

PERFORMANCE EVALUATION OF COMPUTER AND COMMUNICATION SYSTEMS

Jean-Yves Le Boudec
EPFL



Version 2.3 of April 4, 2022
Essentially identical to publisher's version, except for formatting
With bug fixes

Available at <https://leboudec.github.io/perfeval/>

ACKNOWLEDGEMENTS

I would like to thank Anthony Davison for allowing me to access a beta version of his book “Statistical Models”, as well as Richard Weber, who made his lecture notes freely available on the web and allowed me to use them as constituent material in an early version of this course. I am grateful to François Bacelli and Pierre Brémaud who helped me obtain some understanding of their fields. Many thanks go to Mourad Kara for discussions and input, to Irina Baltcheva, Manuel Flury, Olivier Gallay, Assane Gueye, Paul Hurley, Ruben Merz, Božidar Radunović, Gianluca Rizzo, Slaviša Sarafijanović, Milan Vojnović, Utkarsh Upadhyay and Jonas Wagner for various inputs and comments. I thank Scouac, Pilou and their friends for visiting our pages here and there. Last but not least, I thank Elias for the artwork.

Contents

Preface	xvii
1 Methodology	1
1.1 What is Performance Evaluation ?	2
1.1.1 Load	2
1.1.2 Metric	2
1.1.3 The Different Goals of Performance Evaluation	4
1.2 Factors	5
1.2.1 The Hidden Factor Paradox	6
1.2.2 Simpson's Paradox	7
1.3 Evaluation Methods	8
1.4 The Scientific Method	8
1.5 Performance Patterns	10
1.5.1 Bottlenecks	10
1.5.2 Congestion Collapse	12
1.5.3 Competition Side Effect	13
1.5.4 Latent Congestion Collapse	16
1.6 Review	17
1.6.1 Check-List	17
1.6.2 Review Questions	18
2 Summarizing Performance Data, Confidence Intervals	21
2.1 Summarized Performance Data	22
2.1.1 Histogram and Empirical CDF	22
2.1.2 Mean, Median and Quantiles	23
2.1.3 Coefficient of Variation and Lorenz Curve Gap	25
2.1.4 Fairness Indices	26
2.2 Confidence Intervals	31
2.2.1 What is a Confidence Interval ?	31

2.2.2	Confidence Interval for Median and Other Quantiles	32
2.2.3	Confidence Interval for the Mean	34
2.2.4	Confidence Intervals for Fairness Indices and The Bootstrap	36
2.2.5	Confidence Interval for Success Probability	38
2.3	The Independence Assumption	39
2.3.1	What does iid mean ?	39
2.3.2	How do I know in Practice if the iid Assumption is Valid ?	40
2.3.3	What Happens If The IID Assumption Does Not Hold ?	41
2.4	Prediction Interval	44
2.4.1	Prediction for an IID Sample based on Order Statistic	45
2.4.2	Prediction for a Normal IID Sample	46
2.4.3	The Normal Assumption	47
2.5	Which Summarization To Use ?	49
2.5.1	Robustness	49
2.5.2	Compactness	52
2.6	Other Aspects of Confidence/Prediction Intervals	53
2.6.1	Intersection of Confidence/Prediction Intervals	53
2.6.2	The Meaning of Confidence	53
2.7	Proofs	54
2.8	Review	55
2.8.1	Summary	55
2.8.2	Review Questions	56
3	Model Fitting	59
3.1	Model Fitting Criteria	60
3.1.1	What is Model Fitting ?	60
3.1.2	Least Squares Correspond to Gaussian, Same Variance	64
3.1.3	ℓ^1 Norm Minimization Corresponds to Laplace Noise	65
3.2	Linear Regression	66
3.3	Linear Regression with ℓ^1 Norm Minimization	70
3.4	Choosing a Distribution	73
3.4.1	Shape	73
3.4.2	Skewness and Kurtosis	74
3.4.3	Power Laws, Pareto Distribution and Zipf's Law	76
3.4.4	Hazard Rate	77
3.4.5	Fitting A Distribution	78
3.4.6	Censored Data	80

3.4.7	Combinations of Distributions	82
3.5	Heavy Tail	84
3.5.1	Definition	84
3.5.2	Heavy Tail and Stable Distributions	85
3.5.3	Heavy Tail in Practice	85
3.5.4	Testing For Heavy Tail	88
3.5.5	Application Example: The Workload Generator SURGE	89
3.6	Proofs	91
3.7	Review	92
3.7.1	Review Questions	92
3.7.2	Useful Matlab Commands	92
4	Tests	97
4.1	The Neyman Pearson Framework	98
4.1.1	The Null Hypothesis and The Alternative	98
4.1.2	Critical Region, Size and Power	99
4.1.3	p -value of a Test.	102
4.1.4	Tests Are Just Tests	103
4.2	Likelihood Ratio Tests	104
4.2.1	Definition of Likelihood Ratio Test	104
4.2.2	Student Test for Single Sample (or Paired Data)	105
4.2.3	The Simple Goodness of Fit Test	106
4.3	ANOVA	108
4.3.1	Analysis of Variance (ANOVA) and F -tests	108
4.3.2	Testing for a Common Variance	112
4.4	Asymptotic Results	114
4.4.1	Likelihood Ratio Statistic	114
4.4.2	Pearson Chi-squared Statistic and Goodness of Fit	114
4.4.3	Test of Independence	117
4.5	Other Tests	118
4.5.1	Goodness of Fit Tests based on Ad-Hoc Pivots	118
4.5.2	Robust Tests	121
4.6	Proofs	123
4.7	Review	125
4.7.1	Tests Are Just Tests	125
4.7.2	Review Questions	125

5 Forecasting	127
5.1 What is Forecasting ?	128
5.2 Linear Regression	129
5.3 The Overfitting Problem	131
5.3.1 Use of Test Data	133
5.3.2 Information Criterion	133
5.4 Differencing the Data	136
5.4.1 Differencing and De-seasonalizing Filters	136
5.4.2 Computing Point Prediction	137
5.4.3 Computing Prediction Intervals	139
5.5 Fitting Differenced Data to an ARMA Model	140
5.5.1 Stationary but non IID Differenced Data	140
5.5.2 ARMA and ARIMA Processes	141
5.5.3 Fitting an ARMA Model	143
5.5.4 Forecasting	147
5.6 Sparse ARMA and ARIMA Models	150
5.6.1 Constrained ARMA Models	151
5.6.2 Holt-Winters Models	152
5.7 Proofs	156
5.8 Review Questions	158
6 Discrete Event Simulation	161
6.1 What is a Simulation ?	162
6.1.1 Simulated Time and Real Time	163
6.1.2 Simulation Types	163
6.2 Simulation Techniques	166
6.2.1 Discrete Event Simulation	166
6.2.2 Stochastic Recurrence	169
6.3 Computing the Accuracy of Stochastic Simulations	171
6.3.1 Independent Replications	171
6.3.2 Computing Confidence Intervals	172
6.3.3 Non-Terminating Simulations	172
6.4 Monte Carlo Simulation	172
6.5 Random Number Generators	175
6.6 How to Sample from a Distribution	176
6.6.1 CDF Inversion	179
6.6.2 Rejection Sampling	181

6.6.3	Ad-Hoc Methods	183
6.7	Importance Sampling	184
6.7.1	Motivation	184
6.7.2	The Importance Sampling Framework	185
6.7.3	Selecting An Importance Sampling Distribution	189
6.8	Proofs	191
6.9	Review	193
7	Palm Calculus, or the Importance of the Viewpoint	195
7.1	An Informal Introduction	197
7.1.1	Event versus Time Averages	197
7.1.2	The Large Time Heuristic	198
7.1.3	Two Event Clocks	200
7.1.4	Arbitrary Sampling Methods	201
7.2	Palm Calculus	204
7.2.1	Hypotheses	204
7.2.2	Definitions	204
7.2.3	Interpretation as Time and Event Averages	206
7.2.4	The Inversion and Intensity Formulas	207
7.3	Other Useful Palm Calculus Results	209
7.3.1	Residual Time and Feller's Paradox	209
7.3.2	The Rate Conservation Law and Little's Formula	212
7.3.3	Two Event Clocks	218
7.4	Simulation Defined as Stochastic Recurrence	219
7.4.1	Stochastic Recurrence, Modulated Process	219
7.4.2	Freezing Simulations	220
7.4.3	Perfect Simulation of Stochastic Recurrence	222
7.5	Application to Markov Chain Models and the PASTA Property	226
7.5.1	Embedded Sub-Chain	226
7.5.2	PASTA	228
7.6	Appendix: Quick Review of Markov Chains	230
7.6.1	Markov Chain in Discrete Time	230
7.6.2	Markov Chain in Continuous Time	231
7.6.3	Poisson and Bernoulli	232
7.7	Proofs	233
7.8	Review Questions	236

8 Queuing Theory for Those Who Cannot Wait	237
8.1 Deterministic Analysis	239
8.1.1 Description of a Queuing System with Cumulative Functions	239
8.1.2 Reich's Formula	241
8.2 Operational Laws For Queuing Systems	242
8.2.1 Departures and Arrivals See Same Averages (DASSA)	243
8.2.2 Little's Law and Applications	244
8.2.3 Networks and Forced Flows	244
8.2.4 Bottleneck Analysis	246
8.3 Classical Results for a Single Queue	247
8.3.1 Kendall's Notation	247
8.3.2 The Single Server Queue	248
8.3.3 The Processor Sharing Queue, M/GI/1/PS	253
8.3.4 Single Queue with B Servers	254
8.4 Definitions for Queuing Networks	256
8.4.1 Classes, Chains and Markov Routing	256
8.4.2 Catalog of Service Stations	257
8.4.3 The Station Function	264
8.5 The Product-Form Theorem	269
8.5.1 Product Form	269
8.5.2 Stability Conditions	270
8.6 Computational Aspects	273
8.6.1 Convolution	273
8.6.2 Throughput	274
8.6.3 Equivalent Service Rate	276
8.6.4 Suppression of Open Chains	280
8.6.5 Arrival Theorem and MVA Version 1	281
8.6.6 Network Decomposition	284
8.6.7 MVA Version 2	288
8.7 What This Tells Us	290
8.7.1 Insensitivity	290
8.7.2 The Importance of Modelling Closed Populations	292
8.8 Mathematical Details About Product-Form Queuing Networks	293
8.8.1 Phase Type Distributions	293
8.8.2 Micro and Macro States	295
8.8.3 Micro to Macro: Aggregation Condition	295

8.8.4	Local Balance In Isolation	296
8.8.5	The Product Form Theorem	296
8.8.6	Networks with Blocking	297
8.9	Case Study	298
8.9.1	Deterministic Analysis	299
8.9.2	Single Queue Analysis	300
8.9.3	Operational Analysis	300
8.9.4	Queuing Network Analysis	302
8.9.5	Conclusions	303
8.10	Proofs	305
8.11	Review	307
8.11.1	Review Questions	307
8.11.2	Summary of Notation	308
A	Tables	311
B	Parametric Estimation, Large Sample Theory	317
B.1	Parametric Estimation Theory	317
B.1.1	The Parametric Estimation Framework.	317
B.1.2	Maximum Likelihood Estimator (MLE)	318
B.1.3	Efficiency and Fisher Information	319
B.2	Asymptotic Confidence Intervals	320
B.3	Confidence Interval in Presence of Nuisance Parameters	323
C	Gaussian Random Vectors in \mathbb{R}^n	327
C.1	Notation and a Few Results of Linear Algebra	328
C.1.1	Notation	328
C.1.2	Linear Algebra	328
C.2	Covariance Matrix of a Random Vector in \mathbb{R}^n	329
C.2.1	Definitions	329
C.2.2	Properties of Covariance Matrix	330
C.2.3	Choleski's Factorization	330
C.2.4	Degrees of Freedom	330
C.3	Gaussian Random Vector	331
C.3.1	Definition and Main Properties	331
C.3.2	Diagonal Form	332
C.4	Foundations of ANOVA	333

C.4.1	Homoscedastic Gaussian Vector	333
C.4.2	Maximum Likelihood Estimation for Homoscedastic Gaussian Vectors . .	333
C.5	Conditional Gaussian Distribution	334
C.5.1	Schur Complement	334
C.5.2	Distribution of \vec{X}_1 given \vec{X}_2	334
C.5.3	Partial Correlation	335
C.6	Proofs	336
D	Digital Filters	339
D.1	Calculus of Digital Filters	340
D.1.1	Backshift Operator	340
D.1.2	Filters	340
D.1.3	Impulse response and Dirac Sequence	341
D.1.4	Composition of Filters, Commutativity	342
D.1.5	Inverse of Filter	342
D.1.6	AR(∞) Representation of Invertible Filter	343
D.1.7	Calculus of Filters	343
D.1.8	z Transform	344
D.2	Stability	345
D.3	Filters with Rational Transfer Function	345
D.3.1	Definition	345
D.3.2	Poles and Zeroes	346
D.4	Predictions	349
D.4.1	Conditional Distribution Lemma	349
D.4.2	Predictions	349
D.5	Log Likelihood of Innovation	351
D.6	Matlab Commands	351
D.7	Proofs	352

Index

- $B_{n,p}$, 33
 $H = \lambda G$ Formula, 216
 $M_{n,\bar{q}}$, 106
 R_s , 136
 Z transform, 265
 Δ_s , 136
 χ_n^2 , 35
 ℓ^1 norm minimization, 65
 $\vec{1}_c = (0, \dots, 0)$, 262
 t_n , 35
 z -transform, 249
 γ_1 , 75
 γ_2 , 75
 m_p : p -quantile, 33
 $\hat{\mu}_n$, 35
 $\hat{\sigma}_n$, 35
 $S_{x,x}$, 318
 $\lceil x \rceil$, 24
 $\lfloor x \rfloor$, 24
- ACF, 143
adaptive sampling, 197
admission control, 13
aging, 222
AIC, 133
Akaike's Information Criterion, 133
alternative, 99
Analysis of variance, 108
Anderson-Darling, 119
ANOVA, 108
 $AR(\infty)$, 343
ARMA, Auto-Regressive Moving Average, 141
Arrival Theorem, 281
asymptotically stationary, 164
Auto-Correlation Function, 143
Auto-regressive, 141
auto-regressive, 343
auto-regressive coefficients, 141
Auto-Regressive Moving Average, 141
- backshift, 340
balance function, 259
balanced fairness, 292
Bayesian Information Criterion, 134
BCMP networks, 256
benchmark, 2
Bernoulli, 33
Bernoulli process, 232
BIC, Bayesian Information Criterion, 134
Binomial distribution, 33
bins, 106, 114
bootstrap method, 36
bootstrap replicates, 37
bottleneck, 246
bottlenecks, 10
Box Plot, 23
Box-Cox transformation, 48
Box-Jenkins, 140
Box-Müller method, 184
- Campbell's Formula, 215
Campbell's Shot Noise Formula, 214
CDF, 23
CDF inversion, 179
censoring, 80
chain equivalent, 257
chain population vector, 273, 278
chains, 257
characteristic function, 331
Chi-Square, 35
Choleski's Factorization, 330
class, 256
class dependent service rate, 261
closed, 257
closed network, 257
Coefficient of Variation, 25
combination of mixture and truncation, 82
communication classes, 230
comparison, 4
Competition side effect, 13

- Complement Network Theorem, 289
 complementary CDFs, 76
 complete order, 3
 composite goodness of fit, 114
 compound distribution, 82
 Confidence intervals, 31
 confidence level, 32
 Congestion, 12
 congestion collapse, 12
 Consistent family of estimators, 318
 convolution algorithm, 273
 convolution equation, 273
 Convolution Theorem, 273
 Corrected Holt-Winters Additive Seasonal Model, 156
 CoV, 25
 covariance matrix, 329
 critical region, 99
 cross covariance matrix, 329
 cumulant generating function, 75
 cumulant of order k , 75
 DASSA, 243
 de-seasonalizing, 136
 degrees of freedom, 330
 Delay, 261
 delay jitter, 241
 designed experiment, 40
 differencing filter, 136
 Dirac sequence, 341
 discipline, 258
 discrete event simulation, 166
 distribution shape, 73
 Distributional Little Formula, 218
 Double Exponential Smoothing with Regression, 152
 Double Exponential Weighted Moving Average, 153
 doubling time, 61
 efficiency, 319
 egalitarian processor sharing, 253
 embedded sub-chain, 226, 227
 Embedded Subchain, 227
 empirical cumulative distribution function, 23
 engineering rules, 5
 Engset formula, 293
 equivalent service rate, 276
 equivalent station, 276, 284
 ergodic, 230
 Erlang Loss Formula, 255
 Erlang- k , 211
 Erlang- n , 294
 Erlang-B Formula, 255
 Erlang-C, 255
 estimator, 317
 event average, 197
 event clock, 197
 event scheduler, 167
 EWMA, exponentially weighted moving average, 152
 excess kurtosis, 75
 expected information, 319
 explanatory model, 61
 explanatory variables, 67
 exponential distribution, 179
 exponential twisting, 188
 exponentially weighted moving average, 152
 Extended Little Formula, 216
 factors, 5
 Fat Tail, 78
 Feller's paradox, 212
 FIFO, 240
 filter, 340
 Finite Impulse Response (FIR), 341
 Fisher information, 319
 Forecasting, 128
 fractional brownian traffic, 242
 gap, 26
 gaussian distribution, 24
 gaussian vector, 331
 generating function, 265
 generator matrix, 231
 Geometric, 48
 geometric distribution, 180
 GI/GI/1, 167
 Gilbert loss model, 220
 Gini coefficient, 28
 Global LCFSPR, 260
 global macro state space, 295
 global micro state space, 295
 Global PS (Processor Sharing), 260
 Harmonic, 48

- hazard rate, 77
heavy tailed, 84
Hidden Factor Paradox, 6
histogram, 22
Holt-Winters Additive Seasonal Model, 156
Homoscedastic, 333
homoscedasticity, 64, 108
Hyper-Exponential, 294
- iid, 32
impatience, 13
Importance sampling, 185
importance sampling distribution, 186
impulse response, 340
Infinite Impulse Response, 341
Infinite Server, 261
information criterion, 133
inner product, 328
innovation, 141
Innovation Formula, 146
input function, 239
Insensitive Station, 259
insensitivity, 262
Insertion Probability, 259
intensity, 205
Intensity formula, 207
intensity of the workload, 2
Intervention Analysis, 68
invariant by re-parametrization, 318
invariant by reversible data transformation, 319
Inversion formula, 207
irreducible, 230
IS, 261
- Jackson, 256
Jain's Fairness Index, 27
Jarque-Bera, 120
JFI, 27
jointly stationary, 204
- Kelly networks, 256
Kelly station, 262
Kelly-Whittle, 259
Kendall's notation, 247
Kiviat Diagram, 4
Kolmogorov-Smirnov, 119
Kruskal-Wallis, 123
kurtosis index, 75
- Lag-Plot, 41
Laplace distribution, 65
Laplace qq-plot, 72
Laplace Stieltjes transform, 243
large time heuristic, 198
Last Come First Serve, Preemptive Resume, 260
latent congestion collapse, 16
LCFSPR, 260
likelihood, 317
likelihood ratio, 104
likelihood ratio statistic, 320
Lindley's equation, 250
linear congruences, 175
Linear constant-coefficient difference, 345
linear regression, 66
Little's Formula, 216, 217
Ljung-Box, 141
load, 2
Local Balance In Isolation, 296
log-likelihood, 319
log-normal distribution, 73
Lorenz Curve, 27
Lorenz Curve Gap, 26
- macro-state, 295
MAD, 25
Markov routing, 256
marks, 214
matrix of selected transitions, 227
Maximum Likelihood, 318
mean, 24
Mean Absolute Deviation, 25
Mean Difference, 28
Mean Value Analysis, 281
mean-variance, 24
metric, 2
micro-state, 295
mixed network, 257
mixture distribution, 82
mixture of exponentials, 293
mixture of gamma, 293
MLE, 318
model fitting, 61
modulated process, 220
moment heuristics, 145
Monte Carlo simulation, 172
Moving Average, 141

- moving average coefficients, 141
- MSCCC Station, 262
- multi-class product form queuing networks, 256
- multinomial, 106
- Multiple Server with Concurrent Classes of Customers, 262
- multiplicative ARIMA, 151
- MVA, 281
- MVA algorithm version 1, 282
- nested model, 99
- non parametric, 121
- non-dominated, 3
- norm, 328
- normal distribution, 24
- normal qqplots, 47
- nuisance factors, 6
- null hypothesis, 99
- observed information, 319
- open, 257
- open network, 257
- order statistic, 33, 47
- orthogonal matrix, 328
- orthogonal projection, 328
- outlier, 51
- output function, 239
- overfitting problem, 132
- p*-value, 102
- PACF, 143
- paired experiment, 32
- Pareto, 76
- Partial Auto-Correlation Function, 143
- partial correlation, 335, 336
- partial covariance, 335
- partial order, 3
- PASTA, 228
- PDF, 33
- Pearson chi-squared statistic, 116
- per chain throughput, 274
- Per-Class LCFSPR, 260
- Per-Class Processor Sharing, 260
- percentile bootstrap estimate, 37
- percentile inversion method, 179
- perfect simulation, 222
- performance pattern, 2
- periodic, 231
- personal equation, 57
- phase type, 293
- pivot, 119
- point prediction, 128
- Poisson process, 232
- Poles, 346
- Pollaczek-Khinchine formula for means, 214
- Pollaczek-Khinchine formula for transforms, 249
- positive semi-definite, 330
- power, 100
- predicted response, 64
- predicted value, 128
- prediction interval, 24, 44, 129
- probability plot, 47
- Processor Sharing, 253
- profile log likelihood, 323
- proportional fairness, 14
- PS, 253
- pseudo-inverse, 179
- pseudo-random number generator, 175
- qq-plot, 47
- Quadratic, 48
- queue size dependent service rate, 261
- random waypoint, 169
- rare event, 184
- Rate Conservation Law, 212
- rate transition matrix, 231
- recurrent, 230
- Reich, 241
- rejection region, 99
- rejection sampling, 181
- replication, 171
- residual time, 209
- response variables, 67
- reversible, 298
- Roberts' Seasonal Model, 155
- round robin, 253
- routing matrix, 257
- sample q - quantile, 24
- sample ACF, 141
- sample autocovariance, 140
- sample median, 24
- sample standard deviation, 35
- Schur complement, 334
- Seasonal ARIMA, 151

- seed, 175
 selected events, 197
 service rate, 259
 service requirement, 258
 service time, 258
 short circuit, 285
 Shot noise, 214
 Signal to Noise ratio, 25
 simple goodness of fit test, 106
 simplified network, 284
 Simpson's paradox, 7
 Simpson's reversal, 7
 size, 100
 skewness, 75
 skewness index, 75
 Spider Plot, 4
 stable, 345
 stable distribution, 85
 standard deviation, 109
 standard deviation s of a data set, 24
 station buffer, 258
 station function, 265
 station function at micro level, 296
 stationary, 204
 stationary marked point process, 214
 stationary point process, 204
 stationary probability, 230
 stations, 256
 statistic, 36
 statistical model, 62
 stochastic majorization, 23
 stochastic recurrence, 169
 Student, 35
 subnetwork in short-circuit, 285
 Suppression of Open Chains, 280
 SURGE, 89
 Symmetric Station, 262
 system dimensioning, 4
 test of independence, 117
 think time, 244
 Throughput Theorem, 274
 time average, 197
 $T^-(t)$, 209
 token pools, 263
 $T^+(t)$, 209
 Transfer Function, 344
 transformed distribution mean, 48
 transformed sample mean, 48
 transient, 230
 transient removal, 172
 transition matrix, 230
 truncation property, 298
 turning point, 123
 type 1, 99
 type 2, 99
 UE, 89
 UMP, Uniformly More Powerful, 100
 Unbiased estimator, 318
 User Equivalents, 89
 utilization, 3, 17
 variance bias, 68
 visit rates, 257
 Voronoi, 219
 Wald's identity, 218
 Weibull distribution, 78
 weighting function, 186
 white gaussian noise, 331
 white noise variance, 141
 Whittle Function, 259
 Whittle Network, 262
 Wilcoxon Rank Sum, 122
 Wilcoxon Rank Sum Statistic, 122
 Wilcoxon Signed Rank, 121
 workload, 2
 Z-transform, 265
 z-transform, 249
 z-transform (signal processing), 344
 Zeroes, 346
 Zeta, 76
 Zipf's law, 77

PREFACE

PERFORMANCE EVALUATION is often the critical part of evaluating the results of a research project. Many of us are familiar with simulations, but it is often difficult to address questions like: Should I eliminate the beginning of the simulation in order for the system to become stabilized ? I simulate a random way point model but the average speed in my simulation is not as expected. What happened ? The reviewers of my study complained that I did not provide confidence intervals. How do I go about this ? I would like to characterize the fairness of my protocol. Should I use Jain's Fairness Index or the Lorenz Curve Gap ? I would like to fit a distribution to the flow sizes that I measured but all my measurements are truncated to a maximum value; how do I account for the truncation ?



This book groups a set of lecture notes for a course given at EPFL. It contains all the material needed by an engineer who wishes to evaluate the performance of a computer or communication system. More precisely, with this book and some accompanying practicals, you will be able to answer the above and other questions, evaluate the performance of computer and communication systems and master the theoretical foundations of performance evaluation and of the corresponding software packages.

In the past, many textbooks on performance evaluation have given the impression that this is a complex field, with lots of baroque queuing theory excursions, which can be exercised only by performance evaluation experts. This is not necessarily the case. In contrast, performance evaluation can and should be performed by any computer engineering specialist who designs a system. When a plumber installs pipes in our house, one expects her to properly size their diameters; the same holds for computer engineers.

This book is not intended for the performance evaluation specialist. It is addressed to *every computer engineer or scientist* who is active in the development or operation of software or hardware systems. The required background is an elementary course in probability and one in calculus.

THE OBJECTIVE OF THIS BOOK is therefore to make performance evaluation usable by all computer engineers and scientists. The foundations of performance evaluation reside in statistics and queuing theory, therefore, *some* mathematics is involved and the text cannot be overly simplified. However, it turns out that much of the complications are not in the general theories, but in the exact solution of specific models. For example, some textbooks on statistics (but none of the ones cited in the reference list) develop various solution techniques for specific models, the vast majority of which are encapsulated in commercially or freely available software packages like Matlab, S-PLUS, Excel, Scilab or R.

To avoid this pitfall, we focused first on the *what* before the *how*. Indeed, the most difficult question in a performance analysis is often “what to do”; once you know what to do, it is less difficult to find a way with your usual software tools or by shopping the web. For example, what do we do when we fit a model to data using least square fitting (Chapter 3) ? What is a confidence interval ? What is a prediction interval (Chapter 2) ? What is the congestion collapse pattern (Chapter 1) ? What is the null hypothesis in a test and what does the result of a test *really* mean (Chapter 4) ? What is an information criterion (Chapter 5) ? If no failure appears out of n experiments, what confidence interval can I give for the failure probability (Chapter 2) ?

Second, for the *how*, we looked for solution methods that as universal as possible, i.e. that apply to many situations, whether simple or complex. There are several reasons for this. Firstly, one should use only methods and tools that one understands, and a good engineer should first invest her time learning tools and methods that she will use more often. Secondly, brute force and a computer can do a lot more than one often seems to believe. This philosophy is in sharp contrast to some publications on performance evaluation. For example, computing confidence or prediction intervals can be made simple and systematic if we use the median and not the mean; if we have to employ the mean, the use of likelihood ratio statistic is quite universal and requires little intellectual sophistication regarding the model. Thus, we focus on generic methods such as: the use of filters for forecasting (Chapter 5), bootstrap and Monte-Carlo simulations for evaluating averages or prediction intervals (Chapter 6), the likelihood ratio statistic for tests (Chapter 2, Chapter 4), importance sampling (Chapter 6), least square and ℓ^1 -norm minimization methods (Chapter 3).

When presenting solutions, we tried *not* to hide their limitations and the cases where they do not work. Indeed, some frustrations experienced by young researchers can sometimes be attributed to false expectations about the power of some methods.

We give a coverage of queuing theory that attempts to strike a balance between depth and relevance. During a performance analysis, one is often confronted with the dilemma: should we use an approximate model for which exact solutions exist, or approximate solutions for a more exact model ? We propose four topics (deterministic analysis, operational laws, single queues, queuing networks) which provide a good balance. We illustrate in a case study how the four topics can be utilized to provide different insights on a queuing question. For queuing networks, we give a unified treatment, which is perhaps the first of its kind at this level of synthesis. We show that complex topics such as queues with concurrency (MSCCC queues) or networks with bandwidth sharing (Whittle networks) all fit in the same framework of product form queuing networks. Results of this kind have been traditionally presented as separate; unifying them simplifies the student’s job and provides new insights.

We develop the topic of Palm calculus, also called “the importance of the viewpoint”, which is so central to queuing theory, as a topic of its own. Indeed, this topic has so many applications to simulation and to system analysis in general, that it is a very good time investment. Here too, we

focus on general purpose methods and results, in particular the large-time heuristic for mapping various viewpoints (Chapter 7).

CHAPTER 1 GIVES A METHODOLOGY and serves as introduction to the rest of the book. Performance patterns are also described, i.e. facts that repeatedly appear in various situations, and knowledge of which considerably helps the performance evaluation.

Chapter 2 demonstrates how to summarize experimental or simulation results, as well as how to quantify their accuracy. It also serves as an introduction to a scientific use of the statistical method, i.e. pose a model and verify its assumptions. In Chapter 3 we present general methods for fitting an explanatory model to data and the concept of heavy tail. Chapter 4 describes the techniques of tests, and Chapter 5 those of forecasting. These four chapters give a coverage of modern statistics useful to our field.

Chapter 6 discusses discrete event simulation and several important, though simple issues such as the need for transient removal, for confidence intervals, and classical simulation techniques. We also discuss importance sampling, which is very useful for computing estimates of rare events; we give a simple, though quite general and broadly applicable method.

Chapter 7 describes Palm calculus, which relates the varying viewpoints resulting from measurements done by different operators. Here, we discuss freezing simulations, a phenomenon which can be a problem for even simple simulations if one is not aware of it. We also present how to perform a perfect simulation of stochastic recurrences. Chapter 8 discusses patterns specific to queuing, classical solution methods for queuing networks, and, perhaps more important, operational analysis for rapid evaluation.

The appendix gives background information that cannot yet be easily found elsewhere, such as a Fourier-free quick crash course on digital filters (used in Chapter 5) and confidence intervals for quantiles.

Performance evaluation is primarily an art, and involves using sophisticated tools such as mathematical packages, measurement tools and simulation tools. See the web site of the EPFL lecture on Performance Evaluation for some examples of **practicals**, implemented in matlab and designed around this book.

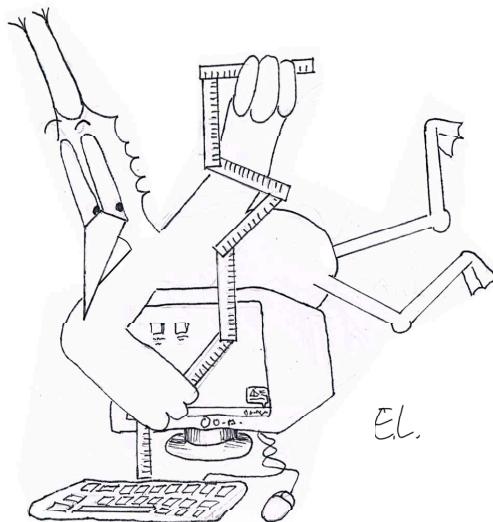
The text is intended for self-study. Proofs are not given when there are easily accessible references (these are indicated in the text); otherwise they can be found in appendixes at the end of the chapters.

The *Index* collects all terms and expressions that are highlighted in the text like **this** and also serves as a notation list.

CHAPTER 1

METHODOLOGY

Perhaps the most difficult part in performance evaluation is where to start. In this chapter we propose a methodology, i.e. a set of recommendations valid for any performance evaluation study. We stress the importance of factors, in particular hidden factors, and the need to use the scientific method. We also discuss a few frequent performance patterns, as a means to quickly focus on important issues.



Contents

1.1	What is Performance Evaluation ?	2
1.1.1	Load	2
1.1.2	Metric	2
1.1.3	The Different Goals of Performance Evaluation	4
1.2	Factors	5
1.2.1	The Hidden Factor Paradox	6
1.2.2	Simpson's Paradox	7
1.3	Evaluation Methods	8
1.4	The Scientific Method	8

1.5 Performance Patterns	10
1.5.1 Bottlenecks	10
1.5.2 Congestion Collapse	12
1.5.3 Competition Side Effect	13
1.5.4 Latent Congestion Collapse	16
1.6 Review	17
1.6.1 Check-List	17
1.6.2 Review Questions	18

1.1 WHAT IS PERFORMANCE EVALUATION ?

In the context of this book, performance evaluation is about quantifying the service delivered by a computer or communication system. For example, we might be interested in: comparing the power consumption of several server farm configurations; knowing the response time experienced by a customer performing a reservation over the Internet; comparing compilers for a multiprocessor machine.

In all cases it is important to carefully define the **load** and the **metric**, and to be aware of the performance evaluation **goals**.

1.1.1 LOAD

An important feature of computer or communication systems is that their performance depends dramatically on the **workload** (or simply **load**) they are subjected to. The load characterizes the quantity and the nature of requests submitted to the system. Consider for example the problem of quantifying the performance of a web server. We could characterize the load by a simple concept such as the number of requests per second. This is called the **intensity of the workload**. In general, the performance deteriorates when the intensity increases, but often the deterioration is sudden; this is due to the non-linearity of queuing systems – an example of **performance pattern** that is discussed in Section 1.5 and Chapter 8.

The performance of a system depends not only on the intensity of the workload, but also its nature; for example, on a web server, all requests are not equivalent: some web server softwares might perform well with *get* requests for frequently used objects, and less well with requests that require database access; for other web servers, things might be different. This is addressed by using standardized mixes of web server requests. They are generated by a **benchmark**, defined as a load generation process that intends to mimic a typical user behaviour. In Chapter 3 we study how such a benchmark can be constructed.

1.1.2 METRIC

A performance **metric** is a measurable quantity that precisely captures what we want to measure – it can take many forms. There is no general definition of a performance metric: it is system

dependent, and its definition requires understanding the system and its users well. We will often mention examples where the metric is throughput (number of tasks completed per time unit), power consumption (integral of the electrical energy consumed by the system, per time unit), or response time (time elapsed between a start and an end events). For each performance metric, we may be interested in average, 95-percentile, worst-case, etc, as explained in Chapter 2.

EXAMPLE 1.1: WINDOWS VERSUS LINUX. Chen and co-authors compare Windows versus Linux in [25]. They use as metric: number of CPU cycles, number of instructions, number of data read/write operations required by a typical job. The load was generated by various benchmarks: “syscall” generates elementary operations (system calls); “memory read” generates references to an array; an application benchmark runs a popular application.

It is also important to be aware of the experimental conditions under which the metric is measured, as illustrated by the coming example:

EXAMPLE 1.2: POWER CONSUMPTION. The electrical power consumed by a computer or telecom equipment depends on how efficiently the equipment can take advantage of low activity periods to save energy. One operator proposes the following metric as a measure of power consumption [29]:

$$P_{\text{Total}} = 0.35P_{\text{max}} + 0.4P_{50} + 0.25P_{\text{sleep}}$$

where P_{Total} is the power consumption when the equipment is running at full load, P_{50} when it is submitted to a load equal to 50% of its capacity and P_{sleep} when it is idle. The example uses weights (0.35, 0.4 and 0.25); they reflect our assumption about the proportion of time that a given load condition typically occurs (for example, the full load condition is assumed to occur during 35% of the time).

In this example, **utilization** is a parameter of the operating conditions. The utilization of a resource is defined as the proportion of time that the resource is busy.

The example also illustrates that it may be important to define which **sampling method** is used, i.e. when the measurements are taken. This is an integral part of the definition of the metric; we discuss this point in more detail in Chapter 7.

A metric may be simple, i.e. expressed by a single number (e.g. power consumption), or **multidimensional**, i.e. expressed by a vector of several numbers (e.g. power consumption, response time and throughput). When comparing two vectors of multidimensional metric values, one should compare the corresponding components (e.g. power consumption of A versus power consumption of B, response time of A versus response time of B, etc). As a result, it may happen that none of the two vectors is better than the other. We say that comparison of vectors is a **partial order**, as opposed to comparison of numbers which is a **complete order**. It is however useful to determine whether a vector is **non-dominated**, i.e. there is no other vector (in the set of available results) which is better. In a finite set of performance results expressed with a multidimensional metric, there are usually more than one non-dominated results. When comparing several configurations, the non-dominated ones are the only ones of interest.

EXAMPLE 1.3: MULTI-DIMENSIONAL METRIC AND KIVIAT DIAGRAM. We measure the performance of a web server submitted to the load of a standard workbench. We compare 5 different

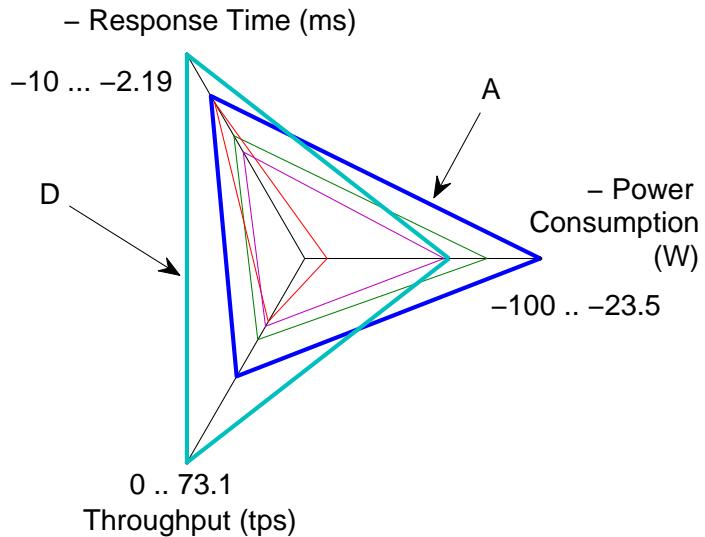


Figure 1.1: Visualisation of the data in Example 1.3 by means of a Kiviat Diagram. Configurations A and D are non-dominated.

configurations, and obtain the results below.

Config	Power (W)	Response (ms)	Throughput (tps)
A	23.5	3.78	42.2
B	40.8	5.30	29.1
C	92.7	4.03	22.6
D	53.1	2.19	73.1
E	54.7	5.92	24.3

We see for example that configuration A is better than B but is not better than D. There are two non dominated configurations: A and D. A is better on power consumption, D is better on throughput and response time.

The numerical values can be visualized on a *Kiviat Diagram* (also called Radar graph or *Spider Plot*) as on Figure 1.1.

1.1.3 THE DIFFERENT GOALS OF PERFORMANCE EVALUATION

The goal of a performance evaluation may either be a *comparison* of design alternatives, i.e. quantify the improvement brought by a design option, or *system dimensioning*, i.e. determine the size of all system components for a given planned utilization. Comparison of designs requires a well-defined load model; however, the exact value of its intensity does not have to be identified. In contrast, system dimensioning requires a detailed estimation of the load intensity. Like any prediction exercise, this is very hazardous. For any performance evaluation, it is important to know whether the results depend on a workload prediction or not. Simple forecasting techniques are discussed Chapter 5.

EXAMPLE 1.4: DIFFERENT GOALS.

QUESTION 1.1.1. Say which is the nature of goal for each of the following performance evaluations statements:¹

- (A1) PC configuration 1 is 25% faster than PC configuration 2 when running Photoshop.
 - (A2) For your video on demand application, the number of required servers is 35, and the number of disk units is 68.
 - (A3) Using the new version of `sendfile()` increases the server throughput by 51%
-

The benefit of a performance evaluation study has to be weighted against its cost and the cost of the system. In practice, detailed performance evaluations are done by product development units (system design). During system operation, it is not economical (except for huge systems such as public communication networks) to do so. Instead, manufacturers provide **engineering rules**, which capture the relation between load intensity and performance. Example (A2) above is probably best replaced by an engineering rule such as:

EXAMPLE 1.5: ENGINEERING RULE.

- (E2) For your video on demand application, the number of required servers is given by $N_1 = \lceil \frac{R}{59.3} + \frac{B}{3.6} \rceil$ and the number of disk units by $N_2 = \lceil \frac{R}{19.0} + \frac{B}{2.4} \rceil$, where R [resp. B] is the number of residential [resp. business] customers ($\lceil x \rceil$ is the ceiling of x , i.e. the smallest integer $\geq x$).
-

In this book, we study the techniques of performance evaluation that apply to all these cases. However, how to implement a high performance system (for example: how to efficiently code a real time application in Linux) or how to design bug-free systems are *outside* the scope.

1.2 FACTORS

After defining goal, load and metric, one needs to establish a list of **factors**: these are elements in the system or the load that affect the performance. One is tempted to focus only on the factor of interest, however, it is important to know all factors that may impact the performance measure, whether these factors are desired or not.

EXAMPLE 1.6: WINDOWS VERSUS LINUX, CONTINUED. In [25], Chen and co-authors consider the following external factors: background activity; multiple users; network activity. These were reduced to a minimum by shutting the network down and allowing one single user. They also consider: the different ways of handling idle periods in Windows and Limux, because they affect the interpretation of measurements.

¹(A1), (A3) are comparisons of design options; (A2) is dimensioning

1.2.1 THE HIDDEN FACTOR PARADOX

Ignoring some hidden factors may invalidate the result of the performance evaluation, as the next example shows.

EXAMPLE 1.7: TCP THROUGHPUT. Figure 1.2, left, plots the throughput achieved by a mobile during a file transfer as a function of its velocity (speed). It suggests that throughput increases with mobility. The right plot shows the same data, but now the mobiles are separated in two groups: one group ('s') is using a small socket buffer (4K Bytes), whereas the second ('L') uses a larger socket buffer (16 K Bytes). The conclusion is now inverted: throughput decreases with mobility. The hidden factor influences the final result: all experiments with low speed are for small socket buffer sizes. The socket buffer size is a hidden factor.

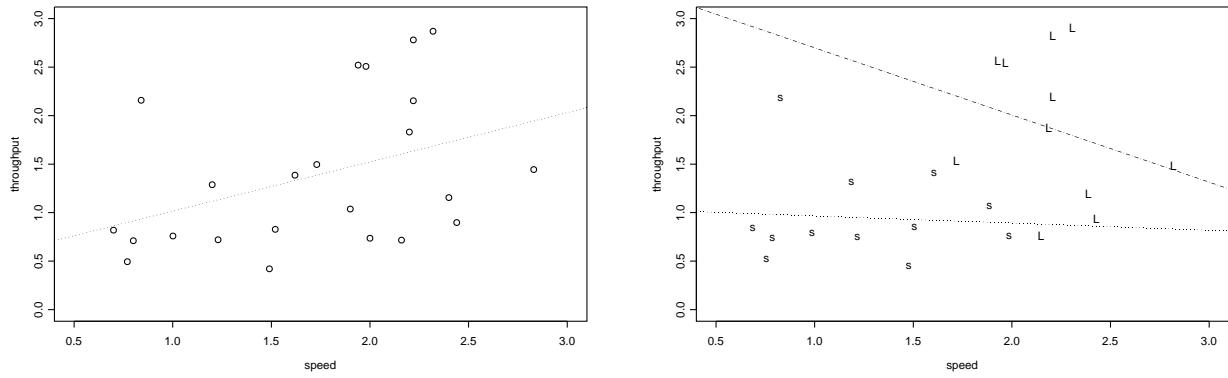


Figure 1.2: Left: plot of throughput (in Mb/s) versus speed (in m/s) for a mobile node. Right: same plot, but showing socket buffer size; s = small buffer, L = large buffer.

Avoiding hidden factors may be done by proper randomization of the experiments. On the example above, a proper design would have distributed socket buffer sizes randomly with respect to the speed.

However, this may not always be possible as some experimental conditions may be imposed upon us; in such cases, all factors have to be incorporated in the analysis. On Figure 1.2, we fitted a linear regression to the two figures, using the method explained in Chapter 3. The slope of the linear regression is negative when we explicit the hidden factor, showing that mobility decreases throughput.

The importance of hidden factors may be interpreted as our tendency to confound cause and correlation [77]. In Figure 1.2, left, the throughput is positively correlated with the speed, but this may not be interpreted as a causal relationship.

In conclusion at this point, knowing all factors is a tedious, but necessary task. In particular, all factors should be incorporated, whether you are interested in them or not (factors that you are not interested in are called *nuisance factors*). This implies that you have to know your system well, or be assisted by people who know it well.

1.2.2 SIMPSON'S PARADOX

Simpson's reversal, also known as *Simpson's paradox* is a well known case of the problem of hidden factors, when the performance metric is a success probability.

EXAMPLE 1.8: **TCP THROUGHPUT, CONTINUED.** We revisit the previous example, but are now interested only in knowing whether a mobile can reach a throughput of at least 1.5 Mb/s, i.e. we say that a mobile is successful if its throughput is ≥ 1.5 Mb/s. We classify the mobiles as slow (speed ≤ 2 m/s) or fast (speed > 2 m/s). We obtain the following result.

	failure	success		$\mathbb{P}(\text{success})$
slow	11	3	14	0.214
fast	5	4	9	0.444
	16	7	23	

from where we conclude that fast mobiles have a higher success probability than slow ones. Now introduce the nuisance parameter "socket buffer size", i.e. we qualify the mobiles as 's' (small buffer size) or 'L' (large buffer size):

's' mobiles	failure	success		$\mathbb{P}(\text{success})$
slow	10	1	11	0.091
fast	1	0	1	0.00
	11	1	12	
'L' mobiles	failure	success		$\mathbb{P}(\text{success})$
slow	1	2	3	0.667
fast	4	4	8	0.500
	5	6	11	

Now in both cases slow mobiles have a higher success probability than fast ones, which is the correct answer. The former answer was wrong because it ignored a hidden factor. This is known as Simpson's reversal.

Simpson's paradox can be formulated in general as follows [65]. Let S denote the fact that the outcome of an experiment is a success, and let C be the factor of interest (in the example, mobile speed). Let N_i , $i = 1 \dots k$ be binary hidden factors (nuisance factors; in the example, there is only one, the socket buffer size). Assume that the factor of interest has a positive influence on the success rate, i.e.

$$\mathbb{P}(S|C) > \mathbb{P}(S|\bar{C}) \quad (1.1)$$

This may happen while, at the same time, the combination of the factor of interest with the hidden factors N_i has the opposite effect:

$$\mathbb{P}(S|C \text{ and } N_i) < \mathbb{P}(S|\bar{C} \text{ and } N_i) \quad (1.2)$$

for all $i = 1 \dots k$. As illustrated in Examples 1.8 and 1.7, the reversal occurs when the effect of hidden factors is large.

The fact that Simpson's reversal is a paradox is assumed to originate in our (false) intuition that an average of factors leads to an average of outcomes, i.e. we may (wrongly) assume that Eq.(1.1) is a weighted sum of Eq.(1.2).

We do have weighted sums, but the weights are $\mathbb{P}(N_i|C)$ for the left-handside in Eq.(1.1) versus $\mathbb{P}(N_i|\bar{C})$ for the right-handside:

$$\begin{aligned}\mathbb{P}(S|C) &= \sum_i \mathbb{P}(S|C \text{ and } N_i) \mathbb{P}(N_i|C) \\ \mathbb{P}(S|\bar{C}) &= \sum_i \mathbb{P}(S|\bar{C} \text{ and } N_i) \mathbb{P}(N_i|\bar{C})\end{aligned}$$

1.3 EVALUATION METHODS

Once goal, load, metric and factors are well defined, performance evaluation can then proceed with a solution method, which usually falls in one of the three cases below. Which method to use depends on the nature of the problem and the skills or taste of the evaluation team.

- **Measurement** of the real system. Like in physics, it is hard to measure without disturbing the system. Some special hardware devices (e.g.: optical splitters in network links) sometimes can prevent any disturbance. If, in contrast, measurements are taken by the system itself, the impact has to be analyzed carefully. Measurements are not always possible (eg. if the system does not exist yet).
- Discrete Event **Simulation**: a simplified model of the system and its load are implemented in software. Time is simulated and often flows orders of magnitude more slowly than real time. The performance of interest is measured as on a real system, but measurement side-effects are usually not present. It is often easier than a measurement study, but not always. It is the most widespread method and is the object of Chapter 6.
- **Analytical**: A mathematical model of the system is analyzed numerically. This is viewed by some as a special form of simulation. It is often much quicker than simulation, but sometimes wild assumptions need to be made in order for the numerical procedures to be applicable. Analytical methods are often used to gain insight during a development phase, or also to learn fundamental facts about a system, which we call “patterns”. We show in Chapter 8 how some performance analyses can be solved approximately in a very simple way, using bottleneck analysis.

1.4 THE SCIENTIFIC METHOD

The scientific method applies to any technical work, not only to performance evaluation. However, in the author’s experience, lack of scientific method is one prominent cause for failed performance studies. In short, the scientific method requires that you do not believe in a conclusion unless it is thoroughly tested.

EXAMPLE 1.9: **JOE’S KIOSK**. Joe’s e-kiosk sells online videos to customers equipped with smart-phones. The system is made of one servers and one 802.11 base station. Before deployment,

performance evaluation tests are performed, as shown on Figure 1.3(a). We see that the throughput reaches a maximum at around 8 transactions per second.

Joe concludes that the bottleneck is the wireless LAN and decides to buy and install 2 more base stations. After installation, the results are on Figure 1.3(b). Surprisingly, there is no improvement. The conclusion that the wireless LAN was the bottleneck was wrong.

Joe scratches his head and decides to go more carefully about conclusions. Measurements are taken on the wireless LAN; the number of collisions is less than 0.1%, and the utilization is below 5%. This confirms that the wireless LAN is *not* a bottleneck. Joe makes the hypothesis that the bottleneck may be on the server side. After doubling the amount of real memory allocated to the server process, the results are as shown on Figure 1.3(c). This confirms that real memory was the limiting factor.

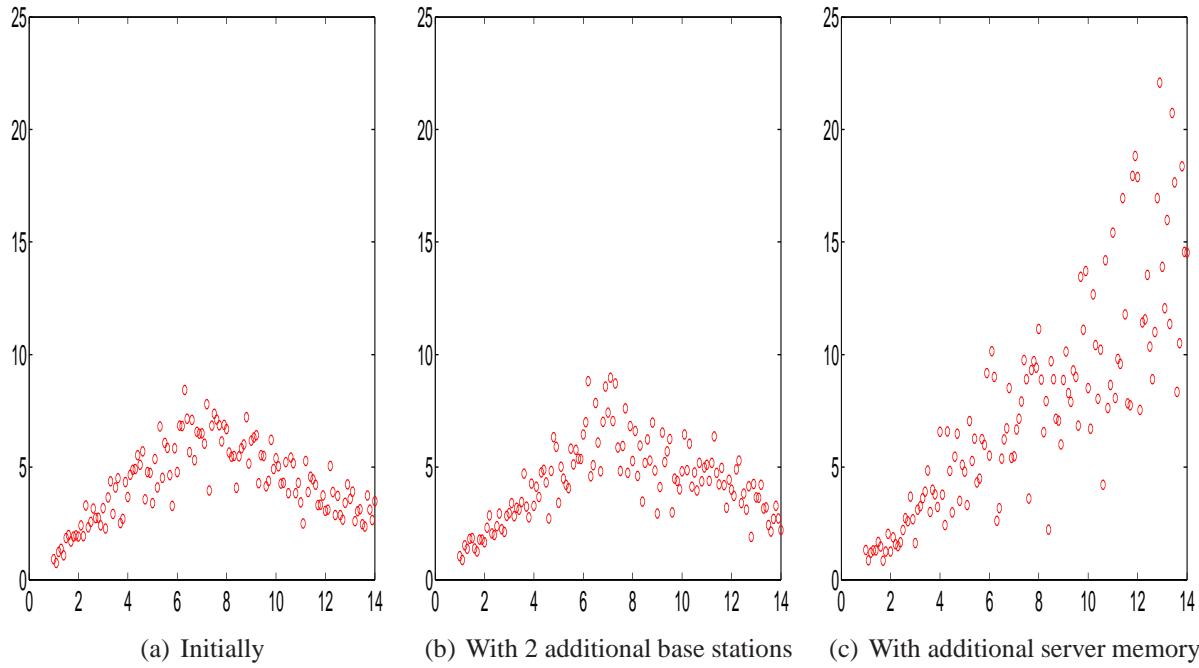


Figure 1.3: Performance results for Joe's server. X-axis: offered load; Y-axis: achieved throughput, both in transactions per second.

A common pitfall is to draw conclusions from an experiment that was not explicitly designed to validate these conclusion. The risk is that hidden factors might interfere, as illustrated by the previous example. Indeed, Joe concluded from the first experiment that the LAN performance would be improved by adding a base station; this may have been *suggested* by the result of Figure 1.3(a), but this is not sufficient. It is necessary to perform other experiments, designed to validate this potential conclusion, before making a final statement. Following Popper's philosophy of science [82], we claim that it is necessary for the performance analyst to take both roles : (1) make tentative statements, and (2) design experiments that try to invalidate them.

EXAMPLE 1.10: ATM UBR BETTER THAN ATM ABR. In [66], the authors evaluate whether the ATM-UBR protocol is better than ATM-ABR (both are alternative methods used to manage switches used in communication networks). They use a typical scientific method, by posing each

potential conclusion as a hypothesis and designing experiments to try and invalidate them:

ABSTRACT. We compare the performance of ABR and UBR for providing high-speed network interconnection services for TCP traffic. We test the hypothesis that UBR with adequate buffering in the ATM switches results in better overall goodput for TCP traffic than explicit rate ABR for LAN interconnection. This is shown to be true in a wide selection of scenarios. Four phenomena that may lead to bad ABR performance are identified and we test whether each of these has a significant impact on TCP goodput. This reveals that the extra delay incurred in the ABR end-systems and the overhead of RM cells account for the difference in performance. We test whether it is better to use ABR to push congestion to the end-systems in a parking-lot scenario or whether we can allow congestion to occur in the network. Finally, we test whether the presence of a “multiplexing loop” causes performance degradation for ABR and UBR. We find our original hypothesis to be true in all cases. We observe, however, that ABR is able to improve performance when the buffering inside the ABR part of the network is small compared to that available at the ABR end-systems. We also see that ABR allows the network to control fairness between end-systems.

Other aspects of the scientific method are:

- Give an evaluation of the **accuracy** of your quantitative results. Consider the measured data in Table 1.11. There is a lot of variability in them; saying that the average response time is better with B than A is not sufficient; it is necessary to give uncertainty margins, or confidence intervals. Techniques for this are discussed in Chapter 2.
- Make the results of your performance evaluation easily **reproducible**. This implies that all assumptions are made explicit and documented.
- Remove what can be removed. Often, at the end of a performance evaluation study, many results are found uninteresting; the right thing to do is to remove such results, but this seems hard in practice !

1.5 PERFORMANCE PATTERNS

Performance evaluation is simpler if the evaluator is aware of performance **patterns**, i.e. traits that are common to many different settings.

1.5.1 BOTTLENECKS

A prominent pattern is **bottlenecks**. In many systems, the overall performance is dictated by the behaviour of the weakest components, called the bottlenecks.

EXAMPLE 1.11: BOTTLENECKS. You are asked to evaluate the performance of an information system. An application server can be compiled with two options, A and B. An experiments was done: ten test users (remote or local) measured the time to complete a complex transaction on four days. On day 1, option A is used; on day 2, option B is. The results are in the table below.

	remote	local		remote	local
A	123	43	B	107	62
189	38		179	69	
99	49		199	56	
167	37		103	47	
177	44		178	71	

The expert concluded that the performance for remote users is independent of the choice of an information system. We can criticize this finding and instead do a bottleneck analysis. For remote users, the bottleneck is the network access; the compiler option has little impact. When the bottleneck is removed, i.e. for local users, option A is slightly better.

Bottlenecks are the performance analysts' friend, in the sense that they may considerably **simplify the performance evaluation**, as illustrated next.

EXAMPLE 1.12: CPU MODEL. A detailed screening of a transaction system shows that one transaction costs in average: 1'238'400 CPU instructions; 102.3 disk accesses and 4 packets sent on the network. The processor can handle 10^9 instructions per second; the disk can support 10^4 accesses per second; the network can support 10^4 packets per second. We would like to know how many transactions per second the system can support.

The resource utilization per transaction per second is: CPU: 0.12% – disk: 1.02% –network: 0.04%; therefore the disk is the bottleneck. The capacity of the system is determined by how many transactions per second the disk can support, a gross estimate is thus $\frac{100}{1.02} \approx 99$ transactions per second.

If we would like more accuracy, we would need to model queuing at the disk, to see at which number of transactions per seconds delays start becoming large. A global queuing model of CPU, disk access and network is probably not necessary.

In Section 8.2.4 we study bottleneck analysis for queuing systems in a systematic way.

However, one should not be fooled by the apparent simplicity of the previous example, as bottlenecks are moving targets. They depend on all parameters of the system and on the load: a component may be a bottleneck in some conditions, not in others. In particular, removing a bottleneck may let some other bottleneck appear.

EXAMPLE 1.13: HIGH PERFORMANCE WEB SITES. In [99], the author discusses how to design high performance web sites. He takes as performance metric user's response time. He observes that modern web sites have highly optimized backends, and therefore their bottleneck is at the front end. A common bottleneck is DNS lookup; entirely avoiding DNS lookups in web pages improves performances, but reveals another bottleneck, namely, script parsing. This in turn can be avoided by making scripts external to the web page, but this will reveal yet another bottleneck, etc. The author describes 14 possible components, any of which, if present, is candidate for being the bottleneck, and suggests to remove all of them. Doing so leaves as bottlenecks network access and server CPU speed, which is desirable.

1.5.2 CONGESTION COLLAPSE

Congestion occurs when the intensity of the load exceeds system capacity (as determined by the bottleneck). Any system, when subject to a high enough load, will become congested: the only way to prevent this is to limit the load, which is often difficult or impossible. Therefore, it is difficult to avoid congestion entirely.

In contrast, it is possible, and desirable, to avoid **congestion collapse**, which is defined as a reduction in system utility, or revenue when the load increases.

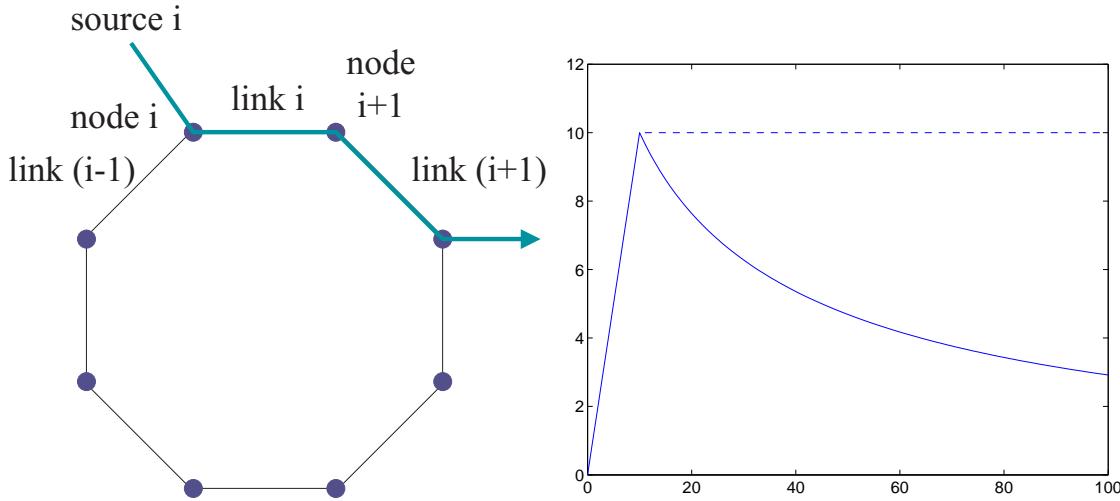


Figure 1.4: First panel: A network exhibiting congestion collapse if sources are greedy. Second panel: throughput per source λ'' versus offered load per source λ , in Mb/s (plain line). Numbers are in Mb/s; the link capacity is $c = 20$ Mb/s for all links. Dotted line: ideal throughput with congestion but without congestion collapse.

EXAMPLE 1.14: CONGESTION COLLAPSE. Consider a ring network as in Figure 1.4 (such a topology is common, as it is a simple way to provide resilience single link or node failure). There are I nodes and links, and sources numbered $0, 1, \dots, I - 1$. At every node there is one source, whose traffic uses the two next downstream links (i.e. source i uses links $[(i + 1) \bmod I]$ and $[(i + 2) \bmod I]$). All links and sources are identical.

Every source sends at a rate λ and let c be the useful capacity of a link (c and λ are in Mb/s). Let λ' the rate achieved by one source on its first hop, λ'' on its second hop (λ'' is the throughput per source). Since a source uses two links, we can assume (in a simplified analysis) that, as long as $\lambda < \frac{c}{2}$, all traffic is carried by the network without loss, i.e.

$$\text{if } \lambda < \frac{c}{2} \text{ then } \lambda' = \lambda'' = \lambda$$

Assume now that sources are greedy and send as much as they can, with a rate $\lambda > \frac{c}{2}$. The network capacity is exceeded, therefore there will be losses. We assume that packet dropping is fair, i.e. the proportion of dropped packets is the same for all flows at any given link. The proportion of packets lost by one source on its first hop is $\frac{\lambda - \lambda'}{\lambda}$; on its second hop it is $\frac{\lambda' - \lambda''}{\lambda'}$. By the fair packet dropping assumption, those proportions are equal, therefore

$$\frac{\lambda'}{\lambda} = \frac{\lambda''}{\lambda'} \quad (1.3)$$

Furthermore, we assume that links are fully utilized when capacity is reached, i.e.

$$\text{if } \lambda > \frac{c}{2} \quad \text{then } \lambda' + \lambda'' = c$$

We can solve for λ' (a polynomial equation of degree 2) and substitute λ' in Eq.(1.3) to finally obtain the throughput per source:

$$\lambda'' = c - \frac{\lambda}{2} \left(\sqrt{1 + 4\frac{c}{\lambda}} - 1 \right) \quad (1.4)$$

Figure 1.4 plots λ'' versus λ ; it suggests that $\lambda'' \rightarrow 0$ as $\lambda \rightarrow \infty$. We can verify this by using a Taylor expansion of $\sqrt{1+u}$, for $u \rightarrow 0$ in Eq.(1.4). We obtain

$$\lambda'' = \frac{c^2}{\lambda} (1 + \epsilon(\lambda))$$

with $\lim_{\lambda \rightarrow \infty} \epsilon(\lambda) = 0$. which shows that the limit of the achieved throughput, when the offered load goes to $+\infty$, is 0. This is a clear case of **congestion collapse**.

Figure 1.4 also illustrates the difference between congestion and congestion collapse. The dotted line represents the ideal throughput per source if there would be congestion without congestion collapse; this could be achieved by employing a feedback mechanism to prevent sources from sending more than $\frac{c}{2}$ (for example by using TCP).

Two common causes for congestion collapse are:

1. The system dedicates significant amounts of resources to jobs that will not complete, as in Figure 1.4, where packets are accepted on the first hop, which will eventually be dropped on the second hop. This is also known to occur on busy web sites or call centers due to customer **impatience**: when response time gets large impatient customers drop requests before they complete.
2. The service time per job increases as the load increases. This occurs for example when memory is paged to disk when the number of active processes increases.

Congestion collapse is very common in complex systems. It is a nuisance since it reduces the total system utility below its capacity. Avoiding congestion collapse is part of good system design. A common solution to the problem is **admission control**, which consists in rejecting jobs when there is a risk that system capacity would be exceeded [50].

1.5.3 COMPETITION SIDE EFFECT

In many systems the performance of one user influence other users. This may cause an apparent paradox, where putting more resources makes the performance worse for some users. The root cause is as follows: increasing some resources may allow some users to increase their load, which may in turn decrease the performance of competing users. From the point of view of the user whose performance is decreased, there is an apparent paradox: resources were added to the system, with an adverse effect.

EXAMPLE 1.15: COMPETING USERS WITH IDEAL CONGESTION CONTROL. Figure 1.5 shows a simple network with 2 users, 1 and 2, sending traffic to destinations D1 and D2 respectively. Both users share a common link $X - Y$.

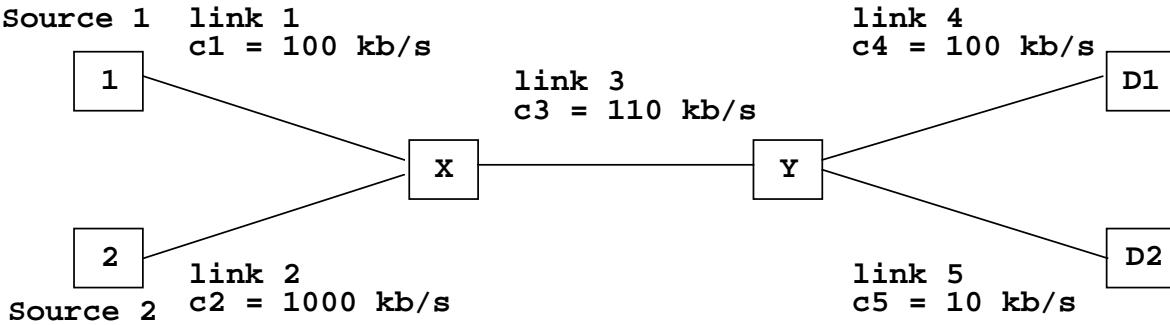


Figure 1.5: A simple network with two users showing the pattern of competition side effect. Increasing the capacity of link 5 worsens the performance of user 1.

Assume that the sources use some form of congestion control, for example because they use the TCP protocol. The goal of congestion control is to limit the source rates to the system capacity while maintaining some fairness objective. We do not discuss fairness in detail in this book, see for example [50] for a quick tutorial; for simplicity, we may assume here that congestion control has the effect of maximizing the logarithms of the rates of the sources, subject to the constraints that all link capacities are not exceeded (this is called *proportional fairness* and is approximately what TCP implements). Let x_1 and x_2 be the rates achieved by sources 1 and 2 respectively. With the numbers shown on the figure, the constraints are $x_1 \leq 100\text{kb/s}$ and $x_2 \leq 10\text{kb/s}$ (other constraints are redundant) so we will have $x_1 = 100\text{kb/s}$ and $x_2 = 10\text{kb/s}$.

Assume now that we add resources to the system, by increasing the capacity of link 5 (the weakest link) from $c_5 = 10\text{kb/s}$ to $c_5 = 100\text{kb/s}$. The constraints are now

$$\begin{aligned}x_1 &\leq 100 \text{ kb/s} \\x_2 &\leq 100 \text{ kb/s} \\x_1 + x_2 &\leq 110 \text{ kb/s}\end{aligned}$$

By symmetry, the rates allocated under proportional fairness are thus $x_1 = x_2 = 55\text{kb/s}$. We see that increasing capacity has resulted in a decrease for source 1.

The competition side effect pattern in the previous example is a “good” case, in the sense that the decrease in performance for some users is compensated by an increase for others. But this is not always true; combined with the ingredients of congestion collapse, the competition side effect may result in a performance decrease without any benefit for any user (“put more, get less”), as shown in the next example.

EXAMPLE 1.16: COMPETING USERS WITHOUT CONGESTION CONTROL. Consider Figure 1.5 again, but assume that there is no congestion control (for example because sources use UDP instead of TCP). Assume that sources send as much as their access link allows, i.e. source 1 sends at the rate of link 1 and source 2 at the rate of link 2.

Assume that we keep all rates as shown on the figure, except for the rate of link 2, which we vary from $c_2 = 0$ to $c_2 = 1000\text{kb/s}$. Define now the rates x_1 and x_2 as the amounts of traffic that do reach the destinations.

If $c_2 \leq 10\text{kb/s}$, there is no loss and $x_1 = 100\text{kb/s}$, $x_2 = c_2$. If $c_2 > 10\text{kb/s}$, there are losses at X. Assume losses are in proportion to the offered traffic. Using the same analysis as in Example 1.14,

we obtain, for $c_2 > 10$:

$$\begin{aligned}x_1 &= 110 \times \frac{100}{c_2 + 100} \\x_2 &= \min \left(110 \times \frac{c_2}{c_2 + 100}, 10 \right)\end{aligned}$$

Figure 1.6 plots the rates versus c_2 . We see that increasing c_2 beyond 10kb/s makes things worse for source 1, with no benefit for source 2.

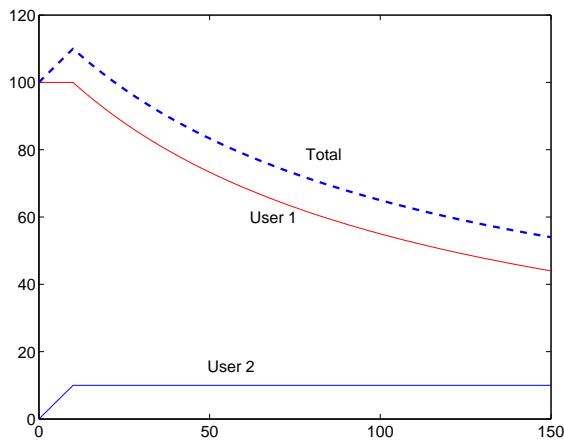
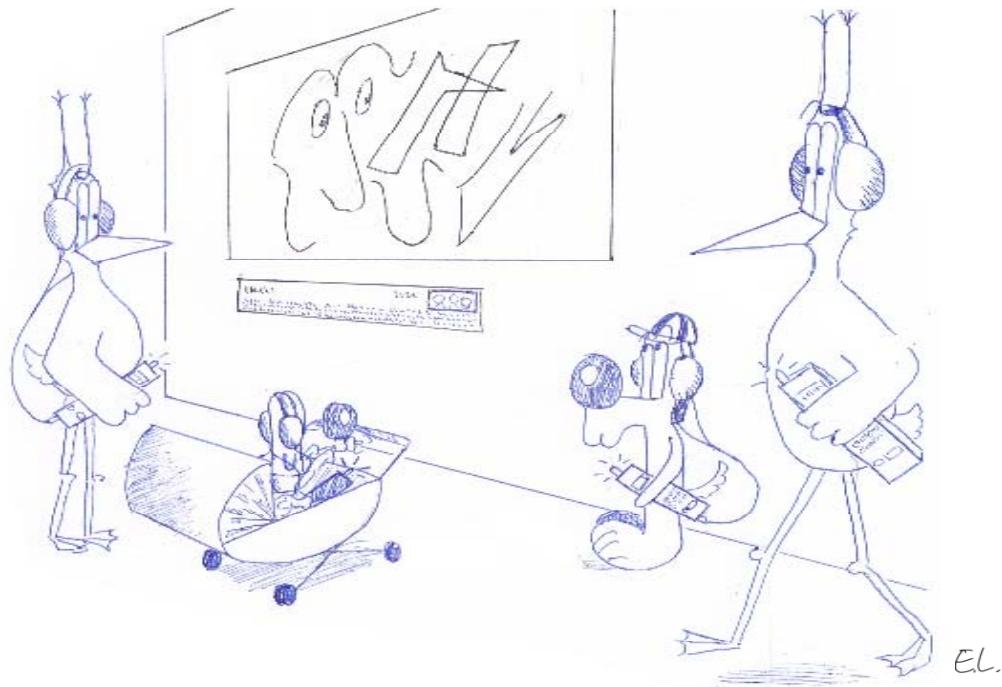


Figure 1.6: Achieved throughputs for the sources in Figure 1.5 versus c_2 .

1.5.4 LATENT CONGESTION COLLAPSE



Many complex systems have several potential bottlenecks, and may be susceptible to congestion collapse. Removing a bottleneck (by adding more resources) may reveal a congestion collapse, resulting in worse performance. Before resources were added, the system was protected from congestion collapse by the bottleneck, which acted as implicit admission control. This results in the “put more, get less” paradox.

EXAMPLE 1.17: MUSEUM AUDIO GUIDES. A museum offers free audio guides to be downloaded on MP3 players. Visitors put their MP3 player into docking devices. The docking devices connect via a wireless LAN to the museum server. Data transfer from the docking device to the MP3 player is via a USB connector. The system was tested with different numbers of docking devices; Figure 1.7(a) shows the download time versus the number of docking devices in use.

The museum later decides to buy better docking devices, with a faster USB connection between device and MP3 player (the transfer rate is now doubled). As expected, the download time is smaller when the number n of docking devices is small, but, surprisingly, it is larger when $n \geq 7$ (Figure 1.7(a)). What may have happened? It is known that the wireless LAN access method is susceptible to congestion collapse: when the offered load increases, packet collisions become frequent and the time to successfully transfer one packet becomes larger, so the throughput decreases. We may conjecture that improving the transfer speed between docking device and MP3 player increases the load on the wireless LAN. The congestion collapse was not possible before because the low speed docking devices acted as an (involuntary) access control method.

We can verify this conjecture by plotting throughput instead of download time, and extending the first experiment to large values of n . We see on Figure 1.7(b) that there is indeed a reduction in throughput, at a point that depends on the speed of the USB connection.

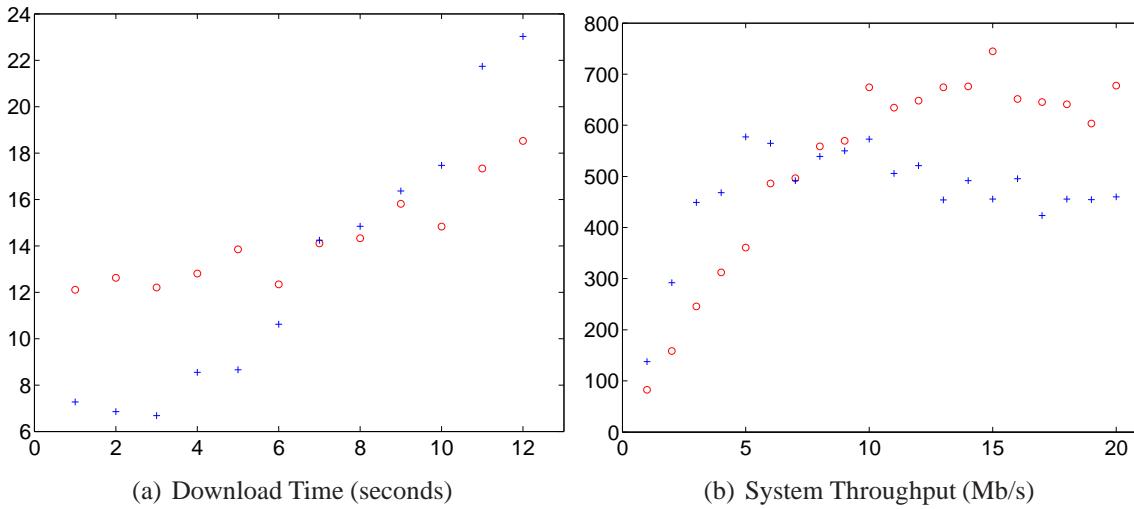


Figure 1.7: Illustration of latent congestion collapse. Download time and System throughput as a function of the number of docking devices, with lower speed USB connections (o) with higher speed USB connections (+).

1.6 REVIEW

1.6.1 CHECK-LIST

PERFORMANCE EVALUATION CHECKLIST

- PE1 Define your goal.** For example: dimension the system, find the overload behaviour; evaluate alternatives. Do you need a performance evaluation study ? Aren't the results obvious ? Are they too dependent on the input factors, which are arbitrary ?
- PE2 Identify the factors.** What are all the factors ? are there external factors which need to be controlled ?
- PE3 Define your metrics.** For example: response time, server occupancy, number of transactions per hour, Joule per Megabyte. Define not only what is measured but also under which condition or sampling method. If the metric is multidimensional, different metric values are not always comparable and there may not be a best metric value. However, there may be non dominated metric values.
- PE4 Define the offered load.** How is it expressed: transactions per second, number of users, number of visits per hour ? Is it measured on a real system ? artificial load generated by a simulator, by a synthetic load generator ? load model in a theoretical model ?
- PE5 Know your bottlenecks.** The performance often depends only on a small number of factors, often those whose utilization (= load/capacity) is high. Make sure what you are evaluating is one of them.
- PE6 Know your system well.** Know the system you are evaluating and list all factors. Use evaluation tools that you know well. Know common performance patterns for your system.

SCIENTIFIC METHOD CHECKLIST

S1 Scientific Method

1.6.2 REVIEW QUESTIONS

QUESTION 1.6.1. *For each of the following examples:*

1. Design web server code that is efficient and fast.
2. Compare TCP-SACK versus TCP-new Reno for hand-held mobile devices.
3. Compare Windows 2000 Professional versus Linux.
4. Design a rate control for an internet audio application.
5. Compare various wireless MAC protocols.
6. Say how many servers a video on demand company needs to install.
7. Compare various compilers.
8. How many control processor blades should this Cisco router have ?
9. Compare various consensus algorithms.
10. Design bug-free code.
11. Design a server farm that will not crash when the load is high.
12. Design call center software that generates guaranteed revenue.
13. Size a hospital's information system.
14. What capacity is needed on an international data link ?
15. How many new servers, if any, should I install next quarter for my business application ?

say whether a detailed identification of the intensity of the workload is required.²

QUESTION 1.6.2. *Consider the following scenarios.*

1. The web server used for online booking at the “Fête des Vignerons” was so popular that it collapsed under the load, and was unavailable for several hours.
2. Buffers were added to an operating system task, but the overall performance was degraded (instead of improved, as expected).
3. The response time on a complex web server is determined primarily by the performance of the front end.
4. When too many users are using the international link, the response time is poor
5. When too many users are present on the wireless LAN, no one gets useful work done
6. A traffic volume increase of 20% caused traffic jams
7. New parking facilities were created in the city center but free parking availability did not increase.

and the following patterns

- (a) non-linearity of response time with respect to load
- (b) congestion collapse (useful work decreases as load increases)
- (c) performance is determined by bottleneck

Say which pattern is present in which scenario³

QUESTION 1.6.3. *Read [63], written by one of Akamai's founders. What topics in this chapter does this illustrate ?⁴*

²Examples 6, 8, 13, 14, 15 are dimensioning exercises and require identification of the predicted workload intensity. Examples 1 and 10 are outside the scope of the book. Examples 11 and 12 are about avoiding congestion collapse.

³1b; 2: perhaps a combination of b and c; 3c; 4a; 5b; 6b; 7c

⁴(1) The performance bottleneck in internet response time is the *middle mile*, i.e. the intermediate providers

between web site provider and end-user ISP. (2) performance metrics of interest are not only response time but also reliability.

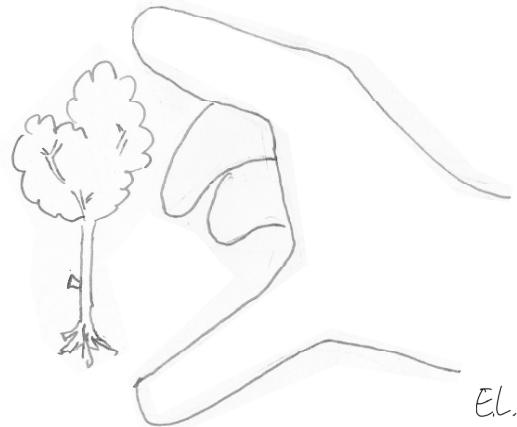
CHAPTER 2

SUMMARIZING PERFORMANCE DATA, CONFIDENCE INTERVALS

In most measurements or simulations, we obtain large amounts of data. Displaying the data correctly is important, and implies to use some graphical packages, or maths packages with graphical capabilities (Different tools have different capabilities, and produce graphics of different aesthetic value; but the most important is to use one tool that you know well). Tools do not do everything and you need to know what to represent. We discuss important and frequent summarizations that can be used to display and compare data: the complete distribution; summarized quantities such as mean, standard deviation, median and tail quantiles; fairness indices.

We discuss some properties of these summarization and indices; they are not all equivalent, and some, though less frequently used, are preferable if one has a choice; for example, the Lorenz curve gap is more robust than Jain's Fairness index (which is essentially the same as the standard deviation rescaled by the mean) and should be preferred.

Simulation and measurement usually contain some randomness, therefore it is important to capture the uncertainty about measured performance. This is done with confidence or prediction intervals; we discuss the use and interpretation of both. There are many different ways for defining a confidence or prediction interval; some are robust and some not. We give useful, and simple formulas and show that, if one has a choice, intervals based on medians and quantiles should be preferred to the more classical mean and standard deviation. We also give useful, though little known results



such as how to compute a confidence interval for a success probability when has seen no success.

Contents

2.1 Summarized Performance Data	22
2.1.1 Histogram and Empirical CDF	22
2.1.2 Mean, Median and Quantiles	23
2.1.3 Coefficient of Variation and Lorenz Curve Gap	25
2.1.4 Fairness Indices	26
2.2 Confidence Intervals	31
2.2.1 What is a Confidence Interval ?	31
2.2.2 Confidence Interval for Median and Other Quantiles	32
2.2.3 Confidence Interval for the Mean	34
2.2.4 Confidence Intervals for Fairness Indices and The Bootstrap	36
2.2.5 Confidence Interval for Success Probability	38
2.3 The Independence Assumption	39
2.3.1 What does iid mean ?	39
2.3.2 How do I know in Practice if the iid Assumption is Valid ?	40
2.3.3 What Happens If The IID Assumption Does Not Hold ?	41
2.4 Prediction Interval	44
2.4.1 Prediction for an IID Sample based on Order Statistic	45
2.4.2 Prediction for a Normal IID Sample	46
2.4.3 The Normal Assumption	47
2.5 Which Summarization To Use ?	49
2.5.1 Robustness	49
2.5.2 Compactness	52
2.6 Other Aspects of Confidence/Prediction Intervals	53
2.6.1 Intersection of Confidence/Prediction Intervals	53
2.6.2 The Meaning of Confidence	53
2.7 Proofs	54
2.8 Review	55
2.8.1 Summary	55
2.8.2 Review Questions	56

2.1 SUMMARIZED PERFORMANCE DATA

2.1.1 HISTOGRAM AND EMPIRICAL CDF

Assume you have obtained a large set of results for the value of a performance metric. This can be fully described by the distribution of the data, and illustrated by a *histogram*. A histogram uses

bins for the data values and plots on the y -axis the proportion of data samples that fall in the bin on the x axis, see Figure 2.1.

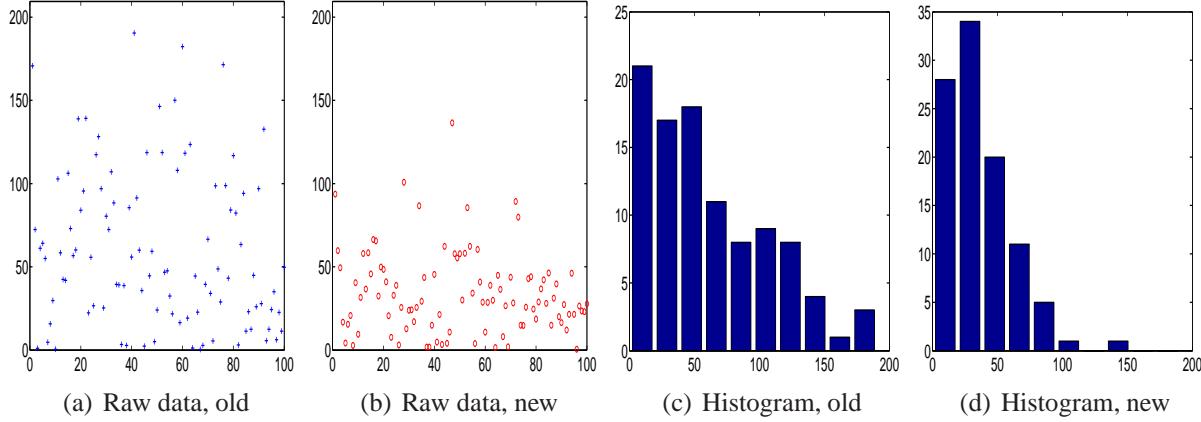


Figure 2.1: Data for Example 2.1. Measured execution times, in ms, for 100 transactions with the old and new code, with histograms.

The *empirical cumulative distribution function* (ECDF) is an alternative to histograms which sometimes makes comparisons easier. The ECDF of a data set x_1, \dots, x_n is the function F defined by

$$F(x) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{\{x_i \leq x\}} \quad (2.1)$$

so that $F(x)$ is the proportion of data samples that do not exceed x .

Data sets may be compared by means of their ECDFs. If one is always above the other, then one may consider that it is superior, even though some data points in the first set may be less good (this is called *stochastic majorization*). On Figure 2.2 we see that the new data set (left) clearly outperforms the old one. Note that stochastic majorization is a partial order, as is the comparison of multidimensional metrics (Section 1.1.2).

Assume the data samples come from a well defined probability distribution; the histogram can then be viewed as an estimate of the PDF of the distribution, and the ECDF as an estimate of the CDF¹.

2.1.2 MEAN, MEDIAN AND QUANTILES

Instead of considering entire histograms or ECDFs, one often would like to summarize, i.e. compress the histogram into one or a few numbers that represent both average and variability. This is commonly done by either one of the following:

Median and Quantile. A median is a value that falls in the middle of the distribution, i.e. 50% of the data is below and 50% above. A $p\%$ -quantile leaves $p\%$ of the observation below and $(100 - p)\%$ above. The median gives some information about the average, while extreme quantiles give information about the dispersion. A commonly used plot is the *Box Plot*. It shows the median, the 25% and 75% quantiles (called “quartiles”) and the “outliers”, defined as data points that are a fixed fraction away from the quartiles (Figure 2.3).

¹The CDF of the random variable X is the function defined by $F(x) = \mathbb{P}(X \leq x)$.

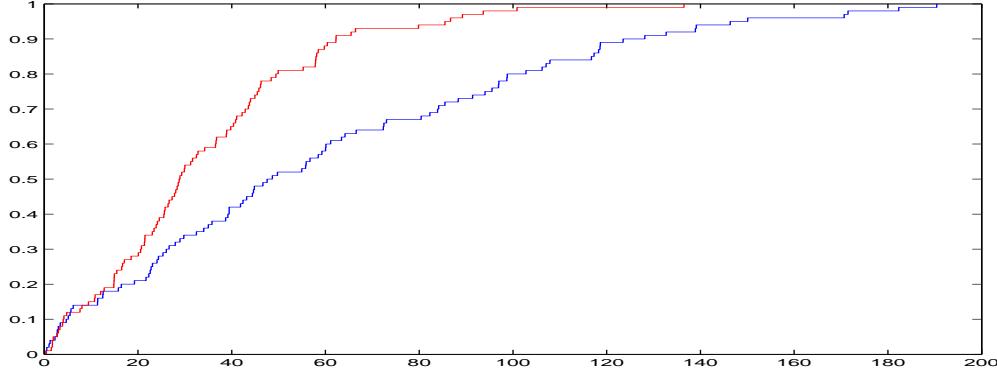


Figure 2.2: Data of Example 2.1. Empirical distribution functions for the old code (right curve) and the new one (left curve). The new outperforms the old, the improvement is significant at the tail of the distribution.

The **sample median** of a data set is defined as follows. Assume there are n data points x_1, \dots, x_n . Sort the points in increasing order and obtain $x_{(1)} \leq \dots \leq x_{(n)}$. If n is odd, the median is $x_{(\frac{n+1}{2})}$, else $\frac{1}{2}(x_{(\frac{n}{2})} + x_{(\frac{n}{2}+1)})$. More generally, the **sample q - quantile** is defined as $\frac{x_{(k')} + x_{(k'')}}{2}$ with $k' = \lfloor qn + (1-q) \rfloor$ and $k'' = \lceil qn + (1-q) \rceil$. $\lfloor x \rfloor$ is the largest integer $\leq x$ and $\lceil x \rceil$ is the smallest integer $\geq x$

Mean and Standard Deviation. The **mean** m of a data set x_1, \dots, x_n is $m = \frac{1}{n} \sum_{i=1}^n x_i$. It gives some information about the center of the distribution. The **standard deviation s of a data set** is defined by $s^2 = \frac{1}{n} \sum_{i=1}^n (x_i - m)^2$ or $s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - m)^2$ (either conventions are used – see Section 2.2 for an explanation). It gives information about the variability. The use of standard deviation is rooted in the belief that data roughly follows a **normal distribution**, also called **gaussian distribution**. It is characterized by a histogram with Bell shape (see wikipedia and Table 3.1 on Page 93); the CDF of the general normal distribution is denoted with N_{μ, σ^2} , where μ is the mean and σ^2 the variance. It is very frequently encountered because of the central limit theorem that says that an average of many things tends to be normal (but there are some exceptions, called heavy tail in Chapter 3). If such a hypothesis is true, and if we had $m \approx \mu$ and $s \approx \sigma$, then with 95% probability, the data sample would lie in the interval $m \pm 1.96s$. This justifies the use of **mean-variance** plots like in Figure 2.3, which use as measure of variability the interval $m \pm 1.96s$. This is also called a **prediction interval** since it predicts a likely range for a future sample (Section 2.4).

EXAMPLE 2.1: COMPARISON OF TWO OPTIONS. An operating system vendor claims that the new version of the database management code significantly improves the performance. We measured the execution times of a series of commonly used programs with both options. The data are displayed in Figure 2.1. The raw displays and histograms show that both options have the same range, but it seems (graphically) that the new system more often provides a smaller execution time. The box plots are more suggestive; they show that the average and the range are about half for the new system.

In Section 2.5 we discuss the differences between these two modes of summarization.

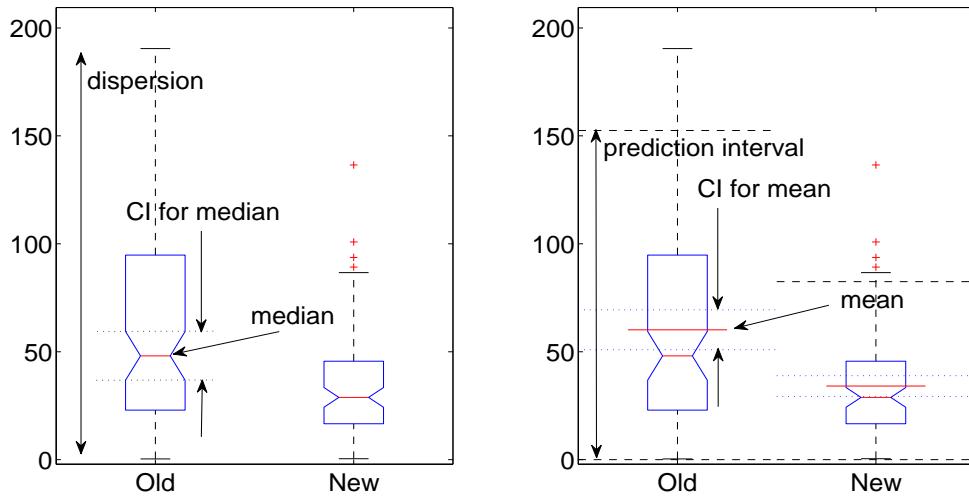


Figure 2.3: Box Plots for the data for Example 2.1. Left: standard box plot commonly used by statisticians showing median (notch) and quartiles (top and bottom of boxes); “dispersion” is an ad-hoc measure, defined here as 1.5 times the inter-quartile distance; the notch width shows the confidence interval for the median. Right: same, overlaid with quantities commonly used in signal processing: mean, confidence interval for the mean ($= \text{mean} \pm 1.96\sigma/\sqrt{n}$, where σ is the standard deviation and n is the number of samples) and prediction interval ($= \text{mean} \pm 1.96\sigma$).

2.1.3 COEFFICIENT OF VARIATION AND LORENZ CURVE GAP

Those are frequently used measures of variation, rescaled to be invariant by change of scale. They apply to a positive data set x_1, \dots, x_n .

COEFFICIENT OF VARIATION. It is defined by

$$\text{CoV} = \frac{s}{m} \quad (2.2)$$

where m is the mean and s the standard deviation, i.e. it is the standard deviation rescaled by the mean. It is also sometimes called **Signal to Noise ratio**. For a data set with n values one always has²

$$0 \leq \text{CoV} \leq \sqrt{n-1} \quad (2.3)$$

where the upper bound is obtained when all x_i have the same value except one of them. The lower bound is reached when all values are equal.

LORENZ CURVE GAP. It is an alternative measure of dispersion, obtained when we replace the standard deviation by the **Mean Absolute Deviation (MAD)**. The MAD is defined by

$$\text{MAD} = \frac{1}{n} \sum_{i=1}^n |x_i - m|$$

²Consider the maximization problem: maximize $\sum_i (x_i - m)^2$ subject to $x_i \geq 0$ and $\sum x_i = mn$. Since $x \mapsto \sum_i (x_i - m)^2$ is convex, the maximum is at an extremal point $x_{i_0} = mn$, $x_i = 0$, $i \neq i_0$.

i.e. we compute the mean distance to the mean, instead of the square root of the mean square distance. Compared to the standard deviation, the MAD is less sensitive to a few very large values. It follows from the Cauchy-Schwarz inequality that it is always less than the standard deviation, i.e.

$$0 \leq \text{MAD} \leq s \quad (2.4)$$

with equality only if x_i is constant, i.e. $x_i = m$ for all i .

If n is large and x_i is iid from a gaussian distribution, then

$$\text{MAD} \approx \sqrt{\frac{2}{\pi}}s \approx 0.8s \quad (2.5)$$

If in contrast, if x_i comes from a heavy tailed distribution with a finite mean m , then $s \rightarrow \infty$ as n gets large, whereas MAD converges to a finite limit.

The *Lorenz Curve Gap* is a rescaled version of MAD, defined by

$$\text{gap} = \frac{\text{MAD}}{2m} \quad (2.6)$$

The reason for the factor 2 is given in the next section. We always have

$$0 \leq \text{gap} \leq 1 - \frac{1}{n} \quad (2.7)$$

thus, contrary to CoV, gap is between 0 and 1. If n is large and x_i is iid from a gaussian distribution, then gap $\approx 0.4\text{CoV}$; if it comes from an exponential distribution, gap ≈ 0.37 and CoV ≈ 1 .

If x_i is iid and comes from a distribution with PDF $f()$, then, for large n , CoV and MAD converge to their theoretical counterparts:

$$\begin{aligned} \text{CoV} &\rightarrow \text{CoV}_{\text{th}} = \frac{\sqrt{\int_0^\infty (x - \mu)^2 f(x) dx}}{\mu} \\ \text{MAD} &\rightarrow \text{gap}_{\text{th}} = \frac{\int_0^\infty |x - \mu| f(x) dx}{2\mu} \end{aligned}$$

with $\mu = \int_0^\infty xf(x)dx$.

If the distribution is gaussian N_{μ, σ^2} then $\text{CoV}_{\text{th}} = \frac{\sigma}{\mu}$ and $\text{gap}_{\text{th}} = \sqrt{\frac{1}{2\pi}}\frac{\sigma}{\mu}$; if it is exponential then $\text{CoV}_{\text{th}} = 1$ and $\text{gap}_{\text{th}} = \frac{1}{e}$.

2.1.4 FAIRNESS INDICES

Often one interprets variability as fairness, and several fairness indices have been proposed. We review here the two most prominent ones. We also show that they are in fact reformulations of variability measures, i.e. they are equivalent to CoV and gap, after proper mapping (so that using these indices may appear superfluous). Like in the previous section, the data set x_i is assumed here to be positive.

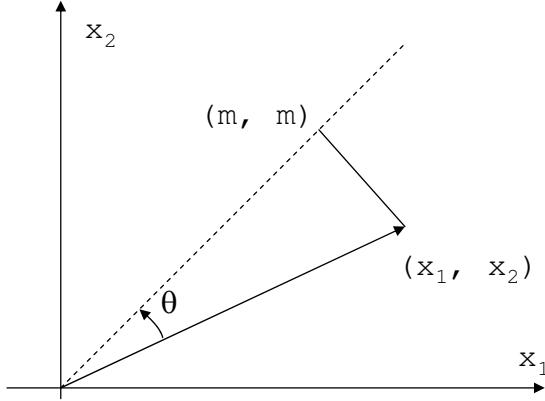


Figure 2.4: Jain's fairness index is $\cos^2 \theta$. x_1, \dots, x_n is the data set and m is the sample mean. The figure is for $n = 2$.

JAIN'S FAIRNESS INDEX (JFI). It is defined as the square of the cosine of the angle between the data set x_i and the hypothetical equal allocation (Figure 2.4). It is given by

$$\text{JFI} = \frac{(\sum_{i=1}^n x_i)^2}{n \sum_{i=1}^n x_i^2} \quad (2.8)$$

A straightforward computation shows that the fairness measure JFI is a decreasing function of the variability measure CoV:

$$\text{JFI} = \frac{1}{1 + \text{CoV}^2} \quad (2.9)$$

so that, by Eq.(2.3), we conclude that JFI ranges from $\frac{1}{n}$ (maximum unfairness) to 1 (all x_i are equal).

LORENZ CURVE The *Lorenz Curve* is defined as follows. A point (p, ℓ) on the curve, with $p, \ell \in [0, 1]$, means that the bottom fraction p of the distribution contributes to a fraction ℓ of the total $\sum_{i=1}^n x_i$.

More precisely, we are given a data set $x_i > 0$, $i = 1 \dots n$. We plot for all $i = 1 \dots n$ the points (p_i, ℓ_i) with

$$\begin{cases} p_i = \frac{i}{n} \\ \ell_i = \frac{\sum_{j=1}^n x_j \mathbf{1}_{\{x_j \leq x_i\}}}{\sum_{j=1}^n x_j} \end{cases} \quad (2.10)$$

See Figure 2.5 for examples. We can make the Lorenz curve a continuous mapping $\ell = L(p)$ by linear interpolation and by setting $L(0) = 0$. The resulting $L()$ is a continuous mapping from $[0, 1]$ onto $[0, 1]$, monotone non decreasing, convex, with $L(0) = 0$ and $L(1) = 1$.

The Lorenz curve $\ell = L(p)$ can be interpreted as a global measure of fairness (or variability). If all x_i 's are equal (maximum fairness) then $L(p) = p$ and $L()$ is the diagonal of the square $[0, 1] \times [0, 1]$ (called the “line of perfect equality”). In the worst case, the Lorenz curve follows the bottom and right edges of the square (called the “line of perfect inequality”) (Figure 2.6). In practice the Lorenz curve is computed by sorting x_i in increasing order ($x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(n)}$) and letting

$$l_i = \frac{x_{(1)} + \dots + x_{(i)}}{nm} \quad (2.11)$$

where m is the sample mean. It follows that $0 \leq l_i \leq \frac{i}{n}$, i.e.

$$0 \leq L(p) \leq p$$

i.e. and the Lorenz curve is always between the lines of perfect equality and perfect inequality.

LOREZ CURVE GAP, AGAIN A measure of fairness is the largest euclidian distance (the gap) from the Lorenz curve to the diagonal, rescaled by its maximum value $\left(\frac{1}{\sqrt{2}}\right)$. It is also equal to the largest vertical distance, $\sup_{u \in [0,1]} (u - L(u))$ (Figure 2.5). The gap can easily be computed by observing that it is reached at index $i_0 = \max\{i : x_{(i)} \leq m\}$, i.e. at a value $p_0 = \frac{i_0}{n}$ such that the bottom fraction p_0 of the data have a value less than the average. Thus

$$\text{gap} = \frac{i_0}{n} - \frac{x_{(1)} + \dots + x_{(i_0)}}{mn} \quad (2.12)$$

We have already introduced the gap in Eq.(2.6), so we need to show that the two definitions are equivalent. This follows from

$$\begin{aligned} \text{MAD} &= \frac{1}{n} \sum_{i=1}^n |x_i - m| = \frac{1}{n} \sum_{i=1}^n |x_{(i)} - m| \\ &= \frac{1}{n} \left(\sum_{i=1}^{i_0} (m - x_{(i)}) + \sum_{i_0+1}^n (x_{(i)} - m) \right) \\ &= \frac{1}{n} \left(i_0 m - \sum_{i=1}^{i_0} x_{(i)} + nm - \sum_{i=1}^{i_0} x_{(i)} - (n - i_0)m \right) \\ &= 2m \text{ gap} \end{aligned}$$

which is the same as Eq.(2.6).

The theoretical Lorenz curve is defined for a probability distribution with cumulative distribution function CDF $F()$ and finite mean μ by

$$L(p) = \frac{1}{\mu} \int_0^p F^{-1}(q) dq \quad (2.13)$$

where F^{-1} is the (right-continuous) pseudo-inverse

$$F^{-1}(q) = \sup\{x : F(x) \leq p\} = \inf\{x : F(x) > q\}$$

If the CDF $F()$ is continuous and increasing, then F^{-1} is the usual function inverse. In this case, the theoretical Lorenz curve gap is then equal to

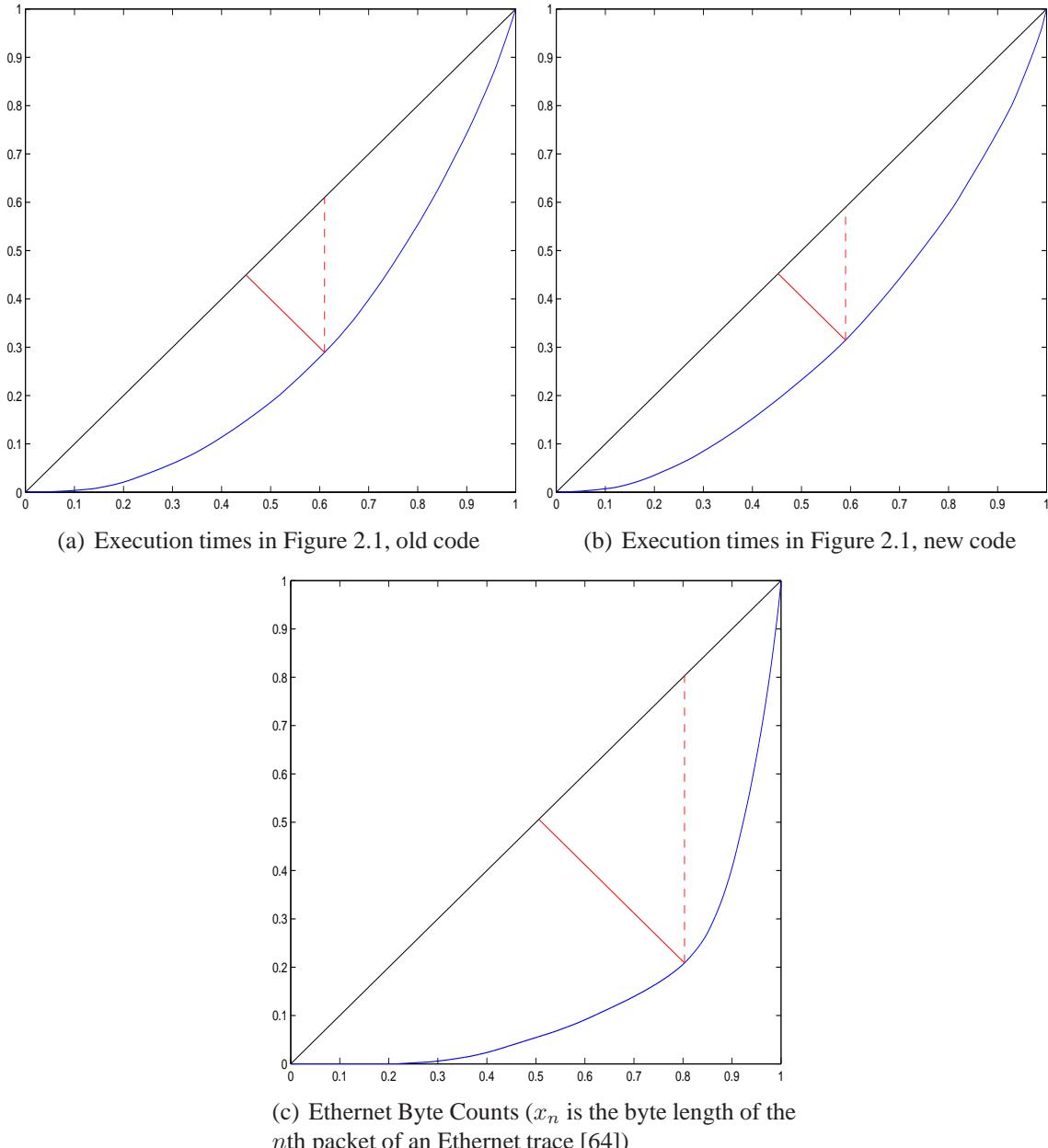
$$\text{gap}_{\text{th}} = p_0 - L(p_0)$$

with $p_0 = F(\mu)$.

The theoretical Lorenz curve is the limit of the Lorenz curve for an iid data sample coming from $F()$, when n is large.

THE GINI COEFFICIENT *Gini coefficient* This is yet another fairness index, very widespread in economy, and, by imitation, in computer and communication systems. Its definition is similar to the Lorenz curve gap, with the mean average deviation replaced by the *Mean Difference*:

$$\text{MD} = \frac{1}{n(n-1)} \sum_{i,j} |x_i - x_j| \quad (2.14)$$



	CoV	JFI	gap	Gini	Gini-approx
Figure 2.1, old code	0.779	0.622	0.321	0.434	0.430
Figure 2.1, new code	0.720	0.658	0.275	0.386	0.375
Ethernet Byte Counts	1.84	0.228	0.594	0.730	0.715

Figure 2.5: Lorenz curves for three data sets, with proportion of users p on x axis and proportion of total sum ℓ on y axis. The diagonal is the line of perfect equality. The maximum distance (plain line) is equal to $\frac{1}{\sqrt{2}}$ times the maximum vertical deviation (dashed line), which is called the Lorenz curve gap. The Gini coefficient is the area between the diagonal and the Lorenz curve, rescaled by its maximum value $\frac{1}{2}$. The table gives the values of Coefficient of Variation, Jain's Fairness Index, Lorenz Curve Gap, Gini coefficient and the Gini coefficient approximation in Eq.(2.17).

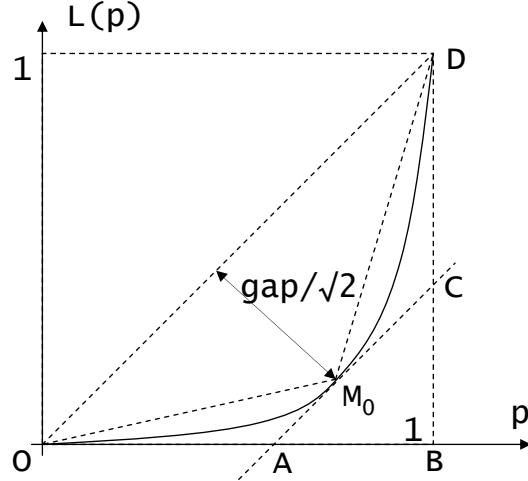


Figure 2.6: Lorenz curve (plain line). The line of perfect equality is OD , of perfect inequality OBD . The Lorenz curve gap is the maximum distance to the line of perfect equality, re-scaled by $\sqrt{2}$. The Gini coefficient is the area between the line of perfect equality and the Lorenz curve, re-scaled by 2.

The Gini coefficient is then defined as

$$\text{Gini} = \frac{MD}{2m} \quad (2.15)$$

where m is the empirical mean of the data set. It can be shown that it is equal to $2 \times$ the area between the line of perfect equality and the Lorenz curve (the rescaling factor 2 makes it lie between 0 and 1). In practice the Gini coefficient can be computed by using Eq.(2.11), which gives

$$\text{Gini} = \frac{2}{mn^2} \sum_{i=1}^n ix_{(i)} - 1 - \frac{1}{n} \quad (2.16)$$

The theoretical Gini coefficient for a probability distribution with CDF $F()$ is defined by

$$\text{Gini}_{\text{th}} = 2 \int_0^1 (q - L(q)) dq = 1 - 2 \int_0^1 L(q) dq$$

where $L()$ is the theoretical Lorenz curve defined in Eq.(2.13).

Since the Lorenz curve is convex, it is straightforward to bound the Gini coefficient by means of the Lorenz curve gap. On Figure 2.6, we see that the area between the Lorenz curve and the diagonal is lower bounded by the triangle OM_0D and upper bounded by the trapeze $OACD$. It follows from this and Eq.(2.16) that

$$\begin{aligned} 0 \leq \text{gap} &\leq \text{Gini} \leq 1 - \frac{1}{n} \\ \text{Gini} &\leq \text{gap} (2 - \text{gap}) \end{aligned}$$

where the lower bound 0 is reached at maximum fairness.

It follows that one can also approximate Gini by the arithmetic mean of the lower and upper bounds:

$$\text{Gini} \approx \text{gap} (1.5 - 0.5 \text{gap}) \quad (2.17)$$

	Jain's Fairness Index (JFI)	Lorenz Curve Gap (gap)	Gini Coefficient (Gini)
Definition	$\frac{1}{1 + \text{CoV}^2}$ Eq.(2.2), Eq.(2.8)	$\frac{\text{MAD}}{2m}$ Eq.(2.6)	$\frac{\text{MD}}{2m}$ Eq.(2.15)
Bounds	$\frac{1}{n} \leq \text{JFI} \leq 1$ (MF)	(MF) $0 \leq \text{gap} \leq 1 - \frac{1}{n}$	(MF) $0 \leq \text{Gini} \leq 1 - \frac{1}{n}$
Relations	$\frac{1}{1+4\text{gap}^2} \leq \text{JFI}$ Equality only at MF		$\text{gap} \leq \text{Gini} \leq \text{gap}(2 - \text{gap})$ $\text{Gini} \approx \text{gap}(1.5 - 0.5\text{gap})$
$\text{Exp}(\lambda), \lambda > 0$	0.5	$\frac{1}{e} \approx 0.368$	0.5
Unif(a, b) $0 \leq a < b$	$\frac{1}{1 + \frac{(b-a)^2}{3(b+a)^2}}$	$\frac{b-a}{4(a+b)}$	$\frac{b-a}{3(a+b)}$
Pareto (p, x_0) $x_0 > 0, p > 1$	$\frac{p(p-2)}{(p-1)^2}$ for $p > 2$	$\frac{1}{p} \left(1 - \frac{1}{p}\right)^{p-1}$	$\frac{1}{2p-1}$

Table 2.1: Relationships between different fairness indices of a data set with n samples and empirical mean m (MF = value when fairness is maximum, i.e all data points are equal).

SUMMARY Since there are so many different variability and fairness indices, we give here a summary with some recommendations.

First, since the Gini coefficient can be essentially predicted from the Lorenz curve gap, we do not use it further in this book. However, it may be useful to know the relationship between the two since you may find that it is used in some performance evaluation results.

Second, Jain's fairness index and the Lorenz curve gap are fundamentally different and cannot be mapped to each other. The former is essentially the same as the standard deviation or the coefficient of variation. If the data comes from a heavy tailed distribution, the theoretical coefficient of variation is infinite, and $\text{CoV} \rightarrow \infty$ as the number of data points gets large. Comparing different CoVs in such a case does not bring much information. In contrast, the Lorenz curve gap continues to be defined, as long as the distribution has a finite mean. It should be preferred, if one has a choice.

We recall the main inequalities and bounds in Table 2.1 on Page 31. See also Figure 2.5 for some examples.

2.2 CONFIDENCE INTERVALS

2.2.1 WHAT IS A CONFIDENCE INTERVAL ?

When we display a number such as the median or the mean of a series of performance results, it is important to quantify their accuracy (this is part of the scientific method, Chapter 1). **Confidence intervals** quantify the uncertainty about a summarized data that is due to the randomness of the measurements.

EXAMPLE 2.2: COMPARISON OF TWO OPTIONS, CONTINUED. We wish to quantify the improvement due to the new system. To this end, we measure the reduction in run time for the same sequence of tasks as on Figure 2.1 (both data sets on Figure 2.1 come from the same transaction sequences

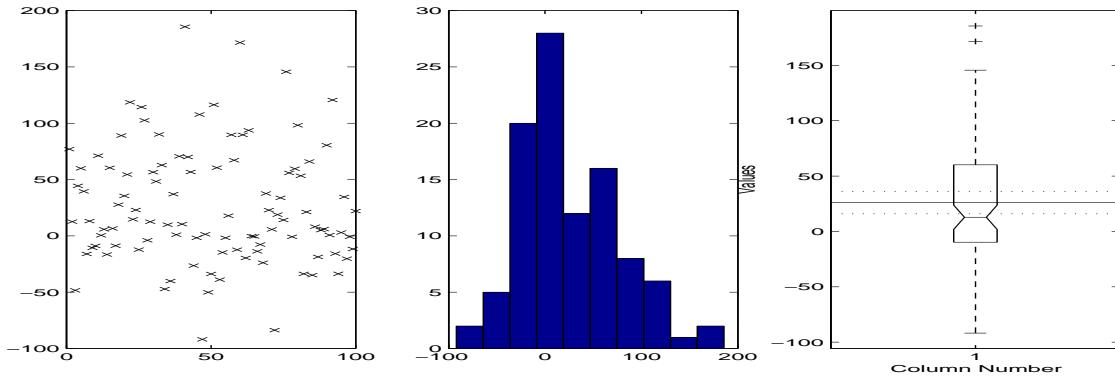


Figure 2.7: Data for Example 2.2: reduction in run time (in ms). Right: Box plot with mean and confidence interval for mean.

– statisticians say that this is a [paired experiment](#)). The differences are displayed in Figure 2.7. The last panel shows confidence intervals for the mean (horizontal lines) and for the median (notches in Box plot). For example, the mean of the reduction in run time is 26.1 ± 10.2 . The uncertainty margin is called the confidence interval for the mean. It is obtained by the method explained in this section. Here, the mean reduction is non negligible, but the uncertainty about it is large.

There is a confidence interval for every summarized quantity: median, mean, quartile, standard deviation, fairness index, etc. In the rest of this section, we explain *how* to compute confidence intervals.

2.2.2 CONFIDENCE INTERVAL FOR MEDIAN AND OTHER QUANTILES

We start with the median and other quantiles, as it is both simplest and most robust; this section also serves as an illustration of the general method for computing confidence intervals.

The main idea (which underlies all classical statistics formulae) is to imagine that the data we have measured was in fact generated by a simulator, whose program is unknown to us. More precisely, we are given some data x_1, \dots, x_n ; we imagine that there is a well defined probability distribution with CDF $F()$ from which the data is sampled, i.e. we have received one sample from a sequence of independent and identically distributed ([iid](#)) random variables X_1, \dots, X_n , each with common CDF $F()$. The assumption that the random variables are iid is capital; if it does not hold, the confidence intervals are wrong. We defer to Section 2.3 a discussion of when we may or may not make this assumption. For now we assume it holds.

The distribution $F()$ is non-random but is unknown to us. It has a well defined median m , defined by : for every i , $\mathbb{P}(X_i \leq m) = 0.5$. We can never know m exactly, but we [estimate](#) it by $\hat{m}(x_1, \dots, x_n)$, equal to the sample median defined in Section 2.1. Note that the value of the estimated median depends on the data, so it is random: for different measurements, we obtain different estimated medians. The goal of a confidence interval is to bound this uncertainty. It is defined relative to a [confidence level](#) γ ; typically $\gamma = 0.95$ or 0.99 :

DEFINITION 2.1. A [confidence interval](#) at level γ for the fixed but unknown parameter m is an

interval $(u(X_1, \dots, X_n), v(X_1, \dots, X_n))$ such that

$$\mathbb{P}(u(X_1, \dots, X_n) < m < v(X_1, \dots, X_n)) \geq \gamma \quad (2.18)$$

In other words, the interval is constructed from the data, such that with at least 95% probability (for $\gamma = 0.95$) the true value of m falls in it. Note that **it is the confidence interval that is random, not the unknown parameter m .**

A confidence interval for the median or any other quantile is very simple to compute, as the next theorem shows.

THEOREM 2.1 (Confidence Interval for Median and Other Quantiles). *Let X_1, \dots, X_n be n iid random variables, with a common CDF $F()$. Assume that $F()$ has a density, and for $0 < p < 1$ let m_p be a p -quantile of $F()$, i.e. $F(m_p) = p$.*

*Let $X_{(1)} \leq X_{(2)} \leq \dots \leq X_{(n)}$ be the **order statistic**, i.e. the set of values of X_i sorted in increasing order. Let $B_{n,p}$ be the CDF of the binomial distribution with n repetitions and probability of success p . A confidence interval for m_p at level γ is*

$$[X_{(j)}, X_{(k)}]$$

where j and k satisfy

$$B_{n,p}(k-1) - B_{n,p}(j-1) \geq \gamma$$

See the tables in Appendix A on Page 311 for practical values. For large n , we can use the approximation

$$\begin{aligned} j &\approx \lfloor np - \eta \sqrt{np(1-p)} \rfloor \\ k &\approx \lceil np + \eta \sqrt{np(1-p)} \rceil + 1 \end{aligned}$$

where η is defined by $N_{0,1}(\eta) = \frac{1+\gamma}{2}$ (e.g. $\eta = 1.96$ for $\gamma = 0.95$).

The **Binomial distribution** $B_{n,p}$, with n repetitions and probability of success p , is the distribution of $Z = \sum_{i=1}^n Z_i$ where Z_i are iid random variables such that $Z_i = 0$ or 1 and $\mathbb{P}(Z_i = 1) = p$, i.e. it is the distribution of the number of successes in an experiment with n trials and individual success probability p . (The random variables Z_i are called **Bernoulli** random variables. $N_{0,1}$ is the CDF of the gaussian distribution with mean 0 and variance 1.)

For $n = 10$, the theorem and the table in Section A say that a 95%-confidence interval for the median (estimated as $\frac{X_{(5)}+X_{(6)}}{2}$) is $[X_{(2)}, X_{(9)}]$. In other words, we obtain a confidence interval for the median of 10 results by removing the smallest and the largest. Could it be simpler?

Note that, for small values of n , no confidence interval is possible at the levels 0.95 or 0.99. This is due to the probability that the true quantile is outside any of the observed data still being large.

For large n , the binomial distribution can be approximated by a gaussian distribution, which explains the approximation in the theorem.

The assumption that the distribution has a density (also called PDF, probability density function) is for simplicity of exposition. If $F()$ does not have a density (e.g. because the numbers X_i are integers) the theorem hold with the modification that the confidence interval is $[X_{(j)}, X_{(k)}]$ (instead of $[X_{(j)}, X_{(k)}]$).

2.2.3 CONFIDENCE INTERVAL FOR THE MEAN

Here too there is a widely used result, given in the next theorem. The proof is standard and can be found in probability textbooks [38, 76].

THEOREM 2.2. *Let X_1, \dots, X_n be n iid random variables, the common distribution of which is assumed to have well defined mean μ and a variance σ^2 . Let $\hat{\mu}_n$ and s_n^2 by*

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n X_i \quad (2.19)$$

$$s_n^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \hat{\mu}_n)^2 \quad (2.20)$$

The distribution of $\sqrt{n} \frac{\hat{\mu}_n - \mu}{s_n}$ converges to the normal distribution $N_{0,1}$ when $n \rightarrow +\infty$. An approximate confidence interval for the mean at level γ is

$$\hat{\mu}_n \pm \eta \frac{s_n}{\sqrt{n}} \quad (2.21)$$

where η is the $\frac{1+\gamma}{2}$ quantile of the normal distribution $N_{0,1}$, i.e $N_{0,1}(\eta) = \frac{1+\gamma}{2}$. For example, $\eta = 1.96$ for $\gamma = 0.95$ and $\eta = 2.58$ for $\gamma = 0.99$.

Note that the amplitudes of the confidence interval decreases like $\frac{1}{\sqrt{n}}$.

Also note however that some caution may be required when using the theorem, as it makes 3 assumptions:

1. the data comes from an iid sequence
2. the common distribution has a finite variance
3. the number of samples is large

each of these assumptions is worth screening, as there are realistic cases where they do not hold. Assumption 1 is the same as for all confidence intervals in this chapter, and is discussed in Section 2.3. Assumption 2 is true unless the distribution is heavy tailed, see Section 3.5. Assumption 3 is usually true even for small values of n , and can be verified using the method in Section 2.5.1.

NORMAL IID CASE

The following theorem is a slight variant of Theorem 2.2. It applies only to the cases where we know a priori that the distribution of the measured data follows a common gaussian distribution N_{μ, σ^2} , with μ and σ fixed but unknown. It gives practically the same result as Theorem 2.2 for the confidence interval for the mean; in addition it gives a confidence interval for the standard deviation. This result is often used in practice, perhaps not rightfully, as the gaussian assumptions are not always satisfied.

THEOREM 2.3. *Let X_1, \dots, X_n be a sequence of iid random variables with common distribution*

N_{μ, σ^2} . Let

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n X_i \quad (2.22)$$

$$\hat{\sigma}_n^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \hat{\mu}_n)^2 \quad (2.23)$$

Then

- The distribution of $\sqrt{n} \frac{\hat{\mu}_n - \mu}{\hat{\sigma}_n}$ is Student's t_{n-1} ; a confidence interval for the mean at level γ is

$$\hat{\mu}_n \pm \eta \frac{\hat{\sigma}_n}{\sqrt{n}} \quad (2.24)$$

where η is the $(\frac{1+\gamma}{2})$ quantile of the student distribution t_{n-1} .

- The distribution of $(n-1) \frac{\hat{\sigma}_n^2}{\sigma^2}$ is χ^2_{n-1} .

A confidence interval at level γ for the standard deviation is

$$[\hat{\sigma}_n \sqrt{\frac{n-1}{\xi}}, \hat{\sigma}_n \sqrt{\frac{n-1}{\zeta}}] \quad (2.25)$$

where ζ and ξ are quantiles of χ^2_{n-1} : $\chi^2_{n-1}(\zeta) = \frac{1+\gamma}{2}$ and $\chi^2_{n-1}(\xi) = \frac{1-\gamma}{2}$.

The distributions χ^2 and t_n are defined as follows. **Chi-Square** (χ^2_n) is the distribution of the sum of the squares of n independent random variables with distribution $N_{0,1}$ (its expectation is n and its variance $2n$). **Student** (t_n) is the distribution of

$$Z = \frac{X}{\sqrt{Y/n}}$$

where $X \sim N_{0,1}$, $Y \sim \chi^2_n$ and X and Y are independent.

Unlike in Theorem 2.2, the magic numbers η, ζ, ξ depend on the confidence level γ but also on the sample size n . For instance, with $n = 100$ and confidence level 0.95, we have $\eta = 1.98$, $\zeta = 73.4$, and $\xi = 128.4$. This gives the confidence intervals for mean and standard deviation: $[\hat{\mu}_n - 0.198\hat{\sigma}_n, \hat{\mu}_n + 0.198\hat{\sigma}_n]$ and $[0.86\hat{\sigma}_n, 1.14\hat{\sigma}_n]$.

QUESTION 2.2.1. Does the confidence interval for the mean in Theorem 2.3 depend on the estimator of the variance? Conversely?³

We can compare the confidence interval for the mean given by this theorem in Eq.(2.24) and by Theorem 2.2 in Eq.(2.21). The latter is only approximately true, so we may expect some small difference, vanishing with n . Indeed, the two formulas differ by two terms.

1. The estimators of the variance $\hat{\sigma}_n^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \hat{\mu}_n)^2$ and $s_n^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \hat{\mu}_n)^2$ differ by the factor $\frac{1}{n}$ versus $\frac{1}{n-1}$. The factor $\frac{1}{n-1}$ may seem unnatural, but it is required for Theorem 2.3 to hold exactly. The factor $\frac{1}{n}$ appears naturally from the theory of maximum likelihood estimation (Section B.1). In practice, it is not required to have an extreme accuracy for the estimator of σ^2 (since it is a second order parameter); thus using $\frac{1}{n-1}$ or $\frac{1}{n}$ makes little difference. Both $\hat{\sigma}_n$ and s_n are called **sample standard deviation**.

³Yes; No

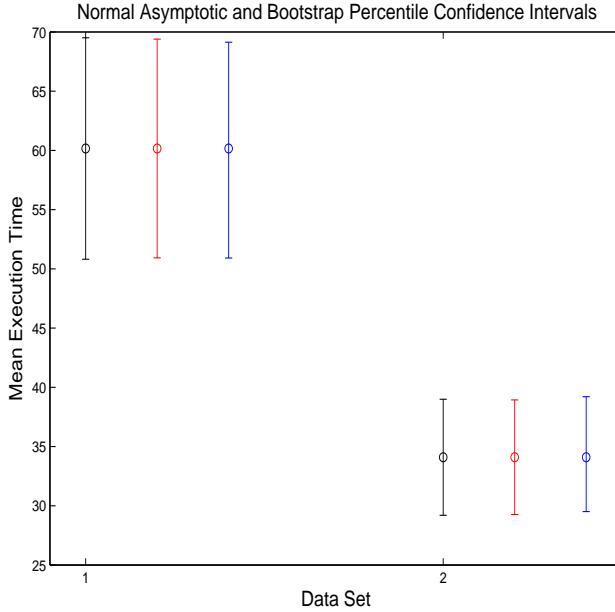


Figure 2.8: Confidence intervals for both compiler options of Example 2.1 computed with three different methods: assuming data would be normal (Theorem 2.3) (left); the general method in and with the bootstrap method (right).

2. η in Eq.(2.24) is defined by the student distribution, and by the normal distribution in Eq.(2.21). For large n , the student distribution is close to normal; for example, with $\gamma = 0.95$ and $n = 100$, we have $\eta = 1.98$ in Eq.(2.24) and $\eta = 1.96$ in Eq.(2.21).

See Figure 2.8 for an illustration.

2.2.4 CONFIDENCE INTERVALS FOR FAIRNESS INDICES AND THE BOOTSTRAP

There is no analytical general method, even when n is large (but see [102] for some special cases, if the data is i.i.d normal or log-normal). Instead, we use a generic, computational method, called the bootstrap. It is general and can be used for any estimator, not just to fairness indices. It applies to all cases where data is iid.

THE BOOTSTRAP Consider a sample $\vec{x} = (x_1, \dots, x_n)$, which we assume to be a realization of an iid sequence X_1, \dots, X_n . We know nothing about the common distribution $F()$ of the X_i s. We are interested in some quantity $t(\vec{x})$ derived from the data, for which we want to find a confidence interval (in this context $t(\vec{x})$ is called a **statistic**). For example, if the statistic of interest is the Lorenz curve gap, then by Section 2.1.3:

$$t(\vec{x}) = \frac{1}{2 \sum_{i=1}^n x_i} \sum_{j=1}^n \left| x_j - \frac{1}{n} \sum_{i=1}^n x_i \right|$$

The **bootstrap method** uses the sample $\vec{x} = (x_1, \dots, x_n)$ as an approximation of the true, unknown distribution. It is justified by the Glivenko-Cantelli theorem which says that the ECDF converges

with probability 1 to the true CDF $F()$ when n gets large.

The method is described formally in Algorithm 1. The loop creates R **bootstrap replicates** \vec{X}^r ,

Algorithm 1 The Bootstrap, for computation of confidence interval at level γ for the statistic $t(\vec{x})$. The data set $\vec{x} = (x_1, \dots, x_n)$ is assumed to be a sample from an iid sequence, with unknown distribution. r_0 is the algorithm's accuracy parameter.

```

1:  $R = \lceil 2 r_0 / (1 - \gamma) \rceil - 1$                                 ▷ For example  $r_0 = 25, \gamma = 0.95, R = 999$ 
2: for  $r = 1 : R$  do
3:   draw  $n$  numbers with replacement from the list  $(x_1, \dots, x_n)$  and call them  $X_1^r, \dots, X_n^r$ 
4:   let  $T^r = t(\vec{X}^r)$ 
5: end for
6:  $(T_{(1)}, \dots, T_{(R)}) = \text{sort}(T^1, \dots, T^R)$ 
7: Prediction interval is  $[T_{(r_0)} ; T_{(R+1-r_0)}]$ 
```

$r = 1, \dots, R$. Each bootstrap replicate $\vec{X}^r = (X_1^r, \dots, X_n^r)$ is a random vector of size n , like the original data. All X_i^r are independent copies of the same random variable, obtained by drawing from the list (x_1, \dots, x_n) **with replacement**. For example, if all x_k are distinct, we have $\mathbb{P}(X_i^r = x_k) = \frac{1}{n}, k = 1, \dots, n$.

For each r , line 4 computes the value of the statistic obtained with the r th “replayed” experiment. The confidence interval in line 7 is the **percentile bootstrap estimate** at level γ . It is based on the order statistic $(T_{(r)})_{r=1,\dots,R}$ of $(T^r)_{r=1,\dots,R}$.

The value of R in line 1 needs to be chosen such that there are sufficiently many points outside the interval, and depends on the confidence level. A good value is $R = \frac{50}{1-\gamma} - 1$. For example, with $\gamma = 0.95$, take $R = 999$ and the confidence interval in line 7 is $[T_{(25)}; T_{(975)}]$.

EXAMPLE 2.3: CONFIDENCE INTERVALS FOR FAIRNESS INDICES. The confidence intervals for the left two cases on Figure 2.5 were obtained with the Bootstrap, with a confidence level of 0.99, i.e. with $R = 4999$ bootstrap replicates (left and right: confidence interval; center: value of index computed in Figure 2.5).

	Jain's Fairness Index			Lorenz Curve Gap		
Old Code	0.5385	0.6223	0.7057	0.2631	0.3209	0.3809
New Code	0.5673	0.6584	0.7530	0.2222	0.2754	0.3311

For the third example, the bootstrap cannot be applied directly, as the data set is not iid and the bootstrap requires i.i.d data. Subsampling does not work as the data set is long range dependent. A possible method is to fit a long range dependent model, such as fractional arima, then apply the bootstrap to the residuals.

The bootstrap may be used for any metric, not just for fairness indices. Figure 2.8 gives a comparison of confidence intervals *for the mean* obtained with the bootstrap and with the classical methods (here $t(\vec{x}) = \frac{1}{n} \sum_{i=1}^n x_i$).

In general, the percentile estimate is an approximation that tends to be slightly too small. For a theoretical analysis of the bootstrap method, and other applications, see [33].

2.2.5 CONFIDENCE INTERVAL FOR SUCCESS PROBABILITY

This is the frequent case where we do n independent experiments and are interested in a binary outcome (success or failure). Assume we observe z successes (with $0 \leq z \leq n$). We would like to find a confidence interval for the probability p of success, in particular when p is small.

Mathematically, we can describe the situation as follows. We have a sequence X_1, \dots, X_n of independent Bernoulli random variables such that $\mathbb{P}(X_k = 0) = 1 - p$ and $\mathbb{P}(X_k = 1) = p$, and we observe $Z = \sum_{i=1}^n X_i$. The number n of experiments is known, but not the success probability p , which we want to estimate. A natural estimator of p is $\frac{1}{n} \sum_{k=1}^n X_i$, i.e. the mean of the outcomes (this is the maximum likelihood estimator, see Section B.1). Therefore, we can apply the method for confidence intervals for the mean in Theorem 2.2; however, this method is valid only asymptotically, and does not work when z is very small compared to n . A frequent case of interest is when we observe no success ($z = 0$) out of n experiments; here, Theorem 2.2 gives $[0; 0]$ as confidence interval for p , which is not correct. We can use instead the following result.

THEOREM 2.4. [43, p. 110] Assume we observe z successes out of n independent experiments. A confidence interval at level γ for the success probability p is $[L(z); U(z)]$ with

$$\begin{cases} L(0) = 0 \\ L(z) = \phi_{n,z-1}\left(\frac{1+\gamma}{2}\right), \quad z = 1, \dots, n \\ U(z) = 1 - L(n-z) \end{cases} \quad (2.26)$$

where $\phi_{n,z}(\alpha)$ is defined for $n = 2, 3, \dots$, $z \in \{0, 1, \dots, n\}$ and $\alpha \in (0; 1)$ by

$$\begin{cases} \phi_{n,z}(\alpha) = \frac{n_1 f}{n_2 + n_1 f} \\ n_1 = 2(z+1), \quad n_2 = 2(n-z), \quad 1 - \alpha = F_{n_1, n_2}(f) \end{cases} \quad (2.27)$$

($F_{n_1, n_2}()$ is the CDF of the Fisher distribution with n_1, n_2 degrees of freedom). In particular, the confidence interval for p when we observe $z = 0$ successes is $[0; p_0(n)]$ with

$$p_0(n) = 1 - \left(\frac{1-\gamma}{2}\right)^{\frac{1}{n}} = \frac{1}{n} \log\left(\frac{2}{1-\gamma}\right) + o\left(\frac{1}{n}\right) \text{ for large } n \quad (2.28)$$

Whenever $z \geq 6$ and $n - z \geq 6$, the normal approximation

$$\begin{cases} L(z) \approx \frac{z}{n} - \frac{\eta}{n} \sqrt{z\left(1 - \frac{z}{n}\right)} \\ U(z) \approx \frac{z}{n} + \frac{\eta}{n} \sqrt{z\left(1 - \frac{z}{n}\right)} \end{cases} \quad (2.29)$$

can be used instead, with $N_{0,1}(\eta) = \frac{1+\gamma}{2}$.

The confidence interval in the theorem is not the best one, but it is perhaps the simplest. It is based on a symmetric coverage interval, i.e. the probability of being above (or below) is $< \frac{1-\gamma}{2}$ and it is the smallest interval with this property. Other, non symmetric intervals can be derived and are slightly smaller [12].

Note that the function $\phi_{n,z}()$ is the reverse mapping of $p \mapsto B_{n,p}(z)$ where $B_{n,p}()$ is the CDF of the binomial distribution (this explains Eq.(2.28)). Eq.(2.27) is used in numerical implementations [43].

For $\gamma = 0.95$, Eq.(2.28) gives $p_0(n) \approx \frac{3.689}{n}$ and this is accurate with less than 10% relative error for $n \geq 20$ already.

The confidence interval in Eq.(2.29) is obtained by application of the asymptotic confidence interval for the mean; indeed, a direct application of Theorem 2.2 gives $\hat{\mu}_n = \frac{z}{n}$ and $s_n^2 = \frac{z(n-z)}{n}$.

EXAMPLE 2.4: SENSOR LOSS RATIO. We measure environmental data with a sensor network. There is reliable error detection, i.e. there is a coding system which declares whether a measurement is correct or not. In a calibration experiment with 10 independent replications, the system declares that all measurements are correct. What can we say about the probability p of finding an incorrect measurement ?

Apply Eq.(2.28): we can say, with 95% confidence, that $p \leq 30.8\%$.

Later, in field experiments, we find that 32 out of 145 readings are declared incorrect. Assuming the measurements are independent, what can we say about p ?

Apply Eq.(2.29) with $z = 32$, $n = 145$: with 95% confidence we can say that $L \leq p \leq U$ with

$$\begin{cases} L \approx \frac{z}{n} - \frac{1.96}{n} \sqrt{z(1-\frac{z}{n})} = 15.3\% \\ U \approx \frac{z}{n} + \frac{1.96}{n} \sqrt{z(1-\frac{z}{n})} = 28.8\% \end{cases}$$

Instead of the normal approximation in Eq.(2.29), we could have used the exact formula in Eq.(2.26), which would give $L = 15.6\%$, $U = 29.7\%$.

Theorem 2.4 is frequently used in conjunction with Monte Carlo estimation of the p -value of a test, see Example 6.8 on Page 175.

2.3 THE INDEPENDENCE ASSUMPTION

All results in the previous and the next section assume the data is a sample of a sequence of independent and identically distributed (iid) random variables. We discuss here in detail the meaning of this assumption (in Section 2.4.3 we also discuss the gaussian assumption, required by Theorems 2.2 and 2.3).

2.3.1 WHAT DOES IID MEAN ?

Iid-ness is a property of a stochastic model, not of the data. When we say, by an abuse of language, that the collected data set is iid, we mean that we can do as if the collected data x_1, \dots, x_n is a sample (i.e. a simulation output) for a sequence of random variables X_1, \dots, X_n , where X_1, \dots, X_n are independent and all have the same (usually unknown) distribution with CDF $F()$.

To generate such a sample, we draw a random number from the distribution $F()$, using a random number generator (see Section 6.6). Independence means that the random numbers generated at every step i are discarded and not re-used in the future steps $i+1, \dots$. Another way to think of independence is with conditional probabilities: for any set of real numbers A

$$\mathbb{P}(X_i \in A \mid X_1 = x_1, \dots, X_{i-1} = x_{i-1}) = \mathbb{P}(X_i \in A) \quad (2.30)$$

i.e. if we know the distribution $F(x)$, observing X_1, \dots, X_{i-1} does not give more information about X_i .

