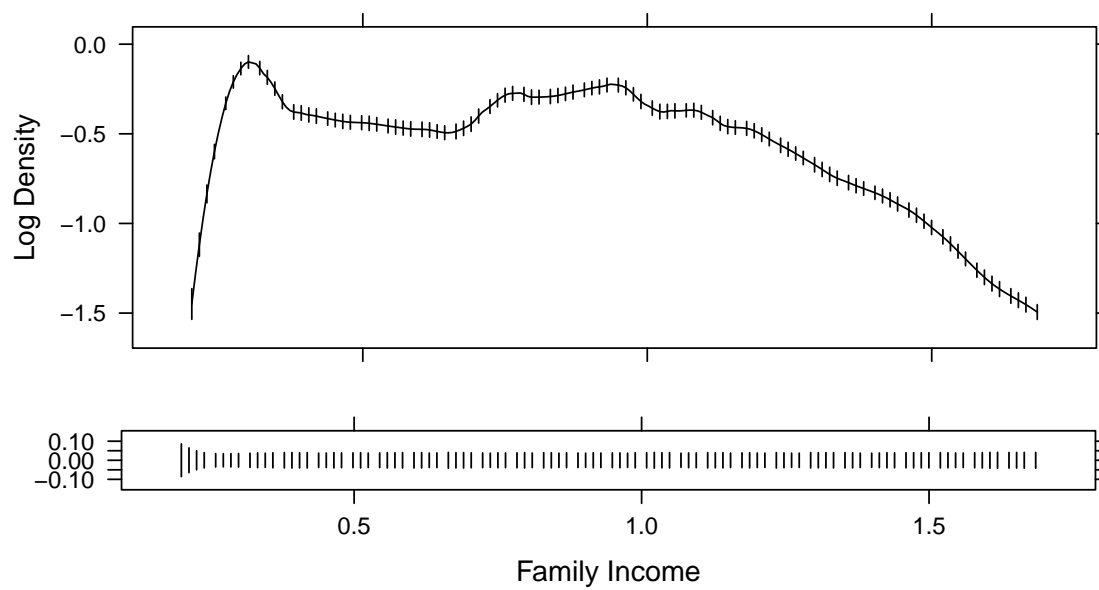


Figure 5.116. Pointwise 99% confidence limits for Family-Income log density estimate.

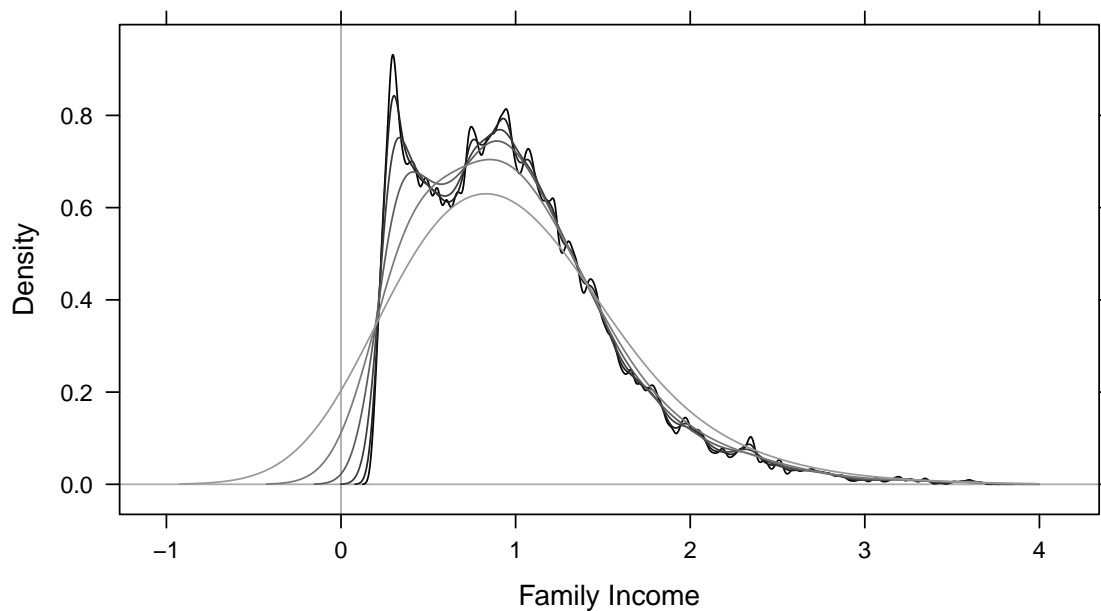


Figure 5.117. Kernel density estimate plot of Family-Income data. Fits for 6 different bandwidths ranging from $h = e^{-4}$ to $h = e^{-1}$ are shown with the darkest line corresponding to the smallest bandwidth and lightest line corresponding to the largest bandwidth.

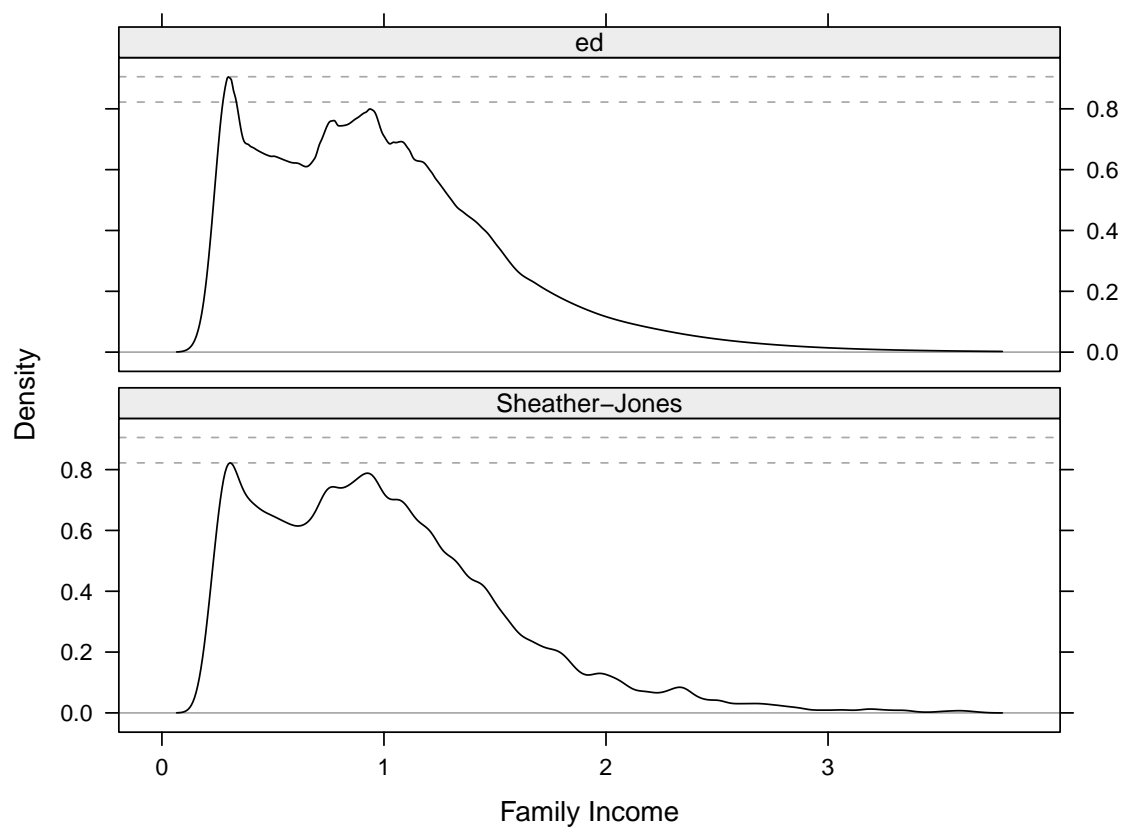


Figure 5.118. Final loess fit to Family-Income data on the identity scale, compared to the kernel density estimate with the Sheather-Jones bandwidth. The dashed lines indicate the differences in the peaks for the two estimates.

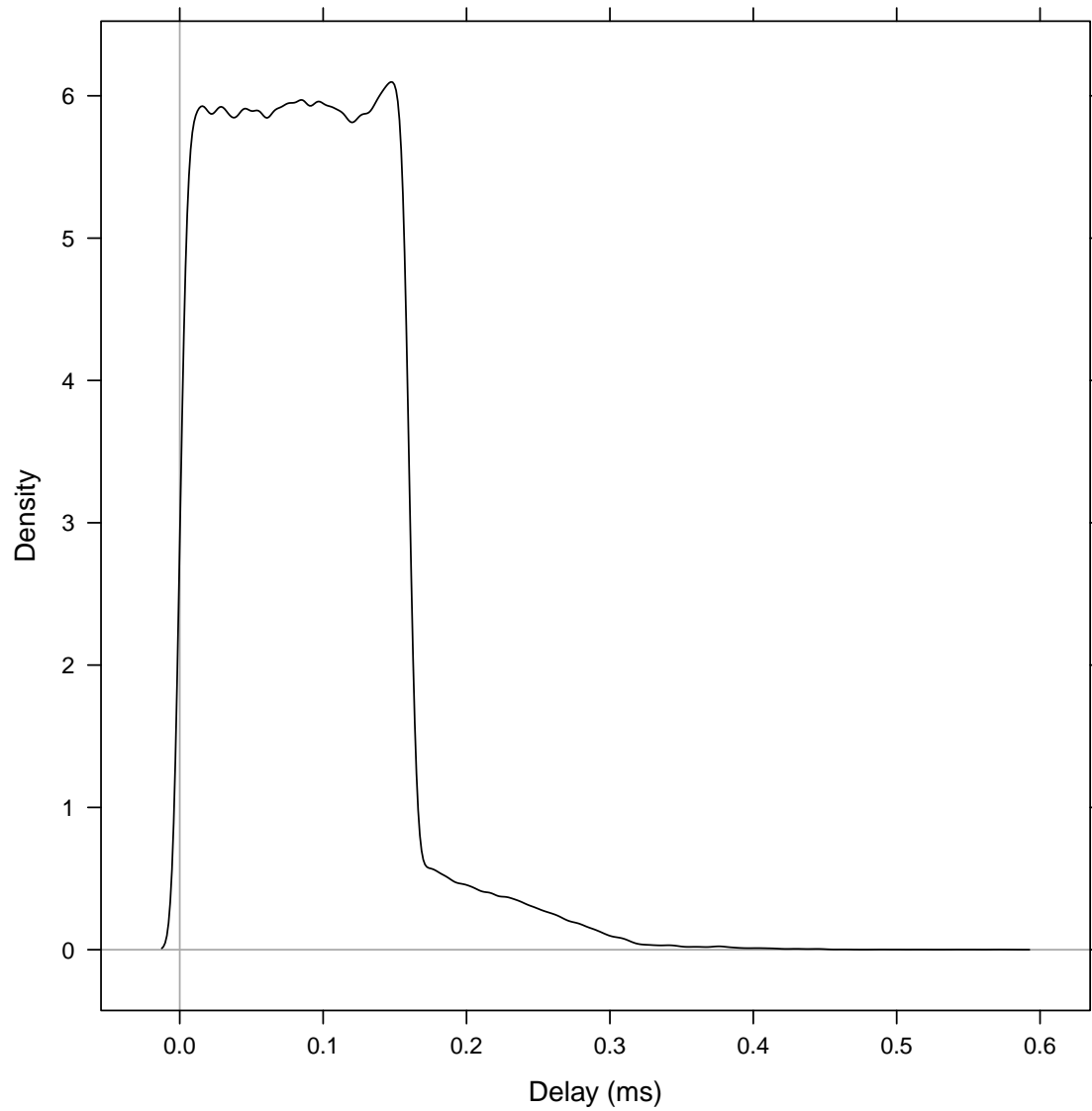


Figure 5.119. Kernel density estimate plot of waiting time in queue for Packet-Delay data.

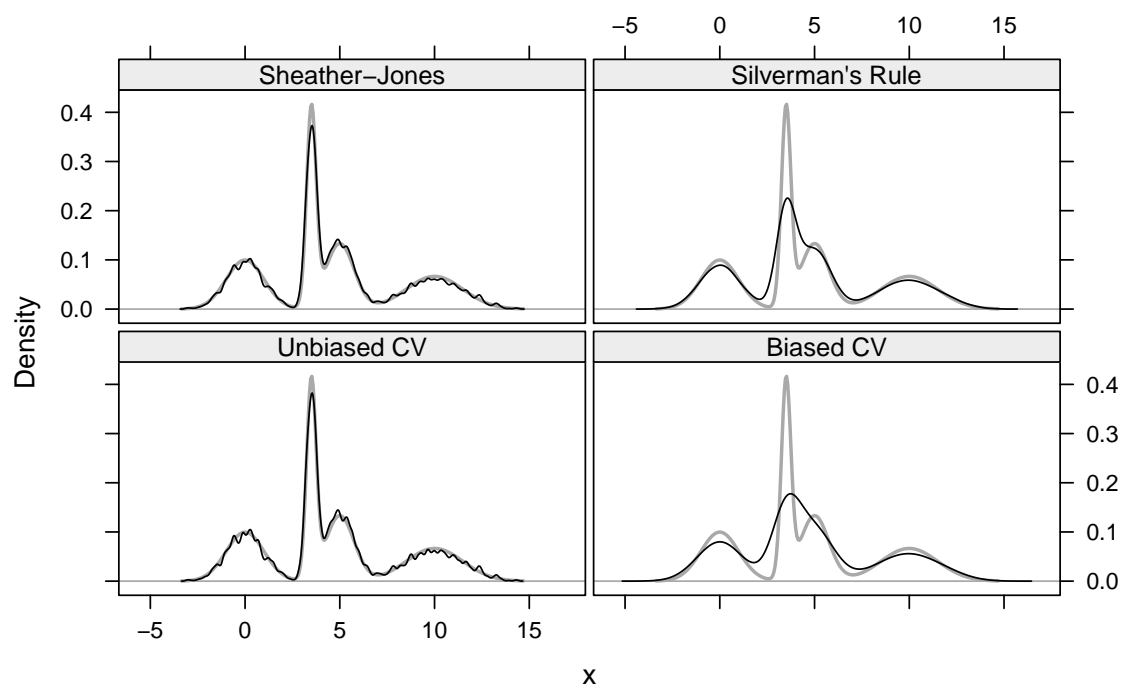


Figure 5.120. Kernel density estimate plots of 3,000 simulated values from the Normal-Mixture density. The gray line is the true density.