Note the importance of the “if” statement in the last sentence: remove it and the sentence is no longer true. To understand why, consider a sample x_1, \dots, x_n for which we assume to know that it is generated from a sequence of iid random variables X_1, \dots, X_n with normal distribution but with unknown parameter (μ, σ^2) . If we observe for example that the average of x_1, \dots, x_{n-1} is 100 and all values are between 0 and 200, then we can think that it is very likely that x_n is also in the interval [0, 200] and that it is unlikely that x_n exceeds 1000. Though the sequence is iid, we did gain information about the next element of the sequence having observed the past. There is no contradiction: if we know that the parameters of the random generator are $\mu = 100$ and $\sigma^2 = 10$ then observing x_1, \dots, x_{n-1} gives us no information about x_n .

2.3.2 HOW DO I KNOW IN PRACTICE IF THE IID ASSUMPTION IS VALID ?

If your performance data comes from a *designed experiment*, i.e. a set of simulation or tests that is entirely under your control, then it is up to you to design things in such a way that the collected data are iid. This is done as follows.

Every experiment has a number of factors, i.e., parameters that are likely to influence the outcome. Most of the factors are not really interesting, but you have to account for them in order to avoid hidden factor errors (see Section 1.2 for details). The experiment generates iid data if the values of the factors are chosen in an iid way, i.e., according to a random procedure that is the same for every measured point, and is memoriless. Consider Example 2.1, where the run time for a number of transactions was measured. One factor is the choice of the transaction. The data is made iid if, for every measurement, we choose one transaction **randomly with replacement** in a list of transactions.

A special case of designed experiment is simulation. Here, the method is to generate **replications** without resetting the random number generator, as explained in Section 6.3.

Else (i.e. your data does not come from a designed experiment but from measurements on a running system) there is little chance that the complete sequence of measured data is iid. A simple fix is to **randomize the measurements**, in such a way that from one measurement point to the other there is little dependence. For example, assume you are measuring the response time of an operational web server by data mining the log file. The response time to consecutive requests is highly correlated at the time scale of the minute (due to protocols like TCP); one common solution is to choose requests at random, for example by selecting one request in average every two minutes.

If there is some doubt, the following methods can be used to verify iid-ness:

1. (Autocorrelation Plot): If the data appears to be stationary (no trend, no seasonal component), then we can plot the sample autocorrelation coefficients, which are an estimate of the true autocorrelation coefficients ρ_k (defined on Page 143). If the data is iid, then $\rho_k = 0$ for $k \geq 1$, and the sample autocorrelation coefficients fall within the values $\pm 1.96/\sqrt{n}$ (where n is the sample size) with 95% probability. An autocorrelation plot displays these bounds as well. A visual inspection can determine if this assumption is valid. For example, on Figure 2.9 we see that there is some autocorrelation in the first six diagrams but not in the last two. If visual inspection is not possible, a formal test can be used (the Ljung-Box test, Section 5.5.1). If the data is iid, any point transformation of the data (such as the Box Cox

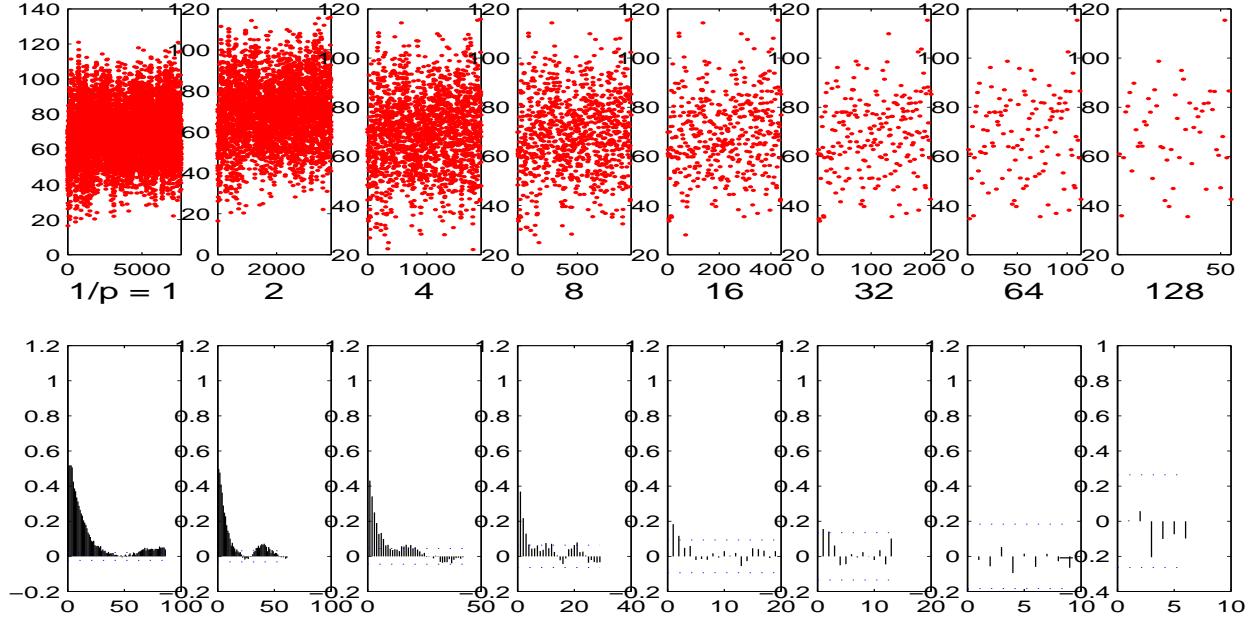


Figure 2.9: Execution times for $n = 7632$ requests (top left) and autocorrelation function (bottom left), and for the data sub-sampled with probability $p = 1/2$ to $1/2^7 = 1/128$. The data appears stationary and roughly normal so the auto-correlation function can be used to test independence. The original data is positively correlated, but the sub-sampled data loses correlation when the sampling probability is $p = 1/64$. The turning point test for the subsampled data with $p = 1/64$ has a p -value of 0.52648, thus at confidence level 0.95 we accept the null hypothesis, namely, the data is iid. The sub-sampled data has 116 points, and the confidence interval obtained from this for the median of the sub-sampled data is [66.7, 75.2] (using Theorem 2.1). Compare with the confidence interval that would be obtained if we would (wrongly) assume the data to be iid : [69.0, 69.8]. The iid assumption underestimates the confidence interval because the data is positively correlated.

transformation for any exponent s , Section 2.4.3) should appear to be non correlated as well.

2. (*Lag-Plot*): We can also plot the value of the data at time t versus at time $t + h$, for different values of h (lag plots). If the data is iid, the lag plots do not show any trend. On Figure 2.10 we see that there is a negative trend at lag 1.
3. (Turning Point Test): A test provides an automated answer, but is sometimes less sure than a visual inspection. A test usually has a null hypothesis and returns a so called “ p -value” (see Chapter 4 for an explanation). If the p -value is smaller than $\alpha = 1 - \gamma$, then the test rejects the null hypothesis at the confidence level γ . See Section 4.5.2 for details.

2.3.3 WHAT HAPPENS IF THE IID ASSUMPTION DOES NOT HOLD ?

If we compute a confidence interval (using a method that assumes iid data) whereas the iid assumption does not hold, then we introduce some bias. Data arising from high resolution measurements are frequently positively correlated. In such cases, the confidence interval is too small: there is not as much information in the data as one would have if they would be iid (since the data tends to repeat itself); see Figure 2.9 for an example.

It may still be possible to obtain confidence intervals when the data does not appear to be iid. Two possible methods are:

Sub-sampling This means select a fraction p of the measured data, and verify that the iid assumption can be made for the selected data. The hope is that correlation disappears between data samples that are far apart.

A simple way would be to keep every pn data sample, where n is the total number of points, but this is not recommended as such a strict periodic sampling may introduce unwanted anomalies (called aliasing). A better method is to decide independently for each data point, with probability p , whether it is sub-sampled or not.

For example, on Figure 2.9, sub-sampling works for $p \leq 1/64$; the confidence interval for the median is much larger than if we would (wrongly) assume the original data to be iid.

Sub-sampling is very simple and efficient. It does not always work, though: it does not work if the data set is small, nor for some large data sets, which remain correlated after repeated sub-sampling (such data sets are called long range dependent).

Modelling is more complex but applies when sub sampling does not. It consists in fitting a parametric model appropriate to the type of data, and computing confidence intervals for the model parameters (for example using Section B.1). We illustrate the method on the next example.

EXAMPLE 2.5: JOE'S BALANCE DATA. Joe's shop sells online access to visitors who download electronic content on their smartphones. At the end of day $t - 1$, Joe's employee counts the amount of cash c_{t-1} present in the cash register and puts it into the safe. In the morning of day t , the cash amount c_{t-1} is returned to the cash register. The total amount of service sold (according to bookkeeping data) during day t is s_t . During the day, some amount of money b_t is sent to the bank. At the end of day t , we should have $c_t = c_{t-1} + s_t - b_t$. However, there are always small errors in counting the coins, in bookkeeping and in returning change. Joe computes the balance $Y_t = c_t - c_{t-1} - s_t + b_t$ and would like to know whether there is a systematic source of errors (i.e. Joe's employee is losing money, maybe because he is not honest, or because some customers are not paying for what they take). The data for Y_t is shown on Figure 2.10. The sample mean is $\mu = -13.95$, which is negative. However, we need a confidence interval for μ before risking any conclusion.

If we would assume that the errors Y_t are iid, then a confidence interval would be given by Theorem 2.2 and we find approximately $[-43, 15]$. Thus, with the iid model, we cannot conclude that there is a fraud.

However, the iid assumption is not valid, as Figure 2.10 shows (there is a strong correlation at lag 1; this is confirmed by the lag plot). We use a modelling approach. A similar problem is discussed in [18, Example 3.2.8], with oil rather than money leakage; the authors in [18] conclude that a moving average model can be used. We apply the same approach here. First note that Y_t appears to be reasonably gaussian (also see Section 2.4.3), and has correlation only at lag 1. We study such processes in Chapter 5; a gaussian process that has correlation only at lag 1 is the moving average process, which satisfies

$$Y_t - \mu = \epsilon_t + \alpha \epsilon_{t-1}$$

where ϵ_t is iid N_{0,σ^2} . This is a parametric model, with parameter (μ, α, σ) . We can fit it using a numerical package or the methods in Chapter 5. A confidence interval for μ can be obtained using

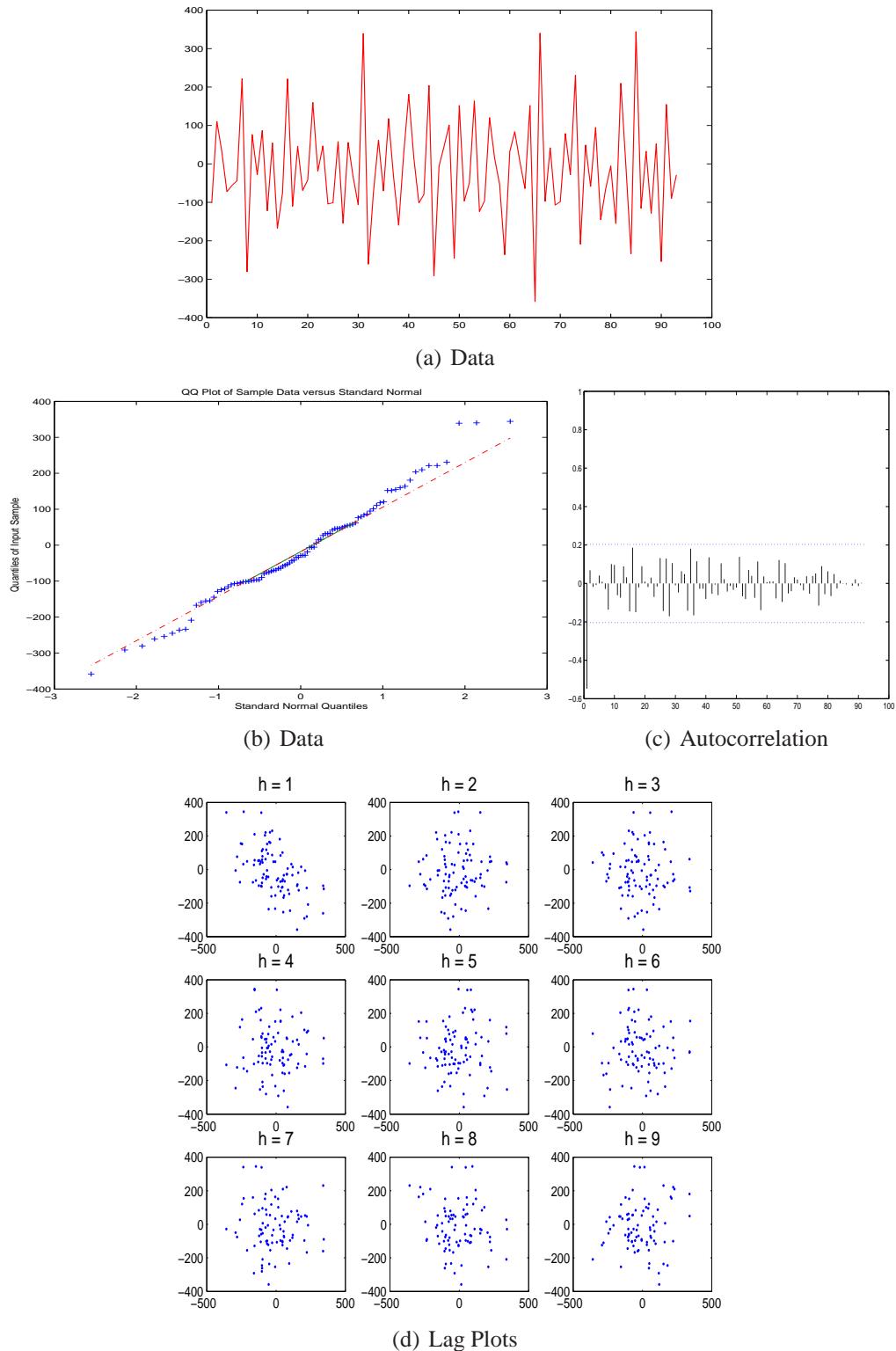


Figure 2.10: Daily balance at Joe's wireless access shop over 93 days. The lag plots show $y(t)$ versus $y(t + h)$ where $y(t)$ is the time series in (a). The data appears to have some correlation at lag 1 and is thus clearly not iid.

Theorem B.2 and Theorem D.2. Here, it is plausible that the sample size is large enough. For any fixed μ , we compute the profile log-likelihood. It is obtained by fitting an MA(1) process to $W_t := Y_t - \mu$. Good statistical packages give not only the MLE fit, but also the log-likelihood of the fitted model, which is exactly the profile log-likelihood $pl(\mu)$. The MLE $\hat{\mu}$ is the value of μ that maximizes $pl(\mu)$, and $-2(pl(\hat{\mu}) - pl(\mu))$ is approximately χ^2_1 . Figure 2.11 shows a plot of $pl(\mu)$.

It follows that $\hat{\mu} = -13.2$ and an approximate 95%-confidence interval is $[-14.1, -12.2]$. Contrary to the iid model, this suggests that there *is* a loss of money, in average 13€ per day.

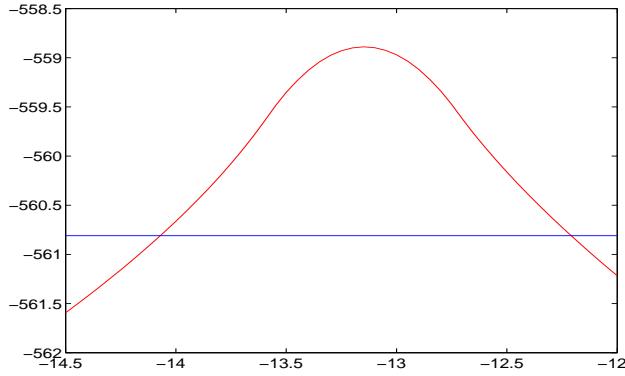


Figure 2.11: Profile Log Likelihood for the Moving Average model of Joe's balance data. The horizontal line is at a value $\eta/2 = 1.92$ below the maximum, with $\chi^2_1(\eta) = 0.95$; it gives an approximate confidence interval for the mean of the data on the x axis.

QUESTION 2.3.1. *Give an example of identically distributed but dependent random variables.*⁴

2.4 PREDICTION INTERVAL

The confidence intervals studied before quantify the accuracy of a mean or median; this is useful for diagnostic purposes, for example we can assert from the confidence intervals on Figure 2.7 that the new option does reduce the run time, because the confidence intervals for the mean (or the median) are in the positive numbers.

Sometimes we are interested in a different viewpoint and would like to characterize the **variability** of the data: for example we would like to summarize which run time can be expected for an arbitrary future (non observed) transaction. Clearly, this run time is random. A ***prediction interval*** at level γ is an interval that we can compute by observing a realization of X_1, \dots, X_n and such that, with probability γ , a future transaction will have a run time in this interval. Intuitively, if the common CDF of all X_i s would be known, then a prediction interval would simply be an interquantile interval, for example $[m_{\alpha/2}, m_{1-\alpha/2}]$, with $\alpha = 1 - \gamma$. For example, if the distribution is normal with known parameters, a prediction interval at level 0.95 would be $\mu \pm 1.96\sigma$. However, there is some additional uncertainty, due to the fact that we do not know the distribution, or its parameters a priori, and we need to estimate it. The prediction interval capture both uncertainties. Formally, the definition is as follows.

⁴Here is a simple one: assume X_1, X_3, X_5, \dots are iid with CDF $F()$ and let $X_2 = X_1, X_4 = X_3$ etc. The distribution of X_i is $F()$ but the distribution of X_2 conditional to $X_1 = x_1$ is a dirac at x_1 , thus depends on x_1 . The random choices taken for X_1 influence (here deterministically) the value of X_2 .

DEFINITION 2.2. Let X_1, \dots, X_n, X_{n+1} be a sequence of random variables. A prediction interval at level γ is an interval of the form $[u(X_1, \dots, X_n), v(X_1, \dots, X_n)]$ such that

$$\mathbb{P}(u(X_1, \dots, X_n) \leq X_{n+1} \leq v(X_1, \dots, X_n)) \geq \gamma \quad (2.31)$$

Note that the definition does not assume that X_i is iid, however we focus in this chapter on the iid case. The trick is now to find functions u and v that are pivots, i.e. their distribution is known even if the common distribution of the X_i s is not (or is not entirely known).

There is one general result, which applies in practice to sample sizes that are not too small ($n \geq 39$), which we give next.

2.4.1 PREDICTION FOR AN IID SAMPLE BASED ON ORDER STATISTIC

THEOREM 2.5 (General IID Case). Let X_1, \dots, X_n, X_{n+1} be an iid sequence and assume that the common distribution has a density. Let $X_{(1)}^n, \dots, X_{(n)}^n$ be the order statistic of X_1, \dots, X_n . For $1 \leq j \leq k \leq n$:

$$\mathbb{P}(X_{(j)}^n \leq X_{n+1} \leq X_{(k)}^n) = \frac{k-j}{n+1} \quad (2.32)$$

thus for $\alpha \geq \frac{2}{n+1}$, $[X_{(\lfloor (n+1)\frac{\alpha}{2} \rfloor)}^n, X_{(\lceil (n+1)(1-\frac{\alpha}{2}) \rceil)}^n]$ is a prediction interval at level at least $\gamma = 1 - \alpha$.

For example, with $n = 999$, a prediction interval at level 0.95 ($\alpha = 0.05$) is $[X_{(25)}, X_{(975)}]$. This theorem is similar to the bootstrap result in Section 2.2.4, but is exact and much simpler.

QUESTION 2.4.1. We have obtained n simulation results and use the prediction interval $[m, M]$ where m is the smallest result and M the largest. For which values of n is this a prediction interval at level at least 95%? ⁵

For very small n , this result gives poor prediction intervals with values of γ that maybe far from 100%. For example, with $n = 10$, the best prediction we can do is $[x_{\min}, x_{\max}]$, at level $\gamma = 81\%$. If we can assume that the data is normal, we have a stronger result, shown next.

⁵The interval is $[X_{(1)}, X_{(n)}]$ thus the level is $\frac{n-1}{n+1}$. It is ≥ 0.95 for $n \geq 39$. We need at least 39 samples to provide a 95% prediction interval.

2.4.2 PREDICTION FOR A NORMAL IID SAMPLE

THEOREM 2.6 (Normal IID Case). *Let X_1, \dots, X_n, X_{n+1} be an iid sequence with common distribution N_{μ, σ^2} . Let $\hat{\mu}_n$ and $\hat{\sigma}_n^2$ be as in Theorem 2.3. The distribution of $\sqrt{\frac{n}{n+1}} \frac{X_{n+1} - \hat{\mu}_n}{\hat{\sigma}_n}$ is Student's t_{n-1} ; a prediction interval at level $1 - \alpha$ is*

$$\hat{\mu}_n \pm \eta' \sqrt{1 + \frac{1}{n}} \hat{\sigma}_n \quad (2.33)$$

where η' is the $(1 - \frac{\alpha}{2})$ quantile of the student distribution t_{n-1} .

For large n , an approximate prediction interval is

$$\hat{\mu}_n \pm \eta \hat{\sigma}_n \quad (2.34)$$

where η is the $(1 - \frac{\alpha}{2})$ quantile of the normal distribution $N_{0,1}$.

For example, for $n = 100$ and $\alpha = 0.05$ we obtain the prediction interval (we drop the index n): $[\hat{\mu} - 1.99\hat{\sigma}, \hat{\mu} + 1.99\hat{\sigma}]$. Compare to the confidence interval for the mean given by Theorem 2.3 where the width of the interval is $\approx 10 = \sqrt{n}$ times smaller. For a large n , the prediction interval is approximately equal to $\hat{\mu}_n \pm \eta \hat{\sigma}_n$, which is the interval we would have if we ignore the uncertainty due to the fact that the parameters μ and σ are estimated from the data. For n as small as 26, the difference between the two is 7% and can be neglected in most cases.

The normal case is also convenient in that it requires the knowledge of only two statistics, the mean $\hat{\mu}_n$ and the mean of squares (from which $\hat{\sigma}_n$ is derived).

Comment Compare the prediction interval in Eq.(2.34) to the confidence interval for the mean in Eq.(2.24): there is a difference of $\frac{1}{\sqrt{n}}$; the confusion between both is frequently done: when comparing confidence interval, check if the standard deviation is indeed divided by \sqrt{n} !

EXAMPLE 2.6: FILE TRANSFER TIMES. Figure 2.12 shows the file transfer times obtained in 100 independent simulation runs, displayed in natural and log scales. The last panel shows 95%-prediction intervals. The left interval is obtained with the method of order statistic (Theorem 2.5); the middle one by (wrongly) assuming that the distribution is normal and applying Theorem 2.5 – it differs largely.

The right interval is obtained with a log transformation. First, a prediction interval $[u(Y_1, \dots, Y_n), v(Y_1, \dots, Y_n)]$ is computed for the transformed data $Y_i = \ln(X_i)$; the prediction interval is mapped back to the original scale to obtain the prediction interval $[\exp(u(\ln(X_1, \dots, \ln(X_n)))), \exp(v(\ln(X_1, \dots, \ln(X_n))))]$. We leave it to the alert reader to verify that this reverse mapping is indeed valid. The left and right intervals are in good agreement, but the middle one is obviously wrong.

The prediction intervals also show the central values (with small circles). For the first one, it is the median. For the second one, the mean. For the last one, $\exp\left(\frac{\sum_{i=1}^n Y_i}{n}\right)$, i.e. the back transformed of the mean of the transformed data (here, the geometric mean).

QUESTION 2.4.2. The prediction intervals in Figure 2.12 are not all symmetric around the central values. Explain why.⁶

⁶First interval: the distribution of the data is obviously not symmetric, so the median has no reason to be in the

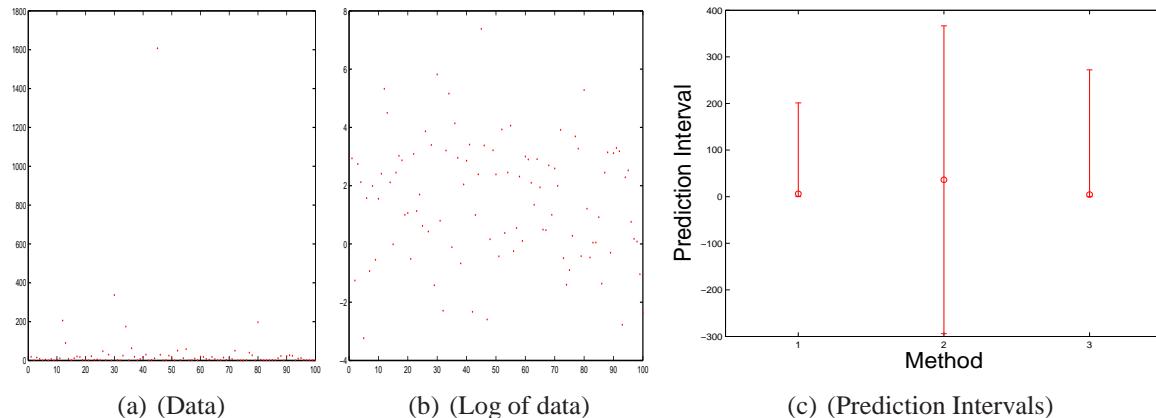


Figure 2.12: File transfer times for 100 independent simulation runs, with prediction intervals computed with the order statistic (1), assuming the data is normal (2) and assuming the log of data is normal (3).

There is no “large n ” result for a prediction interval, like there is in Theorem 2.2: a prediction interval depends on the original distribution of the X_i s, unlike confidence intervals for the mean, which depend only on first and second moments thanks to the central limit theorem. Theorem 2.6 justifies the common practice of using the standard deviation as a measure of dispersion; however it provides useful prediction intervals only if the data appears to be iid *and* normal. In the next section discuss how to verify normality.

2.4.3 THE NORMAL ASSUMPTION

QQPLOTS

This is a simple method for verifying the normal assumption, based on visual inspection. A **prob-ability plot**, also called **qq-plot**, compares two samples $X_i, Y_i, i = 1, \dots, n$ in order to determine whether they come from the same distribution. Call $X_{(i)}$ the **order statistic**, obtained by sorting X_i in increasing order. Thus $X_{(1)} \leq X_{(2)} \leq \dots$. The qq-plot displays the points $(X_{(i)}, Y_{(i)})$. If the points are approximately along a straight line, then the distributions of X_i and Y_i can be assumed to be the same, modulo a change of scale and location.

Most often, we use qqplots to check the distribution of Y_i against a probability distribution F . To do so, we plot $(x_i, Y_{(i)})$, where x_i is an estimation of the expected value of $\mathbb{E}(Y_{(i)})$, assuming the marginal of Y_i is F . The exact value of $\mathbb{E}(Y_{(i)})$ is hard to obtain, but a simple approximation (assuming that F is strictly increasing) is [32]:

$$x_i := F^{-1} \left(\frac{i}{n+1} \right)$$

A *normal qqplot*, is a qqplot such that $F = N_{0,1}$, and is often used to visually test for normality (Figure 2.13). More formal tests are the Jarque Bera test (Section 4.5.1) and the goodness of fit tests in Section 4.4.

middle of the extreme quantiles. Second interval: by nature, it is strictly symmetric. Third interval: it is the exponential of a symmetric interval; exponential is not an affine transformation, so we should not expect the transformed interval to be symmetric.

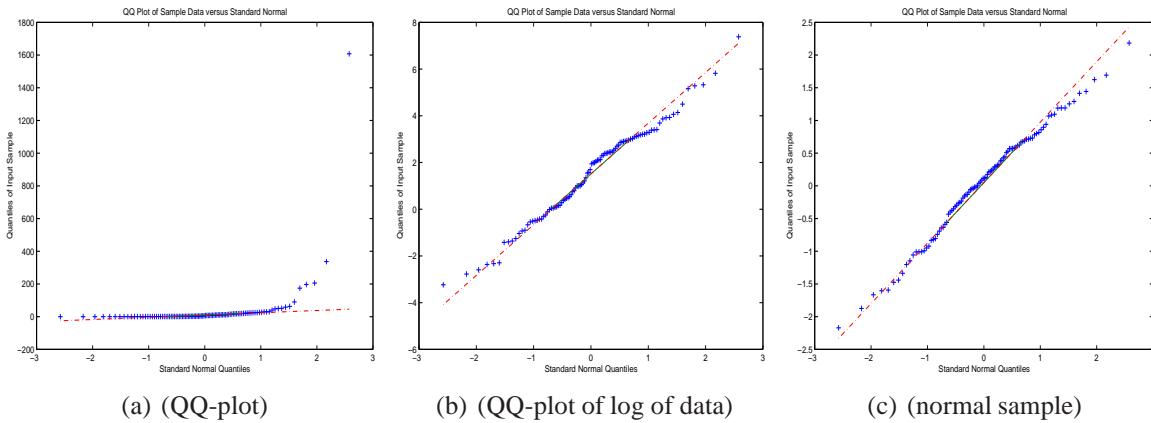


Figure 2.13: Normal qqplots of file transfer times in Figure 2.12 and of an artificially generated sample from the normal distribution with the same number of points. The former plot shows large deviation from normality, the second does not.

RESCALING, HARMONIC, GEOMETRIC AND OTHER MEANS

Figure 2.12 illustrates that the use of standard deviation as a basis for a prediction interval may be better if we re-scale the data, using a point transformation. The [Box-Cox transformation](#) is commonly used for that. It has one shape parameter s and is given by

$$b_s(x) = \begin{cases} \frac{x^s - 1}{s} & , s \neq 0 \\ \ln x & , s = 0 \end{cases} \quad (2.35)$$

Commonly used parameters are $s = 0$ (log transformation), $s = -1$ (inverse), $s = 0.5$ and $s = 2$. The reason for this specific form is to be continuous in s .

It is easy to see (as in Example 2.6) that a ***prediction interval*** for the original data can be obtained by reverse-transforming a prediction interval for the transformed data. In contrast, this does not hold for ***confidence intervals***. Indeed, by reverse-transforming a confidence interval for the mean of the transformed data, we obtain a confidence interval for another type of mean (harmonic, etc.). More precisely, assume we transform a data set x_1, \dots, x_n by an invertible (thus strictly monotonic) mapping $b()$ into y_1, \dots, y_n , i.e. $y_i = b(x_i)$ and $x_i = b^{-1}(y_i)$ for $i = 1, \dots, n$. We called ***transformed sample mean*** the quantity $b^{-1}\left(\frac{1}{n} \sum_{i=1}^n y_i\right)$, i.e. the back-transform of the mean of the transformed data. Similarly, the ***transformed distribution mean*** of the distribution of a random variable X is $b^{-1}(\mathbb{E}(b(X)))$. When $b()$ is a Box-Cox transformation with index $s = -1, 0$ or 2 we obtain the classical following definitions, valid for a positive data set $x_i, i = 1, \dots, n$ or a random variable X :

	<i>Transformation</i>	<i>Transformed Sample Mean</i>	<i>Transformed Distribution Mean</i>
<i>Harmonic</i>	$b(x) = 1/x$	$\frac{1}{\frac{1}{n} \sum_{i=1}^n \frac{1}{x_i}}$	$\mathbb{E}\left(\frac{1}{X}\right)$
<i>Geometric</i>	$b(x) = \ln(x)$	$\left(\prod_{i=1}^n x_i\right)^{\frac{1}{n}}$	$e^{\mathbb{E}(\ln X)}$
<i>Quadratic</i>	$b(x) = x^2$	$\sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}$	$\sqrt{\mathbb{E}(X^2)}$

THEOREM 2.7. A confidence interval for a transformed mean is obtained by the inverse transformation of a confidence interval for the mean of the transformed data.

For example, a confidence interval for the geometric mean is the exponential of a confidence interval for the mean of the logarithms of the data.

2.5 WHICH SUMMARIZATION TO USE ?

In the previous sections we have seen various summarization methods. In this section we discuss the use of these different methods.

The methods differ in their objectives: **confidence interval** for central value versus **prediction intervals**. The former quantify the accuracy of the estimated central value, the latter reflects how variable the data is. Both aspects are related (the more variable the data is, the less accurate the estimated central value is) but they are not the same.

The methods differ in the techniques used, and overlap to a large extend. They fall in two categories: methods based on the order statistic (confidence interval for median or other quantiles, Theorem 2.1; prediction interval computed with order statistic, Theorem 2.5) or based on mean and standard deviation (Theorems 2.3, 2.2, 2.6). The two types of methods differ in their **robustness versus compactness**.

2.5.1 ROBUSTNESS

WRONG DISTRIBUTIONAL HYPOTHESES

The confidence interval for the mean given by Theorem 2.2 requires that the central limit theorem applies i.e. (1) the common distribution has a finite variance and (2) the sample size n is large enough. While these two assumptions very often hold, it is important to detect cases where they do not.

Ideally, we would like to test whether the distribution of $T = \sum_{i=1}^n X_i$ is normal or not, but we cannot do this directly, since we have only one value of T . The bootstrap method can be used to solve this problem, as explained in the next example.

EXAMPLE 2.7: PARETO DISTRIBUTION. This is a toy example where we generate artificial data, iid, from a Pareto distribution on $[1, +\infty)$. It is defined by its cdf equal to $F(c) := \mathbb{P}(X > c) = \frac{1}{c^p}$ with $p = 1.25$; its mean is $= 5$, its variance is infinite (i.e. it is heavy tailed) and its median is 1.74.

Assume we would not know that it comes from a heavy tailed distribution and would like to use the asymptotic result in Theorem 2.2 to compute a confidence interval for the mean.

We use the bootstrap method to verify convergence to the normal distribution, as follows. We are given a data sample x_1, \dots, x_n from the Pareto distribution. We generate R replay experiments: for each r between 1 and R , we draw n samples X_i^r $i = 1, \dots, n$ with replacement from the list (x_1, \dots, x_n) and let $T^r = \frac{i=1}{n} X_i^r$. T^r is the r th bootstrap replicate of T ; we do a qqplot of the T^r , $r = 1, \dots, R$. If the distribution of T is normal, the qqplot should look normal as well.

We see that the qqplots do not appear normal, which is an indication that the central limit theorem might not hold. Indeed, the confidence interval for the mean is not very good.

The previous example shows a case where the confidence interval for the mean is not good, because

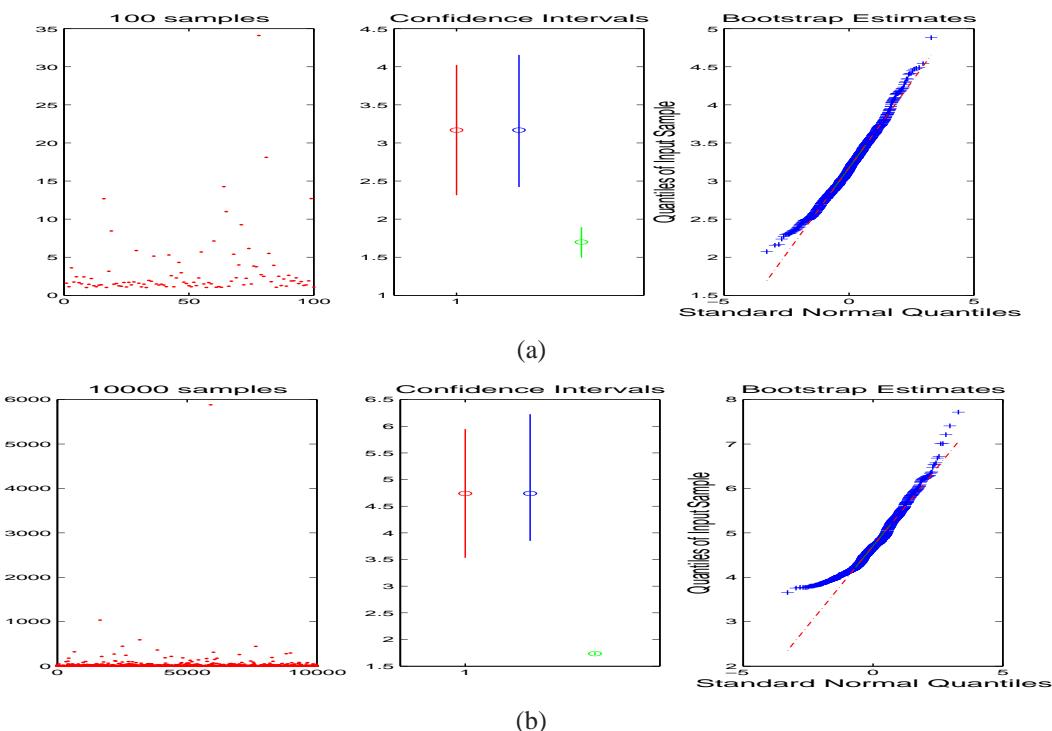


Figure 2.14: (a) Left: Artificially generated sample of 100 values from a Pareto distribution with exponent $p = 1.25$. Center: confidence intervals for the mean computed from Theorem 2.2 (left) and the bootstrap percentile estimate (center), and confidence interval for the median (right). Right: qqplot of 999 bootstrap replicates of the mean. The qqplot shows deviation from normality, thus the confidence interval given by Theorem 2.2 is not correct. Note that in this case the bootstrap percentile interval is not very good either, since it fails to capture the true value of the mean ($= 5$). In contrast, the confidence interval for the median does capture the true value ($= 1.74$). (b) Same with 10000 samples. The true mean is now within the confidence interval, but there is still no convergence to normality.

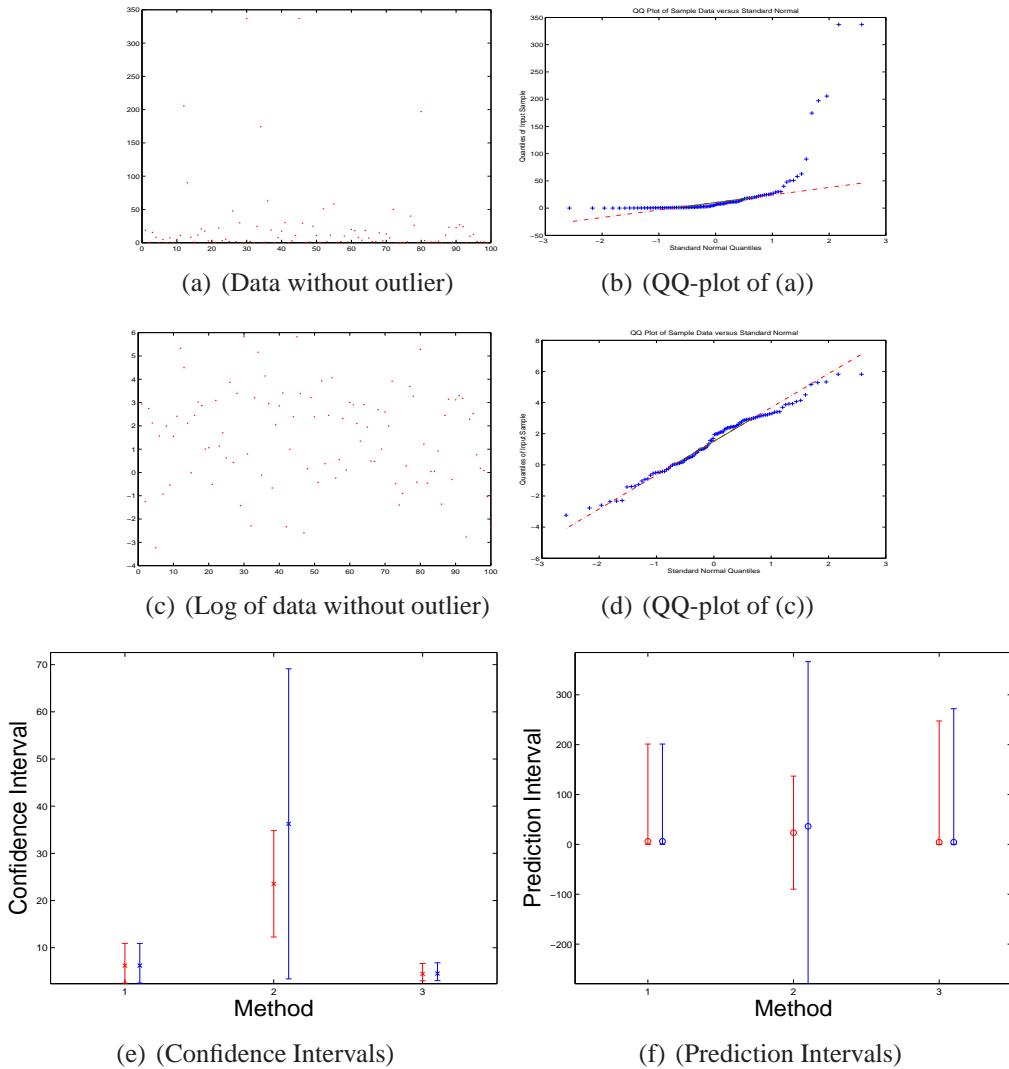


Figure 2.15: File transfer times for 100 independent simulation runs with outlier removed. Confidence intervals are without (left) and with (right) outlier, and with method (1) median (2) mean and (3) geometric mean. Prediction intervals are without (left) and with (right) outlier, computed with the three alternative methods discussed in Example 2.8: (1) order statistics (2) based on mean and standard deviation (3) based on mean and standard deviation after re-scaling.

a distributional assumption was made, which is not correct. In contrast, the confidence interval for the median **is** correct (Figure 2.14), as it does not require any distributional assumption (other than the iid hypothesis).

OUTLIERS

Methods based on the order statistic are more robust to outliers. An **outlier** is a value that significantly differs from the average. The median and the prediction interval based on order statistic are not affected by a few outliers, contrary to the mean and the prediction interval based on mean and standard deviation, as illustrated by the following example.

EXAMPLE 2.8: FILE TRANSFER WITH ONE OUTLIER. In fact in the data of Example 2.8 there is

	Index	Lower Bound, CI	Index	Upper Bound, CI
Without Outlier	JFI	0.1012	0.1477	0.3079
	gap	0.4681	0.5930	0.6903
With Outlier	JFI	0.0293	0.0462	0.3419
	gap	0.4691	0.6858	0.8116

Table 2.2: Fairness indices with and without outlier.

one very large value, 5 times larger than the next largest value. One might be tempted to remove it, on the basis that such a large value might be due to measurement error. A qqplot of the data without this “outlier” is shown on Figure 2.15, compare to the corresponding qq-plot with the outlier in Figure 2.13 (a,b). The prediction intervals based on order statistics are not affected, but the one based on mean and standard deviation is completely different.

Table 2.2 shows the values of Jain’s fairness index and the Lorenz curve gap is very sensitive to the presence of one outlier, which is consistent with the previous observation since Jain’s fairness index is defined by the ratio of standard deviation to mean (coefficient of variation). The Lorenz curve gap is less sensitive.

The outlier is less of an outlier on the re-scaled data (with the log transformation). The qqplot of the rescaled data is not affected very much, neither is the prediction interval based on mean and standard deviation of the rescaled data. Similarly, the confidence intervals for median and geometric mean are not affected, whereas that for the mean is. We do not show fairness indices for the re-scaled data since re-scaling changes the meaning of these indices.

Care should be taken to screen the data collection procedure for **true outliers**, namely values that are wrong because of measurement errors or problems. In the previous example, we should not remove the outlier. In practice it may be difficult to differentiate between true and spurious outliers. The example illustrates the following facts:

- Outliers may affect the prediction and confidence intervals based on mean and standard deviation, as well as the values of fairness indices. Jain’s fairness index is more sensitive than the Lorenz curve gap.
- This may go away if the data is properly rescaled. An outlier in some scale may not be an outlier in some other scale.
- In contrast, confidence intervals for the median and prediction intervals based on order statistics are more robust to outliers. They are not affected by re-scaling.

2.5.2 COMPACTNESS

Assume we wish to obtain both a central value with confidence interval and a prediction interval for a given data set. If we use methods based on order statistics, we will obtain a confidence interval for the median, and, say, a prediction interval at level 95%. Variability and accuracy are given by different sample quantiles, and cannot be deduced from one another. Furthermore, if we later are interested in 99% prediction intervals rather than 95%, we need to recompute new estimates of the quantiles. The same argument speaks in favour of quantifying the variability by means of the Lorenz curve gap.

In contrast, if we use methods based on mean and standard deviation, we obtain both confidence intervals and prediction intervals at any level with just 2 parameters (the sample mean and the sample standard deviation). In particular, the sample standard deviation gives indication on both accuracy of the estimator and variability of the data. However, as we saw earlier, these estimators are meaningful only in a scale where the data is roughly normal, if there is any.

Also, mean and standard deviation are less complex to compute than estimators based on order statistics, which require sorting the data. In particular, mean and standard deviation can be computed incrementally online, by keeping only 2 counters (sum of values and sum of squares). This reason is less valid today than some years ago, since there are sorting algorithms with complexity $n \ln(n)$.

2.6 OTHER ASPECTS OF CONFIDENCE/PREDICTION INTERVALS

2.6.1 INTERSECTION OF CONFIDENCE/PREDICTION INTERVALS

In some cases we have several confidence or prediction intervals for the same quantity of interest. For example, we can have a prediction interval I based on mean and standard deviation or I' based on order statistics. A natural deduction is to consider that the intersection $I \cap I'$ is a better confidence interval. This is almost true:

THEOREM 2.8. *If the random intervals I , I' are some confidence intervals at level $\gamma = 1 - \alpha$, $\gamma' = 1 - \alpha'$ then the intersection $I \cap I'$ is a confidence interval at level at least $1 - \alpha - \alpha'$. The same holds for prediction intervals.*

EXAMPLE 2.9: FILE TRANSFER TIMES. (Continuation of Example 2.8). We can compute two prediction intervals at level 0.975, using the order statistic method and the mean and standard deviation after rescaling (the prediction obtained without rescaling is not valid since the data is not normal). We obtain $[0.0394, 336.9]$ and $[0.0464, 392.7]$. We can conclude that a prediction interval at level 0.95 is $[0.0464, 336.9]$, which is better than the two.

Compare this interval to the prediction intervals at level 95% for each of the two methods; they are $[0.0624, 205.6]$ and $[0.0828, 219.9]$. Both are better.

Thus, for example if we combine two confidence intervals at level 97.5% we obtain a confidence interval at level 95%. As the example shows, this may be less good than an original confidence interval at level 95%.

2.6.2 THE MEANING OF CONFIDENCE

When we say that an interval I is a confidence interval at level 0.95 for some parameter θ , we mean the following. If we could repeat the experiment many times, in about 95% of the cases, the interval I would indeed contain the true value θ .

QUESTION 2.6.1. Assume 1000 students independently perform a simulation of an M/M/1 queue

with load factor $\rho = 0.9$ and find a 95% confidence interval for the result. The true result, unknown to these (unsophisticated) students is 9. The students are unsophisticated but conscientious, and all did correct simulations. How many of the 1000 students do you expect to find a wrong confidence interval, namely one that does not contain the true value? ⁷

2.7 PROOFS

THEOREM 2.1 Let $Z = \sum_{k=1}^n \mathbf{1}_{\{X_k \leq m_p\}}$ be the number of samples that lie below or at m_p . The CDF of Z is $B_{n,p}$ since the events $\{X_k \leq m_p\}$ are independent and $\mathbb{P}(X_k \leq m_p) = p$ by definition of the quantile m_p . Further:

$$\begin{aligned} j \leq Z &\Leftrightarrow X_{(j)} \leq m_p \\ k \geq Z + 1 &\Leftrightarrow X_{(k)} > m_p \end{aligned}$$

thus we have the event equalities

$$\{X_{(j)} \leq m_p < X_{(k)}\} = \{j \leq Z \leq k - 1\} = \{j - 1 < Z \leq k - 1\}$$

and

$$\mathbb{P}(X_{(j)} \leq m_p < X_{(k)}) = B_{n,p}(k - 1) - B_{n,p}(j - 1)$$

It follows that $[X_{(j)}, X_{(k)}]$ is a confidence interval for m_p at level γ as soon as $B_{n,p}(k - 1) - B_{n,p}(j - 1) \geq \gamma$.

The distribution of the X_i s has a density, thus $(X_{(j)}, X_{(k)})$ as well and $\mathbb{P}(X_{(j)} < m_p \leq X_{(k)}) = \mathbb{P}(X_{(j)} < m_p < X_{(k)})$, thus $[X_{(j)}, X_{(k)}]$ is also a confidence interval at the same level.

For large n , we approximate the binomial CDF by N_{μ, σ^2} with $\mu = np$ and $\sigma^2 = np(1-p)$, as follows:

$$\mathbb{P}(j - 1 < Z \leq k - 1) = \mathbb{P}(j \leq Z \leq k - 1) \approx N_{\mu, \sigma^2}(k - 1) - N_{\mu, \sigma^2}(j)$$

and we pick j and k such that

$$\begin{aligned} N_{\mu, \sigma^2}(k - 1) &\geq 0.5 + \frac{\gamma}{2} \\ N_{\mu, \sigma^2}(j) &\leq 0.5 - \frac{\gamma}{2} \end{aligned}$$

which guarantees that $N_{\mu, \sigma^2}(k - 1) - N_{\mu, \sigma^2}(j) \geq \gamma$. It follows that we need to have

$$\begin{aligned} k - 1 &\geq \eta\sigma + \mu \\ j &\leq -\eta\sigma + \mu \end{aligned}$$

We take the smallest k and the largest j which satisfy these constraints, which gives the formulas in the theorem.

THEOREM 2.5 Transform X_i into $U_i = F(X_i)$ which is iid uniform. For uniform RVs, use the fact that $\mathbb{E}(U_{(j)}) = \frac{j}{n+1}$. Then

$$\begin{aligned} &\mathbb{P}(U_{(j)}^n \leq U_{n+1} \leq U_{(k)}^n | U_{(1)}^n = u_{(1)}, \dots, U_{(n)}^n = u_{(n)}) \\ &= \mathbb{P}(u_{(j)} \leq U_{n+1} \leq u_{(k)}) \\ &= u_{(k)} - u_{(j)} \end{aligned}$$

The former is since U_{n+1} is independent of (U_1, \dots, U_n) and the latter since U_{n+1} has a uniform distribution on $[0, 1]$. Thus

$$\mathbb{P}(U_{(j)}^n \leq U_{n+1} \leq U_{(k)}^n) = \mathbb{E}(U_{(k)}^n - U_{(j)}^n) = \frac{k - j}{n + 1}$$

⁷Approximately 50 students should find a wrong interval.

THEOREM 2.6 First note that X_{n+1} is independent of $\hat{\mu}_n, \hat{\sigma}_n$. Thus $X_{n+1} - \hat{\mu}_n$ is normal with mean 0 and variance

$$\text{var}(X_{n+1}) + \text{var}(\hat{\mu}_n) = \sigma^2 + \frac{1}{n}\sigma^2$$

Further, $(n - 1)\hat{\sigma}_n^2/\sigma^2$ has a χ_{n-1}^2 distribution and is independent of $X_{n+1} - \hat{\mu}_n$. By definition of Student's t , the theorem follows.

THEOREM 2.7 Let m' be the distribution mean of $b(X)$. By definition of a confidence interval, we have $\mathbb{P}(u(Y_1, \dots, Y_n) < m' < v(Y_1, \dots, Y_n)) \geq \gamma$ where the confidence interval is $[u, v]$. If $b()$ is increasing (like the Box-Cox transformation with $s \geq 0$) then so is $b^{-1}()$ and this is equivalent to $\mathbb{P}(b^{-1}(u(Y_1, \dots, Y_n)) < b^{-1}(m') < b^{-1}(v(Y_1, \dots, Y_n))) \geq \gamma$. Now $b^{-1}(m')$ is the transformed mean, which shows the statement in this case. If $b()$ is decreasing (like the Box-Cox transformation with $s < 0$) then the result is similar with inversion of u and v .

THEOREM 2.8 We do the proof for a confidence interval for some quantity θ , the proof is the same for a prediction interval. By definition $\mathbb{P}(\theta \notin I) \leq \alpha$ and $\mathbb{P}(\theta \notin I') \leq \alpha'$. Thus

$$\mathbb{P}(\theta \notin I \cap I') = \mathbb{P}((\theta \notin I) \text{ or } (\theta \notin I')) \leq \mathbb{P}(\theta \notin I) + \mathbb{P}(\theta \notin I') \leq \alpha + \alpha'$$

2.8 REVIEW

2.8.1 SUMMARY

1. A **confidence** interval is used to quantify the **accuracy** of a parameter estimated from the data.
2. For computing the central value of a data set, you can use either mean or median. Unless you have special reasons (see below) for not doing so, the median is a preferred choice as it is more robust. You should compute not only the median but also a confidence interval for it, using Table A.1 on Page 313.
3. A **prediction** interval reflects the **variability** of the data. For small data sets ($n < 38$) it is not meaningful. For larger data sets, it can be obtained by Theorem 2.5. The Lorenz curve gap also gives a scale free representation of the variability of the data.
4. Fairness indices are essentially the same as indices of variability. Jain' Fairness index is based on standard deviation, and is less robust than the Lorenz Curve gap, which should be preferred.
5. A confidence interval for the mean characterizes both the **variability** of the data and the **accuracy** of the measured average. In contrast, a confidence interval for the median does not reflect well the variability of the data, therefore if we use the median we need both a confidence interval for the median and some measure of variability (the quantiles, as on a Box Plot). Mean and standard deviation give an accurate idea of the **variability** of the data, but only if the data is roughly normal. If it is not, it should be re-scaled using for example a Box-Cox transformation. Normality can be verified with a qq-plot.
6. The standard deviation gives an accurate idea of the **accuracy** of the mean if the data is normal, but also if the data set is large. The latter can be verified with a bootstrap method.

7. The geometric [resp. harmonic] mean is meaningful if the data is roughly normal in log [resp. $1/x$] scale. A confidence interval for the geometric [resp. harmonic] mean is obtained as the exponential [resp. inverse] of the mean in log [resp. $1/x$] scale.
8. All estimators in this chapter are valid only if the data points are independent (non correlated). This assumption must be verified, either by designing the experiments in a randomized way, (as is the case with independent simulation runs), or by formal correlation analysis.
9. If you have a choice, use median and quantiles rather than mean and standard deviation, as they are robust to distributional hypotheses and to outliers. Use prediction intervals based on order statistic rather than the classical mean and standard deviation.

2.8.2 REVIEW QUESTIONS

QUESTION 2.8.1. Compare (1) the confidence interval for the median of a sample of n data values, at level 95% and (2) a prediction interval at level at least 95%, for $n = 9, 39, 99$.⁸

QUESTION 2.8.2. Call $L = \min\{X_1, X_2\}$ and $U = \max\{X_1, X_2\}$. We do an experiment and find $L = 7.4$, $U = 8.0$. Say which of the following statements is correct: (θ is the median of the distribution). (1) the probability of the event $\{L \leq \theta \leq U\}$ is 0.5 (2) the probability of the event $\{7.4 \leq \theta \leq 8.0\}$ is 0.5⁹

QUESTION 2.8.3. How do we expect a 90% confidence interval to compare to a 95% one ? Check this on the tables in Section A.¹⁰

QUESTION 2.8.4. A data set has 70 points. Give the formulae for confidence intervals at level 0.95 for the median and the mean¹¹

QUESTION 2.8.5. A data set has 70 points. Give formulae for a prediction intervals at level 95%¹²

QUESTION 2.8.6. A data set x_1, \dots, x_n is such that $y_i = \ln x_i$ looks normal. We obtain a confidence interval $[\ell, u]$ for the mean of y_i . Can we obtain a confidence interval for the mean of x_i by a transformation of $[\ell, u]$?¹³

⁸From the tables in Chapter A and Theorem 2.5 we obtain: (confidence interval for median, prediction interval): $n = 9$: $[x_{(2)}, x_{(9)}]$, impossible; $n = 39$: $[x_{(13)}, x_{(27)}]$, $[x_{(1)}, x_{(39)}]$; $n = 99$: $[x_{(39)}, x_{(61)}]$, $[x_{(2)}, x_{(97)}]$. The confidence interval is always smaller than the prediction interval.

⁹In the classical (non-Bayesian) framework, (1) is correct and (2) is wrong. There is nothing random in the event $\{7.4 \leq \theta \leq 8.0\}$, since θ is a fixed (though unknown) parameter. The probability of this event is either 0 or 1, here it happens to be 1. Be careful with the ambiguity of a statement such as “the probability that θ lies between L and U is 0.5”. In case of doubt, come back to the roots: the probability of an event can be interpreted as the ideal proportion of simulations that would produce the event.

¹⁰It should be smaller. If we take more risk we can accept a smaller interval. We can check that the values of j [resp. k] in the tables confidence intervals at level $\gamma = 0.95$ are larger [resp. smaller] than at confidence level $\gamma = 0.99$.

¹¹Median: from the table in Section A $[x_{(27)}, x_{(44)}]$. Mean: from Theorem 2.2: $\hat{\mu} \pm 0.2343S$ where $\hat{\mu}$ is the sample mean and S the sample standard deviation. The latter is assuming the normal approximation holds, and should be verified by either a qqplot or the bootstrap.

¹²From Theorem 2.5: $[\min_i x_i, \max_i x_i]$.

¹³No, we know that $[e^\ell, e^u]$ is a confidence interval for the geometric mean, not the mean of x_i . In fact x_i comes from a log-normal distribution, whose mean is $e^{\mu + \frac{\sigma^2}{2}}$ where μ is the mean of the distribution of y_i , and σ^2 its variance.

QUESTION 2.8.7. Assume a set of measurements is corrupted by an error term that is normal, but positively correlated. If we would compute a confidence interval for the mean using the iid hypothesis, would the confidence interval be too small or too large ? ¹⁴

QUESTION 2.8.8. We estimate the mean of an iid data set by two different methods and obtain 2 confidence intervals at level 95%: $I_1 = [2.01, 3.87]$, $I_2 = [2.45, 2.47]$. Since the second interval is smaller, we discard the first and keep only the second. Is this a correct 95% confidence interval ? ¹⁵

¹⁴Too small: we underestimate the error. This phenomenon is known in physics under the term **personal equation**: if the errors are linked to the experimenter, they are positively correlated.

¹⁵No, by doing so we keep the interval $I = I_1 \cap I_2$, which is a 90% confidence interval, not a 95% confidence interval.

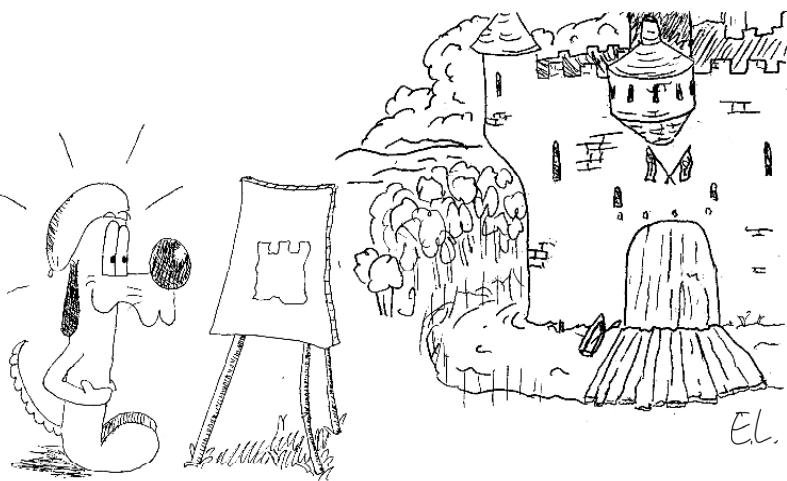
CHAPTER 3

MODEL FITTING

In this chapter we study how to derive a model from data, for example, by fitting a curve to a series of measurements. The method of least squares is widely used, and gives simple, often linear algorithms. However, it should be used with care, as it makes the hidden assumption that error terms are gaussian with same variance. We also discussed the less known alternative called ℓ^1 norm minimization, which implicitly assumes that error terms have a Laplace instead of gaussian distribution.

The resulting algorithms may be less simple, but are often tractable, as they correspond to convex (rather than linear) optimization, and the method is more robust to outliers or wrong distributional assumptions.

We discuss in detail the so-called “linear models”; what is linear here is the dependence on the hidden parameters, not the model itself. This is a very rich family of models with wide applicability. We discuss both least square and ℓ^1 norm minimization in this context.



Then we discuss the issue of fitting a distribution to a data set; we describe commonly used features that are helpful to pick an appropriate distribution: distribution shape, power laws, fat tail and heavy tail. The latter property is often encountered in practice and is often interesting or annoying. We address the practical issues of fitting censored data (i.e. when we could observe only values smaller than some unknown threshold) and how to separately fit the body and the tail of a distribution. We illustrate how the concepts and techniques could be used to build a load generation tool.

Contents

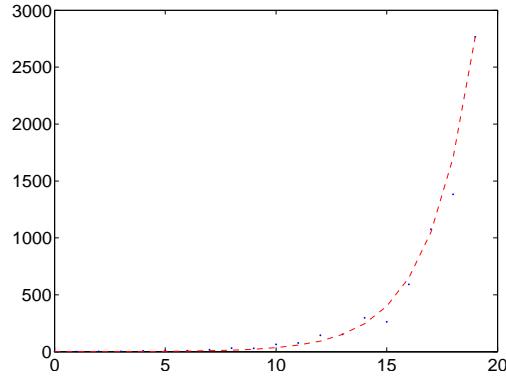
3.1 Model Fitting Criteria	60
3.1.1 What is Model Fitting ?	60
3.1.2 Least Squares Correspond to Gaussian, Same Variance	64
3.1.3 ℓ^1 Norm Minimization Corresponds to Laplace Noise	65
3.2 Linear Regression	66
3.3 Linear Regression with ℓ^1 Norm Minimization	70
3.4 Choosing a Distribution	73
3.4.1 Shape	73
3.4.2 Skewness and Kurtosis	74
3.4.3 Power Laws, Pareto Distribution and Zipf's Law	76
3.4.4 Hazard Rate	77
3.4.5 Fitting A Distribution	78
3.4.6 Censored Data	80
3.4.7 Combinations of Distributions	82
3.5 Heavy Tail	84
3.5.1 Definition	84
3.5.2 Heavy Tail and Stable Distributions	85
3.5.3 Heavy Tail in Practice	85
3.5.4 Testing For Heavy Tail	88
3.5.5 Application Example: The Workload Generator SURGE	89
3.6 Proofs	91
3.7 Review	92
3.7.1 Review Questions	92
3.7.2 Useful Matlab Commands	92

3.1 MODEL FITTING CRITERIA

3.1.1 WHAT IS MODEL FITTING ?

We start with a simple example.

EXAMPLE 3.1: VIRUS SPREAD DATA. The number of hosts infected by a virus is plotted versus time in hours.



The plot suggests an exponential growth, therefore we are inclined to fit these data to a model of the form

$$Y(t) = ae^{\alpha t} \quad (3.1)$$

where $Y(t)$ is the number of infected hosts at time t . We are particularly interested in the parameter α , which can be interpreted as the growth rate; the *doubling time* (time for the number of infected hosts to double) is $\frac{\ln 2}{\alpha}$. On the plot, the dashed line is the curve fitted by the method of least squares explained later. We find $\alpha = 0.4837$ per hour and the doubling time is 1.43 hour. We can use the model to predict that, 6 hours after the end of the measurement period, the number of infected hosts would be ca. 82'000.

In general, *model fitting* can be defined as the problem of finding an *explanatory model* for the data, i.e. a mathematical relation of the form

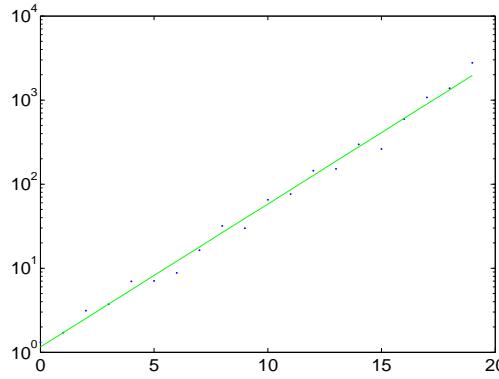
$$y_i = f_i(\vec{\beta}) \quad (3.2)$$

that “explains the data well”, in some sense. Here y_i is the collection of measured data, i is the index of a measurement, f_i is an array of functions, and $\vec{\beta}$ is the parameter that we would like to obtain. In the previous example, the parameter is $\vec{\beta} = (a, \alpha)$ and $f_i(\vec{\beta}) = f_i(a, \alpha) = ae^{\alpha t_i}$ where t_i is the time of the i th measurement, assumed here to be known.

What does it mean to “explain the data well”? It is generally not possible to require that Eq.(3.2) holds *exactly* for all data points. Therefore, a common answer is to require that the model minimizes some metric of the discrepancy between the explanatory model and the data. A very common metric is the mean square distance $\sum_i (y_i - f_i(\vec{\beta}))^2$. The value of the growth rate α in the previous example was obtained in this way, namely, we computed a and α that minimize $\sum_i (y_i - ae^{\alpha t_i})^2$.

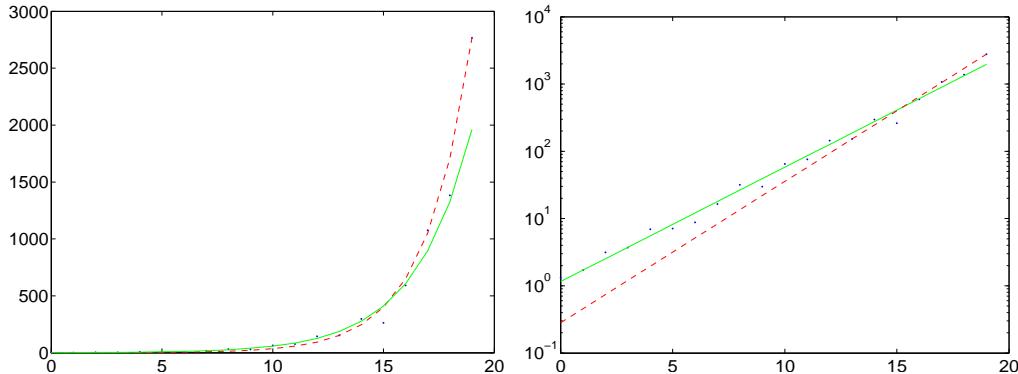
But this raises another question. What metric should one use? What is so magical about least squares? Why not use other measures of discrepancy, for example $\sum_i |y_i - f_i(\vec{\beta})|$ or $\sum_i (\ln(y_i) - \ln(f_i(\vec{\beta})))^2$? The following example shows the importance of the issue.

EXAMPLE 3.2: VIRUS SPREAD DATA, CONTINUED. AMBIGUITY IN THE OPTIMIZATION CRITERION. We also plotted the number of infected hosts in log scale:



and computed the least square fit of Eq.(3.2) in log scale (plain line). Namely, we computed a and α that minimize $\sum_i (\ln(y_i) - \ln(a) - \alpha t_i)^2$. We found for α the value 0.39 per hour, which gives a doubling time of 1.77 hour and a prediction at time +6 hours equal to ca. 39'000 infected hosts (instead of previously 82'000).

The two different models are compared below (in linear and log scales).



Both figures show that what visually appears to be a good fit in one scale is not so in the other. Which one should we use ?

An answer to the issue comes from statistics. The idea is to add to the explanatory model a description of the “noise” (informally defined as the deviation between the explanatory model and the data), and obtain a **statistical model**. We can also think of the statistical model as a description of a simulator that was used to produce the data we have. Its parameters are well defined, but not known to us.

The statistical model usually has a few more parameters than the explanatory model. The parameters of the statistical model are estimated using the classical approach of maximum likelihood. If we believe in the statistical model, this answers the previous issue by saying that the criterion to be optimized is the likelihood. The belief in the model can be checked by examining residuals.

EXAMPLE 3.3: VIRUS SPREAD DATA, CONTINUED. A STATISTICAL MODEL. One **statistical model** for the virus spread data is

$$Y_i = ae^{\alpha t_i} + \epsilon_i \text{ with } \epsilon_i \text{ iid } \sim N_{0,\sigma^2} \quad (3.3)$$

in other words, we assume that the measured data y_i is equal to the ideal value given by the explanatory model, plus a noise term ϵ_i . Further, we assume that all noises are independent, gaussian, and with same variance. The parameter is $\theta = (a, \alpha, \sigma)$.

In Eq.(3.3), we write Y_i instead of y_i to express that Y_i is a random variable. We think of our data y_i as being *one* sample produced by a simulator that implements Eq.(3.3).

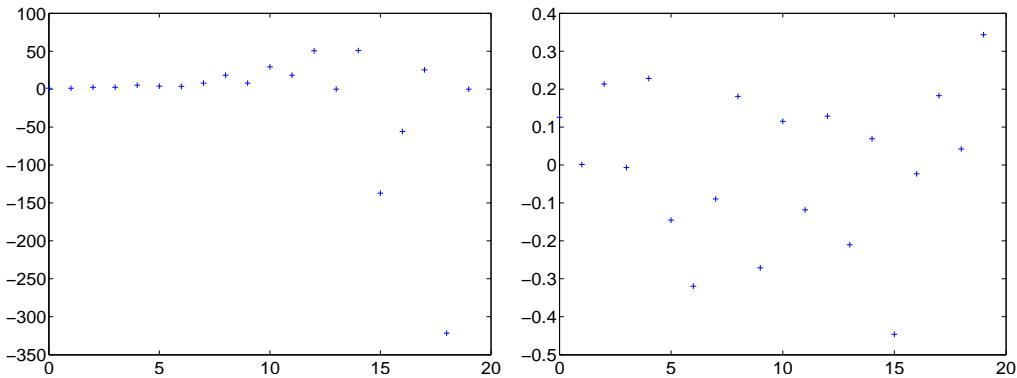
We will see in Section 3.1.2 that the maximum likelihood estimator for this model is the one that minimizes the mean square distance. Thus, with this model, we obtain for α the value in Example 3.1.

A second statistical model could be:

$$\ln(Y_i) = \ln(ae^{\alpha t_i}) + \epsilon_i \text{ with } \epsilon_i \text{ iid } \sim N_{0,\sigma^2} \quad (3.4)$$

Now, we would be assuming that the noise terms in log-scale have the same variance, in other words, the noise is proportional to the measured value. Here too, the maximum likelihood estimator is obtained by minimizing the least square distance, thus we obtain for α the value in Example 3.2.

We can validate either model by plotting the residuals:



We see clearly that the residual for the former model do not appear to be normally distributed, and the converse is true for the former model, which is the one we should adopt. Therefore, an acceptable fitting is obtained by minimizing least squares in log-scale.

We summarize what we have learnt so far as follows.

FITTING A MODEL TO DATA

1. Define a statistical model that contains **both** the deterministic part (the one we are interested in) and a model of the noise.
2. Estimate the parameters of the statistical model using maximum likelihood. If the number of data points is small, use a brute force approach (e.g use `fminsearch`). If the number of data points is large, you may need to look in the literature for efficient, possibly heuristic, optimization methods.
3. Validate the model fit by screening the residuals, either visually, or using tests (Chapter 4). In practice, you will seldom obtain a perfect fit; however, large deviations indicate that the model might not be appropriate.

3.1.2 LEAST SQUARES CORRESPOND TO GAUSSIAN, SAME VARIANCE

A very frequent case is when the statistical model has the form

$$Y_i = f_i(\vec{\beta}) + \epsilon_i \text{ for } i = 1, \dots, I \text{ with } \epsilon_i \text{ iid } \sim N_{0, \sigma^2} \quad (3.5)$$

as in the examples before (Models in Equations (3.3) and (3.4)). Namely, the discrepancy between the explanatory model and the data is assumed to be gaussian with **same variance**. In some literature, the “same variance” assumption is called **homoscedasticity**.

The next theorem explains what we do when we fit the explanatory model $y_i = f_i(\vec{\beta})$ to our data using least squares: we implicitly assume that the error terms in our data are independent, gaussian, and of same amplitude. We have seen in the examples above that care must be taken to validate this assumption, in particular, some rescaling may be needed for a better validation.

THEOREM 3.1 (Least Squares). *For the model in Eq.(3.5),*

1. *the maximum likelihood estimator of the parameter $(\vec{\beta}, \sigma)$ is given by:*

$$(a) \hat{\beta} = \arg \min_{\vec{\beta}} \sum_i (y_i - f_i(\vec{\beta}))^2$$

$$(b) \hat{\sigma}^2 = \frac{1}{I} \sum_i (y_i - f_i(\hat{\beta}))^2$$

2. *Let K be the square matrix of second derivatives (assumed to exist), defined by*

$$K_{j,k} = \frac{1}{\sigma^2} \sum_i \frac{\partial f_i}{\partial \beta_j} \frac{\partial f_i}{\partial \beta_k}$$

If K is invertible and if the number I of data points is large, $\hat{\beta} - \vec{\beta}$ is approximately gaussian with 0 mean and covariance matrix K^{-1} .

Alternatively, for large I , an approximate confidence set at level γ for the j th component β_j of $\vec{\beta}$ is implicitly defined by

$$-2I \ln(\hat{\sigma}) + 2I \ln \left(\hat{\sigma}(\hat{\beta}_1, \dots, \hat{\beta}_{j-1}, \beta_j, \hat{\beta}_{j+1}, \dots, \hat{\beta}_p) \right) \geq \xi_1$$

where $\hat{\sigma}^2(\vec{\beta}) = \frac{1}{I} \sum_i (y_i - f_i(\vec{\beta}))^2$ and ξ_1 is the γ quantile of the χ^2 distribution with 1 degree of freedom (for example, for $\gamma = 0.95$, $\xi_1 = 3.92$).

The set of points in \mathbb{R}^I that have coordinates of the form $f_i(\vec{\beta})$ constitue a “manifold” (for $p = 2$, it is a surface). Item 1 (a) says that $\vec{\beta}$ is the parameter of the point \hat{y} on this manifold that is the nearest to the data point \vec{y} , in euclidian distance. The point \hat{y} is called the **predicted response**; it is an estimate of the value that \vec{y} would take if there would be no noise. It is equal to the orthogonal projection of the data \vec{y} onto the manifold.

The rest of the theorem can be used to obtain accuracy bounds for the estimation. A slight variant of the theorem can be used to make predictions with accuracy bounds, see Theorem 5.1.

3.1.3 ℓ^1 NORM MINIMIZATION CORRESPONDS TO LAPLACE NOISE

Although less traditional than least square, minimization of the absolute deviation of the error is also used. The absolute deviation is the ℓ^1 norm of the error¹, so this method is also called **ℓ^1 norm minimization**. Since it gives less weight to outliers, it is expected to be more robust. As we see now, it corresponds to assuming that errors follow a Laplace distribution (i.e. bilateral exponential).

The **Laplace distribution** with 0 mean and rate λ is the two sided exponential distribution, or, in other words, $X \sim \text{Laplace}(\lambda)$ if and only if $|X| \sim \text{Exp}(\lambda)$. It can be used to model error terms that have a heavier tail than the normal distribution. Its PDF is defined for $x \in \mathbb{R}$ by

$$f(x) = \frac{\lambda}{2} e^{-\lambda|x|} \quad (3.6)$$

The next theorem explains what we do when we fit the explanatory model $y_i = f_i(\vec{\beta})$ to our data by minimizing the ℓ^1 norm of the error: we implicitly assume that the error terms in our data are independent, Laplace with the same parameter, i.e., the data yy_i is a sample generated by the model

$$Y_i = f_i(\vec{\beta}) + \epsilon_i \text{ with } \epsilon_i \text{iid} \sim \text{Laplace}(\lambda) \quad (3.7)$$

THEOREM 3.2 (Least Deviation). *For the model in Eq.(3.7), the maximum likelihood estimator of the parameter $(\vec{\beta}, \lambda)$ is given by:*

1. $\hat{\beta} = \arg \min_{\vec{\beta}} \sum_i |y_i - f_i(\vec{\beta})|$
2. $\frac{1}{\lambda} = \frac{1}{I} \sum_i |y_i - f_i(\hat{\beta})|$

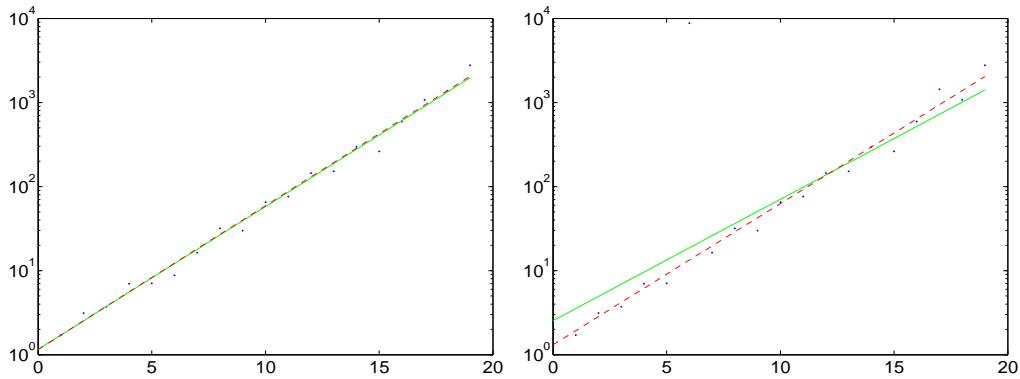


Figure 3.1: Fitting an exponential growth model to the data in Example 3.1, showing the fits obtained with least square (plain) and with ℓ^1 norm minimization (dashed). First panel: original data; both fits are the same; Second panel: data corrupted by one outlier; the fit with ℓ^1 norm minimization is not affected, whereas the least square fit is.

EXAMPLE 3.4: VIRUS PROPAGATION WITH ONE OUTLIER. Assume the data in the virus propagation example (Example 3.1) is modified by changing the value of the second data point. Assume

¹The ℓ^1 norm of a sequence $z = (z_1, \dots, z_n)$ is $\|z\|_1 = \sum_{i=1}^n |z_i|$

we fit the data in log scale. The modified data is an outlier; perhaps one would be tempted to remove it; an alternative is to fit the log of the data to Laplace noise instead of gaussian noise (i.e. do ℓ^1 norm minimization instead of least squares), as this is known to be more robust. Figure 3.1.3, and the table below shows the results (the prediction in the table is a 6-hours ahead point prediction).

	Least Square		ℓ^1 norm minimization	
	rate	prediction	rate	prediction
no outlier	0.3914	30300	0.3938	32300
with one outlier	0.3325	14500	0.3868	30500

We see that one single outlier completely modifies the result of least square fitting, whereas ℓ^1 norm minimization fitting is not impacted much.

The following example is important to understand the difference between least square and ℓ^1 norm minimization.

EXAMPLE 3.5: MEAN VERSUS MEDIAN. Assume we want to fit a data set y_i , $i = 1, \dots, I$ against a constant μ .

With least square fitting, we are looking for μ that minimizes $\sum_{i=1}^I (y_i - \mu)^2$. The solution is easily found to be $\mu = \frac{1}{I} \sum_{i=1}^I y_i$, i.e. μ is the sample mean.

With ℓ^1 norm minimization, we are looking for μ that minimizes $\sum_{i=1}^I |y_i - \mu|$. The solution is the median of y_i .

To see why, consider the mapping $f : \mu \mapsto \sum_{i=1}^I |y_i - \mu|$. Consider to simplify the case where all values y_i are distinct and written in increasing order ($y_i < y_{i+1}$). The derivative f' of f is defined everywhere except at points y_i , and for $y_i < \mu < y_{i+1}$, $f'(\mu) = i - (I - i) = 2i - I$. If I is odd, f decreases on $(-\infty, y_{(I+1)/2}]$ and increases on $[y_{(I+1)/2}, +\infty)$, thus is minimum for $\mu = y_{(I+1)/2}$, which is the sample median. If I is even, f is minimum at all values in the interval $[y_{I/2}, y_{I/2+1}]$ thus reaches the minimum at the sample median $\frac{y_{I/2}, y_{I/2+1}}{2}$.

In terms of computation, ℓ^1 norm minimization is more complex than least squares, though both are usually tractable. For example, if the dependency on the parameter is linear, least square fitting consists in solving a linear system of equations whereas ℓ^1 norm minimization uses linear programming (as shown in the next section).

3.2 LINEAR REGRESSION

This is a special case of least square fitting, where the explanatory model depends linearly on its parameter $\vec{\beta}$. This is called the *linear regression* model. The main fact here is that everything can be computed easily, using linear algebra. Be careful that the term “linear regression” implicitly assumes least square fitting. The popular fitting method called “ANOVA” is a special case of linear regression.

Assume thus that the *statistical model* of our experiment has the form:

DEFINITION 3.1 (Linear Regression Model).

$$Y_i = (X\vec{\beta})_i + \epsilon_i \text{ for } i = 1, \dots, I \text{ with } \epsilon_i \text{ iid } \sim N_{0,\sigma^2} \quad (3.8)$$

where the unknown parameter $\vec{\beta}$ is in \mathbb{R}^p and X is a $I \times p$ matrix. The matrix X supposed to be known exactly in advance. We also assume that

H X has rank p

Assumption **H** means that different values of $\vec{\beta}$ give different values of the explanatory model $X\vec{\beta}$, i.e. the explanatory model is identifiable.

The elements of the known matrix X are sometimes called **explanatory variables**, and then the y_i s are called the **response variables**.

EXAMPLE 3.6: JOE'S SHOP AGAIN, FIGURE 1.3(B). We assume that there is a threshold ξ beyond which the throughput collapses (we take $\xi = 70$). The statistical model is

$$Y_i = (a + bx_i)1_{x_i \leq \xi} + (c + dx_i)1_{\{x_i > \xi\}} + \epsilon_i \quad (3.9)$$

where we impose

$$a + b\xi = c + d\xi \quad (3.10)$$

In other words, we assume the throughput response curve to be piecewise linear. Eq.(3.10) expresses that the curve is continuous. Recall that x_i is the offered load and Y_i is the actual throughput.

Here we take $\vec{\beta} = (a, b, d)$ (we can derive $c = a + (b - d)\xi$ from Eq.(3.10)). The dependency of Y_i on $\vec{\beta}$ is indeed linear. Note that we assume that ξ is known; see in Example 3.8 how to estimate ξ .

Assume that we sort the x_i s in increasing order and let i^* be the largest index i such that $x_i \leq \xi$. Re-write Eq.(3.9) as

$$\begin{aligned} Y_i &= a + bx_i + \epsilon_i \text{ for } i = 1 \dots i^* \\ Y_i &= a + b\xi + d(x_i - \xi) + \epsilon_i \text{ for } i = i^* + 1 \dots I \end{aligned}$$

thus the matrix X is given by:

$$\left(\begin{array}{ccc} 1 & x_1 & 0 \\ 1 & x_2 & 0 \\ \dots & \dots & \dots \\ 1 & x_{i^*} & 0 \\ 1 & \xi & x_{i^*+1} - \xi \\ \dots & \dots & \dots \\ 1 & \xi & x_I - \xi \end{array} \right)$$

It is simple to see that a *sufficient* condition for **H** is that there are at least two distinct values of $x_i \leq \xi$ and at least one value $> \xi$.

QUESTION 3.2.1. Show this.²

²We need to show, if the condition is true, that the matrix X has rank $p = 3$. This is equivalent to saying that the

A model as in this example is sometimes called *Intervention Analysis*.

With the linear regression model, the manifold mentioned in the discussion after Theorem 3.1 is a linear manifold (for $p = 2$, a plane). It is equal to the linear sub-space spanned by the columns of matrix X . The nearest point is given by an orthogonal projection, which can be computed exactly. The details are given in the following theorem which is a consequence of Eq.(C.3) on Page 329 and Theorem C.4; a complete proof is in [32, section 2.3].

THEOREM 3.3 (Linear Regression). *Consider the model in Definition 3.1; let \vec{y} be the $I \times 1$ column vector of the data.*

1. *The $p \times p$ matrix $(X^T X)$ is invertible*

2. *(Estimation) The maximum likelihood estimator of $\vec{\beta}$ is $\hat{\beta} = K\vec{y}$ with $K = (X^T X)^{-1} X^T$*

3. *(Standardized Residuals) Define the i th residual as $e_i = (\vec{y} - X\hat{\beta})_i$. The residuals are zero-mean gaussian but are correlated, with covariance matrix $\sigma^2(Id_I - H)$, where $H = X(X^T X)^{-1} X^T$.*

Let $s^2 = \frac{1}{I-p} \|e\|^2 = \frac{1}{I-p} \sum_i e_i^2$ (rescaled sum of squared residuals). s^2 is an unbiased estimator of σ^2 .

The standardized residuals defined by $r_i := \frac{e_i}{s\sqrt{1-H_{i,i}}}$ have unit variance and $r_i \sim t_{I-p-1}$. This can be used to test the model by checking that r_i are approximately normal with unit variance.

4. *(Confidence Intervals) Let $G = (X^T X)^{-1} = KK^T$; the distribution of $\hat{\beta}$ is gaussian with mean $\vec{\beta}$ and covariance matrix $\sigma^2 G$, and $\hat{\beta}$ is independent of e .*

*In particular, assume we want a confidence interval for a (non-random) linear combination of the parameters $\gamma = \sum_{j=1}^p u_j \beta_j$; $\hat{\gamma} = \sum_j u_j \hat{\beta}_j$ is our estimator of γ . Let $g = \sum_{j,k} u_j G_{j,k} u_k = \sum_k \left(\sum_j u_j K_{j,k} \right)^2$ (g is called the **variance bias**). Then $\frac{\hat{\gamma} - \gamma}{\sqrt{g}s} \sim t_{I-p}$. This can be used to obtain a confidence interval for γ .*

Comments. Item 4 is often used as follows : if we ignore the uncertainty due to the estimation of σ , the estimation error (in estimating $\vec{\beta}$) is approximately gaussian with covariance matrix G (sometimes called the “variance-covariance matrix”).

Item 3 states that the residuals are (slightly) biased, and it is better to use standardized residuals.

The matrix H is the projection onto the subspace spanned by the columns of X (Eq.(C.3) on Page 329). The predicted response is $\hat{y} = X\hat{\beta}$. It is equal to the orthogonal projection of \vec{y} , and is

equation

$$X \begin{pmatrix} a \\ b \\ d \end{pmatrix} = 0$$

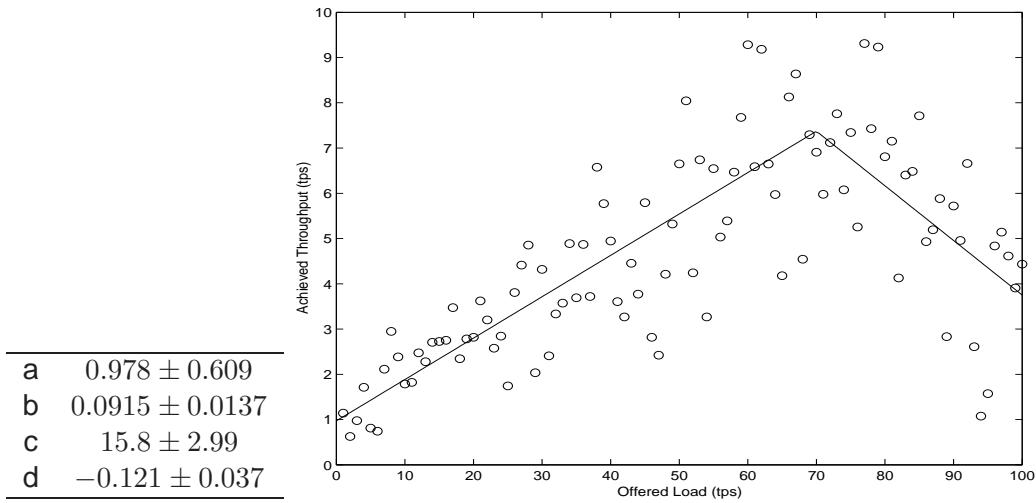
has only the solution $a = b = d = 0$. Consider first a and b . If there are two distinct values of x_i , $i \leq i^*$, say x_1 and x_2 then $a + bx_1 = a + bx_2 = 0$ thus $a = b = 0$. Since there is a value $x_i > \xi$, it follows that $i^* + 1 \leq I$ and $d(x_I - \xi) = 0$ thus $d = 0$.

given by

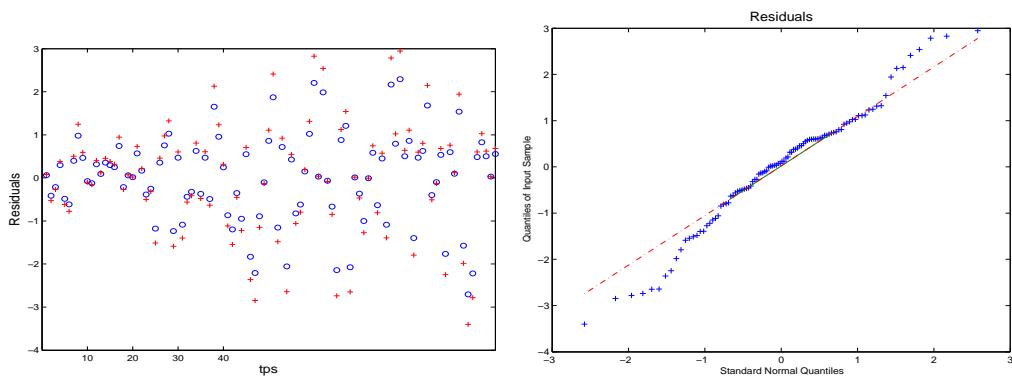
$$\hat{y} = H\vec{y} \quad (3.11)$$

The scaled sum of squared residuals s^2 is also equal to $\frac{1}{I-p} (\|\vec{y}\|^2 - \|\hat{y}\|^2)$. Its distribution is $\frac{1}{I-p} \chi_{I-p}^2$. This can be used to compute a confidence interval for σ .

EXAMPLE 3.7: JOE'S SHOP AGAIN. CONTINUATION OF EXAMPLE 3.6. We can thus apply matrix computations given in Theorem 3.3; item 2 gives an estimate of (a, b, d) and thus of c . Item 4 gives confidence intervals. The values and the fitted linear regression model are shown in the table and figure below.



We also computed the residuals e_i (crosses) and standardized residuals r_i (circles). There is little difference between both types of residuals. They appear reasonably normal, but one might criticize the model in that the variance appears smaller for smaller values of x . The normal qqplot of the residuals also shows approximate normality (the qqplot of standardized residuals is similar and is not shown).



QUESTION 3.2.2. *Can we conclude that there is congestion collapse ?*³

³Yes, since the confidence interval for d is entirely positive [resp. negative].

WHERE IS LINEARITY ? In the previous example, we see that that y_i is a linear function of $\vec{\beta}$, but **not of x_i** . This is quite general, and you should avoid a widespread confusion: linear regression is not restricted to models where the data y_i is linear with the explanatory variables x_i .

EXAMPLE 3.8: JOE'S SHOP - BEYOND THE LINEAR CASE - ESTIMATION OF ξ . In Example 3.6 we assumed that the value ξ after which there is congestion collapse is known in advance. Now we relax this assumption. Our model is now the same as Eq.(3.9), except that ξ is also now a parameter to be estimated.

To do this, we apply maximum likelihood estimation. We have to maximize the log-likelihood $l_{\vec{y}}(a, b, d, \xi, \sigma)$, where \vec{y} , the data, is fixed. For a fixed ξ , we know the value of (a, b, d, σ) that achieves the maximum, as we have a linear regression model. We plot the value of this maximum versus ξ (Figure 3.2) and numerically find the maximum. It is for $\xi = 77$.

To find a confidence interval, we use the asymptotic result in Theorem B.2. It says that a 95% confidence interval is obtained by solving $l(\hat{\xi}) - l(\xi) \leq 1.9207$, which gives $\xi \in [73, 80]$.

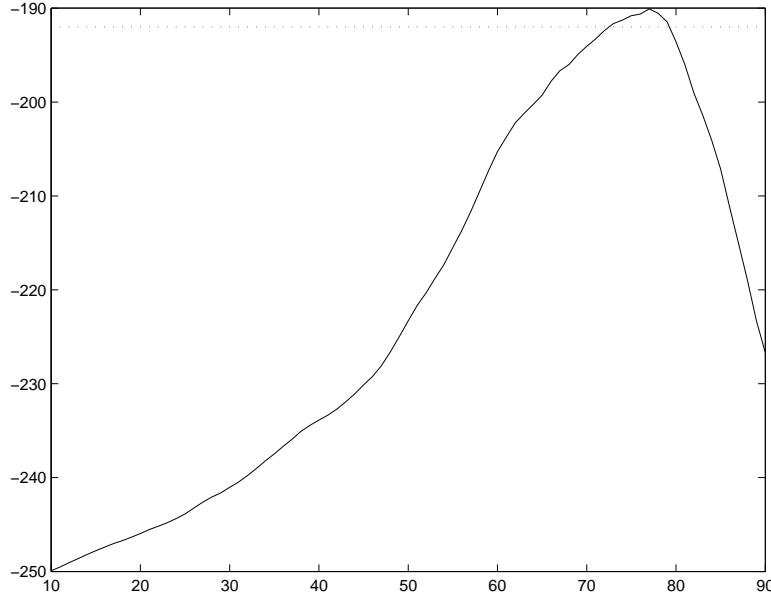


Figure 3.2: Log likelihood for Joes' shop as a function of ξ .

3.3 LINEAR REGRESSION WITH ℓ^1 NORM MINIMIZATION

This is a variant of the linear regression model, but with Laplace instead of Gaussian noise. The theory is less simple, as we do not have explicit linear expressions. Nonetheless, it uses linear programming and is thus often tractable, with the benefit of more robustness to outliers.

The *statistical model* of our experiment has the form:

DEFINITION 3.2 (Linear Regression Model with Laplace Noise).

$$Y_i = (X\vec{\beta})_i + \epsilon_i \text{ for } i = 1, \dots, I \text{ with } \epsilon_i \text{ iid } \sim \text{Laplace}(\lambda) \quad (3.12)$$

where the unknown parameter $\vec{\beta}$ is in \mathbb{R}^p and X is a $I \times p$ matrix. The matrix X supposed to be known exactly in advance. As in Section 3.2, we assume that X has rank p , otherwise the model is non identifiable.

The following is an almost immediate consequence of Theorem 3.2.

THEOREM 3.4. Consider the model in Definition 3.1; let \vec{y} be the $I \times 1$ column vector of the data. The maximum likelihood estimator of $\vec{\beta}$ is obtained by solving the linear program:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^I u_i \\ \text{over} & \vec{\beta} \in \mathbb{R}^p, u \in \mathbb{R}^I \end{array}$$

$$\begin{array}{ll} \text{subject to the constraints} & u_i \geq y_i - (X\vec{\beta})_i \\ & u_i \geq -y_i + (X\vec{\beta})_i \end{array}$$

The maximum likelihood estimator of the noise parameter λ is $\left(\frac{1}{I} \sum_{i=1}^I |y_i - (X\vec{\beta})_i|\right)^{-1}$.

In view of Example 3.5, there is little hope to obtain nice closed form formulas for confidence intervals, unlike what happens with the least square method in Theorem 3.3, and indeed the theorem does not give any. To compute confidence intervals, we can use the bootstrap, with re-sampling from residuals, as described in Algorithm 2.

Algorithm 2 The Bootstrap with Re-Sampling From Residuals. The goal is to compute a confidence interval for some function $\varphi(\vec{\beta})$ of the parameter of the model in Definition 3.1. r_0 is the algorithm's accuracy parameter.

- 1: $R = \lceil 2r_0/(1-\gamma) \rceil - 1$ ▷ For example $r_0 = 25, \gamma = 0.95, R = 999$
 - 2: estimate $\vec{\beta}$ using Theorem 3.4; obtain $\hat{\beta}$
 - 3: compute the residuals $e_i = y_i - (X\hat{\beta})_i$
 - 4: **for** $r = 1 : R$ **do** ▷ Re-sample from residuals
 - 5: draw I numbers with replacement from the list (e_1, \dots, e_I) and call them E_1^r, \dots, E_I^r
 - 6: generate the bootstrap replicate Y_1^r, \dots, Y_I^r from the estimated model:
 - 7: $Y_i^r = (X\hat{\beta})_i + E_i^r$ for $i = 1 \dots I$
 - 8: re-estimate $\vec{\beta}$, using Y_i^r as data, using Theorem 3.4; obtain $\vec{\beta}^r$
 - 9: **end for**
 - 10: $(\varphi_{(1)}, \dots, \varphi_{(R)}) = \text{sort}(\varphi(\vec{\beta}^1), \dots, \varphi(\vec{\beta}^R))$
 - 11: confidence interval for $\varphi(\vec{\beta})$ is $[\varphi_{(r_0)} ; \varphi_{(R+1-r_0)}]$
-

Note that the algorithm applies to any model fitting method, not just to models fitted with Theorem 3.4. As always with the bootstrap, it provides approximate confidence intervals, with a tendency to underestimate.

EXAMPLE 3.9: JOE'S SHOP WITH ℓ^1 NORM MINIMIZATION. We revisit Example 3.6 and estimate a piecewise linear throughput response (as in Eq.(3.9)) with ℓ^1 norm minimization, i.e. assuming the error terms ϵ_i come from a Laplace distribution.

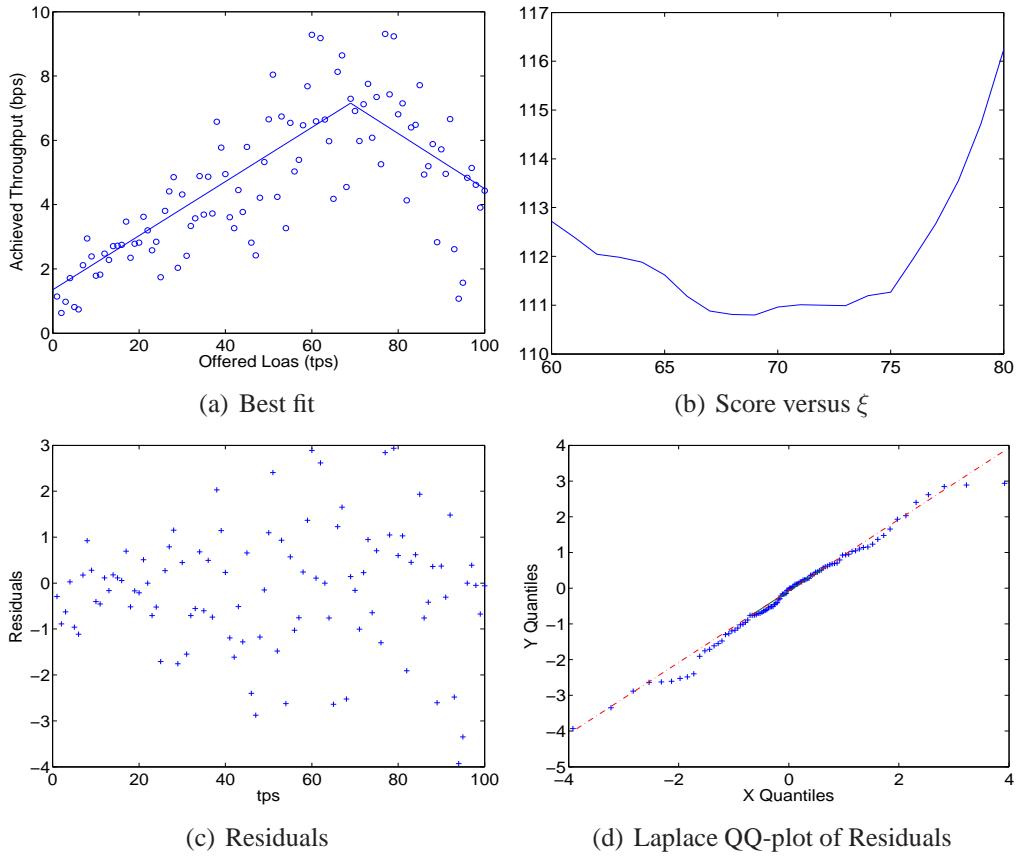


Figure 3.3: Modelling congestion collapse in Joe's shop with a piecewise linear function and ℓ^1 norm minimization of the errors.

The problem is linear and has full rank if we take as parameter for example (a, b, c) , but it is not linear with respect to ξ . To overcome this issue, we first estimate the model, considering ξ as fixed, using linear programming. Then we vary ξ and look for the value of ξ that maximizes the likelihood.

In Figure 3.3(b) we plot ξ versus the score (ℓ^1 norm of the error). By Theorem 3.2, maximizing the likelihood is the same as minimizing the score. The optimal is for $\xi = 69$ (but notice that the score curve is very flat, so any value around 70 would be just as good). For this value of ξ , the estimated parameters are: $\hat{a} = 1.35$, $\hat{b} = 0.0841$, $\hat{c} = 13.1$, $\hat{d} = -0.0858$. We compute the residuals (Figure 3.3(c)) and do a Laplace qq-plot to verify the model assumption.

As explained in Section 2.4.3, a *Laplace qq-plot* of the residuals r_i , $i = 1 \dots I$ is obtained by plotting $F^{-1}(\frac{i}{I+1})$ versus the residuals $r_{(i)}$ sorted in increasing order. Here F is the CDF of the Laplace distribution with rate $\lambda = 1$. A direct computation gives

$$\begin{aligned} F^{-1}(q) &= \ln(2q) \text{ if } 0 \leq q \leq 0.5 \\ &= -\ln(2(1-q)) \text{ if } 0.5 \leq q \leq 1 \end{aligned}$$

Figure 3.3(d) shows the Laplace qq-plot of the residuals; there is a better fit than with least squares (Example 3.7).

We compute 95% confidence intervals for the parameters using the bootstrap (Algorithm 2) and obtain:

a	1.32 ± 0.675
b	0.0791 ± 0.0149
c	11.7 ± 3.24
d	-0.0685 ± 0.0398

The parameter of interest is d , for which the confidence interval is entirely negative, thus there is congestion collapse.

3.4 CHOOSING A DISTRIBUTION

Assume we are given a data set in the form of a sequence of numbers and would like to fit it to a distribution. Often, the data set is iid, but not always. In this section and the next, we review a number of simple guidelines that are useful for finding the right distribution. We illustrate in the next section how this can be used to build a load generator (SURGE).

In this section and the next, a distribution means a probability distribution on the set of real numbers.

3.4.1 SHAPE

Perhaps the first attribute of interest is the shape of the distribution, or more precisely, of its PDF. We say that two distributions on \mathbb{R} , with CDFs $F()$ and $G()$, have the same *distribution shape* if they differ by a change of scale and location, i.e., there exist some $m \in \mathbb{R}$ and $s > 0$ such that $G(sx + m) = F(x)$ for all $x \in \mathbb{R}$. This is equivalent to saying that there are some random variables X, Y with distribution functions $F(), G()$ respectively, and with $Y = sX + m$.

For example, the normal distribution N_{μ, σ^2} and the standard normal distribution $N_{0,1}$ have the same shape, in other words, all normal distributions are essentially the same.

When looking for a distribution, one may get a first feeling plotting a histogram, which is a coarse estimate of the PDF. Since most plotting tools automatically adapt the scales and origins on both axes, what one really gets is a coarse estimate of the distribution shape.

A distribution is usually defined with a number of parameters. When browsing a distribution catalog (e.g. on Wikipedia) it is important to distinguish among those parameters that influence the shape and those that are simply location and scale parameters. For example, with the normal distribution N_{μ, σ^2} , μ is a location parameter and σ a scale parameter; if a random variable X has distribution N_{μ, σ^2} , one can write $X = \sigma Z + \mu$, where $Z \sim N_{0,1}$.

In Tables 3.1 and 3.2 we give a small catalog of distributions that are often used in the context of this book. For each distribution we give only the set of parameters that influence the shape. Other distributions can be derived by a change of location and scale. The effect of this on various formulas is straightforward but is indicated in the table as well, for completeness.

The *log-normal distribution* with parameters $\mu, \sigma > 0$ is defined as the distribution of $X = e^Z$ where Z is gaussian with mean μ and variance σ^2 . It is often used as a result of rescaling in log scale, as we did in Eq.(3.2). Note that

$$X = e^{\sigma Z_0 + \mu} = e^\mu (e^{Z_0})^\sigma \text{ with } Z_0 \sim N_{0,1}$$

thus μ corresponds to a scale parameter $s = e^\mu$. In contrast (unlike for the normal distribution), σ is a shape parameter. Table 3.1 gives properties of the standard log-normal distribution (i.e. for $\mu = 0$; other values of μ can be obtained by re-scaling). Figure 3.4 shows the shape of the log-normal distribution for various values of σ , rescaled such that the mean is constant equal to 1.

QUESTION 3.4.1. *What are the parameters μ, σ of the lognormal distributions in Figure 3.4?*⁴

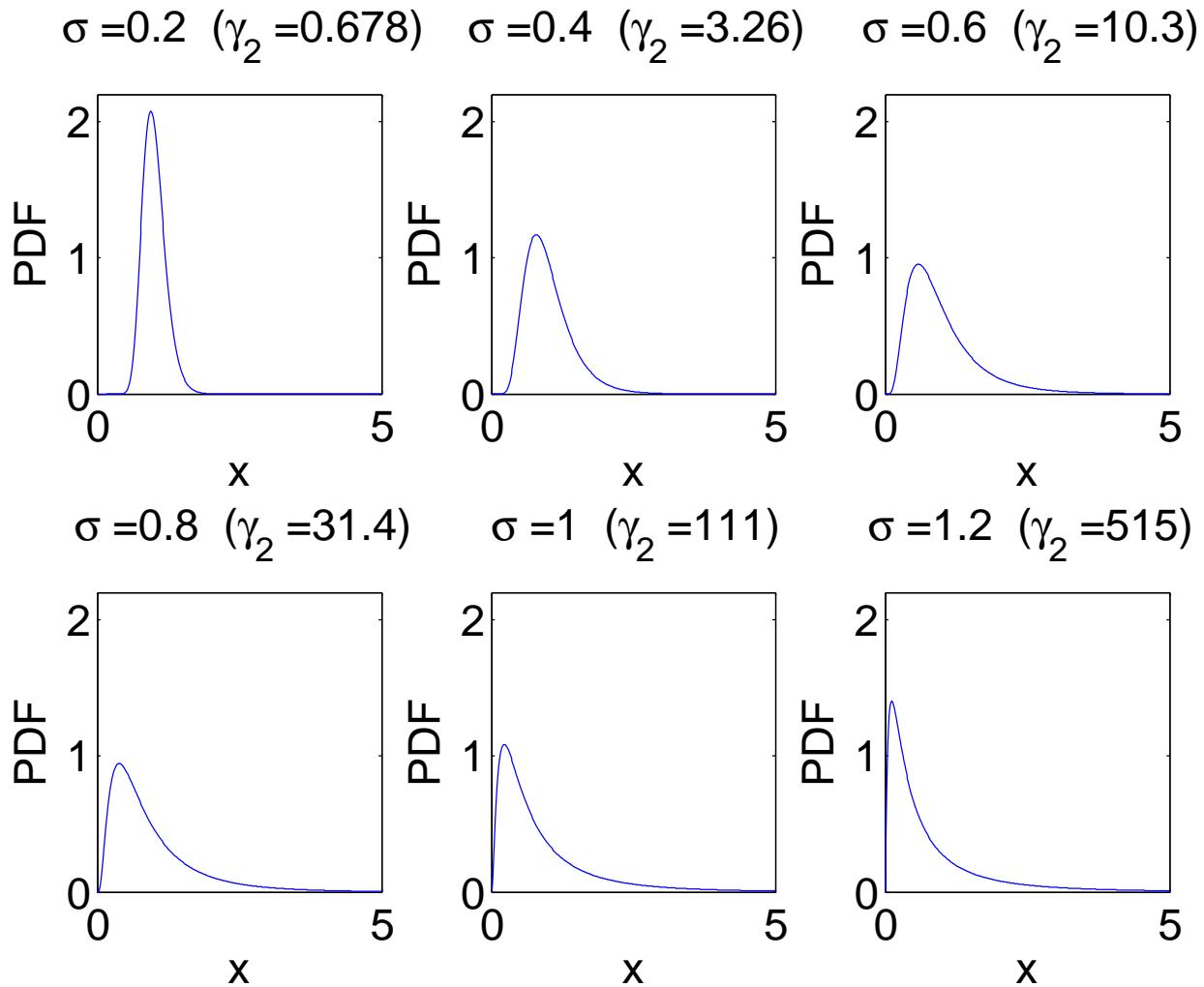


Figure 3.4: Shape of the log-normal distribution for various values of σ . The shape is independent of μ . μ is chosen such that the mean is 1 for all plots. γ_2 is the Kurtosis index.

3.4.2 SKEWNESS AND KURTOSIS

These are indices which may be used to characterize a distribution shape. They are defined for a distribution that has finite moments up to order 4. The definition uses the [cumulant generating](#)

⁴By Table 3.1 the mean is $e^{\frac{\sigma^2}{2}}$ when $\mu = 0$; other values of μ correspond to re-scaling by e^μ therefore the mean is $e^{\frac{\sigma^2}{2} + \mu}$. In the figure we take the mean equal to 1, thus we must have $\mu = -\frac{\sigma^2}{2}$

function of the distribution of a real random variable X : defined by

$$\text{cgf}(s) := \ln \mathbb{E}(e^{sX})$$

Assume that $\mathbb{E}(e^{s_0|X|}) < \infty$ for some s_0 so that the above is well defined for real s around $s = 0$. This also implies that all moments are finite. Then, by a Taylor expansion:

$$\text{cgf}(s) = \kappa_1 s + \kappa_2 \frac{s^2}{2} + \kappa_3 \frac{s^3}{3!} + \dots + \kappa_k \frac{s^k}{k!} + \dots$$

where $\kappa_k = \frac{d^k}{ds^k} \text{cgf}(0)$ is called the **cumulant of order k** . The first four cumulants are :

$$\begin{cases} \kappa_1 = \mathbb{E}(X) \\ \kappa_2 = \mathbb{E}(X - \mathbb{E}(X))^2 = \text{var}(X) \\ \kappa_3 = \mathbb{E}(X - \mathbb{E}(X))^3 \\ \kappa_4 = \mathbb{E}(X - \mathbb{E}(X))^4 - 3\text{var}(X)^2 \end{cases} \quad (3.13)$$

For the normal distribution N_{μ, σ^2} , $\text{cgf}(s) = \mu s + \frac{\sigma^2}{2}s^2$ thus all cumulants of order $k \geq 3$ are 0.

QUESTION 3.4.2. Show that the k th cumulant of the convolution of n distributions is the sum of the k th cumulants⁵

SKEWNESS INDEX κ_3 is called **skewness**. The **skewness index** (sometimes also called skewness) is

$$\gamma_1 := \kappa_3 / \kappa_2^{3/2} = \kappa_3 / \sigma^3$$

The skewness index is insensitive to changes in scale (by a positive factor) or location. For a density which is symmetric around its mean, $\kappa_{2k+1} = 0$; γ_1 can be taken as a measure of asymmetry of the distribution. When $\gamma_1 > 0$ the distribution is right-skewed, and vice-versa. If ϕ is convex, then $\phi(X)$ has greater skewness index than X .

KURTOSIS INDEX κ_4 is called Kurtosis. The **kurtosis index**, also called **excess kurtosis**, is

$$\gamma_2 := \kappa_4 / \kappa_2^2 = \kappa_4 / \sigma^4$$

The Kurtosis index is insensitive to changes in scale or location. It is used to measure departure from the normal distribution. When $\gamma_2 > 0$, the distribution has a sharper peak around the mean and heavier tail; when $\gamma_2 < 0$, it has a flatter top and decays more abruptly. Note that $\gamma_2 \geq -2$, with equality only if the distribution is degenerate, i.e. equal to a constant.

The kurtosis index gives some information about the distribution tail. When large and positive it indicates that the contribution of the tail is large. We see for example in Figure 3.4 and in Table 3.1 that the log-normal distribution has larger tail for larger σ .

⁵By independence: $\ln \mathbb{E}(e^{s(X_1 + \dots + X_n)}) = \sum_i \ln \mathbb{E}(e^{sX_i})$.

3.4.3 POWER LAWS, PARETO DISTRIBUTION AND ZIPF'S LAW

Power laws are often invoked in the context of workload generation. Generally speaking, a power law is any relation of the form $y = ax^b$ between variables x and y , where a and b are constants. In log scales, this gives a linear relationship: $\ln y = b \ln x + \ln a$. Power laws were often found to hold, at least approximately, for the *complementary CDFs*⁶ of some variables such as file sizes or popularity of objects. They are discovered by plotting the empirical complementary CDF in log-log scales and seeing if a linear relationship exists. Depending on whether the distribution is continuous or discrete, we obtain the Pareto and Zeta distributions.

The standard *Pareto* distribution with index $p > 0$ has CDF and PDF

$$\begin{aligned} F(x) &= \left(1 - \frac{1}{x^p}\right) \mathbf{1}_{\{x \geq 1\}} \\ f(x) &= \frac{p}{x^{p+1}} \mathbf{1}_{\{x \geq 1\}} \end{aligned}$$

i.e. the complementary CDF and the PDF follow a power law for $x \geq 1$ (see Table 3.2). The general Pareto distribution is derived by a change of scale and has CDF $\left(1 - \frac{s^p}{x^p}\right) \mathbf{1}_{\{x \geq s\}}$ and PDF $\frac{ps^p}{x^{p+1}} \mathbf{1}_{\{x \geq s\}}$ for some $s > 0$.

The *Zeta* distribution is the integer analog of Pareto. It is defined for $n \in \mathbb{N}$ by $\mathbb{P}(X = n) = \frac{1}{n^{p+1} \zeta(p+1)}$, where $\zeta(p+1)$ is a normalizing constant (Riemann's zeta function).

For $p < 2$ the Pareto distribution has infinite variance and for $p < 1$ infinite mean. The kurtosis index is not defined unless $p > 4$ and tends to ∞ when $p \rightarrow 4$: its tail is called "heavy", (see Section 3.5). Figure 3.4.3 shows the CDF of a Pareto distribution together with normal and log-normal distributions.

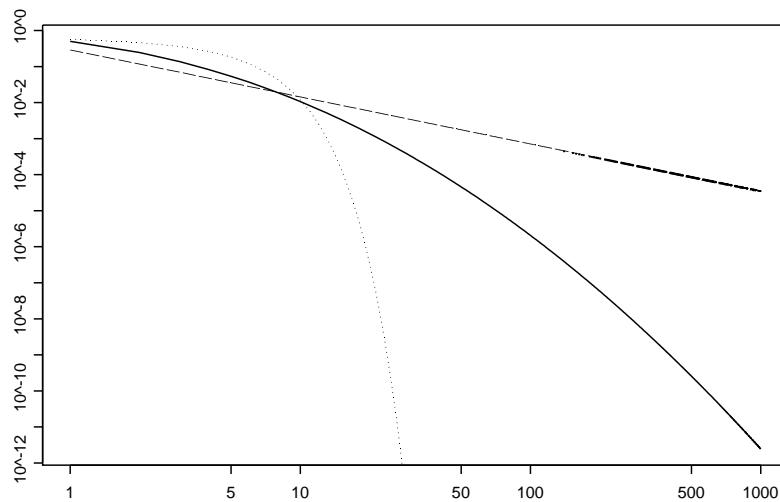


Figure 3.5: $P(X > x)$ versus x on log-log scales, when X is normal (dots), log-normal (solid) or Pareto (dashes). The three distributions have same mean and 99%-quantile.

⁶The complementary CDF is $1 - F()$ where $F()$ is the CDF.

Zipf's law is not a probability distribution, but is related to the Pareto distribution. It states that the popularity of objects is inversely proportional to rank, or more generally, to a power of rank. This can be interpreted as follows.

We have a collection of N objects. We choose an object from the collection at random, according to some stationary process. Call θ_j the probability that object j is chosen; this is our interpretation of the popularity of object j .

Let $\theta_{(1)} \geq \theta_{(2)} \geq \dots$ be the collection of θ s in decreasing order. Zipf's law means

$$\theta_{(j)} \approx \frac{C}{j^\alpha}$$

where C is some constant and $\alpha > 0$. In Zipf's original formulation, $\alpha = 1$.

Now we show the relation to a Pareto distribution. Assume that we draw the θ s at random (as we do in a load generator) by obtaining some random value X_i for object i , and letting $\theta_i = X_i / (\sum_{i=1}^N X_i)$. Assume that the number of objects is large and X_i 's marginal distribution is some fixed distribution on \mathbb{R}^+ , with complementary distribution function $G(x)$. Let $X_{(n)}$ be the reverse order statistic, i.e. $X_{(1)} \geq X_{(2)} \geq \dots$. We would like to follow Zipf's law, i.e., for some constant C :

$$X_{(j)} \approx \frac{C}{j^\alpha} \quad (3.14)$$

Now let us look at the empirical complementary distribution \hat{G} ; it is obtained by putting a point at each X_i , with probability $1/N$, where N is the number of objects. More precisely:

$$\hat{G}(x) = \frac{1}{N} \sum_{i=1}^N \mathbf{1}_{\{X_i \geq x\}}$$

Thus $\hat{G}(X_{(j)}) = j/N$. Combine with Eq.(3.14): we find that, whenever $x = X_{(j)}$, we have $\hat{G}(x) \approx \frac{K}{x^p}$, with $p = \frac{1}{\alpha}$ and $K = c^p/N$. If we take the empirical complementary CDF as approximation of the true complementary CDF, this means that the distribution of X_i is Pareto with index $p = \frac{1}{\alpha}$.

In other words, Zipf's law can be interpreted as follows. The probability of choosing object i is itself a random variable, obtained by drawing from a Pareto distribution with tail index $p = \frac{1}{\alpha}$, then re-scaling to make the probabilities sum to 1.

3.4.4 HAZARD RATE

The hazard rate provides another means of deciding whether a distribution is well suited. Consider a distribution with support that includes $[a, +\infty)$ for some a , with a PDF $f()$ and with CDF $F()$. The **hazard rate** is defined for $x > a$ by

$$\lambda(x) = \frac{f(x)}{1 - F(x)}$$

It can be interpreted as follows. Let X be a random variable with distribution $F()$. Then, for $x > a$

$$\lambda(x) = \lim_{dx \rightarrow 0} \frac{1}{dx} \mathbb{P}(X \leq x + dx | X > x)$$

If X is interpreted as a flow duration or a file size, $\lambda(x)dx$ is the probability that the flow ends in the next dx time units given that it survived until now. See Tables 3.1 and 3.2 for the hazard rates of several distributions.

The behaviour of the hazard rate $\lambda(x)$ when x is large can be used as a characteristic of a distribution. Qualitatively, one may distinguish the following three types of behaviour:

1. (Aging Property) $\lim_{x \rightarrow \infty} \lambda(x) = \infty$: the hazard rate becomes large for large x . This is very often expected, e.g. when one has reasons to believe that a file or flow is unlikely to be arbitrarily large. If X is interpreted as system lifetime, this is the property of aging. The gaussian distribution is in this case.
2. (Memoriless Property) $\lim_{x \rightarrow \infty} \lambda(x) = c > 0$: the hazard rate tends to become constant for large x . This is in particular true if the system is memoriless, i.e. when $\lambda(x)$ is a constant. The exponential distribution is in this case (as is the Laplace distribution).
3. (*Fat Tail*) $\lim_{x \rightarrow \infty} \lambda(x) = 0$: the hazard rate vanishes for large x . This may appear surprising: for a flow duration, it means that, given that you waited a large time for completion of the flow, you are likely to continue waiting for a very long time. The Pareto distribution with index p is in this case for all values of p , as are all lognormal distributions. We may, informally, call this property a “fat tail”. Heavy tail distributions (defined in Section 3.5) are in this case, but there are also some non heavy tail distributions as well.

The *Weibull distribution* is often used in this context, as it spans the three cases, depending on its parameters. The standard Weibull distribution with exponent c has support on $[0, \infty)$ and is defined by its CDF equal to $1 - e^{-(x^c)}$. The general Weibull distribution is derived by a change of scale and location; also see Tables 3.1 and 3.2. For $c = 1$ it is the exponential distribution; for $c > 1$ it has the aging property and for $c < 1$ it is fat tailed. Figure 3.6 shows the shape of the Weibull distributions. The Kurtosis is minimum at $c \approx 3.360128$ and goes to ∞ as $c \rightarrow 0$ [87].

3.4.5 FITTING A DISTRIBUTION

Fitting a distribution to a dataset is often a two step process. First, a qualitative analysis is performed, where one attempts to get a feeling for the distribution shape. Here, one tries to make statements about the distribution shape, the hazard rate or the existence of power laws. These are obtained by appropriate plots (histograms, qq-plots, empirical CDFs, etc). One can also try to determine whether a heavy tailed distribution is the right model, using for example the `aest` tool described in Section 3.5. The goal is to obtain a set of candidate families of distributions.

The second step is to fit the parameters of the distribution. If the data set can be assumed to come from an iid sequence, the method of choice is maximum likelihood estimation (MLE), as explained in Section B.1.2, and illustrated in the next example. In particular, MLE is invariant by re-parametrization and change of scale.

If, as is frequent in practice, the data set may not be assumed to come from an iid sequence, then there is no simple method; maximum likelihood estimation is often used in practice (but no confidence interval for the estimated parameters can be obtained).

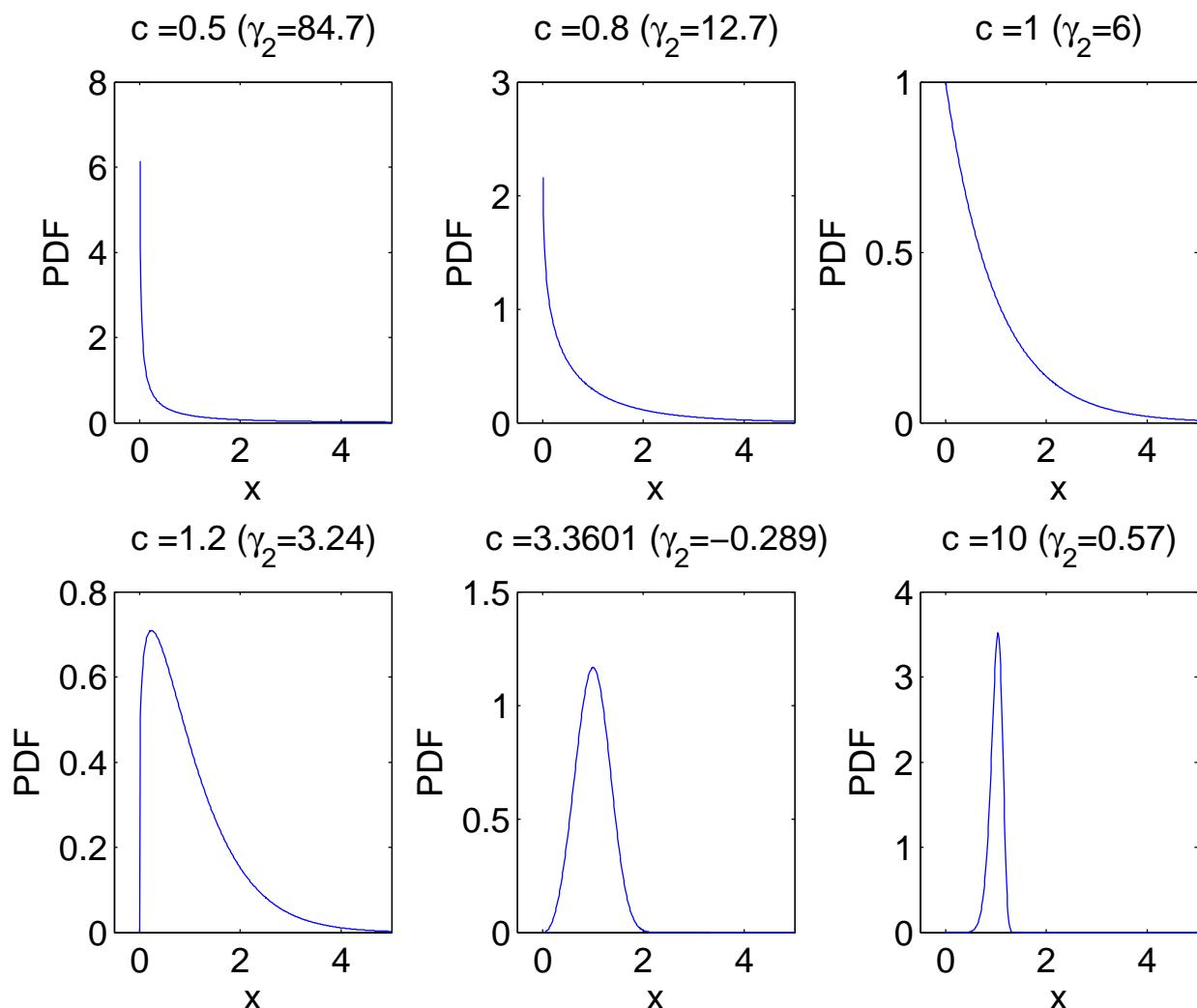


Figure 3.6: Shape of the Weibull distribution for various values of the exponent c . The distribution is re-scaled to have mean = 1. γ_2 is the Kurtosis index.

3.4.6 CENSORED DATA

When fitting the distribution parameters, it may be important to account for the fact that some very large or very small data values are not present, due to impossibilities of the measurement system (for example, flow size durations may not measure very long flows). This is called *censoring* in statistics.

A technique for accounting for censoring is as follows. Assume we know that the data is truncated to some maximum, called a . The distribution for the data can be described by the PDF

$$f_X(x) = \frac{1}{F_0(a)} f_0(x) \mathbf{1}_{\{x \leq a\}} \quad (3.15)$$

where f_0 [resp. F_0] is the PDF [resp. CDF] of the non truncated distribution. The reason for Eq.(3.15) lies in the theory of rejection sampling (Section 6.6.2) which says that when one rejects the data samples that do not satisfy a condition (here $X \leq a$) one obtains a random variable with PDF proportional to the non censored PDF, restricted to the set of values given by the condition. The term $\frac{1}{F_0(a)}$ is the normalizing constant.

Assume that the non truncated distribution F_0 depends on some parameter θ . The log likelihood of the data x_1, \dots, x_n is

$$\ell(\theta, a) = \sum_{i=1}^n \log f_0(x_i | \theta) - n \log F_0(a | \theta) \quad (3.16)$$

We obtain an estimate of θ and a by maximizing Eq.(3.16). Note that we must have $a \geq \max_i x_i$ and for any θ , the likelihood is nonincreasing with a . Thus the optimal is for $\hat{a} = \max_i x_i$.

It remains to maximize $\ell(\theta, \hat{a})$ over θ . This can be done by brute force when the dimensionality of the parameter θ is small, or using other methods, as illustrated in the next example.

EXAMPLE 3.10: CENSORED LOG-NORMAL DISTRIBUTION. Figure 3.7(a) shows an artificial data set, obtained by sampling a log-normal distribution with parameters $\mu = 9.357$ and $\sigma = 1.318$, truncated to 20000 (i.e. all data points larger than this value are removed from the data set).

Here, F_0 is the log-normal distribution with parameters μ and σ . Instead of brute force optimization, we can have more insight as follows. We have to maximize $\ell(\mu, \sigma)$ over $\mu \in \mathbb{R}$, $\sigma > 0$, with

$$\begin{aligned} \ell(\mu, \sigma) &= -n \ln(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^n (\ln x_i - \mu)^2 - n \ln N_{0,1}(\mu + \sigma \ln a) \\ &\quad - \frac{n}{2} \ln(2\pi) - \sum_{i=1}^n \ln x_i \end{aligned} \quad (3.17)$$

We can ignore the last two terms, which do not depend on (μ, σ) . We can also do a change of variables by taking as parameters σ, z instead of σ, μ , with

$$z = \frac{\ln a - \mu}{\sigma} \quad (3.18)$$

For a fixed z , the optimization problem has a closed form solution (obtained by computing the derivative with respect to σ); the maximum likelihood is obtained for $\sigma = \hat{\sigma}(z)$ with

$$\hat{\sigma}(z) = \frac{-\beta z + \sqrt{4s^2 + \beta^2(4 + z^2)}}{2} \quad (3.19)$$

$$\text{with } \beta = \ln a - y_1, \quad y_1 = \frac{1}{n} \sum_{i=1}^n \ln x_i, \quad s^2 = \frac{1}{n} \sum_{i=1}^n (\ln x_i - y_1)^2$$

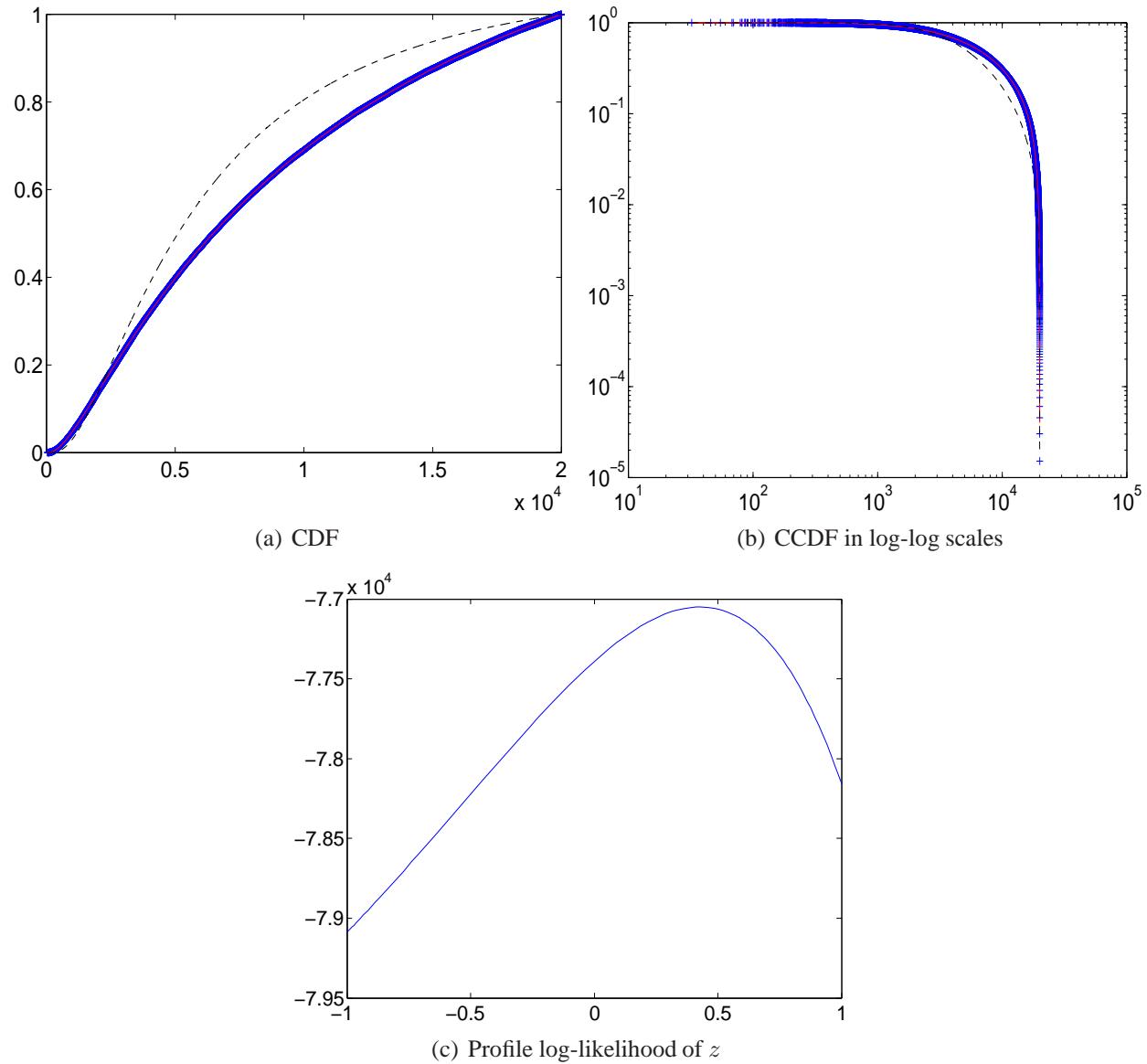


Figure 3.7: Fitting Censored Data in Example 3.10. The data set is an iid sample of a truncated log-normal distribution. Thick lines: data set; plain lines: fit obtained with a technique for censored data; dashed lines: fit obtained when ignoring the censored data.

and the corresponding value of the likelihood (called “profile log-likelihood”) is (we omit the constant terms in Eq.(3.17)):

$$pl(z) = -n \left[\ln(\hat{\sigma}(z)) - \frac{1}{2\hat{\sigma}^2(z)} \left((\hat{\sigma}(z)z - \beta)^2 + s^2 \right) - \ln N_{0,1}(z) \right] \quad (3.20)$$

We need now to minimize the square bracket as a function of $z \in \mathbb{R}$. This cannot be done in closed form, but it is numerically simple as it is a function of one variable only. Figure 3.7(c) shows $pl(z)$. There is a unique maximum at $z = 0.4276$, which, with Eq.(3.19) and Eq.(3.18), gives

$$\hat{\mu} = 9.3428 \quad \hat{\sigma} = 1.3114$$

Compare to the method that would ignore the truncation. Since MLE is invariant by change of scale we can use the log of the data; we would estimate μ by the sample mean of the log of the data, and σ by the standard deviation, and would obtain

$$\hat{\mu}_n = 8.6253 \quad \hat{\sigma}_n = 0.8960$$

3.4.7 COMBINATIONS OF DISTRIBUTIONS

It is often difficult to find a distribution that fits both the tail and the body of the data. In such case, one may use a combination of distributions, also called *compound distribution*.

Given two distributions with CDFs F_1 and F_2 [resp. PDFs f_1 and f_2], a *mixture distribution* of F_1 and F_2 is a distribution with PDF

$$f(x) = qf_1(x) + (1 - q)f_2(x)$$

with $q \in [0, 1]$. A mixture is interpreted by saying that a sample is drawn with probability q from F_1 and with probability $1 - q$ from F_2 .

We are more often interested in a *combination of mixture and truncation*, i.e. in a combination whose PDF has the form

$$f(x) = \alpha_1 \mathbf{1}_{\{x \leq a\}} f_1(x) + \alpha_2 \mathbf{1}_{\{x > a\}} f_2(x) \quad (3.21)$$

where $\alpha_1, \alpha_2 \geq 0$ and $a \in \mathbb{R}$. This is useful for fitting a distribution separately to the tail and the body of the data set. Note that we do not necessarily have $\alpha_1 + \alpha_2 = 1$ as in a pure mixture. Instead, one must have the normalizing condition $\alpha_1 F_1(a) + \alpha_2 (1 - F_2(a)) = 1$, thus (by letting $q = \alpha_1 F_1(a)$) we may rewrite Eq.(3.21) as

$$f(x) = \frac{q}{F_1(a)} \mathbf{1}_{\{x \leq a\}} f_1(x) + \frac{1 - q}{1 - F_2(a)} \mathbf{1}_{\{x > a\}} f_2(x) \quad (3.22)$$

with $q \in [0, 1]$.

Assume the distributions F_1, F_2 depend on some parameters, independent of q , and need to be fitted. Note that q and a need to be fitted as well. If one uses MLE, one can somewhat simplify the fitting by observing that the maximum likelihood estimate must satisfy

$$\hat{q} = \frac{n_1(a)}{n} \quad (3.23)$$

where $n_1(a)$ is the number of data points $\leq a$.

To see why, assume that we are given a data set x_i of n data points, sorted in increasing order, so that $n_1(a) = \sum_{i=1}^n \mathbf{1}_{\{x_i \leq a\}}$. The log-likelihood of the data is

$$\ell = \sum_{i=1}^{n_1(a)} \ln f_1(x_i) + \sum_{i=n_1(a)+1}^n \ln f_2(x_i) + n_1(a) (\ln q - \ln F_1(a)) + (n - n_1(a)) (\ln(1 - q) - \ln(1 - F_2(a)))$$

and maximizing ℓ with respect to q shows Eq.(3.23).

In summary, fitting a compound distribution separately to the body and the tail of a data set is based on fitting Eq.(3.22) to the data, with q given by Eq.(3.23). It remains to fit a and the parameters of F_1 and F_2 . This may be done by: assuming a is known, fitting F_1 and F_2 , and computing the value of a which maximizes the likelihood, as illustrated in the example below.

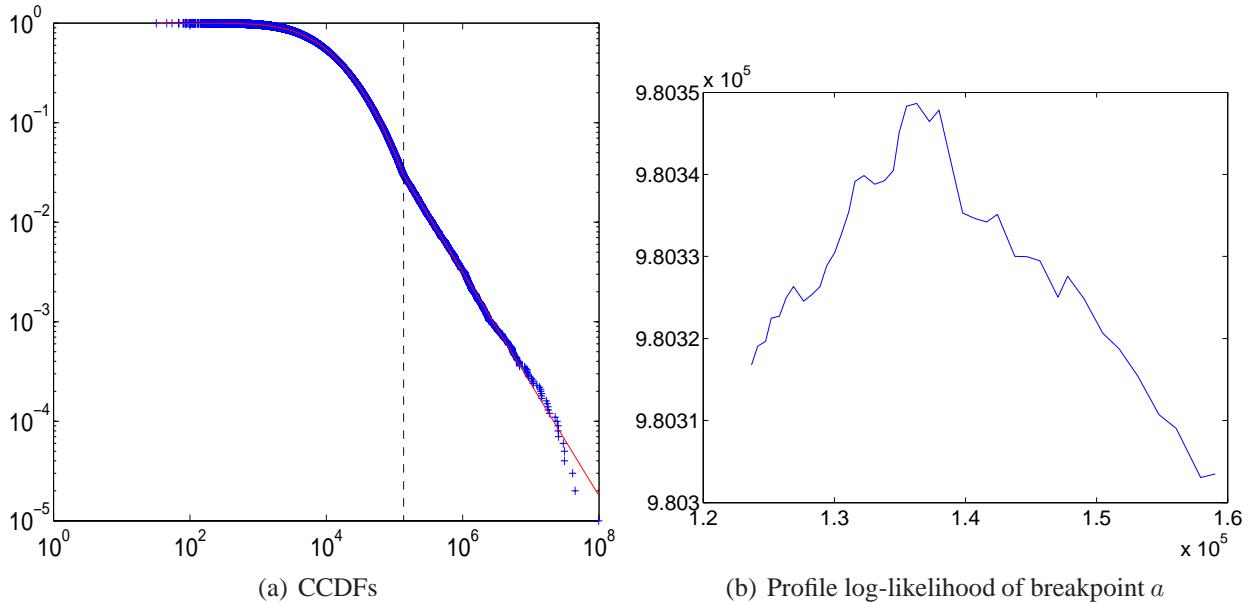


Figure 3.8: Fitting a combination of Log-Normal for the body and Pareto for the tail. Dashed vertical line: breakpoint.

EXAMPLE 3.11: COMBINATION OF LOG-NORMAL AND PARETO. Figure 3.8(a) shows an empirical complementary CDF in log-log scales for a set of 10^5 data points representing file sizes. The plot shows an asymptotic power law, but not over the entire body of the distribution. We wish to fit a combination mixture of truncated log-normal distribution for the body of the distribution, with truncation on $[0, a]$ (left of dashed line) and of a Pareto distribution rescaled to have support on $[a, +\infty)$. The model is thus

$$f_X(x) = q \frac{f_1(x)}{F_1(a)} \mathbf{1}_{\{x \leq a\}} + (1 - q) \frac{f_2(x)}{1 - F_2(a)} \mathbf{1}_{\{x > a\}}$$

where F_1 is a log-normal distribution, F_2 is Pareto with exponent p , and breakpoint a . Note that $F_2(a) = 0$, so the PDF is

$$f_X(x) = \frac{q}{N_{0,1}(\mu + \sigma \ln a)} \frac{1}{\sqrt{2\pi}\sigma x} e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}} \mathbf{1}_{\{0 < x \leq a\}} + (1 - q)p \frac{a^p}{x^{p+1}} \mathbf{1}_{\{x \geq a\}} \quad (3.24)$$

The parameters to be fitted are q, μ, σ, p and the breakpoint a . We first fix a to any arbitrary value and fit the other parameters. By Eq.(3.23), $q = \frac{n_1(a)}{n}$ where $n_1(a)$ is the number of data points $\leq a$. The log-likelihood is thus

$$\begin{aligned}\ell(\mu, \sigma, p, a) &= n_1(a) \ln n_1(a) + n_2(a) \ln n_2(a) - n \ln n \\ &\quad + \ell_1(\mu, \sigma, a) + \ell_2(p, a)\end{aligned}$$

where $n_2(a) = n - n_1(a)$, ℓ_1 is as in Eq.(3.17) (with $n_1(a)$ instead of n) and

$$\ell_2(a, p) = n_2(a) (\ln p + p \ln a) - (p+1) \sum_{i=n_1(a)+1}^n \ln x_i$$

where we assumed that the data x_i is sorted in increasing order. For a fixed a , the optimization of μ, σ on one hand, p on the other, are separated. The optimal $\hat{\mu}(a), \hat{\sigma}(a)$ are obtained as in Example 3.10 using techniques for censored data.

The optimal \hat{p} is obtained directly:

$$\begin{aligned}\max_p \ell_2(a, p) &= n_2(a) (\ln \hat{p}(a) - 1) - \sum_{i=n_1(a)+1}^n \ln x_i \\ \text{with } \hat{p}(a) &= \frac{1}{-\ln a + \frac{1}{n_2(a)} \sum_{i=n_1(a)+1}^n \ln x_i}\end{aligned}\tag{3.25}$$

Putting things together we obtain the profile log-likelihood of a

$$\begin{aligned}pl(a) &= \max_{\mu, \sigma > 0, p > 0} \ell(\mu, \sigma, p, a) \\ &= -n_1(a) \left[\frac{\ln(2\pi)}{2} + \ln(\hat{\sigma}(a)) + \frac{1}{2\hat{\sigma}(a)^2} \left((\hat{\sigma}(a)\hat{z}(a) - \beta(a))^2 + s^2(a) \right) + \ln N_{0,1}(\hat{z}(a)) \right] \\ &\quad + n_2(a) (\ln \hat{p}(a) - 1) - \sum_{i=1}^n \ln x_i\end{aligned}$$

where $\beta(a), \hat{\sigma}(a), s^2(a)$ and $\hat{\mu}(a)$ are as in Example 3.10 and $\hat{z}(a)$ maximizes Eq.(3.20). We determine the maximum of $pl(a)$ numerically Figure 3.8(b) shows that there is some large uncertainty for the value of a , which can be explained by the fact that, in this region, the log-normal distribution locally follows a power law. We find $\hat{a} = 136300$, $\hat{\mu} = 9.3565$, $\hat{\sigma} = 1.3176$ and $\hat{p} = 1.1245$.

3.5 HEAVY TAIL

3.5.1 DEFINITION

In Section 3.4.4 we have seen the definition of fat tail, i.e. a distribution that has vanishing hazard rate. In this section we see an extreme case of fat tail, called “heavy tail”, which has unique, non intuitive features. It is frequently found in models of file sizes and flow durations.

We use the following definition (which is the simplest). We say that the distribution on $[a, \infty)$, with CDF F , is **heavy tailed** with index $0 < p < 2$ if there is some constant k such that, for large x :

$$1 - F(x) \sim \frac{k}{x^p}\tag{3.26}$$

Here $f(x) \sim g(x)$ means that $f(x) = g(x)(1 + \epsilon(x))$, with $\lim_{x \rightarrow \infty} \epsilon(x) = 0$.

A heavy tailed distribution has an infinite variance, and for $p \leq 1$ an infinite mean.

- The Pareto distribution with exponent p is heavy tailed with index p if $0 < p \leq 2$.
- The log-normal distribution is not heavy tailed (its variance is always finite).
- The Cauchy distribution (density $\frac{1}{\pi(1+x^2)}$) is heavy tailed with index 1.

3.5.2 HEAVY TAIL AND STABLE DISTRIBUTIONS

Perhaps the most striking feature of heavy tailed distributions is that the central limit theorem does not hold, i.e. aggregating many heavy tailed quantities does *not* produce a gaussian distribution.

Indeed, if X_i are iid with finite variance σ^2 and with mean μ , then $\frac{1}{n^{\frac{1}{2}}} \sum_{i=1}^n (X_i - \mu)$ tends in distribution to the normal distribution N_{0,σ^2} . In contrast, if X_i are iid, heavy tailed with index p , then there exist constants d_n such that

$$\frac{1}{n^{\frac{1}{p}}} \sum_{i=1}^n X_i + d_n \xrightarrow[n \rightarrow \infty]{\text{distrib}} S_p$$

where S_p has a **stable distribution** with index p . Stable distributions are defined for $0 < p \leq 2$, for $p = 2$ they are the normal distributions. For $p < 2$, they are either constant or heavy tailed with index p . Furthermore, they have a property of closure under aggregation: if X_i are iid and stable with index p , then $\frac{1}{n^{\frac{1}{p}}} (X_1 + \dots + X_n)$ has the same distribution as the X_i s, shifted by some number d_n .

The shape of a stable distribution with $p < 2$ is defined by one skewness parameter $\beta \in [-1, 1]$ (but the skewness index in the sense of Section 3.4.2 does not exist). The *standard* stable distribution is defined by its index p , and when $p < 2$, by β . The general stable distribution is derived by a change of scale and location. When $\beta = 0$ the standard stable distribution is symmetric, otherwise not. The standard stable distribution with skewness parameter $-\beta$ is the symmetric (by change of sign) of the standard stable distribution with parameter β . When $p < 2$ and $\beta = 1$, the support of the stable distribution is $[0, +\infty)$ (and thus when $\beta = -1$ the support is $(-\infty, 0]$), otherwise the support is \mathbb{R} .

Stable distributions that are not constant have a continuous density, which it is not known explicitly, in general. In contrast, their characteristic functions are known explicitly [93, 73], see Table 3.2 . Note that the Pareto distribution is not stable.

Figure 3.9 illustrates the convergence of a sum of iid Pareto random variables to a stable distribution. In practice, stable distributions may be difficult to work with, and are sometimes replaced by heavy tailed combinations, as in Example 3.11.

3.5.3 HEAVY TAIL IN PRACTICE

Heavy tail concretely means that very large outliers are possible. We illustrate this on two examples.

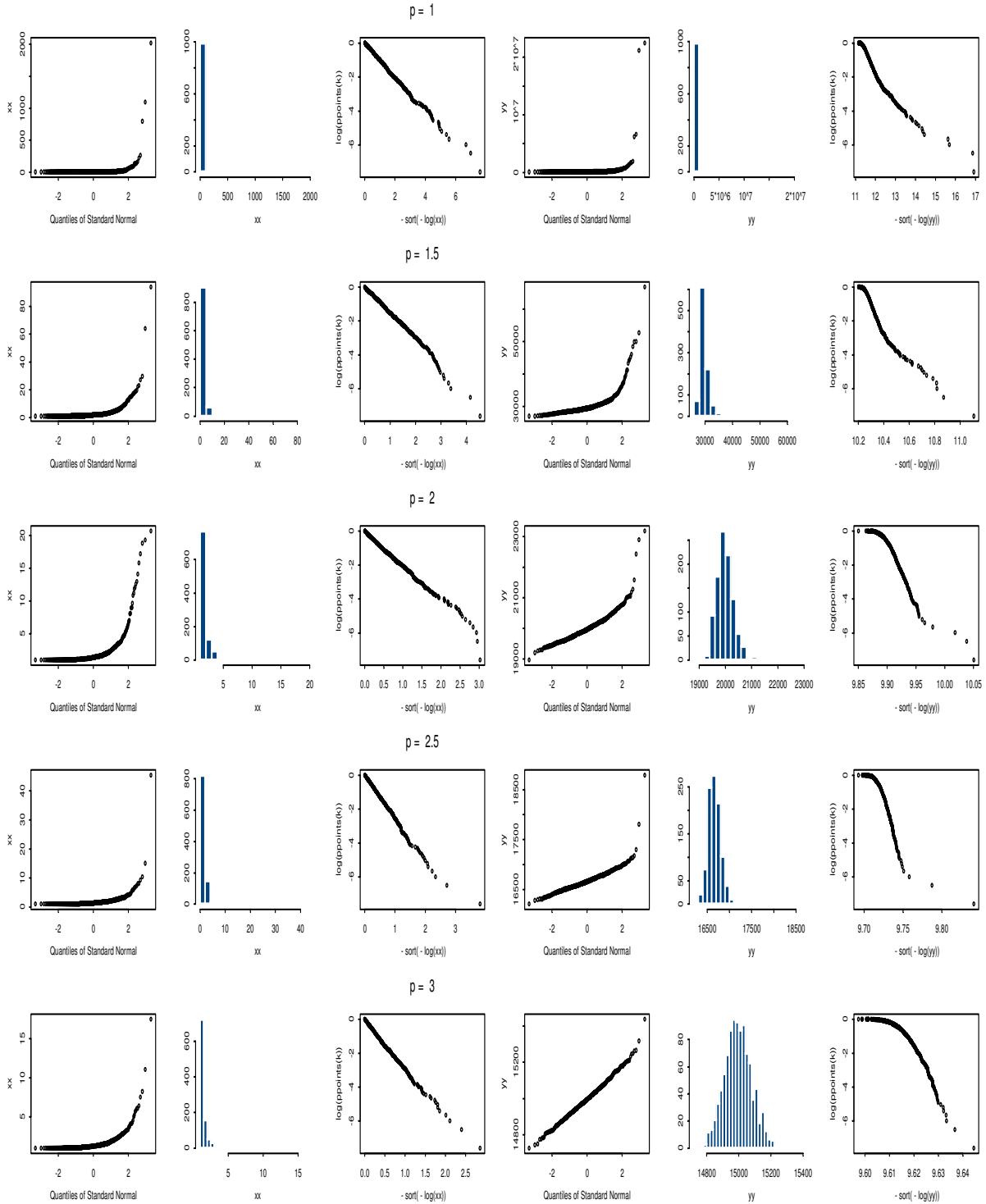


Figure 3.9: Aggregation a sum of iid Pareto random variables ($a = 1, p \in \{1, 1.5, 2, 2.5, 3\}$). On every row: The first three diagrams show the empirical distribution (normal qq-plot, histogram, complementary CDF in log-log scale) of one sample of $n_1 = 10^4$ iid Pareto random variables. The last three show similar diagrams for a sample $(Y_j)_{1 \leq j \leq n}$ of $n = 10^3$ aggregated random variables: $Y_j = \frac{1}{n_1} \sum_{i=1}^{n_1} X_j^i$, where $X_j^i \sim \text{iid Pareto}$. The figure illustrates that for $p < 2$ there is no convergence to a normal distribution, and for $p \geq 2$ there is. It also shows that for $p \geq 2$ the power law behaviour disappears by aggregation, unlike for $p < 2$. Note that for $p = 2$ X_i is heavy tailed but there is convergence to a normal distribution.

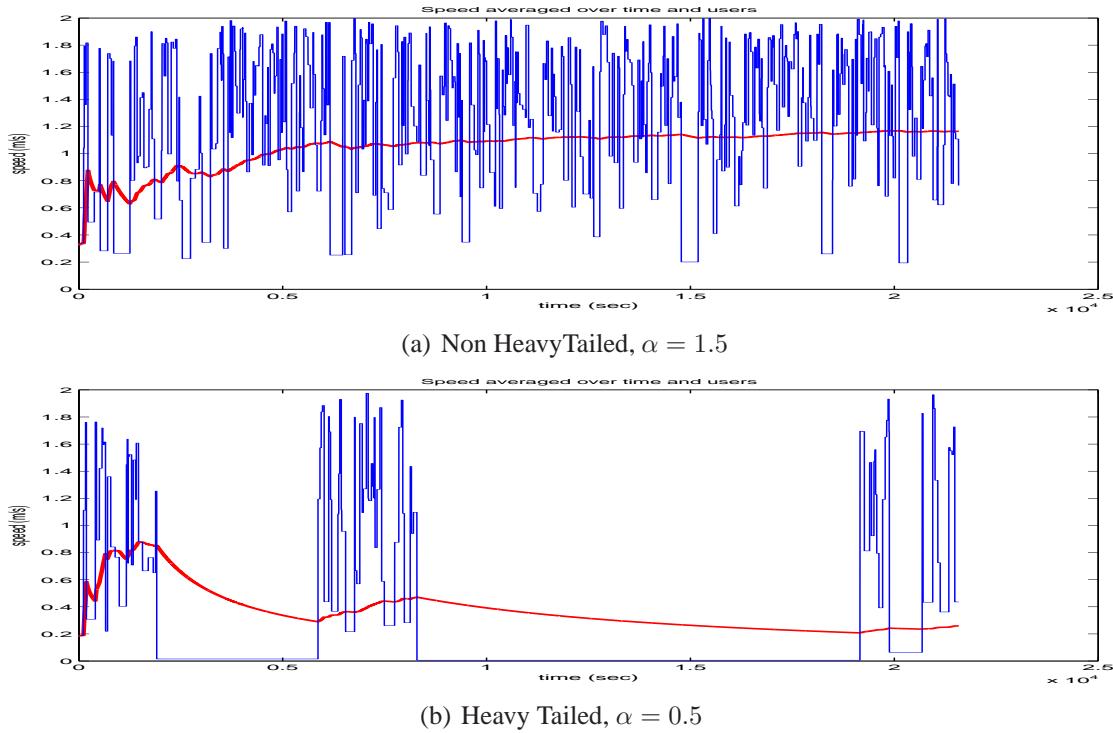


Figure 3.10: Simulation of Random Waypoint with speed density equal to $f_V^0(v) = K_\alpha v^\alpha \mathbf{1}_{\{0 \leq v \leq v_{\max}\}}$, showing instant speed and average speed (smoother line) for one user.

EXAMPLE 3.12: RANDOM WAYPOINT WITH HEAVY TAILED TRIP DURATION. Consider the following variant of the random waypoint mobility model as in Figure 6.3 on Page 170. A mobile moves in some area from one point to the next (we call *trip* the movement from one point to the next). The velocity on one trip is sampled from the distribution with PDF $f_V^0(v) = K_\alpha v^\alpha \mathbf{1}_{\{0 \leq v \leq v_{\max}\}}$, with $\alpha > 0$ and where K_α is a normalizing constant. It follows that the complementary CDF of trip duration is equal to

$$1 - F_T^0(x) = \frac{K_\alpha \bar{D}}{\alpha + 1} \frac{1}{x^{\alpha+1}} \quad (3.27)$$

where \bar{D} is the average length (in meters) of a trip.

For $\alpha = 0.5$ the trip duration is heavy tailed, for $\alpha = 1.5$, it has finite variance and is thus not heavy tailed. Figure 3.10 shows a sample simulation of both cases. In the heavy tailed case, we see that most trip durations are very short, but once in a while, the trip duration is extraordinarily large.

EXAMPLE 3.13: QUEUING SYSTEM. Consider a server that receives requests for downloading files. Assume the requests arrival times form a Poisson process, and the requested file sizes are iid $\sim F$ where F is some distribution. This is a simplified model, but it will be sufficient to make the point.

We assume that the server has a unit capacity, and that the time to serve a request is equal to the requested file size. This again is a simplifying assumption, which is valid if the bottleneck is a single, FIFO I/O device. From Chapter 8, the mean response time of a request is given by the

Pollaczek-Khintchine formula

$$R = \rho + \frac{\rho^2(1 + \frac{\sigma^2}{\mu^2})}{2(1 - \rho)}$$

where: μ is the mean and σ^2 the variance, of F (assuming both are finite); ρ is the utilization factor ($=$ request arrival rate $\times \mu$). Thus the response time depends not only on the utilization and the mean size of requests, but also on the coefficient of variation $C := \sigma/\mu$. As C grows, the response times goes to infinity.

If the real data supports the hypothesis that F is heavy tailed, then the average response time is likely to be high and the estimators of it are unstable.

3.5.4 TESTING FOR HEAVY TAIL

There are many methods for deciding whether a data set is heavy tailed or not. One method consists in fitting a Pareto distribution to the tail, as in Example 3.11.

A more general method is the tool by Crovella and Taqqu called `aest` [31]. It uses the scaling properties and convergence to stable distributions. Consider X_i iid and heavy tailed, with index p . Call $X_i^{(m)}$ the aggregate sequence, where observations are grouped in bulks of m :

$$X_i^{(m)} := \sum_{j=(i-1)m+1}^{im} X_j$$

For large m_1, m_2 , by the convergence result mentioned earlier, we should have approximately the distribution equalities

$$\frac{1}{m_1^{\frac{1}{p}}} X_i^{(m_1)} \sim \frac{1}{m_2^{\frac{1}{p}}} X_j^{(m_2)} \quad (3.28)$$

The idea is now to plot the empirical complementary distributions of $X_i^{(m)}$ for various values of m . Further, the deviation between two curves of the plot is analyzed by means of horizontal and vertical deviations δ and τ as shown in Figure 3.11. We have $\delta = \log x_2 - \log x_1$. By Eq.(3.28), we have $x_2 = (m_2/m_1)^{1/p} x_1$ thus

$$\delta = \frac{1}{p} \log \frac{m_2}{m_1}$$

Also, if X_i is heavy tailed, and m is large, then $X_i^{(m)}$ is approximately stable. Thus, if m_2/m_1 is an integer, the distribution of $X_j^{(m_2)}$ (which is a sum of $X_i^{(m_1)}$) is the same as that of $(m_2/m_1)^{1/p} X_i^{(m_1)}$. We should thus have

$$\tau = \log \mathbb{P}(X_i^{(m_2)} > x_1) - \log \mathbb{P}(X_i^{(m_1)} > x_1) \approx \log \frac{m_2}{m_1}$$

The method in `aest` consists in use only the points x_1 where the above holds, then, at such points, estimate p by

$$\hat{p} = \frac{1}{\delta} \log \frac{m_2}{m_1}$$

Then the average of these estimates is used. See Figure 3.11 for an illustration.

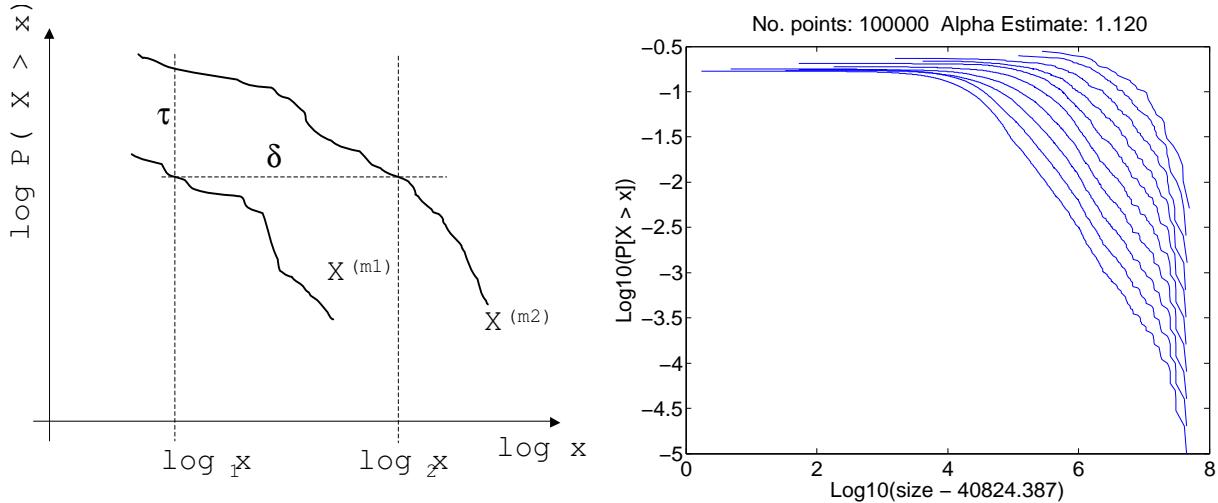


Figure 3.11: First Panel: Deviations used in the aest tool. Second panel: application to the dataset in Example 3.11. There is heavy tail, with an estimated index $p = 1.12$ (same as obtained by the direct method in Example 3.11).

3.5.5 APPLICATION EXAMPLE: THE WORKLOAD GENERATOR SURGE

Many of the concepts illustrated in this chapter are used in the tool Surge[9], which is a load generator for web servers.

The load *intensity* is determined by the number of *User Equivalents* (UEs), each implemented as an independent thread of execution, on one or several machines. The load *nature* is defined by a set of constraints on the arrival process, the distribution of request sizes and the correlation of successive requests to the same object, as described below. The parameters of the distributions were obtained by fitting measured values (Table 3.3).

1. One UE alternates between ON-object periods and “Inactive OFF periods”. Inactive OFF periods are iid with a Pareto distribution .
2. During an ON-object period, a UE sends a request with embedded references. Once the first reference is received, there is an “Active OFF period”, then the request for the second reference is sent, and so on, until all embedded references are received. There is only one TCP connection at a time per UE, and one TCP connection for each reference (an assumption that made sense with early versions of HTTP).
3. The active OFF times are iid random variables with Weibull distributions.
4. The number of embedded references is modelled as a set of iid random variables, with a Pareto distribution.

The references are viewed as requests for downloading files. The model is that there is a set of files labeled $i = 1, \dots, I$, stored on the server. File i has two attributes: size x_i and request probability θ_i . The distribution of attributes has to satisfy the following conditions.

5. The distribution $H(x)$ of file sizes is a combination of truncated Lognormal and Pareto.
6. θ_i satisfy Zipf’s law with exponent $\alpha = 1$

7. The distribution $F(x)$ of requested file sizes is Pareto⁷

The distributions H and F are both file size distributions, sampled according to different viewpoints. Thus (as we discuss in Chapter 7) there must be a relation between these two distributions, which we now derive. Let $I(t)$ be the random variable that gives the index i of the t th file requested. Thus $F(x) = \mathbb{P}(x_{I(t)} = x)$. We can assume that the allocation of file sizes and popularities is done in a preliminary phase, and is independent of $I(t)$. Thus

$$F(x) = \sum_j \mathbb{P}(I(t) = j) \mathbf{1}_{\{x_j \leq x\}} = \sum_j \theta_j \mathbf{1}_{\{x_j \leq x\}} \quad (3.29)$$

Let $x_{(1)} = x_{(2)} = \dots$ be the file sizes sorted in increasing order, and let $z(n)$ be the index of the n th file in that order. z is a permutation of the set of indices, such that $x_{(n)} = x_{z(n)}$. By specializing Eq.(3.29) to the actual values $x_{(m)}$ we find, after a change of variable $j = z(n)$

$$F(x_{(m)}) = \sum_j \theta_j \mathbf{1}_{\{x_j \leq x_{(m)}\}} = \sum_n \theta_{z(n)} \mathbf{1}_{\{x_{(n)} \leq x_{(m)}\}}$$

thus

$$F(x_{(m)}) = \sum_{n=1}^m \theta_{z(n)} \quad (3.30)$$

which gives a constraint between the θ_i s and x_i s.

The file request references $I(t)$, $t = 1, 2, \dots$ are constrained by their marginal distribution (defined by θ_i). The authors find that there is some correlation in the series and model the dependency as follows:

8. For any file index i , define $T_1(i) < T_2(i) < \dots$ the successive values of $t \in \{1, 2, \dots\}$ such that $i = I(t)$. Assume that $T_{k+1}(i) - T_k(i)$ come from a common distribution, called “temporal locality”. The authors find it log-normal (more precisely, it is a discretized log-normal distribution, since the values are integer).

BUILDING A PROCESS THAT SATISFIES ALL CONSTRAINTS

It remains to build a generator that produces a random output conformant to all constraints. Constraints 1 to 4 are straightforward to implement, with a proper random number generator, and using the techniques described in Section 6.6. The inactive OFF periods, active OFF periods and number of embedded references are implemented as mutually independent iid sequences.

Constraints 5 to 7 require more care. First, the x_i are drawn from H . Second, the θ_i s are drawn (as explained in Section 3.4.3) but not yet bound to the file indexes. Instead, the values are put in a set Θ . In view of Eq.(3.30), define

$$\hat{\theta}_{z(m)} = F(x_{(m)}) - \sum_{n=1}^{m-1} \theta_{z(n)}$$

⁷The original paper [9] takes an index $p = 1$ for this Pareto distribution, which implies that the mean request file size is infinite, and thus the process of file size requests is not stationary (this is a freezing simulation problem as in Section 7.4). A value of p larger than 1 would be preferable.

so that we should have $\hat{\theta}_{z(m)} = \theta_{z(m)}$ for all m . If this would be true, it is easy to see that all constraints are satisfied. However, this can be done in [9] only approximately. Here is one way to do it. Assume that $z(m) = m$, namely, we have sorted the file indices by increasing file size. For $m = 1$ we set θ_1 to the value in Θ which is closest to $\hat{\theta}_1 = F(x_1)$. Then remove that value from Θ , set θ_2 to the value in Θ closest to $\hat{\theta}_2 = F(x_2) - \theta_1$, etc.

Lastly, it remains to generate a time series of file requests $I(t)$ such that the marginal distribution is given by the θ_i s and the temporal locality in condition 8 is satisfied. This can be formulated as a discrete optimization problem, as follows. First a trace size T is chosen arbitrarily; it reflects the length of the load generation campaign. Then, for each file i , the number of references N_i is drawn, so as to satisfy Zipf's law (with $\mathbb{E}(N_i) = \theta_i$). Last, a sequence S_1, S_2, \dots is drawn from the distribution in condition 8.

The problem is now to create a sequence of file indices $(I(1), I(2), \dots, I(T))$ such that i appears N_i times and the distances between successive repetitions of file references is as close as possible to the sequence S_1, S_2, \dots . Any heuristic for discrete optimization can be used (such as simulated annealing or tabu search). An ad-hoc heuristic is used in [9].

3.6 PROOFS

THEOREM 3.1 The log likelihood of the data is

$$l_{\vec{y}}(\vec{\beta}, \sigma) = -\frac{I}{2} \ln(2\pi) - I \ln(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^I (y_i - f_i(\vec{\beta}))^2 \quad (3.31)$$

For any fixed σ , it is maximum when $\sum_{i=1}^I (y_i - f_i(\vec{\beta}))^2$ is minimum, which shows item 1. Take the derivative with respect to σ and find that for any fixed $\vec{\beta}$, it is maximum for $\sigma = \frac{1}{I} \sum_i (y_i - f_i(\vec{\beta}))^2$, which shows item 1.

The rest of the theorem is a direct application of Theorem B.1 and Theorem B.2.

THEOREM 3.2 The log likelihood of the data is

$$l_{\vec{y}} = -I \ln(2) + I \ln(\lambda) - \lambda \sum_{i=1}^I |y_i - f_i(\vec{\beta})| \quad (3.32)$$

For any fixed $\vec{\beta}$, it is maximum when $\frac{1}{\lambda} = \frac{1}{I} \sum_i |y_i - f_i(\vec{\beta})|$ and the corresponding value is

$$-I \ln \left(\sum_{i=1}^I |y_i - f_i(\vec{\beta})| \right) + I \ln I - I - I \ln 2$$

which is maximum when $\vec{\beta}$ minimizes $\sum_{i=1}^I |y_i - f_i(\vec{\beta})|$.

THEOREM 3.4 In view of Theorem 3.2, the MLE of $\vec{\beta}$ is obtained by minimizing $\sum_{i=1}^I |y_i - (X\vec{\beta})_i|$. This is equivalent to minimizing $\sum_{i=1}^I u_i$ over $(\vec{\beta}, u)$ with the constraints $u_i \geq |y_i - (X\vec{\beta})_i|$, which is equivalent to the constraints in the theorem.

3.7 REVIEW

3.7.1 REVIEW QUESTIONS

QUESTION 3.7.1. *How would you compute a and α in Example 3.1 ?*⁸

QUESTION 3.7.2. *How would you compute the residuals in Example 3.3 ?*⁹

QUESTION 3.7.3. *How would you compute confidence intervals for the component β_j of $\vec{\beta}$ in Theorem 3.1 using the Bootstrap ? In Theorem 3.2 ?*¹⁰

QUESTION 3.7.4. *Can you name distributions that are fat tailed but not heavy tailed ?*¹¹

QUESTION 3.7.5. *If the tail of the distribution of X follows a power law, can you conclude that X is heavy tailed ?*¹²

QUESTION 3.7.6. *Which of the distributions used in Surge are heavy tailed ? fat tailed ?*¹³

3.7.2 USEFUL MATLAB COMMANDS

- `regress` solves the general linear regression model as in Theorem 3.3
- `linprog` solves the linear program in Theorem 3.4

⁸By minimizing $\sum_i (y_i - ae^{\alpha t_i})^2$. This is an unconstrained optimization problem in two variables; use for example a generic solver such as `fminsearch` in matlab.

⁹The residuals are estimates of the noise terms ϵ_i . Let \hat{a} and $\hat{\alpha}$ be the values estimated by maximum likelihood, for either model. The residuals are $r_i = y_i - \hat{a}e^{\hat{\alpha}t_i}$ for the former model, $r_i = \ln y_i - \ln(\hat{a}e^{\hat{\alpha}t_i})$ for the latter.

¹⁰Draw R bootstrap replicates of \vec{Y} and obtain R estimates $\vec{\beta}^1, \dots, \vec{\beta}^R$ of $\vec{\beta}$, using the theorems. At level 95%, take $R = 999$ and use the order statistics of the j th component of the bootstrap estimates: $\beta_j^{(1)} \leq \dots \leq \beta_j^{(R)}$; obtain as confidence interval $[\beta_j^{(25)}, \beta_j^{(975)}]$.

¹¹The Pareto distributions with $p > 2$, the log-normal distributions, the Weibull distributions with $c < 1$.

¹²No, only if the exponent of the tail is ≤ 2 .

¹³Inactive OFF time, File size, File request size. The number of embedded references is Pareto with $p > 2$ thus is fat tailed but not heavy tailed. The active OFF time and temporal locality are fat tailed but not heavy tailed.

Distribution	Standard Normal $N_{0,1}$	Standard Laplace	Standard Lognormal
Parameters	none	none	$\sigma > 0$
Comment	Page 24	Page 65	Page 73
PDF	$\frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$	$\frac{1}{2} e^{- x }$	$\frac{1}{\sqrt{2\pi}\sigma x} e^{-\frac{(\ln x)^2}{2\sigma^2}} \mathbf{1}_{\{x>0\}}$
support	\mathbb{R}	\mathbb{R}	$[0, +\infty)$
CDF	$1 - Q(x)$ (by definition of $Q()$)	$0.5e^{- x }$ for $x \leq 0$ $1 - 0.5e^{- x }$ for $x > 0$	$(1 - Q(\frac{\ln x}{\sigma})) \mathbf{1}_{\{x>0\}}$
characteristic function	$e^{-\frac{\omega^2}{2}}$	$\frac{1}{1+\omega^2}$	
mean	0	0	$e^{\frac{\sigma^2}{2}}$
variance	1	2	$(e^{\sigma^2} - 1) e^{\sigma^2}$
median	0	0	1
skewness index	0	0	$\sqrt{e^{\sigma^2} - 1} (e^{\sigma^2} + 2)$
kurtosis index	0	3	$e^{4\sigma^2} + 2e^{3\sigma^2} + 3e^{2\sigma^2} - 6$
hazard rate	$\sim x$	= 1	$\sim \frac{\ln x}{\sigma^2 x}$
Effect of change of scale and location			
<i>Original Distribution</i>		<i>Shifted and Re-scaled</i>	
Distribution of X		Distribution of $Y = sX + m$	
Parameters		same plus $m \in \mathbb{R}$ (location), $s > 0$ (scale)	
PDF	$f_X(x)$	$\frac{1}{s} f_X\left(\frac{x-m}{s}\right)$	
CDF	$F_X(x)$	$F_X\left(\frac{x-m}{s}\right)$	
characteristic function	$\Phi_X(\omega)$	$e^{j\omega m} \Phi_X(s\omega)$	
mean	μ	$\mu + m$	
variance	σ^2	$s^2 \sigma^2$	
median	ν	$\nu + m$	
skewness index		same	
kurtosis index		same	
hazard rate	$\lambda_X(x)$	$\frac{1}{s} \lambda_X\left(\frac{x-m}{s}\right)$	

Table 3.1: Catalog of Distributions used in this chapter (continued on Table 3.2). The characteristic function is defined as $\mathbb{E}(e^{j\omega X})$ and is given only when tractable. The notation $a(x) \sim b(x)$ means $\lim_{x \rightarrow \infty} \frac{a(x)}{b(x)} = 1$. Only parameters that affect the shape of the distribution are considered in the table. Other distributions in the same families can be derived by a change of scale and location, using the formulas given in the bottom part of the table.

<i>Distribution</i>	Standard Weibull	Standard Pareto	Standard Stable with index $p < 2$
<i>Parameters</i>	$c > 0$	$0 < p$	$0 < p < 2, -1 \leq \beta \leq 1$
<i>Comment</i>	Page 78; called exponential for $c = 1$	Page 76	The stable definition is also defined for $p = 2$, in which case it is equal to the normal distribution $N_{0,2}$. See Page 85
<i>PDF</i>	$cx^{c-1}e^{-(x^c)}\mathbf{1}_{\{x \geq 0\}}$	$\frac{p}{x^{p+1}}\mathbf{1}_{\{x \geq 1\}}$	well defined but usually not tractable
<i>support</i>	$[0, +\infty)$	$[1, +\infty)$	\mathbb{R} except when $\beta = \pm 1$
<i>CDF</i>	$(1 - e^{-(x^c)})\mathbf{1}_{\{x \geq 0\}}$	$(1 - \frac{1}{x^p})\mathbf{1}_{\{x \geq 1\}}$	well defined but usually not tractable
<i>characteristic function</i>	$\frac{1}{1-j\omega}$ for $c = 1$		$\exp[- \omega ^p(1+A)]$ with $A =$ $-j\beta \text{sgn}(\omega) \tan \frac{p\pi}{2}$ for $p \neq 1$ $\frac{2j\beta}{\pi} \text{sgn}(\omega) \ln \omega $ for $p = 1$
<i>mean</i> μ	$\Gamma\left(\frac{c+1}{c}\right)$	$\frac{p}{p-1}$ for $p > 1$	0 for $p > 1$ else undefined
<i>variance</i> σ^2	$\Gamma\left(\frac{c+2}{c}\right) - \mu^2$	$\frac{1}{(p-1)^2(p-2)}$ for $p > 2$	undefined
<i>median</i>	$(\ln(2))^{1/c}$	$2^{1/p}$	0 when $\beta = 0$, else untractable
<i>skewness index</i> γ_1	$\frac{\Gamma\left(\frac{c+3}{c}\right) - 3\mu\sigma^2 - \mu^3}{\sigma^3}$	$\frac{2(1+p)}{p-3} \sqrt{\frac{p-2}{p}}$ for $p > 3$	undefined
<i>kurtosis index</i>	$\frac{\Gamma\left(\frac{c+4}{c}\right) - 4\gamma_1\mu\sigma^3 - 6\mu^2\sigma^2 - \mu^4}{\sigma^4}$	$\frac{6(p^3 + p^2 - 6p - 2)}{p(p-3)(p-4)}$ for $p > 4$	undefined
<i>hazard rate</i>	$= cx^{c-1}$	$= \frac{p}{x}$	$\sim \frac{p}{x}$

Table 3.2: Continuation of Table 3.1. $\Gamma()$ is the gamma function, defined as $\Gamma(x) = \int_0^\infty e^{-t} t^{x-1} dt$; if $x \in \mathbb{N}$, $\Gamma(x) = (x-1)!$

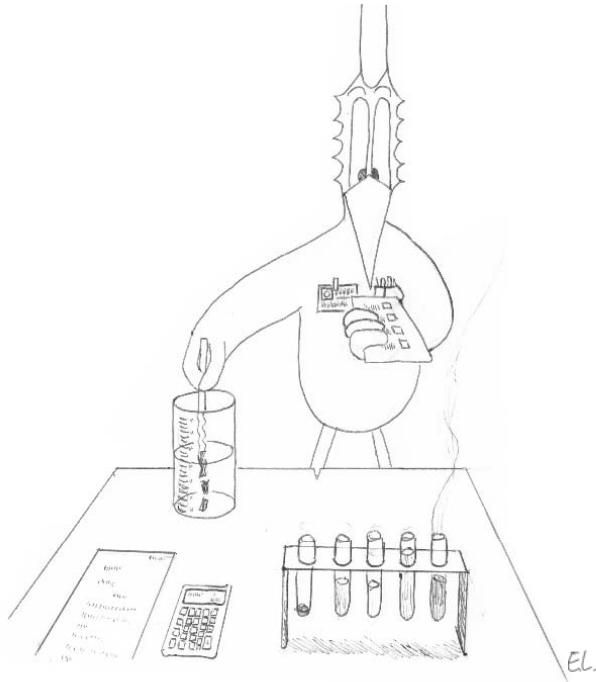
	model	density $f(x)$	value of parameters
Inactive OFF time	Pareto	$\frac{ps^p}{x^{p+1}} \mathbf{1}_{\{x \geq s\}}$	$s = 1, p = 1.5$
No of embedded references	Pareto	$\frac{ps^p}{x^{p+1}} \mathbf{1}_{\{x \geq s\}}$	$s = 1, p = 2.43$
Active OFF time	Weibull	$\frac{c}{s} \left(\frac{x}{s}\right)^{c-1} e^{-\left(\frac{x}{s}\right)^c}$	$s = 1.46, c = 0.382$
File Size	Lognormal	Eq.(3.24)	$\mu = 9.357, \sigma = 1.318$
	comb. Pareto		$a = 133K, p = 1.1$ $q = N_{0,1}(\mu + \sigma \ln a)$
File Request Size	Pareto	$\frac{ps^p}{x^{p+1}} \mathbf{1}_{\{x \geq s\}}$	$s = 1000, p = 1.0$ (see footnote on Page 90)
Temporal Locality	Lognormal	$\frac{e^{-\frac{(\ln x - \mu)^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma x} \mathbf{1}_{\{x > 0\}}$	$\mu = 1.5, \sigma = 0.80$

Table 3.3: Distributions and parameters used in SURGE.

CHAPTER 4

TESTS

We use tests to decide whether some assertion on a model is true or not, for example: does this data set come from a normal distribution ? We have seen in Chapter 2 that visual tests may be used for such a purpose. Tests are meant to be a more objective way to reach the same goal.



Tests are often used in empirical sciences to draw conclusions from noisy experiments. Though we use the same theories, our setting is a bit different; we are concerned with the nested model setting, i.e. we want to decide whether a simpler model is good enough, or whether we do need a more sophisticated one. Here, the question is asymmetric; if in doubt, we give preference to the simpler model – this is the principle of parsimony. The Neyman Pearson framework is well suited to such a setting, therefore we restrict ourselves to it.

There is a large number of tests, and everyone can invent their own (this is perhaps a symptom of the absence of a simple, non equivocal optimality criterion). In practice though, likelihood ratio tests are asymptotically optimal, in some sense, under very large sets of assumptions. They are very general, easy to use and even to develop; therefore, it is worth knowing them. We often make use of Monte Carlo simulation to compute the p -value of a test; this is sometimes brute force, but it avoids spending too much time solving for analytical formulae. We discuss ANOVA, as it is very simple when it applies. Last, we also study robust tests, i.e. tests that make little assumption about the distribution of the data.

Contents

4.1	The Neyman Pearson Framework	98
4.1.1	The Null Hypothesis and The Alternative	98
4.1.2	Critical Region, Size and Power	99
4.1.3	p -value of a Test	102
4.1.4	Tests Are Just Tests	103
4.2	Likelihood Ratio Tests	104
4.2.1	Definition of Likelihood Ratio Test	104
4.2.2	Student Test for Single Sample (or Paired Data)	105
4.2.3	The Simple Goodness of Fit Test	106
4.3	ANOVA	108
4.3.1	Analysis of Variance (ANOVA) and F -tests	108
4.3.2	Testing for a Common Variance	112
4.4	Asymptotic Results	114
4.4.1	Likelihood Ratio Statistic	114
4.4.2	Pearson Chi-squared Statistic and Goodness of Fit	114
4.4.3	Test of Independence	117
4.5	Other Tests	118
4.5.1	Goodness of Fit Tests based on Ad-Hoc Pivots	118
4.5.2	Robust Tests	121
4.6	Proofs	123
4.7	Review	125
4.7.1	Tests Are Just Tests	125
4.7.2	Review Questions	125

4.1 THE NEYMAN PEARSON FRAMEWORK

4.1.1 THE NULL HYPOTHESIS AND THE ALTERNATIVE

We are given a data sample x_i , $i = 1, \dots, n$. We assume that the sample is the output generated by some unknown model. We consider two possible hypotheses about the model, H_0 and H_1 , and

we would like to infer from the data which of the two hypotheses is true. In the Neyman-Pearson framework, the two hypotheses play different roles: H_0 , the *null hypothesis*, is the conservative one. We do not want to reject it unless we are fairly sure. H_1 is the *alternative hypothesis*.

We are most often interested in the *nested model* setting: the model is parameterized by some θ in some space Θ , and $H_0 \stackrel{\text{def}}{=} \{\theta \in \Theta_0\}$ whereas $H_1 \stackrel{\text{def}}{=} \{\theta \in \Theta \setminus \Theta_0\}$, where Θ_0 is a subset of Θ .

In Example 4.1, the model could be: all data points for compiler option 0 [resp. 1] are generated as iid random variables with some distribution F_0 [resp. F_1]. Then H_0 is: “ $F_0 = F_1$ ” and H_1 is “ F_0 and F_1 differ by a shift in location”. This is the model used by the Wilcoxon Rank Sum test (see Example 4.20 for more details). Here $\Theta_0 = \{(F_0, F_0), F_0 \text{ is a CDF}\}$ and $\Theta = \{(F_0, F_1), F_0 \text{ is a CDF and } F_1(x) = F_0(x - m), m \in \mathbb{R}\}$.

Another, commonly used model, for the same example could be: all data points for compiler option 0 [resp. 1] are generated as iid random variables with some normal distribution N_{μ_0, σ^2} [resp. N_{μ_1, σ^2}]. Then H_0 is: “ $\mu_0 = \mu_1$ ” and H_1 is “ $\mu_0 \neq \mu_1$ ”. This is the model used by the so-called “Analysis of variance” (see Example 4.10 for more details). Here $\Theta_0 = \{(\mu_0, \mu_0, \sigma > 0)\}$ and $\Theta = \{(\mu_0, \mu_1, \sigma > 0)\}$. Clearly this second model makes more assumptions, and is to be taken with more care.

EXAMPLE 4.1: NON PAIRED DATA. A simulation study compares the execution time, on a log scale, with two compiler options. See Figure 4.1 for some data. We would like to test the hypothesis that compiler option 0 is better than 1. For one parameter set, the two series of data come from different experiments.

We can compute a confidence interval for each of the compiler options. The data looks normal, so we apply the student statistic and find the confidence intervals shown on the figure.

For parameter set 1, the confidence intervals are disjoint, so it is clear that option 0 performs better. For parameter sets 2 and 3, the intervals are overlapping, so we cannot conclude at this point.

We see here that confidence intervals may be used in some cases for hypothesis testing, but not always. We study in this chapter how tests can be used to disambiguate such cases.

4.1.2 CRITICAL REGION, SIZE AND POWER

The *critical region*, also called *rejection region* C of a test is a set of values of the tuple (x_1, \dots, x_n) such that if $(x_1, \dots, x_n) \in C$ we reject H_0 , and otherwise we accept H_0 . The critical region entirely defines the test¹.

The output of a test is thus a binary decision: “accept H_0 ”, or “reject H_0 ”. The output depends on the data, which is random, and may be wrong with some (hopefully small) probability. We distinguish two types of errors

- A *type 1* error occurs if we reject H_0 when H_0 is true
- Conversely, a *type 2* error occurs if accept H_0 when H_1 is true.

¹In all generality, one also should consider randomized tests, whose output may be a random function of (x_1, \dots, x_n) . See [81] for such tests. We do not use them in our setting

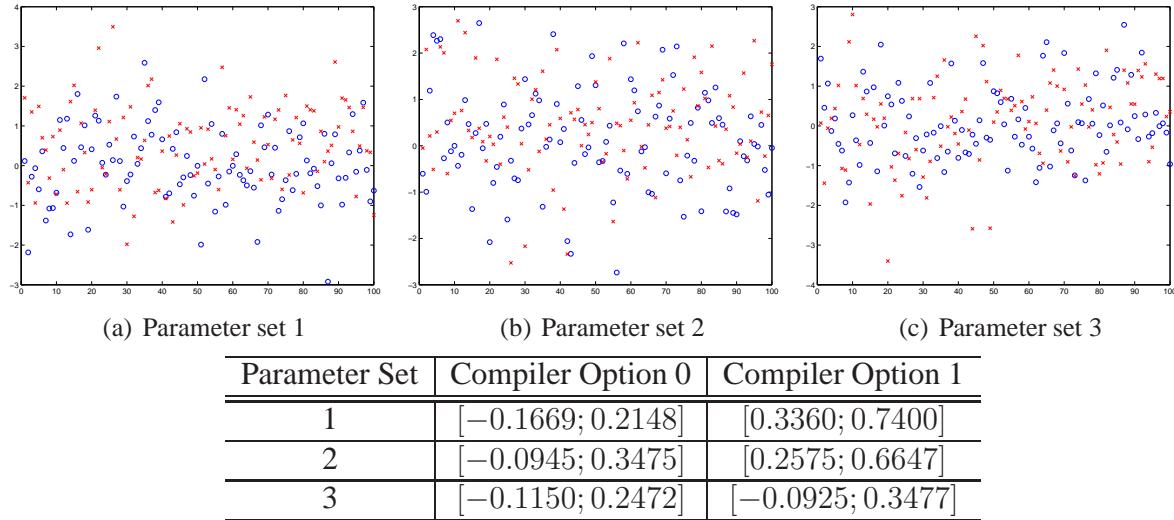


Figure 4.1: Data for Example 4.1. Top: Logarithm of execution time, on a log scale, with two compiler options (\circ =option 0, \times =option 1) for three different parameter sets. Bottom: 95% confidence intervals for the means.

The art of test development consists in minimizing both error types. However, it is usually difficult to minimize two objectives at a time. The maximum probability of a type 1 error, taken over all $\theta \in \Theta_0$ is called the **size** of the test. The **power** function of the test is the probability of rejection of H_0 as a function of $\theta \in \Theta \setminus \Theta_0$. Neyman-Pearson tests are designed such that the size has a fixed, small value (typically 5%, or 1%). Good tests (i.e. those in these lecture notes and those used in Matlab) are designed so as to minimize, exactly or approximately, the power, subject to a fixed size. A test is said to be uniformly more powerful (UMP) if, among all tests of same size, it maximizes the power for every value of $\theta \in \Theta \setminus \Theta_0$. UMP tests exist for few models, therefore less restrictive requirements were developed (the reference for these issues is [62]).

It is important to be aware of the two types of errors, and of the fact that the size of a test is just one facet. Assume we use a UMP test of size 0.05; it does not mean that the risk of error is indeed 0.05, or even is small. It simply means that all other tests that have a risk of type 1 error bounded by 0.05 must have a risk of type 2 error which is the same or larger. Thus we may need to verify whether, for the data at hand, the power is indeed large enough, though this is seldom done in practice.

EXAMPLE 4.2: COMPARISON OF TWO OPTIONS, REDUCTION IN RUN TIME. The reduction in run time due to a new compiler option is given in Figure 2.7 on Page 32. Assume that we know that the data comes from some iid $X_i \sim N_{\mu, \sigma^2}$. This may be argued and will be discussed again, but it is convenient to simplify the discussion here. We do not know μ or σ .

We want to test $H_0: \mu = 0$ against $H_1: \mu > 0$. Here $\theta = (\mu, \sigma)$, $\Theta = [0, \infty) \times (0, \infty)$ and $\Theta_0 = \{0\} \times (0, \infty)$. An intuitive definition of a test is to reject H_0 if the sample mean is large enough; if we rescale the sample mean by its estimated standard deviation, this gives the rejection region

$$C = \left\{ (x_1, \dots, x_n) \text{ such that } \frac{\bar{x}}{s_n/\sqrt{n}} > c \right\} \quad (4.1)$$

for some value of c to be defined later and with, as usual $\bar{x} = \frac{1}{n} \sum_{i=1}^n X_i$ and $s_n^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{x})^2$.

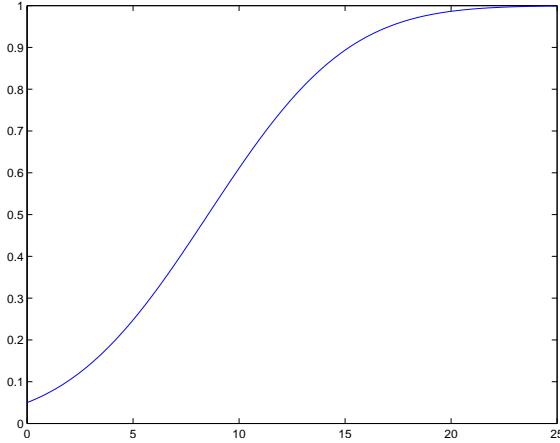


Figure 4.2: Power as a function of μ for Example 4.2.

The size of the test is the maximum probability of C for $\theta \in \Theta_0$. We have

$$\mathbb{P}\left(\sqrt{n} \frac{\bar{x}}{s_n} > c \mid \mu = 0, \sigma\right) \approx 1 - N_{0,1}(c)$$

where $N_{0,1}$ is the CDF of the standard gaussian distribution. Note that this is independent of σ therefore

$$\alpha = \sup_{\sigma > 0} (1 - N_{0,1}(c)) = 1 - N_{0,1}(c)$$

If we want a test size equal to 0.05 we need to take $c = 1.645$. For the data at hand the value of the test statistic is $\sqrt{n} \frac{\bar{x}}{s_n} = 5.05 > c$ therefore we reject H_0 and decide that the mean is positive.

The power function is

$$\begin{aligned} \beta(\mu, \sigma) &\stackrel{\text{def}}{=} \mathbb{P}\left(\sqrt{n} \frac{\bar{x}}{s_n} > c \mid \mu, \sigma\right) \\ &= \mathbb{P}\left(\sqrt{n} \frac{\bar{x} - \mu}{s_n} > c - \sqrt{n} \frac{\mu}{s_n} \mid \mu, \sigma\right) \\ &\approx 1 - N_{0,1}\left(c - \sqrt{n} \frac{\mu}{\sigma}\right) \end{aligned} \tag{4.2}$$

Figure 4.2 plots the power as a function of μ when $c = 1.645$ and for σ replaced by its estimator value s_n . For μ close to the 0, the power is bad (i.e. the probability of deciding H_1 is very small). This is unavoidable as $\lim_{\mu \rightarrow 0} \beta(\mu, \sigma) = \alpha$.

For the data at hand, we estimate the power by setting $\mu = \bar{x}$ and $\sigma = s_n$ in Eq.(4.2). For a test size equal to 0.05 (i.e. for $c = 1.645$) we find 0.9997. The probability of a type 2 error (deciding for H_0 when H_1 is true) is thus approximately 0.0003, a very small value. If we pick as test size $\alpha = 0.1\%$, we find that the type 2 error probability is 2.5%.

The previous example shows that the test size does not say everything. On Figure 4.1, we see that there is a “grey zone” (values of μ below, say, 15) where the power of the test is not large. If the true parameter is in the grey zone, the probability of type 2 error may be large, i.e. it is not improbable that the test will accept H_0 even when H_1 is true. It is important to keep this in mind: a test may accept H_0 because it truly holds, but also because it is unable to reject it. This is the fundamental asymmetry of the Neyman-Pearson framework.

The power function can be used to decide on the size α of the test, at least in theory, as illustrated next.

EXAMPLE 4.3: OPTIMAL TEST SIZE, CONTINUATION OF EXAMPLE 4.2. Say that we consider that a reduction in run time is negligible if it is below μ^* . We want that the probability of deciding H_0 when the true value equal to μ^* or more is similar to the size α , i.e. we want to balance the two types of errors. This gives the equations

$$\begin{aligned} 1 - N_{0,1}(c^*) &= \alpha \\ 1 - N_{0,1}\left(c^* - \sqrt{n}\frac{\mu^*}{s_n}\right) &= 1 - \alpha \end{aligned}$$

thus

$$N_{0,1}(c^*) + N_{0,1}\left(c^* - \sqrt{n}\frac{\mu^*}{s_n}\right) = 1$$

By symmetry of the gaussian PDF around its mean, we have

$$\text{if } N_{0,1}(x) + N_{0,1}(y) = 1 \text{ then } x + y = 0$$

from where we derive

$$c^* = \sqrt{n}\frac{\mu^*}{2s_n}$$

The table below gives a few numerical examples, together with the corresponding test size $\alpha^* = 1 - N_{0,1}(c^*)$.

resolution μ^*	optimal threshold c^*	size α^*
10	0.97	0.17
20	1.93	0.02
40	3.87	5.38e-005

We see that if we care about validly detecting reductions in run time as small as $\mu^* = 10\text{ms}$, we should have a test size of 17% or more. In contrast, if the resolution μ^* is 20ms, then a test size of 2% is appropriate.

4.1.3 p -VALUE OF A TEST.

For many tests, the rejection region has the form $\{T(\vec{x}) > m_0\}$, where \vec{x} is the observation, $T()$ some mapping, and m_0 is a parameter that depends on the size α of the test. In Example 4.2 we can take $T(\vec{x}) = \sqrt{n}\frac{\vec{x}}{s_n}$.

DEFINITION 4.1. *The p-value of an observation \vec{x} is*

$$p^*(\vec{x}) = \sup_{\theta \in \Theta_0} \mathbb{P}(T(\vec{X}) > T(\vec{x}) | \theta)$$

In this formula, \vec{X} is a random vector that represents a hypothetical replication of the experiment, whereas \vec{x} is the data that we have observed.

The mapping $m \mapsto \sup_{\theta \in \Theta_0} \mathbb{P}(T(\vec{X}) > m | \theta)$ is monotonic nonincreasing, and usually decreasing. Assuming the latter, we have the equivalence

$$p^*(\vec{x}) < \alpha \Leftrightarrow T(\vec{x}) > m_0$$

in other words, instead of comparing the test statistic $T(\vec{x})$ against the threshold m_0 , we can compare the p -value against the test size α :

The test rejects H_0 when the p -value is smaller than the test size α .

The interest of the p -value is that it gives more information than just a binary answer. It is in fact the minimum test size required to reject H_0 . Very often, software packages return p -values rather than hard decisions (H_0 or H_1).

EXAMPLE 4.4: CONTINUATION OF EXAMPLE 4.2. The p -value is $p^* = 1 - N_{0,1} \left(\frac{\sqrt{n}\bar{x}}{s_n} \right)$. We find $p^* = 2.2476e-007$ which is small, therefore we reject H_0 .

4.1.4 TESTS ARE JUST TESTS

When using a test, it is important to make the distinction between statistical significance and practical relevance. Consider for example a situation, as in Example 4.2, where we want to test whether a mean μ satisfies $\mu = \mu_0 = 0$ or $\mu > \mu_0$. We estimate the theoretical mean μ by the sample mean \bar{x} . It is never the case that $\mu = \bar{x}$ exactly. A test is about deciding whether the distance between μ_0 and \bar{x} can be explained by the randomness of the data alone (in which case we should decide that $\mu = \mu_0$), or by the fact that, truly, $\mu > \mu_0$. Statistical significance means that, in a case where we find $\bar{x} > \mu_0$, we can conclude that there is a real difference, i.e. $\mu > \mu_0$. Practical relevance means that the difference $\mu - \mu_0$ is important for the system under consideration. It may well happen that a difference is statistically significant (e.g. with a very large data set) but practically irrelevant, and vice versa (e.g. when the data set is small).

In some cases, tests can be avoided by the use of confidence intervals. This applies to matching pairs as in Example 4.2: a confidence interval for the mean can readily be obtained by Theorem 2.2. At level 0.05, the confidence interval is $[15.9, 36.2]$, so we can conclude that $\mu > 0$ (and more, we have a lower bound on μ).

More generally, consider a generic model parameterized with some $\theta \in \Theta \subset \mathbb{R}$. For testing

$$\theta = \theta_0 \text{ against } H_1: \theta \neq \theta_0$$

we can take as rejection region

$$|\hat{\theta} - \theta_0| > c$$

If $\hat{\theta} \pm c$ is a confidence interval at level $1 - \alpha$, then the size of this test is precisely α . For such cases, we do not need to use tests, since we can simply use confidence intervals as discussed in Chapter 2. However, it is not always as simple, or even possible, to reduce a test to the computation of confidence intervals, as for example with unpaired data in Example 4.1 (though it is possible to use confidence sets rather than confidence intervals).

4.2 LIKELIHOOD RATIO TESTS

In this section we introduce a generic framework, very frequently used for constructing tests. It does not give UMP tests (as this is, in general, not possible), but the tests are asymptotically UMP (under the conditions of Theorem 4.3). We give the application to simple tests for paired data and for goodness of fit. Note that deciding which test is best is sometimes controversial, and the best tests, in the sense of UMP, is not always the likelihood ratio test [61]; note also that the issue of which criterion to use to decide that a test is best is disputed [79]. In our context, likelihood ratio tests are appealing as they are simple and generic.

4.2.1 DEFINITION OF LIKELIHOOD RATIO TEST

ASSUMPTIONS AND NOTATION We assume the nested model setting, with $H_0 \stackrel{\text{def}}{=} \theta \in \Theta_0$ whereas $H_1 \stackrel{\text{def}}{=} \theta \in \Theta \setminus \Theta_0$. For a given statistic (random variable) \vec{X} and value \vec{x} of \vec{X} , define :

- $l_{\vec{x}}(\theta) \stackrel{\text{def}}{=} \ln f_{\vec{X}}(\vec{x}|\theta)$ where $f_{\vec{X}}(\cdot|\theta)$ is the probability density of the model, when the parameter is θ .
- $l_{\vec{x}}(H_0) = \sup_{\theta \in \Theta_0} l_{\vec{x}}(\theta)$
- $l_{\vec{x}}(H_1) = \sup_{\theta \in \Theta} l_{\vec{x}}(\theta)$

For example, assume some data comes from an iid sequence of normal RVs $\sim N(\mu, \sigma)$. We want to test $\mu = 0$ versus $\mu \neq 0$. Here $\Theta = \{(\mu, \sigma > 0)\}$ and $\Theta_0 = \{(0, \sigma > 0)\}$.

If H_0 is true, then, approximately, the likelihood is maximum for $\theta \in \Theta_0$ and thus $l_{\vec{x}}(H_0) = l_{\vec{x}}(H_1)$. In the opposite case, the maximum likelihood is probably reached at some $\theta \notin \Theta_0$ and thus $l_{\vec{x}}(H_1) > l_{\vec{x}}(H_0)$. This gives an idea for a generic family of tests:

DEFINITION 4.2. *The likelihood ratio test is defined by the rejection region*

$$C = \{l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) > k\}$$

where k is chosen based on the required size of the test.

The test statistic $l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0)$ is called **likelihood ratio** for the two hypotheses H_0 and H_1 .

Thus we reject $\theta \in \Theta_0$ when the likelihood ratio statistic is large. The Neyman-Pearson lemma [104, Section 6.3] tells us that, in the simple case where Θ_0 and Θ_1 contain only one value each, the likelihood ratio test minimizes the probability of type 2 error. Most tests used in this lecture are actually likelihood ratio tests. As we will see later, for large sample size, there are simple, generic results for such tests.

There is a link with the theory of maximum likelihood estimation. Under the conditions in Definition B.1, define

- $\hat{\theta}_0$: the maximum likelihood estimator of θ when we restrict θ to be in Θ_0
- $\hat{\theta}$: the unrestricted maximum likelihood estimator of θ

Then $l_{\vec{x}}(H_0) = l_{\vec{x}}(\hat{\theta}_0)$ and $l_{\vec{x}}(H_1) = l_{\vec{x}}(\hat{\theta})$. In the rest of this section and in the next two sections we show applications to various settings.

QUESTION 4.2.1. Why can we be sure that $l_{\vec{x}}(\hat{\theta}) - l_{\vec{x}}(\hat{\theta}_0) \geq 0$? ²

EXAMPLE 4.5: **CONTINUATION OF EXAMPLE 4.2, COMPILER OPTIONS.** We want to test $H_0: \mu = 0$ against $H_1: \mu > 0$. The log-likelihood of an observation is

$$l_{\vec{x}}(\mu, \sigma) = \frac{-n}{2} \ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_i (x_i - \mu)^2$$

and the likelihood ratio statistic is

$$l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) = \sup_{\mu \geq 0, \sigma > 0} l_{\vec{x}}(\mu, \sigma) - \sup_{\sigma > 0} l_{\vec{x}}(0, \sigma) = -n \ln \frac{\hat{\sigma}_1}{\hat{\sigma}_0}$$

with

$$\begin{aligned}\hat{\sigma}_0^2 &= \frac{1}{n} \sum_i x_i^2 \\ \hat{\sigma}_1^2 &= \frac{1}{n} \sum_i (x_i - \hat{\mu}_n^+)^2 \\ \hat{\mu}_n^+ &= \max(\bar{x}, 0)\end{aligned}$$

The likelihood ratio test has a rejection region of the form $l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) > K$ for some constant K , which is equivalent to

$$\hat{\sigma}_1 < k\hat{\sigma}_0 \quad (4.3)$$

for some other constant k . In other words, we reject H_0 if the estimated variance under H_1 is small. Such a test is called “Analysis of Variance”.

We can simplify the definition of the rejection region by noting first that $\hat{\sigma}_1 \leq \hat{\sigma}_0$, and thus we must have $k \leq 1$. Second, if $\bar{x} \geq 0$ then Eq.(4.3) is equivalent to $\sqrt{n} \frac{\bar{x}}{\hat{s}_n} > c$ for some c . Third, if $\bar{x} \leq 0$ then Eq.(4.3) is never true. In summary, we have shown that this test is the same as the ad-hoc test developed in Example 4.2.

4.2.2 STUDENT TEST FOR SINGLE SAMPLE (OR PAIRED DATA)

This test applies to a single sample of data, assumed to be normal with unknown mean and variance. It can also be applied to two paired samples, after computing the differences. It is the two sided variant of Example 4.5. The model is: $X_1, \dots, X_n \sim \text{iid } N_{\mu, \sigma^2}$ where μ and σ are not known. The hypotheses are:

$$H_0: \mu = \mu_0 \text{ against } H_1: \mu \neq \mu_0$$

where μ_0 is a fixed value. We compute the likelihood ratio statistic and find after some algebra:

$$l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) = \frac{n}{2} \ln \left(1 + \frac{n(\bar{x} - \mu_0)^2}{\sum_i (x_i - \bar{x})^2} \right)$$

²As long as the MLEs exist: by definition, $l_{\vec{x}}(\hat{\theta}) \geq l_{\vec{x}}(\theta)$ for any θ .

Let $T(\vec{x}) = \sqrt{n} \frac{\bar{x} - \mu_0}{\hat{\sigma}}$ be the student statistic (Theorem 2.3), with $\hat{\sigma}^2 = \frac{1}{n-1} \sum_i (x_i - \bar{x})^2$. We can write the likelihood ratio statistic as

$$l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) = \frac{n}{2} \ln \left(1 + \frac{T(\vec{x})^2}{n-1} \right) \quad (4.4)$$

which is an increasing function of $|T(\vec{x})|$. The rejection region thus has the form

$$C = \{|T(\vec{x})| > \eta\}$$

We compute η from the condition that the size of the test is α . Under H_0 , $T(\vec{X})$ has a student distribution t_{n-1} (Theorem 2.3). Thus

$$\eta = t_{n-1}^{-1} \left(1 - \frac{\alpha}{2} \right) \quad (4.5)$$

For example, for $\alpha = 0.05$ and $n = 100$, $\eta = 1.98$.

The p -value is

$$p^* = 2(1 - t_{n-1}(T(\vec{x}))) \quad (4.6)$$

EXAMPLE 4.6: PAIRED DATA. This is a variant of Example 4.2. Consider again the reduction in run time due to a new compiler option, as given in Figure 2.7 on Page 32. We want to test whether the reduction is significant. We assume the data is iid normal and use the student test:

$$H_0: \mu = 0 \text{ against } H_1: \mu \neq 0$$

The test statistic is $T(\vec{x}) = 5.05$, larger than 1.98, so we reject H_0 . Alternatively, we can compute the p -value and obtain $p^* = 1.80e-006$, which is small, so we reject H_0 .

As argued in Section 4.1.4, the Student test is equivalent to confidence interval, so you do not need to use it. However, it is very commonly used by others, so you still need to understand what it does and when it is valid.

4.2.3 THE SIMPLE GOODNESS OF FIT TEST

Assume we are given n data points x_1, \dots, x_n , assumed to be generated from an iid sequence, and we want to verify whether their common distribution is a given distribution $F()$. A traditional method is to compare the empirical histogram to the theoretical one. Applying this idea gives the following likelihood ratio test. We call it the **simple goodness of fit test** as the null hypothesis is for a given, fixed distribution $F()$ (as opposed to a family of distributions, which would give a *composite* goodness of fit test).

To compute the empirical histogram, we partition the set of values of \vec{X} into **bins** B_i . Let $N_i = \sum_{k=1}^n \mathbf{1}_{\{B_i\}}(X_k)$ (number of observation that fall in bin B_i) and $q_i = \mathbb{P}\{X_1 \in B_i\}$. If the data comes from the distribution $F()$ the distribution of \vec{N} is **multinomial** $M_{n,\vec{q}}$, i.e.

$$\mathbb{P}\{N_1 = n_1, \dots, N_k = n_k\} = \frac{n!}{n_1! \dots n_k!} q_1^{n_1} \dots q_k^{n_k} \quad (4.7)$$

The test is

H_0 : \vec{N} comes from the multinomial distribution $M_{n,\vec{q}}$

against

H_1 : \vec{N} comes from a multinomial distribution $M_{n,\vec{p}}$ for some arbitrary \vec{p} .

We now compute the likelihood ratio statistic. The parameter is $\theta = \vec{p}$. Under H_0 , there is only one possible value so $\hat{\theta}_0 = \vec{q}$. From Eq.(4.7), the likelihood is

$$l_{\vec{x}}(\vec{p}) = C + \sum_{i=1}^k n_i \ln(p_i) \quad (4.8)$$

where $n_i = \sum_{k=1}^n \mathbf{1}_{\{B_i\}}(x_k)$ and $C = \ln(n!) - \sum_{i=1}^k \ln(n_i!)$. C is a constant and can be ignored in the rest. To find $\hat{\theta}$, we have to maximize Eq.(4.8) subject to the constraint $\sum_{i=1}^k p_i = 1$. The function to maximize is concave in p_i , so we can find the maximum by the lagrangian technique. The lagrangian is

$$L(\vec{p}, \lambda) = \sum_{i=1}^k n_i \ln(p_i) + \lambda(1 - \sum_{i=1}^k p_i) \quad (4.9)$$

The equations $\frac{\partial L}{\partial p_i} = 0$ give $n_i = \lambda p_i$. Consider first the case $n_i \neq 0$ for all i . We find λ by the constraint $\sum_{i=1}^k p_i = 1$, which gives $\lambda = n$ and thus $\hat{p}_i = \frac{n_i}{n}$. Finally, the likelihood ratio statistic is

$$l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0) = \sum_{i=1}^k n_i \ln \frac{n_i}{n q_i} \quad (4.10)$$

In the case where $n_i = 0$ for some i , the formula is the same if we adopt the convention that, in Eq.(4.10), the term $n_i \ln \frac{n_i}{n q_i}$ is replaced by 0 whenever $n_i = 0$.

We now compute the p -value. It is equal to

$$\mathbb{P} \left(\sum_{i=1}^k N_i \ln \frac{N_i}{n q_i} > \sum_{i=1}^k n_i \ln \frac{n_i}{n q_i} \right) \quad (4.11)$$

where \vec{N} has the multinomial distribution $M_{n,\vec{q}}$.

For large n , we will see in Section 4.4 a simple approximation for the p -value. If n is not large, there is no known closed form, but we can use Monte Carlo simulation as discussed in Section 6.4.

EXAMPLE 4.7: MENDEL [104]. Mendel crossed peas and classified the results in 4 classes of peas $i = 1, 2, 3, 4$. If his genetic theory is true, the probability that a pea belongs to class i is $q_1 = 9/16, q_2 = q_3 = 3/16, q_4 = 1/16$. In one experiment, Mendel obtained $n = 556$ peas, with $N_1 = 315, N_2 = 108, N_3 = 102$ and $N_4 = 31$. The test is

H_0 : “ $\vec{q} = \vec{p}$ ” against H_1 : “ \vec{p} is arbitrary”

The test statistic is

$$\sum_{i=1}^k n_i \ln \frac{n_i}{n q_i} = 0.3092 \quad (4.12)$$

We find the p -value by Monte-Carlo simulation (Example 6.8) and find $p = 0.9191 \pm 0.0458$. The p -value is (very) large thus we accept H_0 .

QUESTION 4.2.2. Assume we compute the p -value of a test by Monte Carlo simulation with 100 replicates and find an estimated p equal to 0. Can we say that the p -value is small so we reject H_0 ?³

4.3 ANOVA

In this section we cover a family of exact tests when we can assume that the data is normal. It applies primarily to cases with multiple, unpaired samples.

4.3.1 ANALYSIS OF VARIANCE (ANOVA) AND F -TESTS

Analysis of variance (ANOVA) is used when we can assume that the data is a family of independent normal variables, with an arbitrary family of means, but with common variance. The goal is to test some property of the mean. The name ANOVA is explained by Theorem 4.1.

ANOVA is found under many variants, and the basis is often obscured by complex computations. All variants of ANOVA are based on a single result, which we give next; they differ only in the details of the linear operators Π_M and Π_{M_0} introduced below.

ASSUMPTIONS AND NOTATION FOR ANOVA

- The data is a collection of *independent, normal* random variables X_r , here the index r is in some finite set R (with $|R| =$ number of elements in R).
- $X_r \sim N_{\mu_r, \sigma^2}$, i.e. all variables have the *same variance* (this is pompously called “homoscedasticity”). The common variance is fixed but unknown.
- The means μ_r satisfy some linear constraints, i.e. we assume that $\vec{\mu} \stackrel{\text{def}}{=} (\mu_r)_{r \in R} \in M$, where M is a linear subspace of \mathbb{R}^R . Let $k = \dim M$. The parameter of the model is $\theta = (\vec{\mu}, \sigma)$ and the parameter space is $\Theta = M \times (0, +\infty)$
- We want to test the nested model $\vec{\mu} \in M_0$, where M_0 is a linear sub-space of M . Let $k_0 = \dim M_0$. We have $\Theta_0 = M_0 \times (0, +\infty)$.
- Π_M [resp. Π_{M_0}] is the orthogonal projector on M [resp. M_0]

EXAMPLE 4.8: *NON PAIRED DATA.* (Continuation of Example 4.1) Consider the data for one parameter set. The model is

$$X_i = \mu_1 + \epsilon_{1,i} \quad Y_j = \mu_2 + \epsilon_{2,j} \quad (4.13)$$

with $\epsilon_{i,j} \sim \text{iid } N_{0, \sigma^2}$. We can model the collection of variables as $X_1, \dots, X_m, Y_1, \dots, Y_n$ thus $R = \{1, \dots, m+n\}$. We have then

- $M = \{(\mu_1, \dots, \mu_1, \mu_2, \dots, \mu_2), \mu_1 \in \mathbb{R}, \mu_2 \in \mathbb{R}\}$ and $k = 2$
- $M_0 = \{(\mu, \dots, \mu, \mu, \dots, \mu), \mu \in \mathbb{R}\}$ and $k_0 = 1$
- $\Pi_M(x_1, \dots, x_m, y_1, \dots, y_n) = (\bar{x}, \dots, \bar{x}, \bar{y}, \dots, \bar{y})$, where $\bar{x} = (\sum_{i=1}^m x_i)/m$ and $\bar{y} = (\sum_{j=1}^n y_j)/n$.
- $\Pi_{M_0}(x_1, \dots, x_m, y_1, \dots, y_n) = (\bar{z}, \dots, \bar{z}, \bar{z}, \dots, \bar{z})$, where $\bar{z} = (\sum_{i=1}^m x_i + \sum_{j=1}^n y_j)/(m+n)$.

³A confidence interval for the p -value at level γ is given by Theorem 2.4 and is equal to $[0, \frac{3.689}{R}]$ where R is the number of replicates. We obtain that $p \leq 0.037$ at confidence $\gamma = 0.95$ thus we reject H_0 .

This model is an instance of what is called “one way ANOVA”.

EXAMPLE 4.9: NETWORK MONITORING. A network monitoring experiment tries to detect changes in user behaviour by measuring the number of servers inside the intranet accessed by users. Three groups were measured, with 16 measurements in each group. Only the average and standard deviations of the numbers of accessed servers are available:

Group	Mean number of remote servers	standard deviation
1	15.0625	3.2346
2	14.9375	3.5491
3	17.3125	3.5349

(here the **standard deviation** is $\sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2}$). The model is

$$X_{i,j} = \mu_i + \epsilon_{i,j} \quad 1 \leq n_i \quad i = 1, \dots, k \quad (4.14)$$

with $\epsilon_{i,j} \sim \text{iid } N_{0,\sigma^2}$. It is also called one-way ANOVA model (one way because there is one “factor”, index i). Here i represents the group, and j one measurement for one member of the group. The collection is $X_r = X_{i,j}$ so $R = \{(i, j), i = 1, \dots, k = 3 \text{ and } j = 1, \dots, n_i\}$ and $|R| = \sum_i n_i$. We have

- $M = \{(\mu_{i,j}), \text{ such that } \mu_{i,j} = \mu_i, \forall i, j\}$; the dimension of M is $k = 3$.
 - $M_0 = \{(\mu_{i,j}) \text{ such that } \mu_{i,j} = \mu, \forall i, j\}$ and $k_0 = 1$.
 - $\Pi_M(\vec{x})$ is the vector whose (i, j) th coordinate is independent of j and is equal to $\bar{x}_i \stackrel{\text{def}}{=} (\sum_{j=1}^{n_i} x_{i,j})/n_i$.
 - $\Pi_{M_0}(\vec{x})$ is the vector whose coordinates are all identical and equal to the overall mean $\bar{x}_{..} \stackrel{\text{def}}{=} (\sum_{i,j} x_{i,j})/|R|$.
-

THEOREM 4.1 (ANOVA). Consider an ANOVA model as defined above. The p-value of the likelihood ratio test of $H_0: \vec{\mu} \in M_0, \sigma > 0$ against $H_1: \vec{\mu} \in M \setminus M_0, \sigma > 0$ is $p^* = 1 - F_{k-k_0, |R|-k}(f)$ where $F_{m,n}()$ is the Fisher distribution with degrees of freedom m, n , \vec{x} is the dataset and

$$f = \frac{SS2/(k - k_0)}{SS1/(|R| - k)} \quad (4.15)$$

$$SS2 = \|\hat{\mu} - \hat{\mu}_0\|^2 \quad (4.16)$$

$$SS1 = \|\vec{x} - \hat{\mu}\|^2 \quad (4.17)$$

$$\hat{\mu}_0 = \Pi_{M_0}(\vec{x}) \quad (4.18)$$

$$\hat{\mu} = \Pi_M(\vec{x}) \quad (4.19)$$

(The norm is euclidian, i.e. $\|\vec{x}\|^2 = \sum_r x_r^2$.)

The theorem, the proof of which is a direct application of the general ANOVA theorem C.5, can be understood as follows. The maximum likelihood estimators under H_0 and H_1 are obtained by orthogonal projection:

$$\hat{\mu}_0 = \Pi_{M_0}(\vec{x}), \quad \hat{\sigma}_0^2 = \frac{1}{|R|} \|\vec{x} - \hat{\mu}_0\|^2$$

$$\hat{\mu} = \Pi_M(\vec{x}), \hat{\sigma}^2 = \frac{1}{|R|} \|\vec{x} - \hat{\mu}\|^2$$

The likelihood ratio statistic can be computed explicitly and is equal to $-\frac{|R|}{2} \ln \frac{SS1}{SS0} = \frac{|R|}{2} \ln (1 + \frac{SS2}{SS1})$, where $SS0 \stackrel{\text{def}}{=} \|\vec{x} - \hat{\mu}_0\|^2 = |R| \hat{\sigma}_0^2 = SS1 + SS2$. Under H_0 , the distribution of f , given by Eq.(4.15), is Fisher $F_{k-k_0, |R|-k}$, therefore we can compute the p -value exactly. The equality

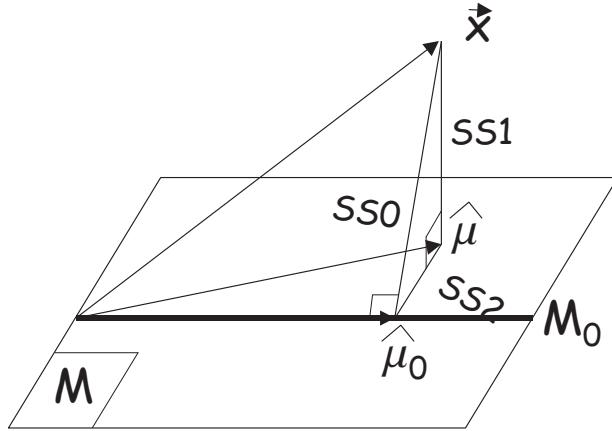


Figure 4.3: Illustration of quantities in Theorem 4.1

$SS0 = SS1 + SS2$ can be interpreted as a decomposition of sum of squares, as follows. Consider Θ_0 as the base model, with k_0 dimensions for the mean; we ask ourselves whether it is worth considering the more complex model Θ , which has $k > k_0$ dimensions for the mean. From its definition, we can interpret those sums of squares as follows.

- $SS2$ is the sum of squares explained by the model Θ , or explained variation.
- $SS1$ is the residual sum of squares
- $SS0$ is the total sum of squares

The likelihood ratio test accepts Θ when $SS2/SS1$ is large, i.e., when the percentage of sum of squares $SS2/SS1$ (also called percentage of variation) explained by the model Θ is high.

The dimensions are interpreted as degrees of freedom: $SS2$ (explained variation) is in the orthogonal of M_0 in M , with dimension $k - k_0$ and the number of degrees of freedom for $SS2$ is $k - k_0$; $SS1$ (residual variation) is the square of the norm of a vector that is orthogonal to M and the number of degrees of freedom for $SS1$ is $|R| - k$. This explains the name “ANOVA”: the likelihood ratio statistic depends only on estimators of variance. Note that this is very specific of homoscedasticity.

EXAMPLE 4.10: APPLICATION TO EXAMPLE 4.1, COMPILER OPTIONS. We assume homoscedasticity. We will check this hypothesis later by applying the test in Section 4.3.2. The theorem gives the following computations:

- $\hat{\mu} = (\bar{X}, \dots, \bar{X}, \bar{Y}, \dots, \bar{Y})$ and $\hat{\sigma} = \frac{1}{m+n}(\sum_i (X_i - \bar{X})^2 + \sum_j (Y_j - \bar{Y})^2)$
- $\hat{\mu}_0 = (\bar{Z}, \dots, \bar{Z}, \bar{Z}, \dots, \bar{Z})$ with $\bar{Z} = (m\bar{X} + n\bar{Y})/(m+n)$ and $\hat{\sigma}_0 = \frac{1}{m+n}(\sum_i (X_i - \bar{Z})^2 + \sum_j (Y_j - \bar{Z})^2)$

Parameter Set 1	SS	df	MS	F	Prob>F
Columns	13.2120	1	13.2120	13.4705	0.0003116
Errors	194.2003	198	0.9808		
total	207.4123	199			
Parameter Set 2	SS	df	MS	F	Prob>F
Columns	5.5975	1	5.5975	4.8813	0.0283
Errors	227.0525	198	1.1467		
total	232.6500	199			
Parameter Set 3	SS	df	MS	F	Prob>F
Columns	0.1892	1	0.1892	0.1835	0.6689
Errors	204.2256	198	1.0314		
total	204.4148	199			

Table 4.1: ANOVA Tests for Example 4.1 (Non Paired Data)

- $SS1 = \sum_i (X_i - \bar{X})^2 + \sum_j (Y_j - \bar{Y})^2 = S_{XX} + S_{YY}$
- $SS2 = m(\bar{Z} - \bar{X})^2 + n(\bar{Z} - \bar{Y})^2 = (\bar{X} - \bar{Y})^2 / (1/m + 1/n)$
- the f value is $\frac{SS2}{SS1/(m+n-2)}$.

The ANOVA tables for parameter sets 1 to 3 are given in Table 4.1. The F-test rejects the hypothesis of same mean for parameter sets 1 and 2, and accepts it for parameter set 3. The software used to produce this example uses the following terminology:

- SS2: “Columns” (explained variation, variation between columns, or between groups)
- SS1: “Error” (residual variation, unexplained variation)
- SS0: “Total” (total variation)

QUESTION 4.3.1. Compare to the confidence intervals given in the introduction.⁴

QUESTION 4.3.2. What are SS0, SS1 and SS2 for parameter set 1 ?⁵

EXAMPLE 4.11: **NETWORK MONITORING.** The numerical solution of Example 4.9 is shown in the table below. Thus we accept H_0 , namely, the three measured groups are similar, though the evidence is not strong.

Source	SS	df	MS	F	Prob>F
Columns	57.1667	2	28.5833	2.4118	0.1012
Errors	533.3140	45	11.8514		
Total	590.4807	47			

QUESTION 4.3.3. Write down the expressions of MLEs, SS1, SS2 and the F-value.⁶

⁴For parameter set 1, the conclusion is the same as with confidence interval. For parameter sets 2 and 3, confidence intervals did not allow one to conclude. ANOVA disambiguates these two cases.

⁵The column “SS” gives, from top to bottom: SS2, SS1 and SS0.

⁶

STUDENT TEST AS SPECIAL CASE OF ANOVA. In the special case where $k - k_0 = 1$ (as in Example 4.1) the F -statistic is the square of a student statistic, and a student test could be used instead. This is used by some statistics packages.

TESTING FOR SPECIFIC VALUES By an additive change of variable, we can extend the ANOVA framework to the case where $M_0 \subset M$ are affine (instead of linear) varieties of \mathbb{R}^R . This includes testing for a specific value. For example, assume we have the model

$$X_{i,j} = \mu_i + \epsilon_{i,j} \quad (4.20)$$

with $\epsilon_{i,j} \sim \text{iid } N_{0,\sigma^2}$. We want to test

H_0 : “ $\mu_i = \mu_0$ for all i ” against H_1 : “ μ_i unconstrained”

We change model by letting $X'_{i,j} = X_{i,j} - \mu_0$ and we are back to the ANOVA framework.

4.3.2 TESTING FOR A COMMON VARIANCE

We often need to verify that the common variance assumption holds. Here too, a likelihood ratio test gives the answer. In the general case, the p -value of the test cannot be computed in closed form, so we use either Monte Carlo simulation or an asymptotic approximation. When the number of groups is 2, there is a closed form using the Fisher distribution.

We are given a data set with I groups $x_{i,j}$, $i = 1, \dots, I$, $j = 1, \dots, n_i$; the total number of samples is $n = \sum_{j=1}^I n_j$. We assume that it is a realization of the model $X_{i,j} \sim \text{iid } N_{\mu_i, \sigma_i^2}$. We assume that the normal assumption holds and want to test

H_0 : $\sigma_i = \sigma > 0$ for all i against H_1 : $\sigma_i > 0$

- $\hat{\mu}$ is the vector whose (i, j) th coordinate is independent of j and is equal to $\bar{X}_{i..} \stackrel{\text{def}}{=} \sum_{j=1}^{n_i} X_{i,j} / n_i$.
- $SS1 = \sum_{i,j} (X_{i,j} - \bar{X}_{i..})^2$
- $\hat{\sigma}^2 = \frac{1}{|R|} SS1$
- $\hat{\mu}_0$ is the vector whose coordinates are all identical and equal to the overall mean $\bar{X}_{..} \stackrel{\text{def}}{=} (\sum_{i,j} X_{i,j}) / |R|$
- $SS2 = \sum_i n_i (\bar{X}_{i..} - \bar{X}_{..})^2$
- $SS0 = SS1 + SS2$
- $\hat{\sigma}_0^2 = \frac{1}{|R|} SS0$
- $F = SS2(|R| - k) / [SS1(k - 1)]$

THEOREM 4.2 (Testing for Common Variance). *The likelihood ratio statistic ℓ of the test of common variance under the hypothesis above is given by*

$$\begin{aligned} 2\ell &= n \ln(s^2) - \sum_{i=1}^I n_i \ln(s_i^2) \\ \text{with } \hat{\mu}_i &\stackrel{\text{def}}{=} \frac{1}{n_i} \sum_{j=1}^I x_{i,j}, \quad s_i^2 \stackrel{\text{def}}{=} \frac{1}{n_i} \sum_{j=1}^I (x_{i,j} - \hat{\mu}_i)^2, \\ s^2 &\stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^I \sum_{j=1}^{n_i} (x_{i,j} - \hat{\mu}_i)^2 = \sum_{i=1}^I \frac{n_i}{n} s_i^2 \end{aligned} \quad (4.21)$$

The test rejects H_0 when ℓ is large. The p-value is

$$p = \mathbb{P} \left(n \log \sum_{i=1}^I Z_i - \sum_{i=1}^I n_i \log Z_i > 2\ell + n \log n - \sum_{i=1}^I n_i \log n_i \right) \quad (4.22)$$

where Z_i are independent random variables, $Z_i \sim \chi_{n_i-1}^2$ and $Z = \sum_{i=1}^I \frac{n_i}{n} Z_i$. The p-value can be computed by Monte Carlo simulation. When n is large:

$$p \approx 1 - \chi_{I-1}^2(2\ell) \quad (4.23)$$

In the special case $I = 2$, we can replace the statistic ℓ by

$$f \stackrel{\text{def}}{=} \frac{\hat{\sigma}_1^2}{\hat{\sigma}_2^2} \quad \text{with } \hat{\sigma}_i^2 \stackrel{\text{def}}{=} \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (x_{i,j} - \hat{\mu}_i)^2$$

and the distribution of f under H_0 is Fisher F_{n_1-1, n_2-1} . The test at size α rejects H_0 when $f < \eta$ or $f > \xi$ with $F_{n_1-1, n_2-1}(\eta) = \alpha/2$, $F_{n_1-1, n_2-1}(\xi) = 1 - \alpha/2$. The p-value is

$$p = F_{n_1-1, n_2-1}(\min(f, 1/f)) - F_{n_1-1, n_2-1}(\max(f, 1/f)) + 1 \quad (4.24)$$

EXAMPLE 4.12: NETWORK MONITORING AGAIN. We want to test whether the data in groups 1 and 2 in Example 4.9 have the same variance. We have $\eta = 0.3494$, $\xi = 2.862$; the F statistic is 0.8306 so we accept H_0 , i.e. that the variance is the same. Alternatively, we can use Eq.(4.24) and find $p = 0.7239$, which is large so we accept H_0 .

Of course, we are more interested in comparing the 3 groups together. We apply Eq.(4.21) and find as likelihood ratio statistic $\ell = 0.0862$. The asymptotic approximation gives $p \approx 0.9174$, but since the number of samples n is not large we do not trust it. We evaluate Eq.(4.22) by Monte Carlo simulation; with $R = 10^4$ replicates we find a confidence interval for the estimated p-value of $[0.9247; 0.9347]$. We conclude that the p-value is very large so we accept that the variance is the same.

4.4 ASYMPTOTIC RESULTS

In many cases it is hard to find the exact distribution of a test statistic. An interesting feature of likelihood ratio tests is that we have a simple asymptotic result. We used this result already in the test for equal variance in Section 4.3.2.

4.4.1 LIKELIHOOD RATIO STATISTIC

The following theorem derives immediately from Theorem B.2.

THEOREM 4.3. [32] Consider a likelihood ratio test (Section 4.2) with $\Theta = \Theta_1 \times \Theta_2$, where Θ_1, Θ_2 are open subsets of $\mathbb{R}^{q_1}, \mathbb{R}^{q_2}$ and denote $\theta = (\theta_1, \theta_2)$. Consider the likelihood ratio test of $H_0 : \theta_2 = 0$ against $H_1 : \theta_2 \neq 0$. Assume that the conditions in Definition B.1 hold. Then, approximately, for large sample sizes, under H_0 , $2\text{lr}_{\text{s}} \sim \chi_{q_2}^2$, where lr_{s} is the likelihood ratio statistic.

It follows that the p -value of the likelihood ratio test can be approximated for large sample sizes by

$$p^* \approx 1 - \chi_{q_2}^2(2\text{lr}_{\text{s}}) \quad (4.25)$$

where q_2 is the number of degrees of freedom that H_1 adds to H_0 .

EXAMPLE 4.13: APPLICATION TO EXAMPLE 4.1 (COMPILER OPTIONS). Using Theorem 4.1 and Theorem 4.3 we find that

$$2\text{lr}_{\text{s}} \stackrel{\text{def}}{=} N \ln \left(1 + \frac{SS_2}{SS_1} \right) \sim \chi_1^2$$

The corresponding p -values are:

```
Parameter Set 1 pchi2 = 0.0002854
Parameter Set 2 pchi2 = 0.02731
Parameter Set 3 pchi2 = 0.6669
```

They are all very close to the exact values (given by ANOVA in Table 4.1).

4.4.2 PEARSON CHI-SQUARED STATISTIC AND GOODNESS OF FIT

We can apply the large sample asymptotic to goodness of fit tests as defined in Section 4.2.3. This gives a simpler way to compute the p -value, and allows to extend the test to the *composite goodness of fit* test, defined as follows.

COMPOSITE GOODNESS OF FIT Similar to Section 4.2.3, assume we are given n data points x_1, \dots, x_n , generated from an iid sequence, and we want to verify whether their common distribution comes from a given family of distributions $F(\cdot|\theta)$ where the parameter θ is in some set Θ_0 . We say that the test is composite because the null hypothesis has several possible values of θ . We compare the empirical histograms: we partition the set of values of \vec{X} into *bins* B_i , $i = 1 \dots I$. Let

$N_i = \sum_{k=1}^n \mathbf{1}_{\{B_i\}}(X_k)$ (number of observation that fall in bin B_i) and $q_i = \mathbb{P}_\theta\{X_1 \in B_i\}$. If the data comes from a distribution $F(\cdot|\theta)$ the distribution of N_i is multinomial $M_{n,\vec{q}(\theta)}$. The likelihood ratio statistic test is

H_0 : N_i comes from a multinomial distribution $M_{n,\vec{q}(\theta)}$, with $\theta \in \Theta_0$

against

H_1 : N_i comes from a multinomial distribution $M_{n,\vec{p}}$ for some arbitrary \vec{p} .

We now compute the likelihood ratio statistic. The maximum likelihood estimator of the parameter under H_1 is the same as in Section 4.2.3. Let $\hat{\theta}$ be the maximum likelihood estimator of θ under H_0 . The likelihood ratio statistic is thus

$$lrs = \sum_{i=1}^k n_i \ln \frac{n_i}{n q_i(\hat{\theta})} \quad (4.26)$$

The p -value is

$$\sup_{\theta \in \Theta_0} \mathbb{P} \left(\sum_{i=1}^k N_i \ln \frac{N_i}{n q_i} > \sum_{i=1}^k n_i \ln \frac{n_i}{n q_i(\hat{\theta})} \right) \quad (4.27)$$

where \vec{N} has the multinomial distribution $M_{n,\vec{q}(\hat{\theta})}$. It can be computed by Monte Carlo simulation as in the case of a simple test, but this may be difficult because of the supremum.

An alternative for large n is to use the asymptotic result in Theorem 4.3. It says that, for large n , under H_0 , the distribution of $2lrs$ is approximately $\chi_{q_2}^2$, with q_2 = the number of degrees of freedom that H_1 adds to H_0 . Here H_0 has k_0 degrees of freedom (where k_0 is the dimension of Θ_0) and H_1 has $I - 1$ degrees of freedom (where I is the number of bins). Thus the p -value of the test is approximately

$$1 - \chi_{I-k_0-1}^2(2lrs) \quad (4.28)$$

EXAMPLE 4.14: IMPACT OF ESTIMATION OF (μ, σ) . We want to test whether the data set on the right of Figure 4.4 has a normal distribution. We use a histogram with 10 bins. We need first to estimate $\hat{\theta} = (\hat{\mu}, \hat{\sigma})$.

1. Assume we do this by fitting a line to the qqplot. We obtain $\hat{\mu} = -0.2652, \hat{\sigma} = 0.8709$. The values of $n q_i(\hat{\theta})$ and n_i are:

7.9297	7.0000
11.4034	9.0000
18.0564	17.0000
21.4172	21.0000
19.0305	14.0000
12.6672	17.0000
6.3156	6.0000
2.3583	4.0000
0.6594	3.0000
0.1624	2.0000

The likelihood ratio statistic as in Eq.(4.26) is $lrs = 7.6352$. The p -value is obtained using a χ_7^2 distribution ($q_2 = 10 - 2 - 1$): $p1 = 0.0327$, thus we would reject normality at size 0.05.

2. It might not be good to simply fit (μ, σ) on the qqplot. A better way is to use estimation theory, which suggests to find (μ, σ) that maximizes the log likelihood of the model. This is equivalent to minimizing the likelihood ratio statistic $l_{H_1}(\vec{x}) - l_{\mu, \sigma}(\vec{x})$ (note that the value of $l_{H_1}(\vec{x})$ is easy to compute). We do this with a numerical optimization procedure and find now $\hat{\mu} = -0.0725, \hat{\sigma} = 1.0269$. The corresponding values of $nq_i(\hat{\theta})$ and n_i are now:

8.3309	7.0000
9.5028	9.0000
14.4317	17.0000
17.7801	21.0000
17.7709	14.0000
14.4093	17.0000
9.4783	6.0000
5.0577	4.0000
2.1892	3.0000
1.0491	2.0000

Note how the true value of $\hat{\mu}, \hat{\sigma}$ provides a better fit to the tail of the histogram. The likelihood ratio statistic is now $lrs = 2.5973$, which also shows a much better fit. The p -value, obtained using a χ^2_7 distribution is now $p1 = 0.6362$, thus we accept that the data is normal.

3. Assume we would ignore that (μ, σ) is estimated from the data, but would do as if the test were a simple goodness of fit test, with H_0 : “The distribution is $N_{-0.0725, 1.0269}$ ” instead of H_0 : “The distribution is normal”. We would compute the p -value using a χ^2_9 distribution ($q_2 = 10 - 1$) and would obtain: $p2 = 0.8170$, a value larger than the true p -value. This is quite general: if we estimate some parameter and pretend it is a priori known, then we overestimate the p -value.

PEARSON CHI-SQUARED STATISTIC. In the case where n is large, $2 \times$ the likelihood ratio statistic can be replaced by the *Pearson chi-squared statistic*, which has the same asymptotic distribution. It is defined by

$$pcs = \sum_{i=1}^I \frac{(n_i - nq_i(\hat{\theta}))^2}{nq_i(\hat{\theta})} \quad (4.29)$$

Indeed, when n is large we expect, under H_0 that $n_i - nq_i(\hat{\theta})$ is relatively small, i.e. $\epsilon_i = \frac{n_i}{nq_i(\hat{\theta})} - 1$ is small. An approximation of $2lrs$ is found from the second order development around $\epsilon = 0$: $\ln(1 + \epsilon) = \epsilon - \frac{1}{2}\epsilon^2 + o(\epsilon^2)$ and thus

$$\begin{aligned} lrs &= \sum_i n_i \frac{n_i}{nq_i(\hat{\theta})} n \sum_i (1 + \epsilon_i) q_i(\hat{\theta}) \ln(1 + \epsilon_i) \\ &= n \sum_i \left(\epsilon_i - \frac{1}{2}\epsilon_i^2 + o(\epsilon_i^2) (1 + \epsilon_i) q_i(\hat{\theta}) \right) \\ &= n \sum_i q_i(\hat{\theta}) \epsilon_i \left(1 - \frac{1}{2}\epsilon_i + o(\epsilon_i)(1 + \epsilon_i) \right) \\ &= n \sum_i q_i(\hat{\theta}) \epsilon_i \left(1 + \frac{1}{2}\epsilon_i + o(\epsilon_i) \right) \\ &= n \sum_i q_i(\hat{\theta}) \epsilon_i + n \sum_i q_i(\hat{\theta}) \frac{1}{2}\epsilon_i^2 + n \sum_i o(\epsilon_i^2) \end{aligned}$$

Note that $\sum_i q_i(\hat{\theta})\epsilon_i = 0$ thus

$$lrs \approx \frac{1}{2}pcs \quad (4.30)$$

The Pearson Chi-squared statistic was historically developed before the theory of likelihood ratio tests, which explains why it is commonly used.

In summary, for large n , the composite goodness of fit test is solved by computing either $2lrs$ or pcs . The p -value is $1 - \chi_{n-k_0-1}^2(2lrs)$ or $1 - \chi_{I-k_0-1}^2(pcs)$. If either is small, we reject H_0 , i.e. we reject that the distribution of X_i comes from the family of distributions $F(|\theta)$.

SIMPLE GOODNESS OF FIT TEST. This is a special case of the composite test. In this case $q_2 = I - 1$ and thus the p -value of the test (given in Eq.(4.11) can be approximated for large n by $1 - \chi_{I-1}^2(2lrs)$ or $\chi_{I-1}^2(pcs)$. Also, the likelihood ratio statistic $\sum_{i=1}^k n_i \ln \frac{n_i}{nq_i}$ can be replaced by the Pearson-Chi-Squared statistic, equal to

$$\sum_{i=1}^I \frac{(n_i - nq_i)^2}{nq_i} \quad (4.31)$$

EXAMPLE 4.15: MENDEL'S PEAS, CONTINUATION OF EXAMPLE 4.7. The likelihood ratio statistic is $lrs = 0.3092$ and we found by Monte Carlo a p -value $p^* = 0.9191 \pm 0.0458$. By the asymptotic result, we can approximate the p -value by $\chi_3^2(2lrs) = 0.8922$.

The Pearson Chi-squared statistic is $pcs = 0.6043$, very close to $2lrs = 0.618$. The corresponding p -value is 0.8954.

4.4.3 TEST OF INDEPENDENCE

The same ideas as in Section 4.4.2 can be applied to a *test of independence*. We are given a sequence (x_k, y_k) , which we interpret as a sample of the sequence (X_k, Y_k) , $k = 1, \dots, n$. The sequence is iid ((X_k, Y_k) is independent of $(X_{k'}, Y_{k'})$ for $k \neq k'$ and both have the same distribution). We are interested in knowing whether X_k is independent of Y_k .

To this end, we compute an empirical histogram of (X, Y) , as follows. We partition the set of values of X [resp. Y] into I [resp. J] bins B_i [resp. C_j]. Let $N_{i,j} = \sum_{k=1}^n \mathbf{1}_{\{B_i\}}(X_k) \mathbf{1}_{\{C_j\}}(Y_k)$ (number of observation that fall in bin (B_i, C_j)) and $p_{i,j} = \mathbb{P}\{X_1 \in B_i \text{ and } Y_1 \in C_j\}$. The distribution of N is multinomial. The test of independence is

H_0 : “ $p_{i,j} = q_i r_j$ for some q and r such that $\sum_i q_i = \sum_j r_j = 1$ ”

against

H_1 : “ $p_{i,j}$ is arbitrary”

The maximum likelihood estimator under H_0 is $\hat{p}_{i,j}^0 = \frac{n_{i,j}}{n} \frac{n_j}{n}$ where $n_{i,j} = \sum_{k=1}^n \mathbf{1}_{\{B_i\}}(X_k) \mathbf{1}_{\{C_j\}}(Y_k)$ and

$$\begin{cases} n_i = \sum_j n_{i,j} \\ n_j = \sum_i n_{i,j} \end{cases} \quad (4.32)$$

The maximum likelihood estimator under H_1 is $\hat{p}_{i,j}^1 = \frac{n_{i,j}}{n}$. The likelihood ratio statistic is thus

$$lrs = \sum_{i,j} n_{i,j} \ln \frac{nn_{i,j}}{n_i n_{\cdot j}} \quad (4.33)$$

To compute the p -value, we use, for large n , a $\chi^2_{q_2}$ distribution. The numbers of degrees of freedom under H_1 is $IJ - 1$, under H_0 it is $(I - 1) + (J - 1)$, thus $q_2 = (IJ - 1) - (I - 1) - (J - 1) = (I - 1)(J - 1)$. The p -value is thus

$$p^* = \left(1 - \chi^2_{(I-1)(J-1)}\right) (2lrs) \quad (4.34)$$

As in Section 4.4.2, $2lrs$ can be replaced, for large n , by the Pearson Chi-squared statistic:

$$pcs = \sum_{i,j} \frac{\left(n_{i,j} - \frac{n_i n_{\cdot j}}{n}\right)^2}{\frac{n_i n_{\cdot j}}{n}} \quad (4.35)$$

EXAMPLE 4.16: BRASSICA OLERACEA GEMMIFERA. A survey was conducted at the campus cafeteria, where customers were asked whether they like Brussels sprouts. The answers are:

$i \setminus j$	Male		Female		<i>Total</i>	
Likes	454	44.69%	251	48.08%	705	45.84%
Dislikes	295	29.04%	123	23.56%	418	27.18%
No Answer / Neutral	267	26.28%	148	28.35%	415	26.98%
<i>Total</i>	1016		522		1538	
	100%		100%		100 %	

We would like to test whether affinity to Brussels sprouts is independent of customer's gender. Here we have $I = 3$ and $J = 2$, so we use a χ^2 distribution with $q_2 = 2$ degrees of freedom. The likelihood ratio statistic and the p -value are

$$lrs = 2.6489, \quad p = 0.0707 \quad (4.36)$$

so we accept H_0 , i.e. affinity to Brussels sprouts is independent of gender. Note that the Pearson Chi-squared statistic is

$$pcs = 5.2178 \quad (4.37)$$

which is very close to $2lrs$.

4.5 OTHER TESTS

4.5.1 GOODNESS OF FIT TESTS BASED ON AD-HOC PIVOTS

In addition to the Pearson χ^2 test, the following two tests are often used. They apply to a continuous distribution, thus do not require quantizing the observations. Assume $X_i, i = 1, \dots, n$ are iid samples. We want to test H_0 : the distribution of X_i is F against non H_0 .

Define the empirical distribution \hat{F} by

$$\hat{F}(x) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^n 1_{\{X_i \leq x\}} \quad (4.38)$$

Kolmogorov-Smirnov A *pivot* is a function of the data whose probability distribution under H_0 is the same for all $\theta \in \Theta_0$. For this test the pivot is

$$T = \sup_x |\hat{F}(x) - F(x)|$$

That the distribution of this random variable is independent of F is not entirely obvious, but can be derived easily in the case where F is continuous and strictly increasing, as follows. The idea is to change the scale on the x -axis by $u = F(x)$. Formally, define

$$U_i = F(X_i)$$

so that $U_i \sim U(0, 1)$. Also

$$\hat{F}(x) = \frac{1}{n} \sum_i 1_{\{X_i \leq x\}} = \frac{1}{n} \sum_i 1_{\{U_i \leq F(x)\}} = \hat{G}(F(x))$$

where \hat{G} is the empirical distribution of the sample U_i , $i = 1, \dots, n$. By the change of variable $u = F(x)$, it comes

$$T = \sup_{u \in [0,1]} |\hat{G}(u) - u|$$

which shows that the distribution of T is independent of F . Its distribution is tabulated in statistical software packages. For a large n , its tail can be approximated by $\tau \approx \sqrt{-(\ln \alpha)/2}$ where $\mathbb{P}(T > \tau) = \alpha$.

Anderson-Darling Here the pivot is

$$A = n \int_{\mathbb{R}} \frac{(\hat{F}(x) - F(x))^2}{F(x)(1 - F(x))} dF(x)$$

The test is similar to K-S but is less sensitive to outliers.

QUESTION 4.5.1. Show that A is indeed a pivot.⁷

EXAMPLE 4.17: FILE TRANSFER DATA. We would like to test whether the data in Figure 4.4 and its log are normal. We cannot directly apply Kolmogorov Smirnov since we do not know exactly in advance the parameters of the normal distribution to be tested against. An approximate method is to estimate the slope and intercept of the straight line in the qqplot. We obtain

```
Original Data
slope      =      0.8155
intercept =      1.0421
```

```
Transformed Data
slope      =      0.8709
intercept =     -0.2652
```

⁷Use the fact that $\hat{F}(x) = \hat{G}(F(x))$ and do the change of variable $u = F(x)$ in the integral.

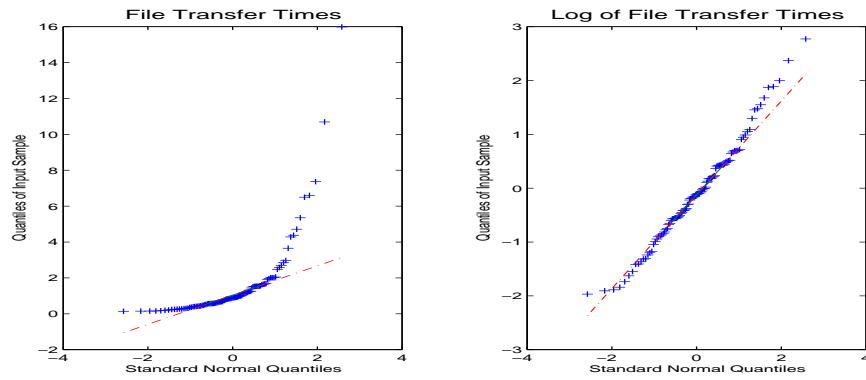


Figure 4.4: Normal qqplots of file transfer data and its logarithm.

For example, this means that for the original data we take for H_0 : “the distribution is $N(\mu = 1.0421, \sigma^2 = 0.8155^2)$ ”. We can now use the Kolmogorov-Smirnov test and obtain

Original Data	
$h = 1$	$p = 0.0493$
Transformed Data	
$h = 0$	$p = 0.2415$

Thus the test rejects the normality assumption for the original data and accepts it for the transformed data.

This way of doing is approximate in that we used estimated parameters for H_0 . This introduces some bias, similar to using the normal statistic instead of student when we have a normal sample. The bias should be small when the data sample is large, which is the case here.

A fix to this problem is to use a variant of KS, for example the Lilliefors test, or to use different normality tests such as Jarque Bera (see Example 4.18) or Shapiro-Wilk. The Lilliefors test is a heuristic that corrects the p -value of the KS to account for the uncertainty due to estimation. In this specific example, with the Lilliefors test we obtain the same results.

JARQUE-BERA. The *Jarque-Bera* statistic is used to test whether an iid sample comes from a normal distribution. It uses the skewness and kurtosis indices γ_1 and γ_2 defined in Section 3.4.2. The test statistic is equal to $\frac{n}{6} \left(\hat{\gamma}_1^2 + \frac{\hat{\gamma}_2^2}{4} \right)$, the distribution of which is asymptotically χ_2^2 for large sample size n . In the formula, $\hat{\gamma}_1$ and $\hat{\gamma}_2$ are the sample indices of skewness and kurtosis, obtained by replacing expectations by sample averages in Eq.(3.13).

EXAMPLE 4.18: APPLICATION TO EXAMPLE 4.17. We would like to test whether the data in Example 4.17 and its transform are normal.

Original Data	$h = 1$	$p = 0.0010$
Transformed Data	$h = 0$	$p = 0.1913$

The conclusions are the same as in Example 4.17, but for the original data the normality assumption is clearly rejected, whereas it was borderline in Example 4.17.

4.5.2 ROBUST TESTS

We give two examples of test that make no assumption on the distribution of the sample (but assume it is iid). They are *non parametric* in the sense that they do not assume a parameterized family of densities.

MEDIAN TEST The model is $X_i \sim \text{iid}$ with some distribution $F()$ with a density. We want to test

H_0 : “the median of F is 0” against H_1 : “unspecified”

A simple test is based on confidence interval, as mentioned in Section 4.1.4. Let $I(\vec{x})$ be a confidence interval for the median (Theorem 2.1). We reject H_0 if

$$0 \notin I(\vec{x}) \quad (4.39)$$

This test is robust in the sense that it makes no assumption other than independence.

WILCOXON SIGNED RANK TEST. It is used for testing equality of distribution in paired experiments. It tests

H_0 : X_1, \dots, X_n is iid with a common symmetric, continuous distribution, the median of which is 0

against

H_1 : X_1, \dots, X_n is iid with a common symmetric, continuous distribution

The *Wilcoxon Signed Rank* Statistic is

$$W = \sum_{j=1}^n \text{rank}(|X_j|) \text{sign}(X_j)$$

where $\text{rank}(|X_j|)$ is the rank in increasing order (the smallest value has rank 1) and $\text{sign}(X_j)$ is -1 for negative data, $+1$ for positive, and 0 for null data. If the median is positive, then many values with high rank will be positive and W will tend to be positive and large. We reject the null hypothesis when $|W|$ is large.

It can be shown that the distribution of W under H_0 is always the same. It is tabulated and contained in software packages. For non small data samples, it can easily be approximated by a normal distribution. The mean and variance under can easily be computed:

$$\mathbb{E}_{H_0}(W) = \sum_{j=1}^n \mathbb{E}_{H_0}(\text{rank}(|X_j|)) \mathbb{E}_{H_0}(\text{sign}(X_j))$$

since under H_0 $\text{rank}(|X_j|)$ is independent of $\text{sign}(X_j)$. Thus $E_{H_0}(W) = 0$. The variance is

$$\mathbb{E}_{H_0}(W^2) = \sum_{j=1}^n \mathbb{E}_{H_0}(\text{rank}(|X_j|)^2 \text{sign}(X_j)^2) = \sum_{j=1}^n \mathbb{E}_{H_0}(\text{rank}(|X_j|)^2)$$

since $\text{sign}(X_j)^2 = 1$. Now $\sum_j \text{rank}(|X_j|)^2 = \sum_j j^2$ is non random thus

$$\text{var}_{H_0}(W) = \sum_{j=1}^n \mathbb{E}_{H_0}(\text{rank}(|X_j|)^2) = \mathbb{E}_{H_0}\left(\sum_j \text{rank}(|X_j|)^2\right) = \sum_{j=1}^n j^2 = \frac{n(n+1)(2n+1)}{6}$$

For large n , the test at size α rejects H_0 if $|W| > \eta \sqrt{\frac{n(n+1)(2n+1)}{6}}$ with $N_{0,1}(\eta) = 1 - \frac{\alpha}{2}$ (e.g. $\eta = 1.96$ at size 0.05). The p -value is:

$$p = 2 \left(1 - N_{0,1} \left(\frac{|W|}{\sqrt{\frac{n(n+1)(2n+1)}{6}}} \right) \right) \quad (4.40)$$

EXAMPLE 4.19: PAIRED DATA. This is a variant of Example 4.2. Consider again the reduction in run time due to a new compiler option, as given in Figure 2.7 on Page 32. We want to test whether the reduction is significant. We assume the data is iid, but not necessarily normal. The median test gives a confidence interval

$$I(\vec{x}) = [2.9127; 33.7597]$$

which does not contain 0 so we reject H_0 .

Alternatively, let us use the Wilcoxon Signed Rank test. We obtain the p -value

$$p = 2.3103e-005$$

and thus this test also rejects H_0 .

WILCOXON RANK SUM TEST AND KRUSKAL-WALLIS. The *Wilcoxon Rank Sum Test* is used for testing equality of distribution in non paired experiments. It tests

H_0 : the two samples come from the same continuous distribution
against

H_1 : the distributions of the two samples are continuous and differ by a location shift

Let $X_i^1, i = 1 \dots n_1$ and $X_i^2, i = 1 \dots n_2$ be the two iid sequences that the data is assumed to be a sample of. The *Wilcoxon Rank Sum Statistic* R is the sum of the ranks of the first sample in the concatenated sample.

As for the Wilcoxon signed rank test, its distribution under the null hypothesis depends only on the sample sizes and can be tabulated or, for a large sample size, approximated by a normal distribution. The mean and variance under H_0 are

$$m_{n_1, n_2} = \frac{n_1(n_1 + n_2 + 1)}{2} \quad (4.41)$$

$$v_{n_1, n_2} = \frac{n_1 n_2 (n_1 + n_2 + 1)}{12} \quad (4.42)$$

We reject H_0 when the rank sum statistic deviates largely from its expectation under H_0 . For large n_1 and n_2 , the p -value is

$$p = 2 \left(1 - N_{0,1} \left(\frac{|R - m_{n_1, n_2}|}{\sqrt{v_{n_1, n_2}}} \right) \right) \quad (4.43)$$

EXAMPLE 4.20: NON PAIRED DATA. The Wilcoxon rank sum test applied to Example 4.1 gives the following p -values:

Parameter Set 1 $p = 0.0002854$
 Parameter Set 2 $p = 0.02731$
 Parameter Set 3 $p = 0.6669$

The results are the same as with ANOVA. H_0 (same distribution) is accepted for the 3rd data set only, at size= 0.05.

The **Kruskal-Wallis** test is a generalization of Wilcoxon Rank Sum to more than 2 non paired data series. It tests (H_0): the samples come from the same distribution against (H_1): the distributions may differ by a location shift.

TURNING POINT TEST

This is a test of iid-ness. It tests

H_0 : X_1, \dots, X_n is iid

against

H_1 : X_1, \dots, X_n is not iid

We say that the vector X_1, \dots, X_n is monotonic at index i ($i \in \{2, \dots, n-1\}$) if

$$X_{i-1} \leq X_i \leq X_{i+1} \text{ or } X_{i-1} \geq X_i \geq X_{i+1}$$

and we say that there is a **turning point** at i if the vector X_1, \dots, X_n is not monotonic at i . Under H_0 , the probability of a turning point at i is $2/3$ (to see why, list all possible cases for the relative orderings of X_{i-1}, X_i, X_{i+1}).

More precisely, let T be the number of turning points in X_1, \dots, X_n . It can be shown [18, 105] that, for large n , T is approximately $N_{\frac{2n-4}{3}, \frac{16n-29}{90}}$. Thus the p -value is, approximatively for large n :

$$p = 2 \left(1 - N_{0,1} \left(\frac{|T - \frac{2n-4}{3}|}{\sqrt{\frac{16n-29}{90}}} \right) \right) \quad (4.44)$$

4.6 PROOFS

PROOF OF THEOREM 4.2

We make a likelihood ratio test and compute the likelihood ratio statistic. We need first to compute the maximum likelihood under H_1 . The log-likelihood of the model is

$$l_{\vec{x}}(\vec{\mu}, \vec{\sigma}) = -\frac{1}{2} \left[\ln(2\pi) + \sum_{i=1}^I \left(2n_i \ln(\sigma_i) + \sum_{j=1}^{n_i} \frac{(\vec{x}_{i,j} - \mu_i)^2}{\sigma_i^2} \right) \right] \quad (4.45)$$

To find the maximum under H_1 , observe that the terms in the summation do not have cross dependencies, thus we can maximize each of the I terms separately. The maximum of the i th term is for $\mu_i = \hat{\mu}_i$ and $\sigma_i^2 = s_i^2$, and thus

$$l_{\vec{x}}(H_1) = -\frac{1}{2} \left[\ln(2\pi) + \sum_{i=1}^I n_i (2 \ln(s_i) + 1) \right] = -\frac{1}{2} \left[\ln(2\pi) + n + 2 \sum_{i=1}^I n_i \ln(s_i) \right] \quad (4.46)$$

Under H_0 the likelihood is as in Eq.(4.45) but with σ_i replaced by the common value σ . To find the maximum, we use the ANOVA theorem C.5. The maximum is for $\mu_i = \hat{\mu}_i$ and $\sigma^2 = s^2$ and thus

$$l_{\vec{x}}(H_0) = -\frac{1}{2} \left[\ln(2\pi) + \sum_{i=1}^I n_i \frac{s_i^2}{s^2} + 2n \ln(s) \right] = -\frac{1}{2} [\ln(2\pi) + n + 2n \ln(s)] \quad (4.47)$$

The test statistic is the likelihood ratio statistic $\ell = l_{\vec{x}}(H_1) - l_{\vec{x}}(H_0)$: and thus

$$2\ell = n \ln(s^2) - \sum_{i=1}^I n_i \ln(s_i^2) \quad (4.48)$$

The test has the form: reject H_0 when $2\ell > K$ for some constant K . The p -value can be obtained using Monte-Carlo simulation. The problem is now to compute $\mathbb{P}(T > 2\ell)$ where T is a random variable distributed like

$$n \ln(s^2) - \sum_{i=1}^I n_i \ln(s_i^2) \quad (4.49)$$

and assuming H_0 holds. Observe that all we need is to generate the random variables s_i^2 . They are independent, and $Z_i = n_i s_i$ is distributed like $\sigma^2 \chi_{n_i-1}^2$ (Corollary C.3). Note that T is independent of the specific value of the unknown but fixed parameter σ , thus we can let $\sigma = 1$ in the Monte Carlo simulation, which proves Eq.(4.22). Alternatively, one can use the large sample asymptotic in Theorem 4.3, which gives Eq.(4.23).

When $I = 2$ we can rewrite the likelihood ratio statistic as

$$\ell = \frac{1}{2} [n \ln(n_1 F + n_2) - n_1 \ln(F)] + C \quad (4.50)$$

where C is a constant term (assuming n_1 and n_2 are fixed) and $F = \frac{s_1^2}{s_2^2}$. The derivative of ℓ with respect to F is

$$\frac{\partial \ell}{\partial F} = \frac{n_1 n_2 (F - 1)}{2F(n_1 F + n_2)} \quad (4.51)$$

thus ℓ decreases with F for $F < 1$ and increases for $F > 1$. Thus the rejection region, defined as $\{\ell > K\}$, is also of the form $\{F < K_1 \text{ or } F > K_2\}$. Now define

$$f = \frac{\hat{\sigma}_1^2}{\hat{\sigma}_2^2} \quad (4.52)$$

Note that $f = FC'$ where C' is a constant, so the set $\{F < K_1 \text{ or } F > K_2\}$ is equal to the set $\{f\eta \text{ or } f > \xi\}$ with $\eta = C'K_1$ and $\xi = C'K_2$. Under H_0 , the distribution of F is Fisher with parameters $(n_1 - 1, n_2 - 1)$ (Theorem C.5), so we have a Fisher test. The bounds η and ξ are classically computed by the conditions $F_{n_1-1, n_2-1}(\eta) = \alpha/2$, $F_{n_1-1, n_2-1}(\xi) = 1 - \alpha/2$.

Last, note that by the properties of the Fisher distribution, the particular choice of η and ξ above is such that $\xi = 1/\eta$, so the rejection region is also defined by $\{f > \xi \text{ or } f < 1/\xi\}$, which is the same as $\{\max(f, 1/f) > \xi\}$, a form suitable to define a p -value (Section 4.1.3). Let $g = \max(f, 1/f)$ and $X \sim F_{m,n}$, then

$$p \stackrel{\text{def}}{=} P(\max(X, 1/X) > g) = P(X < 1/g) + P(X > g) = F_{n_1-1, n_2-1}(1/g) + 1 - F_{n_1-1, n_2-1}(g)$$

which, together with $1/g = \min(f, 1/f)$, shows Eq.(4.24).

4.7 REVIEW

4.7.1 TESTS ARE JUST TESTS

1. The first test to do on any data is a visual exploration. In most cases, this is sufficient.
2. Testing for a 0 mean or 0 median is the same as computing a confidence interval for the mean or the median.
3. Tests work only if the underlying assumptions are verified (in particular, practically all tests, even robust ones, assume the data comes from an iid sample).
4. Some tests work under a larger spectrum of assumptions (for example: even if the data is not normal). They are called robust tests. They should be preferred whenever possible.
5. Test whether the same variance assumption holds, otherwise, use robust tests or asymptotic results.
6. If you perform a large number of different tests on the same data, then the probability of rejecting H_0 is larger than for any single test. So, contrary to non-statistical tests, increasing the number of tests does not always improve the decision.

4.7.2 REVIEW QUESTIONS

QUESTION 4.7.1. *What is the critical region of a test ?*⁸

QUESTION 4.7.2. *What is a type 1 error ? A type 2 error ? The size of a test ?*⁹

QUESTION 4.7.3. *What is the p-value of a test ?*¹⁰

QUESTION 4.7.4. *What are the hypotheses for ANOVA ?*¹¹

QUESTION 4.7.5. *How do you compute a p-value by Monte Carlo simulation ?*¹²

QUESTION 4.7.6. *A Monte Carlo simulation returns $\hat{p} = 0$ as estimate of the p-value. Can we reject H_0 ?*¹³

QUESTION 4.7.7. *What is a likelihood ratio statistic test in a nest model ? What can we say in general about its p-value ?*¹⁴

⁸Call \vec{x} the data used for the test. The critical region C is a set of possible values of \vec{x} such that when $\vec{x} \in C$ we reject H_0 .

⁹A type 1 error occurs when the test says “do not accept H_0 ” whereas the truth is H_0 . A type 2 error occurs when the test says “accept H_0 ” whereas the truth is H_1 . The size of a test is $\sup_{\theta} \text{Pr}_{\theta}(C)$ (= the worst case probability of a type 1 error).

¹⁰It applies to tests where the critical region is of the form $T(\vec{x}) > m$ where $T(\vec{x})$ is the test statistic and \vec{x} is the data. The p-value is the probability that $T(\vec{X}) > T(\vec{x})$ where \vec{X} is a hypothetical data set, generated under the hypothesis H_0 . We reject H_0 at size α if $p < \alpha$.

¹¹The data is iid, gaussian, with perhaps different means but with same variance.

¹²Generate R iid samples T^r from the distribution of $T(\vec{X})$ under H_0 and compute \hat{p} as the fraction of times that $T^r > T(\vec{x})$. We need R large enough (typically order of 10000) and compute a confidence interval for \hat{p} using Theorem 2.4.

¹³We need to know the number R of Monte Carlo replicates. A confidence interval for p is $[0; 3.869/R]$ at level 95%; if R is order of 100 or more, we can reject H_0 at size 0.05.

¹⁴The test statistic is lrs , the log of the likelihood ratios under H_1 and H_0 , and the test rejects H_0 if lrs is large. The nested model means that the model is parametric, with some sets $\Theta_0 \subset \Theta$ such that H_0 means $\theta \in \Theta_0$ and H_1 means $\theta \in \Theta \setminus \Theta_0$. If the data sample is large, the p-value is obtained by saying that, under H_0 , $2lrs \sim \chi_{q_2}$, where q_2 is the number of degrees of freedom that H_1 adds to H_0 .

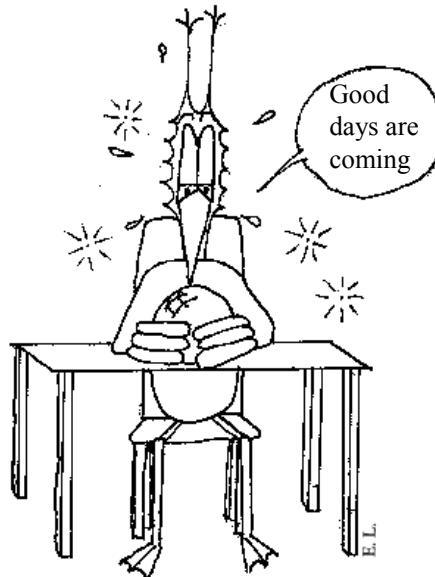
CHAPTER 5

FORECASTING

Forecasting is a risky exercise, and involves many aspects that are well beyond the scope of this book. However, it is the engineer's responsibility to **forecast what can be forecast**. For example, if the demand on a communication line is multiplied by 2 every 6 months, it is wise to provision enough capacity to accommodate this exponential growth. We present a simple framework to understand **what** forecasting is. We emphasize the need to quantify the accuracy of a forecast with a prediction interval.

For the **how**, there are many methods (perhaps because an exact forecast is essentially impossible). We focus on simple, generic methods that were found to work well in a large variety of cases. A first method is linear regression; it is very simple to use (with a computer) and of quite general application. It gives forecasts that are good as long as the data does not vary too wildly.

Better predictions may be obtained by a combination of differencing, de-seasonalizing filters and linear time series models (ARMA and ARIMA processes - this is also called the Box-Jenkins method). We discuss how to avoid model overfitting. We show that accounting for growth and seasonal effects is very simple and may be very effective. We also study five sparse ARMA and ARIMA models, known under other names such as EWMA and Holt Winters; they are numerically very simple and have no overfitting problem. The necessary background on digital filters can be found in Appendix D.



Contents

5.1 What is Forecasting ?	128
-------------------------------------	-----

5.2 Linear Regression	129
5.3 The Overfitting Problem	131
5.3.1 Use of Test Data	133
5.3.2 Information Criterion	133
5.4 Differencing the Data	136
5.4.1 Differencing and De-seasonalizing Filters	136
5.4.2 Computing Point Prediction	137
5.4.3 Computing Prediction Intervals	139
5.5 Fitting Differenced Data to an ARMA Model	140
5.5.1 Stationary but non IID Differenced Data	140
5.5.2 ARMA and ARIMA Processes	141
5.5.3 Fitting an ARMA Model	143
5.5.4 Forecasting	147
5.6 Sparse ARMA and ARIMA Models	150
5.6.1 Constrained ARMA Models	151
5.6.2 Holt-Winters Models	152
5.7 Proofs	156
5.8 Review Questions	158

5.1 WHAT IS FORECASTING ?

A classical forecasting example is capacity planning, where a communication or data center manager needs to decide when to buy additional capacity. Other examples concern optimal use of resources: if a data center is able to predict that some customers send less traffic at nights, this may be used to save power or to resell some capacity to customers in other time zones.

As in any performance related activity, it is important to follow a clean methodology, in particular, define appropriate metrics relevant to the problem area, define measurement methods, and gather time series of data. The techniques seen in this chapter start from this point, i.e. we assume that we have gathered some past measurement data, and would like to establish a forecast.

Informally, one can say that a forecast consists in extracting all information about the future that is already present in the past. Mathematically, this can be done as follows. To avoid complex mathematical constructions, we assume time is discrete. We are interested in some quantity $Y(t)$, where $t = 1, 2, \dots$. We assume that there is *some* randomness in $Y(t)$, so it is modeled as a stochastic process. Assume that we have observed Y_1, \dots, Y_t and would like to say something about $Y_{t+\ell}$ for some $\ell > 0$.

Forecasting can be viewed as computing the conditional distribution of $Y_{t+\ell}$, given Y_1, \dots, Y_t .

In particular, the *point prediction* or *predicted value* is

$$\hat{Y}_t(\ell) = \mathbb{E}(Y_{t+\ell} | Y_1 = y_1, \dots, Y_t = y_t)$$

and a *prediction interval* at level $1 - \alpha$ is an interval $[A, B]$ such that

$$\mathbb{P}(A \leq Y_{t+\ell} \leq B | Y_1 = y_1, \dots, Y_t = y_t) = 1 - \alpha$$

The forecasting problem thus becomes (1) to find and fit a good model and (2) to compute conditional distributions.

5.2 LINEAR REGRESSION

A simple and frequently used method is linear regression. It gives simple forecasting formulas, which are often sufficient. Linear regression models are defined in Chapter 3. In the context of forecasting, a linear regression model takes the form

$$Y_t = \sum_{j=1}^p \beta_j f_j(t) + \epsilon_t \quad (5.1)$$

where $f_j(t)$ are known, non random functions and ϵ_t is iid N_{0,σ^2} . Recall that the model is linear with respect to $\vec{\beta}$, whereas the functions f_j need not be linear with respect to t .

EXAMPLE 5.1: INTERNET TRAFFIC. Figure 5.1 shows a prediction of the total amount of traffic on a coast to coast link of an American internet service provider. The traffic is periodic with period 16 (one time unit is 90 mn), therefore we fit a simple sine function, i.e. we use a linear regression model with $p = 3$, $f_0(t) = 1$, $f_2(t) = \cos(\frac{\pi}{8}t)$ and $f_3(t) = \sin(\frac{\pi}{8}t)$. Using techniques in Section 3.2 we fit the parameters to the past data and obtain:

$$\begin{aligned} Y_t &= \sum_{j=1}^3 \beta_j f_j(t) + \epsilon_t \\ &= 238.2475 - 87.1876 \cos\left(\frac{\pi}{8}t\right) - 4.2961 \sin\left(\frac{\pi}{8}t\right) + \epsilon_t \end{aligned}$$

with $\epsilon_t \sim \text{iid } N_{0,\sigma^2}$ and $\sigma = 38.2667$. A point prediction is:

$$\hat{Y}_t(\ell) = \sum_{j=1}^3 \beta_j f_j(t + \ell) = 238.2475 - 87.1876 \cos\left(\frac{\pi}{8}(t + \ell)\right) - 4.2961 \sin\left(\frac{\pi}{8}(t + \ell)\right) \quad (5.2)$$

and a 95%-prediction interval can be approximated by $\hat{Y}_t(\ell) \pm 1.96\sigma$.

The computations in Example 5.1 are based on the following theorem and the formula after it; they result from the general theory of linear regression in Chapter 3 [32, Section 8.3]:

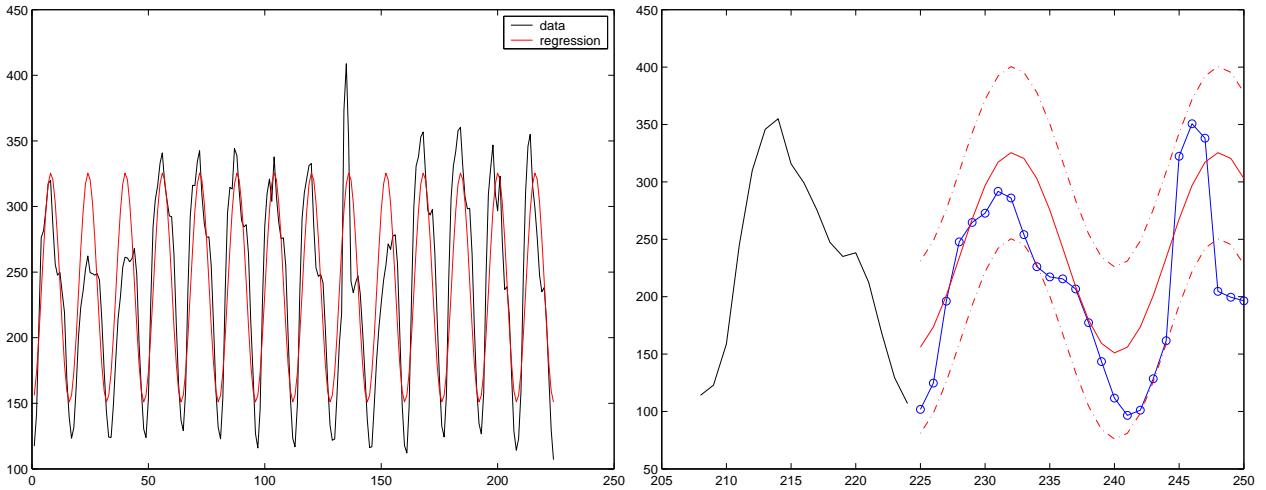


Figure 5.1: Internet traffic on a coast-to-coast link of an American internet service provider. One data point every 90 mn; y -axis is amount of traffic in Mb/s, averaged over 90 mn. Left: data for $t = 1$ to 224 and a sine function fitted to the data; right: zoom on the time interval from 205 to 250, showing the point prediction for the interval 225 to 250, the prediction interval and the true value (circles), not known when the prediction was done.

THEOREM 5.1. Consider a linear regression model as in Eq.(5.1) with p degrees of freedom for $\vec{\beta}$. Assume that we have observed the data at n time points t_1, \dots, t_n , and that we fit the model to these n observations using Theorem 3.3. Assume that the model is regular, i.e. the matrix X defined by $X_{i,j} = f_j(t_i)$, $i = 1, \dots, n$, $j = 1, \dots, p$ has full rank. Let $\hat{\beta}_j$ be the estimator of β_j and s^2 the estimator of the variance, as in Theorem 3.3.

1. The point prediction at time $t_n + \ell$ is $\hat{Y}_{t_n}(\ell) = \sum_{j=1}^p \hat{\beta}_j f_j(t_n + \ell)$
2. An exact prediction interval at level $1 - \alpha$ is

$$\hat{Y}_{t_n}(\ell) \pm \xi \sqrt{1 + g} s \quad (5.3)$$

with

$$g = \sum_{j=1}^p \sum_{k=1}^p f_j(t_n + \ell) G_{j,k} f_k(t_n + \ell)$$

where $G = (X^T X)^{-1}$ and ξ is the $(1 - \frac{\alpha}{2})$ quantile of the student distribution with $n - p$ degrees of freedom, or, for large n , of the standard normal distribution.