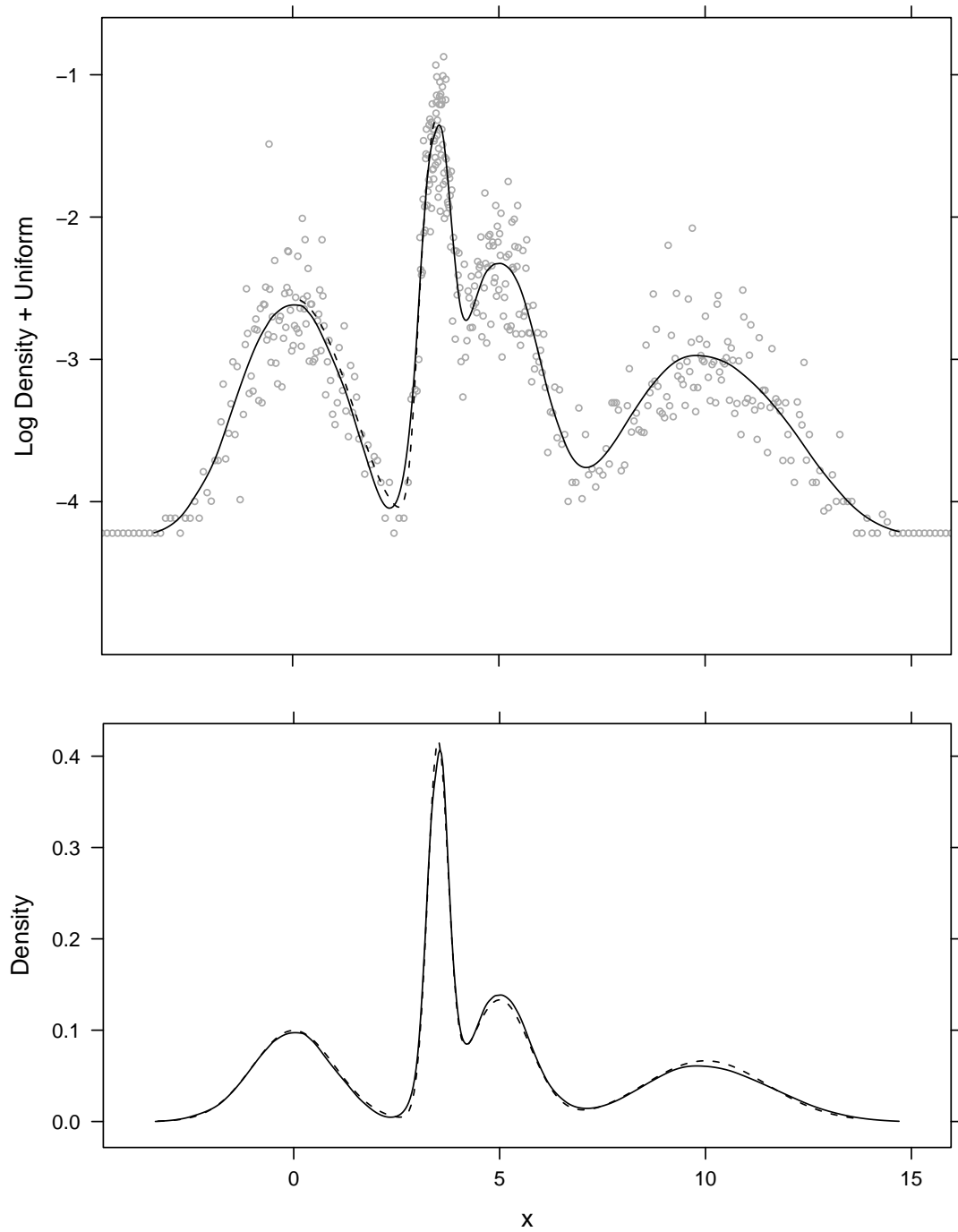


Figure 5.121. Plots of ed fit using grid augmentation. The blending parameter used was $\omega^* = 0.38$, and the mixture $g = 0.62f + 0.38u$ was fit with local cubic fitting and $\alpha = 0.16$. The top panel is the fit, $\log \hat{g}$, and the bottom panel is \hat{f} .

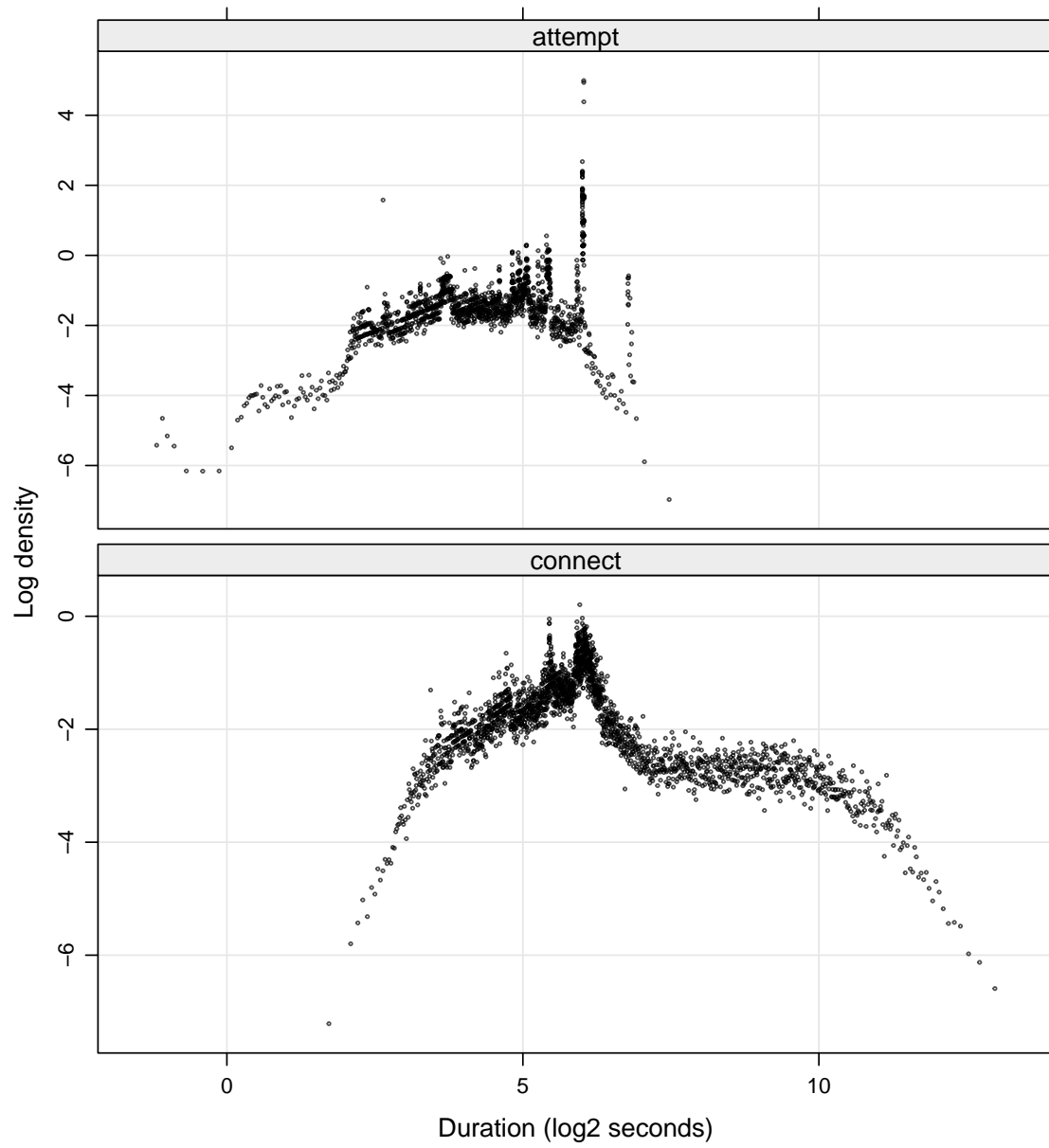


Figure 5.122. Ed raw estimates for VoIP call durations for attempted and connected calls.

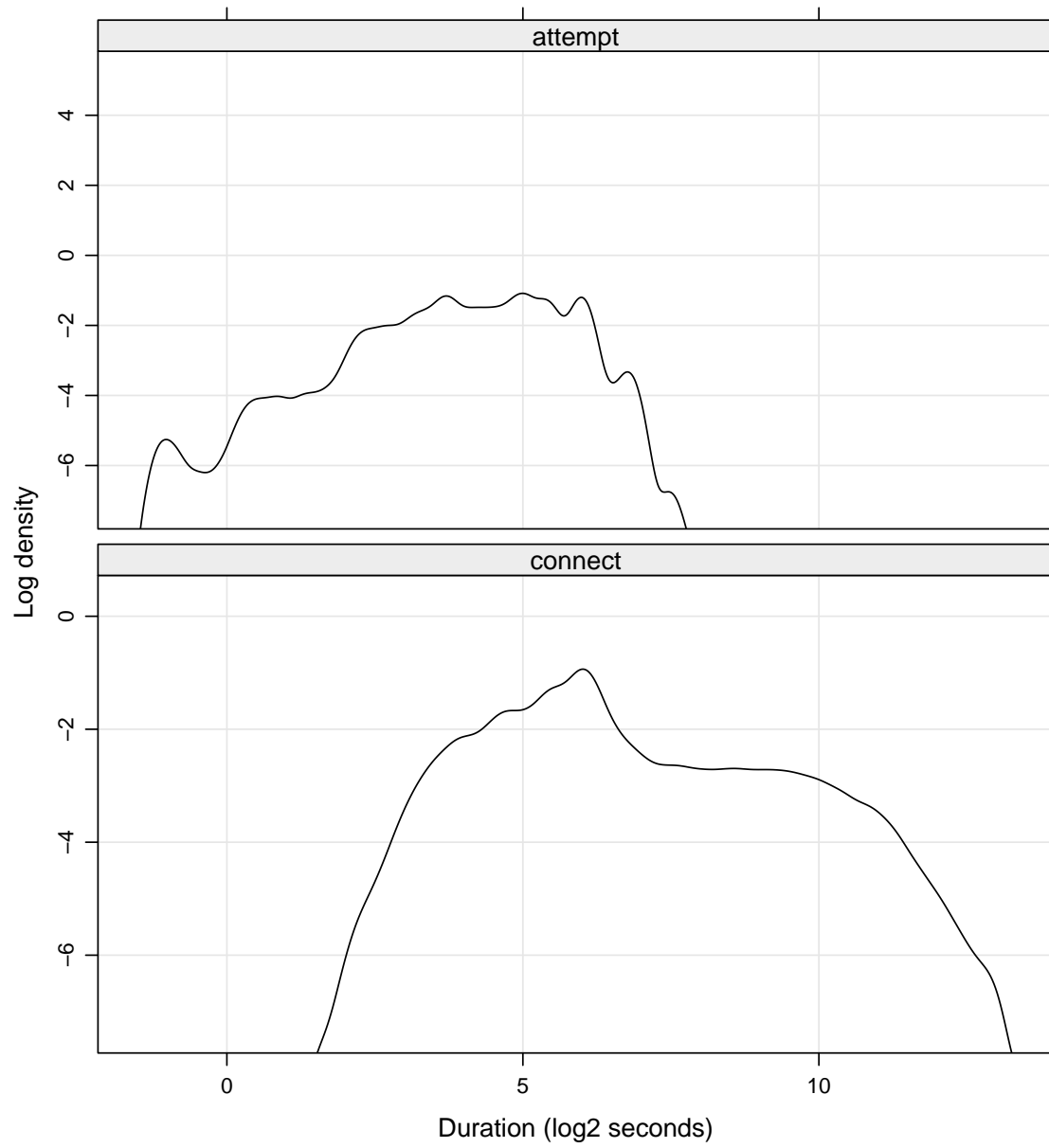


Figure 5.123. KDE estimate using Silverman's bandwidth on the log scale for VoIP call durations for attempted and connected calls. The plot scale is the same as that for figure 5.122