3. An approximate prediction interval that ignores estimation uncertainty is

$$\hat{Y}_{t_n}(\ell) \pm \eta s \quad (5.4)$$

where η is the $1 - \alpha$ quantile of the standard normal distribution.

We now explain the difference between the last two items in the theorem. Item 2 gives an exact result for a prediction interval. It captures two effects: (1) the **estimation error**, i.e. the uncertainty about the model parameters due to the estimation procedure (term g in $\sqrt{1 + g}$) and (2) the **model forecast uncertainty**, due to the model being a random process. In practice, we often expect the

estimation error to be much smaller than the model forecast uncertainty, i.e. g is much smaller than 1. This occurs in the rule when the number n of points used for the estimation is large, so we can also replace student by standard normal. This explains Eq.(5.4).

Figure 5.2 shows the prediction intervals computed by Eq.(5.3) and Eq.(5.4) (they are indistinguishable). By Theorem 3.3, one can also see that that a confidence interval for the point prediction is given by $\pm \xi \sqrt{g} s$ (versus $\pm \xi \sqrt{1+g} s$ for the prediction interval). The figure shows that the confidence interval for the point prediction is small but not negligible. However, its effect on the prediction interval is negligible. See also Figure 5.4 for what may happen when the problem is ill posed.

In the simple case where the data is assumed to be iid, we can see from Theorem 2.6 that g decreases like $\frac{1}{n}$, so in this case the approximation in Eq.(5.4) is always valid for large n .

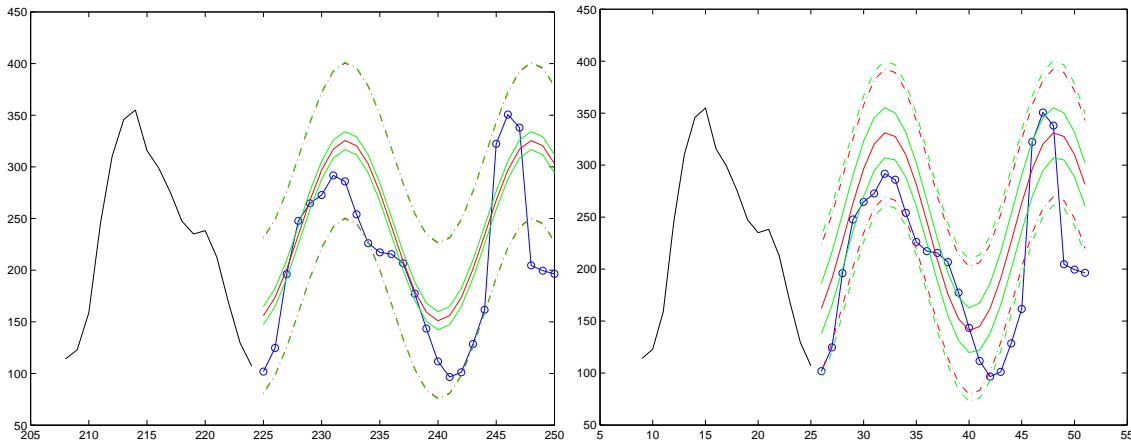


Figure 5.2: Left: Same example as Figure 5.1, showing the prediction interval computed by Theorem 5.1(dot-dashed lines) and the confidence interval for the point prediction (plain lines around center values). The predictions intervals computed by Eq.(5.3) and Eq.(5.4) are indistinguishable. Right: same except only the last 24 points of the past data are used to fitting the model (instead of 224). The confidence interval for the point prediction is slightly larger than in the left panel; the exact prediction interval computed from Theorem 5.1 is only slightly larger than the approximate one computed from Eq.(5.4).

VERIFICATION We cannot verify a prediction until the future comes. However, one can verify how well the model fits by screening the residuals, as explained in Theorem 3.3. The standardized residuals should look grossly normal, not showing large trends nor correlations. Figure 5.3 displays the standardized residuals for the model in Example 5.1. While the residuals fit well with the normal assumption, they do appear to have some correlation and some periodic behaviour. Models that are able to better capture these effects are discussed in Section 5.5.

5.3 THE OVERFITTING PROBLEM

Perhaps contrary to intuition, a parametric model should not have too many parameters. To see why, consider the model in Figure 5.1. Instead of a simple sine function, we now fit a more general model, where we add a polynomial component and a more general periodic function (with

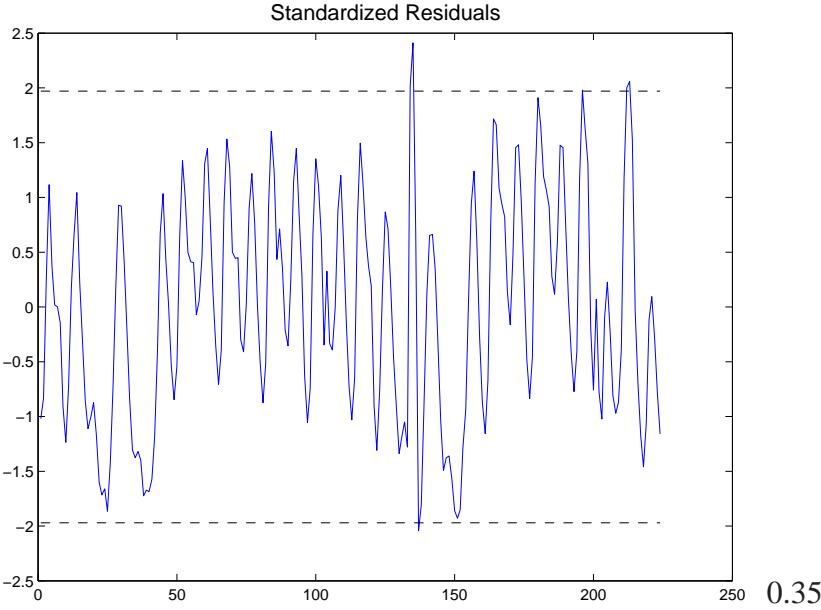


Figure 5.3: Residuals for the model fitted in Figure 5.1.

harmonics), with the hope of improving the fit, thus the prediction. The new model has the form

$$Y_t = \sum_{i=0}^d a_i t^i + \sum_{j=1}^h \left(b_j \cos \frac{j\pi t}{8} + c_j \sin \frac{j\pi t}{8} \right) \quad (5.5)$$

Figure 5.4 shows the resulting fit for a polynomial of degree $d = 10$ and with $h - 1 = 2$ harmonics. The fit is better ($\sigma = 25.4375$ instead of 38.2667), however, the prediction power is ridiculous. This

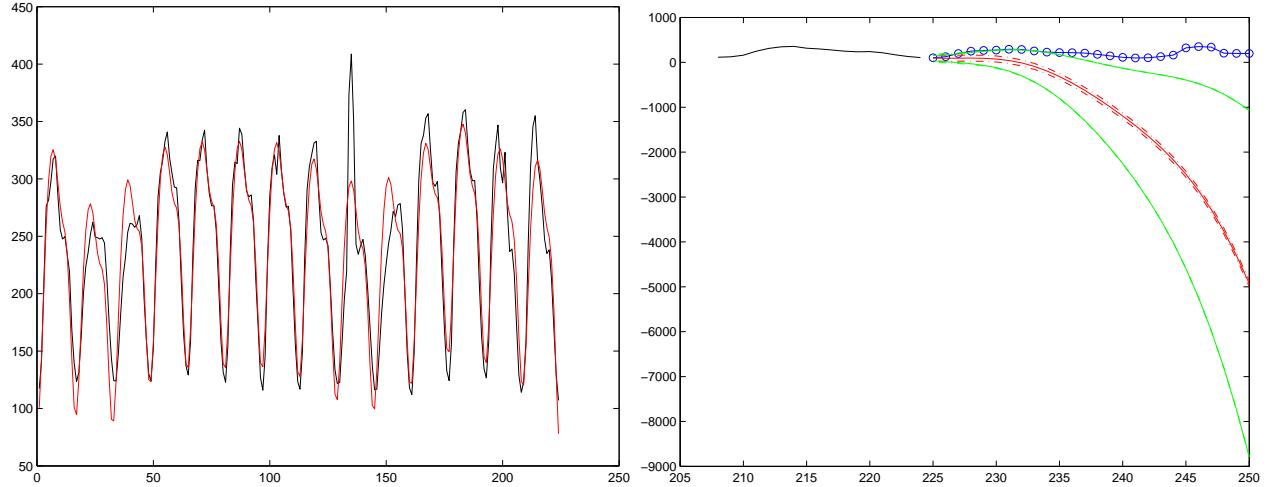


Figure 5.4: More parameters is not always better. Same as Figure 5.1, but with a more general model. Right panel: prediction intervals computed with the simple formula (5.4) (dot-dashed lines) do not coincide with the exact prediction intervals (plain lines). The line with small circles is the exact values.

is the **overfitting problem**. At the extreme, a model with absolute best fit has 0 residual error – but it is no longer an explanatory model.

There are two classical solutions to avoid overfitting: test data and information criterion.

5.3.1 USE OF TEST DATA

The idea is to reserve a small fraction of the data set to test the model prediction. Consider for example Figure 5.5. We fitted the model in Eq.(5.5) with $h - 1 = 2$ harmonics and a polynomial of degree $d = 0$ to 10. The prediction error is defined here as the mean square error between the true values of the data at $t=225$ to 250 and the point predictions given by Theorem 5.1. The estimation error is the estimator s of σ . The smallest prediction error is for $d = 4$. The fitting error decreases with d , whereas the prediction error is minimal for $d = 4$. This method is quite general but has the

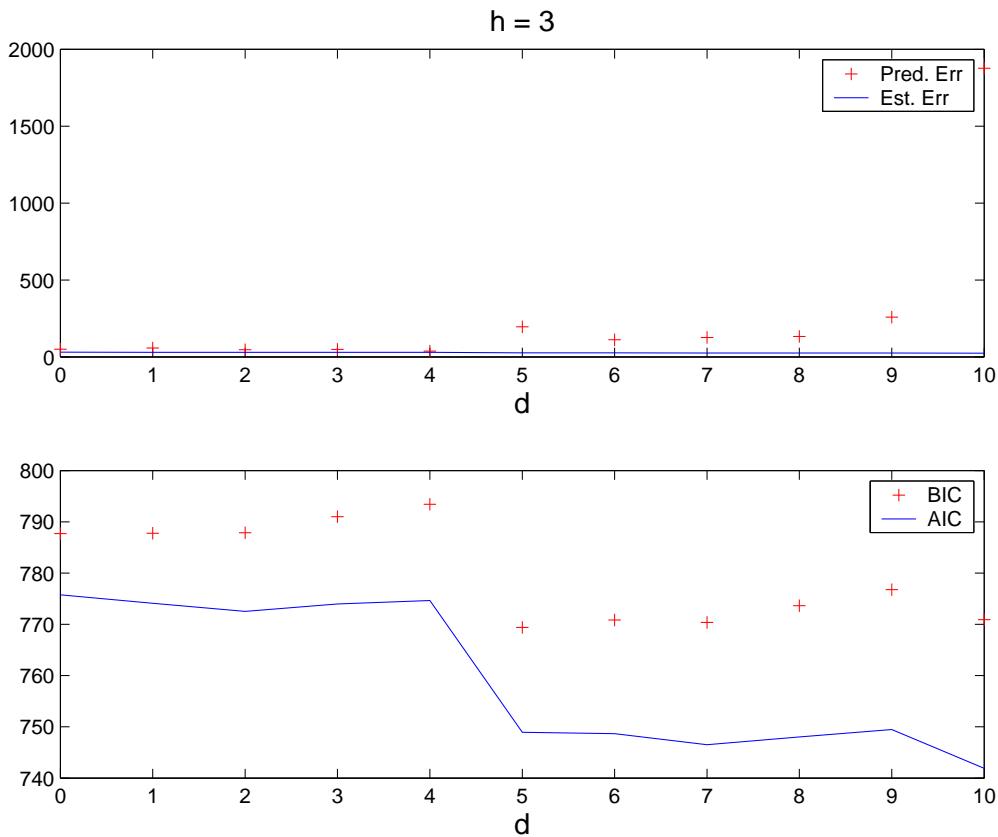


Figure 5.5: Model in Eq.(5.5) with $h - 1 = 2$ harmonics and a polynomial of degree $d = 0$ to 10. Top Panel: Use of test data: estimation and prediction errors. Bottom panel: information criteria. The test data finds that the best model is for $d = 4$, but the information criteria find that the best model is for $d = 10$, which is an aberrant model. Information criteria should be used only for models that match the type of data.

drawback to “burn” some of the data, as the test data cannot be used for fitting the model.

5.3.2 INFORMATION CRITERION

An alternative is to use an *information criterion*, which strikes a balance between model accuracy and number of parameters. *Akaike's Information Criterion* (AIC) is defined for any parametric

model by

$$\text{AIC} = -2l(\hat{\theta}) + 2k \quad (5.6)$$

where k is the dimension of the parameter θ and $l(\hat{\theta})$ is the estimated log-likelihood. It can be interpreted in an information theoretic sense as follows [105, Section 7.3]. Consider an independent replication X_t of the sequence Y_t ; then AIC is an approximate estimate of the number of bits needed by an optimal code to describe the sequence X_t , when the optimal code estimates the distribution of X_t from the sample Y_t . AIC thus measures the efficiency of our model to describe the data. The preferred model is the one with the *smallest* information criterion.

For the linear regression model with n data points and p degrees of freedom for $\vec{\beta}$, the parameter is $\theta = (\vec{\beta}, \sigma)$, thus $k = p + 1$. AIC can easily be computed and one obtains

$$\text{AIC} = 2(p + n \ln \hat{\sigma}) + C \quad (5.7)$$

where $C = 2 + n(1 + \ln(2\pi))$ and $\hat{\sigma}$ is the MLE of σ , i.e.

$$\hat{\sigma}^2 = \left(1 - \frac{p}{n}\right) s^2$$

In practice, the AIC has a tendency to overestimate the model order k when n , the number of observations, is small. An alternative criterion is the *Bayesian Information Criterion* (BIC) [19, 97], which is defined for a linear regression model by

$$\text{BIC} = -2l(\hat{\theta}) + k \ln n$$

Thus one finds

$$\text{BIC} = p \ln n + 2n \ln \hat{\sigma} + C' \quad (5.8)$$

with $C' = n(1 + \ln(2\pi)) + \ln n$ and p is the number of degrees of freedom for the parameter of the linear regression model.

EXAMPLE 5.2: INTERNET TRAFFIC, CONTINUED. We want to find the best fit for the model in Eq.(5.5). It seems little appropriate to fit the growth in Figure 5.1 by a polynomial of high degree, therefore we limit d to be 0, 1 or 2. We used three methods: test data, AIC and BIC and searched for all values of $d \in \{0, 1, 2\}$ and $h \in \{0, \dots, 10\}$. The results are :

```
Test Data: d=2, h=2, prediction error = 44.6006
Best AIC : d=2, h=3, prediction error = 46.1003
Best BIC : d=0, h=2, prediction error = 48.7169
```

The test data method finds the smallest prediction error, by definition. All methods find a small number of harmonics, but there are some minor differences. Figure 5.6 shows the values for $d=1$.

Figure 5.5 shows a different result. Here, we try to use a polynomial of degree up to 10, which is not appropriate for the data. The AIC and BIC find aberrant models, whereas test data finds a reasonable best choice.

Information criterion are more efficient in the sense that they do not burn any of the data; however, they may be completely wrong if the model is inappropriate.

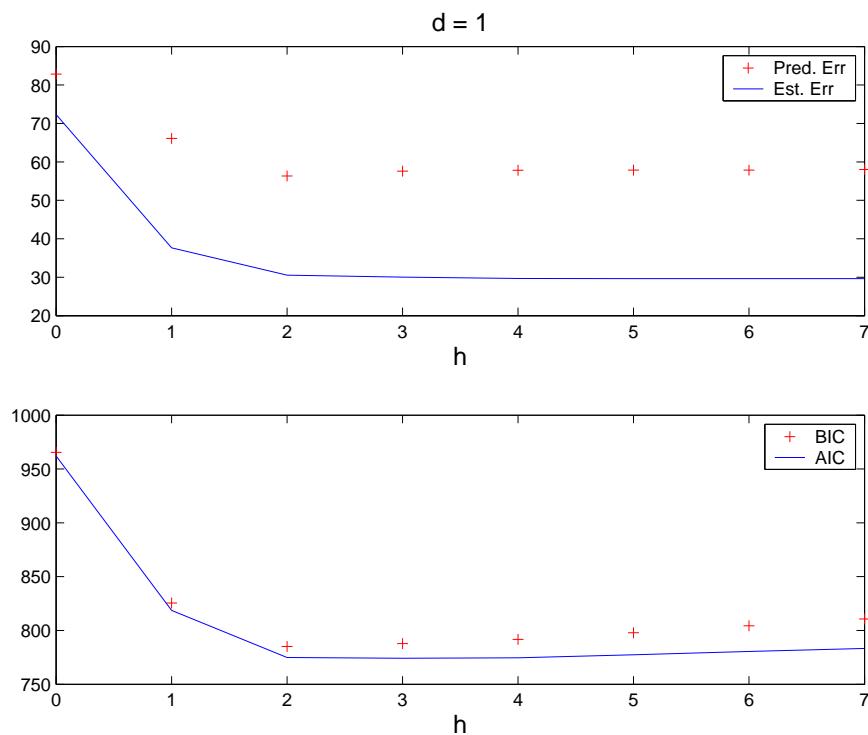


Figure 5.6: Choice of best Model for Eq.(5.5) with degree $d = 1$ and various values of h . Top panel: Use of test data; estimation and prediction errors. Bottom panel: information criteria. The prediction error is about the same for $h \geq 2$, which implies that the most adequate model if for $h = 2$. The information criteria also find here that the best model is for $h = 2$.

5.4 DIFFERENCING THE DATA

A slightly more sophisticated alternative to the regression method is to combine two approaches: first capture trends and periodic behaviour by application of differencing or de-seasonalizing filters, then fit the filtered data to a time series stationary model that allows correlation, as we explain in this and the next section.

5.4.1 DIFFERENCING AND DE-SEASONALIZING FILTERS

Consider a time series $Y = (Y_1, \dots, Y_n)$. Contrary to linear regression modelling, we require here that the indices are contiguous integers, $t = 1, \dots, n$. The *differencing filter* at lag 1 is the mapping, denoted with Δ_1 that transforms a times series Y of finite length into a time series $X = \Delta_1 Y$ of **same length** such that

$$X_t = (\Delta_1 Y)_t = Y_t - Y_{t-1} \quad t = 1, \dots, n \quad (5.9)$$

where by convention $Y_j = 0$ for $j \leq 0$. Note that this convention is not the best possible, but it simplifies the theory a lot. In practice, the implication is that the first term of the filtered series is not meaningful and should not be used for fitting a model (they are removed from the plots on Figure 5.7). Formally, we consider Δ_1 to be a mapping from $\bigcup_{n=1}^{\infty} \mathbb{R}^n$ onto itself, i.e. it acts on time series of any finite length.

The differencing filter Δ_1 is a discrete time equivalent of a derivative. If the data has a polynomial trend of degree $d \geq 1$, then $\Delta_1 Y$ has a trend of degree $d - 1$. Thus d iterated applications of Δ_1 to the data remove any polynomial trend of degree up to d .

Similarly, if the data Y is periodic with period s , then we can use the *de-seasonalizing* filter R_s (proposed by S.A. Roberts in [89]). It maps a times series Y of finite length into a time series $X = R_s Y$ of **same length** such that

$$X_t = \sum_{j=0}^{s-1} Y_{t-j} \quad t = 1, \dots, n \quad (5.10)$$

again with the convention that $Y_j = 0$ if $j \leq 0$. One application of R_s removes a periodic component, in the sense that if Y_t is periodic of period s , then $R_s Y$ is equal to a constant.

The differencing filter at lag s , Δ_s , is defined similarly by

$$(\Delta_s X)_t = Y_t - Y_{t-s} \quad (5.11)$$

It can be easily seen that

$$\Delta_s = R_s \Delta_1 \quad (5.12)$$

i.e. combining de-seasonalizing and differencing at lag 1 is the same as differencing at lag s .

Filters **commute**, e.g. $R_s' R_s Y = R_s R_s' Y$ for all s, s' and $Y \in \mathbb{R}^n$ (see Appendix D). It follows that the differencing filter and de-seasonalizing filter may be used to remove polynomial growth, non zero mean and periodicities, and that one can apply them in any order. In practice, one tries to apply R_s once for any identified period d , and Δ_1 as many times as required for the data to appear stationary.

EXAMPLE 5.3: INTERNET TRAFFIC. In Figure 5.7 we apply the differencing filter Δ_1 to the time series in Example 5.1 and obtain a strong seasonal component with period $s = 16$. We then apply

the de-seasonalizing filter R_{16} ; this is the same as applying Δ_{16} to the original data. The result does not appear to be stationary; an additional application of Δ_1 is thus performed.

Also note that if $Y_t = \mu + Z_t$ where Z_t is stationary, then $\Delta_s Y$ has a zero mean¹. Thus, if after enough differencing we have obtained a stationary but non zero mean sequence, one more differencing operation produces a zero mean sequence.

5.4.2 COMPUTING POINT PREDICTION

With many time series, differencing and de-seasonalizing produces a data set that has neither growth nor periodicity, thus is a good candidate for being fitted to a simple stochastic model. In this section we illustrate a straightforward application of this idea. The method used in this section will also be used in Section 5.5 with more elaborate models for the differenced data.

Assume we have a model for the differenced data X_t that we can use to obtain predictions for X_t . How can we use this information to derive a prediction for the original data Y_t ? There is a very simple solution, based on the properties of filters given in appendix.

We write compactly $X = LY$, i.e L is the combination of filters (possibly used several times each) used for differencing and de-seasonalizing. For example, in Figure 5.7, $L = \Delta_{16}\Delta_1$. Δ_s is an invertible filter for all $s \geq 1$ thus L also is an invertible filter (see Appendix D for more details). We can use the AR(∞) representation of L^{-1} and write, using Eq.(D.16) in appendix:

$$Y_t = X_t - g_1 Y_{t-1} - \dots - g_q Y_{t-q} \quad (5.13)$$

where $(g_0 = 1, g_1, \dots, g_q)$ is the impulse response of the filter L . See the next example and Appendix D for more details on how to obtain the impulse response of L . The following result derives immediately from this and Theorem D.1:

PROPOSITION 5.1. *Assume that $X = LY$ where L is a differencing or de-seasonalizing filter with impulse response $g_0 = 1, g_1, \dots, g_q$. Assume that we are able to produce a point prediction $\hat{X}_t(\ell)$ for $X_{t+\ell}$ given that we have observed X_1 to X_t . For example, if the differenced data can be assumed to be iid with mean μ , then $\hat{X}_t(\ell) = \mu$.*

A point prediction for $Y_{t+\ell}$ can be obtained iteratively by:

$$\begin{aligned} \hat{Y}_t(\ell) &= \hat{X}_t(\ell) - g_1 \hat{Y}_t(\ell-1) - \dots - g_{\ell-1} \hat{Y}_t(1) - g_\ell y_t - \dots \\ &\quad - g_q y_{t-q+\ell} \quad \text{for } 1 \leq \ell \leq q \end{aligned} \quad (5.14)$$

$$\hat{Y}_t(\ell) = \hat{X}_t(\ell) - g_1 \hat{Y}_t(\ell-1) - \dots - g_q \hat{Y}_t(\ell-q) \quad \text{for } \ell > q \quad (5.15)$$

In the proposition, we write y_t in lower-case to stress that, when we perform a prediction at time t , the data up to time t is considered known and non-random.

Note that by differencing enough times we are able to remove any non zero means from the data. Consequently, we often assume that $\mu = 0$.

¹more precisely $\mathbb{E}(\Delta_s Y_t) = 0$ for $t \geq s+1$. i.e. the first s elements of the differenced time series may not be 0 mean.

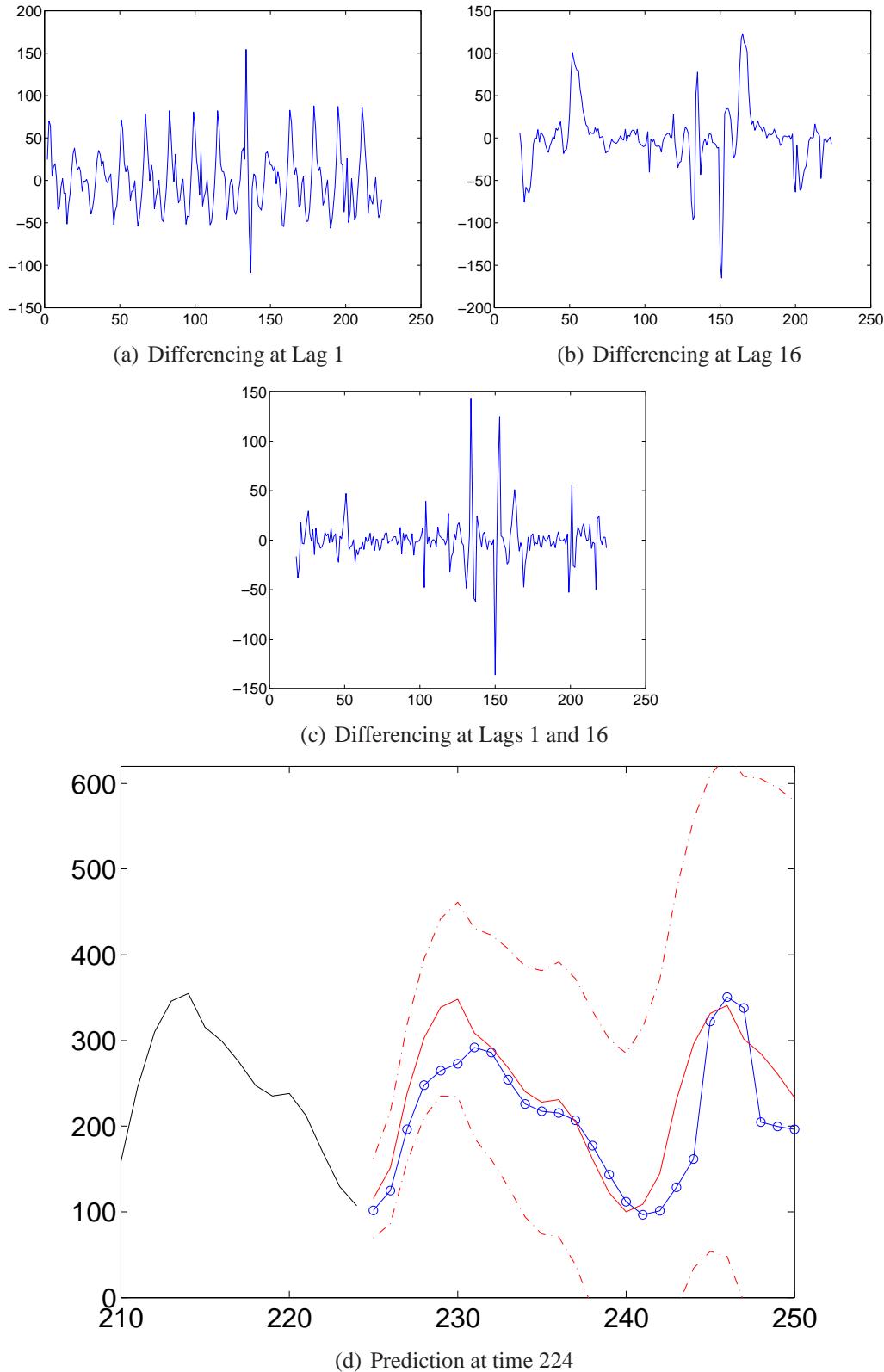


Figure 5.7: Differencing filters Δ_1 and Δ_{16} applied to Example 5.1 (first terms removed). The forecasts are made assuming the differenced data is iid gaussian with 0 mean. \circ = actual value of the future (not used for fitting the model). The point prediction is better than on Figure 5.1, but the prediction intervals are large.

EXAMPLE 5.4: INTERNET TRAFFIC, CONTINUED. For Figure 5.7, we have

$$L = \Delta_1^2 R_{16} = \Delta_1 \Delta_{16} = (1 - B)(1 - B^{16}) = 1 - B - B^{16} + B^{17}$$

thus the impulse response g of L is given by

$$g_0 = g_{17} = 1, \quad g_1 = g_{16} = -1, \quad g_m = 0 \text{ otherwise}$$

If we can assume that the differenced data is iid with 0 mean, the prediction formulae for Y are

$$\begin{aligned}\hat{Y}_t(1) &= y_t + y_{t-15} - y_{t-16} \\ \hat{Y}_t(\ell) &= \hat{Y}_t(\ell-1) + y_{t+\ell-16} - y_{t+\ell-17} \text{ for } 2 \leq \ell \leq 16 \\ \hat{Y}_t(17) &= \hat{Y}_t(16) + \hat{Y}_t(1) - y_t \\ \hat{Y}_t(\ell) &= \hat{Y}_t(\ell-1) + \hat{Y}_t(\ell-16) - \hat{Y}_t(\ell-17) \text{ for } \ell \geq 18\end{aligned}$$

5.4.3 COMPUTING PREDICTION INTERVALS

If we want to obtain not only point predictions but also to quantify the prediction uncertainty, we need to compute prediction intervals. We consider a special, but frequent case. More general cases can be handled by Monte Carlo methods as explained in Section 5.5.4. The following result derives from Theorem D.1 in appendix.

PROPOSITION 5.2. Assume that the differenced data is iid gaussian. i.e. $X_t = (LY)_t \sim \text{iid } N(\mu, \sigma^2)$. The conditional distribution of $Y_{t+\ell}$ given that $Y_1 = y_1, \dots, Y_t = y_t$ is gaussian with mean $\hat{Y}_t(\ell)$ obtained from Eq.(5.14) and variance

$$MSE_t^2(\ell) = \sigma^2 (h_0^2 + \dots + h_{\ell-1}^2) \quad (5.16)$$

where h_0, h_1, h_2, \dots is the impulse response of L^{-1} . A prediction interval at level 0.95 is thus

$$\hat{Y}_t(\ell) \pm 1.96 \sqrt{MSE_t^2(\ell)} \quad (5.17)$$

Alternatively, one can compute $\hat{Y}_t(\ell)$ using

$$\hat{Y}_t(\ell) = \mu (h_0 + \dots + h_{\ell-1}) + h_\ell x_t + \dots + h_{t+\ell-1} x_1 \quad (5.18)$$

The impulse response of L^{-1} can be obtained numerically (for example using the filter command), as explained in Appendix D. If L is not too complicate, it can be obtained in a simple closed form. For example, for $s = 1$, the reverse filter Δ_1^{-1} is defined by

$$(\Delta_1^{-1} X)_t = X_1 + X_2 + \dots + X_t \quad t = 1, \dots, n$$

i.e. its impulse response is $h_m = 1$ for all $m \geq 0$. It is a discrete time equivalent of integration.

The impulse response of $L = (\Delta_1 \Delta_s)^{-1}$ used in Figure 5.7 is

$$h_m = 1 + \left\lfloor \frac{m}{16} \right\rfloor \quad (5.19)$$

where the notation $\lfloor x \rfloor$ means the largest integer $\leq x$.

Note that μ and σ need to be estimated from the differenced data².

EXAMPLE 5.5: INTERNET TRAFFIC, CONTINUED. Figure 5.7 shows the prediction obtained assuming the differenced data is iid gaussian with 0 mean.

It is obtained by applying Eq.(5.18) with $\mu = 0$, Eq.(5.17) and Eq.(5.19).

The point prediction is good, but the confidence interval appear to be larger than necessary. Note that the model we used here is extremely simple; we only need to fit one parameter (namely σ), which is estimated as the sample standard deviation of the differenced data.

Compare to Figure 5.1: the point prediction seems to be more exact. Also, it starts just from the previous value. The point prediction with differencing filters is more adaptive than with a regression model.

The prediction intervals are large and grow with the prediction horizon. This is a symptom that the iid gaussian model for the differenced data may not be appropriate. In fact, there are two deviations from this model: the distribution does not appear to be gaussian, and the differenced appears to be correlated (large values are not isolated). Addressing these issues requires a more complex model to be fitted to the differenced time series: this is the topic of Section 5.5

5.5 FITTING DIFFERENCED DATA TO AN ARMA MODEL

The method in this section is inspired by the original method of Box and Jenkins in [15] and can be called the **Box-Jenkins** method, although some of the details differ a bit. It applies to cases where the differenced data X appears to be stationary but not iid. In essence, the method provides a method to **whiten** the differenced data, i.e. it computes a filter F such that FX can be assumed to be iid. We first discuss how to recognize whether data can be assumed to be iid.

5.5.1 STATIONARY BUT NON IID DIFFERENCED DATA

After pre-processing with differencing and de-seasonalizing filters we have obtained a data set that appears to be **stationary**. Recall from Chapter 6 that a stationary model is such that it is statistically impossible to recognize at which time a particular sample was taken. The time series in panel (c) of Figure 5.7 appear to have this property, whereas the original data set in panel (a) does not. In the context of time series, lack of stationarity is due to growth or periodicity: if a data set increases (or decreases), then by observing a sample we can have an idea of whether it is old or young; if there is a daily pattern, we can guess whether a sample is at night or at daytime.

SAMPLE ACF

A means to test whether a data series that appears to be stationary is iid or not is the **sample autocovariance** function; by analogy to the autocovariance of a process, it is defined, for $t \geq 0$

²Here too, the prediction interval does not account for the estimation uncertainty

by

$$\hat{\gamma}_t = \frac{1}{n} \sum_{s=1}^{n-t} (X_{s+t} - \bar{X})(X_s - \bar{X}) \quad (5.20)$$

where \bar{X} is the sample mean. The **sample ACF** is defined by $\hat{\rho}_t = \hat{\gamma}_t / \hat{\gamma}_0$. The sample PACF is also defined as an estimator of the true partial autocorrelation function (PACF) defined in Section 5.5.2.

If X_1, \dots, X_n is iid with finite variance, then the sample ACF and PACF are asymptotically centered normal with variance $1/n$. ACF and PACF plots usually display the bounds $\pm 1.96/\sqrt{n}$. If the sequence is iid with finite variance, then roughly 95% of the points should fall within the bounds. This provides a method to assess whether X_t is iid or not. If yes, then no further modelling is required, and we are back to the case in Section 5.4.2. See Figure 5.10 for an example.

The ACF can be tested formally by means of the **Ljung-Box** test. It tests H_0 : “the data is iid” versus H_1 : “the data is stationary”. The test statistic is $L = n(n+2) \sum_{s=1}^t \frac{\hat{\rho}_s^2}{n-s}$, where t is a parameter of the test (number of coefficients), typically \sqrt{n} . The distribution of L under H_0 is χ_t^2 , which can be used to compute the p -value.

5.5.2 ARMA AND ARIMA PROCESSES

Once a data set appears to be stationary, but not iid (as in panel (c) of Figure 5.7) we can model it with an **Auto-Regressive Moving Average** (ARMA) process.

DEFINITION 5.1. A 0-mean ARMA(p, q) process X_t is a process that satisfies for $t = 1, 2, \dots$ a difference equation such as:

$$X_t + A_1 X_{t-1} + \dots + A_p X_{t-p} = \epsilon_t + C_1 \epsilon_{t-1} + \dots + C_q \epsilon_{t-q} \quad \epsilon_t \text{ iid } \sim N_{0, \sigma^2} \quad (5.21)$$

Unless otherwise specified, we assume $X_{-p+1} = \dots = X_0 = 0$.

An ARMA(p, q) process with mean μ is a process X_t such that $X_t - \mu$ is a 0 mean ARMA process and, unless otherwise specified, $X_{-p+1} = \dots = X_0 = \mu$.

The parameters of the process are A_1, \dots, A_p (**auto-regressive coefficients**), C_1, \dots, C_q (**moving average coefficients**) and σ^2 (**white noise variance**). The iid sequence ϵ_t is called the noise sequence, or **innovation**.

An ARMA($p, 0$) process is also called an **Auto-regressive** process, AR(p); an ARMA($0, q$) process is also called a **Moving Average** process, MA(q).

Since a difference equation as in Eq.(5.21) defines a filter with rational transfer function (Appendix D), one can also define an ARMA process by

$$X = \mu + F\epsilon \quad (5.22)$$

where ϵ is an iid gaussian sequence and

$$F = \frac{1 + C_1 B + \dots + C_q B^q}{1 + A_1 B + \dots + A_p B^p} \quad (5.23)$$

B is the backshift operator, see Appendix D.

In order for an ARMA process to be practically useful, we need the following:

HYPOTHESIS 5.1. *The filter in Eq.(5.23) and its inverse are stable.*

In practice, this means that the zeroes of $1 + A_1 z^{-1} + \dots + A_p z^{-p}$ and of $1 + C_1 z^{-1} + \dots + C_q z^{-q}$ are within the unit disk.

Eq.(5.22) can be used to simulate ARMA processes, as in Figure 5.8.

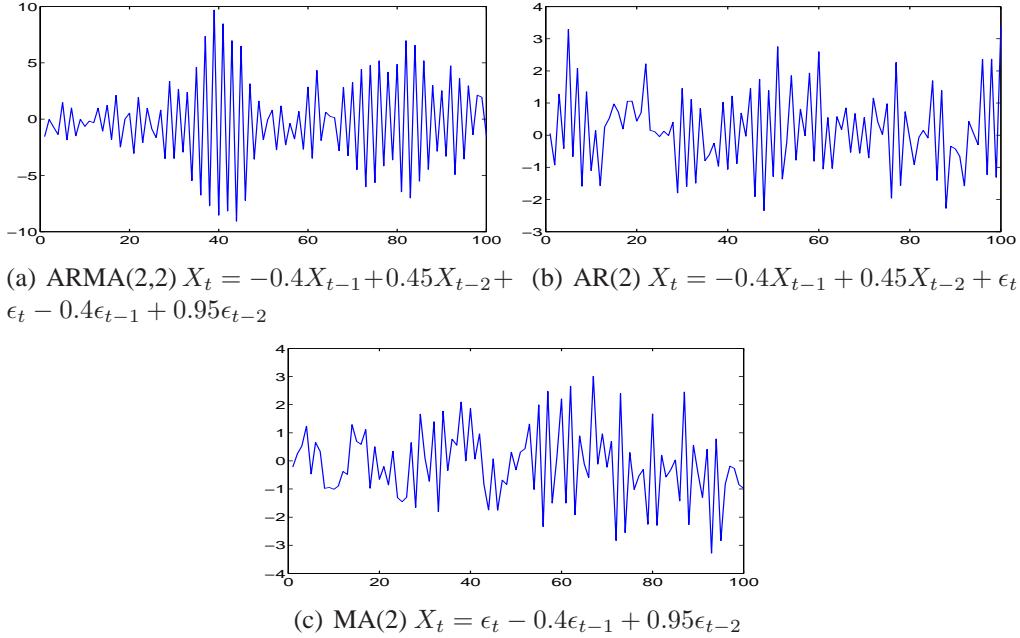


Figure 5.8: Simulated ARMA processes with 0 mean and noise variance $\sigma^2 = 1$. The first one, for example, is obtained by the matlab commands `Z=randn(1,n)` and `X=filter([1 -0.4 +0.95],[1 0.4 -0.45],Z)`.

ARMA PROCESS AS A GAUSSIAN PROCESS Since an ARMA process is defined by linear transformation of a gaussian process ϵ_t it is a gaussian process. Thus it is entirely defined by its mean $\mathbb{E}(X_t) = \mu$ and its covariance. Its covariance can be computed in a number of ways, the simplest is perhaps obtained by noticing that

$$X_t = \mu + h_0\epsilon_t + \dots + h_{t-1}\epsilon_1 \quad (5.24)$$

where h is the impulse response of the filter in Eq.(5.23). Note that, with our convention, $h_0 = 1$. It follows that for $t \geq 1$ and $s \geq 0$:

$$\text{cov}(X_t, X_{t+s}) = \sigma^2 \sum_{j=0}^{t-1} h_j h_{j+s} \quad (5.25)$$

For large t

$$\text{cov}(X_t, X_{t+s}) \approx \gamma_s = \sigma^2 \sum_{j=0}^{\infty} h_j h_{j+s} \quad (5.26)$$

The convergence of the latter series follows from the assumption that the filter is stable. Thus, for large t , the covariance does not depend on t . More formally, one can show that an ARMA

process with Hypothesis 5.1 is asymptotically stationary [19, 97], as required since we want to model stationary data³.

Note in particular that

$$\text{var}(X_t) \approx \sigma^2 \sum_{j=0}^{\infty} h_j^2 = \sigma^2 (1 + \sum_{j=1}^{\infty} h_j^2) \geq \sigma^2 -- \quad (5.27)$$

thus the variance of the ARMA process is larger than that of the noise⁴.

For an MA(q) process, we have $h_j = C_j$ for $j = 1, \dots, q$ and $h_j = 0$ for $j \geq q$ thus the ACF is 0 at lags $\geq q$.

The **Auto-Correlation Function** (ACF) is defined by⁵ $\rho_t = \gamma_t / \gamma_0$. The ACF quantifies departure from an iid model; indeed, for an iid sequence (i.e. $h_1 = h_2 = \dots = 0$), $\rho_t = 0$ for $t \geq 1$. The ACF can be computed from Eq.(5.26) but in practice there are more efficient methods that exploit Eq.(5.23), see [105], and which are implemented in standard packages. One also sometimes uses the **Partial Auto-Correlation Function** (PACF), which is defined in Section C.5.3 as the residual correlation of X_{t+s} and X_t , given that $X_{t+1}, \dots, X_{t+s-1}$ are known.⁶

Figure 5.9 shows the ACF and PACF of a few ARMA processes. They all decay exponentially. For an AR(p) process, the PACF is exactly 0 at lags⁷ $t > p$.

ARIMA PROCESS By definition, the random sequence $Y = (Y_1, Y_2, \dots)$ is an ARIMA(p, d, q) (Auto-Regressive Integrated Moving Average) process if differencing Y d times gives an ARMA(p, q) process (i.e. $X = \Delta_1^d Y$ is an ARMA process, where Δ_1 is the differencing filter at lag 1). For $d \geq 1$ an ARIMA process is not stationary.

In the statistics literature, it is customary to describe an ARIMA(p, d, q) process Y_t by writing

$$(1 - B)^d (1 + A_1 B + \dots + A_p B^p) Y = (1 + C_1 B + \dots + C_q B^q) \epsilon \quad (5.28)$$

which is the same as saying that $\Delta_1^d Y$ is a zero mean ARMA(p, q) process.

By extension, we also call ARIMA process a process Y_t such that LY is an ARMA process where L is a combination of differencing and de-seasonalizing filters.

5.5.3 FITTING AN ARMA MODEL

Assume we have a time series which, after differencing and de-seasonalizing (and possible rescaling) produces a time series X_t that appears to be stationary and close to gaussian (i.e does not have too wild dynamics), but not iid. We may now think of fitting an ARMA model to X_t .

The ACF and PACF plots may give some bound about the orders p and q of the model, as there tend to be exponential decay at lags larger than p and q .

³Furthermore, it can easily be shown that if the initial conditions X_0, \dots, X_{-p} are not set to 0 as we do for simplicity, but are drawn from the gaussian process with mean μ and covariance γ_s , then X_t is (exactly) stationary. We ignore this subtlety in this chapter and consider only asymptotically stationary processes.

⁴Equality occurs only when $h_1 = h_2 = \dots = 0$, i.e. for the trivial case where $X_t = \epsilon_t$

⁵Some authors call autocorrelation the quantity γ_t instead of ρ_t .

⁶The PACF is well defined if the covariance matrix of (X_t, \dots, X_{t+s}) is invertible. For an ARMA process, this is always true, by Corollary C.2.

⁷This follows from the definition of PACF and the fact that X_{t+s} is entirely determined by $X_{t+s-p}, \dots, X_{t+s-p}$.

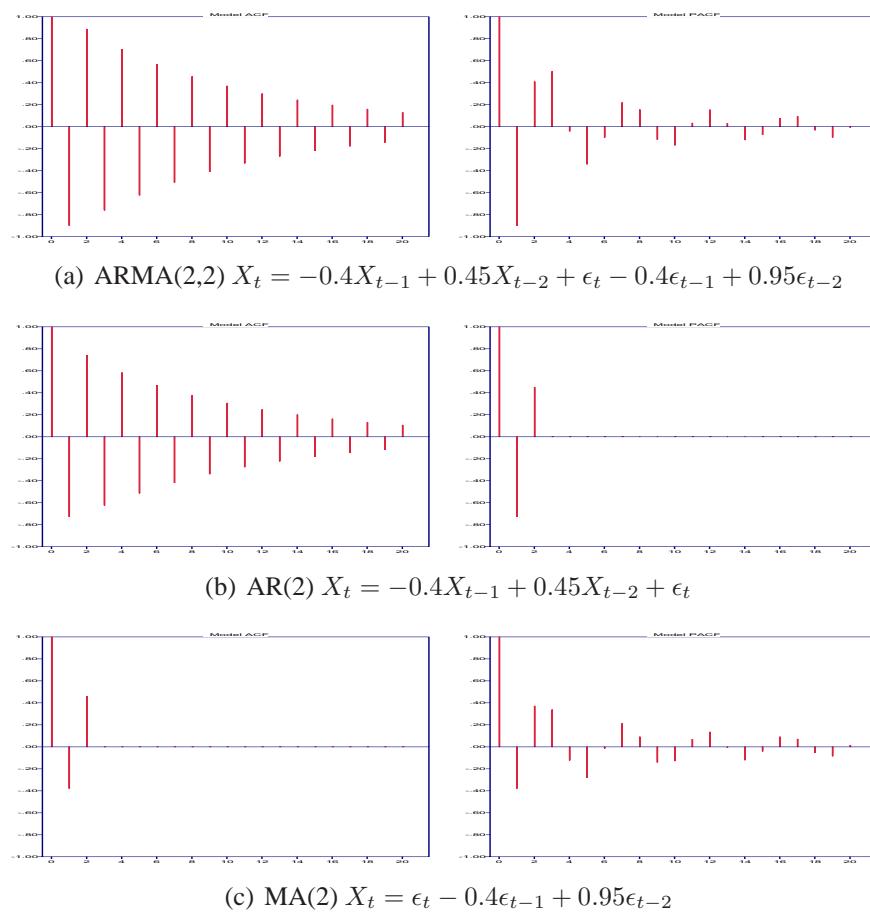


Figure 5.9: ACF (left) and PACF (right) of some ARMA processes.

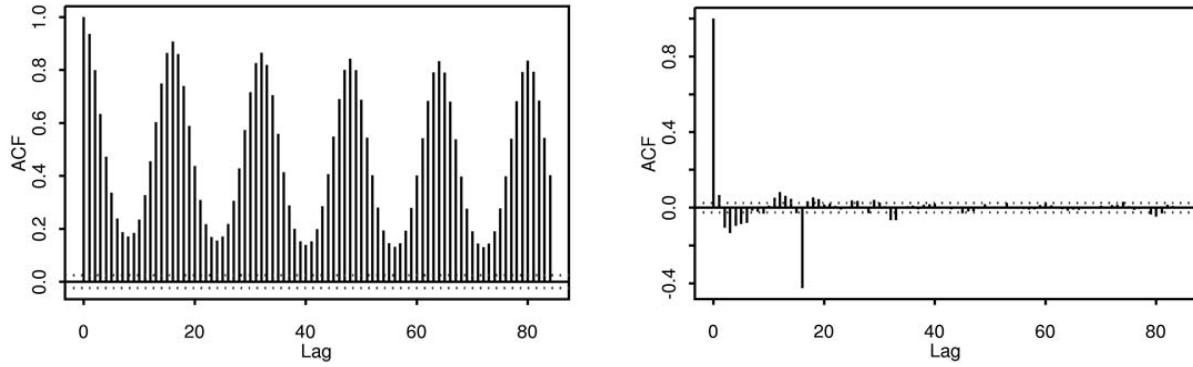


Figure 5.10: First panel: Sample ACF of the internet traffic of Figure 5.1. The data does not appear to come from a stationary process so the sample ACF cannot be interpreted as estimation of a true ACF (which does not exist). Second panel: sample ACF of data differenced at lags 1 and 16. The sampled data appears to be stationary and the sample ACF decays fast. The differenced data appears to be suitable for modelling by an ARMA process.

Note that the sample ACF and PACF make sense only if the data appears to be generated from a **stationary** process. If the data comes from a non stationary process, this may be grossly misleading (Figure 5.10).

MAXIMUM LIKELIHOOD ESTIMATION OF AN ARMA OR ARIMA MODEL

Once we have decided for orders p and q , we need to estimate the parameters $\mu, \sigma, A_1, \dots, A_p, C_1, \dots, C_q$. As usual, this is done by maximum likelihood. This is simplified by the following result.

THEOREM 5.2. *Consider an ARMA or ARIMA model with parameters as in Definition 5.1, where the parameters are constrained to be in some set \mathcal{S} . Assume we are given some observed data x_1, \dots, x_N .*

1. *The log likelihood of the data is $-\frac{N}{2} \ln(2\pi\hat{\sigma}^2)$ where*

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{t=2}^N \left(x_t - \hat{X}_{t-1}(1) \right)^2 \quad (5.29)$$

and $\hat{X}_{t-1}(1)$ is the one step ahead forecast at time $t-1$.

2. *Maximum likelihood estimation is equivalent to minimizing the mean square one step ahead forecast error $\hat{\sigma}$, subject to the model parameters being in \mathcal{S} .*

The one step forecasts $\hat{X}_{t-1}(1)$ are computed using Proposition 5.4 below. Care should be taken to remove the initial values if differencing is performed.

Contrary to linear regression, the optimization involved here is non linear, even if the constraints on the parameter set are linear. The optimizer usually requires some initial guess to run efficiently. For MA(q) or AR(p) there exist estimation procedures (called **moment heuristics**) that are not maximum likelihood but are numerically fast [105]. These are based on the observation that for MA(q) or AR(p) processes, if we know the autocovariance function exactly, then we can compute

the coefficients numerically⁸. Then we use the sample autocovariance as estimate of the autocovariance function, whence we deduce an estimate of the parameters of the process. This is less accurate than maximum likelihood, but is typically used as an initial guess. For example, if we want to compute the maximum likelihood estimate of a general ARMA(p, q) model, we may estimate the parameters $\mu, \sigma, C_1, \dots, C_q$ of an MA(q) model, using a moment fitting heuristic. We then give as initial guess these values plus $A_1 = \dots = A_p = 0$.

It is necessary to verify that the obtained ARMA model corresponds to a stable filter with stable inverse. Good software packages automatically do so, but at times, it may be impossible to obtain both a stable filter and a stable inverse. It is generally admitted that this may be fixed by changing the differencing filter: too little differencing may make it impossible to obtain a stable filter (as the differenced data is not stationary); conversely, too much differencing may make it impossible to obtain a stable inverse [19].

DETERMINATION OF BEST MODEL ORDER

Deciding for the correct order may be done with the help of an information criterion (Section 5.3.2), such as the AIC. For example, assume we would like to fit the differenced data X_t to a general ARMA(p, q) model, without any constraint on the parameters; we have $p + q$ coefficients, plus the mean μ and the variance σ^2 ; thus, up to the constant $-N \ln(2\pi)$, which can be ignored, we have

$$\text{AIC} = -N \ln \hat{\sigma}^2 + 2(p + q + 2) \quad (5.30)$$

Note that the AIC counts as degrees of freedom only continuous parameters, so it does not count the number of times we applied differencing or de-seasonalizing to the original data. Among all the possible values of p, q and possibly among several application of differencing or de-seasonalizing filters, we choose the one than minimizes AIC.

VERIFICATION OF RESIDUALS

The sequence of residuals $e = (e_1, e_2, \dots)$ is an estimation of the non observed innovation sequence ϵ . It is obtained by

$$(e_1, e_2, \dots, e_t) = F^{-1}(x_1 - \mu, x_2 - \mu, \dots, x_t - \mu) \quad (5.31)$$

where (x_1, x_2, \dots) is the differenced data and F is the ARMA filter in Eq.(5.23). If the model fit is good, the residuals should be roughly independent, therefore the ACF and PACF of the residuals should be close to 0 at all lags.

Note that the residuals can also be obtained from the following proposition (the proof of which easily follows from Corollary D.2, applied to X_t and ϵ_t instead of Y_t and X_t)

PROPOSITION 5.3 (Innovation Formula).

$$\epsilon_t = X_t - \hat{X}_{t-1}(1) \quad (5.32)$$

where $\hat{X}_{t-1}(1)$ is the one step ahead prediction at time $t - 1$.

⁸For AR(p) processes, the AR coefficients are obtained by solving the “Yule-Walker” equations, using the “Levinson-Durbin” algorithm [105]

Thus, to estimate the residuals, one can compute the one step ahead predictions for the available data $\hat{x}_{t-1}(1)$, using the forecasting formulae given next; the residuals are then

$$e_t = x_t - \hat{x}_{t-1}(1) \quad (5.33)$$

5.5.4 FORECASTING

Once a model is fitted to the differenced data, forecasting derive easily from Theorem D.1, given in appendix, and its corollaries. Essentially, Theorem D.1 says that predictions for X and Y are obtained by mapping predictions for ϵ by means of the reverse filters. Since ϵ is iid, predictions for ϵ are trivial: e.g. the point prediction $\hat{\epsilon}_t(h)$ is equal to the mean. One needs to be careful, though, since the first terms of the differenced time series X_t are not known, and one should use recursive formulas that avoid propagation of errors. This gives the following formulas:

PROPOSITION 5.4. *Assume the differenced data $X = LY$ is fitted to an ARMA(p, q) model with mean μ as in Definition 5.1.*

1. *The ℓ -step ahead predictions at time t , $\hat{X}_t(\ell)$, of the differenced data can be obtained for $t \geq 1$ from the recursion*

$$\begin{aligned} \hat{X}_t(\ell) - \mu &+ A_1(\hat{X}_t(\ell-1) - \mu) + \dots + A_p(\hat{X}_t(\ell-p) - \mu) = C_1\hat{\epsilon}_t(\ell-1) + \dots + C_q\hat{\epsilon}_t(\ell-q) \\ \hat{X}_t(\ell) &= \begin{cases} X_{t+\ell} & \text{if } \ell \leq 0 \text{ and } 1 \leq t + \ell \\ \mu & \text{if } t + \ell \leq 0 \end{cases} \\ \hat{\epsilon}_t(\ell) &= \begin{cases} 0 & \text{if } \ell \geq 1 \text{ or } t + \ell \leq 0 \\ X_{t+\ell} - \hat{X}_{t+\ell-1}(1) & \text{if } \ell \leq 0 \text{ and } t + \ell \geq 2 \\ X_1 - \mu & \text{if } t + \ell = 1 \text{ and } \ell \leq 0 \end{cases} \end{aligned}$$

In the recursion, we allow $\ell \leq 0$ even though we are eventually interested only in $\ell \geq 1$.

2. *Alternatively, $\hat{X}_t(\ell)$ can be computed as follows. Let $(c_0 = 1, c_1, c_2, \dots)$ be the impulse response of F^{-1} ; then:*

$$\hat{X}_t(\ell) - \mu = -c_1(\hat{X}_t(\ell-1) - \mu) - \dots - c_{\ell-1}(\hat{X}_t(1) - \mu) - c_\ell(x_t - \mu) - \dots - c_{t+\ell-t_0}(x_{t_0} - \mu) \quad \ell \geq 1 \quad (5.34)$$

where (x_{t_0}, \dots, x_t) is the differenced data observed up to time t , and where t_0 is the length of the impulse response of the differencing and de-seasonalizing filter L .

3. *The ℓ -step ahead predictions at time t , $\hat{Y}_t(\ell)$, of the non differenced data follow, using Proposition 5.1.*
4. *Let (d_0, d_1, d_2, \dots) be the impulse response of the filter $L^{-1}F$ and*

$$MSE_t^2(\ell) = \sigma^2 (d_0^2 + \dots + d_{\ell-1}^2) \quad (5.35)$$

A 95% prediction interval for $Y_{t+\ell}$ is

$$\hat{Y}_t(\ell) \pm 1.96 \sqrt{MSE_t^2(\ell)} \quad (5.36)$$

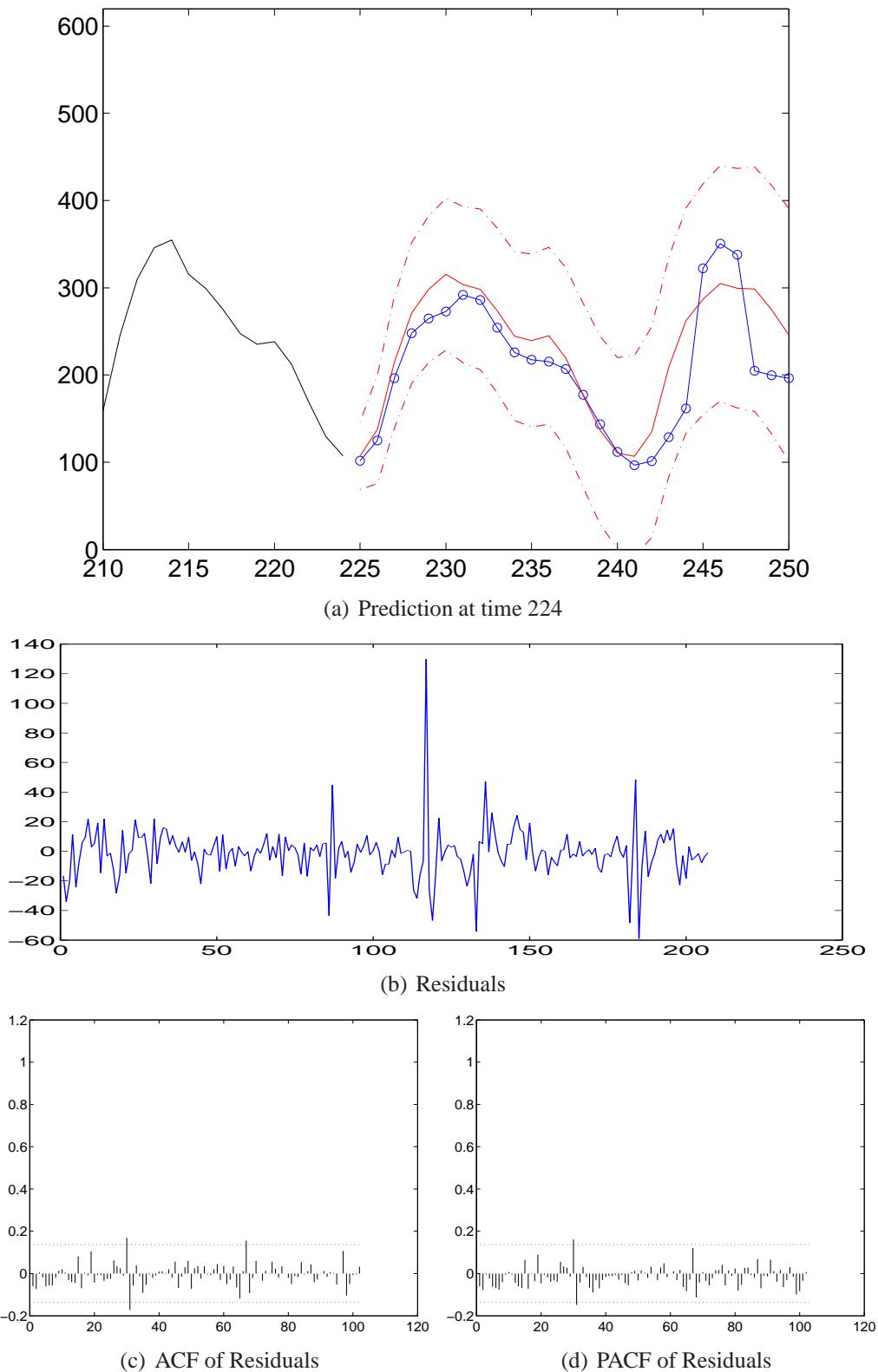


Figure 5.11: Prediction for internet traffic of Figure 5.1, using an ARMA model for the differenced data (o=actual value of the future, no known at time of prediction). Compare to Figure 5.7: the point predictions are almost identical, but the prediction intervals are more accurate (smaller).

Note that we use two steps for computing the point predictions: first for X_t , then for Y_t . One can wonder why, since one could use a single step, based on the fact that $Y = L^{-1}F\epsilon$. The reason is numerical stability: since the initial values of X_t (or equivalently, the past values Y_s for $s \leq 0$) are not known, there is some numerical error in items 1 and 2. Since we assume that F^{-1} is stable, $c_m \rightarrow 0$ for large m so the values of x_t for small t do not influence the final value of Eq.(5.34). Indeed the non-differenced data x_t for small values of t is not known exactly, as we made the simplifying assumption that $y_s = 0$ for $s \leq 0$. This is also why the first t_0 data points of x are removed in Eq.(5.34).

The problem does not exist for the computation of prediction intervals, this is why one can directly use a single step in item 4. This is because the variance of the forecast $\text{MSE}_t^2(\ell)$ is independent of the past data (Theorem C.6 in Appendix).

If one insists on using a model such that F is stable, but not F^{-1} , the theorem is still formally true, but may be numerically wrong. It is then preferable to use the formulae in Section 3.3 of [19] (but in practice one should avoid using such models).

POINT PREDICTIONS FOR AN AR(p) 0 MEAN PROCESS

The formulae have simple closed forms when there is no differencing or de-seasonalizing and the ARMA process is AR(p) with 0 mean. In such a case, $Y_t = X_t$ and Eq.(5.34) becomes (with the usual convention $y_s = 0$ for $s \leq 0$):

$$\begin{aligned}\hat{Y}_t(\ell) &= - \sum_{j=1}^{\ell-1} A_j \hat{Y}_t(\ell-j) - \sum_{j=\ell}^p A_j y_{t-j+\ell} \text{ for } 1 \leq \ell \leq p \\ \hat{Y}_t(\ell) &= - \sum_{j=1}^p A_j \hat{Y}_t(\ell-j) \text{ for } \ell > p\end{aligned}$$

where A_1, A_2, \dots, A_p are the auto-regressive coefficients as in Eq.(5.21). Because of this simplicity, AR processes are often used, e.g. when real time predictions are required.

EXAMPLE 5.6: INTERNET TRAFFIC, CONTINUED. The differenced data in Figure 5.10 appears to be stationary and has decaying ACF. We model it as a 0 mean ARMA(p, q) process with $p, q \leq 20$ and fit the models to the data. The resulting models have very small coefficients A_m and C_m except for m close to 0 or above to 16. Therefore we re-fit the model by forcing the parameters such that

$$\begin{aligned}A &= (1, A_1, \dots, A_p, 0, \dots, 0, A_{16}, \dots, A_{16+p}) \\ C &= (1, C_1, \dots, C_p, 0, \dots, 0, C_{16}, \dots, C_{16+q})\end{aligned}$$

for some p and q . The model with smallest AIC in this class is for $p = 1$ and $q = 3$.

Figure 5.11 shows the point predictions and the prediction intervals for the original data. They were obtained by first computing point predictions for the differenced data (using Matlab's `predict` routine) and applying Proposition 5.1. The prediction intervals are made using Proposition 5.4. Compare to Figure 5.7: the point predictions are only marginally different, but the confidence intervals are much better.

We also plot the residuals and see that they do appear uncorrelated, but there are some large values that do not appear to be compatible with the gaussian assumption. Therefore the prediction

intervals might be pessimistic. We computed point predictions and prediction intervals by re-sampling from residuals. Figure 5.12 shows that the confidence intervals are indeed smaller.

USE OF BOOTSTRAP REPLICATES

When the residuals appear to be uncorrelated but non gaussian, the prediction intervals may be over or under-estimated. It is possible to avoid the problem using a Monte Carlo method (Section 6.4), as explained now.

The idea is to draw many independent predictions for the residuals, from which we can derive predictions for the original data (by using reverse filters). There are several possibilities for generating independent predictions for the residuals: one can fit a distribution, or use Bootstrap replicates (i.e. re-sample from the residuals with replacement). We give an algorithm using this latter solution.

Algorithm 3 Monte-Carlo computation of prediction intervals at level $1 - \alpha$ for time series Y_t using resampling from residuals. We are given: a data set Y_t , a differencing and de-seasonalizing filter L and an ARMA filter F such that the residual $\epsilon = F^{-1}LY$ appears to be iid; the current time t , the prediction lag ℓ and the confidence level α . r_0 is the algorithm's accuracy parameter.

- 1: $R = \lceil 2r_0/\alpha \rceil - 1$ ▷ For example $r_0 = 25$, $R = 999$
 - 2: compute the differenced data $(x_1, \dots, x_t) = L(y_1, \dots, y_t)$
 - 3: compute the residuals $(e_q, \dots, e_t) = F^{-1}(x_q, \dots, x_t)$ where q is an initial value chosen to remove initial inaccuracies due to differencing or de-seasonalizing (for example $q = \text{length of impulse response of } L$)
 - 4: **for** $r = 1 : R$ **do**
 - 5: draw ℓ numbers with replacement from the sequence (e_q, \dots, e_t) and call them $e_{t+1}^r, \dots, e_{t+\ell}^r$
 - 6: let $e^r = (e_q, \dots, e_t, e_{t+1}^r, \dots, e_{t+\ell}^r)$
 - 7: compute $X_{t+1}^r, \dots, X_{t+\ell}^r$ using $(x_q, \dots, x_t, X_{t+1}^r, \dots, X_{t+\ell}^r) = F(e^r)$
 - 8: compute $Y_{t+1}^r, \dots, Y_{t+\ell}^r$ using Proposition 5.1 (with X_{t+s}^r and Y_{t+s}^r in lieu of $\hat{X}_t(s)$ and $\hat{Y}_t(s)$)
 - 9: **end for**
 - 10: $(Y_{(1)}, \dots, Y_{(R)}) = \text{sort}(Y_{t+\ell}^1, \dots, Y_{t+\ell}^R)$
 - 11: Prediction interval is $[Y_{(r_0)} ; Y_{(R+1-r_0)}]$
-

The algorithm is basic in that it gives no information about its accuracy. A larger r_0 produces a better accuracy; a more sophisticated algorithm would set r_0 such that the accuracy is small.

Also note that, as any bootstrap method, it will likely fail if the distribution of the residuals is heavy tailed.

An alternative to the bootstrap is to fit a parametric distribution to the residuals; the algorithm is the same as Algorithm 3 except that line 5 is changed by the generation of a sample residual from its distribution.

5.6 SPARSE ARMA AND ARIMA MODELS

In order to avoid overfitting, it is desirable to use ARMA models that have as few parameters as possible. Such models are called **sparse**. The use of an information criterion gives a means to

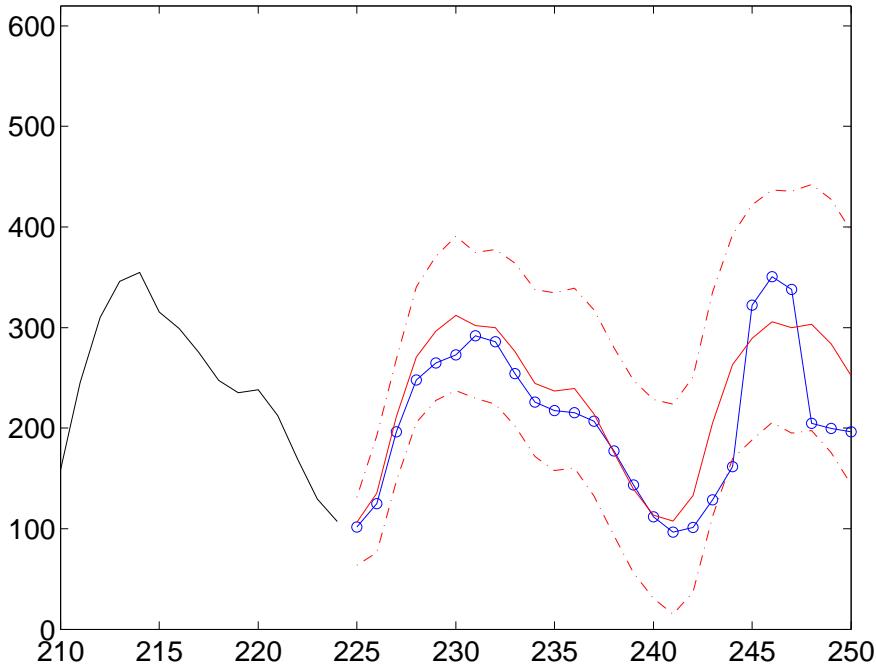


Figure 5.12: Prediction at time 224, same model as Figure 5.11, but prediction obtained with the bootstrap method (re-sampling from residuals).

obtain sparse models, but it involves a complex non linear optimization problem. An alternative is to impose constraints on the model, based on some sensible heuristics.

5.6.1 CONSTRAINED ARMA MODELS

A simple method consists in forcing some of the auto-regressive and moving average coefficients to 0, as in Example 5.6. Another method, more adapted to models with periodicity, is called **Seasonal ARIMA**. Assumes that the data has a period s ; a seasonal ARMA model is an ARMA model where we force the filter F defined in Eq.(5.23) to have the form

$$F = \frac{(1 + \sum_{i=1}^q c_i B^i) \left(1 + \sum_{i=1}^Q C_i B^{si}\right)}{(1 + \sum_{i=1}^p a_i B^i) \left(1 + \sum_{i=1}^P A_i B^{si}\right)} \quad (5.37)$$

Y_t is a seasonal ARIMA model $\Delta_1^d R_s^D Y$ is a seasonal ARMA model, for some nonnegative integers d, D . This model is also called **multiplicative ARIMA** model, as the filter polynomials are products of polynomials.

The only difference with the rest of this section when using a seasonal ARIMA model is the fitting procedure, which optimizes the model parameters subject to the constraints (using Theorem 5.2). The forecasting formulae are the same as for any ARIMA or ARMA model.

5.6.2 HOLT-WINTERS MODELS

These are simple models, with few parameters, which emerged empirically, but can be explained as ARMA or ARIMA models with few parameters. Their interest lies in the simplicity of both fitting and forecasting. Holt Winters models were originally introduced by Holt and Winters in [41, 107], and later refined by Roberts in [89]; we follow the presentation in this latter reference. We discuss five models: EWMA, double EWMA and three variants of the Holt Winters seasonal model.

EXPONENTIALLY WEIGHTED MOVING AVERAGE

This was originally defined as an ad-hoc forecasting formula. The idea is to keep a running estimate \hat{m}_t of the mean of the data, and update it using the *exponentially weighted moving average* mechanism with parameter a , defined for $t \geq 2$ by:

$$\hat{m}_t = (1 - a)\hat{m}_{t-1} + aY_t \quad (5.38)$$

with initial condition $\hat{m}_1 = Y_1$. The point forecast is then simply

$$\hat{Y}_t(\ell) = \hat{m}_t \quad (5.39)$$

The following results makes the link to ARMA models (proof in Section 5.7).

PROPOSITION 5.5 ([89]). *EWMA with parameter a is equivalent to modelling the non-differenced time series with the ARIMA(0, 1, 1) model defined by*

$$(1 - B)Y = (1 - (1 - a)B)\epsilon \quad (5.40)$$

with $\epsilon_t \sim iidN_{0,\sigma^2}$

The parameter a can be found by fitting the ARIMA model as usual, using Theorem 5.2, namely, by minimizing the one step ahead forecast error. There is no constraint on a , though it is classical to take it between 0 and 1.

The noise variance σ^2 can be estimated using Eq.(5.29), which, together with Proposition 5.4, can be used to find prediction intervals.

EWMA works well only when the data has no trend or periodicity, see Figure 5.13.

QUESTION 5.6.1. *What is EWMA for $a = 0$? $a = 1$?*⁹

DOUBLE EXPONENTIAL SMOOTHING WITH REGRESSION

This is another simple model that can be used for data with trend but no season. Like simple EWMA, it is based on ad-hoc forecasting formulae that happen to correspond to ARIMA models. The idea is to keep a running estimate of both the mean level \hat{m}_t and the trend \hat{r}_t . Further, a discounting factor ϕ is applied to model practical cases where the growth is not linear.

⁹ $a = 0$: a constant, equal to the initial value; $a = 1$: no smoothing, $\hat{m}_t = Y_t$.

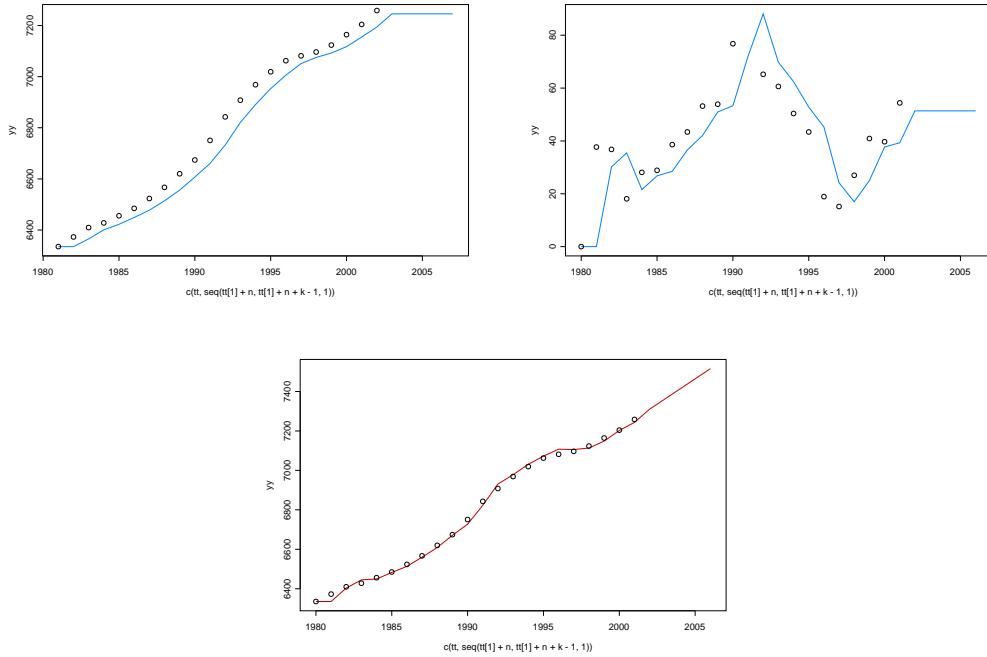


Figure 5.13: First graph: simple EWMA applied to swiss population data Y_t with $a = 0.9$. EWMA is lagging behind the trend. Second graph: simple EWMA applied to the differenced series ΔY_t . Third graph: prediction reconstructed from the previous graph.

The forecasting equation is

$$\hat{Y}_t(\ell) = \hat{m}_t + \hat{r}_t \sum_{i=1}^{\ell} \phi^i \quad (5.41)$$

and the update equations are, for $t \geq 3$:

$$\hat{m}_t = (1 - a)(\hat{m}_{t-1} + \phi \hat{r}_{t-1}) + a Y_t \quad (5.42)$$

$$\hat{r}_t = (1 - b)\phi \hat{r}_{t-1} + b(\hat{m}_t - \hat{m}_{t-1}) \quad (5.43)$$

with initial condition $\hat{m}_2 = Y_2$ and $\hat{r}_2 = Y_2 - Y_1$. We assume $0 < \phi \leq 1$; there is no constraint on a and b , though it is classical to take them between 0 and 1.

For $\phi = 1$ we have the classical Holt Winters model, also called **Double Exponential Weighted Moving Average**; for $0 < \phi < 1$ the model is said “with regression”.

PROPOSITION 5.6 ([89]). *Double EWMA with regression is equivalent to modeling the non differenced data as the zero mean ARIMA(1, 1, 2) process defined by:*

$$(1 - B)(1 - \phi B)Y = (1 - \theta_1 B - \theta_2 B^2)\epsilon \quad (5.44)$$

with

$$\theta_1 = 1 + \phi - a - \phi ab \quad (5.45)$$

$$\theta_2 = -\phi(1 - a) \quad (5.46)$$

with $\epsilon_t \sim iidN_{0,\sigma^2}$.

Double EWMA is equivalent to the zero mean ARIMA(0, 2, 2) model:

$$(1 - B)^2 Y = (1 - \theta_1 B - \theta_2 B^2) \epsilon \quad (5.47)$$

with

$$\theta_1 = 2 - a - ab \quad (5.48)$$

$$\theta_2 = -(1 - a) \quad (5.49)$$

The maximum likelihood estimate of a, b and ϕ is obtained as usual by minimizing the one step ahead forecast error. Figure 5.14 shows an example of double EWMA.

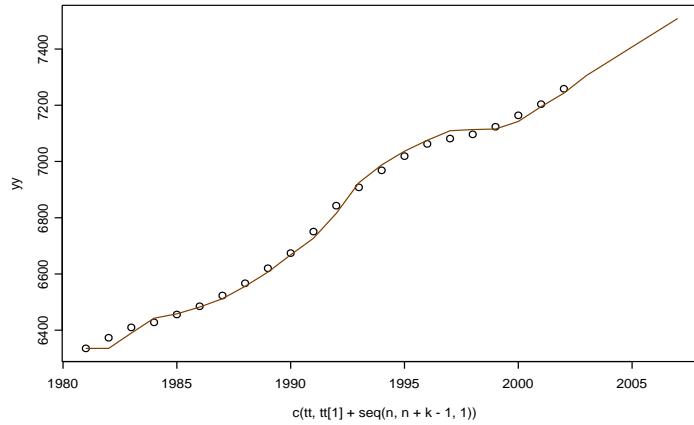


Figure 5.14: Double EWMA with $a = 0.8, b = 0.8$. It gives a good predictor; it underestimates the trend in convex parts, overestimates it in concave parts.

SEASONAL MODELS

For times series with a periodic behaviour there are extensions of the Holt Winters model, which keep the same simplicity, and can be explained as ARIMA models. We present three variants, which differ in the choice of some coefficients.

Assume that we know that the non differenced data has a period s . The idea is to keep the level and trend estimates \hat{m}_t and \hat{r}_t and introduce corrections for seasonality $\hat{s}_t(i)$, for $i = 0, \dots, s - 1$. The forecast equation is [89]:

$$\hat{Y}_t(\ell) = \hat{m}_t + \sum_{i=1}^{\ell} \phi^i \hat{r}_t + w^{\ell} \hat{s}_t(\ell \bmod s) \quad (5.50)$$

where ϕ and w are discounting factors. The update equations are, for $t \geq s + 2$:

$$\hat{m}_t = a(Y_t - w\hat{s}_{t-1}(1)) + (1 - a)(\hat{m}_{t-1} + \phi\hat{r}_{t-1}) \quad (5.51)$$

$$\hat{r}_t = b(\hat{m}_t - \hat{m}_{t-1}) + (1 - b)\phi\hat{r}_{t-1} \quad (5.52)$$

$$\hat{s}_t(i) = w\hat{s}_{t-1}((i+1) \bmod s) + D_i e_t \text{ for } i = 0 \dots s - 1 \quad (5.53)$$

where D_i are coefficients to be specified next and $e_t = Y_t - \hat{Y}_{t-1}(1)$.

The initial values of \hat{m} , \hat{r} , \hat{s} are obtained by using the forecast equation with $t = 1$ and $\ell = 1 \dots s$. More precisely, set $\hat{m}_t = Y_t$ for $t = 1, \dots, s+1$, $\hat{r}_1 = r$, $\hat{s}_1(j) = s_j$ for $j = 0, \dots, s-1$, solve for $r, s_0, s_1, \dots, s_{s-1}$ in

$$\begin{aligned} Y_{j+1} &= Y_1 + r \sum_{i=1}^j \phi^i + w^j s_{j \bmod s} \text{ for } j = 1 \dots s \\ 0 &= \sum_{j=1}^s s_j \end{aligned}$$

and do $\hat{r}_{s+1} = \phi^s \hat{r}_1$, $\hat{s}_{s+1}(j) = w^s s_j$. After some algebra this gives the **initial conditions**:

$$\hat{m}_{s+1} = Y_{s+1} \quad (5.54)$$

$$\hat{r}_{s+1} = \frac{\sum_{j=1}^s (Y_{j+1} - Y_1) w^{s-j}}{\sum_{j=1}^s \sum_{i=1}^j \phi^{i-s} w^{s-j}} \quad (5.55)$$

$$\hat{s}_{s+1}(0) = Y_{s+1} - Y_1 - \hat{r}_{s+1} \sum_{i=1}^s \phi^{i-s} \quad (5.56)$$

$$\hat{s}_{s+1}(j) = \left(Y_{j+1} - Y_1 - \hat{r}_{s+1} \sum_{i=1}^j \phi^{i-s} \right) w^{s-j} \text{ for } j = 1, \dots, s-1 \quad (5.57)$$

Roberts argues we should impose $\sum_{i=0}^s D_i = 0$. **Roberts' Seasonal Model** is obtained by using an exponential family, i.e.

$$D_0 = 1 - c^{s-1} \quad (5.58)$$

$$D_i = -c^{i-1}(1 - c) \text{ for } i = 1, \dots, s-1 \quad (5.59)$$

for some parameter c .

PROPOSITION 5.7 ([89]). *The Roberts seasonal model with parameters a, b, c, ϕ, w is equivalent to the zero mean ARIMA model*

$$(1 - \phi B)(1 - B) \left(1 + \sum_{i=1}^{s-1} w^i B^i \right) Y = \left(1 - \sum_{i=1}^{s+1} \theta_i B^i \right) \epsilon \quad (5.60)$$

with $\epsilon_t \sim iidN_{0,\sigma^2}$ and

$$\begin{aligned} \theta_1 &= 1 + \phi - wc - a(1 + \phi b) \\ \theta_i &= w^{i-2} \{ c^{i-2} [(1 + \phi)wc - \phi - w^2 c^2] - (w - \phi)a - w\phi ab \} \\ &\quad \text{for } i = 2, \dots, s-1 \\ \theta_s &= w^{s-2} \{ c^{s-2} [(1 + \phi)wc - \phi] - (w - \phi)a - w\phi ab \} \\ \theta_{s+1} &= -\phi w^{s-1} (c^{s-1} - a) \end{aligned}$$

The *Holt-Winters Additive Seasonal Model* is also commonly used. It corresponds to $\phi = 1, w = 1$ (no discounting) and

$$D_0 = c(1 - a) \quad (5.61)$$

$$D_i = 0 \text{ for } i = 1, \dots, s - 1 \quad (5.62)$$

It seems more reasonable to impose $\sum_{i=0}^{s-1} D_i = 0$, and Roberts proposes a variant, the *Corrected Holt-Winters Additive Seasonal Model*, for which $\phi = 1, w = 1$ and

$$D_0 = c(1 - a) \quad (5.63)$$

$$D_i = -\frac{c(1 - a)}{s - 1} \text{ for } i = 1, \dots, s \quad (5.64)$$

PROPOSITION 5.8 ([89]). *The Holt-Winters Additive Seasonal models with parameters a, b, c are equivalent to the zero mean ARIMA models*

$$(1 - B)(1 - B^s)Y = \left(1 - \sum_{i=1}^{s+1} \theta_i B^i\right)\epsilon \quad (5.65)$$

with $\epsilon_t \sim iidN_{0,\sigma^2}$ and

$$\begin{aligned} \theta_1 &= (1 - a)(1 + ch) - ab \\ \theta_i &= -ab \text{ for } i = 2, \dots, s - 1 \\ \theta_s &= 1 - ab - (1 - a)c(1 + h) \\ \theta_{s+1} &= -(1 - a)(1 - c) \end{aligned}$$

with $h = \frac{1}{s-1}$ (*Corrected Holt-Winters Additive Seasonal model*) and $h = 0$ (*Holt-Winters Additive Seasonal model*).

For all of these models, parameter estimation can be done by minimizing the mean square one step ahead forecast error. Prediction intervals can be obtained from the ARIMA model representations. There are many variants of the Holt Winters seasonal model; see for example [48] for the multiplicative model and other variants.

EXAMPLE 5.7: INTERNET TRAFFIC WITH ROBERTS MODEL. We applied the seasonal models in this section to the data set of Figure 5.1; the results are in Figure 5.15. We fitted the models by maximum likelihood, i.e. minimizing the one step ahead forecast error. We obtained prediction intervals by using the ARIMA representation and Proposition 5.4.

The best Roberts seasonal model is for $a = 1, b = 0.99, c = 0.90, \phi = 0.050$ and $w = 1$. The best Holt Winters additive seasonal model is for $a = 0.090, b = 0.037$ and $c = 0.64$. Both corrected and non corrected Holt Winters additive seasonal models give practically the same results.

5.7 PROOFS

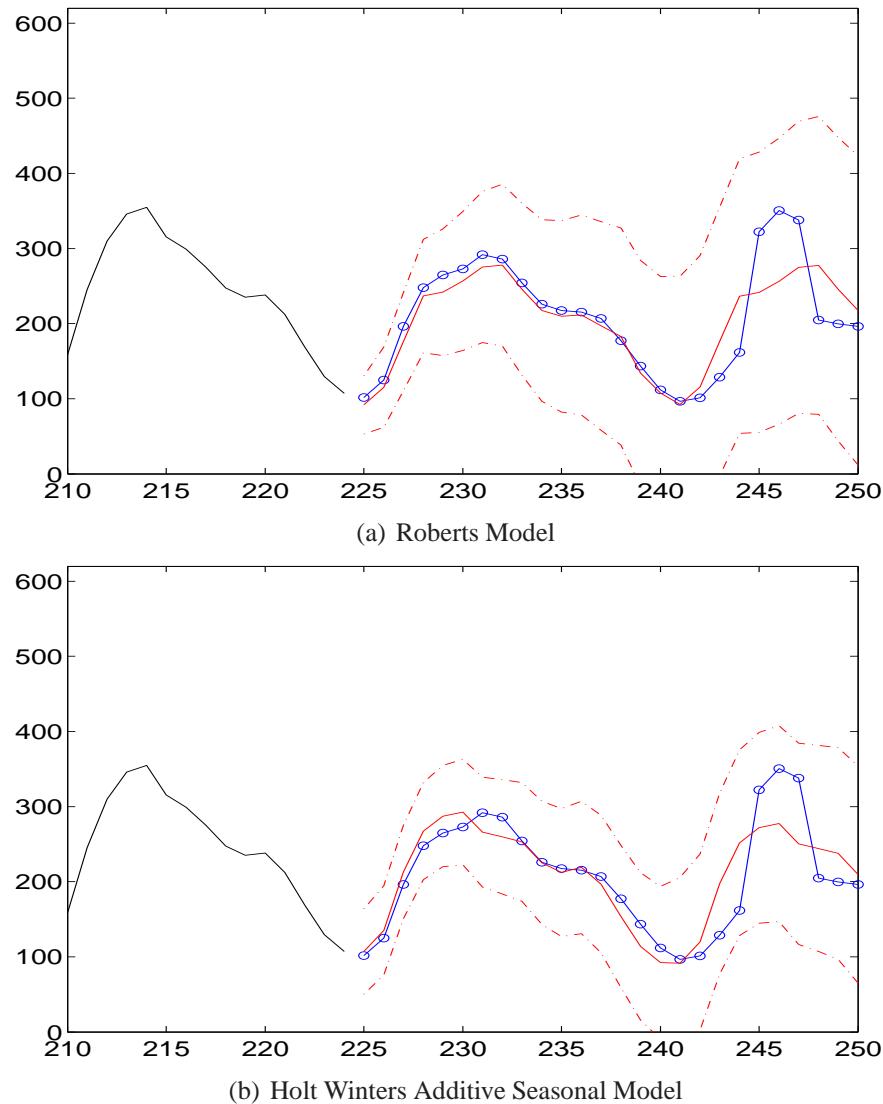


Figure 5.15: Prediction for internet traffic of Figure 5.1, using Additive Seasonal models. (o=actual value of the future, no known at time of prediction). The predictions are less accurate than in Figure 5.11 but the models are much simpler.

THEOREM 5.2 Let $X_t - \mu = F\epsilon_t$ where F is the ARMA filter and $\epsilon_t \sim \text{iid}N_{0,\sigma^2}$. We identify F with an $N \times N$ invertible matrix as in Eq.(D.6). Y_t is a gaussian vector with mean μ and covariance matrix $\Omega = \sigma^2 FF^T$. Thus the log-likelihood of the data x_1, \dots, x_N is

$$-\frac{N}{2} \ln(2\pi) - N \ln \sigma - \frac{1}{2\sigma^2} \left((x^T - \mu \vec{1}^T) F^{-T} F^{-1} (x - \mu \vec{1}) \right)$$

where x is the column vector of the data and $\vec{1}$ is the column vector with N rows equal to 1. For a given x and F the log-likelihood is maximum for

$$\hat{\sigma}^2 = \frac{1}{N} \left((x^T - \mu \vec{1}^T) F^{-T} F^{-1} (x - \mu \vec{1}) \right)^2$$

and is equal to $-\frac{N}{2} \ln(2\pi\hat{\sigma}^2)$. Now

$$\hat{\sigma}^2 = \frac{1}{N} \|F^{-1}(x - \mu \vec{1})\|^2$$

and, by definition of the model, $F^{-1}(x - \mu \vec{1})$ is the vector of residuals (i.e. the value of ϵ_t that correspond to the observed data x_1, \dots, x_N). Now use the innovation formula, Eq.(5.32), to conclude the proof.

PROPOSITION 5.5 Assume that EWMA corresponds to an ARIMA model. Let $\epsilon_t = Y_t - \hat{Y}_{t-1}(1)$ be the innovation sequence. Re-write Eq.(5.38) as

$$\hat{m}_t = \hat{m}_{t-1} + a\epsilon_t$$

Using filters, this writes as $\hat{m} = B\hat{m} + a\epsilon$. Combine with $Y = B\hat{m} + \epsilon$ and obtain $(1 - B)Y = (1 - (1 - a)B)\epsilon$, which is the required ARIMA model. Conversely, use the forecasting equations in Proposition 5.4) to show that we obtain the desired forecasting equations.

The proofs of Propositions 5.6, 5.7 and 5.8 are similar.

5.8 REVIEW QUESTIONS

QUESTION 5.8.1. Does the order in which differencing at lags 1 and 16 is performed matter? ¹⁰

QUESTION 5.8.2. When is EWMA adequate? ¹¹

QUESTION 5.8.3. When is double EWMA adequate? ¹²

QUESTION 5.8.4. When is a seasonal Holt Winters model adequate? ¹³

QUESTION 5.8.5. For ARMA and ARIMA models, what is the relation between the data Y_t , one point ahead forecasts $\hat{Y}_t(1)$ and innovation ϵ_t ? ¹⁴

QUESTION 5.8.6. How do we account for uncertainty due to model fitting when using linear regression models? ARMA models? ¹⁵

QUESTION 5.8.7. What should one be careful about when interpreting an ACF plot? ¹⁶

¹⁰No, because filters commute.

¹¹When the data is stationary and we want a very simple model.

¹²When the data has trends but no seasonality and we want a very simple model.

¹³When the data has trends and seasonality and we want a very simple model.

¹⁴ $Y_t = \hat{Y}_{t-1}(1) + \epsilon_t$, see Eq.(5.32).

¹⁵With linear regression models there are explicit formulas, assuming the residuals are gaussian. In most cases, the uncertainty due to fitting is negligible compared to forecasting uncertainty. For ARMA models, the formulas in this chapter simply ignore it.

¹⁶That the data appears stationary.

QUESTION 5.8.8. *What is the overfitting problem ?*¹⁷

QUESTION 5.8.9. *When do we need an ARIMA model rather than simply applying differencing filters ?*¹⁸

QUESTION 5.8.10. *How does one fit a Holt Winters model to the data ?*¹⁹

QUESTION 5.8.11. *What are sparse ARMA and ARIMA models ? Why do we use them ?*²⁰

QUESTION 5.8.12. *When do we need the bootstrap ?*²¹

QUESTION 5.8.13. *When do we use an information criterion ?*²²

¹⁷A model that fits the past data too well might be unable to predict the future.

¹⁸When the residuals after differencing appear to be very correlated.

¹⁹Like all ARMA or ARIMA models, by minimizing the average one step ahead forecast error, see Theorem 5.2.

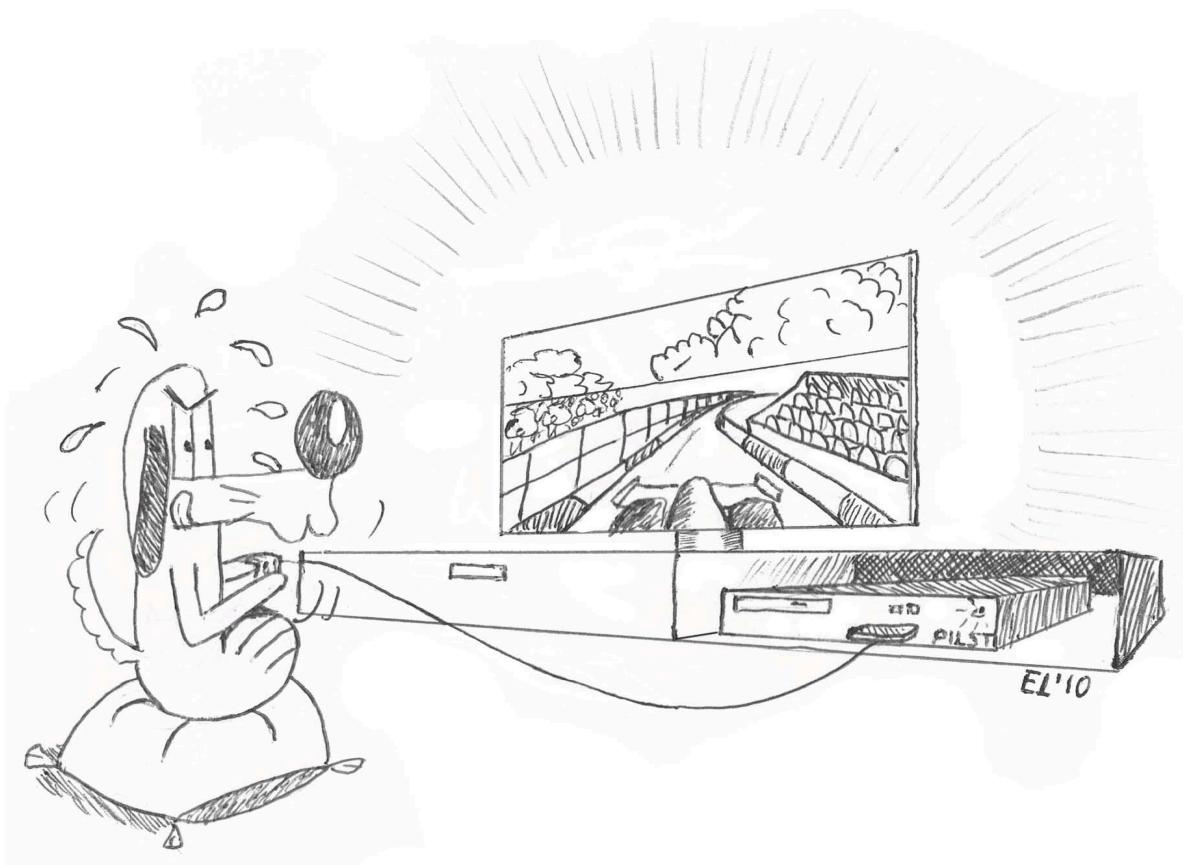
²⁰These are models with very few parameters, hence the computational procedures are much simpler.

²¹When the residuals appear iid but non gaussian and we want prediction intervals.

²²When we want to decide about the model order (number of parameters). It should be used only if the model seems to make sense for the data, otherwise the results may be aberrant.

CHAPTER 6

DISCRETE EVENT SIMULATION



Simulations are often regarded as the simplest, though most time consuming performance evaluation method. However, even simple simulation program may pose problems, if one is not aware of what stationarity means, and of the potential problems that arise when a simulation does not have a stationary regime. We start by discussing this simple, but important issue; the related topic of freezing simulations is in another chapter (Section 7.4).

Then we describe two commonly used techniques for implementing a simulation, namely, discrete events and stochastic recurrences, and discuss how confidence intervals can be applied to such settings. Then we discuss Monte Carlo simulation, viewed here as a method for computing integrals or probabilities, and potential pitfalls about random number generators. Then we present practical techniques for sampling from a distribution (CDF inversion, rejection sampling).

Importance sampling is an efficient technique for computing estimates of rare events, such as a failure rate or a bit error rate. The main difficulty is the choice of an importance sampling distribution. Here too, we propose a very general approach which is widely applicable and does not require heavy developments.

Contents

6.1	What is a Simulation ?	162
6.1.1	Simulated Time and Real Time	163
6.1.2	Simulation Types	163
6.2	Simulation Techniques	166
6.2.1	Discrete Event Simulation	166
6.2.2	Stochastic Recurrence	169
6.3	Computing the Accuracy of Stochastic Simulations	171
6.3.1	Independent Replications	171
6.3.2	Computing Confidence Intervals	172
6.3.3	Non-Terminating Simulations	172
6.4	Monte Carlo Simulation	172
6.5	Random Number Generators	175
6.6	How to Sample from a Distribution	176
6.6.1	CDF Inversion	179
6.6.2	Rejection Sampling	181
6.6.3	Ad-Hoc Methods	183
6.7	Importance Sampling	184
6.7.1	Motivation	184
6.7.2	The Importance Sampling Framework	185
6.7.3	Selecting An Importance Sampling Distribution	189
6.8	Proofs	191
6.9	Review	193

6.1 WHAT IS A SIMULATION ?

A simulation is an experiment in the computer (biologists say “in silico”) where the real environment is replaced by the execution of a program.

EXAMPLE 6.1: MOBILE SENSORS. You want to build an algorithm A for a system of n wireless sensors, carried by mobile users, which send information to a central database. A simulation of the algorithm consists in implementing the essential features of the program in computer, with one instance of A per simulated sensor. The main difference between a simulation and a real implementation is that the real, physical world (here: the radio channel, the measurements done by sensors) is replaced by events in the execution of a program.

6.1.1 SIMULATED TIME AND REAL TIME

In a simulation the flow of time is controlled by the computer. A first task of your simulation program is to simulate parallelism: several parallel actions can take place in the real system; in your program, you serialize them. Serializing is done by maintaining a *simulated time*, which is the time at which an event in the real system is supposed to take place. Every action is then decomposed into instantaneous events (for example, the beginning of a transmission), and we assume that it is impossible that two instantaneous events take place exactly at the same time.

Assume for example that every sensor in Example 6.1 should send a message whenever there is a sudden change in its reading, and at most every 10 minutes. It may happen in your simulation program that two or more sensors decide to send a message simultaneously, say within a window of $10 \mu\text{s}$; your program may take much more than $10 \mu\text{s}$ of *real time* to execute these events. In contrast, if no event happens in the system during 5 minutes, your simulation program may jump to the next event and take just of few ms to execute 5 mn of simulated time. The real time depends on the performance of your computer (processor speed, amount of memory) and of your simulation program.

6.1.2 SIMULATION TYPES

There are many different types of simulations. We use the following classification.

DETERMINISTIC / STOCHASTIC. A deterministic simulation has no random components. It is used when we want to verify a system where the environment is entirely known, maybe to verify the feasibility of a schedule, or to test the feasibility of an implementation. In most cases however, this is not sufficient. The environment of the system is better modelled with a random component, which makes the output of the simulation also random.

TERMINATING / NON-TERMINATING. A terminating simulation ends when specific conditions occurs. For example, if we would like to evaluate the execution time of one sequence of operations in a well defined environment, we can run the sequence in the simulator and count the simulated time. A terminating simulation is typically used when

- we are interested in the lifetime of some system
- or when the inputs are time dependent

EXAMPLE 6.2: JOE'S COMPUTER SHOP. We are interested in evaluating the time it takes to serve n customers who request a file together at time 0. We run a simulation program that terminates at time T_1 when all users have their request satisfied. This is a terminating simulation; its output is the time T_1 .

ASYMPTOTICALLY STATIONARY / NON-STATIONARY. This applies to a non-terminating, stochastic simulation only. Stationarity is a property of the stochastic model being simulated. For an in-depth discussion of stationarity, see Chapter 7.

Very often, the state of the simulation depends on the **initial conditions** and it is difficult to find good initial conditions; for example, if you simulate an information server and start with empty buffers, you are probably too optimistic, since a real server system that has been running for some time has many data structures that are not empty. Stationarity is a solution to this problem: if your simulator has a unique stationary regime, its distribution of state becomes independent of the initial condition.

A stationary simulation is such that you gain no information about its age by analyzing it. For example, if you run a stationary simulation and take a snapshot of the state of the system at times 10 and 10'000 seconds, there is no way to tell which of the two snapshots is at time 10 or 10'000 seconds.

In practice, a non terminating simulation is rarely exactly stationary, but can be *asymptotically stationary*. This means that after some simulated time, the simulation becomes stationary.

More precisely, a simulation program with time independent inputs can always be thought of as the simulation of a Markov chain. A Markov chain is a generic stochastic process such that, in order to simulate the future after time t , the only information we need is the state of the system at time t . This is usually what happens in a simulation program. The theory of Markov chains (see Chapter 7) says that the simulation will either converge to some stationary behaviour, or will diverge. If we want to measure the performance of the system under study, it is most likely that we are interested in its stationary behaviour.

EXAMPLE 6.3: INFORMATION SERVER. An information server is modelled as a queue. The simulation program starts with an empty queue. Assume the arrival rate of requests is smaller than the server can handle. Due to the fluctuations in the arrival process, we expect some requests to be held in the queue, from time to time. After some simulated time, the queue starts to oscillate between busy periods and idle periods. At the beginning of the simulation, the behaviour is not typical of the stationary regime, but after a short time it becomes so (Figure 6.1 (a)).

If in contrast the model is unstable, the simulation output may show a non converging behaviour (Figure 6.1 (b)).

In practice, here are the main reasons for non asymptotic stationarity.

1. **Unstable** models: In a queuing system where the input rate is larger than the service capacity, the buffer occupancy grows unbounded. The longer the simulation is run, the larger the mean queue length is.
2. **Freezing** simulation: this is a more subtle form of non stationarity, where the system does

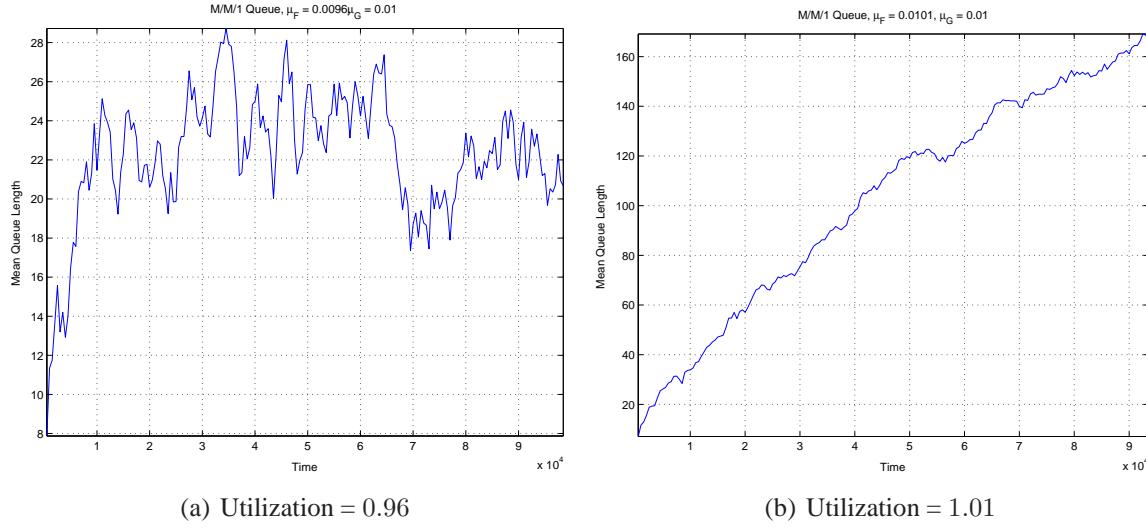


Figure 6.1: Simulation of the information server in Example 6.3, with exponential service and interarrival times. The graphs show the number of requests in the queue as a function of time, for two values of the utilization factor.

not converge to a steady state but, instead, freezes (becomes slower and slower). This is typically due to the occurrence of rare events with large impact. The longer the simulation, the more likely it is that a rare, but important event occurs, and the larger the impact of this event may be. If the simulation has regeneration points (points at which a clean state is reached, for example when the system becomes empty), then the simulation freezes if the time interval between regeneration points has an infinite mean. We study this topic in Section 7.4, where we see an example with the random waypoint.

3. Models with **seasonal or growth** components, or more generally, time dependent inputs; for example: internet traffic grows month after month and is more intense at some times of the day.

In most cases, when you perform a non-terminating simulation, **you should make sure that your simulation is in stationary regime**. Otherwise, the output of your simulation depends on the initial condition and the length of the simulation, and it is often impossible to decide what are realistic initial conditions. It is not always easy, though, to know in advance whether a given simulation model is asymptotically stationary. Chapter 7 gives some examples.

QUESTION 6.1.1. Among the following sequences X_n say which ones are stationary:

1. $X_n, n \geq 1$ is i.i.d.
2. $X_n, n \geq 1$ is drawn as follows. X_1 is sampled from a given distribution $F()$. To obtain $X_n, n \geq 2$ we first flip a coin (and obtain 0 with probability $1 - p$, 1 with probability p). If the coin returns 0 we let $X_n = X_{n-1}$; else we let $X_n = a$ new sample from the distribution $F()$.
3. $X_n = \sum_{i=1}^n Z_i, n \geq 1$, where $Z_n, n \geq 1$ is an i.i.d. sequence

1

¹1. yes 2. yes: (X_1, X_2) has the same joint distribution as, for example (X_{10}, X_{11}) . In general $(X_n, X_{n+1}, \dots, X_{n+k})$ has the same distribution for all n . This is an example of non-i.i.d., but stationary sequence. 3. No, in general. For example, if the common distribution $F()$ has a finite variance σ^2 , the variance of X_n is $n\sigma^2$, and grows with n , which is contradictory with stationarity.

6.2 SIMULATION TECHNIQUES

There are many ways to implement a simulation program. We describe two techniques that are commonly used in our context.

6.2.1 DISCRETE EVENT SIMULATION

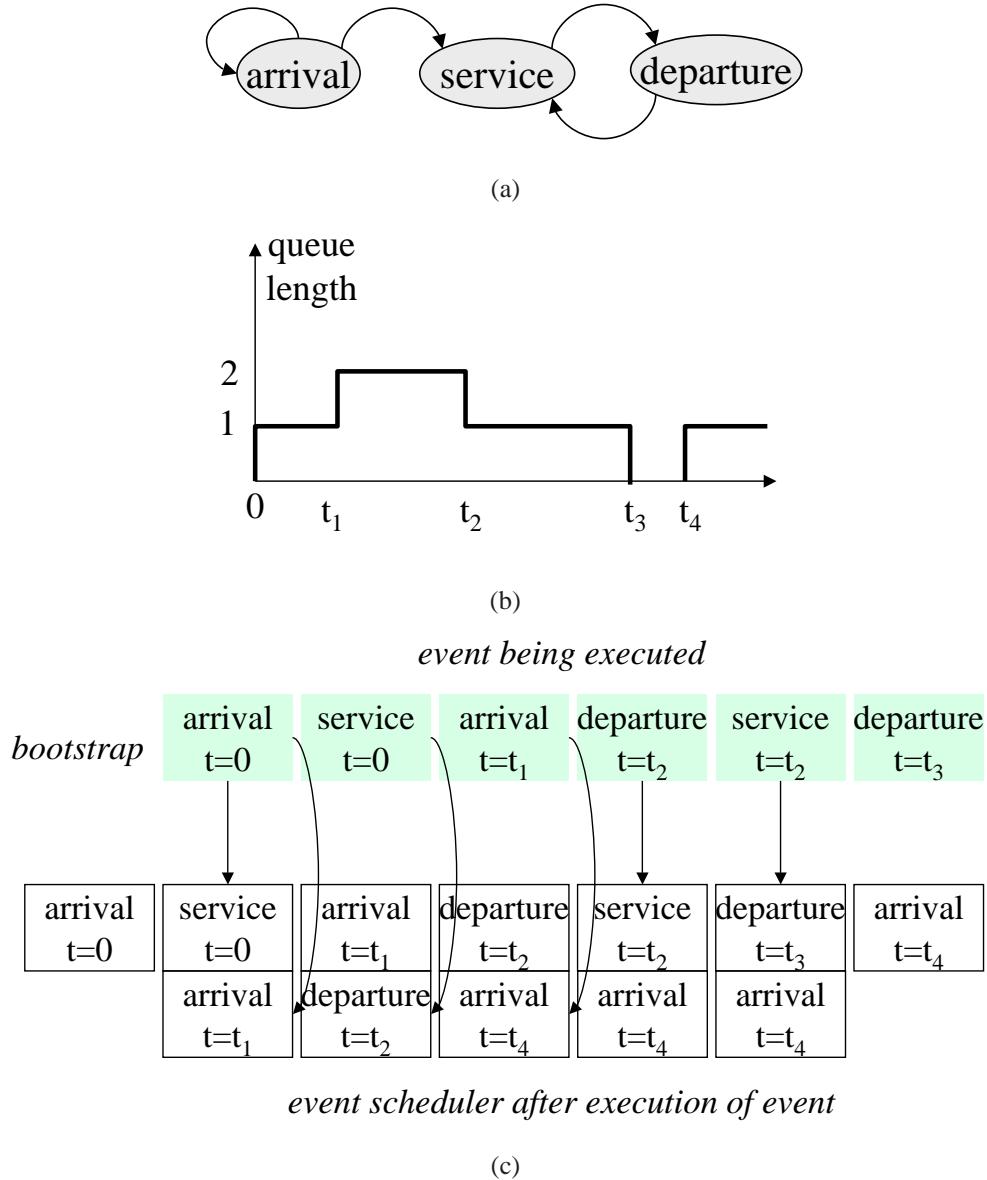


Figure 6.2: (a) Events and their dependencies for Example 6.4. An arrow indicates that an event may schedule another one. (b) A possible realization of the simulation and (c) the corresponding sequence of event execution. The arrows indicate that the execution of the event resulted in one or several new events being inserted into the scheduler.

Many computer and communication systems are often simulated using *discrete event simulation* for example with the ns2 or ns3 simulator [1]. It works as follows. The core of the method is

to use a global time `currentTime` and an *event scheduler*. Events are objects that represent different transitions; all event have an associated firing time. The event scheduler is a list of events, sorted by increasing firing times. The simulation program picks the first event in the event scheduler, advances `currentTime` to the firing time of this event, and executes the event. The execution of an event may schedule new events with firing times $\geq \text{currentTime}$, and may change or delete events that were previously listed in the event scheduler. The global simulation time `currentTime` **cannot** be modified by an event. Thus, the simulation time jumps from one event firing time to the next – hence the name of discrete event simulation. In addition to simulating the logic of the system being modelled, events have to update the counters used for statistics.

EXAMPLE 6.4: DISCRETE EVENT SIMULATION OF A SIMPLE SERVER. A server receives requests and serves them one by one in order of arrival. The times between request arrivals and the service times are independent of each other. The distribution of the time between arrivals has CDF $F()$ and the service time has CDF $G()$. The model is in fact a $GI/GI/1$ queue, which stands for general independent inter-arrival and service times. An outline of the program is given below. The program computes the mean response time and the mean queue length.

CLASSES AND OBJECTS We describe this example using an object oriented terminology, close to that of the Java programming language. All you need to know about object oriented programming to understand this example is as follows. An object is a variable and a class is a type. For example `arrival23` is the name of the variable that contains all information about the 23rd arrival, it is of the class `Arrival`. Classes can be nested, for example the class `Arrival` is a sub-class of `Event`. A method is a function whose definition depends on the class of the object. For example, the method `execute` is defined for all objects of the class `Event`, and is inherited by all subclasses such as `Arrival`. When the method `execute` is applied to the object `arrival23`, the actions that implement the simulation of an arrival are executed (for example, the counter of the number of requests in the system is incremented).

Global Variables and Classes

- `currentTime` is the global simulated time; it can be modified only by the main program.
- `eventScheduler` is the list of events, in order of increasing time.
- An event is an object of the class `Event`. It has an attribute `firingTime` which is the time at which it is to be executed. An event can be executed (i.e. the `Event` class has a method called `execute`), as described later.

There are three `Event` subclasses: an event of the class `Arrival` represents the actions that occur when a request arrives; `Service` is when a request enters service; `Departure` is when a request leaves the system. The event classes are described in detail later.

- The object `buffer` is the FIFO queue of Requests. The queue length (in number of requests) is `buffer.length`. The number of requests served so far is contained in the global variable `nbRequests`. The class `Request` is used to describe the requests arriving at the server. At a given point in time, there is one object of the class `Request` for every request present in the system being modelled. An object of the class `Request` has an arrival time attribute.
- Statistics Counters: `queueLengthCtr` is $\int_0^t q(s)ds$ where $q(s)$ is the value of `buffer.length` at time s and t is the current time. At the end of the simulation, the mean queue length is `queueLengthCtr/T` where T is the simulation finish time.

The counter `responseTimeCtr` holds $\sum_{m=1}^n R_m$ where R_m is the response time for the m th request and n is the value of `nbRequests` at the current time. At the end of the

simulation, the mean response time is `responseTimeCtr/N` where N is the value of `nbRequests`.

Event Classes. For each of the three event classes, we describe now the actions taken when an event of this class is “executed”.

- **Arrival:** *Execute Event’s Actions.* Create a new object of class `Request`, with arrival time equal to `currentTime`. Queue it at the tail of `buffer`.
Schedule Follow-Up Events. If `buffer` was empty before the insertion, create a new event of class `Service`, with the same `firingTime` as this event, and insert it into `eventScheduler`. Draw a random number Δ from the distribution $F()$. Create a new event of class `Arrival`, with `firingTime` equal to this event `firingTime+Δ`, and insert it into `eventScheduler`.
- **Service:** *Schedule Follow-Up Events.* Draw a random number Δ from the distribution $G()$. Create a new event of class `Departure`, with `firingTime` equal to this event’s `firingTime+Δ`, and insert it into `eventScheduler`.
- **Departure:** *Update Event Based Counters.* Let c be the request at the head of `buffer`. Increment `responseTimeCtr` by $d - a$, where d is this event’s `firingTime` and a is the arrival time of the request c . Increment `nbRequests` by 1.
Execute Event’s Actions. Remove the request c from `buffer` and delete it.
Schedule Follow-Up Events. If `buffer` is not empty after the removal, create a new event of class `Service`, with `firingTime` equal to this event’s `firingTime`, and insert it into `eventScheduler`.

Main Program

- *Bootstrapping.* Create a new event of class `Arrival` with `firingTime` equal to 0 and insert it into `eventScheduler`.
- *Execute Events.* While the simulation stopping condition is not fulfilled, do the following.
 - *Increment Time Based Counters.* Let e be the first event in `eventScheduler`. Increment `queueLengthCtr` by $q(t_{\text{new}} - t_{\text{old}})$ where $q = \text{buffer.length}$, $t_{\text{new}} = e.firingTime$ and $t_{\text{old}} = \text{currentTime}$.
 - *Execute e.*
 - Set `currentTime` to $e.firingTime$
 - Delete e
- *Termination.* Compute the final statistics:
 $\text{meanQueueLength} = \text{queueLengthCtr}/\text{currentTime}$
 $\text{meanResponseTime} = \text{responseTimeCtr}/\text{nbRequests}$

Figure 6.2 illustrates the program.

QUESTION 6.2.1. *Can consecutive events have the same firing time ?*²

QUESTION 6.2.2. *What are the generic actions that are executed when an event is executed ?*³

QUESTION 6.2.3. *Is the model in Example 6.4 stationary ?*⁴

²Yes. In Example 6.4, a `Departure` event when the queue is not empty is followed by a `Service` event with the same firing time.

³1. Update Event Based Counters 2. Execute Event’s Actions 3. Schedule Follow-Up Events.

⁴It depends on the parameters. Let a [resp. b] be the mean of $F()$ [resp. $G()$]. The utilization factor of the queue is $\rho = \frac{b}{a}$. If $\rho < 1$ the system is stable and thus asymptotically stationary, else not (see Chapter 8).

QUESTION 6.2.4. *Is the mean queue length an event-based or a time-based statistic ? The mean response time ?⁵*

6.2.2 STOCHASTIC RECURRENCE

This is another simulation method; it is usually much more efficient than discrete event simulation, but requires more work on the model. A *stochastic recurrence* is a recurrence of the form:

$$\begin{cases} X_0 = x_0 \\ X_{n+1} = f(X_n, Z_n) \end{cases} \quad (6.1)$$

where X_n is the state of the system at the n th transition (X_n is in some arbitrary state space \mathcal{X}), x_0 is a fixed, given state in \mathcal{X} , Z_n is some stochastic process that can be simulated (for example a sequence of i.i.d. random variables, or a Markov chain), and f is a deterministic mapping. The simulated time T_n at which the n th transition occurs is assumed to be included in the state variable X_n .

EXAMPLE 6.5: RANDOM WAYPOINT.

The *random waypoint* is a model for a mobile point, and can be used to simulate the mobility pattern in Example 6.1. It is defined as follows. The state variable is $X_n = (M_n, T_n)$ where M_n is the position of the mobile at the n th transition (the n th “waypoint”) and T_n is the time at which this destination is reached. The point M_n is chosen at random, uniformly in a given convex area \mathcal{A} . The speed at which the mobile travels to the next waypoint is also chosen at random uniformly in $[v_{\min}, v_{\max}]$.

The random waypoint model can be cast as a stochastic recurrence by letting $Z_n = (M_{n+1}, V_{n+1})$, where M_{n+1}, V_{n+1} are independent i.i.d. sequences, such that M_{n+1} is uniformly distributed in \mathcal{A} and V_{n+1} in $[v_{\min}, v_{\max}]$. We have then the stochastic recurrence

$$X_{n+1} := (M_{n+1}, T_{n+1}) = (M_{n+1}, T_n + \frac{\|M_{n+1} - M_n\|}{V_n})$$

See Figure 6.3 for an illustration.

Once a system is cast as a stochastic recurrence, it can be simply simulated as a direct implementation of Eq.(6.1), for example in Matlab.

QUESTION 6.2.5. *Is the random waypoint model asymptotically stationary ?⁶*

STOCHASTIC RECURRENCE VERSUS DISCRETE EVENT SIMULATION It is always possible to express a stochastic simulation as a stochastic recurrence, as illustrated by the next example. Both representations may have very different memory and CPU requirements; which representation is best depends on the problem at hand.

EXAMPLE 6.6: **SIMPLE SERVER AS A STOCHASTIC RECURRENCE.** (Continuation of Example 6.4). Consider implementing the simple server in Example 6.4 as a stochastic recurrence. To simplify,

⁵Mean queue length: time based. Mean response time: event based.

⁶For $v_{\min} > 0$ it is asymptotically stationary. For $v_{\min} = 0$ it is not: the model “freezes” (the number of waypoints per time unit tends to 0). See Chapter 7 for a justification).

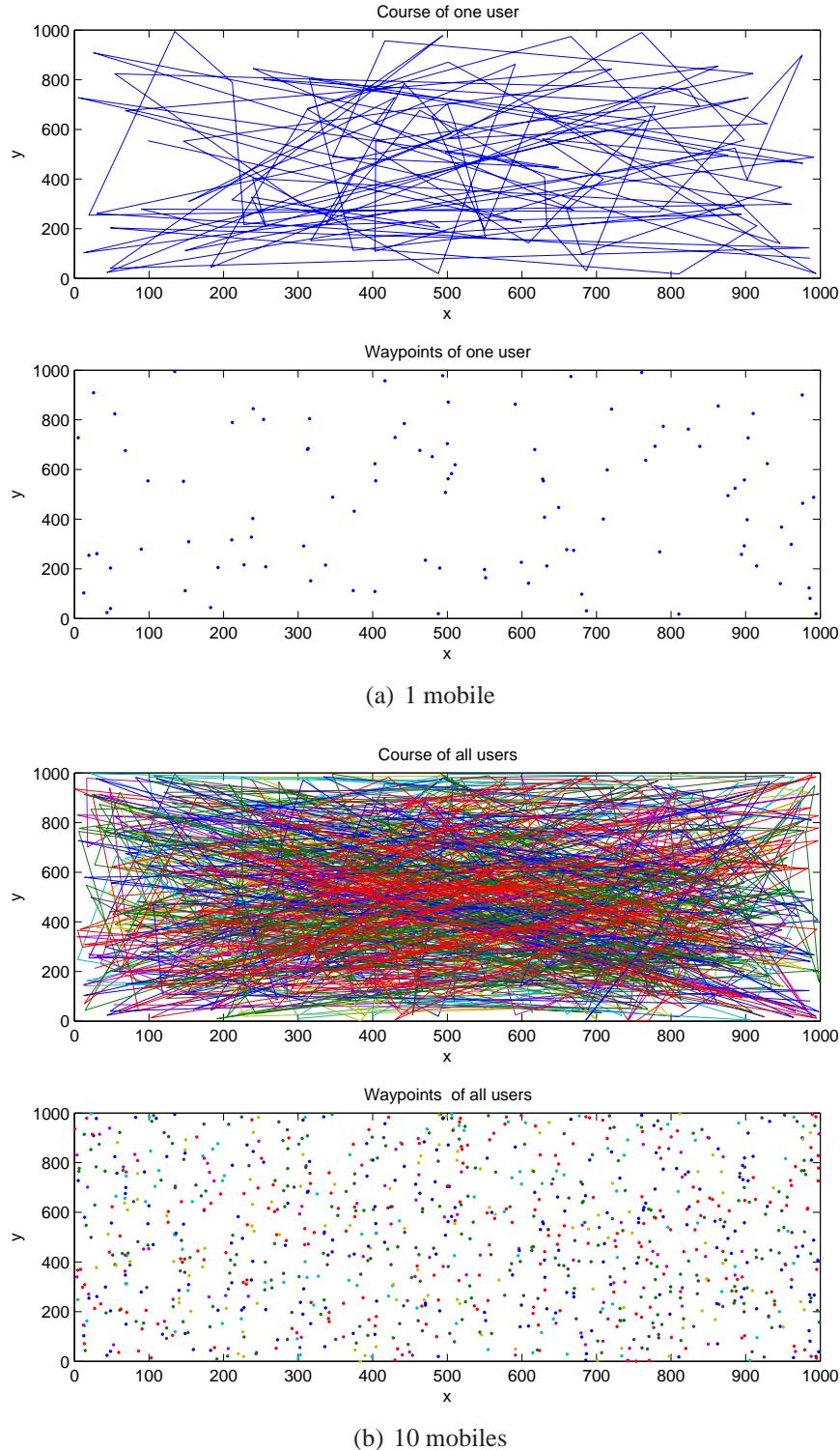


Figure 6.3: Simulation of the random waypoint model.

assume we are interested only in the mean queue length and not the mean response time. This can be implemented as a stochastic recurrence as follows. Let $X_n = (t_n, b_n, q_n, a_n, d_n)$ represent the state of the simulator just after an arrival or a departure, t_n = the simulated time at which this transition occurs, b_n = buffer.length, q_n = queueLengthCtr (both just after the transition),

a_n = the time interval from this transition to the next arrival and d_n = the time interval from this transition to the next departure.

Let Z_n be a couple of two random numbers, drawn independently of anything else, with distribution uniform in $(0, 1)$.

The initial state is

$$t_0 = 0, b_0 = 0, q_0 = 0, a_0 = F^{-1}(u), d_0 = \infty$$

where u is a sample of the uniform distribution on $(0, 1)$. The reason for the formula $a_0 = F^{-1}(u)$ is explained in Section 6.6: a_0 is a sample of the distribution with CDF $F()$.

The recurrence is defined by $f((t, b, q, a, d), (z_1, z_2)) = (t', b', q', a', d')$ with

```

if  $a < d$  // this transition is an arrival
     $\Delta = a$ 
     $t' = t + a$ 
     $b' = b + 1$ 
     $q' = q + b\Delta$ 
     $a' = F^{-1}(z_1)$ 
    if  $b == 0$  then  $d' = G^{-1}(z_2)$  else  $d' = d - \Delta$ 
else // this transition is a departure
     $\Delta = d$ 
     $t' = t + d$ 
     $b' = b - 1$ 
     $q' = q + b\Delta$ 
     $a' = a - \Delta$ 
    if  $b' > 0$  then  $d' = G^{-1}(z_1)$  else  $d' = \infty$ 
```

6.3 COMPUTING THE ACCURACY OF STOCHASTIC SIMULATIONS

A simulation program is expected to output some quantities of interest. For example, for a simulation of the algorithm A it may be the average number of lost messages. The output of a stochastic simulation is random: two different simulation runs produce different outputs. Therefore, it is not sufficient to give one simulation result; in addition, we need to give the accuracy of our results.

6.3.1 INDEPENDENT REPLICATIONS

A simple and very efficient method to obtain confidence intervals is to use *replication*. Perform n independent replications of the simulation, each producing an output x_1, \dots, x_n . **Be careful to have truly random seeds** for the random number generators, for example by accessing computer time (Section 6.5).

6.3.2 COMPUTING CONFIDENCE INTERVALS

You have to choose whether you want a confidence interval for the median or for the mean. The former is straightforward to compute, thus should be preferred in general.

Methods for computing confidence intervals for median and mean are summarized in Section 2.2.

EXAMPLE 6.7: APPLICATION TO EXAMPLE 6.2. Figure 6.4 shows the time it takes to transfer all files as a function of the number of customers. The simulation outputs do not appear to be normal, therefore we test whether n is large, by looking at the qqplot of the bootstrap replicates. We find that it looks normal, and we can therefore use the Student statistic. Out of curiosity, we also compute the bootstrap percentile estimate and find that both confidence intervals are very close, the bootstrap percentile estimate being slightly smaller.

There are other methods of obtaining confidence intervals, but they involve specific assumptions on the model and require some care; see for example [49].

6.3.3 NON-TERMINATING SIMULATIONS

Non-terminating simulations should be asymptotically stationary (Section 6.1.2). When you simulate such a model, you should be careful to do *transient removal*. This involves determining:

- when to start measuring the output (this is the time at which we consider that the simulation has converged to its stationary regime)
- when to stop the simulation

Unfortunately, there is no simple, bullet proof method to determine these two times. In theory, convergence to the stationary regime is governed by the value of the second eigenvalue modulus of the transition matrix of the markov chain that represents your simulation. In all but very special cases, it is impossible to estimate this value. A practical method for removing transients is to look at the data produced by the simulation, and visually determine a time after which the simulation output does not seem to exhibit a clear trend behaviour. For example, in Figure 6.1 (a), the measurements could safely start at time $t = 1$. This is the same stationarity test as with time series (Chapter 5).

Determining when to stop a simulation is more tricky. The simulation should be large enough for transients to be removable. After that, you need to estimate whether running the simulation for a long time reduces the variance of the quantities that you are measuring. In practice, this is hard to predict a priori. A rule of thumb is to run the simulation long enough so that the output variable looks gaussian across several replications, but not longer.

6.4 MONTE CARLO SIMULATION

Monte Carlo simulation is a method for computing probabilities, expectations, or, in general, integrals when direct evaluations is impossible or too complex. It simply consists in estimating the expectation as the mean of a number of independent replications.

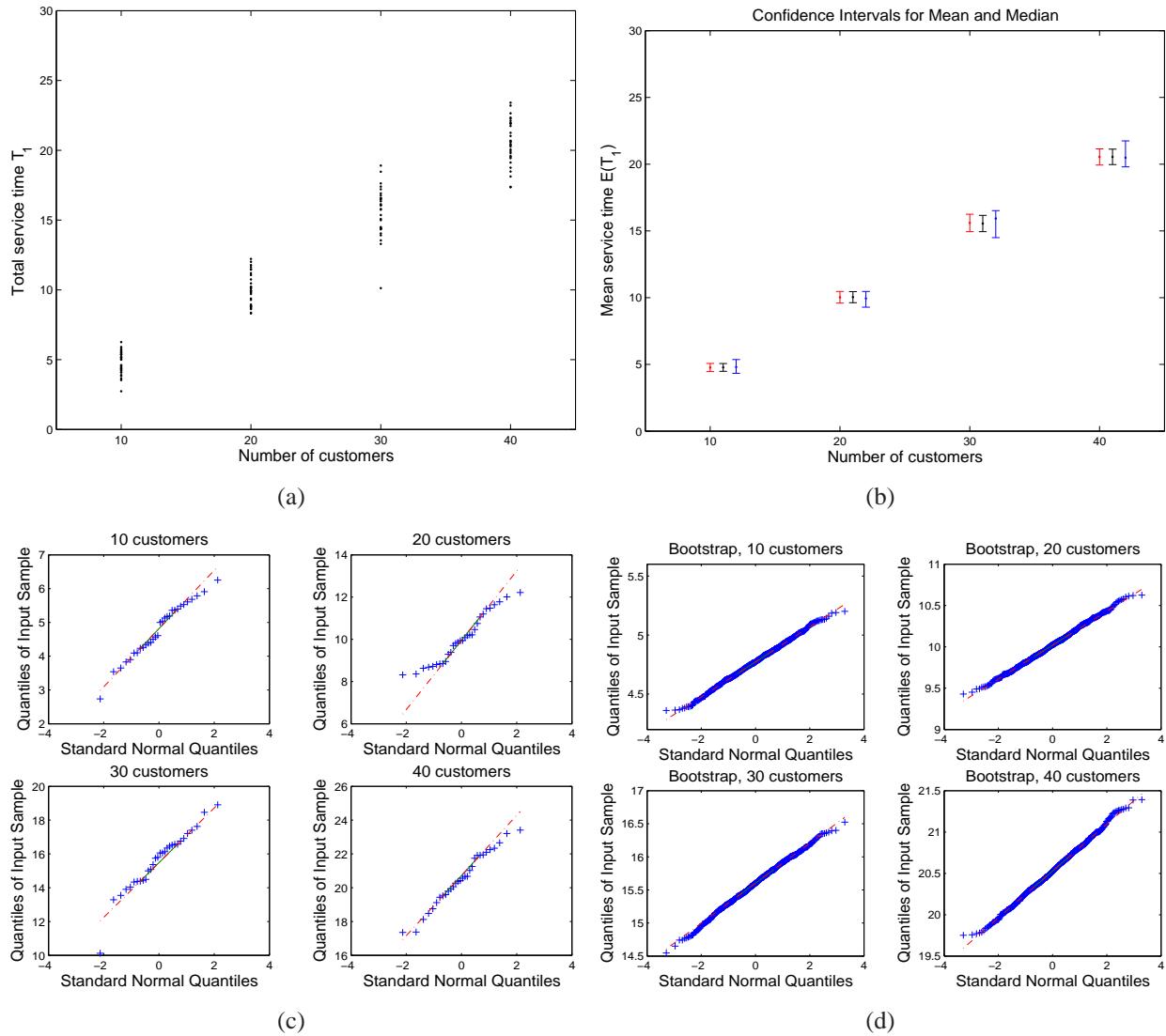


Figure 6.4: Time to serve n files in Joe's computer shop (Example 6.2): (a) results of 30 independent replications, versus number of customers (b) 95% confidence intervals for the mean obtained with the normal approximation (left), with the bootstrap percentile estimate (middle); 95% confidence interval for the median (right). (c) qqplot of simulation outputs, showing deviation from normality (d) qq-plots of the bootstrap replicates, showing normality.

Formally, assume we are given a model for generating a data sequence \vec{X} . The sequence may be i.i.d. or not. Assume we want to compute $\beta = \mathbb{E}(\varphi(\vec{X}))$. Note that this covers the case where we want to compute a probability: if $\varphi(\vec{x}) = \mathbf{1}_{\{\vec{x} \in A\}}$ for some set A , then $\beta = \mathbb{P}(\vec{X} \in A)$.

Monte-Carlo simulation consists in generating R i.i.d. replicates \vec{X}^r , $r = 1, \dots, R$. The Monte-Carlo estimate of β is

$$\hat{\beta} = \frac{1}{R} \sum_{r=1}^R \varphi(\vec{X}^r) \quad (6.2)$$

A confidence interval for β can then be computed using the methods in Chapter 2 (Theorem 2.2 and Theorem 2.4). By adjusting R , the number of replications, we can control the accuracy of the

method, i.e. the width of the confidence interval.

In particular, the theorem for confidence intervals of success probabilities (Theorem 2.4) should be used when the goal is to find an upper bound on a rare probability and the Monte Carlo estimate returns 0, as illustrated in the example below.

EXAMPLE 6.8: *p*-VALUE OF A TEST. Let X_1, \dots, X_n be a sequence of i.i.d. random variables that take values in the discrete set $\{1, 2, \dots, I\}$. Let $q_i = \mathbb{P}(X_k = i)$. Let $N_i = \sum_{k=1}^n \mathbf{1}_{\{X_k=i\}}$ (number of observation that are equal to i). Assume we want to compute

$$p = \mathbb{P}\left(\sum_{i=1}^k N_i \ln \frac{N_i}{nq_i} > a\right) \quad (6.3)$$

where $a > 0$ is given. This computation arises in the theory of goodness of fit tests, when we want to test whether X_i does indeed come from the model defined above (a is then equal to $\sum_{i=1}^k n_i \ln \frac{n_i}{nq_i}$ where n_i is our data set). For large values of the sample size n we can approximate p by a χ^2 distribution (see Section 4.4), but for small values there is no analytic result.

We use Monte-Carlo simulation to compute p . We generate R i.i.d. replicates X_1^r, \dots, X_n^r of the sequence ($r = 1, \dots, R$). This can be done by using the inversion method described in this chapter. For each replicate r , let

$$N_i^r = \sum_{k=1}^n \mathbf{1}_{\{X_k^r=i\}} \quad (6.4)$$

The Monte Carlo estimate of p is

$$\hat{p} = \frac{1}{R} \sum_{r=1}^R \mathbf{1}_{\{\sum_{i=1}^k N_i^r \ln \frac{N_i^r}{nq_i} > a\}} \quad (6.5)$$

Assuming that $\hat{p}R \geq 6$, we compute a confidence interval by using the normal approximation in Eq.(2.29). The sample variance is estimated by

$$\hat{\sigma} = \sqrt{\frac{\hat{p}(1 - \hat{p})}{R}} \quad (6.6)$$

and a confidence interval at level 0.95 is $\hat{p} \pm 1.96\hat{\sigma}$. Assume we want a **relative** accuracy at least equal to some fixed value ϵ (for example $\epsilon = 0.05$). This is achieved if

$$\frac{1.96\hat{\sigma}}{\hat{p}} \leq \epsilon \quad (6.7)$$

which is equivalent to

$$R \geq \frac{3.92}{\epsilon^2} \left(\frac{1}{\hat{p}} - 1 \right) \quad (6.8)$$

We can test for every value of R whether Eq.(6.8) is verified and stop the simulation when this happens. Table 6.1 shows some results with $n = 100$ and $a = 2.4$; we see that p is equal to 0.19 with an accuracy of 5%; the number of Monte Carlo replicates is proportional to the relative accuracy to the power -2 .

If $\hat{p}R < 6$ then we cannot apply the normal approximation. This occurs when the *p*-value to be estimated is very small. In such cases, typically, we are not interested in an exact estimate of the *p*-value, but we want to know whether it is smaller than some threshold α (for example $\alpha = 0.05$).

R	\hat{p}	margin
30	0.2667	0.1582
60	0.2500	0.1096
120	0.2333	0.0757
240	0.1917	0.0498
480	0.1979	0.0356
960	0.2010	0.0254
1920	0.1865	0.0174
3840	0.1893	0.0124
7680	0.1931	0.0088

Table 6.1: Computation of p in Example 6.8 by Monte Carlo simulation. The parameters of the model are $I = 4$, $q_1 = 9/16$, $q_2 = q_3 = 3/16$, $q_4 = 1/16$, $n = 100$ and $a = 2.4$. The table shows the estimate \hat{p} of p with its 95% confidence margin versus the number of Monte-Carlo replicates R . With 7680 replicates the relative accuracy (margin/ \hat{p}) is below 5%.

Eq.(2.26) and Eq.(2.27) can be used in this case. For example, assume the same data as in Table 6.1 except for $a = 18.2$. We do $R = 10^4$ monte carlo replicates and find $\hat{p}R = 0$. We can conclude, with confidence 95%, that $p \leq 1 - (0.025)^{\frac{1}{R}} = 3.7E - 4$.

QUESTION 6.4.1. In the first case of Example 6.8 (Table 6.1), what is the conclusion of the test ? In the second case ?⁷

6.5 RANDOM NUMBER GENERATORS

The simulation of any random process uses a basic function (such as `rand` in Matlab) that is assumed to return independent uniform random variables. Arbitrary distributions can be derived from there, as explained in Section 6.6.

In fact, `rand` is a *pseudo-random number generator*. It produces a sequence of numbers that appear to be random, but is in fact perfectly deterministic, and depends only on one initialization value of its internal state, called the *seed*. There are several methods to implement pseudo random number generators; they are all based on chaotic sequences, i.e. iterative processes where a small difference in the seed produces very different outputs.

Simple random number generators are based on *linear congruences* of the type $x_n = ax_{n-1} \bmod m$. Here the internal state after n calls to `rand` is the last output x_n ; the seed is x_0 . Like for any iterative algorithm, the sequence is periodic, but for appropriate choices of a and m , the period may be very large.

EXAMPLE 6.9: **LINEAR CONGRUENCE.** A widespread generator (for example the default in ns2) has $a = 16'807$ and $m = 2^{31} - 1$. The sequence is $x_n = \frac{sa^n \bmod m}{m}$ where s is the seed. m is a prime number, and the smallest exponent h such that $a^h \equiv 1 \pmod m$ is $m - 1$. It follows that for any value of the seed s , the period of x_n is exactly $m - 1$. Figure 6.5 shows that the sequence x_n indeed looks random.

⁷In the first case we accept the null hypothesis, i.e. we believe that the probability of case i is q_i . In the second case, the p -value is smaller than 0.05 so we reject the null hypothesis.

The period of a random number generator should be much larger than the number of times it is called in a simulation. The generator in Example 6.9 has a period of ca. 2×10^9 , which may be too small for very large simulations. There are other generators with much longer periods, for example the “Mersenne Twister” [67] with a period of $2^{19937} - 1$. They use other chaotic sequences and combinations of them.

Perfect pseudo-random number generators do not exist; only truly random generators can be perfect. Such generators exist: for example, a quantum mechanics generator is based on the fact that the state of a photon is believed to be truly random. For a general text on random number, see [59]; for software implementing good generators, see [60] and L’Ecuyer’s home page. For a general discussion of generators in the framework of simulation, see [40]. Figure 6.6 illustrates a potential problem when the random number generator does not have a long enough period.

USING A RANDOM NUMBER GENERATOR IN PARALLEL STREAMS For some (obsolete) generators as in Example 6.9, choosing small seed values in parallel streams may introduce a strong correlation (whereas we would like the streams to be independent).

EXAMPLE 6.10: PARALLEL STREAMS WITH INCORRECT SEEDS. Assume we need to generate two parallel streams of random numbers. This is very frequent in discrete event simulations; we may want to have one stream for the arrival process, and a second one for the service process. Assume we use the linear congruential generator of Example 6.9, and generate two streams x_n and x'_n with seeds $s = 1$ and $s' = 2$. Figure 6.7 shows the results: we see that the two streams are strongly correlated. In contrast, taking $s' =$ the last value x_N of the first stream does not have this problem.

More modern generators as mentioned above do not have this problem either.

SEEDING THE RANDOM NUMBER GENERATOR A safe way to make sure that replications are reasonably independent is to use the internal state of the generator at the end of the 1st replication as seed for the second replication and so one. This way, if the generator has a long enough sequence, the different replications have non overlapping sequences.

In practice, though, we often want independent replications to be run in parallel, so this mode of operation is not possible. A common practice is to take as seed a truly random number, for example derived from the computer clock or user input with the mouse.

6.6 HOW TO SAMPLE FROM A DISTRIBUTION

In this section we discuss methods to produce a sample X for a random variable that has a known distribution. We assume that we have a random number generator, that provides us with independent samples of the uniform distribution on $(0, 1)$. We focus on two methods of general applicability: CDF inversion and rejection sampling.

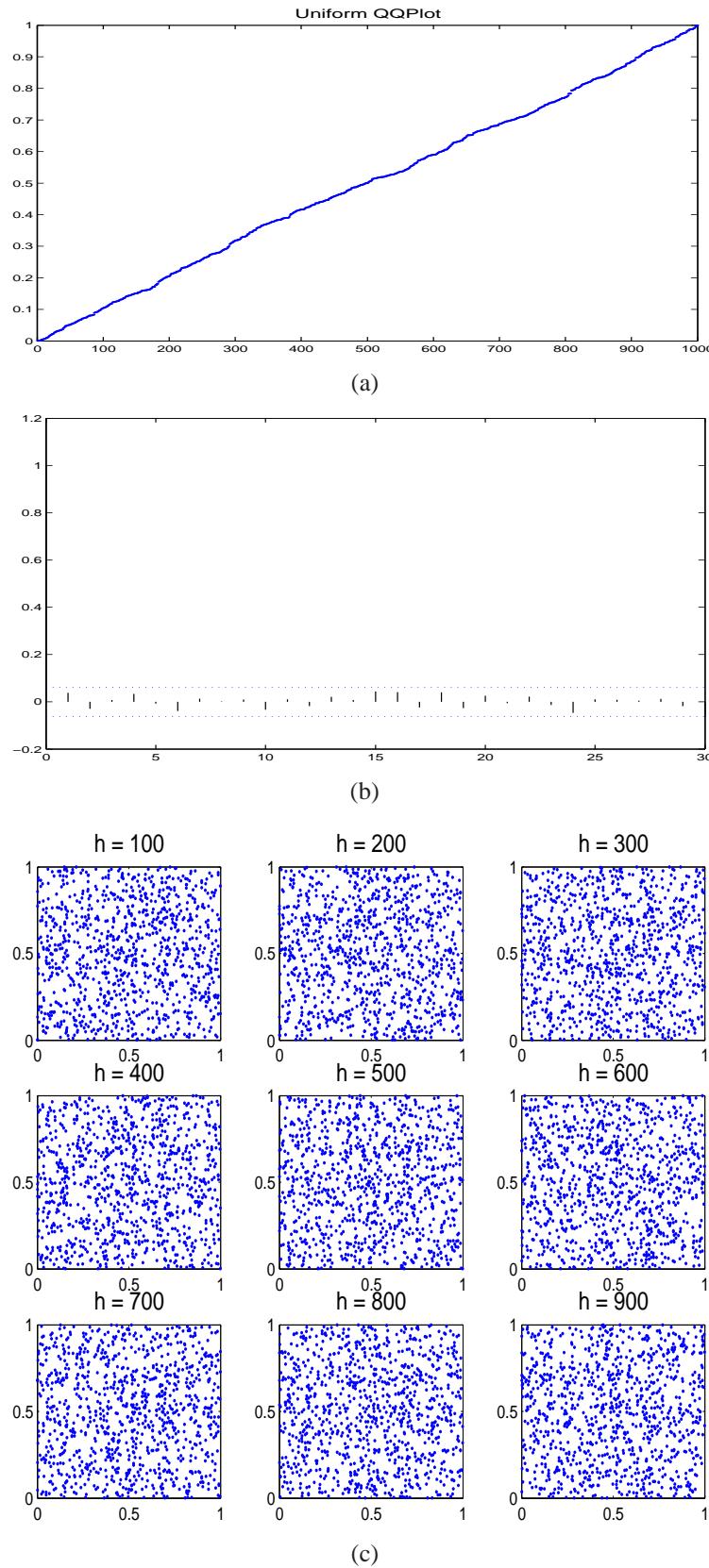


Figure 6.5: 1000 successive numbers for the generator in Example 6.9. (a) QQplot against the uniform distribution in $(0, 1)$, showing a perfect match. (b) autocorrelation function, showing no significant correlation at any lag (c) lag plots at various lags, showing independence.

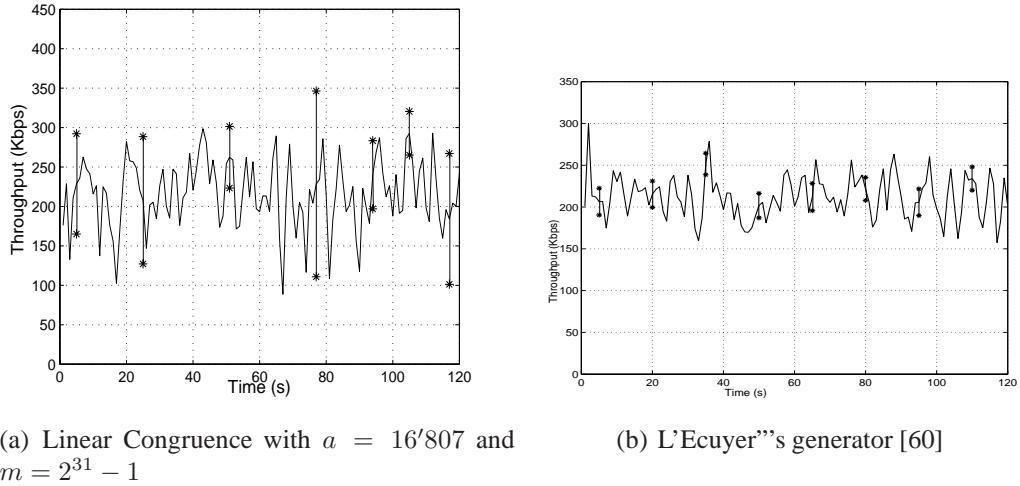


Figure 6.6: Simulation outputs for the throughput of TCP connections over a wireless ad-hoc network. The wireless LAN protocol uses random numbers for its operation. This simulation consumes a very large number of calls to `rand`. The simulation results obtained with both generators are different: Lecuyer's generator produces consistently smaller confidence intervals.

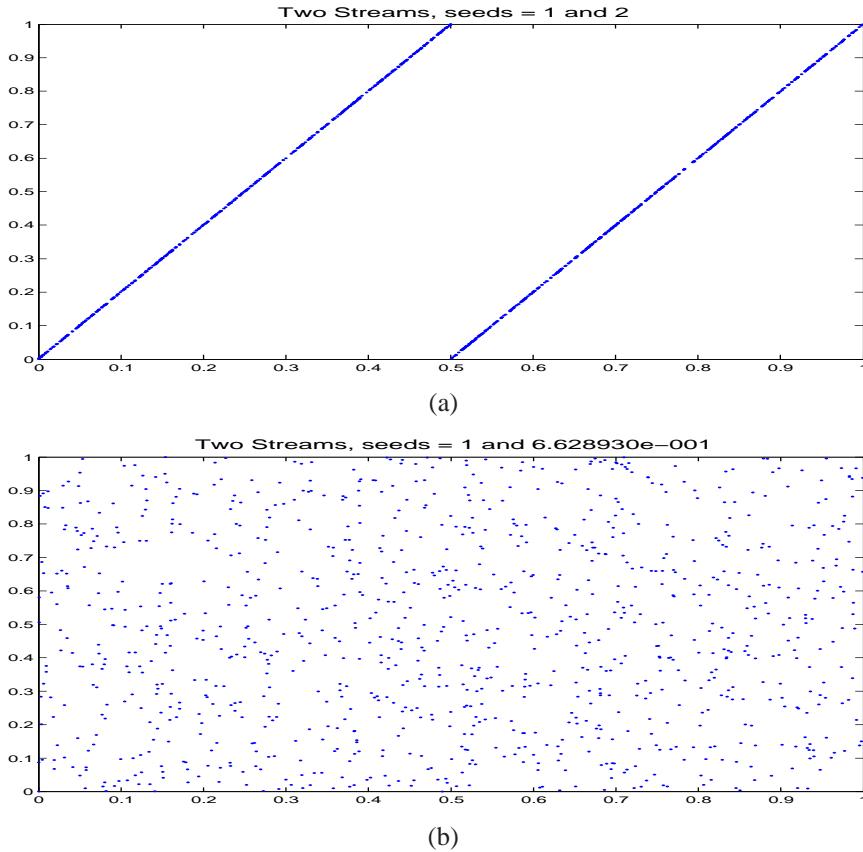


Figure 6.7: x_n versus x'_n for two streams generated with the linear congruential in Example 6.9. (a) seed values are 1 and 2 (b) seed values are (1, last value of first stream).

6.6.1 CDF INVERSION

The method of **CDF inversion**, also called **percentile inversion method**, applies to real or integer valued random variable, when the CDF is easy to invert.

THEOREM 6.1. Let F be the CDF of a random variable X with values in \mathbb{R} . Define the **pseudo-inverse**, F^{-1} of F by

$$F^{-1}(p) = \sup\{x : F(x) \leq p\}$$

Let U be a sample of a random variable with uniform distribution on $(0, 1)$; $F^{-1}(U)$ is a sample of X .

Application to real random variable. In the case where X has a positive density over some interval I , then F is continuous and strictly increasing on I , and the pseudo-inverse is the inverse of F , as in the next example. It is obtained by solving for x in the equation $F(x) = p$, $x \in I$.

EXAMPLE 6.11: EXPONENTIAL RANDOM VARIABLE. The CDF of the **exponential distribution** with parameter λ is $F(x) = 1 - e^{-\lambda x}$. The pseudo-inverse (which in this case is the plain inverse) is obtained by solving the equation

$$1 - e^{-\lambda x} = p$$

where x is the unknown. The solution is $x = -\frac{\ln(1-p)}{\lambda}$. Thus a sample X of the exponential distribution is obtained by letting $X = -\frac{\ln(1-U)}{\lambda}$, or, since U and $1-U$ have the same distribution:

$$X = -\frac{\ln(U)}{\lambda} \tag{6.9}$$

where U is the output of the random number generator.

Application to integer random variable. Assume N is a random variable with values in \mathbb{N} . Let $p_k = \mathbb{P}(N = k)$, then for $n \in \mathbb{N}$:

$$F(n) = \sum_{k=0}^n p_k$$

and for $x \in \mathbb{R}$:

$$\begin{cases} \text{if } x < 0 \text{ then } F(x) = 0 \\ \text{else } F(x) = \mathbb{P}(N \leq x) = \mathbb{P}(N \leq \lfloor x \rfloor) = F(\lfloor x \rfloor) \end{cases}$$

We now compute $F^{-1}(p)$, for $0 < p < 1$. Let n be the smallest integer such that $p < F(n)$. The set $\{x : F(x) \leq p\}$ is equal to $(-\infty, n)$ (Figure 6.8); the supremum of this set is n , thus $F^{-1}(p) = n$. In other words, the pseudo inverse is given by

$$F^{-1}(p) = n \Leftrightarrow F(n-1) \leq p < F(n) \tag{6.10}$$

Thus we have shown:

COROLLARY 6.1. Let N be a random variable with values in \mathbb{N} and let $p_k = \mathbb{P}(N = k)$, for $k \in \mathbb{N}$. A sample of N is obtained by setting N to the unique index $n \geq 0$ such that $\sum_{k=0}^{n-1} p_k \leq U < \sum_{k=0}^n p_k$, where U is the output of the random number generator.

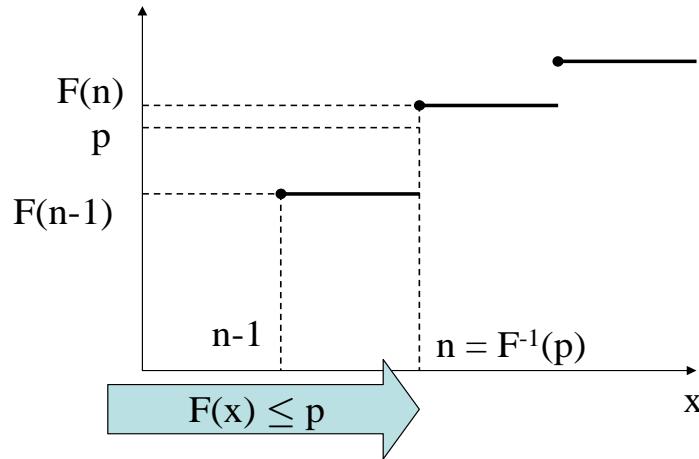


Figure 6.8: Pseudo-Inverse of CDF $F()$ of an integer-valued random variable

EXAMPLE 6.12: GEOMETRIC RANDOM VARIABLE. Here X takes integer values $0, 1, 2, \dots$. The *geometric distribution* with parameter θ satisfies $\mathbb{P}(X = k) = \theta(1 - \theta)^k$, thus for $n \in \mathbb{N}$:

$$F(n) = \sum_{k=0}^n \theta(1 - \theta)^k = 1 - (1 - \theta)^{n+1}$$

by application of Eq.(6.10):

$$F^{-1}(p) = n \Leftrightarrow n \leq \frac{\ln(1 - p)}{\ln(1 - \theta)} < n + 1$$

hence

$$F^{-1}(p) = \left\lfloor \frac{\ln(1 - p)}{\ln(1 - \theta)} \right\rfloor$$

and, since U and $1 - U$ have the same distribution, a sample X of the geometric distribution is

$$X = \left\lfloor \frac{\ln(U)}{\ln(1 - \theta)} \right\rfloor \quad (6.11)$$

QUESTION 6.6.1. Consider the function defined by $\text{COIN}(p) = \text{if } \text{rand}() < p \text{ else } 1$. What does it compute? ⁸

QUESTION 6.6.2. Compare Eq.(6.9) and Eq.(6.11). ⁹

⁸It generates a sample of the Bernoulli random variable that takes the value 0 with probability p and the value 1 with probability $1 - p$.

⁹They are similar, in fact we have $N = \lfloor X \rfloor$ if we let $\lambda = \ln(1 - \theta)$. This follows from the fact that if $X \sim \exp(\lambda)$, then $\lfloor X \rfloor$ is geometric with parameter $\theta = 1 - e^{-\lambda}$

6.6.2 REJECTION SAMPLING

The method of *rejection sampling* is widely applicable. It can be used to generate samples of random variables when the inversion method does not work easily. It applies to random vectors of any dimension.

The method is based on the following result, which is of independent interest. It allows to sample from a distribution given in conditional form.

THEOREM 6.2 (Rejection Sampling for a Conditional Distribution). *Let X be a random variable in some space S such that the distribution of X is the conditional distribution of \tilde{X} given that $\tilde{Y} \in \mathcal{A}$, where (\tilde{X}, \tilde{Y}) is a random variable in $S \times S'$ and \mathcal{A} is a measurable subset of S . A sample of X is obtained by the following algorithm:*

```

do
    draw a sample of  $(\tilde{X}, \tilde{Y})$ 
until  $\tilde{Y} \in \mathcal{A}$ 
return( $\tilde{X}$ )

```

The expected number of iterations of the algorithm is $\frac{1}{\mathbb{P}(\tilde{Y} \in \mathcal{A})}$.

EXAMPLE 6.13: DENSITY RESTRICTED TO ARBITRARY SUBSET. Consider a random variable in some space $(\mathbb{R}, \mathbb{R}^n, \mathbb{Z} \dots)$ that has a density $f_Y(y)$. Let \mathcal{A} be a set such that $\mathbb{P}(Y \in \mathcal{A}) > 0$. We are interested in the distribution of a random variable X whose density is that of Y , restricted to \mathcal{A} :

$$f_X(y) = K f_Y(y) \mathbf{1}_{\{y \in \mathcal{A}\}} \quad (6.12)$$

where $K^{-1} = \mathbb{P}(Y \in \mathcal{A}) > 0$ is a normalizing constant. This distribution is the conditional distribution of Y , given that $Y \in \mathcal{A}$.

QUESTION 6.6.3. *Show this.*¹⁰

Thus a sampling method for the distribution with density in Eq.(6.12) is to draw samples of the distribution with density f_Y until a sample is found that belongs to \mathcal{A} . The expected number of iterations is $1/\mathbb{P}(Y \in \mathcal{A})$.

For example, consider the sampling of a random point X uniformly distributed on some bounded area $\mathcal{A} \subset \mathbb{R}^2$. We can consider this density as the restriction of the uniform density on some rectangle $\mathcal{R} = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$ that contains the area \mathcal{A} . Thus a sampling method is to draw points uniformly in \mathcal{R} , until we find one in \mathcal{A} . The expected numbers of iterations is the ratio of the area of \mathcal{R} to that of \mathcal{A} ; thus one should be careful to pick a rectangle that is close to \mathcal{A} . Figure 6.9 shows a sample of the uniform distribution over a non-convex area.

QUESTION 6.6.4. *How can one generate a sample of the uniform distribution over \mathcal{R} ?*¹¹

Now we come to a very general result, for all distributions that have a density.

¹⁰For any (measurable) subset \mathcal{B} of the space, $\mathbb{P}(X \in \mathcal{B}) = K \int_{\mathcal{B}} f_Y(y) \mathbf{1}_{\{y \in \mathcal{A}\}} dy = K \mathbb{P}(Y \in \mathcal{A} \text{ and } Y \in \mathcal{B}) = \mathbb{P}(Y \in \mathcal{B} | Y \in \mathcal{A})$.

¹¹The coordinates are independent and uniform: generate two independent samples $U, V \sim \text{Unif}(0, 1)$; the sample is $((1 - U)x_{\min} + Ux_{\max}, (1 - V)y_{\min} + Vy_{\max})$.

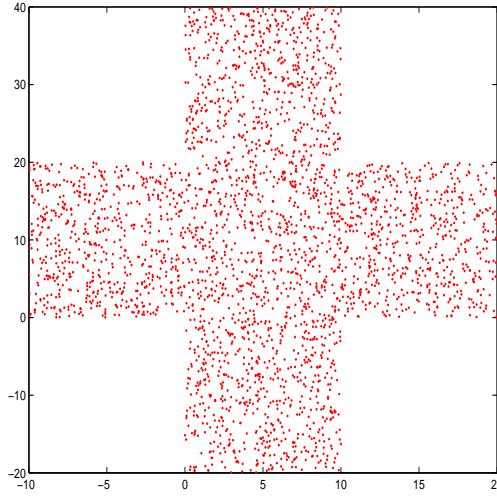


Figure 6.9: 1000 independent samples of the uniform distribution over \mathcal{A} = the interior of the cross. Samples are obtained by generating uniform samples in the bounding rectangle and rejecting those samples that do not fall in \mathcal{A} .

THEOREM 6.3 (Rejection Sampling for Distribution with Density). *Consider two random variables X, Y with values in the same space, that both have densities. Assume that:*

- we know a method to draw a sample of X
- the density of Y is known up to a normalization constant K : $f_Y(y) = K f_Y^n(y)$, where f_Y^n is a known function
- there exist some $c > 0$ such that

$$\frac{f_Y^n(x)}{f_X(x)} \leq c$$

A sample of Y is obtained by the following algorithm:

```

do
    draw independent samples of  $X$  and  $U$ , where  $U \sim \text{Unif}(0, c)$ 
until  $U \leq \frac{f_Y^n(X)}{f_X(X)}$ 
return( $X$ )

```

The expected number of iterations of the algorithm is $\frac{c}{K}$.

A frequent use of Theorem 6.3 is as follows.

ARBITRARY DISTRIBUTION WITH DENSITY Assume that we want a sample of Y , which takes values in the bounded interval $[a, b]$ and has a density $f_Y = K f_Y^n(y)$. Assume that f_Y^n (non normalized density) can easily be computed, but not the normalization constant K which is unknown. Also assume that we know an upper bound M on f_Y^n .

We take X uniformly distributed over $[a, b]$ and obtain the sampling method:

```

do
    draw  $X \sim \text{Unif}(a, b)$  and  $U \sim \text{Unif}(0, M)$ 

```

```

until  $U \leq f_Y^n(X)$ 
return( $X$ )

```

Note that we do *not* need to know the multiplicative constant K . For example, consider the distribution with density

$$f_Y(y) = K \frac{\sin^2(y)}{y^2} \mathbf{1}_{\{-a \leq y \leq a\}} \quad (6.13)$$

K is hard to compute, but a bound M on f_Y^n is easy to find ($M = 1$) (Figure 6.10).

EXAMPLE 6.14: A STOCHASTIC GEOMETRY EXAMPLE. We want to sample the random vector (X_1, X_2) that takes values in the rectangle $[0, 1] \times [0, 1]$ and whose distribution has a density proportional to $|X_1 - X_2|$. We take $f_X =$ the uniform density over $[0, 1] \times [0, 1]$ and $f_Y^n(x_1, x_2) = |x_1 - x_2|$. An upper bound on the ratio $\frac{f_Y^n(x_1, x_2)}{f_X(x_1, x_2)}$ is 1. The sampling algorithm is thus:

```

do
  draw  $X_1, X_2$  and  $U \sim \text{Unif}(0, 1)$ 
until  $U \leq |X_1 - X_2|$ 
return( $X_1, X_2$ )

```

Figure 6.10 shows an example. Note that there is no need to know the normalizing constant to apply the sampling algorithm.

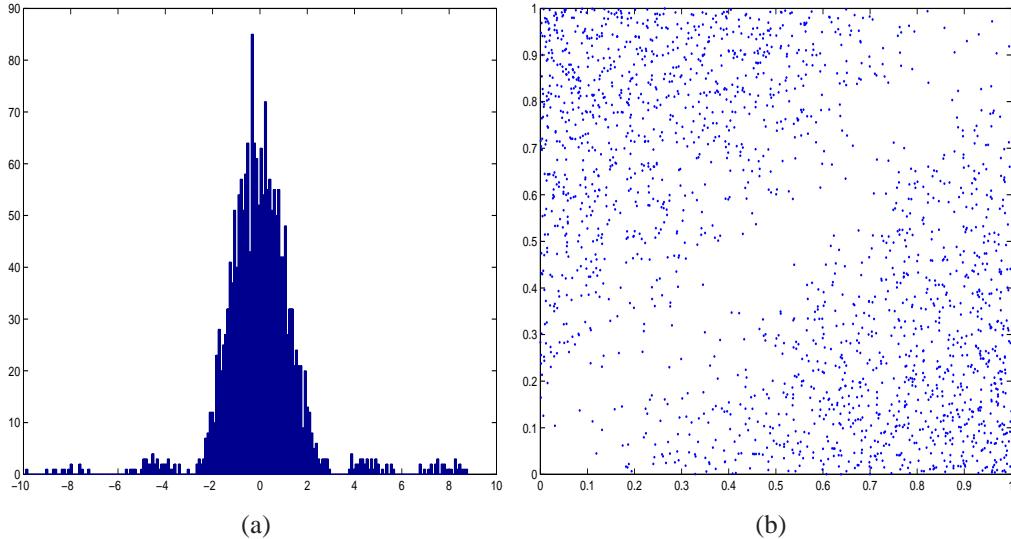


Figure 6.10: (a) Empirical histogram (bin size = 10) of 2000 samples of the distribution with density $f_X(x)$ proportional to $\frac{\sin^2(x)}{x^2} \mathbf{1}_{\{-a \leq x \leq a\}}$ with $a = 10$. (b) 2000 independent samples of the distribution on the rectangle with density $f_{X_1, X_2}(x_1, x_2)$ proportional to $|x_1 - x_2|$.

6.6.3 AD-HOC METHODS

The methods of inversion and rejection sampling may be improved in some special cases. We mention in detail the case of the normal distribution, which is important to optimize because of its

frequent use.

SAMPLING A NORMAL RANDOM VARIABLE. The method of inversion cannot be directly used, as the CDF is hard to compute. An alternative is based on the method of [change of variables](#), as given in the next proposition, the proof of which is by direct calculus.

PROPOSITION 6.1. *Let (X, Y) be independent, standard normal random variables. Let*

$$\begin{cases} R = \sqrt{X^2 + Y^2} \\ \Theta = \arg(X + jY) \end{cases}$$

R and Θ are independent, R has a Rayleigh distribution (i.e is positive with density $re^{-\frac{r^2}{2}}$) and Θ is uniformly distributed on $[0, 2\pi]$.

The CDF of the Rayleigh distribution can easily be inverted: $F(r) = \mathbb{P}(R \leq r) = 1 - e^{-r^2/2}$ and $F^{-1}(p) = \sqrt{-2 \ln(1-p)}$. A sampling method for a couple of two independent standard normal variables is thus ([Box-Müller method](#)):

```
draw U ~Unif(0, 1) and Θ ~Unif(0, 2π)
R = √−2 ln(U)
X = R cos(Θ), Y = R sin(Θ)
return(X, Y)
```

CORRELATED NORMAL RANDOM VECTORS. We want to sample (X_1, \dots, X_n) as a normal random vector with zero mean and covariance matrix Ω (see Section C.2). If the covariance matrix is diagonal (i.e. $\Omega_{i,j} = 0$ for $i \neq j$) then the X_i s are independent and we can sample them one by one (or better, two by two). We are interested here in the case where there is some correlation.

The method we show here is again based on a change of variable. There exists always a change of basis in \mathbb{R}^n such that, in the new basis, the random vector has a diagonal covariance matrix. In fact, there are many such bases (one of them is orthonormal and can be obtained by diagonalisation of Ω , but is much more expensive than the method we discuss next). An inexpensive and stable algorithm to obtain one such basis is called Choleski's factorization method. It finds a triangular matrix L such that $\Omega = LL^T$. Let Y be a standard normal vector (i.e. an i.i.d. sequence of n standard normal random variables). Let $X = LY$. The covariance matrix of X is

$$\mathbb{E}(XX^T) = \mathbb{E}(LY(LY)^T) = \mathbb{E}(L(YY^T)L^T) = L\mathbb{E}(YY^T)L^T = LL^T = \Omega$$

Thus a sample of X can be obtained by sampling Y first and computing LY . Figure 6.6.3 shows an example.

There are many ways to optimize the generation of samples. Good references are [108] and [90].

6.7 IMPORTANCE SAMPLING

6.7.1 MOTIVATION

Sometimes we want to estimate by simulation the probability of a [rare event](#), for example, a failure probability or a bit error rate. In such cases, straightforward Monte Carlo simulation is not

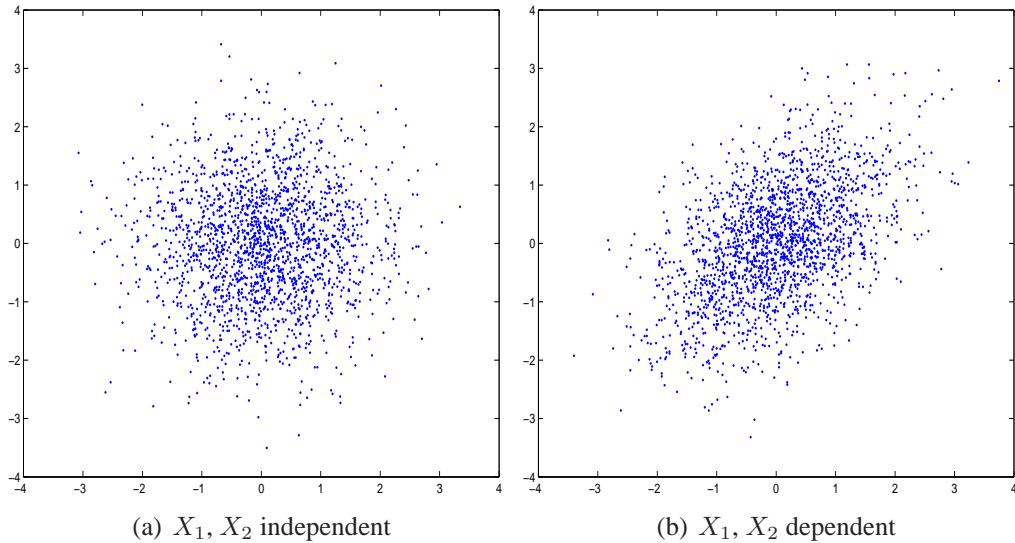


Figure 6.11: 1000 independent samples of the normal vector X_1, X_2 with 0 mean and covariance $\Omega_{1,1} = \sigma_1^2 = 1$, $\Omega_{2,2} = \sigma_2^2 = 1$ and $\Omega_{1,2} = \Omega_{2,1} = 0$ (left), $\Omega_{1,2} = \Omega_{2,1} = 1/2$ (right). The right sample is obtained by the transformation $X = LY$ with $Y i.i.d. \sim N_{0,1}$ and $L = (1, 0; 1/2, \sqrt{3}/2)$.

efficient, as it requires a large number of runs to obtain a reliable estimate; for example, assume the failure probability to be estimated is 10^{-5} . With R independent replications of a Monte Carlo simulation, the expected number of runs which produce one failure is $N/10^{-5}$, so we need 10^7 runs to be able to observe 100 failures. In fact, we need order of $4 \cdot 10^7$ runs in order to obtain a 95% confidence interval with a margin on the failure probability of the order of 10%.

Assume we want to estimate a failure probability p , by doing R replications. A naive Monte Carlo estimate is $\hat{p} = \frac{N}{R}$ where N is the number of runs which produce a failure. A $1 - \alpha$ confidence interval for p has a length of η times the standard deviation of \hat{p} , where $N_{0,1}(\eta) = 1 - \frac{\alpha}{2}$. The relative accuracy of the estimator is ηc , where c is the coefficient of variation of \hat{p} . Now $c = \frac{\sqrt{p(1-p)/R}}{p} = \frac{\sqrt{1-p}}{\sqrt{Rp}} \approx \frac{1}{\sqrt{Rp}}$, where the approximation is for very small p . Assume we want a relative accuracy on our estimation of p equal to β . We should take $\frac{\eta}{\sqrt{Rp}} = \beta$, i.e.

$$R = \frac{\eta^2}{\beta^2 p} \quad (6.14)$$

For example, for $\alpha = 0.05$ we have $\eta = 1.96$ and thus for $\beta = 0.10$ we should take $R \approx \frac{400}{p}$.

6.7.2 THE IMPORTANCE SAMPLING FRAMEWORK

Importance sampling is a method that can be used to reduce the number of required runs in a Monte Carlo simulation, when the events of interest (e.g. the failures) are rare. The idea is to modify the distribution of the random variable to be simulated, in a way such that the impact of the modification can be exactly compensated, and such that, for the modified random variable, the events of interest are not rare.

Formally, assume we simulate a random variable X in \mathbb{R}^d , with PDF $f_X()$. Our goal is to estimate $p = \mathbb{E}(\phi(X))$, where ϕ is the metric of interest. Frequently, $\phi(x)$ is the indicator function, equal

to 1 if the value x corresponds to a failure of the system, and 0 otherwise. We replace the original

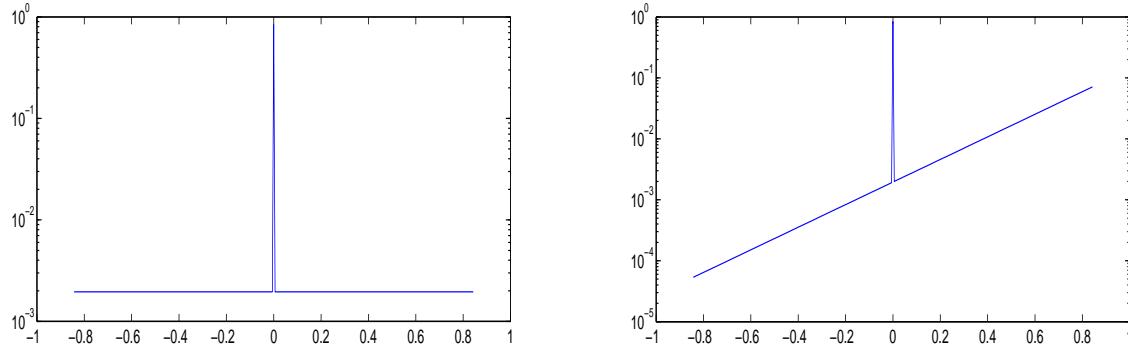


Figure 6.12: First Panel: log of the PDF of X_1 in Example 6.15. Second panel: log of the PDF of the twisted distribution (i.e. distribution of \hat{X}_1) when $\theta = 4.2$.

PDF $f_X()$ by another one, $f_{\hat{X}}()$, called the PDF of the *importance sampling distribution*, on the same space \mathbb{R}^d . We assume that

$$\text{if } f_X(x) > 0 \text{ then } f_{\hat{X}}(x) > 0$$

i.e. the support of the importance sampling distribution contains that of the original one. For x in the support of $f_X()$, define the *weighting function*

$$w(x) = \frac{f_X(x)}{f_{\hat{X}}(x)} \quad (6.15)$$

We assume that $w(x)$ can easily be evaluated. Let \hat{X} be a random variable whose distribution is the importance sampling distribution. We also assume that it is easy to draw a sample of \hat{X} .

It comes immediately that

$$\mathbb{E}(\phi(\hat{X})w(\hat{X})) = \mathbb{E}(\phi(X)) = p \quad (6.16)$$

which is the fundamental equation of importance sampling. An estimate of p is thus given by

$$p_{est} = \frac{1}{R} \sum_{r=1}^R \phi(\hat{X}_r)w(\hat{X}_r) \quad (6.17)$$

where \hat{X}_r are R independent replicates of \hat{X} .

Why would this be easier than the original problem ? Assume we have found a sampling distribution for which the events of interest are not rare. It follows that $w(x)$ is very small, but $\phi(\hat{X})$ is not. So the events $\phi(\hat{X}) = 1$ are not rare, and can be reproduced many times in a short simulation. The final result, p is small because we weight the outputs $\phi(\hat{X})$ by small numbers.

EXAMPLE 6.15: BIT ERROR RATE AND EXPONENTIAL TWISTING. The Bit Error Rate on a communication channel with impulsive interferers can be expressed as [68]:

$$p = \mathbb{P}(X_0 + X_1 + \dots + X_d > a) \quad (6.18)$$

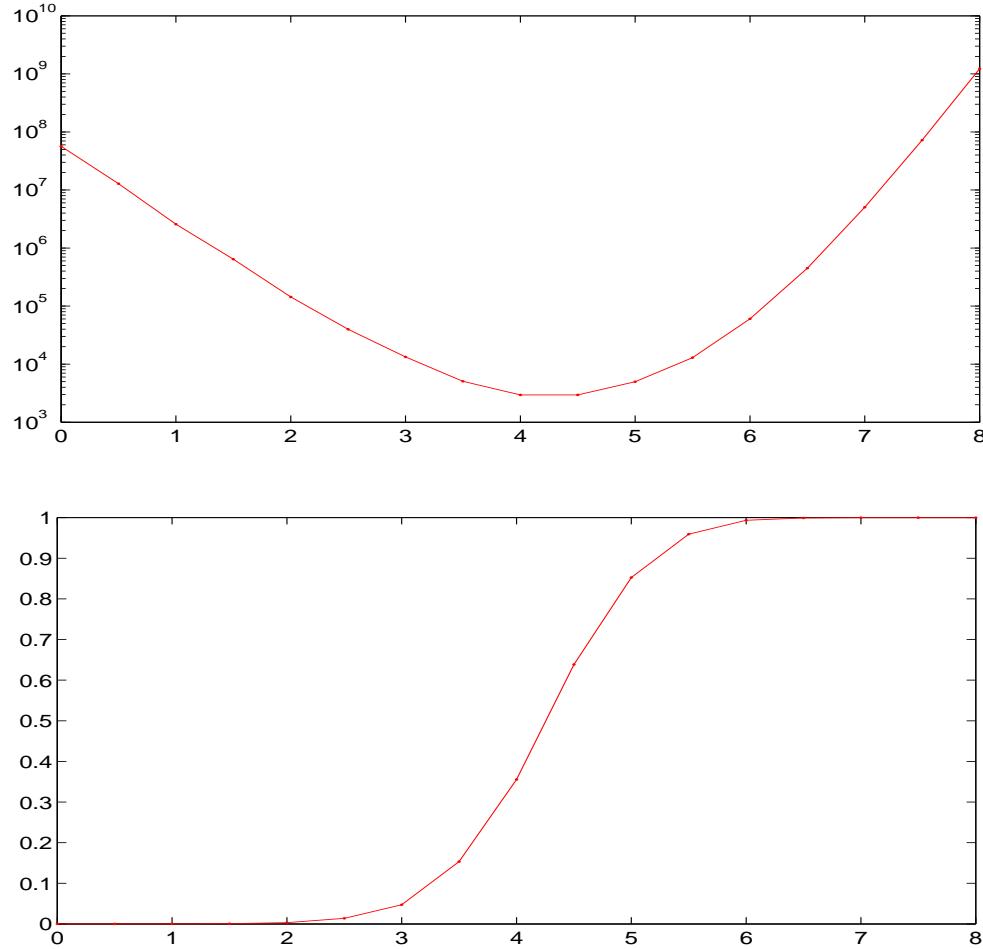


Figure 6.13: First panel: Required number of simulation runs to estimate the bit error rate in Example 6.15 with 10% of relative accuracy, using an importance sampling distribution with parameter θ (on x -axis). All simulations estimate give the same value estimated value of $p = 0.645E - 05$, but the required number of simulation runs R is proportional to the variance. Second panel: all simulations estimate p by the formula $p = \mathbb{E}(\phi(\hat{X})w(\hat{X}))$; the panel shows $\mathbb{E}(\phi(\hat{X}))$, i.e. the probability that there is a bit error when \hat{X} is drawn from the importance sampling distribution with parameter θ . For $\theta = 0$ we have the true value $p = 0.645E - 05$. The smallest number of runs, i.e. the smallest variance, is obtained when $\mathbb{E}(\phi(\hat{X})) \approx 0.5$.

where $X_0 \sim N_{0,\sigma^2}$ is thermal noise and X_j , $j = 1, \dots, d$ represents impulsive interferers. The distribution of X_j is discrete, with support in $\{\pm x_{j,k}, k = 1, \dots, n\} \cup \{0\}$ and:

$$\begin{aligned}\mathbb{P}(X_j = \pm x_{j,k}) &= q \\ \mathbb{P}(X_j = 0) &= 1 - 2nq\end{aligned}$$

where $n = 40$, $q = \frac{1}{512}$ and the array $\{\pm x_{j,k}, k = 1, \dots, n\}$ are given numerically by channel estimation (Table 6.2 shows a few examples, for $d = 9$). The variables $X_j, j = 0, \dots, d$ are independent. For large values of d , we could approximate p by a gaussian approximation, but it can easily be verified that for d of the order of 10 or less this does not hold [68].

A direct Monte Carlo estimation (without importance sampling) gives the following results (R is the number of Monte Carlo runs required to reach 10% accuracy with confidence 95%, as of Eq.(6.14)):

k	j=1	j=2	j=3	j=4	j=5	j=6	j=7	j=8	j=9
1	0.4706	0.0547	0.0806	0.0944	0.4884	0.3324	0.4822	0.3794	0.2047
2	0.8429	0.0683	0.2684	0.2608	0.0630	0.1022	0.1224	0.0100	0.0282
...

Table 6.2: Sample Numerical values of $x_{j,k}$ for Example 6.15; the complete list of values is available on the web site of the book.

σ	a	BER estimate	R
0.1	3	$(6.45 \pm 0.6) \times 10^{-6}$	6.2×10^7

Now we apply importance sampling in order to reduce the required number of simulation runs R . We consider importance sampling distributions derived by *exponential twisting*, i.e. we define the distribution of \hat{X}_j , $j = 0, \dots, d$ by:

$$\begin{cases} \hat{X}_j \text{ has the same support as } X_j \\ \mathbb{P}(\hat{X}_j = x) = \eta_j(\theta) e^{\theta x} \mathbb{P}(X_j = x) \end{cases}$$

where $\eta_j(\theta)$ is a normalizing constant. This gives

$$\begin{aligned} \mathbb{P}(X_j = -x_{j,k}) &= \eta_j(\theta) q e^{-\theta x_{j,k}} \\ \mathbb{P}(X_j = x_{j,k}) &= \eta_j(\theta) q e^{\theta x_{j,k}} \\ \mathbb{P}(X_j = 0) &= \eta_j(\theta)(1 - 2nq) \\ \eta_j(\theta)^{-1} &= q \sum_{k=1}^n \left(e^{-\theta x_{j,k}} + e^{\theta x_{j,k}} \right) + 1 - 2nq \end{aligned}$$

Similarly, the distribution of the gaussian noise \hat{X}_0 is obtained by multiplying the PDF of the standard normal distribution by $e^{\theta x}$ and normalizing:

$$\begin{aligned} f_{\hat{X}_0}(x) &= \eta_0 \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}} e^{\theta x} \\ &= \eta_1 e^{\frac{\theta^2\sigma^2}{2}} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\theta\sigma^2)^2}{2\sigma^2}} \end{aligned}$$

Thus $\eta_0 = e^{-\frac{\theta^2\sigma^2}{2}}$ and \hat{X}_0 is normally distributed with same variance as X_0 but with mean $\sigma^2\theta$ instead of 0. Note that for $\theta > 0$, \hat{X}_j is more likely to take large values than X_j . The weighting function is

$$w(x_0, \dots, x_d) = e^{-\theta \sum_{j=0}^d x_j} \frac{1}{\prod_{j=0}^d \eta_j} \quad (6.19)$$

We perform R Monte Carlo simulations with \hat{X}_j in lieu of X_j ; the estimate of p is

$$p_{est} = \frac{1}{R} \sum_{r=1}^R w\left(\hat{X}_0^r, \dots, \hat{X}_d^r\right) \mathbf{1}_{\{\hat{X}_0^r + \dots + \hat{X}_d^r > a\}} \quad (6.20)$$

Note that $\theta = 0$ corresponds to direct Monte Carlo (without importance sampling). All simulations give the same estimated value $p \approx 0.645E - 05$, but the required number of simulation runs required to reach the same accuracy varies by more than 3 orders of magnitude.

6.7.3 SELECTING AN IMPORTANCE SAMPLING DISTRIBUTION

The previous example shows that importance sampling can dramatically reduce the number of Monte Carlo runs for estimating rare events, but also that it is important to carefully choose the importance sampling distribution, as a wrong choice may give no improvement (or might even be worse).

A first observation can be derived from the analysis of Figure 6.13: the best choice is when the probability of the event of interest, under the importance sampling distribution, is close to 0.5 (i.e. $\mathbb{E}(\phi(\hat{X})) \approx 0.5$). Note that, perhaps contrary to intuition, choosing $\mathbb{E}(\phi(\hat{X})) \approx 1$ is a very bad choice. In other words, we need to make the events of interest not so rare, but not so certain either. This can be explained as follows. If we take $\mathbb{E}(\phi(\hat{X})) \approx 1$, the simulator has a hard time producing samples where the event of interest does *not* occur, which is as bad as the initial problem.

A second observation is that we can evaluate the efficiency of an importance sampling estimator of p by its variance

$$\hat{v} = \text{var}(\phi(\hat{X})w(\hat{X})) = \mathbb{E}(\phi(\hat{X})^2w(\hat{X})^2) - p^2$$

Assume that we want a $1 - \alpha$ confidence interval of relative accuracy β . By a similar reasoning as in Eq.(6.14), the required number of Monte Carlo estimates is

$$R = \hat{v} \frac{\eta^2}{\beta^2 p^2} \quad (6.21)$$

Thus, it is proportional to \hat{v} . In the formula, η is defined by $N_{0,1}(\eta) = 1 - \frac{\alpha}{2}$; for example, with $\alpha = 0.05$, $\beta = 0.1$, we need $R \approx 400\hat{v}/p^2$.

Therefore, the problem is to find a sampling distribution which minimizes \hat{v} , or, equivalently, $\mathbb{E}(\phi(\hat{X})^2w(\hat{X})^2)$. The theoretical solution can be obtained by calculus of variation; it can be shown that the optimal sampling distribution $f_{\hat{X}}(x)$ is proportional to $|\phi(x)| f_X(x)$. In practice, however, it is impossible to compute, since we assume in the first place that it is hard to compute $\phi(x)$.

In Algorithm 4 we give a heuristic method, which combines these two observations. Assume we have at our disposal a family of candidate importance sampling distributions, indexed by a parameter¹² $\theta \in \Theta$. The function `varEst()` estimates, by Monte Carlo, whether a given θ satisfies $\mathbb{E}(\phi(\hat{X})) \approx 0.5$; if so, it returns an estimate of $\mathbb{E}(\phi(\hat{X})^2w(\hat{X})^2)$, else it returns ∞ . Note that the number of Monte Carlo runs required by `varEst()` is small, since we are interested only in returning results in the cases where $\mathbb{E}(\phi(\hat{X})) \approx 0.5$, i.e. we are not in the case of rare events.

The first part of the algorithm (line 8) consists in selecting one value of θ which minimizes $\text{varEst}(\theta)$. This can be done by random exploration of the set Θ , or by any heuristic optimization method (such as Matlab's `fminsearch`).

¹²For simplicity, we do not show the dependency on θ in expressions such as $\mathbb{E}(\phi(\hat{X}))$, which could be more accurately described as $\mathbb{E}(\phi(\hat{X}) | \theta)$.

Algorithm 4 Determination of a good Importance Sampling distribution. We want to estimate $p = \mathbb{E}(\phi(X))$, where X a random variable with values in \mathbb{R}^d and $\phi(x) \in [0; 1]$; \hat{X} is drawn from the importance sampling distribution with parameter θ ; $w()$ is the weighting function (Eq.(6.15)).

```

1: function MAIN
2:    $\eta = 1.96$ ;  $\beta = 0.1$ ; pCountMin= 10;            $\triangleright \beta$  is the relative accuracy of the final result
3:   GLOBAL  $R_0 = 2\frac{\eta^2}{\beta^2}$ ;                       $\triangleright$  Typical number of iterations
4:    $R_{\max} = 1E + 9$ ;                                 $\triangleright R_0$  chosen by Eq.(6.14) with  $p = 0.5$ 
5:    $c = \frac{\beta^2}{\eta^2}$ ;                             $\triangleright$  Maximum number of iterations
6:
7:
8:   Find  $\theta_0 \in \Theta$  which minimizes varest( $\theta$ );
9:
10:  pCount0= 0; pCount= 0;  $m_2 = 0$ ;
11:  for  $r = 1 : R_{\max}$  do
12:    draw a sample  $x$  of  $\hat{X}$  using parameter  $\theta_0$ ;
13:    pCount0=pCount0+ $\phi(x)$ ;
14:    pCount=pCount+ $\phi(x)w(x)$ ;
15:     $m_2 = m_2 + (\phi(x)w(x))^2$ ;
16:    if  $r \geq R_0$  and  $pCount < pCountMin < r - pCountMin$  then
17:       $p = \frac{pCount}{r}$ ;
18:       $v = \frac{m_2}{r} - p^2$ ;
19:      if  $v \leq cp^2r$  then break
20:      end if
21:    end if
22:  end for
23:  return  $p, r$ 
24: end function

25:
26: function VAREST( $\theta$ )            $\triangleright$  Test if  $\mathbb{E}\left(\phi(\hat{X})\right) \approx 0.5$  and if so estimate  $\mathbb{E}\left(\phi(\hat{X})^2w(\hat{X})^2\right)$ 
27:   CONST  $\hat{p}_{\min} = 0.3$ ,  $\hat{p}_{\max} = 0.7$ ;
28:   GLOBAL  $R_0$ ;
29:    $\hat{p} = 0$ ;  $m_2 = 0$ ;
30:   for  $r = 1 : R_0$  do
31:     draw a sample  $x$  of  $\hat{X}$  using parameter  $\theta$ ;
32:      $\hat{p} = \hat{p} + \phi(x)$ ;
33:      $m_2 = m_2 + (\phi(x)w(x))^2$ ;
34:   end for
35:    $\hat{p} = \frac{\hat{p}}{R}$ ;
36:    $m_2 = \frac{m_2}{R}$ ;
37:   if  $\hat{p}_{\min} \leq \hat{p} \leq \hat{p}_{\max}$  then
38:     return  $m_2$ ;
39:   else
40:     return  $\infty$ ;
41:   end if
42: end function
```

The second part (lines 10 to 23) uses the best θ and importance sampling as explained earlier. The algorithms performs as many Monte Carlo samples as required to obtained a given accuracy level, using Eq.(6.21) (line 19).

EXAMPLE 6.16: BIT ERROR RATE, RE-VISITED. We can apply Algorithm 4 directly. With the same notation as in Example 6.15, an estimate of \hat{v} , the variance of the importance sampling estimator, is

$$\hat{v}_{est} = \frac{1}{R} \sum_{r=1}^R w \left(\hat{X}_0^r, \dots, \hat{X}_d^r \right)^2 \mathbf{1}_{\{\hat{X}_0^r + \dots + \hat{X}_d^r > a\}} - p_{est}^2 \quad (6.22)$$

We computed \hat{v}_{est} for different values of θ ; Figure 6.13 shows the corresponding values of the required number of simulation runs R (to reach 10% accuracy with confidence 95%), as given by Eq.(6.21)).

Alternatively, one could use the following optimization. We can avoid the simulation of a normal random variable by noticing that Eq.(6.18) can be replaced by

$$\begin{aligned} p &:= \mathbb{P}(X_0 + X_1 + \dots + X_d > a) \\ &= \mathbb{P}(X_0 > a - (X_1 + \dots + X_d)) \\ &= \mathbb{E}(\mathbb{P}(X_0 > a - (X_1 + \dots + X_d) | X_1, \dots, X_d)) \\ &= \mathbb{E}\left(1 - N_{0,1}\left(\frac{a - (X_1 + \dots + X_d)}{\sigma}\right)\right) := \mathbb{E}(\phi(X_1 + \dots + X_d)) \end{aligned}$$

where, as usual, $N_{0,1}()$ is the cdf of the standard normal distribution and $\phi(x) = 1 - N_{0,1}(x)$. So the problem becomes to compute $\mathbb{E}(\phi(X_1 + \dots + X_d))$.

We applied Algorithm 4 with the same numerical values as in Example 6.15 and with exponential twisting. Note the difference with Example 6.15: we modify the distributions of $X_1 \dots X_d$ but not of the normal variable X_0 . The best θ is now for $\mathbb{E}(\phi(\hat{X})) \approx 0.55$ (instead of 0.5) and the number of simulation runs required to achieve the same level of accuracy is slightly reduced.

In the above example we restricted the choice of the importance sampling distribution to an exponential twist, with the same parameter θ for all random variables $X_1 \dots X_d$. There are of course many possible variants; for example, one might use a different θ for each X_j , or one can use different methods of twisting the distribution (for example re-scaling); note however that the complexity of the choice of an importance sampling distribution should not outweigh its final benefits, so in general, we should aim at simple solutions. The interested reader will find a general discussion and overview of other methods in [98].

6.8 PROOFS

THEOREM 6.1

The pseudo-inverse has the property that [55, Thm 3.1.2]

$$F(x) \geq p \Leftrightarrow F^{-1}(p) \leq x$$

Let $Y = F^{-1}(U)$. Thus $\mathbb{P}(Y \leq y) = \mathbb{P}(F(y) \leq U) = F(y)$ and the CDF of Y is $F()$.

THEOREM 6.2

Let N be the (random) number of iterations of the algorithm, and let $(\tilde{X}_k, \tilde{Y}_k)$ be the sample drawn at the k th iteration. (These samples are independent, but in general, \tilde{X}_k and \tilde{Y}_k are *not* independent). Let $\theta = \mathbb{P}(\tilde{Y} \in \mathcal{A})$. We assume $\theta > 0$ otherwise the conditional distribution of \tilde{X} is not defined. The output of the algorithm is $X = \tilde{X}_N$.

For some arbitrary measurable \mathcal{B} in S , we compute $\mathbb{P}(\tilde{X}_N \in \mathcal{B})$:

$$\begin{aligned}\mathbb{P}(\tilde{X}_N \in \mathcal{B}) &= \sum_{k \geq 1} \mathbb{P}(\tilde{X}_k \in \mathcal{B} \text{ and } N = k) \\ &= \sum_{k \geq 1} \mathbb{P}(\tilde{X}_k \in \mathcal{B} \text{ and } \tilde{Y}_1 \notin \mathcal{A}, \dots, \tilde{Y}_{k-1} \notin \mathcal{A}, \tilde{Y}_k \in \mathcal{A}) \\ &= \sum_{k \geq 1} \mathbb{P}(\tilde{X}_k \in \mathcal{B} \text{ and } \tilde{Y}_k \in \mathcal{A}) \mathbb{P}(\tilde{Y}_1 \notin \mathcal{A}) \dots \mathbb{P}(\tilde{Y}_{k-1} \notin \mathcal{A}) \\ &= \sum_{k \geq 1} \mathbb{P}(\tilde{X}_k \in \mathcal{B} | \tilde{Y}_k \in \mathcal{A}) \theta(1 - \theta)^{k-1} \\ &= \sum_{k \geq 1} \mathbb{P}(\tilde{X}_1 \in \mathcal{B} | \tilde{Y}_1 \in \mathcal{A}) \theta(1 - \theta)^{k-1} \\ &= \mathbb{P}(\tilde{X}_1 \in \mathcal{B} | \tilde{Y}_1 \in \mathcal{A}) \sum_{k \geq 1} \theta(1 - \theta)^{k-1} \\ &= \mathbb{P}(\tilde{X}_1 \in \mathcal{B} | \tilde{Y}_1 \in \mathcal{A})\end{aligned}$$

The second equality is by definition of N . The third is by the independence of $(\tilde{X}_k, \tilde{Y}_k)$ and $(\tilde{X}_{k'}, \tilde{Y}_{k'})$ for $k \neq k'$. The last equality is because $\theta > 0$. This shows that the distribution of X is as required.

$N - 1$ is geometric with parameter θ thus the expectation of N is $1/\theta$.

THEOREM 6.3

Apply Theorem 6.2 with $\tilde{X} = X$ and $\tilde{Y} = (X, U)$. All we need to show is that the conditional density of X given that $U \leq \frac{f_Y^n(X)}{f_X(X)}$ is f_Y .

To this end, pick some arbitrary function ϕ . We have

$$\begin{aligned}\mathbb{E}(\phi(X) | U \leq \frac{f_Y^n(X)}{f_X(X)}) &= K_1 \mathbb{E}(\phi(X) \mathbf{1}_{\{U \leq \frac{f_Y^n(X)}{f_X(X)}\}}) \\ &= K_1 \int \mathbb{E}(\phi(x) \mathbf{1}_{\{U \leq \frac{f_Y^n(x)}{f_X(x)}\}} | X = x) f_X(x) dx \\ &= K_1 \int \phi(x) \frac{f_Y^n(x)}{f_X(x)} f_X(x) dx \\ &= \frac{K_1}{K} \int \phi(x) f_Y(x) dx = \frac{K_1}{K} \mathbb{E}(\phi(Y))\end{aligned}$$

where K_1 is some constant. This is true for all ϕ thus, necessarily, $K_1/K = 1$ (take $\phi = 1$).

6.9 REVIEW

QUESTION 6.9.1. *How do you generate a sample of a real random variable with PDF $f()$ and CDF $F()$?¹³*

QUESTION 6.9.2. *Why do we care about stationarity ?¹⁴*

QUESTION 6.9.3. *What is rejection sampling ?¹⁵*

QUESTION 6.9.4. *How do you generate a sample of a discrete random variable ?¹⁶*

QUESTION 6.9.5. *What is importance sampling?¹⁷*

QUESTION 6.9.6. *Why do we need to run independent replications of a simulation ? How are they obtained ?¹⁸*

QUESTION 6.9.7. *Consider the sampling method: Draw $\text{COIN}(p)$ until it returns 0. The value of the sample N is the number of iterations. Which distribution is that a sample from ? Is this a good method ?¹⁹*

QUESTION 6.9.8. *If we do a direct Monte Carlo simulation (i.e without importance sampling) of a rare event, the theorem for confidence intervals of success probabilities (Theorem 2.4) gives a confidence interval. So why do we need importance sampling ?²⁰*

¹³In many cases matlab does it. If not, if $F()$ is easily invertible, use CDF inversion. Else, if $f()$ has a bounded support, use rejection sampling.

¹⁴Non terminating simulations depend on the initial conditions, and on the length of the simulation. If the simulator has a stationary regime, we can eliminate the impact of the simulation length (in simulated time) and of the initial conditions.

¹⁵Drawing independent samples of an object with some probability distribution $p(.)$, some condition C is met. The result is a sample of the conditional probability $p(.|C)$.

¹⁶With the method of CDF inversion. Let p_k be the probability of outcome k , $k = 1 \dots n$ and $F_k = p_1 + \dots + p_k$ (with $F_0 = 0$). Draw $U \sim \text{Unif}(0, 1)$; if $F_k \leq U < F_{k+1}$ then let $N = k$.

¹⁷A method for computing probabilities of rare events. It consists in changing the initial probability distribution in order to make rare events less rare (but not certain).

¹⁸To obtain confidence intervals. By running multiple instances of the simulation program; if done sequentially, the seed of the random generator can be carried over from one run to the next. If replications are done in parallel on several machines, the seeds should be chosen independently by truly random sources.

¹⁹The distribution of N is geometric with $\theta = 1 - p$, so this method does produce a sample from a geometric distribution. However it draws in average $\frac{1}{\theta}$ random numbers from the generator, and the random number generator is usually considered an expensive computation compared to a floating point operation. If θ is small, the procedure in Example 6.12 (by CDF inversion) is much more efficient.

²⁰Assume we simulate a rare event, without importance sampling, and find 0 success out of R Monte Carlo replicates. Theorem 2.4 gives a confidence interval for probability of success equal to $[0, \frac{3.869}{R}]$ at confidence level 0.95; for example, if $R = 10^4$, we can say that $p < 4 \cdot 10^{-4}$. Importance sampling will give more, it will provide an estimate of, for example $5.33 \cdot 10^{-5} \pm 0.4 \cdot 10^{-5}$. In many cases (for example when computing p -values of tests), all we care about is whether p is smaller than some threshold; then we may not need importance sampling. Importance sampling is useful if we need the magnitude of the rare event.

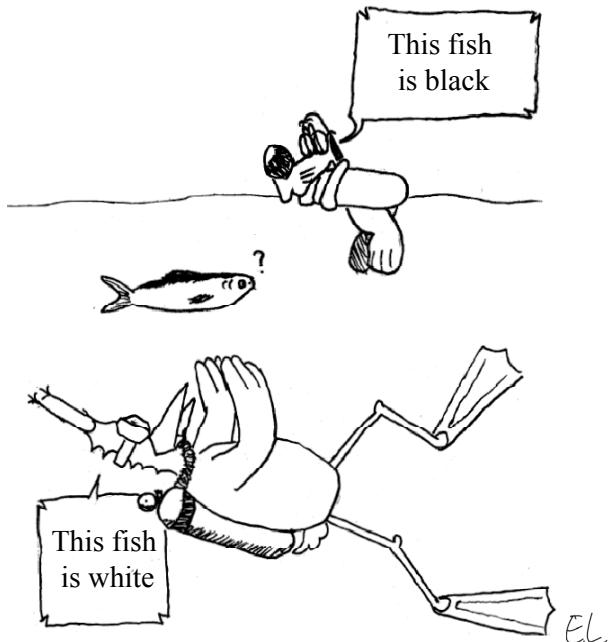
CHAPTER 7

PALM CALCULUS, OR THE IMPORTANCE OF THE VIEWPOINT

When computing or measuring a performance metric (as defined in Chapter 1), one should specify which observer's **viewpoint** is taken. For example, in a simulation study of an information server, one may be interested in the metric "worst case backlog", defined as the 95-percentile of the number of pending requests.

One way is to measure the queue of pending requests at request arrival epochs over a large number of arriving requests, and compute the 95-percentile of the resulting empirical distribution. An alternative is to measure the queue of pending requests at periodic intervals (say every second) over a long period of time and compute the 95-percentile of the resulting empirical distribution. The former method reflects the viewpoint of an arriving request, the latter of an observer at an arbitrary point in time. The former method evaluates the metric using a clock that ticks at every request arrival, whereas the latter uses a standard clock. Both methods will usually provide different values of the metric. Therefore, a metric definition should specify which clock, or viewpoint is used, and the choice should be relevant for the specific issues being addressed.

In Section 7.1, we give an intuitive definition of event clocks and of event versus time averages; we show that subtle, but possibly large, sampling biases are unavoidable. We also show how to use the large time heuristic to derive Palm calculus formulas, i.e. formulas that relate metrics obtained



with different clocks.

In the rest of the chapter we formally present Palm calculus, i.e. a formal treatment of these intuitive definitions and formulas. This is a branch of probability which is not well known, though it is quite important for any measurement or simulation study, and can be presented quite simply. In essence, the Palm probability of an event is the conditional probability, given that some specific point process has a point. Making sense of this is simple in discrete time, but very complex in continuous time, as is often the case in the theory of stochastic processes. We do not dwell on formal mathematical constructions, but we do give formulas and exact conditions under which they apply.

We introduce Feller's paradox, an apparent contradiction in waiting times that can explained by a difference in viewpoints. We give useful formulas such as the Rate Conservation Law and some of its many consequences such as Little's, Campbell's shot noise, Neveu's exchange and Palm's inversion formulas. We discuss simulations defined as stochastic recurrences, show how this can explain when simulations freeze and how to avoid transient removals at all (perfect simulation). Last, we give practical formulas for computing Palm probabilities with Markov models observed along a subchain, and use these to derive the PASTA property.

Contents

7.1 An Informal Introduction	197
7.1.1 Event versus Time Averages	197
7.1.2 The Large Time Heuristic	198
7.1.3 Two Event Clocks	200
7.1.4 Arbitrary Sampling Methods	201
7.2 Palm Calculus	204
7.2.1 Hypotheses	204
7.2.2 Definitions	204
7.2.3 Interpretation as Time and Event Averages	206
7.2.4 The Inversion and Intensity Formulas	207
7.3 Other Useful Palm Calculus Results	209
7.3.1 Residual Time and Feller's Paradox	209
7.3.2 The Rate Conservation Law and Little's Formula	212
7.3.3 Two Event Clocks	218
7.4 Simulation Defined as Stochastic Recurrence	219
7.4.1 Stochastic Recurrence, Modulated Process	219
7.4.2 Freezing Simulations	220
7.4.3 Perfect Simulation of Stochastic Recurrence	222
7.5 Application to Markov Chain Models and the PASTA Property	226
7.5.1 Embedded Sub-Chain	226
7.5.2 PASTA	228
7.6 Appendix: Quick Review of Markov Chains	230
7.6.1 Markov Chain in Discrete Time	230

7.6.2	Markov Chain in Continuous Time	231
7.6.3	Poisson and Bernoulli	232
7.7	Proofs	233
7.8	Review Questions	236

7.1 AN INFORMAL INTRODUCTION

In this section we give an intuitive treatment of event versus time averages and explain the use of event clocks. A formal treatment involves Palm calculus and is given in Section 7.2.

7.1.1 EVENT VERSUS TIME AVERAGES

Consider a discrete event simulation that runs for a long period of time, and let T_0, T_1, \dots, T_N be a sequence of **selected events**, for example, the request arrival times at an information server. Assume that we associate to the stream of selected events a clock that ticks at times T_0, T_1, \dots, T_N (the **event clock**). An **event average** statistic is any performance metric that is computed based on sampling the simulation state at times T_n , i.e. using the event clock. For example, the average queue length at the information server upon request arrival can be defined as

$$\bar{Q}^0 := \frac{1}{N+1} \sum_{n=0}^N Q(T_n^-)$$

(where $Q(t^-)$ is the queue size just before time t) and is an event average statistic.

In contrast, a **time average** statistic is obtained using the standard clock, assumed to have infinite accuracy (i.e. the standard clock ticks every δ time units, where δ is “infinitely small”). For example, the average queue length, defined by

$$\bar{Q} := \frac{1}{T_N - T_0} \int_{T_0}^{T_N} Q(s) ds$$

is a time average statistic.

In signal processing parlance, event averages correspond to **adaptive sampling**.

EXAMPLE 7.1: GATEKEEPER. A multitasking system receives jobs. Any arriving job is first processed by a “gatekeeper task”, which allocates the job to an available “application processor”. Due to power saving, the gatekeeper is available only at times, 0, 90, 100, 190, 200, ... (in milliseconds). For example a job that arrives at time 20ms is processed by the gatekeeper at time 90ms.

A job that is processed by the gatekeeper at times 0, 100, 200... is allocated to an application processor that has an execution time of 1000ms. In contrast, a job that is processed by the gatekeeper at times 90, 190, ... has an execution time of 5000ms (Figure 7.1). We assume there is neither queuing nor any additional delay. We are interested in the average job execution time, excluding the time to wait until the gatekeeper wakes up to process the job.

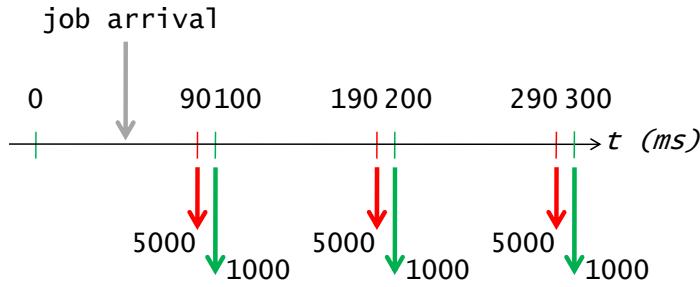


Figure 7.1: Gatekeeper: jobs are dispatched to a processor with processing time equal to 5000 or 1000 ms

The system designer thinks that the average job execution time is

$$W_s = \frac{1000 + 5000}{2} = 3000$$

since there are two application processors and this is the average of their execution times.

A customer may have a different viewpoint. If she sends a job to the system at a random instant, she will be allocated to an application processor depending on the time of arrival. She computes her performance metric assuming that she picks a millisecond at random uniformly in an interval $[0, T]$ where T is large, and obtains

$$W_c = \frac{90}{100} \times 5000 + \frac{10}{100} \times 1000 = 4600$$

The metric W_s is an event average; it can be measured using the event clock that ticks whenever the gatekeeper wakes up. The metric W_c is a time average; it can be measured using a clock that ticks every millisecond.

This example shows that event averages may be very different from time averages, in other words, **sampling bias** may be a real issue. Therefore it is necessary, when defining a metric, to specify which clock (i.e which viewpoint) is adopted. Further, one should discuss which viewpoint makes sense for the performance of interest. In the previous example, the time average viewpoint is a better metric as it directly reflects customer experience.

7.1.2 THE LARGE TIME HEURISTIC

Palm calculus is a set of formulas for relating event and time averages. They form the topic of the other sections in this chapter. However, it may be useful to know that most of these formulas can be derived heuristically using the **large time heuristic**, which can be described as follows.

1. formulate each performance metric as a long run ratio, as you would do if you would be evaluating the metric in a discrete event simulation;
2. take the formula for the time average viewpoint and break it down into pieces, where each piece corresponds to a time interval between two selected events;
3. compare the two formulations.

We explain it on the following example.

EXAMPLE 7.2: GATEKEEPER, CONTINUED. We can formalize Example 7.1 as follows. The two metrics are W_s (event average, system designer's viewpoint) and W_c (time average, customer's viewpoint).

1. In a simulation, we would estimate W_s and W_c as follows. Let T_0, T_1, \dots, T_N be the selected event times (times at which gatekeeper wakes up) and $S_n = T_n - T_{n-1}$ for $n = 1 \dots N$. Let X_n be the execution time for a job that is processed by the gatekeeper at time T_n

$$W_s := \frac{1}{N} \sum_{n=1}^N X_n \quad (7.1)$$

$$W_c := \frac{1}{T_N - T_0} \int_{T_0}^{T_N} X_{N^+(t)} dt \quad (7.2)$$

where $N^+(t)$ is the index of the next event clock tick after t , i.e. a job arriving at time t is processed by the gatekeeper at time T_n with $n = N^+(t)$.

2. We break the integral in Eq.(7.2) into pieces corresponding to the intervals $[T_{n-1}, T_n]$:

$$\begin{aligned} W_c &= \frac{1}{T_N - T_0} \sum_{n=1}^N \int_{T_{n-1}}^{T_n} X_{N^+(t)} dt = \frac{1}{T_N - T_0} \sum_{n=1}^N \int_{T_{n-1}}^{T_n} X_n dt \\ &= \frac{1}{T_N - T_0} \sum_{n=1}^N S_n X_n \end{aligned} \quad (7.3)$$

3. We now compare Eqs.(7.1) and (7.3). Define the sample average sleep time $\bar{S} := \frac{1}{N} \sum_{n=1}^N S_n$, the sample average execution time $\bar{X} := \frac{1}{N} \sum_{n=1}^N X_n$ and the sample cross-covariance

$$\hat{C} := \frac{1}{N} \sum_{n=1}^N (S_n - \bar{S})(X_n - \bar{X}) = \frac{1}{N} \sum_{n=1}^N S_n X_n - \bar{S} \bar{X}$$

We can re-write Eqs.(7.1) and (7.3) as:

$$\begin{aligned} W_s &= \bar{X} \\ W_c &= \frac{1}{N\bar{S}} \sum_{n=1}^N S_n X_n = \frac{1}{\bar{S}} (\hat{C} + \bar{S} \bar{X}) = \bar{X} + \frac{\hat{C}}{\bar{S}} \end{aligned}$$

In other words, we have shown that

$$W_c = W_s + \frac{\hat{C}}{\bar{S}} \quad (7.4)$$

Numerically, we find $\frac{\hat{C}}{\bar{S}} = 1600$ and Eq.(7.4) is verified.

Eq.(7.4) is our first example of Palm calculus formula; it relates the time average W_s to the event average W_c . Note that it holds quite generally, not just for the system in Example 7.1. We do not need any specific assumptions on the distribution of sleep or execution times, nor do we assume any form of independence. The only required assumption is that the metrics W_s and W_c can be

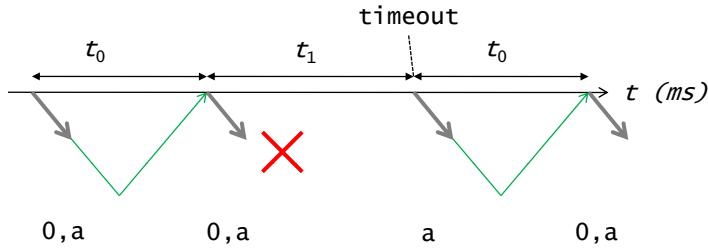


Figure 7.2: The Stop and Go protocol

measured using Eqs.(7.1) and (7.2). In the next section, we give a formal framework where such assumptions hold.

Eq.(7.4) shows that, for this example, the difference in viewpoints is attributed to the cross-covariance between sleep time and execution time. A positive [resp. negative] cross-covariance implies that the time average is larger [resp smaller] than the event average. In Example 7.1, the cross-covariance is positive and we do find a larger time average. If sleep time and execution times are non-correlated, the two viewpoints happen to produce the same metric.

7.1.3 TWO EVENT CLOCKS

There exist formulas not only for relating time and event averages, but also for relating different event averages (see Theorem 7.7). We show in this section how such formulas can be derived, using the following variant of the large time heuristic:

1. formulate each performance metric as a long run ratio, as you would do if you would be evaluating the metric in a discrete event simulation;
2. take the formula for one event average viewpoint and break it down into pieces, where each piece corresponds the time interval between two selected events of the second viewpoint;
3. compare the two formulations.

EXAMPLE 7.3: STOP AND GO PROTOCOL. A source sends packets to a destination. Error recovery is done by the stop and go protocol, as follows. When a packet is sent, a timer, with fixed value t_1 , is set. If the packet is acknowledged before t_1 , transmission is successful. Otherwise, the packet is re-transmitted. The packet plus acknowledgement transmission and processing have a constant duration equal to $t_0 < t_1$. The proportion of successful transmissions (fresh or not) is $1 - \alpha$. We assume that the source is greedy, i.e., always has a packet ready for transmission. Can we compute the throughput θ of this protocol without making any further assumptions ? The answer is yes, using the large time heuristic.

To this end, we compare the average transmission times sampled with the two different event clocks. The former (clock “a”) ticks at every transmission or re-transmission attempt; the latter (clock “0”) ticks at fresh arrivals. Accordingly, let τ_a be the average time between transmission or retransmission attempts, and τ_0 be the average time between fresh arrivals (Figure 7.2).

1. Consider a simulation such that there are $N + 1$ fresh arrivals, at times T_0, T_2, \dots, T_N , with

N large. T_n are the ticks of clock 0. The estimates of τ_a and τ_0 are

$$\begin{aligned}\tau_a &= \frac{T_N - T_0}{N_a} \\ \tau_0 &= \frac{T_N - T_0}{N}\end{aligned}\tag{7.5}$$

where N_a is the number of transmission or retransmission attempts generated by packets 1 to N . The estimate of the throughput θ is

$$\theta = \frac{N}{T_N - T_0} = \frac{1}{\tau_0}$$

Also, by definition of the error ratio α : $N_a(1 - \alpha) = N$ thus

$$\tau_a = (1 - \alpha)\tau_0$$

2. We focus on τ_a and break it down into pieces corresponding to the ticks of clock 0:

$$\tau_a = \frac{T_N - T_0}{N_a} = \frac{1}{N_a} \sum_{n=1}^N X_n$$

where X_n is the total transmission and retransmission time for the n th packet, i.e. the time interval between two ticks of clock 0. Let A_n be the number of unsuccessful transmission attempts for the n th packet (possibly 0). It comes:

$$\begin{aligned}X_n &= A_n t_1 + t_0 \\ \tau_a &= \frac{1}{N_a} \left(t_1 \sum_{n=1}^N A_n + t_0 N \right) = \frac{1}{N_a} (t_1(N_a - N) + t_0 N) \\ &= \alpha t_1 + (1 - \alpha)t_0\end{aligned}\tag{7.6}$$

3. Compare Eqs.(7.5) and (7.6) and obtain $\tau_0 = \frac{\alpha}{1-\alpha}t_1 + t_0$; the throughput is thus:

$$\theta = \frac{1}{\frac{\alpha}{1-\alpha}t_1 + t_0}\tag{7.7}$$

In this example, as in general with Palm calculus formulas, the validity of a formula such as Eq.(7.7) does not depend on any distributional or independence assumption. We did not make any particular assumption about the arrival and failure processes, they may be correlated, non Poisson, etc.

7.1.4 ARBITRARY SAMPLING METHODS

To conclude this section we describe how different viewpoints occur in various situations, with clocks that may not be related to time. Here too, the large “time” heuristic provides useful relationships.

EXAMPLE 7.4: FLOW VERSUS PACKET CLOCK [96]. Packets arriving at a router are classified in “flows”. We would like to plot the empirical distribution of flow sizes, counted in packets. We measure all traffic at the router for some extended period of time. Our metric of interest is the probability distribution of flow sizes. We can take a flow “clock”, or viewpoint, i.e. ask: pick an arbitrary flow, what is its size ? Or we could take a packet viewpoint and ask: take an arbitrary packet, what is the magnitude of its flow ? We thus have two possible metrics (Figure 7.3):

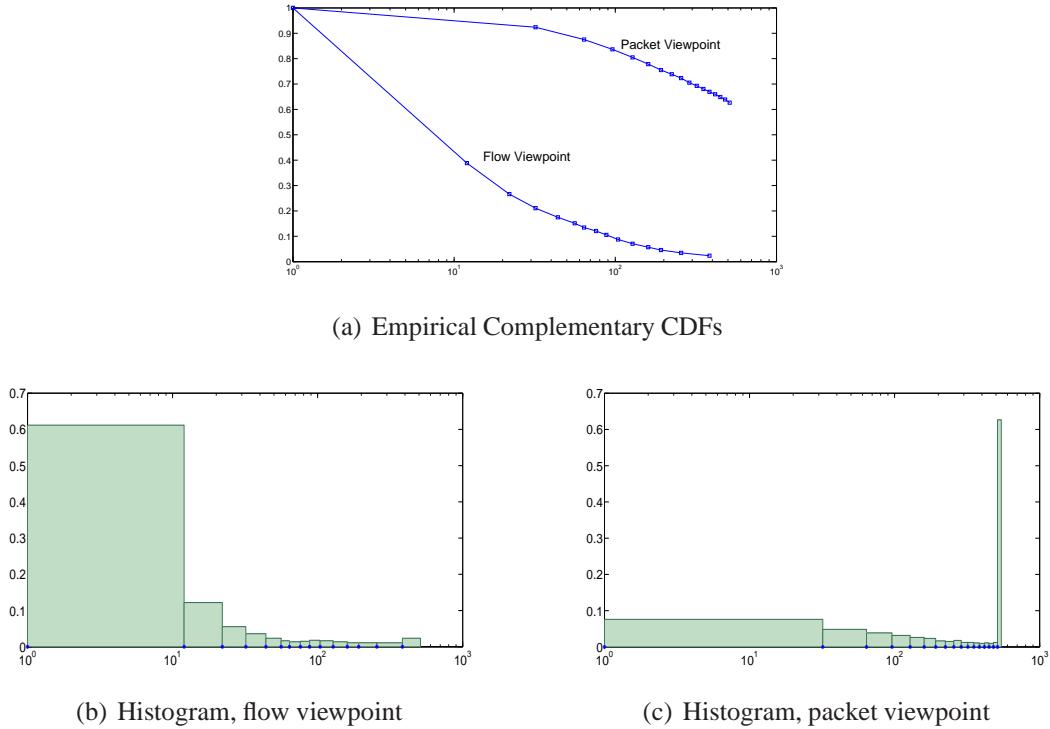


Figure 7.3: Distribution of flow sizes, viewed by an arbitrary flow and an arbitrary packet, measured by an internet service provider.

Per flow $f_F(s) = 1/N \times$ number of flows with length s , where N is the number of flows in the dataset;

Per packet $f_P(s) = 1/P \times$ number of packets that belong to a flow of length s , where P is the number of packets in the dataset;

The large time heuristic helps us find a relation between the two metrics.

1. For s spanning the set of observed flow sizes:

$$f_F(s) = \frac{1}{N} \sum_{n=1}^N \mathbf{1}_{\{S_n=s\}} \quad (7.8)$$

$$f_P(s) = \frac{1}{P} \sum_{p=1}^P \mathbf{1}_{\{S_{F(p)}=s\}} \quad (7.9)$$

where S_n be the size in bytes of flow n , for $n = 1, \dots, N$, and $F(p)$ is the index of the flow that packet number p belongs to.

2. We can break the sum in Eq.(7.9) into pieces that correspond to ticks of the flow clock:

$$\begin{aligned} f_P(s) &= \frac{1}{P} \sum_{n=1}^N \sum_{p:F(p)=n} \mathbf{1}_{\{S_n=s\}} = \frac{1}{P} \sum_{n=1}^N \sum_{p=1}^P \mathbf{1}_{\{F(p)=n\}} \mathbf{1}_{\{S_n=s\}} \\ &= \frac{1}{P} \sum_{n=1}^N \mathbf{1}_{\{S_n=s\}} \sum_{p=1}^P \mathbf{1}_{\{F(p)=n\}} = \frac{1}{P} \sum_{n=1}^N \mathbf{1}_{\{S_n=s\}} s = \frac{s}{P} \sum_{n=1}^N \mathbf{1}_{\{S_n=s\}} \end{aligned} \quad (7.10)$$

3. Compare Eqs.(7.8) and (7.10) and obtain that for all flow size s :

$$f_P(s) = \eta s f_F(s) \quad (7.11)$$

where η is a normalizing constant ($\eta = N/P$).

Eq.(7.11) relates the two ways of computing the distribution of flow sizes. Note that they differ by one exponent, so it could be that the flow size is heavy tailed when sampled with a packet clock, but light tailed when sampled with a flow clock.

EXAMPLE 7.5: KILOMETER VERSUS TIME CLOCK: CYCLIST'S PARADOX. A cyclist rides swiss mountains; his speed is 10 km/h uphill and 50 km/h downhill. A journey is made of 50% uphill slopes and 50% downhill slopes. At the end of the journey, the cyclist is disappointed to read on his speedometer an average speed of only 16.7 km/h, as he was expecting an average of $\frac{10+50}{2} = 30$ km/h. Here, we have two ways of measuring the average speed: with the standard clock (speedometer), or with the kilometer clock (cyclist's intuition). Let us apply the large time heuristic.

1. Pick a unit of length (perhaps smaller than the kilometer) such that the cyclist's speed is constant on a piece of trip of length 1, and let v_l be the speed at the l th piece of the trip, $l = 1, \dots, L$, where L is the trip length. The average speed measured with the standard clock, S_t and with the kilometer clock, S_k are:

$$\begin{aligned} S_t &= L/T \\ S_k &= \frac{1}{L} \sum_{l=1}^L v_l \end{aligned} \quad (7.12)$$

where T is the trip duration.

2. Break L into pieces corresponding to the km clock:

$$\begin{aligned} T &= \sum_{l=1}^L \frac{1}{v_l} \\ S_t &= \frac{L}{\sum_{l=1}^L \frac{1}{v_l}} \end{aligned} \quad (7.13)$$

3. Thus S_t (Eq.(7.13)) is the harmonic mean of v_l whereas S_k (Eq.(7.12)) is the arithmetic mean (Section 2.4.3). The harmonic mean is the inverse of the mean of the inverses. If the speed is not constant throughout the whole trip, the harmonic mean is smaller than the arithmetic mean [106], thus the cyclist's intuition will always have a positive bias (leading to frustration).

In this case the large time heuristic does not give a closed form relationship between the two averages; however, a closed form relationship can be obtained for the two distributions of speeds. Using the same method as in Example 7.4, one obtains

$$f_t(v) = \eta \frac{1}{v} f_k(v) \quad (7.14)$$

where $f_t(v)$ [resp. $f_k(v)$] is the PDF of the speed, sampled with the standard clock [resp. km clock] and η is a normalizing constant; f_t puts more mass on the small values of the speed v , this is another explanation to the cyclist's paradox.

7.2 PALM CALCULUS

Palm calculus is a branch of probability that applies to stationary point processes. We give an intuitive, but rigorous, treatment. A complete mathematical treatment can be found for example in [4, 88] or in [95] in the context of continuous time Markov chains.

7.2.1 HYPOTHESES

STATIONARITY We assume that we are observing the output of a simulation, which we interpret as a sample of a stochastic process $S(t)$. Time t is either discrete or continuous. This process is *stationary* if for any n , any sequence of times $t_1 < t_2 < \dots < t_n$ and any time shift u the joint distribution of $(S(t_1 + u), S(t_2 + u), \dots, S(t_n + u))$ is independent of u . In other words, the process does not change statistically as it gets older. In practice, stationarity occurs if the system has a stationary regime and we let the simulation run long enough (Chapter 6).

We also assume that, at every time t , we are able to make an observation $X(t)$ from the simulation output. The value of $X(t)$ may be in any space. We assume that the process $X(t)$ is *jointly stationary* with the simulation state process $S(t)$ (i.e. $(X(t), S(t))$ is a stationary process). Note that even if the simulation is stationary, one might easily define outputs that are not jointly stationary (such as: $X(t) =$ the most recent request arrival time at an information server). A sufficient condition for $X(t)$ to be jointly stationary with $S(t)$ is

1. at every time t , $X(t)$ can be computed from the present, the past and/or the future of the simulation state $S(t)$, and
2. $X(t)$ is invariant under of change of the origin of times.

For example, if an information server can be assumed to be stationary, then $X(t) =$ time elapsed since the last request arrival time and $X(t) =$ the queue size at time t satisfy the conditions.

7.2.2 DEFINITIONS

POINT PROCESS We introduce now the definition of *stationary point process*. Intuitively, this is the sequence of times at which the simulation does a transition in some specified set.

Formally, a stationary point process in our setting is associated with a subset \mathcal{F}_0 of the set of all possible state transitions of the simulation. It is made of all time instants t at which the simulation does a transition in \mathcal{F}_0 , i.e. such that $(S(t^-), S(t^+)) \in \mathcal{F}_0$.

In practice, we do not need to specify \mathcal{F}_0 explicitly. In contrast, we have a simulation in steady state and we consider times at which something of a certain kind happens; the only important criterion is to make sure that the firing of a point can be entirely determined by observing only the simulation. For example, we can consider as point process the request arrival times at an information server.

Technically, we also need to assume that the simulation process is such that the point process is simple, i.e. with probability 1 two instants of the point process cannot be equal; (this is true in practice if the simulation cannot have several transitions at the same time), and non explosive, i.e. the expected number of time instants over any finite interval is finite. This implies that the instants of the point process can be enumerated and can be described as an increasing sequence of (random) times T_n , where n is integer, and $T_n < T_{n+1}$.

In continuous time, to avoid ambiguities, we assume that all processes are right continuous, so that if there is a transition at time t , $S(t)$ is the state of the simulation just after the transition.

The sequence T_n is (as a thought experiment) assumed to be infinite both in the present and the past, i.e. the index n spans \mathbb{Z} . With the terminology of Section 7.1, $(T_n)_{n \in \mathbb{Z}}$ is the sequence of ticks of the event clock.

THE ARBITRARY POINT IN TIME Since the simulation is in stationary regime, we imagine that, at time 0, the simulation has been running for some time. Because the point process is defined in terms of transitions of the simulation state $S(t)$, it is also stationary. It is convenient, and customary, to denote the time instants of the point process T_n such that

$$\dots < T_{-2} < T_{-1} < T_0 \leq 0 < T_1 < T_2 < \dots \quad (7.15)$$

In other words, T_0 is the last instant of the point process before time 0, and T_1 the next instant starting from time 0. This convention is the one used by mathematicians to give a meaning to “an arbitrary point in time”: we regard $t = 0$ as our random time instant, in some sense, we fix the time origin arbitrarily.

This differs from the convention used in many simulations, where $t = 0$ is the beginning of the simulation. Our convention, in this chapter, is that $t = 0$ is the beginning of the observation period for a simulation that has a stationary regime and has run long enough to be in steady state.

INTENSITY The *intensity* λ of the point process is defined as the expected number of points per time unit. We have assumed that there cannot be two points at the same instant. In discrete or continuous time, the intensity λ is defined as the unique number such that the number $N(t, t + \tau)$ of points during any interval $[t, t + \tau]$ satisfies [4]:

$$\mathbb{E}(N(t, t + \tau)) = \lambda\tau \quad (7.16)$$

In discrete time, λ is also simply equal to the probability that there is a point at an arbitrary time:

$$\lambda = \mathbb{P}(T_0 = 0) = \mathbb{P}(N(0) = 1) = \mathbb{P}(N(t) = 1) \quad (7.17)$$

where the latter is valid for any t , by stationarity.

One can think of λ as the (average) rate of the event clock.

PALM EXPECTATION AND PALM PROBABILITY Let Y be a one time output of the simulation, assumed to be integrable (for example because it is bounded). We define the expectation $\mathbb{E}^t(Y)$ as the conditional expectation of Y given that a point occurs at time t :

$$\mathbb{E}^t(Y) = \mathbb{E}(Y | \exists n \in \mathbb{Z}, T_n = t) \quad (7.18)$$

If $Y = X(t)$ where $X(t)$ and the simulation are jointly stationary, $\mathbb{E}^t(X(t))$ does not depend on t . For $t = 0$, it is called the:

DEFINITION 7.1 (Palm expectation).

$$\mathbb{E}^0(X(0)) = \mathbb{E}(X(0) | \text{a point of the process } T_n \text{ occurs at time } 0) \quad (7.19)$$

By the labeling convention in Eq.(7.15), if there is a point of the process T_n at 0, it must be T_0 , i.e.

$$\mathbb{E}^0(X(0)) = \mathbb{E}(X(0) | T_0 = 0)$$

Note that there is some ambiguity in the notation, as the process T_n is not explicitly mentioned (in Section 7.3.3 we will need to remove this ambiguity).

The Palm *probability* is defined similarly, namely

$$\mathbb{P}^0(X(0) \in W) = \mathbb{P}(X(0) \in W \mid \text{a point of the process } T_n \text{ occurs at time 0})$$

for any measurable subset W of the set of values of $X(t)$. In particular, we can write $\mathbb{P}^0(T_0 = 0) = 1$.

The interpretation of the definition is easy in discrete time, if, as we do, we assume that the point process is “simple”, i.e. there cannot be more than one point any instant t . In this case, Eq.(7.18) has to be taken in the usual sense of conditional probabilities:

$$\mathbb{E}^t(Y) = \mathbb{E}(Y|N(t) = 1) = \frac{\mathbb{E}(YN(t))}{\mathbb{E}(N(t))} = \frac{\mathbb{E}(YN(t))}{\mathbb{P}(N(t) = 1)} = \frac{\mathbb{E}(YN(t))}{\lambda}$$

where $N(t) = 1$ if there is a point at time t , 0 otherwise.

In continuous time, “there is a point at time t ” has probability 0 and cannot be conditioned upon. However, it is possible to give a meaning to such a conditional expectation, similar to the way one can define the conditional probability density function of a continuous random variable:

$$\mathbb{E}^t(Y) = \lim_{\tau \rightarrow 0} \frac{\mathbb{E}(YN(t, t + \tau))}{\mathbb{E}(N(t, t + \tau))} = \lim_{\tau \rightarrow 0} \frac{\mathbb{E}(YN(t, t + \tau))}{\lambda \tau} \quad (7.20)$$

where the limit is in the Radon-Nykodim sense, defined as follows. For a given random variable Y , consider the measure μ defined for any measurable subset B of \mathbb{R} by

$$\mu(B) = \frac{1}{\lambda} \mathbb{E} \left(Y \sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \in B\}} \right) \quad (7.21)$$

where λ is the intensity of the point process T_n . If B is negligible (i.e. its Lebesgue measure, or length, is 0) then, with probability 1 there is no event in B and $\mu(B) = 0$. By the Radon-Nykodim theorem [91], there exists some function g defined on \mathbb{R} such that for any B : $\mu(B) = \int_B g(t) dt$. The Palm expectation $\mathbb{E}^t(Y)$ is defined as $g(t)$. In other words, for a given random variable Y , $\mathbb{E}^t(Y)$ it is defined as the function of t that satisfies, for any B :

$$\mathbb{E} \left(Y \sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \in B\}} \right) = \lambda \int_B \mathbb{E}^t(Y) dt \quad (7.22)$$

7.2.3 INTERPRETATION AS TIME AND EVENT AVERAGES

In this section we make the link with the intuitive treatment in Section 7.1.

TIME AVERAGES. If $X(t)$ is jointly stationary with the simulation, it follows that the distribution of $X(t)$ is independent of t ; it is called the *time stationary* distribution of X .

Assume that, in addition, $X(t)$ is ergodic, i.e that time averages tend to expectations, (which, is for example true on a discrete state space if any state can be reached from any state)), for any bounded function ϕ , we can estimate $\mathbb{E}(\phi(X(t)))$ by (in discrete time):

$$\mathbb{E}(\phi(X(t))) \approx \frac{1}{T} \sum_{t=1}^T \phi(X(t))$$

when T is large. An equivalent statement is that for any (measurable) subset W of the set of values of $X(t)$:

$$\mathbb{P}(X(t) \in W) \approx \text{fraction of time that } X(t) \text{ is in the set } W$$

In other words, the time stationary distribution of $X(t)$ can be estimated by a time average.

EVENT AVERAGES. We can interpret the Palm expectation and Palm probability as event average if the process $X(t)$ is ergodic (note however that Palm calculus does not require ergodicity). Indeed, it follows from the definition of Palm expectation that

$$\mathbb{E}^0(\phi(X(0))) \approx \frac{1}{N} \sum_{n=1}^N \phi(X(T_n))$$

for N large.

It can be shown [4] that if the process $X(t)$ is ergodic and integrable then $\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N \phi(X(T_n)) = \mathbb{E}^0(\phi(X^0))$.

An equivalent statement is that, for any (measurable) subset W of the set of values of $X(t)$:

$$\mathbb{P}^t(X(t) \in W) = \mathbb{P}^0(X(0) \in W) \approx \text{fraction of points of the point process at which } X(t) \text{ is in } W$$

Thus the Palm expectation and the Palm probability can be interpreted as event averages. In other words, they are ideal quantities, which can be estimated by observing $X(t)$ sampled with the event clock.

7.2.4 THE INVERSION AND INTENSITY FORMULAS

formulas that relate time and event averages. Also known under the name of Ryll-Nardzewski and Slivnyak's formula, the inversion formula relates the time stationary and Palm probabilities. The proof for discrete time, a direct application of the definition of conditional probability, is given in appendix.

THEOREM 7.1. (Inversion Formula.)

- *In discrete time:*

$$\mathbb{E}(X(t)) = \mathbb{E}(X(0)) = \lambda \mathbb{E}^0 \left(\sum_{s=1}^{T_1} X(s) \right) = \lambda \mathbb{E}^0 \left(\sum_{s=0}^{T_1-1} X(s) \right) \quad (7.23)$$

- *In continuous time:*

$$\mathbb{E}(X(t)) = \mathbb{E}(X(0)) = \lambda \mathbb{E}^0 \left(\int_0^{T_1} X(s) ds \right) \quad (7.24)$$

By applying the inversion to $X(t) = 1$ we obtain the following formula, which states that the intensity of a point process is the inverse of the average time between points.

THEOREM 7.2. (*Intensity Formula.*)

$$\frac{1}{\lambda} = \mathbb{E}^0(T_1 - T_0) = \mathbb{E}^0(T_1) \quad (7.25)$$

Recall that the only assumption required is stationarity. There is no need for independence or Poisson assumptions.

EXAMPLE 7.6: [GATEKEEPER, CONTINUED.](#) Assume we model the gatekeeper example as a discrete event simulation, and consider as point process the waking ups of the gatekeeper. Let $X(t)$ be the execution time of a hypothetical job that would arrive at time t . The average job execution time, sampled with the standard clock (customer viewpoint) is

$$W_c = \mathbb{E}(X(t)) = \mathbb{E}(X(0))$$

whereas the average execution time, sampled with the event clock (system designer viewpoint), is

$$W_s = \mathbb{E}^t(X(t)) = \mathbb{E}^0(X(0))$$

The inversion formula gives

$$W_c = \lambda \mathbb{E}^0 \left(\int_0^{T_1^-} X(t) dt \right) = \lambda \mathbb{E}^0 (X(T_1^-) T_1)$$

(recall that $T_0 = 0$ under the Palm probability); here $X(T_1^-)$ is the execution time for a job that arrives just before time T_1). Define $X_1 = X(T_1^-)$ and let C be the cross-covariance between sleep time and execution time at the end of the sleep time:

$$C := \mathbb{E}^0(T_1 X_1) - \mathbb{E}^0(T_1) \mathbb{E}^0(X_1)$$

then

$$W_c = \lambda [C + \mathbb{E}^0(X_1) \mathbb{E}^0(T_1)]$$

By the inversion formula $\lambda = \frac{1}{\mathbb{E}^0(T_1)}$ thus

$$W_c = W_s + \lambda C$$

which is the formula we had derived using the heuristic in Section 7.1.

To be rigorous we need to make sure that the process being simulated is stationary. With the data in Example 7.1, this appears to be false, as the wakeup times are periodic, starting at time 0. This is not a problem for such cases: when the simulation state is periodic, say with period θ , then it is customary to consider the simulation as a realization of the stochastic process obtained by drawing the origin of times uniformly in $[0, \theta]$. This produces a stochastic process which is formally stationary. In practical terms, this amounts to choosing the arbitrary point in time uniformly at random in $[0, \theta]$.

EXAMPLE 7.7: [STATIONARY DISTRIBUTION OF RANDOM WAYPOINT \[56\].](#) The random waypoint model is defined in Example 6.5, but we repeat the definitions here. A mobile moves from one waypoint to the next in some bounded space \mathcal{S} . When arrived at a waypoint, say M_n , it picks a new one, say M_{n+1} randomly uniformly in \mathcal{S} , picks a speed V_n uniformly at random between v_{\min} and v_{\max} and goes to the next waypoint M_{n+1} at this constant speed.

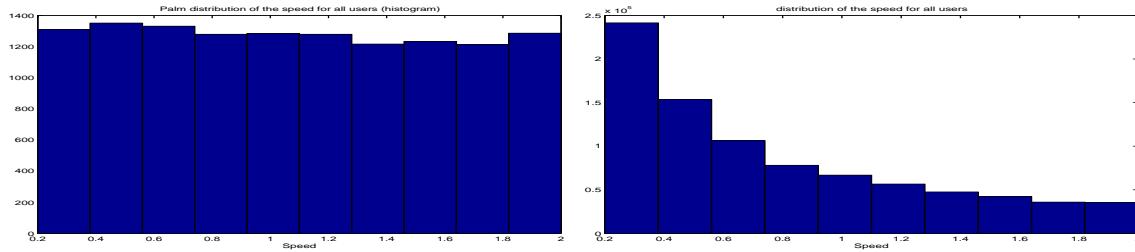


Figure 7.4: Distribution of speed sampled at waypoint (first panel) and at an arbitrary time instant (second panel). $v_{\min} = 0.2$, $v_{\max} = 2$ m/s.

Figure 7.4 shows that the distribution of **speed**, sampled at waypoints, is uniform between v_{\min} and v_{\max} , as expected. In contrast, the distribution, sampled at an arbitrary point in time, is different. We can explain this by Palm's inversion formula.

We assume that this model has a stationary regime, i.e. that $v_{\min} > 0$ (see Section 7.4). The stationary distribution of $V(t)$ is obtained if we know $\mathbb{E}(\phi(V(t)))$ for any bounded, test function ϕ of the speed. Let $f_V^0(v)$ be the PDF of the speed chosen at a waypoint, i.e. $f_V^0(v) = \frac{1}{v_{\max}-v_{\min}}\mathbf{1}_{\{v_{\min}\leq v\leq v_{\max}\}}$. We have

$$\begin{aligned} \mathbb{E}(\phi(V(t))) &= \lambda \mathbb{E}^0 \left(\int_0^{T_1} \phi(V(t)) dt \right) \\ &= \lambda \mathbb{E}^0 (T_1 \phi(V_0)) = \lambda \mathbb{E}^0 \left(\frac{\|M_1 - M_0\|}{V_0} \phi(V_0) \right) = \lambda \mathbb{E}^0 (\|M_1 - M_0\|) \mathbb{E}^0 \left(\frac{1}{V_0} \phi(V_0) \right) \\ &= K_1 \int \frac{1}{v} \phi(v) f_V^0(v) dv \end{aligned} \quad (7.26)$$

where T_n is the time at which the mobile arrives at the waypoint M_n and K_1 is some constant. This shows that the distribution of speed sampled at an arbitrary point in time has PDF

$$f(v) = K_1 \frac{1}{v} f_V^0(v) \quad (7.27)$$

This explains the shape in $\frac{1}{v}$ of the second histogram in Figure 7.4.

A similar argument can be made for the distribution of location. At a waypoint, it is uniformly distributed, by construction. Figure 7.5 shows that, at an arbitrary time instant, it is no longer so. Palm's inversion formula can also be used to derive the PDF of location, but it is very complex [54]. It is simpler to use the perfect simulation formula explained in Section 7.4.3.

7.3 OTHER USEFUL PALM CALCULUS RESULTS

In this section we consider a stationary simulation and a point process following the assumptions in the previous section.

7.3.1 RESIDUAL TIME AND FELLER'S PARADOX

In this section we are interested in the **residual time**, i.e. the time from now to the next point. More precisely, let $T^+(t)$ [resp. $T^-(t)$] be the first point after [resp. before or at] t . Thus, for

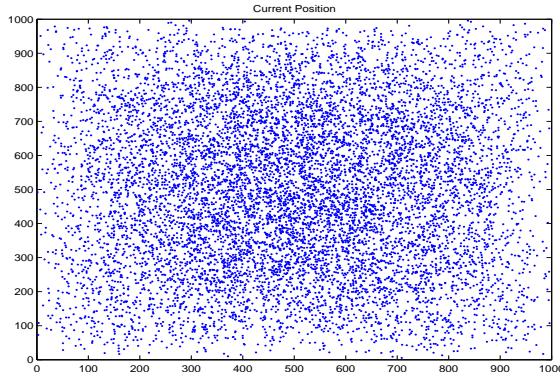


Figure 7.5: A sample of 10^4 points drawn from the stationary distribution of the random waypoint. The distribution is not uniform, even though waypoints are picked uniformly in the area.

example, $T^+(0) = T_1$ and $T^-(0) = T_0$. The following theorem is an immediate consequence of the inversion formula.

THEOREM 7.3. *Let $X(t) = T^+(t) - t$ (time until next point, also called residual time), $Y(t) = t - T^-(t)$ (time since last point), $Z(t) = T^+(t) - T^-(t)$ (duration of current interval). For any t , the distributions of $X(t)$ and $Y(t)$ are equal, with PDF:*

$$f_X(s) = f_Y(s) = \lambda \mathbb{P}^0(T_1 > s) = \lambda \int_s^{+\infty} f_T^0(u) du \quad (7.28)$$

where f_T^0 is the Palm PDF of $T_1 - T_0$ (PDF of inter-arrival times). The PDF of $Z(t)$ is

$$f_Z(s) = \lambda s f_T^0(s) \quad (7.29)$$

In particular, it follows that

$$\mathbb{E}(X(t)) = \mathbb{E}(Y(t)) = \frac{\lambda}{2} \mathbb{E}^0(T_1^2) \quad \text{in continuous time} \quad (7.30)$$

$$\mathbb{E}(X(t)) = \mathbb{E}(Y(t)) = \frac{\lambda}{2} \mathbb{E}^0(T_1(T_1 + 1)) \quad \text{in discrete time} \quad (7.31)$$

$$\mathbb{E}(Z(t)) = \lambda \mathbb{E}^0(T_1^2) \quad (7.32)$$

Note that in discrete time, the theorem means that $\mathbb{P}(X(t) = s) = \mathbb{P}(Y(t) = s) = \lambda \mathbb{P}^0(T_1 \geq s)$ and $\mathbb{P}(Z(t) = s) = \lambda s \mathbb{P}^0(T_1 = s)$.

EXAMPLE 7.8: POISSON PROCESS. Assume that T_n is a Poisson process (see Section 7.6). We have $f_T^0(s) = \lambda e^{-\lambda s}$ and $\mathbb{P}^0(T_1 > s) = \mathbb{P}^0(T_1 \geq s) = e^{-\lambda s}$ thus $f_X(s) = f_Y(s) = f_T^0(s)$.

This is expected, by the memoryless property of the Poisson process: we can think that at every time slot, of duration dt , the Poisson process flips a coin and, with probability λdt , decides that there is an arrival, independent of the past. Thus, the time $X(t)$ until the next arrival is independent of whether there is an arrival or not at time t , and the Palm distribution of $X(t)$ is the same as its time average distribution. Note that this is special to the Poisson process; processes that do not

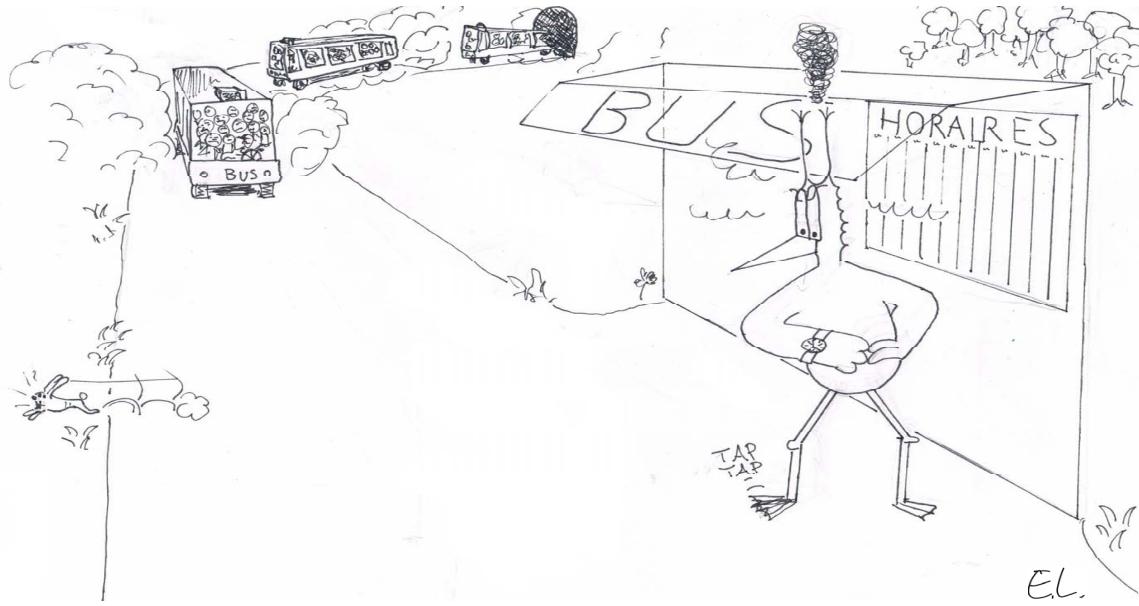
have the memoriless property do not have this feature.

The distribution of $Z(t)$ has density

$$f_T^0(s) = \lambda^2 s e^{-\lambda s}$$

i.e., it is an Erlang-2 distribution¹. Note here that it differs from the Palm distribution, which is exponential with rate λ . In particular, the average duration of the current interval, sampled at an arbitrary point in time, is $\frac{2}{\lambda}$, i.e. twice the average inter-arrival time $\frac{1}{\lambda}$ (this is an instance of Feller's paradox, see later in this section). A simple interpretation for this formula is as follows: $Z(t) = X(t) + Y(t)$, both $X(t)$ and $Y(t)$ are exponentially distributed with rate λ and are independent.

EXAMPLE 7.9: AT THE BUS STOP. T_n is the sequence of bus arrival instants at a bus stop. We do *not* assume here that the bus interarrival times $T_n - T_{n-1}$ are iid. $\mathbb{E}^0(T_1) = \frac{1}{\lambda}$ is the average time between buses, seen by an inspector standing at the bus stop and who spends the hour counting intervals from bus to bus. $\mathbb{E}(T_1) = \mathbb{E}(X(0))$ is the average waiting time experienced by you and me.



By Eq.(7.30):

$$\mathbb{E}(X(t)) = \mathbb{E}(X(0)) = \frac{1}{2} \left(\frac{1}{\lambda} + \lambda \text{var}^0(T_1 - T_0) \right) \quad (7.33)$$

where $\text{var}^0(T_1 - T_0)$ is the variance, under Palm, of the time between buses, i.e. the variance estimated by the inspector. The expectation $\mathbb{E}(X(t))$ is minimum, equal to $\frac{1}{2\lambda}$ when the buses are absolutely regular ($T_n - T_{n-1}$ is constant). The larger the variance, the larger is the waiting time perceived by you and me. In the limit, if the interval between buses seen by the inspector is heavy tailed, then $\mathbb{E}(X(t))$ is infinite. Thus the inspector should report not only the mean time between buses, but also its variance.

¹For $k = 1, 2, 3, \dots$, the **Erlang- k** distribution with parameter λ is the distribution of the sum of k independent exponential distributions with rate λ .

FELLER'S PARADOX. We continue to consider Example 7.9 and assume that Joe would like to verify the inspector's reports by sampling one bus inter-arrival time. Joe arrives at time t and measures $Z(t) = (\text{time until next bus} - \text{time since last bus})$. By Eq.(7.32)

$$\mathbb{E}(Z(t)) = \frac{1}{\lambda} + \lambda \text{var}^0(T_1 - T_0)$$

where $\text{var}^0(T_1 - T_0)$ is the variance of the inter-arrival time ($= \int_0^\infty s^2 f_T^0(s)ds - \frac{1}{\lambda^2}$). Thus, the average of Joe's estimate is *always larger* than the inspector's (which is equal to $\frac{1}{\lambda}$) by a term equal to $\lambda \text{var}^0(T_1 - T_0)$. This happens although both observers sample the same system (but not with the same viewpoint). This systematic bias is known as **Feller's paradox**. Intuitively, it occurs because a stationary observer (Joe) is more likely to fall in a large time interval.

We did not make any assumption other than stationarity about the process of bus arrivals in this example. Thus Feller's paradox is true for any stationary point process.

7.3.2 THE RATE CONSERVATION LAW AND LITTLE'S FORMULA

MIYAZAWA'S RATE CONSERVATION LAW

This is a fundamental result in queuing systems, but it applies to a large variety of systems, well beyond queuing theory. It is best expressed in continuous time.

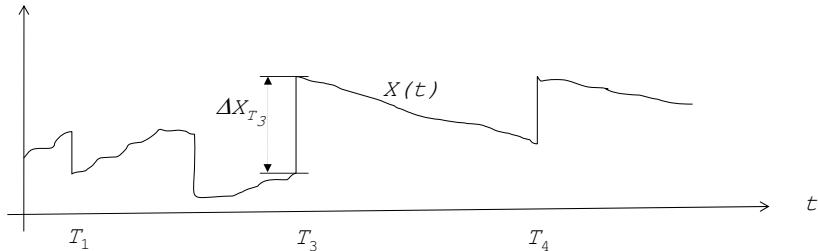


Figure 7.6: Rate Conservation Law.

Consider a random, real valued stochastic process $X(t)$ with the following properties (Figure 7.6):

- $X(t)$ is continuous everywhere except perhaps at instants of a stationary point process T_n ;
- $X(t)$ is continuous to the right;
- $X(t)$ has a right-handside derivative $X'(t)$ for all values of t .

Define Δ_t by $\Delta_t=0$ if t is not a point of the point process T_n and $\Delta_{T_n} = X(T_n) - X(T_n^-)$, i.e. Δ_t is the amplitude of the discontinuity at time t . Note that it follows that

$$X(t) = X(0) + \int_0^t X'(s)ds + \sum_{n \in \mathbb{N}} \Delta_{T_n} \mathbf{1}_{\{t \geq T_n\}} \quad (7.34)$$

THEOREM 7.4. (*Rate Conservation Law [69]*) Assume that the point process T_n and $X(t)$ are jointly stationary. If $\mathbb{E}^0 |\Delta_0| < \infty$ and $\mathbb{E} |X'(0)| < \infty$ then

$$\mathbb{E}(X'(0)) + \lambda \mathbb{E}^0(\Delta_0) = 0$$

where λ is the intensity of the point process T_n and E^0 is the Palm expectation.

The proof in continuous time can be found for example in [70]. We can interpret the theorem as follows.

- $\mathbb{E}(X'(0))$ (also equal to $\mathbb{E}(X'(t))$ for all t) is the average rate of increase of the process $X(t)$, excluding jumps.
- $\mathbb{E}^0(\Delta_0)$ is the expected amplitude of one arbitrary jump. Thus $\lambda \mathbb{E}^0(\Delta_0)$ is the expected rate of increase due to jumps.
- The theorem says that, if the system is stationary, the sum of all jumps cancels out, in average.

Remark. The theorem can be extended somewhat to the cases where some expectations are infinite, as follows [70]. Assume the point process T_n can be decomposed as the superposition of the stationary point processes T_n^j , $j = 1 \dots J$ and that these point processes have no point in common. Let Δ_t^j be the jump of $X(t)$ when t is an instant of the point process T_n^j , i.e.

$$X(t) = X(0) + \int_0^t X'(s)ds + \sum_{j=1}^J \sum_{n \in \mathbb{N}} \Delta_{T_n}^j \mathbf{1}_{\{t \geq T_n^j\}} \quad (7.35)$$

and $\Delta_t^j = 0$ whenever t is not an instant of the point process T_n^j .

Assume that $X'(t) \geq 0$ and the jumps of a point process are all positive or all negative. More precisely, assume that $\Delta_t^j \geq 0$ for $j = 1 \dots I$ and $\Delta_t^j \leq 0$ for $j = I+1, \dots, J$. Last, assume that $X(t)$ and the point processes T_n^j are jointly stationary. Then

$$\mathbb{E}(X'(0)) + \sum_{j=1}^I \lambda_j \mathbb{E}_j^0(\Delta_0^j) = - \sum_{j=I+1}^J \lambda_j \mathbb{E}_j^0(\Delta_0^j) \quad (7.36)$$

where \mathbb{E}_j^0 is the Palm expectation with respect to the point process T_n^j and the equality holds even if some of the expectations are infinite.

EXAMPLE 7.10: M/GI/1 QUEUE AND POLLACZEK-KHINCHINE FORMULA. Consider the M/GI/1 queue, i.e. the single server queue with Poisson arrivals of rate λ and independent service times, with mean \bar{S} and variance σ_S^2 . Assume $\rho = \lambda \bar{S} < 1$ so that there is a stationary regime (Theorem 8.6). Apply the rate conservation law to $X(t) = W(t)^2$, where $W(t)$ is the amount of unfinished work at time t .

The jumps occur at arrival instants, and when there is an arrival at time t , the jump is

$$\Delta_t = (W(t) + S)^2 - W(t)^2 = 2SW(t) + S^2$$

where S is the service time of the arriving customer. By hypothesis, S is independent of $W(t)$ thus the expectation of a jump is $2\mathbb{E}^0(W(t))\bar{S} + \bar{S}^2 + \sigma_S^2$. By the PASTA property (Example 7.19), $\mathbb{E}^0(W(t)) = \mathbb{E}(W(t))$. Thus, the rate conservation law gives

$$\mathbb{E}(X'(t)) + 2\rho \mathbb{E}(W(t)) + \lambda (\bar{S}^2 + \sigma_S^2) = 0$$

Between jumps, $W(t)$ decreases at rate 1 if $W(t) > 0$, thus the derivative of X is $X'(t) = 2W(t)\mathbf{1}_{\{W(t)>0\}}$ and $\mathbb{E}(X'(t)) = -2\mathbb{E}(W(t))$. Putting things together:

$$\mathbb{E}(W(t)) = \frac{\lambda(\bar{S}^2 + \sigma_S^2)}{2(1-\rho)}$$

By the PASTA property again, $\mathbb{E}(W(t))$ is the average workload seen by an arriving customer, i.e. the average waiting time. Thus the average response time (waiting time + service time) is (*Pollaczek-Khinchine formula for means*) :

$$\bar{R} = \frac{\bar{S}(1-\rho(1-\kappa))}{1-\rho} \quad (7.37)$$

with $\kappa = \frac{1}{2}\left(1 + \frac{\sigma_S^2}{\bar{S}^2}\right)$.

Similarly, applying the rate conservation law to $X(t) = e^{-sW(t)}$ for some arbitrary $s \geq 0$ gives the Laplace Stieltjes transform of the distribution of $W(t)$ (see Eq.(8.5)).

CAMPBELL'S SHOT NOISE FORMULA

Consider the following system, assumed to be described by the state of a stationary simulation $S(t)$. Assume that we can observe arrivals of jobs, also called customers, or “shots”, and that the arrival times T_n form a stationary point process.

The n th customer also has an “attribute”, Z_n , which may be drawn according to the specific rules of the system. As usual in this chapter, we do not assume any form of iid-ness, but we assume stationarity; more precisely the attribute Z_n is obtained by sampling the simulation state at time T_n (this is quite general as we do not specify what we put in the simulation state). If the attributes have this property, we say that they are **marks** of the point process T_n and that the process (T_n, Z_n) is a **stationary marked point process**. We do not specify the nature of the attribute, it can take values in any arbitrary space.

When the n th customer arrives, she generates a load on the system, in the form of work to be done. Formally, we assume that there is a function $h(s, z) \geq 0$ (the “shot”) such that $h(s, z)$ is the load at time s , due to a hypothetical customer who arrived at time 0 and would have mark z . The total load in the system at time t , is

$$X'(t) = \sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \leq t\}} h(t - T_n, Z_n)$$

and the total amount of work to be performed, due to customers already present in the system is

$$X(t) = \sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \leq t\}} \int_t^\infty h(s - T_n, Z_n) ds$$

For example, in [7], a customer is an internet flow, its mark is its size in bytes, and the total system load is the aggregate bit rate (Example 7.7). The average load \bar{L} , at an arbitrary point in time, is

$$\bar{L} = \mathbb{E} \left(\sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \leq t\}} h(t - T_n, Z_n) \right) = \mathbb{E} \left(\sum_{n \leq 0} h(-T_n, Z_n) \right)$$

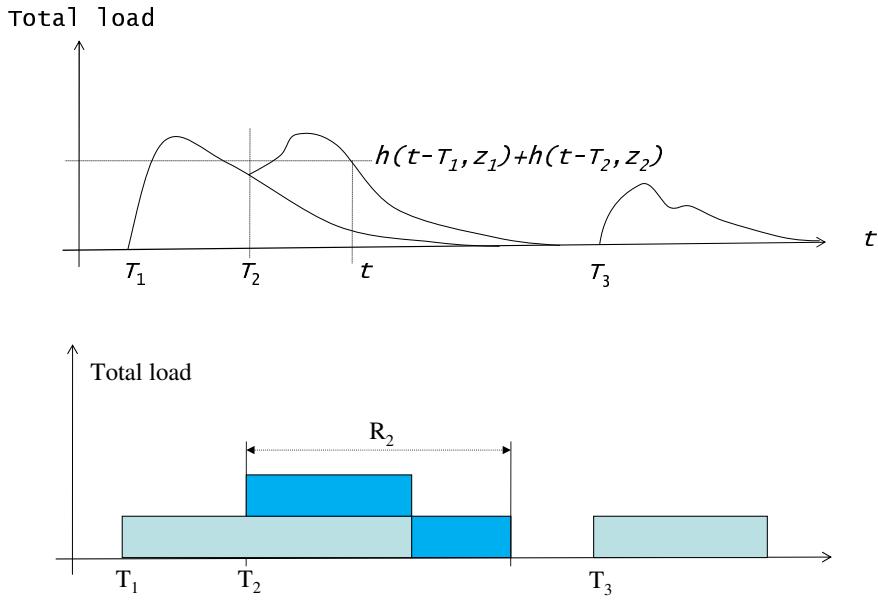


Figure 7.7: Shot Noise (top) and Little's formula (bottom)

where the latter is obtained by taking $t = 0$. The work generated during her lifetime by one customer, given that her mark is z , is $\int_0^\infty h(t, z)dt$. The average load generated by one arbitrary customer can be expressed as a Palm expectation, relative to the point process of customer arrivals, namely as

$$\text{work per customer} = \mathbb{E}^0 \left(\int_0^\infty h(t, Z_0)dt \right) \quad (7.38)$$

(Z_0 in the formula stands for the attribute of an arbitrary customer). Let λ be the intensity of the point process T_n , i.e. the customer arrival rate.

The total work decreases at the rate $X'(t)$ and when a customer arrives at time T_n , jumps by $\Delta_t = \int_0^\infty h(t, Z_0)dt$. The jumps are nonnegative and the derivative is nonpositive thus we can apply Theorem 7.4, more precisely, the remark after it to $-X(t)$, with $J = 1$ and $I = 0$. We have thus shown:

THEOREM 7.5 (Shot Noise). *The average load at an arbitrary point in time is*

$$\bar{L} = \lambda \times \text{work per customer} \quad (7.39)$$

where equality holds also if either \bar{L} or the work per customer is infinite.

Eq.(7.39) is also known as **Campbell's Formula**.

EXAMPLE 7.11: TCP FLOWS. In [7], a customer is a TCP flow, $h(t, z)$ is the bit rate generated at time t by a TCP flow that starts at time 0 and has a size parameter $z \in \mathbb{R}^+$. Thus $\bar{V} = \mathbb{E}^0 (\int_0^\infty h(t, Z_0)dt)$ is the average volume of data, counted in bits, generated by a flow during its entire lifetime. Campbell's formula says that the average bit rate on the system \bar{L} , measured in b/s, is equal to $\lambda \bar{V}$, where λ is the flow arrival rate.

THE $H = \lambda G$ FORMULA

This is an interpretation of the rate conservation law that is quite useful in applications. Consider some arbitrary system where we can observe arrival and departure of jobs (also called customers). Let T_n be the point process of arrivals times, with intensity λ (not necessarily Poisson, as usual in this chapter). Also assume that when a job is present in the system, it uses some amount of resource, per time unit (for example, electrical power or CPU cycles). Assume the system is stationary.

Let G_n be the total amount of system resource consumed by job n during its presence in the system; and let \bar{G} be the average resource consumption per job, i.e. the expectation of G_n when n is an arbitrary customer, and let \bar{H} be the average rate at which the resource is allocated to jobs, i.e. the expectation of $H(t)$ at any time t . Eq.(7.39) can be re-formulated as follows.

The $H = \lambda G$ Formula, or *Extended Little Formula*:

$$\bar{H} = \lambda \bar{G} \quad (7.40)$$

EXAMPLE 7.12: POWER CONSUMPTION PER JOB. A system serves jobs and consumes in average \bar{P} watts. Assume we allocate the energy consumption to jobs, for example by measuring the current when a job is active. Let \bar{E} be the total energy consumed by a job, during its lifetime, in average per job, measured in Joules. By Eq.(7.40):

$$\bar{P} = \lambda \bar{E}$$

where λ is the number of jobs per second, served by the system.

LITTLE' S FORMULA

Consider again some arbitrary system where we can observe arrival and departure of jobs (also called customers), with T_n the point process of arrivals times, with intensity λ . Let R_n be the residence time of the n th customer, $n \in \mathbb{Z}$ (thus her departure time is $T_n + R_n$). Let $N(t)$ be the number of customers present in the system at time t . Assume that the mean residence time \bar{R} (i.e. the expectation of R_n) is finite (by the stationarity assumption it is independent of n).

We did not exactly define what a customer and the system are, therefore we need, formally, to be more precise; this can be done as follows. We are given a sequence $(T_n \in \mathbb{R}, R_n \in \mathbb{R}^+)_{n \in \mathbb{Z}}$ stationary with respect to index n . Assume that T_n can be viewed as a stationary point process, with intensity λ , (i.e. the expectation of $T_n - T_{n-1}$ is finite, Theorem 7.9). The number of customers in the system at time t is then defined by

$$N(t) = \sum_{n \in \mathbb{Z}} \mathbf{1}_{\{T_n \leq t < T_n + R_n\}}$$

Note that, by stationarity, λ is also equal to the departure rate. Define $R(t)$ by $R(t) = R_n$ if and only if $T_n \leq t < T_{n+1}$, i.e. $R(t)$ is the residence time of the most recently arrived customer at time t . Also let

$$\begin{aligned} \mathbb{E}^t(R(t)) &= \mathbb{E}^0(R(0)) = \bar{R} \\ \mathbb{E}(N(t)) &= \mathbb{E}(N(0)) = \bar{N} \end{aligned}$$

We can apply Campbell's formula by letting $Z_n = R_n$ and $h(t, z) = \mathbf{1}_{\{0 \leq t < z\}}$, i.e. the load generated by one customer is 1 as long as it is present in the system; equivalently, we can apply the rate conservation law with $X(t)$ = residual time to be spent by customers present in the system. This gives the celebrated theorem:

THEOREM 7.6 (Little's Formula). *The mean number of customers in the system at time t , $\bar{N} := \mathbb{E}(N(t))$, is independent of t and satisfies*

$$\bar{N} = \lambda \bar{R}$$

where λ is the arrival rate and \bar{R} the average response time, experienced by an arbitrary customer.

Little's formula makes no assumption other than stationarity. In particular, we do not assume that the residence times are independent of the state of the system, and we *do not* assume that the arrival process is Poisson. Also note that the formula holds even if either \bar{N} or \bar{R} is infinite.

Little's formula is very versatile, since it does not say what we call a system and a customer. The next section is an example of this versatility.

DISTRIBUTIONAL LITTLE FORMULA

Assume we are interested not just in the average number of customers in a system, but in the distribution of ages in the system. More precisely, fix $r_0 > 0$; we would like to know $\bar{N}(r_0)$, defined as the average number of customers in the system with an age $\geq r$. Call $f_R()$ the PDF of customer residence time. Consider the virtual system, defined such that we count only customers that have been present in the system for at least r_0 time units:

Original System The n th customer arrives at time T_n and stays for a duration R_n

Virtual System The n th customer arrives at time $T_n + r_0$. If $T_n < r_0$, this customer leaves immediately. Else, this customer stays for a duration $R_n - r_0$.

Apply Little's formula to the virtual system. The average customer residence time in the virtual system is

$$\begin{aligned} \int_{r_0}^{\infty} (r - r_0) f_R(r) dr &= \int_{r_0}^{\infty} \left[\int_{r_0}^r ds \right] f_R(r) dr = \int_{r_0}^{\infty} \int_{r_0}^r f_R(r) ds dr \\ &= \int_{r_0}^{\infty} \left[\int_s^{\infty} f_R(r) dr \right] ds = \int_{r_0}^{\infty} F_R^c(s) ds \end{aligned}$$

where $F_R^c()$ is the complementary CDF of the residence time, i.e. $F_R^c(r) = \int_r^{\infty} f_R(r) dr$. Thus

$$\bar{N}(r_0) = \lambda \int_{r_0}^{\infty} F_R^c(r) dr$$

Let $f_N()$ the PDF of the distribution of ages at an arbitrary point in time, i.e. such that $\bar{N}(r_0) = \bar{N} \int_{r_0}^{\infty} f_N(r) dr$. It follows that $f_N(r) = \frac{\lambda}{\bar{N}} F_R^c(r) = \frac{1}{\bar{R}} F_R^c(r)$, i.e.

$$f_N(r) = \frac{1}{\bar{R}} \int_r^{\infty} f_R(r) dr \tag{7.41}$$

Eq.(7.41) is called a *Distributional Little Formula*. It relates the PDF f_N of the age of a customer sampled at an arbitrary point in time to the PDF of residence times f_R . Note the analogy with Eq.(7.28) (but the hypotheses are different).

7.3.3 TWO EVENT CLOCKS

Assume in this section that we observe two point processes from the same stationary simulation, say $A_n, B_n, n \in \mathbb{Z}$. Let $\lambda(A)$ [resp. $\lambda(B)$] be the intensity of the A [resp. B] point process. Whenever $X(t)$ is some observable output, jointly stationary with the simulation, we can sample $X(t)$ with the two event clocks A , or B , i.e. we can define two Palm probabilities, denoted with $E_A^0(X(0))$ and $E_B^0(X(0))$.

We can also measure the intensity of one point process using the other process's clock; for example, let $\lambda_A(B)$ be the intensity of the B point process measured with the event clock A . Let $N_B[t_1, t_2)$ be the number of points of process B in the time interval $[t_1, t_2)$. We have

$$\lambda_A(B) = \mathbb{E}_A^0(N_B[A_0, A_1]) \quad (7.42)$$

i.e. it is the average number of B points seen between two A points.

THEOREM 7.7. (Neveu's Exchange Formula)

$$\lambda_A(B) = \frac{\lambda(B)}{\lambda(A)} \quad (7.43)$$

$$\mathbb{E}_A^0(X(0)) = \lambda_A(B) \mathbb{E}_B^0 \left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} \right) \quad (7.44)$$

Eq.(7.44) is the equivalent of the inversion formula Eq.(7.23), if we replace the standard clock by clock A and the point process T_n by B_n ; indeed the last term in Eq.(7.44) is the sum of the $X(t)$ values observed at all A points that fall between B_0 and B_1 .

It follows from this theorem that

$$\frac{1}{\lambda_A(B)} = \mathbb{E}_B^0(N_A[0, B_1]) = \mathbb{E}_B^0(N_A[B_0, B_1]) \quad (7.45)$$

which is the equivalent of Eq.(7.25), namely, the intensity of the point process B , measured with A 's clock, is the inverse time between two arbitrary B points, again measured with A 's clock (the last term, $N_A[B_0, B_1]$, is the number of ticks of the A clock between two B points).

The following theorem follows immediately from Theorem 7.7 and Eq.(7.45).

THEOREM 7.8. (Wald's Identity)

$$\mathbb{E}_A^0(X(0)) = \frac{\mathbb{E}_B^0 \left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} \right)}{\mathbb{E}_B^0(N_A[B_0, B_1])} \quad (7.46)$$

Eq.(7.46) is called *Wald's identity*. It is often presented in the context of renewal processes (where interarrival times are i.i.d.), but this need not be: like all Palm calculus formulae, it requires only stationarity, and no independence assumption.

EXAMPLE 7.13: THE STOP AND GO PROTOCOL. We re-visit the computation of the stop and go protocol given in Example 7.3. The A point process consists of the emission times of successful transmissions, and the B point process consists of all transmission and retransmission attempts. Apply Eq.(7.45):

$$\frac{1}{\lambda_A(B)} = \mathbb{E}_B^0(N_A[B_0, B_1])$$

Note that $N_A[B_0, B_1]$ is 1 if the attempt at B_0 is successful and 0 otherwise, thus the right-hand-side in the equation is the probability that an arbitrary transmission or retransmission attempt is successful. By definition of α , this is $1 - \alpha$. Thus $\frac{\lambda(A)}{\lambda(B)} = 1 - \alpha$. Compute $\lambda(B)$ from Eq.(7.25): $\frac{1}{\lambda(B)} = (1 - \alpha)t_0 + \alpha t_1$. Combining the two gives

$$\lambda(A) = \frac{1}{\frac{\alpha}{1-\alpha}t_1 + t_0}$$

as already found.

All formulas in this section continue to hold if we replace the semi-closed intervals that span one tick of an event clock to the next, such as $[A_0, A_1]$ [resp. $[B_0, B_1]$], by the semi-closed intervals $(A_0, A_1]$ [resp. $(B_0, B_1]$], but do not hold if we replace them by closed or open intervals (such as $[A_0, A_1]$ or (A_0, A_1)).

One can even replace them by the so-called *Voronoi* cells, which are the intervals that are bounded by the middle of two successive points, for example one can replace $[A_0, A_1]$ by $[\frac{A_{-1}+A_0}{2}, \frac{A_0+A_1}{2})$ or $(\frac{A_{-1}+A_0}{2}, \frac{A_0+A_1}{2}]$. Thus, for example,

$$\begin{aligned}\lambda_A(B) &= \mathbb{E}_A^0(N_B[A_0, A_1]) = \mathbb{E}_A^0(N_B(A_0, A_1)) \\ &= \mathbb{E}_A^0\left(N_B\left[\frac{A_{-1}+A_0}{2}, \frac{A_0+A_1}{2}\right)\right) = \mathbb{E}_A^0\left(N_B\left(\frac{A_{-1}+A_0}{2}, \frac{A_0+A_1}{2}\right]\right)\end{aligned}$$

and Eq.(7.44) can be generalized to

$$\begin{aligned}\mathbb{E}_A^0(X(0)) &= \lambda_A(B)\mathbb{E}_B^0\left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}}\right) = \lambda_A(B)\mathbb{E}_B^0\left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 < A_n \leq B_1\}}\right) \\ &= \lambda_A(B)\mathbb{E}_B^0\left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{\frac{B_{-1}+B_0}{2} \leq A_n < \frac{B_1+B_2}{2}\}}\right) \\ &= \lambda_A(B)\mathbb{E}_B^0\left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{\frac{B_{-1}+B_0}{2} < A_n \leq \frac{B_1+B_2}{2}\}}\right)\end{aligned}$$

7.4 SIMULATION DEFINED AS STOCHASTIC RECURRENCE

7.4.1 STOCHASTIC RECURRENCE, MODULATED PROCESS

A simulator can be defined as discrete event or as stochastic recurrence (Chapter 6). This also provides a simple, yet powerful model, to analyze stationary but time correlated systems.

Recall that a stochastic recurrence is defined by a sequence $Z_n, n \in \mathbb{Z}$, (also called the modulator state at the n th epoch) and a sequence $S_n > 0$, interpreted as the duration of the n th epoch. The

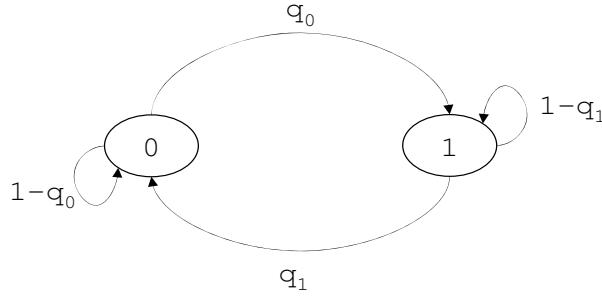


Figure 7.8: The Gilbert Loss Model. When the channel is in state 0 the packet loss ratio is 0, in state 1 it is p_1 . The average number of consecutive periods in state i is $\frac{1}{q_i}$ ($i = 0, 1$).

state space for Z_n is arbitrary, not necessarily finite or even enumerable. We assume that (Z_n, S_n) is random, but stationary² with respect to the index n . As usual, we do not assume any form of independence.

We are interested in the **modulated process** $(Z(t), S(t))$ defined by $Z(t) = Z_n, S(t) = S_n$ whenever t belongs to the n th epoch (i.e. when $T_n \leq t < T_{n+1}$). We would like to apply Palm calculus to $(Z(t), S(t))$.

EXAMPLE 7.14: LOSS CHANNEL MODEL. A path on the internet is modelled as a loss system, where the packet loss ratio at time t , $p(t)$ depends on a hidden state $Z(t) \in \{1, \dots, I\}$ (called the modulator state). During one epoch, the modulator remains in some fixed state, say i , and the packet loss ratio is constant, say p_i . At the end of an epoch, the modulator changes state and a new epoch starts.