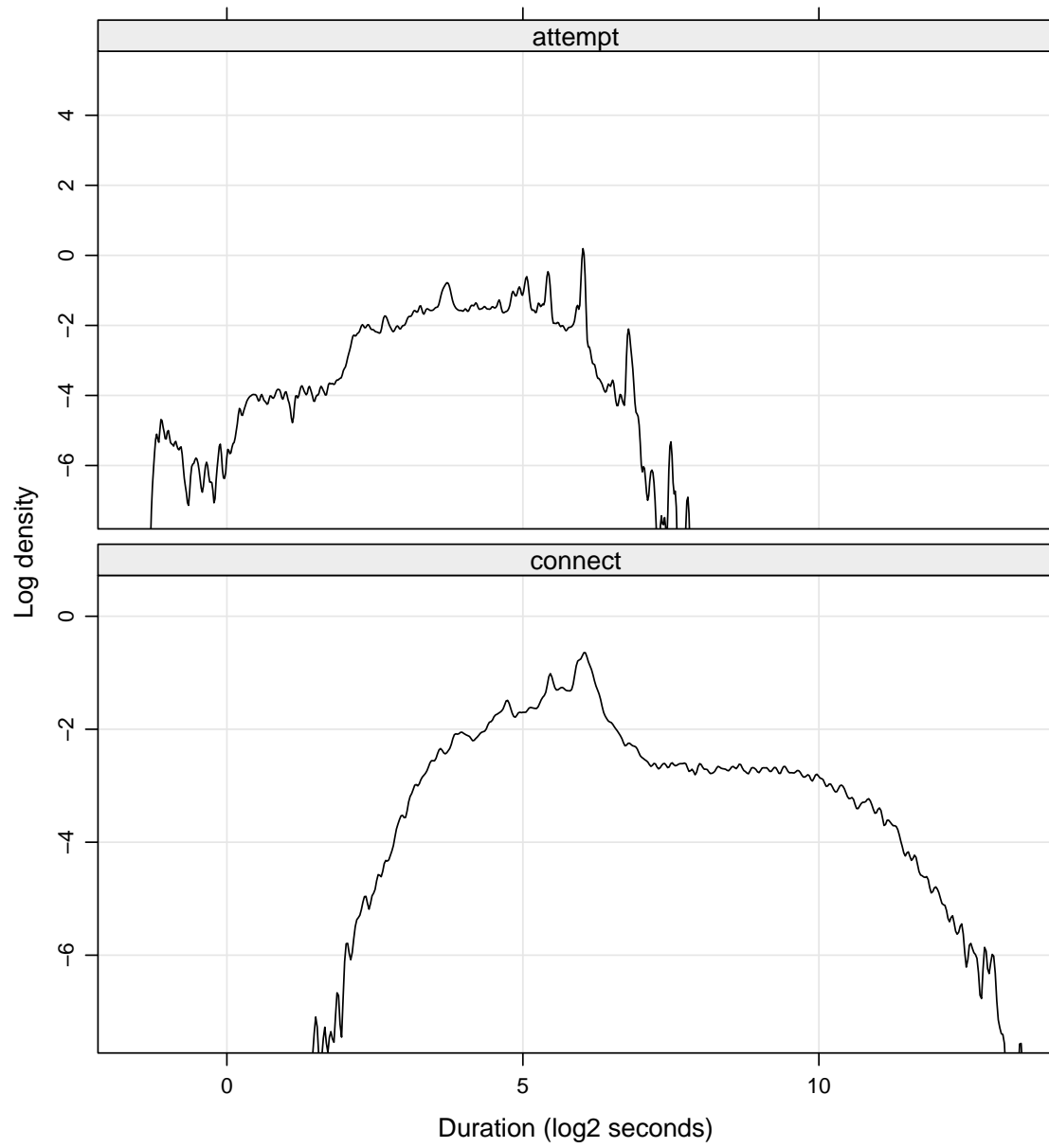


Figure 5.124. KDE estimate using bandwidths selected by unbiased cross-validation on the log scale for VoIP call durations for attempted and connected calls. The plot scale is the same as that for figure 5.122

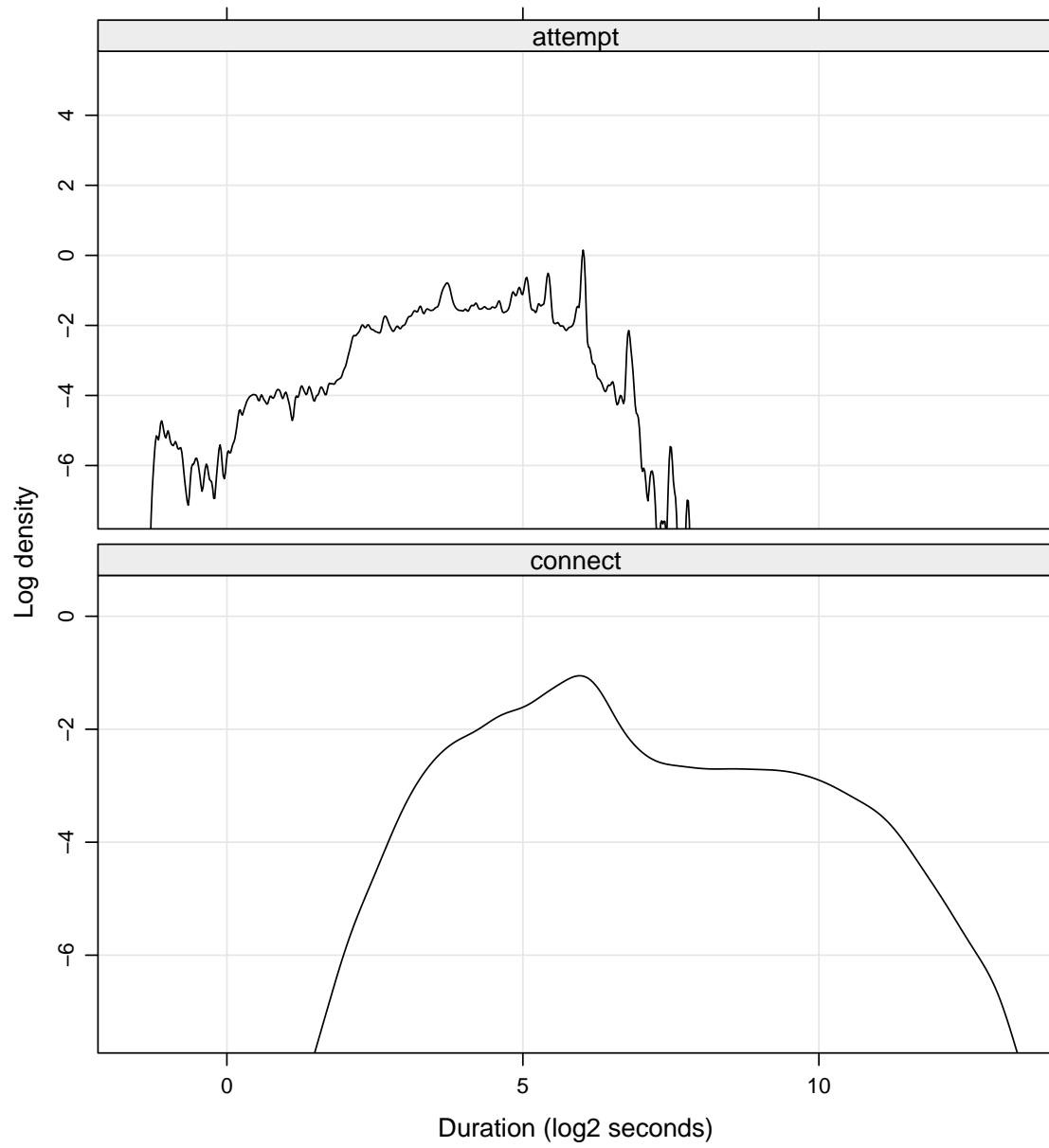


Figure 5.125. KDE estimate using bandwidths selected by unbiased cross-validation on the log scale for VoIP call durations for attempted and connected calls. The plot scale is the same as that for figure 5.122

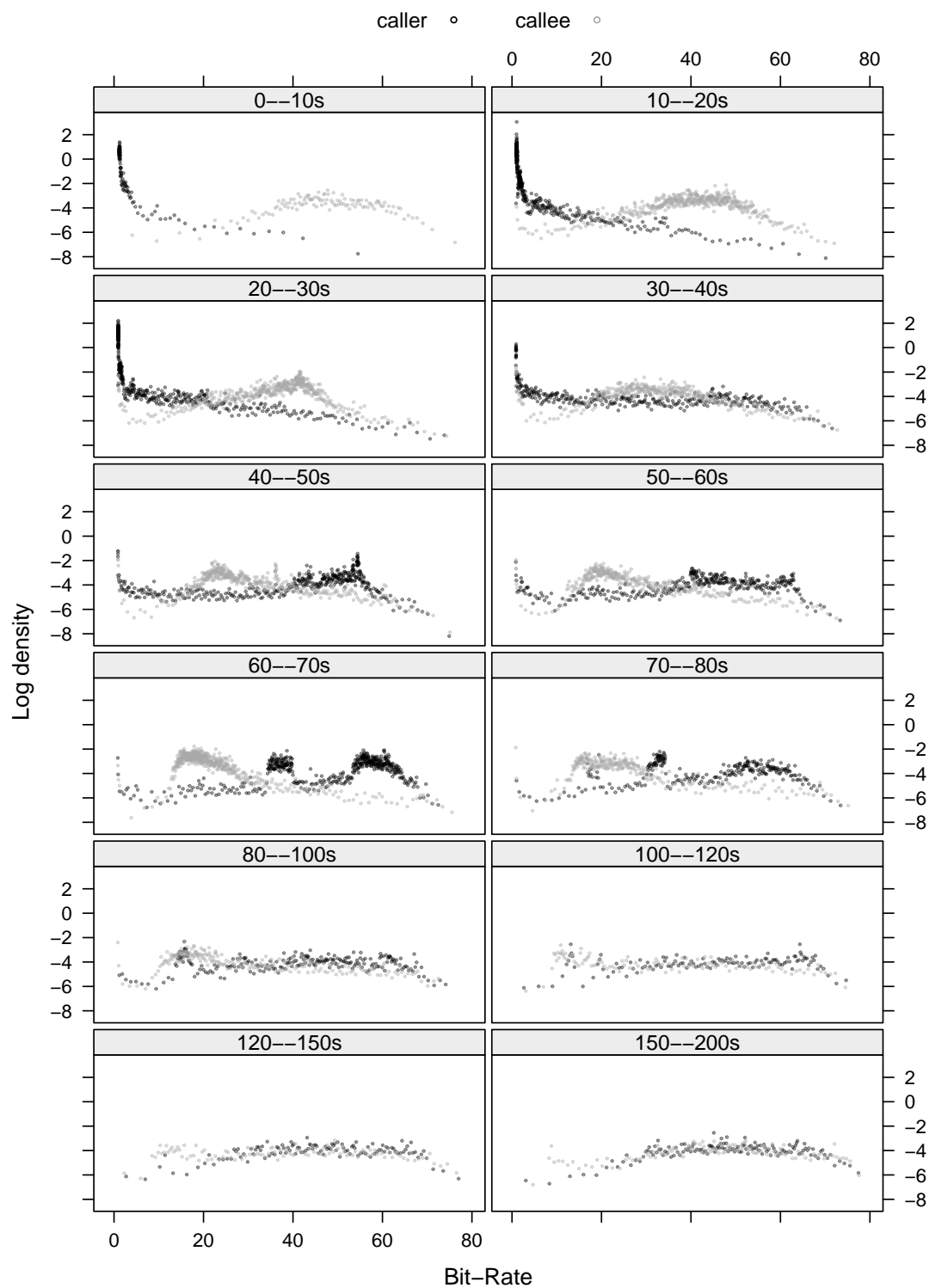


Figure 5.126. Log density raw estimates for bit-rate by call duration.

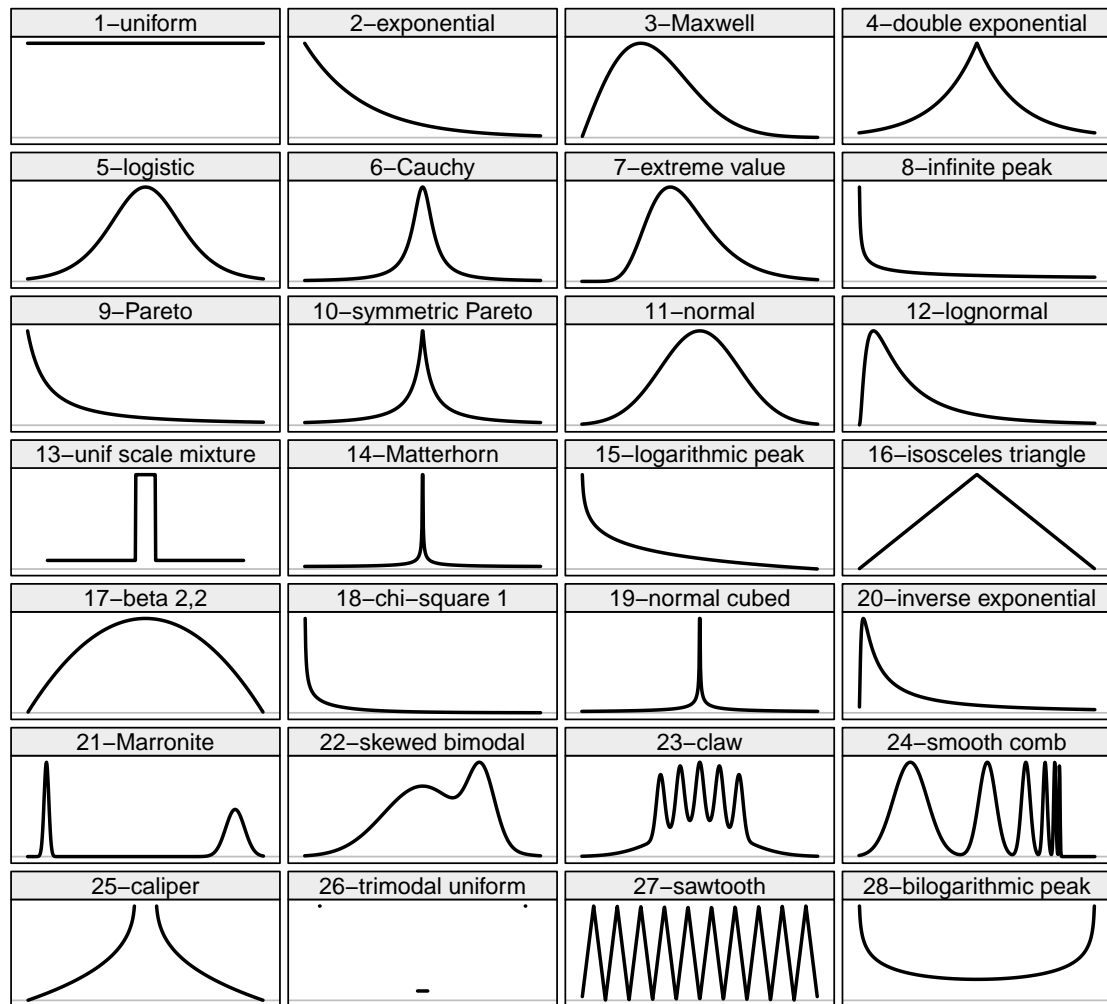


Figure 5.127. 28 benchmark densities from [Devroye and Berlinet \(1994\)](#).