Once in a while we send a probe packet on this path, thus we measure the time average loss ratio \bar{p} . How does it relate to p_i ? Apply the inversion formula:

$$\bar{p} = \frac{\sum_i \pi_i^0 p_i \bar{S}_i}{\sum_i \pi_i^0 \bar{S}_i}$$

where π_i^0 is the probability that the modulator is in state i at an arbitrary epoch (proportion of i epochs) and \bar{S}_i is the average duration of an i -epoch.

For example, assume that Z_n is the **Gilbert loss model** shown in Figure 7.8, i.e. a discrete time two state Markov chain, and S_n is equal to one round trip time. We have $\pi_i^0 = \frac{q_{1-i}}{q_0 + q_1}$, for $i = 0, 1$. It follows that

$$\bar{p} = \frac{q_0 p_1}{q_0 + q_1}$$

7.4.2 FREEZING SIMULATIONS

In the previous example, we had implicitly assumed that we can apply Palm calculus, i.e. that the process $Z(t)$ is stationary. In the rest of this section we give conditions for this assumption to be valid.

²This means that the joint distribution of $(Z_n, S_n, \dots, Z_{n+m}, S_{n+m})$ is independent of n .

We first make a technical assumption. It says that the number of epochs per time unit does not explode. More precisely, for any fixed $t_0 > 0$, call

$$D(t_0) = \sum_{n=1}^{\infty} \mathbf{1}_{\{S_0 + \dots + S_{n-1} \leq t_0\}}$$

We interpret $D(t_0)$ as the number of epochs that are entirely included in the interval $(0, t_0]$, given that we start the first epoch at time 0. The technical assumption is

H1 For every t_0 , the expectation of $D(t_0)$ is finite.

Surprisingly, though (Z_n, S_n) is stationary with respect to n , this is not enough to guarantee stationarity of $Z(t)$. To see why, assume that $Z(t)$ is stationary and that there exists a stationary point process T_n such that $T_{n+1} - T_n = S_n$. Apply the inversion formula:

$$\lambda = \frac{1}{\int_0^\infty t f_S^0(t) dt} \quad (7.47)$$

where $f_S^0(t)$ is the probability density function of S_n (it does not depend on n by hypothesis). Thus we need to assume that the expectation of S_n is finite. The next theorem says that, essentially, this is also sufficient.

THEOREM 7.9. Assume that the sequence S_n satisfies **H1** and has finite expectation. There exists a stationary process $Z(t)$ and a stationary point process T_n such that

1. $T_{n+1} - T_n = S_n$
2. $Z_n = Z(T_n)$

The theorem says that we can apply Palm calculus, and in particular treat Z_n as the state of a stationary simulation sampled with the event clock derived from S_n . The proof can be found in [4], where it is called “inverse construction”.

Condition **H1** is often intuitively obvious, but may be hard to verify in some cases. In the simple case where S_n are independent (thus iid since we assume stationarity with respect to n) the condition always holds:

THEOREM 7.10 (Renewal Case). If the S_n are iid and $S_n > 0$, then condition **H1** holds

The next example shows a non iid case.

EXAMPLE 7.15: RANDOM WAYPOINT, CONTINUATION OF EXAMPLE 7.7. For the random waypoint model, the sequence of modulator states is

$$Z_n = (M_n, M_{n+1}, V_n)$$

and the duration of the n th epoch is

$$S_n = \frac{d(M_n, M_{n+1})}{V_n} \quad (7.48)$$

where $d(M_n, M_{n+1})$ is the distance from M_n to M_{n+1} .

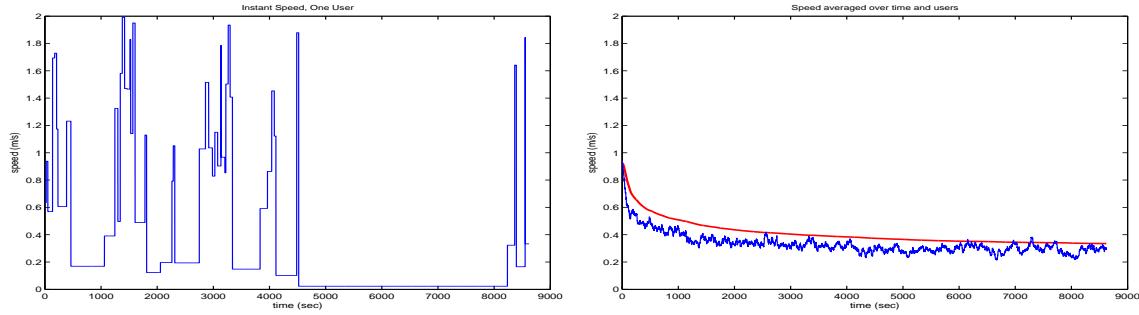


Figure 7.9: Freezing simulation: random waypoint with $v_{\min} = 0$. The model does not have a stationary regime and the simulation becomes slower and slower. First panel: sample of instant speed versus time for one mobile. Second panel: speed averaged over $[0; t]$ for one mobile (zig zag curve) or for 30 mobiles (smoother curve). The average speed slowly tends to 0.

Can this be assumed to come from a stationary process ? We apply Theorem 7.9. The average epoch time is

$$\mathbb{E}(S_0) = \mathbb{E}\left(\frac{d(M_n, M_{n+1})}{V_n}\right) = \mathbb{E}(d(M_n, M_{n+1})) \mathbb{E}\left(\frac{1}{V_n}\right)$$

since the waypoints and the speed are chosen independently. Thus we need that $\mathbb{E}\left(\frac{1}{V_n}\right) < \infty$, i.e. $v_{\min} > 0$.

We also need to verify **H1**. We cannot apply Theorem 7.10 since the epoch times are not independent (two consecutive epoch times depend on one common waypoint). However, but S_m and S_n are independent if $n - m \geq 2$, and one can show that **H1** holds using arguments similar to the proof of Theorem 7.10 [56].

What happens if the expectation of S_n is infinite ? It can be shown (and verified by simulation) that the model **freezes**: as you run the simulation longer and longer, it becomes more likely to draw a very long interval S_n , and the simulation state stays there for long. This is an interesting case where non stationarity is not due to explosion, but to **aging** (Figure 7.9). In the random waypoint example above, this happens if we choose $v_{\min} = 0$.

7.4.3 PERFECT SIMULATION OF STOCHASTIC RECURRENCE

Assume we are interested in simulating the modulator process $(Z(t), S(t))$. A simple method consists in drawing a sample of (Z_0, S_0) from the joint distribution with PDF $f_{Z,S}^0(z, s)$, then decide that the simulation stays in this state for a duration S_0 , then draw Z_1, S_1 from its conditional distribution given (Z_0, S_0) and so on. For a stochastic recurrence satisfying the hypotheses of Theorem 7.9, as the simulation time gets large, the simulation will get into its stationary regime and its state will be distributed according to the stationary distribution of $(Z(t), S(t))$.

It is possible to do better, and start the simulation directly in the stationary regime, i.e. avoid transients at all. This is called **perfect simulation**. It is based on Palm's inversion formula, which gives a way to sample from the stationary distribution, as we explain now.

We want to start a simulation of the modulator process $(Z(t), S(t))$, in stationary regime. We need to draw a sample from the stationary distribution of $(Z(t), S(t))$ but this is not sufficient. We also

need to sample the time until the next change of modulator state. Therefore, it is useful to consider the joint process $(Z(t), S(t), T^+(t))$, where $T^+(t)$ is the residual time, defined in Section 7.3.1 as the time to run until the next change in modulator state, i.e.

$$\text{if } T_n \leq t < T_{n+1} \text{ then } T^+(t) = T_{n+1} - t$$

THEOREM 7.11 (Stationary Distribution of Modulated Process). *Let $(Z_n, S_n)_{n \in \mathbb{Z}}$ satisfy the hypotheses of Theorem 7.9 and let $f_{Z,S}^0(z, s)$ be the joint PDF of (Z_n, S_n) , independent of n by hypothesis. The stationary distribution of $(Z(t), S(t), T^+(t))$ (defined above) is entirely characterized by the following properties:*

1. *The joint PDF of $(Z(t), S(t))$ is*

$$f_{Z,S}(z, s) = \eta s f_{Z,S}^0(z, s) \quad (7.49)$$

where η is a normalizing constant, equal to the inverse of the expectation of S_n :

2. *The conditional distribution of $T^+(t)$ given that $Z(t) = z$ and $S(t) = s$ is uniform on $[0, s]$.*

Recall that Z_n takes values in any arbitrary space, but you may think of it as an element of \mathbb{R}^k for some integer k ³.

Note that the theorem does not directly give a formula for the joint PDF of $(Z(t), S(t), T^+(t))$, though this can be derived, at least in theory, from the combination of items 1 and 2 (see [54] for an example).

Also do not confuse item 2 with the unconditional distribution of the residual time $T^+(t)$. From Theorem 7.3, we know that the distribution of $T^+(t)$ has PDF proportional to $1 - F_S^0(t)$, where $F_S^0()$ is the CDF of S_n , i.e. it is not uniform.

We can recover this result from the above theorem, as follows. Consider a test function $\phi()$ of the residual time $T^+(t)$. The theorem says that

$$\mathbb{E}(\phi(T^+(t)) | Z(t) = s, S(t) = s) = \frac{1}{s} \int_0^s \phi(t) dt$$

thus

$$\begin{aligned} \mathbb{E}(\phi(T^+(t))) &= \eta \int_{z \in \mathcal{Z}} \int_0^\infty \left(\frac{1}{s} \int_0^s \phi(t) dt \right) s f_{Z,S}^0(z, s) dz ds \\ &= \eta \int_{z \in \mathcal{Z}} \int_0^\infty \left(\int_0^s \phi(t) dt \right) f_{Z,S}^0(z, s) dz ds \\ &= \eta \int_0^\infty \left(\int_0^s \phi(t) dt \right) f_S^0(s) ds \\ &= \eta \int_0^\infty \left(\int_t^\infty f_S^0(s) ds \right) \phi(t) dt \\ &= \eta \int_0^\infty (1 - F_S^0(t)) \phi(t) dt \end{aligned}$$

which shows that the PDF of $T^+(t)$ is $\eta(1 - F_S^0(t))$, as given in Theorem 7.3.

³Formally, Z_n may take values in some arbitrary space \mathcal{Z} and S_n is a positive number. We assume that there is a measure μ on \mathcal{Z} and the PDF $f_{Z,S}^0(z, s)$ is defined with respect to the measure product of μ and the Lebesgue measure on $(0, \infty)$.

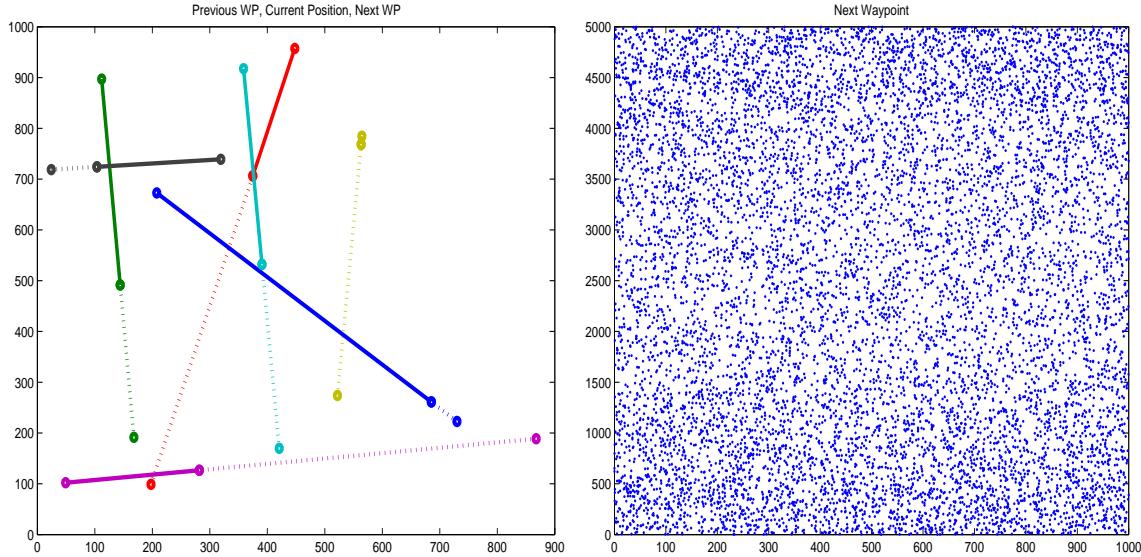


Figure 7.10: Perfect Simulation of Random Waypoint. First Panel: 7 samples of previous waypoint (P), current mobile location (M), and next waypoint (N) sampled at an arbitrary point in time. P and N are not independent; their joint PDF is proportional to their distance. (Compare to the distribution obtained when sampling an arbitrary waypoint: there, by construction, P and N are independent and uniformly distributed), they are independent, by definition of the model). Given P and N , M is uniformly distributed on $[P, N]$. Second panel: 10000 samples of the next waypoint, sampled at an arbitrary point in time. The distribution is not uniform, with a larger density towards the edges.

We obtain a perfect simulation algorithm by immediate application of the above theorem, see Algorithm 5. Note the factor s used when sampling the initial time interval: we can interpret this by saying that the probability, for an observer who sees the system in its stationary regime, of falling in an interval of duration s is proportional to s . This is the same argument as in Feller's paradox (Section 7.3.1).

Algorithm 5 Perfect simulation of Modulated Process

- 1: Sample (z, s) from the joint distribution with PDF $\eta s f_{Z,S}^0(z, s)$ (Eq.(7.49))
 - 2: Sample t uniformly in $[0, s]$
 - 3: Start the simulation with $Z(0) = z, S(0) = s, T^+(0) = T$
-

EXAMPLE 7.16: PERFECT SIMULATION OF RANDOM WAYPOINT. We assume that the model in Example 7.15 has a stationary regime, i.e. that $v_{\min} > 0$. The modulator process is here $Z(t) = (P(t), N(t), V(t), S(t))$ where $P(t)$ [$N(t)$] is the previous [next] waypoint, $V(t)$ is the instant speed and $S(t)$ is the duration of the current trip. Note that $S(t)$ is determined by Eq.(7.48), i.e.

$$S(t) = \frac{d(P(t), N(t))}{V(t)}$$

so it is a deterministic function of $(P(t), N(t), V(t))$ and can be omitted from the description of the modulator process.

Note that by standard change of variable arguments:

$$f_{P,N,V}(p, n, v) = \frac{d(p, n)}{v^2} f_{P,N,S}(p, n, s)$$

$$f_{P,N,V}^0(p, n, v) = \frac{d(p, n)}{v^2} f_{P,N,S}^0(p, n, s)$$

A direct application of Theorem 7.11, item 1, gives the joint PDF of $(P(t), N(t), V(t))$:

$$\begin{aligned} f_{P,N,V}(p, n, v) &= \frac{d(p, n)}{v^2} \eta s f_{P,N,S}^0(p, n, s) = \eta s f_{P,N,V}^0(p, n, v) \\ &= \eta f_{P,N,V}^0(p, n, v) \frac{d(p, n)}{v} \end{aligned}$$

Now, by definition of the random waypoint model, speed and waypoints are chosen independently at a waypoint, i.e.

$$f_{P,N,V}^0(p, n, v) = f_{P,N}^0(p, n) f_V^0(v)$$

Thus

$$f_{P,N,V}(p, n, v) = \eta d(p, n) f_{P,N}^0(p, n) \frac{1}{v} f_V^0(v) \quad (7.50)$$

Since the joint PDF is the product of the PDFs of (P, N) on one hand, V on the other hand, it follows that these two are independent, i.e., when sampled at an arbitrary point in time, the trip endpoints on one hand, and the chosen speed on the other hand, are independent. Furthermore, by marginalization, the joint PDF of $(P(t), N(t))$ is

$$f_{P,N}(p, n) = \eta_1 d(p, n) \quad (7.51)$$

for p, n in the area of interest, and 0 otherwise, and where η is a normalizing constant. Thus the joint PDF of trip endpoints is proportional to their distance, i.e. we are more likely to see long trips in average (this is reminiscent of Feller's paradox in Section 7.3.1, though in space, not in time). It follows also that the distribution of a trip endpoint is not uniform, and that the two endpoints are *not* independent (though they are when sampled at waypoints). Figure 7.10 shows samples from the marginal distribution of $P(t)$ (which is the same as that of $N(t)$). We used rejection sampling (Theorem 6.3), which does not require knowing the normalizing constant η_1 .

We also obtain that the distribution of speed at an arbitrary point in time is proportional to $\frac{1}{v} f_V^0(v)$, which we had already found in Example 7.7. After some algebra, one finds that the CDF of $V(t)$ is

$$F_V(v) = \frac{\ln v - \ln v_{\min}}{\ln v_{\max} - \ln v_{\min}} \quad (7.52)$$

for $v_{\min} \leq v \leq v_{\max}$, 0 if $v \leq v_{\min}$ and 1 if $v \geq v_{\max}$.

Let $M(t)$ be the mobile location at time t . The residual time is related to $M(t)$ by

$$N(t) - M(t) = \frac{T^+(t)V(t)}{d(P(t), N(t))} (N(t) - P(t))$$

so that adding either $T^+(t)$ or $M(t)$ to the modulator process are equivalent. Thus we can take as process state $(P(t), N(t), V(t), M(t))$. A direct application of Theorem 7.11, item 2, together with change of variable arguments as above, give that the conditional distribution of $M(t)$ given that $P(t) = p, N(t) = n, V(t) = v$ is uniform on the segment $[p, n]$. In particular, it is independent of the speed $V(t)$.

We summarize the findings in Algorithm 6.

Algorithm 6 Perfect simulation of Random Waypoint

-
- 1: Sample speed v from the distribution with CDF F_V in Eq.(7.52) (e.g. using CDF inversion, Theorem 6.1).
 - 2: Sample previous waypoint p and next waypoint n from the distribution with PDF proportional to the distance from p to n (e.g. using rejection sampling Theorem 6.3).
 - 3: Sample m uniformly on the segment that joins p and n , e.g. by sampling u uniformly in $[0, 1]$ and letting $m = (1 - u)p + un$.
 - 4: Start the simulation with $P(0) = p, N(0) = n, V(0) = v, M(0) = m$.
-

7.5 APPLICATION TO MARKOV CHAIN MODELS AND THE PASTA PROPERTY

In this section we consider a stochastic process $S(t)$ (the state of the simulation) that can be expressed as a Markov chain, in discrete or continuous time. Formally, this means that the state at time t contains all information for advancing the simulation. Most simulations that we perform in a computer fall in this framework, since the simulation program uses only information available in memory. This does not mean that Markov models are always the best models to analyze a problem, as the state space may be prohibitively large. But it does provide a convenient framework to reason about what we are doing, for example to understand what the PASTA property means (Section 7.5.2). In this section we limit ourselves to Markov chains over a finite state space, as this provides considerable simplifications.

In appendix of this chapter (Section 7.6) we give a quick review of Markov chains. There are many very good books on the topic, see for example [21, 108, 17].

7.5.1 EMBEDDED SUB-CHAIN

If we observe a Markov chain just after some selected transitions, we obtain an *embedded sub-chain*, which is itself a discrete time Markov chain, clocked by the selected transitions. We explain in this section how to compute all elements of the embedded subchain, in particular the Palm probabilities for events observed with the clock of the embedded subchain.

Consider first discrete time. $S(t)$ is a stationary Markov chain with enumerable state space \mathcal{S} . We are interested in observing the transitions of $S(t)$, which is equivalent to observing the process $(S(t-1), S(t))$. Note that this is also a Markov chain. Let $\mathcal{F}_0 \subset \mathcal{S}^2$ be a subset of the set of possible transitions, and call $T_n, n = 1, 2, \dots$ the time instants at which the chain does a transition in \mathcal{F}_0 , i.e.,

$$\begin{aligned} T_1 &\stackrel{\text{def}}{=} \inf \{t > 1 : (S(t-1), S(t)) \in \mathcal{F}_0\} \\ T_n &\stackrel{\text{def}}{=} \inf \{t > T_{n-1} : (S(t-1), S(t)) \in \mathcal{F}_0\} \end{aligned}$$

We assume that there is an infinity of such times, i.e. $T_n < \infty$ with probability 1, and further, that the expected time between visits is also finite⁴. Then, by Theorem 7.9, we can treat T_n as a stationary point process associated with the stationary process $S(t)$.

⁴this is true for example if \mathcal{F}_0 consists of only recurrent non null states of the chain $(S(t-1), S(t))$.

The sequence of states observed just after a transition, $S(T_n)$, is itself a discrete time Markov chain, since the knowledge of the state at the n th transition is sufficient to compute the probabilities of future events (this is the strong Markov property). The sequence $Y_n = S(T_n)$ is called the *embedded sub-chain*. We call *matrix of selected transitions* the matrix of probabilities C defined by

$$C_{i,j} = Q_{i,j} \mathbf{1}_{\{(i,j) \in \mathcal{F}_0\}}$$

for all (i, j) and where Q is the transition matrix of S (see Eq.(7.57)). The matrix C is simply derived by inspection. We also define the matrix $J_{i,j}$ by

$$J_{i,j} \stackrel{\text{def}}{=} \mathbb{P}(S(T_1) = j | S(0) = i)$$

so that $J_{i,j}$ is the transition probability of the chain Y_n if i is a reachable state of Y_n . Note that J is not equal to C , as the next theorem shows.

In continuous time, the definitions are similar (recall that we assume right-continuous sample paths, so a selected transition occurs at time t if $(S(t^-), S(t)) \in \mathcal{F}_0$). The matrix of selected transitions is now a rate matrix, given by

$$C_{i,j} = A_{i,j} \mathbf{1}_{\{(i,j) \in \mathcal{F}_0\}}$$

for all (i, j) and where A is the transition rate matrix of S (with $A_{i,i} = -\sum_{j \neq i} A_{i,j}$). Here we assume that looping transitions are not possible, i.e. $(i, i) \notin \mathcal{F}_0$ for all i . Note that Y_n is a discrete time Markov chain even if $S(t)$ is in continuous time.

THEOREM 7.12. (*Embedded Subchain*) Consider a stationary Markov chain in discrete or continuous time $S(t)$ with $t \in \mathbb{Z}$ or $t \in \mathbb{R}$, with stationary probability π , defined over some enumerable state space. Consider an embedded sub-chain Y_n , $n \in \mathbb{N}$, with the assumptions above, and with matrix of selected transitions C .

1. The transition matrix J of the embedded sub-chain Y_n satisfies $(Id - Q + C)J = C$ (discrete time) or $(C - A)J = C$ (continuous time).
2. The intensity of the point process of selected transitions is $\lambda = \sum_{i,j} \pi_i C_{i,j}$.
3. The probability that an arbitrary selected transition is (i, j) is $\frac{1}{\lambda} \pi_i C_{i,j}$ (in discrete time this is defined as $\mathbb{P}^0(S_{-1} = i, S_0 = j)$; in continuous time as $\mathbb{P}^0(S_{0-} = i, S_0 = j)$).
4. The probability to be in state j just after an arbitrary selected transition is $\frac{1}{\lambda} \sum_i \pi_i C_{i,j}$. The probability to be in state i just before an arbitrary selected transition is $\frac{1}{\lambda} \pi_i \sum_j C_{i,j}$.

EXAMPLE 7.17: QUEUING NETWORK IN FIGURE 8.24. There are two stations, called “Gate” and “Think Time”, one class of customers; we assume to simplify that the service times in both stations are exponentially distributed with parameters μ (at “Gate”) and ν (at “Think Time”). The system can be described as a continuous time Markov chain with state = number of customers at station “Gate”, so that $n \in \{0, \dots, K\}$ where K is the total population size. This is a single class product form network, and from Theorem 8.7, the stationary probability is

$$p(n|K) = \frac{1}{\eta(K)} \frac{1}{\mu^n} \frac{1}{(K-n)! \nu^{K-n}}$$

where we explicitly wrote the dependency on the total population size and $\eta(K)$ is a normalizing constant.

Consider as selected transitions the arrivals at station “Gate”. The matrix of selected transitions is given by

$$C_{n,n+1} = (K - n)\nu \text{ and } C_{n,n'} = 0 \text{ if } n' \neq n + 1$$

The probability that the number of customers is n just after an arrival is, by item 4 of Theorem 7.12:

$$p^0(n) = \frac{1}{\lambda} p(n-1)C(n-1, n) = \frac{1}{\lambda\eta(K)} \frac{1}{\mu^{n-1}} \frac{1}{(K-n)!\nu^{K-n}}$$

This is the same as $p(n-1|K-1)$ if we ignore the normalizing constants, more precisely:

$$p^0(n) = \frac{\eta(K-1)}{\lambda\eta(K)} p(n-1|K-1) \quad (7.53)$$

Since $\sum_{n=1}^K p^0(n) = \sum_{n=1}^K p(n-1|K-1) = 1$, the constant $\frac{\eta(K-1)}{\lambda\eta(K)}$ is 1. i.e.

$$p^0(n) = p(n-1|K-1) \quad (7.54)$$

In other words, an arriving customer samples the network in the same way as if this customer would be removed (this is an instance of the Arrival Theorem 8.16). It follows also that

$$\lambda = \frac{\eta(K-1)}{\eta(K)} \quad (7.55)$$

which is an instance of the Throughput Theorem 8.12.

7.5.2 PASTA

Consider a system that can be modeled by a stationary Markov chain $S(t)$ in discrete or continuous time (in practice any simulation that has a stationary regime and is run long enough). We are interested in a matrix of $C \geq 0$ of selected transitions such that

Independence For any state i of $S(t)$, $\sum_j C_{i,j} \stackrel{\text{def}}{=} \lambda$ is independent of i .

i.e. the rate of occurrence of a selected transition is independent of the global simulation state. Further, this assumption implies that the Point process of selected transitions is a Bernoulli process (discrete time) or a Poisson process (continuous time) with intensity λ (see Section 7.6 for the definition of Poisson and Bernoulli processes).

THEOREM 7.13 (PASTA). *Consider a point process of selected transitions as defined above. The Palm probability just before a transition is the stationary probability.*

The theorem says that, in stationary regime, the Bernoulli, or Poisson clock of selected transitions sees the system in the same way as the standard clock.

Interpret C as external arrivals into a queuing system. The theorem is known as “Poisson Arrivals See Time Averages”, hence the acronym. Note however that this denomination is misleading:

Poisson alone is not sufficient, we need that the point process of selected transition has a rate independent of the state (see Example 7.20).

EXAMPLE 7.18: ARP REQUESTS WITHOUT REFRESHES. IP packets delivered by a host are produced according to a Poisson process with λ packets per second in average. When a packet is delivered, if an ARP request was emitted not more than t_a seconds ago, no ARP request is generated. Else, an ARP request is generated. What is the rate of generation of ARP requests?

Call T_n the point process of ARP request generations, μ its intensity and p the probability that an arriving packet causes an ARP request to be sent. First, we have $\mu = p\lambda$ (to see why, assume time is discrete and apply the definition of intensity).

Second, let $Z(t) = 1$ if the ARP timer is running, 0 if it has expired. Thus p is the probability that an arriving packet sees $Z(t) = 0$. The PASTA property applies, as the IP packet generation process is independent of the state of the ARP timer. (You may establish a formal link with Theorem 7.13 as follows. Think in discrete time. The system can be modeled by a Markov chain with $X(t) = i$ = the residual value of the timer. We have $Q_{i,i-1} = 1$ for $i > 0$, $Q_{0,t_a} = \lambda$, $Q_{0,0} = 1 - \lambda$. The selected transitions are IP packet deliveries, and the probability that one IP packet is delivered in one slot is λ , and does not depend on the state i .)

By the inversion formula:

$$p = \mathbb{P}(Z(t) = 0) = \mu \mathbb{E}^0(T_1 - t_a) = \mu \left(\frac{1}{\mu} - t_a \right) = 1 - \mu t_a \quad (7.56)$$

Combining with $\mu = p\lambda$ gives $p = \frac{1}{\lambda t_a + 1}$, and the rate of generation of ARP requests is $\mu = \frac{\lambda}{1 + \lambda t_a}$.

EXAMPLE 7.19: M/GI/1 QUEUE. A similar reasoning shows that for a queuing system with Poisson arrivals and independent service times, an arriving customer sees the system (just before its own arrival) in the same way as an external observer arriving at an arbitrary instant.

EXAMPLE 7.20: A POISSON PROCESS THAT DOES NOT SATISFY PASTA. The PASTA theorem requires the event process to be Poisson or Bernoulli and independence on the current state. Here is an example of Poisson process that does not satisfy this assumption, and does not enjoy the PASTA property.

Construct a simulation as follows. Requests arrive as a Poisson process of rate λ , into a single server queue. Let T_n be the arrival time of the n th request. The service time of the n th request is assumed to be $\frac{1}{2}(T_{n+1} - T_n)$. The service times are thus exponential with mean $\frac{1}{2\lambda}$, but not independent of the arrival process. Assuming the system is initially empty, there is exactly 1 customer during half of the time, and 0 customer otherwise. Thus the time average distribution of queue length $X(t)$ is given by $\mathbb{P}(X(t) = 0) = \mathbb{P}(X(t) = 1) = 0.5$ and $\mathbb{P}(X(t) = k) = 0$ for $k \geq 2$. In contrast, the queue is always empty when a customer arrives. Thus the Palm distribution of queue length just before an arrival is different from the time average distribution of queue length.

The arrival process does not satisfy the independence assumption: at a time t where the queue is not empty, we know that there cannot be an arrival; thus the probability that an arrival occurs during a short time slot depends on the global state of the system.

APPLICATION TO MEASUREMENTS. The PASTA property shows that sampling a system at random observation instants, distributed like a Poisson or Bernoulli process, independent of the system state, provides an unbiased estimator of the time average distribution.

7.6 APPENDIX: QUICK REVIEW OF MARKOV CHAINS

7.6.1 MARKOV CHAIN IN DISCRETE TIME

Let \mathcal{S} be a **finite** set. A discrete time stochastic process $S(t)$ is a Markov chain on \mathcal{S} if the future evolution of S given all past up to time t is entirely determined by $S(t)$. The **transition matrix** is the matrix of probabilities $Q_{i,j}$ defined by

$$Q_{i,j} = \mathbb{P}(S(t+1) = j | S(t) = i) \quad (7.57)$$

for all i and j in \mathcal{S} .

The state space can be partitioned in **communication classes** as follows: two states i and j are in the same communication class if $i = j$ or if the chain $S(t)$ can go from i to j in a finite number of transitions (each transition must have a positive probability), and vice-versa, from j to i . A communication class is either **recurrent** (once the chain $S(t)$ has entered the class it will remain in the class forever) or not, also called **transient**. If a class is transient, with probability 1, the chain will leave it and never return. States that belong to a transient class are also called transient.

Let $\pi(t)$ be the row vector of probabilities at time t , i.e $\pi_i(t) = \mathbb{P}(S(t) = i)$. Then for all $t \in \mathbb{N}$:

$$\pi(t) = \pi(0)Q^t \quad (7.58)$$

For the chain $S(t)$ to be stationary, we need $\pi(t)$ independent of t , which implies that π satisfies the linear system

$$\begin{cases} \pi = \pi Q \\ \sum_{i \in \mathcal{S}} \pi_i = 1 \end{cases} \quad (7.59)$$

It turns out that this also sufficient, i.e if $\pi(0)$ is solution of Eq.(7.59), then $S(t)$ is stationary. A solution of Eq.(7.59) is called a **stationary probability** of the Markov chain.

Note that, because Q is a stochastic matrix, any solution $\pi \in \mathbb{R}^{\mathcal{S}}$ of Eq.(7.59) is necessarily non-negative. Since \mathcal{S} is finite the situation is simple: stationary probabilities correspond to recurrent classes. More precisely

- There is at least one recurrent class.
- For every recurrent class c there is one stationary probability vector π^c , such that $\pi_i^c > 0$ if $i \in c$ and $\pi_i^c = 0$ otherwise; any stationary probability is a weighted average of the π^c 's.
- If there is only one recurrent class, the chain is called **irreducible**. If the chain is irreducible, there is exactly one stationary probability, and vice-versa, i.e. if Eq.(7.59) has only one solution the chain is irreducible.
- The chain is **ergodic** in the wide sense⁵ if it is irreducible, and vice-versa.

⁵Some authors use a more restrictive definition and say that a finite space markov chain is “ergodic” if it is irreducible and aperiodic, see later. We prefer to use the general definition, which is that time averages tend to expectations.

- If there is more than one recurrent class, the chain will eventually enter one recurrent class and remain there forever. The probability that the chain enters recurrent class c depends on the initial condition.
- If π is a stationary probability vector and i is a transient state, $\pi_i = 0$.

Thus, when \mathcal{S} is finite, there is always at least one stationary regime. If the chain $S(t)$ is not irreducible (i.e. not ergodic) there may be several stationary regimes, and the stationary regime that the chain eventually enters may be random. This happens for example for systems that may have several failure modes.

Consider an ergodic chain (with finite state space). It is stationary if the initial distribution of state is the stationary probability. Otherwise, it becomes stationary as $t \rightarrow \infty$, but there is a technicality due to periodicity. A recurrent class c is called **periodic** with period d if all cycles in the class have a length multiple of some $d \geq 2$ (i.e. whenever $X(t) = i, X(t+s) = i$ for $i \in c$ and $s > 0$, s must be a multiple of d); otherwise, the class is aperiodic. A chain with a single recurrent class is said periodic [resp. aperiodic] if its unique recurrent class is periodic [resp. aperiodic].

If the chain is ergodic and aperiodic then

$$\lim_{t \rightarrow \infty} \pi(t) = \pi$$

where π is the unique stationary probability and thus the chain becomes stationary for large t . Else, if the chain is periodic with period d

$$\lim_{t \rightarrow \infty} \frac{1}{d} (\pi(t) + \pi(t+1) + \dots + \pi(t+d-1)) = \pi$$

which can be interpreted as follows. Change the time origin randomly uniformly in $\{0, 1, \dots, d-1\}$. Then as $t \rightarrow \infty$, the chain becomes stationary.

If the state space is enumerable but infinite, the situation is more complex; there may not exist a recurrent class, and even if there is, there may not exist a stationary probability (the chain “escapes to infinity”). However, there is a simple result. If the chain is irreducible, then Eq.(7.59) has 0 or 1 solution. If it has 1 solution, then it is ergodic and all statements above for an ergodic chain over a finite space continue to hold.

7.6.2 MARKOV CHAIN IN CONTINUOUS TIME

In continuous time, the definition of the Markov Chain is similar, i.e. \mathcal{S} is an enumerable set and the continuous time stochastic process $S(t)$ is a Markov chain on \mathcal{S} if the future evolution of S given all past up to time t is entirely determined by $S(t)$. We assume as usual that $S(t)$ is right-continuous, i.e. $S(t^+) = S(t)$, so that if there is a transition at time t , $S(t)$ is the state just after the transition⁶. Note that some authors reserve the term **Markov chain** to discrete time, whereas some others reserve it to discrete or continuous time processes over a discrete state space (as we do).

The transition matrix is replaced by a matrix of rates, called the **rate transition matrix**, or **generator matrix**, A . It has the property that

$$\mathbb{P}(S(t+dt) = j | S(t) = i) = A_{i,j} dt + o(dt) \quad (7.60)$$

⁶Transitions in continuous time are often called “jumps”

for $i \neq j$. Thus $A_{i,j}$ is interpreted as the rate of transition from state i to j and is necessarily nonnegative. If the state space is infinite, we need to assume that the process is not explosive, which means here that for all $i \in \mathcal{S}$:

$$\sum_{j \neq i} A_{i,j} < \infty \quad (7.61)$$

It is customary to pose

$$A_{i,i} = -\sum_{j \neq i} A_{i,j} \quad (7.62)$$

so that A has non-negative entries everywhere except on the diagonal and $\sum_j A_{i,j} = 0$. It can be shown that the time until the next jump given that $S(t) = i$ is an exponential random variable with parameter $-A_{i,i}$.

Let $\pi(t)$ be the row vector of probabilities at time t , i.e. $\pi_i(t) = \mathbb{P}(S(t) = i)$. Then for all $t \geq 0$:

$$\pi(t) = \pi(0)e^{tA} \quad (7.63)$$

(the exponential of a matrix is defined like for complex numbers by $e^A = \sum_{n=0}^{\infty} A^n / n!$).

A stationary probability is a row vector π that satisfies

$$\begin{cases} \pi A = 0 \\ \sum_{i \in \mathcal{S}} \pi_i = 1 \end{cases} \quad (7.64)$$

which is the replacement for Eq.(7.59). Otherwise, the rest of Section 7.6.1 applies, mutatis mutandi, with one simplification: there is no issue of periodicity. Thus, in particular, a continuous time Markov chain over a finite state space becomes stationary as $t \rightarrow \infty$.

For more details about Markov chains in continuous time, see [94].

7.6.3 POISSON AND BERNOULLI

Those are the two memoriless stationary point processes.

A **Bernoulli process** with intensity $q \in [0, 1]$ is a point process $T_n \in \mathbb{Z}$ in discrete time, such that the points are independently drawn. In other words, at every time t , toss a coin and with probability q decide that there is a point, otherwise there is not. With the terminology of Section 7.2, the sequence $N(t)$ is iid. The time intervals between points, $S_n = T_{n+1} - T_n$, are independent and are such that $S_n - 1$ has a geometric distribution with parameter q . The same holds for the time from now to the next point.

A **Poisson process** $T_n \in \mathbb{R}$ with intensity $\lambda > 0$ is the continuous time equivalent of a Bernoulli process. We do not define it here formally, but, instead, give its main properties:

- The probability that there is a point in $[t, t + dt]$ is $\lambda dt + o(dt)$
- The number of points in disjoint time intervals are independent random variables.
- The number of points in an interval of duration t is a random variable with distribution Poisson(λt)
- The time intervals between points, $S_n = T_{n+1} - T_n$, are independent and have an exponential distribution with parameter λ . The time from now to the next point has the same distribution (but see also Example 7.8).

It can be shown that the Poisson process with intensity λ is the limit, in various senses, when $dt \rightarrow 0$, of the Bernoulli process with intensity $q = \lambda dt$, when we map the time slot of the Bernoulli process to a continuous time interval of duration dt .

7.7 PROOFS

Except for Theorem 7.10 and Theorem 7.12, we give the proofs in discrete time, as they are simple and require only a first course on probability. The proofs in continuous time that are not given can be found in [4], [88] or [70].

THEOREM 7.1 Let $N(t) = 1$ if the point process has a point at time t , 0 otherwise. We show only that $\mathbb{E}(X(0)) = \lambda \mathbb{E}^0 \left(\sum_{s=1}^{T_1} X(s) \right)$, as the second equality is similar. By definition of a conditional probability and of λ :

$$\lambda \mathbb{E}^0 \left(\sum_{s=1}^{T_1} X(s) \right) = \mathbb{E} \left(\sum_{s=1}^{T_1} X(s) N(0) \right)$$

Now for $s > 0$, the event “ $s \leq T_1$ ” is equivalent to “ $N(1, s-1) = 0$ ” thus

$$\begin{aligned} \lambda \mathbb{E}^0 \left(\sum_{s=1}^{T_1} X(s) \right) &= \mathbb{E} \left(\sum_{s=1}^{\infty} X(s) N(0) \mathbf{1}_{\{N(1, s-1)=0\}} \right) \\ &= \mathbb{E} \left(\sum_{s=1}^{\infty} X(0) N(-s) \mathbf{1}_{\{N(1-s, 1)=0\}} \right) = \mathbb{E} \left(X(0) \sum_{s=1}^{\infty} N(-s) \mathbf{1}_{\{N(1-s, 1)=0\}} \right) \end{aligned}$$

where the last line is by stationarity. Let $T^-(-1)$ be the most recent time at which a selected event occurred before or at time -1 . This time is finite with probability 1, by stationarity. We have $N(-s) \mathbf{1}_{\{N(1-s, 1)=0\}} = 1$ if and only if $T^-(-1) = -s$, thus, with probability 1:

$$1 = \sum_{s=1}^{\infty} N(-s) \mathbf{1}_{\{N(1-s, 1)=0\}}$$

which shows the formula.

THEOREM 7.3 $X(t)$ is jointly stationary with T_n , thus its distribution is independent of t , and we can apply the inversion formula. For any $s \geq 0$ we have

$$\mathbb{P}(X(0) = s) = \mathbb{E}(\mathbf{1}_{\{X(0)=s\}}) = \lambda \mathbb{E}^0 \left(\sum_{u=0}^{T_1-1} \mathbf{1}_{\{X(u)=s\}} \right)$$

Given that there is a point at 0 and $0 \leq u \leq T_1 - 1$, we have $X(u) = T_1 - u$, thus

$$\mathbb{P}(X(0) = s) = \lambda \mathbb{E}^0 \left(\sum_{u=0}^{T_1-1} \mathbf{1}_{\{T_1-u=s\}} \right)$$

Now the sum in the formula is 1 if $T_1 > s$ and 0 otherwise. Thus

$$\mathbb{P}(X(0) = \tau) = \lambda \mathbb{E}^0 (\mathbf{1}_{\{T_1>\tau\}}) = \lambda \mathbb{P}^0(T_1 > \tau)$$

which shows the formula for $X(t)$. The formula for $Y(t)$ is similar, using $Y_u = u$ for $0 \leq u \leq T_1 - 1$.

For $Z(t)$, apply the inversion formula and obtain

$$\mathbb{P}(Z_0 = s) = \lambda \mathbb{E}^0 \left(\sum_{u=0}^{T_1-1} \mathbf{1}_{\{Z_u=s\}} \right)$$

Now under P^0 , $Z_u = T_1$ does not depend on u for $0 \leq u \leq T_1 - 1$ thus

$$\mathbb{P}(Z_0 = s) = \lambda \mathbb{E}^0 \left(\mathbf{1}_{\{T_1=s\}} \sum_{u=0}^{T_1-1} 1 \right) = \lambda \mathbb{E}^0 (T_1 \mathbf{1}_{\{T_1=s\}}) = \lambda s \mathbb{P}^0(T_1 = s)$$

THEOREM 7.7 Apply the inversion formula to the B point process and to $X(t)N^A(t)$ where $N^A(t)$ is 1 if there is an A point at t and 0 otherwise. Note that

$$\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} = \sum_{s=B_0}^{B_1-1} X_s N^A(s)$$

thus

$$\begin{aligned} \lambda(B) \mathbb{E}(X(0)N^A(0)) &= \mathbb{E}_B^0 \left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} \right) \\ \lambda(B) \frac{\mathbb{E}_A^0(X(0))}{\lambda(A)} &= \mathbb{E}_B^0 \left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} \right) \\ \lambda(B) \mathbb{E}_A^0(X(0)) &= \lambda(A) \mathbb{E}_B^0 \left(\sum_{n \in \mathbb{Z}} X(A_n) \mathbf{1}_{\{B_0 \leq A_n < B_1\}} \right) \end{aligned} \quad (7.65)$$

Apply the last equation to $X(t) = 1$ and obtain Eq.(7.43). Combine Eq.(7.65) with Eq.(7.43) and obtain Eq.(7.44).

THEOREM 7.10 First note that the expectation of $N(t_0)$ is

$$\sum_{n \geq 1} \mathbb{P}(S_0 + \dots + S_{n-1} \leq t) \quad (7.66)$$

Pick some arbitrary, fixed $s > 0$; by Markov's inequality:

$$\begin{aligned} \mathbb{P}(S_0 + \dots + S_{n-1} \leq t_0) &\leq e^{st_0} \mathbb{E}(e^{-s(S_0 + \dots + S_{n-1})}) \\ &= e^{st_0} G(s)^n \end{aligned}$$

where $G(s) := \mathbb{E}(e^{-sS_0})$ is the Laplace-Transform of S_0 . We have $G(s) = 1$ if and only if $sS_0 = 0$ with probability 1. Thus, by hypothesis, $G(s) < 1$ since $s > 0$. By Eq.(7.66):

$$E(N(t_0)) \leq e^{st} \sum_{n \geq 1} (G(s))^n < \infty$$

THEOREM 7.11 Let ϕ be an arbitrary bounded test function of $Z(t), S(t)$. Apply Palm's inversion formula:

$$\begin{aligned} \mathbb{E}(\phi(Z(t), S(t))) &= \lambda \mathbb{E}^0 \left(\int_0^{T_1} \phi(Z_0, T_1) dt \right) \\ &= \lambda \mathbb{E}^0 (T_1 \phi(Z_0, T_1)) = \lambda \mathbb{E}^0 (S_0 \phi(Z_0, S_0)) \\ &= \lambda \int_{\mathcal{Z} \times (0, \infty)} \phi(z, s) s f_{Z,S}^0(z, s) d\mu(z) ds \end{aligned}$$

from where item 1 follows, with $\eta = \lambda$.

Since the knowledge of $\mathbb{E}(\phi(Z(t), S(t))\psi(T^+(t)))$ for any ϕ, ψ determines the joint distribution of $(Z(t), S(t), T^+(t))$, to show item 2, it is sufficient to show that for any bounded, test function ψ of $T^+(t)$ and any bounded test function of $Z(t), S(t)$, we have:

$$\mathbb{E}(\phi(Z(t), S(t))\psi(T^+(t))) = \int_{z \in \mathcal{Z}, s > 0} \phi(z, s) s f_{Z,S}^0(z, s) \left(\int_0^s \frac{1}{s} \psi(t) dt \right) d\mu(z) ds$$

which, is equivalent to

$$\mathbb{E}(\phi(Z(t), S(t))\psi(T^+(t))) = \int_{z \in \mathcal{Z}, s > 0} \phi(z, s) f_{Z,S}^0(z, s) \left(\int_0^s \psi(t) dt \right) d\mu(z) ds \quad (7.67)$$

Apply Palm's inversion formula again:

$$\begin{aligned} \mathbb{E}(\phi(Z(t), S(t))\psi(T^+(t))) &= \lambda \mathbb{E}^0 \left(\int_0^{S_0} \phi(Z_0, S_0) \psi(S_0 - u) du \right) \\ &= \lambda \mathbb{E}^0 \left(s \phi(Z_0, S_0) \frac{1}{s} \int_0^{S_0} \psi(S_0 - u) du \right) \\ &= \lambda \int_{z \in \mathcal{Z}, s > 0} \phi(z, s) f_{Z,S}^0(z, s) \left(\int_0^s \psi(s - u) du \right) d\mu(z) ds \end{aligned}$$

which, after the change of variable $t = s - u$ in the inner integral is the same as Eq.(7.67).

THEOREM 7.12 By the strong markov property:

$$J_{i,j} = \mathbb{P}^0(X_{T_1} = j | X_{T_0} = i) = \mathbb{P}(X_{T^+(0)} = j | X_0 = i)$$

Condition with respect to the next transition, selected or not:

$$J_{i,j} = \sum_{k:(i,k) \in F} Q_{i,k} + \sum_{k:(i,k) \notin F} Q_{i,k} \mathbb{P}(X_{T^+(0)} = j | X_1 = k \text{ and } X_0 = i)$$

Now, for $(i, k) \notin F$, given that $X_0 = i, X_1 = k$, we have $T^+(0) = T^+(1)$. Thus, the last term in the previous equation is

$$\sum_{k:(i,k) \notin F} Q_{i,k} \mathbb{P}(X_{T^+(1)} = j | X_1 = k \text{ and } X_0 = i) = \sum_{k:(i,k) \notin F} Q_{i,k} J_{k,i}$$

Combining the two gives $J = C + (Q - C)J$ which shows item 1.

Now, by definition of an intensity, $\lambda = \sum_{(i,j) \in F} \mathbb{P}(X_0 = j, X_{-1} = i)$ and $\mathbb{P}(X_0 = j, X_{-1} = i) = \pi_i Q_{i,j}$, which shows item 2.

By definition of the Palm probability:

$$\mathbb{P}^0(X_{-1} = i, X_0 = j) = \frac{1}{\lambda} \mathbb{E}(\mathbf{1}_{\{X_{-1}=j\}} \mathbf{1}_{\{X_0=i\}} \mathbf{1}_{\{(i,j) \in F\}}) = \frac{1}{\lambda} \mathbb{P}(X_{-1} = j, X_0 = i) \mathbf{1}_{\{(i,j) \in F\}}$$

which shows item 3. Item 4 follows immediately.

THEOREM 7.13 The probability that there is a transition at time 1, given that $X_0 = i$, is λ , independent of i . Thus $N(1)$ is independent of the state at time 0. Since we have a Markov chain, the state at time 1 depends on the past only through the state at time 0. Thus $N(1)$ is independent of $N(t)$ for all $t \geq 0$. By stationarity, it follows that $N(t)$ is iid, i.e. is a Bernoulli process.

The relation between Palm and stationary probabilities follows from Theorem 7.12, item 4. The Palm probability to be in state i just before a transition is

$$\frac{1}{\lambda_0} \pi_i \sum_i C(i, j) = \frac{\lambda}{\lambda_0} \pi_i$$

where λ_0 is the λ of Theorem 7.12. The sum of probabilities is 1, thus necessarily $\frac{\lambda}{\lambda_0} = 1$.

7.8 REVIEW QUESTIONS

QUESTION 7.8.1. Consider the Surge model with one user equivalent in Section 3.5.5. Assume the average inactive off period is Z , the average active off period is Z' , the average number of URLs requested per active period is V , and the average response time for a URL request is R . What is the throughput of requests λ ?⁷

QUESTION 7.8.2. A distributed protocol establishes consensus by periodically having one host send a message to n other hosts and wait for an acknowledgement [5]. Assume the times to send and receive an acknowledgement are iid, with distribution $F(t)$. What is the number of consensus per time unit achieved by the protocol? Give an approximation using the fact that the mean of the k th order statistic in a sample of n is approximated by $F^{-1}(\frac{k}{n+1})$.⁸

QUESTION 7.8.3. (ARP protocol with refreshes) IP packets delivered by a host are produced according to a stationary point process with λ packets per second in average. Every packet causes the emission of an ARP if the previous packet arrived more than t_a seconds ago (t_a is the ARP timer). What is the average number of ARP requests generated per second?⁹

QUESTION 7.8.4. Consider the notation of Theorem 7.3. Is the distribution of $Z(t)$ equal to the convolution of those of $X(t)$ and $Y(t)$?¹⁰

⁷Using the large time heuristic, one finds $\lambda = \frac{1}{V(R+Z')+Z}$

⁸Call T_n the point process of the starting points for consensus rounds. The required answer is the intensity λ of T_n . We have $\lambda = \mathbb{E}^0(T_1)$. Now assuming that a round starts at time 0, we have $T_1 = \max_{i=1\dots n} S_i$ where $S_i \sim \text{iid}$ with distribution $F()$. Thus

$$\mathbb{P}^0(T_1 < t) = \mathbb{P}^0(S_1 < t \text{ and } \dots \text{ and } S_n < t) = F(t)^n$$

thus

$$\mathbb{E}^0(T_1) = \int_0^{+\infty} (1 - F(t)^n) dt$$

and

$$\lambda = \frac{1}{\int_0^{+\infty} (1 - F(t)^n) dt}$$

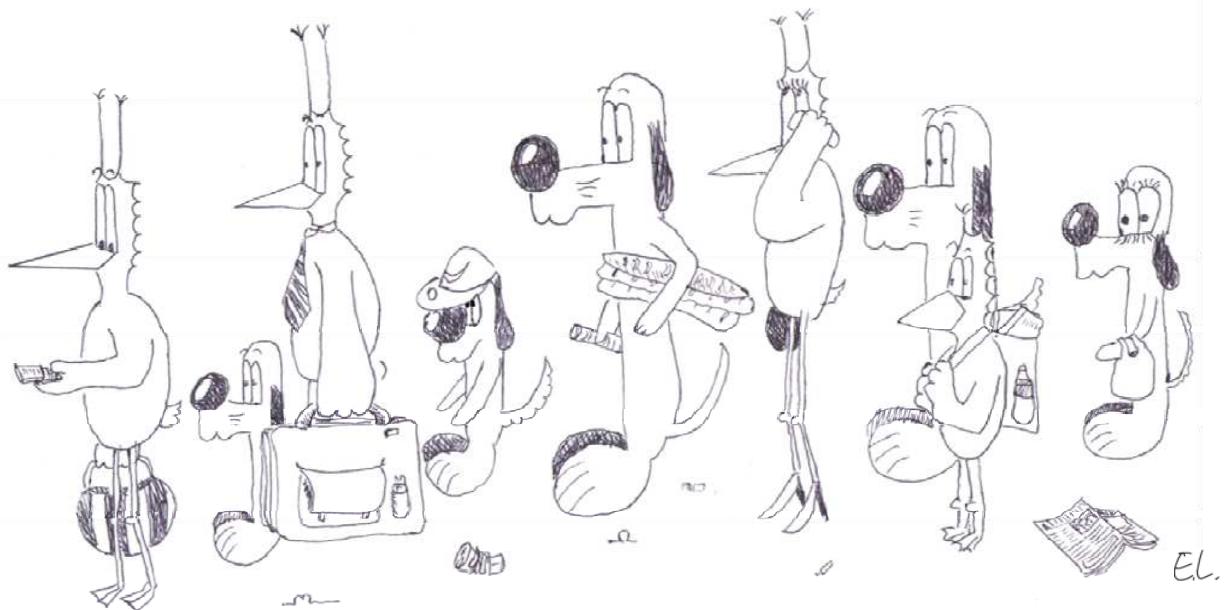
The Palm distribution of T_1 is that of the maximum of n iid random variables, thus $\mathbb{E}^0(T_1) \approx F^{-1}\left(\frac{n}{n+1}\right)$.

⁹Apply Neveu's exchange formula to : first process = ARP request emissions (intensity λ_1); second process = all packet arrivals (intensity λ) and $X_s = 1$. This gives $\lambda_1 = \lambda \mathbb{E}^0(N_1(0, T])$, where \mathbb{E}^0 is the Palm probability for the second point process and N_1 is the number of ARP requests. Given that there is a packet arrival at time 0, $N_1(0, T] = \mathbf{1}_{\{T_1 - T_0 > t_a\}}$. Thus the required throughput is $\lambda_1 = \lambda \mathbb{P}^0(T_1 > t_a)$. It depends only on the tail of the packet inter-arrival time.

¹⁰On one hand, $Z(t) = X(t) + Y(t)$, so it seems tempting to say yes. It is true for a Poisson process. However, consider the case where $T_{n+1} - T_n$ is constant equal to some T under Palm. Then $X(t)$ and $Y(t)$ are uniform on $[0, T]$, the convolution has a positive density on $(0, 2T)$, whereas $Z(t)$ is constant equal to T . The answer is no; $X(t)$ and $Y(t)$ are not independent, in general.

CHAPTER 8

QUEUING THEORY FOR THOSE WHO CANNOT WAIT



Queuing phenomena are very frequent in computer and communication systems, and explain a large number of performance patterns. There is a large body of available results in queuing theory; in this chapter, we focus on results and concepts that are very broadly applicable, some of them are little known. We present four topics, which constitute a good coverage of all the techniques required in practice.

First, we start with simple, **deterministic** results; they provide results on transient phenomenons, and also some worst case bounds. These are often overlooked, but they do provide a first, sometimes sufficient, insight. Second we present **operational laws** for queuing systems; in some sense they are the “physical laws” of queuing: Little’s formula, the DASSA property, network and forced flows law. Here we make frequent use of Palm calculus (Chapter 7). These results also provide tools and bounds for fast analysis. Third, we give a series of simple, though important results for

single queues with one or several servers and for the processor sharing queue; these can be taken as models for systems without feedback. Fourth, we discuss **network of queues**, which can be used to model systems with feedback, and also complex interactions. Here we made the topic as simple as possible, but the result is not too simple, as there is some description complexity.

We give a unified treatment of queuing networks; we discussed items such as the MSCCC station, a powerful model for concurrency in hardware or software, or Whittle networks, which are used to model bandwidth sharing in the Internet. This latter type of network is traditionally presented as a type of its own, a non product form queuing network. We show that it must not be so: all of these are instances of the general theory of multi-class product form queuing networks. Presenting these results in this way simplifies the student's job, as there is a single framework to learn, instead of several disparate results. It is also more powerful as it provides new ways to combine existing building blocks.

Last, we illustrate on a example how the four different topics can be articulated and provide different insights on the same performance question.

Contents

8.1 Deterministic Analysis	239
8.1.1 Description of a Queuing System with Cumulative Functions	239
8.1.2 Reich's Formula	241
8.2 Operational Laws For Queuing Systems	242
8.2.1 Departures and Arrivals See Same Averages (DASSA)	243
8.2.2 Little's Law and Applications	244
8.2.3 Networks and Forced Flows	244
8.2.4 Bottleneck Analysis	246
8.3 Classical Results for a Single Queue	247
8.3.1 Kendall's Notation	247
8.3.2 The Single Server Queue	248
8.3.3 The Processor Sharing Queue, M/GI/1/PS	253
8.3.4 Single Queue with B Servers	254
8.4 Definitions for Queuing Networks	256
8.4.1 Classes, Chains and Markov Routing	256
8.4.2 Catalog of Service Stations	257
8.4.3 The Station Function	264
8.5 The Product-Form Theorem	269
8.5.1 Product Form	269
8.5.2 Stability Conditions	270
8.6 Computational Aspects	273
8.6.1 Convolution	273
8.6.2 Throughput	274
8.6.3 Equivalent Service Rate	276
8.6.4 Suppression of Open Chains	280

8.6.5	Arrival Theorem and MVA Version 1	281
8.6.6	Network Decomposition	284
8.6.7	MVA Version 2	288
8.7	What This Tells Us	290
8.7.1	Insensitivity	290
8.7.2	The Importance of Modelling Closed Populations	292
8.8	Mathematical Details About Product-Form Queuing Networks	293
8.8.1	Phase Type Distributions	293
8.8.2	Micro and Macro States	295
8.8.3	Micro to Macro: Aggregation Condition	295
8.8.4	Local Balance In Isolation	296
8.8.5	The Product Form Theorem	296
8.8.6	Networks with Blocking	297
8.9	Case Study	298
8.9.1	Deterministic Analysis	299
8.9.2	Single Queue Analysis	300
8.9.3	Operational Analysis	300
8.9.4	Queuing Network Analysis	302
8.9.5	Conclusions	303
8.10	Proofs	305
8.11	Review	307
8.11.1	Review Questions	307
8.11.2	Summary of Notation	308

8.1 DETERMINISTIC ANALYSIS

8.1.1 DESCRIPTION OF A QUEUING SYSTEM WITH CUMULATIVE FUNCTIONS

A deterministic analysis is often very simple, and provides first insights of a queuing system. Perhaps the simplest, and most efficient tool in this toolbox is the use of cumulative functions for arrival and departure counts, which we explain now. For a deeper treatment, see [55, 23].

Consider a system which is viewed as a black box. We make no specific assumptions about its operation; it may be a network node, an information system, etc. The cumulative functions are:

- $A(t)$ (*input function*): amount of work that arrives into the system in the time interval $[0, t]$
- $D(t)$ (*output function*): amount of work done in the time interval $[0, t]$

Assume that there is some time $t_0 \leq 0$ at which $A(t_0) = D(t_0) = 0$. We interpret t_0 as an instant at which the system is empty. The main observations are:

- $Q(t) := A(t) - D(t)$ is the backlog (unfinished work) at time t .
- Define $d(t) = \min \{u \geq 0 : A(t) \leq D(t+u)\}$ (horizontal deviation on Figure 8.1). If there is no loss of work (no incoming item is rejected) and if the system is first in, first out (**FIFO**), then $d(t)$ is the response time for a hypothetical atom of work that would arrive at time t .

The next example shows how this can be used for worst case analysis.

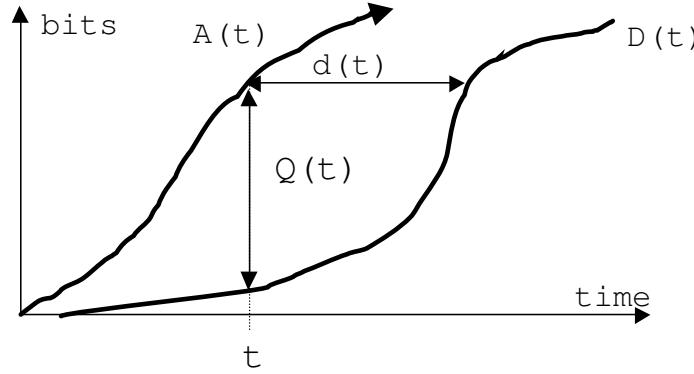


Figure 8.1: Use of cumulative functions to describe a queuing system.

EXAMPLE 8.1: PLAYOUT BUFFER. Consider a packet switched network that carries bits of information from a source with a constant bit rate r (Figure 8.2) as is the case for example, with circuit emulation. We have a first system S , the network, with input function $A(t) = rt$. The network imposes some variable delay, because of queuing points, therefore the output $A'()$ does not have a constant rate r . What can be done to re-create a constant bit stream ? A standard mechanism

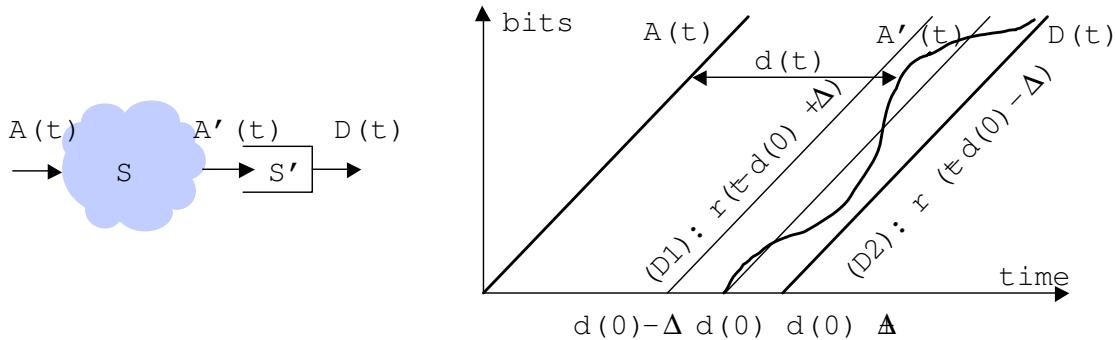


Figure 8.2: A Simple Playout Buffer Example

is to smooth the delay variation in a playout buffer. It operates as follows. When the first bit of data arrives, at time $d(0)$, it is stored in the buffer until some initial delay has elapsed. Then the buffer is served at a constant rate r whenever it is not empty. This gives us a second system S' , with input $A'()$ and output $D()$. What initial delay should we take ? We give an intuitive, graphical solution. For a formal development, see [55, Section 1.1.1].

The second part of Figure 8.2 shows that if the variable part of the network delay (called *delay jitter*) is bounded by some number Δ , then the output $A'(t)$ is bounded by the two lines (D1) and (D2). Let the output $D(t)$ of the playout buffer be the function represented by (D2), namely $D(t) = r(t - d(0)) - \Delta$. This means that we read data from the playout buffer at a constant rate r , starting at time $d(0) + \Delta$. The fact that $A'(t)$ lies above (D2) means that there is never underflow. Thus the playout buffer should delay the first bit of data by an amount equal to Δ , a bound on delay jitter.

QUESTION 8.1.1. *What is the required playout buffer size ?*¹

8.1.2 REICH'S FORMULA

This is a formula for describing the backlog in a single server queue. Consider a lossless, FIFO, system, with a constant service rate c , and with unlimited buffer size.

THEOREM 8.1 (*Reich*). *The backlog at time t in the system defined above is*

$$Q(t) = \max_{s \leq t} (A(t) - A(s) - c(t - s))$$

Note that, in the above, the value of s that reaches the maximum is the beginning of the busy period.

EXAMPLE 8.2: SCALING OF INTERNET DELAY. We are interested in knowing whether queuing delays are going to disappear when the Internet grows to broadband. The following analysis is due to Norros [74] and Kelly [45].

Assume traffic on an internet link grows according to three scale parameters: volume (v), speedup (s) and number of users (u). This is captured by the relation:

$$A(t) = v \sum_{i=1}^u A_i(st) \quad (8.1)$$

We are interested in the delay; assuming the link is a constant rate server with rate c , this is the backlog divided by c . We also assume that the capacity of the link is scaled with the increase in volume: $c = c_0 vsu$. The question is now: how does the delay depend on v, s, u ?

The maximum delay, $D(v, s, u)$ is derived from Reich's formula:

$$D(v, s, u) = \max_{t \geq 0} \left(\frac{A(t)}{c} - t \right)$$

The dependence on v and s is simple to analyze. It comes

$$D(v, s, 1) = \max_{t \geq 0} \left(\frac{v A_1(st)}{c} - t \right) = \max_{t \geq 0} \left(\frac{A_1(t)}{c_0 s} - \frac{t}{s} \right) = \frac{1}{s} D(1, 1, 1)$$

and similarly for $u \neq 1$ we have $D(v, s, u) = \frac{1}{s} D(1, 1, u)$. Thus the delay is independent of volume scaling, and is inversely proportional to the speedup factor s . The dependence on u requires more

¹A bound on buffer size is the vertical distance between (D1) and (D2); from Figure 8.2, we see that it is equal to $2r\Delta$.

assumptions. To go further, we assume a stochastic model, such that the queue length process $Q(t)$ is stationary ergodic. We can use Reich's formula:

$$Q(0) = \max_{t \geq 0} (A(-t) - ct)$$

where $A(-t)$ is now the amount of work that has arrived in the interval $[-t, 0]$. We assume that Eq.(8.1) continues to hold. Further, we model $A_i(-t)$ by a *fractional brownian traffic* [74]. This is a simplified model which captures long range dependence, i.e. the often observed property that the auto-correlation function does not decay exponentially. This means that

$$A_i(-t) = \lambda t + \sqrt{\lambda a} B_H^i(t)$$

where B_H^i is fractional brownian motion, λ the traffic intensity, and a a variance parameter. Fractional brownian motion is a gaussian process, with mean λt and variance $\lambda a t^{2H}$. $B_H(t)$ is self-similar in the sense that the process $B_H(kt)$ has the same distribution as $k^H B_H(t)$.

Assume that the A_i 's are independent. It follows from the properties of fractional brownian motion that $A(-t)$ is also fractional brownian traffic. Its mean is $u\lambda$ and its variance is $u\lambda a t^{2H}$, thus it has intensity $u\lambda$ and same variance parameter a .

By Reich's formula

$$D(1, 1, u) = \max_{t \geq 0} \left(\frac{A(t)}{c_0 u} - t \right) = \max_{t \geq 0} \left[\left(\frac{\lambda}{c_0} - 1 \right) t + \sqrt{\lambda a} B_H(t) \frac{1}{c_0 \sqrt{u}} \right]$$

Do the change of variable $t = k\tau$. It comes

$$D(1, 1, u) \sim \max_{\tau \geq 0} \left[\left(\frac{\lambda}{c_0} - 1 \right) k\tau + \sqrt{\lambda a} k^H B_H(\tau) \frac{1}{c_0 \sqrt{u}} \right]$$

where \sim means same distribution. Take k such that $k = \frac{k^H}{\sqrt{u}}$, i.e. $k = u^{-\frac{1}{2(1-H)}}$. Then we have

$$D(1, 1, u) \sim u^{-\frac{1}{2(1-H)}} D(1, 1, 1)$$

In summary, the delay scales according to

$$D(v, s, u) = \frac{1}{s u^b} D(1, 1, 1)$$

with $b = \frac{1}{2-2H}$. In practice, we expect the Hurst parameter usually lies in the range $[0.67, 0.83]$ thus $1.5 \leq b \leq 3$. In summary, delay decreases with speedup, and more rapidly with number of users.

8.2 OPERATIONAL LAWS FOR QUEUING SYSTEMS

These are robust results, i.e. which are true with very few assumptions on the queuing system. other than stability. Many of them directly derive from Chapter 7, such as the celebrated Little's law. The laws apply to a stationary system; for a single queue, they are true if the utilization is less than 1. This type of analysis was pioneered in [34]; an original, stand-alone treatment can be found in [35].

8.2.1 DEPARTURES AND ARRIVALS SEE SAME AVERAGES (DASSA)

THEOREM 8.2. (DASSA) Consider a system where individual customers come in and out. Assume that the arrival process A_n and the departure process D_n are stationary point processes, and that they have no point in common (thus there are no simultaneous arrivals or departures).

Let $N(t) \in \mathbb{N}$ be the number of customers present in the system at time t . Assume that $N(t)$, A_n and D_n are jointly stationary (see Section 7.2).

Then the probability distribution of $N(t)$ sampled just before an arrival is equal to the probability distribution of $N(t)$ sampled just after a departure.

The proof is given in Section 8.10; it is a direct application of the Rate Conservation law in Theorem 7.4.

EXAMPLE 8.3: INTER-DEPARTURE TIME IN M/GI/1 QUEUE. We want to compute the distribution of inter-departure time in the stable M/GI/1 queue defined in Section 8.3 (i.e. the single server queue, with Poisson arrival and general service time distribution), and would like to know in which case it is the same as the inter-arrival distribution.

First note that the time between two departures is equal to one service time if the first departing customer leaves the system non-empty, and, otherwise, the same plus the time until the next arrival. The time until next arrival is independent of the state of the system and is exponentially distributed, with parameter the arrival rate λ . Thus the *Laplace-Stieltjes transform*² of the inter-departure time is

$$\mathcal{L}_D(s) = (1 - p)\mathcal{L}_S(s) + p\mathcal{L}_S(s)\frac{\lambda}{\lambda + s}$$

where \mathcal{L}_S is the Laplace-Stieltjes transform of the service time and p is the probability that a departing customer leaves the system empty.

By DASSA, p is also the probability that an arriving customer sees an empty system. By PASTA (Example 7.19), it is equal to the probability that the queue is empty at an arbitrary point in time, which is also equal to $1 - \rho$, with $\rho = \lambda\bar{S}$ and \bar{S} = mean service time. Thus

$$\mathcal{L}_D(s) = \mathcal{L}_S(s) \left(\rho + \frac{(1 - \rho)\lambda}{\lambda + s} \right)$$

which entirely defines the probability distribution of inter-departure times.

The inter-departure times have the same distribution as the inter-arrival times if and only if $\mathcal{L}_D(s) = \lambda/(\lambda + s)$. Solving for \mathcal{L}_S gives $\mathcal{L}_S(s) = \frac{\lambda/\rho}{\lambda/\rho + s}$, i.e. the service time must be exponentially distributed and the M/GI/1 queue must be an M/M/1 queue.

²The Laplace-Stieltjes transform of a non-negative random variable X is defined by $\mathcal{L}_X(s) = \mathbb{E}(e^{-sX})$. If X and Y are independent, $\mathcal{L}_{X+Y}(s) = \mathcal{L}_X(s)\mathcal{L}_Y(s)$; X is exponentially distributed with parameter λ if and only if $\mathcal{L}_X(s) = \frac{\lambda}{\lambda+s}$.

8.2.2 LITTLE'S LAW AND APPLICATIONS

THEOREM 8.3 (Operational Law). Consider a stationary system that is visited by a flow of customers (for a formal definition, see Theorem 7.6).

- **[Throughput]** The throughput, defined as the expected number of arrivals per second, is also equal to the inverse of the expected time between arrivals.
- **[Little]**

$$\lambda \bar{R} = \bar{N}$$

where λ is the expected number of customers arriving per second, \bar{R} is the expected response time seen by an arbitrary customer and \bar{N} is the expected number of customers observed in the system at an arbitrary time

- **[Utilization Law]** If the system is a single server queue with arrival rate λ and expected service time \bar{S} :

$$\mathbb{P}(\text{server busy}) = \rho := \lambda \bar{S}$$

If it is an s -server queue:

$$\mathbb{E}(\text{number of busy servers}) = s\rho$$

$$\text{with } \rho := \frac{\lambda \bar{S}}{s}.$$

QUESTION 8.2.1. Consider a single server queue that serves only one customer at a time. What is the average number of customers not in service (i.e. in the waiting room ?)³

THE INTERACTIVE USER MODEL The interactive user model is illustrated in Figure 8.3. n users send jobs to a service center. The **think time** is defined as the time between jobs sent by one user. Call \bar{R} the expected response time for an arbitrary job at the service center, \bar{Z} the expected think time and λ the throughout of the system. A direct application of Little's law to the entire system gives:

THEOREM 8.4 (Interactive User).

$$\lambda(\bar{Z} + \bar{R}) = n$$

EXAMPLE 8.4: SERVICE DESK. A car rental company in a large airport has 10 service attendants. Every attendant prepares transactions on its PC and, once completed, send them to the database server. The software monitor finds the following averages: one transaction every 5 seconds, response time = 2 s. Thus the average think time is 48 s.

8.2.3 NETWORKS AND FORCED FLOWS

We often find systems that can be modeled as a directed graph, called a network. We consider models of the form illustrated on Figure 8.4. If the total number of customers is constant, the

³ $\bar{N}_w = \bar{N} - \rho$, this follows from items 2 and 3 in Theorem 8.3.

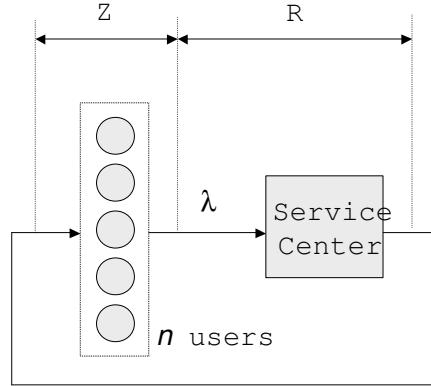


Figure 8.3: The Interactive User Model

network is called “closed”, otherwise “open”. In Section 8.4, we will study such networks in more detail.

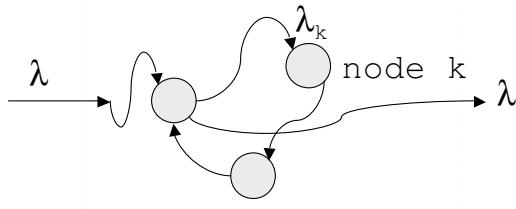


Figure 8.4: Network Model

THEOREM 8.5 (Network Laws). Consider a stationary network model where λ is the total arrival rate.

- **[Forced Flows]** $\lambda_k = \lambda V_k$, where λ_k is the expected number of customers arriving per second at node k and V_k is the expected number of visits to node k by an arbitrary customer during its stay in the network.
- **[Total Response Time]** Let \bar{R} [resp. \bar{R}_k] be the expected total response time \bar{R} seen by an arbitrary customer [resp. by an arbitrary visit to node k].

$$\bar{R} = \sum_k \bar{R}_k V_k$$

EXAMPLE 8.5: Transactions on a database server access the CPU, disk A and disk B (Figure 8.5). The statistics are: $V_{\text{CPU}} = 102$, $V_A = 30$, $V_B = 68$ and $\bar{R}_{\text{CPU}} = 0.192 \text{ s}$, $\bar{R}_A = 0.101 \text{ s}$, $\bar{R}_B = 0.016 \text{ s}$

The average response time for a transaction is 23.7 s .

8.2.4 BOTTLENECK ANALYSIS

Common sense and the guidelines in Chapter 1 tell us to analyze bottlenecks first. Beyond this, simple performance bounds in stationary regime can be found by using the so-called bottleneck analysis. It is based on the following two observations:

1. waiting time is ≥ 0
2. a server utilization is bounded by 1

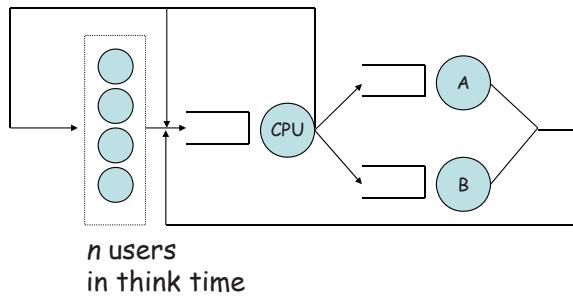


Figure 8.5: Network example used to illustrate bottleneck analysis. Each transaction uses CPU, disk A or disk B. Av. numbers of visits per transaction: $V_{\text{CPU}} = 102, V_A = 30, V_B = 17$; av. service time per transaction: $\bar{S}_{\text{CPU}} = 0.004 \text{ s}, \bar{S}_A = 0.011 \text{ s}, \bar{S}_B = 0.013 \text{ s}$; think time $Z = 1 \text{ s}$.

We illustrate the method on Figure 8.5. The network is a combination of Figure 8.3 and Figure 8.4. Transactions are issued by a pool of n customers which are either idle (in think time) or using the network. In addition, assume that every network node is a single server queue, and let \bar{S}_k be the average service time per visit at node k . Thus $\bar{R}_k - \bar{S}_k$ is the average waiting time per visit at node k . The throughput λ is given by the interactive user model:

$$\lambda = \frac{n}{Z + \sum_k V_k \bar{R}_k} \quad (8.2)$$

and by forced flows, the utilization of the server at node k is $\rho_k = \lambda V_k \bar{S}_k$. Applying the two principles above gives the constraints on λ :

$$\left\{ \begin{array}{l} \lambda \leq \frac{n}{Z + \sum_k V_k \bar{S}_k} \\ \lambda \leq \frac{1}{\max_k V_k \bar{S}_k} \end{array} \right. \quad (8.3)$$

Similarly, using Eq.(8.2) and Eq.(8.3), we find the following constraints on the response time $\bar{R} = \sum_k V_k \bar{R}_k$:

$$\left\{ \begin{array}{l} \bar{R} \geq \sum_k V_k \bar{S}_k \\ \bar{R} \geq n (\max_k V_k \bar{S}_k) - Z \end{array} \right. \quad (8.4)$$

Figure 8.6 illustrates the bounds. See also Figure 8.15.

A node k that maximizes $V_k \bar{S}_k$ is called, in this model, a **bottleneck**. To see why a bottleneck determines the performance, consider improving the system by decreasing the value of $V_k \bar{S}_k$ (by

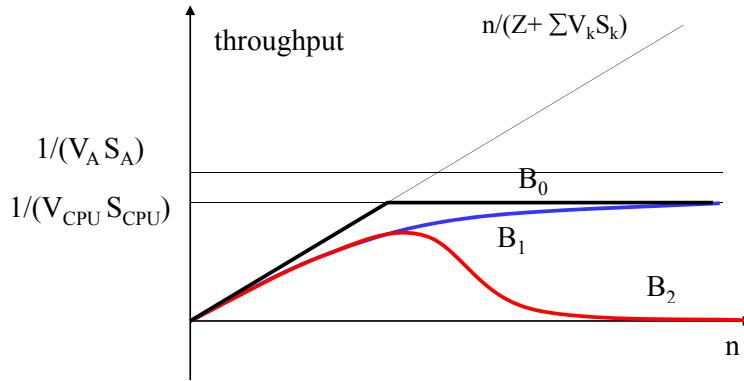


Figure 8.6: Throughput bound (B_0) obtained by bottleneck analysis for the system in Figure 8.5, as a function of the number of users n . B_1 , B_2 : typical throughput values for a system without [resp. with] congestion collapse.

reducing the number of times the resource is used, or by replacing the resource by a faster one). If k is not a bottleneck, this does not affect asymptote on Figure 8.6, and only marginally increases the slope of the bound at the origin, unlike if k is a bottleneck. On Figure 8.6, we see that the bottleneck is the CPU.

Among the two bounds in Eq.(8.3), the former is accurate at low load (when there is no queuing), and the latter is expected to be true at high load (when the bottleneck is saturated). This is what makes bottleneck analysis appealing, as the two bounds cover both ends of the spectrum. Note however that, at high loads, congestion collapse might occur, and then performance would be worst than predicted by the bound.

QUESTION 8.2.2. *What happens to the example of Figure 8.5 if the CPU processing time is reduced from 0.004 to 0.003 ? to 0.002 ?⁴*

8.3 CLASSICAL RESULTS FOR A SINGLE QUEUE

The single queue has received much attention, and there are analytical results available for a large class of systems with random arrivals and service. We give here a minimal, but useful set of result. For more details on some topics, the classical reference is [46, 47]; a more compact and up to date textbook is [71]. We start with some notation and a generic result.

8.3.1 KENDALL'S NOTATION

The classical notation for a queue, in its simplest form, is of the type $A/S/s/K$ where:

- A (character string) describes the type of arrival process: G stands for the most general arrival process, $A = GI$ means that the arrival process is a point process with iid interarrival times, M is for a Poisson arrival process.

⁴The disk A becomes the bottleneck. Decreasing the CPU processing time to 0.002 does not improve the bound significantly.

- S (character string) describes the type of service process: G for the most general service process, $S = \text{GI}$ means that the service times are iid and independent of the arrival process, $S = \text{M}$ is the special case of GI with exponential service times, $S = \text{D}$ with constant service times.
- B and K are integers representing the number of servers and the capacity (maximum number of customers allowed in the system, queued + in service). When $K = \infty$, it may be omitted.
- Let A_n be the arrival time and S_n the service time of the n th customer, labeled in order of arrival. We assume that the sequence (A_n, S_n) is stationary with respect to the index n and that it can be interpreted as a stationary marked point process (i.e. the expectation of $A_{n+1} - A_n$ is finite, see Theorem 7.9).
- The service discipline is by default FIFO, otherwise it is mentioned explicitly.

8.3.2 THE SINGLE SERVER QUEUE

STABILITY

We consider the most general queue with one single server and with infinite capacity. Note that we do not assume Poisson arrivals, and we allow service times to depend on the state of the system. We assume that the system is work conserving. More precisely, let $W(t)$ be the backlog process, i.e. the sum of the service times of all customers that are present in the system at time t . When a customer arrives, $W(t)$ increases by the (future) service time of this customer. The work conserving assumption means that $W(t)$ decreases at rate 1 over any time interval such that $W(t) > 0$.

An important issue in the analysis of the single server queue is stability. In mathematical terms, it means whether the backlog $W(t)$ is stationary. When the system is unstable, a typical behaviour is that the backlog grows to infinity.

The following is the general stability condition for the single server queue. Let \bar{S} be the expectation of the service time, λ the intensity of the arrival process (expected number of arrivals per second) and $\rho = \lambda \bar{S}$ the utilization factor.

THEOREM 8.6. (*Loynes [3, Thm 2.1.1]*)

If $\rho < 1$ the backlog process has a unique stationary regime. In the stationary regime, the queue empties infinitely often.

Furthermore, for any initial condition, the waiting time of the n th customer converges in distribution as $n \rightarrow \infty$ to the waiting time for an arbitrary customer computed in the stationary regime.

If $\rho > 1$ the backlog process has no stationary regime.

A heuristic explanation for the necessary condition is that, if the system is stable, all customers eventually enter service, thus the mean number of beginnings of service per second is λ . From Little's law applied to the server (see Section 8.2), we have $\rho =$ the probability that the server is busy, which is ≤ 1 . For $\rho = 1$ there may or may not be stability, depending on the specific queue. Be careful that this intuitive stability result holds only for a single queue. For networks of interconnected queues, there is no such general result, as discussed in Section 8.4. The theorem is for a queue with infinite capacity. For a finite capacity queue, there is, in general, stability for any value of ρ (but for $\rho > 1$ there must be losses).

QUESTION 8.3.1. Consider a queuing system of the form $G/G/1$ where the service time S_n of

customer n is equal to the inter-arrival time $A_{n+1} - A_n$. What are the values of ρ and of the expected number of customers \bar{N} ?⁵

QUESTION 8.3.2. Give an example of stable and of an unstable single server queue with $\rho = 1$.⁶

M/GI/1 QUEUE The arrival process is Poisson with parameter λ and the service times and independent of each other and of the arrival process, with a general distribution. For $\rho < 1$ the queue is stable and for $\rho \geq 1$ is unstable. Using the rate conservation law as in Example 7.10, we obtain the Laplace Stieltjes transform of the waiting time (*Pollaczek-Khinchine formula for transforms*):

$$\mathcal{L}_W(s) = \frac{s(1-\rho)}{s - \lambda + \lambda \mathcal{L}_S(s)} \quad (8.5)$$

where \mathcal{L}_S is the Laplace Stieltjes transform of the service time. Note that, by PASTA, the waiting time has the same distribution as the workload sampled at an arbitrary point in time.

QUESTION 8.3.3. Give the Laplace Stieltjes transform \mathcal{L}_R of the response time.⁷

The distribution of the number of customers $N(t)$, at an arbitrary point in time, is obtained by first computing the distribution of the number of customers seen at a departure times, and then using DASSA ([46, Section 5.6]). The distribution is known via its *z-transform*⁸:

$$G_{N(t)}(z) = (1-\rho)(1-z) \frac{\mathcal{L}_S(\lambda - \lambda z)}{\mathcal{L}_S(\lambda - \lambda z) - z} \quad (8.6)$$

(this formula is also called a *Pollaczek-Khinchine formula for transforms*). The mean values of number of customers in system or in waiting room and the mean response times and waiting times are easily derived and are given below:

$$\begin{cases} \bar{N} = \frac{\rho^2 \kappa}{1-\rho} + \rho \text{ with } \kappa = \frac{1}{2} \left(1 + \frac{\sigma_S^2}{\bar{S}^2} \right) = \frac{1}{2} (1 + \text{CoV}_S^2) \\ \bar{N}_w = \frac{\rho^2 \kappa}{1-\rho} \\ \bar{R} = \frac{\bar{S}(1-\rho(1-\kappa))}{1-\rho} \\ \bar{W} = \frac{\rho \bar{S} \kappa}{1-\rho} \end{cases} \quad (8.7)$$

Note the importance of the coefficient of variation (CoV) of the service time.

QUESTION 8.3.4. Which of the quantities $\bar{N}, \bar{N}_w, \bar{R}, \bar{W}$ are Palm expectations?⁹

M/M/1 QUEUE This is a special case of the M/GI/1 queue where the service times are exponentially distributed. Here it is possible to obtain all stationary probabilities in explicit (and simple) form, by directly solving the equilibrium equations of the Markov process. One finds that the distribution of the number of customers at an arbitrary point in time is, when $\rho < 1$:

$$\mathbb{P}(N(t) = k) = (1-\rho)\rho^k \quad (8.8)$$

⁵ $\lambda = \frac{1}{S}$ thus $\rho = 1$. There is always exactly one customer in the queue. Thus $\bar{N} = 1$.

⁶ The example in Question 8.3.1 is stable with $\rho = 1$. The M/M/1 queue with $\rho = 1$ is unstable.

⁷ The response time is the sum of the service time and the waiting time, and they are independent. Thus $\mathcal{L}_R(s) = \mathcal{L}_S(s)\mathcal{L}_W(s)$.

⁸ The *z-transform*, $G_N(z)$ of an integer random variable N is defined by $G_N(z) = \mathbb{E}(z^N)$.

⁹ \bar{R}, \bar{W}

and the distribution of the service time of an arbitrary customer is given by

$$\mathbb{P}^0(R_0 \leq x) = 1 - e^{-(1-\rho)\frac{x}{S}} \quad (8.9)$$

Furthermore, Eq.(8.7) applies with $\kappa = 1$.

M/M/1/K QUEUE This is a modification of the M/M/1 queue where the total number of customers is limited to K . If a customer arrives when the queue is fulled, it is dropped. The M/GI/1 formulas cannot be applied, but, instead, one can directly solve the equilibrium equations of the Markov process.

The system has a stationary regime for *any* value of ρ . The distribution of the number of customers at an arbitrary point in time is

$$\begin{aligned} \mathbb{P}(N = k) &= \eta \rho^k \mathbf{1}_{\{0 \leq k \leq K\}} \\ \text{with } \eta &= \frac{1 - \rho}{1 - \rho^{K+1}} \text{ if } \rho \neq 1, \quad \eta = \frac{1}{K + 1} \text{ if } \rho = 1 \end{aligned}$$

By PASTA, the probability that the system is full is equal to the loss probability and is

$$\mathbb{P}^0(\text{arriving customer is discarded}) = \mathbb{P}(N(t) = K) = \frac{(1 - \rho)\rho^K}{1 - \rho^{K+1}}$$

GI/GI/1 QUEUE This is the general single server queue where inter-arrival and service times are independent of each other and are i.i.d. In general, no closed form solution exists, but numerical procedures are available.

One approach is based on the following equation, which is a stochastic recurrence:

$$W_n = (W_{n-1} + S_{n-1} - A_n + A_{n-1})^+$$

where the notation $(x)^+$ means $\max(x, 0)$ and $W_n = W(A_n^-)$ is the workload in the system just before the n th arrival, i.e. the waiting time for the n th customer (here A_n is the arrival time and S_n the service time of the n th customer). Let $C_n = A_n - A_{n-1} + S_n$. Note that C_n is i.i.d. and independent of W_{n-1} thus

$$W_n \stackrel{\text{distrib}}{=} (W_{n-1} - C_n)^+ \quad (8.10)$$

If $\rho < 1$ the system has a stationary regime, and the stationary distribution of waiting time W must satisfy

$$W \stackrel{\text{distrib}}{=} (W - C)^+ \quad (8.11)$$

where C is a random variable with same distribution as $A_n - A_{n-1} + S_n$. This equation is called **Lindley's equation**. It is classical to use CDFs, which gives the following equivalent form of Eq.(8.11):

$$F_W(x) = \begin{cases} 0 & \text{if } x < 0 \\ \int_{-\infty}^x F_W(x-y) f_C(y) dy & \text{otherwise} \end{cases} \quad (8.12)$$

where F_W is the CDF of waiting times and f_C is the PDF of $A_n - A_{n-1} + S_n$. Eq.(8.11) is an equation of the Wiener-Hopf type and can be solved, at least in many cases, using the theory of analytical functions; see [46, Section 8.2].

A second approach consists in solving Eq.(8.10) directly by discretization. Pick a time step δ and let, for $n \in \mathbb{N}$ and $k \in \mathbb{Z}$

$$w_k^n = \mathbb{P}(W^n \in [k\delta, (k+1)\delta)) \quad (8.13)$$

$$s_k = \mathbb{P}(S_n \in [k\delta, (k+1)\delta)) \quad (8.14)$$

$$a_k = \mathbb{P}(-A_n + A_{n-1} \in [k\delta, (k+1)\delta)) \quad (8.15)$$

Note that $w_k = s_k = 0$ for $k < 0$ and $a_k = 0$ for $k > 0$ and that the arrays s and a are independent of n . Eq.(8.10) can be approximated by

$$\begin{cases} w_k^n = (w^{n-1} * s * a)_k & \text{if } k > 0 \\ w_0^n = \sum_{i \leq 0} (w^{n-1} * s * a)_i \\ w_k^n = 0 & \text{if } k < 0 \end{cases} \quad (8.16)$$

where $*$ is the discrete convolution. The error we are making is due to discretization and should decrease with δ . In fact, Eq.(8.16) is exact for the modified system where we replaced the service times inter-arrival times by approximations that are multiples of δ ; such an approximation is by default for the service time, Eq.(8.14), and by excess for the inter-arrival time, Eq.(8.15); thus the approximating system has a ρ value less than the original system. If the original system is stable, so is the approximating one, and by Loynes' theorem, the iteration converges to the stationary distribution of waiting time. The method thus consists in numerically evaluating Eq.(8.16) until the norm of the difference $w^n - w^{n-1}$ becomes small; the convolution can be computed using the fast Fourier transform. See [39] for an example where this method is used.

A third type of methods uses mixtures of exponentials to approximate the distributions of inter-arrival and service times as in Section 8.8.1. Then the stationary distributions can be computed explicitly; see [52, 72].

WHAT THIS TELLS US

Though most practical systems are unlikely to exactly fit the assumptions of any of the models in this section, the analytical formulas do explain patterns that are observed in practice. The models in this section are for systems without feedback, since the arrival process is not influenced by the state of the queuing system. Important features of such systems are:

- **Non Linearity of Response Time:** At low values of the utilization factor ρ , the response time tends to increase slowly, and linearly with ρ . In contrast, as ρ approaches 1, the response time grows to ∞ (Figure 8.7). Thus the impact of a small traffic increase is dramatically different, depending on the initial value of the utilization factor.
QUESTION 8.3.5. *What happens for the system in Figure 8.7 if the traffic volume increases by 20%?*¹⁰
- **Variability Considered Harmful:** The Pollacezk-Khinchine formula for the mean in Eq.(8.7) shows that response time and queue sizes increase with the variability of the service time. See also Figure 8.8.

¹⁰The system becomes unstable $\rho > 1$; in practice it will lose requests, or enter congestion collapse.

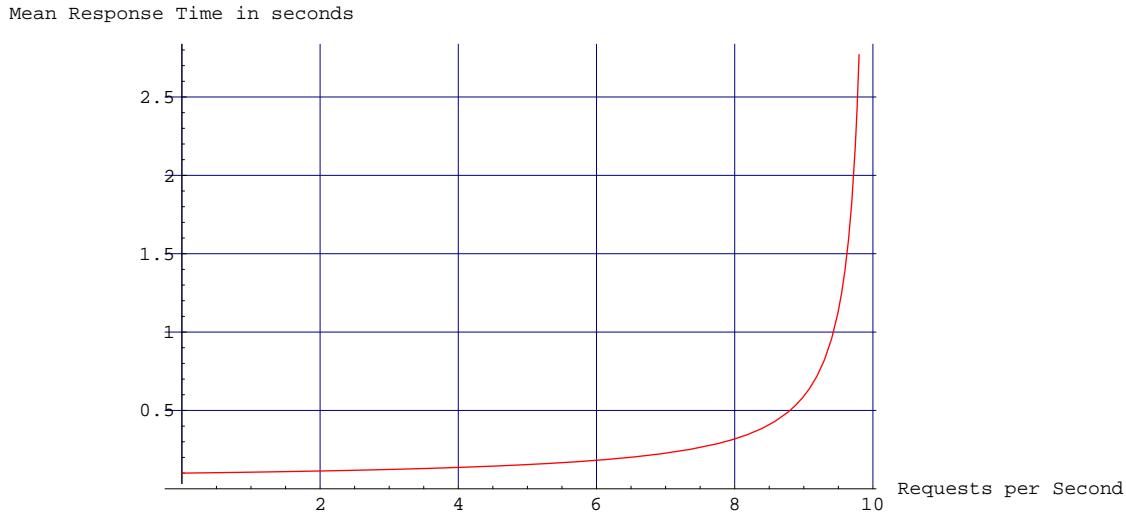


Figure 8.7: Average response time versus requests per second for a database server modeled as M/GI/1 queue. The time needed to process a request is 0.1 second and its standard deviation is estimated to 0.03. The maximum load that can be served if an average response time of 0.5 second is considered acceptable is 8.8 requests per second. If the traffic volume increases by 10%, the response time becomes 1.75, thus is multiplied by a factor of 3.5.

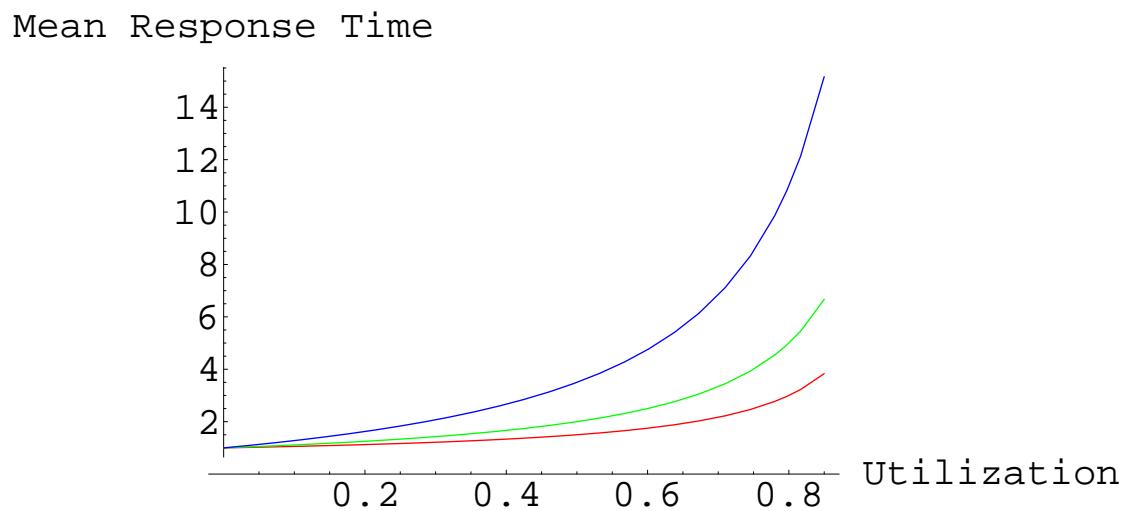


Figure 8.8: Mean response time for M/GI/1 queue, relative to service time, for different values of coefficient of variation $\text{CoV}_S = \frac{\sigma_S}{\bar{S}}$: from top to bottom: $\text{CoV}_S = 1.4$, $\text{CoV}_S = 1$ (M/M/1 queue) and $\text{CoV}_S = 0$ (M/D/1 queue).

8.3.3 THE PROCESSOR SHARING QUEUE, M/GI/1/PS

This is a special case of the single server queue, with the *Processor Sharing (PS)* service discipline instead of FIFO. Here we assume that the server divides itself equally into all present customers; this is an idealization when $\delta \rightarrow 0$ of the *round robin* service discipline, where the server allocates time slices of duration δ in turn to each present customer. If there are N customers in the queue, the residual service time for each of them decreases at a rate $1/N$. This is also called *egalitarian processor sharing*. Loynes's theorem applies and the system is stable when $\rho < 1$.

The workload process $W(t)$ is the same as for FIFO queues, but the distribution of waiting times and of customers is not the same. We give results for the simple case where arrival are Poisson and service times are i.i.d. and independent of the arrival process. They are both simple and striking. We assume $\rho < 1$. First, the stationary probability is [92]:

$$\mathbb{P}(N(t) = k) = (1 - \rho)\rho^k \quad (8.17)$$

which shows in particular that it depends on the service time distribution only through its mean (this *insensitivity* property is common to many queues in the theory of networks presented in Section 8.4). It follows that

$$\begin{cases} \bar{N} = \frac{\rho}{1-\rho} \\ \bar{R} = \frac{\bar{S}}{1-\rho} \end{cases} \quad (8.18)$$

Second, the average response time R_0 of an arbitrary customer, conditional to its service time S_0 satisfies [47]

$$\mathbb{E}^0(R_0 | S_0 = x) = \frac{x}{1 - \rho} \quad (8.19)$$

i.e. it is as if an arbitrary customer sees a server for herself alone, but with a rate reduced by the factor $1/(1 - \rho)$. Eq.(8.18) and Eq.(8.19) can be simply deduced from results in Section 8.4 if the distribution of service times can be decomposed as a mixture of exponentials; see [100]. Eq.(8.17) is a special case of results for product-form queuing networks, see Section 8.4.

WHAT THIS TELLS US

Compare the M/M/1 and M/M/1/PS queues, where it is implicit that the M/M/1 queue is FIFO. The stationary distribution of numbers of customers are identical, therefore (by Little's law) the mean response times are identical, too. However, the conditional mean response time, given the service time, are very different. For M/M/1/PS, it is given by Eq.(8.19). For the M/M/1 queue, the response time is the sum of waiting time plus service time, and the waiting time is independent of the service time. The mean waiting time is given in Eq.(8.7) with $\kappa = 1$, therefore, for the FIFO queue:

$$\mathbb{E}^0(R_0 | S_0 = x) = x + \frac{\rho \bar{S}}{1 - \rho} \quad (8.20)$$

Figure 8.9 plots the conditional response time for both FIFO and PS queues, and several values of x .

PS and FIFO have the same capacity and the same mean response time. However, the PS queue penalizes customers with a large service time, and the penalty is proportional to the service time. This is often considered as a *fairness* property of the PS service discipline.

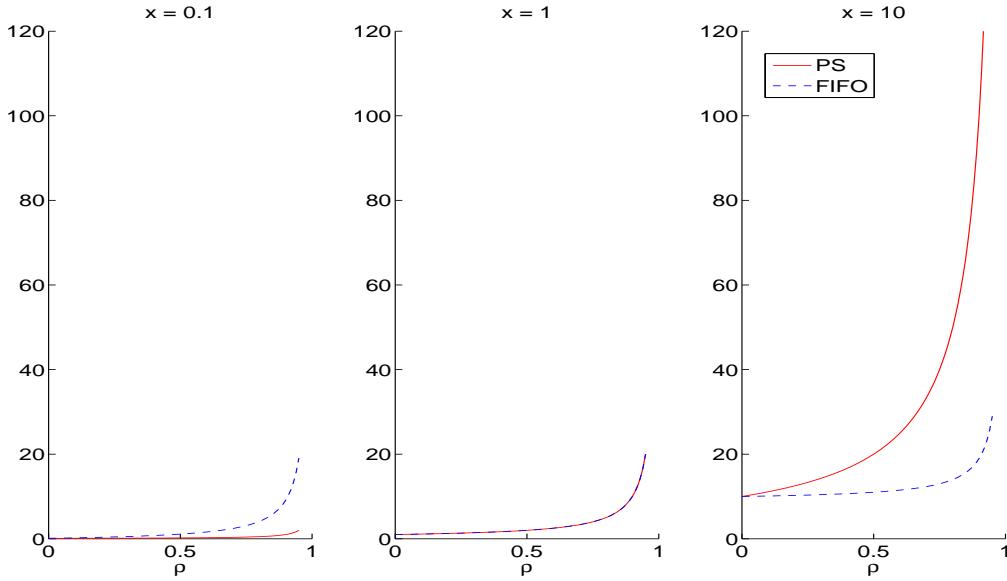


Figure 8.9: Expected response time given that the service time of this customer is x versus utilization ρ , for M/M/1 queues with FIFO (dashed) and PS (plain) service disciplines, for various values of x . Mean service time \bar{S} is 1 time unit

QUESTION 8.3.6. *For which value of the service time x are the expected response times for M/M/1 and M/M/1/PS equal ?¹¹*

8.3.4 SINGLE QUEUE WITH B SERVERS

The multiple server queue is defined by the fact that at most B customers can be served in parallel. Thus, the workload process decreases at a rate equal to $\min(N(t), 1)$ where $N(t)$ is the number of customers present in the queue. The utilization ρ is now defined by $\rho = \frac{\lambda \bar{S}}{B}$. The stability condition is less easy than for single server queues. When $\rho < 1$ there is a stationary regime but it may not be unique [3, 2.3]. When $\rho > 1$ there is no stationary regime.

M/M/B QUEUE

For more specific system, one can say more. A frequently used system is the M/M/B queue, i.e. the system with Poisson arrivals, B servers, exponential service times and FIFO discipline. The system can be studied directly by solving for the stationary probability. Here when $\rho < 1$ there is a unique stationary regime, which is also reached asymptotically when we start from arbitrary initial conditions; for $\rho \geq 1$ there is no stationary regime.

When $\rho < 1$ the stationary probability is given by

$$\mathbb{P}(N(t) = k) = \begin{cases} \eta \frac{(B\rho)^k}{k!} & \text{if } 0 \leq k \leq B \\ \eta \frac{B^B \rho^k}{B!} & \text{if } k > B \end{cases} \quad (8.21)$$

¹¹When the service time x is equal to the mean service time \bar{S} .

$$\text{with } \eta^{-1} = \sum_{i=0}^{B-1} \frac{(B\rho)^i}{i!} + \frac{(B\rho)^B}{B!(1-\rho)}$$

and the stationary CDF of the waiting time for an arbitrary customer is

$$\begin{aligned} \mathbb{P}^0(W_0 \leq x) &= 1 - pe^{-B(1-\rho)\frac{x}{S}} \\ \text{with } p &= \frac{1-u}{1-\rho u} \text{ and } u = \frac{\sum_{i=0}^{B-1} \frac{(B\rho)^i}{i!}}{\sum_{i=0}^B \frac{(B\rho)^i}{i!}} \end{aligned}$$

The probability of finding all servers busy at an arbitrary point in time or at a customer arrival is ([Erlang-C formula](#)):

$$\mathbb{P}(\text{all servers busy}) = \mathbb{P}(N(t) \geq B) = p \quad (8.22)$$

Average quantities can easily be derived:

$$\left\{ \begin{array}{l} \bar{N} = \frac{p\rho}{1-\rho} + B\rho \\ \bar{N}_w = \frac{p\rho}{1-\rho} \\ \bar{R} = \frac{pS}{B(1-\rho)} + \bar{S} \\ \bar{W} = \frac{pS}{B(1-\rho)} \end{array} \right.$$

M/GI/B/B QUEUE This is the system with Poisson arrivals, B servers, arbitrary (but independent) service times and no waiting room. An arriving customer that finds all B servers busy is dropped.

The system is stable for any value of ρ and the stationary probability of the number of customers is given by

$$\mathbb{P}(N(t) = k) = \eta \mathbf{1}_{\{0 \leq k \leq B\}} \frac{(B\rho)^k}{k!} \text{ with } \eta^{-1} = \sum_{k=0}^B \frac{(B\rho)^k}{k!}$$

The probability that an arriving customer is dropped is ([Erlang Loss Formula](#), or [Erlang-B Formula](#)):

$$\mathbb{P}^0(\text{arriving customer is dropped}) = \mathbb{P}(N(t) = B) = \eta \frac{(B\rho)^B}{B!} \quad (8.23)$$

WHAT THIS TELLS US

The simple M/M/B model can be used to understand the benefit of load sharing. Consider the systems illustrated in Figure 8.10.

Assume processing times and job inter-arrival times can be modeled as independent iid exponential sequences. Thus the first [resp. second] case is modeled as one M/M/2 queue [resp. a collection of two parallel M/M/1 queues]. Assume load is balanced evenly between the two processors. Both systems have the same utilization ρ . The mean response for the first system is obtained from Section 8.3.4; we obtain $\frac{\bar{S}}{1-\rho^2}$. For the second system it is simply $\frac{\bar{S}}{1-\rho}$ (Figure 8.10).

We see that for very small loads, the systems are similar, as expected. In contrast, for large loads, the response time for the first system is much better, with a ratio equal to $1 + \rho$. For example, for $\rho = 0.5$, the second system has a response time 1.5 times larger.

However, the capacity is the same for both systems: the benefit of load sharing may be important in terms of response time, but does not change the capacity of the system.

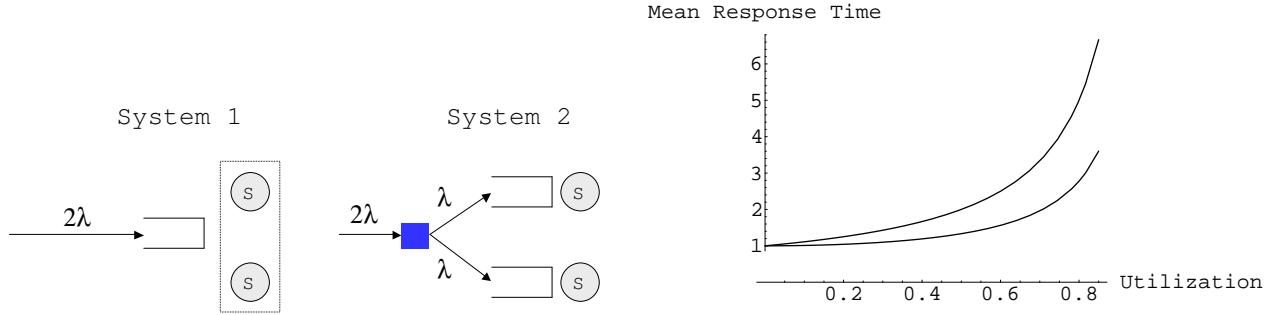


Figure 8.10: Mean response time over service time for systems 1 (bottom) and 2 (top), versus utilization factor ρ .

8.4 DEFINITIONS FOR QUEUING NETWORKS

Realistic models of information and communications systems involve interconnected systems, which can be captured by a queuing networks. In general, not much can be said about a queuing network. Even the stability conditions are not known in general, and there is no equivalent of Loynes' theorem for networks. Indeed, the natural condition that the utilization factor is less than 1 is necessary for stability but may not be sufficient – see [16] for an example of a multi-class queuing network, with FIFO queues, Poisson arrivals and exponential service times, which is unstable with arbitrarily small utilization factor.

Fortunately, there is a broad class of queuing networks, the so called *multi-class product form queuing networks* for which there are simple and exhaustive results, given in this and the following section. These networks have the property that their stationary probability has product form. They were developed as *BCMP networks* in reference to the authors of [10] or *Kelly networks* in reference to [44]. When there is only one class of customers they are also called *Jackson* networks in the open case [42] and Gordon and Newell networks in the closed case [37]. For a broader perspective on this topic, see the recent books [94] and [24]. This latter reference presents in particular extensions to other concepts, including the “negative customers” introduced in [36]. A broad treatment, including approximate analysis for non product form queuing networks can also be found in [101].

We now give the common assumptions required by multi-class product form queuing networks (we defer a formal definition of the complete process that describes the network to Section 8.8).

8.4.1 CLASSES, CHAINS AND MARKOV ROUTING

We consider a network of queues, labeled $s = 1, \dots, S$, also called *stations*. Customers visit stations and queue or receive service according to the particular station service discipline, and once served, move to another station or leave the network. Transfers are instantaneous (delays must be modeled explicitly by means of delay stations, see below).

Every customer has an attribute called *class*, in a finite set $\{1, \dots, C\}$. A customer may change class in transit between stations, according to the following procedure (called *Markov routing*

in [103]). There is a fixed non-negative *routing matrix* $Q = \left(q_{c,c'}^{s,s'} \right)_{s,s',c,c'}$ such that for all s, c : $\sum_{s',c'} q_{c,c'}^{s,s'} \leq 1$. When a class- c customer leaves station s (because her service time is completed), she does a random experiment such that: with probability $q_{c,c'}^{s,s'}$ she joins station s' with class c' ; with probability $1 - \sum_{s',c'} q_{c,c'}^{s,s'}$ she leaves the network. This random experiment is performed independently of all the past and present states of the network. In addition, there are fresh independent Poisson arrivals, also independent of the past and present states of the network; ν_c^s is the intensity of the Poisson process of arrivals of class- c customers at station s . We allow $\nu_c^s = 0$ for some or all s and c .

We say that two classes c, c' are *chain equivalent* if $c = c'$ or if it is possible for a class c -customer to eventually become a class c' customer, or vice-versa. This defines an equivalence relation between classes, the equivalence classes are called *chains*. It follows that a customer may change class but always remains in the same chain.

A chain \mathcal{C} is called *closed* if the total arrival rate of customers $\sum_{c \in \mathcal{C}, s} \lambda_c^s$ is 0. In such a case we require that the probability for a customer of this chain to leave the network is also 0, i.e. $\sum_{c',s'} q_{c,c'}^{s,s'} = 1$ for all $c \in \mathcal{C}$ and all s . The number of customers in a closed chain is constant.

A chain that is not closed is called *open*. We assume that customers of an open chain cannot cycle forever in the network, i.e. every customer of this chain eventually leaves the network.

A network where all chains are closed is called a *closed network*, one where all chains are open is called an *open network* and otherwise it is a *mixed network*.

We define the numbers θ_c^s (*visit rates*) as one solution to

$$\theta_c^s = \sum_{s',c'} \theta_{c'}^{s'} q_{c',c}^{s',s} + \nu_c^s \quad (8.24)$$

If the network is open, this solution is unique and θ_c^s can be interpreted¹² as the number of arrivals per time unit of class- c customers at station s . If c belongs to a closed chain, θ_c^s is determined only up to one multiplicative constant per chain. We assume that the array $(\theta_c^s)_{s,c}$ is one non identically zero, non negative solution of Eq.(8.24).

Chains can be used to model different customer populations while a class attribute may be used to model some state information, as illustrated in Figure 8.11.

It is possible to extend Markov routing to state-dependent routing, for example, to allow for some forms of capacity limitations; see Section 8.8.6.

8.4.2 CATALOG OF SERVICE STATIONS

There are some constraints on the type of service stations allowed in multi-class product form queuing networks. Formally, the service stations must satisfy the property called “local balance in isolation” defined in Section 8.8, i.e., the stationary probability of the station in the configuration of Figure 8.4.3 must satisfy Eq.(8.96) and Eq.(8.97).

In this section we give a catalog of station types that are known to satisfy this property. There are only two categories of stations in our list (“insensitive”, and “MSCCC”), but these are fairly general categories, which contain many examples such as Processor Sharing, Delay, FIFO, Last

¹²This interpretation is valid when the network satisfies the stability condition in Theorem 8.7

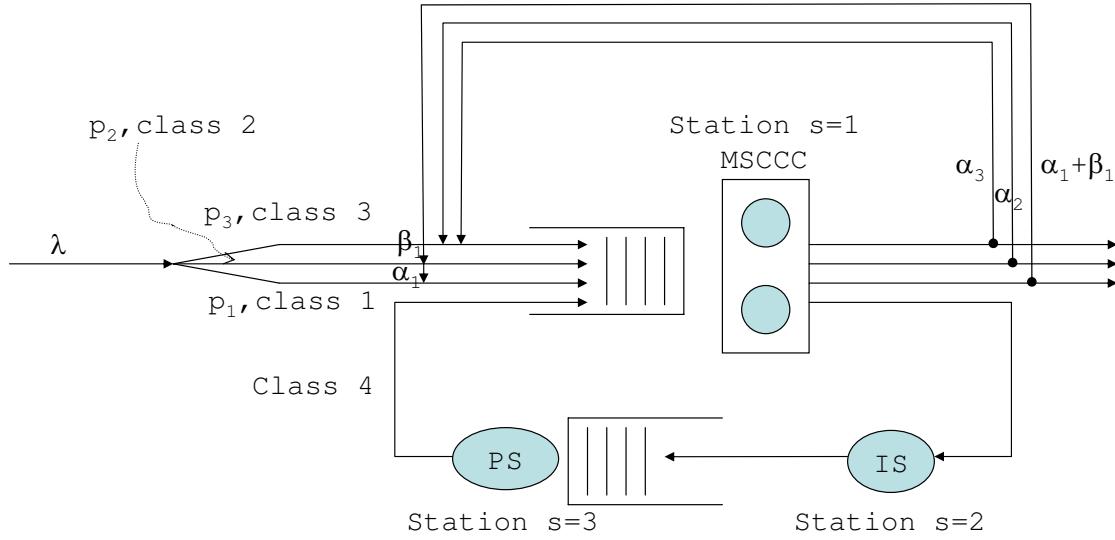


Figure 8.11: A Simple Product Form queuing network with 2 chains of customers, representing a machine with dual core processor. Chain 1 consists of classes 1, 2 and 3. Chain 2 consists of class 4.

Come First Serve etc. Thus, in practice, if you have to determine whether a given station type is allowed in multi-class product form queuing networks, a simple solution is to look up the following catalog.

We use the following definitions. Every station type is defined by

- a **discipline**: this specifies how arriving customers are queued, and which customers receive service at any time. We also assume that there is a **station buffer**: this is where customers are placed while waiting or receiving service, and is represented with some form of data structure such that every position in the station buffer can be addressed by an index $i \in \mathcal{I}$ where \mathcal{I} is some enumerable set. If \mathcal{B} is the state of the station buffer at a given time, \mathcal{B}_i is the class of the customer present at position i (equal to -1 if there is no customer present). Further we will make use of two operations.

$\mathcal{B}' = \text{add}(\mathcal{B}, i, c)$ describes the effect of adding a customer of class c at position indexed by i into the station buffer described by \mathcal{B} .

$\mathcal{B}' = \text{remove}(\mathcal{B}, i)$ describes the effect of removing the customer present at position i , if any (if there is no customer at position i , $\text{remove}(\mathcal{B}, i) = \mathcal{B}$).

For example, if the service discipline is FIFO: the data structure is a linear list such as $\mathcal{B} = (c_1, c_2, \dots, c_n)$ where c_i is the class of the i th customer (labeled in arrival order); the index set is $\mathcal{I} = \mathbb{N}$; $\text{add}(\mathcal{B}, i, c) = (c_1, \dots, c_{i-1}, c, c_i, \dots, c_n)$ and $\text{remove}(\mathcal{B}, i) = (c_1, \dots, c_{i-1}, c_{i+1}, \dots, c_n)$. We call $|\mathcal{B}|$ the number of customers present in the buffer; we assume that it is always finite (but unbounded).

- a **service requirement**, also called **service time**. For example, if a customer is a job, the service requirement may be the number of CPU cycles required; if it is a packet or a block of data, it may be the time to transmit it on a link. We assume that service requirements are random and drawn independently from anything else when a customer joins the service station. Unless otherwise specified, the distribution of service requirements may depend on the station and the class. Allowing service requirements to depend on the class is very powerful: it allows for example to model service times that are correlated from one visit to

the next.

- a **service rate**: this is the speed at which the server operates, which may depend on the customer class. If the service rate is 1, the service duration is equal to the service requirement (but the response time may be larger, as it includes waiting time). The service rate may be used to model how resources are shared between classes at a station.

CATEGORY 1: *InInsensitive Station* OR *Kelly-Whittle Stations*

This category of stations is called “insensitive” or “Kelly-Whittle”, for reasons that become clear below. We first give a formal, theoretical definition, then list the most frequent instances.

FORMAL DEFINITION.

1. The service requirement may be any phase type distribution; in practice, this may approximate any distribution, see Section 8.8.1. The service distribution may be dependent on the class.
2. (*Insertion Probability*) There is an array of numbers $\gamma(i, \mathcal{B}) \geq 0$ defined for any index $i \in \mathcal{I}$ and any station buffer state \mathcal{B} , such that: when a class c customer arrives and finds the station buffer in state \mathcal{B} just before arrival, the position at which this customer is added is drawn at random, and the probability that this customer is added at position indexed by i is

$$\gamma(i, \text{add}(\mathcal{B}, i, c)) \quad (8.25)$$

The same happens whenever a customer finishes a service phase (from the phase type service distribution), at which time the customer is treated as a new arrival.

We assume to avoid inconsistencies that $\sum_{i \in \mathcal{I}} \gamma(i, \text{add}(\mathcal{B}, i, c)) = 1$ and $\gamma(i, \mathcal{B}) = 0$ if there is no customer at position i in \mathcal{B} .

3. (*Whittle Function*) There is a function $\Psi()$, called the Whittle Function, defined over the set of feasible station buffer states, such that $\Psi(\mathcal{B}) > 0$ and the service rate allocated to a user in position i of the station buffer is

$$\gamma(i, \mathcal{B}) \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \quad (8.26)$$

if there is a customer present at position i , and 0 otherwise. Note that any positive function may be taken as Whittle function; the converse is not true, i.e. any rate allocation algorithm does not necessarily derive from a Whittle function.

One frequently considers the case where

$$\Psi(\mathcal{B}) = \Phi(\vec{n}) \quad (8.27)$$

where $\vec{n} = (n_1, \dots, n_C)$ with n_c the number of class- c customers in \mathcal{B} , and $\Phi()$ is an arbitrary positive function defined on \mathbb{N}^C . In other words, the Whittle function in such cases depends on the state of the station only through the numbers of customers (not their position in the buffer). The function Φ is called the **balance function**; the quantity $\frac{\Phi(\vec{n} - \vec{1}_c)}{\Phi(\vec{n})}$ is the rate allocated to class c . As with Whittle function, any positive Φ may be taken as balance function, but the converse is not true, any rate allocation does not necessarily derive from a balance function.

4. We assume that for any index i , class c and station buffer state \mathcal{B}

$$\begin{cases} \text{remove (add } (\mathcal{B}, i, c), i) = \mathcal{B} \\ \text{if } \mathcal{B}_i \text{ is not empty: add (remove}(\mathcal{B}, i), i, \mathcal{B}_i) = \mathcal{B} \end{cases} \quad (8.28)$$

i.e. a state remains unchanged if one adds a customer and immediately removes it, or vice-versa.

This formal definition may seem fairly appalling, but, as we show next, it is rarely necessary to make use of the formal definition. Instead, it may be easier to look up the next list of examples.

EXAMPLES OF INSENSITIVE STATIONS.

For each of these examples, the service requirement distribution may be any phase type distribution, and may be class dependent.

Global PS (Processor Sharing) The station is as in the Processor Sharing queue of Section 8.3.3.

All customers present in the station receive service, at a rate equal to $\frac{1}{n}$ when there are n customers of any class present at the station.

This is a Kelly-Whittle station by taking as station buffer the ordered list of customer classes $\mathcal{B} = (c_1, \dots, c_n)$. Adding a customer at position i has the effect that existing customers at positions $\geq i$ are shifted by one position, thus Eq.(8.28) holds. When a customer arrives, it is added at any position 1 to $n + 1$ with equal probability $\frac{1}{n+1}$, i.e. $\gamma(i, \mathcal{B}) = \frac{1}{|\mathcal{B}|}$ (recall that $|\mathcal{B}|$ is the total number of customers in when the buffer state is \mathcal{B}). The Whittle function is simply $\Psi(\mathcal{B}) = 1$ for every \mathcal{B} . Thus the service rate allocated to a customer is $\frac{1}{n}$, as required.

Global LCFSPR This service station is **Last Come First Serve, Preemptive Resume (LCF-SPR)**.

There is one global queue; an arriving customer is inserted at the head of the queue, and only this customer receives service. When an arrival occurs, the customer in service is preempted (service is suspended); preempted customers resume service where they left it, when they eventually return to service.

This is a Kelly-Whittle station by taking as station buffer the ordered list of customer classes $\mathcal{B} = (c_1, \dots, c_n)$ as in the previous example. When a customer arrives, it is added at position 1, i.e. $\gamma(i, \mathcal{B}) = \mathbf{1}_{\{i=1\}}$. The Whittle function is also $\Psi(\mathcal{B}) = 1$ for every \mathcal{B} . Thus the service rate allocated to a customer is 1 to the customer at the head of the queue, and 0 to all others, as required.

Per-Class Processor Sharing This is a variant of the Processor Sharing station, where the service rate is divided between customers of the same class, i.e. a customer receives service at rate $\frac{1}{n_c}$, where n_c is the number of class c customers present in the system.

This is a Kelly-Whittle station by taking as station buffer a collection of C lists, one per class. Only customers of class c may be present in the c th list. An index is a couple $i = (c, j)$ where c is a class index and j an integer. Adding a customer at position $i = (c, j)$ has the effect that existing customers in the c th list at positions $\geq j$ are shifted by one position, and others do not move thus Eq.(8.28) holds.

When a class c customer arrives, it is inserted into the c th list, at any position 1 to $n_c + 1$, with equal probability. Thus $\gamma((c, j), \mathcal{B}) = 0$ if the customer at position (c, j) is not of class c , and $\frac{1}{n_c}$ otherwise. We take as Whittle function $\Psi(\mathcal{B}) = 1$ for every \mathcal{B} . It follows that the service rate allocated to a customer of class c is $\frac{1}{n_c}$ as claimed above.

Per-Class LCFSPR This is a variant of the LCFSPR station, where one customer per class may be served, and this customer is the last arrived in this class.

This is a Kelly-Whittle station by taking as station buffer a collection of C lists, one per class as for per-class PS. When a class c customer arrives, it is added at the head of the c th queue, thus $\gamma(i, \mathcal{B}) = 1$ if $i = (c, 1)$ and the class at the head of the c th queue in \mathcal{B} is c , otherwise 0. It follows that the service rate allocated to a customer is 0 unless it is at the head of a queue, i.e. this customer is the last arrived in its class. We take as Whittle function $\Psi(\mathcal{B}) = 1$ for every \mathcal{B} . It follows that this station is equivalent to a collection of C independent LCFSPR service stations, one per class, with unit service rate in each.

Infinite Server (IS) or Delay station There is no queuing, customers start service immediately.

This is a Kelly-Whittle station by taking the same station buffer and insertion probability as for Global PS, but with Whittle function $\Psi(\mathcal{B}) = \frac{1}{n!}$ where $n = |\mathcal{B}|$ is the total number of customers present in the station. It follows that the service rate allocated to any customer present in the station is 1, as required.

PS, LCFSPR and IS with class dependent service rate Consider any of the previous examples, but assume that the service rate is class dependent, and depends on the number of customers of this class present in the station (call $r_c(n_c)$ the service rate for class c).

Thus, for Global PS, the service rate allocated to a class c customer is $\frac{r_c(n_c)}{n}$; for Per-Class PS, it is $\frac{r_c(n_c)}{n_c}$. For Global LCFSPR, the service rate allocated to the unique customer in service is $r_c(n_c)$; for Per Class LCFSPR, the service rate allocated to the class c customer in service is $r_c(n_c)$. For IS the rate allocated to every class c customer is $r_c(n_c)$.

This fits in the framework of Kelly-Whittle stations as follows. For PS and LCFSPR (per-class or global) replace the Whittle function by:

$$\Psi(\mathcal{B}) = \prod_{c=1}^C \frac{1}{r_c(1)r_c(2)\dots r_c(n_c)}$$

so that

$$\frac{\Psi(\text{remove } (\mathcal{B}, i))}{\Psi(\mathcal{B})} = r_c(n_c)$$

as required. For IS, replace Ψ by $\Psi(\mathcal{B}) = \frac{1}{n!} \prod_{c=1}^C \frac{1}{r_c(1)r_c(2)\dots r_c(n_c)}$ in order obtain the required service rate.

PS, LCFSPR and IS with queue size dependent service rate Consider any of the first five previous examples, but assume that the service rate is class independent, and depends on the total number of customers n present in the station (call $r(n)$ the service rate). Thus for Global PS, the service rate allocated to one customer is $\frac{r(n)}{n}$ if this customer is of class c ; for Per-Class PS, it is $\frac{r(n)}{n_c}$. For Global LCFSPR, the service rate allocated to the unique customer in service is $r(n)$; for Per Class LCFSPR, the service rate allocated to every customer ahead of its queue is $r(n)$. For IS, the service rate for every customer is $r(n)$.

This fits in the framework of Kelly-Whittle stations as follows. For PS and LCFSPR (per-class or global) replace the Whittle function by:

$$\Psi(\mathcal{B}) = \frac{1}{r(1)r(2)\dots r(n)}$$

so that

$$\frac{\Psi(\text{remove } (\mathcal{B}, i))}{\Psi(\mathcal{B})} = r(n)$$

as required. For IS, replace Ψ by $\Psi(\mathcal{B}) = \frac{1}{n!} \frac{1}{r(1)r(2)\dots r(n)}$ in order obtain the required service rate.

Symmetric Station, also called **Kelly station**: This is a generic type introduced by Kelly in [44] under the name of “symmetric” service discipline.

The station buffer is an ordered list as in the first example above. For an arriving customers who finds n customers present in the station, the probability to join position i is $p(n+1, i)$, where $\sum_{i=1}^{n+1} p(n+1, i) = 1$ (thus $\gamma(\mathcal{B}, i) = p(|\mathcal{B}|, i)$). The rate allocated to a customer in position i is $p(n, i)$ when there are n customers present. The name “symmetric” comes from the fact that the same function is used to define the insertion probability and the rate.

This fits in the framework of Kelly-Whittle stations, with Whittle function equal to 1. The global PS and global LCFSPR stations are special cases of Kelly stations.

Whittle Network This is a Per-Class Processor Sharing station where the Whittle function is a balance function, i.e. $\Psi(\mathcal{B}) = \Phi(\vec{n})$. It follows that the service rate for a class c customer is

$$\frac{1}{n_c} \frac{\Phi(\vec{n} - \vec{1}_c)}{\Phi(\vec{n})} \quad (8.29)$$

where $\vec{1}_c = (0, \dots, 1, \dots, 0)$ with a 1 in position c . This type of station is used in [13] to model resource sharing among several classes.

A network consisting of a single chain of classes and one single Whittle Station is called a Whittle Network. In such a network, customers of class c that have finished service may return to the station, perhaps with a different class.

A Whittle network can also be interpreted as a single class, multi-station network, as follows. There is one station per class, and customers may join only the station of their class. However, class switching is possible. Since knowing the station at which a customer resides entirely defines its class, there is no need for a customer to carry a class attribute, and we have a single class network.

In other words, a Whittle Network is a single class network with PS service stations, where the rate allocated to station c is $\frac{\Phi(\vec{n} - \vec{1}_c)}{\Phi(\vec{n})}$. The product form network in Theorem 8.7 implies that the stationary probability that there are n_c customers in station c for all c is

$$P(\vec{n}) = \frac{1}{\eta} \Phi(\vec{n}) \prod_{c=1}^C \bar{S}_c^{n_c} \theta_c^{n_c} \quad (8.30)$$

where \bar{S}_c is the expected service requirement at station c , θ_c the visit rate and η a normalizing constant.

Note that the stationary probability in Eq.(8.30) depends only on the traffic intensity $\rho_c = \bar{S}_c \theta_c$, not otherwise on the distribution of service times. This is the **insensitivity** property; it applies not only to Whittle networks, but more generally to all service stations of Category 1, hence the name.

CATEGORY 2: **MSCCC Station**

This second category of station contains as special case the FIFO stations with one or any fixed number of servers. It is called **Multiple Server with Concurrent Classes of Customers** in reference to [26, 51, 11]. A slightly more general form than presented here can be found in [2].

The service requirement **must be** exponentially distributed with the same parameter for all classes at this station (but the parameter may be different at different stations). If we relax this assumption,

this station is not longer admissible for multi-class product form queuing networks. Thus, unlike for category 1, this station type is **sensitive** to the distribution of service requirements.

The service discipline is as follows. There are B servers and G **token pools**. Every class is associated with exactly one token pool, but there can be several classes associated to the same token pool. The size of token pool g is an integer $T_g \geq 1$.

A customer is “eligible for service” when both one of the B servers becomes available and there is a free token in the pool g that this customer’s class is associated with. There is a single queue in which customers are queued in order of arrival; when a server becomes idle, the first eligible customer in the queue, or to arrive, starts service, and busies both one server and one token of the corresponding pool. The parameters such as G , B and the mapping \mathcal{G} of classes to token pools may be different at every station.

The FIFO queue with B servers is a special case with $G = 1$ token pool, and $T_1 = B$.

In addition, this station may have a variable service rate which depends on the total number of customers in the station. The rate must be the same for all classes (rates that depend on the population vector are not allowed, unlike for Category 1 stations).

EXAMPLE 8.6: A DUAL CORE MACHINE. Figure 8.11 illustrates a simple model of dual core processor. Classes 1, 2 or 3 represent external jobs and class 4 internal jobs. All jobs use the dual core processor, represented by station 1. External jobs can cycle through the system more than once. Internal jobs undergo a random delay and a variable delay due to communication.

The processor can serve up to 2 jobs in parallel, but some jobs require exclusive access to a critical section and cannot be served together. This is represented by an MSCCC station with 2 servers and 2 token pools, of sizes 1 and 2 respectively. Jobs that require access to the critical section use a token of the first pool; other jobs use tokens of the second pool (the second pool has no effect since its size is as large as the number of servers, but is required to fit in the general framework of multi-class product form queuing networks).

The delay of internal jobs is represented by station 2 (an “infinite server” station) and the communication delay is represented by station 3 (a “processor sharing” station, with a constant rate server).

Internal jobs always use the critical section. External jobs may use the critical section at most once. This is modelled by means of the following routing rules.

- Jobs of classes 1, 2 or 3 are external jobs. Jobs of class 1 have never used the critical section in the past and do not use it ; jobs of class 2 use the critical section; jobs of class 3 have used the critical section in the past but do not use it any more.
After service, a job of class 1 may either leave or return immediately as class 1 or 2. A job of class 2 may either leave or return immediately as class 3. A job of class 3 may either leave or return immediately as class 3.
- Jobs of class 4 represent internal jobs. They go in cycle through stations 1, 2, 3 forever.
- At station 1, classes 2 and 4 are associated with token pool 1 whereas classes 1 and 3 are associated with token pool 2, i.e. $\mathcal{G}(1) = 2, \mathcal{G}(2) = 1, \mathcal{G}(3) = 2$ and $\mathcal{G}(4) = 1$. The constraints at station 1 are thus: there can be up to 2 jobs in service, with at most one job of classes 2 or 4.

The routing matrix is

$$\begin{cases} q_{1,1}^{1,1} = \alpha_1; & q_{1,2}^{1,1} = \beta_1; \\ q_{2,3}^{1,1} = \alpha_2; \\ q_{3,3}^{1,1} = \alpha_3; \\ q_{4,4}^{1,2} = 1; & q_{4,4}^{2,3} = 1; & q_{4,4}^{3,1} = 1; \\ q_{c,c'}^{s,s'} = 0 \text{ otherwise} \end{cases}$$

where all numbers are positive, $\alpha_i \leq 1$ and $\alpha_1 + \beta_1 \leq 1$.

There are two chains: $\{1, 2, 3\}$ and $\{4\}$. The first chain is open, the second is closed, so we have a mixed network.

Let ν be the arrival rate of external jobs and p_i the probability that an arriving job is of class i . The visit rates are

$$\begin{array}{lll} \text{Class 1: } \theta_1^1 = \nu \frac{p_1}{1-\alpha_1}; & \theta_1^2 = 0; & \theta_1^3 = 0; \\ \text{Class 2: } \theta_2^1 = \nu \left(p_2 + \beta_1 \frac{p_1}{1-\alpha_1} \right); & \theta_2^2 = 0; & \theta_2^3 = 0; \\ \text{Class 3: } \theta_3^1 = \nu \frac{1}{1-\alpha_3} \left(p_3 + \alpha_2 p_2 + \alpha_2 \beta_1 \frac{p_1}{1-\alpha_1} \right); & \theta_3^2 = 0; & \theta_3^3 = 0; \\ \text{Class 4: } \theta_4^1 = 1; & \theta_4^2 = 1; & \theta_4^3 = 1. \end{array}$$

Note that the visit rates are uniquely defined for the classes in the open chain(1, 2 and 3); in contrast, for class 4, any constant can be used (instead of the constant 1).

8.4.3 THE STATION FUNCTION

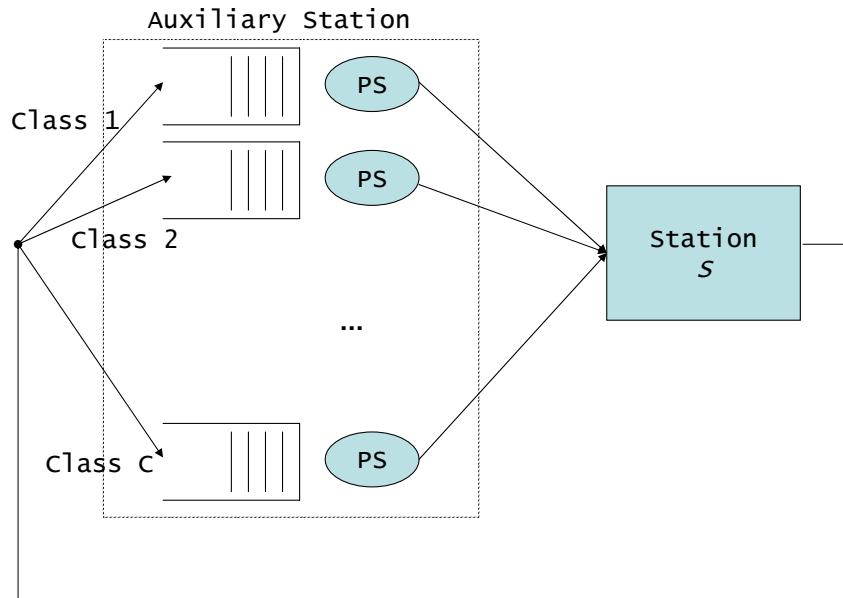


Figure 8.12: Station s in isolation.

STATION IN ISOLATION

The expression of the product form theorem uses the *station function*, which depends on the parameter of the station as indicated below, and takes as argument the vector $\vec{n} = (n_1, \dots, n_C)$ where n_c is the number of class- c customers at this station. It can be interpreted as the stationary distribution of numbers of customers in the station in isolation, up to a multiplicative constant.

More precisely, imagine a (virtual) closed network, made of this station and one external, auxiliary Per Class PS station with mean service time 1 and service rate 1 for all classes, as in Figure 8.12. In this virtual network there is one chain per class and every class c has a constant number of customers K_c . The product form theorem implies that, for any values of the vector $\vec{K} = (K_1, \dots, K_C)$, this network has a stationary regime, and the stationary probability that there are n_1 customers of class 1, ... n_C customers of class C is

$$P^{\text{isol}}(\vec{n}) = \begin{cases} 0 & \text{if } n_c > K_c \text{ for some } c \\ f(\vec{n}) \frac{1}{\eta(\vec{K})} & \text{otherwise} \end{cases} \quad (8.31)$$

where $\eta(\vec{K})$ is a normalizing constant (independent of \vec{n}).

It is often useful to consider the *generating function* $G()$ of the station function, defined as the *Z transform* of the station function, i.e. for $\vec{Z} = (Z_1, \dots, Z_C)$:

$$G(\vec{Z}) = \sum_{\vec{n} \geq 0} f(\vec{n}) \prod_{c=1}^C Z_c^{n_c} \quad (8.32)$$

(Note that, in signal processing, one often uses Z^{-1} instead of Z ; we use the direct convention, called the “mathematician’s z -transform”). The following interpretation of the generating function is quite useful. By Theorem 8.8, $G(\vec{Z})$ is the normalizing constant for the open network made of this station alone, fed by independent external Poisson processes of rates Z_c , one for each class c . Upon finishing service at this station, customers leave the network and disappear.

In the rest of this section we give the station functions for the different stations introduced earlier.

STATION FUNCTION FOR CATEGORY 1

Let $\text{pop}(\mathcal{B}) \stackrel{\text{def}}{=} (n_1, \dots, n_C)$ where n_c is the number of class c customers at this station when the station buffer is in state \mathcal{B} (i.e. $n_c = \sum_{i \in \mathcal{I}} \mathbf{1}_{\{\mathcal{B}_i=c\}}$). The station function is

$$f(\vec{n}) = \sum_{\text{pop}(\mathcal{B})=\vec{n}} \Psi(\mathcal{B}) \prod_{c=1}^C \bar{S}_c^{n_c} \quad (8.33)$$

where the summation is over all station buffer states \mathcal{B} for which the vector of populations is \vec{n} , \bar{S}_c is the mean service time for class c at this station, and Ψ is the Whittle function of this station.

Note that the station function is **independent of the insertion probabilities** γ . For example, the stationary probability is the same whether the station is PS or LCFSPR, since they differ only by the insertion probabilities.

In the case where the Whittle function is a balance function, i.e. $\Psi(\mathcal{B}) = \Phi(\vec{n})$, the summation may in some cases be computed.

1. If the station uses global queuing as in the Global PS and Global LCFSPR examples, there are $\frac{n!}{n_1! \dots n_C!}$ station buffer states for a given population vector, with $n = |\vec{n}| = \sum_{c=1}^C n_c$. The station function is

$$f(\vec{n}) = \frac{n!}{\prod_{c=1}^C n_c!} \Phi(\vec{n}) \prod_{c=1}^C \bar{S}_c^{n_c} \quad (8.34)$$

2. If the station uses per class queuing as in the Per Class PS and Per Class LCFSPR examples, there is one station buffer state for one population vector and the station function is

$$f(\vec{n}) = \Phi(\vec{n}) \prod_{c=1}^C \bar{S}_c^{n_c} \quad (8.35)$$

Global PS/Global LCFSPR/Kelly Station with constant rate. In these cases we can assume that the service rate is 1; for all of these disciplines the station function is given by Eq.(8.34) with $\Phi(\vec{n}) = 1$. The generating function is

$$G(\vec{Z}) = \frac{1}{1 - \sum_{c=1}^C \bar{S}_c Z_c} \quad (8.36)$$

Per Class PS/Per Class LCFSPR with constant rate. Here too we can assume that the service rate is 1; the station function is given Eq.(8.35) with $\Phi(\vec{n}) = 1$. The generating function is

$$G(\vec{Z}) = \prod_{c=1}^C \frac{1}{1 - \bar{S}_c Z_c} \quad (8.37)$$

IS with constant rate. Here too we can assume that the service rate is 1; the station function is given by Eq.(8.34) with $\Phi(\vec{n}) = 1/n!$. The generating function is

$$G(\vec{Z}) = \exp \left(\sum_{c=1}^C \bar{S}_c Z_c \right) \quad (8.38)$$

STATION FUNCTION FOR CATEGORY 2

For the general station in this category, the station function is a bit complex. However, for the special case of FIFO stations with one or more servers, it has a simple closed form, given at the end of this section.

General MSCCC Station Recall that the station parameters are:

- $r(i)$: service rate when the total number of customers is i
- \bar{S} : the mean service time (independent of the class)
- B : number of servers
- G : number of token pools; T_g : size of token pool g ; \mathcal{G} : mapping of class to token pool, i.e. $\mathcal{G}(c) = g$ when class c is associated with token pool g .

The station function is

$$f(\vec{n}) = d(\vec{x}) \frac{\bar{S}^{|\vec{n}|}}{\prod_{i=1}^{|\vec{n}|} r(i)} \frac{\prod_{g=1}^G x_g!}{\prod_{c=1}^C n_c!} \quad (8.39)$$

with $|\vec{n}| = \sum_{c=1}^C n_c$, $\vec{x} = (x_1, \dots, x_G)$ and $x_g = \sum_{c:G(c)=g} n_c$ (the number of customers associated with token pool g). The function d is a combinatorial function of $\vec{x} \in \mathbb{Z}^G$, recursively defined by $d(\vec{x}) = 0$ if $x_g \leq 0$ for some g , $d(0, \dots, 0) = 1$ and

$$d(\vec{x}) \times \text{bs}(\vec{x}) = \sum_{g=1}^G d(\vec{x} - \vec{1}_g) \quad (8.40)$$

where $\text{bs}(\vec{x}) \stackrel{\text{def}}{=} \min \left(B, \sum_{g=1}^G \min(x_g, T_g) \right)$ is the number of busy servers and $\vec{1}_g = (0, \dots, 1, \dots, 0)$ with a 1 in position g . Note that

$$\text{if } \sum_g \min(x_g, T_g) \leq B \text{ then } d(\vec{x}) = \prod_{g=1}^G \frac{1}{\prod_{i=1}^{x_g} \min(i, T_g)}$$

In general, though, there does not appear to be a closed form for d , except when the station is a FIFO station (see below).

For the MSCCC station, the generating function cannot be computed explicitly, in general, but when the service rate is constant, i.e. $r(i) = 1$ for all i , one may use the following algorithm. Let D be the generating function of d , i.e.

$$D(\vec{X}) = \sum_{\vec{x} \in \mathbb{N}^G} d(\vec{x}) \prod_{g=1}^G X_g^{x_g} \quad (8.41)$$

with $\vec{X} = (X_1, \dots, X_G)$. For $\vec{\tau} \in \{0, \dots, T_1\} \times \dots \times \{0, \dots, T_G\}$, let

$$D_{\vec{\tau}}(\vec{X}) \stackrel{\text{def}}{=} \sum_{\vec{x} \geq 0, \min(x_g, T_g) = \tau_g, \forall g} d(\vec{x}) \prod_{g=1}^G X_g^{x_g}$$

so that $D(\vec{X}) = \sum_{\vec{\tau} \in \{0, \dots, T_1\} \times \dots \times \{0, \dots, T_G\}} D_{\vec{\tau}}(\vec{X})$. One can compute $D_{\vec{\tau}}()$ iteratively, using $D_{\vec{0}}(\vec{X}) = 1$, $D_{\vec{\tau}}(\vec{X}) = 0$ if $\tau_g < 0$ for some g and the following, which follows from Eq.(8.40):

$$D_{\vec{\tau}}(\vec{X}) = \frac{1}{\text{bs}(\vec{\tau}) - \sum_{g: \tau_g = T_g} X_g} \sum_{g: \tau_g > 0} X_g D_{\vec{\tau} - \vec{1}_g}(\vec{X}) \quad (8.42)$$

It is sometimes useful to note that

$$D_{\vec{\tau}}(\vec{X}) = \prod_{g=1}^G \frac{X_g^{\tau_g}}{\tau_g! \left(1 - \frac{X_g}{T_g} \mathbf{1}_{\{\tau_g = T_g\}} \right)} \text{ if } \vec{\tau} \geq 0 \text{ and } \text{bs}(\vec{\tau}) < B \quad (8.43)$$

The generating function of the MSCCC station with constant service rate is then given by

$$G(\vec{Z}) = D(X_1, \dots, X_G) \quad (8.44)$$

with $X_g = \bar{S} \left(\sum_{c \text{ such that } G(c)=g} Z_c \right)$ for all token pool g .

FIFO with B servers. This is a special case of MSCCC, with much simpler formulas than in the general case. Here the parameters are

- $r(i)$: service rate when the total number of customers is i
- \bar{S} : the mean service time (independent of the class)
- B : number of servers

The station function is derived from Eq.(8.39) with $G = 1$. One finds $\vec{x} = (|\vec{n}|)$ and $d(j) = \frac{1}{\prod_{i=1}^j \min(B, i)}$ for $j \geq 1$. Thus:

$$f(\vec{n}) = \frac{\bar{S}^{|\vec{n}|}}{\prod_{i=1}^{|\vec{n}|} [r(i) \min(B, i)]} \frac{|\vec{n}|!}{\prod_{c=1}^C n_c!} \quad (8.45)$$

In the constant rate case, the generating function follows from Eq.(8.43):

$$G(\vec{Z}) = 1 + X + \frac{X^2}{2!} + \dots + \frac{X^{B-1}}{(B-1)!} + \frac{X^B}{B! \left(1 - \frac{X}{B}\right)} \quad (8.46)$$

with $X = \bar{S} \sum_{c=1}^C Z_c$.

In particular, for the **FIFO station with one server and constant rate**, the station function is

$$f(\vec{n}) = \frac{\bar{S}^{|\vec{n}|} |\vec{n}|!}{\prod_{c=1}^C n_c!} \quad (8.47)$$

and the generating function is

$$G(\vec{Z}) = \frac{1}{1 - \bar{S} \sum_{c=1}^C Z_c} \quad (8.48)$$

EXAMPLE 8.7: DUAL CORE PROCESSOR IN FIGURE 8.11. The station functions are (we use the notation n_i instead of n_i^1):

$$\begin{aligned} f^1(n_1, n_2, n_3, n_4) &= d(n_2 + n_4, n_1 + n_3) \frac{(n_1 + n_3)!(n_2 + n_4)!}{n_1! n_2! n_3! n_4!} (\bar{S}^1)^{n_1 + n_2 + n_3 + n_4} \\ f^2(n_4^2) &= (\bar{S}^2)^{n_4^2} \frac{1}{n_4^2!} \\ f^3(n_4^3) &= (\bar{S}^3)^{n_4^3} \end{aligned}$$

In the equation, d corresponds to the MSCCC station and is defined by Eq.(8.40). The generating functions for stations 2 and 3 follow immediately from (8.38) and (8.37):

$$\begin{aligned} G^2(Z_1, Z_2, Z_3, Z_4) &= e^{\bar{S}^2 Z_4} \\ G^3(Z_1, Z_2, Z_3, Z_4) &= \frac{1}{1 - \bar{S}^3 Z_4} \end{aligned}$$

For station 1, we need more work.

First we compute the generating function $D(X, Y) \stackrel{\text{def}}{=} \sum_{m \geq 0, n \geq 0} d(m, n) X^m Y^n$, using Eq.(8.40). One finds

$$\begin{aligned} D_{0,0}(X, Y) &= 1 \\ D_{1,0}(X, Y) &= \frac{X}{1 - X} \\ D_{0,1}(X, Y) &= Y \end{aligned}$$

$$\begin{aligned}
D_{1,1}(X, Y) &= \frac{1}{2-Y} (XD_{0,1} + YD_{1,0}) = \frac{XY}{1-X} \\
D_{0,2}(X, Y) &= \frac{1}{2-X} YD_{0,1} = \frac{Y^2}{2-Y} \\
D_{1,2}(X, Y) &= \frac{1}{2-X-Y} (XD_{0,2} + YD_{1,1}) = \frac{XY^2(3-X-Y)}{(2-X-Y)(2-Y)(1-X)}
\end{aligned}$$

and D is the sum of these 6 functions. After some algebra:

$$D(X, Y) = \frac{1}{1-X} \left(1 + Y + \frac{Y^2}{2-X-Y} \right) \quad (8.49)$$

Using Eq.(8.44), it follows that the generating functions of station 1 is

$$G^1(Z_1, Z_2, Z_3, Z_4) = D(\bar{S}^1(Z_2 + Z_4), \bar{S}^1(Z_1 + Z_3)) \quad (8.50)$$

QUESTION 8.4.1. Compare the station function for an IS station with constant service rate and equal mean service time for all classes with a FIFO station with constant rate and $B \rightarrow \infty$. ¹³

QUESTION 8.4.2. What is the station function $f^{\text{aux}}()$ for the auxiliary station used in the definition of the station in isolation ? ¹⁴

QUESTION 8.4.3. Verify that $D(X, 0)$ [resp. $D(0, Y)$] is the generating function of a FIFO station with one server [resp. 2 servers] (where $D()$ is given by Eq.(8.49)); explain why. ¹⁵

8.5 THE PRODUCT-FORM THEOREM

8.5.1 PRODUCT FORM

The following theorem gives the stationary probability of number of customers in explicit form; it is the main available result for queuing networks; the original proof is in [10]; extension to any service stations that satisfies the local balance property is in [78] and [44]; the proof that MSCCC stations satisfy the local balance property is in [51, 11]. The proof that all Kelly-Whittle stations satisfy the local balance property is novel and is given in Section 8.10 (see Section 8.8 for more details).

¹³Both are the same: Eq.(8.45) and Eq.(8.34) with $\Phi(\vec{n}) = 1/n!$ give the same result: $f(\vec{n}) = \frac{\bar{S}^{|\vec{n}|}}{\prod_{c=1}^C n_c!}$.

¹⁴It is a Per Class PS station with $\bar{S}_c = 1$ for all c thus $f^{\text{aux}}(\vec{n}) = 1$. The product form theorem implies that the stationary probability to see n_c customers in the station of interest is $\eta f(\vec{n})$.

¹⁵We find $\frac{1}{1-X}$ and $1 + Y + \frac{Y^2}{2-Y}$ as given by Eq.(8.46).

The generating function $D(X, Y)$ is the z -transform of the station function with one class per token group, and is also equal to the normalizing constant for the station fed by a Poisson process with rate X for group 1 and Y for group 2. If $Y = 0$ we have only group 1 customers, therefore the station is the same as a single server FIFO station with arrival rate X ; if $X = 0$, the station is equivalent to a FIFO station with 2 servers and arrival rate Y .

THEOREM 8.7. Consider a multi-class network as defined above. In particular, it uses Markov routing and all stations are Kelly-Whittle or MSCCC. Assume that the aggregation condition in Section 8.8.3 holds.

Let n_c^s be the number of class c customers present in station s and $\vec{n}^s = (n_1^s, \dots, n_C^s)$. The stationary probability distribution of the numbers of customers, if it exists, is given by

$$P(\vec{n}^1, \dots, \vec{n}^S) = \frac{1}{\eta} \prod_{s=1}^S \left(f^s(\vec{n}^s) \prod_{c=1}^C (\theta_c^s)^{n_c^s} \right) \quad (8.51)$$

where θ_c^s is the visit rate in Eq.(8.24), $f^s()$ is the station function and η is a positive normalizing constant.

Conversely, let \mathcal{E} be the set of all feasible population vectors $\vec{n} = (\vec{n}^1, \dots, \vec{n}^S)$. If

$$\sum_{\vec{n} \in \mathcal{E}} \prod_{s=1}^S \left(f^s(\vec{n}^s) \prod_{c=1}^C (\theta_c^s)^{n_c^s} \right) < \infty \quad (8.52)$$

there exists a stationary probability.

In the open network case, any vector $(\vec{n}^1, \dots, \vec{n}^S)$ is feasible, whereas in the closed or mixed case, the set of feasible population vectors \mathcal{E} is defined by the constraints on populations of closed chains, i.e.

$$\sum_{c \in \mathcal{C}} \sum_{s=1}^S n_c^s = K_c$$

for any closed chain \mathcal{C} , where K_c is the (constant) number of customers in this chain.

Note that the station function depends only on the traffic intensities. In particular, the stationary distribution is not affected by the variance of the service requirement, for stations of Category 1 (recall that stations of Category 2 must have exponential service requirement distributions).

QUESTION 8.5.1. What is the relationship between the sum in Eq.(8.52) and η ? ¹⁶

8.5.2 STABILITY CONDITIONS

In the open case, stability is not guaranteed and may depend on conditions on arrival rates. However, the next theorem says that stability can be checked at every station in isolation, and correspond to the natural conditions. In particular, pathological instabilities as discussed in the introduction of Section 8.4 cannot occur for multi-class product form queuing networks.

¹⁶They are equal.

THEOREM 8.8 (Open Case). Consider a multi-class product form queuing network as defined above. Assume that it is **open**. For every station s , $\vec{\theta}^s = (\theta_1^s, \dots, \theta_C^s)$ is the vector of visit rates, f^s the station function and $G^s()$ its generating function, given in Equations (8.36), (8.38), (8.44), and (8.46).

The network has a stationary distribution if and only if for every station s

$$G^s(\vec{\theta}^s) < \infty \quad (8.53)$$

If this condition holds, the normalizing constant of Theorem 8.7 is $\eta = \prod_{s=1}^S G^s(\vec{\theta}^s)$. Further, let $P^s(\vec{n}^s)$ be the stationary probability of the number of customers in station s . Then

$$P(\vec{n}^1, \dots, \vec{n}^S) = \prod_{s=1}^S P^s(\vec{n}^s) \quad (8.54)$$

i.e. the numbers of customers in different stations are independent. The marginal stationary probability for station s is :

$$P^s(\vec{n}^s) = \frac{1}{G^s(\vec{\theta}^s)} f^s(\vec{n}^s) \quad (8.55)$$

The proof follows from the fact that the existence of an invariant probability is sufficient for stability (as we assume that the state space is fully connected, by the aggregation condition). If the network is closed or mixed, then the corollary does not hold, i.e. the states in different stations are **not independent**, though there is product-form. Closed networks are always stable but it may not be as simple to compute the normalizing constant; efficient algorithms exist, as discussed in Section 8.6.

For mixed networks, which contain both closed and open chains, stability conditions depend on the rate functions, and since they can be arbitrary, not much can be said in general. In practice, though, the following sufficient conditions are quite useful. The proof is similar to that of the previous theorem.

THEOREM 8.9. (Sufficient Stability Condition for Mixed Networks.) Consider a multi-class product form queuing network as defined above. Assume that the network is mixed, with C_c classes in closed chains and C_o classes in open chains. Let $\vec{m} = (m_1, \dots, m_{C_c})$ be the population vector of classes in closed chains, and $\vec{n} = (n_1, \dots, n_{C_o})$ the population vector of classes in open chains. For every station s and \vec{m} define

$$L^s(\theta|\vec{m}) = \sum_{\vec{n} \in \mathbb{N}^{C_o}} f^s(\vec{m}, \vec{n}) \prod_{c=1}^{C_o} (\theta_c^s)^{n_c^s} \quad (8.56)$$

where $f^s(\vec{m}, \vec{n})$ is the station function.

If

$$L^s(\theta|\vec{m}) < \infty, \forall \vec{m}, \forall s$$

the network has a stationary distribution.

In simple cases, a direct examination of Eq.(8.52) leads to simple, natural conditions, as in the

next theorem. Essentially, it says that for the networks considered there, stability is obtained when server utilizations are less than 1.

THEOREM 8.10 (Stability of a Simple Mixed Network). *Consider a mixed multi-class product form queuing network and assume all stations are either Kelly stations (such as Global PS or Global LCFS), IS or MSCCC with constant rates.*

Let \mathcal{C} be the set of classes that belong to open chains. Define the utilization factor ρ^s at station s by

$$\begin{aligned}\rho^s &= \frac{\bar{S}^s}{B^s} \sum_{c \in \mathcal{C}} \theta_c^s && \text{if station } s \text{ is MSCCC with } B^s \text{ servers, and mean service time } \bar{S}^s \\ \rho^s &= \sum_{c \in \mathcal{C}} \theta_c^s \bar{S}_c^s && \text{if station } s \text{ is a Kelly station with mean service time } \bar{S}_c^s \text{ for class } c.\end{aligned}$$

The network has a stationary distribution if and only if $\rho^s < 1$ for every station Kelly or MSCCC station s . There is no condition on IS stations.

EXAMPLE 8.8: DUAL CORE PROCESSOR IN FIGURE 8.11. Let $q \in (0, 1]$ be the probability that an external job uses the critical section and $r > 0$ be the average number of uses of the processor outside the critical section by an external job. Thus $\theta_1^1 + \theta_3^1 = \nu r$ and $\theta_2^1 = \nu q$. By Theorem 8.10, the stability conditions are

$$\begin{aligned}\nu(r+q)\bar{S}^1 &\leq 2 \\ \nu q\bar{S}^1 &\leq 1\end{aligned}$$

where \bar{S}^1 is the average job processing time at the dual core processor. Note that we need to assume that the processing time is independent of whether it uses the critical section, and of whether it is an internal or external job. Thus the system is stable (has a stationary regime) for $\nu < \frac{2}{\bar{S}^1(q+\max(r,q))}$. Note that the condition for stability bears only on external jobs.

Let K be the total number of class 4 jobs; it is constant since class 4 constitutes a closed chain. A state of the network is entirely defined by the population vector $(n_1, n_2, n_3, n_4, n_4^2)$; the number of jobs of class 4 in station 3 is $K - n_4 - n_4^2$, and $n_c^s = 0$ for other classes. The set of feasible states is

$$\mathcal{E} = \{(n_1, n_2, n_3, n_4, n_4^2) \in \mathbb{N}^5 \text{ such that } n_4 + n_4^2 \leq K\}$$

The joint stationary probability is

$$\begin{aligned}P(n_1, n_2, n_3, n_4, n_4^2) &= \frac{1}{\eta(K)} d(n_2 + n_4, n_1 + n_3) \\ &\times \frac{(n_1 + n_3)!(n_2 + n_4)!}{n_1! n_2! n_3! n_4!} ((\theta_1^1)^{n_1} (\theta_2^1)^{n_2} (\theta_3^1)^{n_3} (\bar{S}^1)^{n_1+n_2+n_3+n_4} (\bar{S}^2)^{n_4^2} \frac{1}{n_4^{2!}} (\bar{S}^3)^{K-n_4-n_4^2})\end{aligned}$$

where we made explicit the dependency on K in the normalizing constant. This expression, while explicit, is too complicated to be of practical use. In Example 8.9 we continue with this example and compute the throughput, using the methods in the next section.

8.6 COMPUTATIONAL ASPECTS

As illustrated in Example 8.8, the product form theorem, though it provides an explicit form, may require a lot of work as enumerating all states is subject to combinatorial explosion, and the normalizing constant has no explicit form when there are closed chains. Much research has been performed on providing efficient algorithms for computing metrics of interest for multi-class product form queuing networks. They are based on a number of interesting properties, which we now derive. In the rest of this section we give the fundamental ideas used in practical algorithms; these ideas are not just algorithmic, they are also based on special properties of these networks that are of independent interest.

In the rest of this section we assume that the multi-class product form queuing network satisfies the hypotheses of the product form theorem 8.7 as described in Section 8.4, and has a stationary distribution (i.e. if there are open chains, the stability condition must hold – if the network is closed there is no condition).

8.6.1 CONVOLUTION

THEOREM 8.11. (Convolution Theorem.)

Consider a multi-class product form queuing network with closed and perhaps some open chains, and let \vec{K} be the **chain population vector** of the **closed** chains (i.e. K_c is the number of customers of chain C ; it is constant for a given network).

Let $\eta(\vec{K})$ be the normalizing constant given in the product form theorem 8.7. Let \vec{Y} a formal variable with one component per chain, and define

$$F_\eta(\vec{Y}) \stackrel{\text{def}}{=} \sum_{\vec{K} \geq \vec{0}} \eta(\vec{K}) \prod_c Y_c^{K_c}$$

Then

$$F_\eta(\vec{Y}) = \prod_{s=1}^S G^s(\vec{Z}^s) \tag{8.57}$$

where G^s is the generating function of the station function for station s , and \vec{Z}^s is a vector with one component per class, such that

$$\begin{aligned} Z_c^s &= Y_c \theta_c^s \text{ whenever } c \in \mathcal{C} \text{ and } \mathcal{C} \text{ is closed} \\ Z_c^s &= \theta_c^s \text{ whenever } c \text{ is in an open chain} \end{aligned}$$

The proof is a direct application of the product form theorem, using generating functions. Eq.(8.57) is in fact a **convolution equation**, since convolution translates into product of generating functions. It is the basis for the **convolution algorithm**, which consists in adding stations one after another, see for example [6] for a general discussion and [53] for networks with MSCCC stations other than FIFO. We illustrate the method in Example 8.9 below.

8.6.2 THROUHPUT

Once the normalizing constants are computed, one may derive throughputs for class c at station s , defined as the mean number of class c arrivals at (or departures from) station s :

THEOREM 8.12. (*Throughput Theorem* [20]) *The throughput for class c of the closed chain \mathcal{C} at station s is*

$$\lambda_c^s(\vec{K}) = \theta_c^s \frac{\eta(\vec{K} - \vec{1}_c)}{\eta(\vec{K})} \quad (8.58)$$

It follows in particular that, for closed chains, the throughputs at some station **depend only on the throughput per class and the visit rates**. Formally, choose for every closed chain \mathcal{C} a station $s_0(\mathcal{C})$ effectively visited by this chain (i.e. $\sum_{c \in \mathcal{C}} \theta_c^{s_0} > 0$); define the **per chain throughput** $\lambda_{\mathcal{C}}$ as the throughput at this station $\lambda_{\mathcal{C}}(\vec{K}) \stackrel{\text{def}}{=} \sum_{c \in \mathcal{C}} \lambda_c^{s_0(\mathcal{C})}(\vec{K})$. Since for closed chains the visit rates θ_c^s are determined up to a constant, we may decide to let $\sum_{c \in \mathcal{C}} \theta_c^{s_0(\mathcal{C})} = 1$, and then for all class $c \in \mathcal{C}$ and station s :

$$\lambda_c^s(\vec{K}) = \lambda_{\mathcal{C}}(\vec{K}) \theta_c^s \quad (8.59)$$

Also, the equivalent of Eq.(8.58) for the per chain throughput is

$$\lambda_{\mathcal{C}}(\vec{K}) = \frac{\eta(\vec{K} - \vec{1}_c)}{\eta(\vec{K})} \quad (8.60)$$

(which follows immediately by summation on $c \in \mathcal{C}$).

Note that the throughput for a class c of an *open* chain is simply the visit rate θ_c^s .

Last but not least, the throughput depends only on the normalizing constants and not on other details of the stations. In particular, stations that are different but have the same station function (such as FIFO with one server and constant rate Kelly function with class independent service time) give the same throughputs.

The next example illustrates the use of the above theorems in the study of a general case (a mixed network with an MSCCC station). There are many optimizations of this method, see [22] and references therein.

EXAMPLE 8.9: DUAL CORE PROCESSOR IN FIGURE 8.11, ALGORITHMIC ASPECT. We continue Example 8.8. Assume now that we let all parameters fixed except the arrival rate λ of external jobs and the number K of internal jobs; we would like to evaluate the throughput μ of internal jobs as a function of λ and K as well as the distribution of state of internal jobs.

We can use the throughput theorem and obtain that the throughput $\lambda(K)$ for class 4 is (we drop the dependency on λ from the notation)

$$\lambda(K) = \frac{\eta(K - 1)}{\eta(K)} \quad (8.61)$$

We now have to compute the normalizing constant $\eta(K)$ as a function of K . To this end, we use the convolution equation Eq.(8.57):

$$F_{\eta}(Y) = G^1(\vec{Z}^1)G^2(\vec{Z}^2)G^3(\vec{Z}^3) \quad (8.62)$$

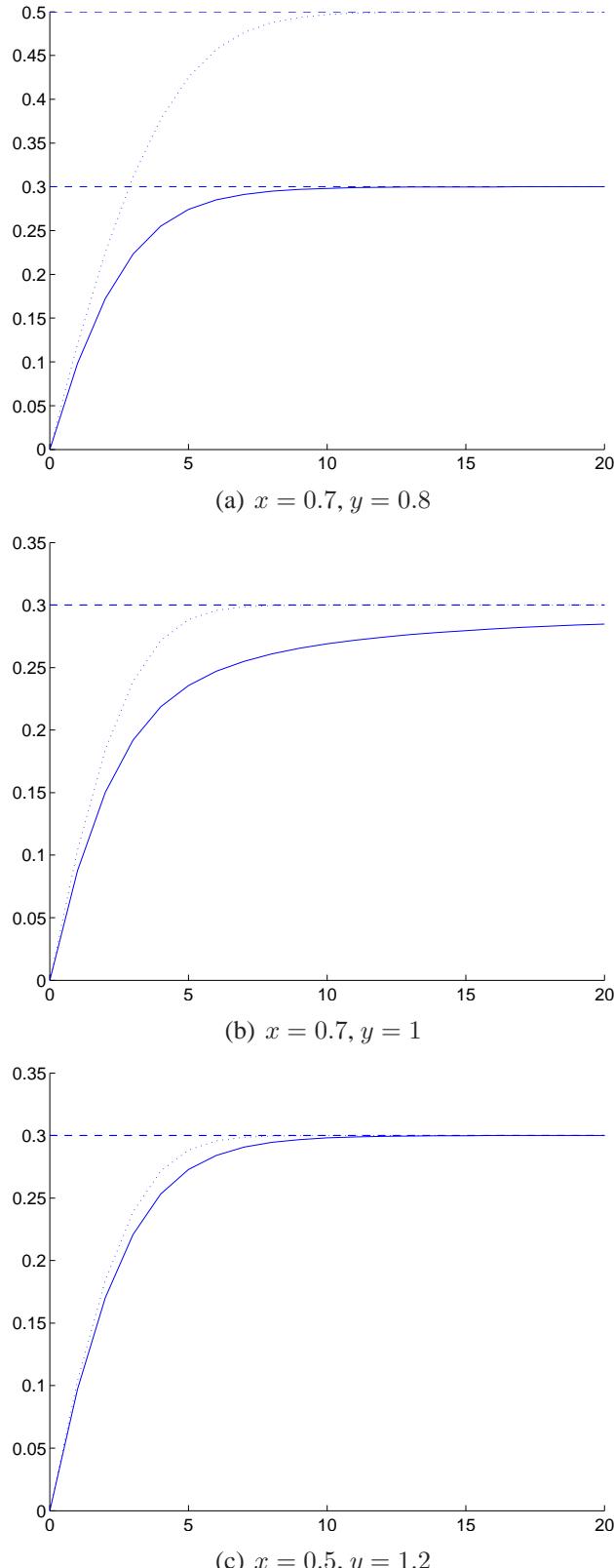


Figure 8.13: Throughput λ of internal jobs for the dual core processor in Figure 8.11, in jobs per millisecond, as a function of the number of internal jobs. Dotted curve: throughput that would be achieved if the internal jobs would not use the critical section, i.e. any job could use a processor when one is idle. x is the intensity of external traffic that uses the critical section and y of other external traffic. There are two constraints : $x + \lambda \leq 1$ (critical section) and $x + y + \lambda \leq 2$ (total processor utilization). For the dotted line only the second constraint applies. In the first panel, the first constraint is limiting and the difference in performance is noticeable. In the last panel, the second constraint is limiting and there is little difference. In the middle panel, both constraints are equally limiting. $\bar{S}^1 = 1, \bar{S}^2 = 5, \bar{S}^3 = 1\text{msec}$.

with

$$\begin{aligned}\vec{Z}^1 &= (\theta_1^1, \theta_2^1, \theta_3^1, Y) \\ \vec{Z}^2 &= (0, 0, 0, Y) \\ \vec{Z}^3 &= (0, 0, 0, Y)\end{aligned}$$

The generating functions G^1, G^2, G^3 are given in Example 8.7. It comes:

$$F_\eta(Y) = D(Y\bar{S}^1 + x, y)e^{\bar{S}^2Y} \frac{1}{1 - \bar{S}^3Y} \quad (8.63)$$

with $x = \nu q \bar{S}^1$, $y = \nu r \bar{S}^1$ and $D()$ defined in Eq.(8.49).

We can compute $\eta(K)$ by performing a power series expansion (recall that $F_\eta(Y) = \sum_{K \in \mathbb{N}} \eta(K)Y^K$) and find $\eta(K)$ numerically. Alternatively, one can interpret Eq.(8.63) as a convolution equation $\eta = \eta_1 * \eta_2 * \eta_3$ with $F_{\eta_1}(Y) = \sum_{k \in \mathbb{N}} \eta_1(k)Y^k \stackrel{\text{def}}{=} D(Y\bar{S}^1 + x, y)$, $F_{\eta_2}(Y) = e^{\bar{S}^2Y}$, $F_{\eta_3}(Y) = \frac{1}{1 - \bar{S}^3Y}$ and use fast convolution algorithms or the `filter` function as in Example 8.11. The throughput for internal jobs follows from Eq.(8.61) and is plotted in Figure 8.13.

8.6.3 EQUIVALENT SERVICE RATE

This is a useful concept, which hides away the details of a station and, as we show in the next section, can be used to aggregate network portions. Consider some arbitrary station s , of any category, with station function $f^s()$. We call **equivalent service rate** for class c at station s the quantity

$$\mu_c^{*s}(\vec{n}^s) \stackrel{\text{def}}{=} \frac{f^s(\vec{n}^s - \vec{1}_c)}{f^s(\vec{n}^s)} \quad (8.64)$$

It can be shown that $\mu_c^{*s}(\vec{n}^s)$ is indeed the average rate at which customers of class c depart from station s when station s is imbedded in a multi class queuing network and given that the numbers of customers at station s is \vec{n}^s , i.e.

$$\mu_c^{*s}(\vec{n}^s) = \sum_{\vec{e} \in \mathcal{E}(s, \vec{n}^s)} \sum_{\vec{f} \in \mathcal{E}'(s, \vec{e})} P(\vec{e}) \mu(\vec{e}, \vec{f})$$

where \vec{e} is a global micro state of the network (see Section 8.8.2 for a definition), $\mathcal{E}(s, \vec{n}^s)$ is the set of global micro-states for which the population vector at station s is \vec{n}^s , $\mathcal{E}'(s, \vec{e})$ is the set of global micro-states such that the transition $\vec{e} \rightarrow \vec{f}$ is a departure from station s , $P()$ is the stationary probability of the network and $\mu(\vec{e}, \vec{f})$ is the transition rate. This is true as long as the network satisfies the hypotheses of the product form theorem, and is a direct consequence of the local balance property.

To s we associate a *per class PS* station with unit service requirement for all classes and with balance function $f^s(\vec{n}^s)$. This virtual station is called the **equivalent station** of station s . By construction, it is a category 1 station and, by Eq.(8.35) the station functions of this virtual station and of s are identical. Further, the rate of service allocated to customers of class c is also $\mu_c^{*s}(\vec{n}^s)$. Thus, as far as the stationary probability of customers is concerned, using the original station or the equivalent station inside a network make no difference. We have an even stronger result.

<i>Station s</i>	<i>Equivalent Service Rate</i> $\mu_c^{*s}(\vec{n}^s)$
Kelly Stations with Class Dependent Service Rate. Recall that this contains as special cases Global PS and Global LCFSPR stations with constant rate.	$r_c^s(n_c^s) \frac{n_c^s}{ \vec{n}^s } \frac{1}{\bar{S}_c^s}$
Kelly Stations with Queue Size Dependent Service Rate.	$r^s(\vec{n}^s) \frac{n_c^s}{ \vec{n}^s } \frac{1}{\bar{S}_c^s}$
IS station with Class Dependent Service Rate]	$r_c^s(n_c^s) n_c^s \frac{1}{\bar{S}_c^s}$
IS station with Queue Size Dependent Service Rate.	$r^s(\vec{n}^s) n_c^s \frac{1}{\bar{S}_c^s}$
FIFO station with B servers and queue size dependent service rate. Recall that this is a station of Category 2 hence the service requirement is exponentially distributed and has the same mean \bar{S}^s for all classes.	$\frac{1}{\bar{S}^s} \min(B, \vec{n}^s) r(\vec{n}^s)$

Table 8.1: Equivalent service rates for frequently used stations. Notation: $\vec{n}^s = (n_1^s, \dots, n_C^s)$ with $n^s =$ number of class c customers at station s ; \bar{S}_c^s is the mean service requirement; $r_c^s(n_c^s)$ is the rate allocated to a class c customer when the service rate is class dependent; $r^s(|\vec{n}^s|)$ is the rate allocated to any customer when the service rate depends on queue size; $|\vec{n}^s|$ is the total number of customers in station s . For a constant rate station, take $r_c^s() = 1$ or $r^s() = 1$.

THEOREM 8.13. (Equivalent Station Theorem [78]) *In a multi-class product form queuing network any station can be replaced by its equivalent station, with equivalent service rate as in Eq.(8.64) so that the stationary probability and the throughput for any class at any station are unchanged.*

Note that the equivalent station and the equivalent service rate depend only on the station, not on the network in which the station is imbedded. It is remarkable that it is thus possible to replace any station by a per class PS station. Note however that the equivalence is only for distributions of numbers of customers and for throughputs, not for delay distributions; indeed, delays depend on the details of the station, and stations with same station function may have different delay distributions.

The equivalent service rates for a few frequently used stations are given in Table 8.1. For some stations such as the general MSCCC station there does not appear to be a closed form for the equivalent service rate. The equivalent service rate is used in the following theorem.

THEOREM 8.14. ([85])

Consider a multi-class product form queuing network with closed and perhaps some open chains, and let \vec{K} be the **chain population vector** of the **closed** chains. For any class c of the closed chain \mathcal{C} and any station s , if $n_c^s \geq 1$:

$$P^s(\vec{n}^s | \vec{K}) = P^s(\vec{n}^s - \vec{1}_c | \vec{K} - \vec{1}_c) \frac{1}{\mu_c^{*s}(\vec{n}^s)} \lambda_c^s(\vec{K}) \quad (8.65)$$

where $P^s(\cdot | \vec{K})$ is the marginal probability at station s and $\lambda_c^s(\vec{K})$ is the throughput for class c at station s .

This theorem is useful if the equivalent service rate is tractable or is numerically known. It can be used if one is interested in the marginal distribution of one station; it requires computing the throughputs $\lambda(\vec{K})$, for example using convolution or MVA. Eq.(8.65) can be used to compute $P^s(\vec{n}^s | \vec{K})$ iteratively by increasing the populations of closed chains [84]. Note that it does not give the probability of an empty station; this one can be computed by using the fact that the sum of probabilities is 1.

EXAMPLE 8.10: DUAL CORE PROCESSOR IN FIGURE 8.11, CONTINUED. We now compute the stationary probability that there are n jobs in station 2 given that there are K internal jobs in total. By Eq.(8.65):

$$P^2(n|K) = P^2(n-1|K-1)\lambda(K) \frac{\bar{S}^2}{n} \quad (8.66)$$

since the equivalent service rate for station 2 (which is an IS station) is $\frac{n}{\bar{S}^2}$ when there are n customers in the station. This gives $P^2(n|K)$ for $1 \leq n \leq K$ if we know $P^2(\cdot|K-1)$; $P(0|K)$ is obtained by the normalizing condition

$$\sum_{n=0}^K P^2(n|K) = 1$$

We compute $P^2(\cdot|K)$ by iteration on K , starting from $P^2(0|0) = 1$ and using the previous two equations. The mean number of jobs in station 2 follows:

$$\bar{N}^2(K) = \sum_{n=0}^K n P^2(n|K) \quad (8.67)$$

Similarly for station 3, with

$$P^3(n|K) = P^3(n-1|K-1)\lambda(K) \bar{S}^3 \quad (8.68)$$

since the equivalent service rate for station 3 (which is a PS station) is $\frac{1}{\bar{S}^3}$. The mean number of internal jobs in station 1 follows: $\bar{N}_4^1(K) = K - \bar{N}^2(K) - \bar{N}^3(K)$.

We derive the mean response times for internal jobs in stations 1 to 3 by using Little's law: $\bar{R}_4^s(K) = \frac{\bar{N}^s(K)}{\lambda(K)}$ for $s = 1, 2, 3$.

By Little's law, $(R_4^1 + R_4^2 + R_4^3)\lambda = K$; for large K , $\lambda \approx \theta_{\max} = \min(1-x, 2-x-y)$ and $R_4^2 \approx \bar{S}^2$, $R_4^3 \approx \bar{S}^3$ (most of the queuing is at station 1), thus $\bar{R}_4^1(K) \approx \frac{K}{\theta_{\max}} - \bar{S}^2 - \bar{S}^3$ for large K . The results are shown in Figure 8.14.

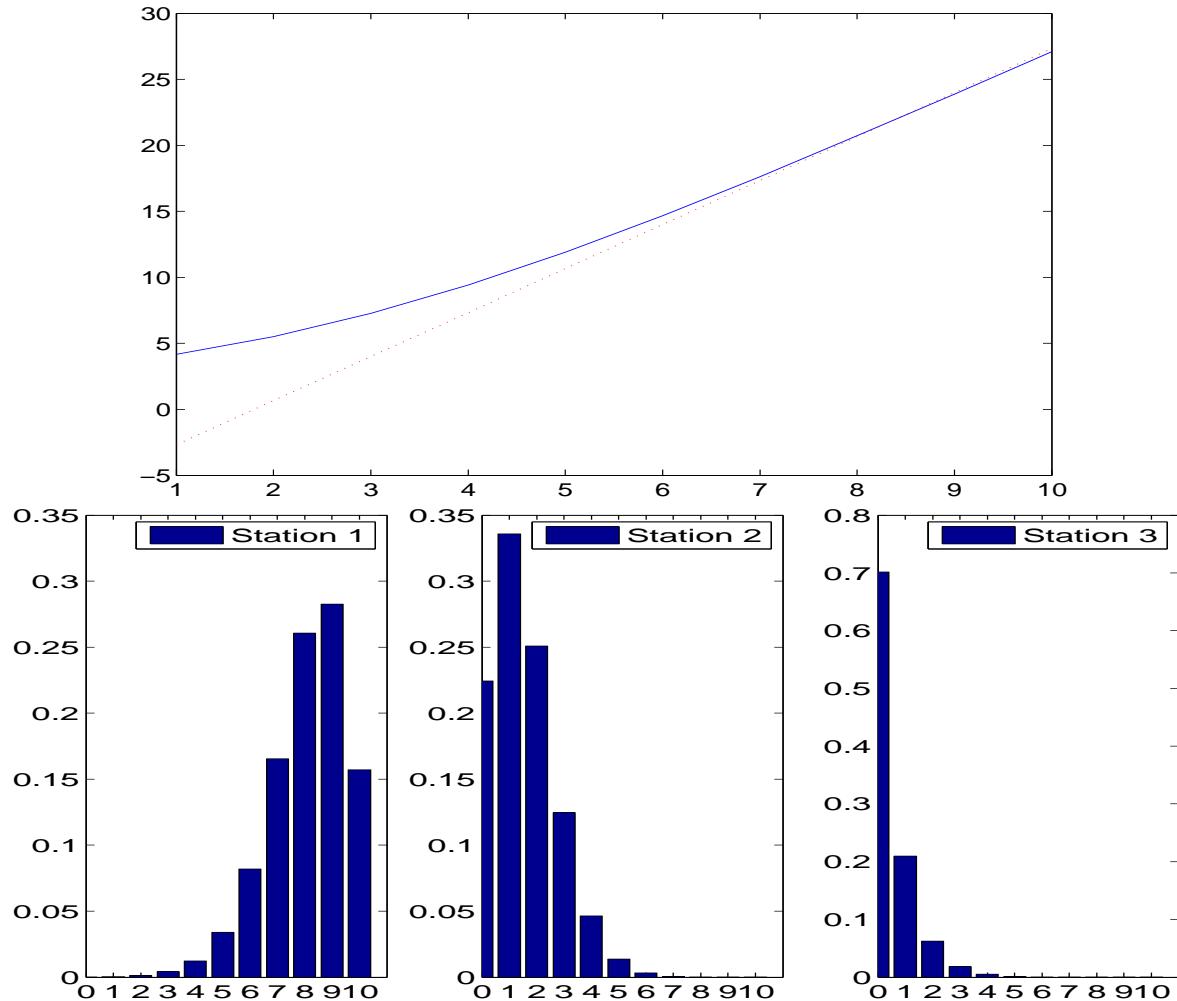


Figure 8.14: First panel: Mean Response time for internal jobs at the dual core processor, in millisecond, as a function of the number K of internal jobs. Second panel: stationary probability distribution of the number of internal jobs at stations 1 to 3, for $K = 10$. (Details of computations are in Examples 8.10 and 8.11; $\bar{S}^1 = 1$, $\bar{S}^2 = 5$, $\bar{S}^3 = 1\text{msec}$, $x = 0.7$, $y = 0.8$.)

8.6.4 SUPPRESSION OF OPEN CHAINS

In an open network, the product form theorem implies that all stations are independent in the stationary regime, and thus the network is equivalent to a collection of stations in isolation. In the mixed or closed case, this does not hold anymore, and station states are mutually dependent.

It is possible to simplify mixed networks by removing open chains. In the modified network, there are only closed chain customers, with the same routing matrix $q_{c,c'}^{s,s'}$ for all c, c' in closed chains; the stations are the same, but with a modified station function. Let $G^s(\vec{Z})$ be the z transform of the station function and θ_c^s the visit rates in the original network with open chains. In the modified network, the z transform of the station function is

$$G'^s(\vec{Z}) = G^s(\vec{Z}') \text{ with } \begin{cases} Z'_c = Z_c & \text{if } c \text{ is in a closed chain} \\ Z'_c = \theta_c^s & \text{if } c \text{ is in an open chain} \end{cases} \quad (8.69)$$

In the above, \vec{Z} is a vector with one component per class in a closed chain, whereas \vec{Z}' has one component per class, in any open or closed chain.

THEOREM 8.15. (Suppression of Open Chains) Consider a mixed multi-class network that satisfies the hypotheses of the product form theorem 8.7. Consider the network obtained by removing the open chains as described above. In the modified network the stationary probability and the throughputs for classes of closed chains are the same as in the original network.

The proof is by inspection of the generating functions. Note that the modified stations may not be of the same type as the original ones; they are fictitious stations as in the equivalent station theorem. Also, the equivalent service rates of the modified stations depend on the visit rates of the open chains that were removed, as illustrated in the next example.

EXAMPLE 8.11: DUAL CORE PROCESSOR IN FIGURE 8.11, CONTINUED. We now compute the stationary probability at station 1. We suppress the open chains and compute the equivalent service rate at station 1. We have now a single chain, single class network, with only customers of class 4. Stations 2 and 3 are unchanged; station 1 is replaced by the station with generating function:

$$G'^1(Z) = G^1(\theta_1^1, \theta_2^1, \theta_3^1, Z)$$

where G^1 is given in Eq.(8.50). With the same notation as in Example 8.9, $G'^1(z) = D(Z\bar{S}^1 + x, y)$ with D given by Eq.(8.49), and thus

$$G'^1(Z) = \frac{1}{1 - x - Z\bar{S}^1} \left(1 + y + \frac{y^2}{2 - x - y - Z\bar{S}^1} \right) \quad (8.70)$$

The station function $f'^1(n)$ of the modified station 1 is obtained by power series expansion $G'^1(Z) = \sum_{n \geq 0} f'^1(n)Z^n$. modified as follows. Since G'^1 is a rational function (quotient of two polynomials), its power series expansion can be obtained as the impulse response of a filter with rational z transform (Section D.1.8). Consider the filter

$$\frac{1}{1 - x - B\bar{S}^1} \left(1 + y + \frac{y^2}{2 - x - y - B\bar{S}^1} \right) \quad (8.71)$$

where B is the backshift operator. The sequence $(f'^1(0), f'^1(1), f'^1(2)\dots)$ is the impulse response of this filter, and can be obtained easily with the `filter` function of matlab. The equivalent service

rate of station 1 for internal jobs is

$$\mu'^1(n) = \frac{f'^1(n-1)}{f'^1(n)} \quad (8.72)$$

Since we know the equivalent service rate, we can obtain the probability distribution $P'^1(n)$ of internal jobs at station 1 using Theorem 8.14 as in Example 8.10. The results are shown in Figure 8.14.

8.6.5 ARRIVAL THEOREM AND MVA VERSION 1

Mean Value Analysis (MVA) is a method, developed in [86], which does not compute the normalizing constant and thus avoids potential overflow problems. There are many variants of it, see the discussion in [6].

In this chapter we give two versions. The former, described in this section is very simple, but applies only to some station types, as it requires to be able to derive the response time from the Palm distribution of queue size upon customer arrival. The second, described in Section 8.6.7 is more general and applies to all stations for which the equivalent service rate can be computed easily.

MVA version 1 is based on the following theorem, which is a consequence of the product form theorem and the embedded subchain theorem of Palm calculus (Theorem 7.12).

THEOREM 8.16. (*Arrival Theorem*)

Consider a multi-class product form queuing. The probability distribution of the numbers of customers seen by customer just before arriving at station s is the stationary distribution of

- the same network if the customer belongs to an open chain;
- the network with one customer less in its chain, if the customer belongs to a closed chain.

Consider now a *closed* network where all stations are FIFO or IS with constant rate, or are equivalent in the sense that they have the same station function as one of these (thus have the same equivalent service rate). Indeed, recall that stationary probabilities and throughput depend only on the station function. For example, a station may also be a global PS station with class independent service requirement of any phase type distribution, which has the same station function as a FIFO station with one server and exponential service time. In the rest of this section we call “FIFO” [resp. IS] station one that has the same station function as a single server, constant rate FIFO [resp. IS] station. Recall that at a FIFO station we need to assume that the mean service requirements are the same for all classes at the same station; for the IS station, it may be class dependent.

First we assume the FIFO [resp. IS] stations are truly FIFO [resp. IS], not just equivalent stations as defined above. We will remove this restriction later. Let $\bar{N}_c^s(\vec{K})$ be the mean number of class c customers at station s when the chain population vector is \vec{K} . The mean response time for a class c customer at a FIFO station s when the population vector is \vec{K} is

$$\bar{R}_c^s(\vec{K}) = \left(1 + \sum_c \bar{N}_c^s(\vec{K} - \vec{1}_c) \right) \bar{S}^s$$

where \mathcal{C} is the chain of class c . This is because of the exponential service requirement assumption: an arriving customer has to wait for \bar{S}^s multiplied by the number of customers present upon arrival; in average, this latter number is $\sum_c \bar{N}_c^s(\vec{K} - \vec{1}_{\mathcal{C}})$ by the arrival theorem. By Little's formula:

$$\bar{R}_c^s(\vec{K}) \lambda_c^s(\vec{K}) = \bar{N}_c^s(\vec{K})$$

Combining the two gives

$$\bar{N}_c^s(\vec{K}) = \lambda_c^s(\vec{K}) \left(1 + \sum_c \bar{N}_c^s(\vec{K} - \vec{1}_{\mathcal{C}}) \right) \bar{S}^s \quad (8.73)$$

which is valid for FIFO stations. For a delay station one finds

$$\bar{N}_c^s(\vec{K}) = \lambda_c^s(\vec{K}) \bar{S}_c^s \quad (8.74)$$

This gives a recursion for $\bar{N}_c^s(\vec{K})$ if one can get determine $\lambda_c^s(\vec{K})$. The next observation is Eq.(8.59), which says that if we know the throughput at one station visited by a chain, then we know the throughputs for all stations and all classes of the same chain. The last observation is that the sum of the numbers of customers across all stations and all classes of chain \mathcal{C} is equal to $K_{\mathcal{C}}$. Combining all this gives: for every chain \mathcal{C} , if $K_{\mathcal{C}} > 0$ then

$$\frac{K_{\mathcal{C}}}{\lambda_{\mathcal{C}}(\vec{K})} = \sum_{c \in \mathcal{C}} \left[\sum_{s: \text{FIFO}} \theta_c^s \left(1 + \sum_{c'} \bar{N}_{c'}^s(\vec{K} - \vec{1}_{\mathcal{C}}) \right) \bar{S}^s + \sum_{s: \text{IS}} \theta_c^s \bar{S}_c^s \right] \quad (8.75)$$

and

$$\lambda_{\mathcal{C}}(\vec{K}) = 0 \text{ if } K_{\mathcal{C}} = 0 \quad (8.76)$$

For every FIFO station s and class c :

$$\bar{N}_c^s(\vec{K}) = \theta_c^s \lambda_{\mathcal{C}(c)}(\vec{K}) \left(1 + \sum_{c'} \bar{N}_{c'}^s(\vec{K} - \vec{1}_{\mathcal{C}(c)}) \right) \bar{S}^s \text{ if } K_{\mathcal{C}(c)} > 0 \quad (8.77)$$

$$= 0 \text{ if } K_{\mathcal{C}(c)} = 0 \quad (8.78)$$

Second, we observe that the resulting equations depend only on the station function, therefore they apply to equivalent stations as well.

The **MVA algorithm version 1** iterates on the total population, adding customers one by one. At every step, the throughput is computed using Equation (8.75). Then the mean queue sizes at FIFO queues are computed using Equation (8.77), which closes the loop. We give the algorithm in the case of a single chain. For the multi-chain case, the algorithm is similar, but there are many optimizations to reduce the storage requirement, see [6].

EXAMPLE 8.12: MEAN VALUE ANALYSIS OF FIGURE 8.5. We model the system as a single class, closed network. The CPU is modelled as a PS station, disks A and B as FIFO single servers, and think time as an IS station. We fix the visit rate $\theta^{\text{thinktime}}$ to 1 so that $\theta^{\text{CPU}} = V_{\text{CPU}}$, $\theta^A = V_A$ and $\theta^B = V_B$. Note that the routing probabilities need not be specified in detail, only the visit rates are required.

The CPU station is not a FIFO station, but is has the same station function, therefore we may apply MVA and treat it as if it would be FIFO.

Figure 8.15 shows the results, which are essentially as predicted by bottleneck analysis in Figure 8.6.

Algorithm 7 MVA Version 1: Mean Value Analysis for a single chain closed multi-class product form queuing network containing only constant rate FIFO and IS stations, or stations with same station functions.

- 1: $K = \text{population size}$
 - 2: $\lambda = 0$ ▷ throughput
 - 3: $Q^s = 0$ for all station $s \in \text{FIFO}$ ▷ total number of customers at station s , $Q^s = \sum_c N_c^s$
 - 4: Compute the visit rates θ_c^s using Eq.(8.24) and $\sum_{c=1}^C \theta_c^1 = 1$
 - 5: $\theta^s = \sum_c \theta_c^s$ for every $s \in \text{FIFO}$
 - 6: $h = \sum_{s \in \text{IS}} \sum_c \theta_c^s \bar{S}_c^s + \sum_{s \in \text{FIFO}} \theta^s \bar{S}^s$ ▷ constant term in Eq.(8.75)
 - 7: **for** $k = 1 : K$ **do**
 - 8: $\lambda = \frac{k}{h + \sum_{s \in \text{FIFO}} \theta^s Q^s \bar{S}^s}$ ▷ Eq.(8.75)
 - 9: $Q^s = \lambda \theta^s \bar{S}^s (1 + Q^s)$ for all $s \in \text{FIFO}$
 - 10: **end for**
 - 11: The throughput at station 1 is λ
 - 12: The throughput of class c at station s is $\lambda \theta_c^s$
 - 13: The mean number of customers of class c at FIFO station s is $Q^s \theta_c^s / \theta^s$
 - 14: The mean number of customers of class c at IS station s is $\lambda \theta_c^s \bar{S}_c^s$
-

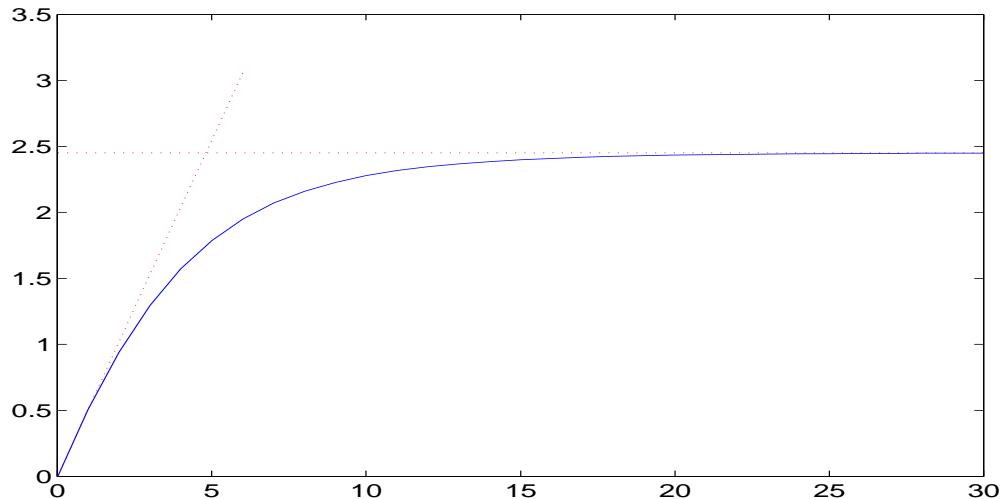


Figure 8.15: Throughput in transactions per second versus number of users, computed with MVA for the network in Figure 8.5. The dotted lines are the bounds of bottleneck analysis in Figure 8.6.

8.6.6 NETWORK DECOMPOSITION

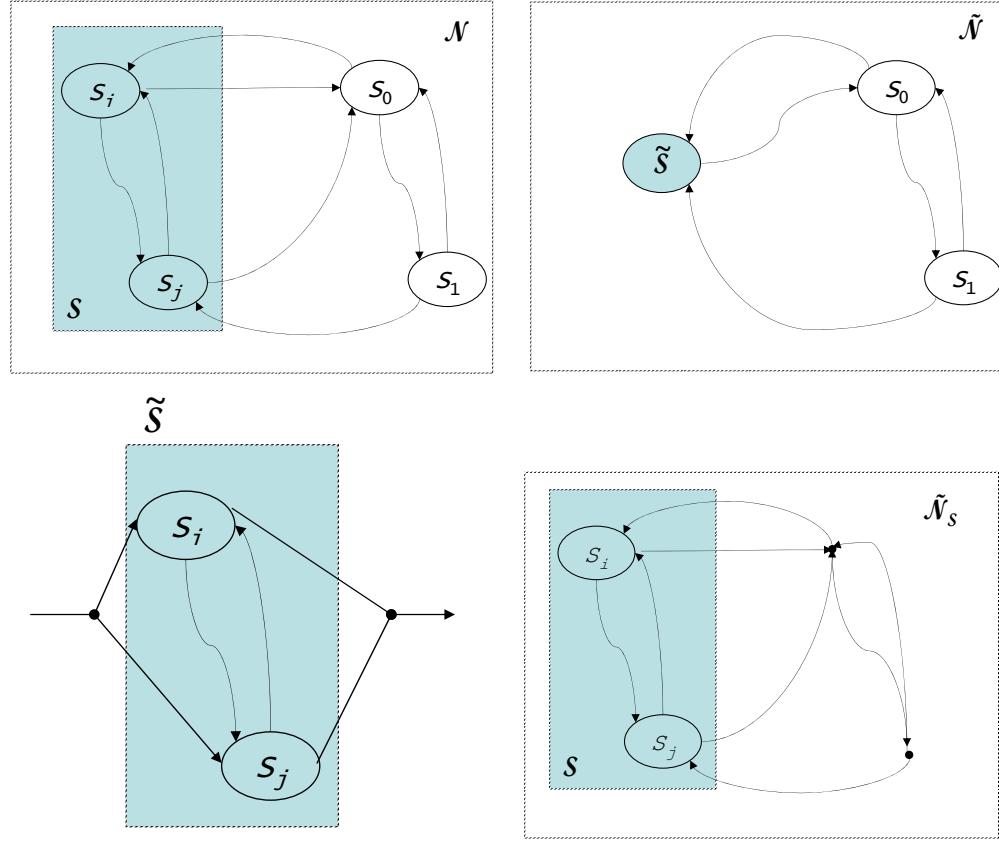


Figure 8.16: Decomposition procedure: original network \mathcal{N} , with subnetwork S ; simplified network $\tilde{\mathcal{N}}$; equivalent station \tilde{s} ; subnetwork in short-circuit $\tilde{\mathcal{N}}_S$.

A key consequence of the product form theorem is the possibility to replace an entire subnetwork by an equivalent single station. This can be done recursively and is the basis for many algorithms, such as MVA version 2.

Consider a multi-class product form network \mathcal{N} and a subnetwork S . The stations in S need not directly connected and the network can be closed, mixed or open. If the network is mixed or open, we consider that outside arrivals are from some fictitious station 0, and $0 \notin S$. We create two virtual networks: $\tilde{\mathcal{N}}$ and $\tilde{\mathcal{N}}_S$ and a virtual station \tilde{s} as follows (Figure 8.16).

The virtual station \tilde{s} , called the **equivalent station** of S , is obtained by isolating the set of stations S from the network \mathcal{N} and collapsing classes to chains. Inside \tilde{s} , there is only one class per chain, i.e. a customer's attribute is its chain C ; furthermore, the station is a “per class PS” station, with service rate to be defined later¹⁷.

$\tilde{\mathcal{N}}$, called the **simplified network**, is obtained by replacing all stations in S by the equivalent station \tilde{s} . In $\tilde{\mathcal{N}}$, routing is defined by the corresponding natural aggregation, i.e. is the same as if the stations in S were still present but not observable individually. Thus the routing matrix \tilde{q} is:

$$\tilde{q}_{c,c'}^{s,s'} = q_{c,c'}^{s,s'} \text{ if } s \notin S \text{ and } s' \notin S$$

¹⁷Observe that within one service station customers cannot change class, therefore if we aggregate a subnetwork into a single station, we must aggregate classes of the same chain as well.

$$\begin{aligned}
\tilde{q}_{c,c'}^{\mathcal{S},s'} &= \begin{cases} 0 & \text{if } c' \notin \mathcal{C} \\ \frac{1}{\tilde{\theta}_c} \sum_{s \in \mathcal{S}, c \in \mathcal{C}} \theta_c^s q_{c,c'}^{s,s'} & \text{if } c' \in \mathcal{C} \end{cases} \\
\tilde{q}_{c,\mathcal{C}}^{s,\mathcal{S}} &= \begin{cases} 0 & \text{if } c \notin \mathcal{C} \\ \sum_{s' \in \mathcal{S}, c' \in \mathcal{C}} q_{c,c'}^{s,s'} & \text{if } c \in \mathcal{C} \end{cases} \\
\tilde{q}_{\mathcal{C},\mathcal{C}'}^{\mathcal{S},\mathcal{S}} &= \begin{cases} 0 & \text{if } \mathcal{C} \neq \mathcal{C}' \\ \frac{1}{\tilde{\theta}_c} \sum_{s,s' \in \mathcal{S}, c, c' \in \mathcal{C}} \theta_c^s q_{c,c'}^{s,s'} & \text{if } \mathcal{C} = \mathcal{C}' \end{cases} \\
\tilde{\theta}_c &= \sum_{s \in \mathcal{S}, c \in \mathcal{C}} \theta_c^s
\end{aligned}$$

where, for example, $\tilde{q}_{c,c'}^{\mathcal{S},s'}$ is the probability that a chain \mathcal{C} customer leaving station $\tilde{\mathcal{S}}$ joins station s' with class c' . If there are some open chains, recall that $s = 0$ represents arrivals and departures and we assumed $0 \notin \mathcal{S}$; in such cases, the external arrival rate of chain \mathcal{C} customers to the virtual station $\tilde{\mathcal{S}}$ is

$$\lambda_{\mathcal{C}}^{\mathcal{S}} = \sum_{s \in \mathcal{S}, c \in \mathcal{C}} \lambda_c^s$$

and the probability that a chain \mathcal{C} customers leaves the network after visiting $\tilde{\mathcal{S}}$ is

$$\frac{1}{\tilde{\theta}_c} \sum_{s \in \mathcal{S}, c \in \mathcal{C}} \theta_c^s q_c^{s,0}$$

where $q_c^{s,0} \stackrel{\text{def}}{=} 1 - \sum_{s',c'} q_{c,c'}^{s,s'}$ is the probability that a class c customer leaves the network after visiting station s .

The visit rates in $\tilde{\mathcal{N}}$ are the same as in \mathcal{N} for stations not in \mathcal{S} ; for the equivalent station $\tilde{\mathcal{S}}$, the visit rate for chain \mathcal{C} is $\tilde{\theta}_c$ given above. The station function of the equivalent station $\tilde{\mathcal{S}}$ is computed in such a way that replacing all stations in \mathcal{S} by $\tilde{\mathcal{S}}$ makes no difference to the stationary probability of the network. It follows, after some algebra, from the product form theorem; the precise formulation is a bit heavy:

$$f^{\mathcal{S}}(\vec{k}) = \sum_{(\vec{n}^s)_{s \in \mathcal{S}} \text{ such that } \sum_{s \in \mathcal{S}, c \in \mathcal{C}} n_c^s = k_{\mathcal{C}}} \prod_{s \in \mathcal{S}} \left[f^s(\vec{n}^s) \prod_c \left(\frac{\theta_c^s}{\tilde{\theta}_c^{\mathcal{S}}} \right)^{n_c^s} \right] \quad (8.79)$$

where \vec{k} is a population vector of closed or open chains. Note that it may happen that some chain \mathcal{C}_0 be “trapped” in \mathcal{S} , i.e. customers of this chain never leave \mathcal{S} . The generating function of the virtual station \mathcal{S} has a simple expression

$$G^{\mathcal{S}}(\vec{Z}) = \prod_{s \in \mathcal{S}} G^s(\vec{X}^s) \text{ with } X_c^s = Z_{\mathcal{C}(c)} \frac{\theta_c^s}{\tilde{\theta}_c^{\mathcal{S}}} \quad (8.80)$$

where $\mathcal{C}(c)$ is the chain of class c . Here \mathcal{C} spans the set of all chains, closed or open. Thus, the equivalent station $\tilde{\mathcal{S}}$ is a per-class PS station, with one class per chain, and with balance function $f^{\mathcal{S}}(\vec{k})$. In the next theorem, we will give an equivalent statement that is easier to use in practice.

The second virtual network, $\tilde{\mathcal{N}}_{\mathcal{S}}$ is called the **subnetwork in short-circuit**. It consists in replacing anything not in \mathcal{S} by a short-circuit. In $\tilde{\mathcal{N}}_{\mathcal{S}}$, the service times at stations not in \mathcal{S} are 0 and customers instantly traverse the complement of \mathcal{S} . This includes the virtual station 0 which represents the

outside, so $\tilde{\mathcal{N}}_{\mathcal{S}}$ is a closed network¹⁸. The population vector \vec{k} remains constant in $\tilde{\mathcal{N}}_{\mathcal{S}}$; the visit rates at stations in \mathcal{S} are the same as in the original network for closed chains. For classes that belong to a chain that is open in the original network, we obtain the visit rates by setting arrival rates to 1.

THEOREM 8.17. (Decomposition Theorem [78])

Consider a multi-class network that satisfies the hypotheses of the product form theorem 8.7. Any subnetwork \mathcal{S} can be replaced by its equivalent station $\tilde{\mathcal{S}}$, with one class per chain and station function defined by Eq.(8.80). In the resulting equivalent network $\tilde{\mathcal{N}}$, the stationary probability and the throughputs that are observable are the same as in the original network.

Furthermore, if \mathcal{C} effectively visits \mathcal{S} , the equivalent service rate to chain \mathcal{C} (closed or open) at the equivalent station $\tilde{\mathcal{S}}$ is

$$\mu_{\mathcal{C}}^{*\mathcal{S}}(\vec{k}) = \lambda_{\mathcal{C}}^{*\mathcal{S}}(\vec{k}) \quad (8.81)$$

where $\lambda_{\mathcal{C}}^{\mathcal{S}}(\vec{k})$ is the throughput of chain \mathcal{C} for the subnetwork in short-circuit $\tilde{\mathcal{N}}_{\mathcal{S}}$ when the population vector for all chains (closed or open) is \vec{k} .*

The phrase “that are observable” means: the numbers of customers of any class at any station not in \mathcal{S} ; the total number of customers of chain \mathcal{C} that are present in any station of \mathcal{S} ; the throughputs of all classes at all stations not in \mathcal{S} ; the throughputs of all chains. Recall that the per chain throughput $\lambda_{\mathcal{C}}(\vec{K})$ (defined in Eq.(8.59)) is the throughput measured at some station $s_{\mathcal{C}}$ effectively visited by chain \mathcal{C} . The station $s_{\mathcal{C}}$ is assumed to be the same in the original and the virtual networks, which is possible since the visit rates are the same.

If \mathcal{C} does not effectively visit \mathcal{S} (i.e. if $\tilde{\mathcal{C}} \stackrel{\text{def}}{=} \sum_{s \in \mathcal{S}, c \in \mathcal{C}} \theta_c^s = 0$) then the equivalent service rate $\mu_{\mathcal{C}}^{*\mathcal{S}}$ is undefined, which is not a problem since we do not need it.

By the throughput theorem, Eq.(8.81) can also be written $\mu_{\mathcal{C}}^{*\mathcal{S}}(\vec{k}) = \frac{\eta^*(\vec{k} - \vec{1}_{\mathcal{C}})}{\eta^*(\vec{k})}$ where $\eta^*(\vec{k})$ is the normalizing constant for the subnetwork in short-circuit $\tilde{\mathcal{N}}_{\mathcal{S}}$.

If \mathcal{S} consists of a single station with one class per chain at this station, then the equivalent station is the same as the original station, as expected. Also, the theorem implies, as a byproduct, that the equivalent service rate for class c at a station s , as defined in Eq.(8.64), is equal to the throughput for class c at the network made of this station and a short circuit for every class (i.e. every class c customer immediately returns to the station upon service completion, with the same class).

EXAMPLE 8.13: DUAL CORE PROCESSOR IN FIGURE 8.11, CONTINUED. We replace stations 2 and 3 by one aggregated station $\tilde{\mathcal{S}}$ as in Figure 8.17. This station receives only customers of class 4 (internal jobs). Its equivalent service rate is

$$\mu^*(n_4) = \frac{\eta^*(n_4 - 1)}{\eta^*(n_4)} \quad (8.82)$$

where $\eta^*(n_4)$ is the normalizing constant for the network $\tilde{\mathcal{N}}_{\mathcal{S}}$ obtained when replacing station 1 by a short-circuit as in Figure 8.17; the z transform of η^* is given by the convolution theorem 8.11:

$$F_{\eta^*}(Y) = e^{\bar{S}^2 Y} \frac{1}{1 - \bar{S}^3 Y} \quad (8.83)$$

¹⁸Be careful that this is different from the procedure used when defining the station in isolation. In $\tilde{\mathcal{N}}_{\mathcal{S}}$, \mathcal{S} is connected to a short-circuit, i.e. a station where the service requirement is 0; in contrast, in the configuration called “ \mathcal{S} in isolation”, \mathcal{S} is connected to a station with unit rate and unit service requirement.

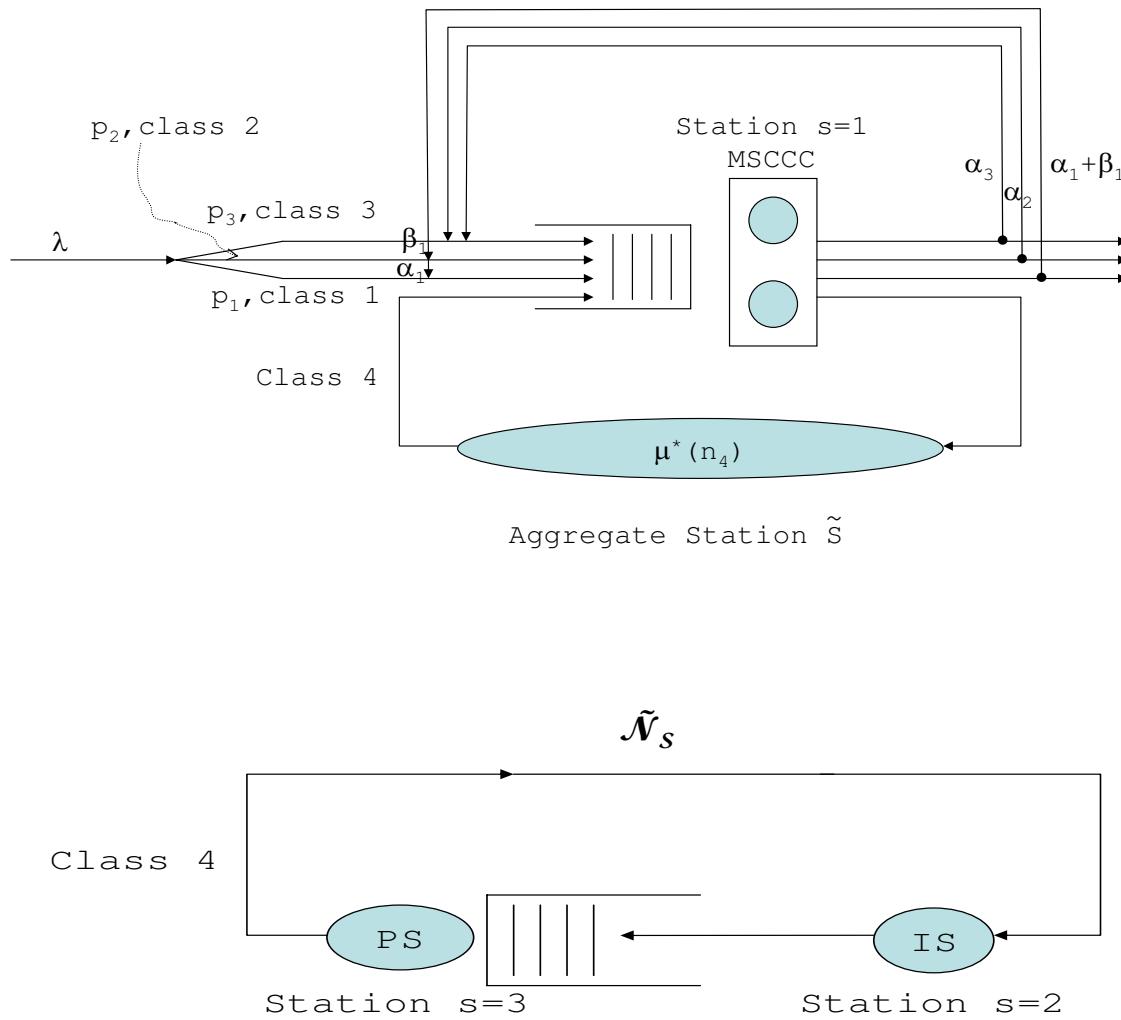


Figure 8.17: Aggregation of stations applied to the dual core processor example of Figure 8.11. First panel: stations 2 and 3 are replaced by \tilde{S} . Bottom panel: the network in short-circuit $\tilde{\mathcal{N}}_S$ used to compute the equivalent service rate $\mu^*(n_4)$ of \tilde{S} .

One can compute a Taylor expansion and deduce $\eta^*(n)$ or use `filter` as in the other examples, but here one can also find a closed form

$$\eta^*(n) = (\bar{S}^3)^n \sum_{k=0}^n \left(\frac{\bar{S}^2}{\bar{S}^3}\right)^k \frac{1}{k!} \quad (8.84)$$

Note that for large n , $\eta^*(n) \approx (\bar{S}^3)^n \exp\left(\frac{\bar{S}^2}{\bar{S}^3}\right)$ and thus $\mu^*(n) \approx \frac{1}{\bar{S}^3}$, i.e. it is equivalent to station 3 (but this is true only for large n). We can deduce the equivalent service rate $\mu^*(n)$ and obtain the probability distribution $P^*(n)$ of internal jobs at stations 2 or 3 using Theorem 8.14 as in Example 8.10.

Note that internal jobs are either at station 1, or at stations 2 or 3. Thus we should have

$$P^*(n|K) = P'^1(K - n|K) \quad (8.85)$$

where $P'^1(.|K)$ is the probability distribution for internal jobs at station 1, already obtained in Example 8.11, and we can verify this numerically.

8.6.7 MVA VERSION 2

This is an algorithm which, like MVA version 1, avoids computing the normalizing constant, but which applies to fairly general station types [84]. We give a version for single chain (but multi-class) networks. For networks with several chains, the complexity of this method is exponential in the number of chains, and more elaborate optimizations have been proposed; see [27, 28] as well as [6] and the discussion therein.

The starting point is the decomposition theorem, which says that one can replace a subnetwork by a single station if one can compute its throughputs in short circuit. For example, using MVA version 1, one can compute the throughputs of a subnetwork made of single server FIFO or IS stations (or equivalent), therefore one can replace the set of all such stations in a network by one single station.

MVA version 2 does the same thing for general stations in closed networks. This can be reduced to the simpler problem of how to compute the throughput of a network of 2 stations, with numerically known service rates. If we can solve this problem, we can replace the 2 stations by a new one, the service rate is equal to the throughput (by Theorem 8.17), and we can iterate. This problem is solved by the next theorem. It uses the concept of networks in short-circuit.

THEOREM 8.18. (*Complement Network Theorem*) Consider a closed multi-class product form queueing network \mathcal{N} . Let $\mathcal{S}^1, \mathcal{S}^2$ be a partition of \mathcal{N} in two subnetworks and let $\mathcal{N}_{\mathcal{S}^1}, \mathcal{N}_{\mathcal{S}^2}$ be the corresponding subnetworks in short circuit (in $\mathcal{N}_{\mathcal{S}^1}$, all stations in \mathcal{S}^2 are short circuited). Define:

- $P^1(\vec{k}|\vec{K})$ = the stationary probability that the number of customers of chain \mathcal{C} present in \mathcal{S}^1 is $k_{\mathcal{C}}$ for all \mathcal{C} when the total network population vector is \vec{K} ;
- $\eta^1(\vec{K})$ = [resp. $\eta^2(\vec{K})$, $\eta(\vec{K})$] the normalizing constant of $\mathcal{N}_{\mathcal{S}^1}$ [resp. $\mathcal{N}_{\mathcal{S}^2}, \mathcal{N}$] when the total network population vector is \vec{K} ;
- $\lambda_{\mathcal{C}}^{*1}(\vec{K})$ = [resp. $\lambda_{\mathcal{C}}^{*2}(\vec{K}), \lambda_{\mathcal{C}}(\vec{K})$] the per chain throughput of chain \mathcal{C} in $\mathcal{N}_{\mathcal{S}^1}$ [resp. $\mathcal{N}_{\mathcal{S}^2}, \mathcal{N}$] when the total network population vector is \vec{K} .

Then for $\vec{0} \leq \vec{k} \leq \vec{K}$:

$$P^1(\vec{k}|\vec{K}) = \frac{\eta^1(\vec{k})\eta^2(\vec{K} - \vec{k})}{\eta(\vec{K})} \quad (8.86)$$

and for any chain \mathcal{C} such that $k_{\mathcal{C}} > 0$:

$$P^1(\vec{k}|\vec{K}) = P^1(\vec{k} - \vec{1}_{\mathcal{C}}|(\vec{K} - \vec{1}_{\mathcal{C}}) \frac{\lambda_{\mathcal{C}}(\vec{K})}{\lambda_{\mathcal{C}}^{*1}(\vec{k})} \quad (8.87)$$

$$P^1(\vec{k}|\vec{K}) = P^1(\vec{k}|(\vec{K} - \vec{1}_{\mathcal{C}}) \frac{\lambda_{\mathcal{C}}(\vec{K})}{\lambda_{\mathcal{C}}^{*2}(\vec{K} - \vec{k})} \quad (8.88)$$

The inequalities $\vec{0} \leq \vec{k} \leq \vec{K}$ are componentwise. The proof is by direct inspection: recognize in Eq.(8.86) the convolution theorem; Eq.(8.87) and Eq.(8.88) follow from Eq.(8.86) and the throughput theorem.

Note that Eq.(8.87) is an instance of the equivalent service rate formula Eq.(8.65), since $\lambda_{\mathcal{C}}^{*1}(\vec{k}) = \mu_{\mathcal{C}}^{*1}(\vec{k})$ is also equal to the equivalent service rate of \mathcal{S}^1 . Eq.(8.88) is the symmetric of Eq.(8.87) when we exchange the roles of \mathcal{S}^1 and \mathcal{S}^2 since $P^1(\vec{k}|\vec{K}) = P^2(\vec{K} - \vec{k}|\vec{K})$.

\mathcal{S}^2 is called the complement network of \mathcal{S}^1 in the original work [84], hence the name.

THE MVA COMPOSITION STEP In the rest of this section we consider that there is only one chain, and drop index \mathcal{C} . Assume that we know the throughputs of the two subnetworks $\lambda^{*1}(K), \lambda^{*2}(K)$; the goal of the composition step is to compute $\lambda(K)$. We compute the distribution $P^1(\cdot|K)$ by iteration on K , starting with $P^1(0|0) = 1, P^1(n|0) = 0, n \geq 1$. Eq.(8.87) and Eq.(8.88) become

$$\text{for } k = 1 \dots K : P^1(k|K) = P^1(k-1|(K-1)) \frac{\lambda(K)}{\lambda^{*1}(k)} \quad (8.89)$$

$$\text{for } k = 0 \dots K-1 : P^1(k|K) = P^1(k|(K-1)) \frac{\lambda(K)}{\lambda^{*2}(K-k)} \quad (8.90)$$

None of the two equations alone is sufficient to advance one iteration step, but the combination of the two is. For example, use the former for $k = 1 \dots K$ and the latter for $k = 0$. $\lambda(K)$ is then obtained by the condition $\sum_{n=0}^K P^1(k|K) = 1$.

MVA VERSION 2 The algorithm works in two phases. In phase 1, the throughput is computed. The starting point is a network \mathcal{N}_0 ; first, we compute the throughput of the subnetwork \mathcal{S}^0 made of all stations to which MVA version 1 applies, as this faster than MVA version 1. We replace \mathcal{S}^0 by its equivalent station; let \mathcal{N}_1 be the resulting network.

In one step we match stations 2 by 2, possibly leaving one station alone. For every pair of matched stations we apply the MVA Composition Step to the network made of both stations in short circuit (all stations except the two of the pair are short-circuited); we thus obtain the throughput of the pair in short-circuit. Then we replace the pair by a single station, whose service rate is the throughput just computed. This is repeated until there is only one aggregate station left, at which time the phase 1 terminates and we have computed the throughput $\lambda(K)$ of the original network.

In phase 2, the distributions of states at all stations of interest can be computed using the equivalent service rate theorem (Eq.(8.65)) and normalization to obtain the probability of an empty station; there is no need to use the complement network in this phase.

The number of steps in Phase 1 is order of $\log_2(N)$, where N is the number of stations; the MVA Composition Step is applied in total order of N times (and not 2^N as wrongly assumed in [6]). The complexity of one MVA Composition Step is linear in K , the population size.

In Algorithm 8 in Section 8.9.4 we give a concrete implementation.

8.7 WHAT THIS TELLS US

8.7.1 INSENSITIVITY

Multi-class product form queuing networks are **insensitive** to a number of properties:

- The distribution of service times is irrelevant for all insensitive stations; the stationary distributions of numbers of customers and the throughput depend only on traffic intensities (by means of the visit rates θ_c^s) and on the station functions, which express how rates are shared between classes. The service distribution depends on the class, and classes may be used to introduce correlations in service times. The details of such correlations need not be modelled explicitly, since only traffic intensities matter.
By Little's law, the mean response times are also insensitive (but not the distribution of response time, see Section 8.3.3).
- The nature of the service station plays a role only through its station function. Very different queuing disciplines such as FIFO or global PS, or global LCFSPR with class independent service times have the same station function, hence the same stationary distributions of numbers of customers, throughputs and mean response times also irrelevant as long
- The details of routings are also irrelevant, only the visit rates matter. For example, in Figure 8.11, it makes no difference if we assume that external jobs visit station 1 only once, without feedback.

EXAMPLE 8.14: [INTERNET MODEL \[13\]](#). Internet users as seen by an internet provider are modelled by Bonald and Proutière in [13] as follows (they use a slightly different terminology as they do not interpret a Whittle network as a product form station as we do).

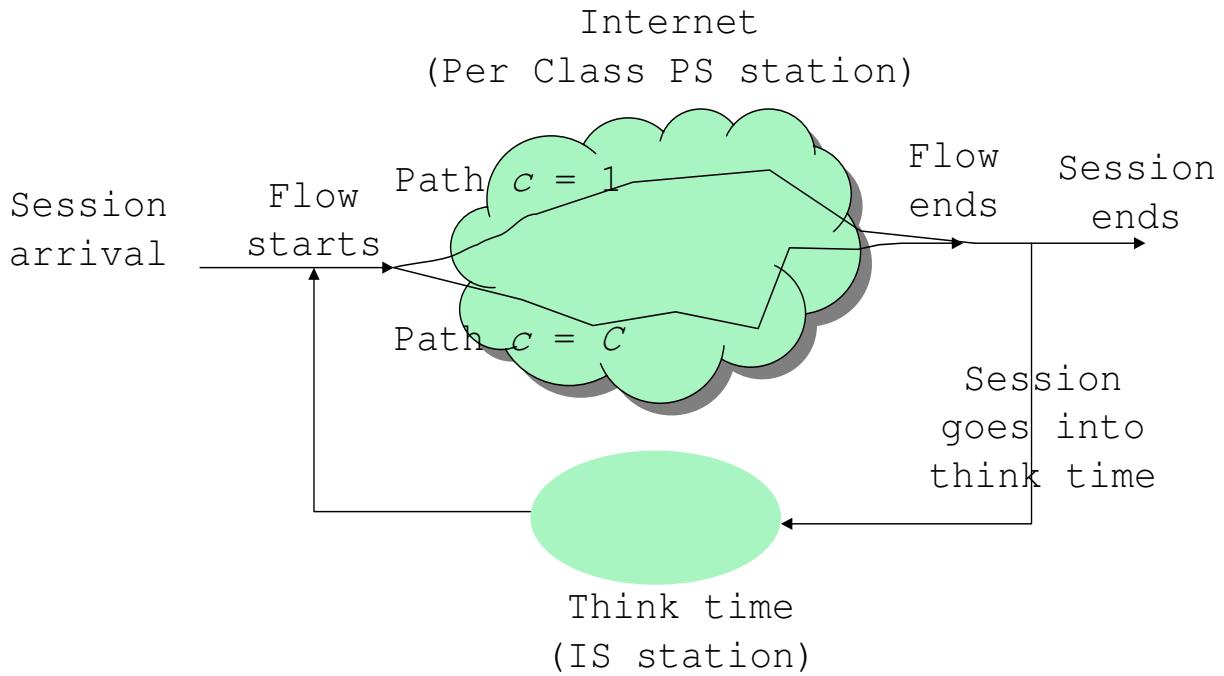


Figure 8.18: Product form queuing network used to model the Internet in [13].

Users sessions arrive as a Poisson process. A session alternates between active and think time. When active, a session becomes a flow and acquires a class, which corresponds to the network path followed by the session (there is one class per possible path). A flow of class c has a service requirement drawn from any distribution with finite mean \bar{S}_c . The network shares its resources between paths according to some “bandwidth” allocation strategy. Let $\mu_c(\vec{n})$ be the rate allocated to class c flows, where $\vec{n} = (n_1, \dots, n_C)$ and n_c is the number of class c flows present in the network. We assume that it derives from a balance function Φ , i.e.

$$\mu_c(\vec{n}) = \frac{\Phi(\vec{n} - \vec{1}_c)}{\Phi(\vec{n})} \quad (8.91)$$

All flows in the same class share the bandwidth allocated to this class fairly, i.e. according to processor sharing.

When a flow completes, it either leaves the network, or mutates and becomes a session in think time. The think time duration has any distribution with a finite mean S_0 . At the end of its think time, a session becomes a flow.

This can be modelled as a single chain open network with two stations: a Per-Class PS station for flow transfers and an IS station for think time, as in Figure 8.18.

A session in think time may keep the class it inherited from the flow. This means that we allow the classes taken by successive flows to be non iid, as is probably the case in reality (for example the next flow of this session might be more likely to take the same path). In fact, we may imagine any dependence, it does not matter as long as the above assumptions hold, since we have a product form queuing network; only the traffic intensities on each flow path matter, as we see next.

With the assumption in Eq.(8.91), flow transfers are represented by means of a per-class processor sharing station with Whittle function $\Phi(\vec{n})$ (this is also called a Whittle network); think times

are represented by a constant rate infinite server station; both are category 1 stations, thus the network has product form.

More precisely, let θ_c be the visit rate at the Per-Class PS station, class c ; it is equal to the number of class c flow arrivals per time unit. Similarly, θ_0 is the number of arrivals of sessions in think time per time unit. Let n_0 be the number of flows in think time; the stationary probability distribution of (n_0, \vec{n}) is, by the product form theorem:

$$\begin{aligned} P(n_0, \vec{n}) &= \eta \Phi(\vec{n}) \prod_{c=1}^C (\bar{S}_c \theta_c)^{n_c} (\theta_0 \bar{S}_0)^{n_0} \\ &= \eta \Phi(\vec{n}) \prod_{c=1}^C \rho_c^{n_c} \rho_0^{n_0} \end{aligned} \quad (8.92)$$

where η is a normalizing constant and $\rho_c = \theta_c \bar{S}_c$, $\rho_0 = \theta_0 \bar{S}_0$ are the traffic intensities.

Eq.(8.92) is a remarkably simple formula. It depends only on the traffic intensities, not on any other property of the session think times or flow transfer times. It holds as long as bandwidth sharing (i.e. the rates $\mu_c(\vec{n})$) derives from a balance function. In [13] it is shown that this is also a necessary condition.

This is used by the authors in [13] to advocate that bandwidth sharing be performed using a balance function. Bandwidth sharing is the function, implemented by a network, which decides the values of $\mu_c(\vec{n})$ for every c and \vec{n} . The set \mathcal{R} of feasible rate vectors $(\mu_c(\vec{n}))_{c=1\dots C}$ is defined by the network constraints. For example, in a wired network with fixed capacities, \mathcal{R} is defined by the constraints $\sum_{c \in \ell} \mu_c < R_l$ where ℓ is a network link, R_l its rate, and “ $c \in \ell$ ” means that a class c flow uses link ℓ . The authors define **balanced fairness** as the unique allocation of rates to classes that (1) derives from a balance function and (2) is optimal in the sense that for any \vec{n} , the rate vector $(\mu_c(\vec{n}))_{c=1\dots C}$ is at the boundary of the set of feasible rate vectors \mathcal{R} . They show that such an allocation is unique; algorithms to compute the balance function are given in [14].

8.7.2 THE IMPORTANCE OF MODELLING CLOSED POPULATIONS

Closed chains give a means to account for feedback in the system, which may provide a different insight than the single queue models in Section 8.3; this is illustrated in Section 8.9, where we see that the conclusion (about the impact of capacity doubling) is radically different if we assume an infinite population or a finite one.

Another useful example is the Engset formula, which we now describe. The Erlang loss formula gives the blocking probability for a system with B servers, general service time and Poisson external arrivals. If the population of tasks using the system is small, there is a feedback loop between the system and the arrival process, since a job that is accepted cannot create an arrival. An alternative to the Erlang loss formula is the model in Figure 8.19, with a finite population of K jobs, a single class of customers, and two stations. Both stations are IS; station 1 represents the service center with B resources, station 2 represents user think time. If station 1 has B customers present, arriving customers are rejected and instantly return to station 2 where they resume service. Service requirements are exponentially distributed. This is equivalent to the form of blocking called partial blocking in Section 8.8.6. This form of blocking requires that routing be reversible; since there are only two stations, the topology is a bus and the routing is reversible, thus the network has product form.

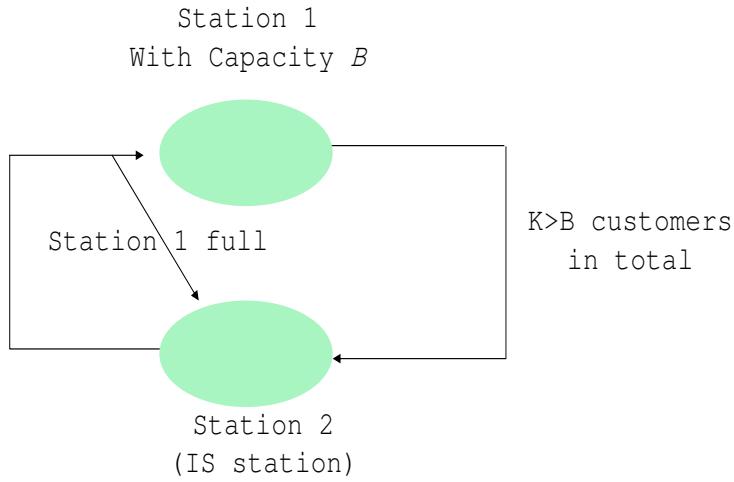


Figure 8.19: Model used to derive the Engset formula

It follows that the probability $P(n|K)$ that there are n customers in service, given that the total population is $K \geq B$, is given by the product form theorem and the station functions for IS:

$$P(n|K) = \frac{1}{\eta} \frac{(\bar{S}^1)^n}{n!} \frac{(\bar{S}^2)^{K-n}}{(K-n)!} \quad (8.93)$$

where η is a normalizing constant, \bar{S}^1 is the average processing time and \bar{S}^2 the average think time. Let $\rho = \frac{\bar{S}^1}{\bar{S}^2}$; it comes:

$$\eta = \sum_{n=0}^B \frac{\rho^n}{n!(K-n)!}$$

The blocking probability $P^0(B|K)$ for is equal to the Palm probability for an arriving customer to find B customers in station 1. By the arrival theorem, it is equal to $P(B|K-1)$. Thus for $K > B$

$$P^0(B|K) = \frac{\frac{\rho^B}{B!(K-B-1)!}}{\sum_{n=0}^B \frac{\rho^n}{n!(K-n-1)!}} \quad (8.94)$$

and $P^0(B|K) = 0$ for $K \leq B$. Eq.(8.94) is called the **Engset formula** and gives the blocking probability for a system with B resources and a population of K . Like the Erlang-loss formula the formula is valid for any distribution of the service time (and of the think time). When $K \rightarrow \infty$, the Engset formula is equivalent to the Erlang-loss formula.

8.8 MATHEMATICAL DETAILS ABOUT PRODUCT-FORM QUEUING NETWORKS

8.8.1 PHASE TYPE DISTRIBUTIONS

For insensitive stations, the service time distribution is assumed to be a **phase type** distribution; this is also called a **mixture of exponentials** or a **mixture of gamma** distribution and is defined next. Note that the product form

theorem implies that the stationary distribution of the network is insensitive to any property of the distribution of service requirement other than its mean; thus it seems plausible to conjecture that the product form network continues to apply if we relax the phase type assumption. This is indeed shown for networks made of Kelly stations and of “Whittle network” stations in [8].

A non negative random variable X is said to have a phase type distribution if there exists a continuous time Markov chain with finite state space $\{0, 1, \dots, I\}$ such that X is the time until arrival into state 0, given some initial probability distribution.

Formally, a phase type distribution with n stages is defined by the non negative sequence $(\alpha_j)_{j=1\dots n}$ with $\sum_j \alpha_j = 1$ and the non negative matrix $(\mu_{j,j'})_{j=1\dots n, j'=0\dots n}$. α_j is the probability that, initially, the chain is in state j and $\mu_{j,j'} \geq 0$ is the transition rate from state j to j' , for $j \neq j'$. Let $F_j(s)$ be the Laplace-Stieltjes transform of the time from now to the next visit to state 0, given that the chain is in state j now. By the Markov property, the Laplace-Stieltjes transform of the distribution we are interested in is $\mathbb{E}(e^{-sX}) = \sum_{j \neq 0} \alpha_j F_j(s)$ for all $s > 0$. To compute $F_j(s)$ we use the following equations, which also follow from the Markov property:

$$\forall j \in \{0, 1, \dots, n\} : \left(s + \sum_{j' \neq j} \mu_{j,j'} \right) F_j(s) = \mu_{j,0} + \sum_{j' \neq j, j' \neq 0} \mu_{j,j'} F_{j'}(s) \quad (8.95)$$

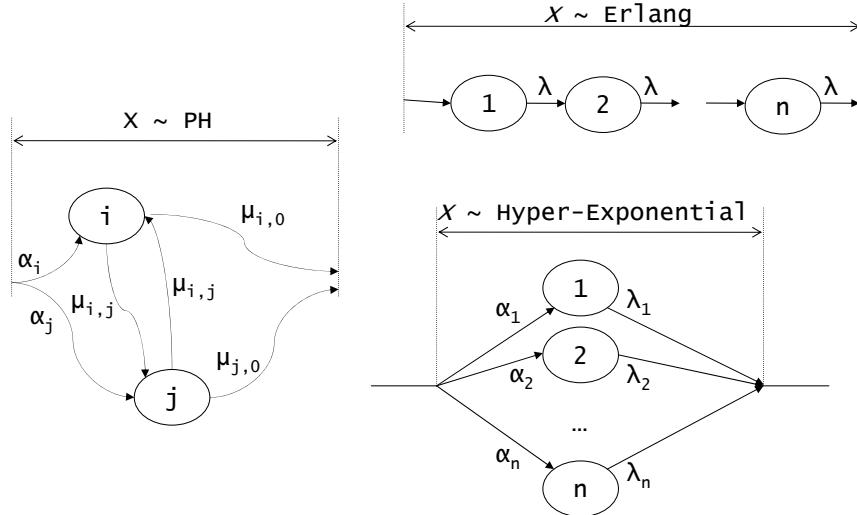


Figure 8.20: Mixtures of Exponential: a Phase Type distribution is the distribution of the time until absorption into state 0 (state 0 represents the exit and is not shown). The Erlang and Hyperexponential are special cases.

Consider for example the *Erlang-n* and *Hyper-Exponential* distributions, which correspond to the Markov chains illustrated in Figure 8.20. The Laplace-Stieltjes transform of the Erlang- n distribution is $F_1(s)$, which is derived from Eq.(8.95):

$$\begin{aligned} (\lambda + s)F_1(s) &= \lambda F_2(s) \\ &\dots \\ (\lambda + s)F_{n-1}(s) &= \lambda F_n(s) \\ (\lambda + s)F_n(s) &= \lambda \end{aligned}$$

and is thus $\left(\frac{\lambda}{\lambda+s}\right)^n$. This could also be obtained by noting that it is the convolution of n exponentials. (Note that this is a special case of Gamma distribution). The PDF is $f(x) = \lambda^n \frac{x^{n-1}}{(n-1)!} e^{-\lambda x}$. The mean is $\bar{S} = \frac{n}{\lambda}$; if we set the mean to a constant and let $n \rightarrow \infty$, the Laplace Stieltjes transform converges, for every $s > 0$, to $e^{-s\bar{S}}$, which is the

Laplace Stieltjes transform of the constant concentrated at \bar{S} . In other words, the Erlang- n distribution can be used to approximate a constant service time.

Similarly, the Laplace Stieltjes transform of the Hyper-Exponential distribution follows immediately from Eq.(8.95) and is $\sum_{j=1^n} \frac{\alpha_j \lambda_j}{\lambda_j + s}$ and the PDF is $f(x) = \sum_{j=1^n} \alpha_j e^{-\lambda_j x}$. This can be used to fit any arbitrary PDF.

8.8.2 MICRO AND MACRO STATES

The state of every station is defined by a *micro-state*, as follows.

In-sensitive Station The micro state is $(\mathcal{B}, \mathcal{J})$ where \mathcal{B} is the state of the station buffer introduced in Section 8.4.2 and \mathcal{J} is a data structure with the same indexing mechanism, which holds the service phase for the customer at this position. In other words, for every index i in the index set of the buffer, \mathcal{B}_i is the class of the customer present at this position, and \mathcal{J}_i is the service phase of the same customer (if there is no customer present at this position, both are 0). A customer at position i receives a service rate $\rho_i(\mathcal{B})$ given by Eq.(8.26). This means that the probability that this customer moves from the service phase $j = \mathcal{J}_i$ to a next phase j' in a time interval of duration dt is $\rho_i(\mathcal{B})\mu_{j,j'}^c dt + o(dt)$ where $c = \mathcal{B}_i$ is this customer's class and $\mu_{j,j'}^c$ is the matrix of transition rates at this station for class c customers, in the phase type representation of service requirement. If the next service phase is $j' = 0$, this customer will leave the station. When a class c customer arrives at this station, it is inserted at position i in the buffer with probability given in Eq.(8.25); the initial stage is set to j with probability α_j^c , the initial stage distribution probability for customers of this class at this station, and \mathcal{J}_i is set to j .

MSCCC Station The micro-state is an ordered sequence of classes (c_1, c_2, \dots, c_M) where M is the number of customers present in the station. When a customer arrives, it is added at the end of the sequence. The customers in service are the first B *eligible* customers; a customer in position m is eligible if and only if there is a token available (i.e. $\sum_{m'=0}^{m-1} \mathbf{1}_{\{G(c_{m'})=g\}} < T_g$ with $g = G(c_m)$) and there is a server available (i.e. $\sum_g \min(T_g, \sum_{m'=0}^{m-1} \mathbf{1}_{\{G(c_{m'})=g\}}) < B$). There is no state information about the service stage, since this category of station requires that the service times be exponentially distributed, hence memoryless. The probability that an eligible customer leaves the station in a time interval of duration dt is $\frac{1}{\bar{S}}r(M)dt + o(dt)$ where $r(M)$ is the rate of this station when M customers are present and \bar{S} is the mean service time (both are independent of the class). Non eligible customers may not leave the station.

The *global micro state* of the network is the sequence (e_1, e_2, \dots, e_S) where e_s is the micro-state of station s . With the assumptions above, this defines a continuous time Markov chain. A network is defined by the population in closed chains, K_C . The *global micro state space*, \mathcal{M} , is the set of all (e_1, e_2, \dots, e_S) that are possible given the rules of each station and

1. the total number of customers in chain C present anywhere in the network is K_C , if C is a closed chain, and is any non negative integer otherwise;
2. if the visit rate θ_c^s is 0 for some station s and class c , then there may not be any customer of class c at station s .

The *macro-state* of station s is the vector $\vec{n}^s = (n_1^s, \dots, n_C^s)$ where n_c^s is the number of class- c customers present at this station. The global macro-state is the collection $(\vec{n}^s)_{s=1 \dots S}$; the global macro state does not define a Markov chain as too much information is lost (for MSCCC stations, we lost the order of customers; for insensitive stations, we lost the service phase). The micro-state description is required to prove theorems, but most formulas of interest are expressed in terms of macro-states. The *global macro state space*, \mathcal{L} , is the set of all $(\vec{n}^s)_{s=1 \dots S} \geq \vec{0}$ such that

1. $\sum_{c \in C, s} n_c^s = K_C$ for every closed chain C ;
2. if the visit rate θ_c^s is 0 for some station s and class c , then $n_c^s = 0$.

8.8.3 MICRO TO MACRO: AGGREGATION CONDITION

All results in the previous sections apply to the macro-state description of the network. In the given form, they require that the aggregation condition holds, which says that

aggregation of state from micro to macro does not introduce non feasible micro states.

This is equivalent to saying that the set \mathcal{M} is fully connected, i.e. any global micro state can be reached from any initial condition in a finite number of transitions of the underlying Markov chain. This is generally true except in pathological cases where the order of customers is preserved throughout the network lifetime. Consider for example a cyclic network with only FIFO stations and one customer per class. The initial order of customers cannot be changed and only states in \mathcal{M} that preserve the initial ordering are feasible. In such a network, product form does hold, but formulas for macro states are different than given in this chapter as the numbers of microstates that give one specific macro-state is smaller.

8.8.4 LOCAL BALANCE IN ISOLATION

The station function can be defined both at the micro and macro levels. Formally, the *station function at micro level* is a function $F(e)$, if it exists, of the micro state e of the function in isolation, such that $F(\emptyset) = 1$, where \emptyset is the empty state, and the stationary probability of state e in the station in isolation is $\eta(\vec{K})F(e)$, where $\eta(\vec{K})$ is a normalizing constant that depends on the total populations of customers K_c for every class c , in the station in isolation.

We say that a station satisfies the property of *Local Balance In Isolation* if the following holds. For every micro-state e and class c :

$$\begin{aligned} & \text{departure rate out of state } e \text{ due to a class } c \text{ arrival} \\ & = \\ & \text{arrival rate into state } e \text{ due to a class } c \text{ departure} \end{aligned} \tag{8.96}$$

In this formula, the rates are with respect to the stationary probability of the station in isolation, as defined earlier. It follows that one must also have

$$\begin{aligned} & \text{departure rate out of state } e \text{ due to a departure or an internal transfer, of any class} \\ & = \\ & \text{arrival rate into state } e \text{ due to an arrival or an internal transfer, of any class} \end{aligned} \tag{8.97}$$

where an internal transfer is a change of state without arrival nor departure (this is for insensitive stations, and is a change of phase for one customer in service). The collection of all these equations is the local balance in isolation. If one finds a station function such that local balance in isolation holds, then this must be the stationary probability of the station in isolation, up to a multiplicative constant.

For example, consider a FIFO station with 1 server and assume that there is one class per chain in the network (i.e. customers do not change class). Let $F(c_1, \dots, c_M)$ be the stationary probability for the station in isolation. Local balance here writes:

$$\begin{aligned} F(c_1, \dots, c_M) \mathbf{1}_{\{\sum_{m=1}^M \mathbf{1}_{\{c_m=c\}} < K_c\}} &= F(c, c_1, \dots, c_M) \mu \text{ for all class } c \\ F(c_1, \dots, c_M) \mu &= F(c_1, \dots, c_{M-1}) \mathbf{1}_{\{\sum_{m=1}^{M-1} \mathbf{1}_{\{c_m=c_M\}} < K_{c_M}\}} \end{aligned}$$

where K_c is the number of class c customers in the system and $\mu = \frac{1}{S}$. The function $F(c_1, \dots, c_M) = \bar{S}^M$ satisfies both of these types of equations, therefore it is equal to the stationary probability of the station in isolation, up to a multiplicative constant. $F(c_1, \dots, c_M) = \bar{S}^M$ is the microscopic station function. The station function $f(\vec{n})$ given earlier follows by aggregation; indeed, let $\mathcal{E}(n_1, \dots, n_C)$ be the set of micro-states of the FIFO station with n_c customers of class c , for every c .

$$f(n_1, \dots, n_C) = \sum_{e \in \mathcal{E}(n_1, \dots, n_C)} \bar{S}^{(n_1 + \dots + n_C)} = \frac{(n_1 + \dots + n_C)!}{n_1! \dots n_C!} \bar{S}^{(n_1 + \dots + n_C)}$$

since $\frac{(n_1 + \dots + n_C)!}{n_1! \dots n_C!}$ is the number of elements of $\mathcal{E}(n_1, \dots, n_C)$. This is exactly the station function for the FIFO station described in Eq.(8.47).

8.8.5 THE PRODUCT FORM THEOREM

The product form theorem in 8.7 is a direct consequence of the following main result.

THEOREM 8.19. Consider a multi-class network with Markov routing and S stations. Assume all S stations satisfy local balance in isolation, and let $F^s(e^s)$ be the station function at micro level for station s , where e^s is the micro state of station s . Then

$$p(e^1, e^2, \dots, e^S) \stackrel{\text{def}}{=} \prod_{s=1}^S F^s(e^s) \quad (8.98)$$

is an invariant measure for the network.

The theorem implies that, if appropriate stability conditions hold, the product $p(e^1, e^2, \dots, e^S)$ must be equal to a stationary probability, up to a normalizing constant. The proof can be found in [78]; see also [44, 10]. It consists in direct verification of the balance equation. More precisely, one shows that, in the network:

$$\begin{aligned} & \text{departure rate out of state } e \text{ due to a departure of any class} \\ & = \\ & \text{arrival rate into state } e \text{ due to an arrival of any class} \end{aligned} \quad (8.99)$$

In this formula, the rates are with respect to the joint network probability of all stations at micro level, obtained by re-normalizing $p()$. Note that the local balance property, as defined in Eq.(8.96), does not, in general, hold inside the network at the micro level.

If the aggregation condition holds, then one can sum up Eq.(8.98) over all micro states for which the network population vector is \vec{n} and obtain Eq.(8.51), which is the macro level product form result. Note that, at the macro-level, one has, in the network, and for any class c :

$$\begin{aligned} & \text{departure rate out of state } e \text{ due to a class } c \text{ departure} \\ & = \\ & \text{arrival rate into state } e \text{ due to a class } c \text{ arrival} \end{aligned} \quad (8.100)$$

In this formula, the rates are with respect to the joint network probability of all stations at macro level. Note the inversion with respect to local balance.

The resulting independence for the open case in Theorem 8.8 therefore also holds for micro-states: in an open network, the micro-states at different stations are independent.

The proof of the product form theorem 8.7 follows immediately from Theorem 8.19 and the fact that all stations in our catalog satisfy the property of local balance in isolation. The proof that MSCCC stations satisfy the local balance property is in [51, 11]. For Kelly-Whittle stations, the result was known before for some specific cases. For the general case, it is novel:

THEOREM 8.20. *Kelly-Whittle stations satisfy local balance in isolation.*

The proof is in Section 8.10.

8.8.6 NETWORKS WITH BLOCKING

It is possible to extend Markov routing to state-dependent routing. In particular, it is possible to allow for some (limited) forms of blocking, as follows. Assume that there are some constraints on the network state, for example, there may be an upper limit to the number of customers in one station. A customer finishing service, or, for an open chain, a customer arriving from the outside, is denied access to a station if accepting this customer would violate any of the constraints. Consider the following two cases:

Transparent Stations with Capacity Limitations The constraints on the network state are expressed by L capacity limitations of the form

$$\sum_{(s,c) \in \mathcal{H}_\ell} n_c^s \leq \Gamma_\ell, \quad \ell = 1 \dots L \quad (8.101)$$

where n_c^s is the number of class c customers present at station s , \mathcal{H}_ℓ is a subset of $\{1, \dots, S\} \times \{1, \dots, C\}$ and $\Gamma_\ell \in \mathbb{N}$. In other words, some stations or groups of stations may put limits on the number of customers of some classes or groups of classes.

If a customer is denied access to station s , she continues her journey through the network, using Markov routing with the fixed matrix Q , until she finds a station that accepts her or until she leaves the network.

Partial Blocking with Arbitrary Constraints The constraints can be of any type. Further, If a customer finishes service and is denied access to station s , she stays blocked in service. More precisely, we assume that service distributions are of phase type, and the customer resumes the last completed service stage. If the customer was arriving from the outside, she is dropped.

Further, we need to assume that Markov routing is *reversible*, which means that

$$\theta_c^s q_{c,c'}^{s,s'} = \theta_{c'}^{s'} q_{c',c}^{s',s} \quad (8.102)$$

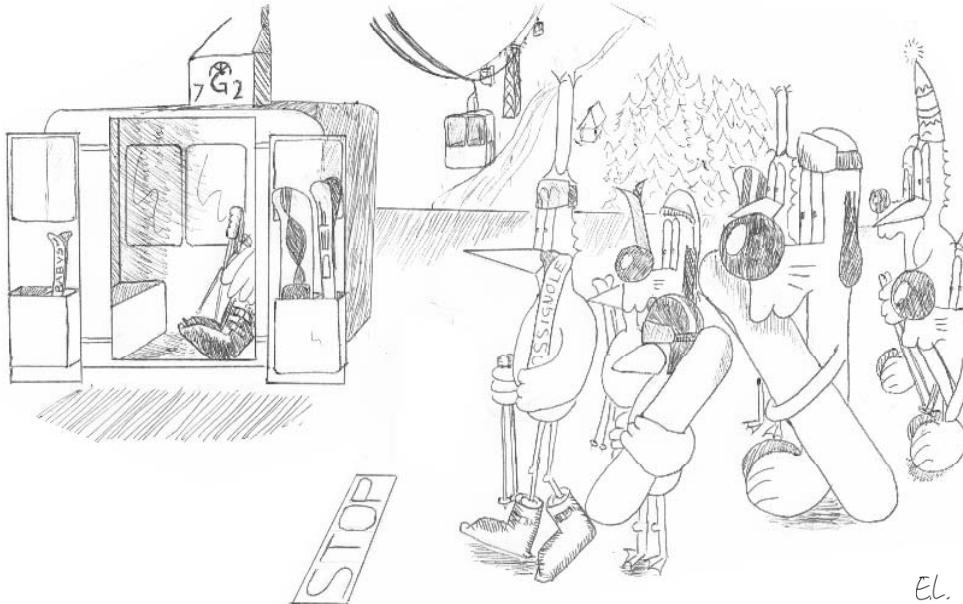
for all s, s', c, c' . Reversibility is a constraint on the topology; bus and star networks give reversible routing, but ring networks do not.

Assume in addition that the service requirements are exponentially distributed (but may be class dependent at insensitive stations). Then the product form theorem continues to apply for these two forms of blocking [80, 58, 44]. There are other cases, too, see [6] and references therein.

There is a more general result: if the service distributions are exponential and the Markov routing is reversible, then the Markov process of global micro-states is also reversible [57]. Let X_t be a continuous time Markov chain with stationary probability $p()$ and state space \mathcal{E} . The process is called *reversible* if $p(e)\mu(e, e') = p(e')\mu(e', e)$ for any two states $e, e' \in \mathcal{E}$, where $\mu(e, e')$ is the rate of transition from e to e' . Reversible Markov chains enjoy the following *truncation property* [44]. Let $\mathcal{E}' \subset \mathcal{E}$ and define the process X'_t by forcing the process to stay within \mathcal{E}' ; this is done by taking some initial state space $e \in \mathcal{E}'$ and setting to 0 the rate of any transition from $e \in \mathcal{E}$ to $e' \in \mathcal{E}'$. Then the restriction of p to \mathcal{E}' is an invariant probability; in particular, if \mathcal{E}' is finite and fully connected, the stationary probability of the truncated process is the restriction of p to \mathcal{E}' , up to a normalizing constant.

Note that setting to 0 the rates of transitions from $e \in \mathcal{E}$ to $e' \in \mathcal{E}'$ is equivalent to saying that we allow the transition from e to e' but then force an immediate, instantaneous return to e ; this explains why we have product form for networks with partial blocking with arbitrary constraints.

8.9 CASE STUDY



In this section we show how the four topics in the previous section can be combined to address a queuing issue. Recently, one could read on the walls of the city where I live the following advertisements for a ski resort: “capacity doubled, waiting time halved”. Does this statement hold? I was intrigued by this sweeping statement, and realized that it can be found repeatedly

in many different situations: doubling the processor speed or doubling the number of cores in a computer, doubling the web front end in a server farm, etc. In the rest of this section we focus on the ski resort example.

First we apply the principles in Chapter 1 and define the goals and factors.

- Goal: evaluate impact of doubling the capacity of a skilift on the response time.
- Factors: c = capacity of skilift in people per second.
- Metrics: response time. A more detailed reflection leads to considering the waiting time, as this is the one that affects customer's perception.
- Load: we consider two load models : (1) heavy burst of arrival (after a train or a bus arrives at the skilift) (2) peak hour stationary regime

8.9.1 DETERMINISTIC ANALYSIS

We can model the skilift as the queuing system illustrated in Figure 8.21. The first queue models the gate; it is a single server queue. Its service time is the time between two passages through the gate, when there is no idle period and is equal to $1/c$. The second queue represents the transportation time. It is an infinite server queue, with no waiting time. Since our performance metric is the waiting time, we may ignore the second queue in the rest of the analysis.

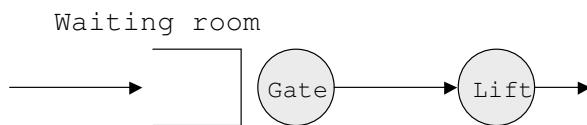


Figure 8.21: Queuing Model of Skilift

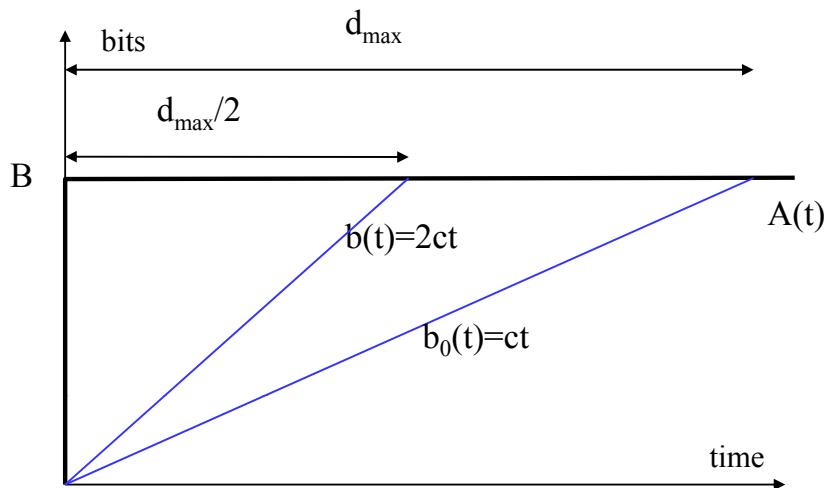


Figure 8.22: Transient Analysis: A burst of skiers arrives at time 0. Impact of doubling the capacity of the skilift.

Assume the arrival of skiers is one single burst (all arrive at the same time). Also assume that all skiers spend the same time to go through the gate, which is roughly true in this scenario. The model in Section 8.1.1 applies, with $A(t)$ = the number of skiers arriving in $[0, t]$ and $D(t)$ = the number of skiers that entered the skilift in $[0, t]$. Thus the delay $d(t)$ is the waiting time, excluding the time spent on the skilift. We also have $\beta(t) = ct$, with c = the capacity of the skilift, in skiers per second. We have $A(t) = B$ for $t \geq 0$. Figure 8.22 shows that doubling the capacity does divide the worst case waiting time by two.

Is the average waiting time also divided by 2? To answer this question we take the viewpoint of an arbitrary customer. We see that the waiting time seen by a customer arriving as number y ($0 \leq y \leq B$) is linear in y , thus the average waiting time is equal to the worst case response time divided by a 2. Here too, doubling the capacity divides the average waiting time by 2.

QUESTION 8.9.1. *In reality, even if the arrival of skiers is bursty, it may not be as simultaneous as we just described. We can account for this by taking $A(t) = kct$ for $0 \leq t \leq t_0$ and $A(t) = A(t_0)$ for $t \geq t_0$, with $k \geq 1$. What is now the conclusion?*¹⁹

8.9.2 SINGLE QUEUE ANALYSIS

Assume now we are observing the system in the middle of the peak hour. We can model the gate as a single queue, with one or perhaps several servers. It is difficult to give a more accurate statement about the arrival process without performing actual measurements. Whatever the details, doubling the capacity halves the utilization factor ρ . A major pattern of single queue systems is the non linearity of response time, as in Figure 8.7.

The effect on response time depends on where we stood on the curve. If the system was close to saturation, as was probably the case, the effect is a large reduction of the average waiting time, probably much larger than 2. With this model, doubling the capacity decreases the waiting time by more than two.

8.9.3 OPERATIONAL ANALYSIS

It is probably unrealistic to assume that a reduction in waiting time has no effect on the arrival rate. A better, though simplified, model is illustrated in Figure 8.23. It is a variant of the interactive user model in Figure 8.3. Here we assume that the mean number \bar{N} of skiers in the system is independent of c .

We apply bottleneck analysis. Let λ be the throughput of the skilift, \bar{S} the time spent serving one customer at the lift and \bar{Z} the time spent going up on the lift or down on the slope and \bar{W} the average waiting time at the lift. We have

$$\begin{cases} \lambda(\bar{W} + \bar{S} + \bar{Z}) = \bar{N} \\ \lambda \leq c \end{cases}$$

and \bar{S} is assumed to be negligible compared to \bar{Z} , thus

$$\bar{W} \gtrapprox \max\left(\frac{\bar{N}}{c} - \bar{Z}, 0\right)$$

¹⁹The response time is reduced by a factor higher than 2.

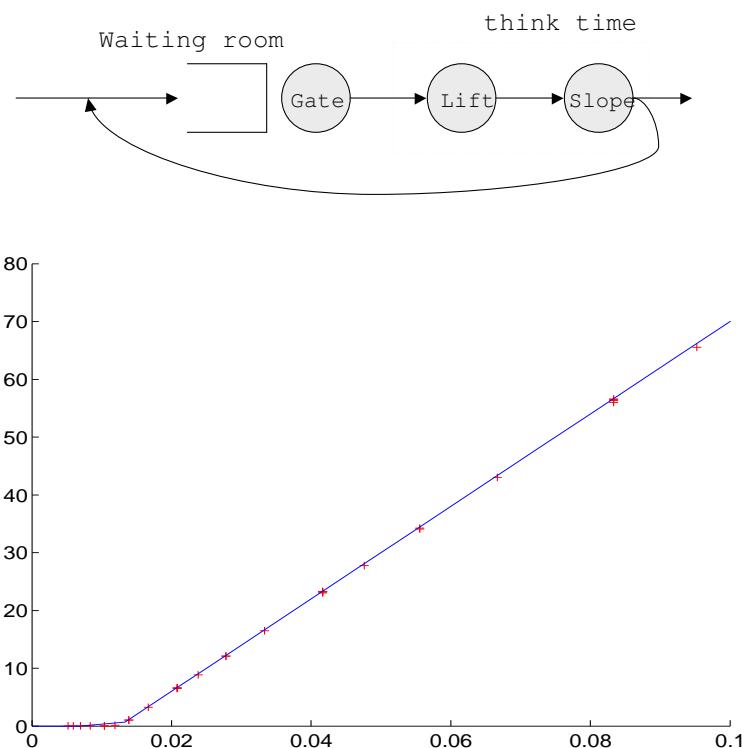


Figure 8.23: First Panel: A Model that accounts for dependency of arrival rate and waiting time. Second panel: Waiting time in minutes for this model versus $\frac{1}{c}$, where c is skilift capacity (in people per minute). The solid line is the approximation by bottleneck analysis. The crosses are obtained by analytical solution of the queuing network model in Figure 8.24, with the following parameters: population size $K = 800$ skiers; number of servers at gate $B \in \{1, 2, \dots, 7, 8\}$; service time at gate $\bar{S} \in \{2.5, 5, 10, 20\}$ seconds; time between visits to the gate $Z = 10$ minutes.

Figure 8.23 shows the approximate bound as a function of $\frac{1}{c}$ for the sake of comparison with Figure 8.7. Points obtained by mean value analysis are also plotted and we see that the bound is in fact a very good approximation.

This strongly suggests that the function f that maps $\frac{1}{c}$ to the average response time is convex; the graph of a convex function is below its chords, thus

$$f\left(\frac{1}{2c}\right) < \frac{1}{2}f\left(\frac{1}{c}\right)$$

and doubling the capacity **does reduce the waiting time by more than 2**.

We also see that a key value is $c^* = \frac{\bar{N}}{\bar{Z}}$. Note that $\frac{1}{\bar{Z}}$ is the rate at which one customer would arrive at the gate if there would be no queuing, thus c^* is the rate of customers if the gate would not delay them. If c is much larger than c^* , the waiting time is small, so doubling the capacity has little effect anyhow. For c much smaller than c^* , the waiting time increases at an almost constant rate. Thus we should target c of the order of c^* , in other words, we should match the capacity of the gate to the “natural” rate c^* .

QUESTION 8.9.2. Assume the system is highly congested before doubling the capacity. What is the reduction in waiting time after doubling capacity? ²⁰

8.9.4 QUEUING NETWORK ANALYSIS

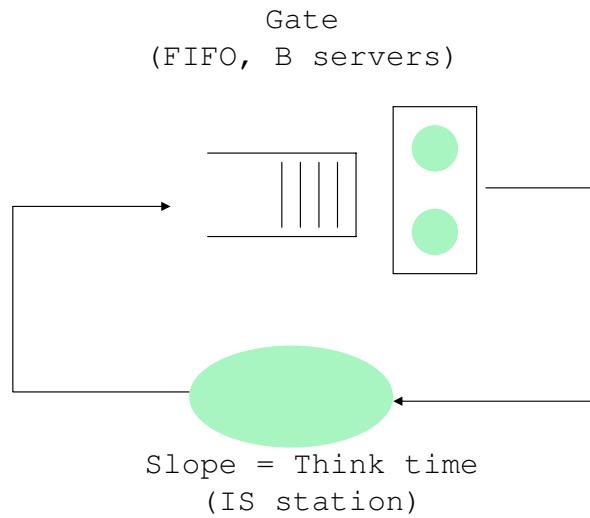


Figure 8.24: A Queuing Network model of Figure 8.23.

We can model the network in Figure 8.23 as a single class, closed product form queuing network as in Figure 8.24. There is no specific assumption on the time spent on the slopes (“think time”); in contrast we need to assume that the service time at the gate is exponentially distributed. Let \bar{S}

²⁰For a highly congested system ($2c$ much smaller than c^*) the offset at 0 becomes negligible and the response time is almost linear in $1/c$. Thus doubling the capacity does reduce the waiting time by 2, roughly speaking – but the system is still congested after doubling the capacity.

be the mean service time at the gate and B the number of servers, so that $c = B/\bar{S}$. The mean service time at the IS station is \bar{Z} .

The total number of customers is fixed and equal to K . Let $\lambda(K)$ and $\bar{W}(K)$ be the throughput and the average waiting time at the gate. By Little's law

$$\lambda(K) (\bar{W}(K) + \bar{S} + \bar{Z}) = K$$

thus

$$\bar{W}(K) = \frac{K}{\lambda(K)} - \bar{S} - \bar{Z} \quad (8.103)$$

We compute $\lambda(K)$ by mean value analysis, which avoids computing the normalizing constants and the resulting overflow problems. Let $P(n|K)$ be the stationary probability that there are n customers present (in service or waiting) at the FIFO station, when the total number of customers is K . The mean value analysis equations are (Section 8.6.7):

$$P(n|K) = P(n-1|K-1) \frac{\lambda(K)}{\mu^*(n)} \text{ if } n \geq 1 \quad (8.104)$$

$$P(0|K) = P(0|K-1) \frac{\lambda(K)}{\lambda^{[1]}(K)} \quad (8.105)$$

$$\sum_{n=0}^K P(n|K) = 1 \quad (8.106)$$

where $\mu^*(n)$ is the equivalent service rate of the FIFO station and $\lambda^{[1]}(K)$ the throughput of the complement of this station. By Table 8.1:

$$\mu^*(n) = \frac{\min(n, B)}{\bar{S}}$$

The complement network is obtained by short circuiting the FIFO station; it consists of the IS station alone. Thus

$$\lambda^{[1]}(K) = \frac{K}{\bar{Z}}$$

The mean value algorithm is given in Algorithm 8. Figure 8.23 and Figure 8.25 shows a few numerical results. The capacity $c = B/\bar{S}$ depends on both the number of FIFO servers B and the service time at the gate \bar{S} . The points in Figure 8.23 are obtained by varying both B and \bar{S} . The figure shows that the bottleneck analysis provides an excellent approximation. Thus this section confirms the conclusions obtained by operational analysis.

8.9.5 CONCLUSIONS

Doubling the capacity does reduce the waiting time by a factor of 2 during bursts of arrivals, and by a factor of 2 or more during the stationary regime. This is independent of whether the capacity increase is by increasing the number of servers or by reducing the service time at the gate.

The findings assume that the arrival rate is not impacted by the capacity increase and does not account for long term effects. Over the long term, a reduction in waiting time might attract more customers and this will in turn increase the waiting time.

Algorithm 8 Implementation of MVA Version 2 to the network in Figure 8.24.

```

1:  $K$  := population size
2:  $p(n)$ ,  $n = 0 \dots K$ : probability that there are  $n$  customers at the FIFO station
3:  $\lambda$ : throughput
4:  $p(0) = 1$ ,  $p(n) = 0$ ,  $n = 1 \dots K$ 
5: for  $k = 1 : K$  do
6:    $p^*(n) = p(n-1)\bar{Z} / \min(n, B)$ ,  $n = 1 \dots k$             $\triangleright$  Unnormalized  $p(n|k)$ , Eq.(8.104)
7:    $p^*(0) = p(0)\bar{Z}/k$                                       $\triangleright$  Unnormalized  $p(0|k)$ , Eq.(8.105)
8:    $\lambda = 1/\sum_{n=0}^k p^*(n)$ 
9:    $p(n) = p^*(n)/\lambda$ ,  $n = 0 \dots k$ 
10: end for

```

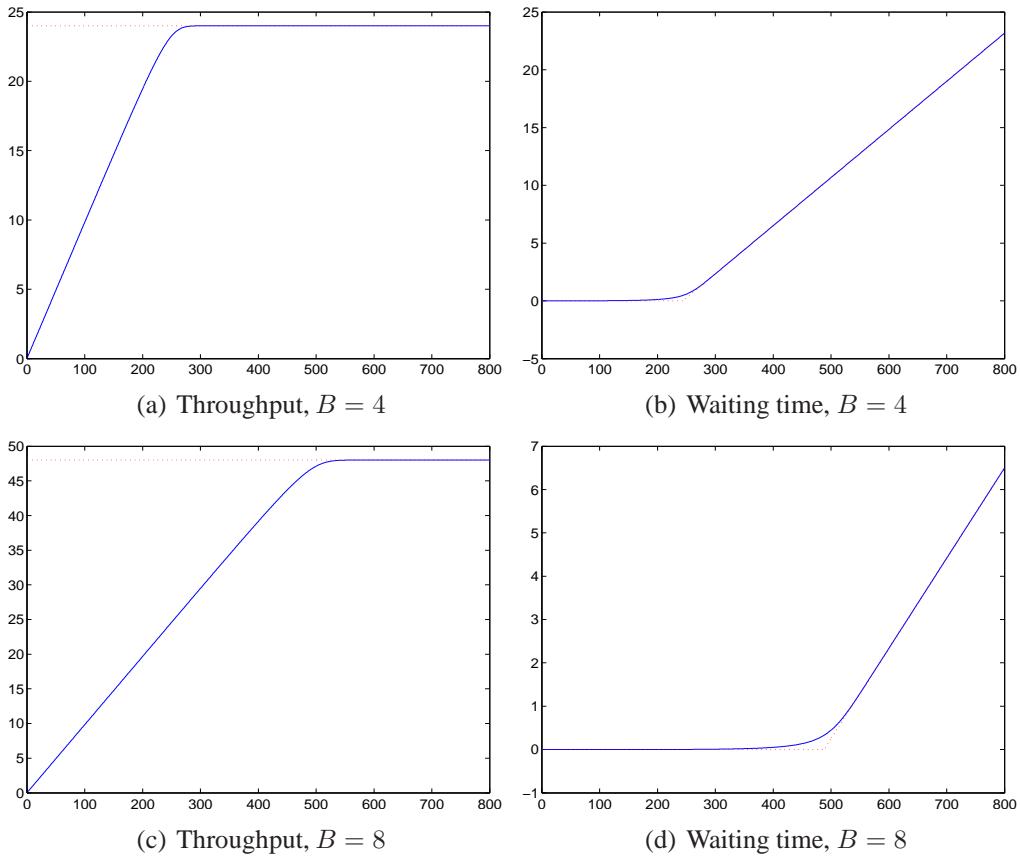


Figure 8.25: Throughput $\lambda(K)$ in customers per minute and waiting times $W(K)$ in minutes for the skilift example in Figure 8.24 with B servers at the gate, versus number of customers K . The results are obtained by analytical solution of the queuing network model (using the MVA algorithm). The dotted lines are the maximum throughput B/\bar{S} and the waiting times predicted by bottleneck analysis. $\bar{S} = 10\text{sec}$ and $\bar{Z} = 10\text{mn}$.

There is an optimal capacity c^* , for any target customer population size K^* (maximum number of customers that the ski resort can accommodate on the slopes), given by $c^* \approx K^*/\bar{Z}$ where \bar{Z} is the mean time between visits to the gate. If the capacity is below c^* , waiting time is large; increasing c beyond c^* brings little benefit on waiting time.

8.10 PROOFS

THEOREM 8.2

Apply Theorem 7.4 to $X(t) = N(t)$ and T_n = the superposition of arrivals and departures. The derivative of $N(t)$ is 0, and the jumps are +1 at instants of arrival, and -1 at instants of departures. Thus $\mathbb{E}^0(\Delta N_0) = 0$. Now $\mathbb{E}^0(\Delta N_0) = +1p_a^0 - 1p_d^0$ where p_a^0 is the probability that an arbitrary point is an arrival [resp. departure]. It follows that $p_a^0 = p_d^0$ and since $p_a^0 + p_d^0 = 1$, it follows that $p_a^0 = p_d^0 = 0.5$, which is not so surprising since there should be in average as many departures as arrivals.

Apply again the theorem to $X(t) = \frac{1-z^{N(t)}}{1-z}$ where z is some arbitrary number in $(0, 1)$. $X(t)$ is constant except at arrival or departure times, thus $X'(t) = 0$. Further, $\Delta X_t = z^{N(t)-1}$ if t is an arrival instant and $\Delta X_t = -z^{N(t)}$ if t is a departure instant. Thus

$$0 = \mathbb{E} \left(z^{N(t)-1} \mid t \text{ is an arrival instant} \right) p_a^0 - \mathbb{E} \left(z^{N(t)} \mid t \text{ is a departure instant} \right) p_d^0$$

Now $N(t)$ is right-hand continuous so $N(t) - 1$ is the number of customers just before t when t is an arrival epoch. Since $p_a^0 = p_d^0$, the distributions of the number of customers just before an arrival and just after a departure are equal.

THEOREM 8.5

We apply Campbell's formula. Let $F(s, t)$ be the random function which returns 1 if $t \geq s$ and the last customer who arrived before or at $-t$ is in node k at time s , else returns 0. By definition of intensity:

$$\lambda_k = \mathbb{E} \left(\sum_{n \in \mathbb{Z}} F(-A_n, 0) \right)$$

where A_n is the point process of customer arrivals. Campbell's formula applied to $F(-t, 0)$ gives:

$$\mathbb{E} \left(\sum_{n \in \mathbb{Z}} F(-A_n, 0) \right) = \lambda \sum_{t \in \mathbb{N}} \mathbb{E}^{-t}(F(t, 0)) = \lambda \sum_{t \in \mathbb{N}} \mathbb{E}^0(F(0, t))$$

where the last part is by stationarity. Thus

$$\lambda_k = \lambda \mathbb{E}^0 \left(\sum_{t \in \mathbb{N}} F(0, t) \right) = \lambda V_k$$

(Total Response Time) Let \bar{N} [resp. \bar{N}_k] be the expected number of customers in the service system [resp. in node k]. We have $\bar{N} = \sum_k \bar{N}_k$. Apply Little' and the Forced Flows laws.

THEOREM 8.20

We consider a Kelly-Whittle station in isolation, i.e. connected to a unit rate per class station, with K_c customers of class c in total. We want to show that local balance holds (at the micro level). The micro state of the station is $(\mathcal{B}, \mathcal{J})$,

where \mathcal{B}_i is the class of the customer in position $i \in \mathcal{I}$ of the station buffer and \mathcal{J}_i is the phase for this customer, in the phase type representation of service times. If there is no customer in position i , we let $\mathcal{B}_i = \mathcal{J}_i = -1$. We assume that the index set \mathcal{I} is enumerable, and that the initial number of occupied positions is finite, so that it remains finite for ever.