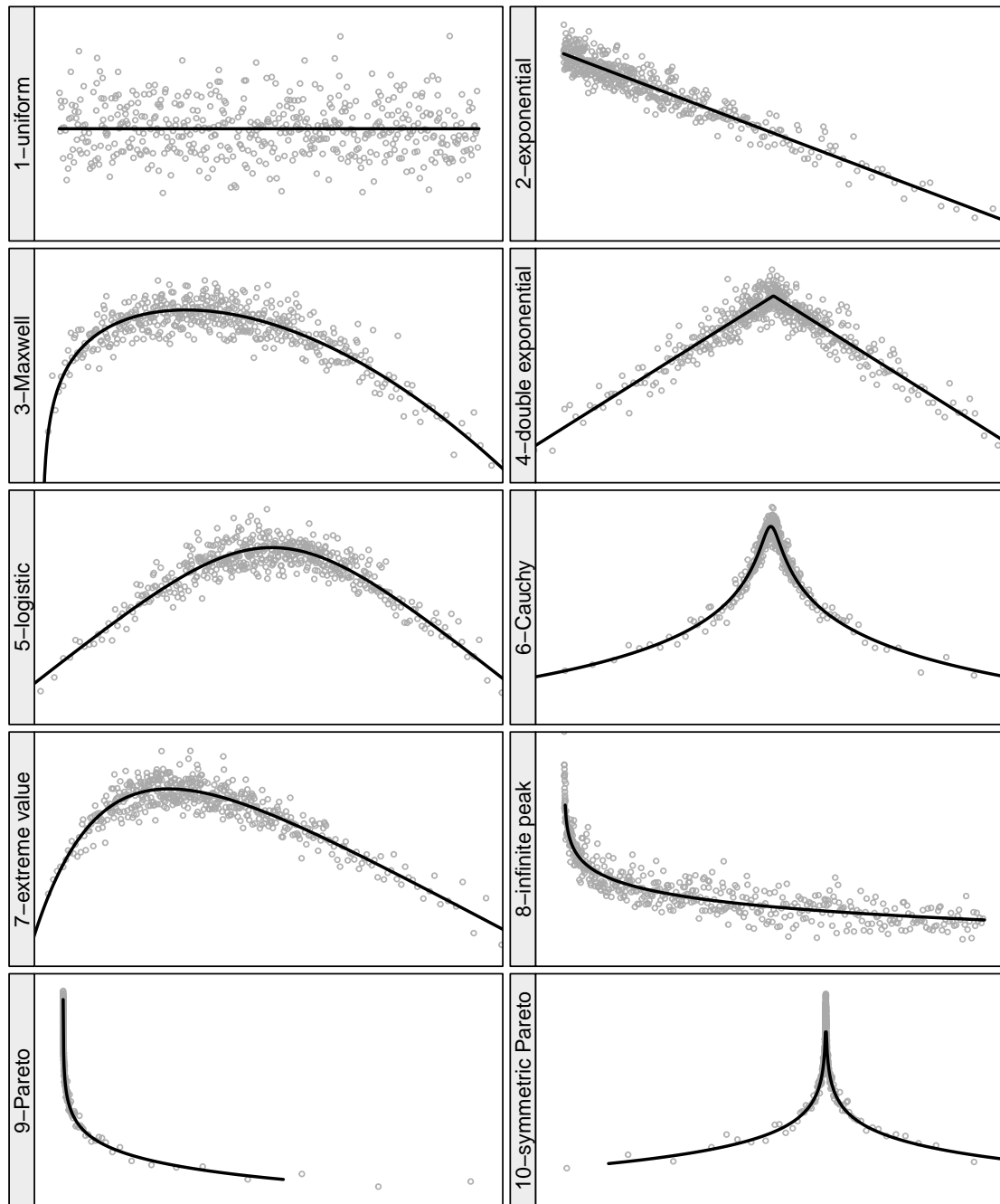


Figure 5.128. Ed raw estimates with true log density superposed for samples of size 5,000 from each of the 28 benchmark densities, with $\kappa = 10$ (page 1).

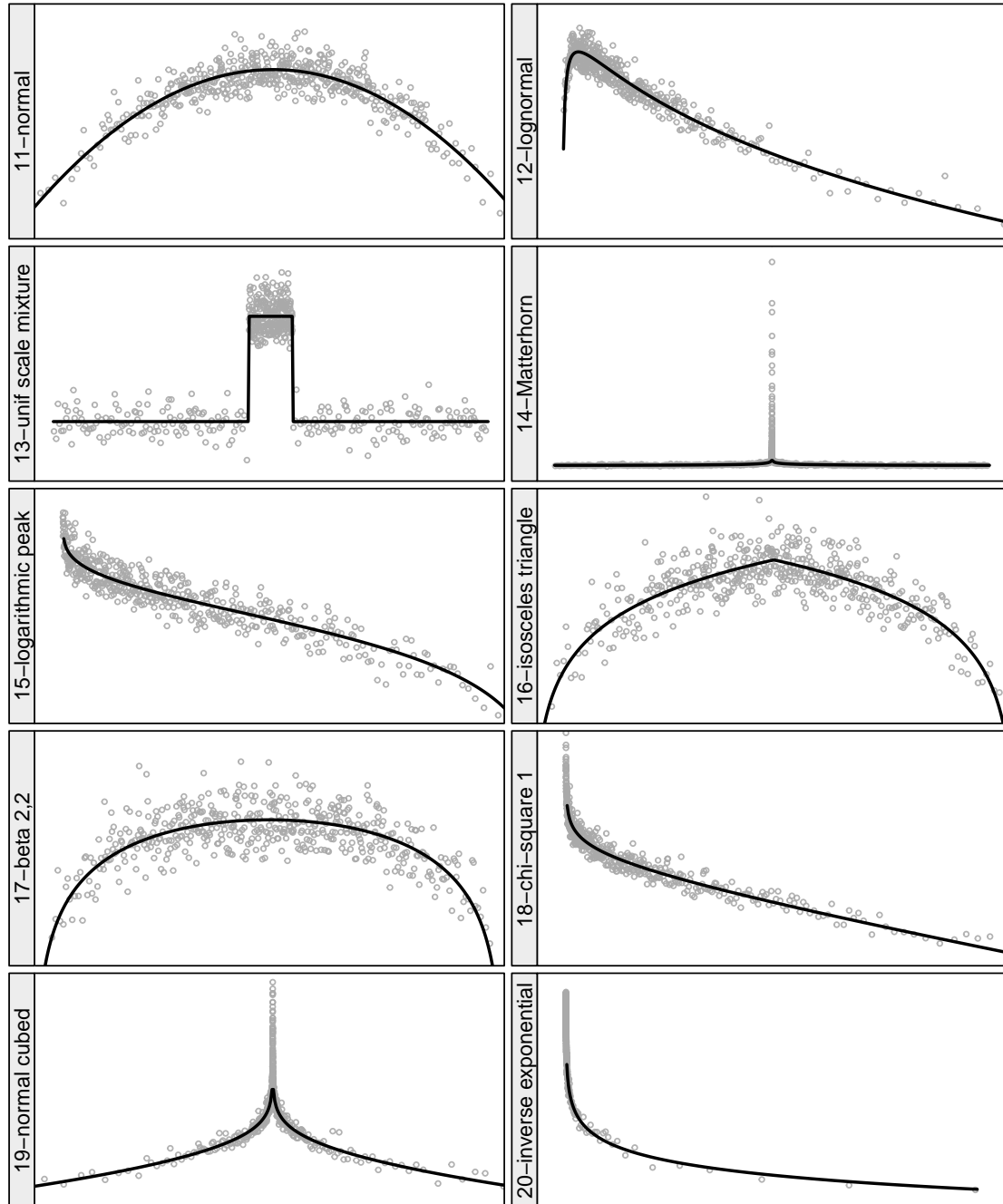


Figure 5.129. Ed raw estimates with true log density superposed for samples of size 5,000 from each of the 28 benchmark densities, with $\kappa = 10$ (page 2).

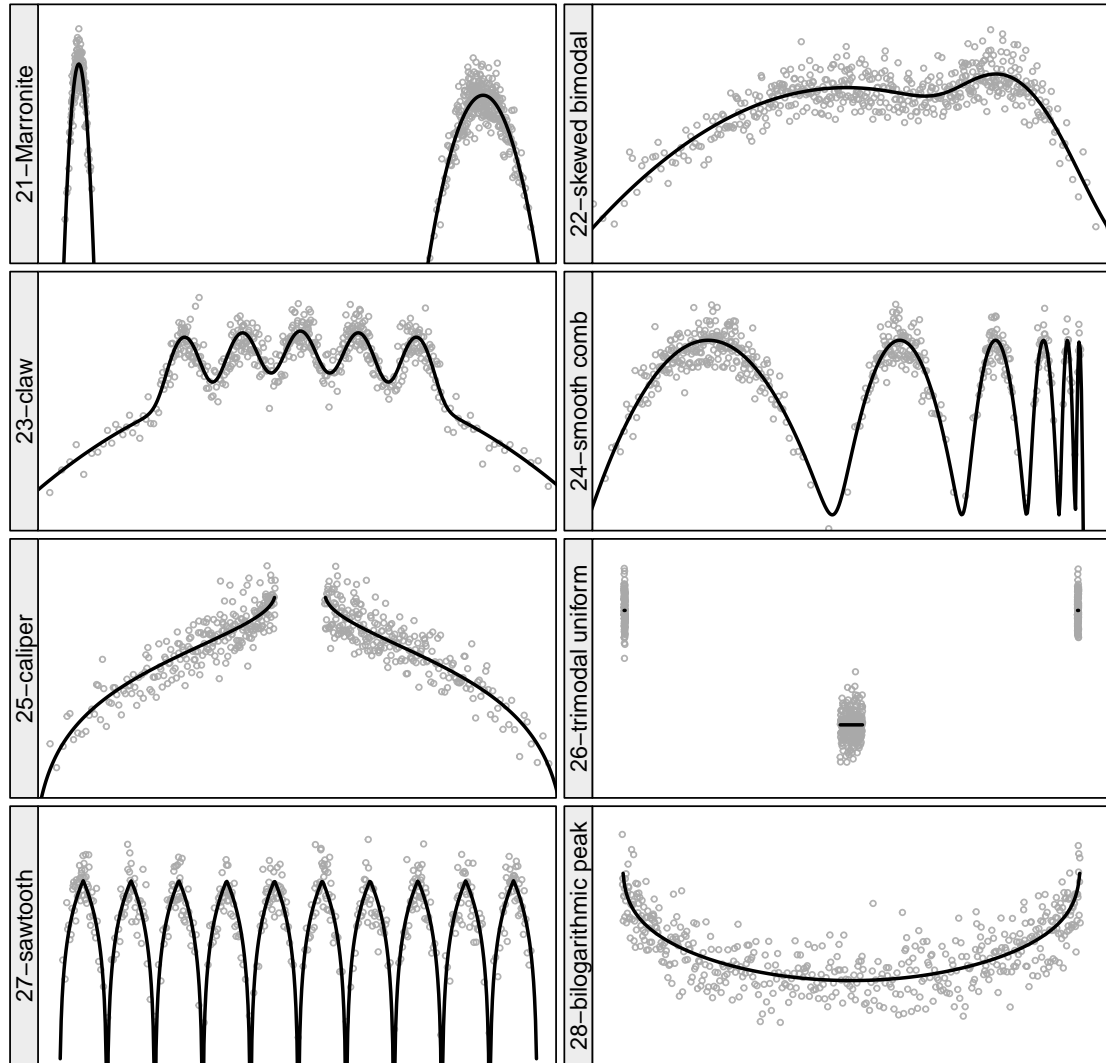


Figure 5.130. Ed raw estimates with true log density superposed for samples of size 5,000 from each of the 28 benchmark densities, with $\kappa = 10$ (page 3).

APPENDIX

Appendix: R package documentation

Package ‘stl2’

February 8, 2010

Type Package

Title Implementation of Seasonal-Trend Decomposition using Loess (STL) in R

Version 1.0

Date 20010-02-08

Author Ryan Hafen <rhafen@gmail.com>

Maintainer Ryan Hafen <rhafen@gmail.com>

Description Another implementation of STL that can handle NA values and higher order loess smoothing, with plotting methods

Depends lattice, yaImpute

Suggests operator

License GPL (version 2 or later)

R topics documented:

plot.cycle	1
plot.rembycycle	2
plot.seasonal	3
plot.stl2	4
plot.trend	4
raw, seasonal, trend, remainder, fitted, time	5
stl2	6

Index	11
-------	----

plot.cycle

Cycle-Subseries Plot for an stl2 Object

Description

Plots the seasonal component by cycle-subseries, with lines emanating from the midmean of the values within each cycle-subseries.

Usage

```
plot.cycle(layout=c(x$pars$n.p, 1), col="#0080ff",
  xlab="Time", ylab="Seasonal",
  panel= function(x, y, ...) {
    panel.segments(x, rep(midmean(y), length(x)), x, y, col=col)
  }, ...)
```

Arguments

x object of class "stl2".
 layout, col, xlab, ylab, type, ...
 parameters to be passed to xyplot.

Value

object of class "trellis"

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also

[stl2](#)

plot.rembycycle

Plot of Remainder Component by Cycle-Subseries

Description

Plots the remainder component by cycle-subseries with a loess line.

Usage

```
plot.rembycycle(x, col = "black", locol = "red", lolwd = 2, xlab =
  "Time", ylab = "Remainder", ...)
```

plot.seasonal

3

Arguments

`x` object of class "stl2".
`col, locol, lolwd, xlab, ylab, ...`
 parameters to be passed to `xyplot`. `locol` and `lolwd` are the line color and width for the loess line.

Value

object of class "trellis"

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also

[stl2](#)

`plot.seasonal`

Seasonal Diagnostic Plot for an stl2 Object

Description

Plots each cycle-subseries of the detrended data (or equivalently, seasonal plus remainder), with the mean subtracted. The fitted seasonal component is plotted as a line through the points.

Usage

```
plot.seasonal(x, col = c("black", "red"), lwd = 2, xlab = "Time",
  ylab = "Seasonal + Remainder", ...)
```

Arguments

`x` object of class "stl2".
`col, lwd, xlab, ylab, ...`
 parameters to be passed to `xyplot`.

Details

Helps decide how much of the variation in the data other than the trend should go into the seasonal component, and how much in the remainder.

Value

object of class "trellis"

4

plot.stl2

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also[stl2](#)

plot.stl2

*Lattice plot of the raw, seasonal, trend, and remainder components***Description**

Lattice plot of the raw, seasonal, trend, and remainder components. If post-trend smoothing was done, these components will be plotted instead of the trend component.

Usage

```
## S3 method for class 'stl2':
plot(x, ...)
```

Arguments

`x` object of class "stl2".
`...` parameters to be passed to `xyplot`.

Value

object of class "trellis"

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also[stl2](#)

plot.trend

5

*plot.trend**Trend Diagnostic Plot for an stl2 Object***Description**

Plots the trend+remainder with the trend component overlaid, and the remainder component in a separate panel.

Usage

```
plot.trend(x, xlab = "Time", ylab = "Trend", type = c("p", "l",
  "h"), scales = list(y = list(relation = "free")), lwd = c(1,
  2), ...)
```

Arguments

x object of class "stl2".
xlab, *ylab*, *type*, *scales*, *lwd*, ...
 parameters to be passed to *xyplot*.

Value

object of class "trellis"

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also

[stl2](#)

```
raw, seasonal, trend, remainder, fitted, time
```

Accessor functions for elements of an stl and stl2 object

Description

Retrieves the raw, seasonal, trend, remainder, or time components from an stl2 object. The methods `seasonal.stl`, ... also exist as a convenience for extracting components from R's `stl()`.

Usage

```

getraw(x)
## S3 method for class 'stl2':
seasonal(x)
## S3 method for class 'stl2':
trend(x)
## S3 method for class 'stl2':
remainder(x)
## S3 method for class 'stl2':
time(x)

## S3 method for class 'stl':
seasonal(x)
## S3 method for class 'stl':
trend(x)
## S3 method for class 'stl':
remainder(x)
## S3 method for class 'stl':
time(x)

```

Arguments

`x` Object of class "stl" or "stl2"

Value

Returns a vector of either the `getraw` time series, the `seasonal`, `trend`, or `remainder` components, or the `time` values of the time series. If `times` are requested but were not supplied in the initial `stl2` call, the `1:n` vector is returned, where `n` is the number of data points.

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also

[stl2](#)

Examples

```

co2.stl <- stl2(co2, t=as.vector(time(co2)), n.p=12, l.window=13,
t.window=19, s.window=35, s.degree=1, sub.labels=substr(month.name, 1, 3))

plot(seasonal(co2.stl))

```

stl2

7

stl2

*Seasonal Decomposition of Time Series by Loess***Description**

Decompose a time series into seasonal, trend and irregular components using `loess`, acronym STL. A new implementation of STL. Allows for NA values, local quadratic smoothing, post-trend smoothing, and endpoint blending. The usage is very similar to that of `stl`.

Usage

```
stl2(x, t = NULL, n.p, s.window, s.degree = 1,
     t.window = NULL, t.degree = 1,
     fc.window = NULL, fc.degree = NULL, fc.name = NULL,
     fc.jump = NULL, s.jump=ceiling(s.window/10),
     t.jump=ceiling(t.window/10), l.jump=ceiling(l.window/10),
     l.window = NULL, l.degree = t.degree, critfreq = 0.05,
     s.blend = 0, t.blend = 0, l.blend = t.blend, fc.blend=NULL,
     inner = 2, outer = 1, sub.labels = NULL,
     details = FALSE, ...)
```

Arguments

<code>x</code>	vector of time series values, in order of time. If <code>x</code> is a time series object, then <code>t</code> and <code>n.p</code> do not need to be specified, although they still can be.
<code>t</code>	times at which the time series values were observed. Not required.
<code>n.p</code>	Periodicity of the seasonal component. In R's <code>stl</code> function, this is the frequency of the time series.
<code>s.window</code>	either the character string "periodic" or the span (in lags) of the loess window for seasonal extraction, which should be odd. This has no default.
<code>s.degree</code>	degree of locally-fitted polynomial in seasonal extraction. Should be 0, 1, or 2.
<code>t.window</code>	the span (in lags) of the loess window for trend extraction, which should be odd. If <code>NULL</code> , the default, <code>nextodd(ceiling((1.5*period) / (1-(1.5/s.window))))</code> , is taken.
<code>t.degree</code>	degree of locally-fitted polynomial in trend extraction. Should be 0, 1, or 2.
<code>l.window</code>	the span (in lags) of the loess window of the low-pass filter used for each sub-series. Defaults to the smallest odd integer greater than or equal to <code>frequency(x)</code> which is recommended since it prevents competition between the trend and seasonal components. If not an odd integer its given value is increased to the next odd one.
<code>l.degree</code>	degree of locally-fitted polynomial for the subseries low-pass filter. Should be 0, 1, or 2.
<code>critfreq</code>	the critical frequency to use for automatic calculation of smoothing windows for the trend and high-pass filter.
<code>fc.window</code>	vector of lengths of windows for loess smoothings for other trend frequency components after the original STL decomposition has been obtained. The smoothing is applied to the data with the STL seasonal component removed. A frequency component is computed by a loess fit with the window length equal to

the first element of `fc.window`, the component is removed, another component is computed with the window length equal to the second element of `fc.window`, and so forth. In most cases, the values of the argument should be decreasing, that is, the frequency bands of the fitted components should increase. The robustness weights from original STL are used as weights in the loess fitting if specified.

<code>fc.degree</code>	vector of degrees of locally-fitted polynomial in the loess smoothings for the frequency components specified in <code>fc.window</code> . Values of 0, 1 and 2 are allowed. If the length of <code>fc.degree</code> is less than that of <code>fc.window</code> , the former is expanded to the length of the latter using <code>rep</code> ; thus, giving the value 1 specifies a degree of 1 for all components.
<code>fc.name</code>	vector of names of the post-trend smoothing operations specified by <code>fc.window</code> and <code>fc.degree</code> (optional).
<code>inner</code>	integer; the number of ‘inner’ (backfitting) iterations; usually very few (2) iterations suffice.
<code>outer</code>	integer; the number of ‘outer’ robustness iterations. Default is 0, but Recommended if outliers are present.
<code>sub.labels</code>	optional vector of length <code>n.p</code> that contains the labels of the subseries (such as month name, day of week, etc.), used for strip labels when plotting.
<code>details</code>	if TRUE, returns a list of the results of all the intermediate iterations.
<code>s.jump</code> , <code>t.jump</code> , <code>l.jump</code> , <code>fc.jump</code>	integers at least one to increase speed of the respective smoother. Linear interpolation happens between every <code>*.jumpth</code> value.
<code>s.blend</code> , <code>t.blend</code> , <code>l.blend</code> , <code>fc.blend</code>	vectors of proportion of blending to degree 0 polynomials at the endpoints of the series.

Details

The seasonal component is found by *loess* smoothing the seasonal sub-series (the series of all January values, ...); if `s.window = "periodic"` smoothing is effectively replaced by taking the mean. The seasonal values are removed, and the remainder smoothed to find the trend. The overall level is removed from the seasonal component and added to the trend component. This process is iterated a few times. The `remainder` component is the residuals from the seasonal plus trend fit.

Value

returns an object of class "`stl2`", containing

<code>data</code>	data frame containing all of the components: <code>raw</code> , <code>seasonal</code> , <code>trend</code> , <code>remainder</code> , <code>weights</code>
<code>pars</code>	list of parameters used in the procedure
<code>fc.number</code>	number of post-trend frequency components fitted
<code>fc</code>	data frame of the post-trend frequency components
<code>time</code>	vector of time values corresponding to the raw values, if specified
<code>n</code>	the number of observations
<code>sub.labels</code>	the cycle-subseries labels

stl2

9

Note

This is a complete re-implementation of the STL algorithm, with the loess part in C and the rest in R. Moving a lot of the code to R makes it easier to experiment with the method at a very minimal speed cost.

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, **6**, 3–73.

See Also

[plot.stl2](#) for plotting the components.

[getraw](#), [seasonal](#), [trend](#), [remainder](#) for accessing the components.

Examples

```
co2.stl <- stl2(co2, t=as.vector(time(co2)), n.p=12,
               l.window=13, t.window=19, s.window=35, s.degree=1,
               sub.labels=substr(month.name, 1, 3))

plot(co2.stl, ylab="CO2 Concentration (ppm)", xlab="Time (years)")
plot.seasonal(co2.stl)
plot.trend(co2.stl)
plot.cycle(co2.stl)
plot.rembycycle(co2.stl)

# with NAs
y <- co2
y[201:224] <- NA

y.stl <- stl2(y, l.window=13, t.window=19, s.window=35,
              s.degree=1, sub.labels=substr(month.name, 1, 3))

plot(y.stl, ylab="CO2 Concentration (ppm)", xlab="Time (years)")
plot.seasonal(y.stl)
plot.trend(y.stl)
plot.cycle(y.stl)
plot.rembycycle(y.stl)

# if you don't want to use a time series object:
y.stl <- stl2(y, t=as.vector(time(y)), n.p=12,
              l.window=13, t.window=19, s.window=35, s.degree=1,
              sub.labels=substr(month.name, 1, 3))

# with an outlier
y2 <- co2
y2[200] <- 300
```

10

stl2

```
y2.stl <- stl2(y2, t=as.vector(time(y2)), n.p=12,
  l.window=13, t.window=19, s.window=35, s.degree=1,
  sub.labels=substr(month.name, 1, 3), outer=10)

plot(y2.stl, ylab="CO2 Concentration (ppm)", xlab="Time (years)")
plot.seasonal(y2.stl)
plot.trend(y2.stl)
plot.cycle(y2.stl)
plot.rembycycle(y2.stl)

# compare to R's stl

x1 <- stl2(co2, t=as.vector(time(co2)), n.p=12,
  l.window=13, t.window=19, s.window=11, s.degree=1,
  sub.labels=substr(month.name, 1, 3))

x2 <- stl(co2, l.window=13, t.window=19, s.window=11, s.degree=1)

plot(seasonal(x1) - seasonal(x2))
# ...
```

Index

```

getraw,8
getraw(raw, seasonal, trend,
      remainder, fitted, time),
5

plot.cycle,1
plot.rembycycle,2
plot.seasonal,3
plot.stl2,4,8
plot.trend,4

raw, seasonal, trend,
      remainder, fitted, time,5
remainder,8
remainder(raw, seasonal, trend,
          remainder, fitted, time),
5

seasonal,8
seasonal(raw, seasonal, trend,
        remainder, fitted, time),
5
stl2,2-5,6,6

time.stl2(raw, seasonal, trend,
         remainder, fitted, time),
5
trend,8
trend(raw, seasonal, trend,
     remainder, fitted, time),
5

```


Package ‘operator’

February 9, 2010

Version 1.0

Title Loess and STL operator matrix computation for time series (equally-spaced design)

Author Ryan Hafen <rhafen@purdue.edu>

Maintainer Ryan Hafen <rhafen@purdue.edu>

Description Computes the operator matrices for local polynomial smoothing methods (loess and STL) applied to time series (or any equally-spaced design setting). Includes capabilities for model selection, prediction, and confidence intervals.

Depends lattice, stl2

License GPL (version 2 or later)

Note Remember that this is experimental software distributed free of charge and comes with no warranty! Exercise professional caution.