Let α_j^c and $\mu_{j,j'}^c$ be the matrices of initial probabilities and transition rates in the phase type representation of service rates for class c , with $j = 1 \dots J^c$ and $j' = 0 \dots J^c$. Without loss of generality we assume $J^C = J$. Recall that $j' = 0$ corresponds to an end of service. For every c , let the array $\theta_j^c, j = 1 \dots J$, be a solution of

$$\begin{aligned} 1 &= \sum_{j=1}^J \theta_j^c \frac{\mu_{j,0}^c}{\bar{\mu}_j^c} \\ \theta_j^c &= \alpha_j^c + \sum_{j'=1}^J \theta_{j'}^c \frac{\mu_{j',j}^c}{\bar{\mu}_{j'}^c} \\ \text{with } \bar{\mu}_j^c &= \sum_{j'=0}^J \mu_{j,j'}^c \end{aligned}$$

so that θ_j^c is the mean number of visits to stage j during one class c customer's service time. Note that the mean service requirement for class c is

$$\bar{S}^c = \sum_{j=1}^J \theta_j^c / \bar{\mu}_j^c \quad (8.107)$$

We will show that the stationary probability of the station in isolation is proportional to

$$F(\mathcal{B}, \mathcal{J}) \stackrel{\text{def}}{=} \Psi(\mathcal{B}) \prod_{i \in \mathcal{I}, \mathcal{B}_i \neq -1} \frac{\theta_{\mathcal{J}_i}^{\mathcal{B}_i}}{\bar{\mu}_{\mathcal{J}_i}^{\mathcal{B}_i}} \quad (8.108)$$

where Ψ is the Whittle function. Clearly, this will imply that F is the station function. Note that the product is always finite. We now show that the equations of local balance Eq.(8.96) and Eq.(8.97) hold. Consider first Eq.(8.96). The departure rate due to a class c arrival is simply $F(\mathcal{B}, \mathcal{J}) \mathbf{1}_{\{n_c(\mathcal{B}) < K_c\}}$, by definition of the station in isolation, where $n_c(\mathcal{B}) \stackrel{\text{def}}{=} \sum_{i \in \mathcal{I}} \mathbf{1}_{\{\mathcal{B}_i=c\}}$ is the number of class c customers. The arrival rate due to a class c departure is 0 if $n_c(\mathcal{B}) < K_c$ (one cannot reach a state where all class c customers are in the station by a departure) and else, by definition of the service rate:

$$\begin{aligned} &= \sum_{i \in \mathcal{I}, j=1 \dots J} F(\text{add}(\mathcal{B}, i, c), \text{add}(\mathcal{J}, j, c)) \gamma(i, \text{add}(\mathcal{B}, i, c)) \frac{\Psi(\text{remove}(\text{add}(\mathcal{B}, i, c), i))}{\Psi(\text{add}(\mathcal{B}, i, c))} \mu_{j,0}^c \\ &= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}, j=1 \dots J} \frac{\theta_j^c}{\bar{\mu}_j^c} \gamma(i, \text{add}(\mathcal{B}, i, c)) \mu_{j,0}^c \\ &= F(\mathcal{B}, \mathcal{J}) \left(\sum_{i \in \mathcal{I}} \gamma(i, \text{add}(\mathcal{B}, i, c)) \right) \left(\sum_{i \in \mathcal{I}, j=1 \dots J} \frac{\theta_j^c}{\bar{\mu}_j^c} \mu_{j,0}^c \right) = F(\mathcal{B}, \mathcal{J}) \end{aligned}$$

thus Eq.(8.96) holds. We now show Eq.(8.97). The left-hand side is

$$F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{B}} \sum_{c=1}^C \sum_{j=1}^J \gamma(i, \mathcal{B}) \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \bar{\mu}_j^c \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}}$$

and the right-hand side is $\text{RHS}_a + \text{RHS}_t$ where the former term corresponds to an arrival, the latter to an internal transfer:

$$\begin{aligned} \text{RHS}_a &= \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j=1}^J F(\text{remove}(\mathcal{B}, i), \text{remove}(\mathcal{J}, i)) \gamma(i, \mathcal{B}) \alpha_j^c \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \\ &= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j=1}^J \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \frac{\bar{\mu}_j^c}{\theta_j^c} \gamma(i, \mathcal{B}) \alpha_j^c \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \end{aligned}$$

We use the notation $\mathcal{B}^{i',i} \stackrel{\text{def}}{=} \text{add}(\text{remove}(\mathcal{B}, i), i', \mathcal{B}_i)$. By hypothesis, $\mathcal{B}^{i,i} = \mathcal{B}$ and $\mathcal{B}^{i',i}$ is the only buffer state \mathcal{B}' such that $\text{remove}(\mathcal{B}', i') = \text{remove}(\mathcal{B}, i)$ and $\mathcal{B}'_i = c$. Also note that $\text{add}(\text{remove}(\mathcal{B}, i), i, \mathcal{B}_i) = \mathcal{B}$. Thus

$$\begin{aligned}
\text{RHS}_t &= \sum_{i,i' \in \mathcal{I}} \sum_{c=1}^C \sum_{j,j'=1}^J F(\mathcal{B}^{i',i}, \text{add}(\text{remove}(\mathcal{J}, i), i', j')) \gamma(i', \mathcal{B}^{i',i}) \frac{\Psi(\text{remove}(\mathcal{B}^{i',i}, i'))}{\Psi(\mathcal{B}^{i',i})} \mu_{j',j}^c \\
&\quad \times \gamma(i, \text{add}(\text{remove}(\mathcal{B}^{i',i}, i'), i, c)) \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \\
&= \sum_{i,i' \in \mathcal{I}} \sum_{c=1}^C \sum_{j,j'=1}^J F(\mathcal{B}^{i',i}, \text{add}(\text{remove}(\mathcal{J}, i), i', j')) \gamma(i', \mathcal{B}^{i',i}) \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B}^{i',i})} \mu_{j',j}^c \\
&\quad \times \gamma(i, \text{add}(\text{remove}(\mathcal{B}, i), i, c)) \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \\
&= F(\mathcal{B}, \mathcal{J}) \sum_{i,i' \in \mathcal{I}} \sum_{c=1}^C \sum_{j,j'=1}^J \frac{\theta_{j'}^c \bar{\mu}_j}{\theta_j^c \bar{\mu}_{j'}} \gamma(i', \text{add}(\text{remove}(\mathcal{B}, i), i', \mathcal{B}_i)) \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mu_{j',j}^c \\
&\quad \times \gamma(i, \mathcal{B}) \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \\
&= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j,j'=1}^J \frac{\theta_{j'}^c \bar{\mu}_j}{\theta_j^c \bar{\mu}_{j'}} \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mu_{j',j}^c \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \gamma(i, \mathcal{B}) \\
&\quad \times \sum_{i' \in \mathcal{I}} \gamma(i', \text{add}(\text{remove}(\mathcal{B}, i), i', c)) \\
&= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j,j'=1}^J \frac{\theta_{j'}^c \bar{\mu}_j}{\theta_j^c \bar{\mu}_{j'}} \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mu_{j',j}^c \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \gamma(i, \mathcal{B}) \\
&= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j=1}^J \frac{\bar{\mu}_j}{\theta_j^c} \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \gamma(i, \mathcal{B}) \sum_{j'=1}^J \frac{\theta_{j'}^c}{\bar{\mu}_{j'}} \mu_{j',j}^c \\
&= F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j=1}^J \frac{\bar{\mu}_j}{\theta_j^c} \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \gamma(i, \mathcal{B}) (\theta_j^c - \alpha_j^c)
\end{aligned}$$

Thus, combining the two:

$$\text{RHS}_a + \text{RHS}_t = F(\mathcal{B}, \mathcal{J}) \sum_{i \in \mathcal{I}} \sum_{c=1}^C \sum_{j=1}^J \bar{\mu}_j \frac{\Psi(\text{remove}(\mathcal{B}, i))}{\Psi(\mathcal{B})} \mathbf{1}_{\{\mathcal{B}_i=c\}} \mathbf{1}_{\{\mathcal{J}_i=j\}} \gamma(i, \mathcal{B})$$

which is equal to the right-hand side as required.

8.11 REVIEW

8.11.1 REVIEW QUESTIONS

QUESTION 8.11.1. *Why are stations of category 1 called “insensitive”?*²¹

QUESTION 8.11.2. *Consider a multi-class queuing network, with FIFO queues, Poisson arrivals and exponential service times; under which condition does it satisfy the hypotheses of the product form theorem?*²²

²¹Their station function depends on the distribution of service time only through the mean.

²²The service time distributions must be independent of the class.

QUESTION 8.11.3. Explain Eq.(8.17) and Eq.(8.21) by the product form theorem.²³

QUESTION 8.11.4. Consider the network in Figure 8.24 and assume there is only one class of customers. Assume that the service requirement at the bottom station is exponential (ν). Say of which category each station is. Write the station functions for both functions and verify the product-form theorem when the number of servers is $B = 1$. Compute the throughput and verify the throughput theorem.²⁴

QUESTION 8.11.5. In Section 8.4 we mention the existence of a network in [16] which is unstable with utilization factor less than 1. Can it be a product-form multi-class queuing network? Why or why not?²⁵

8.11.2 SUMMARY OF NOTATION

SINGLE SERVER QUEUE

Notation	Definition
A/S/s/K	Kendall notation: arrival process/service process/ number of servers/capacity of queue including customers in service
λ	arrival rate
B	number of servers
$\bar{S}, \sigma_S, \mathcal{L}_S$	mean, standard deviation and Laplace Stieltjes transform of service time
$\rho = \frac{\lambda \bar{S}}{B}$	server utilization
N, \bar{N}, σ_N	number of customers in system, its mean and standard deviation
$N_w, \bar{N}_w, \sigma_{N_w}$	number of customers waiting, its mean and standard deviation
R, \bar{R}, σ_R	time spent in system (residence time), its mean and standard deviation
V_k	mean number of visits per customer to node k
W, \bar{W}, σ_W	waiting time, its mean and standard deviation
\bar{Z}	av. think time in interactive user model

QUEUING NETWORKS

²³The M/GI/1/PS queue is an open queuing network with one class of customers and one station, with visit rate equal to λ . The station function for a constant rate PS station is $f(n) = \bar{S}^n$, thus the stationary probability of the M/GI/1/PS queue is $\eta\rho^n$. By normalization, $\eta = 1/(1 - \rho)$, which is Eq.(8.17). Similarly for Eq.(8.21), using the station function of the FIFO station with B servers.

²⁴The ‘Gate’ station is a FIFO station, therefore a station of Category 2. Its station function is $f^1(n) = \frac{1}{\mu^n}$ where $1/\mu$ is its mean service time. The second station is a station of category 1 and its station function is $f^2(n) = \frac{1}{n!\nu^n}$. The stationary probability is $p(n) = \frac{f^1(n)f^2(K-n)}{\eta(K)}$ when there are K customers. The balance equations are

$$p(n)(\mu + (K - n)\nu) = (K - n + 1)\nu p(n - 1)\mathbf{1}_{\{n \geq 1\}} + \mu p(n + 1)\mathbf{1}_{\{n \leq K - 1\}}$$

The verification is by direct computation (the terms match by pair). For the throughput, see Example 7.17.

²⁵It cannot be a product-form multi-class queuing network because they are stable when utilization is less than 1. It violates the assumptions because of FIFO stations with class-dependent service rates.

Notation	Definition
\mathcal{B}	State of buffer in insensitive station, containing the list of customer classes
c	customer class
\mathcal{C}	customer chain; does not change for a given customer
$d(\vec{n})$	combinatorial function used by MSCCC station, Eq.(8.40)
$D(\vec{Z})$	Z -transform of δ , computed by Eq.(8.42)
$f^s(\vec{n}^s)$	station function, Eq.(8.31)
$G^s(\vec{z})$	generating function of station function, Eq.(8.32)
$\mathcal{G}(c)$	token group of class c at an MSCCC station
$\lambda_c^s(\vec{K})$	throughput of class c observed at station s
$\lambda_{\mathcal{C}}(\vec{K})$	throughput of chain \mathcal{C} , Section 8.6.2
\vec{K}	network population vector; $K_{\mathcal{C}}$: number of chain \mathcal{C} customers in network
ν_c^s	external arrival rate of class at station s
$\Phi(\vec{n})$	Balance function at some Kelly-Whittle stations
$\Psi(\mathcal{B})$	Whittle function at Kelly-Whittle station
$q_{c,c'}^{s,s'}$	routing probability, Section 8.4.1
\vec{S}_c^s	mean service requirement at station s for class c customers
T_g	isze of token pool g at MSCCC station
θ_c^s	visit rate to station s , class c (Eq.(8.24))

APPENDIX A

TABLES

The following tables can be used to determine confidence intervals for quantiles (including median), according to Theorem 2.1. For a sample of n iid data points x_1, \dots, x_n , the tables give a confidence interval at the confidence level $\gamma = 0.95$ or 0.99 for the q -quantile with $q = 0.5$ (median), $q = 0.75$ (quartile) and $q = 0.95$. The confidence interval is $[x_{(j)}, x_{(k)}]$, where $x_{(j)}$ is the j th data point sorted in increasing order.

The confidence intervals for $q = 0.05$ and $q = 0.25$ are not given in the tables. They can be deduced by the following rule. Let $[x_{(j)}, x_{(k)}]$ be the confidence interval for the q -quantile given by the table. A confidence interval for the $1 - q$ -quantile is $[x_{(j')}, x_{(k')}]$ with

$$\begin{aligned} j' &= n + 1 - k \\ k' &= n + 1 - j \end{aligned}$$

For example, with $n = 50$, a confidence interval for the third quartile ($q = 0.75$) at confidence level 0.99 is $[x_{(29)}, x_{(45)}]$, thus a confidence interval for the first quartile ($q = 0.25$) at confidence level 0.99 is $[x_{(6)}, x_{(22)}]$.

For small values of n no confidence interval is possible. For large n , an approximate value is given, based on a normal approximation of the binomial distribution.

Note. The tables give p , the actual confidence level obtained, as it is not possible to obtain a confidence interval at exactly the required confidence levels. For example, for $n = 10$ and $\gamma = 0.95$ the confidence interval given by the table is $[X_{(2)}, X_{(9)}]$; the table says that it is in fact a confidence interval at level 0.979.

The values of j and k are chosen such that j and k are as symmetric as possible around $\frac{n+1}{2}$. For example, for $n = 31$ the table gives the interval $[X_{(10)}, X_{(22)}]$. Note that this is not the only interval that can be obtained from the theorem. Indeed, we have:

j	k	$\mathbb{P}(X_{(j)} < m_{0.5} < X_{(k)})$
9	21	0.959
10	22	0.971
11	23	0.959

Thus we have **several** possible confidence intervals. The table simply picked one for which the indices are closest to being symmetrical around the estimated median, i.e. the indices j and k are equally spaced

around $\frac{n+1}{2}$, which is used for estimating the median. In some cases, like $n = 32$, we do not find such an interval exactly; we have for instance:

j	k	$\mathbb{P}(X_{(j)} < m_{0.5} < X_{(k)})$
10	22	0.965
11	23	0.965

Here, the table arbitrarily picked the former.

n	j	k	p
$n \leq 5$: no confidence interval possible.			
6	1	6	0.969
7	1	7	0.984
8	1	7	0.961
9	2	8	0.961
10	2	9	0.979
11	2	10	0.988
12	3	10	0.961
13	3	11	0.978
14	3	11	0.965
15	4	12	0.965
16	4	12	0.951
17	5	13	0.951
18	5	14	0.969
19	5	15	0.981
20	6	15	0.959
21	6	16	0.973
22	6	16	0.965
23	7	17	0.965
24	7	17	0.957
25	8	18	0.957
26	8	19	0.971
27	8	20	0.981
28	9	20	0.964
29	9	21	0.976
30	10	21	0.957
31	10	22	0.971
32	10	22	0.965
33	11	23	0.965
34	11	23	0.959
35	12	24	0.959
36	12	24	0.953
37	13	25	0.953
38	13	26	0.966
39	13	27	0.976
40	14	27	0.962
41	14	28	0.972
42	15	28	0.956
43	15	29	0.968
44	16	29	0.951
45	16	30	0.964
46	16	30	0.960
47	17	31	0.960
48	17	31	0.956
49	18	32	0.956
50	18	32	0.951
51	19	33	0.951
52	19	34	0.964
53	19	35	0.973
54	20	35	0.960
55	20	36	0.970
56	21	36	0.956
57	21	37	0.967
58	22	37	0.952
59	22	38	0.964
60	23	39	0.960
61	23	39	0.960
62	24	40	0.957
63	24	40	0.957
64	24	40	0.954
65	25	41	0.954
66	25	41	0.950
67	26	42	0.950
68	26	43	0.962
69	26	44	0.971
70	27	44	0.959
$n \geq 71$		$\approx [0.50n - 0.980\sqrt{n}]$	0.950
$n \geq 73$		$\approx [0.50n - 1.288\sqrt{n}]$	0.990
$n \leq 7$: no confidence interval possible.			
8	1	8	0.992
9	1	9	0.996
10	1	10	0.998
11	1	11	0.999
12	2	11	0.994
13	2	12	0.997
14	2	12	0.993
15	3	13	0.993
16	3	14	0.996
17	3	15	0.998
18	4	15	0.992
19	4	16	0.996
20	4	16	0.993
21	5	17	0.993
22	5	18	0.996
23	5	19	0.997
24	6	19	0.993
25	6	20	0.996
26	7	20	0.991
27	7	21	0.994
28	7	21	0.992
29	8	22	0.992
30	8	23	0.995
31	8	24	0.997
32	9	24	0.993
33	9	25	0.995
34	10	25	0.991
35	10	26	0.994
36	10	26	0.992
37	11	27	0.992
38	11	27	0.991
39	12	28	0.991
40	12	29	0.994
41	12	30	0.996
42	13	30	0.992
43	13	31	0.995
44	14	31	0.990
45	14	32	0.993
46	15	33	0.992
47	15	33	0.992
48	15	33	0.991
49	16	34	0.991
50	16	35	0.993
51	16	36	0.995
52	17	36	0.992
53	17	37	0.995
54	18	37	0.991
55	18	38	0.994
56	18	38	0.992
57	19	39	0.992
58	20	40	0.991
59	20	40	0.991
60	20	40	0.990
61	21	41	0.990
62	21	42	0.993
63	21	43	0.995
64	22	43	0.992
65	22	44	0.994
66	23	44	0.991
67	23	45	0.993
68	23	45	0.992
69	24	46	0.992
70	24	46	0.991
71	25	47	0.991
72	25	47	0.990

Table A.1: Quantile $q = 50\%$, Confidence Levels $\gamma = 95\%$ (left) and 0.99% (right)

n	j	k	p
$n \leq 10$: no confidence interval possible.			
11	5	11	0.950
12	6	12	0.954
13	7	13	0.952
14	7	14	0.972
15	8	15	0.969
16	9	16	0.963
17	9	17	0.980
18	9	17	0.955
19	10	18	0.960
20	12	20	0.956
21	12	20	0.960
22	13	21	0.956
23	13	22	0.974
24	14	23	0.970
25	14	24	0.982
26	15	24	0.959
27	16	25	0.958
28	17	26	0.954
29	17	27	0.971
30	17	27	0.954
31	18	28	0.958
32	20	30	0.956
33	20	30	0.958
34	21	31	0.955
35	22	32	0.950
36	22	33	0.968
37	22	34	0.979
38	23	34	0.961
39	24	35	0.960
40	25	36	0.958
41	25	37	0.972
42	25	37	0.961
43	26	38	0.963
44	28	40	0.961
45	28	40	0.963
46	28	40	0.951
47	29	41	0.953
48	31	43	0.952
49	31	43	0.954
50	32	44	0.952
51	32	45	0.966
52	33	46	0.964
53	33	47	0.975
54	34	47	0.959
55	35	48	0.959
56	36	49	0.957
57	36	50	0.969
58	37	50	0.951
59	38	51	0.951
60	39	53	0.961
61	39	53	0.963
62	39	53	0.954
63	40	54	0.956
64	42	56	0.955
65	42	56	0.956
66	43	57	0.955
67	44	58	0.952
68	44	59	0.966
69	44	60	0.975
70	45	60	0.962
71	46	61	0.961
72	47	62	0.960
73	47	63	0.971
74	48	63	0.956
75	49	64	0.956
$n \geq 76$		$\approx [0.75n - 0.849\sqrt{n}]$	0.950
$n \geq 82$		$\approx [0.75n - 1.115\sqrt{n}]$	0.990
$n \leq 16$: no confidence interval possible.			
17	7	17	0.992
18	8	18	0.993
19	9	19	0.993
20	10	20	0.993
21	11	21	0.991
22	11	22	0.995
23	12	23	0.994
24	13	24	0.992
25	13	25	0.996
26	13	25	0.993
27	15	27	0.992
28	15	27	0.993
29	16	28	0.992
30	16	29	0.995
31	17	30	0.994
32	18	31	0.993
33	18	32	0.996
34	19	32	0.991
35	20	33	0.990
36	21	35	0.991
37	21	35	0.993
38	21	35	0.990
39	23	37	0.990
40	23	37	0.991
41	23	39	0.997
42	24	39	0.994
43	25	40	0.993
44	26	41	0.992
45	26	42	0.995
46	27	42	0.990
47	28	44	0.993
48	29	45	0.991
49	29	45	0.993
50	29	45	0.990
51	31	47	0.990
52	31	47	0.991
53	31	49	0.996
54	32	49	0.993
55	33	50	0.993
56	34	51	0.992
57	34	52	0.995
58	35	52	0.991
59	36	53	0.990
60	37	55	0.992
61	37	55	0.993
62	37	55	0.991
63	39	57	0.991
64	39	57	0.991
65	40	58	0.991
66	41	59	0.990
67	41	60	0.993
68	42	61	0.993
69	42	62	0.995
70	43	62	0.992
71	44	63	0.991
72	45	64	0.991
73	45	65	0.994
74	45	65	0.992
75	47	67	0.992
76	48	68	0.991
77	48	68	0.992
78	48	68	0.991
79	50	70	0.991
80	50	70	0.991
81	51	71	0.990

Table A.2: Quantile $q = 75\%$, Confidence Levels $\gamma = 95\%$ (left) and 0.99% (right)

n	j	k	p	n	j	k	p
$n \leq 58$: no confidence interval possible.							
59	50	59	0.951	90	76	90	0.990
60	52	60	0.951	91	79	91	0.990
61	53	61	0.953	92	80	92	0.990
62	54	62	0.955	93	81	93	0.991
63	55	63	0.957	94	82	94	0.991
64	56	64	0.958	95	83	95	0.991
65	57	65	0.959	96	84	96	0.992
66	58	66	0.961	97	85	97	0.992
67	59	67	0.962	98	86	98	0.992
68	60	68	0.963	99	87	99	0.992
69	61	69	0.964	100	88	100	0.993
70	62	70	0.964	101	89	101	0.993
71	63	71	0.965	102	90	102	0.993
72	64	72	0.965	103	91	103	0.993
73	65	73	0.966	104	92	104	0.993
74	66	74	0.966	105	93	105	0.993
75	67	75	0.966	106	94	106	0.993
76	68	76	0.966	107	95	107	0.993
77	69	77	0.966	108	96	108	0.993
78	70	78	0.966	109	97	109	0.993
79	71	79	0.966	110	98	110	0.993
80	72	80	0.965	111	99	111	0.993
81	73	81	0.964	112	100	112	0.993
82	74	82	0.964	113	101	113	0.993
83	75	83	0.963	114	102	114	0.992
84	76	84	0.962	115	103	115	0.992
85	77	85	0.961	116	104	116	0.992
86	78	86	0.960	117	105	117	0.992
87	79	87	0.959	118	106	118	0.991
88	80	88	0.957	119	107	119	0.991
89	81	89	0.956	120	108	120	0.991
90	82	90	0.954	121	109	121	0.990
91	83	91	0.952	122	109	122	0.995
92	84	92	0.950	123	110	123	0.995
93	84	93	0.974	124	111	124	0.995
94	85	94	0.973	125	112	125	0.994
95	86	95	0.972	126	113	126	0.994
96	87	96	0.971	127	114	127	0.994
97	88	97	0.970	128	115	128	0.994
98	89	98	0.969	129	116	129	0.993
99	90	99	0.967	130	117	130	0.993
100	91	100	0.966	131	118	131	0.993
101	91	100	0.952	132	119	132	0.992
102	92	101	0.953	133	120	133	0.992
103	93	102	0.953	134	121	134	0.992
104	94	103	0.954	135	122	135	0.991
105	95	104	0.954	136	123	136	0.991
106	96	105	0.954	137	124	137	0.990
107	97	106	0.954	138	124	138	0.995
108	98	107	0.954	139	125	139	0.995
109	99	108	0.954	140	126	140	0.995
110	100	109	0.954	141	127	141	0.994
111	101	110	0.954	142	127	141	0.992
112	102	111	0.953	143	128	142	0.992
113	103	112	0.953	144	129	143	0.992
114	104	113	0.952	145	130	144	0.992
115	105	114	0.951	146	131	145	0.992
116	106	115	0.950	147	133	147	0.992
117	107	117	0.965	148	134	148	0.992
118	108	118	0.963	149	135	149	0.992
119	109	119	0.961	150	136	150	0.991
120	110	120	0.959	151	137	151	0.991
121	110	120	0.967	152	138	152	0.990
122	111	121	0.966	153	138	152	0.992
123	112	122	0.966	154	139	153	0.992
$n \geq 124$		$\approx [0.95n - 0.427\sqrt{n}]$	$\approx [0.95n+1 + 0.427\sqrt{n}]$	$n \geq 155$		$\approx [0.95n - 0.561\sqrt{n}]$	$\approx [0.95n+1 + 0.561\sqrt{n}]$

Table A.3: Quantile $q = 95\%$, Confidence Levels $\gamma = 95\%$ (left) and 0.99% (right)

APPENDIX B

PARAMETRIC ESTIMATION, LARGE SAMPLE THEORY

B.1 PARAMETRIC ESTIMATION THEORY

In this appendix we give a large sample theory which is used for some asymptotic confidence interval computations in Chapter 2 and for the general framework of likelihood ratio tests of Chapter 4.

B.1.1 THE PARAMETRIC ESTIMATION FRAMEWORK.

Consider a data set $x_i, i = 1 \dots, n$, which we view as the realization of a stochastic system (in other words, the output of a simulator). The framework of parametric estimation theory consists in assuming that the parameters of the stochastic system are well defined, but unknown to the observer, who tries to estimate it as well as she can, using the data set.

We assume here that the model has a density of probability, denoted with $f(x_1, \dots, x_n | \theta)$, where θ is the parameter. It is also called the *likelihood* of the observed data. An *estimator* of θ is any function $T()$ of the observed data. A good estimator is one such that, in average, $T(x_1, \dots, x_n)$ is “close” to the true value θ .

EXAMPLE 2.1: **I.I.D. NORMAL DATA.** Assume we can believe that our data is iid and normal with mean μ and variance σ^2 . The likelihood is

$$\frac{1}{(\sqrt{2\pi}\sigma)^n} \exp\left(-\frac{1}{2} \sum_{i=1}^n \frac{(x_i - \mu)^2}{\sigma^2}\right) \quad (\text{B.1})$$

and $\theta = (\mu, \sigma)$. An estimator of θ is $\hat{\theta} = (\hat{\mu}_n, \hat{\sigma}_n)$ given by Theorem 2.3. Another, slightly different estimator is $\hat{\theta}_1 = (\hat{\mu}_n, s_n)$ given by Theorem 2.2.

An estimator provides a random result: for every realization of the data set, a different estimation is produced. The “goodness” of an estimator is captured by the following definitions. Here \vec{X} is

the random data set, $T(\vec{X})$ is the estimator and \mathbb{E}_θ means the expectation when the unknown but fixed parameter value is θ .

- **Unbiased estimator:** $\mathbb{E}_\theta(T(\vec{X})) = \theta$. For example, the estimator $\hat{\sigma}_n^2$ of variance of a normal i.i.d. sample given by Theorem 2.3 is unbiased.
- **Consistent family of estimators:** $\mathbb{P}_\theta(|T(\vec{X}) - \theta|) > \epsilon) \rightarrow 0$ when the sample size n goes to ∞ . For example, the estimator $(\hat{\mu}_n, \hat{\sigma}_n)$ of Theorem 2.3 is consistent. This follows from the weak law of large numbers.

B.1.2 MAXIMUM LIKELIHOOD ESTIMATOR (MLE)

A commonly used method for deriving estimators is that of **Maximum Likelihood**. The maximum likelihood estimator is the value of θ that maximizes the likelihood $f(x_1, \dots, x_n | \theta)$. This definition makes sense if the maximum exists and is unique, which is often true in practice. A formal set of conditions is the regularity condition in Definition B.1.

In Section B.2, we give a result that shows that the MLE for an i.i.d. sample with finite variance is asymptotically unbiased, i.e. the bias tends to 0 as the sample size increases. It is also consistent.

EXAMPLE 2.2: MLE FOR I.I.D. NORMAL DATA. Consider a sample (x_1, \dots, x_n) obtained from a normal i.i.d. random vector (X_1, \dots, X_n) . The likelihood is given by Eq.(B.1). We want to maximize it, where x_1, \dots, x_n are given and $\mu, v = \sigma^2$ are the variables. For a given v , the maximum is reached when $\mu = \hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n x_i$. Let μ have this value and find the value of v that maximizes the resulting expression, or to simplify, the log of it. We thus have to maximize

$$-\frac{n}{2} \ln v - \frac{1}{2v} S_{x,x} + C \quad (\text{B.2})$$

where $S_{x,x} \stackrel{\text{def}}{=} \sum_{i=1}^n (x_i - \hat{\mu}_n)^2$ and C is a constant with respect to v . This is a simple maximization problem in one variable v , which can be solved by computing the derivative. We find that there is a maximum for $v = \frac{S_{x,x}}{n}$. The maximum likelihood estimator of (μ, v) is thus precisely the estimator in Theorem 2.2.

We say that an estimation method **invariant by re-parametrization** if a different parametrization gives essentially the same estimator. More precisely assume we have a method which produces some estimator $T(\vec{X})$ for θ . Assume we re-parametrize the problem by considering that the parameter is $\phi(\theta)$, where ϕ is some invertible mapping. For example, a normal i.i.d. sample can be parametrized by $\theta = (\mu, v)$ or by $\phi(\theta) = (\mu, \sigma)$, with $v = \sigma^2$. The method is called invariant by re-parametrization if the estimator of $\phi(\theta)$ is precisely $\phi(T(\vec{X}))$.

The maximum likelihood method is invariant by re-parametrization. This is because the property of being a maximum is invariant by re-parametrization. It is an important property in our context, since the model is usually not given a priori, but has to be invented by the performance analyst.

A method that provides an unbiased estimator cannot be invariant by re-parametrization, in general. For example, $(\hat{\mu}_n, \hat{\sigma}_n^2)$ of Theorem 2.3 is an unbiased estimator of (μ, σ^2) , but $(\hat{\mu}_n, \hat{\sigma}_n)$ is a **biased** estimator of (μ, σ) (because usually $\mathbb{E}(S)^2 \neq \mathbb{E}(S^2)$ except if S is non-random). Thus, the property of being unbiased is incompatible with invariance by re-parametrization, and may thus be seen as an inadequate requirement for an estimator.

Furthermore, maximum likelihood is also *invariant by reversible data transformation*, i.e. the MLE of θ is the same, whether we look at the data or at a one to one transform, independent of θ . More precisely, assume $\vec{X} = (X_i)_{i=1 \dots n}$ has a joint PDF $f_{\vec{X}}(\vec{x}|\theta)$, and let $\vec{Y} = \varphi(\vec{X})$, with φ a one-to-one, differentiable mapping independent of θ .

Take \vec{X} as data and estimate θ ; we have to maximize $f_{\vec{X}}(\vec{x}|\theta)$ with respect to θ , where $\vec{x} = (x_i)_{i=1 \dots n}$ is the available data. If, instead, we observe $y_i = \varphi(x_i)$ for all i , we have to maximize

$$f_Y(\vec{y}) = \frac{1}{|\varphi'(\vec{x})|} f_{\vec{X}}(\vec{x})$$

where $|\varphi'(\vec{x})|$ is the absolute value of the determinant of the differential of φ (i.e. the Jacobian matrix).

In particular, the MLE is invariant by *re-scaling* of the data. For example, if Y_i is a log-normal sample (i.e. if $Y_i = e^{X_i}$ and $X_i \sim \text{iid } N_{\mu, \sigma^2}$), then the MLE of the parameters μ, θ can be obtained by estimating the mean and standard deviation of $\ln(Y_i)$.

B.1.3 EFFICIENCY AND FISHER INFORMATION

The *efficiency* of an estimator $T(\vec{X})$ of the parameter θ is defined as the expected square error $\mathbb{E}_\theta(\|T(\vec{X}) - \theta\|^2)$ (here we assume that θ takes values in some space Θ where the norm is defined). The efficiency that can be reached by an estimator is captured by the concept of Fisher information, which we now define. Assume first to simplify that $\theta \in \mathbb{R}$. The *observed information* is defined by

$$J(\theta) = -\frac{\partial^2 l(\theta)}{\partial \theta^2}$$

where $l(\theta)$ is the *log-likelihood*, defined by

$$l(\theta) = \ln \text{lik}(\theta) = \ln f(x_1, \dots, x_n | \theta)$$

The *Fisher information*, or *expected information* is defined by

$$I(\theta) = \mathbb{E}_\theta(J(\theta)) = \mathbb{E}_\theta\left(-\frac{\partial^2 l(\theta)}{\partial \theta^2}\right)$$

For an i.i.d. model X_1, \dots, X_n , $l(\theta) = \sum_i \ln f_1(x_i | \theta)$ and thus $I(\theta) = n I_1(\theta)$, where $I_1(\theta)$ is the Fisher information for a one point sample X_1 . The Cramer-Rao theorem says that the efficiency of any *unbiased* estimator is lower bounded by $\frac{1}{I(\theta)}$. Further, under the conditions in Definition B.1, the MLE for an i.i.d. sample is asymptotically maximally efficient, i.e. $\mathbb{E}(\|T(\vec{X}) - \theta\|) / I(\theta)$ tends to 1 as the sample size goes to infinity.

In general, the parameter θ is multi-dimensional, i.e., varies in an open subset Θ of \mathbb{R}^k . Then J and I are symmetric matrices defined by

$$[J(\theta)]_{i,j} = -\frac{\partial^2 l(\theta)}{\partial \theta_i \partial \theta_j}$$

and

$$[I(\theta)]_{i,j} = -\mathbb{E}_\theta\left(\frac{\partial^2 l(\theta)}{\partial \theta_i \partial \theta_j}\right)$$

The Cramer-Rao lower bound justifies the name of “information”. The variance of the MLE is of the order of the Fisher information: the higher the information, the more the sample tells us about the unknown parameter θ . The Fisher information is not the same as entropy, used in information theory. There are some (complicated) relations – see [30, Chapter 16].

In the next section we give a more accurate result, which can be used to give approximate confidence intervals for large sample sizes.

B.2 ASYMPTOTIC CONFIDENCE INTERVALS

Here we need to assume some regularity conditions. Assume the sample comes from an i.i.d. sequence of length n and further, that the following regularity conditions are met.

DEFINITION B.1. *Regularity Conditions for Maximum Likelihood Asymptotics [32].*

1. *The set Θ of values of θ is compact (closed and bounded) and the true value θ_0 is not on the boundary.*
2. *(identifiability) for different values of θ , the densities $f(\vec{x}|\theta)$ are different.*
3. *(regularity of derivatives) There exist a neighborhood B of θ_0 and a constant K such that for $\theta \in B$ and for all $i, j, k, n : \frac{1}{n}\mathbb{E}_\theta(|\partial^3 l_{\vec{X}}(\theta)/\partial\theta_i\partial\theta_j\partial\theta_k|) \leq K$*
4. *For $\theta \in B$ the Fisher information has full rank*
5. *For $\theta \in B$ the interchanges of integration and derivation in $\int \frac{\partial f(\vec{x}|\theta)}{\partial\theta_i} dx = \frac{\partial}{\partial\theta_i} \int f(\vec{x}|\theta) dx$ and $\int \frac{\partial^2 f(x|\theta)}{\partial\theta_i\partial\theta_j} dx = \frac{\partial}{\partial\theta_i} \int \frac{\partial f(\vec{x}|\theta)}{\partial\theta_j} dx$ are valid*

The following theorem is proven in [32].

THEOREM B.1. *Under the conditions in Definition B.1, the MLE exists, converges almost surely to the true value. Further $I(\theta)^{\frac{1}{2}}(\hat{\theta} - \theta)$ converges in distribution towards a standard normal distribution, as n goes to infinity. It follows that, asymptotically:*

1. *the distribution of $\hat{\theta} - \theta$ can be approximated by $N(0, I(\hat{\theta})^{-1})$ or $N(0, J(\hat{\theta})^{-1})$*
2. *the distribution of $2(l(\hat{\theta}) - l(\theta))$ can be approximated by χ_k^2 (where k is the dimension of Θ).*

The quantity $2(l(\hat{\theta}) - l(\theta))$ is called the **likelihood ratio statistic**.

In the examples seen in this book, the regularity conditions are always satisfied, as long as : the true value θ lies within the interior of its domain, the derivatives of $l(\theta)$ are smooth (for example, if the density $f(\vec{x}|\theta)$ has derivatives at all orders) and the matrices $J(\theta)$ and $I(\theta)$ have full rank. If the regularity conditions hold, then we have an equivalent definition of Fisher information:

$$[I(\theta)]_{i,j} \stackrel{\text{def}}{=} -\mathbb{E}_\theta \left(\frac{\partial^2 l(\theta)}{\partial\theta_i\partial\theta_j} \right) = \mathbb{E}_\theta \left(\frac{\partial l(\theta)}{\partial\theta_i} \frac{\partial l(\theta)}{\partial\theta_j} \right)$$

this follows from differentiating with respect to θ the identity $\int f(x|\theta) dx = 1$.

Item 2 is more approximate than item 1, but does not require to compute the second derivative of the likelihood.

Theorem B.1 also holds for non-i.i.d. cases, as long as the Fisher information goes to infinity with the sample size.

EXAMPLE 2.3: FISHER INFORMATION OF NORMAL I.I.D. MODEL. Assume $(X_i)_{i=1\dots n}$ is i.i.d. normal with mean μ and variance σ^2 . The observed information matrix is computed from the likelihood function; we obtain:

$$J = \begin{pmatrix} \frac{n}{\sigma^2} & \frac{2n}{\sigma^3}(\hat{\mu}_n - \mu) \\ \frac{2n}{\sigma^3}(\hat{\mu}_n - \mu) & \frac{-n}{\sigma^2} + \frac{3}{\sigma^4}(S_{xx} + n(\hat{\mu}_n - \mu)^2) \end{pmatrix}$$

and the expected information matrix (Fisher's information) is

$$I = \begin{pmatrix} \frac{n}{\sigma^2} & 0 \\ 0 & \frac{2n}{\sigma^2} \end{pmatrix}$$

The following corollary is used in practice. It follows immediately from the theorem.

COROLLARY B.1 (Asymptotic Confidence Intervals). *When n is large, approximate confidence intervals can be obtained as follows:*

1. *For the i th coordinate of θ , the interval is: $\hat{\theta}_i \pm \eta \sqrt{[I(\hat{\theta})^{-1}]_{i,i}}$ or $\hat{\theta} \pm \eta \sqrt{[J(\hat{\theta})^{-1}]_{i,i}}$, where $N_{0,1}(\eta) = \frac{1+\gamma}{2}$ (for example, with $\gamma = 0.95$, $\eta = 1.96$).*
2. *If θ is in \mathbb{R} : the interval can be defined implicitly as $\{\theta : l(\hat{\theta}) - \frac{\xi}{2} \leq l(\theta) \leq l(\hat{\theta})\}$, where $\chi_1^2(\xi) = \gamma$. For example, with $\gamma = 0.95$, $\xi = 3.84$.*

EXAMPLE 2.4: LAZY NORMAL I.I.D.. Assume our data comes from an i.i.d. normal model X_i , $i = 1, \dots, n$. We compare the exact confidence interval for the mean (from Theorem 2.3) to the approximate ones given by the corollary.

The MLE of (μ, σ) is $(\hat{\mu}_n, s_n)$. The exact confidence interval is

$$\hat{\mu}_n \pm \eta' \frac{\hat{\sigma}_n}{\sqrt{n}}$$

with $\hat{\sigma}_n^2 = S_{x,x}/(n-1)$ and $t_{n-1}(\eta') = \frac{1+\gamma}{2}$.

Now we compute the approximate confidence interval obtained from the Fisher information. We have

$$I(\mu, \sigma)^{-1} = \begin{pmatrix} \frac{\sigma^2}{n} & 0 \\ 0 & \frac{\sigma^2}{2n} \end{pmatrix}$$

thus the distribution of $(\mu - \hat{\mu}_n, \sigma - s_n)$ is approximately normal with 0 mean and covariance matrix $\begin{pmatrix} \frac{\sigma^2}{n} & 0 \\ 0 & \frac{\sigma^2}{2n} \end{pmatrix}$. It follows that $\mu - \hat{\mu}_n$ is approximately $N(0, \frac{s_n^2}{n})$, and an approximate confidence interval is

$$\hat{\mu}_n \pm \eta \frac{s_n}{\sqrt{n}}$$

with $s_n = s_{x,x}/n$ and $N_{0,1}(\eta) = \frac{1+\gamma}{2}$.

n	30	60	120
Exact	$0.7964 - 1.3443$	$0.8476 - 1.2197$	$0.8875 - 1.1454$
Fisher	$0.7847 - 1.3162$	$0.8411 - 1.2077$	$0.8840 - 1.1401$

Table B.1: Confidence Interval for σ for an i.i.d., normal sample of n data points by exact method and asymptotic result with Fisher information (Corollary B.1). The values are the confidence bounds for the ratio $\frac{\sigma}{\hat{\sigma}_n}$ where σ is the true value and $\hat{\sigma}_n$ the estimated standard deviation as in Theorem 2.3.

Thus the use of Fisher information gives the same asymptotic interval for the mean as Theorem 2.2. This is quite general: the use of Fisher information is the generalization of the large sample asymptotic of Theorem 2.2.

We can also compare the approximate confidence interval for σ . The exact interval is given by Theorem 2.3: with probability γ we have

$$\frac{\xi_2}{n-1} \leq \frac{\hat{\sigma}_n^n}{\sigma^2} \leq \frac{\xi_1}{n-1}$$

with $\chi_{n-1}^2(\xi_2) = \frac{1-\gamma}{2}$ and $\chi_{n-1}^2(\xi_1) = \frac{1+\gamma}{2}$. Thus an exact confidence interval for σ is

$$\hat{\sigma}_n \left[\sqrt{\frac{n-1}{\xi_1}}, \sqrt{\frac{n-1}{\xi_2}} \right] \quad (\text{B.3})$$

With Fisher information, we have that $\sigma - s_n$ is approximately $N_{0, \frac{\sigma^2}{2n}}$. Thus with probability γ

$$|\sigma - s_n| \leq \eta \frac{\sigma}{\sqrt{2n}}$$

with $N_{0,1}(\eta) = \frac{1+\gamma}{2}$.

Divide by σ and obtain, after some algebra, that with probability γ :

$$\frac{1}{1 + \frac{\eta}{\sqrt{2n}}} \leq \frac{\sigma}{s_n} \leq \frac{1}{1 - \frac{\eta}{\sqrt{2n}}}$$

Taking into account that $s_n = \sqrt{\frac{n-1}{n}} \hat{\sigma}_n$, we obtain the approximate confidence interval for σ

$$\hat{\sigma}_n \left[\sqrt{\frac{n-1}{n}} \frac{1}{1 + \frac{\eta}{\sqrt{2n}}}, \sqrt{\frac{n-1}{n}} \frac{1}{1 - \frac{\eta}{\sqrt{2n}}}, \right] \quad (\text{B.4})$$

For $n = 30, 60, 120$ and $\gamma = 0.95$, the confidence intervals are as shown in Table B.1, where we compare to exact values; the difference is negligible already for $n = 30$.

QUESTION B.2.1. Which of the following are random variables: $\hat{\theta}, \theta, l(\theta), l(\hat{\theta}), J(\theta), I(\theta), J(\hat{\theta}), I(\hat{\theta})$? ¹

¹In the classical, non Bayesian framework: $\hat{\theta}, l(\theta), l(\hat{\theta}), J(\theta), J(\hat{\theta}), I(\hat{\theta})$ are random variables; θ and $I(\theta)$ are non-random but unknown.

B.3 CONFIDENCE INTERVAL IN PRESENCE OF NUISANCE PARAMETERS

In many cases, the parameter has the form $\theta = (\mu, \nu)$, and we are interested only in μ (for example, for a normal model: the mean) while the remaining element ν , which still needs to be estimated, is considered a nuisance (for example: the variance). In such cases, we can use the following theorem to find confidence intervals.

THEOREM B.2 ([32]). *Under the conditions in Definition B.1, assume that $\Theta = M \times N$, where M, N are open subsets of $\mathbb{R}^p, \mathbb{R}^q$. Thus the parameter is $\theta = (\mu, \nu)$ with $\mu \in M$ and $\nu \in N$ (p is the “dimension”, or number of degrees of freedom, of μ).*

For any μ , let $\hat{\nu}_\mu$ be the solution to

$$l(\mu, \hat{\nu}_\mu) = \max_{\nu} l(\mu, \nu)$$

*and define the **profile log likelihood** pl by*

$$pl(\mu) \stackrel{\text{def}}{=} \max_{\nu} l(\mu, \nu) = l(\mu, \hat{\nu}_\mu)$$

Let $(\hat{\mu}, \hat{\nu})$ be the MLE. If (μ, ν) is the true value of the parameter, the distribution of $2(pl(\hat{\mu}) - pl(\mu))$ tends to χ_p^2 .

An approximate confidence region for μ at level γ is

$$\{\mu \in M : pl(\mu) \geq pl(\hat{\mu}) - \frac{1}{2}\xi\}$$

where $\chi_p^2(\xi) = \gamma$.

The theorem essentially says that we can find an approximate confidence interval for the parameter of interest μ by computing the profile log-likelihood for all values of μ around the estimated value. The estimated value is the one that maximizes the profile log-likelihood. The profile log likelihood is obtained by fixing the parameter of interest μ to some arbitrary value and compute the MLE for the other parameters. A confidence interval is obtained implicitly as the set of values of μ for which the profile log likelihood is close to the maximum. In practice, all of this is done numerically.

EXAMPLE 2.5: LAZY NORMAL I.I.D. REVISITED. Consider the log of the data in Figure 2.12, which appears to be normal. The model is $Y_i \sim i.i.d. N_{\mu, \sigma^2}$ where Y_i is the log of the data. Assume we would like to compute a confidence interval for μ but are too lazy to apply the exact student statistic in Theorem 2.3.

For any μ , we estimate the nuisance parameter σ , by maximizing the log-likelihood:

$$l(\mu, \sigma) = -\frac{1}{2} \left(n \ln \sigma^2 + \frac{1}{\sigma^2} \sum_i (Y_i - \mu)^2 \right)$$

It comes

$$\hat{\sigma}_\mu^2 = \frac{1}{n} \sum_i (Y_i - \mu)^2 = \frac{1}{n} S_{YY} + (\bar{Y} - \mu)^2$$

and thus

$$pl(\mu) \stackrel{\text{def}}{=} l(\mu, \hat{\sigma}_\mu) = -\frac{n}{2}(\ln \hat{\sigma}_\mu^2 + 1)$$

On Figure B.1 we plot $pl(\mu)$. We find $\hat{\mu} = 1.510$ as the point that maximizes $pl(\mu)$. A 95%-confidence interval is obtained as the set $\{pl(\mu) \geq pl(\hat{\mu}) - \frac{1}{2}3.84\}$. We obtain the interval $[1.106, 1.915]$. Compare to the exact confidence interval obtained with Theorem 2.3, which is equal to $[1.103, 1.918]$: the difference is negligible.

QUESTION B.3.1. *Find an analytical expression of the confidence interval obtained with the profile log likelihood for this example and compare with the exact interval.*²

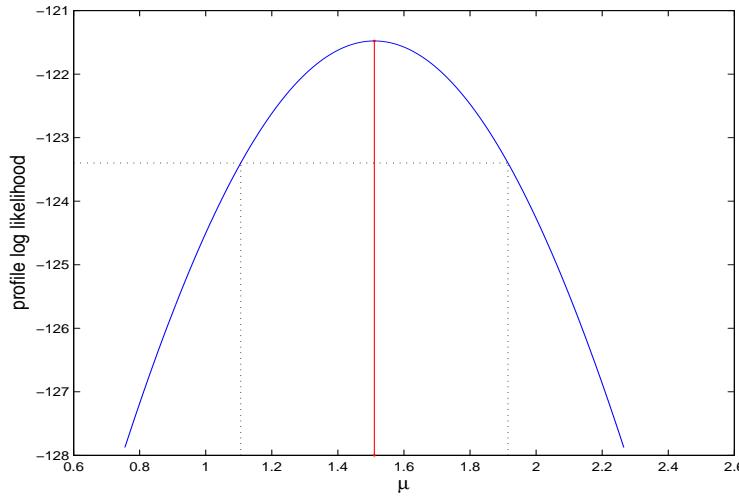


Figure B.1: Profile log-likelihood for parameter μ of the log of the data in Figure 2.12. The confidence interval for μ is obtained by application of Theorem B.2.

EXAMPLE 2.6: RE-SCALING. Consider the data in Figure 2.12, which does not appear to be normal in natural scale, and for which we would like to do a Box-Cox transformation. We would like a confidence interval for the exponent of the transformation.

The transformed data is $Y_i = b_s(X_i)$, and the model now assumes that Y_i is i.i.d. $\sim N_{\mu, \sigma^2}$. We take the unknown parameter to be $\theta = (\mu, \sigma, s)$. The distribution of X_i , under θ is:

$$f_{X_i}(x|\theta) = b'_s(x)f_{Y_i}(b_s(x)|\mu, \sigma) = x^{s-1}h(b_s(x)|\mu, \sigma^2)$$

²The profile log likelihood method gives a confidence interval defined by

$$\frac{(\hat{\mu} - \mu)^2}{\frac{S_{YY}}{n}} \leq e^{\frac{\eta}{n}} - 1 \approx \frac{\eta}{n}$$

Let $t \stackrel{\text{def}}{=} \frac{\hat{\mu} - \mu}{\sqrt{\frac{S_{YY}}{n(n-1)}}}$ be the student statistic. The asymptotic confidence interval can be rewritten as

$$t^2 \leq (n-1)(e^{\frac{\eta}{n}} - 1) \approx \frac{\eta(n-1)}{n}$$

An exact confidence interval is

$$t^2 \leq \xi^2$$

where $\xi = t_{n-1}(1 - \alpha/2)$. For large n , $\xi^2 \approx \eta$ and $\frac{n-1}{n} \approx 1$ so the two intervals are equivalent.

where $h(x|\mu, \sigma^2)$ is the density of the normal distribution with mean μ and variance σ^2 .

The log-likelihood is

$$l(\mu, \sigma, s) = C - n \ln \sigma + \sum_i \left((s-1) \ln x_i - \frac{(b_s(x_i) - \mu)^2}{2\sigma^2} \right)$$

where C is some constant (independent of the parameter). For a fixed s it is maximized by the MLE for a Gaussian sample

$$\hat{\mu}_s = \frac{1}{n} \sum_i b_s(x_i)$$

$$\hat{\sigma}_s^2 = \frac{1}{n} \sum_i (b_s(x_i) - \hat{\mu})^2$$

We can use a numerical estimation to find the value of s that maximizes $l(\hat{\mu}_s, \hat{\sigma}_s, s)$; see Figure B.2 for a plot. The estimated value is $\hat{s} = 0.0041$, which gives $\hat{\mu} = 1.5236$ and $\hat{\sigma} = 2.0563$.

We now give a confidence interval for s , using the asymptotic result in Theorem B.2. A 95% confidence interval is readily obtained from Figure B.2, which gives the interval $[-0.0782, 0.0841]$.

QUESTION B.3.2. *Does the confidence interval justify the log transformation ?*³

Alternatively, by Theorem B.1, we can approximate the distribution of $\hat{\theta} - \theta$ by a centered normal distribution with covariance matrix $J(\hat{\theta})^{-1}$. After some algebra, we compute the Fisher information matrix. We compute the second derivative of the log-likelihood, and estimate the Fisher information by the observed information (i.e. the value of the second derivative at $\theta = \hat{\theta}$). We find:

$$J = \begin{pmatrix} 23.7 & 0 & -77.1 \\ 0 & 47.3 & -146.9 \\ 77.1 & -146.9 & 1291.1 \end{pmatrix}$$

and

$$J^{-1} = \begin{pmatrix} 0.0605 & 0.0173 & 0.0056 \\ 0.0173 & 0.0377 & 0.0053 \\ 0.0056 & 0.0053 & 0.0017 \end{pmatrix}$$

The last term of the matrix is an estimate of the variance of $\hat{s} - s$. The 0.95 confidence interval obtained from a normal approximation is $\hat{s} \pm 1.96\sqrt{0.0017} = [-0.0770, 0.0852]$.

³Yes, since 0 is in the interval.

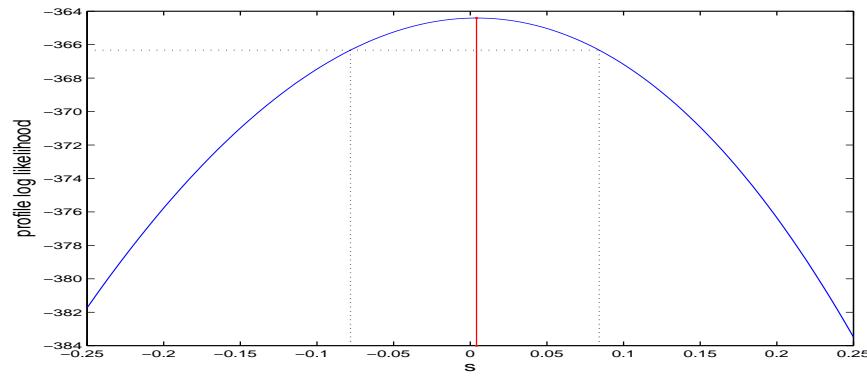


Figure B.2: Profile log-likelihood for Example 2.6, as a function of the Box-Cox exponent s . The maximum likelihood estimator of s is the value that maximizes the profile log likelihood: a confidence interval for s is the set of s for which the profile log likelihood is below the horizontal dashed line.

APPENDIX C

GAUSSIAN RANDOM VECTORS IN \mathbb{R}^n

Contents

C.1 Notation and a Few Results of Linear Algebra	328
C.1.1 Notation	328
C.1.2 Linear Algebra	328
C.2 Covariance Matrix of a Random Vector in \mathbb{R}^n	329
C.2.1 Definitions	329
C.2.2 Properties of Covariance Matrix	330
C.2.3 Choleski's Factorization	330
C.2.4 Degrees of Freedom	330
C.3 Gaussian Random Vector	331
C.3.1 Definition and Main Properties	331
C.3.2 Diagonal Form	332
C.4 Foundations of ANOVA	333
C.4.1 Homoscedastic Gaussian Vector	333
C.4.2 Maximum Likelihood Estimation for Homoscedastic Gaussian Vectors .	333
C.5 Conditional Gaussian Distribution	334
C.5.1 Schur Complement	334
C.5.2 Distribution of \vec{X}_1 given \vec{X}_2	334
C.5.3 Partial Correlation	335
C.6 Proofs	336

C.1 NOTATION AND A FEW RESULTS OF LINEAR ALGEBRA

C.1.1 NOTATION

Unless otherwise specified, we view a vector in \mathbb{R}^n as a column vector, and denote identifiers of vectors with an arrow, as in

$$\vec{X} = \begin{pmatrix} X_1 \\ \vdots \\ X_n \end{pmatrix}$$

The identity matrix is denoted with *Id*.

Matrix transposition is denoted with T , so, for example $\vec{X}^T = (X_1, \dots, X_n)$ and $\vec{X} = (X_1, \dots, X_n)^T$.

The *inner product* of $\vec{u}, \vec{v} \in \mathbb{R}^n$ is

$$\vec{u}^T \vec{v} = \vec{v}^T \vec{u} = \sum_{i=1}^n u_i v_i$$

The *norm* of \vec{u} is, otherwise specified, the euclidian norm, i.e.

$$\|\vec{u}\| = \sqrt{\vec{u}^T \vec{u}}$$

An *orthogonal matrix* U is one that satisfies any one of the following equivalent properties:

1. its columns have unit norm and are orthogonal
2. its rows have unit norm and are orthogonal
3. $UU^T = Id$
4. $U^T U = Id$
5. U has an inverse and $U^{-1} = U^T$.

C.1.2 LINEAR ALGEBRA

If M is a linear subspace of \mathbb{R}^n , the *orthogonal projection* on M is the linear mapping, Π_M , from \mathbb{R}^n to itself such that $\Pi_M(\vec{x})$ is the element of M that minimizes the distance to \vec{x} :

$$\Pi_M(\vec{x}) = \arg \min_{\vec{y} \in M} \|\vec{y} - \vec{x}\| \quad (\text{C.1})$$

$\Pi_M(\vec{x})$ is also the unique element $\vec{y} \in M$ such that $\vec{y} - \vec{x}$ is orthogonal to M . Π_M is symmetric ($\Pi_M = \Pi_M^T$) and idempotent ($\Pi_M^2 = \Pi_M$).

Π_M can always be put in diagonal form as follows:

$$\begin{aligned} \Pi_M &= U^T D U \\ \text{with } D &= \begin{pmatrix} 1 & 0 & \cdots & & \\ & \ddots & & & \\ \cdots & 0 & 1 & 0 & \cdots \\ & & \cdots & 0 & \cdots \\ & & & \ddots & \\ & & \cdots & 0 & \end{pmatrix} \end{aligned} \quad (\text{C.2})$$

where the number of 1s on the diagonal is the dimension of M and U is an orthogonal matrix.

Let H be an $n \times p$ matrix, with $p \leq n$, and M the linear space spanned by the columns of the matrix H , i.e.

$$M = \{\vec{y} \in \mathbb{R}^n : \vec{y} = H\vec{z} \text{ for some } \vec{z} \in \mathbb{R}^p\}$$

If H has full rank (i.e. has rank p) then $H^T H$ has an inverse and

$$\Pi_M = H(H^T H)^{-1}H^T \quad (\text{C.3})$$

C.2 COVARIANCE MATRIX OF A RANDOM VECTOR IN \mathbb{R}^n

C.2.1 DEFINITIONS

Let \vec{X} be a random vector with values in \mathbb{R}^n . If each of the components X_1, \dots, X_n has a well defined expectation, then $\mathbb{E}(\vec{X})$ is defined as

$$\mathbb{E}(\vec{X}) = \begin{pmatrix} \mathbb{E}(X_1) \\ \vdots \\ \mathbb{E}(X_n) \end{pmatrix}$$

For any non-random matrices H and K (with appropriate dimensions such that the matrix products are valid):

$$\mathbb{E}(H\vec{X}K) = H\mathbb{E}(\vec{X})K \quad (\text{C.4})$$

Further, if $\mathbb{E}(X_i^2) < \infty$ for each $i = 1, \dots, n$, the **covariance matrix** of \vec{X} is defined by

$$\Omega = E\left((\vec{X} - \vec{\mu})(\vec{X} - \vec{\mu})^T\right) \quad (\text{C.5})$$

with $\vec{\mu} = \mathbb{E}(\vec{X})$. This is equivalent to

$$\Omega_{i,j} = \text{cov}(X_i, X_j) \stackrel{\text{def}}{=} \mathbb{E}((X_i - \mathbb{E}(X_i))(X_j - \mathbb{E}(X_j))) \quad (\text{C.6})$$

for all $i, j \in \{1, \dots, n\}$.

Further, for any $\vec{u}, \vec{v} \in \mathbb{R}^n$:

$$\mathbb{E}\left((\vec{u}^T(\vec{X} - \vec{\mu}))(\vec{v}^T(\vec{X} - \vec{\mu}))\right) = \vec{u}^T \Omega \vec{v} \quad (\text{C.7})$$

Also

$$\Omega_{i,i} = \text{var}(X_i) \quad (\text{C.8})$$

If \vec{X} and \vec{Y} are random vectors in \mathbb{R}^n and \mathbb{R}^p with a well covariance matrices, the **cross covariance matrix** of X and Y is the $n \times p$ matrix defined by

$$\Gamma = E\left((\vec{X} - \vec{\mu})(\vec{Y} - \vec{\nu})^T\right) \quad (\text{C.9})$$

with $\vec{\mu} = \mathbb{E}(\vec{X})$ and $\mathbb{E}(\vec{Y}) = \vec{\nu}$.

C.2.2 PROPERTIES OF COVARIANCE MATRIX

The covariance matrix is symmetric ($\Omega = \Omega^T$) and **positive semi-definite**. The latter means that $u^T \Omega u \geq 0$ for all $u \in \mathbb{R}^n$, which follows immediately from Eq.(C.7).

If $\vec{X}' = \vec{X} + \nu$ where $\nu \in \mathbb{R}^n$ is a non-random vector, then the covariance matrices of \vec{X}' and \vec{X} are identical.

If $\vec{X}' = AX$ with \vec{X}' a random vector in $\mathbb{R}^{n'}$ and A a non random $n' \times n$ matrix, then the covariance matrix Ω' of \vec{X}' is

$$\Omega' = A\Omega A^T \quad (\text{C.10})$$

Any covariance matrix can be put in standard diagonal form as follows:

$$\Omega = U^T \begin{pmatrix} \lambda_1 & 0 & \cdots & & \\ \ddots & & & & \\ \cdots & 0 & \lambda_r & 0 & \cdots \\ & & \cdots & 0 & \cdots \\ & & & & \ddots \\ & & & & \cdots & 0 \end{pmatrix} U \quad (\text{C.11})$$

where U is an orthogonal matrix ($U^T = U^{-1}$), r is the rank of Ω and $\lambda_1 \geq \dots \geq \lambda_r > 0$.

It follows from this representation that the equation $\vec{x}^T \Omega \vec{x} = 0$ has a non zero solution ($\vec{x} \neq \vec{0}$) if and only if Ω has full rank.

C.2.3 CHOLESKI'S FACTORIZATION

Eq.(C.11) can be replaced by a computationally much less expensive reduction, called **Choleski's Factorization**. This is a polynomial time algorithm for finding a lower triangular matrix L such that $\Omega = LL^T$. Choleski's factorization applies to positive semi-definite matrices and is readily available in many software packages.

C.2.4 DEGREES OF FREEDOM

Let $V = \text{span}(\Omega)$ be the linear sub-space of \mathbb{R}^n spanned by the columns (or rows, since Ω is symmetric) of Ω . Recall that \vec{X} is not necessarily gaussian.

PROPOSITION C.1. \vec{X} is constrained to the affine sub-space parallel to V that contains $\vec{\mu} = \mathbb{E}(\vec{X})$, i.e. $\vec{X} - \vec{\mu} \in V$ with probability 1.

It follows that the number of **degrees of freedom** of \vec{X} (defined in this case as the smallest dimension of an affine space that \vec{X} can be imbedded in) is equal to the dimension of V , namely, the rank of Ω . In particular, if Ω does not have full rank, V has zero mass (its Lebesgue measure is 0) and the integral of any function on V is 0. Thus, it is impossible that \vec{X} has a probability density function. Conversely:

COROLLARY C.1. If \vec{X} has a probability density function (pdf) then its covariance matrix Ω is invertible.

EXAMPLE 3.1: In \mathbb{R}^3 , let the covariance matrix of \vec{X} be

$$\Omega = \begin{pmatrix} a & 0 & a \\ 0 & b & b \\ a & b & a+b \end{pmatrix} \quad (\text{C.12})$$

where a, b are positive constants. The rank is $r = 2$. The linear space generated by the columns of Ω is the plane defined by $x_1 + x_2 - x_3 = 0$. Thus the random vector $\vec{X} = (X_1, X_2, X_3)^T$ is in the plane defined by $X_1 + X_2 - X_3 = \mu_1 + \mu_2 - \mu_3$ where $\vec{\mu} = (\mu_1, \mu_2, \mu_3)^T$.

C.3 GAUSSIAN RANDOM VECTOR

C.3.1 DEFINITION AND MAIN PROPERTIES

DEFINITION C.1. A random vector \vec{X} with values in \mathbb{R}^n is a **gaussian vector** if any of the following is true:

1. For any non random $u \in \mathbb{R}^n$, $u^T \vec{X}$ is a normal random variable.
2. \vec{X} is a non random linear combination of p iid normal random variables, for some $p \in \mathbb{N}$
3. The expectation $\vec{\mu}$ and covariance matrix Ω of \vec{X} are well defined and its **characteristic function** is

$$\phi_{\vec{X}}(\vec{\omega}) \stackrel{\text{def}}{=} \mathbb{E}(e^{j\vec{\omega}^T \vec{X}}) = e^{j\vec{\omega}^T \vec{\mu} - \frac{1}{2}\vec{\omega}^T \Omega \vec{\omega}} \quad (\text{C.13})$$

for all $\vec{\omega} \in \mathbb{R}^n$

EXAMPLE 3.2: The vector $(\epsilon_1, \dots, \epsilon_n)^T$ with $\epsilon_i \sim N_{0, \sigma^2}$, and $\sigma \neq 0$ is a gaussian vector, called **white gaussian noise**. It has $\vec{\mu} = 0$ and $\Omega = \sigma^2 Id$.

The vector

$$\vec{X} = \begin{pmatrix} \sqrt{a}\epsilon_1 \\ \sqrt{b}\epsilon_2 \\ \sqrt{a}\epsilon_1 + \sqrt{b}\epsilon_2 \end{pmatrix}$$

is gaussian with $\vec{\mu} = 0$ and Ω as in Eq.(C.12).

The constant (non-random) vector $\vec{X} = \vec{\mu}$ is gaussian with covariance matrix $\Omega = 0$.

It follows immediately that any (non-random) linear combination of gaussian vectors is gaussian. In particular, if \vec{X} is gaussian and A is a non random matrix, then $A\vec{X}$ is gaussian.

Gaussian vectors are entirely defined by their first and second order properties. In particular:

THEOREM C.1 (Independence equals Non-Correlation). Let \vec{X} [resp. \vec{Y}] be a gaussian random vector in \mathbb{R}^n [resp. \mathbb{R}^p]. \vec{X} and \vec{Y} are independent if and only if their cross-covariance matrix is 0, i.e.

$$\text{cov}(X_i, Y_j) = 0 \text{ for all } i = 1, \dots, n, \quad j = 1, \dots, p$$

Note that this is special to gaussian vectors. For non gaussian random vectors, independence implies non correlation, but the converse may not be true.

THEOREM C.2 (Density). *If Ω is invertible, \vec{X} has a density, given by*

$$f_{\vec{X}}(\vec{x}) = \frac{1}{\sqrt{(2\pi)^n \det \Omega}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Omega^{-1}(\vec{x}-\vec{\mu})}$$

Conversely, we know from Corollary C.1 that if Ω is not invertible (as in the previous example), \vec{X} cannot have a density. A frequent situation where Ω is invertible is the following.

PROPOSITION C.2. *Let $\vec{X} = L\vec{\epsilon}$ where $\vec{X} = (X_1, \dots, X_p)^T$, $\epsilon = (\epsilon_1, \dots, \epsilon_n)^T$ is white gaussian noise and L is a non-random $p \times n$ matrix. The vector \vec{X} is gaussian with covariance matrix $\Omega = LL^T$. The rank of Ω is equal to the rank of L .*

We use this properties in the following case, which arises in the analysis of ARMA and ARIMA processes.

COROLLARY C.2. *Let ϵ_i , $i = 1, \dots, n$ be white gaussian noise. Let $m \leq n$ and X_{n-m+1}, \dots, X_n be defined by*

$$X_i = \sum_{j=1}^i c_{i,j} \epsilon_j \quad \text{for } i = m+1, \dots, n \tag{C.14}$$

with $c_{i,i} \neq 0$. The covariance matrix of $\vec{X} = (X_{n-m+1}, \dots, X_n)$ is invertible.

C.3.2 DIAGONAL FORM

Let where U is the orthogonal transformation in Eq.(C.11) and define $\vec{X}' = U\vec{X}$. The covariance matrix of \vec{X}' is

$$\Omega' = \begin{pmatrix} \lambda_1 & 0 & \cdots & & \\ \ddots & & & & \\ \cdots & 0 & \lambda_r & 0 & \cdots \\ & & \cdots & 0 & \cdots \\ & & & & \ddots \\ & & & & \cdots & 0 \end{pmatrix} \tag{C.15}$$

thus X'_{r+1}, \dots, X'_n have 0 variance and are thus non-random, and X'_i , X'_j are independent (as $\text{cov}(X'_i, X'_j) = 0$ for $i \neq j$).

Since $\vec{X} = U^T \vec{X}'$, it follows that any gaussian random vector is a linear combination of exactly r independent normal random variables, where r is the rank of its covariance matrix. In practice, one obtains such a representation by means of Choleski's factorization. Let $\vec{\epsilon}$ be gaussian white noise sequence with unit variance and let $\vec{Y} = L\vec{\epsilon}$. Then \vec{Y} is a gaussian vector with covariance matrix Ω and 0 expectation (and $\vec{Y} + \vec{\mu}$ is a gaussian vector with expectation $\vec{\mu}$, for any non random vector $\vec{\mu}$). This is used to simulate a random vector with any desired covariance matrix.

EXAMPLE 3.3: One Choleski fatorization of Ω in Eq.(C.12) is $\Omega = LL^T$ with

$$L = \begin{pmatrix} \sqrt{a} & 0 & 0 \\ 0 & \sqrt{b} & 0 \\ \sqrt{a} & \sqrt{b} & 0 \end{pmatrix}$$

Let $\epsilon = (\epsilon_1, \epsilon_2, \epsilon_3)$ be gaussian white noise with unit variance, i.e. such that the covariance matrix of ϵ is equal to Id . Let $\vec{Y} = L\vec{\epsilon} + \vec{\mu}$, i.e.

$$\begin{aligned} Y_1 &= \mu_1 + \sqrt{a}\epsilon_1 \\ Y_2 &= \mu_2 + \sqrt{b}\epsilon_2 \\ Y_3 &= \mu_3 + \sqrt{a}\epsilon_1 + \sqrt{b}\epsilon_2 \end{aligned}$$

then \vec{Y} has covariance matrix Ω and expectation $\vec{\mu}$. This gives a means to simulate a gaussian vector with expectation $\vec{\mu}$ and covariance matrix Ω .

Note that we find, as seen in Example 3.1, that $Y_1 + Y_2 - Y_3$ is a (non random) constant.

C.4 FOUNDATIONS OF ANOVA

C.4.1 HOMOSCEDASTIC GAUSSIAN VECTOR

DEFINITION C.2. A gaussian vector is called *Homoscedastic* with variance σ^2 if its covariance matrix is $\sigma^2 Id$ for some $\sigma > 0$. The expectation $\vec{\mu}$ is not necessarily 0.

Let $\vec{X} = (X_1, X_2, \dots, X_n)^T$. This definition is equivalent to saying that $X_i = \mu_i + \epsilon_i$, with μ_i non-random and $\epsilon_i \sim \text{iid } N_{0, \sigma^2}$.

A homoscedastic gaussian vector always has a density (since its covariance matrix is invertible), given by

$$f_{\vec{X}}(\vec{x}) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} e^{-\frac{1}{2\sigma^2} \|\vec{x} - \vec{\mu}\|^2} \quad (\text{C.16})$$

Homoscedasticity is preserved by orthogonal transformations:

THEOREM C.3. Let U be an orthogonal matrix (i.e. $U^{-1} = U^T$). If \vec{X} is homoscedastic gaussian and $\vec{Y} = U\vec{X}$, then \vec{Y} is also homoscedastic gaussian with same variance.

The following theorem underlies the ANOVA theory:

THEOREM C.4. Let \vec{X} be homoscedastic gaussian in \mathbb{R}^n , $\vec{\mu} = \mathbb{E}(\vec{X})$ and M some linear sub-space of \mathbb{R}^n , of dimension k . Let Π_M be the orthogonal projection on M .

1. $\Pi_M \vec{X}$ and $\vec{Y} = \vec{X} - \Pi_M \vec{X}$ are independent
2. $\|\Pi_M \vec{X} - \Pi_M \vec{\mu}\|^2 \sim \chi_k^2$
3. $\|\vec{Y} - \vec{\mu} + \Pi_M \vec{\mu}\|^2 \sim \chi_{n-k}^2$

where χ_n^2 is the Chi-square distribution with n degrees of freedom.

C.4.2 MAXIMUM LIKELIHOOD ESTIMATION FOR HOMOSCEDASTIC GAUSSIAN VECTORS

THEOREM C.5 (ANOVA). Let \vec{X} be homoscedastic gaussian in \mathbb{R}^n with variance σ^2 and expectation $\vec{\mu}$. Assume that $\vec{\mu}$ is restricted to a linear subspace M of \mathbb{R}^n ; let $k = \dim M$. We are interested in estimating the true values of $\vec{\mu}$ and σ^2 .

1. The MLE of $(\vec{\mu}, \sigma^2)$ is $\hat{\mu} = \Pi_M \vec{X}$, $\hat{\sigma}^2 = \frac{1}{n} \|\vec{X} - \hat{\mu}\|^2$.
2. $\mathbb{E}(\hat{\mu}) = \vec{\mu} = \mathbb{E}(\vec{X})$
3. $\vec{X} - \hat{\mu}$ and $\hat{\mu}$ are independent gaussian random vectors and $\|\vec{X} - \vec{\mu}\|^2 = \|\vec{X} - \hat{\mu}\|^2 + \|\vec{\mu} - \hat{\mu}\|^2$.
4. $\|\vec{X} - \hat{\mu}\|^2 \sim \chi_{n-k}^2 \sigma^2$ and $\|\hat{\mu} - \vec{\mu}\|^2 \sim \chi_k^2 \sigma^2$
5. (Fisher distribution) $\frac{\frac{\|\hat{\mu} - \vec{\mu}\|^2}{k}}{\frac{\|\vec{X} - \hat{\mu}\|^2}{n-k}} \sim F_{k, n-k}$

A special case is the well known estimation for iid normal random variables, used in Theorem 2.3:

COROLLARY C.3. Let $(X_i)_{i=1 \dots n} \sim N(\mu, \sigma^2)$.

1. The MLE of (μ, σ) is $\hat{\mu} = \bar{X} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^n X_i$, $\hat{\sigma}^2 = \frac{1}{n} S_{XX}$, with $S_{XX} \stackrel{\text{def}}{=} \sum_{i=1}^n (X_i - \bar{X})^2$.
2. S_{XX} and \bar{X} are independent and $\sum_i (X_i - \mu)^2 = S_{XX} + n(\bar{X} - \mu)^2$.
3. $S_{XX} \sim \chi_{n-1}^2 \sigma^2$ and $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$.
4. (Student distribution): $\frac{\sqrt{n}(\bar{X} - \mu)}{\sqrt{\frac{S_{XX}}{n-1}}} \sim t_{n-1}$

C.5 CONDITIONAL GAUSSIAN DISTRIBUTION

C.5.1 SCHUR COMPLEMENT

Let M be a square matrix, decomposed in blocks as $M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ where A and D are square matrices (but B and C need not be square) and A is invertible. The **Schur complement** of A in M is defined as

$$S = D - CA^{-1}B$$

It has the following properties.

1. $\det(M) = \det(A) \det(S)$;
2. If M or S is invertible then both are and M^{-1} has the form $\begin{pmatrix} * & * \\ * & S^{-1} \end{pmatrix}$, where $*$ stands for unspecified blocks of appropriate dimensions;
3. If M is symmetrical [resp. positive definite, positive semi-definite] so is S .

C.5.2 DISTRIBUTION OF \vec{X}_1 GIVEN \vec{X}_2

Let \vec{X} be a random vector in $\mathbb{R}^{n_1+n_2}$ and let $\vec{X} = \begin{pmatrix} \vec{X}_1 \\ \vec{X}_2 \end{pmatrix}$, with \vec{X}_i in \mathbb{R}^{n_i} , $i = 1, 2$. We are interested in the conditional distribution of \vec{X}_2 given that $\vec{X}_1 = \vec{x}_1$ (this is typically for prediction purposes). By general results of probability theory, this conditional distribution is well defined; if \vec{X} is gaussian, it turns out that this conditional distribution is also gaussian, as explained next.

Let $\vec{\mu}_2 = \mathbb{E}(\vec{X}_2)$, $\vec{\mu}_1 = \mathbb{E}(\vec{X}_1)$ and decompose the covariance matrix of \vec{X} into blocks as follows.

$$\Omega = \begin{pmatrix} \Omega_{1,1} & \Omega_{1,2} \\ \Omega_{2,1} & \Omega_{2,2} \end{pmatrix}$$

with $\Omega_{i,j}$ (cross-covariance matrix) defined by

$$\Omega_{i,j} = \mathbb{E}((\vec{X}_i - \vec{\mu}_i)(\vec{X}_j - \vec{\mu}_j)^T) \quad i, j = 1, 2$$

Note that $\Omega_{2,1} = \Omega_{1,2}^T$ and X_2 and X_1 are independent if and only if $\Omega_{2,1} = 0$.

THEOREM C.6 ([32]). *Let \vec{X} be a gaussian random vector in $\mathbb{R}^{n_1+n_2}$. The conditional distribution of \vec{X}_2 given that $\vec{X}_1 = \vec{x}_1$ is gaussian. If $\Omega_{1,1}$ is invertible, its expectation is $\vec{\mu}_2 + \Omega_{2,1}\Omega_{1,1}^{-1}(\vec{x}_1 - \vec{\mu}_1)$ and its covariance matrix is the Schur complement of the covariance matrix $\Omega_{1,1}$ of \vec{X}_1 in the covariance matrix Ω of (\vec{X}_1, \vec{X}_2) . In particular, the conditional covariance of \vec{X}_2 given that $\vec{X}_1 = \vec{x}_1$ does not depend on \vec{x}_1 .*

The property that the conditional covariance matrix is independent of \vec{x}_1 holds true only for gaussian vectors, in general. By the properties of covariance matrices, if Ω is invertible, then $\Omega_{1,1}$ also (this follows from the last sentence in Section C.2.2). In this case, by the properties of the Schur complement, the conditional covariance matrix also has full rank.

C.5.3 PARTIAL CORRELATION

Theorem C.6 provides a formula for the conditional covariance. Though it is true only for gaussian vectors, it is used as the basis for the definition of **partial covariance** and **partial correlation**, used in time series analysis. Informally they quantify the residual correlation between X_1 and X_n when we know the values of X_2, \dots, X_{n-1} .

DEFINITION C.3 (Partial Covariance and Correlation, Gaussian case). *Let $\vec{X} = (X_1, X_2, \dots, X_{n-1}, X_n)^T$ be a gaussian vector such that its covariance matrix is invertible. Let*

$$\Gamma = \begin{pmatrix} \gamma_{1,1} & \gamma_{1,n} \\ \gamma_{1,n} & \gamma_{n,n} \end{pmatrix}$$

*be the covariance matrix of the conditional distribution of (X_1, X_n) given $(X_2 = x_2, \dots, X_{n-1} = x_{n-1})$. By Theorem C.6, Γ is independent of x_2, \dots, x_{n-1} . The **partial covariance** of X_1 and X_n is $\gamma_{1,n}$ and the **partial correlation** of X_1 and X_n is*

$$r_{1,n} = \gamma_{1,n} / \sqrt{\gamma_{1,1}\gamma_{n,n}}$$

If X_1, \dots, X_n is a Markov chain, and $n > 1$, then X_n is independent of X_1 , given X_2, \dots, X_{n-1} . In such a case, the partial correlation of X_1 and X_n is 0 (but the covariance of X_1 and X_n is not 0). Partial correlation can be used to test if a Markov chain model is adequate. The following theorem gives a simple way to compute partial correlation.

THEOREM C.7 ([32]). *Let $\vec{X} = (X_1, X_2, \dots, X_{n-1}, X_n)^T$ be a gaussian vector such that its covariance matrix Ω is invertible. The partial correlation of X_1 and X_n is given by*

$$r_{1,n} = \frac{-\tau_{1,n}}{\sqrt{\tau_{1,1}\tau_{n,n}}}$$

where $\tau_{i,j}$ is the (i, j) th term of Ω^{-1} .

The classical definition of partial correlation consists in extending Theorem C.7:

DEFINITION C.4 (Partial Correlation). *Let $\vec{X} = (X_1, X_2, \dots, X_{n-1}, X_n)^T$ be a random vector such that its covariance matrix Ω is well defined and is invertible. The **partial correlation** of X_1 and X_n is defined as*

$$r_{1,n} = \frac{-\tau_{1,n}}{\sqrt{\tau_{1,1}\tau_{n,n}}}$$

where $\tau_{i,j}$ is the (i, j) th term of Ω^{-1} .

C.6 PROOFS

PROPOSITION C.1 Let $v \in \mathbb{R}^n$ be in the kernel of Ω , i.e. $\Omega v = 0$ and let $Z = v^T(X - \mu)$. We have

$$\mathbb{E}(Z^2) = \mathbb{E}(v^T(X - \mu)(X - \mu)^T v) = v^T \Omega v = 0$$

thus $Z = 0$ w.p. 1, i.e. $X - \mu$ is orthogonal to the kernel of Ω .

Since Ω is symmetric, the set of vectors that are orthogonal to its kernel is V , thus $X - \mu \in V$.

PROPOSITION C.2 X is gaussian with covariance matrix LL^T by Eq.(C.10). We now show that the rank of LL^T is equal to the rank of L^T , by showing that LL^T and L^T have same null space. Indeed, if $L^T x = 0$ then $LL^T x = 0$. Conversely, if $LL^T x = 0$ then $x^T LL^T x = \|L^T x\|^2 = 0$ thus $L^T x = 0$. Finally, the rank of a matrix is equal to that of its transpose.

THEOREM C.3 The covariance matrix of $U\vec{X}$ is $U(\sigma^2 Id)U^T = \sigma^2 Id$.

THEOREM C.4 Let $\vec{X}' = \vec{x} - \mu$ and $\vec{Y}' = \vec{X}' - \Pi_M \vec{X}'$. By linearity of Π_M , $\Pi_M \vec{X}'$ and $\Pi_M \vec{X}$ [resp. \vec{Y}' and \vec{Y}] differ by a constant (non-random) vector, thus the cross-covariance Γ of \vec{X} and \vec{Y} is that of \vec{X}' and \vec{Y}' . Thus

$$\begin{aligned} \Gamma &= \mathbb{E}(\Pi_M \vec{X}' \vec{Y}'^T) = \mathbb{E}(\Pi_M \vec{X}' (\vec{X}' - \Pi_M \vec{X}')^T) = \mathbb{E}(\Pi_M \vec{X}' \vec{X}'^T - \Pi_M \vec{X}' \vec{X}'^T \Pi_M^T) \\ &= \Pi_M \mathbb{E}(\vec{X}' \vec{X}'^T) - \Pi_M \mathbb{E}(\vec{X}' \vec{X}'^T) \Pi_M^T \end{aligned}$$

Now $\mathbb{E}(\vec{X}' \vec{X}'^T) = \sigma^2 Id$ thus

$$\Gamma = \sigma^2 \Pi_M - \sigma^2 \Pi_M \Pi_M^T = 0$$

since $\Pi_M = \Pi_M^T$ and $\Pi_M^2 = \Pi_M$. By Theorem C.1, $\Pi_M \vec{X}$ and \vec{Y} are independent. This proves item 1.

Let $Z = \Pi_M X - \Pi_M \mu$. Put Π_M in diagonal form as in Eq.(C.2) and let $\tilde{X} = U^T(\vec{x} - \mu)$ and $\tilde{Z} = U^T Z$, so that

$$\tilde{Z} = D\tilde{X}$$

thus

$$\begin{aligned} \tilde{Z}_i &= \tilde{X}_i \text{ for } i = 1 \dots m \\ \tilde{Z}_i &= 0 \text{ for } i = m + 1 \dots n \end{aligned}$$

Note that

$$\|\tilde{Z}\| = \|\Pi_M \vec{X} - \Pi_M \vec{\mu}\| \tag{C.17}$$

since U is orthogonal. Now \tilde{X} is homoscedastic gaussian with 0 expectation and variance σ^2 (Theorem C.3), thus $\tilde{X}_i \sim \text{iid } N_{0,\sigma^2}$, and finally

$$\left\| \Pi_M \vec{X} - \Pi_M \vec{\mu} \right\|^2 = \sum_{i=1}^m \tilde{X}_i^2$$

This proves item 2, and similarly, item 3.

THEOREM C.5 The log likelihood of an observation $\vec{x} = (x_1, \dots, x_n)^T$ is

$$\begin{aligned} l_{\vec{x}}(\vec{\mu}, \sigma) &= -\frac{N}{2} \ln(2\pi) - N \ln(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^n (x_r - \mu_r)^2 \\ &= -\frac{N}{2} \ln(2\pi) - N \ln(\sigma) - \frac{1}{2\sigma^2} \|\vec{x} - \vec{\mu}\|^2 \end{aligned} \quad (\text{C.18})$$

For a given σ , by Eq.(C.1), the log-likelihood is maximized for $\vec{\mu} = \hat{\mu} = \Pi_M(\vec{x})$, which is independent of σ . Let $\vec{\mu} = \hat{\mu}$ in Eq.(C.18) and maximize with respect to σ , this gives the first item in the theorem. The rest follows from Theorem C.4.

APPENDIX D

DIGITAL FILTERS

Here we review all we need to know for Chapter 5 about causal digital filters. It is a very small subset of signal processing, without any Fourier transform. See for example [83, 75] for a complete and traditional course.

Contents

D.1	Calculus of Digital Filters	340
D.1.1	Backshift Operator	340
D.1.2	Filters	340
D.1.3	Impulse response and Dirac Sequence	341
D.1.4	Composition of Filters, Commutativity	342
D.1.5	Inverse of Filter	342
D.1.6	AR(∞) Representation of Invertible Filter	343
D.1.7	Calculus of Filters	343
D.1.8	z Transform	344
D.2	Stability	345
D.3	Filters with Rational Transfer Function	345
D.3.1	Definition	345
D.3.2	Poles and Zeros	346
D.4	Predictions	349
D.4.1	Conditional Distribution Lemma	349
D.4.2	Predictions	349
D.5	Log Likelihood of Innovation	351
D.6	Matlab Commands	351
D.7	Proofs	352

D.1 CALCULUS OF DIGITAL FILTERS

D.1.1 BACKSHIFT OPERATOR

We consider data sequences of finite, but arbitrary length and call \mathcal{S} the set of all such sequences (i.e. $\mathcal{S} = \bigcup_{n=1}^{\infty} \mathbb{R}^n$). We denote with $\text{length}(X)$ the number of elements in the sequence X .

The **backshift** operator is the mapping B from \mathcal{S} to itself defined by:

$$\begin{aligned}\text{length}(BX) &= \text{length}(X) \\ (BX)_1 &= 0 \\ (BX)_t &= X_{t-1} \quad t = 2, \dots, \text{length}(X)\end{aligned}$$

We usually view a sequence $X \in \mathcal{S}$ as a column vector, so that we can write:

$$B \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix} = \begin{pmatrix} 0 \\ X_1 \\ \vdots \\ X_{n-1} \end{pmatrix} \quad (\text{D.1})$$

when $\text{length}(X) = n$.

If we know that $\text{length}(X) \leq n$, we can express the backshift operator as a matrix multiplication:

$$BX = B_n X \quad (\text{D.2})$$

where B_n is the $n \times n$ matrix:

$$B_n = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & \vdots \\ \vdots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$

Obviously, if $n = \text{length}(X)$ then applying B n times to X gives a sequence of 0s; in matrix form:

$$(B_n)^n = 0 \quad (\text{D.3})$$

D.1.2 FILTERS

DEFINITION D.1. A **filter** (also called “causal filter”, or “realizable filter”) is any mapping, say F , from \mathcal{S} to itself that has the following properties.

1. A sequence of length n is mapped to a sequence of same length.
2. There exists an infinite sequence of numbers h_m , $m = 0, 1, 2, \dots$ (called the filter’s **impulse response**) such that for any $X \in \mathcal{S}$

$$(FX)_t = h_0 X_t + h_1 X_{t-1} + \dots + h_{t-1} X_1 \quad t = 1, \dots, \text{length}(X) \quad (\text{D.4})$$

EXAMPLE 4.1: The backshift operator B is the filter with $h_0 = 0, h_1 = 1, h_2 = h_3 = \dots = 0$.

The identical mapping, Id , is the filter with $h_0 = 1, h_1 = h_2 = \dots = 0$.

The de-seasonalizing filter of order s , R_s , is the filter with $h_0 = \dots = h_{s-1} = 1, h_m = 0$ for $m \geq s$.

The differencing filter at lag s , Δ_s , is the filter with $h_0 = 1, h_s = -1$ and $h_m = 0$ for $m \neq 0$ and $m \neq s$.

Eq.(D.4) can also be expressed as

$$F = \sum_{m=0}^{\infty} h_m B^m \quad (D.5)$$

where $B^0 = Id$. Note that the summation is only apparently infinite, since for a sequence X in \mathcal{S} of length n we have $FX = \sum_{m=0}^{n-1} h_m B^m X$.

In matrix form, if we know that $\text{length}(X) \leq n$ we can write Eq.(D.4) as

$$FX = \begin{pmatrix} h_0 & 0 & \cdots & 0 & 0 \\ h_1 & h_0 & & \vdots & \vdots \\ h_2 & h_1 & \ddots & & \\ \vdots & \vdots & \ddots & h_0 & 0 \\ h_{n-1} & h_{n-2} & \cdots & h_1 & h_0 \end{pmatrix} X \quad (D.6)$$

A filter is called ***Finite Impulse Response (FIR)*** if $h_n = 0$ for n large enough. Otherwise, it is called ***Infinite Impulse Response***.

D.1.3 IMPULSE RESPONSE AND DIRAC SEQUENCE

Define the ***Dirac sequence*** of length n

$$\delta_n = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \quad (D.7)$$

The impulse response of a filter satisfies

$$\begin{pmatrix} h_0 \\ h_1 \\ \vdots \\ h_{n-1} \end{pmatrix} = F\delta_n \quad (D.8)$$

This is used to compute the impulse response if we know some algorithm to compute FX for any X .

D.1.4 COMPOSITION OF FILTERS, COMMUTATIVITY

Let F and F' be filters. The composition of F and F' , denoted with FF' , is defined as the mapping from \mathcal{S} to \mathcal{S} obtained by applying F' first, then F , i.e. such that for any sequence X

$$(FF')(X) = F(F'(X)) \quad (\text{D.9})$$

It can easily be seen that FF' is a filter. Furthermore, **the composition of filters commute**, i.e.

$$FF' = F'F \quad (\text{D.10})$$

The first n terms of the impulse response of FF' can be obtained by

$$\begin{pmatrix} g_0 \\ g_1 \\ \dots \\ g_{n-1} \end{pmatrix} = (FF')\delta_n = F(F'\delta_n) = (F'F)\delta_n = F'(F\delta_n) \quad (\text{D.11})$$

EXAMPLE 4.2: Let us compute the impulse response of FF' when $F = Id - B$ (differencing at lag 1) and $F' = Id - B^5$ (differencing at lag 5). Let n be large:

$$\begin{aligned} F'\delta_n &= (1, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, \dots)^T \\ F(F'\delta_n) &= (1, -1, 0, 0, 0, -1, 1, 0, 0, 0, 0, \dots)^T \end{aligned}$$

thus the impulse response g of FF' is given by

$$\begin{cases} g_0 = g_6 = 1 \\ g_1 = g_5 = -1 \\ \text{else } g_m = 0 \end{cases} \quad (\text{D.12})$$

Alternatively, we can compute in the reverse order and obtain the same result:

$$\begin{aligned} F\delta_n &= (1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, \dots)^T \\ F'(F\delta_n) &= (1, -1, 0, 0, 0, -1, 1, 0, 0, 0, 0, \dots)^T \end{aligned}$$

D.1.5 INVERSE OF FILTER

Since the matrix in Eq.(D.6) is triangular, it is invertible if and only if its diagonal terms are non zero, i.e. if $h_0 \neq 0$, where h_0 is the first term of its impulse response. If this holds, it can also be seen that the reverse mapping F^{-1} is a filter, i.e it satisfies the conditions in Definition (D.1). Thus **a filter F is invertible if and only if $h_0 \neq 0$** .

For example, the inverse filter of the filter with impulse response $h_m = 1$ for $m \geq 0$ (integration filter) is the filter with impulse response $h_0 = 1, h_1 = -1, h_m = 0$ for $m \geq 2$ (differencing filter). This can also be written as

$$\left(\sum_{n=0}^{\infty} B^n \right)^{-1} = Id - B \quad (\text{D.13})$$

D.1.6 AR(∞) REPRESENTATION OF INVERTIBLE FILTER

Let F be an invertible filter and $Y = FX$. Let g_0, g_1, \dots be the impulse response of F^{-1} . We have $X = F^{-1}Y$ thus for $t \geq 1$

$$X_t = g_0 Y_t + g_1 Y_{t-1} + \dots + g_{t-1} Y_1 \quad (\text{D.14})$$

Note that $g_0 = 1/h_0$, thus

$$Y_t = c_0 X_t + c_1 Y_{t-1} + \dots + c_{t-1} Y_1 \quad (\text{D.15})$$

with

$$\begin{cases} c_0 = \frac{1}{g_0} = h_0 \\ c_m = -\frac{g_m}{g_0} = -g_m h_0 \text{ for } m = 1, 2, \dots \end{cases} \quad (\text{D.16})$$

The sequence c_0, c_1, c_2, \dots used in Eq.(D.15) is called the **AR(∞)**¹ representation of F . It can be used to compute the output Y_t as a function of the past output and the current input X_t . This applies to any invertible filter.

If F^{-1} is FIR, then there is some q such that $c_m = 0$ for $m \geq q$. The filter F is called **auto-regressive** of order q (AR(q)).

D.1.7 CALCULUS OF FILTERS

When the filter F' is invertible, the composition $F(F'^{-1})$ is also noted $\frac{F}{F'}$. There is no ambiguity since composition is commutative, namely

$$\frac{F}{F'} = F(F'^{-1}) = (F'^{-1})F \quad (\text{D.17})$$

We have thus defined the product and division of filters. It is straightforward to see that the addition and subtraction of filters are also filters. For example, the filter $F + F'$ has impulse response $h_m + h'_m$ and the filter $-F$ has impulse response $-h_m$.

It is customary to denote the identity filter with 1. With this convention, we can write the differencing filters as

$$\Delta_s = 1 - B^s \quad (\text{D.18})$$

and the de-seasonalizing filter as

$$R_s = 1 + B + \dots + B^{s-1} \quad (\text{D.19})$$

We can also rewrite Eq.(D.13) as

$$\frac{1}{\sum_{n=0}^{\infty} B^n} = 1 - B$$

or

$$\frac{1}{1 - B} = \sum_{n=0}^{\infty} B^n \quad (\text{D.20})$$

¹AR stands for “Auto-Regressive”

The usual manipulations of fractions work as expected, and can be combined with the usual rules for addition, subtraction, multiplication and division (as long as the division is valid, i.e. the filter at the denominator is invertible). Thus, if F and F' are invertible, the inverse of $\frac{F}{F'}$ is $\frac{F'}{F}$:

$$\frac{1}{\frac{F}{F'}} = \frac{F'}{F}$$

EXAMPLE 4.3: We can recover Eq.(D.12) as follows:

$$FF' = (1 - B)(1 - B^5) = 1 - B - B^5 + B^6$$

EXAMPLE 4.4:

$$\frac{\Delta_5}{\Delta_1} = \frac{1 - B^5}{1 - B} = \frac{(1 - B)(1 + B + B^2 + B^3 + B^4)}{1 - B} = 1 + B + B^2 + B^3 + B^4 = R_5 \quad (\text{D.21})$$

If F and G are FIR, then FG , $F + G$ and $F - G$ are also FIR, but F/G is (generally) not.

D.1.8 z TRANSFORM

It is customary in signal processing to manipulate transforms rather than the filters themselves. By definition, the *Transfer Function* of the filter with impulse response h is the power series

$$H(z) = h_0 z + h_1 z^{-1} + h_2 z^{-2} + \dots \quad (\text{D.22})$$

i.e. it is the z transform of the impulse response. This is considered as a formal series, i.e. there is no worry about its convergence for any value of z . Note the use of z^{-1} (customary in signal processing) rather than z (customary in maths).

It follows from the rules on the calculus of filters that using transfer functions is the same as replacing B by z^{-1} everywhere.

EXAMPLE 4.5: The transfer function of the filter

$$F = \frac{Q_0 + Q_1 B + \dots + Q_q B^q}{P_0 + P_1 B + \dots + P_p B^p} \quad (\text{D.23})$$

with $P_0 \neq 0$ is precisely

$$H(z) = \frac{Q_0 + Q_1 z^{-1} + \dots + Q_q z^{-q}}{P_0 + P_1 z^{-1} + \dots + P_p z^{-p}} \quad (\text{D.24})$$

You may find it more convenient to use z -transforms and thus transfer functions if you do not feel comfortable manipulating the backshift operator B (and vice-versa: if you do not like transfer functions, use the backshift operator instead).

D.2 STABILITY

A filter F with impulse response h_n is called **stable**² iff

$$\sum_{n=0}^{\infty} |h_n| < +\infty \quad (\text{D.25})$$

For a sequence $X \in \mathcal{S}$, let $\|X\|_{\infty} = \max_{t=1 \dots \text{length}(X)} |X_t|$. If F is stable and $Y = FX$ then

$$\|Y\|_{\infty} \leq M \|X\|_{\infty} \quad (\text{D.26})$$

where $M = \sum_{n=0}^{\infty} |h_n|$. In other words, if the input to the filter has a bounded magnitude, so does the output. In contrast, if F is not stable, the output of the filter may become infinitely large as the length of the input increases. A stable filter has an impulse response h_n that decays quickly as $n \rightarrow \infty$.

For example, the filter in Eq.(D.21) is stable (as is any FIR filter) and the filter in Eq.(D.20) is not stable.

In practice, if a filter is not stable, we may experience numerical problems when computing its output (Figure D.1).

D.3 FILTERS WITH RATIONAL TRANSFER FUNCTION

D.3.1 DEFINITION

Filters with Rational Transfer Function are filters of the form in Eq.(D.23), or, equivalently, whose transfer function has the form in Eq.(D.24), with $P_0 \neq 0$. Many filters used in practice are of this type. Note that

$$\frac{Q_0 + Q_1B + \cdots + Q_qB^q}{P_0 + P_1B + \cdots + P_pB^p} = \frac{Q'_0 + Q'_1B + \cdots + Q'_qB^q}{1 + P'_1B + \cdots + P'_pB^p}$$

with $Q'_m = \frac{Q_m}{P_0}$ and $P'_m = \frac{P_m}{P_0}$, so we can always assume that $P_0 = 1$.

A filter with rational transfer function can always be expressed as a **Linear constant-coefficient difference** equation. Indeed, consider F as in Eq.(D.23) with $P_0 \neq 0$ and let $Y = FX$. Recall that this is equivalent to

$$Y = (P_0 + P_1B + \cdots + P_pB^p)^{-1} (Q_0 + Q_1B + \cdots + Q_qB^q) X$$

i.e.

$$(P_0 + P_1B + \cdots + P_pB^p) Y = (Q_0 + Q_1B + \cdots + Q_qB^q) X$$

Thus for $t = 1 \dots \text{length}(X)$:

$$P_0 Y_t + P_1 Y_{t-1} + \cdots + P_p Y_{t-p} = Q_0 X_t + Q_1 X_{t-1} + \cdots + Q_q X_{t-q} \quad (\text{D.27})$$

with the usual convention $Y_t = X_t = 0$ for $t \leq 0$. Since $P_0 \neq 0$, this equation can be used to iteratively compute $Y_1 = Q_0 X_1 / P_0$, $Y_2 = (Q_0 X_2 + Q_1 X_1) / P_0$ etc.

²or Bounded Input, Bounded Output (BIBO) -stable

The **impulse response** of F is usually computed by applying `filter` to a Dirac sequence. It may also be computed by Taylor series expansion, using classical rules for Taylor series of functions of one real variable.

EXAMPLE 4.6: The impulse response of the filter $G = \frac{1-2B}{1-B^2}$ is obtained as follows. We use the rule $\frac{1}{1-x} = 1 + x + x^2 + \dots$ and obtain:

$$\begin{aligned}\frac{1-2B}{1-B^2} &= (1-2B)(1+B^2+B^4+\dots) \\ &= 1+B^2+B^4+B^6+\dots -2B-2B^3-2B^5-\dots \\ &= 1-2B+B^2-2B^3+B^4-2B^5\dots\end{aligned}$$

thus the impulse response of G is $(1, -2, 1, -2, 1, -2 \dots)$.

Note that, in general, a filter with rational transfer function has an infinite impulse response.

The **inverse** of the filter F exists if $Q_0 \neq 0$ and is

$$F^{-1} = \frac{P_0 + P_1 B + \dots + P_p B^p}{Q_0 + Q_1 B + \dots + Q_q B^q} \quad (\text{D.28})$$

i.e. is obtained by exchanging numerator and denominator.

D.3.2 POLES AND ZEROES

By definition, the **Poles** of a filter with rational transfer functions are the values of z , other than 0, for which the transfer function is not defined. If the transfer function is in a form that cannot be simplified³ the poles are the zeroes of the denominator. Similarly, the **Zeroes** of the filter are the values of $z \neq 0$ such that $H(z) = 0$.

A filter with rational transfer function is stable iff it has no pole or its **poles are all inside the unit disk**, i.e. have modulus less than 1. This follows from the definition of stability and standard results on the theory of Taylor series of rational fractions in one variable.

The location of zeroes is useful to assess stability of the reverse filter. Indeed, if the filter is invertible (i.e. $Q_0 \neq 0$), then the inverse filter is stable iff all zeroes of the original filter are within the unit disk.

EXAMPLE 4.7: NUMERICAL STABILITY OF INVERSE FILTER. Consider the filter

$$F = \frac{0.1 + 0.2B + 0.3B^2}{1 - 0.2B} \quad (\text{D.29})$$

We apply the filter to an input sequence X (thin line on Figure D.1) and obtain the output sequence Y (thick line). F is a filter with rational transfer function, and is equivalent to the linear constant-coefficient difference equation:

$$Y_t = 0.1X_t + 0.2X_{t-1} + 0.3X_{t-2} + 0.2Y_{t-1}$$

³i.e. of the form $\frac{p(z^{-1})}{q(z^{-1})}$ where p, q are polynomials with no common root

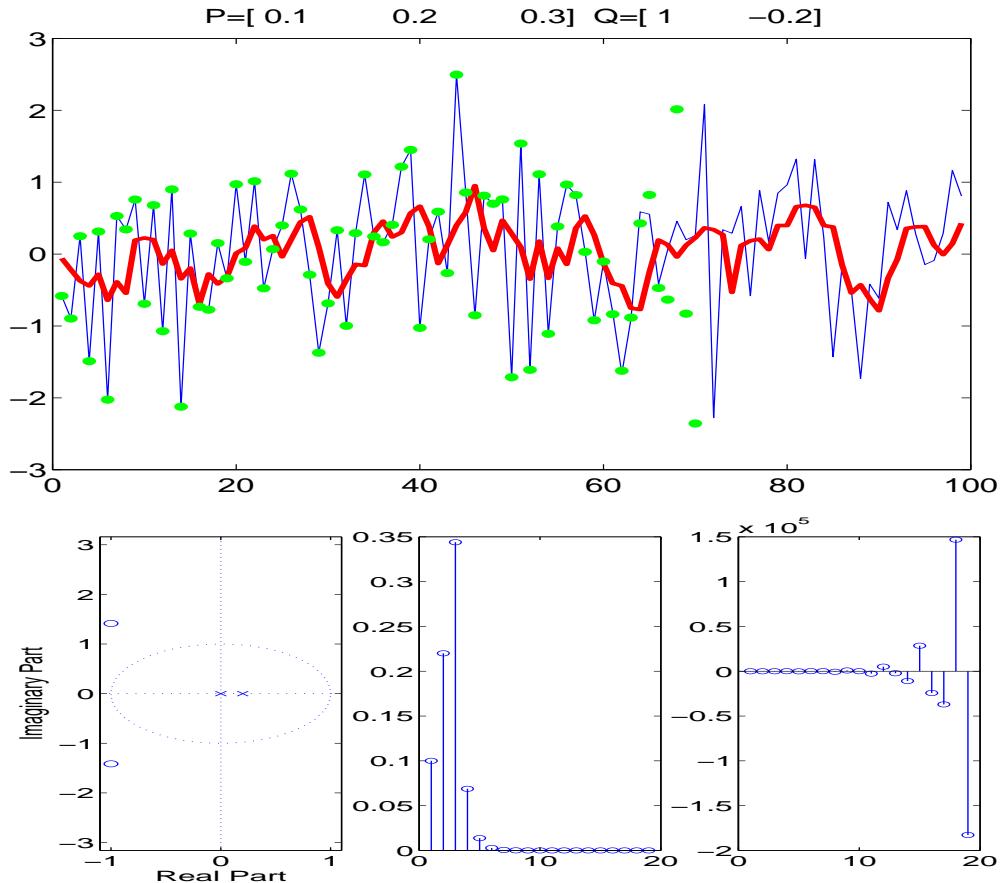


Figure D.1: Numerical illustration of the filter $F = \frac{0.1+0.2B+0.3B^2}{1-0.2B}$. Top: a random input sequence X (thin line), the corresponding output $Y = FX$ (thick line), obtained by the matlab command `Y=filter([0.1 0.2 0.3],[1 -0.2],X)` and the reconstructed input $F^{-1}Y$ obtained by `filter([1 -0.2],[0.1 0.2 0.3],Y)` (small disks). Bottom left: poles (x) and zeroes (o) of F , obtained by `zplane([0.1 0.2 0.3],[1 -0.2])`. The filter F is stable (poles within unit disk) but F^{-1} is not (at least one zero outside the unit disk). Bottom middle and right: impulse response of F (`h=filter([0.1 0.2 0.3],[1 -0.2],D)`, where D is a dirac sequence) and F^{-1} (`h=filter([1 -0.2],[0.1 0.2 0.3],D)`). Reconstruction of X as $F^{-1}Y$ fails for $t \geq 60$; this is a symptom of F^{-1} being unstable.

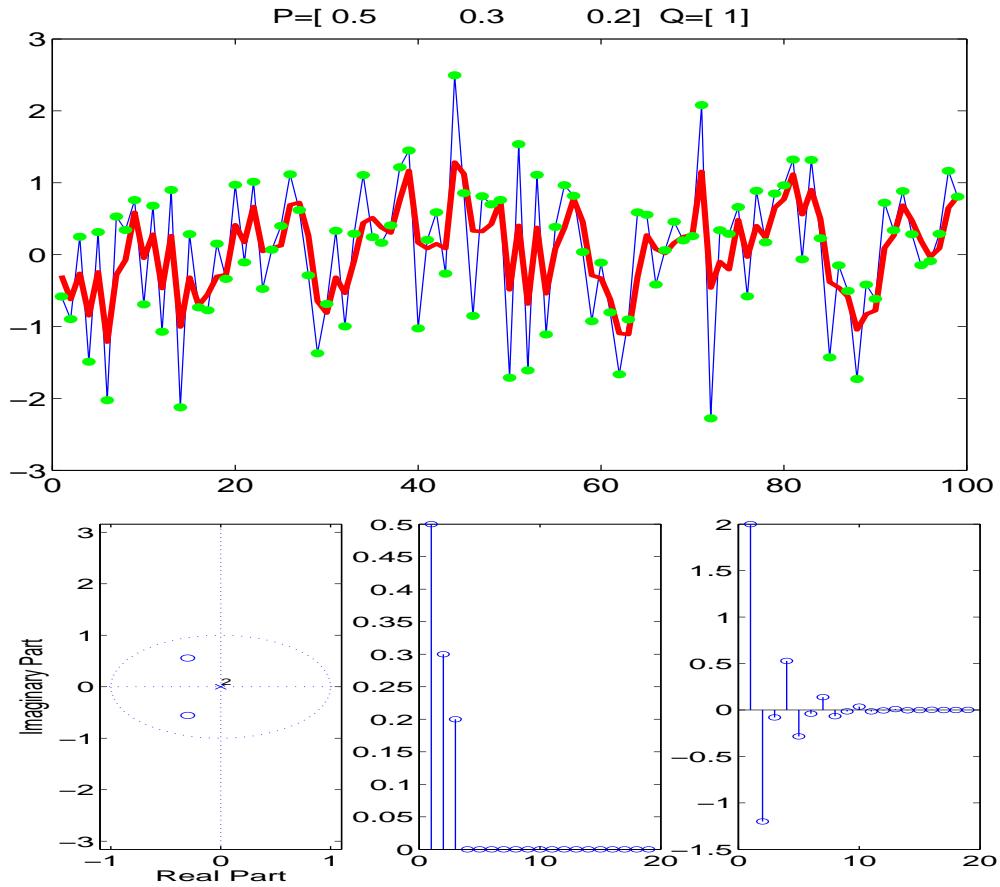


Figure D.2: Numerical illustration of the filter $G = G = 0.5 + 0.3B + 0.2B^2$. Top: a random input sequence X (thin line), the corresponding output $Z = GX$ (thick line) and the reconstructed input $G^{-1}Z$ (small disks). Bottom left: poles (x) and zeroes (o) of G . Depending on the conventions, the origin may or may not be considered as a pole. With our conversion, there is no pole but the software used shows a pole of multiplicity 2 at 0. The filter G and its inverse are stable (poles and zeroes are within the unit disk). Bottom middle and right: impulse response of G and G^{-1} . Reconstruction works perfectly.

The poles are the zeroes of $1 - 0.2z^{-1}$, which are the same as the zeroes of $z - 0.2$ (i.e $z = 0.2$).

The poles lie inside the unit disk, so the filter is stable. Its impulse response quickly decays to 0. The filter is invertible but the inverse is not stable as the zeroes are not all inside the unit disk. The impulse response of the inverse filter does not decay. We also compute $F^{-1}(Y)$ which should, in theory, be equal to X (small disks); however, the inverse filter is not stable and can be difficult to apply in practice; we see indeed that rounding errors become significant for $t \geq 60$.

If we consider instead $G = 0.5 + 0.3B + 0.2B^2$ then both the filter and its inverse are stable, and there are no numerical errors in the reconstruction (Figure D.2).

D.4 PREDICTIONS

We use filter to model time series and perform predictions. Many formulas in Chapter 5 are based on the following result.

D.4.1 CONDITIONAL DISTRIBUTION LEMMA

LEMMA D.1. *Let (X_1, X_2) , (Y_1, Y_2) be two random vectors, both with values in the space $\mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$, and such that*

$$\begin{aligned} Y_1 &= F_1 X_1 \\ Y_2 &= F_{21} X_1 + F_{22} X_2 \end{aligned}$$

where F_1, F_{21}, F_{22} are non random linear operators and F_1 is invertible.

Let X'_2 be a random sample drawn from the conditional distribution of X_2 given that $X_1 = x_1$ and

$$\begin{aligned} y_1 &= F_1 x_1 \\ Y'_2 &= F_{21} x_1 + F_{22} X'_2 \end{aligned}$$

The law of Y'_2 is the conditional distribution of Y_2 given that $Y_1 = y_1$.

D.4.2 PREDICTIONS

Let X_t, Y_t be two real valued random sequences (not necessarily iid), defined for $t \geq 1$. Assume that $Y = FX$ where F is an invertible filter with impulse response $h_0, h_1, h_2 \dots$ and AR(∞) representation $c_0, c_1, c_2 \dots$. The following theorem says that making a prediction for X is equivalent to making a prediction for Y . It is a direct consequence of Lemma D.1.

THEOREM D.1 (Conditional Distribution of Futures). *Assume that $(y_1, \dots, y_t)^T = F(x_1, \dots, x_t)^T$ and let $\ell \geq 1$.*

Assume that $(X'_{t+1}, \dots, X'_{t+\ell})$ is a random sample drawn from the conditional distribution of $(X_{t+1}, \dots, X_{t+\ell})$ given that $X_1 = x_1, \dots, X_t = x_t$. Let

$$(y_1, \dots, y_t, Y'_{t+1}, \dots, Y'_{t+\ell})^T = F(x_1, \dots, x_t, X'_{t+1}, \dots, X'_{t+\ell})^T$$

then $(Y'_{t+1}, \dots, Y'_{t+\ell})$ is distributed according to the conditional distribution of $(Y_{t+1}, \dots, Y_{t+\ell})$ given that $Y_1 = y_1, \dots, Y_t = y_t$.

We can derive explicit formulae for point predictions.

COROLLARY D.1 (Point Prediction). *Define the ℓ -point-ahead predictions by*

$$\begin{aligned}\hat{X}_t(\ell) &= \mathbb{E}(X_{t+\ell} | X_1 = x_1, \dots, X_t = x_t) \\ \hat{Y}_t(\ell) &= \mathbb{E}(Y_{t+\ell} | Y_1 = y_1, \dots, Y_t = y_t)\end{aligned}$$

then

$$(y_1, \dots, y_t, \hat{Y}_t(1), \dots, \hat{Y}_t(\ell))^T = F(x_1, \dots, x_t, \hat{X}_t(1), \dots, \hat{X}_t(\ell))^T \quad (\text{D.30})$$

and in particular

$$\hat{Y}_t(\ell) = h_0 \hat{X}_t(\ell) + h_1 \hat{X}_t(\ell-1) + \dots + h_{\ell-1} \hat{X}_t(1) + h_{\ell-1} x_t + \dots + h_{t-1} x_1 \quad (\text{D.31})$$

and

$$\hat{Y}_t(\ell) = c_0 \hat{X}_t(\ell) + c_1 \hat{Y}_t(\ell-1) + \dots + c_{\ell-1} \hat{Y}_t(1) + c_\ell y_t + \dots + c_{t-1} y_1 \quad (\text{D.32})$$

In the frequent case where X_t is assumed to be iid, we can deduce more explicit results for the point prediction and mean square prediction errors:

COROLLARY D.2. *Assume in addition that X_t is iid with mean $\mu = \mathbb{E}(X_t)$ and variance $\sigma^2 = \text{var}X_t$. Then*

1. (Point Predictions)

$$(y_1, \dots, y_t, \hat{Y}_t(1), \dots, \hat{Y}_t(\ell))^T = F(x_1, \dots, x_t, \mu, \dots, \mu)^T \quad (\text{D.33})$$

and in particular

$$\hat{Y}_t(\ell) = (h_0 + h_1 + \dots + h_{\ell-1})\mu + h_{\ell-1} x_t + \dots + h_{t-1} x_1 \quad (\text{D.34})$$

and

$$\hat{Y}_t(\ell) = c_0 \mu + c_1 \hat{Y}_t(\ell-1) + \dots + c_{\ell-1} \hat{Y}_t(1) + c_\ell y_t + \dots + c_{t-1} y_1 \quad (\text{D.35})$$

2. (Mean Square Prediction Error) Define

$$\begin{aligned}MSE_t^2(\ell) &\stackrel{\text{def}}{=} \text{var}(Y_{t+\ell} | Y_1 = y_1, \dots, Y_t = y_t) \\ &= \mathbb{E}((Y_{t+\ell} - \hat{Y}_t(\ell))^2 | Y_1 = y_1, \dots, Y_t = y_t)\end{aligned}$$

then

$$MSE_t^2(\ell) = \sigma^2 (h_0^2 + \dots + h_{\ell-1}^2) \quad (\text{D.36})$$

COROLLARY D.3 (Innovation Formula). *For $t \geq 2$:*

$$Y_t - \hat{Y}_{t-1}(1) = h_0(X_t - \mu) \quad (\text{D.37})$$

This is called innovation formula as it can be used to relate X_t (the “innovation”) to the prediction error.

D.5 LOG LIKELIHOOD OF INNOVATION

Let X_t, Y_t be two real valued random sequences (not necessarily iid), defined for $t \geq 1$. Assume that $Y = FX$ where F is an invertible filter with impulse response h_0, h_1, h_2, \dots . Also assume that for any n , the random vector (X_1, \dots, X_n) has a PDF $f_{\vec{X}_n}(x_1, \dots, x_n)$.

THEOREM D.2. *Assume that the impulse response of F is such that $h_0 = 1$. Then for all n the random vector (Y_1, \dots, Y_n) has a PDF equal to*

$$f_{\vec{Y}_n}(y_1, \dots, y_n) = f_{\vec{X}_n}(x_1, \dots, x_n)$$

with $(y_1, \dots, y_n)^T = F(x_1, \dots, x_n)^T$

Theorem D.2 can be used for estimation in the context of ARMA models, where X_t is the non observed innovation, assumed to be iid. The theorem says that the log-likelihood of the model is the same as if we had observed the innovation; estimation methods for iid sequences can then be applied, as in Example 2.5.

D.6 MATLAB COMMANDS

filter $Y = \text{filter}([Q_0 Q_1 \dots Q_q], [P_0 P_1 \dots P_p], X)$ with $P_0 \neq 0$ applies the filter

$$\frac{Q_0 + Q_1 B + \dots + Q_q B^q}{P_0 + P_1 B + \dots + P_p B^p} \quad (\text{D.38})$$

to the input sequence X and produces an output sequence Y of same length as X .

poles and zeroes can be obtained with `zplane`

de-seasonalizing The de-seasonalizing filter with period s is $R_s = \sum_{i=1}^{s-1} B^i$; $X = R_s Y$ can be obtained using

```
R = ones(1,s)
X = filter(R,1,Y)
```

differencing filter $X = \Delta_s Y$ can be obtained by

```
X = filter([1,0,...,0,-1],[1],Y)
```

where -1 is at position $s+1$. The inverse filter is obtained by exchanging the first two arguments:

```
Y = filter([1],[1,0,...,0,-1],X)
```

and the terms h_0, h_1, \dots, h_ℓ of the impulse response of Δ_s^{-1} are obtained by the command:

```
h = filter([1],[1,0,...,0,-1],[1,0,...,0])
```

where the last vector has ℓ zeroes.

The command `Y=diff(X,s)` also applies the differencing filter Δ_s to X but it removes the first s entries instead of setting them to 0 as `filter` does

impulse response `impz([P0 P1 ... Pp], [Q0 Q1 ... Qq], n)` gives the first n terms of the impulse response of the filter in Eq.(D.38). It is equivalent to using `filter([P0 P1 ... Pp], [Q0 Q1 ... Qq], deltan)` with `deltan` equal to the sequence `[1 0 ... 0]` (with $n-1$ zeroes).

parameter estimation of an ARMA model can be done with direct application of Theorem 5.2 and `lsqnonlin` for the solution of the non linear optimization problem. For simple ARMA models, it can be done in one step with `armax`.

convolution `c=conv(a,b)` computes the sequence of length $\text{length}(a) + \text{length}(b) - 1$ such that $c_k = \sum_i a_i b_{k-i}$, where the sum is for i such that a_i and b_{k-i} are defined. The command `Y = filter(P1,Q1,filter(P2,Q2,X))` is equivalent to

```
P = conv(P1,P2)
Q = conv(Q1,Q2)
X = filter(P,Q, X)
```

simulation of an ARMA process as defined in Definition 5.1 can be done with

```
e = sigma * randn(n,1)
x = mu + filter(A,C,e)
```

D.7 PROOFS

LEMMA D.1 The characteristic function of Y'_2 is

$$\begin{aligned} \phi_{Y'_2}(\omega_2) &= \mathbb{E}\left(e^{-j(\langle\omega_2, F_{21}x_1\rangle + \langle\omega_2, F_{22}X'_2\rangle)}\right) = e^{-j\langle\omega_2, F_{21}x_1\rangle} \mathbb{E}\left(e^{-j\langle\omega_2, F_{22}X'_2\rangle}\right) \\ &= e^{-j\langle\omega_2, F_{21}x_1\rangle} \mathbb{E}\left(e^{-j\langle\omega_2, F_{22}X_2\rangle} | X_1 = x_1\right) \\ &:= e^{-j\langle\omega_2, F_{21}x_1\rangle} h(x_1) := f(x_1) \end{aligned} \quad (\text{D.39})$$

where $\langle \cdot, \cdot \rangle$ is the inner product. Now let $g(y_1) := f(F_1^{-1}y_1)$. We want to show that

$$g(y_1) = \mathbb{E}\left(e^{-j\langle\omega_2, Y_2\rangle} | Y_1 = y_1\right) \quad (\text{D.40})$$

By definition of a conditional probability, this is equivalent to showing that for any $\omega_1 \in \mathbb{R}^{n_1}$:

$$\mathbb{E}\left(e^{-j\langle\omega_1, Y_1\rangle} g(Y_1)\right) = \mathbb{E}\left(e^{-j\langle\omega_1, Y_1\rangle} e^{-j\langle\omega_2, Y_2\rangle}\right) \quad (\text{D.41})$$

Now by the definition of $g()$

$$\begin{aligned} \mathbb{E}\left(e^{-j\langle\omega_1, Y_1\rangle} g(Y_1)\right) &= \mathbb{E}\left(e^{-j\langle\omega_1, Y_1\rangle} e^{-j\langle\omega_2, F_{21}F_1^{-1}Y_1\rangle} h(F_1^{-1}Y_1)\right) \\ &= \mathbb{E}\left(e^{-j\langle\omega_1, F_1X_1\rangle} e^{-j\langle\omega_2, F_{21}X_1\rangle} h(X_1)\right) \\ &= \mathbb{E}\left(e^{-j\langle\omega_1, F_1X_1\rangle} e^{-j\langle\omega_2, F_{21}X_1\rangle} e^{-j\langle\omega_2, F_{22}X_2\rangle}\right) \end{aligned}$$

where the last equality is by the definition of $h()$ as a conditional expectation in Eq.(D.39). This shows Eq.(D.41) as desired.

Note that the proof is simpler if X_1, X_2 has a density, but this may not always hold, even for gaussian processes.

THEOREM D.2 The random vector $\vec{Y}_n = (Y_1, \dots, Y_n)^T$ is derived from the random vector $\vec{X}_n = (X_1, \dots, X_n)^T$ by $\vec{Y}_n = H_n \vec{X}_n$ where H_n is the matrix in Eq.(D.6), with $h_0 = 1$. By the formula of change of variable, we have

$$f_{\vec{X}_n}(x_1, \dots, x_n) = |\det(H_n)| f_{\vec{Y}_n}(y)$$

and $\det(H_n) = 1$.

Bibliography

- [1] The ns-3 network simulator. <http://www.nsnam.org/>.
- [2] I. Adan, J. Visschers, and J. Wessels. Sum of product forms solutions to MSCCC queues with job type dependent processing times. *Memorandum COSOR 98-19*, 1998.
- [3] F. Baccelli and P. Brémaud. *Elements of queueing theory: Palm Martingale calculus and stochastic recurrences*. Springer Verlag, 2003.
- [4] François Baccelli and Pierre Brémaud. *Palm Probabilities and Stationary Queues*. Springer LNS, 1987.
- [5] O. Bakr and I. Keidar. Evaluating the running time of a communication round over the Internet. In *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 243–252. ACM New York, NY, USA, 2002.
- [6] S. Balsamo. Product form queueing networks. *Lecture Notes in Computer Science*, pages 377–402, 2000.
- [7] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski. Modeling Internet backbone traffic at the flow level. *IEEE Transactions on Signal processing*, 51(8):2111–2124, 2003.
- [8] A.D. Barbour. Networks of queues and the method of stages. *Advances in Applied Probability*, 8(3):584–591, 1976.
- [9] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. *SIGMETRICS Perform. Eval. Rev.*, 26(1):151–160, 1998.
- [10] F. Baskett, K.M. Chandy, R.R. Muntz, and F.G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM (JACM)*, 22(2):260, 1975.
- [11] SA Berezner, CF Kriel, and AE Krzesinski. Quasi-reversible multiclass queues with order independent departure rates. *Queueing Systems*, 19(4):345–359, 1995.
- [12] C.R. Blyth and H.A. Still. Binomial confidence intervals. *Journal of the American Statistical Association*, pages 108–116, 1983.
- [13] T. Bonald and A. Proutiere. Insensitive bandwidth sharing in data networks. *Queueing systems*, 44(1):69–100, 2003.
- [14] T. Bonald and J. Virtamo. Calculating the flow level performance of balanced fairness in tree networks. *Performance Evaluation*, 58(1):1–14, 2004.

- [15] G. E. P. Box and G. M. Jenkins. *Time Series Analysis, Forecasting and Control*. Holden-Day, San Francisco, 1970.
- [16] M. Bramson. Instability of FIFO queueing networks with quick service times. *The Annals of Applied Probability*, 4(3):693–718, 1994.
- [17] P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation and Queues*. Springer, 1999.
- [18] P.J. Brockwell and R.A. Davis. *Introduction to time series and forecasting*. Springer Verlag, 2002.
- [19] P.J. Brockwell and R.A. Davis. *Introduction to Time Series and Forecasting, second edition*. Springer-Verlag, New York, 2002.
- [20] J.P. Buzen. Computational algorithms for closed queueing networks with exponential servers. 1973.
- [21] Erhan Çinlar. *Introduction to Stochastic Processes*. Prentice Hall, 1975.
- [22] K. Mani Chandy and Charles H. Sauer. Computational algorithms for product form queueing networks. *Commun. ACM*, 23(10):573–583, 1980.
- [23] C.S. Chang. *Performance Guarantees in Communication Networks*. Springer-Verlag, New York, 2000.
- [24] X. Chao, M. Miyazawa, and M. Pinedo. *Queueing networks: customers, signals and product form solutions*. Wiley, 1999.
- [25] JB Chen, E. Yasuhiro, and C. Kee. The Measured Performance of Personal Computer Operating Systems,” 15 thACM SOSP. *Colorado, United States: Copper Mountain*, pages 169–173, 1995.
- [26] G. Chiola, MA Marsan, and G. Balbo. Product-form solution techniques for the performance analysis of multiple-bus multiprocessor systems with nonuniform memory references. *IEEE Transactions on Computers*, 37(5):532–540, 1988.
- [27] AE Conway and ND Georganas. RECAL - a new efficient algorithm for the exact analysis of multiple-chain closed queuing networks. *Journal of the ACM (JACM)*, 33(4):768–791, 1986.
- [28] A.E. Conway, E.S. Silva, and S.S. Lavenberg. Mean value analysis by chain of product form queueing networks. *IEEE Transactions on Computers*, 38(3):432–442, 1989.
- [29] Verizon Corporation. Verizon NEBS(TM) Compliance: Energy Efficiency Requirements for Telecommunications Equipment. Technical Report VZ.TPR.9205, September 2008.
- [30] T.M. Cover and JA Thomas. Elements of Information Theory, 1991.
- [31] M.E. Crovella and M.S. Taqqu. Estimating the heavy tail index from scaling properties. *Methodology and computing in applied probability*, 1(1):55–79, 1999.
- [32] A.C. Davison. *Statistical Models*. Cambridge University Press, 2003.

- [33] A.C. Davison and DV Hinkley. *Bootstrap methods and their application*. Cambridge Univ Pr, 1997.
- [34] P.J. Denning and J.P. Buzen. The operational analysis of queueing network models. *ACM Computing Surveys (CSUR)*, 10(3):225–261, 1978.
- [35] M. El-Taha and Shaler Jr. Stidham. *Sample-path analysis of queueing systems*. Kluwer Academic Pub, 1998.
- [36] E. Gelenbe. Product-form queueing networks with negative and positive customers. *Journal of Applied Probability*, pages 656–663, 1991.
- [37] W.J. Gordon and G.F. Newell. Closed queueing systems with exponential servers. *Operations Research*, 15(2):254–265, 1967.
- [38] G. Grimmett and D. Stirzaker. *Probability and random processes*. Oxford University Press, USA, 2001.
- [39] M. Grossglauser and J.C. Bolot. On the relevance of long-range dependence in network traffic. *IEEE/ACM Transactions on Networking (TON)*, 7(5):629–640, 1999.
- [40] B. Hechenleitner and K. Entacher. On shortcomings of the ns-2 random number generator. *vectors*, 1000:1.
- [41] C. C. Holt. Forecasting Seasonal and Trends by Exponentially Weighted Moving Averages. Carnegie Institute of Technology, Pittsburgh, Pennsylvania, 1957.
- [42] J.R. Jackson. Jobshop-like queueing systems. *Management science*, 50(12):1796–1802, 1963.
- [43] N.L. Johnson, S. Kotz, and A.W. Kemp. *Univariate discrete distributions*. Wiley-Interscience, 2005.
- [44] F. P. Kelly. *Reversibility and Stochastic Networks*. Wiley, 1979.
- [45] F.P. Kelly. Models for a self-managed Internet. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, pages 2335–2348, 2000.
- [46] Leonard Kleinrock. *Queueing Systems Volume I: Theory*, volume 1. John-Wiley & Sons, 1975.
- [47] Leonard Kleinrock. *Queueing Systems Volume II: Computer Applications*, volume 2. John-Wiley & Sons, 1976.
- [48] Anne B. Koehler, Ralph D. Snyder, and Ord J. Keith. Forecasting models and prediction intervals for the multiplicative holt-winters method. *International Journal of Forecasting*, 17:269–286, April-June 2001.
- [49] M. A. Law and W.D. Kelton. Simulation modeling and analysis. *Edited by Averill M. Law, W. David Kelton*, 2000.
- [50] J.-Y. Le Boudec. Rate adaptation, congestion control and fairness: a tutorial. http://ica1www.epfl.ch/PS_files/LEB3132.pdf.

- [51] J.-Y. Le Boudec. A BCMP extension to multiserver stations with concurrent classes of customers. *ACM SIGMETRICS Performance Evaluation Review*, 14(1):78–91, 1986.
- [52] J.-Y. Le Boudec. Steady-state probabilities of the PH/PH/1 queue. *Queueing Systems*, 3(1):73–87, 1988.
- [53] J.-Y. Le Boudec. The MULTIBUS algorithm. *Performance Evaluation*, 8(1):1–18, 1988.
- [54] J.-Y. Le Boudec. Understanding the simulation of mobility models with palm calculus. *Performance Evaluation*, 64(2):126–147, 2007.
- [55] J.-Y. Le Boudec and P. Thiran. *Network Calculus*. Springer Verlag Lecture Notes in Computer Science volume 2050 (available online at <http://infoscience.epfl.ch/record/282>), July 2001.
- [56] Jean-Yves Le Boudec and Milan Vojnovic. The Random Trip Model: Stability, Stationary Regime, and Perfect Simulation The Random Trip Model: Stability, Stationary Regime, and Perfect Simulation. *IEEE/ACM Transactions on Networking*, 14(6):1153–1166, 2006.
- [57] JY Le Boudec. Interinput and Interoutput Time Distribution in Classical Product-Form Networks. *IEEE Transactions on Software Engineering*, pages 756–759, 1987.
- [58] LM Le Ny. Étude analytique de reseaux de files d’attente multiclassées à routage variable. *RAIRO Recherche Opérationnelle/Oper. Res*, 14:331–347, 1980.
- [59] P. L’Ecuyer. Random number generation. *Handbook of simulation: principles, methodology, advances, applications, and practice*, 1998.
- [60] P. L’Ecuyer. Software for uniform random number generation: Distinguishing the good and the bad. In *Proceedings of the 33rd conference on Winter simulation*, pages 95–105. IEEE Computer Society Washington, DC, USA, 2001.
- [61] E.L. Lehmann. On likelihood ratio tests. In *IMS Lecture Notes- 2nd Lehmann Symposium*, volume 49, pages 1–8, 2006.
- [62] E.L. Lehmann and J.P. Romano. *Testing statistical hypotheses*. Springer, 2005.
- [63] Tom Leighton. Improving performance on the internet. *Commun. ACM*, 52(2):44–51, 2009.
- [64] WE Leland, MS Taqqu, W. Willinger, DV Wilson, and M. Bellcore. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on networking*, 2(1):1–15, 1994.
- [65] Gary Malinas and John Bigelow. Simpson’s paradox. Stanford Encyclopedia of Philosophy, online.
- [66] Sam Manthorpe and Jean-Yves Le Boudec. A comparison of ABR and UBR to support TCP traffic. *Networking and Information Systems Journal*, 2(5-6):764–793, 1999.
- [67] M. Matsumoto and T. Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 8(1):3–30, 1998.

- [68] R. Merz and J.-Y. Le Boudec. Conditional bit error rate for an Impulse Radio UWB channel with interfering users. In *2005 IEEE International Conference on Ultra-Wideband, 2005. ICU 2005*, pages 130–135, 2005.
- [69] M. Miyazawa. The derivation of invariance relations in complex queueing systems with stationary inputs. *Advances in Applied Probability*, 15(4):874–885, 1983.
- [70] Masakiyo Miyazawa. Rate conservation laws: A survey. *Queueing Syst.*, 15(1-4):1–58, 1994.
- [71] P. Nain. Basic elements of queueing theory: Application to the Modelling of Computer Systems. *Course notes*.
- [72] M.F. Neuts. Stationary waiting-time distributions in the GI/PH/1 queue. *Journal of Applied Probability*, 18(4):901–912, 1981.
- [73] J.P. Nolan. Stable distributions. *Math/Stat Department, American University*, 2009.
- [74] I. Norros. A storage model with self-similar input. *Queueing systems*, 16(3):387–396, 1994.
- [75] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck. *Discrete-time signal processing* (2nd ed.). Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999.
- [76] A. Papoulis, S.U. Pillai, Papoulis A, and Pillai SU. *Probability, random variables, and stochastic processes*. McGraw-Hill New York, 1965.
- [77] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.
- [78] J. Pellaumail. Formule du produit et décomposition de réseaux de files d’attente. *Ann. Inst. H. Poincaré Sect. B (N.S.)*, 15(3):261–286, 1979.
- [79] M.D. Perlman and L. Wu. The emperor’s new tests. *Statistical Science*, pages 355–369, 1999.
- [80] B. Pittel. Closed exponential networks of queues with saturation: The Jackson-type stationary distribution and its asymptotic analysis. *Mathematics of Operations Research*, 4(4):357–378, 1979.
- [81] H.V. Poor. *An introduction to signal detection and estimation*. Springer, 1994.
- [82] K.R. Popper. *Logik der Forschung: Zur Erkenntnistheorie der modernen Naturwissenschaft*. J. Springer, 1935.
- [83] P. Prandoni and M Vetterli. *Signal Processing for Communications*. EPFL Press, Communication and Information Sciences, 2008.
- [84] M. Reiser. Mean-Value Analysis and Convolution Method for Queue-Dependent Servers in Closed Queueing Networks. *Performance Evaluation*, 1(1):7–18, 1981.
- [85] M. Reiser and H. Kobayashi. Queuing networks with multiple closed chains: theory and computational algorithms. *IBM Journal of Research and Development*, 19(3):283–294, 1975.

- [86] M. Reiser and SS Lavenberg. Mean-value analysis of closed multichain queuing networks. *Journal of the ACM (JACM)*, 27(2):313–322, 1980.
- [87] H. Rinne. *The Weibull Distribution: A Handbook*. Chapman & Hall/CRC, 2008.
- [88] Philippe Robert. *Stochastic Networks and Queues*. Stochastic Modelling and Applied Probability Series. Springer-Verlag, 2003.
- [89] S. A. Roberts. A General Class of Holt-Winters Type Forecasting Models. *Management Science*, 28(7):808–820, July 1982.
- [90] S.M. Ross. *Simulation*. Academic Press, 2006.
- [91] Walter Rudin. *Real and Complex Analysis*. McGraw-Hill Series in Mathematics, 1987.
- [92] M. Sakata, S. Noguchi, and J. Oizumi. Analysis of a processor shared queueing model for time sharing systems. In *Proc. 2nd Hawaii International Conference on System Sciences*, volume 625628, 1969.
- [93] G. Samorodnitsky and M.S. Taqqu. *Stable non-Gaussian random processes: stochastic models with infinite variance*. Chapman & Hall/CRC, 1994.
- [94] R. Serfozo. *Introduction to stochastic networks*. Springer Verlag, 1999.
- [95] R. Serfozo. *Basics of Applied Stochastic Processes*. Springer Verlag, 2009.
- [96] A. Shaikh, J. Rexford, and K.G. Shin. Load-sensitive routing of long-lived IP flows. In *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, pages 215–226. ACM New York, NY, USA, 1999.
- [97] Robert H. Shumway and David S Stoffer. *Time Series Analysis and Its Applications With R Examples*. Springer-Verlag, New York, 2006.
- [98] PJ Smith, M. Shafi, and H. Gao. Quick simulation: a review of importance sampling techniques in communications systems. *IEEE Journal on Selected Areas in Communications*, 15(4):597–613, 1997.
- [99] Steve Souders. High-performance web sites. *Commun. ACM*, 51(12):36–41, 2008.
- [100] JL Van den Berg and OJ Boxma. The M/G/1 queue with processor sharing and its relation to a feedback queue. *Queueing Systems*, 9(4):365–401, 1991.
- [101] N.M. Van Dijk. *Queueing networks and product forms: a systems approach*. John Wiley & Sons, 1993.
- [102] Steve Verrill. Confidence Bounds for Normal and Lognormal Distribution Coefficients of Variation. Technical Report Research Paper 609, USDA Forest Products Laboratory, Madison, Wisconsin, 2003.
- [103] J. Walrand. *An introduction to queueing networks*. Prentice Hall, 1988.
- [104] Richard Weber. C11: Statistics. online lecture notes, <http://www.statslab.cam.ac.uk>.

- [105] Richard Weber. Time series. online lecture notes, <http://www.statslab.cam.ac.uk>.
- [106] Wikipedia. Harmonic mean. <http://en.wikipedia.org>.
- [107] P. R. Winters. Forecasting sales by exponentially weighted moving averages. *Management Science*, 6(60):324–342, 1960.
- [108] Bernard Ycart. *Modèles et algorithmes markoviens*, volume 39. Springer Verlag, 2002.