R topics documented:

anova.op	2
cp	3
frequ	4
frequlo	6
loessOp	7
opVar	9
plot.frequ	10
plotOp	10
plotVar	11
predict.op	12
respsim	14
stlOp	15

Index	19
--------------	-----------

anova.op

*ANOVA for objects of class list("op")***Description**

ANOVA table calculation for objects of class "op"

Usage

```
## S3 method for class 'op':
anova(mod1, mod2, y)
```

Arguments

mod1	null model operator matrix of class "op".
mod2	alternative model operator matrix of class "op".
y	data that the models are applied to

Details

Calculates ANOVA as described in the reference below.

Value

An object of class "anova".

Author(s)

Ryan Hafen

References

W.<a0>S. Cleveland and S.<a0>J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988.

See Also

[loessOp](#), [loessOp](#).

Examples

```
# a simulated example with periodic component
rf <- function(x) {
  n <- length(x)
  sin(x * 2 * pi/200) + rnorm(n, sd = 0.5) + rep(c(0.5, 0.25,
    0, -0.25, -0.5, -0.25, 0), ceiling(n/7))[1:n]
}
n <- 200
x <- c(1:n)
set.seed(8765)
ysim <- rf(x)

# null model: no seasonality
```

cp

3

```

lop <- loessOp(n, span=105, degree=2)

# alternative model: seasonality
sop <- stlOp(n, n.p=7, t.window=105, t.degree=2,
  s.window="periodic")

anova(sop$fit, lop, ysim)
# result:

```

*cp**Calculate Mallows Cp Statistic for Linear Operator***Description**

Calculates Mallows Cp statistic for linear operators such as loess and the STL time series modeling method.

Usage

```

## S3 method for class 'op':
cp(op, y, sigmasq = 1)
## S3 method for class 'stlop':
cp(stlop, y, sigmasq = 1)
## S3 method for class 'loess':
cp(ll, y, sigmasq = 1)

```

Arguments

<i>op</i> , <i>stlop</i>	objects of class "op" or "stlop" which provide operator matrices from which the Cp statistic is calculated. If an ARMA model was included in the "stlop" object, this will be used.
<i>y</i>	the data that the operator is to be applied to.
<i>sigmasq</i>	an unbiased estimate of the residual variance.

Details

The Cp statistic is calculated as

$$\hat{M} = RSS/\hat{\sigma}^2 - tr(\Lambda) + \nu$$

where RSS is the residual sum of squares, $\hat{\sigma}^2$ is the estimated residual variance of an unbiased model, Λ is the regulator matrix $(I - L)'(I - L)$, and ν is $tr(L)$.

If an unbiased estimate of the residual variance is not known, the function can be called with default `sigmasq = 1` and then C_p can be reconstructed using the output of the function.

Value

A data frame consisting of the smoother degrees of freedom `df`, which is ν in the above equation, the C_p statistic `cp`, the estimated residual standard deviation $\hat{\sigma}$ as `sigmahat`, the trace of the regulator matrix, $tr(\Lambda)$ as `delta1`, and the residual sum of squares.

Author(s)

Ryan Hafen

References

W.<a0>S. Cleveland and S.<a0>J. Devlin. Locally weighted regression: An approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988.

Examples

```
n <- 500
x <- c(1:n)
set.seed(2456)
y <- sin(x/100) + rnorm(n, sd=0.5)

cpres <- NULL
for(sp in seq(251, 351, by=5)) {
  ll <- loess(y ~ x, degree=2, span=sp/n)
  cpres <- rbind(cpres,
    cp(ll, sigmasq=0.5^2)
  )
}

xyplot(cp ~ df, data=cpres)
```

freqr

Frequency response calculation for STL decomposition

Description

Calculates the frequency response function for the seasonal, trend, and high-pass filters of an STL decomposition at a specified grid of fourier frequencies for the symmetric smoothing case. This can be used to visualize the power transfer function or phase shift function to evaluate the spectral properties of an STL model.

Usage

```
## Default S3 method:
freqr(n.p, s.window, s.degree = 0,
      t.window = NULL, t.degree = 1, l.window = nextodd(n.p),
      l.degree = t.degree, periodic = FALSE, M = 400)
## S3 method for class 'stl':
freqr(x, ...)
## S3 method for class 'stl2':
freqr(x, ...)
## S3 method for class 'stlop':
freqr(x, ...)
```

Arguments

`n.p`, `s.window`, `s.degree`, `t.window`, `t.degree`, `l.window`, `l.degree`, `periodic`
STL parameters (see [stlOp](#)).

`x` object of class "stl", "stl2", or "stlop". Parameters from these objects are passed on to `freqr.default`.

`M` density of the grid of fourier frequencies, which range from 0 to 0.5.

Value

A data frame of class "freqr" consisting of frequencies `fk` and frequency response values `frfv`.

Note

To investigate the spectral properties of an STL model at an asymmetric smoothing location, one can use the [freqrlo](#) function with a "stlop" object to obtain the frequency response function at any smoothing point.

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6, 3-73.

R.<a0>Shumway and D.<a0>Stoffer. *Time series analysis and its applications*. Springer Verlag, 2000.

See Also

[stlOp](#), [freqrlo](#), [plot.freqr](#)

Examples

```
co2.stl <- stl2(co2, t=as.vector(time(co2)), n.p=12,
               s.window=35, s.degree=1, t.degree=1,
               s.blend=0.5, t.blend=0.5)

co2op <- stlOp(co2.stl)
```

```
# symmetric filter
co2fr <- fregir(co2op, M=1000)
# seasonal, trend, and overall filter at endpoint
co2fr1 <- fregirlo(co2op$fit, at=1, M=800)
co2fr1.s <- fregirlo(co2op$seas, at=1)
co2fr1.t <- fregirlo(co2op$trend, at=1)
# overall filter at 10 points from end and middle
co2fr10 <- fregirlo(co2op$fit, at=10, M=800)
co2frm <- fregirlo(co2op$fit, at=234, M=800)

plot(co2fr, aspect=0.25)
plot(co2fr1, aspect=0.25)
plot(co2fr1.s, aspect=0.25)
plot(co2fr1.t, aspect=0.25)
plot(co2frm, aspect=0.25)
```

fregirlo

Frequency response calculation for loess (or general) smoothing matrix

Description

Returns the frequency response function for given loess smoothing parameters or for a specified operator matrix. Can return values for asymmetric fits.

Usage

```
## Default S3 method:
fregirlo(span, degree, M = 400, at = "symmetric")
## S3 method for class 'op':
fregirlo(op, M = 400, at = "symmetric")
## S3 method for class 'loess':
fregirlo(x, M = 400, at = "symmetric")
```

Arguments

span	span of loess smooth (must be odd integer).
degree	degree of loess smooth (must be 0, 1, or 2).
M	density of the grid of fourier frequencies, which range from 0 to 0.5.
at	at which design point the frequency response is to be calculated. This is interpreted as number of points from the end, with 1 being the endpoint. If an operator matrix is supplied, it is the row of the matrix to use. For symmetric neighborhoods, the value "symmetric" can be specified.
op	object of class "op". The operator matrix in this object is used to calculate the frequency response.
x	object of class "loess". Parameters from this objects are passed on to <code>fregirlo.default</code> .

Value

A data frame of class "fregir" consisting of frequencies `fk` and frequency response values `frfv`.

loessOp

7

Author(s)

Ryan Hafen

References

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

R. Shumway and D. Stoffer. *Time series analysis and its applications*. Springer Verlag, 2000.

See Also

[freqr](#), [plot.freqr](#), [loess](#), [loessOp](#)

Examples

```
# symmetric loess smooth
a <- freqrlo(l1, 2)
plot(a, strip=FALSE, axis=axis.period)

# endpoint loess smooth
b <- freqrlo(l1, 2, at=1)
plot(b, strip=FALSE, axis=axis.period)

# accessing from loess
l1 <- loess(rnorm(100) ~ c(1:100), degree=1, span=15/100)
l1.fr <- freqrlo(l1)
plot(l1.fr)

# STL overall operator
co2.stl <- stl2(co2, t=as.vector(time(co2)), n.p=12,
  s.window=35, s.degree=1, t.degree=1,
  s.blend=0.5, t.blend=0.5)

co2op <- stlOp(co2.stl)
co2fr1 <- freqrlo(co2op$fit, at=1, M=800)
plot(co2fr1, aspect=0.25)
```

loessOp

*Loess operator matrix calculation***Description**

Calculates the hat matrix, L , used to obtain fitted values for given loess smoothing parameters, for an equally-spaced design $1, \dots, n$. Also allows extrapolation beyond the design points.

Usage

```
loessOp(n, span, degree = 1, blend = 0, at = 1:n,
  stats = TRUE)
```

Arguments

<code>n</code>	number of equally-spaced design points, assigned values $1, \dots, n$.
<code>span</code>	number of design points to be used in the local neighborhood – must be odd.
<code>degree</code>	degree of local polynomial (currently can be either 0, 1, or 2).
<code>blend</code>	the amount of blending to degree 0 smoothing at the endpoints.
<code>at</code>	which rows of the operator matrix to calculate.
<code>stats</code>	whether or not to calculate auxiliary statistics.

Details

If all that is desired is the loess estimate, it is more efficient and flexible to use the built-in `loess()`. The main purpose of this function is its use in `stlOp` or subsequent smoothings after using `stlOp()`.

Value

A list of class "op".

<code>O</code>	the operator matrix of dimension <code>length(at) × n</code> .
<code>at</code>	at as specified in <code>loessOp()</code> .
<code>span</code>	span as specified in <code>loessOp()</code> .
<code>deg</code>	deg as specified in <code>loessOp()</code> .
<code>var</code>	the squared l2 norm of each row in <code>O</code> – used in variance calculations.
<code>stats</code>	only if <code>stats=TRUE</code> . This is a list with the regulator matrix <code>lambda=(I-O)'(I-O)</code> , the equivalent number of parameters <code>enp (tr O'O)</code> , <code>delta1 (tr lambda)</code> , <code>delta2 (tr lambda^2)</code> , and <code>trace.hat (tr O)</code> .

Note

This requires $o(n^2)$ storage and computation. Rows of the operator matrix corresponding to interior design points will be identical, so keep this in mind.

Author(s)

Ryan Hafen

References

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

See Also

`stlOp`, `plotVar`, `plotOp`

`opVar`

9

Examples

```

n <- 200
L <- loessOp(n, span=91, degree=2, at=c(-9:(n+10)))
plotVar(L)
plotOp(L)

## get fitted values for some data
x <- c(1:n)
# generate some data
y <- sin(x*2*pi/n) + rnorm(x, sd=0.5)
# get the fitted values
yhat <- predict(L, y)
# another way: yhat <- L$O %*% y

plot(x, y, xlim=range(L$at))
lines(L$at, yhat, col="red")

```

`opVar`*Compute sum of squared values in each row of operator matrix***Description**

Compute sum of squared values in each row of operator matrix

Usage

```
opVar(O)
```

Arguments

`O` a matrix.

Value

vector of length `nrow(O)`.

Author(s)

Ryan Hafen

See Also

`stlOp`, `loessOp`, `plotVar`

10

plotOp

plot.freqr	<i>Plot power transfer function of objects of class list("freqr")</i>
------------	---

Description

Plots the power transfer function of objects of class "freqr".

Usage

```
## S3 method for class 'freqr':
plot(x, critfreq = NA,
     xlab = "Frequency", ylab = "Transfer Function",
     layout = c(1, length(unique(x$which))),
     type = c("g", "l"), as.table = TRUE,
     panel = freqrPanel, n.p = NULL, ...)
```

Arguments

x	objects of class "freqr".
critfreq	if specified, calculates the critical frequencies for each component plotted that yield a transfer function of critfreq. Then a vertical line is plotted where this occurs.
n.p	if specifying critfreq, must also specify n.p (see stlOp).
xlab, ylab, layout, type, as.table, panel, ...	arguments passed on to <code>xyplot()</code> .

Value

An object of class "trellis".

Author(s)

Ryan Hafen

See Also

[freqr](#), [freqrlo](#)

plotOp	<i>A "heat map" plot of the operator matrix</i>
--------	---

Description

Plots the row and column values of the operator matrix as a "heat map".

Usage

```
plotOp(op, nbreaks = 30, panel = op.panel, aspect = "xy",
       xlab = "Column", ylab = "Row", ...)
```

plotVar

11

Arguments

`op` object of class "op", "opblend", or "stlop".
`nbreaks` number of breaks in the color graduation.
`panel, aspect, xlab, ylab, ...`
 override the default arguments to the lattice call.

Details

The color white correspond to values around 0.

Value

an object of class "trellis".

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6, 3-73.

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

See Also

[stl2](#)

Examples

```
sop <- stlOp(75, n.p=7, s.window="periodic")
plotOp(sop$fit)
```

*plotVar**Plot of the variance factor of the loess or stl fits at each design point***Description**

Trellis plot of the values of the variance factor of the loess or stl fits at each design point.

Usage

```
plotVar(op, ylab = "variance", xlab = "x",
        panel = var.panel, auto.key =
        list(lines = TRUE, points = FALSE, columns = 3), ...)
```

Arguments

`op` object of class "op", "stlop", or "opblend"
`ylab, xlab, panel, auto.key, ...`
 trellis plotting parameters

Details

If `op` is of class "op", then the values of `op$var` are plotted. If `op` is of class "opblend", the variance of the two fits and the blended variance are overplotted. If `op` is of class "stlop", the variance for the fitted values, trend, and seasonal components are overplotted.

Value

an object of class "trellis"

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6, 3-73.

W.S. Cleveland, E. Grosse and W.M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J.M. Chambers and T.J. Hastie, Wadsworth & Brooks/Cole.

See Also

[stl2](#)

Examples

```
n <- 200
L1 <- loessOp(n, span=91, degree=2, at=c(-9:(n+10)))
L2 <- loessOp(n, span=45, degree=0, at=c(-9:(n+10)))
Lb <- opBlend(L1, L2, n.b=50, blend.p=0.5)

p <- plotVar(L1)
p
plotVar(L2, ylim=p$y.limits)
plotVar(Lb)

sop <- stlOp(50, n.p=7, s.window="periodic", s.degree=1, n.ahead=7)
plotVar(sop)
plotVar(sop$S)
```

predict.op

Obtain predicted values for a given operator matrix and data vector

Description

Returns fitted values, optionally with standard errors.

Usage

```
## S3 method for class 'op':
predict(op, y, se = FALSE, newdata = op$at, interval = c("none",
  "confidence", "prediction"), level = 0.95)
## S3 method for class 'stlop':
predict(stlop, y, ...)
```

Arguments

op	object of class "op" or "opblend".
stlop	object of class "stlop".
y	data vector with same length as the number of columns of the operator matrix in op or the fit operator matrix in stlop.
se	should standard errors be computed?
newdata	at which values should fits be computed? Valid values are 1 to n.ahead.
interval	should confidence or prediction intervals be computed?
level	level for the confidence or prediction intervals.

Details

If `se=TRUE` or `interval` is "confidence" or "prediction", and the argument `op` does not have a `stats` element, the auxiliary statistics will be computed. Also, if `interval="prediction"` and `newdata` was not specified, it will by default be all values beyond the length of the original series, up to `n.ahead`.

Value

If `se = FALSE` and `interval="none"`, a vector giving the prediction for each point in the design space. If `se = TRUE` or `interval` is "confidence" or "prediction", a list containing a data frame `data` with components

x	the time values.
y	the observed series values.
at	the time values at which the fit was computed.
fit	the fitted/predicted values.
se.fit	an estimated standard error for each predicted value.
lower	lower confidence/prediction limit, if requested.
upper	upper confidence/prediction limit, if requested and additional elements.
residual.scale	the estimated scale of the residuals used in computing the standard errors.
df	an estimate of the effective degrees of freedom used in estimating the residual scale, intended for use with t-based confidence intervals.

Note

All of this can be done for loess fitting easily using `loess()` and `predict.loess()`. More important is `predict.stlop()` for obtaining predicted values and standard errors for the STL decomposition.

Author(s)

Ryan Hafen

References

R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6, 3-73.

W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

See Also

[stlOp](#), [loessOp](#)

Examples

```
co2op <- stlOp(468, n.p = 12, l.window = 13, t.window = 19,
  s.window = 35, s.degree = 1, n.ahead = 36)

co2pi <- predict(co2op, co2, interval="prediction")

plot(co2pi, start=400, xlab="Time", ylab="CO2Concentration(ppm)")

# a simulated example
rf <- function(x) {
  n <- length(x)
  sin(x * 2 * pi/200) + rnorm(n, sd = 0.5) + rep(c(0.5, 0.25,
    0, -0.25, -0.5, -0.25, 0), ceiling(n/7))[1:n]
}
n <- 200
x <- 1:n
set.seed(8765)
ysim <- rf(x)

sop <- stlOp(200, n.p=7, t.window=105, t.degree=2, s.window="periodic")
ysimCI <- predict(sop, ysim, interval="confidence", level=0.99)
plot(ysimCI, fcol="gray", CIalpha=0.75)
```

respsim

Simulated daily emergency department respiratory chief complaint counts

Description

A time series of daily counts of patients who checked in to an emergency department with a chief complaint categorized as respiratory.

Usage

```
respsim
```

stlOp

15

Format

A vector containing 1399 observations.

Source

Simulated from a model based on real data.

References

R.<a0>P. Hafen, D.<a0>E. Anderson, W.<a0>S. Cleveland, R.<a0>Maciejewski, D.<a0>S. Ebert, A.<a0>Abusalah, M.<a0>Yakout, M.<a0>Ouzzani, and S.<a0>J. Grannis. Syndromic surveillance: STL for modeling, visualizing, and monitoring disease counts. *BMC Medical Informatics and Decision Making*, 9(1):21, 2009.

*stlOp**Calculate operator matrices for the STL decomposition***Description**

Calculate operator matrices for the STL decomposition for a univariate equally-spaced design time series.

Usage

```
## Default S3 method:
stlOp(n, n.p, s.window, s.degree = 1,
      t.window = NULL, t.degree = 1, fc.window = NULL, fc.degree = NULL,
      l.window = nextodd(n.p), l.degree = t.degree, critfreq
      = 0.05, inner = 2, n.ahead = 0, s.blend = 0, t.blend =
      0, l.blend = t.blend, fc.blend = NULL, fc.name = NULL,
      arma = NULL, stats = TRUE)
## S3 method for class 'stl':
stlOp(x, n.ahead=0, s.blend=0, t.blend=0,
      l.blend=t.blend, critfreq=0.05, arma=NULL, stats=TRUE)
## S3 method for class 'stl2':
stlOp(x, n.ahead=0, arma=NULL, stats=TRUE)
```

Arguments

<i>n</i>	number of observations in the time series.
<i>n.p</i>	the periodicity of the seasonal component.
<i>s.window</i>	either the character string "periodic" or the span (in lags) of the loess window for seasonal extraction, which should be odd. This has no default.
<i>s.degree</i>	degree of locally-fitted polynomial in seasonal extraction. Should be 0, 1, or 2.
<i>t.window</i>	the span (in lags) of the loess window for trend extraction, which should be odd. If NULL, the default, <code>nextodd(ceiling((1.5*period) / (1-(1.5/s.window))))</code> , is taken.
<i>t.degree</i>	degree of locally-fitted polynomial in trend extraction. Should be 0, 1, or 2.

<code>l.window</code>	the span (in lags) of the loess window of the low-pass filter used for each sub-series. Defaults to the smallest odd integer greater than or equal to <code>frequency(x)</code> which is recommended since it prevents competition between the trend and seasonal components. If not an odd integer its given value is increased to the next odd one.
<code>l.degree</code>	degree of locally-fitted polynomial for the subseries low-pass filter. Should be 0, 1, or 2.
<code>critfreq</code>	the critical frequency to use for automatic calculation of smoothing windows for the trend and high-pass filter.
<code>fc.window</code>	vector of lengths of windows for loess smoothings for other trend frequency components after the original STL decomposition has been obtained. The smoothing is applied to the data with the STL seasonal component removed. A frequency component is computed by a loess fit with the window length equal to the first element of <code>fc.window</code> , the component is removed, another component is computed with the window length equal to the second element of <code>fc.window</code> , and so forth. In most cases, the values of the argument should be decreasing, that is, the frequency bands of the fitted components should increase. The robustness weights from original STL are used as weights in the loess fitting if specified.
<code>fc.degree</code>	vector of degrees of locally-fitted polynomial in the loess smoothings for the frequency components specified in <code>fc.window</code> . Values of 0, 1 and 2 are allowed. If the length of <code>fc.degree</code> is less than that of <code>fc.window</code> , the former is expanded to the length of the latter using <code>rep</code> ; thus, giving the value 1 specifies a degree of 1 for all components.
<code>fc.name</code>	vector of names of the post-trend smoothing operations specified by <code>fc.window</code> and <code>fc.degree</code> (optional).
<code>inner</code>	integer; the number of 'inner' (backfitting) iterations; usually very few (2) iterations suffice.
<code>s.blend, t.blend, l.blend, fc.blend</code>	vectors of proportion of blending to degree 0 polynomials at the endpoints of the series.
<code>arma</code>	object of class "Arima", which specifies the arma model to use on the remainder component.
<code>stats</code>	whether or not to calculate auxiliary statistics for the overall STL fit operator matrix.

Details

The STL operator matrix is the matrix that yields the fitted values (seasonal + trend) through a linear operation with the observed time series. It is obtained through a series of linear filters. If post-trend smoothing is specified through `fc.window`, ..., then the overall operator matrix will be the seasonal operator plus the operator for each post-trend smoothing. If ARMA modeling is specified, this will also be factored in to the calculation of the overall operator matrix.

Value

a list of class "stlop".

`fit` the overall operator matrix, a list of class "op", which is calculated as described in the details section. If `stats=TRUE`, then auxiliary statistics will also be computed and will be available as an element of this list.

stlOp

17

seas	the operator matrix for just the seasonal component, an object of class "op".
trend	the operator matrix for just the trend component, a list of class "op".
fc	a list of operator matrices corresponding to post-trend frequency components, if specified.
at	the values at which the operator matrices were calculated, which will either be 1, ..., n or 1, ..., n + n.ahead.
pars	parameters used in the fitting.

Note

This function does a lot of $n \times n$ matrix multiplication. Be aware of this when choosing n . While the loess operator matrices are calculated in C, the matrix multiplication happens in R. This is because of the speed of BLAS, which is especially good if R has been compiled with threaded BLAS.

Author(s)

Ryan Hafen

References

- R. B. Cleveland, W. S. Cleveland, J.E. McRae, and I. Terpenning (1990) STL: A Seasonal-Trend Decomposition Procedure Based on Loess. *Journal of Official Statistics*, 6, 3-73.
- W. S. Cleveland, E. Grosse and W. M. Shyu (1992) Local regression models. Chapter 8 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

See Also

[stl](#), [plotVar](#), [loessOp](#), [predict.stlop](#)

Examples

```
sop <- stlOp(50, n.p=7, s.window="periodic")
plotVar(sop)
plotVar(sop$seas)

# stl blending the trend
t.blend <- list(deg=0, span=11, blend.p=0.5, n.b=20)
sop2 <- stlOp(50, n.p=7, s.window="periodic", t.blend=t.blend)
plotVar(sop2)
plotOp(sop$fit)

# predicting ahead
sop <- stlOp(100, n.p=7, s.window=51, s.degree=1, n.ahead=7)
plotVar(sop)

# now stl + further loess smoothing, predicting ahead 7 days
# first get day-of-week component, then smooth with span 1001
# followed by smooth of span 91
n <- 200
rop <- stlOp(n, 7, s.window="periodic", t.window=39, n.ahead=7,
  fc.window=c(1001, 91), fc.degree=c(1, 2), fc.blend=c(0, 0.5))
plotVar(rop)

# do same thing the hard way (without specifying fc.degree, etc.)
```

```
# just to illustrate handling elements of stlop objects
tmp <- stlOp(n, 7, s.window="periodic", t.window=39, n.ahead=7)
dow <- tmp$seas$0
fc1 <- loessOp(n, span=1001, degree=1, at=1:207)$0
fc2 <- loessOp(n, span=91, degree=2, at=1:207, blend=0.5)$0
rop2 <- dow + fc1 - fc1
      fc2
```

Index

*Topic **datasets**

respsim, [14](#)

anova.op, [1](#)

cp, [3](#)

freqr, [4](#), [6](#), [9](#)

freqrlo, [5](#), [5](#), [9](#)

loess, [6](#)

loessOp, [2](#), [6](#), [7](#), [9](#), [13](#), [17](#)

opVar, [8](#)

plot.freqr, [5](#), [6](#), [9](#)

plotOp, [8](#), [10](#)

plotVar, [8](#), [9](#), [11](#), [17](#)

predict.op, [12](#)

predict.stlop, [17](#)

predict.stlop(*predict.op*), [12](#)

respsim, [14](#)

stl, [17](#)

stl2, [10](#), [11](#)

stlOp, [4](#), [5](#), [8](#), [9](#), [13](#), [14](#)

LIST OF REFERENCES

LIST OF REFERENCES

- R. B. Bacastow. Modulation of atmospheric carbon dioxide by the southern oscillation. 1976. (page 30).
- S. Bianconcini. LOESS Asymmetric Filters for Real Time Economic Analysis. In *Second Italian Congress of Econometrics and Empirical Economics*, 2007. (page 51).
- P. Bloomfield. *Fourier analysis of time series: an introduction*. Wiley-Interscience, 2004. (page 14).
- M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448 – 461, 1973. (page 110).
- G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time series analysis: forecasting and control*. Holden-day San Francisco, 1976. (page 14).
- J. C. Brillman, T. Burr, D. Forslund, E. Joyce, R. Picard, and E. Umland. Modeling emergency department visit patterns for infectious disease complaints: results and application to disease surveillance. *BMC Medical Informatics and Decision Making*, 5(4):1–14, 2005. (page 71).
- A. Buja, T. Hastie, and R. Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, 17(2):453–555, 1989. (page 8).
- H. S. Burkom. Development, adaptation, and assessment of alerting algorithms for biosurveillance. *Johns Hopkins APL Technical Digest*, 24(4):335–342, 2003. (pages 71, 80).
- H. S. Burkom, S. P. Murphy, and G. Shmueli. Automated time series forecasting for biosurveillance. *Statistics in Medicine*, 26(22):4202–4218, Mar 2007. (page 71).
- T. Burr, T. Graves, R. Klamann, S. Michalak, R. Picard, and N. Hengartner. Accounting for seasonal patterns in syndromic surveillance data for outbreak detection. *BMC Medical Informatics and Decision Making*, 6(1):40, 2006. (pages 71, 75).
- P. Chaudhuri and J. S. Marron. Sizer for exploration of structures in curves. *Journal of the American Statistical Association*, 94(447):807–823, 1999. (page 110).
- R. B. Cleveland, W. S. Cleveland, J. E. McRae, and I. Terpenning. STL: a seasonal-trend decomposition procedure based on loess (with discussion). *Journal of Official Statistics*, 6:3–73, 1990. (pages 20, 25, 26, 28, 45, 46).
- W. S. Cleveland. *Visualizing Data*. Hobart Press, 1993. ISBN 0963488406. (page 10).

W. S. Cleveland and S. J. Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988. (pages 2, 9).

W. S. Cleveland and E. Grosse. Computational methods for local regression. *Statistics and Computing*, 1(1):47–62, 1991. (pages 2, 110).

W. S. Cleveland and C. Loader. Smoothing by local regression: principles and methods. *Statistical Theory and Computational Aspects of Smoothing*, pages 10–49, 1996. (page 2).

W. S. Cleveland, D. M. Dunn, and I. J. Terpenning. SABL – a resistant seasonal adjustment procedure with graphical methods for interpretation and diagnosis. *Seasonal Analysis of Economic Time Series*, pages 201–231, 1978. (page 20).

W. S. Cleveland, S. J. Devlin, and E. Grosse. Regression by local fitting: methods, properties, and computational algorithms. *Journal of Econometrics*, 37(1):87–114, 1988. (pages 2, 11).

W. S. Cleveland, E. Grosse, and W. M. Shyu. Local regression models. *Statistical Models in S*, pages 309–376, 1992. (page 2).

W. S. Cleveland, C. L. Mallows, and J. E. McRae. ATS Methods: Nonparametric Regression for Nongaussian Data. *Journal of the American Statistical Association*, 88(423):821–835, 1993. (page 110).

J. D. Cryer and K. S. Chan. *Time series analysis with applications in R*. Springer, 2008. (page 14).

E. B. Dagum. Modelling, Forecasting and Seasonally Adjusting Economic Time Series with the X-11 ARIMA Method. *The Statistician*, pages 203–216, 1978. (page 20).

C. Daniel, F. S. Wood, and J. W. Gorman. *Fitting equations to data*. Wiley New York, 1980. (page 92).

L. Devroye and A. Berlinet. A Comparison of Kernel Density Estimates. *Publications de l'Institut de Statistique de l'Université de Paris*, 75, 1994. (pages 109, 239).

J. Fan. Local linear regression smoothers and their minimax efficiencies. *The Annals of Statistics*, 21(1):196–216, 1993. (page 93).

R. W. Farebrother. The Distribution of a quadratic form in normal variables. *Applied statistics*, 39(2):294–309, 1990. (page 6).

M. Farnen and J. S. Marron. An assessment of finite sample performance of adaptive methods in density estimation. *Computational Statistics & Data Analysis*, 30(2): 143–168, 1999. (pages 92, 110).

C. P. Farrington, N. J. Andrews, A. D. Beale, and M. A. Catchpole. A statistical algorithm for the early detection of outbreaks of infectious disease. *Journal of the Royal Statistical Society Series A: Statistics in Society*, 159(3):547–563, 1996. (pages 71, 72).

- D. F. Findley and B. C. Monsell. Invited commentary on STL: A seasonal-trend decomposition procedure based on loess by R. B. Cleveland et al.. *Journal of Official Statistics*, 6(1):47–54, 1990. (pages 45, 50).
- D. R. Franz, P. B. Jahrling, A. M. Friedlander, D. J. McClain, D. L. Hoover, W. R. Bryne, J. A. Pavlin, G. W. Christopher, and E. M. Eitzen. Clinical recognition and management of patients exposed to biological warfare agents. *Journal of the American Medical Association*, 278(5):399–411, 1997. (page 73).
- S. Grannis, M. Wade, J. Gibson, and J. M. Overhage. The Indiana public health emergency surveillance system: Ongoing progress, early findings, and future directions. In *American Medical Informatics Association Annual Symposium Proceedings*, volume 2006, page 304. American Medical Informatics Association, 2006. (page 72).
- A. G Gray and A. W. Moore. Rapid evaluation of multiple density models. *Artificial Intelligence and Statistics*, 2003, 2003a. (page 91).
- A. G. Gray and A.W. Moore. Nonparametric density estimation: Toward computational tractability. *SIAM International Conference on Data Mining*, 2003, 2003b. (page 91).
- A. G. Gray and P. J. Thomson. Invited commentary on STL: A seasonal-trend decomposition procedure based on loess by R. B. Cleveland et al.. *Journal of Official Statistics*, 6(1):47–54, 1990. (pages 50, 52).
- R. P. Hafen, D. E. Anderson, W. S. Cleveland, R. Maciejewski, D. S. Ebert, A. Abusalah, M. Yakout, M. Ouzzani, and S. J. Grannis. Syndromic surveillance: STL for modeling, visualizing, and monitoring disease counts. *BMC Medical Informatics and Decision Making*, 9(1):21, 2009. (page 89).
- T. J. Hastie and R. J. Tibshirani. Local likelihood estimation. *Journal of the American Statistical Association*, 82:559–567, 1987. (page 100).
- L. Hutwagner, W. Thompson, G. M. Seeman, and T. Treadwell. The bioterrorism preparedness and response Early Aberration Reporting System (EARS). *Journal of Urban Health: Bulletin of the New York Academy of Medicine*, 80(1):i89–i96, 2003. (pages 71, 72, 80, 88).
- L. Hutwagner, T. Browne, G. M. Seeman, and A. T. Fleischauer. Comparing aberration detection methods with simulated data. *Emerging Infectious Diseases*, 11(2):314–316, 2005. (pages 72, 81).
- L. C. Hutwagner, E. K. Maloney, N. H. Bean, L. Slutsker, and S. M. Martin. Using laboratory-based surveillance data for prevention: an algorithm for detecting salmonella outbreaks. *Emerging Infectious Diseases*, 3:395–400, 1997. (pages 71, 72).
- A. J. Izenman. Recent Developments in Nonparametric Density Estimation. *Journal of the American Statistical Association*, 86(413):205–224, 1991. (page 91).
- M. L. Jackson, A. Baer, I. Painter, and J. Duchin. A simulation study comparing aberration detection algorithms for syndromic surveillance. *BMC Medical Informatics and Decision Making*, 7:6–6, 2007. (pages 79, 81, 88).

N. L. Johnson, S. Kotz, and A. W. Kemp. *Univariate Discrete Distributions*. Wiley, New York, 1992. (page 77).

M. C. Jones, J. S. Marron, and S. J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association*, 91(433), 1996. (page 91).

R. J. Karunamuni and T. Alberts. On boundary correction in kernel density estimation. *Statistical Methodology*, 2(3):191–212, 2005. (page 104).

C. D. Keeling, S. C. Piper, R. B. Bacastow, M. Wahlen, T. P. Whorf, M. Heimann, and H. A. Meijer. Exchanges of atmospheric CO₂ and ¹³CO₂ with the terrestrial biosphere and oceans from 1978 to 2000. I. *Global Aspects. UC San Diego: Scripps Institution of Oceanography*, 2001. (page 16).

K. P. Kleinman, A. M. Abrams, M. Kulldorff, and R. Platt. A model-adjusted space–time scan statistic with an application to syndromic surveillance. *Epidemiology and Infection*, 133(03):409–419, 2005. (page 71).

S. Kotz and S. Nadarajah. *Extreme Value Distributions: Theory and Applications*. World Scientific, 2000. (page 99).

C. Loader. *Local regression and likelihood*. Springer Verlag, 1999. (pages 92, 104).

C. Loader. Smoothing: Local Regression Techniques. *Handbook of Computational Statistics*, 2004. (page 92).

C. R. Loader. Local likelihood density estimation. *The Annals of Statistics*, 24(4):1602–1618, 1996. (page 93).

D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *Annals of Mathematical Statistics*, 36(1049-1051):2, 1965. (pages 92, 95).

R. Maciejewski, S. Rudolph, R. P. Hafen, A. Abusalah, M. Yakout, M. Ouzzani, W. S. Cleveland, S. J. Grannis, M. Wade, and D. S. Ebert. Understanding syndromic hotspots – a visual analytics approach. In *IEEE Symposium on Visual Analytics Science and Technology, 2008. VAST’08*, pages 35–42, 2008. (page 89).

C. L. Mallows. Some comments on C_p . *Technometrics*, 15:661–675, 1973. (page 11).

K. D. Mandl, B. Reis, and C. Cassa. Measuring outbreak-detection performance by using controlled feature set simulations. *MMWR Morbidity Mortality Weekly Report*, 53:130–6, 2004. (page 79).

J. S. Marron and H. P. Schmitz. Simultaneous Density Estimation of Several Income Distributions. *Econometric Theory*, 8:476–476, 1992. (page 94).

J. S. Marron and F. Udina. Interactive local bandwidth choice. *Statistics and Computing*, 9(2):101–110, 1999. (pages 92, 110).

M. Meselson, J. Guillemin, M. Hugh-Jones, A. Langmuir, I. Popova, A. Shelokov, and O. Yampolskaya. The Sverdlovsk anthrax outbreak of 1979. *Science*, 266(5188):1202–1208, 1994. (page 79).

- M. C. Minnotte and David W. Scott. The mode tree: A tool for visualization of nonparametric density features. *Journal of Computational and Graphical Statistics*, 2(1):51–68, 1993. (page 111).
- R. T. Olszewski. Bayesian classification of triage diagnoses for the early detection of epidemics. *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference. Menlo Park, CA: AAAI*, pages 412–7, 2003. (page 72).
- J. Opsomer, Y. Wang, and Y. Yang. Nonparametric regression with correlated errors. *Statistical Science*, pages 134–153, 2001. (page 17).
- E. Parzen. On estimation of a probability density and mode. *Annals of Mathematical Statistics*, 33(1065-1076):478, 1962. (pages 91, 103).
- B. Y. Reis and K. D. Mandl. Time series modeling for syndromic surveillance. *BMC Medical Informatics and Decision Making*, 3:2, Jan 2003. (pages 71, 72, 81).
- M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist*, 27(3):832–837, 1956. (pages 91, 103).
- P. E. Sartwell. The distribution of incubation periods of infectious disease. *American Journal of Epidemiology*, 51(3):310–318, 1950. (pages 79, 192).
- F. E. Satterthwaite. An Approximate distribution of estimates of variance components. *Biometrics Bulletin*, pages 110–114, 1946. (page 6).
- D. W. Scott. *Multivariate density estimation: theory, practice, and visualization*. Wiley-Interscience, 1992. (page 91).
- D W Scott and G R Terrell. Biased and unbiased cross-validation in density estimation. *Journal of the American Statistical Association*, 82(400):1131–1146, 1987. (page 103).
- S. J. Sheather. Density estimation. *Statistical Science*, 19(4):588–597, 2004. (pages 91, 92, 104).
- S. J. Sheather and M. C. Jones. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B. Methodological*, 53(3):683–690, 1991. (page 103).
- R. H. Shumway and D. S. Stoffer. *Time series analysis and its applications*. Springer Verlag, 2000. (pages 14, 15).
- B. W. Silverman. Algorithm AS 176: Kernel Density Estimation Using the Fast Fourier Transform. *Applied Statistics*, 31(1):93–99, 1982. (page 91).
- B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986. (pages 91, 103, 104).
- L. Simonsen. The impact of influenza epidemics on mortality: introducing a severity index. *American Journal of Public Health*, 87(12):1944–1950, 1997. (pages 71, 72).
- L. Stern and D. Lightfoot. Automated outbreak detection: a quantitative retrospective analysis. *Epidemiology and Infection*, 122(01):103–110, 1999. (pages 71, 72).

- D. F. Stroup, M. Wharton, K. Kafadar, and A. G. Dean. Evaluation of a method for detecting aberrations in public health surveillance data. *American Journal of Epidemiology*, 137(3):373–380, 1993. (pages 71, 72).
- P. A. Tukey and J. W. Tukey. Agglomeration and sharpening. In *The collected works of John W. Tukey*, volume V., pages 233–243. Wiley New York, 1981. (pages 92, 95).
- B. N. Tyner. *Experimental methods for model selection*. PhD thesis, Purdue University, 2007. (page 12).
- S. Wallenstein and J. Naus. Scan statistics for temporal surveillance for biologic terrorism. *MMWR Morbidity and Mortality Weekly Report*, 53 Suppl:75–8, 2004. (page 71).
- B. Wallgren and A. Wallgren. Invited commentary on STL: A seasonal-trend decomposition procedure based on loess by R. B. Cleveland et al.. *Journal of Official Statistics*, 6(1):47–54, 1990. (page 50).
- M. P. Wand and M. C. Jones. *Kernel smoothing*. Chapman & Hall/CRC, 1995. (page 104).
- W. K. Wong, A. Moore, G. Cooper, and M. Wagner. Rule-based anomaly pattern detection for detecting disease outbreaks. In *Proceedings of the 18th National Conference on Artificial Intelligence*, pages 217–223. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002. (page 71).
- B Xi, H Chen, W S Cleveland, and T Telkamp. Statistical analysis and modeling of internet voip traffic for network engineering. September 2009. (pages 94, 108).

VITA

VITA

Ryan Hafen was born in Idaho Falls, Idaho in 1980. He graduated from Snow Canyon High School in 1998. He earned an Associate of Arts degree at Dixie College in 2002, a BS in Statistics at Utah State University in 2004, and a Master of Statistics in Mathematics from the University of Utah in 2006. His academic interests include statistical model building, exploratory data analysis, visualization, massive data, computational statistics, density estimation, time series, and nonparametric statistics.